Volume 33 Number 4 November 2009 ISSN 0350-5596

Informatica

An International Journal of Computing and Informatics

Special Issue:

System Modeling and Transformation Principles

Guest Editor:

Anna Derezińska

EDITORIAL BOARDS, PUBLISHING COUNCIL

Informatica is a journal primarily covering the European computer science and informatics community; scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor from the Editorial Board can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the list of referees. Each paper bears the name of the editor who appointed the referees. Each editor can propose new members for the Editorial Board or referees. Editors and referees inactive for a longer period can be automatically replaced. Changes in the Editorial Board are confirmed by the Executive Editors.

The coordination necessary is made through the Executive Editors who examine the reviews, sort the accepted articles and maintain appropriate international distribution. The Executive Board is appointed by the Society Informatika. Informatica is partially supported by the Slovenian Ministry of Higher Education, Science and Technology.

Each author is guaranteed to receive the reviews of his article. When accepted, publication in Informatica is guaranteed in less than one year after the Executive Editors receive the corrected version of the article.

Editor's Introduction to the Special Issue

System Modeling and Transformation Principles

Development of information systems results in high demands on efficiency of software engineering processes, as well as correctness and good performance of the designed and implemented products. Therefore, different solutions dealing with system modeling and precise specification of transformation principles of various artifacts are important research subjects. In this mini-special issue, three contributions on selected areas on principles of information technology are presented.

The first paper, entitled "Simulation and Performance Analysis of Distributed Internet Systems Using TCPNs" was written by S. Samolej and T. Rak. It concerns a method of modeling and analysis of distributed Internet systems. The method is based on the concept of Timed Coloured Petri Nets (TCPN). A distributed system described in terms of classical Queueing Theory is mapped onto TCPN structure, then executed and analyzed. The paper provides the results of simulation experiments and comparison of some results of experiments on real Internet environment.

The second paper is entitled "Transformation of XML data into XML normal form" and was proposed by T. Pankowski and T. Pilka. The paper tackles the problem of a "tradeoff" between dependencies and redundancy while normalizing relational schemes in XML schemes. The language to express functional dependencies and algorithms is proposed. The authors present algorithms that are used to define procedure to transform a class of XML instances into XML normal form. The transformation is processed by XQuery programs which are automatically generated for a given XML schema together with a class of XFDs defined over this schema.

In the third paper, "Realization of UML class and state machine models in the C# Code Generation and Execution Framework", A. Derezinska and R. Pilitowski present selected issues of the "Framework for eXecutable UML (FXU)". FXU is the tool designed to verify UML class and state machine models for correctness, to generate corresponding program code using C# as a target language, and to execute this code with the help of the runtime library. One of particular qualities of the tool is exploitation of C# constructs to create concise representation of state machines, including all complex concepts of UML behavioral state machines. Additionally, the correctness rules for UML models are presented, aimed at transformation of class and state machine models into C# applications.

This special issue has extended versions of the best papers from the *International Conference on Principles of Information Technology and Applications* PITA'08 (http://www.pita.imcsit.org). It was one of events organized within *International Multiconference on Computer Science and Information Technology* IMCSIT held in Wisla, Poland in October 20-22, 2008. The authors of four papers were asked to thoroughly review their papers and extend them for journal publication. The rewritten papers were reviewed once more, and finally three accepted for this special issue.

The guest editor wishes to thank Professor Matjaz Gams (Editor-in-Chief of Informatica) for providing the opportunity to edit this special issue on System Modeling and Transformation Principles. Finally, I would like to thank the authors of the papers for their contributions and all referees for their critical and valuable comments. Their efforts helped to ensure the high quality of the material presented here.

Anna Derezińska
Institute of Computer Science,
Warsaw University of Technology, Poland

Simulation and Performance Analysis of Distributed Internet Systems Using TCPNs

Slawomir Samolej and Tomasz Rak
Department of Computer and Control Engineering, Rzeszow University of Technology, Poland
E-mail: ssamolej, trak@prz-rzeszow.pl and http://ssamolej, trak.prz-rzeszow.pl

Keywords: distributed systems modeling, performance analysis, timed coloured Petri nets

Received: December 6, 2008

This paper presents a Timed Coloured Petri Nets based programming tool that supports modeling and performance analysis of distributed World Wide Web environments. A distributed Internet system model, initially described in compliance with Queueing Theory (QT) rules, is mapped onto the Timed Coloured Petri Net (TCPN) structure by means of queueing system templates. Then, it is executed and analyzed. The proposed distributed Internet systems modeling and design methodology has been applied for evaluation of several system architectures under different external loads.

Povzetek: Predstavljeno je orodje na osnovi Petri mrež za modeliranje spletnih okolij.

1 Introduction

One of modern Internet (or Web) systems development approaches assumes that the systems consist of a set of distributed nodes. Dedicated groups of nodes are organized in layers (clusters) conducting predefined services (e.g. WWW service or data base service) [2, 6, 8]. This approach makes it possible to easily scale the system. Additionally, a distributed structure of the system assures its higher dependability. Fig. 1 shows an example cluster–based Internet system structure. The Internet requests are generated by the clients. Then they are distributed by the load balancer among set of computers that constitute the front-end or WWW cluster. The front–end cluster offers a system interface and some procedures that optimize the load of the next system layer–the database servers cluster. In the standard scenario the client produces the request by e.g. filling out the formula on the web side. Then the request is converted into e.g. a SQL query and forwarded to the database layer. The result of the query is sent back to the front-end layer. Finally, the client receives the result of his request on the website.

Simultaneously, for a significant number of Internet applications some kind of soft real-time constraints are formulated. The applications should provide up-to-date data in set time frames [20]. Stock market or multimedia applications may be good examples of hardware/software systems that may have such timing requirements.

The appearing of new above mentioned development paradigms cause that searching for a new method of modeling and timing performance evaluation of distributed Internet systems seems to be an up-to-date research path.

One of intensively investigated branch of Internet systems software engineering is formal languages application for modeling and performance analysis. Amid suggested solutions there are: algebraic description [11], mapping through Queueing Nets (QNs) [8, 18], modeling using both Coloured Petri Nets (CPNs) [12] and Queueing Petri Nets (QPNs) [6, 7].

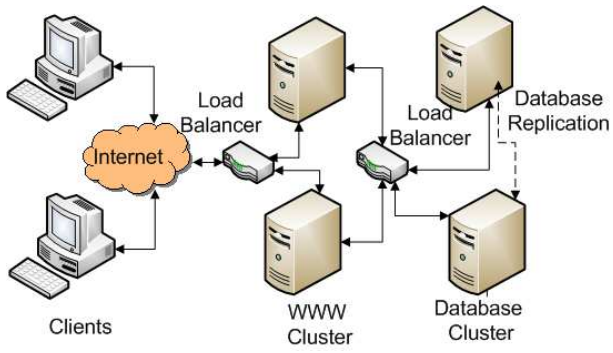


Figure 1: Example distributed cluster–based Internet system.

Our approach proposed in this paper[1] may be treated as extension of selected solutions summed up in [6, 7], where Queueing Petri Nets (QPNs) language [1] has been successively applied to the web–cluster modeling and performance evaluation. The final QPNs based model can be executed and used for modeled system performance prediction. In our solution we propose alternative Queueing Systems models defined as in [3] expressed into Timed Coloured Petri Nets (TCPNs) [4]. The models has been used as a background for developing a programming tool which is able to map timed behavior of queueing nets by means of simulation. Subsequently we developed our individual TCPNs–based method of modeling and analysis

[1] An earlier version of the manuscript was published in Proceedings of the International Multiconference on Computer Science and Information Technology, 2008, pp. 559–566.

of distributed Internet systems. The well known software toolkits as Design/CPN or CPN Tools can be naturally used for our models simulation and performance analysis [10, 19]. The preliminary version of our software tool was announced in [13], the more mature its description can be found in [16].

The remaining work is organized as follows. In section 2 we informally introduce basic concepts of TCPNs and then in section 3 we provide rules of mapping queueing systems into TCPNs. In the next section, we present a method of applying the TCPNs based queueing systems models (TCPNs templates) to distributed Internet system modeling. Section 5 focuses on results of simulation some detailed Internet system models while section 6 sums up the paper and includes our future research plans.

2 Hierarchical timed coloured Petri nets' basic concepts

As TCPNs is the main formal language exploited in the paper we decided to briefly introduce it. The introduction will have an informal form focusing only on the most important TCPNs features. The more thorough TCPNs informal introductions can be found in [9, 5]. The detailed TCPNs features and some applications are presented in [4].

Informally, a Timed Coloured Petri Net is a bipartite graph consisting of "place" nodes and "transition" nodes. The places, drawn as circles or ellipses, are used to represent conditions; the transitions, drawn as bars, are used to represent events.

The places can have some "tokens" associated with. Each token is equipped with an attached data value - the token "colour". The data value may be freely complex (e.g. integer number or record of data). For each place, a colour set is defined which characterizes acceptable colours of the token in the place. All declarations concerning the behavior of the net and colours of tokens are written in the CPN ML language. It is possible to define colours, variables, expressions and functions connected to the elements of the net. The distribution of tokens in the places is called marking. The initial marking determines the initial state of the net and is specified by "initialization expressions". The marking of each place is a multi-set over the colour set attached to the place (compare [4]).

Directed arcs (arrows) connect the places and transitions, with some arcs directed from the places to the transitions and other arcs directed from the transitions to the places. An arc directed from place to transition defines the place to be an input of the transition. Multiple inputs to a transition are indicated by multiple arcs from the input places to the transition. An output place is indicated by an arc from the transition to the place. Again, multiple outputs are represented by multiple arcs. The tokens can "use" the arcs to move from place to place. Moreover, "arc expressions" (the expressions that may be attached to arcs) decide on the token flow in the net. Arc expressions may denote the number and the colour set of flowing tokens, as well as may be functions that manipulate tokens and their colours.

The marking of a Petri net changes by the "occurrence" of transitions. Before the occurrence of a transition, the variables defined in the net are bounded to colours of corresponding types, which is called a binding. A pair `(t, b)` where `t` is a transition and `b` a binding for `t` is called a binding element. For each binding, it can be checked if the transition is enabled to fire in a current marking. An enabled transition may occur. The occurrence of a transition is an instantaneous event during which tokens are removed from each of transition's input places and tokens are deposited in its output places.

The transitions of the net may have "guards" attached to them. The guards are boolean expressions that may define additional constraints that must be fulfilled before the transition is enabled.

For the effective modeling TCPN enable to distribute parts of the net across multiple subnets. The ability to define the subnets enables to construct a large Hierarchical TCPN by composing a number of smaller nets. Hierarchical TCPNs offer two mechanisms for interconnecting TCPN structure on different layers: "substitution transitions" and "fusion places". A substitution transition is a transition that stands for a whole subnet of net structure. A fusion place is a place that has been equated with one or more other places, so that the fused places act as a single place with a single marking. Each hierarchical TCPN can be translated into behaviorally equivalent non-hierarchical TCPN, and vice versa. Thanks to the ability to handle additional information or data in a TCPN structure and thanks to introduction of the hierarchy, the nets manage to model complex systems in a consistent way.

To model a timing behavior of systems TCPNs posses the following features. There exist a global clock whose values represent the current model time. Tokens may carry a time value, also denoted as a time stamp. The time stamp represents the earliest model time at which the token can be used. Hence, to occur, a transition must be both colour enabled and ready. It means that the transition must fulfill the usual enabling rule and all the time stamps of the removed tokens must be less than or equal to the current model time. When a token is created, the time stamp is specified by an expression. This means it is possible to specify all kinds of delays (e.g. constant, interval, or probability distribution). Moreover, the delay may depend upon the binding of the transition that creates the token.

Fig. 2 includes an example HTCPN modeling a simple data distribution system. The declaration node at the top of the figure defines 4 token types (colours). `DATA` token type defines 3 possible enumerated values: A, B or C. `NUMBER` and `COUNTER` token types rename the basic integer type. `PACK` colour is a tuple including two elements: the number and the value of the data package distributed over the modeled system. Two variables: `p` and `n` are also defined in the declaration node. They may carry `PACK` and `COUNTER` data values, respectively.

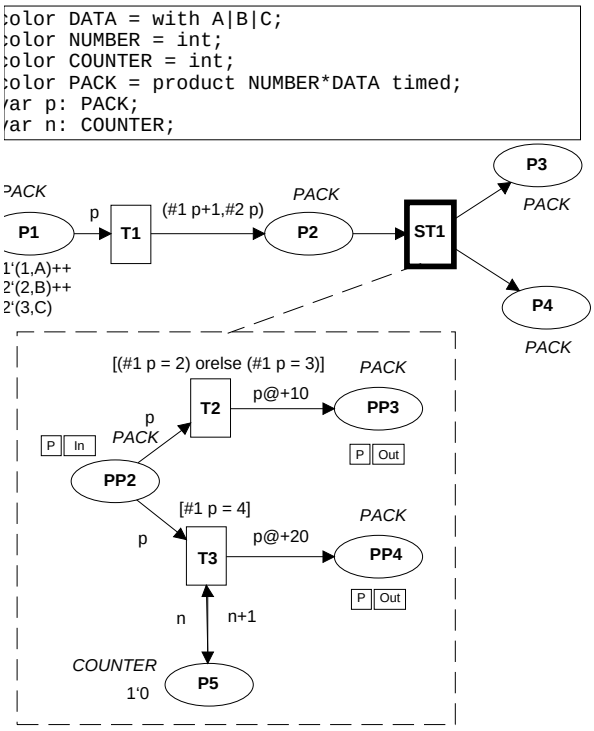


Figure 2: Example HTCP Net

For P1 and P5 places the initial markings are defined. For example, initially place P1 will include 1 token of (1,A) value, 2 tokens of (2,B) value and 2 tokens of (3,C) value. As P1 place has some tokens T1 transition may occur. During the occurrence of T1 transition arc inscription associated to the arc connecting T1 transition and P2 place modifies the currently transported token. It increments by 1 each first element of the PACK type tuple. Then the packets are distributed over the P3 and P4 places. The distribution procedure is "hidden" under the ST1 substitution transition. Guard function connected to T2 transition allows to "pass" only these data packets whose NUMBER field is 2 or 3. The remaining tokens may go through T3 transition. Moreover the firing of T3 transition increments a token value stored in P5 place. As a result P5 stores the number of data packets received by P5 place. Data packets that are transported over the T2 and T3 transitions acquire additionally some new time stamps. This may be interpreted that the event connected to T2 or T3 transition occurrence takes 10 or 20 time units, respectively.

The remaining HTCPNs structures presented in this paper use some similar techniques to manage the token distribution. Some arc inscriptions are defined as "CPN ML functions". The selected parameters of the modeled systems are stored as "values" not mentioned in the example.

3 Queueing system implementation

Queueing Net usually consists of a set of connected *queueing systems*. Each *queueing system* is described by an arrival process, a waiting room, and a service process. In the proposed programming tool, we worked out several TCPNs based *queueing system templates* (e.g. – /M/PS/∞, – /M/FIFO/∞) most frequently used to represent properties of distributed Internet system components. Each template represents a separate TCPN net (subpage) which may be included in the model of the system as a substitution transition (using hierarchical CP nets mechanisms [4]).

3.1 Queueing system mapping

Queueing system properties are mapped to the TCPNs net as follows. At a certain level of system description, a part of hardware/software is modeled as a TCPN, where some dedicated *substitution transitions* are understand as queueing systems (compare fig. 3). To have the queueing functionality running "under" selected transitions the mapping to adequate TCPNs subpages must be done. The corresponding subpages include the implementation of the adequate queueing system. In fig. 3 an example -/M/1/PS/∞ (exponential service times, single server, Processor Sharing service discipline and unlimited number of arrivals in the system; the queue's arrival process in our modeling approach is defined outside of queueing system model) queueing model definition path is presented.

Packets to be served by the example queueing system are delivered by port place INPUT_PACKS. Then, they are scheduled in a queue in PACK_QUEUE place. Every given time quantum (regulated by time multiset included in TIMERS place) the first element in the queue is selected to be served (execution of transition EXECUTE_PS). Then, it is placed at the end of the queue or directed to leave the system (execution of transition ADD_PS1 or REMOVE_PS respectively). Number of tokens in TIMERS place represents number of servers for queueing system.

3.2 Internet requests modeling

Full description of the model requires colors and functions definition in CPN ML language connected to the net elements:

```
(*--------System parameters:---------*)
val ps_ser_mean_time=100.0;
val pack_gen_mean_time=220.0;
(*--Internet request definition:----*)
color ID=int;    color PRT=int;
color START_TIME=int; color PROB=int;
color AUTIL=int; color RUTIL=int;
color INT=int; color PACKAGE=
product ID*PRT*START_TIME*PROB
        *AUTIL*RUTIL timed;
color PACK_QUEUE=list PACKAGE;
(*--Auxiliary types and variables:-*)
color random_val=int with 1..100;
var tim_val:INT; var n:INT;
color TIMER=int timed;
```

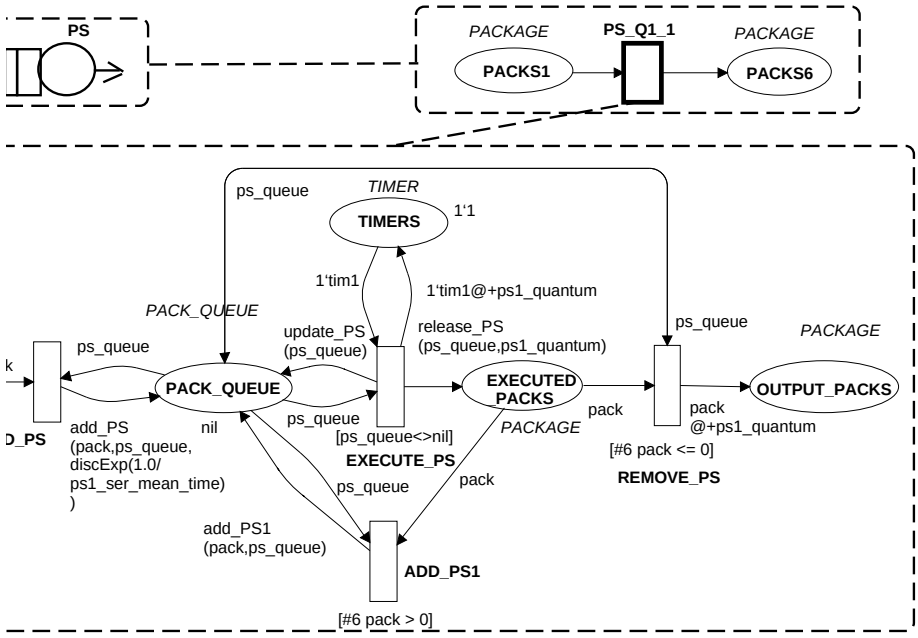


Figure 3: TCPNs model of -/M/1/PS/∞ queuing system.

```
var tim1:TIMER;  var pack:PACKAGE;
var ps_queue: PACK_QUEUE;
```

Corresponding arc functions (`add_PS()`, `add_PS1()`, `update_PS()`, `release_PS()`) release or insert tokens within the queue:

```
(*--Add request to queue(1):-------------*)
fun add_PS(pack:PACKAGE, queue:PACK_QUEUE,
ser_time:int)=if queue = nil
then [(#1 pack,#2 pack,#3 pack,#4 pack,
ser_time,ser_time)]
else (#1 pack,#2 pack,#3 pack,#4
pack,ser_time,ser_time)::queue;
(*--Add request to queue(2):-------------*)
fun add_PS1(pack:PACKAGE, queue:PACK_QUEUE)=
if queue=nil
then [pack] else pack::queue;
(*--Withdraw request from queue (1):----*)
fun update_PS(queue:PACK_QUEUE)=
rev(tl(rev queue));
(*--Withdraw request from queue (2):----*)
fun release_PS(queue:PACK_QUEUE,
ps_quantum:INT)=let
val r_pack=hd (rev queue) In
(#1 r_pack,#2 r_pack,#3 r_pack,
ran'random_val(), #5 r_pack,
#6 r_pack-ps_quantum) end;
```

As it was mentioned above, the main application of the software tool presented in the paper is modeling and evaluation of distributed Internet systems. To effectively model the Internet requests (or data packets) from the clients a separate token type (or token colour) has been proposed. The state of the system is determined by the number and distribution of the tokens representing data packets within the TCPN model. Each of the tokens representing a packet is a tuple `PACKAGE = (ID, PRT, START_TIME, PROB, AUTIL, RUTIL)` (compare with source code including color's definitions), where: `ID` - token identification (allowing token class definition etc.), `PRT` - priority, `START TIME` - time of a token occurrence in the system, `PROB` - probability value (used in token movement distribution in the net), `AUTIL` - absolute value of token utilization factor (for PS queue) and `RUTIL` - relative value of token utilization factor. Tokens have *timed* attribute scheduling them within places which are not queues.

While packets are being served, the components of a tuple are being modified. At the moment the given packet leaves the queueing system, a new `PROB` field value of `PACKAGE` tuple is being generated randomly (`release_PS` function). The value may be used to modify the load of individual branches in the queueing system model. Generally, the queueing system template is characterized by the following parameters: average tokens service time (`ps_ser_mean_time`), number of servers (number of tokens in `TIMERS` place) and service discipline (the TCPN's structure).

In the software tool developed, it is possible to construct queueing nets with queueing systems having PS and FIFO disciplines. These disciplines are the most commonly used for modeling Internet systems. Some our previous works include the rules of mapping TCPNs into queues of tokens scheduled according priorities [14, 15]. The presented templates have been tested on their compatibility with mathematical formulas determining the average queue length and service time as in [3].

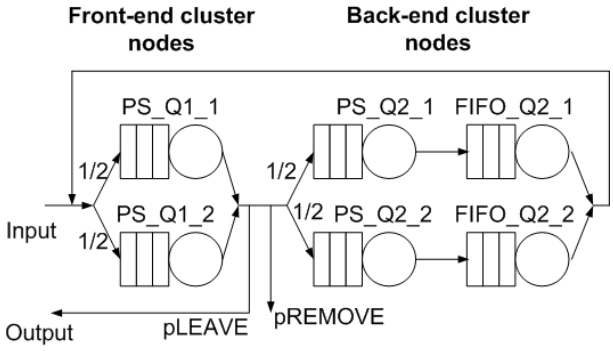


Figure 4: Queuing model of an example distributed Internet system environment.

4 Internet system modeling and analysis approach

Having a set TCPN based queueing systems models a systematic methodology of Internet system modeling and analysis may be proposed. Typically, modern Internet systems are composed of layers where each layer consists of a set of servers - a server cluster. The layers are dedicated for proper tasks and exchange requests between each other.

4.1 System structure modeling

To efficiently model typical Internet systems structures we proposed 3 modeling levels:

- superior - modeling of input process, transactions between layers and requests removal,
- layer - modeling of cluster structure,
- queue - modeling of queueing system.

To explain our approach to Internet system modeling a typical structure of distributed Internet system structure will be modeled and simulated. The example queueing model of the system informally sketched in fig. 1 is presented in fig. 4. It consists of two layers of server clusters and is constructed following the rules introduced in [6, 7, 8].

The front-end layer is responsible for presentation and processing of client requests. Nodes of this layer are modeled by PS queues. The next layer (back-end) implements system data handling. Nodes of this layer are modeled by using the serially connected PS and FIFO queues. The PS queue models the server processor and FIFO models the hard disc drive of server. Requests are sent to the system and then can be processed in both layers or removed after processing in front-end layer. The successfully processed requests are turned to the front-end layer and then send back to the customer.

Figure 5 shows the TCPN based model of above mentioned queueing network. The superior level of system description is presented in fig. 5a, whereas in fig. 5b and 5c detailed queueing systems topologies at each layer of the system are shown (compare fig. 4). Server cluster of the first layer (e.g. WWW servers; fig. 5b) as well as the second layer cluster (e.g. database; fig. 5c) have been demonstrated on the main page of TCPN net as substitution transitions: `front-end_cluster` and `back-end_cluster`.

On the superior level of system description (fig. 5a) we have also defined arrival process of the queueing network (`T0` transition with `TIMER0` and `COUNTER` places). `TIMER0` place and `T0` transition constitute a clock-like structure that produces tokens (requests) according to random, exponentially distributed frequency. These tokens are accumulated in a form of timed multiset in `PACKS1` place and then forwarded into the queueing-based model of the Internet system. When each token is being generated its creation time is memorized in the `PACKAGE` tuple. This makes it possible to conduct an off-line analysis of the model.

`T1` and `T5` as well as `T2` and `T3` transitions (compare fig. 5a) are in conflict. Execution of transition `T5` removes a serviced request from the net (modeling the system answer). If `T1` fires the request needs next transaction with the back-end system layer. Similarly, execution of transition `T2` removes a token from `T1` the net (modeling the possible loss of data packet). However, if `T3` fires, the data packet is transferred for processing in the second layer of the system. Guard functions connected to the mentioned transitions determine proportions between the tokens (packets) rejected and the ones remaining in the queueing net (in the example model approximately `30%` of the tokens are rejected or send back to the client).

Consequently, an executable (in a simulation sense) queueing network model is obtained. Tokens generated by arrival process are transferred in sequence by models of WWW server layer, by the part of the net that models loss (expiration) of some packets and by database layer. Provided that the system is balanced and has constant average arrival process, after some working time, the average values of the average queue length and response time are constant. Otherwise, their increase may occur.

The main parameters of the system modeled are the queue mean service time, the service time probability distribution function and the number of servicing units defined for each queueing system in the model. In the demonstrated model it has been assumed that queues belonging to a given layer have identical parameters.

4.2 System rerformance analysis

At this stage of our research it has been decided that simulation will be the main mechanism used to do analysis of the constructed model. In our simulations we applied the performance analysis. It allows collecting selected elements of the net state at the moment of an occurrence certain events during simulation. It has been assumed that in

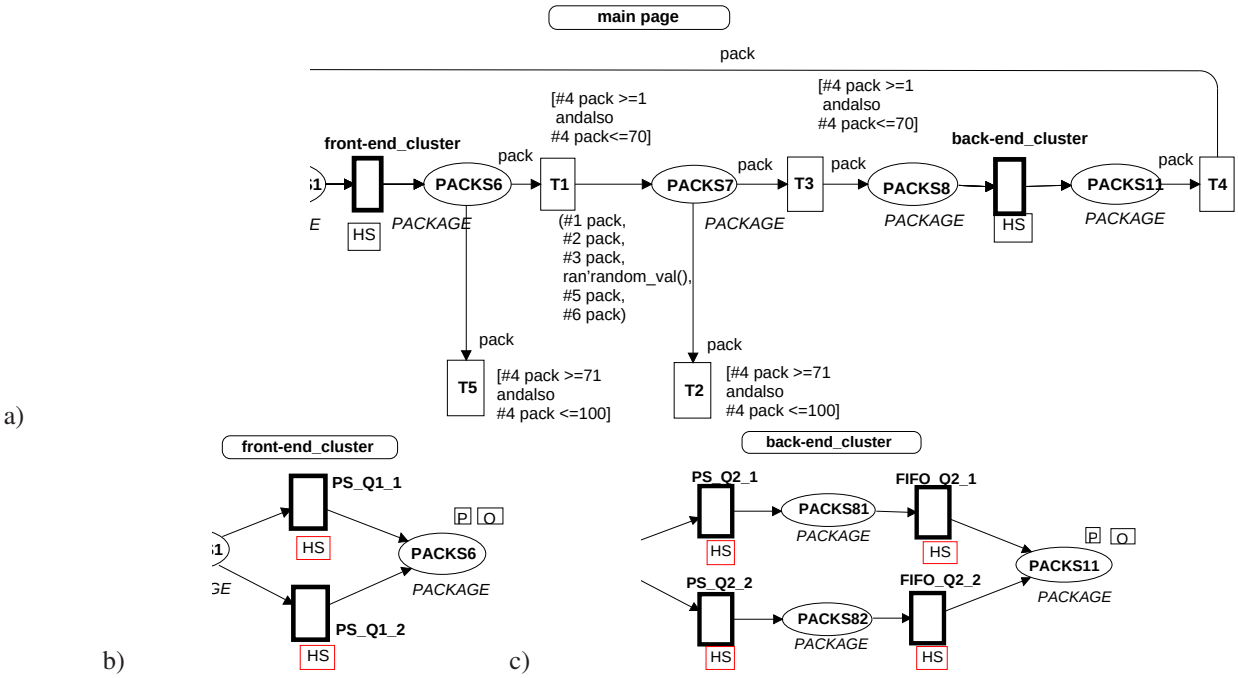


Figure 5: TCPNs based queueing system model: a) main page, b) front-end_cluster subpage and c) back-end_cluster subpage.

each of the model layers, queue lengths and response time will be monitored. Monitoring of the above mentioned parameters helps to determine whether the model of the system is balanced. Fig. 6 shows example plots obtained in the simulation of the discussed model.

The example experiment covered model time range from `0` to `100 000` time units. Fig. 6a shows the state of selected queue when the modeled system was balanced. Response time does not increase and remains around average value. System is regarded as balanced if the average queue lengths in all layers do not increase. In fig. 6b response time for unbalanced system was shown. The results concern the same layer as previously and identical time range for the simulation. It is clear that response time (fig. 6b) increase during the experiment. On the basis of the plot in fig. 6b, it can be concluded that the modeled system under the assumed external load would be overload and probably appropriate modifications in the structure of the system would be necessary. The software tool introduced in our paper makes it possible to estimate the performance of developing Internet system, to test and finally to help adjust preliminary design assumptions.

Having the possibility to capture the net's state during the simulation within a certain time interval, it can be possible to select model parameters in such a manner that they meet assumed time restrictions. Additionally, the parameters of real Internet system can be used to fit parameters of the constructed model.

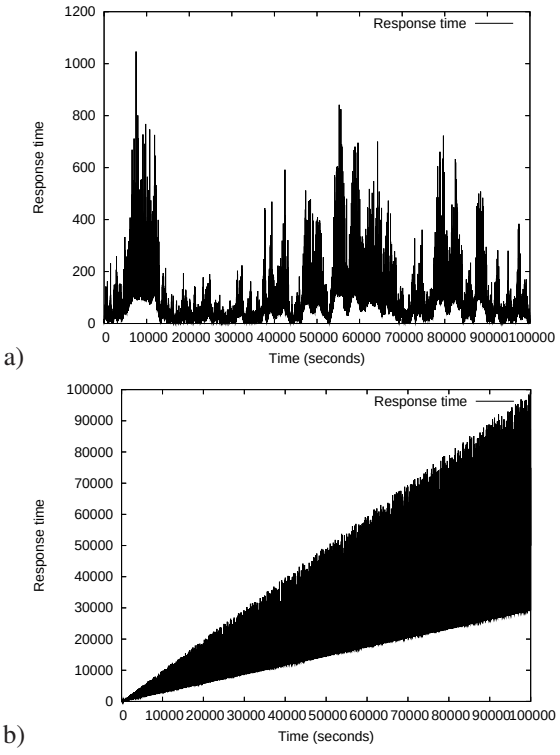


Figure 6: Sample system response time history: a) system balanced and b) system unbalanced.

5 Detailed vs. simplified Web cluster modeling

The worked out modeling and analysis methodology was used for construction and evaluation of several detailed models of architectures of distributed Internet systems. The analysis of the models were executed with the use of performance analysis tools for TCPN nets [20]. CSIM [17] simulating environment and experiments on real Internet system were used for TCPNs simulations evaluation. The overview of typical TCPNs based Internet system models analyzed so far can be found in [12].

In the remaining part of the paper we would like to discuss the following issue. During our research we proposed a *detailed TCPNs-based model of an example distributed Internet system*. The model makes it possible to express such phenomena as suspension of database service or database replication. The accuracy of the model was checked by comparison to a physical experimental computer cluster and a CSIM model. Obviously, the detailed modeling process requires the adequate knowledge of a database engine properties and consumes an amount of time. On the other hand, some authors (e.g. [18, 6, 8, 7]) "suspends" the modeling process on the level similar to one presented in section 4. This modeling level seems to be more acceptable for a broaden amount of Web systems developers. Taking into consideration the aforementioned rationale we decided to compare three models of an example distributed Internet system and evaluate the level of inaccuracy that simpler models contribute.

5.1 Detailed model

In fig. 7 a detailed queueing model of the example system has been presented. It consists of two cluster layers of servers. Customer requests are sent to the chosen node of front-end cluster with `1/2` probability. Then they are placed in the queue to get service. The service in the service unit (processor) can be suspend many times, if for example the requests need the database access. When the database access occurs, requests are sent to back-end layer. Any request can also be removed following `pREMOVE` path. In case of sending to the database, a requests steers itself to service in one of back-end nodes with `1/2` probability. The service in the database service unit may be suspend if access to the input/output subsystem of the database storage device is necessary. Requests are returned to the database service unit after the storage device is served. This operation can be repeated many times. After finishing servicing in the back-end layer, requests are returned to front-end servers (`pDB`). Requests can visit back-end layer during processing many times. After finish servicing in front-end server requests are sent to the customer (`pLEAVE`).

If the necessity of the database replication appears the requests are sent to next database node (`pREP`). Unless none of these two situations appear the requests are send to front-end layer (`pDB`). Replication can also cause resignation from transaction. Both the replication and the rejection of realizing transaction are modeled in simplistic way as delivery of task to the next location of replication.

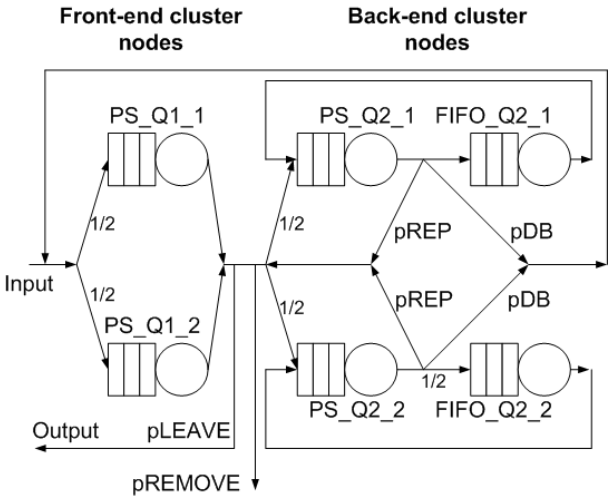


Figure 7: Detailed queueing model of the example distributed Internet system with suspension of transactions and database replication.

Probability	Probability values for model [%]
pREMOVE	30
pLEAVE	30
pDB	55
pREP	10

Table 1: The value of probabilities for model

In this model:

- `PS_Q1_1\2` is a queue `PS` modeling element that processes in front-end layer,
- `PS_Q2_1\2` is a queue `PS` modeling element that processes in back-end layer,
- `FIFO_Q2_1\2` is a queue `FIFO` modeling device that stores data in back-end layer.

Tab. 1 includes the probabilities values for Internet requests distribution for considered model. The parameters assumed for discussed model are as follows:

- external load,
- the identical parameters of queues,
- request distribution probabilities,
- the number of nodes in experimental environment.

In fig. 8 the detailed TCPNs-based model of the back-end cluster is presented. The structure of the TCP net corresponds with the detailed queing model depicted in fig. 7. Its most essential properties are:

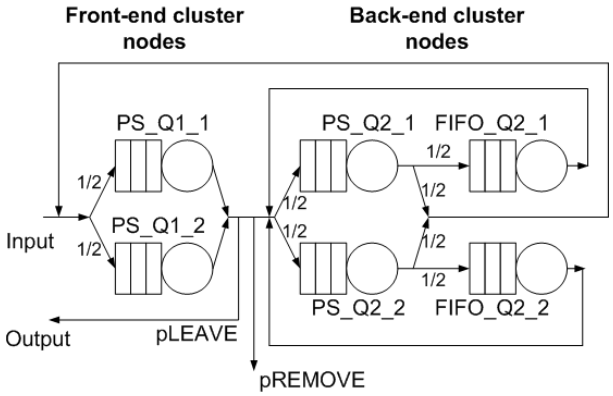


Figure 9: Semi-detailed queueing model of the example distributed Internet system.

- the possibility of directing tokens to any node (the location of replication),
- the return of tokens at the beginning of the layer,
- the realization of data synchronization in individual locations.

It is worth to notice that the complete detailed TCPNs-based model of the example Internet system can be derived as follows. The superior page of the model is identical at the one in fig. 5a. The front-end cluster model corresponds with the one from fig. 5b.

5.2 Semi-detailed model

In fig. 9 a semi-detailed queueing model of the example Internet system has been proposed. At this level of model simplification a database replication has been omitted. Still the service in the database service unit may be suspend if access to the input/output subsystem of the database storage device is necessary (compare subsection 5.1). A separate TCPNs-based semi-detailed model of the example system has been constructed, but due to limited size of the manuscript it has not been mentioned.

5.3 Simplified model

The so-called simplified model was thoroughly presented in section 4. Its queueing implementation can be seen in fig. 4, whereas its TCPNs-based implementation in fig. 5. The model does not takes into consideration suspension during the database access, nor database replicattion phenomenon.

5.4 Detailed model evaluation

Experimental environment and the CSIM packet were used to verify the detailed TCPNs-based model of the example system. The experimental system consisted of a net segment (100Mb/s), set of computers (Pentium 4, 2.8 GHz, 256 MB RAM) with Linux operating system (kernel 2.4.22) and Apache2 software (for WWW servers) as well as MySQL, version 4.0.15 (for database servers) [12].

The verification model was written by using CSIM simulator. This is a process oriented discreet event simulation package used with C or C++ compilers [17]. It provides libraries that a program written can be used in order to model a system and to simulate it. The models created by using CSIM [12] were based on presented queueing models (fig. 7) (similarly as TCPN models).

As a result we obtained the evaluated TCPNs based model of the Internet system discussed. The model made it possible to predict response time of the system developed. Tab. 2 shows the comparison of the detailed TCPNs-based model, detailed CSIM model of the example system, and a real experimental system. The system response times in each of the layer were measured during simulation (TCPNs model) and execution of the real environment. The experiments were conducted under different average external load values. In fig. 10 the response time of the layers during TCPNs-based model simulation under 100 [req./s] load are shown. It can be easily noticed, that system was balanced and the second layer may be the potential bottleneck due to higher response time.

5.5 Models evaluation

As we mentioned before, we treat the ”real” example system as the reference for the set of models to evaluate. In our experiments we assumed identical model parameters (e.g. queuing systems parameters, external load, probability of choosing different paths by the requests). Only the structures of modeled system were subsequently ”simplified”.

Tab. 3 includes the example response time statistics for the same external load for all types of models proposed. Tab. 4 includes the list of percentage of inaccuracy of the TCPNs-based and and CSIM models with respect to the real experimental system, respectively.

It can be easily noticed, that the simpler system model, the faster answer it gives. What is more, if we assume, that the detailed model is the ”closest” to the real system, an attempt to simplify it to the level discussed in section 5.3 without any modifications of queueing systems parameters is purposeless. Simultaneously, the semi-detailed model for some developers can have the quality sufficient to follow the real system behavior.

In fig. 11 an average response times history under the same external load for all types of the proposed models has been included. The average response times seem to oscillate around three constant values. It seems that a kind of equation showing the dependence between detailed, semi-detailed and simplified models may be derived.

6 Conclusions

It is still an open issue how to obtain an appropriate distributed Internet system. The demonstrated research results

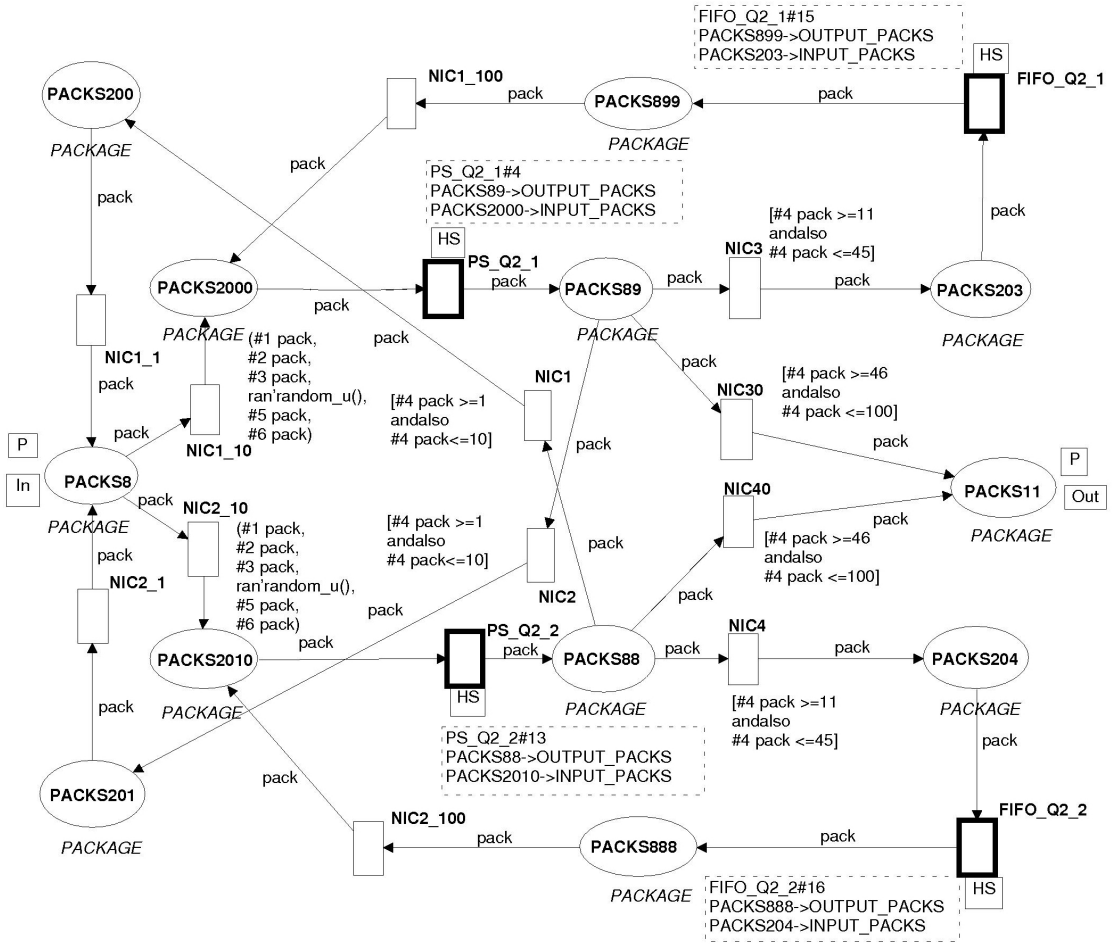


Figure 8: Back-end_cluster subpage of the detailed TCPNs-based queueing system model.

Load	Layer	TCPN Model	CSIM Model	Experiments	TCPN Model	CSIM Model
[req./s]		[ms]	[ms]	[ms]	Error [%]	Error [%]
100	front-end	49	50	36	26.5	28.0
100	back-end	567	598	409	27.5	31.0
300	front-end	701	743	579	17.4	22.1
300	back-end	813	798	648	20.4	39.7
500	front-end	1001	1109	921	8.0	16.4
500	back-end	1050	1083	973	7.3	10.0

Table 2: Layers response time for TCPNs model, CSIM model, and for experimental reference system

Layer	TCPN Simply [ms]	CSIM Simply [ms]	TCPN Semi-detailed [ms]	CSIM Semi-detailed [ms]	TCPN Detailed [ms]	CSIM Detailed [ms]	Experiments [ms]
front-end	93	114	123	159	167	197	153
back-end	115	135	157	192	222	257	201

Table 3: Layers response time for three TCPN, CSIM models and experimental results (the same load)

are an attempt to apply Queueing Theory (QT) and TCPNs formalism to the development of a software tool that can support distributed Internet system design. The idea of linking Queueing Nets Theory and Coloured Petri Nets was

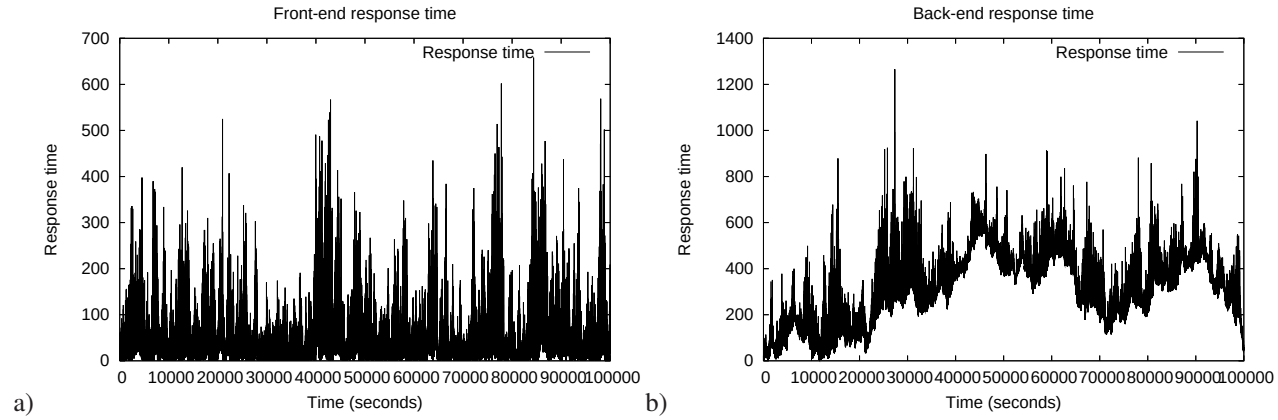


Figure 10: Detailed TCPNs-based model response time for layers a) - front-end layer, b) - back-end layer.

Layer	TCPN-Exp. Simply [%]	CSIM-Exp. Simply [%]	TCPN-Exp. Semi-detailed [%]	CSIM-Exp. Semi-detailed [%]	TCPN-Exp. Detailed [%]	CSIM-Exp. Detailed [%]
front-end	-57.73	-68.13	-22.40	-28.57	18.18	23.11
back-end	-63.41	-71.79	-34.89	-40.55	22.98	27.17

Table 4: Error - TCPN-CSIM-experiments

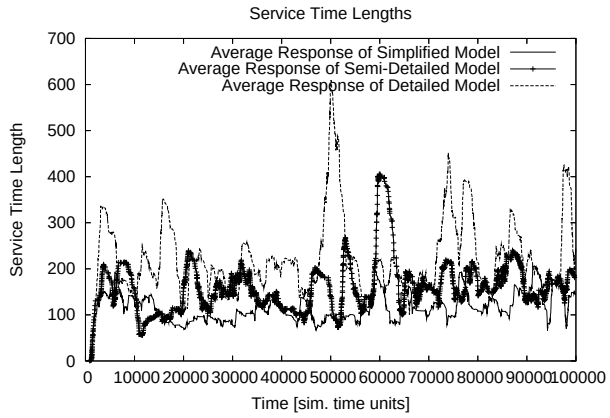


Figure 11: Average service times for Simplified, Semi-Detailed and Detailed models of the example system.

proposed previously by other authors in [6, 7]. However, in the presented approach queueing systems have been implemented using TCPNs formalism exclusively. As a consequence, alternative implementation of Coloured Queueing Petri Nets has been proposed. What was more, the rules of modeling and analysis of distributed Internet systems applying described net structures was introduced.

This paper deals with the problem of calculating performance values like the response time in distributed Internet systems environment. The values are calculated by using TCPNs. It is shown how the Coloured Petri Net model of a distributed Internet system is created with some of its data structures and functions, and gives an examples of system analysis. Finally, the paper discuses whether the detailed Internet system modeling brings substantial improvement in a real system mapping. The preliminary results show that probably adequate modification of queueing systems parameters can produce acceptable level of compatibility between ”simplified” models and the real systems.

Our future research will focus on dealing and analyzing another structures of distributed Internet systems using the software tool developed. It will be also dedicated to demonstrate compatibility of the models with the real systems. TCPN features such as tokens distinction will be of more extensive use. We will also make an attempt to create queueing model systems with defined token classes and consider a possibility to use state space analysis of TCPN net to determine properties of the system.

References

[1] F. Bause (1993) Queueing Petri Nets – a formalism for the combined qualititative and quantitative analysis of systems, *PNPM'93*, IEEE Press, pp. 14-23.

[2] V. Cardellini, E. Casalicchio, M. Colajanni (2002) The State of the Art in Locally Distributed Web-Server Systems, *ACM Computing Surveys, Vol. 34, No. 2*, ACM, pp.263–311.

[3] B. Filipowicz (2006) *Modeling and optimize queueing systems*, POLDEX, Krakow, (In Polish).

[4] K. Jensen (1996) *Coloured Petri Nets, Basic Concepts, Analysis Methods and Practical Use*, Springer.

[5] K. Jensen, L.M. Kristensen, L. Wells, (2007) Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems, *International Journal on Software Tools for Technology Transfer (STTT)*, Vol. 9, Numbers 3-4, pp. 213-254.

[6] S. Kounev (2006) *Performance Engineering of Distributed Component-Base Systems, Banchmarking, Modeling and Performance Prediction*, Shaker Verlag.

[7] S. Kounev (2006) Performance Modelling and Evaluation of Distributed Component–Based Systems Using Queuing Petri Nets, *IEEE Trans. on Soft. Eng., Vol 32. No. 7*, IEEE, pp. 486-502.

[8] S. Kounev, A. Buchmann, (2003) Performance Modeling and Evaluation of Large-Scale J2EE Applications, *Proceedings of the 29th Int. Conf. of the Comp. Meas. Group on Res. Manag. and Perf. Eval. of Enterprise Comp. Syst.*, ACM, Dallas, Texas, pp. 486-502.

[9] L. M. Kristensen, S. Christensen, K. Jensen, (1998) The Practitioner's Guide to Coloured Petri Nets *International Journal on Software Tools for Technology Transfer*, Vol. 2, pp. 98-132.

[10] B. Linstrom, L. Wells (1999) *Design/CPN Perf. Tool Manual*, CPN Group, Univ. of Aarhus, Denmark.

[11] T. Rak (2003) Model of Internet System Client Service, *Computer Science*, AGH Krakow, Vol. 5, pp. 55-65.

[12] T. Rak (2007) *The Modeling and Analysis of Interactive Internet Systems Realizing the Service of High-Frequency Offers*, PhD dissertation supervised by J. Werewka, Krakow, AGH, (In Polish).

[13] S. Samolej, T. Rak (2005) Time Properties of Internet Systems Modeling Using Coloured Petri Nets, *Real Time Systems* WKL, pp. 91-100, (In Polish).

[14] S. Samolej, T. Szmuc (2005) Time Constraints Modeling And Verification Using Timed Colored Petri Nets, *Real-Time Programming 2004*, Elsevier, pp. 127-132.

[15] S. Samolej, T. Szmuc (2005) TCPN-Based Tool for Timing Constraints Modelling and Validation, *Software Engineering, Evolution and Emerging Technologies, Volume 130*, IOS Press, pp. 194-205.

[16] S. Samolej, T. Szmuc (2008) Coloured Petri Nets based WWW Cluster Modeling and Development Method, *Inzynieria oprogramowania-od teorii do praktyki*, WKL, Warszawa, pp. 49-59 (In Polish).

[17] H. Schwetman (2001) CSIM19: A Powerfull Tool for Bilding System Models, *Proceedings Winter Simulation Conference*, IEEE, pp. 250-255.

[18] B. Urgaonkar, et. all (2005) An Analytical Model for Multi-tier Internet Service and Its Applications, *Proceedings of the ACM SIGMETRICS Inter. Conf. on Measurement and Model. of Comp. Sys.*, ACM, New York, pp. 291–302.

[19] L. Wells (2006) Performance analysis using CPN tools *Proc. of the 1st Inter. Conf. on Performance Evaluation Methodolgies and Tools, Pisa, Italy*, Article No. 59.

[20] L. Wells, et. all (2001) Simulation Based Performance Analysis of Web Servers, *Proc. of the 9th Inter. Workshop on Petri Nets and Performance Models*, IEEE, pp. 59–68.

Transformation of XML Data into XML Normal Form

Tadeusz Pankowski
Institute of Control and Information Engineering
Poznań University of Technology
Pl. M.S.-Curie 5, 60-965 Poznań, Poland
E-mail: tadeusz.pankowski@put.poznan.pl

Tomasz Piłka
Faculty of Mathematics and Computer Science
Adam Mickiewicz University
ul. Umultowska 87, 61-614 Poznań, Poland
E-mail: tomasz.pilka@amu.edu.pl

Keywords: XML, XML functional dependencies, XML normal forms, XML schema design

Received: March 4, 2009

Normalization as a way of producing good database designs is a well understood topic for relational data. In this paper we discuss the problem of eliminating redundancies and preserving data dependencies in XML data when an XML schema is normalized. Normalization is one of the main tasks in relational database design, where 3NF or BCNF, is to be reached. However, neither of them is ideal: 3NF preserves dependencies but may not always eliminate redundancies, BCNF on the contrary – always eliminates redundancies but may not preserve constraints. In this paper we consider the possibility of achieving both redundancy-free and dependency preserving form of XML schema. We show how the XML normal form can be obtained for a class of XML schemas and a class of XML functional dependencies. We relate our approach to the decomposition algorithm proposed by Arenas and Libkin in [1].

Povzetek: Razvita je metoda pretvorbe XML podatkov v normalizirano obliko s poudarkom na odstranjevanju redundanc.

1 Introduction

Normal forms, as desired forms for schemas defining structures and properties of data collections, were first proposed and investigated by Codd in early 70s. The most important of them are 3NF (*Third Normal Form*) [2] and BCND (*Boyce-Codd Normal Form*) [3]), where BCNF is a stronger definition of 3NF. Definitions of normal forms are based on the functional dependencies defined among the attributes constituting the relational schema, and specify requirements to be satisfied by the set of these functional dependencies. Then any relation that is an instance of the schema and satisfies the given set of functional dependencies, is free of harmful redundancies and anomalies. In the normalization process, the initial poor designed relational schema is decomposed into an equivalent set of well-designed schemas, i.e. into schemas in desired normal forms (usually in 3NF and often also in BCNF).

Recently, as XML becomes popular as the standard data model for storing and interchanging data on the Web and more companies adopt XML as the primary data model for storing information, XML schema design has become an increasingly important issue. Thus, we observe attempts to extend normal forms and database design principles to XML databases. Research on normalization of XML data was reported in a number of papers. In [1], Arenas and Libkin extended the relational tuple-oriented definition of functional dependencies to so-called *tree tuple*-based functional dependencies and developed the first theory of XML functional dependencies (XFDs) and XML normal forms (XNFs). This approach was further studied by Kolahi [4, 5], and Kolahi and Libkin [6]. Integrity constraints for XML data (including keys) were studied extensively in Buneman et al. in [7], and Fan and Simeon in [8]. The equivalence between XFDs and relational FDs was investigated by Vincent et al. in [9, 10]. Yu and Jagadish proposed in [11] a new XML normal form, called GTT-XNF, that is based on *Generalized Tree Tuples* (GTT).

Central objectives of a good schema design is to avoid data redundancies and to preserve dependencies enforced by the application domain (these dependencies are formalized by means of functional dependencies). Existence of redundancy can lead not only to a higher data storage cost but also to increased costs for data transfer and data manipulation. It can also lead to update anomalies [12].

One strategy to avoid data redundancies is to design redundancy-free schema. One can start from an intuitively correct XML schema and a given set of functional dependencies reflecting some rules existing in application domain. Then the schema is normalized, i.e. restructured, in such a way that the newly obtained schema has no re-

dundancy, preserves all data (is a lossless decomposition) and preserves all dependencies. In general, obtaining all of these three objectives is not always possible, as has been shown for relational schemas [13]. However, in the case of XML schema, especially thanks to its hierarchical structure, this goal can be more often achieved [4].

1.1 Related work

An algorithm, called *the Decomposition Algorithm* (DA) normalizing XML schemas was proposed in [1]. The algorithm converts any DTD, given a set of XML functional dependencies (XFDs), into DTD in XML normal form (XNF). The decomposition algorithm consists of two basic operations: *moving attributes*, and *creating new element types*. These operations are performed when an XFD violating XNF is identified. Thus, the basic idea is similar to normalization of relational data, when the second normal form (2NF), the third normal form (3NF), or the Boyce-Codd normal form (BCND) are to be achieved. In the relational counterparts during the normalization process a relational schema is decomposed into a set of its projections. Thus, we can obtain a set of separate relational schemas as the result of the normalization process performed for an initial relational schema. In the case of XML documents, the result is still one XML document restructured accordingly. In [14], an information-theoretic approach to normal forms for relational and XML data has been developed. Some other papers, e.g. [5, 15, 16, 6] study XML design and normalization for native or relational storage of XML documents. In [17, 18] approaches for obtaining well-designed XML schemas from conceptual ER (*Entity-Relationship*) schemas have been discussed.

1.2 Contribution

In this paper we describe a systematic approach to the normalization of XML data when so-called *cyclic functional dependencies exist*, i.e. dependencies, which in the relational case are functional dependencies of the form $\{A, B \rightarrow C, C \rightarrow A\}$, where A, B, C are attributes in a relational schema $R(A, B, C)$. It is well known from relational database theory [13], that the schema $R(A, B, C)$ is then in 3NF, but is not in BCNF. As a result, instances of $R(A, B, C)$ have redundancies, but decomposition the schema into BCNF leads to two schemas $R1(C, A)$ and $R2(C, B)$, which are free of redundancy but do not preserve dependency $A, B \rightarrow C$.

We focus on normalization procedure for XML schemas with cyclic XFDs. The contributions of the paper are the following:

- We use a language based on tree patterns [19, 20] to express XML schemas and XML functional dependencies. This notation is used in the formal analysis of properties of XML normal form as well as the base for developing transformation algorithms.
- We propose an approach to obtain XNF starting from ER schema. In the first step the ER schema is converted into an XML schema satisfying a *necessary conditions* (see Theorem 7.3) which is a prerequisite to successful applying of DA algorithm [1]. We show that the presence of cyclic dependencies results in a bad behavior of the DA algorithm.

This paper is organized as follows. In Section 2 we introduce a running example and motivate the research. In Section 3 a relational form of the running example is considered and some problems with its normalization are discussed. Basic notations relevant to the discussed issue from the XML perspective, are introduced in Section 4. We define tree patterns and use them to formal definition of tree tuples and instances. In Section 5, tree patterns are used to define data dependencies: XML functional dependencies (XFDs) and keys (a subclass of XFDs). In Section 6, an XML normal form (XNF) is defined (according to [1]) and we show how this form can be obtained for our running example. We discuss different normalization alternatives – we show advantages and drawbacks of some schema choices. A method for transforming an XML schema into XNF is proposed in Section 7. First, for the XML schema the conceptual model in a form of ER schema is created. This schema and functional dependencies among its attributes are the basis for creating an initial XML schema satisfying the necessary condition formulated in Theorem 7.3. This schema is the subject for further normalization. Section 8 concludes the paper.

2 Redundancies in XML data - motivation example

Our primary goal is to devise methods which will allow checking correctness of XML data and designing its expected (normalized) form. We expect such data to be devoid of the redundancies and immune to the update anomalies.

In the case of XML schemas some redundancy problems may occur because of bad design of hierarchical structure of XML document. On the other hand, the hierarchical structure of this data can sometimes help conduct the normalization of XML data.

Example 2.1. *Let us consider an XML schema tree (Figure 1) that describes a fragment of a database for storing data about parts and suppliers offering these parts. Its DTD specification is given in Figure 2, and an instance of this schema is depicted in Figure 3. Each* $part$ *element has identifier* pId*. One part may be offered by zero or more suppliers. Offers are stored in* $offer$ *elements. Each offer has: offer identifier* `oId`*, supplier identifier* `sId`*, price* `price`*, delivery time* `delivTime`*, and delivery place* `delivPlace`*.*

We assume that the following constraints must be satisfied by any instance of this schema:

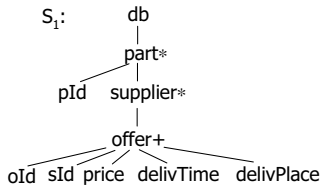


Figure 1: Sample XML schema tree

```
db        →  part*
part      →  pId supplier*
supplier  →  offer+
offer     →  oId sId price delivTime
             delivPlace
```

Figure 2: DTD productions describing the XML schema in Figure 1

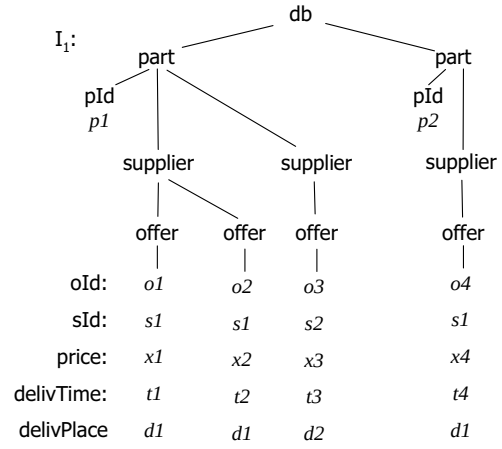


Figure 3: Sample instance of schema S_1

- all $offer$ children of the same $supplier$ must have the same values of sId; this is similar to relational functional dependencies, but now we refer to both the values (text value of sId), and to structure (children of the same $supplier$).
- $delivPlace$ functionally depends on part (pId) and supplier (sId), i.e. when a supplier has two different offers for the same part (possibly with different $delivTime$ and/or $price$) the $delivPlace$ is the same - see offers $o1$ and $o2$ in Figure 3.
- $delivPlace$ functionally determines supplier (sId). It means that having a delivery place ($delivPlace$) we exactly now which supplier is associated to this place; although one supplier can own many delivery places. For example, in Figure 3 $d1$ is delivery place uniquely associated to the supplier $s1$.

It is easily seen that schema in Figure 1 leads to redundancy: sid (and also all other data describing suppliers such as e.g.: name and address) and $delivPlace$ are stored multiple times for a supplier.

Further on we will show that a special caution should be paid to such kind of dependencies as these in which participates $delivPlace$. In this case we have to do with "cyclic" dependencies, i.e. $delivPlace$ depends on pId and sId ($pId, sId \rightarrow delivPlace$) and sId depends on $delivPlace$ ($delivPlace \rightarrow sId$).

3 Redundancies and dependencies in relational databases

3.1 Relational schemas and functional dependencies

In relational data model, a relational schema is understood as a pair $\mathcal{R} = (U, F)$, where U is a finite set of *attributes*, and F is a set of *functional dependencies* over F. A functional dependence (FD) as an expression of he form

$$X \rightarrow Y,$$

where $X, Y \subseteq U$ are subsets of U. If $Y \subseteq X$, then $X \rightarrow Y$ is a *trivial* FD. By F^+ we denote all dependencies which can be infer from F using, for example, Armstrong's axioms [21, **?**].

A *relation* of type U is a finite set of tuples of type U. Let $U = \{A_1, ..., A_n\}$ and $dom(A)$ be the *domain* of attribute $A \in U$. Then a tuple $[A_1 : a_1, ..., A_n : a_n]$, where $a_i \in dom(A_i)$, is a tuple of type U.

A relation R conforms to a schema $\mathcal{R} = (U, F)$ (is an instance of this schema) if R is of type U, and all dependencies from F^+ are satisfied by R. An FD $X \rightarrow Y$ is satisfied by R, denoted $R \models X \rightarrow Y$, if for each tuples $r_1, r_2 \in R$ holds

$$\pi_X(r_1) = \pi_X(r_2) \Rightarrow \pi_Y(r_1) = \pi_Y(r_2),$$

where $\pi_X(r)$ is the *projection* of tuple r on the set X of attributes.

A *key* in $\mathcal{R} = (U, F)$ is such a minimal set K of attributes that $K \rightarrow U$ is in F^+. Then each $A \in K$ is called a *prime* attribute.

3.2 Normalization of relational schemas

The main task in relational schema normalization is producing such a set of schemas that posses the required form, usually 3NF or BCNF. The normalization process consists in decomposition of a given input schema. The other approach consists in synthesizing 3NF from functional dependencies [22].

Ideally, a decomposition of a schema should be lossless, i.e. should preserve data and dependencies. Let $\mathcal{R} = (U, F)$, $U_1, U_2 \subseteq U$, and $U_1 \cup U_2 = U$, then schemas $\mathcal{R}_1 = (U_1, F_1)$ and $\mathcal{R}_2 = (U_2, F_2)$ are a lossless decomposition of $\mathcal{R} = (U, F)$, iff:

– The decomposition preserves data, i.e. for each instance R of $\mathcal{R}$ the natural join of projections of R on U_1 and U_2 produces the relation equal to R, i.e.

$$R = \pi_{U_1}(R) \bowtie \pi_{U_2}(R).$$

– The decomposition preserves dependencies, i.e.

$$F^+ = (F_1 \cup F_2)^+,$$

where $F_1 = \{X \to Y \mid X \to Y \in F \wedge X \cup Y \subseteq U_1\}$, and similarly for F_2.

The decomposition $((U_1, F_1), (U_2, F_2))$ of (U, F) preserves data, if $U_1 \cap U_2 \to U_1 \in F^+$ (or, symmetrically, $U_1 \cap U_2 \to U_2 \in F^+$) [23]. Then we say that the *decomposition is determined* by the functional dependence $U_1 \cap U_2 \to U_1 \in F^+$.

A schema $\mathcal{R} = (U, F)$ is in 3NF if for every FD $X \to A \in F^+$, holds:

– X is a superkey, i.e. a key is a part of X, or

– A is prime.

The second condition says that only prime attributes may be functionally dependent on a set of attributes which is not a key. A schema is in BCNF if only the first condition of the two above is allowed. It means, that if whenever a set X determines functionally an attribute A, then X is a superkey, i.e. determines the whole set U.

The aim of a normalization process is to develop normal forms by analyzing functional dependencies and successive decomposition of the input relational schema into its projections. In this way a well-designed schema can be obtained, where unnecessary redundancies and update anomalies had been eliminated. In practice, 3NF is accepted as the most desirable form of relational schemas It does not eliminate all redundancies but guaranties dependency preservation. On contrast, BCNF eliminates all redundancies but does not preserve all dependencies.

In [6] it was shown that 3NF has the least amount of redundancy among all dependency preserving normal forms. The research adopts a recently proposed information-theoretic framework for reasoning about database designs [14].

3.3 Relational analysis of XML schema

Let us consider the relational representation of the data structure presented in Figure 1. Then we have the following relational schema:
$\mathcal{R} = (U, F)$, where
$U = \{oId, sId, pId, price, delivTime, delivPlace\}$,
$F = \{oId \to sId, pId, price, delivTime, delivPlace,$
$sId, pId \to delivPlace,$
$delivPlace \to sId\}$.

In $\mathcal{R}$ there is only one *key*. The key consists of one attribute oId since all attributes in U functionally depends on oId. Thus, R is in 2NF and oId is the only prime (key) attribute in $\mathcal{R}$. Additionally, we assume that a given supplier delivers a given part exactly to one place ($pId, sId \to delivPlace$). Moreover, delivery place $delivPlace$ is connected with only one supplier ($delivPlace \to sId$).

R is not in 3NF because of the functional dependency $sId, pId \to delivPlace$:

– sId, pId is not a superkey, and

– $delivPlace$ is not a prime attribute in U.

Similarly for $delivPlace \to sId$.

In this case the lack of 3NF is the source of redundancies and update anomalies. Indeed, for example, the value of $delivPlace$ will be repeated as many times as many different tuples with the same value of the pair (sId, pId) exist in the instance of $\mathcal{R}$. To eliminate this drawback, we can decompose $\mathcal{R}$ into two relational schemas, $\mathcal{R}_1$ and $\mathcal{R}_2$, which are in 3NF. The decomposition must be based on the dependency $sId, pId \to delivPlace$ which guarantees that the decomposition preserves data. In the result we obtain:

$\mathcal{R}_1 = (U_1, F_1)$, where
$U_1 = \{oId, sId, pId, price, delivTime\}$,
$F_1 = \{oId \to sId, pId, price, delivTime\}$.

$\mathcal{R}_2 = (U_2, F_2)$, where
$U_2 = \{sId, pId, delivPlace\}$,
$F_2 = \{sId, pId \to delivPlace,$
$delivPlace \to sId\}$.

The discussed decomposition is both data and dependencies preserving, since:
$R(U) = \pi_{U_1}(R) \bowtie \pi_{U_2}(R)$,
for every instance R of schema $\mathcal{R}$, and $F = (F_1 \cup F_2)^+$.
However, we see that $\mathcal{R}_2$ is not in BCNF, since $delivPlace$ is not a superkey in $\mathcal{R}_2$.

The lack of BCNF in $\mathcal{R}_2$ is the reason of redundancies. For example, in table R_2 we have as many duplicates of sId as many tuples with the same value of $delivPlace$ exist in this table.

R_2

sId	pId	$delivPlace$
s1	p1	d1
s1	p2	d1
s1	p3	d2
s2	p1	d3

We can further decompose $\mathcal{R}_2$ into BCNF schemas $\mathcal{R}_{21}$ and $\mathcal{R}_{22}$, taking $delivPlace \to sId$ as the base for the decomposition. Then we obtain:

$\mathcal{R}_{21} = (U_{21}, F_{21})$, where
$U_{21} = \{delivPlace, sid\}$,
$F_{21} = \{delivPlace \rightarrow sId\}$.

$\mathcal{R}_{22} = (U_{22}, F_{22})$, where
$U_{22} = \{pId, delivPlace\}$,
$F_{22} = \emptyset$.

After applying this decomposition to R_2 we obtain tables R_{21} and R_{22}:

R_{21}

sId	$delivPlace$
s1	d1
s1	d2
s2	d3

R_{22}

pId	$delivPlace$
p1	d1
p2	d1
p3	d2
p1	d3

This decomposition is information preserving, i.e.

$$R_2 = R_{21} \bowtie R_{22},$$

but does not preserve functional dependencies, i.e.

$F_2 \neq (F_{21} \cup F_{22})^+ = F_{21}$.

We can observe some negative consequences of the loss of functional dependencies in the result decomposition.

Assume that we insert the tuple $(p1, d2)$ into R_{22}. The tuple will be inserted because it does not violate any constrain imposed on $\mathcal{R}_{22}$. However, taking into account table R_{21} we see that supplier $s1$ (determined by $d2$ in force of $delivPlace \rightarrow sId$) offers part $p1$ in the place $d1$. Thus, the considered insertion violates functional dependency $sId, pId \rightarrow delivPlace$ defined in $\mathcal{R}_2$.

The considered example shows that in the case of relational databases we are not able to completely eliminate redundancies and also preserve all functional dependencies. It turns out ([6]) that the best form for relation schema is 3NF, although some redundancies in tables having this form can still remain.

In next section we will show that the hierarchical structure of XML documents can be used to overcome some of the limitations of relational normal forms [24]. As it was shown in [5], there are decompositions of XML schemas that are both information and dependency preserving. In particular, we can obtain a form of XML schema that is equivalent to BCNF, i.e. eliminates all redundancies, and additionally preserves all XML functional dependencies.

4 XML schemas and instances

Schemas for XML data are usually specified by DTD or XSD [25, 26]. In this paper an *XML schema* (a schema for short) will be specified by means of a slightly simplified version of DTD. We will assume that both attributes and elements of type `#PCDATA` will be represented by so called *terminal elements* labeled with *terminal labels* and having text values. Additionally, terminal elements may have only one occurrence within children of one non-terminal element.

Definition 4.1. *Let L be a set of labels, $Ter \subset L$ be a set of terminal labels and $r \in L - Ter$ be a root label. Let ρ be a set of productions of the form:*

$$l \rightarrow \alpha,$$

where:

- $l \in L - Ter$;
- *r does not occur on the right-hand side of any production;*
- *α is a regular expression over $L - \{r\}$ defined as follows:*
  $\alpha ::= \beta \mid \gamma$
  $\beta ::= l \mid \beta? \mid \beta* \mid \beta+$
  $\gamma ::= A \mid A? \mid \beta \mid \gamma\,\gamma \mid \gamma|\gamma$
  $l \in L - Ter - \{r\}$, $A \in Ter$.

Then the quadruple

$$S = (r, L, Ter, \rho),$$

is an XML schema.

XML schemas will be often represented as XML schema trees (Figure 1) (or as XML schema graphs when the schema is recursive). In this paper we restrict ourselves to non-recursive schemas. An example of XML schema, corresponding to the schema tree in Figure 1, was given in Figure 2, where: db is the root, $part, supplier$, and $offer$ are non-terminal labels, and $pid, oid, sid, price, delivTime, delivPlace$ are terminal labels.

The following notion of *tree patterns* [19] will be useful to define tree tuples, tree formulas and XML functional dependencies.

Definition 4.2. *Let L be a set of labels, and $Ter \subset L$ be its subset of terminal labels. A tree pattern is an expression conforming to the syntax:*

$$e ::= A \mid l \mid l/e \mid l[e_1, \ldots, e_k] \mid l[e_1, \ldots, e_k]/e,$$

where $A \in Ter$, $l \in L - Ter$, $n \leq 1$.

To denote that a tree pattern ϕ is build using the set $\{A_1, \ldots, A_n\}$ of terminal labels, we will write $\phi(A_1, \ldots, A_n)$.

Definition 4.3. *Let $\phi(A_1, \ldots, A_n)$ be a tree pattern, and $x_1, \ldots, x_n$ be text-valued variables. Then the expression*

$$\phi(A_1 : x_1, \ldots, A_n : x_n),$$

is a tree formula.

Definition 4.4. *Let $\phi(A_1 : x_1, \ldots, A_n : x_n)$ be a tree formula, and ω be a variable valuation, i.e. a function*

$$\omega : \{x_1, \ldots, x_n\} \to \texttt{Str} \cup \{\bot\},$$

where Str *is a set of text values, and $\bot$ is a distinguished null value. Then*

$t = \phi(A_1 : x_1, \ldots, A_n : x_n)(\omega) = \phi(A_1 : \omega(x_1), \ldots, A_n : \omega(x_n))$

is called a tree tuple of type $\phi(A_1, \ldots, A_n)$ with values $(A_1 : \omega(x_1), \ldots, A_n : \omega(x_n)$.

Without loss of generality, a tree tuple of type $\phi(A_1, \ldots, A_n)$ will be denoted also as:

$t = \phi(A_1 : a_1, \ldots, A_n : a_n)$,
$t = \phi(A_1, \ldots, A_n)(\omega)$,
$t = \phi(\omega(A_1), \ldots, \omega(A_n))$,
$t = \phi(A_1 : t.A_1, \ldots, A_n : t.A_n)$.

Definition 4.5. *Let t be a tree tuple of type $\phi(U)$, and let $U' = \{A_1, \ldots, A_n\} \subseteq U$. The projection of t on U', denoted $\pi_{U'}(t)$, is the set of fields $(A_i : t.A_i)$ occurring in t, i.e.:*

$$\pi_{U'}(t) = \{A_1 : t.A_1, \ldots, A_n : t.A_n\}.$$

We assume that there is a function $type(t)$ that for a tree tuple t returns its type, it will be denoted by

$$type(t) = \phi(A_1, \ldots, A_n).$$

An XML database consists of a set of XML data, and an XML data is an instance of an XML schema. It will be convenient to distinguish between two kinds of instances:

- an instance as a set of tree tuples, referred to as a *canonical instance*,
- an instance as a labeled tree, referred as an XML tree.

For one canonical instance there may be a number of XML trees representing it. In the set of all such XML trees we will be interested in such that have a required form, so called XML normal form (XNF). The canonical instance as well as all corresponding XML trees must conform to the given XML schema.

Definition 4.6. *The set of tree tuples*

$$D = \{t_1, \ldots, t_N\},$$

is a canonical instance of an XML schema $S = (r, L, Ter, \rho)$ if the type, $type(t)$, of any tuple $t \in D$ conforms to S.

The conformance of a tree tuple type to an XML schema is defined as follows.

Definition 4.7. *Let $\phi(A_1, \ldots, A_n)$ be the type of a tree tuple t. This tree pattern conforms to the XML schema $S = (r, L, Ter, \rho)$, if:*

1. *$\phi(A_1, \ldots, A_n) = r[e]$, and*
2. *e conforms to ρ according to r. The conformance of a pattern e to the set of productions ρ according to a label l, is defined as follows:*
   - *if e is A_i, then $A_i \in Ter$, and there is such the production $l \to \alpha$ in ρ that A_i occurs in α;*
   - *if e is l'/e', then there is such the production $l \to \alpha$ in ρ that l' occurs in α, and e' conforms to ρ according to l';*
   - *if e is $l'[e_1, \ldots, e_n]$, then there is such the production $l \to \alpha$ in ρ that l' occurs in α, and every pattern e_i, $1 \leq i \leq n$, conforms to ρ according to l'.*
   - *if e is $l'[e_1, \ldots, e_n]/e'$, then $l'[e_1, \ldots, e_n]$ must conform to ρ according to l, and e' must conform to ρ according to l'.*

We define XML tree as an ordered rooted node-labeled tree over a set L of labels, and a set $\texttt{Str} \cup \{\bot\}$, which elements are used as values of terminal nodes.

Definition 4.8. *An XML tree I is a tuple $(root, N^e, N^t, child, \lambda, \nu)$, where:*

- *$root$ is a distinguished root node, N^e is a finite set of non-terminal element nodes, and N^t is a finite set of terminal nodes;*
- *$child \subseteq (\{r\} \cup N^e) \times (N^e \cup N^t)$ – a relation introducing tree structure into the set $\{r\} \cup N^e \cup N^t$, where r is the root, each non-terminal node has at least one child, terminal nodes are leaves;*
- *$\lambda : \{root\} \cup N^e \cup N^t \to L$ – a function labeling nodes with labels;*
- *$\nu : N^t \to Str \cup \{\bot\}$ – a function assigning terminal nodes with values.*

Definition 4.9. *We say that an XML tree $I = (root, N^e, N^t, child, \lambda, \nu)$ conforms to an XML schema $S = (r, L, Ter, \rho)$, denoted $I \models S$, if*

- *$\lambda(root) = r$; if $n \in N^e$, then $\lambda(n) \in L - Ter$; if $n \in N^t$, then $\lambda(n) \in Ter$;*
- *if $\lambda(n) = l$, $l \to \alpha \in \rho$, and $n_1, ..., n_k$ are children of n, then the sequence $\lambda(n_1) \cdots \lambda(n_k)$ is a word in the language generated by α.*

It will be useful to perceive an XML canonical instance D with tuples of type ϕ as a pair (ϕ, Ω) (called a *description*), where Ω is a set of valuations for ϕ.

Example 4.1. *For the instance I_1 in Figure 3 we have:*
$(\phi(pId, oId, sId, price, delivTime, delivPlace)$
$\{(p1, o1, s1, x1, t1, d1), (p1, o2, s1, x2, t2, d1),$
$(p1, o3, s2, x3, t3, d2), (p2, o4, s1, x4, t4, d1)\})$.

An XML tree I satisfies a description (ϕ, Ω), if the $root$ of I satisfies ϕ for every valuation $\omega \in \Omega$, where:

1. $(I, root) \models (r[e], \omega)$, iff $\exists n \in N^e\ child(r, n) \wedge (I, n) \models (e, \omega)$;
2. $(I, n) \models (A, \omega)$, iff $\lambda(n) = A$ and $\nu(n) = \omega(A)$.
3. $(I, n) \models (l/e, \omega)$, iff $\lambda(n) = l$ and $\exists n' \in N^e$ $child(n, n') \wedge (I, n') \models (e, \omega)$
4. $(I, n) \models (l[e_1, ..., e_k], \omega)$, iff $\lambda(n) = l$ and for each i, $1 \leq i \leq k$, exists n_i such that $child(n, n_i) \wedge (I, n_i) \models (e_i, \omega))$;
5. $(I, n) \models (l[e_1, ..., e_k]/e, \omega)$, iff $(I, n) \models (l[e_1, ..., e_k], \omega)$ and exists n' such that $child(n, n') \wedge (I, n') \models (e, \omega))$.

A description (ϕ, Ω) represents a class of ϕ instances with the same set of values (the same Ω), since elements in instance trees can be grouped and nested in different ways.

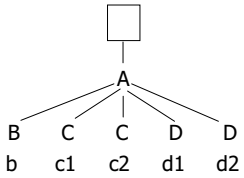


Figure 4: A simple XML tree

For example, the XML tree in Figure 4 satisfies (among others) the following two descriptions (ϕ_1, Ω_1), and (ϕ_2, Ω_2), where:

$\phi_1(B, C) = \quad /A[B, C],$
$\Omega_1 = \quad \{(b, c1), (b, c2)\};$
$\phi_2(B, C, D) = \quad /A[B, C, D],$
$\Omega_2 = \quad \{(b, c1, d1), (b, c2, d1)\}.$

5 XML functional dependencies and keys

Over an XML schema we can define some constraints, which specify functional dependencies between values and/or nodes in instances of the schema. These constraints are called XML functional dependencies (XFD).

Definition 5.1. *A tree pattern of the form $\phi(A_1, \ldots, A_k)/A$ conforming to an XML schema $S = (r, L, Ter, \rho)$ is an XML functional dependency (XFD) over S. Then we say that the set of paths terminating in $(A_1, \ldots, A_k)$ and satisfying ϕ, determines the path terminating in A and satisfying ϕ/A.*

Satisfaction of an XFD is defined against an canonical instance of XML schema.

Definition 5.2. *Let S be an XML schema, D be a canonical instance of S and $f = \phi(A_1, \ldots, A_k)/A$ be an XFD on S. We say the D satisfies f, denoted $D \models f$ if for any two tree tuples $t_1, t_2 \in D$ the following implication holds*

$$\pi_{A_1,\ldots,A_k}(t_1) = \pi_{A_1,\ldots,A_k}(t_2) \Rightarrow \pi_A(t_1) = \pi_A(t_2).$$

Assume that $A_1, \ldots, A_k, A$ terminates, respectively, paths $p_1, \ldots, p_n, p$. Then the XPath-oriented XFD $\phi(A_1, \ldots, A_k)/A$ corresponds to the following path-oriented XFD defined in [1]:

$$p_1.A_1, \ldots, p_k.A_k \rightarrow p.A.$$

The XPath-oriented definition has the following advantages:

- it is easy to check, using only the XPath semantics, whether an XML tree satisfies the XFD or not (see below),
- this form of XFD can be used to generate an XQury program performing some transformation operations (see the next section).

An XFD f can be interpreted as XPath expression, i.e. $f(x_1, \ldots, x_k) := \phi(A_1 : x_1, \ldots, A_k : x_k)/A$, that for a given valuation ω of its variables returns a sequence of objects (nodes or text values).

An XML tree $I = (S, \Omega)$ satisfies an XFD $f(\mathbf{x})$ if for each valuation $\omega \in \Omega$ of its variables, $f(\omega(\mathbf{x}))$ evaluated on I returns an empty sequence or a singleton , i.e.

$$count([\![f(\omega(\mathbf{x}))(I)]\!]) \leq 1,$$

where $[\![expr]\!](I)$ is the result of evaluation $expr$ on the instance I.

An XML tree $I = (S, \Omega)$ satisfies an XFD f if for each valuation $\omega \in \Omega$, $f(\omega)$ evaluated on I returns an empty sequence or a singleton, i.e.

$$count([\![f(\omega)(I)]\!]) \leq 1,$$

where $[\![expr]\!](I)$ is the result of evaluation $expr$ on the instance I.

Example 5.1. *Over S_1 the following XFDs can be defined:*

$f_1(oid) = /db/part[supplier/offer/oId],$
$f_2(oid) = /db/part[supplier/offer/oId]/pId,$
$f_3(pid, sid) = /db/part[pId]/supplier/$
$\quad offer[sId]/delivPlace,$
$f_4(delivPlace) = /db/part/supplier/offer[$
$\quad delivPlace]/sId,$
$f_5(pid, sid) = /db/part[pId]/supplier[offer/sId].$

According to XPath semantics [27] the expression $f_1(oid)$ by a valuation ω, is evaluated against the instance I_1 (Figure 3) as follows: (1) first, a sequence of nodes of type $/db/part$ is chosen; (2) next, for each selected node the predicate $[supplier/offer/oId = \omega(oid)]$ is tested, this predicate is true in a node n, if there exists a path of type $supplier/offer/oId$ in I_1 leading from n to a text node with the value $\omega(oid)$. We see that $count([\![f_1(\omega(oid))]\!])$ equals 1 for all four valuations satisfied by I_1, i.e. for $oid \mapsto o_1$, $oid \mapsto o_2$, $oid \mapsto o_3$, and $oid \mapsto o_4$.

Similarly, execution of $f_2(oid)(I)$ gives a singleton for any valuation of oid. These singletons are text values of the path $/db/part/pId$, where only nodes satisfying the predicate $[supplier/offer/oId = \omega(oid)]$ are taken from the set of nodes determined by $/db/part$. So, also this XFD is satisfied by I_1.

However, none of the following XFDs is satisfied in I_1:
$g_1(sid) = /db/part[supplier/offer/sId]$,
$g_2(pid) = /db/part[pId]/supplier/offer/sId$
$g_3(pid) = /db/part[pId]/supplier/offer$
$g_4(pid, sid) = /db/part[pId]/supplier/offer[sId]$,
$g_5(delivPlace) = /db/part/pId/supplier/$
$offer[delivPlace]$.

Evaluating the above XFDs against I_1, we obtain, for example:
$count([\![g_1(sid)([sid \mapsto s1])(I_1)]\!]) = 2$,
$count([\![g_2(pid)([pid \mapsto p1])]\!]) = 2$,
$count([\![g_3(pid)([pid \mapsto p1])]\!]) = 3$.

An XFD can determine functional relationship between a tuple of text values of a given tuple of paths and a path denoting either a text value (e.g. $f_2(oid)$) or a subtree (a node being the root of the subtree) (e.g. $f_1(oid)$) Ű the latter XFDs will be referred to as *XML keys*.

Definition 5.3. *A functional dependence $f = \phi(A_1, \ldots, A_k)$, where f is of type q/l and l is a non-terminal label, is an XML key for subtrees of the type q/l.*

Another notation for XML keys has been proposed in [7]. In that notation

$$(db.part, \{pId\})$$

is an *absolute key* saying that a subtree of type $db/part$ is uniquely determined by the path $db/part/pid$ (this constraint holds in the instance I_1 in Figure 3). In our XPath-oriented notation this key is the following XFD:

$$db/part[pid].$$

An example of a *relative key* ([7]) is

$$(db.part, (supplier, \{offer.sId\})),$$

that says that that in the context of $db/part$, a tree $supplier$ is determined by the path $offer/sId$. In our XPath-oriented notation this key is expressed as follows:

$$db/part[pid]/supplier[sid].$$

6 Normal form for XML

To eliminate redundancies in XML documents, some normal forms (XNF) for XML schemas have been proposed [1, 24, 4, 5]. In this paper we will define XNF in the spirit of BCNF defined for relational schemas. This approach was also used in [1].

Definition 6.1. *Let S be an XML schema and Σ be a set of XFDs defined over S. (S, Σ) is in XNF iff for any XFD $\phi(A_1, \ldots, A_k)/A \in (S, \Sigma)^+$, also $\phi(A_1, \ldots, A_k) \in (S, \Sigma)^+$,where $(S, \Sigma)^+$ is a set of XFDs being consequences of (S, Σ).*

In other word, if $\phi(A_1, \ldots, A_k)/A$ is an XFD for $q/l/A$, then $\phi(A_1, \ldots, A_k)$ is a key for q/l. It means that there is at most one subtree of type q/l for any different valuation of $(A_1, \ldots, A_k)$ in ϕ, if the value of a child A of q/l is determined by this valuation. In this way the redundancy, i.e. repetition of values in different subtrees of type q/l, is eliminated.

Let us consider the schema S_2 in Figure 5 and its instance I_2 in Figure 6.

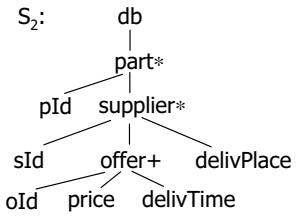


Figure 5: Restructured form of schema in Figure 1

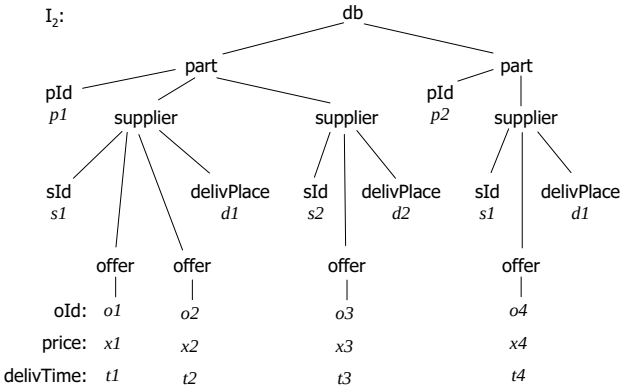


Figure 6: Instance of schema in Figure 5

The following XFD over S_2

$$/db/part/supplier[delivPlace]/sId$$

says that *delivery place* ($delivPlace$) determines the supplier (sId). However, S_2 is not in XNF, since its instance I_2 does not satisfy the key

$$/db/part/supplier[delivPlace].$$

It means that I_2 (Figure 6) is not free of redundancy (there are two different subtrees of type $supplier$ describing the same supplier, i.e. its possible name, address, etc.).

In the case of schema S_3 (Figure 7) the corresponding XFD and the key are:

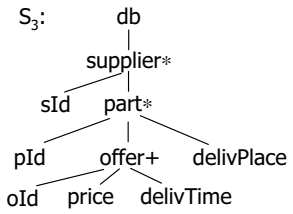


Figure 7: Restructured form of schema in Figure 1

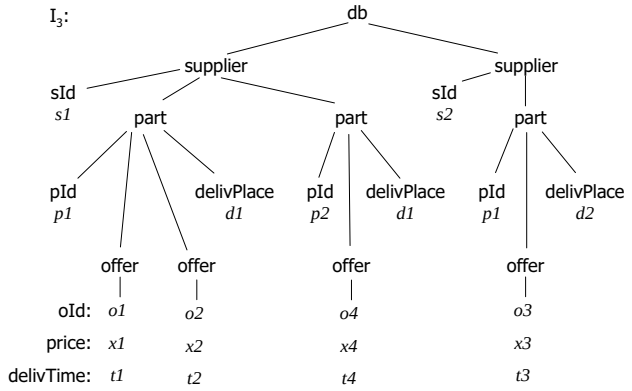


Figure 8: Instance of schema in Figure 7

$db/supplier[part/delivPlace]/sId$
$db/supplier[part/delivPlace]$.

These constraints are satisfied by I_3 (Figure 8). We see that this time, there is only one subtree of type $supplier$ for any value of $delivPlace$.

The other dependency of interest is $sId, pId \rightarrow delivPlace$. Its specification with respect to S_2 and S_3 is as follows:

$$db/part[pId]/supplier[sId]/delivPlace,$$

and

$$db/supplier[sId]/part[pId]/delivPlace.$$

It is easy to see then if these XFDs hold, then also keys

$$db/part[pId]/supplier[sId],$$

and

$$db/supplier[sId]/part[pId]$$

are satisfied.

However, neither S_2 nor S_3 is in XNF. We have already shown that there is redundancy in instances of S_2. Similarly, we see that also in instances of S_3 redundancies may occur. Indeed, since one part may be delivered by many suppliers then the description of one part may be multiplied under each supplier delivering this part, so such data as *part name*, *type*, *manufacturer* etc. will be stored many times. It results from the fact that satisfaction of $db/supplier/part[pid]/pname$ would not imply the satisfaction of $db/supplier/part[pid]$.

In Figure 9 there is schema S_4 that is in XNF. To make the example more illustrative, we added node $name$ to $part$ data. Also the instance in Figure 10 was slightly extended as compared to instances I_2 and I_3.

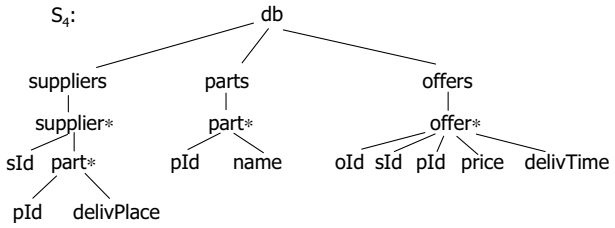


Figure 9: XNF schema for schemas S_1, S_2, and S_3

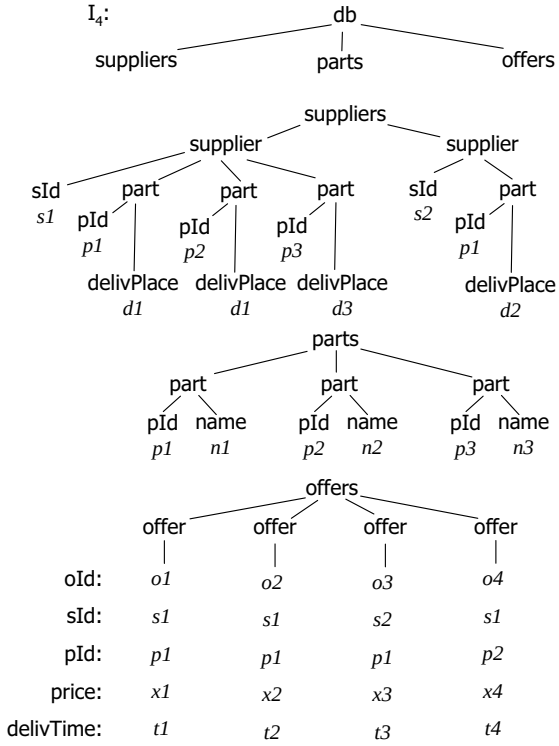


Figure 10: Instance of schema in Figure 9

XSD (XML Schema Definition) for S_4 in notation proposed in [26] is shown in Figure 11.

Note that we cannot use DTD since there are two subtrees labeled $part$, where each of them has different type: the $part$ subtree under $supplier$ consists of pId and $delivPlace$, whereas the $part$ subtree under $parts$ consists of pId and $name$. Recall that in the case of DTD each non-terminal symbol (label) can have only one type (definition), i.e. can appear on the left-hand side of exactly one production rule [26]. This difficulty might be overcame by introducing new labels (for example *partDesc*) for full descriptions of a parts.

It can be shown that S_4 satisfies the condition of XNF. Thus, this schema is both redundant-free and dependency

$$
\begin{array}{lcl}
db & \rightarrow & \mathrm{db}[content] \\
content & \rightarrow & \mathrm{suppliers}[suppliers], \mathrm{parts}[parts], \\
 & & \quad \mathrm{offers}[offers] \\
suppliers & \rightarrow & \mathrm{supplier}[supplier]* \\
parts & \rightarrow & \mathrm{parts}[part_1]* \\
offers & \rightarrow & \mathrm{offers}[offer]* \\
supplier & \rightarrow & \mathrm{sId}, \mathrm{part}[part_2] \\
part_2 & \rightarrow & \mathrm{pId}, \mathrm{delivPlace} \\
part_1 & \rightarrow & \mathrm{pId}, \mathrm{name} \\
offer & \rightarrow & \mathrm{oId}, \mathrm{sId}, \\
 & & \quad \mathrm{pId}, \mathrm{price}, \mathrm{delivTime}
\end{array}
$$

Figure 11: An XSD describing the XML schema in Figure 9

preserving. However, as we will show in the next section, schema S_4 is not the best XNF for the considered running example.

7 Transforming XML schemas to XML normal form

In the previous section we discussed an example of transforming an XML schema into XNF. We started with the schema S_1 in Figure 1, and the final schema was S_4 in Figure 9. However, the final schema has been created in a rather intuitive way. Thus, although it is in XNF it is not clear whether there exists another XNF reformulation of S_1, maybe better than X_4, or not. A systematic algorithm (the Decomposition Algorithm, DA) for normalizing an XML schema was proposed in [1]. If we apply DA to S_1, we obtain a schema that is worse than S_4. It is so due to the following reasons:

1. In DA we always obtain a normal form *relative* to a *context path*, i.e. the XNF is restricted to the subtree determined by this context path. Only if the context path is equal to the *root label* (db in our case), the XNF is *absolute*.

2. In DA it is not possible to change ordering of elements on a path, because we can only move and discard attributes or create new element types [1]. For example, the $part$ element precedes $supplier$ (is above it) in S_1 and in the result schema this ordering must remain unchanged. But then the absolute XNF for S_1 cannot be achieved.

3. The result of normalization strongly depends on the starting XML schema.

7.1 Transforming ER model to XNF

In this section we will discuss how to transform an ER (*Entity-Relationship*) schema [28] into XML schema in XML normal form (XNF). Our method is based on analyzing functional dependencies among attributes of entities involved in the ER schema.

In Figure 12 there is an ER schema corresponding to the XML schema considered in previous sections (see S_1 in Figure 1) with two additional attributes $sname$ (supplier name) and $pname$ (part name). $delivPlace$ is treated as an entity with the key attribute did.

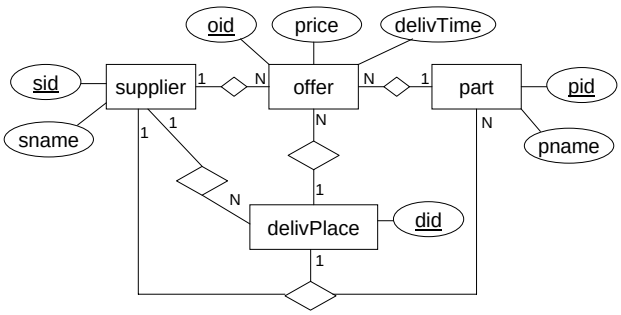


Figure 12: ER schema corresponding to S_1

The following functional dependencies are captured by the ER schema in Figure 12:

$$
\begin{array}{l}
oid \rightarrow sid, pid, did \\
sid, pid \rightarrow did \\
did \rightarrow sid \\
sid \rightarrow sname \\
pid \rightarrow pname \\
oid \rightarrow price, delivTime
\end{array}
\tag{1}
$$

The first three of them are of particular importance because they state constraints between entities (key attributes of entities).

We will proceed in two steps:

1. An initial XML schema is created using the entities names from the ER schema, their attributes and functional dependencies between key attributes. The initial schema must follow the *necessary condition* formulated in Theorem 7.3. Satisfaction of this condition is the prerequisite to obtain an XML schema in absolute XNF in the next step.

2. The DA algorithm [1] is applied to obtain the final XNF schema. In fact, only the step called *Creating New Element Types* from this algorithm is to be applied. In this way all violations caused by functional dependencies over non-key attributes (appearing on the right-hand sides of these dependencies) are resolved. Such functional dependencies in our example are for example the three last in (1).

Definition 7.1. *Let $\phi(A_1, \ldots, A_n)/B$ be an XFD of type $q/l/B$ over an XML schema S. S is called* XNF-consistent *with $\phi(A_1, \ldots, A_n)/B$, if for any instance I of S holds the implication:*

$$
\begin{array}{l}
I \models \phi(A_1, \ldots, A_n)/B \Rightarrow \\
merge_{q/l}(I) \models \phi(A_1, \ldots, A_n),
\end{array}
$$

where $merge_{q/l}(I)$ is the result of merging subtrees of type q/l in I.

Let I be an XML tree and $T = (r, \tau)$ be a subtree of type q/l of I, i.e. T belongs to the result of evaluation of the path expression q/l on the instance I, $T \in [\![q/l(I)]\!]$. Then r is a list of the form $r := (A_1 : a_1, \ldots, A_n : a_n)$, and τ is a list of subtrees (each of these lists may be empty).

Definition 7.2. *Two subtrees $T_1 = (r_1, \tau_1)$, and $T_2 = (r_2, \tau_2)$ of type q/l of an instance I are joinable if $r_1 = r_2$, and then:*

- *$(r, \tau_1) \bowtie (r, \tau_2) = (r, \tau_1||\tau_2)$, where $\tau_1||\tau_2$ is concatenation of lists τ_1 and τ_2;*
- *the merge operation on all subtrees of type q/l of an instance I is defined as follows:*
  $merge_{q/l}(I) =$
  for each $T_1, T_2 \in [\![q/l(I)]\!]$
  if T_1 **and** T_2 are joinable **then**
  $I := I - q/l[T_1] - q/l[T_2] \cup q/l[T_1 \bowtie T_2]$.

Example 7.1. *Let S be defined by*
$l \to l_1*$
$l_1 \to A_1\ B\ l_2*$
$l_2 \to A_2\ C$
and let terminal labels A_1, A_2 functionally determine B, i.e. $A_1, A_2 \to B$. Then the corresponding XFD is

$$\phi(A_1, A_2)/B := l/l_1[A_1, l_2[A_2]]/B.$$

We see that I (Figure 13) satisfies $\phi(A_1, A_2)/B$, i.e. for the values of the pair of paths $(l/l_1/A_1, l/l_1/l_2/A_2)$ the value of the path $l/l_1/B$ is uniquely determined. Similarly, the merged form of I, $merge_{l/l_1}(I)$, satisfies $\phi(A_1, A_2)$, i.e. the pair $(l/l_1/A_1, l/l_1/l_2/A_2)$ of path values uniquely determines the node of type l/l_1. Note that I in its unmerged form does not satisfy $\phi(A_1, A_2)$, because there are two nodes of type l/l_1 corresponding to the same pair of path values of the type $(l/l_1/A_1, l/l_1/l_2/A_2)$.

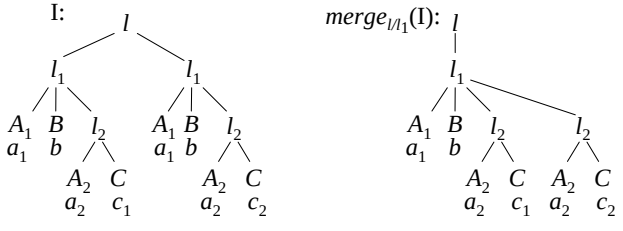


Figure 13: Instance I of the schema from Example 7.1 and its merged form

The following theorem formulates the necessary condition for XML schema to be XNF-consistent with an XDF defined over this schema.

Theorem 7.3. *Let $f := \phi(A_1, \ldots, A_n)/B$ be an XFD of type $q/l/B$ over an XML schema S. Then S is XNF-consistent with f if the set consisting of labels of all terminal children of q/l is functionally dependent on the set of terminal labels occurring in f.*

Proof. We will proof the theorem by contradiction. Let us assume that there exists a terminal child of l labeled C and C is not functionally dependent on the set $\{A_1, \ldots, A_n\}$ of terminal labels occurring in f. Then the tree of the form
$I = \phi(A_1 : a_1, \ldots, A_n : a_n)$
$\cup\{q/l[B : b, C : c_1], q/l[B : b, C : c_2]\}$,
is a subtree of an instance of S in which $\phi(A_1, \ldots, A_n)/B$ is satisfied. However, this subtree violates $\phi(A_1, \ldots, A_n)$. This violation follows from the fact that there are two subtrees rooted in q/l, one with terminal children $(B : b, C : c_1)$, and the other with terminal children $(B : b, C : c_2)$, so this subtrees cannot be merged. Thus S is not XDF-consistent with f. □

In the schema in the following example the necessary condition formulated in Theorem 7.3 does not hold.

Example 7.2. *Let S be defined by*
$l \to l_1*$
$l_1 \to A_1\ l_2*$
$l_2 \to A_2\ B\ C$
and let terminal labels A_1, A_2 functionally determine B, i.e. $A_1, A_2 \to B$. Then the corresponding XFD is

$$\phi(A_1, A_2)/B := l/l_1[A_1]/l_2[A_2]/B.$$

Assume that C does not depend on $\{A_1, A_2\}$. Schema S is not XNF-consistent with $\phi(A_1, A_2)/B$, since we have (see Figure 14):
$I = l/l_1(A_1 : a_1)/l_2(A_2 : a_2)$
$\cup\{l/l_1/l_2((B : b, C : c_1), (B : b, C : c_2))\}$
$= merge_{l/l_1/l_2}(I)$
Thus, I violates $\phi(A_1, A_2)$ before and after application of the merging operation, in both cases there are two nodes of type $l/l_1/l_2$ corresponding to the same pair of values (a_1, a_2) of the pair of paths $(l/l_1/A_1, l/l_1/l_2/A_2)$.

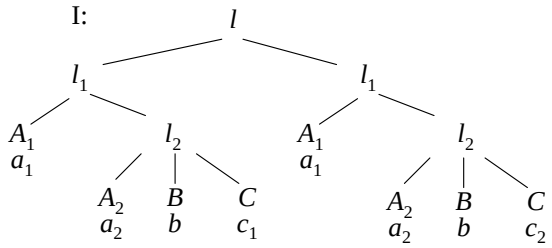


Figure 14: Instance of the schema from Example 7.2 that satisfies $\phi(A_1, A_2)/B$ and violates $\phi(A_1, A_2)$

Now, using the Theorem 7.3 and DA, the transformation of ER into XNF is realized in the following two steps:

1. We start with functional dependencies defined over key attributes of entities modeled by ER schema. Following the requirements of Theorem 7.3 we obtain the

following DTD for ER schema in Figure 12:

$$\begin{array}{l} db \rightarrow supplier* \\ supplier \rightarrow sid\ \ sname\ \ part* \\ part \rightarrow pid\ \ pname\ \ did\ \ offer* \\ offer \rightarrow oid\ \ price\ \ delivTime \end{array} \quad (2)$$

Now, the remaining functional dependencies specified in (1) must be taken into account.

2. The DTD obtained in the first step is the subject of the decomposition by means of the DA algorithm. It is easily seen that the XFD corresponding to $pid \rightarrow pname$ is anomalous. The step *Creating new element types* of DA converts (2) into

$$\begin{array}{l} db \rightarrow supplier*\ \ partDesc* \\ supplier \rightarrow sid\ \ sname\ \ part* \\ partDesc \rightarrow pid\ \ pname \\ part \rightarrow pid\ \ did\ \ offer* \\ offer \rightarrow oid\ \ price\ \ delivTime \end{array} \quad (3)$$

Applying the above steps to the ER schema from Figure 12 gives the XML schema in XNF depicted in Figure 15 and its instance in Figure 16.

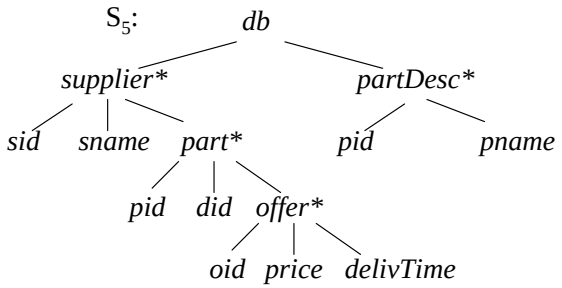


Figure 15: The result of transformation ER schema from Figure 12 to XML schema in XNF

8 Conclusion

In this paper, we discussed how the concept of database normalization can be used in the case of XML data. Normalization is commonly used to develop a relational schema free of unnecessary redundancies and preserving all data dependencies existing in application domain. In order to apply this approach to design XML schemas, we introduced a language for expressing XML functional dependencies. In fact, this language is a class of XPath expressions, so its syntax and semantics are defined precisely. We define the notion of satisfaction of XML functional dependence by an XML tree. To define XNF we use the approach proposed in [24].

All considerations are illustrated by the running example. We discuss various issues connected with normalization and compare them with issues faced in the case of relational databases. We show how to develop redundancy-free and dependency preserving XML schema. It is worth mentioning that the relational version of the schema cannot be structured in redundancy-free and dependency preserving form. In this case, preservation of all dependencies requires 3NF but then some redundancy is present. Further normalization to BCNF eliminates redundancies but does not preserve dependencies. In the case of XML, thanks to its hierarchical nature, we can achieve both properties. However, it is not clear if this is true in all cases (see e.g. [5].

We have proposed a method for normalizing XML data in to steps. First, we build a conceptual model by means of ER schema and specify all functional dependencies among its attributes. Following the necessary condition formulated in Theorem 7.3, an initial XML schema is created. This schema is XNF-consistent with all XML functional dependencies under consideration. Such the schema can be further normalized, for example using the decomposition algorithm (DA) [1]. It was shown that in the presence of cyclic functional dependencies the procedure proposed in DA results in bad design (only a local XNF can be obtained, i.e. only subschemas of the schema can be transformed into XNF).

Acknowledgement: This work was supported in part by the Polish Ministry of Science and Higher Education under grant N N516 369536.

References

[1] M. Arenas and L. Libkin, "A normal form for XML documents." *ACM Trans. Database Syst.*, vol. 29, pp. 195–232, 2004.

[2] E. Codd, "Further normalization of the data base relational model," *R. Rustin (ed.): Database Systems, Prentice Hall, and IBM Research Report RJ 909*, pp. 33–64, 1972.

[3] E. F. Codd, "Recent investigations in relational data base systems," in *IFIP Congress*, 1974, pp. 1017–1021.

[4] S. Kolahi, "Dependency-Preserving Normalization of Relational and XML Data," *Database Programming Languages, DBPL 2005, Lecture Notes in Computer Science*, vol. 3774, pp. 247–261, 2005.

[5] ——, "Dependency-preserving normalization of relational and XML data," *Journal of Computer and System Sciences*, vol. 73, no. 4, pp. 636–647, 2007.

[6] S. Kolahi and L. Libkin, "On redundancy vs dependency preservation in normalization: an information-theoretic study of 3NF," in *PODS '06*. ACM, New York, USA, 2006, pp. 114–123.

[7] P. Buneman, S. B. Davidson, W. Fan, C. S. Hara, and W. C. Tan, "Reasoning about keys for XML," *Information Systems*, vol. 28, no. 8, pp. 1037–1063, 2003.

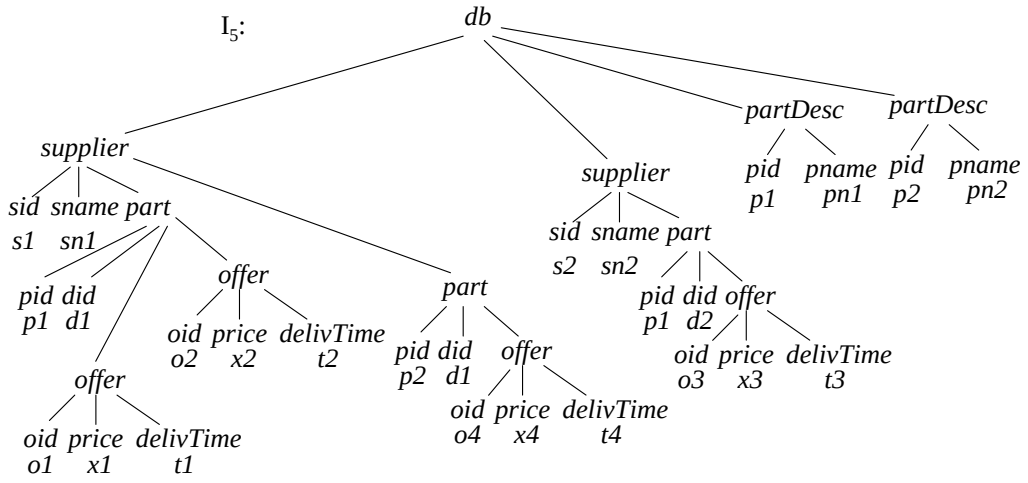


Figure 16: Instance of S_5 from Figure 15

[8] W. Fan and J. Siméon, "Integrity constraints for XML," *J. Comput. Syst. Sci.*, vol. 66, no. 1, pp. 254–291, 2003.

[9] M. W. Vincent, J. Liu, and M. K. Mohania, "On the equivalence between FDs in XML and FDs in relations," *Acta Inf.*, vol. 44, no. 3-4, pp. 207–247, 2007.

[10] M. W. Vincent and J. Liu, "Checking functional dependency satisfaction in XML," *Database and XML Technologies – XSym 2005, Lecture Notes in Computer Science*, vol. 3671, pp. 4–17, 2005.

[11] C. Yu and H. V. Jagadish, "XML schema refinement through redundancy detection and normalization," *VLDB Journal*, vol. 17, no. 2, pp. 203–223, 2008.

[12] R. Elmasri and S. B. Navathe, *Fundamentals of Database Systems*. Redwood City: The Benjamin/Cummings, 1994.

[13] S. Abiteboul, R. Hull, and V. Vianu, *Foundations of Databases*. Reading, Massachusetts: Addison-Wesley, 1995.

[14] M. Arenas and L. Libkin, "An information-theoretic approach to normal forms for relational and XML data," *J. ACM*, vol. 52, no. 2, pp. 246–283, 2005.

[15] P. A. Boncz, T. Grust, M. van Keulen, S. Manegold, J. Rittinger, and J. Teubner, "Pathfinder: XQuery - the relational way," in *VLDB 2005*. ACM, 2005, pp. 1322–1325.

[16] S. Pal, I. Cseri, O. Seeliger, M. Rys, G. Schaller, W. Yu, D. Tomic, A. Baras, B. Berg, D. Churin, and E. Kogan, "XQuery implementation in a relational database system," in *VLDB 2005*. ACM, 2005, pp. 1175–1186.

[17] P. Pigozzo and E. Quintarelli, "An algorithm for generating XML Schemas from ER Schemas," in *SEBD*, 2005, pp. 192–199.

[18] R. Schroeder and R. dos Santos Mello, "Improving query performance on XML documents: a workload-driven design approach," in *DocEng*. ACM, 2008, pp. 177–186.

[19] W. Xu and Z. M. Özsoyoglu, "Rewriting XPath queries using materialized views," in *VLDB 2005*, 2005, pp. 121–132.

[20] T. Pankowski, "XML data integration in SixP2P – a theoretical framework," in *EDBT Workshop Data Management in P2P Systems (DAMAP 2008),*ACM Digital Library, 2008, pp. 11–18.

[21] W. W. Armstrong, "Dependency structures of data base relationships," in *IFIP Congress*, 1974, pp. 580–583.

[22] P. A. Bernstein, "Synthesizing third normal form relations from functional dependencies," *ACM Transactions on Database Systtems*, vol. 1, no. 4, pp. 277–298, 1976.

[23] J. Rissanen, "Independent components of relations," *ACM Transactions on Database Systems*, vol. 2, no. 4, pp. 317–325, 1977.

[24] M. Arenas, "Normalization theory for XML," *SIGMOD Record*, vol. 35, no. 4, pp. 57–64, 2006.

[25] XML Schema Definition Language (XSDL) 1.1, 2007, www.w3.org/TR/xmlschema11-1.

[26] W. Martens, F. Neven, and T. Schwentick, "Simple off the shelf abstractions for XML schema," *SIGMOD Record*, vol. 36, no. 3, pp. 15–22, 2007.

[27] XML Path Language (XPath) 2.0, 2006, www.w3.org/TR/xpath20.

[28] P. P. Chen, "The entity-relationship model - toward a unified view of data," *ACM Transactions on Database Systtems*, vol. 1, no. 1, pp. 9–36, 1976.

Realization of UML Class and State Machine Models in the C# Code Generation and Execution Framework

Anna Derezińska and Romuald Pilitowski
Institute of Computer Science, Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warsaw, Poland
E-mail: A.Derezinska@ii.pw.eu.pl, http://ii.pw.edu.pl/~adr

Keywords: state machine, UML, code generation, MDE, C#

Received: April 3, 2009

Many benefits are expected due to usage of code generation tools. A reliable application should be created effectively based on complex structural and behavioral models. Model driven approach for program development is realized in Framework for eXecutable UML (FXU). The tool transforms UML models into C# source code and supports execution of the application reflecting the behavioral model. The framework consists of two components: code generator and run time library. The generated and executed code corresponds to structural model specified in class diagrams and behavioral model described by state machines of these classes. All single concepts of behavioral state machines included in the UML 2.x specification are taken into account, including all kinds of events, states, pseudostates, submachines etc. The paper discusses the transformation of UML state machines into C# language. It presents checking the correctness of classes and state machines decided in the framework in order to run a model-related and high quality C# application. The solution was tested on set of UML models.

Povzetek: Predstavljeno je orodje za avtomatsko generacijo kode iz UML v C#.

1 Introduction

Model Driven Engineering (MDE) represents software development approaches in which creation and manipulation of models should result in building of an executable system [1]. There are two general directions towards model execution. The first one is aimed at the direct model execution in a virtual machine of a modelling notation. This idea resulted in the development of the Foundation Subset for Executable UML Models specification (FUML [2]). It defines a basic virtual machine for UML.

The second trend assumes transformation of a model into possibly more refined models, and finally into a target code. The output code is usually expressed in a general purpose language. It can be further modified, completed and used for building a final application, with commonly used development environments. In this paper we discuss problems concerning building an executable C# application from UML classes and state machines.

Industrial product development puts a lot of attention on fast implementation of needed functionalities. Model-driven approach to program development offers a promising solution to these problems. Complex behavioral models can be designed and verified at early stages of the whole product creation cycle and automatically transformed into the code preserving the desired behavior.

State machines, also in the form of statecharts incorporated in the UML notation [3], are a widely used concept for specification of concurrent reactive systems. Proposal for execution of behavioral UML models suffers from the problem that no generally accepted formal semantics of UML models is available. Therefore, validation of UML transformation and model behavior depicted in the resulting code is difficult. Rather than completely formalizing UML models, we try to deal with selected aspects of the models.

Inconsistency and incompleteness allowed by UML can be a source of problems in software development. A basic type of design faults is concerned with the well-formedness of diagrams [3]. Typically, completeness of a design requires that model elements are specified with their features and usage of one element can imply a usage of another, directly related model element. In the current modeling CASE tools some completeness conditions can be assured automatically (e.g., default names of roles in associations, attributes, operations etc.). Incompleteness of models can be strongly related to their inconsistency, because it is often impossible to conclude whether diagrams are inconsistent or incomplete [4]. Therefore, within this paper we will refer to model defects as to correctness issues.

The Framework for eXecutable UML (FXU) offers a foundation for applying MDA ideas in automation of software design and verification [5]. The FXU framework was the first solution that supported generation and execution of all elements of behavioral state machine UML 2.0 using C# language. In order to build an application reflecting the modeled classes and

their behaviors specified by state machines, we resolved necessary semantic variation points [6]. Semantic variation points are aspects that were intentionally not determined in the specification [3] and its interpretation is left for a user.

It was also necessary to provide some correctness checking of a model. This paper is devoted to these issues. To present potential problems we selected one target application environment, i.e., creation of application in C# language. The verification of an input UML model is based on a set of hard coded rules. Some of the rules are general and can be applied for any object-oriented language, as they originate directly from the UML specification [3]. Other rules depend on the programming environment because they take also into account the features of the target language - C#. The verification is performed during transformation of class and state machine models into the corresponding code; it is so-called static verification. Other set of rules is used during execution of the code corresponding to given state machines; so-called dynamic verification. For all correctness rules the appropriate reaction on the detected flaws were specified.

One of the contributions of the paper is exploitation of C# constructs to create concise representation of state machines, including also all complex concepts of UML behavioral state machines. Additionally, the correctness rules for UML models are presented, aimed at executing class and state machine models as C# applications.

In the next section we discuss the related works. Next, the FXU framework, especially solutions used for state machines realization, will be presented. In Sec. 4 we introduce correctness issues identified in the transformation process and during execution of state machines. Remarks about experiments performed and the conclusions finish the paper.

2 Related work

2.1 Code generation and execution support

There are different policies dealing with UML models to be transformed. Transformation of a model into the corresponding target code can be realized for any general UML model. The main restrictions concern model correctness but not the direct correspondence to any target notation. Many code generators incorporated in modeling tools, and also the FXU framework, support this approach. It helps dealing with not complete and not specialized models, which often encounter in software development and evolution praxis.

An opposite strategy is the refinement of a model towards the concepts of the target notation, which can be a programming language. This refinement can be completed, for example, using a set of stereotypes included in a UML profile dedicated for the considered notation. This approach is represented by IBM Rational modeling extension for Microsoft .NET [7]. However, it should be noted that the tool supports only selected C# concepts and the relations between refined model elements are not validated. Moreover, state machines are not taken into account in code generation.

Many modeling tools have a facility of transforming models into code in different programming languages. However, the most of them consider only class models. We compared functionality of twelve tools that could also generate code from behavioral state machines. Only few of them took into account more complex features of state machines, like choice pseudostates, deep and shallow history pseudostates, deferred events or internal transitions. The most complete support for state machines UML 2.0 is implemented in the Rhapsody tool [8] of IBM Telelogic (formerly I-Logix). However it does not consider C# language.

There exist different approaches to building an executable application basing on behavioral UML models. In the first one, the code created as the target of model transformation includes the mapping of the state machine structure as well as the logic supporting model execution [9, 10]. Therefore, large number of code must be generated even for simple state diagrams. All semantic issues have to be resolved directly in the generated code.

Another solution is usage of a kind of a run-time environment. It assumes an existence of a library or virtual machine that provides an engine for state machine execution [8, 11, 12]. The generated code depicts only the structure of the input state machine. The code is more compact and easier to understand and to modify. The FXU framework is based on the second solution, applying a run-time library.

2.2 State machine semantics

A huge amount of research efforts is devoted to formalization of UML models, specification of their semantics and verification methods [13]-[17]. However they are usually not resolving the practical problems which are faced while building an executable code, because of many variation semantic points of the UML specification.

An attempt for incorporation of different variation points into one solution is presented in [18]. The authors intend to build models that specify different variants and combine them with the statechart metamodel. Different policies should be implemented for these variants.

The semantics defined in the FUML specification [2] are generally a precise definition of a subset of the UML semantics given in the UML 2.2 Superstructure Specification. The FUML specification is limited to the selected UML elements considered as mostly used. Therefore it does not deal, for example, with all features of state machines.

2.3 Model correctness

Our work relates also to the field of correctness of UML models. The consistency problems in UML designs were extensively studied in many papers. It could be mentioned workshops co-located to the Models (former UML) series of conferences, and other works [4, 19-22].

Checking of models is important in Model Driven Architecture (MDA) approaches [23, 24] where new diagrams and code are automatically synthesized from the initial UML model: all the constructed artifacts would inherit the initial inconsistency [19].

Current UML case tools allow constructing incorrect models. They provide partial checking of selected model features, but it is not sufficient if we would like to create automatically a reliable application. More comprehensive checking can be found in the tools aimed at model analysis. For example, the OO design measurement tool SDMetrics [25] gives the rules according to which the models are checked. We used the experiences of the tool (Sec. 4), but it deals neither with state machine execution nor with C# language.

The consistency problems remain also using tools for building executable UML models [26-28]. Different subsets of UML being used and we cannot assure that two interchanged models will behave in the same way.

Solutions to consistency problems in class diagrams were presented in [29]. The problem refers to constrains specifying generalization sets in class diagram, which is still not commonly used in most of UML designs.

An interesting investigation about defects in industrial projects can be found in [30]. However the study takes into account only class diagrams, sequence diagrams and use case diagrams. It discusses mostly relations among elements from different diagram types. The state machines were not considered.

3 Code generation and execution in FXU

Transformation of UML models into executable application can be realized in the following steps.

1. A model, created using a CASE modeling tool, is exported and saved as an XML Metadata Interchange (XMI) file.
2. The model (or its parts) is transformed by a generator that creates a corresponding code in the target programming language.
3. The generated code is modified (if necessary), compiled and linked against a Runtime Library. The Runtime Library contains realization of different UML meta-model elements, especially referring to behavioral UML models.
4. The final application, reflecting the model behavior, can be executed.

It should be noted, that steps 1) and 2) can be merged, if the considered code generator is associated with the modelling tool.

The process presented above is realized in the FXU framework [5]. The target implementation language is C#. The part of UML model taken into account comprises classes and behavioral state machines. Protocol state machines are not considered.

The FXU framework consists of two components - FXU Generator and FXU Runtime Library. The Generator is responsible for realization of step 2. The FXU Runtime Library includes over forty classes that correspond to different elements of UML state machines. It implements the general rules of state machine behavior, independent of a considered model, e.g., processing of events, execution of transitions, entering and exiting states, realization of different pseudostates. It is also responsible for the runtime verification of certain features of an executed model.

3.1 Model transformation

Transforming class models into C# code, all model elements are implemented by appropriate C# elements. Principles of code generation from class models are similar to other object-oriented languages and analogues to solutions used in other tools. It is not so straightforward for state machine models.

State machines can be used at different levels of abstraction as behavioral state machines or protocol state machines. Protocol state machines are intended to model protocols. Behavioral state machines specify behavior of various model elements, like a class, a component, an operation. These elements constitute a context of a machine.

The primary application of behavioral state machine in an object-oriented model is description of a class. A class can have attributes keeping information about a current state of an object. Classes have operations that can trigger transitions, send and receive events. The FXU framework is limited to the most typical case, when a behavioral state machine models behavior of class instances. Model elements available in the context of the class are also available in the state machine.

A distinctive feature of FXU is dealing with all UML elements of behavioral state machines and their realization in C# application. Therefore we present selected concepts of state machines with their implementation in C#. We point out different C# specific mechanisms used in the generated application. Using selected solutions we would like to obtain an efficient and reliable application.

For any state machine of a class, a new attribute of *StateMachine* type is created. The structure of the state machine is build in a method of the class - *InitFXU()*. States, pseudostates, regions, transitions and events are created as local variables of the method.

Any state can have up to three types of internal activities *do, entry, exit.* The activities of a state are realized using a delegate mechanism of C#. Three methods *DoBody, EntryBody* and *ExitBody* with empty bodies are created for any state by default. If an activity exists a corresponding method with its body is created, using information taken from the model. Applying delegate mechanism allows defining the methods for states without using of inheritance or overloaded methods. Therefore the generated code can be simple, and generation of a class for any single state can be avoided. A state machine is not generated as a design pattern - "state" [31]. In the state design pattern, a single class is created for any state and any substate; and we would like to prevent an explosion of number of classes.

Signals, in opposite to other elements like states or events, are not created as local variables of the

initialisation method. They are created as classes, because they can be generalized and specialized building a signals hierarchy. If a certain signal can trigger an event also all signals that are its descendants in the signal hierarchy can trigger the same event. This feature of signals was implemented using the reflection mechanism of C# [32].

Three transition kinds can be specified for a transition, *external, internal* and *local* transitions. Triggering an internal transition implies no change of a state, exit and entry activities are not invoked. If an external transition is triggered it will exit its source state (a composite one), i.e. its exit activity will be executed. A local transition is a transition within a composite state. No exit for the composite (source) state will be invoked, but the appropriate exits and entries of the substates included in the state will be executed.

A kind of a transition can be specified in a model, but in praxis this information is rarely updated and often inaccurate. Therefore we assumed that in case of composite states a kind of generated transition is determined using a following heuristics:

- If the target state is different than the source state of a transition and the source state is a composite state, the transition is external.
- Else, if the transition is defined in a model as internal it is treated as an internal transition.
- Otherwise, the transition is local.

A transition can have its guard condition and actions. They are created similarly to activities in states, using delegate mechanism of C#. If a body of an appropriate guard condition or action is nonempty in a model, it is put in the generated code. It should be noted that verification of logical conditions written in C# is postponed to the compilation time.

Events should have some identifiers in order to be managed. Change events and call events are identified by unique natural numbers assigned to the events. A time event is identified by a transition which can be triggered by this event. A completion event is identified by a state in which the event was generated. Finally, for a signal event the class of the signal, i.e., its type, is used as its identifier.

There are some elements of a UML model that include a description in a form not precisely specified in the standard, but dependent on a selected notation, usually a programming language. There are, for example, guard conditions, implementation of actions in transitions or in states, body of operations in classes. They can be written directly in a target implementation language (e.g., C#). During code generation these fragments are inserted into the final code. Verification of the syntax and semantics of such code extracts is performed during the code compilation and execution according to a selected programming language.

3.2 Model example

Fragments of an exemplary UML model are shown in Fig. 1. *Runway* class belongs to an airport control system. Selected attributes, operations and a state machine of the class are presented. The state machine describes different states of the runway. A runway can be opened, closed or deleted. State *deleted* is simple; two remaining states are composite ones. An opened runway can be either free or occupied. A runway can be closed due to temporary maintenance or emergency. Complex state *closed* consists of simple state *temporaryMainenance,* simple state *preparation* and complex state *restoration* including two orthogonal regions.

In guard conditions and triggers, the operations and attributes of the class are used. Several *entry* and *do* activities are omitted due to legibility reasons.

Using an FXU template the resulting programming class can be created for the *Runway* class and its behavioral model. Extracts of the C# code corresponding to the example and created by the FXU generator are given in the Appendix.

The *StateMachine* attribute of the *Runway* class defines the structure and features of its state machine. Except of methods implementing operations modelled in the class, the class has also two additional methods *InitFXU* and *StartFXU*. The *InitFXU* method is responsible for creation and initialization of all objects corresponding to all elements of state machine(s) associated with the class, such as regions, states, pseudostates, transitions, activities, events, triggers, guards, actions, etc. Bodies of *entry, do, exit* activities, guard conditions and actions are implemented with delegates. The *StartFXU* method is used for launching the behavior of the state machine.

3.3 Model execution

The structure of basic elements of the FXU Runtime Library corresponds to the simplified state machine meta-model (Fig. 2). A vertex of a state machine graph is handled as a state or a pseudostate. A specialized state can be a state machine. Any transition is defined by its source and destination vertices. A transition can be triggered by an event. Classes of all events are a direct specialization of the base class *Event*. Meta-model class *MessageEvent* was omitted, because it is an intermediate level abstract class and was not necessary to perform any tasks. An additional class *CompletitionEvent* is responsible for dealing with completion events. They are triggered once the entry actions, *do* activities and activities of the internal elements have been completed. The completion event can be triggered just after entering the state if there are no activities and no internal elements.

Event processing during state machine execution is performed according to the rules given in the UML specification [3]. The processing of a single event occurrence by a state machine is interpreted as a *run-to-completion step.* Before initiation and after completion of a step, a state machine is in a stable configuration. All *entry, exit* or internal activities of all states (complex and nested states) are completed, but *do* activities can last.

Basic algorithms of FXU realization, like execution of a state machine, entry to a state, exit from a state, were

presented in [5]. For every state a queue was implemented that pools incoming events (Fig. 2). Event pool is served by a producers-consumer algorithm.

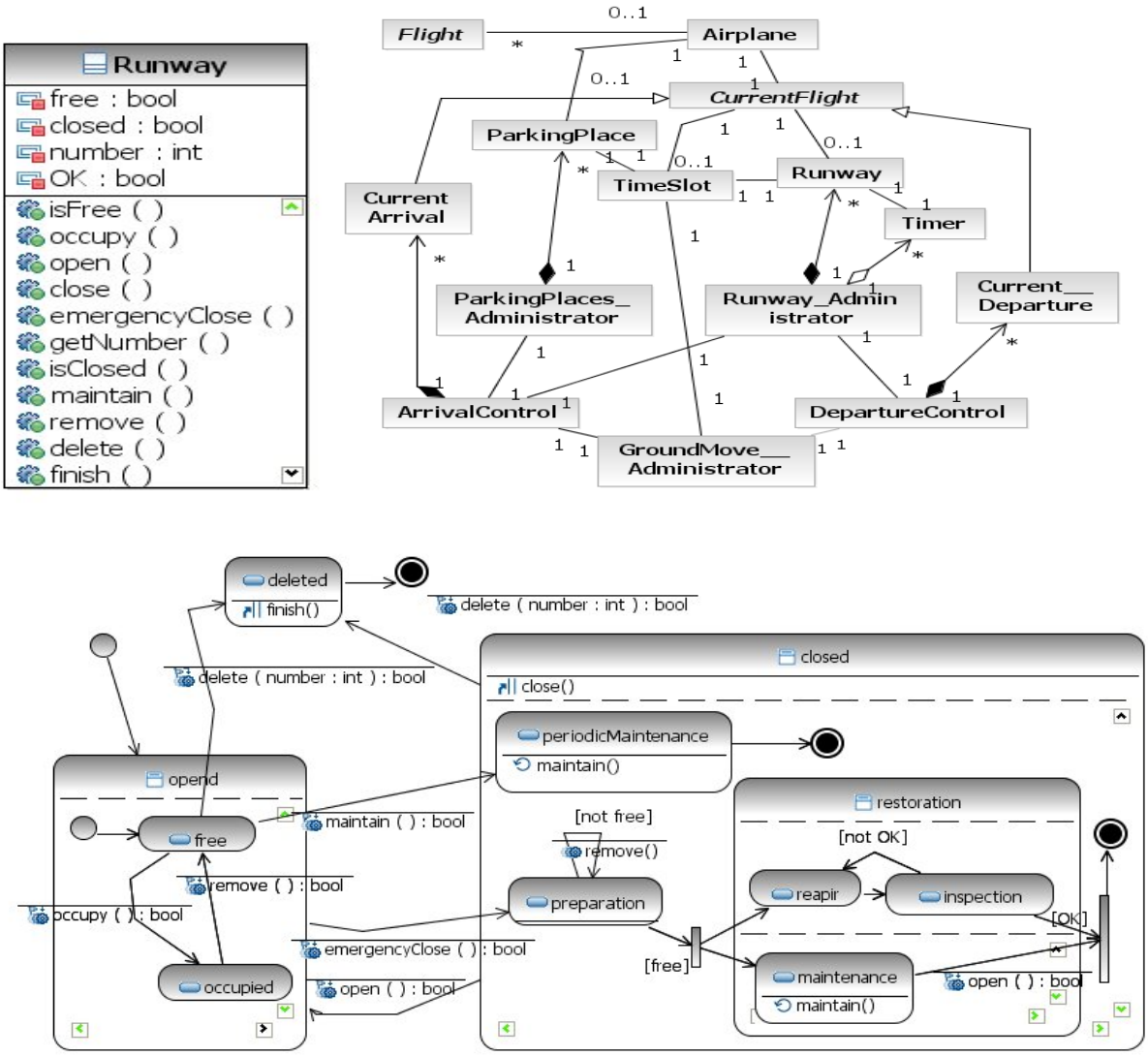


Figure 1: Example – *Runway* class, its state machine and class model

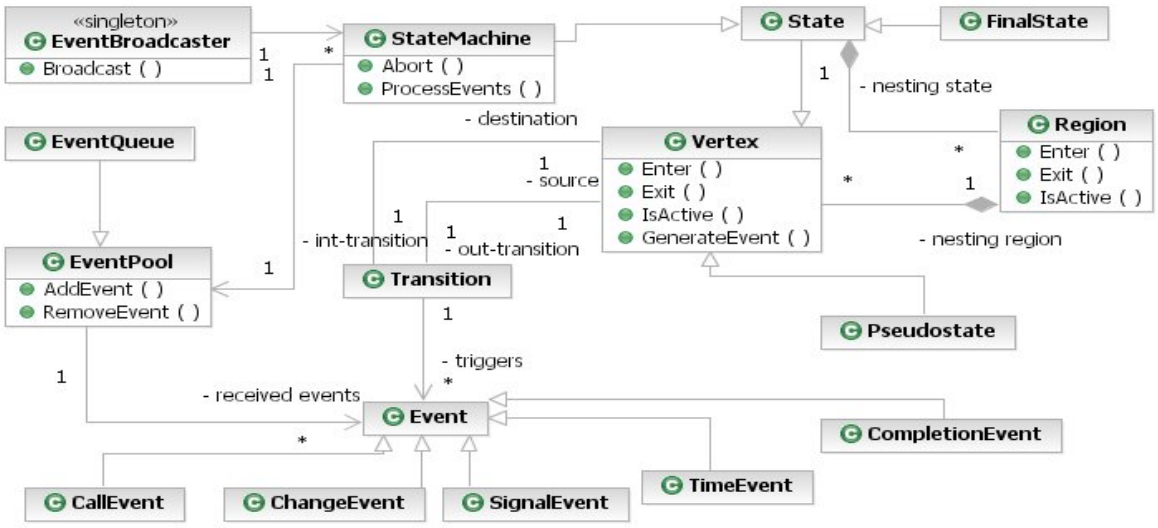


Figure 2: FXU Runtime Library - statemachines and event processing.

Events can be broadcasted or sent directly to the selected state machines. Events trigger transitions that have an active source state and their guard conditions evaluate to true. Transitions to be fired are determined as *the maximal set of non-conflicting transitions* [3]. If many transitions can be fired, transition priorities are used for their selection and resolve some transition conflicts. According to the specification, the priorities of conflicting transitions are based on their relative position in the state hierarchy. A transition originating from a substate has higher priority than a conflicting transition originating from any of its containing states.

Using this priority definition not all transition conflicts are resolved in case many transitions can be fired. Therefore, we had proposed and implemented an extended definition of transitions priority. We obtained one unique set of non-conflicting transitions in any situation. The detailed algorithm of selecting non-conflicting transitions and the extended firing priorities can be found in [6]. Also resolving of other variation points, especially dealing with entering and exiting orthogonal states, is shown in [6].

Interpreting different concepts of state machines we can use parallel execution. In the FXU RunTime Library it is implemented by multithreading. Multithreading is used for processing of many state machines which are active in the same time, e.g., state machines of different classes. It is used also for handling submachine states and orthogonal regions working within states, and for other processing of events. In the Appendix, examples of an output trace generated during execution of an application created from the model (Fig. 1) are shown. Different threads, which were created to deal with encountering events, are identified by number in brackets. For example, realization of a transition from the pseudostate fork to substate *maintenance* launched thread "[14]". Thread "[15]" was created to implement a transition from the fork pseudostate to substate *repair*. In other execution runs of the application, the numbers and ordering of the threads can be different.

4 Checking of model correctness

While generating valid C# code from UML class and state machine diagrams the certain conditions should be satisfied. There are many possible shortcomings of models that are not excluded by modeling tools, or should be not prohibited due to possible model incompleteness at different evolution stages. They were analyzed taking into account the practical weaknesses of model developers.

The prepared correctness rules were based on three main sources: the specification of UML [3], the rules discussed in related works and other comparable tools, in particular in [25], and finally our own study, especially taking into account the features of C# language - the target of the model transformation [32].

Various shortcomings can be detected during different steps of application realization (Sec. 3). Many of them can be identified directly in the model, and therefore detected during model to code transformation step (step 2). Verification of such problems will be called static, as it corresponds to an automated inspection of a model. Other flaws are detected only during execution of the resulting application (step 4). Such dynamic verification will be completed by the appropriate classes of the FXU Runtime Library.

In tables I-III defects identified in classes and state machines are presented. The last column shows severity associated to the shortcomings. Three classes of severity are distinguished. If a defect detected in a model is called as *critical* the model is treated as invalid and the code

Table I
Defects detected in UML class diagrams (static)

No	Detected defects	Reaction	Severity
1	A generalization of an interface from a class was detected	Stop code generation	critical
2	A name of an element to be generated (e.g. a class, an operation, an attribute) is a keyword of C# language	Stop code generation	critical
3	A class relates via generalization to more than one general class	Stop code generation	critical
4	A cycle in class generalization was detected	Stop code generation	critical
5	A name of an element to be generated is missing	Generate the element pattern without its name. The element name has to be supplemented in the generated code	medium
6	A name of an element to be generated is not a valid C# name. It is assumed that white characters are so common shortcoming that they should be automatically substituted by an underline character	As above	medium
7	An interface visibility is *private* or *protected*	Use *package* visibility	low
8	A class visibility is *private* or *protected*.	Use *package* visibility	low
9	An interface is *abstract*	Treat the interface as no abstract	low
10	An interface has some attributes	Ignore attributes of the interface	low
11	An interface has nested classes	Ignore classes nested in the interface	low
12	A class that is no *abstract* has abstract operations	Treat the class as *abstract*	low

generation is interrupted without producing the output. Later cases are classified as *medium* and *low*. In both cases the code generation is proceeded, although for *medium* severity it can require corrections before compilation. In all cases information about all detected shortcomings is delivered to a user. A detailed reaction to the found defect is described in the third column. While assigning severity levels and reactions to given defects we took into account general model correctness features but also requirements specific for C# applications.

4.1 Verification of class models

Class diagrams describe a static structure of a system, therefore many their features can be verified statically before code generation. Table I summaries defects that are checked during static analysis of UML class models. It was assumed that some improvements can be added more conveniently in the generated code than in a model. The class models can be incomplete to some extent and we can still generate the code. Admission of certain model incompleteness can be practically justifiable because of model evolution.

It should be noted that not all requirements of generated code are checked by the generator. Some elements are verified later by the compiler. It concerns especially elements that are not directly defined by the UML specification, like bodies of operations.

4.2 Verification of state machines

Similarly to class diagrams, different defects of state machines can be detected statically in the models. They are listed in Tab. II. Static detection of shortcomings in state machines is realized twice. First, it is made before model to source transformation (step 2). Second correctness checking is fulfilled before state machine execution. It is a part of step 4, during the initialization of the structure of a state machine.

For example, a static verification can be illustrated using a state machine from Fig. 1. Transition outgoing state *maintenace* has an event trigger - calling of an

Table II.
Defects detected in UML state machines (static)

No	Detected defects	Reaction	Severity
1	A cycle in signal generalization was detected	Stop code generation	critical
2	A signal inherits after an element that is not another signal	Stop code generation	critical
3	A signal relates via generalization to more than one general signal	Stop code generation	critical
4	A region has more than one initial pseudostate	Stop code generation	critical
5	A state has more than one deep history pseudostate or shallow history pseudostate	Stop code generation	critical
6	There are transitions from pseudostates to the same pseudostates (different than a choice pseudostate)	Stop code generation	critical
7	There are improper transitions between orthogonal regions	Stop code generation	critical
8	A transition trigger refers to an nonexistent signal	Stop code generation	critical
9	An entry point, join or initial pseudostate has no incoming transition or more than one incoming transition	Stop code generation	critical
10	A deep or shallow history pseudostate has more than one outgoing transition	Stop code generation	critical
11	A transition from an entry/exit point to an entry/exit point	Stop code generation	critical
12	An exit point has no any incoming transition	Stop code generation	critical
13	Transitions outgoing a fork pseudostate do not target states in different regions of an orthogonal states	Stop code generation	critical
14	Transitions incoming to a join pseudostate do not originate in different regions of an orthogonal state	Stop code generation	critical
15	There is a transition originating in an initial pseudostate or a deep/shallow history pseudostate and outgoing a nested orthogonal state	Stop code generation	critical
16	The region at the topmost level (region of a state machine) has no initial pseudostate	Warn a user	medium
17	A transition outgoing a pseudostate has a trigger	Ignore the trigger	medium
18	A transition outgoing a pseudostate (different from a choice or junction vertex) has a nonempty guard condition	Ignore the guard condition	medium
19	A transition targeting a join pseudostate has a trigger or nonempty guard condition	Ignore the trigger and/or condition	medium
20	A trigger refers to a non-existing operation	The transition will be generated but it cannot be triggered by this event	medium
21	A trigger refers to an abstract operation or to an operation of an interface	as above	medium
22	A time event is deferred	Treat the event as not being deferred	medium
23	A final state has an outgoing transition	Warn a user	medium
24	A terminate pseudostate has an outgoing transition	Warn a user	low

Table III.
Defects detected in UML state machines (dynamic)

No	Detected defects	Reaction	Severity
1	There is no enabled and no "else" transition outgoing a choice or junction pseudostate	Suspend execution - terminate	critical
2	A deep or shallow history pseudostate was entered that has no outgoing transitions and is "empty", i.e. either a final state was a last active substate or the state was not visited before	Suspend execution - terminate	critical
3	More than one transition outgoing a choice or junction pseudostate is enabled	Select one enabled transition and ignore the others	medium
4	There is no enabled transition outgoing a choice or junction pseudostate and there is one or more "else" transition outgoing this pseudostate	Select one "else" transition and ignore other transitions	medium
5	More than one transition outgoing the same state is enabled	Select one transition and ignore the others	medium

operation *open()*. However, this transition targets the join pseudostate. Therefore neither a trigger nor a guard condition can be associated with the transition. It violates the correctness rule 19 (Tab. II). This model flaw is quite often and is not critical. The trigger will be omitted in the generated code and the designer will be warned about this exclusion.

The same rule is violated in a transition outgoing state *inspection*. The guard [OK] can not be used in this context. The generated code will be incomplete and the warning reminds of the correcting the state machine model.

State machines model system behavior; therefore not all their elements can be verified statically. A part of defects is detected dynamically, i.e., during execution of state machines. For example, a situation that two enabled transitions are outgoing the same choice pseudostate can be detected after evaluation of appropriate guard conditions, namely during program execution. Defects detected dynamically in state machines are listed in Tab. III.

5 Experiments

The FXU framework is not directly associated with any modelling tool but UML models are passed between tools using files. Input models in some XMI variants, UML2 and UML formats, supported by Eclipse, are accepted. Therefore the solution is not tool-dependent. However, all experiments mentioned in this Section were performed with UML models created using IBM Rational Software Architect [33].

The presented approach for building the C# code and executing the automatically created applications was tested on over fifty models. The first group of ten models was aimed at classes. In experiments the correct and incorrect constructions encountering in class diagrams were checked, concerning especially association and generalization. Moreover, two bigger projects were tested. The first one was a design of a web page, which was a part of MDA project called Acceleo [34]. The model described a design of a web page. The second one presented a metamodel of an object-oriented modeling language [35].

Models from the next group (above forty models) comprised different diagrams, including both classes and their state machines. All possible constructs of UML 2.x behavioral state machines were used in different situations in the models. The biggest design included five state machines with about 80 states and 110 transitions, using complex and orthogonal states, different kinds of pseudostates and submachine states.

The programs realizing state machines were run taking into account different sequences of triggering events. The behavior modeled by state machines was observed and verified using detailed traces generated during program runs. They helped to test whether the obtained program behavior conforms to desired state machine semantics. For complex models, filtered traces that included selected information were also used.

In the performed experiments, applications realizing behavior specified in state machine models were developed in an automated way. For example, different airport subsystems were modeled in order to simulate a desired behavior. The essential part of the class model of *Airport_FlightControl* subsystem is shown in Fig. 1. It models occupation of runways and airplane parking places. Behavior of classes can be defined by state machines realising different policies. One exemplary state machine is shown for *Runway* class. Comparison of the policies is easily performed combining code generated for different versions of state machines in final applications.

6 Conclusion and future work

In this paper we discussed the problems of creation of valid C# applications realizing ideas modeled by classes and their state machines. Different C# mechanisms were effectively used for implementation of the full state machine model defined in the UML 2.x specification. We showed which correctness issues of models have to be checked during model transformation (static verification) and during application execution (dynamic verification). The detailed correctness rules help a developer to cope with possible flaws present in UML models. In the difference to other tools, using FXU the state machines including any complex features can be

effectively transformed into corresponding C# application. The tool support assists building of reliable applications including complex behavioral specifications. It can be especially useful for developing programs in which non-trivial state machines are intensely used, e.g., dependable systems, embedded reactive systems.

In the future work, we prepare other complex models implementing telecommunication problems. Capability of using advance state machine features and building reliable applications is very important in these cases.

As a complementary approach, another solution for C# code generation based on C# profiles is under development. Transform OCL Fragments Into C# (T.O.F.I.C.) tool supports labelling of UML model elements with stereotypes reflecting C# concepts. Target code is generated from a refined UML model and OCL constraints. In this approach, a model can be verified both during placing stereotypes and/or code generation process. Using dedicated profiles enforce more precise mapping to a given target language and therefore also checking of model correctness. However it requires more effort of a developer while creating a refined model.

Appendix

The appendix includes selected extracts of C# code generated for an exemplary class and its state machine shown in Fig. 1. Code of class operations is omitted. Method *InitFxu()* creates appropriate structure of the state machine and method *StartFxu()* initializes its behavior.

```
public class Runway    {
  private bool free;
// other attributes and operations (omitted)
//
  StateMachine sm1 =
       new StateMachine("RunwayStateMachine");
  public void InitFxu() {
     Region r1 = new Region("Region1");
     sm1.AddRegion(r1);
     InitialPseudostate v2 = new
               InitialPseudostate("");
     r1.AddVertex(v2);
     State v4 =  new State ("opend");
     r1.AddVertex(v4);
//...
     State v8 =  new State ("closed");
     r1.AddVertex(v8);
     v8.EntryBody = delegate(){ close(); };
     Region r3 = new Region("Region1");
     v8.AddRegion(r3);
//...
     State v11 =  new State ("restoration");
     r3.AddVertex(v11);
     Region r4 = new Region("Region1");
     v11.AddRegion(r4);
     Region r5 = new Region("Region2");
     v11.AddRegion(r4);
//...
     State v14 =  new State ("maintenance");
     r5.AddVertex(v14);
     v14.DoBody = delegate(){ maintain(); };
//
     Fork v15 = new Fork("");
     r3.AddVertex(v15);
//
     Transition t1 = new Transition(v2, v4);
     Transition t8 = new Transition(v8, v4);
     t8.AddTrigger(new CallEvent("open", 1))
//...
     Transition t11 = new Transition(v10, v10);
     t11.GuardBody = delegate()
               {return not free;};
     t11.ActionBody = delegate(){remove(); };
//...
} //End of InitFXU
  public void StartFxu(){
     sm1.Enter();      }
}
```

Log items selected from a detailed execution trace of the exemplary state machine (Fig. 1) are shown below. All labels of the items, including time stamps and item types (*Warning*, *Information*, *Debugging*), are omitted for the brevity reasons. A number in brackets denotes a number of a thread that realizes a considered part of machine execution.

[1] State diagram < RunwayStateMachine >: Entered.

[1] State diagram < RunwayStateMachine >: Execution of **entry-activity started**. State is now active.

[1] State diagram < RunwayStateMachine >: Execution of **entry-activity finished.**

[7] Initial **pseudostate** < RunwayStateMachine:: Region1{::UnNamedVertex}>: **Entered.**

[7] Transition from Initial pseudostate < RunwayState Machine::Region1{::UnNamedVertex}> to State < RunwayStateMachine::Region1::opend>: **Traversing** started.

[7] State < RunwayStateMachine::Region1:: opend>: Execution of **entry-activity started**. State is now active.

After *emergencyClose* trigger

[3] State diagram < RunwayStateMachine >: **Call-event** <emergencyClose [ID=1]> has been dispatched.

[9] State < RunwayStateMachine::Region1:: opend:: Region1::occupied>: Execution of **exit-activity started.**

Transition to fork from *preparation* state

[3] State diagram < RunwayStateMachine >: **Completion event** <> generated by State < RunwayStateMachine:: Region1::closed::Region1::preparation> has been dispatched.

[12] State < RunwayStateMachine::Region1::closed:: Region1::preparation >: Execution of **exit-activity started**.

[13] Transition from State < RunwayStateMachine:: Region1::closed::Region1::preparation > to Fork < RunwayStateMachine::Region1::closed::Region1{::UnNamedVertex}>: **Traversing** started.

[14] Transition **from Fork** < RunwayStateMachine:: Region1::closed::Region1{::UnNamedVertex}> to State < RunwayStateMachine::Region1::closed:::restoration::Region 2::maintenance>: **Traversing** started.

[16] State < RunwayStateMachine::Region1::closed::: restoration >: Execution of **entry-activity started**. State is now active.

[15] Transition **from Fork** < RunwayStateMachine:: Region1::closed::Region1{::UnNamedVertex}> to State < RunwayStateMachine::Region1::closed:::restoration::Region 1::repair>: **Traversing** started.

Acknowledgement

The authors would like to thank Kamil Raś for his help in preparing UML models.

References

[1] R. France, B. Rumpe (2007). Model-driven Development of Complex Software: A Research Roadmap. *Future of Software Engineering at ICSE'07*, IEEE Soc., pp. 37-54.

[2] Semantics of a foundation subset for executable UML models (FUML) (2008). http://www.omg.org/spec//FUML/

[3] Unified Modeling Language Superstructure v. 2.1.2 (2007). OMG Document formal/2007-11-02, http://www.uml.org

[4] C. Lange at al. (2003). An empirical investigation in quantifying inconsistency and incompleteness of UML designs. in *Proc. of 2nd Workshop on Consistency Problems in UML-based Software Development co-located at UML'03 Conf.*, San Francisko, USA, Oct 2003, pp. 26-34.

[5] R. Pilitowski, A. Derezinska (2007). Code Generation and Execution Framework for UML 2.0 Classes and State Machines. T. Sobh (eds.) *Innovations and Advanced Techniques in Computer and Information Sciences and Engineering*, Springer, pp. 421-427.

[6] A. Derezinska, R. Pilitowski (2007). Event Processing in Code Generation and Execution Framework of UML State Machines. in L. Madeyski at al. (eds.) *Software Engineering in progress.* Nakom, Poznań, pp.80-92.

[7] L. K. Kishore, D. Saini, IBM Rational Modeling Extension for Microsoft .NET, http://www.ibm.com/developerworks/rational/library/07/0306_kishore_saini/

[8] Rhapsody, http://www.telelogic.com/

[9] A. Niaz, J. Tanaka (2004). Mapping UML Statecharts into Java code. in *Proc. of the IASTED Int. Conf. Software Engineering*, Acta Press, Anheim, Calgary, Zurich, pp. 111-116.

[10] SmartState, http://www.smartstatestudio.com

[11] Hugo/RT, http://www.pst.ifi.lmu.de/projekte/hugo/

[12] A. Knapp, S. Merz (2002). Model Checking and Code Generation for UML State Machines and Collaborations. In D. Haneberg, G. Schellhorn, and W. Reif, Eds, Proc. 5th Workshop Tools for System Design and Verif., pp. 59-64. Tech. Rep. 2002-11, Inst. für Informatik, Univ. Augsburg,

[13] D. Harel, H. Kugler (2004). The Rhapsody Semantics of Statecharts (or On the Executable Core of the UML) (preliminary version), *SoftSpez Final Report*, LNCS, vol. 3147, Springer, Heidelberg, pp. 325-354.

[14] STL: UML 2 Semantics Project, References, Queen's University http://www.cs.queensu.ca/home/stl/internal/uml2/refs.htm

[15] M. Crane, J. Dingel (2005). UML vs. Classical vs. Rhapsody Statecharts: Not All Models are Created Equal. in: *MoDELS/UML* 2005, LNCS, vol. 3713, Springer, Heidelberg, pp. 97-112.

[16] Y. Jin, R. Esser and J. W. Janneck (2004). A Method for Describing the Syntax and Semantics of UML Statecharts. *Software and System Modeling*, vol. 3 no 2, Springer, 2004, pp. 150-163.

[17] H. Fecher, J. Schönborn (2007). UML 2.0 state machines: Complete formal semantics via core state machines. in *FMICS and PDMC* 2006, LNCS vol. 4346, Springer, Hildelberg, pp. 244-260.

[18] F. Chauvel, J-M. Jezequel (2005). Code Generation from UML Models with Semantic Variation Points. *MoDELS/UML* 2005, LNCS, vol. 3713, Springer, Heidelberg, pp. 97-112.

[19] A. Baruzzo, M. Comini (2006). Static verification of UML model consistency. *Proc. of the 3rd Workshop on Model Development, Validation and Verific., at MoDELS'06,* Genoa, Italy, pp. 111-126.

[20] A. Egyed (2007). Fixing inconsistencies in UML designs, in *Proc. of 29th Intern. Conf. on Software Engineering,* ICSE'07, IEEE Comp. Soc.

[21] S. Prochanow, R. von Hanxleden (2007). Statecharts development beyond WYSIWIG, in G. Engels et al. (Eds.) *MODELS* 2007, LNCS 4735, Springer, Berlin Heidelberg, pp. 635-649.

[22] L-K. Ha, B-W Kang. (2003). Meta-Validation of UML Structural Diagrams and Behavioral Diagrams with Consistency Rules. *Proc. of IEEE PACRIM*, Vol. 2., 28-30 Aug. pp. 679-683.

[23] MDA Guide Ver. 1.0.1 (2003). OMG Document omg/2003-06-01.

[24] S. Frankel (2003). *Model Driven Architecture: Appling MDA to enterprise computing*. Wiley Press, Hoboken, NJ.

[25] J. Wuest, SDMetrics - the UML design measurement tool, http://www.sdmetrics.com/manual?LORules.html

[26] S. J. Mellor, M. J. Balcer (2002). *Executable UML a Foundation for Model-Driven Architecture.* Addison-Wesley.

[27] C. Raistrick, at al. (2004). *Model Driven Architecture with Executable UML* Cambridge University Press.

[28] K. Carter, iUMLite - xUML modeling tool, http://www.kc.com (visited 2009)

[29] A. Maraee, M. Balaban (2006). Efficient decision of consistency in UML diagrams with constrained generalization sets. *Proc. of the 1st Workshop on Quality in Modeling, co-located. at MoDELS'06*, Genoa, Italy, pp. 1-14.

[30] F. J. Lange, M. R. V. Chaudron (2007). Defects in industrial UML models - a multiple case study. *Proc. of the 2nd Workshop on Quality in Modeling, at MoDELS'07*, Nashville, TN, USA, pp. 50-64.

[31] E. Gamma, R. Helm, R. Johnson, J. Vlissides (1995). *Design patterns: elements of reusable object-oriented software.* Boston Addison-Wesley.

[32] J. Liberty (2005). *Programming C#,* O'Reilly Media.

[33] IBM Rational Software Architect, http://www-306.ibm.com/software/rational

[34] Acceleo project http://www.acceleo.org

[35] G. Booch, Metamodel of object-oriented modeling language, http://www.booch.com/architecture/architecture/artifacts/architecture.emx.

A Pseudo Random Bit Generator Based on Chaotic Logistic Map and its Statistical Testing

Vinod Patidar and K. K. Sud
Department of Basic Sciences, School of Engineering,
Sir Padmapat Singhania University, Bhatewar, Udaipur – 313 601, India
E-mail: vinod_r_patidar@yahoo.co.in

N. K. Pareek
University Computer Centre, Vigyan Bhawan, New Campus
M.L.S. University, Udaipur 313002, Rajasthan, India

Keywords: pseudo random, random, PRBG, random bit generator, logistic map, cryptography, stream cipher

Received: May 20, 2008

During last one and half decade an interesting relationship between chaos and cryptography has been developed, according to which many properties of chaotic systems such as: ergodicity, sensitivity to initial conditions/system parameters, mixing property, deterministic dynamics and structural complexity can be considered analogous to the confusion, diffusion with small change in plaintext/secret key, diffusion with a small change within one block of the plaintext, deterministic pseudo randomness and algorithmic complexity properties of traditional cryptosystems. As a result of this close relationship several chaos-based cryptosystems have been put forward since 1990. In one of the stages of the development of chaotic stream ciphers, the application of discrete chaotic dynamical systems in pseudo random bit generation has been widely studied recently. In this communication, we propose a novel pseudo random bit generator (PRBG) based on two chaotic logistic maps running side-by-side and starting from random independent initial conditions. The pseudo random bit sequence is generated by comparing the outputs of both the chaotic logistic maps. We discuss the suitability of the logistic map by highlighting some of its interesting statistical properties, which make it a perfect choice for such random bit generation. Finally, we present the detailed results of the statistical testing on generated bit sequences, done by the most stringent tests of randomness: the NIST suite tests, to detect the specific characteristics expected of truly random sequences.

Povzetek: Predstavljen je psevdo naključni generator bitov na osnovi kaotičnega pristopa.

1 Introduction

New rapid developments in the telecommunication technologies especially the Internet and mobile networks have extended the domain of information transmission, which in turn present new challenges for protecting the information from unauthorized eavesdropping. It has intensified the research activities in the field of cryptography to fulfill the strong demand of new secure cryptographic techniques [1, 2].

Recently researchers from the nonlinear dynamics community have noticed an interesting relationship between chaos and cryptography. According to that, many properties of chaotic systems such as: ergodicity, sensitivity to initial conditions/system parameters, mixing property, deterministic dynamics and structural complexity can be considered analogous to the confusion, diffusion with small change in plaintext/secret key, diffusion with a small change within one block of the plaintext, deterministic pseudo randomness and algorithmic complexity properties of traditional cryptosystems [3]. As a result of this close relationship several chaos-based cryptosystems have been put forward since 1990 [4]. These chaos-based cryptosystems can be broadly classified into two categories: analog and digital. Analog chaos-based cryptosystems are based on the techniques of control [5, 6] and synchronization [5, 6] of chaos. There are several ways through which analog chaos-based cryptosystems can be realized such as: chaotic masking [7-11], chaotic modulation [12-15], chaotic switching [16, 17], inverse system approach [18, 19] etc. On the other hand in digital chaos-based cryptosystems, chaotic discrete dynamical systems are implemented in finite computing precision. Again there are number of ways through which digital chaos-based cryptosystems be realized: block ciphers based on forward and/or reverse iterations of chaotic maps [4, 20-23], block ciphers based on chaotic round functions [24-27], stream ciphers implementing chaos-based pseudo random bit generators (PRBG) [28-33] etc.

The subject of the present manuscript is the generation of cryptographically secure pseudo random bit sequences, which can be further used in the development of fool-proof stream ciphers and its statistical testing. In the following paragraph, we briefly summarize a few efforts undertaken recently in this direction.

The first, relatively unnoticed, idea of designing a pseudo-random number generator by making use of chaotic first order nonlinear difference equation was proposed by Oishi and Inoue [34] in 1982 where they could construct a uniform random number generator with an arbitary Kolmogorov entropy. After a long gap, in 1993 Lin and Chua [35] designed a pseudo random number generator by using a second-order digital filter and realized it on digital hardware. In 1996 Andrecut [36] suggested a method for designing a random number generator based on logistic map and also compared the congruential random generators, which are periodic, with the logistic random number generator, which is infinite and aperiodic. In 1999 Gonzalez and Pino [37] generalized the logistic map and designed a truly unpredictable random function, which helped in the generation of truly random numbers. In 2001 Kolesov et al [38] developed a digital random-number generator based on the discrete chaotic-signal. The suggested digital generator employed the matrix method of chaotic-signal synthesis. Further, Kocarev [39] and Stojanovski et al [40] analyzed the application of a chaotic piecewise-linear one-dimensional map as random number generator. Li et al [32] did a theoretical analysis, which suggests that piecewise linear chaotic maps have perfect cryptographic properties like: balance in the defined interval, long cycle length, high linear complexity, good correlation properties etc. They also pointed out that bit streams generated through a single chaotic system are potentially insecure as the output may leak some information about the chaotic system. To overcome this difficulty, they proposed a pseudo random bit generator based on a couple of piecewise linear chaotic maps, which are iterated independently and the bit streams are generated by comparing the outputs of these chaotic maps. They also justified their theoretical claims through a few numerical experimentations on the proposed pseudo random bit generator. In 2003 Kocarev and Jakimoski [41] discussed the different possibilities of using chaotic maps as pseudo-random number generators and also constructed a chaos-based pseudorandom bit generator. In 2004 Fu et al [42] proposed a chaos-based random number generator using piecewise chaotic map. Further, a one-way coupled chaotic map lattice was used by Huaping et al [43] for generating pseudo-random numbers. They showed that with suitable cooperative applications of both chaotic and conventional approaches, the output of the spatiotemporal chaotic system can meet the practical requirements of random numbers i.e. excellent random statistical properties, long periodicity of computer realizations and fast speed of random number generations. This pseudo-random number generator can be used as an ideal synchronous and self-synchronizing stream cipher for secure communications. In 2005 Li et al [44] designed and analysed a random number generator based on a piecewise-linear map. A new pseudo-random number generator (PRNG) based on modified logistic map was proposed by Liu [45] and a design of a chaotic stream cipher using it was also suggested. Further, a chaotic random number generator was developed by Wang et al [46] and realized it by an analog circuit. In 2006, Wang et al [47] proposed a pseudo-random number generator based on z-logistic map, where the binary sequence through the chaotic orbit was realized under finite computing precision. Recently in 2007, Ergun and Ozogur [48] showed that the bit streams, generated from the stroboscopic Poincare map of a non-autonomous chaotic electronic circuit, pass the four basic tests of FIPS-140-2 as well as NIST tests suite. Very recently, Hu et al [49] proposed a true random number generator (which generates a 256-bit random number by computer mouse movement), where the authors used three chaos-based approaches namely: discretized 2D chaotic map permutation, spatiotemporal chaos and MASK algorithm to eliminate the effect of similar mouse movement patterns. The results have been tested through NIST tests suite. Recently, Patidar et al [50] proposed a pseudorandom bit generator based on the chaotic standard map and presented its testing analysis using the NIST as well as DIEHARD test suites. No failure has been observed in any of the tests of these two test suites.

In this paper, we propose a pseudo random bit generator (PRBG) based on two chaotic logistic maps. Most of the existing pseudo random bit generators [34-47] are based on a single chaotic system and there are known techniques in chaos theory to extract information about the chaotic systems from its trajectory, which makes such chaos-based pseudo random bit generators insecure [32]. However the proposed pseudo random bit generator is based on two chaotic systems running side-by-side, which of course increases the complexity in the random bit generation and hence becomes difficult for an intruder to extract information about the chaotic system. In the next section, we briefly introduce the logistic map, which is a basic building block of the proposed pseudo random bit generator and its properties, which make it a suitable choice for the generation of random bit sequences.

2 The logistic map

The logistic map is a very simple mathematical model often used to describe the growth of biological populations. In 1976 May [51] showed that this simple model shows bewildering complex behaviour. Later Fiegenbaum [52, 53] reported some of the universal quantitative features, which became the hallmark of the contemporary study of chaos. Because of its mathematical simplicity, this model continues to be useful test bed for new ideas in chaos theory as well as application of chaos in cryptography [4]. The simple modified mathematical form of the logistic map is given as:

$$X_{n+1} = f(X_n) = \lambda X_n (1 - X_n), \quad (1)$$

where X_n is a state variable, which lies in the interval [0, 1] and λ is called system parameter, which can have any value between 1 and 4.

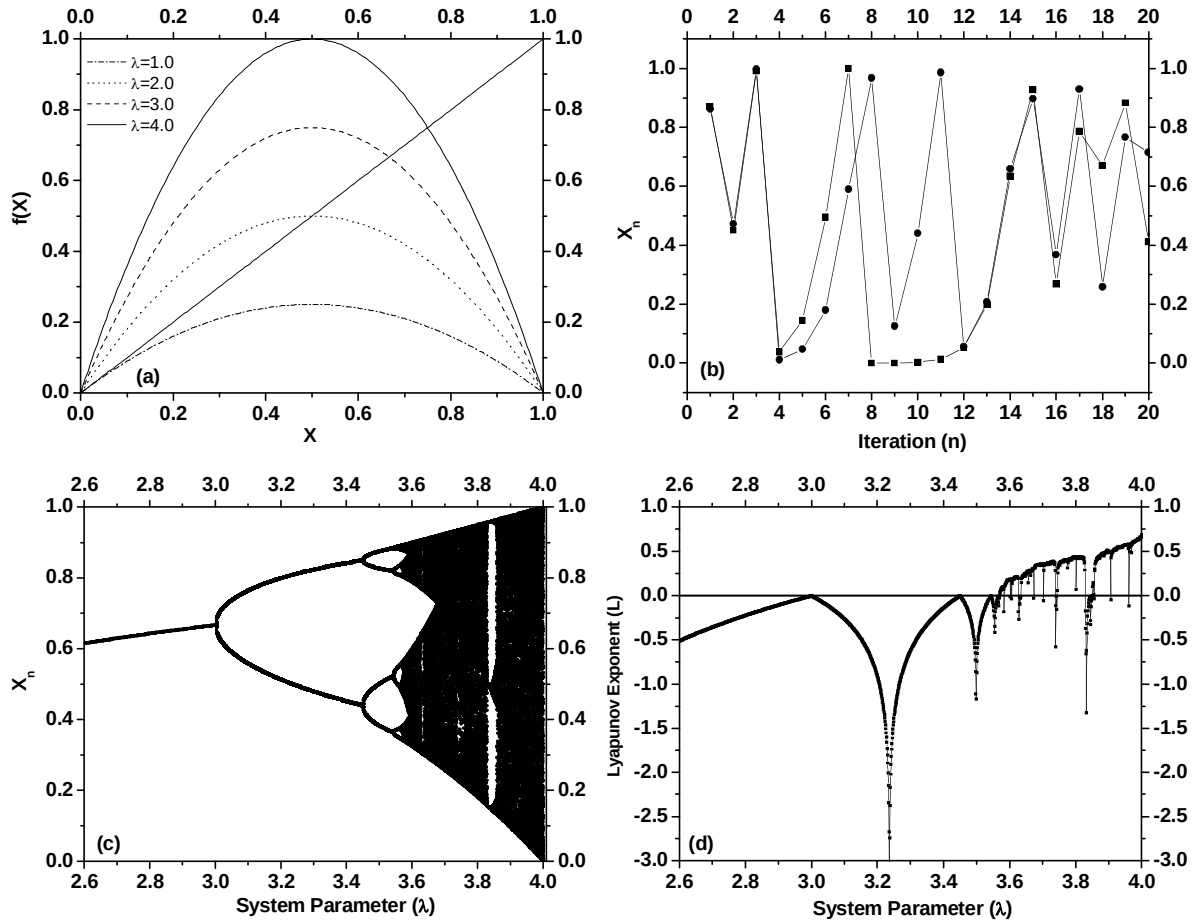


Figure 1: Behaviour of the logistic map: (a) map function $f(X) = \lambda X(1-X)$ for different values of parameter λ, (b) sensitivity on initial conditions for $\lambda = 4.0$, (c) bifurcation plot showing the qualitative changes in the dynamical behaviour as a function of parameter λ and (d) Lyapunov exponent (quantitative measurement of chaos) as a function of parameter λ.

In Figure 1(a), we have plotted the map function $f(X)$ as a function of X for different values of system parameter λ. It is clear that the map function $f(X)$ is symmetric about the mid point of the interval [0, 1]. This iterative map shows a strange complex behaviour for the system parameter values $\lambda > 3.5699...$, where map function never repeats its history. This peculiar behaviour is termed as chaos and more precisely, it can be described by the phrase 'sensitivity on initial conditions'. In Figure 1(b), we have depicted one such example of sensitivity on initial conditions for $\lambda = 4.0$. It is clear that the two trajectories of the logistic map starting nearby, soon diverge exponentially in the course of time and have no correlation between them. If we calculate the correlation coefficient for these two data sets (for $N =$ 1 to 10^6), it comes out equal to -0.000839 at the significance level of $\alpha = 0.01$, which confirms the completely uncorrelated behaviour of two trajectories, which are starting from almost same initial conditions. In Figure 1(c), we have summarized the complete dynamical behaviour of the logistic map by using the bifurcation plot: a plot illustrating the qualitative changes in the dynamical behaviour of the logistic map as a function of system parameter λ. It is also clear from the bifurcation diagram that the map function is surjective/onto in the complete interval [0, 1] only at $\lambda = 4.0$ i.e., each and every value of $f(X)$ in the interval [0, 1] is an image of at least one value of X in the same interval [0, 1]. The interval of surjectivity reduces as we decrease the value of λ form 4.0. In Figure 1(d), we have displayed the Lyapunov exponent ($L = \lim_{N\to\infty} \sum_{i=1}^{N} \ln | f'(X_i) |$, which is a quantative measure of chaos and a positive Lyapunov exponent indicates chaos) as a function of system parameter λ.

Invariant density measure and ergodicity: If we divide the complete range of state variable [0, 1] into a set of M equal sub-intervals and calculate the number that a trajectory visits a particular sub-interval i ($1 \le i \le M$), if it is m_i then the probability associated with the sub-interval i is $p_i = m_i / N$ (where N is the total number of trajectory points considered). A graph of

p_i as a function of i gives us the natural probability distribution or probability measure.

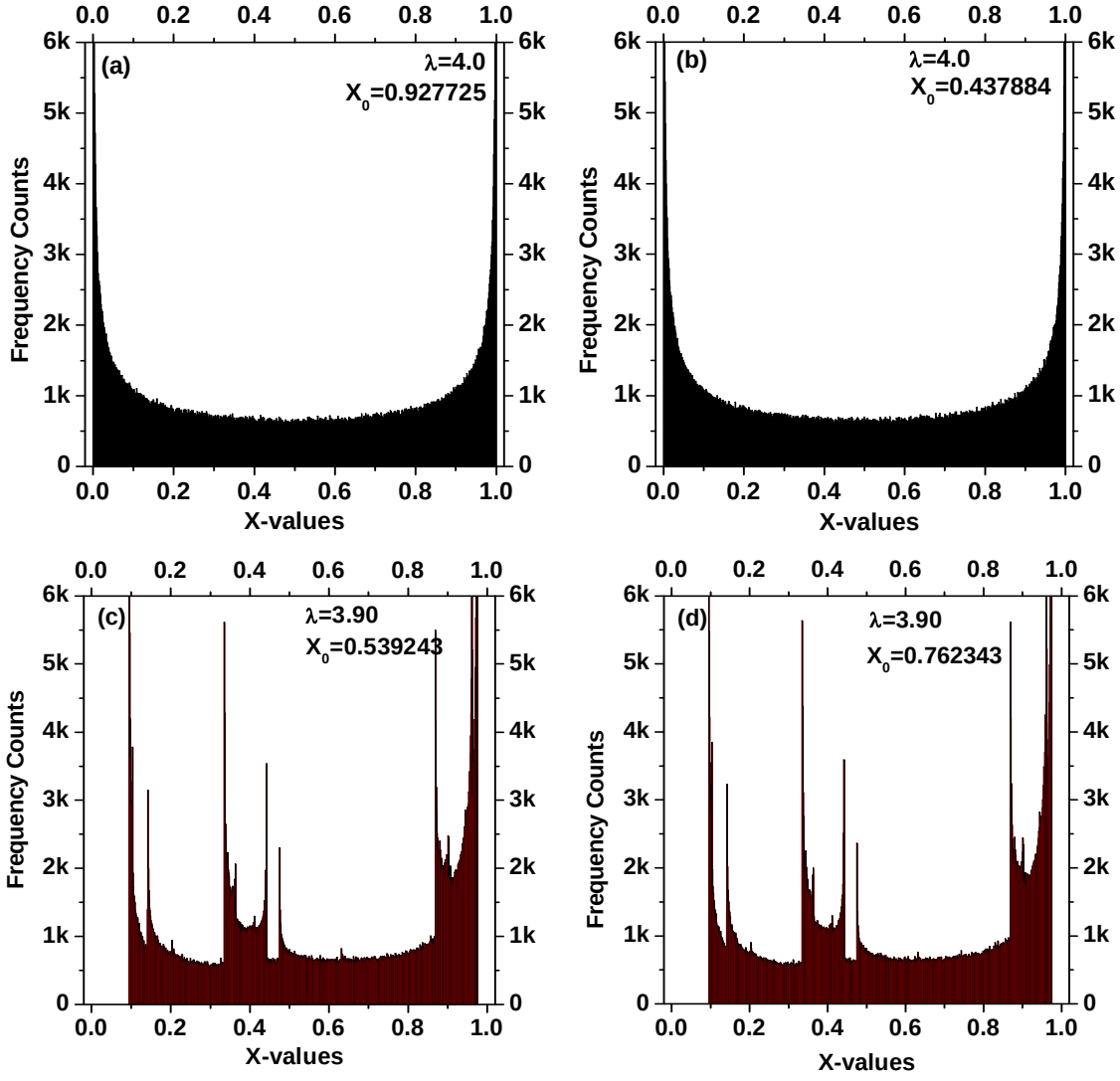


Figure 2: Probability distributions for the logistic map trajectories (a) and (b) for $\lambda = 4.0$ & (c) and (d) for $\lambda = 3.90$.

For a chaotic trajectory, this probability distribution does not depend on the starting point of the trajectory (if we observe the trajectory for a long enough duration) i.e., the probability measure is unchanged under the dynamics of the system, we term it as *invariant probability measure* or *invariant density measure*. It has been shown analytically that for the logistic map with system parameter $\lambda = 4$ the probability distribution is given by [54],

$$P(X) = \frac{1}{\pi\sqrt{X(1-X)}}. \tag{2}$$

If such an invariant distribution exists for a system then it allows us to replace the time averages by the spatial averages and the system is called *ergodic*. This ergodic property provides us a very simple way for calculating the average properties of the system. For example the average Lyapunov exponent for the logistic map with system parameter $\lambda = 4.0$ can be calculated with the help of above invariant probability distribution as:

$$L = \int_0^1 P(X)\,\lambda(X)\,dX, \tag{3}$$

here $\lambda(X)$ is the local Lyapunov exponent. Using (3) we have

$$L = \int_0^1 \frac{1}{\pi\sqrt{X(1-X)}} \ln|f'(X)|\,dX, \tag{4}$$

$$= \int_0^1 \frac{1}{\pi\sqrt{X(1-X)}} \ln|4(1-2X)|\,dX = \ln 2, \tag{5}$$

which is positive and confirms the chaotic nature of the logistic map at $\lambda = 4.0$ .In Figures 2(a) and 2(b), we have shown probability distributions for two different trajectories of logistic map starting from different initial conditions ($X_0 = 0.927725$ and 0.437884) with system parameter $\lambda = 4.0$. Here the interval [0, 1] has been divided into 1000 equal sub-intervals and total $N = 10^6$

points are used for each trajectory. Clearly both the distributions are same hence the logistic map exhibits unique invariant probability measure for $\lambda = 4.0$. It is also clear that the probability distributions are symmetric about the mid point of the interval [0, 1]. However in Figures 2(c) and 2(d), probability distributions are displayed for the two logistic trajectories starting form $X_0 = 0.835283$ and 0.582735 with the system parameter $\lambda = 3.90$. It is clear that the logistic map also exhibits invariant probability measure for $\lambda = 3.90$ but the distribution is not symmetric about the mid point of the interval [0, 1]. From Figure 2, one may also conclude that the logistic map has surjective character in the complete interval [0, 1] only very near to $\lambda = 4$. In the next section, we discuss the basic terminology for the random bit generation and details of the proposed pseudo random bit generator (PRBG).

3 The proposed PRBG

A *random bit generator (RBG)* is a device or algorithm, which outputs a sequence of statistically independent and unbiased binary digits. Such generator requires a naturally occurring source of randomness (non-deterministic). In most practical environments designing a hardware device or software programme to exploit the natural source of randomness and produce a bit sequence free from biases and correlation is a difficult task. In such situations, the problem can be ameliorated by replacing a random bit generator with a pseudo random bit generator (PRBG).

A *pseudo random bit generator (PRBG)* is a deterministic algorithm, which uses a truly random binary sequence of length *k* as input called seed and produces a binary sequence of length *l*>>*k*, called pseudo random sequence, which appears to be random. The output of a PRBG is not truly random; in fact the number of possible output sequences is at most a small fraction ($2^k/2^l$) of all possible binary sequences of length *l*. The basic intent is to take a small truly random sequence of length *k* and expand it to a sequence of much larger length *l* in such a way that an adversary can not efficiently distinguish between output sequence of PRBG and truly random sequence of length *l* [2].

In this paper, we are proposing a PRBG, which is based on two logistic maps, starting from random independent initial conditions $(X_0, Y_0 \in (0,1)$ and $X_0 \neq Y_0$)

$$X_{n+1} = \lambda_1 X_n (1 - X_n), \quad (6)$$

$$Y_{n+1} = \lambda_2 Y_n (1 - Y_n). \quad (7)$$

The bit sequence is generated by comparing the outputs of both the logistic maps in the following way:

$$g(X_{n+1}, Y_{n+1}) = \begin{cases} 1 & if \ X_{n+1} > Y_{n+1} \\ 0 & if \ X_{n+1} \le Y_{n+1} \end{cases}, \quad (8)$$

The set of initial conditions $(X_0, Y_0 \in (0,1)$ and $X_0 \neq Y_0)$ serves as the seed for the PRBG, if we supply the exactly same seed to the PRBG, it will produce the same bit sequence due to the above deterministic procedure. The schematic block diagram of the proposed PRBG is shown in Figure 3.

In a recent analytical study Li et al [32] showed that the binary sequences produced by comparing the outputs of two chaotic maps will have perfect cryptographic properties if following requirements are satisfied:

(i) Both the maps should produce asymptotically independent trajectories as $n \to \infty$,
(ii) both the maps are surjective on the same interval,
(iii) both the maps have unique invariant density distributions $P_1(x)$ and $P_2(x)$ and are ergodic on the defined interval,
(iv) either $P_1(x) = P_1(x)$ or $P_1(x)$ and $P_2(x)$ are symmetric about the mid point of the interval.

It is clear from the discussion of Section 2 that the logistic map exhibits all the above mentioned properties wherever it shows chaotic behaviour. In view of the condition (ii), we have to choose the same value of λ for both the chaotic maps (i.e., $\lambda_1 = \lambda_2 = \lambda$) to maintain its surjectivity in the same interval. However it would be most appropriate to choose λ very near to 4.0 to make available a large interval for the seed values X_0 and Y_0, which will in turn increase the key space of the stream cipher, where the proposed cipher is going to be used. It is also suggested that before choosing $\lambda_1 = \lambda_2 = \lambda$ other than 4.0, a careful analysis of Lyapunov exponent must be done to take care of the asymptotic independence of two trajectories (property (i)), larger the Lyapunov exponent lesser the correlation between the trajectories starting from almost same initial conditions.

In the next section, we mention various resources for statistical testing of PRBGs which are available to researchers from academia and industry who wish to analyze their newly developed PRBG. We also briefly introduce the resource (NIST tests suite), which we have used for analyzing the proposed PRBG and discuss the

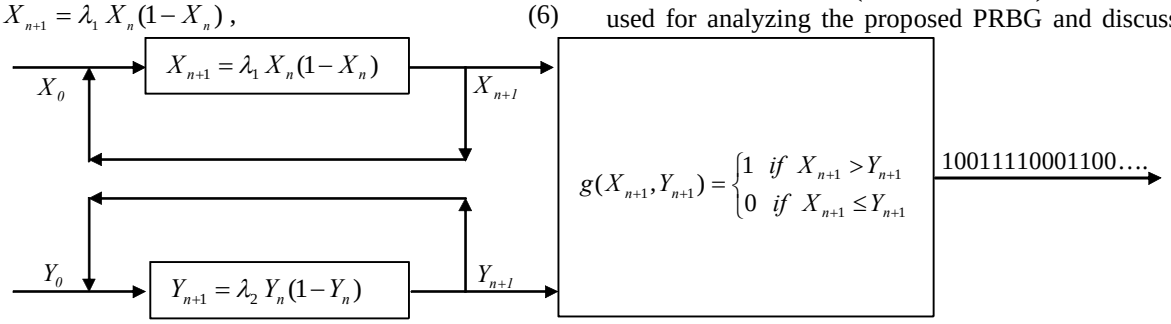


Figure 3: Schematic block diagram of the proposed pseudo random bit generator (PRBG).

results of our analysis in detail. It is to be noted here that the PRBG and the analysis proposed in [32] does not present any idea about the performance of PRBG in respect to the NIST test suite (whether successful or not). Hence we can not compare both the PRBGs in terms of their superiority/inferiority. However the idea of the present PRBG has emerged from the analytical study and properties reported in [32].

4 Statistical testing

In order to gain the confidence that newly developed pseudo random bit generators are cryptographically secure, they should be subjected to a variety of statistical tests designed to detect the specific characteristics expected of truly random sequences. There are several options available for analyzing the randomness of the newly developed pseudo random bit generators. The four most popular options are: (i) NIST suite of statistical tests [55], (ii) The DIEHARD suite of statistical tests [56], (iii) The Crypt-XS suite of statistical tests [57] and (iv) The Donald Knuth's statistical tests set [58]. There are different number of statistical tests in each of the above mentioned test suites to detect distinct types of non-randomness in the binary sequences. Various efforts based on the principal component analysis show that not all the above mentioned suites are needed to implement at a time as there are redundancy in the statistical tests (i.e., all the tests are not independent). The results also suggest that the NIST statistical tests suite contains a sufficient number of nearly independent statistical tests, which detect any deviation from the randomness [59]. Hence for analyzing the randomness of the proposed pseudo random bit generator (PRBG), we use the most stringent tests of randomness: the NIST suite tests. In the following subsection, we briefly mention the various statistical tests of NIST suite their focuses and purposes.

4.1 The NIST Tests Suite

The NIST tests suite is a statistical package comprising of 16 tests that are developed to test the randomness of (arbitrary long) binary sequences produced by either hardware or software based cryptographic random or pseudo random bit generators. These tests focus on a variety of different types of non-randomness that could exist in a binary sequence. Broadly, we may classify these sixteen tests into two categories: (i) non-parameterized tests and (ii) parameterized tests.

4.1.1 Non-parameterized tests

Frequency (monobit) test: The focus of the test is the proportion of zeroes and ones for the entire sequence. The purpose of this test is to determine whether the number of ones and zeros in a sequence are approximately the same as would be expected for a truly random sequence. The test assesses the closeness of the fraction of ones to ½, that is, the number of ones and zeroes in a sequence should be the same.

Runs test: The focus of this test is the total number of runs in the sequence, where a run is an uninterrupted sequence of identical bits. A run of length k consists of exactly k identical bits and is bounded before and after with a bit of the opposite value. The purpose of the runs test is to determine whether the number of runs of ones and zeros of various lengths is as expected for a random sequence. In particular, this test determines whether the oscillation between such zeros and ones is too fast or too slow.

Test for longest run of ones in a block: The focus of the test is the longest run of ones within M-bit blocks. The purpose of this test is to determine whether the length of the longest run of ones within the tested sequence is consistent with the length of the longest run of ones that would be expected in a random sequence.

Lempel-Ziv compression test: The focus of this test is the number of cumulatively distinct patterns (words) in the sequence. The purpose of the test is to determine how far the tested sequence can be compressed. The sequence is considered to be non-random if it can be significantly compressed. A random sequence will have a characteristic number of distinct patterns.

Binary matrix rank test: The focus of the test is the rank of disjoint sub-matrices of the entire sequence. The purpose of this test is to check for linear dependence among fixed length substrings of the original sequence.

Cumulative sums test: The focus of this test is the maximal excursion (from zero) of the random walk defined by the cumulative sum of adjusted (-1, +1) digits in the sequence. The purpose of the test is to determine whether the cumulative sum of the partial sequences occurring in the tested sequence is too large or too small relative to the expected behavior of that cumulative sum for random sequences. This cumulative sum may be considered as a random walk. For a random sequence, the excursions of the random walk should be near zero.

Discrete Fourier transform (spectral) test: The focus of this test is the peak heights in the Discrete Fourier Transform of the sequence. The purpose of this test is to detect periodic features (i.e., repetitive patterns that are near each other) in the tested sequence that would indicate a deviation from the assumption of randomness.

Random excursions test: The focus of this test is the number of cycles having exactly K visits in a cumulative sum random walk. The cumulative sum random walk is derived from partial sums after the (0,1) sequence is transferred to the appropriate (-1, +1) sequence. A cycle of a random walk consists of a sequence of steps of unit length taken at random that begin at and return to the origin. The purpose of this test is to determine if the number of visits to a particular state within a cycle deviates from what one would expect for a random sequence. This test is actually a series of eight tests (and conclusions), one test and conclusion for each of the states: x = -4, -3, -2, -1 and +1, +2, +3, +4.

Random excursions variant test: The focus of this test is the total number of times that a particular state is visited (i.e., occurs) in a cumulative sum random walk. The purpose of this test is to detect deviations from the expected number of visits to various states in the random walk. This test is actually a series of eighteen tests (and

conclusions), one test and conclusion for each of the states: x = -9, -8, …, -1 and +1, +2, …, +9.

4.1.2 Parameterized tests

Frequency test within a block: The focus of the test is the proportion of ones within M-bit blocks. The purpose of this test is to determine whether the frequency of ones in an M-bit block is approximately M/2, as would be expected under an assumption of randomness. For block size M=1, this test degenerates to the Frequency (Monobit) test.

Approximate entropy test: The focus of this test is the frequency of all possible overlapping m-bit patterns across the entire sequence. The purpose of the test is to compare the frequency of overlapping blocks of two consecutive/adjacent lengths (m and $m+1$) against the expected result for a random sequence.

Linear complexity test: The focus of this test is the length of a linear feedback shift register (LFSR). The purpose of this test is to determine whether or not the sequence is complex enough to be considered random. Random sequences are characterized by longer LFSRs.

Maurer's universal statistical test: The focus of this test is the number of bits between matching patterns (a measure that is related to the length of a compressed sequence). The purpose of the test is to detect whether or not the sequence can be significantly compressed without loss of information. A significantly compressible sequence is considered to be non-random.

Serial test: The focus of this test is the frequency of all possible overlapping m-bit patterns across the entire sequence. The purpose of this test is to determine whether the number of occurrences of the 2^m m-bit overlapping patterns is approximately the same as would be expected for a random sequence. Random sequences have uniformity; that is, every m-bit pattern has the same chance of appearing as every other m-bit pattern. Note that for m = 1, the Serial test is equivalent to the Frequency test.

Non-overlapping template matching test: The focus of this test is the number of occurrences of pre-specified target strings. The purpose of this test is to detect generators that produce too many occurrences of a given non-periodic (aperiodic) pattern. For this test and for the Overlapping Template Matching test, an m-bit window is used to search for a specific m-bit pattern. If the pattern is *not* found, the window slides one bit position. If the pattern is found, the window is reset to the bit after the found pattern, and the search resumes.

Overlapping template matching test: The focus of the Overlapping Template Matching test is the number of occurrences of pre-specified target strings. Both this test and the Non-overlapping Template Matching test use an m-bit window to search for a specific m-bit pattern. It differs from the non-overlapping template matching test in the sense that in this case when the pattern *is* found, the window slides only one bit before resuming the search.

For the detailed description of above mentioned 16 tests, we refer the readers to the NIST document [55].

4.2 Testing strategy

The NIST framework, like many statistical tests, is based on hypothesis testing. A hypothesis test is a procedure for determining if an assertion about a characteristic of a population is reasonable. In the present case, the test involves determining whether or not a specific sequence of zeroes and ones is random (it is called null hypothesis H_0).

For each test, a relevant randomness statistic be chosen and used to determine the acceptance or rejection of the null hypothesis. Under an assumption of randomness, such a statistic has a distribution of possible values. A theoretical reference distribution of this statistic under the null hypothesis is determined by mathematical methods and corresponding probability value (*P-value)* is computed, which summarizes the strength of the evidence against the null hypothesis. For each test, the *P-value* is the probability that a perfect random number generator would have produced a sequence less random than the sequence that was tested, given the kind of non-randomness assessed by the test. If a *P-value* for a test is determined to be equal to 1, then the sequence appears to have perfect randomness. A *P-value* equal to zero indicates that the sequence appears to be completely non-random. A significance level (α) be chosen for the tests and if $P-value \geq \alpha$, then the null hypothesis is accepted i.e., the sequence appears to be random. If $P-value < \alpha$, then the null hypothesis is rejected; i.e., the sequence appears to be non-random. Typically, the significance level (α) is chosen in the interval $[0.001, 0.01]$. The $\alpha = 0.01$ indicates that one would expect 1 sequence out of 100 sequences to be rejected. A $P-value \geq 0.01$ would mean that the sequence would be considered to be random with a confidence of 99 %.

For the numerical experimentations on the proposed pseudo random bit generator, we have generated 2000 (sample size $m = 2000$) different binary sequences (each sequence has been generated from a randomly chosen seed $X_0, Y_0 \in (0,1)$ with $X_0 \neq Y_0$ and $\lambda_1 = \lambda_2 = \lambda = 4.0$) each of length 10^6 bits and computed the *P-value* corresponding to each sequence for all the 16 tests of NIST Suite (in all we have computed total 48×2000 = 96000 *P-values*). All the computations have been performed in double precision floating point representation. We refer the readers to Rukhin et al [55] for the detailed mathematical procedure for calculating the *P-value* for each individual test of NIST suite. For the analysis of *P-values* obtained from various statistical tests, we have fixed the significance level at $\alpha = 0.01$. In Tables 1 and 2 respectively, we have summarized the results obtained after implementing non-parameterized and parameterized tests of NIST suite on the binary sequences produced by the proposed pseudo random bit generator.

4.3 Interpretation of results:

(i) *Uniform distribution of P-values:* For each test, the distribution of *P-values* for a large number of binary sequences ($m = 2000$) has been examined. Visually, it has been done by plotting the histograms, where we have divided the complete interval of *P-values* [0, 1] into 10

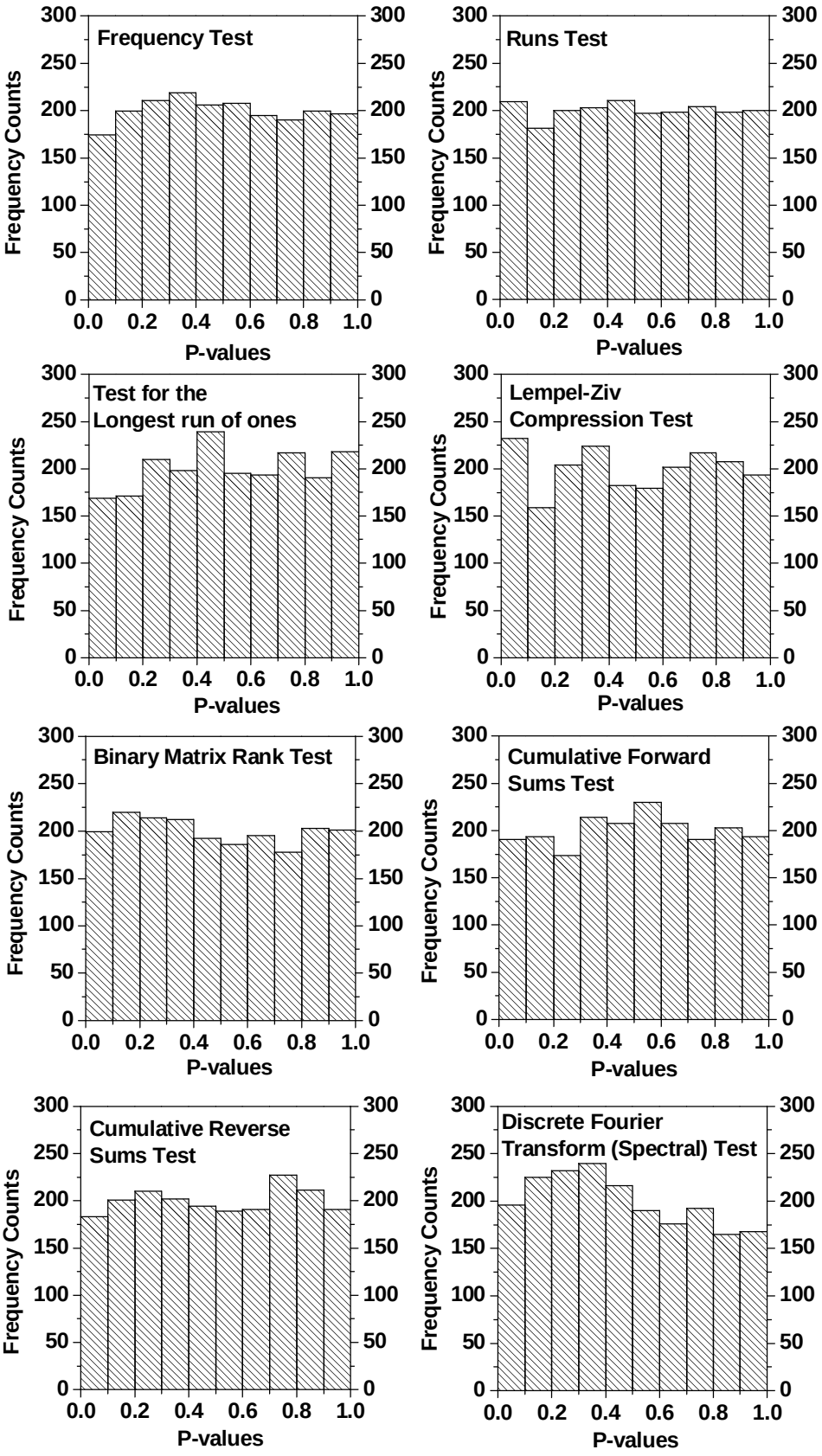


Figure 4: Histograms of *P-values* for non-parameterized tests of NIST suite.

equal sub-intervals and the *P-values* that lie in each subinterval has been counted and displayed. We have displayed the result of one such analysis in Figure 4 for some of the non-parameterized tests. It is clear from Figure 4 that the *P-values* for each statistical test are uniformly distributed in the complete interval of *P-values* i.e., [0, 1]. We obtain the similar results for the remaining non-parametric and parametric tests also.

The uniformity of the P-values has also been examined quantitatively via an application of χ^2 test and the determination of a *P-value* corresponding to the Goodness-of-Fit distributional test on the *P-values* obtained for each statistical test (i.e., a *P-value* of the *P-values*, which is denoted by $P-value_T$). The computation is as follows:

$$\chi^2 = \sum_{i=1}^{10} \left(f_i - \frac{m}{10} \right)^2 \Big/ \left(\frac{m}{10} \right), \quad (9)$$

where f_i is the number of P-values in the sub-interval i and m is the size of the sample, which is $m = 2000$ for the present analysis. The *P-value* of the *P-values* (i.e., $P-value_T$) is obtained from the χ^2 by using

$$P-value_T = igamc\left(\frac{9}{2}, \frac{\chi^2}{2} \right), \quad (10)$$

where *igamc*( ) is the incomplete Gamma function. If $P-value_T \geq 0.0001$ then the P-values are considered to be uniformly distributed.

The computed $P-value_T$ corresponding to each statistical test has been given in Tables 1 and 2. In Figures 5(a) and (b) respectively, we have graphically depicted the computed $P-value_T$ for each non-parameterized and parameterized test along with the threshold value (0.0001). It is clear that the computed $P-value_T$ for each test lies above the threshold value, which confirms the uniformity of the *P-values* for all the 16 tests of NIST suite.

(ii) *Proportions of the sequences passing the tests:* We have calculated the proportion of the sequences passing a particular statistical test and compared it with the range of acceptable proportion. The range of acceptable proportion is determined by using the confidence interval given by $\hat{p} \pm 3\sqrt{\frac{\hat{p}(1-\hat{p})}{m}}$, (11)

where m is the sample size and $\hat{p} = 1 - \alpha$, which are $m = 2000$ and $\hat{p} = 1 - 0.01 = 0.99$ for the present analysis. So the range of acceptable proportion is $[0.9833245, 0.9966745]$. The quantitative results of proportions are given the Tables 1 and 2 respectively for

Table 1: Non-parameterized tests results.

- Number of binary sequences tested (m): 2,000
- Length of each binary sequence: 1,000,000 bits
- Significance level (α) = 0.01
- The range of acceptable proportion is [0.9833245, 0.9966745]

- Null hypothesis (H_0): The binary sequence is random
- If $P-value \geq \alpha(0.01)$ then the null hypothesis (H_0) is accepted.
- If $P-value < \alpha(0.01)$ then the null hypothesis ( ) is rejected.
- If $P-value_T$ ($P-value$ corresponding to the Goodness-of-Fit distributional test on the $P-values$ obtained for a particular test i.e., a $P-value$ of the $P-values$) ≥ 0.0001 then $P-values$ can be considered uniformly distributed.

S. No.	Statistical Test	No. of sequences with $P-values \geq 0.01$ (Success) (m_p)	No. of sequences with $P-values < 0.01$ (Failure) (m_f)	χ^2 of distribution of $P-values$	$P-value$ corresponding to the goodness of fit ($P-value_T$)	Proportion of sequences passing the test (m_p / m)
1.	Frequency (monobit) test	1982	18	7.0152	0.635558	0.9910
2.	Runs test	1982	18	2.92	0.967382	0.9910
3.	Test for longest run of ones in a block	1979	21	21.07	0.0123432	0.9895
4.	Lempel-Ziv compression test	1977	23	22.34	0.00786166	0.9885
5.	Binary matrix rank test	1978	22	7.6	0.574903	0.9890
6.	Cumulative sums test					
	1) Forward sums test	1983	17	11.15	0.265567	0.9915
	2) Reverse sums test	1981	19	7.76	0.558502	0.9905
7.	Discrete Fourier transform (spectral) test	1990	10	32.55	0.000159908	0.9950
8.	Random excursions test					
	1) x = -4	1977	23	21.6	0.0102369	0.9885
	2) x = -3	1971	29	7.42	0.593478	0.9855
	3) x = -2	1983	17	20.74	0.0138562	0.9915
	4) x = -1	1977	23	25.54	0.00242845	0.9885
	5) x = 1	1980	20	20.35	0.0158711	0.9900
	6) x = 2	1990	10	20.93	0.0129649	0.9950
	7) x = 3	1983	17	19.91	0.018476	0.9915
	8) x = 4	1987	13	15.93	0.0683578	0.9935
9.	Random excursions variant test					
	1) x = -9	1991	9	17.31	0.044077	0.9955
	2) x = -8	1985	15	26.87	0.00146972	0.9925
	3) x = -7	1990	10	11.15	0.265567	0.9950
	4) x = -6	1988	12	10.795	0.290023	0.9940
	5) x = -5	1981	19	15.59	0.075953	0.9905
	6) x = -4	1974	26	13.78	0.130369	0.9870
	7) x = -3	1978	22	7.64	0.570792	0.9890
	8) x = -2	1979	21	11.45	0.24612	0.9895

various non-parameterized and parameterized statistical tests of NIST suite. In Figures 6(a) and (b) respectively, we have graphically depicted the computed proportions for each non-parameterized and parameterized test along with the confidence interval i.e., $[0.9833245, 0.9966745]$. It is clear that the computed proportion for each test lies inside the confidence interval; hence the tested binary sequences generated by the proposed PRBG are random with respect to all the 16 tests of NIST suite.

5 Conclusion

We have proposed a design of a pseudo random bit generator (PRBG) based on two chaotic logistic maps iterated independently starting from independent initial conditions. The pseudo random bit sequence is obtained by comparing the outputs of both the chaotic logistic maps. We have also tested rigorously the generated sequences using the NIST suite, which consists of 16 independent statistical tests devised to detect the specific characteristics expected of truly random bit sequences. The results of statistical testing are encouraging and show that the proposed PRBG has perfect cryptographic properties and hence can be used in the design of new stream ciphers.

References

[1] Schneier B. *Applied Cryptography-Protocols, algorithms and source code in C.* John Wiley & Sons, New York, USA, 1996.

[2] Menezes A.J., Oorschot P.C.V. and Vanstone S.A. *Handbook of Applied Cryptography.* CRC Press, Boca Raton, 1997.

[3] Alvarez G. and Li S. *Some basic cryptographic requirements for chaos based cryptosystems*. Int. J. of Bifur. and Chaos, vol. 16, pp. 2129-2151, 2006.

[4] Pareek N. K., Patidar Vinod and Sud K. K. *Discrete chaotic cryptography using external secret key.* Physics Letters A, vol. 309, pp. 75-82, 2003.

[5] Boccaletti S., Grebogi C., Lai Y.-C., Mancini H. and Maza D. *The control of chaos: theory and applications.* Phys. Reports, vol. 329, pp. 103-197, 2000.

[6] Schuster H. G. (Ed.) *Hand book of chaos control.* Wiley-VCH Verlag, Weinheim, Germany, 1999.

[7] Kocarev, L., Halle, K. S., Eckert, K., Chua, L. O. & Parlitz, U. *Experimental demonstration of secure communications via chaotic synchronization.* Int. J. Bifurc. Chaos, vol. 2, pp. 709-713, 1992.

[8] Wu, C. W. & Chua, L. O. *A simple way to synchronize chaotic systems with applications to secure communications systems.* Int. J. Bifurc. Chaos, vol. 3, pp. 1619–1627, 1993.

[9] Cuomo, K. M., Openheim, A. V. & Strogatz, S. H. *Synchronization of lorenz-based chaotic circuits with applications to communications.* IEEE Trans. Circuits Syst. II, vol. 40, pp. 626–633, 1993.

[10] Morgul, O. & Feki, M. *A chaotic masking scheme by using synchronized chaotic systems*. Phys. Lett. A, vol. 251, pp. 169–176, 1999.

[11] Shahruz, S. M., Pradeep, A. K. & Gurumoorthy, R.

Table 2: Non-parameterized tests results

- Number of binary sequences tested (m): 2,000
- Length of each binary sequence: 1,000,000 bits
- Significance level (α) = 0.01
- The range of acceptable proportion is [0.9833245, 0.9966745]

- Null hypothesis (H_0): The binary sequence is random
- If $P-value \geq \alpha(0.01)$ then the null hypothesis (H_0) is accepted.
- If $P-value < \alpha(0.01)$ then the null hypothesis (H_0) is rejected.
- If $P-value_T$ ($P-value$ corresponding to the Goodness-of-Fit distributional test on the $P-values$ obtained for a particular test i.e., a $P-value$ of the $P-values$) ≥ 0.0001 then $P-values$ can be considered to be uniformly distributed.

S. No.	Statistical Test	No. of sequences with $P-values \geq 0.01$ (Success) (m_p)	No. of sequences with $P-values < 0.01$ (Failure) (m_f)	χ^2 of distribution of $p-values$	$P-value$ corresponding to the goodness of fit ($P-value_T$)	Proportion of sequences passing the test (m_p/m)
1.	Frequency test within a block (Block length = 10^4)	1977	23	8.79	0.456881	0.9885
2.	Approximate entropy test (Block length = 10)	1980	20	4.08	0.906069	0.9900
3.	Linear complexity test (Block length = 10^3)	1977	23	13.92	0.1252	0.9885
4.	Maurer's universal statistical test (No. of blocks = 7, Block length = 1280)	1978	22	9.37	0.403844	0.9890
5.	Serial test (Block length = 16)	1979	21	3.65	0.932904	0.9895
6.	Overlapping template matching test (Template length = 9)	1977	23	9.65	0.379555	0.9885
7.	Non-overlapping template matching test (Template length = 9)					
	1) Template = 000000001	1987	13	9.98	0.352107	0.9935
	2) Template = 000100111	1974	26	5.8	0.759756	0.9870
	1) Template = 001010011	1981	19	10.02	0.348869	0.9905
	2) Template = 010001011	1978	22	7.39	0.596584	0.9890
	3) Template = 011101111	1981	19	12.04	0.211064	0.9905
	4) Template = 101101000	1974	26	8.27	0.507182	0.9870
	5) Template = 110100100	1981	19	8.66	0.469232	0.9905
	6) Template = 111100000	1976	24	6.01	0.738917	0.9880

Design of a novel cryptosystem based on chaotic oscillators and feedback inversion. J. Sound Vibrat., vol. 250, pp. 762–771, 2002.

[12] Halle, K. S., Wu, C. W., Itoh, M. & Chua, L. O. *Spread spectrum communication through modulation of chaos in Chua's circuit*. Int. J. Bifurc. Chaos, vol. 3, pp. 469–477, 1993.

[13] Cuomo, K. M. & Openheim, A. V. *Circuit implementation of synchronized chaos with applications to communications*. Phys. Rev. Lett., vol.71, pp. 65–68, 1993.

[14] Chen, J. Y., Wong, K. W., Cheng, L. M. & Shuai, J. W. *A secure communication scheme based on the phase synchronization of chaotic systems*. Chaos, vol. 13, pp. 508–514, 2003.

[15] Yang, T. & Chua, L. O. *Secure communication via chaotic parameter modulation*. IEEE Trans. Circuits Syst. I , vol. 43, pp. 817–819, 1996.

[16] Dedieu, H., Kennedy, M. P. & Hasler, M. *Chaos shift keying: Modulation and demodulation of a chaotic carrier using self-synchronizing*. IEEE Trans. Circuits Syst. II , vol. 40, pp. 634–641, 1993.

[17] Parlitz, U., Chua, L. O., Kocarev, L., Halle, K. S. & Shang, A. *Transmission of digital signals by chaotic synchronization*. Int. J. Bifurc. Chaos , vol. 2, pp. 973–977, 1992.

[18] Feldmann, U., Hasler, M. & Schwarz, W. *Communication by chaotic signals: The inverse system approach*. Int. J. Circuit Theory Appl. , vol. 24, pp. 551–579, 1996.

[19] Zhou, H. & Ling, X. *Problems with the chaotic inverse system encryption approach*. IEEE Trans. Circuits Syst. I, vol. 44, pp. 268–271, 1997.

[20] Habutsu, T., Nishio, Y., Sasase, I. & Mori, S. *A secret key cryptosystem by iterating a chaotic map*. in Advances in Cryptology – EUROCRYPT'91, Lecture Notes in Computer Science, vol. 547, pp. 127–140, 1991 (Spinger-Verlag).

[21] Pareek N. K., Patidar Vinod and Sud K. K. *Cryptography using multiple one-dimensional chaotic maps*. Communications in Nonlinear Science and Numerical Simulation, vol. 10, pp. 715-723, 2005.

[22] Pareek N. K., Patidar Vinod and Sud K. K. *Image encryption using chaotic logistic map*. Image and Vision Computing, vol. 24, pp. 926-934, 2006.

[23] Fridrich, J. *Symmetric ciphers based on two-dimensional chaotic maps*. Int. J. Bifurc. Chaos, vol. 8, pp. 1259–1284, 1998.

[24] Tang, G., Liao, X. & Chen, Y. A novel method for designing S-boxes based on chaotic maps. Chaos Solitons Fractals, vol. 23, pp. 413–419, 2005.

[25] Kocarev, L., Jakimoski, G., Stojanovski, T. & Parlitz, U. *From chaotic maps to encryption schemes*. in Proc. IEEE Int. Symposium Circuits and Systems (ISCAS'98), vol. 4, pp. 514–517,1998.

[26] Guo, D., Cheng, L. M. & Cheng, L. L. *A new symmetric probabilistic encryption scheme based on chaotic attractors of neural networks*. Applied Intelligence, vol. 10, pp. 71–84, 1999.

[27] Jakimoski, G. & Kocarev, L. *Chaos and cryptography: Block encryption ciphers based on chaotic maps*. IEEE Trans. Circuits Syst. I, vol. 48,

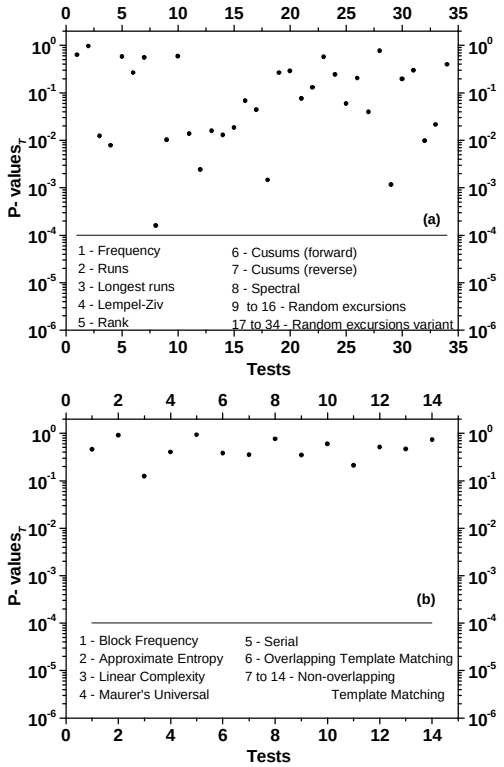


Figure 5: $P-value_T$ (i.e, *P-value* of the *P-values*) for (a) non-parameterized tests and (b) parameterized tests of NIST suite. The horizontal line represents the threshold value of $P-value_T$.

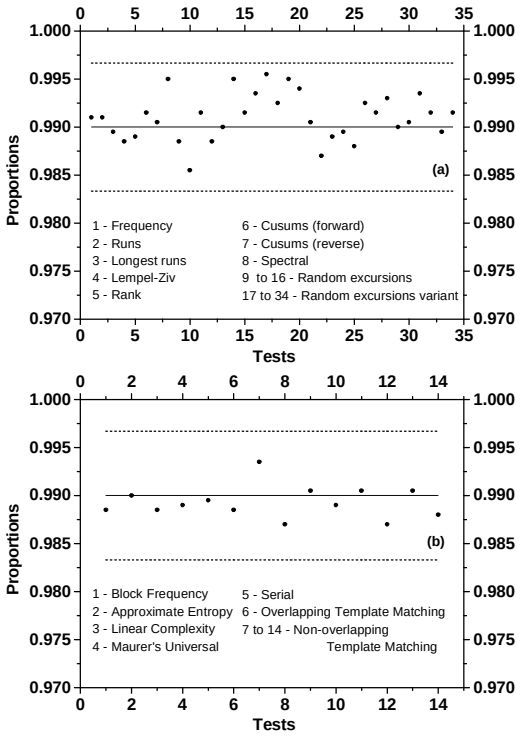


Figure 6: Proportions of the sequences passing the tests for (a) non-parameterized and (b) parameterized tests of NIST suite. The region between two horizontal dashed lines is the acceptable range of proportion.

pp. 163–169, 2001.

[28] Wolfram, S. *Cryptography with cellular automata*. in Advances in Cryptology – CRYPTO'85, Lecture Notes in Computer Science, vol. 218, pp. 429–432, 1985.

[29] Matthews, R. A. J. *On the derivation of a 'chaotic' encryption algorithm*. Cryptologia, vol. XIII, 29–42, 1989.

[30] Berstein, G. M. & Lieberman, M. A. *Method and apparatus for generating secure random numbers using chaos*. US Patent No. 5007087, 1991.

[31] Zhou, H. & Ling, X. *Generating chaotic secure sequences with desired statistical properties and high security*. Int. J. Bifurc. Chaos, vol. 7, 205–213, 1997.

[32] Li, S., Mou, X. & Cai, Y. *Pseudo-random bit generator based on couple chaotic systems and its application in stream-ciphers cryptography*. in Progress in Cryptology – INDOCRYPT 2001, Lecture Notes in Computer Science, vol. 2247, 316–329, 2001.

[33] Lee, P.-H., Pei, S.-C. & Chen, Y.-Y. *Generating chaotic stream ciphers using chaotic systems*. Chinese J. Phys. , vol. 41, 559–581, 2003.

[34] Oishi S. and Inoue H. *Pseudo-random number generators and chaos*. Transactions of the Institute of Electronics and Communication Engineers of Japan E, vol. 65, 534-541, 1982.

[35] Lin T. and Chua L. O. *New class of pseudo-random number generator based on chaos in digital filters*. International Journal of Circuit Theory and Applications, vol. 21, 473-480, 1993.

[36] Andrecut M. [1998] Logistic map as a random number generator, International Journal of Modern Physics B, vol. 12, 921-930.

[37] Gonzalez J. A..and Pino R. *Random number generator based on unpredictable chaotic functions*. Computer Physics Communications, vol.120, 109-114, 1999.

[38] Kolesov V. V., Belyaev R. V. and Voronov G. M. *A Digital random-number generator based on the chaotic signal algorithm*. Journal of Communications Technology and Electronics, vol. 46, pp. 1258-1263, 2001.

[39] Stojanovski T. and Kocarev L. *Chaos-based random number generators - Part I: Analysis.* IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, vol. 48, pp. 281-288, 2001.

[40] Stojanovski T., Pihl J. and Kocarev L. Chaos-*based random number generators - Part II: Practical realization.* IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, vol. 48, pp. 382-385, 2001.

[41] Kocarev L. and Jakimoski G. *Pseudorandom bits generated by chaotic maps*. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, vol. 50, pp. 123-126, 2003.

[42] Fu S. –M., Chen Z. –Y. and Zhou Y. –A. *Chaos-based random number generators*. Computer Research and Development, vol. 41, pp. 749-754, 2004.

[43] Huaping L., Wang S. and Gang H. *Pseudo-random number generator based on coupled map lattices*. International Journal of Modern Physics B, vol. 18, pp. 2409-2414, 2004.

[44] Li X. –M., Shen H. –B. and Yan X. –L. *Characteristic analysis of a chaotic random number generator using piece-wise-linear map.* Journal of Electronics and Information Technology, vol. 27, pp. 874-878, 2005.

[45] Liu J. *Design of a chaotic random sequence and its application*, Computer Engineering, vol.31, pp. 150-152, 2005.

[46] Wang Y., Shen H. and Yan X. *Design of a chaotic random number generator*. Chinese Journal of Semiconductors, vol. 26, pp. 2433-2439, 2005.

[47] Wang L., Wang F. –P. and Wang Z. –J. *Novel chaos-based pseudo-random number generator*. Acta Physica Sinica, vol. 55, pp. 3964-3968, 2006.

[48] Ergun S. and Ozoguz S. *Truly random number generators based on a non-autonomous chaotic oscillator*. AEU-International J. Electronics & Communications, vol. 62, pp. 235-242, 2007.

[49] Hu Y., Liao X., Wong K.-W. and Zhou Q. *A true random number generator based on mouse movement and chaotic cryptography*. Chaos Solitons and Fractals, vol. 40, pp. 2286-2293, 2009.

[50] Patidar Vinod and Sud K. K. *Anovel pseudo random bit generator based on chaotic standard map and its testing.* Electronic J. of Theoretical Physics, vol. 20, pp. 327-344, 2009.

[51] May R.M. *Simple mathematical models with very complicated dynamics*. Nature, vol. 261, pp. 459-467, 1976.

[52] Feigenbaum M. J. *The universal metric properties of nonlinear transformations.* J. Stat. Phys., vol. 21, pp. 669-706, 1979.

[53] Feigenbaum M. J. *Universal behaviour in nonlinear systems*. Los Alamos Science, vol. 1, pp. 4-27, 1980.

[54] Litchenberg A. J. and Lieberman M. A. *Regular and stochastic motion*. Springer Verlag, New York, USA, 1983.

[55] Runkin et al. *Statistical test suite for random and pseudo random number generators for cryptographic applications*. NIST special publication 800-22, 2001.

[56] Marsaglia G. *DIEHARD statistical tests, http://stst.fsu.edu/pub/diehard*, 1995.

[57] Gustafson H. et al. *A computer package for measuring the strength of encryption algorithms*, J. Computer Security, vol. 13, pp. 687-697, 1994.

[58] Knuth D. *The art of computer programming: semiemperical algorithms*. Addison Wesley, Reading, USA, 1998

[59] Sato J. *Statistical testing of random number generators*. Proceedings of 22nd National Information System Security Conference, http://csrc.ncsl.gov/nissc/1999/proceeding/papers/p24.pdf

Computational Reduction of Wilson's Primality Test for Modern Cryptosystems

Chia-Long Wu
Department of Aviation & Communication Electronics, Chinese Air Force Institute of Technology
Kaohsiung, Taiwan, R.O.C.
E-mail: chialongwu@seed.net.tw

Der-Chyuan Lou and Te-Jen Chang
Department of Electrical Engineering, Chung Cheng Institute of Technology, National Defense University
Taoyuan, Taiwan, R.O.C.
E-mail: {dclouprof, karl591218}@gmail.com

Keywords: information security, cryptography, modular arithmetic, primality test, number theory

Received: March 18, 2008

In this paper, a method of diminishing computational reduction to improve Wilson's primality test method is proposed. Basically, the RSA algorithm entails a modular exponentiation operation on large integers, which is considerably time-consuming to implement. Since ancient time, number theory has been an important study subject and modular arithmetic has also been widely used in cryptography. The Wilson's primality test method is one of the most well-known deterministic prime number test methods. It states that n *is a prime number if and only if* $(n-1)! \equiv -1 \bmod n$. *In this paper, we compare two primality test algorithms for implementing the Wilson's method, which need* $2*[(\frac{n-1}{2})!]$ *and* $n(\log_2 n)^2$ *multiplications, respectively. However, by using the proposed reduction algorithm, only* $\frac{n+1}{2}*[1-(\frac{1}{2})^k]+1$ *multiplications are needed for the Wilson's primality test method, where* $k=\left\lfloor \log_2 \frac{n-1}{2} \right\rfloor$ *and the "n" means a prime number. The proposed computational reduction method can efficiently perform Wilson's deterministic primality test, and it is faster than other proposed methods. By using the proposed method, it can not only reduce the overall computational complexity of the original Wilson's primality test method but also reduce the computational space.*

Povzetek: Opisana je metoda redukcije za moderne kriptografske sisteme.

1 Introduction

Modular exponentiation (ME) is the cornerstone computations performed in public-key cryptosystems. Taking the RSA cryptosystem [1] for example, the public and private keys are functions of a pair of large prime numbers, and the encryption and decryption operations are accomplished by modular exponentiation.

This modular exponentiation problem can be described as follows. Given M (message), E (public key), and N (the product of two large primes), compute ciphertext C ≡ ME mod N. For the computation of modular exponentiation, the very intuitive way is to break the modular exponentiation operation into a series of modular multiplications.

Meganet corporation [2] has announced its 13-year research results in the prime number testing area. Meganet corporation has implemented the algorithm in an ANSI C application running on a single CPU 450 MHZ PC. Some results of Meganet corporation are depicted in Table 1.

Wilson's primality test method states that n is a prime number if and only if $(n-1)! \equiv -1 \bmod n$. In this paper we compare two algorithms by its multiplication numbers for implementing the Wilson's primality method: Naccache-Donio's needs $2*[(\frac{n-1}{2})!]$ multiplications from a little trick about factories [3] and Rosen's method needs $[n(\log_2 n)^2]$ multiplications [4-9].

To design a fast primality test for finding a prime number is necessary and very important. We apply a method based on the modular arithmetic to advance the Wilson's primality test. The motivation of this paper is to reduce the numbers of multiplication, modular multiplication, and square. Besides, we will describe how to have a better space usage by using the proposed method.

The paper is organized as follows. In Section 2 we describe modern primality test methods such as probabilistic and deterministic primality test methods. The Wilson's primality test method and some mathematical preliminaries are introduced in Section 3, and the proposed method using modular arithmetic is described in details. In Section 4 we analyze the computational complexity and area usage for our proposed improved Wilson's primality test method and compare the performance with Naccache-Donio's method [3] and Rosen's method [4-9]. In Section 5 we draw some figures and tables to compare the above different methods.

Table 1. Experiments of Meganet corporation.

Bit number of the primality test	Time (in second)
1,000	0.5
2,000	1
3,000	3
4,000	8
5,000	15
6,000	26
7,000	41
8,000	62
9,000	87
10,000	118

2 Modern primality test methods

For the modern primality test theory [10-15], two fields of test methods have been published. They can enhance the security in public key cryptosystem such as probabilistic primality tests and deterministic primality tests. They are Solovay-Strassen, Lehman, Lucas, Miller-Rabin methods and so on, which have been issued in the probabilistic primality test field [16, 17]. We also have other primality test papers which have been issued in the deterministic test field [18-20] such as Demytko, Wilson, Proth methods etc.

2.1 Probabilistic primality test methods

2.1.1Fermat probabilistic primality test method:

This theorem assures us that if n is a prime number then $b^{n-1} \equiv 1 \bmod n$ for every integer b co-prime to n. In contrast, if n is a composite number, it is quite rare for the above congruence to be satisfied with b.

2.1.2Lucas probabilistic primality test method:

For any two nonzero integers, this equation is $D = a^2 - 4b \neq 0$. We define Lucas sequence as $U_k = \frac{\alpha^k - \beta^k}{\alpha - \beta}$, for $k \geq 0$, and α, β are two roots of the equation $x^2 - ax + b = 0$. If p is a prime number, p cannot divide b, and p will satisfy this equation $\frac{D}{p} = -1$, where $\frac{D}{p}$ is Jacobi symbol. We can get p |Up+1. So we use this principle to presume that if n is a positive odd number and n can not divide Un+1, then n is a composite number.

2.1.3 Miller-Rabin probabilistic primality test method:

Given a positive odd integer n and let $n = 2^r s + 1$, where s is an odd number. Then follow the testing numbers: choose a random positive integer a with $1 \leq a \leq n-1$. If $a^s \equiv 1 \bmod n$ or $a^{2^j s} \equiv -1$ mod N for some $0 \leq j \leq r-1$, then n passes the test. A prime number will pass the test for all a.

2.2 Deterministic primality test methods

Compared with probabilistic primality test methods, the output results of deterministic primality test methods are absolutely correct. In other words, when a positive odd number is tested, the output result has only two possible situations by using deterministic primality test methods. Either this number is a prime number or this number is a composite number. By using this method, the found number can be assumed as whether this number is a prime number or not.

2.2.1 Demytko deterministic primality test method:

If " $p_{i+1} = h_i * p_i + 1$ " meets the four following conditions, then pi+1 is sure to be a prime number.

(a) Input a positive odd prime number p_i. Let it be regarded as a seed generating prime number. We also look for them by using Look-Up Table (LUT) or other primality test methods.

(b) For h_i<4(p_i+1) *Hi*, *hi* is an even number, so we must use all of the even numbers from 2 to h_i during the test procedures.

(c) $2^{h_i p_i} \equiv 1 \bmod p_{i+1}$.

(d) $2^{h_i} \neq 1 \bmod p_{i+1}$.

2.2.2 Wilson deterministic primality test method:

If and only if n is a prime number, then (n-1)! +1 is a multiple of n, that is

$(n-1)! \equiv -1 \bmod n$.

This theorem was proposed by John Wilson and published by Edward Waring in 1770 though it was previously known for Leibniz [19]. It was proved by Lagrange [20] in 1773. Unlike the Fermat probabilistic primality test method, the Wilson's theorem is not only necessary but also sufficient for the primality test.

2.2.3 Proth deterministic primality test method:

For $N = k * 2^n + 1$ with k odd and $2^n > k$, if there exists an integer α such that

$\alpha^{(\frac{N-1}{2})} \equiv -1 \bmod N$,

then N is prime.

The difference between probabilistic primality test methods and deterministic primality test methods is that the result of the later methods can be precisely accurate. Namely, we are sure the number we calculate is a prime number.

3 Wilson's primality test method

Number theory has played an important role in the public key cryptosystems [7, 21]. In this section we will review modular arithmetic in number theory. Then we will introduce the proposed improved primality test method.

3.1 Mathematical preliminaries

As what is introduced in the above section, the Wilson's primality test method could provide an absolutely correct result, but it needs too much space and time if we calculate the "n!" directly. If our proposed algorithm is not used for Wilson's primality test method, it will need enormous operation digits and take much time. For example, when the number n = 1,597, it can nearly produce 4,500 digits decimal number result. More tests are shown in Table 2.

In this section, we review some definitions and theorems [21] which are well-suited for our proposed primality test method.

Theorem 1:

If a, b, c, and m are integers with m > 0 so that $a \equiv b \bmod m$, then

(a) $(a+b) \bmod m \equiv [(a \bmod m)+(b \bmod m)] \bmod m$,

(b) $(a-b) \bmod m \equiv [(a \bmod m)-(b \bmod m)] \bmod m$,

(c) $(a*b) \bmod m \equiv [(a \bmod m)*(b \bmod m)] \bmod m$,

(d) $(a+c) \equiv (b+c) \bmod m$,

(e) $(a-c) \equiv (b-c) \bmod m$,

(f) $(a*c) \equiv (b*c) \bmod m$.

Theorem 2:

If $a \equiv b \bmod m_1, a \equiv b \bmod m_2, ..., a \equiv b \bmod m_k$, where a, b, m1, m2, …, mk are integers with m1, m2, …, mk positive, then $a \equiv b \bmod [m_1, m_2, ..., m_k]$, where [m1, m2, …, mk] is the least common multiple of m1, m2, …, mk.

Theorem 3:

A geometric series $\Sigma_k a_k$ is a series for which the ratio of each two consecutive terms $\frac{a_{k+1}}{a_k}$ is a constant function of the summation index k. For the case of the ratio $\frac{a_{k+1}}{a_k} = r$ equals to a constant r, the terms a_k are the form $a_k = ar^k$. Sn is a summation of $a + ar + ar^2 + ar^3 + ... + ar^n$.

$$S_n = \sum_{k=0}^{n} ar^k = a + ar + ar^2 + ar^3 + ... + ar^n \quad (1)$$

Multiplying both sides by r gives

$$rS_n = r\sum_{k=0}^{n} ar^k = ar + ar^2 + ar^3 + ... + ar^n + ar^{n+1} \quad (2)$$

and subtracting (2) from (1) gives $(1-r)S_n = a - ar^{n+1}$,

so

$$S_n = \frac{a(1-r^{n+1})}{1-r} \quad (3)$$

Theorem 4:

If $(a*c) \equiv (b*d) \bmod m, c \equiv d \bmod m$, and (c, m) = 1, then $a \equiv b \bmod m$ where (c, m) represents the greatest common divisor. (c, m) = 1 means they don't have any factors between c and m except 1.

Definition:

A complete system of residues modulo *m* is a set of integers so that every integer is congruence modulo *m* to exactly one integer of the set.

Table 2. Comparison decimal digits.

Test number	The decimal digits of test number	Decimal digits in original Wilson's primality test method (*n*-1)!	Decimal digits in the proposed Wilson's primality test method
97	2	150	4
127	3	212	5
251	3	493	5
367	3	781	6
499	3	1,129	6
541	3	1,243	6
677	3	1,622	6
727	3	1,764	6
877	3	2,200	6
977	3	2,496	6
1,009	4	2,592	7
1,103	4	2,876	7
1,213	4	3,213	7
1,301	4	3,486	7
1,423	4	3,868	7
1,597	4	4,421	7

3.2 Improved Wilson's primality test method

The Wilson's primality test method is described as:

$$(n-1)! \equiv -1 \bmod n \quad (4)$$

where *n* represents a prime number.

It can be written as:

$$[(n-1)*(n-2)*(n-3)*(n-4)*(n-5)*(n-6)*(n-7)*...*5*4*3*2*1] \equiv -1 \bmod n \quad (5)$$

Equations (5) can be rewritten in details as following:

$$[(n-1)(n-2)*...*(\tfrac{n-1}{2}+2)*(\tfrac{n-1}{2}+1)*(\tfrac{n-1}{2})*(\tfrac{n-1}{2}-1)*...*5*4*3*2*1] \equiv -1 \bmod n \quad (6)$$

The following equations are based on [4] following:

$X_m \equiv b \bmod r, X_m = \prod X_i X_{i+1}$, i = 1, 3, …, m-1 where m, i, and r represent each integer. (7)

$[\prod (X_i X_{i+1} \bmod r)] \bmod r \equiv b \bmod r$, i = 1, 3, .., m-1 where m, i, and r represent each integer. (8)

If $a \equiv b \bmod m$, then $(a-m) \equiv b \bmod m$ (9)

Based on Equation (9), we subtract n from item (n-1) to item $\frac{n-1}{2}+1$, and other items remain the same in Equation (6) to obtain Equation (10) as follows.

$$\{[(n-1)-n][(n-2)-n]*...*[(\frac{n-1}{2}+2)-n]*[(\frac{n-1}{2}+1)-n]*(\frac{n-1}{2})*...*2*1\} \equiv -1 \bmod n \quad (10)$$

The result of Equation (10) can be rewritten as:

$$[(-1)*(-2)*(-3)*...*(\frac{3-n}{2})*(\frac{1-n}{2})*(\frac{n-1}{2})*(\frac{n-3}{2})*...*5*4*3*2*1] \equiv -1 \bmod n \quad (11)$$

Because *n* is an odd number, we can skip the minus symbol, and Equation (11) can be transformed to be:

$$[1*2*3*4*5...*(\frac{n-3}{2})*(\frac{n-1}{2})]^2 \equiv -1 \bmod n \quad (12)$$

In other words, we can recompose the original Wilson's primality test method [Equation (4)] as Equation (12).

Based on Equation (8), Equation (12) can be recomposed as follows.

$$\{[(1*2) \bmod n][(3*4) \bmod n]*...*[(\frac{n-3}{2}*\frac{n-1}{2}) \bmod n]\}^2 \bmod n \equiv -1 \bmod n \quad (13)$$

Based on Equation (13), we can now process each item in our proposed Wilson's primality test method, which involves multiplications and modulus inside each square bracket entry. Then we can square them in the last step. At last we use this modulus n to get the final result.

The proposed method is depicted as follows. These items inside the square bracket entries in Equation (12) can be represented in the following form:

$\prod A_{k'}A_{k'+1} = A_1*A_2*A_3*A_4*A_5*A_6*A_7*...*A_{\frac{n-1}{2}-1}*A_{\frac{n-1}{2}}$; k' = 1, 3, 5, 7, 9, ….., $\frac{n-1}{2}-1$.

Hence, Equation (13) can also be represented by using a different form as follows.

$$(\prod A_{k'}A_{k'+1}) \bmod n \equiv [(A_1*A_2) \bmod n]*[(A_3*A_4) \bmod n]*...*[(A_{\frac{n-1}{2}-1}*A_{\frac{n-1}{2}}) \bmod n]$$

,for k' = 1,3,5,7,9, ……, $\frac{n-1}{2}-1$ (14)

In this modification, based on the fundamental modular arithmetic, we need some variables during the following computational procedures to solve Equation (12).

a = 1, 3, 5, 7, 9, ……, ($\left\lceil\frac{n+1}{4}\right\rceil$-1),

b = 1, 3, 5, 7, 9, ……, ($\left\lceil\frac{n+1}{8}\right\rceil$-1),

⋮

k' = 1, 3, 5, 7, 9, ……, ($\left\lceil\frac{n+1}{2^{k+1}}\right\rceil$-1),

k =1, 2, 3, 4, 5, ……, $\left\lceil\log_2\frac{n-1}{2}\right\rceil$.

Here we use the following procedures to evaluate Equation (12).

The first procedure:

$$A_{1a} \equiv \prod A_{k'}A_{k'+1} \bmod n$$
$$\equiv [(A_1*A_2) \bmod n]*[A_3*A_4) \bmod n]*...*[(A_{\frac{n-1}{2}-1}*A_{\frac{n-1}{2}}) \bmod n] \quad (15)$$

The second procedure:

$$A_{2b} \equiv \prod A_{1a}A_{1(a+1)} \bmod n$$
$$\equiv [(A_{11}*A_{12}) \bmod n]*[A_{13}*A_{14}) \bmod n]*...*[(A_{1(\lceil\frac{n+1}{4}\rceil-1)}*A_{1(\lceil\frac{n+1}{4}\rceil)}) \bmod n] \quad (16)$$

⋮

The kth procedure:

$$A_{kk'} \equiv \prod A_{(k-1)k'}A_{(k-1)(k'+1)} \bmod n$$
$$\equiv [(A_{(k-1)1}*A_{(k-1)2}) \bmod n]*[A_{(k-1)3}*A_{(k-1)4}) \bmod n]*...*[(A_{(k-1)(\lceil\frac{n+1}{2^k}\rceil-1)}*A_{(k-1)(\lceil\frac{n+1}{2^k}\rceil)}) \bmod n] \quad (17)$$

The final procedure:

We assure $B \equiv (A_{kk'})^2 \bmod n$ (18)

If B = -1, n is a prime number.

If B ≠ -1, n is a composite number.

Complexity analyses

Now we generalize the above procedures from Equation (15) to Equation (18), and analyze the complexity of the proposed algorithm in detail as follows.

The first procedure:

($\left\lceil\frac{n+1}{4}\right\rceil$ -1) modular multiplications are needed to evaluate Equation (15).

The second procedure:

($\left\lceil\frac{n+1}{8}\right\rceil$ -1) modular multiplications are needed to evaluate Equation (16).

⋮

The kth procedure:

($\left\lceil\frac{n+1}{2^{k+1}}\right\rceil$-1) modular multiplications are needed to evaluate Equation (17).

In the above procedures, we proceed k' numbers for each procedure, where

k' = 1, 3, 5, 7, 9, ……, ($\left\lceil\frac{n+1}{2^{k+1}}\right\rceil$-1),

and we need to execute k procedures, where

k =1, 2, 3, 4, 5, ……, $\left\lceil\log_2\frac{n-1}{2}\right\rceil$.

The final procedure:

One modular square is needed to evaluate Equation (18).

To simplify the discussions in this paper, the modular operation is ignored and only the multiplication and the square are referred to [20, 23]. So we can sum up the computational amounts (the number of modular multiplication and modular square) in all of the above procedures below.

$$(\left\lceil\frac{n+1}{4}\right\rceil-1)+(\left\lceil\frac{n+1}{8}\right\rceil-1)+(\left\lceil\frac{n+1}{16}\right\rceil-1)+(\left\lceil\frac{n+1}{32}\right\rceil-1)+(\left\lceil\frac{n+1}{64}\right\rceil-1)+...+(\left\lceil\frac{n+1}{2^{k+1}}\right\rceil-1)+1 \quad (19)$$

where k = 1, 2, 3, …, $\left\lceil\log_2\frac{n-1}{2}\right\rceil$

We rearrange the above equation as follows.

$$[(\frac{n+1}{4}+1-1)+(\frac{n+1}{8}+1-1)+(\frac{n+1}{16}+1-1)+(\frac{n+1}{32}+1-1)+...+(\frac{n+1}{2^{k+1}}+1-1)]+1 \quad (20)$$

where "+1" inside each parenthesis means we get the maximum for each item, which marks ceiling symbol in Equation (19).

Based on Theorem 3, we calculate Equation (20) to obtain the final result as follows.

$\frac{1}{2}*(n+1)*[1-(\frac{1}{2})^k]+1$, where $k=\left\lceil\log_2\frac{n-1}{2}\right\rceil$. (21)

The original Wilson's primality test method is (*n*-1)! ≡ -1 mod *n*. From Table 2, we know the larger the test

number is, the larger the decimal-digit size is. However, by using the proposed algorithm, the maximum decimal-digit size is generated by the (n-1)*(n-2) item. Note, this item should be bounded to 2q if we assume that the test number "n" has q decimal-digit size. Some experimental results are shown in Table 2.

4 Example

Let us take n = 29 to depict our proposed Wilson's primality test method and show the correctness of the proposed method. The Wilson's primality test method is based on Equation (4):

$(n-1)! \equiv -1 \bmod n$, where n represents a prime number. Based on Equation (5) to Equation (6), and Equation (10) to Equation (12), the original Wilson's primality test method can be changed as follows.

$$(1*2*3*4*5*6*...*11*12*13*14)^2 \bmod 29 \equiv -1 \bmod 29 \quad (22)$$

Based on Equation (8) and Equation (22), our proposed method executes basically the following steps:

The first step,

$$\{[(1*2) \bmod 29]*[(3*4) \bmod 29]*...*[(11*12) \bmod 29]*[(13*14) \bmod 29]\}^2 \bmod 29 \equiv -1 \bmod 29$$

$$\Rightarrow \{[2*12*1*27*3*16*8] \bmod 29\}^2 \bmod 29 \equiv -1 \bmod 29 .$$

The second step,

$$\{[(2*12) \bmod 29]*[(1*27) \bmod 29]*[(3*16) \bmod 29]*[(8) \bmod 29]\}^2 \bmod 29 \equiv -1 \bmod 29$$

$$\Rightarrow \{[24*27*19*8] \bmod 29\}^2 \bmod 29 \equiv -1 \bmod 29 .$$

The third step,

$$\{[(24*27) \bmod 29]*[(19*8) \bmod 29]\}^2 \bmod 29 \equiv -1 \bmod 29$$

$$\Rightarrow \{[10*7] \bmod 29\}^2 \bmod 29 \equiv -1 \bmod 29 .$$

The fourth step,

$$\{[10*7] \bmod 29\}^2 \bmod 29 \equiv -1 \bmod 29$$

$$\Rightarrow \{12 \bmod 29\}^2 \bmod 29 \equiv -1 \bmod 29 .$$

The fifth step,

$$\{12\}^2 \bmod 29 \equiv -1 \bmod 29$$

$$\Rightarrow \{144 \bmod 29\} \equiv -1 \bmod 29 .$$

From the first step to the fifth step, the proposed Wilson's primality test method requires 7, 4, 2, 1 modular multiplication and one modular square, respectively. To sum up, the whole evaluation of the proposed Wilson's primality test method requires 14 modular multiplications and one modular square.

5 Conclusions and future works

In this paper, we apply modular arithmetic to improve the original Wilson's primality test method for reducing the computational complexity and getting a better area usage. Compared these criterions depicted in [3] [4] with the proposed algorithm in this paper, we can clearly understand that the test number "n" becomes larger and the other two methods will require much space and time as shown in Figure 1 and Table 3. They become infeasible. By using our proposed algorithm, even though n grows larger, the space and time we require can be still under control and save much more.

In the future, we will try to further effectively improve the Wilson's primality test method by transforming integer n from decimal number system into binary system [21, 27, 28] and reduce the redundant computational complexity [29-31]. Secondly, starting from many studies emphasized in this field [32-33], we will further study and search for more efficient methods and useful mathematical theorem to speed up the Wilson's primality test method. To sum up, we can therefore perform this deterministic primality test method more effectively when applying it in modern cryptosystem.

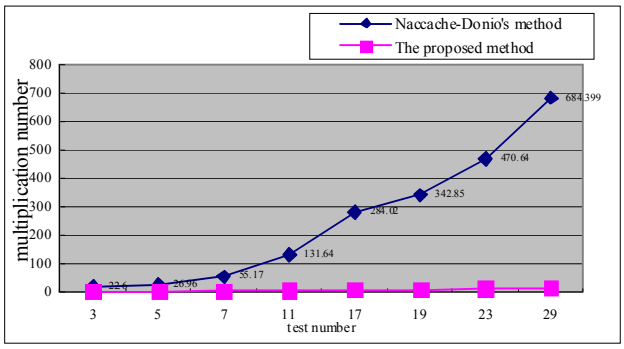


Figure 1. Complexity comparisons between Naccache-Donio's method and the proposed method.

Table 3. Complexity comparisons using smaller test number "n".

n	Naccache-Donio's method	Rosen's method	The proposed method
3	2	22.6	2
5	4	26.96	2.5
7	12	55.17	3
11	240	131.64	6.25
17	80,640	284.02	8.875
19	725,760	342.85	10.375
23	79,833,600	470.64	12.25
29	174,356,582,400	684.399	15.0625

Naccache-Donio's method: $2*[(\frac{n-1}{2})!]$.

Rosen's method: $[n(\log_2 n)^2]$.

The proposed method: $\frac{n+1}{2}*[1-(\frac{1}{2})^k]+1$, where $k=\left\lceil \log_2 \frac{n-1}{2} \right\rceil$.

References

[1] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, Oct. 1978.

[2] http://www.meganet.com/primality.htm.

[3] D. Naccache and M. Donio, "Accelerating Wilson's Primality Test," *Revue Technique Thomson-CSF*, vol. 23, no. 3, pp. 595-599, 1991. http://library.wolfram.com/search/.

[4] K.-H. Rosen, *Elementary Number Theory and Its Applications*, 3rd Ed., Addison-Wesley, 1988.

[5] M.-R. Schroeder, *Number Theory Science and Communication with Applications in Cryptography*, Berlin, N. Y. Springer-Verlag, 1986.

[6] R. Kumanduri and C. Romero, *Number Theory with Computer Applications*, Upper Saddle River, N. J. Prentice Hall, 1998.

[7] W. Stallings, *Cryptography and Network Security Principles and Practice*, 3rd Ed., Prentice-Hall, 2003.

[8] I. Koren, A.-K. Peters, and M.-A. Natick, *Computer Arithmetic Algorithms*, 2nd Ed., 2002.

[9] S.-S. Wagstaff, *Cryptanalysis of Number Theoretic Ciphers*, CRC Press Chapman & Hall, 2003.

[10] M. Agrawal, "On derandomizing tests for certain polynomial identities," *Proceedings of 18th IEEE Annual Conference on Computational Complexity*, 2003, vol. 7-10, pp. 355-359.

[11] E.-W. Weisstein, "Primality testing is easy," *MathWorld Headline News*, Aug. 7, 2002. http://mathworld.wolfram.com/news/2002-08-07/primetest/.

[12] M. Agrawal, N. Kayal, and N. Saxena, "Primes in *P*," *Preprint*, Aug. 6, 2002.

[13] D.-J. Bernstein, "An exposition of the Agrawal-Kayal-Saxena primality-proving theorem," http://cr.yp.to/papers/aks.ps, 2002.

[14] D. Mukhopadhyay and D. Roy Chowdhury, "An efficient end to end design of Rijndael cryptosystem in 0.18 μ CMOS", *Proceedings of the 18th International Conference on VLSI Design*, pp. 405-410, Jan. 2005.

[15] J. Linn, "Technology and web user data privacy: a survey of risks and countermeasures," *IEEE Security & Privacy Magazine,* vol. 3, no. 1, pp. 52-58, Jan.-Feb. 2005.

[16] R. Silverman, "Massively distributed computing and factoring large integers," *Communications of the ACM*, vol. 34, no. 11, pp. 95-103, 1991.

[17] M. Rabin, "Probabilistic algorithm for testing primality," *Journal of Number Theory*, vol. 12, pp. 128-138, 1980.

[18] http://mathworld.wolfram.com/

[19] N. Demytko, "Generating multi-precision integers with guaranteed primality," *IFIP,* Elsevier Science publishers, North-Holland, 1989.

[20] http://scienceworld.wolfram.com/

[21] D.-E. Knuth, *The Art of Computer Programming*, vol. 2: Seminumerical Algorithms, 3rd Ed., Addison-Wesley, 1998.

[22] D.-C. Lou, C.-L. Wu, and R.-Y. Ou, "Application of parallel virtual machine framework to the strong prime problem," *International Journal of Computer Mathematics*, vol. 79, no. 7, pp. 797-806, June 2002.

[23] D.-C. Lou and C.-C. Chang, "Fast exponentiation method obtained by folding the exponent in half," *IEE Electronics Letters*, vol. 32, no. 11, pp. 984-985, May 1996.

[24] C.-W. Chou, "Parallel implement of the RSA public-key cryptosystem," *International Journal Computer Mathematics*, vol. 78, no.5, pp. 153-155, 1993.

[25] M. Joye and S.-M. Yen, "Optimal left-to-right binary signed-digit recoding," *IEEE Transactions on Computers*, vol. 49, no. 7, pp. 740-748, July 2000.

[26] A. Arora and R. Telang, "Economics of software vulnerability disclosure", *IEEE Security & Privacy Magazine,* vol. 3, no. 1, pp. 20-25, Jan.-Feb. 2005.

[27] X. Ruan and R.-S. Katti, "Left-to-right optimal signed-binary representation of a pair of integers," *IEEE Transactions on Computers,* vol. 54, no. 2, pp. 124-131, Feb. 2005.

[28] M.-E. Kaihara and N. Takagi, "A hardware algorithm for modular multiplication/division," *IEEE Transactions on Computers,* vol. 54, no. 1, pp. 12 – 21, Jan. 2005.

[29] D.-C. Lou and C.-L. Wu, "Parallel exponentiation using common multiplicand multiplication and signed-digit-folding techniques," *International Journal of Computer Mathematics* , vol. 81 , no. 10 pp. 1187-1202, June 2004.

[30] C.-L. Wu, D.-C. Lou, and T.-J. Chang, "An efficient Montgomery exponentiation algorithm for cryptographic application," *Informatica – An International Journal*, vol. 16, no. 3, pp. 449-468, Sept. 2005.

[31] C.-L. Wu, D.-C. Lou, T.-J. Chang, and S.-Y. Chen, "Integer factorization for RSA cryptosystem under a PVM environment," *International Journal of Computer Systems Science & Engineering*, vol. 1, no. 2, pp. 25-35, Jan. 2007.

[32] C.-L. Wu, D.-C. Lou, and T.-J. Chang, "Fast parallel exponentiation algorithm for RSA public-key cryptosystem," *Informatica–An International Journal*, vol. 17, no. 3, pp. 445-462, Sept. 2006.

[33] D.-C. Lou,, J.-C. Lai, C.-L. Wu, and T.-J. Chang, "An efficient Montgomery exponentiation algorithm by using signed-digit-recoding and folding techniques," *Applied Mathematics and Computation*, vol. 185, no. 1, pp. 31-44, Feb. 2007.

Multi-attribute Decision Analysis in GIS: Weighted Linear Combination and Ordered Weighted Averaging

Samo Drobne and Anka Lisec
University of Ljubljana, Faculty of Civil and Geodetic Engineering,
Jamova 2, SI-Ljubljana, Slovenia
E-mail: {samo.drobne, anka.lisec}@fgg.uni-lj.si

Keywords: multi-criteria decision analysis, multi-attribute decision analysis, weighted linear combination method, WLC, ordered weighted averaging method, OWA

Received: July 8, 2009

Decision analysis can be defined as a set of systematic procedures for analysing complex decision problems. Differences between the desired and the actual state of real world geographical system is a spatial decision problem, which can be approached systematically by means of multi-criteria decision making. Many real-world spatially related problems give rise to geographical information system based multi-criteria decision making. Geographical information systems and multi-criteria decision making have developed largely independently, but a trend towards the exploration of their synergies is now emerging. This paper discusses the synergistic role of multi-criteria decisions in geographical information systems and the use of geographical information systems in multi-attribute decision analysis. An example is provided of analysis of land use suitability by use of either weighted linear combination methods or ordered weighting averages.

Povzetek: V prispevku predstavljamo porabo tehnologije GIS pri večkriterijskih odločitvenih postopkih.

1 Introduction

Decision making is based on numerous data concerning the problem at hand. It has been estimated that 80% of data used by managers and decision makers are geographical in nature [37]. Decision problems that involve geographical data are referred to as *geographical* or *spatial decision problems* [21].

Informed decision making and problem solving rely on the effective communication and exchange of ideas and information, the type and amount of information available and necessary to tackle a particular decision problem being related to the complexity of the situation. Spatial decision problems often require that a large number of feasible alternatives be evaluated on the basis of multiple criteria. Spatial decisions are multi-criteria in nature [4, 26, 28].

The types of decision problems that are referred to as geographical involve a large set of feasible alternatives and multiple conflicting and incommensurate evaluation criteria. Accordingly, many real world spatial problems give rise to multi-criteria decision making (MCDM) based on geographical information system (GIS). These two distinct areas of research, GIS and MCDM, can benefit from each other. GIS techniques and procedures have assumed an important position in decision making in the sense that they offer unique capabilities for automating, managing, and analysing a variety of spatial data for decision making. On the other hand, MCDM and a wide range of related methodologies, such as multi-objective decision making (MODM), multi-attribute decision making (MADM), multi-attribute utility theory (MAUT), public choice theory, and collaborative decision making, offer a rich collection of techniques and procedures with which to reveal decision makers' preferences and allowing their incorporation into GIS-based decision making [21].

Spatial multi-criteria decision analysis can be thought of as a process that combines and transforms geographical data (input) into a resultant decision (output). Geographical information can be defined as georeferenced data that has been processed into a form meaningful to the recipient. The data in geographical information systems are most commonly organized by separate thematic maps or sets of data, referred to as a *map layer, coverage* or *level*. The alternative to the layer approach is object-oriented GIS, where the objects are intended to closely represent real world elements. Irrespective of spatial data organisation, the ultimate aim of GIS is to provide support for spatial decisions. The multi-criteria decision-making procedures define a relationship between "input maps" and "output maps" [21].

Maps have a long history of use in support of decision making. Ever since they first appeared as a means of navigation, they were also used as a form of decision support tool. Good maps often meant the differences between success and failure and it is not unusual to find that maps have played a very important role in modern decision making. The GIS environment

allows aggregation of qualitative and quantitative geo-referenced data [14]. In this paper, the GIS capabilities for supporting spatial decisions are analyzed in the context of the major phases of the decision-making process, each stage of which requires different types of information. Tools such as GIS offer a unique opportunity to tackle spatial problems traditionally associated with more efficient and effective data collection, analysis, and alternative evaluation.

Two methods of multi-criteria evaluation (MCE) in GIS, the Weighted Linear Combination (WLC) and the Ordered Weighted Average (OWA) methods are discussed. A generalised framework of GIS-based spatial decision-making procedure is defined and following the procedure proposed, an example of multi-attribute decision analysis (MADA) in GIS is performed using both WLC and OWA. A comparison of these two different approaches has been made, based on results of land-use suitability analysis for the study area, the municipality of Ig, Slovenia.

2 Multi-criteria decision making and GIS

2.1 Multi-criteria decision making

It is generally assumed that multi-criteria decision analysis (MCDA) originated at the beginning of 1960s. Most of practitioners of MCDA consider that their field stems largely from the early work on goal programming and research of Simon [35]. He suggests a structure for analyzing human decision-making processes by distinguishing between the *intelligence*, *design*, and *choice* phases.

Any decision-making process begins with the recognition of the problem to be decided. In the intelligence phase, a situation is examined for conditions calling for a decision. In the design phase, decision makers develop alternative solutions to the decision problem already identified. Typically, a formal model is used to support a decision maker in determining the set of alternatives. In the choice phase, decision makers evaluate the decisions and choose the best alternative. In the context of decision problems with a spatial connotation, the potential for application of spatially enabled methods in Simon's decision phases has already been examined [21]. While the intelligence and design activities can mostly be covered by multi-purpose spatial analysis methods, the choice phase requires specific methods still absent from most GIS [2, 21, 24, 25, 32].

The choice phase requires formal methods (decision rules) to select feasible alternatives and to rank them with respect to the decision-makers' preferences. As humans tend to base rational decisions on an assessment of multiple decision criteria, MCDA methods have become important tools in management sciences and operations research. By incorporating quantifiers (i.e. the relative importance of different criteria) for the decision-maker's preferences, these types of decision rules are capable of solving semi-structured decision problems.

2.2 Geographical information and GIS

Most of definitions of GIS focus on two aspects: technology and/or problem solving. The technological approach defines GIS as a set of tools for the input, storage and retrieval, manipulation, analysis and output of spatial data. This approach however ignores the problem solving aspects of GIS and it has been argued that GIS functionality can play a crucial role in a comprehensive decision-making process [11, 12, 13, 20, 21].

GIS have the ability to perform numerous tasks utilizing spatial and attribute data. Such functions distinguish GIS from other management information systems. Furthermore, GIS as an integrated technology allows for integration of a variety of geographical technologies (such as remote sensing, global positioning systems, computer-aided design, automated mapping and facilities management) that can be in turn integrated with analytical and decision-making techniques. The way in which data are entered, stored and analyzed must mirror the way in which information will be used for analysis or decision-making tasks. GIS should therefore be viewed as a process rather than as merely software or hardware. The system possesses a set of procedures that facilitate the data input, data storage, data manipulation and analysis, and data output to support decision-making activities [13].

In general, a GIS has three main components and is a *computer system* that includes hardware, software and appropriate procedures (or techniques and orders for task implementation). In addition, GIS are distinguished by their use of *spatially (geographically) referenced* data, and for carrying out various *management and analysis* tasks on these data. By allowing data to be organised, presented and analyzed efficiently, by integrating them with other data and by the creation of new data that can be operated on in turn, GIS creates useful information t which can help decision making [14]. Geographical information can be defined as georeferenced data that has been processed into a form that is meaningful to the recipient decision-maker and which is of real or perceived value in the decision-making process. In general, the MCDA in GIS should be viewed as a process of conversion of data to information that adds extra value to the original data [21, 22].

2.3 Multi-criteria decision problems

Multi-criteria decision-making problems can be classified on the basis of the major components of multi-criteria decision analysis: *multi-objective* decision making (MODM) versus *multi-attribute* decision making (MADM), *individual* versus *group* decision-maker problems, and decision under *certainty* versus decision under *uncertainty*. The distinction between MODM and MADM is based on the classification of evaluation criteria into attributes and objectives.

A *criterion* is the basis for a decision and can be measured and evaluated. In case of the spatial decision problem, attributes are the properties of geographical entities. More specifically, an attribute is a measurable

quantity or quality of a geographical entity or a relationship between geographical entities. In the context of a decision-making problem, the entities and the relationships are referred to as the objects of decisions.

Multi-attribute decision making methods are data-oriented. An *attribute* is a concrete descriptive value, a measurable characteristic of an entity, including inter-entity relationships. Multi-attribute techniques are referred to as discrete methods because they assume that the number of alternatives is explicit. Multi-attribute decision problems require that choices be made among alternatives described by their attributes. This implies that attribute-objective relationships are specified in such a form that attributes can be regarded as both objectives and decision variables. Attributes are used as both decision variables and decision criteria [21].

An *objective* is a more abstract variable with a specification of a relative desirability of the levels of that variable. The multi-objective methods are mathematical programming model-oriented, where the alternatives, identified by solving a multi-objective mathematical programming problem, must be generated [16]. Multi-objective methods define the set of alternatives in terms of a decision model consisting of two or more objective functions and a set of constraints imposed upon the decision variables. The model implicitly defines the alternatives in terms of decision variables. Multi-objective models are often approached by converting them to a single objective problem solvable by standard linear/integer programming methods [33]. It is significant however, that the definition of "objective" is somewhat broader than is typically encountered in the mathematical programming literature. In mathematical programming, the term objective is often used to refer to a specific objective function. An objective is a statement about the desired state of the system under consideration and includes purposes and perspectives of a decision making. It serves as the defining role as to how the decision is structured. Purposes define the number of alternatives to be considered and the nature of decision set; perspective determines the decision rule: what criteria will be chosen, how they are evaluated, and how the final decision is made [8].

Objectives are functionally related to, or derived from, a set of attributes. An objective indicates the directions of improvement (change) of one or more attributes. For a given objective, several different attributes might be necessary to provide a complete assessment of the degree to which the objective might be achieved. If there is a direct correspondence between attributes and objectives, the multi-objective problem becomes a multi-attribute problem. In multi-attribute decision analysis, attributes are used both as decision variables and decision criteria. Generally speaking, MADM approaches are searched-based approaches and in GIS they use raster-based data structure, while MODM are choice-based approaches and use vector-based data structure [21, 23].

2.4 Framework for spatial decision making

Decision making is a sequence of activities starting with decision problem recognition and ending with a recommendation, and eventually with a final choice of alternative. As the storage and processing capacity of human memory is limited, humans develop simplifying cognitive shortcuts or processing rules to solve complex problem [5]. There being a number of alternative ways to organize the sequence of activities in the decision-making process, the quality of the decision making arguably depends on the sequence in which the activities are undertaken [21].

According to Kenney [19], two major approaches include the alternative-focus approach, which focuses on generating decision alternatives, and the value-focus approach, which uses the values (evaluation criteria) as a fundamental element of the decision analysis. The differences between these two approaches are related to the question of whether alternatives should be generated first followed by specification of the value structure, or conversely, the alternatives should be derived from the value structure (Figure 1). The general principle for structuring the decision-making process is that decision alternatives should be generated in such a way that the values specified for the decision situation are best achieved [19].

Any decision-making process begins with the recognition and definition of the decision problem, which is the perceived difference between the desired and existing states of a system. The intelligence phase of decision-making involves searching the decision environment for conditions requiring a decision: raw data are obtained, processed, and examined for clues that may identify opportunities or problems (Figure 1).

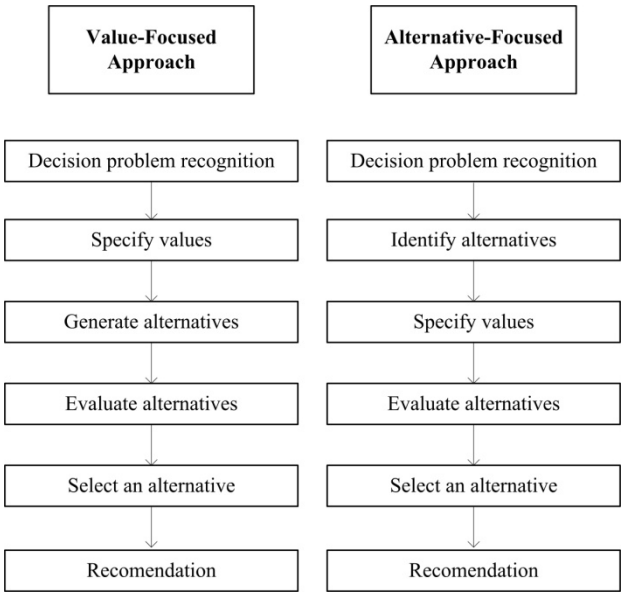


Figure 1: The sequences of alternative- and value-focused approaches (based on Kenney [19]).

A significant proportion of human problems have a geographical component. Decision making as a scientific discipline has a much longer history than GIS. Within the

wider field of decision research, computers have been used to develop decision-support systems (DSS). GIS has been referred to as a specific kind of decision-support system dealing with problems which involve a high degree of spatiality [14] and which can provide a framework for the development of spatial decision-support system (SDSS), particularly when coupled either loosely or tightly coupled with other model software. Spatial decision-support system and decision-support system share the same characteristics but the former (SDSS) presents in fact an extension of DSS (Figure 2).

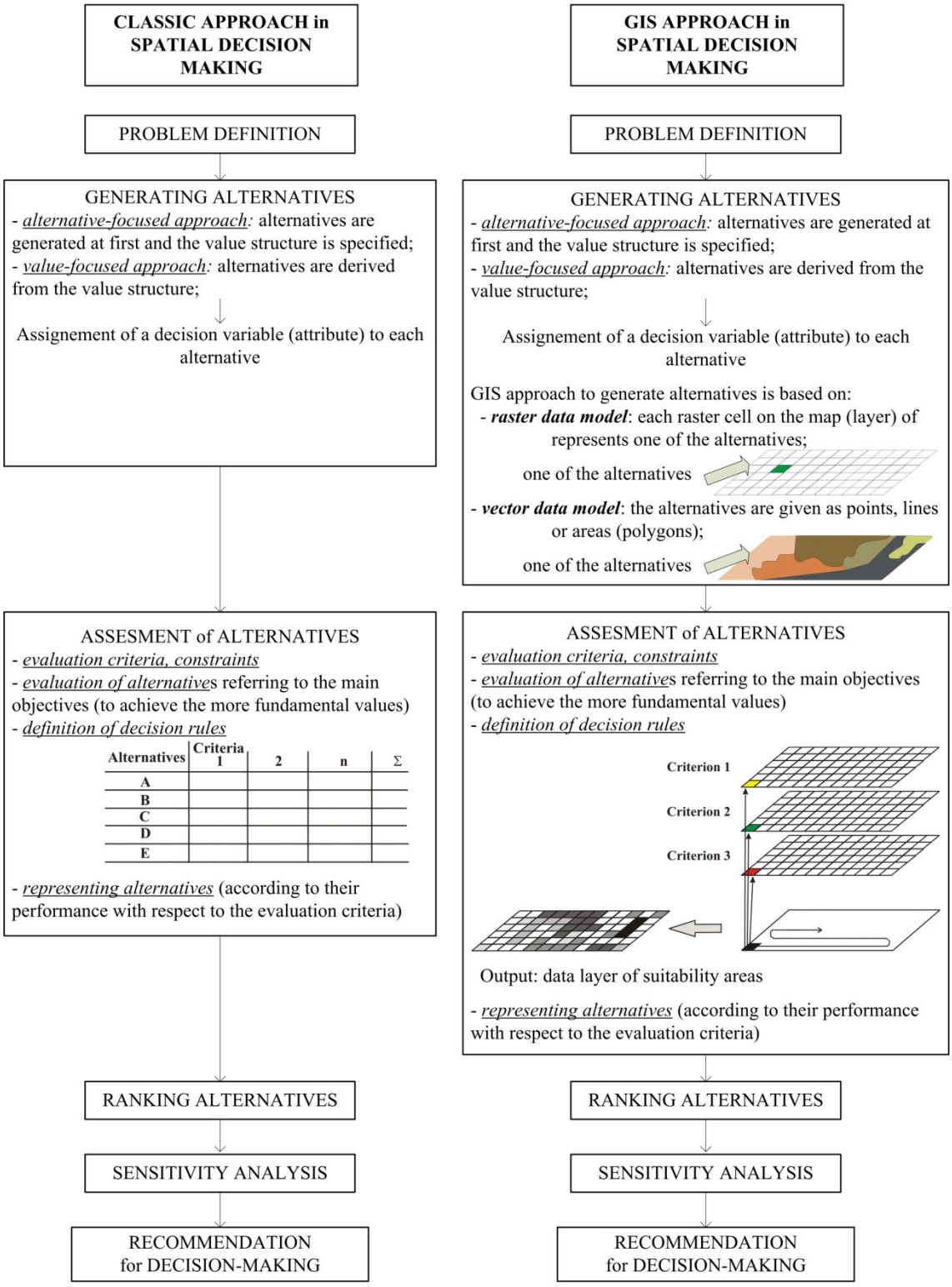


Figure 2: Classic and GIS-based spatial decision-making procedures.

One of the most important rules governing the use of GIS for SDSS is that GIS themselves do not make decisions – people do. In SDSS the emphasis is on the use of spatial data and in GIS it is on supporting decision makers in the decision-making process to choose the alternative (decision) which is the best solution to the problem that needs to be solved. Multi-criteria spatial decision support systems (MC-SDSS) integrate GIS-based data processing and analysis techniques and multi-criteria decision analysis. MC-SDSS, which is discussed more in detail below, can be viewed as a part of a broader field of spatial decision support systems.

Figure 2 shows the sequence of actions in classic spatial decision making and in GIS-based spatial decision making. Although the alternative-focused approach is mentioned in connection with the generation of alternatives, values are in general more fundamental than alternatives with respect to a decision problem. In other words, alternatives are the means to achieving the more fundamental values. Once the decision problem is defined, the spatial multi-criteria analysis focuses on evaluation criteria, which means specifying a comprehensive set of objectives that reflects all concerns relevant to the decision problem, and measures for achieving those objectives. Such measures are called *attributes*. A measurement scale must be established for each attribute. The degree, to which the objectives are met, as determined by the attributes, is the basis for comparing alternatives. The evaluation criteria are associated with geographical entities and relationships between entities and therefore can be represented in the forms of maps (a raster or a vector model). GIS data-handling and analysis capabilities are used to generate inputs to spatial multi-criteria decision analysis [21].

During the process of multi-criteria decision making, a decision variable is assigned to each alternative. Variables or "attributes" are used by the decision maker to measure the performance of alternative decisions. With respect to the evaluation criteria, the decision maker's preferences are incorporated into the decision model. The preferences are typically expressed in terms of the weights or relative importance assigned to the evaluation criteria under consideration. Given the set of alternatives, attributes and associated weights, the input data can be organized in the form of decision matrix or table. Eventually, the one-dimensional measurements (in GIS geographic data layers) and judgments (preferences and uncertainty) must be integrated to provide an overall assessment of alternatives. This is accomplished by an appropriate decision rule or aggregation function. Since a decision rule provides an ordering of all alternatives according to their performance with respect to the set of evaluation criteria, the decision problem depends on the selection of the best outcome (or an ordered set of outcomes) and the identification of the decision alternatives leading to this outcome [21].

After obtaining a ranking of alternatives, sensitivity analysis should be performed to determine robustness. This is aimed at identifying the effects of changes in the inputs (geographical data and the decision makers' preferences) on the outputs (ranking of alternatives). It helps to learn how the various decision elements interact to determine the most preferred alternative and which elements are important sources of disagreement among decision makers or interest groups. Spatial decision making typically involves a large numbers of alternatives evaluated on the basis of multiple, possibly conflicting criteria, and some systematic method of identifying the best alternatives (or classifying or ranking alternatives) is required.

The final result of a decision-making process is a recommendation for future action. The decision or recommendation should be based on the ranking of alternatives and the sensitivity analysis. It may include the description of the best alternative or a group of alternatives considered candidates for implementation. Visualisation techniques such as maps are of major importance in presenting and communicating the results to decision makers and interest groups [21].

3 Multi-attribute decision analysis

GIS-based multi-criteria decision analysis can be thought of as a process that combines and transforms spatial data into a resultant decision. The MCDM procedures are decision rules which define a relationship between the input maps and an output map. The procedures use geographical data, the decision maker's preferences, data manipulation, and preferences according to decision rules. Two considerations of critical importance for spatial MCDA are the GIS capabilities of data acquisition, storage, retrieval, manipulation and analysis, and the MCDM ability to combine the geographical data and the decision maker's preferences into one-dimensional values of alternative decisions [22].

There are many ways in which decision criteria can be combined in MCDA. A *Weighted linear combination* (*WLC*) and its variants [3, 7, 9, 31] requires summation of the weighted criteria. The *Analytical hierarchical process* (*AHP*), an adoption of WLC, can be used in two distinctive ways within the GIS environment: first, it can be employed to derive the weights associated with criteria map layers, and second, the AHP principle can be used to aggregate the priority for all hierarchical levels including the level representing alternatives [1, 34]. *Concordance-discordance analyses* are methods in which each pair of alternatives, represented as raster pixels or polygons, is analysed for the degree to which one outranks the other in the specified criteria [3, 18, 27, 29, 36]. The *ideal point methods* avoid some of the difficulties associated with the multi-attribute methods [15, 30]. These approaches order a set of alternatives on the basis of their distance from an ideal point. Recently, Malczewski [23] established a very good body of literature on GIS-based multi-criteria decision analysis.

Over the last decade, a number of multi-attribute (or multi-criteria) evaluation methods have been introduced in the GIS environment. Among these procedures, the WLC and Boolean overlay operations, such as intersection (AND) and union (OR), are considered the most straightforward and the most employed in the GIS environment [22]. The Ordered Weighted Averaging

(OWA) and its variant, Weighted Linear Combination (WLC) are discussed in this paper.

3.1 Weighted linear combination (WLC)

Weighted linear combination, or simple additive weighting, is based on the concept of a weighted average in which continuous criteria are standardized to a common numeric range, and then combined by means of a weighted average. The decision maker assigns the weights of relative importance directly to each attribute map layer. The total score for each alternative is obtained by multiplying the importance weight assigned to each attribute by the scaled value given for that attribute to the alternative and then summing the products over all attributes. The scores are calculated for all of the alternatives and that with the highest overall score is chosen. The method can be executed using any GIS system with overlay capabilities, and allows the evaluation criterion map layers to be combined in order to determine the composite map layer which is output. The methods can be implemented in both raster and vector GIS environments. Some GIS systems, e.g. Idrisi [9], have built-in routines for the WLC method, and there are available freeware modules or scripts, e.g. for ArcGIS [2], to perform that kind of MCDA of this sort.

With the weighted linear combination, factors are combined by applying a weight to each followed by a summation of the results to yield a suitability map:

$$S = \sum w_i x_i \tag{1}$$

where S is suitability, w_i is weight of factor i, and x_i is the criterion score of factor i. In cases, where Boolean constraints also apply, the procedure can be modified by multiplying the suitability calculated from the factors by the product of the constraints:

$$S = \sum w_i x_i \cdot \prod c_j \tag{2}$$

where c_j is the criterion score of the constraint j.

All GIS software systems provide the basic tools for evaluation of such a model [9].

3.1.1 Standardization of criterion scores

The first step in this process is digital GIS database development. Because criteria are measured on different scales, it is necessary that factors be standardized before combination, and that they be transformed, if necessary, so that all factor maps are positively correlated with suitability.

Voogd [36] reviewed a variety of procedures for standardization, typically using the minimum and maximum values as scaling points. The simplest is a *linear scaling* such as:

$$x_i = \frac{(R_i - R_{\min})}{(R_{\max} - R_{\min})} \cdot SR \tag{3}$$

where R_i is the raw score of factor i, $R_{\min}$ is the minimum score, $R_{\max}$ the maximum score, and SR is the standardized range.

The process of standardizing evaluation criteria can be seen also as one of recasting values into a statement of set membership [7, 9]. If the continuous factors are really fuzzy sets, this is easily recognizable as just one of many possible set membership functions. Eastmann [7] suggested the standardization of factors using a range of *fuzzy set membership functions* to either a 0-1 real number scale or a 0-255 byte scale. The latter option is recommended because it optimizes the computation. Importantly, the higher value of the standardized scale must represent the case of being more likely to belong to the decision set. Besides this deterministic (linear scaling) and fuzzy approach, there are other processes for standardizing evaluation criteria, such as the *value/utility function approach*, and the *probability approach* [21].

A critical issue in the standardization of factors is the choice of the end points at which set membership reaches either 0.0 or 1.0 (0 or 255). Blind use of linear scaling (or indeed any other scaling) between the minimum and maximum values of the image is ill advised. In setting these critical points for the set membership function, it is important to consider their inherent meaning.

3.1.2 Evaluation of criterion weights

MCDM problems involve criteria of varying importance to decision makers and information about the relative importance of the criteria is required. This is usually obtained by assigning a weight to each criterion. The derivation of weights is a central step in defining the decision maker's preferences. A weight can be defined as a value assigned to an evaluation criterion indicative of its importance relative to other criteria under consideration. The larger the weight, the more important is the criterion in the overall utility [21].

A variety of techniques exist for the development of weights. In very simple cases, assignment of criteria weights may be accomplished by dividing 1.0 among the criteria. When more than a few criteria are involved and many considerations apply, it becomes difficult to make weight evaluations on the set as a whole. The weights are then usually normalized so that they sum to 1. In the case of n criteria, a set of weights is defined as follows:

$w = (w_1, w_2, ..., w_j, ..., w_n)$, and

$\sum w_j = 1$.

There are four main groups of techniques for the development of weights [21]:

- *ranking methods*, which are the simplest methods for assessing the importance of weights: every criterion under consideration is ranked in the order of the decision maker's preferences;
- *rating methods,* which require the estimation of weights on the basis of predetermined scale;
- *pairwise comparison methods,* which involve pairwise comparison to create a ratio matrix;
- *trade-off analysis methods,* which make use of direct trade-off assessments between pairs of alternatives.

In this paper, we focus on a pairwise comparison method which has the added advantages of providing an organized structure for group discussions, and helping the decision making group focus on areas of agreement and disagreement when setting criterion weights.

The technique of pairwise comparisons has been developed by Saaty [34] in the context of a decision making process known as the Analytical Hierarchy Process (AHP). This technique was developed outside the GIS software using a variety of analytical resources and its first use with a GIS application was in 1991 [31]. In Saaty's technique, weights of this nature can be derived by taking the principal eigenvector of a square reciprocal matrix of pair-wise comparisons between the criteria. The comparisons deal with the relative importance of the two criteria involved in determining suitability for the stated objective. Ratings are provided on a nine-point continuous scale (Table 1).

Table 1: Scale for pairwise comparison [34].

Intensity of Importance	Definition
1	Equal importance
2	Equal to moderate importance
3	Moderate importance
4	Moderate to strong importance
5	Strong importance
6	Strong to very strong importance
7	Very strong importance
8	Very to extremely strong importance
9	Extreme importance

In developing weights, an individual or group compares every possible pairing and enters the ratings into a pairwise comparison matrix or *ratio matrix*. Since the matrix is symmetrical, only the lower triangle actually needs to be filled in. The remaining cells are then simply the reciprocals of the lower triangle. Eastmann [9] noted that if empirical evidence about the relative efficacy of a pair of factors exists, this evidence can also be used.

The procedure then requires that the principal eigenvector of the pairwise comparison matrix must be computed to produce the best fit set of weights. A good approximation to this result can be achieved by following the operations below [21]:

- sum the values in each column of the pairwise comparison matrix;
- divide each element in the matrix by its column total (the resulting matrix is referred to as the *normalized pairwise comparison matrix*); and
- compute the average of the elements in each row of the normalized matrix, that is, divide the sum of normalized scores for each row by the number of criteria.

These averages provide an estimate of the relative weights of the relevant criteria. Here, the weights are interpreted as the average of all possible ways of comparing the criteria.

Since the complete ratio matrix contains multiple paths by which the relative importance of criteria can be assessed, it is also possible to determine the degree of consistency that has been used in developing the ratings. Saaty [34] describes a procedure by which an index of consistency, and a *consistency ratio* (CR), can be produced. The consistency ratio (CR) defines the probability that the matrix ratings were randomly generated and Saaty suggests that matrices with CR ratings greater than 0.10 should be re-evaluated. In addition to the overall consistency ratio, it is also possible to analyze the matrix to determine where the inconsistencies arise.

Estimation of the consistency ratio involves the following operations:

- determination of the weighted sum vector by multiplying the weight for the first criterion times the first column of the original pairwise comparison matrix, then multiplying the second weight times the second column, the third criterion times the third column of the original pairwise matrix, and so on to the last weight, and finally summing these values over the rows; and
- determination of the consistency vector by dividing the weighted sum vector by the criterion weights determined previously.

The consistency ratio is defined as:

$$CR = \frac{CI}{RI} \tag{4}$$

where RI is the *random index*, and CI is the *consistency index* which provides a measure of departure from consistency.

The consistency index is calculated as:

$$CI = \frac{\lambda - n}{n - 1} \tag{5}$$

where λ is the average value of the consistency vector, and n is the number of criteria.

The random index is the consistency index of the randomly generated pairwise comparison matrix. and depends on the number of elements being compared. Table 2 shows random inconsistency indices (RI) for different numbers of criteria.

Table 2: Random inconsistency indices (RI) for different number of criteria [34].

n	RI	n	RI	n	RI
1	0.00	6	1.24	11	1.51
2	0.00	7	1.32	12	1.54
3	0.58	8	1.41	13	1.56
4	0.90	9	1.45	14	1.57
5	1.12	10	1.49	15	1.59

3.1.3 Evolution using the WLC decision rule

The procedure by which criteria are selected and combined to produce a particular evaluation, and by which evaluations are compared and acted upon, is known as a *decision rule*. A decision rule might be as

simple as a threshold applied to a single criterion or it may be as complex as one involving the comparison of several multi-criteria evaluations. Decision rules typically contain *choice function* for combining criteria into a single composite index and a *choice heuristics*, which is a statement of how alternatives are to be compared. Choice functions and heuristics provide a mathematical means of comparing alternatives. Since they involve some form of optimization such as maximizing or minimizing some measurable characteristic, they theoretically require that each alternative must be evaluated in turn. *Choice heuristics* specify a procedure to be followed rather than a function to be evaluated and are commonly used because they are often simpler to understand and also easier to implement [9].

Once the criteria maps (factors and constraints) are developed, an evaluation (or aggregation) stage is undertaken to combine the information from the various factors and constraints. The simplest type of aggregation is the *Boolean intersection* or logical AND. This method is used only when factor maps have been strictly classified into Boolean suitable/unsuitable images with values 1 and 0. The evaluation is simply the multiplication of all the images.

The weighted linear combination (WLC) aggregation method multiplies each standardized factor map (i.e., each raster cell within each map) by its factor weight and then sums the results. Since the sum of the set of factor weights for an evaluation must be one, the resulting suitability map will have the same range of values as the standardized factor maps that were used. This result is then multiplied by each of the constraints in turn to "mask out" unsuitable areas.

3.1.4 Limitations of WLC

There are some fundamental limitations, discussed by Jiand and Eastman (see [17]) in the use of weighted linear combinatorial procedures in a decision making process.

The first problem in using WLC as a decision rule concerns the different aggregation methods employed in decision making. Despite an expectation that the WLC method and Boolean method should yield similar results, they very often fail to do so because they cause logically methods of aggregation. In the WLC method, a low score on one criterion can be compensated by a high score on another; this is known as *trade-off* or *substitutability* and is quite different from the Boolean options, which are absolute in nature.

The second problem of the WLC stems from its *standardization of factors.* The most common approach to this is to rescale the range to a common numerical basis by simple linear transformation. However, the rationale for doing so is unclear [10, 36] and in some cases, a non-linear scaling may seem appropriate.

The third problem concerns *decision risk* which may be considered to be the likelihood that the decision will be wrong. For a Boolean procedure, decision risk can be estimated by propagating measurement error through the decision rule, thereby determining the risk that the decision made for a given location is wrong. Continuous criteria of weighted linear combination would appear however to express a further uncertainty that is not so readily estimated with stochastic methods. The standardized factors of WLC each express suitability: the higher the score, the more suitable the location is for the intended land use. There is no real threshold, however, that allows definitive allocation of areas to be chosen and areas to be excluded. Jiang and Eastmann [17] suggested that those kinds of problems could be solved by considering decision-making as a set problem and through the application of *fuzzy measures* in multi-criteria evaluation. They suggested that the ordered weighted averaging approach may provide an extension to and generalization of the conventional map combination methods in GIS.

3.2 Ordered weighted averaging (OWA)

Ordered Weighted Averaging (OWA) uses a class of multi-criteria operators [38] and involves two sets of weights: criterion, or importance weights and order weights [2]. A criterion weight is assigned to a given criterion or attribute for all locations in a study area to indicate its relative importance, according to the decision-maker's preferences, in the set of criteria under consideration. The order weights are associated with the criterion values on a location-by-location basis. They are assigned to a location's attribute values in decreasing order with no consideration of the attribute source of each value. The re-ordering procedure involves associating an order weight with a particular ordered position of the weighted attribute values. The first order weight is assigned to the highest weighted attribute values for each location, the second order weight to the second highest values, and so on.

Order weights are central to the OWA combination procedures. They are associated with the degree of ORness, which indicates the degree to which an OWA operator is similar to the logical connective OR in terms of its combinatorial behaviour. Order weight is also associated with a trade-off measure indicating the degree of compensation between criteria. The parameters associated with the OWA operations serve as a mechanism for guiding the GIS-based land-use suitability analysis. The ORness measure allows for interpreting the results of OWA in the context of the behavioural theory of decision making. OWA operations for example facilitate the development of a variety of land use strategies ranging from an extremity pessimistic (the minimum-type strategy based of the logical AND combination) through all intermediate neutral-towards-risk strategise (corresponding to the conventional WLC) to an extremely optimistic strategy, the maximum-type strategy based on the logical OR combination.

Thus, OWA can be considered as an extension and a generalization of the conventional combination procedures in GIS [17]. Indeed, WLC is just one variant of the OWA technique [9].

3.2.1 Order weights, trade-off and risk using OWA

In Weighted Linear Combination, factor weights are weights that apply to specific factors; all the pixels of a particular factor image receive the same factor weight in the raster data model. They indicate the relative degree of importance of factor in determining the suitability for an objective. In the case of WLC, the weight given to each factor also determines how it trades-off relative to other factors but, as described below, order weights in OWA determine the overall level of trade-off allowed. The use of order weights allows for aggregation solutions that fall anywhere along the risk continuum between AND and OR.

Order weights are quite different from factor weights because they do not apply to any specific factor. Rather, they are applied on a pixel-by-pixel basis to factor scores as determined by their rank ordering across factors at each location, or pixel. Order weight 1 is assigned to the lowest-ranked factor for that pixel (i.e., the factor with the lowest score), order weight 2 to the next higher-ranked factor for that pixel, and so forth. It is possible that a single order weight could be applied to pixels from any of the various factors depending upon their relative rank order.

Boolean approaches are extreme functions that result either in very risk-averse solutions when the AND operator is used or in risk-taking solutions when the OR operator is used. The WLC approach is an averaging technique that softens the hard decisions of the Boolean approach, avoiding the extremes. In a continuum of risk, WLC falls exactly in the middle; it is neither risk-averse nor risk-taking. But, any assignment of order weights results in a decision rule that falls somewhere in a triangular decision strategy space that is defined by the dimensions of risk and trade-off as shown in Figure 3.

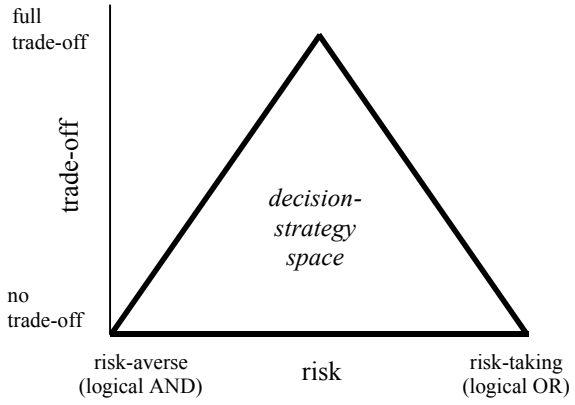


Figure 3: Triangular decision-strategy space defined by the dimension of risk and trade-off.

Table 3 shows, how order weights alter MCE results by controlling levels of trade-off and risk (see also [9]). Consider the case where factor weights are equal for three factors A, B, and C. Holding factor weights equal will make the effect of the order weights clearer. If a single pixel has factor scores A (210), B (197), and C (224), the factor weight for each of the factors will be 0.33. Ranking from minimum to maximum value, the order of these factors for this pixel is [B, A, C]. For this pixel, factor B will be assigned order weight 1, A order weight 2 and C order weight 3. In Table 3, there are thirteen sets of order weights that have been applied to this set of factor scores [197, 210, 224]. Each set yields a different MCE result even though the factor scores and the factor weights are the same in each case.

Table 3: Example of applied order weights to the set of factor scores (197, 210, 224).

Order weights			**Result**
min (1)	(2)	max (3)	
1.00	0.00	0.00	197
0.90	0.10	0.00	198
0.80	0.20	0.00	200
0.70	0.20	0.10	202
0.50	0.30	0.20	206
0.40	0.30	0.30	209
0.33	0.33	0.33	210
0.30	0.30	0.40	212
0.20	0.30	0.50	214
0.10	0.20	0.70	219
0.00	0.20	0.80	221
0.00	0.10	0.90	223
0.00	0.00	1.00	224

The first set of order weights in Table 3 is [1, 0, 0]. The weight of factor B (the factor with the minimum value in the set [B, A, C]) will receive all possible weight while factors A and C will be given zero weight. Such a set of order weights makes the factor weights irrelevant. Indeed, the order weights have altered the evaluation such that no trade-off is possible. As it can be seen in the Table 3, this has the effect of applying a minimum operator to the factors, thus producing the traditional intersection operator (AND) of fuzzy sets. Similarly, the last set of order weights [0, 0, 1] has the effect of a maximum operator, the traditional union operator (OR) of fuzzy sets. Again, there is no trade-off and the factor weights are not employed. Where the order weights are equal [0.33, 0.33, 0.33], all ranked positions are assigned the same weight; this makes trade-off fully possible and locates the analysis exactly midway between AND and OR. Equal order weights produce the same result as WLC.

In each of these three cases, the order weights have determined not only the level of trade-off but have situated the analysis on a continuum from (risk-averse, minimum, AND) to (risk-taking, maximum, OR). The order weights are not restricted to these three options, but instead any combination of values that sum to 1.0 can be assigned. As already noticed any assignment of order weights results in a decision rule that falls somewhere in a triangular decision strategy space (see Figure 3).

The degree of trade-off in OWA is governed by the relative distribution of order weights between the ranked factors. If the sum of the order weights is evenly spread

between the factors, there is strong trade-off, whereas if all the weight is assigned to a single factor rank, there is no trade-off. Order weights of [0.5, 0.3, 0.2] would indicate a strong (but not perfect) degree of risk aversion and some degree of trade-off. Weights of [0, 1, 0], would imply neither risk aversion nor acceptance, and no trade-off because all the weight is assigned to a single rank [9].

3.2.2 Evolution using OWA decision rule

There have already been several implementations in the last decade of OWA in GIS environments. As an example, OWA is already included for more than a decade in Idrisi GIS software [7]. Eastmann suggested the following guidelines for the use of the OWA option of MCE: (a) their criteria should be divided into three groups: hard constraints, factors that should, or should not trade-off. For example, factors with monetary implications typically trade-off, while those factors associated with some safety (or environment) concern typically do not; (b) if factors both trade-off and do not trade-off, their consideration should be separated into two stages of analysis. In the first stage, aggregate the factors that trade-off using the OWA option. The degree of trade-off can be controlled by manipulation of the order weights. In the second, use the result of the first stage as a new factor that is included in the analysis of those that do not trade-off; (c) if you run an analysis with absolutely no trade-off, the factor weights have no real meaning and can be set to any value.

Boroushaki and Malczewski have implemented an OWA-approach in the ArcGIS environment, pointing out that OWA combination operators can be recognized as the conventional AHP combination with modified criterion weights [2]. The weights are obtained by multiplying the criterion weights by order weights. With different sets of order weights, one can generate a wide range of OWA operators including the three aforementioned $S = \sum w_i x_i$ special cases of the WLC, Boolean overlay combination AND and OR.

4 Application of WLC and OWA

To demonstrate the WLC and OWA techniques for development of factor weights, let us consider an actual suitability problem. The objective is to find the suitable areas for residential development in the small municipality of Ig, which is a semi-rural community located near the Slovenian capital Ljubljana.

The whole procedure of decision rule (the procedure by which criteria are selected and combined to arrive at a particular evaluation, and by which evaluations are compared and acted upon) will not be presented here. The evaluation of criterion weights, trade-off and risk using OWA as well as WLC techniques of real problem is discussed. An example has been implemented in Idrisi Andes GIS [9], using MCE and OWA modules.

4.1 Application of WLC

A group of professionals who had developed a professional basis for the spatial plan of municipality of Ig, identified seven factors as the most important in searching for suitable areas for residential development in the municipality; those were: (1) distance from an existing residential zones, (2) slope, (3) solar illumination radiation, (4) distance from state and municipal roads, (5) distance from bus stops, (6) distance from flowing water, and (7) distance from forest. Table 4 shows pairwise comparison matrix (or ratio matrix) for these seven factors.

One of the advantages of the WLC method is the ability to give different relative weights to each of the factors by aggregation. Factor weights, sometimes called trade-off weights, are assigned to each factor. They indicate the importance of a factor relative to all other factors and they control how factors will trade-off or compensate for each other. In the case of WLC, where factors fully trade-off, factors with high suitability can be compensated for other factors with low suitability in a given location. The degree to which one factor can compensate for another is determined by its factor or trade-off weight.

Table 4: Ratio (or pairwise) matrix for seven factors.

	(1) resid. zones	(2) slope	(3) solar rad.	(4) roads	(5) bus stops	(6) flow. water	(7) forest
(1) resid. zones	1						
(2) slope	3	1					
(3) solar rad.	1	1/3	1				
(4) roads	3	1/3	4	1			
(5) bus stops	2	1/4	1/3	1/4	1		
(6) flow. water	1/3	1/6	1/3	1/6	1	1	
(7) forest	3	1/6	1/3	1/6	1/3	1	1

Table 5: Factor weights using the WLC method (A - factor weights derived by the approximation method; B - factor weights resulting from eigenvector using module WEIGHT, Idrisi Andes; CR – consistency ratio).

Factor	Factor weight - A	Factor weight - B
(1) distance from existing residential zones	0.0806	0.0839
(2) slope	0.3375	0.3512
(3) solar illumination radiation	0.1228	0.1204
(4) distance from state and municipal roads	0.2596	0.2653
(5) distance from bus stops	0.0851	0.0953
(6) distance from flowing water	0.0463	0.0457
(7) distance from forest	0.0680	0.0382
		CR = 0.06

After entry of the ratio matrix, factor weights were calculated in two ways: (A) using the approximation method described in 3.1.2, or (B) using the module WEIGHT in Idrisi Andes GIS, which calculates the eigenvector directly. WEIGHT utilizes a pairwise comparison technique to develop a set of factor weights that will sum to 1.0. Factors are compared two at a time in terms of their importance relative to the stated objective (locating residential development). When all possible combinations of two factors have been generated, the module calculates a set of weights and, importantly, a consistency ratio. This ratio indicates any inconsistencies that may have been arisen during the pairwise comparison process. The module allows repeated adjustments to the pairwise comparisons and reports the new weights and consistency ratio for each iteration. Table 5 shows both factor weights as well as consistency ratio calculated in Idrisi Andes GIS.

One of the most common procedures for aggregating data by WLC method is to multiply each standardized factor by its corresponding weight. These data are then summed and the sum is divided by the number of factors [9]. Once this weighted average is calculated for each pixel, the resultant image is multiplied by the relevant Boolean constraints to mask out areas that should not be considered at all. The final image is a measure of aggregate suitability that ranges from 0 to 255 for non-constrained locations (Figure 5a). The WLC aggregation method allows standardization of the criteria in a continuous fashion, retaining important information about degrees of suitability. It also allows differentially weighted criteria to trade-off with each other. In the next application, another aggregation technique, ordered weighted averaging is explored. This allows control of the amount of risk and trade-off to be included in the result.

4.2 Application of OWA

The aggregation method of ordered weighted averaging (see 3.2) offers control over the position of the MCE along the risk and trade-off continuum. Using OWA, we can control the level of risk we wish to assume in our MCE, and the degree to which factor (trade-off) weights will influence the final suitability map [9]. OWA offers a wealth of possible solutions for our residential development problem. In our application, seven order weights were applied corresponding to the seven factors that were rank-ordered for each location after the modified factor weights were applied. Table 6, gives six typical sets of order weights for the seven factors: (a) average level of risk and full trade-off, (b) low level of risk and no trade-off, (c) high level of risk and no trade-off, (d) low level of risk and average trade-off, (e) high level of risk and average trade-off, (f) average level of risk and no trade-off. Figure 4 shows the locations of typical sets of order weights in the decision-support space.

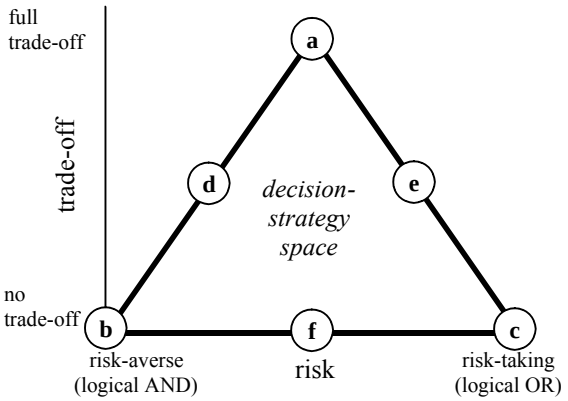


Figure 4: Decision-strategy space and typical sets of order weights (see Table 6).

It is evident that the set of order weights are in accordance with factor weights derived by WLC. The weight is distributed evenly among all factors regardless of their rank-order position from minimum to maximum for any given location. They are not skewed toward the minimum (AND operation) or the maximum (OR operation). As in the WLC procedure, the result of order weights (a) is exactly in the middle in terms of risk. In addition, because all rank order positions are given the same weight, no rank-order position will have a greater influence over another in the final result. Set (a) gives full trade-off between factors, allowing the factor weights to be fully employed.

Table 6: Typical sets of order weights for seven factors.

	(**a**) Average level of risk and full trade-off						
order weight	0.1428	0.1428	0.1428	0.1428	0.1428	0.1428	0.1428
rank	1st	2nd	3rd	4th	5th	6th	7th
	(**b**) Low level of risk and no trade-off						
order weight	1	0	0	0	0	0	0
rank	1st	2nd	3rd	4th	5th	6th	7th
	(**c**) High level of risk and no trade-off						
order weight	0	0	0	0	0	0	1
rank	1st	2nd	3rd	4th	5th	6th	7th
	(**d**) Low level of risk and average trade-off						
order weight	0.4455	0.2772	0.1579	0.0789	0.0320	0.0085	0
rank	1st	2nd	3rd	4th	5th	6th	7th
	(**e**) High level of risk and average trade-off						
order weight	0	0.0085	0.032	0.0789	0.1579	0.2772	0.4455
rank	1st	2nd	3rd	4th	5th	6th	7th
	(**f**) Average level of risk and no trade-off						
order weight	0	0	0	1	0	0	0
rank	1st	2nd	3rd	4th	5th	6th	7th

To produce a low risk result for the residential development problem, the one close to AND (minimum) on the risk continuum, then greater order weight is given to the lower rank-orders (the minimum suitability values) – set (b) in Table 6. Such a weighting results in no trade-off. If a high risk result (OR (maximum)), is sought, then greater order weight has to be given to the higher rank-orders (the maximum suitability values) – set (c).

Using the OWA approach, the order weights can be altered in terms of their skew and dispersion. It is possible to produce an almost infinite range of possible solutions to the problem. In our residential development problem, the decision-makers and administrators may be interested in a conservative or low-risk solution to the identification of suitable areas for development. They also know that their estimates for how different factors should trade-off with each other are important and worthy of consideration. An AND operation will not let them consider any trade-off, and the WLC operation, where they would have full trade-off, is too liberal in terms of risk. They will then want to develop a set of order weights that would give them some amount of trade-off but would maintain a low level of risk in the solution.

There are several sets of order weights that could be used to achieve this. Let us consider the set of order weights (d). These specify an operation midway between the extreme of AND and the average risk position of WLC. In addition, they set the level of trade-off to be intermediate between the no trade-off situation of the AND operation and the full trade-off situation of WLC.

While it is clear that suitability generally increases from AND to OR for any given location, the character of the increase between any two operations is different for each location. The extremes of AND and OR are clearly dictated by the minimum and maximum factor values, however, the results from the middle trade-off operations are determined by an averaging of factors that depends upon the combination of factor values, factor weights, and order weights [9]. In general, in locations where the heavily weighted factors (slopes and roads) have similar suitability scores, the three results with trade-off will be strikingly similar. At the locations where these factors do not have similar suitability scores, the three results with trade-off will be more influenced by the difference in suitability (toward the minimum, the average, or the maximum).

4.2.1 Grouping factors by trade-off

Eastmann [9] suggested that the OWA approach could also be used to aggregate the suitability maps of groups of factors. Our factors are of two distinct types: factors relevant to *development cost* and factors relevant to *environmental concerns*, which do not necessarily have the same level of trade-off. Factors relevant to the cost of development clearly can fully trade-off. Where financial cost is the common concern, savings in development cost in one factor can compensate for a high cost in another. Environmentally relevant factors on the other hand, do not easily trade-off. To cope with this discrepancy, we treated our factors as two distinct sets with different levels of trade-off specified by two sets of ordered weights. This yields two intermediate suitability maps, one the result of combining five financial factors, and the other the result of combining both environmental factors. We then combined these intermediate results using a third MCE operation.

We decided that factors relevant to cost (1-5) could fully trade-off and selected an average risk; for this reason, the WLC procedure to combine them has been used. For the second group of factors, those relevant to environmental concerns, we decided to use the same procedure as for those relevant to costs – however, environmental factors were treated separately. Table 7 shows the revalued factor weights and the old factor weights, when all factors together (1-7) have been calculated (see 4.1).

Table 7: Re-valued factor weights using WLC (relevant to development cost) and OWA (relevant to environmental concerns) (B - factor weights as the result of eigenvector using module WEIGHT, Idrisi Andes GIS before grouping).

Factors	Factor weight – B (from Table 4)	Revalued factor weight
Cost factors		
(1) distance from existing residential zones	0.0839	0.1040
(2) slope	0.3512	0.3834
(3) solar illumination radiation	0.1204	0.1314
(4) distance from state and municipal roads	0.2653	0.2896
(5) distance from bus stops	0.0953	0.0916
Environmental factors		
(6) distance from flowing water	0.0457	0.5447
(7) distance from forest	0.0382	0.4553

The final step in defining the suitable areas for residential development was to combine two intermediate results using a third MCE operation. In that aggregation, factors relevant to costs and factors relevant to environment were treated as factors in a separate aggregation procedure. There is no clear rule how to combine these two results [9] and so we assumed that decision-makers would be unwilling to give more weight to either the developers' or the environmentalists' factors, these factor weights being equal, and they would not allow the two new consolidated factors to trade-off with each other, nor did they want anything but the lowest level of risk when combining the two intermediate results. For these reasons, we used an OWA procedure that yielded a low risk result with no trade-off (the order weights were 1 for the 1st rank and 0 for the 2nd).

Figure 5 shows both results of MCE: (a) using WLC approach, and (b) using OWA approach. Constraints including built-up zones, electric mains and water bodies have been applied to the final suitability maps to mask out unsuitable areas. The darker colours denote more suitable areas for residential development in the municipality of Ig.

4.3 Discussion

In this paper, two methods for MCE in GIS, weighted linear combination (WLC) and ordered weighted averaging (OWA), were presented and tested. Both techniques are used most effectively with factors that have been standardized to a continuous scale of suitability and weighted according to their relative importance. The relative importance weights of factors were estimated using the analytical hierarchy process (AHP). Constraints were defined as Boolean masks. The methods have some similar procedures but, in this paper, they were based on different statements about how criteria dealing with land use suitability analysis could be evaluated. Consequently, they yielded to two different results (Figure 5).

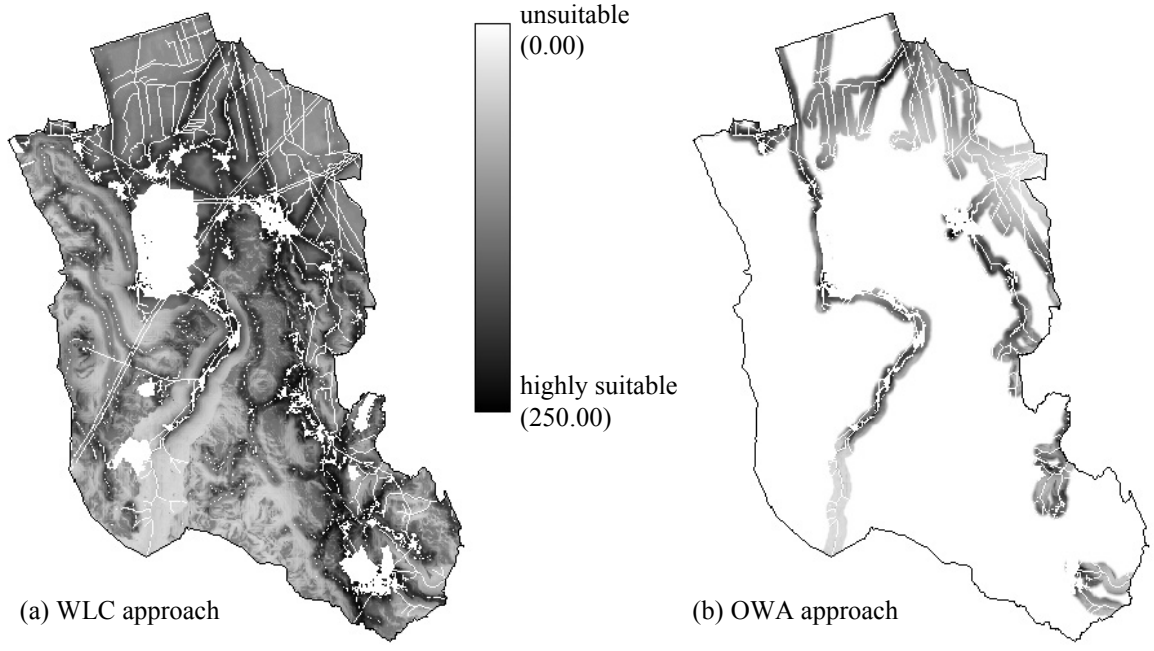


Figure 5: Results of aggregation of weighted factors and constraints using WLC and OWA approaches (suitability for residential development).

The land evaluation was performed on a cell by cell basis. WLC allowed us to use the full potential of our factors as continuous surfaces of suitability. The identified factors were standardized using fuzzy functions, and then weighted and combined using an averaging technique. The factor weights used expressed the relative importance of each criterion to the overall objective, and they determined how factors were able to trade off with each other. The final map of continuous suitability for residential development (image (a) on Figure 5) is a result that is neither extremely risk-averse nor extremely risk-taking. In WLC, all factors were allowed to fully trade-off. Any factor could compensate for any other in proportion to its factor weight.

The second aggregation method we used OWA, gave us control over the position of the MCE along both the risk and trade-off axes (see Figure 4). It allowed control of the level of risk we wished to assume in our application, and the degree to which factor weights (trade-off weights) influenced the final suitability map. Control over risk and trade-off was made possible through a set of order weights for the different rank-order positions of factors at every location (pixel). With order weights we combined costs and environmental factors with a very low level of risk and no trade-off between them. So, the order weights first modified the degree to which factor group weights had influence in the aggregation procedure, thus they governed the overall level of trade-off. After weights were applied to the factor groups (to some degree dependent upon the overall level of trade-off used), the results were ranked from low to high suitability for each location. This had the effect of weighting factor groups based on their rank from minimum to maximum value for each location. The relative skew toward either minimum or maximum of the order weights controlled the level of risk in the evaluation. Additionally, the degree to which the order weights were evenly distributed across all positions controlled the level of overall trade-off, i.e., the degree to which factor group weights had influence.

In our application of WLC we used factor weights defined in Table 5 and Table 7, and in OWA we applied the order weights (1 for the 1st rank and 0 for the 2nd) that yielded a low risk result with no trade-off. Defined and fully traded-off weights in the WLC approach led to a result in which a larger proportion of the municipality area was indicated as highly suitable for residential development, as compared to the result of OWA approach (see Figure 5). However, with the continuous result of WLC, the best locations for residential development can be defined by setting the lowest degree of suitability; e.g. 200 in Figure 5 (a).

The standardization and aggregation techniques discussed here are important in exploration of any multi-criteria problem and they result in images that show the suitability of locations in the entire study area. Multi-criteria problems often concern eventual site selection for some development, land allocation, or land-use change and there are many techniques for site selection using images of suitability. However, the main purpose of this application was not to suggest the best areas for residential development in tested area. It was rather to define and apply two methods of MCE (WLC and OWA) in GIS as the generalised framework of GIS-based spatial decision-making procedure, and to show the policy makers and decision makers what kind of tools useful for calculations of images of suitability. When our results are compared with the professional basis for spatial plan in Ig, made by classical approach as defined in Figure 2, the most similar result was from the OWA approach with low risk and no trade-off between costs and environmental factors.

The results of our research indicate that applications of decision making in GIS are multifunctional and can incorporate different levels of complexity of the decision problem. In this case, the choice of weights and weighting techniques played a crucial role. It is obvious that decision makers with a preference for a subjective scale may not arrive at the same weights for the factor criteria. This may lead to different results for suitability maps and can affect the final decision with regard to the overall objective. However, it must be noted that the presented methods are only tools to aid decision makers; they are not the decision itself.

5 Conclusions

Spatial multi-criteria analysis, with its explicit geographic component represents a significant departure from the conventional MCDM techniques. In contrast to the conventional multi-criteria decision making, spatial multi-criteria analysis requires both data on criterion values and the geographical location of alternatives. Increasing computer power, user-friendly GIS and decision support software, and increased access to and familiarity with computers among decision makers are a few of the reasons for the rapid growth in both research and practice in GIS-based multi-criteria spatial decision making. GIS technology provides the capabilities of data acquisition, storage, retrieval, manipulation, and data analysis to develop information that can support decisions. MCDM techniques provide the tools with which to integrate the geographical data and the decision maker's preferences into one-dimensional value array of alternative decisions. The use of GIS in SDSS in addition provides spatial data models, the means of entering and displaying spatial data and additional spatial analysis tools. A significant contribution of the SDSS concept to geographic information science is that it integrates distinct tool sets (data and models) into a unified whole more valuable than the sum of the parts.

The results presented in this paper demonstrate the application of weighted linear combination (WLC) and ordered weighted averaging (OWA) within a GIS for the purpose of determining the most suitable locations for residential areas in the municipality of Ig (see also [6]). It is clear that integrated decision support tools in the GIS software system allow exploration of variety of rationales and perspectives in suitability evaluation and land allocation. The test study of suitability analysis for a residential area is a simple case with only seven main attributes. In the real world, the situation is much more

complex. There are still several topics referring to the spatial multi-criteria decision analysis in GIS that must be investigated and developed. These include selection of attributes, which must take account of their completeness, independence and real influence, or weight; the scale and methods of aggregation of attributes; error assessment and finally, the incorporation of database and decision rule uncertainty and sensitivity analysis. The tools currently available however, offer significant advantages for decision makers in spatial decision problem fields.

References

[1] Banai R. (1993). Fuzziness in geographic information systems: contributions from the analytic hierarchy process. International Journal of Geographical Information Systems, 7(4), pp. 315–329.

[2] Boroushaki S. and J. Malczewski (2008). Implementing an extension of the analytical hierarchy process using ordered weighted averaging operators with fuzzy quantifiers in ArcGIS, Computers & Geosciences, (34), pp. 399–410.

[3] Carver S. J. (1991). Integrated multi-criteria evaluation with geographical information systems, International Journal Geographical Information Systems, 5(3), pp. 321–339.

[4] Chakhar S. and V. Mousseau (2008). Spatial multicriteria decision making. In: Shehkar S. and H. Xiong (Eds.), Encyclopedia of GIS, Springer-Verlag, New York, pp. 747–753.

[5] Diaz III J. and J. A. Hansz (2002). Behavioral Research into the Real Estate Valuation Process: Progress Toward a Descriptive Models. In: Wang K., Wolverton M., L. (Eds), Real Estate Valuaton Theory. Kluwer Academic Publishers, Boston, pp. 3–29.

[6] Drobne S., A. Lisec and M. Cemič (2008). Večkriterialno odločanje pri izbiri lokacije dejavnosti v geografskem informacijskem sistemu = Multi-criterial decision-making of the activity alocation in geographical information system. In: Dnevi slovenske informatike 2008 - DSI, Interoperabilnost kot izziv informatiki, Zbornik prispevkov. Slovensko društvo Informatika, Ljubljana (on CD).

[7] Eastman J. R. (1997). Idrisi for Windows, Version 2.0: Tutorial Exercises, Graduate School of Geography – Clark University, Worcester, MA.

[8] Eastman J. R., H. Jiang and J. Toledano. (1998). Multi-criteria and multi-objective decision making for land allocation using GIS. In: Beinat E. And Nijkamp P. (Eds.), Multicriteria Analysis for Land-Use Management. Kluwer Academic Publishers, Dordrecht, pp. 227–251.

[9] Eastman J. R. (2006). Idrisi Andes – Tutorial, Clark Labs., Clark University, Worcester, MA.

[10] Eastman J. R., P. A. K. Kyem, J. Toledano and W. Jin (1993). GIS and decision making. Explorations in geographic information system technology, 4, UNITAR, Geneva.

[11] FAO (1976). A Framework for Land Evaluation, Soils Bulletin 32. Food and Agricultural Organization of the United Nations, Rome.

[12] Goodchild M. F. (1987). A spatial analytical perspective on geographical information systems. International Journal of Geographical Information Systems, 1(4), pp. 327–334.

[13] Grimshaw D. J. (1994). Bringing geographical information systems into business. Longman Scientific and Technical, Harlow, Essex.

[14] Hywood I., S. Cornelius and S. Carver (2006). An Introduction to Geographical Information Systems. Third Edition. Pearson Education Limited, Harlow, Essex.

[15] Jankowski P. (1995). Integrating geographical information systems and multiple criteria decision making methods. International Journal of Geographical Information Systems, 9(3), pp. 251–273.

[16] Janssen R. and P. Rietveld (1990). Multicriteria Analysis and Geographical Information Systems: An Application to Agricultural Land Use in the Netherlands. In: Scholten H. J. and J. C. H. Stillwell (Eds.), Geographical Information Systems for Urban and Regional Planning. Cluwer Academic Publishers, Dordrecht, pp. 129–139.

[17] Jiang H. and J. R. Eastman (2000). Application of fuzzy measures in multi-criteria evaluation in GIS. International Journal of Geographical Information Systems, 14(2), pp. 173–184.

[18] Joerin F., M. Theriault and A. Musy (2001). Using GIS and outranking multicriteria analysis for land-use suitability assessment. International Journal of Geographical Information Science, 15(2), pp. 153–174.

[19] Keeney R. L. (1992). Value-focused thinking: a path to creative decision making. MA: Harvard University Press, Cambridge.

[20] Laurini R. and D. Thompson (1996). Fundamentals of Spatial Information Systems, Academic Press Inc., London.

[21] Malczewski J. (1999). GIS and Multicriteria Decision Analysis, John Wiley and Sons, Toronto.

[22] Malczewski J. (2004). GIS-based land-use suitability analysis: a critical overview. Progress in Planning, 62(1), pp. 3–65.

[23] Malczewski J. (2006). GIS-based multicriteria decision analysis: a survey of the literature. International Journal of Geographical Information Science 20(7), pp. 703–726.

[24] Malczewski J. (2006). Integrating multicriteria analysis and geographic information systems: the ordered weighted averaging (OWA) approach. Int. J. Environmental Technology and Management, 6(1/2), pp. 7–19.

[25] Malczewski J. and C. Rinner, 2005. Exploring multicriteria decision strategies in GIS with linguistic quantifiers: a case study of residential

quality evaluation. Journal of Geographical Systems, 7 (2), pp. 249–268.
[26] Nijkamp P. (1979). Multidimensional spatial data and decision analysis, Wiley, Chichester, West Sussex.
[27] Nijkamp P. and A. van Delft (1977). Multi-criteria analysis and regional decision-making. Springer, Amsterdam.
[28] Nijkamp P. and P. Rietveld (1986). Multiple objective decision analysis in regional economics. In: P. Nijkamp (Ed.): Handbook of regional and urban economics. Elsevier, New York, pp. 493–541.
[29] Nijkamp P., P. Rietveld and H. Voogd (1990). Multicriteria evaluation in physical planning. North-Holland, Amsterdam.
[30] Pereira J. M. C. and L. Duckstein (1993). A multiple criteria decision-making approach to GIS-based land suitability evaluation. International Journal of Geographical Information Systems, 7(5), pp. 407–424.
[31] Rao M., S. V. C. Sastry, P. D. Yadar, K. Kharod, S. K. Pathan, P. S. Dhinwa, K. L. Majumdar, D. S. Kumar (1991). A Weighted Index Model for Urban Suitability Assessment – A GIS Approach. Bombay Metropolitan Regional Development Authority, Bombay.
[32] Rinner C. (2008). Mobile Maps and More – Extending Location-Based Services with Multi-Criteria Decision Analysis. In L. Meng, A. Zipf and S. Winter (Eds.), Map-Based Mobile Services. Lecture Notes in Geoinformation and Cartography. Springer, Berlin, pp. 335–349.
[33] Rosenthal R. E. (1985). Concepts, Theory and Techniques: Principals of Multiobjective Optimization. Decision Sciences, 16(2), pp. 133–152.
[34] Saaty T. L. (1977). A Scaling Method for Priorities in Hierarchical Structures. J. Math. Psychology, 15, pp. 234–281.
[35] Simon H. A. (1977). The New Science of Management Decision, Upper Saddle River, NJ: Prentice Hall, 3rd Edition.
[36] Voogd H. (1983). Multicriteria Evaluation for Urban and Regional Planning. Pion, Ltd., London.
[37] Worral L. (1991). Spatial Analysis and Spatial Policy using Geographic Information Systems, Belhaven Press, London.
[38] Yager R. R. (1988). On ordered weighted averaging aggregation operators in multi-criteria decision making. IEEE Transactions on Systems, Man and Cybernetics, 18 (1), pp. 183–190.

Eyeball Localization Based on Angular Integral Projection Function

Ghassan J. Mohammed, Bingrong Hong and Ann A. Jarjes
Computer Science and Engineering Department
Harbin Institute of Technology
Harbin, 150001, P.R. China
E-mail:ghassanjasim@yahoo.com, hongbr@hit.edu.cn, ann_kazzaz2004@yahoo.com

Ghassan J. Mohammed
Foreigner Students Building A-13
Harbin Institute of Technology
92 West Dazhi Street
Nangang District
Harbin , P. R. China
E-mail:ghassanjasim@yahoo.com

Keywords: eyeball localization, iris boundaries localization, human facial features, biometrics, integral projection functions

Received: June 26, 2008

Iris boundaries localization is a critical model in facial feature-based recognition systems. It has a close relationship to the results' accuracy of these applications. In this paper, we propose a new algorithm based on the angular integral projection function (AIPF) to localize eyeball (iris outer boundary) in iris images. The proposed algorithm adopts boundary points detection and curve fitting within gray level images. Mainly it finds the iris outer boundary's features, the center and radius in three steps. First, we detect the approximate position of iris center. Then, using AIPF, a set of radial boundary points are detected. Finally, the features are obtained by fitting a circle to the detected boundary points. The experimental results on 756 iris images from CASIA show high accuracy and efficiency.

Povzetek: Razvit je nov algoritem biometričnega prepoznavanja na osnovi mrežnice.

1 Introduction

The localization of human facial features plays a very important role in biometric systems and has received a great deal of interest in recent years. Such biometric systems include iris recognition, face recognition, disease diagnosis and human-computer interaction. Iris feature is considered to be the most important feature among other facial features that include eyes, nose, mouth and eyebrows. Iris localization aims to find the parameters, the centers and radii, of the inner and outer boundaries of iris. However, the localization of eyeball or the iris outer boundary is more difficult since it is not sharp and clear as the inner boundary, and due to the occlusion caused by eyelashes and eyelids. As localization accuracy has great influence on subsequent feature extraction and classification, many different methods have been presented for the purpose of iris localization. For the first time, Daugman [4] proposed an integrodifferential operator to detect both iris inner and outer boundaries. Wildes [15] and Masek [8] used binary edge map and voting procedure, which realized by Hough transform, to detect iris boundaries. Further, Ma et al. [7] presented an iris segmentation method based on Canny edge detection and Hough transform, Tisse et al. [14] used integrodifferential operator with Hough transform, and Cui et al. [3] used Haar wavelet transform followed by Hough transform for detecting the iris inner boundary and differential integral operator for localizing the outer boundary. More recently, other iris localization algorithms have been proposed [5] [6] [13] [18] [12]. However, most of the localization algorithms mentioned above are based on edge detection followed by Hough transform, that is to search the iris boundaries over three-parameter space exhaustively, which makes the process time-consuming ,thus they can not be employed in real time iris recognition systems. Moreover, they require threshold values to be chosen for edge detection and this may cause critical edge points being removed, resulting in failure to detect circles. In this paper, we address these two problems and propose a new algorithm to localize eyeball in iris images efficiently and accurately. As eyeball is almost circle, we localize it by extracting the features, the center and radius, of the iris outer boundary. Thus, through the paper we denote by iris features the center and radius of the iris outer boundary. The proposed algorithm adopts boundary points detection followed by curve fitting. It does not need to find all boundary points, so its localization speed is very fast.

In our earlier work [9], the AIPF has been developed as a general function to perform integral projection along angular directions, both the well known

vertical integral projection function (IPF_v) and horizontal integral projection function (IPF_h) can be viewed as special cases of AIPF. In our approach, AIPF is adopted to detect boundary points. First, the approximate position of the iris center is detected by calculating the center of mass for the binarized eye image. Then, a set of iris outer boundary's radial points are obtained using AIPF. Finally, we get the precise iris features through fitting a circle to the above boundary points by making use of the least squares method. Being evaluated on 756 eye images from CASIA [2] and quantitatively analysed based on own ground truth, the experimental results show high accuracy and efficiency. The rest of this paper is organized as follows. In Section 2, integral projection functions are described. In Section 3, the algorithm of iris features extraction is detailed. Section 4 provides the experimental results of the algorithm on CASIA iris database. Section 5 concludes the paper.

2 Projection functions

2.1 Integral projection functions

Due to their simplicity and robustly, image integral projection functions have been used widely for the detection of the boundary between different image regions. Among them, the vertical and horizontal integral projection functions are most popular. Here, suppose $I(x,y)$ is the intensity of a pixel at location (x,y), the vertical integral projection function $IPFv(x)$ and horizontal integral projection function $IPFh(y)$ of $I(x,y)$ in intervals [y1, y2] and [x1,x2] can be defined respectively as:

$$IPF_v(x) = \int_{y1}^{y2} I(x,y)dy. \tag{1}$$

$$IPF_h(v) = \int_{x1}^{x2} I(x,y)dx. \tag{2}$$

The above two functions are used to detect the boundary of different image regions in the vertical and horizontal directions. Assuming PF is a projection function and ξ is a small constant. Thus, if the value of PF rapidly changes form z_0 to $(z_0+\xi)$, it indicates that z_0 lie on the boundary between two homogeneous regions. In detail, given a threshold T, the vertical boundary in the image can be identified by:

$$\Theta_V = \max\left\{\left|\frac{\partial PF_V(x)}{\partial x}\right| > T\right\}. \tag{3}$$

where Θ_v is the set of vertical boundary points, such as $\{(x_1, PF_v(x_1)), (x_2, PF_v(x_2)),\ldots,(x_k, PF_v(x_k))\}$, which vertically divides the image into different areas. It is obvious that the horizontal boundary points can be identified similarly [17].

2.2 Angular integral projection function

Besides the sets of vertical and horizontal boundary points that can be detected using IPF_v and IPF_h respectively. Other boundary point sets can be identified on other directions rather than those are on the vertical and horizontal directions. Considering this fact and in order to capture the boundary point sets along all directions within an image, the AIPF has been proposed in our earlier work [9] as a general function to perform integral projection along angular directions it is defined as:

$$AIPF(\theta,\rho,h) = \frac{1}{h+1}\int_{j=-h/2}^{h/2} I((x_0+\rho\cos\theta)+(j\cos(\theta+\pi/2)), (y_0+\rho\sin\theta)+(j\sin(\theta+\pi/2)))dj \tag{4}$$

where (x_0,y_0) is the image center, $I(x,y)$ is the gray level of the pixel at location (x,y), θ is the angle of the integration rectangle with x-axis, $\rho = 0,1,\ldots,w$, w is the width of the integration rectangle, and h represents the height of the integration rectangle or the number of pixels to be integrated within each line. Specifically, the application of AIPF on θ direction carries out within an integration rectangle with $w\times h$ dimensions, it extends along a central line irradiated from the image center and having θ with x-axis. Here, it is worth to mention that even the most commonly used projection functions IPF_v and IPF_h can be implemented using AIPF by assigning θ=0°,180° and θ=90°,270° respectively.

3 Iris features extraction

In this section we are mainly concerned with the extraction of the iris features: iris center and iris radius in the segmented gray level eye images. Figure 1(a) shows an example of eye images used in this paper.

3.1 Approximate iris center detection

Since the centers of both iris and pupil are close to each other, we consider the pupil center as the approximate iris center. In order to determine the pupil center, the gray levels histogram is plotted and analysed. Figure 1(b) shows the histogram of gray levels for the image in Figure 1(a). Depending on the eye image histogram, a threshold value T is determined as the intensity value associated with the first important peak within histogram. Then, all intensity values in the eye image below or equal T are changed to 0(black) and above T are changed to 255(white),as:

$$g(x,y) = 255, \text{ if } I(x,y) \geq T$$
$$g(x,y) = 0, \text{ otherwise.} \tag{5}$$

where $I(x,y)$ is the intensity value at location (x,y), $g(x,y)$ is the converted pixel value and T represents threshold. This process converts a gray image to binary image and

efficiently segments the pupil from the rest of the image as shown in Figure 1(c). However, morphological processing is still necessary to remove pixels that located outside the pupil region. Figure 1(d) shows the clear pupil region obtained from Figure 1(c) after noise removing by using dilate operator. Now, the center of the segmented pupil can be easily determined. Basing on [1], the center of a simple object like circle and square coincides with its center of mass. The center of the mass refers to the balance point $(\bar{x}, \bar{y})$ of the object where there is equal mass in all directions:

$$\bar{x} = \frac{1}{\sum g(x,y) \in F} \sum_{g(x,y) \in F} x. \quad (6)$$

$$\bar{y} = \frac{1}{\sum g(x,y) \in F} \sum_{g(x,y) \in F} y. \quad (7)$$

where $g(x,y)$ is a pixel in the position (x,y), and F is the object under consideration. We make use of the equations (6) and (7) to find the pupil center $P(x_p,y_p)$ which approximates the iris center $I(x_i,y_i)$. The pupil center detection process is shown in Fig. 1

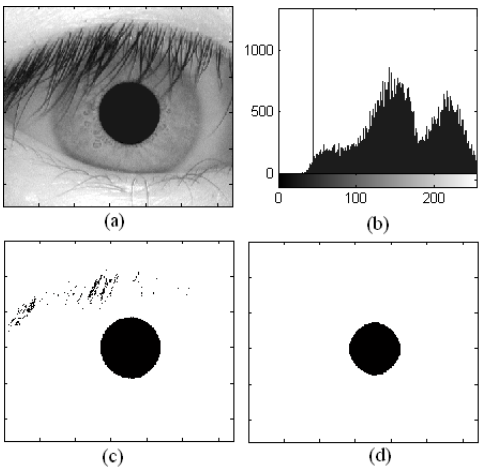


Figure 1: The pupil center detection. (a) Original image. (b)Gray levels histogram. (c) Binary image. (d) Binary image after morphological dilation operation.

3.2 Iris radius estimation

After the approximate iris center is detected, the precise iris center and radius can be estimated as the center and radius of a circle fitted to the iris outer boundary. Here, in order to reduce computational time, two rectangles are set on both sides of the iris basing on the estimated location of iris center and the number of integration rectangles that to be applied, as shown in Figure 3. In our approach, we divide the iris radius estimation task into three stages. In the first stage, as an image pre-processing step, image nonlinear filtering is performed. In the second stage, iris outer boundary points are detected. Note that both the two previous stages are performed within the above predefined rectangles. Finally, in the third stage, a circle is fitted to these points using the least squares method. The reminder of this section details each of them.

3.2.1 Image pre-processing

As a preliminary stage to the iris boundary points detection, image filtering is performed to minimize the influence of the occlusion caused by the eyelashes in both iris' left and right rectangles, which in turns help us to detect iris boundary accurately. In this work, a nonlinear filter is adopted basing on anisotropic diffusion (an image enhancement process that removes noise and irrelevant details while preserving the edges). The filter that we used is based on the formulation of Perona and Malik [10] for the anisotropic diffusion. Applying to an image, such a filter encourages intraregional diffusion while preserving contours as shown in Figure 2. Thus, it serves to a better edge detection in a potential noisy image.

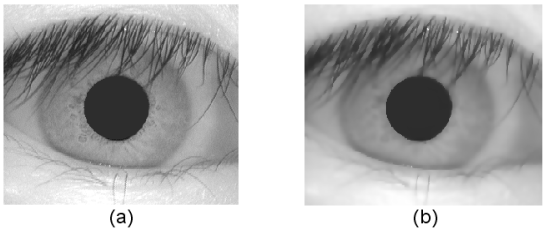


Figure 2: Anisotropic filtering based on Perona and Malik's formulation [10]. (a) Original image. (b) Filtered image.

3.2.2 Boundary points detection

From the iris image shown in Figure 1(a), it is clear that the iris is circular and darker than the surrounding area. Accordingly, considering the approximate iris center detected in the previous section as the image center, AIPF can be applied to find a set of radial boundary points. Here, in order to reduce the influence of potential occlusion caused by eyelids or eyelashes to minimum, θ is limited in the ranges -30°~ 5° and 175°~ 210° within the right iris rectangle and the left iris rectangle respectively. This is because that the iris region within these θ ranges are barely influenced by occlusion. Figure 3 shows four integration rectangles within each of the left and right iris rectangles.

Next, we find only a radial boundary point for each integration rectangle on θ direction. This is accomplished by computing the gradient of the projection curve resulted by each application of AIPF. Then we obtain the iris' boundary point by searching the gradient curve for the local maximum that corresponds to the iris edge. Clearly, the more integration rectangles thus iris' boundary points, the finer iris outer boundary localization. However, as a circle can be fit through three boundary points, at least three integration rectangles have to be established within both iris rectangles, here a

reasonable angular shift between successive integration rectangles have to be considered.

3.2.3 Curve fitting

As iris boundary is considered as a circular contour. Hence, we get the precise iris center $I(x_i,y_i)$ and radius R_i through fitting a circle to the collection of the above iris' boundary points. Figure 3 shows a circle fitted to iris boundary based on boundary point detected in previous section. Here, in order to obtain a best circle fit, we make use of the least squares method which minimizes the summed square of errors. The error for the i^{th} boundary point r_i is defined as the difference between the detected boundary point $p\_detcted_i$ and the fitted circle point p_fitted_i, as:

$$r_i = p\_detcted_i - p\_fitted_i \tag{8}$$

Thus the summed square of errors is given by:

$$S = \sum_{i=1}^{n} r_i^2 \tag{9}$$

where n is the number of the detected radial boundary points.

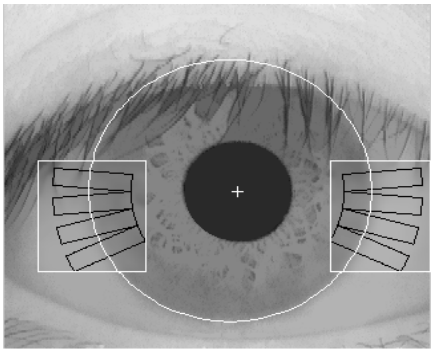


Figure 3: An example illustrates the application of the AIPF with: θ_1=5°, θ_2=-5°, θ_3=-15°, θ_4=-25°, and with θ_1=175°, θ_2=185°, θ_3=195°, θ_4=205°, in the right iris rectangle and the left integration rectangle respectively within an iris image. Each black rectangle represents an integration rectangle with $w \times h$ dimensions. The white rectangles are for the left and right iris rectangles. White cross denotes the center of the circle fitted to the iris outer boundary.

4 Experimental results

We perform experiments to evaluate the performance of the proposed algorithm over iris images supplied by CASIA [2]. The algorithm was applied for each image in the database. All experiments are performed in Matlab (version 6.5) on a PC with P4 3GHz processor and 512 M of DRAM. Figure 4 shows part of the experimental results.

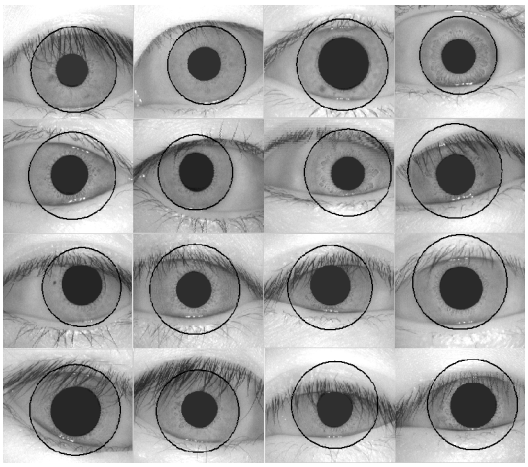


Figure 4: Eyeball localization examples. Row 1 and 2 show the localization results for typical irises and irises with different size and location. Row 3 and 4 show the localization results for irises occluded with eyelash and/or eyelid. Six integration rectangles are used with: θ_1=5°, θ_2=0°, θ_3=-5°, θ_4=-10°, θ_5=-15°,θ_6=-20°, in the right iris rectangle and other six are used with: θ_1=175°, θ_2=180°, θ_3=185°, θ_4=190°, θ_5=195°,θ_6=-200°, in the left integration rectangle. Thus 12 radial edge points are considered here. It was assumed that w=60 and h= 15.

4.1 Database characteristics

The CASIA iris database was adopted for testing. Here, it must be mentioned that this database has been manually edited [11] but it does not effect the application of AIPF to detect iris' edge points since editing was limited to the pupil region. CASIA V1.0 includes 108 classes and each class has seven iris images captured in two sessions, three in the first session and two in the second. Sessions were taken with an interval of one month. So there are totally 756 iris images with a resolution of 320×280 pixels.

4.2 Result analysis

It is known that the evaluation of the localization results of an algorithm based on observation by eye is likely to be inaccurate. Thus to achieve quantitative analysis of our algorithm's results, an analysis approach based on ground truth is adopted. In such approach, getting an accurate ground truth is vital to evaluate the performance of results. Since no ground truth is available, we hand-localized the iris center and radius for all iris images within the database, they serve as ground truth. Here, we localize the iris center and radius by manually fitting a circle on the iris outer boundary through three steps, all by hand. First, we identify the approximate iris center location, here we can show the location of the calculated iris center as a reference point. Then, we estimate the iris radius by finding a point located on the iris outer boundary. Finally, considering the center and radius

obtained from the previous two steps, a moveable and resizable circle is fitted to the iris outer boundary. After that and according to [16], a circularity confidence interval centered at the hand-localized pixel with five pixel radius, is defined. Let $H(i,j)$ denote hand-localized pixel and $E(i,j)$ the detected pixel. The distance Dis of H and E is defined as $\|H\text{-}E\|_2$. Then the accuracy of the algorithm is:

$$A = \begin{cases} (1 - Dis/5 \times 0.5) \times 100\%, & Dis \le 5 \\ 0, & Dis > 5 \end{cases} \quad (10)$$

The satisfactory factor is set to 0.5 in Eq.(10). That is to say, the accuracy is 50% if the detected pixel position is on the boundary of the confidence interval. If it is out of the confidence interval, the accuracy is set to 0. Meanwhile, the accuracy is 100% if the detected pixel and hand-localized one are at the same position. For more reliable performance evaluation, Daugman's integrodifferential method [4] as a prevailing segmentation method, is also implemented on the same database. The experiments were done under the condition that the same initial estimate of iris center is provided for both AIPF and Daugman methods. For the application of AIPF, six integration rectangles were adopted with: (θ_1=5°, θ_2=0°, θ_3=-5°, θ_4=-10°, θ_5=-15°,θ_6=-20°) in the right iris rectangle, other six integration rectangles are adopted with: (θ_1=175°, θ_2=180°, θ_3=185°, θ_4=190°, θ_5=195°, θ_6=-200°) in the left integration rectangle, and each integration rectangle has 60×15 dimensions. It is clear form Table 1 that AIPF method achieves better accuracy than that of Daugman since 72.75 % of the AIPF's detected iris centers are within the 5-pixel confidence interval of the ground truth, and 89.4% of the AIPF's estimated iris radii are within the confidence interval. Moreover, AIPF method performs faster than Daugman method. Therefore, the proposed method demonstrates high accuracy with faster execution.

Method	Accuracy within 5- pixel confidence		Time		
	Iris center	Iris radius	Mean	Min.	Max.
Daugman	69.84 %	79.1 %	0.73s	0.36s	1s
Proposed	72.75 %	89.4 %	0.27s	0.23s	0.4s

Table 1. Performance of the AIPF method.

5 Conclusion

An algorithm for the localization of eyeball or the iris outer boundary is reported in the paper. The proposed algorithm adopts radial edges detection with curve fitting in gray level iris images. First, the rough iris center is detected. Then, a set of radial edge points are detected based on AIPF. Finally, getting the precise iris features through fitting a circle to the detected edge points. Experimental results on a set of 756 iris images from CASIA V1.0 indicate high accuracy and faster execution due to its simplicity in implementation. Our future work has two fields. First, we will perform iris segmentation based on AIPF for iris recognition. Second, we will utilize AIPF for detecting other facial features.

References

[1] Baxes, G. A.: Digital Image Processing: Principles and Applications, Wiley, New York, 1994.

[2] Chinese Academy of Sciences - Institute of Automation, CASIA Iris Image Database (ver. 1.0), Available on: http://www.sinobiometrics.com.

[3] Cui, J., Wang, Y., Tan, T., Ma, L., Sun, Z.: An Iris Recognition Algorithm Using Local Extreme Points, Proceedings of the First International Conference on Biometrics Authentication, ICBA '04, Hong Kong, 2004, pp. 442-449.

[4] Daugman, J. G.: High Confidence Visual Recognition of Persons by a Test of Statistical Independence, IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 15**,** 1993, No. 11, pp. 1148–1161.

[5] Feng, X., Fang, C., Ding, X., Wu, Y.: Iris Localization with Dual Coarse-to-fine Strategy, Proceedings of the 18th International Conference on Pattern Recognition, ICPR'06, 2006, pp. 553-556.

[6] He, X., Shi, P.: A Novel Iris Segmentation Method for Hand-held Capture Device, Proceedings of the International Conference on Biometrics, ICB 06, Hong Kong, China, 2006, pp. 479-485.

[7] Ma, L., Tan, T., Wang, Y., Zhang, D.: Efficient Iris Recognition by Characterizing Key Local Variations, IEEE Transactions on Image Processing, Vol. 13, 2004, pp. 739-750.

[8] Masek, L. : Recognition of Human Iris Patterns for Biometric Identification, Master Thesis, University of Western Australia, 2003.

[9] Mohammed, G., Hong, B., Al-Kazzaz, A.: Accurate Pupil Features Extraction Based on New Projection Function, Computing and Informatics, To be appear in Vol. 29, 2010.

[10] Perona, P., Malik, J.: Scale-Space and Edge Detection Using Anisotropic Diffusion, IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 12, 1990, No.7, pp. 629-639.

[11] Philips, P., Bowyer, K., Flynn, P.: Comments on The CASIA Version 1.0 Iris Data Set, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 29, 2007, No. 10.

[12] Sudha, N., Puhan, N., Xia, H., Jiang, X.: Iris Recognition on Edge Maps, IET Computer Vision , Vol. 3, 2009, No.1, pp. 1-7.

[13] Sun, C., Zhou, C., Liang, Y., Liu, X. : Study and Improvement of Iris Location Algorithm, Proceedings of the International Conference on Biometrics, ICB 06, Hong Kong, China, 2006, pp. 436-442.

[14] Tisse, C., Martin, L., Torres, L., Robert, M.: Person Identification Technique Using Human Iris Recognition, Proceedings of the 15th International Conference on Vision Interface, VI '02, Calgary, Canada, 2002, pp. 294-299.

[15] Wildes, R. P.: Iris Recognition: An Emerging Biometric Technology, Proceedings of the IEEE, Vol. 85**,** 1997, No. 9, pp. 1348–1363.

[16] Zheng, Z., Yang, J., Yang, L.: A Robust Method for Eye Features Extraction on Color Image, Pattern Recognition Letters, Vol. 26**,** 2005, No. 14, pp. 2252-2261.

[17] Zhou, Z., Geng, X.: Projection Functions for Eye Detection, Pattern Recognition, Vol. 37, 2004, pp. 1049-1056.

[18] Zuo, J., Ratha, N. , Connell, J.: A New Approach for Iris Segmentation, Proceeding of IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Anchorage, Alaska, 2008,pp. 1-6.

Secure Convertible Authenticated Encryption Scheme Based on RSA

Tzong-Sun Wu
Department of Computer Science and Engineering, National Taiwan Ocean University, Keelung, 202, Taiwan
E-mail: ibox456@gmail.com

Han-Yu Lin
Department of Computer Science, National Chiao Tung University, Hsinchu, 300, Taiwan
E-mail: hanyu.cs94g@nctu.edu.tw

Keywords: authenticated encryption, digital signature, conversion, RSA

Received: November 24, 2008

A convertible authenticated encryption (CAE) scheme is a better way to simultaneously provide cryptographic schemes with the properties of confidentiality, authenticity and non-repudiation. The authors propose a RSA based secure CAE scheme which is different from previously proposed ones based on the discrete logarithms or elliptic curve discrete logarithms. The proposed scheme has the nice arbitration mechanism allowing the designated recipient to convert the authenticated ciphertext into an ordinary signature without any extra computation efforts or communication overheads for the public arbitration. Additionally, the security requirement of confidentiality against adaptive chosen ciphertext attacks (IND-CCA2) and that of unforgeability against existential forgery on adaptive chosen-message attacks (EU-CMA2) are proved in the random oracle model.

Povzetek: Predlagana shema se uporablja za sprotne zaupne transakcije.

1. Introduction

Since Diffie and Hellman [3] proposed the first public key system based on the discrete logarithms, the public key system has been widely applied to many fields. The encryption and the digital signature are two fundamental functions of public key systems. The former ensures the confidentiality while the latter ensures the authenticity and non-repudiation. Yet, some applications, such as the credit card transactions have to simultaneously fulfill the above properties. In 1994, Horster *et al*. [7] proposed an authenticated encryption (AE) scheme to realize the concept of providing digital signature schemes with the confidentiality. An AE scheme allows the signer to produce an authenticated ciphertext such that only the designated recipient has the ability to recover the message and verify its signature. It can be seen that the requirement of confidentiality is achieved in an AE scheme. In addition, it is not necessary to establish a session key between the signer and the designated recipient in advance. However, a later dispute over repudiation might occur, since the authenticated ciphertext is not publicly verifiable. To deal with the problem, in 1999, Araki *et al*. [1] proposed a convertible limited verifier signature scheme with a signature conversion mechanism. To complete the signature conversion process, the signer has to release an extra parameter, which is considered unworkable if the signer refuses to cooperate with. Besides, the computation cost of the conversion is rather high. In 2002, Wu and Hsu [16] proposed a convertible authenticated encryption (CAE) scheme with the efficient signature conversion process. In their scheme, the converted signature is just embedded in the authenticated ciphertext. Therefore, the designated recipient can solely reveal the converted signature in case of a later repudiation. Because the converted signature is derived during the verification process of the authenticated ciphertext, the signature conversion takes no additional computation cost or communication overhead. Since then, several CAE variants were proposed. In 2005, Chen and Jan [2] proposed CAE schemes using self-certified public key system. Later, Peng *et al*. [14] proposed a publicly verifiable authenticated encryption scheme with message linkages for transmitting a large message. Lv *et al*. [11] further proposed practical CAE schemes using self-certified public keys. Next, Wu *et al.* [17] proposed generalized CAE schemes based on elliptic curves [10, 13] for facilitating gradually popular applications like mobile phones and PDAs. In 2008, Wu *et al.* [18] elaborated the merits of CAE and multi-signature schemes [4-6, 8] to propose a convertible multi-authenticated encryption (CMAE) scheme. Nevertheless, these schemes are primarily based on the discrete logarithm problem (DLP) [3] or the elliptic curve discrete logarithm problem (ECDLP) [12] and not applicable to RSA-based systems [15].

1.1. Our Results

In this paper, the authors focus on the solution to confidential transactions of RSA-based systems and

propose a secure CAE scheme based on RSA. The signer can generate an authenticated ciphertext and only the designated recipient has the ability to verify it. The proposed scheme is efficient because it is not necessary to establish a session key in advance. The arbitration mechanism enables the recipient to reveal the ordinary signature for the public verification without extra costs, since the converted signature is obtained during the verification process of the authenticated ciphertext. Moreover, the security requirement of confidentiality against adaptive chosen ciphertext attacks (IND-CCA2) and that of unforgeability against existential forgery on adaptive chosen-message attacks (EU-CMA2) are proved in the random oracle model.

2. Preliminaries

In this section, we first define involved parties of a CAE scheme and then review the security notions with respect to RSA cryptosystems [15].

2.1. Involved Parties

A CAE scheme has two involved parties: a signer and a designated recipient. Each is a polynomial-time-bounded probabilistic Turing machine (PPTM). The signer will generate an authenticated ciphertext and deliver it to the designated recipient. Yet, a dishonest signer might repudiate his generated ciphertext. Finally, the designated recipient decrypts the ciphertext and verifies the signature.

2.2. Security Notions

***Definition 1 (RSA Problem)*:**
Let $N = pq$ where p and q are two large primes, e an integer satisfying that $gcd(e, (p-1)(q-1)) = 1$ and d an integer such that $ed = 1 \bmod (p-1)(q-1)$. Given $c \equiv m^e \pmod N$ as the input, output the integer m satisfying $m \equiv c^d \pmod N$.

***Definition 2 (RSA Assumption)*:**
Let $\boldsymbol{G}$ be the RSA key generator which takes the security parameter 1^k as its input and outputs (N, e, d, p, q). Given a RSA instance (N, e, c), the advantage for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, every positive polynomial $P(\cdot)$ and all sufficiently large k to solve the RSA problem is at most $1/P(k)$, i.e.,

$$\Pr[\mathcal{A}(N, e, c) = m, c \leftarrow m^e \bmod N, m \leftarrow Z_N, (N, e, d, p, q) \leftarrow \boldsymbol{G}] \le 1/P(k).$$

3. Proposed CAE Scheme Based on RSA

In this section, we propose a CAE scheme based on RSA. The proposed scheme can be divided into three phases: the authenticated ciphertext generation, the message recovery and signature verification, and the signature conversion phases. Initially, each user chooses two large primes p, q, and computes $N = pq$. Next, each user chooses an integer e relatively prime to $(p-1)(q-1)$ and computes d satisfying $ed \equiv 1 \pmod{\phi(N)}$ where $\phi(N)$ is the Euler function of N. Here, (N, e) and (p, q, d) are the public and the private keys of each user, respectively. Let h be a secure one-way hash function which accepts two variable-length inputs and generates a fixed-length output of size l. Details of each phase are described below:

***The authenticated ciphertext generation phase*:** For signing the message M, the signer U_s chooses an integer $c \in \{0, 1\}^l$ and computes

$$r = Mc^c \bmod N_v, \tag{1}$$

$$t = c^{e_v} \bmod N_v, \tag{2}$$

$$s = (h(M, c))^{d_s} \bmod N_s, \tag{3}$$

and then deliveries the authenticated ciphertext (s, r, t) to the designated recipient U_v. Note that l is a predefined security parameter to determine the output length of hash function.

***The message recovery and signature verification phase*:** Upon receiving the ciphertext (s, r, t), U_v first computes

$$c = t^{d_v} \bmod N_v. \tag{4}$$

He then recovers the message M as

$$M = rc^{-c} \bmod N_v, \tag{5}$$

and checks the redundancy embedded in M. U_v can further verify (s, r, t) by checking

$$s^{e_s} = h(M, c) \bmod N_s. \tag{6}$$

***The signature conversion phase*:** Since the parameter c is obtained during the verification of the authenticated ciphertext, the recipient can easily reveal the converted signature (s, c) along with the message M in case of a later repudiation. One can see that the conversion process is efficient for that it will not incur extra computation costs or communication overheads. Anyone can perform Eq. (6) to verify the correctness of the converted signature.

4. Security Proof

In this section, we first prove that the security of our proposed scheme is computationally related to RSA. We demonstrate that the proposed CAE scheme is correct and achieves the security requirements of confidentiality, unforgeability and non-repudiation. Then we evaluate the performance of our scheme and compare it with some previous works.

4.1. Security Proof

Correctness. A CAE scheme is correct if the signer can generate a valid authenticated ciphertext and only the designated recipient is capable of decrypting and verifying it. We prove the correctness of our proposed scheme as Theorems 1 and 2.

Theorem 1. *The designated recipient U_v can correctly recover the message M with embedded redundancy by Eq. (5).*

Proof: From the right-hand side of Eq. (5), we have

$$\begin{aligned} & rc^{-c} \\ &= (Mc^c)c^{-c} && \text{(by Eq. (1))} \\ &= M \pmod{N_v} \end{aligned}$$

which leads to the left-hand side of Eq. (5).

Q.E.D.

Theorem 2. *The designated recipient U_v can correctly verify the signature* (s, c) *with Eq. (6).*

Proof: From the right-hand side of Eq. (6), we have

$$\begin{aligned} & h(M,c) \\ &= h(M,c)^{d_s \cdot e_s} \\ &= s^{e_s} \pmod{N_s} && \text{(by Eq. (3))} \end{aligned}$$

which leads to the left-hand side of Eq. (6).

Q.E.D.

Message Confidentiality. A CAE scheme satisfies the security requirement of message confidentiality if the resulted authenticated ciphertext is computationally indistinguishable even with respect to two candidate plaintexts. We prove that the proposed scheme achieves the IND-CCA2 security as Theorem 3. The proof idea is a security reduction from the RSA problem to the adaptive chosen ciphertext attacks against our proposed scheme in the random oracle model. Let t_λ be the average running time of one oracle-query in the following proof.

Theorem 3. *The proposed CAE scheme is* $(t', q_h, q_{\text{sig}}, q_{\text{ver}}, \varepsilon')$*-secure against adaptive chosen ciphertext attacks in the random oracle model if there exists no probabilistic polynomial-time algorithm B that can* (t, ε)*-break the RSA problem, where*

$$\varepsilon \geq (1/q_h)(1 - q_{\text{ver}} q_h 2^{-l})\varepsilon', \tag{7}$$

$$t < t' + t_\lambda(q_h + q_{\text{sig}} + q_{\text{ver}}). \tag{8}$$

Proof: Suppose that A is a $(t', q_h, q_{\text{sig}}, q_{\text{ver}}, \varepsilon')$-PPTM that breaks the proposed CAE scheme with the chosen ciphertext attack, where t' denotes the running time, q_h the times of h-oracle queries, q_{sig} the times of authenticated ciphertext oracle queries, q_{ver} the times of signature verification oracle queries and ε' the probability that A succeeds. We will take A as a subroutine to construct a (t, ε)-algorithm B that solves the RSA problem with respect to the designated recipient's key pair in time t with the probability ε. The algorithm B is said to (t, ε)-break the RSA problem. Let U_v be the designated recipient with the key par (N_v, e_v) and (p_v, q_v, d_v). The objective of B is to obtain α ($\equiv b^{d_v} \pmod{N_v}$) by taking (N_v, e_v) and $b \in Z_{N_v}$ as inputs. In this proof, B simulates a challenger to A in the following game.

Phase 1: A issues the following kinds of queries adaptively:

– h-oracle query: When A issues a h-oracle query of $h(M, c)$, B first randomly chooses $a \in \{0, 1\}^l$, and computes $u \equiv a^{e_s} \pmod{N_s}$. B then keeps (M, c, a, u) in a hash oracle query table and returns u as a result.

– Authenticated-ciphertext-oracle query: When A issues a authenticated ciphertext oracle query of a message M, B randomly chooses $c \in \{0, 1\}^l$ to compute

$$r = Mc^c \bmod N_v,$$

$$t = c^{e_v} \bmod N_v.$$

B then checks whether $h(M, c)$ has been queried in the hash oracle query table. If it has not, B randomly chooses $a \in \{0, 1\}^l$ to compute $u \equiv a^{e_s} \pmod{N_s}$, writes (M, c, a, u) into the hash oracle query table and sets $s = a \bmod N_s$; else, B finds the corresponding a and sets $s = a \bmod N_s$. Finally, B returns the ciphertext (s, r, t) as the result of authenticated-ciphertext-oracle query for M.

– Signature-verification-oracle query: When A submits an authenticated ciphertext $\delta = (s, r, t)$, B searches the hash oracle query table of (M, c, a, u)'s. If one of them satisfies $s = a$ and $M = rc^{-c} \bmod N_v$, B outputs M. Otherwise, the $\square$ symbol is returned as a result to signal that the authenticated ciphertext is invalid.

Challenge: The PPTM A generates two messages, M_0 and M_1, of the same length. The challenger B flips a coin $\lambda \leftarrow \{0, 1\}$ and generates an authenticated ciphertext $\delta^* = (s^*, r^*, t^*)$ where $t^* = b \pmod{N_v}$ and (s^*, r^*) are randomly selected strings from some appropriate space.

Phase 2: A issues new queries as those stated in Phase 1. It is not allowed to make a signature-verification-oracle query for the target challenge δ^*.

Guess: A outputs a bit λ' as the result. If $\lambda' = \lambda$, A wins this game. We define A's advantage as $Adv(A) = \Pr[\lambda' = \lambda] - 1/2$.

Output: Finally, B randomly picks c from an entry (M, c, a, u) of the hash oracle query table and outputs it as the solution to $b^{d_v} \pmod{N_v}$. The probability of outputting a correct answer and the running time are bounded by the inequalities of Eqs. (7) and (8).

Analysis of the game: If A guesses correctly, i.e., $\lambda' = \lambda$, it has to compute

$$\alpha = t^{d_v} \bmod N_v,$$

and query $h(M_\lambda, \alpha)$ for checking whether

$$s^{*e_s} = h(M_\lambda, \alpha) \bmod N_s$$

holds or not. Then an entry $(M_\lambda, \alpha, a_i, u_i)$ should be recorded in the hash oracle query table for some a_i and u_i. It can be seen that the distribution of the PPTM A's view in the simulation is identical to that A is playing

in a real CAE scheme except the failure of signature-verification-oracle queries for some valid authenticated ciphertexts. Since there are at most q_h queries, the probability of rejecting a valid authenticated ciphertext is not greater than $q_h 2^{-l}$. In addition, A makes at most q_{ver} signautre-verification-oracle queries and β randomly chooses c from one of at most total q_h entries in the hash oracle query table. We can express the probability ε as $\varepsilon \geq (1/q_h)(1 - q_{\text{ver}} q_h 2^{-l})\varepsilon'$ which implies Eq. (7). The running time t of B is that of all oracle queries along with that of the PPTM A. Consequently, we obtain $t < t' + t_\lambda(q_h + q_{\text{sig}} + q_{\text{ver}})$ which implies Eq. (8).

Q.E.D.

Unforgeability. A signature scheme fulfills the security requirement of unforgeability if it is secure against adaptive chosen-message attacks. The security of unforgeability against existential forgery on adaptive chosen-message attacks (EU-CMA2) is proved in the random oracle model as Theorem 4. The proof concept of Theorem 4 is a security reduction from the RSA problem to the existential forgery attack against our proposed scheme in the random oracle model. Let t_λ be the average running time of one oracle-query in the following proof.

Theorem 4. *The proposed CAE scheme is* $(t', q_h, q_{\text{sig}}, \varepsilon')$*-secure against existential forgery on adaptive chosen-message attack in the Random Oracle model if there exists no polynomial-time algorithm* B *that can* (t, ε)*-break the RSA problem, where*

$$\varepsilon \geq (1/q_h)\varepsilon', \tag{9}$$

$$t < t' + t_\lambda(q_h + q_{\text{sig}}). \tag{10}$$

Proof: Suppose that A is a PPTM that can $(t', q_h, q_{\text{sig}}, \varepsilon')$-break the proposed scheme with the existential forgery attack, where t' denotes the running time, q_h the times of h-oracle queries, q_{sig} the times of authenticated-ciphertext-oracle queries and ε' the probability that A succeeds. We will take A as a subroutine to construct a (t, ε)-algorithm B that solves the RSA problem with respect to the signer's key pair in time t with the probability ε. Let U_s be the signer with the key par (N_s, e_s) and (p_s, q_s, d_s). The objective of B is to derive $\alpha (\equiv b^{d_s} (\bmod N_s))$ by taking (N_s, e_s) and $b \in Z_{N_s}$ as inputs. In this proof, B simulates a challenger to A in the following game.

Phase 1: A issues h-oracle and authenticated-ciphertext-oracle queries as those defined in Theorem 3 adaptively.

Challenge: The challenger B randomly chooses a message M^* for A to forge a signature.

Phase 2: A issues new queries as those stated in Phase 1. It is not allowed to make an authenticated ciphertext oracle query for the target challenge M^*. When A issues a h-oracle query of $h(M, c)$ with $M = M^*$ for the first time, B directly outputs b. Otherwise, B follows the same procedures as stated in Phase 1.

Response of forgery: A outputs a valid authenticated ciphertext (s, r, t) for M^* with the probability ε'.

Output: B outputs s as the solution to $b^{d_s} (\bmod N_s)$. The probability of outputting a correct answer and the running time are bounded by the inequalities of Eqs. (9) and (10).

Analysis of the game: Consider the case when $s \equiv \alpha \pmod{N_s}$, and then B has successfully computed $\alpha (\equiv b^{d_s} (\bmod N_s))$. It can be seen that the distribution of the PPTM A's view in the simulation is identical to that A is playing in a real CAE scheme. Besides, B has answered one of total q_h h-oracles A queried with the value b which will lead to the forged authenticated ciphertext (s, r, t) with $s \equiv \alpha \pmod{N_s}$. Consequently, the success probability ε to solve the RSA problem for B can be further expressed as $\varepsilon \geq (1/q_h)\varepsilon'$ which implies Eq. (9). The running time t of B is that of all oracle queries along with that of the PPTM A. Therefore, we can express it as $t < t' + t_\lambda(q_h + q_{\text{sig}})$ which implies Eq. (10).

Q.E.D.

According to Theorem 4, the proposed scheme is secure against existential forgery attacks. That is, the signature key can not be forged and the signer can not repudiate having generated his signatures. Hence, we obtain the following corollary.

Corollary 1. *The proposed CAE scheme satisfies the security requirement of non-repudiation.*

4.2. Performance and Comparison

For facilitating the reader with the following performance evaluation, we first define some used notations:

- T_h: the time for performing a one-way hash function h;
- T_e: the time for performing a modular exponentiation computation;
- T_m: the time for performing a modular multiplication computation;
- T_i: the time for performing a modular inverse computation;

Note that the time for performing modular addition and modular subtraction is ignored because it is negligible as compared to those of performing other computations. The detailed evaluation of our proposed scheme in terms of computational costs is shown as Tables 1.

Phase	Computational costs
Authenticated ciphertext generation	$3T_e + T_m + T_h$
Message recovery and signature verification	$3T_e + T_m + T_i + T_h$
Signature conversion	0

Table 1: Performance evaluation of proposed scheme.

We compare the proposed scheme with some previous works including the Wu-Hsu scheme [16] (WH for short), Lv *et al*.'s [11] (LWK for short), Araki *et al*.'s scheme [1] (AUI for short) and Huang *et al*.'s [9] (HLL for short). Detailed comparisons in terms of the security and functionalities are demonstrated as Table 2. To the best of our knowledge, the proposed scheme is the first provably secure one based on RSA assumption.

Item \ Scheme	WH	LWK	AUI	HLL (2)	Ours
No conversion cost	O	O	×	O	O
Security assumption	DLP	DLP	DLP	RSA	RSA
Confidentiality	N.A.	-	-	N.A.	IND-CCA2
Unforgeability	- (1)	-	N.A.	N.A.	EU-CMA2

Table 2: Comparisons of proposed and other schemes.

Remarks: (1) It is unknown that what security level with respect to the evaluated item the scheme can achieve, since it provides no formal proofs.

(2). To obtain fair comparison results, we assume that only one signer is involved and responsible for generating the authenticated ciphertext.

5. Conclusions

In this paper, the authors proposed a secure CAE scheme based on RSA as a solution to confidential transactions of RSA-based systems. The proposed scheme allows the signer to produce an authenticated ciphertext and only the designated recipient can recover the message and verify the signature for ensuring the confidentiality. The arbitration mechanism provides the designated recipient with the ability to solely reveal the ordinary signature for the public verification. It can be seen that the signature conversion process is rather simple and efficient for that the converted signature is obtained during the verification process of the authenticated ciphertext. That is, the conversion process takes no extra computation efforts or communication overheads. Moreover, the security requirement of confidentiality against adaptive chosen ciphertext attacks (IND-CCA2) and that of unforgeability against existential forgery on adaptive chosen-message attacks (EU-CMA2) are proved in the random oracle model.

References

[1] S. Araki, S. Uehara and K. Imamura, "The limited verifier signature and its application," *IEICE Transactions on Fundamentals*, 1999, **E82-A**(1): 63-68.

[2] Y. H. Chen and J. K. Jan, "Enhancement of digital signature with message recovery using self-certified public keys and its variants," *ACM SIGOPS Operating Systems Review*, 2005, **39**(3): 90-96.

[3] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, 1976, **IT-22**(6): 644-654.

[4] L. Harn, "Group-oriented (t, n) threshold digital signature scheme and digital multisignature," *IEE Proceedings - Computers and Digital Techniques*, 1994, **141**(5): 307-313.

[5] L. Harn and T. Kiesler, "New scheme for digital multisignature," *Electronics Letters*, 1989, **25**(15): 1002-1003.

[6] L. Harn, C.Y. Lin and T.C. Wu, "Structured multisignature algorithms," *IEE Proceedings - Computers and Digital Techniques*, 2004, **151**(3): 231-234.

[7] P. Horster, M. Michel and H. Peterson, "Authenticated encryption schemes with low communication costs," *Electronics letters*, 1994, **30**(15): 1212-1213.

[8] C.L. Hsu, T.S. Wu and T.C. Wu, "New nonrepudiable threshold proxy signature scheme with known signers," *The Journal of Systems and Software*, 2001, 58(2): 119-124.

[9] C.H. Huang, C.Y. Lee, C.H. Lin, C.C. Chang and K.L. Chen, "Authenticated encryption schemes with message linkage for threshold signatures," *Proceedings of the IEEE 19th International Conference on Advanced Information Networking and Applications*, Vol. 2, 2005, pp. 261-264.

[10] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, 1987, **48**(177): 203-209.

[11] J. Lv, X. Wang and K. Kim, "Practical convertible authenticated encryption schemes using self-certified public keys," *Applied Mathematics and Computation*, 2005, **169**(2): 1285-1297.

[12] A. Menezes, *Elliptic Curve Public Key Cryptosystems*, Kluwer Academic Publishers, 1993.

[13] V. Miller, "Use of elliptic curves in cryptography," *Advances in Cryptology, CRYPTO'85*, Springer-Verlag, 1985, pp. 417-426.

[14] Y.Q. Peng, S.Y. Xie, Y.F. Chen, R. Deng and L.X. Peng, "A publicly verifiable authenticated encryption scheme with message linkages," *Proceedings of the Third International Conference on Networking and Mobile Computing*, ICCNMC, Zhangjiajie, China, 2005, pp. 1271-1276.

[15] R. Rivest, A. Shamir and L. Adleman, "A method

for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, 1978, **21**(2): 120-126.

[16] T.S. Wu and C.L. Hsu, "Convertible authenticated encryption scheme," *The Journal of Systems and Software*, 2002, **62**(3): 205-209.

[17] T.S. Wu, C.L. Hsu and H.Y. Lin, "Efficient convertible authenticated encryption schemes for smart card applications in network environments," *Proceedings of the 9th World Multi-Conference on Systemics, Cybernetics and Informatics, WMSCI2005*, Orlando, Florida, U.S.A., July 2005.

[18] T.S. Wu, C.L. Hsu, K.Y. Tsai, H.Y. Lin and T.C. Wu, "Convertible multi-authenticated encryption scheme," *Information Sciences*, 2008, **178**(1): 256-263.

Extended Symbolic Mining of Textures with Association Rules

Igor Kononenko and Matjaž Bevk
Faculty of computer and information science, University of Ljubljana, Tržaška 25
E-mail: igor.kononenko@fri.uni-lj.si and lkm.fri.uni-lj.si/xaigor

Keywords: texture description, association rules, image mining, multi-resolution, meta-association rules, ArTex algorithm

Received: January 25, 2008

The association rules algorithms can be used for describing textures if an appropriate texture representation formalism is used. This representation has several good properties like invariance to global lightness and invariance to rotation. Association rules capture structural and statistical information and are very convenient to identify the structures that occur most frequently and have the most discriminative power. This paper presents the extended textural features which are based on association rules. We extend the basic textural features of our algorithm ArTex in three ways, using (a) various interestingness measures, (b) the multi-resolution approach, and (c) the meta-parameters. Results of our experiments show that the extended representation improves the utility of basic textural features and often gives better results, comparable to standard texture descriptions.

Povzetek: Z metodami asociacijskih pravil je razvit algoritem za analizo tekstur.

1 Introduction

Texture is a commonly used term in computer vision, although it is difficult to precisely define it. We can regard an image texture as a function of spatial variation in pixel values. There exist many different approaches to characterize textures. Most texture features are based on structural, statistical or spectral properties of the image. Well known statistical features are based on gray-level cooccurrence statistics[12]. Examples of structural features are features of Voronoi tesselation[27], representations using graphs[28], representations using grammars[22] and representations using association rules[23]. We have developed[2] a similar approach to that of Rushing et al.[23]. We showed that our approach, implemented with algorithm ArTex, as opposed to that of Rushing et. al, is rotation-invariant as well as brightness-invariant, produces significantly smaller descriptions and is significantly faster[2]. Due to excellent efficiency, we decided to upgrade our approach with additional descriptors that may lead to better descriptions of target images, although with an increased time and description complexity. This paper presents the extended textural features, based on association rules, of algorithm ArTex, as described in Ref. [2]. We extend the basic textural features in three ways, using (a) various interestingness measures, (b) the multi-resolution approach, and (c) the meta-parameters. We show that the extended parameters together with the basic ones are appropriate for effective description of images and can be efficiently induced. The purpose of using association rules is to obtain a structural description of textures. The ultimate goal, which we hope to reach in further work, is to get higher order association rules, which would capture the global structure of the image and would also allow for a transparent (human readable) description of the image. The present study is a step towards that goal.

This paper is organized as follows: Section 2 gives a brief definition of association rules, describes a texture representation which is suitable for processing with association rule algorithms, and shows how association rules are used for feature description of textures with algorithm ArTex. In Section 3 we define the three extensions to the basic feature set. The following section shows a comparison between the extended feature set and the basic one, as well as the comparison with other algorithms for image description. In Section 5 we show the results of the Extended ArTex in four real world data sets, and finally the last section concludes and gives ideas for further work.

2 Texture descriptions with association rules

2.1 Association rules

Association rules were introduced by Agrawal et al.[1] back in 1993. The following is a formal statement of the problem: Let $\mathcal{I} = \{i_1, i_2, ..., i_m\}$ be a set of literals, called items (also called transaction elements). Let $\mathcal{D}$ be a set of transactions, where each transaction T is a set of transaction elements such that $T \subseteq \mathcal{I}$. We say that a transaction T contains B, if $B \subseteq T$. An *association rule* is an implication of the form $B \Longrightarrow C$, where $B \subset \mathcal{I}$, $C \subset \mathcal{I}$ and $B \cap C = 0$. The rule $B \Longrightarrow C$ holds in the transaction set $\mathcal{D}$ with *confidence* c if c% of transactions in $\mathcal{D}$ that contain B also contain C. The rule $B \Longrightarrow C$ has *support* s in the

transaction set $\mathcal{D}$ if s% of transactions in $\mathcal{D}$ contain $B \cup C$. The problem is to find all association rules in transaction set $\mathcal{D}$ with confidence of at least *minconf* and support of at least *minsup*, where *minconf* and *minsup* represent the lower boundaries for confidence and support of association rules, respectively.

2.2 Texture representation

To apply association rules algorithms on textures one must first define the terms which are used in association rules in the context of textures.

Pixel $\vec{A}$ of a texture p is a vector $\vec{A} = (X, Y, I) \in p$, where X and Y represent absolute coordinates and I represents the intensity of pixel A.

Root pixel $\vec{K}$ is the current pixel of a texture $\vec{K} = (X_K, Y_K, I_K)$.

e **neighborhood** $N_{e,\vec{K}}$ is a set of pixels which are located in the circular area of radius e with root pixel $\vec{K}$ at the center. Root pixel $\vec{K}$ itself is not a member of its neighborhood.

$$N_{e,\vec{K}} = \{(X, Y, I) \,|\delta \leq e\} \setminus \vec{K}$$
$$\delta = \left\lfloor \sqrt{(X_K - X)^2 + (Y_K - Y)^2} + 0.5 \right\rfloor \quad (1)$$

Transaction $T_{e,\vec{K}}$ is a set of elements based on its corresponding neighborhood. The elements of transaction are represented with Euclidean distance and the intensity difference from the root pixel.

$$T_{e,\vec{K}} = \left\{ (\delta, I_K - I) | \, (X, Y, I) \in N_{e,\vec{K}} \right\} \quad (2)$$

Transaction element is a two dimensional vector $(r, i) \in T_{e,\vec{K}}$, where the first component represents Euclidean distance from the root pixel and the second component represents the intensity difference from the root pixel.

Association rule is composed of transaction elements; therefore it looks like this

$$(r_1, i_1) \wedge \cdots \wedge (r_m, i_m) \Longrightarrow (r_{m+1}, i_{m+1}) \wedge \cdots \wedge (r_{m+n}, i_{m+n})$$

Transaction set $D_{p,e}$ is composed of transactions, which are derived from all possible root pixels of a texture p at certain neighborhood size e. Possible root pixels are only those for which an e neighborhood would not extend outside of the texture's borders.

$$D_{p,e} = \left\{ T_{e,\vec{K}} \,\middle|\, \forall \vec{K} : \vec{K} \in p \right\}$$

This representation of a texture replaces exact information of the location and intensity of the neighboring pixels with more vague information of distance and the relative intensity of neighboring pixels. This description is also rotation invariant.

The described representation is almost suitable for processing with general association rule algorithms. What is still to be considered is the form of a transaction element. The association rule algorithms expect scalar values for transaction elements, whereas our representation produces a two dimensional vector for a transaction element. Let us say that intensity of each texture point can have values from interval $[0..(Q-1)]$ and that the neighborhood size is e. Take some transaction element (r, i), where i holds a value from $[-(Q-1)..+(Q-1)]$ and r holds a value from $[1..e]$. What is needed here is a bijective mapping that transforms each vector to its scalar representation. A possible and quite straightforward solution is:

$$s = (2Q-1)(r-1) + i + (Q-1) \quad (3)$$

Equation (3) maps each (r, i) pair into its unique scalar representation s, so that values from $[0..2Q-2]$ represent pixels which are located on distance 1 from the root pixel, values from $[2Q-1..4Q-3]$) represent pixels which are located in distance 2 from the root pixel, and so on.

The transformation is also reversible:

$$r = 1 + s \; div \; (2Q-1)$$
$$i = s \; mod \; (2Q-1) - (Q-1)$$

Now it is possible to define a transaction that suits the general association rule algorithms:

$$T_{e,Q,\vec{K}} = \left\{ s \;\middle|\; \begin{array}{l} (r, i) \in T_{e,\vec{K}}, \\ s = (2Q-1)(r-1) + i + (Q-1) \end{array} \right\}$$

And finally we obtain the appropriate transaction set definition:

$$\mathcal{D}_{p,e,Q} = \left\{ T_{e,Q,\vec{K}} \,\middle|\, \forall \vec{K} : \vec{K} \in p \right\}$$

2.3 From association rules to feature description

We are ready to describe the algorithm ArTex[2]. Preprocessing consists of the following steps: conversion to gray scale, pixel quantization to Q levels (to make it faster and more accurate), and selection of the neighborhood size e (there is a trade-off between better descriptions with greater e and better time complexity with smaller e).

It is important to understand that ArTex does not perform the final concept induction, it extracts features from images. These features are later used by external machine learning algorithm to induce the model (for classification of images). Let P represent the whole set of images. We isolate a small subset of images $P_f \subset P$ which will be used for feature extraction, the rest of images $P_l = P \backslash P_f$ will be used for learning. The selection of images for P_f is random, but it has to be ensured that each class is represented with equal amount of images.

2.3.1 Feature extraction with ArTex

The whole algorithm ArTex is given in Algorithm 1. The most important is the part that follows the preprocessing phase and the image subset selection phase. This part of the algorithm is denoted with lines from 6 to 12. Each image from $p \in P_f$ is transformed into a transaction set $\mathcal{D}_{p,e,Q}$. The process of this transformation is described in Section 2.2. Transaction sets are then processed by algorithms AprioriN and GenRulesN, which return most important coocurrencies of pixels in the form of association rules.

2.3.2 Calculation of feature values

ArTex (Algorithm 1) is composed of two major parts. The first part (lines from 2 to 12) is the feature extraction, and the second part (lines from 13 to 16) is a process of evaluating feature values. The second part iterates through images from P_l, where features for each image are evaluated. There are two types of features: support and confidence for extracted association rules. We presume that we have predefined functions $\rho_i(p, q)$ which evaluate feature q of type i (support or confidence) on image p. The result is stored in matrix d, where element d_{ij} represents the value of feature j on image i.

2.3.3 Modifications of Apriori and GenRules

Original algorithm Apriori[1] (line 8 in Algorithm 1) takes as a parameter the minimal required support (minsup) of transaction tuples. It is impossible to know in advance what is the appropriate value of minsup. If it is too large, no tuples will be found, and if it is too small, too many tuples will be generated which may drastically increase the time complexity (and also the description complexity) of ArTex. For that reason we use a slightly modified algorithm Apriori which as the input takes the number of tuples (ntup) to return, rather than the minimal support. It executes standard Apriori in an iterative manner, reducing the minimal support in each iteration. The modified Apriori is much more stable than standard Apriori and has no problems with time and description complexity.

Similarly as we modified Apriori, we also modified the GenRules algorithm, which is used to generate association rules from tuples[1] (line 9 in Algorithm 1). The same problem, as with minsup parameter in Apriori, algorithm Genrules has with the parameter minconf (minimal required confidence). We therefore use a modified GenRules algorithm which, instead of the minimal confidence, takes the minimal number of rules (minrul) to return.

```
 1: procedure ARTEX(P set of images, e, Q, ntup, minrul)
 2:     for all p ∈ P do
 3:         quantize pixels of image p to Q levels;
 4:     end for
 5:     select feature extraction subset P_f and learn/test subset P_l, P = P_f ∪ P_l;
 6:     for all p ∈ P_f do                          ▷ feature extraction
 7:         represent p in transaction form D_{p,e,Q};
 8:         r_sup = apriori(D_{p,e,Q}, ntup);
 9:         r_conf = genrules(r_sup, minrul);
10:         ρ_sup = ρ_sup ∪ r_sup;
11:         ρ_conf = ρ_conf ∪ r_conf;
12:     end for
13:     i = 0;
14:     for all p ∈ P_l do   ▷ calculate feature values for P_l
15:         j = 0;
16:         for all ϱ ∈ ρ_sup do
17:             d_{i,j} = φ_sup(p, ϱ)
18:             j = j + 1
19:         end for
20:         for all ϱ ∈ ρ_conf do
21:             d_{i,j} = φ_conf(p, ϱ)
22:             j = j + 1
23:         end for
24:         i = i + 1;
25:     end for
26:     return d;  ▷ returns a matrix of extracted features d
27: end procedure
```

Algorithm 1: Algorithm ArTex for feature extraction. Note that the number of features is determined dynamically and therefore the upper bounds for indices i and j cannot be determined in advance.

3 Extending the default feature set

Our model of texture is such that the structure of association rules also describes some aspects of the textural structure. Since we are interested in parametric description of

a texture, this structure has to be represented with one or more numerical parameters (numerical features).

3.1 Using various interestingness measures

Until now we presented the algorithm which uses only the basic interestingness measures support and confidence, which were defined together with association rules[1]. Nowadays, they are still most widely used, but studies have shown that there are some concerns especially with confidence measure, which can be misleading in many practical situations, as shown by Brin et al.[3]. They also offered an alternative to evaluate association rules using the χ^2 test. Contrary to confidence measure, the χ^2 test could be used to find both positively and negatively correlated association patterns. However, the χ^2 test alone may not be the ultimate solution because it does not indicate the strength of correlation between items of the association pattern. It only decides whether items of the association pattern are independent of each other, thus it cannot be used for ranking purposes.

We use χ^2 test just to select interesting association patterns, which are later described by the Pearson's correlation coefficient (ϕ-coefficient) as advised in Ref. [25]. Besides ϕ-coefficient, ArTex can also compute seven additional association rule interestingness measures, which are a subset of collection made by Tan et al.[26]. Table 1 lists interestingness measures that form an extended parameter set of the Extended ArTex.

Table 1: Extended set of interestingness measures for association rules ($A \Rightarrow B$). $P(X)$ represents the probability of X.

#	*measure*	*formula*
1	ϕ-coefficient	$\frac{P(A,B)-P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	odds ratio	$\frac{P(A,B)P(\overline{A},\overline{B})}{P(A,\overline{B})P(\overline{A},B)}$
3	conviction	$\frac{P(A)P(\overline{B})}{P(A,\overline{B})}$
4	interest	$\frac{P(A,B)}{P(A)P(B)}$
5	Jaccard	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
6	Pia.-Shapiro	$P(A,B) - P(A)P(B)$
7	J-measure	$P(A,B)\log(\frac{P(B\|A)}{P(B)}) + P(A,\overline{B})\log(\frac{P(\overline{B}\|A)}{P(\overline{B})})$
8	gini index	$P(A)[P(B\|A)^2 + P(\overline{B}\|A)^2] + P(\overline{A})[P(B\|\overline{A})^2 + P(\overline{B}\|\overline{A})^2] - P(B)^2 - P(\overline{B})^2$

It is possible now to define an upgraded version of ArTex - ArTexG - which extracts features based on an arbitrary set of interestingness measures. Besides other input parameters, ArTexG takes a list of interestingness measures $\vec{\varphi}$ and their corresponding constraints $\vec{t}$. In the heart of the algorithm is $Genrules_k$ function, which extracts best rules according to k-th function from Table 1. This function is obtained by replacing the confidence criteria in the common $GenRules$ algorithm with corresponding criteria from Table 1. When the first part of the algorithm finishes (lines from 1 to 14) it returns best association rules in sets ρ_{φ_k}, which were selected according to the selection criteria φ_k. The second part (lines from 15 to 31) is the feature evaluation process, which is very similar to original ArTex (Algorithm 1), except that it calculates feature values for all types of features $\vec{\varphi}$.

```
1:  procedure ArTexG(P set of images,e,Q,φ⃗ measures, t⃗ constraints)
2:      for all p ∈ P do
3:          quantize pixels of image p to Q levels;
4:      end for
5:      select feature extraction subset P_f and learn/test subset P_l, P = P_f ∪ P_l;
6:      for all p ∈ P_f do                          ▷ feature extraction
7:          represent p in transaction form D_{p,e,Q};
8:          r_sup = apriori(D_{p,e,Q}, t_1);
9:          ρ_sup = ρ_sup ∪ r_sup;
10:         for all φ_k do
11:             r_{φ_k} = genrules_k(r_sup, t_k);      ▷ uses the k-th interestingness measure
12:             ρ_{φ_k} = ρ_{φ_k} ∪ r_{φ_k};
13:         end for
14:     end for
15:     i = 0;
16:     for all p ∈ P_l do  ▷ calculate feature values for P_l
17:         j = 0;
18:         for all ϱ ∈ ρ_sup do
19:             d_{i,j} = φ_sup(p, ϱ)
20:             j = j + 1
21:         end for
22:         for all φ_k do
23:             for all ϱ ∈ ρ_{φ_k} do
24:                 d_{i,j} = φ_k(p, ϱ)
25:                 j = j + 1
26:             end for
27:         end for
28:         i = i + 1;
29:     end for
30:     return d;  ▷ returns a matrix of extracted features d
31: end procedure
```

Algorithm 2: General algorithm ArTexG for feature extraction. Note that the number of features is determined dynamically and therefore the upper bounds for indices i and j cannot be determined in advance.

3.2 Using more than one resolution

Each induced association rule discovers a certain pattern in the textures. The next extension of parameters comes from the issue of the pattern's scale. Not every combination of scale and neighborhood size can guarantee that the pattern

would be detected. The problem is illustrated in Figures 1 and 2, where a fixed neighborhood e requires an appropriate resolution.

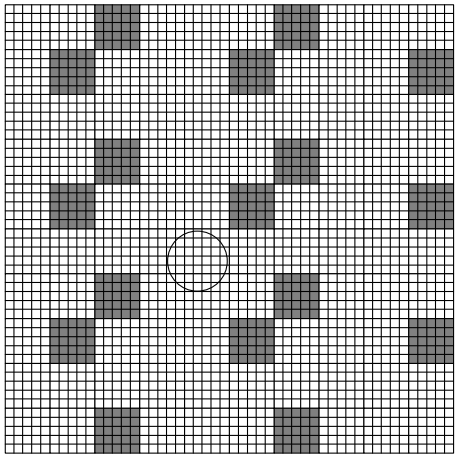


Figure 1: A very inadequate resolution for a fixed neighborhood.

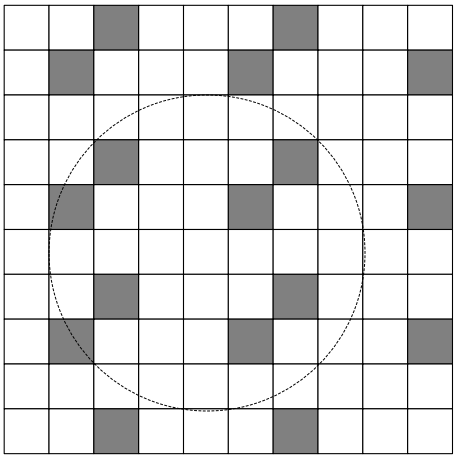


Figure 2: An adequate resolution for the example from Fig. 1.

We use the following definition: An image T_2 of size $M_2 \times N_2$ has a resolution $\frac{1}{2}R$, if and only if an image T_1 of size $M_1 \times N_1$ has resolution R where $\frac{M_2}{M_1} = \frac{N_2}{N_1} = \frac{1}{2}$ and images T_1 and T_2 represent the same pattern.

To increase the possibility that the pattern will be detected, we propose a framework, where the extraction of association rules is repeated at different texture resolutions.

3.3 Meta parameters

To supply the learning algorithm with a more general view of the texture, we extended the parameter set with parameters based on meta association rules.

Let us split the learning set of images as follows:

$$P_l = P_{l_1} \cup P_{l_2},$$

Previously we defined P_f as a small subset that we use for feature extraction (see Section 2.3), thus set $P_f \cup P_l$ represents all textures. Meta association rules are calculated on a learning subset of images P_{l_1}. Further, let us denote with φ_i the i-th interestingness measure, with ρ_{p_j,φ_i} an association rule that was discovered by using measure φ_i on texture p_j, and similarly with ρ_{*,φ_i} a set of association rules that was discovered by using measure φ_i on the feature extraction texture set P_f. The generation of meta-features is divided into three steps as follows:

1. First a set of binary features is constructed by using the default association rule set as follows. For each texture $p \in P_{l_1}$ and for each interestingness measure φ_i and for each association rule $\varrho \in \rho_{*,\varphi_i}$ the value of the binary feature is 1 if association rule ϱ was selected as important on texture p according to measure φ_i, otherwise the value of binary feature is 0.

2. This feature set along with the texture classes are then fed to an algorithm for generating associations rules, which induces rules with only the class on the consequent side. In our implementation we use algorithm Tertius[8]. Therefore, the output of Tertius is a set of rules of the form $A \Rightarrow B$ where consequent B always consists only of the class, and features in the antecedent A are directly responsible for the outcome of the class. This way the output could easily be used for classification. Note that the"useful knowledge" of such a rule lies in the antecedent A which is a conjunctive combination of the binary features that were derived from the basic association rules.

3. The meta feature set is constructed by calculating the rules' antecedents A on each texture from P_l. Note that the learning algorithm would have a trivial task and a strong bias if the learning set would consist of only of P_{l_1}, however, by adding P_{l_2} the task is hard enough. The introduced bias (hopefully) expresses the meta-regularity, discovered by the Tertius learning algorithm. Also note that meta features are binary features.

4 Experimental evaluation of the extended feature set

We performed experiments for comparing various versions of our algorithm with each other and with other texture description algorithms. For that purpose we used publicly available data bases.

4.1 Benchmark problem Domains

Here is a description of data bases used in our experiments:

– Outex[20]
This data base contains a large variety of surface textures. The collection is well defined in terms of vari-

ations in illumination, rotation and spatial resolution. We chose the following collections of textures from the classification group:

- Outex 0
  The collection contains 480 images of 24 textures (classes). Each texture has 20 images. The dimension of images is 128×128 pixels. The images contain no intentionally induced variation of illumination, rotation or spatial resolution.
- Outex 1
  The collection is similar to Outex 0, except that this collection contains 2112 images (88 per texture) and that they are of size 64×64 pixels.
- Outex 2
  The collection is similar to Outex 0, except that this collection contains 8832 images (368 per texture) and that they are of size 32×32 pixels.
- Outex 10
  The collection contains 4320 images, 24 textures; 180 images per texture. Textures are captured at different rotations of the surface: $0°$, $5°$, $10°$, $15°$, $30°$, $45°$, $60°$, $75°$, $90°$. The size of images is 128×128 pixels.
- Outex 11
  The collection contains 960 images, 24 textures; 40 images per texture. Textures are captured at two resolutions 100dpi and 120dpi. The size of images is 128×128 pixels.
- Outex 12
  The collection contains 4800 images, 24 textures; 200 images per texture. The textures contain illumination variations and are captured at different rotations of the surface: $0°$, $5°$, $10°$, $15°$, $30°$, $45°$, $60°$, $75°$, $90°$. The size of images is 128×128 pixels.

– JerryWu[33]
  This data base was composed by Jerry Wu for his PhD. The collection contains 2100 images, 36 textures; 53 images per texture. Images were captured at different surface rotations: $0°$, $30°$, $60°$, $90°$, $120°$, $150°$, $180°$ and different camera tilt angles. The size of images is 128×128 pixels.

– Brodatz A[29]
  This collection contains a subset of textures from Brodatz album of materials[4]. It is composed of 2048 images of 32 textures, each texture is represented with 64 images, of which 16 are original images, 16 are randomly rotated images, 16 are captured at different resolutions and 16 are randomly rotated images at various resolutions. The size of images is 64×64 pixels.

– Brodatz B[11]
  This collection contains a subset of textures from Brodatz album of materials[4]. It is composed of 1248 images of 13 textures, each texture is represented with 96 images. Textures are captured at 6 different surface rotations: $0°$, $30°$, $60°$, $90°$, $120°$, $150°$. The size of images is 128×128 pixels.

– Brodatz C[5]
  This collection is also a subset of textures from Brodatz album of materials[4]. It is composed of 6720 images of 15 textures, each texture is represented with 448 images. Textures are captured at 6 different surface rotations: $0°$, $30°$, $60°$, $90°$, $120°$, $150°$, $200°$. The size of images is 32×32 pixels.

4.2 Analysis of the extended feature set

In this section we present results of experiments on benchmarck domains, described above. The quality of image descriptions was tested by measuring the classification accuracy, obtained by algorithm SMO[21], a support vector machine variant, implemented in Weka data mining tool[32]. SMO was selected among a set of classifiers because it consistently achieved the best classification accuracy in most of the experiments. The presented results are obtained by 10-fold cross validation.

4.2.1 The effectiveness of additional interestingness measures

The classification accuracies, obtained by extending the default parameter set each time with one of the measures from Table 1, are presented in Table 2.

Due to unreliability of t-test[30, 6] we performed further analysis using the Friedman test[9, 10], followed with Bonferroni-Dunn test[7]. The tests confirmed that seven measures significantly improved the results, while the odds-ratio significantly decreased the accuracy. The best result was obtained with J-measure. With further experiments we tried to verify if the we can obtain a better extension of the default parameter set with several parameters at once. The results were verified with Wilcoxson test[31] which revealed that adding any other measure besides J-measure does not improve the descriptions. Therefore, it is sensible to extend the default parameter set with J-measure alone.

Due to space limitation we do not show the analysis of time complexity and the numbers of generated features. The conclusions of this analysis is, that the generation of the default parameter set and the extended parameter set have approximately the same time complexity, however the number of generated features is for the extended parameter set much greater (on average for 83%), as could be expected.

4.2.2 The effectiveness of the multi-resolution approach

In the experiments we extended the default parameter set, with added J-measure, on original resolution R by adding

Table 2: Classification accuracies (%) of SMO on the default description versus the extended ones by various interestingness measures. The last four lines give averages and st. deviation over all datasets, the value of p for t-test and the number of wins by the extended description.

	default	ϕ-k.	Jacc.	conv.	gini	inter	J-meas	odds.	P.S.
outex 0	96.32	94.61	96.08	92.90	95.10	95.34	95.83	93.14	95.34
outex 1	83.04	82.45	82.99	82.06	82.84	83.63	83.53	81.86	83.38
outex 2	39.10	39.59	39.21	37.71	39.68	39.45	39.57	37.88	39.27
outex 10	93.48	94.66	93.81	94.37	94.37	94.14	94.63	94.07	93.74
outex 11	95.27	95.72	95.38	94.03	96.29	96.17	96.17	93.47	95.50
outex 12	96.62	97.50	96.64	96.81	97.19	97.12	97.19	96.64	96.66
JerryWu	86.29	98.59	87.85	98.34	97.02	84.97	96.12	95.61	93.80
brodatz B	89.75	91.57	90.16	92.23	91.15	91.40	91.40	91.40	89.50
brodatz A	78.13	84.63	79.41	85.96	82.22	85.04	82.32	82.12	79.61
brodatz C	57.36	58.20	57.71	58.05	58.16	59.12	58.41	57.23	57.72
average	81.54	83.75	81.92	83.25	83.40	82.64	83.52	82.34	82.45
st. dev.	19.05	19.60	19.03	19.84	19.46	18.84	19.45	19.57	19.23
p (t-test)		0.06	0.03	0.14	0.06	0.08	0.03	0.25	0.13
wins		8	8	6	8	8	9	5	8

parameters for resolutions $\frac{1}{2}R$ and for resolutions $\frac{1}{2}R+\frac{1}{4}R$ (therefore, in the latter case the feature extraction was run three times, each time on a different resolution). Table 3 gives classification accuracies of SMO classifier on all benchamark datasets for three sets of resolutions. The Friedman test shows $(p < 0.05)$ that the differences are significant, and the Bonferroni-Dunn test confirms that both multiresolutional descriptions (with two and three resolutions) achieved significantly better results than the single resolution description, however, the former two do not differ significantly between each other. The number of extracted features, as expected, increases approximately linearly with the number of additional resolutions, and the time complexity increases somewhat less than linearly.

4.2.3 The effectiveness of meta parameters

We compared the quality of description of the default parameter set, extended with J-measure, on two resolutions (R and $\frac{1}{2}R$), with the same description, extended with meta parameters, as described in Section 3. The classification accuracies for both descriptions are given in Table 4. In seven out of ten domains the meta parameters helped to improve the description, however, the Wilcoxson test $(p < 0.05)$ shows that overall there are no significant differences between the two descriptions. The number of features increases for 12% on average, and the average time complexity is 7 times higher for the extended description.

4.3 Comparison of Extended ArTex with other algorithms

We compared the Extended ArTex with several other algorithms that describe images in terms of a set of numerical features. We selected the following algorithms: a similar algorithm to ArTex, also based on association rules, developed by Rushing et al.[23], second order statistics as a standard benchmark algorithm[12] and the three advanced approaches which are effective and also often applied[13]: Laws filters[18], Haar waves, and Gabor waves[19]. Table 5 gives the classification accuracies of SMO algorithm using the 10-fold cross-validation.

Table 3: Classification accuracy (%) of SMO using the default parameter set, with J-measure included, when generated on one resolution R, two resolutions $R + \frac{1}{2}R$, and three resolutions $R + \frac{1}{4}R + \frac{1}{8}R$. The last four lines have the same interpretation as in Table 2.

	R	$R+\frac{1}{2}R$	$R+\frac{1}{2}R+\frac{1}{4}R$
outex 0	95.83	98.05	95.11
outex 1	83.53	83.97	81.32
outex 2	39.57	52.26	54.78
outex 10	94.63	98.38	98.26
outex 11	96.17	97.07	95.84
outex 12	97.19	97.95	97.84
JerryWu	96.12	99.40	99.75
brodatz B	91.40	97.60	97.60
brodatz A	82.32	94.01	94.77
brodatz C	58.41	60.93	58.41
average	83.52	87.96	87.37
st. dev.	19.45	17.24	17.03
p (t-test)		0.01	0.03
wins		10	7

The Friedman test $(p < 0.05)$ followed by the Bonferroni-Dunn test $(p < 0.05)$, shows that the Extended

Table 4: Classification accuracy (%) of SMO using the default parameter set, with included J-measure, on two resolutions $R + \frac{1}{2}R$, without and with added meta-parameters. The last four lines have the same interpretation as in Table 2. Columns d in |d| represent the difference of classification accuracies and its absolute value, and the last column contains ranks for |d|, used by the Wilcoxson test.

	without meta p.	with meta p.	d	|d|	rank
outex 0	98.05	97.92	-0.13	0.13	2
outex 1	83.97	87.71	+3.74	3.74	9
outex 2	52.26	55.00	+2.74	2.74	7
outex 10	98.38	98.54	+0.16	0.16	3
outex 11	97.07	97.50	+0.43	0.43	4
outex 12	97.95	98.96	+1.01	1.01	6
JerryWu	99.40	99.41	+0.01	0.01	1
brodatz B	97.60	96.84	-0.76	0.76	5
brodatz A	94.01	97.19	+3.18	3.18	8
brodatz C	60.93	53.12	-7.81	7.81	10
average	87.96	88.22			
st. dev.	17.24	18.31			
p (t-test)		0.40			
wins		7			

ArTex is significantly better than the second order statistics, while there are no significant differences between ArTex and other algorithms. This confirms that the Extended ArTex achieves the performance of best algorithms for generating texture descriptions. With respect to the time complexity, the most time consuming algorithms are Gabor waves, Rushing and Extended Artex. When compared with Extended Artex, the Gabor waves algorithm is on average 141 times slower, the algorithm by Rushing et al. has on average the same time complexity, while the Laws filters are 2 times faster, the second order statistics 10 times faster and the Haar waves 15 times faster.

Table 5 indicates that the Extended ArTex achieves best results in domains with relatively large images (128×128 or more), and performs relatively poorly on domains with small images, like *Outex 2* and *Brodatz C*. This is in accordance with our expectations. For reliable description of images, ArTex requires a large texture in order to be able to reliably evaluate the values of parameters - statistical estimates - whose accuracy depends on the number of pixels.

5 The performance of Extended Artex in several real-world applications

In this section we provide the performance comparison with other feature extraction algorithms in some applications which were implemented at our institution by using the Extended ArTex. We briefly outline the real-world data sets and then we provide the results.

We applied our approach on four real-world problems:

– *pH6pH10*
The problem is to differentiate microscopic pictures of 0.1% suspension of Al_2O_3 in destilled water (H_2O) with two different acidities: pH6 and pH10. The pictures were prepared at the University of Stuttgart[16, 17]. The size of images is 300×300 pixels. There are 30 images for each of the two classes.

– *Materials*
Each of six different materials was photographed 50 times on various places. The size of images is 200×200. Therefore this domain consists of six classes, where each class contains 50 samples. The task is to classify new images into one of six classes.

– *SyringeStactometer*
This domain consists of two classes of microscopic images of dried drops of tap water, that have to be differentiated. The first class of 52 drops was created using a syringe, whereas the second class of 61 drops was created using a stactometer. The size of images is 640×640.

– *Coronas*
This domain consists of GDV images of human fingertips. GDV images are obtained from BEO GDV Camera, developed by Korotkov[15]. The camera captures 320×240 gray scale images. The first class of 289 images consists of fingertips of humans in normal state, and the second class of 217 images consists of fingertips of humans in the altered state of consciousness. Most of images were obtained from Technical University SPIFMO in St. Petersburg, Russia. The task is to automatically detect the altered state of human

Table 5: Classification accuracies (%) of SMO using descriptions of various feature extraction algorithms. The last four lines have the same interpretation as in Table 2.

	Ext.ArTex	Laws	Haar	2^{nd}Stats	Gabor	Rushing
outex 0	97.92	91.25	93.13	72.29	99.38	97.07
outex 1	87.71	90.25	89.39	71.73	98.30	87.40
outex 2	55.00	84.59	72.09	59.70	94.49	24.35
outex 10	98.54	83.96	86.94	82.62	97.80	86.16
outex 11	97.50	92.19	93.75	75.63	99.79	97.97
outex 12	98.96	82.72	87.81	76.98	96.83	86.23
JerryWu	99.41	22.95	83.19	75.86	53.76	83.51
brodatz B	96.84	78.21	94.07	96.00	86.70	69.15
brodatz A	97.19	62.30	65.43	18.31	87.89	12.96
brodatz C	53.12	66.96	65.43	48.88	70.77	11.73
average	88.22	75.54	83.12	67.80	88.57	65.65
st. dev.	18.31	20.99	11.34	21.43	15.12	35.07
p (t-test)		0.10	0.14	0.01	0.47	0.01
wins		3	3	1	5	2

consciousness, using the GDV camera.

Table 6 gives the classification accuracies of SMO obtained by 10-fold cross-validation on the described datasets. The best results were achieved by the Extended ArTex, except in domain Coronas, where the second order statistics perform the best.

6 Conclusions

We described the extended textural features which are based on association rules. We extended the basic textural features of our algorithm ArTex[2] in three ways: by using various interestingness measures, by using the multi-resolution approach, and by using the meta-parameters.

Among various interestingness measures, the most promising results were achieved with J-measure. J-measure is, as far as we know, the best measure for evaluating the quality of decision (if-then) rules, developed by Smyth and Goodman back in 1990[24]. It has nice properties, such as nonnegativity, transparent interpretation (the sum of J-measures over all possible values is equal to the information gain[14]), it is possible to estimate the upper bound, and it is approximately χ^2 distributed. Therefore, it is not surprising, that it achieves the best results in our experiments.

Results from our experiments show that the multi-resolution approach enables to obtain significantly better descriptions of images. For texture descriptions with association rules this could be expected. Not every combination of scale and neighborhood size can guarantee that the pattern would be detected. As illustrated by Figures 1 and 2, the effectiveness of descriptions, which use an upper bound for the neighborhood around the central pixel, highly depends on the scale. As it is not known in advance which resolution will provide the best results, our approach combines the features, obtained from several resolutions. In our current research we try to develop an efficient algorithm for automatic detection of a small subset of relevant resolutions.

Our definition of meta-parameters is the first step towards the ultimate goal, which we hope to reach in further work: to get higher order association rules, which would capture the global structure of the image and would also allow for transparent (human readable) description of the images. The results, presented in this paper, are promising and have encouraged us to continue with research in this direction.

Overall, the extended representation improves the utility of basic textural features and often gives better results with respect to basic features, derived from association rules, as well as with respect to standard texture descriptions. The results in real-world applications prove the utility of our approach.

Our current research is devoted to describe medical images in order to obtain reliable diagnostic rules. The preliminary experiments in two medical domains are quite promising. In diagnosing the coronary artery disease from myocardial perfusion scintigraphy images the preliminary results show the significant improvement over the accuracy of physicians experts. In another diagnostic problem, based on the whole-body bone scintigrams, the picture is not so clear, still however, the Extended Artex achieves significantly better results than any other approach to feature extraction.

Yet another line of research is the visualization of textures from their association rule descriptions. As the number of extracted features may be relatively large (typically several hundreds of features are generated in each problem domain), such amount of features cannot be transparent to

Table 6: Classification accuracies (%) on real-world data sets of SMO using descriptions of various feature extraction algorithms. The last four lines have the same interpretation as in Table 2.

	Ext.ArTex	Laws	Haar	2^{nd}Stat.	Gabor	Rushing
pH6pH10	100.00	100.00	100.00	100.00	100.00	100.00
Materials	100.00	97.00	99.67	99.00	100.00	100.00
Syrin.Stact.	89.91	87.42	86.74	85.96	85.83	80.73
Coronas	82.80	80.25	80.23	84.78	78.81	70.00
average	93.18	91.17	91.66	92.44	91.16	87.68
st. dev.	8.40	9.04	9.81	8.18	10.60	14.88
p (t-test)		0.03	0.08	0.30	0.09	0.10
wins		0	0	1	0	0

a human expert. To overcome this we plan to develop a visualization tool for a user-friendly presentation of derived (large) sets of features in the form of artificial textures. The preliminary results in this area indicate that users may find on such artificially generated textures important patterns, related to the target concepts, and that users may be able to notice also more or less obvious differences in artificially generated textures for different classes of images.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26-28 1993.

[2] M. Bevk and I. Kononenko. Towards symbolic mining of images with association rules: Preliminary results on textures. *Intelligent Data Analysis*, 10(4):379–393, 2006.

[3] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations. In J. Peckham, editor, *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*, pages 265–276. ACM Press, 1997.

[4] P. Brodatz. *Textures - A Photographic Album for Artists and Designers*. Reinhold, Dover, New York, 1966.

[5] J. M. Carstensen. Cooccurrence feature performance in texture classification. In *The 8th Scandinavian Conference on Image Analysis, Tromsø, Norway*, pages 831–838, may 1993.

[6] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[7] O. J. Dunn. Multiple comparisons among means. *Journal of the American Statistical Association*, 56:52–64, 1961.

[8] P. A. Flach and N. Lachiche. Confirmation-guided discovery of first-order rules with tertius. *Machine Learning*, 42(1/2):61–95, 2001.

[9] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 2:675–701, 1937.

[10] M. Friedman. A comparison of alternative tests of significance for the problem of m rankings. *Annals of Mathematical Statistics*, 11:86–92, 1940.

[11] G. M. Haley and B. S. Manjunath. Rotation-invariant texture classification using a complete space-frequency model. *IEEE Tr. Im. Proc.*, 8(2):255–269, Feb. 1999.

[12] R. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, 3(11):610–621, 1973.

[13] P. E. T. Jorgensen. *Analysis and probability: wavelets, signals, fractals*, volume 234 of Graduate Texts in Mathematics. Springer, New York, 2006.

[14] I. Kononenko and M. Kukar. *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. Horwood Publ., 2007.

[15] K. Korotkov. *Aura and Consciousness*. State Editing & Publishing Unit “Kultura”, 1998. St.Petersburg, Russia.

[16] B. Kröplin, M. Hein, J. Herrmann, and B. Heusel. Report on the microoptic investigations of water and watery solutions. Technical report, University of Stuttgart, Institute for Statics and Dynamics of Aerospace Structures, 05-2000.

[17] B. Kröplin, M. Hein, J. Herrmann, and B. Heusel. Mikrooptische Untersuchungen zum Einfluss von elektromagnetischen Feldern, Magneten und Handys auf Wasser. Technical report, Universität Stuttgart, Institut für Statik und Dynamik der Luft und Raumfahrtkonstruktionen, 06-2000.

[18] K. I. Laws. *Textured image segmentation*. PhD thesis, Dept. Electrical Engineering, University of Southern California, 1980.

[19] B. S. Manjunath and W. Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(8):837–842, 1996.

[20] T. Ojala, T. Mäenpää, M. Pietikäinen, J. Viertola, J. Kyllönen, and S. Huovinen. Outex- new framework for empirical evaluation of texture analysis algorithms. In *ICPR (1)*, pages 701–706, 2002.

[21] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods: support vector learning*, pages 185–208. MIT Press, 1999.

[22] A. Rosenfeld and A. Kak. *Digital Picture Processing (2nd Edition)*, volume 2. Academic Press, Inc., 1982.

[23] J. A. Rushing, H. S. Ranganath, T. H. Hinke, and S. J. Graves. Using association rules as texture features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(8):845–858, 2001.

[24] P. Smyth and R. M. Goodman. Rule induction using information theory. In G. Piatetsky-Shapiro and W. Frawley, editors, *Knowledge Discovery in Databases*. MIT Press, 1990.

[25] P. Tan and V. Kumar. Interestingness measures for association patterns: A perspective. Technical Report TR00-036, Department of Computer Science, University of Minnesota, 2000.

[26] P. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *Proceedings of the Eight ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002.

[27] M. Tuceryan and A. Jain. Texture segmentation using voronoi polygons. *IEEE Transactions on Patern Analysis and Machine Intelligence*, pages 211–216, 1990.

[28] M. Tuceryan and A. Jain. *The Handbook of Pattern Recognition and Computer Vision (2nd Edition)*. World Scientific Publishing Co., 1998.

[29] K. Valkealahti and E. Oja. Reduced multidimensional cooccurrence histograms in texture classification. *PAMI*, 20(1):90–94, January 1998.

[30] G. I. Webb. MultiBoosting: A technique for combining Boosting and Wagging. *Machine Learning*, 40(2):159–196, 2000.

[31] F. Wilcoxson. Individual comparisons by ranking methods. *Biometrics*, 1, 1945.

[32] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.

[33] J. Wu. *Rotation Invariant Classification of 3D Surface Texture Using Photometric Stereo*. PhD thesis, Heriot-Watt University, Edinburgh, 2003.

Systematic Construction of Software Architecture Supported by Enhanced First-Class Connectors

Abdelkrim Amirat[1,2] and Mourad Oussalah[1]
[1]LINA Laboratory LINA CNRS UMR 6241, University of Nantes, France
[2]University Center of Souk Ahras, Algeria
E-mail: {abdelkrim.amirat; mourad.oussalah}@univ-nantes.fr

Keywords: logical architecture, physical architecture, first class connector, connection manager, modelling software architecture, C3 metamodel

Received: November 10, 2008

To provide hierarchical description from different software architectural viewpoints we need more than one abstraction hierarchy and connection mechanisms to support the interactions among components. Also, these mechanisms will support the refinement and traceability of architectural elements through the different levels of each hierarchy. Current methods and tools provide poor support for the challenge posed by developing system using hierarchical description. This paper describes an architecture-centric approach allowing the user to describe the logical architecture view where a physical architecture view is generated automatically for all application instances of the logical architecture.

Povzetek: Prispevek se ukvarja s povezovalnimi mehanizmi za podporo interakcijam med komponentami na področju programskega inženirstva.

1 Introduction

Modeling and representation of software architectures are the main phases of the development process of complex software systems [36]. The representation of an architecture is based on the concepts of component (loci of computation), connector (loci of communication), and configuration (arrangement of components and connectors, and properties of that arrangement) in order to describe the structure of the system at a higher level of abstraction than objects or lines of code. This representation provides several advantages over the life cycle of a software [10].

Components have always been considered to be the fundamental building blocks of software systems. The ways the components of a system interact are determinant for establishing the global system properties that emerge from the way the individual components are interconnected. Hence, component interactions have been promoted to first class design entities as well, and architectural connectors have emerged as a powerful tool for supporting the design of these interactions [29, 32].

Although the use of connectors is widely accepted at the conceptual level, their explicit representation at the implementation level is not always left to be necessary. For example, the Darwin [14] architecture description language does not include connectors. However, we feel that distinct conceptual entities should correspond to distinct implementation entities, so that they can truly become first-class and be manipulated as such. In fact, as argued in [20], the current level of support that ADLs provide for connector building is still far from the one awarded to components. For instance, although a considerable amount of work can be found on several aspects of connectors [2, 20, 33, 35]. Further steps are still necessary to achieve a systematic way of constructing new connectors from existing ones. Yet, the ability to manipulate connectors in a systematic and controlled way is essential for promoting reuse and incremental development, and to make it easier to address complex interactions.

Certainly, having a representation of the software architecture allows an easy exchange between the architect and programmer. Also, during the phases of maintenance and evolution, this representation helps to locate defects and reduces the risk of improper assembly of a new feature in the system. In addition, the distinction which exists between components and connectors allows a more explicit representation between the functional aspects and these of communication and therefore, makes the system easier to understand and to change. Finally, architecture-based components are also useful to facilitate the reuse of certain parts of the system represented by configurations [1].

In contrast to the industrial world, which offers components strongly linked to servers, systems or models owners [30], the academic approach is interested in formalizing the notion of software architecture (ADL). The ADLs provide a high level of abstraction for the specification and development of software systems. Today, several ADLs are defined, to help in the development of component-based systems, such as Rapide [11], SADL [22], UniCon [34], C2 [37], Darwin [12], MetaH [31], Wright [1], and ACME [9, 10] from the "*first generation*" of ADLs and UML 2.0 [5] , AADL [3], Koala [25], and xADL 2.0 [7] from the "*second*

generation" of ADLs. The classification of ADLs in generations has been introduced by Medvidovic [19].

In this article, we take a step towards this goal by proposing a metamodel for the description of software architecture called C3 (three "C" for *Component, Connector, and Configuration*). The specificities of this metamodel are: First, proposing a new structure and new types of connectors, second, definition and manipulation of configurations as first classes entities and third, description of architectures from two different views, a model architecture view (logical architecture) created by the architect and an application architecture view (physical architecture instances of the logical architecture) generated automatically which serves as support to maintain the consistency and the evolution of the application architectures.

After this introduction, the remainder of this article is organized as follows: Section 2 provides the motivations of our research. In section 3 presents the concept of a logical architecture with the key elements of the proposed metamodel. The physical architecture is defined in section 4. The last section concludes this work with a summary of our ongoing research.

2 Motivations

Our main motivation is to propose a metamodel to maintain the consistency of an architecture using new types of connectors with a richer semantics. Using these connectors, systems are built like a Lego Blocks (*Puzzle*) by assembling components and connectors, where each element can be only placed in the right place in the architecture puzzle. We find in most existing ADLs and notation languages that:

- The definition and instantiation of connectors are often merged in a single operation.
- The management of connectors does not take into account the semantic composition hierarchies when positioning and establishing links between components and their composites.
- Few models allow reuse connectors (for example through inheritance) and to define new connectors by their reuse.
- There is no direct and automatic correspondence between architectures (*model*s) and applications built following these architectures (*instances*).

In order to overcome these shortcomings we propose in this paper, a metamodel (C3) for describing hierarchical software architecture, based on the definition of two types of architecture. A logical architecture defined by the user and a physical architecture built by the system and conforms to the logical architecture. The metamodel will make its contribution towards the following objectives:

***O1*:** Provide a higher abstraction level for connectors in order to make them more generic and more reusable.

O2: Take into account the semantics of several types of relationships. In our case; we explore the association relationship between components, the composition relationship among architectural elements, and the propagation relationship to describe software systems at different levels of details.

O3: Promote the maintenance and the evolution of architectures by the possibility of adding, deleting and substitution of different elements in the architecture.

O4: The principle of reuse should be widely exploited. New components and connectors can be defined by combining already existing elements through inheritance and/or composition mechanisms. Basically, we have defined a set of generic, reusable connectors and extensible to support new structural and behavioural relations among components.

O5: Explicit connectors must be preserved through a declarative interface that hides the management mechanism of the inside glue-protocol.

O6: Using the physical and the logical architecture, we can separate the functional aspects of architectural elements and the non-functional aspects related to the management of their consistency.

3 Logical Architecture (LA)

Our approach is based on the description of software architecture following two architectural views. The first one is a logic view defined by the architect by assembling the compatible elements available in the library of element types and the second one is a physical view constructed automatically by the system and serves as a support for user applications built in accordance with the logical architecture.

The large majority of ADLs consider components as entities of first class. So, they make a distinction between component-types and component-instances. However, this is not the case with other concepts such as connectors and configurations. In our metamodel we consider each concept recognized by the C3 metamodel as an architectural element of the first class citizen. So, each architectural element may be positioned on one of the three abstraction levels defined in the following section. We believe that it is necessary to reify the core architectural elements in order to be able to represent and manipulate them and let them evolve easily.

3.1 Abstraction levels

In our approach, software architectures are described in accordance to the first three levels of modelling defined by the OMG [23, 24]. The application level (A_0) which represents the real word application (an instance of the architecture), the architecture level (A_1) which represents the architecture model and meta-architecture level (A_2) which represents the meta-language for the description of the architecture. The three abstraction levels are defined as follows (on Figure 1).

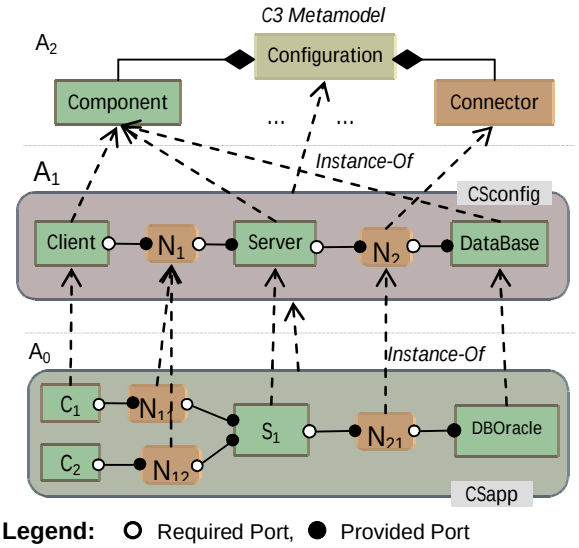


Figure 1: Architecture abstraction levels.

3.1.1 Meta-architecture level (A_2)

In this level we find the standard definition of any architectural element proposed by a large set of ADLs to describe software architectures. We consider the most common elements namely components, connectors, and configurations. Section 3.2 will summarize the description of the core elements of the C3 metamodel.

3.1.2 Architecture level (A_1)

This level is used to describe any architecture model using one or more instances of architectural building blocks defined at the meta-architecture level (A_2). Figure 1 shows a client/server architecture configuration (*CSconfig*) type which is defined using the following three components types: client component type, server component type and data base component type; and two variants of RPC connector types: N_1 between the client type and the server one, and N_2 between the server type and the data base type.

3.1.3 Application level (A_0)

At this level (implementation level) one or more applications can be built according to the architecture described at the above level (A_1). Each architectural element of the implementation level is an instance of an element-type of the architecture model. For example we can build from the previous client/server architecture the application *SCapp* (Figure 1) which is an instance of the *CSconfig* configuration assembled from C_1 and C_2 instances of the client component; *DBOracle* instance the Data base component; S_1 instance of the server component; N_{11} and N_{12} instances of connector type N_1 and finally N_{21} instance of connector type N_2. This figure shows only one application architecture (*CSapp*), more application architectures could be instantiated.

We have presented in this section the concept software architecture through its core concepts and its various abstraction levels. We have focused on the important concepts to address the key issue of connectors in software architecture description.

3.2 Basic concepts of C3 metamodel

3.2.1 Architectural elements

In our metamodel described in Figure 2, an architectural element may be a component, a connector or architectural configuration[1]. A configuration represents a graph of components and connectors. A component or a connector is a composite when it is composed of other internal architectural elements. A component or connector is primitive when it is atomic (without internal structure).

An architectural element may have several properties as well as constraints on these properties, as it may have one or more possible implementations. The interaction points of each architectural element with its environment are the interfaces. Each architectural element is defined by its interfaces through which they publish its required and provided services to and from its environment. Each service may use one or more ports. We approach in the following sections with more detail the most important concepts of our C3 metamodel.

3.2.2 Component

A generally accepted view of a software component is that it is a software unit with *provided services* and *required services*. The provided services are operations performed by the component. The required services are the services needed by the component to produce the provided services. The interface of a component consists of the specifications of its provided and required services. It should specify any *dependencies* between its provided and required services. To specify these dependencies precisely, it is necessary to match the required services to the corresponding provided services. Services are carried using ports. Thus, we can define a generic interface of a component type as follows:

Component typeName (*requiredInterf , provideInterf*);

3.2.3 Connector

Connectors are architectural building blocks used to model the interactions between components and rules that govern these interactions. They correspond to lines in box-line descriptions. Examples are pipes, procedure call, method in-vocation, client-server protocol, and SQL link between database and application. Unlike components, connectors may not correspond to compilation entities. However, the specifications of connectors in an ADL may also contain rules to implement a specific type of connectors. In general connectors have been developed without regard to reuse or extension. Current ADLs can be classified into three different kinds: 1- ADLs without connectors, ADLs with

[1] "Architectural configuration" will, at various times in this paper, be referred to simply as "graph" or "topology".

predefined set of connectors, and ADLs with explicit connector types.

- **ADLs with implicit connectors.** There are ADLs that prefer the absence of connector because they distort the compositional nature of software architectures. Some ADLs, such as Darwin [13], Leda [6], and Radipe [11] do not consider connectors as first class citizens. However these ADLs make difficult the reusability of components because they have the coordination process tangled with the computation inside them, and they are aware of the coordination process that has to happen in order to communicate with the rest. The notion of connector emerges from the need to separate the interaction from the computation in order to obtain more reusable and modularized components and to improve the level of abstraction of software architecture description [18]. Mary Shaw [32] presents the need for connectors due to the fact that the specification of software systems with complex coordination protocols is very difficult without the notion of connector. Hence, connector provides not only a high level of abstraction and modularity to software architectures, but also an architectural view of the system instead of the object-oriented view of compositional approaches. So, it is important to defend the idea of considering connectors as first-order citizens of ADLs.
- **ADLs with predefined set of connectors.** UniCon [33, 34] is a typical representative of ADLs supporting a predefined set of built-in connector types only. The semantics of built-in connector types are defined as part of the language, and are intended to correspond to the usual interaction primitives supported by underlaying operating system or programming language. A connector in the UniCon language is specified by its *protocol*. A connector's protocol consists of the connector's type, specific set of properties, and a list of typed roles. Each *role* serves as a point through which the connector is connected to a component. UniCon currently supports seven built-in connector types which represent the basic classes of interactions among components: Pipe, FileIO, Procedure Call, Remote Procedure Call, Data Access, RT Scheduler, and PL Bundler. These connectors cannot be instantiated nor evolved. Composite connectors are composed only from connectors.
- **ADLs with explicit connector types.** Most ADLs provide connectors as first order citizens of the language such as: ACME [10], Aesop [8], C2 [15, 16, 17], SADL [21], Wright [1], ArchWare's π-ADL [26, 27], xADL [7], AADL [3] etc. All of these languages go a step forward with regard to the previous kind of ADLs. They improve the reusability of components and connectors by separating computation from coordination.

In our approach we opt for the third category of connectors (explicit connector types). So, in the C3 metamodel we present some explicit and generic types of connectors that the user can specialize following her/his needs in each application field. We will focus with details on this concept in section 3.3.

3.2.4 Configuration

A configuration represents a graph of components and connectors. Configuration specifies how components are connected with connectors (Figure 3). This concept is needed to determine if the components are well connected, whether their interfaces agree, and so on. A configuration is described by an interface which enables the communication between: the configuration and its external environment, and the configuration and its internal components.

Configuration typeName (*requiredInterf, provideInterf*);

The following UML diagrams (Figure 2 and 3) represent the main elements of C3 metamodel. For clarity reason, these diagrams present a simplified version of our metamodel. In the rest of this article we will only deal with connectors with more detail as they represent the mainstream of our research topic in this paper. In addition, the relationship connector-configuration and connector-component will be highlighted in the text.

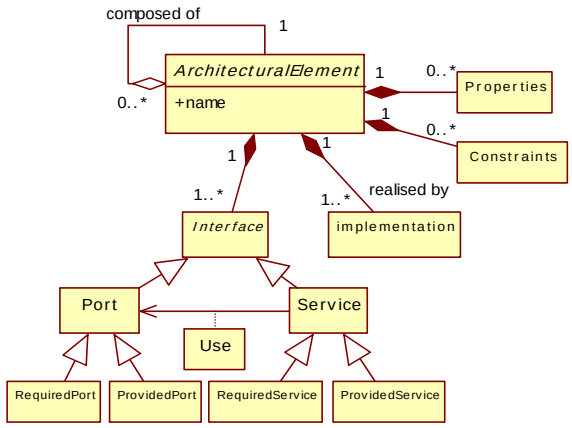


Figure 2: Structure of an architectural element in C3.

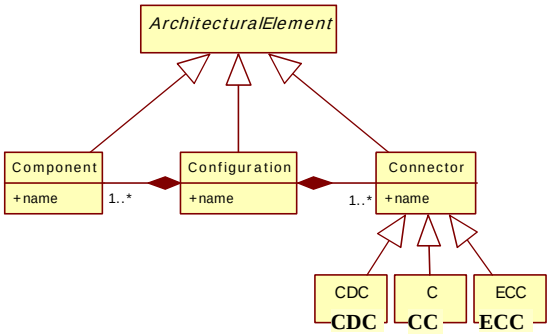


Figure 3: Component, connector, and configuration in C3.

3.3 Connector in C3

A connector is mainly represented by an *interface* and a *glue* specification [28]. Basically, the *interface* shows the necessary information of the connector, including the number of interaction points, service type that a connector provides (communication, conversion,

coordination, facilitation), connection mode (synchronous, asynchronous), transfer mode (parallel, serial) etc. In C3 interaction points of an interface are called *Ports*. A *port* is the interface of a connector intended to be tied to a component interface (a component's *port*). In the context of the frame, a *port* is either a *provided port* or a *required port*. A *provided port* serves as entry point to a component interaction represented by a connector type instance and it is intended to be connected to the *required port* of a component (or to the *required port* of another connector). Similarly, a *require port* serves as the outlet point of a component interaction represented by a connector type instance and it is intended to be connected to the *provide port* of a component (or to the *provide role* of another connector). The number of ports within a connector denotes the *degree* of a connector type. For example, in client-server architecture a connector type representing procedure call interaction between client and server entities is a connector with degree two. More complex interactions among three or more components are typically represented by connector types of higher degrees. Consequently, the interface is the visible part of connector; hence it must contain enough information regarding the service and the type of this connector. By doing this, one can decide whether or not a given connector suits its qualifications by examining its interface only.

The *glue* specification describes the functionality that is expected from a connector. It represents the hidden part of a connector. The *glue* could be just a simple protocol links ports or it could be a complex protocol that does various operations including linking, conversion of data format, transferring, adapting, etc. in general the glue of a connector represents the connection type of that connector. Connectors can also have an internal architecture that includes computation and information storage. For example a connector would execute an algorithm for converting data from format A to format B or an algorithm for compressing data before it transmits them. Hence, the service provided by connectors is defined by its *glue*; the services of a connector could be either communication service, conversion service, coordination service, or facilitation service.

In case of composite connectors the sub-connectors and sub-components of the composite connector must be defined in the glue, as well as the binding among the sub-connectors and sub-components.

The general signature form of the connector interface is a follows:

Connector typeName (requiredInterf, provideInterf);

3.3.1 Connector structure

Our contribution at this level consists in enhancing the structure of connectors by encapsulating the attachment links (figure 4). So, the application builder will have to spend no effort in connecting connectors with its compatible components and/or configurations. Consequently, the task of the developer consists only in choosing from the library the suitable type of connectors where its interfaces are compatible with the interfaces of component/configuration types of which are expected to be assembled.

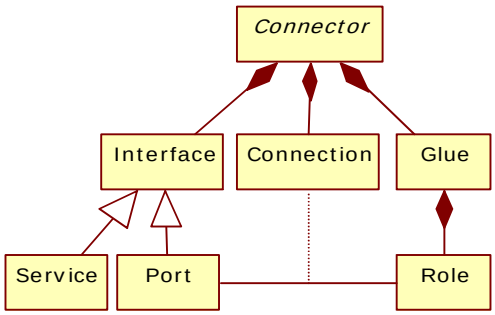


Figure 4: Connector structure.

In order to illustrate the properties of C3 metamodel and the associated connector definition, a case study is going to be used throughout the paper. The case study is a client-server configuration (CS-config) organized around a client-server relationship. In this configuration we have a client and a server. The server component itself is defined by a configuration (S-config) whose internal components are Coordinator (Coor.), securityManager (SM) and dataBase (DB). These elements are interconnected via connector services that determine the interactions that can occur between the server and client on one hand and between the server and its internal elements on the other hand. These connectors are represented in Figure 5 by solid-lines.

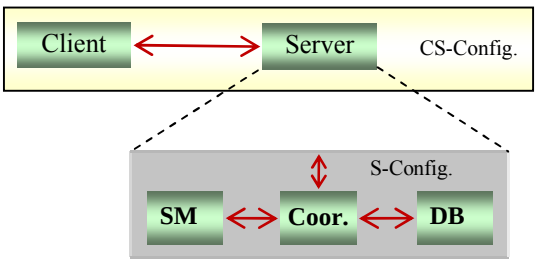


Figure 5: Client–Server Architecture.

In Figure 6.a we describe the structure of the RPC connector used to connect the client component (C) with the server component (S). In this new structure the RPC connector encapsulates attachments that represent links between the client and server.

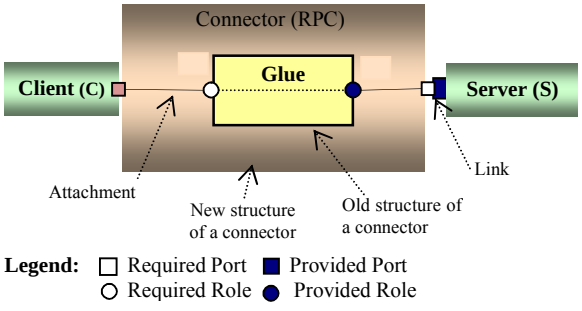


Figure 6.a: Connector structure in C3.

```
Connector RPC ( C.P1, S.P1 )    // Connector interface
{
   Proprieties = { List of properties };
   Constraints = { List of constraints };
   Services = { List of services };
   HierarchicalLevel = (C.Level = S.Level); //decomposition level
   Glue = {Roles ={{R1 , R2}; R1 = R2 }};  // simple case of a glue
   Attachments = { R1 to C.P1, R2 to S.P1 }; //attachments
      }
```

Figure 6.b: Connector description in C3.

Figure 6.b represents the signature specification of the connector RPC. Inside this connector type we have the glue code which describes how the activities of the client and server are coordinated. It must indicate that the activities should be sequenced in a well defined order: the customer asks for a service, the server processes the request, the server provides the result and the customer gets the result.

So, by encapsulating attachments inside connectors and having well defined connector interfaces with previously known element types to be connected by each connector type components and/or configurations are assembled in an easy and coherent way in the form of an architectural puzzle (*Lego Blocks*) without any effort to describe links among components and connectors or between configurations and connectors. Consequently, this approach accelerates the development of component-based systems, improves their evolution, coherence, maintainability and promotes component markets [4].

3.3.2. Connector taxonomy

In C3 we have defined three connector types as illustrated in Figure 3: the connection connector type (CC), the composition decomposition connector type (CDC), and expansion compression connector type (ECC). Each type has its own semantic and has the following signature form:

Connector typeName (*requiredInterf, providedInterf*);

Where *requiredInterf* represents all required ports and services and *providedInterf* represents all provided ports and services of a connector. Obviously each interface also contains services, but in the following definitions we focus only on structural aspect of the interface (ports). The functional aspect (*services*) will not be addressed in this paper and therefore they will not be specified in the descriptions that follow. Consider that each service can use one or more ports of the same interface. In the following we give the exact function of each type of connector in C3 metamodel.

Connection connector (CC)

CC connector type is used to connect components and / or configurations belonging to the same level of decomposition (the same abstraction level) as illustrated by Figure 7.a. The ports of this type of connector can be "*required*" or "*provided*". Thus, through these ports elements can exchange services between them.

Connector CC ({X_i.requiredPort}, {Y_j.providedPort})

where $X_i , Y_j \subset$ {component, configuration},
$X_i , Y_j \subset L_k$; // the same hierarchical level (L_k),
X_i.Level = Y_j.Level, with
i = 1, 2, .., M ; j = 1, 2, .., N, k = 1, 2, .., R.

Where (M+N) is the maximum number of elements which can be linked by CC connector. Hence, CC may have to (M+N) ports. The mapping between the inputs and outputs is described by an exchange protocol called *glue* defined inside of the connector. The various possibilities of links that a connection connector can have are depicted in Figure 7.a.

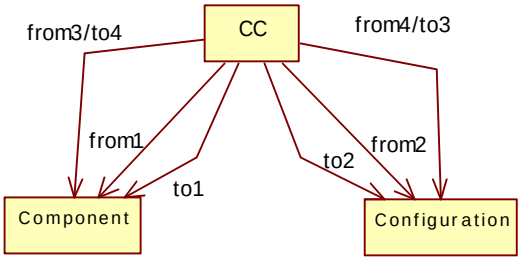


Figure 7.a: Possible links of CC Connector

Figure 7.b represents CC_1 a connection connector type used to link a *client* component with *s-config* configuration of the previous example. This type connector has two ports: portC1 in client side and portS1 in server side. Hence, the interface CC_1 will be defined as follows:

Connector CC$_1$ *(portC1, portS1);*

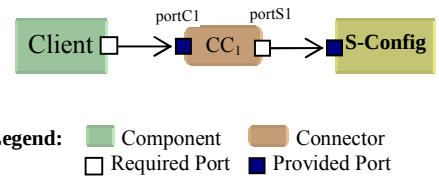


Figure 7.b: Connector CC1 in client-server architecture

Composition / decomposition connector (CDC)

CDC connector type is used to realize a top-down refinement (i.e. to link a configuration with its internal elements) also we call this relationship a decomposition model. Likewise CDC connector can be used to realize bottom-up abstraction (i.e. to link a set of elements to their container or configuration) also we call this relationship a composition model. However, this type of connectors can play two semantic roles with two different glue protocols. The first one is the decomposition process of a configuration and the second one is composition process of a configuration.

- *Decomposition of a configuration X to its internals*

  Connector CDC(X.requiredPort, { Y_i .providedPort});

- *Composition of Y_i elements to constitute a configuration X*

  Connector CDC({Y_i.requiredPort}, X.providedPort);

where X is a configuration,
$Y \subset$ {component, configuration}, i =1,2,..,N ,
$X \subset L_k$ and $Yi \subset L_{k-j}$ (i.e. X.Level > Y_i.Level)
L_k is the hierarchical level.

Thus, a CDC connector will have (N+1) ports, where N is the number of internal elements in the corresponding configuration. This type of connector has the following interests: first it allows us to shape the genealogical tree of the different elements deployed in an architecture, second it enables a configuration to spread information to all these internal elements without exception (to-down propagation) and inversely (i.e. it allows any internal element to send information to its configuration). Therefore, when designing this type of connector we can choose to define the glue corresponding to the decomposition function or that corresponding to the composition function. Also, we can define glue corresponding to the two functions together in the same connector type. Figure 8.a represents the possible links that a CDC connector type may have in a given architecture.

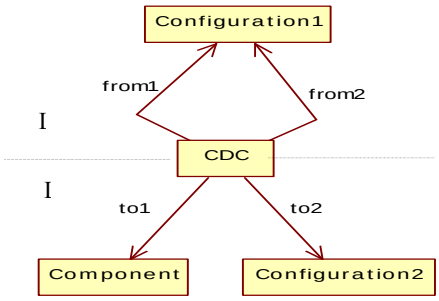


Figure 8.a: Possible links of CDC Connector

Figure 8.b represents CDC1 a decomposition composition connector type used to link client-server configuration (CS-config) defined at the hierarchical level (L_2) with its internals namely client component (Client) and server configuration (s-config) defined at the lower hierarchical level (L_1). Consequently, the interface of CDC1 connector type will be specified as follows:

Connector *CDC_1 (portCS2, portC2, portS2);*

Where portC2, portS2, and portCS are respectively used to connect CDC1 with the client component, the server configuration, and client-server configuration (CS-config).

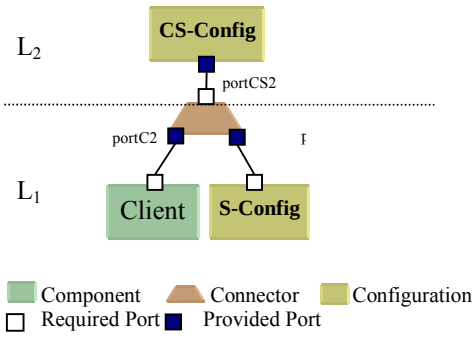


Figure 8.b: Possible links of CDC1 connector

Expansion/compression connector (ECC)

The ECC is used to establish a service link between a configuration and its internal elements. Also, ECC can be used as an expansion operator of services to several sub-services and it can be used in reverse as a compression operator of set of services to a global service. The CDC may have an interface for expansion and another for compression. So, these interfaces are defined as follows:

- *Expansion role*

  Connector ECC (X.requiredPort , { Y_i.providedPort }) ;

- *Compression role*

  Connector ECC ({ Y_i.requiredPort } , X.providedPort) ;

where X is a configuration,
$Y \subset$ {component, configuration},
i =1,2,..,N, and N ≤ number of internal elements.
$X \subset L_k$ et $Y_i \subset L_{k-1}$; (i.e. X.Level > Y_i.Level)
L is the hierarchical level.

ECC connector type can be implemented using either single glue for one function (expansion or compression) or using two separate glues for expansion and compression functions. This will depend on the design decision.

Figure 9.a represents the various possibilities of connections that an ECC connector type can have in a given architecture. So, in this case the configuration $config_0$ contains two components ($comp_1$, $comp_2$) and two configurations ($config_1$, $config_2$) but $config_0$ have only two service relationships with $comp_1$ and $config_1$ and no service relationship with $comp_2$ and $config_2$.

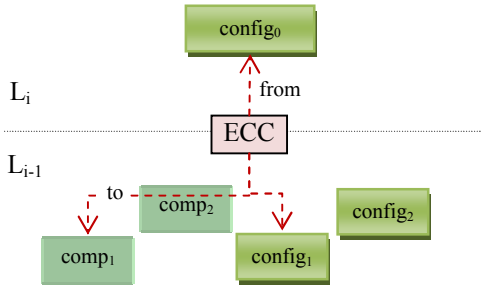


Figure 9.a: Possible links of ECC connector

Figure 9.b illustrates the connector type ECC1 which allows exchange of information between the server configuration (s-config) and the coordinator component (Coor.). Thus, to achieve a bidirectional communication between the server and coordinator, ECC1 must have the following ports:

portS3 and portCo1 are used to ensure the expansion function from the server to coordinator. The portCo2 and portS4 are used to ensure compression function. The interface of this ECC1 type will be as follows:

Connector *ECC1 (portS3, portCo1, portS4, portCo2) ;*

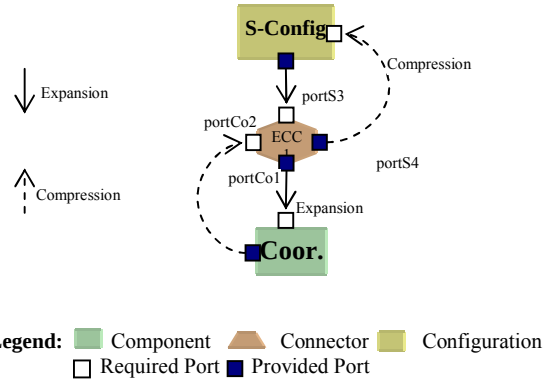


Figure 9.b: Possible links of ECC1 connector in client-server architecture

4 Physical Architecture (PA)

The physical architecture is a memory image of the application instance of the logical architecture. This image is built in the form of a graph whose nodes are instances of a connections manager. Each instance created corresponds to a component or a configuration instanced to construct the real application. Nodes of this graph are connected by arcs. We have three types of arcs. Each type of arc corresponds to specific type of connector. The physical architecture is built to serve as support for updating and evolution operations of the application instance like addition, removal, and replacement of elements in the application instance.

4.1 Connections Manager (CM)

The physical architecture is described using only two levels of abstractions; model or type level and level instance level as illustrated in Figure 9.a. In the type level we have the connections manager type represented by a class that encapsulates all different link of information on the links that a component or a configuration may have with its environment.

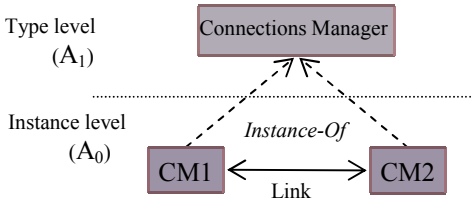


Figure 10.a: Abstraction levels in physical architecture

Each CM is identified by a name and has four attributes as indicated in Figure 10.b.

```
connectorManager Name
{
    ElementName : string ;
    CDC_Link : string ;
    CC_Links : list ;
    ECC_Link : string ;
}
```

Figure 10.b: Structure of a connections manager

- *ElementName:* represents the name of the architectural element associated with this CM (i.e. the name of the component or the configuration corresponding);
- *CC_Links:* list of connection connector names connected to the element associated with this CM;
- *CDC_link:* the name of the composition decomposition connector connected to the element associated with this CM;
- *ECC_Link:* the name of the expansion compression connector connected to the element associated with this CM;

4.2 Operations on Connections Manager

The possible operations on the connections manager are:

- *Instantiation:* the connection manager is instantiated at the instance level (A_0) of the physical architecture. Whenever a configuration or component is instantiated at the application level the associated CM is automatically created in the physical architecture.
- *Installation:* each time a connector is installed at the application level between a set of element instances, so the attributes of the associated CMs are updated with the necessary information about this connector instance.

• *Propagation:* the mechanism of propagation is used to update information about links needed between CMs. These links are published by the interface of the connector installed at the application level.

The physical architecture corresponding to the application instance of client-server architecture is illustrated in Figure 11. In this application we assume having two clients connected to a single server.

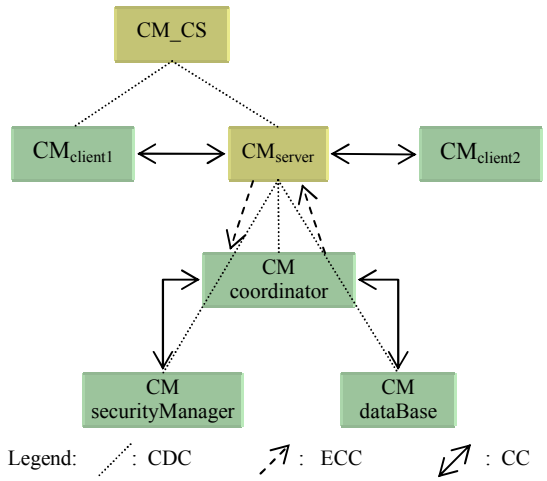
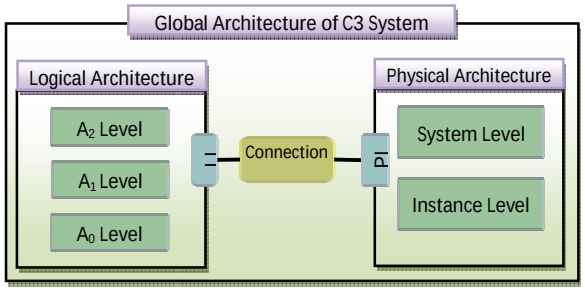


Figure 11: Physical architecture of client-server application .

Once the application is built by the user, the corresponding physical architecture is also built in parallel. Thereafter if we need to intervene on the application to maintain or evolve it we must locate the concerned elements on the physical architecture using graph searching routines and graph updating operations like add (node), delete (node) or replace (node).

Finally we can represent the logical architecture (LA) and the physical architecture (PA) and their relationship by an architecture model described in C3 metamodel. Thus, the LA and the PA are represented by two components and the relationship between them by a connection connector (Figure 12). Any action performed at the LA level causes a sending a message to the PA level. This message will be interpreted as an action to be performed by the PA. Exchanged messages (services) between these two types of architectures are:

- A component instantiation at the LA level causes sending a message "*CM_creation*" from LA interface (LI) to PA. When this message is received by the PA interface (PI) a connection manager instance will be created to represent this component at the PA level.
- A connector instantiation at the LA level causes sending a message "*CM_connection*" from LA to PA. When this message is received by the PA a set of links are created to link CM instances corresponding to all components connected by this connector instance.
- Any updating action at the LA level causes sending a message "*CM_update*" from LI to PI. When this message is received by the PA a set of updating operations are performed to rearrange links among the corresponding CMs.



Legend: LI: logical interface, PI: physical interface

Figure 12: Architectures relationship.

5 Conclusion

In this article we have presented the core elements of C3 metamodel and how to describe software architecture using C3. The elements defined by C3 are assembled through their interfaces to build software architectures. So, we must ensure syntactic checks by checking the compatibility of interfaces types of various elements assembled in the architecture and are in interaction with each other.

Mainly, our approach is defined by two types of architectures. A logical architecture described by the architect. And a physical architecture generated automatically by the system. The logical architecture uses architectural concepts most commonly accepted by all ADLs namely components, connectors and configurations.

We found interesting to give a new structure for connectors in which attachments are encapsulated within the definition of connectors. Hence, the interface connector is now a set of services and ports. This new structure allows us to assemble connectors only with elements that are defined in its interface.

We have defined a set of generic, reusable connectors and extensible to support new structural and behavioural relations among components and we have identified three types of connectors. Connection Connectors (CC) which refer to the links among components belonging to the same level of decomposition. Composition / Decomposition Connectors (CDC) which refer to the links between a configuration and its internal components and connectors. Expansion/Compression connectors (ECC) which refer to the links used to realize any transformation of information or data exchanged between a configuration and its internal components.

Also, we have defined a physical architecture as a graph whose nodes are connections managers associated with architectural elements and arcs represent links that correspond to the connectors. The physical architecture reflects the application architecture which is an instance of the logical architecture and serves as a support for maintenance and evolution operations applied on architecture of the application.

As extension for this work, we planned to define more than one hierarchical view to describe component-based architectures. Among those hierarchies we will use a structural hierarchy to develop the structural aspects of any architecture described according to C3 metamodel, a behaviour hierarchy to make explicit functional aspects of the system, a conceptual hierarchy to clarify the relationships between different elements types developed by the architects and stored in libraries, and metamodeling hierarchy to define the core elements of our C3 metamodel and locate its position in the pyramid of abstraction levels defined by OMG's standards. Obviously, we will focus also on the relationship between these hierarchies, and the different connection mechanisms used to enable interactions between elements from different hierarchy views.

References

[1] Allen R.J. *A Formal Approach to Software Architecture*. PhD thesis, School of Computer Science, Carnegie Mellon University, 1997.

[2] Allen R., and Garlan, D. A Formal Basis for Architectural Connection, *ACM Transactions on Software Engineering and Methodology*. Volume 6, issue 3, pp. 213-249, July 1997.

[3] Allen R., Vestal S., Lewis B., and Cornhill D. Using an architecture description language for quantitative analysis of real-time systems. *In Proceedings of the Third International Workshop on Software and Performance*, ACM Press, Rome, Italy, pp. 203–210, 2002.

[4] Amirat A., Oussalah M., and Khammaci T., Towards an Approach for Building Reliable Architectures. *In Proceeding of IEEE IRI'07*. Las Vegas, Nevada, USA, pp. 467-472, August 2007.

[5] Booch G., Rumbaugh J., and Jacobson I. *The Unified Modeling Language User Guide*. Second Ed., Addison-Wesley Object Technology Series, Addison-Wesley Professional Reading, Massachusetts, 2005.

[6] Canal C., Pimentel E., and Troya J. M. Specification and Refinement of Dynamic Software Architectures. *In Software Architecture*, Kluwer Academic Publishing, pp. 107–126, San Antonio, Texas, February 1999.

[7] Dashofy E., Hoek A.v.d., Taylor R.N. A comprehensive approach for the development of Modular Software Architecture Description Languages. ACM *Transactions on Software Engineering Methodology*. Volume 14, issue 2, pages 199–245, 2005.

[8] Garlan D., Allen R., and Ockerbloom J. Exploiting Style in Architectural Design Environments. *In SIGSOFT'94: Foundations of Software Engineering*. pages 175–188, New Orleans, December 1994.

[9] Garlan D., Monroe R.T., and Wile D. ACME: An Architecture Description Interchange Language. *In Proceedings of the CASCON '97*. IBM Center for Advanced Studies, pages 169–183,Toronto, Ontario, Canada, November, 1997.

[10] Garlan D., Monroe R.T., and Wile D. Acme: Architectural Description Component-Based Systems. *Foundations of Component-Based Systems*. Cambridge University Press, pages 47-68, 2000.

[11] Luckham D.C. Rapide: A Language and Toolset for Simulation of Distributed Systems by Partial Ordering of Events. *In Proceedings of the DIMACS Partial Order Methods Workshop IV*, Princeton University, July, 1996.

[12] Magee J. N., Dulay N., Eisenbach S., and Kramer J. Specifying Distributed Software Architectures *In Proceeding of the Fifth European Software Engineering Conference (ESEC)*. Barcelona, 1995.

[13] Magee J., and Kramer J. Dynamic Structure in Software Architectures. *In Proceedings of ACM SIGSOFT'96: Fourth Symposium on the Foundations of Software Engineering (FSE4)*. pages 3-14, San Francisco, CA, October 1996.

[14] Magee J., Kramer J., and Giannakopoulou D. Behaviour analysis of software architectures. *In Software Architecture*. Kluwer Academic Publishers, pages 35–50, 1999.

[15] Medvidovic N., Oreizy P., Robbins J. E., and Taylor R. N. Using Object-Oriented Typing to Support Architectural Design in the C2 Style. *In ACM SIGSOFT˙S96: Fourth Symposium on the Foundations of Software Engineering (FSE4)*, pages 24–32, San Francisco, 1996.

[16] Medvidovic N., Rosenblum D. S., and Taylor R. N. A Language and Environment for Architecture-Based Software Development and Evolution. *In 21st International Conference on Software Engineering (ICSE'99)*. Los Angeles, May 1999.

[17] Medvidovic N. *Architecture-Based Specification-Time Software Evolution*. PhD Thesis, University of California, Irvine, 1999.

[18] Medvidovic N. and. Taylor R.N. A Classification and Comparison Framework for Software Architecture Description Languages. *IEEE Transactions on Software Engineering*. volume 26, issue 1, January 2000.

[19] Medvidovic N., Dashofy E., and Taylor R.N., Moving Architectural Description from Under the Technology Lamppost. *Information and Software Technology*, volume 49, issue 1, pages 12-31.2007.

[20] Mehta N., Medvidovic N., and Phadke S. Towards a taxonomy of software connectors. *In Proceedings of the 22nd International Conference on Software Engineering*. ACM, New York, pp. 178–187, 2000.

[21] Moriconi M., Qian X., and Riemenschneider R. A. Correct Architecture Refinement. *IEEE Transactions on Software Engineering*. Volume 21, issue 4, pp. 356 – 372, April 1995.

[22] Moriconi M., Riemenschneider R.A., Introduction to SADL 1.0, A Language for Specifying Software Architecture Hierarchies. *Report SRI-CSL-97-01*, 1997.

[23] OMG: *Unified Modeling Language Infrastructure*. from http://www.omg.org/docs/ formal/07-02-06.pdf, 2007.

[24] OMG: *Unified Modeling Superstructure*. from http://www.omg.org/docs/ptc/06-04-02.pdf, 2006.

[25] Ommering R.V., Linden F.V.D., Kramer J., and Magee J. The Koala Component Model for Consumer Electronics Software. *IEEE Computer*. Volume 33, issue 3, pp. 78–85, 2000.

[26] Oquendo F. π-ADL: An Architecture Description Language based on the Higher-Order Typed π-Calculus for Specifying Dynamic and Mobile Software Architectures. *ACM Software Engineering Notes*. Volume 29, issue 3, May 2004.

[27] Oquendo F., Warboys B., Morrison R., Dindeleux R., Gallo F., Garavel H., and Occhipinti C. ArchWARE: Architecting Evolvable Software. *In Software Achitecture (EWSA 2004)*. Volume 3047 of Lecture Notes in Computer Science, pp. 257–271, St Andrews, 2004.

[28] Oussalah M., Smeda A., and Khammaci T. An Explicit Definition of Connectors for Component-Based Software Architecture. *In Proceedings of the 11th IEEE Conference on Engineering of Computer Based Systems (ECBS 2004)*. Brno, Czech Republic, May 24-27, 2004.

[29] Perry D.E. and Wolf A. Foundations for the Study of Software Architectures. *ACM SIGSOFT Software Engineering Notes*. Volume 17, issue 4, pp. 40–52, 1992.

[30] Pinto M., Fluentes L., and Troya M. A Dynamic Component and Aspect-Oriented Platform. *The Computer Journal*. Volume 48, issue 4, pp. 401-420, 2005.

[31] Binns P., Englehart M., Jackson M., and Vestal S. Domain-specific software architectures for guidance, navigation and control. *International Journal of Software Engineering and Knowledge Engineering*. Volume 6, issue 2, pp. 201–227, 1996.

[32] Shaw M. Procedure Calls Are the Assembly Language of System Interconnection: Connectors Deserve First-Class Status. *Lecture Notes in Computer Science*. Volume 1078, pp. 17–32, 1993.

[33] Shaw M., DeLine R., Klein D.V., Ross T. L., Young D. M., and Zalesnik G. Abstractions for Software Architecture and Tools to Support Them. *IEEE Transactions on Software Engineering*. volume 21, issue 4, pp. 314–335, April 1995.

[34] Shaw M., DeLine R., Zelesnik G., Abstractions and Implementations for Architectural Connections. *Proceedings of the 3rd International Conference on Configurable Distributed Systems*. May 1996.

[35] Spitznagel B. and Garlan D., A compositional approach for constructing connectors. *In The Working IEEE/IFIP Conference on Software Architecture (WICSA'01)*. Royal Netherlands Academy of Arts and Sciences Amsterdam, Netherlands. 2001.

[36] Szyperski C. *Component Software: Beyond Object-Oriented Programming*". 2nd Edition, Addison-Wesley, January 2002.

[37] Taylor R. N., Medvidovic N., Anderson K. M., Whitehead JR., Robbins J. E., Nies K. A., Oreizy P., and Dubrow D. L. A component and message-based architectural style for GUI software. *IEEE Transaction on Software Engineering*. Volume 22, issue 6, pp 390–406, June, 1996.

Improvements to a Roll-Back Mechanism for Asynchronous Checkpointing and Recovery

Monika Kapus-Kolar
Department of Communication Systems, Jožef Stefan Institute
Jamova cesta 39, 1000 Ljubljana, Slovenia
E-mail: monika.kapus-kolar@ijs.si

Keywords: asynchronous checkpointing, recovery, maximum consistent state

Received: September 19, 2007

We indicate the existence of a logical flaw in a recently published recovery algorithm for distributed systems and suggest a correction. We also improve the associated communication protocol, the prevention of multiple concurrent instantiations of the algorithm, the handling of obsolete messages and the organization of the stable storing of the relevant data.

Povzetek: Opozarjamo na logično napako v pred kratkim objavljenem algoritmu za okrevanje v porazdeljenih sistemih in predlagamo popravek.

1 Introduction

Gupta, Rahimi and Yang recently proposed a novel recovery algorithm for distributed systems in which checkpoints are taken asynchronously [1]. A checkpoint taken by a process is a snapshot of its local state, stored in a stable storage, so that the process can roll back to it, if this becomes necessary. The start of a process is also one of its checkpoints. Asynchronous checkpointing means that processes take their checkpoints independently.

A failure in a distributed system in principle requires that all its constituent processes roll back, to a global state from which the system can resume its operation as if it had started from it, i.e., to a *globally consistent set of local checkpoints* (GCSLC), ideally to the so called *maximum GCSLC*, in which every local checkpoint is as recent as possible. In the case of asynchronous checkpointing, when a failure occurs, the processes have yet to find the maximum GCSLC. They do that by running a *checkpoint coordination algorithm* (CCA).

Alternatively, the processes might agree to restart from the GCSLC which they currently treat as the starting state of the system, i.e., from the current *recovery line*. This is the most recently computed maximum GCSLC, initially the actual starting state of the system. [1] suggests that the processes occasionally initiate a CCA just for advancing the recovery line.

In this paper, we demonstrate that, because of a subtle logical flaw, the CCA of [1] sometimes returns a checkpoint set which is *not* globally consistent. We correct the flaw and also suggest some other improvements. The rest of the paper is organized as follows. In the next section, we describe the system and the testing of checkpoint consistency and give a brief outline of the CCA of [1]. In Section 3, we explain and correct the flaw. In Section 4, we suggest several other improvements of the CCA. A detailed specification of the improved CCA is given in Section 5. Section 6 suggests that checkpointing and recovery-line advancing in the absence of failures should be more flexible. As the proposed CCA correction increases the usage of the local stable storages, we in Section 7 suggest how to organize them. Section 8 comprises a discussion and conclusions.

2 Preliminaries

2.1 System properties

Of the assumptions explicitly or implicitly stated in [1] for the distributed system considered, the following seem important:

- The set of the constituent processes is a fixed $\{P_j | j \in N\}$ with N a $\{1, \ldots, n\}$. Every process is virtually always aware of the global time.
- From every process P_j to every other process $P_{j'}$, there is virtually a reliable first-in-first-out channel with the worst-case transit delay not exceeding a predefined $T_{j,j'}$. The channels are the only means of inter-process communication. Processes exchange *application messages* (AMs) and *CCA messages* (CMs).
- When a process fails, no other process fails simultaneously and the system does not experience another failure until every process restarts.

2.2 Testing checkpoint consistency

A GCSLC is characterized by all its members being mutually consistent. A checkpoint $C_{j,r}$ of a process P_j is consistent with a checkpoint $C_{j',r'}$ of a process $P_{j'}$ if none of

the processes has recorded a reception of an AM from the partner for which the partner has not recorded a transmission [1].

As channels are reliable queues, it is acceptable that processes record their AM transmissions and receptions simply by counting them, relative to the recovery line [1], for this is where the system virtually started. For a checkpoint $C_{j,r}$ and a process $P_{j'}$, let $S_{j,r,j'}$ and $R_{j,r,j'}$ indicate how many AMs P_j has sent to or, respectively, received from $P_{j'}$, where $S_{j,r,j}$ and $R_{j,r,j}$ are by definition zero.

Suppose that the currently considered checkpoint set is a $\{C_{j,r(j)}|j \in N\}$. To check that it is a GCSLC, one in principle has to check $R_{j,r(j),j'} \leq S_{j',r(j'),j}$ for every two processes P_j and $P_{j'}$. This is what the CCA of [2] does, by letting every process P_j check $\bigwedge_{j' \in (N \setminus \{j\})}(R_{j,r(j),j'} \leq S_{j',r(j'),j})$. The CCA of [1] works under the assumption that this can be done simply by letting every process P_j check $\sum_{j' \in (N \setminus \{j\})} R_{j,r(j),j'} \leq \sum_{j' \in (N \setminus \{j\})} S_{j',r(j'),j}$. In both CCAs, if the adopted test fails for a P_j, the process starts considering an earlier checkpoint, namely the most recent checkpoint $C_{j,r'(j)}$ with $\bigwedge_{j' \in (N \setminus \{j\})}(R_{j,r'(j),j'} \leq S_{j',r(j'),j})$ or $\sum_{j' \in (N \setminus \{j\})} R_{j,r'(j),j'} \leq \sum_{j' \in (N \setminus \{j\})} S_{j',r(j'),j}$, respectively.

2.3 Outline of the algorithm

In the CCA of [1], all communication goes over the initiator of the particular algorithm instance (see Example 1 in the next section). The initiator process repeatedly polls every process, virtually also itself, for the value of the transmission counters of the local candidate for a recovery-line checkpoint. Every such request carries all the information on the transmission counters of the current candidate checkpoints, if any, which the recipient needs for deciding which checkpoint to consider in the next iteration. The first candidate checkpoint of a process is always its most recent checkpoint.

After every process replies, the initiator might detect that the candidate checkpoint set has changed. In that case, it starts another iteration. Otherwise, it broadcasts an indication that a GCSLC has been found. Finally, every process promotes its currently considered checkpoint into its recovery-line checkpoint, from which it, if so requested by the initiator, subsequently restarts. The algorithm covers also the possibility that the initiator requests an immediate restart, from the current recovery line.

3 A flaw and a correction

3.1 The problem of inaccurate information

When a $\{C_{j,r(j)}|j \in N\}$ is checked for being a GCSLC, each $S_{j,r(j),j'}$ may be any natural up to and including the number of the AMs sent from P_j to $P_{j'}$, and each $R_{j,r(j),j'}$ may be any natural up to and including the number of the AMs received at P_j from $P_{j'}$, for remember that the checkpointing is entirely asynchronous. It might, hence, happen that for a process P_j, the test $\sum_{j' \in (N \setminus \{j\})} R_{j,r(j),j'} \leq \sum_{j' \in (N \setminus \{j\})} S_{j',r(j'),j}$ adopted by [1] succeeds in spite of $\bigwedge_{j' \in (N \setminus \{j\})}(R_{j,r(j),j'} \leq S_{j',r(j'),j})$ false, so that P_j fails to detect that $\{C_{j,r(j)}|j \in N\}$ is *not* a GCSLC. If the other processes also fail to detect that the checkpoint set is not consistent, the CCA terminates prematurely and the set unacceptably becomes the new recovery line.

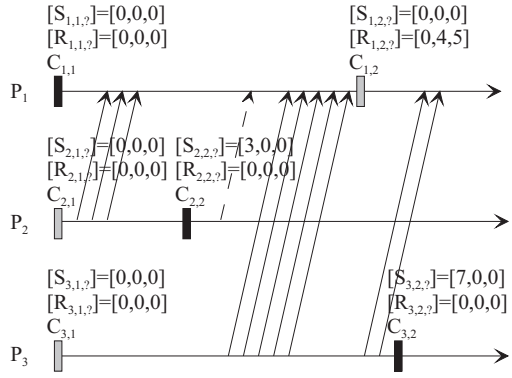


Figure 1: The situation considered in Examples 1–5. The black checkpoints represent the maximum GCSLC. The dashed message is the one making $C_{1,2}$ inconsistent with $C_{2,2}$.

Example 1. *A possible problematic scenario of a system consisting of processes P_1 to P_3 and using the CCA of [1] (see also Fig. 1):*

(1) Upon the start of the system, every process P_j takes a checkpoint $C_{j,1}$, with every $S_{j,1,j'}$ and $R_{j,1,j'}$ zero.

(2) P_2 sends to P_1 three promptly received AMs and then takes a checkpoint $C_{2,2}$, with $S_{2,2,1} = 3$ and $S_{2,2,3} = R_{2,2,1} = R_{2,2,3} = 0$.

(3) P_2 sends to P_1 another promptly received AM and P_3 sends to P_1 five promptly received AMs. P_1 then takes a checkpoint $C_{1,2}$, with $R_{1,2,2} = 4$, $R_{1,2,3} = 5$ and $S_{1,2,2} = S_{1,2,3} = 0$.

(4) P_3 sends to P_1 two more promptly received AMs and then takes a checkpoint $C_{3,2}$, with $S_{3,2,1} = 7$ and $S_{3,2,2} = R_{3,2,1} = R_{3,2,2} = 0$.

(5) P_1 undergoes a failure.

(6) After P_1 recovers, it starts considering its most recent checkpoint $C_{1,2}$ and invites P_2 and P_3 to a new instance of the CCA.

(7) In response, P_2 starts considering its most recent checkpoint $C_{2,2}$ and sends to P_1 a CM carrying $S_{2,2,1}$ and $S_{2,2,3}$, while P_3 starts considering its most recent checkpoint $C_{3,2}$ and sends to P_1 a CM carrying $S_{3,2,1}$ and $S_{3,2,2}$.

(8) When P_1 receives the responses, it sends $(S_{1,2,2} + S_{3,2,2})$ to P_2 and $(S_{1,2,3} + S_{2,2,3})$ to P_3. It also observes $(R_{1,2,2} + R_{1,2,3}) = 9 \leq (S_{2,2,1} + S_{3,2,1}) = 10$ and consequently decides to continue considering $C_{1,2}$, recording the decision by setting the local flag to 0.

(9) Upon receiving $(S_{1,2,2} + S_{3,2,2})$, P_2 *observes* $(R_{2,2,1}+R_{2,2,3}) = 0 \leq (S_{1,2,2}+S_{3,2,2}) = 0$ *and therefore continues considering* $C_{2,2}$*, indicating that to* P_1 *by a CM carrying flag* 0 *and* $S_{2,2,1}$ *and* $S_{2,2,3}$*. Likewise,* P_3 *upon receiving* $(S_{1,2,3} + S_{2,2,3})$ *observes* $(R_{3,2,1} + R_{3,2,2}) = 0 \leq (S_{1,2,3} + S_{2,2,3}) = 0$ *and therefore continues considering* $C_{3,2}$*, indicating that to* P_1 *by a CM carrying flag* 0 *and* $S_{3,2,1}$ *and* $S_{3,2,2}$*.*

(10) After receiving the two indications, P_1*, observing that the flag of every process is* 0*, concludes that a GCSLC has been found, tells* P_2 *and* P_3 *to restart from* $C_{2,2}$ *and* $C_{3,2}$*, respectively, and finally restarts from* $C_{1,2}$*.*

(11) Because of $S_{2,2,1} = 3$ *before the restart,* P_2 *after restarting retransmits the fourth AM to* P_1*. Because of* $R_{1,2,2} = 4$ *before the restart,* P_1 *erroneously interprets the AM as the fifth from* P_2 *and consequently takes an inappropriate action.*

The simplification from [1] described in Section 2.2 is, hence, unacceptable.

3.2 The Necessary Changes of the Algorithm

The indispensable changes to the CCA are the following:

- Where a process P_j originally stores in its stable storage a $\Sigma_{j' \in (N \setminus \{j\})} R_{j,r(j),j'}$, it must actually store $R_{j,r(j),j'}$ of every other process $P_{j'}$.
- Where the initiator process originally sends to a process P_j a $\Sigma_{j' \in (N \setminus \{j\})} S_{j',r(j'),j}$, it must actually send a CM containing $S_{j',r(j'),j}$ of every process $P_{j'}$ other than P_j.
- Where a process P_j originally tests a $\sum_{j' \in (N \setminus \{j\})} R_{j,r(j),j'} \leq \sum_{j' \in (N \setminus \{j\})} S_{j',r(j'),j}$, it must actually test $\bigwedge_{j' \in (N \setminus \{j\})} (R_{j,r(j),j'} \leq S_{j',r(j'),j})$.

Example 2. *If the indispensable CCA corrections are made in Example 1, the scenario after its seventh step takes the following direction (see also Fig. 1):*

(8) After P_1 *receives the responses, it sends a CM carrying* $S_{1,2,2}$ *and* $S_{3,2,2}$ *to* P_2 *and a CM carrying* $S_{1,2,3}$ *and* $S_{2,2,3}$ *to* P_3*. It also observes* $R_{1,2,2} = 4 > S_{2,2,1} = 3$ *and consequently, because* $R_{1,1,2} = 0 \leq S_{2,2,1} = 3$ *and* $R_{1,1,3} = 0 \leq S_{3,2,1} = 7$*, starts considering* $C_{1,1}$*, recording the change of focus by setting the local flag to* 1*.*

(9) Upon receiving $S_{1,2,2}$ *and* $S_{3,2,2}$*,* P_2 *observes* $R_{2,2,1} = 0 \leq S_{1,2,2} = 0$ *and* $R_{2,2,3} = 0 \leq S_{3,2,2} = 0$ *and therefore continues considering* $C_{2,2}$*, indicating that to* P_1 *by a CM carrying flag* 0 *and* $S_{2,2,1}$ *and* $S_{2,2,3}$*. Likewise,* P_3 *upon receiving* $S_{1,2,3}$ *and* $S_{2,2,3}$ *observes* $R_{3,2,1} = 0 \leq S_{1,2,3} = 0$ *and* $R_{3,2,2} = 0 \leq S_{2,2,3} = 0$ *and therefore continues considering* $C_{3,2}$*, indicating that to* P_1 *by a CM carrying flag* 0 *and* $S_{3,2,1}$ *and* $S_{3,2,2}$*.*

(10) After receiving the two indications, P_i*, observing that there is a process with a non-zero flag, sends a CM carrying* $S_{1,1,2}$ *and* $S_{3,2,2}$ *to* P_2 *and a CM carrying* $S_{1,1,3}$ *and* $S_{2,2,3}$ *to* P_3*. It also observes* $R_{1,1,2} = 0 \leq S_{2,2,1} = 3$ *and* $R_{1,1,3} = 0 \leq S_{3,2,1} = 7$ *and consequently decides to continue considering* $C_{1,1}$*, recording the decision by setting the local flag to* 0*.*

(11) Upon receiving $S_{1,1,2}$ *and* $S_{3,2,2}$*,* P_2 *observes* $R_{2,2,1} = 0 \leq S_{1,1,2} = 0$ *and* $R_{2,2,3} = 0 \leq S_{3,2,2} = 0$ *and therefore continues considering* $C_{2,2}$*, indicating that to* P_1 *by a CM carrying flag* 0 *and* $S_{2,2,1}$ *and* $S_{2,2,3}$*. Likewise,* P_3 *upon receiving* $S_{1,1,3}$ *and* $S_{2,2,3}$ *observes* $R_{3,2,1} = 0 \leq S_{1,1,3} = 0$ *and* $R_{3,2,2} = 0 \leq S_{2,2,3} = 0$ *and therefore continues considering* $C_{3,2}$*, indicating that to* P_1 *by a CM carrying flag* 0 *and* $S_{3,2,1}$ *and* $S_{3,2,2}$*.*

(12) After receiving the two indications, P_1*, observing that the flag of every process is* 0*, correctly concludes that a GCSLC has been found, tells* P_2 *and* P_3 *to restart from* $C_{2,2}$ *and* $C_{3,2}$*, respectively, and finally restarts from* $C_{1,1}$*.*

4 Optimization of process communication

If processes exchange transmission counters instead of just their sums, minimization of the number and the size of the exchanged CMs becomes even more important. In the following, we suggest some additional optimizations.

4.1 Early reporting of test results

In a CCA iteration testing a checkpoint set $\{C_{j,r(j)} | j \in N\}$, the initiator process, a P_i, originally (1) transmits every $S_{j,r(j),j'}$ with $j' \notin \{i,j\}$, (2) tests $\bigwedge_{j \in (N \setminus \{i\})} (R_{i,r(i),j} \leq S_{j,r(j),i})$ and (3) receives the expected responses. After the test, the candidate checkpoint of P_i is a $C_{i,r'(i)}$. If it is different from $C_{i,r(i)}$, it was useless to transmit the counters $S_{i,r(i),j'}$. It is, hence, more appropriate that the test comes before the transmissions, so that P_i can instead transmit the counters $S_{i,r'(i),j'}$. With the *early reporting of test results*, the checkpoint which P_i considers in the current iteration is virtually $C_{i,r'(i)}$, which might spare a subsequent iteration.

Example 3. *With early reporting of test results, the scenario fragment in Example 2 simplifies to the following (see also Fig. 1):*

(8) After P_1 *receives the responses, it observes* $R_{1,2,2} = 4 > S_{2,2,1} = 3$ *and consequently starts considering* $C_{1,1}$*. It then sends a CM carrying* $S_{1,1,2}$ *and* $S_{3,2,2}$ *to* P_2 *and a CM carrying* $S_{1,1,3}$ *and* $S_{2,2,3}$ *to* P_3*.*

(9) The same as (11) in Example 2.

(10) After receiving the two indications, P_1*, observing that every flag received was* 0*, correctly concludes that a GCSLC has been found, tells* P_2 *and* P_3 *to restart from* $C_{2,2}$ *and* $C_{3,2}$*, respectively, and finally restarts from* $C_{1,1}$*.*

4.2 Immediate counter reporting

Originally, the CMs inviting processes to another instance of the CCA do not comprise the relevant transmission counters of the initiator. We think that they should, because such *immediate counter reporting* might spare an iteration.

Example 4. *Remember that in Example 3, the first seven steps are as in Example 1. If the process which fails in the fifth step is changed to P_2, the subsequent scenario fragment changes to the following (see also Fig. 1):*

(6a) After P_2 recovers, it starts considering its most recent checkpoint $C_{2,2}$ and invites P_1 and P_3 to a new instance of the CCA.

(7a) In response, P_1 starts considering its most recent checkpoint $C_{1,2}$ and sends to P_2 a CM carrying $S_{1,2,2}$ and $S_{1,2,3}$, while P_3 starts considering its most recent checkpoint $C_{3,2}$ and sends to P_2 a CM carrying $S_{3,2,1}$ and $S_{3,2,2}$.

(8a) After P_2 receives the responses, it observes $R_{2,2,1} = 0 \leq S_{1,2,2} = 0$ and $R_{2,2,3} = 0 \leq S_{3,2,2} = 0$ and therefore continues considering $C_{2,2}$. It then sends a CM carrying $S_{2,2,1}$ and $S_{3,2,1}$ to P_1 and a CM carrying $S_{1,2,3}$ and $S_{2,2,3}$ to P_3.

(9a) Upon receiving $S_{2,2,1}$ and $S_{3,2,1}$, P_1 observes $R_{1,2,2} = 4 > S_{2,2,1} = 3$ and consequently starts considering $C_{1,1}$, indicating that to P_2 by a CM carrying flag 1 and $S_{1,1,2}$ and $S_{1,1,3}$. On the other hand, P_3 upon receiving $S_{1,2,3}$ and $S_{2,2,3}$ observes $R_{3,2,1} = 0 \leq S_{1,2,3} = 0$ and $R_{3,2,2} = 0 \leq S_{2,2,3} = 0$ and therefore continues considering $C_{3,2}$, indicating that to P_2 by a CM carrying flag 0 and $S_{3,2,1}$ and $S_{3,2,2}$.

(10a) After receiving the two indications, P_2, observing that a non-zero flag has been received, observes $R_{2,2,1} = 0 \leq S_{1,1,2} = 0$ and $R_{2,2,3} = 0 \leq S_{3,2,2} = 0$, consequently deciding to continue considering $C_{2,2}$, and then sends a CM carrying $S_{2,2,1}$ and $S_{3,2,1}$ to P_1 and a CM carrying $S_{1,1,3}$ and $S_{2,2,3}$ to P_3.

(11a) Upon receiving $S_{2,2,1}$ and $S_{3,2,1}$, P_1 observes $R_{1,1,2} = 0 \leq S_{2,2,1} = 3$ and $R_{1,1,3} = 0 \leq S_{3,2,1} = 7$ and therefore continues considering $C_{1,1}$, indicating that to P_2 by a CM carrying flag 0 and $S_{1,1,2}$ and $S_{1,1,3}$. Likewise, P_3 upon receiving $S_{1,1,3}$ and $S_{2,2,3}$ observes $R_{3,2,1} = 0 \leq S_{1,1,3} = 0$ and $R_{3,2,2} = 0 \leq S_{2,2,3} = 0$ and therefore continues considering $C_{3,2}$, indicating that to P_2 by a CM carrying flag 0 and $S_{3,2,1}$ and $S_{3,2,2}$.

(12a) After receiving the two indications, P_2, observing that every flag received was 0, correctly concludes that a GCSLC has been found, tells P_1 and P_3 to restart from $C_{1,1}$ and $C_{3,2}$, respectively, and finally restarts from $C_{2,2}$.

If immediate counter reporting is introduced, the scenario fragment simplifies to the following:

(6b) After P_2 recovers, it starts considering $C_{2,2}$, sending to P_1 an invitation carrying $S_{2,2,1}$ and to P_3 an invitation carrying $S_{2,2,3}$.

(7b) In response, P_1, observing that $R_{1,2,2} = 4 > S_{2,2,1} = 3$, starts considering $C_{1,1}$ and sends to P_2 a CM carrying $S_{1,1,2}$ and $S_{1,1,3}$, while P_3, observing that $R_{3,2,2} = 0 \leq S_{2,2,3} = 0$, starts considering its most recent checkpoint $C_{3,2}$ and sends to P_2 a CM carrying $S_{3,2,1}$ and $S_{3,2,2}$.

(8b) After P_2 receives the responses, it observes $R_{2,2,1} = 0 \leq S_{1,1,2} = 0$ and $R_{2,2,3} = 0 \leq S_{3,2,2} = 0$ and therefore continues considering $C_{2,2}$. It then sends a CM carrying $S_{2,2,1}$ and $S_{3,2,1}$ to P_1 and a CM carrying $S_{1,1,3}$ and $S_{2,2,3}$ to P_3.

(9b) The same as (11a).

(10b) The same as (12a).

4.3 Update reporting

Like [1], we assume that the initiator process maintains an $(n \times n)$-array variable V in which every component $V_{j,j'}$ with $j \neq j'$ is during GCSLC construction regularly updated to the value of the $S_{j,r(j),j'}$ belonging to the checkpoint $C_{j,r(j)}$ currently considered by P_j. Every process P_j repeatedly contributes values for the j^{th} row and checks the values in the j^{th} column of V. The search for a GCSLC terminates when V stabilizes.

Suppose that a P_j and a $P_{j'}$ are currently considering checkpoints $C_{j,r(j)}$ and $C_{j',r(j')}$, respectively, and that $R_{j,r(j),j'} \leq S_{j',r(j'),j}$, as required. Then suppose that P_j and $P_{j'}$ at some later point move their attention towards some older checkpoints, a $C_{j,r'(j)}$ and a $C_{j',r'(j')}$, respectively. $R_{j,r'(j),j'} \leq R_{j,r(j),j'}$ implies that the problematic $R_{j,r'(j),j'} > S_{j',r'(j'),j}$ is possible only if $S_{j',r'(j'),j} < S_{j',r(j'),j}$, i.e., if $V_{j',j}$ has changed, for $S_{j',r'(j'),j} > S_{j',r(j'),j}$ is impossible. It, hence, suffices that *updates* to $V_{j',j}$ are reported by $P_{j'}$ to the initiator process, a P_i, and received by P_j from P_i. A CM sent to P_i is then a *zero-flag CM* exactly if it carries *no transmission counters*. In [1], a CM carrying information on V unnecessarily always carries an entire row or column, respectively, and flags are sent explicitly.

Example 5. *If only updates to V are reported, the scenario fragment (8b)-(10b) in Example 4 simplifies to the following (see also Fig. 1):*

(8b) After P_2 receives the responses, it observes $R_{2,2,1} = 0 \leq S_{1,1,2} = 0$ and $R_{2,2,3} = 0 \leq S_{3,2,2} = 0$ and therefore continues considering $C_{2,2}$. It then sends a CM carrying $S_{3,2,1}$ to P_1 and a CM carrying $S_{1,1,3}$ to P_3.

(9b) Upon receiving $S_{3,2,1}$, P_1 observes $R_{1,1,3} = 0 \leq S_{3,2,1} = 7$ and therefore continues considering $C_{1,1}$, indicating that to P_2 by a CM carrying no transmission counters. Likewise, P_3 upon receiving $S_{1,1,3}$ observes $R_{3,2,1} = 0 \leq S_{1,1,3} = 0$ and therefore continues considering $C_{3,2}$, indicating that to P_2 by a CM carrying no transmission counters.

(10b) After receiving the two indications, P_2, observing that no transmission counters have been received, correctly concludes that a GCSLC has been found, tells P_1 and P_3 to restart from $C_{1,1}$ and $C_{3,2}$, respectively, and finally restarts from $C_{2,2}$.

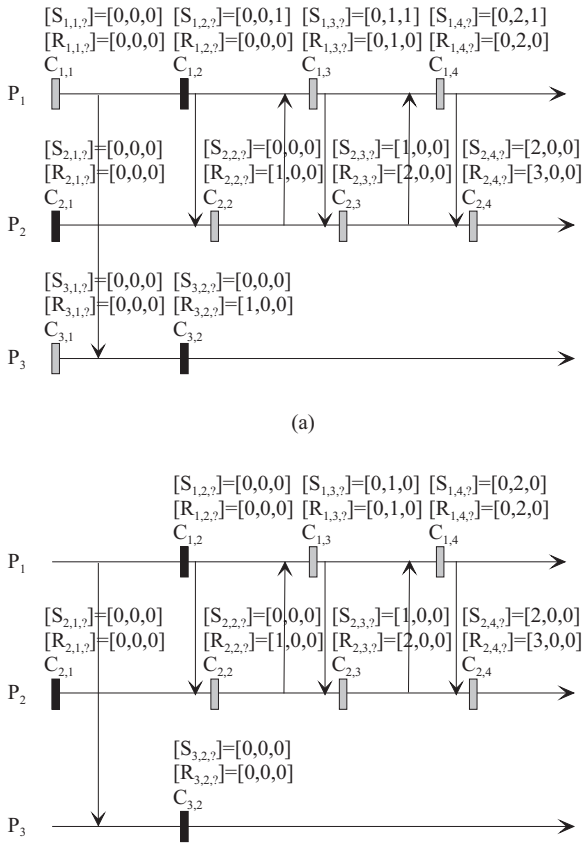


Figure 2: The situation considered in Example 6 (a) before and (b) after the recovery line is advanced to the black checkpoints, which represent the maximum GCSLC.

4.4 Selective polling

If a CM which the initiator process, a P_i, is originally supposed to send to a process P_j is the j^{th} column of V and P_i discovers that P_j already has a faithful copy of the column, *the CM may be omitted*, for even if received, P_j would continue considering the same checkpoint and produce no update for the j^{th} row of V. Moreover, if P_j does not receive the CM, it will not attempt to send a response, meaning that it will *skip an entire iteration* (see Example 6, step 13). Besides, if P_i discovers that all the other processes may skip the current iteration, it may *immediately indicate CCA termination* (see Example 6, step 15).

Example 6. *With the above optimizations, the following scenario is possible in a system consisting of processes P_1 to P_3 (see also Fig. 2):*

(1) Upon the start of the system, every process P_j takes a checkpoint $C_{j,1}$, with every $S_{j,1,j'}$ and $R_{j,1,j'}$ zero.

(2) P_1 sends an AM to P_3 and takes a checkpoint $C_{1,2}$, with $S_{1,2,3} = 1$ and $S_{1,2,2} = R_{1,2,2} = R_{1,2,3} = 0$.

(3) P_3 receives the AM and takes a checkpoint $C_{3,2}$, with $R_{3,2,1} = 1$ and $S_{3,2,1} = S_{3,2,2} = R_{3,2,2} = 0$.

(4) P_1 sends an AM to P_2. P_2 receives the AM and takes a checkpoint $C_{2,2}$, with $R_{2,2,1} = 1$ and $S_{2,2,1} = S_{2,2,3} = R_{2,2,3} = 0$.

(5) P_2 sends an AM to P_1. P_1 receives the AM and takes a checkpoint $C_{1,3}$, with $S_{1,3,2} = S_{1,3,3} = R_{1,3,2} = 1$ and $R_{1,3,3} = 0$.

(6) P_1 sends an AM to P_2. P_2 receives the AM and takes a checkpoint $C_{2,3}$, with $S_{2,3,1} = 1$, $R_{2,3,1} = 2$ and $S_{2,3,3} = R_{2,3,3} = 0$.

(7) P_2 sends an AM to P_1. P_1 receives the AM and takes a checkpoint $C_{1,4}$, with $S_{1,4,2} = R_{1,4,2} = 2$, $S_{1,4,3} = 1$ and $R_{1,4,3} = 0$.

(8) P_1 sends an AM to P_2. P_2 receives the AM and takes a checkpoint $C_{2,4}$, with $S_{2,4,1} = 2$, $R_{2,4,1} = 3$ and $S_{2,4,3} = R_{2,4,3} = 0$.

(9) P_2 decides to initiate the CCA just for advancing the recovery line. So it starts considering $C_{2,4}$, sending to P_1 an invitation carrying $S_{2,4,1}$ and to P_3 an invitation carrying $S_{2,4,3}$.

(10) In response, P_1, observing that $R_{1,4,2} = 2 \leq S_{2,4,1} = 2$, starts considering $C_{1,4}$ and sends to P_2 a CM carrying $S_{1,4,2}$ and $S_{1,4,3}$, while P_3, observing that $R_{3,2,2} = 0 \leq S_{2,4,3} = 0$, starts considering $C_{3,2}$ and sends to P_2 a CM carrying $S_{3,2,1}$ and $S_{3,2,2}$.

(11) After P_2 receives the responses, it observes $R_{2,4,1} = 3 > S_{1,4,2} = 2$ and therefore starts considering $C_{2,3}$. It then sends a CM carrying $S_{2,3,1}$ and $S_{3,2,1}$ to P_1 and a CM carrying $S_{1,4,3}$ to P_3.

(12) Upon receiving $S_{2,3,1}$ and $S_{3,2,1}$, P_1, observing that $R_{1,4,2} = 2 > S_{2,3,1} = 1$, starts considering $C_{1,3}$, indicating that to P_2 by a CM carrying $S_{1,3,2}$. On the other hand, P_3 upon receiving $S_{1,4,3}$ continues considering $C_{3,2}$, indicating that to P_2 by a CM carrying no transmission counters.

(13) After receiving the two indications, P_2, observing that $R_{2,3,1} = 2 > S_{1,3,2} = 1$, starts considering $C_{2,2}$, sending to P_1 a CM carrying $S_{2,2,1}$. P_3 is not involved in the iteration, *because $S_{1,3,3} = S_{1,4,3}$ and $S_{2,2,3} = S_{2,3,3}$.*

(14) Upon receiving $S_{2,2,1}$, P_1, observing that $R_{1,3,2} = 1 > S_{2,2,1} = 0$, starts considering $C_{1,2}$, indicating that to P_2 by a CM carrying $S_{1,2,2}$.

(15) After receiving the indication, P_2, observing that $R_{2,2,1} = 1 > S_{1,2,2} = 0$, starts considering $C_{2,1}$. It then, because $S_{1,2,3} = S_{1,3,3}$ and $S_{2,1,1} = S_{2,2,1}$ and $S_{2,1,3} = S_{2,2,3}$, immediately *concludes that a GCSLC has been found, indicates that to P_1 and P_3 and finally makes $C_{2,1}$ its recovery-line checkpoint.*

(16) Upon receiving the indication, P_1 makes $C_{1,2}$ its recovery-line checkpoint, meaning that it deletes $C_{1,1}$ and subtracts $S_{1,2,3}$ from $S_{1,2,3}$, $S_{1,3,3}$ and $S_{1,4,3}$. Likewise, P_2 makes $C_{3,2}$ its recovery-line checkpoint, meaning that it deletes $C_{3,1}$ and subtracts $R_{3,2,1}$ from $R_{3,2,1}$. The resulting situation is given in Fig. 2(b).

5 Details of the algorithm

5.1 Introduction

For a CCA instance I, let $P_{In(I)}$ denote its initiator process, Adv_I the predicate telling whether it starts by an attempt to advance the recovery line, and Rst_I the predicate telling whether it terminates by a system restart, i.e., whether it is an instance initiated upon a failure. We assume $Adv_I \vee Rst_I$, so that I has a purpose.

Every CM of a CCA instance I is either an *invitation CM* (ICM) or an *ordinary CM* (OCM). If it is sent by $P_{In(I)}$ and carries no transmission counters, it is also a *termination CM* (TCM).

In Sections 5.2–5.4, we specify a CCA instance, an I, as if it runs in isolation. The given CCA is essentially the CCA of [1] corrected and slightly modified as proposed in Sections 3 and 4. In Section 5.5, we, unlike [1], specify also how processes are assumed to react if they during the execution of a specific CCA instance receive an AM or a CM belonging to another CCA instance.

5.2 Behaviour of the initiator

In a CCA instance I, $P_{In(I)}$, having decided on Adv_I and Rst_I, executes the following sequence of steps, starting at a time $t^1_{I,In(I)}$ and broadcasting a TCM at a time t^2_I:

(1) If $\neg Adv_I$, it broadcasts an ICM carrying just Adv_I (and thereby implicitly also Rst_I) and the current time, i.e. a TCM carrying $t^1_{I,In(I)}$ and t^2_I, and then goes directly to the procedure specified in Section 5.4. Otherwise, it proceeds as follows.

(2) It sets CC, the variable denoting its currently considered checkpoint, to its most recent checkpoint, a $C_{In(I),r}$, and initializes in its $(n \times n)$-array variable V every $V_{In(I),j}$ to $S_{In(I),r,j}$, every $V_{j,j}$ to zero and every other $V_{j,j'}$ to a value so large that no transmission or reception counter can ever acquire it.

(3) To every other process P_j, it sends, and records that by setting a Boolean variable Q_j to true, an ICM carrying Adv_I, Rst_I, $V_{In(I),j}$ and in the case of Rst_I also $t^1_{I,In(I)}$.

(4) It makes a copy V' of V.

(5) From every other process P_j with Q_j true, it receives, and records that by setting Q_j to false, an OCM carrying at least Rst_I and then changes every $V_{j,j'}$ for which the CM carries a value to the value received.

(6) It sets CC to its most recent checkpoint $C_{In(I),r}$ with $R_{In(I),r,j} \leq V_{j,In(I)}$ for every process P_j and sets every $V_{In(I),j}$ to $S_{In(I),r,j}$.

(7) If for every other process P_j, every $V_{j',j}$ is the same as $V'_{j',j}$, it broadcasts an OCM carrying just Rst_I and the current time, i.e. a TCM carrying t^2_I, and then goes directly to the procedure specified in Section 5.4. Otherwise, it proceeds as follows.

(8) To every other process P_j with a $V_{j',j}$ different from $V'_{j',j}$, it sends, and records that by setting Q_j to true, an OCM carrying Rst_I and every such $V_{j',j}$.

(9) It iterates to the step (4).

5.3 Behaviour of the other processes

In a CCA instance I, a process P_j other than $P_{In(I)}$ for every variable $V_{j,j'}$ and $V_{j',j}$ of $P_{In(I)}$ maintains a copy with the same name, executing the following sequence of steps:

(1) It receives, at a time $t^1_{I,j}$, from $P_{In(I)}$ an ICM and sets Adv_I, Rst_I, $t^1_{I,In(I)}$, t^2_I and $V_{In(I),j}$ to the value received, if any. We assume that $(t^1_{I,j} - t^1_{I,In(I)})$ does not exceed a predefined $T^1_{In(I),j}$. An appropriate value for a $T^1_{j',j}$ with $j' \neq j$ would typically be slightly over $T_{j',j}$, while every $T^1_{j,j}$ is by definition zero.

(2) If $\neg Adv_I$, it goes directly to the procedure specified in Section 5.4. Otherwise, it proceeds as follows.

(3) If possible and desired, it takes a fresh checkpoint.

(4) It sets $V_{j,j}$ to zero and every $V_{j',j}$ with $j' \notin \{In(I), j\}$ and $V_{j,j'}$ with $j \neq j'$ to a value so large that no transmission or reception counter can ever acquire it.

(5) It sets its variable CC to its most recent checkpoint $C_{j,r}$ with $R_{j,r,j'} \leq V_{j',j}$ for every process $P_{j'}$, sends to $P_{In(I)}$ an OCM carrying Rst_I and, as a new value for $V_{j,j'}$, every $S_{j,r,j'}$ different from $V_{j,j'}$, and then sets every $V_{j,j'}$ to $S_{j,r,j'}$.

(6) It receives from $P_{In(I)}$ an OCM carrying at least Rst_I and changes every $V_{j',j}$ for which the CM carries a value to the value received. If the CM was a TM, it sets t^2_I to the value received and goes directly to the procedure specified in Section 5.4. Otherwise, it iterates to the step (5).

The time when a process P_j gets involved into a CCA instance I, hence, does not exceed a $t^3_{I,j}$ defined as $min\{(t^1_{I,In(I)} + T^1_{In(I),j}), t^2_I\}$ in the case of Adv_I and as $(t^1_{I,In(I)} + T^1_{In(I),j})$ otherwise.

5.4 Moving the recovery line and/or restarting

Here are the steps of the procedure which in a CCA instance I, every process P_j, with its CC in the case of Adv_I a $C_{j,r}$, executes at the end of the CCA, terminating at a time $t^4_{I,j}$:

(1) If Adv_I, it deletes from its storage every checkpoint older than $C_{j,r}$.

(2) If Rst_I, it deletes from its storage every checkpoint except the oldest one.

(3) For every preserved checkpoint $C_{j,r'}$, it decreases every $S_{j,r',j'}$ by $S_{j,r,j'}$ and every $R_{j,r',j'}$ by $R_{j,r,j'}$.

(4) If Rst_I, it restarts from its oldest preserved checkpoint.

We assume that $(t^4_{I,j} - t^2_I)$ does not exceed a predefined $T^2_{In(I),j}$. An appropriate value for a $T^2_{j',j}$ would typically be slightly over $T_{j',j}$.

5.5 Handling of unexpected messages

If a process waiting for an OCM of a CCA instance I instead receives an AM, it freely decides whether to process it concurrently to or after I.

If a process P_j waiting for an OCM of a CCA instance I instead receives a CM of another CCA instance, an I', it reacts as follows: An ICM with $Rst_{I'}$ false or an OCM with $Rst_{I'} \neq Rst_I$ is just discarded. Upon an ICM with $Rst_{I'}$ true, P_j abandons I and starts executing I' instead. An OCM with $Rst_{I'} = Rst_I$ is erroneously recognized as an OCM of I, so such situation must be prevented (see Section 6.3).

If a process P_j currently involved in no CCA instance receives an OCM or an ICM for which it suspects that it belongs to an obsolete CCA instance, it just discards it. An ICM of a CCA instance I', received at a time t from a process $P_{j'}$, is recognized as obsolete if $\neg Rst_{I'}$ and P_j has participated in a CCA instance I with Rst_I true and $(t^3_{I,j'} + T_{j',j}) \geq t$. This is because the estimated worst-case scenario for I overriding I' is that $P_{j'}$ sends the ICM just before it gets involved into I at the time $t^3_{I,j'}$ and then the ICM reaches P_j with the delay $T_{j',j}$.

6 Optimizing algorithm activation

6.1 Introduction

[1] suggests that in the absence of failures, the CCA is initiated periodically, with the period very long, so that one can assume that when a new CCA instance starts, the previous one, if any, has long been completed. More precisely, [1] specifies that the period is much longer than the checkpointing period of any individual process. [1] also suggests that processes initiate the CCA in turns, so that they never attempt to initiate it concurrently. To implement the policy, [1] has all system processes organized in a *virtual ring* along which the right to initiate the CCA is passed as a *token*, merely by the passing of time. Every reception of the token is interpreted as an obligation to initiate the CCA immediately.

In [1], the frequency of token passing is exactly the desired frequency of running the CCA in the absence of failures. We find this scheme too rigid. On the one hand, forcing processes to take a checkpoint or initiate a recovery-line advancement periodically is seldom optimal. Processes should rather wait for a substantial reason. On the other hand, the initiator token should circulate as fast as possible, so that any process wanting it receives it quickly. Therefore, *we drop the periodicity assumption for checkpointing and recovery-line advancing, and keep the virtual ring just as a means of concurrency control, giving it a more appropriate timing*.

6.2 Reasons for checkpointing and recovery-line advancing

For taking a checkpoint, a typical reason would be that the process has since its last checkpoint accomplished a lot of work or exchanged a lot of AMs.

For advancing the recovery line, we see three reasons:

- A process is running out of stable storage and therefore wants that some of the stored checkpoints become obsolete (remember Section 5.4, step 1).
- A reception or a transmission counter of a process is approaching overflow and the process therefore wants more events to become obsolete for counting (remember Section 5.4, step 3).
- A process wants to reduce the discrepancy between its current state and its recovery-line checkpoint, so that, if a failure occurs, its roll-back in the worst case would not be so drastic.

How often such a reason occurs, depends not only on static parameters, such as the size of the local stable storages and the speed of the processes and the channels, but also on the demands of the executed application, which tend to vary with time.

6.3 Prevention of concurrent instantiations

At any moment, the initiator token is virtually either in transit or resides with one of the processes. If a process P_j possesses the token at a time t, the next time when another process $P_{j'}$ possesses it should ideally be soon, but not until $P_{j'}$ has had a chance to detect whether P_j has initiated the CCA at time t, i.e., not until after $(t + T^1_{j,j'})$. The delay is necessary because when initiating a CCA instance I with $\neg Rst_I$, the process $P_{In(I)}$ must be aware of the most recently initiated previous CCA instance, an I', if any, so that it can, by suitably delaying I, secure that the following is satisfied at the time $t^1_{I,In(I)}$:

- $t^1_{I,In(I)} > t^4_{I',In(I)}$ and $t^1_{I,In(I)} > t^2_{I'} + T^2_{In(I'),j}$ for every other process P_j, so that every process has already terminated execution of I'.
- If $Rst_{I'}$, $t^1_{I,In(I)} > t^3_{I',j} + T_{j,j'}$ for every two processes P_j and $P_{j'}$, so that (1) the CMs which I' made obsolete, if any, have already been received and (2) the ICM of I will not be discarded upon reception (remember Section 5.5).

It might happen that while a process is waiting for an opportunity to initiate a CCA instance I with $\neg Rst_I$, another process initiates a CCA instance I' with $Adv_{I'}$. Upon detecting that, $P_{In(I)}$ should drop its pursuit, for the task of advancing the recovery line is already being taken care of.

7 Organization of the storage

When a process gets involved into a CCA instance, it has to access information on its previous checkpoints. [1] suggests that a process maintains this information primarily in its working memory, so that any access to it can be quick. However, a process having just recovered from a failure can safely rely only on the copy of the information which it has made in its stable storage. How efficiently individual parts of the copy can be accessed, strongly depends on the organization of the storage, which, hence, deserves some discussion.

Obviously, each checkpoint $C_{j,r}$ requires that process P_j stores every $R_{j,r,j'}$ with $j \neq j'$ individually, and not just $\sum_{j' \in (N \setminus \{j\})} R_{j,r,j'}$, as in [1]. For each $C_{j,r}$, P_j would, hence, store vectors $\vec{S}_{j,r}$ and $\vec{R}_{j,r}$ of size $(n-1)$.

If we imitated the policy of [1], every process P_j would for every checkpoint $C_{j,r}$ store in the stable storage the list of all $\vec{R}_{j,r'}$ (originally of all $\sum_{j' \in (N \setminus \{j\})} R_{j,r',j'}$) belonging to a $C_{j,r'}$ not later than $C_{j,r}$, so that, when deciding how far back to move from $C_{j,r}$, P_j could fetch all the necessary information in a single access to the storage. However, this would mean that each element of such a list is stored several times, first in the list of the checkpoint to which it belongs and then in the list of every subsequent checkpoint. With list elements vectors of size $(n-1)$, i.e. not of size 1 as expected in [1], this would be unacceptable for large n.

We think that P_j should maintain a single list, for every $C_{j,r}$ comprising a triplet consisting of $\vec{R}_{j,r}$, $\vec{S}_{j,r}$ and a pointer to all the other details which P_j might need if it actually restarts from $C_{j,r}$. As n is fixed, so can be the size of the triplets, meaning that such a list can be easily maintained and accessed as virtually a single block of consecutive storage locations.

Alternatively, P_j could maintain three lists, one for each kind of the items in the triplets. With this approach, P_j could fetch the stable data more selectively. With all the lists consisting of fixed-sized elements, finding their corresponding elements would still be trivial.

Let us also note that fetching the entire R list in a single step might not always be a good idea, as this might be really a lot of data. In other words, when optimizing communication between processes and their stable storage, one is usually forced to make compromises between the number and the size of the messages, where the optimal strategy depends on the specifics of the system and the current circumstances.

It might happen that a CCA instance I does not advance the recovery line in spite of Adv_I true. In such a case, a process wanting to take fresh checkpoints in spite of running out of stable storage may start deleting its earlier checkpoints, with the only constraint that it must never delete its recovery-line checkpoint or its current candidate for a new recovery-line checkpoint. As the starting checkpoint of any process is stored implicitly, processes can survive even without a stable storage for checkpoints [2], because further checkpoints only increase the probability that in the case of a restart, the required roll-back will not be too drastic.

8 Discussion and conclusions

8.1 Contributions of the paper

We have proposed the following improvements to the CCA of [1], its activation and its storage management:

- A logical flaw has been identified and corrected.
- Instead of exchanging entire rows or columns of the array V, processes now exchange only their updates.
- Whenever possible, processes now skip individual iterations of the algorithm.
- It can no longer happen that the algorithm executes a superfluous iteration.
- Obsolete CMs are now properly recognized and ignored.
- By recognizing the policy of recovery-line advancement and the prevention of concurrent instantiations as two separate issues, we have been able to make the former more flexible and the latter, through faster token circulation, less restrictive. The assumption that processes take their checkpoints periodically has also become unnecessary.
- The organization of the stable storage no longer includes multiple copies of reception counters.
- Process are allowed to replace their old checkpoints with fresh ones.

8.2 Correctness and performance of the algorithm

The proposed CCA is essentially that of [1], except that we properly increased the rigour of checkpoint comparison, in every CM added the missing and removed the redundant information, and removed the CMs which consequently lost every purpose. Hence, if the original CCA is correct up to the rigour of checkpoint comparison, our CCA is also correct.

In [1], it is demonstrated that, for the adopted system topology, the there proposed CCA is an improvement over the CCA of [3], in that the number of checkpoint comparisons grows, linearly, much slower with the average number of checkpoints per process, and linearly instead of quadratically with the number of processes. It is also demonstrated that the proposed CCA is an improvement over the CCA of [2], in that the number of the exchanged CMs grows linearly instead of quadratically with the number of processes.

If the checkpoint comparisons in the CCA of [1] are made sufficiently rigorous, the algorithm preserves all the above advantages. With the proposed additional optimizations of process communication, the number of checkpoint comparisons and the number of the exchanged CMs are sometimes even lower, because the redundant iterations are skipped and because the redundant requests for information on transmission counters are removed. Besides, the remaining CMs are sometimes shorter, because processes exchange just updates to V.

8.3 Concluding remarks

For a distributed algorithm, it is equally important to decide on the task which the process should perform in cooperation, on the protocol securing the necessary coordination and on the management of concurrent instances of the algorithm. For the CCA of [1], we proposed a correction of the task and several improvements to the protocol and the concurrency management. For further work, we propose quantitative assessment of the expected benefits of the additional optimizations.

References

[1] Gupta B., Rahimi S., Yang Y. (2007) A novel roll-back mechanism for performance enhancement of asynchronous checkpointing and recovery, *Informatica*, 31(1), pp. 1–13.

[2] Juang T., Venkatesan, S. (1991) Crash recovery with little overhead, *Proceedings of the 11th International Conference on Distributed Computing Systems*, IEEE, Arlington, Texas, pp. 454–461.

[3] Ohara M., Arai M., Fukumoto S., Iwasaki K. (2004) Finding a recovery line in uncoordinated checkpointing, *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops*, IEEE, Hachioji, Tokyo, Japan, pp. 628–633.

CONTENTS OF *Informatica* Volume 33 (2009) pp. 1–523

Papers

Editorials

JOŽEF STEFAN INSTITUTE

Jožef Stefan (1835-1893) was one of the most prominent physicists of the 19th century. Born to Slovene parents, he obtained his Ph.D. at Vienna University, where he was later Director of the Physics Institute, Vice-President of the Vienna Academy of Sciences and a member of several scientific institutions in Europe. Stefan explored many areas in hydrodynamics, optics, acoustics, electricity, magnetism and the kinetic theory of gases. Among other things, he originated the law that the total radiation from a black body is proportional to the 4th power of its absolute temperature, known as the Stefan–Boltzmann law.

The Jožef Stefan Institute (JSI) is the leading independent scientific research institution in Slovenia, covering a broad spectrum of fundamental and applied research in the fields of physics, chemistry and biochemistry, electronics and information science, nuclear science technology, energy research and environmental science.

The Jožef Stefan Institute (JSI) is a research organisation for pure and applied research in the natural sciences and technology. Both are closely interconnected in research departments composed of different task teams. Emphasis in basic research is given to the development and education of young scientists, while applied research and development serve for the transfer of advanced knowledge, contributing to the development of the national economy and society in general.

At present the Institute, with a total of about 800 staff, has 600 researchers, about 250 of whom are postgraduates, nearly 400 of whom have doctorates (Ph.D.), and around 200 of whom have permanent professorships or temporary teaching assignments at the Universities.

In view of its activities and status, the JSI plays the role of a national institute, complementing the role of the universities and bridging the gap between basic science and applications.

Research at the JSI includes the following major fields: physics; chemistry; electronics, informatics and computer sciences; biochemistry; ecology; reactor technology; applied mathematics. Most of the activities are more or less closely connected to information sciences, in particular computer sciences, artificial intelligence, language and speech technologies, computer-aided design, computer architectures, biocybernetics and robotics, computer automation and control, professional electronics, digital communications and networks, and applied mathematics.

The Institute is located in Ljubljana, the capital of the independent state of S**love**nia (or S♡nia). The capital today is considered a crossroad between East, West and Mediterranean Europe, offering excellent productive capabilities and solid business opportunities, with strong international connections. Ljubljana is connected to important centers such as Prague, Budapest, Vienna, Zagreb, Milan, Rome, Monaco, Nice, Bern and Munich, all within a radius of 600 km.

From the Jožef Stefan Institute, the Technology park "Ljubljana" has been proposed as part of the national strategy for technological development to foster synergies between research and industry, to promote joint ventures between university bodies, research institutes and innovative industry, to act as an incubator for high-tech initiatives and to accelerate the development cycle of innovative products.

Part of the Institute was reorganized into several high-tech units supported by and connected within the Technology park at the Jožef Stefan Institute, established as the beginning of a regional Technology park "Ljubljana". The project was developed at a particularly historical moment, characterized by the process of state reorganisation, privatisation and private initiative. The national Technology Park is a shareholding company hosting an independent venture-capital institution.

The promoters and operational entities of the project are the Republic of Slovenia, Ministry of Higher Education, Science and Technology and the Jožef Stefan Institute. The framework of the operation also includes the University of Ljubljana, the National Institute of Chemistry, the Institute for Electronics and Vacuum Technology and the Institute for Materials and Construction Research among others. In addition, the project is supported by the Ministry of the Economy, the National Chamber of Economy and the City of Ljubljana.

Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
Tel.:+386 1 4773 900, Fax.:+386 1 251 93 85
WWW: http://www.ijs.si
E-mail: matjaz.gams@ijs.si
Public relations: Polona Strnad

INFORMATICA

AN INTERNATIONAL JOURNAL OF COMPUTING AND INFORMATICS

INVITATION, COOPERATION

Submissions and Refereeing

Please submit an email with the manuscript to one of the editors from the Editorial Board or to the Managing Editor. At least two referees outside the author's country will examine it, and they are invited to make as many remarks as possible from typing errors to global philosophical disagreements. The chosen editor will send the author the obtained reviews. If the paper is accepted, the editor will also send an email to the managing editor. The executive board will inform the author that the paper has been accepted, and the author will send the paper to the managing editor. The paper will be published within one year of receipt of email with the text in Informatica MS Word format or Informatica LaTeX format and figures in .eps format. Style and examples of papers can be obtained from http://www.informatica.si. Opinions, news, calls for conferences, calls for papers, etc. should be sent directly to the managing editor.

QUESTIONNAIRE

☐ Send Informatica free of charge

☐ Yes, we subscribe

Please, complete the order form and send it to Dr. Drago Torkar, Informatica, Institut Jožef Stefan, Jamova 39, 1000 Ljubljana, Slovenia. E-mail: drago.torkar@ijs.si

Since 1977, Informatica has been a major Slovenian scientific journal of computing and informatics, including telecommunications, automation and other related areas. In its 16th year (more than sixteen years ago) it became truly international, although it still remains connected to Central Europe. The basic aim of Informatica is to impose intellectual values (science, engineering) in a distributed organisation.

Informatica is a journal primarily covering the European computer science and informatics community - scientific and educational as well as technical, commercial and industrial. Its basic aim is to enhance communications between different European structures on the basis of equal rights and international refereeing. It publishes scientific papers accepted by at least two referees outside the author's country. In addition, it contains information about conferences, opinions, critical examinations of existing publications and news. Finally, major practical achievements and innovations in the computer and information industry are presented through commercial publications as well as through independent evaluations.

Editing and refereeing are distributed. Each editor can conduct the refereeing process by appointing two new referees or referees from the Board of Referees or Editorial Board. Referees should not be from the author's country. If new referees are appointed, their names will appear in the Refereeing Board.

Informatica is free of charge for major scientific, educational and governmental institutions. Others should subscribe (see the last page of Informatica).

ORDER FORM – INFORMATICA

Name: .....

Title and Profession (optional): .....

.....

Home Address and Telephone (optional): .....

.....

Office Address and Telephone (optional): .....

.....

E-mail Address (optional): .....

Signature and Date: .....

Informatica WWW:

http://www.informatica.si/

Referees:

Witold Abramowicz, David Abramson, Adel Adi, Kenneth Aizawa, Suad Alagić, Mohamad Alam, Dia Ali, Alan Aliu, Richard Amoroso, John Anderson, Hans-Jurgen Appelrath, Iván Araujo, Vladimir Bajič, Michel Barbeau, Grzegorz Bartoszewicz, Catriel Beeri, Daniel Beech, Fevzi Belli, Simon Beloglavec, Sondes Bennasri, Francesco Bergadano, Istvan Berkeley, Azer Bestavros, Andraž Bežek, Balaji Bharadwaj, Ralph Bisland, Jacek Blazewicz, Laszlo Boeszoermenyi, Damjan Bojadžijev, Jeff Bone, Ivan Bratko, Pavel Brazdil, Bostjan Brumen, Jerzy Brzezinski, Marian Bubak, Davide Bugali, Troy Bull, Sabin Corneliu Buraga, Leslie Burkholder, Frada Burstein, Wojciech Buszkowski, Rajkumar Bvyya, Giacomo Cabri, Netiva Caftori, Particia Carando, Robert Cattral, Jason Ceddia, Ryszard Choras, Wojciech Cellary, Wojciech Chybowski, Andrzej Ciepielewski, Vic Ciesielski, Mel Ó Cinnéide, David Cliff, Maria Cobb, Jean-Pierre Corriveau, Travis Craig, Noel Craske, Matthew Crocker, Tadeusz Czachorski, Milan Češka, Honghua Dai, Bart de Decker, Deborah Dent, Andrej Dobnikar, Sait Dogru, Peter Dolog, Georg Dorfner, Ludoslaw Drelichowski, Matija Drobnič, Maciej Drozdowski, Marek Druzdzel, Marjan Družovec, Jozo Dujmović, Pavol Ďuriš, Amnon Eden, Johann Eder, Hesham El-Rewini, Darrell Ferguson, Warren Fergusson, David Flater, Pierre Flener, Wojciech Fliegner, Vladimir A. Fomichov, Terrence Forgarty, Hans Fraaije, Stan Franklin, Violetta Galant, Hugo de Garis, Eugeniusz Gatnar, Grant Gayed, James Geller, Michael Georgiopolus, Michael Gertz, Jan Goliński, Janusz Gorski, Georg Gottlob, David Green, Herbert Groiss, Jozsef Gyorkos, Marten Haglind, Abdelwahab Hamou-Lhadj, Inman Harvey, Jaak Henno, Marjan Hericko, Henry Hexmoor, Elke Hochmueller, Jack Hodges, John-Paul Hosom, Doug Howe, Rod Howell, Tomáš Hruška, Don Huch, Simone Fischer-Huebner, Zbigniew Huzar, Alexey Ippa, Hannu Jaakkola, Sushil Jajodia, Ryszard Jakubowski, Piotr Jedrzejowicz, A. Milton Jenkins, Eric Johnson, Polina Jordanova, Djani Juričič, Marko Juvancic, Sabhash Kak, Li-Shan Kang, Ivan Kapustøk, Orlando Karam, Roland Kaschek, Jacek Kierzenka, Jan Kniat, Stavros Kokkotos, Fabio Kon, Kevin Korb, Gilad Koren, Andrej Krajnc, Henryk Krawczyk, Ben Kroese, Zbyszko Krolikowski, Benjamin Kuipers, Matjaž Kukar, Aarre Laakso, Sofiane Labidi, Les Labuschagne, Ivan Lah, Phil Laplante, Bud Lawson, Herbert Leitold, Ulrike Leopold-Wildburger, Timothy C. Lethbridge, Joseph Y-T. Leung, Barry Levine, Xuefeng Li, Alexander Linkevich, Raymond Lister, Doug Locke, Peter Lockeman, Vincenzo Loia, Matija Lokar, Jason Lowder, Kim Teng Lua, Ann Macintosh, Bernardo Magnini, Andrzej Małachowski, Peter Marcer, Andrzej Marciniak, Witold Marciszewski, Vladimir Marik, Jacek Martinek, Tomasz Maruszewski, Florian Matthes, Daniel Memmi, Timothy Menzies, Dieter Merkl, Zbigniew Michalewicz, Armin R. Mikler, Gautam Mitra, Roland Mittermeir, Madhav Moganti, Reinhard Moller, Tadeusz Morzy, Daniel Mossé, John Mueller, Jari Multisilta, Hari Narayanan, Jerzy Nawrocki, Rance Necaise, Elzbieta Niedzielska, Marian Niedq'zwiedziński, Jaroslav Nieplocha, Oscar Nierstrasz, Roumen Nikolov, Mark Nissen, Jerzy Nogieć, Stefano Nolfi, Franc Novak, Antoni Nowakowski, Adam Nowicki, Tadeusz Nowicki, Daniel Olejar, Hubert Österle, Wojciech Olejniczak, Jerzy Olszewski, Cherry Owen, Mieczyslaw Owoc, Tadeusz Pankowski, Jens Penberg, William C. Perkins, Warren Persons, Mitja Peruš, Fred Petry, Stephen Pike, Niki Pissinou, Aleksander Pivk, Ullin Place, Peter Planinšec, Gabika Polčicová, Gustav Pomberger, James Pomykalski, Tomas E. Potok, Dimithu Prasanna, Gary Preckshot, Dejan Rakovič, Cveta Razdevšek Pučko, Ke Qiu, Michael Quinn, Gerald Quirchmayer, Vojislav D. Radonjic, Luc de Raedt, Ewaryst Rafajlowicz, Sita Ramakrishnan, Kai Rannenberg, Wolf Rauch, Peter Rechenberg, Felix Redmill, James Edward Ries, David Robertson, Marko Robnik, Colette Rolland, Wilhelm Rossak, Ingrid Russel, A.S.M. Sajeev, Kimmo Salmenjoki, Pierangela Samarati, Bo Sanden, P. G. Sarang, Vivek Sarin, Iztok Savnik, Ichiro Satoh, Walter Schempp, Wolfgang Schreiner, Guenter Schmidt, Heinz Schmidt, Dennis Sewer, Zhongzhi Shi, Mária Smolárová, Carine Souveyet, William Spears, Hartmut Stadtler, Stanislaw Stanek, Olivero Stock, Janusz Stokłosa, Przemysław Stpiczyński, Andrej Stritar, Maciej Stroinski, Leon Strous, Ron Sun, Tomasz Szmuc, Zdzislaw Szyjewski, Jure Šilc, Metod Škarja, Jiři Šlechta, Chew Lim Tan, Zahir Tari, Jurij Tasič, Gheorge Tecuci, Piotr Teczynski, Stephanie Teufel, Ken Tindell, A Min Tjoa, Drago Torkar, Vladimir Tosic, Wieslaw Traczyk, Denis Trček, Roman Trobec, Marek Tudruj, Andrej Ule, Amjad Umar, Andrzej Urbanski, Marko Uršič, Tadeusz Usowicz, Romana Vajde Horvat, Elisabeth Valentine, Kanonkluk Vanapipat, Alexander P. Vazhenin, Jan Verschuren, Zygmunt Vetulani, Olivier de Vel, Didier Vojtisek, Valentino Vranić, Jozef Vyskoc, Eugene Wallingford, Matthew Warren, John Weckert, Michael Weiss, Tatjana Welzer, Lee White, Gerhard Widmer, Stefan Wrobel, Stanislaw Wrycza, Tatyana Yakhno, Janusz Zalewski, Damir Zazula, Yanchun Zhang, Ales Zivkovic, Zonling Zhou, Robert Zorc, Anton P. Železnikar

Informatica

An International Journal of Computing and Informatics

Web edition of Informatica may be accessed at: http://www.informatica.si.

Subscription Information Informatica (ISSN 0350-5596) is published four times a year in Spring, Summer, Autumn, and Winter (4 issues per year) by the Slovene Society Informatika, Vožarski pot 12, 1000 Ljubljana, Slovenia.
The subscription rate for 2009 (Volume 33) is
– 60 EUR for institutions,
– 30 EUR for individuals, and
– 15 EUR for students
Claims for missing issues will be honored free of charge within six months after the publication date of the issue.

Typesetting: Borut Žnidar.
Printing: Dikplast Kregar Ivan s.p., Kotna ulica 5, 3000 Celje.

Orders may be placed by email (drago.torkar@ijs.si), telephone (+386 1 477 3900) or fax (+386 1 251 93 85). The payment should be made to our bank account no.: 02083-0013014662 at NLB d.d., 1520 Ljubljana, Trg republike 2, Slovenija, IBAN no.: SI56020830013014662, SWIFT Code: LJBASI2X.

Informatica is published by Slovene Society Informatika (president Niko Schlamberger) in cooperation with the following societies (and contact persons):
Robotics Society of Slovenia (Jadran Lenarčič)
Slovene Society for Pattern Recognition (Franjo Pernuš)
Slovenian Artificial Intelligence Society; Cognitive Science Society (Matjaž Gams)
Slovenian Society of Mathematicians, Physicists and Astronomers (Bojan Mohar)
Automatic Control Society of Slovenia (Borut Zupančič)
Slovenian Association of Technical and Natural Sciences / Engineering Academy of Slovenia (Igor Grabec)
ACM Slovenia (Dunja Mladenič)

Informatica is surveyed by: Citeseer, COBISS, Compendex, Computer & Information Systems Abstracts, Computer Database, Computer Science Index, Current Mathematical Publications, DBLP Computer Science Bibliography, Directory of Open Access Journals, InfoTrac OneFile, Inspec, Linguistic and Language Behaviour Abstracts, Mathematical Reviews, MatSciNet, MatSci on SilverPlatter, Scopus, Zentralblatt Math

The issuing of the Informatica journal is financially supported by the Ministry of Higher Education, Science and Technology, Trg OF 13, 1000 Ljubljana, Slovenia.

Volume 33 Number 4 November 2009 **ISSN 0350-5596**

Informatica

An International Journal of Computing and Informatics

Informatica **33** (2009) Number 4, pp. 403–523