StuCoSReC

Proceedings of the 2015 2nd Student Computer Science Research Conference

University of Primorska Press

StuCoSReC
Proceedings of the 2015 2nd Student Computer Science Research Conference

Edited by
Iztok Fister jr. and Andrej Brodnik

Reviewers and Programme Committee
Zoran Bosnić ▪ University of Ljubljana, Slovenia
Borko Bošković ▪ University of Maribor, Slovenia
Janez Brest ▪ University of Maribor, Slovenia
Andrej Brodnik ▪ University of Primorska
and University of Ljubljana, Slovenia
Mojca Ciglarič ▪ University of Ljubljana, Slovenia
Jani Dugonik ▪ University of Maribor, Slovenia
Iztok Fister ▪ University of Maribor, Slovenia
Iztok Fister Jr. ▪ University of Maribor, Slovenia
Matjaž Gams ▪ Jozef Stefan Institute, Slovenia
Tao Gao ▪ North China Electric Power University, China
Andres Iglesias ▪ Universidad de Cantabria, Spain
Branko Kavšek ▪ University of Primorska, Slovenia
Miklos Kresz ▪ University of Szeged, Hungary
Niko Lukač ▪ University of Maribor, Slovenia
Uroš Mlakar ▪ University of Maribor, Slovenia
Vili Podgorelec ▪ University of Maribor, Slovenia
Peter Rogelj ▪ University of Primorska, Slovenia
Xin-She Yang ▪ Middlesex University, United Kingdom
Aleš Zamuda ▪ University of Maribor, Slovenia
Borut Žalik ▪ University of Maribor, Slovenia

Published by
University of Primorska Press
Titov trg 4, si-6000 Koper

Editor-in-Chief
Jonatan Vinkler

Managing Editor
Alen Ježovnik

Koper, 2015

ISBN 978-961-6984-01-0 (pdf)
www.hippocampus.si/ISBN/978-961-6984-01-0.pdf
ISBN 978-961-6984-02-7 (html)
www.hippocampus.si/ISBN/978-961-6984-02-7/index.html

CIP - Kataložni zapis o publikaciji
Narodna in univerzitetna knjižnica, Ljubljana

004(082)(0.034.2)

STUDENT Computer Science Research Conference (2 ; 2015)
StuCoSReC [Elektronski vir] : proceedings of the 2015 2nd Student Computer Science Research Conference / [edited by Iztok Fister and Andrej Brodnik]. - El. knjiga. - Koper : University of Primorska, 2015

Način dostopa (URL): http://www.hippocampus.si/isbn/978-961-6984-01-0.pdf
Način dostopa (URL): http://www.hippocampus.si/isbn/978-961-6984-02-7/index.html

ISBN 978-961-6984-01-0 (pdf)
ISBN 978-961-6984-02-7 (html)
1. Gl. stv. nasl. 2. Fister, Iztok, ml.
281595648

Preface

Dear reader,

in front of you are the proceedings of *The 2nd Student Computer Science Research Conference (StuCoReC)*[1]. The conference started its journey among Slovenian higher education institutions and it is this time hosted by the *Jožef Stefan International Postgraduate School*.

Furthermore, the conference expanded its program to invite even a broader audience and therefore included the national session for master and undergraduate students to present their work. Consequently on the conference were presented three contributions in the national session and seven contributions in the international session. The authors come from all three Slovenian universities, or more precisely from their faculties with the Computer Science program and from Szeged University. All papers were reviewed by at least two members of the international programme committee who made comments that were sent back to authors to further improve their papers

The topics of presented contributions prove the variety of research done by students. On one side we have purely theoretical topics in algorithm theory and then expanding to very applied research in bioinformatics and linguistics. Quite a number of contributions are in the area of engineering and also HCI.

Let us wrap up the preface with an observation that the conference does serve its purpose to bring together student researchers to exchange their experiences and ideas, and to establish future contacts. See you on *The 3rd Student Computer Science Research Conference*!

1 http://labraj.feri.um.si/stucosrec2015/

Multikonferenca Informacijska družba 2015

Predgovor

Multikonferenca Informacijska družba (http://is.ijs.si) je z osemnajsto zaporedno prireditvijo osrednji srednjeevropski dogodek na področju informacijske družbe, računalništva in informatike. Letošnja prireditev traja tri tedne in poteka na Fakulteti za računalništvo in informatiko in Institutu »Jožef Stefan«.

Informacijska družba, znanje in umetna inteligenca se razvijajo čedalje hitreje. V vse več državah je dovoljena samostojna vožnja inteligentnih avtomobilov, na trgu je moč dobiti čedalje več pogosto prodajanih avtomobilov z avtonomnimi funkcijami kot »lane asist«. Čedalje več pokazateljev kaže, da prehajamo v naslednje civilizacijsko obdobje, hkrati pa so konflikti sodobne družbe čedalje težje razumljivi.

Letos smo v multikonferenco povezali dvanajst odličnih neodvisnih konferenc. Predstavljenih bo okoli 300 referatov v okviru samostojnih konferenc in delavnic, prireditev bodo spremljale okrogle mize in razprave ter posebni dogodki kot svečana podelitev nagrad. Referati so objavljeni v zbornikih ultikonference, izbrani prispevki pa bodo izšli tudi v posebnih številkah dveh znanstvenih revij, od katerih je ena Informatica, ki se ponaša z 38-letno tradicijo odlične znanstvene revije.

Multikonferenco Informacijska družba 2015 sestavljajo naslednje samostojne konference:

Inteligentni sistemi
Kognitivna znanost
Izkopavanje znanja in podatkovna skladišča
Sodelovanje, programska oprema in storitve v informacijski družbi
Vzgoja in izobraževanje v informacijski družbi
Soočanje z demografskimi izzivi
Kognitonika
Delavnica »SPS EM-zdravje«
Delavnica »Pametna mesta in skupnosti kot razvojna priložnost Slovenije«
Druga študentska konferenca s področja računalništva in informatike za doktorske študente
Druga študentska konferenca s področja računalništva in informatike za vse študente
ISSEP15 - Osma mednarodna konferenca o informatiki v šolah: razmere, evolucija in perspektiva.

Soorganizatorji in podporniki konference so različne raziskovalne institucije in združenja, med njimi tudi ACM Slovenija, SLAIS in Inženirska akademija Slovenije. V imenu organizatorjev konference se zahvaljujemo združenjem in inštitucijam, še posebej pa udeležencem za njihove dragocene prispevke in priložnost, da z nami delijo svoje izkušnje o informacijski družbi. Zahvaljujemo se tudi recenzentom za njihovo pomoč pri recenziranju.

V 2015 bomo tretjič podelili nagrado za življenjske dosežke v čast Donalda Michija in Alana Turinga. Nagrado Michie-Turing za izjemen življenjski prispevek k razvoju in promociji informacijske družbe bo prejel

Information Society 2015

Foreword

In its 18th year, the Information Society Multiconference (http://is.ijs.si) remains one of the leading conferences in Central Europe devoted to information society, computer science and informatics. In 2015 it is extended over three weeks located at Faculty of computer science and informatics and at the Institute "Jožef Stefan".

The pace of progress of information society, knowledge and artificial intelligence is speeding up. Several countries allow autonomous cars in regular use, major car companies sell cars with lane assist and other intelligent functions. It seems that humanity is approaching another civilization stage. At the same time, society conflicts are growing in numbers and length.

The Multiconference is running in parallel sessions with 300 presentations of scientific papers at twelve conferences, round tables, workshops and award ceremonies. The papers are published in the conference proceedings, and in special issues of two journals. One of them is Informatica with its 38 years of tradition in excellent research publications.

The Information Society 2015 Multiconference consists of the following conferences:

Intelligent Systems
Cognitive Science
Data Mining and Data Warehouses
Collaboration, Software and Services in Information Society
Education in Information Society
Facing Demographic Challenges
Cognitonics
SPS EM-Health Workshop
Workshop »Smart Cities and Communities as a Development Opportunity for Slovenia«
2nd Computer Science Student Conference, PhD Students
2nd Computer Science Student Conference, Students
8th International Conference on Informatics in Schools: Situation, Evolution, and Perspective.

The Multiconference is co-organized and supported by several major research institutions and societies, among them ACM Slovenia, i.e. the Slovenian chapter of the ACM, SLAIS and the Slovenian Engineering Academy. In the name of the conference organizers we thank all societies and institutions, all participants for their valuable contribution and their interest in this event, and the reviewers for their thorough reviews.

For 2013 and further, the award for life-long outstanding contributions will be delivered in memory of Donald Michie and Alan Turing. The life-long outstanding contribution to development and promotion of information society in our country is awarded to Dr. Jurij Tasič. In addition, a reward for current achievements was pronounced to Dr. Domnu Mongusu. The informa-

prof. dr. Jurij Tasič. Priznanje za dosežek leta je pripadlo dr. Domnu Mongusu. Že petič podeljujemo nagradi »informacijska limona« in »informacijska jagoda« za najbolj (ne)uspešne poteze v zvezi z informacijsko družbo. Limono je dobilo počasno uvajanje informatizacije v slovensko pravosodje, jagodo pa spletna aplikacija »Supervizor«. Čestitke nagrajencem!

Nikolaj Zimic, predsednik programskega odbora

Matjaž Gams, predsednik organizacijskega odbora

tion strawberry is pronounced to the web application "Supervizor, while the information lemon goes to lack of informatization in the national judicial system. Congratulations!

Nikolaj Zimic, Programme Committee Chair

Matjaž Gams, Organizing Committee Chair

Contents

Izdelava ogrodja robota s pomočjo 3D tiskalnika

Primož Bencak
Univerza v Mariboru
Fakulteta za strojništvo
Smetanova 17, Maribor
primoz.bencak@
student.um.si

Fister Dušan
Univerza v Mariboru
Fakulteta za strojništvo
Smetanova 17, Maribor
dusan.fister@
student.um.si

Riko Šafarič
Univerza v Mariboru
Fakulteta za elektrotehniko,
računalništvo in informatiko
Smetanova 17, Maribor
riko.safaric@um.si

POVZETEK

V tem članku predstavljamo izdelavo mobilnega robota, ki je zmožen prevoziti čim dlje v robotskem labirintu, sestavljenem iz manjših kvadratnih gradnikov - polj. 3D tiskanje predstavlja revolucionarno metodo konstruiranja vsakdanjih objektov z metodo nanašanja materiala. Uporaba te metode se je začela širše uveljavljati šele v zadnjih letih, čeprav je princip tiskanja znan že dobrih trideset let. Dandanes si jo z nekaj inženirske žilice lahko privošči vsakdo. Enostavnost te metode omogoča konstruiranje poljubnih gradnikov, zato smo se odločili preizkusiti to metodo za konstruiranje ogrodja mobilnega robota.

Kjučne besede

tehnologija dodajanja, 3D tiskanje, hitra izdelava prototipov

1. UVOD

Robot je običajno elektro-mehanska naprava, ki jo vodi računalniški program. Deluje v okolju, ki ga je s pomočjo akcij, za katere je pooblaščen, sposoben tudi spreminjati. Izvaja ponavljajoče se akcije, ki jih je prej izvajal človek. Slednjega pri tem izpodriva predvsem pri nevarnih opravilih, npr. lakiranju avtomobilov v avtomobilski industriji, izvajanju poskusov v vesolju, ipd.

Načrtovanje robota je zapleten projekt, saj vključuje znanja iz različnih domen, tj. elektrotehnike, strojništva, računalništva, ipd. Orientirani na mobilne robote domače izdelave poznamo številne alternativne pristope izdelave ogrodja. Najpogosteje izbiramo med uporabo plastične posode ali plastičnih plošč [4], Lego kock [1], lesenih konstrukcij ali celo predelavo ogrodja modelov avtomobilčkov, ipd. Za robote industrijske izdelave taka ogrodja ne pridejo v poštev, nosilno funkcijo največkrat opravlja lahko aluminijasto ogrodje, ki ostalim gradnikom omogoča trdno oporo.

Labirint, ki ga robot mora prevoziti, ima postavljene stene, v katere se ne sme zaleteti in med katerimi manevrira. Eden izmed naših ciljev je sestaviti primerno ogrodje za mobilnega robota. Zaradi čimvečje popularnosti 3D nanašanja in tiskanja nas je zanimalo, ali lahko s to tehnologijo ustvarimo dovolj kakovostno ogrodje, ki bo po zahtevanih lastnostih primerljivo tistim, ki so izdelane po bolj konvencionalnem načinu. 3D tiskanje je začelo svoj razvoj že v zgodnjih devetdesetih letih prejšnjega stoletja [9]. Tehnologija je napredovala celo do te mere, da so s 3D tiskalnikom natisnili prototip avtomobila Audi [7].

Za izdelavo konstrukcije robota smo v naši študiji uporabili odprtokodni 3D tiskalnik Mendel podjetja RepRap, ki je na tržišču na voljo v obliki kompleta *naredi sam* (angl. do-it-yourself). S samo konstrukcijo 3D tiskalnika se nismo ukvarjali, saj je bilo to delo pred nami že opravljeno. Naš del je zajemal načrtovanje in tiskanje samega ogrodja robota. Načrtovanje ponavadi poteka v katerem od strojniških programov, kjer uporabnik želeno ogrodje izriše, določi končne mere in s pomočjo preprostih pretvorniških programov risbo pretvori v tiskalniku razumljiv program. Za izris smo uporabili program SolidWorks [10], medtem ko za pretvorbo v G-kodo program Slic3r [6].

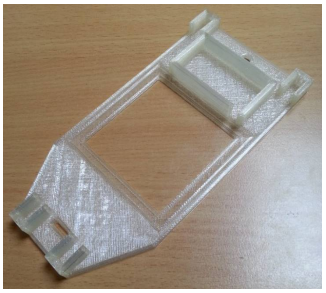
Struktura članka v nadaljevanju je naslednja. Poglavje 2 govori o problemu izdelave ogrodja robota s pomočjo 3D tiskalnika. V poglavju 3 opišemo postopek izdelave ogrodja robota na 3D tiskalniku. Preizkus izdelanega ogrodja je opisan v poglavju 4. Članek zaključimo s poglavjem 5, kjer nakažemo možne izboljšave našega dela v prihodnje.

2. 3D TISKANJE

3D tiskalnik je mehatronska naprava, ki omogoča izdelavo tridimenzionalnih objektov s pomočjo nanosa posameznih slojev gradiva (angl. additive manufacturing) [5]. Razvit je bil kot nadgradnja običajnega 2D tiskalnika, ki pa podpira tiskanje v treh dimenzijah. 3D tiskalnik je dobrodošel pripomoček inženirjev, arhitektov, oblikovalcev in ostalih tehnoloških navdušencev.

Tradicionalni pristop izdelave raznih izdelkov iz polizdelkov je potekal z odnašanjem (substrakcijo) gradiva. Nov, *revolucionarni* pristop pa deluje po principu dodajanja (adicije) gradiva [8]. Tako lahko sedaj s 3D tiskalnikom izdelamo izdelke, ki jih prej sploh ni bilo moč izdelati [2]. Prednost take izdelave je tudi v tem, da ni odpadkov, saj porabimo samo toliko materiala, kolikor ga je v končnem izdelku. Izdelki so tako cenejši, konkurenčnejši in tudi ekološko prijaznejši. Sprva so bili 3D tiskalniki namenjeni tiskanju manj zahtev-

nih oblik ali prototipov za izdelke, ki so jih potem izdelovali z drugo tehnologijo. Dandanes pa se vse bolj nagibajo k temu, da je določen izdelek že v celoti izdelan s 3D tiskalnikom. V dobrih tridesetih letih svojega razvoja smo odkrili marsikaj novega. Novosti so predvsem v različnih materialih, s katerimi lahko izdelke enostavno *tiskamo* (slika 1).



Slika 1: Izdelek tiskanja.

3D tiskalnik je sestavljen iz različnih elektronskih in strojnih elementov. Najpomembejši so:

- iztiskalna šoba (angl. extruder nozzle): skrbi za brizganje tekočega gradiva - filamenta (nanašalnega vlakna),
- koračni motorji: zagotavljajo premikanje v ravnini X-Y in osi Z,
- grelec: utekočini filament in skrbi za vzdrževanje temperature,
- grelna mizica,
- napajalni del z elektronskim vezjem.

Omenjeni sestavni deli opredeljujejo tiskalnik kot mehatronsko napravo. Čeprav je tiskalnik statična naprava uporablja številne aktuatorje gibanja. Ti skrbijo za premikanje iztiskalne šobe, ki iztiska predhodno utekočinjen filament. Utekočinja ga reguliran grelec, ki poleg grelne mizice predstavlja osnovo za kakovostno izdelano ogrodje. Iztiskalno šobo poganja dodatni električni motor. Šoba predstavlja izmed vseh gradnikov najobčutljivejši del.

Koračni motorji opravljajo dve funkciji. Poleg premikanja šobe skrbijo tudi za merjenje položaja, kar je glavna značilnost koračnih motorjev. Zaradi tega jih odlikuje enostavna uporaba, vendar po drugi strani omejuje visok vrtilni moment in visoka vrtilna frekvenca. V kolikor motor doseže katero ob obeh lastnosti, pride do izpada iz koraka, kar prepozna elektronsko vezje in gibanje ustavi.

3. IZDELAVA OGRODJA ROBOTA

Za 3D tisk smo se odločili, ker ima nekaj prednosti pred ostalimi metodami izdelave ogrodja robota. Da bi izdelali uporabno ogrodje, ki zadošča našim potrebam, smo se morali osredotočiti na bistvene lastnosti, ki jih ogrodje mora imeti, tj.:

- trdnost,
- teža,
- zahtevnost izdelave,
- čas izdelave.

Ugotovili smo, da 3D tisk ponuja veliko prednosti pred ostalimi oblikami načrtovanja in izdelave, zato smo jo tudi uporabili v naši študiji. Postopek izdelave je sestavljen iz petih faz [11]:

- načrtovanje 3D modela v programu SolidWorks,
- prenos 3D modela v Slic3r,
- prevajanje G-kode (angl. G-code),
- optimizacija nastavitev temperature grelca in pozicije 3D modela,
- predgretje in 3D tisk.

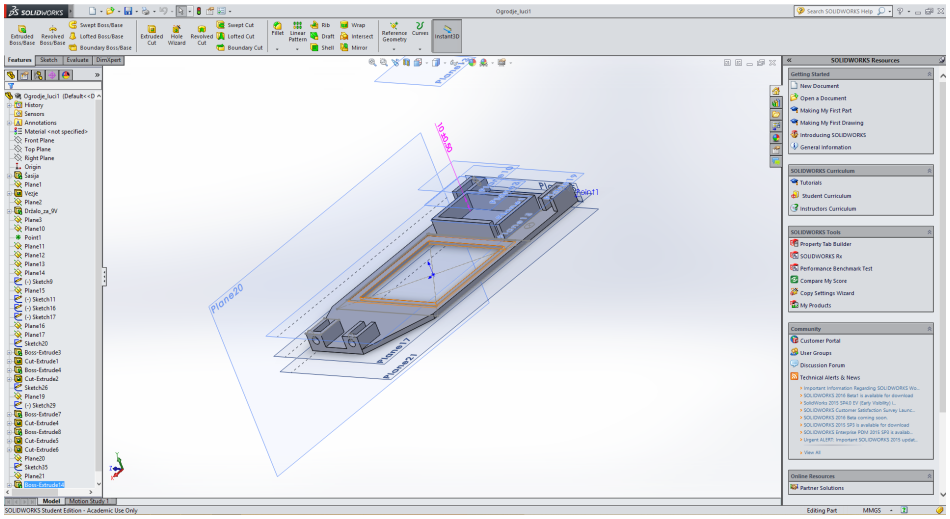
Načrtovanje 3D modela smo izvedli v razvojnem orodju SolidWorks (slika 2), ki je zelo cenjen med inženirji in oblikovalci, saj ponuja vrsto različnih načinov risanja. Za samo konstruiranje je sicer potrebno nekaj predznanja, ki pa ga ni težko usvojiti.

Gradnjo 3D modela smo začeli z osnovnim likom – pravokotnikom, ki smo mu prirezali stranice. Z orodjem Extruded Boss/Base smo pravokotniku določili še tretjo dimenzijo - višino, ter dobili kvader. V sredini kvadra smo izrezali odprtino kvadratne oblike, namenjeno tiskanemu vezju. Na koncu smo dodali še nosilce za LED diode, nosilec za 9V baterijo, ter luknjo za stikalo. Končnemu modelu smo določili še dimenzije ter model shranili v datoteko *.STL. Bistveno pri načrtovanju 3D modela je, da se držimo nekaj osnovnih napotkov, ki pripomorejo k boljšemu končnemu izdelku, tj.:

- dimenzije končnega izdelka morajo biti v skladu z razsežnostjo našega 3D tiskalnika,
- izogibamo se konstrukciji *mostičkov*, tj. delov, kjer bi naša ekstruderska šoba filament vlekla po zraku in ne po prejšnjem sloju,
- če bo naš model imel izvrtine (luknje), morajo biti dimenzije na modelu večje od dejanskih, sicer jih moramo povrtavati,
- izogibamo se pretankim stenam (nizka trdnost), hkrati pa tudi nepotrebno debelim (čas tiskanja se zelo podaljša, porabimo več gradiva).

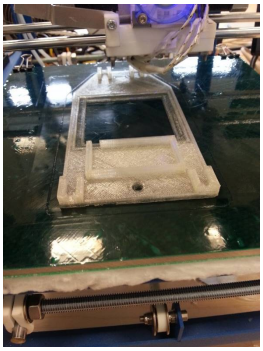
Datoteko izvozimo v formatu *.STL (Lithography), ki izrisan trodimenzialni model zapiše kot množico trikotnikov. Datoteka *.STL je namenjena za pretvorbo v tiskalniku razumljivo obliko, G-kodo. G-koda predstavlja numerični zapis modela, razumljivega numerično krmiljenim strojem (angl. numerical control, krajše NC) [3]. Pretvorbo opravi preprost program Slic3r, ki vsebuje navodila za:

- premikanje šobe po tiskalni površini [2],
- uporabo kombinacije koordinat in slojev (angl. layer), po katerem se premika,
- količino gradiva, ki se bo uporabil pri tiskanju,
- nastavitve temperature iztiskalne šobe in grelne mizice,
- način 3D tiskanja, ipd.



Slika 2: Razvojno orodje Solidworks za 3D-tiskanje.

Ročno lahko nastavimo tudi način zapolnitve medrobnega prostora, spreminjamo delež le-tega in podamo navodila za premikanje šobe po določenem vzorcu, morebiti izberemo način *satje*(angl. honeycomb) ali *premočrtno* (angl. rectilinear) gibanje. Različne nastavitve vplivajo na končno trdoto izdelka in na količino porabljenega filamenta (slika 3). Končno datoteko z vsemi nastavitvami nato prenesemo na 3D tiskalnik, ki prične s tiskanjem. Po uspešno naloženem



Slika 3: 3D-tiskanje.

programu se izvajanje le-tega ne prične takoj. Sprva se začne predgretje šobe in grelne mizice, kar ponavadi traja do 10 minut. Šoba se potem premakne v izhodiščno lego na koordinate (0,0), ki je referenčna točka za G-kodo. Programski stavek je sestavljen iz digitalnih besed (16-bitna dolžina niza), ki vsebujeta ukaz in ukazni parameter.

Tabela 1: Predstavitev G-kode

Naslov	Vrednost	Naslov	Vrednost	Naslov	Vrednost
G	01	X	800	Y	360
BESEDA		BESEDA		BESEDA	

Ker 3D tiskalnik nima nobenih senzorjev, ki bi preverjali trenutno lego na koordinatni mreži (površini za tiskanje), ampak pravilno pozicioniranje šobe zagotavlja koračni motor, je zelo pomembno, da mize, na kateri je 3D tiskalnik ne premikamo. Prvi sloj tiskalnik tiska nekoliko več časa, medtem ko se preostali sloji tiskajo hitreje. Vzrok za daljše tiskanje prvega sloja je v zapolnitvi prostora oz. večji kakovosti končnega izdelka. Na ta način je spodnji del, ki se dotika ogrevalne mizice na otip gladek ter svetleč. Primer programa v G-kodi prikazuje algoritem 1 [12].

Algoritem 1 G-koda

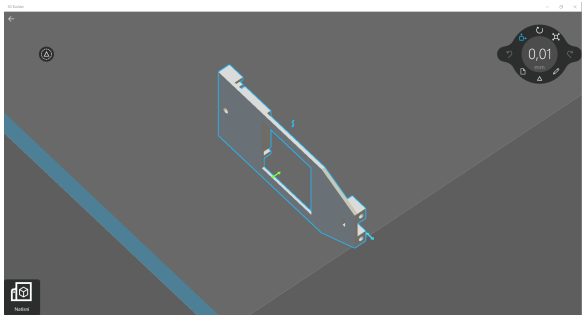
```
1: M107              ; izklop ventilatorja
2: M104 S205         ; nastavi temperaturo
3: G28               ; pojdi na koordinatno izhodišče (0,0)
4: G1 Z5 F5000       ; dvig ekstruderske šobe
```

Naprava mora biti priklopljena na stabilno električno omrežje, saj v primeru izpada energije, s tiskom ne moremo nadaljevati in moramo začeti znova. Čas tiska je pogojen s hitrostjo koračnih motorjev, še bolj kot s tem, pa z razsežnostjo in debelino sten našega izdelka. Za naše ogrodje je tiskalnik porabil okoli tri ure. Po koncu tiskanja je potrebno počakati, da se ogrevana podloga ohladi na okoli 50°C, saj takrat predmet najlaže ločimo od podloge. Končan izdelek je takoj trd na otip in pripravljen na nadaljnjo obdelavo (npr. barvanje). Pri morebitnem dodatnem povrtavanju ali brušenju moramo biti zelo pazljivi, saj ima tako PLA kot ABS filament, nizko temperaturo tališča. Pri nepravilni obdelavi lahko izdelek trajno deformiramo in nepopravljivo poškodujemo.

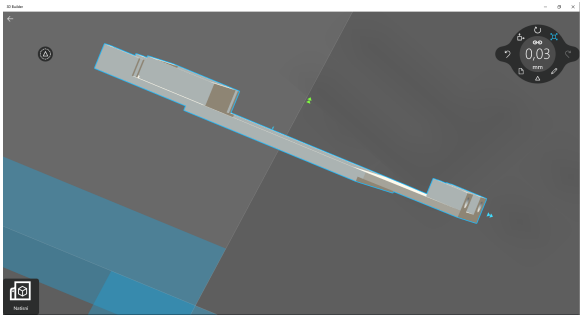
4. POSKUSI IN REZULTATI

Na poti do uspešne izdelave ogrodja smo se srečali z nekaterimi težavami. Luknje na natisnjenem objektu niso bile enako velike kot tiste, ki smo jih načrtovali v SolidWorks-u. Zahtevale so dodatno obdelavo s pilo. Vzrok za ta pojav je natančnost koračnih motorjev, ki ne morejo natisniti popolne izvrtine, saj jo ustvarijo kot množico tankih črt, zloženo drugo ob drugi. Natančnost izdelave izvrtine je odvisna od ločljivosti koračnega motorja.

Dodaten problem predstavlja tvorba mostičkov. Tiskanje objekta poteka od spodnjega sloja proti zgornjemu, kar pomeni, da mora spodnji sloj biti vedno enak ali večji od zgornjega. Primer pravilnega načrtovanja je prikazan na sliki 4a,



(a) Pravilno načrtovanje

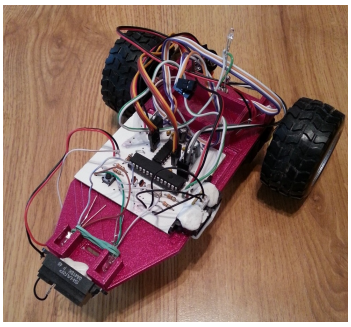


(b) Nepravilno načrtovanje

Slika 4: Primerjava dobrega in slabega načrtovanja.

kjer spodnji sloj v vsakem primeru služi kot opora zgornjemu - ta se lahko nanaša na spodnjega. Slika 4b je primer nepravilnega načrtovanja objektov. Spodnji sloj ne služi kot trdna opora celotni strukturi, vendar le delu. Posledico tega predstavlja tiskanje filamenta v zrak, zaradi česar objekta ni moč uporabiti.

Težavo predstavlja tudi odvijanje filamenta s tulca. Zaradi teže navitja filamenta, ga motor ne zmore vleči enakomerno, kar preprosto rešimo z ročnim odvijanjem vsakih nekaj minut. V nasprotnem primeru iztiskalna šoba ne dobiva dovolj filamenta, kar pomeni, da sloji objekta niso enakomerno nanešeni. Ogrodje, ki smo ga izdelali, je namenjenu mobil-



Slika 5: Končna izvedba mobilnega robota.

nemu robotu, čigar naloga je vožnja skozi labirint (slika 5). Na ogrodje sta na zadnjem delu pritrjena modificirana servo motorčka, nad motorčkoma pa se nahaja držalo za 9V baterijo. V sredini je okvir, v katerem je nameščeno vezje robota, ki ga nadzoruje mikrokrmilnik PIC18F25K22, podjetja Microchip. Zaradi dobrega razmerja med močjo servomotorčkov in nizko težo, robot dosega zadovoljive hitrosti, ki pa so pogojene s kapaciteto baterije. Na prednjem delu in levi strani mobilnega robota, sta nameščena dva infrardeča senzorja oddaljenosti, SHARP GP2Y0A41SK0F. S senzorjema robot zaznava oddaljenost od ovir. Na podlagi izhodnih vrednosti senzorjev, se mikrokrmilnik, ki smo ga prej sprogramirali v programskem orodju MikroC, ustrezno odzove z vrtenjem osi koračnim motorjem.

4.1 Diskusija

Prednost gradnje ogrodja s 3D tiskalnikom v primerjavi s tistimi, izdelani iz bolj konvencionalnih materialov vidimo predvsem pri končnem času, ki ga potrebujemo za celotno izdelavo - potrebna sta zgolj modeliranje in tiskanje. Ker je tiskalnik avtonomna naprava, človeku ni potrebno biti prisoten, kar izdelovalni čas dodatno skrajša. Filamenti so na voljo v različnih barvah, zato izdelka ni potrebno barvati, prav tako pa v večini primerov odpade tudi brušenje. Na ta način namreč odpade velik del časa, ki bi ga sicer potrebovali, da bi izdelek naprej načrtovali (opravljanje meritev), primerno mehansko obdelali ter v nekaterih primerih tudi pobarvali. Za omenjeno metodo niso potrebne ročne spretnosti, ki v primeru rokovanja z obdelovalnimi stroji v veliki meri vplivajo na kakovost izdelka. Namesto več naprav potrebujemo samo računalnik in tiskalnik. Za pot od ideje do končnega izdelka smo porabili okoli štiri ure, izmed katerih je več kot dobra polovica časa pripadala samo tiskanju. Ob izbiri konvencialne metode gradnje ogrodja iz lesa primerljive oblike je potrebno:

- poiskati primeren kos s pravšnjo trdoto in kakovostjo,
- prenesti mere na obdelovanec,
- izrezati ogrodje z namizno žago,
- izvrtati luknje,
- pobrusiti površino in
- ogrodje prebarvati.

Celotni postopek časovni okvir izdelovanja ogrodja s konvencialno metodo tako drastično poveča. Rrobot, prikazan na sliki 5, skupaj z devet voltno (blok) baterijo tehta le 352 gramov. Čeprav je ogrodje sorazmerno veliko, pa je vzrok za malo težo v razporeditvi filamenta po volumnu. Notranjost sten namreč v našem primeru ni bila zapolnjena stoodstotno, saj smo na ta način privarčevali tudi s filamentom.

5. SKLEP

V članku smo predstavili izdelavo ogrodja za robota s pomočjo 3D tiska. S končnim izdelkom smo dosegli želene rezultate. Ogrodje je namreč dovolj trdno, da se tudi ob močnejši tlačni obremenitvi ne poruši. Uporaba ogrodja v okolju s povišano temperaturo predstavlja splošno težavo pri večini postopkov nanašanja materiala, saj je temperaturna obstojnost takih materialov omejena na nižje temperature.

Ta vpliv lahko zmanjšamo z uporabo grelne mizice, vendar ga v celoti ne moremo odpraviti.

Nadaljno težavo predstavlja natančnost tiskalnika. Z zmanjševanjem premera iztiskalne šobe bi jo lahko povečali, prav tako z zamenjavo koračnih motorjev za servomotorje in namestitvijo natančnih dajalnikov položaja. Ti ukrepi pa so povezani s povečanimi stroški tiskanja.

6. REFERENCES

[1] D. Baum and R. Zurcher. *Definitive Guide to Lego Mindstorms*, volume 2. Apress, 2003.

[2] B. Berman. 3-d printing: The new industrial revolution. *Business Horizons*, 55(2):155 – 162, 2012.

[3] P. Bézier. Numerical control. 1970.

[4] G. Caprari, P. Balmer, R. Piguet, and R. Siegwart. The autonomous micro robot "alice": a platform for scientific and commercial applications. In *Micromechatronics and Human Science, 1998. MHS'98. Proceedings of the 1998 International Symposium on*, pages 231–235. IEEE, 1998.

[5] C. K. Chua and K. F. Leong. *3D Printing and Additive Manufacturing : Principles and Applications (with Companion Media Pack) Fourth Edition of Rapid Prototyping*. World Scientific Publishing Co., Singapore, 2015.

[6] G. Hodgson. Slic3r manual, 2015.

[7] H. Lipson and M. Kurman. *Fabricated: The New World of 3D Printing*. John Wiley and Sons, New York, 2013.

[8] R. I. Noorani. *Prototyping: Principles and Applications*. John Wiley and Sons, New York, 2006.

[9] G. Ratajc. *Priprava tridimenzionalnega modela za tiskanje s 3D-tiskalnikom*. Diplomsko delo : Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 2005.

[10] D. Systémes. Solidworks help, 2015.

[11] M. Vaezi, H. Seitz, and S. Yang. A review on 3d micro-additive manufacturing technologies. *The International Journal of Advanced Manufacturing Technology*, 67(5-8):1721–1754, July 2013.

[12] Wikidot. G-code guide, 2015.

Nastavljanje parametrov regulatorja z optimizacijskimi algoritmi

Dušan Fister
Univerza v Mariboru
Fakulteta za strojništvo
Smetanova 17, Maribor
dusan.fister@student.um.si

Riko Šafarič
Univerza v Mariboru
Fakulteta za elektrotehniko, računalništvo in informatiko
Smetanova 17, Maribor
riko.safaric@um.si

Iztok Jr. Fister
Univerza v Mariboru
Fakulteta za elektrotehniko, računalništvo in informatiko
Smetanova 17, Maribor
iztok.fister1@um.si

Iztok Fister
Univerza v Mariboru
Fakulteta za elektrotehniko, računalništvo in informatiko
Smetanova 17, Maribor
iztok.fister@um.si

POVZETEK

Vsak regulacijski sistem za optimalno delovanje zahteva pravilno nastavitev vhodnih parametrov. Uporabnik tega sistema strmi k temu, da bo njegov sistem deloval čim bolj optimalno. Za industrijske robote to pomeni, da sistem zagotavlja varnost in funkcionalnost. Velikokrat uvrščamo funkcionalnost pred varnost, kar povzroča nevšečnosti v praksi. Sami želimo robotu optimalno nastaviti regulacijske parametre tako, da bo zagotavljal tako varnost kot tudi funkcionalnost. Čeprav je naš robot miniaturna različica industrijskega robota, tudi ta za pravilno delovanje zahteva dovolj kakovostne nastavitve regulacijskih parametrov. Cilj naše študije je za iskanje optimalnih regulacijskih parametrov razviti algoritem po vzoru obnašanja netopirjev in ga uspešno uporabiti v praksi.

Ključne besede

regulacije, mehatronika, optimizacijski algoritmi

1. UVOD

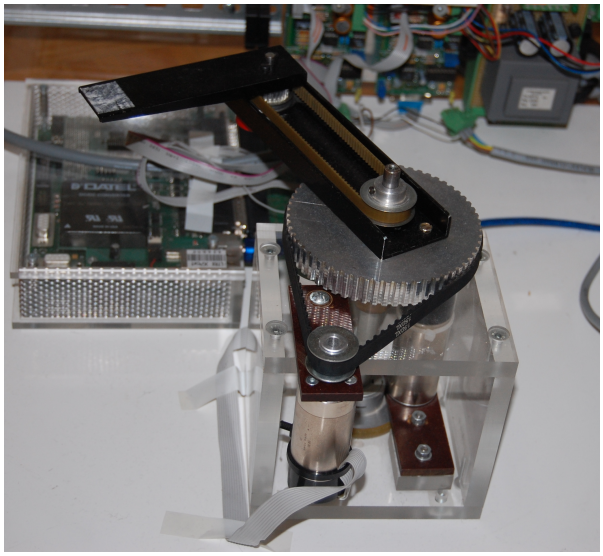
Optimizacijske algoritme uporabljamo za povečanje produktivnosti, storilnosti, kakovosti in hitrosti dela robotskih strojev. Vsak robot mora skozi dobo obratovanja delovati zanesljivo, varno ter funkcionalno. Pričakovati je, da bo robot zaradi povečanja produktivnosti deloval karseda hitro, vendar zaradi tega ne bo ogrožal varnosti ljudi v njegovi bližini. Pomembni faktor za delovanje robotskih mehanizmov predstavlja regulator, tj. element, ki zmanjšuje napako, oz. razliko med želeno in dejansko vrednostjo. Poznamo več vrst regulatorjev, od najpreprostejšega P-regulatorja, ki omogoča le ojačanje napake, do nekoliko kompleksnejših PI- in PD-regulatorjev, do najkompleksnejšega PID-regulatorja. V kraticah imen teh regulatorjev pomeni I-člen integriranje in D-člen diferenciranje napake. Da zagotovimo pravilno nastavljen regulator, moramo pravilno nastaviti vsak posamezen člen posebej. Robot optimalno deluje šele, ko so glede na njegove zahteve dela optimalno nastavljeni vsi členi regulatorja. Enačba (1) matematično prikazuje regulacijske člene in njegove nastavljive parametre, tj.

$$u = K_P(y_{zel} - y_{dej}) + K_I \int (y_{zel} - y_{dej}) + K_D(y_{zel} - y_{dej})', \quad (1)$$

kjer y_{zel} predstavlja želeno referenčno vrednost, medtem ko y_{dej} dejansko vrednost prenihaja. Naš uporabljen regulator, opisan z enačbo (1), sestoji iz izključno PI-člena, kar pomeni, da D-člen v našem primeru odpade.

Najpreprostejši način nastavljanja parametrov je ročno nastavljanje, ki pa zahteva izkušenega tehnika in obilo potrpežljivosti. Obstajajo različne avtomatske metode nastavljanja, kjer največkrat v te namene uporabljamo optimizacijske algoritme po vzoru iz narave. Korenine teh algoritmov segajo v leto 1871, ko je Charles Darwin objavil znanstveno delo o naravni selekciji [1]. Ta je navdihnila Alana Turinga pri razvoju t.i. *genetskega iskanja*, ki uporablja Darwinovo evolucijsko teorijo pri reševanju optimizacijskih problemov [14]. Leta 1988 je John Holland algoritem s selekcijo in mutacijo poimenoval *genetski algoritem*, kar se je ohranilo vse do danes [7]. Podrobnejši zgodovinsko pregled je naveden v [4].

Z robotom SCARA (angl. Selective Compliance Assembly Robot Arm) [12], prikazanim na sliki 1, se je prvi začel ukvarjati Albin Jagarinec. Leta 2005 je uspešno razvil adaptivni regulator [8], medtem ko je leto kasneje Marko Kolar preizkusil algoritem z mehko logiko [11]. Jure Čas se je ukvarjal z zveznim nevronskim sliding-mode regulatorjem [15], medtem ko je Tomaž Slanič iskal optimalne nastavitve parametrov omenjenega robota z genetskim algoritmom [13]. V tem prispevku opisujemo avtomatsko nasta-



Slika 1: Dvoosni robot.

vljanje parametrov PI-položajnega regulatorja na dvoosnem robotu SCARA z algoritmom po vzoru obnašanja netopirjev (angl. Bat Algorithm, krajše BA) [18]. Slednji obljublja enostavno implementacijo algoritma za reševanje realnega problema in hkrati ponuja hitro konvergenco rešitev. Gre za novejšega predstavnika algoritmov iz družine inteligence rojev (angl. Swarm Intelligence), ki za svoje delovanje uporabljajo biološko-socialno obnašanje insektov (npr. roji čebel, družine mravelj, ipd.) oz. živali (npr. jate ptic, rib, ipd.) [10].

2. REGULACIJSKA PROGA

Glavnino regulacijske proge predstavlja dvoosni robot SCARA, ki pooseblja gibanje človeške roke okoli rame in komolca, za kar skrbita dva enosmerna motorja. H glavni gredi vsakega motorja je prigrajen tudi inkrementalni merilnik položaja, ki služi kot dajalnik povratnega signala. Za učinkovito gibanje sklopa skrbita gonilnika motorjev, ki združujeta položajni regulator, komparator in smerni diskriminator za identifikacijo signalov iz inkrementalnega dajalnika ter ojačevalnik referenčnega signala. Slednjega določa vmesnik med računalnikom in robotom, katerega imenujemo DSP-2 Roby kartica [16]. Razvita je bila na UM, FERI in je neposredno povezana z računalnikom, od koder pridobiva zahtevane podatke in kamor pošilja povratne informacije. Optimizacija zato v celoti teče na računalniku.

2.1 Model robota

Regulacijska proga robota je nelinearna, saj model robota zapišemo kot člen drugega reda [17]. Pomeni, da običajni metodi nastavljanja regulatorja, npr. Bodejeva metoda [9] in krivulja lege korenov [3] pri načrtovanju nista več uporabni Enačba (2) predstavlja model robota.

$$\begin{bmatrix}\tau_{mot1}\\ \tau_{mot2}\end{bmatrix} = \begin{bmatrix} J_{m1}n_1 + \frac{a_1+2a_2cos(q_2)}{n_1} & \frac{a_3+a_2cos(q_2)}{n_1} \\ \frac{a_3+a_2cos(q_2)}{n_2} & J_{m2}n_2 + \frac{a_3+J_{3o}}{n_2}\end{bmatrix} \cdot \begin{bmatrix}\ddot{q_1}\\ \ddot{q_2}\end{bmatrix} + 2 + \begin{bmatrix} \frac{-a_2\dot{q_2}(2\dot{q_1}+\dot{q_2})sin(q_2)}{n_1} \\ \frac{a_2sin(q_2)\dot{q_1}^2}{n_2}\end{bmatrix} = \begin{bmatrix} a & b \\ c & d\end{bmatrix} \cdot \begin{bmatrix}\ddot{q_1}\\ \ddot{q_2}\end{bmatrix} + \begin{bmatrix}e\\ f\end{bmatrix} \quad (2)$$

Osnovni parametri za izračun modela robota so dolžine l, mase m in težiščni vztrajnosti momenti J posameznih osi ter gonil, medtem ko dodatni členi predstavljajo položaj q, hitrost $\dot{q}$ in pospešek osi $\ddot{q}$.

Optimizacijski problem je definiran kot $OP = \{I, S, f, goal\}$, kjer I predstavlja množico vseh nalog $\mathbf{x} \in I$, ki se lahko pojavijo na vhodu, S je množica dopustnih rešitev $\mathbf{x} \in S$ in f kriterijska funkcija. *goal* določa, ali kriterijsko funkcijo minimiziramo ali maksimiziramo. Vhod v optimizacijski problem je podan z dvema dvojicama parametrov:

$$\mathbf{x} = \{Kp_1, Kp_2, Kv_1, Kv_2\}, \quad (3)$$

kjer parametra Kp_1 in Kp_2 označujeta velikost proporcionalnega dela (P-člena) prve oz. druge osi, ter Kv_1 oz. Kv_2 velikost integralnega dela (I-člena) prve oz. druge osi.

Med optimizacijskim postopkom pri znanem modelu in znanem izhodu iščemo optimalne vhodne spremenljivke [2]. Pri tem izhodne spremenljivke razvrstimo v tri različne kategorije, tj.:

- $Over_i$ - dejanski prenihaj posamezne osi,
- Ess_i - statični pogrešek posamezne osi in
- $Time_i$ - nastavitveni čas posamezne osi.

Vse tri izhodne spremenljivke pridobivamo neposredno iz kartice DSP-2 Roby, medtem ko jih na računalniku dodatno obdelamo in uporabljamo za vrednotenje kakovosti vhodnih podatkov. Ti predstavljajo osnovo za vrednotenje kakovosti poljubnega odziva na stopnično funkcijo. Robotu tako podajamo navodila za delo, npr. premakni obe osi za določen kot. Robot poskuša doseči najhitrejši premik ob upoštevanju zahtevanega prenihaja, minimalnega statičnega pogreška in minimalnega nastavitvenega časa. Zahtevan prenihaj vnese uporabnik programa pred začetkom optimizacije.

Vrednotenje, oz. ocenjevanje odziva in posledično kakovosti izhodnih podatkov (posredno vhodnih) izvaja ocenjevalna funkcija (angl. objective function), prikazana v enačbi (4):

$$f_i = E_{1i}\cdot(1-|P_i - Over_i|)+E_{2i}\cdot(1-Time_i)+E_{3i}\cdot(1-Ess_i), \quad (4)$$

ki jo maksimiziramo. Zaradi medsebojno mehanske sklopljenosti dvoosnega robota večkriterijske optimizacije ne moremo vršiti. Zato uporabimo uteženo vsoto kriterijev, ki utežijo posamezen parameter tako, da na koncu znaša maksimalna vrednost (popolni odziv) ocenjevalne funkcije $f_i = 1$. Bližje enici torej smo, kakovostnejše rezultate pridobivamo.

3. ALGORITEM PO VZORU OBNAŠANJA NETOPIRJEV

Algoritem po vzoru obnašanja netopirjev (angl. Bat Algorithm, krajše BA) je novejši predstavnik optimizacijskih algoritmov po vzoru iz narave. Nastal je leta 2010 pod okriljem matematika Xin-She Yanga. Algoritem temelji na sposobnosti navigacije netopirjev v popolni temi. Netopirji predstavljo eno redkih bioloških vrst, ki se v naravi orientirajo s pomočjo t.i *eholokacije*. Pojav označuje periodično oddajanje kratkih ultrazvočnih pulzov, odbijajočih se od plena ali

ovire in merjenjem časa odmeva. Posledično lahko ob znani hitrosti potovanja zvoka po zraku določimo oddaljenost od plena ali ovire.

Obnašanje netopirja lahko zapišemo z matematičnim modelom, ki temelji na naslednjih osnovnih pojmih:

- dejanski frekvenci oddajanja Q_i,
- dejanski hitrosti leta $\mathbf{v}_i^{(t)}$ in
- dejanskemu položaju netopirja $\mathbf{x}_i^{(t)}$,

medtem ko naključni let netopirja zapišemo kot kombinacijo treh enačb:

$$Q_i = Q_{min} + (Q_{min} - Q_{min}) \cdot \beta, \tag{5}$$

$$\mathbf{v}_i^{(t+1)} = \mathbf{v}_i^{(t)} + \left[\mathbf{x}_i^{(t)} - \mathbf{x}_{best}^{(t)}\right] \cdot Q_i, \tag{6}$$

$$\mathbf{x}_i^{(t+1)} = \mathbf{x}_i^{(t)} + \mathbf{v}_i^{(t)}, \tag{7}$$

kjer Q_{max} in Q_{min} predstavljata zgornjo in spodnjo mejo frekvence inicializiranih ob zagonu, krmilni parameter β je naključno število generirano v intervalu $[0, 1]$ in $\mathbf{x}_{best}^{(t)}$ trenutna najboljša rešitev.

Proces iskanja v algoritmu BA je odvisen od dveh procesov, tj. preiskovanja (angl. exploration) in izkoriščanja (angl. exploitation). Preiskovanje je močnejše zlasti ob začetku optimizacije, ko so položaji posameznikov razmetani naključno po preiskovalnem prostoru. Izkoriščanje pomeni, da začne preiskovalni proces izboljševati najdeno rešitev običajno z metodami lokalnega iskanja. Vsak posameznik se pri preiskovanju prostora giba v smeri trenutne najboljše rešitve, kar v naravi pomeni, da netopir oddaja ultrazvočne pulze visokih glasnosti A_i, ki posledično potujejo dlje. Te pulze oddaja le nekajkrat v sekundi, npr. osem do desetkrat, kar definiramo s parametrov emisija pulzov r_i. Ko netopir opazi plen, se mu začne približevati, glasnost začne upadati, medtem ko emisija pulzov raste. V tej fazi začne preiskovalni algoritem preiskani prostor rešitev izkoriščati. Matematično obstajata torej dva računska postopka za opis obeh komponent preiskovalnega proces, tj. preiskovanje je zapisano v enačbah (5)-(7), medtem ko izkoriščanje sledi zapisu v enačbi (8):

$$\mathbf{x}_{novi} = \mathbf{x}_{stari} + \epsilon \cdot \bar{A}^{(t)}. \tag{8}$$

Glasnost A_i in emisija pulzov r_i se sicer v originalnem algoritmu spreminjata, vendar se je za potrebe naše optimizacije izkazalo najbolje, da oba parametra nastavimo fiksno in s tem uravnavamo mejo med procesoma preiskovanja in izkoriščanja. S tem dosežemo nekoliko nižjo konvergenco v začetnih generacijah. Oba parametra sta v originalnem algoritmu odvisna od funkcijskih predpisov, ki pa začetno vrednost parametra v nekaj generacijah fiksirata na konstantne vrednosti, zato se parametra v poznejših generacijah obnašata podobno, kot če bi ju fiksirali. Vse omenjene enačbe lahko strnemo v optimizacijski algoritem, katerega psevdokod je prikazan v algoritmu 1. Algoritem začnemo z inicializacijo naključnih posameznikov. Populacijo ovrednotimo ter poiščemo posameznika z najkakovostnejšo rešitvijo.

Algoritem 1 Algoritem na osnovi obnašanja netopirjev

Vpis: Populacija netopirjev $\mathbf{x_i} = (x_{i1}, \ldots, x_{iD})^T$ za $i = 1 \ldots Np$, MAX_FE.
Izpis: Najboljša rešitev $\mathbf{x}_{best}$ in njena vrednost $f_{max} = \max(f(\mathbf{x}))$.

```
1: init_bat();
2: eval = vrednoti_novo_populacijo;
3: f_max = išči_najboljšo_rešitev(x_best);
4: while termination_condition_not_met do
5:   for i = 1 to Np do
6:     y = generiraj_novo_rešitev(x_i);
7:     if rand(0,1) < r_i then
8:       y = izboljšaj_najboljšo_rešitev(x_best)
9:     end if{ lokalno iskanje }
10:    f_new = vrednoti_novo_rešitev(y);
11:    eval = eval + 1;
12:    if f_new ≤ f_i and N(0,1) < A_i then
13:      x_i = y; f_i = f_new;
14:    end if{ shrani najboljšo rešitev pogojno }
15:    f_max=išči_najboljšo_rešitev(x_best);
16:  end for
17: end while
```

Slednji ima najvišjo vrednost ocenjevalne funkcije in predstavlja trenutno najboljšo globalno rešitev. Jedro algoritma predstavlja globalna zanka, ki se izvaja dokler ne presežemo maksimalnega števila iteracij, oz. dovolj kakovostne predpisane rešitve. V tej zanki iz vsakega posameznika generiramo novo rešitev s pomočjo preiskovanja. Glede na verjetnost, da emisija pulzov pade pod določeno mejo če je ta dovolj blizu rešitve pa nadaljujemo tudi na proces izkoriščanja, imenovanega tudi lokalno iskanje. Novo pridobljeno rešitev shranimo le, če je glasnost oddajanja ultrazvočnih pulzov dovolj majhna, kar pomeni, da se netopir nahaja blizu plena. V nasprotnem primeru rešitev zavržemo.

Algoritem išče optimalne parametre za nastavitev regulatorja v omejenem definicijskem območju, ki preprečuje nestabilno delovanje robota. Po vsakem izračunanem premiku novo nastale parametre vpiše v regulator ter počaka na izvršitev le-teh. Kriterijska funkcija vsako dvojico parametrov ovrednoti ter rezultat posreduje algoritmu. Vsako ovrednotenje zaradi mehanskih omejitev traja natančno deset sekund, zato za nas hitrost izvajanja algoritma ni pomembna. Vsekakor pa s povečevanjem števila posameznikov ter generacij premo sorazmerno povečujemo tudi čas potreben za optimizacijo.

4. REZULTATI

V tem poglavju predstavljamo rezultate pridobljene na realni laboratorijski aplikaciji. Zaradi stohastičnosti algoritma je posnetih več odzivov na stopnično funkcijo, čeprav se v praksi uporablja le osnovno testiranje, ko deluje robot brez prenihaja. Rezultati so bili posneti s pomočjo orodja MATLAB/Simulink ter vmesniške kartice DSP-2 Roby. Prikazani rezultati veljajo za dvoosnega robota ($n = 2$). Krmilni parametri algoritma BA so bili nastavljeni med eksperimenti na naslednje vrednosti:

- število posameznikov $N_p = 10$,
- število iteracij $n_{gen} = 10$,
- emisijo pulzov $r_i = 0.1$,
- glasnost oddajanja pulzov $A_i = 0.9$ in

- faktor skaliranja (frekvenca) $Q_i = \{0.5, 1.5\}$.

V sklopu eksperimentalnega dela smo ozvedli tri teste, ki se med seboj razlikujejo po različnih vrednostih zahtevanih prenihajev:

- prvo testiranje: $P_1 = 0$ %, $P_2 = 0$ %,
- drugo testiranje: $P_1 = 10$ %, $P_2 = 10$ % in
- tretje testiranje: $P_1 = 15$ %, $P_2 = 25$ %.

Vsa testiranja so prikazana tabelarično in grafično.

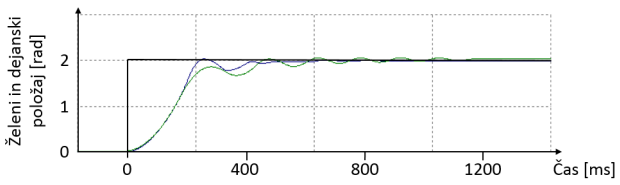
4.1 Prvo testiranje

Prvo in najosnovnejše testiranje je prikazano v tabeli 1 in sliki 2. Vsaka tabela in slika na kratko opisujeta dogajanje po končani optimizaciji, vhodni podatki v realni laboratorijski sistem sta optimizirani dvojici. Velja omeniti tudi zanimivost aplikacije, katero predstavlja povratni gib robota, izveden s preprostim P-regulatorjem za gib nazaj na ničelno točko.

Tabela 1: Prvo testiranje

Izmerjeni rezultati	1. os	2. os
Vrednost ocenjevalne funkcije f_i	0.96516	0.98324
Vrednost prenihaja $Over_i$	0.0187	0
Vrednost statičnega pogreška Ess_i	0.00058	0.00544
Vrednost nastavitvenega časa $Time_i$	0.906	0.504

Testiranje je bilo dokaj uspešno izvedeno, kar dokazuje tudi visoka vrednost povprečne ocenjevalne funkcije. Prenihaj prve osi je bil nastavljen nekoliko nenatančno, medtem ko je prenihaj druge osi natančno zadel zahtevano nastavitev. Statični progrešek je pomemben del robotove natančnosti. Višjo natančnost je dosegel s prvo osjo, saj ta beleži nižjo stopnjo napake. Nastavitveni čas je parameter, ki pove kako robot zaniha blizu rešitve. Daljši kot je nastavitveni čas, slabša je kakovost parametrov. Določa ga ozko tolerančno območje, tj. hitreje kot se robot ustali v tem območju, krajši nastavitveni čas ga odlikuje.



Slika 2: Prvo testiranje

Iz slike 2 je razvidno, da se je robot uspešno premaknil za kot dva radiana, vendar v okolici nekoliko zanihal in povzročil dolg nastavitveni čas, kar se sklada s tabelo 1. Z dodatnim eksperimentalnim nastavljanjem krmilnih parametrov bi lahko nastavitveni čas drastično izboljšali, vendar oteževali preprosto uporabo naše aplikacije. Poleg tega bi bilo potrebno nekajkrat ponastaviti spekter možnih števil iz katerih pridobivamo posamezne člene dvojic parametrov in jih čimbolj približati k optimalnim nastavitvam, kar dodatno zaostruje uporabo aplikacije. Predpostavili smo, da mora biti algoritem in njegova uporaba enostavna in funkcionalna, kar pa nekoliko slabša kakovost posameznih rezultatov. V oči bode tudi podnihaj, ki se pojavi takoj po dosegu zahtevane vrednosti in dodatno dodeljuje manevrski prostor za finejšo obdelavo.

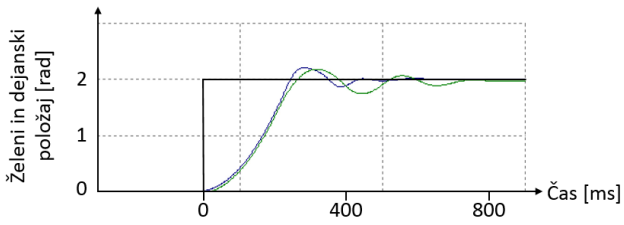
4.2 Drugo testiranje

Drugo testiranje predstavljata tabela 2 in slika 3. Testiranje je bilo izvedeno za zahtevana prenihaja $P_1 = 10$ % in $P_2 = 10$ %. Nastavljanje parametrov se za ta režim v praksi ne uporablja več, mi smo ga uporabili kot dodaten optimizacijski problem za podkrepitev rezultatov.

Tabela 2: Drugo testiranje

Izmerjeni rezultati	1. os	2. os
Vrednost ocenjevalne funkcije f_i	0.9739	0.9854
Vrednost prenihaja $Over_i$	0.10733	0.10237
Vrednost statičnega pogreška Ess_i	0.00579	0.00113
Vrednost nastavitvenega časa $Time_i$	0.714	0.444

Tabela 2 kaže končne rezultate drugega testiranja. Razvidno je, da je v tem primeru nekoliko kakovostneje nastavljena druga os. O tem priča višja vrednost ocenjevalne funkcije. Tudi prenihaja sta v veliki meri natančno načrtana, maksimalna napaka znaša zgolj 0.7 %. Statični pogrešek je nekoliko ohlapneje načrtan v primerjavi s prvim primerom. Ta sicer znaša precej manj za drugo os, medtem ko se je za prvo os enormno povečal. Nastavitveni čas je v obeh primerih krajši, kar dodatno izboljšuje vrednost ocenjevalne funkcije.



Slika 3: Drugo testiranje

Grafično predstavljen odziv dopolnjuje kakovost tabelarično predstavljenih rezultatov. Odziv je iz grafičnega stališča kakovostneje optimiziran kot prvi, največ k temu pripomore krajši nastavitveni čas. Iz fizikalnega stališča je treba omeniti, da so začetna eksperimentalna testiranja krmilnih parametrov potekala prav na tem režimu delovanja - $P_1 = 10$ % in $P_2 = 10$ %. Pomeni, da so ti parametri najugodnejši za opisovano testiranje, medtem ko za druga testiranja ne veljajo več v popolni meri. To pa je prednost in obenem slabost nastavljanja krmilnih parametrov.

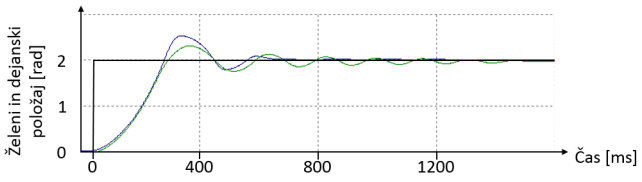
4.3 Tretje testiranje

Tretje testiranje poosebljata tabela 3 in slika 4, zahtevana kombinirana prenihaja znašata $P_1 = 15$ % in $P_2 = 25$ %. Tabelarični rezultati tretjega testiranja prinašajo glede na vrednost ocenjevalne funkcije pozitivne lastnosti, vendar zahtevajo glede na natančnost načrtanega prenihaja in

Tabela 3: Tretje testiranje

Izmerjeni rezultati	1. os	2. os
Vrednost ocenjevalne funkcije f_i	0.95527	0.97839
Vrednost prenihaja $Over_i$	0.16721	0.25872
Vrednost statičnega pogreška Ess_i	0.00202	0.00027
Vrednost nastavitvenega časa $Time_i$	1.222	0.606

trajanja nastavitvenega časa prve osi dodatno optimizacijo. Kakovostno so načrtani vsi izhodni podatki za drugo os. Pojav dolgega nastavitvenega časa in posledično nižje oce-



Slika 4: Tretje testiranje

njevalne funkcije prikazuje tudi slika 4. Čeprav je druga os uspešno nastavljena, zaradi velikega prenihaja in vztrajnostnih mas vpliva na prvo os. Znano je namreč, da sta obe osi medsebojno mehansko sklopljeni, kar posledično zmanjšuje kakovost pridobljenih podatkov druge osi.

5. SKLEP

Z našo aplikacijo smo hoteli pokazati, da so optimizacijski algoritmi po vzoru iz narave, konkretneje algoritem BA, primerni za nastavljanje parametrov regulatorjev. Algoritem BA, ki je znan zaradi svoje enostavne implementacije ter hitre konvergence, poleg zvezne optimizacije matematičnih modelov ponuja oporo tudi takim vrstam problemov. Četudi regulator v vseh treh primerih ni bil optimalno nastavljen, izstopajo namreč rezultati prve osi, smo bili objektivno z dobljenimi rešitvami zadovoljni. Vsekakor bi s povečanjem števila posameznikov v populaciji ter števila iteracij lahko pričakovali kakovostnejše rezultate, vendar bi po drugi strani povečali potreben optimizacijski čas.

Rezultati poskusov so pokazali, da najbolj izstopa dolg nastavitveni čas prve osi, saj ta predstavlja nosilno oporo tudi drugi osi. S hitrim spreminjanjem položaja in vztrajnostnih momentov pa slednja vpliva na prvo, kar predstavlja glavno mehansko težavo. Težavo bi lahko rešili z manjšanjem regulacijskega I-člena, oz. ponovno optimizacijo z ožjim naborom razpoložljivih vrednosti tega člena (Kv_1).

V prihodnje želimo algoritem BA hibridizirati s strategijami diferencialne evolucije in tako izboljšati rezultate optimizacije [6]. Trenutno so v teku tudi testiranja genetskega algoritma (GA), optimizacije z roji delcev (PSO) in diferencialne evolucije (DE). Dodatno izboljšavo predstavlja adaptacija krmilnih parametrov, kjer postopek ročnega nastavljanja parametrov avtomatiziramo ter s tem eliminiramo predčasno konvergenco v lokalne optimume [5]. Prav ta lastnost predstavlja glavno težavo omenjenega algoritma, saj optimizacija teče dokler izboljšujemo trenutno najboljšo najdeno rešitev.

6. REFERENCES

[1] C. Darwin. R.(1859): On the origin of species by means of natural selection. *Murray. London*, 1871.

[2] A. E. Eiben and J. E. Smith. *Introduction to evolutionary computing*. Springer Science & Business Media, 2003.

[3] W. R. Evans. Control system synthesis by root locus method. *American Institute of Electrical Engineers, Transactions of the*, 69(1):66–69, 1950.

[4] D. Fister. *Načrtovanje samonastavljivega regulatorja 2 DOF robota s pomočjo BA algoritma*. Diplomsko delo : Univerza v Mariboru, Fakulteta za strojništvo, 2015.

[5] I. Fister, D. Strnad, X.-S. Yang, and I. Fister Jr. Adaptation and hybridization in nature-inspired algorithms. In *Adaptation and Hybridization in Computational Intelligence*, pages 3–50. Springer, 2015.

[6] I. Fister Jr, D. Fister, and X.-S. Yang. A hybrid bat algorithm. *arXiv preprint arXiv:1303.6310*, 2013.

[7] D. E. Goldberg and J. H. Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[8] A. Jagarinec. *Adaptivni regulator z mehko logiko za dvoosni SCARA mehanizem*. Diplomsko delo : Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 2005.

[9] L. H. Keel and S. P. Bhattacharyya. A bode plot characterization of all stabilizing controllers. *Automatic Control, IEEE Transactions on*, 55(11):2650–2654, 2010.

[10] J. Kennedy, J. F. Kennedy, R. C. Eberhart, and Y. Shi. *Swarm intelligence*. Morgan Kaufmann, 2001.

[11] M. Kolar. *Vodenje SCARA robota z mehko logiko*. Diplomsko delo : Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 2005.

[12] H. Makino, N. Furuya, K. Soma, and E. Chin. Research and development of the scara robot. In *Proceedings of the 4th International Conference on Production Engineering*, pages 885–890, 1980.

[13] T. Slanič. *Genetski regulator za dvoosnega SCARA robota*. Diplomsko delo : Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 2006.

[14] A. M. Turing. Intelligent machinery, a heretical theory. *The Turing Test: Verbal Behavior as the Hallmark of Intelligence*, page 105, 1948.

[15] J. Čas. *Izdelava zveznega nevronskega sliding-mode regulatorja za teleoperiranje SCARA robota*. Diplomsko delo : Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 2006.

[16] M. Čurkovič. *Vgrajeni sistemi DSP/FPGA v sistemih vodenja*. Magistrsko delo Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 2010.

[17] R. Šafarič and A. Rojko. *Inteligentne regulacijske tehnike v mehatroniki*. Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 2005.

[18] X.-S. Yang. A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pages 65–74. Springer, 2010.

Analiza sentimenta z uporabo emotikonov in besednih zvez

Tadej Jerovšek
Fakulteta za elektrotehniko računalništvo in informatiko
Smetanova ulica 17, 2000 Maribor
Maribor, Slovenija
tadej.jerovsek@student.um.si

Tadej Ganza
Fakulteta za elektrotehniko računalništvo in informatiko
Smetanova ulica 17, 2000 Maribor
Maribor, Slovenija
tadej.ganza@student.um.si

Tomaž Cebek
Fakulteta za elektrotehniko računalništvo in informatiko
Smetanova ulica 17, 2000 Maribor
Maribor, Slovenija
tomaz.cebek@student.um.si

Martin Kraner
Fakulteta za elektrotehniko računalništvo in informatiko
Smetanova ulica 17, 2000 Maribor
Maribor, Slovenija
martin.kraner1@student.um.si

Borko Bošković
Fakulteta za elektrotehniko računalništvo in informatiko
Smetanova ulica 17, 2000 Maribor
Maribor, Slovenija
borko.boskovic@um.si

POVZETEK

V članku smo obravnavali analizo sentimenta kratkih sporočil popularnega spletnega družbenega omrežja Tweeter[1], oz. "tweetov". Sentimentalno analizo smo izvajali nad lematiziranimi besedami sporočila z uporabo ustaljenih metod. Ker so sporočila kratka (omejena na 140 znakov), je sentiment težko oceniti. Da bi dobili več informacij, smo v algoritem vključili emotikone ("hashtag"), ter bazo fraz in besednih zvez, ki pri analizi po besedah dajejo napačen rezultat. Posebej smo analizirali sentiment besedila in emotikonov in ga nato s pomočjo uteži združili v eno končno oceno. Predstavljen algoritem smo primerjali z algoritmom iz literature v večih kategorijah, kot so samo analiza besedila, analiza z emotikoni ter naša izboljšana rešitev. Dobljeni rezultati kažejo, da je naš algoritem dosegel za 1,35% boljšo oceno metrike F1.

Kategorije in opisi teme

H.4 [**Information Systems Applications**]: Miscellaneous; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Linguistic processing*
; I.2.7 [**Artificial intelligence**]: Natural Language Processing—*Language parsing and understanding, Text analysis*

Splošni pojmi

Therory
Languages
Experimentation

[1]Dostopno na spletnem naslovu `http://www.tweeter.com`

1. UVOD

Ljudje v digitalni dobi vse pogosteje objavljajo svoje misli in kaj počnejo na javnih spletnih mestih. Med temi informacijami podajo svoje mnenje glede določenih dogodkov, izdelkov, politikov in podobno. Analiza sentimenta se ukvarja z ekstrakcijo teh informacij, kar omogoča hitro in lahko ugotavljanje mnenja velikega števila ljudi glede določene tematike. Primer tega so lahko tudi možnosti zmage na volitvah, kar se v času volitev na medijih "prodaja" kot med.

Pri tem je več pristopov, ki pa v veliki večini temeljijo na besedilni analizi. Slabost tega pristopa je, da je praktično nemogoče ugotoviti informacije, ki bi jih v naravnem govoru sporočevalec podal s telesnim govorom. Tako kaj hitro napačno razvrstimo sentiment, ko govorec uporabi sarkazem ali kakšno drugo retorično obliko.

Kot dopolnilo klasičnemu pisanju je profesor Scott Fahlman leta 1982 poskusil uporabiti prva emotikona ":-)"in ":-(". S tema je predlagal, da bi pisec izražal razliko med resno mislijo, ali šalo. Ta koncept je hitro postal zelo priljubljen in se je razširil po celotnem svetovnem spletu, kjer danes emotikoni predstavljajo splošen koncept izražanja.

Zaradi uporabe emotikonov, je mogoče iz kratkih sporočil izvedeti precej bolj podrobno informacijo o sentimentu kakor iz klasičnih besedilnih sporočil, vendar je na tem področju malo poudarka.

Tabela 1: Označevanje emotikonov

Simbol	Določen sentiment	Vrednost
xS >.> :#	Negativno	-1
:'(:@ :[	Negativno	-1,5
l8^O >:-) >//<	Pozitivno	1
:-) ♥_♥ :D	Pozitivno	1,5
<:3( ) >—^*<	Neznano (odstranjeno)	/

2. SORODNA DELA

V članku [6] je predstavljen model, ki združuje ročno označene podatke s šumnimi podatki o emotikonih. Uporabili so model učen na ročno označenih podatkih, za glajenje pa so bili uporabljeni šumni podatki.

V [4] so sentiment izačunavali iz tweetov in forum sporočil, kjer so imeli slovar besed z označenimi sentimenti ter tudi ročno označene določene emotikone. Sentiment so določali glede na odstavke, kjer so sentiment utežili s prisotnimi emotikoni tako, da je sentiment emotikona prevladal nad sentimentom besedila (pozitivno ali negativno glede na razliko med sentimentom besedila in sentimentom emotikonov).

3. POSTOPEK ANALIZE

V tem poglavju opisujemo naš algoritem za analizo sentimenta. Najprej bomo opisali obdelavo emotikonov, saj smo le te uporabili v koraku pridobivanja besedil s Twitterja. Nato sledi opis algoritma za predobdelavo in analizo sentimenta. Na koncu podajamo še uporabljen način združevanja sentimentalnih analiz v skupno oceno.

3.1 Izgradnja baze sentimentalno označenih emotikonov

V namen izboljšane analize sentimenta smo ustvarili bazo emotikonov, ki imajo označen sentiment. Našli smo 6 obstoječih baz, ki se nahajajo na spletu [3, 7, 8, 9, 11, 12]. Uporabili smo format XML, zaradi potrebe po čim lažji prenosljivosti. Združili smo vseh šest baz, pri čemer smo podvojene emotikone odstranili. Vse baze niso imele označenih sentimentalnih vrednosti, zato smo vse emotikone v bazi tudi ročno označi, pri čemer smo uporabili za negativni sentiment naslednje vrednosti: **-1,5** in **-1,0**, za emotikone, ki se ne morejo pojavit v kontekstu s pozitivnim sentimentom **-1,5** in za emotikone s pozitivnim sentimentom, **1,5** in **1,0**. Primer označenih emotikonov je naveden v tabeli 1.

Iz obstoječih baz smo dobili **1411** emotikonov, katere smo ročno označili. Iz te baze smo odstranili **465** emotikonov, ki ne nosijo sentimentalne vrednosti. Odstraniti smo še emotikone, katerih se ne da uporabit v korpusih od Twitterja. Problem namreč nastane pri emotikonih, ki se pričnejo z znakom "@", saj se tako označujejo osebe, na katere se pisec sklicuje v tweetu, s čimer smo odstranili nadaljnjih **31** emotikonov. V končni bazi smo tako dobili **919** emotikonov, ki so bili primerni za korpuse iz Twitterja in so nosili ročno označen sentiment.

V okviru tega dela smo se osredotočili le na sporočila, ki vsebujejo emotikone. V vseh sporočilih smo iskali emotikone in odstranili tista sporočila, ki niso vsebovala emotikonov, ali so vsebovala emotikone katerim se ni dala določiti sentimentalna vrednost.

Za iskanje smo uporabili regularne izraze (angl. regular expresion). Najprej smo uredili emotikone tako, da smo najprej imeli najdaljše, nato pa jih zaporedno dodajali v regularni izraz. S tem smo zagotovili, da se v primeru, da se začnejo na enako črko, najdejo tudi krajši. Najdene emotikone smo omejili tudi glede na predhodni znak. Veliko emotikonov je namreč sestavljenih tako, da se lahko zamenja tudi del normalne besede kot emotikon. To smo rešili tako, da smo za pogoj postavili, da mora biti emotikon na začetku vrstice, ali mora biti predhodni znak del drugega emotikona, ali pa mora biti predhodni znak nečrkovni.

$$"((:DD)|(:D)|(:\))|(:S))"$$

Primer regularnega izraza za iskanje emotikonov.

Za ocenjevanje sentimenta emotikonov v sporočilu, smo poiskali njihove ocene sentimenta v bazi, ter nato sešteli njihove vrednosti. To nam je omogočilo natančnejši izračun verjetnosti sentimentalne vrednosti in prav tako izločiti sporočila z enakomernim pozitivnim in negativnim sentimentom oz., odstraniti nevtralna sporočila.

Za namene sentimentalne analize smo pripravili korpus, kjer so bili vsi emotikoni odstranjeni, da smo se izognili napakam pri lematizaciji oz. določevanju sentimenta besedila.

3.2 Pridobivanje tweetov

Sama sporočila družabnega omrežja Twitter smo pridobivali s pomočjo programskega vmesnika (angl. API), ki je namenjen za razvijalce. Vmesnik deluje po arhitekturi REST (angl. Representational State Transfer) preko spletnega protokola HTTP (angl. Hypertext Transfer Protocol) kjer se podatki prenašajo v formatu JSON. Za to komunikacijo smo uporabili programsko knjižnico CoreTweet [10]. Zaradi omejitev obstoječih orodij za analizo slovenskega jezika smo se omejili samo na tweete angleškega jezika.

Izbrali smo **150** tweeter uporabnikov, ki dokaj pogosto uporabljajo emotikone in od vsakega pridobili **200** najnovejših sporočil (omejitev API-ja). Vsa pridobljena sporočila smo združili v en korpus in dobili **30.000** različnih sporočil, te smo s pomočjo postopka opisanega v prejšnjem poglavju filtrirali tako, da smo obdržali le tista, ki so dejansko vsebovala emotikone z nezanemarljivimi sentimentalnimi utežmi. Dobili smo **3.041** sporočil, ki so vsebovala emotikone. Sentiment teh sporočil so ločeno ocenili **4** ljudje, avtorji tega članka, študenti na magistrskem študiju računalništva. Te vrednosti smo nato združili v eno referenčno oceno.

3.3 Lematizacija besedila

Enolična določitev sentimentalne ocene besedila je le možna, če lahko večini posameznih besed znotraj besedila določimo sentimentalno oceno ne glede na sklanjatev. Postopku spreminjanja besede nazaj v osnovno obliko imenujemo lematizacija oz. krnjenje in besede, ki jih dobimo leme oz. krni.

Ker smo želeli biti neodvisni od spletne povezave smo se odločili za programsko knjižnico za opravljanje tega postopka po imenu LemmaGen [5]. Knjižnica tudi v določenih prime-

Tabela 2: Lematizacija z LemmaGen.

Izvorna beseda	Lematizirano	Brez opozoril
don't	do+not	do not
Don't	Do+not	Do not
they're	theybe	they be

rih opozori na besede oz. besedne zveze na katere moramo paziti s opozorilnim znakom +. S tem želi povedati, da je bila to samo ena beseda oz. besedna zveza pred lematizacijo. To informacijo uporabimo za bolj uspešno oceno sentimenta. Nekateri primeri teh besednih zvez so navedeni v tabeli 2.

3.4 Opravljanje sentimentalne analize besedila

Sentimentalno analizo besedila smo opravili s pomočjo dveh orodij za analizo besedila. Prvo izmed takšnih orodij, je **OpenNLP** [1]. To orodje smo uporabili za določanje tipa besed, uporabljenih v tweetih (angl. Part-of-Speech taging). Drugo orodje, s katerim smo si pomagali pri opravljanju sentimentalne analize besedila, je **SentiWordNet** (SWN) [2]. Sentimentalno analizo smo tako opravili, da smo s pomočjo OpenNLP določili tip besede (npr. pridevnik, glagol...), ker pa OpenNLP omogoča več možnosti klasifikacije, kot pa SWN smo morali klasifikacije OpenNLP združiti v skupine, ki so se ujemale s tipi besed, ki jih SWN pozna.

V tem koraku smo imeli na voljo posamezne besede in tip posamezne besede. S pomočjo teh podatkov je SWN vrnil vektor pozitivnega in negativnega sentimenta iskane besede. Ker pa enačba vektorja SWN dovoljuje tudi izračun objektivnega sentimenta (angl. objective), smo lahko določili tudi objektivnost posameznih besed, kot prikazuje naslednja enačba:

$$1 = poz_i + neg_i + obj_i. \quad (1)$$

V tej enačbi $\mathbf{poz}_i$ pomeni pozitivni sentiment $\boldsymbol{i}$-te besede, $\mathbf{neg}_i$ negativni sentiment $\boldsymbol{i}$-te besede, ter $\mathbf{obj}_i$ prikazuje objektivni sentiment $\boldsymbol{i}$-te besede. Na te vrednosti se bomo v nadaljevanju sklicevali kot na 3-dimenzionalni vektor.

Sentiment posameznega tweeta smo določili s povprečnimi vrednostmi vektorjev besed, ki so se pojavile v tweetu. Kot izboljšavo smo določili prag izrazitosti sentimenta, ki je potreben da smo besedo vključili v izračun vrednosti pozitivnega in negativnega sentimenta tweeta. Prag sentimenta smo izračunali z opazovanjem absolutne razdalje med $\mathbf{poz}_i$ in $\mathbf{neg}_i$ $\boldsymbol{i}$-te besede. V kolikor vrednost ni prekoračila praga, smo upoštevali le njeno objektivno vrednost.

Pridobljeni vektor smo nato normalizirali s pomočjo števila besed, ki so bile uporabljene pri izračunu posamezne vrednosti sentimenta. Končni sentiment besedila smo nato dobili tako, da smo od vrednosti pozitivnega sentimenta odšteli vrednost negativnega sentimenta. Ker pa se ta vrednost nahaja na intervalu med **[-1, 1]** medtem, ko se vrednosti za emotikone nahajajo na intervalu **[-1,5, 1,5]** smo morali končno vrednost skalirati z **1,5**. Tako smo dobili interval, kjer vrednosti večje od **0**, pomenijo pozitiven sentiment, vrednosti nižje od **0** pa pomenijo negativen sentiment.

Tabela 3: Tipični primeri kako uporabimo emotikone za prikaz sentimenta.

Poved	Emotikon	Sentiment
I love my work :-D	Okrepitev	pozitiven
The movie was bad :-D	Negacija	pozitiven
:-D I got a promotion	Samo sentiment	pozitiven
-_- I love my work	Negacija	negativen
The movie was bad -_-	Okrepitev	negativen
I got a promotion -_-	Samo sentiment	negativen

Tabela 4: Pravilnost sentimenta.

Pričakovan	Ocenjen	Pravilnost
> 0,5	> 0,5	Pravilen
< -0,5	< -0,5	Pravilen
> 0,5	< 0,5	Nepravilen
< -0,5	> 0,5	Nepravilen
> -0,5 in < 0,5	/	Neupoštevan

3.4.1 Združevanje sentimenta besedila in emotikonov

Združevanje obeh vrednosti sentimenta smo opravili kot je opisano v [4]. Za združevanje smo predpostavili **6** tipičnih situacij, kot prikazuje tabela 3.

Kot je prikazano v tabeli 3 smo združili vrednosti sentimenta emotikona in besedila glede na ustrezno situacijo. Situacijo smo določili glede na določeno vrednost sentimenta besedila, ter emotikona. V kolikor sta oceni besedila in emotikonov enako predznačeni, vrednost sentimenta okrepimo. V kolikor sta različno predznačeni, vrednost analize sentimenta besedila negiramo. V primeru, da analiza besedila ne poda vrednosti sentimenta, pa vzmamemo samo vrednost analize emotikonov. Kot dodatno izboljšavo smo tudi tukaj uvedli prag minimalne vrednosti sentimenta za posamezno klasifikacijo.

4. REZULTATI EKSPERIMENTA

Najprej smo ročno označili sentimentalne vrednosti vseh tweetov. Učinkovitost smo ocenili s primerjavo ocenjenega sentimenta ter izračunanega. V primerjavi ustreznosti smo ignorirali zaokrožitveno napako, ob ročno določitvi smo se omejili na **5** različnih vrednosti: **{-1,5; -1; 0; 1; 1,5}** medtem ko so se izračunane vrednosti lahko pojavijo kjerkoli v tem intervalu. Iz tega razloga smo preverjali ujemanje sentimenta v predznaku (pozitiven/negativen). Rezutati so prikazali **97,28%** uspešnost po metriki F1. Ob pregledu, smo opazili, da uporabniki uporabljajo besedne zveze s katerimi svoj sentiment bolj izrazito izražajo. Iz stališča stroja te besede kot tudi njihove kombinacije ne nosijo velikega pomena, vendar iz stališča uporabnika pa prikazujejo visok nabor informacij. Izbrali smo **43** tovrstnih besednih zvez, ki so se pojavljale v našem naboru tweetov. Tem besednim zvezam smo določili sentimentalno vrednost, katero smo dodali predhodno izračunani vrednosti. Po ponovnem pregledu so rezultati prikazali **98,63%** uspešnost s čimer smo število nepravilno razvrščenih tweetov zmanjšali skoraj za polovico.

Tabela 5: Tabela uspešnosti zaznave sentimenta.

	Sentiment besedla	Metoda [4]	Naša metoda
TP	447	1181	1191
FP	175	40	17
FN	760	26	16
TN	111	246	269
Σ	1493	1493	1493

4.1 Primerjava algoritmov

V našem eksperimentu bomo primerjali naši algoritem z algoritmom iz [4]. Ker smo uporabili drugačen korpus kakor avtorji omenjenega članka, smo implementirali tudi njihov algoritem, da smo lahko primerjali uspešnost. Za vsako izmed metod smo izračunali natančnost, preciznost, priklic in metriko F1.

Po vrsti smo preiskusili naslednje algoritme:

1. Metoda iz članka brez emotikonov (analiza sentimenta besedila)
2. Metoda iz članka z emotikoni
3. Naša metoda z emotikoni in besednimi zvezami

V eksperimentu smo uporabili bazo **1493** ročno označenih tweetov, ločeno s strani **4**-ih oseb (avtorjev članka), oz. v dveh prehodih, katera smo nato povprečili. Rezultate oznak smo potem združili v eno referenčno oceno sentimenta. Naša uspešnost napram samemu sentimentu besedila in osnovni metodi [4] je prikazana v slikah 1, 2, 3, 4 ter tabelah 5 in 6.

V osnovi smo z analizo samega sentimenta besedila dobili boljše rezultate kakor v [4], kar kaže, da so naša sporočila bila nekoliko bolj jasna za analizo brez emotikonov. Pri tem smo uporabili vrečo besed, kakor v primerjanem delu.

Pri analizi s filtriranjem z emotikoni smo dobili podobne rezultate kakor v [4], vendar ne moremo narediti primerjave zaradi različnih korpusov. Naši rezultati so bili par odstotkov slabši, saj ljudje na Twitterju emotikone uporablajo iz navade, ne z namenom sporočanja svojega počutja. Kljub temu smo delež pravilno klasificiranih več kot podvojili s predlagano metodo.

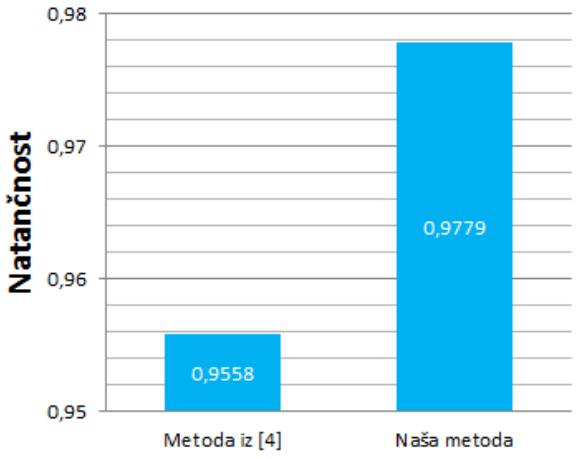
Do sedaj opisane metode so narobe klasificirale **199** sporočil (**11,3%**). Metodi iz [4] namreč ocenjujeta sentiment na podlagi vreče besed, kjer se pa informacija besednih zvez izgubi. Kot nadaljnjo izboljšavo smo poskusili omogočiti pravilno klasifikacijo z informacijo le teh. V napačno razpoznanih smo nato iskali besedne zveze s katerimi smo pri samem označevanju sklepali na nasprotno vrednost sentimenta. Te smo nato uporabili za popravljeno klasifikacijo, pri čemer vsaka besedna zveza spremeni vrednost analize sentimenta za določeno vrednost. Pri tem smo uspeli **199** napačno razpoznanih tweetov znižali na **168** napačno razpoznanih tweetov.

Tabela 6: Tabela rezultatov metrik.

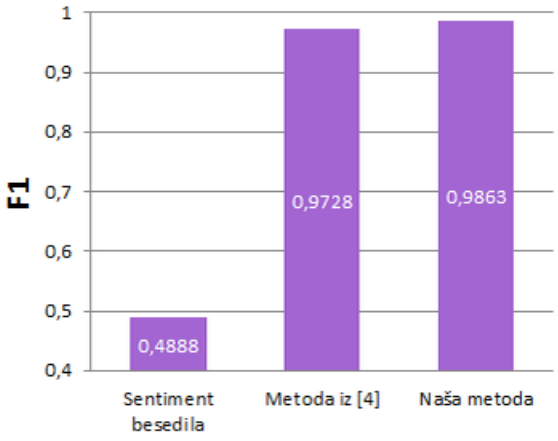
	Sentiment besedla	Metoda [4]	Naša metoda
A	0,2994	0,7910	0,7977
P	0,7186	0,9672	0,9859
R	0,3703	0,9785	0,9867
F1	0,4888	0,9728	0,9863

5. ZAKLJUČEK

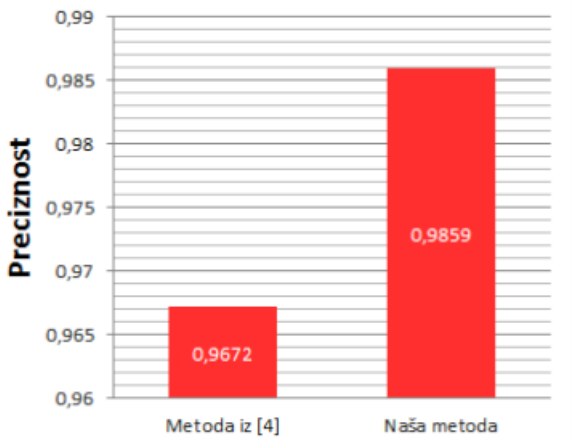
V prispevku smo predstavili algoritem za analizo sentimenta nad objavami na Twitterju. Ta vsebuje več korakov, ki vključuje sentimenta besedila in emotikonov. Ta smo združili v skupno oceno, nato pa smo za primere, kjer obe analizi sentimenta dajeta napačno razpoznavo, dodali še popravke ocene v obliki razpoznavanja besednih zvez, ki spremenijo vrednost združene ocene za absolutno vrednost. Analiza sentimenta besedila je dala najslabše rezultate, analiza z emotikoni je uspešnost več kot podvojila. Dodatno smo prikazali popravljanje združene ocene z besednimi zvezami, s čimer smo pridobili dodatna 1,35% uspešnosti po metriki F1.



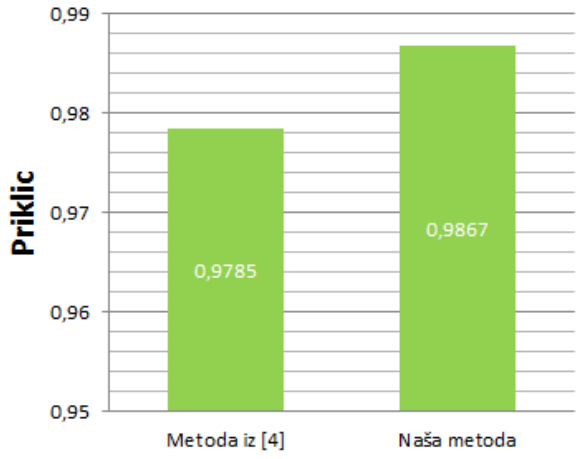
Slika 1: Natančnost klasificiranih kratkih sporočil.



Slika 2: Metrika F1 za uspešnost razpoznav sentimenta z različnimi metodami.



Slika 3: Primerjava preciznosti metod za razpoznavo sentimenta z emotikoni



Slika 4: Primerjava priklica algoritmov za razpoznavo sentimenta z emotikoni.

6. VIRI

[1] Apache Software Foundation. OpenNLP. `http://https://opennlp.apache.org`, Maj 2015.

[2] S. Baccianella A. Esuli, in F. Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. V *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, Maj 2010. European Language Resources Association (ELRA).

[3] Computeruser.com. Emoticon list. `http://www.computeruser.com/resources/dictionary/emoticons.html`, Maj 2015.

[4] A. Hogenboom D. Bal F. Frasincar M. Bal F. de Jong, in U. Kaymak. Exploiting emoticons in sentiment analysis. V *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, strani 703–710, New York, NY, USA, 2013. ACM.

[5] M. Juršič I. Mozetič T. Erjavec, in N. Lavrač. LemmaGen: Multilingual lemmatisation with induced ripple-down rules. *Journal of Universal Computer Science*, 16(9):1190–1214, Maj 2010. `http://www.jucs.org/jucs_16_9/lemma_gen_multilingual_lemmatisation`.

[6] K. Liu W. Li, in M. Guo. Emoticon smoothed language models for twitter sentiment analysis. *AAAI Conference on Artificial Intelligence*, 2012.

[7] T. Marks. Smileyworld emoticons, characters, wallpapers, games, blogs. `http://www.windweaver.com/emoticon.htm`, Marec 2012.

[8] Msgweb.nl. List of emoticons in MSN Messenger. `http://www.msgweb.nl/en/MSN_Images/Emoticon_list`, Maj 2015.

[9] Pc.net. Text-based emoticons. `http://pc.net/emoticons/`, Maj 2015.

[10] T. Sakurai in Y. Sato. Coretweet. `http://coretweet.github.io`, Maj 2015.

[11] M. Thelwall K. Buckley G. Paltoglou D. Cai, in A. Kappas. Sentiment strength detection in short informal text. *J. Am. Soc. Inf. Sci. Technol.*, 61(12):2544–2558, Dec. 2010.

[12] Wikipedia, the free encyclopedia. List of emoticons. `http://en.wikipedia.org/wiki/List_of_emoticons`, April 2015.

MySQL Database On-line Update Technology Based on JSP

DU Xiaocheng[1] **GAO Tao**[1, 2, *] **YANG Yaoquan**[1]

1. Department of Automation, North China Electric Power University, Baoding 071003, Hebei Province, China
2. Knowledge Engineering and Discovery Research Institute, Auckland University of Technology, Auckland 1010, New Zealand

Email: gaotao863@163.com

Abstract With the booming of the network technology, JSP (Java Server Pages) has been widely applied, and it is playing an increasingly important role. This paper introduces the technology of JSP, and expounds the concrete process of MySQL database connection technology by JDBC (Java Data Base Connectivity); finally it presents the methods of operating MySQL database on-line based on JSP.

Keywords JSP, MySQL, JDBC, On-line Update

0 Introduction

Today, the development of network technology is constantly changing, and it has become the main theme in our lives and society. Almost every area of human life is changed due to network. JavaServer Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. JSP is similar to PHP, but it uses the Java programming language. JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, such as HTML, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, contain Java byte code rather than machine code. Like any other Java program, they must be executed within a Java virtual machine (JVM) that interacts with the server's host operating system to provide an abstract, platform-neutral environment.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack. MySQL can be built and installed manually from source code, but this can be tedious so it is more commonly installed from a binary package unless special customizations are required. Though MySQL began as a low-end alternative to more powerful proprietary databases, it has gradually evolved to support higher-scale needs as well. It is still most commonly used in small to medium scale single-server deployments.

JDBC is a Java database connectivity technology (Java Standard Edition platform) from Oracle Corporation. This technology is an API for the Java programming language that defines how a client may access a database. It provides methods for querying and updating data in a database. JDBC is oriented towards relational databases. A JDBC-to-ODBC bridge enables connections to any ODBC-accessible data source in the JVM host environment. JDBC allows multiple implementations to exist and be used by the same application. The API provides a mechanism for dynamically loading the correct Java packages and registering them with the JDBC Driver Manager. The Driver Manager is used as a connection factory for creating JDBC connections.

This paper is mainly focus on the combining technology between MySQL database and JSP by JDBC for the database data online updates, and thus to optimize the data flow in web network technology.

1 The JSP Technology

The fundamental part of JSP (Java Server Pages) is a simplified design of Servlet. It is proposed by Sun Microsystems and also many companies involved for the

* Contact Author: Tao Gao, Email: gaotao863@163.com

establishment of a dynamic web technology standards [1]. Java program segment (Script let) and JSP tags is inserted into a traditional web HTM file to form a JSP file which expands the Html function by java syntax. JSP is easy to be used, fully object-oriented, platform-independent and secure, with all the characteristics of the need of Internet system design. Java code is not required to be complete or self-contained within a single script let block. It can straddle markup content, provided that the page as a whole is syntactically correct. The JSP syntax adds additional tags, called JSP actions, to invoke built-in functionality. Additionally, the technology allows for the creation of custom JSP tag libraries that act as extensions to the standard JSP syntax. A JavaServer Pages compiler is a program that parses JSPs, and transforms them into executable Java Servlets. A program of this type is usually embedded into the application server and run automatically the first time a JSP is accessed, but pages may also be precompiled for better performance, or compiled as a part of the build process to test for errors.

2 Combination of JSP and MySQL Data Base

2.1 JDBC Technology

JDBC (Java Database Connectivity) is a Java API [2] to execute a SQL statement, and program can be connected to a relational database by JDBC API. The query and update of the database can be achieved by using the Structured Query Language [3] [4].

2.2 The Structure of JDBC

In order to make cross-platform use of JDBC program, the appropriate drivers are need which can combine different database provide vendors. The frame of JDBC structure is shown in Figure.1 [5] [6].

By the JDBC driver conversion, the same program written in JDBC API can run on different database systems.

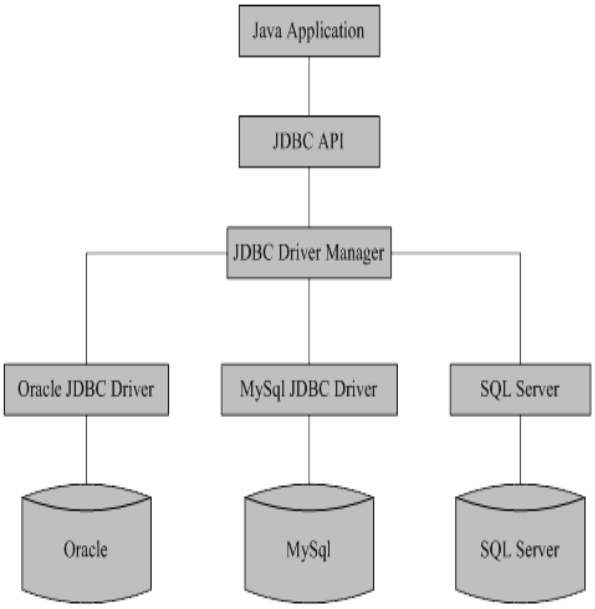


Fig1. The frame of JDBC

2.3 The Process of JDBC Programming

The public class 'DBManager' of Java is as follows.

```
public class DBManager {
    public final static String TABLE_NAME = "login_info";
    public final static String"""";

    public Statement getStatement() {
        Connection connection = null;
        Statement stmt = null;
        try {
            Class.forName("com.mysql.jdbc.Driver");
connection = (Connection) DriverManager. getConnection("jdbc:mysql://
localhost:3306/""""?useUnicode=true&characterEncoding=utf8", "root", """""");
stmt = connection.createStatement();
        } catch (ClassNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return stmt;
    }

public ArrayList<HashMap<String, Object>> getDatabaseContents() {
ArrayList<HashMap<String, Object>> list = new ArrayList<HashMap<String, Object>>();
HashMap<String, Object> map = null;
String sql = "select * from " + TABLE_NAME;
Statement stmt = getStatement();
ResultSet rst = null;
try {
        rst = stmt.executeQuery(sql);
        if(rst != null) {
        while(rst.next()) {
        map = new HashMap<String, Object>();
        map.put("""", rst.getString(""""));
        list.add(map);
              }
```

```
            }
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        return list;
    }

    public ResultSet query(String sql) {
        ResultSet rst = null;
        Statement stmt = getStatement();
System.out.println("stmt = " + stmt);
try {
    rst = stmt.executeQuery(sql);
        } catch (SQLException e) {
// TODO Auto-generated catch block
    e.printStackTrace();
        }
        return rst;
    }
    public int update(String sql) {
    Statement stmt = getStatement();
    int result = 0;
        try {
    result = stmt.executeUpdate(sql);
        } catch (SQLException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
        }
        return result;
    }
}
```

The process of JDBC programming contains six steps (with MySQL for example)

(1) To load the database driver

Before connecting to the database, first thing is to load the database drive to connect the Java Virtual Machine, which can be achieved by the static method of 'java.lang.Class' class: forName (String className) for implementation.

The method to load the MySQL drive is as:

```
Class.forName("com.mysql.jdbc.Driver").
```

(2) Connection of database

Connection of the database is achieved by the 'DriverManager' which defines the protocol of connection between the database, the sub-protocol and data source identification:

```
Connection conn = DriverManager.getConnection ("jdbc:mysql://
localhost: port number/database numbe", "user name", "key code")
```

(3) To create the statement object

To execute the SQL statements, the java.sql.Statement instance should be obtained, which can be divided into the following three types:

- Perform static SQL statements. The function 'createStatement()' is used to establish the 'Statement' instance.
- Execute dynamic SQL statements. The 'PreparedStatement' instance can be created by 'prepareStatement()' function.
- Perform a database stored procedure. The 'CallableStatement' instance can be created by 'prepareCall()' method.

(4) To execute the SQL statements

Statement interface provides three methods for executing SQL statements:

- execute(): perform any SQL statement.
- executeUpdate(): mainly used to perform DML and DDL statements.
- executeQuery(): execute the query.

(5) To operate the result set

Results of the SQL query will return a 'ResultSet' object, the program can use the object to obtain the query results as follows:

- first(), next(), last(): to move the record pointer.
- get(): the method to obtain the record pointer to the value of a particular line or column.

(6) To retake the database resources

After the operation, all JDBC objects should be close to release JDBC resources, the closing order is reversed to the declaration order as:

```
rs.close();
stmt.close();
conn.close();
```

Take the 'table' form for example, the implement is as follows:

```
import java.sql.*;
public class JDBC {
public static void main(String[] args) {
    Connection conn = null;
Statement stmt = null;
    ResultSet rs = null;
    try{
        Class.forName("com.mysql.jdbc.Driver");
        conn = DriverManager.getConnection("jdbc:
        mysql://localhost:3306/MySQL", "user", "password");
        stmt = conn.createStatement();
        rs = stmt.executeQuery("select * from table");
        while(rs.next()){
            System.out.print(rs.getString(1)+"\t");
            System.out.print(rs.getString(1)+"\t");
            System.out.print(rs.getString(1)+"\t");
...
        }
    } catch (ClassNotFoundException e){
        e.printStackTrace();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally{
        try{
            if(rs != null){
                rs.close();
                rs = null;
            }
            if(stmt != null){
                stmt.close();
                stmt = null;
            }
            if(conn != null){
                conn.close();
                conn = null;
            }
```

```
        } catch(SQLException e){
            e.printStackTrace();
        }
    }
  }
}
```

3 MySQL Database Online Update

3.1 Online Adding

Online adding operations can be achieved by 'insert into' statement, and it is used to insert data to the specified table. The syntax of 'insert into' statement is as follows: insert into table values (value1, value2 ...)

Taking the 'table' as an example, the specific method is as follows:

```
try {
...
        String sql = "insert into table values (?, ?, ?...)";
        String pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, value1);
        pstmt.setString(2, value2);
        pstmt.setString(3, value3);
...
pstmt.executeUpdate();
...
}
```

3.2 Online Deleting

Online delete can be achieved by 'delete from' statement. The syntax of the statement is as follows: 'delete from' table where 'field name = record name'.

For example, named 'table', field named 'id', the specific method is as follows:

Recording name is the data to be deleted.

3.3 Online Changing

According to 'add' and 'delete' SQL implementations, the operation of changing the database-related data is as follows:

```
try {
...
String sql1 = "delete from table where id = ?";
String pstmt1 = conn.prepareStatement(sql1);
String sql2 = "insert into table values (?, ?, ?...)";
String pstmt2 = conn.prepareStatement(sql2);
        pstmt1.setString(1, Recording name);
        pstmt2.setString(1, value1);
        pstmt2.setString(2, value2);
        pstmt2.setString(3, value3);
...
pstmt.executeUpdate();
...
}
```

The number of 'value' is related to the data account.

4 Conclusions

This paper introduces how to use JSP combined with SQL statements to achieve the processes about online adding, online changing and deleting operations for the database data to implement the MySQL database online update, which makes the database more convenient for common using.

5 Acknowledgments

This work is supported by National Natural Science Foundation of China (No. 71102174 and No. 51306058), the Fundamental Research Funds for the Central Universities (No. 2014QN46), the Hebei Science and Technology Support Project (No. 15212204D), the Tianjin Science and Technology Support Project (No. 15ZCZDNC00130), and the State Scholarship Fund for Visiting Scholar by the China Scholarship Council (No. 201406735004).

References

[1] Emmanuel Cecchet, Julie Marguerite and Willy Zwaenepoel. C-JDBC: Flexible Database Clustering Middleware. *USENIX Annual Technical Conference Freenix track*, June 2004.

[2] P. Chen, E. Lee, G. Gibson, R. Katz and D. Patterson. RAID: High-Performance, Reliable Secondary Storage. *ACM Computing Survey*, volume 26, n2, pp. 145-185, 1994.

[3] P. Furtado. Efficiently Processing Query-Intensive Databases over a Non-dedicated Local Network. *In: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05)*, IEEE Computer Society, 2005.

[4] Bradford W. Wade, Donald D. Chamberlin. IBM Relational Database Systems: The Early Years. *IEEE Annals of the History of Computing*, vol.34, no. 4, pp. 38-48, 2012.

[5] W.C. McGee. The Information Management System (IMS) Program Product. *IEEE Annals of the History of Computing*, vol. 31 no. 4, pp. 66–75, 2009.

[6] Ergsten, Hans. JavaServer Pages (3rd Edition ed.). O'Reilly Media. 2002.

Towards the Development of a Parameter-free Bat Algorithm

Iztok Fister Jr., Iztok Fister
University of Maribor
Faculty of Electrical Engineering and Computer Science
Smetanova 17, 2000 Maribor
iztok.fister1@um.si

Xin-She Yang
Middlesex University
School of Science and Technology
The Burroughs
London, United Kingdom

ABSTRACT

The bat algorithm is a very simple and efficient nature-inspired algorithm belonging to the swarm intelligence family. Recent studies showed its potential in solving the continuous and discrete optimization problems and many researchers have applied bat algorithm to solve real-world problems. However, empirical studies showed that the main issue of most algorithms is the setting of control parameters. An algorithm such as the bat algorithm typically has a few parameters, and it is very time-consuming to find its best parameter combination. Here, we propose a new parameter-free bat algorithm variant without the need for control parameters. Initial experiments on basic benchmark functions showed the potential of this new approach.

Keywords

bat algorithm, control parameters, optimization

1. INTRODUCTION

As the world is becoming a global village, advances in technologies means that new challenges are emerging. Better and greener products are needed for companies to have a competitive edge. Thus, the optimization of product designs and manufacturing processes become ever-more important. It is expected that artificial intelligence is among the most promising developments for the next 20 years. Many artificial intelligence tasks can be achieved by optimization and machine learning techniques.

One of the most important tasks for many productions is how to improve the production of products and services for the market, which requires the optimal design and the optimal use of resources. Until recently, many methods have been used to help designers and developers to solve such optimization problems. Some methods are pure mathematical, while others are combined with computer sciences. In line with this, optimization algorithms play a big role. Recently, stochastic nature-inspired methods are among the promising kind of optimization algorithms. This family consists of the following algorithms: genetic algorithms [7], genetic programming [11], evolution strategies [1], evolutionary programming [5], differential evolution [14], particle swarm optimization [9], firefly algorithm [18], bat algorithm [19] and dozens of others [4].

The bat algorithm (BA)is one of the latest algorithms in the swarm intelligence (SI) domain. Due to its simplicity, it is a very efficient as well. The original BA works with five parameters that represent a potential problem for users which usually may not know how to specify their values properly. Therefore, this paper introduces a new parameter-free or parameterless BA (PLBA) that eliminates this drawback and then proposes techniques for a rational and automated parameter setting on behalf of the user. This study is based on the paper of Lobo and Goldberg [12].

The structure of the remainder of the paper is as follows. Section 2 presents a description of the original BA. In Section 3, a design of the parameter-free or parameterless BA (also PLBA) is presented. Experiments and results are discussed in Section 4. The paper concludes in Section 5 where the directions for the future work are also outlined.

2. BAT ALGORITHM

The origin of the BA development can date back to the year of 2010 when Xin-She Yang in [19] proposed the new nature-inspired algorithm that mimics the phenomenon of echolocation by some species of bats for the optimization process. This algorithm can integrate the characteristics of many algorithms such as particle swarm optimization (PSO) algorithm [9] and simulated annealing (SA) [10]. Primarily, BA was developed for continuous optimization, but many recent papers showed the potential of the algorithm in solving discrete optimization problems. In a nutshell, this algorithm is widespread in many areas of optimization and industrial applications [8,13,15–17]. Yang proposed the following rules which mimics the bat behavior:

- All bats use echolocation to sense the distance to target objects.
- Bats fly with the velocity v_i at position x_i, the frequency $Q_i \in [Q_{min}, Q_{max}]$ (also the wavelength λ_i), the rate of pulse emission $r_i \in [0, 1]$, and the loudness

$A_i \in [A_0, A_{min}]$. The frequency (and wavelength) can be adjusted depending on the proximities of their targets.

- The loudness varies from a large (positive) A_0 to a minimum constant value A_{min}.

These three rules mimics the natural behavior of bats, while the full basic pseudo-code is presented in Algorithm 1.

Algorithm 1 Bat algorithm

Input: Bat population $\mathbf{x_i} = (x_{i1}, \ldots, x_{iD})^T$ for $i = 1 \ldots Np$, MAX_FE.
Output: The best solution $\mathbf{x}_{best}$ and its corresponding value $f_{min} = \min(f(\mathbf{x}))$.

```
1: init_bat();
2: eval = evaluate_the_new_population;
3: f_min = find_the_best_solution(x_best); {initialization}
4: while termination_condition_not_met do
5:   for i = 1 to Np do
6:     y = generate_new_solution(x_i);
7:     if rand(0,1) < r_i then
8:       y = improve_the_best_solution(x_best)
9:     end if { local search step }
10:    f_new = evaluate_the_new_solution(y);
11:    eval = eval + 1;
12:    if f_new ≤ f_i and N(0,1) < A_i then
13:      x_i = y; f_i = f_new;
14:    end if { save the best solution conditionally }
15:    f_min=find_the_best_solution(x_best);
16:  end for
17: end while
```

The BA is a population-based algorithm, where the population size is controlled by a parameter Np. The main loop of the algorithm (lines 4-16 in Algorithm 1) starts after the initialization (line 1), the evaluation of the generated solutions (line 2) and determination of the best solutions (line 4). It consists of the following elements:

- generating a new solution (line 6),
- improving the best solution (lines 7-9),
- evaluating the new solution (line 10),
- saving the best solution conditionally (lines 12-14),
- determining the best solution (line 15).

The generation of a new solution obeys the following equation:

$$\begin{aligned} Q_i^{(t)} &= Q_{min} + (Q_{max} - Q_{min})N(0,1), \\ \mathbf{v}_i^{(t+1)} &= \mathbf{v}_i^t + (\mathbf{x}_i^t - \mathbf{x}_{best})Q_i^{(t)}, \\ \mathbf{x}_i^{(t+1)} &= \mathbf{x}_i^{(t)} + \mathbf{v}_i^{(t+1)}, \end{aligned} \tag{1}$$

where $N(0,1)$ is a random number drawn from a Gaussian distribution with a zero mean and a standard deviation of one, and $Q_i^{(t)} \in [Q_{min}^{(t)}, Q_{max}^{(t)}]$ is the frequency determining the magnitude of the velocity change. The improvement of the current best solution is performed according to the following equation:

$$\mathbf{x}^{(\mathbf{t})} = \mathbf{x}_{best} + \epsilon A_i^{(t)} N(0,1), \tag{2}$$

where $N(0,1)$ denotes the random number drawn from a Gaussian distribution with a zero mean and a standard deviation of one. In addition, ϵ is the scaling factor and $A_i^{(t)}$ is the loudness. The improvement is controlled by a parameter r_i. It is worth pointing that the parameter balances the exploration and exploitation components of the BA search process, where exploitation is governed by Eq. 1 and exploitation by Eq. 2.

The evaluation function models the characteristics of the problem to be solved. The archiving of the best solution conditionally is similar to that used in SA, where the best solution is taken into the new generation according to a probability controlled by the parameter A_i in order to avoid getting stuck into a local optimum.

2.1 Control parameters in the bat algorithm and their bottlenecks

Control parameters guide algorithms during the search space and may have a huge influence on the quality of obtained solutions. In summary, the BA is controlled by the following control parameters:

- the population size NP,
- loudness A_i,
- pulse rate r_i,
- minimum frequency Q_{min} and
- maximum frequency Q_{max}.

As we can see, the BA has five control parameters that can be difficult for users to set a proper combination of control parameter settings that are the most suitable for the particular problem. Though some algorithms have fewer parameters such as differential evolution [14] that employs only three basic control parameters. However, the parameter tuning can be a time-consuming task for all algorithms. On the other hand, there are also some other robust self-adaptive [3] BA variants that can modify the values of control parameters during the run.

The main task for us is to develop a parameterless variant of the bat algorithm.

3. DESIGN OF A PARAMETERLESS BAT ALGORITHM

In order to develop a new parameterless BA (PLBA), the influence of control parameters was studied and extensive studies revealed that some algorithm parameters can be effectively set. For example, the parameter $Q_i \in [Q_{min}, Q_{max}]$ determines the magnitude of the change and settings of parameters Q_{min} and Q_{max} depend on the problem of interest. However, the rational setting of this parameter can be approximated with the lower x_i^{Lb} and upper x_i^{Ub} bounds of the

particular decision variables as follows:

$$Q_i^{(t)} = \frac{x_i^{(Ub)} - x_i^{(Lb)}}{Np} \cdot N(0,1), \quad (3)$$

where $N(0,1)$ has the same meaning as in Eq. (1). For example, when the $x_i^{(Lb)} = -100.0$ and $x_i^{(Ub)} = 100.0$, the frequency is obtained in the interval $Q_i^{(t)} \in [0,2]$.

The rationality for setting the values of parameters r_i and A_i are obtained, based on the following consideration. Parameter r_i controls the exploration/exploitation components of the BA search process. The higher the value, the more the process is focused on the exploitation. However, the higher r_i also means that the modified copies of the best solution are multiplied in the population. As a result, the premature convergence to the local optimum can occur. The appropriate value of this parameter is $r_i = 0.1$, as revealed during the extensive experimental work.

Similar consideration is valid also for parameter A_i; i.e., the lower the parameter, the less time the best current solution is preserved in the new generation. Therefore, the rational selection of this parameter is $A_i = 0.9$ that preserves the 90% of each best solutions and ignores the remaining 10%.

The population size is also a crucial parameter in the BA. A lower population size may suffer from the lack of diversity, while a higher population size may cause slow convergence. However, the rational setting of the population size depends on the problem of interest. Therefore, this parameter is varied in the interval $Np \in [10, 1280]$ in the proposed PLBA such that each population size multiplied by two in each run starting with $Np = 10$. In this way, eight instances of the PLBA is executed and a user selects the best results among the instances.

4. EXPERIMENTS

The purpose of our experimental work was to show that the results of the proposed PLBA are comparable if not better than the results of the original BA. In line with this, the original BA was compared with the PLBA using the rational selected parameter setting, i.e., Q_i was calculated according Eq. (3) (e.q., $Q_i \in [0.0, 2.0]$), pulse rate and loudness were fixed as $r_i = 0.1$ and $A_i = 0.9$, while the population size was varied in the interval $Np = \{10, 20, 40, 80, 160, 320, 640, 1280\}$. In contrast, the original BA was run using the following parameters: $Np = 100$, $r_i = 0.5$, $A_i = 0.5$ and $Q_i \in [0.0, 2.0]$ as proposed in Fister et al. [3].

Experiments were run on the benchmark suite consisting of five functions (Fig. 1). Here, the domains of parameters were limited into the interval $[-10, 10]$ by functions, while the dimensions of functions $D = 10$ were applied during this preliminary work. The optimization process of functions was stopped after 10,000 fitness/function evaluations. Each algorithm was run 25 times.

The results of the comparative study are summarized in Table 2, where the mean values and standard deviations are given for each algorithm. The BA in Table 2 denotes the original BA while PL-i the parameter-less BA using different population sizes Np.

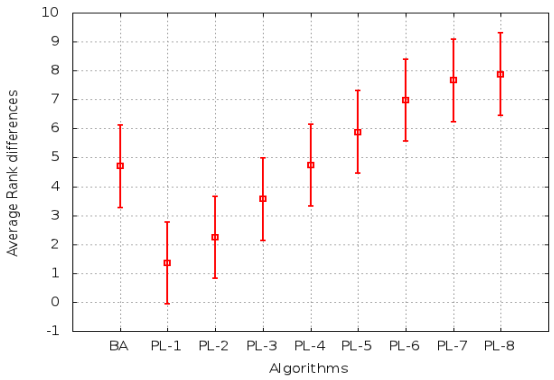


Figure 1: Summary of the comparison analysis.

The results are also statistically evaluated using the Friedman's non-parametric statistical tests [2, 6]. These results are shown in Table 3 where the ranks, confidence intervals with significance level $\alpha = 0.05$, and dagger signs denoting the significance difference are summarized.

Table 3: Friedman's statistical tests.

Alg.	Np	Rank	CI	Sign.
BA	100	4.7	[3.28,6.12]	†
PL-1	10	1.36	[-0.06,2.78]	†
PL-2	20	2.24	[0.82,3.66]	†
PL-3	40	3.56	[2.14,4.98]	†
PL-4	80	4.74	[3.32,6.16]	†
PL-5	160	5.88	[4.46,7.30]	†
PL-6	320	6.98	[5.56,8.40]	
PL-7	640	7.66	[6.24,9.08]	
PL-8	1280	7.88	[6.46,9.30]	‡

The same results are also presented graphically in Fig. 1, where ranges and confidence intervals are shown as points and lines, respectively. Two algorithms are significantly different, if their confidence intervals do not overlap. From Fig. 1, it can be seen that the population size has a significant influence on the results of the PLBA. The higher the population size, the better the results. Interestingly, using the PLBA with population sizes $Np = 640$ and $Np = 1,280$ significantly outperformed the results of the original BA.

5. CONCLUSION

Population-based nature-inspired algorithms can be a powerful tool for solving the hard optimization problems. However, these kinds of algorithms can depend crucially on the good parameter setting. Unfortunately, finding the proper parameter setting for a specific problem of interest is not easy and can pose challenges for users. In this paper, a parameterless BA (also PLBA) algorithm has been proposed in order to avoid these issues.

The PLBA sets parameters either by guessing their rational values, such as setting the parameters pulse rate r_i, loudness A_i, minimum Q_{min} and maximum Q_{max} frequencies, or by searching for their proper values experimentally, such as setting the population size Np. The results of the proposed PLBA are comparable with the results of the original BA by solving the benchmark suite of five well-known functions. A statistical analysis of the results has showed that

Table 1: Summary of the benchmark functions.

Tag	Function	Definition	Domain
f_1	Ackley's	$f(\mathbf{x}) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	$[-10, 10]$
f_2	Griewank	$f(\mathbf{x}) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})$	$[-10, 10]$
f_3	Rastrigin	$f(\mathbf{x}) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$	$[-10, 10]$
f_4	Sphere	$f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$	$[-10, 10]$
f_5	Whitley	$f(\mathbf{x}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\left[\frac{(100(x_i^2-x_j)^2+(1-x_j)^2)^2}{4000} - \cos(100(x_i^2 - x_j)^2 + (1-x_j)^2) + 1\right]$	$[-10, 10]$

Table 2: Experimental results.

Alg.	Np	f_1	f_2	f_3	f_4	f_5
BA	40	1.11e+01±8.66e-01	2.76e-01±2.70e-01	1.70e+02±3.92e+01	7.79e+01±2.53e+01	3.21e+04±1.95e+04
PL-1	10	1.15e+01±8.91e-01	3.96e-01±2.68e-01	2.18e+02±4.61e+01	9.65e+01±3.33e+01	4.87e+04±2.47e+04
PL-2	20	1.13e+01±1.15e+00	3.63e-01±2.94e-01	1.72e+02±5.09e+01	7.80e+01±2.82e+01	4.65e+04±2.65e+04
PL-3	40	1.06e+01±7.11e-01	2.52e-01±2.59e-01	1.46e+02±4.47e+01	6.56e+01±2.00e+01	3.03e+04±1.27e+04
PL-4	80	1.01e+01±7.59e-01	1.79e-01±1.64e-01	1.27e+02±3.58e+01	5.49e+01±1.88e+01	1.90e+04±1.26e+04
PL-5	160	9.68e+00±8.44e-01	1.07e-01±3.24e-02	1.24e+02±3.59e+01	4.58e+01±1.58e+01	1.63e+04±7.32e+03
PL-6	320	9.42e+00±7.32e-01	1.16e-01±3.82e-02	1.01e+02±2.25e+01	3.67e+01±1.17e+01	1.16e+04±3.95e+03
PL-7	640	8.76e+00±9.29e-01	1.02e-01±2.45e-02	9.28e+01±2.45e+01	3.16e+01±1.38e+01	9.08e+03±3.49e+03
PL-8	1280	8.57e+00±8.62e-01	1.02e-01±2.08e-02	9.22e+01±2.63e+01	3.23e+01±1.07e+01	1.08e+04±4.40e+03

the proposed PLBA can achieve comparable results as those obtained by the original BA. Such findings encourage us to continue with this work in the future. As the first goal, we would like to develop also a variant of a parameterless cuckoo search algorithm. We will also investigate the effect of the population size further in various applications.

6. REFERENCES

[1] Thomas Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, Oxford, UK:, 1996.

[2] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, December 2006.

[3] Iztok Fister Jr., Simon Fong, Janez Brest, and Iztok Fister. A novel hybrid self-adaptive bat algorithm. *The Scientific World Journal*, 2014, 2014.

[4] Iztok Fister Jr, Xin-She Yang, Iztok Fister, Janez Brest, and Dušan Fister. A brief review of nature-inspired algorithms for optimization. *Elektrotehniški vestnik*, 80(3):116–122, 2013.

[5] Lawrence Jerome Fogel, Alvin J. Owens, and Michael John Walsh. *Artificial Intelligence through Simulated Evolution*. John Willey & Sons, Inc., New York, USA:, 1966.

[6] Milton Friedman. A comparison of alternative tests of significance for the problem of m rankings. *Ann. Math. Statist.*, 11(1):86–92, 03 1940.

[7] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, USA:, 1989.

[8] Andrés Iglesias, Akemi Gálvez, and Marta Collantes. Global-support rational curve method for data approximation with bat algorithm. In *Artificial Intelligence Applications and Innovations: 11th IFIP WG 12.5 International Conference, AIAI 2015, Bayonne, France, September 14-17, 2015, Proceedings*, volume 458, page 191. Springer, 2015.

[9] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942–1948. IEEE, 1995.

[10] S Kirkpatrick, CD Gelatt Jr, and MP Vecchi. Optimization by simulated annealing. *Science*, 220(4598), 1983.

[11] John Koza. *Genetic Programming 2 - Automatic Discovery of Reusable Programs*. MIT Press, Cambridge, USA:, 1994.

[12] Fernando G. Lobo and David E. Goldberg. An overview of the parameter-less genetic algorithm. In *Proceedings of the 7th Joint Conference on Information Sciences. (Invited paper)*, pages 20–23, 2003.

[13] Navaneethakrishna Makaram and Ramakrishnan Swaminathan. A binary bat approach for identification of fatigue condition from semg signals. In *Swarm, Evolutionary, and Memetic Computing*, pages 480–489. Springer, 2014.

[14] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

[15] Anass Taha, Mohamed Hachimi, and Ali Moudden. Adapted bat algorithm for capacitated vehicle routing problem. *International Review on Computers and Software (IRECOS)*, 10(6):610–619, 2015.

[16] TP Talafuse and EA Pohl. A bat algorithm for the redundancy allocation problem. *Engineering Optimization*, (ahead-of-print):1–11, 2015.

[17] Jinfeng Wang, Xiaoliang Fan, Ailin Zhao, and Mingqiang Yang. A hybrid bat algorithm for process planning problem. *IFAC-PapersOnLine*, 48(3):1708–1713, 2015.

[18] Xin-She Yang. Firefly algorithm. In *Nature-Inspired Metaheuristic Algorithms*, pages 79–90. Luniver Press, London, UK:, 2008.

[19] Xin-She Yang. A new metaheuristic bat-inspired algorithm. In *Nature inspired cooperative strategies for optimization (NICSO 2010)*, pages 65–74. Springer, 2010.

Using trees to speed up the Floyd-Warshall algorithm

Marko Grgurovič
UP DIST
University of Primorska
marko.grgurovic@famnit.upr.si

ABSTRACT

A classic problem in algorithmic graph theory is to find shortest paths, and a common variant is the all-pairs shortest path problem (APSP). We consider non-negatively weighted APSP. Due to its simplicity and efficiency, the Floyd-Warshall algorithm is frequently used to solve APSP in practice.

We propose a combination of the Floyd-Warshall algorithm with a hourglass-like tree structure, which reduces the number of path combinations that have to be examined. Only those path combinations that provably cannot change the values in the shortest path matrix are omitted. The resulting algorithm is simple to implement and uses no fancy data structures.

In empirical tests, the new algorithm is faster than the Floyd-Warshall algorithm for random complete graphs on $256 - 4096$ nodes by factors of $3.5 - 4.8$x. When we inspect the number of path combinations examined compared to the Floyd-Warshall algorithm, they are reduced by a factor of $12 - 90$x. All code was written in *C*.

Categories and Subject Descriptors

F.2.2 [**Theory of Computation**]: Nonnumerical Algorithms and Problems

General Terms

Algorithms, Theory

Keywords

Dynamic programming, shortest path

1. INTRODUCTION

Let $N = (V, A, \ell)$ be a directed network where we refer to $V = \{v_1, v_2, ..., v_n\}$ as the set of nodes and A as the set of arcs (directed edges). To simplify notation we will write $|V| = n$ and $|A| = m$. The function $\ell : A \to \mathbb{R}$ maps arcs to (possibly negative) lengths. Finding shortest paths in such networks is a classic problem in algorithmic graph theory. Two of the most common variants of the problem are the single-source shortest path (SSSP) problem and the all-pairs shortest path problem (APSP). In the SSSP variant, we are tasked with finding the path with the least total length from a fixed node $s \in V$ to every other node in the network. Similarly, the APSP problem asks for the shortest path between every pair of nodes $u, v \in V$. In this paper we will focus exclusively on the APSP variant of the problem.

The asymptotically fastest APSP algorithm for dense graphs runs in $O(n^3 \log\log^3 n / \log^2 n)$ time [2]. For non-negative arc length functions and for sparse graphs, there exist asymptotically fast algorithms for worst case inputs [10, 12, 11], and algorithms which are efficient average-case modifications of Dijkstra's algorithm [6, 3, 9]. In fact, it turns out that any SSSP algorithm can be asymptotically sped up in the average-case setting when solving APSP [1].

In practice, the Floyd-Warshall algorithm is frequently used, along with variations of Dijkstra's algorithm when the graph is sparse. One can also approach APSP through funny matrix multiplication, and practical improvements have been devised to this end through the use of sorting [8]. There exist many optimizations for the Floyd-Warshall algorithm, ranging from better cache performance [13], optimized program-generated code [4], to parallel variants for the GPU [5, 7].

In spite of intensive research on efficient implementations of the Floyd-Warshall algorithm, the authors are not aware of any improvement to the number of path combinations examined by the algorithm. In this paper, we propose a modification of the Floyd-Warshall algorithm that combines it with a hourglass-like tree structure, which reduces the number of paths that have to be examined. Only those path combinations that provably cannot change the values in the shortest path matrix are omitted. The resulting algorithm is simple to implement, uses no fancy data structures and in our tests is faster than the Floyd-Warshall algorithm for random complete graphs on $256 - 4096$ nodes by factors ranging from $2.5-8.5$x. When we inspect the number of path combinations examined however, our modification reduces the number by a staggering factor of $12 - 90$x.

2. ALGORITHM

The simplest improvement over Floyd-Warshall involves the use of a tree, denoted as OUT_k, which is the shortest path tree containing paths that begin in node v_k and end in some

node $w \in V \setminus \{v_k\}$, but only go through nodes in the set $\{v_1, ..., v_{k-1}\}$. In order to reconstruct the shortest paths, the Floyd-Warshall algorithm needs to maintain an additional matrix, which specifies the path structure. In our algorithm this information is essential, since the path structure is used during the execution. We augment the Floyd-Warshall algorithm with a matrix $L[i][j]$ which specifies the penultimate node on the shortest path from i to j. This suffices for reconstructing the shortest path tree for all paths going out of k as follows: create n trees $\{T_1, ..., T_n\}$, now go through $j = 1$ to n and place T_j as the child of $T_{L[k][j]}$. This takes $O(n)$ time.

We then have the following algorithm:

Algorithm 1 Single-tree Algorithm.

```
1: procedure SINGLE-TREE(W)
2:     Initialize L, a n × n matrix, as L[i][j] := i.
3:     for k := 1 to n do
4:         Construct OUT_k.
5:         for i := 1 to n do
6:             Stack := empty
7:             Stack.push(v_k)
8:             while Stack ≠ empty do
9:                 v_x := Stack.pop()
10:                for all children v_j of v_x in OUT_k do
11:                    if W[i][k] + W[k][j] < W[i][j] then
12:                        W[i][j] := W[i][k] + W[k][j]
13:                        L[i][j] := L[k][j]
14:                        Stack.push(v_j)
15:                    end if
16:                end for
17:            end while
18:        end for
19:    end for
20: end procedure
```

For the sake of brevity, we omit proofs. We can augment Algorithm 1 with another tree. The second tree is similar to OUT_k, except that it is the incoming shortest path for v_k. Strictly speaking, this is not a tree[1], but we can reverse the directions of the arcs, which turns it into a tree with v_k as the root. The augmented algorithm is referred to as Hourglass (as opposed to Single-tree).

3. EMPIRICAL RESULTS

Our implementations were written in *C* and compiled with *gcc* using flags `-std=c99 -O3`. We ran the experiments on an Intel i7-2600 CPU running on Windows 7. All tests were ran 20 times and averaged.

The input graphs were pseudorandomly generated. For complete graphs, this meant assigning each arc an independently uniformly distributed random length in the range $(0, 1)$. Sparse graphs were generated by starting with an empty graph on 1024 nodes and adding a desired number of arcs, which were chosen independently according to the uniform random distribution, and assigned an independently uniformly distributed random length in the range $(0, 1)$. Arc lengths were represented using floating-point numbers in both cases.

We performed two sets of experiments, one on random complete graphs, and one on random sparse graphs. In both cases, we measured two quantities: number of path combinations examined, and actual running time. Since the results are numbers that range from very small to very large in both cases, we display the results as a percentage of the Floyd-Warshall algorithm, which is always 100% in the plots, but is not drawn explicitly.

The first set of experiments was for the input of random complete graphs of varying sizes. The results are shown in Figure 1. The second set of experiments was for the input of a random graph of 1024 nodes whose number of arcs varied from $10 - 80\%$ where $100\% = n^2$. To make the comparison between Floyd-Warshall and the modified versions fairer in the second set of experiments, we augmented the Floyd-Warshall algorithm with a simple modification, that allowed it to skip combinations i, k where $W[i][k] = \infty$, which reduced the running time and number of path combinations examined. The results of the second set of experiments are shown in Figure 2.

4. CONCLUSIONS

In the proposed algorithm, we can observe a significant reduction in terms of path combinations examined (see left plot of Figure 1) . This quantity dominates the algorithm's asymptotic running time and, as observed, decreases compared to the cubic algorithm when inputs grow larger. It might be possible to obtain sub-cubic asymptotic bounds in the average-case model. Despite this reduction in path combinations examined, the actual time savings shown in the right plot of Figure 1 are more modest. A variety of factors contribute to this disparity, ranging from cache performance to implementation details that have not been thoroughly optimized. Regardless, the speedups remain significant for practical applications, and future work could improve this further. The experiments on sparse graphs in Figure 2 show a reduction in path combinations examined as the graph becomes sparser, but the effect on the running time seems to be minor.

Overall, the Single-tree algorithm is the simplest to implement and offers good performance. The Hourglass algorithm has the potential to be even faster, but would likely require a better implementation. It is also worthwhile to note that the additional space requirements for the Single-tree algorithm are very modest, as most applications would typically require storing the path reconstruction matrix regardless.

5. REFERENCES

[1] A. Brodnik and M. Grgurovič. Speeding up shortest path algorithms. In K.-M. Chao, T. sheng Hsu, and D.-T. Lee, editors, *ISAAC*, volume 7676 of *Lecture Notes in Computer Science*, pages 156–165. Springer, 2012.

[2] T. M. Chan. More algorithms for all-pairs shortest paths in weighted graphs. *SIAM J. Comput.*, 39(5):2075–2089, 2010.

[3] C. Demetrescu and G. F. Italiano. Experimental analysis of dynamic all pairs shortest path algorithms.

[1]The hourglass name comes from placing this structure atop the OUT_k tree, which gives it a hourglass-like shape, with v_k being the neck.

Figure 1: The left plot shows the percentage of path combinations examined by the two modifications of Floyd-Warshall, when compared to the original algorithm (which is always at 100%, not shown), for the input of complete graphs of various sizes. The right plot shows the difference in execution time, compared to the original algorithm.

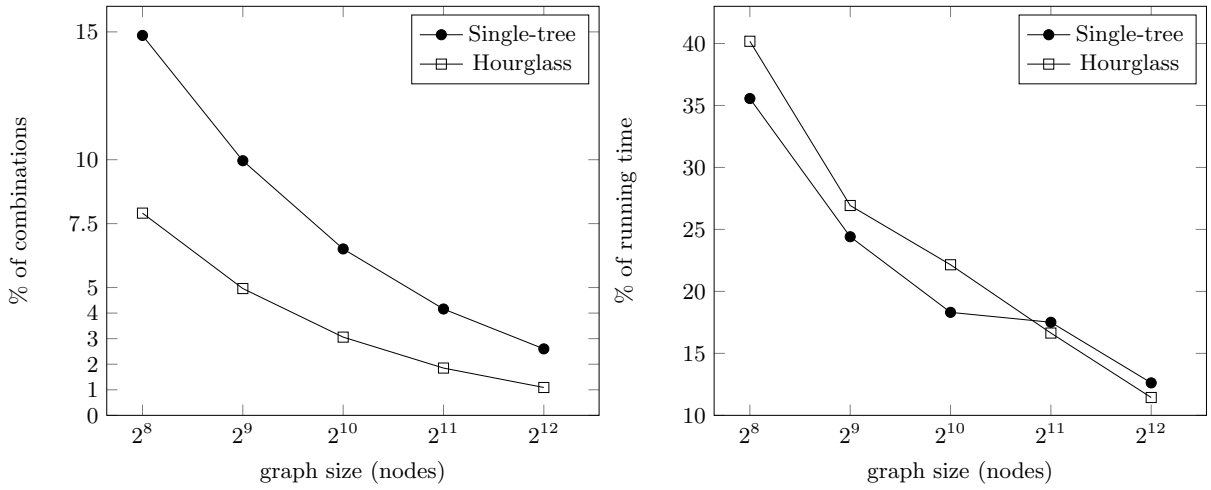
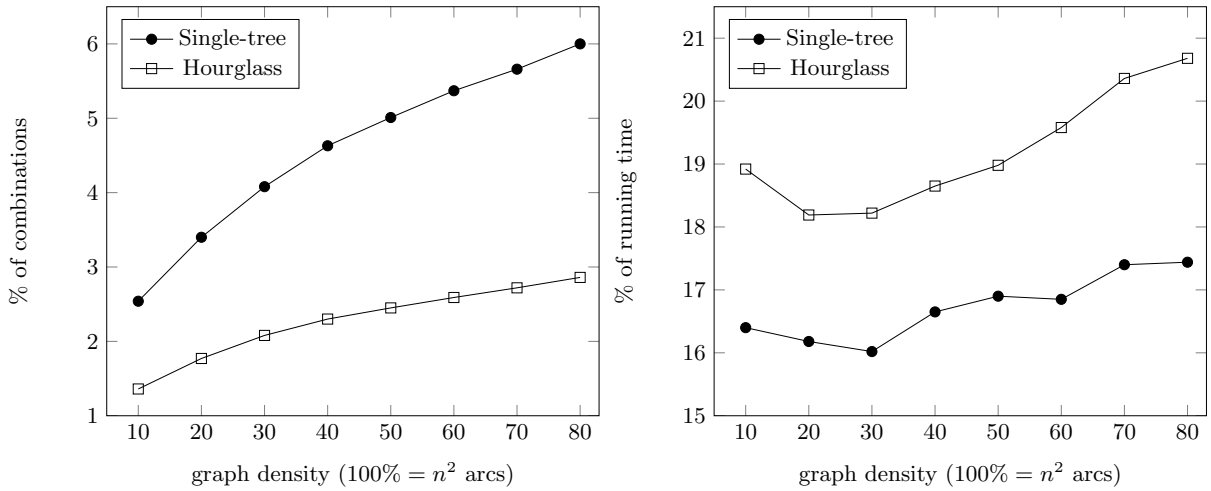


Figure 2: The left plot shows the percentage of path combinations examined by the two modifications of Floyd-Warshall, when compared to the original algorithm (which is always at 100%, not shown), for the input of a graph with 1024 nodes and various arc densities. The right plot shows the difference in execution time, compared to the original algorithm.



ACM Transactions on Algorithms, 2(4):578–601, 2006.

[4] S.-C. Han, F. Franchetti, and M. Püschel. Program generation for the all-pairs shortest path problem. In *Proceedings of the 15th international conference on Parallel architectures and compilation techniques*, PACT '06, pages 222–232, New York, NY, USA, 2006. ACM.

[5] P. Harish and P. J. Narayanan. Accelerating large graph algorithms on the GPU using CUDA. In *Proceedings of the 14th international conference on High performance computing*, HiPC'07, pages 197–208, Berlin, Heidelberg, 2007. Springer-Verlag.

[6] D. Karger, D. Koller, and S. J. Phillips. Finding the hidden path: time bounds for all-pairs shortest paths. *SIAM Journal on Computing*, 22(6):1199–1217, 1993.

[7] G. J. Katz and J. T. Kider, Jr. All-pairs shortest-paths for large graphs on the GPU. In *Proceedings of the 23rd ACM SIGGRAPH/EUROGRAPHICS symposium on Graphics hardware*, GH '08, pages 47–55, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association.

[8] J. J. McAuley and T. S. Caetano. An expected-case sub-cubic solution to the all-pairs shortest path problem in R. *CoRR*, abs/0912.0975, 2009.

[9] Y. Peres, D. Sotnikov, B. Sudakov, and U. Zwick. All-pairs shortest paths in $O(n^2)$ time with high probability. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 663–672, Washington, DC, USA, 2010. IEEE Computer Society.

[10] S. Pettie. A new approach to all-pairs shortest paths on real-weighted graphs. *Theor. Comput. Sci.*, 312(1):47–74, January 2004.

[11] S. Pettie and V. Ramachandran. A shortest path algorithm for real-weighted undirected graphs. *SIAM J. Comput.*, 34(6):1398–1431, June 2005.

[12] M. Thorup. Undirected single-source shortest paths with positive integer weights in linear time. *J. ACM*, 46(3):362–394, May 1999.

[13] G. Venkataraman, S. Sahni, and S. Mukhopadhyaya. A blocked all-pairs shortest-paths algorithm. *J. Exp. Algorithmics*, 8, Dec. 2003.

3D walk through the references of Dutch women writers

Jernej Grosar
Faculty of Computer and Information Science, University of Ljubljana
Večna pot 113
Ljubljana, Slovenia
gosar.jernej@gmail.com

Assist. Jure Demšar
Faculty of Computer and Information Science, University of Ljubljana
Večna pot 113
Ljubljana, Slovenia
jure.demsar@fri.uni-lj.si

Assist. Prof. Dr. Narvika Bovcon
Faculty of Computer and Information Science, University of Ljubljana
Večna pot 113
Ljubljana, Slovenia
narvika.bovcon@fri.uni-lj.si

ABSTRACT
The goal in this project was to create visualization of connections in interactive 3D world between Dutch female writers in years between 1790 and 1914. In this world, writers are represented by houses and windmills, and player can discover these connections by travelling between them.

Categories and Subject Descriptors
H.5.1 [**Multimedia Information Systems**]: Artificial, augmented and virtual realities

General Terms
Visualization

Keywords
Women writers, Dutch, 3D walk, visualization, virtual reality, Digital Humanities, spatialized reading, interdisciplinary collaboration, Unity3D

1. INTRODUCTION
The field of Digital Humanities has been a territory of experimental collaborations between different disciplines for the last thirty years, with especially intense developments in the last decade. However, the possibilities to pose research questions, the answers to which are possible only now with the use of new information technologies, and the creation of media hybrids as representation and visualization techniques are far from being exhausted. The paper will present a case of interdisciplinary collaboration, an attempt at a digital humanities project that connects and displaces the traditional conceptions of text, map and space.

2. PROJECT
2.1 The institutional framework and the concept of the project
This project was realized as a seminar work at the Introduction to Design class at the Faculty of Computer and Information Science, University of Ljubljana, under the supervision of Assist. Prof. Dr. Narvika Bovcon and Assist. Jure Demšar. It was a part of an international collaboration with HERA project "Travelling texts 1790-1914: The transnational reception of women's writing at the fringes of Europe".

The students were given the task to create an interactive 3D space that would contain the network of quotations of Dutch women writers. The concept behind this type of presentation of data from the field of literary history was quite experimental for both disciplines that were put in dialogue in this project, i. e. the literary studies and the computer games development, however, the middle ground that allowed for a successful collaboration was found in new media art approaches that research virtual spaces beyond the use for gaming. The concept of "spatialized reading" was proposed in the form of a script that guided the user's walk through the 3D space from one object to the next.

The project was inspired by various examples of 3D spaces used in conjunction with language or artworks and by adventure computer games with 3D text and graphic elements. In the project The Legible City [1] the interface is in the form of a bicycle, which enables riding through the computer generated space, where instead of buildings in the city there are letters which make up text. ArsDoom [2] is a patch of the first person shooter game Doom, where the scene is modelled as the space of the new media art festival Ars Electronica and the objects in the space are artworks, instead of shooting, one can paint with a brush. Srečo Dragan and Computer Vision Laboratory made The Virtual Jakopic Gallery (1998) [3] as a 3D web gallery written in VRML language. If you look back, it won't be there anymore [4] is a 3D computer generated space that is created based on what the user has already seen and what not; objects in space are letters.

2.2 Realization
The main goal for the students was to create the visualization of the connections between Dutch women writers in the

Figure 1: House and windmill representing two writers along with quote on the top.

years between 1790 and 1914 - how they referenced or quoted each other in letters that they wrote to their friends or to other writers, in newspaper articles or in books. Because of the limited time for realization of the project, only a few writers and connections between them were used.

The visual representation of the connections between Dutch women writers has been created as a 3D computer "game" with the help of the Unity3D game engine. This gaming engine has been chosen because it is easy to use and one can very quickly create landscape and logic behind it. However, in our project the gaming experience is reduced to walking in space, finding the objects and reading the quotations (Figure 1), and of course the contemplation of the connections thus represented. In this 3D space there are two types of interactive objects: the writers are represented by the houses, if they had mentioned other writers in their work, or by windmills (Figure 2), if they have only been mentioned by others.

The user discovers the connections between the women writers by traveling between the houses. As he/she approaches a house so that he/she is close enough to it, the quote by this writer (that is the one represented by the house), in which she mentioned another writer, is shown on the upper side of the screen. There are only two writers (houses) shown in this virtual world at the same time – the one that has referenced the other and the one that has been referenced. When a writer that has no connections (has not mentioned another writer) is visited, the player moves back to a previous one that has another connection (Figure 3) and only then the new connection to a new writer is shown. The connections and the successive appearance of the houses and windmills are ordered by the length of the continuous path that connects the writers by their references, that means that the chain of connections that connects the most writers that had mentioned others is shown first.

There are also instructions at the bottom of the screen for easier traveling and navigation. The instruction about which house or windmill should be visited next appears on the right. A mini map on the left side of the screen shows where the player is (the blue square on the mini map in Figure 3) and where houses are (the yellow and red circles). The quote, the instructions and the position of the indicators on the mini map are changed when a new connection is discovered, showing the position of the new house.

The reading of quotes is temporal and corresponds to the user's movement through 3D space. The quote is shown on the screen for as long as the user takes to walk to the next house. The map of the connections between the writers is not disclosed in the form of a diagram, instead it shows always just the two writers connected by the quote that is being read at this moment. The map is revealed temporally and dependent on the user's reading. In this way the contemplation of the text is put in the foreground. If the entire network of connections would be shown and all the quotes would be present in the space at the same time, the spatial relations between them would have to be determined in a completely different way, so that they would carry meaning.

The landscape of this game has been designed to look like a typical Dutch landscape with water channels, bridges and boats. Houses and windmills that represent Dutch women writers have also been chosen for this reason, so that a homogenous image of the landscape is achieved. House and windmill models had been taken from website http://tf3dm.com/ (uploaded by user "3dregenerator"), other objects, which appear in landscape, are part of the standard assets in Unity3D development program. To use this objects one can simply apply them by "painting" them on the landscape. The height map for landscape has been created in Adobe Photoshop as a gray scale image and later applied to this virtual world. For illumination of the landscape, directional light has been used, placed above the landscape.

2.3 Technical aspect of the project

As mentioned earlier, Unity3D gaming engine has been used for realization of this project. This engine uses scripts for manipulation with game objects which can be written in JavaScript or C# programming languages.

Figure 2: One of many windmills representing writers.

Figure 3: Player has already visited windmill.

In this project three different scripts have been used. First script, written in JavaScript, is the main script of this project and contains almost all of the game logic. It is used for manipulation of the quotes and instructions shown on the screen as well as for showing and hiding houses and windmills. It is attached to first person controller (the object which moves around and is controlled by player) and is triggered when this controller comes to the predetermined distance from the house or windmill - the area which object collider covers.

Other two scripts are written in C# and they control buttons in main menu and teleportation of the player which makes the interaction easier and enables player to quickly return from windmill to the house. When player enters teleportation area, he/she is teleported back to the vicinity of the house that he/she came from to this windmill, instead of walking all the way back.

3. CONCLUSIONS

The project of this kind involves the expertise of collaborators from different disciplines, where one of the problems that surface is also how to find a common language to communicate what is important for each of them. This entails communicating the way, how each of the collaborators has to prepare the necessary materials with which the other discipline has to work, and also explaining, what are the goals and standards of each of the collaborators that have to be met in the project. Our collaboration was successful, the 3D visualization of connections between Dutch women writers that has been presented in this paper will be exhibited at the conference related to the Women Writers database in Letterkundig Museum in Hague in September 2015.

4. ACKNOWLEDGMENTS

The project was realized in collaboration with the HERA project "Travelling Texts 1790-1914: The Transnational Reception Of Women's Writing At the Fringes of Europe".

The presentation concept for Dutch literary field focusing on women writers and the selection of quotations was prepared by Prof. Dr. Suzan Van Dijk, Huygens Institute for the History of Netherlands. The concept and script for "spatialized reading" was prepared by Narvika Bovcon and Assist. Prof. Dr. Aleš Vaupotič from the Research Center for Humanities at the University of Nova Gorica. The project was realized in 2015 at the Faculty of Computer and Information Science, University of Ljubljana.

5. REFERENCES

[1] Web source. Jeffrey Shaw: The Legible City (1989). http://www.jeffrey-shaw.net/html_main/show_work.php?record_id=83#.

[2] Web source. Orhan Kipcak, Reini Urban: ArsDoom (1995). http://www.gamescenes.org/2009/11/interview-orphan-kipcak-arsdoom-arsdoom-ii-1995.html.

[3] Web source. Franc Solina: Virtual technology and remote observation over the Internet for art applications (2000). http://eprints.fri.uni-lj.si/52/.

[4] Web source. Narvika Bovcon, Barak Reiser, Aleš Vaupotič, coding Igor Lautar: If you look back, it won't be there anymore (2005). http://black.fri.uni-lj.si/datadune/x_dd_binder_crop_optim_redu.pdf.

Budget travelling through virtual reality

Žiga Kerec
Faculty of computer and information science, Univeristy of Ljubljana
Večna pot 113
1000 Ljubljana, Slovenia
ziga.kerec@gmail.com

Marko Balažic
Faculty of computer and information science, Univeristy of Ljubljana
Večna pot 113
1000 Ljubljana, Slovenia
marko.balazic@gmail.com

Jure Demšar
Faculty of computer and information science, Univeristy of Ljubljana
Večna pot 113
1000 Ljubljana, Slovenia
jure.demsar@fri.uni-lj.si

Narvika Bovcon
Faculty of computer and information science, Univeristy of Ljubljana
Večna pot 113
1000 Ljubljana, Slovenia
narvika.bovcon@fri.uni-lj.si

ABSTRACT
This paper consists of information about an application developed for Ljubljana's Flow festival. We decided to develop a virtual reality app, due to the fast growing market and interesting end results. The Application was built in Unity3D - free professional game engine software, and it represents a journey through an abstract landscape. The Application is developed for Android mobile operating system and can be viewed through Google's budget friendly solution for virtual reality - Google Cardboard.

Keywords
Virtual reality, Android, Unity3D

1. INTRODUCTION
Virtual reality (VR) is one of the fastest growing markets in computer science industry today. All major companies are either creating or acquiring products and ideas that are bound to be the next big thing. One of the greatest drawbacks of VR technology is pricing of the products capable of presenting virtual 3D world. This problem was overcome recently when Google released a contraption called Google Cardboard. Google Cardboard is a low-budget VR headset, its biggest asset is its low price, which makes it very accessible for us students and various of our projects.

The idea for a VR app was born during the Introduction to Design class at the Faculty of Computer and Information Science, University of Ljubljana. The product was created as part of class's required seminar and was later on presented at Ljubljana Flow festival in June 2015. This festival is connected to the alternative art scene and is open to contemporary technological experiments such as building mapping projections and travelling through 3D virtual world. The initial idea consisted of a simple path leading people through different visual representations of an abstract virtual world that would trigger corresponding emotions. The whole experience is backed up with suitable background music and audio effects. The final product can be broken down to 4 different sections, the first one being a welcome scene with a trigger to start the second scene or the main part of the application, as shown in Figure 1. Objects that appear in all scenes are designed in low-poly technique, which gives them a sharp and abstract look.

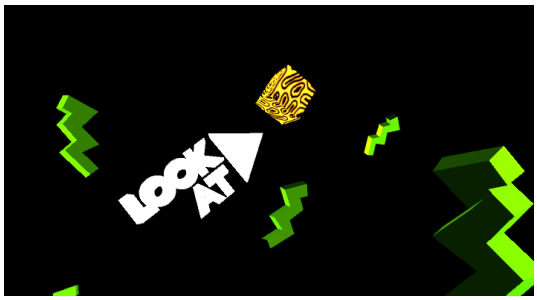


Figure 1: Start scene with trigger to start the main part of application.

In this paper we present how we conceptualized and built the virtual reality app for the Flow festival. In Section 2 we present some related work and the tools required to build and run the application. In third section we present the narrative and imagery that shape the user experience on his/her travel in our virtual world. We conclude this paper with findings and possible future work.

2. RELATED WORK AND GAME ENGINE

Unity3D is a very popular game development engine developed and maintained by a company called Unity Technologies. The engine is especially popular among indie game developers, mainly because it is free, and the games or applications made with Unity3D are easily ported on a number of different platforms – from Microsoft Windows, Android, iOS to Playstation [1]. Its main intention is game design, however Unity already proved as a good tool for various types of applications, e.g. urban studies [2] and virtual reality [3]. In their application Wang et al. [3] used Unity to simulate a quite complex study environment – a whole building complex in which students can study. As Unity proved as a good tool for simulating complex virtual reality environments we choose it as the tool for visualizing our much simpler environment.

3. THE VIRTUAL WORLD

Our applications offers a unique travelling experience into a different dimension. As our virtual word is in 3D we used Blender to model the objects surrounding the inter-dimension traveller (user of the application). Blender is a free professional and open-source 3D computer graphics software. All models used in the application were built from the ground up. One of the most important pieces of software is Android software development kit (SDK), which allowed us to transfer game/animation from Unity3D software to Android phones for testing and later on Google Play Store for general use.

The main part of the VR application consists of three visually and emotionally very different scenes. Here we describe, how we have designed the three parts (we discuss the selection of narrative elements, of colours, lights, of the speed of camera travelling) to achieve the corresponding user experience for each of them.

Figure 2: First section in the main scene. Dark mythical forrest, with thick fog and trees hovering over the path.

First stage in the main scene was meant to be a mythical forest, a dark and lonely place, where we should feel a bit cramped. As we are travelling down the path we are surrounded with trees hovering over us (Figure 2). The whole scene is filled with thick white fog and an occasional light bulb levitating in the sky. To back up the unpleasant feeling, we added a flock of crows flying over the scene, making loud noises.

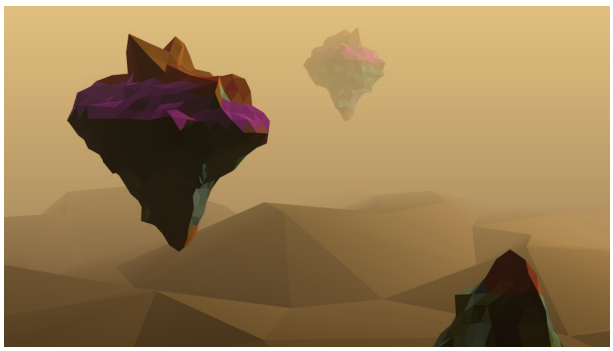


Figure 3: Brighter yellow section, comes right after mythical forrest.

After this dark emotional part, we should begin to see the light at the end of the forest, and as the journey evolves we should eventually reach the next, brighter scene (Figure 3). In this scene we should feel a bit happier and relaxed, therefore it is backed up with different style of music. Again we are travelling through thick fog, this time in bright yellow colour, and are surrounded by colourful islands. We added flowers (Figure 4) and other motives that gave a warmer feeling to the entire section. Last and final sector takes place mainly in a tube. This is a tube with animated textures of black and white stripes that cause us to loose our sense of speed and location. At the end of a long and twisted tube (Figure 5), we fall down the white cliff and stop right at the bottom, on top of red spikes.

Figure 4: Flower hill, last part of the second sector.

The whole space was designed from the ground up and it is put together with dozen objects, crafted in Blender software. We of course incorporated more objects, that are duplicated and just slightly altered for different visual effect. The fog was added for the purpose of masking the sense of space and for not noticing empty background behind the last line of objects.

4. CONCLUSION

Virtual reality has come a long way in the last few years, but there is still many ground to cover. Powerful pieces of hardware are still not as accessible to general public and small

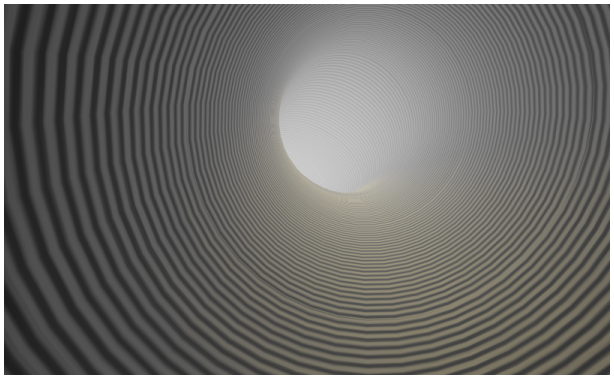


Figure 5: Tube with animated textures.

development studios, as budget friendly solutions such as Google Cardboard, that are great for developers to start exploring the world of Virtual Reality, even tho mobile smartphones are (still) not powerful enough for more sophisticated and realistic effects. This is exactly the target platform for our application. Budget friendly platform and an application with intermediate graphics complexity for optimal results on almost any modern Android phone. Virtual reality certainly should and will be the thing that will shape the future of computer graphics and gaming.

5. REFERENCES

[1] R. H. Creighton. *Unity 3D Game Development by Example: A Seat-of-Your-Pants Manual for Building Fun, Groovy Little Games Quickly*. Packt Publishing Ltd, 2010.

[2] A. Indraprastha and M. Shinozaki. The investigation on using unity3d game engine in urban design study. *Journal of ICT Research and Applications*, 3(1):1–18, 2009.

[3] S. Wang, Z. Mao, C. Zeng, H. Gong, S. Li, and B. Chen. A new method of virtual reality based on unity3d. In *Geoinformatics, 2010 18th International Conference on*, pages 1–5. IEEE, 2010.

Comparison of DE strategies for Gray-Level MultiLevel Thresholding

Uroš Mlakar
University of Maribor
Smetanova 17
Maribor, Slovenia
uros.mlakar@um.si

Iztok Fister
University of Maribor
Smetanova 17
Maribor, Slovenia
iztok.fister@um.si

ABSTRACT

In this paper we investigate the impact of different mutation strategies in differential evolution applied to the problem of gray-level multilevel thresholding. Four different strategies are compared, namely rand/1, best/1, rand to best/1, and current to best/1. These strategies are tested on four standard test images taken from literature. The quality of segmented images was compared on the basis of the PSNR metric, which showed that the best performing strategy was current to best/1 based on the mean PSNR value, but in the case of best result found, the rand/1 strategy performed best. When comparing the mean and best objective values, the best/1 strategy outperformed the others.

Keywords

Multilevel thresholding, Otsu criterion, evolutionary algorithm, differential evolution

1. INTRODUCTION

Image segmentation is a process of dividing an image into disjoint sets, which share similar properties such as intensity or color. Image segmentation is usually the first step for many high-level methods, such as feature extraction, image recognition, and classification of objects. Simply put, image segmentation is a process of dividing an image into regions, which are used as input for further specific applications. Image thresholding is one of the most used and simplest segmentation techniques, which performs image segmentation based on values contained in the image histogram. In the case of separating an image into two classes, the process is called bilevel thresholding, but when separating the image into several regions we deal with multilevel thresholding. The selection of optimal threshold values is crucial, since the results of good segmentation are a good foundation for applications, which further process the segmented images.

Multilevel thresholding can be regarded as an optimization process, usually maximizing certain criteria like between-class variance or various entropy measures. Many heuristic methods have gained a lot of attention recently, since exhaustive methods are usually computationally inefficient.

In this paper we investigate the influence of different mutation strategies on the quality of segmentation by using the between-class variance as the objective function proposed by Otsu [5]. The rest of the paper is organized as follows. In Section 2 some of related work is presented in the area of multilevel thresholding. In Section 3, image segmentation and the Otsu crietrion are described, while in Section 4 the differential evolution algorithm (DE), along with 4 mutation strategies are presented, which have been used for this study. In Section 5 we define the metric, which was used to assess the quality of the segmentation and present the experimental results. In Section 6 we will conclude this paper with some findings.

2. RELATED WORK

Many works has been done for multilevel thresholding using evolutionary algorithms. Some of these algorithms can be found in [7] and [4]. Alihodzic et al. [1] introduced a hybrid bat algorithm with elements from the differential evolution and artificial bee colony algorithms. Their result show their algorithm outperforms all other in the study, while significantly improving the convergence speed. Bhandari et al. [2] investigated the suitability of the cuckoo search (CS) and the wind driven optimization algorithm for multilevel thresholding using Kapur's entropy as the objective function. The algorithms were tested on a standard set of satellite images by using various number of thresholds. They concluded that both algorithms can be efficiently used for the multilevel thresholding problem. Duraisamy et al. [3] proposed a novel particle swarm optimization (PSO) algorithm maximizing the Kapur's entropy and the between-class variance. Their algorithm has been tested on 10 images, and the results compared with a genetic algorithm. The PSO proved better in terms of solution quality, convergence, and robustness. Zhang et al. [8] presented an artificial bee colony algorithm (ABC), for the problem of multilevel thresholding. They compared their algorithm to PSO and GA, where their conclusions were that the ABC is more rapid and effective, using the Tsallis entropy.

3. IMAGE SEGMENTATION

For the purpose of multilevel threshold selection the Otsu criterion was selected as the objective function. It operates on the histogram of the image, maximizing the between-class

variance. For a bilevel thresholding problem is this formally defined as:

$$\sigma_B^2(t^*) = \max_{1 \leq t \leq L} \sigma_B^2(t), \tag{1}$$

where

$$\sigma_B^2 = \omega_0 \omega_1 (\mu_1 - \mu_0)^2, \tag{2}$$

with ω_0 and ω_1 being probabilities of class occurrence and μ_0 and μ_1 the mean values of each class.

This problem is easily expandable to a multilevel problem as:

$$\sigma_B^2(t_1^*, t_2^*, ..., t_{n-1}^*) = \max_{1 \leq t_1 < t_2 < ... < t_{n-1} < L} \sigma_B^2(t_1, t_2, ..., t_n) \tag{3}$$

4. EVOLUTIONARY ALGORITHM

An evolutionary algorithm is a generic population-based optimization algorithm. It uses mechanisms which are inspired by biological evolution, such as reproduction, mutation, recombination, and selection. The solutions for the optimization problems are the individuals in the population, and the fitness function provides a metric for measuring the quality of the solutions. In this paper an EA called differential evolution (DE) has been used for optimal multilevel thresholding. Hereinafter the DE algorithm will be described in detail.

4.1 Differential Evolution

Differential evolution (DE) [6] is a population based optimization algorithm used for global optimization. It is simple, yet very effective in solving various real life problems. The idea of DE is a simple mathematical model, which is based on vector differences.

The population of the DE algorithm consists of Np individuals $\mathbf{x}_i$, where $i = 1, 2, ..., Np$, and each vector consists of D-dimensional floating-point encoded values $\mathbf{x}_i = \{x_{i,1}, x_{i,2}, x_{i,1}, ..., x_{i,D}\}$. In the evolutionary process, individuals are evolved using crossover, mutation and selection operators, which are controled by a scale factor F and crossover rate C_r. The following mutation strategies were considered in this paper:

- "rand/1":

$$\mathbf{v}_i^g = x_{r1}^g + F\,(x_{r2}^g - x_{r3}^g) \tag{4}$$

- "best/1" :

$$\mathbf{v}_i^g = x_{best}^g + F\,(x_{r1}^g - x_{r2}^g) \tag{5}$$

- "current to best/1" :

$$\mathbf{v}_i^g = x_i^g + F\,(x_{best}^g - x_i^g) + F\,(x_{r1}^g - x_{r2}^g) \tag{6}$$

- "rand to best/1" :

$$\mathbf{v}_i^g = x_{r1}^g + F\,(x_{best}^g - x_{r1}^g) + F\,(x_{r2}^g - x_{r3}^g) \tag{7}$$

For the creation of a mutant vector $\mathbf{m}_i$, three random vectors from the population are selected, defined by indexes $r1$, $r2$ and $r3$. indexes are mutually different and different from i. They are selected uniformly within the range $\{1, 2, ..., Np\}$. The scale factor F is defined within the range $[0, 2]$. The mutant vector $\mathbf{m}_i$ is obtained by adding a scaled vector difference to a third vector.

The trial vector $\mathbf{t}_i$ is then generated using the crossover rate C_r and a corresponding vector $\mathbf{x}_i$ from the population as:

$$t_{i,j} = \begin{cases} m_{i,j} & \text{if } rand(0,1) \leq C_r \text{ or } j = j_{rand}, \\ x_{i,j} & otherwise. \end{cases} \tag{8}$$

As can be seen from Eq. 8, the crossover rate C_r is defined at the interval $[0, 1]$ and it defines the probability of creating the trial vector parameters $t_{i,j}$. The j_{rand} index is responsible for the trial vector to contain at least one value from the mutant vector. After crossover, some values of the vector may fall out of bounds, which means that these values must be mapped back to the defined search space.

The next step in the evolutionary process is the selection of the fittest individuals. During the selection process the trial vector $\mathbf{u}_i$, competes with vector $\mathbf{x}_i$ from the population. The one with the better fitness value survives and is transfered to the next generation.

5. RESULTS

In this section, the obtained results are presented, and also the experimental environment is described.

5.1 Experimental environment

For the purpose of this study 4 gray scale standard test images were taken form literature. All images are of same size (512 × 512 pixels) and in uncompressed format. All images are presented in Figure 1. For evaluating the quality of the results during the evolution, the Otsu criterion for multilevel thresholding has been used (see Section 3).

All experiments were conducted with the following number of thresholds: 5, 8, 10, 12, and 15. The algorithm was coded in C++.

5.2 Image quality assessment

For evaluating the segmentation quality the well established peak-signal-to-noise ratio metric has been used. It gives the similarity of an image against a reference image based on the MSE of each pixel. It is defined as follows:

$$PSNR = 10 \log_{10} \left(\frac{255^2}{MSE} \right) \tag{9}$$

$$MSE = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} [I(i,j)) - J(i,j)]^2 \tag{10}$$

where I and J are the original and segmented images respectively.

(a) Lena (b) PEppers (c) Baboon (d) Woman

Figure 1: Test images Lena, Peppers, Baboon, and Woman.

Table 1: Comparison of best mean objective values, with mean CPU times computed by rand/1, best/1, current to best/1, and random to best/1 mutation strategies using Otsu criterion.

Test image	M	Mean objective values					CPU Time(s)			
		rand/1	best/1	current to best/1	random to best/1		rand/1	best/1	current to best/1	random to best/1
Baboon	5	0.958493	**0.958852**	0.958749	0.95865		0.01023	0.01213	0.01143	0.01103
	8	0.979531	0.97988	**0.979964**	0.979874		0.009967	0.01197	0.01207	0.01323
	10	0.985483	**0.985823**	0.985814	0.985742		0.01107	0.01267	0.0124	0.013
	12	0.989188	0.989286	**0.989347**	0.989276		0.01267	0.01263	0.01283	0.01183
	15	0.992333	0.992483	**0.992529**	0.992401		0.01227	0.01283	0.0168	0.0132
Lena	5	0.968098	0.968212	**0.96839**	0.968315		0.0115	0.001267	0.0105	0.005033
	8	0.984382	**0.985277**	0.985055	0.984949		0.01257	0.0039	0.01413	0.01327
	10	0.989208	**0.990203**	0.98978	0.98956		0.0131	0.005233	0.01417	0.014
	12	0.991884	**0.992653**	0.992276	0.992222		0.01293	0.007367	0.01543	0.01503
	15	0.994345	**0.994909**	0.994507	0.994439		0.0145	0.008767	0.0158	0.01633
Peppers	5	0.967926	**0.96823**	0.968125	0.968077		0.01173	0.01117	0.0123	0.01173
	8	0.984386	**0.984845**	0.984844	0.984592		0.01073	0.01217	0.01207	0.01263
	10	0.988955	0.989191	**0.989244**	0.989009		0.0116	0.01287	0.01317	0.01277
	12	0.991711	**0.991907**	0.99185	0.991807		0.0117	0.013	0.01307	0.01203
	15	0.994071	0.994177	0.994206	**0.994216**		0.0138	0.01337	0.0144	0.01357
Woman	5	0.963698	**0.96401**	0.96393	0.963852		0.009533	0.0109	0.0111	0.01127
	8	0.982118	0.982415	**0.982431**	0.982393		0.01077	0.0122	0.0132	0.01137
	10	0.987687	0.987929	**0.988018**	0.987783		0.01097	0.0124	0.01287	0.0117
	12	0.990841	0.99098	**0.990999**	0.990945		0.01157	0.01313	0.0126	0.0126
	15	0.993554	0.9937	**0.993711**	0.993619		0.01237	0.01377	0.0137	0.01343
Best		0	10	9	1	$\overline{time}$	0.0117(2)	0.0106(1)	0.0132(4)	0.0124(3)

5.3 Experimental results

The following settings of the DE were used for the experiments in this paper : Np was set to 50, G to 200, while the F and Cr were set to 0.9 and 0.2 respectively at the start of the evolutionary process. The stopping criteria was set to 10000 function evaluations, while each algorithm was run 30 times.

The results in Table 1 show the mean objective values, computed by each mutation strategy. Additionally the average computation times are reported. Based on mean values of the objective functions, the best results are obtained by using the best/1 mutation strategy, with the current to best/1 being a close second. When comparing average computation times the best/1 strategy again obtained the best result. In Table 2 the results of the PSNR metric is depicted (mean and best values). Based on the PSNR, the best mean value was obtained by the current to best/1 strategy, but when considering the best obtained results the rand/1 was the winner. The overall best results in both tables are marked in bold face.

6. CONCLUSION

The influence of different DE strategies has been investigated for the purpose of optimal multilevel thresholding problem. The strategies were tested on a set of four standard test images of size 512×512 pixels. The objective function during the evolution was maximizing the between-class variance proposed by Otsu. The obtained thresholds were compared on the basis of mean objective values, while also the well established metric PSNR was used to assess the quality of the segmented images. The experiments showed that the best performing strategy was best/1 when considering the mean objective values, while it also converged the fastest. On the other hand the mean PSNR metric showed that the best segmented images came from the current to best/1 strategy, but the best results were obtained using the classical rand/1 strategy.

Further work includes testing the mutation strategies by using other objective functions (i.e. Kapur's or Tsallis entropy).

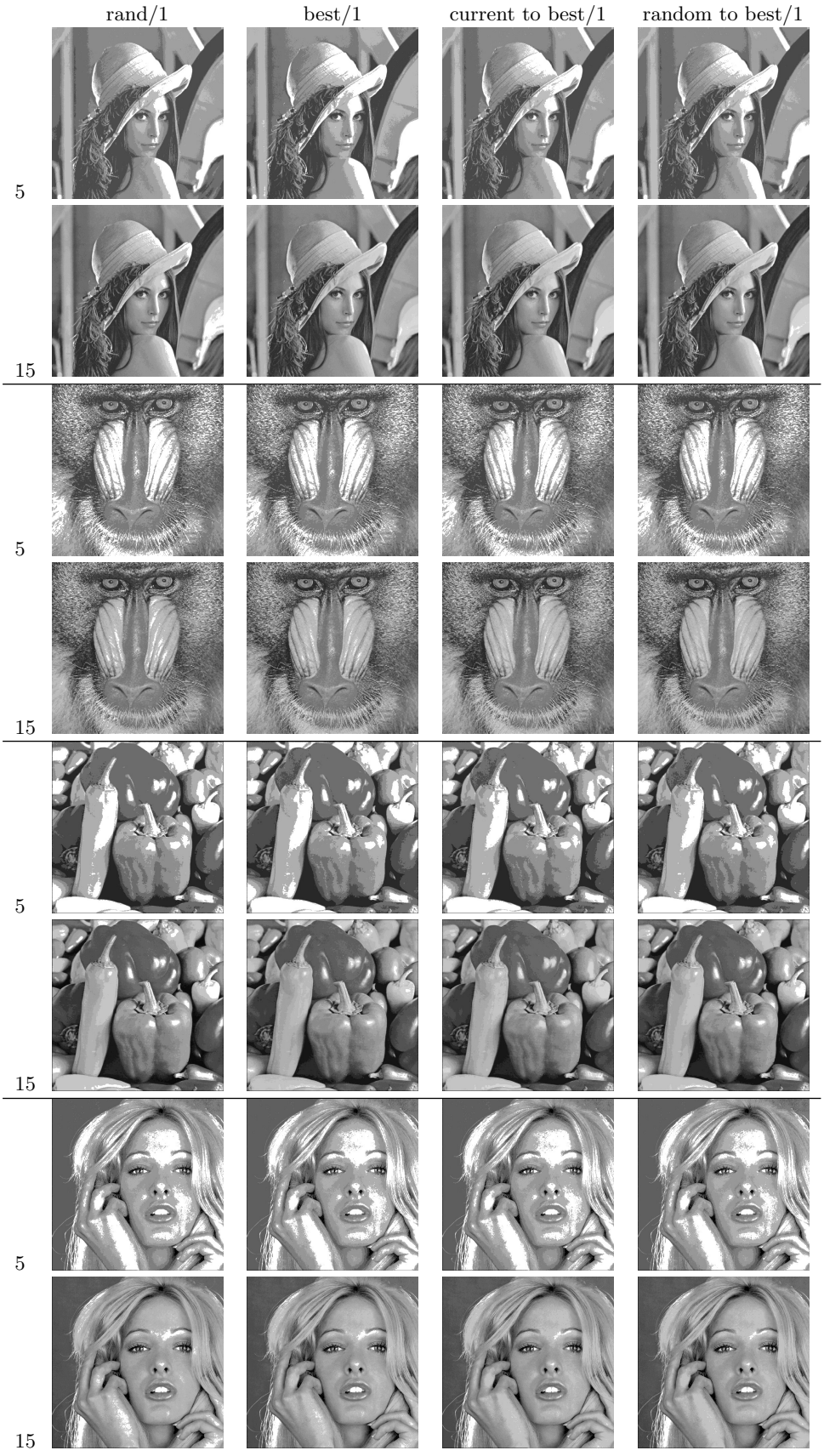


Figure 2: Test images segmented into 5 and 15 levels using different DE mutation strategies.

Table 2: Comparison of best mean objective values, with mean CPU times computed by DE, PSO, ABC, CS, and hjDE using Otsu criterion.

Test image	M	Mean PSNR values					Best PSNR values			
		rand/1	best/1	current to best/1	random to best/1		rand/1	best/1	current to best/1	random to best/1
Baboon	5	18.1344	**18.3655**	18.3612	18.1425		19.055000	19.057400	19.060400	**19.313000**
	8	21.0159	21.7262	**22.1329**	21.3687		**24.365000**	24.010100	22.890600	23.928900
	10	22.4483	22.8946	23.3143	**24.5666**		**27.102000**	26.624900	26.259100	26.598400
	12	**28.6212**	25.8498	23.5886	24.918		**29.052900**	27.718700	28.932900	28.730200
	15	24.2914	28.7404	27.4966	**28.8786**		30.405500	31.318300	32.102300	**32.506900**
Lena	5	17.3731	17.7323	**17.7754**	17.4229		**18.436100**	17.807900	17.808400	18.126300
	8	20.303	20.0005	19.9081	**20.8732**		22.146300	22.228500	**22.260300**	21.598900
	10	21.2402	21.2492	**22.5055**	21.9101		24.120400	23.964400	23.392400	**24.931900**
	12	**25.3207**	22.7806	22.7796	21.4321		26.859200	**28.144400**	26.572900	26.420300
	15	23.8341	24.7654	**27.3944**	23.1946		**30.855000**	28.871100	29.830600	29.663600
Peppers	5	17.4975	18.1431	**18.3017**	17.846		**18.612700**	18.157400	18.419600	18.447400
	8	23.442	23.2891	22.9984	**24.0323**		**24.656400**	24.653500	24.487500	24.623600
	10	**25.8584**	24.9253	25.4991	25.5769		**26.449000**	26.019600	26.139400	25.585600
	12	25.9307	25.9187	**26.665**	26.2836		26.941100	27.028100	26.770500	**27.283500**
	15	**27.4515**	25.8253	26.0986	26.9089		28.997200	30.301300	**30.381100**	28.991100
Woman	5	**21.5085**	21.0176	21.0176	21.0176		**21.508500**	21.017600	21.167400	21.089100
	8	23.1922	23.2705	23.2292	**23.436**		24.397700	24.742200	24.264600	**24.787800**
	10	25.5869	24.1715	**26.0503**	25.8962		**27.353000**	26.242400	26.260400	27.177800
	12	27.7128	**28.7043**	26.8407	26.298		28.725100	**28.754200**	28.746300	28.699800
	15	26.8773	29.841	**31.6153**	27.5219		31.824400	**31.917100**	31.646000	31.114900
Best		5	2	8	5		10	3	2	5

7. REFERENCES

[1] Adis Alihodzic and Milan Tuba. Improved bat algorithm applied to multilevel image thresholding. *The Scientific World Journal*, 2014, 2014.

[2] Ashish Kumar Bhandari, Vineet Kumar Singh, Anil Kumar, and Girish Kumar Singh. Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using kapur's entropy. *Expert Systems with Applications*, 41(7):3538–3560, 2014.

[3] Sathya P Duraisamy, Ramanujam Kayalvizhi, et al. A new multilevel thresholding method using swarm intelligence algorithm for image segmentation. *Journal of Intelligent Learning Systems and Applications*, 2(03):126, 2010.

[4] Iztok Fister Jr., Xin-She Yang, Iztok Fister, Janez Brest, and Dusan Fister. A brief review of nature-inspired algorithms for optimization. *CoRR*, abs/1307.4186, 2013.

[5] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27, 1975.

[6] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.

[7] Xin-She Yang and Xingshi He. Swarm intelligence and evolutionary computation: Overview and analysis. In Xin-She Yang, editor, *Recent Advances in Swarm Intelligence and Evolutionary Computation*, volume 585 of *Studies in Computational Intelligence*, pages 1–23. Springer International Publishing, 2015.

[8] Yudong Zhang and Lenan Wu. Optimal multi-level thresholding based on maximum tsallis entropy via an artificial bee colony approach. *Entropy*, 13(4):841–859, 2011.

Processing of next-generation sequencing (NGS) data with MapReduce and HADOOP cluster

Nándor Póka*
Applications of Informatics Department
University of Szeged
Faculty of Education
Szeged, Hungary
nandor.poka@gmail.com

Miklós Krész, PhD. †
Applications of Informatics Department
University of Szeged
Faculty of Education
Szeged, Hungary
kresz@jgypk.u-szeged.hu

ABSTRACT

Modern biological and medical research is heavily centered around molecular biology and genomics, that is a discipline in genetics that applies recombinant DNA, DNA sequencing methods, and bioinformatics to sequence, assemble, and analyze the function and structure of the complete genetic material of an organism. See references [1] and [2] for more information. Researchers came to the point where they have to deal with several (or even hundreds of) gigabytes of data. The rapid growth of cloud computing and local clustering of common machines present great opportunity, thus high-performance, distributed computing is no longer the privilege of large high-tech companies. Since manually orchestrating parallel computing is circumstantial, therefore some sort of framework is used to handle parallelization and data management. One of such frameworks is MapReduce and its open source implementation Hadoop. De-novo assembly is a data intensive problem that can benefit from the use of clusters and distributed computing. Current solutions like Velvet assembler (ref [3]), are multi-threaded at best. Our aim was to create an algorithm for de novo genome assembly, using MapReduce and Hadoop. Similar has been done by Yu-Jung Chang et al. (ref [4]), their key idea is called âĂŸedge alignmentâĂŹ, but their solution can only handle a narrow, predefined set of data. In this work we present a de Bruijn graph based algorithm that is able to completely reconstruct an artificially generated random sequence and assemble large segments of real life test sequence. In the following chapters we will overview key biology, bioinformatics and informatics concepts necessary for understanding this work, followed by the compact overview algorithm.

*Is also PhD candidate at Department of Medical Biology, Faculty of Medicine, University of Szeged, author carried out algorithm design, development and testing

†Supervisor and Head of Department, to whom correspondence should be sent

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]: Biology and Genetics—*Genomics and Bioinformatics*; D.1.3 [**Programming techniques**]: Concurrent Programming—*MapReduce, HADOOP*; D.1.1 [**Programming techniques**]: Applicative (Functional) Programming

General Terms

Algorithm design

Keywords

Bioinformatics, MapReduce, HADOOP

1. INTRODUCTION TO BIOLOGICAL CONCEPTS

The DNA - deoxyribonucleic acid - is the genetic material of all living organisms. It is made of four different nucleobases, namely: adenine (A), cytosine (C), guanine (G) and thymine (T). These nucleobases are able to form chemical bonds with each other by the following strict rules [1]:

- Pairing only occurs between purine (A and G) - pyrimidine (C and T) pairs (no intragroup pairing)
- Adenine can only pair with thymine
- Cytosine can only pair with guanine

Also, owing to the chemical properties of the sugar-phosphate backbone of the nucleobases, DNA can form virtually infinite linear chains. These long linear DNA molecules are called DNA strands. The DNA strands have directionality - the 5' (five-prime) end is where the phosphate group is free and the 3' (three-prime) end is where the hydroxyl-group is free on the backbone. Naturally DNA forms a double-helix structure, in which two strands of DNA are paired up by their bases - the two strands of a double-helix are fully complementary to each other. By convention, a single strand of DNA is strictly written in 5' to 3' direction. The size / length of a given DNA is measured in bases or base-pairs with standard prefixes for indicating thousand (kilo-), million (mega-), and billion (giga-) - kilobases, megabases, gigabases. For further details see references [5], [6], [7]. Biological functions

[1]This is generally referred as base pairing

in living cells are mostly carried out by proteins that are encoded by genes on the DNA. However, the genetic material has many other types of sequences, for example: regulatory sequences, repeats of various length (up to several kilobases) and number of occurrence, special telomer sequences at the end of chromosomes, pseudo-genes (sequences almost identical to functional genes) etc. The protein coding portion of the human genome is approx. 1-2%. Since most eukaryotic organisms (organisms with cell nucleus) are diploid (the genome is present in two copies) natural variations between copies may occur, not to mention some plants where the genome may be present in 6-8 copies.

2. INTRODUCTION TO NEXT-GENERATION SEQUENCING (NGS) AND BIOINFORMATICS

2.1 Next-Generation Sequencing

The term sequencing refers to the process through which researchers obtain the nucleotide order of a specific DNA molecule. The first sequencing method was introduced in 1975, by Sanger and co-workers. This method was slow and could only obtain about 400 bp length of information at a time. Since then, the 3rd generation of sequencing methods were developed and the 4th generation is already on its way. In this work by the term NGS we mean 3rd generation sequencers of which the first was introduced in 2005. The common properties of these platforms are:

- Most of them utilize sequencing by synthesis method
- Simultaneously sequence many - up to millions of - templates
- Produce millions of short reads [2] (typically 30-100 bases, rarely 200-300 depending on the platform), resulting in hundreds of millions (or even a few billions) high-quality bases
- Utilization of universal adapters, to prime the synthesis reaction and also to immobilize templates

During sequence by synthesis, information is obtained through synthesizing a new complementary strand to the template and reading the information of the newly incorporated base. Enzymes carrying out DNA synthesis cannot produce new DNA strands on their own, they require a template they can "copy". Labelled bases can be recoded when incorporated into the newly forming strand and due to the complementarity, the sequence of the two strands is interchangeable.

The general schema of an NGS method is to: acquire small fragments (templates), separate, then sequence them. There are various methods to generate such amount of templates (eg. random priming, PCR amplification, breaking of DNA), and each may produce characteristic errors that need to be corrected later on, during data processing. Also it is important to note that the different NGS platforms have different, yet characteristic error types and various error rates. The high data throughput, achieved by massive parallelization, enables sequencing of entire genomes within days (although the evaluation of the data obtained may take weeks). One drawback of NGS technologies compared to conventional methods is the relatively high error rate per read, but these techniques rely on quantity over quality and eventually, due to the immense number of reads produced, incorrect / faulty reads can be either corrected or ignored. More details can be found in references [8] and [9].

[2]In bioinformatics the sequence obtained from a single piece of DNA molecule is called a read

2.2 Insights into Bioinformatics and de novo assembly

Data analysis starts with âĂIJcleaningâĂİ the data of faulty reads and artifacts left behind by template generation methods etc. During sequencing, determining the identity of a single nucleotide based on the signal information, is termed base calling. Since no system is perfect, base calling has an error rate. Each platform has its unique measurement for raw base calling quality, but for the final output it is converted to a so called Phred score. Phred score by definition is: $-10 \times \log_{10} P$, where P is the probability of the given nucleotide being called incorrectly. Raw sequences coming directly from the sequencers have typical Phred scores of 30 to 40 and rarely exceed 60, higher values may occur after processing of raw data. A Phred score of 30 is considered good, since most NGS methods produce reads far shorter than 1000 bases, and in general, bases with Phred score lower than 20 should be ignored.

Coverage (the number of times a single position is read / sequenced, preferably by overlapping reads and not identical ones) is another important attribute of sequencing experiments. Coverage can be applied to single bases or positions, but without data processing, only an approximation of average coverage can be given. If the length of the target sequence is known (or there is an acceptable approximation), the number of reads required for desired average coverage can easily be calculated. For most applications 20-30× coverage is suitable, but for example, identifying rare variants, and de novo assembly often require coverage up to 80-100 times. Actual coverage may largely differ from theoretical coverage, because generation of the template is non-random. When analyzing data, regions with low or too high coverage must be handled with caution.

Sequence information is usually stored in FastQ format, that is a regular text file, and contains short meta data regarding the sequence (eg. read id, sequence accession id, date/time, platform / machine specific information) the actual sequence, and the corresponding Phred quality score encoded as ASCII characters instead of numbers.

Filtering experimental data is rather data-, than CPU-intensive as raw data from sequencers can reach 600 GiB per run, and computations required for 'data cleaning' are relatively simple. Yet, as all applications handling vast amount of data, this step can also benefit from a distributed system. There are many applications of NGS, but we will only introduce de novo assembly in this work. De novo genome assembly is when the reference genome for an organism is built, or in general, assembly of the genome for any species, for validation etc. As from the technical / informatics point of view, this is analogous to reconstructing a large (unknown) string

from its sub-strings. One of the key difficulties is, that the genetic material is far from random sequence of letters. As mentioned before, the genetic material has many different types of sequences and also variations between the copies of the genome that naturally occur, must be taken into consideration when reconstructing a genome (coverage plays important role in filtering out plausible and artificial variants of the same sequence). Due to these characteristics of the genetic material, automated full reconstruction of a reference genome is quite difficult, even current solutions aim to create as large contiguous fragments (contigs) as possible, and then these contigs are assembled manually or algorithmically using references from closely related species (although even in the latter case, manual proofreading is still a must).

Modern genome assemblers use de Bruijn digraphs or methods based on them. The concept behind de Bruijn digraphs in de novo genome assembly (see details in reference [10]), is to use these k-mers as nodes, and connect the nodes if they overlap all but one base, that can be done with a computationally simple exact matching. The use of k-mers is very useful, because at given genome size (L) and fixed read length, the number of k-mers is also fixed, thus using more reads to achieve higher coverage does not result in increased complexity. The n-dimensional k-ary de Bruijn digraph (D(k,n)) is a directed graph with k^n nodes that are labelled as an n-digit k-ary number (the k-mers of the original sequence), and node $a_1a_2a_3a_4...a_n$ is adjacent to k other nodes that are $a_2a_3a_4...a_n\alpha$, where α is from $\{0, 1, 2, ...k-1\}$. Due to biological constraints genomes rarely contain all possible subsequence even for short read lengths, thus a slightly different graph is used instead, that is called de Bruijn based digraph, where the principle is the same, but nodes have different in-, and out-degrees. A single Eulerian-path can reconstruct the original sequence in such graph. However, in large graphs bubbles and tips may occur and to avoid these, further transformation of the graph is required. Let S be a genomic sequence of length L. A generalized de Bruijn digraph (GDB) G(p',e',K,T,H) is a directed graph with p' nodes that are labelled as $b_1b_2b_3...b_k$ for some k in $K = \{k_1k_2k_3...k_n\}$ set of integers and $k_i < L$ and e' arcs labelled as $(t, c_1c_2c_3...c_k)$ with t in $T = \{t_1t_2t_3t_i\}, t_i \leq k-1$, and some h in $H = \{h_1h_2h_3...h_s\}, h_i \leq L$ in a way that node $b_1b_2b_3...b_k$ is adjacent to $b_(t+1)b_(t+2)...b_kc_1c_2c_3...c_h$, where b_1 and c_i are from the set $\{A, C, T, G\}$. It is clear from the definition that nodes in the GDB have variable length. Another observation can be made: in the GDB a node can be shifted t spaces to the left (whereas in DBB $t = 1$, thus only a single base can be shifted), and a sequence of various length of h can be concatenated form the right. An Eulerian-path in the graph gives back the original sequence.

3. THE MAPREDUCE FRAMEWORK AND THE DEVELOPED ALGORITHMS

MapReduce can refer to a programming paradigm and an execution framework at the same time. MapReduce as a programming paradigm has it roots in the functional programming, where higher-order functions exist, which can accept other functions as arguments. This paradigm defines a two phase general logic to process large amounts of data. The data is first transformed by Mappers in a programmer defined function (in a parallel manner) to yield intermediate results (the immediate results are sorted and grouped by their key, and values with the same key are passed to the same reducer), which is then aggregated - again by a programmer defined function - by Reducers. The actual processing is coordinated by the execution framework (including the division of data to several splits and gathering intermediate data). Although, it may seem that this two phase processing structure is very restrictive, in reality many algorithms can be carried out this way, especially if complex problems are decomposed to a series of MapReduce jobs. In MapReduce key-value pairs are the basic data structure and both keys and values can be primitives or arbitrarily complex structures. The data is stored in a distributed file system (HDFS) across data nodes. To avoid moving large amount of data around the cluster, Hadoop brings the code to be executed to the data and tries to schedule tasks on nodes where the required data is present, to achieve data locality. It is important to note that each Mapper and Reducer runs in isolation and there are no real global variables in Hadoop. More information about Hadoop and MapReduce can be found at the official website: `https://hadoop.apache.org/`.

To reduce file size and speed up computation we have developed a compression method that can reduce pure sequence information to 25% of its original size, and can compress (lossy compression) FastQ files to 50% of their size, on average. The compression stores only a subset of the original Phred scores, and the practical idea behind this, is that in most cases the range 20 to 83 is more than enough handle every scenario as Phred scores below 20 is so poor and scores above 80 are so good that values beyond these can be cut off. Also ambiguity codes (letters that may represent more than one nucleobase see ref [11]) are not considered useful in de novo assembly, therefore they are omitted from this compressed file format. This small program was developed separately, but can easily be expressed as a MapReduce job.

The de novo assembly program is written in Java since Hadoop itself is implemented in this language. The algorithm itself is a series of MapReduce jobs, where each subsequent job takes its input from the previous job's output stored on the distributed file system. Currently the algorithm contains four jobs that do the following tasks:

1. counts unique reads present in the input
2. produce k-mers of user defined size from reads
3. assembles of contigs from k-mers
4. assembles contigs into a single genome (if possible). - Currently in development.

Processing of input arguments, configuring separate jobs and submitting them to the framework is orchestrated by a MapReduce Driver. This is a separate class that holds the main entry point for executing the Java program (Hadoop accepts a collection of jobs in jar format).

The first job is a simple word count implemented in MapReduce fashion. The Mapper emits a sequence and a integer 1, and the reducer counts the unique sequences and emits each sequence and an integer representing the number of occurrence. This serves as input for the second job, that produces

k-mers ("the nodes" of the de Bruijn based graph) of user defined length. The first two jobs prepare data for the third job which assembles k-mers into as large contigs as possible (this is carried out in the Reduce phase of the job). A simple but efficient trick is utilized to ensure that k-mers that most likely belong together, end up at the same Reducer and can be assembled into a contig. Throughout the algorithm the sequence information is used as key, thus k-mers arrive to this job in lexicographically sorted format, and since a strict overlapping policy is used (adjacent "nodes" must overlap all but one base), sequences that are close to each other in the data, are most likely belong together. When data is read by the framework from the disk, it is split to smaller chunks and these data splits are sent to Mappers. In this job each unique Mapper uses a unique key when it emits key-value pairs, thus when the framework groups the intermediate results, data that was processed by a particular Mapper gets grouped together and sent to same Reducer.

We neither explicitly build any graph nor contigs are assembled via an Eulerian path, but rather the connectivity of the de Bruijn based graph is used. The Reducer iterates through the data is was assigned to, and for each k-mer it collects the start- and end-slices. These are stored in TreeMap structures and the k-mers ("nodes") are stored in a set. After this, the contig assembly is carried out by an iterative process. The first element of the node set is taken and attempt is made to extend it in both directions. Also forking (multiple adjacent "node" is present) is kept track of. This extension step is continued as long as the current node can be combined with another at least in one direction. The nodes used in the extension are removed from the set. If a node cannot be extended it is saved and a new starting node is taken from the node set.

4. EVALUATION OF CORRECTNESS

The software / assembly algorithm was first tested for correctness by using artificially generated test data, using the small utility tool developed alongside the main program. A random sequence of the four letters A,T,C,G was generated in arbitrary length and then random subsequences were generated as reads in the amount to reach 5-10 times average coverage. Given this input, the algorithm is capable to fully reconstruct the original sequence. For the next stage of testing, we chose to use the genome of the Pseudorabies virus (PRV). The Kaplan strain of PRV has genome size of approximately 143kbp (kilobase pair). It serves our purpose well, because it is small, it is very tightly packed (approx. 70 are contained within the genome and it has nested and overlapping genes), it has two large repeat regions that have opposite orientation. Due to these features it poses a great challenge for a de novo genome assembly algorithm such as detailed above. Using 40 base pair reads with 38 base pair long k-mers the algorithm does fairly well. It produces 238 contigs, with maximal contig length of 58210bps, minimum of 47bps and an average of 12020bps. With some manual work the original sequence can be assembled.

5. CONCLUSION

Our aim was to create a de novo assembly algorithm using the MapReduce paradigm and Hadoop as a MapReduce execution framework. Our current work presents the development of such program along with a useful compression utility program. Although the de novo assembly algoritm is still under development it already shows promise and great results. Further work includes assembly of contigs into a full genome if possible, add data cleaning part to the suite to process raw data and widen the accepted type and format of input data.

6. REFERENCES

[1] Stefan K. Bohlander ABCs of genomics ASH Education Book December 6, 2013 vol. 2013 no. 1 316-323

[2] Ombrello MJ, Sikora KA, Kastner DL. Genetics, genomics and their relevance to pathology and therapy. Best practice & research Clinical rheumatology. 2014;28(2):175-189. doi:10.1016/j.berh.2014.05.001.

[3] Velvet: algorithms for de novo short read assembly using de Bruijn graphs. D.R. Zerbino and E. Birney. Genome Research 18:821-829.

[4] Y.-J. Chang, C.-C. Chen, C.-L. Chen, and J.-M. Ho, A de novo next generation genomic sequence assembler based on string graph and MapReduce cloud computing framework., BMC Genomics, vol. 13 Suppl 7, no. Suppl 7, p. S28, Jan. 2012.

[5] R. Dahm, Discovering DNA: Friedrich Miescher and the early years of nucleic acid research, Human Genetics, vol. 122, no. 6. pp. 565-581, 2008.

[6] J. D. Watson and F. H. C. Crick, Molecular structure of nucleic acids, Nature, vol. 171, no. 4356, pp. 737-738, 1953.

[7] F. H. C. Crick, The origin of the genetic code, Journal of Molecular Biology, vol. 38, no. 3. pp. 367-379, 1968.

[8] A. Alekseyenko et al, NEXT-GENERATION DNA SEQUENCING INFORMATICS. New York, New York: Cold Spring Harbor Laboratory Press, 2013.

[9] L. Liu, et al., Comparison of next-generation sequencing systems, Journal of Biomedicine and Biotechnology, vol. 2012. 2012.

[10] Phillip E C Compeau, Pavel A Pevzner & Glenn Tesler, How to apply de Bruijn graphs to genome assembly Nature Biotechnology 29, 987-991 (2011) doi:10.1038/nbt.2023

[11] http://www.bioinformatics.org/sms/iupac.html.

Developing a web application for DNA data analysis

Aleksandar Tošić
University of Primorska Faculty of Mathematics, Natural Sciences and Information Technologies (UP FAMNIT)
Slovenia, Koper
aleksandar.tosic@upr.si

ABSTRACT

The article outlines the phases of developing a custom web application, for cultivar DNA analysis. The nature of the data forces the database to expand in dimensionality, which effects performance. We review the current statistical methods for analyzing genotypization data in population genetics, and discuss a clustering approach for identifying individuals using multidimensional convex hulls to represent clusters.

Keywords

Database, Cultiavar, Population genetics, Clustering, Convex hull

1. INTRODUCTION

Genetics holds high hopes for new scientific discoveries. A promising method is to find ways, in which information is inherited within and between spices. Molecular markers are commonly used for managing genetic resources of organisms and their relatives. Genotyping using molecular markers is particularly appealing because it significantly accelerates the identification process by fingerprinting of each genotype at any stage of development of a plant. Among many DNA marker systems microsatellites combine several properties of an ideal marker such as polymorphic nature and information content, co-dominance, abundance in genome, availability, high reproducibility, and easy exchange of data between laboratories [7] . With the popularity of microsatellites several studies addressed the problem of consistency of genotyping data in different laboratories. The inconsistencies appear mostly due to different equipment and chemicals used to amplify the DNA. One of the main goals in population genetics is to identify a species using genotpying data but due to before mentioned inconsistencies this is not a straightforward task. The existing statistical indicators offer some insight on the data but in most cases require researchers to determine the identity. In this paper we present a database application as a tool that helps researchers view and analyze data. Besides implementing the usual statistical indicators, we encourage researchers to identify their individuals and compare them with the existing data. This way we gather expert's knowledge that will be used to improve the error rates of automatic individual identification using a clustering approach.

The department of Mediterranean Agriculture on the Faculty of Mathematics and Information Technology provided us with genotyping data of Slovenian Olive trees. We combined our data with the published datasets from [1], [3], and a Spanish SSR database [11] to create a bigger dataset.

We implemented a database application that computes most commonly used statistical identifiers for analyzing microsatelite markers. We implemented an easy upload tool, which allows researchers to upload their datasets, retrieve the statistical identifiers, and compare it with other datasets. The application also charts all the results for easier analysis. Ideally, the database containing genotyping data of all sequenced plants would be publicly available. Researchers could upload their genotyping data and the application would identify the cultivars. We tackle the problems of cultivar identification in large databases and discuss possible solutions.

2. BACKGROUND

We devote this chapter to quickly explain the nature of the data obtained from genotypization.

For basic understanding we say that a gene is a subset of the DNA sequence that encodes information needed to form a protein. Genes are inherited from parents and provide information needed to for the organism to evolve. This article focuses on genotyping, which is the process of determining differences in genetic make-up of an individual, by examining the DNA sequence and comparing them to other individuals. The goal is to structure the data for each individual (in our case cultivar) in order to compare groups of individuals and find relationships between them. Every cultivar is represented by a number of loci. Loci are specific locations of a gene, DNA, or positions on a chromosome. These positions are found to hold important information of an species or an individual.When sequencing an individual researchers usually check, which markers are most used amongst the research community; hence, some individuals have missing data on some loci. Each locus contains two (sometimes even more) alleles stored as an integer representing their distance. The distances are crucial for identifying changes in a genetic fingerprint.

3. APPLICATION

The application was developed using a conglomerate of frameworks and open source libraries such as Googles AngularJS, Codegniter and Jquery. A modern choice of DOM manipulation tools with a small footprint yet powerful server side framework was used to make an architectural model. The architectural pattern is somewhat different from the

traditional Model-View-Controller(MVC). Instead a Model-Controller-Controler-View (MCCV) approach is used like illustrated in Figure 1.

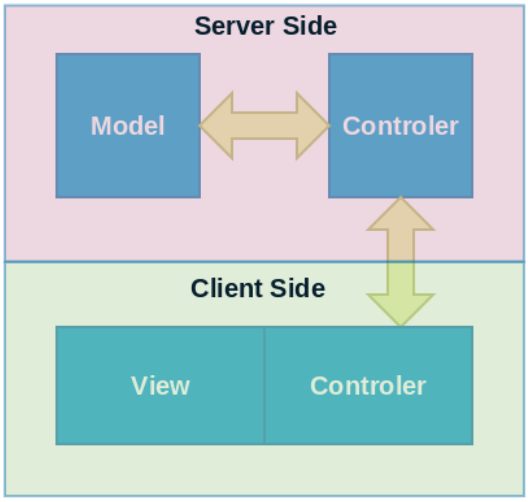


Figure 1: Architecture model

3.1 Client side

The client side was built with AngularJS that provided an entirely new paradigm in web application development. The main feature of the client side is to offer a user a simple tool of constructing queries for viewing parts of the database or dataset. A query is constructed using an arbitrary number of relations between attributes. The query construction is all done client side. On query execution the constructed query is sent to the server-side that examines examines the code for possible SQL injections before translating it into an SQL statement. Communication between the server side and the client side is done with the help of Representational State Transfer (REST) interface. Exploring the data is very simple trough the powerful filters that use fuzzy searching to filter data on every dimension separately. The client side is also capable of drawing charts of all the statistical indicators like illustrated in Figure 2.

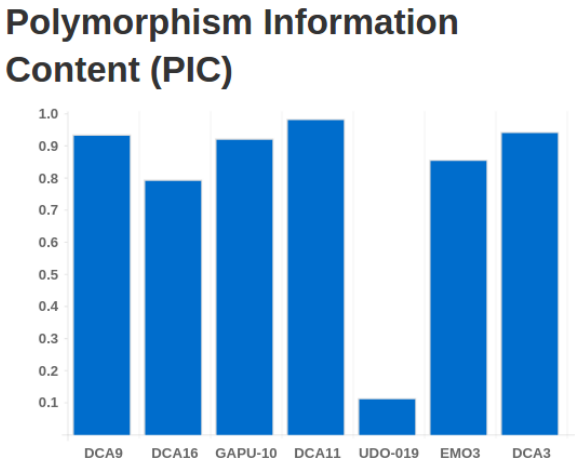


Figure 2: Charting PIC values on all Loci

3.2 Server side

The server side is implemented in PHP, and features dirty checking of input data for security, logging capabilities and data transformation. The server must compute all of the statistical indicators on the entire database, but also supports arbitrary subsets defined by the user. Due to the dynamically defined data the previously computed values can not be cached and must therefore be computed every time.

The server also handles dataset uploads by first checking the uploaded dataset for consistency, security, and validity of the format. After the checking is done the dataset is parsed and inserted into the database.

4. STATISTICAL INDICATORS

In population genetics, statistical indicators are commonly used to gain insight into a population's biology and history [14] . To obtain these indicators, researchers usually use a variety of tools and programs [12] that are usually command line tools with poor user experience. We implemented the most commonly used indicators to allow easier comparison of datasets. We dedicate this chapter to outline each of the indicators and briefly explain their usage.

4.1 Observed Heterozygosity (H_o) and Expected Heterozygosity (H_e))

Loci that have both or all alleles of the same length are called homozygous and vice versa, loci with different alleles are called heterozygous. Heterozygosity is of major interest to researchers of genetic variation in natural populations. It is often one of the first indicators used to asses gene diversity in a dataset. It tells us a great deal about the structure and even history of a population. Expected heterozygosity is mathematically defined as

$$1 - \frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{k} p_i^2$$

where the first summation is for the i_{th} of m loci and the second summation is for the alleles i_{th} allele.

Often, the observed level of heterozygosity is compared to the expected level under Hardy-Weinberg equilibrium [4]. Low heterozygosity means there is little genetic variability, which can be associated with a small population size but it is usually due to inbreeding, whereas if its higher it indicates the possibility of mixing two previously isolated populations.

4.2 Unique Allels (n_u)

Unique alleles can be calculated for each individual genotype or an entire population. Unique alleles in an individual genotype are usually observed when we seek a possible mutation that introduced a new allele in the genotype. Unique alleles in a population are usually checked when trying to determine if breeding between two populations occurred. This is more likely to occur in plants when they get pollinated by bees that carry genes in pollen grains. In order to make an educated guess on why unique alleles are present, researchers usually check if the organisms are geographically distant and

hence, eliminate the possibility of unique alleles occurring because of different populations breeding. This is one of the reasons why we implemented the possibility to include GPS coordinates of each sample genotyped. Comparison between individuals can therefore be done also by viewing geographical locations where samples were obtained.

4.3 Allele Frequency

Allele frequency is the number of copies of a particular allele divided by the number of copies of all alleles at the genetic place (locus) in a population. It is usually expressed as a percentage. In population genetics allele frequencies are used to depict the amount of genetic diversity at the individual, population, and species level. It is also the relative proportion of all alleles of a gene that are of a designated type. The frequencies of alleles at a given locus are usually graphed as a histogram or allele frequency spectrum or a allele frequency distribution. We have done the later. In population genetics it is important to study the changes in the distribution of frequencies of alleles that usually occur on account of genetic drift, mutation, or migration of a population [5].

4.4 Polymorphism Information Content (PIC)

The PIC value is mainly used to assess the diversity of a gene or DNA segment in a population, which throws light on the evolutionary pressure of the allele and the mutation the locus might have undergone over a time period. The PIC value will be close to zero if there is little alleleic variation, and it can reach a maximum of 1.0 if a genotype has only new alleles which is a rare phenomenon. PIC values are defined as:

$$PIC_i = 1 - \sum_{j=1}^{n} P_{ij}^2$$

where PIC_i is the polymorphic information content of a marker i, p_{ij} is the frequency of the j_{th} pattern for marker i, and the summation extends over n patterns.

4.5 Effective number of alleles (n_e)

This measure is the number of equally frequent alleles it would take to achieve a given level of gene diversity. That is, it allows us to compare populations where the number and distributions of alleles differ drastically. The formula is:

$$A_E = \frac{1}{r} \sum_{j=1}^{r} \frac{1}{1 - D_j}$$

where D_j is the gene diversity of the jth of r loci.

4.6 Probability of Identity (PI)

Identifying individuals is of great importance in conservation genetics and molecular ecology. Statistical estimates are commonly used to compute the probability of sampling identical genotypes. These statistical estimates usually assume a random association between alleles within and among loci. Probability of Identity is a popular estimator [10], [6], [8], [13] that represents the probability that two individuals randomly drawn from a populations have the same genotype at multiple loci. The PIC estimator is defined as:

$$P_{(ID)} = \sum_{min}^{max} p_i^4 + \sum \sum_{min}^{max} (2 p_i p_j)^2$$

where p_i and p_j are the frequencies of the i_{th} and j_{th} alleles and $i \neq j$ [9].

We implemented the $P_{(ID)}$ for each locus using allele frequencies from a population sample.

5. FUTURE WORK

The database application is a basic tool that aids researchers to gain more insight in their genotypization data, but they key problem remains unsolved. Individual identification is challenging because the allele lengths are somewhat inaccurate due to mutation, but even more importantly due to different laboratory equipment and processes used for amplifying DNA. We explored different ideas one of which is clustering using convex hulls. The data is by nature multidimensional and hence, the clustering is subject to the curse of dimensionality [2].

Computing convex hulls in high dimensions is exponential on the number of dimensions. Arguably, in our case we are not required to compute the convex hulls, instead we only need to answer if a point would fall inside a convex hull of a set of points. The following are a few approaches/ideas on how to efficiently answer such queries.

1. Finding a feasible solution to a Linear Program.
   For a set $A = (X[1]X[2]...X[n])$
   Solve the following Linear problem:

   minimize (over x) : 1
   subject to: $Ax = P$
   $x^T * [1] = 1$ $x[i] \geq 0$ for all i

   where:
   x^T is the trasnpose of x
   [1] is the all -1 vector

   The problem has a solution if and only if the point is in the convex hull.

2. Solving m linear equations with n unknowns.
   We use the following property:

   Any vector point v inside a convex hull of points $[v_1, v_2, ..., v_n]$ can be represented as a $\sum(k_i * v_i)$ where $0 \leq k_i \leq 1$ and $\sum(k_i) = 1$. Correspondingly, no point outside of the convex hull will have such representation. In m-dimensional space, this will result in to a set of m linear equations with n unknowns.

3. A point lies outside of the convex hull of a set of points, if and only if the direction of all the vectors from it to all other points are on less then one half of a circle as

can be seen in Figure 3. The dots represent a set of points $1..n$ in a cluster, the query point is the center of the circle. If the maximum angle formed by the vectors is greater then $180°$, then the query point is outside of the convex hull of the set of points. In two dimensions this is very intuitive and the time complexity is $\mathbb{O}(m * n^2)$. This property is very useful in two dimensions, but it is quite tricky to check in m dimensions. Probably not easier then solving αm linear equations with n unknowns.

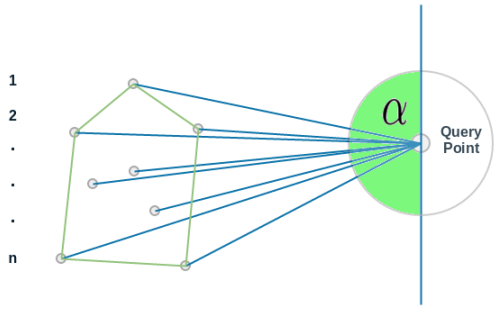


Figure 3: Convex hull membership

6. CONCLUSION

Quite a few methods exist for checking membership of a point in a convex hull without actually computing the convex hull. In future work we will analyze the accuracy of different clustering algorithms compared to our convex hull clustering idea. In cases where convex hulls overlap we will explore the possibility of transforming one or more convex hull to a concave hull to avoid overlapping and test if such an approach results in over-fitting. A few key points will be handling missing data and weighting dimensions depending on how important/informative a locus (dimension) is for each individual.

7. ACKNOWLEDGMENT

We thank our colleagues from the department of Mediterranean Agriculture who provided data, insight and, expertise that greatly assisted the research.

8. REFERENCES

[1] L. Baldoni, N. G. Cultrera, R. Mariotti, C. Ricciolini, S. Arcioni, G. G. Vendramin, A. Buonamici, A. Porceddu, V. Sarri, M. A. Ojeda, I. Trujillo, L. Rallo, A. Belaj, E. Perri, A. Salimonti, I. Muzzalupo, A. Casagrande, O. Lain, R. Messina, and R. Testolin. A consensus list of microsatellite markers for olive genotyping. *Molecular Breeding*, 24(3):213–231, 2009.

[2] R. E. Bellman. Dynamic Programming. *Reference Reviews incorporating ASLIB Book Guide*, 17(301):1–39, 2003.

[3] S. Doveri, F. Sabino Gil, A. Díaz, S. Reale, M. Busconi, A. da Câmara Machado, A. Martín, C. Fogher, P. Donini, and D. Lee. Standardization of a set of microsatellite markers for use in cultivar identification studies in olive (Olea europaea L.). *Scientia Horticulturae*, 116(4):367–373, 2008.

[4] A. W. F. Edwards. G. H. Hardy (1908) and Hardy-Weinberg equilibrium, 2008.

[5] Hartl DL, Clark G, and Andrew G. *Principales of population genetics*. Sinauer associates Sunderland, 1997.

[6] M. H. Kohn, E. C. York, D. A. Kamradt, G. Haught, R. M. Sauvajot, and R. K. Wayne. Estimating population size by genotyping faeces. *Proceedings. Biological sciences / The Royal Society*, 266(1420):657–663, 1999.

[7] P. Kumar, V. K. Gupta, A. K. Misra, D. R. Modi, and B. K. Pandey. Potential of Molecular Markers in Plant Biotechnology. *Plant Omics*, 2:141–162, 2009.

[8] L. S. Mills, J. J. Citta, K. P. Lair, M. K. Schwartz, and D. A. Tallmon. Estimating animal abundance using noninvasive DNA sampling: Promise and pitfalls. *Ecological Applications*, 10(1):283–294, 2000.

[9] D. Paetkau and C. Strobeck. Microsatellite analysis of genetic variation in black bear populations. *Molecular ecology*, 3(5):489–495, 1994.

[10] J. Z. Reed, D. J. Tollit, P. M. Thompson, and W. Amos. Molecular scatology: the use of molecular genetic analysis to assign species, sex and individual identity to seal faeces. *Molecular ecology*, 6(3):225–234, 1997.

[11] I. Trujillo, M. A. Ojeda, N. M. Urdiroz, D. Potter, D. Barranco, L. Rallo, and C. M. Diez. Identification of the Worldwide Olive Germplasm Bank of Córdoba (Spain) using SSR and morphological markers. *Tree Genetics and Genomes*, 10(1):141–155, 2014.

[12] Wagner H-W and Sefc K-M. Identiy 1.0, 1999.

[13] J. L. Waits and P. L. Leberg. Biases associated with population estimation using molecular tagging. *Animal Conservation*, 3:191–199, 2000.

[14] S. Wright. The Interpretation of Population Structure by F-Statistics with Special Regard to Systems of Mating. *Evolution*, 19(3):395–420, 1965.

Personalization of intelligent objects

Aleksandar Tošić [1] aleksandar.tosic@upr.si
Aleksandar Todorović [1] aleksandar.todorovic@upr.si
Tanja Štular [1] tanja.stular@upr.si
Nejc Radež [3] nejc.radez@gmail.com
Tadej Krivec [3] tadej.krivec@gmail.com
Tjaž Brelih [2] tjaz.brelih@gmail.com
Neli Kovšca [1] neli.kovsca@gmail.com
Anja Rudež [1] anja.rudez@gmail.com
Aleš Papič [2] ap5327@student.uni-lj.si
Vladan Jovičić [1] vladan.jovicic@upr.si
Marko Palangetić [1] marko.palangetic@upr.si

[1] University of Primorska, Faculty of Mathematics, Natural Sciences and Information Technologies
[2] University of Ljubljana, Faculty of Computer and Information Science
[3] University of Ljubljana, Faculty of Electrical Engineering

ABSTRACT

Nowadays houses take advantage of a variety of IT controlled devices. Traditionally these devices were controlled by programmable logic controllers (PLC) with the devices as their periphery. Nowadays capable and cheep microcomputers are available and permit use of standards, open source development environments, that further reduce the price and improves the use. Furthermore, PLC based design did not put a lot of emphasis on security, which is easier to achieve through standard services and protocols.

In previous work [7] we showed how we can use cheep, readily available hardware and software to augment living environments with security and comfort. In this article we present a system for personalization of objects equipped with such hardware.

Categories and Subject Descriptors

H.4 [**INFORMATION SYSTEMS APPLICATIONS**]; C.3 [**SPECIAL-PURPOSE AND APPLICATION-BASED SYSTEMS**]

General Terms

Personalization, Inteligent object, PLC, Microcontroller, Smart House

Keywords

PLC, Inteligent object

1. INTRODUCTION

Home automation is fast growing market with many useful applications. Despite the increasing demand the market is still relatively new and is therefore expensive for early adopters.Fast development of cheep electronic, controllable periphery, and open-source software has rendered home automation relatively inexpensive. Arguably, personal home automation is only a subset of all the different objects that can be electronically controlled, but equipping an object with electronics does not make it intelligent. In order to achieve automatic control an intelligent object must be able to adapt to different circumstances. To make this possible we propose a system that controls the periphery units by taking into account the user preferences. Ideally, an intelligent object would identify users and adapt its output parameters to best suite all of the users. We explore the possibility of using intelligent objects for individuals who are physically or mentally weak [2], [3], [6]. The psychological advantage is mostly that the individuals can avoid living in a public institution (e.g. retirement homes), but at home among, their family. Smart houses and technology is known to increase safety and independence of a person, which would make individuals more inclined to accept intelligent objects [1].

2. SYSTEM ARCHITECTURE

The system is built using a combination of open-source software and protocols. The system is modular and very robust so that it can be adapted to any intelligent object. The system features two servers, which both implement a Representational State Transfer (REST) interface for efficient communication amongst other modules. The Supervisory Control And Data Acquisition (SCADA) server is responsible for monitoring the state of all peripheral units at a given time and broadcasting the data to all connected clients. A SCADA client is independent of the server and can be a mobile application or in our case a web application used for manual control of peripheral units. The User Tracking server is also an implementation of a REST interface responsible for user identification independent of the technology used to identify the user. Both servers communicate with the control unit that is responsible for carrying out orders issued by the servers. Additionally both servers subscribe to events from the control unit. The control unit sends states of requested peripheral units every time a change in their state is made. Besides implementing the REST interface the control unit also runs the algorithm that controls the peripherals to

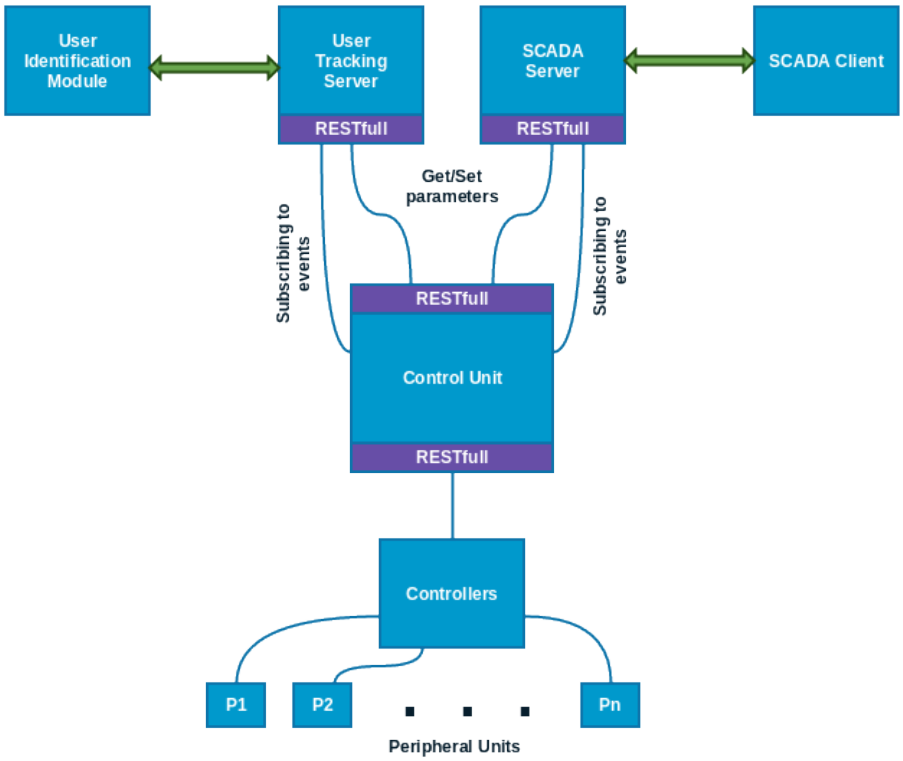


Figure 1

best suite the identified user. The control unit is directly connected to the controllers that are responsible for controlling individual peripherals. The actual peripherals and their controllers are independent of the system. Communication between entities of the system is made by utilizing TCP/IP protocol with Secure Sockets Layer (SSL) encryption. The topology of the system is shown in Figure 1.

3. REST INTERFACES

To make communication between modules easier, we implemented REST services. The first service is receiving requests coming from a mobile device using REST (Representational State Transfer) architecture. After the request is received the service then uses the data within to execute CRUD (create, read, update, delete) commands in a relational database. Using this service users can register, login, and update their location.

The second service is used for communication with peripheral units using REST architecture. The first part of the service is a client that can send requests to peripheral units and read their responses. Using these requests we can get or set variables in peripheral units. The second part of the service listens for requests sent from peripheral units. These requests are events, which happened when a variable in one of the peripheral units has changed.

4. SCADA SERVER

For controlling house devices we set up a SCADA as a web server, which enables control of the peripheral units trough the controller module. We implemented a SCADA client as a web application that enables users to intuitively control the server and consequently respective peripherals [4], [5], [8]. User has an option to select a floor depending on which, a map and elements are shown. One of the main task of a SCADA system is to control electrical devices such as lights, window blinds, temperature, humidity, televisions, wall colors, windows and mosquito nets, consequently permitting full control of the house. User is informed of the position and ongoing states of periphery devices. When changing states of a periphery device, a message containing new state values is sent to target device. If a change occurs on the device, the event is also updated on SCADA system.

To make user interface user friendly and easy to use we designed all elements in graphics. Lights are represented as bulbs and have only two states. To turn the light on or off one has to click on the bulb. For Dali lights, which can control brightness levels, a slider was created for setting the opacity from 0 to 100. Temperature is designed as a simple thermometer with draggable indicator line overlay. By dragging the line on the thermometer value between 0-100 is chosen. For blinds we enable setting their position with draggable line and the state (open or close) by mouse click. Humidity is shown as a drop and its state is controlled with the help of a slider. Television can be turned on or off by clicking on its icon. Wall colors are designed with many color bulbs, which brightness can be controlled through slider and color state with color panel launched on mouse click. Windows blinds and mosquito nets are represented with appropriate icons. Upon clicking the icons, their state is switched from open to close or vise versa. Through SCADA one can also set holiday mode that puts the system into a power sav-

ing and security mode. When these modes are in effect the system minimizes power consumption of periphery devices, locks doors, closes window blinds and adjust the timetable of autonomous task such as vacuuming.

5. USER IDENTIFICATION AND LOCATION MODULE

The purpose of this module is locating and identifying the user in a collection of intelligent objects i.e rooms in a smart house. In smart identifying and locating users can be challenging. Due to the signal attenuation caused by construction materials, the satellite base Global Positioning System (GPS) loses significant power indoors affecting the required coverage for receivers by at lest four satellites. We implemented a smart phone application that will monitor movement indoors. Nowadays almost every person has a smart phone, which makes smart phones a good target platform. The application is developed for Android operating system. The application runs as a daemon, detecting Wi-Fi signals and their strength. The application is able to detect differences between obtained signal strengths in attempt to determine the position of the smart phone using triangulation. In urban areas the application works as expected, whereas inaccuracy is expected in areas with poor access point coverage.

The identification is done with the help of a login screen as can be seen in Figure 2.

User logs into the application that triangulates the smart phone position and determines in which room the user currently is. Considering the battery consumption, measurements are done only when moving. When the application detects the change in the room location it sends the information to the server trough a REST interface. This module allows us to track the user inside the building. Identifying and locating users inside an intelligent object is key. Based upon the user data and his or her location the system can control periphery units accordingly.

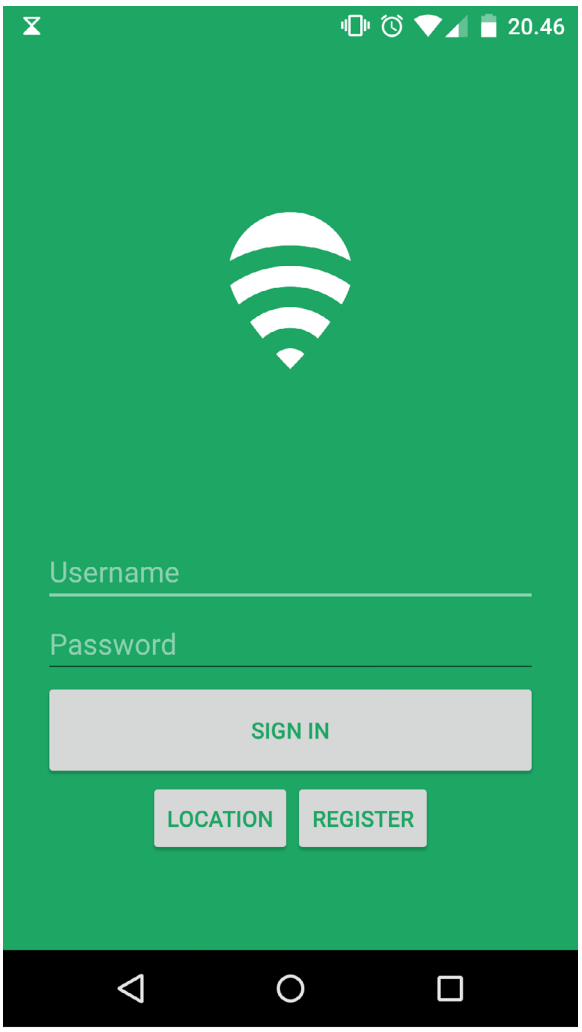


Figure 2

6. METHODOLOGY

One of the main goals of this project is to design an adaptive algorithm that will adjust the parameters of external units, such as room temperature, room brightness etc. The algorithm will adapt the output parameters with respect to users wishes and changes the user has made in the past. An important feature and challenge of the algorithm was the ability to adjust the output parameters in cases where multiple users have different requirements. In such cases we cannot expect the algorithm to fulfill requirements of all users, instead the algorithm finds the best parameter values that will hopefully suit all users to some extent.
We use controlling temperature as an example and assume it's change to be continuous. At the initialization we assume the desired temperature of the user is known. The temperature function will be normalized so that the integral from minus infinity to infinity equals 1 and the limits to infinity and minus infinity equal 0. Suppose we now have multiple users with each of them having its own desired temperature. Consequently we now have multiple such functions, and our goal is to predict a temperature, such that in all the other values the function has lesser value then the search value. This point will consequently be the best possible state for all the users because any other value, would have been worse for at least one of the users. It is important to note that the continues function where the limit to infinity equals 0 is bounded in the set of real numbers. This means that the function has a maximum value.

We describe the steps of the algorithm in more detail below:

1. We pick a large enough value so that it is bigger then all the maxima of all the functions. Because all the functions are positive we pick a minimum value of 0.
2. With the two picked values we perform bisection as follows: We take the middle of the two values and for every function we find such intervals in which the function is greater than the given value. We then find the intersection of all the intervals. If the intersection is empty, the new middle value becomes the new maximum value. We repeat step 2.
3. The algorithm stops when the intersection of all in-

tervals has a finite amount of points (optimally only one). All such points are best approximations of users preferred values.

At first we assume that our functions derived from user preferences are normally distributed. Each user will have two additional values:

1. The middle value of the function will be the probability of output value most suited for the user.
2. Standard deviation of the given function will tell us how much the user is willing to adapt.

If we implement our algorithm on a family of such functions then each function will only have one interval where the values of the function are grater then the given value. In such a case the finite intersection of the intervals will also be an interval. Because the intersection can be ether empty of an interval the algorithm will stop when we only have one point in the intersection. This point will represent the "sweet spot of balance" for all users.

6.1 Algorithm

Define:

$$f(x) = min(d_1(x), d_2(x), \ldots, d_k(x))$$

where $d_i(x)$ is distribution function, which represents adaptability to particular value of *ith* user. It has a maximum at the user's desired value.
Our goal is to find x such that f is maximized i.e. we want to find value

$$max_{x\in[-\infty,+\infty]} f(x)$$

.
We will also define the following set of intervals: $[a_{d_i,q}, b_{d_i,q}] = \{x \in (a,b) \text{ s.t. } d_i(x) > q\}$.Each of them represents an interval where function $d_i(x)$ is greater then some q. One can prove that this will result into a single interval and not the union of disconnected intervals.
Also, we define the sequence of intervals recursively as:

$$[x_0, y_0] = [0, 100]$$

$$[x_n, y_n] = \begin{cases} [z_{n-1}, y_{n-1}] & \text{if } \cap_{i=1}^{k} [a_{d_i,z_{n-1}}, b_{d_i,z_{n-1}}] \neq \emptyset \\ [x_{n-1}, z_{n-1}] & \text{otherwise} \end{cases}$$

where $z_{n-1} = \frac{x_{n-1}+y_{n-1}}{2}$
In the limit:

$$\lim_{n\to\infty} [x_n, y_n]$$

we will get the desired point.

7. CONCLUSION

Home automation has proven it will grow into a new market in the future. There is a growing amount of suppliers of custom periphery unity that can be controlled by users remotely. Some solutions even implement standards multiple standards. The current state standardization is still far from ideal. Popular suppliers like Z-wave, Zigbee, Insteon tend to implement their own closed-source communication protocol instead of adopting existing protocols. Many times the systems are not compatible and upgrades are regularly needed as the protocols and devices change. Besides the lack of standardization the current solutions that implement self control do not adapt to users automatically. Most systems are require the user to manually configure the system, which cuts out a huge market of non tech savvy people. We see personalization of intelligent systems to be a future market especially solving growing problems such as the rapidly growing amount of old population in need of care. Personalizing objects can also be the future for businesses offering users services or public buildings.

8. ACKNOWLEDGMENT

Part of the project is financed by the European Union, specifically from the European social fund. The project is carried out under the Operational program for the development of human resources for the period 2007-2013, 1st development priority "Business promotion, adaptation, and priority policies" 1.3. "Fellowship scheme", within the project "On a creative path to practical knowledge".

9. REFERENCES

[1] A. Brandt, K. Samuelsson, O. Töytäri, and A.-L. Salminen. Activity and participation, quality of life and user satisfaction outcomes of environmental control systems and smart home technology: a systematic review. *Disability and rehabilitation. Assistive technology*, 6(3):189–206, 2011.

[2] G. Demiris, M. Rantz, M. Aud, K. Marek, H. Tyrer, M. Skubic, and A. Hussam. Older adults' attitudes towards and perceptions of "smart home" technologies: a pilot study. *Medical informatics and the Internet in medicine*, 29(2):87–94, 2004.

[3] S. Martin, G. Kelly, W. G. Kernohan, B. McCreight, and C. Nugent. Smart home technologies for health and social care support. *Cochrane database of systematic reviews*, pages 1–11, 2008.

[4] B. Qiu and H. B. Gooi. Web-based SCADA display systems (WSDS) for access via Internet. *IEEE Transactions on Power Systems*, 15(2):681–686, 2000.

[5] B. Qiu, H. B. Gooi, Y. Liu, and E. K. Chan. Internet-based SCADA display system. *IEEE Computer Applications in Power*, 15(1):14–19, 2002.

[6] V. Rialle, C. Ollivet, C. Guigui, and C. Hervé. What Do family caregivers of Alzheimer's disease patients desire in smart home technologies? Contrasted results of a wide survey. *Methods of Information in Medicine*, 47(1):63–69, 2008.

[7] Tošić Aleksandar, Radež Nejc, Lukunič Primož, Cuder Sašo, Krivec Tadej, Štular Tanja, Jovičić Vladan, and Brodnik Andrej. PAHIMA pametna hiša malina. *International Electrotechnical and Computer Science Conference*, 23, 2014.

[8] G. Zecevic. Web based interface to SCADA system. *POWERCON '98. 1998 International Conference on Power System Technology. Proceedings (Cat. No.98EX151)*, 2, 1998.

StuCoSReC

University of Primorska Press
www.hippocampus.si
ISBN *978-961-6984-01-0*
Not for resale