Elektrotehniški vestnik 76(1-2): 19–24, 2009
Electrotechnical Review, Ljubljana, Slovenija

Optimalno permutacijsko usmerjanje v heksagonalnih omrežjih

Maja Rotovnik[1], **Jurij Šilc**[2], **Janez Žerovnik**[3,1]

[1] *Inštitut za matematiko, fiziko in mehaniko, Jadranska ulica 19, SI-1000 Ljubljana, Slovenija*
[2] *Institut "Jožef Stefan", Odsek za računalniške sisteme, Jamova cesta 39, SI-1000 Ljubljana, Slovenija*
[3] *Univerza v Ljubljani, Fakulteta za strojništvo, Aškerčeva cesta 6, SI-1000 Ljubljana, Slovenija*
E-pošta: mrotovnik@yahoo.com, jurij.silc@ijs.si, janez.zerovnik@imfm.uni-lj.si

Povzetek. V sestavku najprej vpeljemo komunikacijska omrežja in usmerjanje podatkov v njih, v nadaljevanju pa se osredotočimo na permutacijsko usmerjanje. Osrednji del sestavka je problem optimalnega permutacijskega usmerjanja na trikotniških mrežah. Predstavljen je optimalni permutacijski usmerjevalni algoritem, ki za usmerjanje vseh permutacij potrebuje l_{max} korakov, kjer je l_{max} najdaljša izmed vseh najkrajših poti sporočil.

Ključne besede: teorija grafov, heksagonalno omrežje, permutacijsko usmerjanje, trikotniške mreže

Optimal Permutation Routing on Hexagonal Networks

Extended abstract. At the beginning of the paper we introduce communication networks and data routing in networks. Later we focus on permutation routing, where each base station is the origin of at most one package and at the same time is the destination of no more than one package. The main part of the paper represents the problem of optimal permutation routing in triangular meshes with full-duplex edges.
We describe an optimal permutation routing algorithm for full-duplex triangular meshes. The basic idea of the algorithm is that a saturated package should not wait any longer because it has already waited as long as it could, otherwise the algorithm becomes suboptimal. Packet p is saturated if the number of waiting steps of the packet is $l_{max} - l_p$, where l_{max} is the maximum length over the shortest paths of all packets and l_p is the length of the shortest path of packet p.
The algorithm routes every permutation in the l_{max} routing steps and is optimal, because l_{max} is a lower bound of every permutation routing algorithm in triangular meshes.

Key words: graph theory, hexagonal network, permutation routing, triangular mesh

1 Uvod

Teorija grafov je ena najbolj proučevanih in uporabnih matematičnih smeri v zadnjih nekaj desetletjih. Aplikacije, ki jih lahko predstavimo z grafi, najdemo na več različnih področjih, kot so operacijske raziskave, družboslovje, kemija, računalniške mreže in predvsem komunikacijska omrežja (KO). KO je sestavljeno iz baznih postaj (BP) in povezav, po katerih lahko pošiljamo med njimi sporočila. KO lahko predstavimo z grafom, kjer so BP vozlišča grafa, fizične povezave med BP pa so povezave grafa. Osnovna naloga KO je omogočanje izmenjave podatkov med BP. Način izbire poti, po katerih se bodo podatki premikali, imenujemo *usmerjanje* in je bistvenega pomena za uspešno delovanje KO, saj lahko slaba izbira vodi do neučinkovitega in počasnega omrežja. Če usmerjanje povezuje vse (urejene) pare vozlišč, ga imenujemo *popolno usmerjanje*, sicer ga imenujemo *delno usmerjanje*. Literature na tu obravnavano temo je veliko, tu omenimo samo pogosto citirano knjigo [3] in zapis v slovenskem jeziku [7].

Uspešno usmerjanje je odvisno od izbire topologije KO, tipa povezav, usmerjevalnega problema itd. [1]. Topologija KO je način, kako predstavimo KO s pomočjo grafov in kako so BP povezane v določene tipe grafov. Topologija je torej fizična povezanost KO. Od njene izbire so odvisne številne lastnosti, kot so premer, stopnja vozlišč, razširljivost, odpornost na napake itd. Od dobrega KO pričakujemo, da bo imel majhen premer pri nizki stopnji vozlišč, da bo dobro razširljiv in bo čim bolj odporen na napake. Idealnega KO, ki bi imel vse te lastnosti, ni, saj z zmanjševanjem stopnje vozlišč večamo premer. Topologijo izberemo torej glede na pomembnost teh lastnosti v KO. Najbolj znane topologije so mreže. Povezave med vozlišči KO so lahko neusmerjene, usmerjene ali hkratno dvosmerne. Glede na vrsto povezav, ki so v posameznem KO, ločimo različne modele usmerjanja: *neusmerjen model*, ki omogoča pretok podatkov po povezavi v obe smeri, vendar le v eno smer hkrati, *usmerjen model*, ki omogoča pretok podatkov po povezavi le v eno, in to vedno isto smer, ter *hkratni dvosmerni model*, ki omogoča prenos podatkov v obe smeri, vendar le en podatek v eno smer hkrati. Usmerjevalni model KO

Prejet 6. avgust, 2008
Odobren 20. oktober, 2008

določa način izmenjave sporočil. Poznamo več tipov usmerjevalnih modelov, kot npr. *preklapljanje povezav*, *usmerjanje z vodili* in *paketno usmerjanje*. Najpogosteje je uporabljeno paketno usmerjanje, kjer se sporočilo, ki ga želimo poslati, preoblikuje v enega ali več paketov. Vsak paket pozna svoj cilj, ki je enak cilju prvotnega sporočila. Ko paketi prispejo na cilj, se ponovno združijo v prvotno sporočilo. Usmerjanje poteka v več zaporednih korakih, ki jih imenujemo *usmerjevalni koraki*. V vsakem koraku se lahko paket premakne iz vozlišča, v katerem je, v eno izmed sosednjih vozlišč.

2 Splošni usmerjevalni problem

Usmerjevalni problem, kjer je število paketov določeno pred začetkom usmerjanja in imajo vsi paketi že pred začetkom usmerjanja določene cilje, imenujemo *splošni usmerjevalni problem* ali (N, p, k_1, k_2)*-usmerjevalni problem*, kjer je N število paketov, ki so na začetku usmerjanja v p vozliščih in pred začetkom usmerjanja, ni v nobenem vozlišču več kot k_1 in na koncu več kot k_2 paketov. Splošni usmerjevalni problem zajema več problemov, ki jih delimo v tri razrede.

Problemi več vozlišč – isto sporočilo so usmerjevalni problemi, pri katerih več vozlišč oddaja ali sprejema isto sporočilo, torej gre za usmerjevalne probleme *eden-več* in *več-eden*.

Izotonični problemi so usmerjevalni problemi, pri katerih se razdalja med celoštevilskimi reprezentacijami izvorov in ciljev paketov bodisi ohranja bodisi spreminja z natančno določenimi metričnimi pravili.

Permutacijski problemi so usmerjevalni problemi, pri katerih je podana permutacija Π množice vozlišč omrežja. V začetku usmerjanja je v vsakem vozlišču v omrežja natanko en paket, ki je namenjen v vozlišče $\Pi(v)$. Gre torej za $(n, n, 1, 1)$ - usmerjevalni problem na omrežju z n vozlišči. *Delni permutacijski usmerjevalni problem* se od permutacijskega razlikuje po tem, da je v omrežju manj kot n paketov, preslikava Π pa je injektivna. V skrajnem primeru, ko je v omrežju le en paket, govorimo o *osnovnem usmerjevalnem problemu*. Preslikava Π je v tem primeru transpozicija.

Naloga usmerjevalnega algoritma je rešitev usmerjevalnega problema. Algoritem za vsak podatek v omrežju določi pot, po kateri bo ta potoval po omrežju od začetka do cilja. Pri tem mora upoštevati vse omejitve, ki jih narekujeta topologija omrežja in usmerjevalni problem. Preprečiti mora *trk*, to je situacijo, v kateri želi več sporočil hkrati dostopati do iste povezave.

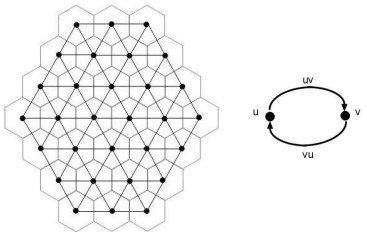
Usmerjevalne algoritme delimo na *statične* in *dinamične*. Statični algoritmi določijo parametre potovanja paketov že pred začetkom usmerjanja, medtem ko dinamični algoritmi pot paketov določajo sproti. Ker je rešitev danega usmerjevalnega problema lahko več, mora usmerjevalni algoritem izbrati najboljšo rešitev glede na *kriterij kakovosti*.

Kakovost usmerjevalnih algoritmov lahko ocenjujemo z več vidikov, odvisno od namena, načina uporabe in razpoložljivih virov. Pogost način ocenjevanja je glede na prostorsko zahtevnost. Včasih nas zanima tudi, ali je algoritem vodil pakete po najkrajših poteh in ali je vse pakete sploh dostavil. Največkrat pa algoritme ocenjujemo po najdragocenejšem viru – času. Časovno zahtevnost usmerjevalnega algoritma merimo v številu usmerjevalnih korakov, ki jih algoritem potrebuje, da dostavi vse pakete do cilja.

3 Optimalno permutacijsko usmerjanje na trikotniških mrežah

Ravninske mreže spadajo med najbolj proučevane topologije KO. Znano je, da obstajajo le tri porazdelitve ravnine v pravilne večkotnike: trikotniška porazdelitev ravnine, kvadratna porazdelitev ravnine ter porazdelitev ravnine na šestkotnike, iz katerih dobimo trikotniške, kvadratne in heksagonalne mreže.

Porazdelitev ravnine na šestkotnike je zelo primerna pri uporabi brezžičnih in mobilnih omrežij, saj imajo celice optimalni premer glede na območno razmerje. Če centre sosednjih celic med sabo povežemo, dobimo trikotniško mrežo. Torej so heksagonalna omrežja končni podgrafi trikotniške mreže (slika 1, levo).



Slika 1. Heksagonalno omrežje in trikotniška mreža (levo); dvosmerna povezava (desno)
Figure 1. Hexagonal network and triangular mesh (left); a full--duplex edge (right)

V nadaljevanju bomo obravnavali optimalni permutacijski usmerjevalni algoritem za primer, ko je vsaka povezava grafa hkrati dvosmerna (slika 1, desno). Kadar to ne velja, lahko iz algoritma za dvosmerne povezave konstruiramo 2-aproksimacijo s pomočjo sodih in lihih korakov, kot je razloženo npr. v [2].

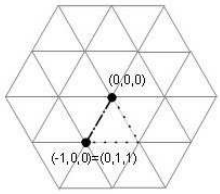
3.1 Problem usmerjanja

V neskončni trikotniški mreži imamo podano končno podmnožico vozlišč V in permutacijo $\Pi : V \to V$. Vsako vozlišče v iz množice V želi poslati sporočilo vozlišču $\Pi(v)$ iz V, torej imamo $|V|$ sporočil, ki jih želimo dostaviti.

Po vsaki povezavi grafa lahko potuje v eno smer hkrati le en paket, ki v vsakem koraku algoritma bodisi počaka v vozlišču bodisi se pomakne v sosednje vozlišče. V istem vozlišču je dovoljeno čakanje več paketov hkrati, zato potrebujemo čakalne vrste. Ker je največja izhodna stopnja vozlišč grafa šest, potrebujemo v vsakem vozlišču šest vrst.

Naš cilj je minimizirati število korakov, ki jih potrebujemo, da vsi paketi prispejo do svojih ciljev.

Vsakemu vozlišču mreže določimo koordinatni naslov na način, ki so ga vpeljali v [6] (slika 2). Koordinatni sistem ima tri osi x, y, z s kotom med njimi 120. Naj bodo i, j, k enotski vektorji na teh treh oseh. Ti vektorji so linearno odvisni, saj med njimi velja zveza $i + j + k = 0$. Vsako vozlišče grafa lahko zapišemo kot linearno kombinacijo teh vektorjev, žal pa zapis ni enoličen.



Slika 2. Koordinatni sistem Figure 2. Coordinating system

Definicija 3.1 *Naj bo G trikotniška mreža. Poljubno vozlišče grafa določimo za izhodišče in mu dodelimo koordinate $(0, 0, 0)$. Za vsako drugo vozlišče A v grafu G obstaja več poti od izhodišča do vozlišča A, različnih dolžin, ki so sestavljene iz a enot vektorja i, b enot vektorja j in c enot vektorja k, za neka cela števila a, b, c. Zato lahko vse te poti zapišemo kot $(a, b, c) = ai + bj + ck$ in tak zapis vsake izmed teh poti imenujemo koordinatni naslov vozlišča A.*

Ker med izhodiščem in vozliščem A obstaja več poti, obstaja tudi več koordinatnih naslovov vozlišča A. Naj bo tudi (a', b', c') koordinatni naslov vozlišča A. Potem velja $(a, b, c) = (a', b', c') \Leftrightarrow ai + bj + ck = a'i + b'j + c'k$. Če velja $(a, b, c) = (a', b', c')$, potem obstaja tako celo število d, da velja $a' = a + d,\ b' = b + d,\ c' = c + d$. Ob združitvi teh dveh dejstev vidimo, da če je (a, b, c) koordinatni naslov vozlišča A, potem so vsi koordinatni naslovi vozlišča A oblike $(a + d, b + d, c + d)$ za vsako celo število d.

Definicija 3.2 *Koordinatni naslov vozlišča A je oblike najkrajše poti, če obstaja pot od izhodišča do vozlišča A, ki je sestavljena iz a enot vektorja i, b enot vektorja j, c enot vektorja k in ima med vsemi najkrajšo dolžino.*

Med dvema vozliščema je lahko več najkrajših poti, vendar se izkaže, da imajo vse enako vektorsko reprezentacijo.

Izrek 3.3 [6] *Koordinatni naslov vozlišča $A = (a, b, c)$ je oblike najkrajše poti natanko tedaj, ko velja:*

1. *Vsaj ena komponenta je ničelna ($abc = 0$).*
2. *Komponente so paroma različno predznačene ($ab \leq 0,\ ac \leq 0,\ bc \leq 0$).*

Dokaz. Dokaz s protislovjem.
($\Rightarrow$) Najprej predpostavimo, da ne velja prva točka, torej $abc \neq 0$. Potem sta vsaj dve izmed števil a,b,c enako predznačeni. Brez izgube za splošnost lahko predpostavimo $a > 0, b > 0$ (druge možnosti obravnavamo analogno). Ker velja $i + j + k = 0$, je $ai + bj + ck = ai+bj+ck-(i+j+k) = (a-1)i+(b-1)j+(c-1)k$. Torej je $(a - 1, b - 1, c - 1)$ tudi koordinatni naslov vozlišča A. Dolžina poti, ki ustreza koordinatnemu naslovu (a, b, c), je enaka $|a| + |b| + |c|$, dolžina poti, ki ustreza koordinatnemu naslovu $(a - 1, b - 1, c - 1)$, pa je enaka $|a - 1| + |b - 1| + |c - 1|$. Ker velja $a > 0$ in $b > 0$, je $|a-1|+|b-1|+|c-1| \leq |a-1|+|b-1|+|c|+1 = |a|-1+|b|-1+|c|+1 < |a|+|b|+|c|$. Torej je dolžina poti, ki ustreza koordinatam $(a - 1, b - 1, c - 1)$, krajša od najkrajše poti (a, b, c). Prišli smo v protislovje s tem, da je (a, b, c) oblike najkrajše poti. Torej je $abc = 0$ in je vsaj ena komponenta ničelna.

Zdaj predpostavimo, da ne velja druga točka, torej da komponente niso paroma različno predznačene. Brez izgube za splošnost lahko predpostavimo $ab > 0$ (druge možnosti obravnavamo analogno). Potem po prvi točki velja, da je $c = 0$. Iz tega sledi, da je bodisi $a > 0$ in $b > 0$, bodisi $a < 0$ in $b < 0$. Potem je $ai+bj+ck = ai+bj = ai + bj - (i + k + j) = (a - 1)i + (b - 1)j - k$. Dolžina poti, ki ustreza $(a-1, b-1, -1)$, je $|a-1|+|b-1|+|-1|$. Ker je $|a|+|b|+|c| = |a|+|b| = |a-1|+1+|b-1|+1 > |a-1|+|b-1|+|-1|$, dobimo pot, krajšo od najkrajše. Protislovje. Zato velja $ab \leq 0,\ ac \leq 0,\ bc \leq 0$.
($\Leftarrow$) Zdaj moramo pokazati, da je, če veljata pogoja, koordinatni naslov (a, b, c) oblike najkrajše poti. Brez izgube za splošnost lahko predpostavimo $c = 0,\ a \geq 0,\ b \leq 0$ (vse druge možnosti obravnavamo analogno). Vsi mogoči koordinatni naslovi za vozlišče A so oblike $(a + d, b + d, c + d)$, za vsako celo število d. Zdaj ločimo dve možnosti:

1. $d > 0$: $|a + d| + |b + d| + |c + d| = |a| + |d| + |b + d|+|d| \geq |a|+|d|+|b|-|d|+|d| = |a|+|b|+|d| > |a| + |b|$
2. $d < 0$: $|a + d| + |b + d| + |c + d| = |a + d| + |b| + |d|+|d| \geq |a|-|d|+|b|+|d|+|d| = |a|+|b|+|d| > |a| + |b|$

Vidimo, da je (a, b, c) res oblike najkrajše poti. □

Posledica 3.4 [6] *Koordinatni naslov, podan v obliki najkrajše poti, je enoličen.*

Dokaz. Če je koordinatni naslov vozlišča $A = (a, b, 0)$ podan v obliki najkrajše poti, potem so vsi koordinatni naslovi za A oblike $(a + d, b + d, d)$ za vsako celo število d. Če je $d \neq 0$, je $|a + d| + |b + d| + |d| > |a| + |b|$, torej mora biti $d = 0$ in je zapis res enoličen. □

Na začetku permutacijskega usmerjanja vsak paket pozna svoje izvorno in ciljno vozlišče. Naj bo S izvorno vozlišče paketa in D njegovo ciljno vozlišče. Izračunamo lahko relativni naslov $D - S$, ki je dolžina najkrajše poti med S in D.

Posledica 3.5 [6] *Če je $D-S = (a, b, c)$ oblike najkrajše poti je dolžina najkrajše poti med S in D enaka $|a|+|b|+|c|$.*

Izrek 3.6 [6] *Če je $D - S = ai + bj + ck$, potem je $|D - S| = \min\{|a-c|+|b-c|,\ |a-b|+|b-c|,\ |a-b|+|a-c|\}$.*

Dokaz. Ker so a, b, c cela števila, je eno izmed njih vedno med drugima dvema, torej je eno število mediana vseh treh.

1. c je mediana števil a, b, c. Potem je $(a, b, c) = (a - c, b - c, 0)$, ki je oblike najkrajše poti, zato je $|D - S| = |a - c| + |b - c|$.
2. a je mediana števil a, b, c. Potem je $(a, b, c) = (0, a - b, a - c)$, ki je oblike najkrajše poti, zato je $|D - S| = |a - b| + |a - c|$.
3. b je mediana števil a, b, c. Potem je $(a, b, c) = (a - b, 0, b - c)$, ki je oblike najkrajše poti, zato je $|D - S| = |a - b| + |b - c|$.

Če vse troje združimo, res dobimo $|D - S| = \min\{|a - c| + |b - c|,\ |a - b| + |b - c|,\ |a - b| + |a - c|\}$. □

3.2 Optimalni permutacijski usmerjevalni algoritem

Definicija 3.7 *Naj bo p paket, podan s koordinatnim naslovom oblike najkrajše poti. Naj bo l_p dolžina najkrajše poti paketa p, l_{max} pa naj bo najdaljša med najkrajšimi potmi vseh paketov. Potem je $l_p = |a| + |b| + |c|$ in $l_{max} = \max_p(l_p)$.*

Lema 3.8 *Število korakov kateregakoli permutacijskega usmerjevalnega algoritma je najmanj l_{max}.*

Dokaz. l_{max} je najdaljša izmed najkrajših poti. Ker se paket v enem koraku lahko premakne kvečjemu za eno enoto, potrebujemo vsaj l_{max} korakov. □

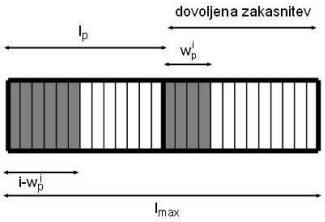
Definicija 3.9 *Pravimo, da paketa p in p' trkneta, če sta hkrati v isti izhodni vrsti istega vozlišča. Če hkrati trkne več paketov, se kvečjemu eden izmed njih pomakne naprej.*

Definicija 3.10 *Številu korakov, ki jih paket p čaka do i-tega koraka permutacijskega usmerjevalnega algoritma, pravimo čas čakanja paketa p ali krajše, zakasnitev paketa p. Označimo jo s w_p^i.*

Zakasnitev w_p^i je dovoljena, če je $l_p + w_p^i \leq l_{max}$. Sicer je zakasnitev prepovedana. Paket p je nasičen na koncu i-tega koraka, če je $w_p^i = l_{max} - l_p$.

Ideja optimalnega permutacijskega usmerjevalnega algoritma, ki ga bomo predstavili, je, da nasičen paket ne sme več čakati, sicer algoritem ne bo več optimalen.

Na sliki 3 je prikazan mogoči paketni model. Vsak paket v tem paketnem modelu ima dve škatli, eno dolžine l_p, drugo pa dolžine $l_{max} - l_p$.



Slika 3. Model paketa Fig. 3: Model of a package

V vsakem koraku algoritma ima paket dve možnosti. Če se premakne v sosednje vozlišče, se zapolni pravokotnik v škatli dolžine l_p, sicer se zapolni pravokotnik v škatli dolžine $l_{max} - l_p$. Če je polna prva škatla, je paket na cilju. Če se zapolni druga škatla, je paket nasičen in ne sme več čakati. Model, ki je prikazan na sliki 3, rabi zgolj za analitične namene. V resnici paket ne potrebuje vseh informacij, na vsakem koraku potrebuje le vrednosti l_p in w_p^i.

Predpostavimo še, da ima omrežje globalno uro in da so vsa vozlišča sinhronizirana. Paketi se premikajo le ob diskretnih urnih ciklih, druge naloge so narejene v vmesnem času.

V nadaljevanju bomo podrobneje predstavili optimalni permutacijski algoritem $\mathfrak{A}$ in podrobneje opisali pomembnejše postopke. Predpostavimo lahko, da je število vozlišč grafa končno, torej neko naravno število n. Čea je vozlišč neskončno, dobimo neskončno zanko.

Algoritem $\mathfrak{A}$
V vsakem vozlišču u omrežja naredi:
begin
  PREDPROCESIRANJE;
  $i = 0$;
  while $i < l_{max}$ **do**
    //* *Faza sprejemanja:*
    za vsak paket v vozlišču u
      POSODOBI_PAKET;
    //* *Faza pošiljanja:*
    za vsak paket v vozlišču u
      DOLOČI_IZHODNO_POVEZAVO;
    za vsako vrsto UREDI_VRSTO;
    pošlji prvi paket;
    $i = i + 1$

endwhile
end

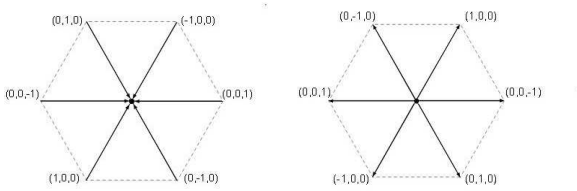
3.2.1 Postopek PREDPROCESIRANJE

Pred začetkom usmerjanja je v vsakem vozlišču en paket, ki pozna svoje ciljno vozlišče D. Zato lahko vsako izvorno vozlišče S izračuna koordinatni naslov paketa $D - S = (a, b, c)$. Po izreku 3.6 je dovolj preveriti, kateri izmed koordinatnih naslovov $(0, b - a, c - a)$, $(a - b, 0, c - b)$, $(a - c, b - c, 0)$ ima neničelni koordinati različno predznačeni. Nato izračunamo dolžino najkrajše poti med S in D, ki je $l_p = |a| + |b| + |c|$. Vsakemu paketu se doda še glava, ki nosi informacije o $D - S$, l_p in števcu w_p^i.

Časovna zahtevnost predprocesiranja je $O(1)$, če je treba določiti fiksno število naslovov, sicer pa $O(\log n)$, kjer je n največja velikost števila, ki je potrebno za določitev naslovov vozlišč.

3.2.2 Postopek POSODOBI_PAKET

Posodobitev koordinatnega naslova na vsakem koraku algoritma pripomore k večji robustnosti. Preveriti moramo, ali je paket že v ciljnem vozlišču ali ne. Vsako vozlišče ima šest vhodnih povezav, ki jih lahko brez izgube za splošnost označimo, kot je prikazano na sliki 4 levo.



Slika 4. Vhodne povezave (levo); izhodne povezave (desno)
Figure 4. Input edges (left); output edges (right)

Postopek POSODOBI_PAKET izračuna razliko med starim naslovom in vhodno povezavo in to razliko priredi kot novi naslov. Če je razlika enaka $(0, 0, 0)$, je paket že v ciljnem vozlišču, sicer še ni.

3.2.3 Procedura DOLOČI_IZHODNO_POVEZAVO

S pomočjo procedure DOLOČI_IZHODNO_POVEZAVO se odločimo, po kateri se bo paket premaknil. Vsako vozlišče ima šest izhodnih povezav, ki jih lahko brez izgube za splošnost označimo, kot je prikazano na sliki 4 desno.

Vseh $|V|$ koordinatnih naslovov lahko razdelimo v 12 disjunktnih razredov glede na predznake koordinat: $(+, 0, 0)$, $(-, 0, 0)$, $(0, +, 0)$, $(0, -, 0)$, $(0, 0, +)$, $(0, 0, -)$, $(+, -, 0)$, $(+, 0, -)$, $(-, +, 0)$, $(-, 0, +)$, $(0, +, -)$ in $(0, -, +)$.

Postopek DOLOČI_IZHODNO_POVEZAVO določi, v katero smer se bo paket premaknil v naslednjem koraku:

begin
  if koordinatni naslov ima le eno neničelno komponento **then**
    izhodna_povezava =
      povezava, ki ustreza neničelni komponenti;
  else
    izhodna_povezava =
      povezava, ki ustreza negativni komponenti;
  endif
end

3.2.4 Postopek UREDI_VRSTO

Vsako vrsto uredimo po padajočem številu preostalih korakov. Torej paket, ki ima še največ preostalih korakov, ima prioriteto 1, naslednji ima prioriteto 2 itd. Ta ureditev je optimalna, kot bomo pokazali v nadaljevanju.

Zaradi optimalnosti je pomembno, kako uredimo vrsto. Druga možnost bi bila, da pakete razvrstimo po dolžini njihove poti l_p in potem v primeru enakosti po številu preostalih korakov. Vendar druga možnost vodi do neoptimalnega algoritma. (Primer je dan v [5], glej tudi [6].)

Lema 3.11 [5] *Paket lahko čaka le pred zadnjo smerjo.*

Dokaz. Denimo, da dva paketa trkneta, ko imata še dve neničelni komponenti. Po postopku DOLOČI_IZHODNO_POVEZAVO mora biti smer, v kateri potujeta paketa ista, zato bi morala imeti isto izvorno vozlišče. Protislovje s tem, da ima vsak paket v permutacijskem usmerjanju različno izvorno vozlišče.

Paketi, ki imajo le eno neničelno komponento, lahko trknejo le pred smerjo, ki ustreza tej komponenti. Torej paketi res čakajo le pred zadnjo smerjo. □

Lema 3.12 [5] *Paketa v dani vrsti ne moreta imeti istega števila preostalih korakov.*

Dokaz. Če sta paketa p in p' v isti vrsti, morata biti pred svojo zadnjo smerjo. Če bi imela isto število preostalih korakov, bi imela isto ciljno vozlišče. Protislovje s tem, da ima vsak paket v permutacijskem usmerjanju različno ciljno vozlišče. □

3.3 Dokaz optimalnosti algoritma

Lema 3.13 [5] *Naslednji razvrstitvi paketov sta ekvivalentni.*

1. *Razvrstitev paketov v vrsto po padajočem številu preostalih korakov*
2. *Razvrstitev paketov v vrsto po naraščajočem številu dovoljene zakasnitve*

Dokaz. Število preostalih korakov paketa p po i-tem koraku algoritma je $l_p - (i - w_p^i) = l_p - i + w_p^i$. Preostalo število dovoljene zakasnitve paketa p po i-tem koraku algoritma pa je $l_{max} - l_p - w_p^i$. Števili se poleg predznaka razlikujeta le v konstantah i in l_{max}, torej je razvrstitev res ekvivalentna. □

Trditev 3.14 [5] *Algoritem $\mathfrak{A}$ ne povzroči dodatne zakasnitve paketov.*

Dokaz. Z indukcijo po številu korakov bomo pokazali, da ni dodatne zakasnitve po i-tih korakih.

Baza indukcije, $i = 1$. Po prvem koraku so nasičeni le paketi z dolžino l_{max}. Ker so vsi paketi na začetku usmerjanja v različnih vozliščih, po prvem koraku ne moreta biti dva v isti vrsti istega vozlišča. Zato imajo vsi paketi z dolžino poti l_{max} največjo prioriteto glede na število preostalih korakov in zato noben ne čaka.

Korak indukcije. Zdaj predpostavimo, da po $i - 1$ korakih ni bilo dodatne zakasnitve in dokažimo, da je ni niti po i-tem koraku. Zadošča pokazati, da je v vsaki čakalni vrsti le en nasičen paket. Predpostavimo nasprotno. Denimo, da sta p in p' oba nasičena paketa v isti vrsti. Potem po definiciji nasičenosti velja $l_p + w_p^i = l_{p'} + w_{p'}^i = l_{max}$. Število preostalih korakov paketa p je $l_p - (i - w_p^i) = l_{max} - i$, število preostalih korakov paketa p' pa je $l_{p'} - (i - w_{p'}^i) = l_{max} - i$. Torej imata p in p' pred zadnjo smerjo še isto število korakov in zato isto ciljno vozlišče. Protislovje, saj v permutacijskem usmerjanju dva paketa ne moreta imeti istega ciljnega vozlišča. □

Izrek 3.15 [5] *Algoritem $\mathfrak{A}$ je optimalni permutacijski algoritem za trikotniške mreže z dvosmernimi povezavami.*

Dokaz. Zaradi trditve 3.14 vsi paketi dosežejo cilje v največ l_{max} korakih, torej je dosežena spodnja meja. □

Definicija 3.16 *Usmerjevalni algoritem je pozabljiv, če je pot vsakega paketa p odvisna le od njegovega izvornega in ciljnega vozlišča, čeprav je čas čakanja v vmesnih vozliščih odvisen od drugih poti.*

Definicija 3.17 *Naj bo P_{uv} pot med vozliščema u in v. Pozabljiv algoritem $P_{uv} : u, v \in V$ je translacijsko invarianten, če je $P_{(u+w)(v+w)} = P_{uv+w}$ za $\forall u, v, w \in V$.*

Posledica 3.18 [4] *Algoritem $\mathfrak{A}$ je pozabljiv, translacijsko invarianten in optimalen.*

Dokaz. Pozabljiv je očitno, saj algoritem potrebuje za pot vsakega paketa le izvorno in ciljno vozlišče. Ker je za usmerjanje pomembna le razlika $D - S$ med izvornim vozliščem S in ciljnim vozliščem D, je algoritem translacijsko invarianten. □

4 Sklep

Obravnavali smo problem optimalnega permutacijskega usmerjanja in optimalne permutacijske usmerjevalne algoritme na trikotnih mrežah. Algoritem za trikotne mreže usmeri vsako permutacijo v l_{max} korakih, kjer l_{max} označuje najdaljšo med vsemi najkrajšimi potmi paketov.

5 Literatura

[1] T. Dobravec, Usmerjevalni algoritmi v omrežjih s topologijo krožnih grafov, doktorska dizertacija, Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, 2004.

[2] T. Dobravec, J. Žerovnik, B. Robič, Permutation Routing in Double-Loop Networks: Design and Empirical Evaluation, *Journal of Systems Architecture*, 48(13-14):387–402, 2003.

[3] T. Leighton, Introduction to Parallel Algorithms and Architectures: Arrays-Trees-Hypercubes, Morgan-Kaufman, San Mateo, California, 1992.

[4] M. Rotovnik, Optimalno permutacijsko usmerjanje, Diplomska naloga, Univerza v Mariboru, Fakulteta za naravoslovje in matematiko, Maribor, 2008.

[5] I. Sau Walls, J. Žerovnik, An Optimal Permutation Routing Algorithm for Full-Duplex Hexagonal Mesh Networks, *Discrete Mathematics and Theoretical Computer Science*, 10(3):49–62, 2008.

[6] I. Stojmenović, Honeycomb networks: topological properties and communication algorithms, *IEEE Transactions on Parallel and Distributed Systems*, 8(10):1036–1042, 1997.

[7] J. Žerovnik, Usmerjanje sporočil v grafih, *Obzornik za matematiko in fiziko*, 51(6):161-170, 2004.

Maja Rotovnik je univerzitetna diplomirana matematičarka in je zaposlena kot mlada raziskovalka na Inštitutu za matematiko, fiziko in mehaniko v Ljubljani.

Jurij Šilc je višji znanstveni sodelavec na Odseku za računalniške sisteme na Institutu "Jožef Stefan" v Ljubljani in docent za napredno procesorsko arhitekturo na Mednarodni podiplomski šoli Jožefa Stefana v Ljubljani. Raziskovalno se ukvarja z računalniškimi sistemi in strukturami ter metahevrističnim optimiziranjem.

Janez Žerovnik je redni profesor za matematiko na Univerzi v Ljubljani, do leta 2008 pa je bil redni profesor za diskretno in računalniško matematiko in redni profesor za računalništvo na Univerzi v Mariboru. Dopolnilno je zaposlen kot raziskovalec na Inštitutu za matematiko, fiziko in mehaniko v Ljubljani. Raziskovalno dela predvsem na področju diskretne optimizacije in teorije grafov z aplikacijami.