Univerzita Komenského
Fakulta Matematiky, Fyziky a Informatiky
Katedra informatiky

Robert Kasanický

Prehľad výsledkov o sieťach

Diplomová práca
Diplomový vedúci: Doc. RNDr. Ferdinand Gliviak CSc.

Bratislava 2005

Čestne prehlasujem, že som diplomovú prácu vypracoval samostatne, s odbornou pomocou školiteľa.

Bratislava, apríl 2005 Robert Kasanický

Obsah

Úvod

Moja diplomová práca *Prehľad výsledkov o sieťach* má za cieľ predstaviť čitateľovi dva významné matematické modelovacie nástroje, ktoré majú široké praktické využitie – kapacitné siete a Petriho siete.

Kapacitné siete sú pre čitateľa so základným prehľadom o teórií grafov známou témou, preto je základná teória spracovaná iba stručne, v rozsahu potrebnom pre to, aby ďalej mohli byť rozpracované niektoré zaujímavé a praktické zovšeobecnenia pôvodného modelu. Práca obsahuje aj pomerne neformálnu kapitolu o využití modelu kapacitných sietí v ekonómií, konkrétnejšie v projektovom manažmente.

V prípade Petriho sietí sa nepredpokladá, že sa s týmto modelom čitateľ už stretol, a preto je teória budovaná podstatne precíznejšie ako v prvej časti. Dôraz sa kladie najmä na pochopenie základných konceptov a predstavenie najdôležitejších problémov analýzy. Táto časť by mala slúžiť ako dobrý základ pre ďalšie štúdium problematiky, ktorá je už dnes neobyčajne široká a intenzívne sa ďalej rozvíja.

Mojou snahou bolo napísať túto prehľadovú prácu štýlom, ktorý bude dostatočne formálny na to, aby mohol byť text užitočný pri serióznom štúdiu, zároveň však bude ľahko zrozumiteľný a čitateľa nebude unavovať prílišným množstvom formálnych technických detailov. Ako sa mi to podarilo, to už posúdi sám čitateľ.

Časť I

Kapacitné siete

Kapitola 1

Teoretické základy

V tejto časti sa budeme zaoberať teóriou, ktorú prví rozpracovali páni Ford a Fulkerson. Venuje sa tokom v sieťach, pričom hlavná pozornosť sa sústreďuje na problém nájdenia maximálneho toku. Stručne zrekapitulujeme základné výsledky Forda a Fulkersona a ďalej sa budeme venovať niektorým praktickým zovšeobecneniam tejto teórie.

Tok v sieti v nami uvažovanom zmysle môže predstavovať doslovne tok ropy, či vody v systéme potrubí. Vo vedeckých prácach sa častejšie uvažuje tok elektriny, emailových správ, či rôznych tovarov. Táto práca sa sústredí prevažne práve na praktické aplikácie, no matematický model sietí a tokov v sebe skrýva podstatne väčšie bohatstvo, než iba priamočiare modelovanie a riešenie problémov súvisiacich s uvedenými príkladmi. Teória sieťových tokov spája viaceré rôznorodé a zdanlivo nesúvisiace oblasti kombinatoriky, vďaka čomu výsledky dosiahnuté v tejto teórii majú veľmi široké uplatnenie.

1.1 Toky a rezy v sieťach

Definícia 1.1.1 *Orientovaným grafom (digrafom) nazývame usporiadanú dvojicu* $G = (V, E)$*, kde* V *je množina vrcholov a* E *je množina hrán, pričom* $E \subseteq V \times V$*.*

Definícia 1.1.2 *Majme daný orientovaný graf* $G = (V, E)$*. Potom definujeme:*

$In(v) = \{e \in E : e = uv\}$ $(\forall v \in V)$
$Out(v) = \{e \in E : e = vu\}$ $(\forall v \in V)$

$e^+ = v$ *ak* $e = uv$ $(\forall e \in E)$
$e^- = v$ *ak* $e = vu$ $(\forall e \in E)$

Definícia 1.1.3 *Sieťou nazývame štvoricu* $N = (G, cap, s, t)$*, kde* G *je spojitý orientovaný graf s dvoma vyznačenými vrcholmi* s *a* t *takými, že* $In(t) > 0$ *a* $Out(s) > 0$. cap *je nezáporná kapacitná funkcia* $cap : E \rightarrow N$.

Poznámka 1.1.4 *Kapacitnú funkciu sme obmedzili na prirodzené čísla iba z estetických dôvodov. V praxi počítame iba s aproximáciami, teda racionálnymi číslami a tie vieme previesť na prirodzené vypočítaním najmenšieho spoločného násobku.*

Definícia 1.1.5 *Nech* N *je sieť. Tokom v sieti* N *sa nazýva nezáporná funkcia* $f : E \rightarrow \mathcal{R}$ *taká, že:*

1. $f(e) \leq cap(e)$
2. $\sum_{e \in In(v)} f(e) = \sum_{e \in Out(v)} f(e)$ *pre všetky vrcholy* v *rôzne od* s *a* t

Definícia 1.1.6 *Hodnotou toku* f *označujeme* $val(f)$ *a kladieme*

$$val(f) = \sum_{e \in Out(s)} f(e) - \sum_{e \in In(s)} f(e)$$

Definícia 1.1.7 *Maximálny tok* f^* *je tok s maximálnou hodnotou, teda taký, že* $val(f^*) \geq val(f)$ *pre všetky* f.

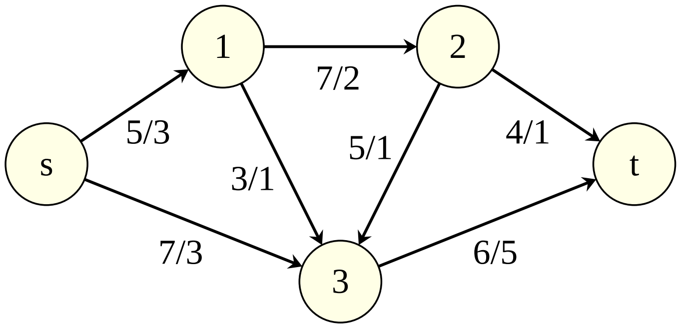
Zadefinovali sme si sieť a tok v sieti, teraz ešte uvedieme definíciu rezu a neskôr ukážeme úzku súvislosť medzi rezmi a tokmi v sieti.

Definícia 1.1.8 *Nech množiny* V_s *a* V_t *tvoria rozklad množiny* V *taký, že* $s \epsilon V_s$ *a* $t \in V_t$*. Potom množinu všetkých hrán orientovaných z množiny* V_s *do množiny* V_t *nazývame s-t rezom a označujeme* (V_s, V_t).

Definícia 1.1.9 *Kapacitu rezu označujeme* $cap(V_s, V_t)$ *a kladieme*

$$cap(V_s, V_t) = \sum_{e \in (V_s, V_t)} cap(e)$$

Definícia 1.1.10 *Minimálnym rezom nazývame rez s najmenšou kapacitou.*



Obr. 1.1: Tok v kapacitnej sieti

1.2 Výpočet maximálneho toku

Základná myšlienka algoritmu na výpočet maximálneho toku spočíva v iteratívnom zväčšovaní hodnoty toku, pričom sa začína nulovým tokom. Pri tomto postupe budeme využívať *nenasýtené polocesty*, ktoré si hneď definujeme.

Definícia 1.2.1 *Polocestou nazývame postupnosť*

$$(v_0, e_1, v_1, ..., v_k - 1, e_k, v_k)$$

vrcholov a hrán, ktoré tvoria cestu v neorientovanom zmysle.

Poznámka 1.2.2 *Polocestou je teda postupnosť vrcholov a hrán, ktoré na seba nadväzujú, pričom orientácia hrán nemusí byť súhlasná.*

Definícia 1.2.3 *s-t polocestou nazývame polocestu, ktorá začína vo vrchole s a končí vo vrchole t.*

Definícia 1.2.4 *Nenasýtenou polocestou v sieti N s tokom f nazývame polocestu takú, že*

$f(e) < cap(e)$ *ak e je dopredná hrana*

$f(e) > 0$ *ak e je spätná hrana*

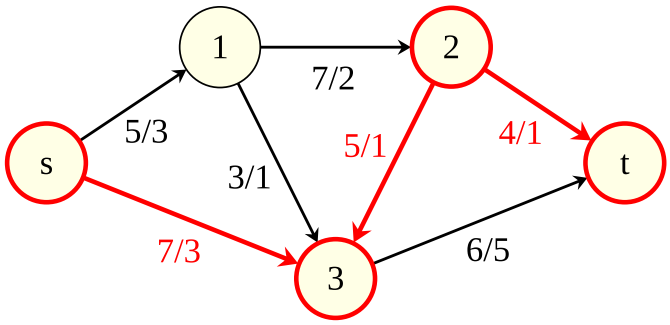
Definícia 1.2.5 *Pre každú hranu e v sieti N s tokom f zavedieme označenie*

$\Delta_e = cap(e) - f(e)$ *ak e je dopredná hrana*

$\Delta_e = f(e)$ *ak e je spätná hrana*

Definícia 1.2.6 *Pre polocestu Q zavedieme označenie*

$\Delta_Q = \min \Delta_e$



Obr. 1.2: Nenasýtená s-t polocesta v sieti s $\Delta_Q = 1$

Uvedieme vetu, ktorá ukazuje, ako sa dajú nenasýtené polocesty použiť na zväčšenie toku v sieti.

Veta 1.2.7 *Nech f je tok v sieti N a nech Q je nenasýtená s-t polocesta. Potom f' definované*

$$f'(e) = \begin{cases} f(e) + \Delta_Q, & \textit{ak e je dopredná hrana cesty } Q \\ f(e) - \Delta_Q, & \textit{ak e je spätná hrana cesty } Q \\ f(e), & \textit{inak} \end{cases}$$

je tok v sieti N a $val(f') = val(f) + \Delta_Q$.

Dôkaz: Dôkaz spočíva v jednoduchom overení, že takto definovaný f' spĺňa tokové podmienky. Čitateľ si to môže vyskúšať ako jednoduché technické cvičenie.

Veta 1.2.8 (Charakerizácia maximálneho toku) *Nech f je tok v sieti N. Potom f je maximálny tok práve vtedy, ak v N_f neexistuje nenasýtená s-t polocesta.*

Dôkaz: *Nutnosť* ($\Rightarrow$)

Nech f je maximálny tok v sieti N. Ak by existovala nenasýtená polocesta, potom by sme ju podľa predchádzajúcej vety vedeli použiť na zväčšenie toku.

Postačujúcosť ($\Leftarrow$)

Predpokladajme, že v sieti N neexistuje nenasýtená s-t polocesta a uvažujme množinu všetkých nenasýtených polociest začínajúcich vo vrchole s. Nech V_s je množina všetkých vrcholov obsiahnutých v takýchto polocestách. Pretože neexistuje nenasýtená s-t polocesta, zrejme platí $t \notin V_s$. Položme $V_t = V_N \setminus V_s$. Potom (V_s, V_t) je s-t rez a navyše platí:

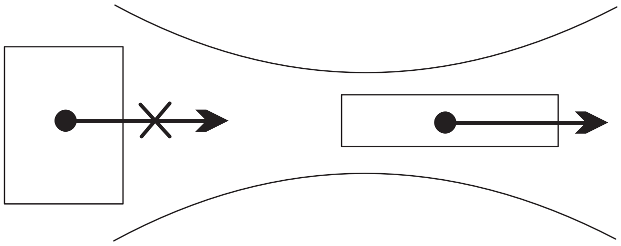
$$f(e) = \begin{cases} cap(e), & \text{ak } e \in (V_s, V_t) \\ 0, & \text{ak } e \in (V_t, V_s) \end{cases}$$

Pretože veľkosť toku v sieti je zhora ohraničená kapacitou ľubovolného rezu (tento fakt nechávame čitateľovi na rozmyslenie, vyplýva priamo z definícií rezu a toku) a v tomto prípade nastala rovnosť, musí byť f maximálny tok a (V_s, V_t) minimálny s-t rez.

Veta 1.2.9 (Max-flow Min-cut Theorem) *Pre danú sieť je hodnota maximálneho toku rovná kapacite minimálneho rezu.*

Ilustrujme si na príklade význam vety, po čom sa snáď platnosť tvrdenia stane pomerne intuitívnou. Predstavte si namiesto siete dvojrozmerné potrubie s nerovnakým priemerom. Kapacitou rezu v niektorom bode potrubia bude priemer potrubia v tomto mieste. Toky budú predmety, ktoré dokážeme cez potrubie prestrčiť na druhú stranu. Hodnotou toku bude šírka predmetu. Potom zrejme nemôžeme prestrčiť predmet širší ako je priemer potrubia v najužšom mieste. Najširší predmet, ktorý sa nám prestrčiť podarí bude mať šírku rovnú priemeru potrubia v najužšom mieste.

Keď poznáme vetu o charakterizácii maximálneho toku, ľahko vymyslíme základnú myšlienku algoritmu na nájdenie maximálneho toku: začneme nulovým tokom a budeme iteratívne hľadať nenasýtené polocesty a zvyšovať tok, až kým už žiadna nenasýtená polocesta nebude existovať. Nejasným zatiaľ zostáva spôsob ako sa budeme rozhodovať, ktorú nenasýtenú polocestu si zvolíme v danom kole iterácie (zrejme budeme mať často v danom kroku na výber viacero možností). My uvedieme



Obr. 1.3: Ilustračný obrázok k vysvetleniu vzťahu max. toku a min. rezu

riešenie Edmondsa a Karpa, ktoré je založené na myšlienke vyberať vždy najkratšiu možnú nenasýtenú polocestu. Poznamenajme však, že na toto riešenie sa neprišlo hneď, ale až potom, čo sa ukázalo, že nenasýtenú polocestu je naozaj potrebné vyberať vhodným spôsobom. O problémoch, ktoré môže spôsobiť nevhodný výber sa možno dočítať v [1] ako aj v [2].

Algoritmus 1.2.10 (Aloritmus FFEK na zostrojenie maximálneho toku) *Nech $N = (G, c, s, t)$ je sieť. Algoritmus vypočíta pre vstup N maximálny tok f a množiny S, T, také, že (V_S, V_T) je minimálny rez.*

FFEK - MAX. TOK$(N; f, S, T)$

for $e \in E$
  do $f(e) \leftarrow 0$;
 daj s značku $(-, \infty)$;
for $v \in V$
  do $u(v) \leftarrow false$; $d(v) \leftarrow \infty$;
repeat
   vyber najskôr označený vrchol v taký, že u(v)=false;
   for $e \in \{e \in E : e^- = v\}$
    do if $w = e^+$ nie je označený **and** $f(e) < c(e)$
     then $d(w) \leftarrow \min\{c(e) - f(e), d(v)\}$
      a daj vrcholu v značku $(v, +, d(w))$

   for $e \in \{e \in E : e^+ = v\}$
    do if $w = e^-$ nie je označený **and** $f(e) > 0$
     then $d(w) \leftarrow \min\{f(e), d(v)\}$
      a daj vrcholu v značku $(v, -, d(w))$

   $u(v) \leftarrow true$;

```
19          if t je označený
20             then  nech d je posledná zložka značky t;
21                   w ← t;
22                   while w ≠ s
23                         do  zisti prvú zložku v značky patriacej w;
24                             if  druhá zložka značky patriacej w je +
25                                then f(e) ← f(e) + d pre hranu e = vw
26                                else  f(e) ← f(e) − d pre hranu e = wv
27                             w ← v;
28                    vymaž všetky značky okrem tej, ktorá patrí s;
29                   for v ∈ V
30                         do d(v) ← ∞; u(v) ← false
31    until u(v) = true pre všetky označené vrcholy v;
32  Nech S je množina označených vrcholov a T ← V \ S.
```

Veta 1.2.11 (Korektnosť FFEK) *Nech N je sieť. Potom algoritmus FFEK počíta maximálny tok f a minimálny rez(S,T).*

Dôkaz: Dôkaz trdenia je uvedený v [1].

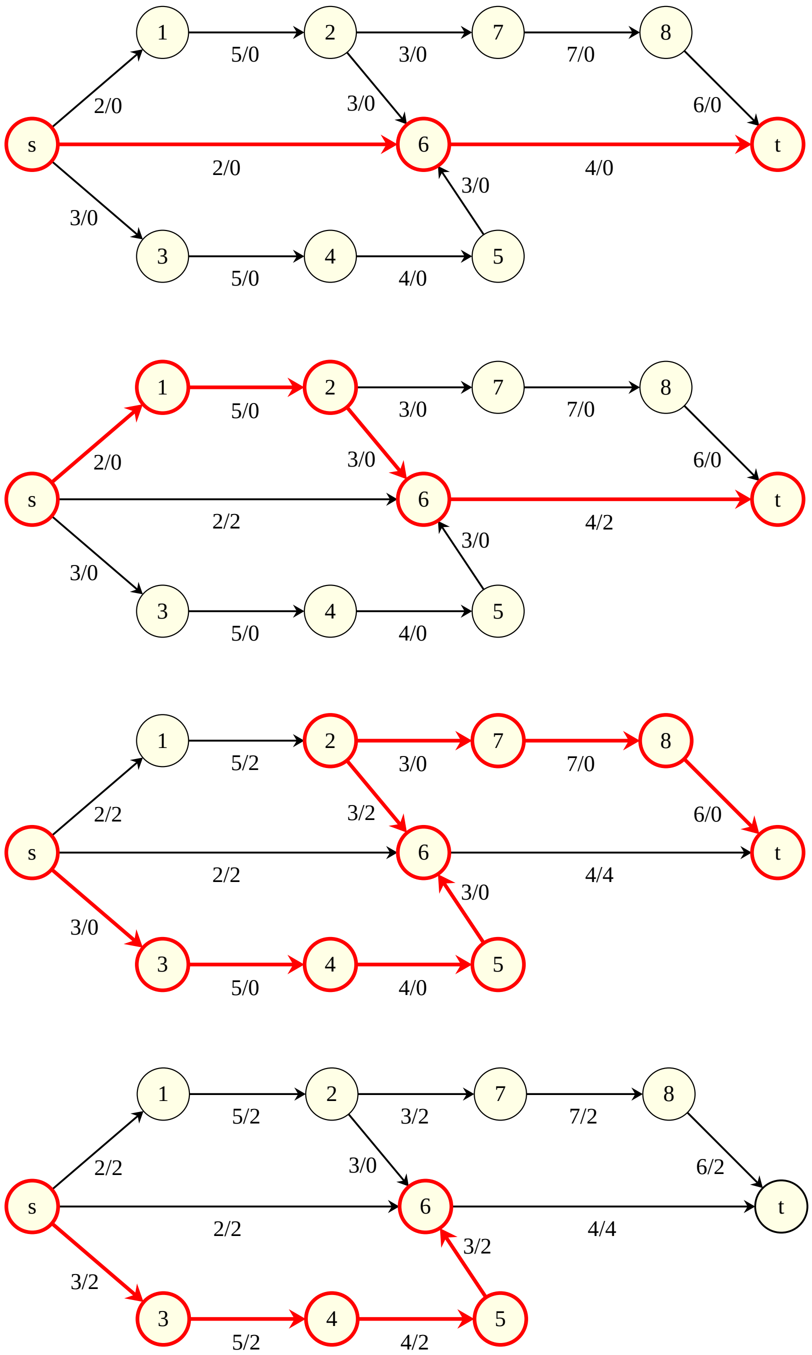
Algoritmus FFEK je pomenovaný po Fordovi, Fulkersonovi, Edmondsovi a Karpovi. Ako hovorí veta o korektnosti, tento algoritmus zostrojí maximálny tok v sieti. Na pohľad vyzerá FFEK algoritmus možno komplikovane, ale v zásade je pomerne jednoduchý. Postupne prehľadáva graf siete do šírky, pričom si značkuje vrcholy a tak konštruuje nenasýtené polocesty. Keď označí aj vrchol t, tak spätne pomocou značiek nájde nenasýtenú polocestu z t do s, ktorú použije na zväčšenie toku. Potom značky z vrcholov (s výnimkou vrochola s) zmaže. Tento postup sa iteruje pokiaľ existujú nenasýtené polocesty. Pre efektívnosť algoritmu je dôležitý riadok (7), ktorý zabezpečuje to, že sa na zväčšenie toku vyberá vždy najkratšia nenasýtená polocesta.

Veta 1.2.12 *Časová zložitosť algoritmu FFEK je $O(|V||E|^2)$.*

Dôkaz: Dôkaz trdenia je uvedený v [1].

Keďže kardinalita E sa pohybuje medzi $|V|$ a $|V|^2$, dostávame:

Dôsledok 1.2.13 *Algoritmus FFEK má časovú zložitosť $O(|V|^3)$ pre riedke grafy (napr. planárne) a $O(|V|^5)$ pre husté grafy.*



Obr. 1.4: Ilustrácia algoritmu FFEK

1.3 Význam problému maximálneho toku

Už v úvode kapitoly sme spomenuli, že problém maximálneho toku v sieti má široké uplatnenie nielen pri priamočiarom modelovaní praktických problémov, ale aj v oblasti abstraktnej matematiky. Uvedieme teraz niektoré významné tvrdenia z teórie grafov, ktoré s problémom maximálneho toku úzko súvisia a následne vysvetlíme v akom sú vzťahu. Poznamenávame, že uvedené tvrdenia obsahujú pojmy, ktoré sme nedefinovali, ale z teórie grafov sú veľmi dobre známe. Preto je táto časť určená najmä čitateľom so základným prehľadom o teórii grafov, ktorým sú nasledujúce tvrdenia dobre známe.

Veta 1.3.1 (Max-flow Min-cut Theorem) *Pre danú sieť N je hodnota maximálneho toku rovná kapacite minimálneho rezu.*

Veta 1.3.2 (Mengerova veta - hranová verzia) *Nech s a t sú dva rôzne vrcholy grafu G. Potom maximálny počet hranovo disjunktných s-t ciest v grafe G je rovný minimálnemu počtu hrán v s-t oddeľujúcej množine grafu G.*

Veta 1.3.3 (Mengerova veta - vrcholová verzia) *Nech s a t sú dva rôzne vrcholy grafu G. Potom maximálny počet vrcholovo disjunktných s-t ciest v grafe G je rovný minimálnemu počtu vrcholov v s-t oddeľujúcej množine grafu G.*

Veta 1.3.4 (Königova veta) *Nech G je bipartitný graf. Potom počet hrán maximálneho párenia grafu G je rovný počtu vrcholov v minimálnom vrcholovom pokrytí grafu G.*

Veta 1.3.5 (Königova-Egerváryho veta) *Nech A je matica obsahujúca iba prvky 0 a 1. Potom maximálny počet 1-tiek takých, že žiadne dve sa nenachádzajú v rovnakom riadku ani stĺpci je rovný minimálnemu počtu riadkov a stĺpcov, ktoré dokopy obsahujú všetky 1-tky v matici A*

Veta 1.3.6 (Hallova veta) *Nech G je bipartitný graf s partíciami $\{X, Y\}$. Potom G obsahuje párenie X práve vtedy, keď*

$$|W| \leq |N(W)| \text{ pre všetky } W \subseteq X,$$

pričom $N(W)$ označuje množinu všetkých susedov vrcholov patriacich do W.

Skutočnosť je taká, že uvedené tvrdenia sú v istom zmysle ekvivalentné, presnejšie z ľubovolného jedného z týchto tvrdení možno dokázať všetky ostatné. Nebudeme uvádzať transformácie jednotlivých problémov na ostané, čitateľ niektoré z nich môže nájsť v [2].

Nami bližšie rozoberaná Max-Flow Min-Cut Theorem má teda principiálne rovnakú výpovednú hodnotu ako ostatné uvedené tvrdenia, je však výnimočná svojím dôkazom, ktorý je konštruktívny. So znalosťou tohoto dôkazu sme boli schopní už pomerne priamočiaro zostrojiť aj algoritmus na nájdenie maximálneho toku, resp. minimálneho rezu. Vďaka tomu, že ľubovolný z horeuvedených problémov vieme transformovať na problém nájdenia maximálneho toku, znamená to, že pomocou algorimu FFEK vieme efektívne nájsť aj maximálne párenie grafu (resp. minimálne vrcholové pokrytie) a tak ďalej aj pre ostatné uvedené problémy. Problémy o ktorých už vieme, že sú ekvivalentné s problémom maximálneho toku, sú abstraktné matematické problémy a existuje opäť veľké množstvo praktických problémov, ktoré na ne vieme previesť. Algoritmus na nájdenie maximálneho toku týmto získava neobyčajne široké uplatnenie a veľký význam.

Kapitola 2

Zovšeobecnenia modelu sietí

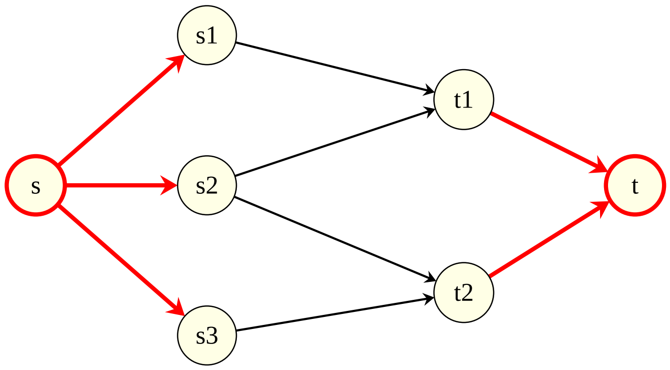
2.1 Siete s viacerými zdrojmi a ústiami

V prvom uvažovanom zovšeobecnení sa budeme zaoberať tým, ako vypočítať maximálny tok, ak dovolíme aby sieť mala viacero zdrojov a ústí. Takéto zovšeobecnenie je pomerne prirodzené najmä z praktického hľadiska, kde zdroj spravidla reprezentuje miesto kde sa niečo vyrába a ústie miesto kde sa spotrebováva. Takých miest býva často viacero.

Riešenie tohoto problému je našťastie veľmi jednoduché a vyžaduje iba malú transformáciu pôvodnej siete, na ktorej potom požijeme algoritmus FFEK. Konkrétne sieť s viacerými zdrojmi a ústiami transformujeme na sieť s jedným zdrojom a ústím tak, že pridáme dva nové vrcholy (nový zdroj a ústie) a tieto napojíme na pôvodné zdroje a ústia hranami s dostatočne veľkou kapacitou. Algoritmus FFEK spustený na takejto sieti potom zrejme počíta tok, ktorý je maximálny pre pôvodnú sieť.

2.2 Siete so spodnou hranicou pre tok

V druhom zovšeobecnení budeme uvažovať siete, ktoré majú navyše pre každú hranu určenú nezápornú spodnú hranicu pre tok, teda číslo pod ktoré nesmie hodnota toku klesnúť. Toto už je očividne vážnejší zásah do pôvodného konceptu - zatiaľ čo v pôvodnej teórií tok v sieti vždy existoval (nulový tok vždy spĺňal tokové podmienky), keď pridáme spodné hranice tak problémom sa stáva samotná existencia toku, ktorý spĺňa dodatočné podmienky (takýto tok budeme nazývať prípustný).



Obr. 2.1: Riešenie pre viacero zdrojov a ústí

Pridanie funkcie spodnej hranice je v našom prípade motivované praxou. Ak sieťou modelujeme napríklad prepravu súčiastok pomocou kamiónov, potom by zrejme nikto neposlal na cestu kamión vezúci iba jednu skrutku. Pre podobné prípady je vhodné vedieť riešiť problém maximálneho toku aj v sieťach s danou funkciou spodnej hranice.

Budeme postupovať takto: najprv sformulujeme algoritmus, ktorý zostrojí nejaký tok v sieti (za predpokladu, že tok existuje), inak vráti informáciu, že v danej sieti neexistuje prípustný tok. Keď budeme mať zostrojený nejaký prípustný tok, budeme ho pomocou mierne modifikovaného algoritmu FFEK iteratívne zväčšovať až na maximálny možný.

Definícia 2.2.1 *Nech $N = (G, cap, s, t)$ je sieť nech $b(e)$ je k nej prislúchajúca funkcia spodnej hranice. Potom tok f na sieti N nazývame prípustný, ak platí*

$$b(e) \leq f(e) \text{ pre všetky } e \in E(G)$$

Definícia 2.2.2 *Majme danú sieť $N = (G, cap, s, t)$ a k nej funkciu spodnej hranice $b : E(G) \rightarrow \mathcal{R}_0^+$. K sieti N definujeme pomocnú sieť $N' = (H, cap', s', t')$, kde*

$$V(H) = V(G) \cup \{s', t'\}$$
$$E(H) = E(G) \cup \{ts\} \cup \{s'v : v \in V(G)\} \cup \{s'v : v \in V(G)\}$$

$cap'(e) = cap(e) - b(e)$ *pre* $e \in E(G)$
$cap'(s'v) = \sum_{e^+=v} b(e)$ *pre každý* $v \in V(G)$
$cap'(vt') = \sum_{e^-=v} b(e)$ *pre každý* $v \in V(G)$
$cap'(ts) = \infty$

Poznámka 2.2.3 *Symbolom* ∞ *označujeme nejaké dostatočne veľké číslo, napríklad to môže byť súčet kapacít všetkých hrán siete* N*.*

Nasledujúca veta ukazuje, prečo je pre nás takto definovaná pomocná sieť zaujímavá.

Veta 2.2.4 *Nech* $N = (G, cap, s, t)$ *je sieť a nech* $b(e)$ *je k nej prislúchajúca funkcia spodnej hranice. Potom v sieti* N *existuje prípustný tok práve vtedy, keď maximálny tok pre pomocnú sieť* $N' = (H, cap', s', t')$ *má hodnotu* $W = \sum_{e \in E(G)} b(e)$*.*

Dôkaz: Označme W maximálnu hodmotu toku v sieti N'. Ľahko vidno, že platí

$$W = \sum_v cap'(s'v) = \sum_e b(e) = \sum_v cap'(vt')$$

Nech f' je tok na sieti N' s hodnotou $val(f') = W$. Definujeme tok f na sieti N takto:

$$f(e) = f'(e) + b(e) \tag{2.1}$$

Pretože f' spĺňa podmienku $0 \leq f'(e) \leq cap'(e) = cap(e) - b(e)$, hneď dostávame, že f spĺňa dolné aj kapacitné ohraničenie, teda že $b(e) \leq f(e) \leq cap(e)$ pre všetky hrany e. Na to aby sme dokázali, že f je prípustný tok musíme ešte ukázať, že spĺňa aj podmienku zachovania toku. Nech $v \in V(G) \setminus \{s, t\}$. f' je tok na sieti N', preto platí

$$f'(s'v) + \sum_{e^+=v} f'(e) = f'(vt') + \sum_{e^-=v} f'(e) \tag{2.2}$$

Pretože $val(f') = W$, všetky hrany z H incidentné s vrcholmi s' alebo t' sú saturované, takže platí

$$f'(s'v) = \sum_{e^+=v} b(e) \text{ a zároveň } f'(vt') = \sum_{e^-=v} b(e). \tag{2.3}$$

Z (2.1) a (2.2) dostávame

$$\sum_{e^+=v} f(e) = \sum_{e^-=v} f(e) \tag{2.4}$$

Tým sme dokázali prvú implikáciu, teraz ešte dokážeme opačnú implikáciu. Nech f je prípustný tok na N. Potom môžeme definovať f' pre N' pomocou (2.1) a (2.3). Pretože f je prípustný tok, máme $0 \leq f'(e) \leq cap'(e)$ pre hrany $e \in E(G)$. Z (2.3) vyplýva, že platí $cap'(s'v) = f'(s'v)$ a $cap'(vt') = f'(vt')$. Všetky hrany incidentné s s' sú teda saturované, z čoho plynie, že $val(f') = W$. Je zrejmé, že f' je tok, pretože (2.1), (2.3) a (2.4) implikujú (2.2). Tým je dôkaz tvrdenia podaný.

Z dôkazu vety dostávame priamočiaro nasledujúci agoritmus na konštrukciu prípustného toku.

Algoritmus 2.2.5 *Nech* $N = (G, cap, s, t)$ *je sieť a nech* $b(e)$ *je funkcia spodnej hranice pre sieť* N. *Hodnota legal indikuje existenciu prípustného toku,* f *je maximálny tok.*

LEGALFLOW($N, b; legal, f$)
Zostroj pomocnú sieť N' k sieti N
$FFEK(N'; f', S, T)$;
if $f'(sv) = cap'(sv)$ pre všetky $v \in V(G)$
    then $legal \leftarrow$ **true**
    else $legal \leftarrow$ **false**
if $legal =$ **true**
    then for $e \in E(G)$
            do $f(e) \leftarrow f'(e) + b(e)$

Vyriešili sme problém konštrukcie prípustného toku a máme k dispozícií algoritmus FFEK, ktorý dokáže tok iteratívne zvyšovať až na maximum. Zostáva ešte ošetriť to, aby sa pri zvyšovaní toku nenarušili podmienky spodnej hranice, s ktorými algoritmus FFEK nepočíta. To môžeme docieliť tak, že modifikujeme pôvodný algoritmus. Konkrétne v riadku (14) nahradíme výraz $f(e) > 0$ výrazom $f(e) > b(e)$ a v riadku (15) nahradíme $\min\{f(e), d(v)\}$ výrazom $\min\{f(e) - b(e), d(v)\}$. Úprava v riadku (14) hovorí, že uvažovať o znížení toku na hrane môžeme iba vtedy, keď jeho hodnota je ostro väčšia ako spodná hranica. Úprava v riadku (15) hovorí, že tok na hrane môžeme znížiť najviac tak, aby neklesol pod spodnú hranicu.

2.3 Siete s ocenenými hranami

Posledné zovšeobecnenie sa zaoberá problémom nájdenia najlacnejšieho maximálneho toku, keď máme danú nezápornú cenovú funkciu na hranách. Takéto zovšeobecnenie je opäť prakticky užitočné. Ak uvažujeme už raz spomínaný príklad prepravy súčiastok pomocou kamiónov, potom cenová funkcia môže vyjadrovať dĺžku cesty medzi jednotlivými miestami, reprezentovanými vrcholmi grafu.

Algoritmus FFEK ukazuje svoju silu aj v tomto zovšeobecnení. Ako ďalej uvidíme, na riešenie problému stačí, ak požijeme algoritmus FFEK modifikovaný tak, že zvyšuje tok pozdĺž najlacnejšej nenasýtenej polocesty. Opäť si zadefinujeme pomocnú sieť a uvedieme lemu, ktorá nám zdôvodní správnosť algoritmu.

Definícia 2.3.1 *Nech $N = (G, cap, s, t)$ je sieť s cenovou funkciou γ. Definujeme cenovú funkciu pre tok f v sieti N takto:*

$$\gamma(f) = \sum_{e} \gamma(e) f(e)$$

Definícia 2.3.2 *Nech $N = (G, cap, s, t)$ je sieť s cenovou funkciou γ. Definujeme k nej pomocnú sieť $N' = (G', cap', s, t)$ s cenovou funkciou γ' takto:*

1. *$V(G') = V(G)$*
2. *pre každú hranu $e \in E(G)$ takú, že $e = uv$ a $f(e) < cap(e)$ pridaj do $E(G')$ hranu $e' = uv$, polož $cap'(e') = cap(e) - f(e)$ a $\gamma'(e') = \gamma(e)$*
3. *pre každú hranu $e \in E(G)$ takú, že $e = uv$ a $f(e) \neq 0$ pridaj hranu do $E(G')$ hranu $e'' = vu$, polož $cap'(e'') = f(e)$ a $\gamma'(e'') = -\gamma(e)$*

Lema 2.3.3 *Nech $N = (G, cap, s, t)$ je sieť s cenovou funkciou γ. Ďalej nech f je najlacnejší tok taký, že $val(f) = w$ a nech W je najlacnejšia cesta v pomocnej sieti $N' = (G', cap', s, t)$ vzhľadom na cenovú funkciu γ'. Ak f' je tok, ktorý dostaneme zvýšením toku f pozdĺž cesty W (s kapacitou δ), potom f' je najlacnejší tok taký, že $val(f') = w + \delta$.*

Lema formálne vyjadruje myšlienku, že ak máme v sieti najlacnejší tok danej hodnoty a zväčšíme tok podĺž najlacnejšej nenasýtenej polocesty, potom získame opäť najlacnejší tok pre vyššiu hodnotu toku. Dôkaz lemy je čiste technického charakteru, preto ho neuvádzame (možno ho nájsť v [1]).

Nasledujúci algoritmus počíta najlacnejší tok honoty v, takže pomocou neho je možné nielen vypočítať maximálny tok postupným zvyšovaním hodnoty v, ale aj nájsť zobrazenie $v \to \gamma(v)$ (tzv. *cost curve*).

Algoritmus 2.3.4 *Nech* $N = (G, cap, s, t)$ *je sieť a nech* γ *je cenová funkcia na hranách siete* N.

OPTFLOW$(N, \gamma, v; f, sol)$
for $e \in E$
    do $f(e) \leftarrow 0$
$sol \leftarrow$ **true**; **val** $\leftarrow$ **0**;
while $sol = true$ **and** $val < v$
    do zostroj pomocnú sieť $N' = (G', cap', s, t)$ s cenovou funkciou γ';
        if t nie je dosiahnuteľný z s v G'
          then $sol \leftarrow$ **false**
          else nájdi najlacnejšiu cestu W z s do t v (G', γ') ;
              $\delta \leftarrow \min\{cap'(e) : e \in W\}$; $\delta' \leftarrow \min\{\delta, v - val\}$;
              $val \leftarrow val + \delta'$;
              zväčši f na ceste W o δ'

Kapitola 3

Siete projektov

Kapacitné siete majú významné uplatnenie v ekonómií. Priblížime si teóriu sieťovej analýzy, ktorá projektu priradzuje kapacitnú sieť, na ktorej následne dokáže analyzovať vlastnosti projektu. Sieťová analýza je silným nástrojom využívaným pri manažmente projektov.

Táto kapitola bude vyložená iba voľne, pretože s pohľadu tejto diplomovej práce nie je ťažisková. Podrobnejšie sa môže čitateľ s problematikou sieťovej analýzy v ekonómií oboznámiť v [3] a v [5].

3.1 Digraf projektu

Pod pojmom projektu rozumieme nejakú komplexnú úlohu, ako napríklad výstavbu obchodného centra, údržbu bloku elektrárne, zavedenie nového výrobku na trh alebo vývoj softvéru.

Projekt pozostáva z veľkého počtu čiastkových činností, ktoré my pri modelovaní budeme považovať za nedeliteľné. Projekt môžeme zrejme rozdeliť na činnosti rôznymi spôsobmi v závislosti od toho, aký stupeň abstrakcie si zvolíme. Tento problém my riešiť nebudeme. Budeme uvažovať, že máme dané činnosti, z ktorých projekt pozostáva a ku každej činnosti je určené jej trvanie. Ďalej pre každú činnosť máme daný zoznam činností, ktoré jej bezprostredne predchádzajú. Takýto projekt sa nazýva konjunktívno-deterministický.

Projektu môžeme priradiť digraf dvoma odlišnými spôsobmi, v závislosti od toho, či činnosti projektu reprezentujeme vrcholmi alebo hranami digrafu. My si ukážeme spôsob, ktorý reprezentuje činnosti vrcholmi. Tento prístup považujem za lepší a je využívaný aj vo všetkých moderných softvérových implementáciach.

Ku danému projektu P konštruujeme jeho digraf projektu D takto:

1. Každú činnosť i projektu P reprezentujeme vrcholom i digrafu D.

2. Dvojicu vrcholov i, j spojíme hranou (i, j) práve vtedy, keď činnosť i v projekte P bezprostredne predchádza činnosti j. Hranu (i, j) ohodnotíme číslom w_{ij}, rovným počtu časových jednotiek potrebných na vykonanie činnosti i v projekte P.

3. Do digrafu D pridáme dva nové vrcholy s pre začiatok a z pre koniec a nasledovné hrany:

    - (s, x) pre každý vrchol x, pre ktorý $In(x) = 0$, pričom hranu ohodnotíme číslom 0.

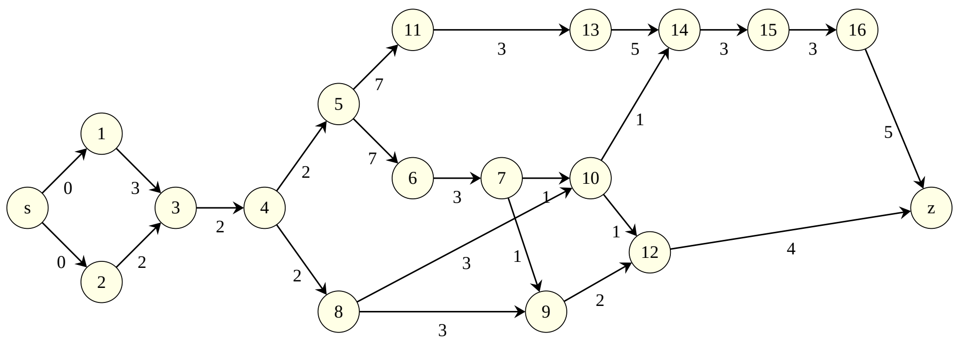
    - (y, z) pre každý vrchol y, pre ktorý $Out(y) = 0$, pričom hranu ohodnotíme číslom w_{yz}, rovným počtu časových jednotiek potrebných na vykonanie činnosti y v projekte P.

4. Iné vrcholy a hrany do digrafu D projektu P nepridávame.

Poznamenávame, že ak je projekt zadaný zmysluplne, potom získaný graf je acyklický a jeho vrcholy môžeme topologicky usporiadať v čase $O(E)$(pri topologickom usporiadaní platí, že ak $(i, j) \in E(D)$, potom $i < j$).

Uvedieme teraz na ilustráciu príklad výstavby rodinného domu, ktorý je prebraný z [3].

Číslo	Činnosť	Čas	Predispozície
1	Príprava stavebného pozemku	3	-
2	Zabezpečenie stavebného materiálu	2	-
3	Výkop základov	2	1,2
4	Výstavba základov	2	3
5	Výstavba stien	7	4
6	Urobenie základov strechy	3	5
7	Pokrytie strechy	1	6
8	Urobenie potrubí zvonka	3	4
9	Omietka zvonku	2	7,8
10	Osadenie okien	1	7,8
11	Prístup na povalu	3	5
12	Upratanie záhrady	4	9,10
13	Vedenie vo vnútri	5	11
14	Inštalácie na stenách	3	10,13
15	Namaľovanie stien	3	14
16	Nasťahovanie	5	15



Obr. 3.1: Digraf projektu výstavby rodinného domu

3.2 Časová analýza projektu

Pokiaľ nechceme pri realizácií projektu zbytočne strácať čas a peniaze, je potrebné jednotlivé činnosti projektu vhodne koordinovať. Ukážeme si, ako s využitím digrafu projektu dokážeme vypočítať najkratší čas potrebný na realizáciu projektu, ako aj časový interval, kedy má byť realizovaná každá jednotlivá činnosť projektu.

Označme t_i najskôr možný začiatok činnosti i. Pretože všetky činnosti bezprostredne predchádzajúce činnosti i musia už byť dokončené skôr, dostávame nasledujúci rekurentný vzťah pre výpočet t_i. Obvykle sa nazýva výpočet vpred.

$t_s = 0$
$t_i = \max\{t_k + w_{ki}\}$ pre všetky hrany $(k, i) \in E(D)$

Analogicky, keď označíme T_i najneskôr možný začiatok činnosti $i$taký, aby projekt P bol ukončený najneskôr v čase T, odvodíme výpočet vzad. Obvykle kladieme $T = t_z$.

$T_z = T$
$T_i = \min\{T_j - w_{ij}\}$ pre všetky hrany $(i, j) \in E(D)$

Ľahko vidno, že minimálny čas potrebný na realizáciu projektu P je rovný dĺžke $t_z = T$ najdlhšej cesty digrafu D od štartu s do cieľa z. Takejto najdlhšej ceste hovoríme kritická cesta. Vo všeobecnosti môže mať projekt zrejme viacero kritických ciest.

Rozdiel $m = T_i - t_i$ medzi najneskôr a najskôr možným začiatkom činnosti i sa nazýva celková rezerva činnosti i.

Dôležitým problémom je nájsť kritickú cestu (resp. všetky kritické cesty). Popíšeme si neformálne postup, z ktorého by čitateľ mal získať dobrú predstavu o konkrétnom algoritme. Základnou myšlienkou je ku každému vrcholu v v digrafe D priradiť značku $l(v)$ tak, aby na konci značka $l(v)$ obsahovala maximálnu dĺžku $s-v$ cesty (v časových jednotkách). Označíme $t(v)$ čas potrebný na vykonanie činnosti v a vrcholy označkujeme takto:

$l(s) = 0$
$l(v) = \max\{l(u) + t(v) \mid (u, v) \in E(D)\}$

Pri značkovaní vrcholu v si navyše budeme pamätať

$$PARENT(v) = \{u \mid (u, v) \in E(D) \wedge l(v) = l(u) + t(v)\}$$

pomocou čoho po označení vrcholu z spätne zrekonštruujeme kritické cesty.

3.3 Analýza zdrojov

Pod pojmom zdroje projektu rozumieme materiály, mechanizmy, pracovné sily a podobne. Vyššie sme uvažovali časový plán projektu za predpokladu, že všetky zdroje máme k dispozícií v dostatočnom množstve, čo však v praxi nie je pravidlom. Časté sú situácie, kedy požiadavky na zdroje v niektorých časových okamihoch prevyšujú množstvá, ktoré sú k dispozícií. Problémy v praxi spôsobuje aj vysoko nerovnomerné využívanie zdrojov. Zdrojová analýza projektu teda rieši dve základné úlohy:

1. Vyrovnávanie nárokov na zdroje pri dodržaní plánovaného termínu ukončenia projektu. Používajú sa tri prístupy, pri ktorých sa minimalizuje:
   (a) Disperzia nárokov na zdroje
   (b) Maximálna odchýlka od priemeru nárokov na zdroje
   (c) Maximálne nároky na zdroje
2. Prideľovanie obmedzených zdrojov - chceme nájsť také využitie obmedzeného množstva zdrojov, ktoré zabezpečí najkratšiu dobu realizácie projektu.

Známe metódy riešenia týchto úloh majú heuristický charakter a sú staré desiatky rokov. Aktuálne algoritmy používané v komerčných softvérových produktoch nie sú zverejňované. Nebudeme sa preto analýzou zdrojov projektu bližšie zaoberať.

3.4 Analýza nákladov

Pod pojmom náklady projektu rozumieme peňažné prostriedky na jeho realizáciu. Dali by sa zaradiť pod zdroje projektu, avšak medzi ostatnými zdrojmi majú osobité postavenie, pretože spravidla je možné za cenu zvýšenia nákladov znižovať čas potrebný na realizáciu projektu. Nákladová analýza rieši problém optimálneho zosúladenia výšky nákladov a dĺžky trvania projektu. Má dve hlavné úlohy:

1. Nájsť minimálne náklady projektu pri stanovenom termíne ukončenia
2. Nájsť takú dobu trvania projektu, pre ktorú budú náklady na projekt minimálne

Pre správne pochopenie najmä druhého bodu je potrebné vysvetliť, že rozlišujeme dva druhy nákladov:

1. *Priame náklady* na činnosť sú finančné náklady na vykonanie činnosti projektu. Priame náklady projektu potom získame ako súčet nákladov všetkých činností.

2. *Nepriame náklady* sú také náklady projektu, ktoré nemožno prisúdiť konkrétnej činnosti, ale možno ich priradiť projektu ako celku. Zvyčajne sa udávajú ako náklady na jednotku času.

Výška priamych aj nepriamych nákladov je závislá od dĺžky trvania projektu. Priame náklady rastú so skracovaním dĺžky projektu, nepriame náklady naopak klesajú.

Pre nákladovú analýzu je kľúčovým problémom výpočet nákladovej krivky projektu, teda nájdenie závislosti medzi trvaním projektu a jeho nákladmi. Je to netriviálny optimalizačný problém a existuje viacero prístupov k jeho riešeniu. Vzhľadom na obmedzený rozsah kapitoly o sieťach projektov uvádzame iba ako fakt, že jeden z možných prístupov je založený na riešení problému maximálneho toku.

Časť II

Petriho siete

Kapitola 4

Definícia Petriho siete

Petriho siete sú širokou triedou diskrétnych matematických modelov, ktoré umožňujú popisovať riadiace toky a závislosti vovnútri modelovaných systémov. Zaviedol ich v roku 1962 nemecký matematik C. A. Petri.

V súčasnosti sa Petriho siete využívajú pri modelovaní a analýze paralelných a distribuovaných systémov. To zahŕňa nielen modelovanie počítačových sietí alebo priemyselných systémov, ale aj popis a analýzu komunikačných protokolov, paralelných architektúr, paralelných a distribuovaných algoritmov a mnohých ďalších. Petriho siete sa používajú napríklad aj na popis sémantiky objektového programovacieho jazyka Smalltalk.

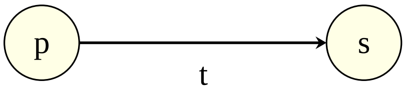
Uvedené príklady ilustrujú, že Petriho siete sú silným nástrojom nielen pri praktickom modelovaní, ale aj v teoretickej oblasti.

4.1 Základné koncepty

Skôr ako pristúpime k formálnej definícií Petriho siete, neformálne si vysvetlíme základné koncepty tohoto modelu. Robíme tak vo viere, že formálne definície budú po takomto úvode už ľahko zrozumiteľné.

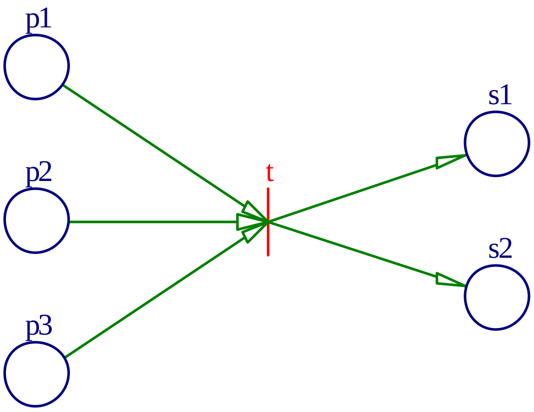
Petriho siete vznikli rozšírením modelovacích schopností konečných automatov. Tie popisujú systém pomocou konečnej množiny stavov a prechodov medzi stavmi.

Často stav systému je daný stavmi svojich subsystémov, takúto skutočnosť ale nevieme v jazyku konečných automatov dobre vyjadriť. Ak stav p resp. s je účelné popísať pomocou podmienok alebo parciálnych stavov napr. p_1, p_2, p_3, resp. s_1, s_2,



Obr. 4.1: Modelovanie pomocou konečného automatu

potom v jazyku Petriho sietí vyjadríme situáciu tak ako je naznačené na obrázku.



Obr. 4.2: Modelovanie pomocou Petriho siete

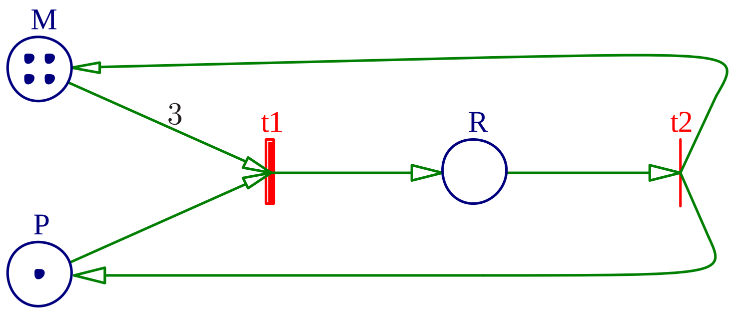
Distribúcia stavu systému na parciálne stavy a ich spojenie s udalosťami systému tvorí koncepčné jadro prístupu Petriho sietí. Parciálne stavy zobrazujeme graficky ako kružnice a ďalej ich budeme nazývať **miesta Petriho siete**. Udalosti, ktoré znázorňujeme ako úsečky, nazývame **prechody Petriho siete**.

Parciálne stavy môžeme niekedy vyjadriť v tvare nejakých logických podmienok. Situáciu na obrázku potom možno interpretovať takto: udalosť u môže nastať, ak sú splnené podmienky zodpovedajúce miestam p_1, p_2, p_3, a ak nastane spôsobí, že systém prejde do stavu, v ktorom sú splnené podmienký zodpovedajúce miestam s_1, s_2.

Na vyjadrenie stavu systému slúži v prípade Petriho sietí **značkovanie miest**. Značky znázorňujeme ako čierne bodky. V prípade keď miesta Petriho siete vyjadrujú logické podmienky, prítomnosť značky v určitom mieste znamená, že podmienka je splnená, neprítomnosť značky znamená, že podmienka splnená nie je. Nutnou podmienkou aby mohla prebehnúť nejaká udalosť je potom prítomnosť značiek vo všetkých vstupných miestach prechodu, ktorý zodpovedá danej udalosti. Vykonanie prechodu realizujeme tak, že odstránime značky zo vstupných miest a pridáme značky výstupným miestam prechodu.

My nebudeme ďalej zužovať interpretáciu miest siete iba na logické podmienky, ale budeme ich chápať všeobecnejšie, ako parciálne stavy systému a preto dovolíme, aby malo jedno miesto aj viacero značiek. Každé miesto siete teda bude ohodnotené prirodzeným číslom vyjadrujúcim aktuálny počet značiek v tomto mieste. Ďalej ohodnotíme aj hrany, celými kladnými číslami, pričom hodnoty budú vyjadrovať, koľko značiek pri vykonaní prechodu uberieme zo vstupných miest a koľko značiek pribudne vo výstupných miestach. Napokon ešte pridáme miestam kapacity, teda každé miesto bude mať určenú hornú hranicu pre počet značiek (neohraničený počet budeme označovať ∞). Ak by spustenie prechodu malo spôsobiť prekročenie kapacity niektorého miesta, potom tento prechod nemôžeme spustiť ani keby mal vo vstupných miestach dostatok značiek.

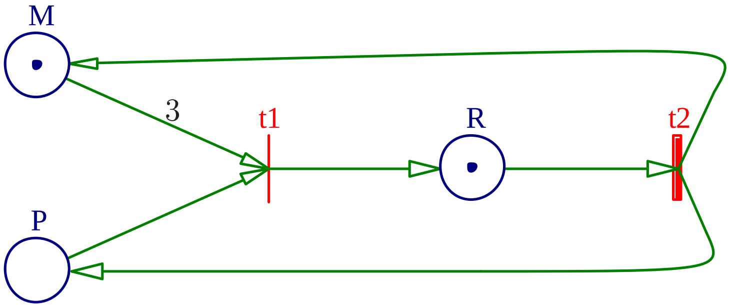
Tento model ilustrujú obrázky (4.3) a (4.4):



Obr. 4.3: Sieť so spustiteľným prechodom t_1

4.2 Definície

Definícia 4.2.1 *Sieťou nazývame trojicu $N = (P, T, F)$, kde P a T sú disjunktné množiny a $F \subseteq (P \times T) \cup (T \times P)$ je binárna relácia.*



Obr. 4.4: Sieť po spustení prechodu t_1

Prvky množiny P nazývame miesta (places) siete N, prvky množiny T nazývame prechody (transitions) siete N. F sa nazýva toková (flow) relácia.

Definícia 4.2.2 *Graf siete $N = (P, T, F)$ je daný množinou vrcholov $P \cup T$ a množinou hrán F. Miesta znázorňujeme ako kružnice, prechody ako úsečky.*

Definícia 4.2.3 *Nech $N = (P, T, F)$ je sieť. Potom pre všetky $x \in (P \cup T)$*
$\cdot x = \{y : yFx\}$ *sa nazýva vstupnou množinou prvku x*
$x \cdot = \{y : xFy\}$ *sa nazýva výstupnou množinou prvku x*

Definícia 4.2.4 *Šesticu $PN = (P, T, F, W, K, M_0)$ nazývame Petriho sieťou, pričom platí:*

1. *(P, T, F) je konečná sieť*
2. *$W : F \to \mathcal{N} \setminus \{0\}$ je ohodnotenie hrán grafu siete určujúce váhu hrán*
3. *$K : P \to \mathcal{N} \cup \{\infty\}$ je zobrazenie určujúce kapacitu miest siete*
4. *$M_0 : P \to \mathcal{N} \cup \{\infty\}$ je počiatočné označkovanie miest, rešpektujúce kapacity miest, t.j. $M_0(p) \leq K(p)$ pre všetky $p \in P$*

Definícia 4.2.5 *Nech $PN = (P, T, F, W, K, M_0)$ je Petriho sieť*

1. *Zobrazenie $M : P \to \mathcal{N} \cup \{\infty\}$ nazývame označkovanie Petriho siete PN, ak platí $M(p) \leq K(p)$ pre všetky $p \in P$*

2. *Prechod* t *je* M*-spustiteľný, ak platí:*
   $\forall p \in \cdot t : M(p) \geq W(p,t)$
   $\forall p \in t\cdot : M(p) \leq K(p) - W(t,p)$

3. *Ak je* $t \in T$ M*-spustiteľný, potom po jeho spustení pri označkovaní* M *dostaneme označkovanie* M' *definované nasledovne:*

$$M'(p) = \begin{cases} M(p) - W(p,t), & ak\ p \in \cdot t \setminus t\cdot \\ M(p) + W(t,p), & ak\ p \in t\cdot \setminus \cdot t \\ M(p) - W(p,t) + W(t,p), & ak\ p \in \cdot t \cap t\cdot \\ M(p), & inak \end{cases}$$

   Spustenie prechodu t *z označkovania* M *do označkovania* M' *zapisujeme nasledovne:*

$$M[t\rangle M'.$$

4. *Symbolom* $[M\rangle$ *označíme najmenšiu množinu označkovaní siete* PN *takú, že platí:*

   (a) $M \in M'$

   (b) *ak* $M_1 \in [M\rangle$ *a pre niektoré* $t \in T$ *platí* $M_1[t\rangle M_2$*, potom* $M_2 \in [M\rangle$

   Množinu $[M\rangle$ *nazývame množinou dosiahnuteľných označkovaní z označkovania* M*. Množina* $[M_0\rangle$ *sa nazýva množina dosiahnuteľných označkovaní siete* PN*.*

Definícia 4.2.6 *Nech* $PN = (P, T, F, W, K, M_0)$ *je Petriho sieť a* $[M_0\rangle$ *je jej množina dosiahnuteľných označkovaní. Prechodovou funkciou Petriho siete* PN *budeme nazývať (čiastočnú) funkciu* $\delta : [M_0\rangle \times T \rightarrow [M_0\rangle$ *pre ktorú platí:*

$$(\forall t \in T)(\forall M, M' \in [M_0\rangle)(\delta(M,t) = M' \Leftrightarrow M[t\rangle M')$$

4.3 Komplementácia Petriho siete

Ukážeme si jednu zaujímavú a zároveň významnú transformáciu Petriho siete, najprv ale uvedieme potrebnú definíciu.

Definícia 4.3.1 *Petriho sieť* $PN = (P, T, F, W, K, M_0)$ *sa nazýva bezkontaktná, ak pre všetky* $M \in [M_0\rangle$ *a pre všetky* $t \in T$ *platí:*

$$[\forall p \in \cdot t : M(p) \geq W(p,t)] \rightarrow [\forall p \in t\cdot : M(p) \leq K(p) - W(t,p)]$$

Definícia nám hovorí, že ak chceme pri bezkontaktnej sieti zistiť, či je prechod spustiteľný pri danom označkovaní, potom nám stačí skontrolovať iba jeho vstupné miesta. Inak povedané, stačí sa zaoberať iba vstupnými podmienkami a nemusíme pri spúšťaní prechodu dopredu kontrolovať, či jeho spustenie nespôsobí narušenie kapacitných obmedzení siete, pretože to pri bezkontaktnej sieti nie je možné.

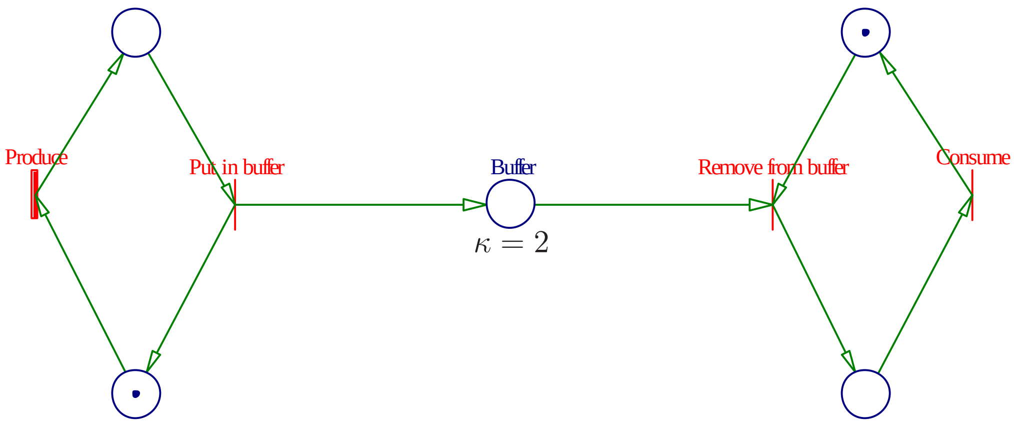
Teraz popíšeme postup, ako možno ku každej Petriho sieti zostrojiť bezkontaktnú sieť s rovnakým správaním. Takáto sieť sa nazýva **komplementárna**.

Algoritmus 4.3.2 (Komplementácia Petriho siete) *Majme danú Petriho sieť* $PN = (P, T, F, W, K, M_0)$*. Komplementárnu sieť* $PN' = (P', T', F', W', K', M'_0)$ *zostrojíme tak, že rozšírime sieť* PN *nasledovne:*

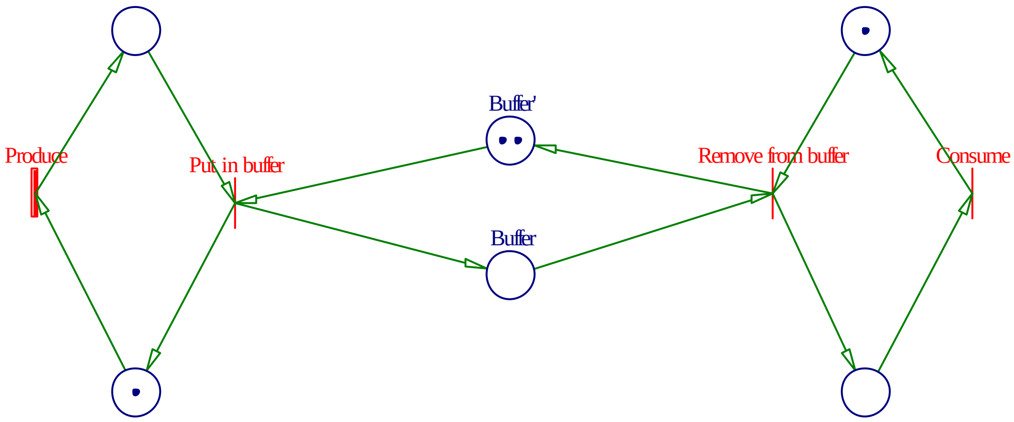
1. *Pre každé* $p \in P$ *s konečnou kapacitou vytvoríme nové miesto* p'
2. *Pre všetky hrany* (p, t) *resp.* (t, p) *z* F *vytvoríme nové hrany* (t, p') *resp.* (p', t) *a položíme* $W'(t, p') = W(p, t)$ *resp.* $W'(p', t) = W(t, p)$
3. *Položíme* $K'(p') = K(p)$ (tento krok je čisto formálny)
4. *Položíme* $M'_0(p') = K(p) - M_0(p)$

Výsledná sieť je zrejme bezkontaktná, pretože pre všetky dosiahnuteľné označkovania M platí $M(p) + M(p') = K(p)$. Navyše skonštruovaná sieť má rovnaké správanie ako pôvodná v tom zmysle, že ak uvažujeme reštrikciu označkovania M' na množinu P, potom toto označkovanie je rovnaké ako označkovanie M siete PN pre všetky dosiahnuteľné označkovania M. Tieto vlastnosti nebudeme formálne dokazovať, pretože sú ľahko nahliadnuteľné priamo z definície.

Z vlastností označkovania siete PN' vyplýva, že v tejto sieti môžeme všetky kapacity položiť rovné ∞ a správanie siete sa nezmení. Vďaka tomuto faktu môžeme v prípade potreby abstrahovať od kapacitných obmedzení, čo nám môže uľahčiť najmä teoretické úvahy. Naopak pri praktickom modelovaní skutočnosti môže byť jednoduchšie kapacitné obmedzenia používať, pretože umožňujú explicitne zachytiť vlastnosti systému, ktoré by sme inak museli zahrnúť do štruktúry Petriho siete. Ilustrujeme si to na jednoduchom príklade modelu producenta a konzumenta s konečným bufferom.



Obr. 4.5: Model producent-konzument s explicitne obmedzenou kapacitou buffera



Obr. 4.6: Model producent-konzument s obmedzením kapacity buffera pomocou komplementárneho miesta

Kapitola 5

Vlastnosti Petriho sietí

V tejto kapitole definujeme niektoré zaujímavé vlastnosti Petriho sietí, ktoré sa ďalej pokúsime analyzovať. Pre celú nasledujúcu kapitolu predpokladajme, že máme danú Petriho sieť $PN = (P, T, F, W, K, M_0)$, čím sa vyhneme únavnému opakovaniu tejto formulácie v každej definícií.

5.1 Bezpečnosť

Definícia 5.1.1 *Miesto $p \in P$ nazývame bezpečné, ak platí:*

$$\forall M \in [M_0\rangle : M(p) \leq 1$$

Ak je každé miesto siete PN bezpečné, potom PN nazývame bezpečnou sieťou.

Bezpečnosť siete zaručuje, že počet značiek v žiadnom mieste siete neprevýši hodnotu 1. Bezpečnosť sa požaduje napríklad pri modelovaní logických obvodov, kde miesta predstavujú klopné obvody, prípadne ak všetky miesta siete modelujú logické podmienky.

S bezpečnosťou siete súvisí, či sieť spĺňa nasledujúce podmienky:

1. $\forall p \in P : M_0(p) \leq 1$
2. $\forall f \in F : W(f) \leq 1$

Ľahko sa nahliadne, že tieto podmienky niesú postačujúce pre bezpečnosť siete. Sú však zaujímavé preto, že sieť komplementárna k sieti spĺňajúcej tieto podmienky už bezpečnou je. Inými slovami, zo siete spĺňajúcej uvedené podmienky vieme algoritmicky zostrojiť bezpečnú sieť s rovnakým chovaním.

Bezpečnosť siete je pomerne špecifickou vlastnosťou. Preto nie je prekvapivé, že je špeciálnym prípadom všeobecnejšej vlastnosti nazývanej ohraničenosť siete.

5.2 Ohraničenosť

Definícia 5.2.1 *Miesto $p \in P$ nazývame k-ohraničené ak platí:*

$$(\exists k \in \mathcal{N})(\forall M \in [M_0\rangle)(M(p) \leq k)$$

Miesto sa nazýva ohraničné, ak je ohraničené pre niektoré $k \in \mathcal{N}$. Sieť PN sa nazýva ohraničená, ak je každé jej miesto ohraničené.

Ohraničenosť siete sa vyžaduje napríklad pri modelovaní konečných zdrojov (napríklad konečná kapacita vyrovnávacej pamäti alebo konečná veľkosť haldy, konečná dĺžka fronty atď).

5.3 Konzervatívnosť

Definícia 5.3.1 *Petriho sieť PN nazývame striktne konzervatívnou ak platí:*

$$\forall M \in [M_0\rangle : \sum_{p \in P} M(p) = \sum_{p \in P} M_0(p)$$

Striktná konzervatívnosť je pomerne silná vlastnoť a ľahko možno nahliadnuť, že v konzervatívnej sieti musí platiť podmienka

$$\forall t \in T : \sum_{p \in \cdot t} W(p,t) = \sum_{p \in t \cdot} W(t,p)$$

pretože inak by sa spustením prechodu t zmenil počet značiek v sieti. V sieti sú ale často miesta, ktorých sa podmienka konzervácie prostriedkov netýka. Nielen ale aj kvôli ošetreniu tohoto prípadu sa vlastnosť konzervatívnosti formuluje všeobecnejšie.

Definícia 5.3.2 *Nech $v : P \to \mathcal{N}$ je vektor. Sieť PN nazývame konzervatívnou vzhľadom k váhovému vektoru v, ak platí:*

$$\forall M \in [M_0\rangle : \sum_{p \in P} v(p)M(p) = \sum_{p \in P} v(p)M_0(p)$$

Petriho sieť PN nazývame konzervatívnou, ak je konzervatívnou vzhľadom k nejakému nezápornému váhovému vektoru v, ktorý má aspoň jednu zložku kladnú.

Konzervatívnosť siete sa vyžaduje napríklad pri modelovaní prideľovania daných prostriedkov. Nechceme aby prostriedky (reprezentované značkami) vznikali a zanikali, ale chceme aby ich počet bol konštantý (striktná konzervatívnosť) alebo zostal zachovaný v širšom zmysle (konzervatívnosť).

5.4 Živosť

Živosť siete patrí z praktického hľadiska medzi najdôležitejšie vlastnosti. Súvisí s problémom zablokovania modelovaného systému známym ako *deadlock*.

Definícia 5.4.1 *Označkovanie $M \in [M_0\rangle$ sa nazýva živé, ak pre všetky $t \in T$ existuje označkovanie $M' \in [M\rangle$ také, že prechod t je M'-spustiteľný. Sieť sa nazýva živou, ak sú všetky označkovania z $[M_0\rangle$ živé.*

Živosť vieme definovať aj pre prechody siete, pričom rozlišujeme niekoľko hladín živosti.

Definícia 5.4.2 *Živosť pre prechod Petriho siete definujeme takto:*

1. *Prechod $t \in T$ je živý na hladine 0, ak nie je spustiteľný pri žiadnom označkovaní z $[M_0\rangle$.*

2. *Prechod $t \in T$ je živý na hladine 1, ak existuje také $M \in [M_0\rangle$, že t je M-spustiteľný.*

3. *Prechod $t \in T$ je živý na hladine 2, ak pre každé $n \in \mathcal{N}$ existuje postupnosť spúšťania prechodov taká, že sa v nej t vyskytuje aspoň n krát.*

4. *Prechod $t \in T$ je živý na hladine 3, ak existuje postupnosť spúšťania prechodov taká, že t sa v nej vyskytuje nekonečne veľa krát.*

5. *Prechod $t \in T$ je živý na hladine 4, ak pre každé $M \in [M_0\rangle$ existuje označkovanie M' také, že $M' \in [M\rangle$ a prechod t je M'-spustiteľný.*

Petriho sieť je živá na hladine $h \in \{0, 1, 2, 3, 4\}$ *ak každý jej prechod je živý na hladine* h.

Poznamenávame, že *živá Petriho sieť* podľa prvej definície zodpovedá *Petriho sieti živej na hladine 4* podľa druhej definície.

5.5 Dosiahnuteľnosť označkovania

Všetky doposiaľ uvedené vlastnosti súviseli s dosiahnuteľnými označkovaniami Petriho siete. Preto kľúčovým problémom pre analýzu Petriho sietí je *problém dosiahnuteľnosti označkovania*, teda zistiť, či pre dané označkovanie M platí $M \in [M_0\rangle$.

5.6 Pokrytie

Väčšinou nie sú dôležité konkrétne hodnoty označkovania, ale je podstatné zistiť, či je možné dosiahnuť označkovanie, ktorého zložky sú väčšie ako nejaké zadané hodnoty. Tento problém sa nazýva *problém pokrytia*. Práve prostredníctvom riešenia tohoto problému budeme v nasledujúcej kapitole riešiť aj ďalšie problémy.

Kapitola 6

Analýza Petriho sietí

6.1 Graf pokrytia

Popíšeme si metódu analýzy Petriho siete vychádzajúcu z grafovej reprezentácie stavového priestoru Petriho siete, nazývanej graf pokrytia.

Algoritmus 6.1.1 (Konštrukcia grafu pokrytia) *Vrcholy rozdelíme do štyroch skupín:* ***čelné, koncové, duplikované a vnútorné****. Každému vrcholu priradíme označkovanie $M_x : P \to \mathcal{N} \cup \{\infty\}$. Čelné vrcholy budú tie, ktoré bude algoritmus spracovávať a tým ich premení na koncové, duplikované alebo vnútorné. Algoritmus začne od koreňa stromu, ktorý má priradené počiatočné značenie M_0 a na začiatku je jediným čelným vrcholom. Algoritmus pracuje kým existujú čelné vrcholy takto:*

Nech x je práve spracovávaný vrchol s ohodnotením M_x.

1. *Ak existuje iný vrchol y, ktorý nie je čelný a je ohodnotený rovnakým označkovaním ako vrchol x, potom x sa stáva duplikovaným vrcholom.*

2. *Ak pre označkovanie M_x neexistuje prechod $t \in T$, ktorý by bol M_x-spustiteľný, potom x je koncový vrchol stromu.*

3. *Pre prechod t, ktorý je M_x-spustiteľný, vytvoríme nový vrchol z. Ohodnotenie M_z definujeme takto:*

   Pre všetky $p \in P$:

   (a) Ak $M_x(p) = \infty$, potom $M_z(p) = \infty$

(b) Ak na ceste z koreňa do vrcholu x leží vrchol y taký, že $M_y \leq \delta(M_x, t)$ a pre niektoré $p \in P$ platí $M_y(p) < \delta(M_x, t)(p)$, potom $M_z(p) = \infty$

(c) Inak $M_z(p) = \delta(M_x, t)(p)$

(d) Do grafu pridáme hranu (x, z) a označíme ju prechodom t

(e) Vrchol z pridáme do množiny čelných vrcholov

Poznamenávame, že takto zostrojený graf pokrytia je vždy konečný. Dôkaz vyžaduje viacero pomocných tvrdení a možno ho nájsť v [8].

6.2 Analýza pomocou grafu pokrytia

Ukážeme si, ako pomocou grafu pokrytia vieme riešiť definované problémy. Uvidíme aj, že využitie grafu pokrytia má niektoré zásadné obmedzenia.

6.2.1 Bezpečnosť a ohraničenosť

Keď pre danú Petriho sieť zostrojíme graf pokrytia, potom už vieme veľmi jednoducho zistiť, či je sieť bezpečná alebo ohraničená. Zrejme sieť nie je ohraničená práve vtedy, keď niektorý z vrcholov grafu pokrytia obsahuje symbol ∞. Ak graf pokrytia neobsahuje ∞, potom je sieť k-ohraničená. Hodnota k je rovná najväčšiemu číslu v strome. Ak $k = 1$, potom je sieť bezpečná.

6.2.2 Konzervatívnosť

S využitím grafu pokrytia vieme efektívne riešiť problém konzervatívnosti Petriho siete. Ukážeme, ako vieme k danej konzervatívnej Petriho sieti PN nájsť zodpovedajúci váhový vektor v.

Nech $P = \{p_1, p_2, \ldots, p_n\}$ je množina miest siete PN, množina vrcholov grafu pokrytia má hodnoty $\{M_1, M_2 \ldots, M_m\}$ zodpovedajúce označkovaniam, pokrývajúcim všetky dosiahnuteľné označkovania. Predpokladáme, že Petriho sieť PN je konzervatívna, teda že platí

$$\forall M \in [M_0\rangle : \sum_{p \in P} M(p)v(p) = const$$

Označkovania siete môžeme zrejme uvažovať aj ako vektory rozmeru $|P|$. Potom pre konzervatívnu sieť musí existovať riešenie nasledujúcej sústavy lineárnych rovníc a nerovností pre $n+1$ neznámych $v(p_i)$ v obore kladných racionálnych čísiel:

$vM_i = const$ pre $i = 1, 2, \ldots, m$

$v(p_j) > 0$ pre $j = 1, 2, \ldots, n$

Na riešenie tejto sústavy možno použiť metódu lineárneho programovania. Ak existuje riešenie na obore kladných racionálnych čísiel, potom výsledné celočíselné váhy získame prevedením výsledkov na spoločného menovateľa. Ak existuje riešenie v tvare $v(p_i) = 1$ pre $i = 1, 2, \ldots, n$), potom je sieť striktne konzervatívna. V prípade, že riešenie neexistuje, sieť nie je konzervatívna.

Ak chceme riešiť problém konzervatívnosti pre určitú podsieť, môžeme niektoré miesta siete z rovníc vylúčiť tak, že položíme $v(p_i) = 0$ pre zvolené indexy i.

6.2.3 Dosiahnuteľnoť označkovania a živosť

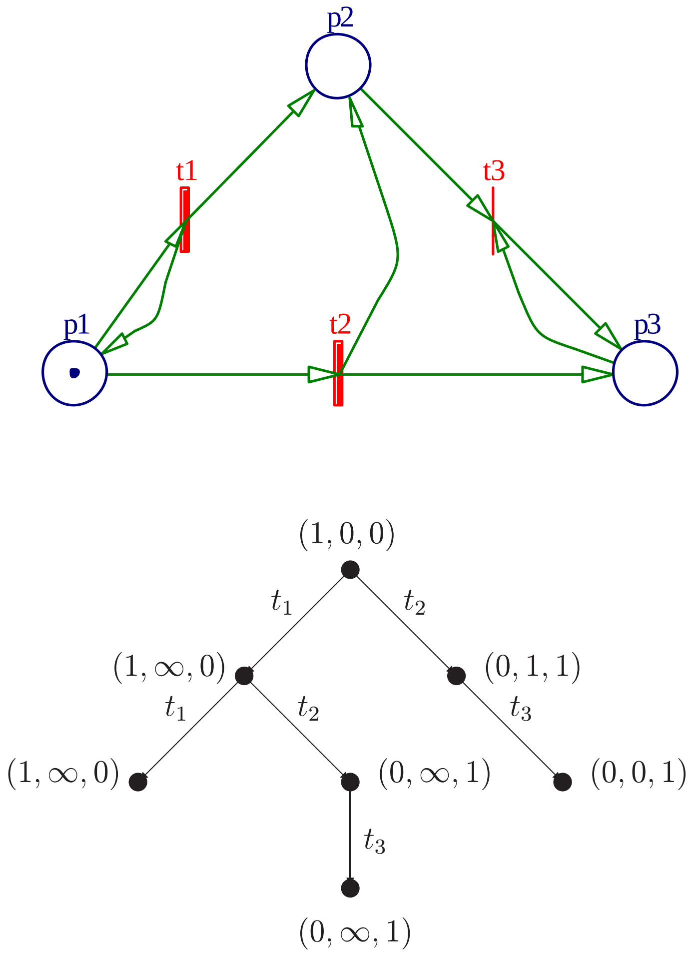
Dosiahnuteľnosť značkovania a živosť siete sú problémy, pre riešenie ktorých graf pokrytia nie je dostatočne jemným nástrojom. Obrázok (6.2) ukazuje dve Petriho siete s rovnakými grafmi pokrytia, ale rôznymi množinami dosiahnuteľných označkovaní. Uvedené siete nie sú ekvivalentné ani podľa kritéria živosti.

V konkrétnych prípadoch ale môže byť graf pokrytia predsa len užitočný. Ak graf pokrytia obsahuje vrchol bez potomkov, potom sieť nemôže byť živá. Ak označkovanie nie je pokryté žiadnym vrcholom grafu, potom nie je dosiahnuteľné a ak označkovanie zodpovedá niektorému vrcholu grafu, potom je určite dosiahnuteľné.

6.3 Výpočtové obmedzenia

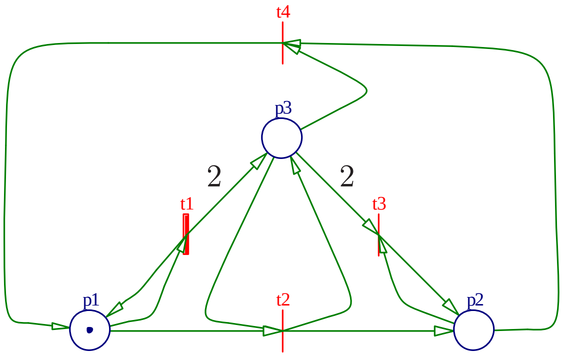
Graf pokrytia je dobrý teoretický nástroj na analýzu Petriho sietí, jeho praktická využiteľnosť je ale obmedzená, vzhľadom na jeho výpočtovú zložitosť. Je známy nasledujúci teoretický výsledok:

Veta 6.3.1 *Existuje postupnosť Petriho sietí $N_1, N_2, \ldots,$ ktorých veľkosť (celkový počet vrcholov, hrán a značiek siete) rastie lineárne taká, že zodpovedajúce grafy pokrytia $G_1, G_2, \ldots$ rastú rýchlejšie ako ľubovolná primitívne rekurzívna funkcia.*

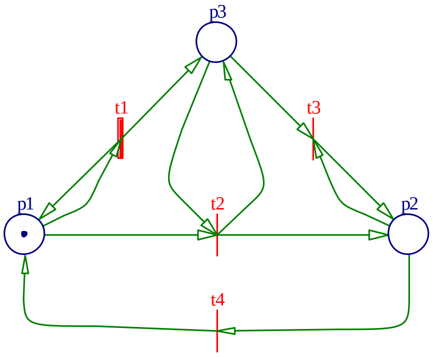


Obr. 6.1: Petriho sieť a zodpovedajúci graf pokrytia

N_1 :



N_2 :



Obr. 6.2: Sieť N_2 je živá, pri sieti N_1 si všimnite postupnosť spúšťania prechodov t_1, t_2, t_3.

Záver

Čitateľ sa mal možnosť oboznámiť s dvoma odlišnými modelovacími nástrojmi. Teória kapacitných sietí bola spracovaná tématicky pomerne vyčerpávajúco a čitateľ mal možnosť získať ucelený prehľad o tom, aké sú možnosti využitia základných konceptov v zovšeobecnených podmienkach. Ukázalo sa, že základná teória o tokoch v kapacitných sieťach je silným nástrojom aj v podstatne všeobecnejších podmienkach než sú tie, pre ktoré bola pôvodne vypracovaná. Ak čitateľa zaujala problematika tokov v kapacitných sieťach, môže ďalej pokračovať štúdiom cirkulácií na grafoch. Ide o všeobecnejšiu tému, ktorá v sebe zahŕňa všetky v tejto práci uvažované zovšeobecnenia ako špeciálne prípady. Väčšia všeobecnosť teórie cirkulácií už ale vyžaduje aj odlišný spôsob riešenia problémov.

Petriho siete boli čitateľovi prezentované ako silný modelovací nástroj, no vonkoncom nebola problematika spracovaná v celej svojej šírke. Uvedený model Petriho sietí a prezentovaná metóda na ich analýzu tvorí iba odrazový mostík pre pokročilejšie štúdium. Čitateľ má neobyčajne široké možnosti ďalšieho štúdia. Predovšetkým je to analýza vlastností Petriho sietí pomocou skúmania jej štrukturálných vlastností, známych ako P-invariant a T-invariant. Pre teoreticky zameraného čitateľa so záujmom o formálne jazyky, bude poučné študovať jazyky akceptované Petriho sieťami. Čitateľ, ktorý chce Petriho siete využiť prakticky na modelovanie reálnych systémov veľkých rozmerov, môže mať záujem o štúdium Petriho sietí vyšších rádov, Petriho sietí s časom, farebných Petriho sietí, fuzzy Petriho sietí ...

Dúfam, že vo Vás moja diplomová práca vzbudila záujem o rozoberané problémy a bude podnetom pre ďalšie štúdium, či už v oblasti kapacitných sietí alebo Petriho sietí.

Literatúra

[1] D. Jungnickel: **Graphs, Networks and Algorithms**, Springer, 2000

[2] J. L. Gross, J. Yellen: **Graph Theory and Its Applications**, 1998

[3] F. Gliviak, P. Vadkerti: **Sieťová analýza a manažment projektov**, FPV UCM Trnava, 2001

[4] F. Gliviak, P. Sakál: **Network Analysis and Project Management - An Overview**, preprint, 2004

[5] L. Štekauerová: **Matematické metódy sieťovej analýzy**, diplomová práca, FMFI UK Bratislava, 2001

[6] M. Češka: **Petriho sítě**, Akademické nakladatelství CERM Brno, 1994

[7] B. Hrúz, L. Mrafko: **Modelovanie a riadenie diskrétnych udalostných dynamických systémov s využitím Petriho sietí a iných nástrojov**, STU Bratislava, 2003

[8] J. Peterson: **Petri Net Theory and the Modelling of Systems**, Prentice Hall, Englewood Cliffs, New Jersey, 1981