VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

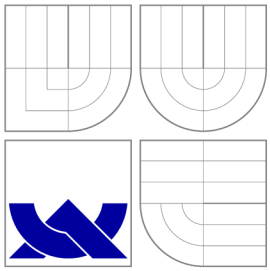
GRAFICKÉ ZNÁZORNĚNÍ SMĚROVACÍCH ALGORITMŮ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

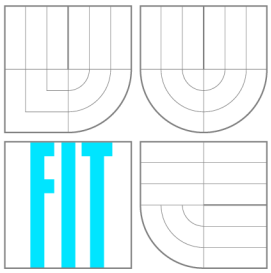
AUTOR PRÁCE
AUTHOR

OSKÁR HANZÉLY

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

GRAFICKÉ ZNÁZORNĚNÍ SMĚROVACÍCH ALGORITMŮ

GRAPHICAL VIZUALIZATION OF ROUTING ALGORITHMS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

OSKÁR HANZÉLY

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. JIŘÍ JAROŠ

BRNO 2008

Abstrakt

Táto práca popisuje princípy smerovania počítačových sietí, prehľad topológií a prehľad používaných smerovacích algoritmov. Obsahuje detailný popis vytvoreného programu pre grafické zobrazenie sieťových topológií a smerovania. Zahrňuje proces transformácie niektorých schém a vizualizáciu krokov smerovania v podobe animácie. Stručne sú popísané možnosti ďalšieho vývoja a použitia.

Kľúčové slová

topológie sietí, smerovací algoritmus, vizualizácie

Abstract

This thesis deals with routing processes of computer networks, overview of netwrok topologies and commonly used routing algorythms. It includes detailed description of created application to graphycal dysplay of networks schemes and routing. It contains transformation process of some schemes and the visualisation of all routing steps like an animation. The potency of future development and usage is commented briefly.

Keywords

network topology, routing algorythm, visualization

Citácia

Oskár Hanzély: Grafické znázornění směrovacích algoritmů, bakalářská práce, Brno, FIT VUT v Brně, 2008

Grafické znázornění směrovacích algoritmů

Prehlásenie

Prehlasujem, že som svoju bakalársku prácu vypracoval samostatne pod vedením vedúceho bakalárskej práce a s použitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry.

.....
Oskár Hanzély
14. mája 2008

Obsah

Kapitola 1

Úvod

Myšlienka smerovania siaha asi do 50tych rokov minulého storočia, keď počítačová sieť bola ešte len futuristickou myšlienkou. Dnes je to ale jedna a najdôležitejších funkcií sietí. Do internetu sa každým dňom pripájajú ďalšie pracovné stanice a zložitosť jeho topológie narastá geometrickou radou. Pre spoľahlivé a hlavne rýchle doručenie dát, či už v rámci firemných LAN sietí, alebo na druhý koniec planéty je potrebné vybrať tu najefektívnejšiu cestu kadiaľ informácia poputuje. Pri tak rozsiahlých možnostiach výberu ciest je smerovanie doslova „vedou“, využívajúcou zložité matematické vzťahy a sofistikované postupy tzv. *smerovacie algoritmy*.

Námetom tejto odbornej práce je grafické znázornenie používaných algoritmov. Aplikácia, ktorá podopiera teoretické základy práce má za úlohu zobrazovať na určených topológiách sietí priebeh smerovacieho algoritmu. Uvedený program môže slúžiť ako demonštračný nástroj vo výučbe a utvárať tak vizuálnu predstavu o ceste informácie poslanej po linkách sieti. Je výsledkom pochopenia problematiky smerovania a základných princípov počas semestrálneho projektu. Pri štúdiu problematiky som prevažne čerpal z knihy *Směrování v sítích IP* (Sportack, 2004) [7], ako aj z iných zdrojov spomenutých v texte práce i zozname literatúry. Teoretické informácie z oblasti matematiky mi pomohli doplniť skirpta [6].

V rozsiahlej druhej kapitole popisujem sieťovú architektúru, jej históriu, význam a techniky fungovania s detailnejším zameraním sa na smerovanie. Rozbor niekoľkých druhov smerovacích protokolov ukazuje rozdiely a hlavné rysy konkrétnych algoritmov. V podkapitole o smerovaní je vytvorený i detailný rozbor smerovacích metrík a ich klasifikácia. Posledný odstavec je akousi úvahou o možnostiach a vývoji týchto techník v budúcnosti. Tretia kapitola predkladá návrh koncepcie vizualizácie v programe a popis niektorých použitých postupov pri výbere najjednoduchšieho a užívateľsky príjemného ovládania celej aplikácie. Obrazne je ukázaný mechanizmus fungovania programu, a zjednodušovanie zložitého mechanizmu smerovania do názorne a ľahko pochopiteľnej formy. Obsahuje taktiež rozbor vývoja i navrhnutú hierarchiu programu. Samotnej implementácii a popisu zdrojového kódu, praktickej časti projektu, je venovaná štvrtá kapitola. V závere práce zhrňujem dosiahnuté poznatky, možnosti ďalšieho vývoja i eventuálne praktické uplatnenie programu. Dosiahnuté výsledky krátko konfrontujem s komerčným softwarom na trhu. V Prílohe nájdete ukážku spomenutej aplikácie v animačnom móde.

Kapitola 2

Počítačová sieť

2.1 Sieťová architektúra

Počítačová sieť je vo všeobecnosti systém prostriedkov, ktoré zaobstarávajú spojenie a výmenu informácií medzi počítačmi. Zabezpečuje tak možnosť posielania dát medzi účastníkmi na rôznych miestach, v rôznom čase s použitím čo najefektívnejších prostriedkov z hľadiska dostupnosti a reakcie. Sieťová architektúra predstavuje práve súhrn činností umožňujúci prenos týchto dát.

2.1.1 História

História sietí siaha do prvej polovice 60tych rokov minulého storočia, keď sa dá hovoriť o prvých pokusoch spojenia počítačov. Bolo to prepojenie niekoľkých zariadení s dominantným prvkom združujúcim všetky okolité stanice. Táto technológia sa nazýva *mainframe*. Dnes pre pojem „hlavného rámu“ používame pomenovanie *sálový počítač*.

2.1.2 Mainframe

Mainframy ale nie sú len zastarané technológie minulosti. Stále viac sa používajú vo veľkých strediskách národných spoločností, bankovom sektore, univerzitných výpočtových strediskách, všade kde je nutné spracovávať veľké objemy dát s vysokou mierou bezpečnosti a stability. I keď túto platformu majú v ponuke rôzne firmy zaoberajúce sa poskytovaním služieb v IT, celosvetovou jednotkou v outsourcingu[1] mainframových produktov je spoločnosť IBM, s dostupnosťou systému 24 hodín denne 7 dní v týždni.
Výhod *mainframu* je značné množstvo, spomeniem len najväčšie:

- dostupnosť (teoretická dostupnosť systému je až 99,99 %)
- flexibilita a škálovateľnosť
- parallel sysplex (spojenie až 32 počítačových systémov do jedného celku za účelom maximálneho výkonu)

[1] Outside Resource Using - delegácia činností a zodpovednosti na špecializovanú spoločnosť

Viac o tejto platforme a jej výhodách je možné nájsť v produktovom portfóliu IBM [3] .

2.1.3 Dnešné siete

Siete sa od prvého diaľkového prenosu z roku 1973 značne zmenili. Vtedajšia záplava rôznych protokolov, keď každá väčšia firma vyvíjala vlastný spôsob komunikácie, sa pretvorila do škály celosvetovo používaných, normou definovaných, štandardov (IETF, RFC2026 [1]). Rozsiahlejší prístup k informáciám podnietil prepojenie sveta, vytváranie veľkých celkov, lokálnych, regionálnych, národných, ktoré vytvárajú celosvetovú „sieť sietí" – Internet. Z pohľadu hardwaru ju tvorí milióny koncových počítačov, *pasívne komponenty siete* ako komunikačné linky (optické vlákna, drôtené spoje, satelity), *aktívne komponenty* ako smerovače (routery) či opakovače (huby), a ďalšie sieťové zariadenia. Je to ale predovšetkým komunikačná infraštruktúra umožňujúca komunikáciu distribuovaných aplikácií (web, email, zdieľanie súborov,...).

2.1.4 Vrstvový model

Riadenie komunikácie v sieťovej architektúre je zložitý systém činností, preto bol rozdelený na skupiny tzv. *sieťové vrstvy*. Každá vrstva má špecifikované funkcie tvoriace časť riadenia komunikácie, ako aj definovaný spôsob komunikácie so susednou nižšou a vyššou vrstvou. Nižšia vrstva vždy poskytuje služby vyššej. Komunikácia prebieha stále na rovnocenných (angl. *peer*) vrstvách na základe istých pravidiel, procedúr a formátov, ktoré tvoria *komunikačný protokol*. Organizáciou ISO bol preto prijatý referenčný ISO/OSI[2] model sieťovej architektúry, ktorý špecifikuje 7 základných vrstiev. Väčšina aktuálne používaných protokolov vo svete má už vlastný vrstvový model, kde sa málo dodržuje pôvodný OSI a základný systém vrstiev sa prispôsobuje stále novým štandardom. Dnešný model internetu najbližšie popisuje TCP/IP model s 5 vrstvami, ktorý podporuje rozsiahle množstvo OS a je dominantný v rámci WAN.

2.2 Smerovanie

IP (Internet Protocol), je základný protokol sieťovej vrstvy (3. vrstva Internetového modelu) a na základe IP adries vysiela tzv. IP *datagramy* (samostatné dátové jednotky obsahujúce údaje o adresátovi i prijímateľovi). Na rozdiel od lokálnych sietí LAN, kde jednotlivé pracovné stanice sú usporiadané v jednom segmente a stačí počítačom pri vzájomnej komunikácii posielať priamo dátové rámce, je posielanie informácií v Internete omnoho zložitejšie. Medzi vysielačom a príjemcom dát je množstvo hardwarových zariadení a stovky rôznych možností ciest, preto by nebolo reálne možné rozposlať informáciu všetkým počítačom a čakať, až dáta dorazia na určené miesto. Jediným možným riešením, je presne identifikovať cestu od cieľa. Práve hľadanie tej najvhodnejšej cesty v spleti spojení obrovskej svetovej „pavučiny" umožňuje smerovanie (angl. *routing*), zrejme tá najkomplikovanejšia a zároveň najdôležitejšia funkcia počítačových sietí.

[2]Open Systems Interconnection Reference Model (štandard ISO 7498)

2.2.1 Smerovač

Na rozdiel od väčšiny sieťových prvkov je smerovač (angl. *router*) aktívny sieťový prvok, ktorého prvoradou prácou je zabezpečiť preposielanie IP datagramov na sieťovej vrstve. Toto zariadenie pracuje s dvojicou protokolov (smerovateľné a smerovacie), oba pôsobiace na tretej vrstve.

Základnou využívanou štruktúrou je tzv. smerovacia tabuľka (*routing table*). Predstavuje akúsi tabuľku ukazateľov, so záznamom typu: cieľová adresa –> akcia, ktorá sa má vykonať. Pre každý[3] prichádzajúci paket (datagram) sa nájde v tabuľke jemu odpovedajúci prefix a vykoná sa príslušná akcia, či už je to priame doručenie cieľu, alebo preposielanie na ďalší smerovač, ak je cieľ vzdialený.

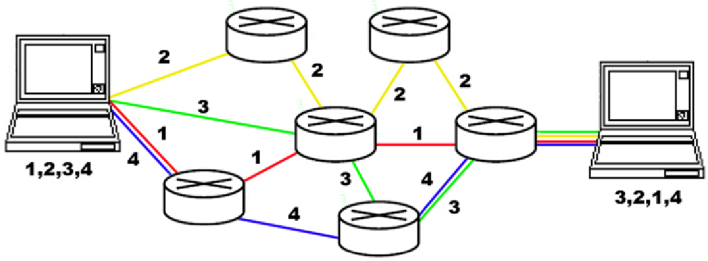
Smerovacie protokoly sú využívané na vnútornú komunikáciu medzi smerovačmi, prenos sieťových parametrov a poskytujú všetky potrebné informácie o sieti. Implementujú v sebe smerovacie stratégie a smerovacie algoritmy, podľa nich dokáže každý smerovač stanoviť dostupné cesty a po týchto cestách potom posielať pakety smerovateľného protokolu. V prípade, že cieľ nie je priamo dostupný, ale v ceste stoja ďalšie aktívne sieťové prvky, prostredníctvom týchto protokolov sa to daný vysielajúci smerovač dozvie. Zistí teda nielen samotnú optimálnu cestu, ale rozpozná i topológiu internetovej sieti, či adresy sietí a hostiteľov. Protokoly tiež udržujú a aktualizujú smerovacie tabuľky. Najznámejšie smerovacie protokoly: RIP (Routing Information protocol), IGRP (Interior Gateway Routing Protocol), OSPF (Open Shortest Path First).

Smerovateľné protokoly zabaľujú dáta a užívateľské informácie do paketov a zabezpečujú ich transport k príslušnému cieľu. Príklad takéhoto protokolu je datagramový protokol IP (Intenret Protocol). Jeho nepríjemná vlastnosť je, že nezaručuje zachovanie poradia vyslaných paketov, a dokonca ani ich úspešné doručenie. Vo svete Internetu je to ale jeden z dvojice (TCP/IP) najčastejšie používaných riešení.

2.2.2 Smerovanie v sieti

Smerovateľné protokoly môžeme podľa spojenia rozdeliť na spojovo-orientované (služba TCP) a nespojovo-orientované (tzv. bezspojové; služba IP). Samotné smerovanie sa v sieťach s použitím oboch protokolov líši. V spojovo-orientovanej sieti smerovač hľadá ďalší uzol len vtedy, ak sa vytvára nový virtuálny kanál. Na druhej strane stoji bezspojová sieť, kde smerovač prechádza routovaciu tabuľku pri každom prijatí paketu, a prevádza výber cesty.

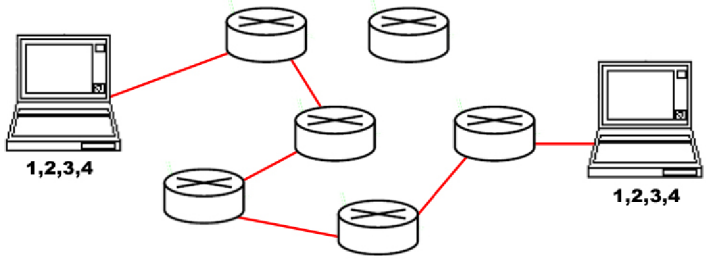
[3]pozn. platí to len pre nespojovo-orientované služby ako IP.



Obrázok 2.1: Znázornenie prepínania paketov v bezspojovej sieti

Pri použití nespojovo-orientovaných protokolov sa pred posielaním paketov nevytvára relácia, tzn. nie je vopred známy cieľ a smerovanie môže prebiehať vždy len k ďalšiemu uzlu. Nie je zaručené poradie doručenia paketov, cesta sa vyberá a mení v priebehu transportu sieťovými zariadeniami a oneskorenie spôsobené nesprávnym výberom cesty je v tomto prípade irelevantné. Túto situáciu popisuje obrázok 2.1.

Použitie spojovo-orientovaných protokolov má väčšie výhody. Pri ustanovení spojenia sa vytvorí relácia, ktorá sa udržiava počas celého priebehu komunikácie. Exituje teda viacero nezávislých okruhov pre každý pár (odosielateľ– prijímateľ). Dáta prichádzajú v poradí, akom boli vystavené na sieť, komunikácia je teda bezpečná a cieľové zariadenie sa nemusí zaoberať usporadúvaním prijatých informácií.



Obrázok 2.2: Znázornenie cesty smerovania v spojovo-orientovanej sieti

2.2.3 Beztriedne smerovanie

Bežné smerovanie využíva k identifikácií sietí, podsietí a hostiteľov triedy adries IPv4. V roku 1992 združenie IETF v reakcii na stále sa zvyšujúci počet pripojených organizácií v internete muselo riešiť dosť závažný problém. Niektoré triedy sa využívali menej ako trieda C, celková voľná kapacita nepridelených IP adries sa rapídne zmenšovala a routovacie tabuľky smerovačov neustále narastali, čo spôsobovalo ich enormné zaťaženie. Niektorý skeptici dokonca predpovedali „súdny deň“, ktorý mal nastať v roku 1994 a mal spôsobiť úplne vyčerpanie kapacity internetu. Jedinou možnosťou bolo vytvoriť nový protokol IP, ktorý by mal rozsiahlejšie možnosti a nový spôsob adresovania. Z dlhodobého hľadiska mal uzrieť svetlo sveta protokol IPv6, ale rýchlosť vývoja nebola priamo úmerná spotrebe možností IPv4. Odpoveďou na naliehavý problém bolo zbavenie sa hraníc triedneho adresovania. V septembri 1993 bolo zadefinované *beztriedne smerovanie medzi doménami* (CIDR) v doku-

mentoch[4] RFC [4]. Princíp CIDR zbúral zažité tradície tým, že každá IP adresa nesie so sebou údaj o maske. Smerovače s podporou CIDR získavajú číslo siete podľa počtu bitov uvádzaných za lomkou. Bity reprezentujú masky podsiete s premenlivou dĺžkou. Tým bolo možné využiť prakticky celý adresový priestor predtým výhradne určený pre triedy siete.

2.2.4 Autonómne systémy

Vrchol hierarchie IP sietí, skupiny smerovačov a prislúchajúcich segmentov patriacich pod správu jednej firmy, či organizácie sa definuje ako *autonómny systém* (AS). Vnútorná komunikácia v systéme prebieha za pomoci Interior Gateway Protokolov (IGP), typicky OSPF, a pre smerovanie medzi jednotlivými AS sa používa Border Gateway Protokol (BGP). Internet je tak rozdelený na viacero AS z dôvodu vyťaženosti routovacích tabuliek. Mohli by sme i celosvetovú sieť chápať ako jeden rozsiahly AS, ale routovacie tabuľky by boli nezvládnuteľne veľké. Aby protokol BGP dokázal medzi týmito samostatnými celkami komunikovať, každá autonómna zóna ma pridelené tzv. Autonomus System Number (ASN). Je to 16bitové číslo v dekadickom tvare, môže teda nadobúdať hodnoty 0-65535, pričom prvá a posledná hodnota sú rezervované hodnoty. Čísla prideľuje a spravuje rovnaká organizácia, ako IP adresy napr. IANA. Oficiálny názov AS pridelený pre OSI je *autonómna doména*. V súvislosti s obmedzeným a nedostatkovým počtom čísel, sa pripravuje v budúcnosti prechod na 32bitové číslovanie [5].

2.3 Klasifikácia a metriky smerovania

Smerovače pracujú pri určovaní cesty dvoma základnými spôsobmi. Jedna možnosť je využívať statické (*neadaptatívne*), vopred naprogramované, cesty, alebo dopočítavať kroky dynamicky (*adaptatívne smerovanie*). K adaptatívnemu smerovaniu, musí byť router vybavený možnosťou prijímať informácie o stave a topológi sieti a tieto informácie si vymieňať prostredníctvom niektorého z množstva smerovacích protokolov[5]. Preto statické routery nemôžu nikdy byť zapojené v sieti, kde nie je dobre známa jej topológia a možnosť vopred naprogramovateľných ciest. Pakety totiž posielajú na základe definícií administrátora. Okrem statického smerovania sú k dispozícii i široká škála protokolov dynamického smerovania rozdelených do sekcií na: protokoly s vektorom vzdialeností, protokoly so stavom linky a hybridné protokoly.

Statické smerovanie. Tento typ smerovania je najjednoduchší. Všetky údaje, na základe ktorých smerovač rozosiela pakety, sú naprogramované a pevne v ňom zapísané sieťovým administrátorom. Smerovač ich vysiela na vopred určených portoch, a nepracujú tu žiadne smerovacie algoritmy, nenastáva žiadna výmena informácii medzi smerovačmi v sieti a nie sú použité ani smerovacie protokoly. Toto zapojenie má celý rad výhod i nevýhod. Kladná stránka takéhoto zapojenia je presná a bezpečná cesta (väčšinou len jedna) ku ktorémukoľvek zariadeniu v malej sieti, čím sa stáva sieť bezpečnejšia. Zníženie vnútornej komunikácie smerovačov tiež úmerne znižuje nároky na prenosové pásmo, ktoré môže byť efektívnejšie využité na prenos dát. V neposlednom rade, sú náklady na kúpu smerovača nižšie. I cez tieto využiteľné výhody má statické smerovanie i značné nevýhody, ktoré ho

[4]RFC dokumenty 1517, 1518, 1519 a 1512
[5]viď smerovací protokol str.5

odsúvajú do veľmi úzkeho spektra jeho využitia. V prípade akejkoľvek zmeny v sieti, či dokonca jej kolapsu, je všetka réžia, spojená so zmenou nastavení výlučne so schopnosťami a rýchlosťou človeka (administrátora). Čo značne spomaľuje opätovné „oživenie“ sieťového spojenia. Tento typ je ale niekedy žiadaný i pre väčšie a zložitejšie sieťové topológie. Správna konfigurácia môže i tu posilniť samotnú bezpečnosť.

Smerovanie s vektorom vzdialenosti. V prípade väčších sietí s redundantnými spojeniami, je statické smerovanie nedostatočné. Rýchla zmena sietí WAN by nebola dostatočne reflektovaná zmenami v routovacích tabuľkách a v prípade výpadku niektorého uzlu by mohlo dôjsť k nedostupnosti častí sietí v globálnejšom rozsahu. Preto je na mieste vysvetliť možnosti dynamického smerovania a popísať jeho výhody a nevýhody.

Smerovacie algoritmy na báze vektorovej vzdialenosti, nazývané tiež Bellman-Ford algoritmy, pracujú na princípe predávania smerovacích tabuliek susedom v sieti. Smerovač prepošle obsah svojej routovacej tabuľky najbližšiemu susedovi, a ten k nej pridá svoj *vektor vzdialenosti*, čo je hodnota rozdielu ich vzájomnej „vzdialenosti“. Výslednú aktualizovanú tabuľku prepošle následne ďalej bezprostrednému susedovi, vo všetkých smeroch. Touto vzájomnou výmenou informácii o sebe, sa smerovače postupne dozvedia o veľkosti a tvare siete. I táto varianta smerovania má ale isté nevýhody. Pri určitých haváriách, sa síce nový „obraz“ siete opätovne utvorí, ale z časového hľadiska to nie vždy dostatočne rýchlo. Kým sa smerovače „dohodnú“ (*konvergujú*) na novom tvare a rozložení siete chvíľu to trvá. Je to spôsobené nedostupnosťou niektorých uzlov, a rôznymi nekorešpondujúcimi záznamami v routovacích tabuľkách v okolí poruchy. Preto je výkon siete pred dosiahnutím stavu konvergencie mierne ohrozený.

Schopnosť rýchlosti konvergencie smerovacieho protokolu je jeden z faktorov rozhodujúcich v nasadení pre rozsiahle WAN siete. Častým javom po zmene topológie siete je tzv. kolísanie cesty (*route flapping*) spôsobujúce nestabilitu siete. Je to príznak rýchleho vymieňovania si informácii o stave siete medzi smerovačmi pred zhodou o podobe novej topológie. Typickým zástupcom smerovacieho protokolu s vektorom vzdialenosti je RIP.

Smerovanie so stavom linky. Smerovacie algoritmy so stavom linky, označované tiež ako *protokoly najkratších ciest* sú zložitejšie ako predchádzajúci typ. Uchovávajú si zložitejšiu databázu nielen o presnej topológii siete, ale smerovače si vymieňajú i tzv. *oznámenia o stave linky*. Každý smerovač si z dostupných informácií o stave linky pomocou algoritmu najkratších ciest vytvorí databázu dostupnosti jednotlivých uzlov v sieti a aktualizuje smerovaciu tabuľku. Takto zaisťuje, že dokáže rozpoznať nielen nedostupnosť časti siete, ale i stále nové prírastky v topológii. Výmena oznámení o stave sieti (LSA) sa spúšťa vždy periodicky len pri vzniku udalosti v sieti, čím sa urýchli proces konvergencie (nečaká na vypršanie nastavených časovačov). I cez značnú flexibilitu týchto protokolov nastávajú u nich dva možné problémy.

Behom prvotného rozpoznávania siete zaťažujú prenosové pásmo niekoľkonásobne viac, než je bežné a spomaľujú tak tok dát prenášaných linkami. Toto zaťaženie, je síce len dočasného charakteru, no pre siete s nižšou šírkou prenosového pásma a väčším počtom smerovačov môžu predstavovať citeľné spomalenie.

Druhý problém je v nákladoch vynaložených na smerovače tohto typu, ktoré predstavujú značné zvýšenie oproti smerovačom pracujúcim s vektorom vzdialenosti. Tieto, skôr úvodné, problémy sa dajú ale vyriešiť správnym návrhom a plánovaním siete. Typickým zástupcom týchto protokolov je OSPF, ktorý zabezpečuje lepšiu škálovateľnosť siete a kratšiu dobu

konvergencie, vďaka rozdeleniu siete na tzv. oblasti (angl. *areas*).

Hybridné smerovanie. Hybridné smerovacie protokoly využívajú metriky vektoru vzdialeností uplatňovaním prísnejších pravidiel. V konečnom výsledku konvergujú rýchlejšie a zároveň sa vyhnú réžii spojenej s aktualizáciou stavu liniek. Oproti smerovacím protokolom so stavom linky, nepracujú v pravidelných intervaloch, ale sú riadené udalosťami. Vďaka tomu u nich nie je pozorovateľné zahltenie šírky prenosového pásma pri veľkom počte smerovačov v sieti a zachovávajú miesto pre reálne aplikácie. Vývoj týchto protokolov je v kompetencii jedinej spoločnosti, Cisco Systems, Inc. Vytorili protokol EIGRP (Enhanced Interior Gateway Protocol), ktorý v sebe integruje to najlepšie z protokolov s vektorom vzdialenosti a stavom linky, pričom nemá žiadne významné nevýhody, či obmedzenia.

Necitlivé a deterministické smerovanie. Popis týchto dvoch smerovacích algoritmov som nechal na koniec, pretože nenesú výhody ostatných dynamických smerovacích protokolov, ale svojím správaním sa skôr pripomínajú statické smerovanie (obzvlášť druhý z nich). Pri necitlivom smerovaní vyberajú smerovacie algoritmy cestu zo zdrojovej stanice do cieľovej úplne náhodne. Každý takýto výber je teda odlišný od predchádzajúceho, a i keď sú podmnožinou dynamického smerovania, nezaručujú efektivitu tohto typu. Deterministické smerovanie je veľmi podobné necitlivému. Vyberá ale pri každom smerovaní zo známeho zdroja do cieľového vždy tú istú cestu. Nereflektuje teda aktuálnu záťaž siete. Oba typy smerovania sa zdajú byť najmenej žiadanou voľbou, ale podobne ako statické smerovanie je ich implementácia značne jednoduchá a menej nákladná. Napomáha tým predchádzať tzv. *deadlockom* (situácia, keď smerovač čaká na dokončenie časovačov v sieti, aby mohol odoslať informáciu zvolenou cestou).

2.3.1 Smerovacie metriky

Dvoma kľúčovými faktormi, pri výbere a rozlíšení smerovacích protokolov, by sa dali považovať schopnosť konvergencie smerovacieho protokolu a spôsob vypočítavania ciest. Práve doba konvergencie (doba, za ktorú sa smerovače v sieti zjednotia na novej podobe sieti po jej zmene) je závislá na efektivite výpočtu ciest. Smerovacie metriky sú dôležitým mechanizmom každého smerovacieho protokolu. Jednoduché protokoly implementujú jednu či dva rôzne metriky, zložitejšie zvládajú i päť. S najprimitívnejšou metrikou sa stretneme u spomínaných protokolov s vektorom vzdialeností a tou je „vzdialenosť“. Táto vzdialenosť neudáva žiadnu fyzickú vzdialenosť smerovačov, či dĺžku kabeláže, meria sa v tzv. *hopoch*[6]. Protokoly so stavom linky vypočítavajú cestu už na základe niekoľkých metrík:

- Komunikačné zaťaženie siete
- Dostupná šírka pásma
- Oneskorenie pri šírení signálu
- Sieťové náklady na dané spojenie[7]

Tieto metriky neponúkajú stabilné hodnoty, kolísajú spolu so zaťažením siete v čase a preto optimálne podporujú rozhodovanie pri smerovaní. Smerovače spracovávajú vždy najaktuálnejšie údaje. Naproti tomu u statických metrík máme možnosti ich hodnotu prispôsobiť

[6] jeden *hop* je prenesenie informácie zo smerovača na ďalší, susedný smerovač.

[7] táto metrika je skôr považovaná, za orientačnú hodnotu určenú odhadom, než presne merateľnú hodnotu

a sú stabilnejšie. Rozhodovanie smerovačov, pri riadení sa nimi, nemusí ale zodpovedať aktuálnemu stavu na sieti a teda nie je tak efektívne ako pri použití dynamických metrík.

2.4 Smerovanie v budúcnosti

Smerovanie v sieťach je postavené na riešení problémov z druhej polovice minulého storočia. Vývoj technológií používaných dodnes sa datuje viac ako 20 rokov dozadu a je hardwarovo orientovaný. Aká je teda budúcnosť hardwarového smerovania, či dokonca smerovania vôbec? Technologický pokrok v oblasti informačných technológií, predkladá stále rafinovanejšie a menej nákladné riešenia práce, ktorú robia dnešné smerovače.

Jedným s nich je vývoj smerovacieho softwaru, ktorým by sa odstránila závislosť na hardware a použitých smerovacích protokoloch. Bežný počítač by zvládal všetky prvky smerovania od jednoduchých (statických), po zložité dynamické spôsoby a algoritmy hľadania ciest, za pomoci výkonných jadier procesorov. Potenciál je hlavne v multitaskingu, paralelnom vyhodnocovaní úloh, čím môže spracovávať úlohy niekoľkých smerovačov. Táto technológia nie je neznáma, bola dokonca používaná na Unixových počítačoch pred nástupom, rýchlejších, špecializovaných a samostatných smerovačov. Dnešný výpočtový výkon je ale na úrovni, kde prevyšuje kapacita potreby a mohlo by sa to v tomto smere pozitívne odzrkadliť. Úplne vytlačenie a nahradenie smerovačov však nie je možné očakávať, skôr je tu predpoklad doplnenia aktuálnych technológií o tieto nové možnosti.

Ďalším potenciálnym smerom vývoja, ktorý šliape na päty smerovačom sú viacvrstvové prepínače. Pracujú na podobnom princípe ako LAN switche, ale preposielanie rámcov sa riadi nie na základe MAC, ale IP adries. Sú akýmsi „hybridom“ preposielania datagramov na druhej a tretej vrstve internetového modelu, nemôžu však úplne nahradiť prepínače druhej vrstvy. Prijatý rámec udržujú v pamäti do tej doby, než zistia IP adresu datagramu v prijatom rámci. Vytvárajú si routovacie tabuľky a na základe porovnaní získanej IP a vlastnej tabuľky sa rozhodne o odoslaní k cieľu. Používajú pritom bežné smerovacie protokoly ako napr. RIP. Obmedzenie týchto prepínačov je v ich architektúre, ktorá sa zakladá na vstupno-výstupných portoch bežných LAN switchov. Preto podporujú len lokálne siete a poväčšine rozhranie Ethernet, čím nie sú plnohodnotnými smerovačmi schopnými pracovať v rozsiahlých WAN sieťach. Ich hlavná výhoda je v nižších nákladoch (než na smerovač) a schopnosť prevziať zaťaženie sieťovej komunikácie vnútropodnikových sietí. Tým ponechávajú smerovaču možnosť zefektívniť svoju činnosť, ktorá je tak využívaná len na prístup k sieti WAN.

Podobné možnosti dnešných trendov vo vývoji smerovania v sieti doplňujú hlavne oblasti, kde klasické smerovače zachádzali za svoje hardwarové možnosti. Rozširujú a obohacujú tak pojem smerovania v sieťach budúcnosti. I rozvoj a vývoj nových smerovacích protokolov napreduje a výsledkom sú tak efektívnejšie, stabilnejšie internetové siete, než tie, ktoré sme schopní postaviť za pomoci dnes zaužívaných sieťových komponentov.

2.5 Prepínanie v sieti

Pri presune údajov linkami je možné rozdeliť prenosy na základe spôsobu zaistenia spojenia. Spomeniem dva známe typy prenosových prostriedkov a to: prepínanie okruhov (circuit

switching) a prepínanie paketov (packet switching).

2.5.1 Prepínanie okruhov

Circuit switching, alebo prepínanie okruhov, je telekomunikačná technika naväzovania spojenia medzi účastníkmi prenosu dát. Je to technika, pri ktorej sa pred uskutočnením prenosu vytvorí medzi koncovími stanicami vyhradená cesta. Takéto spojenie sa zriadi cez rad prepínacích obvodov, a každý poslaný datagram prechádza po rovnakej fyzickej ceste. Po vytvorení spojenia nemôže byť daný okruh používaný ďalšími zariadeniami, až do doby, kým sa spojenie neukončí. Má to aj ďalšie nevýhody, aj keď aktuálne neprebieha na danej linke prenos, spojenie sa udržiava a zamedzuje tak využitie siete inými účastníkmi. Kanál, ktorý sa uvoľní a je pripravený na komunikáciu, sa označuje ako *nečinný* (angl. idle). Príkladom praktického využívania prepínania okruhov je technológia ISDN liniek. Moja aplikácia, znázorňujúca smerovacie algoritmy, zohľadňuje práve smerovanie v sieti s prepínaním okruhov.

2.5.2 Prepínanie paketov

Niekedy tiež označované *step-and-forward switching*, na rozdiel od predošlého typu nenadväzuje vopred komunikačný kanál. Nie je teda medzi dvoma miestami pevne vytvorená relácia, ale zariadenie sa pripojí do verejnej siete operátora a datové pakety odosiela cez túto komerčnú sieť. Informácia je prijatá medzi-stanicou a uložená pre neskoršie odoslanie k cieľu, alebo ďalšej stanici. Pred preposlaním informácie každá stanica, či uzol v sieti musí overiť integritu siete. Vo všeobecnosti sa táto metóda používa v sieťach s vysokou mobilitou, alebo premenlivou stabilitou spojenia, kde by trvalo naviazané spojenie bolo neustále prerušované. Čas potrebný na doručenie, by tak bol predĺžený o intervaly, potrebné na niekoľko-násobné opätovné naviazanie spojenia. Použitie je tiež vhodné pri komunikácii s veľkou mierou chybovosti, alebo s častými oneskoreniami pri doručovaní infomácie. Typickým použitím bol UUCP(Unix to Unix Copy) protokol, kde pri doručovaní informácie (napr. emailu), putovala po celej štruktúre siete, cez uzly ktoré neposielali informáciu okamžite, ale na druhej strane tak umožňovali takmer simultánne využívanie kapacity liniek. Paket switching je modifikáciou pôvodnej technológie store-and-forward, ktorý sa aktuálne používa sa napr. v sieťach X.25, či Frame Relay.

Kapitola 3

Návrh aplikácie

3.1 Výber jazyka

Pri voľbe implementačného prostredia som zvažoval niekoľko faktorov.

1. platformu na ktorej aplikácia pobeží
2. jednoduché užívateľské rozhranie
3. možnosti využitia profesionálnych knižníc

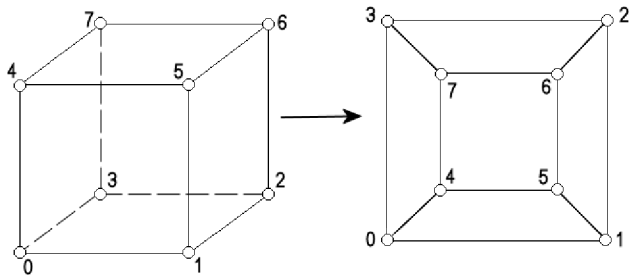
Vo výbere obstálo grafické vývojové prostredie Delphi od firmy Borland, a teda jazyk Object Pascal. Je to prostredie slúžiace k vývoju aplikácií pre platformu MS Windows, na ktorej moja aplikácia pobeží. Hlavnou výhodou tohto prostredia je, že umožňuje vizuálny návrh aplikácie a je postavené na využívaní integrovaných komponentov. Komponent je istý „balíček funkcií“, vykonávajúci určitú špecifickú činnosť. Vytváranie aplikácií pomocou Delphi uľahčuje a urýchľuje samotný vývoj.

3.2 Topológie

Počítačovú sieť je možné reprezentovať matematickým modelom – *grafom* [6]. Graf je usporiadaná dvojica množín vrcholov a hrán, ktorú môžeme zapísať ako $G = (N^*, E)$, kde N^* je množina všetkých vrcholov grafu a E je množina hrán z dvojíc vrcholov. Graf reprezentujeme tzv.*diagramom grafu*, čo je jeho grafická reprezentácia skladajúca sa z uzlov a hrán. Pri väčšom počte uzlov sa stáva neprehľadný preto je jeho účinok obmedzený, no pre aplikovanie v mojom návrhu dostačujúci. Hrana grafu môže byť zobrazená úsečkou, alebo vektorom (orientovaná úsečka so šípkou) a predstavuje linku v sieti. Linka (spojnica dvoch zariadení) reprezentuje vždy dvojicu komunikačných kanálov, v každom smere jeden. Kanál označuje spojenie počiatočného uzla a koncového uzla v jednom prechode procesu smerovania.

Sieťové topológie je potrebné prekresliť do vhodnej formy, zobraziteľnej v rovine. Príklad prekreslenia 3D modelu je vidieť na obrázku 3.1. Nemôžeme ale hovoriť o rovinných grafoch,

pretože jednotlivé spojnice sa budú v určitých prípadoch pretínať mimo vrcholy grafu (topológia typu „Oktagon“). V návrhu som uvažoval aj vykreslenie vopred neznámej schémy siete, podmienkou však u všetkých topológií je ich uzavretosť. Topológia je teda vopred známa už pri načítaní, alebo ju je možné prekresliť do uzatvoreného cyklického grafu (pravidelného n-uholníka).



Obrázok 3.1: Transformácia topológie typu „kocka“ do rovinného zobrazenia

3.3 Koncepcia programu

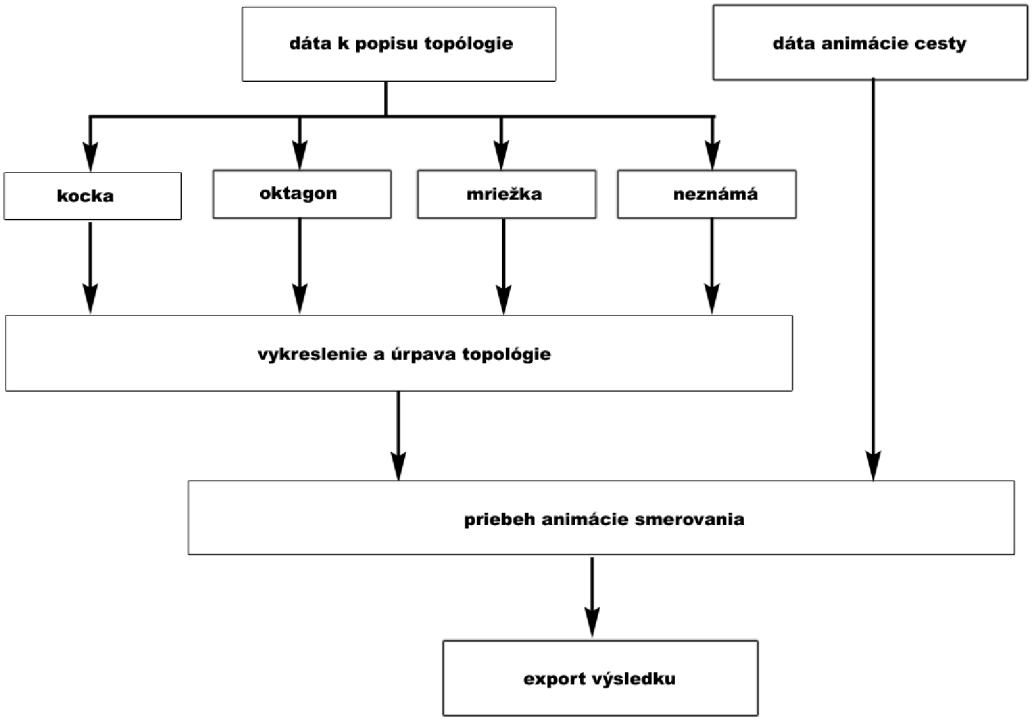
Tento program neaplikuje žiadne algoritmy smerovania v sieti. Jeho účel je vytvorenie grafického výstupu, pre textovú reprezentáciu komunikačného vzoru, vytvoreného iným nezávislým programom. Aplikácia má slúžiť ako demonštračný materiál využívaný k vizuálnemu znázorneniu topológií. Následne vo vykreslenom grafe bude znázornený priebeh smerovacieho algoritmu pre vybraný komunikačný vzor.

Program je rozdelený na tieto podproblémy (prehľadne znázornené na obrázku 3.2):

- Nahranie vstupných údajov
- Transformácia na zvolený rovinný graf a jeho vykreslenie
- Načítanie údajov o smerovaní
- Animácia nahranej komunikácie
- Export výsledku do obrázku

Po načítaní vstupných dát, ktoré budú obsahovať typ topológie (pokiaľ je vopred známa) a jej detailný popis (okolia jednotlivých uzlov), sa prevedie vykreslenie schémy. V prípade nečitateľnosti niektorej časti je možné tvar topológie meniť. Prevádza sa to výhradne posunom vykreslených uzlov, následkom čoho sa aktuálny graf prekreslí. Tento režim programu je nazývaný *editačný*.

Vstup údajov vizualizácie smerovania nasleduje až po vykreslení schémy. Týmto krokom vstúpi program do *animačnej* fázy a až do ukončenia, alebo zrušenia animácie, nie je

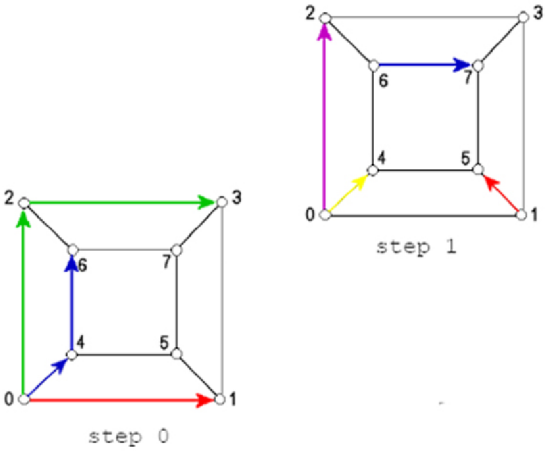


Obrázok 3.2: Bloková schéma postupu práce programu

možná ďalšia editácia vykreslených objektov. Po spustení sekvencie sa nastaví požadovaný, užívateľom definovaný, interval medzi jednotlivými krokmi prekresľovania. Aktuálnu pozíciu v animácii a prechody smerovania je možné sledovať graficky na zobrazenej topológii, ale i v vo forme textového výpisu. V grafickej podobe sú prechody v kroku odlíšené od seba farebne, aby bolo možné rozpoznať jednotlivé fázy smerovania. Príklad farebného rozlíšenia krokov smerovania demonštruje obrázok 3.3. Výsledok všetkých krokov smerovania nazývame *Cesta*. Je to usporiadaná množina kanálov $C = \{c_1, c_2, \ldots, c_n\}$. Jej dĺžka od zdroja k cieľu je $|C|$ (udáva s počtom hopov) [2]. Animačný mód je možné v ktoromkoľvek kroku pozastaviť, či manuálne posunúť vpred, alebo dozadu po časovej osi. V každom kroku pozastavenej animácie je možné exportovať grafické zobrazenie do externého súboru vo formáte JPEG, s takmer bezstratovou kompresiou, alebo BMP (Windows Bitmap). Rozdiel je v kvalite oboch formátov (bmp je kvalitnejšie zobrazený) a veľkosti výsledných súborov (jpeg je omnoho menší). Aplikácia je taktiež schopná uložiť celý priebeh animácie do samostatných súborov pri možnosti uložiť všetko.

Nakoľko každá linka pozostáva z dvoch kanálov smerovanie môže byť prevádzané v jednom kroku oboma smermi. Bolo potrebné túto skutočnosť prehľadne zakresliť do schémy. Navrhnutá koncepcia animácie túto možnosť zohľadňuje. Pri každom znázornení smeru komunikácie je vektor vykreslený o niekoľko pixelov mimo pôvodnú linku. Jeho orientácia je stále v smere putovania informácie a v grafe znázornená vždy v pravej polrovine vzhľadom k polohe úsečky. Súradnice počiatočného vektoru sú zhodné súradnicami úsečky a jeho orientácia daná poradím bodov v animácii. Posunutie bodov vektoru oproti pôvodnej linke je

```
--- 0. step of comunication ---
  Message from 0 -> 0:
0,
  Message from 0 -> 1:
0, 1,
  Message from 0 -> 3:
0, 2, 3,
  Message from 0 -> 6:
0, 4, 6,
--- 1. step of comunication ---
  Message from 0 -> 2:
0, 2,
  Message from 0 -> 4:
0, 4,
  Message from 1 -> 5:
1, 5,
  Message from 6 -> 7:
6, 7,
```



Obrázok 3.3: Farebné zobrazenie krokov cesty smerovania

vypočítané na základe analytického vzorca.

Príkald: linka v grafe je znázornená dvoma bodmi A, B so súradnicami $A[a_1, b_2]$ a $B[b_1, b_2]$. Dĺžku vektoru dostaneme ako $c = \sqrt{|(a_1 - b_1)| + |(a_2 - b_2)|}$. Teoretické posunutie pre každú súradnicu o „x“ pixelov vypočítame takto:

$$v_x = \frac{a_2 - b_2}{c} \cdot x \tag{3.1}$$

$$v_y = \frac{b_1 - a_1}{c} \cdot x \tag{3.2}$$

Potom súradnice posunutého vektoru $\overrightarrow{CD}$ budú $C[a_1 + v_x, a_2 + v_y],\ D[b_1 + v_x, b_2 + v_y]$.

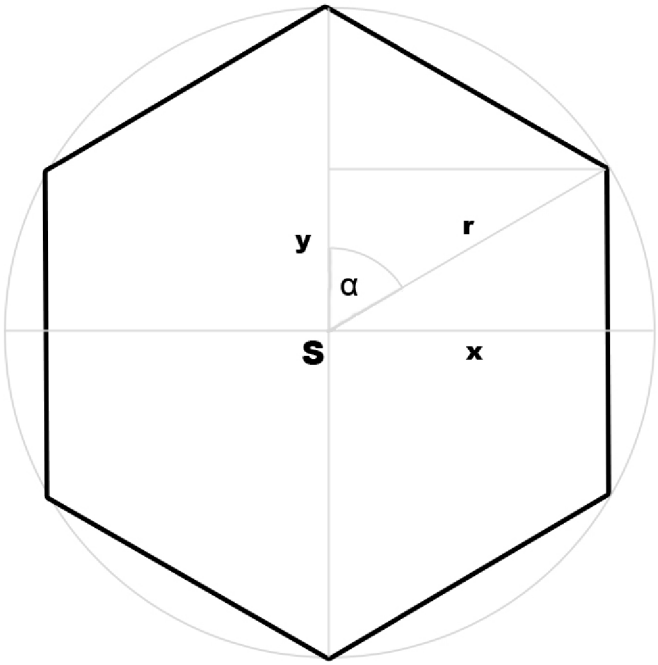
Po naplnení štruktúry bodov je prevedený výber geometrického tvaru topológie. V prípade kocky nasleduje jej transformácia. Tá prebieha v niekoľkých krokoch. V prvom kroku sa nájdu uzly vnútorného štvorca. Prvý bod, ktorý bol načítaný, sa ustanoví za referenčný. Na základe jeho susedov, sa nájde protiľahlý uzol, ktorý leží diagonálne oproti referenčnému uzlu. Týmto sa uzavrie vnútorný štvorec a je známy i prvý bod vonkajšieho štvorca (tretí sused referenčného bodu). Porovnaním susedov ostatných troch vnútorných bodov sa doplní vonkajší štvorec. Následne sa všetky uzli prenesú do predpripravenej šablony rovinnej topológie.

V prípade neznámej topológie sa musí podľa počtu uzlov vykresliť pravidelný n-uholník. Súradnice bodov sa odvodzujú podľa veľkosti stredového uhla α pre daný počet uzlov na kružnici a to takto (vzorce sú pre 1. kvadrant kružnice):

$$y = r \cdot \sin(\frac{\pi}{2} - \alpha) \tag{3.3}$$

$$x = \sqrt{r^2 - y^2} \tag{3.4}$$

Potom pre bod X n-uholníka platí: $X[s_1 + x, s_2 + y]$, kde s_1, s_2 sú súradnice stredu kružnice. Výpočet zobrazuje obrázok 3.4. Vzorec pre výpočet bodu v inom, než prvom kvadrante, je analogický.



Obrázok 3.4: Výpočet súradníc bodu n-uholníka

Kapitola 4

Implementácia

4.1 Načítanie vstupných údajov

Program očakáva vstupné dáta v dvoch samostatných súboroch. Prvým, je textový súbor s parametrami topológie sieti. Implicitne očakáva súbor s príponou TXT, ale je možné otvoriť prakticky akýkoľvek textový súbor s rovnakou štruktúrou obsahu. Je možné používať celoriadkové komentáre začínajúce znakom „`#`“, na prvej pozícii v riadku. Informácia uvedená za týmto znakom je automaticky ignorovaná. Topológia nevyžaduje žiadne informácie v úvode súboru, je načítaná automaticky, bez ohľadu na uvedený tvar či vopred známi počet uzlov. Jedinou výnimkou je typ „mriežka“, kde síce nie sú požadované žiadne informácie vopred, ale je nevyhnutné aby bola tvaru štvorca, teda odmocnina z počtu uzlov je celé číslo. Má ešte jedno obmedzenie a to v poradí uzlov. Transformácia ľubovoľne zadanej mriežky nie je implementovaná, preto sa na vstupe vyžaduje poradie uzlov v smere zľava doprava a zhora nadol. Topologické dáta sú očakávané v celočíselných vyjadreniach názvov uzlov oddelených medzerou:

`<číslo uzlu>` ␣ `<číslo susedného uzlu>` ␣ ... ␣ `<číslo posldného suseda>`

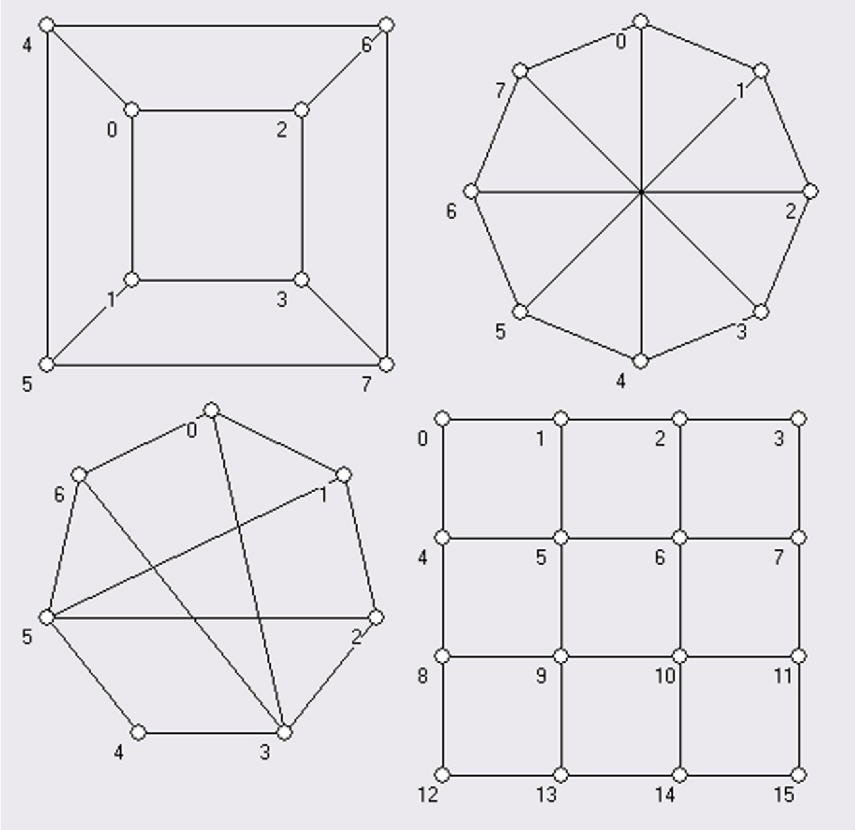
a v tomto poradí sú aj načítavané do štruktúry. Pokiaľ je naznačené smerovanie linky z bodu A do bodu B, ale jeho ekvivalent nie je uvedený (v susedoch druhého bodu sa nenachádza bod prvý), program spojenie medzi týmito uzlami vykreslí i na základe čiastočnej (jednostrannej) referencie. Pravidlom by ale malo byť, že každá linka je v popise siete uvedená 2x.

Po výbere sesty k súboru so sieťovou topológiou je užívateľ požiadaný o zvolenie typu načítanej topológie. Má na výber zo štyroch možností, reprezentujúcich jednotlivé typy siete, načrtnuté na obrázku 4.1.

- *kostka* – táto voľba prekresľuje ľubovoľne orientovanú šesťhrannú kocku
- *oktagon* – oktagonálny tvar siete je pravidelný osemuholník, s ľubovoľným množstvom uhlopriečok
- *mřížka* – prekresľuje štvorcovú mriežku

- *neznámy* – táto možnosť je použitá pre ostatné, samostatne neimplementované, topológie.

V prípade, že nie je vybraný žiadny zvolený tvar siete, nastaví sa implicitne typ „neznámy“.



Obrázok 4.1: Možnosti implementovaných typov topógií sietí (v smere hodinových ručičiek: kocka, oktagon, mriežka, nedefinovaný)

4.1.1 Kocka

Na popise tohto typu siete, vysvetlím základný postup prekresľovania údajov a v nasledujúcich častiach už len spomeniem odlišnosti pre jednotlivé druhy topológií.

Ako už bolo v práci spomenuté údaje o popise topológie sa nahrajú zo súboru v „surovej forme“. Nie je teda známe o aký typ ide, ani o počet uzlov. Po výbere typu užívateľom, nasleduje úprava. Kocka môže byť zadaná bodmi v akomkoľvek poradí. Jej prevod do správnej podoby zaručuje procedúra `TransformujKostka()`, ktorej je ako parameter predaný zoznam prvkov štruktúry typu S_Point. Táto štruktúra obsahuje údaj o názve uzlu (podľa prvého údaju v každom riadku súboru s popisom), súradnice $[x, y]$ bodov na vykresľovanom plátne, a pole okolitých bodov (znova ich názvov). Transformáciou sa vypočítajú súradnice jednotlivých bodov do pripravenej kostry grafu tvaru ako na obrázku 3.1. Následne

je volaná procedúra `Vykresli()`, ktorá podľa parametru vybranej topológie prevedie prekreslenie uzlovej štruktúry grafu na kresliace plátno. Kreslenie prebieha pomocou Delphi komponenty Image, s použitím primitívnych geometrických tvarov. Užívateľská editácia grafu spočíva v posune jeho uzlov, čím sa zapisujú nové súradnice do štruktúry S_Point vybraného bodu. Posuv začne po zaregistrovaní udalosti myši `onMouseDown` v okolí bodu do 5px (čo je veľkosť kružnice značiacej uzol grafu). Pri posuve bodu (tvar kurzoru myši sa v tomto momente zmení na `crHandPoint`) prebieha okamžité prekresľovanie zmenenej topológie. Po udalosti `onMouseUp` sa prekresľovanie zastaví a graf zachová poslednú polohu bodov, spolu s nastavením kurzora na pôvodný typ ukazateľa. V okolí každého bodu je popisné číslo podľa názvu uzlu v štruktúre a bude využívané pri hľadaní uzlu v animačnom režime.

4.1.2 Mriežka

Na tento typ sa neaplikuje, žiadna následná transformácia bodov zo vstupu, ale predpokladá sa ich správne poradie v súbore. Inak nie je možné graf vygenerovať a vykresliť. Mriežka štvorcového tvaru ľubovoľných rozmerov (3x3,4x4,5x5...), je teda hneď po načítaní pripravená. Podľa dĺžky strany cyklus graficky znázorní jej tvar na kresliace plátno. Vykresľovanie bodov na plátno prebieha v dvoch cykloch typu `for`, ktoré prechádzajú štruktúru mriežky poradkoch bod po bode. Každý susedný bod je vzdialený 70px vodorovne i zvisle od ďalšieho. Editácia prebieha rovnakým spôsobom, ako u predchádzajúceho typu.

4.1.3 Oktagon a Nedefinovaný tvar

Tieto dva tvary sietí popisujem spoločne, pretože ich princíp je rovnaký, rozdiel je len v jeho aplikovaní (na oktagon konkrétny, na neznámy typ všeobecný). Vopred nedefinovaný tvar topológie bude vykreslený na plátne ako pravidelný n-uholník z potrebným množstvom uhlopriečok. Súradnice oktagonu (pravidelného 8-uholníka) sú už vopred vypočítané a body grafu sú len správne umiestnené do vopred pripravenej topológie. Je to rýchlejšie a časovo menej náročné, pretože vieme vopred známy počet bodov. V prípade odlišného počtu bodov, než 8, je potrebné zvoliť „neznámy“ typ, ktorý si s tým poradí bez problémov. Počet susedov nie je dôležitý, každý bod môže mať vopred nešpecifikovaný a rôzny počet výstupných liniek, a platí aj tu princíp automatického vykreslenia hrany v prípade naznačenia linky aspoň jedným smerom (nájdenie suseda i bez spätnej referencie v popise tvaru siete).

Transformácia akejkoľvek inej topológie siete prebieha uložením bodov na kružnicu a vytvorením pravidleného n-uholníka. Súradnice bodov tohto polygónu sa vypočítavajú na základe ich počtu podľa definovaných vzťahov 3.3 a 3.4, poprip. ich analógií. Program je nastavený tak, že v prípade počtu bodov väčšieho než 15 zväčší polomer kružnice, na ktorú body vysádza, aby rozšíril vzdialenosti medzi nimi a zvýšil tak prehľadnosť a čitateľnosť napr. popisných čísel uzlov.

4.2 Animácia

4.2.1 Vstup

Dáta potrebné pre animáciu sa načítavajú z externého súboru, textového formátu, podobne ako tomu bolo pre vstup popisu topológie. Pre celo-riadkové komentáre ostal zachovaný znak `#`. Kroky vizualizácie sú formátovane nasledovne:

```
# toto je jednoriadkový komentár, ktorý program preskakuje
--- 0.step of communication ---
Message from 1 -> 6:
1, 3, 4, 6,
```

Každý krok smerovania je uvedený a zakončený „`---`“. Nasleduje správa odkiaľ, kam smeruje aktuálna cesta. Najdôležitejšia časť je postupný výpis uzlov grafu oddelených čiarkou (biele znaky sa vo výsledku zanedbávajú). Po nahraní všetkých ciest animácie, je možné spustiť vizualizíciu. V tomto okamžiku už program prešiel z editačného módu do animačného, a nie je možné meniť tvar grafu. Užívateľ je vyzvaný aby zadal požadovaný časový interval v sekundách medzi jednotlivými krokmi animácie. Pokiaľ nezadá nič, implicitne sa nastaví interval na 3s, a program na túto skutočnosť upozorní dialógovým oknom.

4.2.2 Mechanizmus

Vizualizácia ciest smerovania je znázorňovaná vektormi (farebnými úsečkami so šípkou v smere cesty). Nachádzajú sa vždy v pravej polrovine od hrany, ktorú znázorňujú. Princíp výpočtu presnej polohy vektoru bol popísaný rovnicami (3.1, 3.2) na strane 15. Farba sa vyberá tak, aby bola cesta z počiatočného bodu do cieľového znázornená jednou farbou, odlišnou od ostatných ciest v danom smerovacom kroku. Krok sa prevedie vždy po spustení úlohy `onTimer` časovača, nastaveného užívateľom.
V menu je možné priebeh ovládať prostredníctvom nasledovných položiek:

- *Načti* (Shift+O) – Slúži k nahratiu súboru s dátami pre vizualizáciu a zároveň prepne program do animačného režimu
- *Animuj* (Shift+A) – Spúšťa priebeh animácie
- *Zastav* (Shift+P) – Pozastaví priebeh na aktuálnom kroku
- *Pokračuj* (Shift+R) – Spusti pozastavenú animáciu od posledného kroku
- *Zruš* (Shift+Q) – Zruší animáciu, a vráti sa do editačného režimu

Pri pozastavení, alebo ukončení animácie, je možné sa presunúť na jej ľubovoľný krok pomocou zmeny pozície komponenty `trackBar`, umiestnenej na spodnom okraji vykresľovaého plátna. Tento element je viditeľný len v animačnom režime aplikácie.

4.2.3 Export

Export výsledku animácie do obrazového formátu prebieha priamym uložením kresliacej plochy do súboru. V prípade JPEG formátu je bitmapa uložená so 100% kvalitou obrazu, aby sa predišlo znehodnotenie obrazu kompresiou. Pri voľbe „Uložit vše“ je vynulovaný časovač a vnútorne počítadlo, následne spustená ukládacia sekvencia s veľmi malým krokom, dostatočným na uloženie vykresleného obrazu. Každý súbor dostáva okrem časti názvu zadaného užívateľom i automaticky generovaný index kroku v ktorom sa nachádza aktuálne vyobrazený stav smerovania.

Kapitola 5

Záver

Zložité smerovanie v internetových sieťach dnes riešia vysoko prepracované algoritmy. Smerovače nájdu najoptimálnejšiu cestu k cieľu nielen v jednoduchých LAN sieťach, ale cez hlavné chrbticové siete kontinentov. Počas príprav návrhu vytvorenej aplikácie som sa s týmito technikami oboznámil podrobnejšie a rozšíril som si i obzory z oblasti tvorby užívateľských rozhraní.

Využitie aplikácie je v prvom rade vo výučbe, ako demonštračný materiál k názornému pochopeniu smerovacích protokolov v sieti, či tvorbe vizuálnych zobrazení smerovacích algoritmov. Na zvolených topológiách zobrazuje a animuje jednotlivé kroky smerovania podľa komunikačného vzoru. Informácie v grafickej podobe je možné exportovať do súborov. Aplikácia splňuje zadané požiadavky pre niekoľko rôznych typov topológií. V budúcnosti je možné program doplniť a rozšíriť tak jeho funkcie, napr. o analýzu smerovacích techník, či o ďalšie niekoľko-rozmerné sieťové štruktúry.

Pri písaní tejto odbornej práce som sa dozvedel i o aplikáciách podobného typu. Tieto produkty sú označované ako *komplexné riešenia pre správu sietí*. Komerčné voľne šíriteľné programy, ktoré nielen zvolené siete zobrazujú, ale úplne autonómne dokážu rozpoznávať zložité štruktúry a analyzovať postupy smerovania v nich. Freeware riešenia ako napr. Zabbix, Nagios, OpenNMS, sú aplikácie monitorujúce siete, zisťujúce dostupnosť jednotlivých zariadení a pri zistení problému sú schopné upozorniť sieťového administrátora. Niektoré rozpoznávajú pridané zariadenie do štruktúry sieti úplne autonómne, a tým vytvárajú výborné podmienky pre monitoring väčších a zložitejších sietí. Pre viac informácií odporúčam preštudovať príslušnú publikáciu [8].

Literatúra

[1] BRADNER, S. *The Internet Standards Process*. 1996. [online]. Posledná aktualizícia október 1996. [cit. 2008-05-05].
URL: `http://tools.ietf.org/html/rfc2026`

[2] DVROŘÁK, V. *Architektura a programování paralelních systémů*. 2007. [online]. Fakulta Informačních Technologií VUT v Brně. Prezentace. [cit. 2008-05-07].

[3] IBM *About IBM mainframe*. [online]. Posledná aktualizícia 2008. [cit. 2008-05-04].
URL: `http://www-03.ibm.com/systems/z/advantages/index.html`

[4] IESG *Applicability Statement for the Implementation of Classless Inter-Domain Routing (CIDR)*. Sept 1993. [online]. [cit. 2008-05-06].
URL: `http://www.faqs.org/rfcs/rfc1517.html`

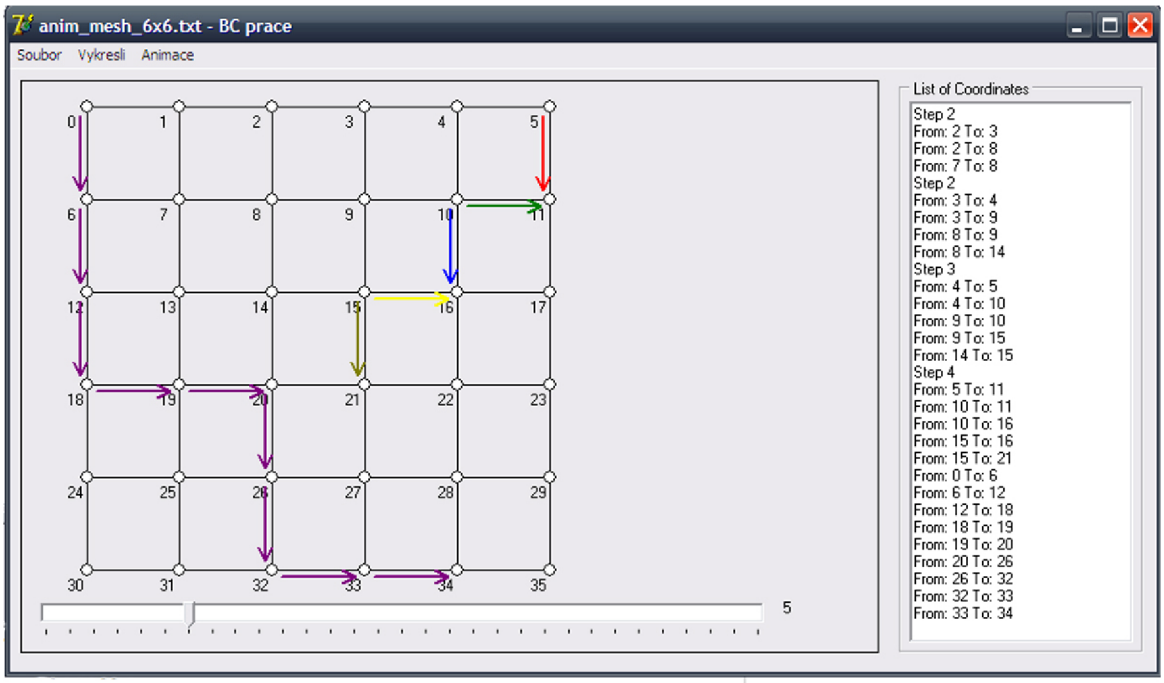
[5] JAKAB, F. *Počítačové siete*. 2007. [online]. posledná aktualizácia 2007. [cit. 2008-05-04].
URL: `http://www.cnl.tuke.sk/sk`

[6] PALÚCH, S. *Teória grafov*. 2001. [online]. [cit. 2008-04-29]. elektronická verzia. ISBN 80-7100-874-5.
URL: `http://frcatel.fri.uniza.sk/users/paluch/`

[7] SPORTACK, M. A. *Směrování v sítích IP*. Komputer Press. prvé vydání. ©2004. 351str. ISBN 80-251-0127-4.

[8] ZAREMBA, P. *Vizualizácia prevádzky a topológie sietových infraštruktúr*. Diplomová práca. Technická univerzita v Košiciach, Fakulta elektrotechniky a informatiky. 2006. 48str.

Príloha A



Obrázok 5.1: Ukážka aplikácie v animačnom móde

Príloha B

Na priloženom CD sú okrem elektronickej podoby tejto práce i zdrojové kódy programu a spustiteľný súbor. Obsahuje tiež adresár s príkladmi vstupných súborov popisu topológií a animácií. Aplikácia bola testovaná na OS Microsoft Windows XP SP2.