UNIVERZITA PARDUBICE
ÚSTAV ELEKTROTECHNIKY
A INFORMATIKY

POČÍTAČOVÁ PODPORA ŘÍZENÍ ROZSÁHLÝCH PROJEKTŮ

BAKALÁŘSKÁ PRÁCE

2007 **MARTIN ŠVÁHA**

UNIVERZITA PARDUBICE
ÚSTAV ELEKTROTECHNIKY
A INFORMATIKY

POČÍTAČOVÁ PODPORA ŘÍZENÍ ROZSÁHLÝCH PROJEKTŮ

BAKALÁŘSKÁ PRÁCE

AUTOR PRÁCE: Martin Šváha
VEDOUCÍ PRÁCE: doc. Ing. Josef Volek, CSc.

2007

UNIVERSITY OF PARDUBICE
INSTITUTE OF ELECTRICAL ENGINEERING AND INFORMATICS

COMPUTER SUPPORT FOR LARGE PROJECTS

BACHELOR WORK

AUTHOR: Martin Šváha
SUPERVISOR: doc. Ing. Josef Volek, CSc.

2007

Vysokoškolský ústav: Ústav elektrotechniky a informatiky
Katedra/Ústav: Ústav elektrotechniky a informatiky
Akademický rok: 2006/2007

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Pro: Šváha Martin

Studijní program: Informační technologie

Studijní obor: Informační technologie

Název tématu: Počítačová podpora řízení rozsáhlých projektů

Zásady pro zpracování: Cílem bakalářské práce je vytvoření počítačové podpory pro řízení projektů s ohledem na minimalizaci doby realizace s respektováním kapacitních omezení jednotlivých činností. Matematickým aparátem jsou dvě disciplíny operačního výzkumu: Teorie grafů a Síťová analýza. Práce bude mimo jiné zahrnovat metodu CPM, PERT a Branch and Bound.

Předpokládaná struktura bakalářské práce:

1. Úvod, motivace
2. Teoretické základy
3. Formulace problému
4. Návrh metody a algoritmu řešení
5. Počítačová implementace v jazyku C++
6. Řešení praktické aplikace
7. Zhodnocení vlastního příspěvku a závěr

Seznam odborné literatury:

[1] Klusoň, V.: Kritická cesta a metoda PERT v řídící praxi. Praha, SNTL 1968
[2] Plesník, J.: Grafové algoritmy. Bratislava, Veda 1983
[3] Volek, J.: Operační výzkum 1. Pardubice, Univerzita Pardubice 2005
[4] Demel, J.: Grafy a jejich aplikace. Praha, Academia 2002

Rozsah: 30-50 stran textu, 5 tabulek, 10 obrázků

Vedoucí práce: doc. Ing. Josef Volek, CSc.

Vedoucí katedry (ústavu): prof. Ing. Pavel Bezoušek, CSc.

Datum zadání práce: 31. 10. 2006

Termín odevzdání práce: 18. 5. 2007

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně Univerzity Pardubice.

V Pardubicích dne 17. 05. 2007

Martin Šváha

Poděkování

Rád bych poděkoval zejména vedoucímu práce panu doc. Ing. Josefu Volkovi za jeho připomínky, rady a konzultaci některých problémů této práce.

Abstrakt

Práce se zabývá softwarovou podporou pro řízení rozsáhlých projektů. Je zkoumána problematika rozmístění zdrojů v projektu při minimalizaci celkového času pro jeho vykonání. Dále jsou analyzovány některé algoritmy pro práci s metodami PERT, CPM a Branch & Bound.

Obsah

Úvod

V současnosti je stále více využívána počítačová podpora pro řízení rozsáhlých projektů v různých vědních oborech. Jednotlivé projekty se skládají z mnoha dílčích činností a je zapotřebí vhodně naplánovat, jak tyto činnosti na sebe budou navazovat, kdy a kým budou vykonávány. Za tímto účelem bylo vytvořeno několik algoritmů, které optimalizují celkový čas na daný projekt s ohledem na omezení zdrojů. Tyto úlohy lze řešit pomocí ručních výpočtů, ale při velmi rozsáhlých projektech jsou tyto výpočty zdlouhavé a snadno může dojít k chybám, které mají vliv na výsledek. Počítačová podpora k řešení těchto úloh umožňuje tyto projekty v krátkém časovém intervalu provést, zaručit přesné výsledky a v přehledné formě je zobrazit.

Práce se zabývá vytvoření počítačové podpory pro řízení projektů s ohledem na minimalizaci doby realizace s respektováním kapacitních omezení jednotlivých činností. Dále se práce věnuje metodě kritické cesty, metodě PERT a metodě Branch & Bound.

1. Motivace

Rozvoj průmyslových odvětví a různých vědeckých oborů vedl k potřebě plánování, koordinace a kontroly rozsáhlých a složitých projektů v těchto oblastech. Pro řízení úkolů byla vytvořena síťová analýza, což je souhrnný název pro metody, jejichž základ tvoří teorie grafů, teorie pravděpodobnosti a vědecké programování. Základ síťové analýzy tvoří metoda kritické cesty (CPM – Critical Path Method) a plánovací systém PERT (Program Evalution and Review Technique). Dále do této oblasti spadá nespočet modifikací těchto metod, jelikož obě základní metody se vyznačují svou jednoduchostí a přehledností.

1.1 Historie

1.1.1 Předchůdci CPM a PERT

Jedním z prvních způsobů zobrazení a řešení optimalizace byly Ganttovy diagramy, které Henry Gantt publikoval v roce 1910. Tato metoda byla používána téměř 50 let. Z hlediska řešení projektů byla nedostatkem Ganttových diagramů skutečnost, že nezachycují vztahy a závislosti mezi dílčími činnostmi. Nehodí se na řízení činností, kde je nutná koordinace.

Objevily se i další metody, které danou problematiku graficky zobrazovaly jiným způsobem. Jednou z nich byly postupové diagramy, které dokázaly vyloučit zbytečné pohyby na pracovišti, a zvýšit tak výkon práce co v nejkratším možném čase za minimální pracovní námahy. Po 1. světové válce byl v Německu vytvořen systém REFA, který pomohl vyvést Německo z ekonomické krize. Tento systém znázorňoval pracovní činnosti. Jednotlivé činnosti rozebíral, dokud nebyla každá složka jasná a časově měřitelná.

1.1.2 Historie kritické cesty

V roce 1957 vyvinuly společnosti E. I. Du Pont de Nemours & Co. a Sperry-Rand Corporation metodu kritické cesty. Cílem tohoto vývoje bylo vytvoření nástroje pro řízení složitých činností při výstavbě zařízení, rekonstrukce zařízeních a při vývoji nových chemických výrobků. První zkušební aplikace se uskutečnila v září 1957. Její

výsledky byly dokončeny v květnu následujícího roku. Jednalo se o výstavbu zařízení v hodnotě 10 mil. dolarů. Ukázalo se, že lze přesněji stanovit počet pracovníků a zároveň zkrátit dobu realizace o 2 měsíce a to bez zvýšení nákladů. Při vynaložení nákladů vyšších o 1% bylo možné zkrátit dobu o další 2 měsíce. Tyto výsledky vzbudily velkou pozornost. Další využití aplikace této metody byla oblast údržby. Šlo o preventivní údržbu reaktoru v továrně na výrobu neoprénu. Tato aplikace byla ukončena roku 1959 a její výsledky byly překvapující. Podařilo se snížit dobu trvání ze 125 hodin na 93. Bylo nalezeno 25 kritických činností z celkových 225, jejich urychlením bylo možné snížit dobu trvání opravy až na 78 hodin. Významné změny byly provedeny i v oblasti postupu práce. Autory metody CPM jsou James E. Kelly a Morgan R. Walker.

1.1.3 Vznik metody PERT a Branch & Bound

Počátkem roku 1958 tým operační analýzy práce v Úřadu válečného námořnictva USA začal vyvíjet nové řídící metody v oblasti vývoje raketových systémů. Jednalo se o vývoj systému rakety Polaris. Na vývoji spolupracovali Williard Hazard, pracovníci společnosti Booz Allen & Hamilton, zástupci společnosti Lockheed Missiles a pracovníci Úřadu válečného námořnictva. V červnu roku 1958 vydala tato skupina zprávu PERT, Summary Report, Phase I. V říjnu tohoto roku byl navržený systém zkušebně aplikován pro vývoj základních tří položek této zbraně. Šlo o 23 síťových diagramů, 2000 uzlů a 3000 činností. Uvádí se, že došlo ke zkrácení vývoje rakety o dva roky. Za autory jsou považováni D. G. Malcolm, J. H. Roseboom, W. Fazar a C. E. Clark, který položil důležité podklady ve své publikaci o síťových grafech.

Obě tyto metody se rychle rozšířily zejména ve stavebnictví a průmyslu. Postupem času byly rozšířeny dalšími postupy pro specifické účely. V roce 1960 byla představena metoda Větve a hranice (Branch and Bound) pro lineární programování. Autory jsou A. H. Land a A.G. Doig. Základem jsou přibližné výpočty, které vylučují možnosti mající nepříznivou dobu trvání. Využívána je pro různá optimální řešení, zejména v diskrétní a kombinatorické optimalizaci.

2. Teoretické základy

2.1 Některé pojmy teorie grafů

Incidence p grafu přiřazuje každé jeho hraně neuspořádanou dvojici vrcholů. Je-li incidence hrany $h \in X : p(h) = (u,v)$ hovoříme, že hrana h inciduje s vrcholy u a v. [4]

Neorientovaným grafem rozumíme uspořádanou trojici $G = (V, X, p)$. Prvky množiny V nazýváme vrcholy grafu G, prvky množiny X hranami grafu G a zobrazení množiny X na množinu všech neuspořádaných dvojic $V \otimes V$ $-p$ incidencí grafu G.

Orientovaným grafem $D = (V, Y, p)$ nazýváme uspořádanou trojici množin V, Y a zobrazení p množiny $Y \to V \times V$. Množiny V a Y mají obdobný význam jako u neorientovaných grafů s tím rozdílem, že množina Y je tvořena uspořádanými dvojicemi $[u, v]$ prvků množiny V, přičemž $u \neq v$ a každá takováto dvojice se vyskytuje v množině Y nejvýše jednou.

Sled S je střídavá posloupnost bezprostředně po sobě následujících vrcholů a hran grafu $G = (V, X, p)$, pokud:

$$S = \{u_0, h_1, u_1, h_2, u_2, \ldots, u_{n-1}, h_n, u_n\}, \text{ kde}$$

$$h_i \in X, p(h_i) = (u_{i-1}, u_i), \text{pro } i = 1,\ldots,n,\ u_i \in V \text{ pro } i = 1,\ldots,n,$$

$$u_0 = u, u_n = v.$$

Tah je sled, ve kterém se neopakuje žádná hrana.

Cesta je tah, ve kterém se neopakuje žádný vrchol.

Dráhou (orientovanou cestou) nazveme orientovaný sled, ve kterém se neopakuje žádný vrchol. Orientovanou cestu mezi dvojicí vrcholů u, v v orientovaném grafu $G = (V, Y, p)$ značíme $m(u,v)$.

Maximální dráhou z vrcholu u do vrcholu v orientovaného grafu $G = (V, Y, p)$ nazveme dráhu $m^*[u, v]$, pro kterou platí:

$$\sum_{h \in m^*[u,v]} o[h] = \max_{m[u,v] \in M} \left\{ \sum_{h \in d[u,v]} o(h) \right\}, \text{ kde } M \text{ je množina všech drah } m[u, v].$$

Acyklický graf je orientovaný graf, který neobsahuje žádný cyklus.

Síťový graf je orientovaný, neorientovaně souvislý, hranově ohodnocený, acyklický graf vyjadřující časovou a věcnou návaznost jednotlivých činností projektu.

Vrcholově (hranově) ohodnoceným grafem nazveme graf $G = (V, Y, p)$, pokud existuje funkce $o(v)$ (resp. $o(h)$), která přiřadí každému vrcholu $v \in V$ (hraně $h \in Y$) nezáporné číslo vyjadřující určitou kvantitativní nebo kvalitativní vlastnost vrcholu (hrany).

Matice přímých vzdáleností

Matice přímých vzdáleností hranově ohodnoceného grafu G (V, Y, p) je čtvercová matice $D = \left(d_{ij}\right)_{i,j=1}^{n}$, pro kterou platí:

$d_{ij} = o(h)$, jestliže $\exists\ h \in Y$, pro kterou $p(h) = (v_i, v_j),\ i \neq j$

$d_{ij} = \infty$, jestliže $\nexists\, h \in Y$, pro kterou $p(h) = (v_i, v_j),\ i \neq j$

$d_{ij} = 0$, pro $i = j$

2.2 Pojmy z oblasti algoritmů pro operace s grafy

Asymptotická složitost klasifikuje počítačové algoritmy. Jedná se o funkci udávající, jakým způsobem se bude chování algoritmu měnit v závislosti na změně velikosti vstupních dat. [6]

Topologické uspořádání vrcholů a hran grafu G je taková posloupnost $v_1, v_2, \ldots, v_n$ všech vrcholů grafu, že kdykoliv vede orientovaná hrana z v_i do v_j, platí $i < j$. To znamená, že počáteční vrchol hrany je v topologickém uspořádání uveden vždy dříve než koncový vrchol téže hrany. Pokud existuje topologické uspořádání vrcholů a hran, pak je graf acyklický. [2]

2.3 Pojmy z oblasti datových struktur

Abstraktní datový typ (Abstract Data Type - ADT) je matematická struktura skládající se z:

- jedné nebo více domén
- jedné nebo více matematických operací na prvcích těchto domén

Abstraktní datový typ umožňuje skrýt implementaci jednotlivých operací nad daným datovým typem a způsob fyzického uložení informace v proměnné daného datového typu. Hlavním cílem je zjednodušit a zpřehlednit program, který provádí operace s daným datovým typem. [6]

3. Formulace problému

Pro účely řízení projektů je nutná dokonalá znalost problému. Systém pro řízení projektů by měl splňovat následující podmínky[3]:

a) Systém by měl nabízet kompletní a přesné informace určující vztahy a závislosti mezi jednotlivými dílčími činnostmi. Pomocí těchto informací bychom měli být schopni vytvořit přesnou organizaci pracovních postupů tak, aby byla koordinace všech dílčích prvků projektu co nejefektivnější.

b) Umožnění vytvoření lhůtového plánu v takové míře, abychom dokázali určit, které činnosti mají větší prioritu pro dokončení celého projektu. Pokud by vznikly nějaké problémy, měli bychom být schopni na ně reagovat a provádět různá opatření pro zamezení vzniku zpoždění celkové doby projektu.

c) Určování efektivního využití veškerých zdrojů projektu. Jedná se o rozmístění zdrojů v projektu při minimalizaci nákladů a současně využití těchto zdrojů při minimalizaci časového trvání projektu.

Tyto požadavky jsou splněny oběma metodami – CPM i PERT. Hlavní rozdíl mezi těmito metodami je pohled na časové, případně věcné ohodnocení činností grafu. Metoda kritické cesty (CPM) má dobu trvání dílčích činností jednoznačně určenou, zatímco PERT je metodou stochastickou. Jednotlivé činnosti jsou vyjádřeny třemi odhady, které jsou brány jako náhodné proměnné s určitým rozdělením pravděpodobnosti. Metoda PERT umožňuje určit pravděpodobnost ukončení celého projektu i jeho vymezených etap.

K tomu, aby byl projekt realizován, nestačí pouze určit časový průběh, ale i řešit otázku pracovních sil a materiálního zabezpečení. Často dochází k nedostatku požadovaných zdrojů, čímž se může zkomplikovat včasné splnění projektu.

Metody síťové analýzy umožňují tyto problémy řešit a provádějí rozbor, zda-li je nutné začít dílčí činnosti dříve, případně je urychlit nebo zpomalit.

Na začátku projektu je nutné vytvořit časovou organizaci celého projektu, rozvrhnout závislosti jednotlivých aktivit a brát v potaz dostupné zdroje a pracovní síly. Poté se přistoupí ke konstrukci síťového grafu.

Při sestavování síťového grafu je třeba nejdříve určit pro jaký projekt má být sestaven a jaká metoda síťové analýzy bude použita a to tak, aby výsledky, které jsou požadovány, byly co nejvíce dostačující k jeho realizaci. Jelikož se tato práce zabývá hned několika metodami, budou dané problémy a použité algoritmy rozebrány odděleně.

3.1 Sestavení síťového diagramu

Prvkem grafu je vrchol, který znázorňuje okamžik v čase, kdy začíná nebo končí některá z činností. Všechny činnosti vycházející z jednoho vrcholu mají společné všechny bezprostředně předcházející činnosti. Výjimku tvoří počáteční nebo koncový vrchol, do kterého nevchází žádná z hran, respektive z něj žádná hrana nevychází.

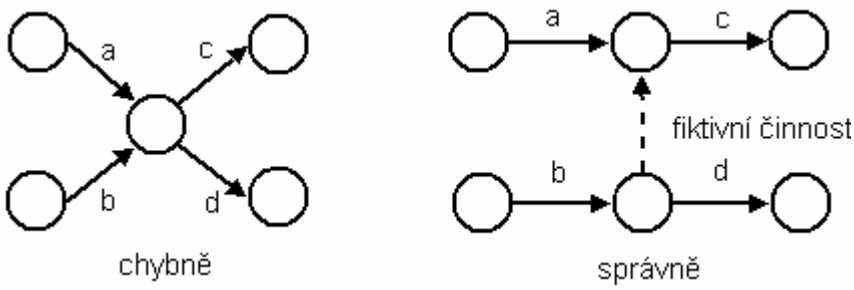
Jednotlivé dílčí činnosti jsou v síťovém grafu znázorněny pomocí hran. Každá hrana má 2 uzlové body. Každá spojnice mezi vrcholy tedy odpovídá určité činnosti. Tento způsob však nestačí k řešení většiny problémů. Nejde pouze určit, které dílčí operace je nutné vykonat, ale je nezbytné dodat, v jakém pořadí a jak na sebe navazují. K znázornění projektu je nutné použít nejen reálných hran, představujících reálné činnosti, ale i fiktivní činnosti, které neprobíhají v žádném čase, ani nespotřebovávají žádné zdroje.

Základní pravidla pro sestavení síťového grafu:

1. Každá činnost má vždy jeden uzel počáteční a jeden uzel koncový. Pomocí těchto vrcholů je každá činnost jednoznačně určena.

2. Síťový graf má vždy jeden počáteční a jeden koncový uzel. Pokud má diagram více vstupních vrcholů, řeší se daný projekt tak, že je vybrán libovolný z počátečních uzlů a z něj jsou vedeny fiktivní činnosti do ostatních počátečních vrcholů. Tento postup se nazývá normalizace síťového diagramu.

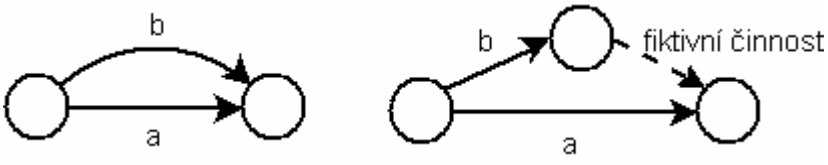
3. Síťový graf musí popisovat správně závislosti jednotlivých dílčích činností. Tyto závislosti lze vyjádřit pomocí fiktivních činností.



Obrázek. 1: Fiktivní činnost – Zdroj: [4]

Jestliže jen určitá část následných činností závisí na dokončení předchozích činností, pak je potřeba tyto činnosti oddělit od ostatních tak, aby byly nezávislé. A to pomocí fiktivních hran jako je znázorněno na obrázku 1. Činnost *c* závisí na činnostech *a*, *b*, ale činnost d závisí pouze na dokončení činnosti *b*. Díky fiktivní hraně je činnost *d* uvolněna ze závislosti na činnosti *a*.

4. Souběžné činnosti v síťovém diagramu jsou oddělovány činnostmi fiktivními. Takto zakreslené činnosti (obrázek 2) by byly v rozporu s prvním pravidlem o jednoznačnosti určení činnosti.



Obrázek 2: Násobná hrana – Zdroj: [4]

5. Síťový diagram nesmí obsahovat cyklus. Vznik v cyklu znamená, že diagram obsahuje logickou chybu a nelze projekt nikdy dokončit.

Při sestavování síťového grafu se předpokládá znalost všech činností, činností jim předcházejících a činností, které za nimi bezprostředně následují.

3.2 Problematika kritické cesty

3.2.1 Lhůtové ukazatele a zásady metody CPM

Pro výpočty lhůtových ukazatelů jsou potřeba tyto základní informace:

- Seznam všech činností, které jsou potřeba k realizaci projektu
- Potřebný čas k vykonání jednotlivých činností
- Vazby a závislosti mezi činnostmi

S využitím těchto údajů pomocí metody kritické cesty můžeme vypočítat pro každou činnost její lhůtové ukazatele. Metoda kritické cesty rozhoduje, které aktivity jsou kritické pro dokončení projektu (tyto činnosti leží na kritické cestě) a odhaluje, které činnosti mají pružnou dobu pro jejich vykonávání (činnosti nekritické). Kritickou cestu tvoří posloupnost činností s nejdelší dobou trvání, určující nejkratší možný čas trvání realizace projektu. Jakákoliv prodleva na kritické cestě má za následek prodloužení celkového času na dokončení projektu. To znamená, že neexistuje žádný volný čas pro tyto činnosti. Projekt může mít více kritických cest. Současně s kritickou cestou existují i cesty, jejichž čas na dokončení je kratší. Tyto cesty se nazývají nekritické.

Tyto výsledky umožňují projektantům a řídícím projektantům rozdělovat prioritu jednotlivým činnostem ke zkrácení doby dokončení projektu. K provedení algoritmu pro určení kritické cesty používáme několik lhůtových ukazatelů.

$o[v_i, v_j]$ – doba trvání činnosti $[v_i, v_j]$

t_i – nejdříve možný termín zahájení činnosti $[v_i, v_j]$

t_j – nejdříve možný termín ukončení činnosti $[v_i, v_j]$

$t_i^{/}$ – nejpozději nutný termín zahájení činnosti $[v_i, v_j]$

$t_j^{/}$ – nejpozději nutný termín ukončení činnosti [v_i, v_j]

D_{ij} – maximální disponibilní čas, $D_{ij} = t_j^{/} - t_i$

d_{ij} – minimální disponibilní čas, $d_{ij} = t_j - t_i^{/}$

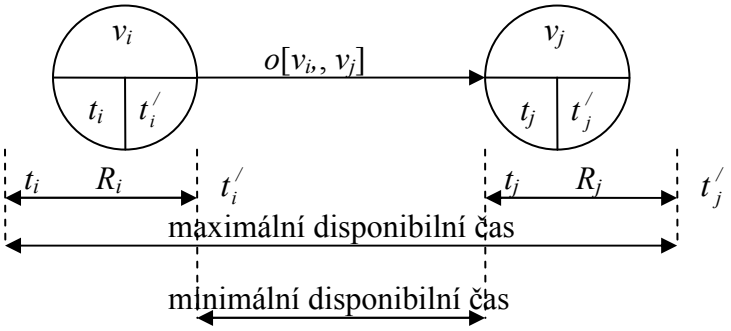
R_i – interferenční rezerva *i*–tého uzlu, $R_i = t_i^{/} - t_i$

R_j – interferenční rezerva j–tého uzlu, $Rj = t_j^{/} - t_j$

R_{ij}^c – celková časová rezerva, $R_{ij}^c = D_{ij} - o[v_i, v_j]$

R_{ij}^n – nezávislá časová rezerva, $R_{ij}^n = d_{ij} - o[v_i, v_j]$

R_{ij}^v – volná časová rezerva, $R_{ij}^v = t_j - t_i - o[v_i, v_j]$



Obrázek 3: Lhůtové ukazatele metody CPM – Zdroj: [3]

3.2.2 Postup výpočtu kritické cesty [3]

1. **krok:** V síťovém grafu položíme hodnotu nejdříve možného začátku projektu $t_0 = 0$.

2. **krok:** Určíme časy nejdříve možných začátků dílčích činností projektu t_i pomocí algoritmu na určení maximální dráhy.

3. **krok:** Poslední určená hodnota udává nejdříve možný čas t_n dokončení celého projektu.

4. **krok:** Určíme čas nejpozději možného ukončení celého projektu, který je roven času nejdříve možného ukončení projektu $t_n \left(t_n = t_n^{/}\right)$.

5. krok: Určíme časy nejpozději nutného ukončeních jednotlivých dílčích činností $t_i^{/}$.Každému vrcholu $v_i \in V$ přiřadíme hodnotu $t_i^{/} = \infty$, pro $i = 1,2,\ldots,n-1$. Hledáme orientovanou hranu $h \in Y, p[h] = [v_i, v_j]$ pro kterou platí:

$$t_i^{/} - o[v_i, v_j].$$

5a. krok: Pokud orientovaná hrana, která splňuje danou nerovnost, existuje, pak můžeme určit čas nejpozději nutného ukončení činnosti:

$$t_i = t_i^{/} - o[v_i, v_j].$$

5b. krok: Jestliže neexistuje hrana, která by splňovala tento vztah, pokračujeme krokem 6.

Čas nejpozději nutného ukončení činnosti ve výchozím uzlu $v_0 \notin V$ je roven $t_0 = 0$.

6. krok: Kritickou cestu $m[v_0, v_n]$ vytvoříme tak, že pro všechny činnosti určíme celkovou časovou rezervu:

$$R_{ij}^c = t_j^{/} - t_i - o[v_i, v_j].$$

Činnosti, pro které platí $R_{ij}^c = 0$ leží na kritické cestě a jsou kritické.

Kritická cesta je libovolná dráha, u které je součet ohodnocení hran je maximální. Všechny činnosti na této dráze jsou **kritické činnosti**. Prodloužením libovolné kritické činnosti o čas *e* dojde k prodloužení celkové doby pro dokončení projektu t_n o čas *e*.

3.3 Problematika metody PERT

Při řízení lidské činnosti je velice obtížné přesně odhadnout délku trvání. Musíme brát v úvahu i míru rizika při vykonávání dílčích činností, jakožto určitý vliv nahodilosti a nějakým způsobem ji ohodnotit.

Při rozsáhlejších projektech výrazně roste stupeň nejistoty a riziko s ním spojené včasného ukončení celého projektu. Tento problém si žádá brát v úvahu prvek nahodilosti, který je potřeba nějakým způsobem kvantitativně vyjádřit pro výpočet.

3.3.1 Časové odhady

Vzhledem k nemožnosti opakování většiny činností vzniká problém nedostatku statistických dat, na jejichž základě by byla určena délka celého projektu. Nezbývá, než se spolehnout na zkušenosti a úsudek pracovníků – ti jsou do jisté míry schopni odhadnout podmínky a rizika realizace dílčích činností. Odhad se provádí v časovém intervalu, který je určen minimální a maximální dobou trvání činnosti. Dále je potřeba ještě odhadnout, v jakém čase bude činnost s největší pravděpodobností ukončena. Máme tedy tři odhady, které se nazývají:

Optimistický odhad trvání činnosti – označujeme jej symbolem *a*. Je to nejkratší uvažovaná doba trvání činnosti. Dodržení tohoto termínu je ale velice nepravděpodobné. Ke splnění tohoto termínu by musela být dokonalá souhra všech okolností s jistou dávkou štěstí. [3]

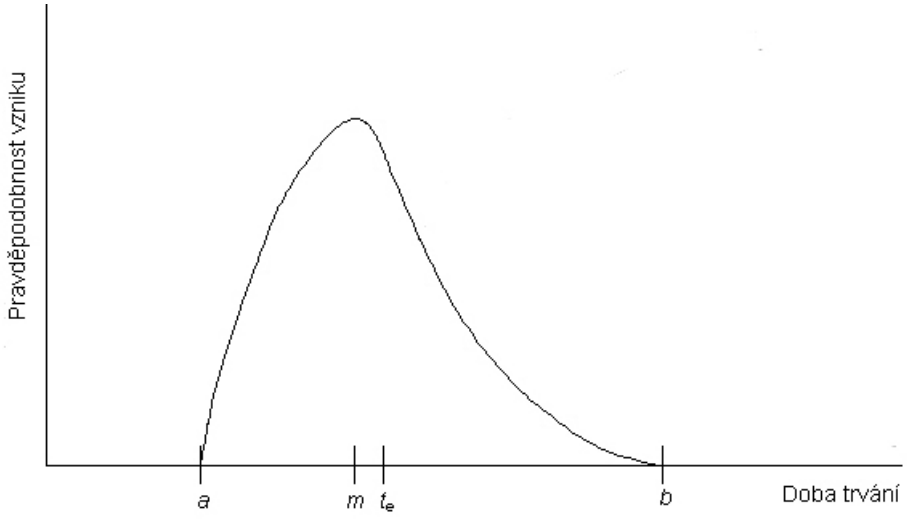
Odhad nejpravděpodobnější doby trvání činnosti – značíme *m*. Pokud bychom danou činnost prováděli opakovaně, pak by právě tento odhad času byl nejčastěji realizován. Jedná se o modus příslušného rozdělení pravděpodobnosti.

Pesimistický odhad doby trvání činnosti – *b*. Doba, která je odhadována jako nejpozději možná k dokončení činnosti. Tento termín má stejně jako optimistický odhad velmi malou pravděpodobnost, jelikož je brán se všemi možnými předvídatelnými obtížemi, které mohou při vykonávání nastat.

3.3.2 Rozdělení pravděpodobnosti doby trvání činnosti

Předchozí tři odhady jsou transformovány na jeden odhad t_e, který je dále používán pro výpočet dalších lhůtových ukazatelů. Tyto odhady umožňují konstruovat hypotetickou křivku funkce rozdělení pravděpodobnosti.

Křivka hustoty pravděpodobnosti může být symetrická v případě, že $t_e = m$. Nebo asymetrická, kdy je její zkosení buď vlevo nebo vpravo. V praxi se ovšem používá případ znázorněný na obrázku 4, protože odhad *b* bývá záměrně nadhodnocen.



Obrázek 4: Beta rozdělení[1]

Modus (*m*) a **rozpětí** (*a* – *b*) jsou důležitými charakteristikami polohy a variability rozdělení doby trvání činnosti, ale nelze s nimi dále pracovat, jelikož s nimi nelze provádět odhady pravděpodobnosti a výpočty lhůtových ukazatelů.

Tyto charakteristiky ovšem vedou k výpočtu dalších dvou charakteristik, střední hodnoty t_e a směrodatné odchylky σ_{t_e} doby trvání činnosti, které již splňují požadavky odpovídající našim potřebám.

Systém PERT předpokládá, že náhodná veličina, kterou je doba trvání činnosti, má rozdělení beta.

Funkce hustoty pravděpodobnosti je dána vztahem:

$$f(x) = \frac{1}{B(p,q)} x^{p-1}(1-x)^{q-1}, \text{ pro } 0 \le x \le 1, \text{ [3]}$$

kde $B(p,q) = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}$ vycházející ze vztahu funkce beta k funkci gama.

[1] Zdroj: http://www.ce.cmu.edu/pmbook/images/fig9_9.gif

Distribuční funkce je dána vztahem:

$$F(x) = \frac{1}{B(p,q)} \int_0^1 x^{p-1} (1-x)^{q-1} \, \mathrm{d}x \, .$$

S využitím pravidla tří sigma určíme hodnotu σ_{t_e} pro *B*–rozdělení aproximovánu jednou šestinou rozpětí:

$$\sigma_{t_e} = \frac{b-a}{6},$$

z čehož plyne vztah pro rozptyl:

$$\sigma_{t_e}^2 = \frac{(b-a)^2}{36} \, .$$

Výpočtem počátečního momentu z rovnice rozdělení hustoty *B*–pravděpodobnosti lze s využitím několika dílčích úprav dospět ke vzorci pro očekávaný čas:

$$t_e = \frac{a + 4m + b}{6}.$$

3.3.3 Výpočet hodnot $T_E^{v_i}$, $T_L^{v_i}$, $\sigma_{T_E^{v_i}}$, $\sigma_{T_L^{v_i}}$ a R^{v_i}

Časy jednotlivých uzlů grafu chápeme jako náhodné proměnné rozdělení pravděpodobnosti blížící se asymptoticky k normálnímu rozdělení $N(\mu, \sigma^2)$, kde μ je střední hodnota rozdělení a σ^2 rozptyl rozdělení.

Hustota pravděpodobnosti normálního rozdělení je dána vzorcem:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}} \, .$$

Středními hodnotami normálního rozdělení jsou lhůtové ukazatele $T_E^{v_i}$ a $T_L^{v_i}$ [3]. $T_E^{v_i}$ je délka maximální dráhy z počátečního uzlu v_0 do uzlu v_i. Výsledný vztah je pak dán vztahem:

$$T_E^{v_i} = \sum_{h \in \tilde{m}[v_0, v_i]} t_e[h].$$

Směrodatná odchylka času $T_E^{v_i}$ je pak analogicky vyjádřena:

$$\sigma_{T_E^{v_i}} = \left(\sum_{h \in \tilde{m}[v_0, v_i]} \sigma_{t_E}^2 [h] \right)^{\frac{1}{2}}.$$

$T_L^{v_i}$ je celkový čas maximální dráhy z uzlu v_i do koncového uzlu v_n:

$$T_L^{v_i} = T_L^{v_n} - \sum_{h \in \tilde{m}[v_i, v_n]} t_e [h].$$

Pro směrodatnou odchylku času $T_L^{v_i}$ platí vztah:

$$\sigma_{T_L^{v_i}} = \left(\sum_{h \in \tilde{m}[v_i, v_n]} \sigma_{t_E}^2 [h] \right)^{\frac{1}{2}}.$$

Časy T_E a T_L představují střední hodnoty dvou na sobě nezávislých normálních rozdělení, jejichž křivky hustoty pravděpodobnosti jsou závislé na $\sigma_{T_E}^2$ a $\sigma_{T_L}^2$. Čím vyšší je hodnota směrodatné odchylky, tím má křivka hustoty pravděpodobnosti plošší tvar.

Hodnota rezervy v uzlech se může zvětšovat nebo zmenšovat. Může se stát, že bude dokonce nabývat záporných hodnot. Tento případ může nastat pouze tehdy, jestliže se obě křivky protínají. To znamená, že hodnota $T_E^{v_i}$ může být větší než hodnota $T_L^{v_i}$, potom platí:

$$T_E^{v_i} - T_L^{v_i} = -R \text{ . [3]}$$

V takovém případě může dojít k ohrožení včasného splnění celého projektu. Z tohoto důvodu je někdy účelné zkoumat i nekritické uzly a zjišťovat s jakou pravděpodobností se tyto uzly mohou stát kritickými $P\{R \leq 0\}$.

Je potřeba určit podíl rezervy se záporným znaménkem a druhé odmocniny součtu rozptylů obou rozdělení:

$$u = \frac{T_E - T_L}{\sqrt{\sigma_{T_E}^2 + \sigma_{T_L}^2}} = -\frac{R}{\sigma_{T_R}}, \text{ kde}$$

$$\sigma_{(T_E - T_L)}^2 = \sigma_{T_E}^2 + \sigma_{T_L}^2 = \sigma_{T_R}^2 . \text{ [4]}$$

Hodnoty pravděpodobnosti odečteme z tabulky distribuční funkce normálního rozdělení $N(0, 1)$:

$$P\{R \le 0\} = \Phi\left(\frac{-R}{\sigma_{T_R}}\right) = \Phi(u).$$

Hodnota u závisí jak na velikosti časové rezervy, tak i na velikosti směrodatných odchylek obou rozděleních.

3.3.4 Pravděpodobnost dodržení plánovaného termínu

Mnohem významnější úlohu plní odhad pravděpodobnosti dodržení plánovaných termínů, a to jak termínu ukončení celého projektu, tak i jeho významných etap.

Je–li pro určitý uzel dán termín $T_S^{v_i}$, pak odhad pravděpodobnosti pro dodržení plánovaného termínu ukončení činností vcházejících do tohoto uzlu je $P\left(T_E^{v_i} = T_S^{v_i}\right)$. Stačí položit $T_S^{v_i} = T_L^{v_i}$ a hodnotu $T_S^{v_i}$ dosadit do vzorce pro výpočet u:

$$u' = \frac{T_E^{v_i} - T_L^{v_i}}{\sqrt{\sigma_{T_E^{v_i}}^2 + \sigma_{T_L^{v_i}}^2}} = \frac{T_E^{v_i} - T_S^{v_i}}{\sigma_{T_E^{v_i}}}, \text{ neboť } \sigma_{T_S^{v_i}} = 0 . \text{ [3]}$$

Z rozdílu $T_E^{v_i} - T_S^{v_i}$ vyplývá, že hodnota u' vede k odhadu pravděpodobnosti překročení plánovaného termínu $T_S^{v_i}$:

$$P\left\{T_E^{v_i} \le T_S^{v_i}\right\} = \Phi(u) = \Phi\left(\frac{T_S^{v_i} - T_E^{v_i}}{\sigma_{T_E^{v_i}}}\right).$$

Jestliže $T_S^{v_i} > T_E^{v_i}$, je pravděpodobnost dodržení termínu $p > 0{,}5$.

- Jestliže $T_S^{v_i} = T_E^{v_i}$, je pravděpodobnost dodržení termínu $p = 0{,}5$.
- Jestliže $T_S^{v_i} < T_E^{v_i}$, je pravděpodobnost dodržení termínu $p < 0{,}5$.

3.4 Problematika metody Branch & Bound (Větve a meze)

Množství skutečných plánovacích problémů, které jsou nazývány kombinatorické úlohy optimalizace, má několik společných prvků. Optimalizují problémy, ale obvykle mají velké množství proveditelných řešení.

Metoda Branch & Bound je jedním z hlavních nástrojů řešící problémy diskrétní optimalizace. Hlavní myšlenkou metody je nalézt minimální nebo maximální hodnotu argumentu *x* v oblasti přípustných řešení. U minimalizace se jedná například o čas, u maximalizace zhodnocení, apod. Metoda Branch & Bound používá dva níže popsané nástroje.

Prvním z nich je větvení. Metoda rozebírá množinu řešení na menší podmnožiny řešení. Metoda je opakována rekurzivně pro každou podmnožinu a celkovým výsledkem je stromová struktura, která je nazývána např. „prastrom řešení“, „strom větve a meze“ i jinými podobnými termíny. Uzly stromu vyjadřují vytvořené podmnožiny.

Druhým nástrojem této metody je vytváření mezí. Jedná se o rychlý způsob nalezení dolní nebo horní hranice pro optimální řešení v dané podmnožině. Metoda odstraní ta řešení, která mají v případě hledání minimálních řešení vyšší hodnotu nežli mez, kterou si určíme jako prozatím nejvýhodnější řešení, a tyto možnosti jsou poté ignorovány.

3.4.1 Princip metody Branch & Bound

Nechť $E = \{S_1, S_2, S_3 \ldots, S_n\}$ je množina diskrétní optimalizace a *f* je funkce definována na této množině. Pokud je hledáno minimum, tak

je potřeba najít takovou podmnožinu E_m množině E, na které tato funkce dosahuje minimální hodnoty. [4]

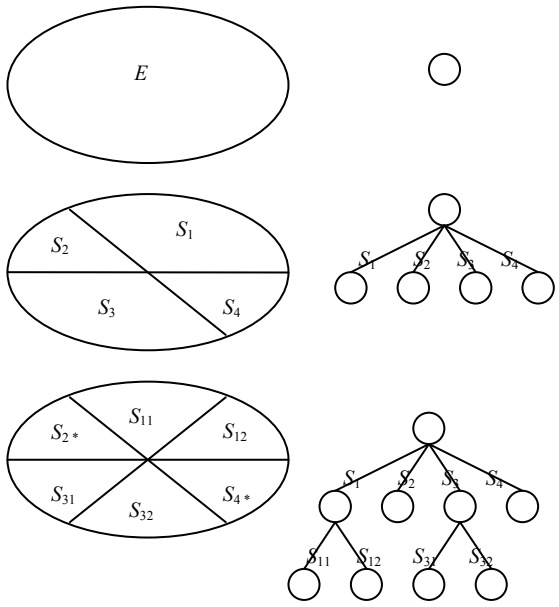
Máme vlastnost P_A takovou, že můžeme množinu E rozdělit na dvě podmnožiny $A, \overline{A}$, kde platí $A \cap \overline{A} = \{\}$, $A \cup \overline{A} = E$.

Najdeme spodní mez b_0 hodnot funkce f na množině E. Dále předpokládejme, že dokážeme určit i spodní meze $b_1 \geq b_0, b_1' \geq b_0$ funkce f na podmnožinách A, $\overline{A}$.

Vlastnosti P_B, P_C dovolují rozložit množinu E na dvě části. Pak vlastnostem $P_A \wedge P_B, \overline{P_A} \wedge P_B, P_A \wedge \overline{P_B}, \overline{P_A} \wedge \overline{P_B}$ odpovídají podmnožiny $A \cap B, \overline{A} \cap B, A \cap \overline{B}, \overline{A} \cap \overline{B}$. Každá z těchto podmnožin představuje vrchol grafu. Tyto jednotlivé vrcholy umožňují vytvořit prastrom řešení.

Postup při vytváření prastromu je následující:

- Vybereme vrchol, který je listem prastromu řešení a má nejnižší mez.
- S pomocí k vlastnosti můžeme získat další dva nové vrcholy, pro které určíme jejich dolní mez b.



Obrázek 5: Příklad větvení a vytváření mezí – Zdroj: [5]

Na obrázku 5 máme původně jednu množinu řešení E. Toto řešení tvoří uzel, který je kořenem prastromu. Množinu dle vlastností jednotlivých podmnožin rozdělíme na čtyři disjunktní podmnožiny řešení S_1, S_2, S_3, S_4. Vzniknou nám čtyři nové uzly představující nová řešení. Na uzlech, které jsou listy stromu zjistíme, že funkce f nabývá nejnižších hodnot na podmnožinách S_1 a S_3. Ostatní vrcholy (S_2 a S_4) nemají optimální řešení, hodnota funkce f nabývá vyšších hodnot než na podmnožinách S_1 a S_3 . Tyto vrcholy dále rozdělíme, dle jejich vlastností na další podmnožiny řešení S_{11} a S_{12}, respektive S_{31} a S_{32}. Zjistíme z visících vrcholů, kde nabývá hodnota funkce f nejnižších hodnot. Další postup je aplikován analogicky rekurzivním způsobem.

Prastrom řešení má tu vlastnost, že pokud sjednotíme všechny uzly, které jsou v dané fázi visícími vrcholy stromu, získáme původní množinu řešení E . Je-li výsledkem uzel, který je listem, potom funkce f nabývá na této množině minimální hodnotu.

4. Návrh metody a algoritmu řešení

Návrh metody a algoritmů je třeba řešit již v počátcích vytváření projektu. Program má řešit problém kritické cesty, metody PERT a řízení projektů, kde je nutné respektovat kapacitní omezení jednotlivých činností.

O implementaci síťového grafu bude pojednáno v kapitole 5.1. V programu je použito několik metod z teorie grafů. K řešení projektu s kapacitním omezením jednotlivých činností je použita zároveň metoda CPM a Branch & Bound.

Dále pro tyto metody jsou nutné některé algoritmy pro průchod grafem a předání potřebných dat pro výpočet kritické cesty, PERT a metody větvení a mezí. Pro příklad je uveden algoritmus pro topologické uspořádaní s úpravou pro acyklický síťový graf a algoritmus pro rozmístění zdrojů v síti.

4.1 Algoritmy pro výpočet metody kritické cesty a metody PERT

Obě metody využívají podobný postup pro svůj výpočet. Používají v postupu početní operace s jedním argumentem reprezentujícím hranu představující činnost. U metody CPM je tímto argumentem časový odhad, u metody PERT hodnota t_e , která reprezentuje očekávaný čas jako náhodnou veličinu, odvození je uvedeno v kap. 3.2.2.

4.1.1. Algoritmus topologického uspořádání

Tento algoritmus se užívá v případech, kdy je potřeba vykonat množinu činností, které je nutné vykonat postupně jednu po druhé, což znamená, že není možné vykonávat více činností najednou a vykonávání žádné činnosti nesmí být přerušeno činností jinou. Aplikace tohoto algoritmu je možná pouze na acyklické grafy.

Pro tento algoritmus si vytvoříme dvě posloupnosti. Jednu pro hrany a druhou bude tvořit posloupnost vrcholů.

Jako první vrchol vybereme vrchol v_1, do kterého nevede žádná hrana a všechny hrany vycházející z tohoto vrcholu odstraníme a vložíme je do příslušných posloupností. Zbývající graf je opět acyklický, tj. opět existuje vrchol, do něhož nevede žádná hrana. Takový vrchol označíme v_2 a odstraníme jej i všechny hrany, pro které je počátečním vrcholem; odstraněné hrany poté zařadíme do posloupností. Naznačený postup opakujeme, dokud nejsou uspořádány všechny vrcholy a hrany. Pokud bychom nechtěli během výpočtu měnit graf, lze se na daný postup dívat i tak, že vybíráme pouze vrchol, jehož předchůdci jsou již v posloupnosti zařazeni. Dále lze algoritmus ještě upravit tak, že pro každý vrchol grafu v_i udržujeme číslo $C(v_i)$ rovné počtu hran vcházejících do vrcholu v_i z vrcholů, které ještě nejsou zařazeny v posloupnosti. Tyto hodnoty postupně snižujeme a vrcholy, které mají $C(v_i) = 0$, ukládáme do seznamu M jako kandidáty pro zařazení do posloupnosti.

Algoritmus: [2]

1. krok: Pro všechny vrcholy $v_i \in V(G)$ položíme $C(v_i) = 0$. Pro všechny koncové vrcholy Kv(h) hran $h \in Y(G)$ provedeme $C(\mathrm{Kv}(h)) = C(\mathrm{Kv}(h)) + 1$. Pro všechny vrcholy v_i, kde $i \neq 0$, položíme ohodnocení vrcholu určující maximální délku času $U(v_i) = -\infty$. Pro počáteční vrchol tuto hodnotu nastavíme na $v_0 = 0$.

2. krok: Do množiny M vložíme všechny vrcholy v, v_i, pro které platí $C(v_i) = 0$.

3. krok: Je-li $M = \emptyset$, pak výpočet končí. Je-li $M \neq \emptyset$, odstraníme z M některý vrchol, označíme jej v_i a zařadíme jej na konec posloupnosti. Vrcholů.

4. krok: Pro každou hranu $h \in H^+(v_i)$ provedeme krok 4a a pak pokračujeme krokem 3.

5. krok: Zařadíme hranu h na konec posloupnosti hran, označíme $y = \mathrm{Kv}(h)$, kde Kv je koncový vrchol hrany, a

položíme $C(y) = C(y) - 1$. Jestliže nyní platí $C(y) = 0$, pak vrchol *y* zařadíme do seznamu *M*. Jestliže ohodnocení koncového vrcholu *U*(Kv(*h*)) hrany *h* menší než ohodnocení počáteční vrcholu *U*(Pv(*h*)) hrany *h* a jejím ohodnocením, tedy *U*(Kv(*h*)) < *U*(Pv(*h*)) + o(*h*), pak délka maximální cesty v daném vrcholu *U*(Kv(*h*)) = *U*(Pv(*h*)) + o(*h*).

Asymptotická složitost algoritmu je $O(m + n)$, kde *m* je počet hran a *n* je počet vrcholů.

Tohoto poznatku lze využít, a v programu jej bylo využito, k určení acykličnosti grafu. Vycházíme ze věty, která je vlastností acyklických grafů:

Nechť *G* je orientovaný graf. Pak následující podmínky jsou ekvivalentní:

1. *G* je acyklický.
2. *G* nemá smyčky a každá jeho silná komponenta má jen jeden vrchol.
3. Existuje topologické uspořádání vrcholů grafu *G*.
4. Existuje topologické uspořádání hran grafu *G*.

Z toho vyplývá, že pokud nejsou po dokončení algoritmu topologického uspořádání všechny vrcholy zařazeny v posloupnosti vrcholů, pak graf není acyklický.

4.2 Algoritmus pro rozmístění zdrojů v síti

Problém časového rozvrhu zdrojů v případě jejich omezení, spočívá v nalezení takového rozvrhu zdrojů, který minimalizuje celkovou dobu trvání projektu. Použitá metoda využívá kombinace metody větvení a meze (Branch & Bound) a metody kritické cesty (CPM).

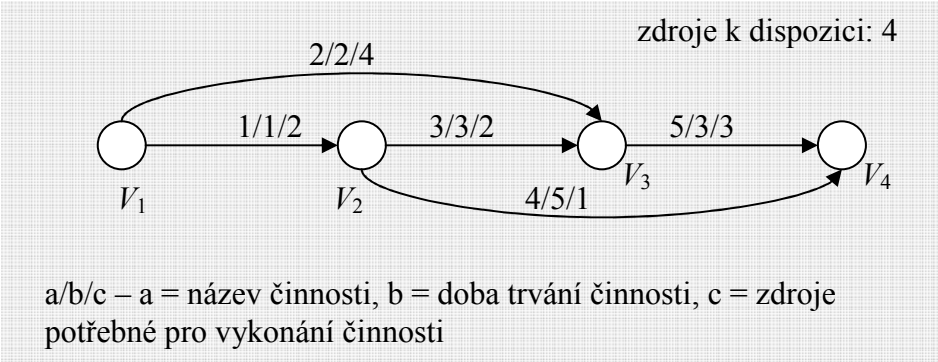
Velké množství literatury popisuje použití síťových grafů pro časovou analýzu a časový rozvrh projektu. Hlavním problémem v této

oblasti operačního výzkumu je určení optimálního časového rozvrhu činností, respektujícího skutečnost, že zdroje jsou omezeny.

Daný problém dále popíšeme pomocí konkrétního příkladu, který bude obsahovat program jako vzorový. Obrázek 6 a tabulka 1 zobrazují potřebné údaje pro daný příklad. Je brán v úvahu příklad, kde zdroje (pracovníci) mohou vykonávat libovolnou činnost. Lze rozlišovat dvě varianty postupu při řešení této úlohy:

1) Pokud je započata některá z činností, nelze její vykonávání přerušit a je nutné ji dokončit.
2) Jestliže je činnost rozpracována, může být její vykonávání pozastaveno a zdroje se mohou použít na jinou libovolnou činnost projektu.

Tato práce se zabývá druhou z těchto variant, to znamená, že lze kdykoliv rozpracovanou činnost pozastavit a uvolnit tak zdroje na vykonávání jiné činnosti projektu.



Obrázek 6: Příklad rozmístění zdrojů v síti

Činnost	**Počáteční vrchol**	**Koncový vrchol**	**Doba trvání**	**Zdroje**
1.	V_1	V_2	1	2
2.	V_1	V_3	2	4
3.	V_2	V_3	3	2
4.	V_2	V_4	5	1
5.	V_3	V_4	3	3

Tabulka 1: Rozmístění zdrojů

Pro daný příklad máme v libovolném čase k dispozici 4 pracovníky, kteří mohou vykonávat libovolnou činnost. Naším cílem je sestavit seznam časově seřazených činností tak, abychom minimalizovali dobu trvání celého projektu. Pro řešení použijeme metodu Branch & Bound na sestavení konstrukce prastromu řešení. Každý vrchol prastromu představuje podproblém, ve kterém jsou některé činnosti dokončené nebo částečně dokončené. Každá hrana prastromu představuje množinu činností, jak jdou za sebou v časové návaznosti jejich realizace. Rekurzivním rozvojem stromu dojdeme k optimálnímu řešení úlohy.

Prastrom, který představuje řešení našeho příkladu je uveden v příloze A. Každý vrchol je označen v horní části. Vrchol V_1 představuje původní projekt, respektive jeho zadání. Tento případ je uveden v příloze B v levém horním rohu. Pro ostatní vrcholy prastromu jsou uvedeny další grafy představující jednotlivé časové momenty. Projekt může začít jednou ze tří možností. Buď lze vykonat činnosti 1, 2 samostatně nebo obě činnosti 1 a 2 současně. Poslední případ ovšem potřebuje 6 pracovníků, ale k dispozici jsou jen 4. Tedy tuto variantu není možné provést. První dva realizovatelné případy znázorníme pomocí hran vycházejících z vrcholu V_1 do vrcholu V_3 a V_4. Čísla u hran udávají číslo činnosti a jejich pořadí ve vykonávání. Vrchol V_2 odpovídá podproblému bez činnosti 1, vrchol V_3 bez činnosti 2. V popisu vrcholu je v horní části jeho název, hodnota t_j představuje představuje čas, ve kterém může podproblém V_j začít. Číslo b_j je spodní hranice času dokončení celého projektu bez omezení zdrojů v případě, že budeme pokračovat po dané hraně.

Spodní hranici určíme tak, že neuvažujeme žádná omezení zdrojů. Pro vrchol V_1 to znamená čas dokončení celého projektu bez omezení zdrojů. Jedná se tedy o shodný výpočet jako u metody kritické cesty. To znamená, že celý projekt by bez omezení zdrojů trval 7 dní, takže $b_1 = 7$ a $t_1 = 0.$ Jestliže činnost 1 začne v čase 0 a čas pro její dokončení trvá 1 den, pak vrchol V_2 představující podproblém po dokončení činnosti 1 může začít v čase $t_2 = 1.$ Zbývající čas pro dokončení podproblému bez omezení zdrojů je 6 dní, tedy hodnota spodní hranice času pro dokončení

projektu je 7 dní. Činnost 2 začne v čase 0, pak v čase 2 zbývá vyřešit podproblém V_3. Čas pro ukončení podproblému V_3 bez zdrojových omezení je roven 7 dnům. Z toho vyplývá $t_3 = 2$ a $b_3 = 9$.

Poté vybereme jeden z visících vrcholů s nižší spodní hranicí. Tímto vrcholem je V_2. Pro podproblém V_2 jsou přípustné pouze tyto varianty: buďto samostatně vykonané činnosti 2, 3, 4, nebo činnosti 3 a 4 současně. Tyto možnosti jsou prezentovány hranami vedoucích k vrcholům V_4, V_5, V_6 a V_7.

Tam, kde několik činností probíhá současně, hrana končí, když je kratší z činností dokončena. To znamená, že podproblém V_7 má dokončeny činnosti 1, 3 a z činnosti 4 má dokončeny 4 dny.

Když jsme se dostali k vrcholu V_2 prastromu, vzniká potřeba volby vrcholu, který budeme větvit. V praxi je vhodné větvit dále ten vrchol, který je následníkem právě zkoumaného vrcholu a má přitom nejmenší spodní hranici ze všech vrcholů, které jsou listy prastromu. Situace vzniklá v našem příkladu poukazuje na vrcholy V_3, V_4, V_5 a V_7. Tyto vrcholy mají spodní hranici 9 dní. Vybereme z těchto vrcholů ten, který má vykonaných nejvíce činností, to znamená, že daný vrchol má nejvíce předcházejících hran vedoucích až do prvního vrcholu V_1. Vybereme tedy vrchol V_7, kde podproblém má dále dvě možnosti. Můžeme dokončit činnost 4 samostatně, nebo zpracovat kompletně činnost 2. Tím získáme další podproblémy V_8, V_9. Pak opět prohledáme všechny visící vrcholy s nejnižší spodní hranicí a nejvyšším počtem splněných činností. Tímto vrcholem je podproblém V_9. Zde nastávají 3 možnosti. Buď vykonáme činnost 5 samostatně, nebo dokončíme činnost 4. Třetí možností je dokončit činnost 4 a současně započít činnost 5. Zde je vidět, že nemá smysl brát v úvahu první dvě možnosti vykonávání činností odděleně, když mohou být vykonávány současně. Vznikne nám tedy pouze jeden vrchol V_{10}. V celém projektu nám nyní zbývá dokončit poslední činnost 5, která je již rozpracována. Ve vrcholu V_{11}, dostáváme hodnoty $b_{11} = 9$ a $t_{11} = 9$. Znamená to, že jsme nalezli jedno z možných řešení projektu.

Časový plán pro zpracování projektu je tedy následný. Projekt začneme vykonáním činnosti 1 v čase 0. Dále provedeme současně činnost 3 a část činnosti 4 po dobu 3 dnů. Navážeme činností 2 v čase 4. Pak zpracujeme zároveň dokončení činnosti 4 a rozpracování činnosti 5. Posledním krokem je dokončení činnosti 5 v čase 8. Výsledný čas pro dokončení celého projektu je tedy 9 dní.

Poté můžeme ještě prověřit všechny visící vrcholy, ve kterých nejsou ukončeny všechny činnosti a jejichž spodní hranice není vyšší než výsledný čas pro dokončení projektu. Výsledky všech řešení jsou uvedeny v tabulce 2.

Varianta 1					
Zpracovávaná činnost (činnosti)	1	3, 4´	2	4´´, 5´	5´´
Začátek vykonávání činnosti (činností)	0	1	4	6	8
Varianta 2					
Zpracovávaná činnost (činnosti)	1	2	3, 4´	4´´, 5´	5´´
Začátek vykonávání činnosti (činností)	0	1	3	6	8
Varianta 3					
Zpracovávaná činnost (činnosti)	2	1	3, 4´	4´´, 5´	5´´
Začátek vykonávání činnosti (činností)	0	2	3	6	8

Tabulka 2: Výsledky rozmístění zdrojů v síti

5. Počítačová implementace v jazyku C++

V první fázi zpracování daného problému v implementační části je důležité navrhnout vhodnou datovou strukturu, která bude daný síťový graf obsluhovat základními metodami pro práci s grafy a bude jej zobrazovat pro jednoduchou a přehlednou manipulaci. Pro daný problém byl zvolen abstraktní datový typ síť.

Práce byla vytvořena v Microsoft Visual Studiu 2005 v MFC (Microsoft Foundation Class).

5.1 Návrh datových struktur

5.1.1 Reprezentace síťového grafu v paměti

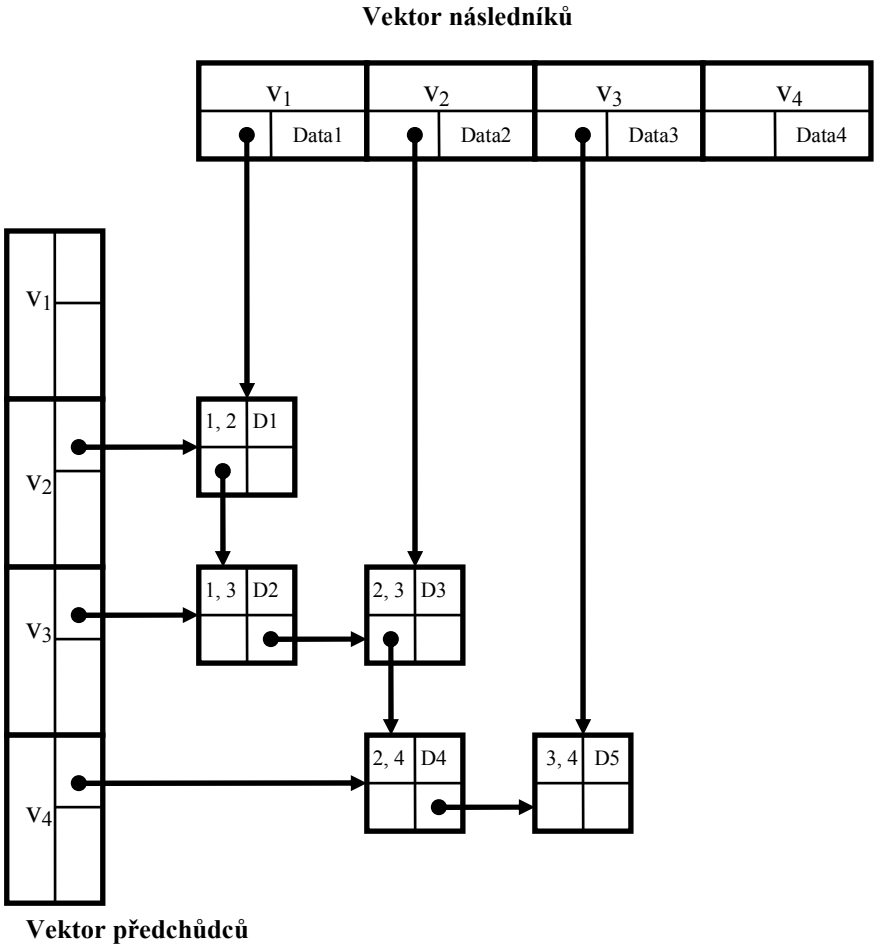
Síťové grafy nabývají velkého množství prvků, na kterých jsou velmi často prováděny výpočty. Je tedy potřeba vytvořit datovou strukturu, která bude odpovídat kompromisu mezi paměťovou a časovou (asymptotickou) složitostí.

V této práci je zvolena pro zobrazení grafu křížová reprezentace neboli řídká matice. Jedná se o paměťově méně náročnou reprezentaci grafu, která umožňuje přístup oběma směry. Řídké matice jsou speciální třídou matic, jejichž většina prvků se rovná nule, a proto umožňuje úspornější uložení v paměti počítače. Na uložení grafu v klasické matice $m \times n$ je potřeba $m \cdot n$ hodnot, kdy některé prvky matice zůstávají během výpočtů nevyužity a zbytečně zabírají v paměti místo. Na uložení řídké matice rozměrů $m \times n$ s k nenulovými reálnými hodnotami je třeba:

- 2 celočíselné hodnoty na uložení dimenze matice
- $2k$ celočíselných hodnot na uložení indexů nenulových prvků
- k reálných hodnot na uložení hodnot nenulových prvků

Je-li r počet bytů potřebných na uložení reálného čísla a z počet bytů na uložení celého čísla, pak uložení matice jako řídké se vyplatí v případě, že $2(k+1)z + kr < mnr$. [1]

Tato implementace umožňuje přístup jak ke svým předchůdcům, tak ke svým následníkům, což je pro naše výpočty související s častým procházením grafu výhodné. Pro přístup k vrcholům se udržují dvě prvotní struktury (pro přístup k následníkům a pro přístup k předchůdcům). Druhotná struktura je pouze jedna, ale umožňuje pohyb oběma směry k předchůdcům a také k následníkům. Toto řešení může být implementováno jako pole-seznam × pole-seznam nebo seznam-seznam × seznam-seznam. Práce používá první z těchto případů.



Obrázek 7: Řídká matice – reprezentace v paměti

Prvotní struktury pole pro následníky (předchůdce) uchovávají jednotlivé vrcholy. Obsahují jednotlivé klíče vrcholů, ukazatel na jejich data (D1, D2, …) a ukazatel na první hranu, která vychází z daného uzlu, což je ve skutečnosti hlava seznamu hran. Druhotná struktura je lineární seznam s hlavou, jehož hlava je uložena ve struktuře vrcholu. Obsahuje čtyři základní údaje; klíče vrcholů (následníka a předchůdce) udávající

jednoznačně hranu počátečním a koncovým vrcholem, data dané hrany dané těmito vrcholy, ukazatele na následníky a předchůdce (tzn. veškeré hrany vycházející a vcházející do vrcholu). Z obrázku 7 vyplývá, že vrchol v_1 je počáteční vrchol grafu, jelikož do něj nevchází žádná z hran, zatímco vrchol v_2 je koncovým vrcholem, protože z něj žádná hrana nevede. Obrázek reprezentující zobrazení řídké matice v paměti odpovídá grafu z obrázku 6 v kapitole 4.2.

5.1.2 Reprezentace prastromu řešení v paměti

Pro výsledné zobrazení prastromu řešení je použita stejná datová struktura jakou používá síťový graf. Jde pouze o transformaci grafu na příslušný k–cestný strom, která je řešena nad abstraktním datovým typem v obslužné části programu.

5.2. Problém acykličnosti grafu

Acykličnost grafu lze řešit několika způsoby. V programu je implementován způsob ověření existence cyklu pomocí algoritmu pro topologické uspořádání. Tento algoritmus je popsán v kapitole 4.1.1 i s úpravami pro tuto aplikaci.

Na začátku programu se ověří, zda existují počáteční a koncový vrchol síťového grafu. To znamená, že do tohoto vrcholu nevchází žádná hrana, respektive z něj žádná nevychází. Algoritmus topologického uspořádání postupně odebírá ty vrcholy grafu, do kterých žádná hrana nevede. Po jejich odstranění je graf opět acyklický.

Pokud se ovšem stane, že není nalezen žádný vrchol, do kterého nevchází žádná z hran a ještě všechny vrcholy nebyly zpracovány, pak to znamená, že se v grafu vyskytnul cyklus a tedy neexistuje topologické uspořádání grafu.

5.3. Implementace algoritmu pro rozmístění zdrojů v síťovém grafu

Tento problém lze řešit několika způsoby. Můžeme buď vytvořit další instanci grafu, který bude přesnou kopií původního grafu nebo vkládat do pomocných struktur právě zpracovávané prvky síťového

grafu. Mnohem šetrnější k paměti a pro rychlejší zpracování je vytvoření vhodných pomocných struktur. Toto řešení je použito i v této práci. V této kapitole si zjednodušeně popíšeme algoritmus.

Nejdříve si vytvoříme posloupnosti hran *M*, *Z, N*.

1. krok: Nejprve vložíme kořen prastromu řešení. Tímto kořenem je počáteční vrchol síťového grafu. Nastavíme jeho první atribut určující maximální celkový čas pro dokončení projektu bez omezení zdrojů. Pro tyto účely je použit upravený algoritmus na výpočet kritické cesty, s tím, že je možné mít více počátečních vrcholů. Aktuální čas všech doposud vykonaných činností je nastaven na 0.

2. krok: Vybereme všechny vrcholy, které jsou listy prastromu řešení. Z těchto vrcholů vezmeme ten s nejnižším aktuálním časem $V_{\text{aktuální}}$. Pokud má vrchol hodnotu maximálního času pro dokončení celého projektu bez omezení zdrojů rovnu 0, pak vybereme jiný vrchol, který má stejnou hodnotu aktuálního času jako vrchol $V_{\text{aktuální}}$. Jestliže takový vrchol neexistuje, algoritmus končí.

3. krok: Projdeme všechny činnosti, které předcházely podproblému vybraného vrcholu $V_{\text{aktuální}}$. Jestliže hrana byla vykonána celá, vložíme je do posloupnosti *M* a odstraníme ji z grafu. Pokud hrana byla jen částečně vykonána, vložíme ji do posloupnosti *Z* a upravíme její skutečnou dobu trvání zkrácenou o již vykonanou část činnosti.

4. krok: Projdeme všechny vrcholy grafu a vybereme ty, které nemají vcházející hranu a zároveň z nich alespoň jedna hrana vychází. Všechny vycházející hrany z těchto vrcholů vložíme do posloupnosti hran *N*.

5. krok: Hrany z posloupnosti *N* postupně projdeme a odstraníme vždy jednu z grafu. Vložíme do prastromu

vrchol, vypočítáme pro něj aktuální čas projektu a celkovou dobu pro dokončení podproblému bez zdrojových omezení. Odebranou hranu opět vrátíme do grafu.

6. krok: Vytvoříme kombinace všech hran z posloupnosti *N*, pro které je dostatek zdrojů k jejich vykonání. V této práci jsou řešeny pro zjednodušení pouze kombinace dvou hran. Pokud mají hrany shodnou dobu trvání, pokračujeme krokem 6a. Krok 6b vykonáme v případě, že tyto časy jsou rozdílné.

6a. krok: Odstraníme obě hrany z grafu. Vložíme do prastromu vrchol, vypočítáme pro něj aktuální čas projektu a celkovou dobu pro dokončení podproblému bez zdrojových omezení. Odebrané hrany opět vrátíme do grafu. Pokračujeme krokem 7.

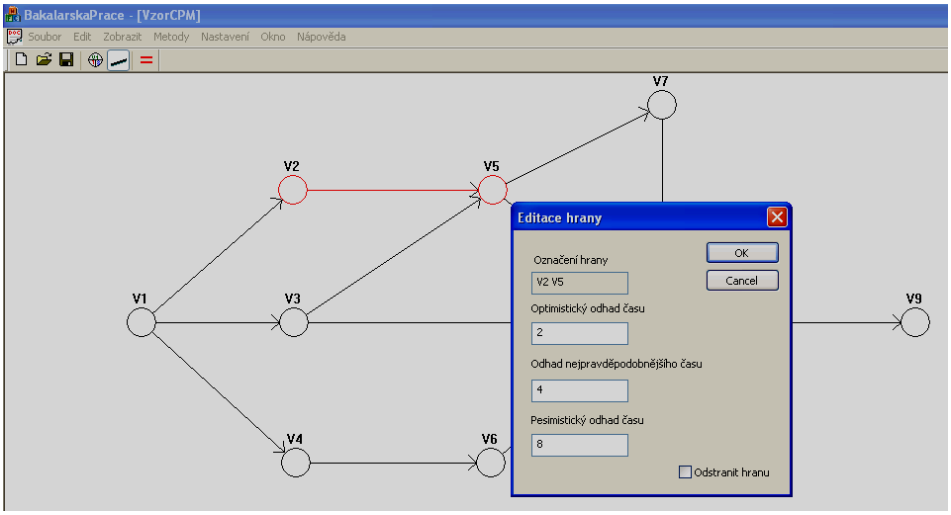
6b. krok: Vybereme kratší z hran, její hodnotu si uchováme v pomocné proměnné a odstraníme ji z grafu. Druhou hranu o tuto hodnotu zkrátíme. Vložíme do prastromu vrchol, vypočítáme pro něj aktuální čas projektu a celkovou dobu pro dokončení podproblému bez zdrojových omezení. Vrátíme zpět kratší z hran do grafu a ke zkrácené (původně delší hraně) přičteme hodnotu, kterou jsme si uchovali v pomocné proměnné. Pokračujeme krokem 7.

7. krok: Všechny hrany z posloupností *M* vložíme zpět do grafu. Hrany z posloupnosti *Z* editujeme na jejich původní hodnoty, které jsou uloženy v této posloupnosti. A pokračujeme krokem 2.

6. Řešení praktické aplikace

Jak již bylo řešeno v kapitole 5, program zpracovává data zadaná pomocí síťového grafu. Počítá metodu kritické cesty, metodu PERT a hledá vhodné rozmístění zdrojů v síti. Celý program je naprogramován ve Visual Studiu 2005 v jazyce C++ jako MFC aplikace (Microsoft Fundation Class).

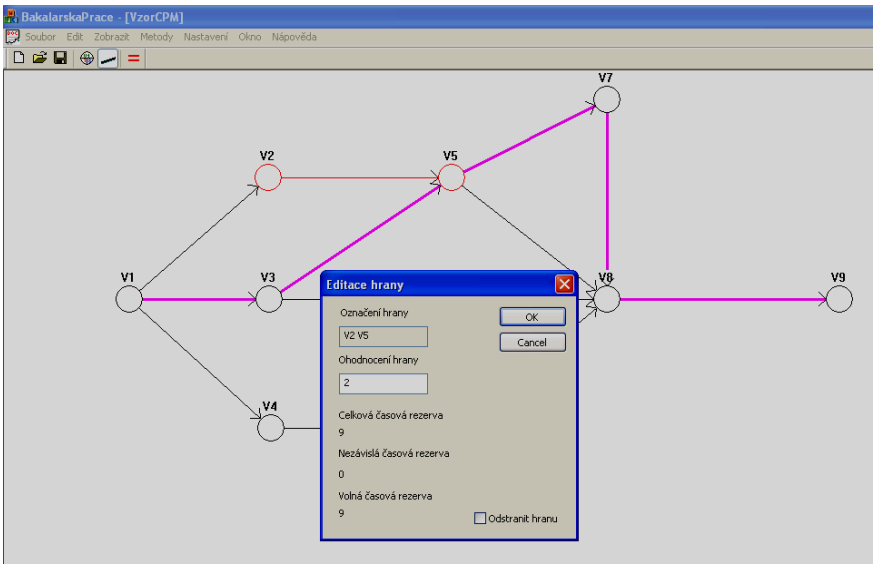
Při vytváření síťového grafu se zadávají dle selekce buď vrcholy nebo hrany grafu. Editace jednotlivých vrcholů nebo hran se provádí pomocí dvojkliku na daný objekt. Vrcholy i hrany lze odstranit při jejich editaci zaškrtnutím položky odstranit.



Obrázek 8: Editace objektů grafu v aplikaci

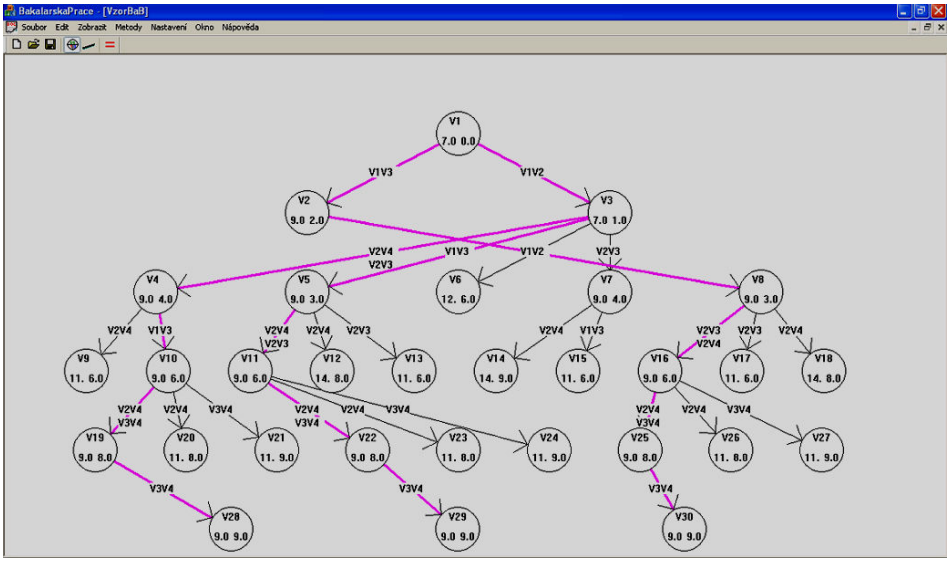
Libovolný graf lze uložit do souboru například pomocí klávesové zkratky CTRL+S a opětovně jej i otevřít. V nastavení programu si můžeme zvolit velikost zobrazovaných vrcholů v pixelech a kolik zdrojů máme pro daný projekt k dispozici. V metodách si můžeme určit, zda chceme řešit metodou CPM, PERT nebo rozmístění zdrojů v síti. Každá z metod má své specifické nastavení, takže je nutné si jej nastavit již před vyplňováním hodnot potřebných k výpočtu.

Po dokončení zadávání síťového grafu můžeme provést výpočet. Tato položka je buď v metodách nebo přímo v nástrojové liště.



Obrázek 9: Zobrazení výsledků metody CPM

Uživatel se nezabývá definicí počátečního a koncového vrcholu, které jsou v síťovém grafu zastoupeny pouze jednou. Tyto vrcholy si aplikace určí implicitně sama. Předpokládá se, že uživatel má nějakou představu o dané problematice a graf bude zadán ve správném formátu. Pokud se pokusí zadat více vrcholů, které mohou být považovány za počáteční nebo konečné, bude mu zobrazeno příslušné chybové hlášení. Aplikace je ošetřena proti vkládání násobných hran a smyček, které nejsou v síťovém grafu přípustné. Program dále řeší problém acykličnosti grafu. Podrobnější popis acykličnosti grafu je kapitole 5.1.



Obrázek 10: Prastrom řešení v aplikaci

Závěr

Práce se zabývá teoretickými základy síťové analýzy a jejich praktickým použitím. Soustřeďuje se zejména na problematiku metody kritické cesty (CPM), metody PERT a metody Branch & Bound (Větve a meze).

Pomocí metod CPM a Branch & Bound je řešena problematika rozmístění zdrojů v síťovém grafu za předpokladu, že jednotlivé zdroje mohou být použity na libovolnou činnost. Praktické použití těchto metod ilustruje uvedený příklad.

Praktická část práce se zabývá tvorbou aplikace implementující zmíněné metody. Jsou zde popsány datové struktury použité v této aplikaci, stěžejní algoritmy a případné problémy, které mohou při běhu aplikace nastat.

Poslední kapitola prezentuje výslednou aplikaci vytvořenou v jazyku C++. Aplikace řeší metodu kritické cesty a metodu PERT na libovolném síťovém grafu. Výstupem aplikace je prastrom řešení. Vzhledem k nedostatku času při řešení této části práce zobrazuje aplikace v prastromu i podproblémy, které jsou při manuálním řešení úlohy díky lidskému úsudku automaticky ignorovány. V důsledku toho může při větších síťových grafech nabývat prastrom řešení obrovských rozměrů.

Výsledkem práce je funkční aplikace implementující metody CPM, PERT a Branch & Bound.

SEZNAM LITERATURY

[1] BARTKO, R., MILLER, M. *MATLAB I. Algoritmizácia a riešenie úloh.* Trenčín: Digital Graphic. ISBN 80-968337-3-1.

[2] DEMEL, J. *Grafy a jejich aplikace*. Praha: Academia, 2002. ISBN 80-200-0990-6.

[3] KLUSOŇ, V. *Kritická cesta a PERT v řídící praxi*. Praha: SNTL – Nakladatelství technické literatury, 1973.

[4] VOLEK, J. *Operační výzkum I*. Pardubice: Univerzita Pardubice, leden 2002. ISBN 80–7194–410–6.

Elektronické dokumenty

[5] CLAUSEN, Jens. Branch and Bound Algorithms – Principles and Examples [online]. Kodaň: University of Copenhagen, 1999 [cit. 2007-05-05]. Dostupný z WWW: http://www.imada.sdu.dk/Courses/DM85/TSPtext.pdf

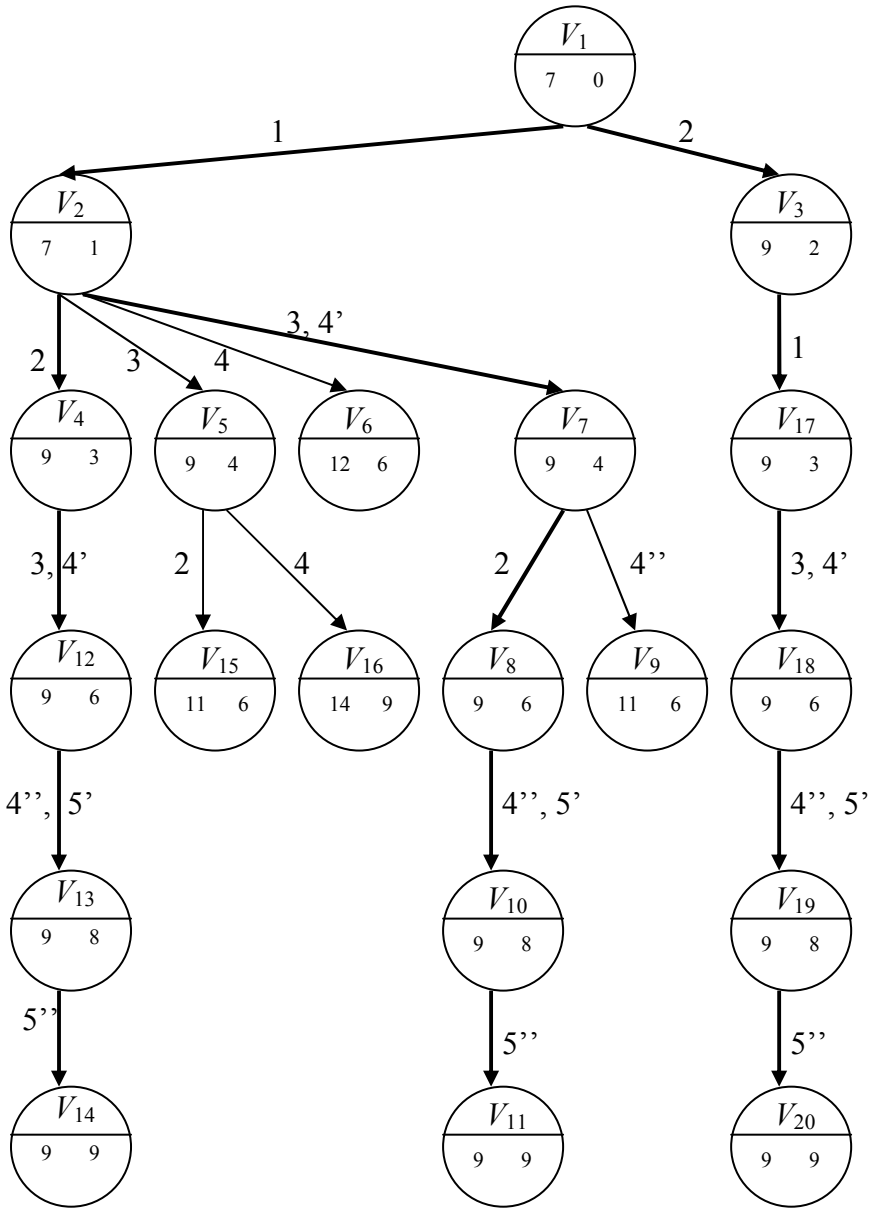
[6] *Wikipedia, otevřená encyklopedie* [online]. 2007 [cit. 2007-05-05]. Dostupný z WWW:

http://cs.wikipedia.org/wiki/Asymptotick%C3%A1_slo%C5%BEitost.

SEZNAM TABULEK

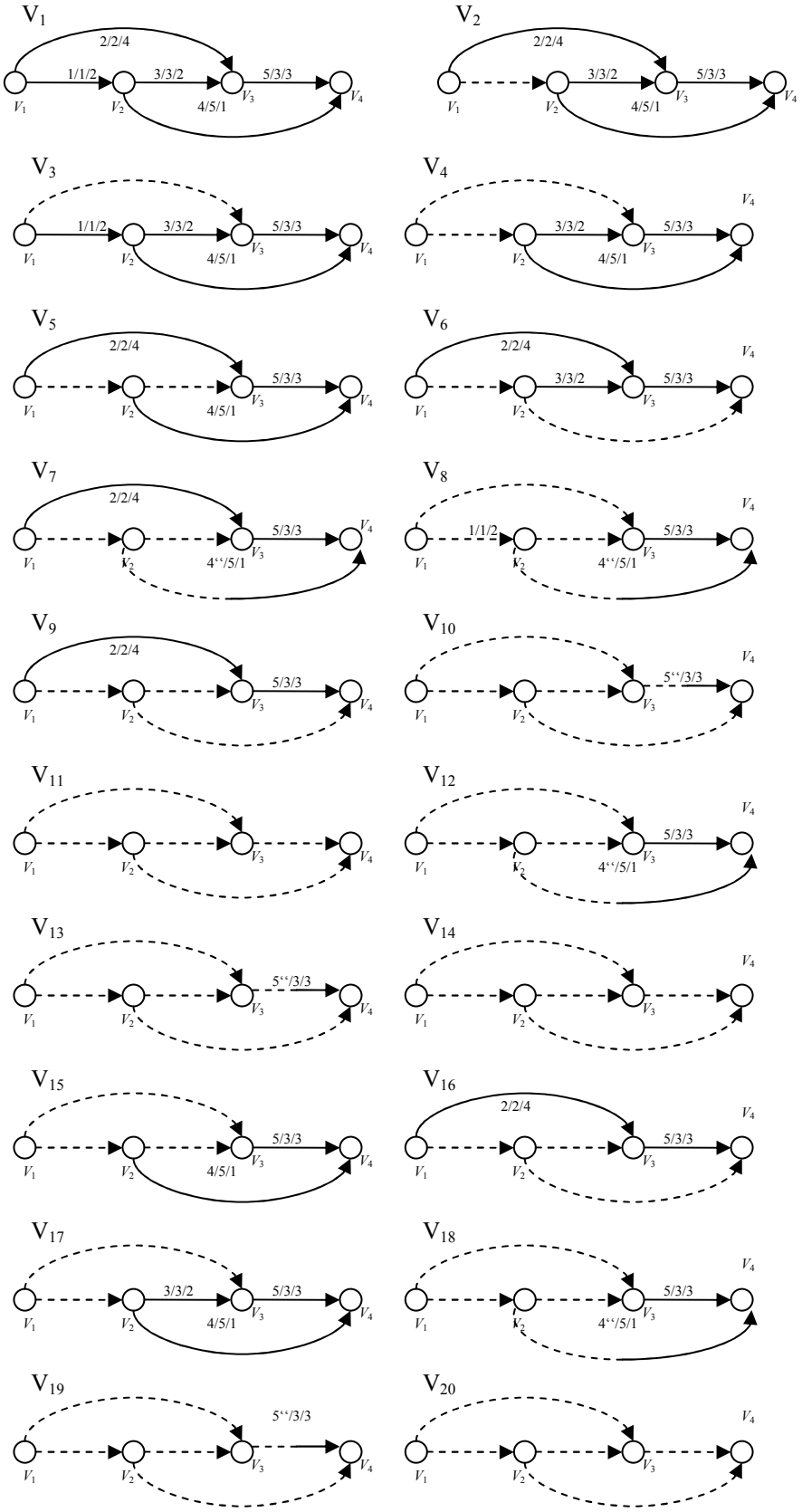
SEZNAM OBRÁZKŮ

Příloha A



Obrázek – Prastrom řešení k přikladu z kapitoly 4.2

Příloha B



Dokončené činnosti (nebo jejich části) jsou vyznačeny přerušovanou čarou.

Obrázek – Grafy podproblémů k příkladu z kapitoly 4.2

ÚDAJE PRO KNIHOVNICKOU DATABÁZI

Název práce	Počítačová podpora řízení rozsáhlých projektů
Autor práce	Martin Šváha
Obor	Informační technologie
Rok obhajoby	2007
Vedoucí práce	Doc. Ing. Josef Volek, CSc.
Anotace	Analýza rozmístění zdrojů v síťovém grafu při řešení praktické aplikace.
Klíčová slova	CPM, PERT, Branch and Bound, historie CPM, historie PERT