VYSOKÁ ŠKOLA EKONOMICKÁ V PRAZE

FAKULTA INFORMATIKY A STATISTIKY

Hlavní specializace: Matematické metody v ekonomii

Název diplomové práce

Testovanie softvéru pre zmiešané celočíselné programovanie

Diplomant:	Daniela Kmetonyová
Vedoucí diplomové práce:	Prof. Ing. Josef Jablonský, CSc.

Prohlášení:

Prehlasujem, že som diplomovú prácu na téma „Testovanie softvéru pre zmiešané celočíselné programovanie“ spracovala samostatne. Všetkú použitú literatúru a ďalšie podkladové materiály uvádzam v zozname použitej literatúry.

V Prahe, 5.5.2008. Daniela Kmetonyová

Poděkování:

Rada by som poďakovala Prof. Ing. Josefovi Jablonskému, CSc. za jeho pomoc, ochotu a užitočné rady a pripomienky. Moja vďaka patrí aj mojim rodičom za ich podporu počas celej doby môjho štúdia a pomoc pri gramatickej korektúre tejto práce.

1. ÚVOD

V dnešnej dobe charakteristickej rýchlym vývojom výpočtovej techniky a softvérových produktov rastú prirodzene aj požiadavky užívateľov na rýchlosť a kvalitu týchto produktov. Spotrebiteľským svetom vládne reklama a každý výrobca prezentuje svoj výrobok ako ten najlepší. Rovnako je tomu aj na trhu s optimalizačným softvérom.

Časy, kedy sa optimalizačné úlohy riešili na veľkých sálových počítačoch pomocou vkladania diernych štítkov, sú už dávno preč. Dnes je možné úlohy veľkých rozmerov, ktorých výpočet by predtým trval niekoľko týždnov, riešiť na bežných osobných počítačoch, a to za podstatne kratší čas. Postupným vývojom výpočtovej techniky a implementovaných postupov a metód riešenia úloh, sa rozmery riešených úloh postupne zväčšovali a výpočtový čas sa skracoval. Postupom času pribúdalo spoločností, ktoré sa venovali vývoju optimalizačného softvéru a snažili sa implementovať najnovšie vedomosti a poznatky, aby udržali svoj produkt čo najvýkonnejší a pre zákazníka najatraktívnejší.

Dnes je ponuka optimalizačných softvérom pomerne široká. Medzi najznámejšie produkty patria softvéry ako CPLEX, Xpress-MP, MOSEK, LINDO, FortMP, OML atď. Vo svojej diplomovej práci budem porovnávať produkty CPLEX od spoločnosti ILOG Optimalization, Xpress od spoločnosti Dash Optimization a MOSEK od spoločnosti MOSEK. Tieto tri optimalizačné softvéry som si vybrala pre svoje testovanie z dôvodu, že patria medzi momentálne špičky na trhu s optimalizačným softvérom a jednak z dôvodu ich dostupnosti v rámci školskej licencie.

Cieľom mojej diplomovej práce, je otestovať vybraný optimalizačný softvér z hľadiska rýchlosti výpočtu a presnosti riešenia úloh zmiešaného celočíselného programovania a užívateľskej prístupnosti ovládania softvéru a na základe získaných výsledkov tieto optimalizačné softvéry medzi sebou porovnať. Aj napriek tomu, že každý výrobca utajuje svoje „know-how", pokúsim sa priblížiť fungovanie a použité postupy pri výpočte každého z testovaných softvérov, a to hlavne s dôrazom na riešenie zmiešaných celočíselných úloh. K testovaniu použijem úlohy z knižnice MIPLIB 2003, ktoré sú používané ako štandard pre porovnávanie výkonnosti optimalizačných softvérov.

V prvej kapitole sa venujem vývoju a teoretickým základom zmiešaného celočíselného programovania s bližším popisom niektorých významných metód riešenia týchto úloh. V tejto kapitole je popísaný aj formát MPS, ktorý tvorí dnes celosvetový štandard pre zadávanie a archiváciu úloh lineárneho a zmiešaného celočiselného programovania a ktorý bol použitý pri testovaní ako vstupný formát testovacích úloh. V úvode ďalšej kapitoly popisujem metódu testovania, ktorú som použila. Kapitola pokračuje popisom testovaných softvérov CPLEX, Xpress a MOSEK, s dôrazom na možnosti riešenia zmiešaných celočíselných úloh. V poslednej kapitole porovnávam testované softvéry z rôznych hľadísk a komentujem dosiahnuté výsledky.

2. ZMIEŠANÉ CELOČÍSELNÉ PROGRAMOVANIE

2.1 FORMULÁCIA ÚLOHY MIP

Zmiešané celočíselné programovanie (mixed integer programming, MIP) je špeciálny prípad lineárneho optimalizačného problému, kde niektoré z premenných sú celočíselné a niektoré nie. Úlohu MIP zapisujeme v tvare:

$$\min\{cx + dy : Ax + Dy \le b, x \in Z_+^n, y \in R_+^p\},$$

kde:

- Z_+ je množina celých nezáporných čísiel,
- R_+ je množina reálnych nezáporných čisel,
- $cx + dy$ je lineárna funkcia, ktorú sa snažíme minimalizovať a nazývame ju *účelová funkcia*,
- $Ax + Dy \le b$ sú obmedzujúce podmienky,
- x a y označujú vektory premenných úlohy,
- A je matica $m \times n$ koeficientov m obmedzení a n celočíselných premenných,
- c je n-členný vektor *cien* celočíselných premenných,
- d je p-členný vektor *cien* neceločíselných premenných,
- D je matica $m \times p$ koeficientov m obmedzení a p neceločíselných premenných,
- b je m-členný vektor pravých strán.

Ďalej budeme predpokladať, že všetky parametre (c, d, A, D, b) sú racionálne čísla, čo nie je pre prax obmedzujúce a navyše pri práci s počítačom ani nie je možné s iracionálnymi číslami počítať.

$$S = \{x \in Z_+^n, y \in R_+^p : Ax + Dy \le b\},$$

kde S je množina prípustných riešení, ktorá obsahuje body (x, y), ktoré splňajú sústavu obmedzení v tvare $Ax + Dy \le b$, kde zložky vektoru x sú čísla nezáporné a celočíselné a zložky vektoru y sú iba čísla nezáporné.

V úlohe MIP je možné bez straty na obecnosti definovať obmedzujúce podmienky ako rovnice pomocou doplnkových premenných.

Je možné riešiť aj maximalizačnú úlohu, a to tak, že vynásobíme účelovú funkciu číslom -1.

Špeciálny prípad MIP, kde všetky celočíselné premenné môžu nadobudnúť hodnotu len 1 alebo 0 sa nazýva zmiešané bivalentné programovanie (mixed-binary programming, 0 – 1 MIP):

$$\min\{cx + dy : Ax + Dy \le b, x \in B^n, y \in R_+^p\},\ B = \{0,1\}.$$

Je možné konvertovať úlohu MIP na úlohu 0-1 MIP pokiaľ množina prípustných riešení je obmedzená. Každú celočíselnú premennú je totiž možné nahradiť binárnym vyjadrením hodnoty tejto premennej.[1]

Podmienka celočíselnosti, typická pre úlohu MIP, je to, čo mení konvexný problém na problém nekonvexný, a teda po formálnej stránke na úlohu NP-ťažkú (nondeterministic polynomial-time hard).[2] Toto je dôvod, prečo neexistuje polynomiálny algoritmus pre riešenie úloh MIP, a preto museli byť vytvorené špeciálne algoritmy.

Typickými príkladmi úloh MIP z ekonomickej praxe sú napríklad úloha batohu (Bin Packing Problem), úloha optimálneho rozmiestnenia zariadenia (Plant Location Problem), pokrývací problém (Set-Covering Problem), kontajnerový problém (Packing Problem), atď. S využitím úloh MIP sa často stretávame aj v iných teoretických a praktických oboroch tejto reality.

2.2 METÓDY RIEŠENIA ÚLOH MIP

2.2.1 Metóda vetvenia a medzí

Metóda vetvenia a medzí (Branch-and-bound, B&B) je univerzálna metóda na riešenie úloh zmiešaného celočíselného programovania a celočíselného programovania. Je možné ju formulovať aj pre riešenie špeciálnych úloh ako je napríklad úloha bivalentného programovania, úloha batohu a ďalšie.

Tento princíp bol po prvýkrát použitý autorkami A. H. Land and A. G. Doig v roku 1960.[3] Algoritmus založený na tejto metóde je nepolynomiálny, čo sa môže v praxi prejaviť tým, že výpočet úlohy nemusí prebehnúť v čase, ktorý máme k dispozícii pre výpočet. V tomto prípade je výpočet prerušený a metóda poskytuje suboptimálne riešenie s určitým odhadom odchylky od optimálneho riešenia.. Konkrétnu dobu trvania výpočtu metódou B&B je možné ovplyvniť určitými modifikáciami v algoritme, tak aj vhodnou formuláciou úlohy.[4]

Majme úlohu zmiešaného celočíselného programovania

$$\min\left\{cx + dy : Ax + Dy \le b, x \in Z_+^n, y \in R_+^p\right\}$$

a príslušnú množinu pripustných riešení

$$S = \left\{x \in Z_+^n, y \in R_+^p : Ax + Dy \le b\right\}.$$

Tento algoritmus začína lineárnou relaxáciou úlohy MIP, čo znamená riešiť úlohu lineárneho programovania (LP) získanú z pôvodnej úlohy MIP vynechaním všetkých podmienok celočíselnosti. Ako výsledok môžeme dostať:

---

[1] PELIKÁN, J. *Diskrétní modely v operačním výzkumu*. 1 vyd. Praha : Professional Publishing, 2001. str. 9-11. ISBN 80-86419-17-7.

[2] BIBXY, R.E. *MIP: THEORY AND PRACTICE*. str. 8. Dostupný z <http://www.ilog.com/products/optimization/tech/research/mip.pdf>, cit. 07-12-08.

[3] KOVÁCS, L.B. *Combinatorial methods of discrete programming*. 2 vyd. Budapest : AKADÉMIAI KIADÓ 1980. str. 57. ISBN 963-05-2004-4.

[4] PELIKÁN, 2001, str. 91.

a) Úloha LP nemá prípustné riešenie, čo znamená, že ani pôvodná úloha MIP nemá prípustné riešenie.

b) Úloha LP má prípustné riešenie, ale nie sú splnené podmienky celočíselnosti. Riešenie úlohy MIP sme zatiaľ nenašli.

c) Úloha LP má prípustné riešenie a zároveň sú splnené podmienky celočíselnosti. Našli sme optimálne riešenie pôvodnej úlohy MIP.

d) V prípade, že lineárna relaxácia je neobmedzená, potom úloha MIP je buď neobmedzená alebo nemá prípustné riešenie.[5] Tento prípad sa nevyskytuje príliš často, a pokiaľ sa vyskytne, je to v priebehu prvej relaxácie. Najjednoduchším riešením je modifikácia úlohy pridaním dodatočných obmedzení napríklad vo forme sumy všetkých premenných s veľmi vysokou hodnotou pravej strany.[6]

V prípadoch a) a c) je úloha MIP vyriešená. V ďalších dvoch prípadoch použijeme algoritmus metódy vetvenia a medzí, ktorý nasleduje takzvanú "divide-and-conquer" stratégiu, kde množina prípustných riešení úlohy MIP je rozdelená na podmnožiny, na ktorých sa následne hľadá optimum pôvodnej úlohy. Algoritmus si udržuje listinu L aktívnych podúloh, ktoré tvoria optimalizačné problémy nad týmito podmnožinami.

Nech $MIP(k)$ označuje podúlohu *k*, pôvodnej úlohy MIP. Hodnota účelovej funkcie akéhokoľvek prípustného riešenia $MIP(k)$ nám poskytuje horný odhad optimálnej hodnoty účelovej funkcie pôvodnej úlohy MIP. Hodnotu účelovej funkcie najlepšieho prípustného riešenia, ktoré sme do určitého okamžiku výpočtu našli, označíme z^{best}. Nech x^k je optimálna hodnota lineárnej relaxácie podúlohy $MIP(k)$ s hodnotou účelovej funkcie rovnej z^k. V prípade, že x^k splňuje podmienky celočíselnosti, potom je optimálnym riešením podúlohy $MIP(k)$ a zároveň prípustným riešením pôvodnej úlohy MIP. Aktualizujeme teda hodnotu z^{best} ako $\min\{z^k, z^{best}\}$. V prípade, že x^k podmienku celočíselnosti nesplňuje:

a) Ak $z^k \geq z^{best}$, potom optimálne riešenie $MIP(k)$ nemôže prispieť ku zlepšeniu hodnoty účelovej funkcie pôvodnej úlohy MIP, a preto sa ním ďalej nebudeme zaoberať.

b) Pokiaľ však $z^k < z^{best}$, potom podúloha $MIP(k)$ vyžaduje ďalšie preskúmanie. Na to použijeme proces vetvenia, a to napríklad tak, že vytvoríme $q \geq 2$ nových podúloh $MIP(k(i)), i = 1,2,\ldots,q$ nad pôvodnou úlohou $MIP(k)$ rozdelením jej množiny prípustných riešení S^k na q podmnožín $S^{k(i)} i = 1,2,\ldots,q$. Ďalej vyberieme premennú x_i, pre ktorú x_i^k nie je celočíselná a vytvoríme dve podúlohy. V jednej podúlohe pridáme podmienku $x_i \leq \lfloor x_i^k \rfloor$, ktorá tvorí dno pre x_i^k a v druhej podúlohe pridáme podmienku $x_i \geq \lfloor x_i^k \rfloor + 1$, ktorá tvorí strop pre x_i^k.

---

[5] ATAMTURK, A., SAVELSBERGH, M.W.P. *Integer- Programming Software Systems*.2005. str. 5. Dostupný z <http://ieor.berkeley.edu/~atamturk/pubs/_published/aor140-2005.pdf>, cit. 07-12-10.

[6] BARTOŇ, J. *Testování software pro řešení úloh smíšeného celočíselného programování: diplomová práca*. Praha : VŠE, 2002. str.2.

Úloha $MIP(k)$ je potom nahradená jej podúlohami v listine L aktívnych úloh. Najmenšia hodnota účelovej funkcie medzi všetkými lineárnymi relaxáciami pridružená k aktívnym úlohám nám poskytuje dolnú hranicu globálnej hodnoty účelovej funkcie. Algoritmus končí, keď sa horná a dolná hranica rovnajú.

Algoritmus 1 Algoritmus metódy vetvenia a medzí

Krok 0. Začiatok.

L = {MIP}. $z^{best} = \infty$. $x^{best} = \varnothing$

Krok 1. Ukončenie?

Je L = $\varnothing$? Ak áno, potom riešenie x^{best} je optimálne.

Krok 2. Výber.

Vyber a vymaž úlohu $MIP(k)$ z listiny **L**.

Metóda výberu úlohy z listiny **L** môže byť náhodná alebo sa môže riadiť určitými heuristickými pravidlami, ktoré budú popísané ďalej v texte.[7]

Krok 3. Ohodnotenie.

Vyrieš lineárnu relaxáciu $LP(k)$ k úlohe $MIP(k)$.

Pokiaľ úloha $LP(k)$ nemá prípustné riešenie, choď na Krok 1, inak z^k bude hodnota jej účelovej funkcie a x^k jej riešením.

Krok 4. Orezávanie.

Ak $z^k \geq z^{best}$, choď na Krok 1.

Ak x^k nie je celočíselné, choď na Krok 5.

Inak polož $z^k = z^{best}$, $x^k = x^{best}$. Choď na Krok 1.

Krok 5. Vetvenie.

Rozdel množinu prípustných riešení S^k úlohy $MIP(k)$ na podmnožiny $S^{k(i)}\ i = 1,\ldots,q$, tak aby $\bigcup_{i=1}^{q} S^{k(i)} = S^k$ a pridaj podúlohy $MIP(k(i)), i = 1,\ldots,q$ do listiny **L**. Choď na Krok 1.

Je možné znázorniť algoritmus metódy vetvenia a medzí ako graf (strom), kde uzly grafu predstavujú jednotlivé podúlohy. Uzol na vrchole grafu sa nazýva koreň a predstavuje pôvodnú úlohu MIP. Ďalšie nasledujúce uzly predstavujú podúlohy získané vetvením. V každom uzle je uvedený výsledok optimalizácie relaxácie na množine, ktorá je týmto uzlom znázornená.[8]

V metóde vetvenia a medzí pre riešenie úloh celočíselného zmiešaného programovania sa vyskytujú dva body, v ktorých nie je výpočet jasne určený:

1. výber úlohy $MIP(k)$ z listiny **L**,
2. výber vetviacej premennej v okamžiku kedy vytvárame z množiny prípustných riešení S^k úlohy $MIP(k)$ podmnožiny $S^{k(i)}$.

[7] PELIKÁN, 2001. str. 93.
[8] ATAMTURK, SAVELSBERGH, 2005, str. 5-6.

Výber úlohy $MIP(k)$ z listiny **L** je založený na prehľadávaní uzlov stromu. Existuje niekoholko stratégií prehľadávania:

- "*last in, last out*", "*diving*", "*depth-first search plus backtracking*", prípadne metóda implicitného vetvenia. Je to veľmi populárna metóda, ktorá vyberá posledný vygenerovaný uzol. Vyberá teda v poradí v akom boli uzly vytvorené. Výhoda tejto metódy je jednoduché tvorenie relaxácií v každom uzle, ktoré spočívá v pridávaní a uberaní horných a dolných medzí k relaxácii predchadzajúceho uzla.[9]
- *"best-bound"* alebo "*best-first search"*, je ďalšiou známou metódou výberu uzlu na preskúmanie. Princíp je taký, že sa vyberie uzol s najmenšou hodnotou z^k. Táto stratégia výberu uzlu sa zameriava na zvyšovanie hodnoty globálnej dolej hranice účelovej funkcie, pretože jediný spôsob ako zlepšiť túto hodnotu je zlepšiť relaxáciu v uzle s najmenšou hodnotou lineárnej relaxácie.
- *"breadth-first search"*
- *"quick improvement"*
- *"best projection"*
- *"best estimate"*
- *atď.*

Väčšina celočíselných riešiteľov používa hybrid metódy *best-bound* a metódy *diving*, pokúšajúc sa využiť výhody každej z nich. Počas výpočtu sú často tieto metódy striedané, za účelom získal čo najlepší výsledkok v čo najkratšom čase. Na začiatku sa obvykle dáva väčší dôraz na metódu *diving* z dôvodu získania výsledku dobrej kvality čo najrýchlejšie. V neskoršom štádiu prehľadávania je dôraz kladený na metódu *best-bound*, a to za účelom zlepšenia dolnej hranice.

Výber vetviacej premennej je založený na fakte, že efektivita metódy vetvenia a medzí závisí na rýchlosti konvergencie dolnej a hornej hranice. Z tohoto dôvodu sa snažíme vetviť na premennej, ktorá čo najviac zlepší tieto hranice. Existujú metódy na určenie premennej, ktorá najviac zlepší dolnú hranicu, pokiaľ má byť celočíselná. Ale čo sa týka hornej hranice, je obtiažne navrhnúť obecné pravidlo pre výber premennej, ktorá by najviac ovplyvnila hornú hranicu.[10]

- *Metóda odhadu založená na pseudocenách*
- *Metóda silného vetvenia (Strong branching)*
- *SOS vetvenie (Special ordered set, SOS)*

Počas procesu vetvenia sú generované dve dôležité veličiny, horná a dolná hranica účelovej fukncie. Hornú hranicu získame nájdením celočíselného prípustného riešenia. Dolnú hranicu získame tým, že vyberieme najmenšiu hodnotu účelovej funkcie

[9] PELIKÁN, 2001, str. 106.
[10] ATAMTURK, A., SAVELSBERGH, 2005, str. 8-12.

lineárnej relaxácie medzi všetkými aktívnymi uzlami. Z hľadiska týchto dvoch hraníc, existuje niekoľko metód, ktoré zlepšujú metódu vetvenia a medzí:

- **Preprocessing / Presolving** – vzťahuje sa k manipulovaniu s formuláciou úlohy MIP v snahe o rýchlejší výpočet, a to bez vynechania akéhokoľvek prípustného riešenia. Preprocessing zvyčajne výrazne zmení pôvodnú formuláciu úlohy vynechaním alebo substituovaním premenných a obmedzení, ale taktiež aj zmenou koeficientov v obmedzeniach a v účelovej funkcii. Bežné techniky preprocessingu zahrňujú odhalovanie nekonzistentných obmedzení, eliminovanie nadbytočných obmedzení, a posilovanie hraníc u premenných. Operácie preprocessingu v uzloch v rámci stromu postihujú obe, dolnú aj hornú hranicu simultánne. Odvodzovaním tesnejších hraních pre celočíselné premenné, preprocessing často spôsobuje zvýšenie hodnoty účelovej funkcie pridruženej relaxácie, a tým zlepšenie dolnej hranice. Vynechaním zlomkových hodnôt sa zvyšuje pravdepodobnosť, že riešenie LP relaxácie v uzle je celočíselne prípustné, a teda potenciálne zlepšuje aj hornú hranicu. Preprocessing sa používa aj ako súčasť uzlových heuristík.
- **Uzlové heuristiky (Node heuristics) –** Myšlienka uzlových heuristík je jednoduchá, Namiesto toho, aby sme čakali, že behom vetvenia narazíme na celočíselné riešenie, izolujeme celočíselnú úlohu, ktorá vznikla v preskúmávanom uzle a aplikujeme na ňu postupy pre získanie celočíselného optima. Hlavným dôvodom pre aplikáciu heuristík je získanie celočíselného optima čo možno najrýchlejšie, pretože tým výrazne znížime počet preskúmávaných uzlov, zosilníme rez, a tým urýchlime prepočítavanie optima duálne simplexovou metódou v danom uzle. V ekonomickej realite sa taktiež často stáva, že dôležitejšie ako nájdenie optima je nájdenie dobrého celočíselného riešenia. V uzlových heuristikách sa najčastejšie používa tzv. “fixing”, pri ktorom dochádza k eliminácii premenných v závislosti napríklad na redukovaných nákladoch.
- **Rezné nadroviny (Cutting planes) –** Rezná nadrovina je platná nerovnosť, ktorá nie je splnená všetkými prípustnými bodmi LP relaxácie. Pokiaľ nájdeme rez, porušený daným riešením LP relaxácie, môžeme ho pridať do formulácie úlohy, a tým posíliť LP relaxáciu. Pôvodná formulácia je potom modifikovaná takým spôsobom, že množina prípustných riešení relaxácie sa zmenší, ale množina prípustných riešení pôvodnej úlohy MIP zostane nezmenená. Potom LP relaxácia novej formulácie je vyriešená a pokiaľ je to nutné proces tvorenia rezov sa opakuje. Je taktiež možné pridať rezné nadroviny a odstrániť časť množiny prípustných riešení pôvodnej úlohy MIP pokiaľ aspoň jedno celočíselné riešenie zostane neovplyvnené.

2.2.2 Algoritmy rezných nadrovín

Čisté algoritmy rezných nadrovín sú také, kde sú rezy pridávané dovtedy, pokiaľ nie je nájdené optimálne riešenie. Generáciu algoritmov pre rezné nadroviny implementovanú do MIP riešiteľov môžeme rozdeliť do dvoch hlavných kategórií:

Prvú kategóriu tvoria obecné rezy, ktoré sú platné pre všetky úlohy MIP:

- *Gomoryho zmiešané celočíselné rezy (Gomory mixed-integer cuts, Balas et al., 1996; Gomory, 1960),*
- *MIR rezy (Mixed-integer rounding (MIR) cuts, Marchand and Wolsey, 2001; Nemhauser and Wolsey, 1990).*

Druhá kategória zahrňuje:

- *Silné špeciálne rezy pre úlohu batohu (Strong special cuts from knapsack, Balas, 1975; Balas and Zemel, 1978; Crowder, Johnson, and Padberg, 1983; Gu, Nemhauser, and Savelsbergh, 1998; Hammer, Johnson, and Peled, 1975; Padberg, 1979),*
- *Fixed-charge flow (Gu, Nemhauser, and Savelsbergh, 1999; Padberg, Roy, and Wolsey, 1985),*
- *Path relaxations of MIPs (Van Roy and Wolsey, 1987),*
- *Vertex packing relaxations of MIPs (Padberg, 1973).*[11]

Pre ilustráciu ďalej uvádzam popis Gomoryho zmiešaných celočíselných rezov.

Gomoryho zmiešané celočíselné rezy

Gomoryho metóda rezov (Gomory mixed-integer cuts) bola prvýkrát predstavená v roku 1960, ale počas mnoho rokov mala reputáciu neefektívnej metódy v praxi. Dôvodom, prečo bola Gomoryho metóda tak dlho zavrhovaná bol fakt, že s aplikáciou rezov obyčajne rastie hustota matice, a to pridávaním nenulových prvkov do matice obmedzení. Vtedajší riešitelia nevedeli zvládnuť túto zvyšujúcu sa hustotu matice.

Ďalším dôvodom bolo to, že v začiatkoch sa rezy aplikovali spôsobom, ktorý je dnes očividne nesprávny, ale v tej dobe bol úplne prirodzený. Gomoryho algoritmus, nie iba rezy, ktoré predstavil, boli považované ako potenciálne kompletné riešenie pre celočíselné úlohy, rovnako ako aj simplexová metóda bola považovaná za "úplnú" metódu pre riešenie lineárneho programovania. A tak namiesto pridávania skupín rezov, kde je skupina tvorená z toľkých "dobre" porušených rezov, koľko sa dá nájsť, rezy boli pridávané po jednom a vetvenie bolo ignorované. Výsledok bol taký, že konvergencia buď vôbec nenastala alebo prebiehala veľmi pomaly.

Časy sa zmenili, výpočtový softvér má väčšie možnosti a už vieme, že rezy sa majú aplikovať po skupinách a hlavne nikto už nepovažuje rezy za všemocný nástroj pre riešenie celočíselných úloh. V kombinácii s vetvením je to v dnešnej dobe najsilnejší nástroj pre svoju univerzálnosť (obecne platí, že vždy je možné nájsť v úlohe celočíselného programovania porušený Gomoryho rez), jednoduchú aplikáciu a prehľadnosť, a to vďaka kombinovanému používaniu zaokrúhlovania a delenia. Gomoryho rezy sa stali štandardom pre dnešných MIP riešiteľov.

Ďalej uvádzam odvodenie Gomoryho rezu pre úlohu MIP:

Algoritmus 2 Gomoryho zmiešaný celočíselný rez

Nech $y, x_j \in Z_+$ a $y + \sum_j a_{ij} x_j = d = \lfloor d \rfloor + f,\ f > 0$.

Predstavme si túto rovnosť ako riadok v optimálnej simplexovej tabuľke.

Ďalej nasleduje zaokrúhľovanie a zápis $a_{ij} = \lfloor a_{ij} \rfloor + f_j$ a môžeme definovať

[11] ATAMTURK, SAVELSBERGH, 2005, str. 18.

$$t = y + \sum\left(\lfloor a_{ij} \rfloor x_j : f_j \le f\right) + \sum\left(\lceil a_{ij} \rceil x_j : f_j > f\right) \in Z_+.$$

Odčítaním dostanem

$$\sum\left(f_j x_j : f_j \le f\right) + \sum\left(\left(f_j - 1\right)x_j : f_j > f\right) = d - t.$$

Ďalej nasleduje vetvenie na t a dostaneme

$$t \le \lfloor d \rfloor \Rightarrow \sum\left(f_j x_j : f_j \le f\right) \ge f \text{ a}$$

$$t \ge \lceil d \rceil \Rightarrow \sum\left(\left(1 - f_j\right)x_j : f_j > f\right) \ge 1 - f$$

Delením pravej strany v každom prípade dostaneme veličinu, ktorá je vždy nezáporná a pre korešpondujúce oblasti hodnoty t je rovná prinajmenšom 1. Z toho dôvodu je suma rovná prinajmenšom 1:

$$\sum\left(\frac{f_j}{f} x_j : f_j \le f\right) + \sum\left(\frac{1 - f_j}{1 - f} x_j : f_j > f\right) \ge 1$$

Táto nerovnosť sa nazýva Gomoryho zmiešaný celočíselný rez.[12]

2.2.3 Metóda vetvenia a rezov

Metóda vetvenia a rezov (Branch-and-cut) je zobecnením metódy vetvenia a medzí, v ktorej pridávame porušené rezy do formulácie v uzloch prehľadávacieho stromu. V prípade, že nie sú nájdené žiadne porušené rezy alebo efektivita rezných nadrovín v zlepšovaní LP medzí klesá, potom použijeme vetvenie. Algoritmus metódy vetvenia a rezov zobecňuje algoritmus čistých rezných nadrovín, kde sa pridávajú rezy až pokiaľ nie je nájdené optimálne riešenie, a teda neprebieha žiadne vetvenie, a algoritmus vetvenia a medzí, kde sa nepridávajú žiadne rezy. Algoritmus metódy vetvenia a rezov stráví veľa času pri riešení LP relaxácií v jednotlivých uzloch stromu. Výsledok vylepšených LP medzí je zvyčajne o dosť menší prehľadávací strom. V takomto prípade algortimus vetvenia a rezov môže byť o dosť rýchlejší ako algoritmus metódy vetvenia a medzí. Nájdenie vhodného kompromisu medzi rezaním a vetvením je esenciálne pri redukcii výpočtového času a je závislé na charakteristike riešenej úlohy.[13]

2.2.4 Metóda vetvenia a oceňovania

Metóda vetvenia a oceňovania (Branch-and-price) je opäť zobecnením metódy vetvenia a medzí, špeciálne navrhnutá pre úlohy MIP s veľkým počtom premenných. Základná myšlienka tejto metódy je, že niektoré stĺpce simplexovej tabuľky LP relaxácie sú vynechané, a to z toho dôvodu, že tabuľka obsahuje príliš veľa stĺpcov na to, aby sa s ňou dalo pracovať efektívne. Väčšina stĺpcov bude mať aj tak na miestach pridružených premenných v optimálnom riešení hodnoty rovné nule. Potom podúloha, označovaná ako „oceňovacia úloha“ (pricing problem) je riešená v snahe identifikovať stĺpce s výhodnými redukovanými cenami pre kontrolu optimality LP riešenia. Pokiaľ

[12] BIBXY, str. 17-19.
[13] ATAMTURK, SAVELSBERGH, 2005, str. 18.

sa také stĺpce nájdu, úloha LP je znova optimalizovaná. V prípade, že žiadne výhodné stĺpce nie sú nájdené a výsledok LP relaxácie nesplňuje podmienky celočíselnosti, potom sa použije vetvenie. Táto metóda používa generovanie stĺpcov v každom uzle prehľadávacieho stromu.[14]

2.2.5 Heuristiky a metaheuristiky

Ako bolo zmienené už predtým, úlohy MIP sú NP-ťažké, takže získanie optimálneho výsledku môže trvať dlhú dobu. V týchto prípadoch môžeme použiť heuristiky, čo sú vlastne algoritmy, ktoré sa snažia nájsť prípustné riešenia úlohy čo najrýchlejšie. Aj napriek tomu, že účelová funkcia nemusí byť optimálna, jej hodnota je v praxi postačujúca.

Heuristiky na rozdiel od exaktných algoritmov sú rýchle, polynomiálne a nemajú toľko problémov pri riešení rozsiahlych úloh. Umožňujú taktiež flexibilné úpravy pre rôzne varianty typových úloh a pre špecifické podmienky.

Heuristiky sú zväčša formulované pre konktrétnu úlohu. Existujú však aj obecné heuristické postupy, ktoré môžu byť použité na riešenie obecných optimalizačných úloh a nazývajú sa *metaheuristiky*.

Majme obecný optimalizačný problém s množinou prípustných riešení X. Hľadáme optimálne riešenie $x^* \in X$, ktoré minimalizuje hodnotu účelovej funkcie $f(x)$ na množine X. Ďalej predpokladajme, že pre každé $x \in X$ je definovaná množina $N(x) \subset X$, ktorú nazývame okolie bodu x, pre ktoré platí $x \in N(x)$.

Spoločný základný princíp metaheuristík spočíva v tom, že najskôr nájdeme nejaké východzie prípustné riešenie. Nasleduje postupnosť prípustných riešení, kde nasledujúce riešenie nájdeme v okolí predchádzajúceho riešenia. Táto postupnosť je ukončená splnením určitého kritéria ukončenia metódy.[15]

Medzi najznámejšie metaheuristiky patria:

- **Metóda lokálneho hľadania (Local search)** - je metaheuristika na riešenie výpočtovo zložitých optimalizačných problémov. Táto metóda začína v ľubovoľnom prípustnom riešení úlohy, ktoré označíme x. Potom sa postupne presúva do ďalšieho riešenia, ktoré je v okolí bodu x a zároveň má nižšiu hodnotu účelovej funkcie Metóda končí, keď v okolí bodu neexistuje žiadne iné riešenie s nižšou hodnotou účelovej funkcie, čo znamená, že sme sa dostali do lokálneho minima účelovej funkcie. Toto je najväčšia nevýhoda tejto metódy, že je nájdené iba lokálne minimum, a nie globálne minimum účelovej funkcie, čo je naším cieľom. Pokiaľ by sme chceli zlepšiť túto metódu, musíme nájsť spôsob, ako sa dostať z lokálneho minima a pokračovať v hľadaní lepšieho riešenia.

[14] SAVESBERGH, M.W.P. BRANCH-*AND-PRICE: INTEGER PROGRAMMING WITH COLUMN GENERATION, BP. str*. 11. Dostupné z <http://www2.isye.gatech.edu/~mwps/publications/eoo.pdf>, cit. 07-12-08.

[15] PELIKÁN, 2001, str.152.

Algoritus 3 Metóda lokálneho hľadania

Krok 1

Nájdi akékoľvek prípustné riešenie $x \in S$.

Krok 2

Postupne pre každé $x' \in N(x)$ sprav:

Ak $f(x') < f(x)$, potom dosaď $x := x'$, tj. pokiaľ hodnota účelovej funkcie v bode x' je nižšia než v bode x, potom sa presunieme z x do x'.

Ak $x := x'$, choď na krok 2.

Krok 3

V prípade, že sa nepresunieme z x do x' v kroku 2, potom je výpočet ukončený a x je riešením metódy lokálneho hľadania.

- **Metóda tabu search** – táto metóda rieši problém metódy lokálneho hľadania popísanej vyššie, a to tým, že pridáva do tzv. „tabu“ zoznamu, riešenia, ktoré už boli spracované**.**

Algoritmus 4 Metóda tabu search

Krok 1

Nájdi akékoľvek prípustné riešenie $x \in S$ a polož $x^* := x$, kde x^* označuje najlepšie doteraz nájdené riešenie.

Krok 2

Nájdeme $x' \in \bar{N}(x) \subset N(x)$, ktoré dosahuje minima funkcie f na množine $\bar{N}(x)$.

Množina $\bar{N}(x)$ obsahuje tie riešenia z $N(x)$, ktoré nie sú v „tabu“ zozname.

Krok 3

Zapíš riešenie x do „tabu“ zoznamu a dosaď $x := x'$.

Pokiaľ $f(x) < f(x^*)$, potom dosaď $x^* := x$.

Krok 4

Pokiaľ je splnené ukončovacie kritérium metódy (napr. po určitom počte krokov nedošlo k zmene riešenia x^*), výpočet končí a výsledkom je riešenie x^*. Inak choď na krok 2.

- **Metóda simulovaného žíhania (SIAM, Simulated annealing)** – je generická stochastická metaheuristiha. Vychádza z metódy lokálneho hľadania a rozširuje ju. Jej meno pochádza z procesu žíhania v metalurgii. Je to technika zahrňujúca zahrievanie a kontrolované ochladzovanie materiálu za účelom zmeny jeho vlastností. Metóda je založená na analógii medzi týmto fyzikálnym procesom a procesom minimalizácie účelovej funkcie $f(x)$. Analógia je založená na hodnote, kvalite a veľkosti kryštálov železa v oceli, a tým odvodenej kvalite oceli a hodnote $f(x)$. V tejto metóde T označuje teplotu, $r \in (0,1)$ označuje chladenie a n predstavuje čas žíhania. Presun z x do x' je viazaný na pravdepodobnosť $\exp -(f(x') - f(x))/T$. Z tohoto vzorca je jasné, že v prípade, že rozdiel $\delta = f(x') - f(x)$ je malý (riešenie x' je len o trochu horšie ako x) potom prechod z x do x' sa uskutoční s vysokou pravdepodobnosťou a naopak. V prípade, že x' je lepšie ako x, prechod sa uskutoční s pravdepodobnosťou rovnou 1.

Algoritmus 5 Metóda simulovaného žíhania

Krok 1

Nájdi akékoľvek prípustné riešenie $x \in S$ a vyber parametre metódy: teplotu $T > 0$, parameter ochladzovania $r \in (0,1)$ a dobu žíhania n. Polož $x^* := x$.

Krok 2

Opakuj n-krát.

Vyber riešenie x' z okolia $N(x)$ náhodne (najlepšie s rovnomerným rozložením pravdepodobnosti výberu)

a) Ak $f(x') < f(x)$ dosaď $x := x'$,

b) Ak $f(x') \geq f(x)$ potom dosaď $x := x'$ s pravdepodobnosťou $\exp -(f(x') - f(x))/T$,

c) Ak $f(x^*) > f(x)$ dosaď $x^* := x$.

Krok 3

Pokiaľ sa v kroku 2 riešenie x nezmenilo (nedošlo k presunu $x := x'$), hovoríme, že proces zamrzol a potom metóda končí s výsledkom x^*. V opačnom prípade znížime teplotu T podľa vzorca $T := rT$, kde r je generované náhodne a pokračujeme krokom 2. [16]

Táto metóda má tendenciu vylúčiť „veľmi dobrých" kandidátov, rovnako ako aj „veľmi zlých", avšak ten druhý prípad je bežnejší ako prvý, takže táto metaheuristika je považovaná za všeobecne efektívnu.[17]

[16] PELIKÁN, 2001, str. 153-155.

[17] Dostupné z <http://en.wikipedia.org/wiki/Simulated_annealing>, cit. 07-12-08.

Pre metódu SIAM bolo dokázané, že štatisticky konverguje do globálneho minima.

- **Genetický algoritmus (Genetic algorithm)** – je metaheuristika, ktorá používá techniky inšpirované evolučnou biológiou. Algoritmus pracuje s určitými podmnožinami prípustných riešení (nazývaných populácie) abstraktných reprezentácií (nazývaných chromozómy) kandidátnych riešení (nazývaných individuá). Riešenia sú tradične reprezentované binárnymi reťazcami čísiel 0 a 1, ale iné zakódovanie je taktiež možné. Evolúcia obvykle začína z populácie náhodne vybraných individuí a deje sa v generáciách, až pokiaľ určité ukončovacie kritérium nie je splnené. V každej generácii sa stanovuje fitnes (vhodnosť/zdravie) každého individua na základe tzv. fitnes funkcie. Potom niekoľko individuí je stochasticky vybraných zo súčasnej populácie na základe procesu, kde riešenia s lepším ohodnotením fitnes majú väčšiu pravdepodobnosť byť vybrané. Určité metódy výberu stanovujú fitnes každého riešenia v populácii a vyberajú najlepšie ohodnotené riešenia. Iné metódy ohodnocujú iba náhodnú vzorku populácie. Väčšina funkcií je stochastických a vytvorených tak, aby aj malá časť menej vhodných/zdravých riešení bola vybraná, a to z dôvodu udržania veľkej rôznorodosti populácie, čo má za účel predchádzať predčasnej konvergencii ku slabým riešeniam. Ďalší krok je generovať druhú generáciu populácie. Pre každé nové „dieťa“ je vybraný pár „rodičovských“ riešení na výchovu z množiny určenej v prechádzajúcom kroku. Potom použitím kríženia a mutácie sú vytvorené nové riešenia, ktoré majú mnoho spoločných charakteristík ako ich „rodičia“. Pre každé dieťa sú vybraní noví rodičia a proces pokračuje až pokiaľ nevznikne nová populácia požadovanej veľkosti. Tieto procesy majú za následok vytvorenie novej generácie populácie chromozónov, ktoré sú rozdielne od počiatočnej generácie. Priemerné zdravie novej populácie by sa malo obecne zlepšiť touto procedúrou, nakoľko iba najvhodnejší/najzdravší jedinci boli vybraní na výchovu (aj spolu s určitou časťou menej zdravých riešení z dôvodu vysvetlenom vyššie). Ďalej nasleduje ohodnotenie fitnes jednotlivých potomkov. Po tomto kroku, časť populácie s najhorším ohodnotením je vymenená za potomkov. Takto novo vytvorená populácia je použitá v ďalšej iterácii algoritmu a proces pokračuje, až pokiaľ nie je splnené určité ukončovacie kritérium a potom algoritmus končí.

  Príklady typických ukončovacích kritérií sú:

  - riešenie, ktoré splňuje minimálne kritéria je nájdené,
  - je vytvorený určítý fixný počet generácií,
  - pridelený rozpočet (výpočtový čas/peniaze) je vyčerpaný,
  - ohodnotenie najlepšieho riešenia dosiahlo vrchol, takže ďalšie iterácie neprodukujú lepšie riešenia,
  - manuálna revízia,
  - kombinácia uvedených.[18]

[18] Dostupné z <http://en.wikipedia.org/wiki/Genetic_algorithm>, cit. 07-12-08.

- **Mravenčí algoritmus (ACO, Ant colony optimalization)** – je pravdepodobnostná technika na riešenie úloh, ktoré je možné redukovať na hľadanie dobrej cesty v grafe. Metóda je inšpirovaná chovaním mravcov pri hľadaní cesty od potravy do mraveniska. V realite, mravci spočiatku blúdia náhodne a pri nájdení potravy sa vracajú späť do mraveniska zanechávajúc za sebou stopy feromónu. Ak ostatní mravci nájdu takúto stopu, pravdepodobne ju budú následovať namiesto náhodného blúdenia, posilujúc feromónmi túto cestu v prípade, že nájdu potravu. Po čase sa začne feromónová stopa odparovať, čo znamená zníženie atraktivity danej cesty. Čím viac času trvá mravcovi cestovať po ceste k jedlu a naspäť, tým dlhšie sa feromónová stopa vyparuje. Krátka cesta je naopak častokrát prechádzaná, a preto hustota feromónov na nej zostáva vysoká. Vyparovanie feromónu má tú výhodu, že sa vďaka nemu vyhneme konvergencii k lokálnemu optimálnemu riešeniu. V prípade, že by sa feromóny nevyparovali, cesta vybraná prvým mravcom, by mala sklon byť prehnane príťažlivá pre ďalšie mravce a v takom prípade preskúmávanie priestoru by bolo omedzené. Pokiaľ teda jeden mravec nájde dobrú cestu (napr. krátku) z mraveniska k potrave, ostatné mravce budú pravdepodobne následovať túto cestu a pozitívna spätná väzba môže prípadne viesť k tomu, že všetky mravce budú následovať jedinú cestu . Idea mravenčieho algoritmu je imitovať toto chovanie pomocou „simulovaných mravcov“, ktorí chodia po grafe reprezentujúceho riešenú úlohu. Mravenčí algoritmus má výhodu oproti metóde SIAM a genetickému algoritmu, v prípade, keď sa graf mení dynamicky. V takom prípade mravenčí algoritmus môže bežať kontinuálne a adaptovať sa zmenám v reálnom čase. Toto nachádza uplatnenie napríklad v okružných úlohách a systémoch mestskej hromadnej dopravy.[19]

2.3 HISTORICKÝ PREHĽAD

2.3.1 Operačný výzkum (Operations research)

Počiatok operačného výzkumu sa datuje späť do doby druhej svetovej vojny, kedy vedenie britskej armády vytvorilo tím vedcov k štúdiu strategických a taktických problémov spojených so vzdušnou a pozemnou obranou krajiny. Ich cieľom bolo nájsť spôsob čo najefektívnejšieho využitia limitovaných vojenských zdrojov. Táto disciplína bola nazvaná „operačný výzkum“, pretože tento tím sa zaoberal štúdiom prevádzky v armáde (operations research).

Povzbudzujúce výsledky dosiahnuté britským tímom motivovali ostatné krajiny k započatiu podobných aktivít. Po konci vojny zaujal tento úspech priemyslových manažerov, ktorí hľadali riešenia svojich problémov. Operačný výzkum začal teda po druhej svetovej vojne prenikať i do civilného priemyslu, a to hlavne oceliarskeho, naftového, textilného, energetického, chemického a strojárenského.[20]

Prvá matematická technika v obore bola simplexová metóda na riešenie úloh linerárneho programovania. Bola vytvorená americkým matematikom George B. Dantzigom v roku 1947. Odvtedy boli vyvinuté mnohé ďalšie techniky a aplikácie ako

[19] Dostupné z <http://en.wikipedia.org/wiki/Ant_colony_optimization>, cit. 07-03-08.
[20] Dostupné z <http://www.seminarky.cz/Operacni-vyzkum-otazky-ke-zkousce-5287>, cit. 12-03-2008.

napr. 1949 – Simulácia Monte Carlo (S. M. Ulam, J. von Neumann), 1950 – Dynamické programovanie (R. Bellman), 1951 – Nelineárne programovanie (H. Kuhn, A. Tucker), 1954 – Parametrické programovanie (S. I. Gass, T. L. Saaty) atď.[21]

Obrovský progres v oblasti operačného výzkumu bol spôsobený paralelným rozvojom moderných digitálnych počítačov a ich kapacitou vo výpočtovej rýchlosti a úschove informácií.

Dnes vplyv operačného výzkumu môžeme nájsť v mnohých oblastiach. Tieto aktivity prekročili použitie v armáde a v priemysle a je ich možné nájsť napr. v nemocniciach, finančných inštitúciách, knihovniach, dopravných systémoch atď.

2.3.2 Lineárne programovanie a výpočtová technika

Pre lineárne programovanie (LP) sa začala používať výpočtová technika vo chvíli, keď Dantzig predstavil v roku 1947 svoju simplexovú metódu. Prvý prípad netriviálnej LP úlohy vyriešený pomocou simplexovho algoritmu bol Stiglerov dietný problém vyriešený Leadremanom. Táto úloha mala 9 obmedzení, 77 premenných a tímu deviatich matematikov trvalo na elektonických kalkulačkách 120 „človeko – dní“ než vypočítali optimálne riešenie.

Prvý pokus o počítačovú implementáciu simplexovej metódy pri riešení úloh lineárneho programovania prebehol v roku 1952 v National Bureau of Standard na počítači SEAC. Alex Orden a A.J.Hoffman testovali úlohu s 48 obmedzeniami a 71 premennými a po 18 hodinách, počas ktorých prebehlo 73 iterácií, získali riešenie.

V roku 1953 William Orchard-Hays začal jeho priekopnickú prácu na počítačovej implementácii simplexovej metódy. Táto práca bola začiatkom rozvoja komerčne dostupným LP kódov. Použil na to počítač IBM programovatelný pomocou diernych štítkov, teda niečo, čo sa určite nedá považovať za počítač podľa dnešných štandardov. Najväčšia úloha vyriešená pomocou tohto kódu mala 26 obmedzení a 76 premenných a trvalo 8 hodín, kým bolo nájdené riešenie.

Po W. Orchard-Haysovi sa mnoho ďalších vedcov venovalo zlepšovaniu implementácie simplexovho algoritmu. Po čase, spolu s rozvojom výpočtovej techniky, nový kód s názvom RLSL1 bol vyvinutý pre počítač IBM 704. Úlohy pre tento kód mohli mať maximálne 255 obmedzení a teoreticky neobmedzený počet premenných (IBM 704 mal necelých 4 KB pamäte).

V rokoch 1962 - 1966 bol na počítači IBM 709 používaný kód LP/90 následovaný kódom LP/90/94 na počítačí IBM 7094. Počet obmedzení úlohy vzrástol na 1024.

Najväčší zlom nastal v roku 1966, kedy firma IBM predstavila počítač IBM 360 a bol vyvinutý kód MPS/360. Po čase bol tento kód nahradený kódom MPSX a neskôr kódom MPSX/370.

[21] Dostupné z <http://www.lionhrtpub.com/orms/orms-10-02/frhistorysb1.html>, cit. 15-03-2008.

Na ich základe v roku 1971 firma Ketron Corporation vyvinula pre platformu IBM kód MPS III. Tento kód predstavoval „kvantový skok" v rýchlosti výpočtu a počtu obmedzení, ktorý vzrástol až na 32 000.

Počas tejto periódy boli vyvinuté aj iné výkonné systémy ako napríklad systém UMPIRE pre počítač UNIVAC 1108, APEX III pre stroje CDC, a LAMPS napísaný nezávisle Johnom Forrestom. Až do konca 80.rokov boli tieto kódy prevládajúcim štandardom.[22]

Mnoho ďalších kódov bolo vytvorených, ale najviac rozšíreným kódom sa stal kód MPS, ktorý detailnejšie popíšem v nasledujúcej kapitole.

Vývoj výpočtovej techniky spolu s aplikáciou efektívnejších algoritmov a výpočtových procedúr mal za následok zvýšenie rýchlosti výpočtového času. Ako príklad môže slúžiť rozdiel vo výpočtovom čase pri riešení úlohy LP o rozmeroch 49944 x 177628 na počítači Sun UltraSparc (296 MHz) programom CPLEX 1.0. Tento program, predstavený v roku 1988 optimalizoval túto úlohu za 57 840 sekúnd. Neskoršia verzia programu CPLEX 5.0 predstavená v roku 1997 vyriešila túto úlohu na rovnakom počítači za 3 835 sekúnd. Ako môžete vidieť, zrýchlenie bolo takmer dvadsaťnásobné.[23] Dnes je najnovšia verzia programu CPLEX 11.01 schopná vyriešiť úlohu o rozmeroch 72600 x 304800 pri dobrom technickom vybavení iba za 311 sekúnd.[24]

Výpočtová technika rovnako ako aj lineárne programovanie prešla od svojich počiatkov obrovským vývojom. V posledných desaťročiach vzrástli jej možnosti natoľko, že dnes je možné za pár minút vypočítať úlohy s tisícami premenných, a to na obyčajných kancelárskych počítačoch.

2.4 FORMÁT MPS

Formát MPS (Mathematical Programming System) je vstupný formát pre zadávanie a archiváciu úloh lineárneho a zmiešaného celočiselného programovania. Ako už bolo spomenuté, tento formát vyvinula firma IBM začiatkom šesťdesiatych rokov minulého storočia za účelom štandardizácie zadávania úloh lineárneho a celočíselného programovania do počítačov IMB. Vzhľadom k tomu, že typickým pamäťovým médiom v tej dobe boli dierne štítky (kapacita dierneho štítku bola 80 znakov), vo formáte MPS každý riadok obsahuje maximálne 80 znakov (každý riadok reprezentuje jeden dierny štítok). Je to textový formát, v ktorom niektoré informácie musia byť zapísané na presne dané pozície. Jedná sa teda o format s pevnou štruktúrou.25 Dáta sú zapisované do formátu MPS v ASCII kóde a musia byť uložené do súboru s povinnou príponou "mps".[26]

[22] BIBXY, str. 7.
[23] BARTOŇ, 2002, str. 6.
[24] Dostupné z < http://plato.asu.edu/ftp/cpx_msk.html>, cit. 23-03-2008.
[25] JABLONSKÝ, J. *Programy pro matematické modelování.* 1.vyd. Praha : Nakladatelství Oeconomica, 2007. str. 122. ISBN 978-80-245-1178-8.
[26] BARTOŇ, 2002, str. 6.

Po svojom uvedení firmou IMB sa tento formát stal rýchlo štandardom i pre ďalšie počítače a dnes je podporovaný prevažnou väčšinou optimalizačných softvérov. Mnoho z týchto softvérov umožňuje konverziu vlastného formátu do formátu MPS. Dnes už je nemyslitelné, aby nejaký skutočne profesionálny optimalizačný softvér nevedel načítať dáta v tomto formáte.

Tabuľka 2.1: Štruktúra MPS formátu

	Pole 1	**Pole 2**	**Pole 3**	**Pole 4**	**Pole 5**	**Pole 6**
Pozícia	2-3	5-12	15-22	25-36	40-47	50-61
Obsah	Indikátor	Názov	Názov	Hodnota	Názov	Hodnota

Dáta sú rozdelené do rôznych sekcií v následovnom poradí:

- **NAME**
- **ROWS**
- **COLUMNS**
- **RHS**
- **RANGES (nepovinné)**
- **BOUNDS (nepovinné)**
- **ENDATA**

Prvé štyri sekcie a posledná sekcia sú povinné a ďalšie dve sú používané podľa potreby. Uvedené sekcie musia byť zapísané v presne danom poradí (viz. vyššie) a môžu byť použité maximálne raz.[27]

Nasleduje podrobnejší popis jednotlivých sekcií:

1. Sekcia **NAME –** jedná sa o identifikačnú sekciu, ktorá obsahuje iba jeden riadok s názvom úlohy. Na pozíciách 1-4 obsahuje slovo "NAME" a na pozíciách 15-22 samotný názov úlohy.

2. Sekcia **ROWS** – táto sekcia začína slovom "ROWS" (pozícia 1-4), následovaná dátami pre každý riadok. V tejto sekcii sú definované všetky názvy a typy obmedzení úlohy (prípadne účelovej funkcie). Typ obmedzenia je uvedený v poli 1 (na pozícii 2 alebo 3) a názov v poli 2 (pozícia 5-12).

Tabuľka 2.2: Symboly pre zápis rôznych typov obmedzení vo formáte MPS

Symbol	
N	účelová funkcia
E	obmedzujúca podmienka typu =
G	obmedzujúca podmienka typu ≥
L	obmedzujúca podmienka typu ≤

Pre každé riadkové obmedzenie sa v minulosti používal jeden dierny štítok. Pokiaľ sa jedná o účelovú funkciu musí mať riadok atribút N a úloha môže mať aj viacero účelových funkcií. Samotný výber účelovej funkcie, podľa ktorej sa bude hľadať

[27] JABLONSKÝ, 2007, str. 121-122.

optimum sa definuje príkazom OBJECTIVE. V prípade, že užívateľ sám neurčí účelovú funkciu podľa ktorej sa má optimalizovať, použije sa prvá v poradí. V tejto sekcii je taktiež možné definovať lineárne kombinácie jednotlivých obmedzení. V tomto prípade sú typy obmedzení označené atribútmi DE, DL, DG a DN, na pozíciách 2-3. Pole 2 obsahuje názov lineárnej kombinácie, polia 3-5 definujú riadky, z ktorých sa skladá lineárna kombinácia a polia 4-6 obsahujú násobiteľov, ktorí vytvárajú lineárnu kombináciu.

3. Sekcia **COLUMNS** – Táto sekcia začína slovom "COLUMNS" (pozícia 1-7). V tejto sekcii sú definované názvy premenných, koeficienty účelovej funkcie c_j a všetky nenulové hodnoty matice štruktúrnych koeficientov obmedzujúcich podmienok a_{ij}. Meno premennej sa uvádza v poli 2 (pozícia 5-12). Názov riadku v poli 3 (pozícia 15-22) a hodnota koeficientov a_{ij} alebo c_j v poli 4 (pozícia 25-36). Polia 5 a pole 6 nie sú povinné. Pole 5 sa používa pre prípadné ďalšie pomenovanie riadku, ktorý obsahuje nenulové koeficienty uvedennej premennej (pozícia 40–47) a v poli 6 je hodnota koeficientu odpovedajúceho riadku špecifikovaného v poli 5. To znamená, že každý riadok sekcie COLUMN môže obsahovať najviac dva nenulové koeficienty patriace jednej premennej. Prípadné premenné sa nemusia špecifikovať, nakoľko je o ne postarané v sekcii ROWS definovaním typu riadku.

4.Sekcia **RHS** – Posledná povinná sekcia je sekcia pravých strán. Táto sekcia začína slovom „RHS" (pozícia 1–3). Nakoľko pravú stranu môžeme brať ako ďalší stĺpec matice obmedzení, riadky špecifikujúce nenulové hodnoty môžu byť zapísané v takom istom formáte, ako v sekcii COLUMN, okrem poľa 2 (pozícia 5 – 12), kam sa zapisuje názov pravej strany. V tejto sekcii teda môže byť špecifikovaná viac ako jedna pravá strana.

5.Sekcia **RANGES** (voliteľná) – Táto sekcia je prvou voliteľnou sekciou formátu MPS. Pokiaľ je použitá, musí následovať bezprostredne za sekciou RHS a začínať slovom RANGES (pozícia 1-6). Táto sekcia je určená pre obmedzenia typu:

$h_i \le a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \le u_i$ s rozpätím $r_i = u_i - h_i$. V sekcii ROWS je treba definovať typ obmedzujúcej podmienky. Pokiaľ sa uvedie typ L(≤) potom hodnota konštanty r_i zadanej v sekcii RANGES pre toto obmedzenie udáva, o koľko musí byť ľavá strana obmedzujúcej podmienky menšia alebo rovná zadanej hodnote pravej strany (h_i), ktorá je špecifikovaná v sekcii RHS. Pokiaľ sa naopak uvedie typ G(≥), potom hodnota konštanty r_i zadanej v sekcii RANGES pre toto obmedzenie udáva, o koľko musí byť ľavá strana obmedzujúcej podmienky väčšia alebo rovná zadanej hodnote pravej strany, ktorá je špecifikovaná v sekcii RHS. Názov súboru rozpätia sa zapisuje do poľa 1 (pozícia 5-12). Do poľa 2 (pozícia 15-22) sa zapisuje meno riadku (obmedzujúca premenná), pre ktorý sa špecifikuje šírka rozpätia a do poľa 3 (pozícia 25-36) sa zapisuje šírka rozpätia r_i.

6.Sekcia **BOUNDS** (voliteľná) – V tejto sekcii sú uvedené horné a dolné medze pre premenné, prípadne ďalšie informácie o charaktere premenných. Táto sekcia začína slovom „BOUNDS" (pozícia 1-6). Každý riadok obsahuje informáciu o typu medze, názov premennej a hodnotu medze, pokiaľ má uvedenie hodnoty význam. Nenulové hodnoty odpovedajú stĺpcom matice, a preto musia byť zapísané v rovnakom tvare ako

v sekcii COLUMN. Pokiaľ nie sú určené medze pre premenné , predpokladá sa $0 \leq x_i \leq \infty$. Formát zápisu je následovný:

- Pole 1 (pozícia 2-3) špecifikuje typ medze:

  LO – Dolná medz premennej
  UP – Horná medz premennej
  FX – Premenná je pevne fixovaná na stanovenú hodnotu
  MI – Dolná hranica $-\infty$
  BV – Binárna premenná
  UI – Celočíselná premenná so stanovenou hornou medzou

- Pole 2 (pozícia 5-12) špecifikuje názov súboru medzí.
- Pole 3 (pozícia 15-22) špecifikuje meno riadku, pre ktorý sa špecifikuje šírka rozpätia.
- Pole 4 (pozícia 25-36) špecifikuje hodnotu medze b_j.

7.Sekcia **ENDATA** - Táto sekcia signalizuje ukončenie zápisu dát do súboru a obsahuje jediné slovo ENDATA na pozícii 1-6. [28]

Zvláštnosťou je, že nič vo formáte MPS nešpecifikuje smer optimalizácie a zároveň ani neexisuje žiadne predvolené nastavenie tohto smeru. Niektoré LP kódy automaticky maximalizujú účelovú funkciu, iné zasa minimalizujú a niektoré neurobia nič, pokiaľ užívateľ sám explicitne neurčí smer optimalizácie. Ukážka príkladu vo formáte MPS je uvedená v prílohe číslo 1.

[28] Dostupné z <http://www-fp.mcs.anl.gov/OTC/Guide/OptWeb/continuous/constrained/linearprog/mps.html>, cit. 02-04-2008.

3. TESTOVANIE SOFTVÉRU

3.1 METÓDA TESTOVANIA

Testovanie softvéru je proces používaný k meraniu softvérovej kvality vyvinutého počítačového programu/aplikácie/systému. Testovanie je proces technickej investigácie a jeho cieľom je získanie informácií o kvalite produktu. Kvalita a formalizácia softvérového testovania závisí predovšetkým na jednotlivcovi (firme) ako toto testovanie pojme. Pri testovaní jednoduchých aplikácií sa môže jednať napr. o základné otestovanie funkcionality systému zadaním očakávaných hodnôt a parametrov a následná kontrola správnosti výsledkov.

Pri zložitejších programoch je však proces komplikovanejší. Veľké softvérové spoločnosti, častokrát vytvárajúce veľmi zložité systémy, si z pravidla zakladajú na vyššej kvalite a spoľahlivosti a chcú byť navonok vnímané profesionálne. Taktiež ich produkty musia (samozrejme aj z dôvodu vyššej cene) splňovať štandardy, ktoré zaistujú určitú minimálnu úroveň kvality. Pre tieto účely bol vytvorený štandard ISO 9126, Mezinárodnou spoločnosťou pre štandardizáciu (ISO - International Organization for Standardization). Štandard ISO 9126 sa zameriava na tieto charakteristiky: funkcionalitu, spoľahlivosť, využitelnosť, výkonnosť, udržovatelnosť a prenosnosť [29]

Metódy testovania softvéru sa delia na takzvaný "*black box testing*", "*grey box testing*" a "*white box testing*".

- *Black box testing* - zaobchádza so softvérom ako s čiernou skrinkou, bez akéhokoľvek poznania jeho vnútornej štruktúry fungovania. Cieľom je otestovať funkcionalitu softvéru na základe určitých požiadaviek. Testujúci teda vloží vstupné dáta a sleduje výstupy, ktoré následne porovnáva s predom známymi výsledkami.

- *White box testing* – je opačný prípad, keď testér má prístup k vnútornej štruktúre, kódu a algoritmom testovaného softvéru.

- *Grey box testing* – sa začal používať až v posledných rokoch. V tomto prípade má testujúci prístup k vnútornej štruktúre a algoritmom testovaného softvéru za účelom vytvorenia testovacích úloh. Samotné testovanie ale prebieha na úrovni užívateľa, teda pomocou metódy *black box testing*. [30]

Na testovanie vybraných softvérov pre úlohy zmiešaného celočíselného programovania som použila v mojej diplomovej práci metódu *black box testing*, ktorú považujem za najvhodnejšiu z toho dôvodu, že primárnym cieľom samotného testovania je porovnávanie rýchlosti a presnoti výstupov jednotlivých softvérových produktov. Teda vnútorná štruktúra fungovania nás v tomto prípade nezaujíma.

[29] Dostupné z <http://cs.wikipedia.org/wiki/Testování_software>, cit. 02-04-2008.
[30] Dostupné z <http://en.wikipedia.org/wiki/Software_testing>, cit 02-04-2008.

Testovanie prebiehalo na počítači s nasledujúcimi technickými parametrami:

Procesor: AMD Athlon 64 2.2GHz

RAM: 960 MB

HDD: 160 GB

OS: Win XT

Ako testovacie úlohy boli použité úlohy z knižnice MIPLIB 2003, čo je elektronická knižnica reálnych celočíselných a zmiešaných celočíselných úloh. MIPLIB sa stal štandardom pre porovnávanie výkonnosti optimalizačných softvérov. Každá úloha obsahuje doplňujúce informácie popisujúce štruktúru matice obmedzení z dôvodu identifikácie rôznych tried týchto obmedzení. To sa potom môže využiť v rôznych fázach výpočtu riešenia úlohy ako napríklad preprocessingu, generovaní rezov alebo vetvení.[31] MIPLIB 2003 obsahuje 60 úloh s veľkosťou od 12,5KB do 34MB. Všetky úlohy sú v MPS formáte. Úlohy sa delia na 3 skupiny:

1. Úlohy, ktoré by mal byť komerčný softvér schopný vypočítať do 1 hodiny.
2. Úlohy, ktorých optimálne riešenie je známe.
3. Úlohy, ktorých optimálne riešenie sa zatiaľ nepodarilo vypočítať. [32]

Na ďalších stránkach uvádzam výpis a základné charakteristiky úloh z jednotlivých skupín knižnice MIPLIB 2003. V prílohe číslo 2 sú uvedené zdroje a tvorci matematických modelov daných úloh.

Pre testovanie som si vybrala softvéry CPLEX, Xpress a MOSEK , ktoré tvoria špičku v optimalizačných softvéroch. Použila som verzie, ktoré boli v dobe začiatku môjho testovania aktuálne a dostupné. Pri testovaní vybraných softvérových produktov som nemenila žiadne parametre. Všetko zostalo vo svojich prednastavených hodnotách. Pri zmene parametrov, by sa samozrejme mohlo riešenie úloh urýchliť, prípadne spresniť, ale prednastavenú konfiguráciu parametrov som zvolila z dôvodu možnosti objektívneho porovnania s výkonom ostatných testovaných softvérov.

[31] Dostupné z <http://mathforum.org/library/view/7098.html>, cit. 29-10-2007.

[32] Dostupné z <http://miplib.zib.de/miplib2003.php>, cit. 30-10-2007.

Tabuľka 3.1: Úlohy prvej skupiny knižnice MIPLIB 2003

Názov úlohy	Počet obmedzení	Celkový počet premenných	Počet celoč. prem.*	Počet bin. prem.**	Najlepšie známe celočíselné riešenie	Hodnota optima bez podmienok celočíselnosti
10teams	230	2,025		1,800	924	917
aflow30a	479	842		421	1,158	983
air04	823	8,904		8,904	56,137	55,535
air05	426	7,195		7,195	26,374	25,878
cap6000	2,176	6,000		6,000	-2,451,377	-2,451,537
disktom	399	10,000		10,000	-5,000	-5,000
fast0507	507	63,009		63,009	174	172
fiber	363	1,298		1,254	405,935	156,083
fixnet6	478	878		378	3,983	1,201
gesa2	1,392	1,224	168	240	25,779,856	25,476,490
gesa2-o	1,248	1,224	336	384	25,779,856	25,476,490
harp2	112	2,993		2,993	-73,899,799	-74,353,342
manna81	6,480	3,321	3,303	18	-13,164	-13,297
mas74	13	151		150	11,801	10,483
mas76	12	151		150	40,005	38,894
misc07	212	260		259	2,810	1,415
mod011	4,480	10,958		96	-54,558,535	-62,121,983
modglob	291	422		98	20,740,508	20,430,948
mzzv11	9,499	10,240	251	9,989	-21,718	-22,945
mzzv42z	10,460	11,717	235	11,482	-20,540	-21,623
noswot	182	128	25	75	-41	-43
nw04	36	87,482		87,482	16,862	16,311
opt1217	64	769		768	-16	-20
p2756	755	2,756		2,756	3,124	2,689
pk1	45	86		55	11	0
pp08aCUTS	246	240		64	7,350	5,481
pp08a	136	240		64	7,350	2,748
qiu	1,192	840		48	-133	-932
rout	291	556	15	300	1,078	982
set1ch	492	712		240	54,538	32,008
vpm2	234	378		168	14	10

* - počet celočíselných premenných
** - počet binárnych premenných

Tabuľka 3.2: Úlohy druhej skupiny knižnice MIPLIB 2003

Názov úlohy	Počet obmedzení	Celkový počet premenných	Počet celoč. prem.*	Počet bin. prem.**	Najlepšie známe celočíselné riešenie	Hodnota optima bez podmienok celočíselnosti
a1c1s1	3,312	3,648		192	11,503	998
aflow40b	1,442	2,728		1,364	1,168	1,006
arki001	1,048	1,388	123	415	7,580,813	7,579,600
atlanta-ip	21,732	48,738	106	46,667	90	81
danoint	664	521		56	66	63
glass4	396	322		302	1,200,012,600	800,002,400
markshare1	6	62		50	1	0
markshare2	7	74		60	1	0
mkc	3,411	5,325		5,323	-564	-612
momentum1	42,680	5,174		2,349	109,143	72,793
momentum2	24,237	3,732	1	1,808	12,314	7,225
msc98-ip	15,850	21,143	53	20,237	19,839,497	19,520,966
net12	14,021	14,115		1,063	214	17
nsrand-ipx	735	6,621		6,620	51,200	48,880
protfold	2,112	1,835		1,835	-31	-42
rd-rplusc-21	125,899	622		457	165,395	100
roll3000	2,295	1,166	492	246	12,890	11,097
seymour	4,944	1,372		1,372	423	404
swath	884	6,805		6,725	467	334
timtab1	171	397	107	64	764,772	28,694
tr12-30	750	1,080		360	130,596	14,210

* - počet celočíselných premenných
** - počet binárnych premenných

Tabuľka 3.3: Úlohy tretej skupiny knižnice MIPLIB 2003

Názov úlohy	Počet obmedzení	Celkový počet premenných	Počet celoč. prem.*	Počet bin. prem.**	Najlepšie známe celočíselné riešenie	Hodnota optima bez podmienok celočíselnosti
dano3mip	3,202	13,873		552	neznáme	576
ds	656	67,732		67,732	neznáme	57
liu	2,178	1,156		1,089	neznáme	346
momentum3	56,822	13,532	1	6,598	neznáme	91,952
sp97ar	1,761	14,101		14,101	neznáme	652,560,397
stp3d	159,488	204,880		204,880	neznáme	482
t1717	551	73,885		73,885	neznáme	134,531
timtab2	294	675	181	113	neznáme	83,592

* - počet celočíselných premenných
** - počet binárnych premenných

3.2 CPLEX 10.1

3.2.1 Úvod

Americká spoločnosť ILOG Optimalization bola založená v roku 1987 za účelom vytvoriť výkonný optimalizačný softvér na riešenie úloh lineárneho programovania. Dnes je to jedna z najznámejších spoločností zaoberajúca sa optimalizačným softvérom s vyše 850 zamestnancami po celom svete.[33] V súčasnosti má vyše 2 500 zákazníkov a medzi najznámejších patria napr. Deutsche Bank, American Express, eBay, Amazon, Delta Airlines, Lufthansa, Deutsche Telekom, Alcatel atď.[34]

CPLEX bol prvý komerčný optimalizačný softvér vytvorený v programovacom jazyku C. Názov CPLEX pochádza z kombinácie slova „C“, reprezentujúceho programovací jazyk a slova „simplex“, označujúceho simplexovú metódu na riešenie úloh lineárneho programovania.

Prvá verzia CPLEX 1.0 bola uvedená na trh v roku 1988 s podporou základnej simplexovej metódy. Počas doby spoločnosť zdokonalovala softvér a pridávala ďalšie funkcie ako napr. riešiteľa pre úlohy MIP (rok 1991) a pod. Dnes je najaktuálnejšia verzia CPLEX 11.0 uvedená na trh v októbri 2007, sľubujúca podľa výrobcu prelomové prínosy v oblasti zmiešaného celočíselného programovania.[35]

Testovanou verziou v tejto diplomovej práci je verzia CPLEX 10.1 uvedená na trh v januári 2006, ktorá bola v dobe začiatku môjho testovania najaktuálnejšia a zároveň dostupná.

Program CPLEX je vo svojej podstate riešiteľ s možnosťou tvorby vlastných úloh. Jeho súčasťou sú tzv. knižnice, ktoré umožňujú programátorom vyvíjať vlastné aplikácie využívajúce procedúry programu CPLEX. [36]

CPLEX je tvorený troma hlavnými komponentami tak, aby boli pokryté rôzne požiadavky jednotlivých užívateľov:

- *CPLEX Interaktívny Program (Interactive Optimizer)* – je spustitelný program, ktorý je schopný načítať úlohu interaktívne alebo z určitého podporovaného formátu a dodať výsledok priamo alebo ho zapísať do textového súboru. Program pozostáva zo súboru cplex.exe pre platformu Windows alebo cplex pre platformu UNIX. V mojej diplomovej práci som pracovala iba s týmto komponentom programu CLPEX.
- *Concern Technológia (Concern Technology)* – je sada C++, Java a .NET knižníc tried ponúkajúc API, ktoré umožňuje programátorom použiť riešiteľov programu CPLEX do C++, Java a .NET aplikácií.
- *CPLEX Callable Knižnica (The CPLEX Callable Library)* – je C knižnica, ktorá umožňuje programátorom použiť riešiteľov programu CPLEX do aplikácií

[33] Dostupné z <http://www.ilog.com/corporate/profile/index.cfm>, cit. 25-12-2007.
[34] Dostupné z <http://www.ilog.com/success/>, cit. 25-12-2007.
[35] Dostupné z <http://www.ilog.com/products/cplex/news/history.cfm>, cit. 10-01-2008.
[36] BARTOŇ, 2002, str. 19.

napísaných v jazyku C, Visual Basic, FORTRAN alebo akomkoľvek inom jazyku, ktorý dokáže použiť funkcie jazyka C. [37]

Komerčná verzia softvéru CPLEX stojí podľa aktuálnych informácií zo spoločnosti ILOG 15 000 EURO, čo je v prepočte podľa aktuálneho kurzu ČNB zo dňa 27.4.2008 cca 376 000 Kč pre jedného užívateľa.

3.2.2 Inštalácia

Program sa dodáva buď vo forme CD-ROM alebo je možné si ho stiahnuť priamo z internetu pomocou protokolu FTP. Program je viazaný na licenčný kľúč a súčasne môže byť nainštalovaný a fungovať len na jednom počítači, pretože licencia je viazaná na konfiguráciu hárdvéru a jej prenos na iný počítač je pomerne zložitý. [38]

Tabuľka 3.4: Podporované platformy softvérom CPLEX – platforma UNIX

Platforma Unix			
Počítač	**OS**	**C++ kompilátor**	**Java**
HP PA-RISC	HP-UX 11.x	HP aC++ A.03.73	JDK 1.4, 5.0
HP PA-RISC	HP-UX 11.x	HP aC++ A.03.73 (64-bit)	JDK 1.4, 5.0
Intel IA-64 processor family (Itanium)	HP-UX 11i (v2 and v3)	HP aC++ A.06.13 (64-bit)	JDK 1.4, 5.0
Intel IA-64 processor family (Itanium)	SUSE Linux Enterprise Server 10.0	g++ 4.1 (64-bit)	JDK 1.4, 5.0
IBM Power architecture	AIX 5.2, 5.3, 6.1	IBM XL C/C++ 7.0, 8.0	JDK 1.4, static_stl only
IBM Power architecture	AIX 5.2, 5.3, 6.1	IBM XL C/C++ 7.0 , 8.0 (64-bit)	JDK 1.4, static_stl only
SUN UltraSPARC	Solaris 9, 10	Sun Studio 8	JDK 1.4, 5.0, 6.0
SUN UltraSPARC	Solaris 9, 10	Sun Studio 8 (64-bit)	JDK 1.4, 5.0, 6.0
x64 (aka AMD64, Opteron, EM64T, ia32e)	Solaris 10	SUN Studio 11 (64-bit)	JDK 5.0, 6.0
x86	Red Hat Enterprise Linux 4.0, 5.0 (AS, ES, WS)	g++ 3.4, 4.1	JDK 1.4, 5.0, 6.0
x86-64 (AKA AMD64, Opteron, EM64T, ia32e)	Red Hat Enterprise Linux 4.0, 5.0 (AS, ES, WS)	g++ 3.4, 4.1 (64-bit)	JDK 5.0, 6.0
x86-64 (AKA AMD64, Opteron, EM64T, ia32e)	SUSE Linux Enterprise Server 9.0	g++ 3.3 (64-bit)	JDK 5.0, 6.0
x86	Mac OS X 10.4	g++ 4.0	JDK 5.0

[37] Dostupné z < http://www.ilog.com/products/cplex/product/libs.cfm>, cit. 17-01-2008.
[38] Dostupné z < http://www.ilog.com/products/jviews/documentation/jviews-framework81/doc/html/usrlicfwork/usrlicfwork.html>, cit. 17-01-2008.

Tabuľka 3.5: Podporované platformy softvérom CPLEX – platforma Windows

Platforma Windows			
Počítač	**OS**	**C++ kompilátor**	**Java**
X86	Windows XP, Server 2003	Microsoft Visual Studio .NET 2003	JDK 1.4, 5.0, 6.0
X86	Windows XP, Server 2003, Vista	Microsoft Visual Studio 2005	JDK 1.4, 5.0, 6.0
X86	Windows XP, Server 2003, Vista	Microsoft Visual Studio 2008	JDK 1.4, 5.0, 6.0
x64 (aka AMD64, Opteron, EM64T, ia32e)	Windows XP x64, Server 2003 (64-bit), Vista (64-bit)	Microsoft Visual Studio 2005 (64-bit)	JDK 5.0, 6.0
x64 (aka AMD64, Opteron, EM64T, ia32e)	Windows XP x64, Server 2003 (64-bit), Vista (64-bit)	Microsoft Visual Studio 2008 (64-bit)	JDK 5.0, 6.0

Vyššie sú uvedené platformy, ktoré softvér CPLEX podporuje.[39]

Samotná inštalácia programu je pomerne jednoduchá, doprevádzaná interaktívnym prehľadným menu. Ďalší krok k plnej funkčnosti programu je aktivácia licencie. Aj napriek pomernej zložitosti tohto úkonu, spoločnosť ILOG zabezpečila v dokumentácii bohatú podporu na toto téma, takže by sa to malo zvládnuť bez väčších problémov.

3.2.3 Možnosti softvéru CPLEX

CPLEX je univerzálny riešiteľ pre riešenie úloh lineárneho, kvadratického a zmiešaného celočíselného programovania. Program obsahuje základné algoritmy na rýchle a spoľahlivé riešenie náročných matematických problémov. Tieto algoritmy sú prístupné cez CPLEX knižnicu komponentov alebo cez interaktívny program CPLEX.

CPLEX obsahuje tri základné moduly riešiteľov:[40]

- *ILOG CPLEX Simplexoví Riešitelia (Simplex Optimizers)* – riešia lineárne a kvadratické úlohy. Obsahujú implementáciu simplexovej a duálne simplexovej metódy, rovnako ako aj sieťové metódy.
- *ILOG CPLEX Bariérový Riešiteľ (Barrier Optimizers)* – poskytuje alternatívu pre riešenie úloh lineárneho a kvadratického programovania veľkých rozmerov.
- *ILOG CPLEX Zmiešaný Celočíselný Riešiteľ (Mixed Integer Optimizers)*[41] – používa najnovšie algortimy a techniky pre výpočet ťažkých úloh zmiešaného celočíselného lineárneho programovania (MILP), úloh zmiešaného

[39] Dostupné z <http://www.ilog.com/products/cplex/product/platforms.cfm>, cit. 17-01-2008.
[40] Dostupné z <http://www.ilog.com/products/cplex/product/algorithms.cfm>, cit. 17-01-2008.
[41] Dostupné z <http://www.ilog.com/products/cplex/product/mip.cfm>, cit. 17-01-2008.

celočíselného kvadratického programovania (MIQP) a úloh zmiešaného celočíselného programovania s kvadratickými obmedzeniami (MIQCP). Zahrňuje algoritmus presolvingu, sofistikované stratégie rezných nadrovín a vhodných heuristík. Štandardné nastavenie funguje dobre pre väčšinu programov, ale užívateľ si môže prispôsobiť stratégiu prehľadávania alebo vybrať konkrétnu techniku, ktorá vyhovuje štruktúre jeho programu. Napríklad je možné si prispôsobiť proces vetvenia, vybrať si vetviacu premennú alebo kontrolovať voľbu heuristík a frekvenciu ich používania pri hľadaní celočíselného prípustného riešenia. Je taktiež možné si určiť, či chceme aby CPLEX našiel optimálne riešenie alebo našiel dobré prípustné riešenie za kratší čas. Tento riešiteľ obsahuje nasledujúce funkcie:

- Rôzne druhy rezných plánov,
  * *clique cuts*,
  * *cover cuts*,
  * *disjunctive cuts*,
  * *flow cover cuts*,
  * *flow path cuts*,
  * *gomory fractional cuts*,
  * *GUB cover cuts*,
  * *implied bound cuts*,
  * *mixed integer rounding (MIR) cuts*,
  * *zero half cuts*,
- special ordered sets,
- heuristiky,
- možnosti nastavenia optimality a prípustnosti riešenia,
- algoritmy presolvingu a postsolvingu,
- breadth-first, best-first alebo depth-first search,
- užívateľsky definované priority procesu vetvenia a jeho smeru,
- užívateľom zvolený algoritmus voľby vetviaceho uzlu a vetviacej premennej,
- rôzne možnosti voľby úvodnej relaxácie,
- cut-off a shortcut techniky,
- prispôsobitelné stratégie vetvenia,
- možnosti kontroly pamäte užívateľom,
- sondovanie.

Program CPLEX sa spúšta v príkazovom riadku príkazom : *cplex*.

Pokiaľ je všetko správne nainštalované, mala by sa na obrazovke zobraziť hláška:

```
Welcome to CPLEX Interactive Optimizer 10.0.0
with Simplex, Mixed Integer & Barrier Optimizers
Copyright (c) ILOG 1997-2006
CPLEX is a registered trademark of ILOG
Type help for a list of available commands.
Type help followed by a command name for more
information on commands.
CPLEX>
```

Program sa ďalej ovláda zapisovaním jednotlivých príkazov do príkazového riadku, a to buď pomocou skratiek alebo ich celých názvov. CPLEX nie je citlivý na to, či sa používajú v príkazoch veľké alebo malé písmená.

Na vyvolanie nápovedy (help) sa používa príkaz *help* alebo len písmeno *h*. Po jej vyvolaní sa zobrazí prehľad základných príkazov:

Tabuľka 3.6: Prehľad základných príkazov programu CPLEX

Príkaz	Význam
add	- pridaj ďalšie obmedzenie do úlohy
baropt	- rieš úlohu pomocou barierového algoritmu
change	- otvor menu s možnosťami zmien v načítanej úlohe
display	- zobraz úlohu, jej riešenie alebo nastavené parametre
enter	- zadaj novú úlohu
feasopt	- nájdi relaxáciu k neprípustnej lineárnej úlohe
help	- vyvolaj nápovedu v danej úrovni menu
mipopt	- spusti riešiteľa úloh MIP
netopt	- spusti riešiteľa sieťových úloh
optimize	- optimalizuj
primopt	- spusti riešenie úlohy primárne simplexovou metódou
quit	- ukonči chod programu
read	- načítaj úlohu zo súboru
set	- otvor menu s nastaveniami parametrov programu
tranopt	- spusti riešenie úlohy duálne simplexovou metódou
write	- zapíš úlohu alebo riešenie do súboru
xecute	- spusti príkaz z operačného systému

Podrobné informácie o každom jednom príkaze získame spustením nápovedy v nasledujúcej forme:

CPLEX> help názov príkazu, prípadne

CPLEX> h skratka názvu príkazu.

CPLEX poskytuje viacero možností vloženia vstupných dát. Najčastejším spôsobom je načítanie už vytvorených súborov pomocou príkazu *read*.

Druhou možnosťou je zápis dát priamo do programu. Nová úloha sa zadáva pomocou príkazu *write* a odpovedá rovnicovému zápisu matematického modelu. Nakoľko vlastným formátom programu CPLEX je formát LP, potom bude tento súbor automaticky uložený vo formáte LP. Existuje tu samozrejme možnosť uloženia súboru aj v inom podporovanom formáte.

CPLEX podporuje celkovo 18 typov výstupných formátov, ktoré je samozrejme schopný aj spätne načítať. Niektoré z nich sú obecné formáty, ktoré je možné použiť aj v iných optimalizačných softvéroch a niektoré sú formáty slúžiace výlučne pre vnútornú potrebu programu CPLEX, a teda nepoužiteľné akýmkoľvek iným softvérom. Medzi podporované formáty patria napríklad :

- *MPS* – je obecným štandardom, tento formát bol detailne popísaný v kapitole 2.4,
- *LP* – je vlastný formát programu CPLEX určený k zápisu úlohy v matematickej podobe,
- *NET* – je formát, ktorý sa zapisuje pomocou ASCII kódu a slúži pre popis sieťových a tokových úloh,
- *PRM* – súbor obsahuje informácie o nastaveniach parametrov Callable knižnice,
- *BAS* – obsahuje informácie o bazických premenných úlohy.
- *SAV* – slúži k vnútornej potrebe programu a uchováva informácie o aktuálnom riešení úlohy,
- *MST* – je taktiež formát programu CPLEX a používa sa pri zápise úloh zmiešaného celočíselného programovania,
- *ORD* – vlastný formát programu CPLEX, ktorý používa riešiteľ úloh MIP a obsahuje informácie o prioritách poradia a smeroch vetvenia špecifických premenných,
- *SOL* – je formát pre zápis výsledkov,
- atď.

Významným súborom je *cplex.log* , kam sa ukladajú informácie o všetkom dianí po spustení systému CPLEX ako napr. zmeny v nastavení systému, zmeny v spôsobe výpočtu a najdôležitejšia informácia, výsledky riešenia jednotlivých úloh.[42]

Výpočet je možné kedykoľvek ukončiť stlačením kláves CTLR + C.

3.2.4 Riadenie výpočtu a príkazy

Po tom, čo načítame úlohu príkazom *read*, je možné si nastaviť jednotlivé parametre programu CPLEX. Jednotlivé nastavenia sa líšia podľa charakteru riešenej úlohy. Všetky zmeny sa automaticky zapisujú do súboru *cplex.log*.

Nastavenia programu CPLEX vyvoláme riadkovým príkazom *set*, po ktorom sa nám zobrazí menu s položkami zobrazenými v tabuľke na ďalšej strane.

V prípade, že sme zmenili nastavenia parametrov a neskôr chceme vrátit hodnoty parametrov do ich pôvodných prednastavených hodnôt, použijete príkaz *set default.*

[42] BARTOŇ, 2002, str. 21.

Tabuľka 3.7: Nastavenia parametrov programu CPLEX

Príkaz	**Význam**
advanced	nastavuje parametre pre opakované spúšťanie už vyriešenej úlohy
barrier	nastavuje parametre barierového riešiteľa
clocktype	nastavuje typ hodín, ktoré merajú čas výpočtu
conflict	nastavuje parametre pre hľadanie konfliktov
defaults	nastavuje všetky parametre do pôvodných hodnôt
emphasis	nastavuje optimalizačný dôraz
feasopt	nastavuje parametre pre funkciu *feasopt*
logfile	nastavuje súbor, do ktorého sa zapisujú informácie o výsledkoch výpočtu
lpmethod	nastavuje metódu pre lineárnu optimalizáciu
mip	nastavuje parametre riešiteľa úloh MIP
network	nastavuje parametre pre sieťovú optimalizáciu
output	nastavuje typ a umiestnenie výstupu
preprocessing	nastavuje parametre preprocessingu
qpmethod	nastavuje parametre pre kvadratickú optimalizáciu
read	nastavuje parametre načítania úlohy
sifting	nastavuje parametre pre „*sifting*" optimalizáciu
simplex	nastavuje parametre simplexovej a duálne simplexovej metódy
threads	nastavuje predvolený počet paralelných ciest
timelimit	nastavuje časový limit pre výpočet úlohy v sekundách
workdir	nastavuje adresár pre priebežné ukladanie dočasných súborov
workmem	nastavuje veľkosť pamäte v MB použiteľnú pre úschovu pracovných súborov

Pre potreby výpočtu úloh zmiešaného celočíselného programovania, je možné nastaviť parametre riešiteľa úloh MIP, a to napr. parametre týkajúce sa metódy vetvenia a medzí, prehľadávania, rezov, heuristík a preprocessingu atď. Menu s parametrami sa zobrazí pomocou príkazu *set mip.* Počet nastaviteľných parametrov je pomerne veľký, takže ďalej uvádzam len tie najdôležitejšie z nich :

- *CUTS* - nastavuje aké rezy má riešiteľ použiť pri optimalizácii úloh zmiešaného celočíselného programovania. Na výber sú:
  - *cliques*,
  - *covers*,
  - *disjunctive*,
  - *flowcovers*,
  - *gomory*,
  - *gubcovers*,
  - *implied*,
  - *mircut (mixed integer rounding cut)* ,
  - *pathcut*,
  - *zerohalfcut*.
- *DISPLAY* - nastavuje úroveň zobrazenia jednotlivých uzlov výpočtového stromu:
  - *0* - nezobrazuje nič,
  - *1* - zobrazuje celočíselné prípustné riešenia,
  - *2* - zobrazuje uzly pod kontrolou tzv „mip intervalu“,
  - *3* - rovnaké ako 2, ale zobrazuje navyše informácie o použitých rezoch v uzle,
  - *4* - rovnaké ako 3, ale zobrazuje navyše informácie o koreňovom uzle,
  - *5* - rovnaké ako 3, ale zobrazuje navyše informácie o všetkých uzloch.
- *INTERVAL* - nastavuje interval pre tlač zobrazenia zmiešaného celočíselného uzla.
- *LIMITS* - nastavuje limity pre optimalizáciu úloh MIP:
  - *aggforcut* - nastavuje limity agregácie obmedzení pre tvorbu rezov,
  - *cutpasses* - nastavuje počet priebehov generovania rezných plánov,
  - *eachcutlimit* – nastavuje limit tvorby rezov každého druhu počas jedného priebehu generovania,
  - *gomorypass* – nastavuje limity priebehu generovania Gomoryho rezov,
  - *nodes* – nastavuje limity uzlov metódy vetvenia a rezov,
  - *polishtime* – nastavuje časové limity pre „leštenie“ optimálneho riešenia,
  - *probetime* – nastavuje časové limity pre sondovanie,
  - *repairtries* – nastavuje počet pokusov pre úpravu heuristík,
  - *solutions* – nastavuje limity pre zmiešané celočíselné riešenia,
  - *strongit* – nastavuje limity pre iterácie silného vetvenia,
  - *treememory* – nastavuje horný limit pre veľkosť výpočtového stromu v megabitoch.

- *ORDERTYPE* – určuje typ zvoleného prioritného poradia:
  - *0* - žiadne priority,
  - *1* - klesajúce ceny,
  - *2* - rastúci rozsah medzí,
  - *3* - rastúca cena pre výpočet koeficientu.
- *POOL* – nastavuje charakteristiky spoločného fondu riešení tzv. „*solution pool*“:
  - *absgap* - nastavuje abolútny cieľový rozdiel od optimality,
  - *capacity* - nastavuje kapacitu spoločného fondu riešení,
  - *intensity* - nastavuje intenzitu pridávania riešení do spoločného fondu,
  - *relgap* - nastavuje relatívny cieľový rozdiel od optimality,
  - *replace* - nastavuje pravidlá stratégie nahradzovania riešení v spoločnom fonde.
- *STRATEGY* – určuje stratégiu pre optimalizáciu zmiešaných celočíselných úloh:
  - *backtrack* - nastavuje faktor pre tzv.“*backtracking*“,
  - *branch* – určuje smer prvej vetvy,
  - *dive* – nastavuje stratégiu tzv. „*divingu*“,
  - *file* – určuje súbor, do ktorého budú ukladané uzly v prípade, že bude dosiahnutý limit pamäte pre výpočtový strom,
  - *heuristucfreq* – určuje frekvenciu periodického použitia heuristík,
  - *nodeselect* – nastavuje stratégiu výberu uzlu,
  - *order* – nastavuje indikátor pre použitie prioritného poradia,
  - *probe* – nastavuje sondovanie,
  - *search* – nastavuje indikátor pre prehľadávaciu metódu,
  - *startalgorithm* – nastavuje algoritmus pre riešenie úvodnej relaxácie,
  - *subalgorithm* – nastavuje algoritmus pre riešenie podúloh,
  - *variableselect* – nastavuje sratégiu výberu premennej.
- *TOLERANCES* – nastavuje prípustnú odchylku optimalizácie zmiešaných celočíselných úloh:
  - *absmipgap* – nastavuje absolútnu toleranciu MIP rozdielu od optimality,
  - *integrality* – nastavuje toleranciu celočíselnosti.

Výhodou systému CPLEX je, že sa všetky nastavenia uchovávajú aj pre ďalšie výpočty a zároveň sú všetky informácie o zmenách uložené aj do súboru *cplex.log*, takže pre užívateľa nie je problém zistiť pri akých parametrov bola počítaná daná úloha.

3.2.5 Priebeh výpočtu a výpočtová procedúra

Systém CPLEX používa pri výpočtoch niekoľko postupov ako dosiahnuť maximálneho efektu. Automatická utilita napríklad otestuje dostupnú pamäť počítača a pozmení algoritmus tak, aby boli dosiahnuté čo najlepšie výsledky. To však môže spomaliť výpočet, hlavne u starších počítačov s menšou vnútornou pamäťou. Ďalší diagnostický nástroj zanalyzuje maticu premenných a vypočíta „ kondíciu matice“ , čo je číslo, ktoré meria citlivosť lineárneho riešenia na zmeny v matici úlohy a slúži

programu CPLEX pri ďalšom rozhodovaní o použitých algoritmoch. Táto diagnóza môže trvať pomerne dlho a pokiaľ sa užívateľ pokusí ovplyvniť jej priebeh napríklad necitlivým nastavením príliš tvrdých podmienok, nemusi CPLEX vôbec nájsť optimálne riešenie. Pre neskúseného užívateľa je teda lepšie nechať všetky parametre v pôvodnom nastavení.

Pri mojom testovaní som nezasahovala do jednotlivých nastavení parametrov programu CPLEX a všetky som nechala vo svojich prednastavených hodnotách.

Výhodou systému CPLEX je, že je počas celej doby možné sledovať priebeh výpočtu, a tak mať prehľad čo sa počas doby spracovávania úlohy deje.

Postup je u všetkých úloh rovnaký. Hneď po zadaní riadkového príkazu *mipopt*, aplikuje CPLEX procedúry *PRESOLVE* a *AGGREGATOR*, aby zredukovali načítanú úlohu a zjednodušili tak hľadanie optima. Tieto procedúry je možné vypnúť v menu *SET PREPROCESSING*. Je možné vynechať danú procedúru v prípade, že načítame úlohu s dostatočnou informáciou o premenných, ktoré majú byť bazické.

Ďalším krokom je generovanie rezov, prídavných obmedzení, ktoré „obrezávajú“ neceločíselné riešenia, čo v dôsledku vedie k zníženiu počtu uzlov a vetví, ktoré je nutné preskúmať. Program CPLEX generuje viacero druhov rezov napr. „*cliques*“ a „*covers*“ a iné (viz vyššie).

Po vygenerovaní rezov sa aplikujú heuristiky na získanie počiatočného celočíselného riešenia. Potom začne prebiehať výpočet metódou vetvenia a medzí. Na obrazovke sa generuje priebeh výpočtu a report, ktorý sa po získaní optimálneho riešenia zapíše do súboru *cplex.log*. Behom celej doby sa generuje výpočtový strom, ktorý sa priebežne ukladá do dočasného súboru, ktorý môže mať až niekoľko stoviek megabitov.[43]

3.2.6 Výsledky testovania

K testovaniu boli použité úlohy zmiešaného celočíselného programovania z knižnice MIPLIB 2003, ktoré sa používajú ako štandard pri porovnávaní výkonnosti optimalizačných softvérov. Celkový počet úloh z knižnice MIPLIB 2003 je 60 a delia sa na 3 skupiny:

1. Úlohy, ktoré by mal byť komerčný softvér schopný vypočítať do 1 hodiny. Celkovo sa jedná o 31 úloh.
2. Úlohy, ktorých optimálne riešenie je známe. Celkovo sa jedná o 21 úloh.
3. Úlohy, ktorých optimálne riešenie sa zatiaľ nepodarilo vypočítať. Celkovo sa jedná o 8 úloh.

Žiadne parametre softvéru CPLEX som pri testovaní nemenila a všetky zostali vo svojich prednastavených hodnotách.

[43] BARTOŇ, 2002, str. 25-26.

3.2.6.1 Prvá skupina riešených úloh

Ako prvá je skupina úloh, ktoré by mal byť komerčný softvér schopný vyriešiť do jednej hodiny. Celkovo je týchto úloh 31. Samotný výpočet prebiehal bez väčších problémov, až na dve úlohy, konkrétne *aflow30a* a *disctom*, ktoré CPLEX nedokázal ani pri opakovanom pokuse načítať a vypísal chybu. U ostatných úloh, okrem úloh *cap6000*, *fiber*, *gesa*, *gesa2-o*, *hapr2*, *mod011*, *modglob*, *qui* a *set1ch* bolo vypočítané optimálne riešenie. V prípade týchto vybraných deviatich úloh bolo vypočítané uspokojivé riešenie, ktoré sa, okrem úlohy *qui*, od optimálneho riešenia líši len minimálne, teda nie viac ako o 1%. U úlohy *qui* je rozdiel vypočítaného riešenia a optimálneho riešenia až 99899.9% a toto riešenie sa teda nedá považovať za uspokojivé.

Čo sa týka časového skóre, CPLEX dopadol pomerne dobre. Až na jednu úlohu *manna81* a samozrejme tie úlohy, ktoré nebol schopný načítať, sa mu podarilo vypočítať riešenia ostatných úloh v časovom horizonte 1 hodina.

Na ďalšej stránke je uvedená tabuľka s výsledkami jednotlivých úloh z prvej skupiny knižnice MIPLIB 2003 vypočítaná softvérom CPLEX.

3.2.6.2 Druhá skupina riešených úloh

Druhá skupina obsahuje úlohy, ktorých optimálne riešenie je známe. Jedná sa o komplikovanejšie úlohy, prevažne väčších rozmerov, u ktorých sa predpokladá, že ich výpočet bude trvať dlhšiu dobu. Ako maximálnu dobu pre riešenie jednej úlohy som zvolila 12 hodín, a to z dôvodu časového harmonogramu testovania.

V tejto skupine bolo vypočítané optimálne riešenie pre 5 úloh z celkového počtu 21, a to pre úlohy *aflow40b*, *danoint*, *msc98-ip*, *nsrand-ipx* a *tr12-30*. V siedmych ďalších prípadoch našiel CPLEX uspokojivé riešenie, čo znamená, že nájdené a optimálne riešenie sa nelíšia o viac ako 1%. Zvyšných 9 riešení sa od optima líši o viac ako 1%, a teda ich nepovažujem za uspokojivé.

Čo sa týka časového hľadiska, z celkového počtu 21 úloh, bolo vyriešených 12 úloh do 12 hodín a u zvyšných 9 úloh sa výpočet neskončil do časového limitu 12 hodín a musel byť prerušený.

Tabuľka s výsledkami jednotlivých úloh sa nachádza o dve stránky ďalej.

3.2.6.3 Tretia skupina riešených úloh

Tretiu skupinu tvoria úlohy *dano3mpi*, *ds*, *liu*, *momentum3*, *sp97ar*, *atp3d*, *t1717*, *timtab2*. Spoločným znakom týchto úloh je, že ich optimálne riešenie zatiaľ nebolo nájdené. Výsledok testovania softvéru CPLEX na týchto úlohách bohužiaľ nebude veľkým prekvapením. Ani jednu z týchto úloh sa softvéru CPLEX nepodarilo vypočítať.

Tabuľka 3.8: Výsledky riešenia úloh softvérom CPLEX – prvá skupina úloh

Názov úlohy	Výpočtový čas v sekundách	Najlepšie vypočítané riešenie	Optimálne riešenie**	Absolútny rozdiel***	Relatívny rozdiel****	Poznámky
10teams	21.4	924.0	924.0	0.0	0.0%	
aflow30a	n/a	n/a	1,158.0	n/a	n/a	Chyba *
air04	31.0	56,137.0	56,137.0	0.0	0.0%	
air05	23.2	26,374.0	26,374.0	0.0	0.0%	
cap6000	0.9	-2.45E+06	-2.45E+06	-55.0	0.0%	
disktom	n/a	n/a	-5.00E+03	n/a	n/a	Chyba *
fast0507	3,136.8	174.0	174.0	0.0	0.0%	
fiber	0.4	4.06E+05	4.06E+05	-0.2	0.0%	
fixnet6	1.0	3,983.0	3.98E+03	0.0	0.0%	
gesa2	0.5	2.58E+07	2.58E+07	-131.4	0.0%	
gesa2-o	4.1	2.58E+07	2.58E+07	43.6	0.0%	
harp2	2,311.8	-7.39E+07	-7.39E+07	-16,221.0	0.0%	
manna81	5,477.0	-13,164.0	-13,164.0	0.0	0.0%	
mas74	1,088.2	11,801.2	11,801.2	0.0	0.0%	
mas76	123.2	40,005.1	40,005.1	0.0	0.0%	
misc07	12.9	2,810.0	2,810.0	0.0	0.0%	
mod011	128.2	-5.46E+07	-5.46E+07	35.0	0.0%	
modglob	0.2	2.07E+07	2.07E+07	-8.1	0.0%	
mzzv11	640.2	-21,718.0	-21,718.0	0.0	0.0%	
mzzv42z	130.1	-20,540.0	-20,540.0	0.0	0.0%	
noswot	2,790.1	-41.0	-41.0	0.0	n/a	
nw04	56.4	16,862.0	16,862.0	0.0	0.0%	
opt1217	1,119.2	-16.0	-16.0	0.0	0.0%	
p2756	0.5	3,124.0	3,124.0	0.0	0.0%	
pk1	83.3	11.0	11.0	0.0	0.0%	
pp08aCUTS	3.0	7,350.0	7,350.0	0.0	0.0%	
pp08a	0.9	7,350.0	7,350.0	0.0	0.0%	
qiu	58.4	-132.9	-1.33E+05	-1.33E+05	99899.9%	
rout	32.8	1,077.6	1,077.6	0.0	0.0%	
set1ch	0.7	54,540.3	54,537.8	-2.4	0.0%	
vpm2	1.3	13.8	13.8	0.0	0.0%	

***** - chyba pri načítaní súboru

** - známe optimálne riešenie[44]

******* - vyjadruje absolútny rozdiel najlepšieho vypočítaného riešenia od optimálneho riešenia

******** - vyjadruje relatívny rozdiel najlepšieho vypočítaného riešenia od optimálneho riešenia

[44] Dostupné z <http://miplib.zib.de/miplib2003.php>, cit. 05-04-2008.

Tabuľka 3.9: Výsledky riešenia úloh softvérom CPLEX – druhá skupina úloh

Názov úlohy	Výpočtový čas v sekundách	Najlepšie vypočítané riešenie	Optimálne riešenie**	Absolútny rozdiel***	Relatívny rozdiel****	Poznámky
a1c1s1	Viac ako 12 hodín	10,486.9	11,503.4	1,016.5	9.7%	Prerušenie výpočtu *
aflow40b	34,057.4	1,168.0	1,168.0	0.0	0.0%	
arki001	695.0	7.58E+06	7.58E+06	-266.3	0.0%	
atlanta-ip	Viac ako 12 hodín	83.6	90.0	6.4	7.6%	Prerušenie výpočtu *
danoint	42,058.3	65.7	65.7	0.0	0.0%	
glass4	7,164.0	1.58E+09	1.20E+09	-3.75E+08	23.8%	
markshare1	18,472.0	50.0	1.0	-49.0	98.0%	
markshare2	1,668.9	22.0	1.0	-21.0	95.5%	
mkc	Viac ako 12 hodín	-563.6	-563.8	-0.2	0.0%	Prerušenie výpočtu *
momentum1	Viac ako 12 hodín	1.09E+05	1.09E+05	-14.5	0.0%	Prerušenie výpočtu *
momentum2	Viac ako 12 hodín	12,314.1	12,314.2	0.1	0.0%	Prerušenie výpočtu *
msc98-ip	40,301.0	1.98E+07	1.98E+07	0.0	0.0%	
net12	Viac ako 12 hodín	255.0	214.0	-41.0	16.1%	Prerušenie výpočtu *
nsrand-ipx	35,601.0	51,200.0	51,200.0	0.0	0.0%	
protfold	Viac ako 12 hodín	-24.0	-31.0	-7.0	29.2%	Prerušenie výpočtu *
rd-rplusc-21	Viac ako 12 hodín	100.0	1.65E+05	1.65E+05	-165295.0%	Prerušenie výpočtu *
roll3000	38,469.0	12,920.0	12,890.0	-30.0	0.2%	
seymour	Viac ako 12 hodín	425.0	423.0	-2.0	0.5%	Prerušenie výpočtu *
swath	2,763.0	521.4	467.4	-54.0	10.4%	
timtab1	15,532.0	7.72E+05	7.65E+05	-7,226.0	0.9%	
tr12-30	13,577.0	1.31E+05	1.31E+05	0.0	0.0%	

***** - výpočet neskončil do 12 hodín, nutné prerušenie výpočtu
** - známe optimálne riešenie[45]
******* - vyjadruje absolútny rozdiel najlepšieho vypočítaného riešenia od optimálneho riešenia
******** - vyjadruje relatívny rozdiel najlepšieho vypočítaného riešenia od optimálneho riešenia

[45] Dostupné z <http://miplib.zib.de/miplib2003.php>, cit. 05-04-2008.

3.2.7 Zhrnutie

Optimimalizačný softvér CPLEX od spoločnosti ILOG je výkonný a prehľadný nástroj pre riešenie rôznych druhov optimalizačných úloh. Jeho hlavná prednosť je jednoduché ovládanie, prepracovaný systém menu a kvalitne a prehľadne pripravená nápoveda. Výstupy, ktoré CPLEX poskytuje sú prehľadné, s možnosťou nastavenia ich podrobnosti. Veľkou výhodou je taktiež aj možnosť sledovania výpočtu priamo na obrazovke, rovnako ako aj ukladanie výstupov do zvoleného súboru. Neobvyklé je priebežné ukladanie výsledkou všetkých výpočtov do jedného súboru *cplex.log*, ktorý sa však pri spracovaní viacero úloh stáva pomerne neprehľadný a veľký. Jeho výhoda je, že sa doň ukladajú všetky zmeny v nastaveniach parametrov systému, a teda je možné zistiť pri akom nastavení prebiehal výpočet. Ďalšia výhoda programu CPLEX je jeho kompatibilita s operačnými systémami UNIX a Mac OS.

Pri mojom testovaní som použila verziu CPLEX 10.1. Nemenila som žiadne parametre a všetky hodnoty som nechala vo svojich prednastavených hodnotách.

Pri testovaní prvej skupiny úloh z knižnice MIPLIB 2003 dve úlohy nevedel CPLEX vôbec načítať, pre 20 úloh vypočítal optimálne riešenie a pre ďalších 9 úloh z prvej skupiny našiel uspokojivé riešenie. Len pre jednu vypočítal riešenie, ktoré sa od optimálneho líšilo o viac ako 1% a je považované za neuspokojivé. Z časového hľadiska zvládol túto skupinu úloh CPLEX veľmi dobre. Až na jednu úlohu poskytol riešenie v kratšej dobe ako je jedna hodina.

V prípade testovania komplikovanejších úloh z druhej skupiny úloh knižnice MIPLIB 2003 vypočítal CPLEX optimálne riešenie pre 5 úloh, v 7 ďalších prípadoch našiel uspokojivé riešenie a vo zvyšných 9 úlohách nebolo vypočítané uspokojivé riešenie. Z časového hľadiska bolo vyriešených 12 úloh do časového limitu 12 hodín a vo zvyšných 9 prípadoch musel byť výpočet prerušený.

CPLEX považujem za dobrú voľbu pre svoju jednoduchosť ovládania a prehľadnosť poskytovaných výsledkov. Veľká výhoda, je aj množstvo možností rozšírenia použitia a implementácie systému do iných aplikácií. Pre užívateľa je výhoda dobre pripravená nápoveda a veľké množstvo nastaviteľných parametrov pre ovládanie výpočtu a nastavenia jednotlivých algoritmov. Rýchlosť výpočtou je pomerne vysoká a presnosť výsledkou je taktiež veľmi dobrá.

3.3 Xpress 18.00

3.3.1 Úvod

Optimalizačný softvér Xpress je produktom britskej spoločnosti Dash Optimization založenej v roku 1984. Po dlhej vzájomnej spolupráci bola spoločnosť Dash v januári tohto roku kúpená americkou korporáciou Fair Isaac, ktorá sa venuje analytickej a manažerskej technológii pre podporu rozhodovania.[46] Medzi jej zákazníkov patria napríklad spoločnosti ako Amazon.com, City of New York, London Electricity, Pacific Gas, Unilever, Proctel and Gamble atď.[47]

Hlavným produktom spoločnosti je Xpress-MP, čo je sada programov pre matematické modelovanie a optimalizáciu úloh lineárneho, celočíselného, kvadratického, nelineárneho a stochastického programovania. Do tejto sady patria:

- Riešitelia -
  - *Xpress-Optimizer* – umožňuje riešiť úlohy lineárneho (LP), zmiešaného celošíselného (MIP), kvadratického (QP) a zmiešaného kvadratického programovania (MIQP). Tento riešiteľ bol použitý pri testovaní a bude detailnejšie popísaný ďalej v texte.
  - *Xpress-SLP* – rieši problémy nelineárneho a zmiešaného nelineárneho programovania.
  - *Xpress-SP* – umožňuje riešiť úlohy stochastického programovania.
  - *Xpress-Kalis* – zameriava sa na riešenie diskrétnych kombinatorických problémov.
- Nástroje pre tvorbu a rozvoj modelov -
  - *Xpress-Mosel* – je pokročilý jazyk a prostredie pre modelovanie a riešenie optimalizačných úloh. Umožňuje formulovať vlastný problém a potom ho vyriešiť pomocou jedného z riešiteľov Xpressu. Toto prostredie zahrňuje jazyk Mosel, moduly a I/O ovládače pre možnosť prístupu k iným softvérom, externým dátam a knižnice pre možnosť implementácie modelov do iných aplikácií.
  - *Xpress-BCL* – je objektovo orientovaná knižnica pre tvorbu, riešenie a analýzu úlohy priamo v rámci zvolenej aplikácie .
  - *Xpress-IVE* – je kompletné vizuálne prostredie pre operačný systém Windows.
  - *XAD (Xpress – Applacation Developer)* – je rozšírenie programu Xpress-Mosel o API pre vývoj grafického užívateľského prostredia.[48]

Prvá verzia, ktorú mala naša škola k dispozícii, bola verzia Xpress 11.00. Momentálne najnovšia verzia je Xpress-MP 2007B s riešiteľom *Xpress-Optimizer* 18.10, ktorá bola uvedená na trh v novembri 2007. V mojej diplomovej práci som testovala verziu Xpress 18.00, ktorá bola v dobe začiatku môjho testovania aktuálna a dostupná.

---

[46] Dostupné z <http://www.fairisaac.com/NR/exeres/AAF3DD63-ABE9-4FA7-B432-91E5B671224E,frameless.htm>, cit. 10-04-2008.

[47] Dostupné z <http://www.dashoptimization.com/home//users/end_users.html>, cit. 10-04-2008.

[48] Dostupné z <http://www.dashoptimization.com/home//products/products_overview.html>, cit. 10-04-2008.

Xpress sa predáva vo viacerých verziách. Vo verzii *Hyper* a *Hyper 64* nemá Xpress žiadne obmedzenia na počet premenných a obmedzení. Ďalej je dodávaný vo verziách *Extended* (maximálne 24 000 premenných a 12 000 obmedzení), *Professional* (maximálne 5 000 premenných a 2 000 obmedzení) a spoločnosť ponúka aj bezplatnú študentskú verziu (maximálne 400 obmedzení a 800 premenných). K dispozícii je i verzia *Parralel Xpress-MP*, ktorá umožňuje vzdielať veľké súbory a počítať ich na viacero procesoroch súbežne.

Ako jediný testovaný softvér má Xpress grafické rozhranie, ale len pre platformu Windows. Ďalšia možnosť je riadiť výpočty pomocou riadkových príkazov. V mojej diplomovej práci sa budem podrobnejšie venovať popisu fungovania grafického prostredia *Xpress-IVE*, nakoľko som ho aktívne používala pri testovaní.

Výhodou softvéru Xpress je možnosť dokúpenia modulu *Xpress connect* pre podporu rozhrania *ODBC(Open DataBase Connectivity)*, ktorá umožňuje import z tabuľkových a databázových programov ako je napr. Excel, dBASE, SQL server, Oracle atď.

Plná neobmedzená verzia programu Xpress, ktorá by obsahovala všetky základné prvky pre prácu a riešenie úloh zmiešaného celočíselného programovania, by podľa aktuálneho cenníka stála 18 600 EUR, čo je v prepočte podľa aktuálneho kurzu ČNB zo dňa 27.4.2008 cca 466 500 Kč. Cena je uvedená za product *Xpress-MP Mega Bundle* a licenciu *Hyper64*.

3.3.2 Inštalácia

Program Xpress je dodávaný vo forme CD-ROM alebo je možné si ho stiahnuť priamo na webových stránkach spoločnosti Dash Optimization. Rovnako ako aj program CPLEX je chránený hárdvérovým kľúčom. Je však možné si zakúpiť licenciu, ktorá je viazaná na USB kľúč, a tak je umožnená mobilita použitia medzi rôznymi počítačmi.[49]

Tabuľka na ďalšej stránke zobrazuje počítačové platformy, ktoré softvér Xpress podporuje.[50]

Samotná inštalácia je jednoduchá a počas celého priebehu doprevádzaná prehľadnými inštrukciami.

[49] Xpress-MP Eurozone (€) Price List September 2007, Dash Optimization, 2007, str.1-2.
[50] Dostupné z <http://www.dashoptimization.com/home/products/platforms/platform_policy.html>, cit. 11-04-2008.

Tabuľka 3.10: Podporované počítačové platformy softvérom Xpress

Platforma	Operačný systém	Procesor
Windows 32bit	Windows 2000, Windows XP, Windows server 2003, Vista	AMD Duron/Intel Pentium3 or later x86 CPU
Windows 64bit	Windows XP, Windows server 2003, Vista	Any AMD64 or Intel EM64T enabled 64bit CPU
Windows 64bit	Windows XP, Windows server 2003	Itanium 2
Linux 32bit	RedHat 9 (glibc 2.3+) compatible	AMD Duron/Intel Pentium3 or later x86 CPU
Linux 64bit	RedHat 9 (glibc 2.3+) compatible	Any AMD64 or Intel EM64T enabled 64bit CPU
Solaris 32bit	Solaris 7-10	Sun Ultra Sparc 3 or later
Solaris 64bit	Solaris 7-10	Sun Ultra Sparc 3 or later
Solaris 64bit	Solaris 10	Any AMD64 or Intel EM64T enabled 64bit CPU
AIX 32bit	AIX 5.1	IBM Power PC
AIX 64bit	AIX 5.1	IBM Power PC
HP-UX 32bit	HP-UX 11+	HP PA-RISC
HP-UX 64bit	HP-UX 11+	HP PA-RISC
HP-UX 64bit	HP-UX 11+	Itanium 2

Samotná inštalácia je pomerne jednoduchá a počas celého priebehu doprevádzaná jednoduchými inštrukciami.

3.3.3 Možnosti softvéru Xpress

Po inštalácii sa program spustí v ponuke *Štart → Programy → Xpress-MP* a kliknutím na ikonku *Xpress IVE*, prípadne príkazom *ive* v DOSovskom okne. Spustí sa tým grafické prostredie, ktoré umožňuje jednak vytvárať a riešiť nové modely v jazyku Mosel alebo riešiť už existujúce modely zapísané v súboroch v podporovaných formátoch.[51]

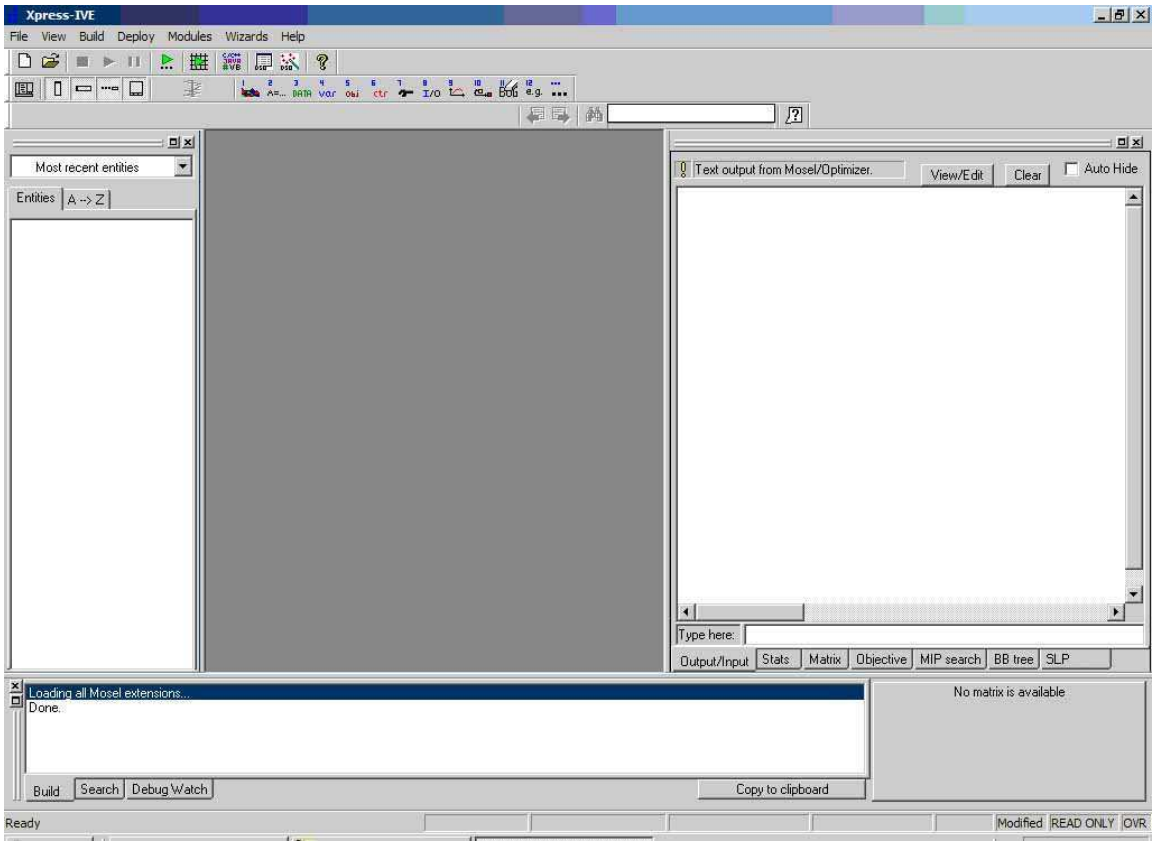
V ponuke *Štart → Programy → Xpress-MP* sa nám ponúkajú ešte ďalšie tri voľby:

- *Online Documentation* - spustí webovú dokumentáciu pre program Xpress-MP.
- *Readme* - spustí webovú stránku obsahujúcu základné údaje o verzii programu a kontakty na technickú podporu.
- *Xpress Host ID tool* - spustí program, ktorý vygeneruje základné údaje o počítači, na ktorý je viazaná licencia.

[51] Xpress-MP Getting Started, Dash Optimization, 2007, str.12.

Po kliknutí na ikonku *Xpress IVE* sa otvorí nasledujúce okno:

Obr. 3.1: Úvodná obrazovka po spustení programu *Xpress IVE*



V hornej časti sa nachádza základné menu a paneli nástrojov. V prílohe č.3 uvádzam bližší popis ich základných komponentov.

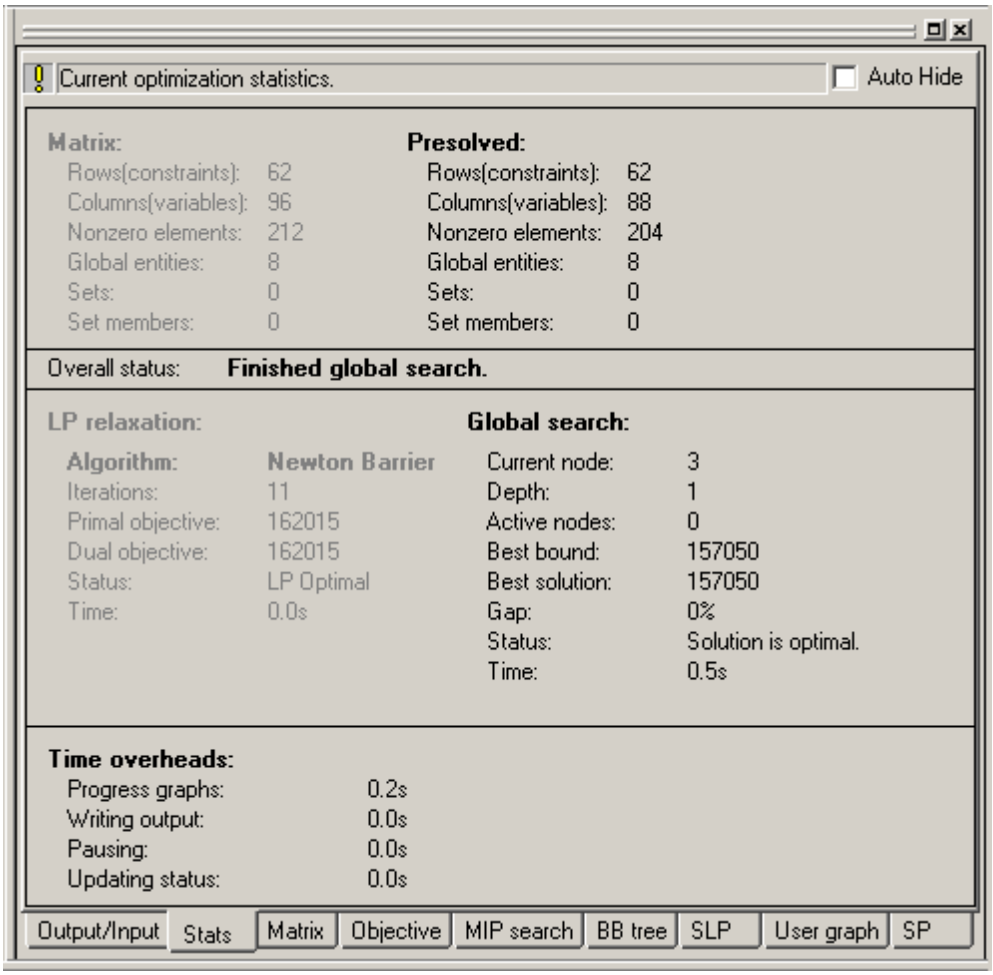
Pracovná plocha je rozdelená do štyroch základných okien. Ľavé okno sa nazýva *Project Bar* a po kompilácii Mosel modelu sa v ňom vytvorí sústava objektov odkazujúca na model. Umožňuje potom prehľadávať model nie len podľa poradia v ktorom bol vytvorený, ale je možné v ňom vyhľadávať aj podľa abecedného poradia objektov. Dolné okno, nazývané *Info bar*, ukazuje stav kompilácie modelu a chyby, pokiaľ sa nejaké vyskytnú. V prípade, že sa model práve počíta, toto okno ukazuje stručné štatistiky a informácie o verzii programu. Okno vpravo, nazvané *Run Bar* združuje niekoľko samostatných okien. Tie sú v stručnosti popísané ďalej:

- *Output/Input* - používa sa v prípade, že užívateľ je vyzvaný programom, aby vložil určité dáta alebo ešte častejšie sa využíva k zobrazovaniu priebežného reportu počas výpočtu. Táto časť bude podrobne popísaná ďalej v texte.
- *Stats* - počas výpočtu zobrazuje stav riešiteľa *Xpress-Oprimizer*. Po dokončení výpočtu toto okno zobrazuje nasledujúce štatistiky:
  - V prvej časti ukazuje výsledky presolvingu, teda zmenšenia rozmeru úlohy. Vľavo šedým písmom sú rozmery pred a vpravo normálnym písmom sú rozmery úlohy po presolvingu.
  - V strednej časti sú informácie o použitom algoritme a vypočítanom výsledku úlohy. Zobrazuje sa tam napríklad

informácia o tom, či je výsledok optimálny a ako dlho trval výpočet.

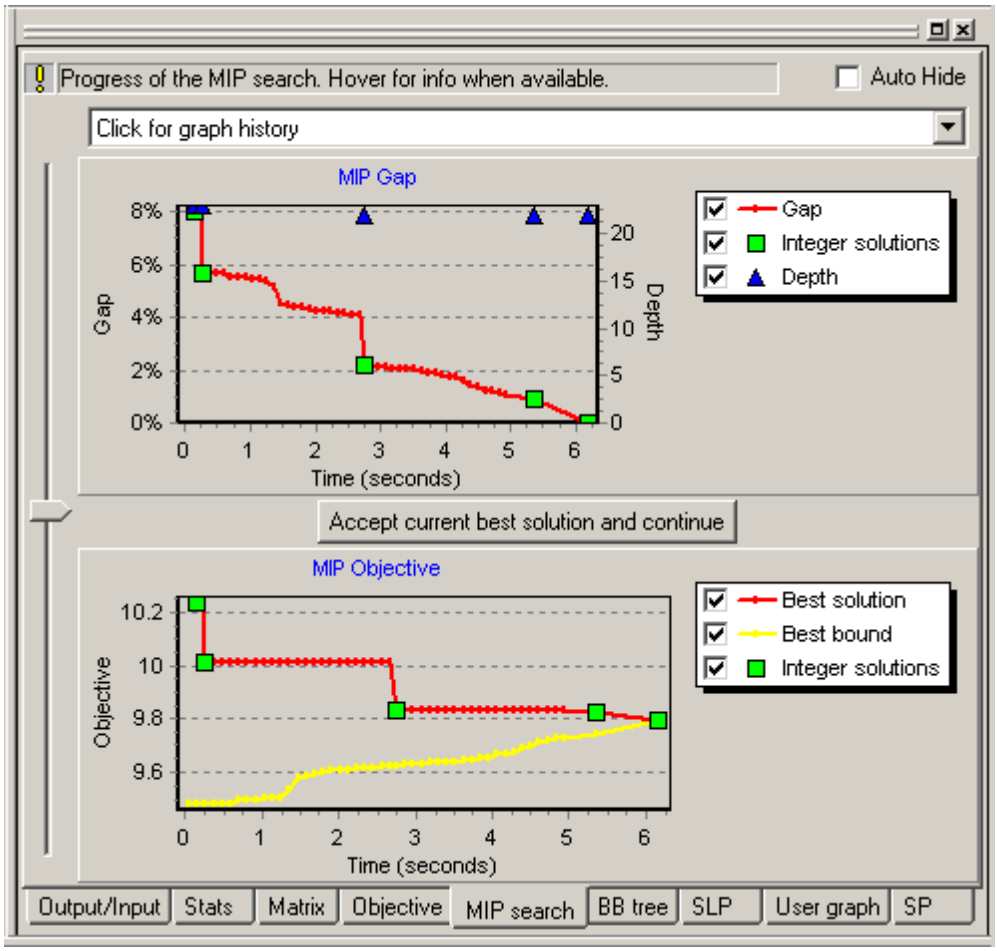
- V dolnej časti označenej *Time overhead* sú zobrazené informácie o tom, koľko času strávil Xpress-IVE na iných úkonoch ako bola samotná optimalizácia.

Obr. 3.2: Ukážka okna *Run Bar – Stats* po dokončení výpočtu - program Xpress IVE



- *Matrix* – táto časť sa snaží usporiadať premenné a obmedzenia do logického útvaru za účelom poskytnúť prehľadnú reprezentáciu matice štruktúrnych koeficientov.
- *Objective* - zobrazuje vývoj účelovej funkcie v čase. Ukážku je možné nájsť v prílohe č. 4.
- *MIP search* – táto časť obsahuje 2 grafy. Graf v hornej časti ukazuje vývoj medzery MIP (MIP Gap) počas globálneho hľadania optimálneho riešenia. Tento graf zobrazuje aj body v čase, kedy bolo nájdené celočíselné riešenie a hĺbku prehľadávacieho stromu v tomto momente. Graf v spodnej časti zobrazuje relatívnu vzdialenosť medzi najlepším nájdeným celočíselným riešením a hodnotou medze, pri ktorej bolo nájdené zatiaľ najlepšie celočíselné riešenie. Ukážka je uvedená ďalej.

Obr. 3.3: Ukážka okna *Run Bar – MIP Search* - program Xpress IVE



- *BB tree* - je grafickým zobrazením stromu metódy vetvenia a medzí. Jeho ukážka sa nachádza v prílohe č. 5.
- *SLP* - zobrazuje informácie o priebehu výpočtu riešiteľom *Xpress-SLP*.[52]

Vloženie vstupných dát je možné rovnako ako aj v programe CPLEX dvoma spôsobmi. Jeden je vytvorením nového súboru a druhým je načítanie dát už z existujúcich súborov.

Pokiaľ chceme vytvoriť nový súbor treba z menu vybrať *File* → *New*, prípadne kliknúť na ikonku . Otvorí sa okno, kde je potreba zadať názov nového súbora a jeho umiestnenie na disku. V strednom okne pracovnej plochy, ktoré bolo predtým neaktívne a vyfarbené šedou farbou, sa potom vytvorí okno, do ktorého je možné zapisovať model v jazyku Mosel. Tento jazyk v mojej diplomovej práci nebudem bližšie popisovať a záujemcov odkazujem na online publikáciu Xpress-Mosel User Guide publikovanú na web stránkach spoločnosti Dash Optimization.

[52] Xpress-IVE version 1.18.05 Help.

Ďalšou možnosťou je otvorenie už hotového modelu v podporovanom formáte. Xpress je schopný importovať tieto typy súborov:

- *MOS* – model uložený vo vlastnom formáte programu Xpress,
- *DAT*, *TXT* – dátové a textové súbory vo formáte ASCII vytvorené Xpressom alebo iným softvérom,
- *MPS* – model vo formáte MPS/MPSX,
- *MAT* – matica vo formáte MPS/MPSX,
- *LP* – matica vo formáte LP.[53]

Xpress používa celkom 19 rôznych typov výstupných súborov. Niektoré slúžia iba pre vnútornú potrebu Xpressu, vytvárajú sa pri výpočte a v prípade, že dôjde k prerušeniu výpočtu z akýchkoľvek príčin, mal by byť Xpress schopný pomocou týchto súborov pokračovať vo výpočte bez toho, aby stratil predchádzajúce výsledky. Pre nás je však najdôležitejší súbor LOG s uloženými výsledkami, ktorý sa zobrazuje v okne *Run Bar - Input/Outpu*t (viz. príloha č. 6 a 7) a taktiež je možné ho uložiť ako textový ASCII súbor (príloha č. 8). Ďalej uvádzam stručný popis jeho základných častí:[54]

- Prvá časť, v programe Express-IVE zobrazená tmavomodrým okrajom, obsahuje informáciu o aktuálnej verzii a autorských právach.
- Ďalej nasleduje časť zobrazená purpurovou farbou písma, ktorá sa používa v prípade, že užívateľ zmení prednastavené hodnoty parametrov (v uvedenej ukážke v prílohe táto funkcia nebola využitá, a tak sa tam táto časť nezobrazuje).
- Zeleným okrajom je vymedzená ďalšia časť, a to fáza načítavania úlohy a jej štatistík (*Problem Statistics)*, ktoré informujú o počte riadkov, stĺpcov a nenulových prvkov matice. Potom nasledujú informácie o globálnych štatistikách (*Global Statistics*), ktoré informujú o premenných zahrnutých do skupín ako napr. *Special Ordered Sets* a pod.
- Bledomodrý okraj zobrazuje naprv informácie o rozmeroch upravenej úlohy po presolvingu, kedy dôjde k odstráneniu nadbytočných obmedzení alebo premenných úlohy, či k pridaniu doplňujúcich obmedzení. Potom nasleduje časť zobrazujúca fázu lineárnej relaxácie úlohy, teda informácie o riešení úlohy bez podmienok celočíselnosti zložená zo stĺpcov:
    - *Its* - zobrazuje počet prebehnutých iterácií,
    - *Obj Value* - informuje o hodnote účelovej funkcie v danej iterácii a ďalší stĺpec informuje o použitej metóde (P- primárne simplexová metóda, D- duálne simplexová metóda),
    - *Ninf* - informuje o počte neprípustných riešení,
    - *Nneg* – informuje o počte premenných, ktoré by mohli zlepšiť dané riešenie v prípade, že by sa u nich zmenili alebo uvolnili obmedzujúce podmienky,
    - *Sum Inf* - je súčet všetkých neprípustných riešení,
    - *Time* - informuje o prebehnutom čase.

  Odstavec je zakončený informáciou o nájdenom optimálnom riešení.
- Oranžovým okrajom sa zobrazuje fáza globálneho hľadania optima. Začína časťou *Generating Cuts*, ktorá podáva informácie o vygenerovaných rezoch. Potom nasleduje tabuľka popisujúca priebeh vytvárania rezov. Pod ňou sú štyri

[53] BARTOŇ, 2002, str. 32.
[54] Xpress-IVE version 1.18.05 Help.

riadky, z ktorých prvé dva informujú o rezoch v matici, ktoré sa použijú pri výpočtoch vždy a o počte prvkov matice, ktoré boli aplikovanými rezmi odrezané. Tretí a štvrtý riadok informujú o počte rezov a počte prvkov, ktoré boli nimi odrezané a ktoré sú umiestnené v tzv. "cutpoolu". Sú to vlastne nadbytočné obmedzujúce podmienky, ktoré sa pri výpočte berú v úvahu iba pokiaľ bude niektorý z nich porušený. V inom prípade sa s nimi nepočíta, a tým sa urýchluje výpočet. Posledný odstavec informuje o priebehu a výsledku výpočtu celočícelného optima metódou vetvenia a medzí. Táto časť je rozdelená do viacerých stĺpcov:

- *Node* - udáva počet preskúmaných uzlov. Pokiaľ je naľavo od čísla uzlu hviezdička (*), označuje to uzol, v ktorom bolo nájdené zatiaľ najlepšie celočíselné riešenie,
- *Sols* - je údaj o počte prípustných celočíselných riešení,
- *Best Sol* - informuje o najlepšej hodnote celočíselného optima, ktoré bolo zatiaľ nájdené,
- *Best Bound* - udáva hodnotu medze, pri ktorej bolo nájdené doteraz najlepšie celočíselné riešenie,
- *Active* - je údaj o počte aktívnych uzlov vo výpočtovom strome metódy vetvenia a medzí,
- *Time* - meria čas potrebný k dosiahnutiu daného riešenia,
- *Search completed* – informuje o tom, že bolo hľadanie ukončené a za ním nasledujú informácie o počte preskúmaných uzlov , hodnote celočíselného optima a dobe trvania výpočtu,
- Poslednou časťou je tabuľka nenulových hodnôt vo vektore riešenia[55]

Pri mojom testovaní som využívala riešiteľa *Xpress-Optimizer* zo sady Xpres-MP. Ako už bolo spomenuté tento riešiteľ umožňuje riešiť úlohy lineárneho (LP), zmiešaného celočíselného (MIP), kvadratického (QP) a zmiešaného kvadratického programovania (MIQP). Je zložený z troch základných častí:

- *Simplexový riešiteľ (The Simplex Optimizer)* – riešiteľ úloh lineárneho programovania. Používa k výpočtu primárne a duálne simplexovú metódu.
- *Bariérový riešiteľ (The Barrier Optimizer)* – tento riešiteľ poskytuje alternatívu k simplexovému riešiteľovi pri výpočte úloh lineárneho (LP) a kvadratického programovania (QP).
- *Riešiteľ pre úlohy MIP (The MIP Optimizer)* – tento riešiteľ používa sofistikovaný algoritmus metódy vetvenia a medzí pre riešenie úloh zmiešaného celočíselného lineárneho (MIP) a kvadratického programovania (MIQP). Tento riešiteľ obsahuje:
  - MIP presolve algoritmus, ktorý pomáha zmenšiť rozmery úlohy, a tým aj výpočtový čas,
  - pokročilé stratégie rezných plánov, ktoré majú za účel zlepšiť kvalitu medzí a redukovať veľkosť globálneho prehľadávania. Medzi používané techniky patria napr. *Flow covers*, *GUB covers*, *Lift and Project*, *Clique cuts*, *Flow paths*, *Mixed integer rounding*, *Gomory fractional cuts*,

[55] BARTOŇ, 2002, str. 34.

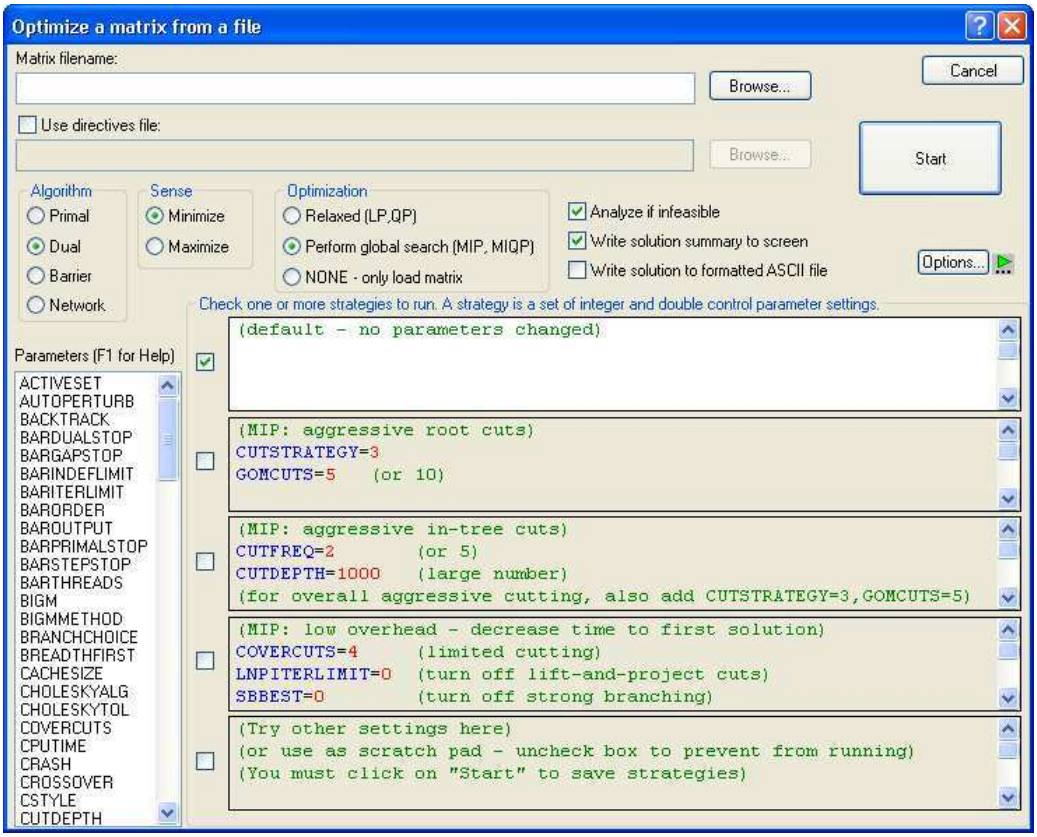
- special ordered sets,
- rôzne metódy prehľadávania uzlov ako napr. *Breadth-first*, *Best-first*, *Depth-first search*,
- užívateľom definované stratégie výberu vetviacej premennej,
- užívateľom definované priority a smer vetvenia,
- viacnásobné LP algoritmy pre riešenie úvodnej LP relaxácie a výpočtu v uzloch,
- heuristiky,
- možnosť použitia tzv. *Parralel-MIP* pre počítače s viacerými procesormi.[56]

3.3.4 Riadenie výpočtu

Základné ovládanie programu Xpress-IVE som popísala v prechádzajúcej kapitole a v tejto kapitole by som sa chcela venovať iba nastaveniam programu, ktoré priamo súvisia s riešením úloh zmiešaného celočíselného programovania.

Nastaviť parametre riešiteľa je možné pomocou menu *Build → Optimize a matrix from a file* alebo stlačením tlačítka  v paneli nástrojov. Po tomto úkone by sa malo otvoriť okno zobrazené na obrázku nižšie.

Obr. 3.4: Nastavitelné parametre riešiteľa *Xpress-Optimizer*



[56] Dostupné z <http://www.dashoptimization.com/home//products/products_optimizer.html>, cit. 17-04-2008.

V tomto okne je možné nastaviť viacero parametrov výpočtu:

- *Algorithm* - voľba úvodného algoritmu. Máme na výber primárne a duálne simplexovú metódu (*Primal, Dual*), ďalej bariérový a sieťový algoritmus (*Barrier, Network*). Prednastaveným algoritmom je duálne simplexová metóda.
- *Sense* – určuje smer optimalizácie. Na výber je buď maximalizovať (*Maximize*) alebo minimalizovať (*Minimize*) účelovú funkciu. Toto nastavenie prepíše smer optimalizácie v načitanom súbore, a preto si treba dávať pozor na výber správneho smeru.
- *Optimalization* – výber optimalizácie podľa typu riešenej úlohy. Na výber sú možnosti:
  - *Relaxed (LP, QP)* – nájde relaxované riešenie. Je určený pre úlohy LP a QP.
  - *Perform Global Search (MIP, MIQP)* – nájde celočíselné riešenie. Určený pre úlohy MIP a MIQP.
  - *NONE –only load matrix* – načíta maticu iba pre kontrolu/vizuálizáciu. Žiadna optimalizácia neprebehne.
- *Analyze if infeasible* – pokiaľ problém nemá prípustné riešenie, spustí príkaz na overenie neprípustnosti.
- *Write solution summary to screen* – vypíše nenulové prvky vektora riešenia do okna *Output* v okne *Run Bar*.
- *Write solution to formatted ASCII file* – zapíše výsledok do súboru vo formáte *PRT* a potom ho otvorí.
- *Use directive files* – existuje možnosť načítania súboru vo formáte *DIR*, obsahujúcom riadiace príkazy a hodnoty parametrov pre výpočet úlohy.
- *Option* – vyvolá okno s ponukou, kde je možné si nastaviť tieto parametre:
  - *Draw progress graph* – nakreslí vývojový graf pre zvolený algoritmus. V ponuke sú algoritmy – *Simplex, Newton Barrier a Global search.*
  - *Branch and bound tree* – na výber sú možnosti:
    - *Draw tree* – nakreslí prehľadávací strom metódy vetvenia a medzí.
    - *Enable node highlighting*- umožní prehľadávanie stromu podľa názvu premennej.
  - *Pause* – pozastaví výpočet a pri každom zápise do LOG súboru (v prípade algoritmov *simplex* a *global search*), prípadne po každej iterácii (v prípade *Newton barrier* algoritmu).
  - *Matrix Visualization* – je možné si zvoliť kontrolu pôvodnej načítanej matice (*Show original matrix)* alebo matice po procese presolvingu (*Show presolved matrix)* a ich vizualizáciu.

  Toto menu je možné vyvolať aj v paneli nástrojov kedykoľvek v priebehu výpočtu, a to kliknutím na ikonku  . Nastavené zmeny sa okamžite premietnu do výpočtu. Grafická ukážka tohto menu sa nachádza v prílohe č. 9.
- *Parameters* – zobrazuje ponuku nastavitelných parametrov. Pokiaľ si nejaký parameter označíme kurzorom a stlačíme klávesu F1, zobrazí sa nám

nápoveda, v ktorej sa dozvieme, čo daný parameter ovplyvňuje a aké sú možnosti jeho nastavenia. Pokiaľ chceme zmeniť hodnotu niektorého z parametov, musíme ho pretiahnuť do vedlajšieho okna a zadať jeho požadovanú hodnotu. Tu by som chcela poukázať na niektoré parametre, ktoré využíva metóda vetvenia a medzí pri počítaní úloh MIP:

- *NODESELECTION* – definuje skupinu kandidátov pre výber uzlu na preskúmanie. Môže nadobúdať hodnoty
  - 1 – vyberá dva descendentné uzly a je použitá ako prednastavená hodnota,
  - 2 – je použitá metóda *breadth fist search*,
  - 3 – je použitá metóda *depth first search*,
  - 4 – vyberá podľa priority zvolenej metódou *breath fist search* a potom sa vracia k výberu podľa pôvodnej prednastavenej hodnoty.
- *MIPPRESOLVE* – pokiaľ má tento parameter nenulovú hodnotu, potom bude proces presolvingu aplikovaný v každom uzle prehľadávacieho stromu metódy vetvenia a medzí. Je možnosť nastavenia hodnôt:
  - 1 – pre techniku *reduced cost fixing*,
  - 2 – pre techniku *variable fixing*,
  - 4 – pre techniku *probing at root node*,
  - Prípadne ich kombinácie dané súčtom ich hodnôt (napr. $1+2=3$, teda bude použitá technika *reduced cost fixing + variable fixing*).
- *CUTSTRATEGY* – tento parameter špecifikuje stratégiu zaraďovania rezných plánov. Čím agresívnejšia stratégia, tým viac bude vytvorených rezov, a teda aj prehľadávaných uzlov, a tým bude dlhší výpočtový čas. Možnosti nastavenia sú
  - -1 – prednastavený výber stratégie,
  - 0 – neaplikujú sa žiadne rezy,
  - 1 – konzervatívna stratégia,
  - 2 – mierna stratégia,
  - 3 – agresívna stratégia .

Samotný výpočet potom spustíme kliknutím na tlačítko *Start* a v prípade, že nechceme zmenené hodnoty nastavení aplikovať, stačí kliknúť na tlačítko *Cancel*. [57]

3.3.5 Priebeh výpočtu a výpočtová procedúra

Výpočet obvykle začína procesom presolvingu, ktorý redukuje rozmery úlohy, a tým aj zkracuje výpočtový čas. Jej účelom ako už bolo spomenuté, je odstrániť nadbytočné obmedzenia alebo premenné, prípadne niektoré obmedzenia pridať alebo posilniť.

Ako ďalší postup používa softvér Xpress generovanie rezov, ktoré „odrežú“ neefektívne neceločíselné premenné. Rezov môže byť pomerne veľké množstvo, čo

[57] Xpress-IVE version 1.18.05 Help.

môže spomaliť výpočet. Softvér Xpress umožňuje zaraďovať rezy do tzv. „cutpoolu“, kde sú uložené, ale v danom momente sa s nimi nepočíta. Do pôvodnej matice úlohy sa pridávajú až v prípade, že je niektorá podmienka výpočtu porušená. Stratégiu zaraďovania rezov je možné ovplyvniť nastavením parametra *CUTSTRATEGY*.

Metóda vetvenia a medzí, ktorú program Xpress používa pre výpočet úloh zmiešaného celočíselného programovania necháva užívateľovi dostatok možností ovplyvnenia výpočtu. Dva hlavné faktory, ktoré ovplyvňujú rýchlosť a úspešnosť nájdenia optimálneho riešenia sú výber uzlu a vetviacej premennej. Vlastnosti týchto faktorov je možné určiť v nastaveniach parametrov *NODESELECTION*, pre výber uzlu a *VARSELECTION*, pre výber vetviacej premennej.[58]

Pre účely testovania a zachovania rovných podmienok pre všetky testované softvéry som nechala všetky parametre vo svojich prednastavených hodnotách. Lineárna relaxácia prebieha duálne simplexovou metódou a generovanie rezov prebieha automaticky.

3.3.6 Výsledky testovania

K testovaniu boli použité úlohy z knižnice MIPLIB 2003. Ich celkový počet je 60 a tvoria tri skupiny, ktoré boli už popísané v predchádzajúcich kapitolách.

Žiadne nastavenia parametrov softvéru Xpress som pri testovaní nemenila a všetky zostali vo svojich prednastavených hodnotách.

3.3.6.1 Prvá skupina riešených úloh

Prvú skupinu tvorí 31 úloh, ktorých výpočet by mal komerčný softvér zvládnuť do jednej hodiny. Výpočet prebiehal bez problémov, až na úlohu *noswot*, pri ktorej bola počas výpočtu vypísaná chybová hláška oznamujúca, že výpočet nemôže pokračovať z dôvodu nedostatočnej kapacity virtuálnej pamäte. Ostatné úlohy až na päť prípadov, bol schopný Xpress vypočítať a nájsť optimálne riešenie. V úlohách *cap6000*, *gesa2-o*, *fiber*, *harp* bolo vypočítané uspokojivé riešenie a jedine u úlohy *qui* sa nájdené riešenie líšilo od optimálneho o viac ako 1%, a to dokonca až o 99900%.

Iba v dvoch prípadoch, a to u úloh *fast0507* a *mas74*, trval výpočet dlhšie ako jednu hodinu.

Tabuľka s výsledkami je zobrazená na ďalej v texte.

3.3.6.2 Druhá skupina riešených úloh

Druhá skupina je tvorená 21 úlohami, ktorých optimálne riešenie je známe. Ako maximálnu dobu pre riešenie jednej úlohy som zvolila, rovnako ako pri testovaní softvéru CPLEX, 12 hodín pre možnosť vzájemného porovnania a zároveň dôvodu časového harmonogramu testovania.

[58] BARTOŇ, 2002, str. 38-39.

Pri testovaní úloh *a1c1s1*, *glass4*, *markshare1*, *markshare2*, *mkc*, *roll3000*, *swath* a *timtab* sa výpočet sám prerušil z neznámych príčin. U ostatných úloh prebiehal výpočet bez problémov.

V piatich prípadoch sa podarilo Xpressu vypočítať optimálne riešenie. Jednalo sa o úlohy *aflow40b*, *danoint*, *net12*, *roll3000* a *tr12-30*. V ďalších piatich prípadoch, úlohy *mkc*, *msc98-ip*, *rd-rplusc-21*, *seymour* a *swath* bolo nájdené uspokojivé riešenie. Všetky ostatné nájdené riešenia sa od optimálneho líšili o viac ako 1%.

Čo sa týka časovej stránky výpočtu, tak iba v jednom prípade, pri výpočte úlohy *momentum2*, sa výpočet neskončil do limitnej doby 12 hodín.

Tabuľka s výsledkami je uvedená ďalej v texte.

3.3.6.3 Tretia skupina riešených úloh

Žiadnu z úloh tretej skupiny sa nepodarilo softvéru Xpress vypočítať.

Tabuľka 3.11: Výsledky riešenia úloh softvérom Xpress – prvá skupina úloh

Názov úlohy	Výpočtový čas v sekundách	Najlepšie vypočítané riešenie	Optimálne riešenie**	Absolútny rozdiel***	Relatívny rozdiel****	Poznámky
10teams	3.6	924.0	924.0	0.0	0.0%	
aflow30a	83.6	1158	1,158.0	0.0	0.0%	
air04	50.3	56,137	56,137.0	0.0	0.0%	
air05	57	26,374	26,374.0	0.0	0.0%	
cap6000	1.1	-2.45E+06	-2.45E+06	-150.0	0.0%	
disctom	2.3	-5000	-5,000.0	0.0	0.0%	
fast0507	42,083.7	174.0	174.0	0.0	0.0%	
fiber	1.1	4.06E+05	4.06E+05	31.0	0.0%	
fixnet6	0.9	3,983.0	3,983.0	0.0	0.0%	
gesa2	0.9	2.58E+07	2.58E+07	0.0	0.0%	
gesa2-o	3.3	2.58E+07	2.58E+07	200.0	0.0%	
harp2	66.9	-7.39E+07	-7.39E+07	-500.0	0.0%	
manna81	0.7	-13,164.0	-13,164.0	0.0	0.0%	
mas74	7,000.4	11,801.2	11,801.2	0.0	0.0%	
mas76	220.6	40,005.1	40,005.1	0.0	0.0%	
misc07	101.1	2,810.0	2,810.0	0.0	0.0%	
mod011	121.9	-5.46E+07	-5.46E+07	0.0	0.0%	
modglob	0.4	2.07E+07	2.07E+07	0.0	0.0%	
mzzv11	937.7	-21,718.0	-21,718.0	0.0	0.0%	
mzzv42z	51.3	-20,540.0	-20,540.0	0.0	0.0%	
noswot	n/a	n/a	-41.0	n/a	n/a	Chyba *
nw04	45.7	16,862.0	16,862.0	0.0	0.0%	
opt1217	0.1	-16.0	-16.0	0.0	0.0%	
p2756	1.2	3,124.0	3,124.0	0.0	0.0%	
pk1	200.7	11.0	11.0	0.0	0.0%	
pp08aCUTS	1.8	7,350.0	7,350.0	0.0	0.0%	
pp08a	1.3	7,350.0	7,350.0	0.0	0.0%	
qiu	182.7	-132.9	-1.33E+05	-132,740.1	99900.0%	
rout	41.8	1,077.6	1,077.6	0.0	0.0%	
set1ch	0.4	54,537.8	54,537.8	0.0	0.0%	
vpm2	1.8	13.8	13.8	0.0	0.0%	

***** - chyba - minimálna hodnota virtuálnej pamäte je príliš malá

** - známe optimálne riešenie[59]

******* - vyjadruje absolútny rozdiel najlepšieho vypočítaného riešenia od optimálneho riešenia

******** - vyjadruje relatívny rozdiel najlepšieho vypočítaného riešenia od optimálneho riešenia

[59] Dostupné z <http://miplib.zib.de/miplib2003.php>, cit. 05-04-2008.

Tabuľka 3.12: Výsledky riešenia úloh softvérom Xpress – druhá skupina úloh

Názov úlohy	Výpočtový čas v sekundách	Najlepšie vypočítané riešenie	Optimálne riešenie	Absolútny rozdiel	Relatívny rozdiel	Poznámky
a1c1s1	6,371.3	11,770.7	11,503.4	-267.3	2.3%	Ukončené *
aflow40b	10,186.3	1,168.0	1,168.0	0.0	0.0%	
arki001	7,552.3	7.85E+06	7.58E+06	-2.70E+05	3.4%	
atlanta-ip	29,333.8	94.0	90.0	-4.0	4.3%	
danoint	15,934.3	65.7	65.7	0.0	0.0%	
glass4	8,595.9	1.60E+09	1.20E+09	-4.00E+08	25.0%	Ukončené *
markshare1	2,653.1	8.0	1.0	-7.0	87.5%	Ukončené *
markshare2	2,403.5	11.0	1.0	-10.0	90.9%	Ukončené *
mkc	11,829.9	-563.4	-563.8	-0.4	0.1%	Ukončené *
momentum1	22,500.0	1.41E+05	1.09E+05	-32,071.7	22.7%	
momentum2	Viac ako 12 hodín	13,911.1	12,314.2	-1,596.9	11.5%	Prerušenie výpočtu**
msc98-ip	18,638.8	1.98E+07	1.98E+07	3.0	0.0%	
net12	7,206.0	214.0	214.0	0.0	0.0%	
nsrand-ipx	6,119.8	5,120.0	51,200.0	46,080.0	900.0%	
protfold	36,491.3	-26.0	-31.0	-5.0	19.2%	
rd-rplusc-21	27,685.0	1.66E+05	1.65E+05	-105.6	0.1%	
roll3000	27,431.3	12,890.0	12,890.0	0.0	0.0%	Ukončené *
seymour	38,427.5	424.0	423.0	-1.0	0.2%	
swath	5,332.0	468.2	467.4	-0.8	0.2%	Ukončené *
timtab1	13,139.4	7.77E+06	7.65E+05	-7.00E+06	90.2%	Ukončené *
tr12-30	6,496.4	1.31E+05	1.31E+05	0.0	0.0%	

***** - ukončené z neznámych dôvodov

****** - výpočet neskončil do 12 hodín, nutné prerušenie výpočtu

3.3.7 Zhrnutie

Optimalizačný softvér Xpress-MP od spoločnosti Dash Optimization je výkonný a prehľadný nástroj na riešenie optimalizačných úloh. Ako jediný z testovaných sofvérov poskytuje prepracované grafické rozhranie pre operačný systém Windows, ktoré je jednoducho ovládatelné a podporené kvalitne spracovanou nápovedou. Jeho súčasťou je i programovací jazyk Mosel a ďalšie nástroje pre tvorbu vlastných modelov a aplikácií. Ďalšou výhodou softvéru Xpress je možnosť dokúpenia modulu *Xpress connect* pre podporu rozhrania *ODBC*.

Pre riadenie výpočtu a nastavenia optimalizácie poskytuje Xpress širokú škálu nastavitelných parametrov, ktoré sú podrobne popísané v užívateľskej príručke alebo v súbore nápovedy na počítači. Výhodou je aj možnosť sledovania výpočtu priamo na obrazovke, rovnako ako aj ukladanie výstupov do zvoleného súboru. Tento súbor obsahuje aj záznam o tom, v akých nastavenia daný výpočet prebiehal. Výstupy sú prehľadné s možnosťou nastavenia stupňa podrobnosti.

Pri testovaní prvej skupiny úloh z knižnice MIPLIB 2003 prebiehal výpočet okrem jednej úlohy bez problémov. V tomto jednom prípade bol výpočet prerušený z dôvodu nedostatočnej kapacity virtuálnej pamäte. Pre ostatné úlohy, až na 5 prípadov,

sa podarilo Xpressu nájsť optimálne riešenie. V 4 z týchto 5 prípadoch bolo nájdené uspokojivé riešenie a iba v jedinom prípade bolo nájdené riešenie, ktoré sa za uspokojivé nedá považovať. Čo sa týka času, tak Xpress zvládol výpočet skoro u všetkých úloh do jednej hodiny. Iba v dvoch prípadoch sa mu to nepodarilo a riešenie počítal dlhšie ako jednu hodinu.

Testovanie druhej skupiny úloh nebolo už tak plynulé. V 8 prípadoch sa výpočet sám prerušil z neznámych príčin. V 5 prípadoch sa podarilo Xpressu nájsť optimálne riešenie, v ďalších 5 prípadoch bolo nájdené uspokojivé riešenie a všetky ostatné vypočítané riešenia boli označené za neuspokojivé. Výpočet sa skončil až na jednu výnimku do limitnej doby 12 hodín, čo je prekvapivo dobré skóre.

Pri mojom testovaní som použila verziu Xpress 18.00. Nemenila som žiadne parametre a všetky hodnoty som nechala vo svojich prednastavených hodnotách, z dôvodu možnosti objektívneho porovnania s výkonom ostatných testovaných softvérov.

Systém Xpress je vhodný pre svoju prehľadnosť a jednoduchosť ovládania. Výsledky, ktoré poskytuje sú presné a v prehľadnej forme. Užívateľovi je ponechaná široká škála možností ovplyvnenia procesu výpočtu. Prednastavené algoritmy sú však nastavené tak, aby fungovali dostatočne dobre vo väčšine prípadov. Vzhľadom na dnešné štandardy pri práci s počítačmi, kedy je väčšina populácie zvyknutá pracovať v grafickom prostredí, túto výhodu Xpressu oproti ostatným dvom softvérom považujem za pomerne dôležitú. Grafické rozhranie bolo pri práci s programom veľmi nápomocné a prehľadné a tento softvér vrele doporučujem.

3.4 MOSEK 5.0.

3.4.1 Úvod

Optimalizačný softvér MOSEK je produktom dánskej spoločnosti MOSEK ApS. Táto spoločnosť bola založená v roku 1997 pánmi Erlingom a Knud D. Andersenom a špecializuje sa na tvorbu softvéru pre riešenie matematických optimalizačných úloh.[60] Medzi jej zákazníkov patria spoločnosti ako napr. BOSH, SSB Cargo AB, TechEdge, The World Bank, GAMS Corp, ORTEC atď.[61]

MOSEK je optimalizačný softvér dizajnovaný na riešenie úloh matematického programovania veľkých rozmerov. Dokáže riešiť úlohy lineárneho, kvadratického, nelineárneho a zmiešaného celočíselného programovania. Je tvorený z viacero riešiteľov:

- *Interior-point optimizer* – je riešiteľ, ktorý používa pre výpočet metódu vnútorného bodu. Rieši všetky spojité úlohy.
- *Primal* a *Dual simplex optimizer* – simplexový riešitelia sú alternatívou riešiteľa *Interior-point optimizer*, určení na riešenie úloh lineárneho programovania.
- *Mixed-integer optimizer* – riešiteľ úloh zmiešaného celočíselného programovania. Jeho základ tvorí technika vetvenia a rezov. [62]

Existuje viacero možností ako pracovať s programom MOSEK:

- Pomocou súborou vo formáte *MPS*, *LP* a *OPF*, ktoré budú detailnejšie popísané neskôr v texte.
- Pomocou APIs. MOSEK môže byť vyvolaný pomocou rôznych programovacích jazykov : C/C++, C# (plus ďalšie .NET jazyky), Delphi, Java a Python.
- Pomocou programov iných výrobcov:
  - *AMPL* – MOSEK môže byť jednoducho použitý pomocou jazyka AMPL, čo je jazyk na modelovanie, ktorý umožňuje formulovať optimalizačné problémy vo veľmi jednoduchej forme.
  - *MATLAB* – je možné použiť nástroje programu MOSEK aj v rámci MATLABu, čo je interaktívny systém pre vedecké a technické výpočty.[63]

Momentálne najaktuálnejšia verzia programu je MOSEK 5.0., ktorá bola uvedená na trh v júni minulého roku. Túto verziu som testovala v mojej diplomovej práci. Oproti starším verziám obsahuje viacero zlepšení ako napr. výrazné zlepšenie rýchlosti a stability simplexového riešiteľa, priemerný čas výpočtu úloh veľkého rozsahu riešiteľom *Primal simplex optimizer* sa zlepšil o 20 % a *Dual simplex oprimizer* o 40 %,

[60] Dostupné z <http://www.mosek.com/index.php?id=18>, cit. 12-04-2008.
[61] Dostupné z <http://www.mosek.com/index.php?id=42>, cit. 12-04-2008.
[62] Dostupné z <http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/tools/node004.html#182958137640>, cit. 12-04-2008.
[63] Dostupné z <http://artax.karlin.mff.cuni.cz/~beda/cz/matlab/primercz/matlab-primer.html#intro>, cit. 12-04-2008.

zaradenie nového simplexového riešiteľa pre sieťové úlohy, pomocou ktorého prebieha výpočet rýchlejšie ako klasickými simplexovými riešiteľmi, atď.[64]

Výhodou tohto softvéru je neobmedzený rozsah riešených úloh. Je v podstate obmedzený len dostupnou pamäťou počítača, na ktorom prebieha výpočet. MOSEK podporuje aj paralelné využívanie viecerých procesorov počítača pri výpočte úlohy riešiteľom *Interior-point optimizer*.

Spoločnosť poskytuje niekoľko druhou licencií pre program MOSEK:

- *Evaluačná licencia* – určená pre užívateľov, ktorí chcú odskúšať MOSEK. K dispozícii je plná verzia programu s časovým obmedzením 30 dní a zákazom použitia pre komerčné účely.
- *Komerčná licencia* – existujú dva druhy komerčnej licencie. Normálna (*normal*) komerčná licencia zakladá právo na použitie programu na dobu neobmedzenú. Prenajatá (*rental*) licencia je časovo obmedzená.
- *Akademická licencia* – určená pre univerzity na akademické účely. Spoločnosť poskytuje aj študentskú verziu, ktorá je zadarmo, ale s obmedzením použitia riešiteľa úloh MIP a AMPL. Táto licencia je časovo obmedzená na 90 dní s možnosťou predĺženia v prípade potreby.[65]

Podľa aktuálneho cenníka spoločnosti MOSEK, by stála komerčná licencia s plným vybavením pre riešenie úloh zmiešaného celočíselného programovania 5 750 USD, čo je v prepočte podľa aktuálneho kurzu ČNB zo dňa 27.4.2008 cca 91 500 Kč.

3.4.2 Inštalácia

Program MOSEK je možné si stiahnuť priamo z webových stránok spoločnosti MOSEK ApS. Demo licencia, určená pre evaluáciu programu, je viazaná licenčným kľúčom na daný počítač a je časovo obmedzená. Druhý typ licencie tzv. „*floating*“, je viazaný na licenčný server, čo znamená, že túto licenciu je možné používať rôznymi počítačmi, ktoré sú pripojené cez internet k licenčnému serveru a celý proces funguje na princípe tzv. „*lincense tokens*“.[66]

Tabuľka na ďalšej stránke zobrazuje počítačové platformy, ktoré sú softvérom MOSEK podporované.

[64] Dostupné z <http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/tools/node003.html>, cit. 12-04-2008.
[65] Dostupné z <http://www.mosek.com/index.php?id=51>, cit. 12-04-2008.
[66] Dostupné z < http://www.mosek.com/index.php?id=51>, cit. 12-04-2008.

Tabuľka 3.13: Podporované počítačové platformy softvérom MOSEK

Popis	Typ	Verzia OS
Linux 32 bit x86	linux32x86	RedHat 9+
Linux 64 bit x86	linux64x86	RedHat Enterprise 4+
Linux 64 bit Itanium2	linuxia64	RedHat Enterprise 4+
MAC OSX 32 bit x86	osx32x86	10.3+
MAC OSX 32 bit Power PC	osx32ppc	10.2+
Solaris 32 bit x86	solaris32x86	10+
Solaris 64 bit x86	solaris64x86	10+
Solaris 32 bit Sparc	solarissparc	9+
Solaris 64 bit Sparc	solarissparc64	9+
Windows 32 bit x86	win	2000/XP/2003/Vista
Windows 64 bit x86	win64x86	XP/2003/Vista

Inštalácia programu je pomerne jednoduchá a je doprevádzaná jednoduchými inštrukciami. Po inštalácii je potreba aktivovať licenciu. Tento proces sa líši od druhu licencie. V prípade zakúpenia licencie je potreba zaslať svoj HOSTNAME a HOSTID, ktoré sa vygenerujú voľbou *Start menu → Mosek Optimalization Tools 5.0 → Generate HOSTID*, na emailovú adresu licenčného serveru. Potom by mal užívateľ obdržat emailom svoj licenčný súbor, ktorý je potreba si uložiť na disk a voľbou *Start menu → Mosek Optimalization Tools 5.0 → Install MOSEK license server* ho aktivovať. V prípade demo verzie programu, nie je potreba generovať HOSTNAME a HOSTID, nakoľko licenčný súbor bude užívateľovi zaslaný hneď potom, čo vyplní formulár žiadosti o demo verziu. V tomto prípade, stačí ak sa licenčný súbor, pôvodne zabudovaný v programe, prepíše novým licenčným súborom.[67]

3.4.3 Možnosti softvéru MOSEK

Ako už bolo spomenuté, MOSEK je softvérový balík pre riešenie matematických optimalizačných úloh. Dokáže riešiť úlohy lineárneho, kvadratického, nelineárneho a zmiešaného celočíselného programovania. Obsahuje viacero riešiteľov, ktorí boli v krátkosti popísaní v úvodnej kapitole. Pri mojom testovaní bol najviac využívaný riešiteľ pre zmiešané celočíselné úlohy (*Mixed-integer optimizer*). Ten dokáže riešiť problémy s lineárnymi, kvadratickými a kónickými premennými. Nevie však riešiť úlohy, ktoré obsahujú súčasne kónické obmedzenia a kvadratickú účelovú funkciu alebo kvadratické obmedzenia. Tento riešiteľ používa 3 základné fázy pri výpočte úlohy:

- *Presolve* – v tejto fáze riešiteľ redukuje rozmery úlohy vypúšťaním nadbytočných premenných a obmedzení vo formulácii úlohy. Pokiaľ je to možné zosilňuje relaxáciu.
- *Heuristics* – riešiteľ používá heuristiky k odhadu dobrého prípustného riešenia. Najprv sa použije veľmi jednoduchá zaokrúhľovacia heuristika, potom pokiaľ to MOSEK považuje za užitočné použije heuristiku *„feasibility pump“* a nakoniec, pokiaľ tieto dve predchodzie heuristiky nenašli dobré prípustné riešenie, sa použijú sofistikovanejšie heuristiky.

[67] Dostupné z <http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/toolsinstall/node004.html>, cit. 13-04-2008.

- *Optimalization* – táto fáza rieši úlohu pomocou algoritmu metódy vetvenia a rezov.[68]

MOSEK sa ovláda pomocou riadkových príkazov. Po inštalácii ho spustíme príkazom *mosek* . Po zadaní príkazu *help* sa zobrazí nápoveda, ktorá obsahuje základné príkazy.[69] Na ďalšej stránke je zobrazená tabuľka týchto základných príkazov softvéru MOSEK.

Možnosti vloženia vstupných dát sú rovnaké ako u predošlých programov, a to buď vytvorením nového súboru alebo načítaním už existujúceho súboru v podporovanom formáte.

Nasledujúca tabuľka zobrazuje podporované vstupné a výstupné formáty súborov, s ktorými dokáže MOSEK pracovať.

Tabuľka 3.14: Formáty súborov podporované softvérom MOSEK

Názov formátu	Čítanie	Zápis	Typ súboru	Prípona súboru
OPF	Áno	Áno	ASCII/UTF8	.opf
MPS	Áno	Áno	ASCII	.mps
LP	Áno	Áno	ASCII	.lp
OSiL XML	Nie	Áno	ASCII/UTF8	.xml
Binary task formát	Áno	Áno	Binárny	.mbt

Formát *OPF* (Optimalization Problem Format) je alternatíva formátov *MPS* a *LP* pre zápis optimalizačných úloh. Tento formát je riadkovo orientovaný a inšpirovaný formátom *CPLEX LP*. Okrem typických informácií ako sú účelová funkcia, obmedzenia atď., môže obsahovať aj nastavenia niektorých parametrov alebo informácie o celom alebo čiastočnom riešení úlohy.[70]

Formát *MPS* je optimalizačný štandard a bol detailne popísaný v druhej kapitole. Formát *LP* je vlastný formát programu CPLEX určený k zápisu úlohy v matematickej podobe.

Formát *OSiL XML* (Optimization Services instance Language) je určený k zápisu optimalizačných úloh. Má svoje korene vo formáte *LPFML*, ktorý bol určený k zápisu lineárnych úloh. Formát *OSiL* je možné použiť pre rôzne typy optimalizačných úloh ako napr. lineárne, zmiešané celočíselné, nelineárne atď.[71] Bol vytvorený v rámci projektu COIN-OR (Computational Infrastructure for Operations Research), ktorý sa zameriava na podporu tvorby open-source softvéru.[72]

[68] Dostupné z <http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/tools/node013.html>, cit. 13-04-2008.
[69] Dostupné z <http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/tools/node017.html>, cit. 13-04-2008.
[70] Dostupné z <http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/tools/node020.html>, cit. 13-04-2008.
[71] Dostupné z <http://www.optimizationservices.org>, cit. 13-04-2008.
[72] Dostupné z <http://www.coin-or.org>, cit. 13-04-2008.

Tabuľka 3.15: Základné príkazy programu MOSEK.

Príkaz	Význam
-a	Mosek sa spustí v móde AMPL.
-AMPL	Vstupný súbor je AMPL nl súbor
-basi name	*Name*, je názov vstupného súboru s výsledkami
-baso name	*Name*, je názov výstupného súboru s výsledkami
-brni name	*Name,* je názov súboru, ktorý sa načíta a obsahuje poradie vetviacich premenných
-brno name	*Name*, je názov súboru, ktorý sa zapíše a obsahuje poradie vetviacich premenných
-d name val	Priradí hodnotu *val* parametru s názvom *name*
-dbgmem name	Zapíše informácie o odstránených chybách do súboru s názvom *name*
-f	Vypíše všetky informácie o danej licencii
-inti name	*Name*, je názov vstupného súboru s celočíselným riešením
-into name	*Name*, je názov výstupného súboru s celočíselným riešením
-itri name	*Name*, je názov vstupného súboru s výsledkami metódy vnútorného bodu
-itro name	*Name*, je názov výstupného súboru s výsledkami metódy vnútorného bodu
-info name	*Name*, je názov súboru, ktorý sa zapíše a obsahuje neprípustné podúlohy
-infrepo name	*Name*, je názov súboru, ktorý sa zapíše a obsahuje informácie o úprave prípustnosti úlohy
-pari name	*Name*, je názov vstupného súboru s parametrami. Je ekvivalentný príkazu *-p*
-paro name	*Name*, je názov výstupného súboru s parametrami
-L name	*Name*, je názov licenčného súboru
-l name	*Name*, je názov licenčného súboru
-max	Prinúti MOSEK, aby maximalizoval účelovú funkciu
-min	Prinúti MOSEK, aby minimalizoval účelovú funkciu
-n	Ignoruje chyby v ďalšom nastavení parametrov
-p name	*Name*, je názov súboru, z ktorého sa načita nové nastavenie parametrov
-q name	*Name*, je názov log súboru.
-r	Pokiaľ je táto voľba vybraná, program vráti hodnotu -1 v prípade, že sa vyskytne chyba, inak vráti hodnotu 0
-rout name	Pokiaľ je táto voľba vybraná, program zapíše kód výsledku do súboru s názvom *name*
-sen file	Uskutoční analýzu citlivosti
-silent	Minimum informácií je poslaných do terminálu
-v	Je vypísané číslo verzie programu MOSEK
-w	Pokiaľ je táto voľba zahrnutá, bude MOSEK čakať na licenciu
-=	Vypíše databázu parametrov
-?	Vypíše nápovedu

MOSEK generuje jeden až dva súbory s výsledkami, v závislosti od typu úlohy a použitom riešiteľovi. V prípade použitia riešiteľa *Interior-point Optimizer* sa generuje súbor s výsledkami s príponou *SOL* . Pokiaľ sa použije simplexový riešiteľ, generuje sa súbor s výsledkami s príponou *BAS* a v prípade riešiteľa úloh MIP, súbor s príponou *INT*. Tieto súbory obsahujú základné informácie o riešení ako je názov riešenej úlohy, stav výpočtu, prípustnosť riešenia, hodnota účelovej funkcie, názvy obmedzení a ich stav, názvy premenných a ich stav. [73]

Špeciálnym formátom je formát *ORD*, ktorý špecifikuje poradie premenných, ktoré používa riešiteľ úloh MIP pri procese vetvenia.

3.4.4 Riadenie výpočtu

Softvér MOSEK umožňuje nastavenie širokej skály parametrov užívateľom, týkajúce sa optimalizácie a jej priebehu. V ďalšom texte by som chcela popísať tie z nich, ktoré sa týkajú optimalizácie úloh zmiešaného celočíselného programovania.

Ako už bolo uvedené v predošlej kapitole, hodnoty parametrov sa menia pomocou príkazu *-d name val* , ktorý priradí hodnotu *val* parametru s názvom *name*.

Ďalej nasleduje prehľad základných parametrov a ich možných hodnôt, ktoré používa riešiteľ úloh zmiešaného celočíselného programovania, ktorý bol využívaný pri riešení testovacích úloh knižnice MIPLIB 2003:

- `MSK_IPAR_LOG_MIO` - Kontroluje množstvo zapisovaných informácií do LOG súboru. Čím má vyššiu hodnotu, tým súbor obsahuje viacej informácií. Predvolená hodnota je 4 s možnosťou nastavenia v rozmedzí od 0 až po $+\infty$.

- `MSK_IPAR_LOG_MIO_FREQ` - Kontroluje frekvenciu v akej riešiteľ úloh MIP zapíše riadok do LOG súboru. Predvolené je číslo 250 s možnosťou použitia akéhokoľvek celého čísla.

- `MSK_IPAR_MIO_BRANCH_DIR` - Kontroluje, ktorý smer pri procese vetvenia používa riešiteľ úloh MIP. Možnosti sú:
  - `1.DIR_DOWN` - riešiteľ vždy vyberá ako prvú spodnú vetvu,
  - `2.DIR_UP` - riešiteľ vždy vyberá ako prvú vrchnú vetvu,
  - `3.DIR_FREE` - riešiteľ sám vyberie smer, je to prednastavená hodnota.

- `MSK_IPAR_MIO_BRANCH_PRIORITIES_USE` - Kontroluje, či riešiteľ úloh MIP používa prioritné radenie premenných pri procese vetvenia. Možnosti sú: `OFF` – neradí sa podľa priorít a `ON` – radí sa podľa priorít, čo je aj prednastavená hodnota.

[73] Dostupné z <http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/tools/node022.html>, cit. 13-04-2008.

- MSK_IPAR_MIO_CUT_LEVEL_ROOT – Kontroluje úroveň rezov v koreňovom uzle. Prednastavená je hodnota -1, ktorá indikuje automatické rozhodovanie riešiteľa úloh MIP. Použité rezy môžu byť kontrolované nastavením nasledujúcich hodnôt: *GUI cover* (+2), *Flow cover* (+4), *Lifting* (+8), *Plant location* (+16), *Dissagregation* (+32), *Knapsack cover* (+64), *Lattice* (+128), *Gomory* (+256), *Coefficient reduction* (+512), *GCD* (+1024), *Obj. Integrality* (+2048).

- MSK_IPAR_MIO_CUT_LEVEL_TREE – Kontroluje úroveň rezov v celom strome. Možnosti nastavenia a prednastavená hodnota sú zhodné s parametrom MSK_IPAR_MIO_CUT_LEVEL_ROOT.

- MSK_IPAR_MIO_HEURISTIC_LEVEL – Kontroluje heuristiku použitú riešiteľom úloh MIP pri hľadaní úvodného prípustného riešenia. Hodnota 0 znamená, že heuristika sa vôbec nepoužije. Hodnota väčšia ako 0 znamená, že postupne čím vyššie číslo, tak tým sofistikovanejšia heuristika sa použije, avšak na úkor predĺženia výpočtového času. Predvolená hodnota -1 implikuje, že riešiteľ sám rozhodne o použitej heuristike.

- MSK_DPAR_MIO_MAX_TIME – Tento parameter určuje maximálnu dobu pre výpočet riešenia pomocou riešiteľa úloh MIP. Možné hodnoty sú od $-\infty$ do $+\infty$, kde záporné číslo znamená nekonečne dlhú dobu. Prednastavená hodnota je -1.

- MSK_IPAR_MIO_MAX_NUM_BRANCHES – Kontroluje maximálny počet vytvorených vetví počas procesu vetvenia metódou vetvenia a medzí. Možné hodnoty sú od $-\infty$ do $+\infty$, kde záporné číslo znamená nekonečne mnoho vetví. Prednastavená hodnota je -1.

- MSK_IPAR_MIO_NODE_SELECTION – Kontroluje stratégiu výberu uzlu riešiteľom úloh MIP. Možnosti nastavenia sú následovné:
  1. SELECTION_PSEUDO – výber uzlov prebieha na základe ohodnotenia pseudonákladov,
  2. SELECTION_HYBRID – použije sa hybridná stratégia výberu,
  3. SELECTION_FREE – je prednastavená voľba a znamená, že riešiteľ sám určí, ktorú stratégiu použije,
  4. ELECTION_WORST – riešiteľ použije stratégiu nejhoršej medze (worst bound strategy),
  5. SELECTION_BEST – riešiteľ použije stratégiu najlepšej medze (best bound strategy),
  6. SELECTION_FIRST – riešiteľ použije stratégiu *depth first search*.

- MSK_DPAR_MIO_REL_ADD_CUT_LIMITED – Kontroluje maximálne koľko rezov môže riešiteľ pridať do úlohy. Majme α, ktorá vyjadruje hodnotu tohto parametru a η počet obmedzení úlohy. Potom riešiteľ úloh MIP môže celkovo pridať maximálne $\alpha*\eta$ rezov.

- `MSK_IPAR_MIO_PRESOLVE_USE` – Kontroluje, či riešiteľ úloh MIP používa presolvingu. Možnosti sú `OFF` – presolving sa nepoužíva a `ON` – presolving je použitý, čo je aj prednastavená hodnota. [74]

Všetky zmeny v parametroch sa automaticky zapisujú do log súboru, čiže je možné spätne zistiť pri akých nastaveniach parametrov bola daná úloha riešená.

3.4.5 Priebeh výpočtu a výpočtová procedúra

Výpočet zmiešaných celočíselných úloh v programe MOSEK prebieha v troch základných fázach.

Ako prvá, je fáza presolvingu, ktorá redukuje rozmery úlohy, čím skracuje výpočtový čas. Odstranujú sa prebytočné obmedzenia a premenné a pokiaľ je to možné, posiluje sa relaxácia úlohy.

Druhou fázou, je použitie heuristík s účelom nájsť dobrého prípustného riešenia za čo najkratší čas. Program MOSEK používa k tomuto účelu niekoľko druhou heuristík ako napr. jednoduchú zaokrúhľovaciu heuristiku, heuristiku *feasibility pump* alebo iné sofistikované heuristiky. To aká heuristika bude použitá si automaticky zvolí riešiteľ. Môže si ju však zvoliť aj užívateľ sám pomocou nastavenia parametru `MSK_IPAR_MIO_HEURISTIC_LEVEL.`

Poslednou fázou je riešenie úlohy pomocou algoritmu metódy vetvenia a rezov. Tu sa ponúka široká škála možností k nastaveniu jednotlivých parametrou týkajúcich sa tohto výpočtu užívateľom. Týmto nastavením môže užívateľ podstatne ovplyvňovať výpočtový čas úlohy.

Ako už vieme výpočet celočíselného riešenia môže trvať aj veľmi dlhú dobu, nakoľko úlohy MIP patria do skupiny NP-ťažkých. Pokiaľ chceme dostať výsledok v rozumnom čase, existujú spôsoby, ako výpočet urýchliť. Medzi ne patria:

- Zadanie dobrého úvodného prípustného riešenia. V mnohých prípadoch sa stáva, že dobré prípustné riešenie je buď známe alebo jednoducho vypočítateľné pomocou odborných znalostí. Pokiaľ takéto riešenie poznáme, oplatí sa ho zadať riešiteľovi úloh MIP ako štart pre výpočet danej úlohy. Tento úkon môže podstatne skrátiť výpočtový čas.
- Zlepšenie formulácie úlohy. Určitá formulácia úlohy nemusí mať riešenie alebo riešenie môže trvať príliš dlhú dobu, kdežto stačí úlohu preformulovať a riešenie môžeme nájsť v kratšom čase.
- Zadanie ukončovacieho kritéria pomocou nastavitelného parametra.[75]

[74] Dostupné z <http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/tools/node024.html#182960593160>, cit. 14-04-2008.

[75] Dostupné z <http://www.mosek.com/fileadmin/products/5_0/tools/doc/html/tools/node013.html#182959166168>, cit. 14-04-2008.

3.4.6 Výsledky testovania

K testovaniu boli rovnako ako v predchádzajúcich prípadoch použité úlohy z knižnice MIPLIB 2003, ktoré boli rovnako ako predtým rozdelené do 3 skupín.

Pri testovaní som žiadne parametre softvéru MOSEK nemenila a všetky zostali vo svojich prednastavených hodnotách.

3.4.6.1 Prvá skupina riešených úloh

Testované boli úlohy, ktoré by mal byť komerčný softvér schopný vyriešiť do jednej hodiny. Celkovo je týchto úloh 31. Výpočet prebiehal takmer bez problémov, až na úlohy *aflow30a* a *fast0507*, ktoré MOSEK nevedel načítať a vypísal chybovú hlášku. U ďalších 21 úloh bolo nájdené optimálne riešenie. Zvyšných 8 úloh bolo vypočítaných, a to 5 s uspokojivým výsledkom (riešenie sa nelíšilo od optimálneho riešenia o viac ako 1%) a ostatné 3 s výsledkom neuspokojivým.

Z časového hľadiska boli všetky úlohy okrem úloh *harp2*, *mas74*, *noswot*, *qui* a *rout* vypočítané do jednej hodiny. Tabuľka s výsledkami sa nachádza na ďalšej stránke.

3.4.6.2 Druhá skupina riešených úloh

Druhú skupinu tvoria úlohy so známym optimálnym riešením. Ako maximálnu dobu pre výpočet jednej úlohy som zvolila taktiež 12 hodín ako v predchádzajúcich prípadoch.

Softvér MOSEK dopadol pri testovaní tejto skupiny pomerne zle. Dve úlohy *aflow40b* a *protfold* nedokázal vôbec načítať a v žiadnom prípade MOSEK neskončil výpočet sám do limitnej doby 12 hodín.

U úloh *timtab1* a *tr12-30* by sa dalo povedať, že MOSEK našiel optimálne riešenie, nakoľko absolútny aj relatívny rozdiel vypočítaného a optimálneho riešenia je rovný nule, avšak výpočet nebol sám ukončený do limitnej doby 12 hodín. V 4 ďalších prípadoch sa mu podarilo vypočítať uspokojivé riešenie a v ďalších prípadoch sa vypočítané riešenie líšilo o viac ako 1 % od optimálneho riešenia. Tabuľka s výsledkami je uvedená ďalej.

3.4.6.3 Tretia skupina riešených úloh

Ani jeden prípad z tretej skupiny úloh knižnice MIPLIB 2003 nebol schopný MOSEK rovnako ako aj ostatné testované softvéry vypočítať.

Tabuľka 3.16: Výsledky riešenia úloh softvérom MOSEK – prvá skupina úloh

Názov úlohy	Výpočtový čas v sekundách	Najlepšie vypočítané riešenie	Optimálne riešenie**	Absolútny rozdiel***	Relatívny rozdiel****	Poznámky
10teams	96.1	924.0	924.0	0	0.0%	
aflow30a	n/a	n/a	1,158.0	n/a	n/a	Chyba*
air04	131.7	56,137.0	56,137.0	0	0.0%	
air05	157.7	26,374.0	26,374.0	0	0.0%	
cap6000	5.5	-2.45E+06	-2.45E+06	-3	0.0%	
disctom	5.9	-5,000.0	-5,000.0	0	0.0%	
fast0507	n/a	n/a	174.0	n/a	n/a	Chyba*
fiber	5.4	4.06E+05	4.06E+05	0	0.0%	
fixnet6	23.1	3,983.0	3,983.0	0	0.0%	
gesa2	7.9	2.58E+07	2.58E+07	44	0.0%	
gesa2-o	22.8	2.58E+07	2.58E+07	44	0.0%	
harp2	5,656.4	-7.39E+07	-7.39E+07	-3	0.0%	
manna81	7.8	-13,164.0	-13,164.0	0	0.0%	
mas74	23,631.4	11,801.2	11,801.2	0	0.0%	
mas76	735.9	4.00E+41	40,005.1	-4.00E+41	100.0%	
misc07	659.0	2,810.0	2,810.0	0	0.0%	
mod011	414.5	5.46E+07	-5.46E+07	-1.09E+08	200.0%	
modglob	6.2	2.07E+07	2.07E+07	-8	0.0%	
mzzv11	964.0	-21,718.0	-21,718.0	0	0.0%	
mzzv42z	944.4	-20,540.0	-20,540.0	0	0.0%	
noswot	38,080.4	-41.0	-41.0	0	0.0%	
nw04	24.0	16,862.0	16,862.0	0	0.0%	
opt1217	10.6	-16.0	-16.0	0	0.0%	
p2756	3.0	3,124.0	3,124.0	0	0.0%	
pk1	699.0	11.0	11.0	0	0.0%	
pp08aCUTS	26.3	7,350.0	7,350.0	0	0.0%	
pp08a	18.0	7,350.0	7,350.0	0	0.0%	
qiu	9,162.1	-132.9	-1.33E+05	-1.33E+05	99899.9%	
rout	6,991.4	1,077.6	1,077.6	0	0.0%	
set1ch	3.7	54,537.8	54,537.8	0	0.0%	
vpm2	9.1	13.8	13.8	0	0.0%	

***** - chyba pri načítaní súboru

** - známe optimálne riešenie[76]

******* - vyjadruje absolútny rozdiel najlepšieho vypočítaného riešenia od optimálneho riešenia

******** - vyjadruje relatívny rozdiel najlepšieho vypočítaného riešenia od optimálneho riešenia

[76] Dostupné z <http://miplib.zib.de/miplib2003.php>, cit. 05-04-2008.

Tabuľka 3.17: Výsledky riešenia úloh softvérom MOSEK – druhá skupina úloh

Názov úlohy	Výpočtový čas v sekundách	Najlepšie vypočítané riešenie	Optimálne riešenie	Absolútny rozdiel	Relatívny rozdiel	Poznámky
a1c1s1	Viac ako 12 hodín	11,857.1	11,503.4	-353.7	3.0%	Ukončené *
aflow40b	n/a	n/a	1,168.0	n/a	n/a	Chyba**
arki001	Viac ako 12 hodín	7.58E+06	7.58E+06	-266.3	0.0%	Ukončené *
atlanta-ip	Viac ako 12 hodín	81.3	90.0	8.7	10.7%	Ukončené *
danoint	Viac ako 12 hodín	63.9	65.7	1.7	2.7%	Ukončené *
glass4	Viac ako 12 hodín	8.00E+08	1.20E+09	4.00E+08	50.0%	Ukončené *
markshare1	Viac ako 12 hodín	9.09E+13	1.0	-9.09E+13	100.0%	Ukončené *
markshare2	Viac ako 12 hodín	0.0	1.0	1.0	100.0%	Ukončené *
mkc	Viac ako 12 hodín	-575.3	-563.8	11.4	2.0%	Ukončené *
momentum1	Viac ako 12 hodín	96,291.1	1.09E+05	12,851.9	13.3%	Ukončené *
momentum2	Viac ako 12 hodín	10,697.8	12,314.2	1,616.4	15.1%	Ukončené *
msc98-ip	Viac ako 12 hodín	1.97E+07	1.98E+07	136,623.0	0.7%	Ukončené *
net12	Viac ako 12 hodín	117.0	214.0	97.0	82.9%	Ukončené *
nsrand-ipx	Viac ako 12 hodín	50,560.0	51,200.0	640.0	1.3%	Ukončené *
protfold	n/a	n/a	-31.0	n/a	n/a	Chyba**
rd-rplusc-21	Viac ako 12 hodín	100.0	1.65E+05	1.65E+05	165295.0 %	Ukončené *
roll3000	Viac ako 12 hodín	13,202.0	12,890.0	-312.0	2.4%	Ukončené *
seymour	Viac ako 12 hodín	427.0	423.0	-4.0	0.9%	Ukončené *
swath	Viac ako 12 hodín	468.5	467.4	-1.1	0.2%	Ukončené *
timtab1	Viac ako 12 hodín	7.65E+05	7.65E+05	0.0	0.0%	Ukončené *
tr12-30	Viac ako 12 hodín	1.31E+05	1.31E+05	0.0	0.0%	Ukončené *

***** - výpočet neskončil do 12 hodín, nutné prerušenie výpočtu
****** - chyba pri načítaní súboru

3.4.7 Zhrnutie

Optimalizačný softvér MOSEK od dánskej spoločnosti MOSEK ApS je výkonný nástroj na riešenie optimalizačných úloh rôzneho charakteru. Jeho výhoda je jednoduchá obsluha, široká škála užívateľom nastaviteľných parametrov a výborne spracovaná online dokumentácia. Tá je na rozdiel od iných dokumentácií voľne prístupná cez webové stránky spoločnosti a prehľadne popisuje všetky možnosti softvéru MOSEK.

Výhodou je aj neobmedzenosť rozmerov riešenej úlohy. Tá je obmedzená iba dostupnou pamäťou počítača, na ktorom výpočet prebieha.

MOSEK poskytuje prehľadné výstupy s možnosťou nastavenia ich podrobnosti a uloženia do zvoleného súboru. Neobvyklé je, že každý riešiteľ vytvára výstupné súbory s riešením vo vlastnom formáte. Formát *SOL* používaný riešiteľom *Interior-point Optimizer*, formát *BAS* pre simplexové riešitele a formát *INT* pre riešiteľa úloh zmiešaného celočíselného programovania. Výhodou týchto súborou je, že obsahujú informácie o nastavených parametroch riešenej úlohy.

Pri mojom testovaní som použila verziu MOSEK 5.0. a všetky nastavenia parametrov som nechala vo svojich prednastavených hodnotách. Tak je možné objektívne porovnať výkony v riešení úloh zmiešaného celočíselného programovania medzi jednotlivými optimalizačnými softvérmi navzájom.

Samotné testovanie v prípade prvej skupiny úloh knižnice MIPLIB 2003 prebiehalo až na dve úlohy bez problémov. Tieto dve úlohy nedokázal MOSEK načítať, a to ani pri opakovanom pokuse. V 21 prípadoch bolo nájdené optimálne riešenie, v 5 prípadoch uspokojivé riešenie a v 3 prípadoch vypočítal MOSEK riešenie, ktoré za uspokojivé nepovažujem. Okrem 4 úloh zvládol MOSEK všetky ostatné výpočty dokončiť do jednej hodiny.

Z druhej skupiny úloh nevedel MOSEK vôbec prečítať dve úlohy a v žiadnom prípade neukončil výpočet pred limitným časom 12 hodín. V 2 prípadoch našiel aj napriek neukončenému výpočtu optimálne riešenie, v ďalších 4 prípadoch riešenie uspokojivé a v ostatných prípadoch bolo výsledné riešenie rozdielne od optimálneho o viac ako 1%.

Softvér MOSEK nedosiahol teda pri testovaní veľmi dobré skóre. Ukázal sa ako pomerne pomalý a nepresný. Je to samozrejme možno iba otázka prednastavených parametrov jednotlivých výpočtových algoritmov, ale ukazuje to na fakt, že v tejto oblasti je ešte čo zlepšovať.

4. POROVNÁVANIE SOFTVÉROV

Ako už bolo spomenuté, pre testovanie boli vybraté softvéry CPLEX, Xpress a MOSEK. Použila som verzie, ktoré boli v dobe začiatku môjho testovania aktuálne a dostupné. Všetky softvéry boli testované na jednom počítači, rovnakej sade úloh a bez zmeny akýchkoľvek parametrov a nastavení, čím som sa snažila zaistiť čo najrovnejšie podmienky pre testovanie a maximálnu objektivitu výsledkov.

Ako testovacie úlohy boli vybrané úlohy z knižnice MIPLIB 2003, ktoré sú považované za štandard pri testovaní optimalizačných softvérov. Ich celkový počet je 60 a delia sa na 3 skupiny. Prvú skupinu tvoria úlohy, ktoré by mal byť schopný komerčný softvér vypočítať do jednej hodiny. Tieto úlohy, nakoľko nie sú príliš veľkých rozmerov, by mali byť testované softéry schopné vypočítať za relatívne rozumnú dobu bez väčších odchyliek od optimálneho riešenia. Pri porovnávaní budem teda túto skupinu úloh porovnávať hlavne z hľadiska doby výpočtu, nakoľko presnosť výsledkov je u všetkých porovnávaných softvérov dostatočne dobrá. Samozrejme najlepší softvér bude ten, ktorý danú úlohu vypočíta čo najrýchlejšie.

Maximálny časový limit pre výpočet jednej úlohy som určila na 12 hodín, a to z dôvodu časového harmonogramu testovania. Táto maximálna doba výpočtu jednej úlohy je samozrejme pre všetky testované softvéry rovnaká.

Druhou skupinou sú úlohy, ktorých optimálne riešenie je známe, a teda komerčný softvér by mal byť schopný ich vypočítať. Jedná sa o úlohy komplikovanejšie, väčších rozmerov, ktorých výpočet bude pravdepodobne trvať dlhšiu dobu. Pri ich výpočte môže hrať dôležitú úlohu nastavenie jednotlivých parametrov. Nakoľko som všetky parametre nechávala vo svojich prednastavených hodnotách, prejaví sa nastavenie, ktoré zvolili autori testovaných softvérov ako štandardné. V mnohých prípadoch nastal problém, že výpočet neskončil do časového limitu 12 hodín, ktorý som určila ako maximálny. V takých prípadoch som uvažovala dovtedy najlepšie nájdené riešenie. To je potom porovnávané s optimálnym riešením úlohy, čím dostanem vzdialenosť vypočítaného riešenia a optimálneho riešenia. Druhú skupinu úloh budem teda porovnávať z hľadiska najlepšieho vypočítaného času a z hľadiska najmenšej vzdialenosti od optimálneho riešenia.

Posledou skupinou sú úlohy doteraz nevyriešené žiadnym optimalizačným softvérom.Žiadny z testovaných softvérov nevyriešil ani jednu úlohu z tejto skupiny, a teda ju pri porovnávaní softvérov nebudem brať do úvahy.

4.1 Prvá skupina testovaných úloh

Ako už bolo spomenuté predtým, budem sa v tejto skupine zameriavať hlavne na výpočtový čas riešenia, nakoľko sa jedná o jednoduchšie typy úloh, ktoré by mal byť komerčný softvér schopný vyriešiť do jednej hodiny.

Na ďalšej stránke nasleduje tabuľka s porovnaním výpočtových časov jednotlivých softvérov.

Tabuľka 4.1: Výpočtové časy v sekundách testovaných softvérov – prvá skupina úloh

Názov úlohy	CPLEX	Xpress	MOSEK
10teams	21.4	3.6	96.1
aflow30a	Chyba*	83.6	Chyba*
air04	31.0	50.3	131.7
air05	23.2	57	157.7
cap6000	0.9	1.1	5.5
disctom	Chyba*	2.3	5.9
fast0507	3,136.8	42,083.7	Chyba*
fiber	0.4	1.1	5.4
fixnet6	1.0	0.9	23.1
gesa2	0.5	0.9	7.9
gesa2-o	4.1	3.3	22.8
harp2	2,311.8	66.9	5,656.4
manna81	5,477.0	0.7	7.8
mas74	1,088.2	7,000.4	23,631.4
mas76	123.2	220.6	735.9
misc07	12.9	101.1	659.0
mod011	128.2	121.9	414.5
modglob	0.2	0.4	6.2
mzzv11	640.2	937.7	964.0
mzzv42z	130.1	51.3	944.4
noswot	2,790.1	Chyba – pamäť**	38,080.4
nw04	56.4	45.7	24.0
opt1217	1,119.2	0.1	10.6
p2756	0.5	1.2	3.0
pk1	83.3	200.7	699.0
pp08aCUTS	3.0	1.8	26.3
pp08a	0.9	1.3	18.0
qiu	58.4	182.7	9,162.1
rout	32.8	41.8	6,991.4
set1ch	0.7	0.4	3.7
vpm2	1.3	1.8	9.1

***** - chyba pri načítaní súboru

****** - chyba - minimálna hodnota virtuálnej pamäte je príliš malá

Samotné porovnávanie je koncipované tým spôsobom, že v každej úlohe určím poradové číslo, podľa toho, ktorý softvér vypočítal úlohu najskôr. Číslo 1 znamená, že danému softvéru trvalo najkratšiu dobu vypočítať riešenie. V prípade chybových hlášok bude softvéru automacky pridelené číslo 3. V prípade, že doba výpočtu úlohy bude rovnaká pre dva a viac softvérov, bude im priradené poradové číslo, spočítané aritmetickým priemerom ich poradia. Pre ukážku uvádzam príklad: pokiaľ majú dva softvéry rovnaký výsledok a tento výsledok je lepší ako výsledok tretieho softvéru, prvým dvom softvérom bude potom priradené číslo 1,5 a tretiemu softvéru číslo 3.

Ďalej nasleduje tabuľka vyhodnotenia výpočtovej doby riešenia jednotlivých testovaných softvérov.

Tabuľka 4.2: Tabuľka porovnania testovaných softvérov z hľadiska výpočtovej doby jednotlivých úloh prvej skupiny

Názov úlohy	CPLEX	Xpress	MOSEK
10teams	2	1	3
aflow30a	3	1	3
air04	1	2	3
air05	1	2	3
cap6000	1	2	3
disctom	3	1	2
fast0507	1	2	3
fiber	1	2	3
fixnet6	2	1	3
gesa2	1	2	3
gesa2-o	2	1	3
harp2	2	1	3
manna81	3	1	2
mas74	1	2	3
mas76	1	2	3
misc07	1	2	3
mod011	2	1	3
modglob	1	2	3
mzzv11	1	2	3
mzzv42z	2	1	3
noswot	1	3	2
nw04	3	2	1
opt1217	3	1	2
p2756	1	2	3
pk1	1	2	3
pp08aCUTS	2	1	3
pp08a	1	2	3
qiu	1	2	3
rout	1	2	3
set1ch	2	1	3
vpm2	1	2	3
Total	**49**	**51**	**87**

Ako vidíme podľa výsledkov, z hľadiska výpočtovej doby v prvej skupine úloh dopadol najlepšie softvér CPLEX, ktorý bol najrýchlejší v riešení úloh v 18 prípadoch. Sofvér Xpress nasleduje hneď za ním. Dopadol s oniečo horším skóre, avšak rýchlosťou výpočtovej doby úloh z prvej skupiny je so softvérom CPLEX porovnateľný. V porovnávaní dopadol najhoršie softvér MOSEK, ktorý bol až v 26 prípadoch z 31 najpomalší.

Dalo by sa teda povedať, že z hľadiska rýchlosti výpočtu je pre jednoduchšie úlohy najvhodnejší softvér CPLEX a najmenej vhodný softvér MOSEK. Samozrejme si treba uvedomiť, že softvéry sú testované pri svojich pôvodných nastaveniach a že v prípade úprav jednotlivých parametrov by mohol byť výsledok úplne iný.

4.2 Druhá skupina testovaných úloh

Druhá skupina úloh knižnice MIPLIB 2003 je tvorená komplikovanejšími úlohami väčšieho rozmeru so známym optimálnym riešením. Výpočtový čas jednotlivých úloh bude pravdepodobne dlhší ako v prípade úloh z prvej skupiny. To je aj dôvod, prečo výsledky riešení tejto skupiny úloh budem porovnávať z dvoch hľadísk.

4.2.1 Výpočtový čas riešenia

Prvým hľadiskom je rovnako ako v prípade prvej skupiny výpočtový čas riešenia jednotlivých úloh. Ďalej nasleduje tabuľka s porovnaním výpočtových časov jednotlivých softvérov.

Tabuľka 4.3: Výpočtové časy v sekundách testovaných softvérov úloh druhej skupiny

Názov úlohy	CPLEX	Xpress	MOSEK
a1c1s1	Viac ako 12 hodín	6,371.3	Viac ako 12 hodín
aflow40b	34,057.4	10,186.3	Chyba pri načítaní súboru
arki001	695.0	7,552.3	Viac ako 12 hodín
atlanta-ip	Viac ako 12 hodín	29,333.8	Viac ako 12 hodín
danoint	42,058.3	15,934.3	Viac ako 12 hodín
glass4	7,164.0	8,595.9	Viac ako 12 hodín
markshare1	18,472.0	2,653.1	Viac ako 12 hodín
markshare2	1,668.9	2,403.5	Viac ako 12 hodín
mkc	Viac ako 12 hodín	11,829.9	Viac ako 12 hodín
momentum1	Viac ako 12 hodín	22,500.0	Viac ako 12 hodín
momentum2	Viac ako 12 hodín	Viac ako 12 hodín	Viac ako 12 hodín
msc98-ip	40,301.0	18,638.8	Viac ako 12 hodín
net12	Viac ako 12 hodín	7,206.0	Viac ako 12 hodín
nsrand-ipx	35,601.0	6,119.8	Viac ako 12 hodín
protfold	Viac ako 12 hodín	36,491.3	Chyba pri načítaní súboru
rd-rplusc-21	Viac ako 12 hodín	27,685.0	Viac ako 12 hodín
roll3000	38,469.0	27,431.3	Viac ako 12 hodín
seymour	Viac ako 12 hodín	38,427.5	Viac ako 12 hodín
swath	2,763.0	5,332.0	Viac ako 12 hodín
timtab1	15,532.0	13,139.4	Viac ako 12 hodín
tr12-30	13,577.0	6,496.4	Viac ako 12 hodín

Porovnávanie je koncipované rovnakým spôsobom ako u prvej skupiny úloh. Prípadom keď sa vyskytla chyba počas výpočtu alebo výpočet neprebehol do limitného času 12 hodín bude automaticky priradená hodnota 3. Ďalej nasleduje tabuľka vyhodnotenia výpočtového času riešenia jednotlivými softvérmi.

Tabuľka 4.4: Tabuľka porovnania testovaných softvérov z hľadiska výpočtového času jednotlivých úloh druhej skupiny

Názov úlohy	CPLEX	Xpress	MOSEK
a1c1s1	3	1	3
aflow40b	2	1	3
arki001	1	2	3
atlanta-ip	3	1	3
danoint	2	1	3
glass4	1	2	3
markshare1	2	1	3
markshare2	1	2	3
mkc	3	1	3
momentum1	3	1	3
momentum2	3	3	3
msc98-ip	2	1	3
net12	3	1	3
nsrand-ipx	2	1	3
protfold	3	1	3
rd-rplusc-21	3	1	3
roll3000	2	1	3
seymour	3	1	3
swath	1	2	3
timtab1	2	1	3
tr12-30	2	1	3
Total	**47**	**27**	**63**

Z tejto tabuľky vidíme, že najlepšie dopadol softvér Xpress. Za ním najlepší bol softvér CPLEX a najhoršie dopadol softvér MOSEK. Z hľadiska doby výpočtu úloh z druhej skupiny je teda softvér Xpress najlepší.

4.2.2 Presnosť výsledkov riešenia

Druhé hľadisko porovnávania výkonnosti softvérov je presnosť výsledkov. Za najlepší budem považovať ten softvér, ktorého vypočítané riešenie je čo najbližšie k optimálnemu riešeniu, prípadne je samo optimálne. Ako porovnávaciu hodnotu budem uvažovať absolútny rozdiel vypočítaného a optimálneho riešenia. Samozrejme najlepšie je, pokiaľ je táto hodnota rovná nule. To znamená, že testovanému softvéru sa podarilo vypočítať optimálne riešenie. Ďalej nasleduje tabuľka s relatívnymi rozdielmi medzi optimálnymi a vypočitanými riešeniami testovaných softvérov.

Tabuľka 4.5: Relatívne rozdiely medzi optimálnymi a vypočítanými hodnotami výsledkov úloh druhej skupiny testovaných softvérov

Názov úlohy	CPLEX	Xpress	MOSEK
a1c1s1	-9.7%	2.3%	3.0%
aflow40b	0.0%	0.0%	n/a – 1 000%
arki001	0.0%	3.4%	0.0%
atlanta-ip	-7.6%	4.3%	10.7%
danoint	0.0%	0.0%	2.7%
glass4	23.8%	25.0%	50.0%
markshare1	98.0%	87.5%	100.0%
markshare2	95.5%	90.9%	100.0%
mkc	0.0%	0.1%	2.0%
momentum1	0.0%	22.7%	13.3%
momentum2	0.0%	11.5%	15.1%
msc98-ip	0.0%	0.0%	0.7%
net12	16.1%	0.0%	82.9%
nsrand-ipx	0.0%	900.0%	1.3%
protfold	-29.2%	19.2%	n/a – 1 000%
rd-rplusc-21	-165 295.0%	0.1%	165 295.0%
roll3000	0.2%	0.0%	2.4%
seymour	0.5%	0.2%	0.9%
swath	10.4%	0.2%	0.2%
timtab1	0.9%	90.2%	0.0%
tr12-30	0.0%	0.0%	0.0%
Total	**-16 5096%**	**1 257%**	**167 680%**

Pre úlohy, v ktorých sa vyskytla chyba, a teda nie je známe riešenie, som pre možnosť porovnania pridelila automaticky hodnotu rozdielu 1000%. Jednalo sa konkrétne o úlohy *trotfold* a *aflow40b* a chyba sa vyskytla pri načítaní úlohy softvérom MOSEK. Ako jedno kritérium porovnania by sa dal uvažovať súčet jednotlivých rozdielov s tým, že chceme tento súčet samozrejme čo najmenší. Ako vidieť, po sčítaní jednotlivých percentuálnych rozdielov dopadol najlepšie softvér Xpress, a to hlavne vďaka tomu, že v porovnaní s ostatnými dvomi testovanými softvérmi, ktoré vypočítali úlohu *rd-rplusc-21* s veľkým rozdielom, sa mu podarilo vypočítať uspokojivé riešenie. Najhoršie dopadol softvér MOSEK, ktorý vypočítal úlohu *rd-rplusc-21* s rovnakým rozdielom ako softvér CPLEX, ale navyše nebol schopný načítať ďalšie dve úlohy.

Porovnanie na základe súčtu jednotlivých rozdielov však nepovažujem za dostatočne objektívne, a preto je samotné porovnávanie opäť založené na prideľovaní poradových čísiel a platia preň rovnaké podmienky prideľovania poradových čísiel ako v predchádzajúcich prípadoch. Pokiaľ je riešenie optimálne, automaticky sa pridelí číslo 1, pokiaľ nastala chyba programu, pridelí sa číslo 3. Tabuľka na ďalšej stránke zobrazuje výsledky porovnania testovaných softvérov z hľadiska presnosti vypočítaných výsledkov. Toto poradie je počítané na základe absolútnych rozdielov od optimálneho riešenia.

Tabuľka 4.6: Výsledky porovnania testovaných softvérov z hľadiska presnosti vypočítaných výsledkov úloh druhej skupiny

Názov úlohy	CPLEX	Xpress	MOSEK
a1c1s1	3	1	2
aflow40b	1	1	3
arki001	1,5	3	1,5
atlanta-ip	2	1	3
danoint	1	1	3
glass4	1	3	2
markshare1	2	1	3
markshare2	3	2	1
mkc	1	2	3
momentum1	1	3	2
momentum2	1	2	3
msc98-ip	1	2	3
net12	2	1	3
nsrand-ipx	1	3	2
protfold	2	1	3
rd-rplusc-21	2,5	1	2,5
roll3000	2	1	3
seymour	2	1	3
swath	3	1	2
timtab1	2	3	1
tr12-30	1	1	1
Total	**32**	**35**	**46**

Z tohto hľadiska bol najlepší program CPLEX. O kúsok horšie dopadol softvér Xpress, ale jeho výkon so softvérom CPLEX je zrovnateľný. Najhoršie dopadol opäť softvér MOSEK, ktorého výsledky výpočtou boli najmenej presné.

4.3 Zhrnutie

Celkové porovnanie a vyhodnotenie výkonnosti jednotlivých softvérov je založené na štyroch rôznych kritériách.

Prvým je rýchlosť výpočtu úloh z prvej skupiny knižnice MIPLIB 2003. Tieto úlohy sú pomerne jednoduché a komerčný softvér by ich mal vedieť vypočítať do jednej hodiny. V tejto skupine úloh je presnosť výsledkov u všetkých testovaných softvérov dobrá, a teda som ju pri porovnávaní nebrala v úvahu. Podstatné je v tomto prípade porovnávať rozdiely v rýchlosti výpočtov. Z tohto hľadiska dopadol najlepšie softvér CPLEX. Hneď za ním, s porovnanteľným výkonom, bol softvér Xpress a najhoršie dopadol softvér MOSEK, ktorý sa ukázal ako najpomalejší v riešení úloh prvej skupiny.

Druhým kritériom je taktiež výpočtový čas, ale testovali sa úlohy z druhej skupiny knižnice MIPLIB 2003. Optimálne riešenie týchto úloh je známe. Jedná sa však o komplikovanejšie úlohy, ktorých výpočet môže trvať pomerne dlho, a preto je doba výpočtu vhodným kritériom pre porovnanie výkonnosti testovaných softvérov. Najlepšie v tomto ohľade dopadol softvér Xpress. Potom následoval softvér CPLEX a ako posledný bol opäť softvér MOSEK.

Ďalším porovnávacím kritériom je presnosť vypočítaného riešenia. Nakoľko v prvej skupine úloh, sme túto stránku neuvažovali, pri druhej skupiny úloh je táto stránka pomerne dôležitá, nakoľko presnosť výsledkov je rôzna pre jednotlivé úlohy a testované softvéry. Najlepšie opäť dopadol softvér CPLEX. Hneď po ňom nasledoval softvér Xpress s porovnateľne dobrými výsledkami. Ako najhorší sa opäť ukázal softvér MOSEK.

Posledným súhrnným kritériom je obsluha a práca s testovaným softvérom. Do tohto kritéria patrí jednoduchosť ovládania, úroveň nápovedy, prehľadnosť výstupov, možnosti nastavenia parametrov výpočtu, komunikačné rozhranie a ďalšie faktory, ktoré uľahčujú užívateľovi prácu so softvérom. Úroveň vypracovanej nápovedy a prehľadnosť výstupov bola u všetkých testovaných softvérov veľmi dobrá. Rovanko aj poskytované možnosti nastavenia parametrov výpočtu jednotlivými softvérmi boli veľmi široké a pre bežného užívateľa úplne postačujúce. Osobne ako užívateľ som veľmi ocenila grafické prostredie, ktoré poskytuje softvér Xpress. Oproti ostatným testovaným softvérom, ktoré sa ovládajú pomocou riadkových príkazov, sa grafické prostredie ukázalo ako prehľadnejšia a príjemnejšia varianta.

Po celkovom zhrnutí výsledkov nám vychádza, že najlepšie dopadol z hľadiska rýchlosti výpočtu malých úloh a presnosti riešenia komlikovanejších úloh softvér CPLEX. Softvér Xpress bol zasa najrýchlejší v dobe výpočtu komplikovanejších úloh a zároveň dosahoval aj porovnateľné výsledky s programom CPLEX v dobe riešenia jednoduchších úloh a výpočtovej presnosti komplikovanejších úloh. Najhoršie dopadol softvér MOSEK, ktorý bol ako posledný v hodnotení podľa kritérií presnosti a rýchlosti výpočtu u oboch typoch úloh. Čo sa týka práce s testovaným softvérom, dali by sa všetky softvéry považovať za porovnateľné. Jediné plus získal v tomto ohľade akurát softvér Xpress, a to vďaka svojmu grafickému prostrediu pre operačný systém Windows.

Osobne by som zvolila asi softvér Xpress, aj napriek tomu, že nie je vo všetkých kritériách najlepší. Moje rozhodutie sa opiera hlavne o faktory ako relatívne dobrá výkonnosť softvéru z hľadiska rýchlosti a presnosti vypočítaných výsledkov, ale hlavne príjemnosť a jednoduchosť pri práci so sofvérom v grafickom prostredí.

5. ZÁVER

Cieľom mojej diplomovej práce bolo otestovať vybraný optimalizačný softvér z hľadiska rýchlosti výpočtu a presnosti riešenia úloh zmiešaného celočíselného programovania, užívateľskej prístupnosti ovládania softvéru a dosiahnuté výsledky medzi sebou porovnať. Na porovnanie výkonnosti testovaných softvérov som použila úlohy zmiešaného celočíselného programovania z knižnice MIPLIB 2003. Stránku užívateľskej prístupnosti ovládania softvéru som hodnotila na základe vlastnej skúsenosti s prácou s testovanými softvérmi.

Pre možnosť vzájomného porovnania testovaných softvérov som sa snažila zachovať čo najrovnejšie podmienky tým, že všetky softvéry boli testované na jednom počítači vo svojich pôvodných nastaveniach. Odvodené závery a vzájomné porovnania testovaných softvérov popisujem detailne vo štvrtej kapitole. Výsledky testovania sa líšili v závislosti od jednotlivých kritérií. Po výkonnostnej stránke dopadol najlepšie softvér CPLEX a najhoršie softvér MOSEK. Program Xpress dopadol v jednom kritériu lepšie ako program CPLEX a v ostatných kritériách dosahoval porovnateľných výsledkov. Čo sa týka môjho osobného hodnotenia, dopadol najlepšie softvér Xpress, a to hlavne z dôvodu, že ako jediný podporoval grafické prostredie, čo bolo pre samotnú prácu s ním veľmi príjemná zmena oproti riadkovým príkazom. Možnosti nastavenia výpočtu pri riešení úloh zmiešaného celočíselného programovania boli u všetkých testovaných softvérov dostatočne dobré a použité postupy hľadania riešenia približne rovnaké. Vždy sa jednalo o fázy presolvingu, použitia heuristík a algoritmu metódy vetvenia a medzí alebo vetvenia a rezov.

Na výsledkoch testovania sa hlavne prejavilo nastavenie zvolené autormi softvérov. V prípade, že by sa menili nastavenia jednotlivých parametrov, mohli by výsledky vyzerať úplne inak. Myslím, že takéto rozšírenie mojej práce, by mohlo byť veľmi zaujímavé, nakoľko by preniklo hlbšie do podstaty testovaných softvérov, a tým aj lepším prehľadom o ich skutočnej kvalite a možnostiach.

Zoznam použitej literatúry

Literatúra:

[1] Atamturk, A., Savelsbergh, M.W.P.: Integer-Programming Software Systems. *Annals of Operations Research* 140 (2005), s. 5-18.

[2] Bartoň, J.: Testování software pro řešení úloh smíšeného celočíselného programování. VŠE, Praha 2002.

[3] Bixby, R., Fenelon, M., Gu, Z., Wunderling, R.: MIP: Theory and Practice – Closing the Gap. ILOG CPLEX Division, USA 1999.

[4] Jablonský, J.: Programy pro matematické modelování. VŠE, Praha 2007.

[5] Kovács, L.B.: Combinatorial methods of discrete programming. Akadémiai Kiadó, Budapest 1980.

[6] Pelikán, J.: Diskrétní modely v operačním výzkumu. Professional Publishing, Praha 2001.

Technická dokumentácia

[1] Xpress-IVE version 1.18.05 – Help. Dash Associates, 2007.

[2] Xpress-MP – Eurozone (€) Price List September 2007. Dash Associates, 2007.

[3] Xpress-MP – Getting started. Dash Associates, 2007.

Internetové zdroje

[1] COIN-OR
http://www.optimizationservices.org
http://www.coin-or.org/

[2] Dash Optimization
http://www.fairisaac.com/NR/exeres/AAF3DD63-ABE9-4FA7-B432-91E5B671224E,frameless.htm
http://www.dashoptimization.com/

[3] Historie – Operačný výzkum
http://www.seminarky.cz/Operacni-vyzkum-otazky-ke-zkousce-5287
http://www.lionhrtpub.com/orms/orms-10-02/frhistorysb1.html

[4] ILOG CPLEX
http://www.ilog.com/products/cplex/

[5] MATLAB
http://artax.karlin.mff.cuni.cz/~beda/cz/matlab/primercz/matlab-primer.html#intro

[6] MIPLIB 3.0
http://mathforum.org/library/view/7098.html
http://miplib.zib.de/miplib2003.php

[7] MOSEK
http://www.mosek.com/

[8] MPS Input Format
http://www-fp.mcs.anl.gov/OTC/Guide/OptWeb/continuous/constrained/linearprog/mps.html

[9] Parallel CPLEX and MOSEK on LP problems
http://plato.asu.edu/ftp/cpx_msk.html

[10] Savelsbergh, M.W.P.: Branch-and-Price: Integer programming with Column Generation.
http://www2.isye.gatech.edu/~mwps/publications/eoo.pdf

[11] Wikipédia
http://wikipedia.org/
http://wikipedia.cz/

Prílohy

Príloha č.1 – Ukážka úlohy vo formáte MPS

```
*NAME: flugpl
*ROWS: 18
*COLUMNS: 18
*INTEGER: 11
*NONZERO: 46
*BEST SOLN: 1201500 (opt)
*LP SOLN: 1167185.73
*SOURCE: Harvey M. Wagner
* John W. Gregory (Cray Research)
* E. Andrew Boyd (Rice University)
*APPLICATION: airline model
*COMMENTS: no integer variables are binary
*
*
NAME                 FLUGPL
ROWS
N KOSTEN
E ANZ1
G STD1
L UEB1
E ANZ2
G STD2
L UEB2
E ANZ3
G STD3
L UEB3
E ANZ4
G STD4
L UEB4
E ANZ5
G STD5
L UEB5
E ANZ6
G STD6
L UEB6
COLUMNS
   STM1        KOSTEN      2700  ANZ1             1
   STM1        STD1        150   UEB1             -20
   STM1        ANZ2        0.9
   MARKOOOO    'MARKER'    'INTORG'
   ANM1        KOSTEN      1500  STD1             -100
   ANM1        ANZ2        1
   MARKOO01    'MARKER'    'INTEND'
   UE1         KOSTEN      30    STDl             1
   UE1         UEB1        1
   MARKOO02    'MARKER'    'INTORG'
   STM2        KOSTEN      2700  ANZ2             -1
   STM2        STD2        150   UEB2             -20
   STM2        ANZ3        0.9
   ANM2        KOSTEN      1500  STD2             -100
   ANM2        ANZ3        1
   MARKOO03    'MARKER'    'INTEND'
   UE2         KOSTEN      30    STD2             1
   UE2         UEB2        1
```

```
MARKOO04    'MARKER'    'INTORG'
STM3        KOSTEN      2700  ANZ3              -1
STM3        STD3        150   UEB3              -20
STM3        ANZ4        0.9
ANM3        KOSTEN      1500  STD3              -100
ANM3        ANZ4        1
MARKOO05    'MARKER'    'INTEND'
```

Príloha č.2 – Pôvod úloh knižnice MIPLIB 2003

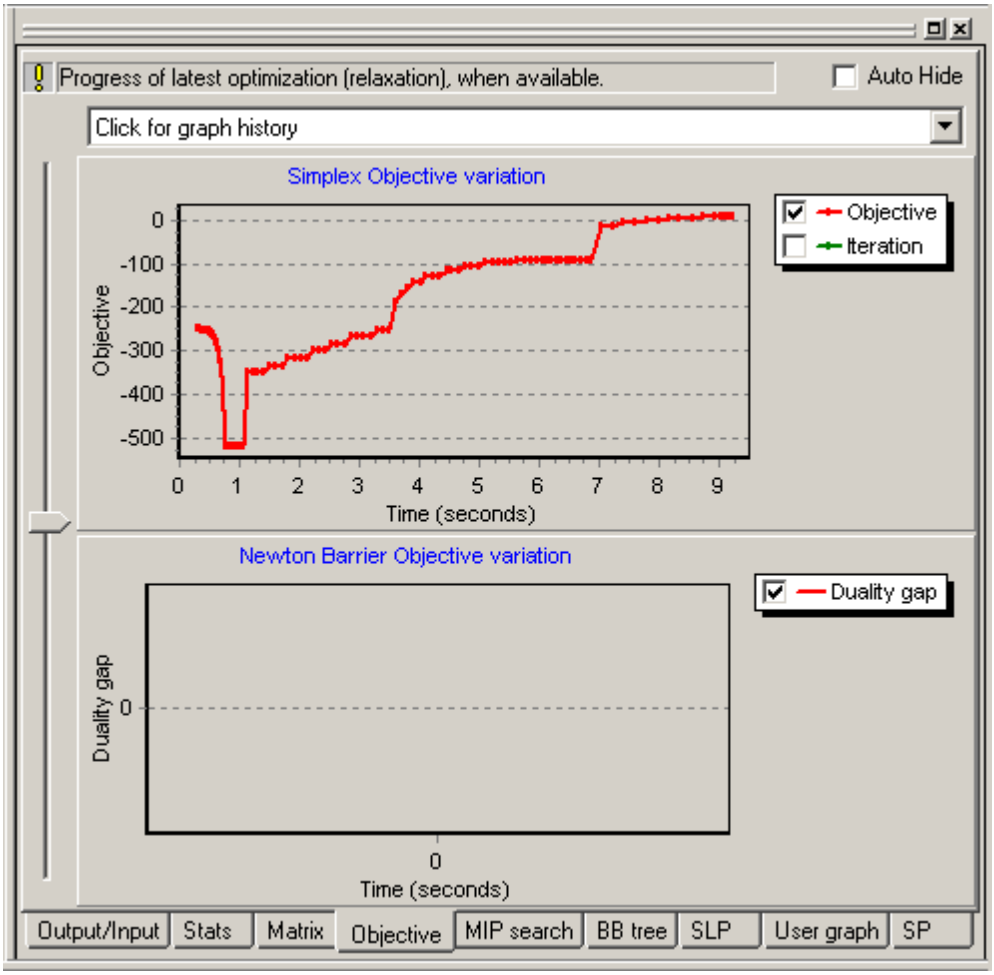
	Name	Originator	Formulator	Donator
1	10teams	Dash Associates	----	Bob Daniel
2	a1c1s1	M. Vyve, E. Pochet	M. Vyve, E. Pochet	Andrea Lodi
3	aflow30a	Tobias Achterberg	Tobias Achterberg	Tobias Achterberg
4	aflow40b	Tobias Achterberg	Tobias Achterberg	Tobias Achterberg
5	air04	----	----	Greg Astfalk
6	air05	----	----	Greg Astfalk
7	arki001	Avesta-Sheffield, Sweden	Nils Holmberg	Arne Stolbjerg Drud
8	atlanta-ip	Daniel Bienstock	Andreas Bley	Andreas Bley
9	cap6000	Karla Hoffman, Manfred Padberg	Telecommunications Corporation	Karla Hoffman
10	dano3mip	Bell Communications Research	----	Daniel Bienstock
11	danoint	Columbia's Center for Telecommunications Research	----	Daniel Bienstock
12	disctom	Sven de Vries	Sven de Vries	Sven de Vries
13	ds	Andreas Loebel	Andreas Loebel	Andreas Loebel
14	fast0507	Italian Railway Company	Pier Luigi Guida	Sebastian Ceria
15	fiber	US West	Youngho Lee	Martin Savelsbergh
16	fixnet6	----	T. J. Van Roy	Martin Savelsbergh
17	gesa2	Spanish Electricity	Laurence A. Wolsey	Sebastian Ceria
18	gesa2-o	Spanish Electricity	GESA (Consulting Company)	Sebastian Ceria
19	glass4	I. Luzzi	I. Luzzi	Andrea Lodi
20	harp2	----	----	Martin Savelsbergh
21	liu	University of Washington Electrical Engineering Dept.	Xiaowen Liu	Ed Klotz, ILOG
22	manna81	Arie Koster, Adrian Zymolka	Arie Koster, Adrian Zymolka	Arie Koster, Adrian Zymolka
23	markshare1	Gerard Cornuejols, Milind Dawande	Gerard Cornuejols, Milind Dawande	Milind Dawande
24	markshare2	Gerard Cornuejols, Milind Dawande	Gerard Cornuejols, Milind Dawande	Milind Dawande
25	mas74	----	----	Jonathan Eckstein
26	mas76	----	----	Jonathan Eckstein
27	misc07	----	----	Greg Astfalk
28	mkc	Jayant Kalagnanam, Milind Dawande	Jayant Kalagnanam, Milind Dawande	Jayant Kalagnanam, Milind Dawande
29	mod011	Uwe H. Suhl	Uwe H. Suhl	John J. Forrest
30	modglob	Y. Smeers	Laurence A. Wolsey	Martin Savelsbergh
31	momentum1	IST-2000-28088 Project MOMENTUM	WP4 Team	Thorsten Koch
32	momentum2	IST-2000-28088 Project MOMENTUM	WP4 Team	Thorsten Koch
33	momentum3	IST-2000-28088 Project MOMENTUM	WP4 Team	Thorsten Koch
34	msc98-ip	E-Plus, Roland Wessaely	Andreas Bley	Andreas Bley
35	mzzv11	Sascha Lukac	Sascha Lukac	Sascha Lukac
36	mzzv42z	Sascha Lukac	Sascha Lukac	Sascha Lukac
37	net12	P. Belotti	P. Belotti	Andrea Lodi
38	noswot	----	Linus E. Schrage	John W. Gregory

39	nsrand-ipx	L. Kroon	L. Kroon	Andrea Lodi
40	nw04	Northwest Airlines	----	Karla Hoffman
41	opt1217	Telecommunication Networks Group, TU Berlin	James Gross	Hans-Florian Geerdes
42	p2756	CJP set	Ellis L. Johnson	E. Andrew Boyd
43	pk1	----	Pinar Keskinocak	Sebastian Ceria
44	pp08aCUTS	----	----	Martin Savelsbergh
45	pp08a	----	----	Martin Savelsbergh
46	protfold	TU Darmstadt	A. Fügenschuh	A. Fügenschuh
47	qiu	Yu-Ping Chiu	Yu-Ping Chiu	Jonathan Eckstein
48	rd-rplusc-21	----	Utz-Uwe Haus, Dennis Michaels, Robert Weismantel	Utz-Uwe Haus
49	roll3000	L. Kroon	L. Kroon	Andrea Lodi
50	rout	S. Graves	Hernan Abeledo	Sebastian Ceria
51	set1ch	----	Laurence A. Wolsey	Martin Savelsbergh
52	seymour	----	Bill Cook	Paul Seymour
53	sp97ar	J. W. Goessens, S. van Hoesel, L. Kroon	J. W. Goessens, S. van Hoesel, L. Kroon	Andrea Lodi
54	stp3d	ZIB	David Grove Joergensen, Thorsten Koch	Thorsten Koch
55	swath	David Panton	David Panton	David Panton
56	t1717	Telebus Berlin	Ralf Borndoerfer	Ralf Borndoerfer
57	timtab1	----	C. Liebchen, R.H.Moehring	C. Liebchen, R.H.Moehring
58	timtab2	----	C. Liebchen, R.H.Moehring	C. Liebchen, R.H.Moehring
59	tr12-30	M. Vyve, E. Pochet	M. Vyve, E. Pochet	Andrea Lodi
60	vpm2	----	Laurence A. Wolsey	Martin Savelsbergh

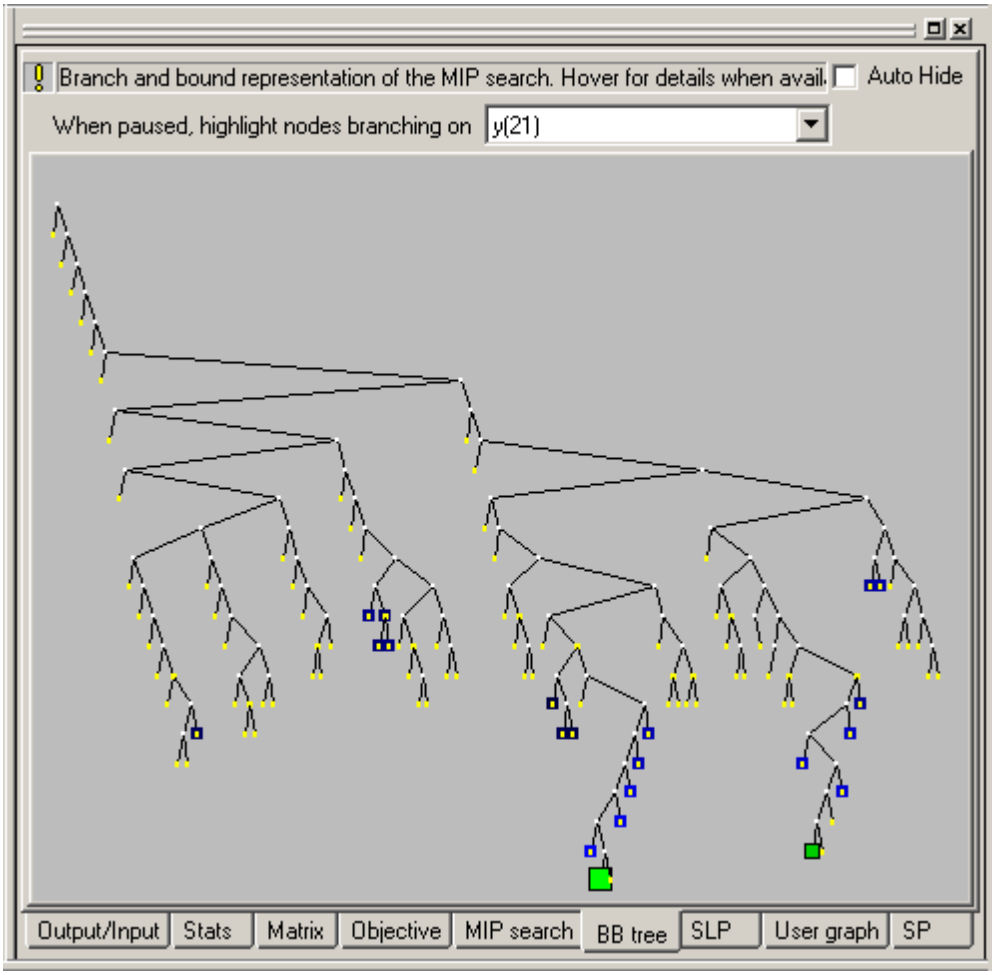
Príloha č.3 – Popis základných položiek menu softvéru Xpress

Menu	Možnosti	Význam
File	*New*	Vytvorí nový súbor
	Open	Otvorí existujúci súbor
	Close	Zatvorí aktuálny súbor
	Save	Uloží aktuálny súbor
	Save as	Uloží aktuálny súbor ako súbor s iným názvom alebo príponou
	Save as HTML	Uloží aktuálny súbor ako web stránku
	Print	Vytiskne aktuálny súbor
	Print Setup	Zmení nastavenia tisku
	Send by email	Pošle aktuálny súbor emailom
	Recently used files	Ukáže posledne otvorené súbory
	Exit	Ukončí program Xpress-IVE
Edit	*Undo*	Choď o krok späť
	Redo	Choď o krok dopredu
	Cut	Vystrihni
	Copy	Skopíruj
	Paste	Vlož
	Find...	Nájdi ...
	Replace...	Nahraď ...
	Go To Line...	Choď na zvolené číslo riadku v editore
	Select All	Označ celý text v editore
View		Zobrazuje a skrýva jednolitvé položky panela nástrojov
Build	*Compile*	Uloží a zkompiluje aktuálny .mos súbor
	Run	Uloží, zkompiluje a spustí výpočet aktuálneho .mos súboru
	Pause	Pozastaví výpočet
	Stop	Preruší výpočet
	Option	Otvorí okno s možnosťami nastavenia Moselu a Optimizeru
	Optimize matrix file...	Otvorí okno s nastaveniami pre optimalizáciu matíc v LP alebo MPS formáte
	Export matrix...	Otvorí okno pre exportovanie výpočtových matíc do formátov LP a MPS
	Send to NEOS...	Otvorí okno pre posielanie Mosel mdelov na NEOS server pre optimalizáciu
Debug		Zobrazuje možnosti odstaňovania chýb
Deploy		Otvorí okno Deploy Dialog
Modules		Zobrazuje možnosť vytvorenia nového modelu a list už existujúcich modulov
Wizard		Zobrazuje možnosti jazyka Mosel
Window		Umožnuje rôzne zobrazenia okien
Help	*Xpress-IVE Help...*	Otvorí menu systému
	Xpress-MP Help...	Otvorí dokumentáciu všetkých komponentov sady Xpress-MP
	Go to Web Site	Otvorí online dokumentáciu
	About Xpress-IVE...	Zobrazí základné informácie o danej verzii programu

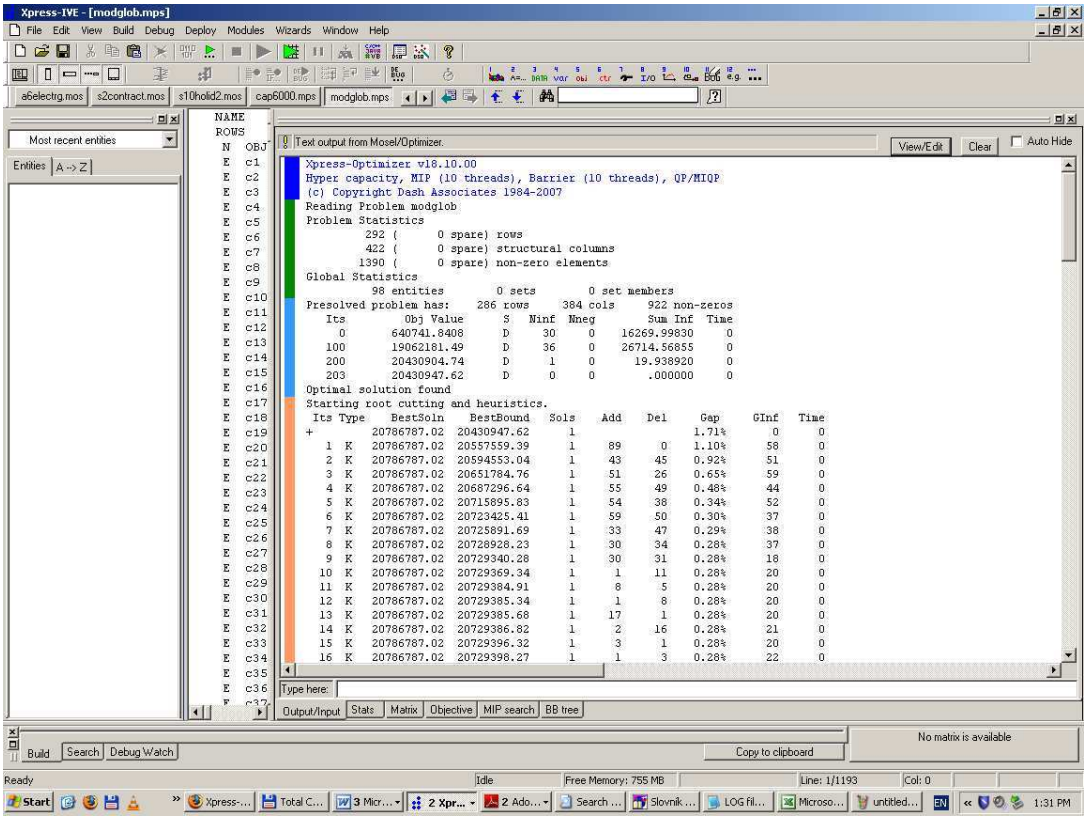
Príloha č.4 – Okno *Run Bar – Objective* - softvér Xpress



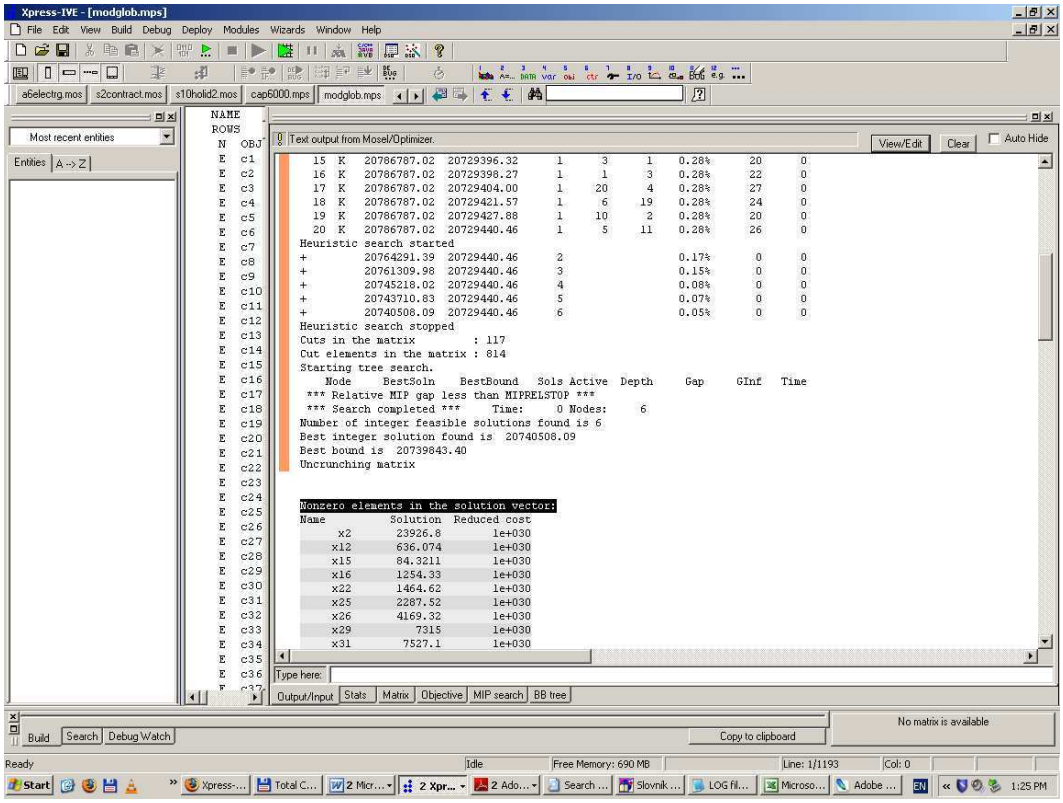
Príloha č.5 – Okno *Run Bar – BB tree* - softvér Xpress



Príloha č.6 – Okno *Run Bar – Output/Input* – časť 1 - softvér Xpress



Príloha č.7 – Okno *Run Bar – Output/Input* – časť 2 - softvér Xpress



Príloha č.8 – Ukážka LOG súboru v ASCII - softvér Xpress

```
Xpress-Optimizer v18.10.00
Hyper capacity, MIP (10 threads), Barrier (10 threads), QP/MIQP
(c) Copyright Dash Associates 1984-2007
Reading Problem fast0507
Problem Statistics
         508 (      0 spare) rows
       63009 (      0 spare) structural columns
      472358 (      0 spare) non-zero elements
Global Statistics
       63009 entities        0 sets        0 set members
Presolved problem has:    482 rows   45382 cols  292379 non-zeros
LP relaxation tightened
   Its         Obj Value      S   Ninf  Nneg        Sum Inf  Time
     0         14.000000      D    482     0     482.000000    79
   100         65.000000      D    312     0     371.000000    80
   200         98.000000      D    206     0     371.000000    80
   300        112.000000      D    174     0     371.000000    80
   400        112.000990      D    179     0    1146.437500    80
   500        114.003143      D    173     0    1146.437500    80
   600        115.002677      D    173     0    1408.500000    80
   700        127.668829      D    204     0    1275.833333    80
   800        130.356443      D    225     0    1275.833333    80
   900        136.712191      D    209     0    1959.454545    81
  1000        139.104326      D    216     0    1959.454545    81
  1100        143.253290      D    270     0    10135.51149    82
  1200        145.976336      D    239     0    2052.943396    82
  1300        147.062448      D    252     0    10368.50837    83
  1400        148.703532      D    271     0     834.182243    83
  1500        150.799264      D    292     0    1277.973917    84
  1600        152.810969      D    302     0    11307.43929    84
  1700        154.420321      D    285     0    2755.858055    85
  1800        158.102296      D    296     0    6659.780231    86
  1900        159.858902      D    289     0    4013.366984    86
   Its         Obj Value      S   Ninf  Nneg        Sum Inf  Time
  2000        161.123156      D    271     0     827.246924    87
  2100        162.427960      D    235     0    1468.416598    87
  2200        163.343563      D    263     0    1874.906559    88
  2300        164.396330      D    264     0    1227.276035    88
  2400        165.202126      D    267     0    5452.839050    89
  2500        165.748464      D    239     0    4521.830649    90
  2600        166.434775      D    272     0    4357.265112    91
  2700        166.924223      D    238     0     464.441380    91
  2800        167.631832      D    246     0    2829.036968    92
  2900        168.106764      D    270     0     370.838566    93
  3000        168.478469      D    244     0     802.351307    94
  3100        168.843749      D    289     0    3040.886243    95
  3200        169.162393      D    261     0    1478.311358    95
  3300        169.442153      D    273     0     249.439053    96
  3400        169.764433      D    206     0    1788.373808    97
  3500        169.901935      D    273     0    1037.300634    98
  3600        170.071129      D    299     0    1851.505671    99
  3700        170.377347      D    281     0    1475.817792   100
  3800        170.658664      D    228     0     295.289230   101
  3900        170.917649      D    208     0     418.803263   102
   Its         Obj Value      S   Ninf  Nneg        Sum Inf  Time
  4000        171.141685      D    205     0     520.815396   103
  4100        171.371928      D    300     0     386.787866   104
```

```
  4200        171.542894      D    248     0     192.162895   105
  4300        171.736545      D    166     0     226.287752   106
  4400        171.850669      D    246     0     392.852583   107
  4500        171.949433      D    131     0     123.985656   108
  4600        172.015712      D    162     0     264.078128   109
  4700        172.104739      D    159     0      61.925872   111
  4800        172.129460      D    103     0     261.474658   112
  4900        172.148463      D      3     0      17.387474   113
  4902        172.145567      P      0     0        .000000   113
  4902        172.145567      P      0     0        .000000   113
Optimal solution found
Starting root cutting and heuristics.
 Its Type    BestSoln    BestBound   Sols    Add    Del     Gap     GInf
Time
+          230.000000   172.145567      1                 25.15%       0
0
+          189.000000   172.145567      2                  8.92%       0
2
   1  K    189.000000   172.145567      2      2      0    8.92%     284
5
   2  K    189.000000   172.145567      2      1      1    8.92%     279
7
   3  K    189.000000   172.145567      2      0      1    8.92%     279
9
   4  G    189.000000   172.155770      2      1      0    8.91%     259
15
   5  G    189.000000   172.155770      2      1      1    8.91%     262
17
Heuristic search started
+          180.000000   172.155770      3                  4.36%       0
99
+          179.000000   172.155770      4                  3.82%       0
101
Heuristic search stopped
Cuts in the matrix         : 2
Cut elements in the matrix : 87093
Starting tree search.
    Node     BestSoln    BestBound   Sols Active  Depth     Gap     GInf
Time
+     77   177.000000   172.331947      5     43     43    2.64%       0
650
*     92   175.000000   172.331947      6     53     53    1.52%       0
738
     100   175.000000   172.331947      6     50     53    1.52%      73
738
     200   175.000000   172.370546      6     62     19    1.50%     186
1083
*    230   174.000000   172.370546      7     77     36    0.94%       0
1197
     300   174.000000   172.370546      7     38     37    0.94%     188
1197
     400   174.000000   172.464953      7     52      5    0.88%     272
1466
     500   174.000000   172.529347      7     85      6    0.85%     281
1766
     600   174.000000   172.576727      7    112     13    0.82%     263
2094
     700   174.000000   172.634523      7    127      9    0.78%     248
2404
```

```
     800   174.000000   172.689114      7    137      7    0.75%     252
2678
     900   174.000000   172.720016      7    145      9    0.74%     256
2954
    1000   174.000000   172.752903      7    151      7    0.72%     258
3207
    1100   174.000000   172.786790      7    144      9    0.70%     262
3443
    1200   174.000000   172.824830      7    124     14    0.68%     272
3696
    1300   174.000000   172.846116      7    103      8    0.66%     243
3905
    1400   174.000000   172.875542      7     72      8    0.65%     256
4126
    1500   174.000000   172.909318      7     34     12    0.63%     263
4340
 *** Search completed ***     Time:  4468 Nodes:  1561
Number of integer feasible solutions found is 7
Best integer solution found is   174.000000
Uncrunching matrix


Nonzero elements in the solution vector:
Name          Solution  Reduced cost
    x246             1        1e+030
    x247             1        1e+030
    x310             1        1e+030
    x582             1        1e+030
    x611             1        1e+030
   x2010             1        1e+030
   x2064             1        1e+030
   x2417             1        1e+030
   x2760             1        1e+030
   x2857             1        1e+030
   x3022             1        1e+030
   x3099             1        1e+030
   x3155             1        1e+030
   x3173             1        1e+030
   x3265             1        1e+030
   x3459             1        1e+030
   x4783             1        1e+030
   x5235             1        1e+030
   x5891             1        1e+030
   x6230             1        1e+030
   x6319             1        1e+030
   x6860             1        1e+030
   x7179             1        1e+030
   x7280             1        1e+030
   x7860             1        1e+030
   x9031             1        1e+030
   x9262             1        1e+030
   x9345             1        1e+030
   x9964             1        1e+030
  x10343             1        1e+030
  x10350             1        1e+030
  x10718             1        1e+030
  x11077             1        1e+030
  x11208             1        1e+030
  x11397             1        1e+030
```

x11595	1	1e+030
x11649	1	1e+030
x11812	1	1e+030
x11959	1	1e+030
x12130	1	1e+030
x12646	1	1e+030
x13075	1	1e+030
x13298	1	1e+030
x15763	1	1e+030
x16397	1	1e+030
x17877	1	1e+030
x18538	1	1e+030
x18571	1	1e+030
x19940	1	1e+030
x20647	1	1e+030
x20668	1	1e+030
x20669	1	1e+030
x23811	1	1e+030
x24173	1	1e+030
x27080	1	1e+030
x27237	1	1e+030
x27848	1	1e+030
x30827	1	1e+030
x32205	1	1e+030
x33131	1	1e+030
x33190	1	1e+030
x36734	1	1e+030
x38071	1	1e+030
x38221	1	1e+030
x38382	1	1e+030
x38464	1	1e+030
x38910	1	1e+030
x39077	1	1e+030
x39709	1	1e+030
x40624	1	1e+030
x40833	1	1e+030
x41128	1	1e+030
x41196	1	1e+030
x41220	1	1e+030
x41414	1	1e+030
x41925	1	1e+030
x42320	1	1e+030
x43766	1	1e+030
x43868	1	1e+030
x44684	1	1e+030
x44977	1	1e+030
x45480	1	1e+030
x45642	1	1e+030
x47177	1	1e+030
x47860	1	1e+030
x47897	1	1e+030
x48024	1	1e+030
x48277	1	1e+030
x48293	1	1e+030
x48492	1	1e+030
x48923	1	1e+030
x49228	1	1e+030
x53276	1	1e+030
x55671	1	1e+030

```
x55675              1         1e+030
x55760              1         1e+030
x56642              1         1e+030
x56704              1         1e+030
x56893              1         1e+030
x56917              1         1e+030
x56965              1         1e+030
x57102              1         1e+030
x57150              1         1e+030
x57906              1         1e+030
x57922              1         1e+030
x60233              1         1e+030
x60915              1         1e+030
x61650              1         1e+030
x61899              1         1e+030
x62291              1         1e+030
x62384              1         1e+030
x62404              1         1e+030
```

Príloha č.9 – Menu *Option* - softvér Xpress