Tartalomjegyzék:

1. Bevezetés:

Diplomamunkám és kutatásom célja, hogy összegyűjtsek több különböző operációkutatási solvert, összehasonlítsam teljesítményüket, numerikus stabilitásukat, árukat, használhatóságukat, és ezek alapján rangsoroljam őket. Célom hogy a solverek közt legyenek kereskedelmi szoftverek és ingyenes szoftverek is. Célom továbbá hogy az összehasonlított solverek között legyen a Lindo programcsomagja is, mert az egyetemünk oktatás során a Lindo szoftvereit használja.

Diplomamunkámban nagyobb hangsúlyt szeretnék fektetni a Lindo programcsomagjára a LindoApi-ra. A LindoApi-t nem csak feladatmegoldásra, é a megoldások analizálására használom a kutatásom során, hanem Microsoft Windows-os felületű GUI-t is fejlesztek hozzá. A programom segítségével az operációkutatásban igen elterjedt MPS formátumban megadott optimalizálási feladatokat lehet megoldani, és a fontosabb paraméterek megadására is lehetőség van.

Kutatásom során csak és kizárólag lineáris programozási feladatokkal foglalkozom, mert ezek a fajta feladatok leggyakoribb operációkutatási problémák. A lineáris programozási feladatok nagy részének igen egyszerű és triviális megoldása van, ezért igyekeztem bonyolult és nagyon rosszul megoldható feladatokat keresni, amik általában rosszul kondicionáltak, illetve ritka mátrixokkal dolgoznak. Ilyen példafeladatokat a Netlib oldaláról gyűjtöttem, amelyek közül több is igen ismert és híres az ezzel foglalkozó körökben. Úgy gondolom, hogy ezen feladatok megoldása során fog az igazi különbség jelentkezni a különböző solverek között, és én ezeket a különbségeket szeretném feltárni. A példafeladatok megoldásai ismertek. A példafeladatok MPS formátumban vannak, ezt a formátumot minden általam használt kereskedelmi és ingyenes solver felismeri.

Kutatásom során nyolc különböző solvert fogok használni az összegyűjtött feladatok megoldására. Vizsgálni fogom az egyes solverek paraméterezhezőségét, megoldásainak pontosságát, és a megoldás kiszámolásához szükséges időt.

A választásom az alábbi solverekre esett:

- LindoApi
- CPLEX
- CoinMP
- GLPK (Gnu Linear Programming Kit)
- LPSolve
- LGO
- Conopt
- Conopt2

LindoApi-t saját fejlesztésű programon keresztül fogok használni, részletesen bemutatva a fejlesztés menetét C nyelv alatt, a többi solvert pedig DLL formában használom feladatmegoldásra. A DLL-ek használatát a *Maximal Software* vállalat *MPL for Windows 4.2* programjának a segítségével oldom meg.

Diplomamunkám célja hogy létrehozzak egy mért adatokon alapuló összehasonlítást a különböző kereskedelmi és ingyenes solverek között, és ezek alapján összehasonlítsam őket. Célom kideríteni, hogy a nagyon nehezen megoldható lineáris programozási feladatok megoldásában vajon többet tudnak-e nyújtani a drága kereskedelmi programok, mint az ingyenes társaik. Célom egy saját fejlesztésű program megírása és dokumentálása LindoApi használatára C nyelv használatával, mert akik komolyan szeretnének foglalkozni operációkutatási problémák megoldásával azoknak nagy segítséget jelenthet ez a folyamatosan fejlődő és nagyon részletesen dokumentált programcsomag.

Diplomamunkám jelentőségét abban látom, hogy ilyen irányú összehasonlítást se magyar se angol nyelven nem találtam sehol ahol én kerestem, és hogy így egy munkába összegyűjtöttem több különböző operációkutatási solvert, azok képességeit, esetleges hiányosságait megkönnyítve így a választást mások számára. Munkám során bemutatom egy példán keresztül a LindoApi-t használó program fejlesztését C nyelven. Ez a diplomamunka segítséget nyújthat azok számára, akik bármilyen okból operációkutatási feladatok megoldásához keresnek már létező solvereket és keresik a lehetséges alternatívákat.

2. Alapok

2.1 Operációkutatás:

Az élet egyre több területén jönnek elő olyan problémák, amikor eszközök és erőforrások használatát a lehető leggazdaságosabban kell beosztani. Ezekre a problémákra, és ezek megoldására jött létre az operációkutatás tudományága. Az operációkutatás tárgya az ilyen problémák matematikai modelljének felállítása és ezen problémákra optimális megoldás keresése.

2.2 Lineáris programozás:

Lineáris programozáson olyan optimalizálási feladat megoldását értjük, ahol mind az optimalizálandó kifejezés, mind a feltételek lineáris függvényei a feladatban szereplő változóknak. A lineáris programozási feladat általános formája:

$$\boldsymbol{cx} \text{ -- > max (vagy min)}$$
$$\boldsymbol{Ax} = \boldsymbol{b}$$
$$(\boldsymbol{x} >= 0)$$

Ahol $\boldsymbol{A}$ egy $\boldsymbol{m}$ x $\boldsymbol{n}$ méretű mátrix, $\boldsymbol{b}$ egy $\boldsymbol{m}$ komponensű vektor, $\boldsymbol{c}$ egy $\boldsymbol{n}$ komponensű vektor, és mind meg vannak adva. Ekkor a feladat az, hogy megkeressük $\boldsymbol{x}$ $\boldsymbol{n}$ komponensű vektort, amely maximalizálja (vagy minimalizálja) a $\boldsymbol{cx}$ lineáris függvényt. Ez az általános forma a lineáris programozási feladat kanonikus alakja. Általános formában egy lineáris programozási feladat tartalmazhat egyenlőtlenségi feltételt vagy előjel kötetlen változókat, de ugyanaz a feladat megadható több különböző ekvivalens módon. Minden lineáris programozási feladat felírható kanonikus alakban.

2.3 MPS fájlformátum

Mivel kutatásom során a lineáris programozási feladatokat különböző solverekkel fogom megoldani, ezért találnom kellett egy egységes fájlformátumot, amit minden solver támogat, hogy ugyanazokat a feladatokat változatlan formában tudjam átadni mindegyik programnak. A választásom az MPS fájlformátumra esett. Az MPS a Mathematical Programming System rövidítése, amely lineáris programozási feladatok és kevert egészértékű (mixed integer) programozási feladatok tárolására és archiválására hoztak létre. Az MPS formátumot egy korai IBM program után neveztek el, és vált „de facto" ASCII szabvánnyá. Azóta az operációkutatási solverek nagyon nagy része (szinte mindegyik) ismeri és használja ezt a formátumot.

Az MPS oszlop-orientált formátum, ami azt jelenti, hogy nem egyenletekként kell megadnunk a modellt. A modell komponensei (változók, sorok) neveket kapnak. Az MPS egy nagyon régi formátum, még lyukkártyákra tervezték ezért nem szabad-formátum. Az MPS-ben megadott feladat szegmenseinek elnevezései vannak, és ezek az elnevezések a szegmens első sorában kapnak helyet. Bár tipikusan mindenhol mindent nagybetűvel kell írni a formátum történelmi okai miatt, de napjaink MPS olvasói/értelmezői elfogadnak bármilyen kis és nagybetűs MPS állományt is, illetve vannak olyanok, ahol megkövetelik, hogy a szegmensek elnevezései csupa nagybetűsek legyenek, de egyébként pedig elfogadja a kis és nagybetűket is bárhol máshol. A nevek, amelyek megadhatóak az egyes változóknak illetve megszorításoknak a solver szempontjából lényegtelenek, tehát igyekezzünk értelmes vagy egyszerű nevet adni nekik a későbbi olvasás és szerkesztés megkönnyítése miatt.

Egy probléma megadása MPS formátumban:

Egy példán keresztül bemutatok egy feladatot általános alakban és MPS formátumban felírva. A példában látszani fog, ahogy MPS formában felírva a lineáris programozási feladat különböző kötelező és opcionális szegmensekre bomlik.

A példafeladat általános alakban:

```
Optimize
 COST:     XONE + 4 YTWO + 9 ZTHREE
Subject To
 LIM1:     XONE + YTWO < = 5
 LIM2:     XONE + ZTHREE > = 10
 MYEQN:    - YTWO + ZTHREE  = 7
Bounds
 0 < = XONE < = 4
 -1 < = YTWO < = 1
End
```

A példafeladat MPS formátumban:

```
NAME          TESTPROB
ROWS
 N  COST
 L  LIM1
 G  LIM2
 E  MYEQN
COLUMNS
    XONE      COST                 1   LIM1                 1
    XONE      LIM2                 1
    YTWO      COST                 4   LIM1                 1
    YTWO      MYEQN               -1
    ZTHREE    COST                 9   LIM2                 1
    ZTHREE    MYEQN                1
RHS
    RHS1      LIM1                 5   LIM2                10
    RHS1      MYEQN                7
BOUNDS
 UP BND1      XONE                 4
 LO BND1      YTWO                -1
 UP BND1      YTWO                 1
ENDATA
```

Az MPS formátum szegmensei:

- NAME: Itt lehet megadni a lineáris programozási feladat nevét, a feladat megoldása szempontjából a név lényegtelen.

- ROWS: Itt lehet megadni a célfüggvényt, a megkötéseket. E-vel lehet megadni az egyenlőségi feltételt, L-el a kisebb-egyenlő relációt, G-vel a nagyobb-egyenlő relációt, N-el pedig a nem megkötött sorokat. A ROWS szegmensben a sorok sorrendje a probléma megoldásának szempontjából lényegtelen.

- COLUMNS: Ebben a szegmensben van megadva minden egyes változónak a szorzója, hogy a célfüggvényben illetve a megkötésekben az egyes változók milyen szorzóval szerepelnek. Természetesen ha egy változó egy megkötésben nem szerepel, vagyis nullaszor van benne, akkor az itt sincs külön megadva. Tulajdonképpen ebben a szegmensben van megadva a feladat $\boldsymbol{A}$ mátrixa. A sorok sorrendje a feladat megoldásának szempontjából lényegtelen.

- RHS: Ebben a szegmensben tudjuk megadni a jobb oldali vektort (right-hand-side). Egy vagy több vektor is megadható, egynél többet csak nagyon ritkán adnak meg. Több jobboldali vektor esetén meg kell adni, hogy a feladat megoldása során melyik jobboldali vektort kell használni. A szegmensben a jobb oldali vektor el van nevezve, és amely megkötésre nincs érték megadva, ott alapértelmezésként 0 szerepel a jobb oldali vektorban.

- RANGES: Ebben a szegmensben lehet a ROWS szegmensben megadott kisebb-egyenlő és nagyobb-egyenlő relációihoz intervallumot hozzárendelni. Tehát amelyik megkötésnek a RANGES szegmensben intervallumot adnak, annak így meg van adva az alsó és a felső korlátja is. Legyen a megkötésünk jelülése C_i, az RHS szegmensben hozzárendelt érték B_i, és a RANGES szegmensben hozzárendelt érték R_i. Ekkor ha C_i kisebb-egyenlő reláció, akkor az értéke $(B_i - | R_i |) <= C_i <= B_i$ tartományba esik. Ha pedig C_i nagyobb-egyenlő reláció, akkor az értéke $B_i <= C_i <= (B_i + | R_i |)$ tartományban kell legyen. A RANGES szegmens opcionális, nem kell benne lennie az MPS állományban, ha nincs rá szükség. A lineáris programozási feladatok nagytöbbségében nincs RANGES szegmens.

- BOUNDS: Ebben a szegmensben a lineáris programozási feladat változóinak értékeire vonatkozó korlátokat adjuk meg. Ha egy változónak nincs alsó korlátja, akkor alapértelmezésképpen nulla az alsó korlátja, vagyis nem vehet fel negatív értéket. Persze ha negatív számot adunk meg alsó korlátnak, akkor felvehet negatív értéket. A BOUNDS szegmens is ugyanúgy opcionális, mint a RANGES.

- ENDDATA: Ebben a szegmensben nem kell semmit megadni, ez jelzi az MPS állomány végét.

2.4 Példafeladatok

A solverek tesztelésére az operációkutatásban jól ismert különleges feladatokat kerestem, olyanokat, amik még elég kicsi problémák ahhoz, hogy a hallgatói licence megkötéseibe beférjenek, de elég nagyok és bonyolultak legyenek ahhoz, hogy ne legyen triviális megoldásuk és programmal kelljen őket megoldani. A választott feladatok jól ismertek, több fórumon is kitárgyaltak, és a megoldásaikat ismerjük. Ezek a feladatok különlegesek valamilyen szempontból. Lehetnek bennük megegyező sorok vagy oszlopok, akár több is, lehet bennük rengeteg szabad változó, lehetnek majdnem megoldhatatlanok, lehet bennük sok szingleton sor, szabad szingleton oszlop (ami az előfeldolgozás során akár el is tüntethető a feladatból), lehetnek nagyon kiegyensúlyozatlanok, lehetnek rosszul kondicionáltak, lehetnek benne ritka mátrixok. Ezek a tulajdonságok mind nehézséget okozhatnak a solvereknek, és okozhatják azt, hogy vagy megoldhatatlannak látják a feladatot, vagy rossz megoldást találnak a solverek.

3. Solverek

3.1 LindoApi:

Teljes neve Lindo Application Programming Interface. Kereskedelmi szoftver az amerikai székhelyű LINDO Systems gondozásában. A LINDO Systems egy világcég, amelynek operációkutatási programok fejlesztése a fő profilja. A LINDO egy 21 éves vállalat, saját állításaik szerint ők a világ vezető operációkutatási programjait gyártó cége. Termékeiket a *Fortune 500* listán szereplő vállalatok fele használja, az első 25 helyezettből 23 is többek közt. A *Fortune 500* lista a CNN pénzügyi részének a listája, amely tartalmazza az 500 legnagyobb amerikai vállalatot.

A LindoApi arra készült, hogy program-fejlesztők ennek a segítségével használják a Lindo már megírt függvényeit a saját programjaikban. Véleményem szerint ez nagyon is jól sikerült, én speciel minden akadály nélkül használatba tudtam venni, pedig ilyen irányú tapasztalataim egyáltalán nem voltak ezelőtt. Mint írtam a LindoApi kereskedelmi szoftver, vagyis pénzbe kerül. Az árképzésbe természetesen benne van, hogy ők maguk világelső cégnek pozícionálják magukat. Oldalukon megtalálható a netes vásárlás lehetősége is LINDO STORE néven, ami itt található: https://www.lindo.com/store/

Lehet venni általános és oktatói licencet is, illetve kipróbálásra van ingyenes hallgatói licence, ami nagyon le van korlátozva. Én hallgatói licencet használtam a kutatásomhoz, illetve próbáltam a kutatásom idejére kérni valami komolyabb licencet, de leveleimet válaszra sem méltatták. A megvásárolt licence mellé természetesen jár szoftver kézikönyv, és felhasználói útmutató. (Ezek egyébként külön is megvásárolhatók) A Lindo folyamatosan fejleszti a programjait, alkalmazásait, de arra vonatkozó információt, hogy a régebben megvett licensszel lehet-e használni az újabb kiadást, vagy milyen kondíciói vannak a frissítésnek, erre vonatkozó információkat sajnos nem találtam. (Erre irányuló levelemre sem érkezett válasz)

Akkor lássuk az árakat:

Oktatói licence: (195 - 995 $ között)

LINDO API 4.1 - educational

Single user educational licenses are available to qualified educational institutions and students. Educational licenses are restricted to educational instruction and educational research.

Super LINDO API	Hyper LINDO API
educational Price: 195.00 $ CONSTRAINTS: 1000 VARIABLES: 2000 INTEGERS: 200 Shipment includes Software and User Manual	educational Price: 395.00 $ CONSTRAINTS: 4000 VARIABLES: 8000 INTEGERS: 800 Shipment includes Software and User Manual
Industrial LINDO API	**Extended LINDO API**
educational Price: 695.00 $ CONSTRAINTS: 16000 VARIABLES: 32000 INTEGERS: 3200 Shipment includes Software and User Manual	educational Price: 995.00 $ CONSTRAINTS: unlimited VARIABLES: unlimited INTEGERS: unlimited Shipment includes Software and User Manual

Általános licence: (395 - 3995 $ között)

LINDO API 4.1 - standard

Single user standard licenses are appropriate for individual, business and government use.

Super LINDO API	Hyper LINDO API
standard Price: 395.00 $ CONSTRAINTS: 1000 VARIABLES: 2000 INTEGERS: 200 Shipment includes Software and User Manual	standard Price: 795.00 $ CONSTRAINTS: 4000 VARIABLES: 8000 INTEGERS: 800 Shipment includes Software and User Manual
Industrial LINDO API	**Extended LINDO API**
standard Price: 2395.00 $ CONSTRAINTS: 16000 VARIABLES: 32000 INTEGERS: 3200 Shipment includes Software and User Manual	standard Price: 3995.00 $ CONSTRAINTS: unlimited VARIABLES: unlimited INTEGERS: unlimited Shipment includes Software and User Manual

A LINGO program, amit nálunk az oktatás során használunk, az 10.0-ás verziószámnál jár, és a LindoApi-hoz hasonló ártáblázatában az oktatói licence 245 és 1195 dollár között mozog, míg az általános licence 495 és 4995 dollár között van.

A LindoApi-hoz letöltés során még egy igen részletes és mindenre kiterjedő 477 oldalas leírás is jár, illetve példaprogramok a legismertebb nyelveken mint: C/C++, C#, Delphi, .NET, Fortran 90, Java/J++, Visual Basic. Ezek elérhetőek az ingyenes tanulói licensszel is.

3.2 CPLEX:

A CPLEX a francia ILOG cég terméke, ez is kereskedelmi szoftver. Az ILOG 20 éves vállalat, egyik fő profilja az operációkutatási szoftverek fejlesztése. A CPLEX szoftvernek is van oktatói és általános licence. Az oktatói licencek 750 és 6160 dollár között mozognak, itt nem teljesítménybeli megkötések vannak, hanem a licence időtartamában térnek el, eltérnek még abban, hogy hány ember használhatja a programot, hogy hányszor lehet a licencet más platformra „átköltöztetni", illetve hogy automatikusan adják-e hozzá a különböző fejlesztéseket. Az időtartam egy és három év között van, a felhasználók száma egy és tíz között van, a platform költözések száma vagy három vagy végtelen, és automatikus fejlesztés vagy megengedett vagy nem. A már meglévő licencek is felfrissíthetők a legújabb programcsomagra, ennek az ára a licence függvényében változó 250 és 2160 dollár közötti összegbe kerül.

Szerencsére a CPLEX-nek is van ingyenes verziója a *CPLEX 300*. A *CPLEX 300* az elviekben ugyanazt tudja mint a CPLEX, annyi különbséggel hogy csak olyan probléma oldható meg vele aminek maximum 300 változója, és maximum 300 megkötése van. A CPLEX 300 egyáltalán nincs dokumentálva, nem találtam hozzá semmi kapaszkodót, segítséget. A „CPLEX 300" kifejezésre rákeresve az ILOG oldalán nulla találatot eredményez. A google kereső is csak 13 találattal segíti a keresést, de mindegyik valamilyen

probléma megoldása, egy output, amibe bele van írva, hogy ezt a megoldást a *CPLEX 300* solver adta. A *CPLEX 300*-at egy DLL állományban tudtam letölteni (cplex9s.dll) és a Maximal Software vállalat *MPL for Windows 4.2* programjával tudtam futtatni. Vagyis a *CPLEX 300*-at csak mint úgynevezett „fekete-dobozt" tudtam használni, egy program segítségével beadtam neki egy problémát, ő pedig a kimenetben adott egy választ, ami a feladattól függően egy megoldás vagy egy hibaüzenet volt.

3.3 CoinMP:

A CoinMP solverről igen kevés információ található meg, de ez egy alapvetően ingyenes programcsomag Windows platform alá. A CoinMP tulajdonképpen egy DLL (dynamic linked library), amiben az alapvető operációkutatási problémákat megoldó függvények vannak összegyüjtve, és ezeket egy megfelelő interfészen keresztül lehet használatba venni. A CoinMP-ről szóló általános információk között le van írva hogyan lehet letölteni magát a solvert, hogyan lehet használatba venni, és lehet találni C++ nyelven írt forráskódokat amikben bennevan hogy hogyan lehet használni a CoinMP-t.

A CoinMP licencében le van írva hogy ez egy ingyenes szoftver, akár kereskedelmi célból is felhasználható, viszont ez esetben a készítő vállalja az esetleges hibákat, károkat, amiket a CoinMP okoz, hogy ne lehessen az eredeti készítőkre hárítani a felelősséget. Ezen kívül egyébként elég szabad a licence, szinte mindent enged, nem is igazából jogi megkötések vannak benne, csak felhívják a figyelmet az illedelmes magatartásra, a normális elvárható emberi viselkedésre.

A CoinMP solverjét is tudja kezelni az *MPL for Windows 4.2* program, így kutatásom során ezen keresztül használtam. Nem találtam semmilyen teljesítménybeli megkötést a CoinMP solverre, de az *MPL for Windows 4.2* programban is benne van a maximum 300 változó és

maximum 300 megkötés ellenőrzése, így ennél nagyobb feladatokat a programon keresztül nem tudtam a solverrel megoldani.

3.4 GLPK (Gnu Linear Programming Kit):

A GLPK a GNU szoftvercsalád része, vagyis rá is a GNU General Public License (GNU GPL) érvényes, aminek a rövid lényege az, hogy a szoftver ingyenes, nyílt forráskódú, és a forráskódokból az eredeti kommenteket nem szabad eltávolítani, illetve hogy a szoftver akár kereskedelmi célú szoftverekhez is használható. A GLPK solverhez dokumentáció a GNU MathProg dokumentációjában található. A GLPK-nak van két különböző levelező listája is, egy az általános dolgok megvitatására (help-glpk@gnu.org) a másik pedig a hibák közlésére és javítására (bug-glpk@gnu.org). A GLPK fejlesztésébe akár be is lehet szállni, be is lehet segíteni. A GLPK is futtatható DLL formátumban az *MPL for Windows 4.2* program segítségével, kutatásom során ezt használom.

3.5 LPSolve:

Az LPSolve solvert Michel Berkelaar kezdte fejleszteni az eindhoveni egyetemen. A solver ingyenesen használható. A 3.0-ás verzió után LGPL licence alá került a fejlesztés. Az LGPL a GNU Lesser General Public License rövidítése, ami teljesen hasonló a GLPK solvernél leírt használati feltételekhez, a legnagyobb lényegi kikötés az az, hogy ezt a solvert nem lehet kereskedelmi célokra felhasználni. Az LPSolve most az 5.5.0.10-es verziószámnál tart. Nemcsak a solvert lehet megszerezni, de letölthető a teljes forráskód, példák, és kézikönyvek is hozzá.

Az LPSolve ANSI C alatt készült, többféle platformra is fordítható. Az LPSolve DLL-ként is hívható, így használható C/C++, Visual Basic, .NET, Excel vagy akár Java környezetben is.

Különféle matematikai programokkal is tud kommunikálni, az LPSolve solvert meg lehet hívni többek között AMPL, MATLAB, O-Matrix, Scilab vagy Octane programokból. Az LPSolve is futtatható DLL formátumban az *MPL for Windows 4.2* program segítségével, kutatásom során ezt használom.

Az LPSolve saját leírása szerint nincs limitálva a megoldandó probléma nagysága, de a nagyobb és bonyolultabb problémák komoly fejtörést okozhatnak neki, és az ilyenek megoldásában felsülhet. Persze ez után még gyorsan hozzáteszik, hogy a kereskedelmi szoftverek is ugyanígy járhatnak. Az LPSolve fejlődését, legújabb fejlesztéseit a saját közösségi oldalunkon lehet nyomon követni, ami a Yahoo Groups-on található az alábbi címen: http://groups.yahoo.com/group/lp_solve. Ez a community elég aktívnak tűnik, rengeteg taggal (több mint 6500 tag) és üzenettel.

3.6 LGO:

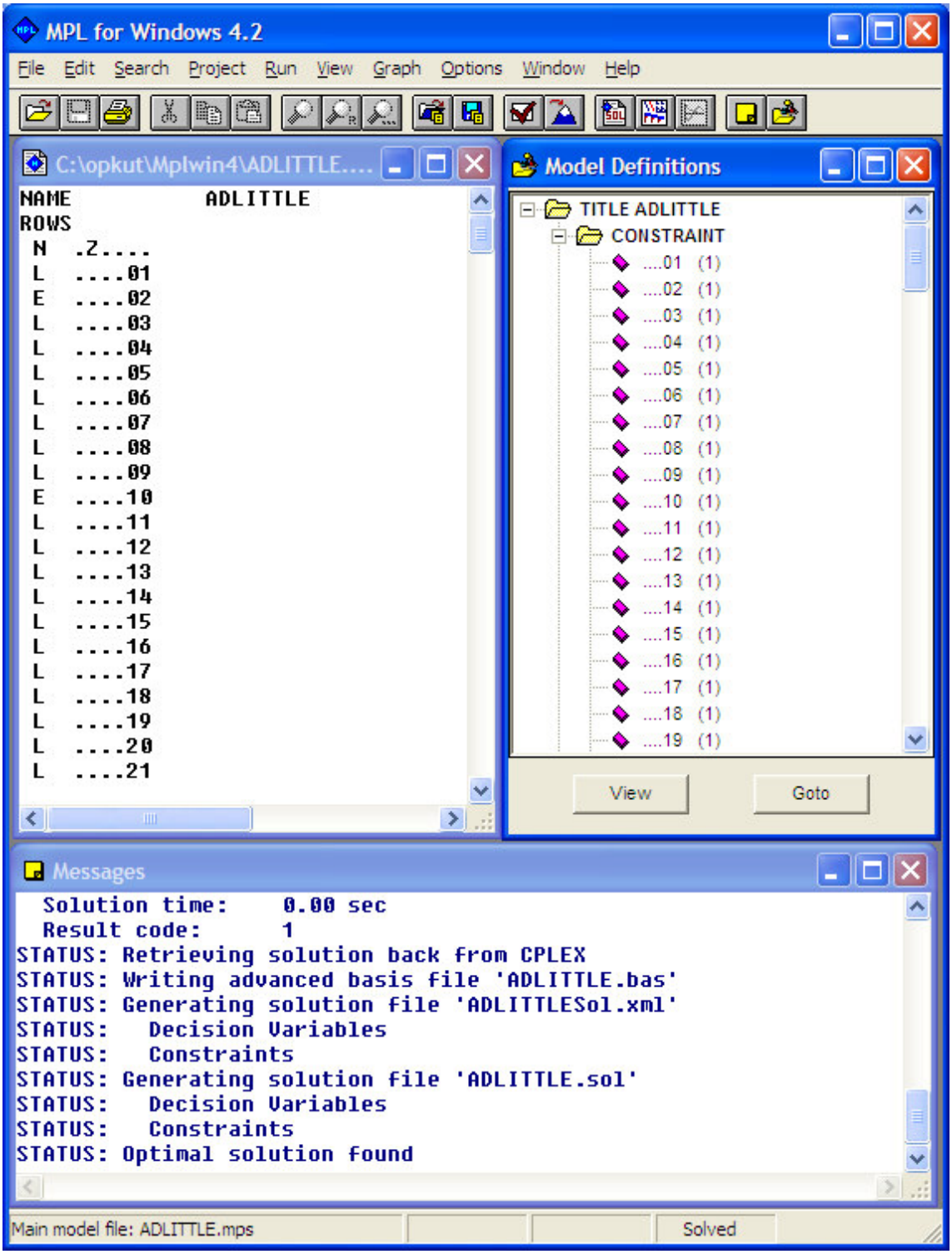
Az LGO a kanadai Pintér Consulting Services vállalat programja, aminek vezetője az ELTE-t végzett magyar származású János D. Pintér. Az LGO szoftvert több mint 20 ország professzorai, diákjai, és vállalatai használják, jelenleg az LGO elérhető C illetve Fortran nyelven, illetve használható többek közt Excel, GAMS, MPL, Maple és Mathematica szoftverekkel. Az LGO kereskedelmi szoftver. Pontos árlistát nem nagyon lehet találni hozzá, csak a licencében utalást hogy létezik speciális licence az oktatási intézményeknek és a kutatási szervezeteknek, és létezik általános licence. A speciális licence az általános licenchez képest szerényebb összegbe kerül. A licence ára függ attól, hogy milyen célra akarják használni, mekkora méretű problémák megoldására akarják használni, és hogy milyen platformokon kívánják használni. Nekem a személyes benyomásom az, hogy az ár alku tárgya, mert nagyságrendi kiinduló árakkal sem találkoztam. Az LGO is futtatható DLL formátumban az *MPL for Windows 4.2* program segítségével, kutatásom során ezt használom.

3.7 Conopt:

A Conopt a dániai ARKI Consulting & Development A/S vállalat programja. A solvert önmagában nem árulja, hanem azt ajánlja, hogy valamilyen modellező rendszerbe integrálva használjuk a solvert. A solver Windows platformon DLL formában létezik, Unix platformon pedig úgynevezett „Shared Library" vagy „Shared Object" formában. A solver licenceléséről, illetve megvásárlásáról az oldalukon szó sem esik, ezek tisztázását teljesen átruházzák a modellező rendszert készítő vállalatnak. A Conopt jelenlegi legfrissebb változata a Conopt 3.10-es. Ehhez a verzióhoz nem fértem hozzá, viszont a Conopt1 és a Conopt2 solverjét felhasználtam a kutatásomban.

A vállalat a saját weboldalán több dologra is felhívja a figyelmet, amit a programjuk vagy egyáltalán nem tud, vagy komoly problémát jelent a számára. Felhívják arra is a figyelmet, hogy egyáltalán nem tudják garantálni, hogy a program jó eredményt produkál, ennek eldöntését a felhasználóra bízzák. Saját állításuk szerint egyébként a Conopt már 25 éve folyamatos fejlesztés alatt áll. Magát a solvert nagy modellek megoldására fejlesztették, állításuk szerint tízezer változó feletti feladatokat még rutinosan old meg, de egymillió változó feletti feladatokat is oldott már meg a Conopt. A maximális feladatméretnek vannak korlátai, de az több dologtól is függ. A Conopt által ajánlott modellező rendszerek közt megtalálható az *MPL modeling system*, így a Conopt és Conopt2 solvereket DLL formátumban használni tudom az *MPL for Windows 4.2* program segítségével, kutatásom során ezt használom. A Conopt2 hallgatói licence nemcsak változó és megkötés szerint van limitálva (300-300) de azt is kiköti, hogy maximum 2000 darab nemnulla érték, és maximum 1000 darab nemlineáris nemnulla érték lehet egy feladatban. A Conopt licencében még nem szerepelnek a nemnulla értékekre való megkötések.

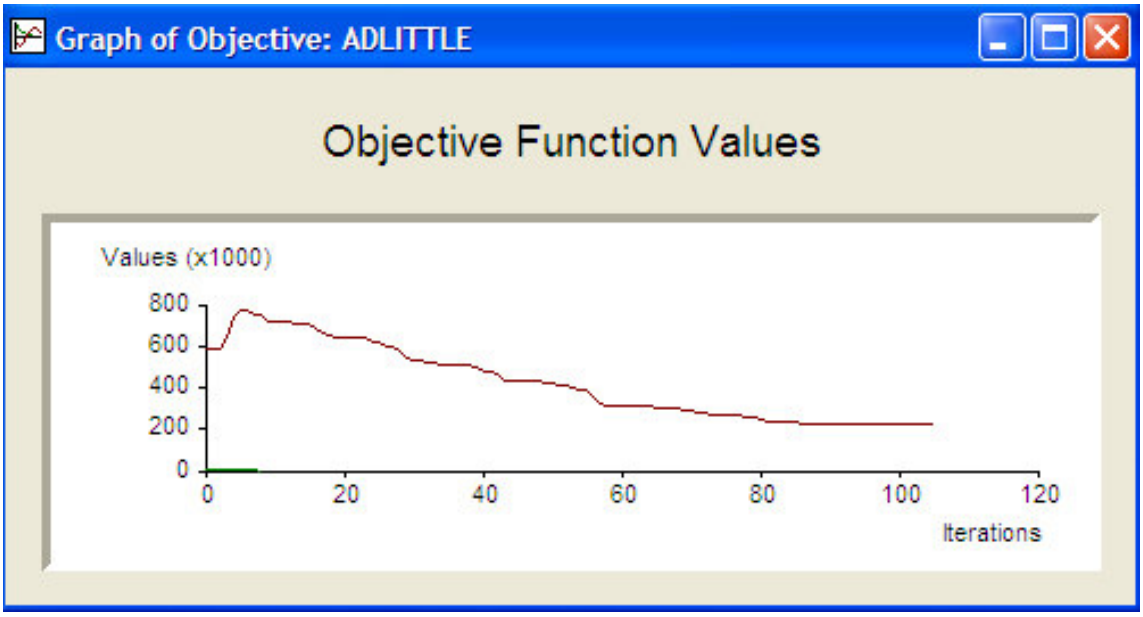
4. MPL for Windows 4.2



Az MPL for Windows 4.2 az amerikai Maximal Software vállalat programja. A programban nincs solver, nem tud megoldani operációkutatási feladatokat, viszont egy nagyon kellemes interfészt nyújt a végfelhasználók és az operációkutatási solverek DLL-jei közt. A program be tud olvasni operációkutatási feladatokat több különböző formátumban és értelmezi is a feladatokat. Az ismert fájformátumok: MPS, MPL, DAT, IDX, MPJ és SPM. A program képes szintaktikai ellenőrzésre, képes megjeleníteni a feladat statisztikáit.

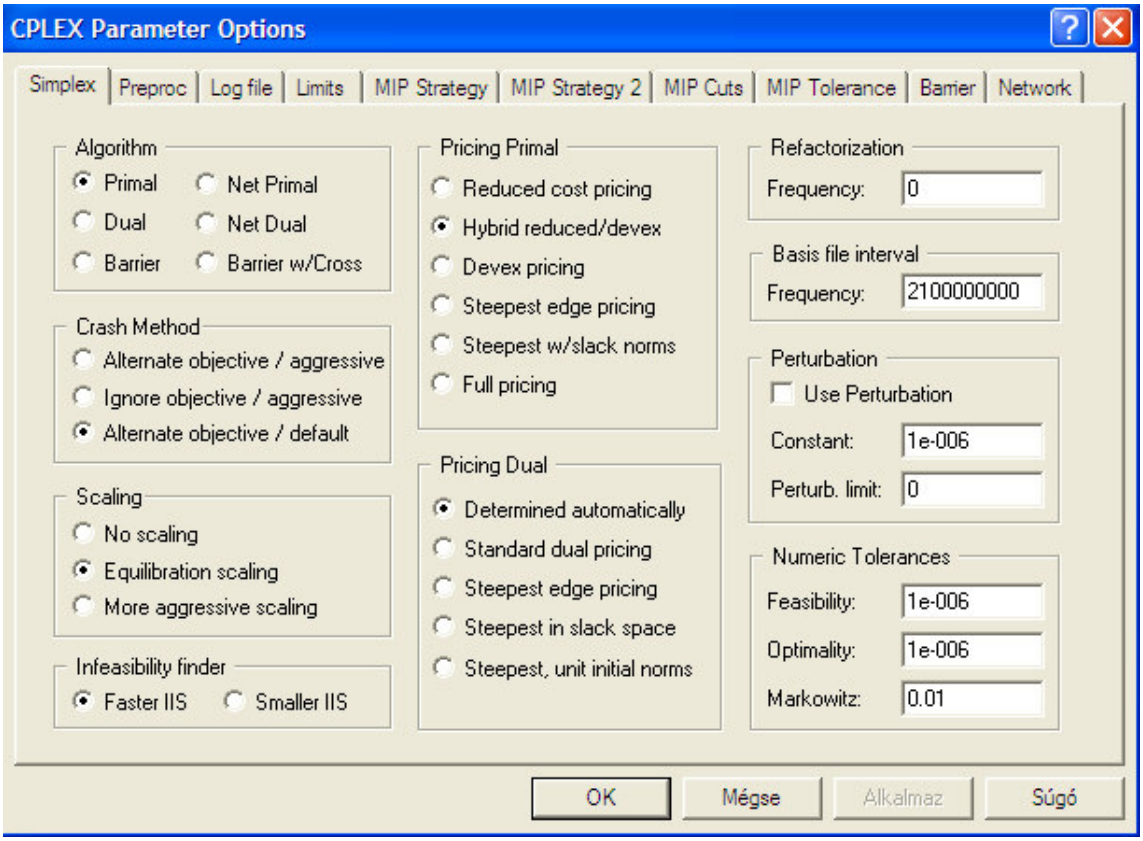
A program segítségével paraméterezhetőek az egyes solverek. A feladatok megoldásáról kaphatunk szöveges értesítést, illetve XML-ben is ad kimenetet. A program méri a feladat megoldásának idejét ezredmásodpercben is, így tisztában lehetünk pontosan a számolás idejével. A kezdőképernyőn a megnyitott feladatoknak van ablaka, az aktuális modell definiálásának van ablaka, és egy „Messages” ablak található még ahol szövegesen ír a program a végfelhasználónak adatokat a program futásáról.

Grafikont tud készíteni a feladat mátrixáról, illetve a magáról a megoldás menetéről is. A megoldás grafikonján az egyik koordinátatengely az iterációk számát mutatja, a másik pedig az abban az iterációban a célfüggvény értékét. Itt egy példa egy minimalizálási feladat megoldás-grafikonjáról.



A program által használható solverek listáját az *Options/Solver Menu…* menüpont alatt lehet megtekinteni illetve módosítani. A program indításakor végigpásztázza a saját mappáját, és keresi az általa ismert solverek DLL-jeit, és így építi fel a *Solver Menu*-t. Ha a program futása során változnak az elérhető solverek, akkor a *Solver Menu Scan* gombjával tudjuk újrageneráltatni a listát.

Magán az *MPL for Windows 4.2* programon is rengeteg dolgot lehet beállítani, de ami számunkra lényeges, hogy a solvereket is képesek vagyunk paraméterezni. Tíz különböző fülön a paraméterek elég széles skáláját tudjuk megszabni, természetesen ilyenkor azért kérdéses hogy a választott solver értelmezi-e a kért paramétert vagy sem. A solverek felparaméterezését ellenőrizhetjük az *Options/Solver Option Lists…* menüpont alatt, ahol egyrészt ellenőrizhetjük, hogy az adott solver milyen paramétereket használ, illetve hogy a kért beállítások rendben megtörténtek-e.



Az *MPL for Windows 4.2* program ingyenes tanulói licensszel történő megszerzése és telepítése nem bonyolult, lényegében elég követni a leírt lépéseket a maximal-usa.com weblapon, a solverek beszerzése annál bonyolultabb lehet.

Első lépésként szerezni kell egy aktivációs kulcsot a tanulói licenchez, ehhez csak egy formot kell megadni néhány személyes adattal az alábbi oldalon:
http://www.maximal-usa.com/form/reqact.html
Ennek a kitöltése után küldenek nekünk egy e-mail az általunk megadott e-mail címre, ami tartalmazza az általános tudnivalókat, és egy többsoros aktivációs kulcsot, amivel majd aktiválni tudjuk a programunkat. A tanulói licence ingyenes, és a megoldható feladatok mérete van lekorlátozva. Tanulói licence segítségével csak olyan operációkutatási feladatot vagyunk képesek megoldani, amiben maximum 300 változó és maximum 300 megkötés szerepel.

Miután elküldtük az igényünket a licencére, második lépésként le kell töltenünk a program installját. A Maximal Software oldalán az installok elérhetőek EXE és ZIP formában is, az EXE állományokat FTP protokollon, a ZIP állományokat http protokollon keresztül lehet letölteni.

Letöltés után a harmadik lépés a programunk installálása. Ha ZIP-ben töltöttük le, akkor előtte ki is kell csomagolni. Aki nem boldogulna elsőre a kicsomagolással, a honlapon található link a pkunzip és winzip alkalmazásokra is. A telepítés végtelenül egyszerű, ezzel különösebben foglalkozni nem kell.

Utolsó lépésként pedig a már feltelepített programunkat aktiválnunk kell a kapott licencünkkel, ami ekkorra már biztosan megérkezik a postafiókunkba. A program megszerzése és telepítése nem okoz bonyodalmat, következő lépés a program által felismert solverek beszerzése és használatba vétele.

Ezek után maga az *MPL for Windows 4.2* program megvan, ezután már csak a választott solvereket kell összegyűjteni. Mivel a Maximal Software oldalán nem találtam a programmal működő solvereket, ezért a solverek beszerzésére kétféle ötletem volt. Az egyik hogy a solverek fejlesztőinek az oldalain megpróbálom megkeresni a nekem kellő DLL-eket, ezt 1-2 oldalon biztosan meg is találtam volna, mert igen könnyen megtalálható, de voltak teljesen reménytelen oldalak is, amiknél esélyt nem láttam, hogy meg fogom találni ami nekem kell, ezért ezt az ötletemet eltettem talonba.

A másik ötlet az volt, hogy rákeresek a google segítségével magukra a DLL-ekre, és ha megtalálom, akkor megpróbálom letölteni őket. Ez az ötlet már életképesebbnek tűnt, leszámítva azt az apróságot, hogy nem tudtam, hogy a solverek DLL-jeit pontosan milyen néven is keressem. De maga az ötlet tetszett, ezért továbbfejlesztettem, és magára az MPL for Windows programra kerestem rá, azt feltételezve, hogy aki használta, és archiválta az interneten, annak solverei is voltak a programhoz, különben mivel is használta volna.

Ez az ötlet fényesen bevált, a google keresőn rákeresve például az „**mplwin42**” kulcsszóra összesen 4 találat van, ebből kettő a Maximal Software oldalára mutat, az egyik link egyáltalán nem releváns, de a négyből egy link egy MPL program archiválására mutat.
http://www.enq.ufrgs.br/cursos/pos/ProcInt/Otimizacao/Linguagens/MPL/

Ilyen módon igen hamar összeszedtem jó pár DLL-t, bemásoltam az installált és licencelt *MPL for Windows 4.2* programom mappájába és örömmel tapasztaltam, hogy a program felismert jó pár solvert, és most már a segítségükkel könnyedén meg tudtam oldani lineáris programozási feladatokat.

Az *Options/Solver Menu…* alatt meg lehet tekinteni a felismert solverek részleteit, így pontosan meg lehet állapítani, hogy melyik solverhez melyik DLL tartozik. A részleteknél még meg lehet adni, hogy az adott solver milyen néven szerepeljen a program menüjében,

illetve meg lehet változtatni az elérési útját, a solver típusát, és meg lehet adni a solverhez licencekulcsot is.

Az általam keresett solverek, és a hozzájuk tartozó DLL.

CPLEX 300	**cplex9s.dll**
CoinMP	**coinmp.dll**
GLPK	**glpk413.dll**
LPSolve	**lpsolve55.dll**
LGO	**lgostud.dll**
Conopt	**conopt3s.dll**
Conopt2	**conopt.dll**

5. Saját fejlesztésű LindoApi program fejlesztése

A saját fejlesztésű LindoApit használó programom két teljesen különálló részre bomlik. Az egyik része kapcsolatot teremt a LindoApival, átadja neki a lineáris programozási feladatot és az esetleges egyéb paramétereket, majd a LindoApi válaszát, ami optimális esetben a feladat megoldása standard outputra küldi és elmenti egy állományba. Ez a rész méri a feladat megoldásához szükséges időt is. Ez a programrész akár egy egyszerű command ablakban is képes futni, csak meg kell hívni a programot és paraméterként átadni neki a lineáris programozási feladatot tartalmazó állomány címét, így nem kell ennek a futtatásához semmilyen különleges környezet.

A második rész egy grafikus felület, ahol szabályos Windows formok segítségével adhatjuk meg a problémát. A lemezről kitallózhatjuk a megoldani kívánt MPS állományt, külön panelen beállíthatjuk a kívánt paramétereket, és így ezen a módon jóval kényelmesebben tudjuk megoldatni a feladatokat.

A programok elkészítéséhez Microsoft Visual C++ környezetet választottam.

5.1 A számításokat elvégző program

Ez a programrész beolvassa a paraméterként megkapott MPS állományt és az egyéb megadott paramétereket, felépíti a kapcsolatot a LindoApival, megoldja a feladatot, és leméri a megoldáshoz szükséges időt. A LindoApihoz való csatlakozáshoz szükségünk van egy licencre. Ingyenes licence a teszteléshez rendelkezésünkre áll a *Licence* könyvtárban egy *lndapi40.lic* állomány formájában.

Példa a LindoApi licencre:

```
LINDO API
Demo
4.00
1
None
Nonlinear  Global  Barrier
Educational
All platforms
Demo License 4.0
>
RKsf-A*#J-vec6-HurL-XNdd-hEXW-6rn@-SVr6-i3iB-Gi6L-
*rwR-6PCU-w59m-V59q-M6J2-kt7Y-uLPM-vuQ7-?4Ce-FE#T-
pWZW-qA?G-J@5%-D9Qf-K2TA-wz%q-s@J7-h6$@-$CMb-MdCG-
rsDk-7345-vS98-s4S*-6Z9f-n2YF-z*A8-?NAc-DZBm-ud8d-
qR8s-oDf7-vY86
>
```

Ahhoz hogy használni lehessen a LindoApit és a függvényeit C nyelv segítségével, első lépésként bele kell venni a programunkba a *header* állományát.
#include "lindo.h"

Ezek után mielőtt még a feladatmegoldásba fognánk, be kell olvasni a licencet, hogy legyenek jogosultságaink.
nErrorCode = LSloadLicenseString("../../../license/lndapi40.lic",MY_LICENSE_KEY);

Ehhez már beépített LindoApi függvényt használunk, ahol meg kell adni a licence-állományunk helyét, és egy üres karaktertömböt, amiben eltárolódik a licence.

Következő lépésként létre kell hoznunk egy modellt a feladatunkhoz.
pModel = LScreateModel (pEnv, &nErrorCode);

Ezek után már van licencünk, van egy modellünk, ezek után a feladatot kell megadnunk. Az én programom a feladatokat MPS formátumból olvassa be.
nErrorCode = LSreadMPSFile(pModel,mpsfile,LS_FORMATTED_MPS);

A programban az *nErrorCode* változó segítségével le lehet ellenőrizni, hogy valóban sikerült-e a feladat beolvasása, így ha nem sikerült volna, akkor még megpróbálja a program *Lindo* illetve *MPI* formátumban beolvasni a kapott állományból a megoldásra váró feladatot.
nErrorCode = LSreadLINDOFile(pModel,mpsfile);
nErrorCode = LSreadMPIFile(pModel,mpsfile);

Miután a feladat beolvasása sikeresen megtörtént, a feladatról több fontos adatot is le lehet kérni az *LSgetInfo* függvény segítségével.

Változók száma: *LSgetInfo(pModel,LS_IINFO_NUM_VARS,&n);*
Megkötések száma: *LSgetInfo(pModel,LS_IINFO_NUM_CONS,&m);*

A *LindoApi* működését rengeteg paraméterrel meg lehet változtatni.
- 31 különböző általános solver paraméterek
- 27 különböző nemlineáris optimalizálási paraméterek
- 59 különböző kevert egészértékű optimalizálási paraméterek
- 26 különböző általános optimalizálási paraméterek
- 15 különböző licence információs paraméterek
- 12 különböző modell-analizáló paraméterek

A programban a paraméterek értékeit lekérhetjük, és meg is változtathatjuk, illetve megadhatunk egy alapállapotot is. Paraméter lekéréséhez az *LSset..()* kezdetű függvények használhatók, például a *LSgetModelIntParameter()* vagy a *LSgetModelDouParameter()*.

Paraméterek beállításához, megváltoztatásához az *LSget…()* kezdetű függvények használhatók, például a *LSsetModelIntParameter()* vagy a *LSsetModelDouParameter()*.
LSsetModelIntParameter(pModel,LS LS_IPARAM_SPLEX_ITRLMT,-1);
Így lehet megadni a szimplex módszer alkalmazása során a maximális iterációk számát. -1 érték esetén nincs limit az iterációk számára vonatkozóan. Nem minden paraméter értéke változtatható meg, van olyan paraméter, aminek csak olvasni lehet az értékét. Ilyen paraméter például az LP_IPARAM_SOLVER_ITER amely paraméternek a megoldás után lesz értéke, és a megoldás során alkalmazott iterációk számát tartalmazza.

A LindoApi háromféle megoldó-rutint ajánl fel a feladat megoldására.

- *LSsolveMIP()* – olyan feladatok megoldására szolgál, amely egy vagy több egészértékű változót tartalmaz.

- *LSoptimize()* – olyan operációkutatási feladatok megoldására szolgál, amelyben minden változó folyamatos értékű.

- *LSsolveGOP()* – nemlineáris modellek megoldására szolgál.

Ezek a függvények visszatérési értéke egész érték, sikeres futás esetén nullával térnek vissza, egyéb esetben előre definiált hibakódokkal. Viszont a megoldásnak még lehetnek különböző státuszai is, ezért ezekbe a függvényekbe át lehet adni egy *status* változó címét, hogy abba eltárolja a függvény a megoldás státuszát.
nErrorCode = LSoptimize(pModel, LS_METHOD_FREE, &status);

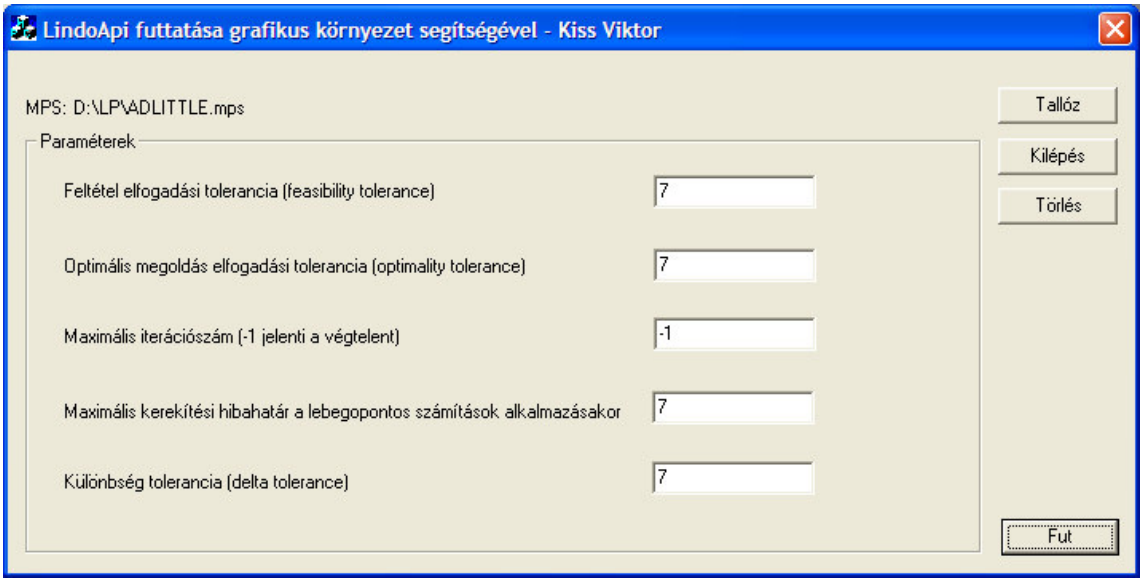
A *status* változóba is egy egész érték íródik, az itt kapható válaszok is definiálva vannak a LindoApi dokumentációban a *Common Parameter Macro Definitions* című fejezet alatt. Összesen 14 féle különböző státuszérték van definiálva.

A megoldott feladatokról mindenféle adatokat az *LSget…()* függvények segítségével kaphatunk. A feladatokat amikre már nincs szükségünk, lehetőségünk van felszabadítani a memóriából, erre a célra szolgálnak az *LSdeleteModel()* és az *LSdeleteEnv()* függvények.

A feladatok végrehajtásához szükséges időt Visual C++ környezet alatt a *GetTickCount()* függvény segítségével oldottam meg. Ez visszaadja a számítógép bekapcsolása eltelt időt ezredmásodpercben, így 2 időpillanat különbségét tekintve ezredmásodperc pontossággal megmondható hogy az adott kódrészlet mennyi ideig futott. Ennek segítségével mérem a feladat megoldásának időtartamát a LindoApit használó programomban.

A saját fejlesztésű alkalmazásom Visual C++ környezetben készült, terminal ablakban futtatható, és a program paraméterezhető. Meghívásnál a megoldani kívánt MPS állomány címét kötelező megadni, a többi paraméter opcionális.

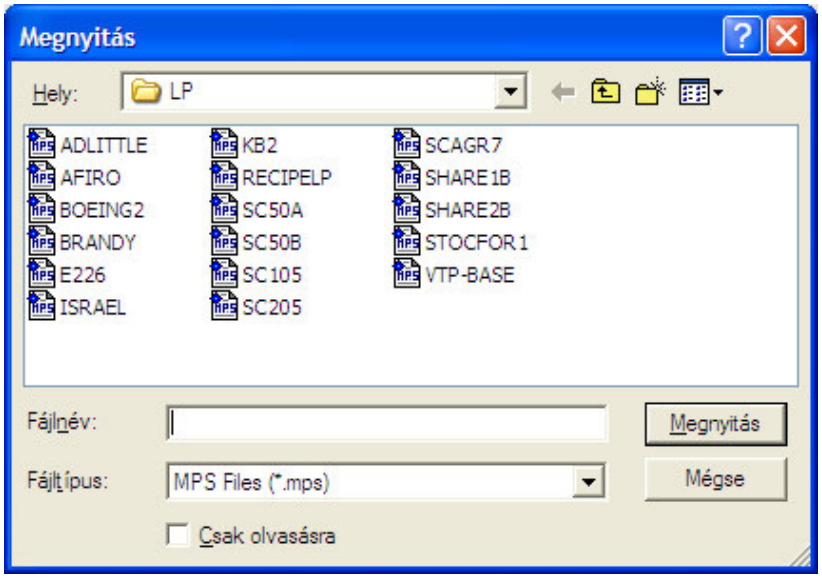
5.2 A saját fejlesztésű LindoApi programhoz készített grafikus felület



Ez a program egy grafikus felület a számításokat végző programhoz. Használatával ki lehet tallózni a feladatot tartalmazó MPS állományt, be lehet állítani különböző paramétereket, és ezekkel a paraméterekkel meg lehet hívni a számolást elvégző programot.

A program *Microsoft Visual C++* környezetben készült *MFC AppWizard* projektként. Ebben a környezetben általános Windows objektumokat lehet használni, és így egy könnyedén használható programot készíteni.

A megoldásra váró feladatot a *Tallóz* gomb segítségével lehet kiválasztani. A *Tallóz* gomb a *CFileDialog* osztály egy példányát hozza létre, amely az alábbi *Megnyitás* ablakot hozza fel:



Ebben az ablakban a számítógép által elérhető állományok közt lehet tallózni és csak MPS formátumú állományt lehet kiválasztani. A grafikus felületen a *Fut* gombot kezdetben nem lehet lenyomni, de ha a *Megnyitás* ablak szabályosan visszatér, vagyis kiválasztott a felhasználó egy MPS kiterjesztésű állományt, akkor a *Fut* gomb lenyomhatóvá válik, és a gomb megnyomása fogja meghívni a feladatot megoldó programot a grafikus felületen beállított paraméterekkel.

A grafikus felületen 5 paramétert lehet beállítani, azért ezt az öt paramétert választottam, mert véleményem szerint ezek a legfontosabbak, ezekkel lehet leginkább a solver működését megváltoztatni. A választott paraméterek a következők:

- *Feltétel elfogadási tolerancia (feasibility tolerance)*

Ennek megfelelője a LindoApi solverben az *LS_DPARAM_SOLVER_FEASTOL*. A solver felfogása szerint a megoldás során egy megkötést elértnek tekint, ha a megkötés értéke,

vagyis a megadott alsó vagy felső korlát, és a függvény értéke közötti különbség kisebb ennél a változónál. Ennek a változónak az alapértéke 10^{-7}, a programban a kitevőt lehet változtatni, a kitevő értékének 1 és 14 közé kell esnie.

- *Optimális megoldás elfogadási tolerancia (optimality tolerance)*

Ennek megfelelője a LindoApi solverben az *LS_DPARAM_SOLVER_OPTTOL*. Más néven ezt duális elfogadási toleranciának is nevezhető (dual feasibility tolerance). Ez hasonló a feltétel elfogadási toleranciához, de itt a két érték, aminek a különbségét nézzük az a célfüggvény értéke és a duális feladat célfüggvényének értéke. Ennek a változónak az alapértéke 10^{-7}, a programban a kitevőt lehet változtatni, a kitevő értékének 1 és 14 közé kell esnie.

- *Maximális iterációszám*

Ennek megfelelője a LindoApi solverben az *LS_IPARAM_SPLEX_ITRLMT*. Ezzel lehet meghatározni, hogy a szimplex módszer során maximálisan hány iterációval próbálkozzon a solver. Ennek az értéke pozitív egész szám, illetve a -1 érték jelenti azt, hogy nincs felső korlát, végtelen iterációs lépés lehetséges.

- *Maximális kerekítési hibahatár a lebegőpontos számítások alkalmazásakor*

Ennek megfelelője a LindoApi solverben az *LS_DPARAM_GOP_FLTTOL*. Ennek nem is kell különösebb magyarázat. A számítógépeknél a véges értékkészlet miatt a lebegőpontos számítások során pontatlanságok csúszhatnak be. Minél kevésbé optimalizáltak ezek a számítások, és minél több van belőlük, a hiba mértéke úgy nőhet. Ezt a lehetséges hibát lehet maximalizálni ezzel a változóval. Ennek a változónak az alapértéke 10^{-10}, a programban a kitevőt lehet változtatni, a kitevő értékének 1 és 20 közé kell esnie.

- *Különbség tolerancia (delta tolerance)*

Ennek megfelelője a LindoApi solverben az *LS_DPARAM_GOP_DELTATOL*. A kerekítési pontatlanságok miatt egy egyenlőség bal és jobb oldala között különbség alakulhat ki, és ez speciális esetekben a megoldáshalmaz konvexifikálásához vezethet. Amíg ez *Különbség tolerancián* belül marad, addig ez nem következik be. Ennek a változónak az alapértéke 10^{-7}, a programban a kitevőt lehet változtatni, a kitevő értékének 1 és 14 közé kell esnie.

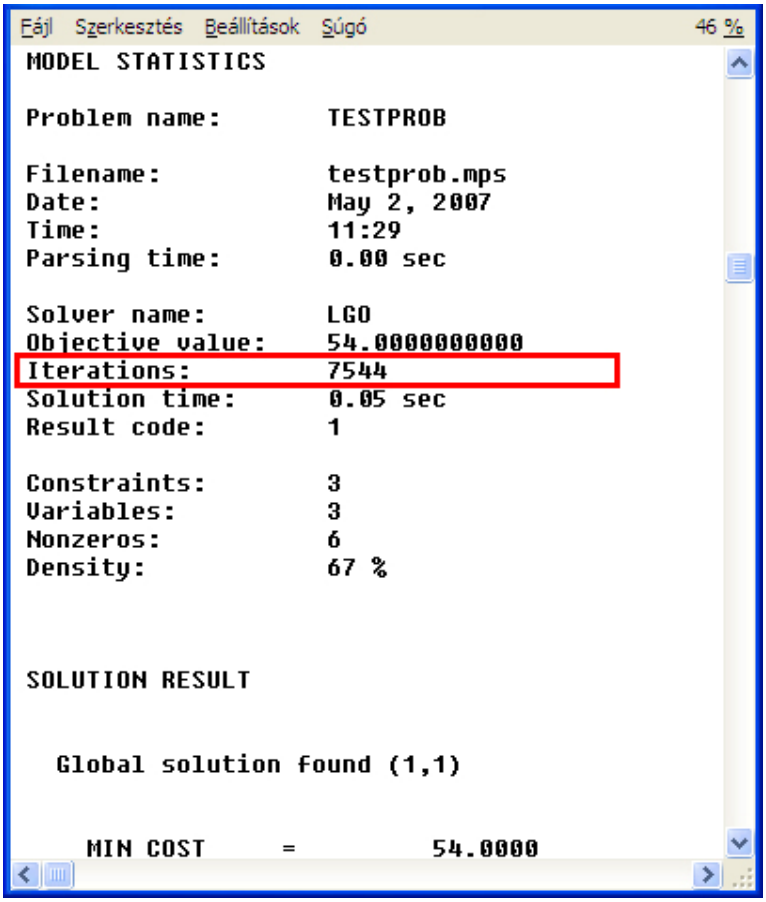
6. Feladatok, mérések, eredmények

A solverek tesztelésére a Netlib oldaláról gyűjtöttem össze híres példafeladatokat, amelyeknek ismert a megoldásuk.

A kiválasztott feladatok:

Feladat neve	Megkötések	Változók	Nemnulla értékek	Ranges	Bounds	Méret
afiro	27	32	83	nincs	nincs	3 528
sc50b	48	48	118	nincs	nincs	4 999
sc50a	49	48	130	nincs	nincs	5 648
kb2	43	41	286	nincs	van	10 875
sc105	104	103	280	nincs	nincs	12 018
adlittle	56	97	383	nincs	nincs	17 766
stocfor1	117	111	447	nincs	nincs	17 802
scagr7	129	140	420	nincs	nincs	22 108
sc205	204	203	551	nincs	nincs	23 539
share2b	96	79	694	nincs	nincs	26 545
recipelp	91	180	663	nincs	van	31 804
vtp-base	198	203	908	nincs	van	42 070
share1b	117	225	1151	nincs	nincs	44 642
boeing2	140	143	1196	van	van	51 514
brandy	182	249	2148	nincs	nincs	75 446
israel	174	142	2269	nincs	nincs	86 680
e226	223	282	2578	nincs	nincs	98 921

Az LGO solverrel ezeket a feladatokat nem lehetett megoldani, mert a hallgatói licence nem engedi. Nagyon le van korlátozva, mindössze öt(!!!) változót és öt(!!!) megkötést engedélyez. Azért hogy mégis ki tudjam próbálni, az MPS leírásnál található példát oldottam meg az LGO solverével. A megoldás során azt tapasztaltam, hogy az optimális megoldást mindegyik solver természetesen tökéletesen megtalálta, de amíg mindegyik solver maximum hat iterációs lépésen belül befejezte a feladat megoldását, addig az LGO solvere 7544 iterációs lépést végzett el. A megoldások kiszámolásának idejében különösebb eltérések nem voltak.



A többi solverrel a kiválasztott feladatok nagy részét meg tudtam oldani, így ezeket a megoldásokat hasonlítom össze, és ezekből vonok le következtetéseket. A kiválasztott hét solver közül öt mind a tizenhét feladatra adott választ, a Conopt2 a kibővített licence-megszorításai miatt három feladatra nem adott megoldást, a LindoApi pedig hat feladatot nem oldott meg ugyancsak tanulói licence miatt. Programhiba, vagy számítási hiba miatt egyik solver sem állt le egyik feladatnál sem.

Feladat / Solver	afiro	sc50b	sc50a	kb2	sc105	adlittle	stocfor1	scagr7	sc205	share2b	recipelp	vtp-base	share1b	boeing2	brandy	israel	e226	átlag
CoinMP megoldás	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	
CoinMP idő	0,015	0,015	0,031	0,015	0,015	0,047	0,031	0,031	0,015	0,031	0,015	0,031	0,032	0,031	0,063	0,031	0,047	0,02918
Conopt megoldás	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	
Conopt idő	0,047	0,047	0,047	0,062	0,047	0,094	0,062	0,047	0,047	0,062	0,062	0,078	0,094	0,062	0,11	0,094	0,141	0,07076
Conopt2 megoldás	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	nincs	nincs	nincs	
Conopt2 idő	0,031	0,047	0,032	0,047	0,047	0,125	0,062	0,047	0,031	0,047	0,047	0,062	0,094	0,078	---	---	---	0,05693
Cplex megoldás	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	
Cplex idő	0,032	0,032	0,031	0,031	0,015	0,063	0,031	0,016	0,031	0,032	0,016	0,031	0,047	0,031	0,031	0,031	0,032	0,03135
GLPK megoldás	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	rossz	jó	jó	jó	
GLPK idő	0,016	0,016	0,016	0,015	0,016	0,078	0,016	0,016	0,031	0,016	0,016	0,031	0,031	0,015	0,063	0,032	0,047	0,02771
LPSolve megoldás	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	jó	
LPSolve idő	0,031	0,032	0,047	0,032	0,047	0,047	0,062	0,078	0,125	0,062	0,047	0,063	0,109	0,078	0,125	0,125	0,187	0,07629
LindoApi megoldás	jó	jó	jó	jó	jó	jó	jó	jó	nincs	jó	jó	nincs	jó	nincs	nincs	nincs	nincs	
LindoApi idő	0,016	0,016	0,016	0,047	0,032	0,015	0,016	0,016	---	0,016	0,032	---	0,016	---	---	---	---	0,02164

6.1 Észrevételeim és tapasztalataim

Először is pár szót az előítéleteimről. Úgy gondoltam már jó előre, hogy a kutatásom is fényesen fogja bizonyítani azt a sztereotípiát, amely elmondható minden nagyobb problémafajtára. Miszerint a problémára léteznek jól kiforrott profi és igen drága rendszerek, ezek a legjobb és legkényelmesebb módja a probléma megoldására, csak hát ugye komoly pénzekbe kerülnek. Aztán e-mellette vannak az ugyancsak pénzbe kerülő szerényebb megoldások, amik kevesebb pénzért, és kevesebbet is nyújtanak, így fenntartva a profizmus látszatát és megcélozva az ár-érzékeny vevőcsoportot. E-mellett léteznek még a kiforrott ingyenes eszközrendszerek, amik egyes szegmensekben nagyon erősek, rengeteg munka és hatalmas, jól képzett közösség áll mögöttük, mégis egy teljes összevetésben egy piacvezető megoldással szemben alulmaradnak. Aztán még léteznek a lelkes kis ingyenes kezdeményezések, amik az alapdolgokat szépen ellátják, de semmi több.

Ezek a csoportok alapján önkéntelenül már az elején beskatulyáztam a kiválasztott solvereket, amikor utánanéztem az eredetüknek, a licencelésüknek. Voltak már az elején bizonyos elképzeléseim a solverekkel kapcsolatban, amelyek közül volt, ami teljesült, de volt olyan solver is, ami felül és olyan is, ami alulmúlta az elképzeléseimet. Ezeket a tapasztalatokat, megfigyeléseket igyekszem összegyűjteni ebben a részben.

Először is kezdeném a LGO-val, amit nagyon szerettem volna tüzetesebben is megvizsgálni, tesztelgetni, de sajnos erre nem volt módom, így magáról a solveréről így nem is tudok semmilyen véleményt alkotni. Pedig érdekes színfoltnak ígérkezett több szempontból is. Az LGO nem is egészen a lineáris programozási feladatokra van kitalálva/kihegyezve, épp csak azokat is meg tudja oldani. Az LGO a *Lipschitz(-continuous) Global Optimizer* rövidítése, amiből következik, hogy már a működése is merőben más lesz, nem a szimplex módszer szerint fog eljárni, és az éleken, csúcspontokban keresni az optimális megoldást, hanem egy belső pontból elindulva, és onnan haladva. Én ennek is tulajdonítom az előző fejezetben érdekességként kiragadott hatalmas iterációszámot.

Az LGO solverével nem tudtam az összegyűjtött speciális feladatokat megoldani, mert a hallgatói licence csak maximum öt változót és öt megkötést engedélyez. Ez nagyon lesújtott, főleg annak tudatában, hogy a többi kereskedelmi solver tesztlicence jóval nagyobb feladatok elvégzésére is alkalmas. Attól tartok, hogyha ingyenes teszteléssel kell döntenem a kereskedelmi solverek között, akkor nem feltétlenül győz meg az alkalmasságáról egy program, ami csak a legalapabb dolgokat mutatja fel. Tesztelni pedig csak olyan feladatokon lehet, amiket kis túlzással egy papíron egy ceruzával is meg lehet oldani. Bár a solver már a legelején kilógott a sorból, de azért is szerettem volna bevenni a kutatásba, mert a feladatok megoldására alkalmas lenne, még ha nem is erre van kihegyezve, és azért is, mert ez a solver egy magyar ember nevéhez kötődik, Pintér Jánoséhoz, aki többek közt a Magyar Tudományos Akadémia doktora.

Áttérve a ténylegesen tesztelt solverekre, a sort a CPLEX-el és a LindoApi-val kezdeném, ezeket gondoltam az elején a nagyon profi, és igencsak borsos áru solverek közé. Ezektől a solverektől alapkövetelmények tekintettem a tökéletesen pontos végeredményt, és a stabil és gyors számolást, inkább annak néztem utána, hogy ezen felül miket biztosítanak, mit nyújtanak a méregdrága licence mellé.

Az alapkövetelményekben nem is kellett csalódnom, a megoldások, amiket adtak hibátlanok, és a feladatok megoldásához szükséges idők is teljesen megállják a helyüket az összehasonlítás során. Amit még érdekességképpen megfigyeltem, az az, hogy amíg a solverek nagy részénél megfigyelhető az a tendencia, hogy minél nagyobb a feladat, annál több időt vesz el a feladat kiszámítása, ez ennél a két solvernél ez nem történt meg, a feladat méretétől függetlenül képes azonos időkön belül megadni a helyes megoldásokat. Mindét solvernél alapnak számítanak a folyamatos fejlesztések, a licencelt programok frissítései, a vásárlók oktatási anyagokkal, könyvekkel való ellátása.

Ami mégis dönthet egyik vagy másik javára. A CPLEX-ről információkat, a licencek feltételeit, árait, általános információkat elég nehézkes a weboldalukon találni. Számomra

legalábbis nem voltak egyértelműek hogy mit hol fogok megtalálni, sokszor elkeveredtem az oldalon belül teljesen máshova, a belső keresőjükkel se nagyon jutottam dűlőre. A Lindo oldalán valahogy egyből minden a „kezem alá" került, ha elnavigáltam az oldalon valahova akkor ott az fogadott amire én számítottam, az engem érdeklő információkra mindig nagyon hamar rábukkantam, a különböző licencekről és az áraikról is részletes információt lehet találni, elősegítik az érdeklődő információhoz jutását.

Ami még nagyon a Lindo malmára hajtja a vizet, az a LindoApi és a hozzá tartozó 477 oldalas mindenre kiterjedő dokumentációja, amikhez ugyancsak hozzáférek teljesen ingyenesen. Rendkívül szerencsésnek tartom, hogy a méregdrága solvert könnyedén bele lehet építeni saját alkalmazásba rengeteg támogatott nyelven, és ennek a használata is rendkívül rugalmas és egyszerű és rengeteg példával illusztrált. Nekem is teljesen idegen volt egy ilyen kész Api felhasználásának a mikéntje, de szinte elakadás nélkül egyből mindent meg tudtam vele oldani, az elakadásaimat is csak az okozta, hogy magát a fejlesztőkörnyezetet nem ismertem, a LindoApi használata már nagyon egyszerűnek volt mondható.

Összességében a CPLEX-el és a LindoApival is meg vagyok elégedve, bár amíg a CPLEX hozta a kötelezőt, addig a LindoApi teljesen lenyűgözött, így én a kettő közül az utóbbit választanám, ha ilyen választásra sor kerülne.

Következő bemutatásra váró solver a Conopt, amelynek két példányát is teszteltem, de a legfrissebb verzióját nem sikerült birtokba venni. A Conoptot skatulyáztam be a fizetős, de komolytalanabb kategóriámba, és nem is nagyon cáfoltak rá a tapasztalatok az előítéleteimre. A Conopt weboldalán szinte semmi információ nem érhető el, illetve felhívják a figyelmet arra, hogy bizonyos hibák lehetségesek, a feladatok megoldásainak helyességéért sem vállalnak semmiféle garanciát. Direkt értékesítést se találtam a solverrel kapcsolatosan, csak mások keresztül lehet hozzájutni a solverhez, így a lehetséges felelősséget, reklamációt is elhárítják magukról.

Azért hogy jót is írjak, a Conopt is hozza a minimumot, vagyis az eredményei rendre hibátlanok, illetve jól látszik a fejlődés is, a Conopt2 már jóval jobb időket produkál mint a Conopt, bár így is jócskán elmarad a nagy drága solverektől, sőt az ingyenesek között is van nála gyorsabb. Bár ha ezt a fejlődést tartani tudták, akkor a már meglévő Conopt3 talán felveheti ilyen téren is a versenyt más solverekkel. Összességében a Conopt solvert nem tartom igazán jó lehetőségnek, engem egyáltalán nem sikerült meggyőznie.

Ezek után következzenek az ingyenes solverek. A GLPK és az LPSolve voltak azok, amiktől többet vártam. Mindkettő igen elterjedt. A GLPK a GNU égisze alatt van, azok között pedig nem ritka, hogy az ember kincseket talál. Az LPSolve solver mögött pedig a nagy támogatottság állt. Komolyságát jelzi a magas verziószám, az elkészült leírások hozzá, és a meglepően aktív és nagyszámú Yahoo Groups-on található community.

Nos, végeredményben ez a két solver nem nyűgözött le a mutatott eredményekkel, bár igaz hogy panasz sem lehet rájuk. A GLPK igen szép futási időket produkált, jó pár feladatot gyorsabban oldott meg, mint a Cplex. Az eredmény így akár még bravúros is lehetett volna, de teljesen lerontotta az összképet, hogy az egyik feladatba (boeing2) teljesen beletört a bicskája. A megoldásban nem csak századokkal, vagy ezredekkel tévedett, de teljesen mellélőtt. Persze mivel ez egy teljesen ingyenes és nyílt forráskódú szoftver, ezért ezt nem lehet felróni neki, viszont így le se nyűgözött. A GLPK-nál már előjött az, hogy ahogy egyre nagyobb a feladat, úgy lesz egyre több idő is a megoldás, így valószínűsítem, hogy olyan ezer változó és megkötés bonyolultságú feladatok mellett már időben se tudná felvenni a versenyt a Cplexel.

Az LPSolve már látszólag pontosan számol, minden kiválasztott feladatra tökéletes megoldást adott, viszont a futási ideje magasan a legnagyobb volt. Futási időben a nagyságrendi különbség is csak nőtt a feladatok méretével, így ezt tekintve nagy a hátránya a kereskedelmi solverekkel szemben.

A GLPK és az LPSolve így összességében a saját szubjektív osztályozásom szerint közepes teljesítményt nyújtottak, vannak erősségeik és vannak gyengéik, de ezek nem akkora problémák tekintve az ingyenességüket.

A végére maradt a CoinMP, amitől nem nagyon vártam igazán sokat. Az egész szoftver egy nem túl komoly gyűjteménynek tűnt elsőre, amiről információt sem sokat lehetett találni, illetve a licencének a megfogalmazása sem sejtetett valami komoly programot. Viszont a tesztelés után a CoinMP minden előítéletemre rám cáfolt, nagyon szép eredményeket produkált. A feladatokra kivétel nélkül hibátlan eredményeket adott, és a futási idők is igen szépek. A vizsgált feladatoknál összességében jobb futási időket lehetett mérni, mint a Cplexnél. Bár itt is észrevehető, hogy a feladatok növekedésével együtt jár a számoláshoz szükséges idő növekedése, így feltételezem, hogy nagyobb feladatoknál már a kereskedelmi szoftver előnye jönne elő, de az eredmény szerintem így is nagyon szép.

6.2 Ajánlásaim a tapasztalataim alapján

A kutatásom során megismerkedtem több különböző operációkutatási solverrel, ezek után már mernék ajánlani különböző helyzetekben különböző solvereket. Tapasztalatom első sorban a lineáris programozási feladatokra terjed ki. Ha valaki kisebb kutatásokhoz, projektekhez már kész solvert szeretne használni, annak elsősorban a CoinMP-t ajánlanám, megtalálható a forráskódja, a licencében nincsenek különösebb megkötések, és teljesen megbízhatónak is tűnik.

Amennyiben az LP feladatok megoldásának eredménye rendkívüli fontossággal bír, akkor még megoldás lehet az, hogy mind a három jól teljesítő ingyenes solvert is futtatjuk, így sokkal biztosabbak lehetünk a kapott eredményben. Természetesen ekkor le kell mondani teljesen a gyors futási időkről. Illetve nem szabad elfelejteni, hogy kereskedelmi célokra a három solverből csak kettő használható fel.

Végül pedig, ha a lineáris programozási feladatok hatalmasak, és a megoldások pontossága is rendkívüli fontosságú, komoly döntések és értékek függnek tőle, akkor mindenképpen érdemes áldozni egy komoly solverre több okból is. Mindenképpen előnyös ezeknek a cégeknek a többéves tapasztalata, ezek a solverek folyamatosan fejlődnek, frissülnek, tökéletesednek. A pénzünkért joggal várhatunk hibátlan eredményeket, illetve felelősséget is mind az eredményekért, mind a solverek hibátlan működéséért. Ha mégis bármi hibát okozna egy kereskedelmi szoftver, úgy joggal lehet igényünk a hiba javítására, és valamiféle kárpótlásra is, illetve a szoftverek beüzemeléséhez is biztosan megkapunk mindenféle segítséget. Ezek az előnyök nem találhatók meg az ingyenes megoldásoknál, és ezekre érdemes lehet áldozni, ha a kívánt solver működésének és eredményeinek fontossága rendkívül nagy.

Ilyen esetben pedig én a Lindot mindenképp a Cplex elé helyezném. Egyrészt a honlapjuk információtartalma között is hatalmas különbségek vannak a Lindo javára, bár a döntésemben ez igen kis szerepet játszik. Ami miatt mindenképpen a Lindora esne a választásom, az a rettenetesen könnyen használható LindoApi, a hozzá tartozó rendkívül részletes 477 oldalas leírás, és hogy mindez már ingyenesen is elérhető, használható és tesztelhető. Az ár viszonylatában pedig nagyjából hasonló a helyzet, mindkettő a „méregdrága” kategóriában foglal helyet.

7. Összefoglalás:

Kutatásom célja az volt, hogy beszerezzek több különböző solvert, amelyek képesek lineáris programozási feladatok megoldására, és ezeket igyekezzek azonos körülmények között tesztelni és összehasonlítani. Célom volt még továbbra az is, hogy az egyik piacvezető kereskedelmi solvert felhasználva Windows platform alá grafikus környezetet fejlesszek, ami ugyancsak képes lineáris programozási feladatokat megoldani. Többek közt azért is esett választásom a Lindora, mert az egyetemi oktatás során az Operációkutatás tantárgy keretein belül a Lindo programjával a Lingoval ismerkedhettünk meg.

Kutatásomhoz segítséget témavezetőmtől Dr. Bajalinov Erik tanár úrtól kaptam, illetve angol nyelvű internetes forrásokból. Magyar nyelvű forrást ezekben a témákban egyáltalán nem találtam. A kitűzött célokat sikerült a kutatásom során megvalósítani, több különböző solvert is sikerült működésre bíznom mind a kereskedelmi, mind az ingyenes szoftverek közül.

A jól megválasztott feladatoknak hála valamiféle különbségek ki is rajzolódtak az egyes szoftverek között. Voltak, amik jobban, és voltak, amik rosszabbul teljesítettek. Igyekeztem minden általam megismert solverről saját véleményt kialakítani, ebben nem csak a teszteredményeket használtam fel, hanem az Interneten található egyéb információkat is.

Igyekeztem diplomamunkámban a kutatás során szerzett gyakorlati tapasztalatokat átadni, illetve a végén egy szubjektív összehasonlítást adni a megismert solverekről. Úgy vélem kutatómunkám hasznos volt, mert a különböző solverekről magyarul szinte semmit nem találtam, illetve ilyen irányú összehasonlítást sem találtam sehol se magyar se angol nyelven, így munkám még eddig érintetlen területeket fedhet le.

Célom volt továbbá hogy kutatásom eredményeként ne csak száraz adatokat tárjak fel, hanem hasznos gyakorlati tanácsokat is átadhassak a diplomamunkámban. Úgy érzem ez sikerült is, mind a solverek beszerzésével és használatával kapcsolatban, mind a saját fejlesztésű program elkészítésével kapcsolatosan megvalósítani.

Irodalomjegyzék:

CoinMP (angol)

https://projects.coin-or.org/CoinMP

https://projects.coin-or.org/svn/CoinMP/branches/jp/LICENSE.txt

https://projects.coin-or.org/svn/CoinMP/branches/jp/CoinMP/src/

Conopt (angol)

http://www.conopt.com/

http://www.conopt.com/Algorithm.htm

http://www.conopt.com/Availableforms.htm

CPLEX Mérési útmutató (magyar)

http://w3.tmit.bme.hu/labor/meresek/cplex/CPLEX.doc

Fortune 500 – 2007 (angol)

http://money.cnn.com/magazines/fortune/fortune500/2007/

GAMS – Commercial Price Schedule – February 14, 2007 (angol)

http://www.gams.com/sales/commercialp.htm

GLPK (angol)

http://www.gnu.org/software/glpk/

http://gnuwin32.sourceforge.net/packages/glpk.htm

ILOG Academic Price List (angol)

http://www.ilog.com/products/optimization/academic/US_AcademicPriceList.pdf

Komáromi Éva – Lineáris programozás (magyar)

http://web.uni-corvinus.hu/~opkut/elibd/linearis_programozas.pdf

LGO – János D. Pintér (angol)

http://www-fp.mcs.anl.gov/otc/Guide/SoftwareGuide/Blurbs/lgo.html

http://www.gams.com/solvers/GAMS_LGO_paper.pdf

http://www.pinterconsulting.com/Pinter_Resume.pdf

http://www.pinterconsulting.com/index.html

http://optnet.itwm.fhg.de/opt-net/documents/v98w02n2

Lindo Systems (angol)

http://www.lindo.com/

Lindo Store (angol)

http://www.lindo.com/store/?p=productsList&iCategory=15

http://www.lindo.com/store/?p=productsList&iCategory=14

http://www.lindo.com/store/

LindoAPI (angol)

http://lindo.com/downloads/cap/opsysdllf.html

Linear Programming (wiki) (angol)

http://en.wikipedia.org/wiki/Linear_programming

Lineáris Programozás (wiki) (magyar)

https://miau.gau.hu/mediawiki/index.php/Line%C3%A1ris_programoz%C3%A1s

LPSolve (angol)

http://lpsolve.sourceforge.net/5.5/

http://sourceforge.net/projects/lpsolve

Maximal Software – MPL for Windows (angol)

http://www.maximal-usa.com/download/

http://www.maximal-usa.com

http://www.maximal-usa.com/mpltutor/Session2.html

Molnár Sándor Dániel – Esszé a lineáris programozásról (magyar)

http://www.stud.u-szeged.hu/Molnar.Sandor.Daniel/koszi_essay.pdf

MPS (angol)

http://www-fp.mcs.anl.gov/otc/Guide/OptWeb/continuous/constrained/linearprog/mps.html

http://en.wikipedia.org/wiki/MPS_(format)

http://www.hit.bme.hu/~lakatos/mps_format.txt

http://lpsolve.sourceforge.net/5.5/mps-format.htm

Netlib – LP problems (angol)

http://www.netlib.org/

http://www-fp.mcs.anl.gov/otc/Guide/TestProblems/LPtest/

http://www.netlib.org/lp/data/

http://cuter.rl.ac.uk/cuter-www/Problems/netlib.shtml

Optimization Softwares and Test Problems (angol)

http://www.misojiro.t.u-tokyo.ac.jp/%7Etomomi/opt-codeE.html

Rapcsák Tamás – Az operációkutatás kialakulásáról és hazai helyzetéről (magyar)

http://www.hors.hu/rapcsak.htm

Köszönetnyilvánítás:

Ez úton szeretnék köszönetet mondani témavezetőmnek, Dr. Bajalinov Erik, tudományos főmunkatárs úrnak, diplomadolgozatom elkészítésében nyújtott segítségéért, munkám irányításáért és értékes tanácsaiért.