

fejezet

Felhasznált technológiák

Szakedolgozatomban a matematikai modellezés technikáját vettem alapul, ami a valóság egyszerűsített verziójára törekszik és ezáltal a dolgok közti összefüggéseket jobban átláthatóvá teszi. Egy modell általánosságban a következő részeket tartalmazza:

- paraméterek: állandók
- változók: az optimalizálási folyamat során kapnak értéket
- korlátozások: korlátozó feltételeket kifejező egyenlőtlenségek
- célfüggvény: a modell célja (minimalizálás/maximalizálás)

Egy valós probléma (elektromos buszok ütemezése) szimulációjára használtam a modellezést és a lineáris programozás segítségével kerestem az optimális megoldást. A következőkben az általam használt technológiákat mutatom be.

MILP

Az elkészített modelleket a MILP (Mixed Integer Linear Programming) vegyes egész értékű lineáris programozási módszer alkalmazásával hoztam létre. A MILP modellek abban különböznek a sima LP modellektől, hogy egész és folytonos változókat is tartalmazhatnak, míg az LP modellek csak folytonosakat. A vegyes egész értékű lineáris programozás egy elég széles körben használt módszer, ezért többféle szoftveres megoldó (solver) is a rendelkezésünkre áll.[1] Néhány példa a teljesség igénye nélkül:

- CPLEX
- GLPK
- Gurobi

Több nyelv áll rendelkezésre a MILP modellek leírására és vannak olyan eszközök, amik többfajta kiterjesztést is támogatnak. A GLPK (GNU Linear Programming Kit) modellező nyelve a GMPL (GNU MathProg Language), általánosan például a .mod és .dat kiterjesztéseket használja, míg például a CPLEX .lp kiterjesztésű fájljait a Gurobival is lehet futtatni.

A solverek közül azért választottam a GLPK-t, mert ingyenesen hozzáférhető és könnyű használni a modellalkotáshoz, a Gurobit pedig azért, mert hatékonyan lehet vele a komplexebb modelleket is tesztelni.

GMPL

Az általam készített modelleket az AMPL matematikai modellező nyelv szintaxisára épülő GMPL nyelven írtam.[2] A GMPL modelleknek két fő része van: a modell és az adatok. A modell részben lehet meghatározni a paramétereket és a változókat, amikkel a célfüggvényt szeretnénk optimalizálni, illetve itt adhatók meg a feltételek is. Az adat részben a paramétereknek kell értéket szolgáltatni. A modell részben a kimenet formázására is van lehetőség.

glpsol

A glpsol a GLPK csomag részeként egy önálló LP/MIP solver programként működik. Én Parancssorból használtam arra, hogy .mod kiterjesztésű fájlból .lp kiterjesztésű fájlt hozzak létre, mivel a Gurobi azt már tudja kezelni. Az átalakítást a következő módon lehet megtenni:

```
glpsol --check --wlp pelda.lp --math pelda.mod
```

Gusek

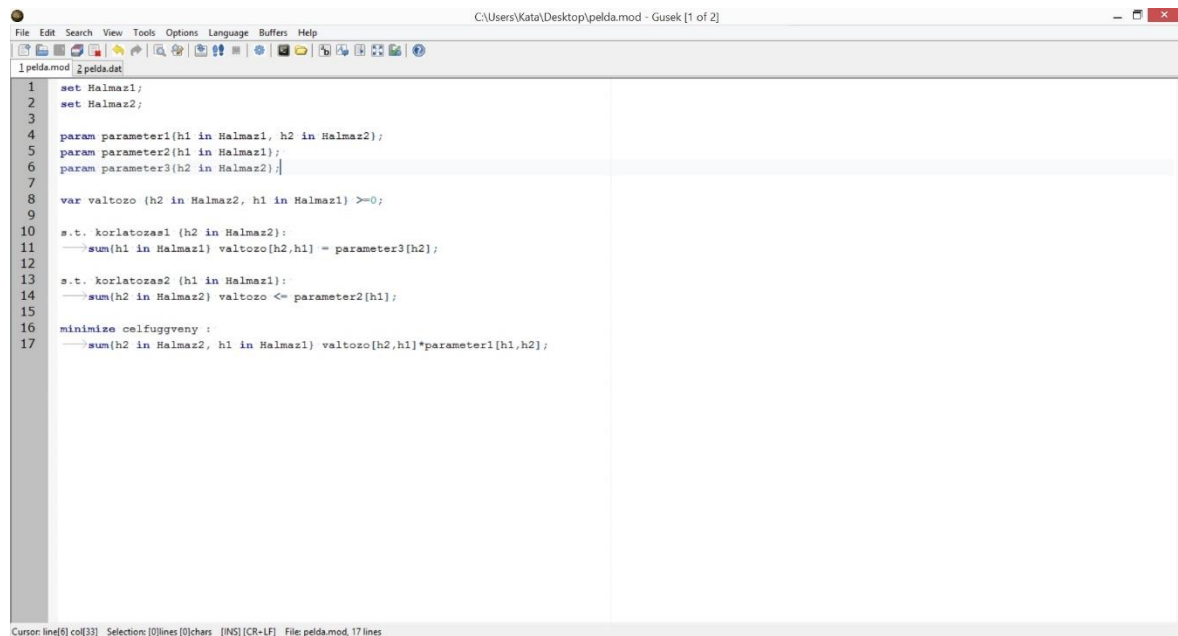
A MILP modelljeim leírására a Gusek nevű nyílt forráskódú fejlesztői környezetet használtam. Ez a szoftver támogatja például a GLPK LP/MIP és a CPLEX LP modellek fejlesztését is.

Az alábbi kiterjesztésű fájlokkal dolgoztam a Gusek-ben:

- .mod: GMPL modellfájl
- .dat: GMPL adatfájl
- .out: GLPK kimeneti fájl

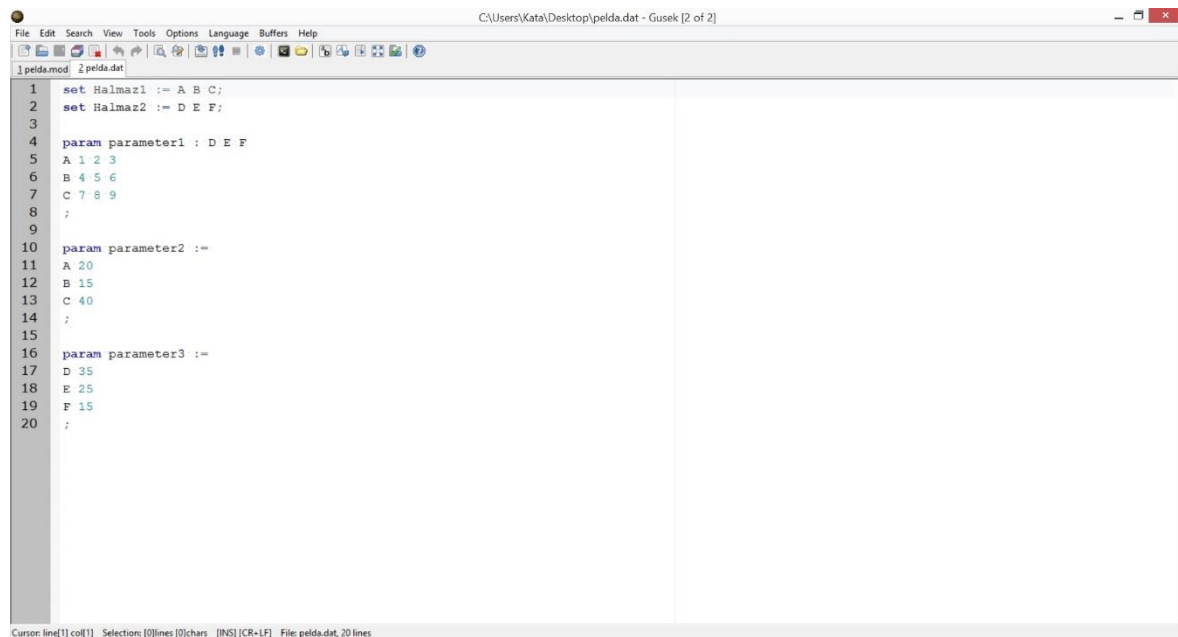
Egyidejűleg több fájl is lehet nyitva a Gusek ablakban, amik esetenként csak a kiterjesztésükben különböznek. Az . és a .ábrán egy rövid példával szemléltetek a Gusek

fejlesztői környezetben egy GMPL nyelven írt modellt. A **set** kulcsszóval adhatok meg halmazokat, a **param** paramétereket jelöl, a **var** kulcsszóval vehetem fel a változókat, az **s.t.** pedig a korlátozásneveket mutatja. A **minimize** szó pedig azt jelenti, hogy a célfüggvényt minimalizál.



```
1 set Halmaz1;
2 set Halmaz2;
3
4 param parameter1(h1 in Halmaz1, h2 in Halmaz2);
5 param parameter2(h1 in Halmaz1);
6 param parameter3(h2 in Halmaz2);
7
8 var változo (h2 in Halmaz2, h1 in Halmaz1) >=0;
9
10 s.t. korlatozas1 (h2 in Halmaz2):
11    ->sum(h1 in Halmaz1) változo[h2,h1] = parameter3[h2];
12
13 s.t. korlatozas2 (h1 in Halmaz1):
14    ->sum(h2 in Halmaz2) változo[h2,h1] <= parameter2[h1];
15
16 minimize celuggveny :
17    ->sum(h2 in Halmaz2, h1 in Halmaz1) változo[h2,h1]*parameter1[h1,h2];
```

ábra: .mod fájl a Gusek IDE-ben



```
1 set Halmaz1 := A B C;
2 set Halmaz2 := D E F;
3
4 param parameter1 : D E F
5 A 1 2 3
6 B 4 5 6
7 C 7 8 9
8 ;
9
10 param parameter2 :=
11 A 20
12 B 15
13 C 40
14 ;
15
16 param parameter3 :=
17 D 35
18 E 25
19 F 15
20 ;
```

ábra: .dat fájl Gusek IDE-ben

Gurobi

Főként a végleges modellem teszteléséhez használtam a Gurobi Optimization megoldót, mivel a Gusek már nem tudott megbirkózni a megnövekedett adatfájllal. Ez a solver több problémátípust is képes kezelni, például a lineáris programozási modelleken kívül a kvadratikus (nemlineáris) programozási modelleket is megoldja.

Többféle modellező és programnyelvet támogat, mint például az AMPL, MATLAB, C, C++, Python, Java, vagy a .NET.

A modelljeim futtatásához a Gurobi parancssori interfészét használtam a `gurobi_cl` paranccsal. A Gusekben fejlesztett `.mod` kiterjesztésű modellfájlokat az előbbieken a `glpsol`-nál megismert módon átalakítottam `.lp` fájlkká, ezután pedig a `ResultFile` paraméter segítségével megadtam, hogy az `.lp` fájl milyen `.sol` kiterjesztésű fájlba mentse az eredményeket. Példa a modell futtatásra:

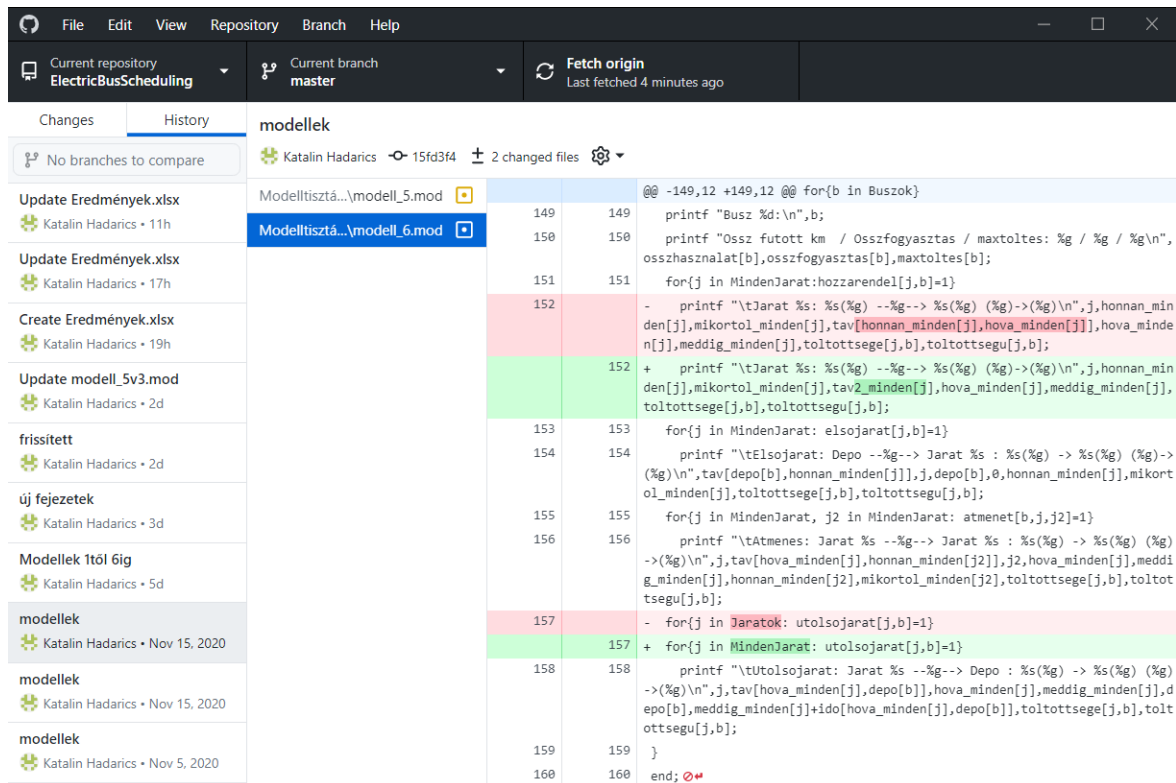
```
gurobi_cl ResultFile=pelda.sol pelda.lp
```

GitHub

A modellek fejlesztése során a kódokat a GitHub-on tároltam. Ez egy verziókövetési szolgáltatás, ami a Git nevű nyílt forráskódú elosztott verziókezelő rendszeren [3] alapul. Néhány alapfogalom:

- commit: egy fájl aktuális verziója
- branch: a projekt egy ága, commitok összessége
- repository: lokális tárolóegység a projekt branch-ei és commit-jai számára
- pull: a tároló áthúzása, a távoli repository tartalmának letöltése a helyi repository-ba
- push: a tároló áttolása, a helyi repository tartalmának feltöltése a távoli repository-ba
- clone: távoli tároló másolása
- fork: egy már létező projekt másolása (pl. továbbfejlesztés céljából)

Az én távoli repositorymnak például az volt a neve, hogy `ElectricBusScheduling`. A GitHub Desktopon található helyi repositorym pedig a .ábrán látható.



Ábra: GitHub Desktop

Hivatkozások:

[1]: Linderroth, J. T., & Lodi, A. (2010). MILP software. *Wiley encyclopedia of operations research and management science*.

[2]: <http://gusek.sourceforge.net/gmpl.pdf>

[3]: Loeliger, J., & McCullough, M. (2012). *Version Control with Git: Powerful tools and techniques for collaborative software development*. " O'Reilly Media, Inc."