INSTALLATION GUIDE OF LPL MODELING SYSTEM ON WINDOWS

Tony Hürlimann

LPL Version 7.05

Table of Contents

Introduction

LPL is a complete **mathematical modeling system** with a point-and-click user interface and a simple yet powerful **modeling language.** The language is a structured mathematical and logical modeling and programming language with an extended index mechanism, which allows one to build, maintain, modify, and document large linear, non-linear, and other mathematical (optimization) models. A language interpreter translates the model automatically into a solver acceptable form; reads data directly from database, calls a solver and can write the results directly back to the database or generate complex solution report files. LPL can communicate with many commercial and free solvers.

The modeling language **is the glue** that brings all modeling tasks under one hat. Hence, LPL communicates with the different parts and acts as an integrator, as shown in Figure 1.

Figure 1: The modeling language as a glue of modeling tasks

1. LPL itself does not store efficiently a large amount of data, this is better done in databases. However, LPL contains instructions on a high-level to communicate with

databases, reading and writing data. Nevertheless, LPL can store efficiently sparse tables of considerable size itself and can manipulate multi-dimensional data cubes as pivot tables itself.

2. LPL itself cannot solve mathematical models, but it can communicate well and transparently with many free and commercial solvers. Switching from one to another solver is easily done and requires normally only to set a parameter. Nevertheless, LPL integrates a linear and a tabu-search solver and can solve small problems itself.
3. LPL itself does not contain a reporting tool, but it has strong links with a professional reporting tool (*FastReport*©), that is callable as a library. Report template creation and complex PDF-report file, and others can be directly piloted by LPL language instructions. Nevertheless, LPL does have itself in addition a limited way to generate formatted output.
4. The very core of LPL is that it can compactly represent the **mathematical structure** of a problem in the form close to the common mathematical notation, in addition it contains **algorithmic instructions** that pilot the different modeling tasks, as reporting, solving, reading/writing data, calculating, etc.

Hence, LPL is an *open* modeling system that can be linked in many ways to various modeling environments. It is important to be aware of that when installing LPL, since for professional use the user also need powerful solvers, access to databases, LaTeX for generating model documentation, etc. These pieces of software also should be installed.

Installing LPL

The whole modeling system can be downloaded as a single file *install-lpl.exe* from the Internet site of MatMod.ch. Just click on the appropriate link and the file will be downloaded through the Internet browser.

After downloading, double-click the file *install-lpl.exe* in order to install it on your computer. Follow the instructions. By default, the installation software will create a new folder, called '*LPL'*, on the harddisk and unpack every file within this folder.

The user may choose another folder name, but he or she should be aware, that various free solvers that can be downloaded also from MatMod should – later on – must be installed in the same folder (otherwise one needs to add an environment variable as explained below).

The installation program also creates a shortcut called 'LPL*'* on the desktop for the 64bit version. You can remove it without any harm if you like.

Everything is placed within the installed folder – '*LPL*' by default – and you may copy the whole folder to another place if you like and it works as before. From now on we call the folder where LPL has been installed as the *LPLEXE folder.*

The lpl model file must always have file extension *.lpl*. In the LPLEXE folder you find a file called *alloy.lpl*. This is our small test model we shall use to see whether everything works fine.

Recommendation: I recommend linking the lpl-files with the *lplw.exe* program in such a way that double-clicking a model file opens directly the *lplw.exe* program. This can be done as follows:

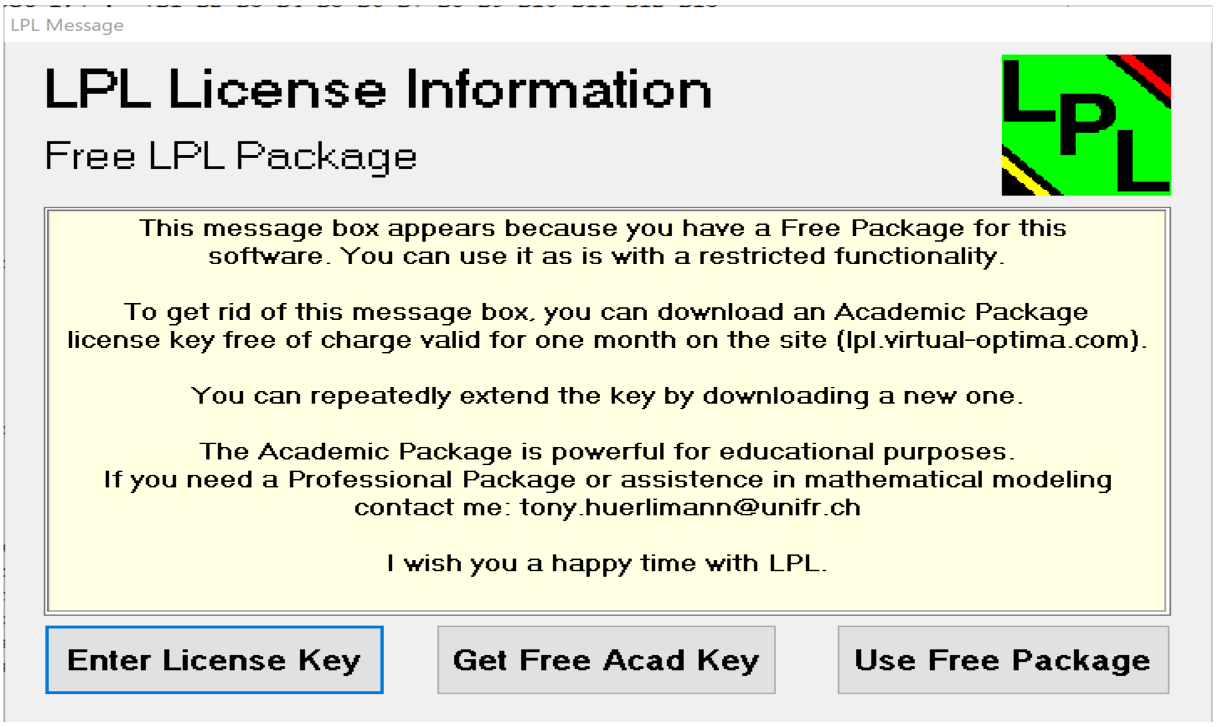
1. Right-click the file *alloy.lpl* in a Windows browser.
2. Click the menu item "Open with…".
3. Choose *lplw exe* from the list of programs, if it is not in the list then choose the button "browse" and look for the *lplw.exe* program.
4. Choose it and all files with file extension *.lpl* will be opened directly with *lplw.exe*.

The installation is complete! You can open the LPL modeling environment by three ways:

1. Double-Click the shortcut '*LPL*' on your desktop. This also opens the model *alloy.lpl* by default.
2. Double-Click the file *lplw.exe* within the folder *LPLEXE*. An empty model opens with a parse error that can be ignored. Choose then the menu item <file/open model> to open a model.
3. Double-Click a model file with extension *lpl* (if you followed the recommendation above).

In either case, after the first installation you will be presented with the following message box (Figure 2) when launching the software. This Initial Message Box informs you that you have a "Free Package". You can immediately download a license key from **Get Free Acad Key** which is a key for an Academic Package of LPL valid for one month. You may then enter this License Key by clicking "Enter License Key". You may also ignore it and just click "Use Free Package". You can use the Free Package free of charge (more on Packages and a concise overview of LPL's functionality see This File).

After entering a license key by clicking "Enter License Key", the user can enter the license key in the About Box (Figure 3). This box informs the user also of the Package installed, the due date of the license and about the author, etc.


LPL Message

LPL License Information

Free LPL Package

This message box appears because you have a Free Package for this software. You can use it as is with a restricted functionality.

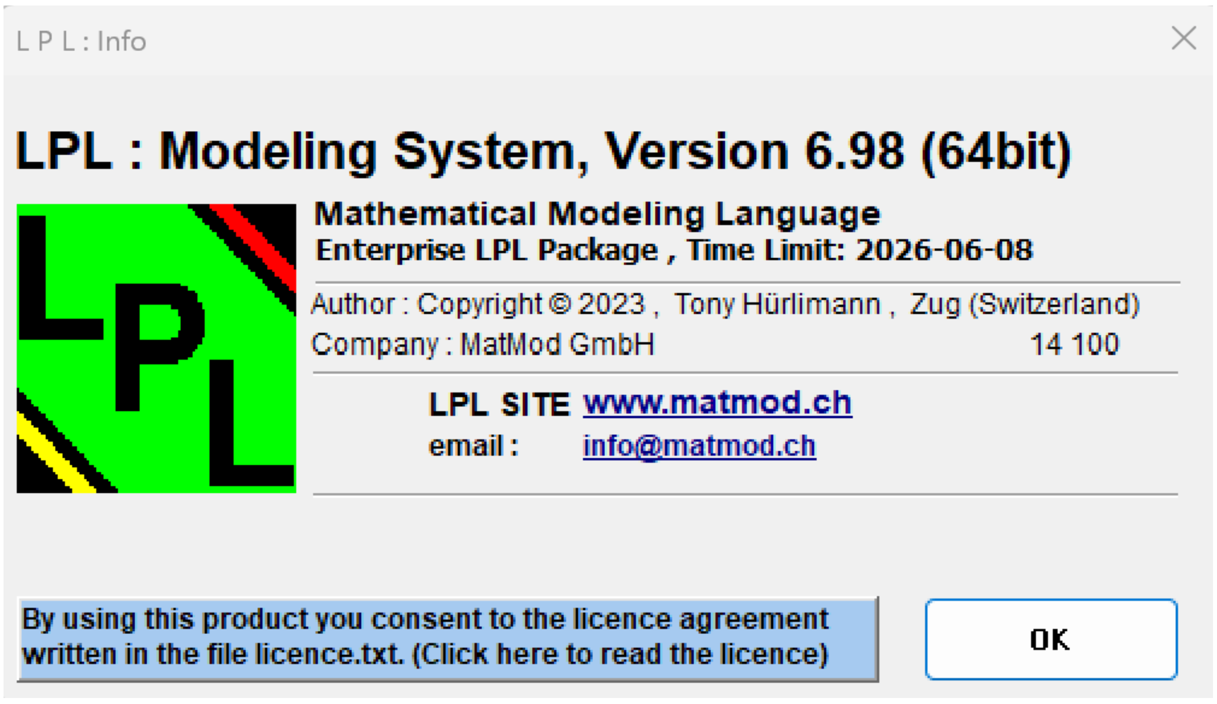
To get rid of this message box, you can download an Academic Package license key free of charge valid for one month on the site (lpl.virtual-optima.com).

You can repeatedly extend the key by downloading a new one.

The Academic Package is powerful for educational purposes.
If you need a Professional Package or assistence in mathematical modeling contact me: tony.huerlimann@unifr.ch

I wish you a happy time with LPL.

Figure 2: Initial Message Box in LPLW


L P L : Info

LPL : Modeling System, Version 6.98 (64bit)

Mathematical Modeling Language
Enterprise LPL Package , Time Limit: 2026-06-08

Author : Copyright © 2023 , Tony Hürlimann , Zug (Switzerland)
Company : MatMod GmbH 14 100

LPL SITE www.matmod.ch
email : info@matmod.ch

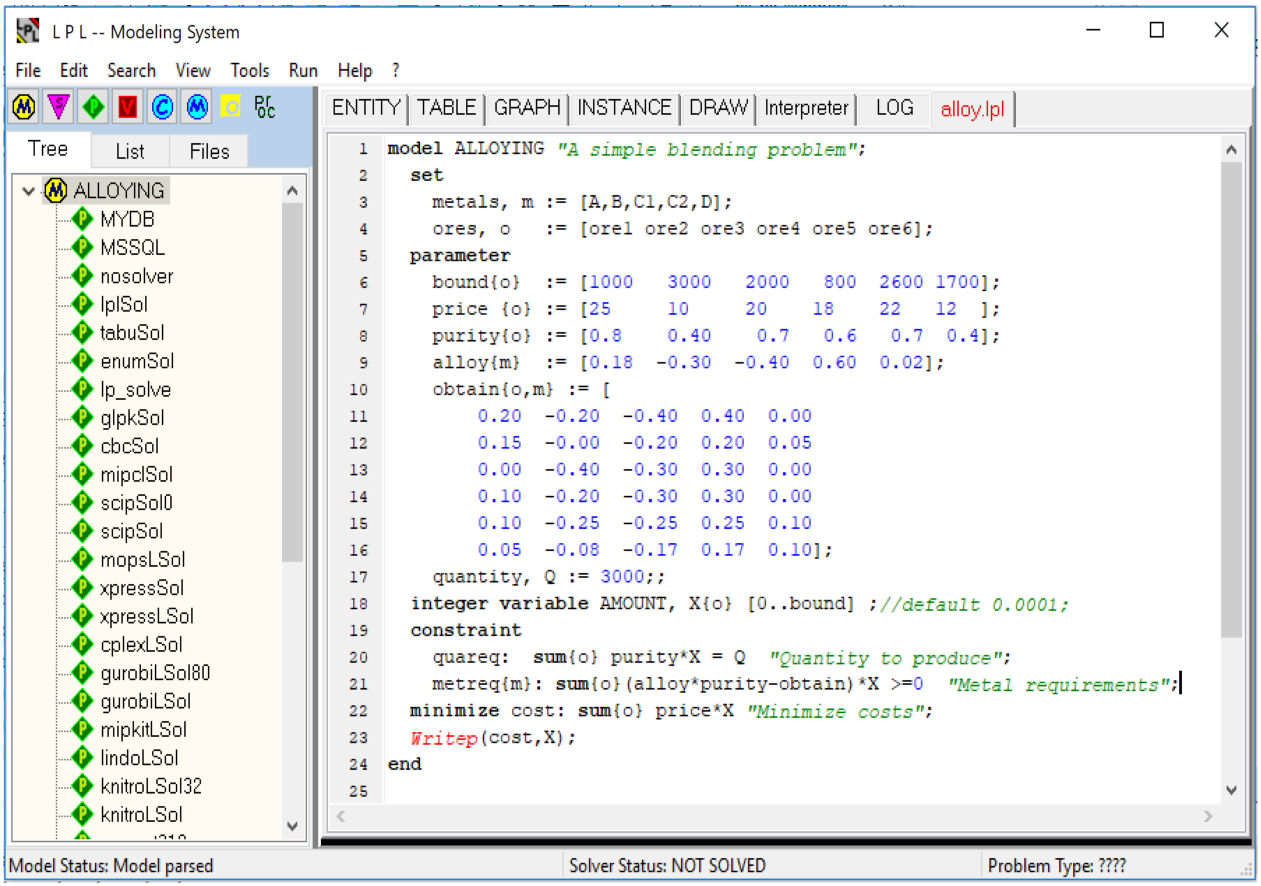
By using this product you consent to the licence agreement written in the file licence.txt. (Click here to read the licence)

Figure 3: About Box in LPLW

After clicking OK, the main screen opens. (This opens directly when a license is installed.)

Make sure that the model *alloy.lpl* is opened, that is, you should see the screen as in Figure 4 after launching LPLW and closing the message box above.

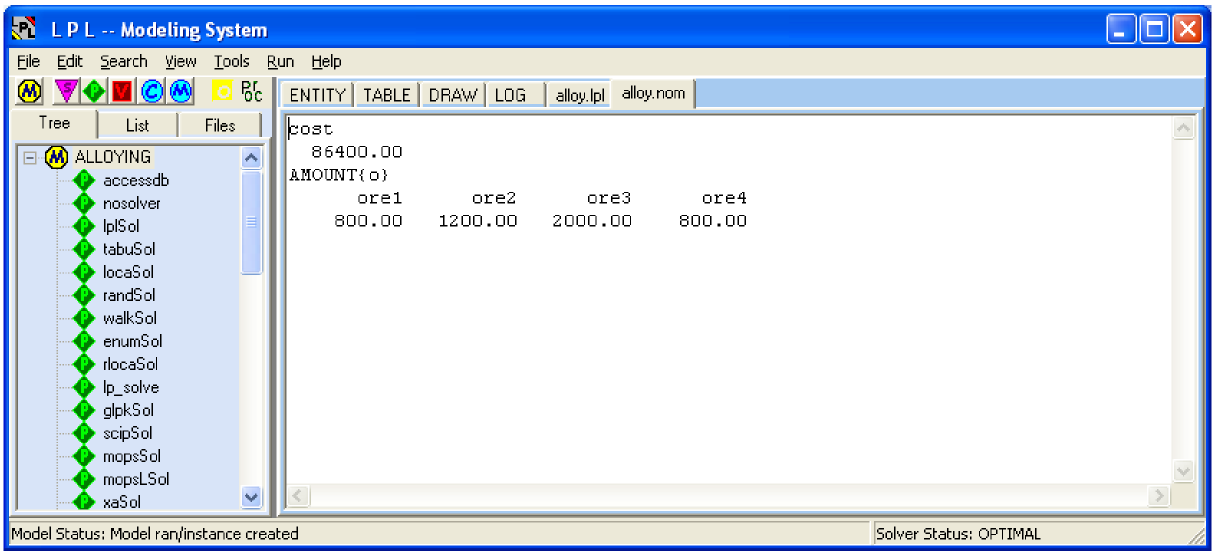
If this is not the case, then choose the menu item <File / Open Model> and open the model file *alloy.lpl* (which is found in the folder *LPLEXE*).



```
model ALLOYING "A simple blending problem";
  set
    metals, m := [A,B,C1,C2,D];
    ores, o   := [ore1 ore2 ore3 ore4 ore5 ore6];
  parameter
    bound{o}  := [1000   3000   2000   800  2600 1700];
    price {o} := [25     10     20     18    22   12  ];
    purity{o} := [0.8    0.40    0.7   0.6   0.7  0.4];
    alloy{m}  := [0.18  -0.30  -0.40  0.60  0.02];
    obtain{o,m} := [
        0.20  -0.20  -0.40  0.40  0.00
        0.15  -0.00  -0.20  0.20  0.05
        0.00  -0.40  -0.30  0.30  0.00
        0.10  -0.20  -0.30  0.30  0.00
        0.10  -0.25  -0.25  0.25  0.10
        0.05  -0.08  -0.17  0.17  0.10];
    quantity, Q := 3000;;
  integer variable AMOUNT, X{o} [0..bound] ;//default 0.0001;
  constraint
    quareq:  sum{o} purity*X = Q  "Quantity to produce";
    metreq{m}: sum{o}(alloy*purity-obtain)*X >=0  "Metal requirements";
  minimize cost: sum{o} price*X "Minimize costs";
  Writep(cost,X);
end
```

Figure 4: After opening the model alloy.lpl

Click now the menu item <Run / Run Model> or press the key F9. The model will run and be solved. The solution is presented in a new tab as follows (Figure 5).



```
cost
   86400.00
AMOUNT{o}
         ore1         ore2         ore3         ore4
      800.00      1200.00      2000.00       800.00
```

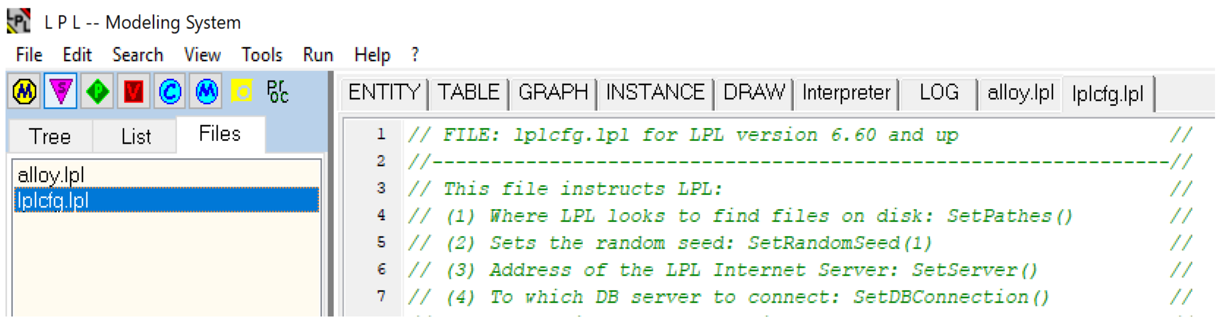
Figure 5: Solution of alloy.lpl

If the solution is not generated as shown in Figure 5 then the solver is not correctly configured. LPL has an integrated linear solver that solved this model. Normally, this solver is configured in the file *lplcfg.lpl*. We shall look at this file later on.

In any case, if Figure 5 does not appear as it should then click the **Files** tab on the left (see Figure 6) and click on *lplcfg.lpl*. The file opens in a new tab on the right side. Go down until you find the line

```
    --SetSolver(lplSol); -- this is the default solver in LPL
```

and remove the two dashes at the beginning of the line. And then make sure that *all lines after that line* begin with two dashes. Now try again menu <Run/Run Model>. You are prompted to save the lplcfg.lpl file. Click Yes.


L P L -- Modeling System
File Edit Search View Tools Run Help ?
ENTITY TABLE GRAPH INSTANCE DRAW Interpreter LOG alloy.lpl lplcfg.lpl
Tree List Files
alloy.lpl
lplcfg.lpl

```
// FILE: lplcfg.lpl for LPL version 6.60 and up                   //
//----------------------------------------------------------------//
// This file instructs LPL:                                       //
// (1) Where LPL looks to find files on disk: SetPathes()         //
// (2) Sets the random seed: SetRandomSeed(1)                     //
// (3) Address of the LPL Internet Server: SetServer()            //
// (4) To which DB server to connect: SetDBConnection()           //
```

Figure 6: Open lplcfg.lpl File

Now you have a working modeling environment.

Uninstalling LPL

In the folder *LPLEXE* you will find a file called *Uninstall-lpl.exe*. To uninstall LPL just double-click *Uninstal-lpll.exe* and everything you installed will have gone. Please note that the whole folder *LPLEXE* will be deleted. Hence, if you created new models that are in that folder they will also be removed. Save them in another place before uninstalling if you like.

Installing a Solver

Various solvers in LPL are preconfigured. By default, LPL comes with an internal linear solver. This solver is very limited and cannot solve mixed integer or non-linear problems. To solve such problems, the user needs to install other solvers.

The HIGHS solver

HIGHS is a free solver available on the Internet (HIGHS PAGE). A version is also ready on the download page at MatMod. To use that solver one needs to do the following steps:

1. Download the file *highs.exe* from MatMod (Link to HIHGS.EXE (Windows)).
2. Copy the exe file into the same folder as LPL. Of course, this is not a necessity: the solver can be installed anywhere you want, however then an environment variable must be set, such that LPL can find the solver. For the moment we suppose that the solver is in the same folder as LPL.
3. Launch LPLW and open the model file *alloy.lpl* as before.
4. You see three tabs on the left: *Tree* / *List* / *Files*. Click the tab *Files* now as before. You are presented with a list of two files. Click on *lplcfg.lpl*. You will now have the

following window (Figure 9). The file lplcfg.lpl has been opened and is shown in a new tab on the right.

The file *lplcfg.lpl*: The file *lplcfg.lpl* is a file that contains “global” LPL instructions and it is automatically linked to each model if present. Therefore, it appears in the *Files-List-Tab* of the LPLW program (left-side).

It contains various instructions that:

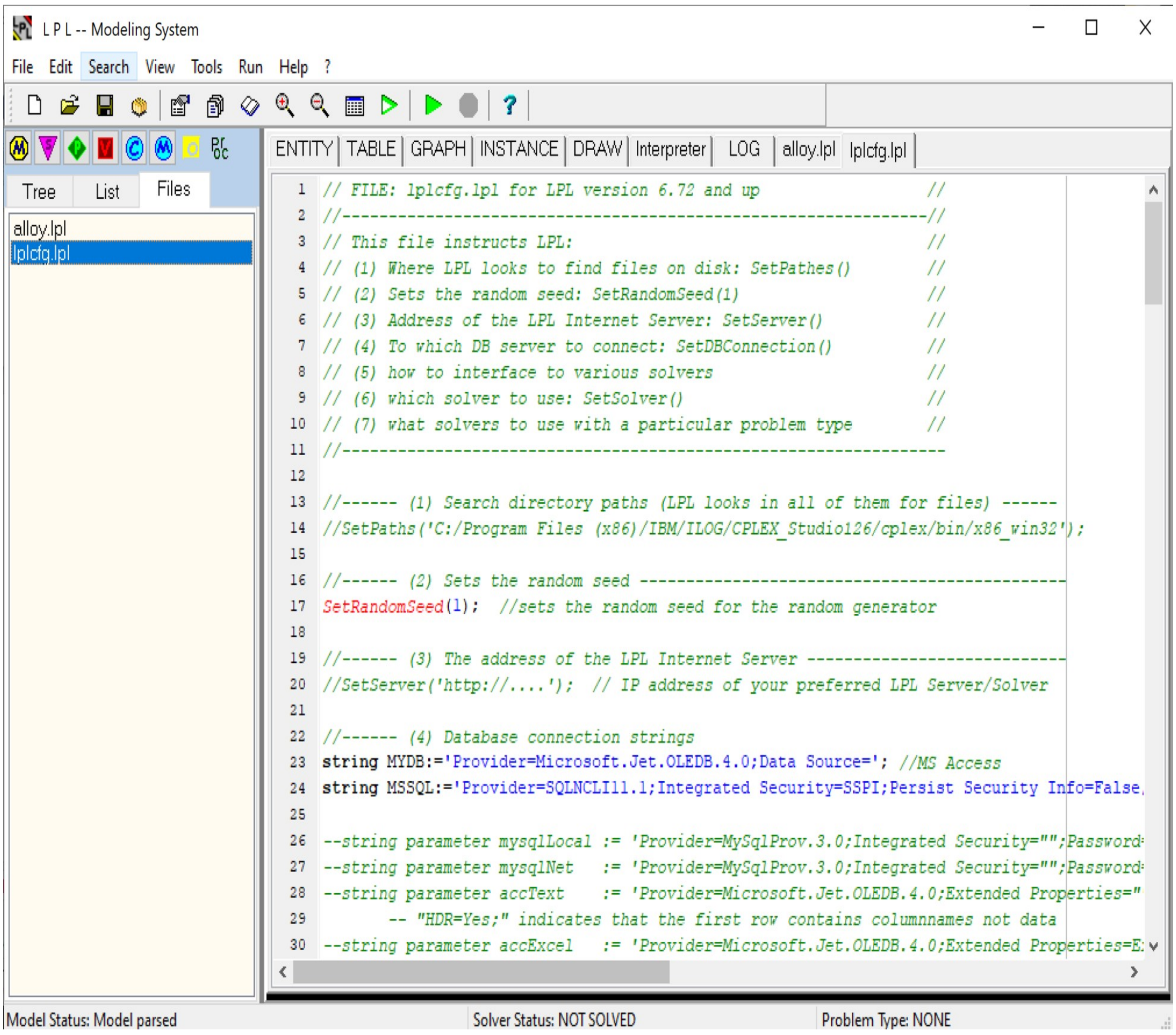
- define the directory paths where LPL looks for files by default.
- set a default random seed for generating random numbers.
- define database connectivity strings for different databases providers.
- set the IP address of various Internet LPL-Server/Solver.
- define “connection strings” that specify how LPL interfaces with various solvers.
- set the solver to be used.
- set the solvers matched to the different type of problems.

The file *lplcfg.lpl* is completely configurable by the user. He or she can add further instructions or remove everything that is not used. LPL does even not use that file: *lplcfg.lpl* could be removed entirely.

5. Scroll down within the file lplcfg.lpl in the LPLW program until you find the line

   ```
   --SetSolver(highsSol);
   ```

6. Remove the two dashes at the beginning of the line and save the file (use the menu item <File / Save File>). Make sure that *all lines after that line* begin with two dashes. You are done. From now on, LPL will solve the models with the HIGHS solver.
7. Try it: Choose the menu item <Run / Run Model> and you will notice that quickly a black window opened and closed, that was the HIGHS program that was called by LPL.


L P L -- Modeling System
File Edit Search View Tools Run Help ?
Tree List Files
alloy.lpl
lplcfg.lpl
ENTITY TABLE GRAPH INSTANCE DRAW Interpreter LOG alloy.lpl lplcfg.lpl

```
// FILE: lplcfg.lpl for LPL version 6.72 and up              //
//-----------------------------------------------------------//
// This file instructs LPL:                                  //
// (1) Where LPL looks to find files on disk: SetPathes()    //
// (2) Sets the random seed: SetRandomSeed(1)                //
// (3) Address of the LPL Internet Server: SetServer()       //
// (4) To which DB server to connect: SetDBConnection()      //
// (5) how to interface to various solvers                   //
// (6) which solver to use: SetSolver()                      //
// (7) what solvers to use with a particular problem type    //
//-------------------------------------------------------------

//------ (1) Search directory paths (LPL looks in all of them for files) ------
//SetPaths('C:/Program Files (x86)/IBM/ILOG/CPLEX_Studio126/cplex/bin/x86_win32');

//------ (2) Sets the random seed ---------------------------------------------
SetRandomSeed(1);  //sets the random seed for the random generator

//------ (3) The address of the LPL Internet Server ---------------------------
//SetServer('http://....');  // IP address of your preferred LPL Server/Solver

//------ (4) Database connection strings
string MYDB:='Provider=Microsoft.Jet.OLEDB.4.0;Data Source='; //MS Access
string MSSQL:='Provider=SQLNCLI11.1;Integrated Security=SSPI;Persist Security Info=False,

--string parameter mysqlLocal := 'Provider=MySqlProv.3.0;Integrated Security="";Password
--string parameter mysqlNet   := 'Provider=MySqlProv.3.0;Integrated Security="";Password
--string parameter accText    := 'Provider=Microsoft.Jet.OLEDB.4.0;Extended Properties="
       -- "HDR=Yes;" indicates that the first row contains columnnames not data
--string parameter accExcel   := 'Provider=Microsoft.Jet.OLEDB.4.0;Extended Properties=E
```

Model Status: Model parsed   Solver Status: NOT SOLVED   Problem Type: NONE

Figure 9: The file *lplcfg.lpl* was opened

The Gurobi solver

Several commercial solvers are linked to LPL through their dynamic link libraries. For example, the **Gurobi solver** (www.gurobi.com) comes with a dll-library on Windows that can be accessed directly. LPL implements the complete interface to Gurobi (with parameter setting and callback functionality).

To solve a model with the Gurobi solver you need to do the following:

1. Buy and download the Gurobi solver from www.gurobi.com. Be sure to have a valid licence key installed. (Academic uses is free of charge.)
2. Install it: Suppose that Gurobi will be installed in the folder '*c:\gurobi1000*' on your harddisk.
3. Identify where the dynamic link library '*gurobi110.dll*' is. (The last version was '*gurobi110.dll*'.) Normally this is in the folder '*c:\gurobi1100\win64\bin*' (or '*c:\gurobi1100\win32\bin*' for the 32-bit version).
4. Open alloy.lpl and lplcfg.lpl file within the LPLW program as before.
5. Find the entry `string parameter gurobiLSol` and verify the path and the

Gurobi version and modify appropriately.

6. In the same file (*lplcfg.lpl*) look for the line :
```
--SetSolver(gurobiLSol);
```
7. Remove the two dashes at the beginning of that line. Make sure that *all* lines after that line in the *lplcfg.lpl* file do begin with two dashes. Then save the file.
8. You are done! Running alloy.lpl now will be solved by Gurobi.

This is all you need to do to configure a predefined solver in LPL. The other solvers can be attached in the same way.

What you do not need to know at this moment is how a solver is interfaced to LPL. However, for the curious, here are some indications:

The interface to the Gurobi solver is specified by the interface string defined in *lplcfg.lpl* also as follows:

```
string parameter gurobiLSol :=
',,lib:gurobi110,gurobi.prm,,TimeLimit=60,,,,,,,,,,,LS:iLS:LP:MIP';
```

This instruction defines a string parameter. The content of this string is explained in the Reference Manual of LPL [see §9.2.1. The solver interface parameters (SIP)].

Basically it defines:

1. which dynamic link library to use (if you installed gurobi110, you should replace the string '*lib:gurobi???*' by '*lib:gurobi110*' probably);
2. which interface solver parameter file to use (*gurobi.prm*),
3. after how much time the solver should stop (*TimeLimit=60* (= 60 secs))
4. what kind of problem type the solver can solve (*LP;MIP;QP;iQP;QCP;iQCP* means that it can solve linear and linear mixed integer problems, as well as various quadratic problems).

In the same way the interface to the free *HIGHS* solver was defined in a parameter string as: follows

```
string parameter highsSol := 'm,h,highs.exe --options_file highs.clp
--time_limit 100  --solution_file \"%1.sol\" --model_file \"%1.mps\",,
highs.clp,threads=1,max2min,,%1.sol,8,24,13,%1.sol,66,
OPTIMAL,INFEAS,UNBOUND,LP;MIP';
```

Basically it defines:

1. that LPL should first create a mps-file before solving (*'m'*),
2. that the solver should write the results to a sol-file after finishing its work (*'h'*), in a way that LPL then can read the solution from that (text)file automatically.
3. it then should call the solver highs as an external process (*highs.exe*),

4. the other parameters specify how to guide the solver through parameters and how LPL should read the data from the sol-file. (all this is explicitly explained in the Reference Manual.

LPL is interfaced through about 24 parameters to any solver. These parameters can be studied for a dozens of solvers in the *lplcfg.lpl* file. If you do not use these solvers you might removed them all together.

The Internet Server/Solver

A particular solver is **the Server/Solver of LPL**. That is a solver "out there somewhere in the Internet", that can be configured in the same way as any other solver locally installed. The Internet Server/Solver does the same as any installed solver, except that you do not need to install any solver in your local machine. To use the Internet-Solver proceed as follows (you need to have at least an Professional Package of LPL – this is not free):

1. Open the LPLW program and the files *alloy.lpl* and the file *lplcfg.lpl* as before.
2. Within the file *lplcfg.lpl*, look for the line :
   --SetSolver(InterSol);
3. Remove the two dashes at the beginning of that line. Make sure that *all* lines after that line in the *lplcfg.lpl* file *do* begin with two dashes. Then save the file.
4. You are done! Running *alloy.lpl* now will be solved by the Internet-Solver!

To use the Internet-Solver you must buy at least an Professional Package. If you try to solve a model with the Free or Academic Package using the Internet-Server, you'll get an error message from LPLW as follows:

```
LPL error: 099 License Info: The installed package does not allow to run a module
```

An important advantage of the Internet-Server is that you do not care about what solver should be used. Independently of what problem your model is: a linear, a non-linear, a quadratic or any other model type, the Internet-Server will choose automatically the "right" solver, because LPL contains a "model-type-recognition-mechanism".

Directories and Paths

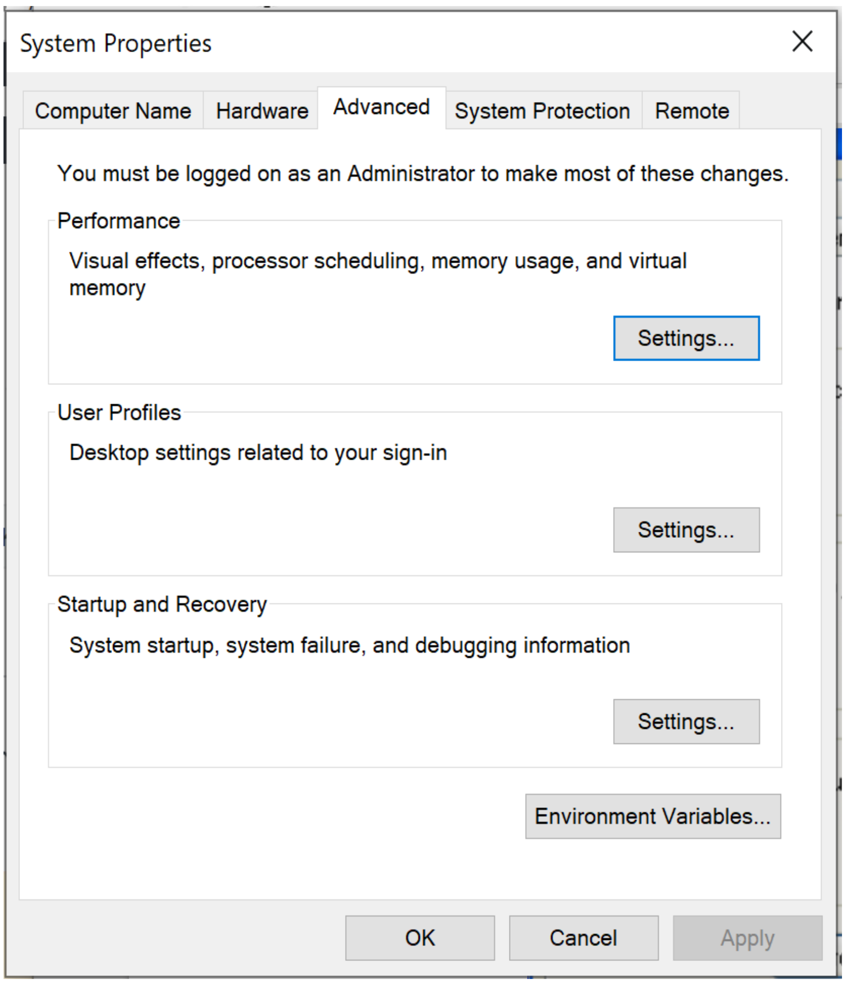
If LPL cannot find a file, then the LPL Paths are not correctly configured. LPL looks in a list of paths for any file (except *lplmsg.txt*). If you open a model file with LPL then the path to the model is always part of this list, the LPLEXE folder (see above) also is part of this list.

Furthermore, the user can add paths in an environment variable of Windows. In Windows, search for "env", it proposes "edit the system environment variables", choose then the tab <Advanced>. The following dialog box of Windows opens (Figure 10). Click on <Enviroment Variables>. Then click the button <new> on the <Systen Variables> part. In the field Variable name, enter '*LPLPATH*' and at the variable value part enter the path list you want to add to LPL. For example, adding

'*c:\gurobi1100\win64\bin;c:\cbc;*'

will add the two paths '*c:\gurobi1100\win64\bin*' and '*c:\cbc*' permanently to LPL, LPL shall look at these paths to find executables, libraries, or other files. Hence, you would not need to add the following instruction anymore in your *lplcfg.lpl* file (for example):

```
SetPaths('c:/gurobi1100/win64/bin;');
```


System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage, and virtual memory

Settings...

User Profiles

Desktop settings related to your sign-in

Settings...

Startup and Recovery

System startup, system failure, and debugging information

Settings...

Environment Variables...

OK | Cancel | Apply

Figure 10: Windows System Properties Dialog

In addition, the user can add further paths using the SetPaths('….') instruction as explained above. This instruction can be added to the *lplcfg.lpl* file (as above) or as instruction within an lpl model. There is a difference, however, of whether the paths are add by the environment variables '*LPLPATH*' or by a *SetPaths()* instruction. The environment variables are read at the very beginning of the run, while the *SetPaths()* instruction is only executed at the appropriate running time of a model.