Installation Guide for DP2PN2SolverV7

Holger Mauch

Eckerd College
mauchh@eckerd.edu

December 22, 2013
1 Purpose and Preparations

This installation and user guide describes what to do with the software you just downloaded as

DP2PN2SolverV7.zip

It will tell you how and where to deploy the files to your computer and how to invoke the two major modules DP2PN and PN2Solver.

1.1 Java Environment

Make sure that Java JDK 7 is installed on your system. (DP2PN2Solver might also work with earlier versions, but this has not been tested.) You may download it for free from http://www.oracle.com/technetwork/java/javase/downloads/index.html

DP2PN2Solver needs the compiler “javac”, so having merely a Java Runtime Environment (JRE) installed will not be sufficient. The compiler “javac” should be universally accessible, so make sure to include the directory where the binary of “javac” resides to your current path.

For example, on Windows XP you might add the following to your path by going through the sequence “Start—Control Panel—System—Advanced—Environment Variables” and then under “System variables” (or “User variables” if you are not the admin), edit the variable “path” by prepending the prefix `C:\Program Files\Java\jdk1.7.0\bin;` to the path. (We assumed that the binary `javac.exe` is located in `C:\Program Files\Java\jdk1.7.0\bin`.)

For example, on Windows 7 you might add the following to your path by going through the sequence “Start—Control Panel—System—Advanced system settings—Advanced tab—Environment Variables” and then under “System variables” (or “User variables” if you are not the admin), edit the variable “Path” by prepending the prefix `C:\Program Files\Java\jdk1.7.0\bin;` to the path. (We assumed that the binary `javac.exe` is located in `C:\Program Files\Java\jdk1.7.0\bin`.)

You can easily check whether the compiler “javac” is universally accessible by typing `javac` on the command line from any directory of your choice; if you always get a response that explains the usage of “javac” then the path is set correctly; if you get a response that “javac” is not recognized as a command, then the path is not set correctly. For example,
C:\Program Files\DP2PN2Solver>javac -version
javac 1.7.0

For more detailed instructions about installing the Java environment, please refer to http://docs.oracle.com/javase/7/docs/webnotes/install/index.html.

1.2 Deployment of the Files

Take the downloaded ZIP file and unzip it into a directory of your choice (the installation directory). Make sure that when you unzip and extract, you preserve the pathname information for the files (check your zip utility for that) — what you definitely do not want to happen is that after unzipping all files are in a single flat directory.

Throughout these instructions let us assume that the installation directory that contains the software is named DP2PN2Solver. (You may pick another name if you wish.) In Windows systems, the software might have been installed to C:\Program Files\DP2PN2Solver, for example. Note that this manual also applies to UNIX-like systems; the DOS/Windows specific separation character ‘\’ (backslash) will have to be read as ‘/’ (slash) instead, and the DOS/Windows typical path separation character ‘;’ (semicolon) has to be read as ‘:’ (colon).

After the unzip is completed the directory DP2PN2Solver contains the following five subdirectories

- bellman_net
- BN2SolverV7
- DP2BNv7
- ILP2gDPSpreprocessor
- pn2java

The directories bellman_net and pn2java are Java packages that contain parts of the implementation. While they might be of interest to developers, they need not concern ordinary users of DP2PN2Solver.

The directory DP2BNv7 contains the software for the module DP2PN; this is also the directory where you have to place your gDPS source files.

The directory BN2SolverV7 contains the software for the module PN2Solver; this is also the directory where you have to place your Bellman net source file, which you probably created using the DP2PN module.
2 The DP2PN Module

2.1 Preparations

Switch to the directory

DP2PN2Solver\DP2BNv7

Make sure you have your gDPS source file ready to be parsed in this directory. Remember, the gDPS source file is the one that contains all the information of your DP problem instance. Usually it needs to be created manually by the DP modeler. (With the exception of integer linear programming problems, for which a preprocessor exists.) You will notice that there are already some example gDPS source files present in this directory, ending with the suffix ".dp", e.g.

bst.dp
edpAlt.dp
...
wlv.dp

For details on how to create a gDPS source file, please refer to the gDPS user manual section and language specification provided in [2] and [1].

The following other files in this directory (ending in .class and in .txt) make up the DP parser and should not be modified:

Main1.txt
Main2.txt
Main3.txt
State.txt
CodeCreator.class
DPFEdata.class
DPSpecificationParser$1.class
DPSpecificationParser$JJCalls.class
DPSpecificationParser$LookaheadSuccess.class
DPSpecificationParser.class
DPSpecificationParserConstants.class
DPSpecificationParserTokenManager.class
FileCopy.class
Out.class
Out2.class

4
Out3.class
ParseException.class
SimpleCharStream.class
StreamGobbler.class
SymbolTableEntry.class
Token.class
TokenMgrError.class

2.2 Invocation of DP2PN

The gDPS2BN parser is invoked as follows, where the name of the gDPS
source specification file (e.g. mcm.dp) is provided as a command line argu-
ment; optionally, a name for the parser log file can be specified.

\texttt{java DPspecificationParser mcm.dp [parserLogFileName]}

As an alternative, on DOS/Windows systems the batch file \texttt{DP2BN.bat}
can simplify the invocation:

\texttt{DP2BN mcm.dp [parserLogFileName]}

2.3 Consequences of Invocation

This will create a directory for your problem instance, named DP2PNXXX
where XXX is the name given in the “NAME” section of the gDPS specifica-
tion, e.g. BST, MCM, etc. The contents of this new directory are explained
below; we will from now on assume that the name of the problem instance
is MCM.

Note that the name of the gDPS source specification file does not neces-
sarily need to match the name given in the “NAME” section of this gDPS
specification (e.g. MCM), but it seems good practice to match them anyway.

The parser log file is stored by default in

\texttt{DP2PN2Solver\DP2BNv7\dpParserLog.txt}

but the log file name can be changed to something else by specifying
\texttt{[parserLogFileName]} in the invocation launch.

Now look at the directory

\texttt{DP2PN2Solver\DP2BNv7\DP2PNMCM}
which contains some intermediate data files that might be useful for debugging, if you encountered a problem parsing or compiling your gDPS source file. Otherwise they are of no further interest to the normal user. If the gDPS source is parsed, compiled, and successfully translated into a Bellman net, then the subdirectory \texttt{MCMBN (XXXBN in general)} will hold the output, i.e. the Bellman net, and a log file of the Bellman net building process in:

\begin{verbatim}
DP2PN2Solver\DP2BNv7\DP2PNNCM\MCMBN
  MCMBN.csv
  DP2PNlog.txt
\end{verbatim}

The first file \texttt{MCMBN.csv} is the desired Bellman net, which can be fed into the PN2Solver software module.

This concludes the use of the DP2PN module. Starting from a gDPS source, we have produced a Bellman net. The next section deals with the use of the PN2Solver module, that automatically produces solver code from a Bellman net.

\section{The PN2Solver Module}

\subsection{Preparations}

The output of DP2PN can and should be used as the input for PN2Solver, so now manually copy or move the Bellman net you just produced (e.g. \texttt{MCMBN.csv}) to the directory

\begin{verbatim}
DP2PN2Solver\BN2SolverV7
\end{verbatim}

which already contains twelve \texttt{.class} files that make up the Bellman net parser. So before you invoke the Bellman net parser, the directory will look like this:

\begin{verbatim}
DP2PN2Solver\BN2SolverV5
  BNSpecificationParser.class
  BNSpecificationParserConstants.class
  BNSpecificationParserTokenManager.class
  Out.class
  Out2.class
  Out3.class
  Out4.class
  Out6.class
  ParseException.class
\end{verbatim}
There might already be additional Bellman net files provided in this directory; they all look like XXXBN.csv

### 3.2 Invocation of PN2Solver

After establishing the Bellman net file `MCMBN.csv` in `DP2PN2Solver\BN2SolverV7`

and changing to this directory we are ready to invoke the BN2Solver module; we provide the name of the Bellman net file `MCMBN.csv` as a command line argument as follows (do not omit the classpath information, otherwise you will get an error message):

```
java -classpath .; .. BNspecificationParser MCMBN.csv
```

As an alternative, on DOS/Windows systems the batch file `BN2Solver.bat` can simplify the invocation:

```
BN2Solver MCMBN.csv
```

### 3.3 Consequences of Invocation

This produces a directory named `MCMSolverCode` that contains three solvers where each solver uses a different technology to produce a final result. In `DP2PN2Solver\BN2SolverV7\MCMSolverCode`

the first solver is `MCM.csv`

This is the spreadsheet solver; load this file into a spreadsheet application such as Microsoft Excel, update the cells, and you get the solution to the DP problem instance.

The second solver is `MCM.xrn`
This is the Petri net solver; import this file (which is in PNML standard format) into a PN application such as Renew, simulate the net, and you get the solution to the DP problem instance.

The third solver is the Java solver

MCMJavaSolver.java

The Java Solver file is automatically compiled to

MCMJavaSolver.class

and executed by the invocation from section 3.2 and the resulting solution tree of the problem instance can be found in the file

MCMSolutionTree.txt

This file is the desired output representing not only to optimal function value but also the optimal decision policy for the DP problem instance!

The file

PN2SolverLog.txt

contains a log of the transformation process from a Bellman net to the solver files.

If for some reason you would like to trace the automated compilation and execution of the JavaSolver have a look at the file

runJavaSolver.bat

which contains the necessary steps to compile and launch the JavaSolver:

javac -classpath ..\.\.. MCMJavaSolver.java
rem for running, do not forget to include the current directory
java -classpath ;;\..\.. MCMJavaSolver

References
