# Installation Manual

## Table of contents

## 1. Requirements

You must have a valid install of Java 1.4.0 or greater. By default, we suppose that java is installed in /usr/local/jdk1.4.0 but if not, you should make sure that your environment variable PATH is correctly set.

Ilog Cplex 7.5 or greater is optional.

Mascopt now uses ant (http://ant.apache.org/) to compile the projects. It offers several advantages and in particular it eases the use of eclipse and enables a greater customisation of the project. Alternatively, you can use the provided Makefile needing GNU make and some UNIX tools like grep, tr, mkdir, ... are included. These scripts needs a standard Linux system. Otherwise you have to compile yourself the source files.

A compiled version of Xerces (http://xml.apache.org/xerces2-j/index.html) and Skinlf (http://www.l2fprod.com/) are included. If you just download the mascoptLib.jar file, then Xerces and Skinlf are not included.

## 2. How to compile ?

### 2.1. The ant way Part 1 - Principles and command line

The ant configuration files are build.xml and javadoc.xml, but you should not need to edit them.

This way you still have to specify the correct paths for your system, but first you must type:

```
ant init
```

That way you create a file named mascoptDev.properties where the default configuration is written. This configuration is done accordingly to the system of INRIA Sophia Antipolis. If you reside elsewhere and need to change things, please change them in the file mascoptdev.properties.

Now that the configuration is set for your site, we can compile the project. In order to compile everything, just type

```
ant
```

the result is exactly the same as make. It cleans the class directory and then recompiles everything. In order to accelerate things and compile only the files that have changed, you can type:

```
ant build
```

As previously, when compiling, the .class files are stored in the subdirectory classes. As this directory is present in the CLASSPATH, you can launch a program from anywhere.

ant enables also the generation of the documentation via the call of javadoc. All you have to do is to type

```
ant javadoc
```

and the documentation will be generated in the directory docs/mascoptDev.

### 2.2. The ant way Part 2 - ant and eclipse

One of the main advantages of ant is that you can call it from a number of IDEs and among them, eclipse (http://www.eclipse.org/) . To this end, you have to define how to call ant and which target to use.

Go into the menu Run->External tools->External tools... and after selecting "Ant Build" click on "New". Let's create a call to the target build of the ant file. Give a name to the configuration you are about to create (for example build), choose the "Location" which is the file build.xml, the base directory is the mascoptLib directory and finally, in the "Target" tab, uncheck all and check build. Then you are done. To call the target build, simply choose Run->External Tools->build and the compilation proceeds. If you want to execute other targets, configure them the same way.

## 3. The Makefile way

The environement variable CLASSPATH must be changed to allow java to find the mascopt.jar file and the sources file. The PATH and LD_LIBRARY_PATH must also be changed. When done, you can compile a single java file with javac or all files with the Makefile. You have to execute, in the mascoptDev directory the following:

```
# To find the class compiled by the user
export CLASSPATH=.:`pwd`/classes

# To find MascoptLib
export CLASSPATH=$CLASSPATH:`pwd`/jar/mascoptLib.jar

# To find CPLEX (which is optional)
export CLASSPATH=$CLASSPATH:/usr/local/cplex75/lib/cplex.jar
export
LD_LIBRARY_PATH=/usr/local/cplex75/bin/i86_linux2_glibc2.1_egcs1.1:$LD_LIBRARY_PATH

# Path to find Java and Javac
export PATH=/usr/local/jdk1.4.0/bin:$PATH
```

A script have been written to perform the changes. It is called SETENV. Check that the paths specified in this file are correct and then type:

```
source SETENV
```

After that, to compile all the source tree, the samples and tests, you can do:

```
make
```

When compiling, the .class files are stored in the subdirectory classes. As this directory is present in the CLASSPATH, you can launch a program from anywhere.

## 4. Launch a program

All the .java file, containing a main() function are situated in the directory tests, samples or launch. In fact, the scr only contains classes providing a service to the user, for example a class wich execute an algorithm given a graph. When you want to test your algorithm written in class MyAlgo, you write a static main() function wich create a new object of class MyAlgo and then call some functions on it.

This is the reason wich lead us to separate the source code of the library, only constitued of classes without main and the test or sample programs. Nevertheless, the Makefile compile the four sub-directories.

To launch a program you wrote you just do the following:

```
java ASampleProgramThatYouWrote
```

For example, you can try:

```
java Creation
```

## 5. Working with CVS at INRIA

cvs is usefull to let a group of developpers to program. When you use mascoptDev, the mascopt.jar library can change of version and can be updated in your local project without breaking your work. And then, all the source code of mascoptDev can change and be updated by other users. In the following some usefull commands that have to be done in the mascoptDev directory.

To get the last mascoptDev version (overwrite all the local modifications !):

```
cvs checkout mascoptDev
```

To get the last modification in the mascoptDev project, keeping your local changes:

```
cvs update mascoptDev
```

To put your modifications into the cvs repository (3 ways):

```
cvs commit mascoptDev
cvs commit
cvs co
```

Note that it may happen that the files you change have already be changed by an other user. In this case, you will say a flag "M" indicating that CVS tryed to merge the two modifications. If cvs failed to merge the two files, it will write the two possibilties into the file and then you have to correct yourself.

When some new directories or new file are created or removed:

```
cvs add file
cvs remove file
```

To compare your local version to the cvs repository:

```
cvs diff mascoptDev
```

You can also have a look to the CVS manual (pdf)

## 6. Structural remark

Mascopt is divided in three different parts:

- mascoptLib.jar which contains the mascoptLib code, the base library to process graphs
- src/mascoptDev which contains the user environement to develop new algorithms
- src/mascoptCplex which contains the user environement to develop new algorithms with Ilog Cplex

Note that when some algorithms are quite tested and works fine, we can integrate them into the mascoptLib part of the code. But in general, the user always works in the mascoptDev project.

## 7. Use Mascopt with .jar file

If you use Mascopt as a java library you call from your program, you just have to put mascoptLib.jar in your CLASSPATH. However, it can be hard to develop with the .jar without the sources, because you may need the javadoc, built from the sources.