MILPFlow version 26 documentation tutorial

Last update: 16/09/2014

Author: Lucio Agostinho Rocha

1. What is MILPFlow?

This MILPFlow version is an open source software to create and deploy OpenFlow flow_mod rules. We develop this toolset to generate computational models of data centers to solve routing problems, and to establish data paths between servers according to the solutions of these models. MILPFlow provides a simplified way to describe and solve these problems. Additionally, it establishes data paths before data flows be sent through the network. As a consequence, MILPFlow contributes to reduce the overhead to discover network routes among hosts of data centers. Currently, it is a renewed version of REALCloudSim project [1]. New topologies can be defined by the user using a BRITE description [2].

2. How this simulation works?

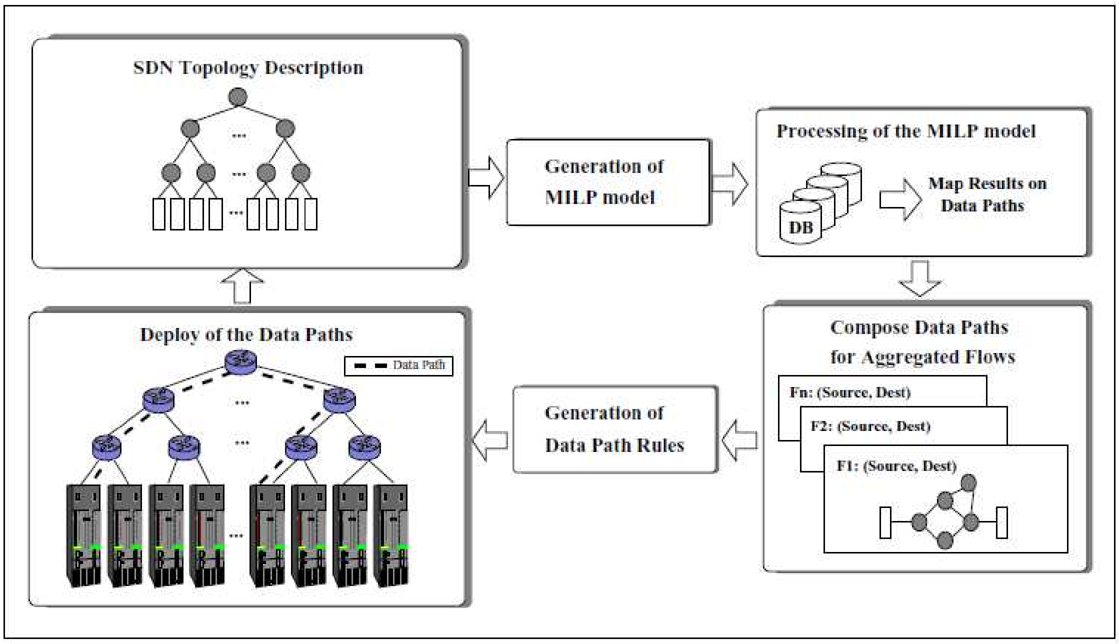


Fig. 1 – MILPFlow methodology.

MILPFlow receives a set of parameters provided by the user. Then, it generates a model in Mixed Integer Linear Programming (MILP) that is solved by the Lingo solver [3]. Figure 2 illustrates the main MILPFlow components in UML notation:

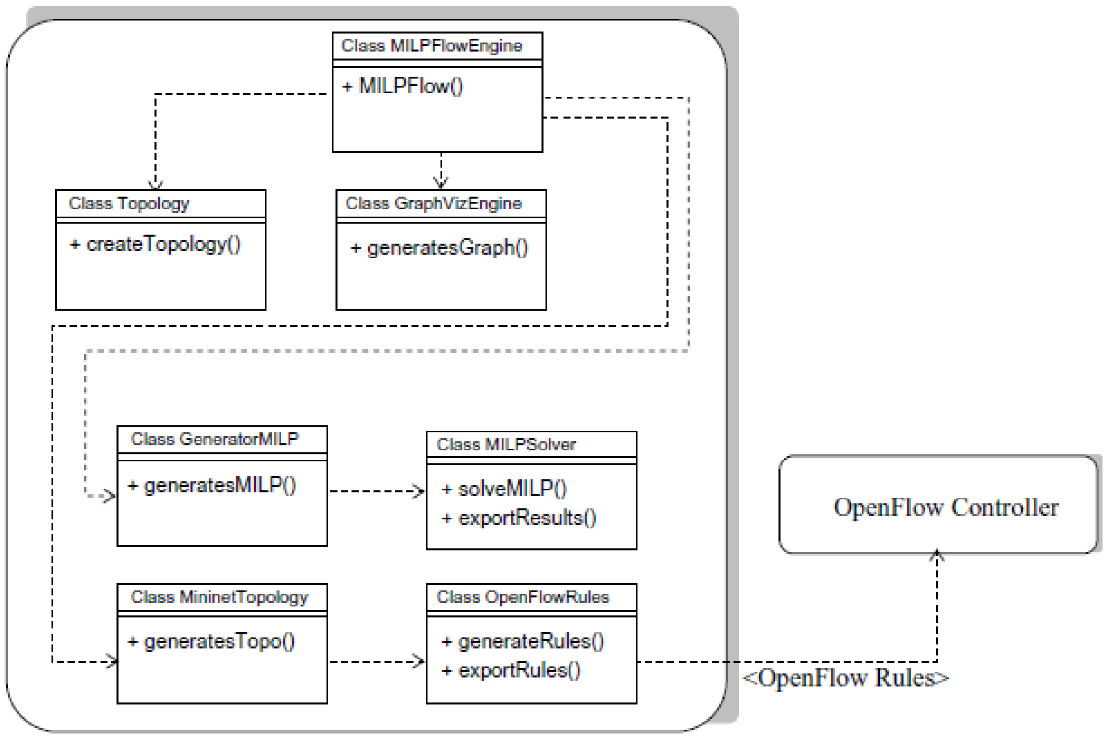


Fig. 2 – MILPFlow Components.

- MILPFlow Engine is the main component that controls the interactions with all other components;
- GraphViz Engine is the component that creates the graphic files to visualize the generated data paths;
- Topology Descriptor is the component that reads the variables of the simulation, such as topology nodes, links, bandwidth, number of servers, number of switches and traffic matrix;
- GeneratorMILP is the component that produces the MILP model to be solved with a MILP solver;
- MILPSolver is the component that invokes the MILP solver;
- MininetTopology is the component that performs the mapping between the MILP results and Mininet network topology;
- OpenFlowRules is the component that produces the executable batch file with dpctl commands and/or HTTP REST commands from the MILP results.

Note: this description is purely informative. See the code of the project for more details.

3. Requirements

JDK1.6+ and the Lingo solver. I suggested also use a Linux distribution (Ubuntu 12.04+ is recommended). Other recommendation is use a recent version of Eclipse IDE to simplify the compilation.

4. How to import this project in the Eclipse?

1) Download the newest version of MILPFlow from [4]:

```
# git clone https://github.com/milpflow/milpflow
```

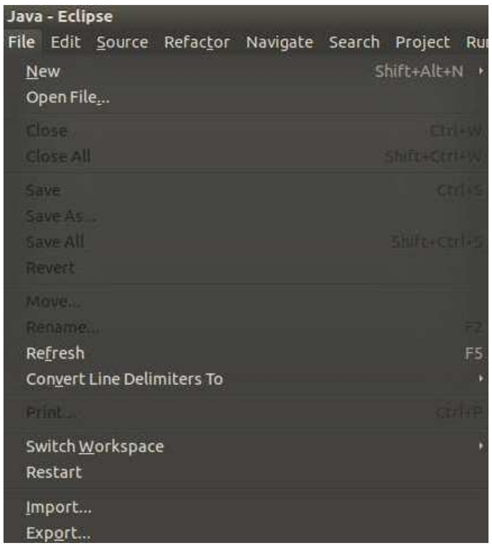
2) Copy the milpflow project folder to your Eclipse IDE workspace. For example:

```
$ cd /usr/local/src/workspace
$ tar xvfz MILPFlow-xyz.tgz
```

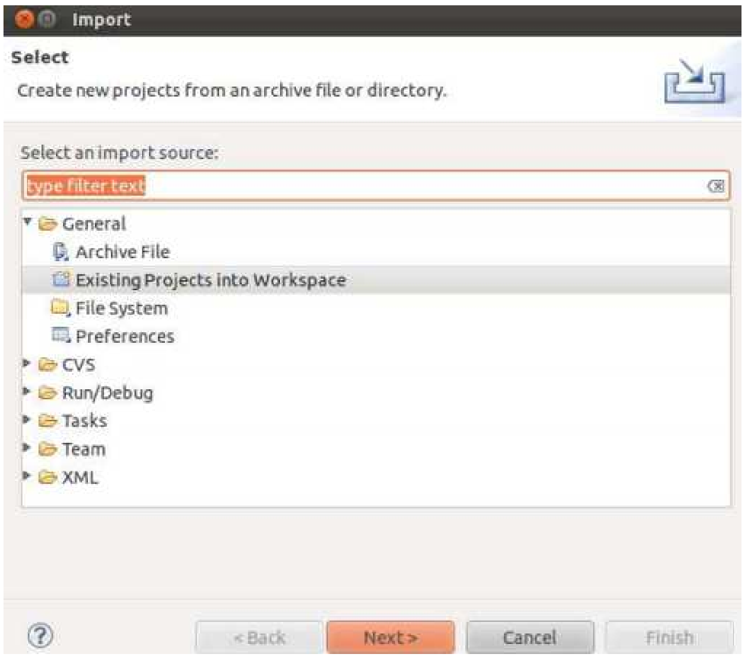
3) Create a symbolic link:

```
$ ln –s MILPFlow-xyz MILPFlow
```

4) Inside Eclipse IDE, import the new project. File → Import…



5) Choose: Existing Projects into Workspace. Click Next.


Import

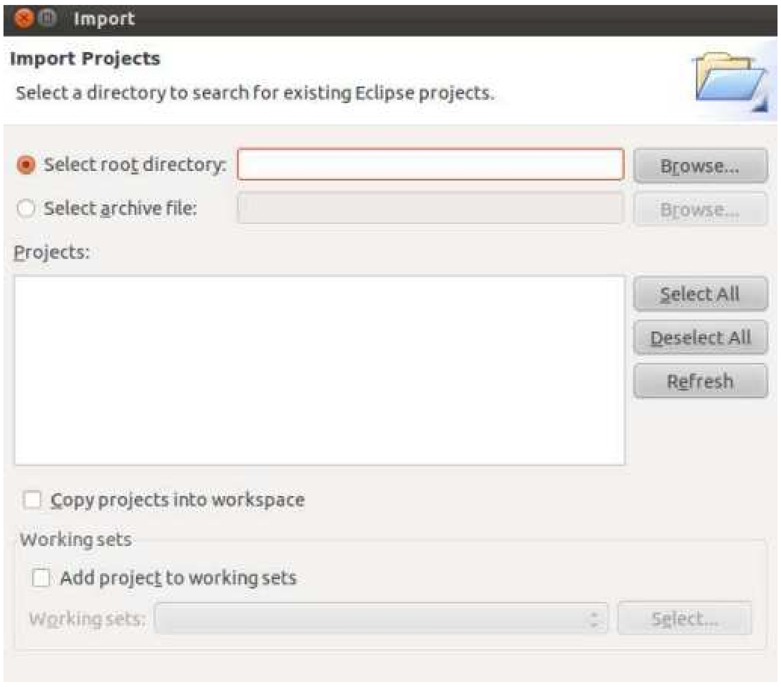
Select

Create new projects from an archive file or directory.

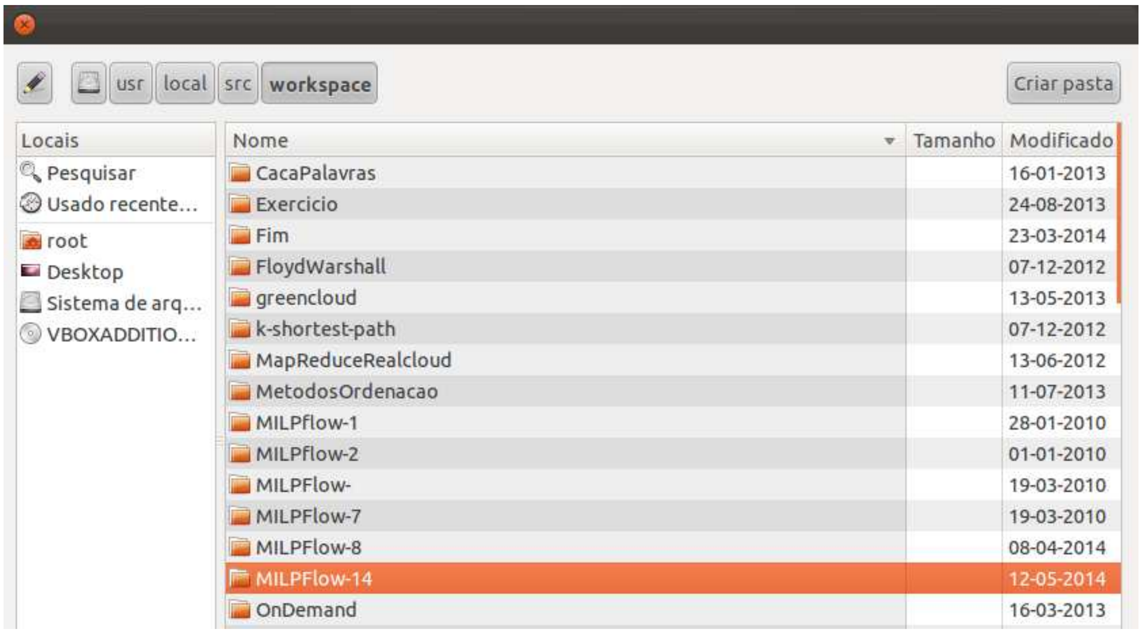
Select an import source:

type filter text

6) In the Import window, choose Browse…

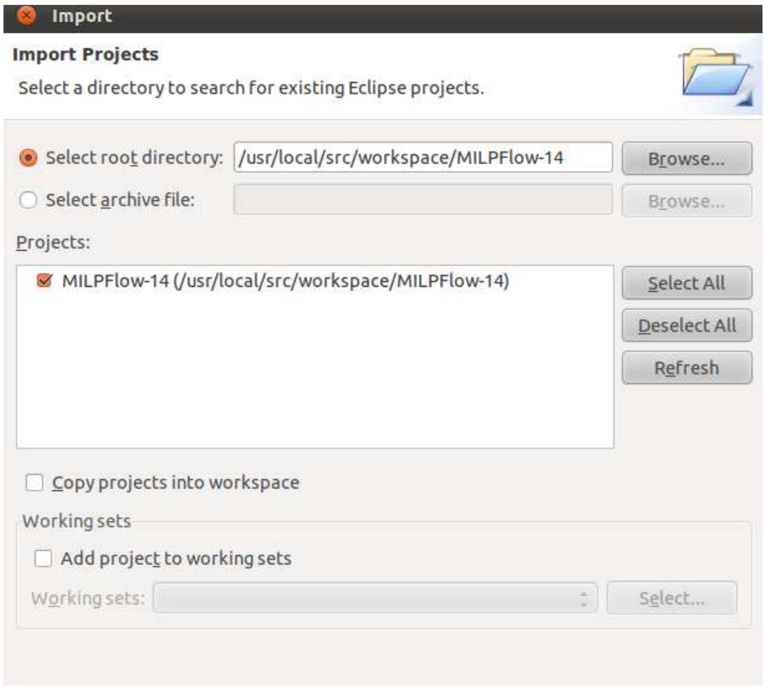


7) Select the MILPFlow folder, and tip ENTER twice.



Nome	Tamanho	Modificado
CacaPalavras		16-01-2013
Exercicio		24-08-2013
Fim		23-03-2014
FloydWarshall		07-12-2012
greencloud		13-05-2013
k-shortest-path		07-12-2012
MapReduceRealcloud		13-06-2012
MetodosOrdenacao		11-07-2013
MILPflow-1		28-01-2010
MILPflow-2		01-01-2010
MILPFlow-		19-03-2010
MILPFlow-7		19-03-2010
MILPFlow-8		08-04-2014
MILPFlow-14		12-05-2014
OnDemand		16-03-2013

8) Observe that MILPFlow project was selected. Click Finish.



Select a directory to search for existing Eclipse projects.

Select root directory: /usr/local/src/workspace/MILPFlow-14

Select archive file:

Projects:

MILPFlow-14 (/usr/local/src/workspace/MILPFlow-14)

Copy projects into workspace

Working sets

Add project to working sets

Working sets:

9) Done! Your project was imported in Eclipse. The main file is MILPFlow.java.
This file is in folder: /src

5. How can I run simulations with MILPFlow?

On this step, we assume that you already install Lingo software in your $HOME folder. Look at 'src/ConfigSetting.java' file if you provide a different path for both softwares.

MILPFlow output its results in the terminal console. A better performance is obtained if simulations were performed in the terminal (not inside Eclipse). But is recommended to keep Eclipse open AND run simulations in a terminal.

1) Keep your Eclipse IDE open to perform changes, compilations and verify errors;
2) Copy a template topology to your home folder:

```
$ cp MILPFlow/modeloLingo_1000serv_500vm.brite \
        $HOME/ modeloLingo_1000serv_500vm.brite
```

(Note: modeloLingo_1000serv_500vm.brite is a topology model of an example of Fat-Tree network topology. You MUST copy this file to your $HOME folder)

3) Configure (if necessary) the many input parameters, only in the src/ConfigSettings.java file.

4) Open a terminal and run your simulation:

```
$ cd /usr/local/src/workspace/MILPFlow
$./executar_linha_comando.sh
```

6. How can I evaluate the results?

After your simulation, a set of files beginning with modelo_* will be generated in your $HOME folder. We describe them as follows:

- modelo*.brite: original topology model provided by the user;
- modelo*.datacenter: modified topology model automatically generated by MILPFlow engine;
- modelo*.lg4: automatically generated Lingo model;
- modelo*.lgr: result of solving Lingo model;
- modelo*.py, modelo*.sh: scripts used to simulation of virtual switches with Mininet;
- modelo*GraphViz*.pdf: a beautiful graphic that shows the data paths between the used links of the ToRs in the current topology. We illustrate an example in Fig. 3:

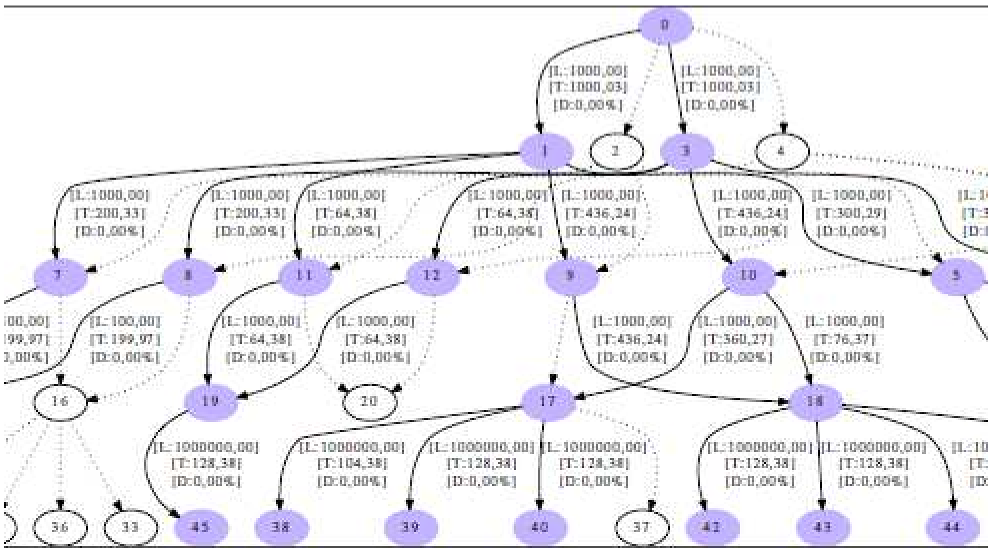


Fig. 3: Partial topology of the data traffic among the ToRs in the current Fat-Tree topology.

7. How can I change the topology?

We provide examples of Fat-Tree, VL2 and BCube topologies in the /src folder of MILPFlow project. You can follow this examples to simulates you own topologies. But:
Important: ToR nodes are only inserted in access switches (A_SWITCH) nodes described in the topology file. Your model can corrupt, or be infeasible, if this assumption was not considered. Look at the topology files examples and note the difference between A_SWITCH nodes, AS_NONE, and CORE_SWITCH nodes. Also, the order of links should be from lower layers to up layers (Example: node 2 → node 1, and not node 1 → node 2).

8. How can I deploy OpenFlow Paths with MILPFlow?

Important: We assume that you have already the Ryu controller installed and running properly.

- Step 1: Solve the MILP model of the data traffic:

```
# cd MILPFlow
# ./executar_linha_comando.sh
```

- Step 2: Start the topology for Mininet:

```
# ./modeloMininet_1000serv_500vm_ryu.py
```

- Step 3: Start the Ryu REST controller:

```
# cd ryu; ryu-manage --verbose ryu/app/ofctl_rest.py
```

- Step 4: Deploy the paths:

(Note: if some rule was not deployed (when occur HTTP 404 in 'ofctl_rest.py', instead HTTP 200), it is necessary to restart 'ofctl_rest.py', and run this step again).

```
# chmod +x modeloLingo_1000serv_500vm_regrasOpenFlowRyu.sh
# ./modeloLingo_1000serv_500vm_regrasOpenFlowRyu.sh
```

- Step 5: Try the connectivity between the hosts: (Note that only the paths between the hosts solved by MILPFlow will be connected).

```
mininet> h18 ping h29
```

9. Something is wrong and I can't get running my model. What can I do?

This project is collaborative software, and you are invited to contribute with it with documentation, improvement of the code, commentaries, doubts, and so on. Take a look into the code, and re-read this documentation. If something can't be running as it should, do not hesitate in contact the author by e-mail: outrosdiasvirao (at) gmail (dot) com.

References

[1] Rocha, L. A. et al., "A Bio-inspired Approach to Provisioning of Virtual Resources in Federated Clouds," *IEEE DASC,* 2011.

[2] A. Medina and e. al., "BRITE: Universal Topology Generation from a User's Perspective," *Available at: http://www.cs.bu.edu/brite/user_manual/.*

[3] LINDO SYSTEMS, "LINGO 13.0 - Optimization Modeling Software for Linear, Nonlinear, and Integer Programming," *Available at: http://www.lindo.com.*

[4] L. A. Rocha, "MILPFlow: An Open Source Project to Simulate VM Placement of Data Centers," <https://sourceforge.net/projects/milpflow> 2014. [Online].