Guide on UrbanSim usage of the travel model plug-in

Thomas W. Nicolai, Kai Nagel

November 22, 2011

Note that the current travel model plug-in implementation is applicable for PSRC[1] parcel, Seattle parcel and Zurich parcel scenario.

Please check for the latest version of this user guide in the subdirectory *opus_matsim/docs* in your opus source directory. Some of the instructions will change until the travel model plug-in final release.

1 Prerequisites

You must have installed UrbanSim, before getting started with the travel model plug-in. The following provides an entry point to install UrbanSim and continues with installation instructions for additional software packages required by the travel model plug-in.

1.1 Hints for installing UrbanSim

To install UrbanSim please follow the *UrbanSim Downloads and Installation Instructions* on `http://urbansim.org/Download/`.

When installing OPUS and UrbanSim manually[2] please make sure to get the source code for the latest stable release as described in *Downloading Sample Data and Source Code* on:

`http://urbansim.org/Download/DownloadingSampleDataAndSourceCode`.

Windows users using the installer are getting the source code and data automatically.

Finally make sure that all UrbanSim environment variables, meaning *OPUS_HOME*, *OPUS_DATA_PATH* and *PYTHONPATH*, are set as described in the installation instructions, see `http://urbansim.org/Download/SixtyFourBitMachines` for Windows, `http://urbansim.org/Download/MacintoshInstallation` for Mac or `http://urbansim.org/Download/LinuxInstallation` for Linux.

Note for users installing UrbanSim manually: For using the travel model plug-in it is sufficient only to install the *required* Python packages, i. e. numpy, scipy, lxml, sqlalchemy and elixir.

[1] Puget Sound Region Council

[2] Windows users may require an additional svn client, e. g. tortoisesvn that is available for free at `http://tortoisesvn.tigris.org/`

1.2 MATSim4UrbanSim prerequisites

In addition to the UrbanSim installation the MATSim travel model plug-in requires the following software installed:

- **Java JDK 1.6 or newer**: Download and install the newest version of the *Java SE Development Kit (JDK)* for your operating system from `http://www.oracle.com/technetwork/java/javase/downloads/index.html`.

  Make sure adding Java's */bin* directory to the *PATH* environment variable.

- **Python XML Schema Bindings (PyXB)**: Download the PyXB distribution file from `http://sourceforge.net/projects/pyxb/` and extract it to a convenient place. Windows users may use Win-Rar to extract tar or gz files, which is available for free at `www.win-rar.com/download`.

  Open a command prompt (Windows) or a terminal (Mac, Linux), go into the extracted directory and type

  ```
  python setup.py install
  ```

  to install PyXB (this may requires administrator or root privileges).

2 MATSim4UrbanSim installation

This section describes how to install MATSim4UrbanSim. For more information about MATSim please visit `www.matsim.org`.

2.1 Automatic installation

Make sure to have the Python lib directory included to the *PYTHONPATH*. This is the directory that contains the site-package directory that is already included in the *PYTHONPATH*. The lib directory should be something like

```
C:\Python2.6\Lib\
```

for Windows,

```
/Library/Frameworks/Python.framework/Versions/2.6/lib/
```

for Mac or

```
/usr/lib/python2.6/
```

for Linux (Ubuntu).

To install MATSim4UrbanSim open command prompt (Windows) or a terminal (Mac, Linux) and navigate to *opus_matsim/configs* in the opus source directory. Than type

```
python install_matsim4urbansim.py
```

This creates the subdirectories *matsim4opus/jar*, loads required MATSim executables and libraries and configures them. After the installation the file/directory structure should look something like Figure 1.

To test whether the MATSim4UrbanSim installation was successful follow the instructions described in Section 2.3.

Note: The installer will replace jar-files and libraries from a previous installation.

2.2 Manual installation

In case that the automatic installation does not work follow these instructions:

- Create the following directory structure in OPUS_HOME:

  `OPUS_HOME/matsim4urbansim/jar`

- Download the following files from `http://www.matsim.org/files/builds/` into the jar directory:
  - MATSim_rXXXXX.jar (where XXXXX refers the current revision)
  - MATSim_libs.zip
  - matsim4urbansim_v1-0.4.0-SNAPSHOT-rXXXXX.zip (where XXXXX refers the current revision)
- Rename MATSim_rXXXXX.jar into "matsim.jar".
- Extract the zip files MATSim_libs.zip and matsim4urbansim_v1-0.4.0-SNAPSHOT-rXXXXX.zip. After that the zip files can be removed.
- Rename the directory matsim4urbansim_v1-0.4.0-SNAPSHOT-rXXXXX into "contrib". Than navigate into contrib and rename the jar-file matsim4urbansim_v1-0.4.0-SNAPSHOT.jar into "matsim4urbansim.jar".

Be careful when renaming files or directories (i. e. make sure that everything is written in lower case and check for spelling errors). After the installation the file/directory structure in the *matsim4opus* directory should look something like Figure 1. To test whether the MATSim4UrbanSim installation was successful follow the instructions described in Section 2.3.

2.3 Test your MATSim4UrbanSim installation

To test your installation open a command prompt (Windows) or a terminal (Mac, Linux) and navigate to *opus_matsim/tests* in the opus source directory (PYTHONPATH).Then type

```
python travel_model_test.py
```

This starts a test scenario. If the test completes without errors, your travel model plug-in should be working.

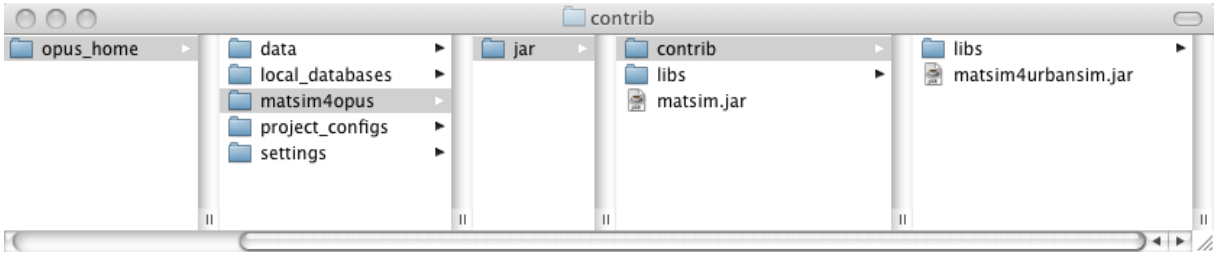

Figure 1: After the installation the *matsim4opus* directory should contain the depicted files and subdirectores.

3 Using MATSim for UrbanSim

This section aims to explain the MATSim travel model plug-in at the example of the Seattle_parcel scenario. **In order run Seattle_parcel with MATSim the following steps are necessary**: Download the "matsim.zip"-file from `http://matsim.org/extensions/matsim4urbansim_v1` and unzip the file into the OPUS_DATA/seattle_parcel/base_ye directory. After that your seattle_parcel base year cache should look like Figure 2. The zip-file contains a road network and a plans-file, used by MATSim.

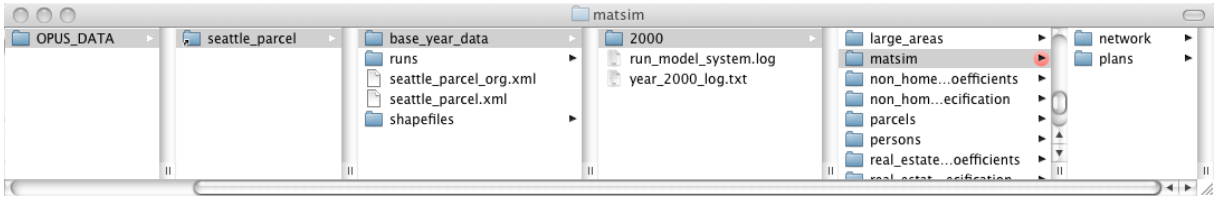

Figure 2: Put the unziped "matsim" directory, including a road network and a plans-file, into your seattle_parcel base year cache in order to run MATSim as a travel model for the Seattle_parcel scenario.

In a recent effort the MATSim configuration is embedded into UrbanSim. This enables the MATSim (travel model) plug-in in UrbanSim and provides all necessary or basic options to run MATSim via the OPUS GUI. To enable the MATSim plug-in requires the *travel_model_configuration* section in the UrbanSim configuration as depicted in Figure 3. Sample configurations, including the travel model configuration section, can be found for the Seattle_parcel (seattle_parcel.xml) and PSRC_parcel (psrc_parcel.xml) scenario in the opus source directory at *opus_matsim/configs*.

The following subsection explain how to configure the travel model plug-in at the example of the Seattle_parcel scenario.

3.1 Travel Model configuration options

This sections explains step by step the MATSim configuration options provided by the travel model plug-in.

Launch the OPUS GUI and open the Seattle_parcel sample configuration located at *opus_matsim/config/seattle_parcel.xml* in opus source directory. Switch to the *Models* tab to get to the *travel_model_configuration* section as shown in Figure 4. The following options are available:

```
<travel_model_configuration type="dictionary">
  <models type="selectable_list">
    <selectable name="opus_matsim.models.get_cache_data_into_matsim" type="selectable">True</selectable>
    <selectable name="opus_matsim.models.run_travel_model" type="selectable">True</selectable>
    <selectable name="opus_matsim.models.get_matsim_data_into_cache" type="selectable">True</selectable>
  </models>
  <matsim4urbansim type="dictionary">
    <sampling_rate type="float">0.01</sampling_rate>
  </matsim4urbansim>
  <matsim_config type="dictionary">
    <common type="dictionary">
      <matsim_network_file type="file">data/seattle_parcel/base_year_data/2000/matsim/network/psrc.xml.gz</matsim_network_file>
      <last_iteration type="integer">1</last_iteration>
    </common>
  </matsim_config>
  <years_to_run key_name="year" type="category_with_special_keys">
    <run_description type="dictionary">
      <year type="integer">2001</year>
    </run_description>
  </years_to_run>
</travel_model_configuration>
```

Figure 3: Adding the *travel_model_configuration* section into an UrbanSim configuration enables MATsim plug-in and configuration options within OPUS GUI.

- **Models**: The models section contains tree models integrating MATSim into UrbanSim:
  - *Get_cache_data_into_matsim* generates input data for MATSim and stores it a specified location.
  - *Run_travel_model* executes MATSim.
  - *Get_matsim_data_into_cache* imports the results of the traffic simulation for the next UrbanSim iteration.

  By default all models are enabled. Only disable models if you know what you are doing.

- **MATSim4UrbanSim**: This section contains options concerning the interaction of both simulation models MATSim and UrbanSim.
  - The *sampling_rate* determines the percentage of considered travellers for a MATSim run. 0.01 means that only one percent of travelers are considered for the traffic simulation. This option allows to speed up computations on the MATSim side, e. g. during testing a scenario.

    Note that low sampling rates cause some peculiarities in terms of realism. In this situation results are useful for sketch planning only, not fore quantitative analysis. Higher sampling rates need more ram and hard drive space.

- **MATSim_Config**: The *common* subsection provides some basic configuration option for MATSim.
  - The *matsim_network_file* points, as the name implies, to a road network in MATSim format. This expects a relative path to the network file located in the OPUS_HOME directory.
  - Determine the number of MATSim iterations with the item *last_iteration*. In MATSim iterations start at zero.

- **Years_To_Run**: This defines the years in which the travel model should run.

  Adding additional years requires to edit the configuration file directly, e. g. with a xml editor, within the *year_to_run* section in the travel_model_configuration. Make sure that each year you are adding is surrounded by the run_description tags like this:

```
<run_description type="dictionary">
     <year type="integer">2002</year>
</run_description>
```

All configuration options can be easily edited in the OPUS GUI by clicking on the value or check box on the right hand side.

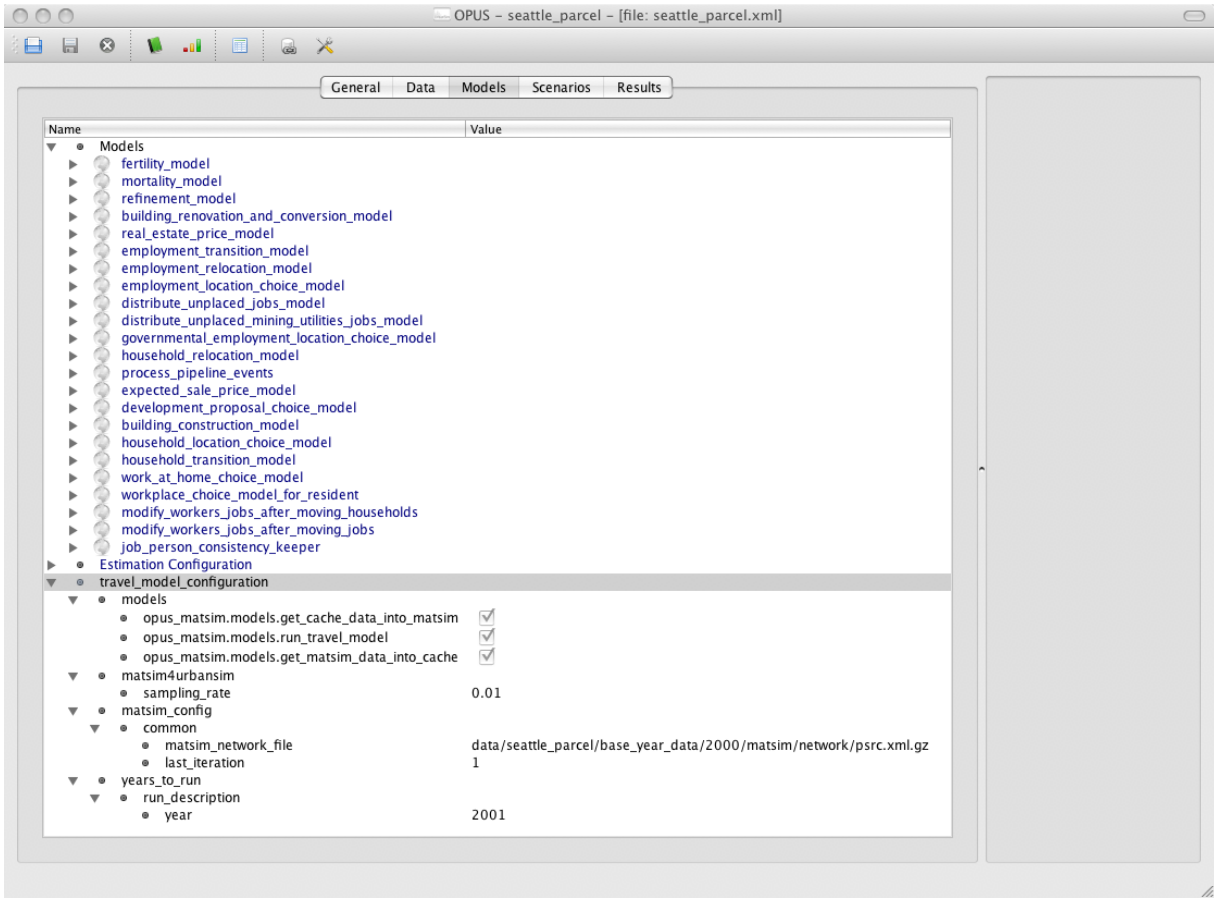

Figure 4: Configuring MATSim via the *travel_model_configuration* section in OPUS GUI.

4 Limitations

The Java virtual machine (VM) can't allocate more than 1.5 GB on Windows systems, no matter how much RAM is available in your computer. For this reason the travel model plug-in runs MATSim with 1.5 GB on Windows and with 2 GB on Mac and Linux systems by default. This may cause longer computing times on Windows computers.