Polish-British Workshop 2013:
Utility–Service Provision Optimisation Competition

Anna Strzelecka, Piotr Skworcow

1 Background

This competition is related to an EPSRC project "All in One: Feasibility Analysis of Supplying All Services Through One Utility Product". More information about the project can be found here: `http://www.allinone.uk.net/`.

The competition is open to anyone with a student status (undergraduate, MSc, PhD). There are prizes for the best work and the runner-up. We encourage to work in groups of two since the prizes come in pairs. However, other number of participants in one group is acceptable (within reason, i.e. up to 4); if there is only one person in a group, he/she will receive both items, if there are more than two, the participants will have to decide themselves what to do with the prizes. The prizes are high-capacity external hard disks and pendrives and are worth £200 in total. The workshop will be advertised on the All-in-One website and the participants names will be published. Additionally, authors of the most interesting solutions may be invited to co-author a journal paper which is under development.

All software and other files necessary to participate can be downloaded from `http://watersoftware.dmu.ac.uk/downloads/`.

2 Problem description

A household can be considered as an input-output system. The range of utility products to be provided include drinking water, gas, electricity, heat, food, etc. Ordinarily each of these products is provided via different utility companies. The provision of utility products can be supplemented by natural resources, i.e. recycling rain water, extracting humidity from the air or ground. In addition, domestic houses can be fitted with renewable technologies which capture and convert the energy from the sun or wind. Where on-site electricity generation which exceeds demand, surplus energy can be sold back to the grid.

A simulation system was developed to enable feasibility study of proposed solutions. In this exercise we consider:

- a common XML database with a graphical interface to browse its data,
- a problem formulation,
- a candidate solutions in the form of a transformation graph,
- simulator – a computational engine to analyse the feasibility and calculate the cost of candidate solutions.

Your task is to propose a transformation graph (format defined below), which will minimise the operational cost. Such graph can be created via manual editing of an XML file or via some software (to be developed by participants) or via mixture of both.

2.1 XML database

Information about products, services, devices and technologies is stored in the XML database and can be browsed using a purposely developed software with graphical interface.

Device transforms products into other products and/or services, see example in Fig. 1. Device has some **maximum throughput** (per hour). This defines the amount of units of products produced or services satisfied by the devices. The ratio between required input and produced output is fixed (i.e. no dynamics).

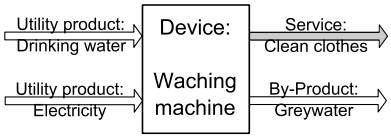

Figure 1: Input/output transformation for a device

Technologies associated with devices are not considered in this exercise.

2.2 Problem formulation

It is a set of requirements and constraints defined in file scenario.xml. It contains:

- time horizon for which the solution of the problem will be simulated - 50 hours.
- required services with the number of units that need to be delivered at each time step.
- maximum amount of products (including natural resources) that can be supplied and/or removed by the infrastructure with associated costs of supply/removal. There are also following flags:
  - can_supply = 0 – the product cannot be supplied by the infrastructure, but can be used within a graph.
  - can_supply = 1 – the product can be supplied by the infrastructure.
  - can_remove = 0 – this product cannot be removed, but can be used within a graph (has to be processed locally).
  - can_remove = 1 – this product can be removed by the infrastructure.
- maximum capacities for storages of each products;

Not all products have to be used in the proposed solution. If a product has both can_supply = 0 and can_remove = 0 it can be used in the graph, but has to be processed locally.

2.3 Transformation graph

Devices, services and product storages connected together form a transformation graph, see Fig. 2. The

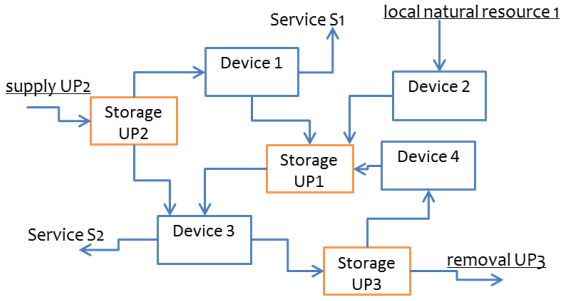

Figure 2: A transformation graph. UP denotes utility product.

transformation graph is defined in an XML file. In Fig. 3 – 6 structure of such file is presented.

Device nodes

Firstly, **device nodes** are defined (both, the ones that deliver services and used for recycling), see Fig. 3. Each node requires an unique node id, and is followed by specification of a node type (device, storage or service). Next, device id is indicated. The field "name" is optional.

While choosing devices to deliver services it is important to check maximum throughput of a device and compare it to service demand. It is possible that several devices will be required to meet the demands.

```xml
<?xml version="1.0" encoding="UTF-8"?>

<graphml>

  <graph id="G1" edgedefault="directed">

    <!-- devices to deliver services -->

    <node id="n0">
      <node_type>device</node_type> <!-- allowed node types: device, storage, service -->
      <device_id>device_45</device_id> <!-- used to retrieve correct record from XML DB -->
      <name>Toilet basin tap</name> <!-- used only for displaying as SVG/PNG -->
    </node>
```

Figure 3: Structure of a transformation graph defined in an XML file - example definition of a device

Storage nodes

Secondly, **storage nodes** are defined as presented in Fig. 4. Here node type is assigned, and the product type has to be specified, and has to match the record form the database. Storage for every product used in the transformation graph has to be defined.

```xml
    <!-- storages -->

    <node id="n6">
    <!-- out: n0 n1 -->
    <!-- in: n3 -->
        <node_type>storage</node_type> <!-- allowed node types: device, storage, service -->
        <product_id>product_1</product_id> <!-- used to retrieve correct record from XML DB -->
        <remove_threshold>1</remove_threshold> <!-- optional, push product to removal when stored amount (normalized to capacity
        <push_to_device_threshold>
            <dev node_id = "n0">1</dev> <!-- Toilet basin tap -->
            <dev node_id = "n1">1</dev> <!-- Shower with electric water heater -->
        </push_to_device_threshold> <!-- optional, push product to device connected to storage output when stored amount (norma
        <pull_from_device_threshold>
            <dev node_id = "n3">0.5</dev> <!-- Filtration and UV water purification system -->
        </pull_from_device_threshold> <!-- optional, pull product from device connected to storage input when stored amount (no
        <supply_threshold>0.0001</supply_threshold> <!-- optional, pull product from supply when stored amount (normalized to c
        <capacity>200</capacity> <!-- compulsory -->
        <initial_stored_amount>50</initial_stored_amount> <!-- optional -->
        <name>Drinking water</name> <!-- used only for displaying as SVG/PNG -->
    </node>
```

Figure 4: Structure of a transformation graph defined in an XML file - example definition of a storage

Each storage node has following **thresholds**, which are normalised to the **storage capacity**, i.e. can have values from 0 to 1. These threshold define operation of the connected devices and the infrastructure, depending on currently stored amount (normalised to storage capacity):

- remove threshold – push product to removal infrastructure when the currently stored amount is higher than this threshold; e.g. remove grey water when the stored amount exceeds 95% capacity.
- push to device threshold – push product to device connected to the storage output when the currently stored amount is higher than this threshold. Each device connected to storage output may have a different threshold assigned; e.g. turn on "Home anaerobic WW treatment" device when the stored wastewater amount exceeds 50% capacity and also turn on "Electro-coagulation WW treatment" device when the stored wastewater amount exceeds 70% capacity. If two devices have the same push to device thresholds, the order of activation of devices will be the same as the order in which they are defined in the XML file, i.e. if the 1st device reaches its maximum throughput and the currently stored amount is still greater than the threshold, then the 2nd device is activated.
- pull from device threshold – pull product from device connected to storage input when the currently stored amount is lower than this threshold. Each device connected to storage input may have a different threshold assigned; e.g. turn on "Silicon photovoltaic system" device when the stored electricity amount is lower than 60% capacity and also turn on "Diesel-powered generator" device when the stored electricity amount is lower than 10% capacity. If two devices have the same pull from device thresholds, the order of

activation of devices will be the same as the order in which they are defined in the XML file, i.e. similarly to the push to device threshold.

- supply threshold – pull product from supply infrastructure when the currently stored amount is lower than this threshold, e.g. take electricity from the grid when the stored amount is lower than 5% capacity.

If these thresholds are not specified, the simulation system will assign default values, i.e. 1, 0.75, 0.25 and 0 for remove threshold, push to device threshold, pull from device threshold and supply threshold, respectively. However, the thresholds are likely to influence the cost, hence it is not advisable to use default values. Additionally, the thresholds should be such that:

remove threshold >push to device threshold >pull from device threshold >supply threshold

Capacity of a storage is a compulsory field, but initial stored amount is optional; if not assigned, the storage is considered to be empty. Note that in the problem formulation maximum capacity of each storage is defined and it cannot be exceeded.

Service nodes

Thirdly, **service** nodes are defined, see Fig. 5. They have to reflect service demands defined in the problem formulation. It can happen that original demand has to be divided into several service nodes, e.g. when the demand is greater than device's maximum throughput; in that case, additional devices may be added to deliver the required service, but the service demand must be divided into (at least) two nodes. For each service type and for each time step, the sum of demands from all service nodes in the transformation graph have to be equal to the demand defined in the problem formulation.

```
191     <!-- services -->
192
193     <node id="n14">
194         <node_type>service</node_type> <!-- allowed node types: device, storage, service -->
195         <service_id>service_1</service_id> <!-- used to retrieve correct record from XML DB -->
196         <demand> <!-- specified demand for each time step -->
197         <time_series>
198                 <ts step = "8">2</ts>
199                 <ts step = "20">2</ts>
200         </time_series>
201         </demand>
202         <name>Full body cleaning</name> <!-- used only for displaying as SVG/PNG -->
203     </node>
```

Figure 5: Structure of a transformation graph defined in an XML file - example definition of a service

Edges

Lastly, connections between device nodes and service nodes are defined as graph edges, see Fig. 6. One device node can be connected to one service node. Only edges between devices and services have to be defined. Devices and storages are connected automatically.

```
230     <!-- edges -->
231
232     <edge source="n1" target="n14" />
233     <edge source="n0" target="n15" />
234     <edge source="n2" target="n16" />
235
236
237   </graph>
238
239 </graphml>
```

Figure 6: Structure of a transformation graph defined in an XML file - example definition of an edge

2.4 Simulator

Based on the problem formulation the simulator calculates mass balances of all products and services, checks feasibility of the proposed solution (transformation graph) and calculates operational cost. Output is saved as a CSV file. The total cost is a sum of the cost of removal and the cost of supply. Further, in the file information about service and product demands as well as produced amounts can be found for each device at each time step. All storages data can be also found in the results file: stored, supplied and removed amounts.

For the purpose of this competition the data about technologies, products, services and devices have been extracted from the database and stored in local files. All files are in an XML format, but it is advisable to use the XML database content browser provided to browse through them. The files are loaded together with the problem formulation (scenario.xml). Next, transformation graph is loaded. If there are no errors, then the results will be saved to results.csv file. All error notifications will appear in the command-line window in which the simulator was run.

We kindly ask not to interfere with the database and the problem formulation files provided, as solutions will be checked on the original files. **The participants should only create/modify their transformation graph file**.

3 Tools provided

- XML database content browser. Please install XmlDB_Editor and then localise and run XmlDatabase-ContentEditor.exe (default location is "C:\Program Files (x86)\WSS\XmlDB_Editor").
- Simulator is a command-line software. After installing it, please use the following calling syntax: simulator.exe scenario_file_name trans_graph_file_name output_file_name. Please keep all files in one folder.
- An example transformation graph (trans_graph.xml).

4 Outcomes expected

Participants should hand in:

- their proposed transformation graph file (please use the format provided above),
- short description (up to two pages) of the approach employed.

Please note that the transformation graph has to be feasible, i.e. we should be able to simulate it without any errors and constraint violations. Please send both files by 12.00 on Saturday the 8th of June 2013 to anna.strzelecka@email.dmu.ac.uk; the results will be announced later on the same day.

Notes

It is advisable to use Notepad++ to work on a transformation graph. We kindly ask not to interfere with the database and the problem formulation files provided, as solutions will be checked on the original files. **The participants should only create/modify their transformation graph file**.

If you have any questions you can approach Anna or Piotr during the PBW. All software and other files necessary to participate can be downloaded from `http://watersoftware.dmu.ac.uk/downloads/`.

The software was tested on Windows 7.