Engineering the Policy-making Life Cycle

Seventh Framework Programme – Grant Agreement 288147

Prototype of the global level policy reasoning system V1

Document type:	Deliverable
Dissemination Level:	PU
Editor:	Marco Gavanelli
Document Version:	1.0
Contributing Partners:	UNIFE, UNIBO
Contributing WPs:	WP3
Estimated P/M (if applicable):	6
Date of Completion:	29 May 2013
Date of Delivery to EC	31 May 2013
Number of pages:	31

ABSTRACT

This document describes the first version of the prototype of the global level policy reasoning system. It is based on the modeling techniques proposed in deliverable D3.1 [3], and it is focussed on the Regional Energy plan of the Emilia-Romagna region.

Authors of this document:

Marco Gavanelli[1], Michela Milano[2], Federico Chesani[2], Elisa Marengo[2]

[1]: EnDiF, University of Ferrara
email: {marco.gavanelli}@unife.it
[2]: DISI, University of Bologna
email: {michela.milano,federico.chesani,elisa.marengo}@unibo.it

Contents

This page has been intentionally left blank.

1 Introduction

Deliverable D3.1 [3] proposed a set of alternatives for addressing the problem of environmental assessment of a Regional plan (the so-called *Strategic Environmental Assessment*), including Causal Probabilistic Logic Programming, Fuzzy logic, and Constraint Logic Programming over the Reals (CLP(R)). That deliverable also contained the evaluation of the pros and cons of the various models, and the selection of CLP(R) as the best choice amongst the envisaged ones.

The choice of using CLP(R) also allowed us to perform regional planning, so the software was also able to *provide plans*, which are thus already assessed, and not only to assess plans proposed by experts.

This deliverable provides the software, as well as all the necessary documentation for installing it, running it, and modifying it to customize it for other domains. The software is currently tailored to the regional energy plan of the Emilia-Romagna region; however, we explain in detail how one can customize it for

- different types of plan (beside the energy plan)
- other regions, possibly including different parameters for the environmental assessment, or different energy sources, etc.
- technological advancements, that might lower the cost of current power sources, or even provide new energy sources.

In order to make the document self-contained, we first recap the constraint model, already provided in deliverable D3.1.

2 The constraint model

Regional Planning is the result of the main policy making activity of European regions. Each region has a budget distributed by the Operational Programme (OP): an OP sets out each region's priorities for delivering the funds. On the basis of these funds, the region has to define its priorities: in the field of energy, one example of priority is increasing the use of renewable energy sources. Then, a region should decide which activities to insert in the plan. Activities may be roughly divided into six types: (1) infrastructures and plants; (2) buildings and land use transformations; (3) resource extraction; (4) modifications of hydraulic regime; (5) industrial transformations; (6) environmental management. Also, a magnitude for each activity should be decided describing how much of a given activity is performed.

Each activity has an outcome (such as the amount of energy produced or consumed) and a cost. Let N_a the number of activities, we have two vectors $\mathbf{Out} = (out_1, \ldots, out_{N_a})$ and $\mathbf{C} = (c_1, \ldots, c_{N_a})$, where each element is associated to a specific activity and represents the outcome and cost per unit of an activity.

We distinguish primary from secondary activities: the former are activities of primary importance in a given plan; the latter are those activities supporting the primary activities by providing the needed infrastructures. There are constraints linking activities: for instance if a regional plan decides to build three biomass power plants (primary activities for an energy plan), each of these plants should be equipped with proper infrastructures (secondary activities like streets, sewage or possibly a small village nearby, power lines). Associated

to activities we have a matrix of dependencies between activities. In particular we have a $N_a \times N_a$ square matrix $\mathcal{D}$ where each element d_{ij} represents the magnitude of activity j per unit of activity i.

Also, each activity has impacts on the environment in terms of positive and negative pressures: example of positive pressure is the increased availability of energy, while a negative pressure is the production of pollutants. Pressures are themselves linked to environmental receptors such as the quality of the air, of surface water. On both pressures and receptors, there are constraints: for example the maximum amount of greenhouse gas emissions of the overall plan.

Taking as an example the Emilia-Romagna Regional Energy Plan approved in 2007, some objectives of the policy makers are the production of a given amount of energy (400 additional MW from renewable energy sources), while reducing the current greenhouse gas emission percentage by 6.5% with respect to 2003. In addition, the budget constraint limiting the amount of money allocated to the energy plan by the Regional Operational Programme was 30.5M€ in 2007.

The policy maker also has to take into account impacts on the environment, economy and society. One of the instruments used in Emilia-Romagna are the so called *coaxial matrices* [2], that are a development of the network method [7].

One matrix $\mathcal{M}$ defines the dependencies between the above mentioned activities contained in a plan and *impacts* (also called *pressures*) on the environment. Each element m^i_j of the matrix $\mathcal{M}$ defines a qualitative dependency between the activity i and the impact j. The dependency can be *high*, *medium*, *low* or *null*.

The second matrix $\mathcal{N}$ defines how the impacts influence environmental receptors. Each element n^i_j of the matrix $\mathcal{N}$ defines a qualitative dependency between the impact i and an environmental receptor j. Again the dependency can be *high*, *medium*, *low* or *null*. Examples of environmental receptors are the quality of surface water and groundwater, quality of landscapes, wildlife wellness and so on.

The matrices currently used in Emilia-Romagna contain 93 activities (including primary and secondary activities), 29 negative impacts, 19 positive impacts and 23 receptors, and have been used to assess 11 types of plans (those for which the assessment is legally required, i.e., Agriculture, Forest, Fishing, Energy, Industry, Transport, Waste, Water, Telecommunication, Tourism, Urban and Environmental plans).

2.1 A CLP model

To design a constraint-based model, we have to define variables, constraints and objectives. Variables represent decisions that have to be taken. Given a vector of activities $\mathbf{A} = (a_1, \ldots, a_{N_a})$, to each activity we associate a variable G_i that defines its magnitude. The magnitude could be represented either in an absolute way, as the amount of a given activity, or in a relative way, as a percentage with respect of the existing quantity of the same activity. We use in this deliverable the absolute representation.

As stated above, we distinguish primary from secondary activities: let A^P be the set of indexes of primary activities (i.e. those of main importance in a given plan) and A^S the set of indexes of secondary activities (i.e. those supporting the primary activities). The dependencies between primary and secondary activities are considered by the constraint:

$$\forall j \in A^S \quad G_j = \sum_{i \in A^P} d_{ij} G_i$$

Given a budget B_{Plan} available for a given plan, we have a constraint limiting the overall plan cost as follows

$$\sum_{i=1}^{N_a} G_i \, c_i \leq B_{Plan}. \tag{1}$$

Also, given an expected outcome out_{Plan} of the plan we have a constraint ensuring to reach the outcome:

$$\sum_{i=1}^{N_a} G_i \, out_i \geq out_{Plan}.$$

For example, in an energy plan the outcome can be to have more energy available in the region, so out_{Plan} could be the increased availability of electrical power (e.g., in megawatts). In such a case, out_i will be the production in MW for each unit of activity a_i.

Concerning impacts of the regional plan, we sum up the contributions of all the activities and obtain the estimate of the impact on each environmental pressure:

$$\forall j \in \{1, \ldots, N_p\} \quad p_j = \sum_{i=1}^{N_a} m_j^i \, G_i. \tag{2}$$

In the same way, given the vector of environmental pressures $\mathbf{P} = (p_1, \ldots, p_{N_p})$, one can estimate the influence on the environmental receptor r_i by means of the matrix $\mathcal{N}$, that relates pressures with receptors:

$$\forall j \in \{1, \ldots, N_r\} \quad r_j = \sum_{i=1}^{N_p} n_j^i p_i. \tag{3}$$

Concerning objectives, there are a number of possibilities suggested by planning experts. From an economic perspective, one can decide to minimize the overall cost of the plan (that is anyway subject to budget constraints). Clearly, in this case the most economic energy sources are preferred, despite their potentially negative environmental effects (which could be anyway constrained). On the other hand, one could maintain a fixed budget and maximize the produced energy. In this case the most efficient energy sources will be pushed forward. Or, the planner could prefer a *green* plan and optimize environmental receptors. For example, one can maximize, say, the air quality, or the quality of the surface water. In this case, the produced plan decisions are less intuitive and the system we propose is particularly useful. The link between decisions on primary and secondary activities and consequences on the environment are extremely complex to be manually considered. Clearly, more complex objectives can be pursued, by properly combining the above mentioned aspects.

2.2 The regional energy plan

We can now describe how to cast the general model for regional plan described above into the model for designing a regional energy plan. The first step is to identify primary and secondary activities. In the context of a regional energy plan, the environmental and planning experts defined the following distinction. Primary activities are those capable of producing energy, namely renewable and non-renewable power plants. Secondary activities are those supporting the energy production, such as activities for energy transportations (e.g., power lines), and infrastructures supporting the primary activities (e.g., dams, yards).

One important aspect is the energy source diversification: one should not use a single energy source, but should cover both renewable and non renewable energy sources. This requirement comes from fluctuations of the price and availability of the various resources. For this reason, we have constraints on the minimal fraction F_i of the total power produced by each source i.

In addition, each region has its own geo-physical characteristics. For instance, some regions are particularly windy, while some others are not. Hydroelectric power plants can be built with a very careful consideration of environmental impacts, the most obvious being the flooding of vast areas of land. This poses constraints on the maximum power that can be produced by a given energy source i.

Finally, the region priorities should be conformant with European guidelines such as the 20-20-20 initiative aimed at achieving three ambitious targets by 2020: reducing by 20% its greenhouse gas emissions, having a 20% share of the final energy consumption produced by renewable sources, and improving by 20% its energy efficiency. For this reason, we can impose constraints on the minimum amount of energy L_{ren} produced by renewable energy sources whose set is referred to as ${A^P}_{ren}$. The constraint that we can impose is

$$\sum_{i \in {A^P}_{ren}} G_i out_i \geq L_{ren}.$$

3 The CLP(R) software

In order to run the Global Optimizer it is necessary to download and install ECL^{i}PS^e and then to download and run the Global Optimizer. Instructions on how to that are reported hereafter in Sections 3.1 and 3.2. In order to try the Global Optimizer without installing it, it is possible to access to the Global Optimization Module Testing Page available online. This service is described in Section 5.

3.1 Downloading and installing ECL^{i}PS^e

The Global Optimizer software runs on top of ECL^{i}PS^e [5, 1] version 6 or greater. ECL^{i}PS^e is an open-source CLP environment, available for many platforms and operative systems, and it can be downloaded

- either from the ECL^{i}PS^e web site `http://www.eclipseclp.org/`
- or from SourceForge `http://sourceforge.net/projects/eclipse-clp/`

Using SourceForge, one can go to `http://sourceforge.net/projects/eclipse-clp/files/`, click on *Binaries XXX*, where XXX is the platform one is installing to (it can be PPC MacOSX, x86-64 MacOSX 64bit, x86 MacOSX, Windows, Linux x86_64 , Linux 32bit, Sparc Solaris or x86 Solaris).

Then, one should select the version of ECL^{i}PS^e he/she wishes to install, and finally is presented with a list of files as the following:

- `if_osiclpcbc.tgz`
- `tcltk.tgz`
- `eclipse_rt.tgz`
- `eclipse_doc.tgz`
- `eclipse_misc.tgz`
- `eclipse_basic.tgz`
- `UNPACK`
- `README_UNIX`
- `README`

Please download *all* files in a chosen directory.

In particular, the Global Optimizer uses the `eplex` library [6], which requires an external MILP solver, so when installing ECL^{i}PS^e, one should make sure to install also the libraries for interfacing with external MILP solvers. ECL^{i}PS^e provides by default an interface to the open-source CLP/CBC solver, simply by installing the optional packet `if_osiclpcbc.tgz`, available on the ECL^{i}PS^e web site.

After downloading all the files, the installation is different according to the platform one is using. Please, read file README, that should provide detailed installation instructions. To simplify the installation, we report here the instructions (taken from the README files) for Windows and Linux.

Windows For the most common Windows install configurations, use the ECLiPSe Windows Installer. To do so, download and execute the single file

`ECLiPSe<Version>_<Build>.exe`

Run this installer program as a user with "administrator" rights.

The installer contains the ECLiPSe kernel, basic libraries, and the following optional packages:

- Documentation (html, txt, pdf)
- An interface to COIN-OR's open source solvers CLP and CBC
- An interface to a version of Dash Optimization's XPRESS-MP solver
- 3rd party libraries
- A bundled standalone distribution of GraphViz (to support the visualisation tools)
- A bundled (but not standalone) distribution of Tcl/Tk (to support the TkECLiPSE GUI)

For installation, double click on `ECLiPSe<Version>_<Build>.exe` and the install wizard will guide you through the installation. By choosing to install 3rd party libraries from the wizard, the `eplex` library you need to run the Global Optimizer will be installed.

Then try to launch `tkeclipse` via the Start menu, or look at the documentation, e.g. the tutorial.

Linux After downloading all files in a folder, open a shell in that folder, then execute:

```
chmod a+x UNPACK
```

```
./UNPACK
```

```
./RUNME
```

Now, the software is installed; the installation terminates suggesting to add a directory to the PATH environment variable; please follow the instructions of you OS to do so.

After doing that, ECL^{i}PS^e can be run either from the command line, with command

```
eclipse
```

or with the graphical interface, by running

```
tkeclipse
```

3.2 Downloading and installing the Global Optimizer

The up-to-date version of the Global Optimizer can be directly obtained from the public SVN repository

```
https://svn.ing.unibo.it/svn/ai/OptModel/trunk/
```

The software can be downloaded with an SVN client; the following command downloads the directory tree necessary to run the software and generate the documentation:

```
svn co https://svn.ing.unibo.it/svn/ai/OptModel/trunk/
```

Otherwise, one can download the whole package (version 1.0) from

```
http://www.epolicy-project.eu/system/files/PUBLIC/deliverables/D3.2.zip
```

In the following, we will assume a basic knowledge of the Prolog language. For users who are novices to Prolog, an example of using the Global Optimizer is provided in Section 4.

3.3 User manual

The program is compiled by consulting file `modello.ecl`. If you are using ECL^{i}PS^e at command line, after running ECL^{i}PS^e type the following command:

```
?- [modello].
```

From the graphical front-end the program can be compiled by choosing compile from the file menu. Section 4 shows the use of the command line and the graphical front.

After compiling the program, the limits for the primary activities should be defined.

3.3.1 Defining limits for energy sources

They are defined by means of `dynamic` predicates, so they can be conveniently provided through the `assert/1` command of Prolog. For each of the possible energy sources, a fact

$$min\_max\_source(Activity, Min, Max).$$

should be provided, where $Activity$ is the name of the energy source, and Min and Max are, respectively, the minimum and maximum number of MW of that energy source that can be installed. It is important to notice that this value considers the new installations, and not the total at the end of the period (that would include, instead, also the existing power plants).

To simplify the input of the bounds, a number of predicates have been defined for the significant cases of energy plans of the Emilia-Romagna region. These predicates are

- $plan(Year)$. $Year$ can currently be either 2013 or 2020. Invoking $plan(2013)$, for each energy source the minimum and maximum value are fixed to the value proposed by the technicians of the Emilia-Romagna region in the 2013 Regional energy plan [4], which includes also a forecast for 2020.
  As both the limits are fixed to the same value, this predicate can be used to re-generate the plans provided in [4], and to provide their environmental assessment, but it is not used for deciding a new plan, nor for generating alternatives.
- $plan(Type, Year)$. $Type$ can either be ele (for *Electrical Energy*) or $therm$ (for *Thermal Energy*). The meaning is the same as the previous item, but here we consider separately the plan for electrical and for thermal energy; for example, in the case of electrical energy, all energy sources that do not provide electrical energy are fixed to zero.
- $plan\_amplified(Year)$. The $Year$ parameter has the same meaning as before. In this case, the asserted limits are a widening of the limits used for $plan$; they are typically amplified by a factor 2 (meaning that the Min is half the value proposed by the technicians for the Regional Energy Plan, while the Max is twice that value), but for some of the energy sources we include limits given by the technicians, that consider, e.g., subsidies already planned by the Region in other plans (e.g., the Agriculture plan) to foster energy sources (e.g., biomasses).
- $plan\_amplified(Type, Year)$. Similarly to the previous item the limits are a widening of the limits used for $plan$, but in this case the plans for electrical and thermal energy are considered separately. This is achieved by means of the parameter $Type$ that can either be ele or $therm$.

3.3.2 Computing Plans

To compute plans, the main predicate is $compute\_plan$. It accepts a number of parameters, not all of them are required. The predicate has various forms, to ease the insertion of optional parameters:

- $compute\_plan(Objective, Budget, ElectricalOut, ThermalOut)$
- $compute\_plan(Objective, Budget, ElectricalOut, ThermalOut, Receptors, Cost)$
- $compute\_plan(Objective, Budget, ElectricalOut, ThermalOut, Receptors, Cost, AddConstr)$
- $compute\_plan(Objective, Budget, ElectricalOut, ThermalOut, Receptors, Cost, AddConstr, ObjValue)$

where the parameters are:

Objective (Required, Input) The objective is the function that the user wants to optimize. It is a required input parameter, and it can be in the form $min(Term)$ or $max(Term)$, where $Term$ is a linear combination of (one or more of) the following:

cost : The total cost of the activities that are performed in the plan. This cost includes both the cost for the region (public cost) and the cost incurred by private stakeholders.

electric : The electric energy produced in a year, in kTOE (TOE stands for Tonnes of Oil Equivalent).

thermal : The thermal energy produced in a year, in kTOE.

rec(X) : where X is the index of one of the receptors, i.e., it can be a number ranging from 1 to N_r. As an example, currently the matrices used by ARPA for the strategic environmental assessment contain 23 receptors.

For example, valid objective functions are

- $min(cost)$: finds the plan with minimum cost
- $max(electric)$: finds the plan that provides the maximum possible energy
- $max(rec(9))$: finds the plan with the best possible value for receptor 9 (that, with the current receptors used by ARPA, is the *air quality*)
- $max(0.5 * rec(9) - 0.5 * cost)$: linear combination of two parameters; intuitively the solver should try to have a high air quality and a low cost. Of course, deciding the coefficients is not trivial.

Budget (Optional, Input) The budget for the plan, in M€. The $cost$ (intended as in the $Objective$, see previous item) should always satisfy the constraint $Cost \leq Budget$.

ElectricalOut,ThermalOut (Optional, Input) These two input parameters, when specified, provide the energy (in kTOE) that should be produced as Electrical Energy and as Thermal Energy.

Receptors (Optional, Output) Provides in output the list of values representing receptors in the optimal plan.

Cost (Optional, Output) Provides in output the cost of the plan

AddConstr (Optional, Input) This parameter is a list that contains further constraints that should be satisfied. The constraints are linear constraints of the form $Term >= Term$, $Term =< Term$ or $Term =:= Term$, where $Term$ has the same syntax given for the *Objective* function.

It is possible to give minimum/maximum values for $cost$, $electric$ or $thermal$ energy or for each receptor, as well as linear combinations of them; e.g., the following is syntactically accepted: $rec(1) >= 0.3 * cost + 2 * electric$.

In order to simplify the input of these data, other predicates are provided, for the typical values of the Regional Energy Plan with forecasts for 2013 and 2020 of the Emilia-Romagna Region. These predicates are

ele(α) invokes predicate compute_plan with the energy requirement for electrical energy as that in the Regional Energy Plan of the Emilia-Romagna region. Can be invoked, e.g., in conjunction with the aforementioned predicates $plan(ele, Year)$ or $plan\_amplified(ele, Year)$. The number α should be a real number between 0 and 1, and the considered objective is $\max((1-\alpha)rec(9) - \alpha cost)$, where $rec(9)$ refers to *air quality*.

term(α) invokes predicate compute_plan with the energy requirement for thermal energy. Can be invoked in conjunction with $plan(term, Year)$ or $plan\_amplified(term, Year)$. The meaning of α is the same as for predicate ele.

p(α) invokes predicate compute_plan with the energy requirement taken from the Regional Energy Plan of the Emilia-Romagna region, considering both electrical and thermal energy. The meaning of α is the same as for predicate ele.

3.3.3 Output results

Beside the usual result of Prolog predicates, i.e., the binding for output parameters, the compute_plan predicate can provide further data in the form of csv files (easily readable through common spreadsheets, like Microsoft Excel, LibreOffice/OpenOffice Calc, GNUmeric, etc.); these data can be printed on standard output, on file or on streams. Predicate $set\_print\_options(NewOptions)$ allows to set the desired output. Specifically, $NewOptions$ is a list of options among the following:

- the empty list makes no output to be produced.
- csv_to_file: print csv-formatted output to the standard files.
- csv_to_stdout: print csv-formatted output to the standard output.
- $csv\_to\_stream(Stream)$: print csv-formatted output to the specified stream.
- $generic\_to\_file$: print tabular-formatted output to the standard files.
- $generic\_to\_stdout$: print tabular-formatted output to the standard output.
- $generic\_to\_stream(Stream)$: print tabular-formatted output to the specified stream.

The outputs generated by the Global Optimizer are in Italian, in accordance with the use case we are considering (the output should be understandable by the policy makers of the Emilia-Romagna region). For the sake of comprehension, in this document the outputs are translated in English. We remind that the output of the Global Optimizer are not the final output of the ePolicy application, and that the final user interface will support the English translation as well.

Let us now list the various files that can be generated.

`result.txt` Predicate compute_plan provides on standard output the following information, which can also be printed on file `result.txt`:

- value of each activity, including its measuring unit (for those activities that have a measuring unit). The primary activities are shown in boldface, if the host O.S. allows for printing of special characters, e.g. in Linux. Figure 1 contains an excerpt of this section of the output.
- value of each pressure. See Figure 2 for an excerpt of the output.
- value of each receptor (Figure 3)
- various objectives that can be of interest for the user, including:
  - the formula the user asked to optimize, with its optimal value
  - the total cost of the plan
  - the total electrical energy

```
====================== Activities ======================
Sewers:  9.41002178337093km
Water Supply Systems:  15.4475217833709km
Sewage treatment and wastewater treatment plants:  89.6002178337093kab.eq.
Waste storage:  89.1502178337093kab.eq.
```
Thermoelectric Biomass Plants: 219.690468827907MWe
Biomass-based Thermal Plant: 671.311709509186MW
```
Aerial Power Line Supports:  1319.00217833709km
Aerial Power Lines:  1319.00217833709km
Underground Power Lines:  1.31900217833709km
Plants for Electricity Transformation:  1.31900217833709km
```
Windmill Electrical Generators: 20.0MWe
```
Gas, Oil and Vapor Pipelines:  671.316709509186km
Lighting Systems:  91.1502178337093km
Tunnel Ventilation Systems:  0.03
Houses and Residential Areas:  693.2837563919771000mc
Squares and Yards:  89.8032178337093ha
Roads:  1.49900217833709km
Paths:  1.49900217833709km
```
Small Hydroelectric Plants: 3.0MWe
```
Artificial Lake for Multiple Uses:  0.031000mc
Dams, dikes, beams, thresholds:  0.03mc
```
Photovoltaic Plants: 400.0MWe
Solar Thermal Panels: 37.5MW
Superficial Geothermal Plants: 5.0MW
Thermodynamic Solar Plants: 5.0MWe
```
Construction Sites (artifacts, traffic):  0.03
Fences (industrial areas):  13.0205217833709km
Disposal of Obsolete Facilities:  8.97585928337093(1000mc)
Caves and Mines:  0.03mc
Excavation and Soil Movements:  16.2600217833709mc
Stores for Excavation Resulting Materials:  0.008mc
Groundwater Extraction Plants:  896.007178337093mc
Well Drilling:  0.5km pozzo
Water Derivation Works:  0.896002178337093km
new Riverbeds (also correction existing riverbeds):  0.0896002178337093km
⋮
```

Figure 1: Excerpt of the section on Activities from a sample output of predicate compute_plan.

```
====================== Pressures ======================
1.  Energy transformation from non-renewable sources:  1189.92779218242
2.  Lithoid material consumption:  95.4096260569318
3.  Soil consumption/alteration:  1977.79100492174
4.  Water consumption:  1389.34348000493
5.  Substantial change of water flows:  936.28214607949
6.  Alteration in surface waters flows:  479.893460442804
7.  Alteration in groundwater flow and filtration:  1095.22781273687
8.  Intercept and alteration of nearshore currents:  0.002
9.  Wastewater dumping, water pollution:  2233.41406347513
10.  Spreading of hazardous materials:  1367.6365139551
11.  Waste production:  1663.6967582833
12.  Gas and dust emissions in atmosphere:  2751.36116421547
13.  Smells production:  1056.62681512071
14.  Noise production:  551.343070072278
15.  Vibrations production:  477.916405607828
16.  Electro-magnetic fields productions:  991.230137020325
17.  Ionizing radiations production:  22.8442548940947
18.  Heat transfer in air:  1189.72108594999
19.  Luminous pollution:  1455.72971978123
20.  Alteration of landscapes perceptions:  2794.47799974232
21.  Alteration of vegetation cover:  2529.00614416692
22.  Habitat fragmentation.:  1948.4804917872
23.  Attraction of unwanted organisms:  784.882817535685
24.  Introduction of exotic flora:  658.950494963014
25.  Transformation of urban functions:  2909.71710852415
26.  Attraction of non planned infrastructures:  813.181234887592
27.  Risk of important accidents:  2364.84899965763
28.  Road accidents:  1663.25432469415
29.  Use of explosives:  0.2875
30.  Creation of income, employment opportunities:  4660.3487467519
31.  Enhancement/creation of tangible assets:  5110.7785891288
32.  Improved operation of facilities/services:  6155.81729275774
33.  Creation of access opportunities:  518.268086257177
34.  Improves waste management:  1076.00177010558
35.  Control/reduction of air pollution:  371.91405425207
36.  Control/reduction of green house gas emissions:  1549.77609164354
37.  Control/reduction of water pollution:  621.173707359073
38.  Control/reduction of noise:  242.373602473276
39.  Control/reduction of ionizing radiations:  114.382922618887
40.  Control/reduction of non-ionizing radiations:  401.717789494567
41.  Saving/production of renewable energy:  1782.98106578519
⋮
```

Figure 2: Excerpt of the section on Pressures from a sample output of predicate compute_plan.

```
====================== Receptors ======================
1.  Subsidence limitation:  48.7491680507435
2.  Embankments stability:  -3644.56862849131
3.  Stability of coasts or seafloor:  -392.823165376105
4.  Stability of river banks and beds:  -1493.21427445758
5.  Soil quality:  -3609.9402259488
6.  Quality of sea water:  -2094.03972784824
7.  Quality of inland surface waters:  -3603.92777322148
8.  Groundwater:  -5096.01723947927
9.  Air quality:  -4100.68627266995
10.  Quality of climate:  -1161.74560627152
11.  Wellness of terrestrial vegetation:  -7143.85439448481
12.  Wellness of wildlife:  -9722.04252109749
13.  Wellness of aquatic plants:  -7513.74158156845
14.  Wellness and health of mankind:  786.159566677553
15.  Quality of sensitive landscapes:  -10598.705514049
16.  Cultural/historical heritage value:  -6819.45183888558
17.  Recreation resources accessibility:  -915.132470840773
18.  Water availability:  -4664.35480988736
19.  Availability of agricultural fertile land:  -4410.00596335353
20.  Lithoid resource availability:  872.635576256134
21.  Energy availability:  165.82269222421
22.  Availability of productive resources:  12735.6272529762
23.  Value of material goods:  9930.7325565569
```

Figure 3: Section on Receptors from a sample output of predicate compute_plan.

```
====================== Objectives ======================
Objective:  max((1 - 0.5) * ric(9) - 0.5 * cost) = -1363566163.55179
cost(2727128226.41731)
energy(177.2698956357)
```

Figure 4: Section on Objectives from a sample output of predicate compute_plan.

Activity	Quantity	Subsidence limitation	Embankments stability	Stability of coasts or seafloor	Stability of river banks and beds	Soil quality	…
Sewers	9.41002178337093	-2.35250544584273	-5.88126361460683	0.588126361460683	-2.35250544584273	-6.46938997606752	
Water Supply Systems	15.4475217833709	-5.7928206687641	-13.5165815604496	-4.82735055730342	-11.5856413375282	-6.75829078022478	
Sewage and Wastewater Treatment Plants	89.6002178337093	5.60001361460683	-56.0001361460683	0.0	-33.600081687641	-67.200163375282	
Waste Storages	89.1502178337093	-27.8594430730342	-55.7188861460683	16.7156658438205	5.57188861460683	-55.7188861460683	
Thermoelectric Biomass Plants	219.690468827907	68.6532715087209	-54.9226172069768	41.1919629052326	13.7306543017442	-96.1145801122093	
Biomass-based Thermal Plant	671.311709509186	209.784909221621	-167.827927377296	125.870945532972	41.9569818443241	-293.698872910269	
Aerial Power Line Supports	1319.00217833709	0.0	-329.750544584273	-82.4376361460683	-164.875272292137	-412.188180730342	
Aerial Power Lines	1319.00217833709	0.0	-577.063453022478	0.0	-164.875272292137	-494.62581687641	
Underground Power Lines	1.31900217833709	0.0	-0.659501089168547	-0.0824376361460683	-0.247312908438205	-0.659501089168547	
Plants for Electricity Transformation	1.31900217833709	0.0	-0.659501089168547	-0.0824376361460683	-0.247312908438205	-0.659501089168547	
Windmill Electrical Generators	20.0	6.25	-6.25	1.25	-1.25	-7.5	
Gas, Oil and Vapor Pipelines	671.316709509186	0.0	-419.572943443241	-41.9572943443241	-125.871883032972	-335.658354754593	
Lighting Systems	91.1502178337093	0.0	-17.0906658438205	0.0	-5.69688861460683	-17.0906658438205	
.							
.							
.							

Figure 5: Excerpt of the content of the file `activities_receptors.csv`

`energies.csv` This file shows the produced energy (in kTOE) of electrical and thermal energies for each of the sources that can provide electrical and thermal energies. For example, using as limits those given by $plan\_amplified(2013)$ and as goal $p(0.5)$, file `energies.csv` contains:

Source	Electrical Energy	Thermal Energy
Thermoelectric Biomass Plants	132.2536622344	0.0
Biomass-based Thermal Plant	0.0	288.66403508895
Windmill Electrical Generators	2.575	0.0
Big Hydroelectric Plants	0.0	0.0
Small Hydroelectric Plants	0.6706451613	0.0
Photovoltaic Plants	41.27058824	0.0
Solar Thermal Panels	0.0	4.84999999875
Superficial Geothermal Plants	0.0	3.2236842105
Thermodynamic Solar Plants	0.5	0.0

`activities_receptors.csv` This table provides the impact of each of the activities on each of the receptors. This table is mainly meant for the environmental expert, that can see the contributions of the activities on the receptors in a plan. Figure 5 shows an excerpt of this file.

`tab_summary.csv` This file provides a table in the same format used in the Regional Energy Plans [4] of the Emilia-Romagna region. An example is shown in Figure 6. The table has two sections, one for electrical energy and one for thermal energy.

For each energy source, it shows the installed power from the last available data (MW), an estimate for the current year, the objective, bot in terms of MW of installed power and in terms of kTOE of energy produced in a year. Finally, it shows the cost, in millions of Euros.

3.3.4 Computing the Pareto frontier

Since there can be different objectives (possibly conflicting) to be pursued, it is interesting to show the Pareto frontier of the different objectives.

Electric Energy Production	Current level at 2009 (MW)	Assessment at 2010 (MW)	Overall Objective at 2013 (MW)	Overall Objective at 2013 (kTOE)	Investments (M€)
Big Hydroelectric Plants	0	0	0.0	0.0	0
Small Hydroelectric Plants	297	300	303.0	67.7351612913	25.2
Photovoltaic Plants	95	230	630.0	65.001176478	1400.0
Thermodynamic Solar Plants	0	0	5.0	0.5	22.5
Windmill Electrical Generators	16	20	40.0	5.15	40.0
Thermoelectric Biomass Plants	371	430	649.690468827907	391.1136622344	768.916640897675
Total	779	980	1627.69046882791	529.5000000037	2256.61664089767

Thermal Energy Production	Current level at 2009 (MW)	Assessment at 2010 (MW)	Overall Objective at 2013 (MW)	Overall Objective at 2013 (kTOE)	Investments (M€)
Solar Thermal Panels	25	25	62.5	8.08333333125	90.0
Superficial Geothermal Plants	23	23	28.0	18.0526315788	34.2
Biomass-based Thermal Plant	100	120	791.311709509186	340.26403508895	346.311585519632
Total	148	168	881.811709509186	366.399999999	470.511585519632

Figure 6: Example of the content of the file `tab_summary.csv`, that is in the same format as the table shown in the Regional Energy Plan of the Emilia-Romagna region.

Label	Air Quality	Cost	Plan	Biomass	Photovoltaic	…
0.996094943839125	-483.501132949337	2256.61664089767				
0.996094942915079	-483.501132949337	2256.61664089768				
0.996094942143827	-483.501132949337	2256.61664089767				
0.996095061302185	-483.501132949337	2256.61664089767				
0.992183685302734	-483.501132949337	2256.61664089768				
1	-483.501132949337	2256.61664089767				
0.9921875	-483.501132949337	2256.61664089767				
0.996101379394531	-483.501132949337	2256.61664089767				
0.99169921875	-483.501132949337	2256.61664089767				
0.5	-483.501132949337	2256.61664089767				
0.996094703674316	-483.501132949337	2256.61664089768				
0.992172241210938	-483.501132949337	2256.61664089767				
0.996094942907803	-483.501132949337	2256.61664089768				
9.5367431640625e-7	-358.236967822676	2381.76198973488				
0	190.706586432088	4045.18597636562				
plan	-212.623837504884		3013.9999996715			
Same air quality	-212.623837504884	2823.00293675325				
Same cost	-149.593240727854	3013.9999996715				
Biomass	-897.397559402556			1030.63892811453		
Photovoltaic	746.964832352661				6013.43146472147	
⋮						

Figure 7: Example of the content of file `pareto_ele.csv`.

Predicate plot computes a number of different plans, including:

- the Pareto graph of two objectives, namely the receptor *Air quality* and the cost of the plan;
- the plan proposed by the Region's technicians;
- two plans, that are the projection of the plan of the technicians onto the Pareto front; in other words, two plans that dominate the plan proposed by the Region's experts;
- the *extreme* solutions (that can be used as a comparison) in which only one type of energy source is used.

Predicate plot is invoked as $plot(Type, Year)$, where Type can take values ele or $therm$, and $Year$ can currently take values 2013 or 2020. The meaning of the parameters is the same as in predicate plan_amplified.

The results are saved in one of the files `pareto_ele.csv` or `pareto_term.csv`, depending on the $Type$ of energy requested.

The content of the file can be though of as consisting of three sections.

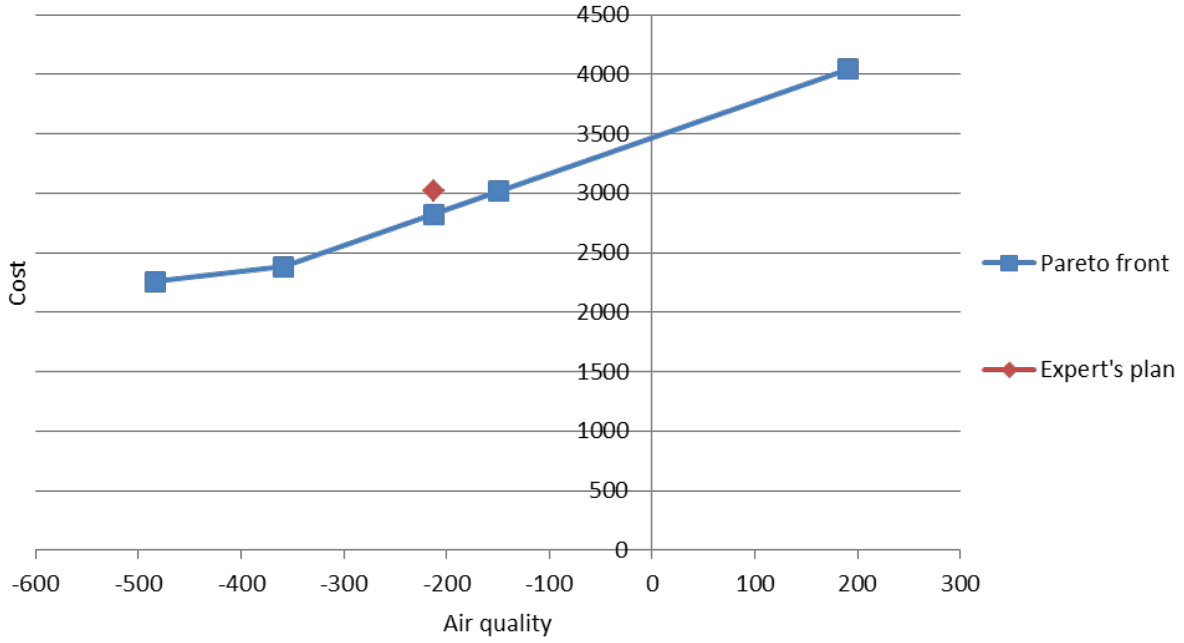

Figure 8: Pareto graph corresponding to the example shown in Figure 7

The first is the Pareto front: for each significant point in the Pareto front we have the two coordinates *air quality* and cost.

The second section contains the comparison of the plan proposed by the technicians with the Pareto front. In fact, the third column is the cost of the plan proposed by the experts, while its air quality is reported in the second column. This makes it very simple, with common spreadsheets, to draw a graph like the one in Figure 8, simply by selecting the three columns, and plotting the data as a scatter plot. Note that the plot shows:

- the experts' plan (in red)
- the Pareto front, with highlighted the significant points, where the curve changes its slope
- two plans on the Pareto front, that dominate the plan by the experts; more precisely, one has the same air quality as the experts' plan, the other has the same cost.

The third section contains, for each of the envisaged energy sources, a hypothetical plan providing all the required energy through that source. Notice that such plans are provided just for the sake of comparison, and they might not satisfy all the constraints imposed by the user[1]. Note that not all energy sources are reported here; only those sources listed in predicate $sources(Type, Sources)$[2]. The data is organized in the file `pareto_ele.csv` as follows. For each of these plans, the air quality is shown in column 2 (labelled air quality), while the cost is in a column containing only one value. Again, this organization allows one to plot the

[1]For example, of course they do not satisfy diversification constraints. Also, one could be interested to plot the impact of a non-renewable source; of course such plan does not satisfy the requirement of having a minimum of energy coming from renewable sources.

[2]This can be useful because some energy sources could be uninteresting for some region, so there is no reason to plot them; on the other hand, the Global Optimizer is designed to be general enough to contain many energy sources.

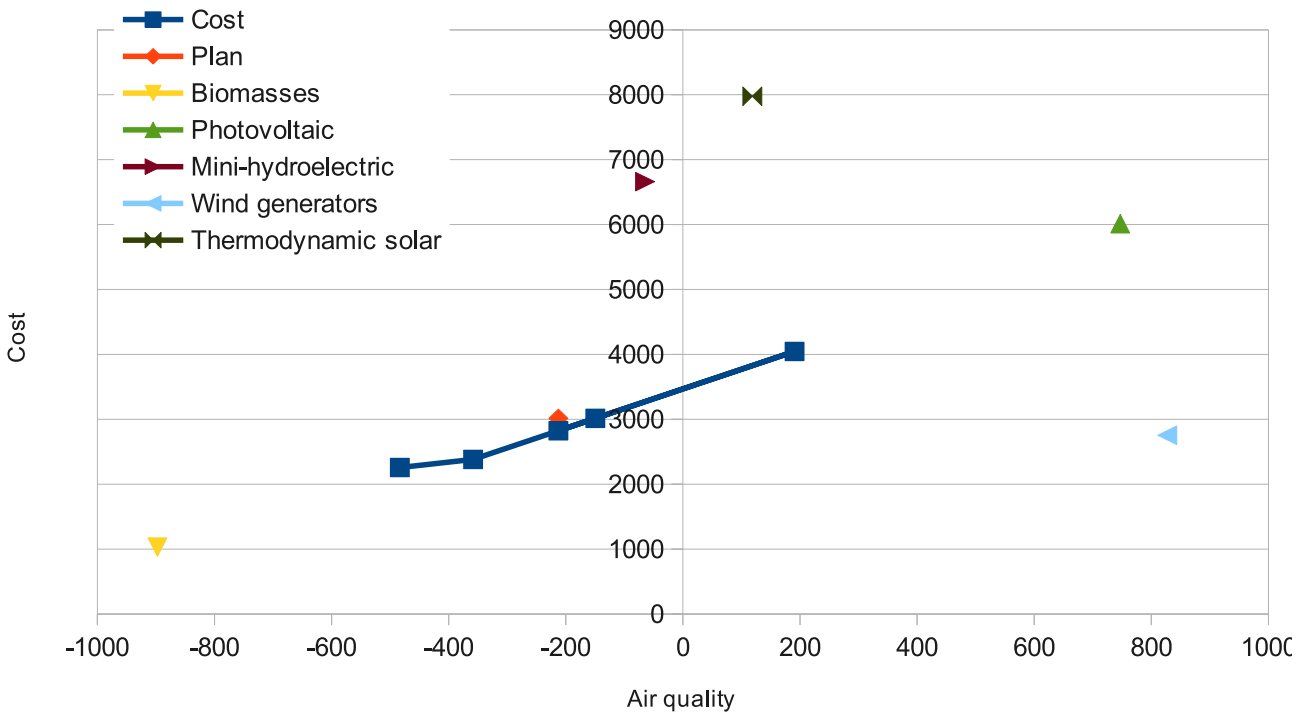

Figure 9: Plot of the single energy sources taken from the example shown in Figure 7

data in a convenient way, by using standard plot types of common spreadsheets; in fact, by selecting that part and choosing a scatter plot, we obtain a graph similar to that in Figure 9. The graph shows in a synoptic way the plans using single energy sources, the Pareto front, and the experts' plan.

For each of these plans, the files described in Section 3.3.3 are saved in a suitable directory. In particular,

- the significant points on the Pareto curve are saved in folders `paretoXX`, where `XX` is a number from 0 to the number of significant points;
- the experts' plan is saved in folder `piano`;
- the plan on the Pareto front with same air quality of the experts' plan is saved in folder `pari_aria`;
- the plan on the Pareto front with same cost of the experts' plan is saved in folder `pari_costo`;
- the plans using only one energy source are saved in a folder named as that source.

3.3.5 What if there is no solution?

In some cases, the user might provide some combinations of the input parameters for which there is no solution; in such cases the output of constraint solver is simply a failure, without any explanation of what is the cause of the failure.

In order to provide some hints on the input parameters, we provide a predicate why_not that can be used to find solutions.

Predicate why_not has three parameters: Budget, electrical energy and thermal energy. Given three values (which can also be variables), it unassigns one of the three parameters, and com-

putes its minimum and maximum possible values, assuming that the other two parameters take the values in input from the user.

For example, the goal $why\_not(2727128226, 177, 90)$ prints as output:

```
electrical=177,thermal=90, --> cost in 2477534862.33179 .. 4529697710.15843
cost=2727128226,thermal=90, --> electrical in 105.2162334013 .. 219.930058550931
cost=2727128226,electrical=177, --> thermal in 89.77368420925 .. 298.045674763611
```

meaning that if the requested electrical energy is 177kTOE and the thermal energy is 90kTOE, then the cost can range between 2477534862.33179 and 4529697710.15843€.

3.3.6 Configuring the program for other types of plans or other regions

The data used in the global optimizer was provided by ARPA Emilia-Romagna, by ENEA, and by the experts in the Emilia-Romagna region.

The global optimizer stores the input data into various files, as Prolog facts. Below, for each input we report the name of the file in which it is defined. For different configurations of the program it is possible to modify these files.

We now describe the data and the format required by the global optimizer; this can be used to tailor the application for other needs, like, for instance, include new pressure, new receptors, update the coefficient to include new technologies for producing energy, preparing a different type of plan, or any customization for other regions, etc.

- **Coaxial matrices.** As explained in Section 2, the Coaxial Matrices link possible Activities with environmental Pressures, and Pressures to environmental Receptors.
  Each Activity, Pressure, and Receptor has a unique identifier, that is given as a Prolog atom. For example, *'wind generators'* could be an activity, *'water consumption'* is a possible pressure, and *'water quality'* is a possible receptor.
  Pressures are declared in the file `press.pl` with a predicate

  $$pressures(List\ of\ pressures). \tag{4}$$

  The program expects the pressures to be always in the same order as given in this list. Predicate

  $$opere\_press(Activity\ Type, Activity, List\ of\ Impacts\ Activity/Pressure).$$

  defines an Activity, and also its impact on the various pressures. This set of predicates is defined in the file `activity_press.pl`. The *A*ctivity Type is currently not used, and it could be used in future extensions.
  The `List of Impacts Activity/Pressure` has the same length as list *List of pressures* of Eq. (4), and encodes the content of matrix $\mathcal{M}$ described in Section 2. Each element of the list can be a qualitative value with the following code:
  - **a** for High impact
  - **m** for Medium impact
  - **b** for Low impact
  - or it can be a Prolog variable if the activity has no impact on the given pressure.

Predicate

$$ricet\_press(Receptor, List\_of\_impacts\_Pressure/Receptor).$$

declares a Receptor, as well as the impact that each pressure has on such receptor. These predicates are defined in the file `rec_press.pl`. The list $List\_of\_impacts\_Pressure/Receptor$ has the same length as the list of pressures (Eq. 4); each element can take one of these values:

- **a** for a High positive impact
- **m** for a Medium positive impact
- **b** for a Low positive impact
- **l** for a Low negative impact
- **i** for an Intermediate negative impact
- **h** for a High negative impact
- or it can be a Prolog variable if the pressure has no effect on that receptor.

The qualitative values are converted into numerical coefficients through predicate `coefficient`, that is currently defined as follows:

```
coefficient(X,0):- var(X),!.
coefficient(a,0.75).
coefficient(m,0.5).
coefficient(b,0.25).
coefficient(h,-0.75).
coefficient(i,-0.5).
coefficient(l,-0.25).
```

This choice makes very easy the addition of more qualitative values, or even the use of quantitative values (it is just a matter of suitably redefining predicate `coefficient`).

- **Primary-secondary activities.** The definition of the relation between primary and secondary activities is declared in the file `activity_settings.pl` through the predicate

$$prim\_sec(Primary, Secondary, Quantity).$$

where both $Primary$ and $Secondary$ should be activities defined in predicate opere_press, and $Quantity$ is a real number providing how much of the Secondary activity is necessary in order to implement one unit of the $Primary$ activity.

- **Outcomes**. Some of the activities provide an *outcome* that is important for the given plan type. For an energy plan, the outcome is the produced energy. Predicate

$$outcome(List\ of\ outcomes).$$

provides for each activity the outcome. The $List\ of\ outcomes$ (which is specified in file `activity_settings.pl`) has the same number of elements as the activities, and assumes the values are given in the same order as given by predicate opere_press.
In the Regional Energy Plan of the Emilia-Romagna region, this value is the electrical energy in kTOE, that one unit of the activity can provide in a year. Note that in the Regional Energy Plan of the Emilia-Romagna region, the activities corresponding to

power plants are measured in MW of installed power, so in this case the outcome measures the *efficiency* of the energy source (e.g., solar power is available only during the day, so on average the energy produced in a year is lower than a biomass plant having the same installed power).
For the energy plan, we also have a predicate

$$outcome\_termico(List\ of\ outcomes).$$

that provides the same information for thermal energy (and which is specified in file `activity_settings.pl` as well).

- **Costs.** The cost of each activity is declared in the file `cost_activities.pl` by means of the predicate

$$cost\_activity(Activity, Unit\ manag.\ cost, Manag.\ cost\ measuring\ unit, Initial\ invest.\ cost, Invest.\ cost\ unit, Years).$$

note, however, that not all the provided data is used in the current implementation, and they are provided in this more general form for possible future developments. In detail,
  - $Activity$ is the activity
  - $Unit\ manag.\ cost$ is the yearly management cost of the activity (per unit of the activity). This data is not currently used.
  - $Manag.\ cost\ measuring\ unit$ is the measuring unit for the $Unit\ manag.\ cost$ (for a power plant, it is usually €/$(MW \cdot year)$). Currently not used.
  - $Initial\ invest.\ cost$ is the cost of installation of the activity, i.e., the initial investment.
  - $Invest.\ cost\ unit$ is the measuring unit for $Initial\ invest.\ cost$. For a power plant, it is usually in €/MW.
  - $Years$ is the expected lifetime of the activity. Currently, it is not used.
- **Current levels.** As we have seen, the tables provided in the regional energy plan (see Figure 6) contain information about the current installed power for each of the energy sources. The current levels are declared in predicates $current\_level(Activity, Installed)$ and $current\_level2010(Activity, Installed)$, respectively for the last available data and for the expected to-date level. Both are declared in file `activity_settings.pl`.
- **Renewable sources.** Predicate $renewable(Activity)$ declares which activities provide energy from renewable sources. This datum is not currently used, because in the 2013 regional energy plans of the Emilia-Romagna region only renewable sources were considered. However, for future extensions distinguishing renewable from non-renewable sources could be used, to enforce policies requiring a minimum share of energy coming from renewable sources. The file containing these predicates is `activity_settings.pl`.
- **Minimum/maximum shares.** In order to avoid that all the required energy should not come from a single energy source, one can state the minimum and maximum percentage of energy that should be provided for (one or more) sources. This can be done in file `activity_settings.pl` through predicate

$$min\_max\_source\_perc(Activity, Min, Max).$$

that declares that the given $Activity$ should not have a magnitude less than Min% (nor more than Max%) of the total of the magnitudes of all the activities for which there exists a fact $min\_max\_source\_perc$. For example, declaring

$$min\_max\_source\_perc(photovoltaic, 10, 100).$$
$$min\_max\_source\_perc(wind, 0, 100).$$
$$min\_max\_source\_perc(nuclear, 0, 100).$$

one declares that at least 10% of the power provided by the energy sources *photo-voltaic*, *wind* and *nuclear* should come from photo-voltaic (independently from the fact that the system could consider other energy sources as well).

3.3.7 Adding new energy sources

This section recaps the information needed to add new energy sources and it is not strictly necessary, since all the needed input is already listed in Section 3.3.6; however we provide it as a to-do list to simplify the work for those that want to tailor the Global Optimizer for other regions, or for other energy sources.

To add a new activity (in particular, energy sources):

- Add a row to the matrix $\mathcal{M}$ (file `activity_press.pl`), by adding a fact $opere\_press(Activity, Impacts)$.
- Add a fact $renewable(Activity)$, if it is a renewable energy source (the file containing this information is `activity_settings.pl`).
- Add an element to the lists outcome and outcome_termico (file `activity_settings.pl`), containing the energy (in kTOE) that a plant of 1MW of that source can provide in a year, considering electrical energy (outcome) and thermal energy (outcome_termico).
- Add the secondary activities in predicate prim_sec (file `activity_settings.pl`). Note that, in order to be considered primary, an activity must have at least a secondary activity (even if with a zero coefficient).
- Add minimum and maximum allowed values in predicate min_max_source, and/or in predicate $min\_max\_source\_perc$ (file `activity_settings.pl`).
- Add the current level of installation (file `activity_settings.pl`) in terms of last-known values (predicate current_level) and foreseen (predicate current_level2010).
- Add the energy source to predicate $sources(Type, SourcesList)$, if the source should be plot as a single source in the graph in Figure 9 (file `activity_settings.pl`).

4 Example

Assuming the ECL^{i}PS^e installation was successful, one can run it either from the command-line or through the graphic front-end `tkeclipse`. In this example outputs are reported in Italian, as they are generated by the Global Optimizer and in accordance with the use case we are considering (the Emilia-Romagna region). The final user interface will support the English version.

From the command-line. If the ECL^{i}PS^e directory is in your PATH environment variable (and that there are no other homonyms applications), then executing `eclipse` should run the ECL^{i}PS^e complier.

Go to the directory where you downloaded the Global Optimizer, then run `eclipse`:

```
marco@marco-Latitude-E6530 ~ $ cd OptModel
marco@marco-Latitude-E6530 ~/OptModel $ eclipse
ECLiPSe Constraint Logic Programming System [kernel]
Kernel and basic libraries copyright Cisco Systems, Inc.
and subject to the Cisco-style Mozilla Public Licence 1.1
(see legal/cmpl.txt or http://eclipseclp.org/licence)
Source available at www.sourceforge.org/projects/eclipse-clp
GMP library copyright Free Software Foundation, see legal/lgpl.txt
For other libraries see their individual copyright notices
Version 6.0 #201 (x86_64_linux), Mon Feb 18 22:23 2013
[eclipse 1]:
```

Now, from the ECL^{i}PS^e shell, you can compile the Global Optimizer with `[modello].`

```
[eclipse 1]:  [modello].
source_processor.eco loaded in 0.01 seconds
hash.eco   loaded in 0.00 seconds
compiler_common.eco loaded in 0.01 seconds
compiler_normalise.eco loaded in 0.00 seconds
compiler_map.eco loaded in 0.00 seconds
compiler_analysis.eco loaded in 0.00 seconds
compiler_peephole.eco loaded in 0.01 seconds
compiler_codegen.eco loaded in 0.00 seconds
compiler_varclass.eco loaded in 0.01 seconds
compiler_indexing.eco loaded in 0.00 seconds
compiler_regassign.eco loaded in 0.01 seconds
asm.eco    loaded in 0.01 seconds
module_options.eco loaded in 0.01 seconds
ecl_compiler.eco loaded in 0.07 seconds
linearize.eco loaded in 0.00 seconds
constraint_pools.eco loaded in 0.00 seconds
loading OSI clpcbc ... done
empty_language.eco loaded in 0.01 seconds
eplex_standalone.eco loaded in 0.04 seconds
eplex.eco  loaded in 0.05 seconds
matrix_util.eco loaded in 0.00 seconds
/home/marco/Software/ECLiPSe/lib/listut.pl compiled 17024 bytes in 0.02 seconds
util.eco   loaded in 0.00 seconds
var_name.eco loaded in 0.00 seconds
modello.ecl compiled 604624 bytes in 0.48 seconds
```

```
Yes (0.55s cpu)
[eclipse 2]:
```

Now, you can define sensible ranges for activities by running `plan_amplified(2013).`

```
[eclipse 2]: plan_amplified(2013).

Yes (0.00s cpu, solution 1, maybe more) ?
```

Finally, you can run the Global Optimizer with `p(0.5).`

```
[eclipse 3]: p(0.5).
====================== Opere ======================
Opere fognarie: 9.41002178337093km
Impianti adduzione idrica (p.e. acqedotti): 15.4475217833709km
Depuratori e impianti trattamento reflui: 89.6002178337093kab.eq.
Stoccaggio rifiuti: 89.1502178337093kab.eq.
Centrali termoelettriche a biomassa: 219.690468827907MWe
...
====================== Pressioni ======================
1. Trasformazione d'energia da fonti finite: 1189.92779218242
2. Consumo di materiali litoidi: 95.4096260569318
3. Consumo, alterazione di suolo: 1977.79100492174
4. Consumo di acqua: 1389.34348000493
5. Variaz. consistente di portate idriche: 936.28214607949
...
====================== Ricettori ======================
1. Limitaz.subsidenza e stabilita' falde: 48.7491680507435
2. Stabilita' di versanti e scarpate: -3644.56862849131
3. Stabilita' di litorali o fondali mare: -392.823165376105
4. Stabilita' di rive o alvei fluviali: -1493.21427445758
5. Qualita' pedologica di suoli: -3609.9402259488
...
====================== Obiettivi ======================
Obiettivo: max((1 - 0.5) * ric(9) - 0.5 * costo) = -1363566163.55179
costo(2727128226.41731)
energia(177.2698956357)

Yes (0.17s cpu, solution 1, maybe more) ?
```

To exit from ECL^{i}PS^e, write `halt.`

```
[eclipse 4]: halt.
marco@marco-Latitude-E6530 ~/OptModel $
```

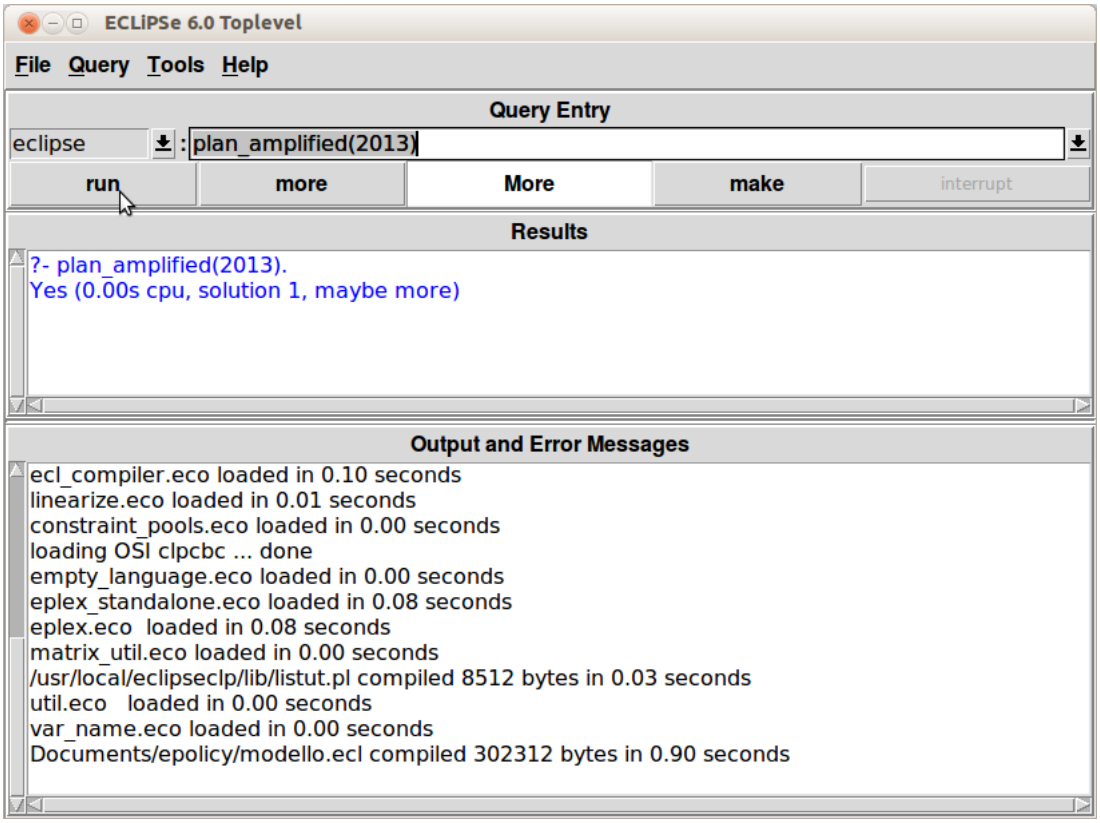

Figure 10: Graphical front-end for running ECL^{i}PS^e.

From the the graphic front-end. If the ECL^{i}PS^e directory is in your PATH environment variable (and that there are no other homonyms applications), then executing `tkeclipse` should open a window like the one depicted in Figure 10. Alternatively, if you are running Windows, you can launch `tkeclipse` from your Start menu.

To compile the Global Optimizer open the file menu, browse to the directory where you downloaded the Global Optimizer and choose the file `modello.ecl`.

In the "Output and Error Messages" part of the window you should obtain the following result.

```
source_processor.eco loaded in 0.00 seconds
hash.eco   loaded in 0.00 seconds
compiler_common.eco loaded in 0.01 seconds
compiler_normalise.eco loaded in 0.00 seconds
compiler_map.eco loaded in 0.00 seconds
compiler_analysis.eco loaded in 0.01 seconds
compiler_peephole.eco loaded in 0.00 seconds
compiler_codegen.eco loaded in 0.02 seconds
compiler_varclass.eco loaded in 0.00 seconds
compiler_indexing.eco loaded in 0.01 seconds
compiler_regassign.eco loaded in 0.00 seconds
asm.eco    loaded in 0.01 seconds
module_options.eco loaded in 0.01 seconds
```

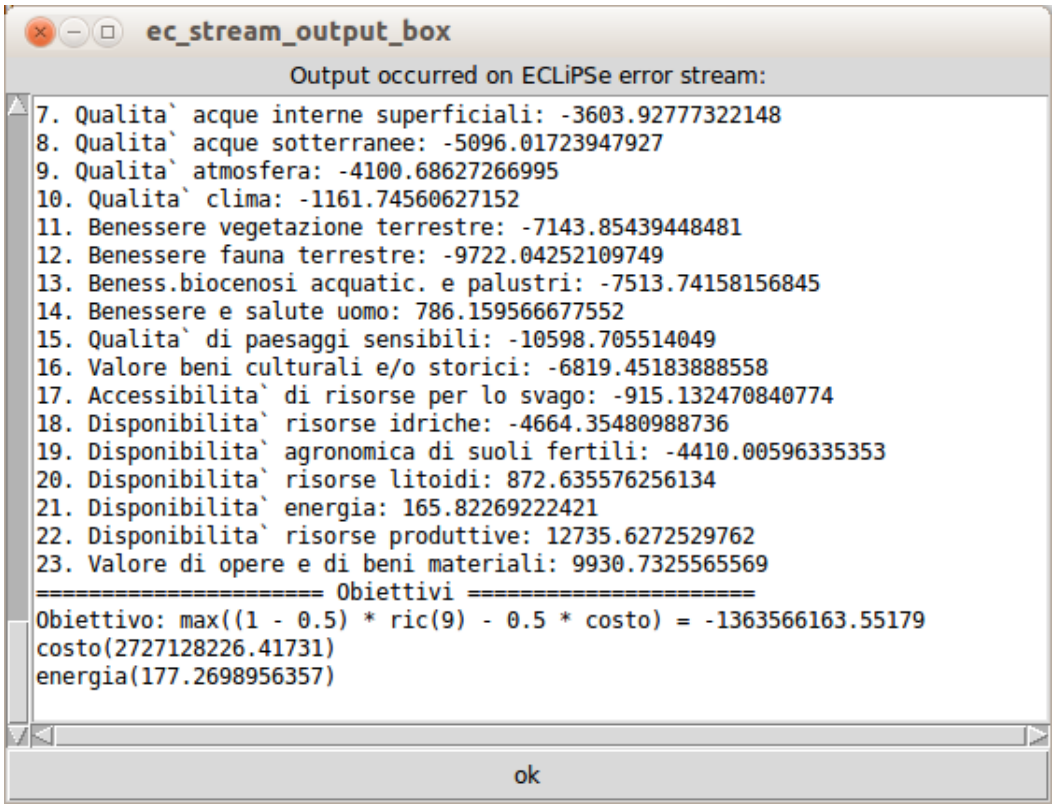

Figure 11: Output window when running the graphical front-end of ECL^{i}PS^e.

```
ecl_compiler.eco loaded in 0.07 seconds
linearize.eco loaded in 0.00 seconds
constraint_pools.eco loaded in 0.01 seconds
loading OSI clpcbc ... done
empty_language.eco loaded in 0.00 seconds
eplex_standalone.eco loaded in 0.06 seconds
eplex.eco  loaded in 0.07 seconds
matrix_util.eco loaded in 0.00 seconds
/usr/local/eclipseclp/lib/listut.pl compiled 8512 bytes in 0.03 seconds
util.eco   loaded in 0.00 seconds
var_name.eco loaded in 0.01 seconds
modello.ecl compiled 302312 bytes in 0.74 seconds
```

For defining sensible ranges for the activities by running the amplified plan, type `plan_amplified(2013)` under "Query Entry" and then click on run. In "Results" you should obtain:

```
?- plan_amplified(2013).
Yes (0.00s cpu, solution 1, maybe more)
```

Finally, you can run the Global Optimizer by typing `p(0.5)` in "Query Entry". Then, click on run. A new window like the one in Figure 11 should appear reporting the generated output.

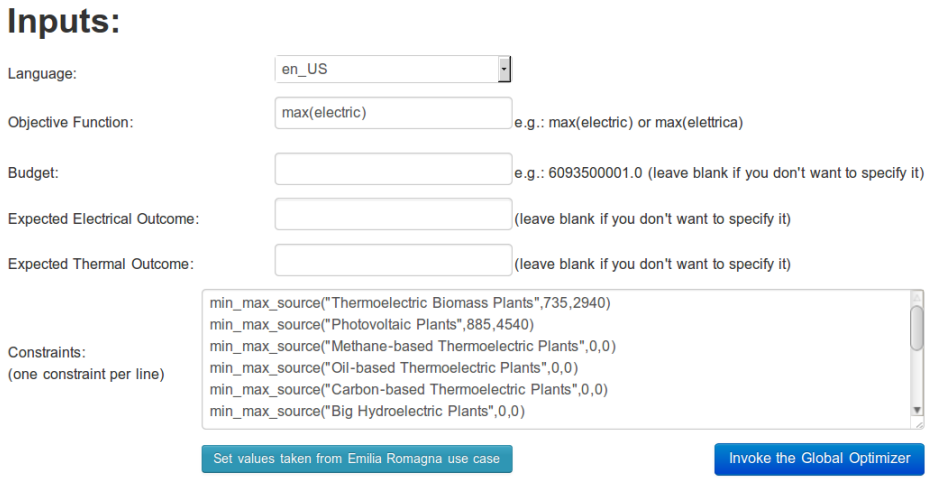

Figure 12: Screenshot of the *Input* section of the Global Optimization Module Testing Page.

5 Global Optimization Module Testing Page

The Global Optimizer can be invoked through the *Global Optimization Module Testing Page* available at

`http://globalopt.epolicy-project.eu/GlobalOptClient/ePolicy.jsp`

The service available at this link is for **testing** purposes only. Notice that it does not represent the Graphical User Interface (GUI) of the ePolicy prototype. The ePolicy GUI will be based on advanced visualization techniques developed within Work Package 7 (WP7). The service allows for the generation of a plan by invoking the predicate *compute_plan*, described in Section 3.3.2.

As explained in Section 3.3, one should define limits for the energy sources, then invoke the Global Optimizer. In order to simplify the definition of limits, the test interface provides a button *Set values taken from the Emilia-Romagna use case*. After that, one should define further input (detailed below), and finally *Invoke the Global Optimizer*.

Let us now enter into the details of the expected inputs and of the generated outputs.

5.1 Inputs

Figure 12 reports a screenshot of the Input section in the Global Optimization Module Testing Page. Specifically, these are:

Language. the user can specify the desired language. Currently the supported languages are English and Italian. By choosing one of them, inputs are expected to be specified in the selected language. Also outputs will be produced accordingly.

Objective function. This is a **required** input and it follows the same syntax given in Section 3.3.2. Briefly, the *objective function* represents the function that the user wants to optimize and it can be in the form $min(Term)$ or $max(Term)$, where $Term$ is a linear combination of (one or more) of the following: cost (in Italian, *costo*), electric (it: *elettrica*), thermal (it: *termica*), rec(X) (it: $ric(X)$) where X is the index of a receptor.

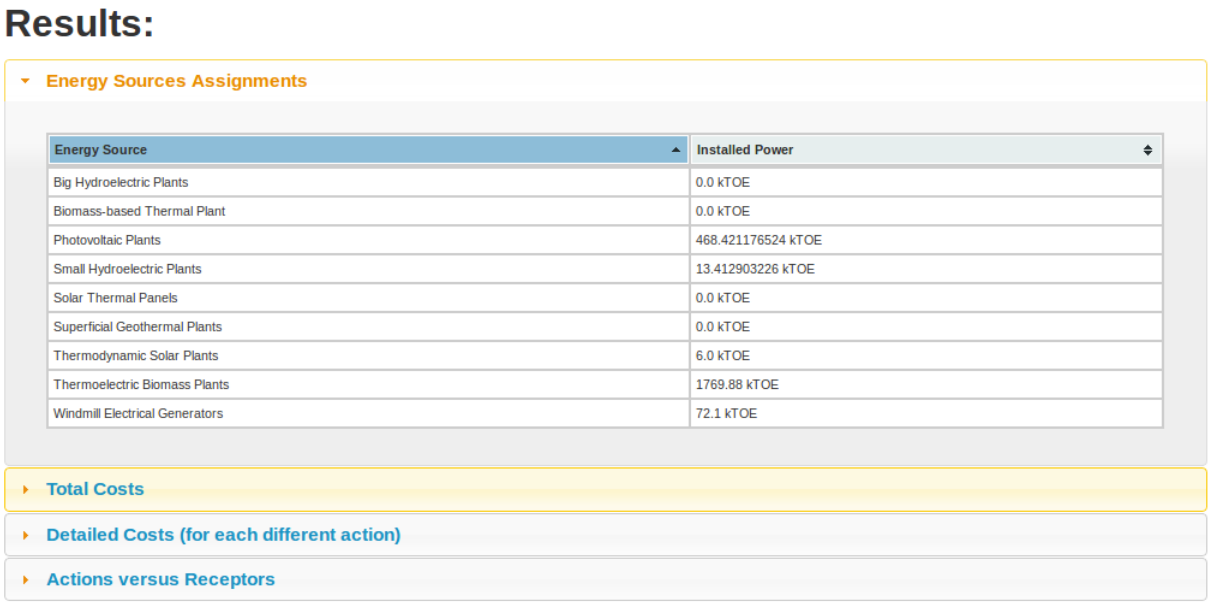

Figure 13: Visualisation of the results in the Global Optimization Module Testing Page.

Budget. This input is optional and represents the budget for the plan, in M€. If specified, it represents an upper limit for the cost of the plan (the total cost of the plan cannot exceed the budget).

Expected Electric Outcome. It represents the energy (in kTOE) that should be produced as Electrical Energy. This input is optional.

Expected Thermal Outcome. Similar to *Expected Electric Outcome*, but for thermal energy.

Constraints. In this area the user can specify a set of constraints the plan must satisfy (new constraints must be specified on new lines). No syntactical checks are performed on the way constraints are specified. Among the constraints that can be specified there are the limits for the energy sources, which can be given as a set of predicates of the form

$$min\_max\_source(Activity, Min, Max)$$

Alternatively, the limits corresponding to *plan_amplified(ele,2020)* can be loaded by pressing the button "Set values taken from the Emilia Romagna use case".

Once inputs are defined it is possible to compute the plan by pressing the button "Invoke the Global Optimizer".

5.2 Outputs

The output produced by the Global Optimizer are organized in four parts described below.

Energy Sources Assignments. Reports the list of Energy sources and, for each of them, the produced energy (in kTOE).

Total Costs. Reports the total costs for primary and secondary activities for producing the quantity of energy reported in the table "Energy Sources Assignments".

Detailed Costs. For each activity the table reports the quantity for that activity and the corresponding cost. At the top of the table it is possible to select the option "hide zero-value quantities" which makes the table more compact.

Actions versus Receptors. This table provides the impact of each activity on each of the receptors. At the top of the table it is possible to select how to visualize the results. Possible alternatives are: (i) showing the values; (ii) coloring cells (using green, red and white respectively for values grater, lower and equal to zero); (iii) showing both colors and values.

For each table it is possible to order the rows according to the values of a certain column by clicking on the header of the corresponding column. In this way rows can be ordered in ascending or descending order.

References

[1] K. Apt and M. Wallace. *Constraint logic programming using ECL^{i}PS^e*. Cambridge University Press, 2007.

[2] Paolo Cagnoli. *VAS valutazione ambientale strategica*. Dario Flaccovio, 2010.

[3] Michela Milano, Attilio Raimondi, Marco Gavanelli, and Fabrizio Riguzzi. Design of the global regional model and its instantiation on the energy plan. Deliverable 3.1, ePolicy project, October 2012. `http://www.epolicy-project.eu/system/files/PUBLIC/deliverables/D3.1.pdf`.

[4] Regione Emilia-Romagna. Il secondo piano triennale di attuazione del piano energetico regionale 2011-2013, July 2011. `http://bur.regione.emilia-romagna.it/dettaglio-inserzione?i=f1fc8c3c7f211ad02579c95ad8dc317a`.

[5] Joachim Schimpf and Kish Shen. ECL^{i}PS^e - from LP to CLP. *Theory and Practice of Logic Programming*, 12(1-2):127–156, 2012.

[6] Kish Shen and Joachim Schimpf. Eplex: Harnessing mathematical programming solvers for constraint logic programming. In P. van Beek, editor, *Principles and Practice of Constraint Programming - CP 2005*, volume 3709 of *Lecture Notes in Computer Science*, pages 622–636, Berlin/Heidelberg, 2005. Springer-Verlag.

[7] Jens C. Sorensen and Mitchell L. Moss. Procedures and programs to assist in the impact statement process. Technical report, Univ. of California, Berkely, 1973.