

General notes on unit commitment instance generator

Luigi Poderico (lupoderi@tiscali.it)

Introduction

This document describes `ucig`, a random generator of “realistic” instances of the hydro-thermal Unit Commitment problem in electric power generation. The generator has been used for the experimental result presented in the papers:

- A. Borghetti, A. Frangioni, F. Lacalandra, C.A. Nucci, “Lagrangian Heuristics Based on Disaggregated Bundle Methods for Hydrothermal Unit Commitment”, IEEE Trans. on power systems, Vol. 18, no., pp 313-323, 2003
- A. Borghetti, A. Frangioni, F. Lacalandra, C.A. Nucci, P. Pelacchi, “Using of a cost-based Unit Commitment algorithm to assist bidding strategy decisions”, 2003 IEEE Bologna Power Tech Conference, June 23th-26th, Bologna, Italy.

These papers describe also the model of unit commitment which the instance generator refers to.

Compiling

The unit commitment instance generator `ucig` is a simple c++ program composed by a unique source file `ucig.cpp`.

For this reason the compilation is very easy and don't need particular attention. For example, under linux it's sufficient the following command:

```
gcc ucig.cpp -o ucig
```

Under windows the compilation depends by the development tools. With the Visual C++ 6.0 it's enough to create a new empty project of kind “win 32 console application”, add `ucig.cpp` in the project and to compile.

The only possibly tricky part is about the ability of the generator to output the instances in XML format (see the `--xml` option below); this requires the `xerces` open-source XML library, that need to be present in the system and properly linked to the executable. Since a plain text output option is also available which does not require `xerces` at all, the C macro `HAVE_XML_OUTPUT` is defined at the beginning of the file; leaving it to the default value 0 disables XML output and does not require linking with the XML library, setting it to a nonzero value enables XML output and requires the XML library.

In the following, the compiled program will be called `ucig`.

Using `ucig`

`Ucig` is a simple program callable by command line with the following syntax:

```
ucig [-h | --help] |  
[--Ic <integer>] [--Gg <integer>] [--Breaks <integer>]  
[--Gmax <integer>] [--Imax <integer>] [--Cmax <integer>]  
[--Seed <integer>] [--CSC <integer>] [--Xml <integer>]  
[--AMin <double>] [--AMax <double>] [--Difficulty <integer>]
```

where:

- | | |
|---------------------|---|
| <code>Ic</code> | is the number of instance to be generate, 0 for MAXINT. [1] |
| <code>Gg</code> | is the number of days of horizon. [1] |
| <code>Breaks</code> | is the number of subdivisions for each day. [24] |

Gmax	is the number of thermals units. [10]
Imax	is the number of hydro units. [10]
Cmax	is the number of hydro cascade units. [0]
Seed	is the seed for the random generator number. [0]
CSC	1 for constant start up cost, 0 otherwise. [0]
Xml	1 the produced result is in xml format, 0 in textual format. [0]
Amin	is the minimum value of the quadratic coefficient for thermal units: typical range [0.0001,0.0005]. [1e-005]
Amax	is the maximum value of the quadratic coefficient for thermal units: typical range [0.001,0.05]. [0.1]
Difficulty	is the Difficulty-Level (1 or 2 or 3) of the output data file. [1]

Note that at the end of each lines the default parameters is shown enclosed in squared bracket.

Load profile

Ucig use the file *perc.dat* that contain useful information for the energy demand profile, called *load profile*.

The file *perc.dat* contains a row for each breaks, each row contains a record (*i*, *b_i*) with space separated fields. The field *i* is an integer used for improve the legibility, the field *b_i* is a real number between 0 and 1. An example of *perc.dat* file is:

```

1 0.13
2 0.15
3 0.17
4 0.22
5 0.30
6 0.44
7 0.60
8 0.74
9 0.90
10 0.92
11 0.94
12 1
13 0.95
14 0.54
15 0.70
16 0.87
17 0.97
18 0.99
19 0.86
20 1
21 0.85
22 0.55
23 0.34
24 0.20

```

The field *b_i* tells that in every considerate days, at *i*-th break will be assigned a load approx equal to $b_i * D$, where *D* is maximum power generates by the power units.

Output file format

Ucig generate the output on the standard output; calling *ucig* as:

```
ucig --Ic 1 --Gg 2 --Breaks 10 --Gmax 2 --Imax 3 --Cmax 2
```

the output generated is:

```
ProblemNum      0
```

```

HorizonLen      20
NumThermal     2
NumHydro       3
NumCascade     2
LoadCurve
MinSystemCapacity 132.63
MaxSystemCapacity 965.27
MaxThermalCapacity 405.588
Loads 2 10
307.841 309.775 251.72 151.575 151.575 159.417 227.897 288.606 341.329 362.633
151.575 151.575 151.575 151.575 151.575 178.404 232.075 280.024 323.124 372.474
SpinningReserve 10
0.0793793 0.0971151 0.0827699 0.0947514 0.0703931
0.0784057 0.0654399 0.0984396 0.0601816 0.0886065
ThermalSection
0 0.000125959 8.47794 447.838 80.8008 233.187 4 6 6
208.931 192.879 1.00858 5 58940.1 93380.2 167.727
1 0.0485937 7.14811 345.723 51.8293 172.401 1 2 3
239.531 178.164 1.88141 5 40302.3 51170.5 68.5504
HydroSection
0 1.01484 6.08508 162.173 159.802 225.449 45.6677 421.628
30.4096 34.1051 36.1312 26.7482 26.6108 16.7201 41.1853 21.8502 40.0673 31.246
42.9469 33.5477 26.4026 42.5321 22.4477 30.5453 22.4912 41.0752 26.4598 27.5061
1 1.03721 5.45551 201.851 194.609 298.024 61.7249 559.204
42.4451 34.1532 47.4486 20.6062 41.965 28.1956 44.7556 48.4131 25.2792 45.2077
34.1126 33.1938 24.3282 36.8212 35.3214 36.0302 22.316 47.9132 49.4681 38.1734
2 1.02544 5.19619 195.658 190.804 349.244 70.1036 607.1
21.9377 48.5681 22.1609 29.6121 40.0042 40.8806 35.4053 30.8747 45.5579 33.3071
19.7163 26.2798 47.1741 32.1546 43.7074 31.5493 50.9241 43.1662 34.9967 45.297
HydroCascadeSection
0 CascadeLen 2
0 1.01841 5.21441 216.38 212.469 509.141 149.037 906.007
40.6101 37.0902 49.199 57.2977 31.3688 35.1657 54.8012 48.0506 52.4852 55.0929
38.3713 54.4982 48.7372 23.6957 24.9768 45.6881 36.0201 31.2145 55.1723 25.8256
51.6954 57.3079 44.8995 31.6343
1 1.0315 5.25278 173.814 168.506 400.068 95.5917 698.804
42.0132 43.4855 29.4877 17.3825 36.915 20.3604 42.5424 42.6423 22.496 29.8054
18.7045 37.5917 43.6889 29.251 30.1771 25.8726 31.3146 29.3248 31.266 23.7991
18.5776 23.1008 44.8408 30.2239
1 CascadeLen 4
0 1.03867 6.0367 211.174 203.311 416.222 86.1507 713.983
26.8092 51.9972 55.5023 54.3925 54.651 41.0379 26.4118 38.2636 35.8118 43.8415
55.7944 28.8918 33.9268 54.2318 52.7399 40.2311 54.0559 40.7849 35.7195 48.5681
54.4338 22.7699 33.7433 44.4507
1 1.00422 5.70088 226.024 225.075 479.704 116.11 826.582
35.5126 38.1007 51.7489 42.7875 59.2582 40.524 26.056 40.1177 41.9317 56.2651
30.3894 43.9295 56.4562 60.1321 51.9497 22.933 31.5482 32.7359 41.1431 35.6148
42.9486 37.0621 60.982 23.2792
2 1.02048 5.30606 225.47 220.945 401.813 82.1554 712.193
50.049 44.0475 27.8754 45.4611 41.219 60.5558 48.2483 41.4621 48.0843 59.3605
58.9581 23.9837 25.8246 55.7721 42.8651 54.7408 48.4253 36.4388 52.199 37.1763
51.1582 40.7458 27.2028 33.3684
3 1.01298 6.05599 228.156 225.233 454.513 124.1 775.228
51.5399 59.8688 37.9169 55.1522 51.0587 37.8832 57.8275 48.944 44.6773 40.6595
57.4137 31.7532 56.7713 52.3687 50.2275 45.0141 51.825 51.1609 33.7067 44.5329
61.8139 26.12 41.7722 23.5782

```

Let we analyze the different file sections.

General information

Contains global information for the overall problem.

ProblemNum	Seed used by the generator.
HorizonLen	Length of problem temporal horizon (Gg * Breaks).
NumThermal	Number of thermal units.

NumHydro	Number of hydro units.
NumCascade	Number of cascade hydro units.

Load curve

MinSystemCapacity	Sum over minimum power generated by all units.
MaxSystemCapacity	Sum over maximum power generated by all units.
MaxThermalCapacity	Sum over maximum power generated by all thermal units.
Loads	For every day a row will be printed with a load value for each breaks of the day.
SpinningReserve	For each breaks of the day will be printed the percentage of loads used as spinning reserve.

Thermal unit description

A row for each unit is printed. A row contains in this mandatory order:

1. A unit index.
2. The quadratic, linear and constant coefficient used for the calculation power generation cost.
3. The minimum and maximum power.
4. The initial unit status. A positive integer t to indicate that the unit is on by t unit times; a negative integer $-t$ to indicate that the unit is off by t unit times.
5. The other parameters are used to calculate the start up cost. Called as coolAndFuelCost, hotAndFuelCost, tau, tauMax, fixedCost, SUCC, P0:
 1. the cool start up cost is equal to $\text{coolAndFuelCost} * (1 - \exp(-\text{downTime}/\text{tau})) + \text{fixedCost}$;
 2. the hot start up cost is equal to $\text{hotAndFuelCost} * \text{downTime} + \text{fixedCost}$;
 3. some parameters are unused but still present due compatibility with old format.

Hydro unit description

A hydro unit is described with two rows. In the first row we will find the parameters id, volumeToPower, b_h, maxUsage, maxSpillage, initialFlood, minFlood, maxFlood:

id	A unit index.
volumeToPower	Liner conversion coefficient from <i>water</i> to <i>power</i> .
b_h	Unused.
maxUsage	Maximum amount of water usable for power generation.
maxSpillage	Maximum amount of spillable water.
initialFlood	Initial amount of water into the basin.
minFlood	Minimum amount of water into the basin.
maxFlood	Maximum amount of water into the basin.

The second row of hydro unit description contains the amount of water that flows into the basin during the different breaks.

Hydro cascade unit description

Each cascade unit description is made of several rows. The first row contains:

1. The cascade unit index.
2. The token *CascadeLen*.
3. The number of single hydro unit composing the cascade.

Now for each composed hydro unit the description of a simple hydro unit is repeated.