EVOLUTIONARY METHODS FOR DESIGN, OPTIMIZATION AND CONTROL
P. Neittaanmäki, J. Périaux and T. Tuovinen (Eds.)
© CIMNE, Barcelona, Spain 2007

SOLUTIONS TO THE UNIT COMMITMENT PROBLEM USING TWO–LEVEL EVOLUTIONARY ALGORITHMS

Charikleia A. Georgopoulou* **Kyriakos C. Giannakoglou**

Parallel CFD & Optimization Unit, Lab. of Thermal Turbomachines,
National Technical University of Athens (NTUA)
P.O. Box 64069, Athens 157 10, Greece
**email: chgeorg@mail.ntua.gr*

Abstract. This paper presents two optimization methods, both based on Evolutionary Algorithms (*EAs*), for the solution of power generating Unit Commitment (*UC*) problems. The proposed methods overcome the inefficiency of conventional *EAs* to optimize the scheduling of power generating systems due to the high number of variables and constraints involved, as demonstrated on two test problems.

Key words: Unit Commitment, Evolutionary Algorithms, Multilevel Search.

1 INTRODUCTION

UC is a combinatorial problem[1] dealing with the optimal scheduling of generating units in power systems, aiming at min. power generation cost. The objective function is defined as the sum of production, start–up and shut–down costs. Operating constraints related to the system power balance, min. and max. production capacities of committed units, min. up/down times, etc. must be satisfied. To solve the *UC* problem, deterministic (Lagrangian relaxation[2], mixed integer–real linear programming[3], *MILP*) and stochastic methods (simulated annealing[4], *EAs*[5]) are in use. *EAs* are suitable for the *UC* problem due to their ability to handle highly constrained, non–linear objective functions. However, the presence of so many decision variables and constraints causes a noticeable degradation in the convergence rate of *EAs* which, for large scale problems, may occasionally be trapped in sub–optimal solutions.

Aiming at overcoming the aforementioned weakness of conventional *EAs*, the present paper proposes and compares two two–level solution methods for the *UC* problem, both inspired by multilevel *EAs*[6] that split the search among levels. So, the two–level *UC* method first solves an appropriately coarsened problem (derivative *UC* problem with a reduced number of decision variables) to get "promising" solutions which are, then, fed into the second level coping with all decision variables

and constraints. An inter-level, one– or two-way communication scheme for the exchange of information between the two *EAs* or levels, is needed. The first method proposed is based on two concurrently evolving *EAs*, one at each level; in the second method, the first level search is carried out once to provide good starting solutions to the second level which undertakes the detailed search.

2 PROBLEM FORMULATION

Consider the scheduling of M power generating units on an hourly basis over the period of T hours. The *i-th* power generation unit, $i \in [1, M]$, along with its known min. and max. power production capacities are abbreviated to u_i, $P_{min,i}$ and $P_{max,i}$, respectively. Also, $\overline{T}_i$, $\overline{\overline{T}}_i$ and $\widehat{T}_i$ are used to denote the min. duration of u_i's start–up (*STUP*), shut–down (*SHDN*) phases and the min. time for u_i to remain in *ON* or *OFF* condition. During u_i's *STUP*/*SHDN*, the fuel consumptions in *MW* are abbreviated to $\overline{C}_i$ and $\overline{\overline{C}}_i$, respectively. With $t^{(j)}$ we denote the *j-th* hour of the time period, $j \in [1, T]$; $d^{(j)}$ and $p^{(j)}$ stand for the hourly power demand and production in *MW*. The u_i's state at $t^{(j)}$ is denoted by the binary variable $s_i^{(j)}$; the *ON* state corresponds to $s_i^{(j)}=1$ and all the rest (*OFF*, *STUP* and *SHDN*) to $s_i^{(j)}=0$; however, one may distinguish them using data for the previous or next unit states.

The chromosome used by the binary encoded *EAs* consists of *MT* binary digits, which represent $s_i^{(j)}$. The economic dispatch problem which computes the $x_i^{(j)}$ ($\frac{P_{min,i}}{P_{max,i}} \leq x_i^{(j)} \leq 1$) values so as to achieve power balance at min. cost, is solved using the augmented Lagrange multipliers method (*ALM*[7]), once all $s_i^{(j)}$ values at $t^{(j)}$ are known. The *UC* problem targets at the minimization of the total operating cost (*TOC*, in *MW*), which sums up fuel, *STUP* and *SHDN* costs, as follows

$$TOC = \sum_{j=1}^{T} \sum_{i=1}^{M} OC_i^{(j)} + \sum_{j=1}^{T} \Phi(|d^{(j)} - p^{(j)}|) \tag{1}$$

where $OC_i^{(j)} = a_i {x_i^{(j)}}^2 + b_i x_i^{(j)} + c_i + \overline{C}_i{}'$ if $s_i^{(j)}=1$ or $OC_i^{(j)} = \overline{\overline{C}}_i{}'$ otherwise. $\overline{C}_i{}' = \overline{C}_i$ if $s_i^{(j-1)}=0$ and $\overline{\overline{C}}_i{}' = \overline{\overline{C}}_i$ if $s_i^{(j-1)}=1$ and $(a_i, b_i, c_i), \forall\, i \in [1, M]$ are known coefficients. $\Phi()$ is a penalty function expressed by a quadratic polynomial. If a *UC* schedule fails to satisfy either the initial conditions of any unit or the imposed min. *STUP*, *SHDN* and "state–change delay" time intervals ($\overline{T}$, $\overline{\overline{T}}$, $\widehat{T}$), the candidate chromosome undergoes repairing as follows

$$s_i^{j+1} - s_i^j = \begin{cases} 1 & if \quad s_i^{j+k}=1, \quad \forall k \in [1, \overline{T}_i + \widehat{T}_i] \\ -1 & if \quad s_i^{j+k}=0, \quad \forall k \in [1, \overline{\overline{T}}_i + \widehat{T}_i] \end{cases} \qquad i \in [1, M],\ j \in [1, T] \tag{2}$$

3 GENERAL CONCEPT OF THE TWO–LEVEL *EA* ALGORITHMS

The proposed two–level *EAs* are conceptually similar and differ only in the inter–level communication scheme as well as the use or not of segregated search at the second level. At the **first level**, an *EA* solves a coarsened variant of the *UC* problem, aiming at acquiring general knowledge about promising regions of the search space at low CPU cost, thanks to the short bit strings the *EA* is dealing with. The coarsened

problem parameterization unavoidably leads to relaxed constraints. For instance, constraints related to the min. time needed for $STUP/SHDN$, "state–change delay" and the initial unit states are not taken into consideration, leading thus to infeasible "optimal" solutions. At the **second level**, the fine–grained UC problem, i.e. the whole time interval of T hours, is solved using EAs, by taking into account all constraints. In order to increase EAs efficiency, all the information gained at the first level is exploited.

3.1 First Level Preparatory Phase

Both methods start with the same three–step preparatory phase that defines the coarse–grained problem. **(1)** For each and every time unit (hour) of the time interval T, the "optimal" units' schedule is determined without taking into account the power demands or constraints dictated by the previous and next unit states. The optimization is carried out using $MILP$-branch–and–bound[8], after the objective function linearization. **(2)** Consecutive hours within the time period which were given the same "optimal" solutions (in the sense that the same units are ON) are grouped together to form coarser time units. **(3)** This step is optional and helps to additionally coarsen the outcome of Step 2. Time units are agglomerated using heuristics, until a coarse problem with desirable size is reached.

The proposed algorithms use $EASY$ as optimization tool at both levels. $EASY$[6] is a general–purpose optimization software developed and brought to market by the National Technical University of Athens. It employs a generalized (μ, λ) EA, where μ and λ are the population sizes of parents and offspring, respectively.

3.2 The Concurrent Two–Level *EA* (C2LEA) Algorithm

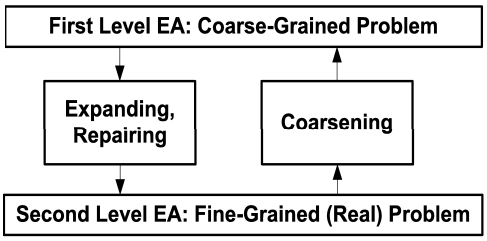

Figure 1: C2LEA flowchart.

In $C2LEA$, fig. 1, the first and second levels are concurrently solved by means of two regularly communicating EAs; promising solutions "migrate" from one level to the other according to a user–defined frequency. Any coarse solution which migrates to the second level requires expansion over T time units and repairing to satisfy all operating constraints. On the other hand, promising solutions which migrate to the first level must be coarsened.

3.3 The Sequential Two–Level *EA* (S2LEA) Algorithm

In the $S2LEA$, the EA–based search is sequentially employed; practically, the second level optimization follows the first level one, without feedback. However, there is another distinguishing feature, namely the use of time period partitioning at the second level and the use of successive optimizations between these partitions. **(1)** At the first level, the coarsened problem, as previously defined, is optimized. The so–obtained "optimal" coarse solution is expanded and repaired, using the same operators as in the previous method. The repaired solution is injected into the randomly generated population of the first generation of the second level EA.
(2) In order to reduce as much as possible the complexity of the second level op-

timization, the entire time interval is partitioned into K sub–intervals of the same (or almost the same) size. Each one of these intervals is optimized using *EAs*, with a chromosome of *TM/K* bits and a modified objective function. The objective function used by each *EA* takes into account possible inconsistencies at the interfaces of neighboring sub–intervals. All sub–intervals are iteratively and sequentially optimized; this procedure terminates when no change to the optimal solution occurs and all matching conditions are satisfied. By the way of example, at the second level, a *UC* of 48 hours can be solved either by handling two sub–problems with 24 hours each or four sub–problems with 12 hours each, etc.

4 Case Study I

The first problem is concerned with the scheduling of a four unit system over a period of T=48 hours, to cover the power demand distribution shown in fig. 3. Case specific data are given in Table 1. During the preparatory phase, 48 single–hour optimization problems were solved using the *MILP* method. Six classes of different operating scenarios were found. According to these classes, at the coarsening step 30 small duration groups were created, fig. 1(right–top). The so–obtained reduction (from 48 down to 30 time units, with durations between 1 and 6 hours each) was inadequate. The heuristic agglomeration was then employed which produced only 10 groups, as shown in fig. 1(right–bottom). Thus, the coarsened *UC* solution was parameterized using a bit string of only $4\times10=40$ digits.

i	$P_{min,i}$	$P_{max,i}$	Operating Cost	$\overline{C}_i$	$\overline{\overline{C}}_i$	$\overline{T}_i$	$\overline{\overline{T}}_i$	$\widehat{T}_i$	s_0
1	25	50	$4x^2$+60x+80	120	100	2	2	1	*OFF*
2	35	75	$9x^2$+65x+85	180	150	2	2	1	*OFF*
3	40	100	$4x^2$+78x+112	240	200	2	2	1	*SHDN*
4	60	120	$3x^2$+150x+100	240	200	2	2	1	*ON*

Table 1: Case I: Problem definition (all costs are in *MW*).

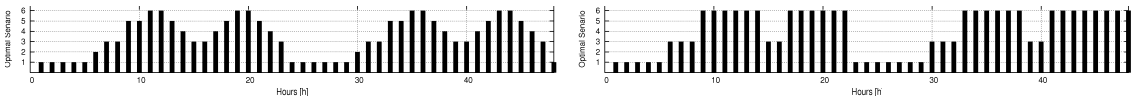

Figure 2: Case I: Initial (left) and final (heuristics–based; right) grouping of time units.

The problem was solved using both proposed methodologies, with (μ,λ)=$(20,60)$ at both levels, as well as conventional *EAs*, with the same (μ,λ), see fig. 3. *C2LEA* and *S2LEA* reached good solutions within the first 2000–3000 evaluations and, then, their refinement became slow as an *EA* cannot exploit local information. Conventional *EAs* required more than 5000 evaluations to even reach a feasible solution. In *C2LEA*, a two–way migration of the 5 top solutions at each level, every 10 generations, was used. In *S2LEA*, at that second level, three sub–intervals of 16 hours each were defined and solved sequentially. The final schedules computed using *C2LEA* and *S2LEA* slightly differ with respect to the total operating costs, i.e. 23289 *MW* for *C2LEA* and 23179 *MW* for *S2LEA*. With the conventional *EA* the best cost reached was equal to 24165 *MW*, after 26000 evaluations.

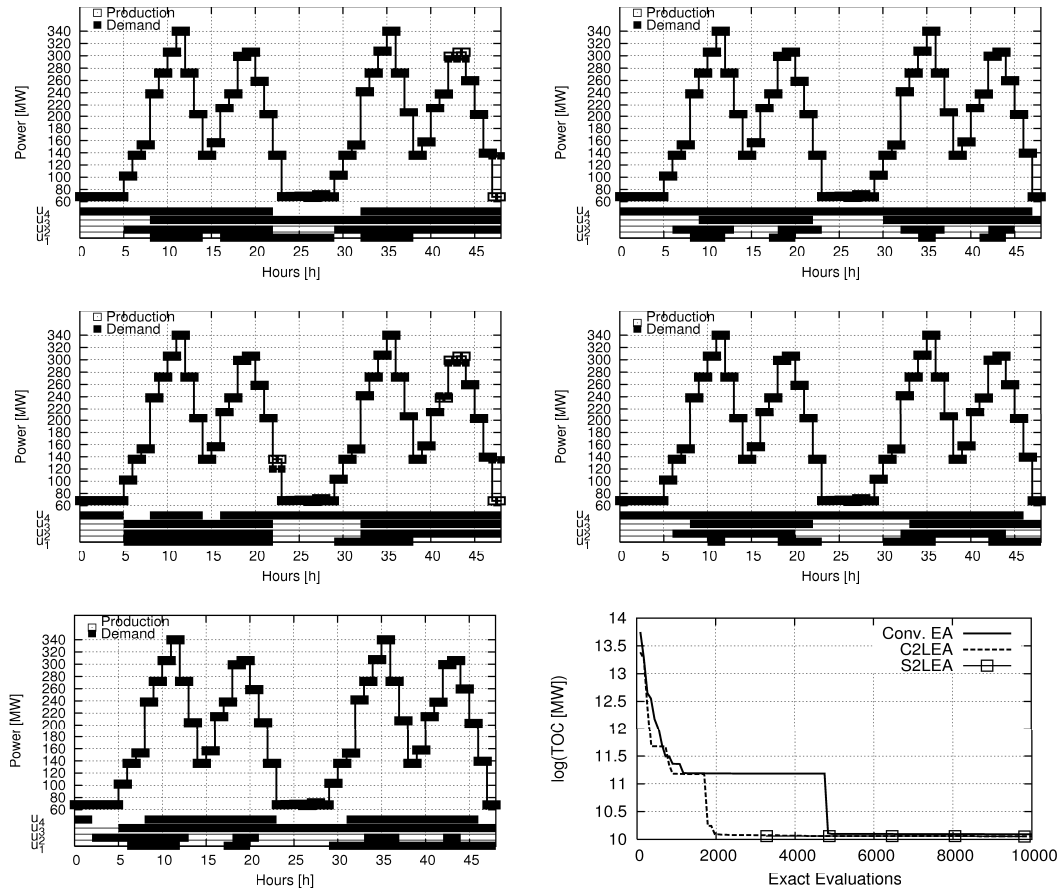

Figure 3: Case I: Top: best solutions for coarse (left) the fine (right) problems using the C2LEA. Mid: best solutions for coarse (left) the fine (right) problems using the S2LEA. Bottom: best solution from a conventional EA (left) and convergence comparisons of the three methods (right).

5 Case Study II

The second test case is a scaled variant of the first problem, with 10 units scheduled for a period of T=72 hours. The preparatory agglomeration step resulted to 18 different classes which finally defined 11 coarse time units, fig. 4. So, the chromosome of the coarse–grained problem consisted of 110 digits, instead of 720 digits of the fine–grained one. The two proposed methods outperformed conventional *EAs*, as shown in fig.5. The final optimal second level solutions differ (77451 MW for $C2LEA$ and 77114 MW for $S2LEA$).

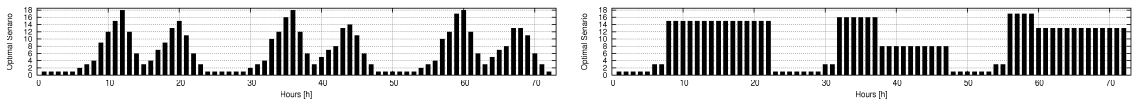

Figure 4: Case II: Initial (left) and final (heuristics–based; right) grouping of time units.

6 Conclusions

Two methods which can be used to optimize power unit commitment problems, have been presented. Both are two–level methods relying on *EAs*, where the first level handles a coarse optimization problem with agglomerated time intervals and

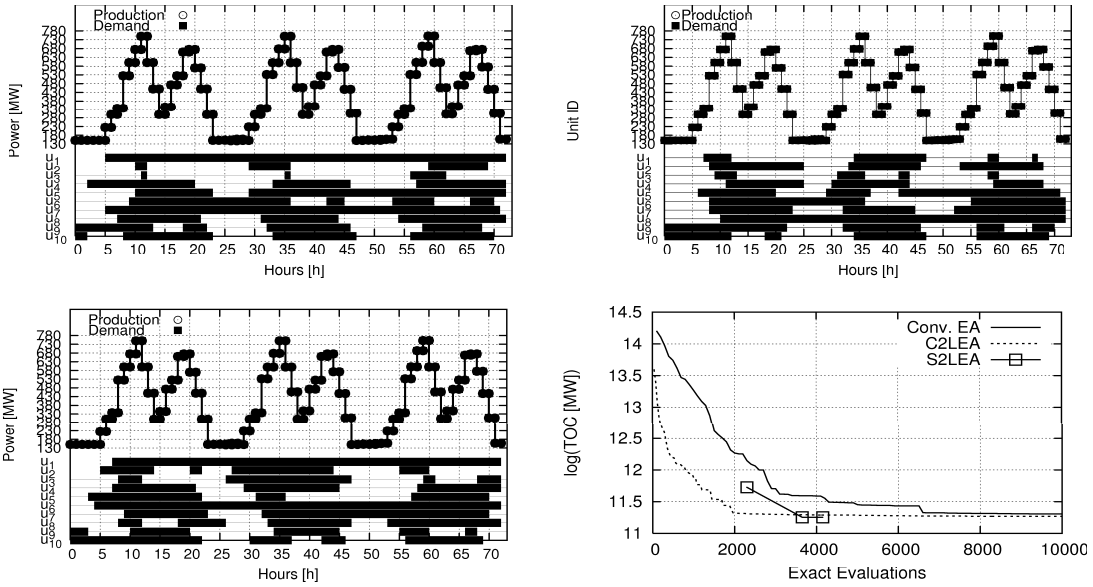

Figure 5: Case II: Top: best solutions using C2LEA (left) and S2LEA (right). Bottom: best solution from a conventional EA (left) and convergence comparisons of the three methods (right).

the second level solves the detailed scheduling problem. Both proved to considerably outperform conventional *EAs* which face major difficulties to locate feasible solutions. They both have similar convergence properties and any further comparison between them requires more tests to be carried out.

Acknowledgment: The project is cofinanced 75% of public expenditure through EC - European Social Fund, 25% of public expenditure through Ministry of Development - General Secretariat of Research and Technology and through private sector (Public Public Corporation), under measure 8.3 of Operational Programme "Competitiveness" in the 3rd Community Support Programme (PENED 2003).

REFERENCES

[1] N. Padhy, *Unit commitment - A bibliographical survey*, IEEE Transactions on Power Systems, Vol.19(2), May 2004.

[2] A. Bakirtzis et.al., *A genetic algorithm solution to the unit commitment problem*, IEEE Trans. Power Systems, Vol.11(1), 1996.

[3] A. Cohen, M. Yoshimura, *A branch-and-bound algorithm for unit commitment*, IEEE Trans. Power App. Syst., Vol.PAS-102, pp.444-451, Feb.1983.

[4] D. Simopoulos et.al.,*Unit commitment by an enhanced simulated annealing algorithm*, IEEE Trans. Power Systems, Vol.21(1), Febr.2006.

[5] A. Bakirtzis, C. Zournas, *Lambda of lagrangian relaxation solution to unit commitment problem*, IEE Proc.-Gener. Transm. Distrib., 147(2), March 2000.

[6] http://velos0.ltt.mech.ntua.gr/EASY

[7] J. Nocedal, S. Wright, *Numerical optimization*, Springer-Verlag NY, 1999.

[8] A. Land, A. Doig, *An automatic method for solving discrete programming problems*, Econometrica, 28, pp.497–520, 1960.