EVOLUTIONARY METHODS FOR DESIGN, OPTIMIZATION AND CONTROL
T. Burczynski and J. Périaux (Eds.)
© CIMNE, Barcelona, Spain 2009

SOLUTION OF STOCHASTIC UNIT COMMITMENT PROBLEMS USING METAMODEL-ASSISTED EVOLUTIONARY ALGORITHMS

Chariklia A. Georgopoulou **Kyriakos C. Giannakoglou**

National Technical University of Athens,
School of Mechanical Engineering,
Lab. of Thermal Turbomachines, Parallel CFD & Optimization Unit,
P.O. Box 64069, Athens 157 10, GREECE,
e-mail: kgianna@central.ntua.gr

Abstract. This paper proposes a Metamodel-Assisted Evolutionary Algorithm ($MAEA$) for solving power generating unit commitment (UC) problems with probabilistic unit outages and locating the scheduling scenario with the minimum expected total operating cost (TOC). The evaluation of candidate solutions relies on Monte Carlo simulations comprising a great number of randomly generated unit failure scenarios. Seeking the global optimal solution via a population-based search method, such as an EA, relying on CPU expensive Monte Carlo simulations (evaluation tool), is computationally demanding. The use of surrogate evaluation models (metamodels, such as artificial neural networks) to inexactly pre-evaluate the EA population and screen out its non-promising members without undergoing Monte Carlo based evaluations, is a way to reduce the high CPU cost. The originality of this paper lies on the selected inputs for the metamodels. For each candidate scheduling scenario, the TOCs corresponding to a few (about 10) representative failure scenarios are computed and, by presenting them to a trained metamodel, the expected TOC is predicted. Training patterns are recorded during previous generations. The optimization kernel is a two-level EA; on the low level, which is exploration-oriented, a coarsened problem without constraints is solved at negligible CPU cost. The high level is mostly used for the refinement of promising immigrants, sent by the low level. There, the full/constrained problem is solved through the decomposition of the scheduling period in subperiods, handled by a distributed EA, with matching conditions at the interfaces. The proposed algorithm is tested on the scheduling of a system with 6 units (gas turbines), for an 168 hour horizon.

Key words: Unit Commitment, Metamodel-Assisted Evolutionary Algorithms, Stochastic Outages, Monte Carlo Method.

1 INTRODUCTION

A *UC* problem is defined as follows: given a system of M power generating units (for instance, gas and/or steam turbines) and a power demand distribution ($d^{(j)}$, $1 \leq j \leq T$) during a T-hour period, find the scheduling of these units that satisfies the power demand with the minimum *TOC*. The problem is subject to various time constraints, such as the duration of the start–up (*STUP*: T_i^{STUP}) and shut–down (*SHDN*: T_i^{SHDN}) phases or the minimum time during which each unit must stay *ON* or *OFF* (T_i^{RAMP}). Candidate solutions are binary strings with $M \cdot T$ digits, corresponding to the *ON* and *OFF* hourly states of all units.

In [1], a two–level optimization technique based on a single-objective *EA* was proposed for solving the *UC* problems. The search is carried out on two levels. On the low level, a *UC* problem of reduced complexity is solved: the time period is coarsened by forming groups of consecutive time units (hours) whereas the time constraints are relaxed. Thus, a much simpler problem is solved, using *EAs*, at negligible *CPU* cost. Well performing solutions migrate from the low to the high level, where the full problem is solved using a distributed *EA*. It is called “distributed” since the scheduling horizon is decomposed into a small number of subperiods, each of which is solved by an *EA*. *EAs* optimizing the various subperiods communicate at their interfaces, where matching conditions should be met. The combind use of the distributed search scheme on the high level and the injection of promising solutions into the starting populations of the high-level *EAs* reduce noticeably the *CPU* cost. This is used as the basic search kernel in the present method, for solving *UC* problems with stochastic unit outages.

When the *UC* problem is subject to stochastic unit outages, the optimal scheduling scenario is the one that gives minimum expected *TOC*. A literature survey on the solution of the *UC* problem with probabilistic outages reveals that the Monte Carlo simulation is the most appropriate tool for evaluating candidate solutions, [2, 3, 4]. To compute the expected *TOC* of a candidate solution (binary string), an adequate number (denoted by J) of unit availability/unavailability scenarios, according to the failure and repair rates of each unit, must be generated at random. The binary string under consideration is adapted to each one of these scenarios and the corresponding *TOC*s are computed. The expected *TOC* is deduced through their averaging. Consequently, a single evaluation based on a Monte Carlo simulation costs as high as J times the cost of evaluating a single *UC* schedule with all units available. Employing the Monte Carlo technique within an *EA*, even a two-level one, increases significantly the *CPU* cost.

To maintain the advantages of the *EA*–based optimization method while extending it to cope with stochastic unit outages with a reasonably low *CPU* cost, the use of surrogate evaluation models within the two-level *EA* is proposed in this paper. Assisting the *EA*–based optimization methods for computationally demanding problems by metamodels is not new. Metamodel–based evaluations displace many calls to the costly evaluation software during the evolution, leading to very efficient *MAEA*, [5, 6, 7]. Metamodels are trained on-the-fly on data collected in the vicinity of each new offspring. They produce approximate scores which are used to screen out non–promising population members and restrict costly evaluations on the problem specific tool only to the most promising among them.

In a *UC* problem, where the design variables are binary digits, a new way of training and using metamodels is proposed. Since the target is to compute the expected *TOC* resulting from J (usually some thousands) randomly generated failure–repair scenarios, the metamodel is presented with the computed *TOC* values of a few ($J_{ind} \ll J$, of the order of $J_{ind} \approx 10$) preselected unit availability/unavailability scenarios and approximates the expected *TOC*. The new method is demonstrated on a test problem, by laying emphasis to the *CPU* cost reduction compared to a two-level *EA* without metamodels.

2 Problem Formulation

The *UC* problem with stochastic unit outages has already been defined in Section 1. For each one of the M units, the minimum ($P_{min,i}$, $1 \leq i \leq M$) and maximum ($P_{max,i}$) power production capacities are known. The values that T_i^{STUP}, T_i^{SHDN} and T_i^{RAMP} take on are also known. In addition, let AFR_i be the average failure rate (average number of failures per unit time) and ARR_i the average repair rate of unit i; both are assumed to be independent of time, [8].

In the *UC* problem, optimization variables are the states of all units (subscript i) at each hour (superscript j). These are represented by binary digits ($s_i^{(j)}=1$ or 0), denoting *generating* (*ON*) or *non–generating* (*OFF*, *STUP* and *SHDN*) states, respectively. Since $s_i^{(j)}$=0 corresponds to any of the three *non–generating* unit states (*OFF*, *STUP* or *SHDN*), these can be distinguished only by taking into account previous or next states of the same unit. Such a binary coding fits perfectly to the *EAs*. Since, however, the chromosome consists of $M \cdot T$ digits, a conventional *EA* becomes *CPU* demanding for high $M \cdot T$ values. For a given chromosome, the optimal loads $x_i^{(j)}$ of the generating units ($\frac{P_{min,i}}{P_{max,i}} \leq x_i^{(j)} \leq 1$) are deduced from the minimization of the hourly cost, i.e. by solving hourly economic dispatch problems. To this end, the augmented Lagrange multipliers method (*ALM*, [9]) is used.

In *UC* problems without stochastic unit outages, the *TOC* (in *MWh* or any currency) is the objective function to be minimized. This sums up fuel consumption and penalties for unmet demands, as follows

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

where the hourly operating cost $OC_i^{(j)}$ is usually given by polynomials with known coefficients. If the power demand is not met, *TOC* is penalized by $\Phi(\Delta d^{(j)})$, which is a user–defined polynomial of $\Delta d^{(j)} = |d^{(j)} - p^{(j)}|$ and $p^{(j)}$ is the power produced at the j–th hour. Before being evaluated, all chromosomes are repaired to satisfy time constraints related to T_i^{STUP}, T_i^{SHDN} and T_i^{RAMP} and, hence, these do not affect the penalty function.

Eq. 1 is valid irrespective of whether unit outages occur or not. To account for stochastic unit outages, the expected *TOC* must be minimized and, thus, a Monte Carlo simulation is used for the evaluation of each candidate schedule. For a unit which is committed at the beginning of the scheduling period, the probability of zero failures in the time interval (0,t) becomes $P_{fail,i}(t) = e^{-AFR_i \cdot t}$, with a failure density function given by $p_{fail,i}(t) = AFR_i \cdot e^{-AFR_i \cdot t}$. Also, the probability of repairing it

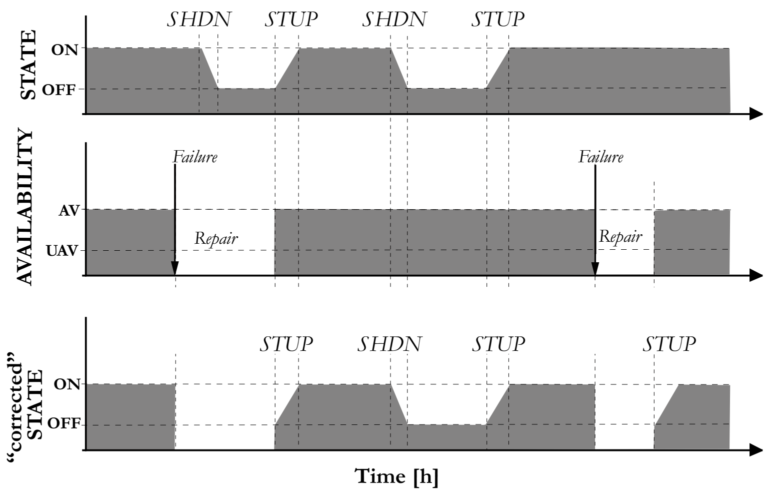

Figure 1: The *UC* problem with stochastic outages. A candidate solution (for a single unit; top) is adapted to a Monte Carlo *AV*–*UAV* scenario (middle), giving rise to a repaired chromosome (bottom) to be evaluated on the adopted cost model.

within t hours from the moment of failure is $P_{rep,i}(t) = e^{-ARR_i \cdot t}$, with a repair density function $p_{rep,i}(t){=}ARR_i \cdot e^{-ARR_i \cdot t}$. A Monte Carlo AV/UAV scenario is generated using a sequence of stochastic variables r following the uniform distribution and the expressions for the corresponding times-to-failure, $T_{fail,i} = -\frac{1}{AFR_i} ln(r)$, and times-to-repair, $T_{rep,i} = -\frac{1}{ARR_i} ln(r)$. The sequence of randomly selected variables r, determining the sequence of $T_{fail,i}$ and $T_{rep,i}$ values, assign availability (AV; from a repair up to the next failure) or unavailability (UAV; from a failure up to the repair) hourly flags to each unit. The binary digits of the UC chromosome to be evaluated must be adapted to the AV/UAV unit scenario, as shown in fig. 1. The repaired UC schedule is evaluated and the corresponding TOC is computed. These steps are repeated J times, by creating and evaluating J replicates. Finally, the expected TOC of this candidate solution is computed by averaging the TOC values of the J replicates.

To conclude, the Monte Carlo evaluation of a single population member of the EA costs as many as J schedule repairs and the solution of $J{\cdot}T$ economic dispatch problems. Even for short–term UC problems, this cost is important since J is usually of the order of 10^4.

3 The Two–Level *MAEA*

For the solution of the UC problem with probabilistic unit outages, the proposed method employs a two–level $MAEA$, inspired by the two-level EA for solving conventional UC problems, as proposed by the authors in [1].

The coarse variant of the problem to be solved on the low level is defined beforehand. Consecutive time units (hours), which may accomodate the same or similar scheduling scenarios, are clustered together to form T_c ($T_c \ll T$) coarse time units. The coarsened problem is optimized (low level optimization) via a binary encoded EA, without considering constraints related to the transient phases or starting states of units. Also, during the low level process, all units are considered to be continuously available. Because of the reduced number of variables ($M{\cdot}T_c$ bit string) the CPU cost of defining and solving the low level problem is negligible.No metamodels are used.

Prior to proceeding to the high level where the detailed problem with all constraints is solved, the optimal solution to the coarsened problem is expanded, repaired and evaluated on the expected TOC using a Monte Carlo simulation with J randomly generated AV/UAV scenarios. The J corresponding replicates are sorted in terms of their TOC values and classified into J_{ind} ($J_{ind} \ll J$) groups. An "indicative" (median) replicate from each group is selected.

On the high level, the system operation during T hours is optimized by considering all constraints, i.e. initial unit states and time constraints (T_i^{STUP}, T_i^{SHDN}, T_i^{RAMP}) through proper repairing of each candidate chromosome, as explained in [1]. To further reduce the CPU cost of the high level optimization, a distributed optimization scheme is used: the scheduling period splits into K (user–defined) subperiods, each of which is solved using a $MAEA$. Each subproblem is associated with a chromosome of $T \cdot M/K$ digits and an objective function which is the sum of the TOC of the corresponding partition and penalties for possible inconsistencies with the neighboring "optimal" schedules. The optimization process is employed sequentially and iteratively, until all matching conditions at the interfaces are met.

To describe the two steps of the $MAEA$–based optimization scheme, we will assume that the optimization is carried out on a single level (the low level) and that on the high level the decomposition of the scheduling period in subperiods is not activated. Thus, in the first step, the (μ, λ) $MAEA$ (with μ parents and λ offspring) starts exactly as a conventional EA by evaluating the first generation members on the Monte Carlo method (evaluation of J replicates) and archiving in a database, for each individual, only the TOC values of the J_{ind} representative replicates, paired with the expected TOC as computed using J replicates.

In second step, in each generation assisted by metamodels, the entire population is firstly evaluated on the J_{ind} representative replicates. Then, for each individual, a radial basis function network (RBF, [10]) is trained on a small number of previously evaluated individuals which are close to the new individual (distances are measured in $\Re^{J_{ind}}$). The trained network approximates the expected TOC by presenting the J_{ind} previously computed TOC values to its sensory units. Procedures for selecting "optimal" training sets for the RBF networks are recommended in [7, 5]. Once approximations to the expected TOC of all the population members have been computed, a few top of them (according to the metamodel predictions) are re–evaluated on the Monte Carlo method and the same, initially generated, J replicates. Let σ be the percentage of the offspring population selected for re–evaluation. The parent selection operator is applied to the λ offspring, among which the expect TOC values of $\sigma\lambda$ of them are "exactly" (Monte Carlo) evaluated and the rest are merely predicted by the RBF network. The offspring population of the next generation is formed through the standard evolution operators.

4 Method Application

In the examined case ([3]), the scheduling of M=6 units on a weekly period (T=168 hours) with the demand distribution of fig. 2 (top), is optimized. Units' features are all given in table 1. The terms $OC_i^{(j)}$ and $\Phi(\Delta d^{(j)})$ in eq. 1, are linear functions of

Table 1: Features of the power generation system.

	$[P_{min},\ P_{max}]$ $[MW]$	$OC_i(P_i)$ [\$]	T_{STUP} $[h]$	T_{RAMP} $[h]$	AFR_i $\times 10^{-4}$	ARR_i
u_1	[55.7, 137.0]	$2707.12P_1$	22	8	7.5	0.1554
u_2	[25.0, 107.0]	$2247.00P_2$	0	8	7.5	0.1554
u_3	[11.0, 51.0]	$1219.41P_3$	0	6	7.5	0.1554
u_4	[11.0, 51.0]	$1243.89P_4$	0	6	7.5	0.1554
u_5	[40.0, 162.0]	$5645.70P_5$	2	3	7.5	0.1554
u_6	[10.0, 46.0]	$2321.16P_6$	0	0	7.5	0.1554

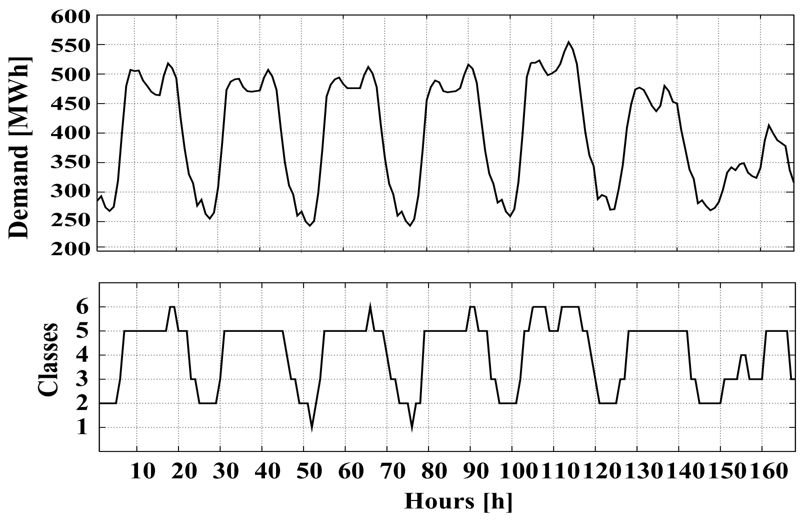

Figure 2: Demand (in MWh) distribution for the scheduling horizon (top) and definition of the coarse level problem (bottom).

the produced power (P_i, MWh) per unit and the unserved demand ($130\Delta d^{(j)}$ \$), respectively.

For the detailed problem in hand, each candidate solution is encoded using $6 \times 168 = 1008$ binary digits. This is a quite long chromosome that makes necessary the use of the two-level search method. During the preparatory phase, this chromosome is shortened significantly due to the coarsening process, which yields 46 coarse time units, fig. 2 (bottom). So, on the coarse level, a candidate chromosome comprises only 6×46=276 digits. On the high level, the 168 hour scheduling period is separated into 6 subperiods (of 28 hours each). Each subperiod is optimized using a *MAEA* which handles populations of $\mu=20$ parents, $\lambda=40$ offspring and chromosomes of 168 digits length. For the first two generations, all offspring members are evaluated on the Monte Carlo simulation based on J=8000 AV/UAV scenaria. Then, metamodels (*RBF* networks) are trained on J_{ind}=10 *TOCs*, corresponding to the repaired chromosome with respect to 10 “indicative” outage scenarios selected among the J replicates.

For solving the *UC* problem, the proposed two-level *MAEA* is compared to a conventional *EA* and a *MAEA*, where the proposed way to train metamodels is used on a single-level evolution scheme. Fig. 3 illustrates the convergence of the three algorithms. The conventional *EA* drifts slowly and endlessly towards a good solution. The single-level *MAEA* performs even better, improving the performance of the

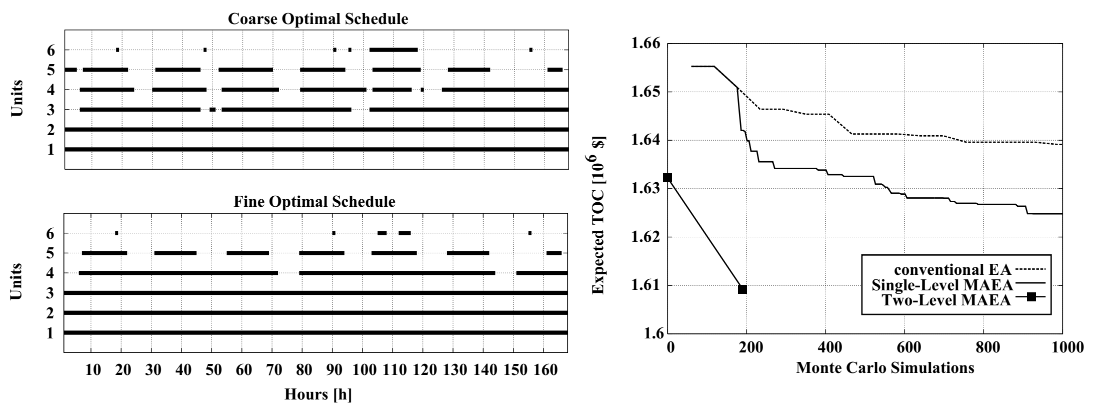

Figure 3: Left–top: Coarse level optimal solution captured with neglected *CPU* cost. Left–bottom: Fine level optimal solution found at the *CPU* cost of only 120 exact (Monte Carlo) evaluations. Right: Convergence of a conventional *EA*, a single-level *MAEA* and the proposed two-level *MAEA* in terms of the *CPU* cost in Monte Carlo simulations with J=8000 AV/UAV scenaria.

latter. The proposed two-level *MAEA* outperforms both single-level algorithms. The coarse level optimization results to a sufficiently good but sub-optimal solution, (fig. 3, left–top) at a negligible *CPU* cost. This solution serves as good initialization on the fine level, where only a few iterations are required to trace the optimal solution (fig. 3, left–bottom).

5 Conclusions

A two-level metamodel-assisted *EA* for the solution of the *UC* problem with stochastic unit outages was presented. The backbone for programming the proposed method is the generic multilevel optimization platform $EASY$[1], developed by the Lab. of Thermal Turbomachines of NTUA. This algorithm was inspired by the concept of hierarchical distributed *MAEAs*, initially developed for solving computationally demanding optimization problems in aeronautics. The evaluation software that computes the expected *TOC* (objective function) relies on the Monte Carlo method, which involves the evaluation of thousands of randomly generated availability/unavalability scenarios. The proposed method leads to considerable reduction in *CPU* cost thanks to the use of on-line trained metamodels. A novel way of using metamodels in optimization problems which involve Monte Carlo simulations was proposed. For each individual, instead of evaluating all Monte Carlo scenarios, it suffices to compute the *TOCs* of just a few pre-selected ones and, then, use a metamodel to guess the expected *TOC*. The optimization cost is further reduced by performing a two-level evolutionary search, with an exploration-oriented (low) and an exploitation- or refinement-oriented (high) level. On the high level, the use of distributed evolution improves even more the algorithms performance.

Acknowledgements This research was co–financed 80% of public expenditure through EC - European Social Fund and 20% of public expenditure through the Ministry of Development - General Secretariat of Research and Technology and the Public Public Corporation, under measure 8.3 of Operational Programme "Competitiveness" in the 3rd Community Support Programme (*PENED 2003*). The authors

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

sincerely thank Dr. E.T. Bonataki (Public Power Corporation, Thermal Projects Engineering & Construction Department, Athens) for her constructive comments and suggestions.

REFERENCES

[1] C.A. Georgopoulou, K.C. Giannakoglou, *Two-level, two-objective evolutionary algorithms for solving unit commitment problems*, Applied Energy 2009, Vol. 86: 1229-1239.

[2] N. Padhy, *Unit commitment - A bibliographical survey*, IEEE Transactions on Power Systems 2004, Vol.19(2): 1196–1205.

[3] M. Mazumdar, L. Chrzan, *Monte Carlo stochastic simulation of electric power generation system production costs under time-dependent constraints.*, Electric Power Systems 1995, Vol.35: 101–108.

[4] U.A. Ozturk, M. Mazumdar, B.A. Norman, *A solution to the stochastic unit commitment problem using chance constrained programming*, IEEE Transactions on Power Systems 2004, Vol.19(3): 1589–1598.

[5] K.C. Giannakoglou, *Design of optimal aerodynamic shapes using stochastic optimization methods and computational intelligence*, International Review Journal Progress in Aerospace Sciences, Vol. 38, pp. 43-76, 2002.

[6] M.K. Karakasis, A.P. Giotis, K.C. Giannakoglou, *Inexact information aided, low-cost, distributed genetic algorithms for aerodynamic shape optimization*, International Journal for Numerical Methods in Fluids 2003, Vol.43(10–11): 1149–1166.

[7] M.K. Karakasis, K.C. Giannakoglou, *On the use of metamodel-assisted, multi-objective evolutionary algorithms*, Engineering Optimization 2005, Vol.38(8): 941–957.

[8] R. Billinton, R.N. Allan, *Reliability evaluation of engineering systems: concepts and techniques*, Plenum Press, 2nd edition, New York, 1992.

[9] J. Nocedal, S. Wright, *Numerical optimization*, Springer-Verlag New York, 1999.

[10] S. Haykin, *Neural networks - A comprehensive foundation*, Prentice Hall, 2nd Edition, 1999.