A BIASED RANDOM KEY GENETIC ALGORITHM APPROACH FOR UNIT COMMITMENT PROBLEM

Luís A. C. Roque
ISEP-DEMA/GECAD, Instituto Superior de Engenharia do Porto, Porto, Portugal

Dalila B. M. M. Fontes
FEP/LIAAD-INESC Porto L.A., Universidade do Porto, Porto, Portugal

Fernando A. C. C. Fontes
FEUP/ISR-Porto, Universidade do Porto, Porto, Portugal

Keywords: Unit commitment, Genetic algorithm, Optimization, Electrical power generation.

Abstract: A Biased Random Key Genetic Algorithm (BRKGA) is proposed to find solutions for the unit commitment problem. In this problem, one wishes to schedule energy production on a given set of thermal generation units in order to meet energy demands at minimum cost, while satisfying a set of technological and spinning reserve constraints. In the BRKGA, solutions are encoded by using random keys, which are represented as vectors of real numbers in the interval $[0,1]$. The GA proposed is a variant of the random key genetic algorithm, since bias is introduced in the parent selection procedure, as well as in the crossover strategy. Tests have been performed on benchmark large-scale power systems of up 100 units for a 24 hours period. The results obtained have shown the proposed methodology to be an effective and efficient tool for finding solutions to large-scale unit commitment problems. Furthermore, form the comparisons made it can be concluded that the results produced improve upon the best known solutions.

1 INTRODUCTION

The Unit Commitment (UC) problem is well known in the power industry and adequate solutions for it have the potential to save millions of dollars per year in fuel and related costs. Therefore, the UC problem plays a key role in planning and operation of power systems. The UC problem is a complex decision making process since it entails the schedule of the turn-on and turn-off of the thermal generation units, as well as the amount of power to be generated by each online unit for each period of the generation horizon. In addition, there are multiple technological constraints and spinning reserve constraints that must be satisfied. Due to its combinatorial nature, multi-period characteristics, and nonlinearities this problem is computationally demanding and, thus, no exact optimization method is capable of solving the UC problem for real-sized systems. In the past, several traditional heuristic approaches based on exact methods have been used, see e.g. (Lee, 1980; Cohen and Yoshimura, 1983; Merlin and Sandrin, 1983). However, more recently most of the developed methods are metaheuristics, evolutionary algorithms, and hybrids of the them, see e.g. (Arroyo and Conejo, 2002; Valenzuela and Smith, 2002; Jenkins and Purushothama, 2003; Dudek, 2004; Simopoulos et al., 2006; Chen and Wang, 2007; Maturana and Riff, 2007; Senjyu et al., 2008; Viana et al., 2008; Abookazemi et al., 2009). These latter types have, in general lead to better results than the ones obtained with the traditional heuristics. Comprehensive and detailed surveys can be found in (Padhy, 2001; Salam, 2007; Raglend and Padhy, 2008).

In this paper we focus on applying Genetic Algorithms (GAs) to find good quality solutions for the UC problem. The majority of the reported GA implementations to address the UC problem are based on the binary encoding. However, studies have shown that other encoding schemes such as real valued random

A. C. Roque L., B. M. M. Fontes D. and A. C. C. Fontes F..
A BIASED RANDOM KEY GENETIC ALGORITHM APPROACH FOR UNIT COMMITMENT PROBLEM.
DOI: 10.5220/0003076703320339
In *Proceedings of the International Conference on Evolutionary Computation* (ICEC-2010), pages 332-339
ISBN: 978-989-8425-31-7

keys (Bean, 1994) can be efficient when accompanied with suitable GA operators, specially for problems where the relative order of tasks is important. In the proposed algorithm a solution is encoded as a vector of n real random keys in the interval $[0,1]$, where n is the number of generation units. The Biased Random Key Genetic Algorithm (BRKGA) proposed in this paper is based on the framework provided by Resende and Gonçalves in (Gonçalves and Resende, 2009). BRKGAs are a variation of the Random key Genetic Algorithms (RKGAs), first introduced by Bean (Bean, 1994). The bias is introduced at two different stages of the GA. On the one hand, when parents are selected we get a higher change of good solutions being chosen, since one of the parents is always taken from a subset including the best solutions. On the other hand, the crossover strategy is more likely to choose alleles from the best parent to be inherited by offspring. In (Gonçalves and Resende, 2009) is presented a tutorial on the implementation and use of biased random key genetic algorithms for solving combinatorial optimization problems and many successful applications are reported in the references therein.

This paper is organized as follows. In Section 2, the UC problem is described and formulated, while in Section 3 the genetic algorithm proposed is explained. Section 4 presents the set of benchmark systems used in the computational experiments and reports on the results obtained. Finally, in Section 5 some conclusions are drawn.

2 UC PROBLEM FORMULATION

In the UC problem one needs to determine the turn-on and turn-off times of the power generation units, as well as the generation output subject to operational constraints, while satisfying load demands at minimum cost. Therefore, we have two types of decision variables. The binary variables, which indicate the status of each unit in each time period and the real variables, which provide the information on the amount of energy produced by each unit in each time period. The choices made must satisfy two sets of constraints: the demand constraints, regarding the load requirements and the spinning reserve requirements and the technical constraints, regarding generation units constraints. The costs are made up two components: the fuel costs, i.e. production costs, and the start-up costs.

Let us now introduce the parameters and decision variables notation.

$Yth_{t,j}$**:** (Thermal) Generation of unit j at time period t, in $[MW]$;

$u_{t,j}$**:** Status of unit j at time period t (1 if the unit is on; 0 otherwise);

T**:** Number of time periods (hours) of the scheduling time horizon;

N**:** Number of generation units;

t**:** Time period index;

j**:** Generation unit index;

D_t**:** Load demand at time period t, in $[MW]$;

Dr_t**:** System spinning reserve requirements at time period t, in $[MW]$;

$Yth_{Min/Max}$**:** Minimum/maximum generation limits, in $[MW]$;

$\Delta_j^{dn/up}$**:** maximum allowed output level decrease/increase in consecutive periods for unit j, in $[MW]$.

$T_{min,j}^{on/off}$**:** Minimum uptime/downtime of unit j, in $[hours]$;

$T_j^{on/off}(t)$**:** Time periods for which unit j has been continuously on-line/off-line until time period t, in $[hours]$;

$T_{c,j}$**:** Number of time periods needed to cool down unit j, in $[hours]$;

$SU_{H/C,j}$**:** Hot/Cold start-up cost of unit j, in [\$];

2.1 Objective Function

As already said, there are two cost components: generation costs and start-up costs. The generation costs, i.e. the fuel costs, are conventionally given by a quadratic cost function as in equation (1), while the start-up costs, that depend on the number of time periods during which the unit has been off, are given as in equation (2).

$$F_j(Y_{th,j}) = a_j \cdot (Yth_{t,j})^2 + b_j \cdot Yth_{t,j} + c_j, \qquad (1)$$

where a_j, b_j, c_j are the cost coefficients of unit j.

$$SU_{t,j} = \begin{cases} SU_{H,j} & \texttt{if } T_{min,j}^{off} \le T_j^{off}(t) \le T_{c,j} \\ SU_{C,j} & \texttt{if } T_j^{off}(t) > T_{c,j} \end{cases}. \qquad (2)$$

where $SU_{H,j}$ and $SUS_{C,j}$ are the hot and cold start-up costs of unit j, respectively.

Therefore, the cost incurred with an optimal scheduling is given by the minimization of the total costs for the whole planning period, as in equation (3).

$$\text{Minimize} \quad \sum_{t=1}^{T}\left(\sum_{j=1}^{N}\{F_j(Yth_{t,j})\cdot u_{t,j} + SU_{t,j}\cdot(1-u_{t-1,j})\cdot u_{t,j}\}\right). \tag{3}$$

2.2 Constraints

The constraints can be divided into two sets: the demand constraints and the technical constraints. Regarding the first set of constraints it can be further divided into load requirements and spinning reserve requirements, which can be written as follows:

1. Power Balance Constraints. The total power generated must meet the load demand, for each time period.

$$\sum_{j=1}^{N} Yth_{t,j}\cdot u_{t,j} \geq D_t, t\in\{1,2,...,T\}. \tag{4}$$

2. Spinning Reserve Constraints. The spinning reserve is the total amount of real power generation available from on-line units net of their current production level.

$$\sum_{j=1}^{N} Yth_{max,j}\cdot u_{t,j} \geq Dr_t + D_t, t\in\{1,2,...,T\}. \tag{5}$$

The second set of constrains includes unit output range, minimum number of time periods that the unit must be in each status (on-line and off-line), and the maximum output variation allowed for each unit.

3. Unit Output Range Constraints. Each unit has a maximum and minimum production capacity.

$$Yth_{min,j}\cdot u_{t,j} \leq Yth_{t,j} \leq Yth_{max,j}\cdot u_{t,j}, \tag{6}$$

for $t\in\{1,2,...,T\}$ and $j\in\{1,2,...,N\}$.

4. Ramp Rate Constraints. Due to the thermal stress limitations and mechanical characteristics the output variation levels of each online unit in two consecutive periods are restricted by ramp rate limits.

$$-\Delta_j^{dn} \leq Yth_{t,j} - Yth_{t-1,j} \leq \Delta_j^{up}, \tag{7}$$

for $t\in\{1,2,...,T\}$ and $j\in\{1,2,...,N\}$.

5. Minimum Uptime/Downtime Constraints. The unit cannot be turned on or off instantaneously once it is committed or uncommitted. The minimum uptime/downtime constraints indicate that there will be a minimum time before it is shut-down or started-up, respectively.

$$T_j^{on}(t) \geq T_{min,j}^{on} \text{ and } T_j^{off}(t) \leq T_{min,j}^{off}, \tag{8}$$

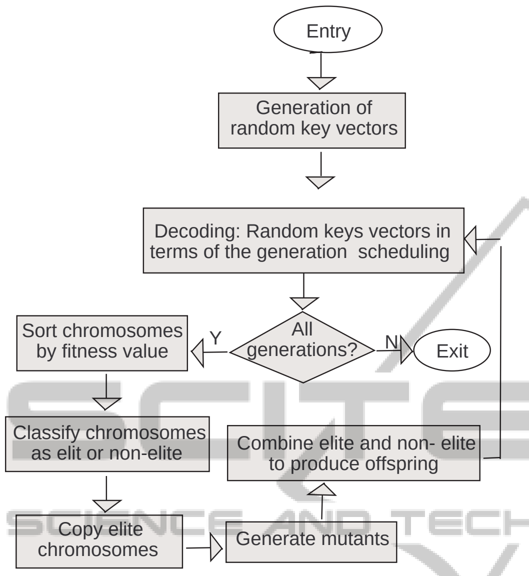
for $t\in\{1,2,...,T\}$ and $j\in\{1,2,...,N\}$.

3 BIASED RANDOM KEY GENETIC ALGORITHM

Genetic Algorithms (GAs) are a optimization technique based on natural genetics and evolution mechanisms such as survival of the fittest law, genetic recombination and selection (Holland, 1975; Goldberg, 1989). GAs provide great modeling flexibility and can easily be implemented to search for solutions of combinatorial optimization problems. Several GAs have been proposed for the unit commitment problem, see e.g. (Kazarlis et al., 1996; Cheng et al., 2000; Swarup and Yamashiro, 2002; Arroyo and Conejo, 2002; Xing and Wu, 2002; Dudek, 2004; Abookazemi et al., 2009), the main differences being the representation scheme, the decoding procedure, and the solution evaluation procedure (i.e. fitness function).

Many GA operators have been used; the most common being copy, crossover, and mutation. Copy consists of simply copying the best solutions from the previous generation into the next one, with the intention of preserving the chromosomes corresponding to best solutions in the population. Crossover produces one or more *offsprings* by combining the genes of solutions chosen to act as their parents. The mutation operator randomly changes one or more genes of a given chromosome in order to introduce some extra variability into the population and thus, prevent premature convergence.

The GA proposed here, i.e. the BRKGA, uses the framework proposed by Resend and Gonçalves in (Gonçalves and Resende, 2009). The algorithm evolves a population of chromosomes that are used to assign priorities to the generation units. These chromosomes are vectors, of size N (number of units), of real numbers from the interval $[0,1]$ (called random keys). A new population is obtained by joining three subsets of solutions as follows: the first subset is obtained by copying the best solutions of the current population; the second subset is obtained by using a (biased) parameterized uniform crossover; the remaining solutions, termed mutants, are randomly generated as was the case for the initial population. The BRKGA framework is illustrated in Figure 1, which has been adapted from (Gonçalves and Resende, 2009).



Entry
Generation of random key vectors
Decoding: Random keys vectors in terms of the generation scheduling
All generations?
Y
N
Exit
Sort chromosomes by fitness value
Classify chromosomes as elit or non-elite
Combine elite and non- elite to produce offspring
Copy elite chromosomes
Generate mutants

Figure 1: The BRKGA framework.

Specific to our problem is the decode and repair procedure, that is how solutions are constructed once a population of chromosomes is given. The decoding procedure is performed in two main steps, as it can be seen in Figure 2. Firstly, a solution satisfying the load demand, for each period is obtained. In this solution the units production is proportional to their priority, which is given by the random key value. Then, these solutions are checked for constraints satisfaction.

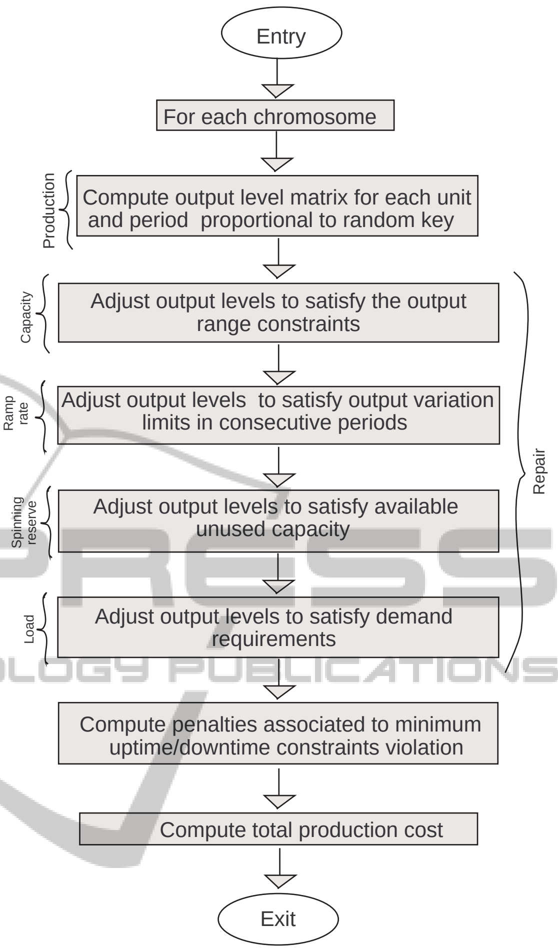
3.1 Decoding Procedure

Given a vector of numbers in the interval $[0,1]$, say $RK = (r_1, r_2, ..., r_N)$ a percent vector $V = (v_1, v_2, ..., v_N)$ is computed. Each element v_j is computed as $v_j = \frac{r_j}{\sum_i^N r_i}, i = 1,2,...,N$.

Then an output generation matrix Yth is obtained, where each element $Yth(t, j)$ gives the production level of unit j for time period t and is computed as in equation (9).

$$Yth(t, j) = D_t \cdot v_j, j = 1,2,...,N. \quad (9)$$

The production level of unit j for each time period t however, may not be admissible and therefore, the solution obtained may be unfeasible. Hence, the decoding procedure also incorporates a repair mechanism. This mechanism forces constraints satisfaction, except for the minimum uptime/downtime constraints, which are handled implicitly by using a penalty function, as already mentioned.



Entry
For each chromosome
Production
Compute output level matrix for each unit and period proportional to random key
Capacity
Adjust output levels to satisfy the output range constraints
Ramp rate
Adjust output levels to satisfy output variation limits in consecutive periods
Spinning reserve
Adjust output levels to satisfy available unused capacity
Load
Adjust output levels to satisfy demand requirements
Repair
Compute penalties associated to minimum uptime/downtime constraints violation
Compute total production cost
Exit

Figure 2: Flow chart of the decoder.

The repair mechanism starts by forcing the output level of each unit to be in its output range as given in equation (10).

$$Yth_{t,j} = \begin{cases} Yth_{max,j} & \text{if } Yth_{t,j} \geq Yth_{max,j} \\ Yth_{t,j} & \text{if } Yth_{min,j} < Yth_{t,j} < Yth_{max,j} \\ Yth_{min,j} & \text{if } \chi \cdot Yth_{min,j} \leq Yth_{t,j} \leq Yth_{min,j} \\ 0 & \text{otherwise,} \end{cases} \quad (10)$$

where $\chi \in [0,1]$ is a scaling factor.

At the same time that the ramp constrains are ensured for a specific time period t, new output limits ($Yth_{t,j}^{max}$ and $Yth_{t,j}^{min}$ upper and lower limits, respectively) must be imposed, for the following period $t+1$, since their value depends on the output level of the current period t. Equations (11) and (12) show how this is done.

$$Yth_{t,j} = \begin{cases} Yth_{t,j}^{max} & \text{if } Yth_{t,j} \geq Yth_{t,j}^{max} \\ Yth_{t,j} & \text{if } Yth_{t,j}^{min} < Yth_{t,j} < Yth_{t,j}^{max} \\ Yth_{t,j}^{min} & \text{if } \mu \cdot Yth_{t,j}^{min} \leq Yth_{t,j} \leq Yth_{t,j}^{min} \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

where $Yth_{1,j}^{max} = Yth_{max,j}$, $Yth_{1,j}^{min} = Yth_{min,j}$ and

$$\begin{aligned} Yth_{t,j}^{max} &= min\left\{Yth_{max,j}, Yth_{t-1,j} + \Delta_j^{up}\right\}, \\ Yth_{t,j}^{min} &= max\left\{Yth_{min,j}, Yth_{t-1,j} - \Delta_j^{dn}\right\}. \end{aligned} \tag{12}$$

Since to ensure that the unit output range constraints and the ramp rate constraints are verified the output level of the units may have been changed, it is no longer guaranteed that load demands are satisfied. Furthermore, for each period, it may happen that the production is either not enough or excessive. If there is excessive production, the on-line units production is decreased to its minimum allowed value, one at the time, until either all are set to the minimum production or the production reaches the load demand value. In doing so, units are considered in descending order of priority, i.e. random key value. It should be notice that by reducing production at time period t the production limits at time period $t+1$ change, and the new values must be respected. Therefore, the minimum allowed production is given by $max\left\{Yth_{t,j}^{min}, Yth_{t+1,j} - \Delta_j^{up}\right\}$. This is repeated no more than N times. If there is lack of production, the on-line units production is increased to its maximum allowed value, one at the time, until either all are set to the maximum production or the production reaches the load demand value. In doing so, units are considered in ascending order of priority, i.e. random key value. It should be notice that by increasing production at time period t the production limits at time period $t+1$ change, and the new values must be respected. Therefore, the maximum allowed production is given by $min\left\{Yth_{t,j}^{max}, Yth_{t+1,j} + \Delta_j^{dn}\right\}$. Again, this is repeated no more than N times.

At the end of the procedures just explained it may happen that the production matches, is larger than or lesser than the demand. Both in the first and second cases, the procedure moves onto the spinning reserve constraints phase, while in the latter case units are turned on-line at least at the required output level (or more if the ramp constraints require so).

Once the spinning reserve phase is reached the production either matches or is larger than the load demand. Therefore, the spinning reserve requirements can be decreased on the amount of the excessive production. This new value is then compared to the unused production capacity. If larger, then units will be turned on-line, in descending order of priority, at minimum output level until their cumulative capacity satisfies the spinning reserve requirements.

Once these four repairing stages have been performed the solutions obtained may not satisfy the uptime or the downtime constraints. If they are not satisfied then a penalty function is applied. This penalty function is explained in Section 3.3i when the fitness function is described.

3.2 GA Operators

In order to obtain a new population

- 20% of the best solutions (elite set) of the current population are copied;
- 20% of the new population is obtained by introducing mutants, that is by randomly generating new sequences of random keys, which are then decoded to obtain mutant solutions. Since they are generated using the same distribution as the original population, no genetic material of the current population is brought in;
- Finally, the remaining 60% of the population is obtained by biased reproduction, which is accomplished by having both a biased selection and a biased crossover.

The selection is biased since, one of the parents is randomly selected from the elite set of solutions (of the current population), while the other is randomly selected from the remainder solutions. This way, elite solutions are given a higher chance of matting, and therefore of passing on their characteristics to future populations. Genes are chosen by using a biased uniform crossover, that is, for each gene a biased coin is tossed to decide on which parent the gene is taken from. This way, the offspring inherits the genes from the elite parent with higher probability (0.7 in our case).

3.3 Fitness Function

The fitness function used for the evaluation of the solutions is composed of two terms. The first term TC represents the total thermal system operating cost, while the second term is the penalty associated with the violation of the uptime and downtime constraints. Recall that the total thermal system operating cost is given by the cost of the fuel needed to produce the energy and the units startup costs, see equation (13).

$$TC_{t,j} = F_j(Yth_{t,j}) \cdot u_{t,j} + SU_{t,j} \cdot (1 - u_{t-1,j}) \cdot u_{t,j}. \tag{13}$$

The penalty function is proportional to the number of violated uptime and downtime constraints, as well as to the magnitude of the violation, and is computed as follows (equation (14)).

$$Pen_{t,j} = \begin{cases} \mu_1 \left| T^{on}_{min,j} - T^{on}_j(t) \right| & \text{if } u_{t,j} = 0 \& u_{t-1,j} = 1, \\ \mu_2 \left| T^{off}_{min,j} - T^{off}_j(t) \right| & \text{if } u_{t,j} = 1 \& u_{t-1,j} = 0, \\ 0 & \text{otherwise}, \end{cases} \tag{14}$$

where μ_1 and μ_2 are penalty multipliers associated with minimum uptime and minimum down time constraints, respectively.

Therefore, the fitness function is given by:

$$fit\,(Yth) = \sum_{t=1}^{T} \sum_{j=1}^{N} \left\{ TC_{t,j} + Pen_{t,j} \right\}. \tag{15}$$

4 NUMERICAL RESULTS

A set of benchmark systems has been used for the evaluation of the proposed algorithm. Each of the problems in the set considers a scheduling period of 24 hours. The set of systems comprises six systems with 10 up to 100 units. A base case with 10 units was initially chosen, and the others have been obtained by considering copies of these units. The base 10 units systems and corresponding 24 hours load demand are given in (Kazarlis et al., 1996). To generate the 20 units problem, the 10 original units have been duplicated and the load demand doubled. An analogous procedure was used to obtain the problems with 40, 60, 80, and 100 units. In all cases, the spinning reserve requirements were set to 10% of the load demand. The BRKGA was implemented with biased crossover probability as main control parameter. The parameter ranges used in our experiments were $0.5 \leq P_c \leq 0.8$ with step size 0.1 which gives 4 possible values for biased crossover probability. Several computational experiments were made in order to choose the other parameters values. The results obtained have shown no major differences. Nevertheless, the results reported here refer to the best obtained ones, for which the number of generations was set to $10N$, the population size was set to $2N$, biased crossover probability was set to 0.7, and the scaling factor $\chi = 0.4$. Due to the stochastic nature of the BRKGA, each problem was solved 20 times.

The BRKGA has been implemented on Matlab and executed on a Pentium IV Core Duo personal computer T 5200, 1.60GHz and 2.0GB RAM. We compare the results obtained with the best results reported in literature. In tables 1, 2, and 3, we compare the best, average, and worst results obtained, for each of the six problems, with the best of each available in literature. As it can be seen, for four out of the six problems solved our best results improve upon the best known results, while for the other two it is within 0.15% and 0.27% of the best known solutions.

For each type of solution presented (best, average, and worst) we compare each single result with the best respective one (given in bold) that we were able to find in the literature. The results used have been taken from a number of works as follows: MRCGA (Sun et al., 2006), LRGA (Cheng et al., 2000), SM(Simopoulos et al., 2006), and GA (Senjyu et al., 2002).

Another important feature of the proposed algorithm is that, as it can be seen in Table 4, the variability of the results is very small. The difference between the worst and best solutions found for each problem is always below 0.65%, while if the best and the average solutions are compared this difference is never larger than 0.25%. The maximum standard deviation over the average is 0.21%. This allows for inferring the robustness of the solution since the gaps between the best and the worst solutions are very small. Furthermore our worst solutions, when worse than the best worst solutions reported are always within 0.6% of the latter, see Table 4. This is very important since the industry is reluctant to use methods with high variability as this may lead to poor solutions being used.

5 CONCLUSIONS

A Biased Random Key Genetic Algorithm, following the ideas presented in (Gonçalves and Resende, 2009), for finding solutions to the unit commitment problem has been presented. In the solution methodology proposed real valued random keys are used to encode solutions, since they have been proved to perform well in problems where the relative order of tasks is important. The proposed algorithm was applied to systems with 10, 20, 40, 60, 80, and 100 units with a scheduling horizon of 24 hours. The numerical results have shown the proposed method to improve upon current state of the art, since only for two problems it was not capable of finding better solutions. Furthermore, the results show a further very important feature, lower variability as was refered in section 4. This is very important since methods to be used in industrial applications are required to be robust, therefore preventing the use of very low quality solutions.

Table 1: Comparison between best results obtained by the BRKGA and the best ones reported in literature.

Size	GA	MRCGA	LRGA	BRKGA	Ratio
10	**563977**	564244	564800	564806	100.15
20	1125516	1125035	**1122622**	1120481	99.81
40	2249715	2246622	**2242178**	2248205	100.27
60	3375065	**3367366**	3371079	3358379	99.73
80	4505614	**4489964**	4501844	4483842	99.86
100	5626514	**5610031**	5613127	5608875	99.98

Table 2: Comparison between average results obtained by the BRKGA and the best averages reported in literature.

Size	MRCGA	SM	BRKGA	Ratio
10	**564467**	566787	565056	100.12
20	**1126199**	1128213	1121655	99.59
40	2249609	**2249589**	2253117	100.16
60	**3371036**	–	3364051	99.79
80	4497346	**4494378**	4485831	99.80
100	5616957	**5616699**	5615193	99.97

Table 3: Comparison between worst results obtained by the BRKGA and the best worst ones reported in literature.

Size	GA	MRCGA	SM	BRKGA	Ratio
10	**565606**	565756	567022	565672	100.01
20	1128790	**1128326**	1128403	1123746	99.59
40	2256824	2252076	**2249589**	2262701	100.58
60	3382886	**3375815**	–	3374920	99.97
80	4527847	4505511	**4494439**	4494916	100.01
100	5646529	5623248	**5616900**	5628072	100.20

Table 4: Analysis of the variability of the results and execution time.

Size	Best	Average	Worst	$\frac{Av-Best}{Best}\%$	$\frac{Worst-Best}{Best}\%$	St. deviation(%)	Av.Time(s)
10	564806	565056	565672	0.044	0.153	0.04	5.6
20	1120481	1121655	1123746	0.105	0.291	0.1	24.6
40	2248205	2253117	2262701	0.219	0.645	0.21	115.5
60	3358379	3364051	3374920	0.169	0.493	0.19	292.6
80	4483842	4485831	4494916	0.044	0.247	0.1	656.2
100	5608875	5615193	5628072	0.113	0.342	0.11	1201.9

ACKNOWLEDGEMENTS

The financial support by FCT, POCI 2010 and FEDER, through project PTDC/EGE-GES/099741/2008 is gratefully acknowledged.

REFERENCES

Abookazemi, K., Mustafa, M., and Ahmad, H. (2009). Structured Genetic Algorithm Technique for Unit Commitment Problem. *International Journal of Recent Trends in Engineering*, 1(3):135–139.

Arroyo, J. M. and Conejo, A. (2002). A parallel repair genetic algorithm to solve the unit commitment problem. *IEEE Transactions on Power Systems*, 17:1216–1224.

Bean, J. (1994). Genetic Algorithms and Random Keys for Sequencing and Optimization. *ORSA Journal on Computing*, 6(2).

Chen, Y. and Wang, W. (2007). Fast solution technique for unit commitment by particle swarm optimisation and genetic algorithm. *International Journal of Energy Technology and Policy*, 5(4):440–456.

Cheng, C., Liu, C., and Liu, G. (2000). Unit commitment by Lagrangian relaxation and genetic algorithms. *IEEE Transactions on Power Systems*, 15:707–714.

Cohen, A. I. and Yoshimura, M. (1983). A Branch-and-Bound Algorithm for Unit Commitment. *IEEE Transactions on Power Apparatus and Systems*, 102:444–451.

Dudek, G. (2004). Unit commitment by genetic algorithm with specialized search operators. *Electric Power Systems Research*, 72(3):299–308.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. New York / USA, Addison-Wesley.

Gonçalves, J. F. and Resende, M. G. C. (2009). Biased random-key genetic algorithms for combinatorial optimization. *Submitted to Journal of Heuristics*.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

Jenkins, L. and Purushothama, G. (2003). Simulated annealing with local search-a hybrid algorithm for unit commitment. *IEEE Transactions on Power Systems*, 18(1):1218–1225.

Kazarlis, S., Bakirtzis, A., and Petridis, V. (1996). A Genetic Algorithm Solution to the Unit Commitment Problem. *IEEE Transactions on Power Systems*, 11:83–92.

Lee, F. (1980). Short-term unit commitment-a new method. *IEEE Transactions on Power Systems*, 3(2):625–633.

Maturana, J. and Riff, M. (2007). Solving the short-term electrical generation scheduling problem by an adaptive evolutionary approach. *European Journal of Operational Research*, 179(3):677–691.

Merlin, A. and Sandrin, P. (1983). A new method for unit commitment at Electricit de France. *IEEE Transactions on Power Apparatus Systems*, 2(3):1218–1225.

Padhy, N. (2001). Unit commitment using hybrid models: a comparative study for dynamic programming, expert system, fuzzy system and genetic algorithms. *International Journal of Electrical Power & Energy Systems*, 23(8):827–836.

Raglend, I. J. and Padhy, N. P. (2008). Comparison of Practical Unit Commitment Solutions. *Electric Power Components and Systems*, 36(8):844–863.

Salam, S. (2007). Unit commitment solution methods. In *Proceedings of World Academy of Science, Engineering and Technology*, volume 26, pages 600–605.

Senjyu, T., Chakraborty, S., Saber, A., Toyama, H., Yona, A., and Funabashi, T. (2008). Thermal unit commitment strategy with solar and wind energy systems using genetic algorithm operated particle swarm optimization. In *IEEE 2nd International Power and Energy Conference, 2008. PECon 2008*, pages 866–871.

Senjyu, T., Yamashiro, H., Uezato, K., and Funabashi, T. (2002). A unit commitment problem by using genetic algorithm based on unit characteristic classification. In *IEEE Power Engineering Society Winter Meeting, 2002*, volume 1.

Simopoulos, D., Kavatza, S., and Vournas, C. (2006). Unit commitment by an enhanced simulated annealing algorithm. *IEEE Transactions on Power Systems*, 21(1):68–76.

Sun, L., Zhang, Y., and Jiang, C. (2006). A matrix real-coded genetic algorithm to the unit commitment problem. *Electric Power Systems Research*, 76(9-10):716–728.

Swarup, K. and Yamashiro, S. (2002). Unit Commitment Solution Methodology Using Genetic Algorithm. *IEEE Transactions on Power Systems*, 17:87–91.

Valenzuela, J. and Smith, A. (2002). A seeded memetic algorithm for large unit commitment problems. *Journal of Heuristics*, 8(2):173–195.

Viana, A., Sousa, J., and Matos, M. (2008). Fast solutions for UC problems by a new metaheuristic approach. *IEEE Electric Power Systems Research*, 78:1385–1389.

Xing, W. and Wu, F. (2002). Genetic algorithm based unit commitment with energy contracts. *International Journal of Electrical Power & Energy Systems*, 24(5):329–336.