Binary Differential Evolution for the Unit Commitment Problem

Ali Keles
Istanbul Technical University, Department of Computer Engineering,
Maslak, Istanbul, 34469, Turkey
ali.keles@gmail.com

ABSTRACT

The Unit Commitment Problem (UCP) is the task of finding an optimal turn on and turn off schedule for a group of power generation units over a given time horizon to minimize operation costs while satisfying the hourly power demand constraints. Various approaches exist in the literature for solving this problem. This paper reports the results of experiments performed on a series of the UCP test data using the binary differential evolution approach combined with a simple local search mechanism. In the future stages of the project, the algorithm will be applied to solve the UCP for the Turkish interconnected power system.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods and Search—*Scheduling*

General Terms

Algorithms, Design, Economics

Keywords

Differential Evolution, Binary Differential Evolution, Hill Climbing, Unit Commitment, Economic Dispatch

1. INTRODUCTION

The operating conditions of a group of power generation units need to be scheduled effectively to obtain the minimum operational cost while satisfying the hourly power demands, subject to some operational constraints. This problem consists of two subproblems [5]: the UCP and the Economic Dispatch Problem (EDP). UCP [10] is the task of finding an optimal turn on and turn off schedule for a group of power generation units for each time window over a given time horizon. EDP [9] is the task of determining the optimal power outputs for each online generator for each time window. Since even minor savings in the operational costs for an hour can lead to significant overall economic savings, the UCP has gained interest.

As it is stated, the arrangement of which unit will generate how much power to meet an hourly demand is the task of EDP. The solution found for the EDP has the lowest cost for this hour but for a time horizon of N time windows, this solution can cause higher costs due to the start-up constraints and costs of power generating units. There are various kinds of start-up constraints, such as: a power generating unit cannot be turned on for an arbitrary number of hours after it has been turned off, or the start-up cost is usually lower if it is turned on again before its cold start-up time has been reached. Operational constraints like these make it important to schedule the power generating units not only according to their fuel costs but also according to their start-up constraints. There are several approaches used in the literature to tackle this problem. A detailed survey can be found in [6]. In this study, a binary differential evolution (BDE) [7] algorithm which uses an angle modulation technique to operate in binary search spaces is used to solve the UCP. Since the focus of this paper is on solving the UCP, the standard lambda-iteration method [9], an iterative process that finds near-optimal solutions and is commonly used in the literature is utilized for solving the EDP. A local search mechanism method is also used to improve existing solutions found by the BDE.

Differential Evolution (DE) [8] has been successfully applied to many problem domains. This makes it an interesting approach to also apply to the UCP. However, the search space of the UCP is binary. This issue makes it impossible to apply the standard DE algorithm since it works on real valued parameters. BDE which is a variation of DE which uses an angle modulation technique, has been specifically developed for use in binary spaces. This paper reports the initial results of an ongoing project. In the initial stages, the BDE algorithm is implemented and tested on a series of test data obtained from the literature and results are compared to a state-of-the-art memetic algorithm approach [10]. The second stage of the project deals with the application of the algorithm to real-world data obtained from the Turkish national interconnected power generation system.

2. UNIT COMMITMENT PROBLEM

As it is stated in the introduction section, UCP is the problem of deciding which units will stay turned on or will be turned off for each time window and how much power will be produced by each generating unit during each time window over a fixed time horizon.

2.1 Problem Formulation

Parameters:

$P_i(t)$: generated power by unit i at time t
$F_i(p)$: cost of producing p MW power by unit i
$PD(t)$: power demand at time t
$PR(t)$: power reserve at time t
$CS_i(t)$: start-up cost of i-th unit at time t
$x_i(t)$: duration that unit i has stayed off since hour t
$v_i(t)$: status of i-th unit at time t (online-offline)

Among these parameters $P_i(t)$ and $v_i(t)$ should be optimized while calculating $F_i(p)$, $CS_i(t)$ and $x_i(t)$. The parameters $PD(t)$ and $PR(t)$ are included in the system data.

According to these system parameters, for N generating units at time t, the minimum total fuel cost is calculated using the following objective function and constraints.

$$\min F_{total}(t) = \sum_{i=1}^{N} F_i(P_i(t))$$

Subject to constraints:

$$\textstyle\sum_{i=1}^{N} P_i(t) = PD(t)$$

$$P_i^{min} \leq P_i(t) \leq P_i^{max}$$

The formulation for calculating the start-up cost is:

$$CS_i(t) = \begin{cases} CS_{hot} & \text{if } x_i(t) \leq t_{coldstart} \\ CS_{cold} & \text{otherwise} \end{cases} \quad (1)$$

According to these fuel cost and start-up cost functions and constraints, the formulation for the UCP for N units and T hours is as given below:

$$\min F_{total} = \sum_{t=1}^{T} \sum_{i=1}^{N} [F_i(P_i(t)).v_i(t) + CS_i(t)]$$

Subject to constraints:

$$\textstyle\sum_{i=1}^{N} P_i(t).v_i(t) = PD(t)$$

$$v_i(t).P_i^{min} \leq P_i(t) \leq v_i(t).P_i^{max}$$

$$\textstyle\sum_{i=1}^{N} P_i^{max}.v_i(t) \geq PD(t) + PR(t)$$

2.2 Lambda Iteration

The fuel cost of generating p MW power for i-th unit is calculated using the following formula.

$$F_i(p) = a_{0i} + a_{1i}.p + a_{2i}.p^2$$

As can be seen, this cost for a generating unit depends on three parameters: a_{0i}, a_{1i} and a_{2i}. Based on these parameters, each power generator unit can have different costs while producing the same amount of power due to the values of these characteristic parameters. The lambda-iteration technique uses this formulation to find the lowest cost for dispatching the amount of power to be generated by the online generating units. This corresponds to the EDP. To solve the EDP by lambda-iteration, an optimal lambda value which also satisfies the constraints is sought [9]. Here are the steps of the lambda-iteration technique:

```
1. Select initial $\lambda$ and $\Delta$
2. Repeat
   (a) Calculate P for each unit from $dF_i/dP_i = \lambda$
   (b) Calculate $P_{total} = \sum_{i=1}^{N} P_i$
   (c) $diff = PD - P_{total}$
   (d) if($diff < 0$)
       • $\lambda = \lambda - \Delta$
   (e) else
       • $\lambda = \lambda + \Delta$
   (f) $\Delta = \Delta/2$
3. until ( abs(diff) $\leq$ tolerance)
```

The initial value of λ is calculated according to λ_{max} and λ_{min} which uses P_{max} and P_{min} as the value of P.

$$\lambda = \frac{\lambda_{max} + \lambda_{min}}{2}$$

Also the initial value of Δ is calculated with λ_{max} and λ_{min}.

$$\Delta = \frac{\lambda_{max} - \lambda_{min}}{2}$$

3. DIFFERENTIAL EVOLUTION ALGORITHM

3.1 Differential Evolution

The Differential Evolution (DE) [8] algorithm was introduced by Storn and Price in 1995. Basically, DE is a population based algorithm which operates in continuous search spaces. DE is based on four main steps: *initialization*, *mutation*, *recombination* and *selection*. Each individual in the population passes through these operations. While the initialization step is only done in the first iteration, the other three steps take place in each iteration of DE.

$$X_{j,i,g} = \begin{cases} j & \text{index of parameter} \\ i & \text{index of individual} \\ g & \text{index of generation} \end{cases} \quad (2)$$

The initialization step sets the initial values of the parameters in the population according to their boundaries.

$$X_{j,i,0} = rand(0,1) * (B_{j,max} - B_{j,min}) + B_{j,min}$$

As stated above, each individual, called the *target vector*, goes through the mutation and recombination steps. There are several mutation operators. One of these operators chooses three different vectors from the population and creates the mutant vector, which will be called the *donor vector*, using the following equation:

$$Vi,g = X_{r0,g} + F(X_{r1,g} - X_{r2,g})$$

F takes values in the range (0,1+) and It is recommended to set F less than 1 [8]. Also, the constraint over deciding the r_0, r_1, r_2 vectors can be changed [8]. For example, in some functions when $r_0 = r_1$ is allowed, better optimized results can be obtained.

The aim of the recombination operation is to create a different vector according to the donor and the target vectors. The parameters of this vector are taken from the target vector when a uniformly distributed random number is greater than a predefined *Cr* value; otherwise, it is taken from the donor vector [8]. *Cr* takes values in the range [0,1]. The vector that is created with the recombination step will be called the *trial vector*.

$$U_{j,i,g} = \begin{cases} V_{j,i,g} & \text{if}(rand_j(0,1) \leq Cr \text{ or } j = j_{rand}) \\ X_{j,i,g} & \text{otherwise} \end{cases} \quad (3)$$

Selection is the step to choose the vector between the target vector and the trial vector with the aim of creating an individual for the next generation.

$$X_{i,g+1} = \begin{cases} U_{i,g} & \text{if } f(U_{i,g}) \leq f(X_{i,g}) \\ X_{i,g} & \text{otherwise} \end{cases} \quad (4)$$

These steps continue until an acceptable solution is found or a predefined number of maximum iterations has been reached.

3.2 Binary Differential Evolution

Binary Differential Evolution (BDE) [7] is the modified version of DE which operates in binary search spaces. The mutation operator that DE uses makes it impossible to use DE for optimization problems with a binary search space. To solve this issue, in BDE, the search space of the problem is mapped onto a more simple representation form and after the solution is obtained, the abstracted space is transformed back into the original space [7]. In brief, after the binary search space has been represented with real valued parameters, the standard DE algorithm optimizes the parameters and these are transformed back into the binary space to obtain the binary solution. The Angle Modulation technique, which generates a bit string using a trigonometric function, can be used to transform continuous spaces into a binary space [7].

Generator function:

$$g(X) = sin(2\pi(X - a).b.cos(A)) + d$$

where

$$A = 2\pi.c.(X - a)$$

The parameters a, b, c and d, which take on values in the range [-1,1], represent the problem in the continuous domain and this transformation allows us to obtain the actual solution in the binary domain. The whole bit string is obtained after sampling the $g(X)$ function by the length of the bit string. X represents the index of sampling. If $g(x)$ takes a positive value at the sampling point, 1 is recorded as a bit value; otherwise, 0 is recorded. By using BDE, the dimension of the problem is also reduced from N to 4 [7] which decreases the search space.

3.3 Local Search through Hill-Climbing

Combining local search techniques with evolutionary algorithms is a method used to generate better performing hybrid approaches [2]. Genetic algorithms enriched with local search methods are commonly known as memetic algorithms [1] and are the state-of-the-art for many application domains [4]. A hill-climber, also known as an iterative improvement algorithm [3], is one of the simplest local search techniques. In the greedy form, all the neighbors of the current solution are searched to find a better solution which is then accepted as the next step. The procedure is repeated until no further improvement is possible. In the heuristic form, the neighbors of the current solution are searched in a random order and the first improving solution is accepted. This process also continues until an improvement is no longer possible. In our implementation, a greedy hill-climber is applied in each iteration to the individual which has the best fitness value. In hybrid evolutionary algorithms which use local search techniques to improve found solutions, two approaches are possible [2]: In the Lamarckian method, the genotype of the current individual is replaced by the genotype of the improved solution. In the Baldwinian method, the genotype of the current solution is kept but the fitness of the improved solution is used for the further steps. In our approach, a Baldwinian approach is implemented.

4. EXPERIMENTS

A test problem [11] including four power generating units and a time horizon of eight hours is used to test the implemented algorithm. The data for this test system is given in Table 1 and Table 2. For every individual in the population, the return value of the fitness function is the summation of the fuel cost, the cost of start-up and a penalty value. Cost for power generation is calculated using lambda-iteration based on the status of each power generator unit. For each hour, depending on whether the start-up is a cold start or a hot start, the appropriate cost is added to total start-up cost. In addition to these, if the total generated power does not meet the hourly power demands plus some reserve power amounts, a penalty term is added to the fitness value. Details on the fitness evaluation and the penalty calculation method can be found in [10]. The fitness value of an individual V is calculated by the following formula:

$$fitness = fuelcost(V) + startupcost(V) + penalty(V)$$

Cr is selected as 0.2 and *F* is selected as 0.5 for the DE algorithm. Also the number of individuals in the population is set to 50. The BDE algorithm runs for 600 iterations and the result with the smallest fitness value is chosen as the final result. 20 runs of the algorithm are performed to obtain the average final result for the total cost. The on/off status of power generators vary in different solutions. Due to this, taking the average of results will not make sense. So the EDP's result which is given with table 2 is randomly chosen among the 20 outcomes. After we obtained the test results, they are compared with the results that are generated by a state-of-the-art memetic algorithm [10]. In Valenzuela and Smiths implementation, there is an additional constraint which does not allow an online generator to become offline for the duration of a specific amount of time. This additional constraint does not prevent us from comparing the results of the two approaches because the majority of generators stay online for many consecutive hours and the amount of time specified by the constraint is small. The lambda-iteration function for solving the EDP is allowed to run for a maxmum of 5000 iterations if it does not find a feasible solution according to the tolerance value until then. The result of the ECD problem for each hour is presented

Table 1: Test System (Wood and Woolenberg, 1996).

	Unit1	Unit2	Unit3	Unit4
Pmax (MW)	300	250	80	60
Pmin (MW)	75	60	25	20
a_0	684.74	585.62	213.00	252.00
a_1	16.83	16.95	20.74	23.60
a_2	0.0021	0.0042	0.0018	0.0034
$S_h(hotstart)(\$)$	500	170	150	0.00
$S_c(coldstart)(\$)$	1100	400	350	0.02
$t_{coldstart}(h)$	5	5	4	0
Initial state(h)	8	8	-5	-6

Table 2: Demand and Reserve (Wood and Woolenberg, 1996).

Hour	1	2	3	4
Demand (MW)	450	530	600	540
Reserve (MW)	45	53	60	54
Hour	5	6	7	8
Demand (MW)	400	280	290	500
Reserve (MW)	40	28	29	50

in Table 3. If the corresponding generated power is equal to zero, it means that the unit is offline. The UC results are presented in Table 4. The best result obtained using our approach is approximately 75,077 while the best result of the memetic algorithm is given as 74,675 in [10].

A second test is performed using a larger set of system data taken from [10], which has 10 units and a time horizon of 24 hours. The best result of the memetic algorithm is 566453 [10] as compared to our result which is approximately 629827 for this test set.

5. CONCLUSION AND FUTURE WORK

This work is a preliminary study. As can be seen from the above results, they are not as good as those obtained by the memetic algorithm [10], however they are very close and this is promising especially since the algorithm used in this study is still in its initial stages of development. Different mutation and crossover techniques, population sizes and parameter settings will be experimented with for the BDE. A simple hill-climber is used as the local search mechanism, but better mechanisms could be used for the solution improvement phase. The main idea of this project emerged as a result of the actual need for optimizing the operational costs for the Turkish interconnected power system network.

Table 3: Results of EDP for Each Hour

	$P1_{out}$	$P2_{out}$	$P3_{out}$	$P4_{out}$
Hour 1	296.19	133.81	0.0	20.0
Hour 2	300.0	205.0	25.0	0.0
Hour 3	300.0	250.0	30.0	20.0
Hour 4	300.0	215.0	25.0	0.0
Hour 5	259.5	115.48	25.0	0.0
Hour 6	235.0	0.0	25.0	20.0
Hour 7	265.0	0.0	25.0	0.0
Hour 8	300.0	180	0.0	20.0

Table 4: Results of UCP over 20 Runs

Best Total Cost (\$)	75077.02
Worst Total Cost (\$)	76384.77
Average Total Cost (\$)	75618.26

To answer this need, the resulting algorithm will be run on the real system data.

6. ACKNOWLEDGMENTS

This work is being developed as a senior year graduation project under the supervision of Assist. Prof. Dr. A. Sima Uyar and Assoc. Prof. Dr. Belgin Turkay by the 8th term student Ali Keles in the Computer Engineering Department of Istanbul Technical University, Turkey.

7. REFERENCES

[1] E. K. Burke and G. Kendall. *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques.* Springer, 2005.

[2] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing.* Springer, 2003.

[3] H. H. Hoos and T. Stuetzle. *Stochastic Local Search: Foundations and Applications.* Elsevier, 2005.

[4] N. Krasnogor and J. Smith. A tutorial for competent memetic algorithms: Model, taxonomy and design issues. *IEEE Transactions on Evolutionary Computation*, 9:5:474–488, 2005.

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

[6] N. P. Padhy. Unit commitment - a bibliographical survey. *IEEE Transactions on Power Systems*, 19:2:1196–2005, 2004.

[7] G. Pampara, A. Engelbrecht, and N. Franken. Binary differential evolution. pages 1873–1879. IEEE Congress on Evolutionary Computation (CEC 2006), July 2006.

[8] K. V. Price, R. M.Storn, and J. A. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization.* Springer, 2005.

[9] Saramourtsis, Damousis, Bakirtzis, and Dokopoulos. Genetic algorithm solution to the economic dispatch problem - application to the electrical power grid of crete island. *http://citeseer.ist.psu.edu/268758.html*, 1994.

[10] J. Valenzuela and A. E. Smith. A seeded memetic algorithm for large unit commitment problems. *Journal of Heuristics*, 8:173–195, 2002.

[11] A. Wood and B. Wollenberg. *Power Generation, Operation, and Control.* 1996.