Large Scale Energy Management Problem

Manisha badgurjar
Mechanical Department
Uce,RTU
Kota ,Rajasthan, India
manishabadgurjar@gmail.com

Anand Kumar Chaturvedi
Mechanical Department
Uce,RTU
Kota ,Rajasthan, India
chaturvedi101@gmail.com

Abstract **- Electric energy is an essential resource in today's life. So we are try to solving Challenge *ROADEF/EURO 2010*: A large-scale energy management problem with varied constraints. No matter if we make our first cup of coffee or tea in the morning or run a multi-million euro business, all the time we rely on a secure and inexpensive supply of electricity. Our goal is to full the respective demand of energy over a time horizon of several years, with respect to the total operating cost of all machinery. Determining optimal maintenance schedules and production plans is not easy because of the number of alternatives to assess. As the exact electricity demand of each forthcoming day is unknown and depends on a large variety of factors, this leads to the need of multiple uncertainty scenarios. Additionally, the increasing proportion of renewable energies in today's energy mix makes things more complicated for an utility company, because it has to feed the energy of third-party solar or wind power plants into its electricity networks and regulate its own power plants accordingly. The examined problem comprises three fields of optimization: maintenance scheduling, production planning and determining refueling amounts. It is a tactical model, neither considering short-term operational restrictions (like intraday load following) nor containing strategic decisions (like adding new power plants). However, the proposed model allows a generic formulation of other concerns like electricity network stability, safety considerations, availability of staff and tools, as well as legal restrictions. All of these limitations can be expressed as mathematical constraints. Here we are using genetic algorithms to solve its problem.**

Genetic algorithms are based on the underlying genetic process in biological organisms and on the natural evolution principles of populations. These algorithms process a population of chromosomes, which represent search space solutions, with three operations: selection, crossover and mutation. Under its initial formulation, the search space solutions are coded using the real coding. In this paper we review the features of real coded genetic algorithms. Different models of genetic operators and some mechanisms available for studying the behavior of this type of genetic algorithms are revised and compared.

Keywords: Genetic algorithms, Real coding, Continuous search spaces.

I. INTRODUCTION

In this work we are solving challenge, which was proposed at the ROADEF/EURO Challenge 2010, a competition announced by the French Operational Research and Decision Support Society (ROADEF) and the European Operational Research Society(EURO). Her we take the perspective of a large utility company, tackling their problems in modeling and planning production assets, i.e., a multitude of power plants. The goal is to fulfill the respective demand of energy over a time horizon of several years, with respect to the total operating cost of all machinery. Determining optimal maintenance schedules and production plans is not easy because of the number of alternatives to assess. The scheduling of outages has to comply with various constraints, regarding safety, maintenance, logistics and plant operation while it must lead to production programs with minimum costs[8].

II. PROBLEM STATEMENT

Her we introduce a model for medium-term electricity production planning utilizing a large set of power plants. As it is a tactical model. The model extends over a period of time. This period is split into uniform time steps of configurable length. The two first class entities of our model are a set of various power plants and a set of uncertainty demand scenarios. For each scenario we are looking for a production assignment, such that the sum of energy produced by all available power plants equals the demand during each time step. The need for multiple scenarios arises from the numerous uncertainties that have to be taken into account. In our model there are two very different types of facilities. Power plants of the first type can operate continuously and their fuel supply is outside the scope of our problem. During each time step they can produce an amount of energy in an interval depending on the scenario and time step. We call them Type-1 power plants. Production at these power plants induces cost that is proportional to the power output of a plant and also depend on the scenario and time step. Power plants of this type might be coal- or gas-fired or even virtual power plants for exporting and importing energy, whose available power levels and unit cost we cannot influence. The other type of power plants, called Type-2 power plants, has to be shut down for refueling and maintenance regularly. As the plants have only limited capacities to store fuel and running out of it would stop the production of energy, these power plants have to be refueled regularly. Refueling can only take place when a power plant is offline for several weeks. Hence, the operation of a Type-2 power plant is organized in cycles - successions of an offline period and the following production campaign. In the model, the fuel unit cost depends on the cycle and power plant. For each Type-2 power plant a number of (fuel-related) production constraints apply. To conclude, the proposed subject consists of modeling the production assets and finding an optimal outage schedule. It includes two dependent subproblems [7]:

1. Determine a schedule of plant outages. This schedule must satisfy the given constraints in order to comply with limitations on resources, which are necessary to perform refueling and maintenance operations.
2. Determine an optimal production plan to satisfy demand.

The objective is to minimize the expected cost of production.

Sets
- I: Set of type 2 plants (nuclear). Indexed by i.
- J: Set of type 1 plants (other thermal). Indexed by j.
- T : Set of time steps. Indexed by t.
- W: Set of weeks. Indexed by w.
- S: Set of scenarios. Indexed by s.
- Ki: Set of cycles for each plant i ∈ I.

Global Parameters

- DEMs,t : Demand to satisfy for scenario s and time step t
- D: Length of a time step (all time steps have equal length)

Parameters of each Type-1 Power Plant j

- PMINs,tj : Minimum production level during time step t of scenario s
- PMAXs,tj : Maximum production level during time step of scenario s
- Cs,tj : Cost of production per unit during time step t of scenario s

Parameters of each Type-2 Power Plant i

- PMAXti : Maximum production level during time step t in all scenarios
- XIi : Initial fuel level (i.e., in time step 0)
- Ci,T : Discount per unit for residual fuel at the end of the time horizon

The following parameters are provided for each cycle k of a Type-2 power plant:

- DAi,k : Duration of the outage in weeks
- Ci,k : Cost of refueling per unit during this cycle's outage
- RMINi,k : Minimum refueling amount
- RMAXi,k: Maximum refueling amount
- MMAXi,k : Maximum modulation over production campaign
- Qi,k : Refueling coefficient
- BOi,k : Fuel level threshold activating the imposed power profile for this campaign
- PBi,k : Decreasing power profile
- ∈ : Tolerance for the imposed power profile
- AMAXi,k : Upper bound of fuel level before refueling
- SMAXi,k: Upper bound of fuel level after refueling

Constants

CT1: coupling load and production

$$\sum_{j\in J} pj,s,t + \sum_{i\in I} pi,s,t = DEM\ s,t \qquad s \in S;\ t \in T$$

CT2: Bound of production type 1 plant

$$PMINs,tj \le pj,s,t \le PMAXj,s,t \qquad j \in J; s \in S; t \in T$$

CT3: Offline power

$$Pi,s,t=0 \qquad i \in I;\ s \in S; t \in T$$

CT4: Bound of production of type 2 plant

$$0 \le pi,s,t \le PMAXti \qquad i \in I; s \in S; t \in T$$

CT5:Maximum power after power profil imposition

$$(x(i,s,t)<BOi,k \wedge (x(i,s,t)<PBi,k(x(i,s,t).PMAXit.D) \Rightarrow pi,s,t = 0$$

CT6: Bounds of refueling

$$RMINi,k \le ri,k \le RMAXi,k \qquad i \in I; k \in K$$

CT7: Initial fuel level

$$X(i,s,0) = XIi \qquad i \in I;\ s \in S$$

CT8:Fuel level variation during production campaing

$$x(i,s,t+1) = x(i,s,t) - pi,s,t \cdot D \qquad i \in I;\ s \in S;\ t \in T$$

CT9: Fuel level variation during outage

$$x(i,s,t^{-}_{i,k} + 1) = ((Q_{i,k} - 1) \div Q_{i,k})(x(i,s,t^{-}_{i,k}) - BO_{i,k-1}) + r_{i,k} + BO_{i,k} \qquad i \in I;\ s \in S;\ t \in T$$

CT10: Fuel level bounds around refueling

$$0 \le x(i,s,t_{i,k}) \le AMAXi,k$$

$$x(i,s,t_{i,k}) \le SMAXi,k$$

$$i \in I;\ s \in S;\ k \in K$$

CT11: Maximum modulation over a cycle

$$\sum x(i,s,t) \ge BOi,k \ (PMAXti - pi,s,t) \cdot D \le MMAXi,k \qquad i \in I;\ s \in S;\ k \in K$$

CT12: Outage scheduling constraint

$$TA_{i,k} \ne -1 \Rightarrow TO_{i,k} \le ha_{i,k} \le TA_{i,k}$$
$$TA_{i,k} = -1 \Rightarrow ha_{i,k} = -1 \vee TO_{i,k} \le ha_{i,k}$$

CT13: Oder of outage

$$ha_{i,k} \ne -1 \Rightarrow ha_{i,k-1} + DA_{i,k-1} \le ha_{i,k}$$
$$ha_{i,k-1} = -1 \Rightarrow ha_{i,k} = -1 \qquad i \in I; k \in K$$

CT14: Minimum spacing / maximum overlapping between outages

- *A*: set of considered Type -2 power plants

- Se: Duration in weeks of minimum authorized spacing. Negative values are interpreted as maximum authorized overlapping.

$$(ha_{i,k} - ha_{i',k'} - DA_{i',k'} \geq Se) \vee (ha_{i',k'} - ha_{i,k} - DA_{i,k} \geq Se) \quad i,i' \in A; k,k' \in K; i \neq i'$$

CT15: Minimum spacing/Maxium overlapping between outages during a specific period

- ID: First week of specific period
- IF: Last week of specific period

$$(ID - DA_{i,k} + 1 \leq ha_{i,k} \leq IF) \wedge (ID - DA_{i',k'} + 1 \leq ha_{i',k'} \leq IF) \Rightarrow$$
$$(ha_{i,k} - ha_{i',k'} - DA_{i',k'} \geq Se) \vee (ha_{i',k'} - ha_{i,k} - DA_{i,k} \geq Se) \quad i,i' \in A; k,k' \in K; i \neq i'$$

CT16: Minimum spacing between decoupling dates

$$|ha_{i,k} - ha_{i',k'}| \geq Se \quad i,i' \in A; k,k' \in K; i \neq i'$$

CT17: Minimum spacing between coupling dates

$$|ha_{i,k} + DA_{i,k} - ha_{i',k'} - DA_{i',k'}| \geq Se \quad i,i' \in A; k,k' \in K; i \neq i'$$

CT18: Minimum spacing between decoupling and coupling dates

$$|ha_{i,k} + DA_{i,k} - ha_{i',k'}| \geq Se \quad i,i' \in A; k,k' \in K; i \neq i'$$

CT 19: Limited resources

- $L_{i,k}$: First week of resource usage ($0 \leq L_{i,k} < DA_{i,k}$)
- $TU_{i,k}$: Time of usage of the resource in weeks ($0 \leq TU_{i,k}$; $L_{i,k} + TU_{i,k} \leq DA_{i,k}$)
- Q: Available quantity of the resource

$$\sum_{i \in A} \sum_{k \in K} 1([ha_{i,k} + L_{i,k}, ha_{i,k} + L_{i,k} + TU_{i,k}), w) \leq Q \quad w \in W$$

CT 20: Maximum number of outage during a giving week

H: Considered week

N: maximum number of outage during this week

$$\sum_{i \in A} \sum_{k \in K} 1([hai,k, hai,k + DAi,k), H) \leq N$$

CT21: Maximum offline power capacity during a specific period

- ID: First time step of period
- IF: Last time step of period
- IMAX: Maximum offline power capacity

$$\sum_{i \in A} \sum_{t \in T} PMAX^t_i \leq IMAX \quad t \in T; ID \leq t \leq IF$$

Objective Function

$$\sum_{i \in I} \sum_{k \in K} C_{i,k}.r_{i,k} + (1/|S|) \sum_{s \in S} \sum_{t \in T} \left(\sum_{j \in J} C_j^{s,t}.p_{j,s,t}.D \right) - \sum_{i \in I} C_{i,T}.x(i,s,T)$$

PP2 refuling cost | PP1 production cost | PP2 residual fuel refund

III. REVIEW WORK

1. Author present the solution of the given problem by Local search for mixed-integer nonlinear optimization methodology has been design to solve various industrial problem. The first particularity of our local search is to be *pure* and *direct*. Indeed, *no decomposition* is done; the problem is tackled frontally. In particular, the specificity of our local search is to be *highly randomized*, in order to avoid bias while exploring the search space. Such a diversification of the search is obtained by exploring in a first-improvement fashion a large variety of randomized neighborhoods. The union of these (small) randomized neighborhoods induces in effect a very large neighborhood, allowing to onverge in practice combinatorial and continuous parts of the problem are treated together: combinatorial and continuous decisions can be simultaneously modified by a move during the search. By avoiding decompositions or reductions, no solution is lost and the probability to find good-quality ones is increased. Then, *no hybridization* is done: no particular metaheuristic is used. Then, the second toward high-quality local optima despite hard constraints. Finally, its third specificity is to be *very aggressive: millions of feasible solutions are visited within the time limit*. Indeed, randomized local search is a nondeterministic, incomplete exploration of the search space. Therefore, exploring a huge focused on: *designing a large variety of randomized moves* allowing an effective exploration of the search space, and *speeding up the evaluation of these moves*. In a mixed-integer optimization context, these two points are declined as follows.

First, the moves are designed in order to treat together the combinatorial and continuous dimensions of the problem. For this, discrete and continuous decisions are simultaneously modified by the moves during the search. Then, a difficulty arises: recovering the feasibility of the continuous part of the solution for evaluating the move. It imposes to be able to solve the continuous subproblem, which is generally very time-consuming. That is why the second point is concentrated on implementing an *incremental randomized combinatorial algorithm* for solving approximately but very efficiently the continuous subproblem [1]. Number of (feasible) solutions during

the allocated time augments the probability to find good-quality solutions. Our local-search heuristics are composed of three layers: general heuristic, moves, evaluation machinery. The evaluation machinery forms the engine of the local search; it computes the impacts of moves on constraints and objectives during the search.

2. Author present a Benders decomposition based framework for solving the problem. Because of the nature of the problem, not all constraints can be modelled satisfactorily as linear constraints and the approach is therefore divided into two stages:

   in the first stage Benders feasibility and optimality cuts are added based on the linear programming Relaxation of the Benders Master problem, while in the second stage feasible integer solutions are enumerated and a procedure is applied to each solution in an attempt to make them satisfy the constraints that are not included in the mixed integer program. A number of experiments are performed on the available benchmark instances. These experiments show that the approach is competitive on the smaller instances, but not for the larger ones. We believe the exact approach gives insight into the problem, and additionally makes it possible to find lower bounds on the problem, which is typically not the case for the competing heuristics[2].

3. Author discuss here about constraint programming to solve optimizing problem. Constraint programming (CP) is a declarative programming paradigm and is a powerful method for solving combinatorial problems. CP has proven to be useful in many other fields of combinatorial optimization such as electrical engineering, molecular biology, and natural language processing. As a declarative paradigm, CP formulations define the properties of a solution to be found rather than specifying the exact sequence of the steps to execute. These properties can be stated as simple logical conditions (e.g., "A =>B"), relational constraints (e.g., "$x \leq y$") or much more complex domain specific constraints. Afterwards, solving a concrete problem is delegated to some specialized CP solver. A solver therefore offers a set of predefined models and constraint types, which can be extended almost arbitrarily to fit any application-specific needs. Furthermore, the model might be modified during the execution of the solver[6].

IV. PRESENT WORK

We try to solve this problem by genetic algorithem with real coading. Recently, genetic algorithms (GAs) are successfully used for solution a range of the optimization Problems. Genetic algorithms are probabilistic population-based search techniques.

1) Basic features of genetic algorithms:

1.1) *Terminology*

Genetic algorithms (GAs) are stochastic techniques whose search methods model a natural evolution. That is why the terminology used in GAs is taken from biology[4].

1.2) *General description of genetic algorithms*

The genetic algorithms start with randomly chosen parent chromosomes from the search space to create a population. They work with chromosome genotype. The population "evolves" towards the better chromosomes by applying genetic operators modeling the genetic processes occurring in the nature—selection, crossover and mutation. Selection compares the chromosomes in the population aim in to choose these, which will take part in the crossover process. The selection occurs with a given probability on the base of fitness functions. The fitness function plays a role of the environment to distinguish between good and bad solutions. The crossover is carried out after selection process is finished. It combines, with predefined probability, the features of two selected parent chromosomes forming similar children. After crossover offspring undergoes to mutation. Generally, the mutation refers to the creation of a new chromosome from one and only one individual with predefined probability [5].

After three operators are carried the offspring is inserted into the population, replacing the parent chromosomes in which they were derived from, producing a new generation. This cycle is performed until the optimization criterion is met. This can explain by fig 1.

1.2.1) selection:

At the first generation, initializes a population of randomly created individuals. Check initial population for all constraint. Afterwards they are used to calculate the values of the objective function and to determine respective fitness functions. At the next steps the evolutionary operators take place to create the offspring. Firstly, a biased selection for reproduction is carried out. The algorithm operates with the fitness function values to provide the most prospective samplings for a crossover.

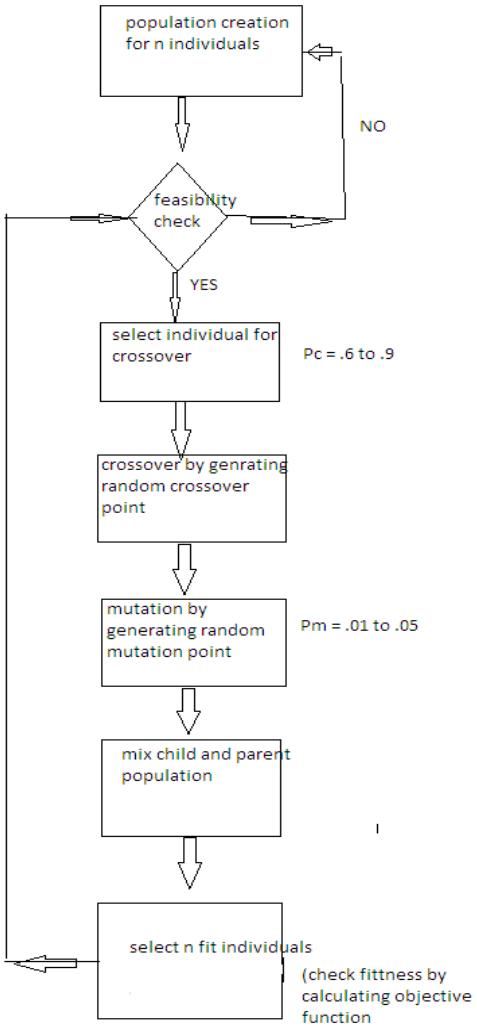

Fig 1. Steps for genetic algo.

1.2.3) Crossover

Crossover probability (Pc) is .6 to .9.crossover can be done 1 point or multipoint crossover. In this problem we are using 1 point crossover. In one Point Crossover select 1 crossover point randomly. Crossover will be done in bits from 1 parent and to the left of the crossover point are combined with bits from the other parent and to the right of the crossover point.

Real crossover is almost the same is in binary:

- Take the list of real numbers from one parent, combine them with a list from the other parent
- {5,6,7,8},{1,2,3,4} combine at crossover point between 2 and 3 to create children {5,6,3,4} and {1,2,7,8}

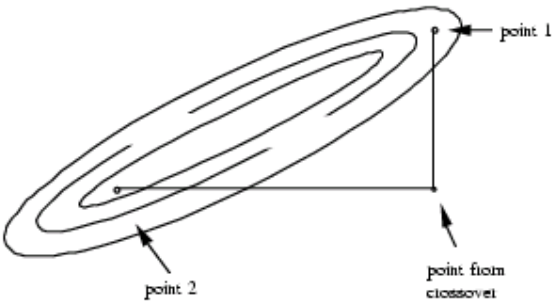

Figure 2

Fig 2. Real crossover[5]

1.2.4) Mutation

Mutation probability is very small than the crossover, it is .01 to .05.Mutation is the occasional introduction of new features in to the solution strings of the population pool to maintain diversity in the population.

Though crossover has the main responsibility to search for the optimal solution, mutation is also used for this purpose. First choose mutation point randomly.in mutation number inter change their places. For eg. 6 and 7 interchange there places in first individual and 3 and 4 interchange there places in second individual. So after mutation individuals are : {5,7,6,8}{1,2,4,3}.

1.2.5) Fitness

A fitness function value quantifies the optimality of a solution. The value is used to rank a particular solution against all the other solutions. A fitness value is assigned to each solution depending on how close it is actually to the optimal solution of the problem. Fitness value is choose by calculating objective function.

CONCLUSION

In this paper we have reviewed several issues relating to one of the most important alternatives to solve problem of optimizing by *the real coding*. We have presented and compared the genetic operators described in the literature for GAs based on this problem. Tools that allow the behavior of these algorithms to be studied have also been explained. The most important feature of the real coding in genetic algorithm (RCGAs) is their capacity to exploit local Continuities. We discuss about the step of solving optimization problem by genetic algorithm. We found that GA is very good technique to solve scheduling and optimizing problem.

REFRENCES

[1] Thierry Benoist, Bertrand Estellon ,Antoine Jeanjean elat.” Local search for mixed-integer nonlinear optimization: methodology and applications”.

[2] Lusby, Richard Martinm, Muller, Laurent Flindt ”Danish Nation Research Database”DTU management 2010.

[3] Dr. Rajib Kumar Bhattacharjya “Introduction to Genetic Algorithum”2012.

[4] Elisaveta G. Shopova , Natasha G. Vaklieva-Bancheva .” A genetic algorithm for engineering problems solution” 2006.
[5] Alden H. Wright “Genetic algorithum for real parameter optimization” 2004.
[6] Prof. Dr. rer. nat. Dorothea Wagner” Solving a Large-Scale Energy Management Problem with Varied Constraints”2010.
[7] Marc PORCHERON, Agnès GORGE, Olivier JUAN, Tomas SIMOVIC and Guillaume DEREU“A large-scale energy management problem with varied constraints”Challenge ROADEF/EURO 2010.
[8] Mirsad Buljubasic, Haris Gavranovic” Orchestrating Constrained Programming and Local Search to Solve a Large Scale Energy Management Problem” mirsad bulj@yahoo.com.