CEM 510

Term Paper

# Particle Swarm Optimization based Approach for Generator Maintenance Scheduling

**Proposed to:**
Dr. Ibrahim Alamin

**By:**
Ayman Hamdan
ID: 270181

# iContents

## i. Introduction

The last three decades have seen the development of many algorithms and heuristics for solving constraint satisfaction problems (CSPs). Determining which algorithms are superior to others remains difficult. Theoretical analysis provides worst-case guarantees which often do not reflect average performance. For instance, a backtracking based algorithm that incorporates features such as variable ordering heuristics will often in practice have substantially better performance than a simpler algorithm without this feature, and yet the two share the same worst-case complexity.

Similarly, one algorithm may be better than another on problems with a certain characteristic, and worse on another category of problem. Ideally, we would be able to identify this characteristic in advance and use it to guide our choice of algorithm.

Algorithms and heuristics have often been compared by observing their performance on benchmark problems, such as the 8-queens puzzle, or on suites of random instances generated from a simple, uniform distribution. The advantage of using a benchmark problem is that if it is an interesting problem (to someone), then information about which algorithm works well on it is also interesting. The drawback is that if algorithm A beats algorithm B on a single benchmark problem, it is hard to extrapolate from this fact. An advantage of using random problems is that there are many of them, and researchers can design carefully controlled experiments and report averages and other statistics. A drawback of random problems is that they may not reflect any real life situations.

In this research, we demonstrate another method for comparing CSP search algorithms, by applying them to random problems that have been generated with a particular structure. The structure, in the present case, was derived from a well-studied problem of the electric power industry: optimally scheduling preventive maintenance of power generating units within an electric power plant. Our approach was to define a formal model which captures most of the interesting characteristics of maintenance scheduling, and then to cast the model as a constraint satisfaction problem.

***What is the PSO?***

We can speech on A Particle Swarm Optimization as :

- Method for solving a multi-objective generator maintenance scheduling problem with many constraints.
- **Particle Swarm Algorithm is:** Population based optimization tool, where the system is initialized with a population of random solutions and the algorithm searches for optima by updating generations.
- In ***PSO***, the potential solutions, called particles, are "flown" through the problem space by following the current optimum particles.
- The velocity and positions of the particles are updated using the following equations:

  ***PSO velocity update***

$$[\,][\,] = w \times v[\,][\,] + 2 \times \text{rand}() \times \{\text{pbest}[\,][\,] - \text{present}[\,][\,]\}$$

$$+ 2 \times \text{rand}() \times \{\text{pbest}[\,][\text{gbest}] - \text{present}[\,][\,]\}$$

  ***PSO position update***

$$\text{Present}[\,][\,] = \text{present}[\,][\,] + v[\,][\,]$$

The first square bracket represents the dimension and second bracket represent the index of the particle

Each particle stores its personal best position (***pbest***) in memory and the current velocity (v) and position (present).

The index of global hest particle (***gbest***) in the population is shared with the rest of the particles.

The use of inertia weight (w) has improved performance on many test problems.

- Effective in obtaining feasible schedules in a reasonable time.
- Actual data from a practical power system was used in this study and results were compared against those from other evolutionary methods on the same set of data and selection mechanism in a hybrid particle swarm algorithm.
- Particle swarm optimization (***PSO***) has been successfully applied in many areas such as:
  - *Function optimization*
  - *Artificial neural network training*
  - *Fuzzy system control*
  - *Genetic Algorithms (**GA**)*
  - *Evolutionary Strategy (**ES**) can he applied.*

Recent research has attempted to combine the attributes of evolutionary computation with ***PSO*** concept to solve many real applications with many constraints and objectives.

First, Lagrange multiplier is introduced to transform the optimization problem into a min-max problem with the saddle-point solution. Next, two ***PSO***s work simultaneously with one ***PSO*** finding the minimum part and the other focus on the maximum part of the problem. At any one time, one ***PSO*** serves as an

environment to the other just like in evolutionary computation. Some researchers combines the features of ***ES*** and ***PSO*** to solve real world applications in Power systems and Opto-electronics. The hybrid model has proven to be effective over classical ***PSO*** in many test problems.

The use of selection in evolutionary technique and adaptive behavior of ***PSO*** have been investigated in some recent works. The researcher investigates the effect of selection in particle swarm with standard test functions. The result suggests that selection may provide some advantage over classical ***PSO*** for certain functions. Adaptive ***PSO*** that can automatically track variations in a dynamic system is introduced. A dynamic system changes its state and hence the optimum value may vary. The proposed method is that if there is no improvement of ***gbest*** particle for a certain number of iterations, then re-randomizes a percentage of the search particles with the hope of finding the new global optimum value. One drawback is how to set the fixed-duration number to give satisfying response time of the system.

All variants of ***PSO*** have similar features like evaluation of fitness, modification of the current population either through ***PSO*** update or mutation and selection to remove poor candidate solutions. The main contrast of the various techniques is the different implementation of evolutionary operators with the classical ***PSO***. However, all the hybrid techniques have similar algorithm structure and perhaps can be broadly classified as follows:

Pseudo code for ***PSO*** + Evolutionary Technique

1. Initialize population with particles
2. Calculate fitness for each particle
3. Introduce some evolutionary techniques and parameters
4. For each particle, update its own ***pbest*** value if got improvement
5. For each particle, update its position by moving towards the ***gbest*** particle or its own ***pbest*** position
6. Repeat step 2 if terminating condition not satisfied

The focus will be on the application of ***PSO*** concept to solve a generator maintenance scheduling that has various constraints and objectives. Its performance will be compared with results obtained by the heuristic, ***GA*** and ***ES*** methods. Further variation of ***PSO*** will be suggested and discussed in the hybrid ***PSO*** model with spawning mechanism to provide some adaptive ability over classical ***PSO***.

## ii. Generator Maintenance Scheduling - Problem Description

Maintenance schedule is a preventive outage schedule for generating units in a power system within a specific time horizon. Maintenance scheduling becomes a complicated problem when the power system contains a number of generating units with different specifications, and when numerous constraints have to be taken into consideration to obtain a practical and feasible solution.

Generator maintenance scheduling is done for time horizons of different durations as following.

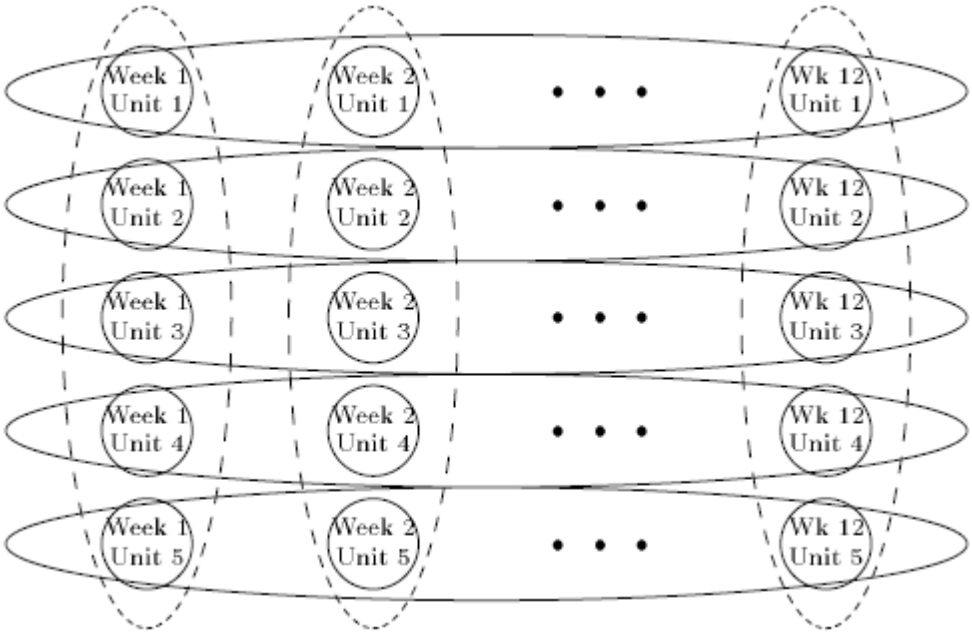
- Short-term maintenance scheduling for one hour to one day ahead is important for day-to-day operations, unit commitment, and operation planning of power generation facilities.
- Medium-term scheduling for one day up to a year ahead is essential for resource management.
- Long-term scheduling of a year to two years ahead is important for future planning.

The problem of scheduling off-line preventive maintenance of power generating units is of substantial interest to the electric power industry. A typical power plant consists of one or two dozen power generating units which can be individually scheduled for preventive maintenance. Both the required duration of each unit's maintenance and a reasonably accurate estimate of the power demand that the plant will be required to meet throughout the planning period are known in advance. The general purpose of determining a maintenance schedule is to determine the duration and sequence of outages of

power generating units over a given time period, while minimizing operating and maintenance costs over the planning period, subject to various constraints. A maintenance schedule is often prepared in advance for a year at a time, and scheduling is done most frequently on a week-by-week basis. The power industry generally Figure bellow: A diagrammatic representation of a maintenance scheduling constraint satisfaction problem. Each circle stands for a variable representing the status of one unit in one week. The dashed vertical ovals indicate constraints between all of the units in one week: meeting the minimum power demand and optimizing the cost per week. The horizontal ovals represent constraints on one unit over the entire period: scheduling an adequate period for maintenance.

considers shorter term scheduling, up to a period of one or two weeks into the future, to be a separate problem called “unit commitment”. As a problem for an electric power plant operator, maintenance scheduling must take into consideration such complexities as local holidays, weather patterns, constraints on suppliers and contractors, national and local laws and regulations, and other factors that are germane only to a particular power plant.

The domain of this problem is based on a real power system of two industrial parks located in Bintan and Batam in Indonesia. In the simulation, a planning horizon of 25 weeks is considered in the generators scheduling problem. In each week, there can only be a maximum of 3 generators in maintenance due to crew and resource constraints. The maintenance of a generator must he done in consecutive weeks and care must be taken to ensure that the solution provides a feasible schedule.




***Overall Objective Function to minimize***

The overall function to be minimized can he represented in a compact form as follows:

= 168 ( + + ) + + Penalty Cost

Where:

**X** = unit in operation for that week
**Y** = unit in maintenance
**T**= length of the maintenance planning schedule (week)
= generator output (MW) of operation unit
$\mathbf{a_x}$, $\mathbf{b_x}$, $\mathbf{c_x}$ = fuel cost coefficient
$\mathbf{V_y}$ = maintenance cost per week **($/week)**
**D** = downtime (weeks)

Penalty cost is added to the evaluation function if the schedule cannot meet the power demand or the crew and resource constraints. Detailed explanations of each constraint and formulation of the objective function can be found in.

## iii. Problem Representation

Proper care has to be taken in the initial random generation of the candidate solutions due to the following constraints:

1. Each generator should be taken off for maintenance in consecutive weeks according to its downtime.
2. In each week, the number of generators that can be maintained is limited to three due to resources and crew constraints.
3. A generator can only be taken off once within a week. Hence, there should be no repeated values within a respective week of the schedule. Except for the number 0, where it represents no generator to be maintained.

After much consideration, a useful representation of the candidate solution is in the form of two-dimensional matrix:

- The rows of the matrix represent the number of weeks in the schedule.
- The columns represent the index of the generators to be taken off for maintenance.

## iv. Evolutionary Approaches for Solving Maintenance Scheduling

Generator maintenance schedules are typically generated by power plant engineers who devise the schedule based on their experience and knowledge of the system.

A hybrid Fuzzy-genetic algorithm system was developed in this research to handle this complex problem for which, at present, there exists no effective planning tool due to the presence of various soft constraints and uncertainties. In this hybrid of fuzzy knowledge based system and ***GA***, a fuzzy knowledge based system evaluates the downtime of generating units, and several constraints. The fuzzy system allows for uncertainties and impreciseness in evaluating the number of operating hours, which give the flexibility in determining the downtimes. The knowledge of power plant engineers is emulated in the form of if-then-rule to check the condition of the engines. The

power of genetic algorithm is used to optimize the maintenance schedule giving the least cost of maintenance and operation.

Another approach developed later replaced the genetic algorithm in this implementation with evolutionary strategy. The results with ***ES*** were better than those with ***GA***.

Since evolutionary strategy emphasize on mutation as a search operator, an algorithm for the mutation process was developed as shown in ***Figure.*****1**. Two heuristics were applied to the standard ***ES*** algorithm in this problem, which results in better schedule with lower cost.

1. A new variable called multiple-mutations was introduced in the generation of the offspring solution matrix to model the global search ability in the beginning and refined search towards the end.
2. Since the selection is based on the best individuals, it is highly possible that there may be repeated solutions with the same fitness. In order to increase diversity and at the same time not to discard other potential solutions. These repeated solutions will be discarded. In other words, all the best individuals are distinct from each other to ensure diversity in the next mutation process.

The results obtained with these two evolutionary approaches are presented in ***Table*** 1.

**Table** I: **Simulation R*esults***

| Method | Gen | Pop | Average cost over 10 runs | Optimized Cost | | Approximate time per run |
|---|---|---|---|---|---|---|
| | | | | Best | Worst | |
| Heuristic | - | - | - | 4,533,639.8487 | - | 1 week |
| GA | 400 | 100 | - | 4,533,435.2287 | - | 40mins |
| ES | 400 | 100 | 4,495,024.3566 | 4,495,625.0839 | 4,498,777.6901 | 45mins |
| PSO | 1000 | 20 | 4,496,197.1266 | 4,495,673.1839 | 4,499,052.9101 | 23mins |
| | 1500 | 30 | 4,496,552.2382 | 4,495,699.3339 | 4,499,614.6501 | 60mins |
| | 2000 | 40 | 4,495,067.1756 | 4,495,511.2439 | 4,498,343.8601 | 107mins |
| SPSOGS | 1000 | 20 | 4,495,290.0729 | 4,495,625.0839 | 4,496,294.0339 | 50mins |
| | 1500 | 30 | 4,495,782.5539 | 4,495,625.0839 | 4,496,119.8339 | 110mins |
| | 2000 | 40 | 4,495,863.1289 | 4,495,472.3739 | 4,496,193.9839 | 194mins |

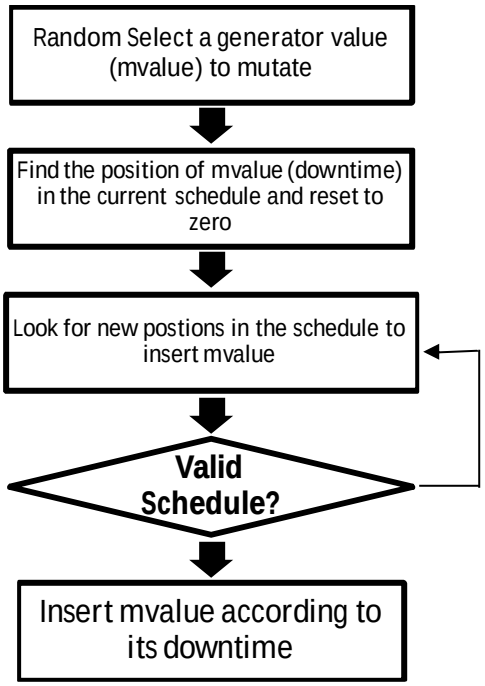



***Figure.***1: Mutation process in evolutionary strategy-bared algorithm for maintenance scheduling

## v. Particle Swarm Optimization-Based Approach for Maintenance Scheduling

### *A. Pure Particle Swarm Optimization-based Maintenance Scheduling*

From Equation 1, one can infer that the particle will update its velocity of flying by either moving towards the ***gbest*** particle or moving towards its own ***pbest*** position that it stored in memory. Although ***PSO*** has no explicit crossover and mutation search operation, but one can infer from Equation 1that ***PSO*** has combined these two concepts in one single operation.

Crossover operation is implicitly implemented with the ***gbest*** particle sharing global information with the rest of the particles. Also, each particle can move towards its ***pbest*** position in memory, hence crossover process within particle itself can take place which may lead to faster convergence.

The implicit mutation process is represented by the randomness introduced from the product of velocity ***v*** and inertia weight ***w***. For test functions optimization, ***w*** is often assumed to decrease linearly from 0.7 to 0.4 to model the concept of global search ability in the beginning and refined search towards the end of the run.

Hence, pure evolutionary crossover was modified to implement ***PSO*** concept, which includes two important parameters: probability of crossover (***pcr***) and probability of mutation (***pmutate***).

When ***pcr*** is set high, information crossover from ***gbest***/***pbest*** solution to current solution is encouraged. When ***pmutate*** is set high, random mutation within current solution is encouraged rather than information crossover.

The update of the current solution is implemented with the modified crossover operation that models ***PSO*** concept. The current solution either moves towards ***gbest*** solution or its own ***pbest*** solution with equal probability.

A program to stimulate the modified crossover process in ***PSO*** has to be carefully written to obtain feasible schedule with the constraints met. The program starts with two input schedules: ***gbest***/***pbest*** and current schedules. For every generator in the schedule, it either undergoes mutation process with probability ***pmutate*** or crossover process with probability (1-***pmutate***). For mutation process, the maintenance weeks (positions) of that generator are randomly inserted in the updated schedule (Figure.2). If crossover process is chosen, then the schedule will be updated with information from ***gbest***/***pbest*** schedule with probability ***pcr*** (***Figure.***3).

In other words, if ***pmutate*** (0.1) is set low, then most of the time, the schedule will be updated with information from ***gbest***/***pbest*** with probability per or current schedule with probability (1-***pcr***).

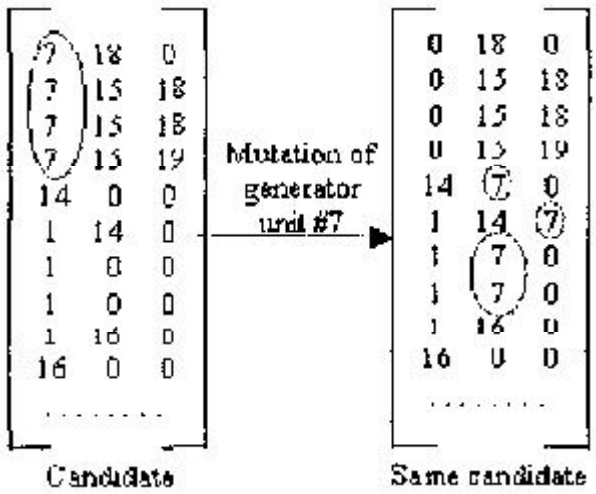



***Figure.*2:** Graphical illustration of the mutation process where generator unit 7 is randomly inserted in other consecutive positions within the same candidate solution.

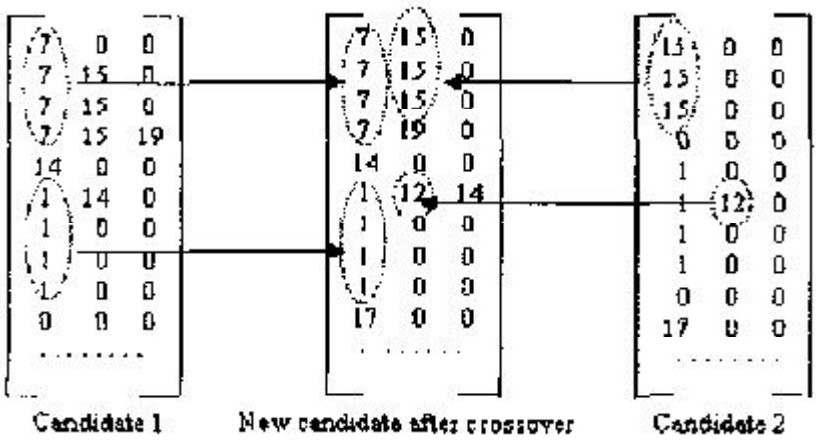



***Figure.*3:** Graphical illustration of the crossover process between two candidates where a new candidate solution will be created and updated with information either mm the first or second candidate

This means that in the event when no random mutation takes place, if ***gbest/pbest*** and current schedules are exactly the same, then the current schedule will still remain the same even after update. ***Figure.***4 shows the implementation of ***PSO*** to solve the generator scheduling problem.

## *B. Hybrid Spawning PSO with Evolutionary Strategy (SPSOES)*

A novel hybrid approach that combines concepts from particle swarm optimization and evolutionary strategy (***ES***) has been developed in this

discussion. From the results obtained for ***PSO*** and ***ES*** (***Table*** 1), it is evident that there is a lack of selection pressure in ***PSO***. In ***ES***, the 'survival of the fitness' concept in the selection of the best individuals ensures that the population moves as a group towards better solutions. In ***PSO***, each agent only compares his current fitness with the best fitness that it has stored in memory. This means that the selection pressure is only contained within itself and not with other agents.

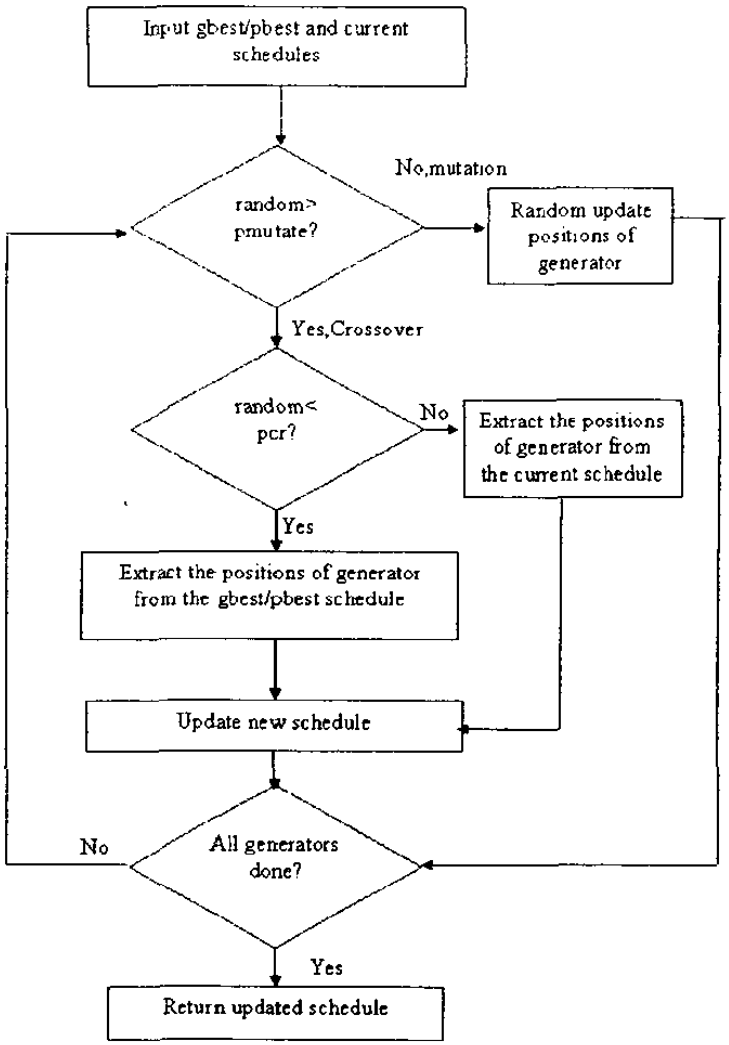



***Figure.4*****:** Flow Chart for ***PSO*** Implementation

In the hybrid ***PSO*** model (***SPSOES***), an explicit mutation parameter similar to ***ES*** was introduced. Each agent generates its off springs through mutation and the original agent is replaced if the off springs are better. This increases the

selection pressure within each agent as it has to compete with its off springs. This approach greatly increases the flexibility in programming as the population size is kept constant, but the number of off springs for each agent can be varied easily with a single mutation parameter.

Fitness evaluation assigns fitness values to the candidate solutions based on the objective function to optimize. The ***ES*** mutation rate (mrate) determines the number of off springs to be generated from each parent. If mrate is set to 3, then each parent will have 3 off springs. The variable mrate need not be a constant and can be set to any desired value during the run and hence provides the freedom to increase or decrease search ability to suit different applications.

Although mutation process dominates in ***ES***, a small probability of crossover (0.1) is introduced. This means that 90% of the off springs are generated through mutation of the parent whereas 10% of them are created through crossover between the parent and a randomly selected parent in the population.

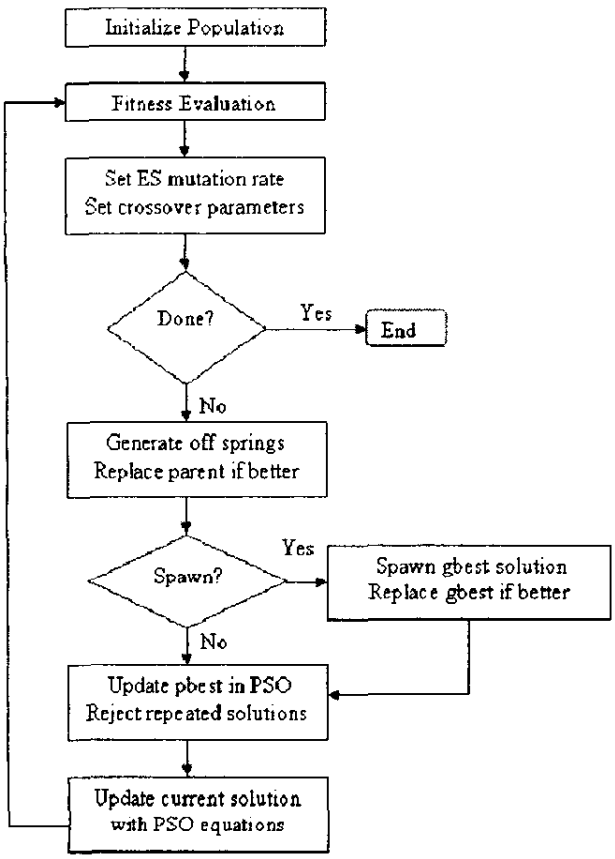



***Figure.*5:** Flow Chart for ***SPSOES*** model

For each parent, off springs are generated and they replace the original parent if they are better based on fitness values.

An additional feature called the spawning mechanism was introduced in the algorithm to introduce some adaptive ability into ***PSO***. The concept was analogous to the natural adaptation of amoeba with the environment. When amoeba receives positive feedback, it reproduces by releasing more spores and hence increases the search ability for food. Where food is scarce, it remains unchanged. Similarly, in potentially optima region, it is encouraged to increase the number of search agents in the region. The amoeba concept was used to spawn potential solutions found during the run.

To keep computation low, the spawning mechanism only spawns the ***gbest*** solution. Whenever a new ***gbest*** solution is found, the number of spawns was set to 10 initially and decrement with each iteration till zero where the spawn mechanism stops. If a better ***gbest*** solution was found during the spawn process, the number of spawns was reset to 10 and the whole process repeats.

The surviving candidates are compared individually to its personal best (***pbest***) fitness value so far stored in the memory and replace if better. However, to ensure diversity in the optima solutions, a check is performed to reject repeated ***pbest*** solutions.

The update of the current solution is exactly the same as pure ***PSO*** explained in section V (part A) shown in ***Figure.***4.

## vi.Experimental Results

The maintenance schedule in the thermal system used for this study is currently prepared based on experience and considerations of the power plant maintenance engineers. The engineer may take days or even weeks in scheduling the maintenance of generating units, yet the schedule they made may not be an optimal one. Moreover, the schedule does not utilize the resources optimally. Furthermore, the engineers currently take approximately

one week to prepare the maintenance schedule for 19 generating units for six month planning horizon.

The following table compares this heuristic schedule prepared by the engineers in a week, with the best schedule obtained using the four approaches, namely, Genetic Algorithm, Evolutionary Strategy, standard ***PSO*** and Hybrid ***SPSOES***. The results with ***GA*** were obtained with a maximum iteration of400 with population size of 100. For a fair comparison, the parameters for ***ES*** are set to be the same as ***GA*** with a mutation rate of 1.

For both ***PSO*** and ***SPSOES***, a maximum iteration of 1000, 1500 and 2000 with population sizes of 20, 30 and 40 respectively are tested. To keep computation low, the ***ES*** mutation rate in ***SPSOES*** model was set to 1 in the generation of off springs.

The schedules obtained by ***PSO*** and ***ES*** prove to be superior over ***GA*** and heuristic methods with much lower cost. The average cost of ***ES***-based solution is slightly lower than standard ***PSO*** probably due to the selection pressure in ***ES***. However, ***PSO*** is able to provide near optimal solution in the shortest time possible with only a population size of 20. Except for the case (Gen=2000, Parents=40) in ***PSO***, ***ES*** performs slightly better than ***PSO*** in terms of optimized cost and worst case so far due to survival of the fitness concept. With increased search ability in ***PSO*** (Gen=2000, Parents-40). ***PSO*** is able to obtain better optimized cost than ***ES*** with the tradeoff for increase in timing.

The hybrid technique proposed here, ***SPSOES*** overcomes the limitations of ***PSO*** and evolutionary strategy by employing a synergistic combination of these two approaches. The average cost of ***SPSOES***-based solution, as well as worst case so far, for the 3 tested settings is much lower than standard ***ES*** or ***PSO***. This means that ***ES*** or ***PSO*** is likely to be stuck in a local optimum whereas ***SPSOES*** has a higher chance of escape from the local attraction. Although the timings of ***SPSOES*** are higher when compared to ***PSO***, but

***SPSOES*** that incorporates spawning and selection mechanism provides the extra stochastic kick to get out of local optimum which results in the most optimized cost so far.

Figure.4 a compares the average convergence rate of the fitness value between standard ***PSO*** and ***SPSOES*** for the case (Gen=1000, Parents=20). The last 500 out of 1000 iterations in the 10 runs are averaged and are plotted to give an illustration of the improvement of the fitness during the run. Hence, it can be observed that hybrid ***PSO*** with spawning of the ***gbest*** solution generally converge faster to better solution than standard ***PSO*** alone. Similar plots for the cases of (Gen=1500, Parents=30) and (Gen=2000, Parents=40) are given in ***Figure.***4b and 4c respectively.

## vii. Discussion

### *A. Heuristic approach*

Currently in practice, the maintenance schedule is obtained by the power plant engineers on a trial and error basis. Due to the many factors involved, scheduling becomes a complicated problem. Heuristic method does not ensure optimal allocation of resources and is not versatile to changing environment factors. Hence, it is the most inefficient and has the highest cost.

### *B. Generic Algorithm*

***GA*** attempts to add flexibility in the scheduling problem by finding a set of optimum solutions through its population based technique. ***GA*** emphasizes on the replacement of individuals over time based on fitness. Those candidates with higher fitness are more likely to be chosen as parents for the next generation. Off springs are generated through crossover or recombination of the parents, replacing the original parents. Occasionally, mutation may take place in the off springs.

The working principle of ***GA*** depends on the underlying assumption that crossover increases the genes pool where off springs benefit from both of the

surviving parents. In theory, crossover may provide a means of escape from a local optimum into other region and hence allows a more thorough search of the solution space. Although sound in theory, but in practice, crossover between parents may not always produce a better off spring. This probably explains why the result obtained by ***GA*** is inferior when compared to ***PSO***. Clearly, some other search operations are needed to improve upon the solution.

Figure 4a: Convergence rate of PSO and SPSOES (Gen=1000,Parents=20)

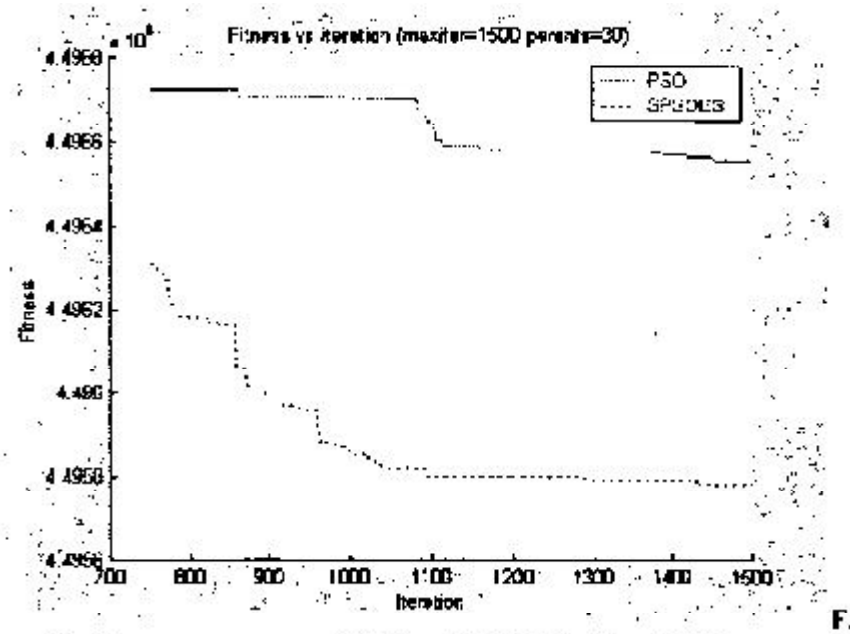



Figure 4b: Convergence rate of PSO and SPSOES (Gen=1500,Parents=30)

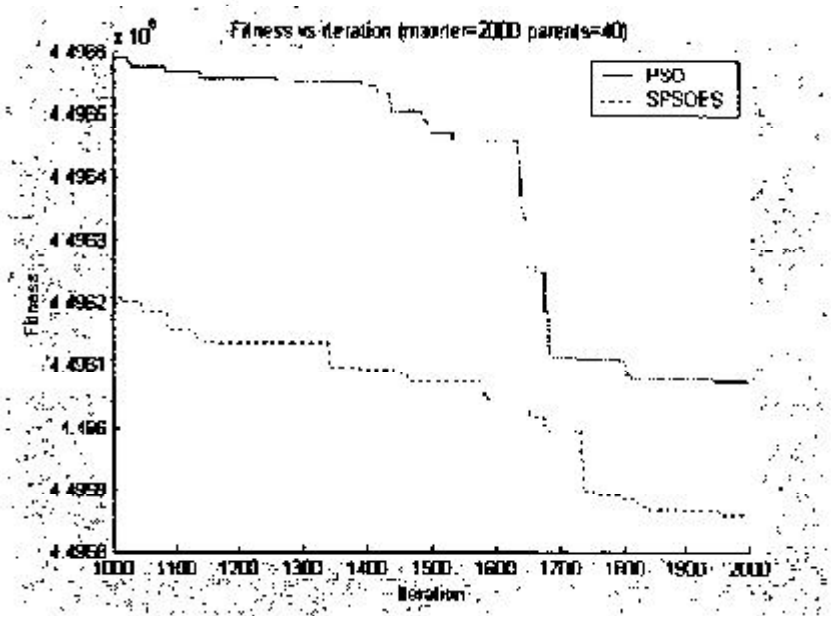



Figure 4c: Convergence rate of PSO and SPSOES (Gen=2000,Parents=40)

Unlike ***GA*** which replaces individuals. ***PSO*** models changes in individuals over time and all individuals survive into the next generation. The fate of each individual is constantly altered based on the global optimal point discovered so far. This swarming effect, which cannot be found in ***GA***, allows the population

to quickly converge into optimal regions of the search space. This probably explains why standard ***PSO*** can achieve better results than ***GA***.

### *C. Evolutionary Strategy*

The subtle difference between ***ES*** and ***GA*** is in the parameter representation. ***ES*** works with real values of the variables (phenotype) whereas CA works with binary strings which are subsequently mapped to object variables. Since ***ES*** works completely on a phenotypic level, one can represent more knowledge about the application domain into the coding of the problem. Parents are mutated to generate off springs. In the simulation, the best individuals are selected from the mutated and current population for the next population. This ensures that the surviving individuals have a higher average fitness progressively. The competitive selection pressure among candidates explains why ***ES*** can achieve a lower average cost when compared to standard ***PSO***.

### *D. Particle Swam Optimization Algorithm*

***PSO*** will be the most time efficient method to use when looking for a near optimal solution as the population size can be kept small. The superiority of ***PSO*** over heuristic and ***GA*** methods is clearly illustrated in ***Table*** 1. However, the average cost ofPS0 is higher than that of ***SPSOES***. This implies that although ***PSO*** is able to obtain satisfying optimal solution within a short time span, it still lacks the ability to continuously improve upon the solution.

### *E. Hybrid Spawned Particle Swarm Optimization and Evolutionary Strategy (SPSOES)*

The hybrid approach proposed here (***SPSOES***) with spawning and selection mechanism proves to be superior over classical ***PSO*** in the cost obtained. Although ***SPSOES*** is not as time efficient as standard ***PSO***, it provides more consistent and reliable results. This can be observed by the low average cost obtained by ***SPSOES*** over the three experimental settings. The convergence graphs in Figures.4a-4c illustrate that ***SPSOES*** is able to converge to better solution faster than ***PSO***.

## viii. Conclusion

- Particle swarm optimization-based approaches yield superior performance compared to ***GA*** or evolutionary strategy.
- The researcher also presents a hybrid spawning ***PSO*** and evolutionary strategy. In this approach, valuable features from both ***PSO*** and evolutionary strategy are combined to provide a simple hybrid model that is readily useable in many other applications.
- The results suggest that this hybrid model converges to better solution faster than standard ***PSO*** algorithm.
- It is envisaged that this hybrid approach can be easily implemented for similar optimization and scheduling problems to obtain better convergence.

## ix. References

[1] Qingsong Xu and Yangmin Li , "Stiffness Optimization of a 3-DOF Parallel Kinematic Machine Using Particle Swarm Optimization", 2006

[2] David L. Cushman , "A Particle Swarm Approach to Constrained Optimization Informed by "Global Worst"", 2007

[3] Syhlin Kuah and Joc Cing Tay , "Combining Simplex with Niche-based Evolutionary Computation for Job-Shop Scheduling", 2006

[4] Keshav P. Dahal and Nopasit Chakpitak, "Generator maintenance scheduling in power systems using metaheuristic-based hybrid approaches", 2006

[5] Tim Schoenharl and Scott Christley , "Patisserie: Support for Parameter Sweeps in a Fault-Tolerant, Massively Parallel, Peer-to-Peer Simulation Environment", 2005

[6] Chin Aik Koay and Dipti Srinivasan, "Particle Swarm Optimization-based Approach for Generator Maintenance Scheduling", 2003