M a t h e m a t i c a
B a l k a n i c a

New Series Vol. 23, 2009, Fasc. 3-4

A Particle Swarm Optimization Approach to the Multi Level Capacitated Minimum Spanning Tree Problem

Chryssa A. Papagianni, Christos A. Pappas, Nikos Lefkaditis, Iakovos S. Venieris

In the presented study Particle Swarm Optimization will be applied on two instances of the Multi Level Capacitated Minimum Spanning Tree Problem. Specifically a diversity preservation global variant of the PSO meta-heuristic will be presented. The particular PSO variant includes Gaussian mutation to avoid premature convergence and alternative selection of the flight guide per particle. Obtained results are compared with corresponding evolutionary approaches. Potential tree solutions are encoded/decoded using Network Random Keys. A real world network design case is introduced

1. Introduction

Network design problems apply to a wide range of application domains, spanning from transportation to telecommunications etc. Imposing a set of constraints to common polynomially solved problems such as the minimum spanning tree or the shortest path three problems, usually renders them NP-complete. The increasing complexity of network design problems calls for advanced optimization techniques.

Specifically in the telecommunications industry, one problem network planners usually face is the communication node (terminal, router etc) layout problem; that is the Capacitated Minimum Spanning Tree Problem (CMSTP). The CMST problem is fundamental to the design of centralized communication networks and has been studied extensively for its importance in practical applications. The aim is to find a minimum cost tree, spanning a given set of nodes whilst satisfying a set of capacity constraints associated with every arc in the network. For the CMST a single type of capacity is available for the interconnection of any two nodes in the network.

However in real world applications there are usually more facility types available to the network planner. The Multi Level Capacitated MST (MLCMST) problem generalizes the aforementioned CMST problem by allowing only a predefined set of capacities with different cost functions [10]. The problem formally is defined as: given a fully connected graph $G(N, A)$, node 0 represents the central node where traffic demands r_{k0} are gathered originating from the rest $|K| = |N| - 1$ nodes. To facilitate the traffic demands there are $|L|$ available capacity types $Z^1 < Z^2 < \cdots < Z^{lmax}$ with different cost functions C_{ij}^l denoting the cost of capacity of type l connecting nodes i, j. Only one facility type is allowed to be installed on every link. Since MLCMST is a generalization of the NP-hard CMST problem the problem is NP-hard [10].

Considering the CMST problem, various heuristic approaches have been presented in literature, classified either as classical heuristics or meta-heuristics [22]. While classical heuristics performs a limited exploration of search space and result in inferior solutions (e.g. the Esau-Williams algorithm [8], meta-heuristics are general purpose optimization methods that usually involve a large set of parameters that needs to be fine-tuned. However, especially for large scale instances of the CMST problem, these techniques effectively produce satisfying (sub) optimal solutions. Meta-heuristic approaches such as simulated annealing, tabu search and genetic algorithms have been successfully applied to the CMST problem space. Extended reference on the subject can be found in [10] [17]. An interesting review of background material on network design problems and the application of CMST can be found in [3]. Gavish was the first to address the capacitated minimum spanning tree problem in the context of communication networks, bearing no constraints in the number of hops from a source to a sink, employing non-bifurcated routing e.g. [11] [112]. A number of research studies on solving the MLCMST with genetic and other heuristic algorithms can be found in literature e.g. Gamvros et al [9], Berger et al [3], Martins et al [18] etc.

In the presented study Particle Swarm Optimization will be applied effectively on the MLCMST problem. Specifically a diversity preservation global variant of the PSO meta-heuristic will be presented and evaluated against a common genetic algorithm. In addition a real world network design instance in the region of Greece will be utilized. The application of PSO on constraint MST problems is not extended. In [13] the authors utilize a multi-objective discrete PSO algorithm on a multi-criteria MST problem. In [23] [24] the application of a local PSO algorithm employing uniform/gauss mutation on an instance of the Optimal Communication Spanning Tree is presented by the authors of this study. In [21] [20] a hybrid PSO method is utilized initially for the Shortest Path Tree problem and following for the Delay Constrained Least Cost Path problem.

In Section 2 an overview of the PSO algorithm is provided as well as the variations proposed in this study. In Section 3 the network design problem formulation is presented along with constraint-handling techniques adopted. In Section 4, the obtained results of the proposed methodology are critically evaluated against the evolutionary algorithms' performance in terms of design goals satisfaction and convergence behavior. In addition PSO is applied on an extended instance of the presented CMST problem to prove its efficiency on generating larger network topologies. In Section 5 the most important conclusions and future work are briefly discussed.

2. Particle Swarm Optimization

A. Basic PSO and variations

The concept of PSO is inspired by the flocking behavior of the birds [15]. Like evolutionary algorithms, PSO is a population based heuristic. Each potential solution $\vec{x_i} \in \Omega \subseteq R^n$ is called a particle and their aggregation forms the swarm P. The particles fly through the n-dimensional problem space with velocity $\vec{u_i} \in R^n$, subject to both deterministic and stochastic update rules to new positions, which are subsequently scored by a fitness function. The velocity of the particles is updated on each iteration according to their best encountered position $\vec{p_{ip}}$ (personal best) and the best position encountered by any particle $\vec{p_{il}}$ within its neighborhood $N_i \subset P$ (local best) or any particle $\vec{p_{ig}}$ within the swarm P (global best).Consequently the individual particles are drawn stochastically toward the positions of their own previous best performance and best performance of their neighbors, in accordance with the following equations (index i is omitted for clarity of illustration):

$$u_d(k+1) = u_d(k) + c_1\varrho_1(k)(p_{pd}(k) - x_d(k)) + c_2\varrho_2(k)(p_{gd}(k) - x_d(k)) \tag{2.1}$$

$$x_d(k+1) = x_d(k) + u_d(k+1)$$

where k denotes the iteration number, $x_d(k)$ and $u_d(k)$ represent the particle's current position and velocity for dimension $d = 1..n$, and $\varrho_1(k)$,$\varrho_2(k)$ are random numbers in the range $[0, 1]$.

The cognitive component $c_1\varrho_1(k)(p_{pd}(k) - x_d(k))$ associates the particle's own experience with its current position whereas the social component

$c_2\varrho_2(k)(p_{gd}(k) - x_d(k))$ is associated with social interaction between the particles of the neighborhood, resembling a group standard which individuals seek to attain. The acceleration coefficients c_1,c_2 along with the random vectors ϱ_1, ϱ_2 control the stochastic influence of the cognitive and social components on the overall velocity of the particle. Proper fine-tuning of c_1 and c_2 parameters may result in faster algorithm convergence, and reduce the probability of entrapment around sub-optimal solutions. The efficiency of the algorithm is based on its cooperative nature and is most efficient when c_1,c_2 coexist in good balance, that is $c_1 \approx c_2$ [7]. Default values of $c_1 = c_2 = 2$ were proposed in [15], but experimental results indicate that alternative configurations of $c_1\varrho_1$,$c_2\varrho_2$ depending on the problem, may produce superior results [2]. Despite the fact that acceleration coefficients are usually static with their optimized values found empirically, adaptive coefficients have also been proposed (e.g. [28]).

In the initial applications of the PSO algorithm, it was reported that the velocity vector exploded quickly to large values that resulted in particles diverging, moving beyond the boundaries of the search space. To control swarm explosion, the velocity clamping parameter $\vec{u}_{max}$ is specified as an upper limit to all dimensional components of the particle's velocity. When $\vec{u}_{max}$ is used, new velocities are constrained according to the relation $u_d(t) = sign(u_d(t))min\{|u_d(t)|, u_{max,d}\}$ where small values of $\vec{u}_{max}$ favor local exploration. Maximum velocity per dimension is defined as $u_{max,d} = \delta(x_{max,d} - x_{min,d})$ where $x_d \in [x_{min,d}, x_{max,d}]$ and $\delta(0,1]$ parameters are problem-dependent. However the velocity clamping mechanism does not confine the positions of the particles within the search space; it solely controls the step size of the move and the direction of the flight.

One of the most widely adopted improvements of the PSO algorithm is the introduction of inertia weight, which is employed to control the exploration/ exploitation ability of the algorithm [31]. The inertia weight controls the effect of the previous flight direction during velocity updates. However it does not eliminate the need for velocity clamping [32]. A large inertia weight favors global exploration with increased diversity whereas small value encourages local search:

$$(2.2)\quad u_d(k+1) = wu_d(k) + c_1\varrho_1(k)(p_{pd}(k) - x_d(k)) + c_2\varrho_2(k)(p_{gd}(k) - x_d(k))$$

Shi and Eberhart observed that choosing $w \in [0.8, 1.2]$ results in faster convergence, but when it exceeds a threshold value of 1.2, the algorithm fails to converge consistently. Van De Bergh et al [4] proved that the parameter configuration $\{w = 0.7298, c_1 = 1.49618, c_2 = 1.49618\}$ proposed by Eberhart and Shi [6] leads to better convergence characteristics (fitness value, convergence rate).

Therefore this is the parameter configuration that will be used in the presented study.

B. Pbest Hybrid PSO

A common problem with PSO is that it tends to prematurely converge to an equilibrium state [7], which results in the entrapment of the particles in the vicinity of sub-optimal solutions. The phenomenon usually emerges when the optimization problem's dimensionality increases and the fitness landscape is highly irregular (e.g. non-convex) and is mainly attributed to the loss of population diversity. On this direction in [24] the utilization of Gaussian mutation along with attractive repulsive PSO employing Von Neumann topology was presented (H-ARPSO). However the global version of the H-ARPSO algorithm does not exhibit the same satisfactory convergence behaviour on network design problems.

To overcome the problem of premature convergence for the global version of the basic PSO algorithm, each particle explicitly chooses its guide randomly from the set of particles' personal best vectors taking into consideration duplicates in the objective space. Specifically the selection scheme is implemented as roulette wheel wherep_i, the probability that a particle will be selected as guide is given by:

$$p_i = \frac{g_i(\vec{x_i})}{\sum_{POP} g_i(\vec{x_i})}$$

$$g_i(\vec{x_i}) = 1/N_{f(\vec{p}_{ip})}$$

where $N_{f(\vec{p}_{ip})}$ is the number of particles with the same fitness value $f(\vec{p}_{ip})$ as i-th particle.

The technique resembles the basic version of Cooperative Learning PSO [16] in the sense that other particles' pbests are also used as exemplars to guide a particle's flying direction, though not differentiating the leader per dimension. By taking into consideration solutions that produce the same tree topology, hence the same objective value, diversity is increased. These solution vectors are not necessarily identical according to NetKeys encoding. Particles that are aggregated in areas of the search space that deliver the same solution have overall the same probability of being selected as every other particle in the swarm.

In addition, mutation has proved to successfully complement PSO and improve its exploration capability. Mutation operators usually applied with PSO include [7] linear mutation operators, non- linear mutation operators, Gaussian mutation, Cauchy mutation etc. Mutation can be applied on the velocity vector, the position vector, personal best position or neighborhood best position. In this case, a modified Gaussian mutation scheme is utilized, presented by the authors in [24] where the position vector of each particle is mutated. Each coordinate of every particle has a chance of getting mutated with probability p_m according to:

$$x_d{}'(k) = x_d(k) + n_d r_d(k)(1 - k/k_{max}) \tag{2.3}$$

where

$$n_d \sim N(0,1) \quad , \quad r_d(k) = \begin{cases} x_{max,d} - x_d(k) & \text{if } q(k) \sim U(0,1) > 0,5 \\ x_{min,d} - x_d(k) & \text{otherwise} \end{cases}$$

As the algorithm evolves in time the mutation step size decreases to zero when $k = k_{max}$. Therefore wide-range searches are possible at initial stage while solutions are refined in later time steps.

The time complexity of the aforementioned hybrid PSO remains governed by the position update function that is $O(Md)$ where M denotes the swarm size and d problem dimensionality.

C. Tree encoding

Of crucial importance to the success of the optimization procedure is the choice of candidate solutions representation. Regarding tree representation on evolutionary algorithms a variety of encodings have been proposed as characteristic vectors, predecessors, Prufer numbers, link and node biasing, edge sets etc. An interesting comparison with corresponding references is presented in [27].

In the presented work a tree is encoded with the network random keys (NetKeys) scheme introduced in [29]. In this encoding, given a connected graph $G = (N, A)$ where N represents the node set and A the arc set, a tree solution $\vec{x}_i \in \Omega \subseteq R^n$ is represented by a string of $n = |N|(|N| - 1)/2$ continuous real-valued weights, one for each arc of the complete graph where $x_{id} \in [0, 1]$. These continuous variables can be interpreted as the importance of the link, allowing for distinction among more and less important links. In order to decode the string, the edges are sorted by their weights and Kruskal's minimum spanning tree algorithm is used to create the tree. Therefore for a network of 16 nodes, 120 control parameters are needed, bounded within $[0, 1]$.

NetKeys were initially fabricated for evolutionary and GA applications [29]. Decoding the solution for evaluation is computationally expensive $O(nlogn)$. However there are certain crucial advantages for utilizing NetKeys in general and particularly along with PSO. As the construction of the tree is based on the relative ordering of the keys, the locality is high, thus mutation is applied effectively. In addition the use of Kruskal's algorithm ensures that the resulting graph will be a valid tree. Moreover, using NetKeys that represent a tree with normalized continuous real values, allows for direct application of the position update formulas.

3. Tree Network Design Problem

A. Problem Formulation

The Multi Level Capacitated MST problem is formally is defined as: given a fully connected directed graph $G(N, A)$, node 0 represents the central node where traffic demands $h^k = r_{k0}$ are gathered originating from the rest $|K| = |N| - 1$ nodes. To facilitate the traffic demands there are $|L|$ available capacity types $Z^1 < Z^2 < \cdots < Z^{lmax}$ with different cost functions C^l_{ij} denoting the cost of capacity of type l connecting nodes i, j. Only one facility type is allowed to be installed on every link. The deployment cost of the network comprises of the sum of costs of all links that are present on the tree solution.

$$Minimize \sum_{(i,j)\in A} \sum_{l\in L} C^l_{ij} y^l_{ij} \tag{3.1}$$

$$\sum_{j\in N} f^k_{ji} - \sum_{l\in N} f^k_{il} = \begin{cases} -1 & \text{if } i = k \\ 1 & \text{if } i = 0 \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in N, \forall k \in K \tag{3.2}$$

$$\sum_{k\in K} f^k_{ij} h^k \leq \sum_{l\in L} Z^l y^l_{ij} \quad \forall (i,j) \in A \tag{3.3}$$

$$\sum_{l\in L} y^l_{ij} = x_{ij} \quad \forall (i,j) \in A \tag{3.4}$$

$$f^k_{ij} \leq x_{ij} \quad \forall (i,j) \in A, k \in K \tag{3.5}$$

$$\sum_{j\in N} x_{0j} = 0 \tag{3.6}$$

$$\sum_{j \in N} x_{ij} = 1 \quad \forall i \in K \tag{3.7}$$

$$x_{ij} + x_{ji} \leq 1 \quad \forall (i,j) \in A \tag{3.8}$$

$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in A \tag{3.9}$$

$$y_{ij}^{l} \in \{0,1\} \quad \forall (i,j) \in A, l \in L \tag{3.10}$$

$$f_{ij}^{k} \in \{0,1\} \quad \forall (i,j) \in A, k \in K \tag{3.11}$$

Four types of facility types are used. The binary variable x_{ij} denotes whether link (i,j) is present on the tree solution, y_{ij}^{l} denotes whether link type l is installed on link (i,j) and f_{ij}^{k} denotes whether the flow originating from source k is routed through the link (i,j).

Constraints (3.2), (3.5), (3.6), (3.7) and (3.8) ensure that the topology is a directed spanning tree towards the central node. More specifically, (3.8) ensures that only one-way link is installed between edges i,j , (3.6), (3.7) ensure that each node i has only one outgoing link, except from the root node (the sink), (3.5) ensures that the flow of source k can pass through link (i,j) only if link (i,j) is present on the solution tree and (3.2) ensures flow conservation.

Constraint (3.4) ensures that exactly one link type l is installed on link (i,j) while (3.5) ensures that the sum of all flows that are routed through the link (i,j) do not exceed the capacity Z^{l} of linktype l installed on link (i,j). The distances among nodes are computed as the Euclidian distance d_{ij}. The cost C_{ij}^{l} per link type l is constituted of a fixed setup cost and variable cost piecewise linear (Table 1).

B. Handling of constraints

Constraint handling is of extreme importance when dealing with real-world problems. Constraints usually need to be incorporated into the optimization procedure to avoid convergence towards infeasible solutions. Some of the most popular constraint-handling techniques are methods either preserving feasibility of solutions (i.e. [14], [26]]. Such methods ensure that particles' adjustments do not violate any constraints by not allowing particles to move to infeasible space. Particles are generated into the feasible domain and they move through the search space following rules so that the resulting solutions still lie within the feasible domain.

Moreover special repair methods may be applied to move the particles towards feasible space / change the particles to feasible ones. Falling in this category for boundary constraints, soft (e.g. reflective or absorbing boundary conditions) and hard boundary conditions (position clipping) may be utilized in PSO to constrain the obtained solutions within the search space defined [19]. Reflective Boundary Conditions inverse and Absorbing Boundary Conditions nullify the velocity vector. In the event of RBC, if $x_d(k+1) \notin [x_{min,d}, x_{max,d}]$ then $u_d(k+1) = -u_d(k+1)$. For most optimization problems the RBC has been shown to improve the convergence behavior of the algorithm [19].

In addition constrained single-objective optimization problems can be transformed into unconstrained ones. One very common technique in this sense is penalty functions (i.e. [25]). Multi-objective optimization concepts have been also used to handle constraints in single-objective optimization problems. Two basic methods [5] have been adopted in that direction, the latter being the most popular (i) bi-objective transformation where the sum of constraint violation is used as a second objective function and (ii) multi-objective transformation where the constrained optimization problem is transformed to a multi-objective optimization problem with $(m+1)$ objectives, where m is the number of constraints, utilizing either non-Pareto or Pareto based techniques.

Regarding the particular problem, a penalty capacity type has been added to the problem definition so as to penalize solutions that violated constraint (7). Moreover, utilizing NetKeys as the tree encoding scheme ensures that all tree topology con straints are met. Finally in order to keep the particles within the boundaries of the search space, reflective boundary conditions are applied.

4 Results

Two node sets of different size (16 and 30 nodes) were used for the CMST problem. These test instances involve the design of a tree-structured network in a geographical region in Greece. The set of nodes to be connected was chosen randomly from a set of possible locations with the corresponding geographical coordinates. The root (gateway) of the tree (network) is chosen explicitly as the administrative center of the region. Necessary input for the model is considered apart from node location, the set of available capacities and their corresponding cost functions (Table 1) as well as peak inter- node traffic matrix. Illustrative peak source/destination traffic is randomly generated from a uniform distribution (0-10 Mbps). The proposed pbest hybrid PSO (pbest H-PSO) algorithm is applied to the instances of the MLCMST problem and compared against a genetic algorithm, as well as, a global and a local version of the PSO algorithm that adopts the presented mutation scheme (local/global Hybrid PSO, local/global H-PSO). Random population initialization is utilized.

A Java based optimization toolbox, created by the authors (also in [23], [24]), was utilized in this study where various meta-heuristics (PSO, GAs) and Tree encoding algorithms (Prufer, Netkeys etc) are implemented.

The mutation operator[1] described in section II was employed on all four examined algorithms with probability $pm = 1\%$. Parameter configuration $\{w = 0.7298, c1 = c2 = 1.49618,\}$ was selected (see section II-A). Velocity clamping parameter is crucial to the convergence behaviour of the algorithm, since it defines the maximum allowable change of particles' position at each generation while exploring the search space. Often it has been set at 10% - 20% of the dynamic range of each control parameter (e.g.[24]) providing a good trade-off between convergence speed and convergence to a global minimum point. In this case velocity clamping was set to $u_{max,d} = 0,1(x_{max,d} - x_{min,d})$. The local version of the PSO algorithm uses the Von Neumann neighborhood topology [7], where particles are connected using a 2-dimensional grid and each particle is connected to its four neighbor particles (above, below, right, and left particles). Concerning the GA employed, algorithmic parameters were fine-tuned accordingly. In this sense ranking selection was utilized along with uniform crossover with probability pcross=90%. The population is complemented (10%) with elitism. On both cases a population of 100 particles/individuals was selected that is commonly used in many applications of the PSO algorithm e.g. pp. 427 [1]. In the 16-node network design the population was allowed to evolve for 1000 iterations, whereas in the 30-node design 2000 iterations were used.

In order to evaluate the effect of the algorithm's parameters and to quantify the performance of PSO, measures of accuracy, reliability, robustness and efficiency are examined. Accuracy refers to the quality of the obtained solution. Accuracy is expressed as the error of the mean best fitness value computed over executed runs: $f_{err} = |f_{opt} - f_{average}|$. Given the fact that the optimal solution is not known for each examined problem instance, we used ferr regarding the best fitness value found as fopt (e.g. figure 1 for the 16-node network design). Complementary to the accuracy measure, robustness of the swarm is quantified by the standard deviation of the aforementioned error ferr. A small value indicates a small range of values where the swarm converges. Reliability is measured as a percentage of the simulations that reached the optimum solution denoted as success rate. Finally the efficiency of the algorithm is expressed as the average number of iterations to converge to the best found solution.

[1]The mutation step size $(1 - k/k_{max})$ was not applied to the GA..

A. 16-node network design

The presented results were obtained after repeating the optimization process for 100 times in order to obtain more statistically important indicators on the algorithm's convergence behavior over several runs. The results regarding the algorithmic performance are presented in Table 3.

In Figure 1 and Table 2, the results of the design process are depicted; that is dimensioning and load for optimal topology found. The maximum available capacity was set to $Z^l_{max} = 34$ Mbps. It is obvious from the utilization of the links that traffic routed through the links does not exceed the installed capacities; hence constraint (3.2) is met.

In Figure 2 the convergence behavior of the pbest H-PSO is presented against H-PSO with a global neighborhood, H-PSO with a local neighborhood (Von Neumann) and GA while solving the single objective network design problem. The best run for the gbest H-PSO version was able to locate the global best configuration of the network within 81 generations, requiring a total of 8,100 fitness function evaluations (for a swarm population of 100 particles), while for the lbest version is 14,500, the pbest version is 30,000 and the GA 10,300.

As depicted also in Figure 2 the pbest H-PSO converges slower than the other examined algorithms, while the GA and the global H-PSO have the fastest convergence. Due to the trade-off between fast convergence and quality of found solution, the proposed algorithm outperforms all the other examined algorithms in terms of f_{err}, $\sigma_{f_{err}}$ and success rate (Table 3). In addition it needs fewer iterations than the examined variations of PSO to find the optimal solution (including unsuccessful attempts).

In Figure 3 the convergence behaviour for a single run of the pbest H-PSO can be evaluated against the diversity of the fitness value of the personal best for each member of the swarm. We observe that after the 283rd iteration where the best found solution is reached (therefore $f_{err} = 0$) particles are gradually drawn to this solution in the objective space. In the 400th iteration all members of the population are found in close vicinity of the optimal (indicating loss of diversity $|p_p - p_g{}'|$ where $p_g{}'$ is selected guide per particle).

B. 30-node network design

Regarding the 30-node design problem, the results presented in Table 5 were obtained after repeating the optimization process for 30 times. Results in terms of algorithmic efficiency are consistent with the 16-node example.

As depicted also in Figure 4 the GA and the global H-PSO converge faster compared to local and pbest H-PSO. As a result they present higher values of

f_{err}, $\sigma_{f_{err}}$. The proposed algorithm also in this case outperforms the other PSO versions as well as the GA (table 5).

In Table 4, the results of the design process are depicted; that is dimensioning and load for optimal topology found. The maximum available capacity was set to $Z^{lmax} = 155$ Mbps. It is obvious from the utilization of the links that traffic routed through the links does not exceed the installed capacities; hence constraint (3.3) is met.

5. Conclusions

In the particular study, a variant of a single objective PSO algorithm was applied successfully on a communications network topology design problem. Specifically two instances of the Multi-Level Capacitated Minimum Spanning Tree problem were introduced. The presented hybrid version of the particle swarm algorithm applies Gaussian mutation to the swarm whereas an additional diversity preserving mechanism is applied; each particle chooses explicitly its guide from the set of particles' personal best vectors taking into consideration duplicates in the objective space. NetKeys representation of candidate solutions is utilized. The proposed methodology shows an improvement in the optimization process in comparison to GA and standard PSO.

References

[1] A. Abraham, C. Grosan, W. Pedrycz, *Engineering Evolutionary Intelligent Systems* , Springer, 2008.

[2] M. S. Arumugam, M.V.C. Rao, On the improved performances of the particle swarm optimization algorithms with adaptive parameters, cross-over operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems , *Applied Soft Computing*, **8**, Issue 1, 2008, 324-336.

[3] D. Berger, B. Gendron, J.-Y. Potvin, S. Raghavan, P. Soriano, Tabu Search for a Network Loading Problem with Multiple Facilities, *Journal of Heuristics*, **6**, No. 2, 2000, 253-267.

[4] F. van den Bergh, A.P. Engelbrecht, A study of particle swarm optimization particle trajectories, *Information Sciences*, **176**, Issue 8, 2006, 937-971.

[5] C. A. Coello Coello, G. B. Lamont, D.A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Springer US, 2007.

[6] R.C. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, in: *Proceedings of the IEEE Congress on Evolutionary Computation*, 84-88, 2000.

[7] A.P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, Wiley 2006.

[8] L.R. Esau and K.C. Williams, On teleprocessing system design. Part II - A method for approximating the optimal network, *IBM Systems Journal*, **5**, Issue 3, 1966, 142-147.

[9] I. Gamvros, B. Golden, An evolutionary approach to the multi-level capacitated minimum spanning tree problem in *Telecommunications Network Design and Management*, Kluwer Academic Publishers, 2003, 99-124.

[10] I. Gamvros, B. Golden, D. Stanojevic, Tutorial: Heuristic Search for Network Design, in *Emerging and Applications in Operations Research*, Springer, 2005.

[11] B. Gavish, Formulations and algorithms for the capacitated minimal directed tree problem, *Journal of the ACM*, **30**, Issue 1, 1983, 118-132.

[12] *B. Gavish*, Topological Design of Telecommunications Networks - Local Access Design Methods, *Annals of Operations Research*, **33**, 1991, 17-71.

[13] W. Guo; G. Chen; X. Feng; L. Yu, Solving Multi-criteria Minimum Spanning Tree Problem with Discrete Particle Swarm Optimization, *Proceedings of the Third International Conference on Natural Computation*, **4**, 2007, 471-478.

[14] X. Hu, RC. Eberhart, Solving constrained nonlinear optimization problems with particle swarm optimization., In: *Proceedings of the sixth world multiconference on systemics, cybernetics and informatics*, 2002, 203-206.

[15] J. Kennedy, R. Eberhart, Particle swarm optimization, in *IEEE International Conference on Neural Networks*, 1995, 1942-1948.

[16] J. J. Liang, A. K. Qin, P. N. Suganthan, S. Baskar, Evaluation of Comprehensive Learning Particle Swarm Optimizer, *LNCS Neural Information Processing*, 2004, 230-235.

[17] F. Mathew, C. Rego, Recent advances in heuristics for the capacitated minimum spanning tree problem, *The Decision Sciences Institute Conference*, 2006, 31021-31026.

[18] A. X. Martins, M. C. Souza, M. J. Souza, T. A. Toffolo, GRASP with hybrid heuristic-subproblem optimization for the multi-level capacitated minimum spanning tree problem, *Journal of Heuristics*, **15**, Issue 2, 2009, pp. 133- 151.

[19] S. Mikki, A. Kishk, Improved particle swarm optimization technique

using hard boundary conditions, in *IEEE Microwave and Technology Optical Letters*, **46**, Issue 5, 2005, 422-426.

[20] A.W. Mohemmed, N.C. Sahoo, Cooperative Particle Swarm Optimization for the Delay Constrained Least Cost Path Problem,*LNCS Evolutionary Computation in Combinatorial Optimization*, 2008, 25-35.

[21] A.W. Mohemmed, N.C. Sahoo, Efficient Computation of Shortest Paths in Networks Using Particle Swarm Optimization and Noising Metaheuristics, *Discrete Dynamics in Nature and Society,* **2007**, Article ID 27383, 2007.

[22] T. Oncan, Design of capacitated minimum spanning tree with uncertain cost and demand parameters, *Information Sciences*, **177**, Issue 20, 15 October 2007, 4354-4367.

[23] C. Papagianni,K. Papadopoulos,C. Pappas, N.Tselikas, D. Kaklamani, I. Venieris, Communication Network Design Using Particle Swarm Optimization, in: *International Multi-conference on Computer Science and Information Technology*, 2008, 915 - 920.

[24] C. Papagianni,K. Papadopoulos,C. Pappas, N. Tselikas, D. Kaklamani, I. Venieris, Communication network design using particle swarm optimization, to be published in *Analele Universitatii din Timisoara*, **XLVII**, issue 2, 2009

[25] K.E. Parsopoulos, M.N. Vrahatis, Particle swarm method for constrained optimization problems., In: *Proceedings of the Euro-international symposium on computational intelligence*, 2002, 214-220.

[26] U. Paquet, A. Engelbrecht, Particle Swarms for Linearly Constrained Optimisation, in *Fundamenta Informaticae*, **78**, Issue 1-2, 2007, 147-170.

[27] G.R. Raidl, B.A. Julstrom, Edge sets: an effective evolutionary coding of spanning trees, *IEEE Transactions on Evolutionary Computation*, **7**, 2003, Issue 3, 225-239.

[28] A. Ratnaweera, S. K. Halgamuge, H. C. Watson, article Swarm Optimization with Self-Adaptive Acceleration Coefficients, *International Conference on Fuzzy Systems and Knowledge Discovery*, 2002, 264-268.

[29] D. Rothlauf, A. Goldberg, A. Heinz, Network random keys - a tree network representation scheme for genetic and evolutionary algorithms, *Evolutionary Computation*, **10**, Issue 1, 2002, 75-97.

[30] K. P. Scheibe, C. T. Ragsdale, A model for the capacitated, hop-constrained, per-packet wireless mesh network design problem, *European Journal of Operational Research*, **197**, Issue 2, 1 September 2009, 773-784.

[31] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, *Proceedings of the IEEE International Conference on Evolutionary Computation Anchorage*, 1998, 69-73.

[32] Y. Shi, R. C. Eberhart, Parameter Selection in Particle Swarm Optimization, Evolutionary Programming VII, *Lecture Notes in Computer Science*, **1449**, 1998, 591-600.

[33] OTE, Hellas stream tariffs, 2005.

School of Electrical and Computer Engineering,
National Technical University of Athens,
Iroon Polytechneioy 9 Str.
15773 Zografou
Athens, GREECE
E-mail: chrisap@telecom.ntua.gr, chrispap@icbnet.ntua.gr,
el01168@central.ntua.gr, venieris@cs.ntua.gr

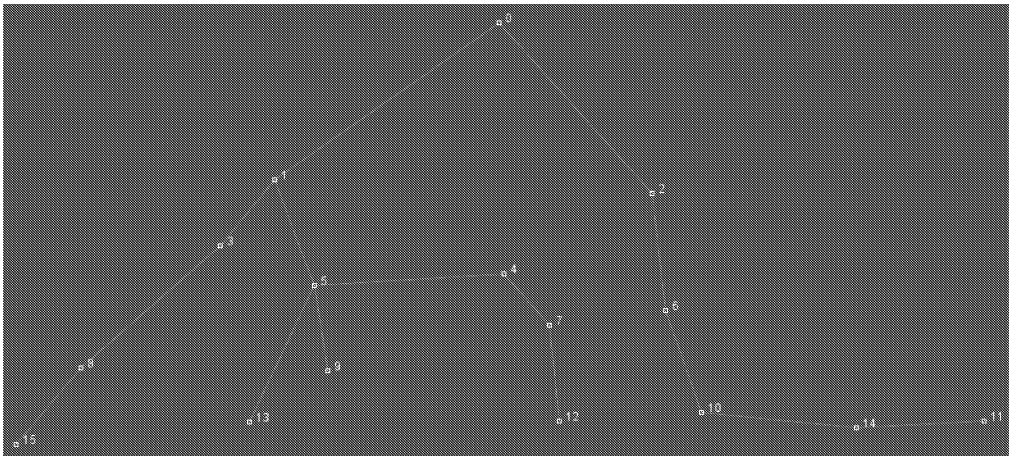


Figure 1: Optimal solution found (5,377.7€)

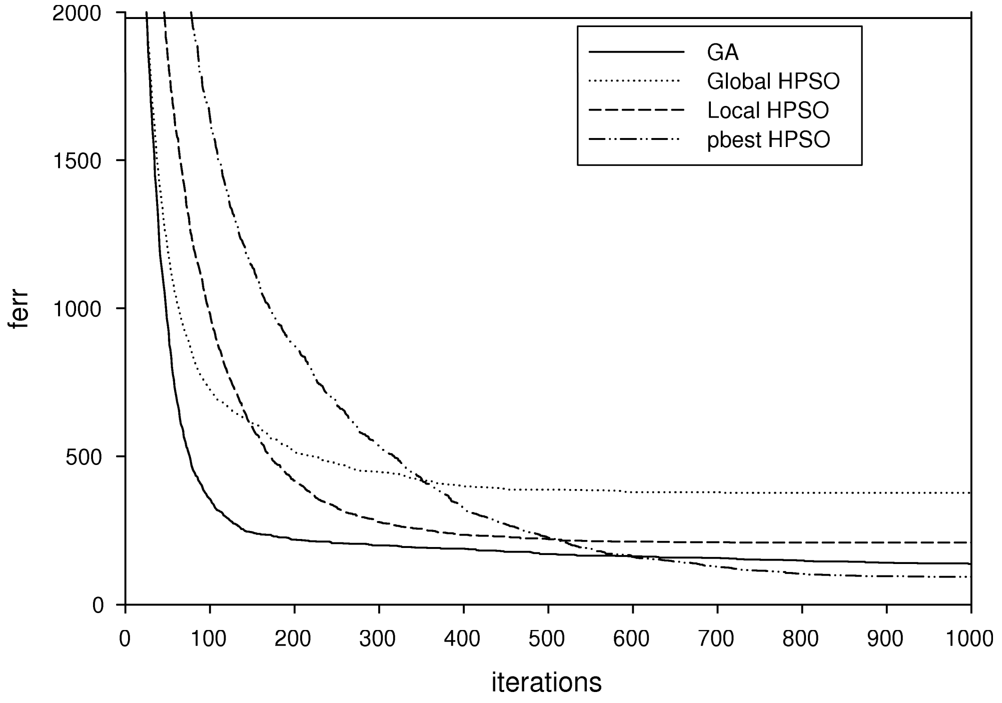
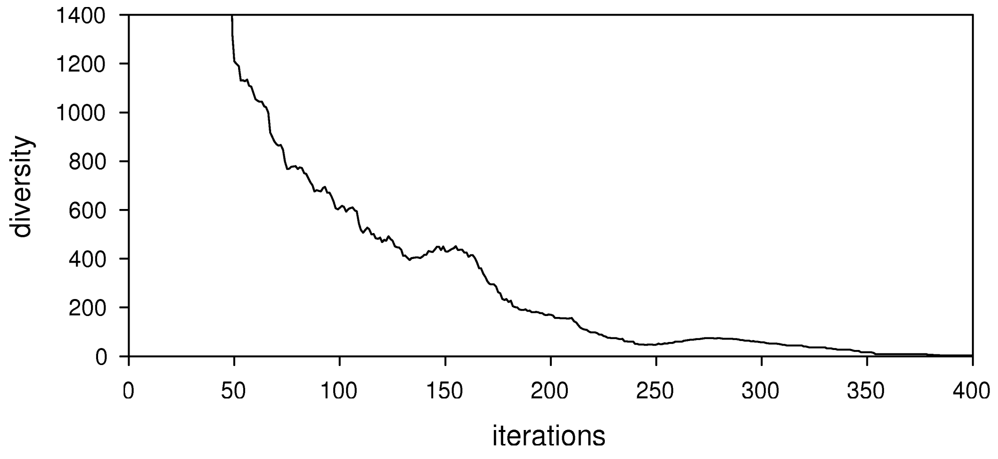


Figure 2: Convergence behavior - 16 Nodes



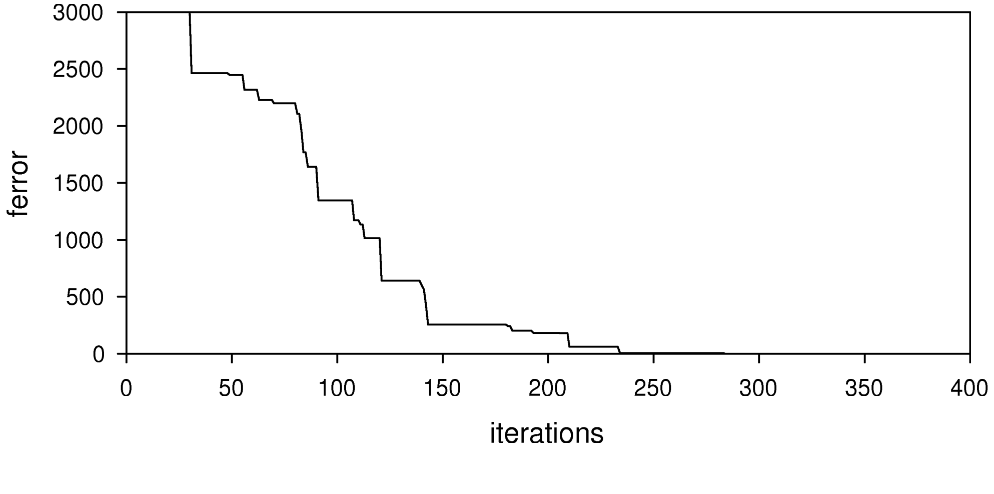


Figure 3: Convergence behavior and Fitness diversity of pbest hybrid PSO

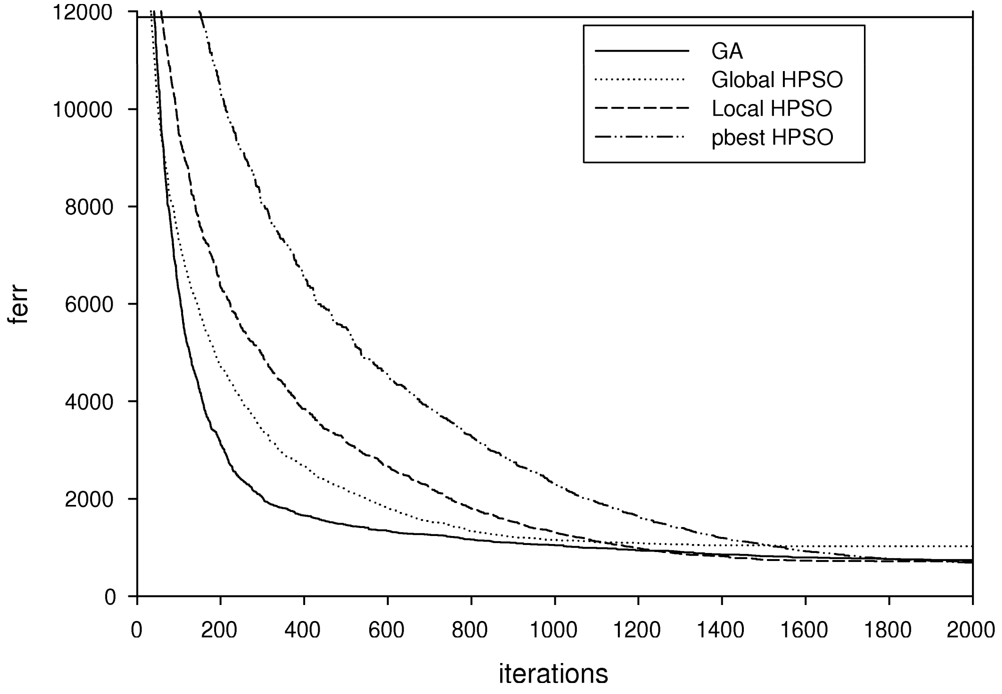


Figure 4: Convergence behavior - 30 Nodes

Z^l	d_{ij}(km)	C^l_{ij}(€)
512Kbps	[0,35]	3.34 d_{ij}
	(35,70]	116.9+1.91(d_{ij} -35)
	(70,150]	183.75+1.34(d_{ij} -70)
	(150, ∞)	290.95+1.26(d_{ij} -150)
1024Kbps	[0,35]	6.37 d_{ij}
	(35,70]	222.95+3.72(d_{ij} -35)
	(70,150]	183.75+1.34(d_{ij} -70)
	(150, ∞)	560.35+2.44(d_{ij} -150)
2048Kbps	[0,35]	10.23 d_{ij}
	(35,70]	358.05+6.05(d_{ij} -35)
	(70,150]	569.80+4.19(d_{ij} -70)
	(150, ∞)	905.00+4.19(d_{ij} -150)
34Mbps	[0,35]	75.00 d_{ij}
	(35,70]	2,625.00+45.00(d_{ij} -35)
	(70,150]	4,200.00+29.47(d_{ij} -70)
	(150, ∞)	6,557.60+26.15(d_{ij} -150)
155Mbps	[0,35]	188.00 d_{ij}
	(35,70]	6,580.00+112.00(d_{ij} -35)
	(70,150]	10,500.00+72.98(d_{ij} -70)
	(150, ∞)	16,338.40+65.25 (d_{ij} -150)
penalty capacity	[0, ∞)	50,000+500000 d_{ij}

Table 1: Cost per link capacity type C^l_{ij} per km [33]

Link	Capacity (Mbps)	Utilization	Distance (km)
8-3	34	21.6%	7.48
5-1	34	61.3%	4.57
1-0	34	94.1%	11.07
10-6	34	39.4%	4.70
5-4	34	44.6%	7.70
2-0	34	83.4%	9.26
6-2	34	55.8%	4.75
14-10	34	29.7%	6.29
15-8	34	6.8%	4.046
14-11	2.048	51.5%	5.21
13-5	2.048	57.3%	6.10
9-5	2.048	82.8%	3.49
3-1	34	35.4%	3.46
7-4	2.048	92.8%	2.79
12-7	2.048	92.8%	3.90

Table 2: 16-Node optimal dimensioning (5,377.7€)

	Population=100 , Runs=1000			
	global H-PSO	local H-PSO	GA	pbest H-PSO
f_{err}	377.59	209.62	137.52	93.28
$\sigma_{f_{err}}$	231.18	194.14	167.01	146.19
It.	925.06	778.77	656.32	768.75
SR	11%	35%	46%	63%

Table 3: PSO-GA Comparison - 16 Node Design

Link	Capacity (Mbps)	Utilization	Distance (km)
24-17	34000	0.59	6.32
23-21	34000	0.14	4.79
28-24	34000	0.15	4.47
6-2	34000	0.32	4.75
16-10	34000	0.49	1.85
10-6	34000	0.48	4.36
8-3	34000	0.51	7.47
19-11	2048	0.82	4.17
9-5	2048	0.49	3.49
23-18	34000	0.42	10.01
20-15	34000	0.23	3.59
17-12	34000	0.73	6.70
27-25	34000	0.40	4.16
2-0	34000	0.10	9.26
15-8	34000	0.30	4.05
4-1	155000	0.83	10.02
29-24	34000	0.31	4.47
19-10	34000	0.89	8.58
5-3	34000	0.16	4.11
14-10	34000	0.86	6.29
26-25	34000	0.21	7.42
13-8	2048	0.57	7.18
27-14	34000	0.60	20.77
12-7	34000	0.95	3.89
7-4	155000	0.79	2.78
3-1	34000	0.78	3.45
29-22	34000	0.15	4.43
10-7	155000	0.56	7.08
19-18	34000	0.68	9.14

Table 4: 30-Node optimal dimensioning (14,911€)

	Population=100 , Runs=2000			
	global H-PSO	local H-PSO	GA	pbest H-PSO
f_{err}	1,026.21	719.37	746.04	691.03
$\sigma_{f_{err}}$	674.946	378.50	558.00	325.37

Table 5: PSO-GA Comparison - 30 Node Design