# A Grid Enabled Parallel Hybrid Genetic Algorithm for SPN

Giuseppe Lo Presti[1,2], Giuseppe Lo Re[2], Pietro Storniolo[2], and Alfonso Urso[2]

[1] Dinfo – Università di Palermo.
[2] ICAR - Istituto di Calcolo e Reti ad Alte Prestazioni
C.N.R. – Consiglio Nazionale delle Ricerche, Palermo, Italy
{lopresti, lore, storniolo, urso}@icar.cnr.it

**Abstract.** This paper presents a combination of a parallel Genetic Algorithm (GA) and a local search methodology for the Steiner Problem in Networks (SPN). Several previous papers have proposed the adoption of GAs and others metaheuristics to solve the SPN demonstrating the validity of their approaches. This work differs from them for two main reasons: the dimension and the features of the networks adopted in the experiments and the aim from which it has been originated. The reason that aimed this work was namely to assess deterministic and computationally inexpensive algorithms which can be used in practical engineering applications, such as the multicast transmission in the Internet. The large dimensions of our sample networks require the adoption of an efficient grid based parallel implementation of the Steiner GAs. Furthermore, a local search technique, which complements the global search capability of the GA, is implemented by means of a heuristic method. Finally, a further mutation operator is added to the GA replacing the original genome with the solution achieved by the heuristic, providing thus a mechanism like the genetically modified organisms in nature. Although the results achieved cannot be applied directly to the problem we investigate, they can be used to validate other methodologies that can find better applications in the telecommunication field.

**Keywords:** Steiner Problem, Parallel Genetic Algorithm, Grid Computing.

## 1 Introduction

The Steiner Problem in Networks (SPN) [5] is a classic combinatorial optimization problem which, in its general case decision version, has been demonstrated [2] NP-complete. Its applications cover many scientific fields such, for instance, the VLSI and pipeline design, the Internet multicast routing, the telephone network design, etc. Many efforts have been produced in the last years to design polynomial-time algorithms to determine sub-optimal solutions: several heuristics have been developed capable of providing approximate solutions [3], [4]. Mathematical proofs constrain the solutions determined by these heuristics to

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3036, pp. 156–163, 2004.

the optimal solution, binding them by some multiplicative factors. This property allows their adoption for many applications. Among the practical applications of the SPN there is the construction of a minimal distribution tree to connect a set of Internet routers involved in a multicast transmission. The extremely dynamic nature of this application imposes the development of efficient heuristics capable of determining, in a very short time, sub-optimal solutions that however may represent good approximations. In order to validate the effectiveness of a given algorithm it is useful to compare the approximations obtained with the exact solutions. However, the NP-complete nature of the problem, at least to the current knowledge, does not allow to perform complete algorithms for graphs whose dimensions are comparable with the current size of the Internet. Among the most efficient approximating algorithms, recently some metaheuristics such as Genetic Algorithms [9], tabu-search [10], and Simulated Annealing [7] have been proposed. Although these approaches can be considered the best approximating methodologies, they suffer the disadvantage of their non-deterministic behavior that does not allow their adoption in fields requiring a distributed coordination among several independent entities. However, the good performances produced by these evolutionary methods suggests the idea to exploit their results as an assessment term. The best suitable approach for exploiting the coarse grain parallelism available in our laboratory is a parallel implementation of genetic algorithm. This technique results extremely scalable and the software implementation, we carried out, allows us to extend its execution to very large grid computing systems, which currently are becoming available on the Internet. The relevant computing power available allowed us to solve very large instances of the problem, and in most of the cases to determine the best solutions ever obtained. The experiments have been carried out on several different sets of graphs, characterized by different topological features, with the aim to effectively evaluate and compare the performances over a wide range of samples. Furthermore, to demonstrate the general validity of the methodology we tested our implementation over a classical public library set of experiments, SteinLib [13], which represents a commonly accepted assessment term for the Steiner problem. The remainder of the paper is organized as follows. The Steiner Problem in Network is formulated in section II. Section III contains the description of the parallel hybrid genetic algorithm, and section IV describes the experimental results. Finally, section V concludes this work and discusses future directions.

## 2 The Steiner Tree Problem in Networks

Formally, the Steiner Tree Problem in Networks can be formulated as follows. Let $G = (V, E)$ be an undirected graph, $w : E \to R^+$ a function that assigns a positive weight to each edge, and $Z \subseteq V$ be a set of multicast or terminal nodes. Determine a connected subgraph $G_S = (V_S, E_S)$ of $G$ such that:

- $Z \subseteq V_S$;
- the total weight $w(G_S) = \sum_{e \in E_S} w(e)$ is minimal.

The $V_S - Z$ set is called the Steiner nodes set and is denoted by $S$. Since the weight function assumes positive values, the resulting subgraph is called the Steiner minimum tree $T$, which spans each node in $V_S$. Throughout this paper, let $n = |V|, m = |E|, p = |Z|$. Many heuristics proposed in the past years are capable of identifying sub-optimal solutions with polynomial time complexities: among these, the Distance Network Heuristic (DNH) [5], the Shortest Path Heuristic (SPH) [3], the K-Shortest Path Heuristic (K-SPH) [1], the Average Distance Heuristic (ADH) [4], and the Stirring heuristic [11], which were used in the experimental tests. In particular, K-SPH builds a forest of subtrees joining together the closest nodes or subtrees until a single solution tree has been obtained. ADH is a generalization of K-SPH. It repeatedly connects nodes or subtrees through the most central node, which is determined by a heuristic function, and terminates when a single tree remains, spanning all the Z-nodes. The ADH algorithm is the most effective among these heuristics, though the better performances involve a higher computational cost, $O(n^3)$ versus the upper bound of $O(pn^2)$ of all other heuristics. The Stirring heuristic is a local search optimization method, constrained to assume a deterministic behavior, which uses a solution found from the above heuristics to determine better solutions.

## 3 Parallel Genetic Algorithm

A Genetic Algorithm (GA) provides a universal optimization technique that imitates processes of genetic adaptation that occur in natural evolution. By using this analogy, GA is able to evolve towards a solution for real-world optimization problems. The main advantage of the GA is its capability of achieving global optimization solution even for nonlinear, high-dimensional, multimodal and discontinuous problems [6].

Genetic Algorithms are naturally suited to be implemented on a parallel architecture. A survey on parallel GAs can be found in [8]. Several approaches to parallel implementations of GAs have been proposed ([15]). Among these, for the solution of the SPN we consider the two basic ones: the simple *global* model and the *coarse grained* model. In a previous work [17], the first approach has been used; in this implementation a master process is responsible of the main execution of the genetic algorithm and exploits the availability of different processors by allocating a slave process on each of them. Each slave is required to execute the evaluation function for some individual of the current population on the basis of its availability. In this paper the coarse-grained model will be studied and compared with the previous one. The coarse-grained model divides the population into smaller subpopulations, termed *demes*, constituting a given number of islands. A standard GA is executed on each island and is responsible for initializing, evaluating and evolving its own individuals. Furthermore, the standard GA is enforced by a migration operator, which periodically involves the transfer of individuals among the different subpopulations.

To perform the analysis of the solution space, a GA needs the representation of the problem solutions as basic individuals of its population, which are

called genomes. During the execution of the algorithm new individuals will be generated by means of the mutation and crossover operators. To encode the feasible solutions of the SPN as binary genomes, we adopted the following representation: for each instance of the problem we define the genome as a binary array whose length corresponds to the dimension of the set $V - Z$, i.e. the set of all the nodes which are potential candidates for belonging to a given solution. The value of the $i_{th}$ bit represents if the correspondent node in the set $V - Z$ should be considered as complementary node to generate a tree which connects the multicast $Z$ nodes. To follow the genome indication of including the correspondent nodes in a solution tree we map each genetic individual in a new instance of the problem where the original $Z$ nodes are extended with the nodes coded by the genes. This new instance of the problem is solved using the K-SPH or ADH heuristics, and the solution is pruned with regard to the original multicast set. K-SPH is a $O(pn^2)$ algorithm which is capable of isolating good solutions, although it uses only nodes which are along the shortest paths between the multicast nodes. ADH is a $O(n^3)$ algorithm which is capable of determining better solutions because it considers all the nodes in the network. To obtain a trade-off between execution time and competitiveness we adopted alternatively both heuristics in order to exploit their different features. The fitness value is straightforwardly calculated as the inverse of the tree cost, thus to restrict the range of the fitness function to the interval $(0, 1]$. The adoption of the heuristic methods on individuals provided by the GA represents a local search technique which complements the global search capability naturally owned by the GA. This way a hybrid optimization algorithm is obtained. Furthermore, considering that the evaluation process described above determines the minimal set of genes which forms the current solution, we introduce a further mutation mechanism which replaces the original genome with the solution achieved by ADH. This process could be viewed as the implementation in the GA of the procedure that leads to a Genetically Modified Organism in nature. This technique introduces the advantage of faster convergence towards the optimal solution.

### 3.1 Grid Based Implementation

The parallel implementation has been carried out on a cluster, simulating a grid environment of distributed machines in order to exploit further computing resources. The experimental cluster is composed by forty workstations managed by the Globus 3.0 toolkit [19]. The communication activities are carried out using MPIGH-G2 [16], the grid-enabled implementation of the Message Passing Interface (MPI). All nodes are equipped with an Intel Pentium 4 1.7 GHz CPU, 256 Mbytes of RAM, four 100Mbps Ethernet cards and they are managed by the version 7.2 of the Red Hat Linux distribution. A redundant degree of connectivity is achieved by means of eight 100Mbps Ethernet switches. The software system exploits the facilities provided by the GAlib, a C++ Library of Genetic Algorithm Components [18]. The master-slave paradigm has been adopted to implement the parallel version, embedding the MPICH-G2 primitives and GAlib object-oriented classes.

## 4 Experimental Results

In this section we discuss the experimental results obtained on three different test sets of sample graphs, taken respectively from the public SteinLib library [13], the BRITE [14] topology generator, and the Mercator project [12]. On this experimental testbed, we execute the two models of the parallel Genetic Algorithm, the classical heuristics SPH, DNH, K-SPH, and ADH, and the stirring heuristic. The GA parameters for the two different parallel implementations are shown in Table 1.

**Table 1.** GAs parameters

| Global | Coarse Grained |
|---|---|
| number of generations = 25 | number of generations = 4 |
| population size = 120 | population size = 50 |
| crossover probability $PC = 0.7$ | crossover probability $PC = 0.7$ |
| mutation probability $PM = 0.001$ | mutation probability $PM = 0.001$ |
| | number of demes = 5 |

We maintain these values constant for all the executions in order to compare all problems on a homogeneous basis. Furthermore, the local search technique and the additional mutation mechanism described in the above section are implemented. Figure 1 shows the fitness function values distributions for the *global* and *coarse grained* parallel implementations. The charts plot the score obtained by the first 1250 individuals of the *global* model and by all the individuals of the *coarse grained* model; it is possible to note a faster convergence of the *coarse grained* model. Moreover, the speedup achieved by the *coarse grained* model is significantly better than that of the *global* model, because the first one has few synchronization points due to its nature, and thus it can be run on a wide-area grid environment without affecting its performances.

The better performances obtained by the *coarse grained* model, together with its better speedup, motivate the adoption of it in all the following experiments. The first test set is a subset of the SteinLib library, a public collection of Steiner tree problems in graphs with different characteristics, taken from VLSI applications, genetic contexts, computer networks applications, etc. More specifically we adopt the subset constituted by *Beasley's* series C, D, E, formerly known as the OR-library, which are random-weights graphs with sizes ranging from 500 to 2,000 nodes. The connection degree is relatively high, ranging from 0.1% up to 10%. The networks in this sample do not present any similarity with the Internet like topologies [17]. However, we adopted it as test for our parallel implementation of GA, because it represents a commonly accepted assessment term since the optimal solutions are known. Figure 2 shows the cumulative cost competitiveness of parallel GA and the classical heuristics over the above graphs. The competitiveness is determined as the ratio between the costs of trees produced by heuristics and the optimal ones. From the comparison of the solutions obtained by the GA with the optimal values, it can be observed that in about 80%

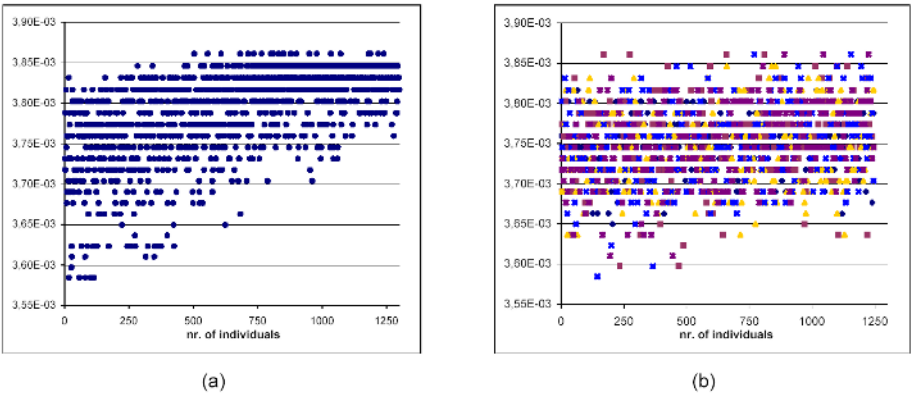


**Fig. 1.** Fitness function values distribution for the global (a) and coarse grained (b) parallel implementations

of the cases both the GAs are able to determine the optimal solution, and for the 90% of the instances the obtained solution is at most 1% larger than the optimal value.

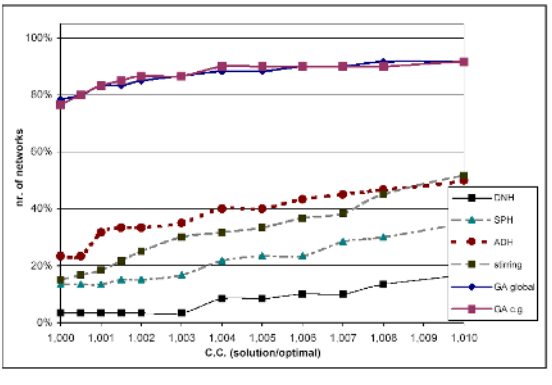


**Fig. 2.** Cumulative Cost Competitiveness on C, D, E SteinLib nets.

The following set of experiments is devoted to investigate the graphs with topological features similar to the Internet graphs. BRITE (Boston university Representative Internet Topology gEnerator) was developed to investigate the growth of large computer networks, and to compare several topology generation models. In our experiments, we tested several networks ($\sim$ 400) with homogeneous topological characteristics and sizes ranging from 1,000 to 2,000 nodes. Figure 3 (a) shows the cumulative cost competitiveness curves for a test set composed of fifty networks, each of them with 2,000 nodes. In this and in the following experiments, the competitiveness is determined as the ratio between the costs of trees produced by heuristics and the best-known sub-optimal solu-

tion. As it can be clearly observed, GA finds the best-known solutions on almost all the instances, thus confirming its effectiveness to be used as a validation term for the other heuristics.

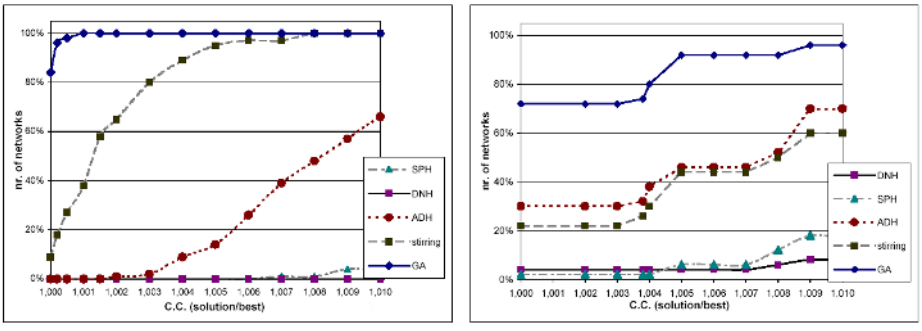


**Fig. 3.** Cumulative Cost Competitiveness on (a) Brite nets and (b) Mercator subnets.

In the last experiment, the test set is created starting from the real Internet data description produced by the Mercator project. This project has produced a real Internet snapshot, by merging an enormous amount of measurements taken over the time and gathered into a central database. The resulting network, obtained in November 1999, includes more than 280,000 nodes and nearly 450,000 edges, with a connection degree lower than $0,001\%$. To set up our experiment, we extracted 50 subnetworks of $2,000$ node size from the original map, starting from a randomly selected node and repeatedly including its neighbors. Differently from the previous example, since the Mercator data do not provide any cost associated to the edges, the metric is hop count based. The analysis of the cumulative cost competitiveness curves, shown in figure 3 (b), reveals the parallel GA effectiveness since the best-known solutions are found on about 70% of the instances. The relatively worse performances of all algorithms and the particular shape of the curves in this case are mainly due to the hop count metric, which leads to a higher quantization of the input data.

## 5 Conclusions

In this work we proposed the adoption of a parallel implementation of genetic algorithm and local search methodologies to obtain near-optimal solution to the Steiner Problem in Networks for large graphs with topological features similar to the Internet ones. The results have shown that our implementations achieved high competitiveness in all the experimented test sets, differentiated for topological characteristics. In most of the well known examples of the SteinLib library we found the optimal solutions. On the sample networks generated by the Brite tool or extracted from the Mercator graph, which simulate the Internet structure with the best accuracy, we almost always obtained the best calculated

sub-optimal solutions, thus achieving a useful result for the comparison of the competitiveness of the polynomial and deterministic heuristics. As regards the future directions, we are currently developing more sophisticated parallel models, with the aim of further improving the GA performances and optimizing the total execution times.

## References


1. J. Kruskal, On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem, Proc. Amer. Math. Soc., vol. 7, pp. 48 - 50, 1956.
2. R. M. Karp, Reducibility among Combinatorial Problems, in R. E. Miller, J. W. Thatcher, Complexity of Computer Computations, Plenum Press, New York, pp.85-103, 1972.
3. H. Takahashi, A Matsuyama, An approximate solution for the Steiner problem in graphs, Math. Japan, 1980, pp. 573-577.
4. V. J. Rayward-Smith, The computation of nearly minimal Steiner trees in graphs, Int. Math. Ed. Sci. Tech. 14, 1983, pp. 15-23.
5. P. Winter, Steiner problem in networks: a survey, Networks, 17, 1987, pp. 129-167.
6. D. E. Goldberg, Genetic algorithm in Search, Optimization, and Machine Learning, Reading Ma: Addison Wesley 1989.
7. K. A. Dowsland, Hill-climbing, Simulated Annealing and the Steiner Problem in Graphs, Engineering Optimisation, 17, 1991, pp. 91-107.
8. E. Cantu-Paz, A summary of research on parallel genetic algorithms, Illinois GALab, Univ. Illinois Urbana-Champaign, Urbana, IL, Tech. Rep. 950076, July 1995.
9. H. Esbensen Computing Near-Optimal Solutions to the Steiner Problem in a Graph Using a Genetic Algorithm, Networks: An International Journal 26, 1995.
10. M. Gendreau, J. F. Larochelle, B. Sanso A Tabu Search Heuristic for the Steiner Tree Problem Networks 34: 162?172, 1999 1999 John Wiley & Sons, Inc.
11. G. Di Fatta, G. Lo Re, Efficient tree construction for the multicast problem, Special issue of the Journal of the Brazilian Telecommunications Society, 1999.
12. R. Govindan, H. Tangmunarunkit, Heuristics for Internet Map Discovery, Proc IEEE Infocom 2000, Tel Aviv, Israel, www.isi.edu/scan/mercator/mercator.html.
13. S. Voss, A. Martin, T. Koch, SteinLib Testdata Library, February 2001, elib.zib.de/steinlib/steinlib.php.
14. A. Medina, A. Lakhina, I. Matta, J. Byers, BRITE Topology Generator, April 2001 cs-pub.bu.edu/brite.
15. G.Folino, C. Pizzuti, G. Spezzano, Parallel Hybrid Method for SAT That Couples Genetic Algorithms and Local Search, IEEE Transactions on Evolutionary Computation, Vol. 5, No. 4, pp. 323-334, August 2001.
16. N. Karonis, B. Toonen, I. Foster, MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface, Journal of Parallel and Distributed Computing (JPDC), Vol. 63, No. 5, pp. 551-563, May 2003.
17. G. Di Fatta, G. Lo Presti, G. Lo Re, A Parallel Genetic Algorithm for the Steiner Problem in Networks, Proc. of the 15th IASTED Int. Conference on Parallel and Distributed Computing and Systems, Marina del Rey (CA), USA, November 2003.
18. GAlib: a C++ Library of Genetic Algorithm Components, http://lancet.mit.edu/ga/.
19. T. Sandholm, J. Gawor, Globus Toolkit 3 Core – A Grid Service Container Framework, http://www-unix.globus.org/toolkit/3.0/ogsa/docs/gt3core.pdf.