A parallel genetic algorithm for the Steiner Problem in Networks

A PARALLEL GENETIC ALGORITHM FOR THE STEINER PROBLEM IN NETWORKS

Giuseppe Di Fatta, Giuseppe Lo Presti, Giuseppe Lo Re
ICAR – Istituto di Calcolo e Reti ad Alte Prestazioni, Consiglio Nazionale delle Ricerche
Viale delle Scienze, 90128 Palermo
Italy
{difatta, lopresti, lore}@pa.icar.cnr.it

Abstract
This paper presents a parallel genetic algorithm to the Steiner Problem in Networks. 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 characteristics 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 build a comparison term for validating deterministic and computationally inexpensive algorithms which can be used in practical engineering applications, such as the multicast transmission in the Internet. On the other hand, the large dimensions of our sample networks require the adoption of a parallel implementation of the Steiner GA, which is able to deal with such large problem instances.

Key Words
Steiner Problem, Parallel Genetic Algorithm, Internet topology.

1. Introduction

The Steiner Problem in Networks (SPN) [22], is a classic combinatorial optimization problem which in its general case decision version has been shown [10] NP-complete. Its applications cover many different scientific and technological fields such, for instance, the VLSI and pipeline design, the Internet multicast routing, the telephone network design, etc. Given the importance that the problem entails in many scientific fields, 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 [11], [16], [19]. Mathematical proofs constrain the solutions determined by these heuristics to the optimal solution, binding them by some multiplicative factors. This property allows their adoption for many practical applications. However, it remains a scientific challenge to determine the optimal solutions for those small instances treatable by exhaustive algorithms, and the best sub-optimal solutions in the other cases.

As previously mentioned, 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 and effective heuristics capable of determining, in a very short time, sub-optimal solutions that however may represent good approximations. Many of such methods have been proposed in the last years, with the further constraint to produce deterministically the solutions. To validate the effectiveness of the proposed 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 meta-heuristics such as Genetic Algorithms [5], tabu-search [7], and Simulated Annealing [4] 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 distribute coordination among several independent entities. However, the good performances produced by these evolutionary methods suggested us the idea to exploit their results as comparison term. The approach that better than each other has been evaluated suitable for exploiting the coarse grain parallelism available in our laboratory was 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 [21],

which represents a commonly accepted comparison 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 genetic algorithm, and section IV discusses some topological metrics used to evaluate the algorithm performances. Finally, section V describes the experimental results, and section VI concludes this work and discusses future directions.

2. The Steiner Problem in Networks

Formally, the Steiner Tree Problem in Network can be formulated as follows.

Definition 1. *Let $G = (V, E)$ be an undirected graph, $w : E \rightarrow R^+$ 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) [22], the Minimum or Shortest Path Heuristic (MPH or SPH) [19], the K-Shortest Path Heuristic (K-SPH) [12], the Average Distance Heuristic (ADH) [17], and the Stirring heuristic [2]. Some of the heuristics used as comparison terms and exploited in our algorithm are briefly described here. The DNH builds the distance network K_z induced by Z, and constructs the minimum spanning tree (MST) on the network K_z. It replaces the virtual links with the real paths (the nodes and links of the initial network), thus obtaining a subgraph of the initial network, G_z. It then computes the MST on G_z and finally prunes all the S-vertices of degree one. The MPH builds a subtree of G in an incremental fashion: it starts off by selecting an arbitrary node among the terminal nodes (typically the source node) and then progressively adds the terminal node nearest to the tree, including the nodes and edges of the connecting path. The K-SPH is an improvement of the MPH algorithm. It 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. ADH 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. Furthermore, many algorithms capable of identifying the optimal solution tree have been proposed in the literature. All of them are characterized by an exponential complexity. Among these, the Spanning Tree Enumeration Algorithm [22], has $O(p^2\, 2^{n-p} + n^3)$ complexity, and the Dynamic Programming Algorithm with $O(n^3 + n^2\, 2^{p-1} + n\, 3^{p-1})$ complexity. However, their exponential nature does not allow their adoption in any practical field.

3. The Parallel Genetic Algorithm

In this paper we will not describe the basic genetic algorithms, and for their description we remand the interested reader to [8]. To solve a problem using a genetic algorithm, it is necessary to perform a mapping of the problem elements into the basic components of the GA. 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. New generated individuals should own the basic property to still represent feasible solutions. To encode the feasible solutions of the SPN as binary genomes, we adopted the following representation. For each particular 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 may be considered potential candidates for belonging to a given solution. The value of the *i-th* bit represents if the correspondent node i in the set $V - Z$ should be considered as complementary node to generate a tree which connects the 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 heuristic [12] and the solution pruned with regard to the original multicast set. 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]. Among the heuristics presented in the previous section, we chose the K-SPH as evaluation criterion, since its performances represent a valuable trade-off between execution time and solution competitiveness.

Genetic Algorithms are naturally suited to be implemented on a parallel architecture. Surveys on parallel GAs can be found on [1] and [20]. Several approaches to parallel implementations of GAs have been proposed. In this paper we will consider a simple *global* model. In this approach a master process is responsible of the main execution of the genetic algorithm and exploit the availability of different processors by allocating a slave process on each of them. Each slave will be required

to execute the evaluation function for some individuals of the current population on the basis of its availability. This way, the master process has only to deal with the genomes population, while the slave processes have to deal with the more expensive computation of the evaluation function. This allows a good scalability of the algorithm with respect to the number of available slaves.

4. Topological metrics

Before evaluating the heuristics performances over the sample graphs, we firstly need to establish some criteria to objectively measure the topological characteristics of the graphs. This is in order to compare the experimental graphs with the networks coming from the application world, such as the case of the Internet for multicast transmission applications. Namely, the performances of the heuristics can heavily depend from global network parameters such as the maximum node degree and the connection degree, that in turn depends from the type of the examined graph. In this section we describe some topological parameters used to characterize computer networks, including the node degree, the node rank and the degree frequency distribution (see table 1). Recently, it has been shown [6] that, despite their apparent randomness, current Internet topologies exhibit power laws of the form $y \propto x^{\alpha}$, where α is a constant, between some topological parameters that are examined in table 1.

Metric	*Description*
d_c	Connection degree, i.e. fraction of edges with respect to a same-size fully connected network. It is obtained as $d_c = \frac{2m}{n(n-1)}$.
d_v	Node degree, i.e. the number of outgoing edges from the node v.
r_v	Node rank, i.e. its index ordering all nodes in decreasing degree. It follows from the definition that if $\exists$ two nodes v and w such that $d_v < d_w$, then $r_v > r_w$.
f_d	Degree frequency, i.e. the number of nodes with degree d.

Table 1 - Graph and node parameters used to characterize network topologies.

With the above definitions, we are interested particularly on the following well known power-laws, as they can help in characterizing graphs typologies from each other:
The degree of a node is proportional to the rank of the node to the power of a constant R:

$$d_v \propto r_v^R$$

The frequency of a degree is proportional to the degree to the power of a constant O:

$$f_d \propto d^O$$

In a previous work [3] we found that in order to characterize a network and evaluate its matching degree with a real network structure such the Internet, we can take into account the global network connection degree d_c and the regression coefficients R^2, which determine the fitness degree between the node parameters and the above power laws. Namely, if a topology exhibits an Internet-like structure the d_c parameter is always very low (under 0.01) and the two R^2 coefficients are relatively high, both for the first and for the second law; on the other hand, if the topology does not exhibit any particular structure, the above distributions can assume very different shapes (e.g. gaussian), with meaningless R^2 values and the d_c parameter ranging in the whole interval [0, 1].

5. Experimental results

The parallel implementation has been carried out on forty workstations arranged as a grid cluster managed by the MPI system [15]. All nodes present the same hardware and software configuration. Each of them is equipped with an Intel Pentium 4 processor with a clock frequency of 1.5 GHz, 256 Mbytes of RAM, four 100Mbps Ethernet cards and managed by the Red Hat Linux 7.2 distribution. A redundant degree of connectivity is achieved by means of eight 100Mbps Ethernet switches.
In this section we discuss the experimental results obtained on three different test sets of sample graphs, taken respectively from the public SteinLib library, the BRITE topology generator, and the Mercator project. All the test sets are classified according to the topological indices described in the previous section. On this experimental testbed, we execute the parallel Genetic Algorithm, the classical heuristics SPH, DNH, K-SPH, and ADH, and the stirring heuristic. The GA parameters are set as follows:

- *population size = 120 individuals (three times the processors number),*
- *number of generations = 30,*
- *probability of mutation = 0.05,*
- *probability of crossover = 0.8.*

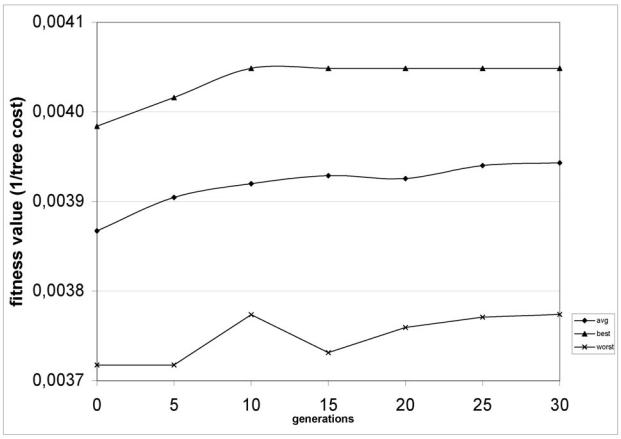

Fig. 1 - Typical fitness evolution vs. number of generations.

We maintain these values constant for all the executions in order to compare all problems on a homogeneous basis. Furthermore, since a preliminary observation (see fig. 1) revealed a fast convergence of the algorithm with regard

to the number of generations, we chose for this parameter a relatively small value, thus to optimize the execution times.

The first test set is a subset of the SteinLib library [21], 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%. This network sample does not exhibit any power law as regards the degree and rank distribution, which means that its graphs do not present any similarity with the Internet-like topologies. However we adopted it as test for our parallel implementation of GA, because it represents a commonly accepted comparison term since the optimal solutions are known.

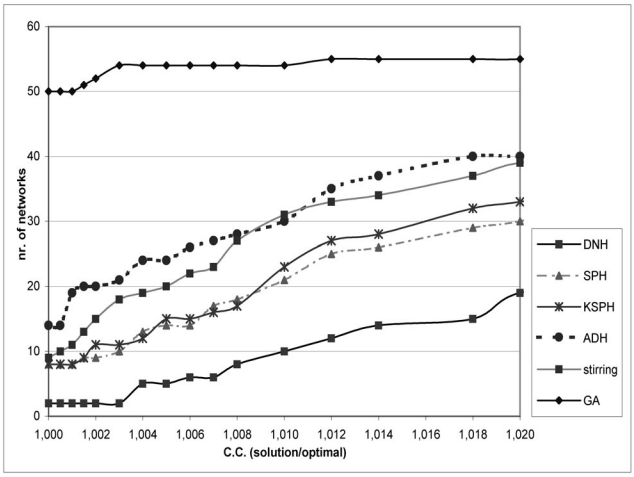

Fig. 2 - Cumulative cost competitiveness on C,D,E SteinLib nets.

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 on 50 over 60 cases the GA is able to determine the optimal solution, and for 55 instances the obtained solution is at most 1% larger than the optimal value.

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 [13], and to compare several topology generation models. The key characteristic of this generator is the *incremental growth* (the network generation goes on in an incremental fashion) and the *preferential connectivity* (the probability that a new node is connected to a randomly selected target node is positively correlated to the degree of the target), used during the generation process; its authors claim that these are the primary reasons for power-laws on the Internet, since the generated topologies exhibit the power-laws with a very high correlation. However, it should be underlined that BRITE adopts an incremental growing strategy, rather than a hierarchical mode; as noted in [14], although it is commonly accepted the idea of a hierarchical Internet, experimental tests have proved that an incremental generator, based on the nodes degree, fits the real networks better than a hierarchy based generator. In our experiments, we tested several networks (~ 400) with homogeneous topological characteristics and sizes ranging from 1,000 to 5,000 nodes.

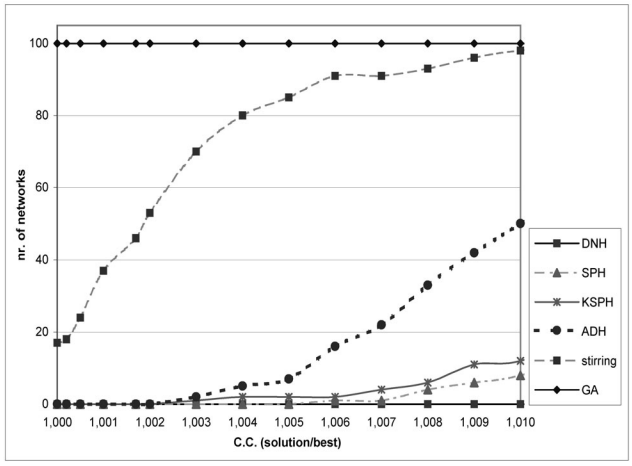

Fig. 3 - Cumulative cost competitiveness on Brite nets.

Figure 3 shows the cumulative cost competitiveness curves for a test set composed of one hundred networks, each of them with 1,000 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 solution. As it can be clearly observed, GA finds the best-known solutions on all the instances, thus confirming its effectiveness to be used as a comparison term for the other heuristics.

In the last experiment, the test set is created starting from the real Internet data description produced by the Mercator project [9].

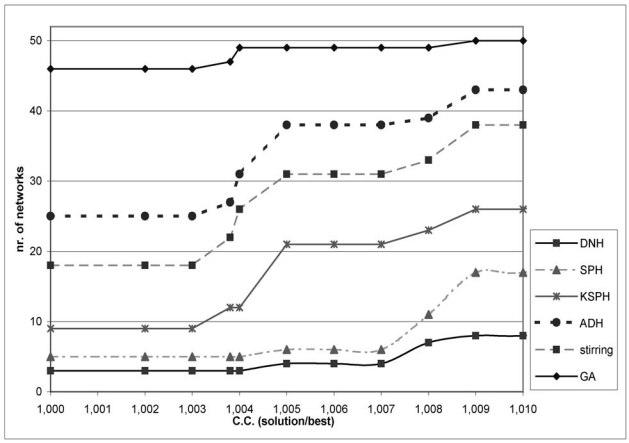

Fig. 4 - Cumulative cost competitiveness on Mercator subnets.

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%.
The analysis of the cumulative cost competitiveness curves, shown in figure 4, reveals the parallel GA effectiveness since the best-known solutions are found on 45 instances out of 50.

6. Conclusion

In this work we proposed the adoption of a parallel implementation of genetic algorithm 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 genetic models, such as a multi population model, with the aim of further improving the GA performances and dealing with larger problem instances.

References

[1] E. Cant-Paz, A summary of research on parallel genetic algorithms, *Technical Report 950076*, Illinois Genetic Algorithm Lab., Univ. Illinois Urbana-Champaign, Urbana, IL, July 1995.
[2] G. Di Fatta, G. Lo Re, Efficient tree construction for the multicast problem, Special issue of the *Journal of the Brazilian Telecommunications Society*, 1999.
[3] G. Di Fatta, G. Lo Presti, G. Lo Re, Computer Network Topologies: Models and Generation Tools, *CE.R.E. Technical Report 5*, July 2001.
[4] K. A. Dowsland, Hill-climbing, Simulated Annealing and the Steiner Problem in Graphs, *Engineering Optimisation*, 17, 1991, pp. 91-107.
[5] H. Esbensen, Computing Near-Optimal Solutions to the Steiner Problem in a Graph Using a Genetic Algorithm, *Networks: An International Journal*, *26*, 1995.
[6] M. Faloutsos, P. Faloutsos, C. Faloutsos, On Power-Law Relationships of the Internet Topology, *ACM SIGCOMM*, 1999.
[7] M. Gendreau, J. F. Larochelle, B. Sanso, A Tabu Search Heuristic for the Steiner Tree Problem, *Networks, 34*, 1999, 162-172.
[8] D. E. Goldberg, *Genetic algorithm in Search, Optimization, and Machine Learning* (Reading, MA: Addison Wesley, 1989).
[9] R. Govindan, H. Tangmunarunkit, Heuristics for Internet Map Discovery, *Proc IEEE Infocom 2000*, Tel Aviv, Israel.
[10] R. M. Karp, Reducibility among Combinatorial Problems, in R. E. Miller, J. W. Thatcher (Eds.), *Complexity of Computer Computations* (Plenum Press, New York, 1972, 85-103).
[11] L. Kou, G. Markowsky, L. Berman, A fast algorithm for Steiner trees, *Acta Inform., 15*, 1981, 141-145.
[12] J. Kruskal, On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem, *Proc. Amer. Math. Soc., 7*, 1956, 48-50.
[13] A. Medina, A. Lakhina, I. Matta, J. Byers, BRITE Universal Topology Generator, 2001, cs-pub.bu.edu/brite.
[14] A. Medina, I. Matta, J. Byers, On the Origin of Power Laws in Internet Topologies, *ACM SIGCOMM 2000, 30*(2), April 2000.
[15] Message Passing Interface Forum. MPI: A new message-passing interface standard (version 1.1), *Technical Report*, University of Tennessee, 1995.
[16] V. J. Rayward-Smith, The computation of nearly minimal Steiner trees in graphs, *Int. Math. Ed. Sci. Tech. 14*, 1983, 15-23.
[17] V. J. Rayward-Smith, A. Clare, On Finding Steiner Vertices, *Networks, 16*, 1986, 283-294.
[18] H. Tangmunarunkit, R. Govindan, S. Jamin, et al., Network Topologies, Power Laws, and Hierarchy, *SIGCOMM 2001*, June 2001.
[19] H. Takahashi, A Matsuyama, An approximate solution for the Steiner problem in graphs, *Math. Japan*, 1980, 573-577.
[20] M. Tomassini, Parallel and distributed evolutionary algorithms: A review, in K. Miettinen, M. Mkel, P. Neittaanmki, J. Periaux, *Evolutionary Algorithms in Engineering and Computer Science* (Eds. New York: Wiley, 1999, 113-133).
[21] S. Voss, A. Martin, T. Koch, *SteinLib Testdata Library*, February 2001, elib.zib.de/steinlib/steinlib.php.
[22] P. Winter, Steiner problem in networks: a survey, *Networks, 17*, 1987, 129-167.