Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

A Parallel Hybrid Genetic Algorithm for SPN

Giuseppe Lo Presti, Giuseppe Lo Re,
Pietro Storniolo, Alfonso Urso

RT-ICAR-PA-03-05 **dicembre 2003**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR) –
Sede di Cosenza, Via P. Bucci 41C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sezione di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*
– Sezione di Palermo, Viale delle Scienze, 90128 Palermo, URL: *www.pa.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

A Parallel Hybrid Genetic Algorithm for SPN

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

Rapporto Tecnico N.: **RT-ICAR-PA-03-05**

Data: **dicembre 2003**

---

[1] Istituto di Calcolo e Reti ad Alte Prestazioni, ICAR-CNR, Sezione di Palermo Viale delle Scienze edificio 11 90128 Palermo
[2] Università degli Studi di Palermo Dipartimento di Ingegneria Informatica Viale delle Scienze 90128 Palermo

A Parallel Hybrid Genetic Algorithm for SPN

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

[1] Dinfo - Dipartimento di Ingegneria Informatica
Università di Palermo - Palermo, Italy.
e-mail: {lopresti}@pa.icar.cnr.it
[2] ICAR - Istituto di Calcolo e Reti ad Alte Prestazioni
C.N.R. - Consiglio Nazionale delle Ricerche Palermo, Italy
e-mail: {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 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. 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 organism 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) [27], is a classic combinatorial optimization problem which, in its general case decision version, has been demonstrated [11] 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 [24], [12], [21]. 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 deterministically produce the solutions. In order 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 metaheuristics such as Genetic Algorithms [6], tabu-search [8], and Simulated Annealing [5] 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 suggests 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 [26], 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 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, $w : E \rightarrow R^+$ 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) [27], the Minimum or Shortest Path Heuristic (MPH or SPH) [24], the K-Shortest Path Heuristic (K-SPH) [13], the Average Distance Heuristic (ADH) [21], 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 [27], has $O(p^2 2^{(n-p)} + n^3)$ complexity, and the Dynamic Programming Algorithm with $O(n^3 + n^2(2^{p-1} - p - 1) + n(3^{p-1} - 2^p + 3)/2)$ complexity. However, their exponential nature does not allow their adoption in any practical field.

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 GA is not considered a mathematically guided algorithm, as it does not require the computation of derivatives of the optimization function. The optimum is obtained by an evolution from generation to generation. This evolutionary strategy can be considered a stochastic, discrete event and nonlinear process in which the obtained optimum is the end product containing the best elements of previous generations where the attributes of better adapted individuals are carried forward into the following generation. The main advantage of the GA is its capability of achieving global optimization solution even for nonlinear, high-dimensional, multimodal and discontinuous problems [9].
Genetic Algorithms are naturally suited to be implemented on a parallel architecture . A survey on parallel GAs can be found in [1]. Several approaches to parallel implementations of GAs have been proposed ([17], [25]). 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 [4], 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 will be 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. 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 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. 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 $(0, 1]$. K-SPH is a $O(n^2)$ 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 decided to adopt alternatively both heuristics in order to exploit their different features. 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 realized on a grid cluster, with forty workstations managed by the Globus 3.0 toolkit [16]. The communication activities are carried out using MPIGH-G2 [18], the grid-enabled implementation of the Message Passing Interface (MPI)[19]. All nodes present the same hardware and software configuration. Each of them is equipped with an Intel Pentium IV processor with a clock frequency of 1.7 GHz, 256 Mbytes of RAM, four 100Mbps Ethernet cards and 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 [20]. 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 [26], the BRITE [14] topology generator, and the Mercator project [10]. 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.

We maintain these values constant for all the executions in order to compare all problems on a homogeneous basis. Furthermore, the local search technique

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

Table 1. *GAs parameters*

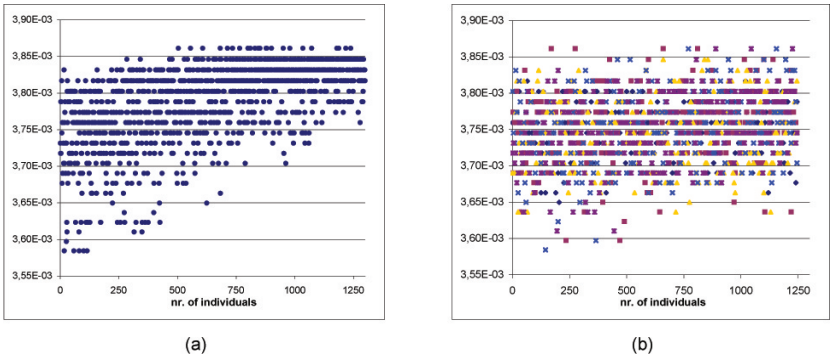

Fig. 1. *Fitness function values distribution for the global (a) and coarse grained (b) parallel implementations*

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 execution time of the *coarse grained* model is significantly lower than that one of the *global* model.

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 [4]. 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.

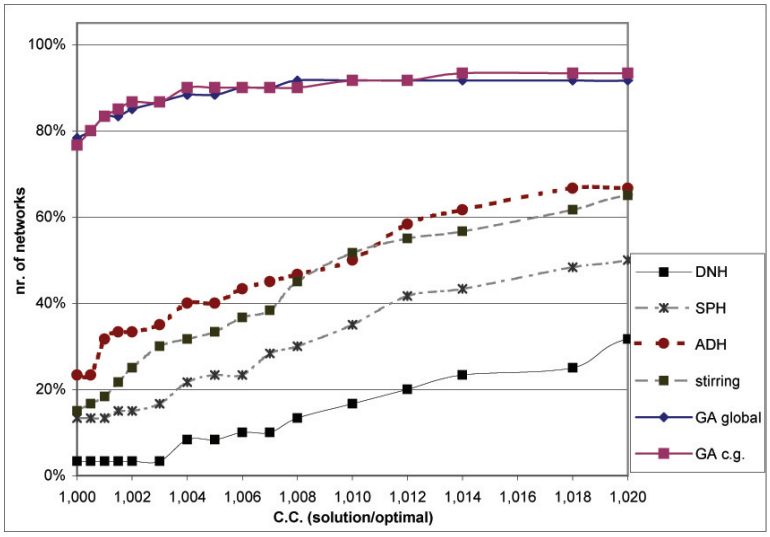

Fig. 2. *Cumulative Cost Competitiveness on C, D, E SteinLib nets.*

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 in about 80% 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.

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. The key characteristic of this generator is the incremental growth and the preferential connectivity used during the generation process. However, it should be underlined that BRITE adopts an incremental growing strategy, rather than a hierarchical model; as noted in [15], 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 ($\sim$ 400) with homogeneous topological characteristics and sizes ranging from 1,000 to 2,000 nodes. Figure 3 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 solution. As it can be clearly observed, GA finds the best-known

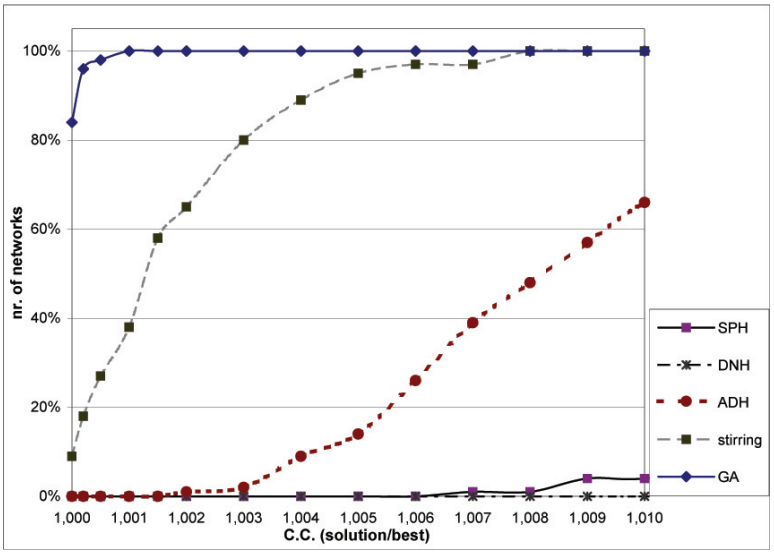

Fig. 3. *Cumulative Cost Competitiveness on Brite nets.*

solutions on almost 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. 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 costs associated to the edges, the metric is hop count based. 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 about 70% of the instances.

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 topo-

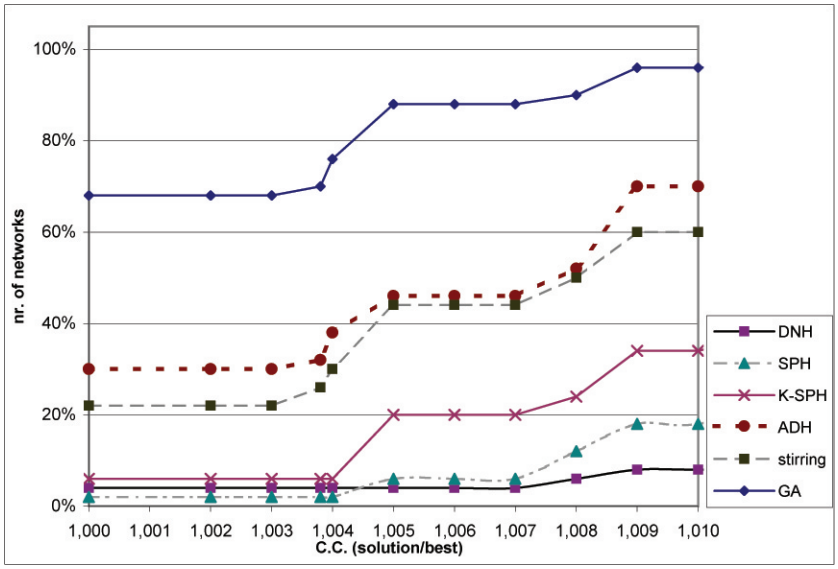

Fig. 4. *Cumulative Cost Competitiveness on Mercator subnets.*

logical 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. E. Cantu-Paz, A summary of research on parallel genetic algorithms, Illinois Genetic Algorithm Lab., Univ. Illinois Urbana-Champaign,Urbana, IL, Tech. Rep. 950076, 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, July 2001
4. G. Di Fatta, G. Lo Presti, G. Lo Re, A Parallel Genetic Algorithm for the Steiner Problem in Networks, Proc. of the Fifteenth IASTED International Conference on Parallel and Distributed Computing and Systems, Marina del Rey, CA, USA, November 3-5 2003.
5. K. A. Dowsland, Hill-climbing, Simulated Annealing and the Steiner Problem in Graphs, Engineering Optimisation, 17, 1991, pp. 91-107.

6. H. Esbensen Computing Near-Optimal Solutions to the Steiner Problem in a Graph Using a Genetic Algorithm, Networks: An International Journal 26, 1995.
7. M. Faloutsos, P. Faloutsos, C. Faloutsos, On Power-Law Relationships of the Internet Topology, ACM SIGCOMM, 1999.
8. 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.
9. D. E. Goldberg, Genetic algorithm in Search, Optimization, and Machine Learning, Reading Ma: Addison Wesley 1989.
10. R. Govindan, H. Tangmunarunkit, Heuristics for Internet Map Discovery, Proc IEEE Infocom 2000, Tel Aviv, Israel, www.isi.edu/scan/mercator/mercator.html.
11. R. M. Karp, Reducibility among Combinatorial Problems" In R. E. Miller, J. W. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, New York, pp. 85-103, 1972.
12. L. Kou, G. Markowsky, L. Berman, A fast algorithm for Steiner trees, Acta Inform., 15, 1981, pp. 141-145.
13. 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.
14. A. Medina, A. Lakhina, I. Matta, J. Byers, BRITE Universal Topology Generator, January 2000 - April 2001 cs-pub.bu.edu/brite.
15. A. Medina, I. Matta, J. Byers, On the Origin of Power Laws in Internet Topologies, ACM SIGCOMM 2000 30, 2, April 2000.
16. 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.
17. 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.
18. 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.
19. Message Passing Interface Forum. MPI: A new message-passing interface standard (version 1.1). Technical report, University of Tennessee, 1995.
20. GAlib: a C++ Library of Genetic Algorithm Components, http://lancet.mit.edu/ga/.
21. V. J. Rayward-Smith, The computation of nearly minimal Steiner trees in graphs, Int. Math. Ed. Sci. Tech. 14, 1983, pp. 15-23.
22. V.J. Rayward-Smith, A. Clare, On Finding Steiner Vertices, Networks vol. 16 (1986) pp. 283-294
23. H. Tangmunarunkit, R. Govindan, S. Jamin, et al., Network Topologies, Power Laws, and Hierarchy, SIGCOMM 2001 poster, June 2001.
24. H. Takahashi, A Matsuyama, An approximate solution for the Steiner problem in graphs, Math. Japan, 1980, pp. 573-577.
25. M.Tomassini, Parallel and distributed evolutionary algorithms: A review, in Evolutionary Algorithms in Engineering and Computer Science K. Miettinen, M. Mkel, P. Neittaanmki, and J. Periaux, Eds. New York: Wiley, 1999, pp.113133.
26. S. Voss, A. Martin, T. Koch, SteinLib Testdata Library, February 2001, elib.zib.de/steinlib/steinlib.php.
27. P. Winter, Steiner problem in networks: a survey, Networks, 17, 1987, pp. 129-167.