# Set Covering and Network Optimization: Dynamic and Approximation Algorithms

Orestis A. Telelis**

Department of Informatics and Telecommunications
National and Kapodistrian University of Athens, Hellas

**Abstract.** In this short note we summarize our results on development and analysis of approximation and dynamic algorithms for set covering and network optimization problems. The results include probabilistic analysis of set covering algorithms, development and analysis of dynamic algorithms for graph optimization problems, game-theoretic analysis of a file-sharing network model, and approximation algorithms for Steiner tree/forest network optimization under uncertainty.

## 1 Introduction

Set covering and network optimization problems lay in the heart of the reserach field known as Combinatorial Optimization. In particular, set covering is a well known special case of the general integer linear programming problem, while it also serves as a general model for network optimization problems. Network optimization has received special attention mainly due to the vast evolution of telecommunication and computer networks, along with the variety of applications built upon them.

Many of the problems found in the aforementioned broad categories are well known to be $NP$-hard. The natural way to cope with the current lack of computational resources for finding an optimum solution is to try to develop polynomial-time approximation algorithms (see [1]). Network models used in modern applications ranging from telecommunications to VLSI design are recognized to be highly dynamic structures, changing at high rates. The need for efficient response (which essentially involves solution of a problem with respect to the current structure of the network) to such dynamic changes has led to the development of the field of *dynamic graph algorithms* [2]. Network optimization under uncertainty is a conceptually related line of research, that has seen extended progress in the last few years, and aims at preplanning cost-robust network designs in the presence of uncertainty of requirements.

Another characteristic of networks that has recently caught attention of research in recent years is their social nature. In many important applications a network is formed by a set of users/agents that want to maximize their individual profit by their participation to it. Game-theoretic network analysis aims

** Dissertation Advisor: Vassilis Zissimopoulos

at analyzing the incentives of selfish agents for participating in a network, and their impact on the network's efficiency.

Our work is concerned with all the aforementioned matters. In this short note we give a summary of the results described in the dissertation [3], and have appeared in [4–11]. The rest of this article is structured as follows. In section 2 we summarize our results regarding probabilistic analysis of approximation algorithms for set covering. In section 3 we describe dynamic algorithms for graph optimization problems on both, undirected and directed graphs. In section 4.1 we describe our results on game-theoretic analysis of a file-sharing network optimization problem. Section 4.2 includes our results regarding approximation algorithms for Steiner tree/forest problems in the presence of uncertainty of input.

## 2 Random Set Covering

A well known heuristic for the Set Covering Problem (SCP) is the *greedy* algorithm, independently studied by L. Lóvasz [12], D. Johnson [13], and V. Chvátal [14]. On an SCP instance of a basic set $X$ and family $\mathcal{F} \subseteq 2^X$, $|X| = m$, $\mathcal{F} = |n|$, the greedy algorithm selects iteratively a subset $A \in \mathcal{F}$ that covers the most out of the still uncovered basic elements of $X$. The greedy algorithm is $O(\log m)$-approximate for the problem. This result holds in the worst case. It is natural to assume that worst case instances are not typical instances occuring in real-world applications. Therefore, study of the average case behavior of algorithms on *typical* instances is also of great interest in practical applications. One way to develop average case analysis is by assuming that the instances of a problem are generated via a probabilistic distribution. Although the assumed distribution may not depict the reality convincingly, still, it can serve for generating benchmarks for validation of heuristic algorithms.

Three probabilistic models have been previously studied in the literature for the set covering problem. None of these models defines costs of $X$ subsets in $\mathcal{F}$, therefore it is assumed that all subsets have a cost of 1. The most general of these models is the constant density *independent* model with parameters $(m, p)$: each element of the 0/1 incidence matrix of the instance is determined to be 1 with probability $p$. As $m$ grows infinitely the density $p$ of the matrix remains constant. Less general models include the *incremental* model studied in [15, 16] and the probabilistic version of Karp's model studied in [17]. In [16, 17] convergence of the sequence of random variables $\{opt_m\}$ (value of optimum solution) is studied in the independent model and it is shown to be $\Theta(\log m)$. Moreover, in [16] two asymptotically optimum randomized polynomial-time algorithms are analyzed for the SCP in the incremental and independent models respectively.

In [11] we investigate the performance of a simple $O(nm)$ complexity deterministic algorithm in the independent model. For the sequence of random variables $\{S_m\}$ specifying the value of returned solutions we show that: $E[S_m] \leq 1 - \frac{\log(pm)}{\log(1-p)} + \frac{1}{p}$.

This result along with the convergence of the optimum solution value given in [16] immediately implies that the simple deterministic algorithm is asymptotically optimum in expectation [3]. Furthermore, we show that the variance of produced solution values is upper-bounded by a constant: $V[S_m] < 1/p$. This analysis implies the first known probabilistic results for the well known greedy algorithm in the independent model. If $\{G_m\}$ is the sequence of random variables specifying the value of a greedy solution, then one can show that $G_m \leq S_m$. Hence we immediately deduce that the greedy algorithm is also asympotically optimum in expectation. However, for the variance of greedy solution values, we were able to show a slightly weaker result, namely that: $\lim_{m\to\infty} V[G_m] \leq \frac{1}{p}$.

These are the first average case results for the greedy algorithm in the most general probabilistic model for the SCP. In [11] we also show that the independent model provides robust feasible solutions, in the sense that it takes addition of at least $\Omega(m)$ probabilistic constraints of density $p$ (i.e. rows to the incidence matrix) in expectation to render any feasible solution unfeasible. We believe that the analysis of the greedy algorithm can be tightened further, so that it can be shown that this algorithm is asymptotically optimum with probability approaching 1 as $m$ grows infinitely.

## 3 Dynamic Graph Algorithms

A dynamic graph algorithm maintains dynamically the solution to a graph problem as the underlying graph changes by edge insertions and deletions [2]. Algorithms that handle both kinds of changes are called *fully* dynamic, while algorithms that handle exclusively one kind are called *incremental* (for edge insertions) or *decremental* (for edge deletions). The core of such an algorithm is essentially a data structure encoding the underlying graph along with additional information needed for the problem at hand, that provides operations for modifying the graph structure with edge insertions and deletions, and for querying the current solution to the problem. The field of dynamic graph algorithms has seen extended progress during the last twenty years.

Connectivity has been studied extensively for dynamic graphs [18, 19]. The first fully dynamic algorithms for maintaining $k$-connected components of a dynamic graph have emerged by usage of the technique of *sparsification* [20]. For 2-vertex and 2-edge connectivity there exist almost optimal fully dynamic algorithms [19]. Regarding optimization problems in the context of dynamic graphs, the *minimum cost spanning tree* (MST) problem has a long history in the field and the latest results are almost optimal [19].

For the case of directed graphs the results are scarce, and it has been a common experience that development of dynamic algorithms for directed graphs is a significantly more challenging matter. Problems that have dominated the interest of the research community for the past fifteen years include dynamic trnsitive closure [21], and dynamic single-source shortest paths [22–24].

In [10, 5] we considered a family of optimization problems with connectivity constraints in the context of dynamic undirected graphs, and in [7, 6] the *directed minimum cost spanning tree* for directed graphs.

### 3.1 Dynamic Bottleneck Optimization for Graph Connectivity

In its *static* version the problem of bottleneck optimization for $k$ connectivity requires the determination of a minimum value $b$ over a weighted complete graph $K_n$, so that all edges $e \in K_n$ with cost $c(e) \leq b$ induce a $k$-connected edge subgraph of $K_n$.

We refer to $b$ as the *bottleneck value*. In the dynamic version of the problem we want to update the bottleneck value efficiently subject to modification of an edge's cost. In [5, 10] we considered the $k$-edge-connectivity constraints for arbitrary constant $k$, and the 2-vertex-connectivity constraint. For these problems we designed dynamic algorithms that can answer the optimum bottleck value in $O(1)$ time, and take care of an edge cost update in $\tilde{O}(n)$ time[1]. For 2-vertex-connectivity in particular, we show that there is a deterministic fully dynamic algorithm with $O(n\alpha(n))$ update complexity and a randomized fully dynamic algorithm with high probability $O(n)$ update complexity. The proposed techniques combine efficiently through the technique of *sparsification* [20] greedy bottleneck optimization procedures with the best known incremental algorithms from the literature for maintaining the $k$-edge/vertex-connected components [18, 25–27] and fully dynamic MST [19]. For the case of 2-vertex-connectivity we also use the randomized linear-time MST algorithm of [28], in order to design randomized linear-time updates.

The complexity results were shown to be tight in the worst case. Edge connectivity constraints were much easier to handle than vertex-connectivity. A challenging open question that we are currently working on concerns the 3-vertex-connectivity constraint under dynamic bottleneck optimization.

### 3.2 Dynamic Directed Minimum Cost Spanning Tree

The *directed minimum spanning tree* (DMST) problem is defined over a directed weighted graph $G(V, E)$ with edge weights given by a cost function $c : E \to \Re^+$, as *a maximal acyclic subset of edges $T \subseteq E$ of $G(V, E)$, such that each vertex $v \in V$ has at most one incoming edge in $T$, and $c(T) = \sum_{e \in T} c(e)$ is minimum over all such maximal subsets.*

The DMST definition is essentially reminiscent of the definition of a minimum cost spanning tree on undirected graphs, differing only by an additional constraint stating that each vertex has at most one incoming edge in the DMST. This constraint gives the DMST a directional property: it is a tree "blossoming" out of its root, towards the other vertices of the digraph. If $G$ is strongly connected, the definition implies indeed such a directed tree (also called *arborescence*) on which, the root is the only vertex with no incoming edge. If $G$ is

[1] Notation $\tilde{O}$ hides polylogarithmic factors from the complexity expression

not strongly connected the resulting structure might be a collection of arborescences, also called a *branching* [29, 30]. A single polynomial time algorithm, due to Jack Edmonds is known for this problem [29]. Improved $O(\min\{m \log n, n^2\})$ complexity implementation for a digraph of $n$ vertices and $m$ edges is given in [30]. The complexity was further improved to $O(m + n \log n)$, by the heavily idiosyncratic implementation of [31]. Further improvements achieved for sparse digraphs are described in [32].

The DMST has been a wide open problem for the field of dynamic graph algorithms in constrast to the almost optimal results obtained for the MST on undirected graphs [19]. In our work we design a fully dynamic algorithm for this problem, and analyze its complexity in the *output complexity* model [7, 6]. The output complexity model has seen extended use in the analysis of dynamic graph algorithms [22] and measures the complexity of updating the output of the algorithm per edge operation as a function of the number of constituents of the output that are affected by the operation (i.e. need to be updated). The algorithm proposed in [7, 6] has an output complexity $O(n + ||\delta|| + |\delta| \log |\delta|)$ per edge operation on general graphs, where $\delta$ denotes roughly the set of vertices that are affected by the operation, while $||\delta||$ is the number of edges incoming to these vertices. It is interesting to note that the complexity of the algorithm reduces to $O(\log n)$ if the underlying dynamic graph remains acyclic in between dynamic edge operations. A partial hardness result derived in [7, 6] states that verification time of a DMST is $O(n^2)$ in the worst case for general graphs, while a DMST may change entirely after an edge operation. However, full determination of the problem's dynamic complexity is still a challenging open question. Experiments with the proposed algorithm have shown a speedup factor of more than 2 on dense graphs, when comparing it against the naive strategy of re-evaluating the DMST from scratch when needed, while for the case of sparse graphs the speedup was significantly larger [7]. The proposed algorithm also finds application in dynamic bottleneck optimization for strong connectivity.

## 4 Network Optimization

Network optimization constitutes a core area of combinatorial optimization. Our work concerns two well-studied network optimization problems, namely the *Object Placement* problem and the *Steiner Tree/Forest* problems. Recent trends in network optimization subsume game-theoretic analysis for deciding the impact of selfish behavior on voluntarily formed networks and optimization under uncertainty. We treat the Object Placement problem under the first framework and the Steiner Tree/Forest problem under the second framework.

### 4.1 Distributed Selfish Replication

We consider a voluntarily formed file-sharing network. The network consists of a set of user-nodes $N = \{1, \ldots n\}$, and each user $j \in N$ has at his/her disposal an

integer amount of local storage $C_j$. Each user demands access to a set of unit-sized objects (that can be files or services), denoted by $R_j$. Let $\mathcal{O} = \cup_{j \in N} R_j$. The frequency by which an object $i \in R_j$ is going to be accessed by user $j$ is given to be $f_{ji}$. All objects are accessible from a distant server $s$. A node $j \in N$ can store some copies of the objects in $R_j$ locally, in its local memory. These objects can then be accessed at a transfer cost $t_l$. Objects that are not stored in local memory, must be retrieved from elsewhere. If such an object is stored at another node $j' \in N$, it can be retrieved at a transfer cost $t_r >> t_l$. If an object of interest to $j$ is not stored on any of the network's nodes, then this object can be retrieved from $s$ at a maximum transfer cost $t_s >> t_r$. The problem amounts to selecting which objects to replicate in local storage qhile respecting the local storage capacity, and assign every ⟨node,object⟩ pair to the nearest place from which the object can be retrieved, so as to minimize the total incurred access cost. Let $\phi_j : R_j \rightarrow N + s$ be such an assignment function, and $c_{jj'}$ denote the transfer cost from node $j'$ to node $j$. Then the total access cost is: $Z = \sum_{j \in N} \sum_{i \in R_j} f_{ji} c_{j\phi_j(i)}$

The model we have just described was originally proposed in [33] where it was also shown that it can be solved optimally in polynomial-time. In [9, 8] we considered the following remarks for the case of voluntarily formed file-sharing networks:

- **Mistreatment/Voluntary Participation**: In an optimum solution the local storage of some user $j \in N$ might be used entirely towards serving the demands of other overactive users, that generate the volume of access cost, and not the demands of $j$. Under the assumption of voluntary participation $j$ will be dissatisfied, and may choose to leave the network. In fact, every user that does not gain from his/her participation to the network as much he/she would gain by managing its own local storage independently, has an incentive to leave the network.
- **System Stability**: A mistreated user may choose, instead of leaving the network, to modify the contents of its local storage, so as to repair his/her poor service and decrease his/her own high access cost. In any case, a user may choose to modify the locally stored objects, so as to decrease the access cost he/she experiences.

Both these considerations state that the network's performance may be unstable, while in the worst case the network may disintegrate as dissapointed users choose to cancel their participation to it. In order to cope with these conditions, we formulated the problem as a *strategic game* [34] among selfish agents that want to minimize their access cost by participating in the network. Each node of the network is an agent associated to strategy space defined by the subsets of objects that this node may choose to replicate locally.

In [9, 8] we showed that this game admits pure strategy Nash equilibrium solutions, by designing an algorithm that finds a placement of objects in the local memories of the nodes in such a way, that no node has incentive to modify its local contents (i.e. it will not decrease its own access cost by unilaterally deviating

from the placement). This solution guarantees system's stability. Furthermore we show that this algorithm guarantees voluntary participation, since it outputs solutions in which every node experiences an access cost at most as much as in the case it was managing its own local storage without participating in the network.

In [9, 8] we also described a modified version of the proposed algorithm for finding pure Nash equilibrium strategies that gurantee voluntary participation, so as to give the chance to all nodes to minimize their individual access costs as much as possible. For this modified version of the algorithm and for the case where all nodes have equal local storage capacity and share a common Zipf-like distribution of access frequency to objects, we have derived an approximate expression for the relative gain of each node in comparison to the gain achievable by the node when acting in isolation.

A distributed implementation of the algorithm is provided in [8], that is efficient in terms of the amount of information exchanged among nodes of the network during its execution: while a naive distributed implementation may require transmission by every node of $O(|\mathcal{O}|)$ information amount, we propose a simple scheme that requires only $O(\sum_{j \in N} C_j)$ order of information exchange, which is generally much less (also in practice) than the total number of required objects bound.

In the context of our ongoing research on this problem, we have extended our results to more general network models. We have shown existence of pure strategy Nash equilibria for balanced hierarchically clustered networks with arbitrarily many servers and have proven an upper bound for the *price of anarchy* [35] (cost of most expensive equilibrium to cost of optimum object placement) and a matching worst-case lower bound for the *price of stability* [36] (cost of least expensive Nash equilibrium to cost of optimum object placement).

### 4.2 Robust Steiner Tree

In [37] we investigate the Steiner Tree problem in the presence of input uncertainty, under the model of *Probabilistic Optimization* [38, 39], along with some extensions to the model of *Robust Discrete Optimization* [40]. We refer to the problem as the *Robust Steiner Tree* problem, and it is extended in two stages as follows. In the first stage we are given a complete weighted graph $G_0(V, E)$, with a cost function $c : E \rightarrow \Re^+$ and a subset $S \subseteq V$ of vertices that require connection via a Steiner Tree. Each vertex $v \in V \setminus S$ has a probability $p(v) \in [0, 1]$ of survival in the second stage of the problem. In the second stage only a subset of vertices $V'$ of $V \setminus S$ survives, giving rise to a complete subgraph $G_1 = G_0[V']$ of $G_0$. If $T_0$ is a first-stage Steiner tree connecting $S$, then a portion $T_0' \subseteq T_0$ survives in second stage. We want to devise an efficient *modification* algorithm for rendering $T_0'$ a feasible Steiner tree $T_1$ for $S$ on $G_1$. Given such a modification algorithm $\mathcal{A}$ and a first-stage feasible solution $T_0$, let $c(\mathcal{A}(G_1, T_0)) = c(T_1)$. The objective is to build *a priori* an appropriate first-stage solution so as to minimize the expected second-stage cost:

$$Z_{exp}(T_0) = \sum_{V' \subseteq V \setminus S} \Pr[V'] c(\mathcal{A}(G_0[V'], T_0))$$

This is the problem formulated precisely in the framework described in [38, 39]. In [37] we also considered the robust objective criterion proposed in [40]:

$$Z_{\max}(T_0) = \max_{V' \subseteq V \setminus S} c(\mathcal{A}(G_0[V'], T_0))$$

which eliminates the need for probabilities of survival over the set $V \setminus S$. It is easy to see why *a priori* optimization of any of these objective functions is $NP$-hard given any *modification* algorithm $\mathcal{A}$: there is always the case that $G_1 = G_0$, hence one must have found an optimum Steiner tree for $S$ in the first stage, which is $NP$-hard. In our work we also show by a reduction from Steiner tree on a complete graph with edge costs in $\{1, 2\}$ that it is $NP$-hard to repair an arbitrary first-stage solution towards minimizing either $Z_{exp}$ or $Z_{max}$, even in the case when the first-stage solution is optimum.

We have designed an $O(n\alpha(n))$ complexity modification algorithm for which we have shown that $Z_{exp}$ is computable in polynomial time $O(n^3)$ given a first stage solution $T_0$. This implies that the problem of *a priori* optimization of $Z_{exp}$ for the proposed algorithm belongs in $NPO$, the class of optimization problems that have their corresponding decision problem in $NP$. If the cost function $c : E \to \Re^+$ is metric, we can show that *the proposed modification algorithm produces a second-stage feasible Steiner tree $T_1$ with cost at most $2c(T_0)$, where $T_0$ is the first-stage feasible solution.*

With simple arguments one can show that every $\alpha$-approximate first-stage Steiner tree, can *a priori* approximate both, $Z_{exp}$ and $Z_{\max}$ by a factor of $2\alpha$. This immediately results in approximation factors of 2, 2.48 for metric edge costs in $\{1, 2\}$ and 3.01 for general metric edge costs [41]. There is a basic observation that makes these results extremely interesting: at first the proposed modification algorithm is much faster than every known approximatin algorithm for the Steiner tree problem. Hence the trivial practice of simply ignoring the remainders of a first-stage solution and approximating the second-stage instance form scratch is not as efficient. Furthermore, under this trivial practice and given an optimum first-stage solution, every $\alpha$-approximation algorithm used from scratch in second stage incurs a $1 + \alpha > 2$ approximation factor of $Z_{exp}$, $Z_{\max}$, as opposed to the 2-approximation achieved by the proposed approximation algorithm. Another remark is that, in metric graphs one could easily produce a 2-approximation factor first-stage solution remaining feasible and approximate also in second-stage, by using the well known *minimum spanning tree* heuristic [1]. However, the proposed modification algorithm is still able to repair efficiently other first-stage solutions such as the optimum, a fact which is interesting in its own right, since such a repair is $NP$-hard to achieve in general.

Another fact that makes the proposed modification algorithm interesting is that it can be generalized in the case of the *Robust Steiner Forest* problem

(see [4]), where the MST heuristic is not applicable In this case and for metric graphs, we can apply the algorithm to the remainders of each tree of a first-stage Steiner forest, thus achieving a $2\alpha$-approximation of $Z_{exp}$,$Z_{\max}$, when the first stage Steiner forest is $\alpha$-approximate. This leads to approximation factors of 2 and 4, depending on whether an optimum first-stage solution was used or a 2-approximate one (by usage of the primal-dual algorithm for Steiner forest [42]).

## References


1. Vazirani, V.: Approximation Algorithms. Springer-Verlag, Heidelberg (2003)
2. Eppstein, D., Galil, Z., Italiano, G.F.: 8: Dynamic graph algorithms. In: Algorithms and Theory of Computation Handbook. CRC Presss (1999)
3. Telelis, O.A.: Set Covering and Network Optimization: Dynamic and Approximation Algorithms (in Greek). PhD thesis, Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Greece (November 2006)
4. Paschos, V.T., Telelis, O.A., Zissimopoulos, V.: Steiner forests on stochastic metric graphs. In: Proceedings of the 1st International Conference on Combinatorial Optimization and Applications (COCOA), Springer, LNCS 4616. (2007) 112–123
5. Telelis, O.A., Zissimopoulos, V.: Dynamic Bottleneck Optimization for Graph Connectivity. Under Revision in Information Processing Letters (2006)
6. Pollatos, G.G., Telelis, O.A., Zissimopoulos, V.: Fully Dynamic Directed Minimum Spanning Trees. Under Revision in Theoretical Computer Science (2006)
7. Pollatos, G.G., Telelis, O.A., Zissimopoulos, V.: Updating Directed Minimum Cost Spanning Trees. In: Proceedings of the 5th Workshop on Experimental Algorithms (WEA), Springer, LNCS 4007. (2006) 291–302
8. Laoutaris, N., Telelis, O.A., Zissimopoulos, V., Stavrakakis, I.: Distributed Selfish Replication. IEEE Transactions on Parallel and Distributed Systems **17**(12) (2006) 1401–1413
9. Laoutaris, N., Telelis, O.A., Zissimopoulos, V., Stavrakakis, I.: Local Utility Aware Content Replication. In: Proceedings of the Fourth IFIP International Conference on Networking (NETWORKING), Springer, LNCS 3462. (2005) 455–468
10. Telelis, O.A., Zissimopoulos, V.: Dynamic Bottleneck Optimization for 2-Vertex and Strong Connectivity. In: Proceedings of the 2nd Balkan Conference in Informatics (BCI). (2005) 52–59
11. Telelis, O.A., Zissimopoulos, V.: Absolute $o(\log m)$ Error in Approximating Random Set Covering: An Average Case Analysis. Information Processing Letters **94**(4) (2005) 171–177
12. Lóvasz, L.: On the ratio of optimal integer and fractional covers. Discrete Mathematics **13** (1975) 383–390
13. Johnson, D.S.: Approximation algorithms for combinatorial problems. Journal of Computer and Systems Sciences **9** (1974) 256–278
14. Chvátal, V.: A greedy heuristic for the set covering problem. Mathematics of Operations Research **4** (1979) 233–235
15. Weide, B.W.: Statistical methods in algorithm design and analysis. PhD thesis (1978)
16. Vercellis, C.: A probabilistic analysis of the set covering problem. Annals of Operations Research **1** (1984) 255–271

17. Fontanari, J.F.: A Statistical Mechanics Analysis of the Set Covering Problem. Journal of Physics A: Mathematical and General **29**(3) (1996)
18. Westbrook, J., Tarjan, R.E.: Maintaining Bridge-Connected and Biconnected Components On-Line. Algorithmica **7**(5&6) (1992) 433–464
19. Holm, J., de Lichtenberg, K., Thorup, M.: Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. Journal of the ACM **48** (July 2001) 723 – 760
20. Eppstein, D., Galil, Z., Italiano, G.F., Nissenzweig, A.: Sparsification - a technique for speeding up dynamic graph algorithms. Journal of the ACM **44**(5) (1997) 669–696
21. Sankowski, P.: Dynamic Transitive Closure via Dynamic Matrix Inverse (Extended Abstract). In: Proceedings of the IEEE Annual Symposium on Foundations of Computer Science (FOCS). (2004) 509–517
22. Ramalingam, G.: Bounded Incremental Computation. Springer, LNCS 1089 (1996)
23. Sankowski, P.: Shortest Paths in Matrix Multiplication Time. In: Proceedings of the European Symposium on Algorithms (ESA). (2005) 770–778
24. Sankowski, P.: Subquadratic Algorithm for Dynamic Shortest Distances. In: Proceedings of the Conference on Computational Optimization of Networks (COCOON). (2005) 461–470
25. Poutré, H.L.: Maintenance of 2- and 3-edge-connected components of graphs II. SIAM Journal on Computing **29**(5) (2000) 1521–1549
26. Dinitz, Y., Westbrook, J.: Maintaining the Classes of 4-Edge-Connectivity in a Graph On-Line. Algorithmica **20**(3) (1998) 242–276
27. Henzinger, M.R.: A Static 2-Approximation Algorithm for Vertex Connectivity and Incremental Approximation Algorithms for Edge and Vertex Connectivity. Journal of Algorithms **24**(1) (1997) 194–220
28. Karger, D.R., Klein, P.N., Tarjan, R.E.: A Randomized Linear-Time Algorithm to Find Minimum Spanning Trees. Journal of the ACM **42**(2) (1995) 321–328
29. Edmonds, J.: Optimum branchings. J. of Res. of the Nat. Bureau for Standards **69B** (1967) 125–130
30. Tarjan, R.E.: Finding optimum branchings. Networks **7** (1977) 25–35
31. Gabow, H.N., Galil, Z., Spencer, T.H., Tarjan, R.E.: Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. Combinatorica **6** (1986) 109–122
32. Mendelson, R., Tarjan, R.E., Thorup, M., Zwick, U.: Melding Priority Queues. In: Proc of the Scandinavian Workshop on Algorithms Theory (SWAT). (2004) 223–235
33. Leff, A., Wolf, J., Yu, P.S.: Replication algorithms in a remote caching application. IEEE Transactions on Parallel and Distributed Systems **4**(11) (1993) 1185–1204
34. Osborne, M.J., Rubinstein, A.: A course in game theory. MIT Press (1994)
35. Koutsoupias, E., Papadimitriou, C.H.: Worst-case Equilibria. In: Proceedings of the Symposium on Theoretical Aspects of Computer Science (STACS). (1999) 404–413
36. Anshelevich, E., Dasgupta, A., Tardos, E., Wexler, T.: Near-optimal network design with selfish agents. In: Proceedings of the Annual ACM Symposium on Theory of Computing (STOC). (2003) 511–520
37. Paschos, V.T., Telelis, O.A., Zissimopoulos, V.: Probabilistic Models for the Steiner Tree Problem. Under Revision in Networks (2007)
38. Bertsimas, D.: Probabilistic Combinatorial Optimization. PhD thesis (1988)
39. Jaillet, P.: Probabilistic Traveling Salesman Problems. PhD thesis (1985)

40. Kouvelis, P., Yu, G.: Robust Discrete Ooptimization and its Applications. Kluwer Academic Publishers (1997)
41. Robins, G., Zelikovsky, A.: Improved Steiner Tree Approximation in Graphs. In: Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA). (2000) 770–779
42. Agrawal, A., Klein, P.N., Ravi, R.: When Trees Collide: An Approximation Algorithm for the Generalized Steiner Problem on Networks. SIAM Journal on Computing **24**(3) (1995) 440–456