An ant approach to membership overlay design*

Results on the dynamic global setting

Vittorio Maniezzo[1], Marco Boschetti[2] and Mark Jelasity[1]**

[1] Department of Computer Science, University of Bologna, Italy,
{maniezzo,jelasity}@cs.unibo.it,
[2] Department of Mathematics, University of Bologna, Italy,
boschett@csr.unibo.it

Abstract. Designing an optimal overlay communication network for a set of processes on the Internet is a central problem of peer-to-peer (P2P) computing. Such a network defines membership and allows for members to disseminate information within the group. The network has to be robust and the available bandwidth has to be utilized in an optimal manner to allow for maximally efficient communication. This problem can be formulated as a dynamic optimization problem where classical combinatorial optimization techniques must face the further challenge of time-varying input data. ACO systems appear to be particularly fit for this class of problems, being able to construct an internal model of the instance to face and to exploit it for fast adaptation to modified contexts. This paper proposes to use elements resulting from mathematical techniques, in this case Lagrangean relaxation, in an ACO framework in order to achieve sound hot start states for fast response to varying network structures.

1 Introduction

The rapid evolution of the Internet and related technology, the increasing bandwidth and the enormous number of network-enabled computing devices resulted in a new field of distributed computing, the so called peer-to-peer (P2P) computing [21], which is an umbrella term for a wide range of areas which include file sharing, grid computing, distributed search, distributed hash tables, etc. This new field poses a large number of new challanges, in particular, optimization problems, most of which are dynamic in nature as the defining elements change over time. A recent research thread in ACO systems supports the intuition that ant algorithms are particularly fit for dynamic optimization problems because of their ability to construct an internal representation of the essential elements of the instance to solve, representation which needs to be updated and not reconstructed when the instance changes ([9, 23, 6, 12, 11]).

* This work was partially supported by the Future & Emerging Technologies unit of the European Commission through Project BISON (IST-2001-38923).

** Also with MTA RGAI, SZTE, Szeged, Hungary

This work reports on our approach of using an ant algorithm for a dynamic network design problem: P2P membership overlay network design. This problem arises in all P2P applications that are large and fully decentralized, like popular file sharing networks [10, 24] or gossip-based protocols for information dissemination [4, 5] and information aggregation (data mining) [22, 15]. Lacking a central service, participating nodes talk to each other directly, typically using a relatively small list of peer nodes they are aware of. Given the heterogenious bandwidth and availability constraints at each node, it is crucial for scalability and performance that participating nodes share the costs of the application in a fair manner. The optimization problem arises through the fact that the peers have to intelligently select the other peers they communicate with so that no participants get overloaded but available bandwidth is utilized reliably in an optimal manner by all network nodes.

For the full specification of the problem the target application has to be specified as well. In this work we are focusing on overlay networks applied by gossip-based protocols. In this case, each node sends gossip messages periodically to its neighbors. The applied membership overlay (as defined by the set of neighbors used to send gossip messages to) is typically a random or pseudo-random topology [8] sometimes taking into account network locality [20, 16]. Random topologies have a number of advantages from a theoretical point of view, however they ignore bandwidth and other constraints.

Unlike our problem, other rigorously formulated network optimization problems that consider communication cost, like the optimum communication cost spanning tree (OCST) problem [13] and the Steiner tree problem [14] are formulated to find a specific topology (e.g., tree) and they target other applications like broadcasting or search. Besides, the extremely dynamic character of P2P environments represents a novel requirement that needed to be incorporated into the formulation of our problem as well.

While our long term objective is to obtain a local algorithm, that—when run at all nodes concurrently—solves the above problem in a distributed manner, in this work we report results on a global algorithm that can however handle dynamicity, our main focus of research. Unfortunately paper length constraints prevent us to provide a detailed account of the rationals and the techniques we used. However, we believe to have achieved a sufficient presentation clarity and we refer the interested reader to [2] for further details.

2 Problem description

As described in the Introduction, we are focusing on overlay networks for gossip based protocols. We assume that we have a set of Internet nodes that wish to form a gossip group, that is, a group over which gossip-based protocols can be applied. Gossip protocols can disseminate information among the members [4, 5] or they can analyze some attributes of the nodes in a distributed fashion [22, 15]. In principle, each node can send a message to any other node applying the routing service of the Internet provided the target IP address is known (if we

ignore the effect of firewalls). In practice, it is not feasible to store all member addresses at all nodes because there can be too many of them and membership can change dynamically, so scalability problems arise. The typical solution that is normally adopted is storing only a limited number of peer addresses and using only those to send gossip messages to, which still allows gossip to spread efficiently.

The set of known peers at each node defines the *overlay network* in which there is a directed link between nodes i and j if i has the IP address of j in its list of peers. This network defines the membership in the group: those nodes that are connected to the overlay network are members since they can participate in the gossiping. The structure of this network has a major impact on the performance of the communication. If nodes with limited bandwidth have to send or receive too many messages (i.e., have too many outgoing or incoming connections) then the network will not function properly. It is important that load is distributed in a fair manner so that the throughput of the network is maximized without any nodes being overloaded.

In the following we will use the term "connection" as a shorthand for "allocation of non-zero bandwidth for possible communication". That is, node j is connected to, or a neighbor of, i if i allocates non-zero bandwidth for communicating with j. This sense of connection is not identical to direct physical connection or even proximity at the network level, it expresses a logical connection in the overlay network.

Let us now formulate this networking problem, that we shall call the Membership Overlay Design Problem (MOP), in a mathematical model as follows. A graph G=(V,E) of n vertices is given, where the nodes correspond to peers that want to communicate with each other, that is, that want to form a gossip group. The edges correspond to possible communication, that is, if there is an edge (ij) then i can possibly send a message to j using the underlying routing infrastructure. Each node can dynamically enter and exit the network, and when it is connected it can make use of a limited bandwidth. Therefore, each node has two associated weights, p_i and w_i, $i = 0, \ldots, n$, corresponding to its uptime (measured as the percentage of time that the peer is available and responding to traffic [25], normalized to 1) and to the available bandwidth of its connection to the Internet, respectively. The finer structure within the core Internet (the high performance routers and backbones) are not taken into account. In other words, as in P2P networks the bandwidth bottleneck is typically represented by the connection between the core Internet and the participating computer on the edge of the Internet, and only rarely by a main backbone within the core Internet, we approximate the bandwidth constraint locally using the bandwidth of the Internet connection of the participating node.

The MOP asks to find a subgraph $G'=(V,E')$ of G. The edges in the graph G' define the fact that two nodes actually decide to allocate some bandwidth to communicate with each other. In other words, when two nodes i and j establish a connection, each one must allocate part of its bandwidth. If b_i and b_j are the bandwidths which could be allocated by i and j, then the bandwidth of the

connection can be at most $b_{ij} = min\{b_i, b_j\}$. The two values b_i and b_j could be equal to w_i and w_j or could be less than that, due to other connections already maintained by the peers. Moreover, there is a lower bound L on the bandwidth of acceptable connections and it is anyway possible to put a limit on the maximal value that b_{ij} can take.

The graph G' has to be such that

1. the expected network throughput is maximized
2. the diameter of G' is minimized.
3. the total bandwidth used by each node i is less than or equal to w_i.

Note that 2) implies that G' is connected.

As mentioned, the final algorithm for solving this problem should be local: no global knowledge of the network is provided, each node i can exchange informations only with the nodes in $\delta'(i)$, that is, with its neighbors in G'. However, at this stage of our research we present a solution algorithm working at a global level.

2.1 The static subproblem

First a mathematical formulation (P) of the static version of the problem will be presented, which will be later adapted to the dynamic case. Formulation P is a mixed integer formulation for which we will later derive a polynomial upper bound and a relaxation framework.

A MIP formulation A comprehensive mathematical analysis of the MOP can be found in [2], here we report only some results which are relevant for this work. Specifically, we present a mathematical formulation of MOP, with reference only to objective 1) and leave the other two objectives to be handled by the ants metaheuristic.

Two sets of decision variables are used: $[x_{ij}]$ and $[\xi_{ij}]$, $(ij) \in E$. The decision variables x_{ij} specify the bandwidth allocated to the connection between peers i and j. Therefore they are continuous variables $0 \leq x_{ij} \leq b_{ij}$, which will be further constrained when they are not 0 to be at least L. Decision variables ξ_{ij} are binary variables which are 1 if arc (ij) is used for a connection, 0 otherwise.

The formulation, denoted P, is the following.

$$z_P = max \sum_{(ij)\in E} p_{ij} x_{ij} \tag{1}$$

$$s.t. \sum_{j\in\delta(i)} x_{ij} \leq w_i \quad i \in V \tag{2}$$

$$x_{ij} \geq L\xi_{ij} \quad (ij) \in E \tag{3}$$

$$x_{ij} \leq b_{ij}\xi_{ij} \quad (ij) \in E \tag{4}$$

$$\xi_{ij} \in \{0, 1\} \quad (ij) \in E \tag{5}$$

where $p_{ij} = p_i * p_j$, for each edge $(ij) \in E$ and $\delta(i)$ represents the neighborhood of i in G (i.e., $V \setminus \{i\}$ if graph G is complete).

The complexity of this problem is under study, but no straight solution methodology is available.

Lagrangean relaxation of P Formulation P can be effectively solved by a sequence of successive relaxations. This is justified by the assumption, to be *a posteriori* verified, that the optimum of the relaxed problem is structurally sufficiently similar to a feasible solution to permit to obtain one with minor adjustments without loosing much in solution quality.

The first relaxation is a LP relaxation of constraints (5), and substitutes them with constraints in the form $0 \leq \xi_{ij} \leq 1$. This makes the problem equivalent to the following problem LP, where variables ξ_{ij} become unnecessary.

$$z_{LP} = max \sum_{(ij)\in E} p_{ij} x_{ij} \tag{6}$$

$$s.t. \sum_{(ij)\in\delta(i)} x_{ij} \leq w_i \qquad i \in V \tag{7}$$

$$0 \leq x_{ij} \leq b_{ij} \qquad (ij) \in E \tag{8}$$

Notice that constraints 3 have been removed because the LP solution has them always satisfied (with fractional variables), they will be later dealt with by the ants procedure (see section 3.2).

Problem LP can obviously be solved by a LP-solver. However, we preferred to further relax it for two reasons. The first one is that we are not interested directly in the optimal LP solution but in a good approximation of its optimal dual variables, to be later used by the ants metaheuristic (see section 3.2). The second one is that, as mentioned in section 1 our ultimate objective is the design of a fully local optimization procedure: this is better supported by a further relaxation, in a Lagrangean fashion, of problem LP than by a straight application of LP solution algorithms.

This further relaxion can be done by associating a positive Lagrangean penalty λ_i to each constraint 7, resulting in the following formulation, denoted LR.

$$z_{LR}(\lambda) = max \sum_{(ij)\in E} (p_{ij} - \lambda_i - \lambda_j) x_{ij} + \sum_{i\in V} w_i \lambda_i \tag{9}$$

$$s.t.\ 0 \leq x_{ij} \leq b_{ij} \qquad (ij) \in E \tag{10}$$

In order to solve problem LP we must now find the minimum over all feasible λ vectors of the $z_{LR}(\lambda)$ costs, i.e., we must solve the Lagrangean dual $min\,[z_{LR}(\lambda) : \lambda \geq 0]$.

2.2 The dynamic case

In actual practice of P2P networks it can be observed that nodes continuously enter and exit the network, some nodes spending more time in the network while others join only for a short time. The problem formulation is not affected by dynamicity, in the sense that at any moment in time the problem formulation is as described. The only effect is that the graph G and all related elements are time-varying.

Whenever the average uptime of a node in the network is higher than the optimization time, it becomes feasible to re-optimize the network, taking into account the new network conditions. This could be done periodically or whenever significant network topology changes are detected. In addition, re-optimizations should be sufficiently frequent to ensure smooth performance. This means that optimization time must be short with respect to average node permanence time, putting a stress on optimization efficiency.

3 Ants for the static case

We used an ants metaheuristic for solving the whole static problem and we considered LP as a subproblem. The upper bound provided by the LP solution can be infeasible because the resulting overlay topology could be disconnected and some connections could be allocated a bandwidth less then L. It will be the task of ants to construct a feasible, high-throughput, low diameter connected solution.

In the following we will first detail how to get a possibly disconnected solution but with feasible bandwidths, then we describe how to include this routine in an ants framework.

3.1 Disconnected upper bound

Let z^*_{LR} be the solution obtained by the subgradient optimization of problem LR using penalties λ^*_i, $i \in V$. The solution could be infeasible for problem P because of the relaxed constraints, thus it could contain arcs (ij) which have an allocated bandwidth less than L.

An heuristic solution is obtained by considering the optimized costs $c^*_{ij} = (p_{ij} - \lambda^*_i - \lambda^*_j)$, ranking all arcs $(ij) \in E$ by non increasing c^*_{ij} values and allocating all possible bandwidth to each successively considered connection. More in detail, the algorithm is the following.

LagrHeuristic(c^*)
1 Order all arcs in E by decreasing c^*
2 initialize $s_i = b_i$ for each $i \in V$
3 **foreach** *arc* (ij) **in** E in nonincreasing c^* order
4 **do** $slack = min\{s_i, s_j\}$
5 **if** $slack \geq L$

6 **then** $x_{ij} = slack$
7 $\xi_{ij} = 1$
8 $s_i = s_i - slack; s_j = s_j - slack$

This approach is derived from the exact method for solving continuous knapsack problems [19]. In our case it is not guaranteed to be optimal but it consistently produces good quality solutions in time $O(n \log n)$, where n is the number of arcs, the highest cost operation being the ordering of the arcs.

3.2 The ants heuristic

The solution obtained by the procedure described in subsection 3.1 can be infeasible for problem P because of its disconnectedness. It is necessary to reduce the solution objective function value in order to introduce new arcs which ensure connectivity. This is the task of the ants algorithm. The objective of each ant is to construct a connected solution. This is done by first identifying all connected components, then all vertices which could be endpoints of a new solution arc and finally by searching in the space of these candidate arcs.

The connected components in the *LagrHeuristic* solution S^d are identified and maintained by means of up-trees [3]. It is assumed that in actual practice it is always possible to identify in each connected component at least one node which has an incoming connection whose allocated bandwidth can be decreased by L. To enter a new arc in the solution it will in fact be necessary to use bandwidth previously allocated by at least one of its endpoints to another, more profitable connection. The new arcs will therefore be introduced with the least feasible bandwidth (i.e., L) in order to disrupt the solution quality the least.

Let S_h^d be the subset of nodes belonging to the $h-th$ connected component which have at least one connection whose bandwidth can be decreased by L. Each ant will try to identify arcs (ij), with $i \in S_h^d$ and $j \in S_l^d$, $h \neq l$, so that the global solution is connected and the solution cost is maximized.

The implemented ants algorithm from a structural viewpoint is a very standard ants algorithm, whose main steps are as follows.

ANTSMOP(c^*, λ)
1 Initialize trails $[\tau_{ij}]$ and solutions $S_k = \emptyset$, $k = 1 \ldots n$
2 **repeat**
3 **repeat**
4 **for** ants k=1 to m
5 **do** Choose the next arc $(ij) \in E$ to append to S_k
6 $S_k = S_k \bigcup (ij)$
7 **until** (all ants have their solution completed)
8 Update trails
9 **until** (Termination condition)

The choice of the next arc to append, at step 5, is obviously made in probability. The formulae used at step 5 and at step 8 are those proposed in [18], that is, each $k-th$ ant moves from its current state ι to a feasible state ψ in probability, coording to the formula:

$$p_{\iota\psi}^{k} = \frac{\alpha \cdot \tau_{\iota\psi} + (1-\alpha) \cdot \eta_{\iota\psi}}{\Sigma_{(\iota\nu) \notin tabu_k} (\alpha \cdot \tau_{\iota\nu} + (1-\alpha) \cdot \eta_{\iota\nu})} \tag{11}$$

Whereas trails are updated by means of formula 12

$$\tau_{\iota\psi}(t) = \tau_{\iota\psi}(t-1) + \tau_{\iota\psi}(0) \cdot (1 - \frac{z_{curr} - LB}{\overline{z} - LB}) \tag{12}$$

where z_{curr} is the cost of each current solution, $\overline{z}$ is the average of the last computed solutions and LB is a lower bound to the optimal problem solution cost. Detailed descriptions in [18].

Parameters to tune are m, number of ants, α, relative importance of visibility w.r.t. trail (multiplicative coefficient), and the termination condition, in our case maximum CPU time allowed. The cost average to use in the trail update formula at step 8 was made over the costs of all solutions computed at each iteration of loop 3-7.

The elements to define for the MOP-specific algorithm are trail and visibility initialization, and the bound to use for trail update.

Trail initialization. Trails were initialized by means of the optimized Lagrangean multipliers λ, which penalize relaxed total node load constraints thus are higher for a nodes i and j when it would be desirable to have connection (ij) in the solution (i.e., $x_{ij} > 0$) but there is not enough bandwidth available to allocate at least L to it. Therefore penalties quantify desirability other than the cost, and we used those values to initialize trails. Notice that this is not the first work proposing the use of non-uniform trail initialization, other relevant contributions to this topic can be found for example in [18], [1] and [17].

Visibility initialization. Being MOP a maximization problem, it comes natural to set η_{ij} equal to the arc cost, for ll $(ij) \in E$. While this is true, we choose to use costs c_{ij}^{*} in order to take into consideration also the global information which comes from bound pricing.

3.3 ANTS in the dynamic setting

The dynamic setting introduced in section 2.2 has been tackled by means of a continuous application of the interwoven Lagrangean and ants procedures described for the static case.

Since variations of the network structure happen continuously, the optimization algorithm is run continuously. It is assumed that the speed of execution of

iterations of the subgradient optimization procedure is higher than the rate of network changes, thus that a few subgradient iterations can be performed between consecutive network changes. The exact number of iterations is currently a parameter, which therefore implicitly quantifies the network variability.

Three algorithms are actually run concurrently and syncronously. The first is a standard subgradient optimization procedure which updates Lagrangean penalties [7]. The second is the lagrangean heuristic *LagrHeuristic* described in section 3.1, which is run every 5 iterations of the subgradient procedure. The third procedure is *AntsMOP*, and we perform one ants iteration for each subgradient iteration. The full pseudocode, at a high abstraction level, is thus the following. It is assumed that network changes are handled programmatically as events, thus in separate threads.

```
DynAntsMOP(G)
 1  Initialize subgradient step and penalties λ_i, i ∈ V
 2  Initialize trails [τ_ij] and iteration counter ItCount
 3  while (true)
 4      do ItCount++
 5          Solve problems SP1 and SP2
 6          Check for infeasibilities and update λ
 7          if (ItCount mod 5 == 0) then call LagrHeuristic
 8          Initialize solutions S_k = ∅, k = 1...n
 9          repeat
10                  for ants k=1 to m
11                      do Choose the next arc (ij) ∈ E to append to S_k
12                          S_k = S_k ⋃ (ij)
13            until (all ants have their solution completed)
14          Update trails
```

4 Computational results

The proposed algorithms were coded partly in c# and partly in c++ and run on a 1000 MHz Pentium III machine, under Windows XP. To validate the described techniques, we conducted a number of experiments on different simulated scenarios. We generated two sets of instance graphs; the first one (set A) has parameters which match those measured on real P2P networks as reported in [25], the second set (set B) has the nodes randomly generated on a x-y plane and p_{ij} inversely proportional to the Euclidean distance of nodes i and j, $\forall i, j \in V$, in order to produce meaningful visualizations.

More precisely, the parameters to be defined when constructing a testset are:
- n: number of network nodes.
- L: minimal acceptable connection bandwidth.
- p_i: uptime of node i.
- w_i: total bandwidth of node i.

For set A we have L=14 (Kbps), the uptime distribution (p_i) is derived from figure 6 of [25] and the bandwidth distribution (w_i is derived from from figure 4 of [25]. For set B we have L=2, uptime distribution such that $p_{ij} = M - dist[i,j]$, where M is a suitably big constant and $dist[i,j]$ is the euclidean distance of nodes i and j, bandwidth distribution uniform random in [2,11] and $L = 2$.

The computational testing was carried out separately for the static and the dynamic case. In the static case we wanted to determine the solution quality disruption, w.r.t. to upper bound z_P, due to the successive heuristics, while in the dynamic case the ease of adaptation to a mutated environment was of interest. Table 1 summarizes the results obtained. The columns show:

- *Id*: an instance identifier.
- n: number of nodes.
- z^*_{LR}: cost of best solution found for problem LR.
- t^*_{LR}: time to find the solution of cost z^*_{LR}.
- $\%z^*_d$: percentage decrease of cost for solution of algorithm *L*agrHeuristic.
- t^*_d: time to find the solution of algorithm *L*agrHeuristic.
- $\%z^*_{ants}$: percentage decrease of cost for solution of algorithm *A*ntsMOP.
- t^*_{ants}: time to find the solution of algorithm *A*ntsMOP.

The last two elements are reported for a number of ants / subgradient optimization iterations which is 10, 20 and 30, respectively. For the ants, parameters were $\alpha = 0.5$ and $num.ants = n/10$.

Problem		**LP**		**LagrHeu**		**10 iter**		**20 iter**		**30 iter**	
Id	**n**	z^*_{LR}	t^*_{LR}	$\%z^*_d$	t^*_d	$\%z^*_{ants}$	t^*_{ants}	$\%z^*_{ants}$	t^*_{ants}	$\%z^*_{ants}$	t^*_{ants}
A20	20	842	0.00	0.04	0.00	0.95	0.00	0.59	0.04	0.27	0.05
A50	50	2137	0.01	0.06	0.01	0.88	0.10	0.52	0.15	0.36	0.19
A100	100	8162	0.02	0.01	0.01	0.83	0.23	0.67	0.37	0.24	0.48
A500	500	20192	0.35	0.01	0.31	0.96	4.46	0.66	7.96	0.23	11.25
A1000	1000	53765	1.37	0.01	1.33	0.92	21.98	0.58	38.64	0.30	56.37
B20	20	53295	0.00	2.22	0.00	4.46	0.17	4.05	0.26	3.71	0.33
B50	50	131891	0.00	2.84	0.00	4.35	0.17	4.00	0.29	3.67	0.39
B100	100	281836	0.00	2.31	0.00	4.50	0.21	4.06	0.37	3.93	0.48
B500	500	1413301	0.33	2.81	0.44	4.59	4.05	4.15	8.14	3.95	10.72
B1000	1000	2940334	1.63	2.96	1.65	4.35	20.90	3.97	40.19	3.84	61.44

Table 1. Results on different MOP instances.

The results show how the proposed problem relaxation is highly effective on both, structurally very different, instance sets. The gap between the upper bound z^*_{LR} and the lower bound z^*_d is always reasonably small, and the feasible solution obtained by the ants does never disrupt excessively the bounds solution quality. Moreover, in the dynamic setting a fast adaptation is achieved, as testified by the solution quality obtained after different numbers of internal iterations. The results reported here are admittedly still incomplete, as computational testing

is still going on. These results are to be read as a validation of the feasibility of the approach, but significant improvements are possible. We did not concentrate much on this as we are now focused on the possibility of designing a fully decentralized, asynchronous Lagrangean optimizer.

5 Conclusions

The work reported in this paper has three main foci of interest. First, it confirms the viability of ants approaches for dynamic problems and it does this in the specific case of a cogent problem in telecommunication network design. Second, it proposes a structural way to hybridize a fundamental mathematical technique — Lagrangean optimization — with ants procedures. Both techniques implement a progressive refinement of an implicit model of the problem to solve and algorithm *A*ntsMOP shows a way to intertwine their evolutions so that each one exploits current results of the other (lower bounds and Lagrangean penalties, respectively). Finally, the paper contains a mathematical analysis of the MOP, which can represent a foundation for further algorithmic advances.

Our final research objective is the definition of a fully distributed, local protocol for optimized overlay network design. The results and the methods reported in this work, besides being of interest in their own, also represent a promising base over which to continue research toward the final objective.

References

1. C. Blum. Aco applied to group shop scheduling: A case study on intensification and diversification. In *Proc. ANTS'02*, 2002.
2. M. Boschetti, M. Jelasity, and V. Maniezzo. A local approach to membership overlay design. Working paper, Department of Computer Science, University of Bologna, 2004.
3. T. Corman, C. Leiserson, and R. Rivest. *Introduction to Algorithms.* MIT Press, 1990.
4. Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic algorithms for replicated database management. In *Proceedings of the 6th Annual ACM Symposium on Principles of Distributed Computing (PODC'87)*, pages 1–12, Vancouver, August 1987. ACM.
5. Patrick Th. Eugster, Rachid Guerraoui, Anne-Marie Kermarrec, and Laurent Massoulié. From epidemics to distributed computing. *IEEE Computer.* to appear.
6. C.J. Eyckelhof and M. Snoek. Ant systems for a dynamic tsp: Ants caught in a traffic jam. In M. Dorigo, G. Di Caro, and M. Sampels, editors, *Ant Algorithms : Third International Workshop, ANTS 2002*, volume 2463 / 2002 of *Lecture Notes in Computer Science*. Springer-Verlag, Heidelberg, 2002.
7. M.L. Fisher. The lagrangean relaxation method for solving integer programming problems. *Management Science*, 27(1):1–18, 1981.
8. Ayalvadi J. Ganesh, Anne-Marie Kermarrec, and Laurent Massoulié. Peer-to-peer membership management for gossip-based protocols. *IEEE Transactions on Computers*, 52(2), February 2003.

9. R.M. Garlick and R.S. Barr. Dynamic wavelength routing in wdm networks via ant colony optimization. In M. Dorigo, G. Di Caro, and M. Sampels, editors, *Ant Algorithms : Third International Workshop, ANTS 2002*, volume 2463 / 2002 of *Lecture Notes in Computer Science*. Springer-Verlag, Heidelberg, 2002.
10. Gnutelliums. http://www.gnutelliums.com/.
11. M. Guntsch, J. Branke, M. Middendorf, and H. Schmeck. Aco strategies for dynamic tsp. In *ANTS'2000 - From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms*, 2000.
12. M. Guntsch and M. Middendorf. Applying population based aco to dynamic optimization problems. In M. Dorigo, G. Di Caro, and M. Sampels, editors, *Ant Algorithms: Third International Workshop, ANTS 2002*, volume 2463 / 2002 of *Lecture Notes in Computer Science*. Springer-Verlag, Heidelberg, 2002.
13. T. C. Hu. Optimum communication spanning trees. *SIAM Journal on Computing*, 3(3):188–195, September 1974.
14. Frank K. Hwang, Dana S. Richards, and Pawel Winter. *The Steiner Tree Problem*. North-Holland, 1992.
15. Márk Jelasity and Alberto Montresor. Epidemic-style proactive aggregation in large overlay networks. In *Proceedings of The 24th International Conference on Distributed Computing Systems (ICDCS 2004)*, pages 102–109, Tokyo, Japan, 2004. IEEE Computer Society.
16. Meng-Jang Lin and Keith Marzullo. Directional gossip: Gossip in a wide area network. In Jan Hlavička, Erik Maehle, and András Pataricza, editors, *Dependable Computing – EDCC-3*, volume 1667 of *Lecture Notes on Computer Science*, pages 364–379. Springer-Verlag, 1999.
17. H. Lourenço and D. Serra. Adaptive search heuristics for the generalized assignment problem. *Mathware and Soft Computing*, 9(2-3):209–234, 2002.
18. V. Maniezzo. Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem. *INFORMS J. on Computing*, 11(4):358–369, 1999.
19. S. Martello and P. Toth. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., 1990.
20. Laurent Massoulié, Anne-Marie Kermarrec, and Ayalvadi J. Ganesh. Network awareness and failure resilience in self-organising overlays networks. In *Proceedings of the 22nd Symposium on Reliable Distributed Systems (SRDS 2003)*, pages 47–55, Florence, Italy, 2003.
21. Dejan S. Milojicic, Vana Kalogeraki, Rajan Lukose, Kiran Nagaraja, Jim Pruyne, Bruno Richard, Sami Rollins, and Zhichen Xu. Peer-to-peer computing. Technical Report HPL-2002-57, HP Labs, Palo Alto, 2002.
22. Alberto Montresor, Márk Jelasity, and Ozalp Babaoglu. Robust aggregation protocols for large-scale overlay networks. Technical Report UBLCS-2003-16, University of Bologna, Department of Computer Science, Bologna, Italy, December 2003. to appear in the proceedings of Distributed Systems and Networks (DSN 2004).
23. S. Nouyan. Agent-based approach to dynamic task allocation. In M. Dorigo, G. Di Caro, and M. Sampels, editors, *Ant Algorithms : Third International Workshop, ANTS 2002*, volume 2463 / 2002 of *Lecture Notes in Computer Science*. Springer-Verlag, Heidelberg, 2002.
24. FastTrack: Wikipedia page. http://en.wikipedia.org/wiki/FastTrack.
25. S. Saroiu, P. Krishna Gummadi, and S.D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking 2002 (MMCN'02)*, San Jose, CA, 2002.