AN OPTIMIZER IN THE TELECOMMUNICATIONS INDUSTRY

MAURICIO G. C. RESENDE

Abstract. This article relates some combinatorial optimization problems encountered by an optimizer in an industrial research laboratory at AT&T, a large telecommunications company. The article will appear in the Fall 2007 issue of SIAM SIAG/Optimization Views-and-News which will try to cover some aspects of the industrial applications of optimization and of the life of optimizers at AT&T, IBM, and Exxon.

1. Introduction

The impact of telecommunications in modern life in the last 100 years is remarkable. Telecommunications has evolved from telegraphy to landline local telephony, to long distance telephony, to mobile telephony, and to the Internet, which now carries voice, video, television, instant messaging, and makes electronic commerce possible. Optimization problems are abundant in the telecommunications industry and the successful solution of these problems has played an important role in telecommunications and its widespread use. Optimization arises in problems as varied as planning and design of optical and wireless networks, routing, restoration, network survivability, e-commerce, and search engine design [30].

In this article, I relate some of the optimization problems I have run into while a Member of Technical Staff at AT&T Bell Labs and AT&T Labs. I have worked in the Algorithms and Optimization Research Department, headed by David S. Johnson, since graduating with a Ph.D. in operations research from the IEOR Department at UC Berkeley. Members of the Algorithms and Optimization Research Department do research on theoretical and experimental algorithmics with a focus on optimization. The department's research spans topics such as computational complexity, approximation algorithms, linear and integer programming, network programming, network design, routing, location, data structures, algorithm engineering, and metaheuristics.

Optimization problems reach us in many ways. Some problems are brought to us by researchers in other fields who often have optimization problems they need to solve. Other problems arise from business-related projects. Sometimes it may be that we have previously done work related to that problem or on a related problem and our tools (solution methods or software) can be directly applied to the problem. In other circumstances, new tools may need to be developed.

The reader will observe that most of the examples described in this article involve the use of metaheuristics [19] to find cost-effective solutions to combinatorial optimization problems. Metaheuristics are high-level procedures that coordinate

Date: June 13, 2007. Revised October 22, 2007.
Key words and phrases. Telecommunications, optimization, metaheuristics.
AT&T Labs Research Technical Report TD-745RJS.

simple heuristics, such as local search, to find solutions that are of better quality than those found by the simple heuristics alone. The metaheuristic most used in this article is GRASP, or greedy randomized adaptive search procedures [15], and hybridizations of GRASP with other metaheuristics. Other metaheuristics used are path-relinking [18], genetic [27] and memetic [26] algorithms, variable neighborhood search [21], and evolutionary path-relinking [33].

The article is organized as follows. In Section 2 we examine two problems that arise in the management of points of presence of an Internet service provider. This is followed in Section 3 with routing problems, in Section 4 with network design problems, in Section 5 with network migration problems, and in Section 6 with a data mining application. Concluding remarks are made in Section 7.

2. Location problems

An Internet Service Provider (ISP) offering dialup access needs to determine where its modems will be located. Such a location is called a point of presence, or simply a PoP. We describe two problems. In the first, PoP locations need to be determined, while in the second, redundant PoPs need to be identified and shut down.

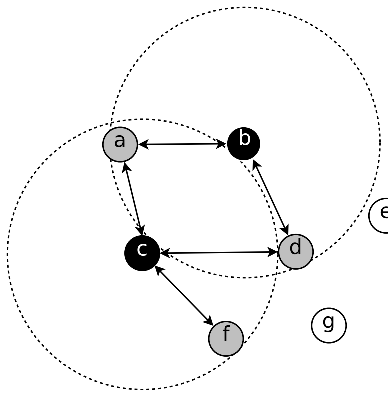
2.1. **PoP placement.** In the mid-1990s, when AT&T planned the U.S. rollout of its ISP (AT&T Worldnet), it was faced with the problem of where to locate the PoPs. Given a fixed number of modem pools that could be deployed, a set of about 50,000 potential PoP locations, and the location of each AT&T customer, the task was to determine the location of each modem pool such that the largest number of customers would be able to place a "free" local call to at least one modem. Free local calls in the U.S. are those made to numbers at most about 15 miles (24 km) away.

For this maximum customer coverage problem we designed an LP-based tool to compute an upper bound on the number of customers that could be covered and a tool [29] that used the metaheuristic GRASP to compute a placement within 1% of the optimum. Besides being very happy with the near-optimum placements produced by the tool, the planners also found the upper bound computation very useful since it enabled them to estimate the minimum number of modem pools needed to achieve a certain level of coverage.

2.2. **PoP elimination.** Over time, the GRASP-based tool was used to increase the number of PoPs deployed and consequently the coverage. Though a call to a PoP is free to the customer, it is not free to the ISP. Each PoP has an associated network cost, which is the hourly rate paid by the ISP to the network company transporting the access traffic. This hourly rate can vary greatly from PoP to PoP.

Since network costs and coverages of PoPs differ, an opportunity to eliminate PoPs could arise as long as coverage remained unchanged and the cost did not increase. In 2003, we conducted a study to determine if there were any PoPs that could be eliminated. We formulated this as a p-median problem which we solved with a tool based on a GRASP with evolutionary path-relinking [33].

Currently covered customers were grouped into about 70,000 exchanges. Each exchange was a p-median user and each of 1035 PoPs was a p-median facility. The distance, or cost, between a user and a facility was defined to be the network cost,



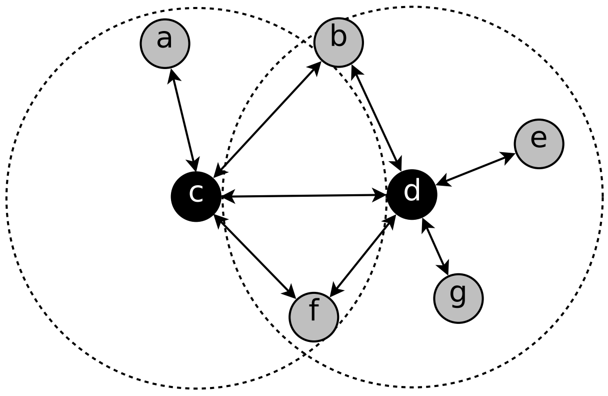


FIGURE 1. Two PoP placement solution are compared. The selected PoPs are shown in black, the covered areas are grey, and the uncovered areas are white. The area of coverage is indicated by the dashed circle around the PoP. The bottom solution ($\{c, d\}$) covers all of the areas while the solution on top ($\{b, c\}$) leaves two areas (e and g) uncovered.

i.e. the product of the PoP rate and the number of hours used by the exchange. Solving the p-median with $p = 1035$ resulted in the network cost with no PoP eliminated. We wanted the smallest value of p that preserved this cost. By solving a series of p-median problems with decreasing values of p, we determined that over 30% of the PoPs could be eliminated while maintaining the same coverage and not increasing the network cost.

3. Traffic routing

Routing of traffic is perhaps the most critical operational problem in telecommunication networks. We consider here two routing problems, the routing of virtual private circuits and Internet traffic routing.

3.1. **Routing of private virtual circuits.**

Telecommunication service providers offer virtual private networks to customers by provisioning a set of permanent (long-term) private virtual circuits (PVCs) between endpoints on a large backbone network. During the provisioning of a PVC, routing decisions are made either automatically by the switch (or router) or by the network designer, through the use of preferred routing assignments and without any knowledge of future requests. Over time, these decisions usually cause inefficiencies in the network and occasional rerouting of the PVCs is needed. The new routing scheme is then implemented on the network through preferred routing assignments. Given a preferred routing assignment, the network will move the PVC from its current route to the new preferred route as soon as this move becomes feasible.

One possible way to create the preferred routing assignments is to appropriately order the set of PVCs currently in the network and apply an algorithm that mimics the routing algorithm used by the switch (or router) to each PVC in that order. However, more elaborate routing algorithms, that take into account factors not considered by the switch, could further improve the efficiency of network resource utilization.

Typically, the routing scheme used to automatically provision PVCs is also used to reroute them in case of network failures, such as trunks (that transport traffic between routers) or cards (located on routers). Therefore, this routing algorithm should be efficient in terms of running time, a requirement that can be traded off for improved network resource utilization when building preferred routing assignments offline.

We solved this problem with variants of a GRASP with path-relinking algorithm for the problem of routing offline a set of PVC demands over a backbone network, such that a combination of the delays due to propagation and congestion was minimized [28, 31]. This problem and its variants are also known in the literature as bandwidth packing problems. The set of PVCs to be routed can include all or a subset of the PVCs currently in the network, and/or a set of forecast PVCs. The explicit handling of propagation delays, as opposed to just handling the number of hops (as in the routing algorithm implemented in some switches) is particularly important in international networks, where distances between backbone nodes vary considerably. The minimization of network congestion is important for providing the maximum flexibility to handle the following situations:

- overbooking, which is typically used by network designers to account for non-coincidence of traffic;
- PVC rerouting, due to link or card failures; and
- bursting above the committed rate, which is not only allowed but sold to customers as one of the attractive features of some services.

3.2. **Routing of Internet traffic.**

3.2.1. *Intradomain routing.*

The Internet is divided into many routing domains, called autonomous systems (ASes). ASes are networks that consist of routers and links connecting the routers. When customer and peer routers are considered, these ASes can have thousands of routers and links. ASes interact to control and deliver Internet Protocol (IP) traffic. They typically fall under the administration of a single institution, such as a company, a university, or a service provider. Neighboring ASes use the Border Gateway Protocol (BGP) to route traffic.

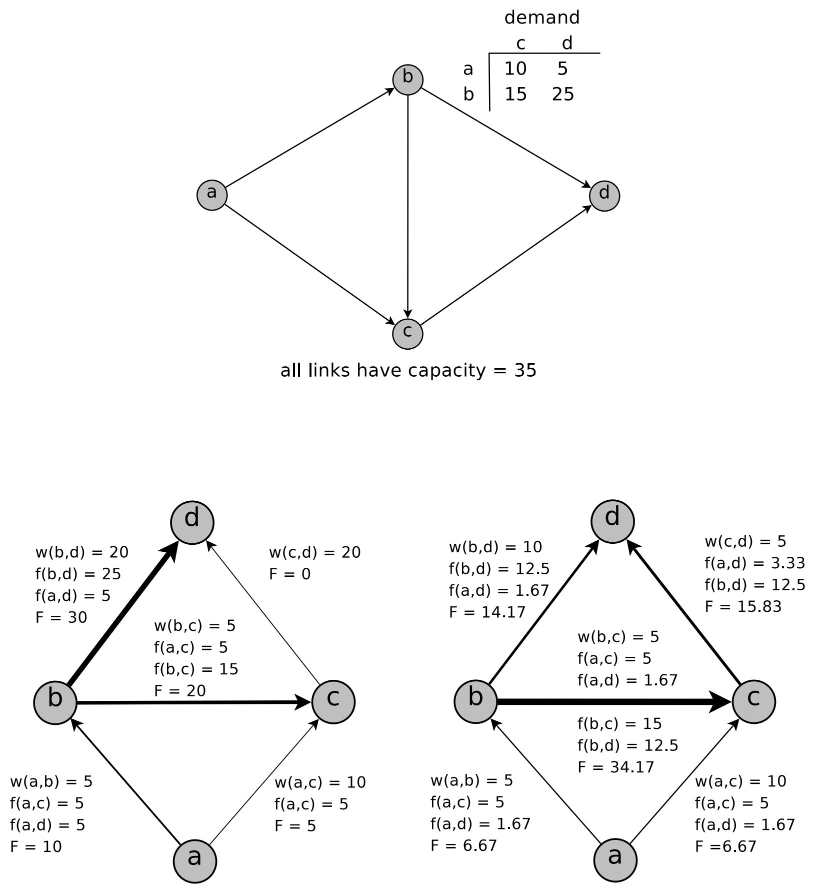


FIGURE 2. Example of OSPF weight setting. The demands shown next to the network on top of the figure must be routed. All links have capacity 35. Two solutions are shown on the bottom of the figure. $w(u, v)$ is the OSPF weight of link (u, v), the $f(x, y)$ indicated on each link is the amount of traffic originated at router x and going to router y that passes through the link, and F is the total traffic on the link. The solution on the left is better than the one on the right. It has a maximum utilization of 86% (link (b, d)), while the other has a maximum utilization of 98% (link (b, c)).

The goal of intradomain traffic engineering consists in improving user performance and making more efficient use of network resources within an AS. Interior Gateway Protocols (IGPs) such as OSPF (Open Shortest Path First) and IS-IS (Intermediate System-Intermediate System) are commonly used to select the paths along which traffic is routed within an AS. These routing protocols direct traffic based on link weights assigned by the network operator. Each router in the AS computes shortest paths and creates destination tables used to direct each IP packet to the next router on the path to its final destination. OSPF calculates routes as follows. Each link is assigned an integer weight ranging from 1 to 65535. The weight of a path is the sum of the link weights on the path. OSPF mandates

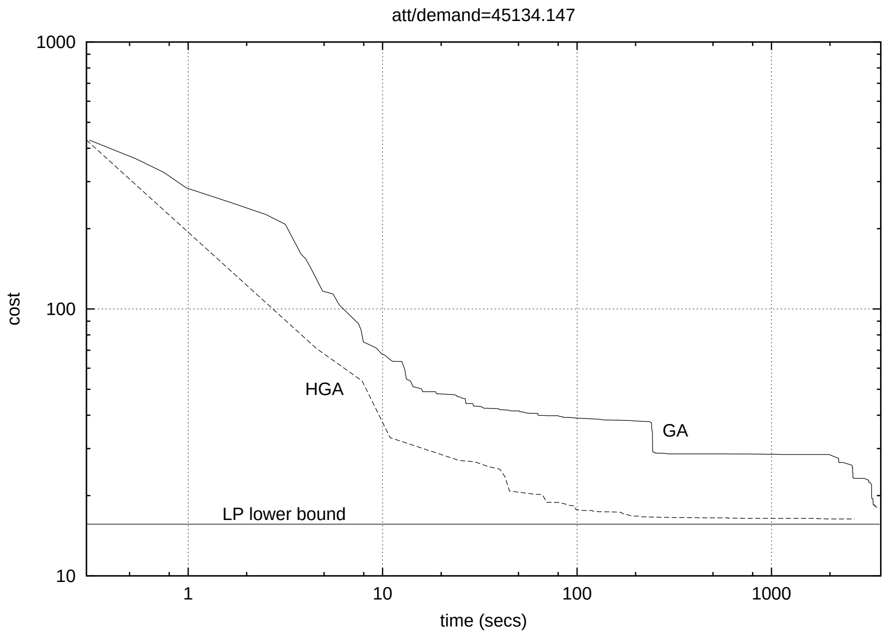


FIGURE 3. Congestion cost comparison of genetic (GA) and hybrid genetic (HGA), or memetic, algorithms on a 1-hour run on an IP network with 90 routers and 274 links. HGA converges faster to a better solution than the pure GA. LP lower bound is obtained by solving a multi-commodity flow problem where traffic is allowed to take any route.

that each router compute a graph of shortest paths with itself as the root. This graph gives the least weight routes (including multiple routes in case of ties) to all destinations in the AS. In the case of multiple shortest paths originating at a router, OSPF is usually implemented so that it will accomplish load balancing by splitting the traffic flow over all shortest paths leaving from each router. OSPF requires routers to exchange routing information with all the other routers in the AS. Complete network topology knowledge is required for the computation of the shortest paths.

Given a set of traffic demands between origin-destination pairs, the *OSPF weight setting problem* consists in determining weights to be assigned to the links so as to optimize a cost function, typically associated with a network congestion measure.

We proposed two solution methods for this problem: a genetic algorithm [14] and a hybrid genetic, or memetic, algorithm [8] incorporating a local improvement procedure to the crossover operator of the genetic algorithm proposed in [14]. The local improvement procedure makes use of an efficient dynamic shortest path algorithm [9] to recompute shortest paths after the modification of link weights. The memetic algorithm improved upon the pure genetic algorithm, producing better-quality solutions in less time as can be seen in Figure 3. The memetic algorithm was also shown to be more robust than the local search based approach of [16, 17].

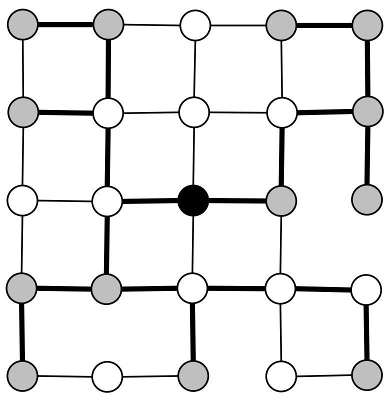
3.2.2. *Interdomain routing.* The Internet's two-tiered routing architecture was designed to have a clean separation between the intradomain and interdomain routing protocols. For example, the interdomain routing protocol allows routers at the border of the AS to learn how to reach external destinations, whereas the intradomain protocol determines how to direct traffic from one router in the AS to another. However, the appropriate roles of the two protocols becomes unclear when the AS learns routes to a destination at multiple border routers – a common situation today. Since service providers peer at multiple locations, essentially all of the traffic from customers to the rest of the Internet has multiple egress routers (routers in which traffic leaves the AS). In addition, many customers connect to their provider in multiple locations for fault tolerance and more flexible load balancing, resulting in multiple egress routers for these destinations as well. Selecting among multiple egress points is now a fundamental part of the Internet routing architecture, independent of the current set of routing protocols.

In the Internet today, border routers learn routes to destination prefixes via BGP. When multiple border routers have routes that are "equally good" in the BGP sense (e.g., local preference, AS path length, etc.), each router in the AS directs traffic to its *closest* border router, in terms of the IGP distances. This policy of *early-exit* or *hot-potato* routing is hard-coded in the BGP decision process implemented on each router, offering consistent forwarding of packets.

Although consistent forwarding is clearly an important property for any routing system, we [34] believe that hot-potato routing is disruptive and convoluted. Small changes in IGP distances can sometimes lead to large shifts in traffic, long convergence delays, and BGP updates to neighboring domains. Network administrators are forced to evaluate the impact of changes in the IGP metrics on BGP routing decisions, rather than viewing the two parts of the routing system separately.

Selecting the egress point and computing the forwarding path to the egress point are two very distinct functions, and we believe that they should be decoupled. Paths *inside* the network should be selected based on some meaningful performance objective, whereas *egress selection* should be flexible to support a broader set of traffic-engineering goals. These objectives vary by network and destination prefix; therefore a mechanism that imposes a single egress selection policy cannot satisfy this diverse set of requirements.

In [34], we propose a new mechanism for each router to select an egress point for a destination, by comparing the candidate egress points based on a weighted sum of the IGP distance and a constant term. The configurable weights provide flexibility in deciding whether (and how much) to base BGP decisions on the IGP metrics. Network-management systems apply linear and integer programming techniques to automatically set these weights to satisfy network-level objectives, such as balancing load and minimizing propagation delays. Our new mechanism is called TIE (Tunable Interdomain Egress) because it controls how routers break ties between multiple equally-good BGP routes. Our solution does not introduce any new protocols or any changes to today's routing protocols, making it possible to deploy our ideas at one AS at a time and with only minimal changes to the BGP decision logic on IP routers.



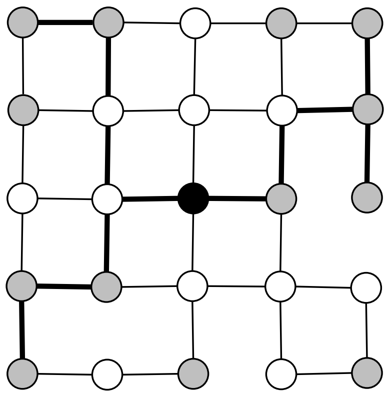


FIGURE 4. Two fiber networks are compared. Backbone node is shown in black, potential revenue generating premises are shown in grey, street corners and zero revenue producing premises are shown in white. Deployed fiber is shown in bold. The top network provides service to all potential revenue generating premises while bottom network provides service only to profitable premises.

4. Design problems

Network design problems are among the most important applications of optimization in telecommunications. We describe two instances of design problems that we have worked on. In the first, we address the optimization of the tradeoff between revenue generation by a network and the cost of building the network. In the second, we consider survivable IP network design.

4.1. **To lay or not to lay fiber.** Telecommunication service providers often must decide whether an investment to lay optical fiber cable to provide broadband service to customers is worthwhile. There is a cost associated with laying fiber which depends not only on the total length of the fiber but also on where and how the fiber is laid. For example, underground fiber could be more expensive than overground fiber.

Given a geographic area made up of customer premises and street segments connecting these premises, an estimate of the present value of potential revenue that could be earned from each customer premise, and the present value of the cost of laying fiber in each street segment, a service provider would like to maximize the difference between the revenue earned from customers reached by the fiber and the total cost of the fiber. An objective function value above some specified threshold would indicate the feasibility of the investment.

This type of problem, called the *prize-collecting Steiner tree problem*, can be used to order the attractiveness of different markets when rolling out services such as broadband Internet access, IPTV, and voice over IP. In fact, our motivation for studying this problem was the Telecommunications Act of 1996 which opened up the telecommunications markets in the U.S. and allowed AT&T to compete in local markets. With so many markets to choose to compete in, it was important to determine which were the most attractive.

Work on implementing the approximation algorithm of Goemans and Williamson [20] was done by Johnson et al. [24] and we were interested in accessing the quality of the solutions found with the approximation algorithm. In [25], we proposed a cutting planes algorithm to produce strong bounds for the prize collecting Steiner tree problem and showed that most solutions found by the approximation algorithm had a gap with respect to the bounds. In [11], we introduced a new type of GRASP, where randomized construction is done by perturbing the data and applying an approximation algorithm (in this case, the Goemans and Williamson algorithm) on the perturbed problem. In addition to the construction, our heuristic consisted of local search, path-relinking, and a variable neighborhood search post-optimization phase. The GRASP heuristic applies the approximation algorithm at least one time using the original data. Consequently, solutions that it produces are always at least as good as those found by the approximation algorithm. On 84.2% of 114 benchmark test problems, solutions produced by the GRASP heuristic were better than those produced by the approximation algorithm alone. On 91.2% of the instances the solutions found by the GRASP heuristic were provably optimal.

4.2. **Survivable IP network design.** With the pervasiveness of IP networks in telecommunications, an important question faced by network operators is how to design robust cost-efficient networks on which traffic will be routed with OSPF. Given a network topology (i.e., a set of nodes and a set of arcs where links can be installed), predicted traffic demands, a set of link types to be deployed, each having a different capacity and installation cost per mile, the *survivable IP network design problem* consists in finding a set of OSPF arc weights and the number of each link type deployed on each arc such that network cost is minimized. We further require that in a no-failure or any node/arc failure situation there is enough installed capacity to move all of the predicted traffic.

In [10], we proposed a genetic algorithm to find cost-efficient solutions for this problem for the case in which there is a single link type and number of copies of the link as well the link OSPF weights have to be determined. Since real networks can be built using many different link types (e.g., OC3, OC12, OC48, OC196) having different capacities as well as costs, in [4], we extended the design algorithm described in [10] to handle different link types.

5. Network migration

Network migration arises when traffic is moved from an outdated network to a new network. We consider in this section two applications of network migration: migration of phone traffic from a 4ESS switch based network to a router based IP network and telephone migration from an old PBX switch to a new PBX switch.

5.1. Voice traffic migration.

Consider the problem where inter-nodal traffic from an outdated telecommunications network is to be migrated to a new network. Nodes are migrated, one at each time period, from the old to the new network. All traffic originating or terminating at a given node in the old network is moved to a specific node in the new network. Routing is predetermined on both networks and therefore arc capacities are known. Traffic between nodes in the same network is routed in that network.

Nodes are migrated, one at a time, in some predetermined order. When a node is migrated, one or more temporary arcs may need to be set up since the node in the new network to which the traffic is migrated may be adjacent to one or more still active nodes in the old network. A temporary arc remains active until both nodes connected by the arc are migrated to the new network.

In one version of the *network migration scheduling problem*, one seeks an ordering of the migration of the nodes that minimizes the maximum sum of capacities of the temporary arcs. In another version, the objective is to minimize the sum of the total capacities of the temporary arcs over each period in the planning horizon.

We were motivated to look at this problem when AT&T began planning the migration of its switch-based telephone traffic to a new router-based IP network. In [6], we present a GRASP with evolutionary path-relinking for these two variants of the migration problem.

5.2. PBX telephone migration.

A PBX, or *private branch exchange*, is a private telephone network such as call forwarding, call recording, call transfer, and voice messaging.

Some PBX features require groups of phone numbers to be defined. These include, for example, multi-line hunt (MLH), call pickup (CPU), intercom (ICOM), series completion (SC), and shared telephone number (STN) groups. An MLH group consists of a cycle of phone numbers. When a call is made to a phone in the cycle and the call is not answered, it is transfered to the next phone in the cycle. This is repeated until someone picks up. A CPU group is a set of phone numbers where any phone in the group can pickup a call made to any other phone in group. Any phone in an ICOM group can speed dial to any other group member. A SC group is an ordered list of phone numbers. If a call made to the first phone is not answered, it is transfered to the next. This is repeated until someone picks up. If the last phone in the list does not pick up, voice mail answers the call. An STN group is a set of phone numbers for which calls made to them are answered by a single phone (e.g. an assistant). In an enterprise there may exist several MLH, CPU, ICOM, SC, and STN groups and a single phone number may be a member of more than one group.

We consider a problem that arises when an enterprise acquires a new PBX to replace an existing one. Phone numbers need to migrate from the old system to the new system over a given time horizon. Each group has a penalty associated with it. If two phones in a given group migrate in different time periods, then a

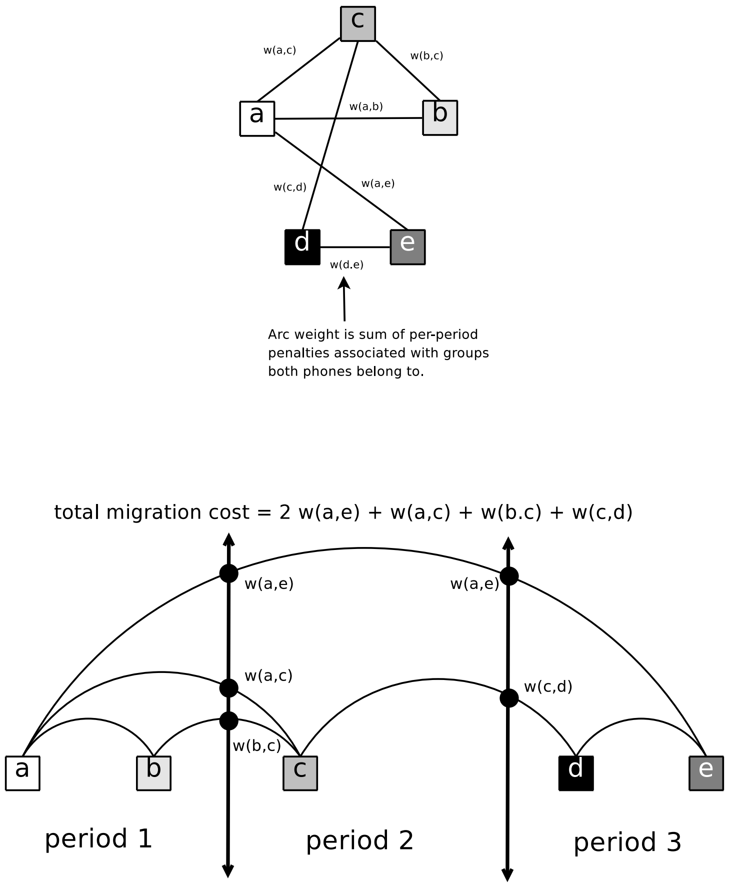

c
w(a,c)
w(b,c)
a
w(a,b)
b
w(c,d)
w(a,e)
d
e
w(d,e)
Arc weight is sum of per-period penalties associated with groups both phones belong to.
total migration cost = 2 w(a,e) + w(a,c) + w(b.c) + w(c,d)
w(a,e)
w(a,e)
w(a,c)
w(c,d)
w(b,c)
a
b
c
d
e
period 1
period 2
period 3

FIGURE 5. Example of PBX telephone migration. Five telephones are to migrate over a three period horizon. In each period at most two telephones can migrate. In the migration schedule shown, telephones a and b are migrated in period 1, telephone c is migrated in period 2, and telephones d and e are migrated in period 3. Penalties $w(a,b)$ and $w(d,e)$ do not contribute to the cost since both a and b migrate in period 1 and d and e both migrate in period 3. Since a-c, b-c, and d-e migrate in consecutive periods, their penalties are contributed once to the total cost. Since a and e are scheduled two periods apart, their penalty contributes twice to the total cost.

penalty is incurred. This penalty depends on the set of groups that these phones both belong to as well as the amount of time between the migrations of each phone. We further require that during each time period a specified maximum number of phones are allowed to migrate and assume that there are sufficient periods in the planning horizon to allow for a feasible schedule.

The objective is to schedule the migration plan so that the total migration penalty is minimized. This involves assigning phone numbers to time periods such that no more than the maximum number of phones are assigned to a single period.

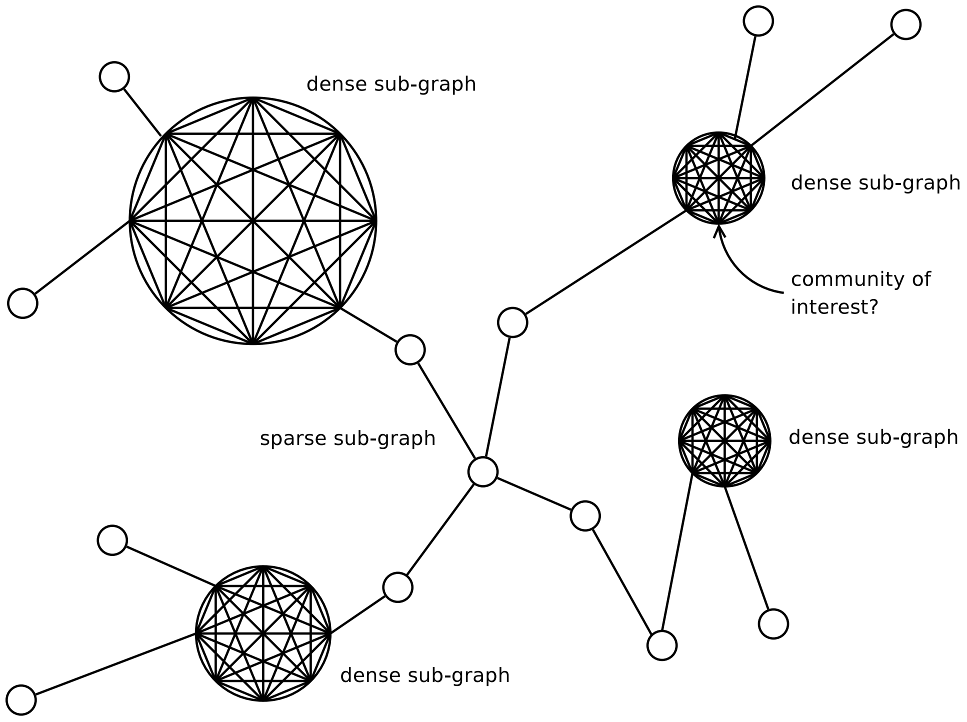


FIGURE 6. Macro structure of call detail graph. Graph is very sparse with spread-out clusters of very dense sub-graphs. These dense clusters can be interpreted as *communities of interest.*

We learned of this problem when we were contacted by people implementing such a move for a large investment bank that acquired a new PBX from AT&T. This problem involved migrating over 2500 phones belong to one or more of about 400 groups. Since there was a limit of 375 phones that could be moved per period, the move could not be done in less than eight periods. Since moving phones sharing one or more groups in different time periods could cause a business disruption, we wanted to minimize any possible disruption caused by the migration. To do this we assigned different penalties to phone pairs sharing different groups. Those groups whose disruption would be the most critical had the largest penalties. In [5], we present a GRASP with three local neighborhood structures for this problem.

6. DATA MINING

The proliferation of massive data sets [2] brings with it a series of special computational challenges to the optimization community. This *data avalanche* arises in a wide range of scientific and commercial applications. With advances in computer and information technologies, many of these challenges are being addressed by diverse inter-disciplinary groups, that include computer scientists, mathematicians, statisticians and engineers, working in close cooperation with application domain experts.

In 1997, we began exploring the massive graphs associated with telephone calls. When we set out to study these graphs, we had no particular application in mind.

We were simply interested in investigating the structure of these graphs. At that time, each call made on the AT&T network generated a record of about 200 bytes with information that included the pair of phone numbers involved in the phone call. The set of all these records is the *call detail database*. About 250 million records were generated on an average day in 1997 with around 320 million on a busy day, corresponding to about 18 terabytes of data per year. Given a time window, a call detail graph can be extracted from the database. In this directed graph, each phone number is a node and for each call placed during the time window an edge exists from the calling number to the called number. Because of privacy concerns, the script that extracted the graph from the database mapped the phone numbers to integers from 1 to n, where n is the number of phones involved in calls.

We describe an experiment involving 12 hours of calls in 1997 [1]. The corresponding graph had 53,767,087 vertices and over 170 million edges. We found 3,667,448 connected components out of which only 302,468 were components of size greater than 3 (there were 255 self-loops, 2,766,206 pairs and 598,519 triplets).

A giant component with 44,989,297 vertices was detected. It is interesting to observe that this is similar to what is predicted by random graph theory even though the call graphs are certainly not random. The giant component has 13,799,430 directed depth first search trees (DFSTs) and one of them is a giant DFST (it has 10,355,749 vertices and 19,072,448 edges). Most of the DFSTs have no more than 5 vertices. The interesting trees have sizes between 5 and 100. Their corresponding induced subgraphs are most of the time very sparse, except for some occasional dense subgraphs with 11 to 32 vertices.

We argued that the largest clique in this component has size not greater than 32. Cliques are either within a subgraph induced by the vertices of a DFST, or distributed among the different DFSTs. We expected the former to occur. There are several large DFSTs, the largest having about 19 million edges. By counting the edges in the trees, one observes that there remain very few edges to go between trees and consequently it is more likely that cliques are within the graphs induced by the nodes of a tree. Since the largest dense subgraph induced by the vertices of a tree had 32 vertices, we did not expect many cliques larger that 32 to be found.

To begin our experimentation, we considered 10% of the edges in the large component from which we recursively removed all vertices of degree one. This resulted in a graph with 2,438,911 vertices and 5,856,224 edges, which fit in memory. In this graph we searched for large cliques with a GRASP for maximum clique. Our first motivation was to identify a lower bound on the size of the maximum clique so that we could delete higher-degree vertices on larger portions of the graph to possibly identify larger cliques. The GRASP was repeated 1000 times, with each iteration producing a locally maximal clique. Though applying local search on every constructed solution may not be efficient from a running time point of view, we applied local search to all constructed solutions to explore its effect in improving clique sizes. Because of the independent nature of the GRASP iterations and since our computer was configured with 20 processors, we created 10 threads, each independently running GRASP starting from a different random number generator seed.

Table 1 summarizes the first part of the experimental results. It shows, for each clique size found, the number of GRASP iterations that constructed or improved such solution, and from sizes 5 to 15, the number of distinct cliques that were

TABLE 1. Cliques found by `construct` and `local`

	cliques found by		distinct
size	`construct`	`local`	cliques
2	63	62	
3	473	320	
4	95	176	
5	73	103	14
6	116	95	11
7	59	38	25
8	54	63	28
9	22	33	14
10	17	10	9
11	15	38	35
12	10	32	22
13	1	26	18
14	0	3	3
15	0	1	1

found by the GRASP iterations. It is interesting to observe that these cliques, even though distinct, share a large number of vertices. Applying a greedy procedure to these cliques to identify a disjoint set of cliques produced one clique of size 15, 12, 9, and 7, four cliques of size 6, and five of size 5.

Next, we considered 25% of the edges in the large component from which we recursively removed all vertices of degree 10 or less. The resulting graph had 291,944 vertices and 2,184,751 edges. 12,188 iterations of GRASP produced cliques of size 26.

Having found cliques of size 26 in a quarter of the graph, we next intensified our search on the entire huge connected component. In this component, we recursively removed all vertices of degree 20 or less. The resulting graph had 27,019 vertices and 757,876 edges.

Over 20,000 GRASP iterations were carried out on the 27,019 vertex – 757,876 edge graph. Cliques of 30 vertices were found. These cliques are very likely to be optimal because we do not expect cliques larger than 32 vertices to be found. The local search can be seen to improve the constructed solution not only for large constructed cliques, but also for small cliques. In fact, in 26 iterations, constructed cliques of size 3 were improved by the local search to size 30.

Finally to increase our confidence that the cliques of size 30 found were maximum, we recursively removed all vertices of degree 30 or less, resulting in a graph with 8724 vertices and about 320 thousand edges. We ran 100,000 GRASP iterations on the graph taking 10 parallel processors about one and a half days to finish. The largest clique found had 30 vertices. Of the 100,000 cliques generated, 14,141 were distinct, although many of them shared one or more vertices.

Quasi-cliques are dense sub-graphs, i.e. they are cliques with a few missing edges. To compute quasi-cliques [3] on this test data, we looked for large quasi-cliques with densities $90\%, 80\%, 70\%$, and 50%. Quasi-cliques of sizes 44, 57, 65, and 98, respectively, were found.

It was surprising to us that in only 12 hours of phone calls we found groups of 30 phone numbers where each one either called or was called by all 29 others and groups of 98 where each called or was called by at least half of the other phones. To date, this is the research that we have been involved with that has received the most attention from the media [7, 12, 13, 22, 23].

7. Concluding remarks

In this article, we show a sample of optimization problems that arise in an optimization research department at a telecommunications service provider. Most of the interesting questions we see are NP-hard combinatorial optimization problems. Though we make use of linear and integer programming solvers in many instances, for most cases we use metaheuristics. Metaheuristics, such as GRASP with path-relinking [32], are widely applicable, produce cost-efficient solutions, are relatively easy to implement, and therefore can quickly provide good-quality solutions to problems that arise in practice.

References

[1] J. Abello, P. M. Pardalos, and M. G. C. Resende. On maximum cliques problems in very large graphs. In J. Abello and J. Vitter, editors, *External memory algorithms*, volume 50 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, pages 119–130. American Mathematical Society, 1999.

[2] J. Abello, P. M. Pardalos, and M. G. C. Resende, editors. *Handbook of Massive Data Sets*. Kluwer, 2002.

[3] J. Abello, M. G. C. Resende, and S. Sudarsky. Massive quasi-clique detection. In S. Rajsbaum, editor, *LATIN 2002: Theoretical Informatics*, volume 2286 of *Lecture Notes in Computer Science*, pages 598–612. Springer Verlag, 2002.

[4] D. V. Andrade, L. S. Buriol, M. G. C. Resende, and M. Thorup. Survivable composite-link IP network design with OSPF routing. In *Proceedings of The Eighth INFORMS Telecommunications Conference*, 2006.

[5] D. V. Andrade and M. G. C. Resende. A GRASP for PBX telephone migration scheduling. In *Proceedings of The Eighth INFORMS Telecommunications Conference*, 2006.

[6] D. V. Andrade and M. G. C. Resende. GRASP with path-relinking for network migration scheduling. In *Proceedings of International Network Optimization Conference (INOC 2007)*, 2007.

[7] A.-L. Barabási. *Linked: How everything is connected to everything else and what it means for business, science, and everyday life*. Plume, 2003.

[8] L. S. Buriol, M. G. C. Resende, C. C. Ribeiro, and M. Thorup. A hybrid genetic algorithm for the weight setting problem in OSPF/IS-IS routing. *Networks*, 46:36–56, 2005.

[9] L. S. Buriol, M. G. C. Resende, and M. Thorup. Speeding up dynamic shortest-path algorithms. *INFORMS J. Comput.*, 2007. To appear.

[10] L. S. Buriol, M. G. C. Resende, and M. Thorup. Survivable IP network design with OSPF routing. *Networks*, 49:51–64, 2007.

[11] S. A. Canuto, M. G. C. Resende, and C. C. Ribeiro. Local search with perturbations for the prize-collecting Steiner tree problem in graphs. *Networks*, 38:50–58, 2001.

[12] B. Cipra. Massive graphs pose big problems. *SIAM NEWS*, 32, 1999.

[13] The Economist. Needles in giant haystacks. *The Economist*, page 74, January 30th – February 5th, 1999.

[14] M. Ericsson, M. G. C. Resende, and P. M. Pardalos. A genetic algorithm for the weight setting problem in OSPF routing. *J. of Comb. Optim.*, 6:299–333, 2002.

[15] T. A. Feo and M. G. C. Resende. Greedy randomized adaptive search procedures. *J. Global Optim.*, 6:109–133, 1995.

[16] B. Fortz and M. Thorup. Internet traffic engineering by optimizing ospf weights. In *Proc. 19th IEEE Conf. on Computer Communications (INFOCOM)*, pages 519–528, 2000.

[17] B. Fortz and M. Thorup. Increasing internet capacity using local search. *Computational Optimization and Applications*, 29(1):13–48, 2004.

[18] F. Glover. A template for scatter search and path relinking. In *AE '97: Selected Papers from the Third European Conference on Artificial Evolution*, pages 3–54, London, UK, 1998. Springer-Verlag.

[19] F. Glover and G. Kochenberger, editors. *Handbook of Metaheuristics.* Kluwer, 2003.

[20] M. X. Goemans and D. P. Williamson. The primal dual method for approximation algorithms and its application to network design problems. In D. Hochbaum, editor, *Approximation algorithms for NP-hard problems*, pages 144–191. PWS Publishing Co., 1996.

[21] P. Hansen and N. Mladenović. Variable neighborhood search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, pages 145–184. Kluwer, 2003.

[22] B. Hayes. Computing science graph theory in practice: Part I. *American Scientist*, 88:9–13, 2000.

[23] B. Hayes. Connecting the dots. *American Scientist*, 94:400–404, 2006.

[24] D. S. Johnson, M. Minkoff, and S. Phillips. The prize collecting steiner tree problem: Theory and practice. In *Proc. 11th Ann. ACM-SIAM Symp. on Discrete Algorithms*, pages 760–769. ACM-SIAM, 2000.

[25] A. Lucena and M. G. C. Resende. Strong lower bounds for the prize collecting Steiner problem in graphs. *Discrete Appl. Math.*, 141:277–294, 2004.

[26] P. Moscato and C. Cotta. A gentle introduction to memetic algorithms. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, pages 105–144. Kluwer, 2003.

[27] C. Reeves. Genetic algorithms. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*, pages 55–82. Kluwer, 2003.

[28] L. I. P. Resende and M. G. C. Resende. A GRASP for frame relay permanent virtual circuit routing. In P. Hansen and C. C. Ribeiro, editors, *Proceedings of the III Metaheuristics International Conference (MIC99)*, pages 397–401, 1999.

[29] M. G. C. Resende. Computing approximate solutions of the maximum covering problem using GRASP. *J. Heuristics*, 4:161–171, 1998.

[30] M. G. C. Resende and P. M. Pardalos, editors. *Handbook of Optimization in Telecommunications*. Springer, 2006.

[31] M. G. C. Resende and C. C. Ribeiro. A GRASP with path-relinking for private virtual circuit routing. *Networks*, 41:104–114, 2003.

[32] M. G. C. Resende and C. C. Ribeiro. GRASP with path-relinking: Recent advances and applications. In T. Ibaraki, K. Nonobe, and M. Yagiura, editors, *Metaheuristics: Progress as real problem solvers*, pages 29–63. Springer, 2005.

[33] M. G. C. Resende and R. F. Werneck. A hybrid heuristic for the p-median problem. *J. Heuristics*, 10:59–88, 2004.

[34] R. Teixeira, T. G. Griffin, M. G. C. Resende, and J. Rexford. TIE Breaking: Tunable Interdomain Egress Selection. *IEEE/ACM Trans. on Networking*, 15:761–774, 2007.

(M.G.C. Resende) Algorithms and Optimization Research Department, Internet and Network Systems Research, AT&T Labs Research, Florham Park, NJ, USA.

E-mail address: `mgcr@research.att.com`