Randomized Metarounding

Robert Carr [*]
Sandia National Labs
Albuquerque NM 87185 USA
Email: bobcarr@cs.sandia.gov

Santosh Vempala[†]
Dept. of Mathematics, M.I.T.
Cambridge MA 02139 USA
Email: vempala@math.mit.edu

December 3, 2001

Abstract

Let P be a linear relaxation of an integer polytope Z such that the integrality gap of P with respect to Z is at most r, as verified by a poly-time heuristic A, which on any positive cost function c returns an integer solution (an extreme point of Z) whose cost is at most r times the optimal cost over P. Then for any point x^* in P (a fractional solution), rx^* dominates some convex combination of extreme points of Z. A constructive version of this theorem is presented here, with applications to approximation algorithms, and can be viewed as a generalization of randomized rounding.

[*]Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

[†]Supported by NSF CAREER award CCR-9875024.

1 Introduction

Randomized rounding has become a standard technique in the design of approximation algorithms for NP-hard optimization problems, especially for *packing* and *covering* problems. These are problems that can be stated as integer linear programs (IP's) of the form $\max cx$ subject to $Ax \leq b$ or $\min cx$ subject to $Ax \geq b$ where each component of x is 0-1 valued and the entries of A, b and c are all non-negative. Given say, a minimization (covering) problem of this form, the approach is to first solve the linear programming relaxation of the integer program, namely

$$\begin{array}{rcl} \min & cx & \\ Ax & \geq & b \\ x & \geq & 0 \\ x & \leq & 1 \end{array}$$

Let x^* be the solution obtained by solving the linear program. The second step is to round this solution to a 0-1 solution as follows: for each variable x_e, set x_e to 1 with probability x_e^* and set it to zero with probability $1 - x_e^*$. The resulting integral setting is not necessarily a solution to the original problem, but it has several nice properties. In particular, the expected cost is cx^* and the expected value of the left hand side of each inequality is the same as the value for x^*.

1.1 Path congestion.

In many combinatorial situations, it is absolutely necessary to satisfy some (or even all) of the given constraints (e.g. a given subset of vertices must be connected in a solution, a solution must be a tour etc.). In such cases the method is not generally useful. However there are some lucky circumstances when randomized rounding can be applied to yield a true solution. This is best illustrated by the problem of *routing with minimum congestion* — we are given a graph, $G = (V, E)$, and pairs of vertices (s_i, t_i) for $i = 1, \ldots, k$. A solution to the problem is a set of k paths such that the i'th path connects s_i and t_i. The congestion of an edge is the number of paths that use the edge, and the congestion of a solution is the maximum congestion over all the edges. The goal is to find a solution with the minimum possible congestion. Given an instance of the problem, one can write down an integer linear program that ensures that there is a path between each pair s_i, t_i, and which minimizes the congestion. The integer program has variables x_e^i, one for each edge e and each pair (s_i, t_i). These variables are constrained to be 0 or 1 and support a flow of one unit from each s_i to each t_i. The linear programming relaxation of this IP, obtained by relaxing the integrality constraint to $0 \leq x_e^t \leq 1$ for all the variables, is a multicommodity flow problem that sets up a flow of value 1 between each s_i and t_i and minimizes the *fractional congestion*, i.e. the maximum, over all the edges, of the total flow through an edge. Any solution to this LP can be decomposed into a flow of value 1 between each pair s_i, t_i. These flows have a very useful property: the flow between s_i and t_i can be *separately* decomposed into a set of paths, each with some fractional flow, so that the sum of the fractional flows is 1. To do this, consider a particular pair, (s_i, t_i) and find a path P_1 from s_i to t_i with nonzero flow. Let λ_1 be the maximum flow that can be sent from s_i to t_i along this path. Reduce all the capacities on the path by λ_1 (which results in at least one edge being deleted entirely) and repeat to get a set of paths whose total flow is 1. Viewing the flow as a vector in $\mathbf{R}^{|E|}$, and each path as a 0-1 vector in $\mathbf{R}^{|E|}$, this property can be restated as

The flow f_i between s_i and t_i can be expressed as a convex combination of paths between s_i and t_i, i.e.

$$f_i = \sum_j \lambda_j \cdot \chi^{P_j}$$

Here f_i is the flow vector for the i'th pair, and χ^{P_j} is the incidence vector of edges on the j'th path P_j. Given such a path decomposition of the flow between s_i and t_i, randomized rounding picks one path according to the probability distribution given by the flow on the paths, i.e. the λ_j's. Doing this for every s_i, t_i pair gives us a solution to the problem. It is easy to check that the expected congestion of any edge is exactly its fractional congestion. Further, since the choice of the path for each s_i, t_i pair is made independently, a Chernoff bound implies that with high probability the congestion of *every* edge is within a constant times the expected value plus $O(\log n)$, where n is the number of vertices in the graph [5].

1.2 Multicast congestion.

Now consider the problem of finding trees (instead of paths) that span specified subsets of vertices (instead of only pairs) so as to minimize the net congestion. This is the *multicast congestion* problem and we will use it throughout the paper to illustrate the new techniques. We are given a graph $G = (V, E)$ and subsets of vertices $S_1, \ldots, S_m \subseteq V$, called multicast requests. The goal is to find a set of m trees so that the i'th tree spans the vertices of the i'th subset and the maximum number of trees that use any single edge is minimized [1]. As in the case of paths, one can write down an LP relaxation of the problem and obtain a fractional solution, but it is no longer obvious how to round this into a set of trees with low congestion. The LP solution itself can be decomposed into fractional solutions for each multicast request. If the fractional solution for each multicast could be written as a convex combination of trees, this would give us a simple rounding algorithm: For each multicast separately, (i) Find a convex combination of trees that equals the fractional solution and (ii) Pick one tree with probability equal to its convex multiplier. Then the expected congestion on an edge is exactly its fractional congestion. Further, the process is independent for each multicast, and so the deviation from the expectation can be bounded, and we get a solution that is within a constant times OPT plus $O(\log n)$, where OPT denotes the value of the optimum integral solution.

Of course, the proposed algorithm hinges on being able to express the fractional solution to a single multicast as a convex combination of trees. Unfortunately, this is impossible. If we could express any fractional solution as a combination of trees, then we could use this to solve the Steiner tree problem optimally. The solutions to a single multicast are Steiner trees spanning the vertices of the multicast. So we could solve an LP relaxation that minimizes the total cost of edges in the solution. Then we can write the solution obtained as a convex combination of trees. The cost of one of the trees must be at most the average cost, which is the value of the fractional solution. Thus we find one tree with cost at most the optimal cost. But the Steiner tree problem is NP-hard. Indeed any LP relaxation (that can

[1] In [6], Raghavan and Thompson present an algorithm with an exponential (in the maximum size of a single multicast) time bound that finds a solution of congestion $O(OPT + \log n)$. One consequence of the main theorem in this paper is a poly-time algorithm with the same quality.

be solved in polytime) must have an *integrality gap* unless $P = NP$. The integrality gap of a relaxation P with respect to an integer polytope Z (both in the positive orthant for the purpose of this paper) is the maximum possible value of the ratio

$$\frac{\max_{x \in P} c^T x}{\max_{x \in Z} c^T x}$$

over nonnegative, nonzero vectors c.

However, the standard LP relaxation of the Steiner tree problem is known to have an integrality gap of at most 2. So while it is impossible to express any fractional solution as a convex combination of integral solutions, it could be possible to express TWICE any fractional solution as a convex combination of integral solutions. Indeed, a recent result of [2] says that if the integrality gap of a relaxation is at most r, then for any fractional solution x^*, the vector rx^* *dominates* (i.e. is greater than or equal in every coordinate) some convex combination of integral solutions. If this convex decomposition could be found in polytime, then we would have a polytime algorithm for the multicast congestion problem that is guaranteed to find a solution within a constant times OPT plus $O(\log n)$.

In this paper we present a polytime algorithm for finding such a convex decomposition. It is described in section 2 and is based on an application of the ellipsoid method. This leads to an improved approximation for the multicast congestion problem (the previous best approximation was a multiplicative $O(\log n)$ [7]).

1.3 Generalized congestion.

As the reader might have realized, the approach just described is applicable in a more general context, which we call *generalized congestion.* Suppose we are given a graph $G = (V, E)$ and a set of *requests*, $R_1, \ldots, R_m$. To satisfy the i'th request R_i we need to pick a subset of edges that satisfy some specified property P_i. So, for instance the first request might be for a Hamilton cycle, the second request might be for a Steiner tree, the third for a 2-edge connected subgraph and so on. We assume that for each i, the set of solutions to the i'th request can be modelled as an integer program and its LP relaxation, P_i is given to us (either explicitly or as a separation oracle [3]). We are also given a polynomial-time heuristic A_i which can be used to find a solution within r times the optimum of P_i for *any* positive cost function on the edges. The problem is to find a solution that satisfies each request and has the minimum possible congestion. The congestion of a solution is the maximum congestion among its edges; the congestion of an edge is the number of requests for which that edge appears in the requests' solution. The main results of this paper can be stated as follows:

Theorem 1 *There is a polytime algorithm that takes as input any instance of the generalized congestion problem and r-approximation heuristics for each of its requests, and finds a solution with congestion $2er \cdot OPT + 6\log n$.*

Theorem 2 *Given an LP relaxation P, an r-approximation heuristic A, and any solution x^* of P, there is a polytime algorithm that finds a polynomial number of integral solutions $z^1, z^2, \ldots$ of P such that*

$$rx^* \geq \sum_j \lambda_j \chi^{z^j}$$

where $\lambda_j \geq 0$ for all j, and $\sum_j \lambda_j = 1$.

2 The main tool: convex decomposition

The decomposition theorem we present in this section applies to a large class of problems which have integer programming formulations. Let us first define the class of problems for which the decomposition applies. A *positive integer program* is an integer program whose variables are constrained to be non-negative.

Definition 1 *A* min-IP (max-IP) *problem is a minimization (maximization) problem whose set of feasible solutions can be described by a positive integer program.*

The *traveling salesman problem* (TSP) is an example of a min-IP problem. The analysis that follows can also be applied, with slight modifications, to max-IP problems, but we will assume our problems are min-IP problems for now. Note that any min-IP (max-IP) problem has a linear programming relaxation obtained by relaxing the integrality constraints of the integer program defining the problem.

Definition 2 *A polynomial-time algorithm $\mathcal{A}$ is an r-*approximation heuristic *to a min-IP problem with linear programming relaxation $\mathcal{P}$, if, for any positive objective function, $\mathcal{A}$ finds a solution whose cost is at most r times the cost of the optimal solution over $\mathcal{P}$.*

For the purpose of the decomposition, a problem in our class takes as input the LP relaxation $\mathcal{P}$ of a positive integer program IP along with an r-approximation heuristic $\mathcal{A}$ and a feasible solution x^* of $\mathcal{P}$.

For a min-IP problem I, denote the integer polyhedron for the integer program by $\mathcal{Z}(I)$ or just Z. Let $\mathcal{P}(I)$ or just P denote the linear programming relaxation of $\mathcal{Z}(I)$ (as well as the polyhedron corresponding to the linear programming relaxation). Also denote the set of extreme points for a polyhedron P by $ext(P)$.

If we have an r-approximation heuristic $\mathcal{A}$ to a min-IP problem I, then clearly the integrality gap between $\mathcal{P}(I)$ and $\mathcal{Z}(I)$ is at most r. Theorem 2.1 of [2] says that if x^* is a feasible solution to the linear program $\mathcal{P}(I)$, then rx^* *dominates* a convex combination of extreme points of $\mathcal{Z}(I)$. That is, we have

$$rx^* \geq \sum_j \lambda_j x^j, \tag{1}$$

where $\sum_j \lambda_j = 1$, $\lambda_j \geq 0$ for all j, and each x^j is an extreme point of $\mathcal{Z}(I)$. We now show how to construct a set of x^j's satisfying (1) in polynomial-time.

2.1 Constructing a Convex Combination

Let x^* be feasible for $\mathcal{P}(I)$. List the elements of $ext(Z)$ as $(x^j \mid j \in J)$. Let E be the index set for the variables in $\mathcal{P}(I)$. For each non-negative objective function c, denote the solution returned by the r-approximation $\mathcal{A}$ by x^c.

In order to obtain (1), we wish to solve the following linear program.

$$\begin{array}{lllll}
\text{maximize} & \sum_{j\in J} \lambda_j & & & \\
\text{subject to} & & & & \\
 & \sum_{j\in J} \lambda_j x_e^j & \leq & r x_e^* & \forall e \in E \\
 & \sum_{j\in J} \lambda_j & \leq & 1 & \\
 & \lambda_j & \geq & 0 & \forall j \in J
\end{array} \tag{2}$$

As explained below, rx^* dominates a convex combination of points in $ext(Z)$ if and only if the optimal objective function value of (2) is 1. Moreover, the solution λ^* of (2) provides an explicit convex decomposition into points in $ext(Z)$. That is, with $J' := \{j \in J \mid \lambda_j^* > 0\}$, one can see that

$$rx^* \geq \sum_{j \in J'} \lambda_j^* x^j. \tag{3}$$

The sum in (3) is a linear combination of $\{x^j \mid j \in J'\} \subset ext(Z)$ which is dominated by rx^*. If $\sum_{j \in J'} \lambda_j^* = 1$, then this linear combination is in fact our desired convex combination. Note, however, that (2) has an exponential number of variables. We will demonstrate how to solve (2) in polynomial time by first solving its dual linear program. The dual linear program to (2) is the following.

$$\begin{array}{lllll} \text{minimize} & rx^* \cdot w + z & & & \\ \text{subject to} & & & & \\ & x^j \cdot w + z & \geq & 1 & \forall j \in J \\ & w_e & \geq & 0 & \forall e \in E \\ & z & \geq & 0 & \end{array} \tag{4}$$

Although (4) has an exponential number of constraints, we will be able to solve it in polynomial time using the ellipsoid method [3].

Lemma 1 *The linear program (4) has an optimal solution of 1.*

Proof. Since $w^* = 0$, $z^* = 1$ is feasible, the optimal solution to (4) is at most 1. Let $w^*, z^* \geq 0$ be given such that (4)'s objective function

$$rx^* \cdot w^* + z^* < 1.$$

We then have $rx^* \cdot w^* < 1 - z^*$. Since the integrality gap between $\mathcal{P}(I)$ and $\mathcal{Z}(I)$ is at most r (where the objective function is w^*), there exists a $j \in J$ such that $x^j \cdot w^* < 1 - z^*$. Hence, w^*, z^* violate the inequality

$$x^j \cdot w + z \geq 1$$

which makes this solution infeasible for (4). $\square$

Lemma 2 *Given an r-approximation heuristic $\mathcal{A}$, the linear program (4) can be solved in polynomial time using the ellipsoid method.*

Proof. Using the previous lemma, without loss of generality, we can add the constraint

$$rx^* \cdot w + z \leq 1$$

to (4). Let w^*, z^* be the center of the current ellipsoid. We need to check whether w^*, z^* is a feasible solution, and if not, find a hyperplane through the current center with the feasible region on one side. Suppose $w^*, z^* \geq 0$ are such that

$$rx^* \cdot w^* + z^* < 1.$$

We will use the r-approximation $\mathcal{A}$ as a separation oracle for finding an inequality of (4) that is violated by w^*, z^*. We have $rx^* \cdot w^* < 1 - z^*$. In fact, the integer solution x^{w^*} produced by $\mathcal{A}$ when the objective function is w^* must satisfy $x^{w^*} \cdot w^* < 1 - z^*$ since the cost of x^{w^*} must be within a factor of r of the cost of x^*. But, then w^*, z^* violates the inequality

$$x^{w^*} \cdot w + z \geq 1,$$

and we can use this to cut the current ellipsoid. On the other hand, if

$$rx^* \cdot w^* + z^* \geq 1$$

we will use the half-space

$$rx^* \cdot w^* + z^* \leq 1$$

to cut the current ellipsoid. Hence, (4) can be solved in polynomial-time using the ellipsoid method. □

Lemma 3 *Given an r-approximation heuristic, we can solve (2) in polynomial time.*

Proof. By lemma 2, we can solve (4) in polynomial time. The violated inequalities

$$x^j \cdot w + z \geq 1 \qquad \forall j \in J'$$

found by our separation oracle correspond to dual variables λ_j for all $j \in J' \subset J$ in (2). There is an optimal dual solution using only these variables, since the inequalities our separation oracle found are sufficient to determine that we have the optimal solution to (4). Furthermore, the number $|J'|$ of these dual variables is polynomial since our separation oracle was used only a polynomial number of times. Therefore, we can solve (2) by solving the smaller linear program that has only the λ_j variables for $j \in J'$. Since this smaller linear program is polynomial in size, it can be solved in polynomial time. □

Proof of Theorem 2. Lemma 3 and its proof show that we can solve (2) in polynomial time by solving a smaller polynomial size version of (2). If λ_j^* for $j \in J'$ is the optimal solution to the smaller version of (2), we have that rx^* dominates the linear combination

$$rx^* \geq \sum_{j \in J'} \lambda_j^* x^j \tag{5}$$

of elements x^j in $ext(Z)$. As pointed out in the previous proof, $|J'|$ is polynomial in magnitude, so there are a polynomial number of terms in (5). Since the optimal objective function value of (2) is 1 by Theorem 1 and duality, we have that

$$\sum_{j \in J'} \lambda_j^* = 1.$$

Hence, the linear combination (5) is our required convex combination, and can be found in polynomial time. □

3 The generic algorithm

In this section we present an algorithm for the generalized congestion problem. Let $A^i x \geq b^i$, $x \geq 0$ be the LP relaxation for the i'th request.

1. Formulate a single linear program $\hat{P}$ for the generalized congestion problem that minimizes the overall congestion, using the linear programs for each request as shown below. A different set of variables x_e^i

is used for each request i.

$$\begin{aligned} \min \quad & z \\ A^i x^i \ \geq \ & b^i \quad \forall i = 1, \ldots, m \\ \sum_i x_e^i \ \leq \ & z \quad \forall e \in E \\ x, z \ \geq \ & 0 \end{aligned}$$

2. Solve $\hat{P}$ to obtain a solution x^*.
3. Separate the solution into a solution for each request, i.e. x^{i*} for $i = 1, \ldots, m$.
4. For each request i, apply the convex decomposition algorithm of section 2 to x^{i*} and obtain a set of integral solutions and a convex combination of them that is dominated by rx^{i*}.
5. For each request, pick one integral solution with probability proportional to its convex multiplier.

Note that the second step above can be carried out even if the individual LP relaxations were given to us as separation oracles.

3.1 Examples

Our first example is the multicast congestion problem described in the introduction. In this problem each request is for a Steiner tree. Let the set of vertices specified by the i'th request be S_i. Then the request can be modelled by the following LP relaxation:

$$\sum_{e \in \delta(S)} x_e \ \geq \ 1 \quad \forall S \subset V \text{ s.t. } S \cap S_i \neq \phi, (V \setminus S) \cap S_i \neq \phi \tag{6}$$

$$x_e \ \geq \ 0 \quad \forall e \in E \tag{7}$$

There is a simple heuristic[2] that shows that the integrality gap of this relaxation is at most 2. Thus we have a 2-approximation heuristic for each request.

[2] Take the minimum spanning tree on the complete graph with vertex set S_i and edge weights given by the shortest paths on the original graph; replace each edge of the spanning tree by the shortest paths in the original graph [8].

There is a potentially stronger relaxation, called the *bidirected cut relaxation* originally proposed by Edmonds, whose integrality gap is unknown.

In step 4 of the algorithm, for each request i, we find a convex combination of Steiner trees that is dominated by $2x^{i*}$. In step 5 we pick one Steiner tree from the combination for each request. This gives us a solution to the multicast congestion problem. As a second example consider a situation where the first request is for a 2-edge connected spanning subgraph of the entire graph. The second request is for a path between two specified vertices s and t and third is for a cycle cover of the graph (i.e. a set of cycles such that each vertex is in some cycle). Then we can write down LP relaxations for each request. The 2-edge connected spanning subgraph problem has a relaxation whose integrality gap is known to be less than $\frac{3}{2}$ [1], while the path and cycle cover problems can be solved in polynomial-time, i.e. they have LP relaxations with integrality gaps equal to 1. In this case, each request has a $\frac{3}{2}$ or better approximation heuristic. Note that instead of just one request for each of the three types described above, we could have any number of requests of each type.

For all these examples, Theorem 1 guarantees that the congestion of the solution obtained is at most a constant times the OPT plus $O(\log n)$.

4 The performance guarantee

Let X_e^i be the random variable indicating whether an edge e is selected by the algorithm to satisfy the i^{th} request.

Lemma 4 $E[X_e^i] \leq r \cdot x_e^{i*}$.

Proof. The random variable X_e^i is 1 if the edge e is present in the integral solution selected for the i'th request and zero otherwise. The expected value of X_e^i is equal to the sum of selection probabilities of integral solutions that contain e. That is, $z^1, z^2, \ldots$ represent the possible integral solutions selected for the i'th request, then

$$\begin{aligned} E[X_e^i] &= \sum_{j:e\in z^j} Pr(z^j \text{ is selected for the i'th request}) \\ &= \sum_{j:e\in z^j} \lambda_j \\ &\le rx_e^{i*} \end{aligned}$$

The last inequality is due to the fact that rx^{i*} dominates the convex combination of z^j's and so each rx_e^{i*} is greater than or equal to the sum of the λ_j's corresponding to z^j's that contain e. □

Lemma 5 *With probability* $1 - \frac{1}{n}$*, the congestion of the solution found by the generic algorithm is at most* $2er \cdot OPT + 6 \log n$*.*

Proof. Fix an edge f. Let $C(f)$ denote the total congestion on the edge e. By definition, $C(f) = \sum_i X_f^i$. From Lemma 4, we can conclude that $E[C(f)] \le r \cdot \sum_i \{x_f^{i*}\} \le r \cdot OPT$. We need to show that it is unlikely that $C(f)$ deviates significantly from its expectation. For each request i, the random variables X_f^i are independent from random variables for other requests. As a consequence, we can apply a Chernoff bound ([4]). Set $C \ge \max\{2\mathrm{e}E[C(f)], 2\cdot(\alpha+2)\cdot\log n\}$ with $\alpha > 0$ denoting an arbitrary constant. Then

$$\mathrm{Prob}[C(f) \ge C] \;\le\; 2^{-C/2} \;\le\; n^{-\alpha-2} \quad .$$

Summing this probability over all edges yields that the congestion does not exceed $C = 2er \cdot OPT + 6 \log n$, with probability at least $1 - \frac{1}{n}$. □

The above lemma along with Theorem 2 implies Theorem 1.

References

[1] J. Cheriyan, A. Sebo and Z.Szigeti, *Improving on the 1.5-approximation of a smallest 2-edge connected spanning subgraph.* Proc. of IPCO 1998, in LNCS 1412.

[2] R. Carr and S. Vempala, *Towards a 4/3 approximation for the asymmetric traveling salesman problem.* Proc. of SODA 2000.

[3] M. Grötschel, L. Lovász and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 1987.

[4] T. Hagerup and C. Rüb, *A guided tour of Chernoff bounds*. Information Processing Letters 33(6):305-308, 1990.

[5] P. Raghavan and C.D. Thompson, *Randomized rounding: a technique for provably good algorithms and algorithmic proofs*. Combinatorica, 7(4):365-374, 1987.

[6] P. Raghavan and C.D. Thompson, *Multiterminal global routing: a deterministic approximation scheme*. Algorithmica, 6:73-82, 1991.

[7] S. Vempala and B. Voecking, *Approximating multicast congestion*. Proc. of ISAAC 1999.

[8] V. V. Vazirani, *Approximation Algorithms*, Springer 2001.