Randomized rounding

Within computer science and operations research, many combinatorial optimization problems are computationally intractable to solve exactly (to optimality). Many such problems do admit fast (polynomial time) approximation algorithms—that is, algorithms that are guaranteed to return an approximately optimal solution given any input.

Randomized rounding (Raghavan & Tompson 1987) is a widely used approach for designing and analyzing such approximation algorithms. The basic idea is to use the probabilistic method to convert an optimal solution of a relaxation of the problem into an approximately optimal solution to the original problem.

Overview

The basic approach has three steps:

1. Formulate the problem to be solved as an integer linear program (ILP).
2. Compute an optimal fractional solution x to the linear programming relaxation (LP) of the ILP.
3. Round the fractional solution x of the LP to an integer solution x' of the ILP.

(Although the approach is most commonly applied with linear programs, other kinds of relaxations are sometimes used. For example, see Goeman's and Williamson's semi-definite programming-based Max-Cut approximation algorithm.)

The challenge in the first step is to choose a suitable integer linear program. Familiarity with linear programming is required, in particular, familiarity with how to model problems using linear programs and integer linear programs. But, for many problems, there is a natural integer linear program that works well, such as in the Set Cover example below. (The integer linear program should have a small integrality gap; indeed randomized rounding is often used to prove bounds on integrality gaps.)

In the second step, the optimal fractional solution can typically be computed in polynomial time using any standard linear programming algorithm.

In the third step, the fractional solution must be converted into an integer solution (and thus a solution to the original problem). This is called *rounding* the fractional solution. The resulting integer solution should (provably) have cost not much larger than the cost of the fractional solution. This will ensure that the cost of the integer solution is not much larger than the cost of the optimal integer solution.

The main technique used to do the third step (rounding) is to use randomization, and then to use probabilistic arguments to bound the increase in cost due to the rounding (following the probabilistic method from combinatorics). There, probabilistic arguments are used to show the existence of discrete structures with desired properties. In this context, one uses such arguments to show the following:

> *Given any fractional solution x of the LP, with positive probability the randomized rounding process produces an integer solution x' that approximates x* according to some desired criterion.

Finally, to make the third step computationally efficient, one either shows that x' approximates x with high probability (so that the step can remain randomized) or one derandomizes the rounding step, typically using the method of conditional probabilities. The latter method converts the randomized rounding process into an efficient deterministic process that is guaranteed to reach a good outcome.

Comparison to other applications of the probabilistic method

The randomized rounding step differs from most applications of the probabilistic method in two respects:

1. The computational complexity of the rounding step is important. It should be implementable by a fast (e.g. polynomial time) algorithm.
2. The probability distribution underlying the random experiment is a function of the solution x of a relaxation of the problem instance. This fact is crucial to proving the performance guarantee of the approximation algorithm --- that is, that for any problem instance, the algorithm returns a solution that approximates the *optimal solution for that specific instance*. In comparison, applications of the probabilistic method in combinatorics typically show the existence of structures whose features depend on other parameters of the input. For example, consider Turán's theorem, which can be stated as "any graph with n vertices of average degree d must have an independent set of size at least $n/(d+1)$. (See this for a probabilistic proof of Turán's theorem.) While there are graphs for which this bound is tight, there are also graphs which have independent sets much larger than $n/(d+1)$. Thus, the size of the independent set shown to exist by Turán's theorem in a graph may, in general, be much smaller than the maximum independent set for that graph.

Set Cover example

The method is best illustrated by example. The following example illustrates how randomized rounding can be used to design an approximation algorithm for the Set Cover problem.

Fix any instance $\langle c, \mathcal{S} \rangle$ of the Set Cover problem over a universe $\mathcal{U}$.

For step 1, let IP be the standard integer linear program for set cover for this instance.

For step 2, let LP be the linear programming relaxation of IP, and compute an optimal solution x^* to LP using any standard linear programming algorithm. (This takes time polynomial in the input size.)

(The feasible solutions to LP are the vectors x that assign each set $s \in \mathcal{S}$ a non-negative weight x_s, such that, for each element $e \in \mathcal{U}$, x' *covers* e -- the total weight assigned to the sets containing e is at least 1, that is,

$$\sum_{s \ni e} x_s \geq 1.$$

The optimal solution x^* is a feasible solution whose cost

$$\sum_{s \in \mathcal{U}} c(S) x^*_s$$

is as small as possible.)

Note that any set cover $\mathcal{C}$ for $\mathcal{S}$ gives a feasible solution x (where $x_s = 1$for $s \in \mathcal{C}$, $x_s = 0$otherwise). The cost of this $\mathcal{C}$ equals the cost of x, that is,

$$\sum_{s \in \mathcal{C}} c(s) = \sum_{s \in \mathcal{S}} c(s) x_s.$$

In other words, the linear program LP is a relaxation of the given set-cover problem.

Since x^* has minimum cost among feasible solutions to the LP, *the cost of* x^* *is a lower bound on the cost of the optimal set cover.*

Step 3: The randomized rounding step

Here is a description of the third step—the rounding step, which must convert the minimum-cost fractional set cover x^* into a feasible integer solution x' (corresponding to a true set cover).

The rounding step should produce an x' that, with positive probability, has cost within a small factor of the cost of x^*. Then (since the cost of x^* is a lower bound on the cost of the optimal set cover), the cost of x' will be within a small factor of the optimal cost.

As a starting point, consider the most natural rounding scheme:

For each set $s \in \mathcal{S}$ *in turn, take* $x'_s = 1$ *with probability* $\min(1, x^*_s)$, *otherwise take* $x'_s = 0$.

With this rounding scheme, the expected cost of the chosen sets is at most $\sum_s c(s)x^*_s$, the cost of the fractional cover. This is good. Unfortunately the coverage is not good. When the variables x^*_s are small, the probability that an element e is not covered is about

$$\prod_{s \ni e} 1 - x^*_s \approx \prod_{s \ni e} \exp(-x^*_s) = \exp\Big(-\sum_{s \ni e} x^*_s\Big) \approx \exp(-1).$$

So only a constant fraction of the elements will be covered in expectation.

To make x' cover every element with high probability, the standard rounding scheme first *scales up* the rounding probabilities by an appropriate factor $\lambda > 1$. Here is the standard rounding scheme:

Fix a parameter $\lambda \geq 1$. *For each set* $s \in \mathcal{S}$ *in turn,*

take $x'_s = 1$ *with probability* $\min(\lambda x^*_s, 1)$, *otherwise take* $x'_s = 0$.

Scaling the probabilities up by λ increases the expected cost by λ, but makes coverage of all elements likely. The idea is to choose λ as small as possible so that all elements are provably covered with non-zero probability. Here is a detailed analysis.

---

lemma (approximation guarantee for rounding scheme)

Fix $\lambda = \ln(2|\mathcal{U}|)$. *With positive probability, the rounding scheme returns a set cover* x' *of cost at most* $2\ln(2|\mathcal{U}|)c \cdot x^*$ *(and thus of cost* $O(\log|\mathcal{U}|)$ *times the cost of the optimal set cover).*

(Note: with care the $O(\log|\mathcal{U}|)$ can be reduced to $\ln(|\mathcal{U}|) + O(\log\log|\mathcal{U}|)$.)

proof

The output x' of the random rounding scheme has the desired properties as long as none of the following "bad" events occur:

1. the cost $c \cdot x'$ of x' exceeds $2\lambda c \cdot x^*$, or
2. for some element e, x' fails to cover e.

The expectation of each x'_s is at most λx^*_s. By linearity of expectation, the expectation of $c \cdot x'$ is at most $\sum_s c(s)\lambda x^*_s = \lambda c \cdot x^*$. Thus, by Markov's inequality, the probability of the first bad event above is at most $1/2$.

For the remaining bad events (one for each element e), note that, since $\sum_{s \ni e} x^*_s \geq 1$ for any given element e, the probability that e is not covered is

$$\prod_{s \ni e} \big(1 - \min(\lambda x^*_s, 1)\big) < \prod_{s \ni e} \exp(-\lambda x^*_s) = \exp\Big(-\lambda \sum_{s \ni e} x^*_s\Big)$$
$$\leq \exp(-\lambda) = 1/(2|\mathcal{U}|).$$

(This uses the inequality $1 + z \leq e^z$, which is strict for $z \neq 0$.)

Thus, for each of the $|\mathcal{U}|$ elements, the probability that the element is not covered is less than $1/(2\mathcal{U})$.

By the naive union bound, the probability that one of the $1+|\mathcal{U}|$ bad events happens is less than $1/2+|\mathcal{U}|/(2\mathcal{U})=1$. Thus, with positive probability there are no bad events and x' is a set cover of cost at most $2\lambda c\cdot x^*$. QED

Derandomization using the method of conditional probabilities

The lemma above shows the *existence* of a set cover of cost $O(\log(|\mathcal{U}|)c\cdot x^*)$. In this context our goal is an efficient approximation algorithm, not just an existence proof, so we are not done.

One approach would be to increase λ a little bit, then show that the probability of success is at least, say, 1/4. With this modification, repeating the random rounding step a few times is enough to ensure a successful outcome with high probability.

That approach weakens the approximation ratio. We next describe a different approach that yields a deterministic algorithm that is guaranteed to match the approximation ratio of the existence proof above. The approach is called the method of conditional probabilities.

The deterministic algorithm emulates the randomized rounding scheme: it considers each set $s\in\mathcal{S}$ in turn, and chooses $x'_s\in\{0,1\}$. But instead of making each choice *randomly* based on x^*, it makes the choice *deterministically*, so as to *keep the conditional probability of failure, given the choices so far, below 1.*

Bounding the conditional probability of failure

We want to be able to set each variable x'_s in turn so as to keep the conditional probability of failure below 1. To do this, we need a good bound on the conditional probability of failure. The bound will come by refining the original existence proof. That proof implicitly bounds the probability of failure by the expectation of the random variable

$$F=\frac{c\cdot x'}{2\lambda c\cdot x^*}+|\mathcal{U}^{(m)}|,$$

where

$$\mathcal{U}^{(m)}=\Big\{e:\prod_{s\ni e}(1-x'_s)=1\Big\}$$

is the set of elements left uncovered at the end.

The random variable F may appear a bit mysterious, but it mirrors the probabilistic proof in a systematic way. The first term in F comes from applying Markov's inequality to bound the probability of the first bad event (the cost is too high). It contributes at least 1 to F if the cost of x' is too high. The second term counts the number of bad events of the second kind (uncovered elements). It contributes at least 1 to F if x' leaves any element uncovered. Thus, in any outcome where F is less than 1, x' must cover all the elements and have cost meeting the desired bound from the lemma. In short, if the rounding step fails, then $F\geq 1$. This implies (by Markov's inequality) that $E[F]$ *is an upper bound on the probability of failure.* Note that the argument above is implicit already in the proof of the lemma, which also shows by calculation that $E[F]<1$.

To apply the method of conditional probabilities, we need to extend the argument to bound the *conditional* probability of failure as the rounding step proceeds. Usually, this can be done in a systematic way, although it can be technically tedious.

So, what about the *conditional* probability of failure as the rounding step iterates through the sets? Since $F\geq 1$ in any outcome where the rounding step fails, by Markov's inequality, the *conditional* probability of failure is at most the *conditional* expectation of F.

Next we calculate the conditional expectation of F, much as we calculated the unconditioned expectation of F in the original proof. Consider the state of the rounding process at the end of some iteration t. Let $S^{(t)}$ denote the sets considered so far (the first t sets in $\mathcal{S}$). Let $x^{(t)}$ denote the (partially assigned) vector x' (so $x^{(t)}_s$ is determined only if $s\in S^{(t)}$). For each set $s\notin S^{(t)}$, let $p_s=\min(\lambda x^*_s,1)$ denote the probability with which

x'_s will be set to 1. Let $\mathcal{U}^{(t)}$ contain the not-yet-covered elements. Then the conditional expectation of F, given the choices made so far, that is, given $x^{(t)}$, is

$$E[F|x^{(t)}] \;=\; \frac{\sum_{s\in\mathcal{S}^{(t)}} c(s)x'_s + \sum_{s\not\in\mathcal{S}^{(t)}} c(s)p_s}{2\lambda c\cdot x^*} \;+\; \sum_{e\in\mathcal{U}^{(t)}} \prod_{s\not\in\mathcal{S}^{(t)}, s\ni e} (1-p_s).$$

Note that $E[F|x^{(t)}]$ is determined only after iteration t.

Keeping the conditional probability of failure below 1

To keep the conditional probability of failure below 1, it suffices to keep the conditional expectation of F below 1. To do this, it suffices to keep the conditional expectation of F from increasing. This is what the algorithm will do. It will set x'_s in each iteration to ensure that

$$E[F|x^{(m)}] \le E[F|x^{(m-1)}] \le \cdots \le E[F|x^{(1)}] \le E[F|x^{(0)}] < 1$$

(where $m = |\mathcal{S}|$).

In the t th iteration, how can the algorithm set $x'_{s'}$ to ensure that $E[F|x^{(t)}] \le E[F|S^{(t-1)}]$? It turns out that it can simply set $x'_{s'}$ so as to *minimize* the resulting value of $E[F|x^{(t)}]$.

To see why, focus on the point in time when iteration t starts. At that time, $E[F|x^{(t-1)}]$ is determined, but $E[F|x^{(t)}]$ is not yet determined --- it can take two possible values depending on how $x'_{s'}$ is set in iteration t. Let $E^{(t-1)}$ denote the value of $E[F|x'^{(t-1)}]$. Let $E^{(t)}_0$ and $E^{(t)}_1$, denote the two possible values of $E[F|x^{(t)}]$, depending on whether $x'_{s'}$ is set to 0, or 1, respectively. By the definition of conditional expectation,

$$E^{(t-1)} \;=\; \Pr[x'_{s'}=0]E^{(t)}_0 + \Pr[x'_{s'}=1]E^{(t)}_1.$$

Since a weighted average of two quantities is always at least the minimum of those two quantities, it follows that

$$E^{(t-1)} \;\ge\; \min(E^{(t)}_0, E^{(t)}_1).$$

Thus, setting $x'_{s'}$ so as to minimize the resulting value of $E[F|x^{(t)}]$ will guarantee that $E[F|x^{(t)}] \le E[F|x^{(t-1)}]$. This is what the algorithm will do.

In detail, what does this mean? Considered as a function of $x'_{s'}$ (with all other quantities fixed) $E[F|x^{(t)}]$ is a linear function of $x'_{s'}$, and the coefficient of $x'_{s'}$ in that function is

$$\frac{c_{s'}}{2\lambda c\cdot x^*} \;-\; \sum_{e\in s'\cap\mathcal{U}_{t-1}} \prod_{s\not\in\mathcal{S}^{(t)}, s\ni e} (1-p_s).$$

Thus, the algorithm should set $x'_{s'}$ to 0 if this expression is positive, and 1 otherwise. This gives the following algorithm.

Randomized-rounding algorithm for Set Cover

input: set system $\mathcal{S}$, universe $\mathcal{U}$, cost vector c

output: set cover x' (a solution to the standard integer linear program for set cover)

```
1. Compute a min-cost fractional set cover $x^*$ (an optimal solution to the LP relaxation).
2. Let $\lambda \leftarrow \ln(2|\mathcal{U}|)$. Let $p_s \leftarrow \min(\lambda x^*_s, 1)$ for each $s\in\mathcal{S}$.
3. For each $s'\in\mathcal{S}$ do:
    1. Let $\mathcal{S} \leftarrow \mathcal{S} - \{s'\}$. ( $\mathcal{S}$ contains the not-yet-decided sets.)
    2. If $\frac{c_{s'}}{2\lambda c\cdot x^*} > \sum_{e\in s'\cap\mathcal{U}} \prod_{s\in\mathcal{S}, s\ni e} (1-p_s)$
       then set $x'_s \leftarrow 0$,
       else set $x'_s \leftarrow 1$ and $\mathcal{U} \leftarrow \mathcal{U} - s'$.
```

($\mathcal{U}$ contains the not-yet-covered elements.)

4. Return x'.

---

lemma (approximation guarantee for algorithm)

The algorithm above returns a set cover x' of cost at most $2\ln(2|\mathcal{U}|)$ times the minimum cost of any (fractional) set cover.

proof

The algorithm ensures that the conditional expectation of F, $E[F \mid x^{(t)}]$, does not increase at each iteration. Since this conditional expectation is initially less than 1 (as shown previously), the algorithm ensures that the conditional expectation stays below 1. Since the conditional probability of failure is at most the conditional expectation of F, in this way the algorithm ensures that the conditional probability of failure stays below 1. Thus, at the end, when all choices are determined, the algorithm reaches a successful outcome. That is, the algorithm above returns a set cover x' of cost at most $2\ln(2|\mathcal{U}|)$ times the minimum cost of any (fractional) set cover.

Remarks

In the example above, the algorithm was guided by the conditional expectation of a random variable F. In some cases, instead of an exact conditional expectation, an *upper bound* (or sometimes a lower bound) on some conditional expectation is used instead. This is called a pessimistic estimator.

References

- Raghavan, Prabhakar; Tompson, Clark D. (1987), "Randomized rounding: A technique for provably good algorithms and algorithmic proofs", *Combinatorica* **7** (4): 365–374, doi: 10.1007/BF02579324 (http://dx.doi.org/10.1007/BF02579324).
- Raghavan, Prabhakar (1988), "Probabilistic construction of deterministic algorithms: approximating packing integer programs", *Journal of Computer and System Sciences* **37** (2): 130–143, doi: 10.1016/0022-0000(88)90003-7 (http://dx.doi.org/10.1016/0022-0000(88)90003-7).

Further reading

- Althöfer, Ingo (1994), "On sparse approximations to randomized strategies and convex combinations", *Linear Algebra and its Applications* **199**: 339–355, doi: 10.1016/0024-3795(94)90357-3 (http://dx.doi.org/10.1016/0024-3795(94)90357-3), MR 1274423 (http://www.ams.org/mathscinet-getitem?mr=1274423)
- Hofmeister, Thomas; Lefmann, Hanno (1996), "Computing sparse approximations deterministically", *Linear Algebra and its Applications* **240**: 9–19, MR 1387283 (http://www.ams.org/mathscinet-getitem?mr=1387283)
- Lipton, Richard J.; Young, Neal E. (1994), "Simple strategies for large zero-sum games with applications to complexity theory", *STOC '94: Proceedings of the twenty-sixth annual ACM symposium on theory of computing*, Montreal, Quebec, Canada: ACM, pp. 734–740, doi: 10.1145/195058.195447 (http://dx.doi.org/10.1145/195058.195447), ISBN 0-89791-663-8 Unknown parameter `|address=` ignored (help)

Article Sources and Contributors

Randomized rounding *Source*: http://en.wikipedia.org/w/index.php?oldid=571101586 *Contributors*: Auntof6, Frazzydee, Giftlite, Headbomb, Kiefer.Wolfowitz, LilHelpa, Melcombe, Nealeyoung, Rjwilmsi, Vadmium, 2 anonymous edits

License