CSC 2421 Approximation Algorithms
Lecture 7, Spring 2005

Lecturer: Allan Borodin
Scribe: Yuli Ye
Department of Computer Science
University of Toronto

1 Introduction

In this lecture, we show Integer Programming/Linear Programming (IP/LP) formulations for approximation algorithms. The standard form of Linear Programming (minimization problems) is defined as follows.

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{n} c_i x_i \\ \text{subject to} & \sum_{j=1}^{n} a_{ij} x_j \geq b_i, \quad 1 \leq i \leq m \\ & x_j \geq 0 \end{array}$$

For simplicity, we often use its vector form as follows.

$$\begin{array}{ll} \text{minimize} & c \cdot x \\ \text{subject to} & Ax \geq b \\ & x \geq 0 \end{array}$$

where c, x are column vectors of size n, b is a column vector of size m, and A is a $m \times n$ matrix.

For Integer Programming, the only difference form Linear Programming is that the domain of variables are the set of integers instead of the set of reals. One special class of IP problems are the $\{0, 1\}$ IP problems, which have variable domain $\{0, 1\}$. An LP problem can be solved efficiently, that is, in polynomial time, while IP problems are NP-hard.

The key idea for designing an approximation algorithm by IP/LP formulation is first to formulate the problem as an IP problem, then relax it to an LP problem, and finally either solve it by rounding the fractional solution or use a primal dual method to guide the construction of a feasible integral solution. In this lecture, we only focus on rounding.

2 Weighted vertex cover problem

The weighted vertex cover can be formulated as a $\{0,1\}$ IP problem. For a given input graph $G = (V, E)$, the IP formulation for the weighted vertex cover is the following.

$$\begin{array}{lll} \text{minimize} & \sum_{i=1}^{n} w_i x_i & \\ \text{subject to} & x_i + x_j \geq 1, & \forall (i,j) \in E \\ & x_i \in \{0,1\}, & \forall i \in \{1,\ldots,n\} \end{array}$$

where w_i is the weight of vertex v_i.

Note that there is a bijection between a weighted vertex cover for graph G and an $\{0,1\}$ solution of this IP problem. This IP formulation is not unique. For example, we can add following constraints.

$$x_i + x_j + x_k \geq 2, \text{ for all triangles } (i,j,k) \text{ in } G$$

The more constraints we formulate into the IP, the more complete information we have for that graph.

We can relax this IP problem to a LP problem by allowing fractional solutions, that is, letting $0 \leq x_i \leq 1$, for all $i \in \{1,\ldots,n\}$. However, $x_i \leq 1$ is redundant since if $x_i > 1$, we can always find a smaller x_i for which the constraints still hold, namely $x_i = 1$. Thus we can only consider $x_i \geq 0$, for all $i \in \{1,\ldots,n\}$. Therefore the LP-relaxation for the problem is the following.

$$\begin{array}{lll} \text{minimize} & \sum_{i=1}^{n} w_i x_i & \\ \text{subject to} & x_i + x_j \geq 1, & \forall (i,j) \in E \\ & x_i \geq 0, & \forall i \in \{1,\ldots,n\} \end{array}$$

2.1 A 2-approximation algorithm for weighted VC using LP-rounding

We first compute the optimal fractional solution x^* to the LP-relaxation, where x^* is a vector of values x_i^* for all $i \in \{1,\ldots,n\}$. We then produce a vector $\hat{x}$ for as follows.

$$\hat{x}_i = \begin{cases} 0 & \text{if } x_i^* < \frac{1}{2} \\ 1 & \text{otherwise} \end{cases}$$

Claim 1: The LP rounded solution $\hat{x}$ is a $\{0,1\}$ feasible solution of the IP.

Proof: Consider each constraint of the form $x_i + x_j \geq 1$, if it is satisfied by the fractional solution x^*, then either $x_i^* \geq \frac{1}{2}$ or $x_j^* \geq \frac{1}{2}$. This implies the integral solution $\hat{x}$ also satisfies $x_i + x_j \geq 1$. Therefore $\hat{x}$ is a $\{0,1\}$ feasible solution of the given IP.

□

We let $\hat{x}$ be the rounded LP solution; x^*, the optimal fractional solution for the LP-relaxation; and $\overline{x}$, the true optimal integral solution for the formulating IP. Let $\hat{C}, C^*$

and $\overline{C}$ denote their costs, respectively, where $\hat{C} = \sum_{i=1}^{n} w_i \hat{x}_i$, $C^* = \sum_{i=1}^{n} w_i x_i^*$ and $\overline{C} = \sum_{i=1}^{n} w_i \overline{x}_i$. We then have the following theorem.

Theorem 1: The above algorithm is a 2-approximation algorithm for the weighted vertex cover problem.

Proof: By Claim 1, the rounded LP solution $\hat{x}$ is a $\{0, 1\}$ feasible solution of the IP. Since for all $i \in \{1, \dots, n\}$, $\hat{x}_i \leq 2x_i^*$, we then have

$$\sum_{i=1}^{n} w_i \hat{x}_i \leq 2 \sum_{i=1}^{n} w_i x_i^*$$

It is also clear that

$$\sum_{i=1}^{n} w_i x_i^* \leq \sum_{i=1}^{n} w_i \overline{x}_i$$

Therefore,

$$\hat{C} = \sum_{i=1}^{n} w_i \hat{x}_i \leq 2 \sum_{i=1}^{n} w_i \overline{x}_i = 2\overline{C}$$

Hence the above algorithm is a 2-approximation algorithm for weighted vertex cover problem.

□

2.2 Integrality gap for IP/LP formulations of cost problems

Let I be an instance of a minimization problem with an IP formulation. Let $C[OPT(I)]$ be the cost of the optimal integral solution of the IP, and $C[OPT_F(I)]$ be the cost of the optimal fractional solution of the LP-relaxation. Then the integrality gap (IG) for this LP-relaxation is

$$\max_{I} \frac{C[OPT(I)]}{C[OPT_F(I)]}$$

For the unweighted vertex cover IP/LP formulation and the rounding strategy given in Section 2.1, the IG is clearly ≤ 2. This IG bound is essentially tight. We now exam two instances of the vertex cover problems.

- Let I be the complete graph K_n with n vertices. Then $x^* = (\frac{1}{2}, \frac{1}{2}, \dots, \frac{1}{2})$ is the fractional optimal solution. Thus $C[OPT_F(I)] = \frac{n}{2}$. For the integral optimal solution, since the graph is complete, we must select all but one vertex in order to cover all the edges. Thus $C[OPT(I)] = n - 1$. Therefore, the IG for this particular instance is $2 - \frac{2}{n}$, which implies that the IG for the unweighted vertex cover is $\geq 2 - \frac{2}{n}$. This shows the IG bound of 2 is tight asymptotically.
  Note that by using the rounding strategy given in Section 2.1, the rounded LP solution has cost n, which is very close to the optimal solution.
- Let I be the odd cycle C_n with n vertices. Then $x^* = (\frac{1}{2}, \frac{1}{2}, \dots, \frac{1}{2})$ is the fractional optimal solution. Thus $C[OPT_F(I)] = \frac{n}{2}$. For the integral optimal solution, since it is an odd cycle, we must select one more than half of the vertices in order to cover all the edges. Thus $C[OPT(I)] = \frac{n+1}{2}$. Therefore, the IG for this particular instance is $1 + \frac{1}{n}$.
  Note that by using the rounding strategy given in Section 2.1, the rounded LP solution has cost n, which is almost twice as much as the optimal solution.

A rounding strategy is *oblivious* if it does not use any other information from I. Intuitively, in the case of oblivious rounding, one cannot do better than the integrality gap of the relaxation.

3 The makespan problem on unrelated machines

The makespan problem was first studied in approximation algorithms by Graham. Its goal is to minimize the maximal completion time of a set of n jobs with processing time $p_1, \ldots, p_n$ on m identical machines. This is also called the makespan problem on identical machines. There are several variations of the makespan problem which have been studied. We first outline these variations and some known results for these problems.

- **Makespan on identical machines,** $P_m \mid\mid C_{max}$ **:**

  The processing time of each job is identical for all machines. This has a simple online $(2 - \frac{1}{m})$-approximation algorithm, and the LPT (Longest Processing Time) greedy algorithm is a $(\frac{4}{3} - \frac{1}{3m})$-approximation algorithm. The problem is NP-hard when $m = 2$.

- **Makespan on related machines,** $Q_m \mid\mid C_{max}$ **:**

  Each machine has a speed s_i. A job j takes time $\frac{p_j}{s_i}$ if it is scheduled on machine i. There is a 2-approximation algorithm (fixed priority) for this problem.

- **Makespan on unrelated machines,** $R_m \mid\mid C_{max}$ **:**

  It takes time p_{ij} to schedule job j on machine i. This has a 2-approximation algorithm, and the $\frac{3}{2}$ hardness of approximation lower bound. The 2-approximation algorithm is obtained by IP/LP formulation and rounding, which will be discussed in a later section.

- **Makespan on unrelated machines, subset model,** $S_m \mid\mid C_{max}$ **:**

  Each job can be only scheduled to a subset of all the machines. This has a 2-approximation algorithm, and the $\frac{4}{3}$ hardness of approximation lower bound.

3.1 An IP/LP formulation of makespan on unrelated machines

We can formulate the makespan problem on unrelated machines as an IP problem. We assume the processing time for each job is always a positive integer and there are more jobs than machines, that is, $n > m$.

$$\begin{array}{lll} \text{minimize} & t & \\ \text{subject to} & \sum_{i=1}^{m} x_{ij} = 1, & \forall j, 1 \leq j \leq n \\ & \sum_{i=1}^{n} x_{ij} p_{ij} \leq t, & \forall i, 1 \leq i \leq m \\ & x_{ij} \in \{0,1\}, & \forall i \in \{1,\ldots,m\}, \forall j \in \{1,\ldots,n\} \end{array}$$

Let $J_i(t)$ denote the set of jobs that require no more than t time units on machine i, and let $M_j(t)$ denote the set of machines that can process job j in no more than t time units. Then we can relax the above IP to an LP problem which allows fractional solutions.

$$\begin{array}{lll} \text{minimize} & t & \\ \text{subject to} & \sum_{i=1}^{m} x_{ij} = 1, & \forall j, j \in J_i(t) \\ & \sum_{i=1}^{n} x_{ij} p_{ij} \leq t, & \forall i, i \in M_j(t) \\ & 0 \leq x_{ij} \leq 1, & \forall i \in \{1, \ldots, m\}, \forall j \in \{1, \ldots, n\} \end{array}$$

This LP-relaxation gives an decision procedure to determine for any give t, whether or not there exists a feasible solution to the above LP. Since t can be easily bounded, we can narrow this t down to some integer value using binary search. It is easy to see that this procedure has the appropriate performance guarantee. First of all, the algorithm runs clearly in polynomial time. Secondly, the integer value we obtained, say T, is a lower bound for the optimal solution for the IP formulation.

Once such T is obtained, we can compute the fractional solution x^* for the LP-relaxation, and this fractional solution can be convert to an integral solution such that its cost is $\leq 2T$.

3.2 A 2-approximation algorithm for makespan on unrelated machines

In this section, we show how to convert the fractional solution x^* to an integral solution $\hat{x}$ so that we achieve approximation ratio of 2.

Theorem 2: There is a 2-approximation algorithm for the makespan problem on unrelated machines.

Proof[1]: Note that there are $m + n$ inequalities. Therefore, by the theory of linear programming, if a feasible solution exists, it must exist a basic feasible solution with at most $m + n$ variables (in this case x_{ij}) are positive.

Assume we have such a basic feasible solution. Then, since all n jobs must be scheduled, there is at lease one positive variable for each job. Therefore, at most m jobs can have more than one positive indicator variable. Furthermore, if a job j only has only one positive variable x_{ij}, then we know that $x_{ij} = 1$.

For every variable x_{ij} such that $x_{ij} = 1$, assign job j to machine i. We call this as schedule S_1. Then it is clear that S_1 has makespan $\leq T$ since it is a subset of the feasible solution to the LP.

Now focus on those jobs that are fractionally assigned. We build a bipartite graph G such that G has n vertices representing jobs and m vertices representing machines, see Fig. 1. We then create an edge (i, j) in G for every positive fractional x_{ij}. Note that the number of edge in G is no more than the number of nodes in G. Furthermore, if G has connected components, say $G_1, \ldots, G_c$, then the same holds for every connected component, that is,

$$|V| - 1 \leq |E| \leq |V|$$

where $|V|$ is the number of nodes and $|E|$ is the number of edges. Therefore, G is a *pseudoforest*, that is, each connected component of G is either a tree or a tree plus one edge. This property allows us to constructs a matching M on G, and for every edge (i, j) in M, we then schedule job j on machine i. We call this as schedule S_2. The matching

[1]This proof is adapted from Karger et. al. [2] page 32

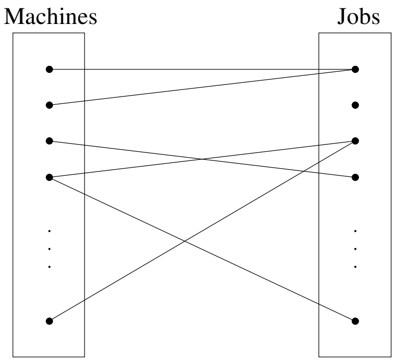

Figure 1: Bipartite graph G

has forced us to schedule at most one job per machine and since j is assigned to i only if $x_{ij} > 0$, we know that $p_{ij} < T$ for each job/machine pair in S_2. Hence the makespan of S_2 is at most T.

Note that S_1 together with S_2 form a integral solution for the original IP formulation, and the cost of this solution is at most $2T$. Therefore, this is a 2-approximation algorithm.

□

References

[1] Borodin, A. (2004) Lecture Notes of CSC2421- Approximation Algorithms, Department of Computer Science, University of Toronto.

[2] Karger, D., Stein, C. & Wein, J. (1997) Scheduling algorithms. *Handbook of Algorithms and Theory of Computation.*

[3] Lenstra, J.K., Shmoys, D.B. & Tardos, E. (1990) Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming* **46** (3) : 259-271.