COM S 783 / OR&IE 634 Approximation Algorithms	**Fall Semester, 2003**

Lecture 6: Sept 17, 2003

Lecturer: Yuval Rabani *Scribe: Frans Schalekamp*

1 MAX SAT

Recall that we looked at the IP-formulation of MAX SAT, last time:

$$
\begin{aligned}
&\max \sum_j z_j \\
&\text{s.t.} \sum_{i:x_i \in C_j} y_i + \sum_{i:\bar{x}_i \in C_j} (1 - y_i) \geq z_j \text{ for all } j \\
&y_i \in \{0,1\} \text{ for all } i \\
&z_j \in \{0,1\} \text{ for all } j
\end{aligned}
$$

where C_j are the clauses of the Boolean formula, x_i the literals, and the variables $y_i = 1$ iff x_i is set to TRUE, and $z_j = 1$ iff C_j is set to TRUE.

The LP-relaxation of this formulation is

$$
\begin{aligned}
&\max \sum_j z_j \\
&\text{s.t.} \sum_{i:x_i \in C_j} y_i + \sum_{i:\bar{x}_i \in C_j} (1 - y_i) \geq z_j \text{ for all } j \\
&0 \leq y_i \leq 1 \text{ for all } i \\
&0 \leq z_j \leq 1 \text{ for all } j.
\end{aligned}
$$

We were investigating how well the algorithm does, that sets x_i to TRUE with probability y_i, independently of the other x_i's, where y is an optimal solution to the LP-relaxation.

Without loss of generality, consider the clause $C_j = (x_1 \vee x_2 \vee \cdots \vee x_k)$:

$$
\begin{aligned}
P(C_j \text{ is not satisfied}) &= P(\ x_1, x_2, \ldots, x_k \text{ are not satisfied}) \\
&= \prod_{i=1}^{k} P(\ x_i \text{ is not satisfied}) \qquad (1) \\
&= \prod_{i=1}^{k} (1 - y_i) \\
&\leq \prod_{i=1}^{k} \left(1 - \frac{z_j}{k}\right) \qquad (2) \\
&= \left(1 - \frac{z_j}{k}\right)^k,
\end{aligned}
$$

where equation (1) follows from the fact that the x_i's are set independently, and inequality (2) follows from the constraint $\sum_{i=1}^{k} y_i \geq z_j$, and noting that $\prod_{i=1}^{k}(1 - y_i)$ is maximal for $y_i = z_j/k$ for $i = 1, 2, \ldots, k$.

Therefore,

$$\begin{aligned} P(C_j \text{ is satisfied}) &= 1 - P(C_j \text{ is not satisfied}) \\ &\geq 1 - \left(1 - \frac{z_j}{k}\right)^k. \end{aligned}$$

Now noting that $f(z_j) = 1 - (1 - z_j/k)^k$ is a concave function in z_j, and that $f(0) = 0$ and $f(1) = 1 - (1 - 1/k)^k$, gives

$$1 - \left(1 - \frac{z_j}{k}\right)^k \geq z_j(f(1) - f(0)) = z_j\left(1 - \left(1 - \frac{1}{k}\right)^k\right) \tag{3}$$

i.e.,

$$P(C_j \text{ is satisfied}) \geq z_j\left(1 - \left(1 - \frac{1}{k}\right)^k\right) \geq z_j(1 - e^{-1}).$$

(Note that the argument given in class to justify the result of (3) was incorrect.)

We can therefore calculate the expected number of clauses that are satisfied:

$$\begin{aligned} E(\#\text{clauses satisfied}) &= \sum_j P(C_j \text{ satisfied}) && (4) \\ &\geq \sum_j z_j(1 - e^{-1}) \\ &= (1 - e^{-1}) \sum_j z_j \\ &\geq (1 - e^{-1})(\textsc{opt}) && (5) \end{aligned}$$

where equation (4) follows from the linearity of expectation, and (5) from the fact that the LP is a relaxation of the IP.

Let's compare the above algorithm, Randomized Rounding or RR for short, with Johnson's algorithm [3]:

size of clause C_j	probability that clause is satisfied	
	Johnson	RR
1	1/2	$\geq z_j$
2	3/4	$\geq (3/4)z_j$
3	7/8	$\geq (19/27)z_j$
$\vdots$	$\vdots$	$\vdots$
k	$1 - (1/2)^k$	$\geq (1 - (1 - 1/k)^k)z_j$

Note that for each clause size, the probability of being satisfied, averaged over the two algorithms, always is at least $(3/4)z_j$. Therefore the algorithm that picks the better assignment of Johnson's algorithm and RR, is a 3/4-approximation algorithm.

In fact, it is not hard to see (and explained below) that

$$\frac{1}{2}\Big(1-\Big(\frac{1}{2}\Big)^k\Big)+\frac{1}{2}\Big(1-\Big(1-\frac{1}{k}\Big)^k\Big)z_j \geq \frac{3}{4}z_j \tag{6}$$

which implies that the randomized algorithm that performs Johnson's algorithm with probability 1/2, and RR with probability 1/2, also is a 3/4-approximation.

For $k=1$ and $k=2$, the validity of (6) can be read off the table above. For higher k, note that $(1-(1-1/k)^k) \geq 1-e^{-1} \geq 5/8$, and $1-(1/2)^k \geq 7/8$. Therefore

$$\begin{aligned}\frac{1}{2}\Big(1-\Big(\frac{1}{2}\Big)^k\Big)+\frac{1}{2}\Big(1-\Big(1-\frac{1}{k}\Big)^k\Big)z_j &\geq \frac{7}{16}+\frac{5}{16}z_j\\ &\geq \frac{7}{16}z_j+\frac{5}{16}z_j\\ &= \frac{3}{4}z_j.\end{aligned}$$

The above algorithms are due to Goemans & Williamson [1], matching the bound of an algorithm due to Yannakakis [7], which "makes non-trivial use of solutions to maximum flow problems" according to the abstract of [1].

Conjecture 1 (Karloff & Zwick 1997 [4]). There exists a 7/8-approximation algorithm for MAX SAT.

Hardness Result 2 (Håstad 1997 [2]). Approximating MAX SAT within a factor of $7/8+\varepsilon$ is NP-hard for all $\varepsilon > 0$.

We may come back to this later in the course.

2 Randomized Rounding

The approximation algorithm above for MAX SAT, is an example of an algorithm that uses *randomized rounding*: creating an integer solution from a feasible fractional solution, using coin tosses in deciding the roundoffs. A common problem, which we didn't have in creating an integer solution for MAX SAT, is creating a *feasible* integer solution. We do have this problem in the following example.

3 Integer multicommodity flow

The input for a multicommodity flow problem consists of

- a graph $G = (V, E)$,
- a nonnegative cost function c on the edges, i.e. $c : E \to \mathbb{Q}^+$, and,
- for each commodity i, a pair of terminals s_i, t_i and a demand d_i.

Now different variants are considered:

- the objective can be to route as much as possible
- the question can be, how much do we have to exceed the capacities by, to satisfy demands
- flows can be restricted to be integer, or even unsplittable

etcetera. We will come back to several variants of this problem.

The variant we will now study has the following characteristics:

- all demands are 1
- all capacities are 1
- the flow has to be integer
- the objective is to minimize *congestion*, defined to be the maximal load on the edges.

I.e., we want to connect each pair s_i, t_i by a path, while we minimize the maximal number of paths that use an edge. Define

$$f_e^i = \begin{cases} 1 & \text{if commodity } i \text{ uses edge } e \\ 0 & \text{if commodity } i \text{ does not use edge } e. \end{cases}$$

Assuming the edges are directed, we can now write down the following IP-formulation for our problem:

$$\min u$$

$$\text{s.t.} \sum_i f_e^i \leq u \text{ for all } e \in E$$

$$\sum_{e:e=(vw)} f_e^i = \sum_{e:e=(wv)} f_e^i \text{ for all } v \in V \backslash \{s_i, t_i\}, \text{ and for all } i \tag{7}$$

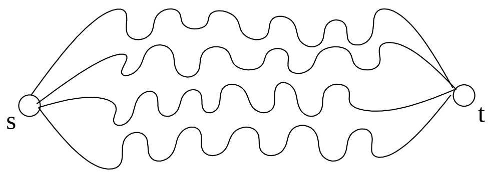
$$\sum_{e:e=(s_i w)} f_e^i = 1 \text{ for all } i \tag{8}$$

$$\sum_{e:e=(w t_i)} f_e^i = 1 \text{ for all } i \tag{9}$$

$$f_e^i \in \{0, 1\} \text{ for all } e \in E, \text{ and for all } i \tag{10}$$

where (7) is the flow conservation constraint, and (8) and (9) make sure that the flow of commodity i will actually go to and from s_i and t_i, and will be of size 1.

As always we relax the integrality constraints (10), and consider an optimal solution to the LP. How do we round this solution? It is highly unlikely that we will get a feasible solution if we round each variable independently. But we can decompose the flow into flow paths:



Denote by q_j^i the characteristic function of the j-th path connecting s_i, t_i, i.e.

$$q_j^i(e) = \begin{cases} 1 & \text{if } e \in j\text{-th path} \\ 0 & \text{if } e \notin j\text{-th path} \end{cases}$$

and define f_j^i = flow along j-th path.

So we have the following LP:

$$\begin{aligned} &\min u \\ &\text{s.t. } \sum_j f_j^i = 1 \text{ for all } i \\ &\sum_{i,j} f_j^i q_j^i(e) \leq u \text{ for all } e \\ &f_j^i \geq 0 \text{ for all } i, j \end{aligned}$$

which, potentially, has an exponential number of variables, but since we can solve the other LP-formulation and transform the solution of that into a solution of this LP-formulation, with only a polynomial number of non-zero variables[1], we are not worried about this.

Now note that $\{f_j^i\}_j$ have the form of probability distributions, for each i. This suggests the following rounding scheme: choose for each commodity i exactly one path, such that commodity i is sent over path j with probability f_j^i. Let's look at the other constraint:

$$\sum_{i,j} f_j^i q_j^i(e) \leq u \text{ for all } e.$$

Note that $\sum_{i,j} f_j^i q_j^i(e) = \sum_i f_e^i$ from the first formulation. Also,

$$P(\text{path for } s_i,\, t_i \text{ uses } e) = f_e^i = \sum_j f_j^i q_j^i(e).$$

[1]Namely, consider the following algorithm for each commodity i: take the graph G, delete all edges with $f_e^i = 0$. In the remaining graph, find any s_i, t_i-path, and look at the minimal f_e^i for this path. Let this be the j-th for the other LP. Now set f_j^i equal to the minimal f_e^i for this path, decrease all variables f_e^i on this path with that amount, and delete the edges which have $f_e^i = 0$. Repeat this process, until the graph has no more edges. It is easy to see that this will indeed create a feasible solution for the second LP. Moreover, since in each iteration at least one edge is deleted, the running time is $O(|E|^2)$, for each commodity. Noting that in each iteration exactly one variable is set to a positive value, shows the claim.

The expected congestion on a particular edge e is therefore, by linearity of expectation,

$$E = E(\text{congestion on } e) = \sum_i f_e^i \leq u.$$

But this is not good enough! If we repeat the rounding scheme "often enough", the solution with the best congestion for an edge e will be lower than $u + \varepsilon$ for each edge seperately, but it is not clear (and not true in general) that the best maximal congestion over all edges, will be lower than $u + \varepsilon$! We really want a "high probability" result.

Define the random variables $X_i :=$ indicator for "path for s_i, t_i uses e". Note that $P(X_i = 1) = f_e^i$ and that the X_i's are independent $0, 1$-random variables, i.e. we can use Chernoff Bounds to bound the probability that the sum of them is "big"! Recall the Chernoff Bound

$$P\Big(\sum_i X_i > (1+\beta)\max\{1, E\}\Big) < \Big(\frac{e^\beta}{(1+\beta)^{1+\beta}}\Big)^{\max\{1,E\}}$$

where the $\max\{1, E\}$ is because every edge capacity is 1, and we are only interested in violations of the capacity. Now taking

$$(1+\beta) = \Theta\Big(\frac{\log n}{\log\log n}\Big)$$

where $n = |V|$, gives

$$P\Big(\sum_i X_i > (1+\beta)\max\{1, E\}\Big) < \frac{1}{\text{poly}(n)}.$$

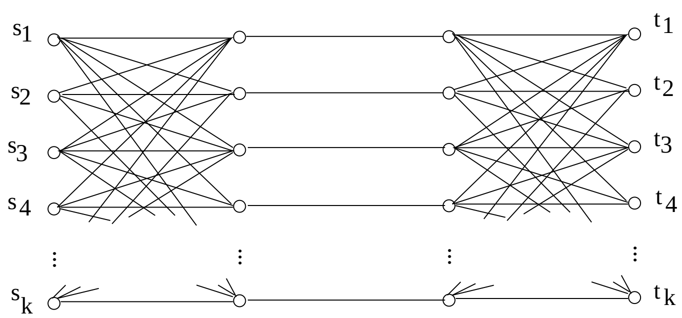
By making an appropriate choice for our constant c in $1+\beta = c(\log n/\log\log n)$, we can make sure that the polynomial is smaller than $n^2/2$. We can use the union bound (Bonferroni's inequality), and have the guarantee that we have a solution with

$$P(\text{there exists } e \text{ with congestion} > (1+\beta)\max\{1, E\}) < \frac{1}{2},$$

I.e., by taking a constant number of rounded solutions, we have a polynomial time algorithm, that with high probability gives a feasible solution with a (relative) performance guarantee.

Remark 1 The bound proves that the integrality gap is $O(\log n/(\log\log n))$.

Remark 2 The bound above is tight, recall "balls-in-bins":



The optimal solution is 1, which can be achieved by routing all flows in a straight line. This is an integral solution. However, the solution such that an amount of $1/k$ of each commodity is routed over every edge in the "middle part" of the graph is also optimal. Rounding this solution will, with high probability, give a congestion of $\log k/(\log\log k)$ on one of the middle edges.

Remark 3 We can construct a deterministic algorithms from this, using derandomization using the method of conditional expectation [6].

Remark 4 The algorithm is an *asymptotic* PTAS, i.e. as the optimal value for the fractional solution E approaches ∞, since we can choose smaller and smaller β, and generate a solution that is $(1+\varepsilon)$ of ($\textsc{opt}$).

Remark 5 The above algorithm is due to Raghavan & Thompson [5].

4 Other variant

We will now consider a different variant. Our assumptions now are

- capacities are rigid (fixed)
- the objective is to route as much flow as possible, where flows have to be integral.

If the capacities are small, we are in trouble. So let's assume these are big (say, at least $O(\log n)$). Note that our previous method won't work, since the solution that we construct may not be feasible. We can work around this problem, by *scaling the variables from the fractional solution down* by a constant fraction, before we do the rounding. Using Chernoff Bounds again, we can show that with high probability the constraints will not be violated.

References

[1] Michel X. Goemans and David P. Williamson. New $\frac{3}{4}$-approximation algorithms for the maximum satisfiability problem. *SIAM J. Discrete Math.*, 7(4):656–666, 1994.

[2] J. Håstad. Some optimal inapproximability results. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 1–10, 1997.

[3] David S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. System Sci.*, 9:256–278, 1974. Fifth Annual ACM Symposium on the Theory of Computing (Austin, Tex., 1973).

[4] Howard Karloff and Uri Zwick. A 7/8-approximation algorithm for MAX 3SAT? In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science, Miami Beach, FL, USA*. IEEE Press, 1997.

[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] Prabhakar Raghavan. Probabilistic construction of deterministic algorithms: approximating packing integer programs. *Journal of Computer and System Sciences*, 37(2):130–143, 1988.

[7] Mihalis Yannakakis. On the approximation of maximum satisfiability. *J. Algorithms*, 17(3):475–502, 1994. Third Annual ACM-SIAM Symposium on Discrete Algorithms (Orlando, FL, 1992).