CMPUT 675: Randomized Algorithms — Fall 2005

Lecture 12: Oct 18

Lecturer: Mohammad R. Salavatipour — *Scribe: Ken Anderson*

12.1 A randomized algorithm for 2-SAT

Consider the 2-SAT problem. In this problem, we are given a formula ϕ in 2-CNF format (i.e. two variables per clause). Assume that the variables of ϕ are $x_1, \ldots, x_n$ and the clauses are $C_1, \ldots, C_m$. We wish to find (if there exists one) a truth assignment to the variables that satisfies ϕ. The following is a simple randomized algorithm.

Start with an arbitrary truth assignment. As long as there is some unsatisfied clause do:

1. Pick an unsatisfied clause C_i uniformly randomly.
2. Pick one of the variables of C_i uniformly randomly and flip its value.

Suppose ϕ is satisfiable and let A be a fixedsatisfying assignment. Value a_i for variable x_i is called "correct" if A sets $x_i = a_i$. Let X_i be the number of variables in current assignment that are correct. It is easy to see that with each step, the value of X_i changes by 1:

$$|X_i - X_{i+1}| = 1$$

The algorithm stops when $X_i = n$, i.e. all variables have correct value (or maybe even before that by finding a truth assignment different from A that satisfies ϕ). In the case that all the variables are incorrect, flipping any variable will move us closer to A.

$$\Pr[X_{i+1} = 1 | X_i = 0] = 1.$$

In all other cases, a change could move us closer to A or further away. The process is like a random walk on integers with a barrier at 0; each step either increases or decreases the number of correct values by one. We stop when we get to n. Let $1 \leq X_i \leq n-1$, and consider a random unsatisfied caluse C_j. It disagrees with A in at least one variable. With probability at least $\frac{1}{2}$ we pick that incorrect variables and once we flip it the number of incorrect variables decreases by one. So

$$\Pr[X_{i+1} = j+1 | X_i = j] \geq 1/2.$$

We will show that the expected number of steps to get to A is n^2. Is this a Markov chain? not exactly. Because the probability of going from one state to another depends on the assignment of clauses (and therefore on the previous states). Instead, we define another random walk which is a pessimistic estimatore of X_i's and is indeed a Markov Chain. Let $Y_0 = X_0$ and for every $i \geq 1$:

$$\Pr[Yi + 1 = 1 | Y_i = 0] = 1$$

$$\Pr[Yi + 1 = j + 1 | Y_i = j] = 1/2$$
$$\Pr[Yi + 1 = j - 1 | Y_i = j] = 1/2$$

Clearly, the expected time for Markov chain Y to reach n is greater than or equal to the expected time required for X. We bound the expected time for Y (which in turn gives an upper bound for the expected time before X hits n). Let Z_j be the number of steps to reach n from state Y_j. Then:

$$\begin{aligned} h_i = \mathrm{E}[Z_i] &= \mathrm{E}[\frac{1}{2}(1 + Z_{i+1}) + \frac{1}{2}(1 + Z_{i-1})] \\ &= \frac{h_{i-1} + 1}{2} + \frac{h_{i+1} + 1}{2} \\ &= \frac{h_{i-1}}{2} + \frac{h_{i+1}}{2} + 1 \end{aligned}$$

We also have $h_0 = h_1 + 1$ and $h_n = 0$. By induction we show $h_i = h_{i+1} + 2i + 1$ The base case $i = 0$ is trivial. For induction step:

$$\begin{aligned} h_j &= \frac{h_{j-1}}{2} + \frac{h_{j+1}}{2} + 1 \\ \Longrightarrow 2h_j &= h_{j-1} + h_{j+1} + 2 \\ \Longrightarrow h_{j+1} &= 2h_j - h_{j-1} - 2 \\ &= 2h_j - (h_j + 2(j-1) + 1) - 2 \\ &= h_j - 2j - 1 \end{aligned}$$

So $h_0 = \sum_{i=1}^{n-1}(2i + 1) = n^2$. Thus if ϕ is satisfiable and we run the algorithm for $2n^2$ steps, by Markov's Inequality: $\Pr[failure] \leq 1/2$. Since we run the trial t times, in $2tn^2$ steps we find a solution with $prob \geq 1 - 2^{-t}$ if there is any.

12.2 Random Walk on Graphs

The algorithm in the previous problem was an example of a random walk on a graph (on a path). In general, we can define a simple random walk on a graph as follows: Start at a vertex u. Pick one of the $d(u)$ neighbors uniformly randomly and go to that neighbor. Keep doing this. When analysing, there are specific quantities of interest:

- H_{uv} : hitting time from u to v = E[number of steps to first reach v|start at$u$]
- C_{uv} : commute time from u to v and back to $u = H_{uv} + H_{vu}$
- C_u : cover time = E[number of steps to visit all vertices|start u] for entire graph: $C_G = \max_u C_u$

Random walks on graphs have several applications. For instance, we can show how to figure out if there is a path between two given vertices s, t in a graph (s, t-connectivity) in only $O(\log n)$-space in randomized polytime.

12.2.1 Analysis of random walks using resistence graph

We will model our graph by representing as a circuit. Recall the following two fundamental laws from physics:

Definition 12.1 *Kirchoff's Law:*

$$\text{Total current going into any point} = \text{total coming out}$$

Definition 12.2 *Ohm's Law:*

$$V = RI$$

Consider G is a circuit and assume every edge has Resistance 1. When resisters are aligned as follows, total resistance R is

- Series: $R = R_1 + R_2$
- Parallel: $\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} \longrightarrow R = \frac{R_1 R_2}{R_1 + R_2}$

Definition 12.3 *Effective Resistance R_{uv} between u and v is the potential difference V between u and v when one unit of flow is going from u to v.*

Theorem 12.4 *Given a graph G. Think of each edge as a unit resistence and let R_{uv} be the effective resistence between u and v. Then:*

$$C_{uv} = 2m * R_{uv}$$

Proof: Take a battery and inject $d(x)$ units of flow into each node x and remove $2m$ units of flow from v. Let ϕ_{uv} be the difference of voltage between u and v. Then:

$$\begin{aligned} d(u) &= \sum_{ux \in E} (\text{current from } u \text{ to } x) \\ &= \sum_{ux \in E} \phi_{ux} \\ &= \sum_{ux \in E} \phi_{uv} - \phi_{xv} \\ &= d(u)\phi_{uv} - \sum_{ux \in E} \phi_{xv} \\ \longrightarrow \phi_{uv} &= 1 + \sum_{ux \in E} \frac{\phi_{xv}}{d(u)} \qquad (12.1) \end{aligned}$$

Note that we have one such equation for every pair uv, i.e. we have a system of equations. Now consider the Random walk experiment. Starting at a vertex u there are $d(u)$ possible paths. If we take a step from u to x then:

$$H_{uv} = 1 + \sum_{ux \in E} \frac{H_{uv}}{d(u)} \qquad (12.2)$$

Since system of equations 12.1 and 12.2 are identical, the systems have the same solutions. Therefore:

$$H_{uv} = \phi_{uv}$$

We can perform the exact same experiment, this time injecting flows to the vertices and taking the total $2m$ units of flow from u. This implies that

$$H_{vu} = \phi'_{vu}.$$

Now in this second experiment, reverse all the flows. In this case, we have a $2m$ units of flow going in from u and $d(x)$ units is coming out from all vertices. Now if we combine the first experiment with the last one, i.e. add the flows injected and taken out in the two experiment, we have:

- a total of $2m$ units of flow is going into u
- a total of $2m$ units of flow is goint out from v
- the net flow going in and out from every other vertex is zero.

Thus:

$$\phi_{uv} + \phi'_{vu} = H_{uv} + H_{vu} = 2m * R_{uv} = C_{uv}$$

■

Example: For a random walk on a path P_n (points $0, \ldots, n-1$): $m = n$ and for any two points i and j: $R_{ij} = j - i$ and $H_{ij} + H_{ji} = 2(n-1)(j-i)$. Thus $H_{0n} + Hn0 = 2n^2$. Since a path is symetric: $H_{0n} = n^2$.

Note that H_{uv} does not necessarily equal H_{vu}. Consider the following example.

Example: Lollipop graph L_n which consist of a path of length $n/2$ hanging from a vertex v in the complete graph $K_{n/2}$. Let u be the other end-point of this path. We have: $R_{uv} = \frac{n}{2}$, $m = \frac{n}{2} - 1 + \binom{n/2}{2} \in \Theta(n^2)$. So $H_{uv} + H_{vu} = 2mR_{uv} = nm \in \Theta(n^3)$. From the previous example we know that $H_{uv} = \left(\frac{n}{2}\right)^2 = \frac{n^2}{4}$. So $H_{vu} \in \Theta(n^3)$.

The following easy fact is used in the next theorem.

Fact: If $uv \in E$ then $R_{uv} \leq 1$.

Theorem 12.5

$$C(G) \leq 2m(n-1)$$

Proof: Let T be some spanning tree of G and do a depth first search (DFS) traversal of T. Let $u = v_0, v_1, ...v_{2n-1}$ be the sequenceo f visits to the verticies (note that each edge is travelled exactly twice, once in every direction). So

$$\begin{aligned}
C_u &= H_{v_0 v_1} + H_{v_1 v_2} + ... + H_{v_{2n-3} v_{2n-2}} \\
&= \sum_{uv \in T} C_{uv} \\
&= \sum_{uv \in E} 2m * R_{uv} \\
&\leq 2m(n-1)
\end{aligned}$$

■

For example, on a complete graph with n vertices, i.e. K_n, we have: $C(K_n) \in \Theta(n^3)$