Randomized Algorithms 2019A – Lecture 3
Effective Resistance and an Algorithm for 2SAT*

Robert Krauthgamer

1 Reminder: Graphs as Electrical Networks

Recall that in an electrical network, we view a graph as a collection of (undirected) resistors. When we impose a potential difference ϕ_{uv} between vertices u, v, it induces an electrical flow (current), which is (i) a feasible flow in the sense of flow preservation (KCL), and (ii) creates potentials (voltages) on all other vertices (KVL), and (iii) the flow along each edge is inverse proportional to the potential difference, and directed accordingly (Ohm's Law).

Recall our notation $\phi_{uv} = \phi_u - \phi_v$ and f is defined on "directed" edges with $f_{uv} = -f_{vu}$, even though all the edges $uv \in E$ are undirected.

Example: Suppose G is a path on 3 vertices u, w, v, and we create potential difference ϕ_{uv}. Then by Ohm's Law and KCL,

$$\phi_{uw} = f_{uw} r_{uw} = f_{wv} r_{wv} = \phi_{wv}.$$

Since the LHS and RHS sum up to ϕ_{uv}, each of them is exactly $\frac{1}{2}\phi_{uv}$, and thus $f_{uw} = f_{wv} = \frac{1}{2}\phi_{uv}$ is the amount of flow.

Observation: The amount of flow shipped from u to v scales linearly with ϕ_{uv}.

Observation: In fact, we can also add two potential-difference functions, and the flows will add up (and vice versa).

Theorem 1 (Thomson's Principle of minimum energy): Let f be a flow that ships a unit flow from s to t, and has minimum total energy dissipation

$$E(f) = \sum_{uv \in E} f_{uv}^2 r_{uv}$$

among all such flows. Then f is an electrical flow.

Proof: Was seen in class.

*These notes summarize the material covered in class, usually skipping proofs, details, examples and so forth, and possibly adding some remarks, or pointers. The exercises are for self-practice and need not be handed in. In the interest of brevity, most references and credits were omitted.

2 Effective Resistance

Effective Resistance: The *effective resistance* between vertices u, v in an electrical network, denoted $R_{\text{eff}}(u, v)$, is the potential difference ϕ_{uv} we need to create between u and v to induce exactly one unit of current flowing from u to v.

By the linearity observations above, if we impose potential difference $\phi_{uv} = 1$ then the current will be $1/R_{\text{eff}}(u, v)$, which is also called the *effective conductance* $C_{\text{eff}}(u, v)$.

The name comes from the viewpoint that the entire network can be "simulated" by a single resistor between u, v, with resistance $r_{uv} = R_{\text{eff}}(u, v)$, then the current between u, v would be the same. Indeed, if we impose the same potential difference $\phi_{uv} = R_{\text{eff}}(u, v)$ on this single resistor, thus the amount of flow will be $f_{uv} = \phi_{uv}/r_{uv} = 1$, exactly as in G.

Notice that $R_{\text{eff}}(u, v)$ is symmetric (by the linearity observations).

We can now show that the effective resistance is essentially the same as the commute time.

Theorem 2 [Chandra, Raghavan, Ruzzo, Smolensky and Tiwari, 1989]: Let $G = (V, E)$ be an undirected graph. Then

$$\forall u, v \in V, \qquad C_{uv} = 2|E| R_{\text{eff}}(u, v).$$

Proof idea: These quantities satisfy the same set of linear equations. For the actual proof it is more convenient to deal with the hitting time.

Lemma 3: Let N_z be the electrical network corresponding to G, when we inject $\deg(u)$ units of flow at every vertex $u \in V$, and extract $\sum_{u \in V} \deg(u) = 2|E|$ units of flow at z. Then the potential differences ϕ^{N_z} satisfy

$$\forall u \in V, \qquad \phi_{uz}^{N_z} = H_{uz}.$$

Proofs of Lemma 3 and Theorem 2: Was seen in class.

Theorem 4 (Thomson's Principle revisited): Let f be a flow that ships one unit from u to v with minimum energy. Then

$$R_{\text{eff}}(u, v) = E(f).$$

It provides an alternative definition for effective resistance.

Proof: Was seen in class.

Theorem 5 (Rayleigh's Monotonicity Law): If $\{r(e)\}$ and $\{r'(e)\}$ are sets of resistances on the edges of the same graph G, such that $r(e) \le r'(e)$ for all $e \in E$,

$$\forall u, v \in V, \qquad R_{\text{eff}}^{(r)}(u, v) \le R_{\text{eff}}^{(r')}(u, v).$$

The proof follows directly from Thomson's Principle above, as the LHS minimizes the energy over all unit flows, including the flow that attains the RHS.

Corollary 6: For all $(u,v) \in E$, we have $R_{\text{eff}}(u,v) \leq 1$ and thus $C_{uv} \leq 2|E|$.

The proof follows by observing that adding an edge is equivalent to reducing the resistance of an edge.

This proves Theorem 3 from last week (claimed earlier without a proof).

Lemma 7 (Bridge Edge): Suppose edge $uv \in E$ is a *bridge* in G (which means that removing this edge disconnects the graph). Then $R_{\text{eff}}(u,w) = 1$, and thus $C_{uv} = 2|E|$.

Exer: Prove this.

Example A: The path: $C_{1n} = H_{1n} + H_{n1} = 2H_{1n}$ by symmetry. By Lemma 7 about bridges, $R_{\text{eff}}(1,n) = n-1$, and thus $C_{1n} = 2(n-1)R_{\text{eff}}(1,n) = 2(n-1)^2$. We conclude that $H_{1n} = (n-1)^2$.

Notice this is also the cover time of that path.

Example B: The lollipop: The "lollipop" graph is a path of $n/2$ edges from u to v, where this last vertex v forms a clique with $n/2-1$ new vertices. It can be easily seen $H_{uv} = (n/2)^2$ while $H_{vu} = \Theta(n^3)$ and also $\text{cov}(G) = \Theta(n^3)$.

Exer: Prove these bounds (it's actually easy to get precise formulas).

Hint: Use the effective resistance formula and Theorem 5 (the spanning tree).

3 Algorithm for 2-SAT

Problem definition: In the 2-SAT problem, the input is a 2-CNF formula F with m clauses over n boolean variables, and the goal is the decide if F is satisfiable.

This problem is in P (notice it is not MAX-2SAT). In contrast, for every $k \geq 3$, the k-SAT problem is NP-hard.

Exer: Show that 2-SAT can be solved in polynomial time.

Algorithm A:

1. start with an arbitrary assignment a

2. while the assignment a does not satisfy F

2.1 pick an arbitrary unsatisfied clause, pick one of its variables at random, and flip its value

3. output the satisfying assignment

Formally, if F is unsatisfiable, then it will never terminate, hence we need to stop the algorithm at some point, and we can call that Algorithm A'. Anyway, our goal is to prove that if F is satisfiable, then the algorithm will (probably) find one quickly enough.

Theorem [Papadimitriou, 1991]: The expected number of iterations for the above algorithm to find a satisfying assignment, assuming one exists, is $O(n^2)$.

Proof: Was seen in class.