Advanced Graph Algorithms and Optimization Spring 2020

Course Introduction

Rasmus Kyng *Lecture 1 — Wednesday, February 19th*

1 Overview

This course will take us quite deep into modern approaches to graph algorithms using convex optimization techniques. By studying convex optimization through the lens of graph algorithms, we'll try to develop a deeper understanding of fundamental phenomena in optimization. Much of our time will be devoted to flow problems on graphs. We will not only be studying these problems for their own sake, but also because they often provide a useful setting for thinking more broadly about optimization.

The course will cover some traditional discrete approaches to various graph problems, especially flow problems, and then contrast these approaches with modern, asymptotically faster methods based on combining convex optimization with spectral and combinatorial graph theory.

2 Electrical flows and voltages - a graph problem from middle school?

We will dive right into graph problems by considering how electrical current moves through a network of resistors.

First, let us recall some middle school physics. If some of these things don't make sense two you, don't worry, in less than paragraph from here, we'll be make to safely doing math.

Recall that a typical battery that buy from Migros has two endpoints, and produces what is called a *voltage difference* between these endpoints.

One end of the battery will have a positive charge (I think that means an excess of positrons[1]), and the other a negative charge. If we connect the two endpoints with a wire, then a current will flow from one end of the battery to the other in an attempt to even out this imbalance of charge.

[1] I'm joking, of course! Try Wikipedia if you want to know more. However, you will not need it for this class.

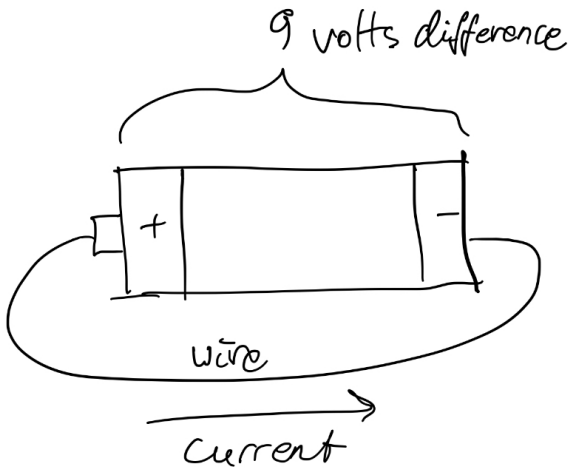

Figure 1: A 9 volts battery with a wire attached.

We can also imagine a kind of battery that tries to send a certain amount of current the wires between its endpoints, e.g. 1 unit of charge per unit of time. This will be a little more convenient to work with, so let us focus on that case.

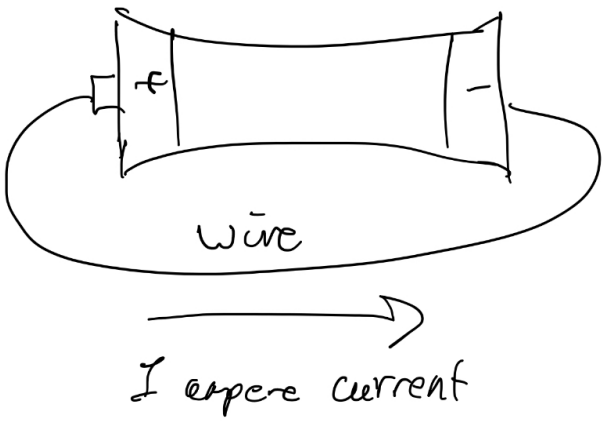

Figure 2: A 1 ampere battery with a wire attached.

A *resistor* is a piece of wire that connects two points u and v, and is completely described by a single number r called its *resistance*.

If the voltage difference between the endpoints of the resistor is x, and the resistance is r then this will create a flow of charge per unit of time of $f = x/r$. This is called Ohm's Law.

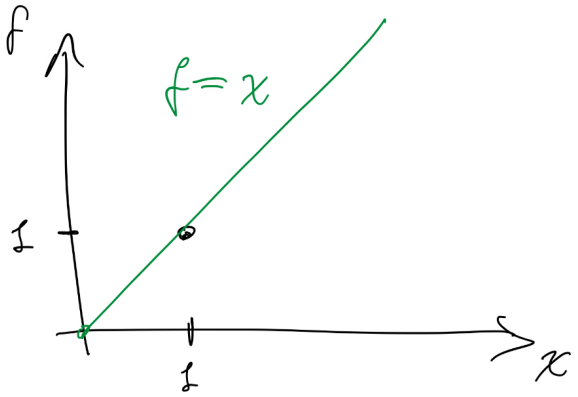

Figure 3: Ohm's Law for a resistor with resistance $r = 1$.

Suppose we set up a bunch of wires that route electricity from our current source s to our current sink t in some pattern:

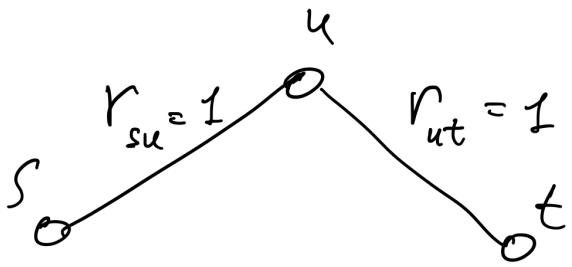

Figure 4: A path of two resistors.

We have one unit of charge flowing out of s per unit of time, and one unit coming into t. Because charge is conserved, the current flowing into any other point u must equal the amount flowing out of it. This is called Kirchoff's Current Law.

To send one unit of current from s to t, we must be sending it first form s to u and then from u to t. So the current on edge (s, u) is 1 and the current on (u, t) is 1. By Ohm's Law, the voltage difference must also be 1 across each of the two wires. Thus if the voltage is x at s, it must be $x+1$ at u and $x+2$ at t. What is x? It turns out it doesn't matter: We only care about the differences. So let us set $x = 0$.

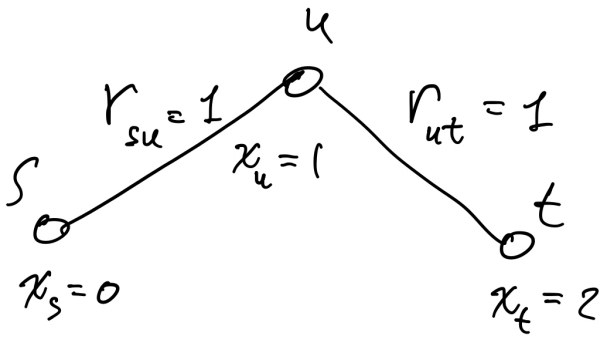

Figure 5: A path of two resistors.

Let us try one more example:

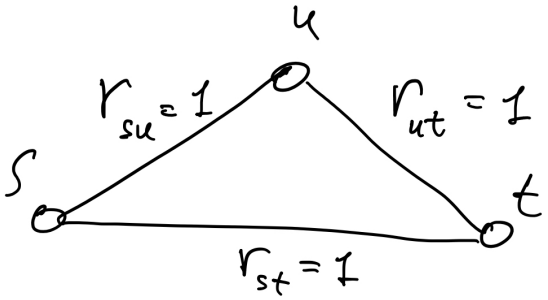

Figure 6: A network with three resistors.

How much flow will go directly from s to t and how much via u?

Well, we know what the net current flowing into and out of each vertex must be, and we can use to set up some equations. Let us say the voltage at s is x_s, at u is x_u and at t is x_t.

- Net current at s: $-1 = (x_s - x_t) + (x_s - x_u)$
- Net current at u: $0 = (x_u - x_s) + (x_u - x_t)$
- Net current at t: $1 = (x_t - x_s) + (x_t - x_u)$

The following is a solution: $x_s = 0$, $x_u = \frac{1}{3}$, $x_t = \frac{2}{3}$. And as before, we can shift all the voltages by some constant x and get another solution $a = x + 0$, $x_u = x + \frac{1}{3}$, $x_t = x + \frac{2}{3}$. You might want to convince yourself that these are the only solutions.

Electrical flows in general graphs. Do we know enough to calculate the electrical flow in some other network of resistors? To answer this, let us think about the network as a graph. Consider a undirected graph $G = (V, E)$ with $|V| = n$ vertices and $|E| = m$ edges, and let us assume G is connected. Let's associate a resistance $\boldsymbol{r}(e) > 0$ with every edge $e \in E$.

To keep track of the direction of the flow on each edge, it will be useful to assign an arbitrary direction to every edge. So let's do that, but remember that this is just a bookkeeping tool that

helps us track where flow is going.

A *flow* in the graph is a vector $\boldsymbol{f} : \mathbb{R}^E$. The *net flow* of $\boldsymbol{f}$ at a vertex $u \in V$ is defined as $\sum_{v \to u} \boldsymbol{f}(v,u) - \sum_{u \to v} \boldsymbol{f}(u,v)$.

We say a flow routes the demands $\boldsymbol{d} \in \mathbb{R}^V$ if the net flow at every vertex v is $\boldsymbol{d}(v)$.

We can assign a voltage to every vertex $\boldsymbol{x} \in R^V$. Ohm's Law says that the electrical flow induced by these voltages will be $\boldsymbol{f}(u,v) = \frac{1}{r_{(u,v)}}(\boldsymbol{x}(u) - \boldsymbol{x}(v))$.

Say we want to route unit of current from vertex $s \in V$ to vertex $t \in V$. As before, we can write an equation for every vertex saying that the voltage differences must produce the desired net current:

- Net current at s: $-1 = \sum_{(s,v)} \frac{1}{\boldsymbol{r}(s,v)}(\boldsymbol{x}(s) - \boldsymbol{x}(v))$
- Net current at $u \in V \setminus \{s,t\}$: $0 = \sum_{(u,v)} \frac{1}{\boldsymbol{r}(u,v)}(\boldsymbol{x}(u) - \boldsymbol{x}(v))$
- Net current at t: $1 = \sum_{(t,v)} \frac{1}{\boldsymbol{r}(t,v)}(\boldsymbol{x}(t) - \boldsymbol{x}(v))$

This gives us n constraints, exactly as many as we have voltage variables. However we have to be a little careful when trying to conclude that a solution exists, yielding voltages $\boldsymbol{x}$ that gives induce an electrical flow routing the desired demand.

You will prove in the exercises (as part of this week's Exercise 3) that a solution $\boldsymbol{x}$ exists. The proof requires two important observations: Firstly that the graph is connected, and secondly that summed over all vertices, the net demand is zero, i.e. as much flow is coming into the network as is leaving it.

The incidence matrix and the Laplacian matrix. To have a more compact notation for net flow constraints, we also introduce the *edge-vertex incidence matrix* of the graph, $\boldsymbol{B} \in \mathbb{R}^{V \times E}$.

$$\boldsymbol{B}(v,e) = \begin{cases} 1 & \text{if } e = (u,v) \\ -1 & \text{if } e = (v,u) \\ 0 & \text{o.w.} \end{cases}$$

Now we can express the net flow constraint that $\boldsymbol{f}$ routes $\boldsymbol{d}$ by

$$\boldsymbol{B}\boldsymbol{f} = \boldsymbol{d}.$$

This is also called a conservation constraint. In our examples so far, we have $\boldsymbol{d}(s) = -1$, $\boldsymbol{d}(t) = 1$ and $\boldsymbol{d}(u) = 0$ for all $u \in V \setminus \{s,t\}$.

If we let $\boldsymbol{R} = \mathrm{diag}_{e \in E}\, \boldsymbol{r}(e)$ then Ohm's law tells us that $\boldsymbol{f} = \boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{x}$. Putting these observations together, we have $\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{x} = \boldsymbol{d}$. The voltages $\boldsymbol{x}$ that induce $\boldsymbol{f}$ must solve this system of linear equations, and we can use that to compute both $\boldsymbol{x}$ and $\boldsymbol{f}$. It is exactly the same linear equation as the one we considered earlier. We can show a that for a connected graph, a solution $\boldsymbol{x}$ exists if and only if the flow into the graph equals the net flow out, which we can express as $\sum_v \boldsymbol{d}(v) = 0$ or $\mathbf{1}^\top \boldsymbol{d} = 0$. You will show this as part of Exercise 3. This also implies that an electrical flow routing $\boldsymbol{d}$ exists if and only if the net flow into the graph equals the net flow out, which we can express as $\mathbf{1}^\top \boldsymbol{d} = 0$.

The matrix $\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^\top$ is called the *Laplacian* of the graph and is usually denoted by $\boldsymbol{L}$.

An optimization problem in disguise. So far, we have looked at electrical voltages and flows as arising from a set of linear equations – and it might not be apparent that this has anything to do with optimization. But transporting current through a resistor requires energy, which will be dissipated as heat by the resistor (i.e. it will get hot!). If we send a current of f across a resistor with a potential drop of x, then the amount of energy spent per unit of time by the resistor will be $f \cdot x$. This is called Joule's Law. Applying Ohm's law to a resistor with resistance r, we can also express this energy per unit of time as $f \cdot x = x^2/r = r \cdot f^2$. Since we aren't bothering with units, we will even forget about time, and refer to these quantities as "energy", even though a physicist would call them "power".

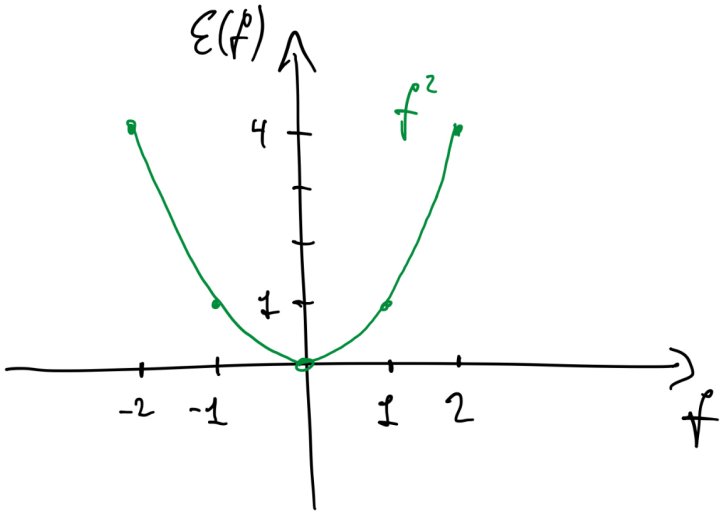

Figure 7: Energy has a function of flow in a resistor with resistance $r = 1$.

Now, another interesting question would seem to be: If we want to find a flow routing a certain demand $\boldsymbol{d}$, how should have flow behave in order to minimize the the electrical energy spent routing the flow? We can phrase this as an optimization problem:

$$\min_{\boldsymbol{f} \in \mathbb{R}^E} \sum_e \boldsymbol{r}(e)\boldsymbol{f}(e)^2$$
$$\text{s.t. } \boldsymbol{B}\boldsymbol{f} = \boldsymbol{d}.$$

We call this problem *electrical energy-minimizing flow*. It turns out, that the flow $\boldsymbol{f}^*$ that minimizes the electrical energy among all flows that satisfy $\boldsymbol{B}\boldsymbol{f} = \boldsymbol{d}$ is precisely the electrical flow.

A pair of problems. What about our voltages, can we also get them from some optimization problem? Well, we can work backwards from the fact that our voltages solve the equation $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{d}$. Consider the function $c(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{d}$. We should ask ourselves some questions about this function $c : \mathbb{R}^V \to \mathbb{R}$. Is it continuous and continuously differentiable? The answer to this is yes, and that is not hard to see. Does the function have a minimum? This is maybe not immediately clear, but the minimum does indeed exist.

When this is minimized, the derivative of $c(\boldsymbol{x})$ with respect to each coordinate of $\boldsymbol{x}$ must be zero. This condition yields exactly the system of linear equations $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{d}$. You will confirm this in Exercise 4.

Based on our derivative considitio for the optimum can also express the electrical voltages as the solution to an optimization problem, namely

$$\min_{\boldsymbol{x}\in\mathbb{R}^V} \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{d}$$

As you are probably be aware, having the derivative of each coordinate equal zero is not a sufficient condition for being at the optimum of a function[2]. It is also interesting to know whether *all* solutions to $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{d}$ are in fact minimizers of c. The answer is yes, and will see some very general tools for proving statements like this in a lecture or two.

Altogether, we can see that routing electrical current through a network of resistors leads to a *pair* of optimization problems, let's call them $\boldsymbol{f}^*$ and $\boldsymbol{x}^*$, and that the solutions to the two problems are related, in our case through the equation $\boldsymbol{f}^* = \boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{x}^*$. In Exercise 5, you will explore this relationship more.

This turns out to be an instance of a much broader phenomenon, known as Lagrangian duality, which allows to to learn a lot about many optimization problems by studying two related pairs of problems.

Solving $\boldsymbol{L}\boldsymbol{x} = \boldsymbol{d}$. Given a graph G with resistances for the edges, and some net flow vector $\boldsymbol{d}$, how quickly can we compute $\boldsymbol{x}$? Broadly speaking, there are two very different families of algorithms we could use to try to solve this problem.

Either, we could solve the linear equation using something like *Gaussian Elimination* to compute an exact solution.

Alternatively, we could start with a guess at a solution, e.g. $\boldsymbol{x}_0 = \boldsymbol{0}$, and then we could try to make a change to $\boldsymbol{x}_0$ to reach a new point $\boldsymbol{x}_1$ with a lower value of $c(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x} - \boldsymbol{x}^\top \boldsymbol{d}$, i.e. $c(\boldsymbol{x}_1) < c(\boldsymbol{x}_0)$. If we repeat a process like that for enough steps, say t, hopefully we eventually reach $\boldsymbol{x}_t$ with $c(\boldsymbol{x}_t)$ close to $c(\boldsymbol{x}^*)$, where $\boldsymbol{x}^*$ is a minimizer of $c(\boldsymbol{x})$ and hence $\boldsymbol{L}\boldsymbol{x}^* = \boldsymbol{d}$. Now, we also need to make sure that $c(\boldsymbol{x}_t) \approx c(\boldsymbol{x}^*)$ implies that $\boldsymbol{L}\boldsymbol{x}_t \approx \boldsymbol{d}$ in some useful sense.

One of the most basic algorithms in this framework of "guess and adjust" is called *Gradient Descent*, which will study in two weeks. The rough idea is the following: if we make a very small from $\boldsymbol{x}$ to $\boldsymbol{x} + \boldsymbol{\delta}$, then a multivariate Taylor expansion suggests that $c(\boldsymbol{x} + \boldsymbol{\delta}) - c(\boldsymbol{x}) \approx \sum_{v\in V} \boldsymbol{\delta}(v)\frac{\partial c(\boldsymbol{x})}{\partial \boldsymbol{x}(v)}$.

If we are dealing with smooth convex function, this quantity is negative if we let $\boldsymbol{\delta}(v) = -\epsilon \cdot \frac{\partial c(\boldsymbol{x})}{\partial \boldsymbol{x}(v)}$ for some small enough ϵ so the approximation holds well. So we should be able to make progress by taking a small step in this direction. That's Gradient Descent! The name comes from the vector of partial derivatives, which is called the gradient.

As we will see later in this course, understanding electrical problems from an optimization perspective is crucial to developing fast algorithms for computing electrical flows and voltages, but to do very well, we also need to borrow some ideas from Gaussian Elimination.

What running times do different approaches get?

[2]Consider the function in one variable $c(x) = x^3$.

1. Using Gaussian Elimination, we can find $\boldsymbol{x}$ s.t. $\boldsymbol{Lx} = \boldsymbol{d}$ in $O(n^3)$ time and with asymptotically faster algorithms based on matrix multiplication, we can bring this down to roughly $O(n^{2.372})$.

2. Meanwhile Gradient Descent will get a running time of $O(n^3 m)$ or so – at least this is a what a simple analysis suggests.

3. However, we can do much better: By combining ideas from both algorithms, and a bit more, we can get $\boldsymbol{x}$ up to very high accuracy in time $O(m \log^c n)$ where c is some small constant.

3 Convex optimization

Recall our plot in Figure 7 of the energy required to route a flow f across a resistor with resistance r, which was $\mathcal{E}(f) = r \cdot f^2$. We see that the function has a special structure: the graph of the function sits below the line joining any two points $(f, \mathcal{E}(f))$ and $(g, \mathcal{E}(g))$. A function $\mathcal{E} : \mathbb{R} \to \mathbb{R}$ that has this property is said to be convex.

Figure 8 shows a the energy as a function of flow, along with two points $(f, \mathcal{E}(f))$ and $(g, \mathcal{E}(g))$. We see the function sits below the line segment between these points.

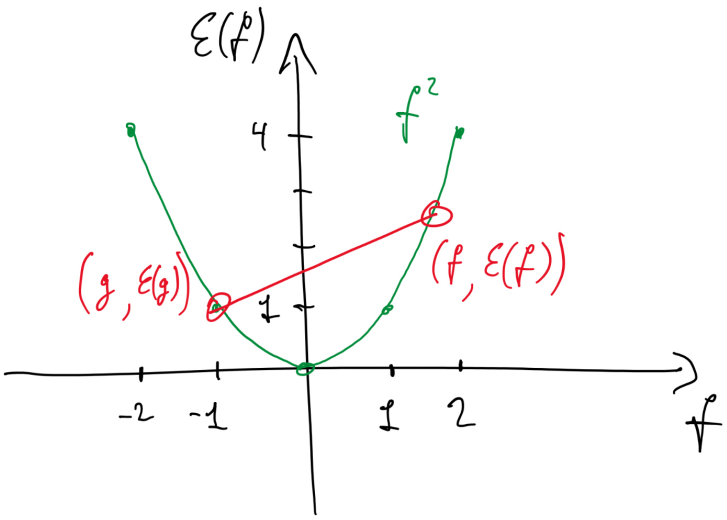

Figure 8: Energy has a function of flow in a resistor with resistance $r = 1$. The function is convex.

We can also interpret this condition as saying that for all $\theta \in [0, 1]$

$$\mathcal{E}(\theta f + (1-\theta)g) \leq \theta \mathcal{E}(f) + (1-\theta)\mathcal{E}(g).$$

This immediately generalizes to functions $\mathcal{E} : \mathbb{R}^m \to \mathbb{R}$.

A *convex set* is a subset of $S \subseteq \mathbb{R}^m$ s.t. if $\boldsymbol{f}, \boldsymbol{g} \in S$ then for all $\theta \in [0, 1]$ we have $\theta \boldsymbol{f} + (1-\theta)\boldsymbol{g} \in S$.

Figure 9 shows some examples of sets that are and aren't convex.

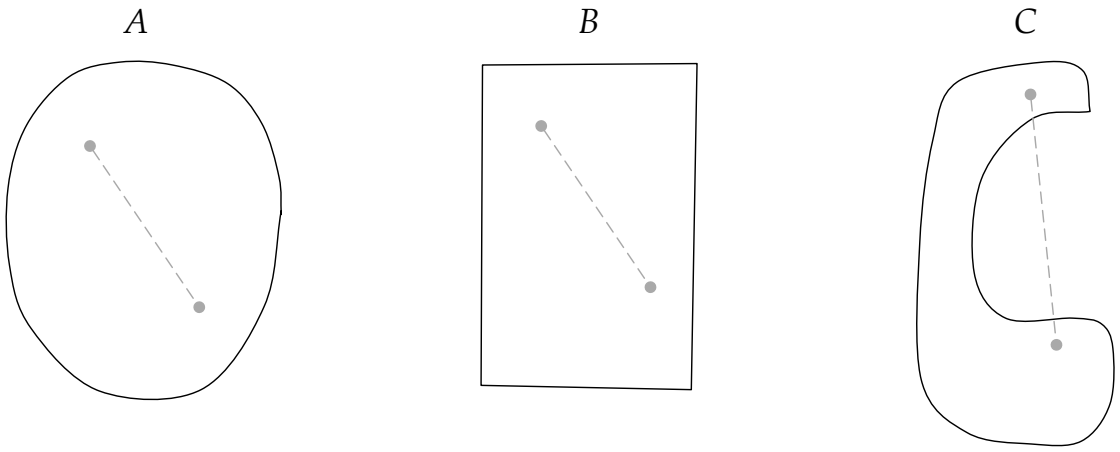

Figure 9: A depiction of convex and non-convex sets. The sets A and B are convex since the straight line between any two points inside them is also in the set. The set C is not convex.

Convex functions and convex sets are central to optimization, because for most problems of minimization a convex function over a convex set, we can develop fast algorithms [3].

So why convex functions and convex sets? One important reason is that for a convex function defined over a convex feasible set, any local minimum is also a global minimum, and this fact makes searching for an optimal solution computationally easier. In fact, this is closely related to why Gradient Descent works well on many convex functions.

Notice that the set $\{\boldsymbol{f} : \boldsymbol{B}\boldsymbol{f} = \boldsymbol{d}\}$ is convex, i.e. the set of all flows that route a fixed demand $\boldsymbol{d}$ is convex. It is also easy to verify that $\mathcal{E}(\boldsymbol{f}) = \sum_e \boldsymbol{r}(e)\boldsymbol{f}(e)^2$ is a convex function, and hence finding an electrical flow is an instance of convex minmization:

4 More graph optimization problems

Maximum flow. Again, let $G = (V, E)$ be an undirected, connected graph with n vertices and m edges. Suppose we want to find a flow $\boldsymbol{f} \in \mathbb{R}^E$ that routes $\boldsymbol{d}$, but instead of trying to minimize electrical energy, we try to pick an $\boldsymbol{f}$ that minimizes the largest amount of flow on any edge, i.e. $\max_e |\boldsymbol{f}_e|$ – which we also denote by $\|\boldsymbol{f}\|_\infty$. We can write this problem as

$$\begin{aligned} &\min_{\boldsymbol{f} \in \mathbb{R}^E} \|\boldsymbol{f}\|_\infty \\ &\text{s.t. } \boldsymbol{B}\boldsymbol{f} = \boldsymbol{d} \end{aligned}$$

This problem is known as the Minimum Congested Flow Problem[4]. It is equivalent to the more famous Maximum Flow Problem.

[3] There are some convex optimization problems that are NP-hard. That said, polynomial time algorithms exist for almost any convex problem you can come up with. The most general polynomial time algorithm for convex optimization is probably the Ellipsoid Method.

[4] This version is called undirected, because the graph is undirected, and *uncapacitated* because we are aiming for the same bound on the flow on all edges.

The behavior of this kind of flow is very different than electrical flow. Consider the question of whether a certain demand can be routed $\|\boldsymbol{f}\|_\infty \leq 1$. Imagine sending goods from a source s to a destination t using a network of train lines that all have the same capacity and asking whether the network is able to route the goods at the rate you want: This boils down to whether routing exists with $\|\boldsymbol{f}\|_\infty \leq 1$, if we set it up right.

We have a very fast, convex optimization-based algorithm for Minimum Congested Flow: In $m\epsilon^{-1}\log^{O(1)} n$ time, we can find a flow $\tilde{\boldsymbol{f}}$ s.t. $\boldsymbol{B}\tilde{\boldsymbol{f}} = \boldsymbol{d}$ and $\left\|\tilde{\boldsymbol{f}}\right\|_\infty \leq (1+\epsilon)\|\boldsymbol{f}^*\|_\infty$, where $\boldsymbol{f}^*$ is an optimal solution, i.e. an actual minimum congestion flow routing $\boldsymbol{d}$.

But what if we want ϵ to be very small, e.g. $1/m$? Then this running time isn't so good anymore. But, in this case, we can use another algorithm, that finds an optimal flow $\boldsymbol{f}^*$ *exactly*, in time[5] $m^{10/7}\log^{O(1)} n$.

Just as the electrical flow problem had a dual voltage problem, so maximum flow has a dual voltage problem, which is know as the *s-t* minimum cut problem.

Maximum flow, with directions and capacities. We can make the maximum flow problem harder by introducing directed edges: To do so, we allow edges in both directions to exist between a vertex to exist, and we require that that flow on a directed edge is always non-negative. So now $G = (V, E)$ is a directed graph. We can also make the problem harder by introducing capacities. We define a capacity vector $\boldsymbol{c} \in \mathbb{R}^E \geq \boldsymbol{0}$ and require now try to minimize $\left\|\boldsymbol{C}^{-1}\boldsymbol{f}\right\|_\infty$, where $\boldsymbol{C} = \mathrm{diag}_{e\in E}\, \boldsymbol{c}(e)$. Then our problem becomes

$$\begin{aligned} \min_{\boldsymbol{f}\in\mathbb{R}^E} & \left\|\boldsymbol{C}^{-1}\boldsymbol{f}\right\|_\infty \\ \text{s.t. } & \boldsymbol{B}\boldsymbol{f} = \boldsymbol{d} \\ & \boldsymbol{f} \geq \boldsymbol{0}. \end{aligned}$$

For this capacitated, directed maximum flow problem, our best algorithms run in about $O(m\sqrt{n})$ time[6], even if we are willing to accept fairly low accuracy solution. If the capacities are allowed to be exponentially large, the best running time we can get is $O(mn)$. For this problem, we do not yet know how to improve over classical combinatorial algorithms using convex optimization.

Multi-commodity flow. We can make the even harder still, by simultaneously trying to route to types of flow (imagine pipes with Coke and Pepsi). Our problem now looks like

$$\begin{aligned} \min_{\boldsymbol{f}_1,\boldsymbol{f}_2\in\mathbb{R}^E} & \left\|\boldsymbol{C}^{-1}(\boldsymbol{f}_1+\boldsymbol{f}_2)\right\|_\infty \\ \text{s.t. } & \boldsymbol{B}\boldsymbol{f}_1 = \boldsymbol{d}_1 \\ & \boldsymbol{B}\boldsymbol{f}_2 = \boldsymbol{d}_2 \\ & \boldsymbol{f}_1, \boldsymbol{f}_2 \geq \boldsymbol{0}. \end{aligned}$$

Solving this problem to high accuracy is essentially as hard as solving a general linear program! We should see later in the course how to make this statement precise.

[5] And there's even a paper on arXiv.org that brings this further down to $m^{11/8}\log^{O(1)} n$.

[6] Provided the capacities are integers satisfying a condition like $\boldsymbol{c} \leq n^{100}\boldsymbol{1}$.

If we in the above problem additionally require that our flows must be integer valued, i.e. $\boldsymbol{f}_1, \boldsymbol{f}_2 \in \mathbb{N}_0$, then the problem becomes NP-complete.

Random walks in a graph. Google famously uses[7] the PageRank problem to help decide how to rank their search results. This problem essentially boils down to computing the *stable distribution* of a random walk on a graph. Suppose $G = (V, E)$ is a directed graph where each edge outgoing edge (v, u), which we will define as going from u to v, has a transition probability $p_{(v,u)} > 0$ s.t. $\sum_{z \leftarrow u} p_{(z,u)} = 1$. We can take a step of a random walk on the vertex set by starting at some vertex $u_0 = u$, and then randomly picking one of it the outgoing edges (v, u) with probability $p_{(v,u)}$ and move to the chosen vertex $u_1 = v$. Repeating this procedure, to take a step from the next vertex u_1, gives us a *random walk* in the graph, a sequence of vertices $u_0, u_1, u_2 \ldots, u_k$.

We let $\boldsymbol{P} \in \mathbb{R}^{V \times V}$ be the matrix of transition probabilities given by

$$\boldsymbol{P}_{vu} = \begin{cases} p_{(v,u)} & \text{for } (u, v) \in E \\ 0 & \text{o.w.} \end{cases}$$

Any probability distribution over the vertices can be specified by a vector $\boldsymbol{p} \in \mathbb{R}^V$ where $\boldsymbol{p} \geq \mathbf{0}$ and $\sum_v \boldsymbol{p}(v) = 1$. We say that probability distribution $\boldsymbol{\pi}$ on the vertices is a *stable distribution* of the random walk if $\boldsymbol{\pi} = \boldsymbol{P}\boldsymbol{\pi}$. A strongly connected graph always has exactly one stable distribution.

How quickly can we compute the stable distribution of a general random walk? Under some mild conditions on the stable distribution[8], we can find a high accuracy approximation of $\boldsymbol{\pi}$ in time $O(m \log^c n)$ for some constant c.

This problem does not easily fit in a framework of convex optimization, but nonetheless, our fastest algorithms for it use ideas from convex optimization.

Topics in this course

In this course, we will try to address the following questions.

1. What are the fundamental tools of fast convex optimization?
2. What are some problems we can solve quickly on graphs using optimization?
3. What can graphs teach us about convex optimization?
4. What algorithm design techniques are good for getting algorithms that quickly find a crude approximate solution? And what techniques are best when we need to get a highly accurate answer?
5. What is special about flow problems?

[7] At least they did at some point.

[8] Roughly something like $\max_v 1/\boldsymbol{\pi}(v) \leq n^{100}$.