1 Some loose ends from last time

1.1 A lemma concerning greedy algorithms and matroids

Lemma 1. *Let $\mathscr{M} = (\mathscr{U}, \mathscr{I})$ be any matroid with a cost function $c : \mathscr{U} \to \mathbb{R}$. If X_k is a minimum-cost independent set of cardinality $k < \text{rank}(\mathscr{M})$, then there exists an element y such that $X_k \cup \{y\}$ is a minimum-cost independent set of cardinality $k+1$.*

Proof. Let Y be any minimum-cost independent set of cardinality $k+1$. By the exchange axiom, there exists an element $y \in Y \setminus X_k$ such that $X_k \cup \{y\}$ is independent. Let $Y_k = Y \setminus \{y\}$. Note that $Y_k \in \mathscr{I}$ by the hereditary property of independent sets. Since X_k is a minimum-cost independent set of cardinality k, we have

$$c(X_k) \leq c(Y_k)$$

and hence

$$c(X_k \cup \{y\}) \leq c(Y_k \cup \{y\}) = c(Y).$$

As Y was a minimum-cost independent set of cardinality $k+1$, we may conclude that $X_k \cup \{y\}$ is also a minimum-cost independent set of cardinality $k+1$. □

Algorithm 1 Greedy$(\mathscr{M}, c)$

```
1: X_0 = ∅
2: k = 0
3: Sort the elements of 𝒰 by cost.
4: for all x ∈ 𝒰 in increasing order of cost do
5:    if X_k ∪ {x} ∈ ℐ then
6:       X_{k+1} ← X_k ∪ {x}
7:       k ← k + 1
8:    end if
9: end for
10: return X_k
```

Theorem 2. *Algorithm* Greedy$(\mathscr{M}, c)$ *outputs a minimum-cost basis of $\mathscr{M}$.*

Proof. Let X be the set that the algorithm outputs. First we will prove that X is a basis. From the pseudocode, it is clear that $X \in \mathscr{I}$. If X is not a basis then there is some element $x \in \mathscr{U} \setminus X$ such that $X \cup \{x\} \in \mathscr{I}$. Let j denote the value of the variable k when the algorithm reached the loop iteration that processed element x. We have $X_j \cup \{x\} \subseteq X \cup \{x\}$ and $X \cup \{x\} \in \mathscr{I}$, hence $X_j \cup \{x\} \in \mathscr{I}$. However, this means that the algorithm would have incorporated x into X_{j+1}, and x would have been in X, contradicting our choice of x.

Next we will prove that X is a minimum-cost basis. The proof is by induction on k, the induction hypothesis being that X_k is a minimum-cost independent set of cardinality k. The

base case $k = 0$ is trivial. If X_k is not a minimum-cost independent set of cardinality k, then by Lemma 1 there is a k-element independent set $X_{k-1} \cup \{y\} \in \mathscr{I}$ whose cost is strictly less than $c(X_k)$. Letting x denote the unique element of $X_k \setminus X_{k-1}$, we find that $c(y) < c(x)$ and hence the algorithm processed the element y earlier than x and chose not to include it in X_k. As before, we can let j denote the value of the variable k at the time the algorithm processed y, and we can conclude that $X_j \cup \{y\} \notin \mathscr{I}$ contradicting the fact that $X_{k-1} \cup \{y\} \in \mathscr{I}$. □

1.2 Kruskal's algorithm

When $\mathscr{M}$ is the graphic matroid of an undirected graph G, the algorithm $\mathsf{Greedy}(\mathscr{M}, c)$ is known as Kruskal's Algorithm. It iterates through edges of the graph in order of increasing cost, maintaining a spanning forest (initially empty) by inserting every edge whose endpoints belong to different components of the spanning forest. Sorting the edges requires $O(m \log n)$ time, and it turns out that the remaining steps of the algorithm can be implemented to run in time $O(m \log n)$ or even faster. For example, we can maintain the vertex set of each component of the forest as a linked list, along with a pointer from each element back to the head of the list (the *root* of the component), as well as a counter that stores the number of vertices in the component. Initially we have n lists each containing a single vertex. To test whether the endpoints of an edge belong to the same component, we test whether their pointers point to the same root; this takes constant time. To merge components, we append the shorter list to the longer one, update the pointers of all vertices in the shorter list to point to the head of the longer one, and update the counter of the merged list to equal the sum of the two counters. All the steps involved in a merge take $O(1)$ except for updating the pointers of all vertices in the shorter list. However, the pointer for vertex v is updated at most $\log n$ in the course of executing the algorithm because the size of v's component at least doubles each time its pointer is updated. Consequently, the combined number of pointer updates is $O(n \log n)$.

To summarize, Kruskal's algorithm begins with a sorting step whose running time is $O(m \log n)$, and the running time of the remaining steps is bounded by $O(m + n \log n)$.

2 Matroids and Exchange Arguments

The proof of correctness of Kruskal's Algorithm using Lemma 1 is reminiscent of the method of analysis denoted by "greedy stays ahead" in the textbook by Kleinberg & Tardos. Interestingly, the analysis of Kruskal's Algorithm in that book illustrates a different principle: an exchange argument. In terms of minimum spanning trees, the exchange argument boils down to an assertion that for every cut C and every minimum spanning tree T, the set $T \cap C$ must contain a minimum-cost element of C. This exchange argument turns out to be more flexible than the "greedy stays ahead" approach; for example, it justifies the correctness of a number of other algorithms such as Prim's. How can we recast this exchange argument in the abstract language of matroids?

It turns out that the proper abstract formulation is expressed in terms of *circuits* and *cocircuits* — abstract analogues of the notions of *cycle* and *cut* in graph theory — which we now define.

2.1 Circuits and cocircuits

Definition 1. A *circuit* in a matroid $\mathscr{M} = (\mathscr{U}, \mathscr{I})$ is a minimal dependent set, i.e. a set $C \notin \mathscr{I}$ such that every proper subset of C belongs to $\mathscr{I}$. A *cocircuit* is a minimal set that intersects every basis.

For example, a circuit in a graphic matroid is an edge set that forms a simple cycle. A cocircuit should be thought of as a cut. More precisely, an edge set C forms a cocircuit of the graphic matroid of G if and only if there are two disjoint nonempty vertex sets A, B such that $A \cup B$ is the vertex set of a connected component of G, and C is the set of all edges from A to B.

Lemma 3. *If $\mathscr{M} = (\mathscr{U}, \mathscr{I})$ is a matroid, B is a basis of $\mathscr{M}$, and x is any element of $\mathscr{U} \setminus B$, then there is a unique circuit C contained in $B \cup \{x\}$, called the* fundamental circuit *of x with respect to B. If y is any element of C, then $(B \cup \{x\}) \setminus \{y\}$ is a basis of $\mathscr{M}$.*

Proof. Let C be *any* minimal dependent subset of $B \cup \{x\}$. Such a C must exist, because $B \cup \{x\}$ itself is dependent, and we can iteratively delete elements from $B \cup \{x\}$ until we arrive at a minimal dependent subset. For any $y \in C$, the set $C \setminus \{y\}$ is independent. Applying the exchange axiom repeatedly, we can find a set $D \subset B$, disjoint from $C \setminus \{y\}$, such that $E := (C \setminus \{y\}) \cup D$ is independent and has $|B|$ elements. Hence, E consists of all but one element of $B \cup \{x\}$. The missing element must be y, since C is dependent and hence not contained in E. Therefore, $E = (B \cup \{x\}) \setminus \{y\}$ and this set is independent. Finally, it follows that C is the *unique* minimal dependent subset of $B \cup \{x\}$, since we have shown that a subset of $B \cup \{x\}$ which is missing even a single element of C must be independent. □

2.2 Contractions

Definition 2. If $\mathscr{M} = (\mathscr{U}, \mathscr{I})$ is a matroid, and $S \subseteq \mathscr{U}$ is any set, then $\mathscr{M}/S = (\mathscr{V}, \mathscr{J})$ is the matroid with ground set $\mathscr{V} = \mathscr{U} \setminus S$, and independent sets

$$\mathscr{J} = \{T \subseteq \mathscr{V} \mid T \cup S_0 \in \mathscr{I}\}, \tag{1}$$

where S_0 is any maximal independent subset of S.

Lemma 4. *For any fixed maximal independent subset $S_0 \subseteq S$, the collection $\mathscr{J}$ defined by* (1) *satisfies the matroid axioms. Furthermore, the contents of $\mathscr{J}$ don't depend on the choice of S_0.*

Proof. It is trivial to check that, for any fixed choice of S_0, the matroid axioms are satisfied by $\mathscr{J}$. To see that $\mathscr{J}$ is independent of S_0, consider some other maximal independent set $S_1 \subseteq S$, and suppose that $T \cup S_1 \in \mathscr{I}$. By the exchange axiom, starting from S_0 we can add elements of $T \cup S_1$ until we obtain an independent set $U \supseteq S_0$ whose cardinality equals that of $T \cup S_1$. Note that U cannot contain any elements of $S_1 \setminus S_0$, as then we would have $U \cap S$ would be an independent set satisfying $S_0 \subsetneq U \cap S \subseteq S$, contrary to our assumption that S_0 is a maximal independent subset of S. Therefore U must be equal to $T \cup S_0$, hence $T \cup S_0$ is independent as desired. □

As an example of the definition of contraction, if $\mathscr{M}$ is the graphic matroid of a graph G, and S is an edge set in G, then $\mathscr{M}/S$ is the graphic matroid of the graph obtained from G by contracting each edge in S, i.e., merging its endpoints into a single vertex. (This contraction

process can create parallel edges and self-loops. One must be careful to say how we deal with such pathologies. When G is a graph with parallel edges and self-loops, its graphic matroid is defined, as always, by defining an independent set to be an edge set without cycles. A self-loop is considered to be a 1-cycle, and a pair of parallel edges are considered to be a 2-cycle.)

Lemma 5. *If $\mathscr{M} = (\mathscr{U}, \mathscr{I})$ is a matroid, D is a cocircuit in $\mathscr{M}$, and $S \subset \mathscr{U}$ is disjoint from D, then D is also a cocircuit in $\mathscr{M}/S$.*

Proof. If B is a basis of $\mathscr{M}/S$ and S_0 is a maximal independent subset of S, then $B \cup S_0$ is a basis of $\mathscr{M}$ and consequently intersects D. As S_0 is disjoint from D, it must be the case that B intersects D. This verifies one half of the definition of cocircuit. It remains for us to verify the minimality, i.e. for every $x \in D$ we must prove that there exists a basis of $\mathscr{M}/S$ that is disjoint from $D \setminus \{x\}$. This is easy: let S_0 be a maximal independent subset of S (hence disjoint from $D \setminus \{x\}$) and let B_0 be a basis of $\mathscr{M}$ that is disjoint from $D \setminus \{x\}$. Using the exchange axiom, add elements of B_0 to S_0 until we have a basis $B_1 \supseteq S_0$. By construction, B_1 is disjoint from $D \setminus \{x\}$. If we now define B to be $B_1 \setminus S_0$, it is a basis of $\mathscr{M}/S$ and it is disjoint from $D \setminus \{x\}$. □

Lemma 6. *If C is a circuit and D is a cocircuit, then $|C \cap D| \neq 1$.*

Proof. Assume, by way of contradiction, that $C \cap D = \{x\}$. Then $S = C \setminus \{x\}$ is disjoint from D, so D is a cocircuit in $\mathscr{M}/S$. Consequently there is a basis B of $\mathscr{M}/S$ that is disjoint from $D \setminus \{x\}$. The single-element set $\{x\}$ is not independent in $\mathscr{M}/S$, since $S \cup \{x\} = C$ is a circuit of $\mathscr{M}$. Consequently, $x \notin B$ which means that B is disjoint from D. This violates Lemma 5, which asserts that D is a cocircuit of $\mathscr{M}/S$. □

2.3 The Blue Rule

Lemma 7 (The Blue Rule, Special Case)**.** *Let $\mathscr{M} = (\mathscr{U}, \mathscr{I})$ be a matroid and $c : \mathscr{U} \to \mathbb{R}$ a cost function. If $D \subseteq \mathscr{U}$ is a cocircuit then every minimum-cost basis of $\mathscr{M}$ must contain a minimum-cost element of D, and every minimum-cost element of D is contained in at least one minimum-cost basis of $\mathscr{M}$.*

Proof. Suppose B is any minimum-cost basis of $\mathscr{M}$ and x is any minimum-cost element of D. If $x \in B$ then we are done. Otherwise, let C be the fundamental circuit of x in B. By Lemma 6 the intersection $C \cap D$ contains at least one element $y \neq x$. By Lemma 3, the set $B' = (B \cup \{x\}) \setminus \{y\}$ is a basis of $\mathscr{M}$. Our assumptions that B is a minimum-cost basis and that $c(x) \leq c(y)$ together imply that $c(x) = c(y)$ and that y is also a minimum-cost element of D. Therefore B contains a minimum-cost element of D (namely, y) and x is contained in a minimum-cost basis of $\mathscr{M}$ (namely, B'), confirming both of the claims in the lemma. □

Theorem 8 (The Blue Rule, General Case)**.** *Let $\mathscr{M} = (\mathscr{U}, \mathscr{I})$ be a matroid and $c : \mathscr{U} \to \mathbb{R}$ a cost function. If $S \subset \mathscr{M}$ is a subset of a minimum-cost basis of $\mathscr{M}$, and D is a cocircuit disjoint from S, then every minimum-cost basis of $\mathscr{M}$ must contain a minimum-cost element of D, and every minimum-cost element of D is contained in at least one minimum-cost basis of $\mathscr{M}$ that is a superset of S.*

Proof. Apply Lemma 7 to the matroid $\mathscr{M}/S$, using the fact that D is a cocircuit in $\mathscr{M}/S$. □

Theorem 8 establishes the correctness of every algorithm that computes a minimum-cost basis by initializing $S = \emptyset$ and repeatedly applying the following rule, known as the *Blue Rule*: while

S is not a basis, let D be a cocircuit disjoint from S, let x be a minimum-cost element of D, and replace S with $S \cup \{x\}$.

In the context of graphs, the blue rule can be stated more succinctly as follows: while S is not a spanning forest of G, find a component of S such that G has at least one edge joining this component to its complement. Find the min-cost such edge and add it into S.

3 Borůvka's Algorithm

Borůvka's algorithm is possibly the earliest published algorithm for the minimum spanning tree problem. It was designed for the purpose of constructing an efficient electricity network for Moravia, in the east of the Czech Republic, in 1926. It is based on applying the blue rule in a series of phases. In the first phase, we find the cheapest edge leaving each vertex, put all of these into the spanning tree, and contract the edges to produce a smaller graph. The subsequent phases recursively apply the same procedure to the smaller graph.

Let G_i denote the contracted graph at the start of phase i. Each vertex of G_i corresponds to one connected component of the forest in G consisting of all the edges that have been chosen so far. The cheapest edge connecting such a component to its complement belongs in the minimum spanning tree, according to the Blue Rule. This proves the correctness of Borůvka's Algorithm. (As might be expected, there are more direct proofs of correctness that avoid the machinery of matroid theory, as in the case of Kruskal's and Prim's algorithms.)

To bound the running time of Borůvka's Algorithm, note that each phase reduces the number of vertices by at least a factor of 2, hence there are at most $\log(n)$ phases. It is easy to implement a phase in time $O(m)$. Given the adjacency list of G_i, we run through the list of edges incident to each vertex, to find the one of minimum cost. Call this set of edges F_i. We now perform a depth-first search on F_i to find the set of connected components. While searching a component of F_i, we store a pointer from each vertex to the root of the component. The set of root vertices of the components of F_i become the vertex set of G_{i+1}. To construct the adjacency list of G_{i+1}, we run through each edge $e = (v, w)$ of G_i; if the root pointers of v and w point to the roots of different components of F_i — say, r_v and r_w — then we add edge e into G_{i+1}, preserving its cost but setting its endpoints to be r_v and r_w.

Thus, the algorithm has $O(\log n)$ phases each running in time $O(m)$, for a total running time of $O(m \log n)$. Andrew Yao discovered a deterministic implementation of Borůvka's Algorithm with running time $O(m \log \log n)$; the main idea is to do some preprocessing, using a linear-time median-finding algorithm, to narrow the set of edges that we must search through when finding the minimum-cost edge adjacent to vertex v.