Introduction to Matroids

Travis Westura

May 1, 2018

1 Introduction

In this course we've seen several examples of algorithms that are "greedy." A greedy algorithm is an algorithm that makes a "locally best" decision. For example, in Dijkstra's algorithm we use a priority queue to keep track of nodes on the friontier and pop them off using the priority of shortest path distance. And in Kruskal's algorithm for finding minimum weight spanning trees, at each step we select the minimum weight edge that does not form a cycle.

We can generalize the idea of "locally make a best decision" by using matroids. The word *matroid* should make you think of the word *matrix*, which you use in linear algebra and multivariable or vector calculus. We'll begin by reviewing the concepts of linear independence, then we'll discuss analogous concepts in graph theory. Then we will relate these concepts by generalizing them and giving the definition of a matroid. And finally we will use matroids to give a proof of the correctness of Kruskal's algorithm.

2 Independence in Linear Algebra

As you take Linear Algebra and Multivariable Calculus you will gain lots of experience working with vectors and matrices. In these notes we'll denote vectors using boldface letters at the end of the alphabet, such as $\vec{\boldsymbol{u}}$ and $\vec{\boldsymbol{v}}$, and matrices using capital letters at the beginning of the alphabet, such as A and B. We'll denote by $\vec{\mathbf{0}}$ the vector of all 0's, and we'll also write vectors as columns and matrices as rectangles containing numbers:

$$\vec{\boldsymbol{u}} = \begin{bmatrix} 3 \\ -1 \\ 3 \end{bmatrix}, \quad \vec{\boldsymbol{v}} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}, \quad \vec{\mathbf{0}} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad A = \begin{bmatrix} 1 & 2 & 5 \\ -2 & 3 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \end{bmatrix}.$$

We say that vectors are elements of a set called a Vector Space, and we also have the ability to add vectors and to multiply them by scalars. In multivariable calculus the most common example of vector spaces are $\mathbb{R}^2$ and $\mathbb{R}^3$, where vectors consist of tuples of 2 and 3 numbers, respectively. We can add vectors and multiply them by scalars as follows:

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} v_2 \\ v_2 \end{bmatrix} = \begin{bmatrix} u_1 + v_1 \\ v_2 + v_2 \end{bmatrix}, \quad a \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} au_1 \\ au_2 \end{bmatrix}.$$

A linear combination of vectors is a sum of scalar multiples of vectors. For example

$$2 \begin{bmatrix} 2 \\ -1 \end{bmatrix} + 3 \begin{bmatrix} -3 \\ 4 \end{bmatrix} = \begin{bmatrix} -5 \\ 10 \end{bmatrix}.$$

We could say that the 3rd vector depends on the first two vectors, since it is a linear combination of them.

A set of vectors[1] $\{\vec{\boldsymbol{v}}_1, \vec{\boldsymbol{v}}_2, \ldots, \vec{\boldsymbol{v}}_n\}$ is *linearly independent* if the only scalars $a_1, a_2, \ldots, a_n$ that satisfy

$$a_1 \vec{\boldsymbol{v}}_1 + a_2 \vec{\boldsymbol{v}}_2 + \cdots + a_n \vec{\boldsymbol{v}}_n = \vec{\mathbf{0}}$$

are $a_1 = a_2 = \cdots = a_n = 0$.This means there is no way to write one of the vectors as a combination of the others, unless we make all the coefficients 0. That is, none of the vectors in the set depend on the others. For example, the standard basis vectors in $\mathbb{R}^3$ are linearly independent,

$$\left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right\},$$

[1] We can also treat the columns of matrices as vectors and define a notion of linear independence on them, although we should be careful that a column may be repeated, whereas a vector in a set is not repeated.

as none of them can be written as a linear combination of the others. The following set of vectors is linearly dependent:

$$\left\{ \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 3 \\ 2 \end{bmatrix}, \begin{bmatrix} 5 \\ 6 \end{bmatrix} \right\}, \quad 2 \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 2+3 \\ 4+2 \end{bmatrix} = \begin{bmatrix} 5 \\ 6 \end{bmatrix}.$$

There is a limit to the number of vectors that we can add to a set while maintaining independence. For example, in $\mathbb{R}^3$, and set of 4 vectors is linearly dependent. An independent set to which we can't add any more vectors is called a *maximal* independent set, with the word maximal meaning we have included as many vectors as possible.[1]

Further, if we have an independent set of vectors, we can remove vectors and still have an independent set. For example, the subset of standard basis vectors,

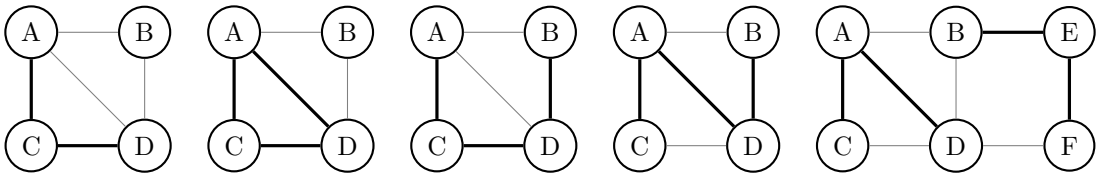
$$\left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \right\},$$

is still independent in $\mathbb{R}^3$.

3 Independence in Graphs

Now let's discuss independence in graphs. Recall that a graph $G = (V, E)$ consists of a set V of vertices and a set E of edges. We will consider only undirected graphs for now.

Recall that a *tree* is a connected graph that does not have any cycles. A *forest* is a graph in which each connected component is a tree. That is, a forest is essentially a collection of trees grouped together. A *spanning tree* of an undirected[2] graph G is a subgraph that is a tree and that includes all of the vertices of G. The "span" part of a spanning tree refers to the tree containing all of the vertices. In the following diagram, from left to right we have: a tree that does not span the graph, a cycle, two spanning trees, and a forest consisting of two trees.



The independent sets of a graph are the forests, that is, the subgraphs that do not include cycles. The dependent sets are the subgraphs that include cycles. To see the analogy to linear algebra, here note that if we have a forest that is not a spanning tree, then there is still an edge we can add to it to obtain a larger subggraph without creating a cycle. A spanning tree gives us a notion of maximal independent set, meaning that we have included as many edges as we possibly can.

And given a forest, removing edges does not create a cycle. Thus we have the same property that removing elements from an independent set preserves independence.

4 Definition of Matroids

Now that we have some intuition about the concept of "independence," let's give a definition of matroids.[3] Many mathematical definitions involve placing a set inside of parentheses with other sets or functions that further describe that set. For example, a graph is given by two sets $G = (V, E)$. A vector space is given by $(V, k, +, \cdot)$, where V is a set of vectors, k is a field of scalars, $+$ is an addition of vectors, and $\cdot$ is a multiplication between a scalar and a vector. A familiar example is $(\mathbb{R}^3, \mathbb{R}, +, \cdot)$. If you have taken CS 2800

[1] A maximal linearly independent set of vectors is called a *basis*.

[2] There is also a notion of spanning tree in an undirected graph, called an *arborescence*.

[3] We could define matroids in many ways. Matroids are referred to as *cryptomorphic*, which means they have many equivalent but ostensibly unrelated definitions.

or other more advanced courses, you will have seen groups, where are often described by writing $(G, +)$, where G is a set and $+$ is a binary operation on that set. Matroids have a similar definition, where we have a set that we write in parentheses together with some other information describing that set. In this definition the additional information is a collection of subsets.

Definition 4.1 (Matroid)**.** A *matroid* M is a pair $(E, \mathcal{I})$ consisting of a finite set E and a collection $\mathcal{I}$ of subsets of E, called the *independent sets*, satisfying the following properties:

1. The collection is nonempty. That is, $\mathcal{I} \neq \varnothing$.
2. All subsets of an independent set are independent. That is, if $A \in \mathcal{I}$ and $B \subseteq A$, then $B \in \mathcal{I}$.
3. If A and B are independent sets and A is larger than B, then we can take an element that is in A but not in B and add it to B to construct an independent set. That is, if $A, B \in \mathcal{I}$ with $|A| > |B|$, then there exists $x \in A \setminus B$ such that $B \cup \{x\} \in \mathcal{I}$. This property is knows as the *independent set exchange property*—we can "exchange" an element from one set to another.

A subset of E that is not independent is called *dependent*.

Let's see how this definition fits in with our first two examples. Recalling that a vector space has a set of vectors V, we form a matroid $(V, \mathcal{I})$, where $\mathcal{I}$ is the collection of independent sets of vectors of V. Recall what we noted previously: given a set of independent vectors, removing vectors still results in an independent set. And if there are two independent sets of vectors $A = \{\vec{\boldsymbol{a}}_1, \ldots, \vec{\boldsymbol{a}}_n\}$ and $B = \{\vec{\boldsymbol{b}}_1, \ldots, \vec{\boldsymbol{b}}_m\}$ with $m < n$, then clearly B is not a maximal independent set, so we can add more vectors to it while maintaining independence.[1]

Next, given an undirected graph $G = (V, E)$, we form a matroid $(E, \mathcal{I})$ with the ground set given by the graph's edges and the independent sets given by the graph's forests. Let's verify each part of the definition.

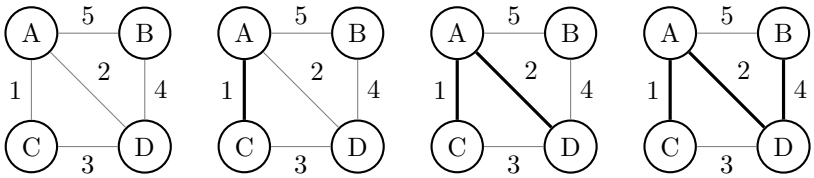
1. The empty collection of edges forms a forest vacuously.
2. If a graph does not form a cycle, then removing edges cannot yield a cycle.
3. Consider two forests A and B where A has more edges than B. Then there is some edge $e \in A$ such that $B \cup \{e\}$ does not have a cycle. The number of connected components in a graph (V, F) is given by $|V| - |F|$. Note that individual vertices form their own component. If A and B are two forests with $|A| > |B|$, then there are more connected components in (V, B) than in (V, A). There there is some edge e in A that connects two of the components in B. Thus $B \cup \{e\}$ forms a forest with one more edge than B.

5 Kruskal's Algorithm and Matroid Optimization

Given a weighted undirected graph, we construct a spanning tree by beginning by starting with the empty set and repeatedly adding the minimum weight edge that does not form a cycle. That is, at each step, add the minimum weight edge that still produces an independent set. The algorithm terminates when we have a spanning tree, or in the language of matroids, when we have a maximal independent set.

In the following example we start with the empty set of edges. The minimum weight edge is between A and C, the we first choose that edge. Next we select the edge between A and D. We then can't select the edge between C and D, as that edge would form a cycle. Thus we select the edge between B and D. At this point there are no other edges we can add without creating a cycle, so we have a spanning tree and terminate out algorithm here.

[1] Being formal about explicitly satisfying the thrid property requires slightly more linear algebra than I want to use right now, so I'll leave this verification as an exercise for those interested.



We are interested in proving the correctness of this abstract algorithm. For now we don't consider the implementation details.[1] Kruskal's spanning tree algorithm is actually a case of the more general greedy algorithm that is used for weighted matroids. A *weighted matroid* is a matroid in which every element of the ground set has a nonnegative weight. That is, a weighted matroid $(E, \mathcal{I})$ has a weight $w_e \geq 0$ for each edge $e \in E$. Finding a minimum weight spanning tree then becomes the problem of finding a minmum weight maximal independent set.[2]

In this general case the greedy algorithm works the same way. Begin with the empty set. Repeated select the minimum weight element of E that maintains independence. Terminate when the only remaining elements of E would form a dependent set if added.

Theorem 5.1. *Let $(E, \mathcal{I})$ be a matroid with weights $w_e \geq 0$ for each $e \in E$. Then the greedy algorithm produces a maximal independent set of minimum weight.*

Proof. Suppose that the greedy algorithm selects elements $[e_1, e_2, \ldots, e_r]$, in that order. These elements form an independent set, and their weights are in ascending order: $w_{e_1} \leq w_{e_2} \leq \cdots \leq w_{e_r}$. To see this set is of minimum weight, suppose there is another maximal independent set consisting of $[x_1, \ldots, x_r]$ of lower weight. Then for some i we must have $w_{x_i} \leq w_{e_i}$. Use the least such i, which we note must be greater than 1, since the greedy algorithm picks the min cost edge at the first step. Now consider $B = \{e_1, \ldots, e_{i-1}\}$ and $A = \{x_1, \ldots, x_i\}$. Since $|A| > |B|$ and both sets are independent, we use the exchange property. There is some j such that $x_j \in A \setminus B$ and $B \cup \{x_j\}$ is independent. But since $w_{x_j} \leq w_{x_i} < w_{e_i}$, we have a contradiction, as the greedy algorithm would have picked x_j before e_i. QED

There is also a converse to the previous result: if an independence system $(E, \mathcal{I})$ is not a matroid, then the greedy algorithm fails for some choice of integer weights. You'll learn more about Greedy Algorithms if you take the undergraduate algorithms course CS 4820, and you'll learn more about matroids if you take the graduate algorithms course CS 6820 or the combinatorics course sequence Math 4410 and 4420.

[1] For the details of an efficient implementation, look up the Union-Find data structure.

[2] To use more terminalogy from linear algebra, a maximal independent set is called a *base*. All maximal independent sets contain the same number of elements, and this number is called the *rank* of the matroid.