Matroids for solving Optimisation Problems

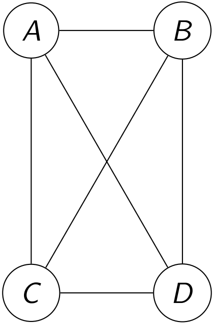
Greedy algorithm as a solution

Owen McDonnell

NUI Galway

2018

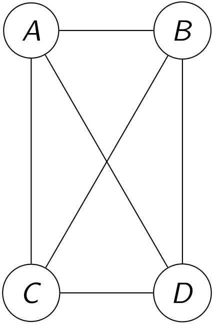
The Problem



The Setup

- G is a connected graph
- Each vertex is a town.
- Cost is assigned to each edge
- Corresponds to providing a rail link between the towns.

The Problem



The Setup

- G is a connected graph
- Each vertex is a town.
- Cost is assigned to each edge
- Corresponds to providing a rail link between the towns.

The Statement

- Finding the minimum cost of a railway linking all n towns.
- Corresponds to finding a minimal weighted spanning tree of G.

A Greedy Algorithm (Kruskal)

Kruskal's algorithm is a greedy algorithm that finds a minimum spanning tree for a connected weighted Graph.

Algorithm

1) Create a graph F containing just the vertices of G.

A Greedy Algorithm (Kruskal)

Kruskal's algorithm is a greedy algorithm that finds a minimum spanning tree for a connected weighted Graph.

Algorithm

1) Create a graph F containing just the vertices of G.
2) Create a set $S = E(G)$; the edge set of G.

A Greedy Algorithm (Kruskal)

Kruskal's algorithm is a greedy algorithm that finds a minimum spanning tree for a connected weighted Graph.

Algorithm

1) Create a graph F containing just the vertices of G.
2) Create a set $S = E(G)$; the edge set of G.
3) While S is non-empty and F is not yet spanning

A Greedy Algorithm (Kruskal)

Kruskal's algorithm is a greedy algorithm that finds a minimum spanning tree for a connected weighted Graph.

Algorithm

1) Create a graph F containing just the vertices of G.
2) Create a set $S = E(G)$; the edge set of G.
3) While S is non-empty and F is not yet spanning
    - 3(a) Remove an edge with minimum weight from S.
    - 3(b) If the removed edge introduces no cycles to F then add the edge to F

Why Greedy works?

Let B_G be a spanning tree created through the greedy algorithm.

Lemma

If $(E, \mathcal{I})$ is a matroid M, then B_G is a solution to the optimization problem.

Why Greedy works?

Let B_G be a spanning tree created through the greedy algorithm.

Lemma

If $(E, \mathcal{I})$ is a matroid M, then B_G is a solution to the optimization problem.

Question

But what is a matroid?

Independence Systems and Matroids

Definition

An *independence system* is a pair $(E, \mathcal{S})$, where E is a set and $\mathcal{S}$ is a collection of sets satisfying:

Independence Systems and Matroids

Definition

An *independence system* is a pair $(E, \mathcal{S})$, where E is a set and $\mathcal{S}$ is a collection of sets satisfying:

(I1) $\mathcal{S}$ is non-empty.
(I2) $\mathcal{S}$ is a hereditary subset of the power set of E.
The elements of $\mathcal{S}$ are called the *independent sets*.

Independence Systems and Matroids

Definition

An *independence system* is a pair $(E, \mathcal{S})$, where E is a set and $\mathcal{S}$ is a collection of sets satisfying:

(I1) $\mathcal{S}$ is non-empty.
(I2) $\mathcal{S}$ is a hereditary subset of the power set of E.
The elements of $\mathcal{S}$ are called the *independent sets*.

Definition

A matroid is a pair $(E, \mathcal{I})$ with finite ground set E and $\mathcal{I}$ being a collection of independent subsets of E satisfying the above conditions (I1), (I2) with the following extra condition:

Independence Systems and Matroids

Definition

An *independence system* is a pair $(E, \mathcal{S})$, where E is a set and $\mathcal{S}$ is a collection of sets satisfying:

(I1) $\mathcal{S}$ is non-empty.
(I2) $\mathcal{S}$ is a hereditary subset of the power set of E.
The elements of $\mathcal{S}$ are called the *independent sets*.

Definition

A matroid is a pair $(E, \mathcal{I})$ with finite ground set E and $\mathcal{I}$ being a collection of independent subsets of E satisfying the above conditions (I1), (I2) with the following extra condition:

(I3): If A and B are two independent sets in $\mathcal{I}$ and $|A| = |B| + 1$, then there exists $x \in A \setminus B$ such that $B \cup \{x\}$ is in $\mathcal{I}$

Bases of a Matroid

Definition

A base is a maximal independent subset of $\mathcal{I}$.
All the maximally independent sets have the same cardinality, this is the *rank* of the matroid.

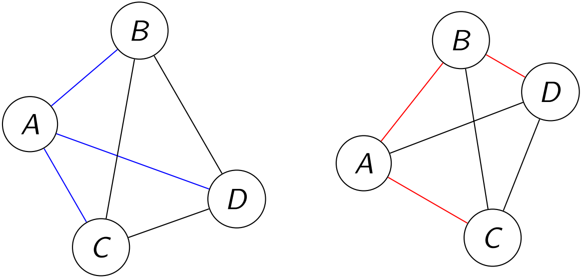
Definition

Let $\mathcal{B}$ be a set of subsets of a finite set E. Then $\mathcal{B}$ is the collection of bases of a matroid on E if and only if $\mathcal{B}$ satisfies the following conditions:
(B1) $\mathcal{B}$ is non-empty.
(B2) If B_1 and B_2 are members of $\mathcal{B}$ and $x \in B_1 \setminus B_2$, then there is an element y of $B_2 \setminus B_1$ such that $(B_1 \setminus \{x\}) \cup \{y\} \in \mathcal{B}$.

Spanning trees are Bases

Definition

A spanning tree T of an undirected graph G is a subgraph that is a *tree* which includes all of the vertices of G.



We see that $\mathcal{B}$ (the collection of maximal elements of $\mathcal{I}$) corresponds to the set of spanning trees of the graph.

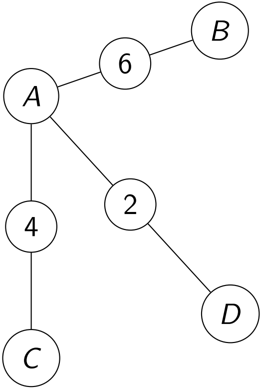
Weight Function

The optimization problem associated with $(E, \mathcal{S})$ is the following: for a given weight function $\omega : E \to \mathbb{R}^+$, we want to find an independent set A whose weight is maximal, where

$$\omega(A) := \sum_{e \in A} \omega(e) \tag{1}$$

Demonstration of Kruskal's algorithm

```
owen@owen-ThinkPad-X201 ~/Matroids $ perl
$VAR1 = {
          'AB' => 6,
          'AC' => 4,
          'AD' => 2,
          'BD' => 8,
          'BC' => 6,
          'CD' => 9
        };
$VAR1 = {
          'A' => [
                   'AD',
                   'AC',
                   'AB'
                 ],
          'B' => [
                   'AB'
                 ],
          'C' => [
                   'AC'
                 ],
          'D' => [
                   'AD'
                 ]
        };
This Graph is Acyclic
owen@owen-ThinkPad-X201 ~/Matroids $ 
```



Why Greedy works: The solution

Let B_G be a base of a matroid generated by the greedy algorithm.

Lemma

If $(E, \mathcal{I})$ is a matroid M, then B_G is a solution to the optimization problem.

Why Greedy works: The solution

Let B_G be a base of a matroid generated by the greedy algorithm.

Lemma

If $(E, \mathcal{I})$ is a matroid M, then B_G is a solution to the optimization problem.

Proof.

If $r(M) = r$, then $B_G = \{e_1, e_2, ..., e_r\}$ is a basis of M.

Why Greedy works: The solution

Let B_G be a base of a matroid generated by the greedy algorithm.

Lemma

If $(E, \mathcal{I})$ is a matroid M, then B_G is a solution to the optimization problem.

Proof.

If $r(M) = r$, then $B_G = \{e_1, e_2, ..., e_r\}$ is a basis of M. Let B be another basis of M, $B = \{f_1, f_2, ..., f_r\}$

Why Greedy works: The solution

Let B_G be a base of a matroid generated by the greedy algorithm.

Lemma

If $(E, \mathcal{I})$ is a matroid M, then B_G is a solution to the optimization problem.

Proof.

If $r(M) = r$, then $B_G = \{e_1, e_2, ..., e_r\}$ is a basis of M. Let B be another basis of M, $B = \{f_1, f_2, ..., f_r\}$ where $\omega(f_1) \geq \omega(f_2) \geq ... \geq \omega(f_r)$.

Why Greedy works: The solution

Let B_G be a base of a matroid generated by the greedy algorithm.

Lemma

If $(E,\mathcal{I})$ is a matroid M, then B_G is a solution to the optimization problem.

Proof.

If $r(M) = r$, then $B_G = \{e_1, e_2, ..., e_r\}$ is a basis of M. Let B be another basis of M, $B = \{f_1, f_2, ..., f_r\}$ where $\omega(f_1) \geq \omega(f_2) \geq ... \geq \omega(f_r)$.We claim that $\omega(e_j) \geq \omega(f_f)\ \forall j$, then it follows that $\omega(B_G) \geq \omega(B)$ for any other basis in $\mathcal{B}$. $\square$

Continued

Lemma

If $1 \leq j \leq r$, *then* $\omega(e_j) \geq \omega(f_j)$.

Proof.

Suppose (seeking a contradiction) that k is the least integer for which $\omega(e_k) < \omega(f_k)$.

Continued

Lemma

If $1 \leq j \leq r$, *then* $\omega(e_j) \geq \omega(f_j)$.

Proof.

Suppose (seeking a contradiction) that k is the least integer for which $\omega(e_k) < \omega(f_k)$.Take $I_1 = \{e_1, e_2, ..., e_{k-1}\}$ and $I_2 = \{f_1, f_2, ..., f_k\}$.

Continued

Lemma

If $1 \leq j \leq r$, *then* $\omega(e_j) \geq \omega(f_j)$.

Proof.

Suppose (seeking a contradiction) that k is the least integer for which $\omega(e_k) < \omega(f_k)$.Take $I_1 = \{e_1, e_2, ..., e_{k-1}\}$ and $I_2 = \{f_1, f_2, ..., f_k\}$.Since $|I_2| = |I_1| + 1$ $(I3)$ implies $I_1 \cup \{f_t\} \in \mathcal{I}$ for some $f_t \in I_2 \setminus I_1$.

Continued

Lemma

If $1 \leq j \leq r$, then $\omega(e_j) \geq \omega(f_j)$.

Proof.

Suppose (seeking a contradiction) that k is the least integer for which $\omega(e_k) < \omega(f_k)$.Take $I_1 = \{e_1, e_2, ..., e_{k-1}\}$ and $I_2 = \{f_1, f_2, ..., f_k\}$.Since $|I_2| = |I_1| + 1$ $(I3)$ implies $I_1 \cup \{f_t\} \in \mathcal{I}$ for some $f_t \in I_2 \setminus I_1$. But this means that $\omega(f_t) \geq \omega(f_k) > \omega(e_k)$

Continued

Lemma

If $1 \leq j \leq r$, *then* $\omega(e_j) \geq \omega(f_j)$.

Proof.

Suppose (seeking a contradiction) that k is the least integer for which $\omega(e_k) < \omega(f_k)$.Take $I_1 = \{e_1, e_2, ..., e_{k-1}\}$ and $I_2 = \{f_1, f_2, ..., f_k\}$.Since $|I_2| = |I_1| + 1$ $(I3)$ implies $I_1 \cup \{f_t\} \in \mathcal{I}$ for some $f_t \in I_2 \setminus I_1$. But this means that $\omega(f_t) \geq \omega(f_k) > \omega(e_k)$ And hence the Greedy algorithm would have chosen f_t over e_k, which gives us our contradiction. $\square$

Wrap Up

- The previous proof shows that the greedy algorithm finds a maximal member of B of $\mathcal{I}$ of maximum weight
- This allows us to deduce that Kruskals algorithm is correct.

Wrap Up

- The previous proof shows that the greedy algorithm finds a maximal member of B of $\mathcal{I}$ of maximum weight
- This allows us to deduce that Kruskals algorithm is correct.
- And that the greedy algorithm gives us a solution to our optimisation problem as long as we have a matroid

Wrap Up

- The previous proof shows that the greedy algorithm finds a maximal member of B of $\mathcal{I}$ of maximum weight
- This allows us to deduce that Kruskals algorithm is correct.
- And that the greedy algorithm gives us a solution to our optimisation problem as long as we have a matroid
- Furthermore, the greedy does not provide a solution if the data does not form a matroid.