# Lecture 19: Matroids and the Greedy Algorithm

Michael Dinitz

November 4, 2025
601.433/633 Introduction to Algorithms

# Introduction

Last time: somewhat greedy algorithm (Prim's), extremely greedy algorithm (Kruskal's)

Question: when does greedy algorithm return optimal solution?

# Introduction

Last time: somewhat greedy algorithm (Prim's), extremely greedy algorithm (Kruskal's)

Question: when does greedy algorithm return optimal solution?

Want abstraction that includes MSTs, but also works for many other problems.

# Introduction

Last time: somewhat greedy algorithm (Prim's), extremely greedy algorithm (Kruskal's)

Question: when does greedy algorithm return optimal solution?

Want abstraction that includes MSTs, but also works for many other problems.

*Weighted Set System:*

- Universe $U$
- Collection $\mathcal{I} \subseteq 2^U$ (so $I \subseteq U$ for all $I \in \mathcal{I}$). Called *independent sets*
- Weights $w : U \to \mathbb{R}^+$

# Introduction

Last time: somewhat greedy algorithm (Prim's), extremely greedy algorithm (Kruskal's)

Question: when does greedy algorithm return optimal solution?

Want abstraction that includes MSTs, but also works for many other problems.

*Weighted Set System:*

- Universe $\boldsymbol{U}$
- Collection $\mathcal{I} \subseteq \mathbf{2}^{\boldsymbol{U}}$ (so $\boldsymbol{I} \subseteq \boldsymbol{U}$ for all $\boldsymbol{I} \in \mathcal{I}$). Called *independent sets*
- Weights $\boldsymbol{w} : \boldsymbol{U} \to \mathbb{R}^+$

Problem: find *max weight* independent set

# MST as Weighted Set System

MST: weighted graph $G = (V, E, w)$. Find MST.

Set system:

- $U = E$
- $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

# MST as Weighted Set System

MST: weighted graph $G = (V, E, w)$. Find MST.

Set system:

- $U = E$
- $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

What about weights? MST is minimize, but problem we defined is maximize.

# MST as Weighted Set System

MST: weighted graph $G = (V, E, w)$. Find MST.

Set system:

- $U = E$
- $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

What about weights? MST is minimize, but problem we defined is maximize.

- Let $\bar{w} > w(e)$ for all $e \in E$, let $w'(e) = \bar{w} - w(e)$ for all $e \in E$

# MST as Weighted Set System

MST: weighted graph $G = (V, E, w)$. Find MST.

Set system:

- $U = E$
- $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

What about weights? MST is minimize, but problem we defined is maximize.

- Let $\bar{w} > w(e)$ for all $e \in E$, let $w'(e) = \bar{w} - w(e)$ for all $e \in E$

For any tree $T$:

$$w'(T) = \sum_{e \in T} w'(e) = \sum_{e \in T} (\bar{w} - w(e)) = \sum_{e \in T} \bar{w} - \sum_{e \in T} w(e) = (n-1)\bar{w} - w(T)$$

# MST as Weighted Set System

MST: weighted graph $G = (V, E, w)$. Find MST.

Set system:

- $U = E$
- $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

What about weights? MST is minimize, but problem we defined is maximize.

- Let $\bar{w} > w(e)$ for all $e \in E$, let $w'(e) = \bar{w} - w(e)$ for all $e \in E$

For any tree $T$:

$$w'(T) = \sum_{e \in T} w'(e) = \sum_{e \in T} (\bar{w} - w(e)) = \sum_{e \in T} \bar{w} - \sum_{e \in T} w(e) = (n-1)\bar{w} - w(T)$$

So under weights $w'$, max-weight IS $=$ max-weight forest $=$ max-weight spanning tree $=$ min-weight spanning tree (weights $w$)

- So finding max-weight forest $=$ finding min spanning tree.

# Useful Properties of Forests

Let $\boldsymbol{U} = \boldsymbol{E}$ and $\mathcal{I} = \{\boldsymbol{F} \subseteq \boldsymbol{E} : (\boldsymbol{V}, \boldsymbol{F}) \text{ a forest}\}$

Useful properties:

1. $\varnothing \in \mathcal{I}$
2. If $\boldsymbol{F} \in \mathcal{I}$ and $\boldsymbol{F}' \subseteq \boldsymbol{F}$, then $\boldsymbol{F}' \in \mathcal{I}$

# Useful Properties of Forests

Let $\boldsymbol{U} = \boldsymbol{E}$ and $\mathcal{I} = \{\boldsymbol{F} \subseteq \boldsymbol{E} : (\boldsymbol{V}, \boldsymbol{F}) \text{ a forest}\}$

Useful properties:

1. $\emptyset \in \mathcal{I}$
2. If $\boldsymbol{F} \in \mathcal{I}$ and $\boldsymbol{F}' \subseteq \boldsymbol{F}$, then $\boldsymbol{F}' \in \mathcal{I}$
3. *Augmentation Property*: If $\boldsymbol{F_1} \in \mathcal{I}$ and $\boldsymbol{F_2} \in \mathcal{I}$ with $|\boldsymbol{F_2}| > |\boldsymbol{F_1}|$, then there is some edge $\boldsymbol{e} \in \boldsymbol{F_2} \setminus \boldsymbol{F_1}$ such that $\boldsymbol{F_1} \cup \{\boldsymbol{e}\} \in \mathcal{I}$.

# Useful Properties of Forests

Let $U = E$ and $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

Useful properties:

1. $\emptyset \in \mathcal{I}$
2. If $F \in \mathcal{I}$ and $F' \subseteq F$, then $F' \in \mathcal{I}$
3. *Augmentation Property*: If $F_1 \in \mathcal{I}$ and $F_2 \in \mathcal{I}$ with $|F_2| > |F_1|$, then there is some edge $e \in F_2 \setminus F_1$ such that $F_1 \cup \{e\} \in \mathcal{I}$.

**Proof Sketch that Forests have Augmentation Property.**

Suppose false: no edge in $F_2 \setminus F_1$ can be added to $F_1$. Let $c_1$ = # components in $F_1$, $c_2$ = # components in $F_2$

# Useful Properties of Forests

Let $U = E$ and $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

Useful properties:

1. $\emptyset \in \mathcal{I}$
2. If $F \in \mathcal{I}$ and $F' \subseteq F$, then $F' \in \mathcal{I}$
3. *Augmentation Property*: If $F_1 \in \mathcal{I}$ and $F_2 \in \mathcal{I}$ with $|F_2| > |F_1|$, then there is some edge $e \in F_2 \setminus F_1$ such that $F_1 \cup \{e\} \in \mathcal{I}$.

## Proof Sketch that Forests have Augmentation Property.

Suppose false: no edge in $F_2 \setminus F_1$ can be added to $F_1$. Let $c_1$ = # components in $F_1$, $c_2$ = # components in $F_2$

$\implies$ every edge of $F_2$ has both endpoints in same component of $F_1$

# Useful Properties of Forests

Let $U = E$ and $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

Useful properties:

1. $\emptyset \in \mathcal{I}$
2. If $F \in \mathcal{I}$ and $F' \subseteq F$, then $F' \in \mathcal{I}$
3. *Augmentation Property*: If $F_1 \in \mathcal{I}$ and $F_2 \in \mathcal{I}$ with $|F_2| > |F_1|$, then there is some edge $e \in F_2 \setminus F_1$ such that $F_1 \cup \{e\} \in \mathcal{I}$.

## Proof Sketch that Forests have Augmentation Property.

Suppose false: no edge in $F_2 \setminus F_1$ can be added to $F_1$. Let $c_1$ = # components in $F_1$, $c_2$ = # components in $F_2$

$\implies$ every edge of $F_2$ has both endpoints in same component of $F_1$

$\implies$ every component of $F_2$ contained in component of $F_1$ $\implies$ $c_2 \geq c_1$

# Useful Properties of Forests

Let $U = E$ and $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

Useful properties:

1. $\emptyset \in \mathcal{I}$
2. If $F \in \mathcal{I}$ and $F' \subseteq F$, then $F' \in \mathcal{I}$
3. *Augmentation Property*: If $F_1 \in \mathcal{I}$ and $F_2 \in \mathcal{I}$ with $|F_2| > |F_1|$, then there is some edge $e \in F_2 \setminus F_1$ such that $F_1 \cup \{e\} \in \mathcal{I}$.

## Proof Sketch that Forests have Augmentation Property.

Suppose false: no edge in $F_2 \setminus F_1$ can be added to $F_1$. Let $c_1$ = # components in $F_1$, $c_2$ = # components in $F_2$

$\implies$ every edge of $F_2$ has both endpoints in same component of $F_1$

$\implies$ every component of $F_2$ contained in component of $F_1$ $\implies$ $c_2 \geq c_1$

But $c_2 = n - |F_2| < n - |F_1| = c_1$.

# Useful Properties of Forests

Let $U = E$ and $\mathcal{I} = \{F \subseteq E : (V, F) \text{ a forest}\}$

Useful properties:

1. $\emptyset \in \mathcal{I}$
2. If $F \in \mathcal{I}$ and $F' \subseteq F$, then $F' \in \mathcal{I}$
3. *Augmentation Property*: If $F_1 \in \mathcal{I}$ and $F_2 \in \mathcal{I}$ with $|F_2| > |F_1|$, then there is some edge $e \in F_2 \setminus F_1$ such that $F_1 \cup \{e\} \in \mathcal{I}$.

**Proof Sketch that Forests have Augmentation Property.**

Suppose false: no edge in $F_2 \setminus F_1$ can be added to $F_1$. Let $c_1 = \#$ components in $F_1$, $c_2 = \#$ components in $F_2$

$\implies$ every edge of $F_2$ has both endpoints in same component of $F_1$

$\implies$ every component of $F_2$ contained in component of $F_1$ $\implies$ $c_2 \geq c_1$

But $c_2 = n - |F_2| < n - |F_1| = c_1$.

Contradiction. □

# Matroids

## Definition

$(U, \mathcal{I})$ is a *matroid* if the following three properties hold:

1. $\varnothing \in \mathcal{I}$,
2. If $F \in \mathcal{I}$ and $F' \subseteq F$, then $F' \in \mathcal{I}$, and
3. If $F_1 \in \mathcal{I}$ and $F_2 \in \mathcal{I}$ with $|F_2| > |F_1|$, then there is some element $e \in F_2 \setminus F_1$ such that $F_1 \cup \{e\} \in \mathcal{I}$.

# Matroids

## Definition

$(U, \mathcal{I})$ is a *matroid* if the following three properties hold:

1. $\varnothing \in \mathcal{I}$,
2. If $F \in \mathcal{I}$ and $F' \subseteq F$, then $F' \in \mathcal{I}$, and
3. If $F_1 \in \mathcal{I}$ and $F_2 \in \mathcal{I}$ with $|F_2| > |F_1|$, then there is some element $e \in F_2 \setminus F_1$ such that $F_1 \cup \{e\} \in \mathcal{I}$.

$(U, \mathcal{I})$ is a *hereditary set system* if the first two properties hold.

# Matroids

## Definition

$(U, \mathcal{I})$ is a *matroid* if the following three properties hold:

1. $\emptyset \in \mathcal{I}$,
2. If $F \in \mathcal{I}$ and $F' \subseteq F$, then $F' \in \mathcal{I}$, and
3. If $F_1 \in \mathcal{I}$ and $F_2 \in \mathcal{I}$ with $|F_2| > |F_1|$, then there is some element $e \in F_2 \setminus F_1$ such that $F_1 \cup \{e\} \in \mathcal{I}$.

$(U, \mathcal{I})$ is a *hereditary set system* if the first two properties hold.

Matroid theory: super interesting area of combinatorics! Surprising amount of structure.

# Matroids

**Definition**

$(U, \mathcal{I})$ is a *matroid* if the following three properties hold:

1. $\emptyset \in \mathcal{I}$,
2. If $F \in \mathcal{I}$ and $F' \subseteq F$, then $F' \in \mathcal{I}$, and
3. If $F_1 \in \mathcal{I}$ and $F_2 \in \mathcal{I}$ with $|F_2| > |F_1|$, then there is some element $e \in F_2 \setminus F_1$ such that $F_1 \cup \{e\} \in \mathcal{I}$.

$(U, \mathcal{I})$ is a *hereditary set system* if the first two properties hold.

Matroid theory: super interesting area of combinatorics! Surprising amount of structure.

Warmup: In any matroid, the maximal independent sets (called bases) have the same size (called the rank of the matroid).

# Examples of Matroids

- Forests in graphs

# Examples of Matroids

- Forests in graphs
- Linearly independent vectors in vector space
  - $U$ a finite set of vectors in $\mathbb{R}^d$
  - $\mathcal{I} = \{F \subseteq U : F \text{ linearly independent}\}$

# Examples of Matroids

- Forests in graphs
- Linearly independent vectors in vector space
  - $U$ a finite set of vectors in $\mathbb{R}^d$
  - $\mathcal{I} = \{F \subseteq U : F \text{ linearly independent}\}$
  - $\varnothing$ linearly independent
  - If $F$ linearly independent and $F' \subseteq F$, then $F'$ linearly independent
  - Augmentation: if $F_1$ linearly independent, $F_2$ linearly independent, and $|F_2| > |F_1| \implies$ $\dim(\text{span}(F_1)) = |F_1| < |F_2| = \dim(\text{span}(F_2))$

# Examples of Matroids

- Forests in graphs
- Linearly independent vectors in vector space
  - $U$ a finite set of vectors in $\mathbb{R}^d$
  - $\mathcal{I} = \{F \subseteq U : F \text{ linearly independent}\}$
  - $\varnothing$ linearly independent
  - If $F$ linearly independent and $F' \subseteq F$, then $F'$ linearly independent
  - Augmentation: if $F_1$ linearly independent, $F_2$ linearly independent, and $|F_2| > |F_1| \implies \dim(\text{span}(F_1)) = |F_1| < |F_2| = \dim(\text{span}(F_2))$

Matroids: generalize both graph theory and linear algebra!

- Originally invented by Whitney as an attempt to generalize the concept of "linear independence"

# Representation

To do algorithms with matroids, need to figure out how they're represented.

# Representation

To do algorithms with matroids, need to figure out how they're represented.

Option 1: list all independent sets

# Representation

To do algorithms with matroids, need to figure out how they're represented.

Option 1: list all independent sets

- Too many of them!

# Representation

To do algorithms with matroids, need to figure out how they're represented.

Option 1: list all independent sets

- Too many of them!

What did we need for MST (Kruskal)?

# Representation

To do algorithms with matroids, need to figure out how they're represented.

Option 1: list all independent sets

- Too many of them!

What did we need for MST (Kruskal)?

**Independence Oracle**: algorithm which take $\boldsymbol{F} \subseteq \boldsymbol{U}$, returns YES if $\boldsymbol{F} \in \mathcal{I}$, NO if $\boldsymbol{F} \notin \mathcal{I}$

# Representation

To do algorithms with matroids, need to figure out how they're represented.

Option 1: list all independent sets

- Too many of them!

What did we need for MST (Kruskal)?

**Independence Oracle**: algorithm which take $F \subseteq U$, returns YES if $F \in \mathcal{I}$, NO if $F \notin \mathcal{I}$

For MST: "does $F$ have any cycles"? Independence oracle: DFS/BFS, union-find

# Representation

To do algorithms with matroids, need to figure out how they're represented.

Option 1: list all independent sets

- Too many of them!

What did we need for MST (Kruskal)?

**Independence Oracle**: algorithm which take $F \subseteq U$, returns YES if $F \in \mathcal{I}$, NO if $F \notin \mathcal{I}$

For MST: "does $F$ have any cycles"? Independence oracle: DFS/BFS, union-find

We'll assume we have independence oracle.

# Greedy Algorithm

Kruskal, generalized to matroids (and max weight)!

# Greedy Algorithm

Kruskal, generalized to matroids (and max weight)!

$\boldsymbol{F} = \emptyset$
Sort $\boldsymbol{U}$ by weight (largest to smallest)
For each $\boldsymbol{u} \in \boldsymbol{U}$ in sorted order $\{$
    If $\boldsymbol{F} \cup \{\boldsymbol{u}\} \in \mathcal{I}$, add $\boldsymbol{u}$ to $\boldsymbol{F}$
$\}$
Return $\boldsymbol{F}$

# Correctness

**Theorem**

*Let $F$ be independent set returned by greedy. Then $w(F) \geq w(F')$ for all $F' \in \mathcal{I}$.*

# Correctness

**Theorem**

*Let $F$ be independent set returned by greedy. Then $w(F) \geq w(F')$ for all $F' \in \mathcal{I}$.*

- $F = \{f_1, f_2, \ldots, f_r\}$, where $w(f_i) \geq w(f_{i+1})$ for all $i$ (order added by greedy)
- $F' = \{e_1, e_2, \ldots, e_r\}$ where $w(e_i) \geq w(e_{i+1})$ for all $i$

# Correctness

**Theorem**

*Let $F$ be independent set returned by greedy. Then $w(F) \geq w(F')$ for all $F' \in \mathcal{I}$.*

- $F = \{f_1, f_2, \ldots, f_r\}$, where $w(f_i) \geq w(f_{i+1})$ for all $i$ (order added by greedy)
- $F' = \{e_1, e_2, \ldots, e_r\}$ where $w(e_i) \geq w(e_{i+1})$ for all $i$

**Claim:** $w(f_i) \geq w(e_i)$ for all $i$.

# Correctness

**Theorem**

*Let $F$ be independent set returned by greedy. Then $w(F) \geq w(F')$ for all $F' \in \mathcal{I}$.*

- $F = \{f_1, f_2, \ldots, f_r\}$, where $w(f_i) \geq w(f_{i+1})$ for all $i$ (order added by greedy)
- $F' = \{e_1, e_2, \ldots, e_r\}$ where $w(e_i) \geq w(e_{i+1})$ for all $i$

**Claim:** $w(f_i) \geq w(e_i)$ for all $i$.

**Proof:** Suppose false, let $j$ smallest integer such that $w(f_j) < w(e_j)$.

# Correctness

**Theorem**

*Let $F$ be independent set returned by greedy. Then $w(F) \geq w(F')$ for all $F' \in \mathcal{I}$.*

- $F = \{f_1, f_2, \ldots, f_r\}$, where $w(f_i) \geq w(f_{i+1})$ for all $i$ (order added by greedy)
- $F' = \{e_1, e_2, \ldots, e_r\}$ where $w(e_i) \geq w(e_{i+1})$ for all $i$

**Claim:** $w(f_i) \geq w(e_i)$ for all $i$.

**Proof:** Suppose false, let $j$ smallest integer such that $w(f_j) < w(e_j)$.

Let $F_1 = \{f_1, \ldots, f_{j-1}\}$ and let $F_2 = \{e_1, \ldots, e_j\}$

# Correctness

**Theorem**

*Let $F$ be independent set returned by greedy. Then $w(F) \geq w(F')$ for all $F' \in \mathcal{I}$.*

- $F = \{f_1, f_2, \ldots, f_r\}$, where $w(f_i) \geq w(f_{i+1})$ for all $i$ (order added by greedy)
- $F' = \{e_1, e_2, \ldots, e_r\}$ where $w(e_i) \geq w(e_{i+1})$ for all $i$

**Claim:** $w(f_i) \geq w(e_i)$ for all $i$.

**Proof:** Suppose false, let $j$ smallest integer such that $w(f_j) < w(e_j)$.

Let $F_1 = \{f_1, \ldots, f_{j-1}\}$ and let $F_2 = \{e_1, \ldots, e_j\}$

$|F_2| > |F_1|$, so by augmentation there is some $e_z \in F_2 \setminus F_1$ such that $F_1 \cup \{e_z\} \in \mathcal{I}$.

# Correctness

**Theorem**

*Let $F$ be independent set returned by greedy. Then $w(F) \geq w(F')$ for all $F' \in \mathcal{I}$.*

- $F = \{f_1, f_2, \ldots, f_r\}$, where $w(f_i) \geq w(f_{i+1})$ for all $i$ (order added by greedy)
- $F' = \{e_1, e_2, \ldots, e_r\}$ where $w(e_i) \geq w(e_{i+1})$ for all $i$

**Claim:** $w(f_i) \geq w(e_i)$ for all $i$.

**Proof:** Suppose false, let $j$ smallest integer such that $w(f_j) < w(e_j)$.

Let $F_1 = \{f_1, \ldots, f_{j-1}\}$ and let $F_2 = \{e_1, \ldots, e_j\}$

$|F_2| > |F_1|$, so by augmentation there is some $e_z \in F_2 \setminus F_1$ such that $F_1 \cup \{e_z\} \in \mathcal{I}$.

$$w(e_z) \geq w(e_j) > w(f_j)$$

# Correctness

**Theorem**

*Let $F$ be independent set returned by greedy. Then $w(F) \geq w(F')$ for all $F' \in \mathcal{I}$.*

- $F = \{f_1, f_2, \ldots, f_r\}$, where $w(f_i) \geq w(f_{i+1})$ for all $i$ (order added by greedy)
- $F' = \{e_1, e_2, \ldots, e_r\}$ where $w(e_i) \geq w(e_{i+1})$ for all $i$

**Claim:** $w(f_i) \geq w(e_i)$ for all $i$.

**Proof:** Suppose false, let $j$ smallest integer such that $w(f_j) < w(e_j)$.

Let $F_1 = \{f_1, \ldots, f_{j-1}\}$ and let $F_2 = \{e_1, \ldots, e_j\}$

$|F_2| > |F_1|$, so by augmentation there is some $e_z \in F_2 \setminus F_1$ such that $F_1 \cup \{e_z\} \in \mathcal{I}$.

$$w(e_z) \geq w(e_j) > w(f_j)$$

Contradiction! Greedy would add $e_z$ next, not $f_j$.

# Converse

So greedy works on matroids. Amazing fact: if greedy works, set system is a matroid!

# Converse

So greedy works on matroids. Amazing fact: if greedy works, set system is a matroid!

**Theorem**

*Let $(U, \mathcal{I})$ be an hereditary set system. If for every weighting $w : U \to \mathbb{R}_{\geq 0}$ the greedy algorithm returns a maximum weight independent set, then $(U, \mathcal{I})$ is a matroid.*

# Converse

So greedy works on matroids. Amazing fact: if greedy works, set system is a matroid!

**Theorem**

*Let $(U, \mathcal{I})$ be an hereditary set system. If for every weighting $w : U \to \mathbb{R}_{\geq 0}$ the greedy algorithm returns a maximum weight independent set, then $(U, \mathcal{I})$ is a matroid.*

So for hereditary set systems, matroids exactly characterize when the greedy algorithm works!

# Proof

Contradiction. Suppose false $\implies (\boldsymbol{U}, \boldsymbol{\mathcal{I}})$ hereditary but not matroid.

# Proof

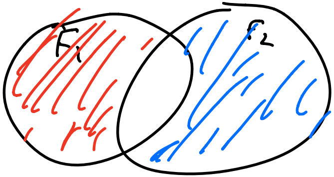
Contradiction. Suppose false $\implies$ $(\boldsymbol{U}, \mathcal{I})$ hereditary but not matroid.

$\implies \exists \boldsymbol{F_1}, \boldsymbol{F_2} \in \mathcal{I}$ such that $|\boldsymbol{F_1}| < |\boldsymbol{F_2}|$ but $\boldsymbol{F_1} \cup \{\boldsymbol{e}\} \notin \mathcal{I}$ for all $\boldsymbol{e} \in \boldsymbol{F_2} \setminus \boldsymbol{F_1}$

# Proof

Contradiction. Suppose false $\implies (U, \mathcal{I})$ hereditary but not matroid.

$\implies \exists F_1, F_2 \in \mathcal{I}$ such that $|F_1| < |F_2|$ but $F_1 \cup \{e\} \notin \mathcal{I}$ for all $e \in F_2 \setminus F_1$



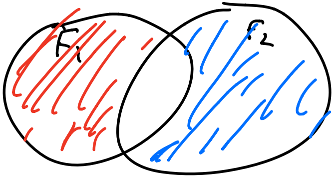

Easy facts:

1. $|F_2 \setminus F_1| > |F_1 \setminus F_2|$
2. $|F_2 \setminus F_1| \geq 1$
3. $|F_1 \setminus F_2| \geq 1$ (hereditary)

# Proof

Contradiction. Suppose false $\implies$ $(U, \mathcal{I})$ hereditary but not matroid.

$\implies$ $\exists F_1, F_2 \in \mathcal{I}$ such that $|F_1| < |F_2|$ but $F_1 \cup \{e\} \notin \mathcal{I}$ for all $e \in F_2 \setminus F_1$




Easy facts:

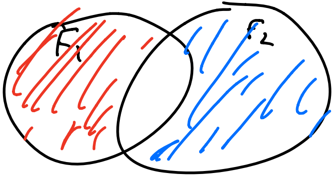
1. $|F_2 \setminus F_1| > |F_1 \setminus F_2|$
2. $|F_2 \setminus F_1| \geq 1$
3. $|F_1 \setminus F_2| \geq 1$ (hereditary)

$\implies$ $\exists \epsilon > 0$ such that $0 < (1+\epsilon)|F_1 \setminus F_2| < |F_2 \setminus F_1|$

# Proof

Contradiction. Suppose false $\implies (U, \mathcal{I})$ hereditary but not matroid.

$\implies \exists F_1, F_2 \in \mathcal{I}$ such that $|F_1| < |F_2|$ but $F_1 \cup \{e\} \notin \mathcal{I}$ for all $e \in F_2 \setminus F_1$




Easy facts:

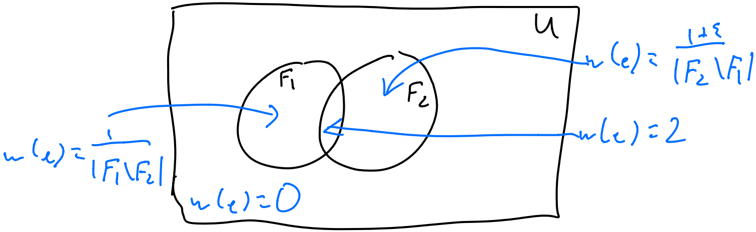
1. $|F_2 \setminus F_1| > |F_1 \setminus F_2|$
2. $|F_2 \setminus F_1| \geq 1$
3. $|F_1 \setminus F_2| \geq 1$ (hereditary)

$\implies \exists \epsilon > 0$ such that $0 < (1+\epsilon)|F_1 \setminus F_2| < |F_2 \setminus F_1|$

$$\implies \frac{1}{|F_1 \setminus F_2|} > \frac{1+\epsilon}{|F_2 \setminus F_1|}$$

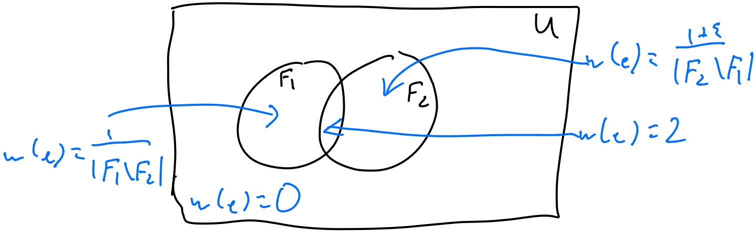
# Proof (cont'd)

Use fact that $\frac{1}{|F_1 \setminus F_2|} > \frac{1+\epsilon}{|F_2 \setminus F_1|}$ to define weights.



# Proof (cont'd)

Use fact that $\frac{1}{|F_1 \setminus F_2|} > \frac{1+\epsilon}{|F_2 \setminus F_1|}$ to define weights.

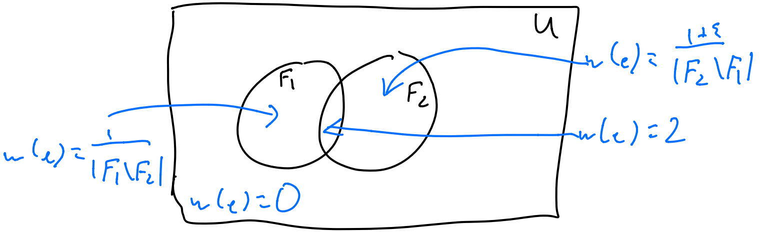



Greedy:

- Adds all of $F_1 \cap F_2$
- Adds all of $F_1 \setminus F_2$
- Can't add any of $F_2 \setminus F_1$

# Proof (cont'd)

Use fact that $\frac{1}{|F_1 \setminus F_2|} > \frac{1+\epsilon}{|F_2 \setminus F_1|}$ to define weights.



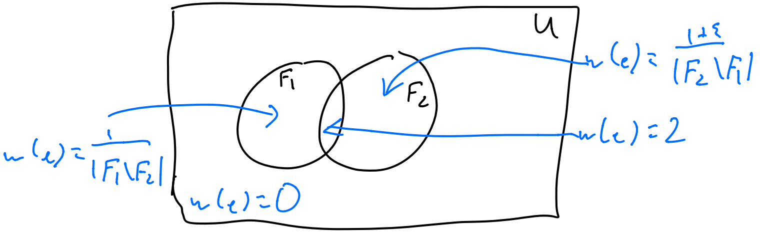

Greedy:

- Adds all of $F_1 \cap F_2$
- Adds all of $F_1 \setminus F_2$
- Can't add any of $F_2 \setminus F_1$

$$w(\text{greedy}) = 2|F_1 \cap F_2| + |F_1 \setminus F_2| \frac{1}{|F_1 \setminus F_2|}$$
$$= 2|F_1 \cap F_2| + 1$$

# Proof (cont'd)

Use fact that $\frac{1}{|F_1 \setminus F_2|} > \frac{1+\epsilon}{|F_2 \setminus F_1|}$ to define weights.




Greedy:

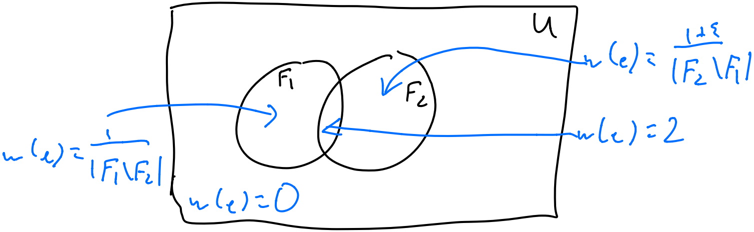
- Adds all of $F_1 \cap F_2$
- Adds all of $F_1 \setminus F_2$
- Can't add any of $F_2 \setminus F_1$

$$
\begin{aligned}
w(\text{greedy}) &= 2|F_1 \cap F_2| + |F_1 \setminus F_2| \frac{1}{|F_1 \setminus F_2|} \\
&= 2|F_1 \cap F_2| + 1
\end{aligned}
$$

$$
\begin{aligned}
w(F_2) &= 2|F_1 \cap F_2| + |F_2 \setminus F_1| \frac{1+\epsilon}{|F_2 \setminus F_1|} \\
&= 2|F_1 \cap F_2| + 1 + \epsilon
\end{aligned}
$$

# Proof (cont'd)

Use fact that $\frac{1}{|F_1 \setminus F_2|} > \frac{1+\epsilon}{|F_2 \setminus F_1|}$ to define weights.




Greedy:

- Adds all of $F_1 \cap F_2$
- Adds all of $F_1 \setminus F_2$
- Can't add any of $F_2 \setminus F_1$

$$w(\text{greedy}) = 2|F_1 \cap F_2| + |F_1 \setminus F_2| \frac{1}{|F_1 \setminus F_2|} = 2|F_1 \cap F_2| + 1$$

$$w(F_2) = 2|F_1 \cap F_2| + |F_2 \setminus F_1| \frac{1+\epsilon}{|F_2 \setminus F_1|} = 2|F_1 \cap F_2| + 1 + \epsilon$$

Greedy not optimal: contradiction!