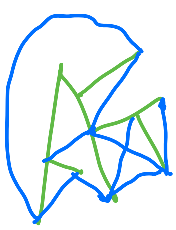
CS 331, Fall 2024
Lecture 10 (9/30)

Today: – Matroids
– Stable matching

Matroids (Part IV, Section 4.2)

Last time: Minimum Spanning tree

Goal: output spanning tree T
(forest of maximal size)



of minimum total weight

$$f(T) = \sum_{e \in T} w_e$$

Our approach: greedy.
(Kruskal)

– sort by weight
– take any edge that forms no cycle

Admits a significant generalization!

Let $(\mathcal{I}, [m])$ be a set system

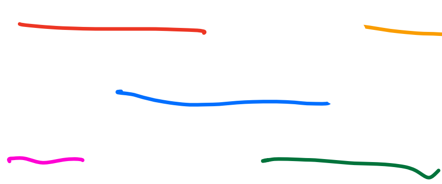
$\mathcal{I}$: independent sets; $[m]$: base set

$\mathcal{I}$ is <u>set of subsets</u>:

$$\mathcal{I} = \{ S_1, S_2, S_3, \ldots S_k \}$$

$$S_1 \subseteq [m], \quad S_2 \subseteq [m], \quad S_3 \subseteq [m], \quad \ldots, \quad S_k \subseteq [m]$$

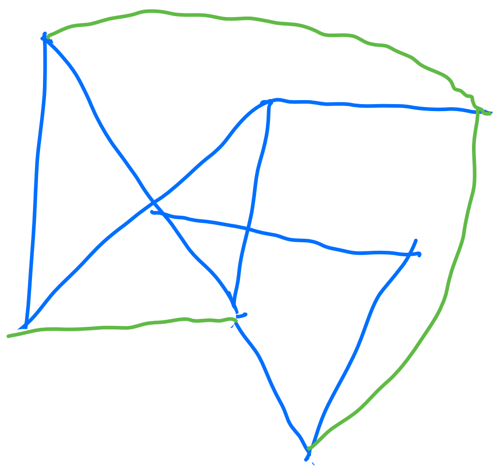
e.g. Scheduling problem



Say a set is independent if it has no overlaps

$$\mathcal{I} = \{ \ , \ , \ , \ , \ , \ldots , \ \ \ \ \}$$

e.g. forests in a graph = graphic matroid



say a set is independent if it has no cycle

Set system satisfies heredity if:

$$S \subset T, \ T \in \mathcal{I} \Rightarrow S \in \mathcal{I}$$

Pretty mild.

scheduling ✓ graphic ✓

Set system satisfies exchange if you can always "steal an item" from any larger independent set.

(the secret sauce for MST greedy)

$$S, T \in \mathcal{I}, \ |S| < |T|$$

$$\Rightarrow S \cup \{e\} \in \mathcal{I} \text{ for some } e \in \mathcal{I}$$

last class!

scheduling ✗

graphic ✓

If both heredity and exchange hold, we call the set system $(\mathcal{I}, [m])$ a matroid.

Basis: maximal independent set

(adding one more element creates a dependence)

Claim: In matroids we have that

$|S| = |T| = r$ for all bases S, T

(rank)

Proof: Suppose otherwise for contradiction,

$|S| < |T|$. Exchange:

$S \cup \{e\} \in \mathcal{I}$, S not maximal.

Basis selection: each $e \in [m]$ has weight.

$$\min_{S \text{ is basis}} f(S) = \sum_{e \in S} w_e.$$

= max ok too.

Claim: Suppose $\mathcal{I}$ satisfies heredity.

Then greedy optimal iff exchange.

Proof: $(\Leftarrow)$

```
BasisSelection((I, (m)), w):
  Sort (m) by weight
  T ← ∅
  For e ∈ (m):
    If T ∪ {e} ∈ I : T ← T ∪ {e}
  Return T
```

The proof is identical to MST.

Suppose ALG no longer optimal after k steps.

Exchange + more weight : We're optimal ?

less weight : we would have taken ?

$(\Rightarrow)$ See lecture notes for a reference in Erickson.

Application 1: Ruling out greedy.

Scheduling greedy by weight: not optimal.

Application 2: feature selection in ML.

Represent data as vectors $a \in \mathbb{R}^n$

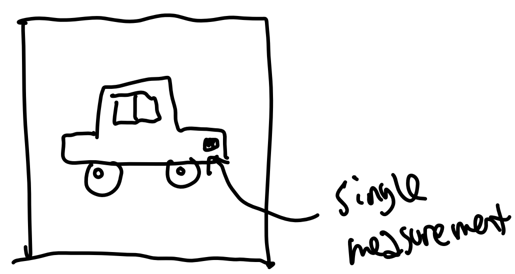
Features = linear measurements of data

$$A = \begin{pmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_m \\ | & | & & | \end{pmatrix} \in \mathbb{R}^{n \times m}$$

$$A^T x = b$$

data features

e.g. pixel values



$\boxed{\text{Aside}}$ Linear independence

We say $v \in \mathbb{R}^n$ is <u>linear combination</u> of $\{a_i\}_{i \in [m]} \equiv A$ if $Ac = v$

$$\begin{pmatrix} | & | & & | \\ a_1 & a_2 & \cdots & a_m \\ | & | & & | \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{pmatrix} = v$$

$c_1 a_1 + c_2 a_2 + \cdots + c_m a_m$

"redundant hint"

If $Ac \neq 0 \ \forall c \neq 0$, we say A linearly independent

Key concepts: $\text{rank}(A)$ = max # columns that are linearly independent

$\text{span}(A)$ = all linear combinations Ac

$\text{rank}(A)$ = dimension of $\text{span}(A)$

Claim: linearly independent vectors = matroid

Proof: Heredity $\quad Ac \neq 0$ if $c \neq 0$

$[A_{:S}] c_S \neq 0$ if $c_S \neq 0$

Exchange $\quad$ Let A, B independent

$\text{rank}(A) < \text{rank}(B)$

If no exchange, $AC = B$

$\text{span}(AC) \subseteq \text{span}(A)$

$\text{rank}(AC) \leq \text{rank}(A) < \text{rank}(B)$

Punchline: greedy suffices for weighted feature selection

e.g.

$$\max_{S \subseteq [m]} \sum_{i \in S} w_i \qquad \text{(feature importances)}$$

$A_{:S}$ independent $\qquad$ (feature diversity)

Stable matching (Part IV, Section 5)

Setup: n applicants {Alice, Bob, ...}

n job openings {Google, Apple, ...}

Input: Preference lists

$a: \alpha > \gamma > \beta$ $\alpha: b > a > c$

$b: \gamma > \alpha > \beta$ $\beta: c > a > b$

$c: \alpha > \beta > \gamma$ $\gamma: a > b > c$

Output: Stable matching

(a, α)	(b, α)
(b, β)	(c, β)
(c, γ)	(a, γ)
unstable	stable

What's the difference?

(b, α) unstable pair:

b prefers $\alpha > \beta$

α prefers $b > a$

backroom deal...

Idea: fix instability (like inversions)

Issue: cycles

$$\begin{matrix}(a,\alpha)\\(b,\beta)\\(c,\gamma)\end{matrix} \xrightarrow{(b,\alpha)} \begin{matrix}(b,\alpha)\\(a,\beta)\\(c,\gamma)\end{matrix} \xrightarrow{(b,\gamma)} \begin{matrix}(c,\alpha)\\(a,\beta)\\(b,\gamma)\end{matrix}$$

$$\xrightarrow{(a,\gamma)} \begin{matrix}(c,\alpha)\\(b,\beta)\\(a,\gamma)\end{matrix} \xrightarrow{(a,\alpha)} \begin{matrix}(a,\alpha)\\(b,\beta)\\(c,\gamma)\end{matrix}$$

Gale-Shapley algo

- Hugely influential in practice
  (Resident matching, faculty hiring, etc.)
- Nobel Prize in Economics, 2012

StableMatching$(\{A_a\}_{a\in[n]}, \{J_\alpha\}_{\alpha\in[n]})$:

$M \leftarrow \emptyset$, $i_\alpha \leftarrow 1 \quad \forall \alpha \in [n]$

While $\exists$ unmatched job α:

$a \leftarrow J_\alpha[i_\alpha]$ // favorite applicant w/o reject

If a unmatched: $M \leftarrow M \cup \{(a,\alpha)\}$

Elif a prefers α to β (current match):

$M \leftarrow M \setminus \{(a,\beta)\} \cup \{(a,\alpha)\}$

i_β++ // reneged

Else: i_α++ // rejected

Return M

Runtime: Algo ends when $|M| = n$.

$|M|$ only grows, so no list exhausted

Every iter: • pointer i_α grows $n^2 \times$

OR

• $|M|$ grows $n \times$

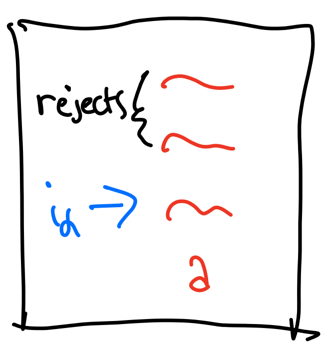
Total: $O(n^2)$ linear time!

Correctness: Perfect matching

• Applicant a unmatched, never offered

• Unmatched α has not reached a

α's list:



Stable matching

- Let $\{(a, \alpha), (b, \beta)\} \in M$
- Suppose (a, β) unstable
- If a had offer from β then would switch
- But β likes $a > b$, $\Rightarrow \Leftarrow$

Structural fact: Outcome always same, regardless of tiebreaking.

Say a feasible for α, α feasible for a $\Big\}$ if $(a, \alpha) \in M$ stable

Every job α gets favorite feasible a

applicant a gets least-favorite feasible α