MS&E 236 / CS 225: Lecture 5
Graphs and graph algorithms

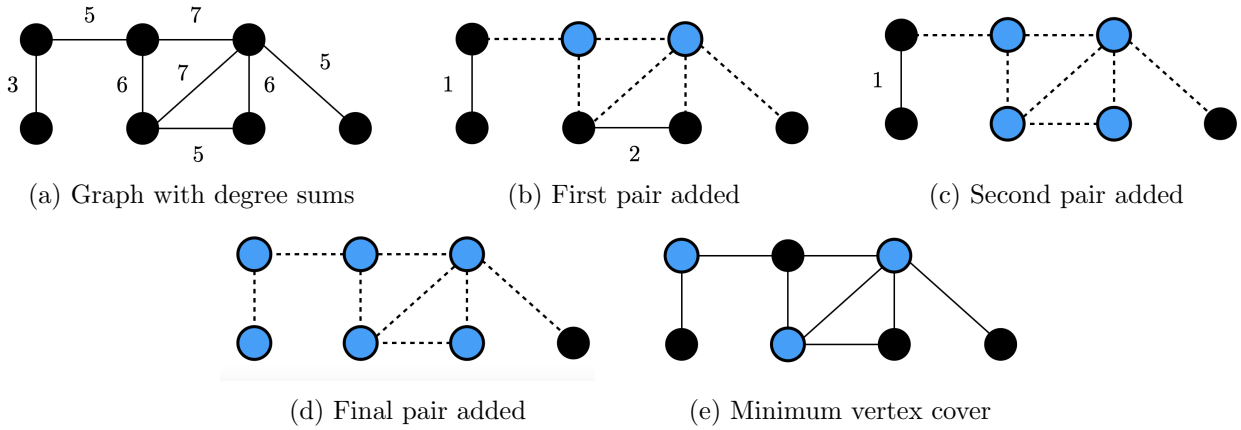
April 14, 2024

1 Plan for today

- Graphs
- Several famous graph algorithms:
  - Minimum vertex cover
  - Maximum cut

2 Graphs

- Set of objects, called *nodes*, connected by *edges*
- Nodes and edges can have data associated with them
- Example: molecule
  - Nodes are atoms, data describe atom type (carbon, nitrogen, hydrogen, ...)
  - Edges are bonds, data describe bond type (single bound, double bond, ...)
- Example: rail network
  - Nodes are cities, edges are connections with data describing journey time
- Example: social network. Nodes are people, edges are friendships
- Notation: Graph $G = (V, E)$
  - $V = \{1, \ldots, n\}$ is a set of nodes/vertices
  - E is a set of edges/links. Edge between $i, j \in V$ denoted (i, j)
  - Neighborhood of node $i \in V$ is the set of nodes it's connected to by an edge
    - Denoted $N(i) \subseteq V$
    - Node's *degree* is $|N(i)|$



(a) Graph with degree sums (b) First pair added (c) Second pair added

(d) Final pair added (e) Minimum vertex cover

Figure 1: 2-approximation algorithm for the minimum vertex cover

3 Minimum vertex cover

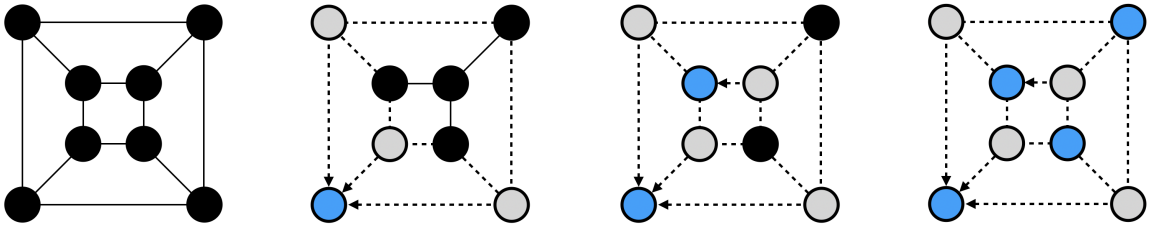
Definition 3.1 (Vertex cover)**.** Set $S \subseteq V$ s.t. for every $(i, j) \in E$, $i \in S$, $j \in S$, or both

- Goal: given a graph G, find smallest vertex cover
- Example application:
  - Installing cameras in corners (nodes) covering all hallways (edges) on a floor
- Finding the minimum vertex cover is NP-hard
- 2-approximation algorithm (see Figure 1):
  1. Initialize the vertex cover $S \leftarrow \emptyset$
  2. While $E \neq \emptyset$:
     (a) Choose the edge in $(i, j) \in E$ that maximizes $|N(i)| + |N(j)|$
     (b) Add i, j to the vertex cover: $S \leftarrow S \cup \{i, j\}$
     (c) Remove all edges incident to i and j from E
  3. Return S

Theorem 3.2. *Let S^* be the minimum vertex cover. Then $|S| \leq 2|S^*|$*

Proof.
- Let E' be the set of edges the algorithm picked in Step 2a
- To cover E', S^* must include at least one node from each edge in E'
- No two edges in E' share an endpoint
  - Once an edge is chosen, all edges incident to its endpoints are deleted
- Therefore, no two edges are covered by the same vertex in S^*, so $|S^*| \geq |E'|$
- As a result, $|S| = 2|E'| \leq 2|S^*|$

□



(a) Original graph (b) First vertex (c) Second vertex (d) Third and fourth

Figure 2: $\frac{1}{1+\Delta}$-approximation algorithm for maximum independent set

4 Maximum independent set

Definition 4.1 (Independent set)**.** $S \subseteq V$ s.t. no vertices in S are connected by an edge

- Goal: given a graph G, find largest independent set
- Δ: maximum degree in G
- $\frac{1}{1+\Delta}$-approximation algorithm (see Figure 2):
  1. Initialize the independent set $S \leftarrow \emptyset$
  2. While $V \neq \emptyset$:
     (a) Choose vertex $v \in V$ with minimum degree $|N(v)|$
     (b) Add v to independent set: $S \leftarrow S \cup \{v\}$
     (c) Remove v and its neighbors from V
  3. Return S

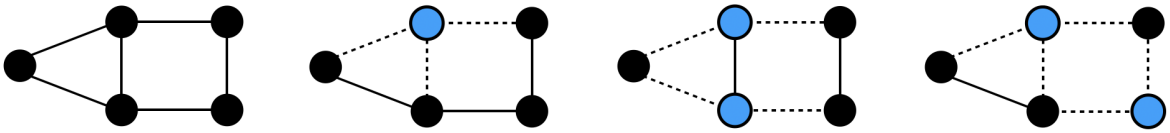
Theorem 4.2. *Let V^* be the maximum independent set. Then $|S| \geq \frac{|V|}{\Delta+1} \geq \frac{|V^*|}{\Delta+1}$*

Proof.
- u is in $V \setminus S$ because it's removed as a neighbor of some node $v \in S$
  - Removed when algorithm adds v to S
  - "Charge" u to v
- $v \in S$ can be charged $\leq \Delta$ times (has $\leq \Delta$ neighbors)
- Implies that $|V \setminus S| \leq \Delta|S|$
- So $|V| = |S| + |V \setminus S| \leq |S|(1 + \Delta)$

□

5 Maximum cut

- One of the most famous approximation algorithms was developed for max-cut
  - Goemans-Williamson algorithm; we'll see a simpler algorithm today
- A *cut* in a graph is a subset of its vertices $S \subseteq V$



(a) Original graph (b) First swap (c) Second swap (d) Optimal cut

Figure 3: 2-approximation algorithm for maximum cut

- The *weight* of a cut $w(S)$ is the number of edges that cross S to $V \setminus S$
- Goal: find a cut with maximum weight
- 2-approximation algorithm (Figure 3):
  1. Begin with arbitrary initial cut $S \subseteq V$
  2. For each $i \in V$, let w_i be the cut's weight if you switched the side i is on
  3. If $w_i \leq w(S)$ for all i, terminate and return S
  4. Else, let $i^* = \text{argmax} w_i$. Switch the side of the cut i^* is on and return to step 2

Theorem 5.1. *Let S^* be the maximum weight cut. Then $w(S) \geq \frac{1}{2} w(S^*)$*

Proof.
- First, we claim that $w(S) \geq \frac{|E|}{2}$
  - For any vertex $i \in S$ some vertices in $N(i)$ are in S, some are in $V \setminus S$
  - Must be more vertices in $V \setminus S$
    - Otherwise, we could switch i to $V \setminus S$ and improve the cut's weight
  - Therefore, number of edges incident to i that cross the cut is $\geq \frac{|N(i)|}{2}$

$$\begin{aligned} w(S) &\geq \frac{1}{2} \sum_{i \in V} (\#\text{ of } i\text{'s incident edges that cross the cut}) \\ &\geq \frac{1}{2} \sum_{i \in V} \frac{|N(i)|}{2} \\ &\geq \frac{1}{4} \sum_{i \in V} |N(i)| \\ &= \frac{1}{4} \cdot 2|E| \end{aligned}$$

- Meanwhile, $w(S^*) \leq |E|$, so $w(S) \geq \frac{|E|}{2} \geq \frac{1}{2} w(S^*)$

□

6 Greedy algorithms

- "Greedily" choose a node to add/remove from the solution maximizing some score
  - In vertex cover: $|N(i)| + N(j)|$

  - Independent set: $|N(v)|$
  - Max cut: improvement of cut weight when node is swapped

- Hand-designed scores we saw today lead to decent approximation algorithms
- This module: can a ML approach learn an even better scoring rule?