Graphs II: Matching and Coloring

Reading

- Malik Magdon-Ismail. Discrete Mathematics and Computing.
  - Chapter 12

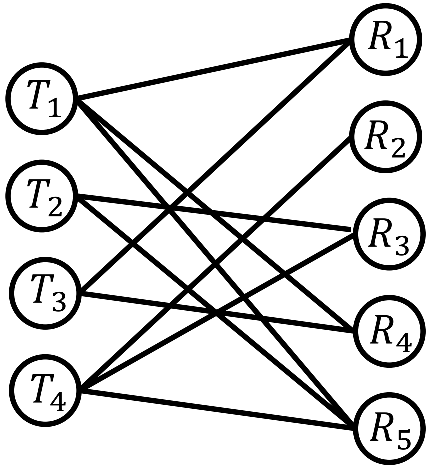
- Office hours:
  - M 1-2pm, W 4-5pm, F 9-10am (Lally 309)

Overview

- Matching
  - Bipartite graphs
  - Stable matching
- Coloring.
  - Conflict graphs
- Other graph problems
  - Connected components, spanning tree, Euler cycle, network flow (easy)
  - Hamiltonian cycle, facility location, vertex cover, dominating set (hard)

Bipartite graphs

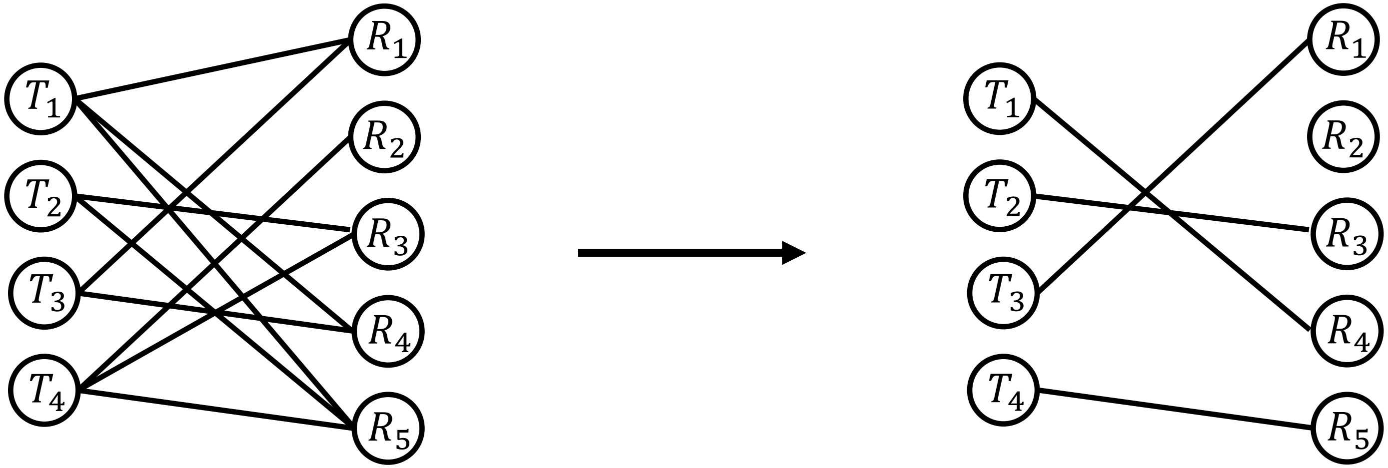
- A bipartite graph consists of two sets of vertices
  - Edges exist only across the two sets



- Bipartite graphs appear everywhere in life
  - Suppose you have a number of resources that are responsible for completing some tasks
    - E.g., each one of you wants to train your favorite ChatGPT model on CCI, but CCI only has so many GPUs
  - Suppose there is a node for each CS class and a node for each CS student
    - An edge exists if a student s is in class c
  - Sports teams/players can be divided similarly
  - Etc.

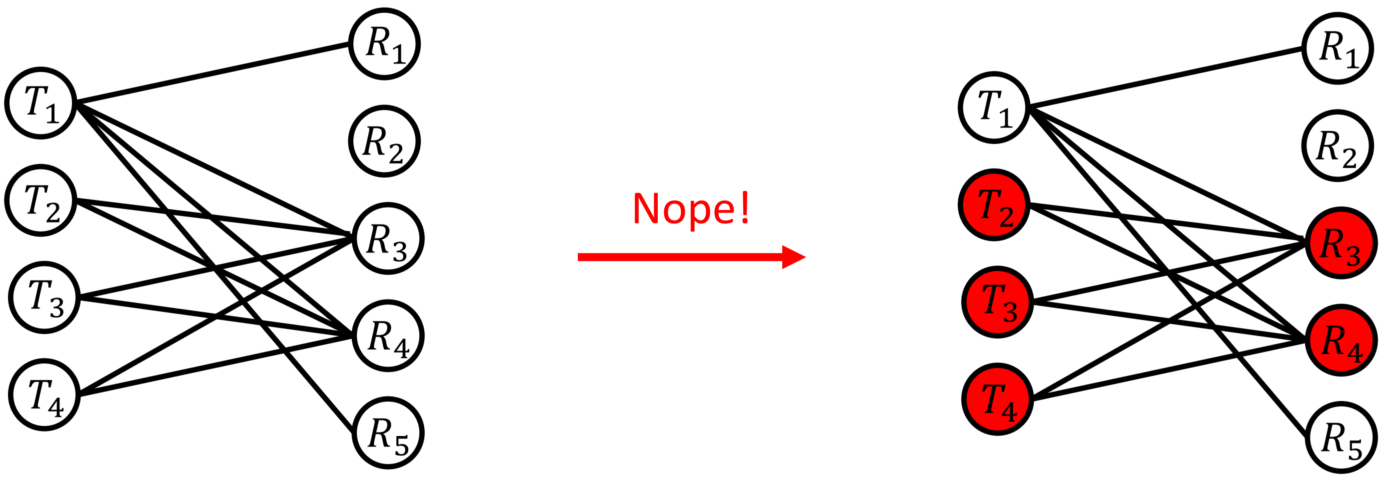
Bipartite Matching

- A bipartite graph can be left-matched if there exists a set of edges such that each left-vertex is covered by exactly one edge
  - E.g., there are enough resources for all tasks
- Can this graph be left-matched?



Bipartite Matching, cont’d

- A bipartite graph can be left-matched if there exists a set of edges such that each left-vertex is covered by exactly one edge
  - E.g., there are enough resources for all tasks
- Can this graph be left-matched?

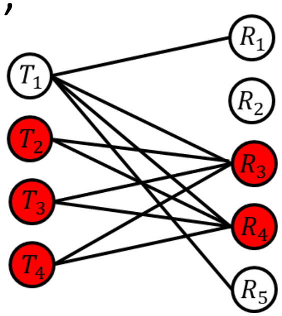


Don’t have enough resources for tasks T_2, T_3, T_4

Hall's Theorem

- Turns out the condition on the previous slide is true for all graphs and is a sufficient condition for matching in any graph
- For a given left subset X, let $N(X)$ be the set of "neighbors" of X, i.e., corresponding nodes on the right with edges to X
  - Let $X = \{T_2, T_3, T_4\}$. What is $N(X)$?

$$N(X) = \{R_3, R_4\}$$



- *Theorem [Hall's Theorem]*. Suppose that for *all* left-subsets X, $|X| \leq |N(X)|$ (Hall's "matching condition"). Then, there is a matching which covers every left-vertex.
  - Hall's Theorem says that the necessary condition is also sufficient.

Proof of Hall's Theorem

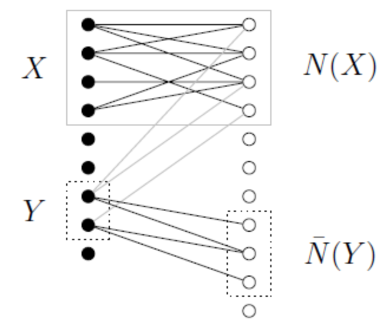
- *Theorem [Hall's Theorem]*. Suppose that for *all* left-subsets X, $|X| \leq |N(X)|$ (Hall's "matching condition"). Then, there is a matching which covers every left-vertex.
- *Proof*. By strong induction on the number of left-vertices.
  - [**Base Case**] Suppose the number of left-vertices is $n = 1$. As long as the left-vertex has at least one outgoing edge, then it can be covered. Check.

Proof of Hall's Theorem

Rensselaer

- *Theorem [Hall's Theorem]*. Suppose that for *all* left-subsets X, $|X| \le |N(X)|$ (Hall's "matching condition"). Then, there is a matching which covers every left-vertex.
- *Proof*. By strong induction on the number of left-vertices.
  - [**Induction Step**] Suppose we have n left-vertices and suppose $P(n)$ is T, i.e., $|X| \le |N(X)|$ for all subsets X. Need to prove that $P(n) \to P(n+1)$.
    - Case 1. There is a proper left-subset X, with $1 \le |X| \le n$, for which $|X| = |N(X)|$.
      - X has a matching into $N(X)$ (using strong induction)
      - Let Y be any left-subset $Y \subseteq \bar{X}$
      - The neighbors of Y, $N(Y)$, could overlap with $N(X)$
        - » Let $\bar{N}(Y) = N(Y) \backslash N(X)$
      - by the matching condition,

$$|N(X)| + \left|\bar{N}(Y)\right| = |N(X \cup Y)| \ge$$
$$\ge |X \cup Y| = |X| + |Y|$$

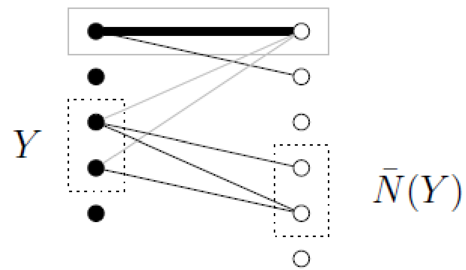


      - Since $|X| = |N(X)|$, it follows that $\left|\bar{N}(Y)\right| \ge |Y|$
      - Since this is true for any subset $Y \subseteq \bar{X}$, then $\bar{X}$ has a separate matching into $\bar{N}(X)$

Proof of Hall's Theorem, cont'd

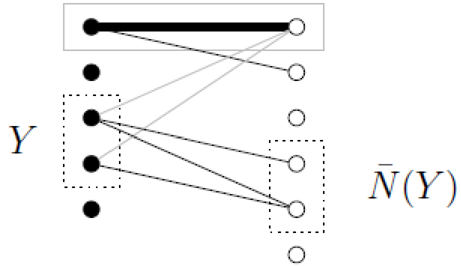
Rensselaer

- *Theorem [Hall's Theorem]*. Suppose that for *all* left-subsets X, $|X| \leq |N(X)|$ (Hall's "matching condition"). Then, there is a matching which covers every left-vertex.
- *Proof*. By strong induction on the number of left-vertices.
  - [**Induction Step**] Suppose we have n left-vertices and suppose $P(n)$ is T, i.e., $|X| \leq |N(X)|$ for all subsets X. Need to prove that $P(n) \rightarrow P(n+1)$.
    - Case 2. For *every* proper left-subset X (with $1 \leq |X| \leq n$), $|X| < |N(X)|$.
      - Match the first left-vertex, X_1, along any edge to a neighbor, n_1
      - Take any left-subset Y of the remaining graph of n left vertices
      - What do we know about $N(Y)$?
        - » Either $n_1 \notin N(Y)$, i.e., $N(Y) = \bar{N}(Y)$
        - » or $n_1 \in N(Y)$, i.e., $|N(Y)| = |\bar{N}(Y)| + 1$
        - » So finally, $|\bar{N}(Y)| \geq |N(Y)| - 1$



Proof of Hall's Theorem, cont'd

- *Theorem [Hall's Theorem]*. Suppose that for *all* left-subsets X, $|X| \leq |N(X)|$ (Hall's "matching condition"). Then, there is a matching which covers every left-vertex.
- *Proof*. By strong induction on the number of left-vertices.
  - [**Induction Step**] Suppose we have n left-vertices and suppose $P(n)$ is T, i.e., $|X| \leq |N(X)|$ for all subsets X. Need to prove that $P(n) \rightarrow P(n+1)$.
    - Case 2. For *every* proper left-subset X (with $1 \leq |X| \leq n$), $|X| < |N(X)|$.
      - Match the first left-vertex, X_1, along any edge to a neighbor, n_1
      - Take any left-subset Y of the remaining graph of n left vertices
$$\begin{aligned}\left|\bar{N}(Y)\right| &\geq |N(Y)| - 1 \\ &\geq |Y| + 1 - 1 = |Y|\end{aligned}$$
      - The remaining left-vertices have a matching to the remaining right-vertices (induction hypothesis).
    - In both cases, there is a left-matching which covers the $n+1$ left-vertices.



Hall’s Theorem Practice.

- **Exercise.** If (min left-degree) $\geq$ (max right-degree) then Hall’s condition holds.
  - Why is this a better idea than actually verifying Hall’s condition?
    - Much easier to check
    - Enumerating all subsets takes a loooong time
- **Example 12.3.** Building Latin Squares.

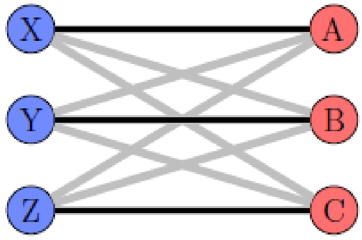
Stable Matching

- Also known as stable marriage
- Suppose that we add preferences to the matching problem
  - Each left-vertex has preferences for all right vertices and vice-versa
  - E.g., suppose left vertices are A, B, C and right-vertices are X, Y, Z

	X	Y	Z
1.	A	A	B
2.	B	C	A
3.	C	B	C

	A	B	C
1.	Z	Y	Z
2.	Y	X	X
3.	X	Z	Y

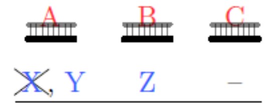
- Stable matching is used for matching medical schools with students applying for residency
  - E.g., student A prefers school Z to Y to X
  - E.g., school X prefers student A to B to C
- The point is to avoid a volatile matchup
  - Why is this volatile?
  - A prefers Y to X and Y prefers A to B



Stable Matching Algorithm Overview

- Round 1. Interviews
  - All schools interview their top candidates
  - Everyone creates their ranking
  - Student A rejects school X as lowest ranked
  - X will not interview A again

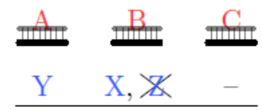
X	Y	Z
A	A	B
B	C	A
C	B	C



A	B	C
Z	Y	Z
Y	X	X
X	Z	Y

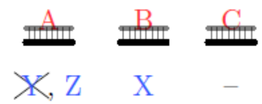
- Round 2. More Interviewing
  - (In practice, this is done algorithmically, after everyone submits their ranking)
  - Y and Z invite A and B respectively
  - X invites B
  - B rejects Z; Z erases B; X and Y will return

X	Y	Z
A	A	B
B	C	A
C	B	C

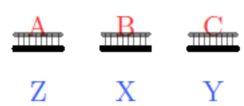


- Round 3. More interviewing
  - Z invites A
  - A rejects Y for their top-choice Z.

X	Y	Z
A	A	B
B	C	A
C	B	C



- Round 4. Final round
  - All parties are paired in a non-volatile manner

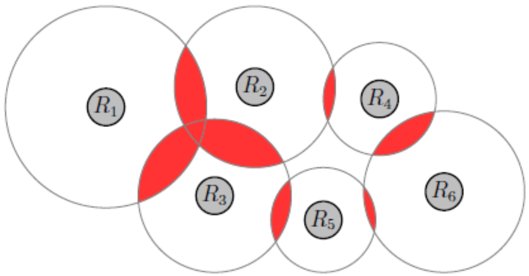


Stable Matching Properties

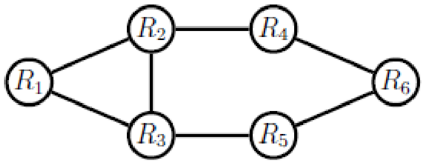
- *Theorem. [Gale-Shapely, 1962]*
  - For n students and schools, the algorithm ends after at most n^2 rounds
  - Every student and school will be matched at the end
  - The resulting set of matchings is stable (no volatile pairs).
- In practice, there's a game theoretic aspect as well, which we won't talk about
  - Suppose a student has a 1/100 chance of getting into their top choice but a 1/10 chance of getting into their 2nd choice
    - The student should probably rank their 2nd choice higher
  - Schools and students collude during interviews
    - They agree to match each other in order to avoid unexpected outcomes through the algorithm
- Stable matching is a weird system but the residency problem was getting out of hand in the mid 20th century
  - Students were getting offers early in their junior years
  - Stable matching provides a fair mechanism that is guaranteed to be stable

Conflict Graphs and Coloring

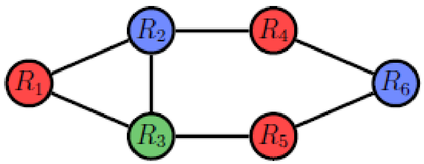
- Task 1: Assigning radio frequencies
  - Suppose we have 6 radio stations arranged as follows



  - Stations broadcasting to the same listener (red areas) need different frequencies (conflict).
  - How do we build a conflict graph based on the above placement?

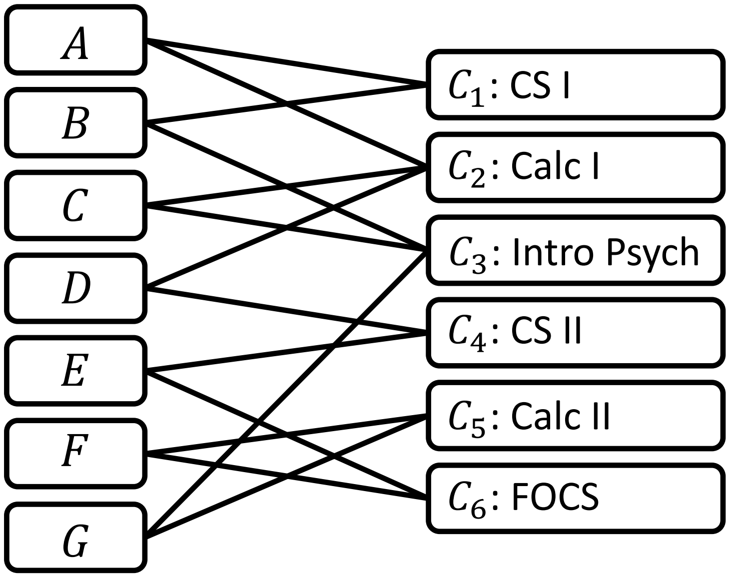


  - How many frequencies do you need?

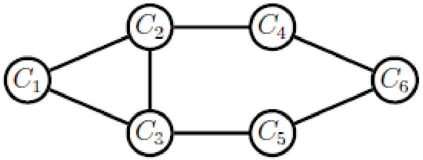


Conflict Graphs and Coloring, cont'd

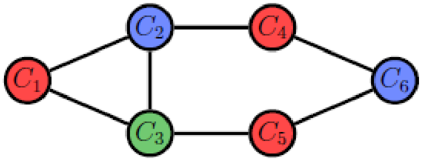
- Task 2: Scheduling Course Exams



  - Courses with the same student need different exam-time (conflict) – A causes CS I and Calc I to conflict.



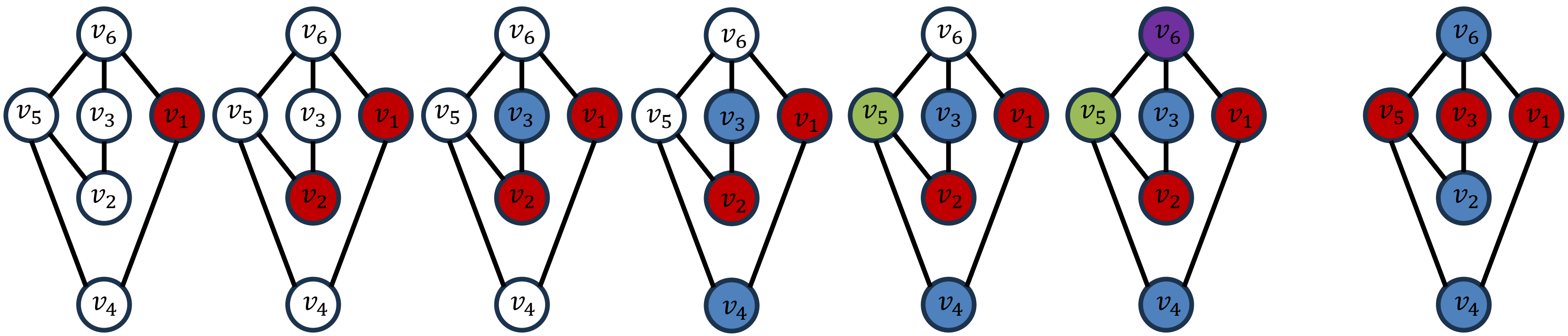
  - All students need to take all their exams. How many exam slots do you *need*?



Sequential Greedy Coloring

1. Colors $\{1, 2, 3, \ldots\}$
2. Let $color(v_1) = 1$.
3. Assume that vertices $v_1, \ldots, v_i$ have been colored. Color v_{i+1} with the *smallest* color so that it does not conflict with any previously colored vertex.

- For visual effect, pick colors $\{1,2,3,4\}$ as $\{red, blue, green, puprple\}$



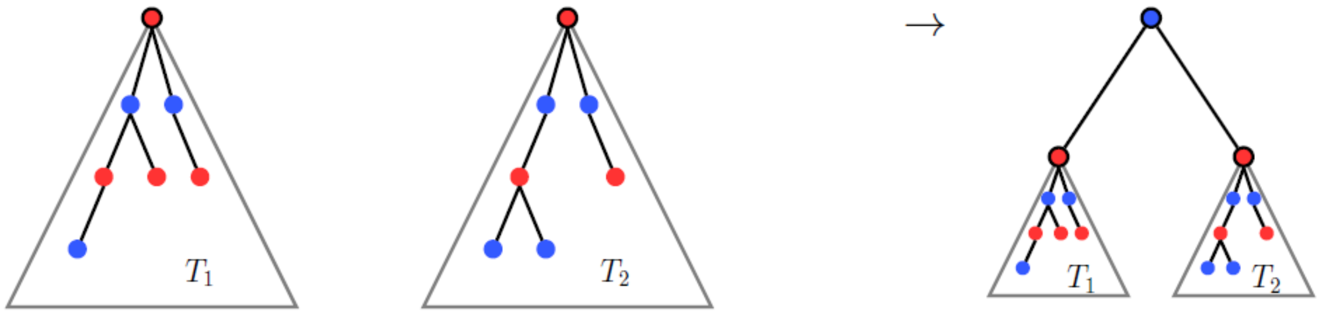
(2 colors suffice)

Sequential Greedy Coloring, cont'd

- **Chromatic Number $\chi(G)$.** The minimum number of colors needed.
- *Lemma***.** Using Sequential Greedy, $color(v_i) \leq \delta_i + 1$.
- *Theorem*. Chromatic number is bounded by maximum degree.
  - i.e., $\chi(G) \leq \Delta(G) + 1$, where $\Delta(G)$ is the max degree in G, $\Delta(G) = \max_i \delta_i$.

Trees are 2-Colorable

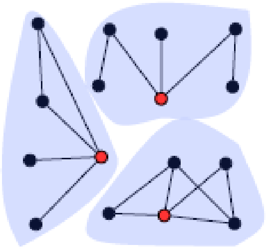
- Let us prove this for RBT's. We show that the constructor rule preserves 2-colorability.



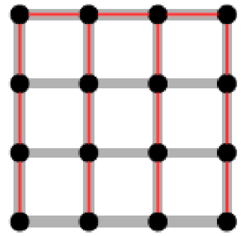
- How do we know T_1's root is colored red?
  - If not red, swap all colors – tree is still 2-colored
- A graph is bipartite if and only if its chromatic number is 2. Trees are bipartite.

Other Graph Problems

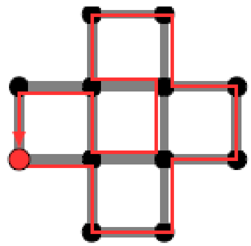
- **Connected Components.** For "viral" marketing, pick one vertex in each *connected component* (e.g. target the "central (red)" vertices). [easy]



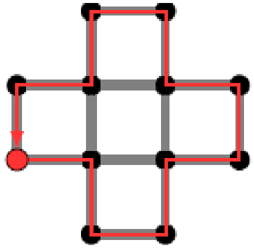
- **Spanning Tree.** In a road grid (gray), to maintain a *minimal* "highway system" that offers high-speed travel we can use a *spanning tree* (red). [easy]



- **Euler Cycle.** Every winter, Troy typically has a 1-foot snowfall. The snowplow should start at the depot, traverse every road *exactly once* and return to the depot, traversing an *Euler Cycle* (red). [easy]

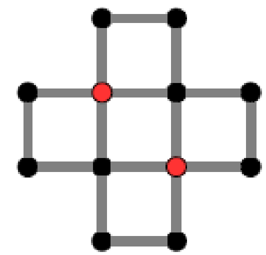


- **Hamiltonian Cycle.** A traveling sales man starts at work and visits every house (vertex) *exactly once*, returning to work. The salesperson follows a *Hamiltonian Cycle*. [hard]

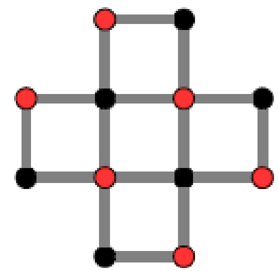


Other Graph Problems, cont'd

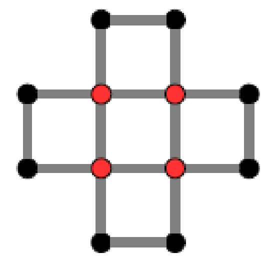
- **Facility Location (K-center).** McDonalds wants to place $K = 2$ restaurants (red) in a road network so that no customer has too drive far to reach their closest McDonalds. [hard]



- **Vertex Cover.** Place the minimum number of police at intersections so that all roads can be surveilled or "covered". The officers form a vertex cover. Can you do it with fewer than 6? [hard]



- **Dominating Set.** Place the fewest hospitals at intersections (vertices) so that every intersection is either at a hospital or one block away from one. The red hospitals are a *dominating set*. [hard]



- **Network Flow.** A *source*-ISP (blue) sends packets to a *sink*-ISP (red). What is the maximum transmission rate achievable without exceeding the link capacities? We achieved flow rate 10. [easy]