Neighbor Sets

- Let $G = (V_1 \cup V_2, E)$ be a bipartite graph.
- For any vertex $a \in V_1$, we define the neighbor set of $a$ as
  \[ N(a) = \{ u \in V_2 \mid (a, u) \in E \}. \]

  \[ N(a) = \{ u, v \} \]
  \[ N(b) = \{ v, w \} \]
More Neighbor Sets

- Let $G = (V_1 \cup V_2, E)$ be a bipartite graph.
- For any subset $A \subset V_1$, we define

$$N(A) = \{ y \in V_2 \mid (x, y) \in E \text{ for some } x \in A \}.$$

- $N(\{b, c, d\}) = \{u, v, w\}$
- $N(\{a, e\}) = \{u, w, x\}$

Hall's Marriage Theorem

- **Theorem.** Let $G = (V_1 \cup V_2, E)$ be a bipartite graph.
  - There exists a matching of size $|V_1|$ in $G$ if and only if every $A \subset V_1$ satisfies $|A| \leq |N(A)|$.

- We say that $G$ satisfies Hall's condition if for every $A \subset V_1$, we have $|A| \leq |N(A)|$.

*Philip Hall*
Neighbor Sets and Matchings

- Explain why there is no matching of size four in this graph:

```
  a -- u
  b -- v
  c   w
  d -- x
```

- $N\{b, c, d\} = \{v, w\}$, so we cannot find a match for all three vertices $b, c, d$.

Proving Hall’s Theorem

- **Easy direction.** If there exists a subset $A \subseteq V_1$ such that $|A| > |N(A)|$, then there is no matching of size $|V_1|$ in $G$.

- **It remains to prove.** If every subset $A \subseteq V_1$ satisfies $|A| \leq |N(A)|$, then there is a matching of size $|V_1|$ in $G$.
  - We now present a proof by induction.

(different than the proof from 6a)
Proof by Induction

- **We prove by induction on** $|V_1|$ **that**
  - If every subset $A \subseteq V_1$ satisfies $|A| \leq |N(A)|$, then there is a matching of size $|V_1|$ in $G$.

- **Induction basis.** When $|V_1| = 1$ the claim obviously holds.

- **Induction step.** We consider two cases:
  - Every nonempty $A \varsubsetneq V_1$ satisfies $|A| < |N(A)|$.
  - There exists $A \varsubsetneq V_1$ such that $|A| = |N(A)|$.

The First Case

- **Assume that** every nonempty $A \varsubsetneq V_1$ satisfies $|A| < |N(A)|$.
  - Consider an edge $(a, b) \in E$. **Remove** $a$ and $b$ **from** $G$ **to obtain a graph** $G' = (V_1' \cup V_2', E')$.
  - Every $A \subseteq V_1'$ satisfies $|A| \leq |N(A)|$.
  - By **the induction hypothesis**, $G'$ contains a matching of size $|V_1'| = |V_1| - 1$.
  - **Adding** the edge $(a, b)$ to the matching yields a matching of size $|V_1|$ in $G$. 
The Second Case

- Assume that there exists a non-empty $A \subsetneq V_1$ such that $|A| = |N(A)|$.
  - By the hypothesis, the induced subgraph on $A \cup N(A)$ contains a matching of size $|A|$.

Our Plan

- The red vertices in $V_1$ form a subset $A$ such that $|A| = |N(A)|$.
  - We have a matching between the red vertices. It remains to find a matching between the blue vertices.
The Second Case (cont.)

- Assume that there exists a non-empty $A \subset V_1$ such that $|A| = |N(A)|$.
  - By the hypothesis, the induced subgraph on $A \cup N(A)$ contains a matching of size $|A|$.
  - Consider the induced subgraph $G'$ on $V_1 \setminus A$ and $V_2 \setminus N(A)$.
  - For every subset $A' \subset V_1 \setminus A$, since in $G$ $|A \cup A'| \leq |N(A) \cup N(A')|$, then in $G'$ $|A'| \leq |N(A')|$. By the hypothesis, $G'$ contains a matching of size $|V_1 \setminus A|$.
- Thus, $G$ contains a matching of size $|V_1|$.

Reminder: National Resident Matching Program

- Every medical student who is about to graduate ranks hospitals in which she wants to do her residency.
- Every hospital ranks students that it is interested in.
- Every year, over 20,000 applicants apply to about 1,800 programs.
- How can we handle this?
The Corresponding Graph

- **Bipartite graph** – a vertex in $V_1$ for each student. A vertex in $V_2$ for each hospital.
- An **edge** between a student and a hospital that are **interested in each other**.
  - A matching corresponds to assigning students to hospitals.

The Problem

- Problem. **We did not consider the rankings of the students and hospitals.**
  - We might have chosen the red matching.
  - However, perhaps student $A$ **prefers** hospital $\beta$, student $B$ **prefers** hospital $\alpha$, and similarly for the hospitals.
Unstable Matchings

- We have a bipartite graph $G = (V_1 \cup V_2, E)$ such that each vertex has a ranking of the vertices that it would like to be connected to.
- We say that a matching $M$ of $G$ is unstable if there exists an edge $(a, b) \in E$ that is not in $M$ and that both $a$ and $b$ prefer to be matched to each other than to their current match (or are unmatched).

Etymology

- Why is this called “unstable”?
- The problem was originally formulated for matching men and women to be married.
  - Man A and woman A are married.
  - Man B and woman B are married.
  - If man A prefers woman B and woman B prefers man A, these are unstable marriages!
Example: An Unstable Matching

- **Rankings:**
  - A: 1. $\alpha$  2. $\gamma$  3. $\beta$
  - B: 1. $\alpha$  2. $\gamma$  3. $\beta$
  - C: 1. $\beta$  2. $\alpha$  3. $\gamma$
  - $\alpha$: 1. A  2. B  3. C
  - $\beta$: 1. C  2. B  3. A

- The red edges form an **unstable** matching since both A and $\alpha$ prefer the edge $(A, \alpha)$.

Stable Marriage Theorem

- **Theorem.** For any bipartite graph $G = (V_1 \cup V_2, E)$ with sets of preferences for each vertex, there exists a **stable matching**.

- Proven in 1962 by **Gale and Shapley**.
  - In 2012, the **Nobel Prize in Economics** was awarded to **Shapley** “for the theory of stable allocations and the practice of market design.”
A Better Matching

- A matching $M$ in $G$ is better than a matching $M'$, if every vertex of $V_2$ prefers its matched vertex in $M$ to its matched vertex in $M'$ (or has the same matched vertex in both).
  - No matched vertex in $V_2$ becomes unmatched.
  - The matching on the left is better iff $\alpha$ prefers $A$ to $B$.

Proof Idea

- We prove the stable marriage theorem by showing that for every unstable matching $M_i$, there exists a better matching $M_{i+1}$.
- We start with an empty matching $M_0$ and repeatedly find a better matching.
  - Starting from an empty matching, a better matching can be found at most $|V_1||V_2|$ times, so we eventually obtain a stable matching.
Acceptable Vertices

- Given a matching $M_i$, we say that $a \in V_1$ is acceptable to $b \in V_2$ if
  - $(a, b) \in E \setminus M_i$, and
  - $b$ prefers $a$ to its match in $M_i$.

- In the figure, $B$ is acceptable to $\beta$ iff $\beta$ prefers $B$ to $C$.

- A student $S$ is acceptable to a hospital if it likes $S$ better than its current student.

Happy Vertices

- We say that a vertex $a \in V_1$ is happy with a matching $M_i$ if either
  - $a$ is unmatched in $M_i$, or
  - $(a, b) \in M_i$, and $a$ prefers $b$ to all of the vertices of $V_2$ that finds $a$ acceptable.

- Intuitively, a student is happy if she is assigned to her favorite hospital out of the ones that like her better than their current choice (or is unmatched).
Building Happy Matchings

- We only consider matchings where *every vertex of $V_1$ is happy.*
  - This is the case for the empty matching $M_0$.

- Given a matching $M_i$:
  - Consider an unmatched vertex $a \in V_1$ and let $b$ denote $a$’s favorite among the vertices of $V_2$ for which $a$ is acceptable.
  - We set $M_{i+1} = M_i \cup \{(a, b)\}$ (possibly also removing an edge $(a', b) \in M_i$).
  - Notice that $M_{i+1}$ is better than $M_i$ and that *every vertex of $V_1$ is happy.*

Final Details

- Given a matching $M_i$:
  - What if every unmatched vertex $a \in V_1$ has no vertex in $V_2$ for which $a$ is acceptable.
    - Then $M_i$ is a stable matching!
  - What if there are no unmatched vertices in $M_i$?
    - Since every vertex of $V_1$ is happy, this again means that $M_i$ is stable.
Finding a Stable Matching

- The proof of the stable marriage theorem presents us with an **algorithm for finding a stable matching**:  
  - Build a sequence of matchings $M_i$ such that each is better than the previous one, until we obtain a stable matching.  
  - At each step the algorithm picks an unmatched element of $V_1$ and finds the best match for it (not paying much attention to the preferences of the vertices of $V_2$).

The End

- **Why not use the stable marriage algorithm to set houses for new students?**