ENGG 2440A: Discrete Mathematics for Engineers **Lecture 5**
The Chinese University of Hong Kong, Fall 2018

Graphs are an especially popular object of study in discrete mathematics. They represent a finite collection of objects and pairwise relationships between them. We already saw one example in disguise: A "friendship graph", which tells us which pairs of people within a group are friends.

To define graphs, we'll need to know a few basic things about sets, so let's talk about sex... oops, I meant sets; we'll come back to sex shortly.

1 Sets

A *set* is an unordered collection of objects, called the *elements* of the set. Each element in the set occurs exactly once. We can specify a set by listing its elements like this:

$$P = \{\text{Alice}, \text{Bob}, \text{Charlie}\}$$

This is the same set as $\{\text{Bob}, \text{Alice}, \text{Charlie}\}$. The elements of a set are unordered. We denote set membership and non-membership like this:

$$\text{Alice} \in P \qquad \text{Dave} \notin P.$$

We can also have sets consisting of other sets. For example, the set F may indicate which pairs of people within P are friends:

$$F = \{\{\text{Alice}, \text{Bob}\}, \{\text{Alice}, \text{Charlie}\}\}.$$

Every once in a while we will need to work with large or infinite sets. In such cases, listing all the elements is not an option, so we specify membership in a set by a predicate that its elements must satisfy.

For example, suppose we are talking about integers. Then the set E of all even numbers is the set of all integers n that satisfy the predicate "n is even". We write this as

$$\begin{aligned} E &= \{n \colon n \text{ is even}\} \\ &= \{n \colon \text{There exists } k \text{ such that } n = 2k\}. \end{aligned}$$

Some sets have standard names, like $\varnothing$ for the empty set, $\mathbb{Z}$ for the integers, $\mathbb{N}$ for the positive integers, $\mathbb{R}$ for the reals. The fact that we are talking about objects of a particular type can be incorporated in the description of the set like this:

$$E = \{n \in \mathbb{Z} \colon n \text{ is even}\}.$$

Cardinality The *cardinality* $|A|$ of a finite set A is the number of elements in A. In the above examples, $|P| = 3$, $|F| = 2$, and $|E|$ is not defined because E is infinite.

Relations between sets and operations on sets We say A is a *subset* of B (denoted by $A \subseteq B$) if every element of A is also an element of B. We call A a *proper subset* of B (denoted by $A \subset B$) if A is a subset of B but they are not equal. Do not confuse $\subset$, $\subseteq$, and $\in$!

The *union* $A \cup B$ of two sets A and B consists of those elements that are in A or in B:

$$A \cup B = \{x \colon x \in A \text{ or } x \in B\}.$$

The *intersection* $A \cap B$ consists of those elements that are in both A and B:

$$A \cap B = \{x \colon x \in A \text{ and } x \in B\}.$$

A and B are *disjoint* if $A \cap B = \varnothing$. The *set difference* $A - B$ consists of those elements that are in A but not in B:

$$A - B = \{x \colon x \in A \text{ and } x \notin B\}.$$

Finally, the *complement* $\overline{A}$ of a set A consists of those elements that are not in A:

$$\overline{A} = \{x \colon x \notin A\}.$$

For example, if we are talking about integers, the complement of the set of even numbers is the set of odd numbers.

I suppose that you are quite familiar with this already so I won't bore you with examples. If you need clarification, please ask.

2 Graphs

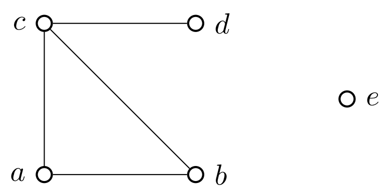
A *(simple) graph* is a pair of sets (V, E), where V is a nonempty, finite set and E is a set of 2-element subsets of V. Elements of V are called *vertices*; elements of E are called *edges*.

Notice that $\{v, v\}$ cannot be an edge because $\{v, v\} = \{v\}$ is a 1-element subset of V.

For example, $G = (V, E)$ where

$$\begin{aligned} V &= \{a, b, c, d, e\} \\ E &= \{\{a, b\}, \{b, c\}, \{a, c\}, \{c, d\}\} \end{aligned}$$

is a graph with 5 vertices and 4 edges. When the graph is small, as in this case, we can draw a diagram of it. The positions of the vertices and the shapes of the edges don't matter.



We say vertex u is *adjacent* to vertex v, or u and v are *neighbours*, if $\{u, v\}$ is an edge. The *degree* $\deg(v)$ of a vertex v is the number of vertices adjacent to it. For example in G, a and b have degree 2, c has degree 3, d has degree 1, and e has degree 0.

Lemma 1. *In every graph G, the sum of the degrees of all the vertices equals twice the number of edges.*

Indeed, G has 4 edges and the sum of its degrees is 8.

Proof. Make two copies of each edge $\{u, v\}$ and assign one copy to u and the other one to v. Then each vertex v is assigned exactly $\deg(v)$ edges, one for each of its neighbours. So the sum of the degrees equals twice the number of edges. $\square$

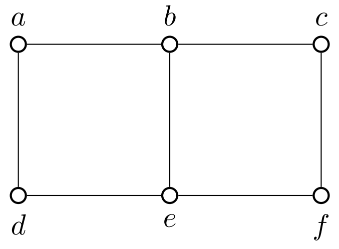
3 Bipartite graphs

The textbook has examples of some interesting surveys about the sexual habits of Americans. One of these, conducted by the U. S. National Center for Health Statistics and featured in the New York Times found out that the average number of partners of the opposite gender one had had sex with was seven for men and four for women.

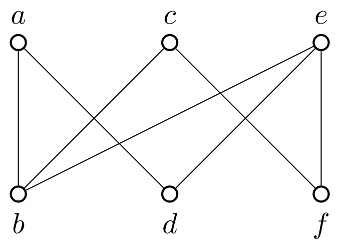
We will see that this data doesn't make sense. It cannot possibly give an accurate picture of reality.

To do this we will look at a special kind of graph called a bipartite graph. We'll need one more set-related concept: We say sets S and T *partition* set U if $S \cup T = U$ and $S \cap T = \varnothing$.

We say a graph $G = (V, E)$ is *bipartite* if the set of vertices V can be partitioned into two sets M and W so that all edges have one vertex in M and one vertex in W. For example, the graph represented by the following diagram is bipartite:



One partition is $M = \{a, c, e\}$, $W = \{b, d, f\}$. An easy way to see this graph is bipartite is to redraw it in a way that groups the vertices of M (on top) and the vertices of W (at the bottom). All edges go between M and W:



In contrast, the graph G from the previous section is not bipartite: No matter how we partition the vertices a, b, c between M and W, at most two of the edges $\{a, b\}, \{a, c\}, \{b, c\}$ will have one vertex in M and one vertex in W.

Bipartite graphs are useful for describing relations between two types of objects. For example, we can model inter-gender sexual relations in America by a bipartite graph: M is the set of all American men, W is the set of all American women, and we connect m and w ($m \in M, w \in W$) by an edge if they have had sexual relations.

The average number of sexual partners of an American man is the average degree of a vertex in M. This is the sum of the degrees of all vertices in M divided by the size of M:

$$\text{average number of sexual partners of a man} = \frac{\sum_{m \in M} \deg(m)}{|M|}$$

and similarly,

$$\text{average number of sexual partners of a woman} = \frac{\sum_{w \in W} \deg(w)}{|W|}.$$

We can relate two of these numbers using the next lemma:

Lemma 2. *Let G be a bipartite graph with partition (M, W). Then the sum of the degrees of M equals the sum of the degrees of W.*

The proof is quite similar to the one of Lemma 1.

Proof. We count the edges of G like this: First, we choose a vertex in M, then we choose a neighbour of this vertex. This counts every edge of G exactly once. On the other hand, each vertex m in M is counted $\deg(m)$ times, so the sum of the degrees of M must equal the number of edges. By the same reasoning, the sum of the degrees of W equals the number of edges. So the two sums are equal to one another. $\square$

Now, if we divide the left-hand sides and right-hand sides of the two equations (assuming the denominator is non-zero, i.e., at least one sexual relation has occurred) we get

$$\frac{\text{average number of sexual partners of a man}}{\text{average number of sexual partners of a woman}} = \frac{|W|}{|M|}.$$

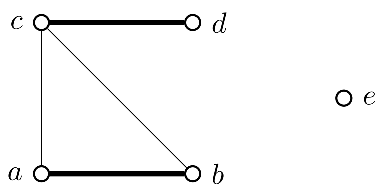
In America the gender ratio is about 51% women and 49% men, so if the sample of survey was representative, we would expect that

$$\frac{\text{average number of sexual partners of a man}}{\text{average number of sexual partners of a woman}} \approx \frac{51\%}{49\%} = 1.041\ldots$$

It is impossible for a man to have 7 partners on average and for a woman to have only 4!

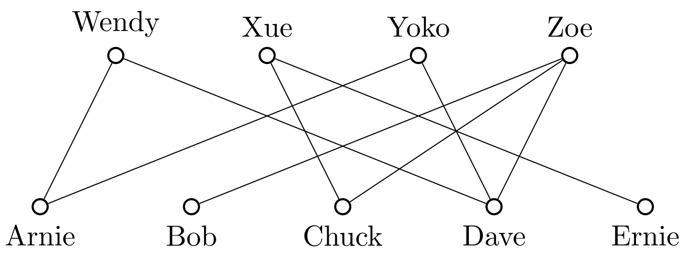
4 Bipartite matchings

A *matching* in a graph G is a subset of the edges of G no pair of which share a common vertex. For example, $\{\{a, b\}, \{c, d\}\}$ is a matching in in the following graph:

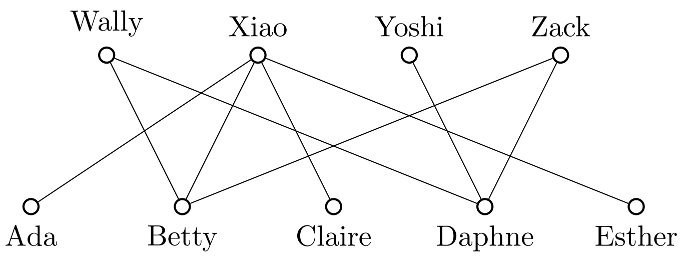


We say a vertex is *matched* in a given matching if there is an edge in the matching that contains v. A matching is *perfect* if all vertices are matched. The matching in the above example is not perfect because vertex e is not matched. For a perfect matching to possibly exist, the graph must have an even number of vertices.

Let's now look at a bipartite graph G. Here is an example:



Can we match all girls to a suitable boy? I'll let you figure this one out. Now let's look at another group. Here, the boys are in the minority.



Unfortunately this is not possible and here is why. Look at Wally, Yoshi, and Zack. They are all into Betty and Daphne and don't have interest in the other girls. We cannot match three boys to two girls.

This is a general phenomenon. To explain it we need a bit of notation. For a graph $G = (V, E)$ and a subset S of the vertices, the *neighbour set* $N(S)$ of S is the set of vertices that have at least one neighbour in S:

$$N(S) = \{v \in V \colon \{v, s\} \text{ is an edge for some } s \in S\}.$$

Theorem 3 (Hall's Theorem)**.** *Let G be a bipartite graph with vertex partition (M, W). There exists a matching that matches all vertices in M if and only if for every subset $S \subseteq M$, $|N(S)| \geq |S|$.*

So the reason why boys and girls cannot be matched is always the same: There is a subset of the boys that is interested in a smaller number of girls.

Proof. First, suppose that all vertices of M can be matched. Let S be an arbitrary subset of M. For every s in S, let v be s's partner in the matching. All such v's are distinct so there are $|S|$ of them. All are neighbors of vertices in S and therefore members of $N(S)$. It follows that $N(S)$ must have at least $|S|$ members, so $|N(S)| \geq |S|$.

Now we prove that if for every subset $S \subseteq M$, $|N(S)| \geq |S|$, then all vertices in M can be matched. The proof is by strong induction on the size of M, which we denote by n.

Base case $n = 1$: M has size 1, and so $N(M)$ has size at least 1. So the unique vertex $m \in M$ has a neighbor in W, to whom it can be matched.

Inductive step: Assume the proposition is true for all M of size 1 up to n. Let G be a bipartite graph in which M has size $n + 1$. We assume that $|N(S)| \geq |S|$ for every $S \subseteq M$ and consider two cases:

- **Case 1:** For every proper subset X of M, $|N(X)| \geq |X| + 1$. Take an arbitrary $m \in M$ and match it with an arbitrary neighbor $w \in W$. Remove s and w from the graph. Since only one vertex from W was removed, $|N(X)|$ must stay at least as large as $|X|$ for every $X \subseteq M$. By the inductive hypothesis, all vertices in $M - \{m\}$ can be matched. None of them is matched to w because w was removed, so all vertices of M are matched.
- **Case 2:** $|N(X)| = |X|$ for some proper subset X of M. All subsets X' of X satisfy $|N(X')| \geq |X'|$, so by our inductive hypothesis, all vertices in X can be matched. Remove all vertices in X and $N(X)$ from the graph. We will show that all remaining subsets of vertices $Y \subseteq M - X$ have at least $|Y|$ remaining neighbours in $W - N(X)$. By the inductive hypothesis, it will

follow that all vertices in $M - X$ can be matched to vertices in $W - N(X)$. Putting the two matchings together, we obtain one that matches all vertices in M.

It remains to show that all subsets $Y \subseteq M - X$ have at least $|Y|$ neighbours outside $N(X)$. The proof is by contradiction. Suppose there exists a subset $Y \subseteq M - X$ with fewer than $|Y|$ neighbours outside $N(X)$. Then the vertices in $X \cup Y$ must have fewer than $|N(X)| + |Y|$ neighbours. Since $|N(X)| = |X|$, it follows that

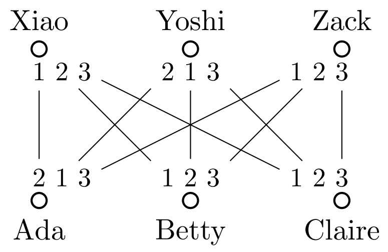
$$|N(X \cup Y)| < |X| + |Y| = |X \cup Y|$$

because X and Y are disjoint. This contradicts our assumption that $|N(S)| \geq |S|$ for every $S \subseteq M$.

It follows by induction that the proposition is true for all $m \geq 1$. $\square$

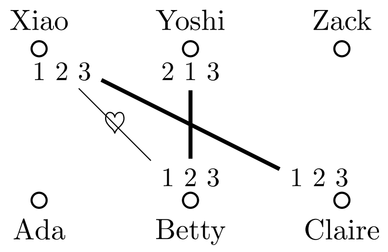
5 Stable matchings

We now have a group of n men and a group of n women. The objective is to marry them off one to one. Every man ranks all the women in order of preference, and every woman does the same for the men. We can represent this information by a *complete* bipartite graph (a bipartite graph in which all possible edges are present) with labels like in this example. Here 1 stands for "most desirable" and 3 stands for "least desirable."

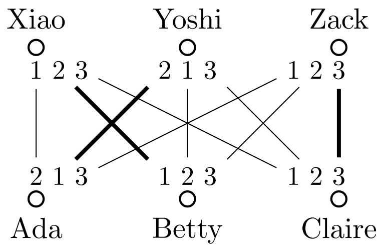


In this example, Ada is Xiao's first choice, but Xiao is Ada's second choice.

We are seeking a perfect matching between the men and the women that is *stable* with respect to their preferences. A matching is stable if it is not unstable. A matching is unstable if there exist two pairs (m, w^*), (m^*, w) so that m is matched to w^*, m^* is matched to w, m prefers w to w^*, and w prefers m to m^*. For example, the matching (Xiao, Claire), (Yoshi, Betty), (Zack, Ada) is unstable: Xiao prefers Betty to Claire, and Betty prefers Xiao to Yoshi. Xiao and Betty are a *rogue couple*: They would both rather run off with each other than stay with their partners in the matching.



In contrast, the matching (Xiao, Betty), (Yoshi, Ada), (Zack, Claire) is stable: There are no rogue couples in this matching.



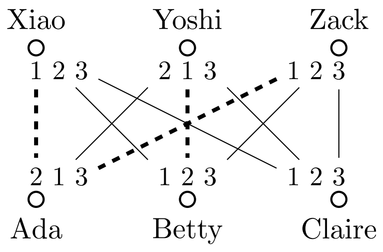
By now you must be burning with the need to know the answer to the next question:

> Given any n men, n women, and complete lists of n preferences for each person, is it always possible to match the men and women so there are no rogue couples?

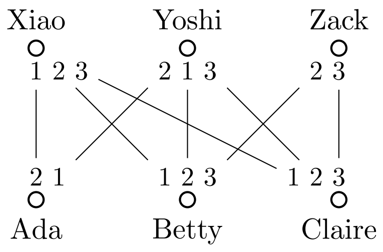
Not only is it possible, but there is a simple way to do it. The procedure is called the Gale-Shapley algorithm after its inventors.

```
Repeat the following in rounds until all women have at most one proposal:
    Each man proposes to the most desirable woman on his list that has not yet rejected him.
    Each woman rejects all proposals except her most desirable one.
```

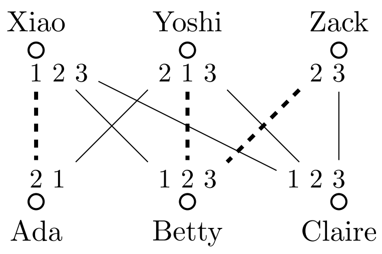
Let's see how this plays out in our example. In the first round, Xiao proposes to Ada, Yoshi proposes to Betty, and Zack also proposes to Ada.



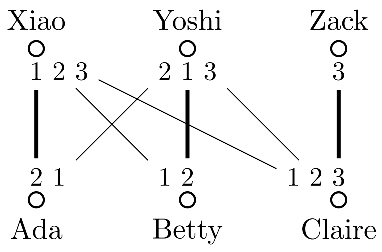
Ada has two proposals. Zack's is the less desirable one so she rejects him. Betty and Claire receive at most one proposal so they issue no rejections.



Now Xiao and Yoshi still go for their first choices, but Zack must move to his second choice Betty:



Betty now has two suitors; she rejects the lower ranked Zack. Ada and Claire issue no rejections. In the next round, Xiao proposes to Ada, Yoshi to Betty, and Zack is only left with Claire.



We obtain a stable matching: Xiao to Ada, Yoshi to Betty, and Zack to Claire. You can check that there are no rogue couples.

Theorem 4. *Given any preference lists as input the Gale-Shapley algorithm terminates with a stable matching.*

Let's first prove the following lemma:

Lemma 5. *For every woman w and every round r of proposals, if w receives at least one proposal in round r, then she receives at least one proposal in round $r+1$. Moreover, the best proposal w receives in round $r+1$ is at least as desirable to her as the best proposal she receives in round r.*

Proof. Let m be the highest ranked man on w's list among the ones that propose to w in round r. Then w was m's best available choice and m was not rejected by w in round r. Therefore m proposes to w in round $r+1$. □

Proof of Theorem 4. First we show that the algorithm terminates. Each woman can issue at most n rejections, so at most n^2 rejections can happen throughout the algorithm. In every round, at least one rejection must happen; otherwise, every woman has at most one suitor. Therefore the algorithm must terminate within at most n^2 rounds.

Next we show that the algorithm produces a matching. We argue by contradiction. Suppose it didn't. Then there must exist a man m who was not matched. The only way this can happen is that if m proposed to all women and was rejected by every single one of them. If every woman rejected m, that means every woman must have received at least one proposal. By Lemma 5, every woman must have a proposal when the algorithm terminates. Since m is not proposing, there are at most $n-1$ men giving out n proposals at termination. This is a contradiction.

We now show that the resulting matching is stable. Again, we argue by contradiction. Suppose there is a rogue couple (m, w) in the final matching. Then m must have proposed to w in some round because w is higher ranked on m's list than his final partner w^*. So m must have been rejected by w. The reason w has rejected m is because she received a proposal from someone higher ranked on her list. By Lemma 5, w's future best proposals may become only more and more desirable to her. So her final partner m^* is more desirable to her than m is. It follows that (m, w) is not a rogue couple, contradicting our assumption. □

Is it better to be a man or better to be a woman in this algorithm? It looks like women may be better off as they do all the choosing. In fact, the opposite is true: It is the men who end up with their best possible choices!

Theorem 6. *For every man m and woman w, if w ever rejects m, then (m, w) are not part of any stable matching.*

If a man was ever served a rejection by a woman, there is not even a one in a million chance that the two might be a stable couple!

Proof. We prove the theorem by strong induction on the round r in which w rejects m.

Base case $r = 1$: If w rejects m in the first round, it means she had at least two proposals from m and m^*, both of whom rank w first on their list. So any matching in which m is matched to w and m^* is matched to some other woman w^* cannot be stable as (m^*, w) would be a rogue couple: w is m^*'s first choice and w likes m^* better than m.

Inductive step: Assume that every man-woman pair for which a rejection was served in rounds 1 up to r are not part of any stable matching. Now suppose w rejects m in round $r + 1$. Assume, for contradiction, that there exists a stable matching Ξ in which m is matched to w. Since w rejected m, she must have been proposed to by some m^* that is higher on her list than m. By the inductive hypothesis, Ξ cannot match m^* to any of the women he proposed to up to round r because they all rejected him and so Ξ wouldn't be stable. Therefore Ξ must match m^* to a woman w^* that is less desirable to him than w. But then (m^*, w) are a rogue couple in Ξ, contradicting our assumption that Ξ is stable. □

6 Greedy algorithms*

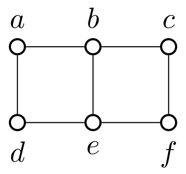
Graphs are a source of many interesting problems in computer science. One example is the question of finding a small vertex cover. We will say that vertex v *covers* edge e if $v \in e$. A *vertex cover* of a graph is a subset of vertices that covers all the edges.

You may imagine that the edges and vertices represents links and routers on the internet in a certain country. The government may like to control all traffic on the network by co-opting a some set of routers. One option would be to take over all routers but this may be expensive. This is a contrived example in which it may be sensible to look for a small vertex cover of a graph.

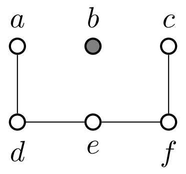
Here is one possible approach for this type of problem: We start with an empty cover C and add vertices to it one by one; at each stage, we choose to include in C the vertex that is "most profitable" by some natural measure. In the example of vertex cover, the objective is to cover all the edges, so it seems reasonable to pick the vertex that by itself will cover most of the remaining edges, namely one with the largest possible degree.

```
Algorithm $A$: Given a graph $G$ as input,
    Set $C = \varnothing$.
    While $G$ has any edges left,
        Choose a vertex $u$ of maximal degree in $G$.
        Add $u$ to $C$.
        Remove $u$ and all its adjacent edges from $G$.
    Output $C$.
```

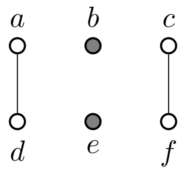
This is an example of a *greedy algorithm*: At each stage, the algorithm makes a choice that is "locally" optimal, without worrying about the "global" consequences of its decision. It should be clear that the algorithm always terminates and outputs a vertex cover of G.[1] Let us see how it might perform on the following example:



The vertices of maximal degree in this graph are b and e. Let's say it includes b in the cover and is left with the following state after the first iteration. The cover consists of the shaded vertices.



There are now three vertices with maximal degree; we did not specify how the algorithm should choose among them. If it happens to select e at the next iteration, the state becomes:



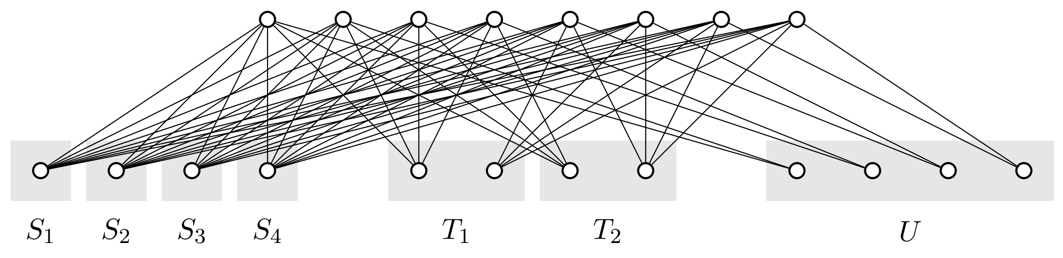
No matter what happens in the following two steps, the algorithm will calculate a vertex cover C of size 4; for example, one possible output is $C = \{a, b, c, e\}$. This is clearly superior to picking all the vertices, but is not the best possible vertex cover in terms of size. This graph has a vertex cover of size 3, for example $\{a, c, e\}$.

The task of calculating the smallest possible vertex cover for an arbitrary graph is notoriously difficult.[2] Short of finding the optimum solution, it would be interesting how the solution found by algorithm A compares to the best possible one.

The following type of graph turns out to be particularly challenging for algorithm A in this respect.

[1]Formally, you can prove termination by induction on the number of vertices, and correctness using an invariant. In this example the predicate "the vertices in C cover all edges in the input graph that are not included in G" is the relevant invariant.

[2]Finding the smallest vertex cover is an example of an NP-hard search problem, a powerful concept that you will learn about later in your studies.

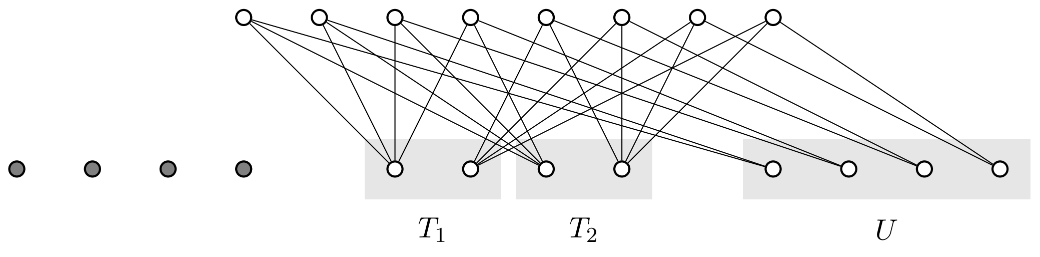


This is a graph with 8 top vertices and 12 bottom vertices, which are partitioned into 7 sets S_1, S_2, S_3, S_4, T_1, T_2, and U. The degree of each vertex depends on the type of set it belongs to as follows:

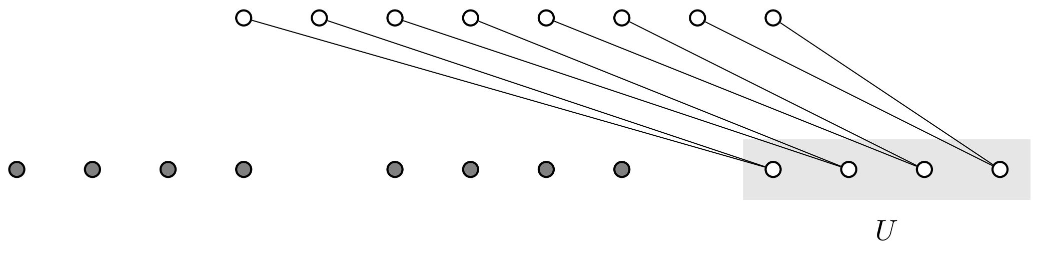
set	degree
top	7
S_1, S_2, S_3, S_4	8
T_1, T_2	4
U	2

Notice that each top vertex has exactly one neighbour in each of the 7 sets $S_1, S_2, S_3, S_4, T_1, T_2, U$.

The vertices in S_1, S_2, S_3, and S_4 are initially those of highest degree, so algorithm A includes all of them in the cover C in its first four steps. At this stage, the state of the algorithm looks like this:



At this stage the degree of the top vertices has dropped to 3, so the next batch of vertices to be included in the cover by algorithm A will be those in T_1 and T_2, resulting in the following state:



The vertices in U have now the highest degrees, so algorithm A will also include those in the cover C. The vertex cover C produced by algorithm A will therefore consist of the bottom 12 vertices. The graph, however, has a vertex cover of size 8 consisting of all the top vertices. To summarize, in this example algorithm A outputs a vertex cover that is (at least) 1.5 times larger than the smallest one. You can generalize this example to prove the following theorem:

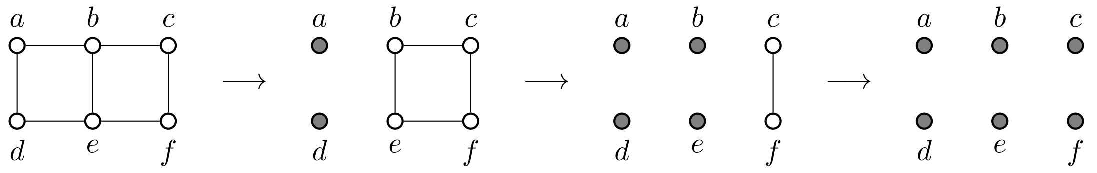
Theorem 7. *For every real number $K > 0$ there exists a graph G such that on input G, algorithm A outputs a vertex cover that is at least K times larger than the smallest possible one.*

In less formal language, if we measure the performance of an algorithm as the ratio between the size of the solution it finds and the size of the best possible solution then there are input graphs on which algorithm A is arbitrarily poor.

Let us now consider a different algorithm for finding a vertex cover. This algorithm greedily picks out uncovered edges e in the graph and includes *both* vertices of e in the cover.

```
Algorithm $B$: Given a graph $G$ as input,
1    Set $C = \varnothing$.
2    While $G$ has any edges left,
3        Choose any edge $\{u, v\}$ of $G$.
4        Add $u$ and $v$ to $C$.
5        Remove $u$, $v$ and all their adjacent edges from $G$.
6    Output $C$.
```

On our first example, one possibility for algorithm B is to execute the following steps



and output the cover $C = \{a, b, c, d, e, f\}$ comprising all the vertices. It appears that algorithm B performs worse than algorithm A, which produces a vertex cover of size 4 on the same input. The smallest vertex cover has size 3. However, this is the worst possible type of input for algorithm B. Algorithm B will never produce a vertex cover that is more than twice the size of the smallest possible one.

Theorem 8. *On every input G, algorithm B outputs a vertex cover that is at most twice the size of the smallest vertex cover in G.*

In the proof, we will use G to denote the graph given to B as input (and not the graphs derived in intermediate stages of the algorithm).

Proof. Let Ξ be the set of edges chosen by algorithm B in step 3 throughout its execution. Then Ξ must be a matching of G: After including an edge e in Ξ all other edges that intersect e are removed, so no two edges in Ξ may intersect.

Any vertex cover of G must in particular cover all the edges of the matching Ξ. It must in particular include one vertex of every edge of Ξ. Therefore any vertex cover of G, including the smallest one, must have size at least $|\Xi|$. On the other hand, the output C includes all the vertices in Ξ, so $|C| = 2|\Xi|$. It follows that C is at most twice the size of the smallest possible matching of G. □

Compare Theorem 8 with Theorem 7: The first one says that algorithm B always outputs a vertex cover which is within twice the size of the smallest possible one, while for algorithm A the corresponding ratio may be unbounded. We may therefore say that in the worst case, algorithm B has a better performance guarantee than algorithm A. On specific inputs, however, algorithm A may do better.

References

This lecture is based on Chapter 12 of the text *Mathematics for Computer Science* by E. Lehman, T. Leighton, and A. Meyer and lecture notes on stable matchings by Prof. Lap Chi Lau. Section 6 is based on Chapter 1 of the book *Approximation Algorithms* by Vijay Vazirani.