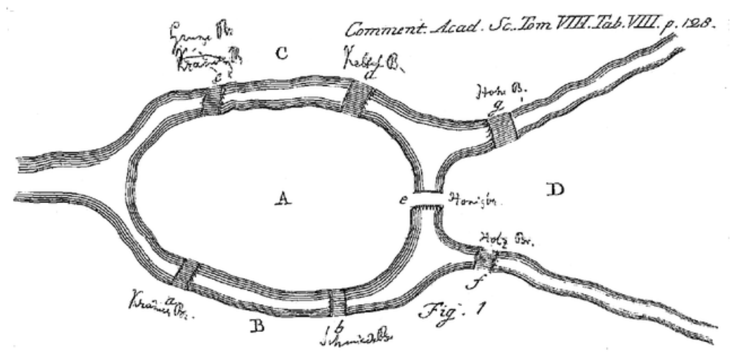
Grade 6 Math Circles
February 14/15/16, 2023
Graph Theory

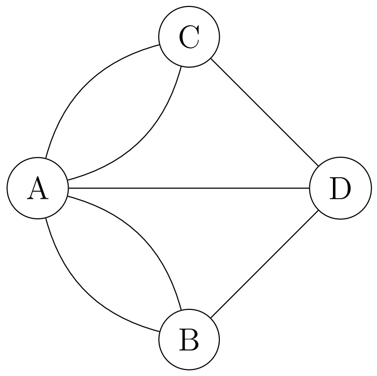
The Seven Bridges of Königsberg



Source: Wikimedia Commons

In 1735, the mayor of a nearby town named Danzig in Prussia wanted to know whether someone could cross all the bridges of Königsberg exactly one time (Hopkins and Wilson, 2004) (1). He proposed this problem to Leonard Euler (now widely known just as Euler), who initially thought this problem was unrelated to math. But as he worked on the problem, he realized that it was related to math, just in a completely new way. So in the process of solving this problem, Euler invented what we now know as graph theory.

This idea of visiting each bridge (or edge as defined later) exactly once became known as an Eulerian path. Below is a simplified version of the problem. Take a few minutes to convince yourself of this, and see if you can find a solution by drawing along each line exactly once without lifting your pencil.

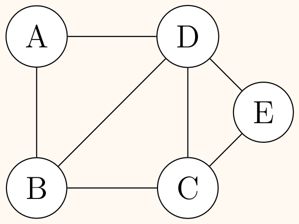


If you were stressed about finding an answer, don't be! There's actually no way to cross all seven bridges exactly once. This is what Euler concluded after looking into this problem.

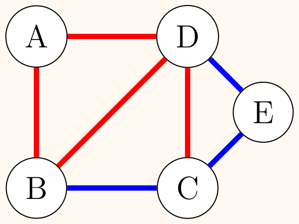
Definitions

Vertex (Vertices pl.): a point, usually represented as a dot or circle. *(Notation: $V = \{...\}$)*
Edge: a connection between exactly two vertices, usually represented as a line. *(Notation: $E = \{...\}$)*
Adjacent: two vertices are adjacent if they are connected by an edge.
Graph: a set of vertices and edges. *(Notation: $G = (V, E)$)*
Walk: an alternating set sequence of vertices and edges that are connected to each other.
Path: a walk that does not visit any vertices more than once.
Cycle: a walk that does not visit any vertex more than once and starts and ends on the same vertex.

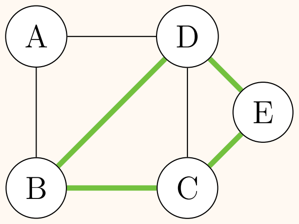
Example A



The vertices are:
$V = \{A, B, C, D, E\}$.
The edges are:
$E = \{AB, AD, BC, BD, CD, CE, DE\}$.
A and D are adjacent, and A and E are not adjacent.



The sequence of edges DA, AB, BD, DC (in red) is a walk from D to C. The sequence of edges BC, CE, DE (in blue) is a path from B to D



The set of edges $\{BC, CE, DE, BD\}$ form a cycle.

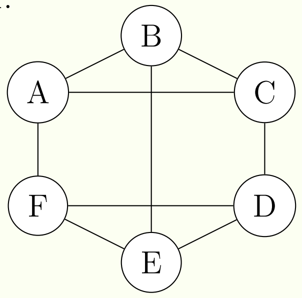
Some things to note:

- Every path is a walk, but not every walk is a path.
- A path and a cycle are both walks, but a path is not a cycle (and vice versa).
- We can also represent the walk from D to C by listing the sequence of vertices. (ie. D, A, B, D, C)
- AB and BA represent the same edge.
- For the purpose of this lesson, we will not consider graphs with multiple edges between the same pair of vertices.

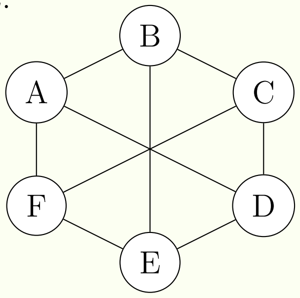
Exercise 1

Match the sets of vertices and edges to the correct visual picture of the graph.

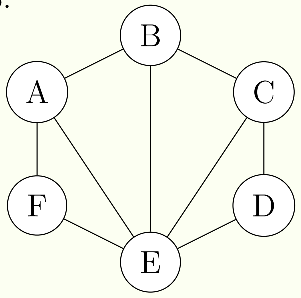
1.



2.



3.



(a) $V = \{A, B, C, D, E, F\}$
$E = \{AB, BC, CD, DE, EF, AF, AD, CF, BE\}$

(b) $V = \{A, B, C, D, E, F\}$
$E = \{AB, AE, BC, CD, DE, EC, EF, AF, BE\}$

(c) $V = \{A, B, C, D, E, F\}$
$E = \{AB, BC, CD, DE, CA, DF, EF, AF, BE\}$

Solution

1. (c) 2. (a) 3. (b)

Matchings and Covers

Definitions

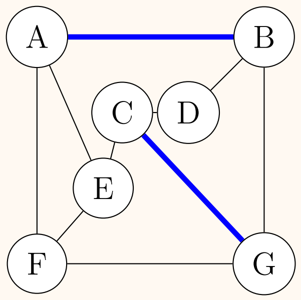
Matching: a group of edges where no two edges in the group share an endpoint. We denote a matching by writing $\mathcal{M} = \{\ldots\text{list of edges}\ldots\}$. A matching is perfect if every vertex is the endpoint to an edge in the matching.

(Vertex) Cover: a group of vertices where every edge in the graph has at least one end point in the cover. We denote a cover by writing $\mathcal{C} = \{\ldots\text{list of vertices}\ldots\}$.

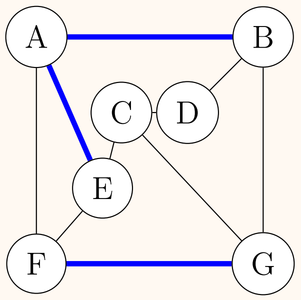
Size of matching/cover: the number of edges or vertices in the group.

Example B

Valid and invalid matchings

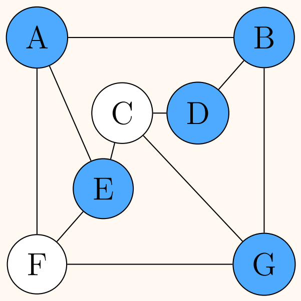


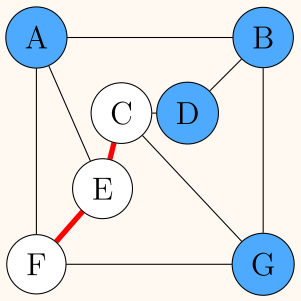
The blue edges gives a valid matching because the two blue edges do not share any vertices. The edges in this matching are
$\mathcal{M} = \{AB, CG\}$.



The blue edges give an *invalid* matching because edges AB and AE share vertex E.

Valid and invalid covers

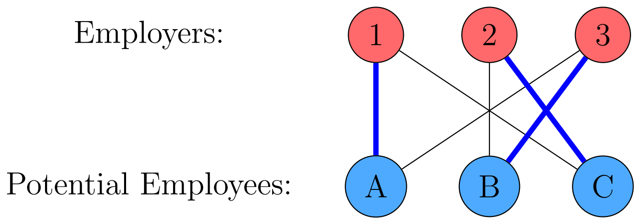




The blue vertices give a valid cover because every edge in the graph has at least one vertex in the cover.
The vertices in the cover are $\mathcal{C} = \{A, B, D, E, G\}$.

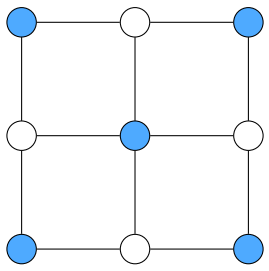
The blue nvertices give an *invalid* cover because edges CE and EF do not have any of their endpoints in the cover.

When are matchings and covers useful in real life? An application for a matching is matching potential employees to an employer. If the blue vertices represent the potential employees and the red vertices represent the employers, a graph for this might look like the following.



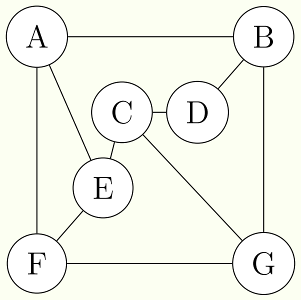
This graph shows that employer 1 matched with employee A, employer 2 matched with employee B, and employer 3 matched with employee C.

An example of a real life application for a vertex cover is placing security cameras at intersections of roads. If the blue vertices represent the security cameras and the edges represent roads, a graph for this might look for the following:



This graph shows that these five cameras placed in this configuration will be able to see every road on the map.

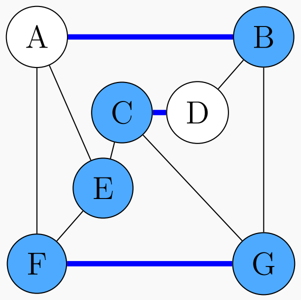
Exercise 2



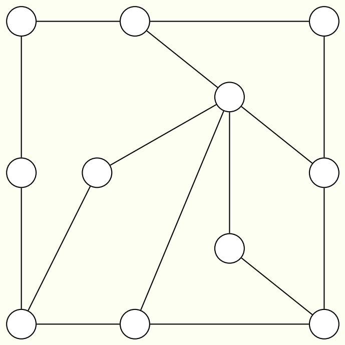
1. Find a different valid matching for the graph above. Show this in a drawing and in proper notation.
2. Find a different valid cover for the graph above. Show this in a drawing and in proper notation.

Solution

Solutions may vary, but a possible solution is $\mathcal{M} = \{AB, CD, FG\}$ and the vertices in the cover are $\{B, C, E, F, G\}$.



Exercise 3

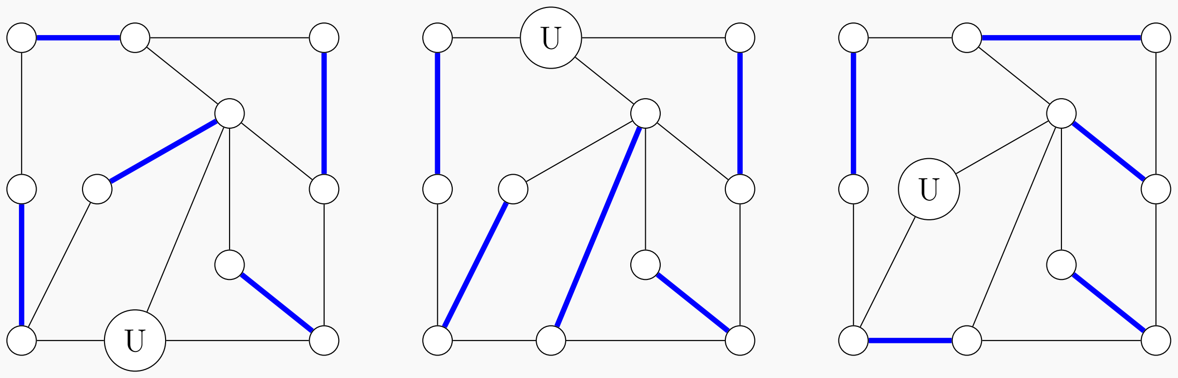


1. A maximum matching is a matching with the largest possible size for a given graph. What is the size of the maximum matching of graph above?
2. A minimum cover is a cover with the smallest possible size for a given graph. What is the size of the minimum cover of the graph above?

We will look at these two ideas later when we talk about bipartite graphs.

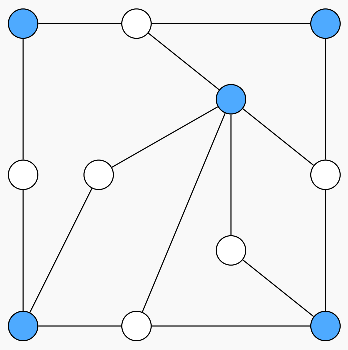
Solution

1. The size of the maximum matching is 5. We will explain why this is the case in Example E. Below are a few possible maximum matchings (in blue). Notice that no more edges can be added to the matching, because to do so would cause two edges to share an endpoint. The only unmatched vertex, U, is highlighted.



2. The size of the minimum cover is 5. We will also explain why this is the case with a theorem. There is actually only one valid minimum cover for this graph. This is not always the case, but notice that since there are 5 edges connected to the blue vertex in

the middle, it makes sense that that vertex would probably need to be in the cover to minimize the number of vertices.



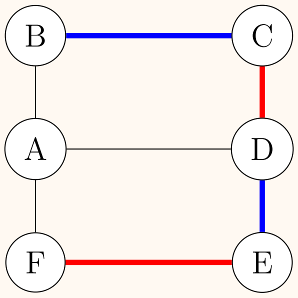
Alternating and Augmenting Paths

We tried to find a maximum matching in the Exercise 3. Was it easy? Probably not! Finding a maximum matching is difficult work, especially when a graph gets very complicated. Augmenting paths will help us to find maximum matchings.

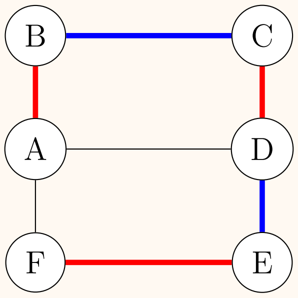
Definitions

Alternating path: a path that has edges alternating in and out of a matching. (A singular edge is always an alternating path.)
Augmenting path: an alternating path that starts and ends on an unmatched vertex. (An unmatched vertex is a vertex that is not the end point of an edge in the matching.)

Example C
Alternating and augmenting paths
Let the blue edges be in the matching.



The highlighted path from B to F is an alternating path but not an augmenting path. Note that the path from B to E is also a valid alternating path but not an augmenting path.



The highlighted path from A to F is an augmenting path because it starts on an unmatched vertex A and ends on an unmatched vertex F and has edges alternating in and out of the matching.

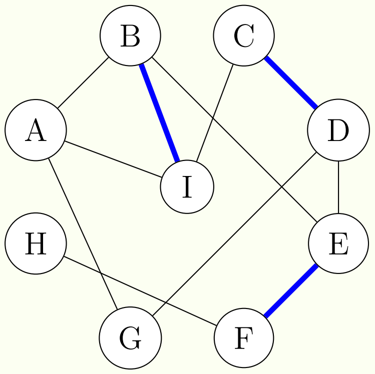
Note that the single edge AF is also an augmenting path since it also starts and ends on an unmatched vertex and is an alternating path by definition.

In Exercise 3 question 1, we looked at the idea of a maximum matching. It's very difficult to guess and check whether you have a maximum matching, especially when a graph gets more and more complicated. A useful theorem to relate maximum matchings and augmenting paths is the following:

Theorem (Berge's Theorem). *A matching is a maximum matching if and only if there is no augmenting path in the graph.*

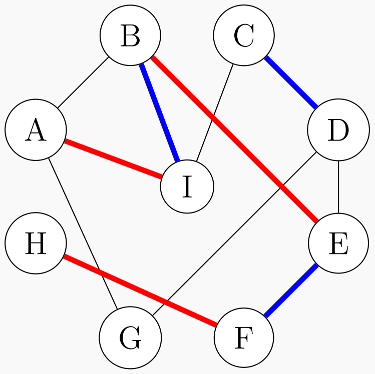
Exercise 4

Given the following graph and matching (in blue), show that the matching is not maximal by finding an augmenting path.



Solution

The beginning of an augmenting path must start with an unmatched vertex. Since EF is in the matching, we can start our augmenting path at H. So we have the following augmenting path highlighted in red and blue:



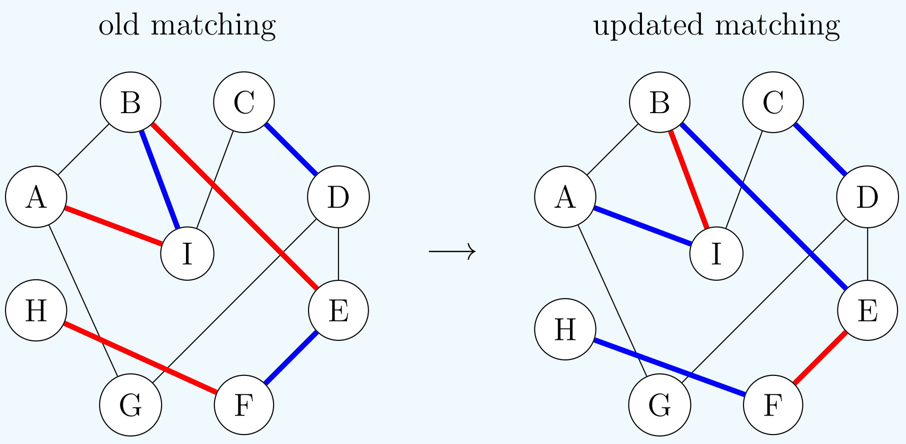
Since we have an augmenting path, we can apply Berge's Theorem to conclude that the given matching is NOT maximal.

Notice that the sequence HF,FE,ED,DC,CI,IB,BA is also a valid augmenting path.

Stop and Think

How might we make the matching in Exercise 4 larger?

We can make the matching larger by simply swapping the edges in the matching with the edges outside of the matching within the path! If take the edges in the matching to be the red edges from the augmenting path and any other blue edges, we will increase our matching by a size of one.



In our updated matching, we see that our matching is now size 4, and the red edges are the edges that were previously in the matching. We will look at an algorithm to find larger matchings later.

Bipartite Graphs and Matching Algorithms

Definition

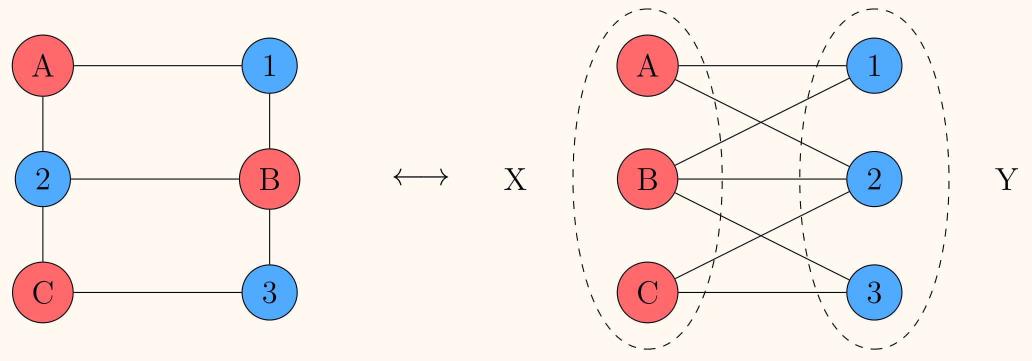
Partition: a separation of the vertices into two groups, where each vertex is in exactly one group.

Bipartite graph: a graph where the vertices can be split into a partition where each vertex is not adjacent to any other vertices in its group.

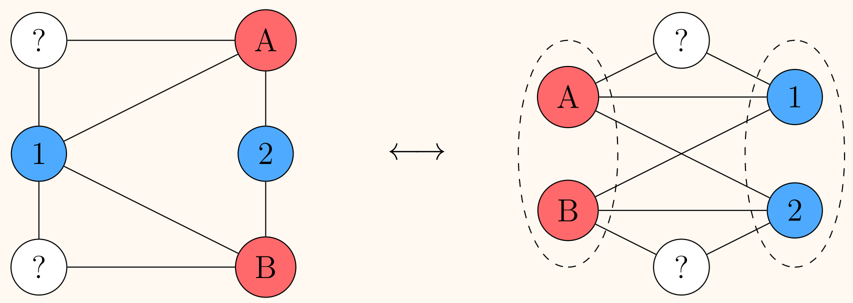
Example D

Bipartite and non-bipartite graphs

(The two colours can be taken as the two groups)



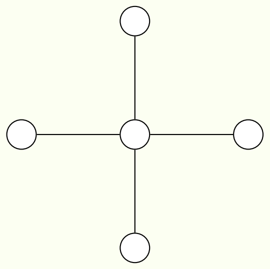
This graph is bipartite because no blue vertices are adjacent to any other blue vertices, and no red vertices are adjacent to any other red vertices. We can separate the vertices into two groups: $X = \{A, B, C\}$ and $Y = \{1, 2, 3\}$. The two graphs are equivalent, and the graph on the right shows our two partitions clearly.

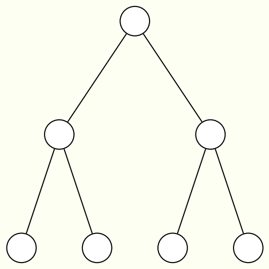


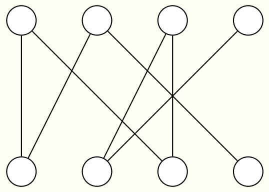
This graph is non-bipartite because we cannot successfully categorize the two white vertices into either the red or blue partition without putting them adjacent to a red or blue vertex. The graph on the right shows this clearly.

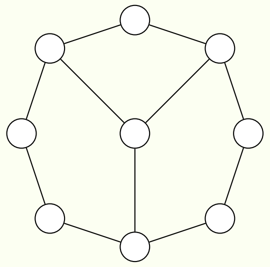
Exercise 5

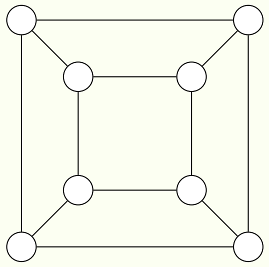
Classify the following as a bipartite or non-bipartite graph.

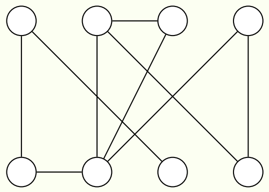
1.


2.


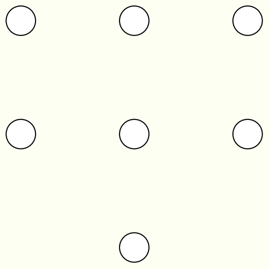
3.


4.


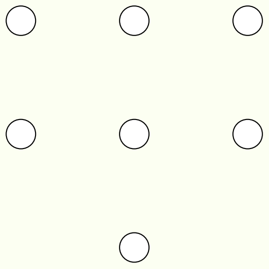
5.


6.


Draw edges between the vertices to make a bipartite and non-bipartite graph.



bipartite



non-bipartite

Solution

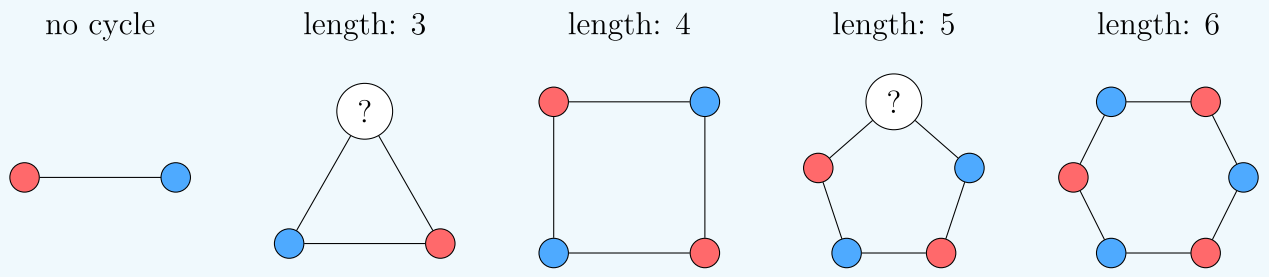
1. bipartite 2. bipartite 3. bipartite 4. non-bipartite 5. bipartite 6. non-bipartite

Answers may vary for the drawing.

Stop and Think

Is there another way to characterize bipartite graphs? Think about the length of a cycle in a bipartite graph. *(The length of a cycle is the number of edges in the cycle.)*

Notice in Example D that the cycles in the bipartite graph are even in length and that the non-bipartite graph has an even length cycle and two odd length cycles. We can reason generally that a bipartite graph must have only cycles of even length (or that non-bipartite graphs must have odd length cycles). Below are some examples of cycle lengths and attempts at splitting the vertices into two differently coloured partitions.



Verify this characterization with your answers in Exercise 5.

Why do we care?

You might be wondering why we care about bipartite graphs. It turns out that there's a neat algorithm (set of clearly defined steps) to find augmenting paths in bipartite graphs. As discussed previously, augmenting paths are useful to increase the size of our matching. We can use the Hopcroft-Karp algorithm to guarantee a maximum matching of a bipartite graph. Essentially, the idea is:

1. Starting on an unmatched vertex
2. Finding an augmenting path
3. Swap the edges (as we have shown before) to create a bigger matching
4. Repeat

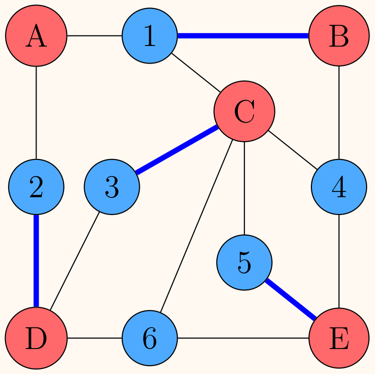
Let's do an example of this algorithm before we look at the instructions.

Note: We call each time we repeat the steps of an algorithm an *iteration*. For example, if we are going through the algorithm's steps the third time, we would be on the third iteration or iteration number 3.

Example E

Let's use the same graph as we did in Exercise 3. The vertices are separated into two different partitions (red letters and blue numbers), $X = \{A, B, C, D, E\}$ and $Y = \{1, 2, 3, 4, 5, 6\}$. You can verify that this graph is bipartite by checking to see that every red vertex is only adjacent to a blue vertex, and every blue vertex is only adjacent to a red vertex.

Suppose that the blue edges are in the matching. Find a maximum matching of the graph below by using the Hopcroft-Karp Algorithm.



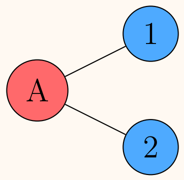
Solution

The first thing to do is to choose one of the partitions to start from. For this example, we will choose partition X (the red letters) as our starting point.

<u>Iteration 1:</u>

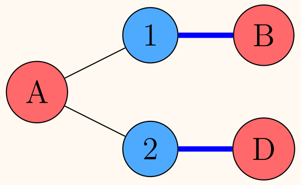
Step 1: We need to find the unmatched vertices from this partition. The only unmatched vertex in X is A, so we draw this vertex and proceed with attempting to find an augmenting path.

Step 1(a): We draw all the edges from A to any adjacent vertices, which are 1 and 2.



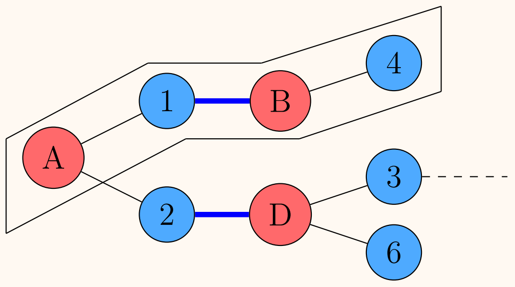
Step 1(b): Then we look for any augmenting paths. Currently, there are no augmenting paths, because 1 and 2 are both matched vertices.

Step 1(c): Then, from our vertices that we added in Step 1(a) (vertices 1 and 2), we draw all the edges to any **matched** vertices. Vertices 1 and 2 are only matched to B and D respectively. We want to connect to a matched vertex because we want to guarantee that a matched edge is next to get edges alternating in and out of the matching.



We repeat this process until we find an augmenting path or run out of vertices to add to the drawing.

Repeat of Step 1(a): So from B and D, we draw the edges to any adjacent vertices. We see that the vertex adjacent B is just 4, and the adjacent vertices from D are 3 and 6.

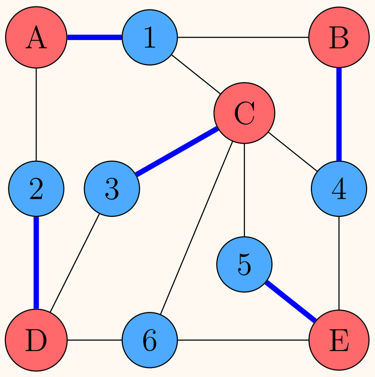


Note: We call this drawing an alternating tree.

Repeat of Step 1(b): Notice that since the vertex 4 is unmatched, the outlined path is an augmenting path. So we go immediately to Step 2. (Note that since the vertex 6 is also unmatched, $\{A2, 2D, D6\}$ is also a valid augmenting path we can use.)

Step 2: Since we have an augmenting path, we want to increase our matching by adding the black edges in the augmenting path ($A1$ and $B4$) to the matching and removing the blue edge in the augmenting path ($1B$) from the matching. This makes our matching increase by a size of 1 since we add 2 edges and remove 1 edge.

This gives us the following updated graph:



Since there are no more unmatched vertices in X (red vertices), we are done. We find a maximum matching $\mathcal{M} = \{A1, B4, C3, D2, E5\}$.

Recall from Exercise 3 that the size of the maximum matching was 5. This aligns with the final answer from our algorithm.

Hopcroft-Karp Algorithm

Input: A bipartite graph with vertices partitioned into groups X and Y and a valid matching.
Output: A maximum matching for the given graph.

Before starting, choose one of the two groups of vertices.

1. Find an augmenting path for the matching. Identify all the unmatched vertices in the graph from your chosen group and draw all of these vertices. If there are no more unmatched vertices from your chosen partition, then *STOP* the algorithm.
   (a) From your most recent vertex, draw edges to any adjacent vertices. Make sure that you do not draw a vertex that you've drawn before and that you do not draw any cycles.
   (b) If there is an augmenting path, then *STOP* and proceed to step 2.
   (c) Draw the connecting edge from your vertices in part (b) to any **matched** vertices. Again, make sure no vertices are repeated and no cycles are drawn.
   (d) Repeat steps (b) and (c). If there are no more vertices can be added to the drawing and there is no augmenting path, then *STOP* the algorithm.
2. Swap the edges in the matching with the edges NOT in the matching within the augmenting path.
3. Go to step 1.

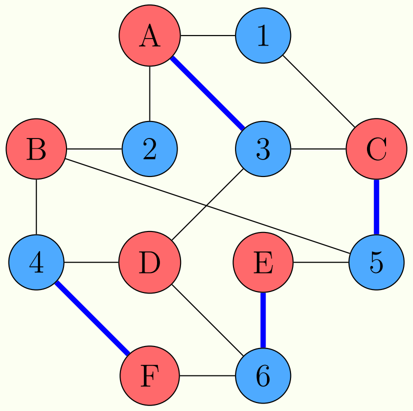
Stop and Think

Why does this work so well? What is so special about augmenting paths in bipartite graphs? To answer this question, think about the starting vertex and the ending vertex.

Consider an augmenting path in a bipartite graph. Every augmenting path must have an even number of vertices, since you have to start with an unmatched vertex and end on an unmatched vertex. Every vertex that is not the beginning or end point of the path must be a matched vertex that is the end point of exactly one matched edge. Since the path has an even number of vertices, the augmenting path must start in X and end in Y (or the other way around). Then because the augmenting path starts on a vertex in X and ends on a vertex in Y, we guarantee that we cannot end on the same vertex that began on.

Exercise 6

Suppose that the blue edges are in the matching. Find a maximum matching of the graph below by using the Hopcroft-Karp Algorithm. The two partitions are $X = \{A, B, C, D, E, F\}$ and $Y = \{1, 2, 3, 4, 5, 6\}$.

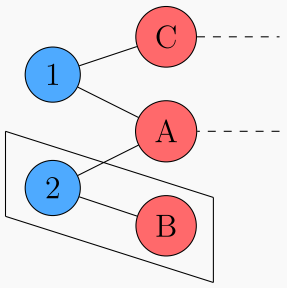


Solution

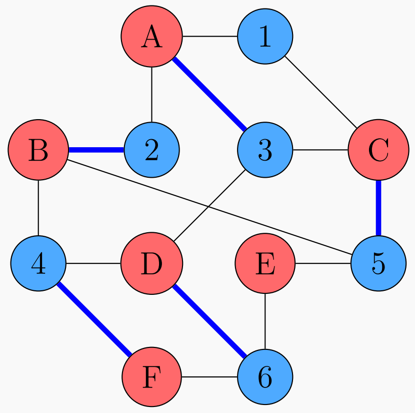
We first start by choosing a partition. Let us choose Y (the blue numbers) as our starting partition.

Iteration 1:

The unmatched vertices in Y are 1 and 2. So we draw the following alternating tree.

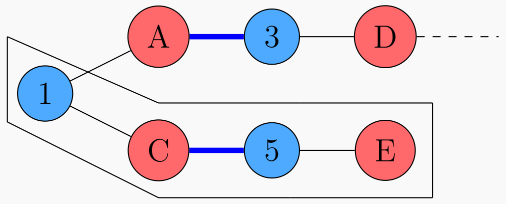


Since A and C are both matched vertices and B is unmatched, we only have one possible augmenting path, $B2$. Since there were no matched edges in our augmenting path, swapping the edges in this path only results in adding $B2$ to our matching.

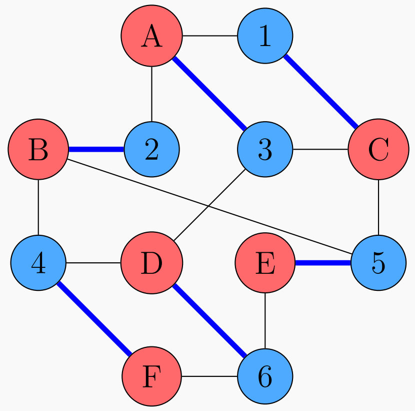


Iteration 2:

The only unmatched vertex in Y is 1. So we draw the following alternating tree.



This gives us the highlighted augmenting path. Swapping the edges in this path then us the following updated graph:



Since we don't have any more unmatched vertices in Y, we stop here. We find a maximum matching $\mathcal{M} = \{A3, B2, C1, D6, E5, F4\}$.

König's Theorem and Minimum Covers

Theorem (König's Theorem)**.** *For any bipartite graph, the number of edges in a maximum matching equals the number of vertices in a minimum vertex cover.*

Stop and Think

How does König's Theorem apply to what you've learned about bipartite graphs and maximum matchings?

Since we can guarantee a maximum matching in a bipartite graph by the Hopcroft-Karp Algorithm, we can then use König's Theorem to also guarantee a minimum vertex cover. But how do we do this?

To find a minimum cover from a maximum matching, we need to make sure to keep track of a few things. Suppose we already have the two partitions of our bipartite graph, X and Y, and a maximum matching. Let us define two more groups of vertices:

- U: The group of unmatched vertices in X. Note that it is possible for there to be no vertices in this group. In this case, we say that U is empty. *(Notation: $U = \emptyset$).*
- Z: All the vertices in U plus any vertex that is connected to the vertices in U with an alternating path.

Minimum Cover Rule

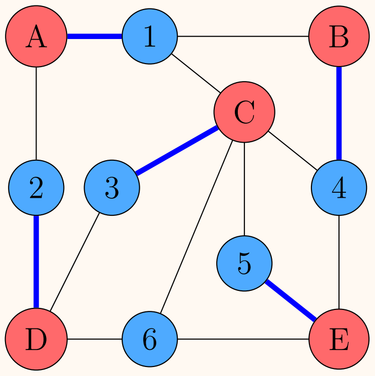
The rule for finding a minimum cover is this:

$$\text{minimum cover} = (\text{vertices in } X \text{ but not in } Z) \text{ and } (\text{vertices in both } Y \text{ and } Z)$$

Let's use our graph from Example E to demonstrate how to find a minimum cover.

Example F

From Example E, we have the following bipartite graph and maximum matching. The two partitions are $X = \{A, B, C, D, E\}$ and $Y = \{1, 2, 3, 4, 5, 6\}$.



Use König's Theorem and the Minimum Cover Rule to find a minimum cover.

Solution

Since this graph is bipartite and we are given a maximum matching, König's Theorem tells us that there is a minimum cover that is the same size as our maximum matching.

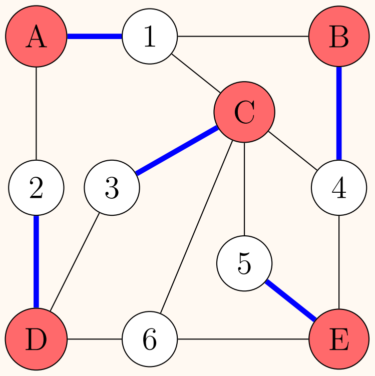
We start by building U and Z. Since X does not have any unmatched vertices, U is empty. Then, since U is empty, Z cannot contain any vertices in U or any vertices that are connected to the vertices in U.

This gives $U =$∅and $Z =$∅.

Since Z is empty, the vertices that are in X but not in Z are just the vertices in X. Also since Z is empty, there are no vertices that are in both Y and Z. So:

$$\begin{aligned}\text{minimum cover} =& (\text{vertices in } X \text{ but not in } Z) \text{ and } (\text{vertices in both } Y \text{ and } Z)\\ =& (\text{vertices in } X) \text{ and } (\text{no vertices})\\ =& \{A, B, C, D, E\} \text{ and } \emptyset\\ =& \{A, B, C, D, E\}\end{aligned}$$

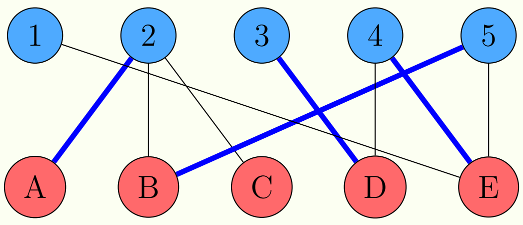
So we have the following minimum cover $\mathcal{C} = \{A, B, C, D, E\}$.



Recall from Exercise 3 that the size of the minimum cover was 5. This aligns with our Minimum Cover Rule and the size of our maximum matching as well by König's Theorem. Note as well that every vertex in the cover is the endpoint of exactly one edge in the matching.

Exercise 7

Given the following bipartite graph, with partitions $X = \{A, B, C, D, E\}$ and $Y = \{1, 2, 3, 4, 5\}$, and the following maximum matching (in blue), use König's Theorem and the Minimum Cover Rule to find a minimum cover.



Solution

Since this graph is bipartite and we are given a maximum matching, König's Theorem tells us that there is a minimum cover that is the same size as our maximum matching.

We start by building U and Z. In X, only vertex C is unmatched. So $U = \{C\}$. Then we consider the alternating paths from C, which is only the sequence $\{C, 2, A\}$. So $Z = \{A, C, 2\}$. This means that:

$$\text{vertices in } X \text{ but not in } Z = \{B, D, E\}$$

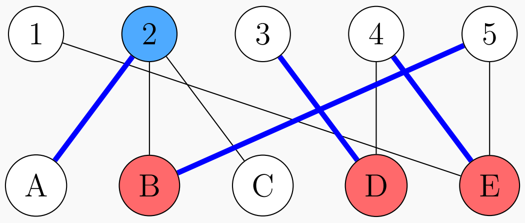
and

$$\text{vertices in both } Y \text{ and } Z = \{2\}$$

So our minimum cover by the Minimum Cover Rule is:

$$\begin{aligned}\text{minimum cover} &= (\text{vertices in } X \text{ but not in } Z) \text{ and } (\text{vertices in both } Y \text{ and } Z)\\ &= \{B, D, E\} \text{ and } \{2\}\\ &= \{B, D, E, 2\}\end{aligned}$$

The highlighted vertices shows the minimum cover $\mathcal{C} = \{B, D, E, 2\}$



Note again that the size of our minimum cover matches the size of our maximum matching.

References

[1] Brian Hopkins & Robin J. Wilson (2004) The Truth about Königsberg, The College Mathematics Journal, 35:3, 198-207, DOI: 10.1080/07468342.2004.11922073