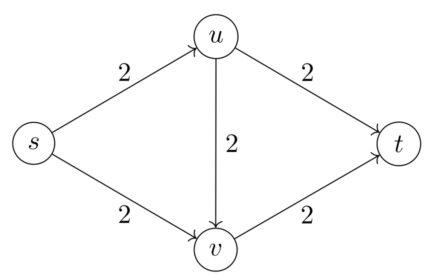
COMP3121/9101
Algorithm Design

Tutorial 5
Flow Networks

Before the Tutorial

Before coming to the tutorial, try and answer these discussion questions yourself to get a firm understanding of how you're pacing in the course. No solutions to these discussion points will be officially released, but feel free to discuss your thoughts on the forum. Your tutor will give you some comments on your understanding.

- Briefly explain the concept of an *augmenting path*. Given such an augmenting path in the flow network, how do you construct the corresponding residual graph?
  - To aid your explanation, it might be helpful to provide an example of an augmenting path and how the residual graph is then constructed.
- We look specifically at two algorithms to solve the maximum flow problem. How are they similar and how do they differ?
  - To understand this best, it might be helpful to discuss when their time complexities match and when their time complexities differ.
- Briefly define what a cut is. Using the definition of a cut, how do we define a *minimum cut*?
- Consider the following s-t flow network G.



  - What is the maximum flow of G?
  - What are the s-t cuts of the flow network?
  - Which one(s) form a *minimum* cut? What is the value of the minimum cut, and how can you tell that it is a minimum cut?
- Read through the problem prompts and think about how you would approach the problems.

Tutorial Problems

Problem 1. You are the mayor of your local town. In this town, there are m schools and n students requiring a school. Each student has a proximity range of schools that they can apply to and, since you value education, you require that every student is assigned a school.

(a) Formulate the problem as a flow network. When does an assignment of students to schools exist?

- What should the source and sink vertices be?
- What should the capacity for each edge be in the flow network?
- How should you interpret the maximum flow?

(b) Suppose that every school can only contain at most k students where $0 \leq k \leq n$. Design an $O(mn^2)$ algorithm that determines whether it it is possible to allocate all n students to m schools, or return that such an allocation is not possible.

- Modify the graph in part **(a)** to account for every school accommodating at most k students.

(c) However, you want to assign students to a school such that the schools aren't too overloaded. You define the *load* of a school to be the number of students assigned to the school. Design an $O(mn^2 \log n)$ algorithm that finds a feasible allocation of students and schools while minimising the maximum load of any particular school, or return that such an allocation is not possible.

- How should you use the solution to part **(b)**?

Problem 2. There are n cities, labelled $1, \ldots, n$, connected by m bidirectional roads. Each road connects two different cities. A criminal is currently in city 1 and wishes to get to city n on road. To catch the criminal, the police plan to prevent the criminal from reaching city n by blocking as few roads as possible. The task is to find the smallest number of roads to block to prevent the criminal from starting at city 1 and ending at city n. If it is not possible, return that it is not possible.

We will solve this problem using the maximum flow-minimum cut theorem.

(a) Construct a corresponding flow network to represent the constraints of the problem statement.

- Make sure you explicitly state what the source and sink vertices are, and how to represent the edges of the flow network along with its capacity.

(b) Using your construction from the previous part, how does the maximum flow correspond to the solution of the problem?

(c) Therefore, using a named maximum flow algorithm, solve the original problem and analyse the time complexity of the algorithm.

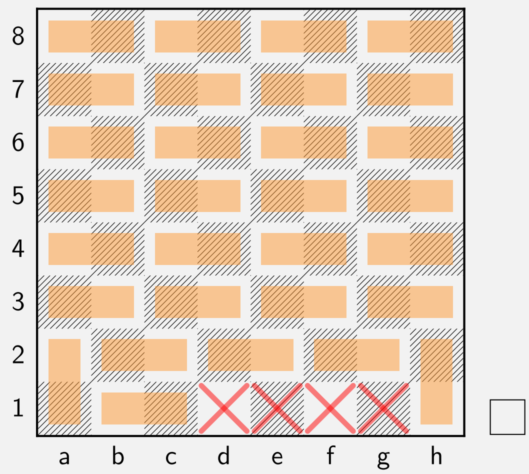
(d) However, some roads are now major roads. In order to avoid disruption, the police cannot close any major roads. The task remains unchanged; that is, the task is to still find the smallest number of roads to block. However, major roads cannot be blocked.

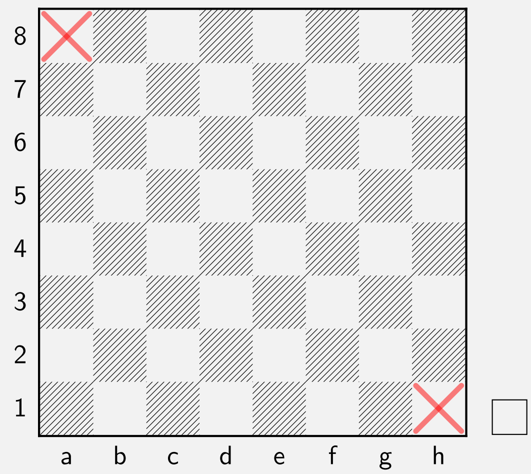
Modify the graph from part **(a)** to now account for major roads.

(e) Hence, using your construction, solve the modified problem and analyse the time complexity of the algorithm.

Problem 3. You were gifted a supply of dominoes and an $n \times n$ chessboard for Christmas. Unfortunately, some of the squares on the chessboard have been removed. A domino covers two adjacent squares and the squares alternate in colour (as shown in the diagrams below).

Your goal is to find a way to cover the chessboard with dominoes, or return that it is not possible. For example, the left board can be tiled while the second board cannot.





It turns out we can model this problem with a flow network.

(a) Explain why it might be feasible to model this as a bipartite matching problem.

- In a bipartite graph, we separate the vertex sets into two disjoint sets. What should these sets represent?
- How should a pair of vertices be connected?
- What is meant by a "matching" in this context?
- When does a bipartite matching admit a tiling?

(b) Hence, solve the original problem and analyse the time complexity of your algorithm.

After the Tutorial

After your allocated tutorial (or after having done the tutorial problems), review the discussion points. Reflect on how your understanding has changed (if at all).

- In your own time, try and attempt some of the practice problems marked [K] for further practice. Attempt the [H] problems once you're comfortable with the [K] problems. All practice problems will contain fully-written solutions.
- If time permits, try and implement one of the algorithms from the tutorial in your preferred language. How would you translate high-level algorithm design to an implementation setting?