A graph consists of a set of vertices $V$, and a set of edges $E$.

**Directed graphs**

$E \subseteq V \times V$

The edge from $v$ to $w$ is written $v \rightarrow w$

**Undirected graphs**

$E \subseteq$ subsets of $V$ of size 2

The edge between $v$ and $w$ is written $v \leftrightarrow w$

... but you'll learn all this from the videos and notes, and the live in-person sessions are for wider-ranging discussion.
Arrangements

- Pre-recorded videos and printed notes cover the examinable material
- Weekly live in-person sessions not recorded or streamed; interactive; non-examinable
- One required tick, several optional ticks (released next week)
“Can I go for a stroll around the city on a route that crosses each bridge exactly once?”
“Can I go for a stroll around the city on a route that crosses each bridge exactly once?”
“Is there a path in which every edge appears exactly once?”

\[ g = \{A: [B,B,D], B: [A,A,C,C,D], C: [B,B,D], D: [A,B,C]\} \]
How should this game agent navigate to the jetty?

1. Draw polygon boundaries around obstacles
2. Divide free space into convex polygons
3. Create a graph, with edges between adjacent polygons
4. Find a path on the graph
5. Draw this path in 2D coordinates on the map (easy, since we’ve used convex polygons)
**Q:** I’ve seen other games similar to Dwarf Fortress die on their pathfinding algorithms. What do you use and how do you keep it efficient?

**A:** Yeah, the base algorithm is only part of it. We use A*, which is fast of course, but it’s not good enough by itself.

Generally, people have used approaches that add various larger structures on top of the map to cut corners. But we can’t take advantage of these innovations since our map changes so much.

*Interview with Tarn Adams (developer) by Ryan Donovan from the StackOverflow blog, Dec 2021*
What this course is about

- Clever algorithms
- Performance analysis
- Proving correctness
Right from the beginning, and all through the course, we stress that the programmer’s task is not just to write down a program, but that his main task is to give a formal proof that the program he proposes meets the equally formal functional specification.

Edsger Dijkstra (1930—2002)
On the cruelty of really teaching computer science, 1988
- A **cycle** is a path $v_0 \leftrightarrow v_1 \leftrightarrow \cdots \leftrightarrow v_k$ with at least two vertices, where $v_0 = v_k$
- A graph is **connected** if for every pair of vertices there is a path between them.
- A **forest** is an undirected graph without any cycles.
- A **tree** is a connected forest.

Which is a tree, which is a forest?
What this course is about

▪ Clever algorithms
▪ Performance analysis
▪ Proving correctness
▪ Interplay between data and algorithm
Q: I’ve seen other games similar to Dwarf Fortress die on their pathfinding algorithms. What do you use and how do you keep it efficient?

A: Yeah, the base algorithm is only part of it. We use A*, which is fast of course, but it’s not good enough by itself.

Generally, people have used approaches that add various larger structures on top of the map to cut corners. But we can’t take advantage of these innovations since our map changes so much.

Interview with Tarn Adams (developer) by Ryan Donovan from the StackOverflow blog, Dec 2021
What this course is about

- Clever algorithms
- Performance analysis
- Proving correctness
- Interplay between data and algorithm
- What we can model with graphs
Once upon a time a farmer went to a market and purchased a wolf, a goat, and a cabbage. On the way home, the farmer came to the bank of a river and rented a boat.

The boat can carry the farmer, plus a single one of the purchases. If the wolf and goat are left unattended together, the wolf will eat the goat. Likewise, the goat and the cabbage.

How should the farmer cross the river?
GAME-PLAY PROBLEMS

Let $V$ be the set of possible game states. Let there be an edge $v \rightarrow w$ if there is an action that transitions from $v$ to $w$.

What is the shortest path from the initial state to the desired end-state?

How can I train a neural network to play the game, i.e. to pick the next action along the path from any starting point?
Training goal: learn a value function $F: V \rightarrow \mathbb{R}$ representing “how much I’ll win, starting from a given state”.

Gameplay: from any state $v$, simply pick next state $v_{\text{next}} = \arg \max_{w: v \rightarrow w} F(w)$.
Training goal: learn a value function $F: V \rightarrow \mathbb{R}$ representing “how much I’ll win, starting from a given state”.

Gameplay: from any state $v$, simply pick next state

$$v_{\text{next}} = \arg \max_{w: v \rightarrow w} F(w)$$
What order should we compute the cells in a spreadsheet?

Is it even computable?
SCHEDULING PROBLEMS

Let \( V \) be the set of tasks.

Let there be an edge \( v \rightarrow w \) if \( v \) must be completed before \( w \).

How can we arrange all the vertices into a sequence such that all edges point right? For what graphs is this even possible?

NOTICE: THIS DOCUMENT CONTAINS INFORMATION AFFECTING THE NATIONAL DEFENSE OF THE UNITED STATES WITHIN THE MEANING OF THE ESPIONAGE LAW, TITLE 18, U.S.C. SECTIONS 793 and 794. THE TRANSMISSION OR THE REVELATION OF ITS CONTENTS IN ANY MANNER TO AN UNAUTHORIZED PERSON IS PROHIBITED BY LAW.
Each edge is labelled with its capacity (trains/day)
Q. What algorithmic questions might the US Air Force ask about this graph?
Fig. 7 — Traffic pattern: entire network available

Legend:
- International boundary
- Railway operating division
- Capacity: 12 each way per day. Required flow of 9 per day toward destinations in direction of arrow with equivalent number of returning trains in opposite direction

All capacities in terms of 1,000's of tons each way per day
Origins: Divisions 2, 3N, 3C, 2A, 1NB, 13N, 13S, 12, 52 (USSR), and Roumania
Destinations: Divisions 3, 6, 9 (Poland); B (Czechoslovakia); and 2, 3 (Austria)
Alternative destinations: Germany or East Germany

Note 11K at Division 9, Poland
Flow networks

Consider a graph where each edge is labelled with a capacity \( c : E \rightarrow \mathbb{R} \).
Let there be a source vertex \( s \).

Let the value of a flow \( f : E \rightarrow \mathbb{R} \) be

\[
\text{value}(f) = \sum_{w:s \rightarrow w} f(s \rightarrow w)
\]

- How can we find a flow of maximum value, given capacity constraints?
- Which are the edges whose removal would reduce the maximum flow value?
Q. Why did Facebook choose to make CHECKIN a vertex, rather than a USER→LOCATION edge?
Alice was at the Golden Gate Bridge with Bob. Cathy: Wish we were there! David likes this.

Q. What algorithmic questions we might ask about this graph?
What this course is about

- Clever algorithms
- Performance analysis
- **Proving correctness**
- Interplay between data and algorithm
- What we can model with graphs

Topic for Fri 25 Feb