

COMP5001 Artificial Intelligence

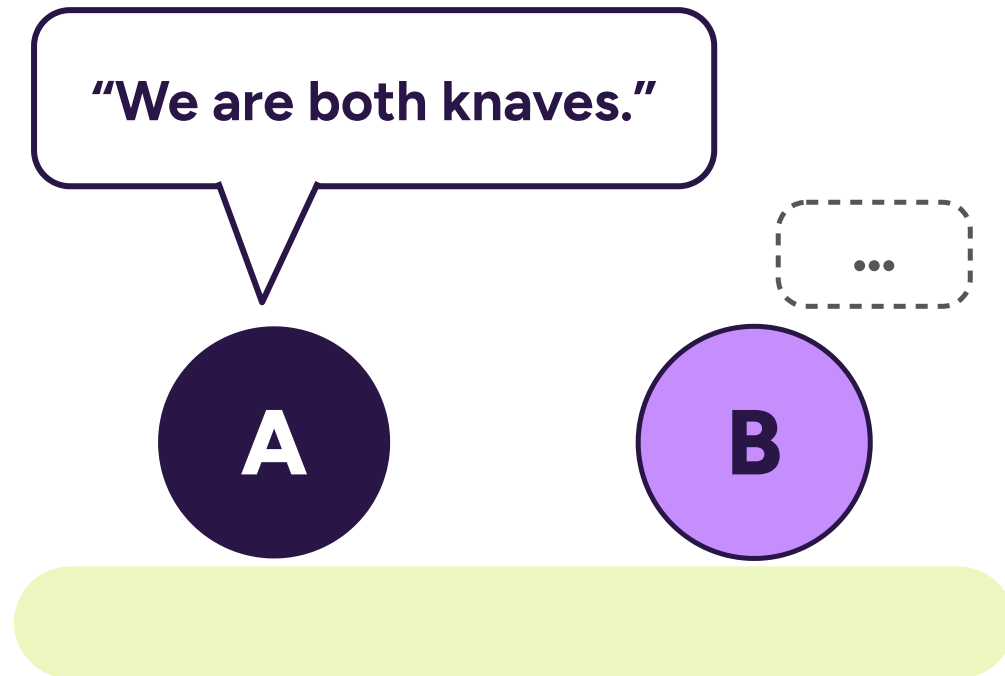
Week 3 — Heuristics and Logic

Dr Sakib Anwar

University of Winchester

2026/27

Remember Week 1?



Every person on the island is a **knight** or a **knave**. Knights always tell the truth. Knaves always lie.

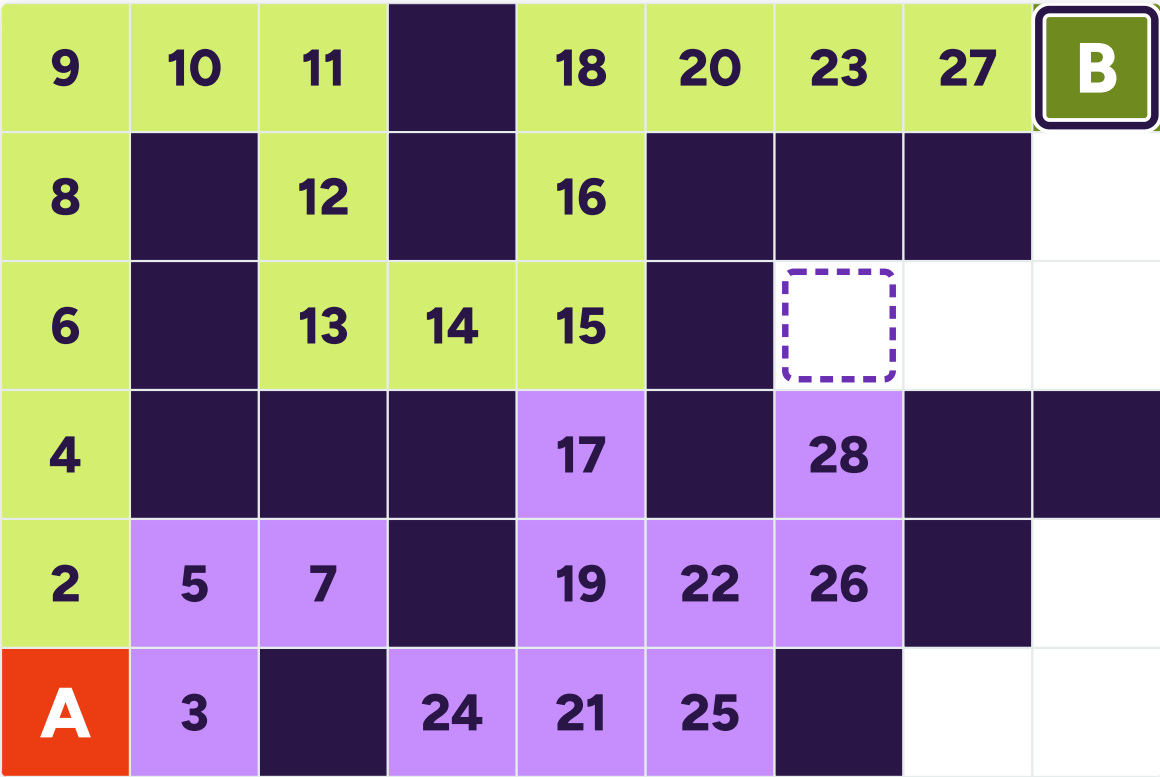
? *In Week 1 you solved this puzzle. What are A and B? How did you work it out?*

GROUPS · 5 MIN

- ▶ **Answer:** A is a knave and B is a knight.
- ▶ **One way:** suppose A is a knight. Then "we are both knaves" is true, so A is a knave. Impossible! So A is a knave, the sentence is false, and B must be a knight.

There are only four possibilities: knight or knave, for A and for B. Later today a computer checks all four, and finds the same answer. It is called **model checking**.

Where Week 2 left off



■ explored by BFS (in order) ■ path

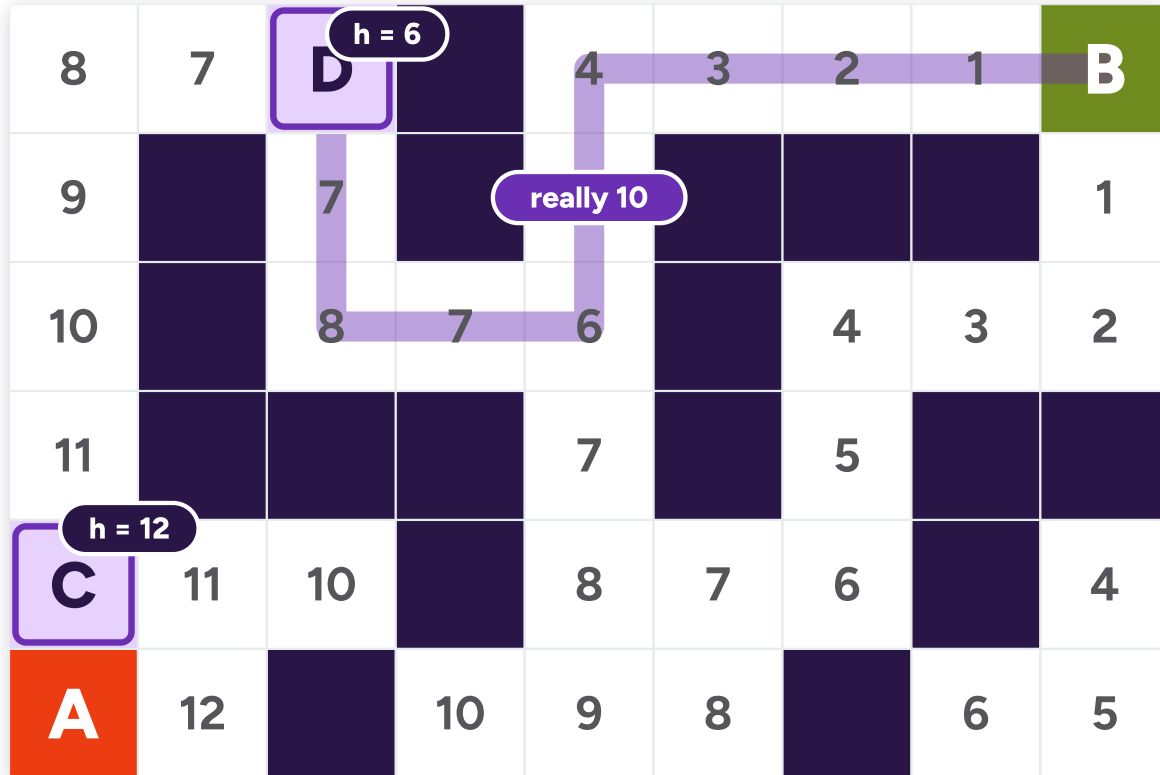
? *Last week BFS explored 29 squares to find the 17-step path. When you solved this maze yourself in Week 1, did you look at every square?*

No. You could **see where B was**, so you headed towards it.

UNINFORMED SEARCH
DFS and BFS. They know nothing about where the goal is, so they try squares in a fixed order.

INFORMED SEARCH
Uses extra knowledge about the problem, such as a guess at how far each square is from the goal, to look in the right direction first.

A heuristic is a guess



The Week 1 maze. C and D are two squares to compare. The grey numbers (last click) are each square's Manhattan distance to B: its h .

? *Ignore the walls. Which square is closer to B: C or D?*

D: 0 rows and 6 columns away, so **6**. **C:** 4 rows and 8 columns away, so **12**.

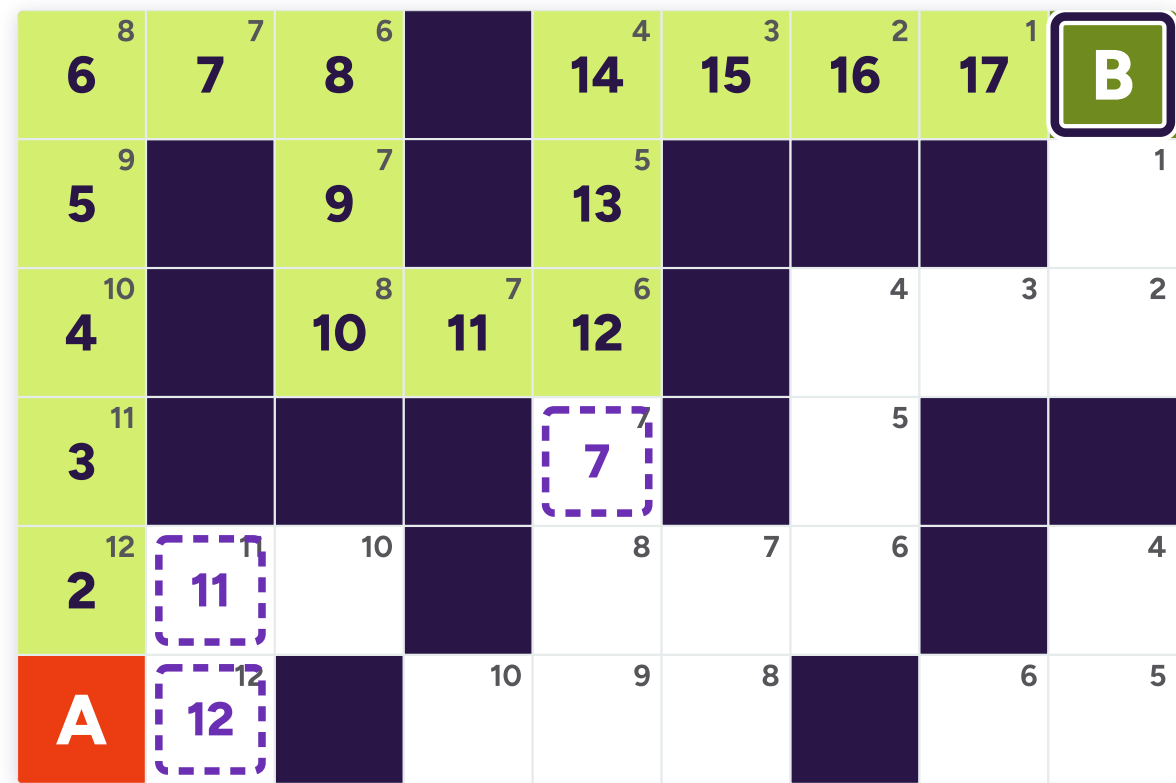
? *Can you really get from D to B in 6 steps?*

No: the wall forces a detour of **10** steps. A heuristic is a guess, and it can be wrong. But it is very quick to work out.

HEURISTIC

$h(n)$: a quick estimate of the cost from square n to the goal. In a maze, the **Manhattan distance**: rows apart + columns apart, ignoring walls.

Greedy best-first search



⁷ = *h*, the guess of steps to B frontier: purple number = its *h*
■ explored (in order) ■ path

Rule: always take the square in the frontier with the **smallest *h*** (the small number in its corner).

Steps 2–3. Up and right have the same *h*. On a tie, greedy takes the square added first: up.

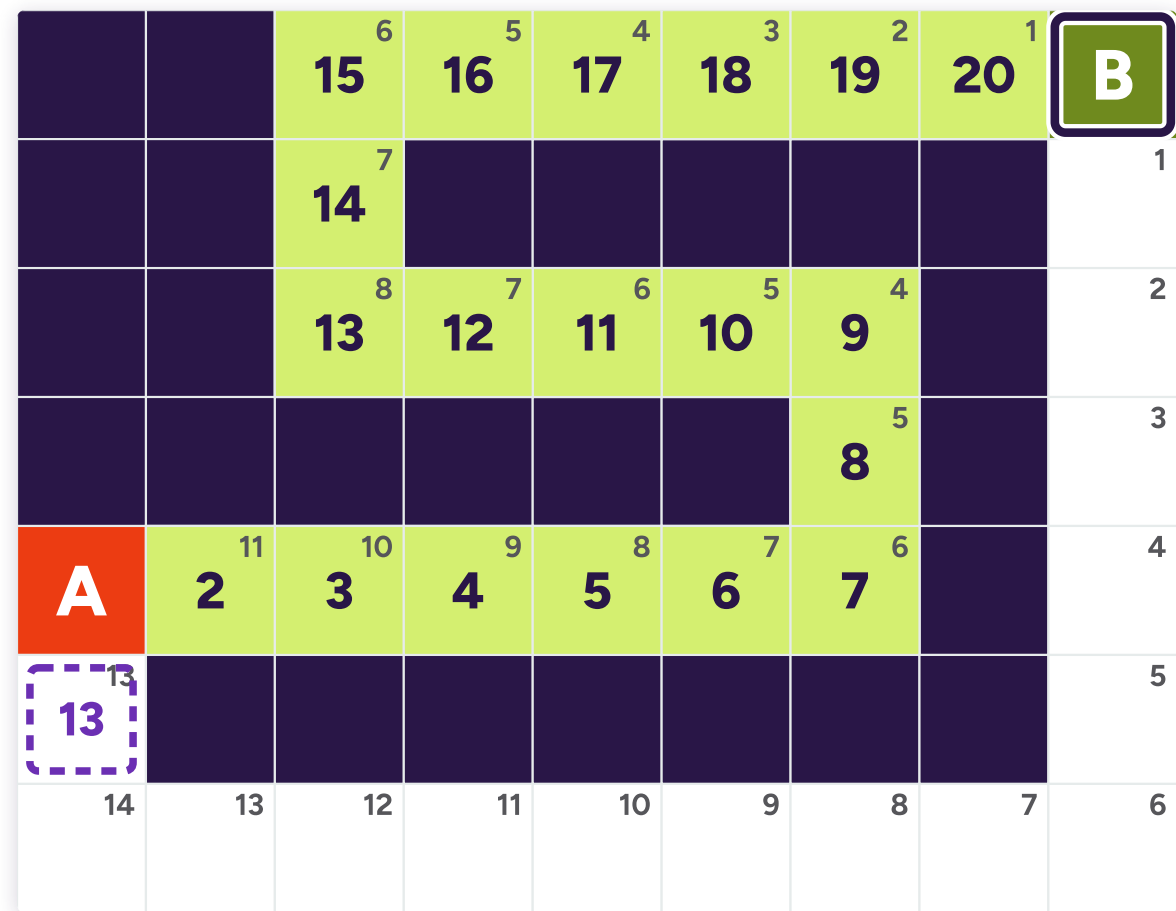
Steps 9–10. Blocked by the wall at the top, it goes down: *h* rises to 7, then 8. The squares still waiting have *h* = 11 or 12, so these still win.

Step 18. B, after only **18** squares. BFS needed 29.

Path: 17 steps, the shortest.

? *Greedy was faster than BFS and still found the shortest path. Is it always this good?*

Where greedy goes wrong



⁷ = h , the guess of steps to B 13 frontier: purple number = its h
■ explored (in order) ■ path

? A new maze. Which way would you go from A? Which way will greedy go?

Step 2. Right has $h = 11$; down has $h = 13$. Greedy goes right.

Step 9. The corner: $h = 4$. B looks close, but there is a wall in the way.

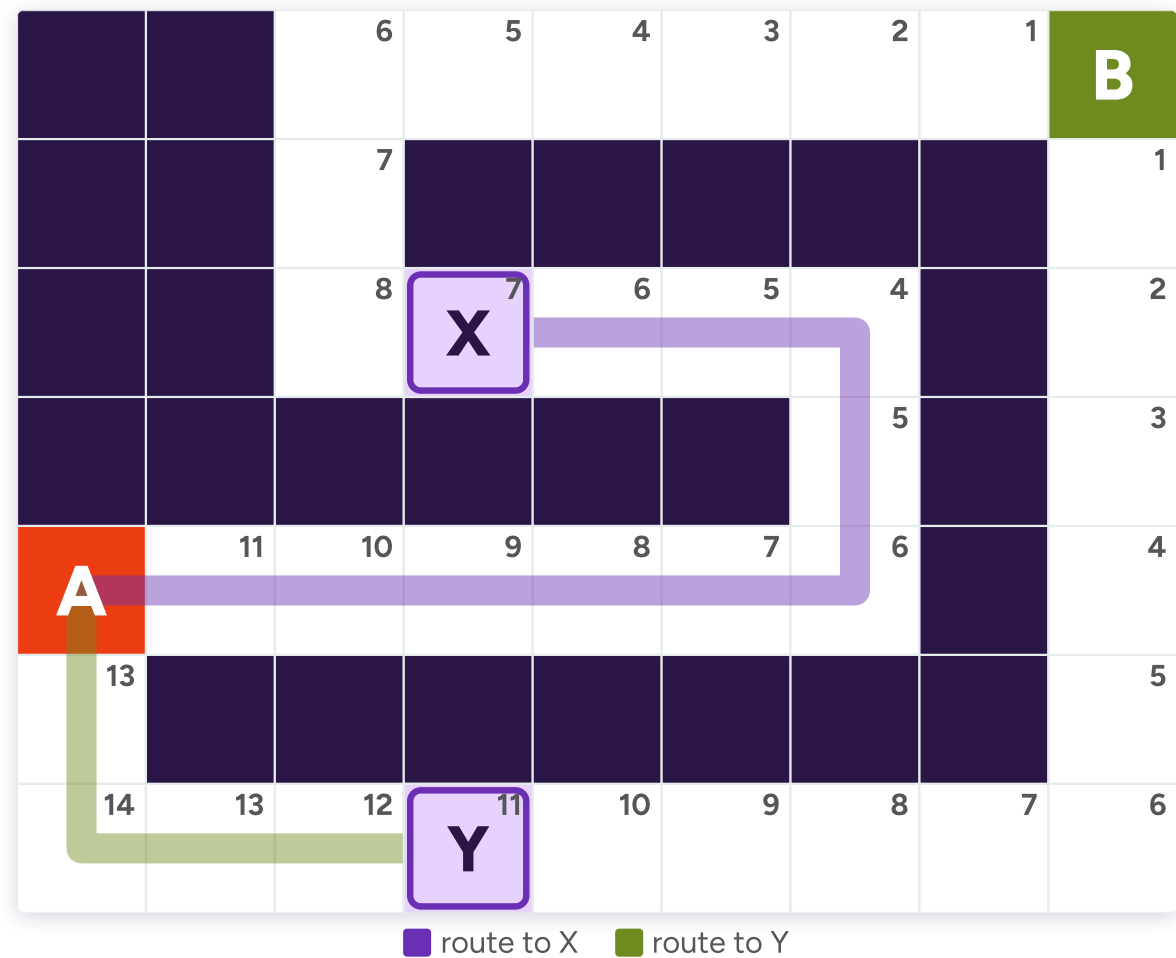
Steps 10–13. The way turns back to the left and h rises to 8. Down (13) still looks worse, so greedy keeps going.

Step 21. B, after 21 squares.

Path: 20 steps.

KEY IDEA
The shortest path goes down and along the bottom: **16 steps**. Greedy only looks at h , the guess to go. It never counts the steps it has already taken.

A*: steps so far + guess to go



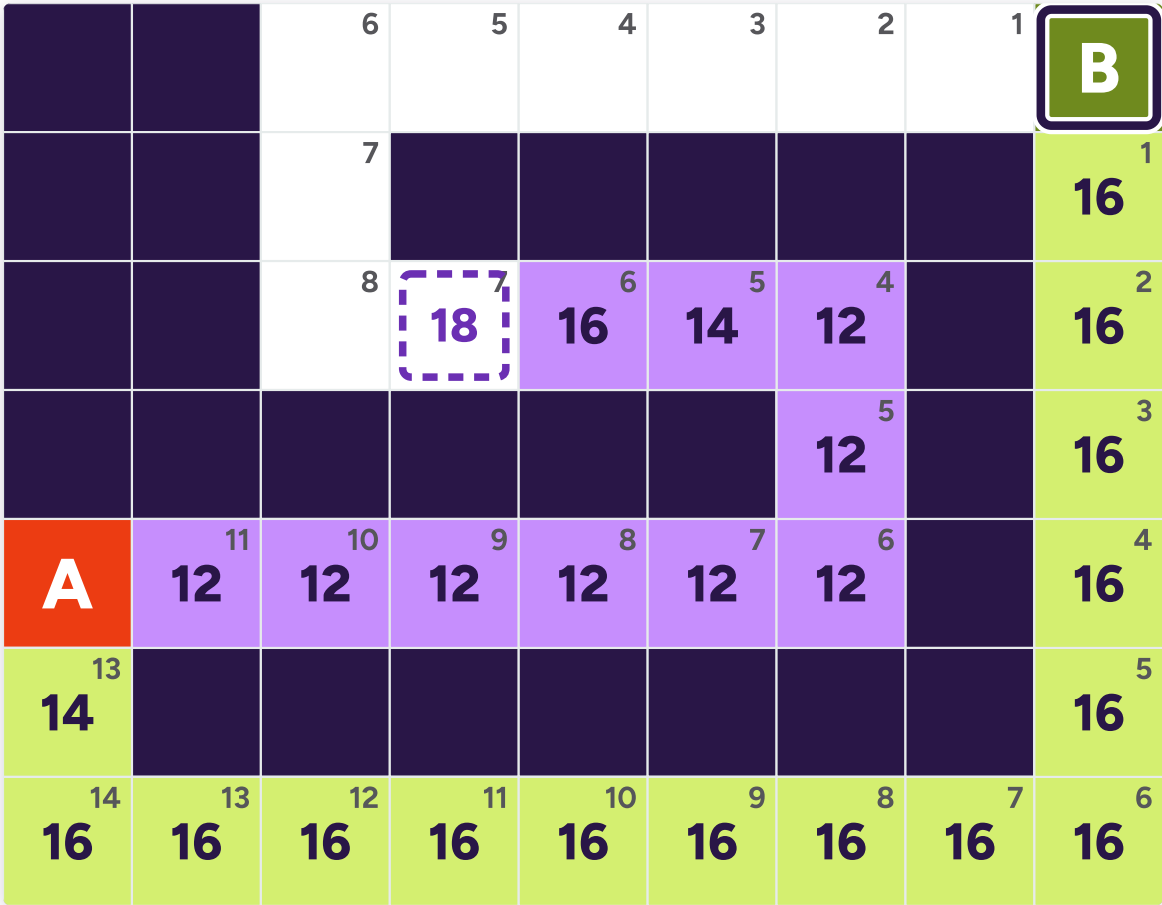
? *X and Y are both waiting in the frontier. Which should the search take next?*

	X	Y
<i>h(n)</i> : guess to B	7	11
<i>g(n)</i> : steps from A so far	11	5
<i>f(n) = g(n) + h(n)</i>	18	16

Greedy looks only at *h*, so it takes **X**.
Counting the steps already taken, **Y** is the better bet.

A* SEARCH
Take the frontier square with the smallest *f(n) = g(n) + h(n)*: the cost so far plus the guessed cost to go.

A* step by step



h = h , the guess of steps to B frontier: purple number = its f
explored: big number = f path

Rule: take the square with the **smallest $f = g + h$** .

Steps 1–9. Along the top route f stays at 12: each step adds 1 to g and takes 1 off h .

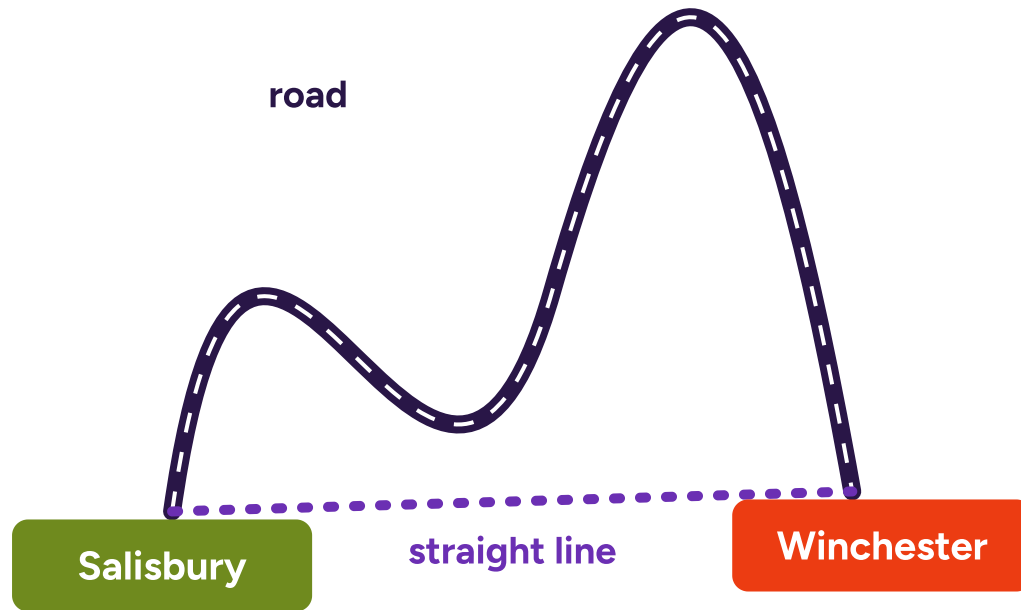
Steps 10–11. Round the corner f jumps to 14, so A* also tries the square below A ($f = 14$).

Steps 12–27. One more square on top ($f = 16$), then the top route reaches 18. The bottom route stays at 16, so A* follows it to B.

Path: 16 steps, the shortest.

maze4.txt	Squares explored	Path
BFS	32	16
Greedy	21	20
A*	27	16

A good guess never overestimates



Two ways to measure Salisbury to Winchester: along the road, or in a straight line.

? *Can the straight line between two towns ever be longer than the road between them?*

No. So straight-line distance never overestimates. Nor does Manhattan distance in a maze: walls only make the real route longer.

ADMISSIBLE HEURISTIC

Never overestimates the true cost to the goal. **A* with an admissible heuristic always finds the shortest path.**

? *What if h guessed too high? Say $h(Y)$ were 30 on the last slide.*

Then $f(Y) = 5 + 30 = 35$, worse than the top route. A* would go the long way and return the 20-step path.

What is logic?



Which of these can be true or false?

WHOLE CLASS

Sentence	True or false?
Winchester is in Hampshire.	true
2 + 2 = 5.	false
Close the door!	neither: an order
Is it raining?	neither: a question
It is raining.	true or false, depending on the day

STATEMENT

A sentence that is either **true** or **false**. Orders and questions are not statements.

LOGIC

A precise way to write statements, with rules for working out what follows from them. It is the language of **symbolic AI** from Week 1.

Today we write statements in logic, then let a computer **work out what follows** from them.

Knowledge-based agents

WHAT WE KNOW

1. If it didn't rain, Harry visited Hagrid today.
2. Harry visited Hagrid or Dumbledore today, but not both.
3. Harry visited Dumbledore today.
4. Harry did not visit Hagrid today. (from 2 and 3)
5. It rained today. (from 1 and 4)

? *Did it rain today? None of the three sentences says so.*

WHOLE CLASS

Harry visited Dumbledore, and not both, so he did **not** visit Hagrid.

If it hadn't rained, he would have visited Hagrid. He didn't, so **it rained**.

KNOWLEDGE-BASED AGENT

An agent that stores what it knows as **sentences** (statements written in a logic), and reasons with them to reach new conclusions.

INFERENCE

Working out new sentences from old ones, as we just did for sentences 4 and 5.

Writing sentences with symbols

Give each statement a short name:

- ▶ It rained today. → **rain**
- ▶ Harry visited Hagrid today. → **hagrid**
- ▶ Harry visited Dumbledore today. → **dumbledore**

PROPOSITIONAL SYMBOL

A short name for one statement. In any world, it is either **true** or **false**. Textbooks often use single letters: P, Q, R.

? *Sentence 1 says: “If it **didn’t rain**, **Harry visited Hagrid** today.” Which of our statements are inside it?*

Two: **rain** (with a “not”) and **hagrid**, joined by “if ... then”.

PROPOSITIONAL LOGIC

Symbols, joined by five words: **not**, **and**, **or**, **if ... then** and **if and only if**. The next two slides give each word a symbol.

Joining sentences: not, and, or

- ▶ **not**: "It did not rain." → $\neg \text{rain}$
- ▶ **and**: "Harry visited Hagrid and Dumbledore." → $\text{hagrid} \wedge \text{dumbledore}$
- ▶ **or**: "Harry visited Hagrid or Dumbledore." → $\text{hagrid} \vee \text{dumbledore}$

Read $\neg$ as "not", $\wedge$ as "and", $\vee$ as "or".

"OR" MEANS AT LEAST ONE

$P \vee Q$ is **true** when P is true, or Q is true, **or both**. To say "but not both", add $\neg(P \wedge Q)$.

? When is each sentence true? Fill in the table.

PAIRS · 3 MIN

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$
false	false	true	false	false
false	true	true	false	true
true	false	false	false	true
true	true	false	true	true

P and Q stand for any two sentences. Each row is one possible world.

? How do we write sentence 2, "Harry visited Hagrid or Dumbledore, but not both"?

$(\text{hagrid} \vee \text{dumbledore}) \wedge \neg(\text{hagrid} \wedge \text{dumbledore})$

If ... then, and if and only if

- ▶ **if ... then:** "If it is raining, then I am indoors."

→ **rain → indoors**

Read **rain → indoors** as "rain implies indoors".

- ▶ **if and only if:** "I am indoors if, and only if, it is raining." → **rain ↔ indoors**

KEY IDEA

$P \rightarrow Q$ is **false** only when P is **true** and Q is **false**. $P \leftrightarrow Q$ is **true** when P and Q are the same.

? *I promised: "If it is raining, I am indoors." In which worlds did I break my promise?*

WHOLE CLASS

rain	indoors	rain → indoors	rain ↔ indoors
false	false	true	true
false	true	true	false
true	false	false: broken!	false
true	true	true	true

When it is **not** raining, the promise says nothing about where I am, so it is not broken: the sentence counts as **true**.

Models: every possible world

World 1

rain true
hagrid true

World 2

rain true
hagrid false

World 3

rain false
hagrid true

~~World 4~~

~~rain false~~
~~hagrid false~~

3 symbols: **8** worlds. n symbols: **2^n** worlds.
Each new symbol doubles the number.

? *Two symbols, rain and hagrid. How many different worlds are possible?*

WHOLE CLASS

MODEL

One possible world: a true or false value for every symbol.

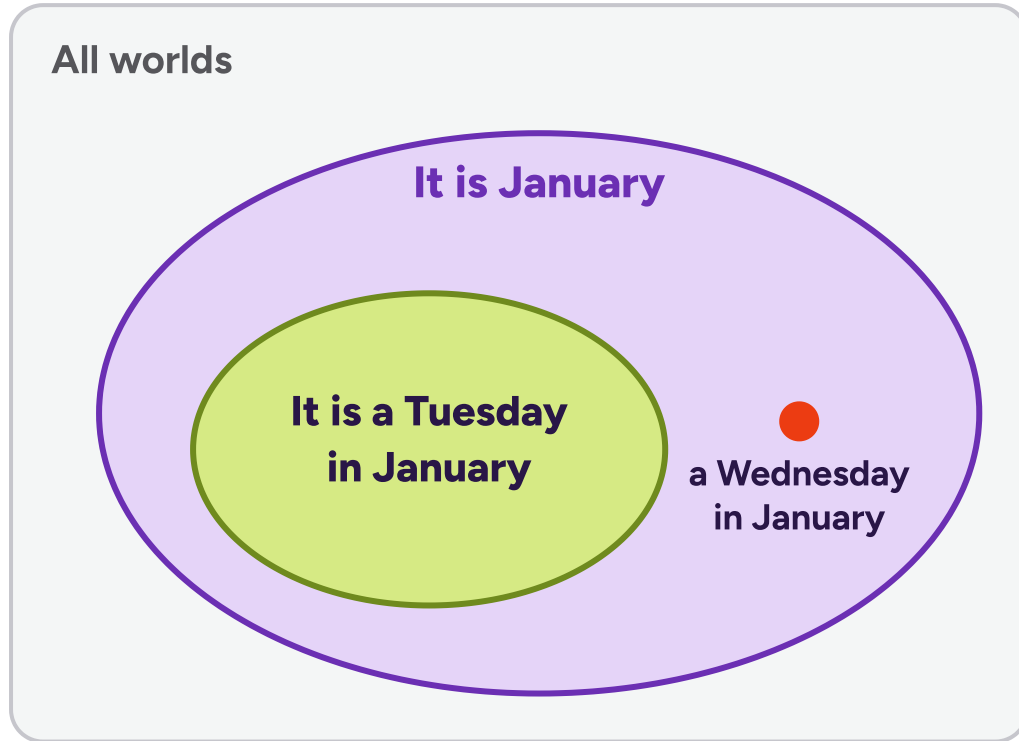
KNOWLEDGE BASE (KB)

The sentences an agent knows are true.

? *Suppose the KB is one sentence, rain v hagrid. Which worlds does it rule out?*

Only world 4, where both are false. **Knowledge rules worlds out.**

Entailment: what follows



? If “It is a Tuesday in January” is true, must “It is January” be true? What about the other way round?

WHOLE CLASS

Yes. Every world where it is a Tuesday in January is a world where it is January.

No. On a Wednesday in January, “It is January” is true, but “It is a Tuesday in January” is not.

ENTAILMENT

A **entails** B when B is **true** in **every** world where A is **true**: B follows from A. Written $A \models B$.

Our question is always: **$KB \models \text{query?}$** Is the query true in every world that fits what we know?

Model checking

KNOWLEDGE BASE

- 1. $\neg \text{rain} \rightarrow \text{hagrid}$
- 2. $\text{hagrid} \vee \text{dumbledore}$
- 3. $\neg(\text{hagrid} \wedge \text{dumbledore})$
- 4. dumbledore

Query: rain . Did it rain?

MODEL CHECKING

List every world. Keep the worlds where the KB is **true**. If the query is **true** in all of them, then $\text{KB} \models \text{query}$.

? Three symbols, so 8 worlds. Which ones fit everything we know?

rain	hagrid	dumbledore	Ruled out by
false	false	false	4
false	false	true	1: no rain, so Hagrid
false	true	false	4
false	true	true	3: not both
true	false	false	4
true	false	true	fits everything
true	true	false	4
true	true	true	3: not both

rain is **true** in every world that is left, so $\text{KB} \models \text{rain}$. It rained.

Knights and knaves, solved

? *A said: “We are both knaves.” Write what we know as sentences.*

WHOLE CLASS

- ▶ Each person is a knight or a knave, not both. →
 $(AKnight \vee AKnave) \wedge \neg(AKnight \wedge AKnave)$, and the same for B
- ▶ If A is a knight, what A said is true. →
 $AKnight \rightarrow (AKnave \wedge BKnave)$
- ▶ If A is a knave, what A said is false. →
 $AKnave \rightarrow \neg(AKnave \wedge BKnave)$

Four symbols give 16 worlds. The first sentence leaves only the four on the right.

A is a	B is a	Ruled out?
~~knight~~	~~knight~~	yes: a knight said “both knaves”
~~knight~~	~~knave~~	yes: a knight said “both knaves”
knave	knight	no: it fits
~~knave~~	~~knave~~	yes: a knave said something true

One world is left, so the KB entails **A is a knave and B is a knight**: the answer you found in Week 1, now checked by a method a computer can follow.

In code: `harry.py`

```
from logic import *

rain = Symbol("rain")
hagrid = Symbol("hagrid")
dumbledore = Symbol("dumbledore")

knowledge = And(
    Implication(Not(rain), hagrid),
    Or(hagrid, dumbledore),
    Not(And(hagrid, dumbledore)),
    dumbledore
)

print(model_check(knowledge, rain))
```

`logic.py` gives `Symbol`, `Not`, `And`, `Or`, `Implication`, `Biconditional` and `model_check`.

? Match the lines of `knowledge` to sentences 1–4 on the model-checking slide. What does the last line print?

Same four sentences, in the same order. `And(...)` joins them: the KB is true only when all four are.

It prints **True**: the knowledge base entails `rain`.

What `model_check` does, in plain steps:

```
for each world (every true/false choice of the symbols):
    if the KB is true in this world:
        if the query is false here: return False
return True
```

Heuristics and logic in the real world

- ▶ **Sat-navs and games:** A* with a good heuristic, such as straight-line distance.
- ▶ **Computer chips:** model checking tries every state of a design, to show that a bug can never happen. Its inventors won the 2007 Turing Award.
- ▶ **Railways:** logic is used to check that signals can never send two trains onto the same track.
- ▶ **Product options:** logic works out which options can go together, for example when you build a car online.

Discuss

❓ *A railway signalling system has been checked by model checking: in every world that fits its knowledge base, no two trains share a track. Can we trust it completely? What could still go wrong?*

GROUPS • 10 MIN

The check is only as good as the **knowledge base**. A missing or wrong sentence, or something the symbols do not cover (a broken sensor), is outside the proof. People who know the railway must check the sentences, and the system still needs testing.

Thank you

Practical: `week04-knowledge/logic` in the module code (`harry.py`)

Next week: Knowledge and Planning. Faster ways to reason than checking every world, then planning with facts and actions

Reading: Russell & Norvig, sections 3.5–3.6 and 7.1–7.4; Poole & Mackworth, sections 3.6 and 5.1

Sakib.Anwar@winchester.ac.uk

Adapted from *CS50's Introduction to Artificial Intelligence with Python* by Brian Yu and David J. Malan, Harvard University, licensed under [CC BY-NC-SA 4.0](#).