COMP9414 Artificial Intelligence

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING · UNSW SYDNEY · EXAM REVISION SHEET

Full guide + ask Sia about COMP9414

SIDE 1/2 PRIORITIES · DEFINITIONS · FORMULAS · DECISION RULES · ASSUMPTIONS

TERM 3 · 2026

v2 · 2026-10-25 · **revision + self-test**

0 · Exam Map

COURSE.JSON + BRIEF

A1 20% | A2 20% | Tut | Final 50%

COMPONENT	WEIGHT	FORMAT
Assignment 1	20%	Individual; W3→W5
Assignment 2	20%	Individual; W7→W9
Tutorial work	10%	Individual; weekly
Final exam	**50%**	Individual; examination period

Permitted materials: not confirmed for this offering in the course materials we hold — check the exam instructions on the LMS.

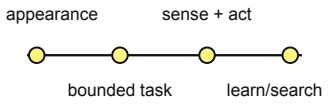
DOUBLE-PASS RULE

```
overall mark ≥ 50/100 AND exam mark ≥ 20/50.
```

Revision priority from the grounded material: **state the model → expose the frontier or fact-set update → name the assumption behind the guarantee → report the path/goal check separately.**

1 · Capability Claims

C1 TOPIC MAP + FAQ



Narrow the claim: observed behaviour ≠ mechanism, transfer or general capability.

WRITE	QUESTION TO ANSWER
Observation	What succeeded, in exactly which tested setting?
Mechanism	Search, learning, retrieval or unknown?
Boundary	Which changed condition tests the broadest word?
Judgement	What is supported; what remains an inference?

Boundary test = changed-condition evaluation that reveals where a claim stops holding. Change wording, world state, feedback or task distribution *one at a time*; failure narrows the claim without erasing the original success.

RECALL → *Replace "intelligent" with a falsifiable sentence: behaviour + setting + mechanism evidence + decisive next test.*

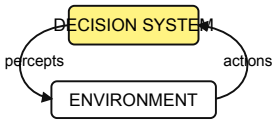
LANDMARKS AS EVIDENCE TYPES

CASE	WHAT THE EVIDENCE ACTUALLY ADDS
Chess cabinet	Appearance can hide a human mechanism.
Deep Blue	Specialised evaluation + many candidate continuations; bounded chess success.
RoboCup	Noisy observations, movement, teamwork, opponents and time pressure.
AlphaGo	Learning combined with tree search in a defined task.

Equal headline success does not imply equal transfer: compare **task model, mechanism and evaluation conditions**, not a single vague intelligence scale.

2 · Decision System Loop

C2 OVERVIEW + FIGURE



Percepts inform a decision; actions alter the world that produces the next percept.

Rational action maximises expected utility relative to the stated objective, model and available information. A sound decision can still have a poor stochastic outcome: decision quality ≠ luck.

TASK MODEL

```
state + percepts + legal actions + transition assumptions + goal test + path cost / utility.
```

State-minimality test: if two histories map to the same proposed state, every future action must have the same legality, goal meaning and future cost. If not, add the missing variable; if yes, discard irrelevant history.

2b · Environment → Design

C2 TOPIC MAP + FAQ

PROPERTY	IF PRESENT	DESIGN CONSEQUENCE
Partially observable	Current percept omits relevant state	Store history/belief; sense again
Stochastic	Action has several outcomes	Contingencies or expected utility
Dynamic	World changes during choice	Monitor; refresh stale facts
Continuous	Variables/actions not discrete	Choose an abstraction or motion model

LIBRARY-RETURN FORMULATION

- **State:** cart location, remaining boxes, load, battery band, latest corridor status.
- **Percepts:** location, box/load and corridor observations.
- **Actions:** move, collect, unload, recheck.
- **Goal:** both boxes cleared; cart at sorting; nothing loaded.
- **Cost:** travel time plus missing-closing penalty; re-sense because the corridor is dynamic and partly observed.

TRAP → *A route-only state cannot represent capacity, completion, battery or corridor knowledge; a stronger algorithm cannot repair a missing state variable.*

INFORMATION + OBJECTIVE AUDITS

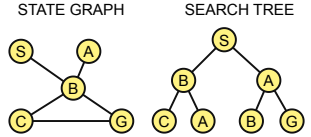
3 · Search Formulation

C3 OVERVIEW + TOPIC MAP

WRITE BEFORE TRACING

```
initial state s₀ · successor function Succ(s) · goal test Goal(s) · path cost g(path).
```

INGREDIENT	IT DECIDES	FAILURE IF WRONG
State	What futures are distinguished	Merges non-equivalent choices
Successors	Legal next states	Allows impossible moves
Goal test	What counts as success	Stops too early / never stops
Path cost	Meaning of "cheapest"	Optimises the wrong quantity



The graph stores world states once; the tree stores path histories, so labels may repeat.

Frontier entry must retain enough to reconstruct a candidate plan: state + parent/full path + depth; cost-sensitive search also stores g and priority. A state label alone cannot explain how it was reached.

3b · Trace Cycle + Invariants

C3 WORKED EXAMPLE

GENERAL LOOP

```
choose frontier entry → test goal → expand → generate ordered successors → apply duplicate/cost rule → insert.
```

RECORD SEPARATELY	MEANING
Expansion order	Entries removed and expanded
Returned path	Parent chain ending at goal
Actual cost	Sum of route edges = g(goal)

Before step 1, write conventions: neighbour order; goal test on generation or removal; tree/graph search; duplicate policy; tie rule. Different conventions can alter the visible sequence without changing the algorithm family.

Debug at first divergence: wrong successor order, wrong visited time, duplicate inserted twice, or a worse path replacing a cheaper parent.

STATE-DESIGN DEFENCE

CANDIDATE DETAIL	KEEP IFF CHANGING IT CAN ALTER…
Resource / tool possession	Action legality or successor facts
Completed jobs	Goal truth or remaining choices
Battery / time band	Feasibility or future cost
Fixed map label	Usually problem data, not changing state

4 · Frontier Chooser

C4 OVERVIEW + FIGURE

DFS	BFS	UCS	A*
deepest	shallowest	lowest g	lowest
LIFO	FIFO	priority	g + h
stack	queue	queue	priority

All share the loop; the frontier priority changes what "best next" means.

METHOD	SELECT	CONDITIONED CONCLUSION
DFS	Deepest/latest path	Not generally optimal; cycles/depth need control
BFS	Fewest actions so far	Complete in finite branching; optimal for equal action cost
UCS	Minimum g	Cheapest path under positive action costs
Greedy	Minimum h	Goal-directed; ignores cost already paid
A*	Minimum f=g+h	Optimal with the taught admissibility condition

SYMBOLS

```
g(n)=actual cost from start
h(n)=estimated cheapest remaining cost
f(n)=g(n)+h(n).
```

BFS minimises edge count, not arbitrary weight. A shallow route can be expensive. UCS ignores depth and expands lowest cumulative cost.

4b · Duplicate Policy

C3 FAQ + C4 STRATEGY

CASE	SAFE RULE
Standard BFS, unweighted	Suppress a state already discovered
UCS / A*	Keep later route iff new g < best known g
Stale priority entry	Discard if its g exceeds best-cost table

Replace a parent only on **strict cost improvement**. Reconstruct the result by following parents backwards; never read the expansion order as if it were the returned route.

RECALL → *Selection rule + assumption + conclusion. A guarantee without its condition is incomplete.*

WHY THE CHOICE RULE MATTERS

RULE	PATHS PROCESSED BEFORE A GOAL	WHAT MAY REMAIN STORED
LIFO	A latest deep branch	Alternatives along that descent
FIFO	All shallower frontier paths	A broad depth layer
Minimum g	Every valid lower-cost frontier path	Cost-ordered alternatives
Minimum g+h	Lowest estimated-total candidates	Promising alternatives plus best-cost records

5 · Heuristic Checks

C4 FAQ + WORKED EXAMPLE

ADMISSIBILITY

```
for every n: 0 ≤ h(n) ≤ h*(n), where h*(n) is the true cheapest remaining cost; h(goal)=0.
```

Admissible means optimistic, not necessarily accurate. One overestimate removes the stated A* optimality condition. h≡0 ⇒ f=g ⇒ A* becomes UCS : admissible but supplies no direction.

CHECK	DO	INTERPRET
Numerical audit	Solve exact remaining cost h*	Compare every h with h*
Route update	Compute new g through parent	Update only if strictly smaller
Priority	Recompute f=g+h	Select smallest valid f
Report	Sum actual edges	Cost is g(goal), not Σh

A* LEDGER TEMPLATE

NODE	PARENT	G	H	F	STATUS
—	—	—	given	g+h	frontier / expanded
—	replace?	new g	fixed	new f	improved / stale

Tie handling: apply only the stated rule. In the guide's worked graph equal f is broken alphabetically. State the rule before expanding so a trace is reproducible.

Do not overclaim: an inadmissible estimate removes the assurance; it does not prove A* must return a bad path on every graph.

5b · Method Chooser

C4 TOPIC MAP

IF THE TASK ASKS…	CHOOSE / SHOW
Any deep route; memory pressure	DFS; show stack and cycle risk
Fewest actions, equal costs	BFS; show FIFO layers
Lowest weighted cost	UCS; show g and improvements
Weighted cost + estimate	A*; show g,h,f and admissibility
Goal-directed but no cost assurance	Greedy; show h-only priority

Compare expansion counts only after aligning neighbour order, goal-test timing, duplicate policy and tie-breaking; otherwise the comparison mixes conventions with algorithms.

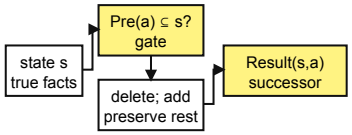
TRAP → *Never add heuristic values along the returned path. h orders; g improves and reports cost.*

HEURISTIC STRESS TEST

1. Solve the tiny remaining-cost problem exactly; write h* beside every node.
2. Set every h to zero and confirm the frontier choices reduce to UCS.
3. Raise one h above h*: identify the lost assurance without claiming failure is certain.

6 · STRIPS Transition

C5 OVERVIEW + FIGURE



Applicability first; then delete, add and preserve every unmentioned fact.

APPLICABLE

```
Pre(a) ⊆ s. If any grounded precondition is absent, STOP: action is illegal.
```

SUCCESSOR

```
Result(s,a) = (s - Del(a)) ∪ Add(a).
```

Closed-world convention: facts absent from the current set are treated as false in the tutorial model. ***Frame assumption:*** facts not named in effects persist.

LAYER	CONTAINS	QUESTION ANSWERED
Domain	Predicates; action schemas	What can exist and change?
Problem	Objects; initial facts; goals	Which concrete case?
Grounded action	Variables replaced by objects	Is this step applicable now?

6b · Validate + Repair

C5 FAQ + WORKED EXAMPLE

1. Ground the next action with concrete objects.
2. Compare **every** precondition with the current fact set.
3. If one is false, name the first illegal action + missing fact ; do not continue the original plan.
4. If legal, cross out deletes, insert adds, preserve all other facts.
5. After the last step, test that every goal fact is present.

Goal interaction occurs when achieving one goal deletes a fact needed to achieve or preserve another. Put the goal-producing actions beside their preconditions and delete effects; repair by a valid interleaving, then replay forward.

FORMAL MODEL ASSUMES	EXECUTION RESPONSE IF FALSE
Fully observable facts	Sense / update belief
Deterministic effects	Conditional plan / monitor
Static world during planning	Replan on changed status
Symbolic feasibility	Check geometry / motion separately

RECALL → *A plausible itinerary is not a valid plan if an earlier action was inapplicable.*

PLANNING AS SEARCH

COMP9414 Artificial Intelligence

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING · UNSW SYDNEY · EXAM REVISION SHEET

Full guide + ask Sia about COMP9414

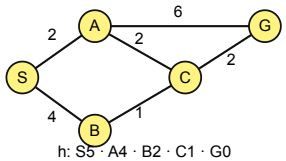
SIDE 2/2 SELF-TEST · 4 WORKED MINI-EXAMPLES · JUDGEMENT CARDS · CLASSIC TRAPS · MORNING PLAN

ANSWER PATH VISIBLE

v2 · 2026-10-25 · cover, solve, check

7 · Worked A* Trace

COURSE WORKED EXAMPLE



h: S5 · A4 · B2 · C1 · G0

Equal f is broken alphabetically; update a parent only when the new g is smaller.

REMOVE	GENERATE / IMPROVE	FRONTIER DECISION
S	A: g2,h4,f6; B: g4,h2,f6	Tie → A
A	C: g4,h1,f5; G: g8,h0,f8	Minimum → C
C	G improves: g6,h0,f6	Minimum → G
G	Goal removed	Return parents

ANSWER

`S-A-C-G; cost = 2+2+2 = 6.`

A later B→C route gives g=4+1=5, worse than C's current g=4, so reject it. The earlier direct goal at cost 8 becomes stale after the improvement to 6.

ADMISSIBILITY AUDIT

N	H	TRUE CHEAPEST H*	CHECK
A	4	4 via C	4≤4 ✓
B	2	3 via C	2≤3 ✓
C	1	2	1≤2 ✓
G	0	0	0≤0 ✓

COVER + ANSWER → *Why is the returned cost 6 rather than the sum of f or h? Because g is the accumulated real edge cost.*

7b · A* Failure Checks

C3–C4 STRATEGY

IF YOU SEE…	REPAIR
Visited rule copied to UCS/A*	Compare new g; reopen/update on improvement
BFS called "cheapest"	Say fewest edges; require equal costs
Σh reported as route cost	Sum real edges; report g(goal)
Admissible called "accurate"	State only h≤h*; h=0 is allowed

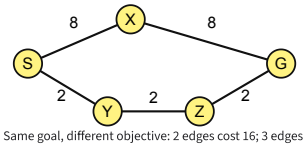
7c · Fast Recall Grid

GLOSSARY

TERM	SAY THIS
Complete	Finds a solution when one exists, under stated conditions
Optimal	Returns minimum defined path cost, under stated assumptions
Frontier	Generated partial paths waiting for selection
Ground fact	Predicate with concrete objects, no variables

8 · Worked BFS vs UCS

C4 SEARCH DRILL



Same goal, different objective: 2 edges cost 16; 3 edges cost 6.

METHOD	RETURNS	REASON
BFS	S–X–G; cost 8+8=**16**	Two edges is shallower
UCS	S–Y–Z–G; cost 2+2+2=**6**	Lowest cumulative g

Answer sentence: BFS is optimal in action count when every action has equal cost; that condition is false here. UCS processes paths by accumulated cost and therefore returns the deeper but cheaper route.

TRAP → *"BFS finds the shortest path" is ambiguous. Say fewest edges; say "cheapest" only under equal action costs.*

9 · Worked Capability Audit

C1 WORKED EXAMPLE

observe test set → method? proof? → vary setting → bounded claim

Audit a campus kiosk without inflating or dismissing its demonstrated performance.

1. **Observation:** tested scripted orientation responses were correct.
2. **Mechanism question:** retrieval, map search, pattern matching or generated language?
3. **Boundary:** ambiguous room, closed corridor or dependent follow-up.
4. **Judgement:** evidence supports the tested query set; broader understanding depends on changed-condition results.

Nearest unsupported word: "understands". Turn it into a test instead of praise or dismissal.

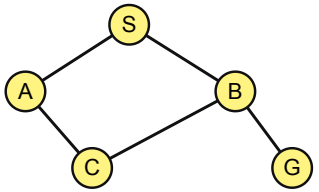
TRANSFER MATRIX

CHANGE ONE COLUMN	DIAGNOSTIC QUESTION
Task wording	Does paraphrase preserve the response?
Environment state	Does a closed corridor update the answer?
Feedback	Can correction alter later behaviour?
Evaluation set	Does performance survive a shifted case distribution?

20-SECOND ANSWER → *supported behaviour → operating scope → open mechanism question → decisive next test.*

10 · Worked FIFO Trace

C3 WORKED EXAMPLE



Successor order A then B; graph search marks on discovery; goal test is on removal.

REMOVE	FRONTIER AFTER INSERTION	EVENT
S	S–A, S–B	Discover A,B
S–A	S–B, S–A–C	Discover C
S–B	S–A–C, S–B–G	Suppress known C; add G
S–A–C	S–B–G	C removed before G
S–B–G	—	Goal → return path

ANSWER

`expansion/removal sequence includes S,A,B,C,G; returned parent chain = S-B-G.`

Expansion order ≠ returned path. C precedes G because A's expansion put C into the FIFO queue earlier, even though C is not on the final parent chain.

11 · Trace Repair Cards

C3 FAQ + STUDY STRATEGY

IF YOU SEE…	DO THIS
Same state, new path	Apply the declared duplicate/cost policy
Wrong final route	Find first frontier divergence, not last answer
Weighted graph + BFS	Separate edge count from cost
UCS/A* later cheaper route	Lower best g; replace parent; insert new priority
Different reference sequence	Compare ties, neighbour order and goal timing

INVARIANTS WHILE TRACING

- FIFO progresses by depth layers.
- LIFO selects the latest path.
- Priority search removes the lowest valid priority.
- Returned edge sum equals `g(goal)`.

SELF-TEST → *Erase the explored set and follow a cycle: what repeated work appears, and which identity test prevents it?*

BLANK LEDGER TO RECONSTRUCT

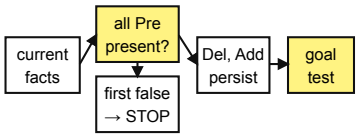
STEP	REMOVED PATH	PRIORITY	GENERATED	DUPLICATE / IMPROVE	FRONTIER AFTER
0	—	—	start	record	[start]
1	_____	_____	_____	_____	_____
2	_____	_____	_____	_____	_____
3	_____	_____	_____	_____	_____

After stopping, write three separate lines: **removed/expanded order**; **returned parent chain**; **actual edge-sum cost**. If a reference differs, compare the convention box before changing the algorithm.

Memory compression drill: replace full stored paths with parent pointers, then reconstruct backwards. This changes the storage representation, not the explored state space or frontier semantics.

12 · Worked Plan Repair

C5 WORKED EXAMPLE



Run the gate before the update; never repair a plan by narrating past an illegal step.

Scenario: one gripper; sample in store; access token in reception; lab door locked. Proposed plan picks up the sample, travels to reception, then tries to pick up the token.

STEP	FACT-SET CONSEQUENCE	VERDICT
pickup_sample	Delete sample-at-store and handempty; add carrying-sample	Legal
move to reception	Update location; preserve carrying-sample	Legal if move preconditions hold
pickup_token	Requires handempty, which is absent	**First illegal action**

Repair: under an explicit separate token-storage fact, collect the token first; then obtain the sample, unlock the door and execute the final drop. Replay every successor and verify `sample-at-lab` in the final state.

ANSWER PATH

`illegal = pickup_token; missing precondition = handempty; cause = pickup_sample deleted it.`

13 · Planning Error Cards

C5 FAQ + STRATEGY

IF YOU SEE…	DO THIS
Variable action schema	Ground every variable before testing
One false precondition	Stop; name it and current state
Delete effect omitted	Look for impossible coexisting facts
Unmentioned fact disappears	Restore it via the frame assumption
All actions legal	Still test every final goal fact
World changes during execution	Monitor and replan from updated facts

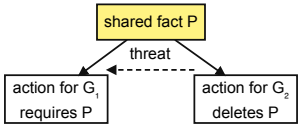
TRAP → *Repair the model and the sequence separately; do not smuggle in hidden storage or capacity.*

FACT LEDGER

ACTION	MISSING PRE?	DEL	ADD	PRESERVED	GOAL NOW?
grounded a_1	_____	_____	_____	_____	_____
grounded a_2	_____	_____	_____	_____	_____

14 · Goal Interaction

C5 TOPIC MAP + FAQ



An action achieving one goal can delete a condition needed for the other.

REPAIR PROCEDURE

1. Write each goal-producing final action.
2. Regress each goal to the facts required immediately before it.
3. Mark shared resources and any delete effect that threatens the other route.
4. Choose an interleaving that preserves prerequisites—or deliberately undo/rebuild them.
5. Validate the proposed repair forward from the initial facts.

Three-crate cue: goals `Red on Blue` and `Blue on Green` share Blue; with a one-crate crane, solving subgoals independently can destroy a precondition. Delete effects expose the dependency.

A repair is complete only after forward replay shows that the later goal remains true without silently changing capacity, storage or action effects.

15 · Method Chooser

ALL GROUNDED CHAPTERS

IF YOU SEE X…	DO Y	FINAL CHECK
Broad capability headline	Observation–mechanism–boundary–judgement	Changed condition is falsifiable
Controller scenario	State, percepts, actions, objective, environment	Every stored detail changes a future
Unweighted graph	Declare order and goal timing; trace FIFO/LIFO	Path ≠ expansion list
Weighted graph	Use g; keep best-cost and parents	Edge sum = g(goal)
Heuristic table	Compute f; test h≤h*	h(goal)=0
Symbolic plan	Ground; Pre gate; Del/Add; preserve	All goal facts true
Two interacting goals	Regress; mark threats; interleave	Replay repair forward

ONE-LINE JUDGEMENT STEMS

Search: "Method M selects ___; under condition ___ this guarantees ___." **Heuristic:** "At node n, h=___ and true remainder=___, so h≤h* [holds/fails]." **Plan:** "Action a first fails because required fact ___ is absent from state ___."

16 · Classic Traps

FAQ + STUDY STRATEGIES

TRAP	WHY WRONG	REPAIR
Broad intelligence claim from one demo	Behaviour does not reveal mechanism or transfer	Bound task + boundary test
Calling poor outcome irrational	Uncertainty separates choice from luck	Judge expected utility with available information
State = location only	May omit load, door, battery, completion	Add variables that change legality/goal/cost
Graph = search tree	Tree nodes are histories; labels can repeat	Carry parent/path explicitly
Apply effects before Pre check	Illegal action has no successor	Gate first; stop on false fact
Drop unmentioned facts	Violates frame assumption	Preserve everything not in Del/Add
Symbolic validity = real reliability	Facts/effects may be stale or incomplete	Monitor, check motion, replan

17 · Exam-Morning Plan

COURSE STUDY STRATEGY

PASS	CLOSED-NOTE SELF-TEST	EVIDENCE OF READINESS
1 · Recite	Define state space, tree, frontier, admissible, grounded action, delete effect	Each definition includes its discriminating clause
2 · Choose	Given task cue, select DFS/BFS/UCS/A*/STRIPS reasoning	Rule + assumption + conclusion
3 · Trace	Redo A*, FIFO and plan-repair examples	Every frontier/fact update visible
4 · Audit	Find one deliberate wrong duplicate or delete step	Identify first divergence
5 · Explain	Say one boundary failure for each model	No guarantee stated without assumptions

60-SECOND FINAL RECALL

`formulate → declare conventions → select by frontier rule → update ledger → test stopping condition → reconstruct route → verify cost/goal → state guarantee + assumption.`

Search ledger: node, parent, depth, g, h, f, status. **Planning ledger:** action, missing Pre, Del, Add, preserved facts, successor, goal truth. **Capability ledger:** observation, mechanism, boundary, judgement.

Do not study logistics from memory. Check the LMS instructions for current time, venue and permitted materials.