CS 660: Topics in AI Planning Algorithms

Prof. Stephen G. Ware

Welcome!

CS 660: Topics in AI: Planning Algorithms

F. Paul Anderson Office Tower, Room 255

Monday, Wednesday, and Friday 2:00 – 2:50 PM

http://cs.uky.edu/~sgware/courses/cs660/index.syl

Instructor

Stephen G. Ware, Ph.D.

Research: AI for interactive narrative and games

Office Hours: M/W/Th, 4:00 - 5:00 PM

Davis Marksbury Building, Room 307

sgware@cs.uky.edu

Text

None! Readings provided on course webpage.

Seminar-Style Course

- Read papers on planning algorithms.
- Submit ~1 page summary before class.
- Presenter presents paper.
- Discussion.

Programming Projects

- Implement 3 styles of planning algorithms.
  - Least commitment planner
  - Graph-based planner
  - State-space planner
- Self-defined final project.

Grading

Assignment	Weight
Paper Summaries	1% each, 30% total
Paper Presentations	25% total
Planning Projects*	15% each, 30% total
Final Project	15%

* Lowest project dropped

Policies

- **Attendance:** Required for a seminar. I will enforce it if people stop coming.
- **Due Dates:** Midnight, but submissions close at 3:00 AM (unofficial grace period).
- **No Curve:** Grades will not be curved.
- **No Individual Bonus:** Bonus available on all projects, but no special assignments for individuals.

Academic Integrity

- Abide by UK's academic integrity policy.
- Study groups for homework and projects OK.
- Assistance from Generative ML (Chat-GPT) for homework or projects is OK.
- Copying answers or code from any source, human or digital, is not OK.

Students with Disabilities

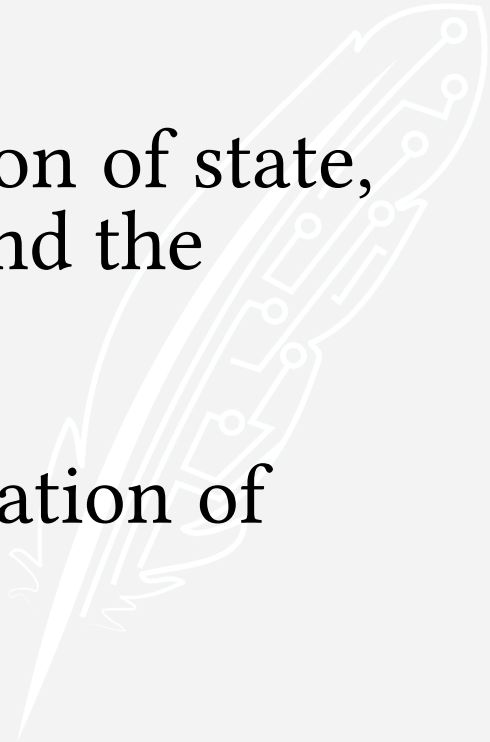
- Arrangements made on a personal basis.
- Please contact me and the UK Disability Resources Center.

Planning

Planning is reasoning about a sequence of actions that achieves some goal.

Planning uses a simple representation of state, action, and goal to deal with time and the frame problem.

Planning uses a logic-like representation of states and actions to allow domain-independent heuristics.



Planning Problem

Given:

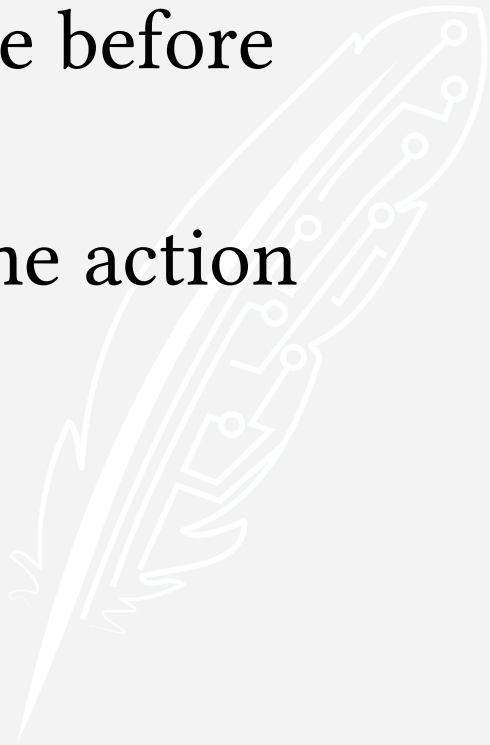
1. A description of the world in the initial state
2. A set of action templates
3. A goal

Find a sequence of ground actions which, when taken from the initial state, achieves the goal.

Actions

Actions have:

- **preconditions** which must be true before the action can be taken
- **effects** which become true after the action has been taken



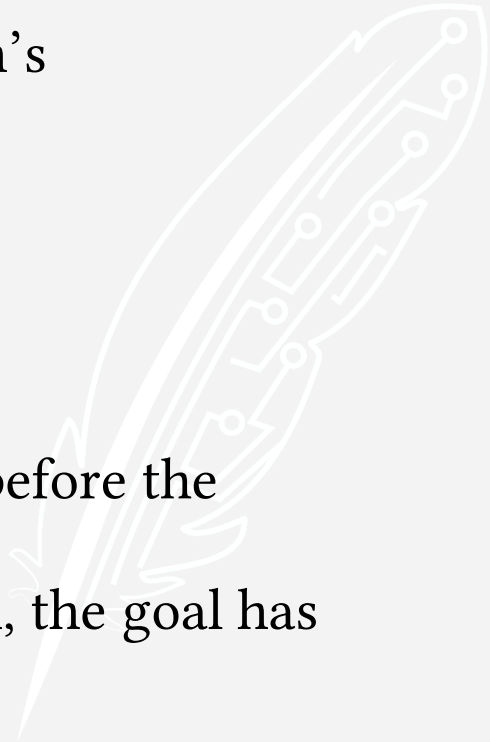
Planning Problem

Given:

1. A description of the world in the initial state
2. A set of action templates describing each action's preconditions and effects
3. A goal

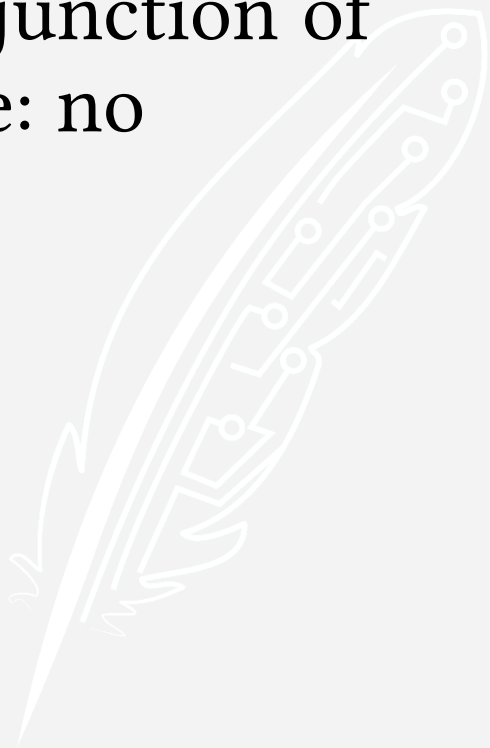
Find:

1. A sequence of ground actions
2. Such that each action's preconditions are true before the action is taken
3. And such that, after all actions have been taken, the goal has been achieved



Logical Language

The initial state, goal, preconditions, and effects are all described using a conjunction of function-free predicate literals (note: no quantifiers).



Cargo Domain

- The world consists of cargo, airplanes, and airports.
- The initial state specifies where each plane and each cargo is.
- There are three actions:
  - Load cargo onto a plane at the same airport.
  - Unload cargo from a plane to the current airport.
  - Fly a plane (and its cargo) to another airport.

Cargo Domain Action Templates

Action: $load(Cargo\ c, Plane\ p, Airport\ a)$

Precondition: $at(c, a) \wedge at(p, a)$

Effect: $in(c, p) \wedge \neg at(c, a)$

Action: $unload(Cargo\ c, Plane\ p, Airport\ a)$

Precondition: $in(c, p) \wedge at(p, a)$

Effect: $at(c, a) \wedge \neg in(c, p)$

Action: $fly(Plane\ p, Airport\ from, Airport\ to)$

Precondition: $at(p, from)$

Effect: $at(p, to) \wedge \neg at(p, from)$

Cargo Problem

Constants:

- 2 Planes, P1 and P2.
- 2 Cargos, C1 and C2.
- 2 Airports, MSY and ATL.

Initial State: $at(P1, ATL) \wedge at(P2, ATL) \wedge at(C1, ATL) \wedge at(C2, ATL)$

Goal: $at(C1, MSY)$

Cargo Problem

Initial State: $at(P1, ATL) \wedge at(P2, ATL) \wedge at(C1, ATL) \wedge at(C2, ATL)$

Goal: $at(C1, MSY)$

Solution:

1. $load(C1, P1, ATL)$
2. $fly(P1, ATL, MSY)$
3. $unload(C1, P1, MSY)$

What Planning Is

- Logic
- Search
- Heuristic Design

What Planning Isn't

- Machine Learning
- Reinforcement Learning