Don't panic!

Bring a photo ID, a writing instrument, and an erasing instrument. You may bring any written material, but nothing else is allowed. Store anything disallowed in the front of the room, and display your photo ID on your desk. Turn off all electronic devices: mobile phones, pagers, and so on.

Do not open the exam until the instructor tells you to. After you open the exam, make sure that you have all 2 pages. Write your name and RUID# on each page. In addition, we also provide some pages of Russell and Norvig's textbook.

The exam is 90 minutes long. You can only leave early during the first 80 minutes, and cannot return until the exam is over. Stop writing as soon as the instructor tells you to. Talking is not allowed—if you have a question, raise your hand.

Read the questions carefully, then circle the single best answer in each non-empty box, and write a single best answer in each empty box. Be brief. Each box is worth the same amount for grading, so don't spend too much time on one box! Feel free to use the reverse side of these sheets as scratch paper; it will not be graded.

**Part 0: The language of planning problems.** Read the first 3 paragraphs under "The language of planning problems" on page 377.

(1) If there are $n$ propositional literals, including *Rich* and *Famous*, and no (other) first-order literals, then there are a total of $\boxed{n \quad n^2 \quad 2^n}$ states up to logical equivalence. The goal $\mathit{Rich} \wedge \mathit{Famous}$ would then pick out $\boxed{\phantom{xxxxxx}}$ of these states as goal states.

(2) Give a set of states that cannot be picked out by this representation of goals.

$\boxed{\phantom{xxxxxxxxxxxxxxxxxxxx}}$

Sketch one way to extend the representation to a richer language of goals that can pick out this set of states.

$\boxed{\phantom{xxxxxxxxxxxxxxxxxxxx}}$

tion breaks down because working on one subgoal is likely to undo another subgoal. These interactions among subgoals are what makes puzzles (like the 8-puzzle) puzzling.

## The language of planning problems

The preceding discussion suggests that the representation of planning problems—states, actions, and goals—should make it possible for planning algorithms to take advantage of the logical structure of the problem. The key is to find a language that is expressive enough to describe a wide variety of problems, but restrictive enough to allow efficient algorithms to operate over it. In this section, we first outline the basic representation language of classical planners, known as the STRIPS language.[2] Later, we point out some of the many possible variations in STRIPS-like languages.

**Representation of states.** Planners decompose the world into logical conditions and represent a state as a conjunction of positive literals. We will consider propositional literals; for example, $Poor \land Unknown$ might represent the state of a hapless agent. We will also use first-order literals; for example, $At(Plane_1, Melbourne) \land At(Plane_2, Sydney)$ might represent a state in the package delivery problem. Literals in first-order state descriptions must be **ground** and **function-free**. Literals such as $At(x, y)$ or $At(Father(Fred), Sydney)$ are not allowed. The **closed-world assumption** is used, meaning that any conditions that are not mentioned in a state are assumed false.

**Representation of goals.** A goal is a partially specified state, represented as a conjunction of positive ground literals, such as $Rich \land Famous$ or $At(P_2, Tahiti)$. A propositional state $s$ **satisfies** a goal $g$ if $s$ contains all the atoms in $g$ (and possibly others). For example, the state $Rich \land Famous \land Miserable$ satisfies the goal $Rich \land Famous$.

GOAL SATISFACTION

**Representation of actions.** An action is specified in terms of the preconditions that must hold before it can be executed and the effects that ensue when it is executed. For example, an action for flying a plane from one location to another is:

$$\begin{array}{l} Action(Fly(p, from, to), \\ \quad \text{PRECOND:} At(p, from) \land Plane(p) \land Airport(from) \land Airport(to) \\ \quad \text{EFFECT:} \neg At(p, from) \land At(p, to)) \end{array}$$

ACTION SCHEMA

This is more properly called an **action schema**, meaning that it represents a number of different actions that can be derived by instantiating the variables $p$, $from$, and $to$ to different constants. In general, an action schema consists of three parts:

- The action name and parameter list—for example, $Fly(p, from, to)$—serves to identify the action.
- The **precondition** is a conjunction of function-free positive literals stating what must be true in a state before the action can be executed. Any variables in the precondition must also appear in the action's parameter list.
- The **effect** is a conjunction of function-free literals describing how the state changes when the action is executed. A positive literal $P$ in the effect is asserted to be true in

PRECONDITION

EFFECT

[2] STRIPS stands for STanford Research Institute Problem Solver.