Planning Agents

Chapter Overview

- planning problems
  - from problem solving to planning

- representations for planning problems
  - states, goals, actions, plans

- partial-order planning
  - keep the number of plans tractable

- abstract examples
  - shopping, blocks world, Shakey’s world

- practical planning
  - hierarchical decomposition, operators, resource constraints

- real-world applications
  - space missions and spacecrafts, job shop scheduling
Search-Based Problem Solver

review from earlier chapter

actions
represented by programs that generate successor state descriptions

states
complete state descriptions are required

goals
goal test, heuristic functions as black boxes

plans
a solution is a sequence of actions
search algorithm generates only contiguous sequences
Planning Problems

from problem solving to planning

reasoning process
structured more flexibly
any part of the problem can be worked on

planning and execution
no necessary connection between the order of
planning and the order of execution

decisions
important or obvious decisions can be made first

hierarchical decomposition
divide-and-conquer strategy
a plan is split up into largely independent
subplans
hierarchical decomposition only works if the sub-problems are independent

Counterexample: Eight-puzzle, where the goal consists of interdependent subgoals
Representations

for planning problems

states
the world is described through logical conditions

goals
conjunctions of literals, possibly with variables

actions
described via operators, with preconditions and effects

plans
sequences of actions

This representation is close to the STRIPS language, one of the first planning systems
Example: [?] p. 343
States

the world decomposed into logical conditions

**specification**
- conjunctions of function-free ground literals:
  - predicates applied to possibly negated constant symbols
  - no functions, no variables

**completeness**
- state descriptions may be incomplete

**closed-world assumption**
- any conditions not explicitly mentioned are assumed to be false

**Examples**
- Lost & Stuck for a disoriented, immobile agent
- \((\text{At (Truck-1, SLO)} & \text{At (Truck-2, SF)})\) for a truck scheduling problem
restricted expressiveness achieved better computational efficiency
Goals

partially specified state that satisfies some condition

specification
conjunction of positive ground literals

goal satisfaction
a state \( s \) satisfies a goal \( g \) if the states contains all the atoms in \( g \), and possibly others

Example Lost & Stuck & Out-Of-Fuel & Tired satisfies the (undesirable) goal Lost & Stuck
Actions

preconditions must hold before the operator is applied, and the effects are the expected outcome

precondition
conjunction of function-free positive literals
state what must be true in a state before an action can be executed

operator
describes the operations to be executed in order to achieve the expected outcome

effects
conjunction of function-free literals
state what is expected to be true after the action is executed (the operator is applied)
for better readability, an add list is used for positive literals,
variables in the precondition and effects must also appear in the parameter list of the operator

Franz J. Kurfess, Cal Poly SLO
Action Schema Example

drive a truck from one location to another

Action(Drive(t, from, to),
    PRECOND: At(t, from) AND Truck(t) AND Location(from) AND Location(to)
    EFFECT: At(t, to) AND NOT At(t, from))
Plan

a sequence of actions

modification of states
  positive literals that appear in the effect of an
  action are added to the modified state, and
  negative literals are removed

common assumption sometimes called Strips assumption
  literals not mentioned in the effect remain
  unchanged

applicable actions
  can be performed in any state that meets the
  precondition

solution for a planning problem
  plan that specifies actions leading from an
  initial state to a goal state
Operators

descriptions of actions

description
  name for an action

precondition
  conjunction of atoms that must be true

effect
  conjunction of literals describing the changed situation

Graphical representation
  box for the action
  preconditions above, effects below
Situation Space

traversed in order to reach the goal

progression planning
  - searches forward from initial to goal situation
  - often inefficient due to high branching factor
  - and huge state space

regression planning
  - searches backward from goal to initial situation
  - possible because only partial descriptions of states are needed
  - complicated for conjunctions of goals
Partial Plan

simple, incomplete plan

operators
   work on plans: add steps, impose ordering,
   instantiate variables, . . .

refinement operators
   constraints are added to a partial plan
   equivalent to the elimination of possible plans

modification operators
   plans are modified
   incorrect plans can be “debugged”
Partial-Order Planning

keep the search focused

partial order  
leave some ordering decisions open

total order  
sequential list of steps, or linearization of a plan
Solution

executable plan that achieves the goal

**complete**
- every precondition of every step is satisfied

**consistent**
- no contradictions in the ordering or binding constraints
Plans

important aspects

plan steps
  each step is one of the operators for the problem

ordering constraints
  temporal order of the steps

variable binding constraints
  no conflicts in instantiations

causal links
  record the purpose of steps

graphical notation: boxes and arrows
Shopping

as an abstract planning problem

initial plan
start situation, goal situation

partial plan
insert steps that can be resolved right away

partial order plan
don’t worry about the particular sequence of steps

solution
complete plan with all necessary ordering and binding constraints

see [?], pp. 349 ff
Truck Delivery

simplified practical planning problem

Init(At(C1, SLO) AND At(C2, SF) AND C3, SB) AND C4 (AG) AND
At(T1, SLO) AND At(T2, SF) AND
Cargo(C1) AND Cargo(C2) AND
Truck(T1) AND Truck(T2) AND
Location(SLO) AND Location(SF) AND Location(SB) AND Location(AG))

Goal(At(C1, SF) AND At (C2, SB))

Action(Load(c, t, l)
PRECOND: At(c, l) AND At(t, l) AND Cargo(c) AND Truck(t) AND Location(l)
EFFECT: On(c, t) AND NOT At(c, l)

Action(Unload(c, t, l)
PRECOND: On(c, t) AND At(t, l) AND Cargo(c) AND Truck(t) AND Location(l)
EFFECT: NOT On(c, t) AND At(c, l)

Action(Drive(t, from, to)
PRECOND: AND At(t, from) AND Truck(t) AND Location(from) AND Location(to)
EFFECT: NOT At(t, from) AND At(t, to)

simplified STRIPS program
Blocks World

states
- objects and their positions

goals
- particular spatial relations between objects

actions
- operators for moving blocks

plans
- sequences of block movements
Blocks World in STRIPS

Init(On(A, Table) AND On(B, Table) AND On(C, Table) AND Block(A) AND Block(B) AND Block(C) AND Clear(A) AND Clear(B) AND Clear(C))

Goal(On(A, B) AND On(B, C))

Action(Move(b, x, y),
    PRECOND: On(b, x) AND Clear(b) AND Clear(y) AND Block(b) AND NEQ(B, x) AND NEQ(b, y) AND NEQ(x, y),
    EFFECT: On(B, y) AND Clear(x) AND NOT On(b, x) AND NOT Clear(y))

Action(MoveToTable(b, x)
    PRECOND: On(b, x) AND Clear(b) AND Block(b) AND NEQ(b, x)
    EFFECT: On(b, Table) AND Clear(x) AND NOT On(b, x))

simplified STRIPS program
Practical Planners

operate in complex, realistic domains

planning methods
language and algorithms must be extended

search
must be focused for specific domains

real-world limitations
resources, time, uncertainty
Hierarchical Decomposition

different levels of abstraction

abstract operators
can be decomposed into a group of steps

primitive operator
can be directly executed
### Language Extensions

**operators** classified into
- primitive
- nonprimitive

**decomposition methods**
- similar to subroutines or macros for operators

**Action Description Language (ADL)** more
- expressive than Strips
- allows positive and negative literals in states
- open world assumption
- also disjunctions, quantified variables in goals
- built-in equality
- types

**Planning Domain Description Language (PDDL)**
- standard syntax for various planning formalisms
- includes sublanguages for Strips, PDDL, . . .

requires modification of the planner
Resource Constraints

representation and execution

resources
  can be produced and consumed

measures
  numeric values for quantifying resources

temporal constraints
  time is just another resource
Distributed Problem Solving

collaboration among agents to achieve a common goal

example problems
tasks that seem suitable for distribution

task sharing
one agent offloads some of his tasks onto other agents

result sharing
several agents work on the same task, and their results are combined
Distributed Planning

specialization of distributed problem solving

distributed formulation of plans
  the planning process itself is distributed

generation of plans for distributed activities
  the plan is generated in such a way that the
  activities it specifies can be executed in a
  distributed way

distributed plan representation
  methods for representing distributed plans in a
  coordinated manner

distributed execution of plans
  combining coordination, planning, and execution
Centralized Planning

for distributed plans

partial order planning
no strict ordering is required between actions
these actions may be executed in parallel via
threads

decomposition of a plan into subgoals
subplans should be self-contained

synchronization between subplans
frequently via communication

subplan allocation
different subplans are allocated to individual
agents
can become complex for heterogeneous agents

plan execution
individual agents execute their subplans
may involve monitoring by the centralized
Distributed Planning

for centralized plans

cooporative planning
  several agents work on the same plan
  mostly interesting for very large or complex plans
  variation of distributed problem solving where the problem happens to be a planning task

task decomposition
  identification of largely independent subtasks that can be tackled by individual agents or teams of agents

task distribution
  allocation of subtasks to (teams of) agents

subtask execution
  individual agents work on their specific tasks

result sharing
  contributions by individual agents are collected
and synthesized into one comprehensive plan
Distributed Planning

for distributed plans

planning process and plan execution are distributed
combines challenges from both approaches

relatively immature field
many different approaches, but not much systematicity
Plan Merging

coordination of multiple individual plans

inherently distributed task
- no central agency to coordinate the planning task
- each agent has its own plan, but they are willing to coordinate their activities

identification of conflicts
- resource utilization
- expected results

resolution of conflicts
- analysis of the individual plans for conflicts
- centralized or distributed approaches
- variation of reachability analysis, which relies on the possibly intractable enumeration of states

identification of unsafe states
- emphasis on actions performed by the agents
- assumes that the “action space” is less complicated than the state space
Distributed Hierarchical Planning

variation of iterative plan formulation

levels of abstraction
agents formulate their plans at different levels
only higher levels are shared
can reduce the overall search space substantially
by pruning away many details

conflicts
it is assumed that conflicts can be recognized
and hopefully resolved at higher levels
is not always the case
Negotiation in distributed planning

resolution of conflicts
once conflicts are discovered, the agents involved negotiate a plan that solves the conflicts
usually based on utility functions for agents may require the revision of plans preserves the autonomy of agents

extension of the planning space
negotiation often results in an even larger state space for the planning problem

self-interested agents
negotiation assumes that agents are willing to cooperate incentives can be introduced to encourage cooperation even for self-interested agents
Distributed Plan Representation

abstract description language for plans

compatibility of plans among different agents
agents may use different planning systems with
various plan representation schemes

plan components
many of the individual components of a plan
may require their own description languages
(environment, agent capabilities, resources,
plots, subplans, . . .)

communication protocols
necessary for the exchange of plans
not sufficient for the description of plans

knowledge exchange languages
in principle capable of representing and
exchanging plans, in practice they may be too
general
Distributed Planning and Execution

pre-planning coordination
may impose constraints on the possible actions
of agents
sometimes formulated as social laws
in general, agents may inform each other about
their plans

post-planning coordination
coordination of plans during execution
agents may formulate contingency plans in
advance, and choose the appropriate branch
during execution

interleaving planning, coordination, execution
continual revision of plans in response to
coordination decisions

observation-based plan coordination
agents that can’t communicate can infer each
other's plans
Summary

Planning Agents

planning problems
from problem solving to planning

representations for planning problems
states, goals, actions, plans

partial-order planning
keep the number of plans tractable

abstract examples
shopping, blocks world, Shakey’s world

practical planning
hierarchical decomposition, operators, resource
constraints

real-world applications
space missions and spacecrafts, job shop
scheduling