Scheduling and Planning

CS410/510CS Lecture 8

Bart Massey <bart@cs.pdx.edu>
PSU CS Department
March 1, 2001

Overview

- Scheduling Problem
- Search In Scheduling
- Planning Problem
- Search In Planning
- Project or Presentation

Scheduling Problem

Find minimum-cost arrangment of a sequence of tasks

- Minimum-cost = shortest time, or might mean eg
  - lowest dollar cost (e.g. amount of workers, amount of equipment, etc)
  - minimum wrt risk (robust scheduling)
- Arrangement = time ordering, or might mean eg
  - Packing into bins
  - Assignment to machines

Tasks

- Atomic unit of scheduling
- Labelled with
  - id
  - start time
  - duration
  - resource consumption
- Task count controls complexity
  - tens: easy (except for MS Project)
  - hundreds: typical
  - thousands-millions: need special techniques

Hierarchical Scheduling

Natural to divide tasks into subtasks (= group tasks into jobs). We mostly do not consider here:

- Hard to define problem
- Hard to solve problem

Schedule: Time To Tasks

For this talk, *scheduling* means

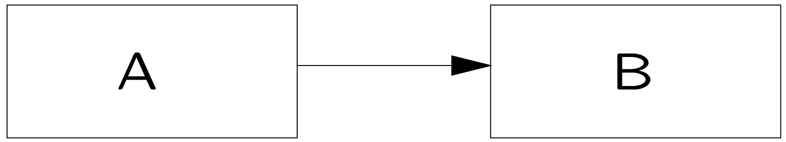
Assigning times to tasks to minimize schedule length ("makespan")

- CSP: var = task, val = time
- Note that val is natural number: how to fix?
- Arbitrary constraints: we look at common cases

Precedences

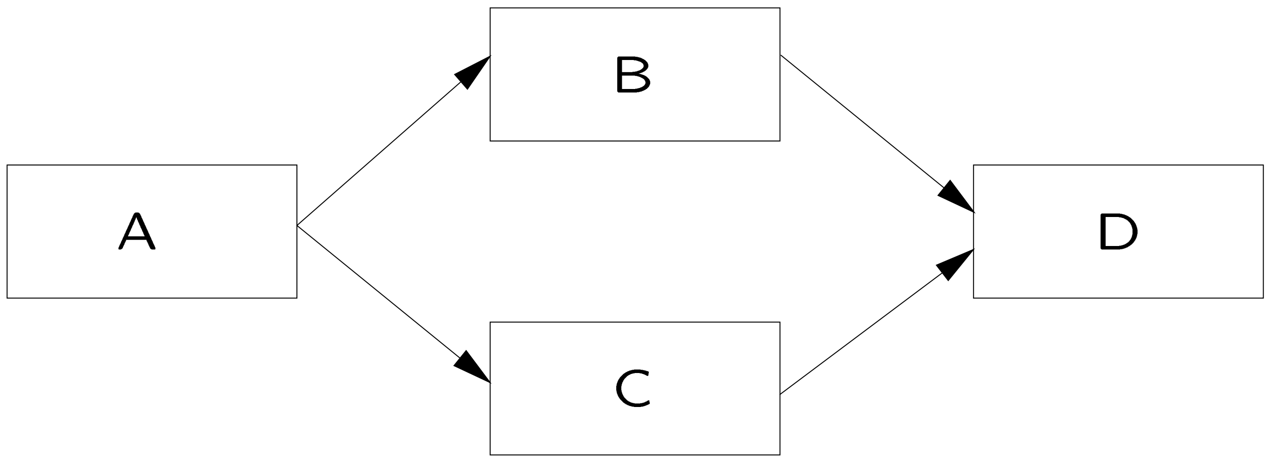
Most common: end-start. If task A must precede task B, then A must finish before B can begin. Other possibilities make sense

- end-end (prepare parts, set glue)
- start-start (crush rock, pave driveway)
- given delay (paint part, install part)



Parallel Tasks

Can typically do tasks in parallel unless precedence (or other) constraints dictate sequencing. Scheduling algorithm: topological sort



Resource Capacity

Resource is external requirement of (= constraint on) task. Capacity resources include

- People
- Equipment
- Space
- Power

Capacity resources may be complicated (eg person can do more than one job, but not in parallel). We consider multiple, simple, non-interacting resources (Resource Constrained Project Scheduling)

Multiple Resources

Multiple resources with fixed capacity = bin packing. So problem may be NPC without precedences

A	
	3
	2
	4

res p capacity 4
res q capacity 6
res r capacity 7

B	
	1
	4
	5

C	
	1
	1
	2

Consumable Resources

Can also have consumable resources that are

- Supplied at specified times in instance
- Supplied by tasks
- “Used up” by tasks

Examples are obvious: parts, ingredients, etc.

Allowing insertion of new actions to supply consumable resource takes problem from scheduling to planning: *much* harder!

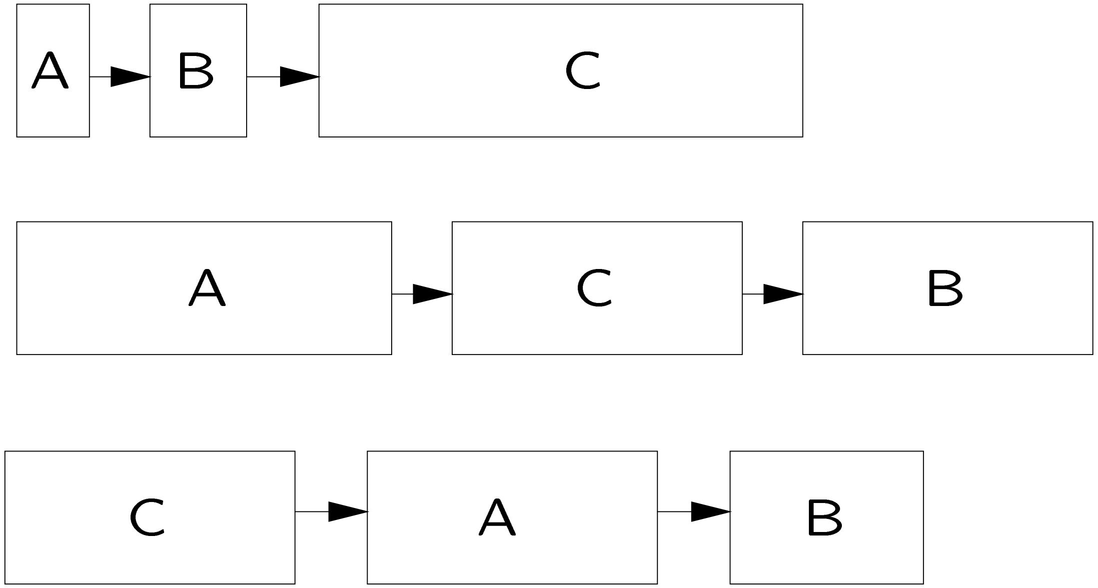
Job Shop Scheduling

OR literature full of special-case problems. Many have poly-time solutions using extreme cleverness

Job-shop problem is

- Strings of tasks (jobs)
- Machines (unit capacity resource), with # machines = # jobs
- Each job has exactly one task on each machine

A Job Shop Problem



Job shop scheduling is NPC

Search In Scheduling

Can see that search is required

- Problems are NPC
- Time assignment is constrained in standard ways

Problems

- Constraint Optimization Problem
- Good search spaces
- Arbitrary times

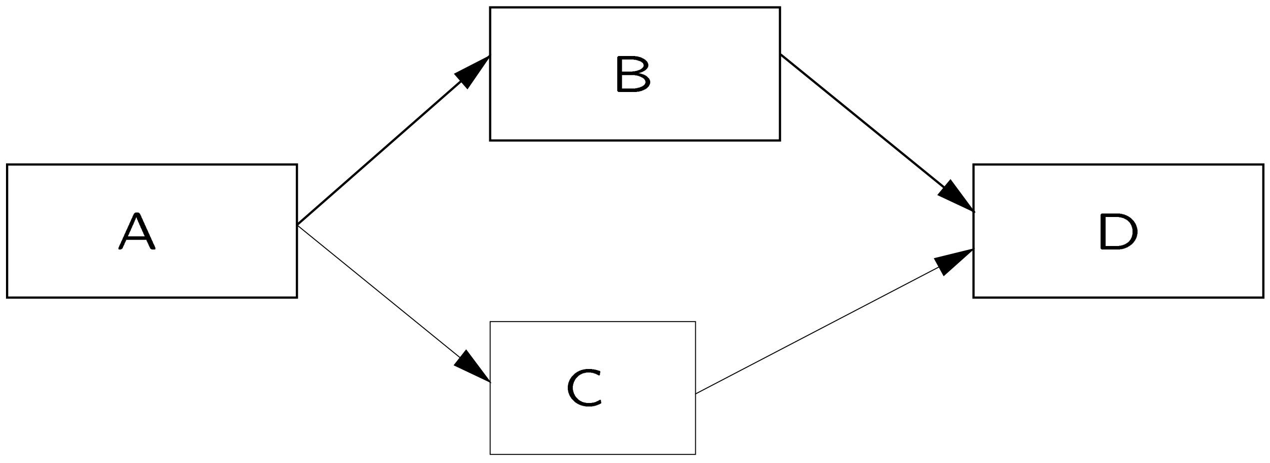
Task Start Variables

Obvious approach: do *not* allow arbitrary start times

- Choose to start task at earliest possible time, or delay
- Binarizes decisions
- “Left-shifted” or “Left-justified” schedule
- Right-shifted schedule may also make sense

PERT and CPM Scheduling

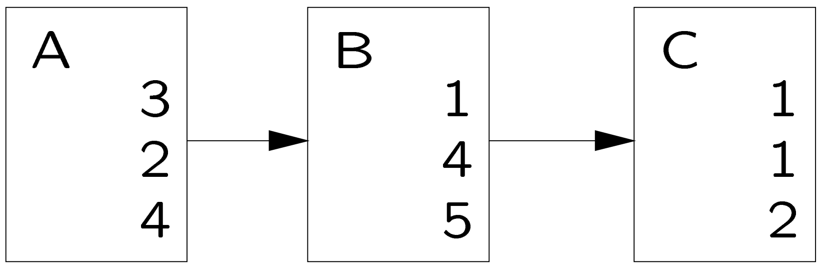
- PERT formulation: probabilistic task duration, best expected schedule
- Critical Path Method: fixed task duration
- Why "critical path"? Identifies tasks which fix schedule bounds



Precedence-Based Scheduling

In presence of resource constraints, CPM topo sort fails. Need alternative search space

Idea: Add artificial precedences to break resource bottlenecks, CPM schedule. "Precedence-based scheduling" (But which way should precedence point?)



res p capacity 4
res q capacity 6
res r capacity 7

Complete Search Algorithms

Can use standard techniques in precedence space:

- LDS
- Branch-and-bound

Heuristics use information like

- tail: what is end time of tasks succeeding this one?
- head: what is end time of tasks preceding this one?
- slack: how much dead time is introduced in schedule?
- bottleneck: is resource overpressured?

Stochastic Search Algorithms

Can also do flips (with obvious meaning in precedence space), or even stranger things:

"Doubleback Optimization" / "Schedule Packing" (Crawford & Fox 1994): Alternately right-shift and left-shift schedule in topo sort order, packing resources. Shifts tasks toward start, end

Adding small number of noise moves (out-of-order shifts) improves performance greatly

Aircraft Assembly Problem

Extensive work on single instance (CIRL 1994–1998)

- Former McDonnell-Douglas problem data
- Standard benchmark: 8 instances (CIRL mostly instance 4)
- 575 tasks, 14 resources
- Labor resources have non-static profile ("shifts")
- "Aircraft assembly", but sanitized data (common practice)

Aircraft Assembly Problem: Results

- MS Project: 80 days (really really hopeless)
- State of art mid-90s: 45-50 days
- CIRL best general purpose: 30 days
- CIRL best handcrafted: 28 days

Runtime in seconds to minutes: anytime, realtime full resched

Planning Problem

General Planning Problem probably single most important AI problem:

- Very human-characteristic
- Very practically important

Idea: Given *initial conditions* and *goal conditions*, description of *world* and possible *actions*, construct plan = schedule of selected actions from initial to goal

Initial and Goal Conditions

Initial condition usually complete state of world at start

- World description itself hard AI problem
- Big problem: framing

Goal usually desired partial state of world: describes things to accomplish, do not care about other things

Prop logic is good language for initial, goal description, world description

STRIPS Actions

STRIPS (Fikes and Nilsson, 197x): make actions

- Atomic, Instantaneous, Sequential
- Precondition: logical formula which must be true to take action
- Effect: list of logical variables/formulae which change state when action is taken
  - Add List: variables which are forced true
  - Delete List: variables which are forced false

First-order logic is confusing: use prop logic instead

The STRIPS Assumption

Big problem with logic world: actions must specify what changes, but also what *does not change*

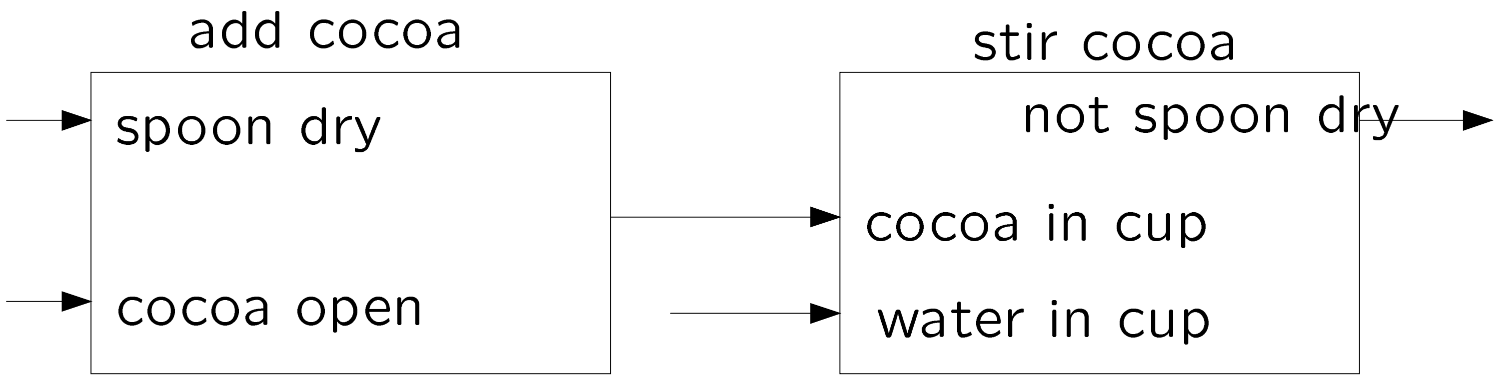
STRIPS assumption (persistence): variables change only when changed explicitly by an action

- Breaks ability to plan with theorem prover
- Makes search tractable
- Captures intuition

Planning: Action Chaining

Goal: find schedule of actions such that

- Each action precondition true
- In particular, initial state allows first actions
- After all actions, goal conditions achieved



Action Selection and Scheduling

Planning is

- Action selection: what is needed to accomplish goal?
- Action scheduling: how to sequence selected actions?

Can do action selection first, or (better) interleave action selection with scheduling

Search In Planning

Questions for planning algorithms

- How to interleave selection and scheduling?
- Whether to search for shortest plans?
- Variable ordering: what part of plan first?
- Value ordering: what actions to try first?

State-Space Forward Search

May be simplest planning algorithm:

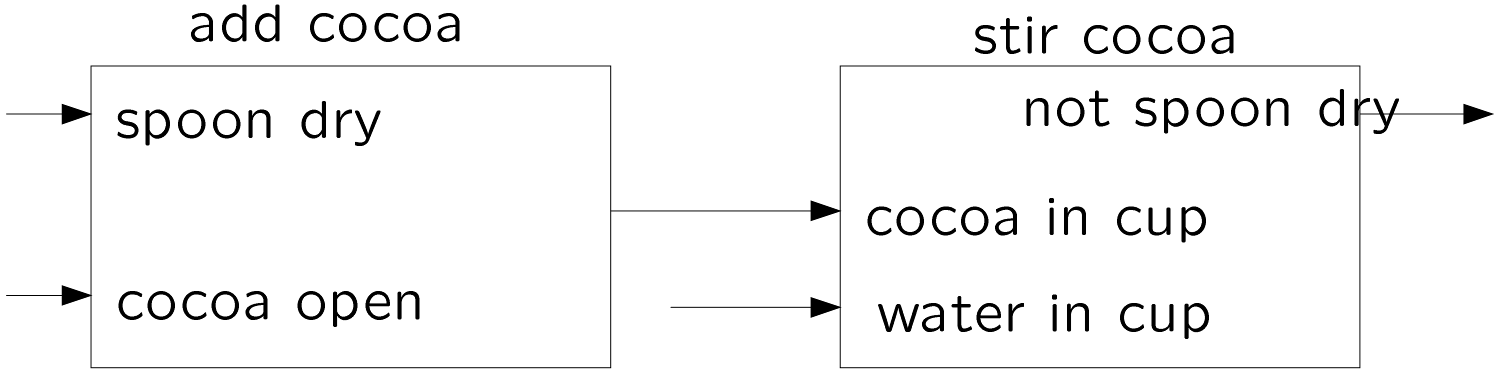
1. Select legal action in initial state
2. Place it at start of plan
3. Select legal action in resulting new state
4. Place it at end of plan
5. Repeat steps 3 and 4 until
   - goal state achieved
   - stuck

Can do various things about being stuck. Blind/dumb DFS is traditional search mechanism

Goal Regression

Instead of initial $\longrightarrow$ goal, other way around?

Problem: may lose determinism. Soln: no big deal



Feature: narrow goal conditions means narrow choice on first few steps (no big deal)

Partial Order Causal Link

Idea: Use search space of *legal plans*!

- Add actions one at a time
- Keep actions schedulable, not scheduled
- Use *causal link* to track "reason" for action

Good idea: doesn't work very well in practice. UCPOP most famous POCL planner, pretty hopeless

Graphplan

Idea: Use advanced static pruning to allow narrow search

Graphplan (Blum & Furst 199x) uses clever static pruning trick with forward planner: reawakened interest in planning

Focused attention on representation + search as tools for planner efficiency

Planning As Satisfiability

Idea: fix plan length, transform planning instance into SAT instance (PSPACE $\longrightarrow$ NP)

SATPLAN, Blackbox (Kautz and Selman): quite successful on large class of instances

Limitations:

- Size of instance + length of plan
- Slight subset of PROP STRIPS

Modern State-Space Forward Search

Research has come full-circle: use modern search techniques with forward state-space planner

- Ordering heuristics
- Admissibility / A* / IDA*
- Stochastic methods

Planning is still very difficult: no great general-purpose planners yet

Giving Up

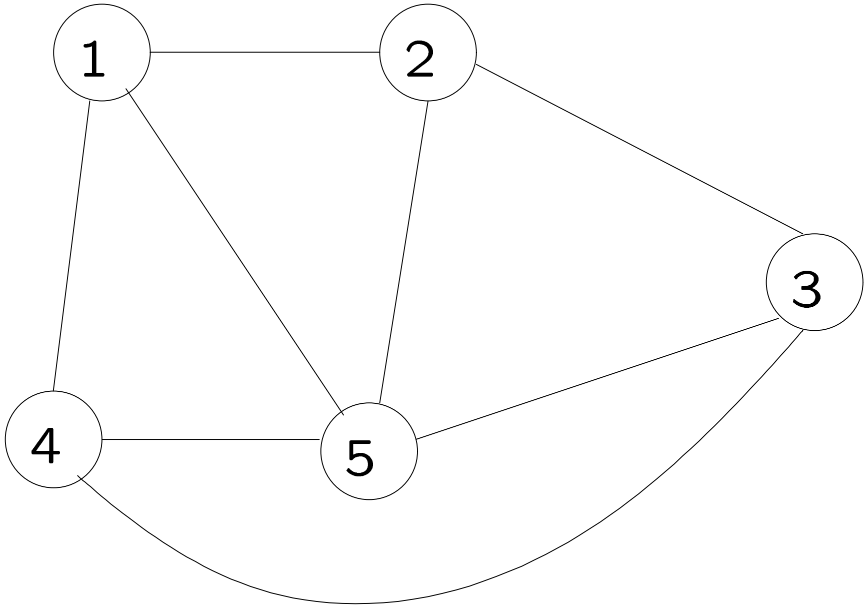
Still need to solve planning instances. One approach: instance-specific heuristics

Lose declarativity, flexibility; gain tractability. OPLAN example of successful guided planner

Note that pathfinding is really special case of planning: approaches are often related

Graph Coloring

Instance: graph, k colors. Question: can color all nodes of graph leaving no edge monochrome?



DIMACS Input Format

```
c These lines are comments
c They are fairly freeform
c Next line gives number of vertices, edges
c Following lines are edges
p edge 5 8
e 1 2
e 1 4
e 1 5
e 2 3
e 2 5
e 3 4
e 3 5
e 4 5
```

DIMACS Output Format

```
s col 4
l 1 1
l 2 2
l 3 3
l 4 4
l 5 2

b 3
```

K-Coloring: Limit vs. Optimum

Two general questions:

- Exists k-coloring for given k
- Smallest coloring

Latter required here

Required: complete method, good method (need not be same)

Speed/Performance Tradeoff

Can go for

- Fast sloppy coloring
- Slow good coloring

Do whatever shows off algorithm best. Remember, judging is subjective

Where To Find Papers

Obvious sources for presentation papers:

- Bibliography of things I have already given out
- Web sites with CS bibliography
- Ask me for ideas (*not* titles!)
- Things I have done in past classes
- Things you are doing at work, in other classes, for fun

20-Minute Talk

20 Minutes (+ questions): not very long!

- Max 12 slides
- Slides should give ideas, not details
- Why is this paper cool? Why is it lame?
- Schedule 0.5 hour meeting with me for week after next to discuss presentation