Lecture slides for

Automated Planning: Theory and Practice

Chapter 3
Complexity of Classical Planning

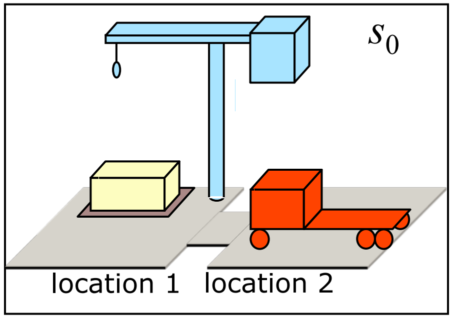
Dana S. Nau

University of Maryland

1:19 PM January 30, 2012

Motivation

- Recall that in classical planning, even simple problems can have huge search spaces
  - Example:
    - DWR with five locations, three piles, three robots, 100 containers
    - 10^{277} states
    - About 10^{190} times as many states as there are particles in universe



- How difficult is it to solve classical planning problems?
- The answer depends on which representation scheme we use
  - Classical, set-theoretic, state-variable

Outline

- Background on complexity analysis
- Restrictions (and a few generalizations) of classical planning
- Decidability and undecidability
- Tables of complexity results
    - Classical representation
    - Set-theoretic representation
    - State-variable representation

Complexity Analysis

- Complexity analyses are done on *decision problems* or *language-recognition problems*
  - Problems that have yes-or-no answers

- A language is a set L of strings over some alphabet A
  - Recognition procedure for L
    - A procedure $R(x)$ that returns "yes" iff the string x is in L
    - If x is not in L, then $R(x)$ may return "no" or may fail to terminate

- Translate classical planning into a language-recognition problem
- Examine the language-recognition problem's complexity

Planning as a Language-Recognition Problem

- Consider the following two languages:

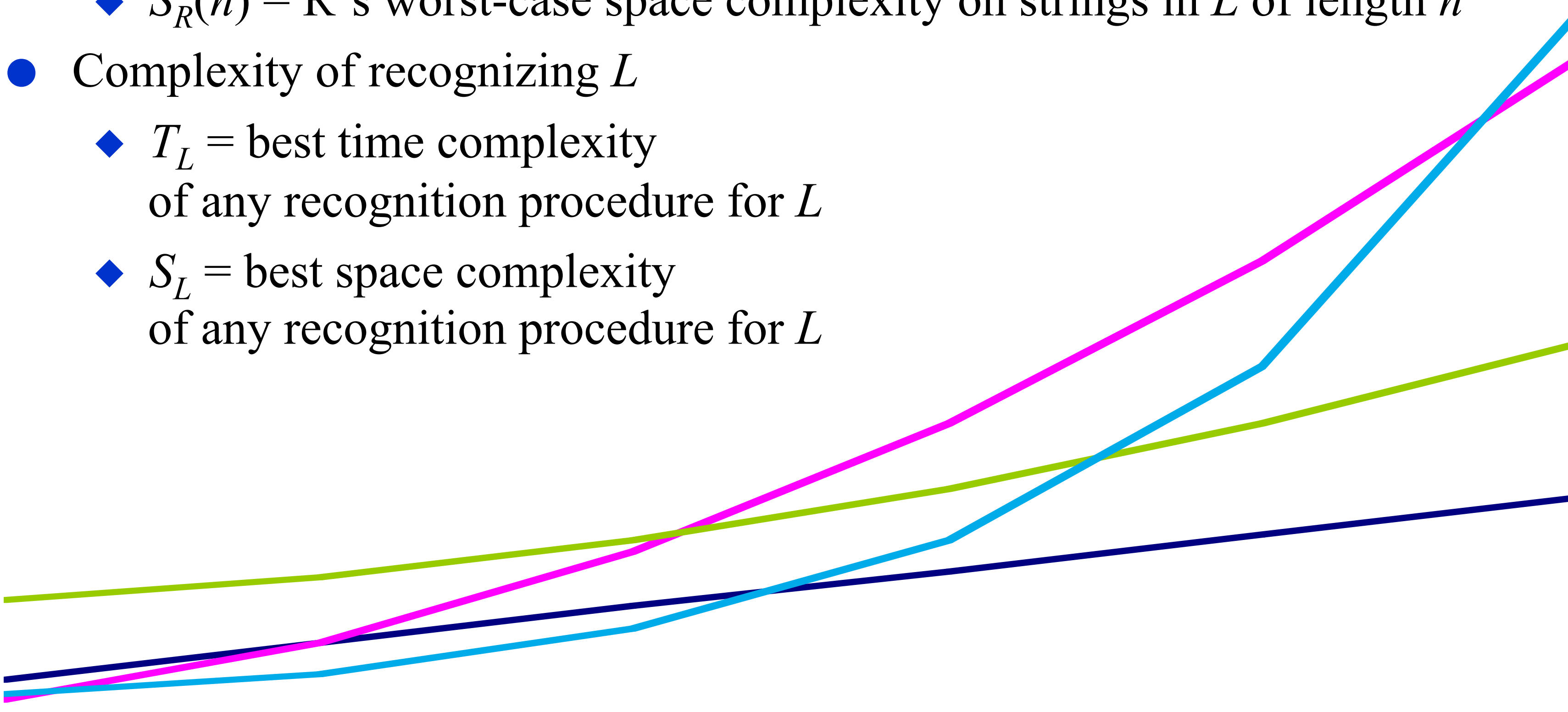
  PLAN-EXISTENCE = $\{P : P$ is the statement of a planning problem that has a solution$\}$

  PLAN-LENGTH = $\{(P,n) : P$ is the statement of a planning problem that has a solution of length $\leq n\}$

- Look at complexity of recognizing PLAN-EXISTENCE and PLAN-LENGTH under different conditions
  - Classical, set-theoretic, and state-variable representations
  - Various restrictions and extensions on the kinds of operators we allow

Complexity of Language-Recognition Problems

- Suppose R is a recognition procedure for a language L
- Complexity of R
  - $T_R(n)$ = R's worst-case time complexity on strings in L of length n
  - $S_R(n)$ = R's worst-case space complexity on strings in L of length n
- Complexity of recognizing L
  - T_L = best time complexity of any recognition procedure for L
  - S_L = best space complexity of any recognition procedure for L



Complexity Classes

- Complexity classes:
  - NLOGSPACE (nondeterministic procedure, logarithmic space)
    $\subseteq$ P (deterministic procedure, polynomial time)
    $\subseteq$ NP (nondeterministic procedure, polynomial time)
    $\subseteq$ PSPACE (deterministic procedure, polynomial space)
    $\subseteq$ EXPTIME (deterministic procedure, exponential time)
    $\subseteq$ NEXPTIME (nondeterministic procedure, exponential time)
    $\subseteq$ EXPSPACE (deterministic procedure, exponential space)
- Let C be a complexity class and L be a language
  - L is C-hard if for every language $L' \in C$, L' can be reduced to L in a polynomial amount of time
    - » NP-hard, PSPACE-hard, etc.
  - L is C-complete if L is C-hard and $L \in C$
    - » NP-complete, PSPACE-complete, etc.

Possible Conditions

- Do we give the operators as input to the planning algorithm, or fix them in advance?
- Do we allow infinite initial states?
- Do we allow function symbols?
- Do we allow negative effects?
- Do we allow negative preconditions?
- Do we allow more than one precondition?
- Do we allow operators to have conditional effects?*
  - i.e., effects that only occur when additional preconditions are true

These take us outside classical planning

Decidability of Planning

Halting problem

Allow function symbols?	Decidability of PLAN-EXISTENCE	Decidability of PLAN-LENGTH
no[α]	decidable	decidable
yes	semidecidable[β]	decidable

[α]This is ordinary classical planning.
[β]True even if we make several restrictions (see text).

Can cut off the search at every path of length *n*

Next: analyze complexity for the decidable cases

- In this case, can write domain-specific algorithms
  - e.g., DWR and Blocks World: PLAN-EXISTENCE is in P and PLAN-LENGTH is NP-complete

Kind of representation	How the operators are given	Allow negative effects?	Allow negative preconditions?	Complexity of PLAN-EXISTENCE	Complexity of PLAN-LENGTH
classical rep.	in the input	yes	yes/no	EXPSPACE-complete	NEXPTIME-complete
		no	yes	NEXPTIME-complete	NEXPTIME-complete
			no	EXPTIME-complete	NEXPTIME-complete
			no $^{\alpha}$	PSPACE-complete	PSPACE-complete
	in advance	yes	yes/no	PSPACE $^{\gamma}$	PSPACE $^{\gamma}$
		no	yes	NP $^{\gamma}$	NP $^{\gamma}$
			no	P	NP $^{\gamma}$
			no $^{\alpha}$	NLOGSPACE	NP

$^{\alpha}$ no operator has >1 precondition

$^{\gamma}$ PSPACE-complete or NP-complete for some sets of operators

- PLAN-LENGTH is never worse than NEXPTIME-complete
  - We can cut off every search path at depth *n*

Kind of representation	How the operators are given	Allow negative effects?	Allow negative preconditions?	Complexity of PLAN-EXISTENCE	Complexity of PLAN-LENGTH
classical rep.	in the input	yes	yes/no	EXPSPACE-complete	NEXPTIME-complete
		no	yes	NEXPTIME-complete	NEXPTIME-complete
			no	EXPTIME-complete	NEXPTIME-complete
			no$^{\alpha}$	PSPACE-complete	PSPACE-complete
	in advance	yes	yes/no	PSPACE $^{\gamma}$	PSPACE $^{\gamma}$
		no	yes	NP $^{\gamma}$	NP $^{\gamma}$
			no	P	NP $^{\gamma}$
			no$^{\alpha}$	NLOGSPACE	NP

Here, PLAN-LENGTH is harder than PLAN-EXISTENCE

Set-Theoretic and Ground Classical

- Set-theoretic representation and ground classical representation are basically identical
  - For both, exponential blowup in the size of the input
  - Thus complexity looks smaller as a function of the input size

Kind of representation	How the operators are given	Allow negative effects?	Allow negative preconditions?	Complexity of PLAN-EXISTENCE	Complexity of PLAN-LENGTH
set-theoretic or ground classical rep.	in the input	yes	yes/no	PSPACE-complete	PSPACE-complete
		no	yes	NP-complete	NP-complete
			no	P	NP-complete
			no$^{\alpha}$/no$^{\beta}$	NLOGSPACE-complete	NP-complete
	in advance	yes/no	yes/no	constant time	constant time

$^{\alpha}$ no operator has >1 precondition

$^{\beta}$ every operator with >1 precondition is the composition of other operators

State-Variable Representation

- Classical and state-variable representations are equivalent, except that some of the restrictions aren't possible in state-variable representations
  - e.g., classical translation of pos(a) ← b
    - » precondition on(a,x)
    - » two effects, one is negative ¬on(a,x), on(a,b)

Like classical rep, but fewer lines in the table

Kind of representation	How the operators are given	Allow negative effects?	Allow negative preconditions?	Complexity of PLAN-EXISTENCE	Complexity of PLAN-LENGTH
state-variable rep.	in the input	yes$^\delta$	yes/no	EXPSPACE-complete	NEXPTIME-complete
	in advance	yes$^\delta$	yes/no	PSPACE $^\gamma$	PSPACE $^\gamma$
ground state-variable rep.	in the input	yes$^\delta$	yes/no	PSPACE-complete	PSPACE-complete
	in advance	yes$^\delta$	yes/no	constant time	constant time

Summary

- If classical planning is extended to allow function symbols
  - Then we can encode arbitrary computations as planning problems
    - » Plan existence is semidecidable
    - » Plan length is decidable
- Ordinary classical planning is quite complex
    - » Plan existence is EXPSPACE-complete
    - » Plan length is NEXPTIME-complete
  - But those are *worst case* results
    - » If we can write domain-specific algorithms, most well-known planning problems are much easier