

Planning and Optimization

B5. Computational Complexity of Planning: Background

Malte Helmert and Gabriele Röger

Universität Basel

October 20, 2016

Motivation

How Difficult is Planning?

- Using **progression** and a state-space search algorithm like breadth-first search, planning can be solved in **polynomial time** in the size of the **transition system** (i.e., the number of states).
- However, the number of states is **exponential** in the number of **state variables**, and hence in general exponential in the size of the input to the planning algorithm.
- ~> Do non-exponential planning algorithms exist?
- ~> What is the precise **computational complexity** of planning?

Why Computational Complexity?

- understand the problem
- know what is not possible
- find interesting subproblems that are easier to solve
- distinguish essential features from syntactic sugar
 - Is STRIPS planning easier than general planning?
 - Is planning for FDR tasks harder than for propositional tasks?

Background: Turing Machines

Reminder: Complexity Theory

Need to Catch Up?

- This and the following section are mostly **reminders**.
- We assume knowledge of complexity theory:
 - languages and decision problems
 - Turing machines: NTMs and DTMs;
polynomial equivalence with other models of computation
 - complexity classes: P and NP
 - polynomial reductions
- If you are not familiar with these topics, we recommend **Parts C and E** of the **Theorie der Informatik** course at <http://informatik.unibas.ch/fs2016/theorie-der-informatik/>.
- The slides are in English, even though the course is not.

Note: the **space complexity classes** (DSPACE, NSPACE, PSPACE, NPSPACE) go beyond the content of the prerequisite course.

Nondeterministic Turing Machines

Definition (Nondeterministic Turing Machine)

A **nondeterministic Turing machine (NTM)** is a 6-tuple $\langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ with the following components:

- **input alphabet** Σ and **blank symbol** $\square \notin \Sigma$
 - alphabets always nonempty and finite
 - **tape alphabet** $\Sigma_{\square} = \Sigma \cup \{\square\}$
- finite set Q of **internal states** with **initial state** $q_0 \in Q$ and **accepting state** $q_Y \in Q$
 - **nonterminal states** $Q' := Q \setminus \{q_Y\}$
- **transition relation** $\delta \subseteq (Q' \times \Sigma_{\square}) \times (Q \times \Sigma_{\square} \times \{-1, +1\})$

Deterministic Turing Machines

Definition (Deterministic Turing Machine)

A **deterministic Turing machine (DTM)** is an NTM where the transition relation is **functional**, i.e., for all $\langle q, a \rangle \in Q' \times \Sigma_{\square}$, there is exactly one triple $\langle q', a', \Delta \rangle$ with $\langle \langle q, a \rangle, \langle q', a', \Delta \rangle \rangle \in \delta$.

Notation: We write $\delta(q, a)$ for the unique triple $\langle q', a', \Delta \rangle$ such that $\langle \langle q, a \rangle, \langle q', a', \Delta \rangle \rangle \in \delta$.

Turing Machine Configurations

Definition (Configuration)

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be an NTM.

A **configuration** of M is a triple $\langle w, q, x \rangle \in \Sigma_{\square}^* \times Q \times \Sigma_{\square}^+$.

- w : tape contents to the left of tape head
- q : current state
- x : tape contents at tape head and to its right

Turing Machine Transitions

Definition (Yields)

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be an NTM.

A configuration c of M **yields** a configuration c' of M , in symbols $c \vdash c'$, as defined by the following rules, where $a, a', b \in \Sigma_{\square}$, $w, x \in \Sigma_{\square}^*$, $q, q' \in Q$ and $\langle \langle q, a \rangle, \langle q', a', \Delta \rangle \rangle \in \delta$:

$$\begin{array}{ll} \langle w, q, ax \rangle \vdash \langle wa', q', x \rangle & \text{if } \Delta = +1, |x| \geq 1 \\ \langle w, q, a \rangle \vdash \langle wa', q', \square \rangle & \text{if } \Delta = +1 \\ \langle wb, q, ax \rangle \vdash \langle w, q', ba'x \rangle & \text{if } \Delta = -1 \\ \langle \varepsilon, q, ax \rangle \vdash \langle \varepsilon, q', \square a'x \rangle & \text{if } \Delta = -1 \end{array}$$

Accepting Configurations

Definition (Accepting Configuration: Time)

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be an NTM,
let $c = \langle w, q, x \rangle$ be a configuration of M , and let $n \in \mathbb{N}_0$.

- If $q = q_Y$, M **accepts c in time n** .
- If $q \neq q_Y$ and M accepts some c' with $c \vdash c'$ in time n , then M **accepts c in time $n + 1$** .

Definition (Accepting Configuration: Space)

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be an NTM,
let $c = \langle w, q, x \rangle$ be a configuration of M , and let $n \in \mathbb{N}_0$.

- If $q = q_Y$ and $|w| + |x| \leq n$, M **accepts c in space n** .
- If $q \neq q_Y$ and M accepts some c' with $c \vdash c'$ in space n , then M **accepts c in space n** .

Note: “in time/space n ” means **at most n** , not **exactly n**

Accepting Words and Languages

Definition (Accepting Words)

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be an NTM.

M **accepts the word $w \in \Sigma^*$ in time (space) $n \in \mathbb{N}_0$**

iff M accepts $\langle \varepsilon, q_0, w \rangle$ in time (space) n .

- Special case: M accepts ε in time (space) $n \in \mathbb{N}_0$
iff M accepts $\langle \varepsilon, q_0, \square \rangle$ in time (space) n .

Definition (Accepting Languages)

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be an NTM, and let $f : \mathbb{N}_0 \rightarrow \mathbb{N}_0$.

M **accepts the language $L \subseteq \Sigma^*$ in time (space) f**

iff M accepts each word $w \in L$ in time (space) $f(|w|)$,

and M does not accept any word $w \notin L$ (in any time/space).

Background: Complexity Classes

Time and Space Complexity Classes

Definition (DTIME, NTIME, DSPACE, NSPACE)

Let $f : \mathbb{N}_0 \rightarrow \mathbb{N}_0$.

Complexity class **DTIME**(f) contains all languages accepted in time f by some DTM.

Complexity class **NTIME**(f) contains all languages accepted in time f by some NTM.

Complexity class **DSPACE**(f) contains all languages accepted in space f by some DTM.

Complexity class **NSPACE**(f) contains all languages accepted in space f by some NTM.

Polynomial Time and Space Classes

Let \mathcal{P} be the set of polynomials $p : \mathbb{N}_0 \rightarrow \mathbb{N}_0$
whose coefficients are natural numbers.

Definition (P, NP, PSPACE, NPSPACE)

$$P = \bigcup_{p \in \mathcal{P}} \text{DTIME}(p)$$

$$NP = \bigcup_{p \in \mathcal{P}} \text{NTIME}(p)$$

$$\text{PSPACE} = \bigcup_{p \in \mathcal{P}} \text{DSPACE}(p)$$

$$\text{NPSPACE} = \bigcup_{p \in \mathcal{P}} \text{NSPACE}(p)$$

Polynomial Complexity Class Relationships

Theorem (Complexity Class Hierarchy)

$$P \subseteq NP \subseteq PSPACE = NPSPACE$$

Proof.

$P \subseteq NP$ and $PSPACE \subseteq NPSPACE$ are obvious because deterministic Turing machines are a special case of nondeterministic ones.

$NP \subseteq NPSPACE$ holds because a Turing machine can only visit polynomially many tape cells within polynomial time.

$PSPACE = NPSPACE$ is a special case of a classical result known as Savitch's theorem (Savitch 1970). □

Summary

Summary

- We recalled the definitions of the most important **complexity classes** from complexity theory:
 - **P**: decision problems solvable in **polynomial time**
 - **NP**: decision problems solvable in **polynomial time** by **non-deterministic** algorithms
 - **PSPACE**: decision problems solvable in **polynomial space**
 - **NPSPACE**: decision problems solvable in **polynomial space** by **non-deterministic** algorithms
- These classes are related by $P \subseteq NP \subseteq PSPACE = NPSPACE$.