

Planning and Optimization

B6. Computational Complexity of Planning: Background

Malte Helmert and Gabriele Röger

Universität Basel

September 30, 2026

Planning and Optimization

September 30, 2026 — B6. Computational Complexity of Planning: Background

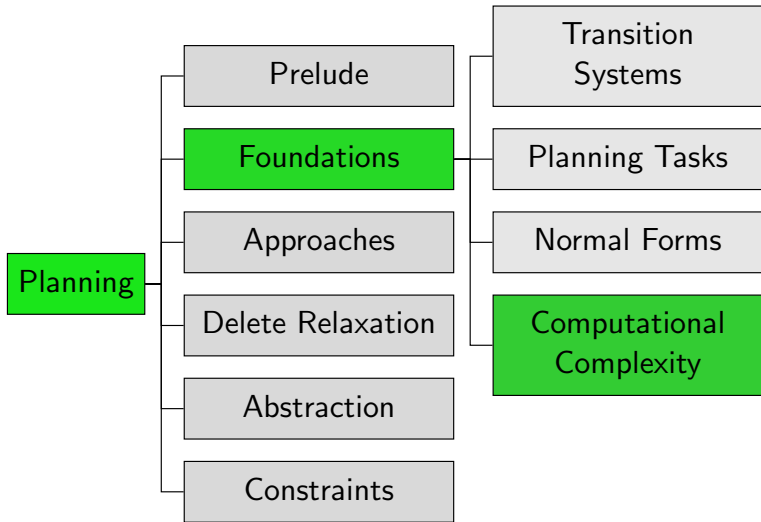
B6.1 Motivation

B6.2 Turing Machines

B6.3 Complexity Classes

B6.4 Summary

Content of the Course



B6.1 Motivation

How Difficult is Planning?

- ▶ Using **state-space search** (e.g., using Dijkstra's algorithm on the transition system), planning can be solved in **polynomial time** in 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?

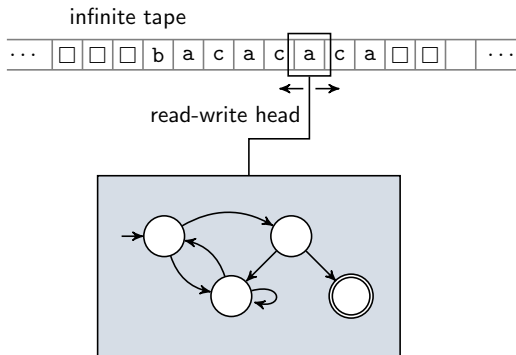
Reminder: Complexity Theory

Need to Catch Up?

- ▶ 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, NP, PSPACE
 - ▶ polynomial reductions
- ▶ If you are not familiar with these topics, we recommend Chapters B11, D1–D3, D6 of the **Theory of Computer Science** course at <https://dmi.unibas.ch/en/studies/computer-science/course-offer-spring-2026/10948-main-lecture-theory-of-computer-science/>

B6.2 Turing Machines

Turing Machines: Conceptually



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** $\delta : (Q' \times \Sigma_{\square}) \rightarrow 2^{Q \times \Sigma_{\square} \times \{-1, +1\}}$

Deterministic Turing machine (DTM):

$$|\delta(q, s)| = 1 \text{ for all } \langle q, s \rangle \in Q' \times \Sigma_{\square}$$

Turing Machines: Accepted Words

► Initial configuration

- state q_0
- input word on tape, all other tape cells contain $\square$
- head on first symbol of input word

► Step

- If in state q , reading symbol s , and $\langle q', s', d \rangle \in \delta(q, s)$ then
- the NTM **can** transition to state q' , replacing s with s' and moving the head one cell to the left/right ($d = -1/+1$).
- Input word ($\in \Sigma^*$) is **accepted** if **some** sequence of transitions brings the NTM from the initial configuration into state q_Y .

B6.3 Complexity Classes

Acceptance in Time and Space

Definition (Acceptance of a Language in Time/Space)

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

A NTM **accepts** language $L \subseteq \Sigma^*$ **in time f** if it accepts each $w \in L$ within $f(|w|)$ steps and does not accept any $w \notin L$ (in any time).

It **accepts** language $L \subseteq \Sigma^*$ **in space f** if it accepts each $w \in L$ using at most $f(|w|)$ tape cells and does not accept any $w \notin L$.

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.

$NP \subseteq PSPACE$ 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). □

B6.4 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 **nondeterministic** algorithms
 - ▶ **PSPACE**: decision problems solvable in **polynomial space**
 - ▶ **NPSPACE**: decision problems solvable in **polynomial space** by **nondeterministic** algorithms
- ▶ These classes are related by $P \subseteq NP \subseteq PSPACE = NPSPACE$.