

Principles of AI Planning

17. Computational complexity of classical planning

Bernhard Nebel and Robert Mattmüller

Albert-Ludwigs-Universität Freiburg

February 10th, 2012

Motivation

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

More
complexity
results

Summary

How hard is planning?

- We have seen that planning can be done in time **polynomial** in the size of the **transition system**.
- However, we have not seen algorithms which are polynomial in the **input size** (size of the task description).
- ↪ What is the precise **computational complexity** of the **planning problem**?

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

More
complexity
results

Summary

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?

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

More
complexity
results

Summary

Background

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Turing machines
Complexity
classes

Complexity of
planning

More
complexity
results

Summary

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** $\delta \subseteq (Q' \times \Sigma_{\square}) \times (Q \times \Sigma_{\square} \times \{-1, +1\})$

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background
Turing machines
Complexity
classes

Complexity of
planning

More
complexity
results

Summary

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 $\langle \langle q, a \rangle, \langle q', a', \Delta \rangle \rangle \in \delta$.

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background
Turing machines
Complexity
classes

Complexity of
planning

More
complexity
results

Summary

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 before tape head
- q : current state
- x : tape contents after and including tape head

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background
Turing machines
Complexity
classes

Complexity of
planning

More
complexity
results

Summary

Turing machine transitions

Definition (yields relation)

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 $\langle \langle q, a \rangle, \langle q', a', \Delta \rangle \rangle \in \delta$:

$$\langle w, q, ax \rangle \vdash \langle wa', q', x \rangle \quad \text{if } \Delta = +1, |x| \geq 1$$

$$\langle w, q, a \rangle \vdash \langle wa', q', \square \rangle \quad \text{if } \Delta = +1$$

$$\langle wb, q, ax \rangle \vdash \langle w, q', ba'x \rangle \quad \text{if } \Delta = -1$$

$$\langle \epsilon, q, ax \rangle \vdash \langle \epsilon, q', \square a'x \rangle \quad \text{if } \Delta = -1$$

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background
Turing machines
Complexity
classes

Complexity of
planning

More
complexity
results

Summary

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** .

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background
Turing machines
Complexity
classes

Complexity of
planning

More
complexity
results

Summary

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 $\langle \epsilon, q_0, w \rangle$ in time (space) n .

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

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background
Turing machines
Complexity
classes

Complexity of
planning

More
complexity
results

Summary

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.

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background
Turing machines
Complexity
classes

Complexity of
planning

More
complexity
results

Summary

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)

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

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

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

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

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Turing machines

Complexity
classes

Complexity of
planning

More
complexity
results

Summary

Polynomial complexity class relationships

Theorem (complexity class hierarchy)

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

Proof.

$P \subseteq NP$ and $PSPACE \subseteq NPSPACE$ is 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). □

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background
Turing machines
**Complexity
classes**

Complexity of
planning

More
complexity
results

Summary

Complexity of propositional planning

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence
PSPACE-
completeness

More
complexity
results

Summary

The propositional planning problem

Definition (plan existence)

The **plan existence** problem (PLANEX) is the following decision problem:

GIVEN: Planning task Π

QUESTION: Is there a plan for Π ?

↪ decision problem analogue of **satisficing planning**

Definition (bounded plan existence)

The **bounded plan existence** problem (PLANLEN) is the following decision problem:

GIVEN: Planning task Π , length bound $K \in \mathbb{N}_0$

QUESTION: Is there a plan for Π of length at most K ?

↪ decision problem analogue of **optimal planning**

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence
PSPACE-
completeness

More
complexity
results

Summary

Plan existence vs. bounded plan existence

Theorem (reduction from PLANEX to PLANLEN)

$\text{PLANEX} \leq_p \text{PLANLEN}$

Proof.

A propositional planning task with n state variables has a plan iff it has a plan of length at most $2^n - 1$.

\rightsquigarrow map instance Π of PLANEX to instance $\langle \Pi, 2^n - 1 \rangle$ of PLANLEN, where n is the number of n state variables of Π

\rightsquigarrow polynomial reduction □

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence

PSPACE-
completeness

More
complexity
results

Summary

Membership in PSPACE

Theorem (PSPACE membership for PLANLEN)

PLANLEN \in PSPACE

Proof.

Show PLANLEN \in NPSPACE and use Savitch's theorem.

Nondeterministic algorithm:

def plan($\langle A, I, O, G \rangle, K$):

$s := I$

$k := K$

while $s \not\equiv G$:

guess $o \in O$

fail if o not applicable in s **or** $k = 0$

$s := app_o(s)$

$k := k - 1$

accept



AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence

PSPACE-
completeness

More
complexity
results

Summary

Hardness for PSPACE

Idea: **generic reduction**

- For an **arbitrary fixed DTM M** with space bound polynomial p and input w , generate planning task which is solvable iff M accepts w in space $p(|w|)$.
- For simplicity, restrict to TMs which never move to the left of the initial head position (no loss of generality).

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence

PSPACE-
completeness

More
complexity
results

Summary

Reduction: state variables

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be the fixed DTM and let p be its space-bound polynomial.

Given input $w_1 \dots w_n$, define **relevant tape positions**
 $X := \{1, \dots, p(n)\}$.

State variables

- state_q for all $q \in Q$
- head_i for all $i \in X \cup \{0, p(n) + 1\}$
- $\text{content}_{i,a}$ for all $i \in X, a \in \Sigma_{\square}$

\rightsquigarrow allows encoding a Turing machine configuration

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence

PSPACE-
completeness

More
complexity
results

Summary

Reduction: initial state

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be the fixed DTM and let p be its space-bound polynomial.

Given input $w_1 \dots w_n$, define **relevant tape positions**
 $X := \{1, \dots, p(n)\}$.

Initial state

Initially true:

- state_{q_0}
- head_1
- $\text{content}_{i,w_i}$ for all $i \in \{1, \dots, n\}$
- $\text{content}_{i,\square}$ for all $i \in X \setminus \{1, \dots, n\}$

Initially false:

- all others

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence
PSPACE-
completeness

More
complexity
results

Summary

Reduction: operators

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be the fixed DTM and let p be its space-bound polynomial.

Given input $w_1 \dots w_n$, define **relevant tape positions**
 $X := \{1, \dots, p(n)\}$.

Operators

One operator for each transition rule $\delta(q, a) = \langle q', a', \Delta \rangle$
and each cell position $i \in X$:

- precondition: $\text{state}_q \wedge \text{head}_i \wedge \text{content}_{i,a}$
- effect: $\neg \text{state}_q \wedge \neg \text{head}_i \wedge \neg \text{content}_{i,a}$
 $\wedge \text{state}_{q'} \wedge \text{head}_{i+\Delta} \wedge \text{content}_{i,a'}$
 - If $q = q'$ and/or $a = a'$, omit the effects on state_q and/or $\text{content}_{i,a}$, to avoid consistency condition issues.

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence

PSPACE-
completeness

More
complexity
results

Summary

Reduction: goal

Let $M = \langle \Sigma, \square, Q, q_0, q_Y, \delta \rangle$ be the fixed DTM and let p be its space-bound polynomial.

Given input $w_1 \dots w_n$, define **relevant tape positions**
 $X := \{1, \dots, p(n)\}$.

Goal

state _{q_Y}

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence

PSPACE-
completeness

More
complexity
results

Summary

PSPACE-completeness for STRIPS plan existence

Theorem (PSPACE-completeness; Bylander, 1994)

PLANEX and PLANLEN are PSPACE-complete.

This is true even when restricting to STRIPS tasks.

Proof.

Membership for PLANLEN was already shown.

Hardness for PLANEX follows because we just presented a polynomial reduction from an arbitrary problem in PSPACE to PLANEX . (Note that the reduction only generates STRIPS tasks.)

Membership for PLANEX and hardness for PLANLEN follows from the polynomial reduction from PLANEX to PLANLEN . □

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

(Bounded) plan
existence
PSPACE-
completeness

More
complexity
results

Summary

More complexity results

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

More
complexity
results

Summary

More complexity results

In addition to the basic complexity result presented in this chapter, there are many special cases, generalizations, variations and related problems studied in the literature:

- different **planning formalisms**
 - e. g., finite-domain representation, nondeterministic effects, partial observability, schematic operators, numerical state variables
- **syntactic restrictions** of planning tasks
 - e. g., without preconditions, without conjunctive effects, STRIPS without delete effects
- **semantic restrictions** of planning task
 - e. g., restricting to certain classes of causal graphs
- **particular planning domains**
 - e. g., Blocksworld, Logistics, FreeCell

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

More
complexity
results

Summary

Complexity results for different planning formalisms

Some results for different planning formalisms:

- **FDR tasks:**
 - same complexity as for propositional tasks (“folklore”)
 - also true for the SAS⁺ special case
- **nondeterministic effects:**
 - fully observable: EXP-complete (Littman, 1997)
 - unobservable: EXPSPACE-complete (Haslum & Jonsson, 1999)
 - partially observable: 2EXP-complete (Rintanen, 2004)
- **schematic operators:**
 - usually adds one exponential level to PLANEX complexity
 - e. g., classical case EXPSPACE-complete (Erol et al., 1995)
- **numerical state variables:**
 - undecidable in most variations (Helmert, 2002)

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

More
complexity
results

Summary

Summary

- **Propositional planning** is **PSPACE-complete**.
- The hardness proof is a polynomial reduction that translates an **arbitrary polynomial-space DTM** into a **STRIPS task**:
 - Configurations of the DTM are encoded by propositional variables.
 - Operators simulate transitions of the DTM.
 - The DTM accepts an input iff there is a plan for the corresponding STRIPS task.
- This implies that there is **no polynomial algorithm** for classical planning unless $P=PSPACE$.
- It also means that classical planning is not polynomially reducible to any problem in NP unless $NP=PSPACE$.

AI Planning

B. Nebel,
R. Mattmüller

Motivation

Background

Complexity of
planning

More
complexity
results

Summary