The class PSPACE

PSPACE is the class of languages that are decidable in polynomial space on a DTM, i.e.

$$\mathrm{PSPACE} = \bigcup_k \mathrm{SPACE}(n^k)$$
$$\mathrm{NPSPACE} = \bigcup_k \mathrm{NSPACE}(n^k)$$

We define $\mathrm{EXPTIME} = \bigcup_k \mathrm{TIME}(2^{n^k})$

Lemma

$\mathrm{SPACE}(f(n)) \subseteq \mathrm{TIME}(2^{\mathcal{O}(f(n))})$

The class PSPACE

PSPACE is the class of languages that are decidable in polynomial space on a DTM, i.e.

$$\mathrm{PSPACE} = \bigcup_k \mathrm{SPACE}(n^k)$$
$$\mathrm{NPSPACE} = \bigcup_k \mathrm{NSPACE}(n^k)$$

We define $\mathrm{EXPTIME} = \bigcup_k \mathrm{TIME}(2^{n^k})$

Lemma

$\mathrm{SPACE}(f(n)) \subseteq \mathrm{TIME}(2^{\mathcal{O}(f(n))})$

What we know so far

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} = \mathrm{NPSPACE} \subseteq \mathrm{EXPTIME}$$

PSPACE-Completeness

Definition

If every $A \in \mathrm{PSPACE}$ is polynomial time reducible to a language B, then B is *PSPACE-hard*.

Definition

A language B is *PSPACE-complete* if it satisfies two conditions:

1. $B \in \mathrm{PSPACE}$
2. B is *PSPACE-hard*

A PSPACE-complete language

Definition

A formula φ is a *fully quantified Boolean formula* if

$$\varphi = Q_1 x_1 Q_2 x_2 \ldots Q_n x_n . \psi$$

where ψ is a Boolean formula in CNF, $x_1, \ldots, x_n$ are the Boolean variables in ψ, and $Q_i \in \{\forall, \exists\}$, $1 \leq i \leq n$. φ is said to be in *prenex normal form*.

Consider the problem

$$TQBF = \{\langle\varphi\rangle \mid \varphi \text{ is a true fully quantified Boolean formula}\}$$

Theorem

TBQF is PSPACE-complete.

$TBQF \in$ PSPACE

The following polynomial space algorithm decides *TQBF*:

$T =$"On input $\langle \varphi \rangle$, in which φ is a fully quantified Boolean formula:

1. If φ contains no quantifiers, then it is an extension with only constants. So, evaluate φ and *accept* if it is true; otherwise, *reject*
2. If $\varphi = \exists x.\, \psi$, recursively call T on ψ, first with 0 substituted for x and then with 1 substituted for x. If either result is "accept", *accept*, otherwise *reject*
3. If $\varphi = \forall x.\, \psi$, recursively call T on ψ, first with 0 substituted for x and then with 1 substituted for x. If both results are "accept", *accept*, otherwise *reject*"