PSPACE-completeness

Evgenij Thorstensen

V18

Space, recap

$\mathsf{NTIME}(f(n)) \subseteq \mathsf{SPACE}(f(n)) \subseteq \mathsf{TIME}(2^{cf(n)})$

$\mathsf{NSPACE}(f(n)) = \mathsf{SPACE}(f(n)^2)$, Savitch.

PSPACE, recap

$$\mathsf{PSPACE} = \bigcup_{k \in \mathbb{N}} \mathsf{SPACE}(n^k)$$

We define PSPACE-completeness by polynomial-time reduction.

The language TQBF, true quantified boolean formulas, is in PSPACE.

Membership by recursive evaluation, giving an and-or tree.

PSPACE-completeness

We will prove that TQBF is PSPACE-complete, by an approach similar to Cook-Levin.

However, will also need to use the recursive speedup from Savitch.

Formulas will encode yieldability between configurations.

Universal quantification will be crucial to keep the size bound.

Preliminaries

Let $L \in \text{PSPACE}$, and let M be a decider for L space-bounded by $O(n^k)$.

We will produce a TQBF formula $\phi(c_1, c_2, t)$ that is true if and only if CanYield(c_1, c_2, t) returns true.

This will encode the tree produced by the recursion.

CanYield compares neighbour configurations in the base case. This we can do without any quantifiers, as seen in the proof of Cook-Levin.

Use variables $T(i, j, k)$ and so on, output a formula specifying which are true and what transition has to hold.

The recursion

Naively, we need

$$\phi(c_1, c_2, t) = \exists c_m . \phi(c_1, c_m, \frac{t}{2}) \wedge \phi(c_m, c_2, \frac{t}{2})$$

The recursion

Naively, we need

$$\phi(c_1, c_2, t) = \exists c_m . \phi(c_1, c_m, \frac{t}{2}) \wedge \phi(c_m, c_2, \frac{t}{2})$$

Here we just get a boatload of $\exists \ldots$ and end up with a SAT instance...

The recursion

Naively, we need

$$\phi(c_1, c_2, t) = \exists c_m . \phi(c_1, c_m, \frac{t}{2}) \wedge \phi(c_m, c_2, \frac{t}{2})$$

Here we just get a boatload of $\exists \ldots$ and end up with a SAT instance...

... proving PSPACE = NP... if only.

The recursion

Naively, we need

$$\phi(c_1, c_2, t) = \exists c_m . \phi(c_1, c_m, \frac{t}{2}) \wedge \phi(c_m, c_2, \frac{t}{2})$$

Here we just get a boatload of $\exists \ldots$ and end up with a SAT instance...

... proving PSPACE = NP... if only.

The construction is correct, by the way. Just not doable in polynomial time, or even polynomial space.

The problem

$$\phi(c_1, c_2, t) = \exists c_m . \phi(c_1, c_m, \frac{t}{2}) \wedge \phi(c_m, c_2, \frac{t}{2})$$

t is cut in half, but the formula doubles in size.

We end up with a SAT formula of size 2^{n^k}.

Need Savitch to the rescue — use $\forall$ to "reuse" space.

Universal power

Let's fold that formula using $\forall$.

$$\phi(c_1, c_2, t) = \exists c_m . \phi(c_1, c_m, \frac{t}{2}) \wedge \phi(c_m, c_2, \frac{t}{2})$$

becomes

$$\phi(c_1, c_2, t) = \exists c_m . \forall c_x, c_y . G \rightarrow \phi(c_x, c_y, \frac{t}{2})$$

with $G = (c_x = c_1 \vee c_x = c_m) \wedge (c_y = c_m \vee c_y = c_2)$

This formula is true iff the following four are true:

Universal power

Let's fold that formula using $\forall$.

$$\phi(c_1, c_2, t) = \exists c_m . \phi(c_1, c_m, \frac{t}{2}) \wedge \phi(c_m, c_2, \frac{t}{2})$$

becomes

$$\phi(c_1, c_2, t) = \exists c_m . \forall c_x, c_y . G \rightarrow \phi(c_x, c_y, \frac{t}{2})$$

with $G = (c_x = c_1 \vee c_x = c_m) \wedge (c_y = c_m \vee c_y = c_2)$

This formula is true iff the following four are true:

- $\phi(c_1, c_m, \frac{t}{2})$
- $\phi(c_1, c_2, \frac{t}{2})$
- $\phi(c_m, c_m, \frac{t}{2})$
- $\phi(c_m, c_2, \frac{t}{2})$

Correctness

We invoke Savitch for the recursion, and Cook-Levin for the coding.

$\forall$ only appears in the recursion.

Correctness (DTM accepts if and ony if the formula is true) thus follows.

The size bound

Due to the $\forall$, our formula grows with the depth of the recursion.

The depth is $\log c^{n^k} = O(n^{dk})$. At each level, we add a subformula of size $O(n^k)$.

End result is of size $O(n^{2dk})$, or $O(p(n)^2)$.

Other PSPACE-complete problems

Generalized two-player games, due to TQBF.

Minimax-style arguments, due to quantifier alteration.

In e.g. chess, white has to select a move such that for all moves by black, white wins. Recursively.

Formula game

Player A and player E play the following game: Given a TQBF formula, player A picks values for $\forall$-quantified variables, and player E for $\exists$-variables, in order of quantifiers.

If the assignment is a satisfying one, E wins.

E has a winning strategy if and only if the formula is true.

A different perspective

NPSPACE is essentially TQBF, by completeness.

TQBF is an and/or tree, each branch is a computation path/assignment.

NP is an OR tree, and coNP is an AND tree.

P is a tree without branching.

Polynomial hierarchy

Bounded quantifier alternation gives us some of the power of PSPACE.

Define Σ_i to be formulas of the form

$$\exists X_1 \forall X_2 \exists X_3 \ldots \phi$$

and Π_i to be formulas of the form

$$\forall X_1 \exists X_2 \forall X_3 \ldots \phi$$

where ϕ is propositional.

Easy observation: $\Sigma_1 \equiv \mathsf{NP}$, $\Pi_1 \equiv \mathsf{coNP}$.

PH

Quantifier alternation corresponds to alternating NP and coNP problems.

$$\mathsf{PH} = \bigcup_{i \in \mathbb{N}} \Sigma_i \cup \Pi_i$$

$\mathsf{P} \neq \mathsf{NP}$ if and only if $\mathsf{P} \neq \mathsf{PH}$.

If $\mathsf{PH} = \mathsf{PSPACE}$, then PH collapses to a fixed level.