Why do we believe that PSPACE ≠ EXPTIME?

I'm having trouble intuitively understanding why PSPACE is generally believed to be different from EXPTIME. If PSPACE is the set of problems solvable in space polynomial in the input size $f(n)$, then how can there be a class of problems that experience greater exponential time blowup and do not make use of exponential space?

Yuval Filmus' answer is already extremely helpful. However, could anyone sketch my a loose argument why it *might* be the case that PSPACE ≠ EXPTIME (i.e. that PSPACE is not a proper subset of EXPTIME)? Won't we need exponential space in order to beat the upperbound for the total number of system configurations achievable with space that scales polynomially with input size? Just to say, I can understand why EXPTIME ≠ EXPSPACE is an open matter, but I lack understanding regarding the relationship between PSPACE and EXPTIME.

complexity-theory complexity-classes intuition

edited Dec 16 '14 at 15:39 Raphael♦ 47.7k 16 115 251

asked Dec 16 '14 at 4:00 user25876 108 1 5

2 Answers

Let's refresh the definitions.

- **PSPACE** is the class of problems that can be solved on a deterministic Turing machine with polynomial space bounds: that is, for each such problem, there is a machine that decides the problem using at most $p(n)$ tape cells when its input has length n, for some polynomial p.
- **EXP** is the class of problems that can be solved on a deterministic Turing machine with exponential time bounds: for each such problem, there is a machine that decides the problem using at most $2^{p(n)}$ steps when its input has length n, for some polynomial p.

First, we should say that these two classes might be equal. They seem more likely to be different but classes sometimes turn out to be the same: for example, in 2004, Reingold proved that symmetric logspace is the same as ordinary logspace; in 1987, Immerman and Szelepcsényi independently proved that **NL** $=$ **co-NL** (and, in fact, that **NSPACE[**$f(n)$**]** $=$ **co-NSPACE[**$f(n)$**]** for any $f(n) \geq \log n$).

But, at the moment, most people believe that **PSPACE** and **EXP** are different. Why? Let's look at what we can do in the two complexity classes. Consider a problem in **PSPACE**. We're allowed to use $p(n)$ tape cells to solve an input of length n but it's hard to compare that against **EXP**, which is specified by a time bound.

How much time can we use for a **PSPACE** problem? If we only write to $p(n)$ tape cells, there are $2^{p(n)}$ different strings that could appear on the tape, assuming a binary alphabet. The tape head could be in any of $p(n)$ different places and the Turing machine could be in one of k different states. So the total number of configurations is $T(n) = k\,p(n)\,2^{p(n)}$. By the pigeonhole principle, if we run for $T(n) + 1$ steps, we must visit a configuration twice but, since the machine is deterministic, that means it will loop around and visit that same configuration infinitely often, i.e., it won't halt. Since part of the definition of being in **PSPACE** is that you have to *decide* the problem, any machine that doesn't terminate doesn't solve a **PSPACE** problem. In other words, **PSPACE** is the class of problems that are decidable using at most $p(n)$ space and at most $k\,p(n)\,2^{p(n)}$ time, which is at most $2^{q(n)}$ for some polynomial q. So we've shown that **PSPACE** $\subseteq$ **EXP**.

And how much space can we use for an **EXP** problem? Well, we're allowed $2^{p(n)}$ steps and the head of a Turing machine can only move one position at each step. Since the head can't move more than $2^{p(n)}$ positions, we can only use that many tape cells.

That's what the difference is: although both **PSPACE** and **EXP** are problems that can be solved in exponential time, **PSPACE** is restricted to polynomial space use, whereas **EXP** can use exponential space. That already suggests that **EXP** ought to be more powerful. For example, suppose you're trying to solve a problem about graphs. In **PSPACE**, you can look at every subset of the vertices (it only takes n bits to write down a subset). You can use some working space to compute on each subset but, once you've finished working on a subset, you must erase that working space and re-use it for the next subset. In **EXP**, on the other hand, you can not only look at every subset but you don't need to reuse your working space, so you can remember what you learnt about each one individually. That seems like it should be more powerful.

Another intuition for why they should be different is that the time and space hierarchy theorems tell us that allowing even a tiny bit more space or time strictly increases what you can compute. The hierarchy theorems only let you compare like with like (e.g., they show that **PSPACE** $\subsetneq$ **EXPSPACE** and **P** $\subsetneq$ **EXP**) so they don't directly apply to **PSPACE** vs

EXP but they do give us a strong intuition that more resource means that more problems become solvable.

edited Jan 1 '15 at 13:09

answered Dec 16 '14 at 9:05

David Richerby
43.8k 9 63 122

1 If EXPTIME allows exponential space, I suppose the right question is, can we say that it might be true that EXPTIME is a proper subset of EXPSPACE because EXPSPACE allows for problems that can be solved in superexponential time? – user25876 Dec 16 '14 at 9:32

If that's true, then I think everything makes sense for me. For some reason I had assumed that EXPTIME forbade the use of exponential space, but that's not the case. This is where my confusion came from. – user25876 Dec 16 '14 at 9:34

1 I like your subset example. IIRC correctly, we know problems that can't be computed online (as well as with full information) so you'd have to keep all elements in memory. Intuitively speaking. – Raphael ♦ Dec 16 '14 at 10:43

@user25876 Yes, the same argument that says a PSPACE machine can use exponential time says that an EXPSPACE machine can use doubly-exponential time (i.e., $2^{2^{\mathrm{poly}(n)}}$). – David Richerby Dec 16 '14 at 11:44

@DavidRicherby I'm accepting your answer. Do you know any paper references BTW discussing the technical barriers for proving or disproving PSPACE as a proper subset of EXPTIME? I'm actually very curious about it now. – user25876 Dec 16 '14 at 11:50

A machine running in exponential time could use exponential space. So a priori it could be that machines restricted to polynomial space would be weaker. A similar situation occurs for P and L. A machine running in polynomial time could use polynomial space, so a priori it could be that machines restricted to logarithmic space would be weaker. It is even conjectured that P is different from NL, the non-deterministic analog of L. (For PSPACE the corresponding conjectures are equivalent, since PSPACE=NPSPACE due to Savitch's theorem.) Unfortunately, we don't know how to prove these conjectures currently. The conjecture EXPTIME≠PSPACE is stronger since it implies P≠NL via a padding argument.

answered Dec 16 '14 at 5:00

Yuval Filmus
137k 6 124 259