Padding argument

In computational complexity theory, the **padding argument** is a tool to conditionally prove that if some complexity classes are equal, then some other bigger classes are also equal.

Example

The proof that P = NP implies EXP = NEXP uses "padding".

$\mathsf{EXP} \subseteq \mathsf{NEXP}$ by definition, so it suffices to show $\mathsf{NEXP} \subseteq \mathsf{EXP}$.

Let *L* be a language in NEXP. Since *L* is in NEXP, there is a non-deterministic Turing machine *M* that decides *L* in time 2^{n^c} for some constant *c*. Let

$$L' = \{x1^{2^{|x|^c}} \mid x \in L\},$$

where '1' is a symbol not occurring in *L*. First we show that L' is in NP, then we will use the deterministic polynomial time machine given by P = NP to show that *L* is in EXP.

L' can be decided in non-deterministic polynomial time as follows. Given input x', verify that it has the form $x' = x1^{2^{|x|^c}}$ and reject if it does not. If it has the correct form, simulate *M*(*x*). The simulation takes non-deterministic $2^{|x|^c}$ time, which is polynomial in the size of the input, x'. So, L' is in NP. By the assumption P = NP, there is also a deterministic machine *DM* that decides L' in polynomial time. We can then decide *L* in deterministic exponential time as follows. Given input x, simulate $DM(x1^{2^{|x|^c}})$. This takes only exponential time in the size of the input, x.

The 1^d is called the "padding" of the language *L*. This type of argument is also sometimes used for space complexity classes, alternating classes, and bounded alternating classes.

References

- Arora, Sanjeev; Barak, Boaz (2009), *Computational Complexity: A Modern Approach* (http://www.cs.princeton.edu/theory/complexity/), Cambridge, p. 57, ISBN 978-0-521-42426-4