CSC2401 Introduction to Computational Complexity **Fall 1998**

Lecture 13: October 27

Lecturer: Micah Adler *Scribe: MohammadReza Salavatipour*

13.1 PSPACE-completeness

We are going to discuss PSPACE-Completeness.

Definition 13.1 *A language L is PSPACE-Complete if it satisfies the following two conditions:*

1. $L \in PSPACE$
2. $\forall\, L' \in PSPACE,\ L' \leq_p L$

We can also define this in term of logspace reductions; all the results to be described for PSPACE-Completeness hold for this definition as well. The first problem we show to be PSPACE-Complete is a generalization of SAT called Quantified Boolean Formula, in which we have:

- Universal qualifiers : $\forall x$ (for all x)
- Existential qualifiers : $\exists x$ (there exists x)

Definition 13.2 *A QBF is a boolean formula prefixed by a string of universal and existential qualifiers, one for each variable in the formula. A QBF is either* True *or* False.

Examples:

- $\forall x \exists y ((x \vee y) \wedge (\overline{x} \vee \overline{y}))$ is a QBF that evaluates to *True*.
- $\exists y \forall z ((y \vee z) \wedge (\overline{y} \vee \overline{z}))$ is a QBF that evaluates to *False*.

Definition 13.3 *TQBF=*$\{\Phi | \Phi$ *is a true QBF*$\}$

In the textbook, this is called QSAT. Note that SAT is a TQBF with only $\exists$ qualifiers and VALIDITY is a TQBF with only $\forall$ qualifiers.

Theorem 13.4 *TQBF is PSPACE-Complete.*

Proof: First we show that TQBF is in PSPACE. To do so, we give a recursive algorithm.

For an input Φ :

- if $\Phi = \exists x \Psi$ then recurse on Ψ with both $x = True$ and $x = False$, accept if either accept.

- if $\Phi = \forall x \Psi$ then recurse on Ψ with both $x = True$ and $x = False$, accept if both accept.

Clearly this computes the correct result. To ensure that we use only polynomial space, for both cases we reuse the space required for each of the two recursive calls. Thus, the space needed by this algorithm is linear in the depth of the recursion, which is equal to the number of variables and so is linear.

For proving the completeness, we show that:

$$\forall L \in PSPACE, L \leq_p TQBF.$$

We use an idea based on Cook's theorem.

Idea : L is decided by a 1-tape T.M. M, which uses at most space n^k, for some k, for any input of size n. For given input w, construct the QBF called Φ_w such that: $\Phi_w \in TQBF \iff$ M accepts w $\iff w \in L$.

First, let's see what happens when we try to use Cook's construction directly. We need a computation tableau. Since we use polynomial space, it has n^k columns but the number of rows can be as large as c^{n^k}. The formula that results from Cook's construction has size $\Theta(l)$ where l is the number of cells in tableau, and thus would be exponential. In order to obtain a polynomial sized formula, we are going to use Savitch's idea: we are going to consider paths of length t between two nodes in the configuration graph, and recursively divide the path into two subpathes of length $\frac{t}{2}$.

Definition 13.5 *For configurations C_a and C_b, we say that we can go from C_a to C_b in t steps, $C_a \stackrel{t}{\Longrightarrow} C_b$ if C_b follows from C_a in t steps.*

We wish to solve the following subproblem: given configurations C_a and C_b, construct a QBF $\Phi_{C_a \stackrel{t}{\rightarrow} C_b}$ in polynomial time, where $\Phi_{C_a \stackrel{t}{\rightarrow} C_b} = True \iff C_a \stackrel{t}{\rightarrow} C_b$. As a first attempt to use Savitch's recursive idea, let's try to construct $\Phi_{C_a \stackrel{a}{\rightarrow} C_b}$ as follows:

base case (t=1): Use Cook's construction.

recursive case (t>1):

$$\exists C_{mid}(\Phi_{C_a \stackrel{\frac{t}{2}}{\rightarrow} C_{mid}} \wedge \Phi_{C_{mid} \stackrel{\frac{t}{2}}{\rightarrow} C_b})$$

where $\exists C_{mid}$ is: $\exists x_1 \exists x_2 \ldots \exists x_r$ where $x_1, x_2, \ldots, x_r$ encode a T.M configuration C_{mid} using Cook's construction.

This still does not lead to a polynomial construction: every row of the computational tableau is C_{mid} once, and thus forces us to introduce a new set of variables $x_1, x_2, \ldots, x_r$. Alternatively, we can see that the construction is not polynomial since each recursive call introduces two new subproblems. Thus, for each level of recursion the number of variables doubles and the total number of levels is $O(n^k)$, which gives us an exponential number of variables.

If we think back to Savitch's Theorem, we saw there how to save space: we reused the space for the two recursive subproblem. We are going to use the same idea here: we are going to use one sub-formula to solve both recursive problems. By doing this, we only create one new subproblem at each level of the recursion, thus giving us a much smaller overall formula size.

Actual Construction: We use the universal qualifiers to reduce the overall formula size. Again, define

$\Phi_{C_a \xrightarrow{t} C_b}$ as:

base case (t=1): use Cook's construction

recursive case (t>1):

$$\exists C_{mid} \forall C_d \forall C_e [((C_d = C_a) \wedge (C_e = C_{mid})) \vee ((C_d = C_{mid}) \wedge (C_e = C_b))] \Longrightarrow \Phi_{C_d \xrightarrow{\frac{t}{2}} C_e}$$

$C_{mid} = x_1, x_2, \ldots, x_r$, $C_d = y_1, y_2, \ldots, y_r$, and $C_e = z_1, z_2, \ldots, z_r$ represent T.M configurations, as before. With this formula, we need to consider the sub-formula with both

- $C_d = C_a$ and $C_e = C_{mid}$
- $C_d = C_{mid}$ and $C_e = C_b$.

Let $\Phi_w = \Phi_{C_{start}} \overset{d^{n^k}}{\rightarrow} C_{accept}$ where C_{start} is the start state and C_{accept} is the accept state, which (as in Cook's Theorem), we can assume to be unique. In this construction, the number of new variables at each level of the recursion is $O(n^k)$ and the number of levels is still $O(n^k)$. Each variable is used only a constant number of times in the final formula, and thus $|\Phi_w| = O(n^{2k})$. ■

13.2 Finding Optimal Strategies in 2-Person Games

One interpretation of PSPACE-Complete languages is those that capture the difficulty of finding optimal strategies in 2-person games. The first example of this we shall see is the Formula Game.

Definition 13.6 *The Formula Game:*

- *Given: a QBF Φ and two players: a $\exists$* player *and a $\forall$* player*.*
- *Play: players take turns choosing values for variables of Φ in order of the qualifiers in Φ.*
- *Winning: if the resulting formula is* True *then the $\exists$ player wins. Otherwise the $\forall$ player wins.*

Theorem 13.7 *QBF is True $\Longleftrightarrow$ The $\exists$ player has a winning strategy in the formula game for Φ.*

Proof: The proof is follows directly from the definition of the qualifiers. ■

Therefore, the question of whether the $\exists$ player has a winning strategy is also PSPACE-Complete.

Some other 2-person games are : GO and Chess.

- GO An ancient game with two players played on an $n \times n$ board. The problem of determining whether a given GO position has a winning strategy for player 1 is also PSPACE-Complete.
- CHESS If we consider an $n \times n$ board and scale the number of pieces, then determining if a given chess position has a winning strategy for White is PSPACE-Hard.

A couple of comments on these results:

- If we bound the allowed length of a CHESS game to be polynomial in the length of the input, then CHESS is also in PSPACE.
- Neither of these results say anything about the starting position of the games of GO and CHESS: they only say that there are difficult to analyze positions. Thus, for all we know, the starting positions could be easy to analyze. Nevertheless, this does give us considerable insight into the question of solving these problems optimally.