

# TQBF is PSPACE-complete

This is the proof that was presented in class on September 25, 2012.

A *quantified boolean formula* (QBF) is an expression of the form

$$Q_1x_1Q_2x_2\cdots Q_nx_n\phi(x_1,x_2,\dots,x_n),$$

where each quantifier  $Q_i$  is either  $\exists$  or  $\forall$ , and  $\phi$  is a quantifier-free boolean formula. Because a QBF contains no free variables, it must be true or false.

For example,  $\forall x_1\exists x_2\forall x_3((x_1\vee\bar{x}_2)\wedge(\bar{x}_1\vee x_3))$  is a false QBF. To see this, note that, if  $x_1$  is true, then  $x_3$  must be true in order to satisfy the clause  $((\bar{x}_1\vee x_3))$ , but  $x_3$  is a universally quantified variable.

The set of all true quantified boolean formulae is denoted TQBF. Our goal here is to prove that TQBF is PSPACE-complete. So we must prove that it is in PSPACE and that every set in PSPACE is reducible to it. Here, the relevant notion of reduction is just  $\leq_P$ , *i.e.*, many-to-one, polynomial-time reduction.

First, we give a recursive, polynomial-space algorithm that decides whether a QBF is true. Consider an input  $\Psi = Q_1x_1Q_2x_2\cdots Q_nx_n\phi(x_1,x_2,\dots,x_n)$ , where  $\phi$  has  $m$  clauses. Note that the size of a clause is at most  $n$ ; so we wish to show that the truth of  $\Psi$  can be decided in  $\text{poly}(n,m)$  space. Our recursive algorithm will proceed by instantiating and “peeling off” quantifiers one by one, starting with  $Q_1$ . For  $0 \leq i \leq n$ , let  $S(\phi,i)$  be the space required to evaluate

$$\Psi_i(\epsilon_1,\dots,\epsilon_{n-i}) = Q_{n-i+1}x_{n-i+1}Q_{n-i+2}x_{n-i+2}\cdots Q_nx_n\phi(\epsilon_1,\dots,\epsilon_{n-i},x_{n-i+1},x_{n-i+2},\dots,x_n),$$

*i.e.*, to evaluate the QBF that results by letting  $x_j = \epsilon_j \in \{0,1\}$ , for  $1 \leq j \leq n-i$ , and leaving the  $i$  variables  $x_j$ , for  $n-i+1 \leq j \leq n$ , quantified as in the original  $\Psi$ . The total space complexity of our algorithm is  $S(\phi,n)$ . In the base case of the recursion, when  $i = 0$ , we have instantiated all of the variables, and the algorithm simply has to evaluate an  $n$ -variable,  $m$ -clause formula on a specific assignment. So  $S(\phi,0) = O(mn)$ . To evaluate the truth of the  $i$ -variable formula  $\Psi_i$ , we instantiate  $x_{n-i+1}$  both ways; that is, we call the algorithm recursively on both  $Q_{n-i+2}x_{n-i+2}\cdots Q_nx_n\phi(\epsilon_1,\dots,\epsilon_{n-i},0,x_{n-i+2},\dots,x_n)$  and  $Q_{n-i+2}x_{n-i+2}\cdots Q_nx_n\phi(\epsilon_1,\dots,\epsilon_{n-i},1,x_{n-i+2},\dots,x_n)$ . If  $Q_{n-i+1} = \forall$ , then we declare  $\Psi$  to be true if and only if both recursive calls return true; if  $Q_{n-i+1} = \exists$ , then we declare  $\Psi$  to be true if and only if at least one of the recursive calls returns true. As in the proof of Savitch’s theorem, we **reuse the space** that we used for the  $x_{n-i+1} = 0$  call when we do the  $x_{n-i+1} = 1$  call. So  $S(\phi,i) = S(\phi,i-1) + O(mn)$ , where  $S(\phi,i-1)$  is the space needed for the recursive call, and  $O(mn)$  is the space needed to write down the partially instantiated instance (which is the input to the recursive call). When we unwind this recurrence relation, we get  $S(\phi,n) = O(mn^2)$ , which is indeed  $\text{poly}(n,m)$ , as desired.

Next, we will show how to reduce the membership problem for an arbitrary PSPACE language to TQBF. Let  $L$  be a language in PSPACE and  $M$  a deterministic polynomial-space machine that recognizes  $L$  in space  $s(n)$ . Recall that  $G_{M,x}$  denotes the configuration graph of  $M$  on input  $x$ . If  $|x| = n$ , then each node  $C \in V(G_{M,x})$  is encoded by a bitstring of

length  $c \cdot s(n)$ , for some constant  $c$ , and  $x \in L$  if and only if there is a path in  $G_{M,x}$  from the (unique) start configuration  $C_{start}^{M,x}$  to the (unique) accept configuration  $C_{accept}^{M,x}$ . (Note that  $c \cdot s(n)$  is polynomial in  $n$ .) We will define a family of QBFs  $\Psi_i$  such  $\Psi_i(C, C')$  is true if and only if there is a path in  $G_{M,x}$  from  $C$  to  $C'$  of length at most  $2^i$ . Then the overall formula that is in TQBF if and only if  $x \in L$  is  $\Phi_{c \cdot s(n)}(C_{start}^{M,x}, C_{accept}^{M,x})$ , and we will have to argue that it can be constructed in polynomial time. (The maximum value of  $i$  that we would have to consider is  $c \cdot s(n)$ , because there are  $2^{c \cdot s(n)}$  possible configurations, *i.e.*,  $2^{c \cdot s(n)}$  nodes in  $G_{M,x}$ .) The family  $\Psi_i$  will be defined recursively.

Let  $k$  be the (constant) number of tapes of  $M$ ,  $\Gamma$  be the alphabet of  $M$ ,  $S$  be the state set of  $M$ , and  $\delta$  be the transition function of  $M$ . An element  $C$  of  $V(G_{M,x})$  describes the configuration of  $M$  at a particular time step, say  $t$ , in its computation on input  $x$ ; that is,  $C$  contains all of the (non-blank) symbols that are on  $M$ 's tapes at time  $t$ , the positions of the  $k$  tape heads at time  $t$ , and the state that  $M$  is in at time  $t$ . The formula  $\Psi_0(C, C')$  should be true if and only if one application of the transition function  $\delta$  to the configuration encoded by  $C$  produces the configuration encoded by  $C'$ . The techniques developed in the proof of the Cook-Levin Theorem can be used straightforwardly to show that there is such a formula and that, moreover, it can be produced in time  $\text{poly}(|C| + |C'|) = \text{poly}(n)$ .

For  $0 < i \leq c \cdot s(n)$ , we have (as in the proof of Savitch's Theorem),

$$\Psi_i(C, C') \longleftrightarrow \exists C'' (\Psi_{i-1}(C, C'') \wedge \Psi_{i-1}(C'', C')). \quad (1)$$

Unfortunately, the recursive definition in (1) produces a  $\Psi_i$  that is twice as long as  $\Psi_{i-1}$ , and this would not yield a polynomial-length formula  $\Psi_{c \cdot s(n)}$ . Instead, we use the following recursive definition, in which  $|\Psi_i|$  is  $|\Psi_{i-1}| + \text{poly}(n)$ :

$$\begin{aligned} \Psi_i(C, C') &\longleftrightarrow \exists C'' \forall D_1 \forall D_2 \\ &(((D_1 = C \wedge D_2 = C'') \vee (D_1 = C'' \wedge D_2 = C')) \longrightarrow \Psi_{i-1}(D_1, D_2)). \end{aligned}$$

To see why this definition is equivalent to (1), think through and put into words what the quantified formula that contains  $\Psi_{i-1}$  means: There is a  $C''$  such that, if  $D_1 = C$  and  $D_2 = C''$ , then  $\Psi_{i-1}(D_1, D_2)$ , and, if  $D_1 = C''$  and  $D_2 = C'$ , then  $\Psi_{i-1}(D_1, D_2)$ . (If the disjunction  $p \vee q$  implies  $r$ , then  $p$  implies  $r$ , and  $q$  implies  $r$ .) So there is a "midpoint"  $C''$  such that, if you need to start at  $C$  and get to  $C''$ , you can do so with a path of length at most  $2^{i-1}$ , and, if you need to start at  $C''$  and get to  $C'$ , you can do so with a path of length at most  $2^{i-1}$ .