In writing formulas, it is convenient to use three other operators that are defined in terms of the above operators.

- If φ is a path formula, then $A\varphi$ is the state formula $\neg E \neg \varphi$, so $M, u \models A\varphi$ iff $M, xx \models \varphi$ for all paths xx starting at u.
- If φ is a path formula, then $F\varphi$ is the path formula **true** $U\ \varphi$, so $M, xx \models F\varphi$ iff there exists an $i >= 0$ such that $M, xx^i \models \varphi$.
- If φ is a path formula, then $G\varphi$ is the path formula $\neg F \neg \varphi$, so $M, xx \models G\varphi$ iff $M, xx^i \models \varphi$ for all $i >= 0$.

Theorem 1. *The satisfiability problem for CTL^* (and hence for $YAPL$) and for CTL^*_f is hard for deterministic doubly exponential time, with respect to logspace reducibility.*

Theorem 5.2. Proof. We give the proof first for CTL^*. Essentially the same construction works for CTL^*_f. From Fact 4.?, if a language can be accepted in deterministic doubly exponential time then the language can be accepted by an ATM with space bound 2^{cn} for some integer constant $c > 0$. (ATM's are defined and discussed before the proof of Theorem 4.??.) Let Z be such an ATM and let y be an input to Z of length n. The proof is done by constructing a CTL^* formula φ_y of length polynomial in n such that Z accepts y iff φ_y is satisfiable. We describe informally what the parts of φ_y should express about the model. Translation into CTL^* formulas is straightforward, although we give the translation for several cases. The construction is similar to the proof of Theorem 4.??. Even though the present proof is self-contained, we point out the analogies with the proof of Theorem 4.?? for the benefit of the reader. As in the lower bound proved for PDL by Fischer and Ladner [FL79], satisfying models of φ_y correspond to accepting computation trees of Z on input y. However, we cannot represent an ID of Z by a state in a Kripke structure as in [FL79], since in our case ID's are of length 2^{cn}. Instead, we represent the contents of a single tape cell by a state in a structure, an ID is represented by a path of 2^{cn} states, and computation paths are obtained by concatenating ID's. The formula contains propositional variables that encode symbols of the alphabet Σ used in the proof of Theorem 4.??. Specifically, the formula contains variables $S_1, \ldots, S_d$ where the constant d is chosen large enough that these variables can encode symbols of $\Delta = \Gamma \cup (Q \times \Gamma)$, where Z by a state in a Kripke structure as in [FL79], since in our case ID's are of length 2^{cn}. Instead, Q (Γ) is the set of states (tape symbols) of Z. In addition, the variable D indicates existential (D true) or universal (D false) ID's, the variable V indicates whether this ID follows from the previous one by the first (V true) or second (V false) move described by Z'transition function, and the variable I indicates whether this state represents part of the computation tree (I true) or does not (I false); the latter case where I is false corresponds to the null symbol in the proof of Theorem 4.??. Given a structure (W, R, PI), we say that a state v is an R- *successor* of a state u if $(u, v) \in R$. Parts of the formula will ensure that I is true at the root of the model, if I is true at a state u then I is true at some R-successor of u and false at some R-successor of u, and if I is false at a state u then I is false at all

R-successors of u. The variables $C_1, \ldots, C_{cn}$ represent a counter; the truth value of C_i at a state gives the ith binary digit of the value of the counter at that state, where true (false) corresponds to 1 (0). Symbols of each ID are numbered consecutively from 0 to $2^{cn} - 1$. Given a structure $M = (W, R, PI)$ and a state $u \in W$, we associate $S(u)$, the symbol encoded by $S_1, \ldots, S_d$ at state u, and $C(u)$, the integer between 0 and $2^{cn} - 1$ represented by the counter at u. $D(u)$ denotes the truth value of D in state u, and similarly for $V(u)$ and $I(u)$. For each symbol $\sigma \in \Delta$, we let " $S = \sigma$" abbreviate a formula that is true in state u iff $S(u) = \sigma$, so " $S = \sigma$"is a conjunction of variables $S_1, \ldots, S_d$ or their negations. We also use the abbreviations " $C = 0$"for $\neg C_1 \wedge \ldots \wedge \neg C_{cn}$ and " $C = 2^{cn} - 1$"for $C_1 \wedge \ldots \wedge C_{cn}$. (Note that $C = 2^{cn} - 1$ at a state means that the state holds the last symbol of some ID.) The formula φ_y has the form $r \wedge AGg$, where the state formula r expresses properties of the root of the structure M, and the state formula g expresses properties that should hold at every state u of M. Some of the conjuncts of g ensure that the counter counts correctly, that is, if $(u, v) \in R$ then $C(v) = C(u) + 1 \pmod{2^{cn}}$. This is expressed by adding as conjuncts to g the formula $A(\neg(C_1 == XC_1))$ and for each $2 \leq k \leq cn$ the formula $A((C_k == XC_k) == (\neg C_1 \vee \neg C_2 \vee \ldots \vee \neg C_{k-1}))$. Having defined the counter, we can now describe the root properties: $r = D \wedge C = 0 \wedge init$. The state formula *init* says that there is a path $x_0, x_1, \ldots, x_m, \ldots$ starting at the root x_0 such that $I(x_i)$ is true for $0 \leq i < m$, $C(x_0) = 0$, $C(x_m) = 0$, $C(x_i) \neq 0$ for $0 < i < m$, and $S(x_0)S(x_1) \ldots S(x_{m-1}) = (q_0, y_1)y_2 \ldots y_n \# \ldots \#$where q_0 is the initial state of Z and $\#$ is the blank symbol. This is expressed by the formula $E(S = (q_0, y_1) \wedge X(S = y_2 \wedge X(S = y_3 \wedge X(\ldots \wedge X(S = y_n \wedge X(S = \# \ U \ C = 0))) \ldots) \wedge I \wedge X(I \ U \ C = 0))$. We now continue describing the properties expressed by the conjuncts of g. The following properties (1)-(5) ensure that the variables I, D and V behave correctly. (These are similar to the properties (A2)-(A5) that are checked by the ta in the proof of Theorem 4.??.) These properties are easily expressed as CTL^* formulas using the operators A, E, and X. We give the formulas for properties (1) and (2) as examples, and leave (3)-(5) to the reader. (1) If $\neg I(u)$ then $\neg I(v)$ for every R-successor v of u. We add as a conjunct of g the formula $\neg I \rightarrow A(X\neg I)$. (2) If $I(u)$ then there exist R-successors v and w of u such that $I(v)$ and $\neg I(w)$. We add to g the formula $I \rightarrow (EXI \wedge EX\neg I)$. (3) The truth values of D and V do not change within the same ID. If $C(u) \neq 2^{cn} - 1$ then $D(u) == D(v)$ and $V(u) == V(v)$ for every R-successor v of u. (4) Since we are assuming that the ATM Z alternates at each step, the truth value of D should change when moving from one ID to the next one(s). If $C(u) = 2^{cn} - 1$ then $\neg(D(u) == D(v))$ for every R-successor v of u. (5) After every universal ID (D false), the computation tree should branch into the two successor ID's. If $C(u) = 2^{cn} - 1 \wedge I(u) \wedge \neg D(u)$, then $V(v) \wedge I(v)$ for some R-successor v of u and $\neg V(w) \wedge I(w)$ for some R-successor w of u. It remains to ensure that the ID symbols follow according to the transition rules of Z. Recall from the discussion of ATM'sinSection 4 that we can check that the ID *beta* is the i-successor of the ID α by checking that every three consecutive symbols of α and the three symbols in the same three positions in β satisfy a

particular 6-ary relation R_Z, i. To do this checking, we use paths of a special form in satisfying structures of φ_y. A *good path starting in state* u is an infinite path $xx = x_0, x_1, \ldots, x_m, x_m + 1, \ldots$ starting in $u = x_0$ such that

GP1. $I(x_i)$ is true for $0 \leq i \leq m$,
GP2. $I(x_i)$ is false for $i > m$,
GP3. $m > 0$,
GP4. $C(x_0) = C(x_m)$,
GP5. $C(x_0) \neq C(x_i)$ for $0 < i < m$,
GP6. $C(x_0) \neq 2^{cn} - 1$ and $C(x_0)$ ne $2^{cn} - 2$.

The portion of xx from x_0 to x_m is called the *I- portion* of the path. Note that properties (1) and (2) ensure that a good path exists starting in every state x_0 that satisfies (GP6). Good paths are useful since the states x_0 and x_m belong to the same position in two consecutive ID's, and the first three states of the path belong to the same ID. Moreover, since x_m is the last state on xx where I is true, we can force certain formulas to be true at state x_m. Specifically, if h is a formula, then we define $lastI(h)$ to be the formula $G((I \wedge X\neg I) \rightarrow h)$. If h is a state formula, then $M, xx \models lastI(h)$ iff $M, x_m \models h$. Similarly, if h is a path formula, then $M, xx \models lastI(h)$ iff $M, xx^m \models h$. The conjunct of g that checks ID symbols has the form $A(\neg badpath \rightarrow check)$, where the formula *badpath* is true of all bad (i.e., not good) paths, and where the formula *check* checks that the ID symbols in all paths of length three starting in x_m follow correctly from the ID symbols in states x_0, x_1 and x_2. The formula *badpath* is a disjunction of several subformulas described next:

BP1 the length of the I-portion of the path is either less than two or infinite: $\neg I \vee X \neg I \vee GI$,
BP2 (GP6)is violated: $C = 2^{cn} - 1 \vee X(C = 2^{cn} - 1)$,
BP3 the counter equals $2^{cn} - 1$ twice (or more) on the I-portion of the path: $F((C = 2^{cn} - 1 \wedge I) \wedge XF(C = 2^{cn} - 1 \wedge I))$,
BP4 $C(x_0) \neq C(x_m)$: this formula is the disjunction overall $1 \leq k \leq cn$ of $\neg(C_k == lastI(C_k))$.

Let W_i denote V if $i = 1$ or $\neg V$if $i = 2$. The formula *check* is the disjunction, over $i = 1, 2$ and overall 6-tuples $(\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6) \in R_Z, i$ of the formulas $\sigma_1 \wedge X(S = \sigma_2) \wedge XX(S = \sigma_3) \wedge lastI(W_i) \wedge lastI(A(S = \sigma_4 \wedge X(S = \sigma_5) \wedge XX(S = \sigma_6)))$. This completes the description of the formula φ_y. It is straightforward to check that Z accepts y iff f_y is satisfiable. In particular, since the formula $(\neg badpath \rightarrow check)$ must hold for every path starting in every state of the structure, it is easy to see that all ID symbols follow according to the transition rules of Z in the portion of the structure that represents the accepting computation tree of Z, i.e., the portion on which I is true. In the case of CTL^*_f, it is still true that if φ_y is satisfiable then Z accepts y, because finite good paths are sufficient to check that ID symbols behave properly. For the other direction (Z accepts y implies that φ_y is satisfiable), a little care must be taken when using the A operator. For example, in the formula used to ensure property (1),

the subformula $A(X\neg I)$ is false at every state u of every structure since $(X\neg I)$ does not satisfy the path consisting of the state u alone. To fix this, note that $(X\ \mathbf{true})$ satisfies a path xx iff xx has at least two states. Therefore, we replace the subformula with $A((X\ \mathbf{true}) \ \rightarrow\ X\neg I)$. With modifications of this type, essentially the same proof works for CTL^*_f. Remark. If the length of y is n and if we take the length of every atomic proposition symbol in φ_y to be 1, then the length of φ_y is O(n^2). Using standard methods (see, for example, Chap. 11 of [AHU74]) this implies a lower bound on the deterministic time complexity of CTL^* that grows as a doubly exponential function of $c\sqrt{n}$ for some constant $c > 0$. The only part of φ_y whose length is not O(n) is the formula that checks that the counter counts correctly. There are other definitions of counters (such as the one used in Chapter 4 of [S74]) that can be used in this proof and such that the correctness of the counter can be expressed by a formula of length O(n). This improves the lower bound to a doubly exponential function of cn. The convention that all proposition symbols have length 1 is the convention used in the upper bound of Theorem 5.1. If we adopt the convention that different proposition symbols are distinguished by subscript, that subscripts are written in binary and that the length of subscripts is included in the length of formulas, then the lower bound becomes doubly exponential in $cn/log(n)$.