Lecture #11: The Cook-Levin Theorem Proofs of Claims

This document includes of claims, from Lecture #11, that are needed to complete this proof of the Cook-Levin Theorem. It is ***for interest only*** — students do not need to read this in order to do well in this course.

1 Proof of a Useful Lemma

While the complexity class $\mathcal{NP}$ was defined using multi-tape nondeterministic Turing machines, the proof the $\mathcal{NP}$-hardness of $L_{\textsf{FSAT}}$ uses one-tape nondeterministic Turing machines instead. The following lemma is needed to establish that this simplification can be made.

Lemma. *Let $L \in \Sigma^\star$ such that $L \in \mathcal{NP}$. Then there exists a deterministic* **one-tape** *Turing machine*[1]

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$$

such that decides L and such that the depth of the computation tree for M and a string ω is bounded by a polynomial function of $|\omega|$, for all $\omega \in \Sigma^\star$.

Sketch of Proof. Let $L \subseteq \Sigma^\star$ such that $L \in \mathcal{NP}$. Then there exists a k-tape nondeterministic Turing machine

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}}),$$

for some positive integer k, that decides L in polynomial time. It follows that there also exist positive integer constants c_1 and d, and a nonnegative integer constant c_0, such that the depth of the computation tree for M and an input string ω is at most $c_1|\omega|^d + c_0$ for every string $\omega \in \Sigma^\star$.

It now suffices to apply a straightforward modification of the construction used to prove Claim #2 in Lecture #3 (which establishes a corresponding result for deterministic Turing machines) to

[1] As with deterministic one-tape Turing machines it will be assumed that the transitions can only specify moves ***left*** or ***right*** for this type of Turing machine.

prove that there exists a nondeterministic one-tape Turing machine

$$\widehat{M} = (\widehat{Q}, \Sigma, \widehat{\Gamma}, \widehat{\delta}, \widehat{q}_0, \widehat{q}_{\text{accept}}, \widehat{q}_{\text{reject}})$$

that also decides L: $\widehat{M}$'s tape represents M's tapes (including their contents and the positions of their tape heads) in exactly the same way as described in the notes for Lecture #3. The initialization phase of the simulation is also identical to the initialization phase in these notes. The only difference in simulating moves is that a "read" phase is used to determine the symbols visible on M's tapes, as in the original simulation — but the machine M must then nondeterministically guess which of the moves allowed by $\widehat{M}$'s transition function is to be applied. The update process, needed to update $\widehat{M}$'s in order to reflect writes onto M's tapes and movements of its tape heads, is then also the same as before.

It is easy to argue that the number of *leaves* in the computation tree for $\widehat{M}$ and an input string $\omega \in \Sigma^\star$ is the same as the number of leaves in the computation tree for M and ω. Furthermore, if the length of the path from the root to some leaf in the computation tree for M and ω is a nonnegative integer T then the analysis of the *number of steps* used in the simulation for deterministic Turing machines can be applied to establish that the length of the path from the root to the corresponding leaf, in the computation tree for $\widehat{M}$ and ω, is at most

- $(2k+1)T^2 + (2k+2)T + k + 2$ if $n = |\omega| = 0$,
- $2nT + 2kT^2 + (2k+3)T + 2n + 1$ if $1 \leq T \leq n = |\omega|$, and
- $(2k+1)T^2 + n^2 + (2k+2)T + n + k$ if $1 \leq n < T$.

This can be used to argue that the depth of the computation tree for $\widehat{M}$ and ω must be in $O(|\omega|^{2d})$, establishing the claim. □

2 Proof of Claim #1

Claim 1. *$L_{\mathsf{F}} \in \mathcal{P}$.*

Sketch of Proof. Consider the execution of the algorithm shown in Figure 1, on page 3, on an input string $\omega \in \Sigma^\star$. A consideration of the details of the step at line 1 should confirm that ω is ***rejected*** during this step unless ω is a concatenation of encodings of Boolean variables and of copies of left and right brackets and of logical operators $\wedge$, $\vee$, and $\neg$. Furthermore, if ω has not been rejected during this step then the string that has been written onto the second tape is simply the string that can be obtained from ω by replacing each encoding of a Boolean variable that is a substring of ω (and that is not followed by another digit) by a copy of the symbol `F`.

On input $\omega \in \Sigma_F^\star \ldots$

1. ***Reject*** if ω is the empty string.

   Otherwise sweep to the right over ω — copying punctuation (brackets) and logical operators onto a second tape. Every time $\mathtt{x}$ is seen, ***reject*** if this is not followed by the unpadded decimal representation of a non-negative integer (processing as many digits as you can). If the input has not been rejected write $\mathtt{F}$ onto the second tape, instead of the encoding of a Boolean variable.

   Do this until all of the input string has been processed, moving back to the beginning of the second tape if the input has not been rejected.

2. If the non-blank string on the second tape has length greater than or equal to two, go to step 3. Go to step 5 otherwise.

3. Sweep over the string on the second tape, copying symbols onto a third tape, except that the following should be changed.

   - If the substring $\neg\mathtt{F}$ is seen on the second tape, write $\mathtt{F}$ onto the third tape instead.

   - If a substring $(\mathtt{F} \wedge \mathtt{F} \wedge \cdots \wedge \mathtt{F})$ — consisting of two or more copies of $\mathtt{F}$, separated by copies of $\wedge$, and enclosed by brackets — is seen on the second tape, write $\mathtt{F}$ onto the third tape instead.

   - If a substring $(\mathtt{F} \vee \mathtt{F} \vee \cdots \vee \mathtt{F})$ — consisting of two or more copies of $\mathtt{F}$, separated by copies of $\vee$, and enclosed by brackets — is seen on the second tape, write $\mathtt{F}$ onto the third tape instead.

4. ***Reject*** if the entire string on the second tape was processed, and the string written onto the third tape is the same as the one on the second.

   Otherwise, replace the string with the second tape with the one on the third, erasing the contents of the third tape, and moving the tape heads for both tapes back to the left. Then go back to step 2.

5. If the non-blank string on the second tape is $\mathtt{F}$ then ***accept***. Otherwise, ***reject***.

Figure 1: A Deterministic Algorithm Deciding Membership in L_F

Now, if $\omega \in L_{\mathsf{F}}$ then it can be shown that this algorithm accepts ω — by induction on the length of a “derivation” of the Boolean formula $\mathcal{F}$ encoded by ω, using the inductive definition of Boolean formulas that is included in the lecture notes.

On the other hand, the algorithm is executed with ω as input, and ω is accepted, then the step at line 3 must be executed some finite number of times, before this happens, and it is possible to prove that ω is an encoding of a Boolean formula, so that $\omega \in L_{\mathsf{F}}$, by induction on the number of times that the step at line 3 is executed when ω is processed.

Thus the language of a Turing machine implementing this algorithm is L_{F}. If this Turing machine halts when executed on input ω, for all $\omega \in \Sigma_F^\star$, then this Turing machine decides L_{F}.

It therefore remains only to prove that this Turing machine halts whenever it is executed on an input string $\omega \in \Sigma^\star$, using a number of moves that is bounded by a polynomial function of $|\omega|$.

The first step can each be carried out using sweeps to the moving right over the input, while moving right on the second tape, and then moving back to the left on the second tape. The non-blank string written onto the second tape cannot be longer than the input, so the number of moves needed for this is certainly at most linear in $|\omega|$.

The second step can certainly be implemented so that each execution of this step requires at most a polynomial number of moves. With a bit of work one can argue that the number of moves needed to execute the step at line 3 is at most linear in the length of the non-blank string that is on the second tape when the execution of this step begins — and that the number of moves needed for the execution of the step at line 4 immediately after that is at most linear in the length of this string as well.

Note, now, that the non-blank string on the second string is replaced by a strictly *shorter* non-blank string every time this pair of moves is carried out. It follows that these steps are each executed at most $|\omega|$ times. Each execution requires $O(|\omega|)$ moves, so that the total number of moves needed for all executions of the steps at lines 2–4 is in $O(|\omega|^2)$.

The step at line 5 requires only a constant number of moves. It follows that a Turing machine implementing this algorithm decides L_{F}, using $O(|\omega|^2)$ moves when executed on an input string $\omega \in \Sigma^\star$. Thus $L_{\mathsf{F}} \in \mathcal{P}$, as claimed. $\square$

3 Proof of Claim #2

Claim 2. *$L_{\mathsf{FSAT}} \in \mathcal{NP}$.*

Sketch of Proof. Consider the algorithm that begins in Figure 2 on page 5 and ends in Figure 3 on page 6. In particular, consider an execution of this algorithm on an input string $\mu \in \widehat{\Sigma}^\star$, where the alphabet $\widehat{\Sigma}$ is as defined in the lecture notes.

A consideration of the first two moves should confirm that μ is ***rejected*** using time linear in the length of its input if the input does not begin with ω# where $\omega \in \Sigma_F^\star$. It follows by Claim 1 that the input is also rejected — using a number of moves that is at most polynomial in $|\omega|$ — if the

On input $\mu \in \widehat{\Sigma}^\star \ldots$

1. ***Reject*** if μ is the empty string. Otherwise, sweep right over μ, copying symbols in μ onto the second tape, as long as these symbols belong to Σ_F — ending when some symbol that is not in Σ_F is seen on the first tape.

   Reject if the symbol that is now visible on the first tape is not #. Otherwise move the tape head for the second tape back to the leftmost cell on this tape, while moving right *once* on the first tape (so that the symbol visible is the one immediately to the right of #).

2. Let $\omega \in \Sigma_F^\star$ be the string that has now been copied onto the second tape. ***Reject*** if $\omega \notin L_{\mathsf{F}}$. Otherwise move the tape head for the second tape back to the leftmost cell.

3. While the symbol visible on the first tape is not blank, copy this symbol onto the third tape, moving right on the first, second and third tapes, in order to check the relationship between $|\omega|$ and the number of symbols that have been copied onto the third tape.

   Reject if either a symbol that would be copied onto the third tape is not in Σ_C or it is discovered the input string includes more than $|\omega| + 2$ symbols after #.

4. Let ν be the non-blank string in $\Sigma_C^\star$, with length at most $|\omega| + 2$, that has now been copied onto the third tape. ***Reject*** unless ν is an encoding of a subset $\mathcal{S}$ of the set $\mathcal{V}$ of Boolean variables such that

   - each Boolean variable x_i, included in $\mathcal{S}$, appears in the Boolean formula $\mathcal{F}$ encoded by ω, and
   - the encodings of Boolean variables included in ν are sorted by increasing index.

   It the input has not been rejected, by now, let $\mathcal{F}$ be the Boolean formula encoded by ω and let $\varphi : \mathcal{V} \to \{\mathtt{T}, \mathtt{F}\}$ be the truth assignment encoded by ν.

Figure 2: Beginning of a Verification Algorithm for L_{FSAT}

input *does* begin with a string ω#, for $\omega \in \Sigma_F^\star$, but $\omega \notin L_{\mathsf{F}}$.

Suppose, now, that the input string *does* begin with ω# for a string $\omega \in L_{\mathsf{F}}$. A consideration of the step at line 3 confirms that the input is ***rejected***, using a number of moves that is at most linear in $|\omega|$, unless $\mu = \omega\#\nu$ for a string $\nu \in \Sigma_C^\star$ such that $|\nu| \leq |\omega| + 2$. Furthermore, if the

5. Sweeping over the strings on the second and third tapes, as needed, copy the symbols on the second tape (encoding $\mathcal{F}$) onto the fourth tape — except that the encoding of each Boolean variable x_i that appears in $\mathcal{F}$ should be replaced on the fourth tape by its truth value under φ, $\varphi(x_i) \in \{\mathtt{T}, \mathtt{F}\}$, as given by the string on the third tape.

6. If the non-blank string on the fourth tape has length at least two then go to step 7. Otherwise, go to step 9.

7. Sweep over the string on the fourth tape, copying symbols onto the fifth tape, except that the following should be changed.

    - If the substring $\neg\mathtt{T}$ is seen on the fourth tape, write $\mathtt{F}$ onto the fifth tape instead. If the substring $\neg\mathtt{F}$ is seen on the fourth tape, write $\mathtt{T}$ on the fifth tape instead.

    - If a substring $(w_1 \wedge w_2 \wedge \cdots \wedge w_k)$ is seen on the fifth tape, where $k \geq 2$ and $w_i \in \{\mathtt{T}, \mathtt{F}\}$ for $1 \leq i \leq k$, then write $\mathtt{T}$ on the fifth tape, instead, if $w_i = \mathtt{T}$ for all i such that $1 \leq i \leq k$, and write $\mathtt{F}$ on the fifth tape instead, otherwise.

    - If a substring $(w_1 \vee w_2 \vee \cdots \vee w_k)$ is seen on the fifth tape, where $k \geq 2$ and $w_i \in \{\mathtt{T}, \mathtt{F}\}$ for $1 \leq i \leq k$, then write $\mathtt{F}$ on the fifth tape, instead, if $w_i = \mathtt{F}$ for all i such that $1 \leq i \leq k$, and write $\mathtt{T}$ on the fifth tape instead, otherwise.

8. Replace the non-blank string on the fourth tape with the non-blank string on the fifth tape, erasing the non-blank string in the fifth tape in the process, and moving the tape heads for each tape to the leftmost cell of the tape.

9. If the nonblank string on the fourth tape is "$\mathtt{T}$" then ***accept***. Otherwise, ***reject***.

Figure 3: Continuation of a Verification Algorithm for L_{FSAT}

input has not been rejected then ν has been copied onto the third tape.

Note, now, that if ν encodes a set $\mathcal{S}$ of Boolean variables then ν is simply the concatenation of encodings of Boolean variables, separated by commas and enclosed by set brackets; it is certainly possible to check whether this is the case (rejecting, if it is not) using time linear in $|\nu|$. Indeed, this test can be carried out using a sweep to the right over the copy of ν on the third tape. If a fourth tape is used to store an encoding of a Boolean variable already found on the third tape, then one can also confirm that the variables included in the encoding are distinct, and sorted by increasing order, by comparing the indices in adjacent encodings. Finally, one can check that each variable, whose encoding is included in ν, is used in $\mathcal{F}$ by sweeping over ω — using a number of moves in $O(|\omega|)$ for each variable that must be checked. Since the number of such variables is certainly at most $|\nu|$, it follows that test at step 4 can be carried out using $O(|\mu| \times |\omega|) \subseteq O(|\omega|^2)$ moves.

Now, either the input has been rejected before an execution of step 4 ends — and the requirements for a verification algorithm are satisfied — or it has been confirmed that the input includes an encoding ω of a Boolean formula $\mathcal{F}$, and an encoding μ of a truth assignment φ. The string ν is a certificate for ω if and only if $\varphi(\mathcal{F}) = \mathtt{T}$ — so it is sufficient, now, to argue that the remaining steps of the algorithm cause the input string to be ***accepted*** if $\varphi(\mathcal{F}) = \mathtt{T}$ and cause it to be ***rejected*** otherwise — using a number of moves that is at most polynomial in $|\omega|$.

Consider the step at line 5. Since this replaces each variable x_i in $\mathcal{F}$ with its truth value $\varphi(x_i)$, this produces a "Boolean expression" — a logical function of the Boolean constants $\mathtt{T}$ and $\mathtt{F}$, produced using the logical operators $\wedge$, $\vee$, and $\neg$, whose *value* is the truth value $\varphi(\mathcal{F})$ under φ. As the step notes this expression can be produced by sweeping over ω, writing symbols onto another tape, replacing each variable x_i with its truth value $\varphi(x_i)$ — which can be found by sweeping over the encoding ν of φ: If this includes the encoding of x_i as a substring (followed by a comma or right bracket) then $\varphi(x_i) = \mathtt{T}$; $\varphi(x_i) = \mathtt{F}$, otherwise. Now, each truth value can be obtained using $O(|\nu|)$ moves and, since there are at most $|\omega|$ truth values to look for, the total number of moves needed to carry out this step is in $O(|\omega| \times |\nu|) \subseteq O(|\omega|^2)$.

Steps 7 and 8 repeatedly replace the Boolean expression, whose encoding is on the fourth tape, with a strictly shorter expression with the same truth value. It can be argued (with a bit of work) that the number of moves needed to execute this pair of steps *once* is at most linear in the length of the Boolean expression that is on the fourth tape before the operations are carried out. Since the expression is shortened every time the steps are executed one can see, by an examination of step 6, that the *number of times* they are executed is at most the length of the expression originally on the fourth tape — which is certainly at most $|\omega|$. It follows that the total number of moves used for all executions of the steps at lines 6–8 is in $O(|\omega|^2)$.

When the step at line 9 is reached, the expression on the fourth tape is a Boolean expression with length one — so that it is either $\mathtt{T}$ or $\mathtt{F}$ — with the same value as $\varphi(\mathcal{F})$. Thus an input of the form $\omega\#\nu$ (where $\omega \in \Sigma_F^\star$ and $\nu \in \Sigma_C^\star$) is accepted if and only if ν is a certificate for ω. When the input has this form the number of moves used is in $O(|\omega|^2)$ so that this is a polynomial-time verification algorithm for L_{FSAT}.

It follows that $L_{\mathsf{FSAT}} \in \mathcal{NP}$, as claimed. $\square$

4 Proof of Claim #3

Claim 3. $L_{\textsf{FSAT}}$ *is* $\mathcal{NP}$*-hard.*

4.1 Overview of Construction

In order to prove Claim 3 it is necessary, and sufficient, to prove that $L \preceq_{\textsf{P, M}} L_{\textsf{FSAT}}$ for an arbitrarily chosen language $L \in \mathcal{NP}$. Thus one should consider a one-tape nondeterministic Turing machine

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\textsf{accept}}, q_{\textsf{reject}})$$

that decides the language $L \subseteq \Sigma^\star$ such that, for every input string $\omega \in \Sigma^\star$, the computation tree for M and ω has depth at most

$$T(|\omega|) = c_1|\omega|^d + c_0$$

for positive integer constants c_0, c_1 and d. Increasing the time bound, we may assume that $c_0 \geq 4$.

As described in the lecture notes, a ***tableau*** — a two-dimensional table with $T(|\omega|) + 1$ rows and $T(|\omega|) + 3$ columns, whose cells store values in $Q \cup \Gamma \cup \{\texttt{\#}\}$ — can then be defined, as an aid to the construction and analysis of a Boolean formula

$$\mathcal{F}_\omega = (\mathcal{F}_{\textsf{cell}} \wedge \mathcal{F}_{\textsf{start}} \wedge \mathcal{F}_{\textsf{accept}} \wedge \mathcal{F}_{\textsf{move}})$$

where each of the subformulas in $\mathcal{F}_\omega$ represents a requirement, as follows.

- $\mathcal{F}_{\textsf{cell}}$ represents the requirement that there is ***exactly*** one symbol from $Q \cup \Gamma \cup \{\texttt{\#}\}$ in each cell of the tableau. As stated in the lecture notes, if $|\omega| = n$ then this can be achieved by setting[2] $\mathcal{F}_{\textsf{cell}}$ to be

$$\bigwedge_{0 \leq i \leq T(n)} \left(\bigwedge_{0 \leq j \leq T(n)+3} \left(\left(\bigvee_{\sigma \in Q \cup \Gamma \cup \{\#\}} x_{i,j,\sigma} \right) \wedge \left(\bigwedge_{\substack{\sigma_1, \sigma_2 \in Q \cup \Gamma \cup \{\#\} \\ \sigma_1 \neq \sigma_2}} (\neg x_{i,j,\sigma_1} \vee \neg x_{i,j,\sigma_2}) \right) \right) \right)$$

[2]In the presentation, Boolean variables $x_{i,j,\sigma}$ are used where $0 \leq i \leq T(|\omega|)$, $0 \leq T(|\omega|)+3$, $\sigma \in Q \cup \Gamma \cup \{\texttt{\#}\}$, and this variable has value `T` if the cell in row i and column j stores σ. As discussed in the lecture notes these variables can replaced "on the fly" with variables in the set $\mathcal{V} = \{x_0, x_1, x_2, \dots\}$ — so the new variables are used to improve readability but do not significantly change the proof.

- $\mathcal{F}_{\text{start}}$ represents the requirement that the top row of the tableau represents the initial configuration of M on input ω. If

  $$\omega = \tau_1\tau_2\dots\tau_n$$

  where $\tau_1, \tau_2, \dots, \tau_n \in \Sigma$ then it suffices to set $\mathcal{F}_{\text{start}}$ to be

  $$x_{0,0,\#} \wedge x_{0,1,q_0} \wedge \left(\bigwedge_{1 \le h \le n} x_{0,h+1,\tau_h} \right) \wedge \left(\bigwedge_{n+2 \le h \le T(n)+2} x_{0,h,\sqcup} \right) \wedge x_{0,T(n)+3,\#}$$

- $\mathcal{F}_{\text{accept}}$ represents the requirement that the *bottom* row of the tableau represents an accepting configuration. If the subformula $\mathcal{F}_{\text{move}}$ represents the property that it is supposed to — as described below — then it suffices to set $\mathcal{F}_{\text{accept}}$ to be

  $$\bigvee_{1 \le h \le T(n)+2} x_{T(n),h,q_{\text{accept}}}$$

Note that, since M is a ***fixed*** nondeterministic Turing machine, c_0, c_1 and d (used to define the function T) are ***fixed*** positive integers, and the structures of each of the above subformulas, it can be argued that encodings of all three of these subformulas can be computed deterministically from a given string $\omega \in \Sigma^\star$ using time at most polynomial in $|\omega|$.

It remains only to consider the final subformula, $\mathcal{F}_{\text{move}}$, which represents the requirement that *every* row of the tableau represents a configuration of M and, furthermore, if $i \in \mathbb{N}$ and $0 \le i \le T(n)$, then either

- rows i and $i+1$ represent configurations η_i and η_{i+1}, respectively, such that $\eta_i \vdash_M \eta_{i+1}$, or
- row i represents a ***halting*** configuration η_i of M, and row $i+1$ represents η_i too.

Windows were then introduced as structures giving conditions concerning the contents of six cells — in two rows, and three columns — of the tableau. In particular, if $C_{i,j}$ denotes the contents in row i and column j of the tableau, for $0 \le i \le T(n)$ and $0 \le j \le T(n)+3$, then the *window* $W_{r,s}$ identifies the contents of cells $C_{r,s}$, $C_{r,s+1}$, $C_{r,s+2}$, $C_{r+1,s}$, $C_{r+1,s+1}$ and $C_{r+1,s+2}$ — and is defined for all integers r and s such that $0 \le r \le T(n) - 1$ and $0 \le j \le T(n) + 1$.

As described in the lecture notes, windows are shown pictorially. In particular, the condition that

$$C_{i,j} = \alpha_1, C_{i,j+1} = \alpha_2, C_{i,j+2} = \alpha_3, C_{i+1,j} = \beta_1, C_{i+1,j+1} = \beta_2, \quad \text{and} \quad C_{i+1,j+2} = \beta_3, \quad (1)$$

can be shown as a window $W_{i,j}$ that is drawn as follows:

α_1	α_2	α_3
β_1	β_2	β_3

(2)

Each window can specify one of $(|Q| + |\Gamma| + 1)^6$ combinations of values for the contents of the cells it involves.

Note that the condition at line (1) is represented by the subformula

$$(x_{i,j,\alpha_1} \wedge x_{i,j+1,\alpha_2} \wedge x_{i,j+2,\alpha_3} \wedge x_{i+1,j,\beta_1} \wedge x_{i+1,j+1,\beta_1} \wedge x_{i+1,j+2,\beta_3})$$

Several of the (combinations of values for) windows will be defined to be ***illegal*** because they can only arise in tableaux representing a sequence of configurations that do not correspond to valid moves of the Turing machine M; the other (combinations of values for) windows will be defined to be ***legal***.

Since there are only finitely many (combinations of values for) windows, there is a fixed set of ***legal*** (combinations of values for) windows. Thus a Boolean formula

$$\text{legal}_{i,j}$$

representing the claim that the (contents of values in) window $W_{i,j}$ is legal, is also easily described: It is the "disjunct," or ***or***, of a fixed number of subformulas that look like the above one, enclosed in parentheses.

A proof that L_{FSAT} is $\mathcal{NP}$-hard can be concluded by establishing the following.

- If $\mathcal{F}_{\text{move}}$ is defined as

$$\mathcal{F}_{\text{move}} = \bigwedge_{0 \le i \le T(n)-1} \left(\bigwedge_{0 \le j \le T(n)+1} \text{legal}_{i,j} \right) \tag{3}$$

  (after specifying the set of windows that should be considered to be legal) then $\mathcal{F}_{\text{move}}$ satisfies the properties that have been identified for it.

- The string $e(\mathcal{F}_{\text{move}})$ can be computed deterministically from ω using time that is at most polynomial in $|\omega|$.

The proof in the lecture notes continued by introducing the following *conditions*.

1. Every row begins and ends with # and there are no other copies of # in the row.
2. Every row includes exactly one cell whose contents is a *state*.
3. The change in state, contents of the tape cell initially visible, and change in tape head location when going from the configuration in one row to the configuration in the next row, is consistent with M's transition function — and all such changes, that are consistent with the transition function, are allowed.

4. No symbol to the left of the position of the tape head is changed when going from the configuration in one row to the configuration in the row after that.
5. No symbol to the right of the position of the tape head is changed when going from the configuration in one row to the configuration in the row after that.

These conditions are handled by refining the set of legal (and illegal) windows in such a way that — when subformulas $\mathcal{F}_{\text{cell}}$, $\mathcal{F}_{\text{start}}$ and $\mathcal{F}_{\text{accept}}$ are also satisfied — they are achieved if and only if the subformula $\mathcal{F}_{\text{move}}$ is satisfied, for $\mathcal{F}_{\text{move}}$ as at line (3).

4.2 Achieving the First Condition

Consider the first condition — that every row begins and ends with # and there are no other copies of # in the row.

In order to satisfy this, let us require that all windows, as shown at line (2), such that either

i. *exactly* one of α_1 or β_1 is $\#$, or
ii. *exactly* one of α_3 or β_3 is $\#$, or
iii. β_2 is $\#$

are ***illegal***.

Lemma 4. *Let φ be any truth assignment such that*

$$\varphi(\mathcal{F}_\omega) = \mathtt{T},$$

where $\mathcal{F}_\omega$ is as described above, and where the windows described above are **illegal***, as specified there.*

Then φ specifies a well-defined assignment of values in $Q \cup \Gamma \cup \{\#\}$ to the cells in a two-dimensional table with $T(|\omega|) + 1$ rows and $T(|\omega|) + 4$ columns, such that the contents of each row begin and end with #, with no copy of # in between.

Proof. Let φ be any truth assignment such that $\varphi(\mathcal{F}_\omega) = \mathtt{T}$. Since $\varphi(\mathcal{F}_{\text{cell}}) = \mathtt{T}$, φ specifies a well-defined assignment of values in $Q \cup \Gamma \cup \{\#\}$ to the cells in a two-dimensional table with $T(|\omega|) + 1$ rows and $T(|\omega|) + 3$ columns. It therefore suffices to show that the following ***property*** is satisfied for ever integer i such that $0 \leq i \leq T(|\omega|)$:

> The contents of the i^{th} row of the table begin and end with # and do not have any other copies of # in-between.

This can be proved by ***induction*** on i. The standard form of mathematical induction can be used.

Basis: An inspection of the subformula $\mathcal{F}_{\text{start}}$ confirms that, since $\varphi(\mathcal{F}_{\text{start}}) = \mathtt{T}$, the property is satisfied when $i = 0$.

Inductive Step: Let i be an integer such that $0 \leq i \leq T(|\omega|) - 1$. It is necessary and sufficient to use the following

> Inductive Hypothesis: The contents of the i^{th} window begin and end with # and do not have any other copies of # in-between.

to prove the following

> Inductive Claim: The contents of the $i + 1^{\text{st}}$ window begin and end with # and do not have any other copies of # in-between.

- The contents of the $i + 1^{\text{st}}$ column must begin with `#` — for, otherwise, it follows by the Inductive Hypothesis that $W_{i,0}$ is a window as shown at line (2) such that $\alpha_1 =$ `#` and $\beta_1 \neq$ `#` — making this an ***illegal*** window, since the above condition (i) is not satisfied. Thus $\varphi(\mathcal{F}_\omega) = \varphi(\mathcal{F}_{\text{move}}) = \varphi(\text{legal}_{i,0}) = \mathtt{F}$, ***contradicting*** the requirement that $\varphi(\mathcal{F}_\omega) = \mathtt{T}$.
- The contents of the $i+1^{\text{st}}$ column must also end with `#` — for, otherwise, it follows by the Inductive Hypothesis that $W_{i,T(n)+1}$ is a window as shown at line (2) such that $\alpha_3 =$ `#` and $\beta_3 \neq$ `#` — making this an ***illegal*** window, since condition (ii) is not satisfied. Thus $\varphi(\mathcal{F}_\omega) = \varphi(\text{legal}_{i,T(n)+1}) = \mathtt{F}$, ***contradicting*** the requirement that $\varphi(\mathcal{F}_\omega) = \mathtt{T}$, once again.
- Suppose (to obtain a contradiction) that the contents in row $i + 1$ and column j is also `#` for an integer j such that $1 \leq j \leq T(|\omega|) - 1$. Then window $W_{i,j-1}$ is a window as shown at line (2) such that $\beta_2 =$ `#` — making this an ***illegal*** window, since condition (iii) is not satisfied. Thus $\varphi(\mathcal{F}_\omega) = \varphi(\text{legal}_{i,j-1}) = \mathtt{F}$, once again ***contradicting*** the requirement that $\varphi(\mathcal{F}_\omega) = \mathtt{T}$. Thus the $i + 1^{\text{st}}$ window does not have any copies of `#`, in-between.

Thus the Inductive Claim is satisfied, as needed to complete the Inductive Step and the proof of the claim. ☐

4.3 Achieving the Second Condition

Consider the second condition — that every row includes exactly one cell whose contents is a *state*.

In order to satisfy this, let us require that all windows as shown at line (2) such that either

i. two or more of β_1, β_2 and β_3 belong to Q, or
ii. $\alpha_2 \in Q$ but none of β_1, β_2 or β_3 belong to Q, or
iii. $\beta_2 \in Q$ but none of α_1,α_2 or α_3 belong to Q

are ***illegal***.

Lemma 5. *Let φ be any truth assignment such that*

$$\varphi(\mathcal{F}_\omega) = \mathtt{T},$$

where $\mathcal{F}_\omega$ is as described above, and where the windows described above are **illegal***, as specified there.*

Then φ specifies a well-defined assignment of values in $Q \cup \Gamma \cup \{\#\}$ to the cells in a two-dimensional table with $T(|\omega|) + 1$ rows and $T(|\omega|) + 4$ columns, such that the contents of each row begin and end with $\#$. Exactly one of the cells in between, in this row, stores a copy of a state, and all other cells in-between store elements of Γ.

Thus every row in this table stores a representation of a configuration of M.

Proof. Let φ be any truth assignment such that $\varphi(\mathcal{F}_\omega) = \mathtt{T}$. Lemma 4 remains correct, even though additional windows have now been specified to be ***illegal*** after it was stated and proved — since this would only reduce the set of truth assignments φ such that $\varphi(\mathcal{F}_\omega)$, rather than adding to this set. It follows by this claim that φ specifies a well-defined assignment of values in $Q \cup \Gamma \cup \{\#\}$ to the cells in a two-dimensional table, with $T(|\omega|) + 1$ rows and $T(|\omega|) + 4$ columns, such that the contents of each row begin and end with #, and no copy of # in between. It follows from this that each of the cells, in between, store an element of $Q \cup \Gamma$. It is therefore sufficient to prove that each row has exactly one cell, storing a copy of a state, to establish the claim. That is, the following ***property*** must be proved to hold for each integer i such that $0 \leq i \leq T(|\omega|)$.

> There is exactly one cell in the i^{th} row of the table that stores a state.

This can be proved by induction on i. The standard form of mathematical induction can be used.

Basis: An inspection of the subformula $\mathcal{F}_{\text{start}}$ confirms that, since $\varphi(\mathcal{F}_{\text{start}}) = \mathtt{T}$, the property is satisfied when $i = 0$.

Inductive Step: Let i be an integer such that $0 \leq i \leq T(|\omega|) - 1$. It is necessary and sufficient to use the following

Inductive Hypothesis: Exactly one cell of the i^{th} row of the table stores a state.

to prove the following

Inductive Claim: Exactly one cell of the $i+1^{\text{st}}$ row of the table stores a state.

Suppose, first, that the $i+1^{\text{st}}$ row does not store any copies of states, at all. Let j be the position of the state in the i^{th} row, so that $1 \le j \le T(n)+2$ (since the row begins and ends with #) and the symbol in row i and column j is a state. Then window $W_{i,j-1}$ is ***illegal*** window because it is as shown at line (2), where $\alpha_2 \in Q$ but $\beta_1, \beta_2, \beta_3 \notin Q$. Thus $\varphi(\mathcal{F}_\omega) = \varphi(\text{legal}_{i,j-1}) = \texttt{F}$, ***contradicting*** the fact that $\varphi(\mathcal{F}_\omega) = \texttt{T}$. Thus there is at least one cell in the $i+1^{\text{st}}$ row storing a state.

Suppose, next, that the $i+1^{\text{st}}$ row includes two or more cells that store states. Let j and k be the two smallest integers such that the cells in row $i+1$ and columns j and k each store states, so that (again, because the contents of this row begin and end with #), $1 \le j < j+1 \le k \le T(n)+2$.

- If $k = j+1$ or $k = j+2$ then $W_{i,j}$ is ***illegal*** because it is as shown at line (2), two or more of β_1, β_2 and β_3 belong to Q, so that $\varphi(\mathcal{F}_\omega) = \varphi(\text{legal}_{i,j}) = \texttt{F}$, ***contradicting*** the fact that $\varphi(\mathcal{F}_\omega) = \texttt{T}$.
- If $k \ge j+3$ and the cell in row i storing a state is in column h, where $h \le j+1$, then window $W_{i,k-1}$ is ***illegal*** because it is as shown at line (2) with $\beta_2 \in Q$ and none of α_1, α_2 or α_3 in Q. Thus $\varphi(\mathcal{F}_\omega) = \varphi(\text{legal}_{i,k-1}) = \texttt{F}$, establishing a ***contradiction*** in this case too.
- The only case that remains is that $k \ge j+3$ and the cell in row i storing a state is in column h, where $h \ge j+2$. It now follows that window $W_{i,j-1}$ is illegal because it is as shown at line (2) with $\beta_2 \in Q$ but none of α_1, α_2 or α_3 in Q. Thus $\varphi(\mathcal{F}_\omega) = \varphi(\text{legal}_{i,j-1}) = \texttt{F}$, giving a ***contradiction*** once again.

Since a contradiction has been established in every case, at most one cell in the $i+1^{\text{st}}$ row can store a state. Thus exactly one such cell stores a state, as needed to establish the Inductive Claim, completing the Inductive Step and the proof of the claim. □

4.4 Achieving the Third Condition

Consider the third condition — that the change in state, contents of the tape cell initially visible, and change in tape head location when going from the configuration in one row to the configuration in the next row, is consistent with M's transition function — and all such changes, that are consistent with the transition function, are allowed.

The following conditions will now be added — noting that these reduce the set of windows that might be legal, without identifying any window to be legal if it is already illegal — because none of the windows, shown below, have been classified as “illegal” already.

(a) A window

$\#$	q	α_1
$\#$	s	β_1

is legal when $q, s \in Q$ and $\alpha_1, \beta_1 \in \Gamma$ ***if and only if*** either

i. $(s, \beta_1, \mathtt{L}) \in \delta(q, \alpha_1)$, or

ii. $q \in \{q_{\text{accept}}, q_{\text{reject}}\}$, $s = q$, and $\alpha_1 = \beta_1$.

(b) A window

$\#$	q	α_1
$\#$	β_1	s

is legal when $q, s \in Q$ and $\alpha_1, \beta_1 \in \Gamma$ ***if and only if*** $(s, \beta_1, \mathtt{R}) \in \delta(q, \alpha_1)$.

(c) A window

α_1	q	α_2
s	β_1	β_2

is legal when $q, s \in Q$ and $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \Gamma$ ***if and only if*** $(s, \beta_2, \mathtt{L}) \in \delta(q, \alpha_2)$ and $\alpha_1 = \beta_1$.

(d) A window

α_1	q	α_2
β_1	β_2	s

is legal when $q, s \in Q$ and $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \Gamma$ ***if and only if*** $(s, \beta_2, \mathtt{R}) \in \delta(q, \alpha_2)$ and $\alpha_1 = \beta_1$.

(e) A window

α_1	q	α_2
β_1	s	β_2

is legal when $q, s \in Q$ and $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \Gamma$ ***if and only if*** $s = q \in \{q_{\text{accept}}, q_{\text{reject}}\}$, $\alpha_1 = \beta_1$, and $\alpha_2 = \beta_2$.

(f) All windows

α_1	q	$\#$
β_1	β_2	β_3

where $\alpha_1 \in \Gamma$ and $q \in Q$ are ***illegal***.

4.5 Achieving the Fourth Condition

Consider the fourth condition — that no symbol to the left of the position of the tape head is changed when going from the configuration in one row to the configuration in the row after that.

The following conditions will be added. Once again, a comparison of the contents of the windows shown below, and the windows in the rules above this, will confirm that these also reduce the set of windows that might be legal, without identifying any window to be legal if it is was already classified as illegal (because none of the windows, shown below, have been classified as illegal already).

(a) A window

α_1	α_2	q
β_1	β_2	β_3

such that $q \in Q$, $\alpha_1, \beta_1 \in \Gamma \cup \{\#\}$ and $\alpha_2, \beta_2 \in \Gamma$, is legal ***if and only if*** either

i. $q \notin \{q_{\text{accept}}, q_{\text{reject}}\}$, $\alpha_1 = \beta_1$, $\alpha_2 = \beta_2$, and $\beta_3 \in \Gamma$, or
ii. $\beta_3 = q \in \{q_{\text{accept}}, q_{\text{reject}}\}$, $\alpha_1 = \beta_1$ and $\alpha_2 = \beta_2$

(b) A window

α_1	α_2	q
β_1	s	β_2

such that $q, s \in Q$, $\alpha_1, \beta_1 \in \Gamma \cup \{\#\}$ and $\alpha_2, \beta_2 \in \Gamma$, is legal ***if and only if*** $q \notin \{q_{\text{accept}}, q_{\text{reject}}\}$, $\alpha_1 = \beta_1$ and $\alpha_2 = \beta_2$.

(c) A window

α_1	α_2	α_3
β_1	β_2	β_3

such that $\alpha_1, \beta_1 \in \Gamma \cup \{\#\}$ and $\alpha_2, \alpha_3 \in \Gamma$, is legal ***if and only if*** either

i. $\alpha_2 = \beta_2$, $\alpha_3 = \beta_3$, and either $\alpha_1 = \beta_1 = \{\#\}$ or $\alpha_1, \beta_1 \in \Gamma$, or
ii. $\alpha_2 = \beta_2$, $\beta_3 \in Q$, and either $\alpha_1 = \beta_1 = \{\#\}$ or $\alpha_1, \beta_1 \in \Gamma$.

4.6 Achieving the Fifth Condition

Finally, consider the fifth condition — that no symbol to the right of the position of the tape head is changed when going from the configuration in one row to the configuration in the row after that. Once again, no window shown below has been classified as illegal before this, so these rules simply reduce the set of legal windows.

(a) A window

q	α_1	α_2
β_1	β_2	β_3

such that $q \in Q$ and $\alpha_1, \beta_1, \beta_2 \in \Gamma$, and $\alpha_2, \beta_3 \in \Gamma \cup \{\#\}$ is legal ***if and only if*** $q \notin \{q_{\text{accept}}, q_{\text{reject}}\}$ and $\alpha_2 = \beta_3$.

(b) A window

q	α_1	α_2
β_1	s	β_2

such that $q, s \in Q$ and $\alpha_1, \beta_1 \in \Gamma$, and $\alpha_2, \beta_2 \in \Gamma \cup \{\#\}$ is legal ***if and only if*** $q \notin \{q_{\text{accept}}, q_{\text{reject}}\}$ and $\alpha_2 = \beta_2$.

(c) A window

q	α_1	α_2
s	β_1	β_2

such that $q, s \in Q$ and $\alpha_1, \beta_1 \in \Gamma$, and $\alpha_2, \beta_2 \in \Gamma \cup \{\#\}$ is legal ***if and only if*** $\alpha_2 = \beta_2$.

(d) A window

α_1	α_2	α_3
r	β_1	β_2

such that $\alpha_1, \alpha_2, \beta_1 \in \Gamma$, $r \in Q$, and $\alpha_3, \beta_2 \in \Gamma \cup \{\#\}$ is legal ***if and only if*** $\alpha_2 = \beta_1$ and $\alpha_3 = \beta_2$.

(e) A window

α_1	α_2	$\#$
β_1	β_2	$\#$

such that $\alpha_1, \alpha_2 \in \Gamma$ is legal ***if and only if*** $\alpha_2 = \beta_2$.

4.7 If $\mathcal{F}_\omega$ is Satisfiable Then $\omega \in L$

Lemma 6. *Let φ be any truth assignment such that*

$$\varphi(\mathcal{F}_\omega) = \mathtt{T},$$

where $\mathcal{F}_\omega$ is as described above, and where "legal" and "illegal" windows are as specified above.

Then φ specifies a well-defined assignment of values in $Q \cup \Gamma \cup \{\#\}$ to the cells in a two-dimension table with $T(|\omega| + 1)$ rows and $T(|\omega|) + 4$ columns in such a way that this table is an **accepting tableau** *for M on input ω.*

Proof. Let φ be any truth assignment such that $\varphi(\mathcal{F}_\omega) = \mathtt{T}$, and where the windows that are ***legal***, as well as the windows that are ***illegal*** are as specified above. Lemma 5 remains correct, even though additional windows have been identified as legal, or as illegal, since this result was stated and proved, because none of the rules following the statement of this result defines a window to be "legal" when it had been defined to be illegal, before this. Thus the additional rules only reduce the set of truth assignments φ such that $\varphi(\mathcal{F}_\omega) = \mathtt{T}$.

It now follows, by this result, that, for $0 \leq i \leq T(n)$, the contents of the i^{th} row of the table represent a configuration η_i of M. Furthermore, η_0 is the initial configuration for M and the input string ω, because $\varphi(\mathcal{F}_{\text{start}}) = \mathtt{T}$, and $\eta_{T(n)}$ is an accepting configuration of M, because $\varphi(\mathcal{F}_{\text{accept}}) = \mathtt{T}$ as well. It remains only to prove that the following conditions are satisfied for every integer i such that $0 \leq i \leq T(n) - 1$:

- If η_i is a non-halting configuration of M then $\eta_i \vdash_M \eta_{i+1}$.
- If η_i is a halting configuration of M then $\eta_i = \eta_{i+1}$.

Notice that, since the top row represents the initial configuration of M, the copy of the state in row 0 is found in column 1. A consideration of part (ii) of the rule given in Subsection 4.3 confirms that the copy of the state in row $i+1$ is at most one column farther to the right than the copy of the state in row i, for every integer i such that $0 \leq i \leq T(n) - 1$. This can be used to show, by induction on i, that if $0 \leq i \leq T(n)$, and the copy of the state in row i is in column j, then $j \leq i + 1$. Thus the index of the column storing a copy of the state in row i is $j \leq i + 1 \leq (T(n) + 1) \leq T(n) + 2$, for every row — the copy of the state is never to the right of column $T(n) + 2$ of the table.

Once again, suppose that $0 \leq i \leq T(n)$ and suppose the copy of the state in row i is in column j. Since the contents of each row begin and end with #, one of the following cases must arise:

- *Case:* $j = 1$. If q is the state included in configuration η_i then window $W_{i,0}$ must have the form

$\#$	q	α_1
$\#$	β_1	β_2

  where $\alpha_1 \in \Gamma$ and $\beta_1, \beta_2 \in Q \cup \Gamma$. Furthermore, it follows by part (ii) of the rule in Subsection 4.3 that at least one of β_1 or β_2 is in Q. Indeed, since row $i+1$ represents a configuration, *exactly* one of these belongs to Q. Renaming the contents of cells, one of the following subcases now applies.

  – *Subcase:* $W_{i,0}$ has the form

$\#$	q	α_1
$\#$	s	β_1

for states $q, s \in Q$ and symbols $\alpha_1, \beta_1 \in \Gamma$. Since this window must be legal, it follows by rule (a) in Subsection 4.4 that either

* $(s, \beta_1, \mathtt{L}) \in \delta(q, \alpha_1)$, or
* $q \in \{q_{\text{accept}}, q_{\text{reject}}\}$, $s = q$, and $\alpha_1 = \beta_1$.

Suppose, first, that $(s, \beta_1, \mathtt{L}) \in \delta(q, \alpha_1)$ (so that η_i is a non-halting configuration). In order to show that $\eta_i \vdash_M \eta_{i+1}$ and, in particular, that configuration η_{i+1} can be obtained *using* the fact that $(s, \beta_i, \mathtt{L}) \in \delta(q, \alpha_1)$, it is necessary and sufficient to establish in rows i and $i+1$ have the same symbol in column j, for every integer j such that $3 \leq j \leq T(n) + 2$.

Now, since the table being described has at least five columns, window $W_{i,1}$ must have the form

q	α_1	α_2
s	β_1	β_2

for $q, s \in Q$ as above, $\alpha_1, \beta_1 \in \Gamma$, and for $\alpha_2, \beta_2 \in \Gamma$. Since this window is legal, it follows by rule (c) in Subsection 4.6 that $\alpha_2 = \beta_2$, so that the symbols in column 3 of rows i and $i+1$ are the same.

For $2 \leq h \leq T(n)$ window $W_{i,h}$ must now have the form

σ_1	σ_2	σ_3
τ_1	τ_2	τ_3

where $\sigma_1, \sigma_2, \sigma_3, \tau_1, \tau_2, \tau_3 \in \Gamma$. Since this window must be legal, it now follows by rule (c) in Subsection 4.5 that $\sigma_2 = \tau_2$ and (since $\tau_3 \notin Q$) $\sigma_3 = \tau_3$. Thus rows i and $i+1$ must have the same symbol in column $h+1$ as well as in column $h+2$.

It follows that these rows have the same symbols in column j, for every integer j such that $3 \leq j \leq T(n) + 2$, as needed to establish that $\eta_i \vdash_M \eta_{i+1}$ in this case.

Suppose, next, that $q \in \{q_{\text{accept}}, q_{\text{reject}}\}$, $s = q$, and $\alpha_1 = \beta_1$ (so that η_i is a halting configuration). An application of the same rules, in essentially the same way, establishes that rows i and $i+1$ have the same value in column j, for every integer j such that $3 \leq j \leq T(n) + 2$ in this case as well, so that $\eta_{i+1} = \eta_i$.

– *Subcase:* $W_{i,0}$ has the form

#	q	α_1
#	β_1	s

for states $q, s \in Q$ and symbols $\alpha_1, \beta_1 \in \Gamma$. Since this window must be legal, it follows by rule (b) in Subsection 4.4 that $(s, \beta_1, \mathtt{R}) \in \delta(q, \alpha_1)$. In order to show that $\eta_i \vdash_M \eta_{i+1}$ — applying the move that has been identified — it is necessary and sufficient to show that rows i and $i+1$ have the same values in column j, for every integer j such that $3 \leq j \leq T(n) + 2$ in this case, as well.

Once again, since the table has at least five columns, window $W_{i,1}$ now has the form

q	α_1	α_2
β_1	s	β_2

for $q, s \in Q$ as above, $\alpha_1, \beta_1 \in \Gamma$ as above, and $\alpha_2, \beta_2 \in \Gamma$. Since this window is legal it follows by an application of rule (b) in Subsection 4.6 that $\alpha_2 = \beta_2$: Rows i and $i+1$ have the same values in column 3.

Once again, for $3 \leq j \leq T(n)$, window $W_{i,j}$ must have the form

σ_1	σ_2	σ_3
τ_1	τ_2	τ_3

where $\sigma_1, \sigma_2, \sigma_3, \tau_1, \tau_2, \tau_3 \in \Gamma$. Once again, since this window is legal, rule (c) in Subsection 4.5 establishes that $\sigma_2 = \tau_2$ and $\sigma_3 = \tau_3$. This can be used to argue that the symbols in column j of rows i and $i+1$ are the same for every integer j such that $4 \leq j \leq T(n)+2$, as needed to establish that $\eta_i \vdash_M \eta_{i+1}$.

- *Case:* $2 \leq j \leq T(n)+1$. If q is the state included in configuration η_i then window $W_{i,j-1}$ must have the form

α_1	q	α_2
β_1	β_2	β_3

where $\alpha_1 \in \Gamma$ and $\beta_1, \beta_2, \beta_3 \in Q \cup \Gamma$. Furthermore, it follows by part (ii) of the rule in Subsection 4.3 that at least one of β_1, β_2 or β_3 is in Q. Indeed, since row $i+1$ represents a configuration, *exactly* one of these belongs to Q. Renaming the contents of cells, on of the following subcases now applies.

  – *Subcase:* $W_{i,j-1}$ has the form

α_1	q	α_2
s	β_1	β_2

for states $q, s \in Q$ and symbols $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \Gamma$. Since this window must be legal, it follows by rule (c) of Subsection 4.4 that $(s, \beta_2, \mathtt{L}) \in \delta(q, \alpha_2)$ and $\alpha_1 = \beta_1$. In order to show that $\eta_i \vdash_M \eta_{i+1}$, where this move is used here, it is necessary and sufficient to show that rows i and $i+1$ have the same entries in column h, for each integer h such that $1 \leq h \leq j-2$ and such that $j+2 \leq h \leq T(n)+2$.

If $j \geq 4$ then, for $0 \leq h \leq j-4$, window $W_{i,h}$ has the form

σ_1	σ_2	σ_3
τ_1	τ_2	τ_3

where $\sigma_1, \tau_1 \in \Gamma \cup \{\#\}$, and $\sigma_2, \sigma_3, \tau_2, \tau_3 \in \Gamma$. Since this window is legal, rule (c) in Subsection 4.5 establishes that $\sigma_2 = \tau_2$ and $\sigma_3 = \tau_3$. It follows that (since $j \geq 4$, so that at least one window is considered, here) rows i and $i+1$ have the same values in column h for every integer h such that $1 \leq h \leq j-2$.

If $j = 3$ then window $W_{i,j-3} = W_{i,0}$ has the form

σ_1	σ_2	σ_3
τ_1	τ_2	s

where $\sigma_1 = \tau_1 = \#$, $\sigma_2, \sigma_3, \tau_2 \in \Gamma$, and $s \in Q$. Since this window is legal, rule (c) of Subsection 4.5 implies that $\sigma_2 = \tau_2$. Thus rows i and $i+1$ have the same values in column 1 — that is, in every column h such that $1 \leq h \leq j-2$ in this case as well.

If $j = 2$ then rows i and $i+1$ have the same rows in column h such that $1 \leq h \leq j-2$ because the claim is vacuous (that is, there is no such integer h).

The window $W_{i,j}$ has the form

q	σ_1	σ_2
τ_1	τ_2	τ_3

for $q \in Q$ as above, and $\sigma_1, \tau_1, \tau_2, \in \Gamma$, and $\sigma_2, \tau_2 \in \Gamma \cup \{\#\}$. Since this window is legal, it follows by rule (a) of Subsection 4.6 that $\sigma_2 = \tau_3$, so that rows i and $i+1$ have the same values in column $j+2$.

If $j \leq T(n) - 1$ then, for $j+1 \leq h \leq T(n)$, window $W_{i,h}$ has the form

σ_1	σ_2	σ_3
τ_1	τ_2	τ_3

for $\sigma_1, \sigma_2, \sigma_3, \tau_1, \tau_2, \tau_3 \in \Gamma$. Once again, since this window is legal, it follows by rule (c) of Subsection 4.5 that $\sigma_2 = \tau_2$ and $\sigma_3 = \tau_3$. Thus (since at least one window is considered here) it follows that rows i and $i+1$ have the same values in column h for every integer h such that $j+2 \leq h \leq T(n)+2$ in this case.

If $j = T(n)$, instead, then window $W_{i,j+1} = W_{i,T(n)+1}$ has the form

σ_1	σ_2	#
τ_1	τ_2	#

for $\sigma_1, \sigma_2, \tau_1, \tau_2 \in \Gamma$. Since this window is legal it follows by rule (e) in Subsection 4.6 that $\sigma_2 = \tau_2$ — so that rows i and $i+1$ have the same values in column $T(n)+2$ — that is, in every column h such that $j+2 \leq h \leq T(n)+2$ in this case too.

Finally, if $j = T(n)+1$ then rows i and $i+1$ have the same values in column h, for every integer h such that $j+2 \le h \le T(n)+2$, because the claim is vacuous.

It now follows that $\eta_i \vdash_M \eta_{i+1}$ in this case.

– *Subcase:* $W_{i,j-1}$ has the form

α_1	q	α_2
β_1	s	β_2

for states $q, s \in Q$ and symbols $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \Gamma$. Since this window must be legal, it follows by rule (e) of Subsection 4.4 that $s = q \in \{q_{\text{accept}}, q_{\text{reject}}\}$, $\alpha_1 = \beta_1$ and $\alpha_2 = \beta_2$. In order to show that $\eta_i = \eta_{i+1}$, it is necessary and sufficient to show that rows i and $i+1$ have the same entries in column h, for every integer h such that $1 \le h \le j-2$ and such that $j+2 \le h \le T(n)+2$.

If $j \ge 3$ then, for $0 \le h \le j-3$, window $W_{i,h}$ has the form

σ_1	σ_2	σ_3
τ_1	τ_2	τ_3

where $\sigma_1, \tau_1 \in \Gamma \cup \{\#\}$, and $\sigma_2, \sigma_3, \tau_2, \tau_3 \in \Gamma$. Since this window is legal, rule (c) in Subsection 4.5 establishes that $\sigma_2 = \tau_2$ and $\sigma_3 = \tau_3$. Thus (since $j \ge 3$, so that at least one window is being considered) rows i and $i+1$ have the same values in column h for every integer h such that $1 \le h \le j-2$. If $j = 2$ then this claim is also satisfied because it is vacuous.

The same argument (including considerations or rule (c) in Subsection 4.4 and rule (d) in Subsection 4.6) can be used to show that rows i and $i+1$ have the same values in column h, for every integer h such that $j+2 \le h \le T(n)+2$ in this case as well — as needed to establish that $\eta_i = \eta_{i+1}$.

– *Subcase:* $W_{i,j-1}$ has the form

α_1	q	α_2
β_1	β_2	s

for states $q, s \in Q$ and symbols $\alpha_1, \alpha_2, \beta_1, \beta_2 \in \Gamma$. Since this window must be legal, it follows by rule (d) of Subsection 4.4 that $(s, \beta_2, \mathtt{R}) \in \delta(q, \alpha_2)$ and $\alpha_1 = \beta_1$. In order to show that $\eta_i \vdash_M \eta_{i+1}$, where this move is used here. it is necessary and sufficient to show that rows i and $i+1$ have the same entries in column h, for every integer h such that $1 \le h \le j-2$ and such that $j+2 \le h \le T(n)+2$.

As above, if $j \ge 3$ then, for $0 \le h \le j-3$, window $W_{i,h}$ has the form

σ_1	σ_2	σ_3
τ_1	τ_2	τ_3

where $\sigma_1, \tau_1 \in \Gamma \cup \{\#\}$, and $\sigma_2, \sigma_3, \tau_2, \tau_3 \in \Gamma$. Since this window is legal, rule (c) in Subsection 4.5 establishes that $\sigma_2 = \tau_2$ and $\sigma_3 = \tau_3$. Once again, (since $j \geq 3$, so that at least one window is being considered) rows i and $i + 1$ have the same values in column h for every integer h such that $1 \leq h \leq j - 2$. If $j = 2$ then this claim is also satisfied because it is vacuous.

Now, if $j = T(n) + 1$ then rows i and $i + 1$ have the same values in column h, for every integer h such that $j + 2 \leq h \leq T(n) + 2$ because this claim is vacuous.

If $2 \leq j \leq T(n)$, instead, then window $W_{i,j}$ has the form

q	σ_1	σ_2
τ_1	s	τ_2

where $q, s \in Q$, $\sigma_1, \tau_1 \in \Gamma$ and $\sigma_2, \tau_2 \in \Gamma \cup \{\#\}$. Since this window is legal, it follows by rule (b) of Subsection 4.6 that $\sigma_2 = \tau_2$, so that rows i and $i + 1$ have the same values in column $j + 2$. Now, if $j = T(n)$ then it follows that rows i and $i + 1$ for every integer h such that $j + 2 \leq h \leq T(n) + 2$, since every such integer h has been considered. It therefore remains only to consider the case that $2 \leq j \leq T(n) - 1$ — in which case, we must show that rows i and $i + 1$ have the same values in column h, for every integer h such that $j + 3 \leq h \leq T(n) + 2$.

If $j = T(n) - 1$ then window $W_{i,j+2} = W_{i,T(n)+1}$ has the form

σ_1	σ_2	#
τ_1	τ_2	#

for $\sigma_1, \sigma_2, \tau_1, \tau_2 \in \Gamma$. Since this rule is legal it follows by rule (e) of Subsection 4.6 that $\sigma_2 = \tau_2$, so that rows i and $i + 1$ have the same values in column $j + 3 = T(n) + 2$ — completing the proof in this case too.

It now remains only to consider the case that $2 \leq j \leq T(n) - 2$. In this case, if $j + 2 \leq h \leq T(n)$ then window $W_{i,h}$ has form

σ_1	σ_2	σ_3
τ_1	τ_2	τ_3

where $\sigma_1, \sigma_2, \sigma_3, \tau_1, \tau_2, \tau_3 \in \Gamma$. Since this window is legal it follows by rule (c) in Subsection 4.5 that $\sigma_2 = \tau_2$ and $\sigma_3 = \tau_3$. Thus rows i and $i + 1$ have the same values in column h for every integer h such that $j+3 \leq h \leq T(n)+2$ — completing the proof that $\eta_i \vdash_M \eta_{i+1}$ in this final case.

Since all cases have been considered, the result now follows. ☐

Now, if $\mathcal{F}_\omega$ is satisfiable then there exists a truth assignment φ such that $\varphi(\mathcal{F}_\omega) = \mathtt{T}$, so that the conditions in Lemma 6 are satisfied. Since there can only be an accepting for M on input ω if M accepts ω, we now have the following.

Corollary 7. *If $\mathcal{F}_\omega$ is satisfiable then $\omega \in L$.*

4.8 If $\omega \in L$ Then $\mathcal{F}_\omega$ is Satisfiable

Lemma 8. *If $\omega \in L$ then $\mathcal{F}_\omega$ is satisfiable.*

Proof. Suppose that $\omega \in L$. Then (if $n = |\omega|$) there exists a sequence of configurations

$$\eta_0, \eta_1, \eta_2, \dots, \eta_{T(n)}$$

such that the following properties are satisfied:

(a) η_0 is the initial configuration of M for ω.

(b) If $0 \leq i \leq T(n) - 1$ and η_i is not a halting configuration then $\eta_i \vdash_M \eta_{i+1}$. If η_i is a halting configuration, instead, then $\eta_i = \eta_{i+1}$.

(c) $\eta_{T(n)}$ is an accepting configuration.

Recall that — using a nonstandard set of names of Boolean variables that was introduced to make the proof more readable — $\mathcal{F}_\omega$ can be regarded as a function of Boolean variables $x_{i,j,\sigma}$ where $0 \leq i \leq T(n)$, $0 \leq j \leq T(n) + 3$, and $\sigma \in Q \cup \Gamma \cup \{\mathtt{\#}\}$. The above sequence of configurations now "induces" a truth assignment φ as follows. For $0 \leq i \leq T(n)$ recall that η_i can be specified as a string

$$\eta_{i,0}\eta_{i,1}\eta_{i,2} \cdots \eta_{i,T(n)+3}$$

where $\eta_{i,j} \in Q \cup \Gamma \cup \{\mathtt{\#}\}$, as described earlier in this document. For $0 \leq i \leq T(n)$, $0 \leq j \leq T(n) + 3$ and $\sigma \in Q \cup \Gamma \cup \{\mathtt{\#}\}$ one can now set

$$\varphi(x_{i,j,\sigma}) = \begin{cases} \mathtt{T} & \text{if } \eta_{i,j} = \sigma, \\ \mathtt{F} & \text{otherwise.} \end{cases} \tag{4}$$

Recall that

$$\mathcal{F}_\omega = (\mathcal{F}_{\text{cell}} \wedge \mathcal{F}_{\text{start}} \wedge \mathcal{F}_{\text{accept}} \wedge \mathcal{F}_{\text{move}})$$

for subformulas $\mathcal{F}_{|textcell}$, $\mathcal{F}_{\text{start}}$, $\mathcal{F}_{\text{accept}}$ and $\mathcal{F}_{\text{move}}$ as described above.

- As described above, the subformula $\mathcal{F}_{\text{cell}}$ modifies the requirement that every cell in the two-dimensional table, being specified, stores exactly one (well-defined) value in $Q \cup \Gamma \cup \{\mathtt{\#}\}$. It therefore follows by the definition of φ at line (4) that $\varphi(\mathcal{F}_{\text{cell}}) = \mathtt{T}$.

- $\mathcal{F}_{\text{start}}$ models the requirement that the top row of the table represents the initial configuration of M on the input string ω. Since η_0 is this initial configuration, the definition of φ at line (4) now implies that $\varphi(\mathcal{F}_{\text{start}}) = \mathtt{T}$ as well.

- $\mathcal{F}_{\text{accept}}$ models the requirement that the *bottom* row of the table represents an accepting configuration — so, since $\eta_{T(n)}$ is an accepting configuration, the definition of φ at line (4) implies that $\varphi(\mathcal{F}_{\text{accept}}) = \mathtt{T}$ too.

- It remains only to confirm that $\varphi(\mathcal{F}_{\text{move}}) = \mathtt{T}$. Recall that

$$\mathcal{F}_{\text{move}} = \bigwedge_{0 \le i \le T(n)-1} \left(\bigwedge_{0 \le j \le T(n)+1} \text{legal}_{i,j} \right)$$

  where legal$_{i,j}$ models the requirement that the window specifying values in rows i and $i+1$, and in columns j, $j+1$, $j+2$ is a legal window, as described in Subsections 4.2–4.6. Recall, as well, that legal$_{i,j}$ is the disjunction (or "or") of a sequence of subformulas listing the legal windows for this position. It is, therefore, necessary and sufficient to confirm that the window

$\eta_{i,j}$	$\eta_{i,j+1}$	$\eta_{i,j+2}$
$\eta_{i+1,j}$	$\eta_{i+1,j+1}$	$\eta_{i+1,j+2}$

  for these positions, specified by the above truth assignment φ, satisfies one of the subformulas included in legal$_{,i,j}$ — that is, this is a "legal window".

  A consideration of the rules in Subsections 4.2 and 4.3 confirms that any conditions required by these are satisfied, simply because φ is a truth assignment defined using a sequence of configurations of M. It therefore remains only to consider the rules in Subsections 4.4, 4.5 and 4.6 to check whether the windows defined using φ are legal.

  A reasonably straightforward — but rather long — case analysis can be used to confirm this. To begin, one can consider an arbitrary row i such that $0 \le i \le T(n) - 1$. As argued above, since the tape head initially rests at the leftmost cell and can move right by at most one position, every time a move is made. Thus configuration η_i is one such that the tape head rests at a cell with distance j from the leftmost cell, where $0 \le j \le i$ — and it follows that $\eta_{i,j} \in Q$ for some integer j such that $1 \le j \le i+1 \le T(n)+1$. The following cases can now be considered.

  i. $j = 1$,
  ii. $j = 2$,
  iii. $3 \le j \le T(n) - 1$,
  iv. $j = T(n)$, or
  v. $j = T(n) + 1$.

Let us consider the *third* of the above cases.

– Consider window $W_{i,j-1}$: This has the form

σ_1	q	σ_2
$\eta_{i+1,j}$	$\eta_{i+1,j+1}$	$\eta_{i+1,j+2}$

for $\sigma_1 = \eta_{i,j-1} \in \Gamma$, $q = \eta_{i,j} \in Q$, and $\sigma_2 = \eta_{i,j+1} \in \Gamma$. Now, if η_i is a non-halting configuration then η_{i+1} is obtained either by using a move including moving the tape head *left*, or by using a move including moving the tape head *right*. Otherwise η_i is a halting configuration and $\eta_i = \eta_{i+1}$. These three subcases should now be considered.

  * *Subcase:* η_{i+1} is obtained from the non-halting configuration η_i using some move including moving the tape head *left*. In particular, $q \notin \{q_{\text{accept}}, q_{\text{reject}}\}$, the fact that
$$(s, \tau_2, \mathtt{L}) \in \delta(q, \sigma_2)$$
is being applied, for a state $s \in Q$ and $\tau_2 \in \Gamma$, and window $W_{i,j-1}$ is, therefore

σ_1	q	σ_2
r	σ_1	τ_2

(5)

The only applicable rule in Subsections 4.4–4.6 is rule (c) in Subsection 4.4, and a consideration of this rule confirms that this window is legal.

  * *Subcase:* η_{i+1} is obtained from the non-halting configuration η_i using some move including moving the tape head *right*. In particular, $q \notin \{q_{\text{accept}}, q_{\text{reject}}\}$ the fact that
$$(s, \tau_2, \mathtt{R}) \in \delta(\sigma_2)$$
is being applied, for a state $s \in Q$ and $\tau_2 \in \Gamma$, and window $W_{i,j-1}$ is, therefore

σ_1	q	σ_2
σ_1	τ_2	r

(6)

The only applicable rule in Subsections 4.4–4.6 is rule (d) in Subsection 4.4, and a consideration of this rule establishes that the window is legal in this case too.

  * *Subcase:* η_i is a halting configuration and $\eta_i = \eta_{i+1}$. Thus $q \in \{q_{\text{accept}}, q_{\text{reject}}\}$ and window $W_{i,j-1}$ is

σ_1	q	σ_2
σ_1	q	σ_2

(7)

In this case the only applicable rule in Subsections 4.4–4.6 is rule (e) in Subsection 4.4, and a consideration of this rule establishes that the window is legal in this case as well.

– Consider window $W_{i,j-2}$ — noting that, since $3 \le j \le T(n) - 1$, $1 \le j - 2 \le T(n) - 3$ and (since either $\eta_i = \eta_{i+1}$ or η_{i+1} is obtained from η_i by a move of M, when the tape head is to the right of this position) the same symbol from Γ must appear in both the top and bottom row in the leftmost column of this window.

  Thus if $W_{i,j-1}$ is as shown at line (5), above, then window $W_{i,j-2}$ is

σ_0	σ_1	q
σ_0	r	σ_1

  where $\sigma_0, \sigma_1 \in \Gamma$, $q \in Q \setminus \{q_{\text{accept}}, q_{\text{reject}}\}$ and $r \in Q$. The only applicable rule in Subsections 4.4–4.6 is rule (b) in Subsection 4.5, and a consideration of this rule confirms that this window is legal.

  If $W_{i,j-1}$ is as shown at line (6), instead, then window $W_{i,j-2}$ is

σ_0	σ_1	q
σ_0	σ_1	τ_2

  for $\sigma_0, \sigma_1, \tau_2 \in \Gamma$ and $q \in Q \setminus \{q_{\text{accept}}, q_{\text{reject}}\}$. The only applicable rule in Subsections 4.4–4.6 is rule (a) in Subsection 4.5, and a consideration of this rule confirms that this window is legal too.

  Finally, if $W_{i,j-1}$ is as shown at line (7) then $W_{i,j-2}$ is

σ_0	σ_1	q
σ_0	σ_1	q

  for $\sigma_0, \sigma_1 \in \Gamma$ and $q \in \{q_{\text{accept}}, q_{\text{reject}}\}$. As above, the only applicable rule in Subsections 4.4–4.6 is rule (a) in Subsection 4.5, and this establishes that this window is also legal.

– A consideration of the above now establishes that for $0 \le h \le j - 3$, window $W_{i,h}$ has the form

σ_1	σ_2	σ_3
σ_1	σ_2	τ

  where $\sigma_1 \in \Gamma \cup \{\#\}$, $\sigma_2, \sigma_3 \in \Gamma$, and either $\tau = \sigma_3$ or $\tau \in Q$. The only applicable rule in Subsections 4.4–4.6 is rule (c) in Subsection 4.5, which establishes that this window is legal as well.

- Next consider window $W_{i,j}$. If window $W_{i,j-1}$ is as shown at line (5), above, then, since $3 \leq j \leq T(n) - 1$, this window has form

q	σ_1	σ_2
τ_1	τ_2	σ_2

  where $q \in Q \setminus \{q_{\text{accept}}, q_{\text{reject}}\}$ and $\sigma_1, \sigma_2, \tau_1, \tau_2 \in \Gamma$. The only applicable rule in Subsections 4.4–4.6 is rule (a) of Subsection 4.6, and it follows by this rule that this window is legal.

  If window $W_{i,j-1}$ is as shown at line (6), instead, then $W_{i,j}$ has the form

q	σ_1	σ_2
τ_1	r	σ_2

  where $q \in Q \setminus \{q_{\text{accept}}, q_{\text{reject}}\}$ and $\sigma_1, \sigma_2, \tau_1 \in \Gamma$. The only applicable rule in Subsections 4.4–4.6 is rule (b) in Subsection 4.6, and it follows by this rule that this window is legal too.

  If window $W_{i,j-1}$ is as shown at line (7), then window $W_{i,j}$ has the form

q	σ_1	σ_2
q	σ_1	σ_2

  for $q \in \{q_{\text{accept}}, q_{\text{reject}}\}$ and $\sigma_1, \sigma_2 \in \Gamma$. The only applicable rule in Subsections 4.4–4.6 is rule (c) in Subsection 4.6, and it follows by this rule that this window is legal as well.

- Since $3 \leq j \leq T(n) - 1$, it follows by the above the window $W_{i,j+1}$ has either the form

σ_1	σ_2	σ_3
τ_1	σ_2	σ_3

  for $\sigma_1, \sigma_2, \sigma_3, \tau_1 \in \Gamma$ (if window $W_{i,j-1}$ is either as shown at line (5) or at line (7)) or has the form

σ_1	σ_2	σ_3
r	σ_2	σ_3

  for $\sigma_1, \sigma_2, \sigma_3 \in \Gamma$ and $r \in Q$ (if window $W_{i,j-1}$ is as shown at line (6)). In the first of the cases the only applicable rule in Subsections 4.4–4.6 is rule (c) in Subsection 4.5; in the second of these cases, the only applicable rule in these subsections is rule (d) in Subsection 4.6. In both cases the applicable rule can be used to confirm that the window is legal.

– If h is an integer such that $j+2 \leq h \leq T(n)$ then window $W_{i,h}$ has the form

σ_1	σ_2	σ_3
σ_1	σ_2	σ_3

where $\sigma_1, \sigma_2, \sigma_3 \in \Gamma$. The only applicable rule in Subsections 4.4–4.6 is rule (c) in Subsection 4.5, which confirms that this rule is legal.

– Finally, since $j \leq T(n)-1$, window $T(n)+1$ has the form

σ_1	σ_2	#
σ_1	σ_2	#

where $\sigma_1, \sigma_2 \in \Gamma$. The only applicable rule in Subsections 4.4–4.6 in this case is rule (e) in Subsection 4.6 — which establishes that this window is legal too.

Since all windows $W_{i,h}$ have been considered such that $0 \leq i \leq T(n)-1$ and $0 \leq h \leq T(n)+1$ have been considered, all such windows are legal when $3 \leq j \leq T(n)-1$, as required for this case.

The cases that $j=1$, $j=2$, $j=T(n)$ and $j=T(n)+1$ can all be handled in a similar way — one can start by considering the window $W_{i,j-1}$, along with the moves that might be made, and then move "left" as well as "right", considering the cases which might arise. As above, only one rule in Subsections 4.4–4.6 will be applicable, and it suffices to identify this rule and confirm that the window is legal in order to handle each subcase that arises. $\square$

Completing the Definition of the Function *f*

As described in the lecture, and above, $\mathcal{F}_\omega$ is a Boolean formula defined as a function of Boolean variables

$$\mathcal{W} = \{x_{i,j,\sigma} \mid 0 \leq i \leq T(n), 0 \leq j \leq T(n)+3, \text{ and } \sigma \in Q \cup \Gamma \cup \{\#\}\}.$$

If we set $\ell = |Q| + |\Gamma| + 1$ and order the elements of this set, so that

$$Q \cup \Gamma \cup \{\#\} = \{\sigma_0, \sigma_1, \sigma_2, \dots, \sigma_{\ell-1}\},$$

then this allows a simple way to define a function $\psi : \mathcal{W} \to \mathcal{V}$, for $\mathcal{V} = \{x_0, x_1, x_2, \dots\}$, by setting

$$\psi(x_{i,j,\sigma_h}) = x_{h+j\cdot\ell+i\cdot\ell\cdot(T(n)+4)} \tag{8}$$

for $0 \le i \le T(n)$, $0 \le j \le T(n)+3$, and $0 \le h \le \ell - 1$. Indeed, this maps each variable in $\mathcal{W}$ to one of the variables in the set

$$\{x_i \mid 0 \le i \le (T(n)+1) \cdot (T(n)+4) \cdot \ell - 1\} \subseteq \mathcal{V}$$

in such a way that distinct variables in $\mathcal{W}$ are mapped to distinct variables in the above set. Now, for each Boolean variable $\mathcal{G}$ defined over the Boolean variables in $\mathcal{W}$, let $\widehat{G}$ be the corresponding Boolean variable, defined over the Boolean variables in $\mathcal{V}$, obtained by replacing each occurrence of a variable $x \in \mathcal{W}$ by an occurrence of the corresponding variable $\psi(x) \in \mathcal{V}$.

It is easily argued that $\mathcal{G}$ is satisfiable if and only if $\widehat{\mathcal{G}}$ is, for every Boolean formula $\mathcal{G}$ that is defined over the variables in $\mathcal{W}$. In particular, $\mathcal{F}_\omega$ is satisfiable if and only if $\widehat{\mathcal{F}}_\omega$ is satisfiable. Now, if $f : \Sigma^\star \to \Sigma_F^\star$ such that $f(\omega)$ is the encoding of $\widehat{\mathcal{F}}_\omega$ (as defined in Lecture #11), for all $\omega \in \Sigma^\star$, then it follows by Corollary 7 and Lemma 8, above, that $\omega \in L$ if and only if $f(\omega) \in L_{\mathsf{FSAT}}$, for all $\omega \in \Sigma^\star$.

f is Computable in Polynomial Time

It remains only to prove the following, in order to complete a proof that $L \preceq_{\mathsf{P,\,M}} L_{\mathsf{FSAT}}$.

Lemma 9. *The function $f : \Sigma^\star \to \Sigma_F^\star$, defined above, is computable deterministically in polynomial time.*

Sketch of Proof. Let $\omega \in \Sigma^\star$ and let $n = |\omega|$. As noted above,

$$\mathcal{F}_\omega = (F_{\mathsf{cell}} \wedge \mathcal{F}_{\mathsf{start}} \wedge \mathcal{F}_{\mathsf{accept}} \wedge \mathcal{F}_{\mathsf{move}}),$$

so that

$$\widehat{\mathcal{F}}_\omega = (\widehat{\mathcal{F}}_{\mathsf{cell}} \wedge \widehat{\mathcal{F}}_{\mathsf{start}} \wedge \widehat{\mathcal{F}}_{\mathsf{accept}} \wedge \widehat{\mathcal{F}}_{\mathsf{move}})$$

and the encoding $f(\omega) = e(\widehat{\mathcal{F}}_\omega)$ of $\widehat{\mathcal{F}}_\omega$ is simply the encodings of the subformulas $\widehat{\mathcal{F}}_{\mathsf{cell}}$, $\widehat{\mathcal{F}}_{\mathsf{start}}$, $\widehat{\mathcal{F}}_{\mathsf{accept}}$ and $\widehat{\mathcal{F}}_{\mathsf{move}}$, separated by copies of the symbol "$\wedge$" and enclosed by brackets. It therefore suffices to argue that the encodings of each of these subformulas can be computed deterministically from ω, in polynomial time, to argue that the function f is computable deterministically in polynomial time as well.

An examination of the specifications of the subformulas $\mathcal{F}_{\mathsf{cell}}$, $\mathcal{F}_{\mathsf{start}}$ and $\mathcal{F}_{\mathsf{accept}}$, in Subsection 4.1, should confirm (since $T(n)$ is a fixed polynomial function of n and M is a fixed nondeterministic Turing machine, so that $|Q|$ and Γ can be viewed as constants) that $e(\widehat{\mathcal{F}}_{\mathsf{cell}})$, $e(\widehat{\mathcal{F}}_{\mathsf{start}})$ and $e(\widehat{\mathcal{F}}_{\mathsf{accept}})$ can all be computed deterministically from ω in polynomial time. It therefore remains to consider the string $e(\widehat{\mathcal{F}}_{\mathsf{move}})$.

Recall, again, that

$$\mathcal{F}_{\textsf{move}} = \bigwedge_{0 \le i \le T(n)-1} \left(\bigwedge_{0 \le j \le T(n)+1} \textsf{legal}_{i,j} \right),$$

so that

$$\widehat{\mathcal{F}}_{\textsf{move}} = \bigwedge_{0 \le i \le T(n)-1} \left(\bigwedge_{0 \le j \le T(n)+1} \widehat{\textsf{legal}}_{i,j} \right).$$

Furthermore, a consideration of the rules in Subsections 4.2–4.6 that there exists a string legal $\in (\Sigma_F \cup \{A, B, C, D, E\})^\star$, whose length is polynomial in $|\omega|$ and that can be generated from $n = |\omega|$, deterministically in polynomial time, such that the encoding of $\widehat{\textsf{legal}}_{i,j}$ can be obtained from this string simply by replacing the new symbols A, B, C, D and E by the unpadded decimal representations of the integers i, $i+1$, j, $j+1$ and $j+2$ respectively. This can be used to argue that $e(\widehat{\mathcal{F}}_{\textsf{move}})$ can be computed from ω, deterministically and in polynomial time, as needed to establish the claim. $\square$

Claim 3 is now a consequence of the results that have been stated and proved, above.

5 A Bit of History: Who Were These Guys?

Stephen Cook is an American-Canadian computer scientist who was a Professor in the Department of Computer Science at the University of Toronto. Professor Cook's 1971 paper, *The Complexity of Theorem-Proving Procedures* [1], included a proof that a problem concerning satisfiability of Boolean formulas was complete for $\mathcal{NP}$ with respect to polynomial-time oracle-reducibility.

This was the first problem that was not completely "artificial" whose $\mathcal{NP}$-completeness could be established — so this was a major result at the time, and is still considered to be one today. Consequently, polynomial-time oracle-reductions are also called *Cook reductions* in recognition of this. Professor Cook won the ACM Turing Award in 1982. He was named as an Officer of the Order of Canada in 2015.

Leonid Levin is a Soviet-American computer scientist who also made an extremely important contribution here.

In the 1970's, relations between east and west were so strained that information about the mathematical sciences did not get communicated, very often or effectively, between one side and the other. Consequently it was not known in the west, until years later, that Professor Levin had discovered the existence of "reasonably natural" problems that are $\mathcal{NP}$-complete, independently of Professor Cook. The timing is so close that it is not clear who knew what first.: While Levin's journal article concerning this did not appear until 1973 [2], he had lectured about this for several years before.

Professor Levin is also noted for significant contributions concerning "average-case complexity." He was awarded the Knuth Prize in 2012 for these contributions.

References

[1] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings 3rd Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[2] Leonid Levin. Universal search problems. *Problems of Information Transmission*, 9:115–116, 1973. In Russian.