This is a technique to prove that $A_{TM}$ is reducible to some languages.

Thus it can be used to prove that a language is undecidable.

We will show that the method is useful when the undecidable language involves testing for the existence of something (or say checking if a language is empty).

For example, it is used to prove the undecidability of Hilbert’s 10th problem, testing for the existence of integral roots in a polynomial (details not shown).
Reductions via Computation Histories II

- What is the computation history of a TM?
  - It is the sequence of configurations the machine goes through

Formal definition: $M$ a TM and $w$ an input string

An accepting computation history for $M$ and $w$ is

$$C_1, \ldots, C_l,$$

where

- $C_1$ is the start configuration,
- $C_l$ is an accepting configuration, and
- $C_i$ follows from $C_{i-1}$
A rejecting computation history is similar, except that $C_i$ is a rejecting configuration.

Computation histories are finite sequences. If $M$ does not halt on $w$, no accepting or rejecting computation history for $M$ on $w$.

Deterministic TM has at most one history on an input. It may have zero if a loop occurs.

NTM may have many histories.
We will use computation histories to show some related problems of LBA are undecidable.

A linear bounded automaton (LBA) is a special TM: head cannot move beyond the end of input.

If the head tries to move right at the end of input, the head stays.

It is similar to moving left at the beginning of the tape.

Therefore, we have
An LBA is a TM with limited memory: the tape length is $n$, the input length

This “limited memory” description seems to be strange as $n$ is not a constant

Let’s use another (equivalent) way to define LBA as follows\(^1\)

\(^1\) $\Gamma$ includes 2 special symbols: left and right end markers
Linear Bounded Automaton (LBA) III

2. For any input, let’s have markers in the beginning and in the end.

3. Transitions may not print other symbols over the endmarkers.

4. Transitions may neither move to the left of the left endmarker nor to the right of the right endmarker.

- Then we still have a tape of infinite length, but impose restricted operations.
- Despite memory or operation constraints, LBAs are quite powerful.
A_{DFA}, A_{CFG}, E_{DFA}, E_{CFG} are all LBAs (details not discussed)

1Descriptions come from https://en.wikipedia.org/wiki/Linear_bounded_automaton
We prove that an LBA has a finite number of configurations.

For an LBA $M$, if

- $q$: number of states $= |Q|$,  
- $g$: number of symbols $= |\Gamma|$, 

then $M$ has exactly $qng^n$ distinct configurations for a tape of length $n$.

Proof: a configuration involves current state, head position, tape contents.
Immediately we see

\[ q^n \]

possibilities for each

Thus the total number of possibilities is \( qng^n \)
A_{LBA} Decidable I

- Consider

\[ A_{LBA} = \{ \langle M, w \rangle \mid M \text{ is an LBA that accepts } w \} \]

- In applying \( M \) on \( w \), the concern is that \( M \) loops on \( w \)
- Now tape length is fixed to \( |w| = n \) (or say we will never use more than \( n \) positions)
- If a loop occurs, one configuration appears twice
- Thus we check if in \( qng^n \) steps, same configurations occur
- This is a finite procedure so \( A_{LBA} \) is decidable