Configuration of a TM

- Recall: TM = 7-tuple \((Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})\)
- \(\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\}\)
- A configuration of a TM specifies three things
  - Current state
  - Tape contents
  - Head position

Configurations

- A configuration is a string \(uqv\) in \((\Gamma \cup Q)^*\).
- It means
  - The TM is in state \(q\)
  - The tape contains \(uv\) followed by \(\infty\) blanks
  - The head is over the first character of \(v\).
- The configuration is accepting if \(q = q_{\text{acc}}\).
Successor of a configuration

- Suppose $u, v \in \Gamma^*$ and $a, b \in \Gamma$ and $q \in Q$.
- The successor of the configuration $uaqbv$ is
  - $uacrv$, if $\delta(q,b) = (r,c,R)$
  - $uracv$, if $\delta(q,b) = (r,c,L)$.
- Special case: The successor of $qbv$ is
  - $crv$, if $\delta(q,b) = (r,c,R)$
  - $rcv$, if $\delta(q,b) = (r,c,L)$.
- Special case: If $q \in \{q_{acc}, q_{rej}\}$, then $uqv$ has no successor.

Yielding

- If configuration $C_2$ is a successor of $C_1$, we say “$C_1$ yields $C_2$”.
- Note: TM is deterministic, so a configuration either yields a unique configuration or yields nothing.

TM computation formalized

- Consider TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$.
- We say $M$ accepts $x \in \Sigma^*$ if
  - $\exists$ sequence $C_0, C_1, \ldots, C_t$ of configurations of $M$ s.t.
  - $C_0 = q_0^x$
  - $C_{i-1}$ yields $C_i$ (for all $i$, $1 \leq i \leq t$)
  - $C_t$ is an accepting configuration
- When does $M$ reject $x$? Two choices:
  - Require $M$ to enter reject state
  - Leave this definition as is (i.e., can’t accept $\Rightarrow$ reject)

Deciders vs Recognizers

- Two types of TMs for lang $L$ over alphabet $\Sigma$
- Deciders
  - If $x \in L$, then accept.
  - If $x \notin L$, then reject.
  - Never “loop”, i.e., always halt for any $x \in \Sigma^*$.
- Recognizers
  - If $x \in L$, then accept.
  - If $x \notin L$, either reject or “loop”.
- Note: “loop” $\Rightarrow$ failure to halt; not repetition
Deciders vs Recognizers

• Clearly, every decider is a recognizer.
• Call a language
  – Decidable if there is a decider TM for it
  – Turing-recognizable if there is a recognizer TM for it
• Every decidable language is Turing-recognizable
• Converse is false:
  – ∃ undecidable languages that are Turing-recognizable
  – Can’t prove this today, but eventually…

Multitape Turing Machines

• Like a TM except that it has $k$ tapes, for some fixed $k$. Therefore, it has $k$ heads, one per tape.
• In one step, the TM
  – reads $k$ tape symbols which determine its next state,
  – writes back $k$ symbols, one on each tape,
  – moves heads left/right independent of each other.
• Transition function $\delta : Q \times \Gamma^k \to Q \times \Gamma^k \times \{L,R\}^k$
• E.g., $\delta(q_0, a, b, a) = (q_{14}, c, b, f, R, L, L)$.  $k = 3$

Computation of a multitape TM

• Start with input followed by $\infty$ blanks on tape 1 and only blanks on tapes 2, 3, …, $k$.
• Start with all heads being at left ends of their respective tapes.
• Run TM; accept/reject as usual.
• Think how you might accept the language of palindromes using a 2-tape TM.

Palindromes using 2-tape TM

• “On input $w$,
  – Scan input on tape 1; put head at right end.
  – Scan tape 1 right-to-left; copy input onto tape 2.  (At this point, tape 2 holds $w^R$.)
  – Move head 2 to left end of tape 2.
  – Scan tapes 1 and 2 left-to-right, check for equality.
  – Accept if $w = w^R$, reject otherwise.”
• This is an implementation description, rather than a formal description, of the TM.
Multitape = Single-tape

- Proof uses very important idea of simulation.
- Let \( M \) be a \( k \)-tape TM, for some fixed \( k \).
- We shall build a (single-tape) TM \( M' \) that will simulate \( M \), i.e.,
  - accept if and only if \( M \) accepts,
  - reject if and only if \( M \) rejects.

Proof of multitape = single-tape

- \( M' \) formats its tape to represent all \( k \) tapes of \( M \).
- E.g., with \( k = 3 \), \( \Gamma = \{a,b,c,\} \):
  - Tape 1: \( c a c b a b \) \( \ldots \) Head on third char
  - Tape 2: \( a a a b \) \( \ldots \) Head on first char
  - Tape 3: \( c b a b \) \( \ldots \) Head on fourth char
  - becomes, for \( M' \),
  - Tape: \( # c a C c b a b \ # a a a b \ # c b a B \ # \)
- Thus, each char in \( \Gamma \) has a “marked” version.

Proof of multitape = single-tape

- Let \( M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej}) \) be a \( k \)-tape TM.
- Then, we can simulate it with the following TM \( M' = (Q', \Sigma, \Gamma', \delta', q'_0, q'_{acc}, q'_{rej}) \)...
  - \( Q' = \)
  - \( \Gamma' = \Gamma \times \{\text{Unmarked}, \text{Marked}\} \cup \{\#\} \)
  - \( \delta' = \)
  - \( \ldots \ldots \)
  - Ah, forget it! Go for implementation description.