Turing Machines (3)

Kuliah Teori Bahasa dan Automata
Program Studi Ilmu Komputer
Fasilkom UI

Prepared by:
Suryana Setiawan

Turing Machine Sebagai Recognizer

- Jika sebelumnya Turing Machine dibahas sebagai string processor, sekarang kita akan membahas sebagai recognizer untuk menjawab "apakah $w \in L$?"
- Dalam bab selanjutnya, akan ditunjukkan
  - ada sejumlah L dimana selalu ada mesin M yang bisa menjawab untuk kasus memang $w \in L$ atau $w \notin L$.
  - Ada sejumlah L dimana hanya ada mesin M yang bisa menjawab untuk kasus memang $w \in L$ saja, tapi tidak dapat menjawab untuk $w \notin L$.
  - Ada sejumlah L dimana tidak ada satupun mesin M yang bisa menjawab apakah $w \in L$ atau $w \notin L$.

Menerima/Menolak String

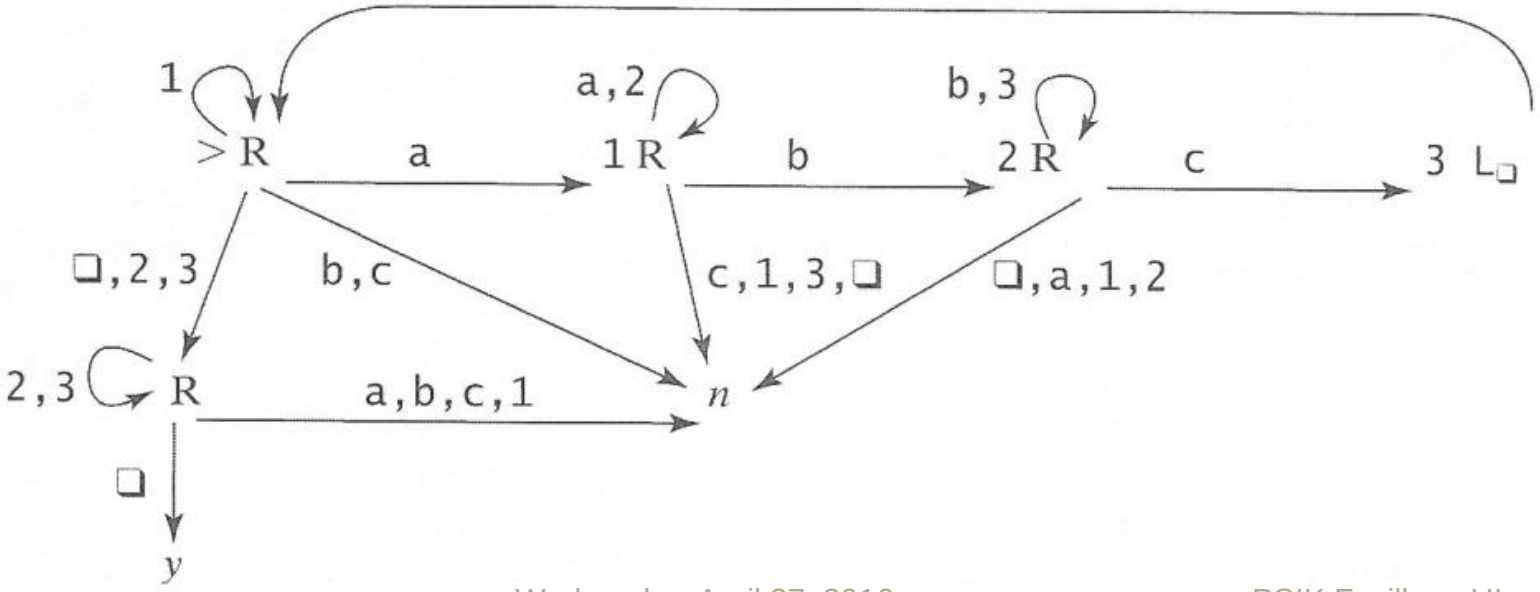
- Jika mesin Turing *M* memiliki start state *s* dan dua halting state $\{y, n\}$, untuk string $w \in \Sigma^*$, kita katakan:
  - *M* **menerima** *w* **iff** $(s, \sqcup w) \vdash_M^* (y, w')$ untuk suatu string *w'*. (y, w') disebut konfigurasi **menerima**.
  - *M* **menolak** *w* **iff** $(s, \sqcup w) \vdash_M^* (n, w')$ untuk suatu string *w'*. (n, w') disebut konfigurasi **menolak**.
- Isi tape tidak diperhatikan lagi saat mencapai halt.
- Jika **tidak halt** maka mesin *M* **tidak** menerima maupun menolaknya.

Memutuskan Bahasa

- Jika Σ alfabet dari mesin M, maka M **memutuskan** bahasa $L \subseteq \Sigma^*$ **iff** untuk setiap string $w \in \Sigma^*$, benar bahwa:
  - Jika $w \in L$, maka M menerima w, dan
  - Jika $w \notin L$, maka M menolak w.
- Setiap bahasa L disebut **decidable** jika terdapat Mesin Turing M yang dapat memutuskan L.

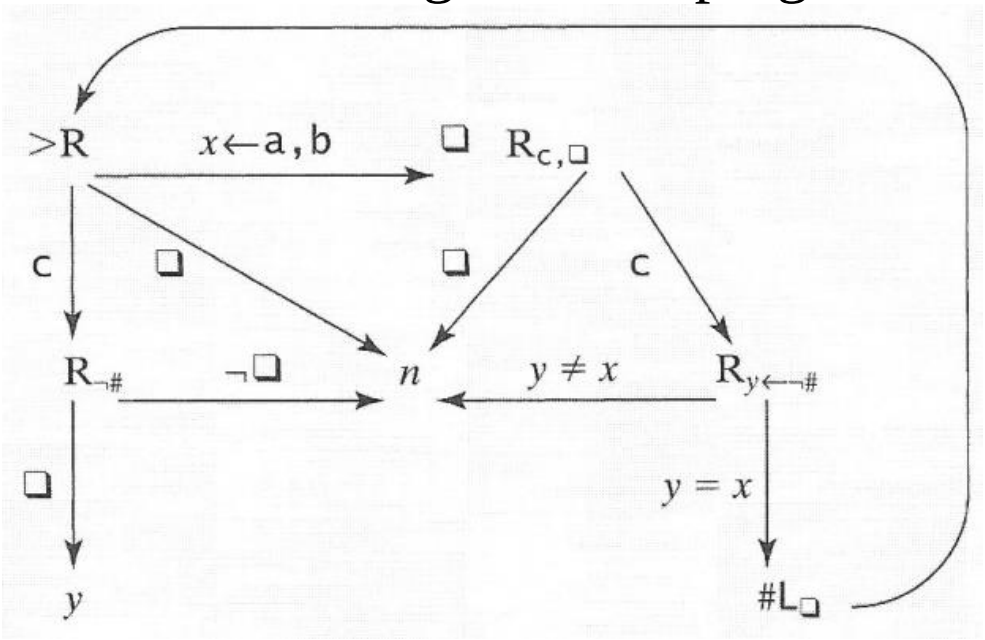
Contoh Bahasa $A^nB^nC^n$

- $A^nB^nC^n$ adalah kelas D (karena TM berikut ini selalu halt di *y* atau *n*) tetapi juga non-CFL (karena tidak ada PDA yang dapat menerimanya).
  - Jelaskan "algoritma" mesin TM ini!
  - Adakah string yang dapat membuatnya looping?



Contoh Bahasa WcW

- WcW juga bahasa decidable tetapi juga non-CFL.
  - Apa manfaat variabel x dan y?
  - Apakah ada kemungkinan looping?



Semi-deciding Bahasa

- Jika Σ alfabet dari mesin M, untuk suatu bahasa $L \subseteq \Sigma^*$ kita akan mengatakan M **semi-deciding** L
  iff untuk setiap string $w \in \Sigma^*$, benar bahwa:
  - Jika $w \in L$, maka M **menerima** w, dan
  - Jika $w \notin L$, maka M tidak menerima w (dalam hal ini tidak menerima bisa artinya **menolak** atau **mengalami infinite-loop**).
- Bahasa L disebut **semidecidable iff** terdapat Mesin Turing yang **semi-deciding** L.

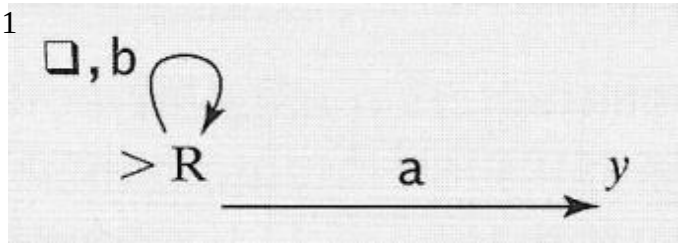
Kelas Bahasa D dan SD

- D adalah himpunan seluruh bahasa **decidable**.
  - Dalam referensi lain disebut bahasa-bahasa **recursive** (R).
- SD adalah himpunan seluruh bahasa **semidecidable**.
  - Dalam referensi lain SD disebut **recursively enumerable** (RE) atau himpunan bahasa **Turing-Recognizable**.

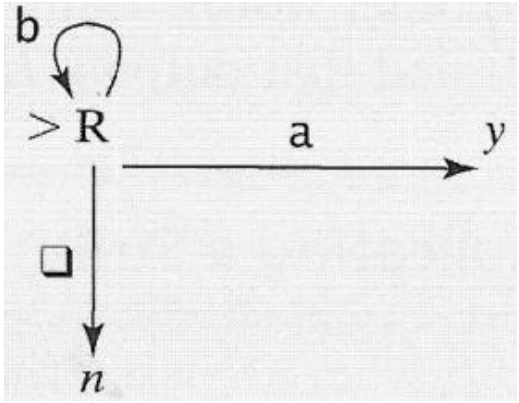
Contoh Mesin Semimemutuskan

- Untuk bahasa L = b*a(a ∪ b)* Mesin yang menerima L akan mencari setidaknya ada satu a.
  - M_1 semi-deciding L.
  - M_2 deciding L.

M_1



M_2



- M_1 akan looping untuk $w \notin L$ (misalnya w=bb).
- Jadi L adalah D tapi juga SD
  - Tetapi, lebih tepat L disebut reguler. Why?

Reminder….

- Reguler $\subset$ CFL $\subset$ D $\subset$ SD
- Sementara bisa terjadi
  $A^nB^nC^n \subseteq A^nB^mC^n \subseteq A^nB^mC^p$ dengan $n,m,p \geq 0$
  - $A^nB^nC^n$ adalah D
  - $A^nB^mC^n$ adalah CFL (berarti juga D) dan
  - $A^nB^mC^p$ adalah reguler (berarti juga CFL dan D).

Mesin Turing Mengkomputasi Fungsi-fungsi

- Di akhir komputasi isi tape menjadi output.
  - Contoh konversi A^nB^m menjadi A^nB^n pada contoh awal.
- Agar TM T dengan start state *s* dan halt state *h* selalu menghasilkan output maka untuk setiap input *w*, mesin M harus selalu halt dengan memenuhi:
  - Definisikan $M(w) = z$ **iff** $(s, \sqcup w) \vdash_M^* (h, \sqcup z)$
- **Konvensi**: head berhenti di posisi di posisi kosong sebelum *z*.
  - Agar *z* bisa menjadi input untuk fungsi lainnya.

Fungsi f: $\Sigma^* \rightarrow \Sigma'^*$

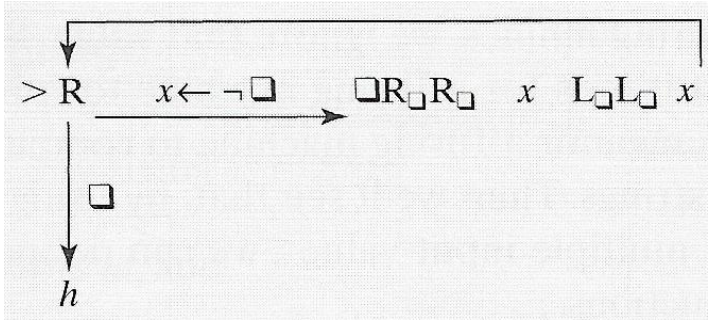
- Mesin Turing M mengkomputasi fungsi f iff, untuk setiap $w \in \Sigma^*$:
  - Jika w input dari domain f, maka $M(w) = f(w)$. Dpl., M halt dengan $f(w)$ sebagai output pada tape.
  - Jika tidak, $M(w)$ tidak halt.

Fungsi Rekursif / Komputabel

- Suatu fungsi f disebut rekursif atau komputabel iff terdapat mesin Turing M yang mengkomputasinya dan selalu halt.
  - Istilah komputasi lebih menggambarkan arti sebenarnya tetapi istilah rekursif merupakan istilah tradisional dalam teori komputasi.

Komputasi Fungsi Duplikasi String

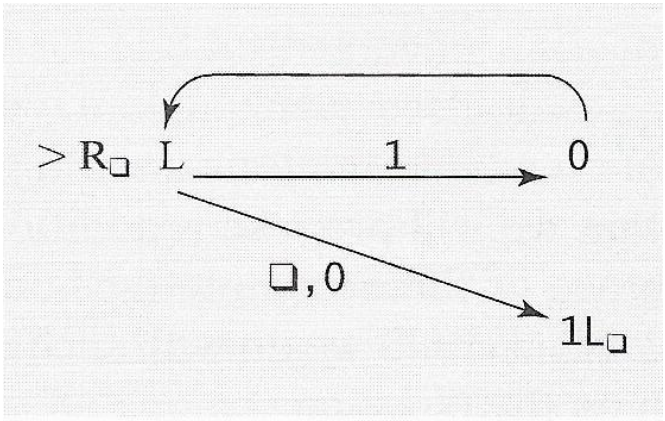
- Fungsi *duplicate*(*w*) = *ww*, dengan *w* string tanpa □.
- Mesin Turing untuk mengkomputasinya dapat dibuat dari dua mesin:
  - Mesin copy *C* dengan operasi:
    $(s_C, \underline{\square} w \square) \vdash_C^* (h_C, \square\, w \underline{\square} w \square)$



  - Mesin shift left $S_{\leftarrow}$ (pernah dibahas) dengan operasi:
    $(s_S, u \underline{\square} w) \vdash_S^* (h_S, uw \underline{\square})$
- Komposisi menghasilkan mesin duplicate(w):
  $> CS_{\leftarrow}L_{\square}$

Komputasi Fungsi Suksesor

- Fungsi succ(w_n) = $w_{(n+1)}$ dengan wn adalah string representasi biner dari n (tanpa prefix 0, kecuali bilangan n=0).
  - Jadi $w_n \in 0 \cup 1\{0,1\}^*$
  - Mesin Turing M harus melakukan komputasi $(s, \sqcup w_n \sqcup) \vdash_M^* (h, \sqcup w_{n+1} \sqcup)$
- Coba pahami apa yang dilakukan mesin M.



Komputasi Fungsi Penambahan

- Untuk x & y non-negatif, fungsi plus(w_x, w_y) = w_{x+y} , dengan w_n adalah string representasi biner dari n (tanpa prefix 0, kecuali n=0).
  - Mesin Turing M harus melakukan komputasi $(s, \underline{\sqcup} w_x;w_y \sqcup) \vdash_M^* (h, \underline{\sqcup} w_{x+y} \sqcup)$, contoh:
  - $(s, \underline{\sqcup}101;1000\sqcup) \vdash_M^* (h, \underline{\sqcup}1101\sqcup)$
- Bagaimana mesinnya? Silakan buat untuk latihan.