Turing Machines (2)

Kuliah Teori Bahasa dan Automata
Program Studi Ilmu Komputer
Fasilkom UI

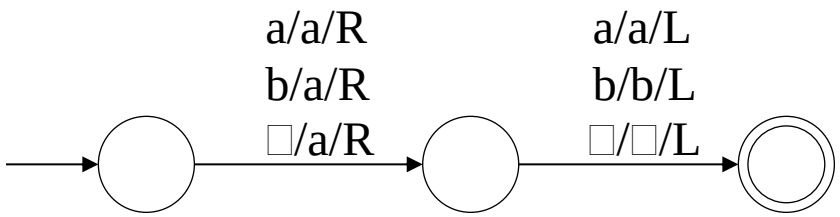
Prepared by:
Suryana Setiawan

Notasi Makro

- Suatu Mesin Turing dapat dipandang sebagai susunan sejumlah Mesin Turing yang lebih sederhana.
- Mesin-mesin Turing sederhana itu bisa muncul berulang di dalam satu Mesin Turing demikian.
- Mesin Turing sederhana dengan fungsionalitas/proses yang jelas dapat digantikan suatu notasi makro guna menyederhanakan gambaran Mesin Turing besarnya.
- Dalam buku teks, satu himpunan notasi digunakan agar suatu Mesin Turing yang kompleks dapat lebih mudah digambarkan.

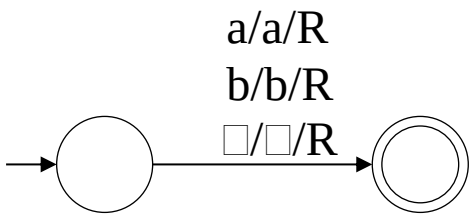
Mesin Penulis Simbol

- Untuk setiap $x \in \Gamma$, $[x]$ didefinisikan sebagai mesin yang menulis x pada posisi head, kemudian, dengan head tetap diposisi yang sama, mesin *halt*.
  - Untuk Γ = {a, b, □}, akan ada mesin $[a]$, $[b]$ dan $[□]$.
  - Misalnya $[a]$ sbb:



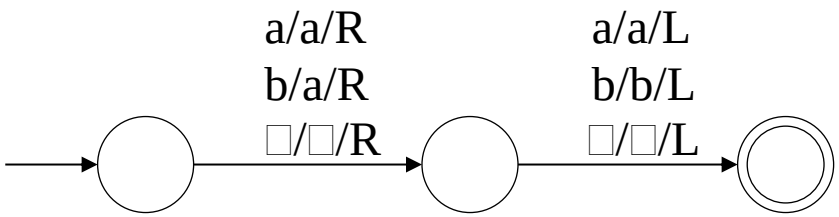
Mesin Pemindah Head

- Dibuat [R] untuk pindah ke kanan, dan [L] untuk pindah ke kiri, keduanya berpindah tanpa mengubah isi tape.
  - Untuk Γ = {a, b, □}, misalnya [R]:



Mesin untuk Halt

- Ada tiga macam halt $[h]$ (halt saja), $[n]$ (halt dengan reject), dan $[y]$ (halt dengan accept).
  - Untuk $\Gamma = \{a, b, \square\}$, misalnya $[h]$:



Mesin Komposisi

- Jika mesin M terbentuk dari komposisi sikuensial antara M_1, M_2 , …, M_n maka dituliskan sebagai konkatenasi sbb.

$$>M_1M_2 \ldots M_n$$

- Tanda mata panah “>” diberikan pada mesin yang pertama akan dijalankan.
- Contoh: >[R][a][R][b][L][L] menyatakan operasi
  - berpindah ke kanan satu posisi,
  - menuliskan a,
  - berpindah ke kanan satu posisi,
  - menuliskan b, dan,
  - berpindah ke kiri dua posisi.
- Note: jika sudah jelas, tanda [dan] dapat dihilangkan.

Mesin Komposisi Kondisional

- Mesin komposisi kondisional dari M_1 dan M_2:
  - menjalankan M_1, kemudian jika M_1 halt, periksa kondisi, dan jika kondisi true, maka jalankan M_2.

$$> M_1 \xrightarrow{\text{kondisi}} M_2$$

- Selanjutnya:

$M_1 \xrightarrow{a} M_2$

Identik dengan

$M_1 \xrightarrow{a/a/R} [L] M_2$

$M_1 \xrightarrow{a, b} M_2$

Identik dengan

$M_1 \xrightarrow[b]{a} M_2$

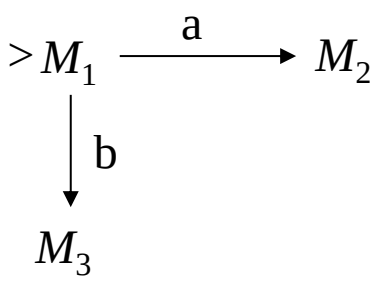
$M_1 \xrightarrow{\neg a} M_2$

Jika Γ = {a, b, □}, identik dengan

$M_1 \xrightarrow{b,□} M_2$

Mesin Pencabangan

- Dengan komposisi kondisional, beberapa cabang dapat dibuat sesuai simbol yang dibaca head
  - Contoh untuk Γ = {a, b, □}:



Mulai dari start state M_1, jalankan M_1.
Kemudian

- Jika simbol pada head a, maka jalankan M_2.
- Jika simbol pada head b, maka jalankan M_3.

Variabel Penyimpan Harga

- Simbol yang dibaca head dapat "direkam" dalam suatu "variabel" untuk digunakan kemudian.
  - Perekaman ditandai dengan tanda assignment "←" setelah variabel, dan diikuti oleh simbolnya.
- Contoh 1:

$$M_1 \xrightarrow{x \leftarrow \neg a} M_2 \quad \cdots \quad M_3 \xrightarrow{x = y} M_4$$

  - berikut ini menggambarkan setelah M_1, periksa kondisi "jika bukan a" adalah true, rekam sebagai x, dan lanjutkan ke M_2, suatu saat setelah M_3, simbol dalam x sama dengan simbol dalam y, maka jalankan M_4.
- Contoh lain:

$$> \xrightarrow{x \leftarrow \neg \square} Rx$$

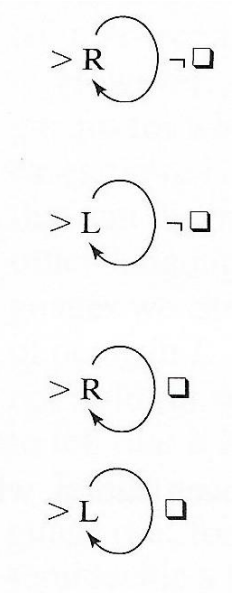
- setiap simbol yang dibaca bukan blank, simpan sebagai x lalu setelah bergeser satu posisi ke kanan tuliskan x disitu.

Realisasi dalam Notasi TM Standard

- Dapat dilakukan dengan
  - penyimpanan di posisi tertentu pada tape, atau
  - Dengan membuat pencabangan ke status-status berbeda untuk setiap kemungkinan.
- Kendala:
  - Penyimpanan di tape menyebabkan head hilir-mudik dari posisi sebenarnya ke posisi penyimpanan lalu kembali ke posisi sebenarnya, tapi
  - Pencabangan ke status-status berbeda dapat melipatgandakan jumlah status dari mesin.

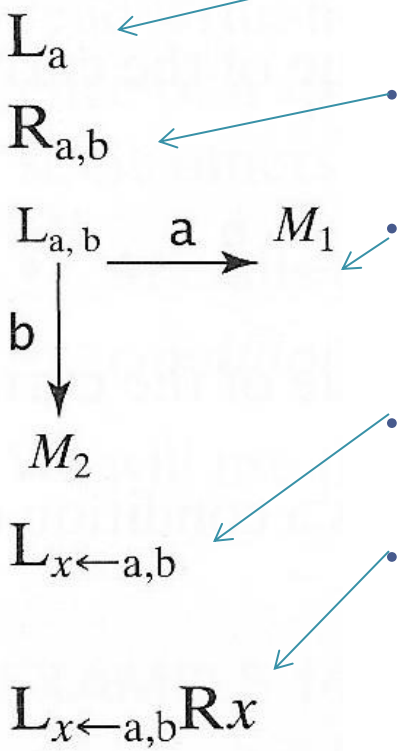
Mesin Iterator (1)

- Melakukan perulangan menjalankan suatu mesin selama kondisi true. Notasi dengan **panah self-cyclic** dengan label kondisinya atau dengan men-**subscript negasi kondisi** setelah nama mesinnya.
- Contoh:
  - Pindahkan head ke **kanan**, lalu selama disitu bukan blank, lakukan kembali.
  - Pindahkan head ke **kiri**, lalu selama disitu bukan blank, lakukan kembali.
  - Pindahkan head ke **kanan**, lalu selama disitu blank, lakukan kembali.
  - Pindahkan head ke **kiri**, lalu selama disitu blank, lakukan kembali.



Mesin Iterator (2)

- Lebih lanjut kita bisa menuliskan iterator-iterator berikut ini.



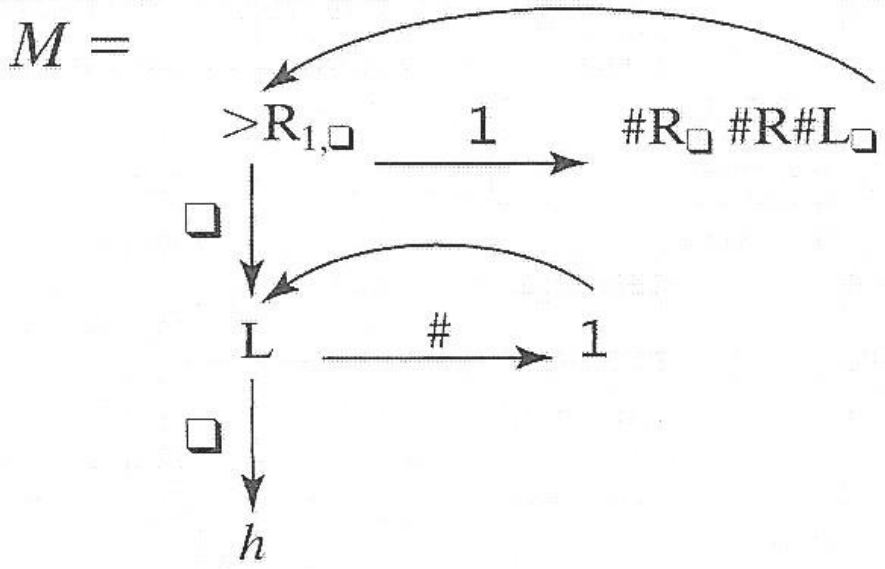
- Temukan simbol a pertama mulai dari posisi sebelah kiri posisi head.
- Temukan simbol a atau b pertama mulai dari posisi di sebelah kanan posisi head.
- Menemukan posisi a atau b pertama mulai dari posisi sebelah kiri posisi head, kemudian jika itu a, jalankan M_1, jika itu b, jalankan M_2.
- Menemukan posisi a atau b pertama mulai dari posisi sebelah kiri posisi head, kemudian catat sebagai *x*.
- Menemukan posisi a atau b pertama mulai dari posisi sebelah kiri posisi head, kemudian catat sebagai *x*. lalu pindahkan head ke sebelah kanannya dan tuliskan *x* disitu.

Contoh Mesin Triplikasi (1)

- Mesin untuk men-triplikasi suatu string w $\in \{1\}^*$
  - Input: □*w* Output: □*www*
  - Contoh input □111 dan output □111111111

- Dalam loop:
  - Temukan simbol 1 atau □ pertama arah ke kanan.
  - Jika simbol □ (semua 1 sudah di-copy), maka keluar dari loop.
    Jika tidak (simbol 1), tandai sebagai # (untuk tidak dicopy ulang).
  - Temukan blank pertama di kanan.
  - Lalu, tuliskan dua buah # (pada dua blank berurutan).
- Lakukan dalam satu pass mengubah # kembali ke 1.

Contoh Mesin Triplikasi (2)



Contoh: Geser Satu Posisi ke Kiri

- Kita akan mebangun mesin penggeser $S_{\leftarrow}$
- Input: $\sqcup u\underline{\sqcup}w\sqcup$ Output: $\sqcup uw\underline{\sqcup}$, untuk $u,w \in \Sigma^*$.
- Contoh input $\sqcup 11\underline{\sqcup}00\sqcup$ dan output $\sqcup 1100\underline{\sqcup}$ dengan $\Sigma = \{0,1\}$.
- Ide: iterasi dari kiri ke kanan dalam w, copy setiap simbol ke sebelah kirinya.