MODUL 16: VARIAN MESIN TURING

Mesin Turing Multitape

Pertanyaan hingga saat ini: apakah model dasar dari TM yang telah kita bahas bisa ditingkatkan kemampuan komputasinya dengan model yang lebih *sophisticated*? Dalam hal ini pengertian "kemampuan komputasi yang sama" adalah dalam memecahkan masalah yang sama serta menemukan jawaban yang sama pula, bukan pada kecepatan dan efisiensi, atau kenyamanannya.

Beberapa kemungkinan varian TM dengan penambahan batasan:

- ♦ TM dengan gerakan head yang selalu ke kanan atau kekiri saja (tidak stasioner).
- ♦ TM dengan pemisahan aktifitas pembacaan dan penulisan isi tape; dalam satu move kedua aktifitas itu tidak dapat dilakukan sekaligus.

Dengan pengurangan batasan:

- ♦ TM dengan gerakan head ke kedua dimensi
- ♦ TM memiliki tape yang tidak berbatas di kiri

Varian-varian di atas bisa dibuktikan bahwa ternyata tidak menambah atau mengurangi kemampuannya (setidaknya tidak berubah secara signifikan).

Berikut ini akan dibahas varian TM yang menggunakan lebih dari satu tape dan akan diperlihatkan juga bahwa ternyata TM multitape ini kemampuannya sama saja dengan TM tape tunggal.

Teorema 9.1. Jika n ≥ 2, dan $T_1 = (Q_1, \Sigma, \Gamma_1, q_1, d_1)$ meru[akan Mesin Turing n-tape, maka akan terdapat suatu mesin Turing 1-tape, T2 = $(Q_2, \Sigma, \Gamma_2, q_2, d_2)$, dengan $T_1 \subseteq T_2$, serta memenuhi dua kondisi sbb.

1. $L(T_1) = L(T_2)$; yaitu untuk setiap $x \in \Sigma^*$, T_2 akan halt untuk x jika dan hanya jika T_1 halt untuk x.
2. Untuk $x \in \Sigma^*$, jika

$$(q_1, \underline{\Delta}x, \underline{\Delta}, \ldots, \underline{\Delta}) \quad {}^*_{T1} (h,\ y\underline{a}z, y_2\underline{a_2}z_2, \ldots, y_n\underline{a_n}z_n)$$

untuk $a, a_i \in \Gamma_1 \cup \{\Delta\}$ dan $y, z, y_i, z_i, \in (\Gamma_1 \cup \{\Delta\})^*$, maka

$$(q_2, \Delta x) \quad {}^*_{T2} (h,\ yaz)$$

Bukti: Kita akan buktikan untuk $n = 2$. Ini dapat dikembangkan untuk membuktikan kasus yang lebih umum. Pembuktian ini dilakukan dengan membentuk suatu TM 1-tape yang mampu mensimulasikan TM 2-tape. Pengertian simulasi: TM yang dibentuk berlaku dan tampak sebagai TM 2-tape. Yang terjadi sebenarnya adalah memperluas himpunan simbol tape Γ_2 dengan simbol-simbol baru sbb ini sehingga membuat tapenya seolah berisikan dua track.. Untuk memudahkan $\Gamma_I \cup \{\Delta\}$ kita singkat menjadi Γ. Γ_2 berisikan

- Simbol-simbol yang ada dalam Γ_1.
- Elemen-elemen $\Gamma\times\Gamma$. Suatu simbol (X, Y) pada kotak -*i*, merepresentasikan simbol X dalam kotak -*i* tape pertama dan simbol Y dalam kotak-*i* tape kedua dari T_1.
- Elemen-elemen $(\Gamma'\times\Gamma) \cup (\Gamma'\times\Gamma) \cup (\Gamma'\times\Gamma')$, dimana Γ' berisikan simbol yang sama seperti Γ dengan penandaan apostrof untuk membedakannya dari simbol dalam Γ.
- Suatu simbol tambahan # yang akan ditempatkan di kotak-0.

Langkah selanjutnya dalam pembentukan T_2, kotak-1 diisi (Δ', Δ') untuk menyatakan bahwa head berada di kotak tsb, dan kembali ke kotak-0.

Simulasi eksekusi $d(p, a_1, a_2) = (q, b_1, b_2, D_1, D_2)$

1. Move head ke kanan hingga pasangan (a'_1, c) ditemukan, dan mengingat a_1 lalu move kambali ke kotak-0 yang berisi #.
2. Lakukan hal yang sama untuk menemukan pasangan (d, a'_2) dan cara yang sama langkah-1 (tanpa kembali ke #).
3. Mengingat status q dan ganti pasangan (d, a'_2) dengan (d, b_2) lalu move head sesuai dengan D_2.
4. Jika kotak yang sekarang (setelah mote D_2) berisi # maka crash karena T_1 akan crash jika mencoba melakukan move head dari tape 2 keluar. Jika tidak, dan jika kotak hanya berisi simbol a maka update menjadi (a, Δ'); jika berisi pasangan simbol (a, b) naja update menjadi (a, b'). Kemudian, move kembali ke kotak-0.
5. Tempatkan pada kotak yang berisi (a'_1, c) kembali (sebagaimana langkah-1), dan mengubahnya menjadi (b_1, c) dan melakukan move dalam arah D_1.
6. Sebagaimana langkah-4 kalau tidak crash, mengubah a menjadi (a', Δ) atau mengubah (a, b) menjadi (a', b). Kemudian kembali ke 0.

Keenam langkah di atas diiterasi sehingga T_2 berjalan mensimulasi T_1. Bila status baru adalah h, T_2 melakukan langkah-langkah terminasi berikut untuk menghasilkan konfigurasi yang benar.

7. Dalam satu pass ubah setiap pasangan (a, b) menjadi simbol a.
8. Hapus # sehingga
9. Pindahkan head ke simbol yang diutamakan, ubah menjadi simbol yang tidak diutamakan, dan berhenti di posisi tersebut.

Contohnya jika setelah sejumlah iterasi telah terbentuk sbb.

$\underline{\#}$	Δ	0’	Δ	0	1	Δ	
	0	1	0	1’			

Iterasi berikutnya untuk $d(p, 0, 1) = (q, \Delta, 0, L, R)$. Setelah langkah ke 3

#	Δ	0’	Δ	$\underline{0}$	1	Δ	
	0	1	0	$\underline{1’}$	Δ’		

Setelah langkah ke 4

$\underline{\#}$	Δ	0’	Δ	0	1	Δ	
	0	1	0	0	Δ’		

Setelah langkah ke 5

#	$\underline{\Delta}$	Δ	Δ	0	1	Δ	
	$\underline{0}$	1	0	0	Δ’		

Setelah langkah ke 6

$\underline{\#}$	Δ’	Δ	Δ	0	1	Δ	
	0	1	0	0	Δ’		

Dengan pembuktian ini maka dapat ditunjukan bahwa suatu TM n-tape dapat disimulasikan oleh suatu TM dalam model dasarnya yang mendukung Thesis Church-Turing. Jadi secara umum dapat dikatakan bahwa suatu bahasa yang diterima oleh TM n-tape akan diterima pula oleh suatu TM dasar, dan juga berlaku kebalikannya.

Mesin Turing Nondeterministik

Ingat bahwa untuk suatu bahasa regular yang dikenali suatu FA Nondeterministik (NFA) akan selalu ada FA Deterministik yang juga dapat mengenali bahasa tersebut, dan sebaliknya. Namun, jika ada suatu CFL yang dikenali oleh suatu PDA Nondeterministis (NPDA) tapi tidak dapat dikenali oleh PDA Deterministik, contohnya Palindrom. Dalam kasus TM, suatu bahasa yang diterima oleh suatu Nondeterministic Turing Machine (NTM) akan selalu ada TM yang dapat menerima bahasa yang sama.

NTM didefinisikan sama dengan TM biasa kecuali fungsi transisi d menghasilkan harga suatu subset dari $(Q \cup \{h\}) \times (\Gamma \cup \{\Delta\}) \times \{R, L, S\}$, sementara pada TM menghasilkan suatu elemen tunggal. Dalam hal ini kita tidak perlu menyatakan suatu fungsi adalah parsial atau bukan karena $d(\ )$ dapat berharga $\emptyset$.

Mesin Turing Universal

Mesin Turing Universal T_u adalah suatu TM yang menerima masukan yang berisikan dua bagian: pertama adalah suatu string yang menspesifikasikan suatu TM lain T_1, dan kedua adalah string z yang akan menjadi masukan dari T_1. Jika diasosiasikan dengan sistem komputer saat ini maka T_u adalah mesin komputer itu sendiri, T_1 adalah program yang dijalankan komputer dengan masukan z.

Dalam hal ini karena akan direpresentasikan sebagai suatu string maka T_1 perlu dikodekan melalui suatu aturan pengkodean tertentu.

T_u *adalah TM 3-tape*

1. Tape pertama berisi masukan yang mana masukan berisi string yang merupakan konkatenasi string yang berisi pengkodean definisi TM T yang akan dieksekusi dan pengkodean string yang berisi masukan untuk T (jadi masukannya $e(T)e(z)$)
2. Tape kedua untuk mensimulasikan tape dari T
3. Tape ketiga mencatat status dari T saat T dieksekusi

Pengkodean TM T ($e(T)$)

- Secara keseluruhan TM T dengan status inisial q_0 dikodekan sebagai $e(T) = s(q_0)1e(m_1)1e(m_2)1\ldots\ e(m_k)1$
- $e(m_i)$ adalah pengkodean fungsi transisi $d(p, a) = (q, b, D)$ yang dinyatakan dengan $e(m_i) = s(p)1s(a)1s(q)1s(b)1s(D)1$
- Jika x adalah status, $s(x)$ adalah pengkodean status. Khusus h harus 0, status inisial q_0 harus 00 dan lainnya: 0^3, 0^4, 0^5, ….dst
- Jika x adalah simbol tape atau Δ, $s(x)$ adalah pengkodean sbb. Khusus Δ harus 0, yang ainnya: 0^2, 0^3, 0^4, 0^5, ….dst
- Arah move dari tape dikode S menjadi 0, L menjadi 00, dan R menjadi 000

Contoh: TM berikut ini akan mentransformasikan string masukan $b^n a^k$ enjadi b^{n+k} untuk n, $k \geq 0$.

- TM memiliki $Q = \{q_0, p, r\}$, dengan pengkodean
  $h = 0$, $q_0 = 00$, $p = 000$, $r = 0000$, dan
- memiliki $\Gamma = \{a, b\}$, dengan pengkodean
  $\Delta = 0$, $a = 00$, $b = 000$.
- Fungsi-fungsi transisinya dikodekan sbb.
  $d(q_0, \Delta) = (p, \Delta, \mathrm{R})$ dikodekan menjadi 001010001010001
  $d(p, b) = (p, b, \mathrm{R})$ dikodekan menjadi
  00010001000100010001
  $d(p, a) = (r, b, \mathrm{L})$ dikodekan menjadi 0001001000010001001
  $d(p, \Delta) = (r, \Delta, \mathrm{L})$ dikodekan menjadi 0001010000101001
  $d(r, b) = (r, b, \mathrm{L})$ dikodekan menjadi
000010001000010001001
  $d(r, \Delta) = (h, \Delta, \mathrm{S})$ dikodekan menjadi 0000101010101

Jadi $e(T)$ =
00100101000101000100010001000100010001000100010010000
10001001000101000010100100001000100001000100010000
101010101

Pengkodean Masukan TM T ($e(z)$)

Jika string masukan adalah $z = z_1 z_2 \ldots z_l$ dikodekan sesuai dengan pengkodean simbol di ata sbb.

$$e(z) = 11s(z_1)1s(z_2)1s(\ldots z_l)1$$

Contoh:
Untuk TM di atas diberikan string masukan baa maka pengkodeannya

110001001001.

Sehingga masukan untuk TM T_u adalah

0010010100010100010001000100010001000100010010000
1000100100010100001010010000100010000100010010000
10101010111000100101

Algoritma T_u

1. Memindahkan e(z) dari tape 1 ke dalam tape 2 mulai dari kotak ke 3 sementara kotak 0 berisi blank, kotak 1 berisi 0 dan kotak 2 berisi 1, dan head dari tape 2 mulai dari kotak 1.
2. Mengisi tape 3 dengan harga penghodean status awal mulai kotak 1, 2 , … (kotak 0 berisi blank)
3. Start melakukan eksekusi dengan mensimulasikan jalannya T
   - berdasarkan status di tape-3 dan current symbol di tape-2, mencari rule ybs di tape-1,
   - kemudian jika ketemu (jika tidak ketemu maka crash!), update status di tape-3, update current symbol di tape-2 dan move head tape-2
4. Iterasi berlanjut hingga status di tape-3 menunjukkan status halt.