MODUL 3:
Finite Automata

DEFINISI FA

mesin yang dapat mengenai bahasa regular tanpa menggunakan storage/memory.

Sejumlah status dapat didefinisikan pada mesin untuk “mengingat” beberapa hal secara terbatas.

Definisi Formal

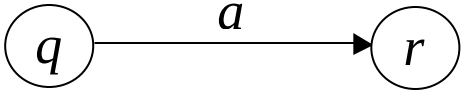
Suatu finite automaton (FA) adalah 5-tuple (Q, Σ,q_0, δ, A), yang mana:

- Q: himp. berhingga dari **status**
- Σ: himp. berhingga **alfabet** simbol masukan
- $q_0 \in Q$: status **inisial** (awal)
- $A \subseteq Q$: himp. status **menerima** (accepting state, kadang-kadang disebut **final state**)
- δ: **fungsi transisi** dimana δ: $Q \times \Sigma \rightarrow Q$

Fungsi Transisi

Untuk $q, r \in Q$ dan $a \in \Sigma$, jika $\delta(q, a) = r$ terdefinisi, maka saat mesin berada dalam status q menerima masukan simbol a, mesin akan bertransisi ke status r.

Dengan diagram status digambarkan dengan bulatan dan transisi dengan panah, sbb:



Untuk pembahasan berikutnya pernyataan

> “M adalah suatu FA dengan himpunan status Q, masukan simbol alfabet Σ, dan status awal q_0, serta himpunan status menerima A, dengan fungsi transisi δ,”

kita singkat menjadi

> “$M = (Q, \Sigma, q_0, \delta, A)$ merupakan FA.”

Contoh: $L \subseteq \{0,1\}^*$ yang mana setiap stringnya selalu memiliki akhiran 10. Ekspresi regular bahasa ini $(0+1)^*10$.

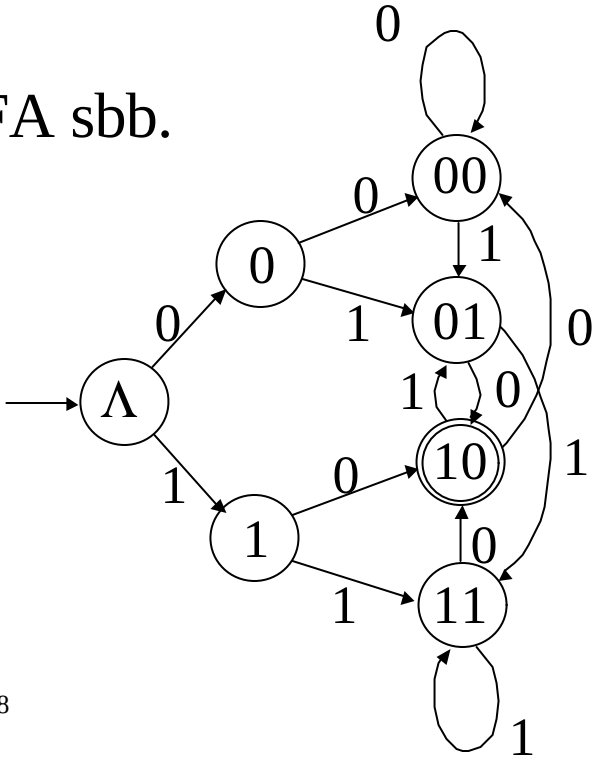
Untuk mengenalinya maka mesin memerlukan

- 4 status untuk menyimpan keempat kemungkinan dua simbol terakhir -- kita sebut status-status tsb. 00, 01, 10, dan 11;
- 2 status ketika baru menerima satu simbol pertama -- kita sebut status-status 0 dan 1; dan

- satu status untuk mesin berada di awal (belum menerima simbol) -- kita sebut status Λ.

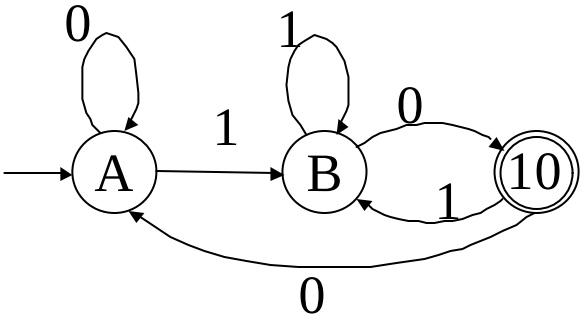
Diagram dan tabel transisi FA sbb.

Status \ masukan	**0**	**1**
Λ	0	1
0	00	01
1	10	11
00	00	01
01	10	11
10	00	01
11	10	11



Minimasi FA mencapai bentuk minimum (jumlah status paling sedikit) dengan tiga status saja, sbb.

Status	masukan	
δ	**0**	**1**
A	A	B
B	10	B
10	A	B



PERLUASAN FUNGSI TRANSISI δ MENJADI δ^*

Definisi: $M = (Q, \Sigma, q_0, \delta, A)$ merupakan FA. Terdefinisi suatu fungsi δ^* yang memetakan $Q \times \Sigma^*$ ke Q dengan sifat sebagai berikut:

- untuk setiap $q \in Q$, berlaku $\delta^*(q,\Lambda) = q$
- untuk setiap $y \in \Sigma^*$, $a \in \Sigma$, dan $q \in Q$, berlaku $\delta^*(q,ya) = \delta(\delta^*(q, y), a)$

Jadi

- $\delta(q, a)$: “status berikutnya dari q dan menerima input simbol a”
- $\delta^*(q, x)$: “status berikutnya dari q dan menerima input string simbol x”

String x adalah string dari simbol-simbol dalam alfabet, x bisa juga Λ. Jika $x = a_1a_2\ldots a_{k-1}a_k$ maka

$$\delta^*(q, x) = \delta(\delta^*(q, a_2a_2\ldots a_{k-1}), a_k) = \ldots = \delta(\delta(\delta(\ldots(\delta(\delta(q, a_1), a_2)\ldots), a_{k-1}), a_k)$$

Contoh: Jika $\delta(q, a) = q_1$, $\delta(q_1, b) = q_2$, dan $\delta(q_2, c) = q_3$, maka

$$\begin{aligned}
\delta^*(q, abc) &= \delta(\delta^*(q, ab), c) \\
&= \delta(\delta(\delta^*(q, a), b), c) \\
&= \delta(\delta(\delta(\delta^*(q, \Lambda), a), b), c) \\
&= \delta(\delta(\delta(q, a), b), c) \\
&= \delta(\delta(q_1, b), c) \\
&= \delta(q_2, c) \\
&= q_3.
\end{aligned}$$

Kesimpulan 1: Menurut definisi $\delta^*(q, a)$ adalah $\delta(\delta^*(q, \Lambda), a)$ namun sebagai implikasi dari definisi dapat kita simpulkan bahwa untuk setiap $a \in \Sigma$, dan $q \in Q$, berlaku $\delta^*(q,a) = \delta(q, a)$. (Jadi: untuk string 1 simbol, kedua fungsi bisa dituliskan bertukaran).

Kesimpulan 2: Jika $x, y \in \Sigma^*$ maka $\delta^*(q,xy) = \delta^*(\delta^*(q, x), y)$

Ini bisa and buktikan dengan induksi matematis (silakan mencoba sebagai latihan!).

Dari contoh di atas dapat diperlihatkan pula bahwa
$\delta^*(q,abc) = \delta^*(\delta^*(q, ab), c) = \delta^*(\delta^*(q, a), bc)$.

FA SEBAGAI RECOGNIZER DARI BAHASA REGULAR

Definisi Penerimaan String

Suatu $M = (Q, \Sigma, q_0, \delta, A)$ merupakan suatu FA.

- Suatu string x dikatakan dikenal (atau diterima) oleh M jika $\delta^*(q, x) \in A$.
- Jika suatu string tidak diterima maka string itu dikatakan tidak diterima atau ditolak oleh M.

Istilah “dikenal” dan ”diterima” adalah sama untuk FA dan bahasa regular dan selanjutnya digunakan bertukaran.

Definisi Penerimaan Bahasa

Jika himp. string yang dikenal oleh *M* adalah

$$L(M) = \{x \in \Sigma^* \mid x \text{ dikenal oleh } M\}$$

maka suatu bahasa *L* (bahasa pada Σ) dikenal oleh *M*, jika dan hanya jika $L = L(M)$.

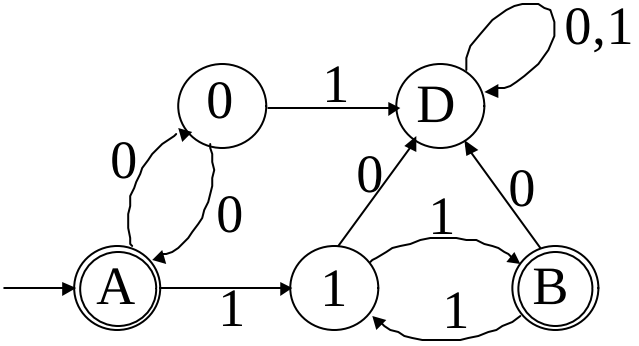
L diterima oleh *M* jika semua string dari *L* dikenal oleh *M*, dan setiap string dari *L* harus diterima oleh *M* jika *L* tersebut dikenal oleh *M* dan menolak setiap string dalam *L'*.

Teorema. Bahasa L pada Σ adalah regular jika dan hanya jika terdapat suatu FA yang mengenal L.

Jadi teorema ini menjamin bahwa

- untuk setiap M, yaitu sebarang FA, terdapat ekspresi regular yang terkait dengan $L(M)$; dan
- untuk regular ekspresi r tersebut, terdapat suatu FA yang mengenal bahasa tsb.

Contoh 1: dari diagram FA berikut ini kita akan mendapatkan ekspresi regular dari bahasa *L* yang dikenal FA tersebut.



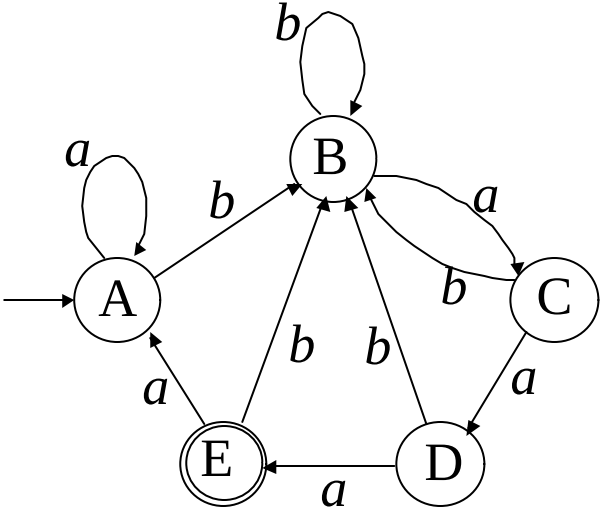
- Status A mrpk status inisial & status menerima, maka $\Lambda \in L$ dan setiap string x dimana $\delta^*(A, x) = A$ adalah deretan pasangan simbol 0. Kedua hal ini bergabung dengan ekspresi $(00)^*$.

- Status D adalah status yang tidak akan pernah bertransisi ke status menerima, jadi setiap x di mana $\delta^*(A, x) = D$, tidak akan pernah menjadi prefiks dari string dalam L.

- Status terima *B* hanya dapat dicapai melalui status 1 yang langsung dicapai dari status *A* dengan jumlah genap simbol 1. Jadi jika tidak revisit A maka *x* memiliki juga ekspresi regular $(11)^+$. Jika terjadi revisit A maka ekspresi regularnya menjadi $(00)^*(11)^+$.

- Menggabungkan kedua status regular di atas maka: $(00)^* + (00)^*(11)^+ = (00)^*(\Lambda + (11)^+) = (00)^*(11)^*$.

Contoh 2: diagram berikut ini dari suatu FA untuk string-string $\in \{a,b\}^*$.



- Status terima hanya satu: E, E hanya dapat dicapai dari D dengan simbol *a*, D hanya dapat dicapai dari C dengan simbol *a*, C hanya dapat dicapai dari B dengan simbol *a*, dan B pertama kali dicapai dari A dengan simbol *b*.
- Dari status mana pun setiap simbol *b* akan selalu bertransisi ke B. Jadi, setiap string yang membawa FA ke status terima E harus memiliki sufiks *baaa*.
- Maka ekspresi regularnya adalah: $(a+b)^* baaa$

Cara untuk mendapatkan ekspresi regular di atas adalah dengan cara yang kira-kira (intuitif). Cara yang lebih sistematis akan dibahas di topik mendatang.

OPERASI-OPERASI HIMPUNAN

L_1 dan L_2 bahasa-bahasa regular pada alfabet Σ serta akan terdapat FA M_1 dan M_2 yang masing-masing mengenal L_1 dan L_2. Sesuai definisinya maka $L_1 \cup L_2$ juga bahasa regular, yang juga akan terdapat FA yang mengenalnya.

Bagaimana dengan irisan, kompelemen dan difference (dan operasi-operasi himpunan lainnya)?

Bahasa Komplemen *L'*

Untuk suatu bahasa regular *L*, maka *L'* (kompelemen dari *L*) adalah bahasa regular karena jika $M = (Q, \Sigma, q_0, \delta, A)$ adalah FA untuk mengenal *L* maka $M' = (Q, \Sigma, q_0, \delta, B)$ dapat dibuat dengan $B = Q\text{-}A$
(status menerima di *A* menjadi tidak menerima di *B* dan sebaliknya semua status tidak menerima di *A* menjadi status menerima di *B*).

Bahasa Hasil Operasi Himpunan Lain
dapat dinyatakan sebagai sejumlah operasi gabungan dan komplemen.

Contoh:

$L_1 \cap L_2 = (L_1' \cup L_2')'$

$L_1 - L_2 = L_1 \cap L_2' = (L_1' \cup L_2)'$

$L_1 \oplus L_2 = (L_1 - L_2) \cup (L_2 - L_1) = (L_1' \cup L_2)' \cup (L_1 \cup L_2')'$

Jadi setiap operasi himpunan pada beberapa bahasa regular akan menghasilkan bahasa regular juga sehingga akan ada mesin FA yang dapat mengenalnya.

Masalahnya sekarang dapatkah kita membentuk mesin FA dari bahasa hasil operasi himpunan tersebut berdasarkan kombinasi mesin-mesin bahasa asalnya secara langsung?

Mesin FA Kombinasi Operasi Himpunan

Berikut ini akan dibahas pembentukan FA untuk bahasa penggabungan kedua bahasa ($L_1 \cup L_2$). Untuk kasus irisan ($L_1 \cap L_2$) dan perbedaan ($L_1 - L_2$) dapat dilakukan dengan sedikit modifikasi.

Berdasarkan FA $M_1 = (Q_1, \Sigma, q_1, A_1, \delta_1)$ untuk mengenali L_1, dan FA $M_2 = (Q_2, \Sigma, q_2, A_2, \delta_2)$ untuk mengenali L_2, suatu FA $M = (Q, \Sigma, q_0, A, \delta)$ akan dibentuk untuk dapat mengenali $(L_1 \cup L_2)$.

Ide: saat memeriksa masukan x, mesin M melakukan sekaligus kedua pemeriksaan apakah $x \in L_1$ serta apakah $x \in L_2$. Jadi dalam M akan terdapat status-status kombinasi antara status-status yang ada dalam M_1 dengan M_2.

- Untuk $p \in Q_1$, $q \in Q_2$ terdefinisi status $(p,q) \in Q$. status inisial $q_0 = (q_1, q_2)$.
- Bila M berada dalam status (p,q), bila ia menerima simbol a maka ia akan bertransisi ke $(\delta_1(p,a), \delta_2(q,a))$.

Himpunan status menerima A adalah himpunan status (p, q) yang mana p atau q (salah satu atau keduanya!) adalah status menerima dari mesin-mesin asalnya.

Perbedaan FA untuk bahasa-bahasa ($L_1 \cap L_2$) dan ($L_1 - L_2$) adalah hanya terletak pada himpunan status menerima A tsb.

- Untuk ($L_1 \cap L_2$), status (p, q) $\in$ A jika keduanya **keduanya** (p dan q) adalah status menerima dari mesin-mesin asalnya.
- Untuk ($L_1 - L_2$), status (p, q) $\in$ A jika ***p* adalah status menerima** dari M_1 dan ***q* bukan status menerima** dari M_2.

Teorema 3.4. Misalkan $M_1 = (Q_1, \Sigma, q_1, A_1, \delta_1)$ dan $M_2 = (Q_2, \Sigma, q_2, A_2, \delta_2)$ masing-masing mengenal L_1 dan L_2. Bila $M = (Q, \Sigma, q_0, A, \delta)$ suatu FA yang mana

$Q = Q_1 \times Q_2$,

$q_0 = (q_1, q_2)$, dan

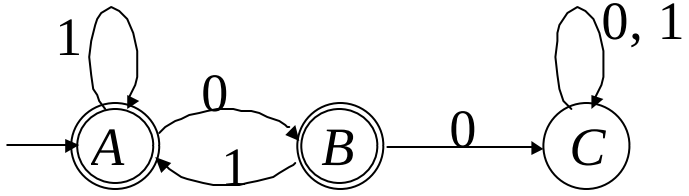
δ terdefinisi sbg $\delta((p, q), a) = (\delta_1(p,a), \delta_2(q,a))$

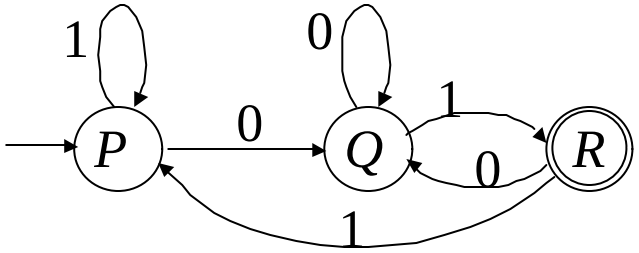
untuk setiap $p \in Q_1$, $q \in Q_2$ dan $a \in \Sigma$ maka,

1. Bila $A = \{(p, q) \mid p \in A_1$ atau $q \in A_2\}$, maka M mengenal $(L_1 \cup L_2)$

2. Bila $A = \{(p, q) \mid p \in A_1 \text{ dan } q \in A_2\}$, maka M mengenal $(L_1 \cap L_2)$
3. Bila $A = \{(p, q) \mid p \in A_1 \text{ dan } q \notin A_2\}$, maka M mengenal $(L_1 - L_2)$

Contoh: misalkan L_1 dan $L_2 \subseteq \{0, 1\}^*$, dengan
$L_1 = \{ x \mid$ tidak ada substring 00 dalam $x \}$
$L_2 = \{ x \mid x$ berakhiran 01$\}$



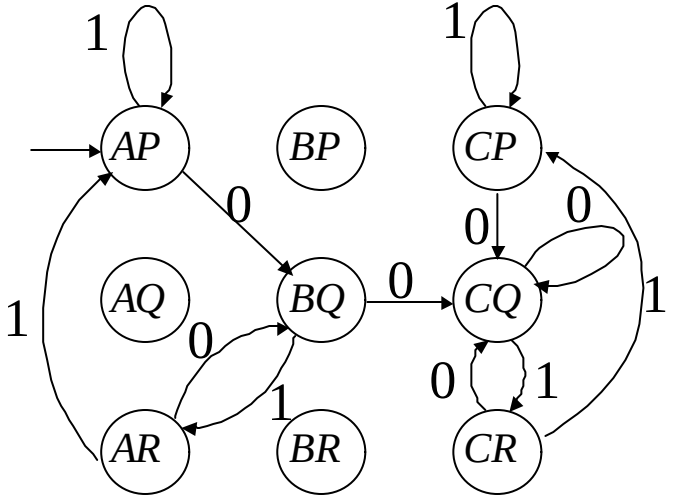


Masing-masing memiliki tiga status sehingga mesin baru yang dibentuk dari kedua mesin tsb. berisikan 9 status sbb. (penyederhanaan penulisan (*A*,*P*) ditulis *AP*, dst.) sehingga

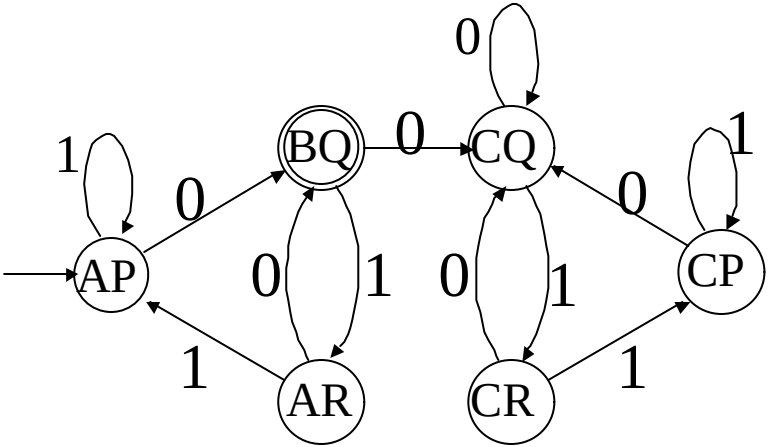
- $Q = \{AP, AQ, AR, BP, BQ, BR, CP, CQ, CR\}$
- Status awal *AP*
- Transisi dari *AP*
  - $\delta(AP, 0) = (\delta_1(A, 0), \delta_2(P, 0)) = BQ$
  - $\delta(AP,1) = (\delta_1(A, 1), \delta_2(P, 1)) = AP$

- Transisi dari BQ
  - $\delta(BQ, 0) = (\delta_1(B, 0), \delta_2(Q, 0)) = CQ$
  - $\delta(BQ, 1) = (\delta_1(B, 1), \delta_2(Q, 1)) = AR$
- Transisi dari CQ
  - $\delta(CQ, 0) = (\delta_1(C, 0), \delta_2(Q, 0)) = CQ$
  - $\delta(CQ, 1) = (\delta_1(C, 1), \delta_2(Q, 1)) = CR$
- Transisi dari AR
  - $\delta(AR, 0) = (\delta_1(A, 0), \delta_2(R, 0)) = BQ$
  - $\delta(AR, 1) = (\delta_1(A, 1), \delta_2(R, 1)) = AP$

… dst maka akan diperoleh diagram transisi berikut ini.



Dalam diagram, hanya enam status yang dapat tercapai dari *AP*. Jadi ketiga status lainnya bisa dihilangkan. Untuk kasus $L = L_1 - L_2$ maka himp. status menerima $A = \{AP, BQ\}$.



Penyederhanaan dapat dilakukan dengan cara yang akan diterangkan dalam pembahasan yad.