OTOMATA 03

KLEENE'S THEOREMS

Materi Pertemuan

- Apa itu Teorema Kleene?
- Metode Pembuktian

Teorema Kleene (1956)

Teorema ini memberikan landasan tentang bagaimana sebuah bahasa reguler secara fleksibel dapat diekspresikan melalui banyak cara.

Bahasa yang dapat didefinisikan melalui salah satu dari :

- **Regular Expression**, atau
- **Finite Automata**, atau
- **Transition Graph**

Pasti dapat didefinisikan pula melalui kedua cara lainnya.

Metode Pembuktian

Pembuktian Teorema Kleene dapat dilakukan melalui 3 tahap :

- Membuktikan bahwa setiap bahasa yang dapat didefinisikan melalui FA, pasti dapat pula didefinisikan melalui TG
- Membuktikan bahwa setiap bahasa yang dapat didefinisikan melalui TG, pasti dapat pula didefinisikan melalui RE
- Membuktikan bahwa setiap bahasa yang dapat didefinisikan melalui RE, pasti dapat pula didefinisikan melalui FA

Pembuktian Tahap I

Membuktikan bahwa setiap bahasa yang dapat didefinisikan melalui FA, pasti dapat pula didefinisikan melalui TG

Karena FA merupakan subset dari TG, maka dapat dikatakan bahwa sebenarnya FA itu sendiri adalah TG

Pembuktian Tahap II (1)

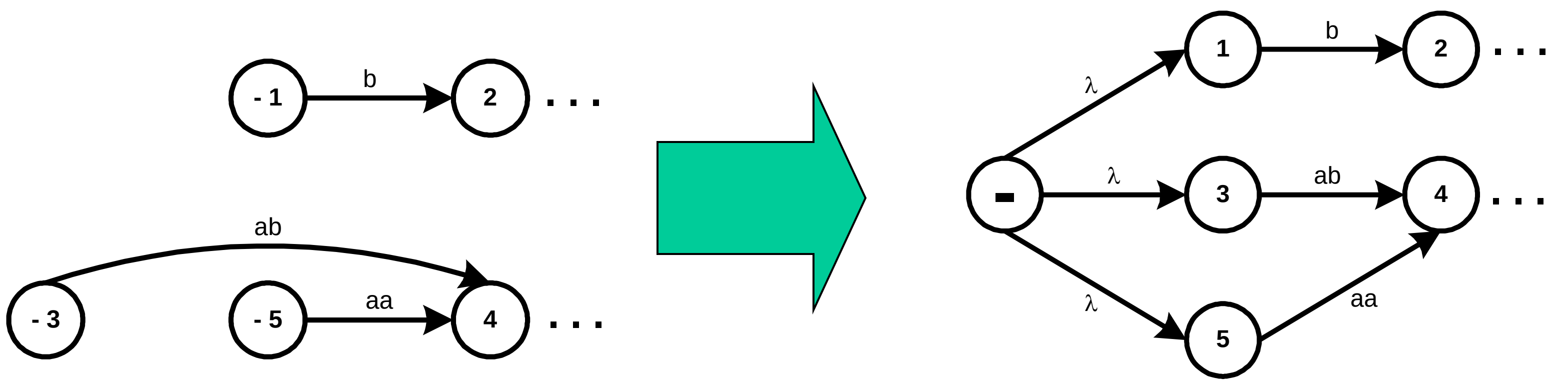
Membuktikan bahwa setiap bahasa yang dapat didefinisikan melalui TG, pasti dapat pula didefinisikan melalui RE

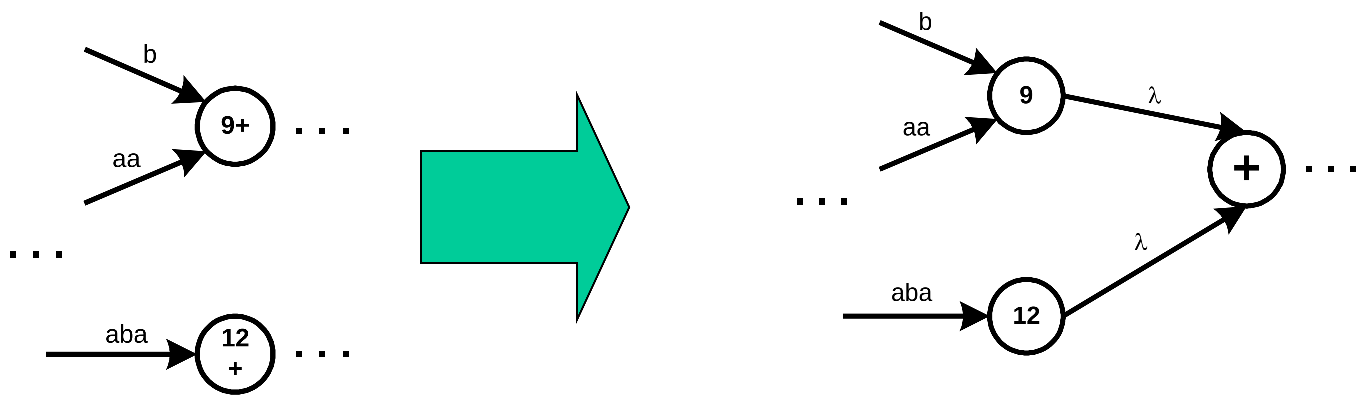
Pendekatan yang digunakan adalah dengan menyusun algoritma untuk menyederhanakan sebuah TG sdmk hingga menjadi sebuah RE yang berasosiasi dengannya

Pembuktian Tahap II (2)

Beberapa operasi yang dapat dilakukan untuk meperoleh RE dari TG adalah :

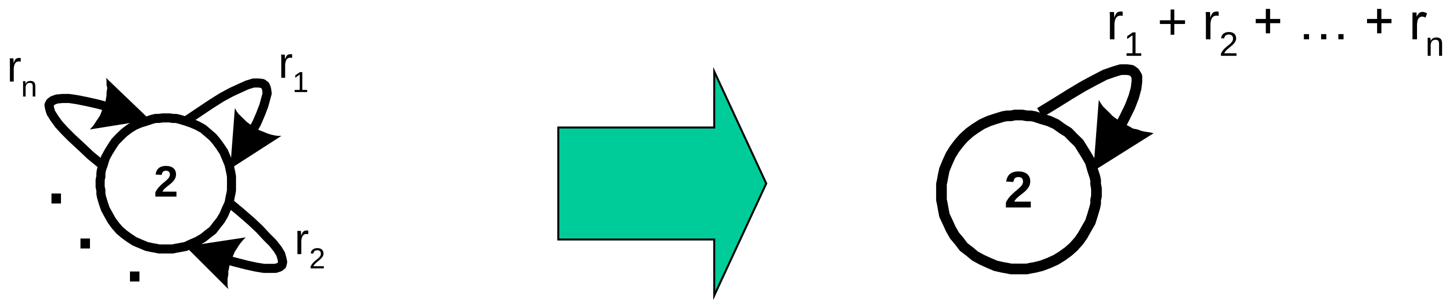
a. Meminimalkan Start dan Final State

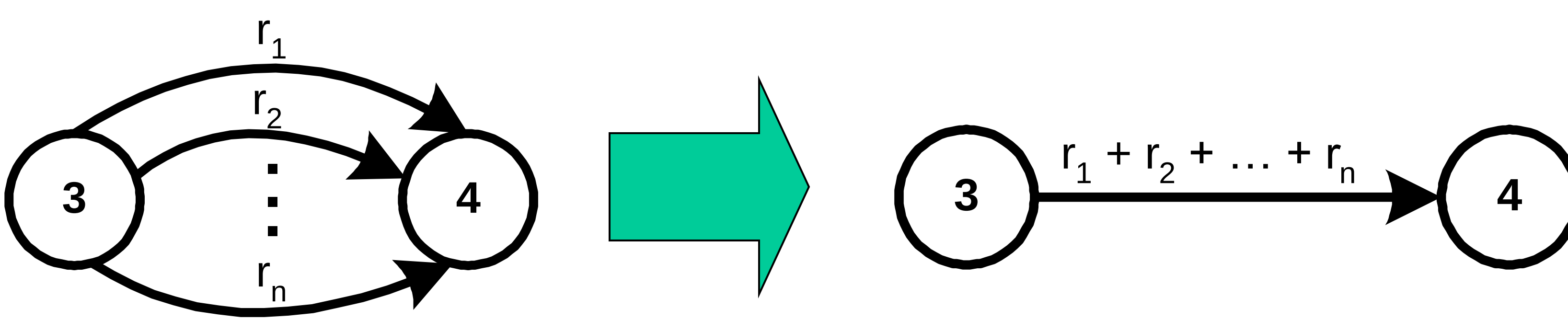




Pembuktian Tahap II (3)

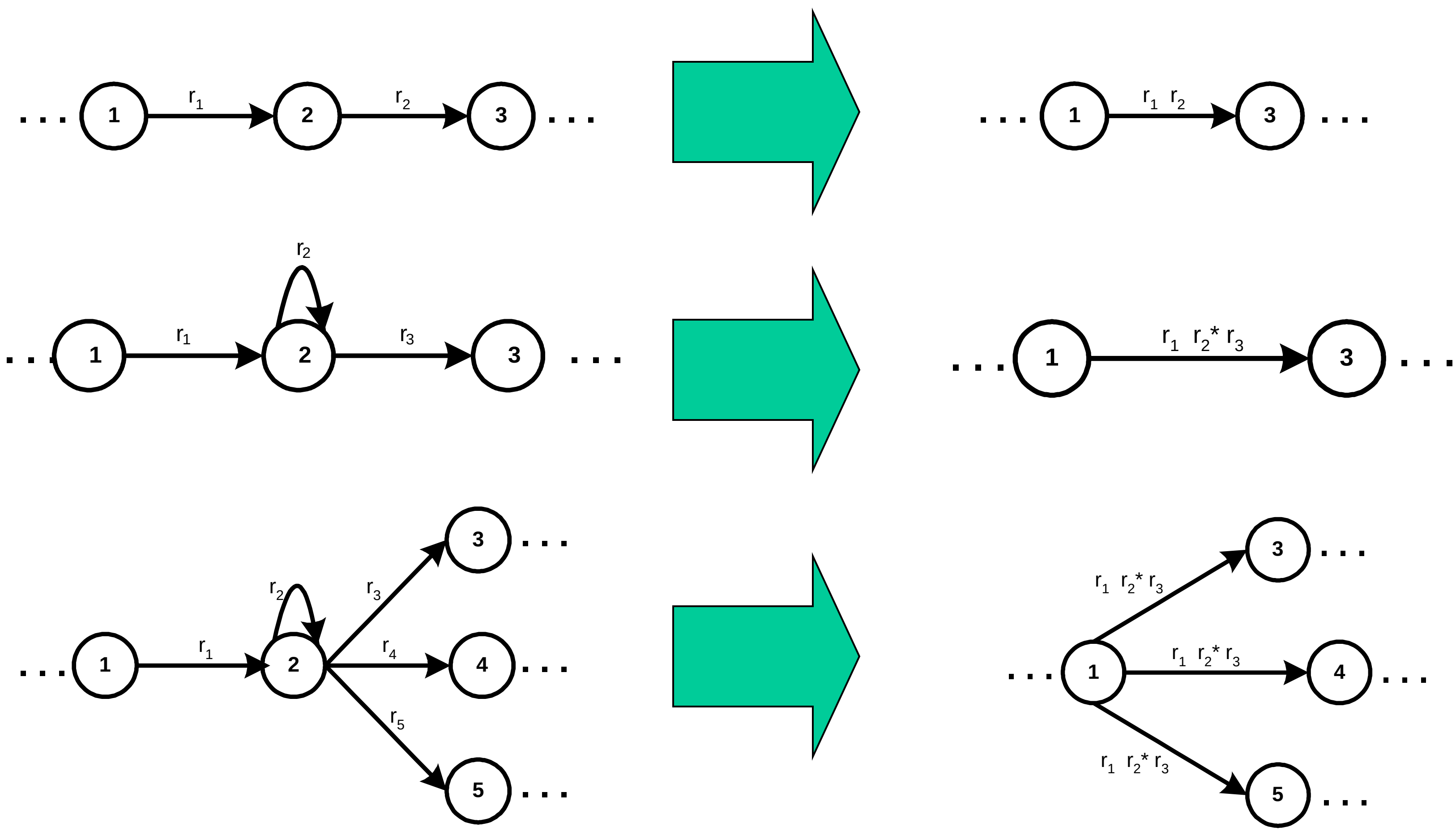
b. Meminimalkan Edge (Arc) dan Labelnya





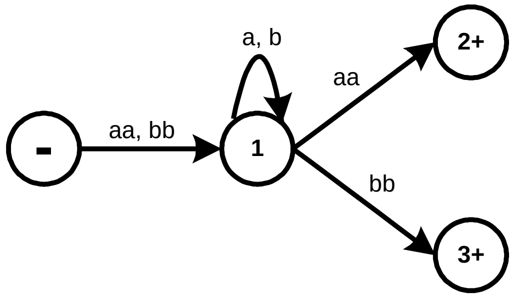
Pembuktian Tahap II (4)

c. Bypass Operation

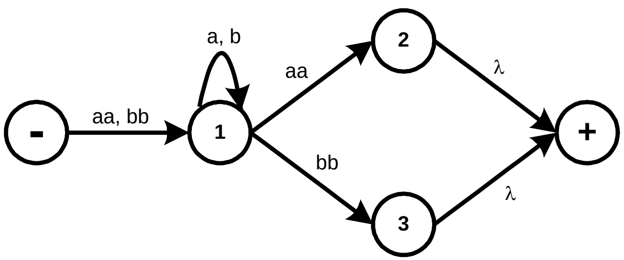


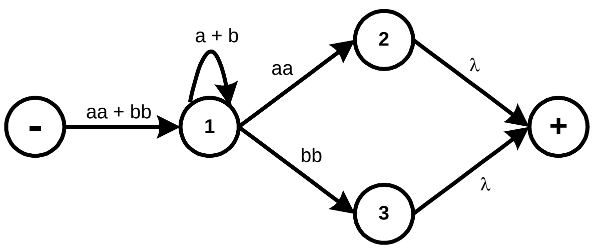
Pembuktian Tahap II (5)

Contoh :

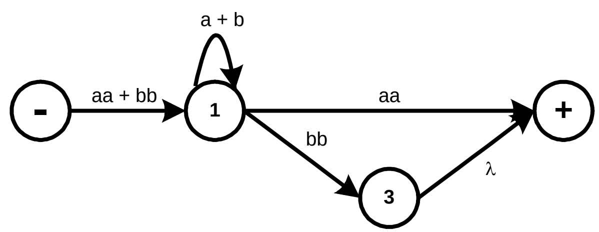


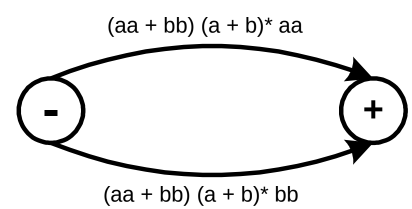
Minimisasi jumlah final sate :

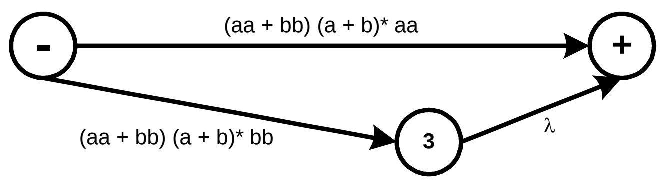




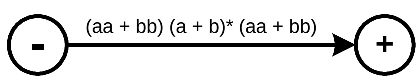
Bypass operation :





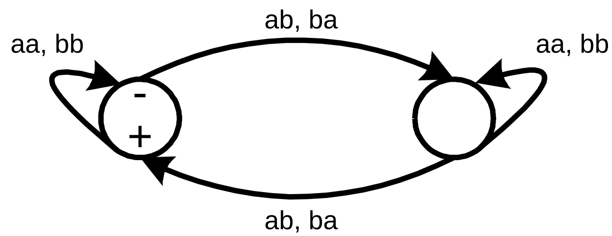


Minimisasi edge :

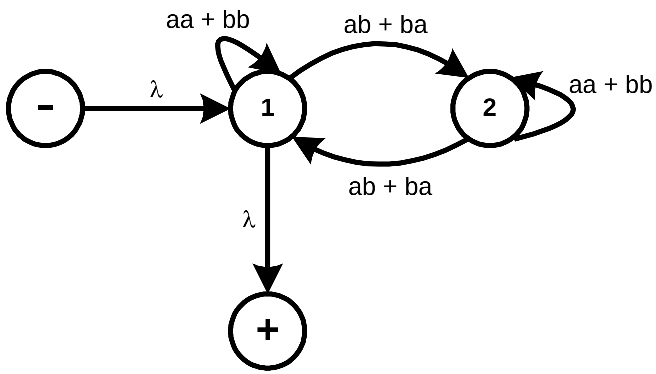


Pembuktian Tahap II (6)

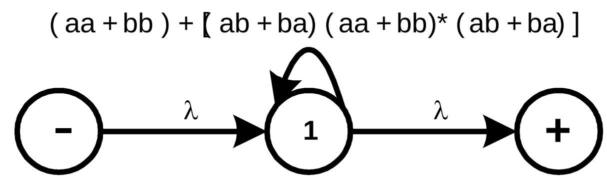
Contoh :



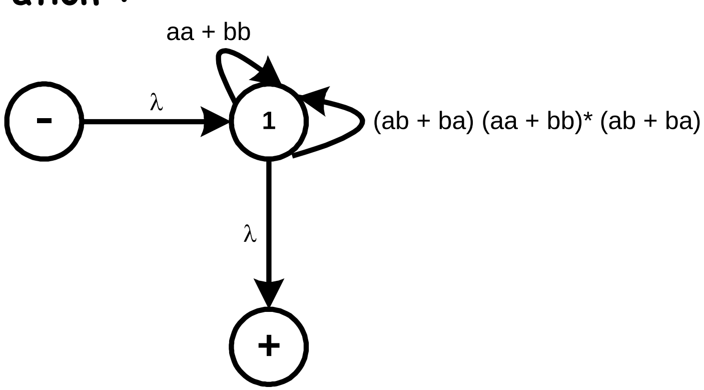
Pemisahan start dan final state :



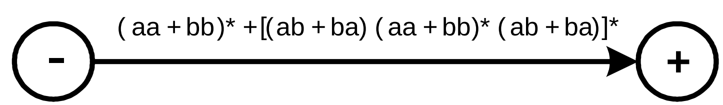
Minimisasi edge :



Bypass operation :

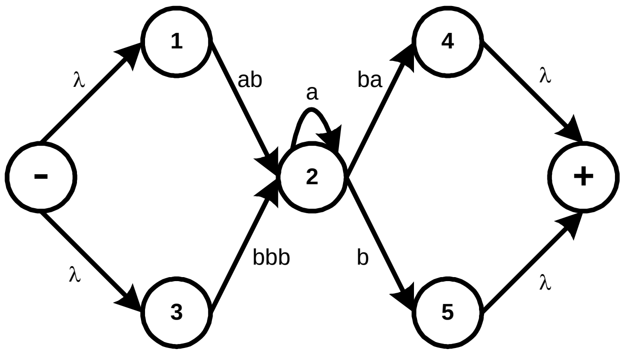


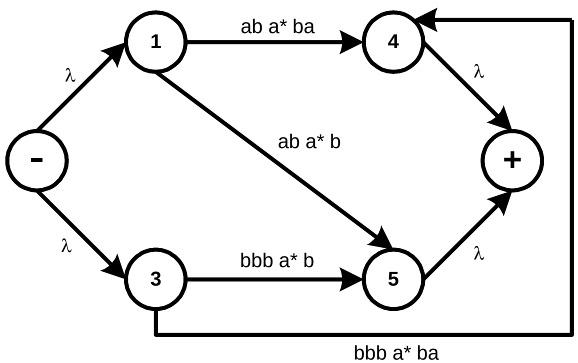
Bypass operation :

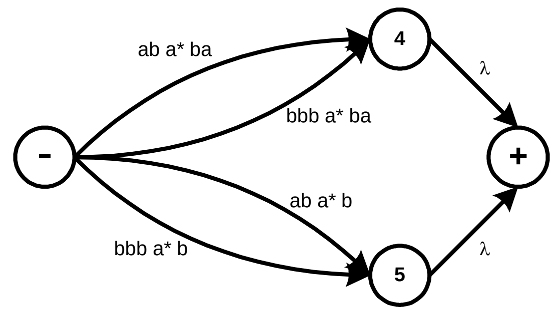


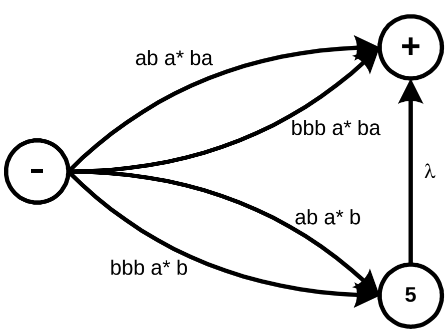
Pembuktian Tahap II (7)

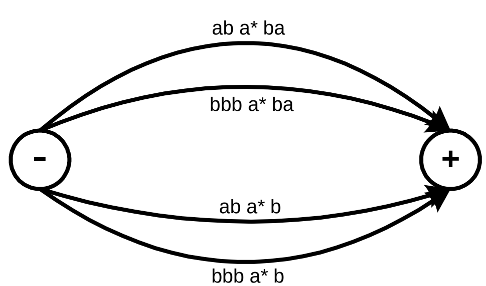
Contoh :

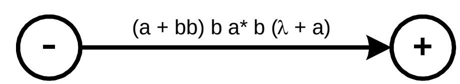












Pembuktian Tahap III (1)

Membuktikan bahwa setiap bahasa yang dapat didefinisikan melalui RE, pasti dapat pula didefinisikan melalui FA

Jika RE dapat menerima operasi-operasi himpunan, seperti

UNION, CONCATENATION dan CLOSURE.

Maka seharusnya FA juga dapat menerima operasi-operasi tersebut.

But how???

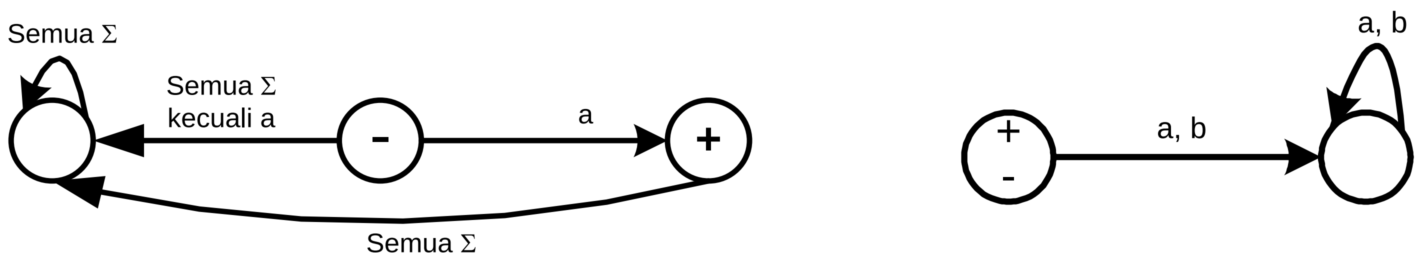
Pendekatan yang akan digunakan adalah menyusun algoritma untuk membentuk FA baru (sebagai hasil operasi himpunan) secara rekursif

Pembuktian Tahap III (2)

Beberapa aturan yang akan kita kaji, antara lain adalah :

ATURAN 1 :

Terdapat FA yang hanya dapat menerima elemen tertentu dari himpunan alphabet Σ dan terdapat pula FA yang hanya menerima λ.



Pembuktian Tahap III (3)

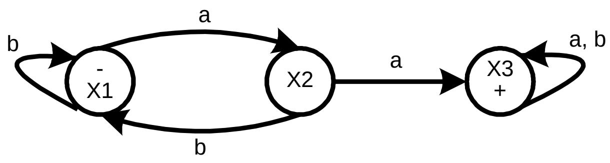
ATURAN 2 :

Misal terdapat mesin FA_1 yang menerima bahasa r_1, dan terdapat mesin FA_2 yang menerima bahasa r_2, maka akan dapat dibuat mesin FA_3 yang menerima bahasa $(r_1 + r_2)$.

Contoh :

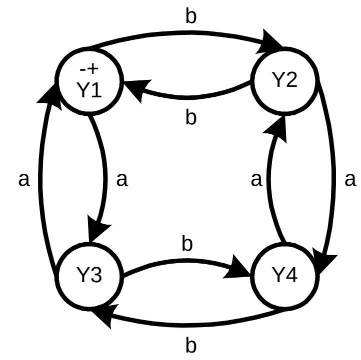
Misal bahasa r_1 yang didefinisikan melalui RE-nya : b* a (b+a)* a (a + b)*

Dan yang berasosiasi dengan RE di atas adalah FA_1 :



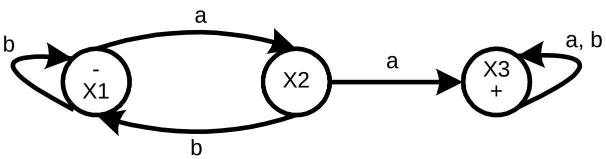
Misal bahasa r_2 adalah bahasa Even-Even (bahasa yang memiliki string dengan jumlah karakter a dan b masing-masing genap)

Dan mesin yang berasosiasi dengan bahasa di atas adalah FA_2 :

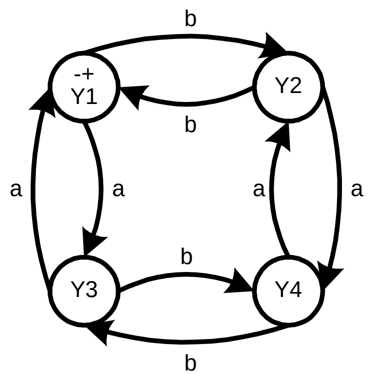


Pembuktian Tahap III (3)

FA$_1$:



FA$_2$:

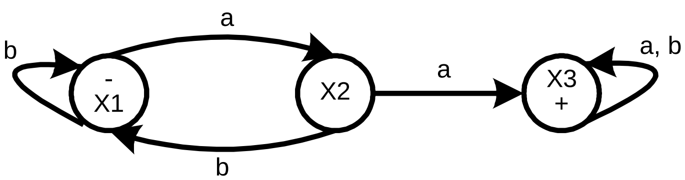


Tabel Transisi untuk FA$_3$:

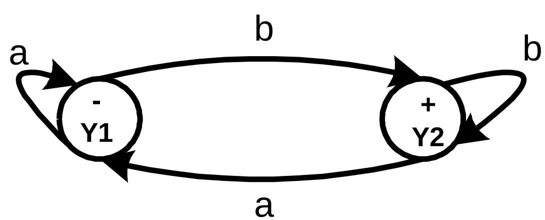
		a	b
+ - Z_1	(X_1, Y_1)	Z_2	Z_3
Z_2	(X_2, Y_3)	Z_4	Z_5
Z_3	(X_1, Y_2)	Z_6	Z_1
+ Z_4	(X_3, Y_1)	Z_7	Z_8
Z_5	(X_1, Y_4)	Z_9	Z_{10}
Z_6	(X_2, Y_4)	Z_8	Z_7
+ Z_7	(X_3, Y_3)	Z_4	Z_{11}
+ Z_8	(X_3, Y_2)	Z_{11}	Z_4
Z_9	(X_2, Y_2)	Z_{11}	Z_1
Z_{10}	(X_1, Y_3)	Z_{12}	Z_5
Z_{11}	(X_3, Y_4)	Z_8	Z_7
+ Z_{12}	(X_2, Y_1)	Z_7	Z_3

Pembuktian Tahap III (4)

FA$_1$:



FA$_2$:



Tabel Transisi untuk FA$_3$:

		a	b
- Z_1	(X_1, Y_1)	Z_2	Z_3
Z_2	(X_2, Y_1)	Z_4	Z_3
+ Z_3	(X_1, Y_2)	Z_2	Z_3
+ Z_4	(X_3, Y_1)	Z_4	Z_5
+ Z_5	(X_3, Y_2)	Z_4	Z_5

Pembuktian Tahap III (5)

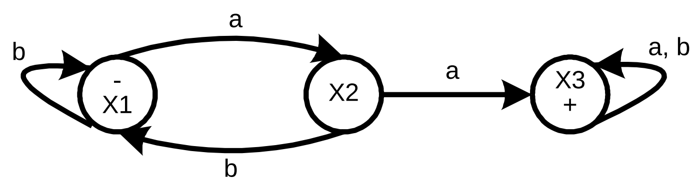
ATURAN 3 :

Misal terdapat mesin FA_1 yang menerima bahasa r_1, dan terdapat mesin FA_2 yang menerima bahasa r_2, maka akan dapat dibuat mesin FA_3 yang menerima bahasa (r_1 r_2).

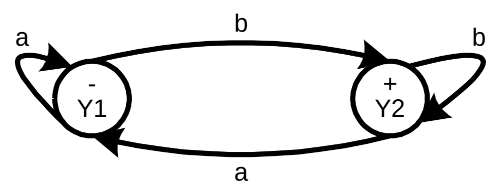
Contoh :

Misal pada himpunan $\Sigma = \{a, b\}$ terdapat :

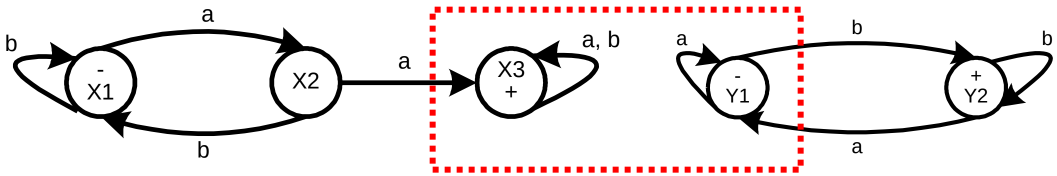
FA_1 = menerima semua string yang mengandung substring aa



FA_2 = menerima semua string dengan karakter terakhir b



Pembuktian Tahap III (6)



– $Z_1 = X_1$

Z_1 diberi input a ke X_2 atau disebut Z_2

Z_1 diberi input b ke X_1 atau disebut Z_1

Z_2 diberi input a ke X_3 (masih di FA_1)
atau Y_1 (mulai di FA_2)
$= Z_3 = X_3 \mid Y_1$

Z_2 diberi input b ke X_1 (= Z_1)

Z_3 diberi input a ke X_3 (looping dan masih di FA_1)
atau Y_1 (looping dan pindah ke FA_2)
atau Y_1 (looping dan masih di FA_2)
(= Z_3)

Z_3 diberi input b ke X_3 (looping dan masih di FA_1)
atau Y_1 (looping dan pindah ke FA_2)
atau Y_2 (dari Y_1 di FA_2)
$= X_3 \mid Y_1 \mid Y_2$ (= + Z_4)

Z_4 diberi input a ke X_3 (looping dan masih di FA_1)
atau Y_1 (looping dan pindah ke FA_2)
atau Y_1 (looping dan masih di FA_2)
atau Y_1 (dari Y_2 di FA_2)
(= Z_3)

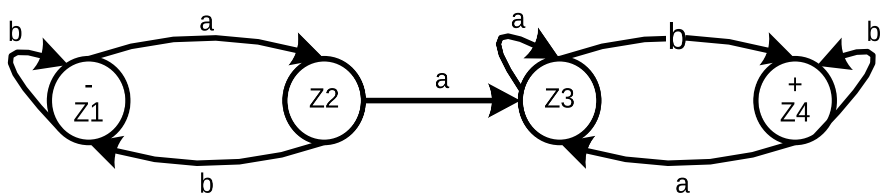
Z_4 diberi input b ke X_3 (looping dan masih di FA_1)
atau Y_1 (looping dan pindah ke FA_2)
atau Y_2 (dari Y_1 di FA_2)
atau Y_2 (looping di Y_2)
= + Z_4

Pembuktian Tahap III (7)

Tabel Transisi :

		a	b
- Z_1	X_1	Z_2	Z_1
Z_2	X_2	Z_3	Z_1
Z_3	$X_3 \mid Y_1$	Z_3	Z_4
+ Z_4	$X_3 \mid Y_1 \mid Y_2$	Z_3	Z_4

DFA :



Pembuktian Tahap III (8)

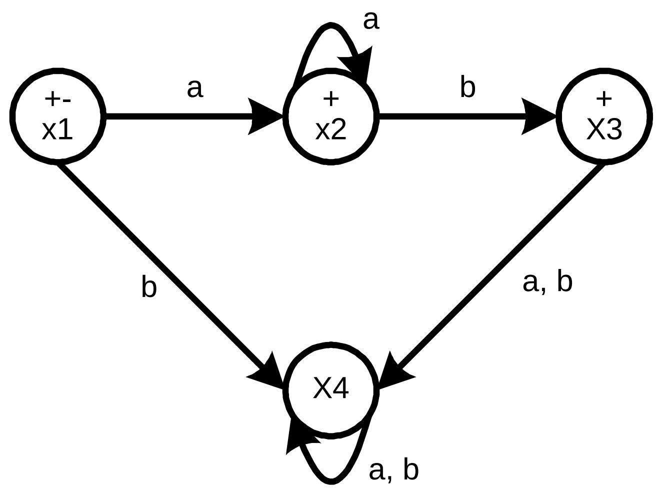
ATURAN 4 :

Misal terdapat mesin FA$_1$ yang menerima bahasa r$_1$maka akan dapat dibuat mesin FA$_2$ yang menerima bahasa (r$_1$)*.

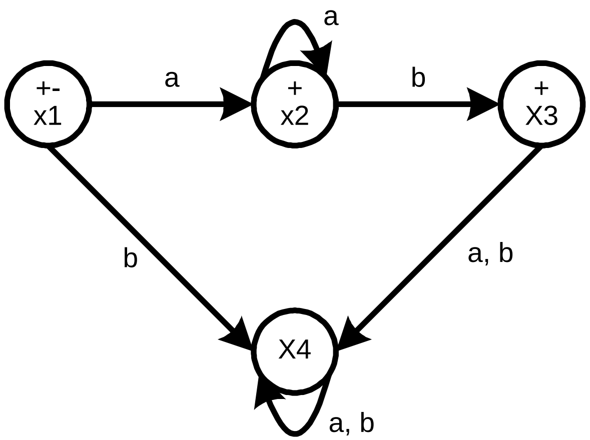
Contoh :

Misal terdapat sebuah RE : a* + aa*b

Dan FA$_1$ yang berkorespondensi dengan RE tersebut adalah :



Pembuktian Tahap III (9)



$\pm Z_1 = X_1$

Z_1 diberi input a ke X_2 (untuk terus)
atau X_1 (selesai di X_2 dan kembali)
$= + Z_2 = X_1 \mid X_2$

Z_1 diberi input b ke X_4 (dead-end state) $= Z_3 = X_4$

$+ Z_2$ diberi input a ke X_2 (untuk terus)
atau X_1 (selesai di X_2 dan kembali)
atau X_2 (untuk terus)
atau X_1 (selesai di X_2 dan kembali)
$= Z_2$

Z_2 diberi input b ke X_4 (dead-end state)
atau X_3 (untuk terus)
atau X_1 (selesai di X_3 dan kembali)
$= + Z_4 = X_1 \mid X_3 \mid X_4$

Z_3 diberi input a ke X_4 (dead-end state) $= Z_3$
Z_3 diberi input b ke X_4 (dead-end state) $= Z_3$

$+ Z_4$ diberi input a ke X_2 (untuk terus)
atau X_1 (selesai di X_2 & kembali)
atau X_4 (dari X_3)
atau X_4 (dari X_4)
$= + Z_5 = X_1 \mid X_2 \mid X_4$

Z_4 diberi input b ke X_4 (dari X_1 X_3 X_4)
$= Z_3$

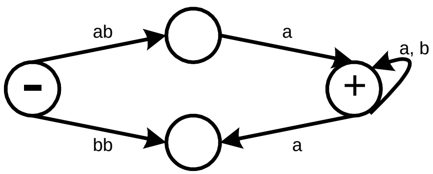
$+ Z_5$ diberi input a ke X_2 (untuk terus)
atau X_1 (selesai di X_2 & kembali)
atau X_2 (untuk terus)
atau X_1 (selesai di X_2 & kembali)
atau X_4 (dari X_4)
$= + Z_5$

Z_5 diberi input b ke X_4 (dead-end state)
atau X_3 (untuk terus)
atau X_1 (selesai di X_3 & kembali)
atau X_4 (dari X_4)
$= + Z_4$

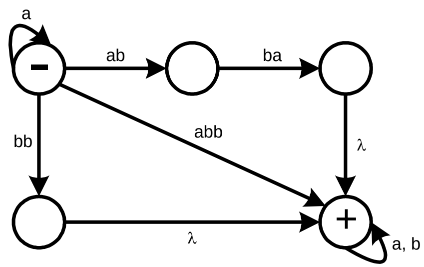
Tugas Individu

Carilah RE yang ekivalen dengan TG berikut :

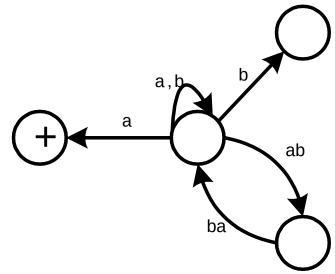
1



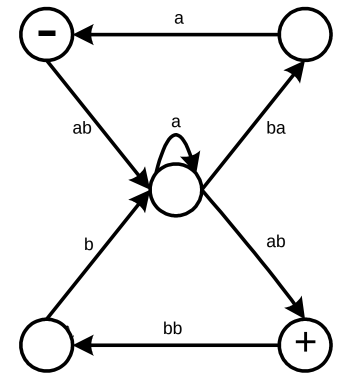
2



3

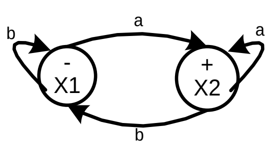


4

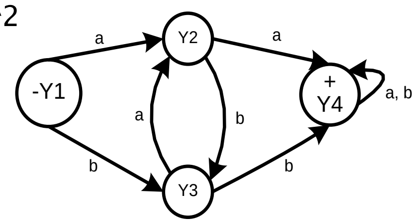


Melalui ketiga FA berikut, carilah FA baru yang dapat menerima bahasa seperti didefinisikan oleh operasi berikut :

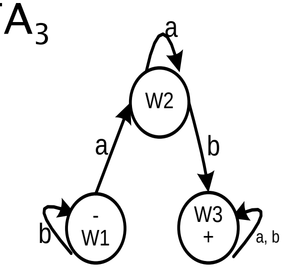
FA_1



FA_2



FA_3



5. $r_1 + r_2$
6. $r_1 r_3$
7. $(r_2)^*$
8. $(r_3)^*$