<u>Konsep Teori Bahasa dan Otomata</u>

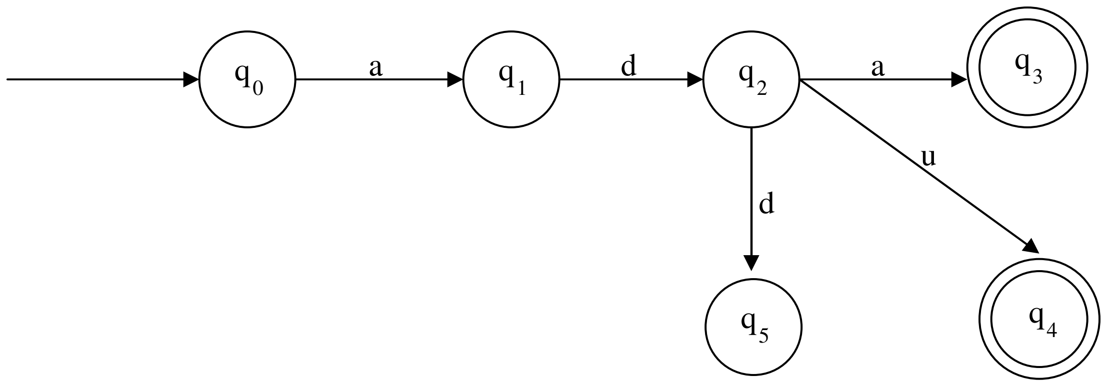
Teori bahasa dan otomata merupakan salah satu mata kuliah yang wajib di jurusan-jurusan teknik informatika maupun ilmu komputer. Teori bahasa dan otomata merupakan mata kuliah yang cenderung bersifat teoritis tidak memuat hal-hal yang 'praktis' untuk diterapkan langsung dalam praktik. Manfaat langsung dari mata kuliah teori bahasa dan otomata akan kita dapatkan ketika mempelajari mata kuliah Teknik Kompilasi.

Bahasa di dalam kamus adalah suatu sistem yang meliputi pengekspresian gagasan, fakta, konsep, termasuk sekumpulan simbol-simbol dan aturan untuk melakukan manipulasinya. Bahasa bisa juga disebut sebagai rangkaian simbol-simbol yang mempunyai makna.

Otomata merupakan suatu sistem yang terdiri atas sejumlah berhingga *state*, di mana *state* menyatakan informasi mengenai *input*. Otomata juga dianggap sebagai mesin otomatis (bukan mesin fisik) yang merupakan suatu model matematika dari suatu sistem yang menerima input dan menghasilkan output, serta terdiri dari sejumlah berhingga state.

Hubungan di antara bahasa dan otomata adalah bahasa dijadikan sebagai input oleh suatu mesin otomata, selanjutnya mesin otomata akan membuat keputusan yang mengindikasikan apakah input itu diterima atau tidak.

Misalnya, kita memiliki sebuah mesin sederhana yang menerima input kata dalam bahasa Indonesia, hal ini bisa dilihat pada gambar berikut ini.



Pada gambar di atas, bila mesin mendapat string input berikut.

1. ada : diterima
2. adu : diterima
3. add : ditolak

Sebuah string input diterima bila mencapai state akhir / final state yang disana digambarkan dengan lingkaran ganda. Mesin ini memiliki 6 state, $\{q_0, q_1, q_2, q_3, q_4, q_5\}$, yang mana adalah himpunan state yang ada pada mesin itu. State awal dari mesin adalah q_0. $\{q_3, q_4\}$ adalah himpunan state akhir / final. Sedangkan himpunan simbol input adalah {a, d, u}.

<u>Hirarki Chomsky</u>

Tata bahasa (*grammar*) bisa didefinisikan secara formal sebagai kumpulan dari himpunan-himpunan variabel, simbol-simbol terminal, simbol awal, yang dibatasi oleh aturan-aturan produksi. Pada tahun 1959, seorang ahli bernama Noam Chomsky melakukan penggolongan tingkatan bahasa menjadi empat, yang disebut dengan hirarki Chomsky. Penggolongan tersebut bisa dilihat pada tabel berikut.

Bahasa	Mesin Otomata	Batasan Aturan Produksi
Regular	*Finite State Automata* (FSA) meliputi *Deterministic Finite Automata* (DFA) & *Non Deterministic Finite Automata* (NFA)	α adalah sebuah simbol variabel. β maksimal memiliki sebuah simbol variabel yang bila ada terletak di posisi paling kanan
Bebas Konteks / *Context Free*	*Push Down Automata* (PDA)	α berupa sebuah simbol variabel
Context Sensitive	*Linier Bounded Automata*	$\lvert\alpha\rvert \leq \lvert\beta\rvert$
Unrestricted / Phase Structure / Natural Language	Mesin Turing	Tidak ada batasan

Secara umum tata bahasa dirumuskan sebagai :

$\alpha \rightarrow \beta$, yang berarti α menghasilkan β atau α menurunkan β.

Di mana α menyatakan simbol-simbol pada ruas kiri aturan produksi (sebelah kiri tanda '→') dan β menyatakan simbol-simbol pada ruas kanan aturan produksi (sebelah kanan tanda '→')

Simbol variabel / non terminal adalah simbol yang masih bisa diturunkan dan ditandai dengan huruf besar seperti A, B, C, dst.

Simbol terminal adalah simbol yang sudah tidak bisa diturunkan dan ditandai dengan huruf kecil seperti a, b, c, dst.

Tata Bahasa Regular

Aturan :

- Simbol pada Sebelah kiri harus berupa sebuah simbol variabel
- Simbol pada sebelah kanan maksimal hanya memiliki sebuah simbol variabel dan bila ada terletak di posisi paling kanan.

Contoh :

A → b (Diterima)

a → B (Ditolak, karena simbol pada sebelah kiri harus berupa sebuah simbol variabel)

A → B (Diterima)

A → bC (Diterima)

A → Bc (Ditolak, karena simbol variabel pada sebelah kanan harus berada pada posisi paling kanan)

A → bcD (Diterima)

A → bCD (Ditolak, karena simbol pada sebelah kanan maksimal hanya memiliki sebuah simbol variabel)

Ab → c (Ditolak, karena simbol pada sebelah kiri harus berupa sebuah simbol variabel)

Tentukan apakah produksi-produksi berikut memenuhi aturan tata bahasa Regular

1. A → b
2. B → bdB
3. B → C
4. B → bC
5. B → Ad
6. B → bcdef
7. B → bcdefg
8. A → aSa
9. A → aSS
10. A → ϵ

11. Ad → dB

Tata Bahasa Bebas Konteks

Aturan :

- Simbol pada Sebelah kiri harus berupa sebuah simbol variabel

Contoh :

A → b (Diterima)

A → B (Diterima)

A → bC (Diterima)

A → Bc (Diterima)

A → BcD (Diterima)

A → AAA (Diterima)

a → b (Ditolak, karena simbol pada sebelah kiri harus berupa sebuah simbol variabel)

Ab → c (Ditolak, karena simbol pada sebelah kiri harus berupa sebuah simbol variabel)

AB → c (Ditolak, karena simbol pada sebelah kiri harus berupa sebuah simbol variabel)

Tentukan apakah aturan produksi-produksi berikut memenuhi aturan tata bahasa bebas konteks.

1. A → aSa
2. A → Ace
3. A → ab
4. A → ϵ
5. B → bcdef
6. B → bcdefG
7. A → aSa
8. A → aSS

9. A → BCDEF
10. Ad → dB
11. A → AAAAA
12. d → A

Tata Bahasa *Context Sensitive*

Aturan : - Simbol pada Sebelah kiri harus minimal ada sebuah simbol variabel - Jumlah simbol pada ruas sebelah kiri harus lebih kecil atau sama dengan jumlah simbol pada ruas kanan

Contoh :

A → bc (Diterima)

Ab → cd (Diterima)

AB → CD (Diterima)

ABC → DE (Ditolak, karena jumlah simbol pada ruas sebelah kiri lebih bayak dari jumlah simbol pada ruas kanan)

Ab → cDe (Diterima)

bA → cd (Diterima)

a → b (Ditolak, karena simbol pada sebelah kiri harus minimal ada sebuah simbol variabel)

Tentukan apakah produksi-produksi berikut memenuhi aturan tata bahasa *context sensitive*.

1. B → bcdefG
2. A → aSa
3. A → aSS
4. A → BCDEF
5. Ad → dB
6. A → ϵ

7. AB → ϵ
8. ad → b
9. ad → ϵ
10. abC → DE
11. abcDef → ghijkl
12. AB → cde
13. AAA → BBB

<u>Tata Bahasa *Unrestricted*</u>

Aturan : - Simbol pada Sebelah kiri harus minimal ada sebuah simbol variabel

Contoh :

Abcdef → g (Diterima)

aBCdE → GHIJKL (Diterima)

abcdef → GHIJKL (Ditolak, karena simbol pada sebelah kiri tidak ada sebuah simbol variabel)

Tentukan apakah produksi-produksi berikut memenuhi aturan tata bahasa *unrestricted*.

1. A → ϵ
2. AB → ϵ
3. ad → b
4. ad → ϵ
5. abC → DE
6. AB → cde
7. e → a
8. ABCDEFG → h
9. bA → CDEFGH

Finite State Automata

Finite State Automata / *State Otomata* berhingga, selanjutnya kita sebut sebagai FSA, bukanlah mesin fisik tetapi suatu model matematika dari suatu sistem yang menerima *input* dan *output* diskrit.

Finite State Automata merupakan mesin otomata dari bahasa regular. Suatu *Finite State Automata* memiliki *state* yang banyaknya berhingga, dan dapat berpindah-pindah dari suatu *state* ke *state* lain.

Secara formal *finite state automata* dinyatakan oleh 5 tupel atau M=(Q, Σ, δ, S, F), di mana :

Q = himpunan *state* / kedudukan

Σ = himpunan simbol *input* / masukan / abjad

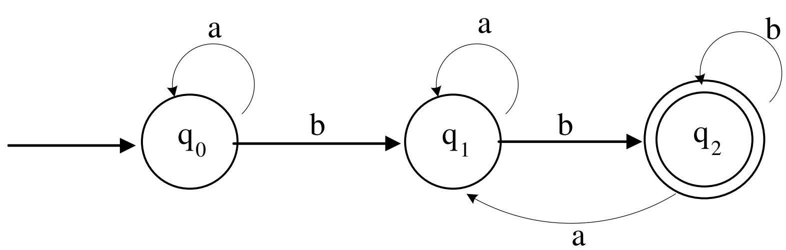
δ = fungsi transisi

S = *state* awal / kedudukan awal (*initial state*)

F = himpunan *state* akhir

Finite State Automata yang memiliki tepat satu *state* berikutnya untuk setiap simbol masukan yang diterima disebut *Deterministic Finite Automata*.

Sebagai contoh, kita memiliki sebuah otomata seperti pada gambar di bawah ini.



Konfigurasi *Deterministic Finite Automata* di atas secara formal dinyatakan sebagai berikut.

Q = $\{q_0, q_1, q_2\}$

Σ = {a,b}

S = q_0

F = $\{ q_2 \}$

Fungsi transisi yang ada sebagai berikut.

$d(q_0, a) = q_0$

$d(q_0, b) = q_1$

$d(q_1, a) = q_1$

$d(q_1, b) = q_2$

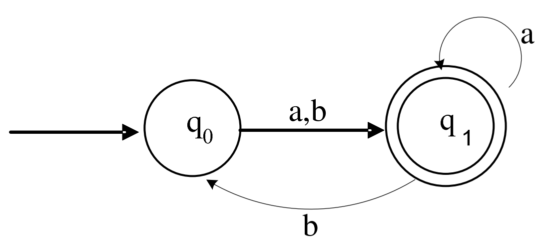
$d(q_2, a) = q_1$

$d(q_2, b) = q_2$

Biasanya fungsi-fungsi transisi ini kita sajikan dalam sebuah tabel transisi. Tabel transisi tersebut menunjukkan *state-state* berikutnya untuk kombinasi *state-state* dan *input*. Tabel transisi dari fungsi transisi di atas sebagai berikut.

δ	a	b
q_0	q_0	q_1
q_1	q_1	q_2
q_2	q_1	q_2

Contoh lain bisa dilihat pada gambar di bawah ini.

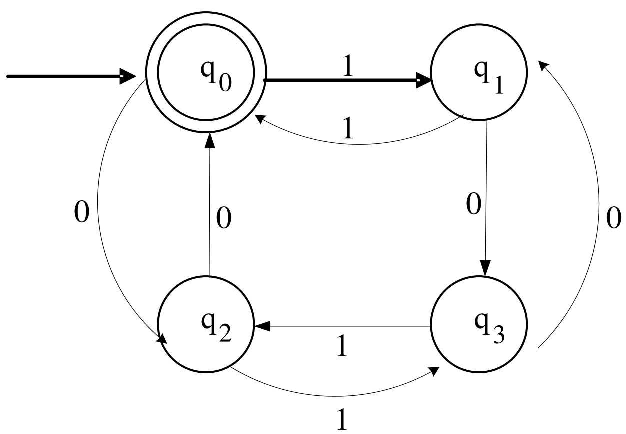


Tabel transisi dari gambar di atas adalah sebagai berikut

δ	a	b
q_0	q_1	q_1
q_1	q_1	q_0

Soal :

Buatlah tabel transisi dari *Deterministic Finite Automata* berikut.



Konversi dari Tabel Transisi ke Diagram Transisi

Sebaliknya, Kita juga dapat menggambar diagram transisi dari suatu tabel transisi.

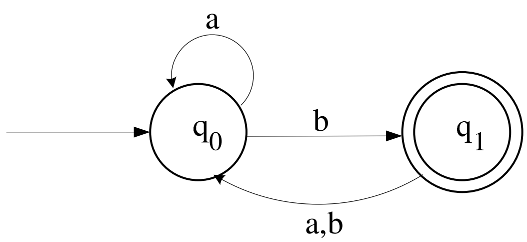
δ	a	b
q_0	q_0	q_1
q_1	q_0	q_0

Dengan

$S = q_0$

$F = \{q_1\}$

Maka diagram transisinya adalah sebagai berikut.



Contoh lain, terdapat tabel transisi sebagai berikut.

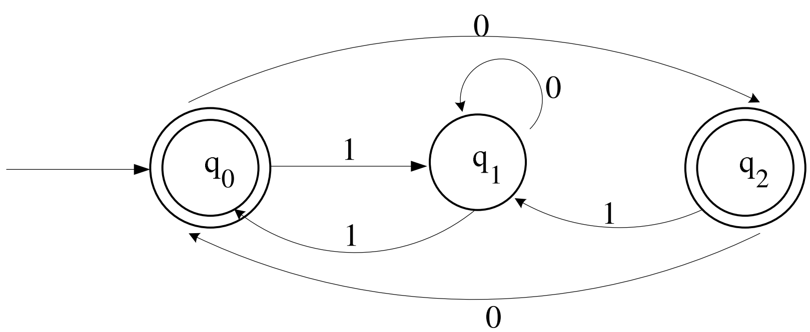
δ	0	1
q_0	q_2	q_1
q_1	q_1	q_0
q_2	q_0	q_1

Dengan

$S = q_0$

$F = \{q_0, q_2\}$

Diagram transisinya dapat kita lihat pada gambar di bawah ini.



Soal :

Gambarkan diagram transisi dari *Deterministic Finite Automata* berikut.

$Q = \{q_0, q_1, q_2\}$

$\Sigma = \{a,b\}$

$S = q_0$

$F = \{q_0\}$

Tabel transisi dari DFA tersebut :

δ	a	B
q_0	q_1	q_2
q_1	q_2	q_0
q_2	q_2	q_2

Gambarkan diagram transisi dari *Deterministic Finite Automata* berikut.

$Q = \{q_0, q_1, q_2, q_3\}$

$\Sigma = \{a,b\}$

$S = q_0$

$F = \{q_0, q_1, q_2\}$

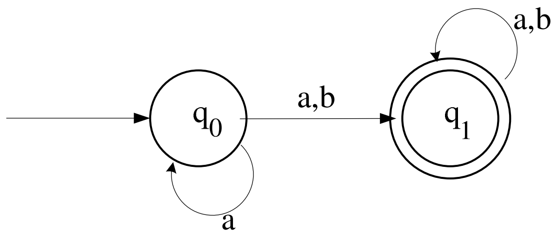
Fungsi transisi dari DFA tersebut :

δ	a	B
q_0	q_0	q_1
q_1	q_0	q_2
q_2	q_0	q_3
q_3	q_3	q_2

Perhatikan pada contoh-contoh *Deterministic Finite Automata* pada contoh-contoh sebelumnya, terlihat bahwa dari setiap *state* selalu tepat ada satu *state* berikutnya untuk setiap simbol *input* yang ada. Berbeda halnya dengan *Non Deterministic Finite Automata* (NFA). Pada NFA, dari suatu input mungkin saja bisa dihasilkan lebih dari satu *state* berikutnya.

Non Deterministic Finite Automata

Non Deterministic Finite Automata didefinisikan pula dengan lima (5) tupel, sama seperti halnya pada *Deterministic Finite Automata*. Perhatikan contoh di bawah ini.



Perhatikan gambar di atas, bila *state* q_0 mendapat input 'a' bisa berpindah ke *state* q_0 atau q_1, yang secara formal dinyatakan :

$\delta(q_0, a) = \{q_0, q_1\}$

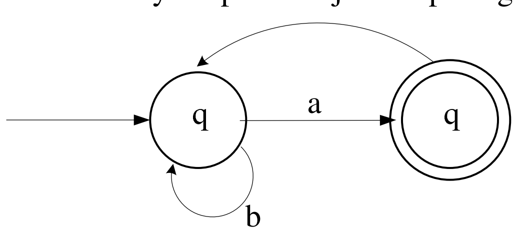
Maka otomata ini disebut non-deterministik (tidak pasti arahnya). Bisa kita lihat tabel transisinya seperti di bawah ini.

δ	a	B
q_0	$\{q_0,q_1\}$	$\{q_1\}$
q_1	$\{q_1\}$	$\{q_1\}$

Catatan :

Perhatikan cara penulisan *state* hasil transisi pada tabel transisi untuk *Non Deterministic Finite Automata* digunakan kurung kurawal '{' dan '}' karena hasil transisinya merupakan suatu himpunan *state*

Contoh lainnya dapat ditunjukkan pada gambar di bawah ini :



Kita bisa melihat tabel transisinya di bawah ini :

δ	a	B
q_0	$\{q_1\}$	$\{q_0\}$
q_1	$\{q_0\}$	Ø

Seperti halnya pada *Deterministic Finite Automata*, pada *Non Deterministic Finite Automata* kita juga bisa membuat diagram transisinya dari tabel transisinya.

Soal :

Gambarlah diagram transisi untuk NFA berikut :

$Q = \{q_0, q_1, q_2, q_3, q_4\}$
$\Sigma = \{0,1\}$
$S = q_0$
$F = \{q_2, q_4\}$

Fungsi transisi dari NFA tersebut :

δ	0	1
q_0	$\{q_0,q_3\}$	$\{q_0,q_1\}$
q_1	Ø	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$
q_3	$\{q_4\}$	Ø
q_4	$\{q_4\}$	$\{q_4\}$

Gambarlah diagram transisi untuk NFA berikut :

$Q = \{q_0, q_1\}$

$\Sigma = \{0,1\}$

$S = q_0$

$F = \{q_1\}$

Fungsi transisi dari NFA tersebut :

δ	0	1
q_0	$\{q_0,q_1\}$	$\{q_1\}$
q_1	Ø	$\{q_0,q_1\}$

Reduksi Jumlah State pada Finite State Automata

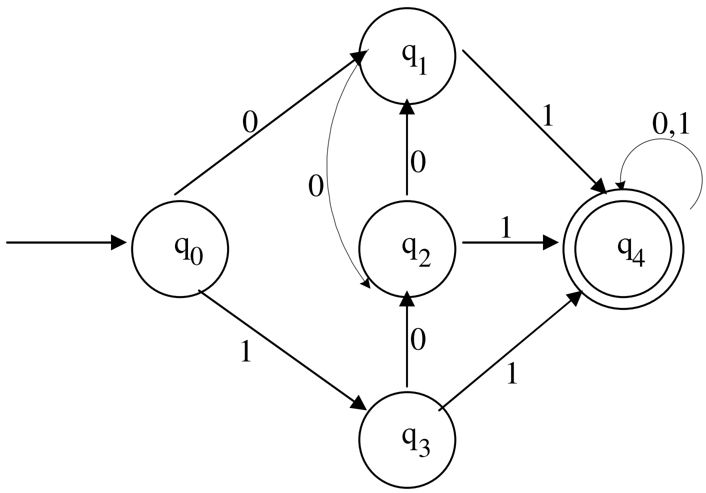
Untuk suatu bahasa regular, kemungkinan ada sejumlah *Deterministic Finite Automata* yang dapat menerimanya. Perbedaannya hanyalah jumlah *state* yang dimiliki otomata-otomata yang saling ekuivalen tersebut. Tentu saja, dengan alasan kepraktisan, kita memilih otomata dengan jumlah *state* yang lebih sedikit.

Sasaran kita di sini adalah mengurangi jumlah *state* dari suatu *Finite State Automata*, dengan tidak mengurangi kemampuannya semula untuk menerima suatu bahasa.

Ada dua buah istilah baru yang perlu kita ketahui yaitu :

1. *Distinguishable* yang berarti dapat dibedakan.
2. *Indistinguishable* yang berarti tidak dapat dibedakan.

Sebagai contoh kita ingin menyederhanakan DFA berikut.



Langkah-Langkahnya :

1. Identifikasilah setiap kombinasi *state* yang mungkin :

Kombinasi state yang mungkin adalah :

(q_0, q_1)

(q_0, q_2)

(q_0, q_3)

(q_0, q_4)

(q_1, q_2)

(q_1, q_3)

(q_1, q_4)

(q_2, q_3)

(q_2, q_4)

(q_3, q_4)

2. *State* yang berpasangan dengan state akhir (q_4) merupakan *state* yang distinguishable

(q_0, q_1)

(q_0, q_2)

(q_0, q_3)

(q_0, q_4) : Distinguishable

(q_1, q_2)

(q_1, q_3)

(q_1, q_4) : Distinguishable

(q_2, q_3)

(q_2, q_4) : Distinguishable

(q_3, q_4) : Distinguishable

3. Untuk pasangan *state* yang lain jika masing-masing state mendapat input yang sama, maka bila satu state mencapai state akhir dan yang lain tidak mencapai state akhir maka dikatakan distinguishable.

Untuk (q_0, q_1) :

$\delta (q_0, 1) = q_3$

$\delta (q_1, 1) = q_4$

$\delta (q_0, 0) = q_1$

$\delta (q_1, 0) = q_2$

Maka (q_0, q_1) : Distinguishable

Untuk (q_0, q_2) :

$\delta (q_0, 1) = q_3$

$\delta (q_2, 1) = q_4$

$\delta(q_0, 0) = q_1$

$\delta(q_2, 0) = q_1$

Maka (q_0, q_2) : Distinguishable

Untuk (q_0, q_3) :

$\delta(q_0, 1) = q_3$

$\delta(q_3, 1) = q_4$

$\delta(q_0, 0) = q_1$

$\delta(q_3, 0) = q_2$

Maka (q_0, q_3) : Distinguishable

Untuk (q_1, q_2)

$\delta(q_1, 1) = q_4$

$\delta(q_2, 1) = q_4$

$\delta(q_1, 0) = q_2$

$\delta(q_2, 0) = q_1$

Maka (q_1, q_2) : Indistinguishable

Untuk (q_1, q_3)

$\delta(q_1, 1) = q_4$

$\delta(q_3, 1) = q_4$

$\delta(q_1, 0) = q_2$

$\delta(q_3, 0) = q_2$

Maka (q_1, q_3) : Indistinguishable

Untuk (q_2, q_3)

$\delta (q_2, 1) = q_4$

$\delta (q_3, 1) = q_4$

$\delta (q_2, 0) = q_1$

$\delta (q_3, 0) = q_2$

Maka (q_2, q_3) : Indistinguishable

4. Maka Didapatkan pasangan state sebagai berikut :

(q_0, q_1) : Distinguishable

(q_0, q_2) : Distinguishable

(q_0, q_3) : Distinguishable

(q_0, q_4) : Distinguishable

(q_1, q_2) : Indistinguishable

(q_1, q_3) : Indistinguishable

(q_1, q_4) : Distinguishable

(q_2, q_3) : Indistinguishable

(q_2, q_4) : Distinguishable

(q_3, q_4) : Distinguishable

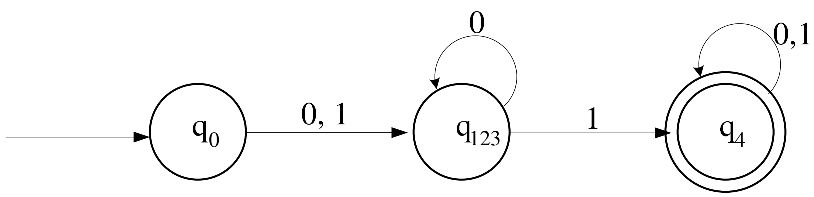
5. Kelompokkan pasangan state yang indistinguishable :

(q_1, q_2) : Indistinguishable

(q_1, q_3) : Indistinguishable

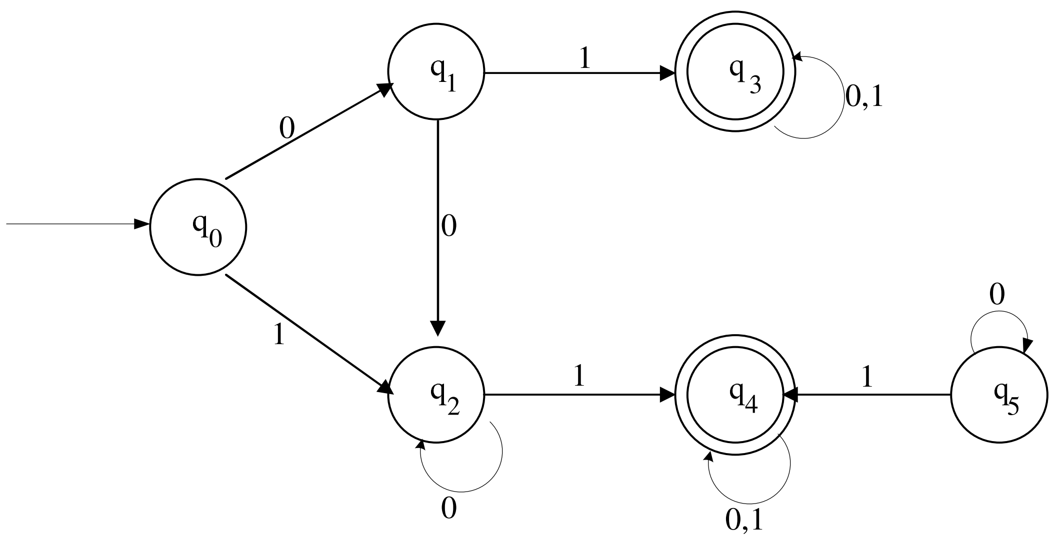
(q_2, q_3) : Indistinguishable

6. Karena q_1 indistinguishable dengan q_2 dan q_2 indistinguishable dengan q_3, maka bisa dikatakan bahwa q_1, q_2, dan q_3 saling indistinguishable dan dapat dijadikan satu state.
7. Sehingga hasil penyederhanaannya adalah sebagai berikut :

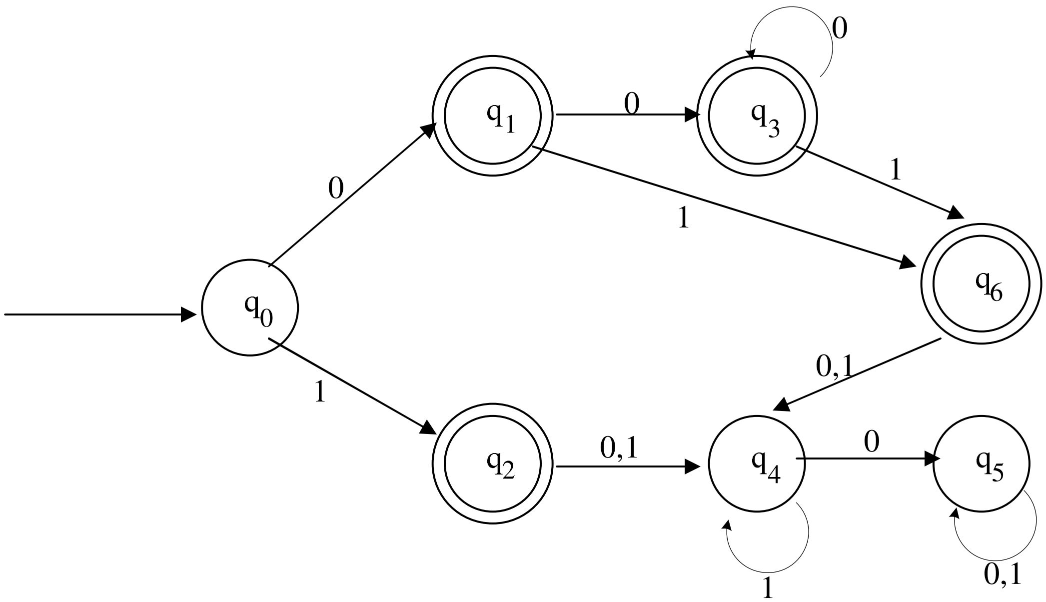


Soal :

Lakukan reduksi jumlah *state* pada *Deterministic Finite Automata* pada gambar berikut.



Lakukan reduksi jumlah state pada *Deterministic Finite Automata* berikut.



Pembahasan :

Soal No. 1

1. Identifikasi setiap kombinasi *state* yang mungkin :

   Kombinasi *state* yang mungkin :

   (q_0, q_1)

   (q_0, q_2)

   (q_0, q_3)

   (q_0, q_4)

   (q_0, q_5)

   (q_1, q_2)

   (q_1, q_3)

   (q_1, q_4)

   (q_1, q_5)

   (q_2, q_3)

   (q_2, q_4)

(q_2, q_5)

(q_3, q_4)

(q_3, q_5)

(q_4, q_5)

2. State yang berpasangan dengan *state* akhir (q_3 dan q_4) merupakan *state* yang distinguishable.

   (q_0, q_1) :

   (q_0, q_2) :

   (q_0, q_3) : Dis

   (q_0, q_4) : Dis

   (q_0, q_5) :

   (q_1, q_2) :

   (q_1, q_3) : Dis

   (q_1, q_4) : Dis

   (q_1, q_5) :

   (q_2, q_3) : Dis

   (q_2, q_4) : Dis

   (q_2, q_5) :

   (q_3, q_4) :

   (q_3, q_5) : Dis

   (q_4, q_5) : Dis

3. Untuk pasangan *state* yang lain jika masing-masing state mendapat input yang sama, maka bila satu state mencapai state akhir dan yang lain tidak mencapai state akhir maka dikatakan distinguishable.

Untuk (q_0, q_1)

$\delta(q_0, 1) = q_2$

$\delta(q_1, 1) = q_3$

$\delta(q_0, 0) = q_1$

$\delta(q_1, 0) = q_2$

Maka (q_0, q_1) : Distinguishable

Untuk (q_0, q_2)

$\delta(q_0, 1) = q_2$

$\delta(q_2, 1) = q_4$

$\delta(q_0, 0) = q_1$

$\delta(q_2, 0) = q_2$

Maka (q_0, q_2) : Distinguishable

Untuk (q_0, q_5)

$\delta(q_0, 1) = q_2$

$\delta(q_5, 1) = q_4$

$\delta(q_0, 0) = q_1$

$\delta(q_5, 0) = q_5$

Maka (q_0, q_5) : Distinguishable

Untuk (q_1, q_2)

$\delta(q_1, 1) = q_3$

$\delta(q_2, 1) = q_4$

δ $(q_1, 0) = q_2$

δ $(q_2, 0) = q_2$

Maka (q_1, q_2) : Indistinguishable

Untuk (q_1, q_5)

δ $(q_1, 1) = q_3$

δ $(q_5, 1) = q_4$

δ $(q_1, 0) = q_2$

δ $(q_5, 0) = q_5$

Maka (q_1, q_5) : Indistinguishable

Untuk (q_2, q_5)

δ $(q_2, 1) = q_4$

δ $(q_5, 1) = q_4$

δ $(q_2, 0) = q_2$

δ $(q_5, 0) = q_5$

Maka (q_1, q_5) : Indistinguishable

Untuk (q_3, q_4)

δ $(q_3, 1) = q_3$

δ $(q_4, 1) = q_4$

δ $(q_3, 0) = q_3$

δ $(q_4, 0) = q_4$

Maka (q_3, q_4) : Indistinguishable

4. Maka didapatkan pasangan *state* sebagai berikut.

   (q_0, q_1) : Dis

   (q_0, q_2) : Dis

   (q_0, q_3) : Dis

   (q_0, q_4) : Dis

   (q_0, q_5) : Dis

   (q_1, q_2) : Indis

   (q_1, q_3) : Dis

   (q_1, q_4) : Dis

   (q_1, q_5) : Indis

   (q_2, q_3) : Dis

   (q_2, q_4) : Dis

   (q_2, q_5) : Indis

   (q_3, q_4) : Indis

   (q_3, q_5) : Dis

   (q_4, q_5) : Dis

5. Kelompokkan pasangan *state* yang indistinguishable

   (q_1, q_2) : Indis

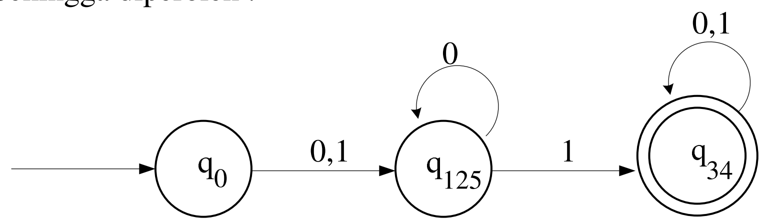
   (q_1, q_5) : Indis

   (q_2, q_5) : Indis

   (q_3, q_4) : Indis

6. Karena q_1 dan q_2 *indistinguishable* dan q_2 *indistinguishable* dengan q_5 serta q_1 juga *indistinguishable* dengan q_5. Maka bisa dikatakan bahwa q_1, q_2, dan q_5 saling *indistinguishable* dan dapat dijadikan satu *state*.

   Selain itu q_3 dan q_4 yang saling *indistinguishable* juga dapat dijadikan satu *state*. Sehingga diperoleh :



Soal No. 2

1. Identifikasilah setiap kombinasi *state* yang mungkin :

   Kombinasi setiap *state* yang mungkin :

   (q_0, q_1)

   (q_0, q_2)

   (q_0, q_3)

   (q_0, q_4)

   (q_0, q_5)

   (q_0, q_6)

   (q_1, q_2)

   (q_1, q_3)

   (q_1, q_4)

   (q_1, q_5)

   (q_1, q_6)

   (q_2, q_3)

   (q_2, q_4)

   (q_2, q_5)

(q_2, q_6)

(q_3, q_4)

(q_3, q_5)

(q_3, q_6)

(q_4, q_5)

(q_4, q_6)

(q_5, q_6)

2. State yang berpasangan dengan *state* akhir (q_1, q_2, q_3, dan q_6) merupakan *state* yang distinguishable.

   (q_0, q_1) : Dis

   (q_0, q_2) : Dis

   (q_0, q_3) : Dis

   (q_0, q_4) :

   (q_0, q_5) :

   (q_0, q_6) : Dis

   (q_1, q_2) :

   (q_1, q_3) :

   (q_1, q_4) : Dis

   (q_1, q_5) : Dis

   (q_1, q_6) :

   (q_2, q_3) :

   (q_2, q_4) : Dis

   (q_2, q_5) : Dis

   (q_2, q_6) :

   (q_3, q_4) : Dis

(q_3, q_5) : Dis

(q_3, q_6) :

(q_4, q_5) :

(q_4, q_6) : Dis

(q_5, q_6) : Dis

3. Untuk pasangan *state* yang lain jika masing-masing state mendapat input yang sama, maka bila satu state mencapai state akhir dan yang lain tidak mencapai state akhir maka dikatakan distinguishable.

Untuk (q_0, q_4)

$\delta(q_0, 1) = q_2$

$\delta(q_4, 1) = q_4$

$\delta(q_0, 0) = q_1$

$\delta(q_4, 0) = q_5$

Maka (q_0, q_4) : Distinguishable

Untuk (q_0, q_5)

$\delta(q_0, 1) = q_2$

$\delta(q_5, 1) = q_5$

$\delta(q_0, 0) = q_1$

$\delta(q_5, 0) = q_5$

Maka (q_0, q_5) : Distinguishable

Untuk (q_1, q_2)

$\delta(q_1, 1) = q_6$

$\delta\ (q_2, 1) = q_4$

$\delta\ (q_1, 0) = q_3$

$\delta\ (q_2, 0) = q_4$

Maka (q_1, q_2) : Distinguishable

Untuk (q_1, q_3)

$\delta\ (q_1, 1) = q_6$

$\delta\ (q_3, 1) = q_6$

$\delta\ (q_1, 0) = q_3$

$\delta\ (q_3, 0) = q_3$

Maka (q_1, q_3) : InDistinguishable

Untuk (q_1, q_6)

$\delta\ (q_1, 1) = q_6$

$\delta\ (q_6, 1) = q_4$

$\delta\ (q_1, 0) = q_3$

$\delta\ (q_6, 0) = q_4$

Maka (q_1, q_6) : Distinguishable

Untuk (q_2, q_3)

$\delta\ (q_2, 1) = q_4$

$\delta\ (q_3, 1) = q_6$

$\delta\ (q_2, 0) = q_4$

$\delta\ (q_3, 0) = q_3$

Maka (q_2, q_3) : Distinguishable

Untuk (q_2, q_6)

$\delta\ (q_2, 1) = q_4$

$\delta\ (q_6, 1) = q_4$

$\delta\ (q_2, 0) = q_4$

$\delta\ (q_6, 0) = q_4$

Maka (q_2, q_6) : InDistinguishable

Untuk (q_3, q_6)

$\delta\ (q_3, 1) = q_6$

$\delta\ (q_6, 1) = q_4$

$\delta\ (q_3, 0) = q_3$

$\delta\ (q_6, 0) = q_4$

Maka (q_3, q_6) : Distinguishable

Untuk (q_4, q_5)

$\delta\ (q_4, 1) = q_4$

$\delta\ (q_5, 1) = q_5$

$\delta\ (q_4, 0) = q_5$

$\delta\ (q_5, 0) = q_5$

Maka (q_4, q_5) : InDistinguishable

4. Maka Didapatkan pasangan *state* sebagai berikut.

   (q_0, q_1) : Dis

   (q_0, q_2) : Dis

   (q_0, q_3) : Dis

   (q_0, q_4) : Dis

   (q_0, q_5) : Dis

   (q_0, q_6) : Dis

   (q_1, q_2) : Dis

   (q_1, q_3) : InDis

   (q_1, q_4) : Dis

   (q_1, q_5) : Dis

   (q_1, q_6) : Dis

   (q_2, q_3) : Dis

   (q_2, q_4) : Dis

   (q_2, q_5) : Dis

   (q_2, q_6) : InDis

   (q_3, q_4) : Dis

   (q_3, q_5) : Dis

   (q_3, q_6) : Dis

   (q_4, q_5) : InDis

   (q_4, q_6) : Dis

   (q_5, q_6) : Dis

5. Kelompokkan pasangan *state* yang indistinguishable

   (q_1, q_3) : InDis

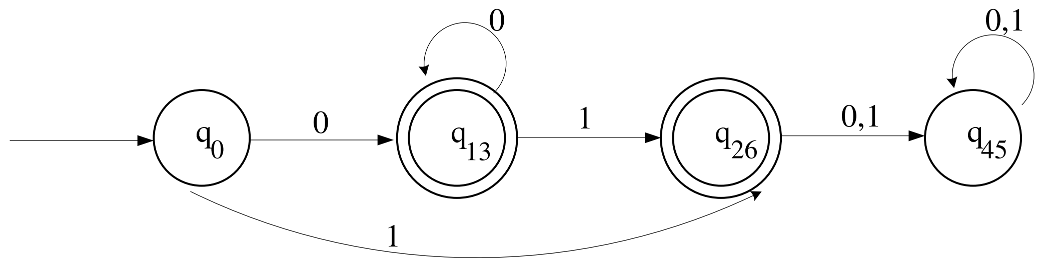
   (q_2, q_6) : InDis

(q_4, q_5) : InDis

6. q_1, q_3 saling *indistinguishable*

   q_2, q_6 saling *indistinguishable*

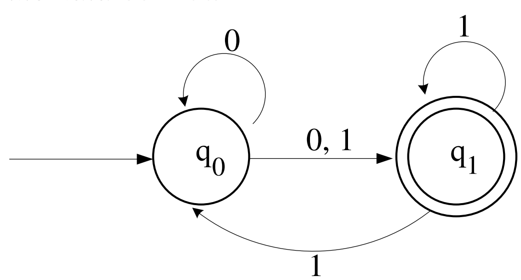
   q_4 dan q_5 juga saling *indistinguishable*.
7. Sehingga diperoleh penyederhanaan sebagai berikut.



Ekuivalensi *Non-Deterministic Finite Automata* ke *Deterministic Finite Automata*

Dari sebuah mesin *Non-Deterministic Finite Automata* dapat dibuat mesin *Deterministic Finite Automata*-nya yang ekuivalen (bersesuaian). Ekuivalen di sini artinya mampu menerima bahasa yang sama.

Sebagai contoh, akan dibuat *Deterministic Finite Automata* dari *Non-Deterministic Finite Automata* berikut.



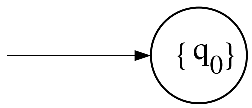
Diketahui $\Sigma = \{0,1\}$

Adapun langkah-langkahnya adalah sebagai berikut.

1. Buatlah tabel transisi dari diagram transisi di atas.

δ	0	1
q_0	$\{q_0,q_1\}$	$\{q_1\}$
q_1	Ø	$\{q_0,q_1\}$

2. Buatlah diagram transisi untuk *finite state automata* dari tabel transisi di atas.
   a. Kita mulai dari state awal yaitu q_0

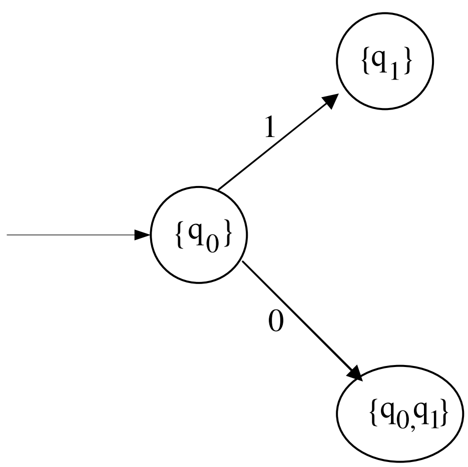


> *Catatan :*
> Perhatikan bahwa di sini pada gambar setiap *state* kita tuliskan sebagai himpunan *state*

   b. Selanjutnya, kita telusuri lebih lanjut tentang q_0, yaitu :

   Bila *state* q_0 mendapat input 0 menjadi state $\{q_0,q_1\}$

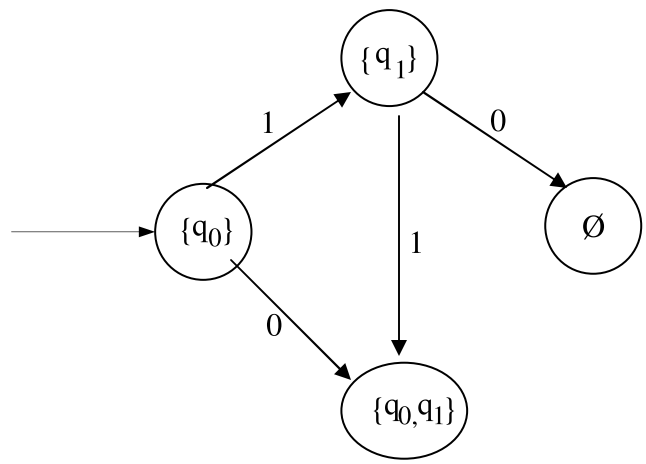
   Bila *state* q_0 mendapat input 1 menjadi state $\{q_1\}$, seperti yang tampak pada gbr.



c. Selanjutnya kita telusuri untuk *state* q_1, yaitu :

Bila *state* q_1 mendapat *input* 0 maka menjadi *state* Ø

Bila *state* q_1 mendapat *input* 1 maka menjadi *state* $\{q_0,q_1\}$, sehingga diperoleh gbr.



d. Selanjutnya kita telusuri untuk *state* $\{q_0,q_1\}$, yang merupakan penggabungan dari *state* q_0 dan *state* q_1, sehingga hasil *state* $\{q_0,q_1\}$ merupakan penggabungan dari hasil *state* q_0 dan *state* q_1.

Bila *state* q_0 mendapat input 0 menjadi state $\{q_0,q_1\}$

Bila *state* q_1 mendapat *input* 0 maka menjadi *state* Ø

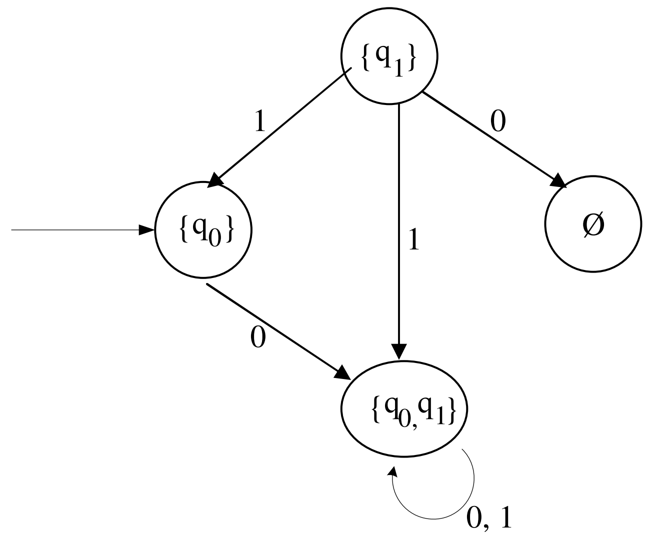
Sehingga diperoleh jika *state* $\{q_0,q_1\}$ mendapat input 0 menjadi *state* $\{q_0,q_1\}$

Bila *state* q_0 mendapat input 1 menjadi state $\{q_1\}$

Bila *state* q_1 mendapat *input* 1 maka menjadi *state* $\{q_0,q_1\}$

Sehingga diperoleh jika *state* $\{q_0,q_1\}$ mendapat input 0 menjadi *state* $\{q_0,q_1\}$

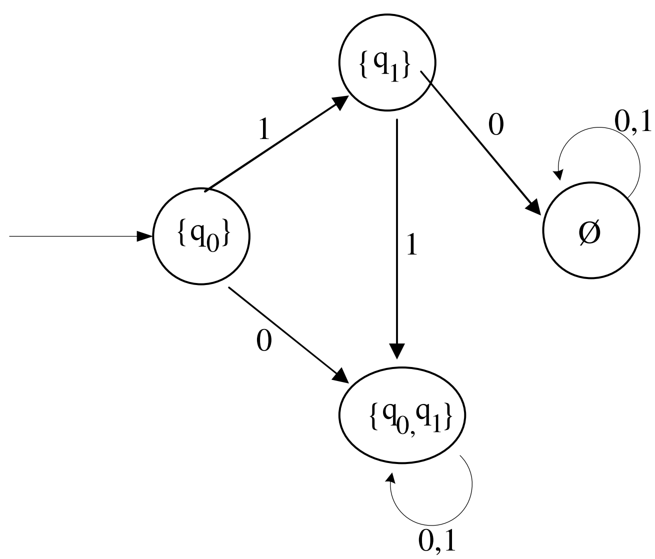
Maka diagram transisi menjadi :



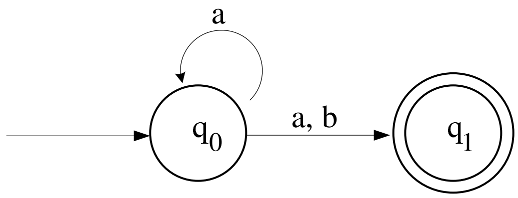
e. Selanjutnya kita telusuri *state* Ø, yaitu :

Bila *state* Ø mendapat input 0 dan 1 maka tetap menghasilkan Ø

Sehingga diperoleh diagram transisi berikut.



Contoh lain, buatlah DFA dari NFA berikut :

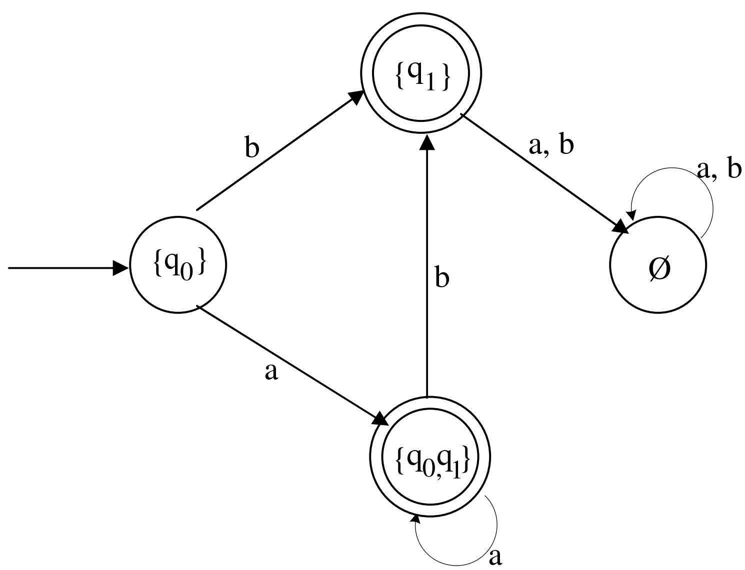


Diketahui $\Sigma = \{a,b\}$

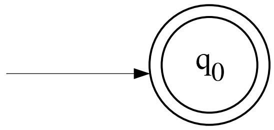
Tabel Transisi untuk NFA pada gambar di atas adalah sebagai berikut.

δ	a	b
q_0	$\{q_0,q_1\}$	$\{q_1\}$
q_1	Ø	Ø

Mesin *Deterministic Finite Automata* yang ekuivalen adalah sebagai berikut.



Buatlah DFA dari NFA berikut.

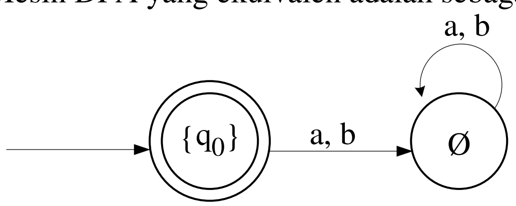


Diketahui Σ = {a,b}

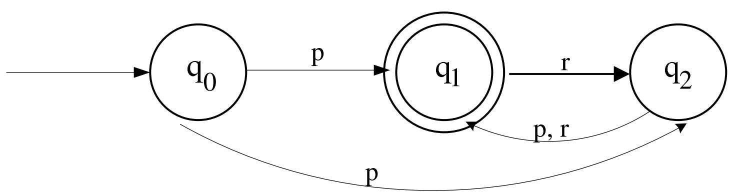
Tabel transisi untuk NFA pada gambar di atas adalah sebagai berikut.

δ	a	b
q_0	Ø	Ø

Mesin DFA yang ekuivalen adalah sebagai berikut.



Buatlah DFA dari NFA berikut.

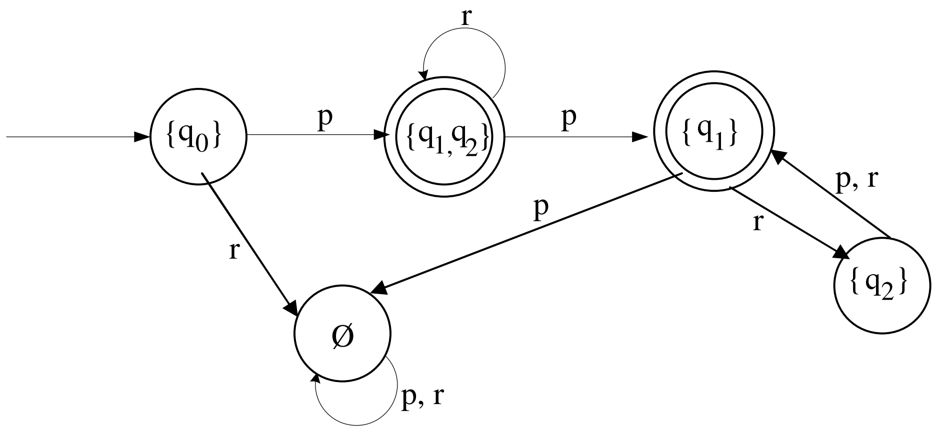


Diketahui Σ = {p,r}

Tabel transisinya adlaah sebagai berikut.

δ	p	R
q_0	{q_1,q_2}	Ø
q_1	Ø	{q_2}
q_2	{q_1}	{q_1}

Mesin DFA dari NFA berikut adalah sebagai berikut.



Soal :

1. Buatlah *Deterministic Finite Automata* yang ekuivalen dengan *Non Deterministic Finite Automata* berikut.

   Q = {p, q, r, s}

   Σ = {0, 1}

   S = p

   F = {s}

   Fungsi transisinya dinyatakan dalam tabel transisi berikut.

δ	0	1
p	{p, q}	{p}
q	{r}	{r}
r	{s}	-
s	s	s

2. Buatlah *Deterministic Finite Automata* yang ekuivalen dengan *Non-Determinitic Finite Automata* berikut.

   Q = {p, q, r, s}

   Σ = {0, 1}

   S = p

F = {q, s}

Fungsi transisinya dinyatakan dalam tabel transisi berikut.

δ	0	1
p	{q, s}	{q}
q	{r}	{q, r}
r	{s}	{p}
s	-	{p}

3. Buatlah *Deterministic Finite Automata* yang ekuivalen dengan *Non Deterministic Finite Automata* berikut.

Q = {q_0, q_1, q_2}

Σ = {0, 1}

S = q_0

F = { q_1}

Fungsi transisinya dinyatakan dalam tabel transisi berikut.

δ	0	1
q_0	{q_0}	{ q_2}
q_1	{q_1}	Ø
q_2	{ q_0, q_1}	{ q_1}

4. Buatlah *Deterministic Finite Automata* yang ekuivalen dengan *Non-Determnistic Finite Automata* berikut.

Q = {q_0, q_1, q_2}

Σ = {a, b}

S = q_0

F = { q_1}

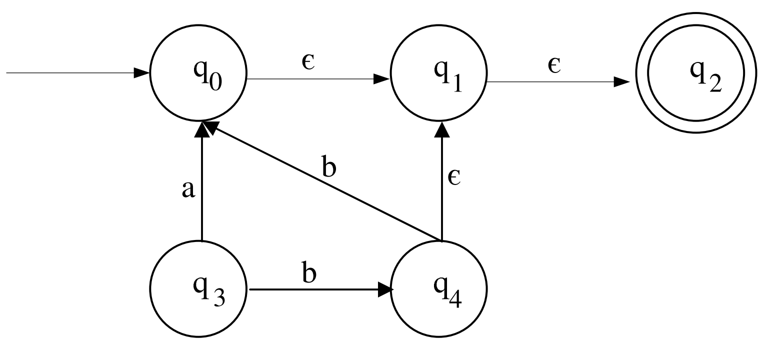
Fungsi transisinya dinyatakan dalam tabel transisi berikut.

δ	a	b
q_0	$\{q_1, q_2\}$	$\{ q_2\}$
q_1	$\{q_1\}$	$\{ q_2\}$
q_2	Ø	$\{ q_0, q_2\}$

<u>Non Deterministic Finite Automata dengan ϵ – Move</u>

Di sini kita mempunyai jenis otomata baru yang disebut *Non Deterministic Finite Automata* dengan ϵ – Move (ϵ di sini bisa dianggap sebagai '*empty*'). Pada *Non – deterministic Finite Automata* dengan ϵ – move (transisi ϵ), diperbolehkan mengubah *state* tanpa membaca *input*. Disebut dengan transisi ϵ karena tidak bergantung pada suatu *input* ketika melakukan transisi.

Contoh :



Penjelasan gambar :

- Dari q_0 tanpa membaca *input* dapat berpindah ke q_1
- Dari q_1 tanpa membaca *input* dapat berpindah ke q_2
- Dari q_4 tanpa membaca *input* dapat berpindah ke q_1

Є – Closure untuk Suatu *Non-Deterministic Finite Automata* dengan Є – Move

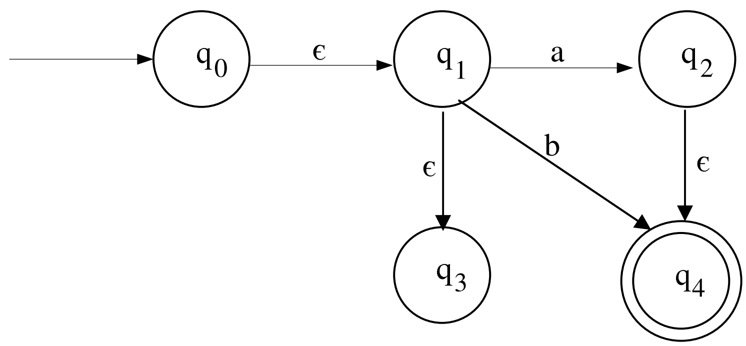
Є – Closure adalah himpunan *state-state* yang dapat dicapai dari suatu *state* tanpa membaca *input*. Perhatikan gambar sebelumnya, maka diperoleh :

Є – Closure (q_1) = { q_1, q_2 }

Є – Closure (q_2) = { q_2 }

Є – Closure (q_3) = { q_3 }

Contoh lain, dapat dilihat pada gambar di bawah ini.



Dari gambar di atas, kita ketahui Є – Closure untuk setiap *state* adalah sebagai berikut.

Є – Closure (q_0) = { q_0, q_1, q_3 }

Є – Closure (q_1) = { q_1, q_3 }

Є – Closure (q_2) = { q_2, q_4 }

Є – Closure (q_3) = { q_3 }

Є – Closure (q_4) = { q_4 }

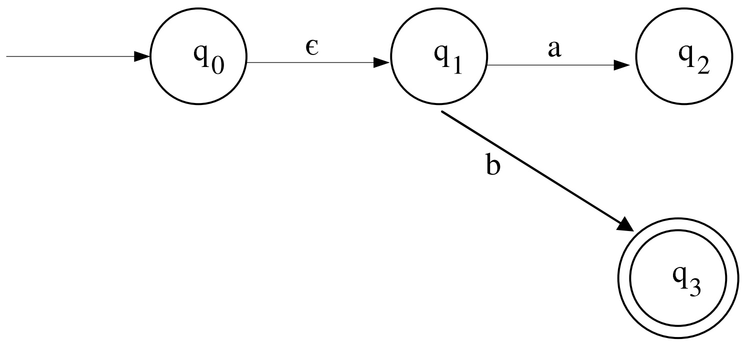
Catatan :

Perhatikan bahwa pada suatu *state* yang tidak memiliki transisi ϵ , maka ϵ – closure – nya adalah *state* itu sendiri

<u>Ekuivalensi Non – Deterministic Finite Automata dengan Є – Move ke Non-Deterministic Finite Automata tanpa Є-Move</u>

Dari sebuah *Non-Deterministic Finite Automata* dengan ϵ – move dapat kita peroleh *Non – Deterministic Finite Automata* tanpa ϵ – move yang ekuivalen.

Contohnya, bila kita punya NFA ϵ – move, seperti pada gambar di bawah ini.



Dari NFA ϵ – move di atas, akan dibuat NFA yang ekuivalen

1. Buatlah tabel transisi dari NFA ϵ – move di atas.

δ	a	b
q_0	Ø	Ø
q_1	$\{q_2\}$	$\{q_3\}$
q_2	Ø	Ø
q_3	Ø	Ø

2. Tentukan ϵ-closure untuk setiap *state*

   Є – Closure (q_0) = { q_0, q_1 }

   Є – Closure (q_1) = { q_1 }

   Є – Closure (q_2) = { q_2 }

   Є – Closure (q_3) = { q_3 }

3. Carilah setiap fungsi transisi hasil dari pengubahan NFA ϵ – move ke NFA tanpa ϵ – move. Fungsi transisi itu ditandai dengan simbol δ'

$\delta'(q_0, a) = \epsilon\_cl(\delta(\epsilon\_cl(q_0), a))$
$= \epsilon\_cl(q_2)$
$= \{ q_2 \}$
$\delta'(q_0, b) = \epsilon\_cl(\delta(\epsilon\_cl(q_0), b))$
$= \epsilon\_cl(q_3)$
$= \{ q_3 \}$

$\delta'(q_1, a) = \epsilon\_cl(\delta(\epsilon\_cl(q_1), a))$
$= \epsilon\_cl(q_2)$
$= \{ q_2 \}$
$\delta'(q_1, b) = \epsilon\_cl(\delta(\epsilon\_cl(q_1), b))$
$= \epsilon\_cl(q_2)$
$= \{ q_3 \}$

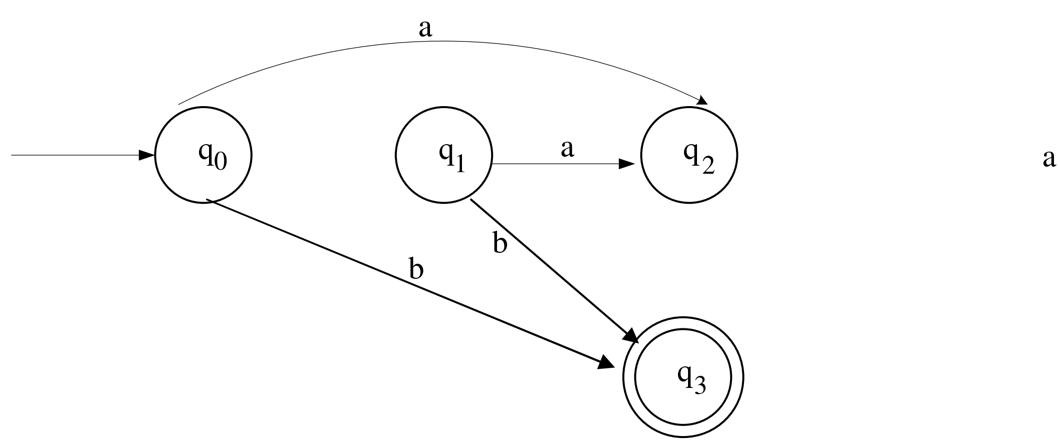
$\delta'(q_2, a) = \epsilon\_cl(\delta(\epsilon\_cl(q_2), a))$
$= \epsilon\_cl(\emptyset)$
$= \emptyset$
$\delta'(q_2, b) = \epsilon\_cl(\delta(\epsilon\_cl(q_2), b))$
$= \epsilon\_cl(\emptyset)$
$= \emptyset$

$\delta'(q_3, a) = \epsilon\_cl(\delta(\epsilon\_cl(q_3), a))$
$= \epsilon\_cl(\emptyset)$
$= \emptyset$
$\delta'(q_3, b) = \epsilon\_cl(\delta(\epsilon\_cl(q_3), b))$
$= \epsilon\_cl(\emptyset)$
$= \emptyset$

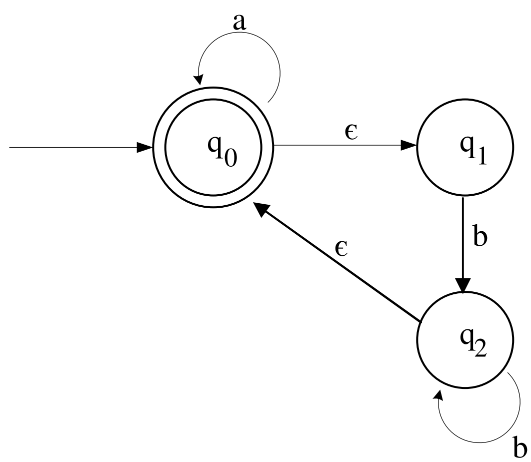
4. Buatlah tabel transisi dari fungsi transisi yang telah dibuat pada langkah sebelumnya.

δ	a	b
q_0	$\{q_2\}$	$\{q_3\}$
q_1	$\{q_2\}$	$\{q_3\}$
q_2	Ø	Ø
q_3	Ø	Ø

5. Kemudian, tentukanlah himpunan *state* akhir untuk NFA tanpa ϵ – move ini. Himpunan *state* akhir semula adalah $\{q_3\}$. Karena tidak ada *state* lain yang ϵ – closure – nya memuat q_3, maka himpunan *state* akhir sekarang tetap $\{q_3\}$. Sehingga diperoleh diagram transisi sebagai berikut.



Contoh lain :



1. Buatlah tabel transisi dari NFA ϵ – move di atas.

δ	a	b
q_0	{ q_0 }	Ø
q_1	Ø	{q_2}
q_2	Ø	{q_2}

2. Tentukan ϵ-closure untuk setiap *state*

   Є – Closure (q_0) = { q_0, q_1 }

   Є – Closure (q_1) = { q_1 }

   Є – Closure (q_2) = { q_0, q_1, q_2 }

3. Carilah setiap fungsi transisi hasil dari pengubahan NFA ϵ – move ke NFA tanpa ϵ – move. Fungsi transisi itu ditandai dengan simbol δ'

   δ' (q_0 , a) = ϵ_cl (δ (ϵ_cl(q_0),a))

   = ϵ_cl (q_0)

   = { q_0, q_1}

   δ' (q_0 , b) = ϵ_cl (δ (ϵ_cl(q_0),b))

   = ϵ_cl (q_2)

   = { q_0, q_1 , q_2}

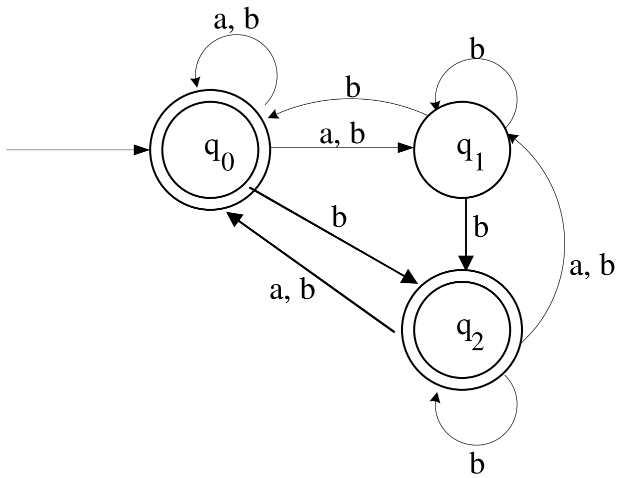
$\delta'(q_1, a) = \epsilon\_cl(\delta(\epsilon\_cl(q_1),a))$
$= \epsilon\_cl(\emptyset)$
$= \emptyset$
$\delta'(q_1, b) = \epsilon\_cl(\delta(\epsilon\_cl(q_1),b))$
$= \epsilon\_cl(q_2)$
$= \{q_0, q_1, q_2\}$

$\delta'(q_2, a) = \epsilon\_cl(\delta(\epsilon\_cl(q_2),a))$
$= \epsilon\_cl(q_0)$
$= \{q_0, q_1\}$
$\delta'(q_2, b) = \epsilon\_cl(\delta(\epsilon\_cl(q_2),b))$
$= \epsilon\_cl(q_2)$
$= \{q_0, q_1, q_2\}$

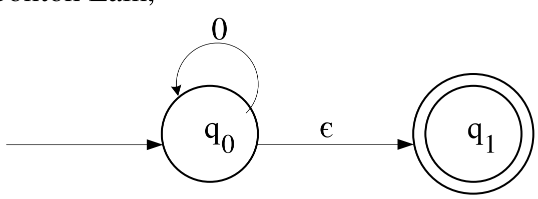
4. Buatlah tabel transisi dari fungsi transisi yang telah dibuat pada langkah sebelumnya.

δ	a	b
q_0	$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$
q_1	$\emptyset$	$\{q_0, q_1, q_2\}$
q_2	$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$

5. Kemudian, tentukanlah himpunan *state* akhir untuk NFA tanpa ϵ – move ini. Himpunan *state* akhir semula adalah $\{q_0\}$. Kita lihat $\epsilon\_cl(q_2) = \{q_0, q_1, q_2\}$, maka himpunan *state* akhir sekarang adalah $\{q_0, q_2\}$. Sehingga diperoleh diagram transisi sebagai berikut.



Contoh Lain,



$\Sigma = \{0\}$

1. Buatlah tabel transisi dari NFA ϵ – move di atas.

δ	0
q_0	$\{ q_0 \}$
q_1	Ø

2. Tentukan ϵ-closure untuk setiap *state*

   Є – Closure (q_0) = { q_0, q_1 }

   Є – Closure (q_1) = { q_1 }

3. Carilah setiap fungsi transisi hasil dari pengubahan NFA ϵ – move ke NFA tanpa ϵ – move. Fungsi transisi itu ditandai dengan simbol δ'

   $\delta'(q_0 , 0)$ = ϵ_cl (δ (ϵ_cl(q_0),0))

   = ϵ_cl (q_0)

$= \{ q_0, q_1 \}$

$\delta'(q_1, 0) = \epsilon\_cl(\delta(\epsilon\_cl(q_1), 0))$

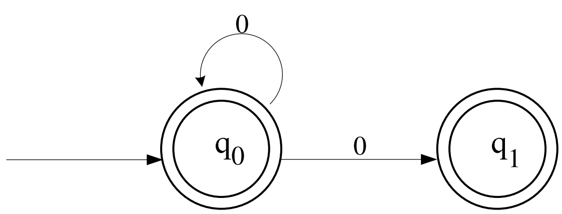
$= \epsilon\_cl(\varnothing)$

$= \varnothing$

4. Buatlah tabel transisi dari fungsi transisi yang telah dibuat pada langkah sebelumnya.

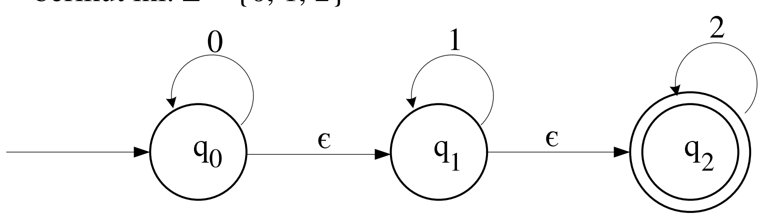
δ	0
q_0	$\{ q_0, q_1 \}$
q_1	$\varnothing$

5. Kemudian, tentukanlah himpunan *state* akhir untuk NFA tanpa ϵ – move ini. Himpunan *state* akhir semula adalah $\{q_1\}$. Kita lihat $\epsilon\_cl(q_0) = \{ q_0, q_1 \}$, maka himpunan *state* akhir sekarang adalah $\{q_0, q_1\}$. Sehingga diperoleh diagram transisi sebagai berikut.

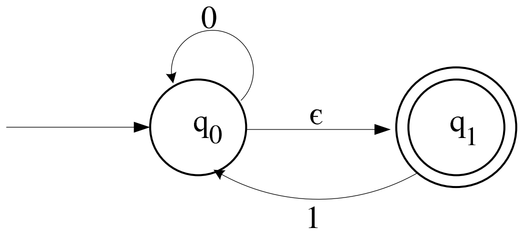


Soal :

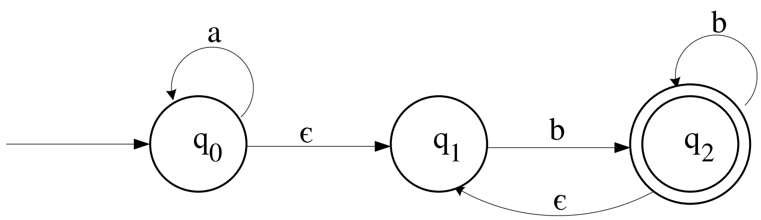
1. Buatlah NFA tanpa ϵ – move yang ekuivalen dengan NFA ϵ – Move pada gambar berikut ini. $\Sigma = \{0, 1, 2\}$



2. Buatlah NFA tanpa ϵ – Move yang ekuivalen dengan NFA ϵ – Move pada gambar di bawah ini. Σ = {0, 1}



3. Buatlah NFA tanpa ϵ – Move yang ekuivalen dengan NFA ϵ – Move pada gambar di bawah ini. Σ = {a, b}



Jawab :

1.

Buatlah tabel transisi dari NFA ϵ – move di atas.

δ	0	1	2
q_0	$\{q_0\}$	Ø	Ø
q_1	Ø	$\{q_1\}$	Ø
q_2	Ø	Ø	$\{q_2\}$

Tentukan ϵ-closure untuk setiap *state*

Є – Closure (q_0) = { q_0, q_1, q_2}

Є – Closure (q_1) = { q_1, q_2}

Є – Closure (q_2) = {q_2}

$\delta'(q_0, 0) \quad = \epsilon\_cl(\delta(\epsilon\_cl(q_0), 0))$

$= \epsilon\_cl(q_0)$

$= \{q_0, q_1, q_2\}$

$\delta'(q_0, 1) \quad = \epsilon\_cl(\delta(\epsilon\_cl(q_0), 1))$

$= \epsilon\_cl(q_1)$

$= \{q_1, q_2\}$

$\delta'(q_0, 2) \quad = \epsilon\_cl(\delta(\epsilon\_cl(q_0), 2))$

$= \epsilon\_cl(q_2)$

$= \{q_2\}$

$\delta'(q_1, 0) \quad = \epsilon\_cl(\delta(\epsilon\_cl(q_1), 0))$

$= \epsilon\_cl(\emptyset)$

$= \{\emptyset\}$

$\delta'(q_1, 1) \quad = \epsilon\_cl(\delta(\epsilon\_cl(q_1), 1))$

$= \epsilon\_cl(q_1)$

$= \{q_1, q_2\}$

$\delta'(q_1, 2) \quad = \epsilon\_cl(\delta(\epsilon\_cl(q_1), 2))$

$= \epsilon\_cl(q_2)$

$= \{q_2\}$

$\delta'(q_2, 0) \quad = \epsilon\_cl(\delta(\epsilon\_cl(q_2), 0))$

$= \epsilon\_cl(\emptyset)$

$= \{\emptyset\}$

$\delta'(q_2, 1) \quad = \epsilon\_cl(\delta(\epsilon\_cl(q_2), 1))$

$= \epsilon\_cl(\emptyset)$

$= \{ \varnothing \}$

$\delta' (q_2 , 2)$ $= \epsilon\_cl (\delta (\epsilon\_cl(q_2),2))$

$= \epsilon\_cl (q_2)$

$= \{q_2\}$

δ	0	1	2
q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	$\varnothing$	$\{q_1, q_2\}$	$\{q_2\}$
q_2	$\varnothing$	$\varnothing$	$\{q_2\}$

Himpunan *state* akhir adalah $\{q_0, q_1, q_2\}$

2.

Buatlah tabel transisi dari NFA ϵ – move di atas.

δ	0	1
q_0	$\{q_0\}$	$\varnothing$
q_1	$\varnothing$	$\{q_0\}$

Tentukan ϵ-closure untuk setiap *state*

Є – Closure (q_0) = { q_0, q_1 }

Є – Closure (q_1) = { q_1 }

$\delta' (q_0 , 0)$ $= \epsilon\_cl (\delta (\epsilon\_cl(q_0),0))$

$= \epsilon\_cl (q_0)$

$= \{ q_0, q_1\}$

$\delta' (q_0 , 1)$ $= \epsilon\_cl (\delta (\epsilon\_cl(q_0),1))$

$= \epsilon\_cl (q_0)$

$= \{ q_0, q_1 \}$

$\delta'(q_1, 0) = \epsilon\_cl(\delta(\epsilon\_cl(q_1),0))$

$= \epsilon\_cl(\emptyset)$

$= \{ \emptyset \}$

$\delta'(q_1, 1) = \epsilon\_cl(\delta(\epsilon\_cl(q_1),1))$

$= \epsilon\_cl(q_0)$

$= \{ q_0, q_1 \}$

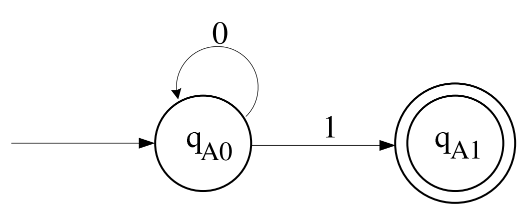
δ	0	1
q_0	$\{ q_0, q_1 \}$	$\{ q_0, q_1 \}$
q_1	$\emptyset$	$\{ q_0, q_1 \}$

<u>Penggabungan dan Konkatenasi Finite State Automata</u>

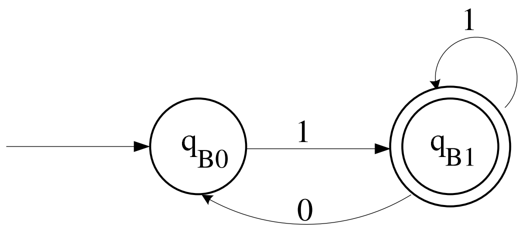
A. Penggabungan Finite State Automata

Pada dua mesin *Finite State Automata*, misalkan M_1 dan M_2 dapat dilakukan penggabungan yang menghasilkan mesin M3 dengan cara :

1. Tambahkan *state* awal untuk M_3, hubungkan dengan *state* awal M_1 dan *state* awal M_2 menggunakan transisi ϵ.
2. Tambahkan *state* akhir untuk M_3, hubungkan dengan *state-state* akhir M_1 dan *state-state* akhir M_2 menggunakan transisi ϵ.

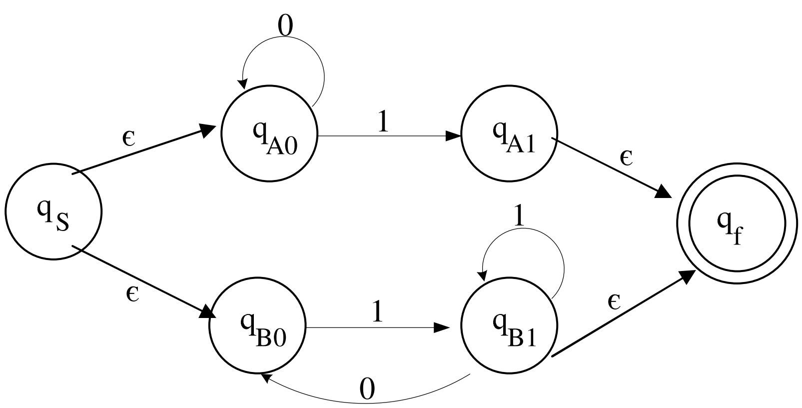


Gambar Mesin M_1



Gambar Mesin M_2

Adapun hasil penggabungan dari Mesin M_1 dan M_2 dapat dilihat pada gambar di bawah ini.

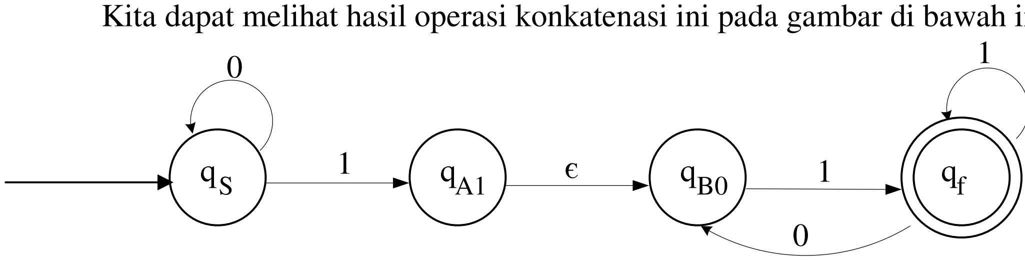


B. Konkatenasi *Finite State Automata*

Pada dua mesin *Finite State Automata*, misalkan M_1 dan M_2 dapat dilakukan konkatenasi yang menghasilkan mesin M_4 dengan cara :

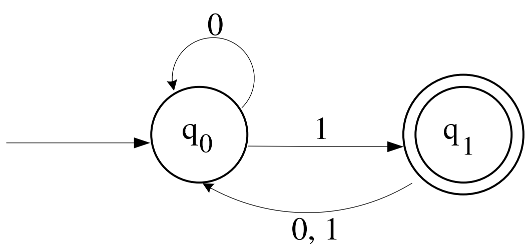
1. *State* awal M_1 menjadi *state* awal M_4
2. *State-state* akhir M_2 menjadi *state* akhir M_4
3. Hubungkan *state-state* akhir M_1 dengan *state* awal M_2 menggunakan transisi ϵ.

Kita dapat melihat hasil operasi konkatenasi ini pada gambar di bawah ini.

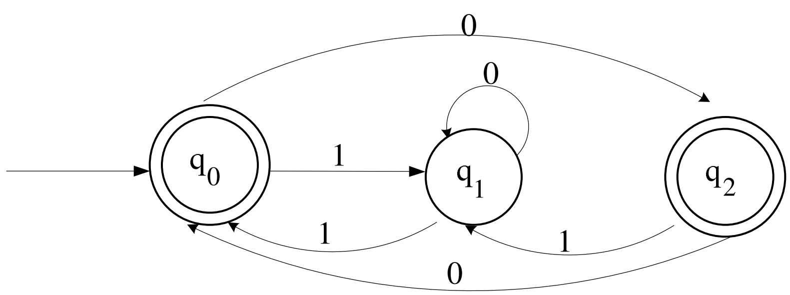


Soal :

1. Bila diketahui L (M_1) adalah bahasa yang diterima oleh M_1 pada gambar 1, dan L(M_2) adalah bahasa yang diterima oleh M_2 pada gambar 2. Diketahui L(M_3) = L(M_1) + L(M_2), serta L(M_4) = L(M_1) L(M_2). Gambarkan :
   a. Mesin M_3 yang menerima bahasa L(M_3).
   b. Mesin M_4 yang menerima bahasa L(M_4).



Mesin M_1

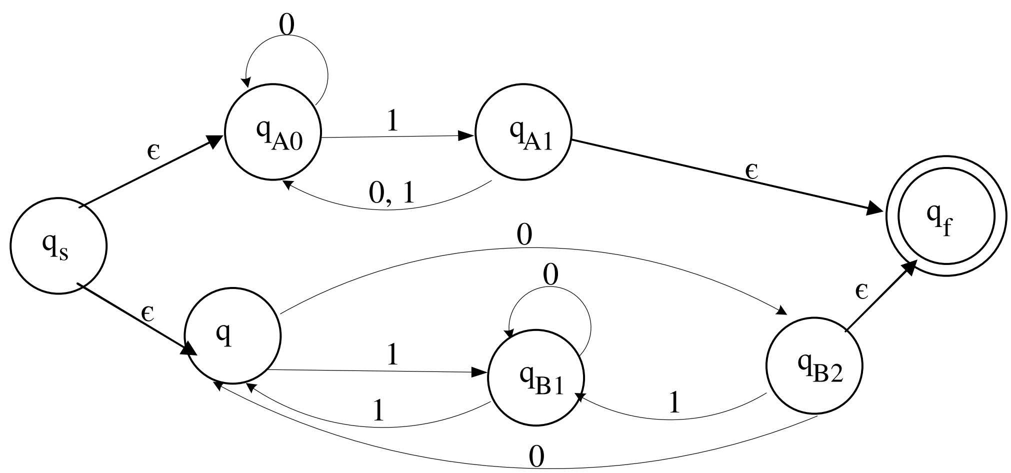


Mesin M_2

Jawab :

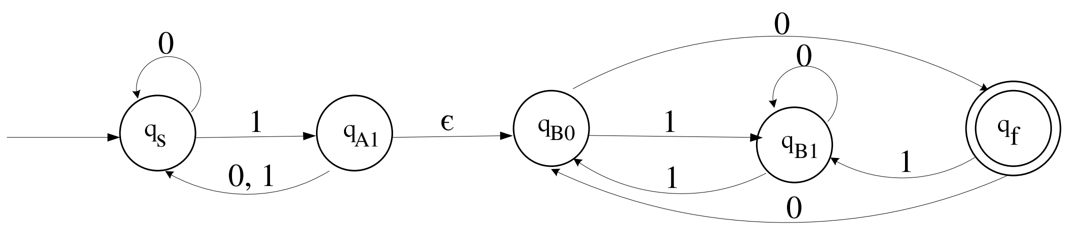
a.

1. Tambahkan *state* awal untuk M_3, hubungkan dengan *state* awal M_1 dan *state* awal M_2 menggunakan transisi ϵ.
2. Tambahkan *state* akhir untuk M_3, hubungkan dengan *state-state* akhir M_1 dan *state-state* akhir M_2 menggunakan transisi ϵ.

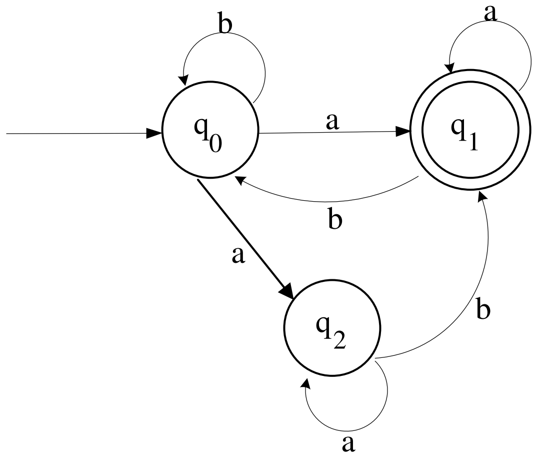


b.

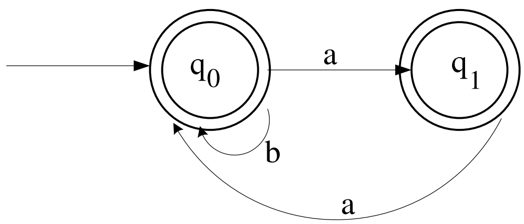
1. *State* awal M_1 menjadi *state* awal M_4
2. *State-state* akhir M_2 menjadi *state* akhir M_4
3. Hubungkan *state-state* akhir M_1 dengan *state* awal M_2 menggunakan transisi ϵ.



2. Bila diketahui L (M_1) adalah bahasa yang diterima oleh M_1 pada gambar 1, dan $L(M_2)$ adalah bahasa yang diterima oleh M_2 pada gambar 2. Diketahui $L(M_3) = L(M_1) + L(M_2)$, serta $L(M_4) = L(M_1)\ \ L(M_2)$. Gambarkan :
   a. Mesin M_3 yang menerima bahasa $L(M_3)$.
   b. Mesin M_4 yang menerima bahasa $L(M_4)$.



Mesin M_1

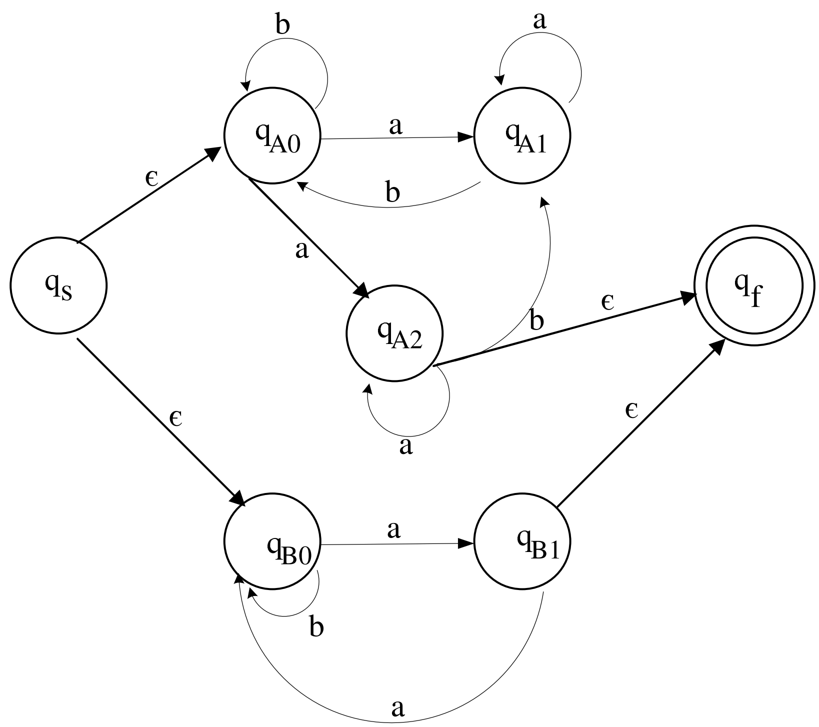


Mesin M_2

Jawab :

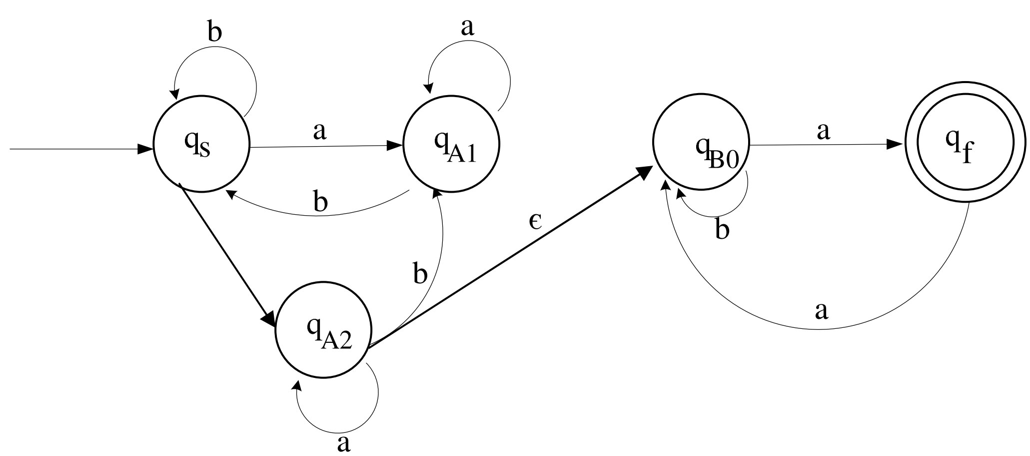
a.

1. Tambahkan *state* awal untuk M_3, hubungkan dengan *state* awal M_1 dan *state* awal M_2 menggunakan transisi ϵ.
2. Tambahkan *state* akhir untuk M_3, hubungkan dengan *state-state* akhir M_1 dan *state-state* akhir M_2 menggunakan transisi ϵ.



b.

1. *State* awal M_1 menjadi *state* awal M_4
2. *State-state* akhir M_2 menjadi *state* akhir M_4
3. Hubungkan *state-state* akhir M_1 dengan *state* awal M_2 menggunakan transisi ϵ.



Ekspresi Regular

Sebuah bahasa dinyatakan regular jika terdapat *finite state automata* yang dapat menerimanya. Bahasa-bahasa yang diterima oleh suatu *finite state automata* bisa dinyatakan secara sederhana dengan ekspresi regular.

Contoh pemakaian ekspresi regular adalah pada perancangan suatu *text editor*.

Notasi Ekspresi Regular

Notasi Ekspresi Regular yang sering dipakai adalah sebagai berikut.

1. * yaitu karakter asterisk, yang berarti bisa tidak muncul, bisa juga muncul lebih dari satu kali.
2. $^{+}$ yaitu minimal muncul satu kali
3. + atau $\cup$ berarti *union*
4. **.** (Titik) berarti konkatenasi, biasanya titik bisa dihilangkan. Misalnya : ab bermakna sama seperti a.b.

Contoh ekspresi regular (selanjutnya kita singkat sebagai ER) adalah sebagai berikut.

- ER : $ab^* cc$

  Contoh *string* yang dibangkitkan : abcc, abbcc, abbbcc, abbbbcc, acc (b bisa tidak muncul atau muncul sejumlah berhingga kali).
- ER : 010^*

  Contoh string yang dibangkitkan : 01, 010, 0100, 01000

  (jumlah 0 diujung bisa tidak muncul, bisa muncul berhingga kali).
- ER : a^*d

  Contoh string yang dibangkitkan : d, ad, aad, aaad
- ER : a^+d

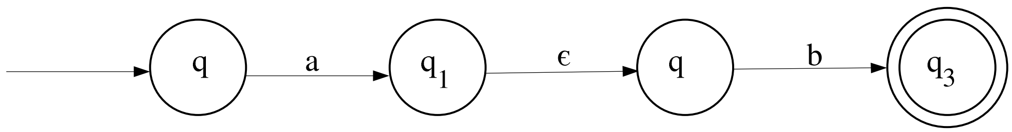
  Contoh string yang dibangkitkan : ad, aad, aaad
- ER : $a^* \cup b^*$ (ingat $\cup$ berarti atau)

  Contoh string yang dibangkitkan : a, b, aa, bb, aaa, bbb, aaaa, bbbb
- ER : $a \cup b$

  Contoh string yang dibangkitkan : a, b
- ER : $01^* + 0$

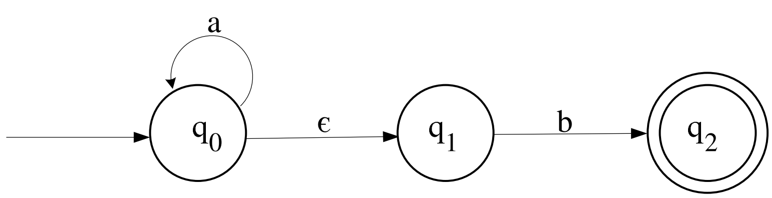
Contoh string yang dibangkitkan : 0, 01, 011, 0111, 01111

<u>Hubungan Ekspresi Regular dan Finite State Automata</u>

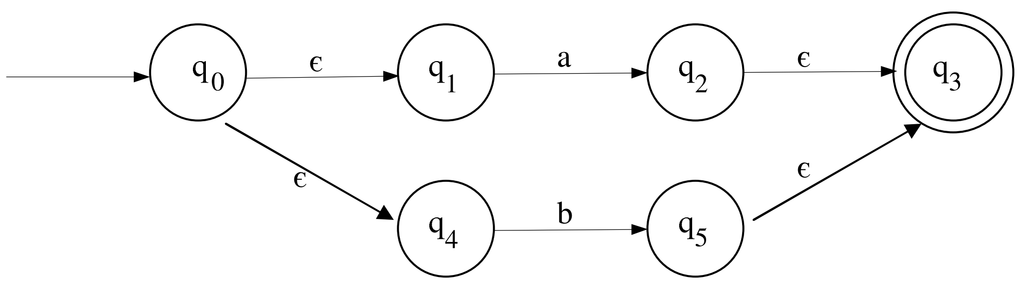


NFA ϵ – move untuk ER : ab

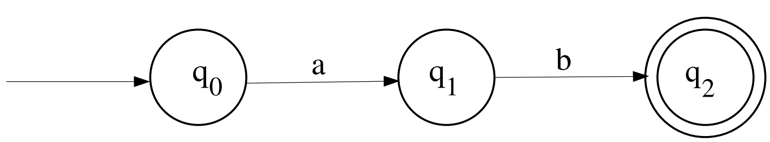
0



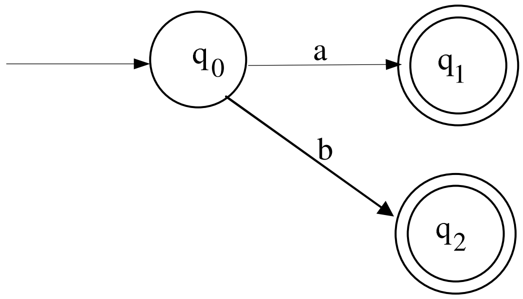
NFA ϵ – move untuk ER : a^*b



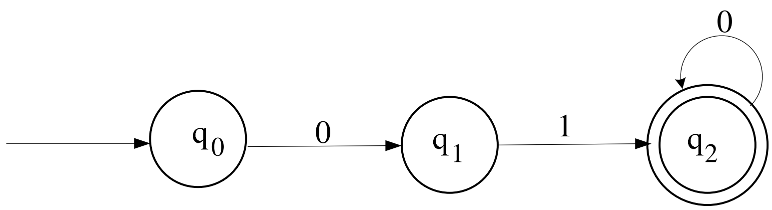
NFA ϵ – move untuk ER : $a \cup b$



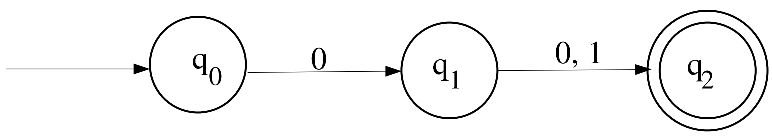
NFA untuk ER : ab



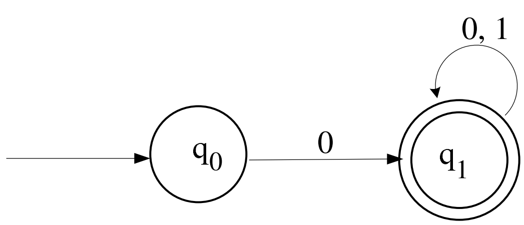
NFA untuk ER : a $\cup$ b



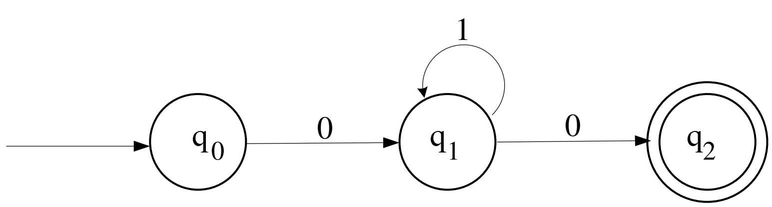
NFA untuk ER : 010^*



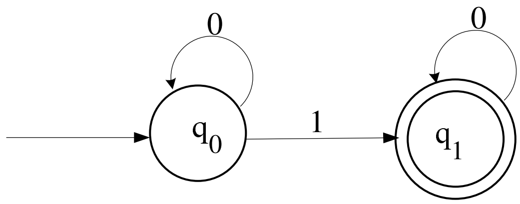
NFA untuk ER : 0 (1 $\cup$ 0)



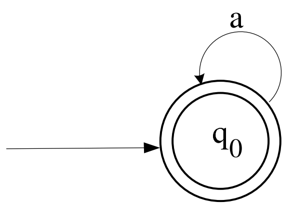
NFA untuk ER : 0 (1 $\cup$ 0)*



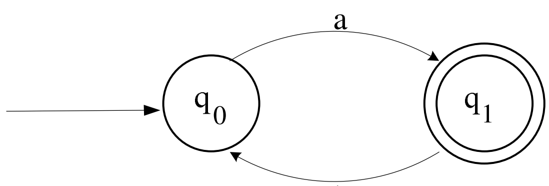
NFA untuk ER : 01^*0



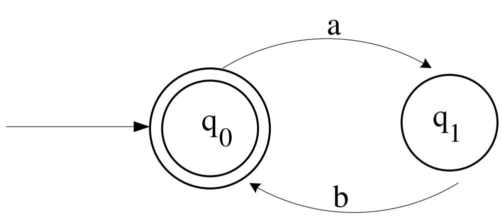
NFA untuk ER : 0^*10^*



NFA untuk ER : a^*

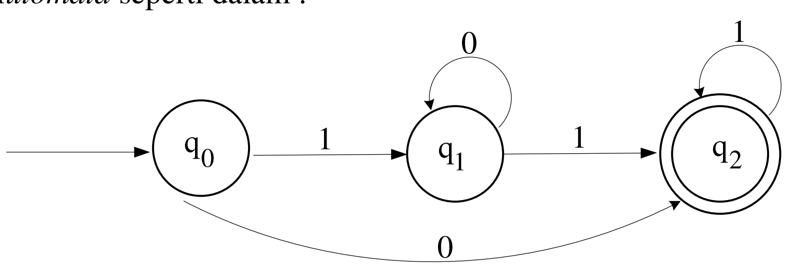


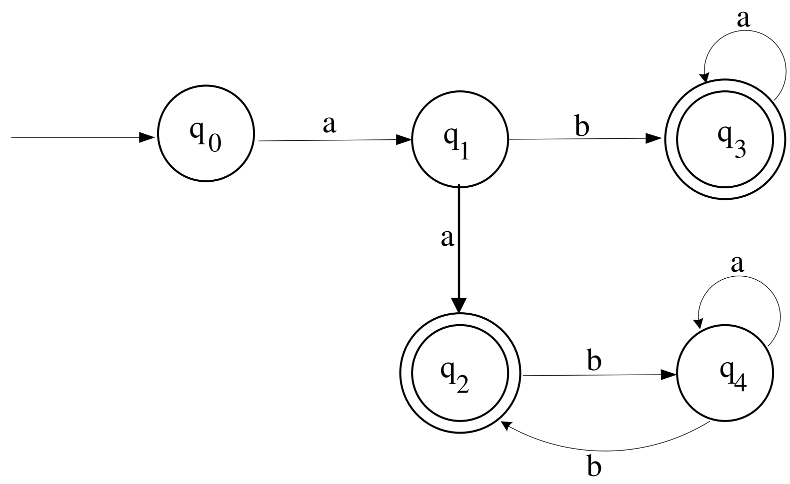
NFA untuk ER : $a\ (ba)^*$

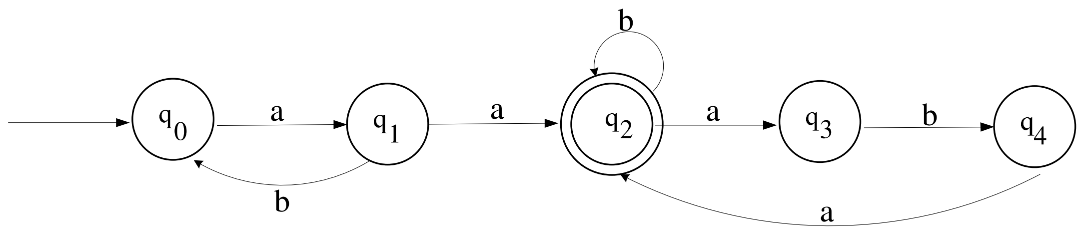


NFA untuk ER : $(ab)^*$

Deskripsikan dalam bahasa Indonesia himpunan string yang diterima oleh *Finite State Automata* seperti dalam :







Jawab :

$01^{*} \cup 10^{*}11^{*}$

$aba^{*} \cup aa(ba^{*}b)^{*}$

$a(ba)^{*}ab^{*}(abab^{*})^{*}$

Aturan Produksi untuk Suatu Tata Bahasa Regular

Batasan aturan produksi untuk bahasa regular :

$\alpha \rightarrow \beta$

Suatu tata bahasa (*grammar*) didefinisikan dengan 4 Tupel yaitu : V, T, P, dan S

Di mana,

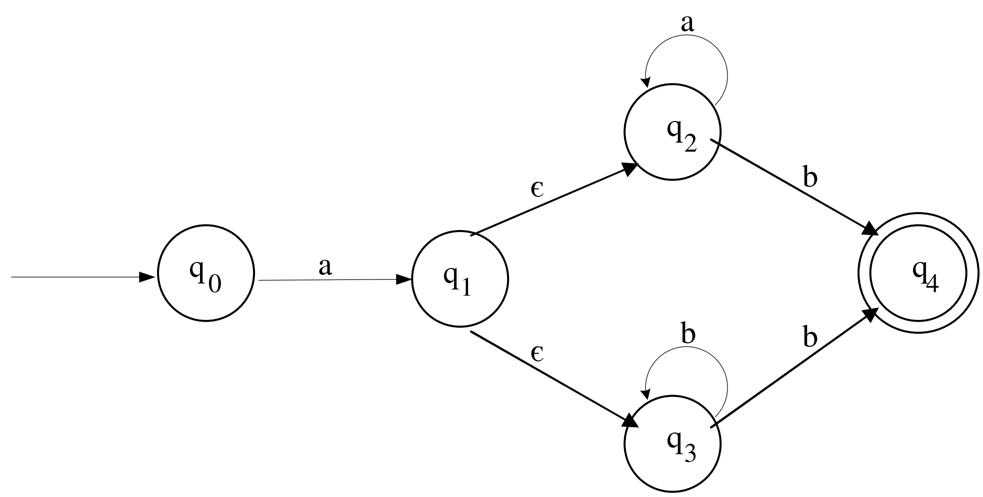
V = Himpunan simbol variabel / non terminal

T = Himpunan simbol terminal

P = Kumpulan aturan produksi

S = Simbol awal

Sebagai contoh terdapat Mesin FSA berikut



Mesin *finite state automata* pada gambar di atas memiliki simbol input ‘a’ dan ‘b’. Simbol ‘a’ dan ‘b’ akan menjadi simbol terminal pada aturan produksi yang akan kita bentuk.

Misalnya kita tentukan simbol awal adalah S. Kita identikkan S dengan *state* awal q_0 . Dari q_0 mendapat *input* a menjadi q_1. Bisa kita tuliskan sebagai aturan produksi :

$S \rightarrow aE$

Di sini kita gunakan sebagai E dan bukan A karena menyatakan bagian yang belum terbangkitkan mulai dari state q_1.

Dari q_1 mendapat transisi – ϵ (tanpa menerima *input*) ke q_2 dan q_3 . Bisa kita tuliskan :

$E \rightarrow A$

$E \rightarrow B$

(Di sini kita identikkan q_2 sebagai A dan q_3 sebagai B)

Dari q_2 jika mendapat input a menuju ke *state* q_2 itu sendiri dan jika mendapat *input* b menuju ke *state* q_4 yang merupakan *state* akhir dan tidak menuju ke *state* yang lainnya sehingga dapat dituliskan menjadi :

$A \rightarrow aA$

$A \rightarrow b$

Dari q_3 jika mendapat input b menuju ke *state* q_3 itu sendiri dan jika mendapat *input* b juga menuju ke *state* q_4 yang merupakan *state* akhir dan tidak menuju ke *state* yang lainnya sehingga dapat dituliskan menjadi :

B → bB

B → b

Kumpulan aturan produksi yang kita peroleh bisa kita tuliskan sebagai berikut.

S → aE

E → A | B

A → aA | b

B → bB | b

Secara formal tata bahasa yang diperoleh dari otomata adalah sebagai berikut.

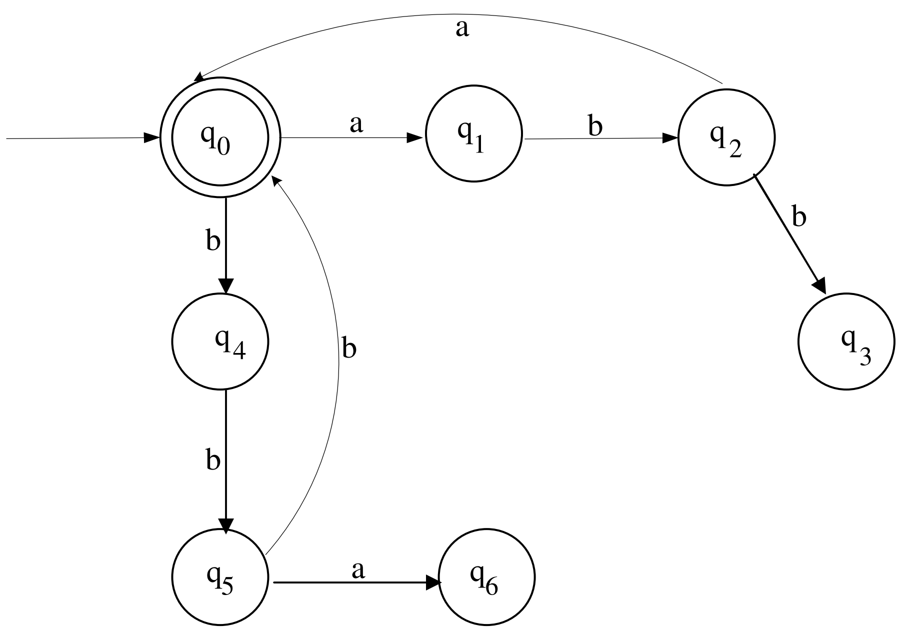
V = {S, E, A, B}

T = {a, b}

P = { S → aE, E → A | B, A → aA | b, B → bB | b }

S = S

Contoh lain dapat dilihat pada gambar di bawah ini.



Kita bisa mengkonstruksi aturan produksi untuk otomata tersebut.

T = {a, b}

S = S

Kita mulai dari *state* awal yaitu q_0 yang dalam hal ini dilambangkan dengan S.

- Bila S mendapat input a maka menuju ke q_1 yang dalam hal ini dilambangkan dengan A.

  $S \rightarrow aA$
- Bila S mendapat input b maka menuju ke q_4 yang dalam hal ini dilambangkan dengan B.

  $S \rightarrow bB$
- Karena q_0 dalam hal ini sebagai state akhir dan masih memiliki transisi keluar, maka untuk menandakannya sebagai state akhir kita buat :

  $S \rightarrow \epsilon$

Kemudian setelah itu kita lihat q_1 yang tadi telah kita lambangkan sebagai A.

- Jika A mendapat input b maka menuju q_2 yang dalam hal ini dilambangkan sebagai C.

  $A \rightarrow bC$

Kemudian kita lihat q_4 yang telah kita identikkan sebagai B.

- Jika B mendapat input b maka menuju ke q_5 yang kita lambangkan sebagai D.

  $B \rightarrow bD$

Kemudian kita lihat q_2 yang telah kita lambangkan sebagai C.

- Jika C mendapat input b maka menuju ke q_3 (Tetapi karena q_3 tidak mempunyai transisi keluar dan bukan merupakan *state* akhir maka dapat kita abaikan.
- Jika C mendapat input a maka menuju ke S.

  $C \rightarrow aS$

Kemudian kita lihat q_5 yang telah kita lambangkan sebagai D.

- Jika D mendapat input b maka menuju ke S.

D → bS

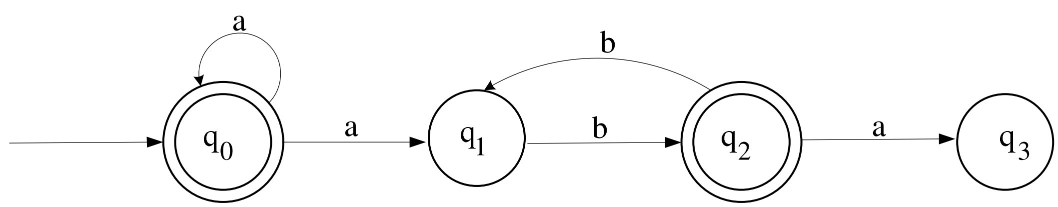
- Jika D mendapat input a maka menuju ke q_6 (tetapi karena q_6 bukan merupakan state akhir dan tidak ada transisi keluar dari q_6 maka dapat diabaikan)

Maka Diperoleh :

V = {S, A, B, C, D}

P = {S → aA | bB | ϵ, A → bC, B → bD, C → aS, D → bS}

Contoh lain dapat dilihat pada gambar di bawah ini.



T = {a, b}

S = S

Kita mulai dari *state* awal yaitu q_0 yang dalam hal ini dilambangkan dengan S.

- Bila S mendapat input a maka menuju ke q_1 yang dalam hal ini dilambangkan dengan A.

  S → aA
- Bila S mendapat input a maka menuju ke S sendiri

  S → aS
- Karena S adalah state awal dan bukan satu-satu nya state akhir maka tidak perlu ditambahkan S → ϵ

Kemudian kita lihat q_1 yang dalam hal ini dilambangkan dengan A.

- Jika A mendapat input b maka menuju ke state q_2 yang dalam hal ini dilambangkan dengan B.

  A → bB

Kemudian kita lihat q_2 yang dalam hal ini dilambangkan dengan B

- Jika B mendapat input b maka menuju ke state q_1 yang dalam hal ini dilambangkan dengan A.

  B → bA

- Jika B mendapat input a maka menuju ke state q_3, tetapi karena q_3 buka merupakan state akhir dan tidak mempunyai transisi keluar maka dapat diabaikan.
- Karena B merupakan state akhir dan mempunyai transisi keluar maka untuk menandainya ditulis.

  B → ϵ

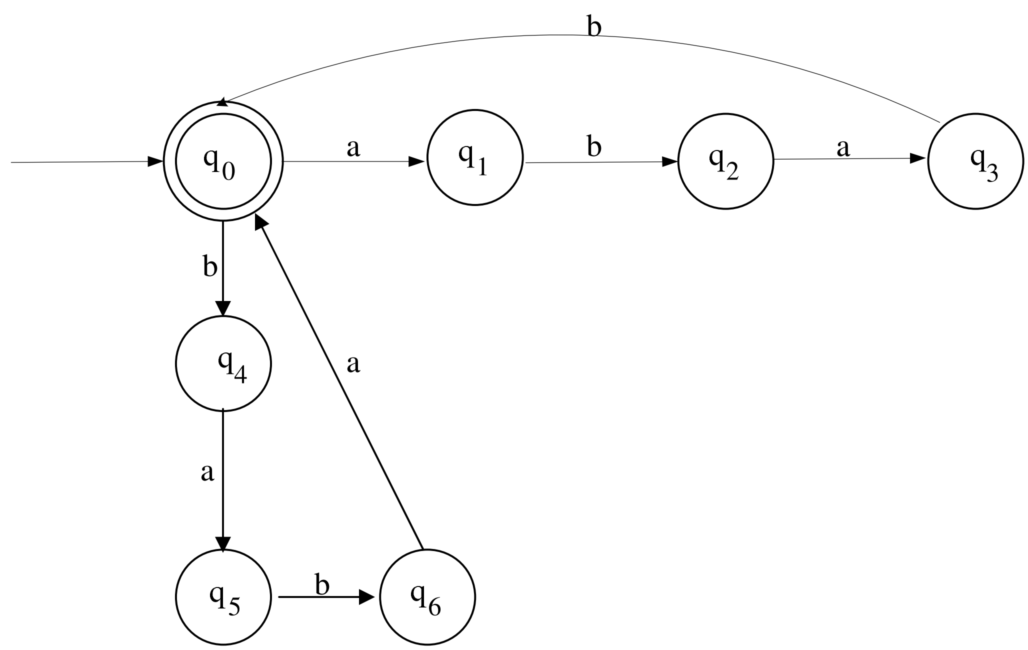
Maka diperoleh :

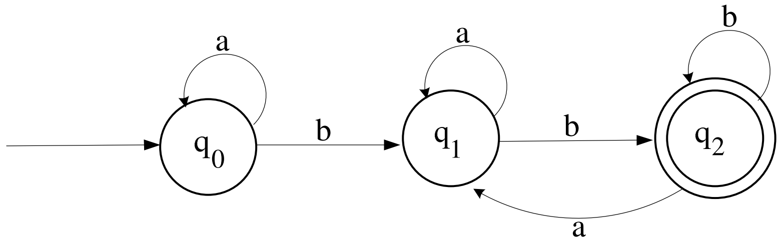
V = {S, A, B}

P = {S → aA | aS, A → bB, B → bA | ϵ}

Soal :

Konstruksikanlah tata bahasa regular untuk automata berikut.





Jawab :

1.

T = {a,b}

S = S

Kita mulai dari state awal yaitu q_0 yang dinotasikan dengan S.

- Jika S mendapat input a maka menuju ke q_1 yang dalam hal ini dinotasikan dengan A.

  $S \rightarrow aA$

- Jika S mendapat input b maka menuju ke q_4 yang dalam hal ini dinotasikan dengan B.

  $S \rightarrow bB$

- Karena S adalah state awal dan juga merupakan state akhir dan masih mempunyai transisi keluar maka

  $S \rightarrow \epsilon$

Kemudian kita lihat q_1 yang tadi sudah kita notasikan dengan A.

- Jika A mendapat input b maka menuju ke q_2 yang dalam hal ini dinotasikan dengan C.

  $A \rightarrow bC$

Kemudian kita lihat q_4 yang tadi sudah kita notasikan dengan B.

- Jika B mendapat input a maka menuju ke state q_5 yang dalam hal ini dinotasikan dengan D.

  $B \rightarrow aD$

Kemudian kita lihat state q_2 yang tadi dinotasikan dengan C.

- Jika C mendapat input a maka menuju ke state q_3 yang dalam hal ini dinotasikan dengan E.

  $C \rightarrow aE$

Kemudian kita lihat state q_5 yang tadi dinotasikan dengan D.

- Jika D mendapat input b maka menuju ke state q_6 yang dalam hal ini dinotasikan dengan F.

  $D \rightarrow bF$

Kemudian kita lihat state q_3 yang tadi kita notasikan dengan E.

- Jika E mendapat input b maka menuju ke state awal yaitu S.

  $E \rightarrow bS$

Kemudian kita lihat state q_6 yang tadi kita notasikan dengan F.

- Jika F mendapat input a maka menuju ke state awal yaitu S.

  $F \rightarrow aS$

V = {A, B, C, D, E, F }

P = {$S \rightarrow aA \mid bB \mid \epsilon$, $A \rightarrow bC$, $B \rightarrow aD$, $C \rightarrow aE$, $D \rightarrow bF$, $E \rightarrow bS$, $F \rightarrow aS$

<u>Finite State Automata untuk Suatu Tata Bahasa Regular</u>

Bila sebelumnya dari suatu diagram transisi *Finite State Automata* kita bisa membuat aturan – aturan produksinya, sebaliknya kita juga bisa mengkonstruksi diagram transisi *finite state automata* untuk suatu tata bahasa regular yang diketahui aturan – aturan produksinya.

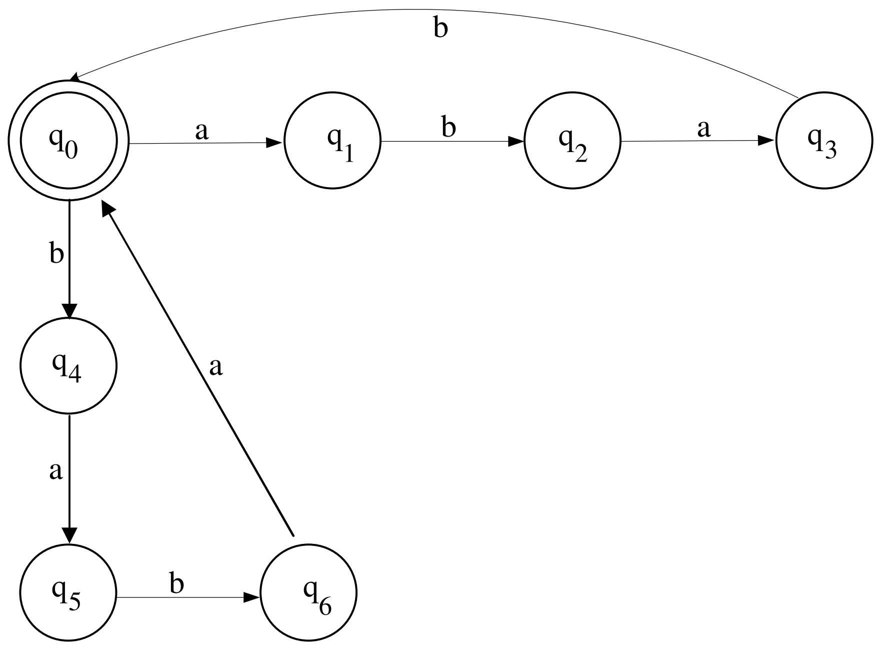
Misalkan terdapat tata bahasa regular dengan aturan produksi.

$S \rightarrow aB \mid bA \mid \epsilon$

$A \rightarrow abaS$

$B \rightarrow babS$

Maka diagram transisinya dapat digambar sebagai berikut.



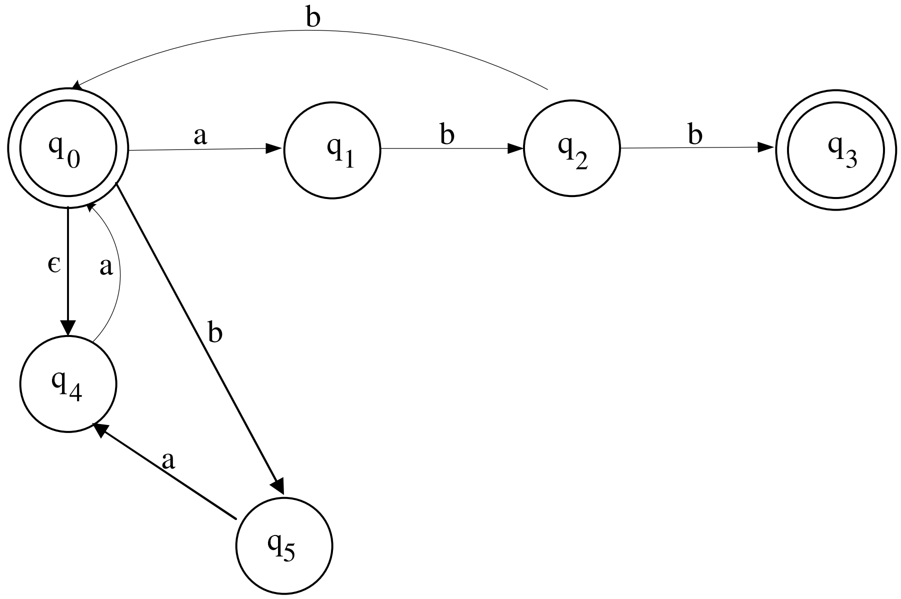
Contoh lain, akan dibuat diagram transisi untuk tata bahasa regular :

$S \rightarrow abA \mid B \mid baB \mid \epsilon$

$A \rightarrow bS \mid b$

$B \rightarrow aS$

Jawab :

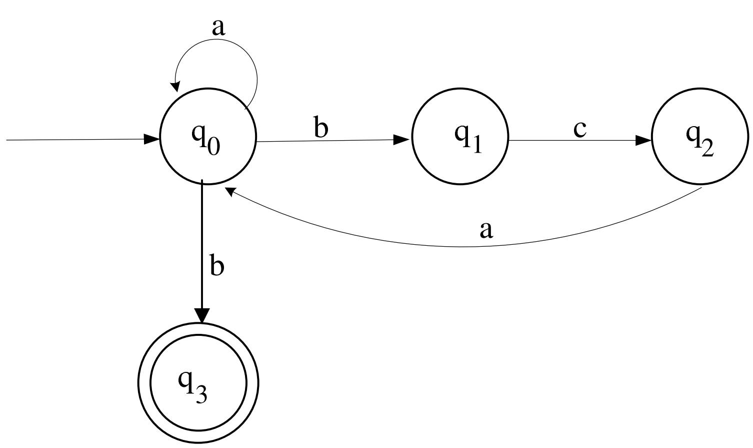


Contoh lain, akan dibuat diagram transisi untuk tata bahasa regular :

S → aS | bB | b

B → cC

C → aS



Tata Bahasa Bebas Konteks

Bila pada tata bahasa regular terdapat pembatasan pada ruas kanan atau hasil produksinya, maka pada tata bahasa bebas konteks / *Context Free Grammar*, selanjutnya kita sebut sebagai CFG, tidak terdapat pembatasan hasil produksinya.

Sebagai contoh :

B → CDeFg

D → BcDe

Pohon Penurunan (*Derivation Tree*)

Pohon penurunan (*derivation tree / parse tree*) berguna untuk menggambarkan bagaimana memperoleh suatu *string* (untai) dengan cara menurunkan simbol-simbol variabel menjadi simbol-simbol terminal. Setiap simbol variabel akan diturunkan menjadi terminal sampai tidak ada yang belum tergantikan.

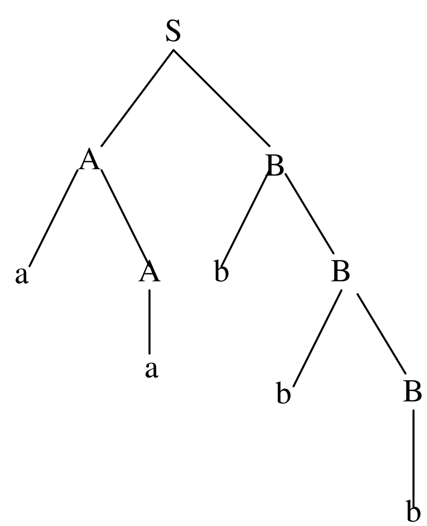
Misalkan terdapat tata bahasa bebas konteks dengan aturan produksi :

S → AB

A → aA | a

B → bB | b

Akan kita gambarkan pohon penurunan untuk memperoleh untai : 'aabbb'



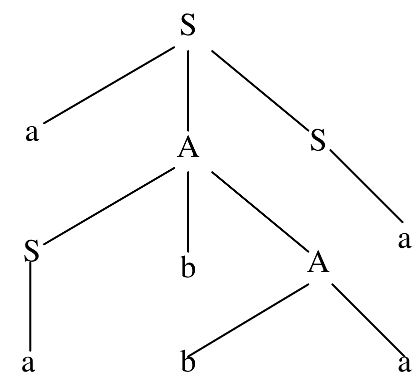
Contoh lain, terdapat tata bahasa bebas konteks :

S → aAS | a

A → SbA | ba

Gambarkan pohon penurunan untuk memperoleh untai 'aabbaa'

Jawab :



Contoh lain, terdapat tata bahasa bebas konteks memiliki aturan produksi sebagai berikut

S → aB | bA

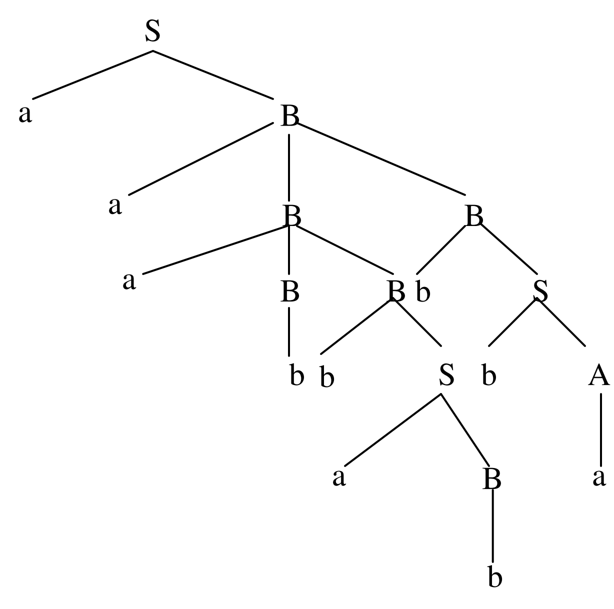
A → a | aS | bAA

B → b | bS | aBB

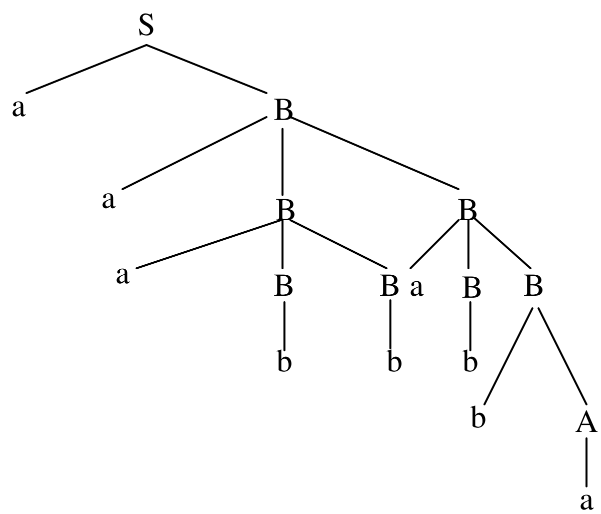
Gambarkan pohon penurunan untuk memperoleh untai 'aaabbabbba'

Jawab :

Versi 1 :



Versi 2 :



Soal :

1. Untuk tata bahasa bebas konteks berikut.

   S → AA

   A → AAA | a | bA | Ab

   Gambarkan pohon penurunan untuk memperoleh untai ‘bbabaaba’

2. Untuk tata bahasa bebas konteks berikut.

   S → aAd | aB

   A → b | c

   B → ccd | ddc

   Gambarkan pohon penurunan untuk memperoleh untai ’accd’

3. Untuk tata bahasa bebas konteks berikut.

   S → AB

   A → Aa | bB

   B → a | Sb

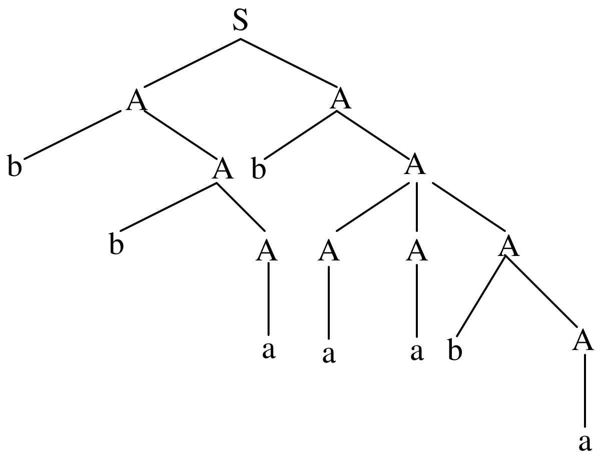
   Berikanlah pohon penurunan untuk memperoleh untai ’baabaab’

4. Untuk tata bahasa bebas konteks berikut.

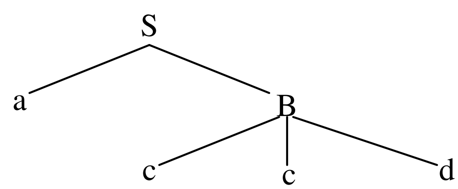
   Gambarkan pohon penurunan untuk memperoleh untai ’bbaaaabb’

Jawab :

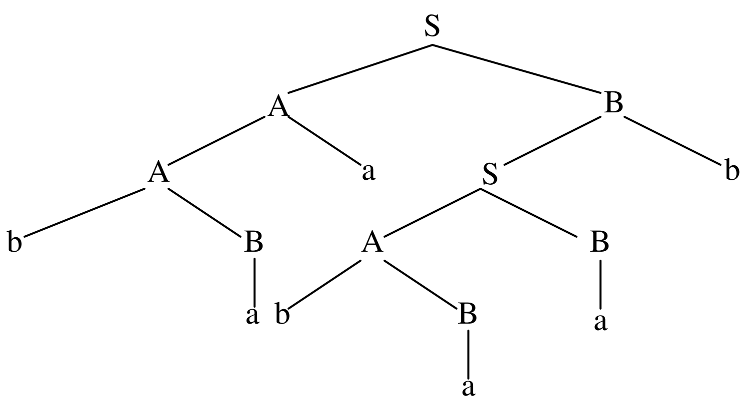
Soal nomor 1



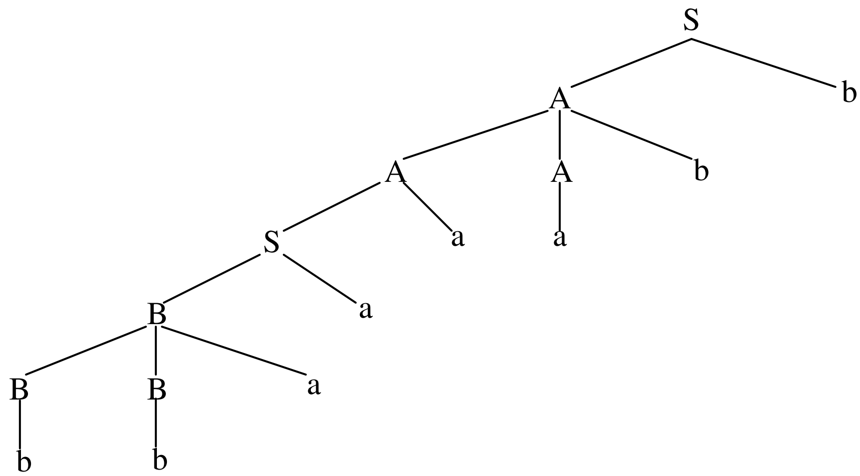
Soal nomor 2



Soal nomor 3



Soal nomor 4



<u>**Ambiguitas**</u>

Ambiguitas / kedwiartian terjadi bila terdapat lebih dari satu pohon penurunan yang berbeda untuk memperoleh suatu untai.

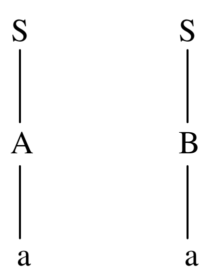
Misalkan terdapat bebas konteks :

S → A | B

A → a

B → a

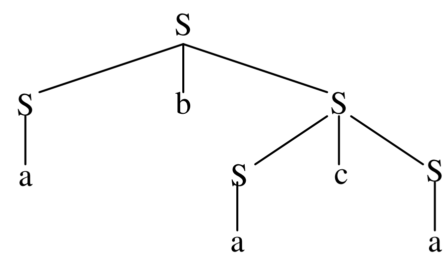
Untuk memperoleh suatu untai ’a’ bisa terdapat dua cara penurunan seperti yang ditunjukkan pada pohon penurunan berikut ini.

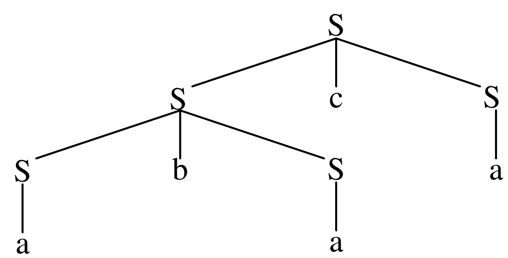


Contoh lain, terdapat tata bahasa bebas konteks :

S → SbS | ScS | a

Kita dapat memperoleh untai ’abaca’ dalam dua cara berikut ini.





Soal :

Buktikan bahwa tata bahasa bebas konteks berikut ambigu :

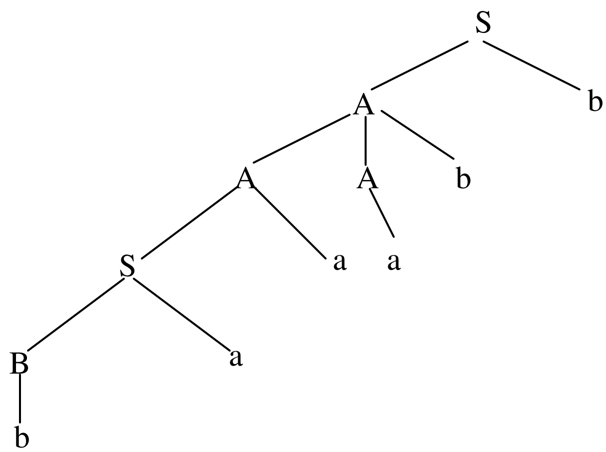
S → aB | bA

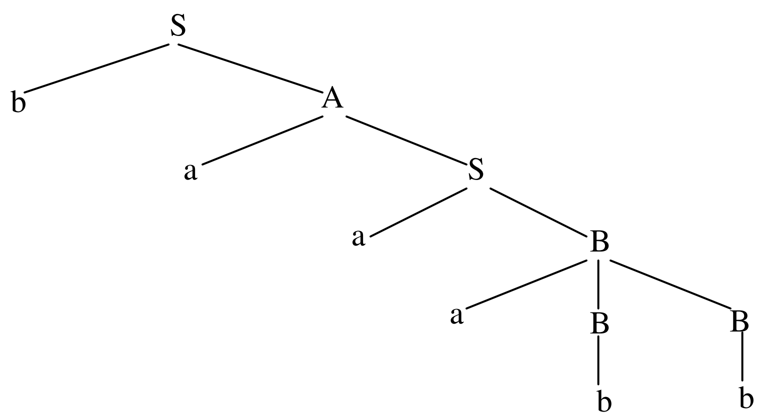
A → a | aS | bAA

B → b | bS | aBB

Jawab :

Terdapat dua cara untuk menghasilkan untai ‘baaabb’





Penyederhanaan Tata Bahasa Bebas Konteks

Penyederhanaan tata bahasa bebas konteks bertujuan untuk melakukan pembatasan sehingga tidak menghasilkan pohon penurunan yang memiliki kerumitan yang tak perlu atau aturan produksi yang tidak berarti. Misalkan terdapat tata bahasa bebas konteks :

$S \rightarrow AB \mid a$

$A \rightarrow a$

Kelemahan tata bahasa bebas konteks di atas, aturan produksi $S \rightarrow AB$ tidak berarti karena B tidak memiliki penurunan.

Untuk tata bahasa bebas konteks berikut.

$S \rightarrow A$

$A \rightarrow B$

$B \rightarrow C$

$C \rightarrow D$

$D \rightarrow a \mid A$

Memiliki kelemahan terlalu panjang jalannya padahal berujung pada $S \rightarrow a$, produksi $D \rightarrow A$ juga memiliki kerumitan.

Suatu tata bahasa bebas konteks dapat disederhanakan dengan melakukan cara berikut ini.

1. Penghilangan produksi *useless* (tidak berguna)
2. Penghilangan produksi unit.
3. Penghilangan produksi ϵ

Penghilangan Produksi Useless

Penghilangan produksi *useless* dilakukan dengan cara sebagai berikut.

1. Menghilangkan produksi yang memuat simbol variabel yang tidak memiliki penurunan yang akan menghasilkan simbol terminal.
2. Produksi yang tidak akan pernah dicapai dengan penurunan apapun dari simbol awal sehingga produksi itu redundan (berlebih).

Contoh, terdapat tata bahasa bebas konteks :

$S \rightarrow aSa \mid Abd \mid Bde$

$A \rightarrow Ada$

$B \rightarrow BBB \mid a$

Kita bisa melihat bahwa :

1. Simbol variabel A tidak memiliki penurunan yang menuju terminal, sehingga bisa dihilangkan.
2. Karena simbol A telah dihilangkan, maka dengan sendirinya $S \rightarrow Abd$ juga dihilangkan.

Maka dari tata bahasa bebas konteks di atas, produksi yang *useless* :

$A \rightarrow Ada$

S → Abd

Maka tata bahasa bebas konteks setelah disederhanakan menjadi :

S → aSa | Bde

B → BBB | a

Contoh lain, terdapat tata bahasa bebas konteks sebagai berikut.

S → Aa | B

A → ab | D

B → b | E

C → bb

E → aEa

Kita bisa melihat bahwa :

1. Aturan produksi A → D, simbol variabel D tidak memiliki penurunan, sehingga bisa dihilangkan
2. Aturan produksi C → bb, bila kita coba melakukan penurunan dari simbol awal S, dengan jalan manapun tidak akan pernah mencapai C, sehingga bisa dihilangkan.
3. Simbol variabel E tidak memiliki aturan produksi yang menuju terminal, sehingga bisa dihilangkan.
4. Konsekuensi dari aturan no. 3 maka aturan produksi B → E, juga mesti dihilangkan.

Maka dari tata bahasa bebas konteks tersebut produksi yang *useless* adalah sebagai berikut.

A → D

C → bb

E → aEa

B → E

Maka tata bahasa bebas konteks setelah disederhanakan menjadi :

S → Aa | B

A → ab

B → b

Contoh lain, terdapat tata bahasa bebas konteks berikut.

S → aAb | cEB

A → dBE | eeC

B → ff

C → ae

D → h

Kita bisa melihat bahwa :

1. S → cEB, E tidak memiliki penurunan, sehingga bisa dihilangkan.
2. A → dBE, E tidak memiliki penurunan sehingga bisa dihilangkan.
3. D → h, tidak dapat dicapai melalui penurunan manapun sehingga redundan.
4. B → ff, karena S → cEB dan A → dBE sudah dihilangkan, maka tidak akan dapat dicapai melalui penurunan manapun sehingga bisa dihilangkan.

Maka aturan produksi yang *useless* adalah sebagai berikut.

S → cEB

A → dBE

D → h

B → ff

Sehingga tata bahasa bebas konteks hasil penyederhanaan adalah sebagai berikut.

S → aAb

A → eeC

C → ae

Contoh lain, terdapat tata bahasa bebas konteks berikut.

S → aB

A → bcD | dAC

B → e | Ab

C → bCb | adF | ab

F → cFB

Kita bisa melihat bahwa :

1. Aturan produksi F → cFB tidak memiliki penurunan yang menuju ke simbol terminal sehingga bisa dihilangkan.
2. A → bcD, variabel D tidak memiliki penurunan, sehingga bisa dihilangkan
3. A → dAC, tidak memiliki penurunan yang menghasilkan simbol terminal sehingga bisa dihilangkan.
4. Konsekuensi dari no. 2 dan no.3, maka B → Ab, juga dihilangkan.
5. C → adF juga dihilangkan, karena F telah dihilangkan.

Maka aturan produksi setelah disederhanakan menjadi :

S → aB

B → e

C → bCb | ab

Contoh lain, terdapat tata bahasa bebas konteks sebagai berikut.

S → aBD

B → cD | Ab

D → ef

A → Ed

F → dc

Kita bisa melihat bahwa :

1. Aturan produksi A → Ed, E tidak memiliki penurunan sehingga bisa dihilangkan.
2. Sebagai konsekuensi dari no.1 maka B → Ab juga dihilangkan.

3. $F \rightarrow dc$, tidak dapat dicapai dari penurunan apapun sehingga bisa dihilangkan.

Maka tata bahasa bebas konteks setelah disederhanakan menjadi :

$S \rightarrow aBD$

$B \rightarrow cD$

$D \rightarrow ef$

Contoh, tata bahasa bebas konteks :

$S \rightarrow Abc \mid ab$

$A \rightarrow AAA \mid \epsilon$

Maka tata bahasa bebas konteks setelah disederhanakan menjadi :

$S \rightarrow Abc \mid ab$

$A \rightarrow AAA \mid \epsilon$

Ingat $A \rightarrow \epsilon$, juga harus diperhitungkan.

Soal :

1. Hilangkanlah semua aturan produksi yang *useless* dari tata bahasa bebas konteks berikut

   $S \rightarrow AB \mid CA$

   $B \rightarrow BC \mid AB$

   $A \rightarrow a$

   $C \rightarrow aB \mid b$
2. Hilangkan semua aturan produksi yang *useless* dari tata bahasa bebas konteks berikut.

   $S \rightarrow aS \mid A \mid C$

   $A \rightarrow a$

   $B \rightarrow aa$

   $C \rightarrow aCb$
3. Hilangkan semua aturan produksi yang *useless* dari tata bahasa bebas konteks berikut.

   $S \rightarrow A$

A → aA | ϵ

B → bA

Jawaban :

Soal nomor 1

Kita bisa melihat bahwa :

1. Aturan produksi B → BC dan B → AB, tidak memiliki penurunan yang menuju ke simbol terminal sehingga bisa dihilangkan.
2. Sebagai konsekuensi dari nomor 1, maka S → AB juga dihilangkan.
3. Sebagai konsekuensi dari nomor 1, maka C → aB juga dihilangkan.

Maka tata bahasa bebas konteks setelah penyederhanaan adalah sebagai berikut.

S → CA

A → a

C → b

Soal nomor 2

Kita bisa melihat bahwa :

1. C → aCb tidak memiliki penurunan ke simbol terminal sehingga bisa dihilangkan.
2. Sebagai konsekuensi dari nomor 1 maka S → C juga bisa dihilangkan.
3. B → aa tidak dapat dicapai dari penurunan apapun sehingga bisa dihilangkan.

Maka tata bahasa bebas konteks setelah penyederhanaan adalah sebagai berikut.

S → aS | A

A → a

Soal nomor 3

Kita bisa melihat bahwa :

1. B → bA tidak dapat dicapai dari penurunan apapun sehingga bisa dihilangkan.

Maka tata bahasa bebas konteks setelah disederhanakan adalah sebagai berikut.

$S \rightarrow A$

$A \rightarrow aA \mid \epsilon$

<u>Penghilangan Produksi Unit</u>

Produksi unit adalah produksi di mana ruas kiri dan kanan aturan produksi hanya berupa satu simbol variabel, misalkan : $A \rightarrow B$, $C \rightarrow D$. Keberadaan produksi unit membuat tata bahasa memiliki kerumitan yang tak perlu atau menambah panjang penurunan. Penyederhanaan ini dilakukan dengan melakukan penggantian aturan produksi unit.

Contoh tata bahasa bebas konteks :

$S \rightarrow Sb$

$S \rightarrow C$

$C \rightarrow D$

$C \rightarrow ef$

$D \rightarrow dd$

Langkah-langkahnya adalah sebagai berikut :

1. Kita lihat untuk $S \rightarrow C$

   $C \rightarrow D$ lalu $D \rightarrow dd$ sehingga $C \rightarrow dd$

   $C \rightarrow ef$

   Maka $S \rightarrow C$ dapat diubah menjadi $S \rightarrow dd \mid ef$
2. Kita lihat untuk $C \rightarrow D$

   $C \rightarrow D$ lalu $D \rightarrow dd$ sehingga $C \rightarrow dd$

   Maka $C \rightarrow D$ dapat diubah menjadi $C \rightarrow dd$

Sehingga tata bahasa bebas konteks setelah penyederhanaan adalah sebagai berikut.

$S \rightarrow Sb$

$S \rightarrow dd \mid ef$

$C \rightarrow dd$

$C \rightarrow ef$

D → dd

Contoh lain, terdapat tata bahasa bebas konteks :

S → A

S → Aa

A → B

B → C

B → b

C → D

C → ab

D → b

Penggantian yang dilakukan :

1. S → A

   Untuk S → A maka setelah ditelusuri menghasilkan S → b | ab

2. A → B

   Untuk A → B maka setelah ditelusuri menghasilkan A → b | ab

3. B → C

   Untuk B → C maka setelah ditelusuri menghasilkan B → b | ab

   Untuk B → b sudah ada, maka cukup ditulis B → ab

4. C → D

   Untuk C → D maka setelah ditelusuri menghasilkan C → b | ab

   Untuk C → ab sudah ada, maka cukup ditulis C → b

Sehingga tata bahasa bebas konteks setelah disederhanakan menjadi :

S → ab | b

S → Aa

A → ab | b

B → ab

B → b

C → b

$C \rightarrow ab$

$D \rightarrow b$

Contoh lain, terdapat tata bahasa bebas konteks :

$S \rightarrow Cba \mid D$

$A \rightarrow bbC$

$B \rightarrow Sc \mid ddd$

$C \rightarrow eA \mid f \mid C$

$D \rightarrow E \mid SABC$

$E \rightarrow gh$

Penggantian yang dilakukan :

1. $D \rightarrow E$ menjadi $D \rightarrow gh$
2. $C \rightarrow C$, kita hapus
3. $S \rightarrow D$ menjadi $S \rightarrow gh \mid SABC$

Sehingga tata bahasa bebas konteks setelah penyederhanaan menjadi :

$S \rightarrow Cba \mid gh \mid SABC$

$A \rightarrow bbC$

$B \rightarrow Sc \mid ddd$

$C \rightarrow eA \mid f$

$D \rightarrow gh \mid SABC$

$E \rightarrow gh$

Soal :

1. Hilangkanlah semua aturan produksi unit dari tata bahasa bebas konteks berikut.

   $S \rightarrow Aa \mid B$

   $B \rightarrow A \mid bb$

   $A \rightarrow a \mid bc \mid B$
2. Hilangkan semua aturan produksi unit dari tata bahasa bebas konteks berikut.

   $S \rightarrow AbaC \mid BaC \mid AaC \mid Aba \mid aC \mid Aa \mid Ba \mid a$

$A \rightarrow B \mid C \mid BC$

$B \rightarrow b$

$C \rightarrow D$

$D \rightarrow d$

Jawaban :

Soal Nomor 1

Penggantian yang dilakukan :

1. $S \rightarrow B$ menjadi $S \rightarrow a \mid bb \mid bc$
2. $B \rightarrow A$ menjadi $B \rightarrow a \mid bc$
3. $A \rightarrow B$ menjadi $A \rightarrow bb$

Maka tata bahasa bebas konteks setelah penyederhanaan menjadi :

$S \rightarrow Aa \mid a \mid bb \mid bc$

$B \rightarrow a \mid bc \mid bb$

$A \rightarrow a \mid bc \mid bb$

Soal Nomor 2

Penggantian yang dilakukan :

1. $A \rightarrow B$ menjadi $A \rightarrow b$
2. $A \rightarrow C$ menjadi $A \rightarrow d$
3. $C \rightarrow D$ menjadi $C \rightarrow d$

Maka tata bahasa bebas konteks setelah penyederhanaan menjadi :

$S \rightarrow AbaC \mid BaC \mid AaC \mid Aba \mid aC \mid Aa \mid Ba \mid a$

$A \rightarrow b \mid d \mid BC$

$B \rightarrow b$

$C \rightarrow d$

$D \rightarrow d$

<u>Penghilangan Produksi ϵ</u>

Produksi ϵ adalah produksi dalam bentuk :

$\alpha \rightarrow \epsilon$

atau bisa dianggap sebagai produksi kosong (*empty*). Penghilangan produksi ϵ dilakukan dengan melakukan penggantian produksi yang memuat variabel yang bisa menuju ke produksi ϵ, atau biasa disebut *nullable*. Prinsip penggantiannya bisa dilihat pada kasus berikut ini.

$S \rightarrow bcAd$

$A \rightarrow \epsilon$

Pada kasus diatas A *nullable*, serta $A \rightarrow \epsilon$ satu-satunya produksi dari A, maka variabel A bisa ditiadakan, hasil penyederhanaan tata bahasa bebas konteks menjadi :

$S \rightarrow bcd$

Tetapi bila kasusnya :

$S \rightarrow bcAd$

$A \rightarrow bd \mid \epsilon$

Pada kasus di atas A *nullable,* tetapi $A \rightarrow \epsilon$ bukan satu-satunya produksi dari A, maka hasil penyederhanaan :

$S \rightarrow bcAd \mid bcd$

$A \rightarrow bd$

Contoh, terdapat tata bahasa bebas konteks :

$S \rightarrow Ab \mid Cd$

$A \rightarrow d$

$C \rightarrow \epsilon$

Variabel yang *nullable* adalah variabel C. Karena penurunan $C \rightarrow \epsilon$ merupakan penurunan satu-satunya dari C, maka kita ganti $S \rightarrow Cd$ menjadi $S \rightarrow d$. Kemudian produksi $C \rightarrow \epsilon$ kita hapus.

Tata bahasa bebas konteks setelah penyederhanaan :

$S \rightarrow Ab \mid d$

$A \rightarrow d$

Contoh lai, terdapat tata bahasa bebas konteks :

$S \rightarrow dA \mid Bd$

$A \rightarrow bc$

$A \rightarrow \epsilon$

$B \rightarrow c$

Variabel yang *nullable* adalah variabel A. $A \rightarrow \epsilon$ bukan penurunan satu-satunya dari A (terdapat $A \rightarrow bc$), maka kita ganti $S \rightarrow dA$ menjadi $S \rightarrow dA \mid d$. $A \rightarrow \epsilon$ kita hapus.
Setelah penyederhanaan :

$S \rightarrow dA \mid d \mid Bd$

$A \rightarrow bc$

$B \rightarrow c$

Contoh lain, terdapat tata bahasa bebas konteks :

$S \rightarrow AaCD$

$A \rightarrow CD \mid AB$

$B \rightarrow b \mid \epsilon$

$C \rightarrow d \mid \epsilon$

$D \rightarrow \epsilon$

Variabel yang *nullable* adalah variabel B, C, D. Kemudian kita lihat $A \rightarrow CD$, maka variabel A juga *nullable*, karena D hanya memiliki penurunan $D \rightarrow \epsilon$, maka kita sederhanakan dulu :

- $S \rightarrow AaCD$ menjadi $S \rightarrow AaC$
- $A \rightarrow CD$ menjadi $A \rightarrow C$
- $D \rightarrow \epsilon$ kita hapus

Selanjutnya kita lihat variabel B dan C memiliki penurunan ϵ, meskipun bukan satu-satunya penurunan, maka kita lakukan penggantian :

- $A \rightarrow AB$ menjadi $A \rightarrow AB \mid A \mid B$
- $S \rightarrow AaC$ menjadi $S \rightarrow AaC \mid aC \mid Aa \mid a$
- $B \rightarrow \epsilon$ dan $C \rightarrow \epsilon$ kita hapus

Setelah penyederhanaan :

$S \rightarrow AaC \mid aC \mid Aa \mid a$

$A \rightarrow C \mid AB \mid A \mid B$

$B \rightarrow b$

$C \rightarrow d$

Contoh lain, terdapat tata bahasa bebas konteks :

$S \rightarrow AB$

$A \rightarrow abB \mid aCa \mid \epsilon$

$B \rightarrow bA \mid BB \mid \epsilon$

$C \rightarrow \epsilon$

Variabel yang *nullable* adalah A, B, dan C. Dari $S \rightarrow AB$, maka S juga *nullable*. Kita lakukan penggantian :

$S \rightarrow AB$ menjadi $S \rightarrow AB \mid A \mid B$

$A \rightarrow abB$ menjadi $A \rightarrow abB \mid ab$

$A \rightarrow aCa$ menjadi $A \rightarrow aa$

$B \rightarrow bA$ menjadi $B \rightarrow bA \mid b$

$B \rightarrow BB$ menjadi $B \rightarrow BB \mid B$

$C \rightarrow \epsilon$, $B \rightarrow \epsilon$, dan $A \rightarrow \epsilon$ kita hapus.

Hasil akhir penyederhanaan menjadi :

$S \rightarrow AB \mid A \mid B$

$A \rightarrow abB \mid ab \mid aa$

$B \rightarrow bA \mid b \mid BB \mid B$

Contoh lain, terdapat tata bahasa bebas konteks :

$S \rightarrow aAb$

$A \rightarrow aAb \mid \epsilon$

Hasil akhir penyederhanaan menjadi :

$S \rightarrow aAb \mid ab$

$A \rightarrow aAb \mid ab$

Contoh lain, terdapat tata bahasa bebas konteks :

$S \rightarrow ABaC$

$A \rightarrow BC$

$B \rightarrow b \mid \epsilon$

$C \rightarrow D \mid \epsilon$

$D \rightarrow d$

Hasil penyederhanaan :

$S \rightarrow ABaC \mid AaC \mid Aba \mid BaC \mid aC \mid Aa \mid Ba \mid a$

$A \rightarrow BC \mid B \mid C$

$B \rightarrow b$

$C \rightarrow D$

$D \rightarrow d$

Soal :

1. Hilangkanlah semua produksi ϵ dari tata bahasa bebas konteks berikut.

   $S \rightarrow ABaC$

   $A \rightarrow Bd$

   $B \rightarrow b \mid \epsilon$

   $C \rightarrow D \mid \epsilon$

   $D \rightarrow BCa$
2. Hilangkanlah semua produksi ϵ dari tata bahasa bebas konteks berikut.

   $S \rightarrow AaB \mid aaB$

   $A \rightarrow \epsilon$

   $B \rightarrow bbA \mid \epsilon$
3. Hilangkanlah semua produksi ϵ dari tata bahasa bebas konteks berikut.

   $S \rightarrow aSb \mid SS \mid \epsilon$

4. Hilangkanlah semua produksi dari tata bahasa bebas konteks berikut.

   $S \rightarrow AB$

   $A \rightarrow aA \mid abB \mid aCa$

   $B \rightarrow bA \mid BB \mid \epsilon$

   $C \rightarrow \epsilon$

   $D \rightarrow dB \mid BCB$

Jawab :

1.

   $S \rightarrow ABaC \mid BaC \mid AaC \mid Aba \mid aC \mid Ba \mid Aa \mid a$

   $A \rightarrow Bd \mid d$

   $B \rightarrow b$

   $C \rightarrow D$

   $D \rightarrow Bca \mid Ba \mid Ca \mid a$

2.

   $S \rightarrow aB \mid a \mid aaB \mid aa$

   $B \rightarrow bb$

3.

   $S \rightarrow aSb \mid ab \mid SS \mid S$

4.

   $S \rightarrow AB \mid A$

   $A \rightarrow aA \mid abB \mid ab \mid aa$

   $B \rightarrow bA \mid BB \mid B$

   $D \rightarrow dB \mid d \mid BB \mid B$

Pada prakteknya, ketiga penyederhanaan tersebut (penghilangan *useless*, unit, dan ϵ) dilakukan bersama pada suatu tata bahasa bebas konteks, yang nantinya menyiapkan tata bahasa bebas konteks tersebut untuk diubah ke dalam suatu *bentuk normal chomsky* yang akan dibahasa pada bagian selanjutnya.

Adapun urutan pengerjaannya adalah sebagai berikut.

1. Hilangkan produksi ϵ
2. Hilangkan produksi unit
3. Hilangkan produksi *useless*.

Kita coba untuk melakukan ketiga penyederhanaan tersebut pada aturan produksi berikut.

S → AA | C | bd

A → Bb | ϵ

B → AB | d

C → de

Pertama – tama kita lakukan penghilangan produksi ϵ, sehingga aturan produksi menjadi:

S → AA | A | C | bd

A → Bb

B → AB | B | d

C → de

Anda bisa melihat bahwa penghilangan produksi ϵ berpotensi untuk menghasilkan produksi unit baru yang sebelumnya tidak ada. Selanjutnya kita lakukan penghilangan produksi unit menjadi :

S → AA | Bb | de | bd

A → Bb

B → AB | d

C → de

Anda bisa melihat bahwa penghilangan produksi unit bisa menghasilkan produksi *useless*. Terakhir kita lakukan penghilangan produksi *useless* :

S → AA | Bb | de | bd

A → Bb

B → AB | d

Soal :

1. Lakukan penghilangan produksi unit, *useless*, dan ϵ dari tata bahasa bebas konteks berikut.

   $S \rightarrow a \mid aA \mid B \mid C$

   $A \rightarrow aB \mid \epsilon$

   $B \rightarrow Aa$

   $C \rightarrow cCD$

   $D \rightarrow ddd$

2. Lakukan penghilangan produksi unit, *useless*, dan ϵ dari tata bahasa bebas konteks berikut.

   $S \rightarrow aB \mid aaB$

   $A \rightarrow \epsilon$

   $B \rightarrow bA$

   $B \rightarrow \epsilon$

3. Hilangkan semua aturan produksi *useless* dan unit dari tata bahasa bebas konteks berikut.

   $S \rightarrow AaC \mid aC \mid Aa \mid a$

   $A \rightarrow C \mid AB \mid A \mid B$

   $B \rightarrow b$

   $C \rightarrow d$

Jawab :

1.

Penghilangan produksi ϵ diperoleh :

$S \rightarrow a \mid aA \mid B \mid C$

$A \rightarrow aB$

$B \rightarrow Aa \mid a$

$C \rightarrow cCD$

$D \rightarrow ddd$

Penghilangan produksi unit diperoleh :

$S \rightarrow a \mid aA \mid Aa \mid a \mid cCD$

$A \rightarrow aB$

$B \rightarrow Aa \mid a$

$C \rightarrow cCD$

$D \rightarrow ddd$

Penghilangan produksi *useless* diperoleh :

$S \rightarrow a \mid aA \mid Aa \mid a$

$A \rightarrow aB$

$B \rightarrow Aa \mid a$

2.

Penghilangan produksi ϵ diperoleh :

$S \rightarrow aB \mid a \mid aaB \mid aa$

$B \rightarrow b$

Penghilangan produksi unit diperoleh :

$S \rightarrow aB \mid a \mid aaB \mid aa$

$B \rightarrow b$

(tetap sama karena tidak ada produksi unit yang dapat dihilangkan)

Penghilangan produksi *useless* diperoleh

$S \rightarrow aB \mid a \mid aaB \mid aa$

$B \rightarrow b$

(tetap sama karena tidak ada produksi *useless* yang dapat dihilangkan)

3.

Penghilangan produksi unit diperoleh :

$S \rightarrow AaC \mid aC \mid Aa \mid a$

$A \rightarrow d \mid AB \mid b$

$B \rightarrow b$

$C \rightarrow d$

Penghilangan produksi *useless* diperoleh :

$S \rightarrow AaC \mid aC \mid Aa \mid a$

$A \rightarrow d \mid AB \mid b$

$B \rightarrow b$

$C \rightarrow d$

(tetap sama karena tidak ada produksi *useless* yang dapat dihilangkan)

Bentuk Normal Chomsky

Bentuk Normal Chomsky (Chomsky Normal Form / CNF) merupakan salah satu bentuk normal yang sangat berguna untuk tata bahasa bebas konteks (CFG). *Bentuk Normal Chomsky* dapat dibuat dari sebuah tata bahasa bebas konteks yang telah mengalami penyederhanaan, yaitu penghilangan produksi *useless*, unit, dan ϵ.

Aturan produksi dalam *bentuk normal Chomsky* ruas kanannya tepat berupa sebuah terminal atau dua variabel. Misalkan :

$A \rightarrow BC$

$A \rightarrow b$

$B \rightarrow a$

$C \rightarrow BA \mid d$

Pembentukan Bentuk Normal Chomsky

Langkah – langkah pembentukan *bentuk normal Chomsky* secara umum sebagai berikut.

- Biarkan aturan produksi yang sudah dalam bentuk *normal Chomsky*.
- Lakukan penggantian aturan produksi yang ruas kanannya memuat simbol terminal dan panjang ruas kanan > 1.
- Lakukan penggantian aturan produksi yang ruas kanannya memuat > 2 simbol variabel.
- Penggantian – penggantian tersebut bisa dilakukan berkali – kali sampai akhirnya semua aturan produksi dalam *bentuk normal Chomsky*.
- Selama dilakukan penggantian, kemungkinan kita akan memperoleh aturan – aturan produksi baru, dan juga memunculkan simbol – simbol variabel baru.

Contoh tata bahasa konteks sebagai berikut.

$S \rightarrow bA \mid aB$

$A \rightarrow bAA \mid aS \mid a$

$B \rightarrow aBB \mid bS \mid b$

Aturan produksi yang sudah dalam bentuk normal Chomsky adalah sebagai berikut.

$A \rightarrow a$

$B \rightarrow b$

Dilakukan penggantian aturan produksi yang belum *bentuk normal Chomsky.*

$S \rightarrow bA$ menjadi $S \rightarrow P_1A$

$S \rightarrow aB$ menjadi $S \rightarrow P_2 B$

$A \rightarrow bAA$ menjadi $A \rightarrow P_1 AA$ menjadi $A \rightarrow P_1 P_3$

$A \rightarrow aS$ menjadi $A \rightarrow P_2 S$

$B \rightarrow aBB$ menjadi $B \rightarrow P_2 BB$ menjadi $B \rightarrow P_2 P_4$

$B \rightarrow bS$ menjadi $B \rightarrow P_1 S$

Terbentuk aturan produksi dan simbol variabel baru :

$P_1 \rightarrow b$

$P_2 \rightarrow a$

$P_3 \rightarrow AA$

$P_1 \rightarrow BB$

Hasil akhir aturan produksi dalam bentuk normal Chomsky adalah sebagai berikut.

$A \rightarrow a$

$B \rightarrow b$

$S \rightarrow P_1 A$

$S \rightarrow P_2 B$

$A \rightarrow P_1 P_3$

$A \rightarrow P_2 S$

$B \rightarrow P_2 P_4$

$B \rightarrow P_1 S$

$P_1 \rightarrow b$

$P_2 \rightarrow a$

$P_3 \rightarrow AA$

$P_4 \rightarrow BB$

Contoh lain, tata bahasa bebas konteks :

$S \rightarrow aB \mid CA$

$A \rightarrow a \mid bc$

$B \rightarrow BC \mid Ab$

$C \rightarrow aB \mid b$

Aturan produksi yang sudah dalam bentuk normal Chomsky :

$S \rightarrow CA$

$A \rightarrow a$

$B \rightarrow BC$

$C \rightarrow b$

Penggantian aturan produksi yang belum dalam bentuk normal Chomsky :

$S \rightarrow aB$ menjadi $S \rightarrow P_1 B$

$A \rightarrow bc$ menjadi $S \rightarrow P_2 P_3$

$B \rightarrow Ab$ menjadi $B \rightarrow A P_2$

$C \rightarrow aB$ menjadi $C \rightarrow P_1 B$

Terbentuk aturan produksi dan simbol variabel baru :

$P_1 \rightarrow a$

$P_2 \rightarrow b$

$P_3 \rightarrow c$

Hasil akhir aturan produksi dalam bentuk normal Chomsky adalah sebagai berikut.

$S \rightarrow CA$

$A \rightarrow a$

$B \rightarrow BC$

$C \rightarrow b$

$S \rightarrow P_1 B$

$S \rightarrow P_2 P_3$

$B \rightarrow A P_2$

$C \rightarrow P_1 B$

$P_1 \rightarrow a$

$P_2 \rightarrow b$

$P_3 \rightarrow c$

Contoh, tata bahasa bebas konteks :

$S \rightarrow aAB \mid ch \mid CD$

$A \rightarrow dbE \mid eEC$

$B \rightarrow ff \mid DD$

$C \rightarrow ADB \mid aS$

$D \rightarrow i$

$E \rightarrow jD$

Aturan produksi yang sudah dalam bentuk normal Chomsky :

$S \rightarrow CD$

$B \rightarrow DD$

$D \rightarrow i$

Penggantian aturan produksi :

$S \rightarrow aAB$ menjadi $S \rightarrow P_1 P_2$

$S \rightarrow ch$ menjadi $S \rightarrow P_3 P_4$

$A \rightarrow dbE$ menjadi $A \rightarrow P_5 P_6$

$A \rightarrow eEC$ menjadi $A \rightarrow P_8 P_9$

$B \rightarrow ff$ menjadi $B \rightarrow P_{10} P_{10}$

$C \rightarrow ADB$ menjadi $C \rightarrow A P_{11}$

$C \rightarrow aS$ menjadi $C \rightarrow P_1S$

$E \rightarrow jD$ menjadi $E \rightarrow P_{12}D$

Terbentuk aturan produksi baru :

$P_1 \rightarrow A$

$P_2 \rightarrow AB$

$P_3 \rightarrow c$

$P_4 \rightarrow h$

$P_5 \rightarrow d$

$P_6 \rightarrow P_7 E$

$P_7 \rightarrow b$

$P_8 \rightarrow e$

$P_9 \rightarrow EC$

$P_{10} \rightarrow f$

$P_{11} \rightarrow DB$

$P_{12} \rightarrow j$

Hasil akhir dalam bentuk normal Chomsky adalah sebagai berikut.

$S \rightarrow CD$

$B \rightarrow DD$

$D \rightarrow i$

$S \rightarrow P_1 P_2$

$S \rightarrow P_3 P_4$

$A \rightarrow P_5 P_6$

$A \rightarrow P_8 P_9$

$B \rightarrow P_{10} P_{10}$

$C \rightarrow A P_{11}$

$C \rightarrow P_1S$

$E \rightarrow P_{12}D$

$P_1 \rightarrow A$

$P_2 \rightarrow AB$

$P_3 \rightarrow c$

$P_4 \rightarrow h$

$P_5 \rightarrow d$

$P_6 \rightarrow P_7 E$

$P_7 \rightarrow b$

$P_8 \rightarrow e$

$P_9 \rightarrow EC$

$P_{10} \rightarrow f$

$P_{11} \rightarrow DB$

$P_{12} \rightarrow j$

Soal :

1. Transformasikan tata bahasa bebas konteks berikut ke dalam bentuk normal Chomsky:

   $S \rightarrow aSb \mid ab$

2. Transformasikan tata bahasa bebas konteks berikut ke dalam bentuk normal Chomsky:

   $S \rightarrow aSaA \mid A$

   $A \rightarrow abA \mid b$

3. Transformasikan tata bahasa bebas konteks berikut ke dalam bentuk normal Chomsky:

   $S \rightarrow abAB$

   $A \rightarrow bAB \mid \epsilon$

   $B \rightarrow Baa \mid A \mid \epsilon$

<u>Mesin Moore</u>

Mesin *Moore* adalah suatu *Finite State Automata* yang memiliki keputusan beberapa keluaran / *output*.

Mesin Moore didefinisikan dalam 6 (enam) tupel, $M = (Q, \sum, \delta, S, \Delta, \lambda)$, di mana :

Q = himpunan *state*

Σ = himpunan simbol *input*

δ = fungsi transisi

S = *state* awal

Δ = himpunan *output*

λ = fungsi *output* untuk setiap *state*

Kita lihat contoh penerapan dari Mesin Moore. Misal kita ingin mmemperoleh pembagian (modulus) suatu bilangan dengan 3. Di mana *input* dinyatakan dalam biner. Mesin Moore yang bersesuaian bisa dilihat pada gambar di bawah ini. Konfigurasi mesinnya adalah sebagai berikut.

$Q = \{ q_0 , q_1 , q_2 \}$

$\Sigma = \{0,1\}$ (*input* dalam biner)

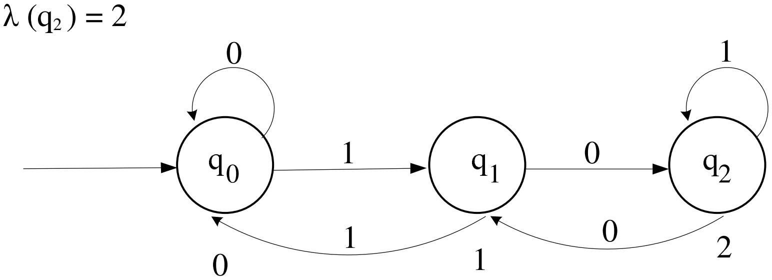
$\Delta = \{0, 1, 2\}$ (untuk *output*-nya pada kasus *mod* dengan 3, maka sisanya kemungkinan adalah 0, 1, 2)

$S = q_0$

$\lambda (q_0) = 0$

$\lambda (q_1) = 1$

$\lambda (q_2) = 2$



Gambar Mesin *Moore* untuk modulus 3

Misalkan saja :

- 5 mod 3 = ?

  Input 5 dalam biner 101

  Bila kita masukkan 101 ke dalam mesin, urutan *state* yang dicapai : q_0 , q_1 , q_2 , q_2

  Perhatikan state terakhir yang dicapai adalah q_2, $\lambda (q_2) = 2$, maka 5 mod 3 = 2

- 10 mod 3 = ?

  Input 10 dalam biner 1010

  Bila kita masukkan ke dalam mesin, urutan *state* yang dicapai : q_0, q_1, q_2, q_2, q_1

  $\lambda(q_1) = 1$, maka 10 mod 3 = 1

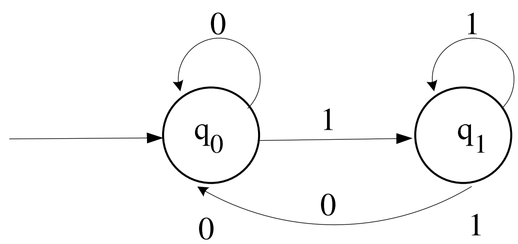
Soal :

Rancanglah mesin *moore* untuk perhitungan :

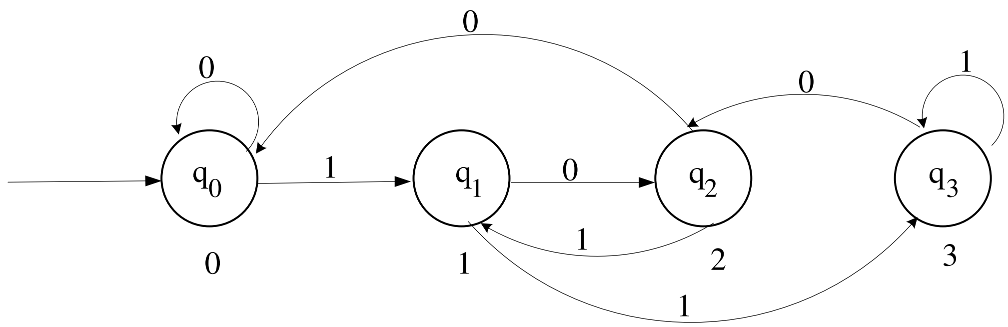
a. Modulus 2
b. Modulus 4
c. Modulus 5
d. Modulus 6

Jawab :

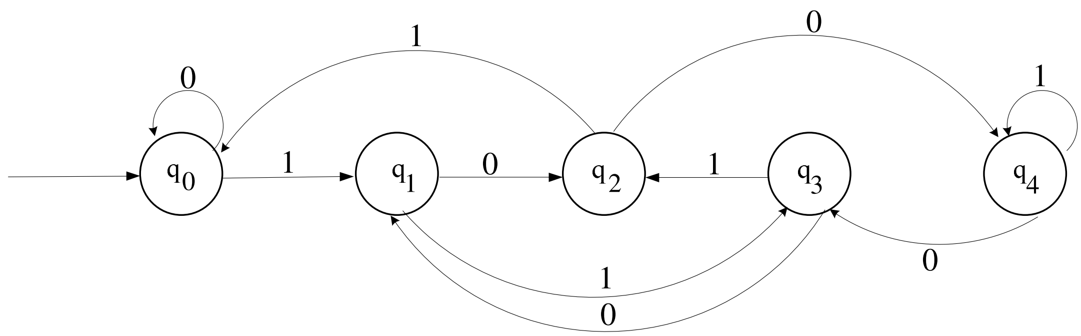
a.



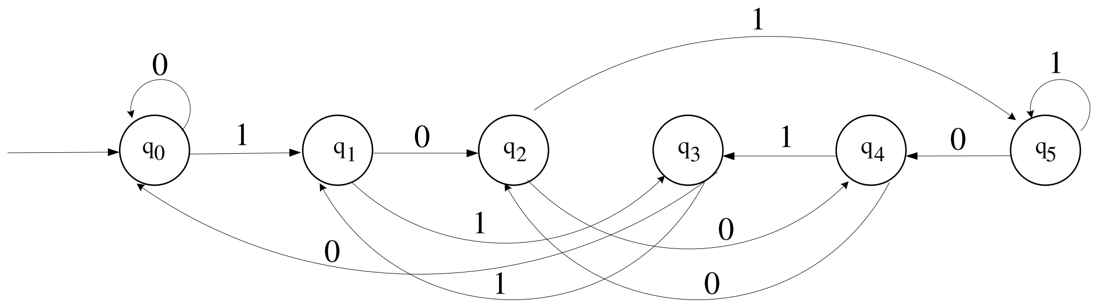
b.



c.



d.



Mesin Mealy

Bila *output* pada mesin Moore berasosiasi dengan *state*, maka *output* pada mesin Mealy akan berasosiasi dengan transisi. Mesin Mealy sendiri didefinisikan dalam 6 tupel, M = (Q, Σ, δ, S, Δ, λ), di mana :

Q = himpunan *state*

Σ = himpunan simbol *input*

δ = fungsi transisi

S = state awal

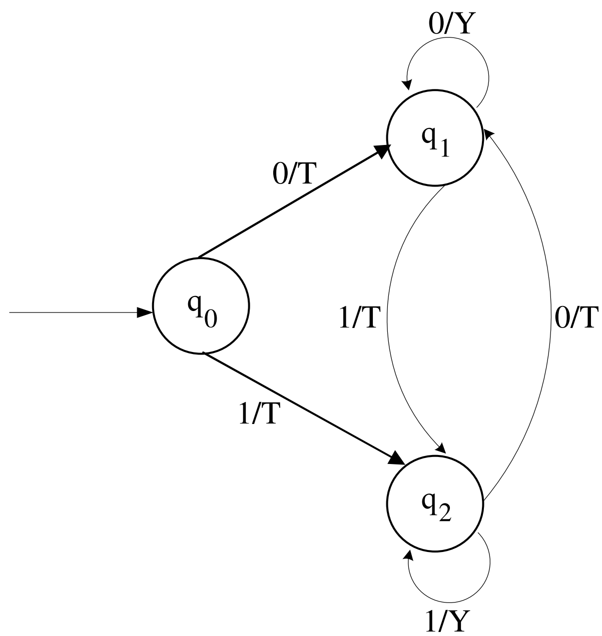
Δ = himpunan *output*

λ = fungsi *output* untuk setiap transisi

Contoh penerapan Mesin Mealy dapat dilihat pada gambar di bawah ini.

Mesin itu akan mengeluarkan *output* apakah menerima (Y) atau menolak (T), suatu masukan, di mana mesin akan mengeluarkan output 'Y' bila menerima untai yang memiliki akhiran 2 simbol berturutan yang sama.

Contoh input yang diterima : 01011, 01100, 1010100, 10110100, 00, 11, 100, 011, 000, 111, dll.



Konfigurasi dari mesin Mealy tersebut adalah sebagai berikut.

$Q = \{ q_0 , q_1 , q_2 \}$

$\Sigma = \{0,1\}$

$\Delta = \{Y, T\}$

$S = q_0$

$\lambda (q_0 , 0) = T$

$\lambda (q_0 , 1) = T$

$\lambda (q_1 , 0) = Y$

$\lambda (q_1 , 1) = T$

$\lambda (q_2 , 0) = T$

$\lambda (q_2 , 1) = Y$

Ekuivalensi Mesin Moore dan Mesin Mealy

Dari suatu mesin Moore dapat dibuat mesin Mealy yang ekuivalen, begitu juga sebaliknya. Untuk mesin Mealy pada contoh sebelumnya dapat kita buat mesin Moore yang ekuivalen. Bisa kita lihat bahwa *state* pada mesin Moore dibentuk dari kombinasi *state* pada mesin Mealy dan banyaknya *output*. Karena jumlah *state* pada mesin Mealy = 3 dan jumlah *output* = 2, maka jumlah *state* pada mesin Moore yang ekuivalen = 6.

Konfigurasi mesin Moore yang dibentuk adalah sebagai berikut.

$Q = \{ q_0Y, q_0T, q_1Y, q_1T, q_2Y, q_2T \}$

$\Sigma = \{0,1\}$

$\Delta = \{Y,T\}$

$S = q_0T$

$\lambda\ (q_0Y) = Y$

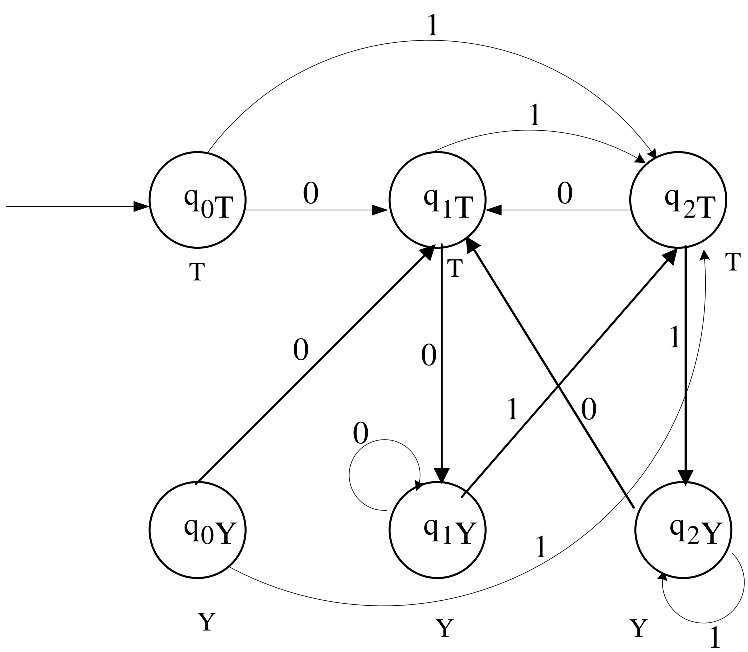
$\lambda\ (q_0T) = T$

$\lambda\ (q_1Y) = Y$

$\lambda\ (q_1T) = T$

$\lambda\ (q_2Y) = Y$

$\lambda\ (q_2T) = T$



Untuk memperoleh ekuivalensi mesin Mealy dari suatu mesin Moore caranya lebih mudah, cukup dengan menambahkan label *output* ke setiap transisi, dan menghapus label *output* pada setiap *state*. Gambar berikut merupakan mesin Mealy yang ekuivalen dengan mesin Moore.

Konfigurasi mesin Mealy tersebut adalah sebagai berikut.

$Q = \{ q_0, q_1, q_2 \}$

$\Sigma = \{0,1\}$

$\Delta = \{0, 1, 2\}$

$S = q_0$

$\lambda (q_0, 0) = 0$

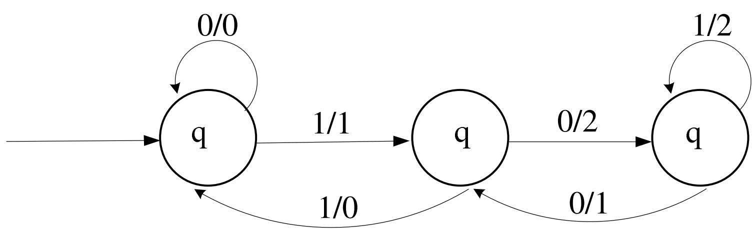
$\lambda (q_0, 1) = 1$

$\lambda (q_1, 0) = 2$

$\lambda (q_1, 1) = 0$

$\lambda (q_2, 0) = 1$

$\lambda (q_2, 1) = 2$



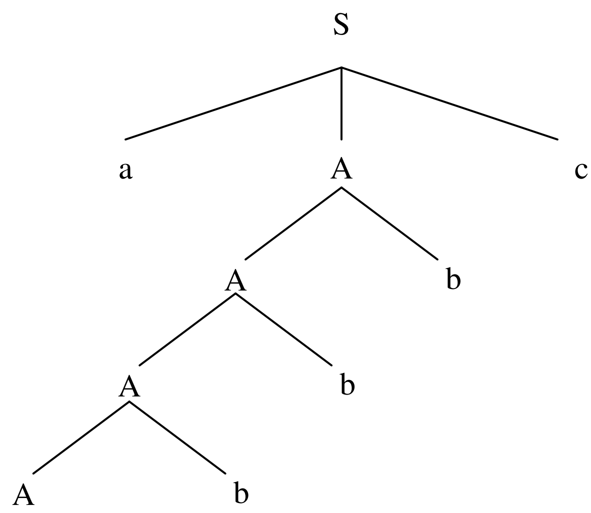
<u>Penghilangan Rekursif Kiri</u>

Aturan produksi yang rekursif memiliki ruas kanan (hasil produksi) yang memuat simbol variabel pada ruas kiri. Produksi yang rekursif kiri menyebabkan pohon penurunan tumbuh ke kiri. Dalam banyak penerapan tata bahasa, rekursif kiri tak diinginkan. Untuk menghindari penurunan yang bisa mengakibatkan *loop* kita perlu menghilangkan sifat rekursif kiri dari aturan produksi. Penghilangan rekursif kiri di sini memungkinkan tata bahasa bebas konteks nantinya diubah ke dalam *bentuk normal greibach*.

Contoh aturan produksi yang rekursif kiri :

$S \rightarrow aAc$

$A \rightarrow Ab \mid \epsilon$



Langkah – langkah penghilangan rekursif kiri adalah sebagai berikut.

- Pisahkan aturan produksi yang rekursif kiri dan yang tidak.
- Lakukan penggantian aturan produksi yang rekursif kiri, menjadi sebagai berikut.
  1) $A \rightarrow \beta_1 Z \mid \beta_2 Z \mid \ldots \mid \beta_m Z$
  2) $Z \rightarrow \alpha_1 \mid \alpha_2 \mid \alpha_3 \mid \ldots \mid \alpha_n$
  3) $Z \rightarrow \alpha_1 Z \mid \alpha_2 Z \mid \alpha_3 Z \mid \ldots \alpha_n Z$

  Penggantian di atas dilakukan untuk setiap aturan produksi dengan simbol ruas kiri yang sama, bisa muncul simbol variabel baru Z_1, Z_2, dan seterusnya, sesuai banyaknya variabel yang menghasilkan produksi yang rekursif kiri.
- Hasil akhir berupa aturan produksi pengganti ditambah dengan aturan produksi semula yang tidak rekursif kiri.

Contoh, tata bahasa bebas konteks :

$S \rightarrow Sab \mid aSc \mid dd \mid ff \mid Sbd$

Pertama – tama kita lakukan pemisahan aturan produksi.

Aturan produksi yang rekursif kiri :

$S \rightarrow Sab \mid Sbd$

Aturan produksi yang tidak rekursif kiri :

$S \rightarrow aSc \mid dd \mid ff$

Kita lakukan penggantian aturan produksi yang rekursif kiri.

$Z_1 \rightarrow ab \mid bd$

$Z_1 \rightarrow abZ_1 \mid bdZ_1$

Untuk aturan produksi yang tidak rekursif kiri diubah menjadi :

$S \rightarrow aScZ_1 \mid ddZ_1 \mid ffZ_1$

Hasil akhir setelah penghilangan rekursif kiri adalah sebagai berikut.

$S \rightarrow aSc \mid dd \mid ff$

$S \rightarrow aScZ_1 \mid ddZ_1 \mid ffZ_1$

$Z_1 \rightarrow ab \mid bd$

$Z_1 \rightarrow abZ_1 \mid bdZ_1$

Contoh lain, terdapat tata bahasa bebas konteks :

$S \rightarrow Sab \mid Sb \mid cA$

$A \rightarrow Aa \mid a \mid bd$

Pertama – tama kita lakukan pemisahan aturan produksi.

Aturan produksi yang rekursif kiri :

$S \rightarrow Sab \mid Sb$

$A \rightarrow Aa$

Aturan produksi yang tidak rekursif kiri :

$S \rightarrow cA$

$A \rightarrow a \mid bd$

Kita lakukan penggantian untuk aturan produksi yang rekursif kiri untuk yang memiliki simbol ruas kiri S.

$Z_1 \rightarrow ab \mid b$

$Z_1 \rightarrow abZ_1 \mid bZ_1$

Kita lakukan penggantian untuk aturan produksi yang rekursif kiri untuk yang memiliki simbol ruas kiri A.

$Z_2 \rightarrow a$

$Z_2 \rightarrow aZ_2$

Untuk aturan produksi yang tidak rekursif diubah menjadi.

$S \rightarrow cAZ_1$

$A \rightarrow a\ Z_2 \mid bdZ_2$

Hasil akhir setelah penghilangan rekursif kiri adalah sebagai berikut.

$S \rightarrow cA$

$A \rightarrow a \mid bd$

$S \rightarrow cAZ_1$

$Z_1 \rightarrow ab \mid b$

$Z_1 \rightarrow abZ_1 \mid bZ_1$

$A \rightarrow a\ Z_2 \mid bdZ_2$

$Z_2 \rightarrow a$

$Z_2 \rightarrow aZ_2$

Contoh lain, terdapat tata bahasa bebas konteks :

$S \rightarrow Sa \mid aAc \mid c \mid \epsilon$

$A \rightarrow Ab \mid ba$

Pertama – tama kita lakukan pemisahan aturan produksi.

Aturan produksi yang rekursif kiri.

$S \rightarrow Sa$

$A \rightarrow Ab$

Aturan produksi yang tidak rekursif kiri.

$S \rightarrow aAc \mid c \mid \epsilon$

$A \rightarrow ba$

Kita lakukan penggantian aturan produksi yang rekursif kiri untuk yang memiliki simbol ruas kiri S.

$Z_1 \rightarrow a$

$Z_1 \rightarrow a\ Z_1$

Kita lakukan penggantian aturan produksi yang rekursif kiri untuk yang memiliki simbol ruas kiri A.

$Z_2 \rightarrow b$

$Z_2 \rightarrow b\ Z_2$

Untuk aturan produksi yang tidak rekursif diubah menjadi :

$S \rightarrow aAcZ_1 \mid cZ_1 \mid Z_1$

$A \rightarrow baZ_2$

Hasil akhir setelah penghilangan rekursif kiri adalah sebagai berikut.

$S \rightarrow aAc \mid c \mid \epsilon$

$S \rightarrow aAcZ_1 \mid cZ_1 \mid Z_1$

$A \rightarrow ba$

$A \rightarrow baZ_2$

$Z_1 \rightarrow a$

$Z_1 \rightarrow a\ Z_1$

$Z_2 \rightarrow b$

$Z_2 \rightarrow b\ Z_2$

Soal :

1. Lakukan penghilangan rekursif kiri pada tata bahasa bebas konteks berikut.

   $A \rightarrow Aa \mid aBc$

2. Lakukan penghilangan rekursif kiri pada tata bahasa bebas konteks berikut.

   $A \rightarrow AbAB \mid \epsilon$

   $B \rightarrow Baa \mid A \mid \epsilon$

3. Lakukan penghilangan rekursif kiri pada tata bahasa bebas konteks berikut.

   $S \rightarrow Sba \mid Ab$

   $A \rightarrow Sa \mid Aab \mid a$

   $B \rightarrow Sb \mid Bba \mid b$

4. Lakukan penghilangan rekursif kiri pada tata bahasa bebas konteks berikut.

   $S \rightarrow SSC \mid SSB \mid abg$

$B \rightarrow abc \mid BSb \mid BCd$

$C \rightarrow ab$

Bentuk Normal Greibach

Bentuk Normal *Greibach* / *Greibach Normal Form* adalah suatu tata bahasa bebas konteks di mana hasil produksinya (ruas kanan) diawali dengan satu simbol terminal, selanjutnya bisa diikuti dengan rangkaian simbol variabel. Contoh tata bahasa bebas konteks dalam bentuk normal *Greibach*.

$S \rightarrow a \mid aAB$

$A \rightarrow aB$

$B \rightarrow cS$

Untuk dapat diubah ke dalam bentuk normal *Greibach*, tata bahasa semula harus memenuhi syarat – syarat sebagai berikut.

- Sudah dalam bentuk normal *chomsky*
- Tidak bersifat rekursif kiri
- Tidak menghasilkan ϵ

Secara umum langkah – langkah untuk mendapatkan *bentuk normal Greibach* adalah sebagai berikut.

1. Tentukan urutan simbol – simbol variabel yang ada dalam tata bahasa. Misalkan, terdapat *m* variabel dengan urutan $A_1, A_1, ..., A_m$
2. Berdasarkan urutan simbol yang ditetapkan pada langkah (1) seluruh aturan produksi yang ruas kanannya diawali dengan simbol variabel dapat dituliskan dalam bentuk :

   $A_h \rightarrow A_i\gamma$

   Di mana h <> i (rekursif kiri sudah dihilangkan), γ bisa berupa simbol – simbol variabel.

   a. Jika h < i, aturan produksi ini sudah benar (tidak perlu diubah).

   b. Jika h > i, aturan produksi belum benar. Lakukan substitusi berulang – ulang terhadap A_i (ganti A_i pada produksi ini dengan ruas kanan produksi dari variabel A_i) sehingga suatu saat diperoleh produksi dalam bentuk :

$A_h \rightarrow A_p\gamma$ (di mana $h \leq p$)

i) Jika h = p, lakukan penghilangan rekursif kiri.

ii) Jika h < p, aturan produksi sudah benar.

3. Jika terjadi penghilangan rekursif kiri pada tahap (2b), sejumlah simbol variabel baru yang muncul dari operasi ini dapat disisipkan pada urutan variabel semula di mana saja asalkan ditempatkan tidak sebelum A_h (di kiri).
4. Setelah langkah (2) & (3) dikerjakan, maka aturan – aturan produksi yang ruas kanannya dimulai simbol variabel seudah berada dalam urutan yang benar.

   $A_x \rightarrow A_y\gamma$ (di mana $x < y$).

   Produksi – produksi yang lain ada dalam bentuk :

   $A_x \rightarrow a\,\gamma$ (a = simbol terminal)

   $B_x \rightarrow \gamma$

   (B_x = simbol variabel baru yang muncul sebagai akibat dari operasi penghilangan rekursif kiri).
5. Bentuk normal *Greibach* diperoleh dengan cara melakukan substitusi mundur mulai dari variabel A_m , lalu A_{m-1} , A_{m-1} , ... Dengan cara ini aturan produksi dalam bentuk $A_x \rightarrow A_y\gamma$ dapat diubah sehingga ruas kanannya dimulai dengan simbol terminal.
6. Produksi yang dalam bentuk $B_x \rightarrow \gamma$ juga dapat diubah dengan cara substitusi seperti pada langkah (5).

Contoh (tata bahasa bebas konteks sudah dalam bentuk normal *Chomsky* dan memenuhi syarat untuk diubah ke bentuk normal *Greibach*), simbol awal adalah S :

$S \rightarrow CA$

$A \rightarrow a \mid d$

$B \rightarrow b$

$C \rightarrow DD$

$D \rightarrow AB$

Kita tentukan urutan simbol variabel, misalnya S, A, B, C, D (S<A<B<C<D).

Kita periksa aturan produksi yang simbol pertama pada ruas kanan adalah simbol variabel, apakah sudah memenuhi ketentuan urutan variabel :

- S → CA (sudah memenuhi karena S<C)
- C → DD (sudah memenuhi karena C<D)
- D → AB (tidak memenuhi karena D>A)

Yang belum memenuhi urutan yang telah kita tentukan adalah :

D → AB

Karena ruas kiri > simbol pertama pada ruas kanan. Maka kita lakukan substitusi pada simbol variabel A, aturan produksi menjadi :

D → aB | dB

Setelah semua aturan produksi sudah memenuhi ketentuan urutan variabel, kita lakukan substitusi mundur pada aturan produksi yang belum dalam bentuk normal *Greibach*.

- C → DD diubah menjadi C → aBD | dBD
- S → CA diubah menjadi S → aBDA | dBDA

Perhatikan bahwa substitusi mundur dimulai dari aturan produksi yang memiliki ruas kiri dengan urutan variabel paling akhir (kasus di atas : S<A<B<C<D, maka C lebih dulu disubstitusi daripada S).

Hasil akhir aturan produksi yang sudah dalam bentuk normal *Greibach* adalah sebagai berikut.

S → aBDA | dBDA

A → a | d

B → b

C → aBD | dBD

D → aB | dB

Contoh lain (simbol Awal A) :

A → BC

B → CA | b

C → AB | a

Kita tentukan urutan simbol : A, B, C (A<B<C).

A → BC (sudah memenuhi karena A<B)

B → CA (sudah memenuhi karena B<C)

C → AB (Padahal C>A sehingga harus diubah)

Yang belum memenuhi urutan yang telah kita tentukan adalah :

C → AB

Karena ruas kiri > simbol pertama pada ruas kanan. Maka kita lakukan substitusi pada simbol variabel A, aturan produksi menjadi :

C → AB diubah menjadi C → BCB diubah lagi menjadi C → bCB

Setelah semua aturan produksi sudah memenuhi ketentuan urutan variabel, kita lakukan substitusi mundur pada aturan produksi yang belum dalam bentuk normal *Greibach*.

- B → CA diubah menjadi B → bCBA
- A → BC diubah menjadi A → bCBAC

Hasil akhir aturan produksi yang sudah dalam bentuk normal *Greibach* adalah sebagai berikut.

A → bCBAC

B → bCBA | b

C → bCB | a

Soal :

1. Buatlah bentuk normal *Greibach* dari tata bahasa bebas konteks berikut (tata bahasa bebas konteks sudah dalam bentuk normal *Chomsky* dan memenuhi syarat untuk diubah ke GNF).

   S → AS | a

   A → a

2. Buatlah bentuk normal *Greibach* dari tata bahasa bebas konteks berikut.

   S → AA | d

   A → SS | b

3. Buatlah bentuk normal *Greibach* dari tata bahasa bebas konteks berikut.

   S → a | AS | AA

   A → SA | a

Jawab :

1.

   Kita tentukan urutan simbol : S, A (S<A).

   S → AS (sudah memenuhi karena S<A)

   Setelah semua aturan produksi sudah memenuhi ketentuan urutan variabel, kita lakukan substitusi mundur pada aturan produksi yang belum dalam bentuk normal *Greibach*.

   - S → AS diubah menjadi S → aS

   Hasil akhir aturan produksi yang sudah dalam bentuk normal *Greibach* adalah sebagai berikut.

      S → aS | a

      A → a

2.

   Kita tentukan urutan simbol : S, A (S<A).

   S → AA (sudah memenuhi karena S<A)

   A → SS (tidak memenuhi karena A>S)

Yang belum memenuhi urutan yang telah kita tentukan adalah :

A → SS

Karena ruas kiri > simbol pertama pada ruas kanan. Maka kita lakukan substitusi pada simbol variabel S, aturan produksi menjadi :

A → SS diubah menjadi A → dS

Setelah semua aturan produksi sudah memenuhi ketentuan urutan variabel, kita lakukan substitusi mundur pada aturan produksi yang belum dalam bentuk normal *Greibach*.

- S → AA diubah menjadi S → dSA

Hasil akhir aturan produksi yang sudah dalam bentuk normal *Greibach* adalah sebagai berikut.

S → dSA | d

A → dS | b

3.

Kita tentukan urutan simbol : S, A (S<A).

S → AS (sudah memenuhi karena S<A)

S → AA (sudah memenuhi karena S<A)

A → SA (tidak memenuhi karena A>S)

Yang belum memenuhi urutan yang telah kita tentukan adalah :

A → SA

Karena ruas kiri > simbol pertama pada ruas kanan. Maka kita lakukan substitusi pada simbol variabel S, aturan produksi menjadi :

A → SA diubah menjadi A → aA

Setelah semua aturan produksi sudah memenuhi ketentuan urutan variabel, kita lakukan substitusi mundur pada aturan produksi yang belum dalam bentuk normal *Greibach*.

- $S \rightarrow AS$ diubah menjadi $S \rightarrow aAS$
- $S \rightarrow AA$ diubah menjadi $S \rightarrow aAA$

Hasil akhir aturan produksi yang sudah dalam bentuk normal *Greibach* adalah sebagai berikut.

$S \rightarrow a \mid aAS \mid aAA$

$A \rightarrow aA \mid a$