التحويل من RG إلى FA :

تستخدم القواعد المنتظمة لتوليد كلمة.

1. عدد حالات الأوتومات = عدد ال variables + 1 هي الحالة النهائية.

2. يوجد حالة مطابقة لكل رمز وحالة إضافية هي الحالة النهائية للأوتومات.

3. الحالة البدائية تطابق ال start symbol.

4. إذا كانت اللغة تحوي $\{\varepsilon\}$ فإن الحالة البدائية هي نهائية.

5. الإنتقالات: سيكون شكل الإنتقالات مثلاَ:

A →aB تعني الانتقال من الحالة A إلى الحالة B عند الرمز a أي $\delta(A,a)=B$

وإذا كان هناك قاعدة A→a أي أن من الحالة A تصل للحالة النهائية عند الرمز a أي $\delta(A,a)=finit\_state$ نلاحظ ان الحالة النهائية لا تظهر في هذا الشكل .

أشكال القواعد: A →aB أو B →b

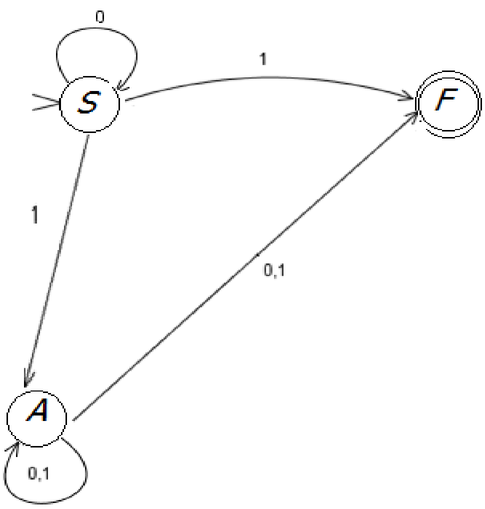
مثال 1:

إذا كان لدينا RG التالي:

S → 0S | 1A | 1

A →0A | 1A | 0 | 1

يكون مخطط NFA :



مثال 2:

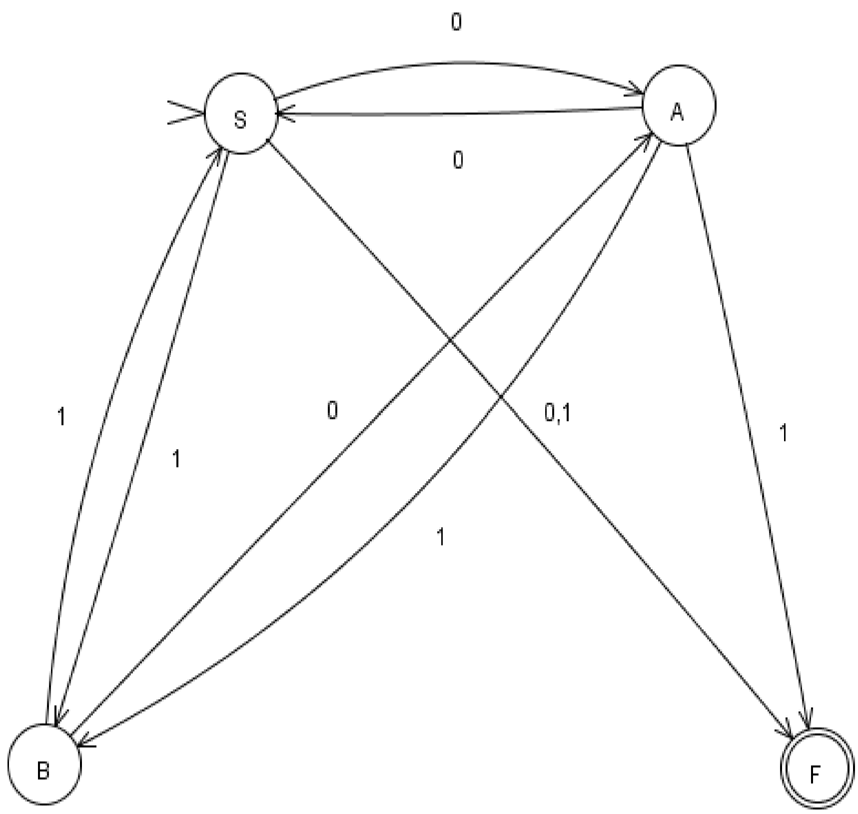
إذا كان لدينا RG التالي:

S → 0A | 1B | 0 | 1

A →0S | 1B | 1

B → 0A | 1S

يكون مخطط NFA :



S=0A+1B+0+1

A=0S+1B+1

B=0A+1S

نلاحظ أننا انتقلنا من الحالة البدائية مثلا للحالة النهائية عند 0,1 وبالعبارة السابقة لم نضع الحالة النهائية وإنما أكتفينا فقط بوضع الرمز الذي أوصلنا للحالة النهائية. بتعويض المعادلات السابقة في معادلة S يتكون لدينا RG الكامل.

التحويل من NFA إلى DFA:

	0	1
S	{A,F}	{B,F}
*{A,F}	{S}	{B,F}
{B,F}	{A}	{S}
{A}	{S}	{B,F}

ونكمل بالطريقة التي نعرفها...

التحويل من FA إلى RG:

1. إسناد متحولات لكل حالة (أي تصبح كل حالة متحول).

2. بعكس العملية السابقة نحول من $\delta(A,a)=B$ إلى A→aB إذا كانت B حالة عادية أما إذا كانت B حالة نهائية يكون :

A →a (قواعد بلا ε)

أو

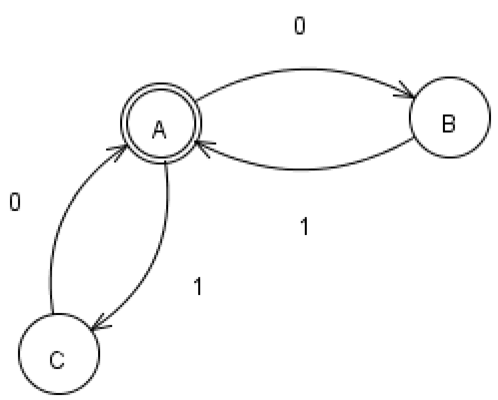
A →aB و B →ε (قواعد مع ε)

نلاحظ هنا أن نولد نوعين من الاوتومات الاول لا يقبل ε والثاني يقبل ε والسبب أننا في الحالة الأولى نطبق ضمنا مايسمى بحذف ال ε كما يلي :

$L(G) \implies L(G1)=L(G)-\{\varepsilon\}$

مثال:

لدينا الأوتومات التالي:



يتم تحويله إلى RG بالشكل:

White Hats
ITE 2016

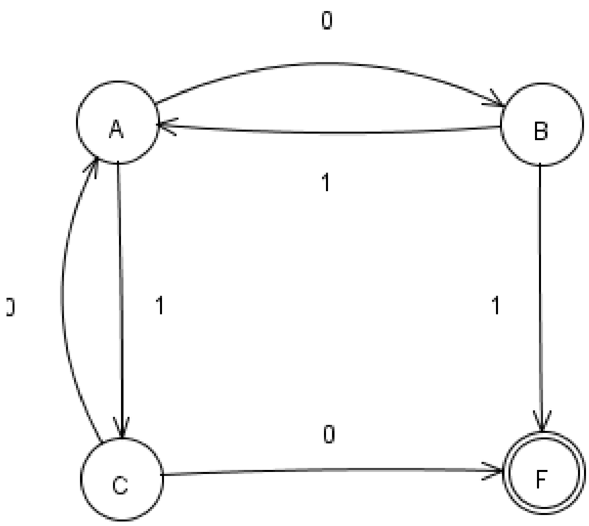
$A \rightarrow 0B\,|\,1C\,|\,\varepsilon$

$B \rightarrow 1A$

$C \rightarrow 0A$

نلاحظ أننا حصلنا على الشكل: $A \rightarrow aB$ و $B \rightarrow \varepsilon$ حيث B تقابل هنا الحالة A وهي الحالة النهائية.

مثال:



يتم تحويل هذا الأوتومات إلى RG بالشكل:

$A \rightarrow 1B\,|\,1C$

$B \rightarrow 1A\,|\,1$

$C \rightarrow 0A\,|\,0$

نلاحظ أننا حصلنا على الشكل: $A \rightarrow a$ حيث تقابل B هنا الحالة F الحالة النهائية.

الفرق بين المثالين السابقين هو أننا فصلنا الحالة النهائية لنضعها في حالة منفصلة عن A وهي الحالة F .

RG هي حالة خاصة من *Context free grammar*.

مثال1:

حول القاعدة المنتظمة التالية إلى أوتومات منتهي:

G1 : A → 0B | 1E

B → 0A | 1F | ε

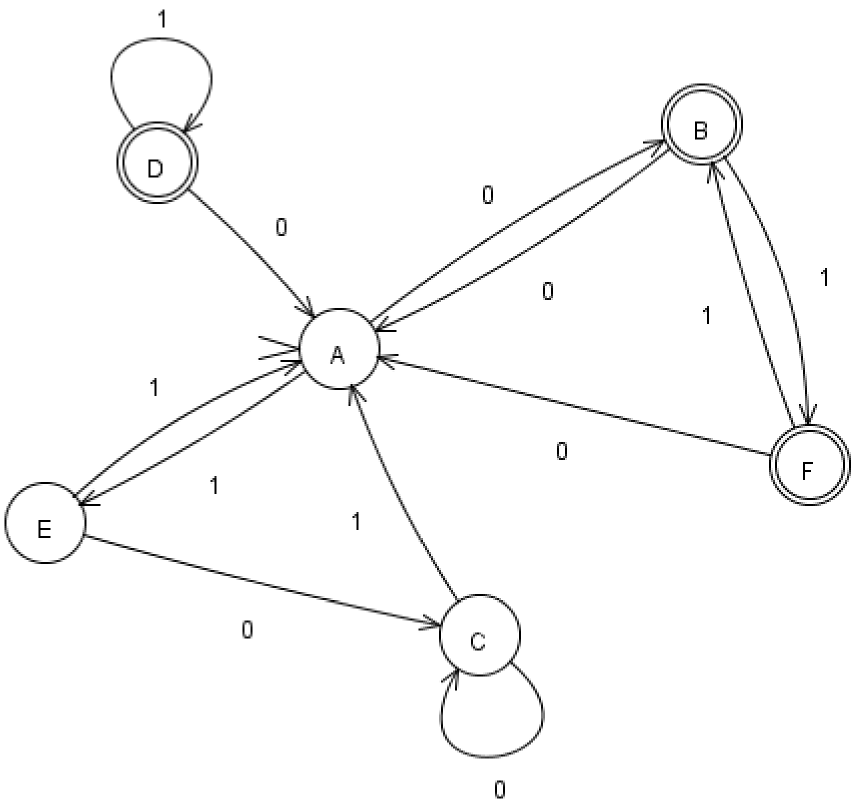
C → 0C | 1A

D → 0A | 1D | ε

E → 0C | 1A

F → 0A | 1B | ε

كل الحالات التي تحوي ε هي حالة نهائية



الآن نقوم بعملية أختصار الحالات التي لا يمكن الوصول إليها...
نجد أن الحالة D لا يمكننا الوصول إليها من أي حالة (inaccessible) وبالتالي يمكننا حذفها.
بتذكرو بالمحاضرة الخامسة شو عملنا لنختصر الحالات المتشابهة؟ هلأ لح نعمل نفس الشي هون ☺

A C E	B F

A,0→ B

C,0 → C

E,0 → C

A	C E	B F

C,1→ A

E,1→ A

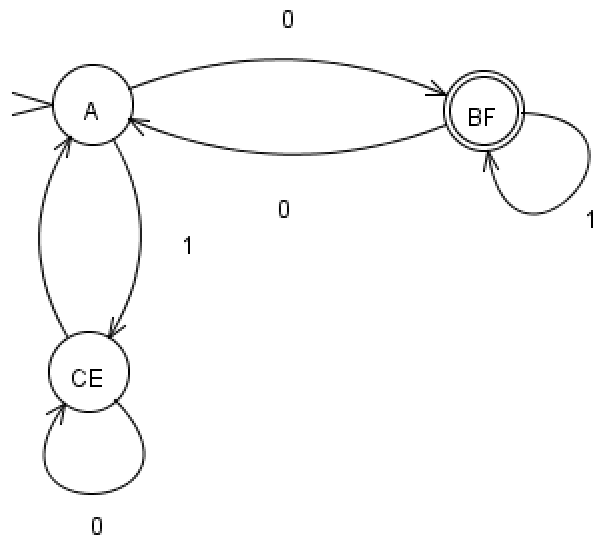
B,0→ A

F,0→ A

B,1→ F

F,1→ B

نجد أنه من المستحيل اختصار حالات أكثر فيصبح الأوتومات:



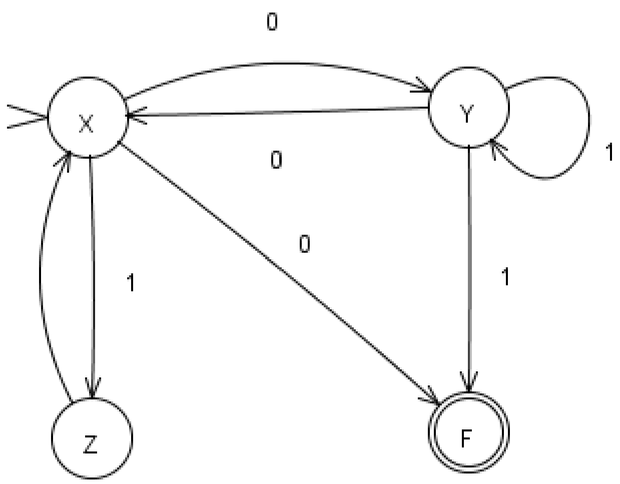
مثال2:

حول القاعدة المنتظمة التالية إلى أوتومات منتهي:

G2: $X \rightarrow 0Y|0|1Z$

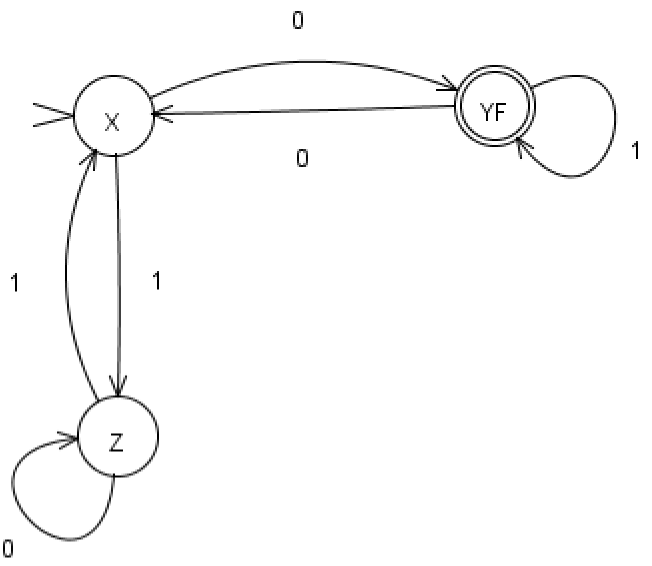
$Y \rightarrow 0X|1Y|1$

$Z \rightarrow 0Z|1X$



نظرا لكون y عند ال 1 تنتقل الى {Y,F} (أي انتقال من حالة الى حالتين عند نفس الرمز) يكون هذا الأوتومات NFA بالطريقة التالية نحوله ل DFA:

	0	1
X	{Y,F}	{Z}
*{Y,F}	{X}	{Y,F}
{Z}	{Z}	{X}



ونستنتج أن G1, G2 متكافئتان لأن لهما نفس ال DFA .

إذا كانت DFA لقاعدتين منتظمتين متساوية فهاتين القاعدتين متكافئتين لأنهما تولدان نفس اللغة وإذا لم تكن الأوتوماتين DFA متساويين يعني أن القاعدتين تولدان لغتين مختلفتين.

خصائص RL:

نقول أن مجموعة ما (لغة منتظمة) مغلقة على عملية ما إذا كان ناتج تطبيق العملية على عنصرين من نفس المجموعة يبقى عنصر من المجموعة نفسها.

الأعداد الطبيعية مغلقة على عملية الجمع لأن الناتج حتماً عدد طبيعي ولكن الأعداد الطبيعية غير مغلقة على عملية الطرح لأن الناتج من الممكن أن يكون سالب وغير طبيعي وكذلك الأمر لعملية القسمة.

CFL	RL	
Y	Y	الاجتماع
Y	Y	الوصل
Y	Y	Kleen_star
N	Y	Intersection
N	Y	Complements
N	Y	Reverse

أي اللغات المنتظمة مغلقة على عمليات الاجتماع والوصل وال * والتقاطع والفرق والمعكوس بينما context free Language غير مغلقة بالنسبة لكل هذه العمليات.

عملية غير مغلقة على RL لا يمكنني تمثيلها بأوتومات.

قاعدة منتظمة U لغة منتظمة >>> أوتومات.

مسألة الطيارات والدفاع الجوي :

لنفرض أن الطائرات الصديقة ترسل كلمات يستطيع نظام الدفاع الجوي التعرف عليها وذلك عن طريق أوتومات تتعرف على هذه الكلمات مثلا كلمات تحوي 101 وتنتهي ب 000ولكن لنفرض أن الطائرة ولدت كلمة فيها 000 قبل أن تولد 101 عندها سيعتبرها الدفاع الجوي طائرة معادية وذلك لأن الأوتومات اعتبرت أن الكلمة أنتهت ولم تحوي 101 لذلك يجب أن يكون للطائرات قواعد منتظمة تحوي الأوتومات A000,A101,A'000 حيث A'000 أوتومات يتعرف على السلاسل التي لا تحوي 000 يعني لازم يكون عنا سلسلة لا تحوي 000 وبعدها سلسلة تحوي 101 وبعدها سلسلة تنتهي ب 000 لذلك ستكون الأوتومات اللازمة للتعرف على هذه السلاسل هي :

نضمن أن الكلمة لاتحوي 000 كما يلي :

$(A_{101} \cap \overline{A_{000}})$

نضمن ان الكلمة تنتهي ب 000 كما يلي :

$(A_{101} \cap \overline{A_{000}}) \cup A_{000}$

وهذا ليس بالحل الصحيح وانما مجرد حل مبدئي ☺

اللغات خارج السياق Context Free Grammars :

نستخدمPush – Down Automata لتمثيل اللغات خارج السياق .
تستخدم هذه الأوتومات في تمثيل السلاسل:

$a^n b^n$ حيث لا يمكننا معرفة حالات فمن الممكن أن تكون لا نهائية.

إذا أتى الكلمة ومعكوسها في نفس السلسلة تسمى هذه الحالة WW^R palindrome
string مثلاً 1010 0101, 1110 0111 , 10 01.

P → 0p0 | 1p1 | ε

لا يوجد لدينا حل لهذه المشكلة إلا بعملية الضخ ☺.

Stack+ ε -NFA → PDA

كيف فينا نعرف بالحالة التالية بهالطريقة؟؟
بمعرفتنا بالحالة الحالية و بقيمة قمة المكدس والرمز الذي معنا يمكننا تحديد الحالة التالية كما سنرى ;)

ليكن لدينا الكلمة: (q0,1111,Z0) حيث q0 تعبر عن الحالة الحالية وZ0 هي رمز خاص لقاع المكدس. نلاحظ أن هذه الكلمة palindrome لذلك نحتاج للضخ في المكدس.

يتم قبول الكلمة في هذا الأوتومات عندما يصبح المكدس فارغ وتنتهي الكلمة في نفس الوقت لايهمني أي حالة هي الحالة النهائية حيث في كل مرحلة لدينا خيارين إما أن ندخل رمز جديد للمكدس أو أن نقارن قمة المكدس مع الرمز الجديد والقيام بإزالة رأس

المكدس مع حذف الرمز المقارن به من الكلمة وهكذا حتى نصل لمكدس فارغ مع انتهاء رموز الكلمة وبهذه المقارنة نأكد أن السلسلة تحوي كلمة ومعكوسها.

شوفو شلون :

$(q_0,1111,z_0)$

- $(q_1,1111,z_0) \rightarrow (q_2,1111,z_0)X$ يموت لأننا انتقلنا لحالة نهائية
- $(q_0,111,1z_0)$

$(q_0,111,1z_0)$

- $(q_1,111,1z_0) \rightarrow (q_1,11,z_0) \rightarrow (q_2,11,z_0)X$ يموت لأننا انتقلنا لحالة نهائية
- $(q_0,11,11z_0)$

$(q_0,11,11z_0)$

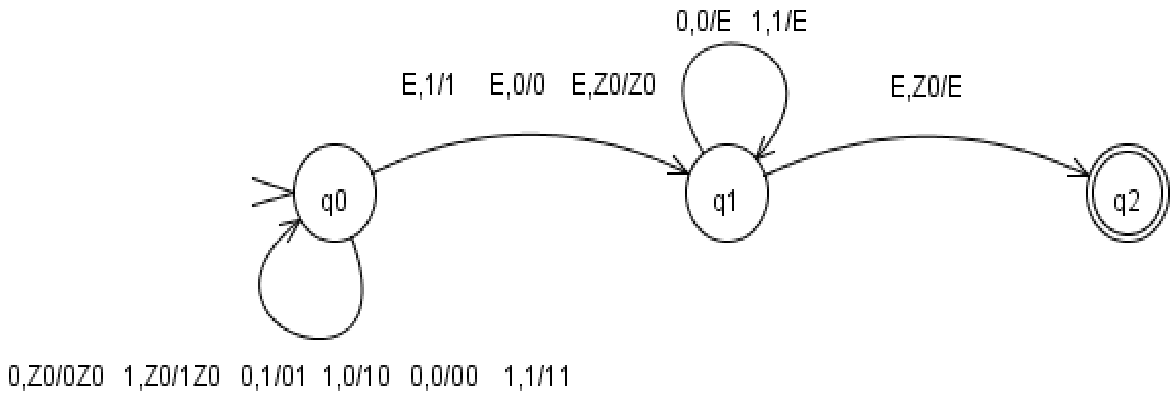
- $(q_1,11,11z_0) \rightarrow (q_1,1,1z_0) \rightarrow (q_1,\varepsilon,z_0) \rightarrow (q_2,\varepsilon,z_0)$
- $(q_0,1,111z_0)$

$(q_0,1,111z_0)$

- $(q_1,1,111z_0) \rightarrow (q_1,\varepsilon,11z_0)X$ المكدس ممتلئ و ينتهي الدخل
- $(q_0,\varepsilon,1111z_0)X$ المكدس ممتلئ و ينتهي الدخل

بأول خط عم نقارن قيمة رأس المكدس مع رمز الدخل وبتاني خط عم نأسند للمكدس الرمز .
ويبرجع تاني خط منعمل عليه العمليتين الأولى تبع المقارنة والتانية تبع الإسناد وهيييييييك حتى تخلص الكلمة لازم يطلع معنا أحد الفروع أنو الكلمة palindrome وهون الفرع يلي عليه سهم هو أكدلنا أنو الكلمة تحوي كلمة ومعكوسها لانو فضي المكدس والكلمة بنفس الوقت.
ملاحظة:الدكتور رسم هي الرسمة على شكل شجرة ثنائية بس تعذر علينا رسمها ما تواخذونا ☹

شكل الأوتومات :



الشكل: بيعطينا/ قمة المكدس,الرمز

نجد PDA في compiler في syntax.

انتهت المحاضرة ...

R & H