ELiC11: Transductores

Clase 2: Autómatas y
Transductores Finitos

Carlos Areces
carlos.areces@gmail.com

La clase pasada

- Que es LC
- Lenguajes Regulares
  - Expresiones Regulares
  - Autómatas Finitos
  - Construcción de Thompson

La clase de hoy

- Repaso
- Autómatas Finitos
  - Operaciones sobre Autómatas
    - Granularidad
    - Eliminación de ϵ
    - Inversión derecha-izquierda
    - Determinización
    - Minimización
- Transductores Finitos

Repaso

Expresiones regulares

- Notación para expresiones regulares
  - Cadena vacía: ϵ
  - Lenguajes unitarios: a
  - Unión: $x|y$
  - Concatenación: xy
  - Iteración: x^*
- Ejemplo (alfabeto = {a, ..., z, ▫}

$$((be \mid it \mid let)\,▫\,)^* \ni let\,▫\,it\,▫\,be\,▫$$

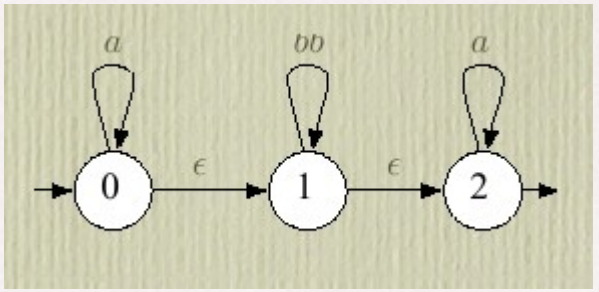
Autómatas de estados finitos

- Un autómata de estados finitos (FSA) es una tupla <Q,Σ,Δ,q_0> donde
  - Q es un conjunto finito de estados
  - Σ es un conjunto finito de **símbolos terminales**, (que no incluye el símbolo especial #).
  - q_0 es el **estado inicial**.
  - Δ es la **función de transición**.

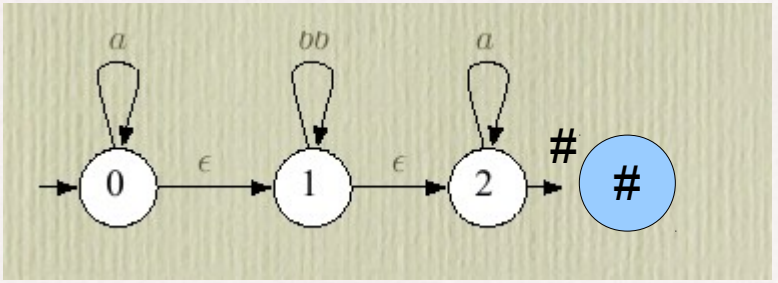
La función de transición

- Sea Σ^* el conjunto de todas las cadenas sobre Σ; la notación $\Sigma^{\#}$ representa el conjunto de cadenas de Σ^* con un símbolo # opcional al final.
- $\Delta \in Q \times \Sigma^{\#} \times (Q \cup \{\#\})$ es un conjunto de transiciones, cada una linqueando un estado **fuente** (de Q) y un estado **destino** (de $Q \cup \{\#\}$) mediante una cadena (quizás vacía) de $\Sigma^{\#}$, y tal que el estado destino es # sii la cadena termina con #
- Estados con transiciones a # se llaman **estados finales**.

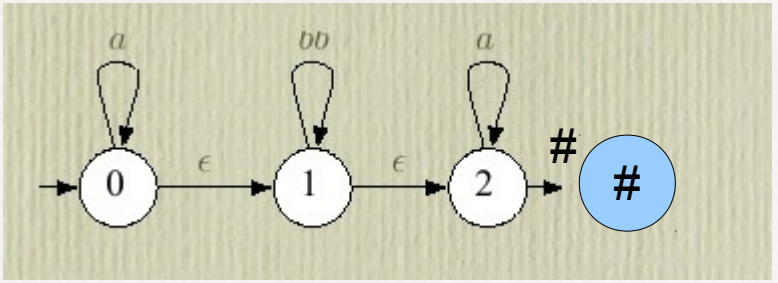
Derivaciones y cadenas aceptadas



Derivaciones y cadenas aceptadas



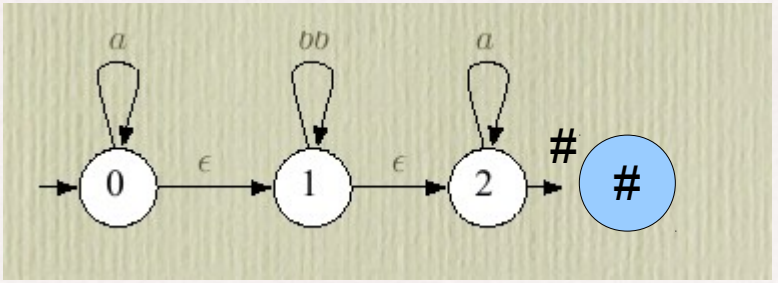
Derivaciones y cadenas aceptadas



Función de transición

$$q_0\, a \vdash q_0 \qquad q_0 \vdash q_1$$
$$q_1\, bb \vdash q_1 \qquad q_1 \vdash q_2$$
$$q_2\, a \vdash q_2 \qquad q_2\, \# \vdash \#$$

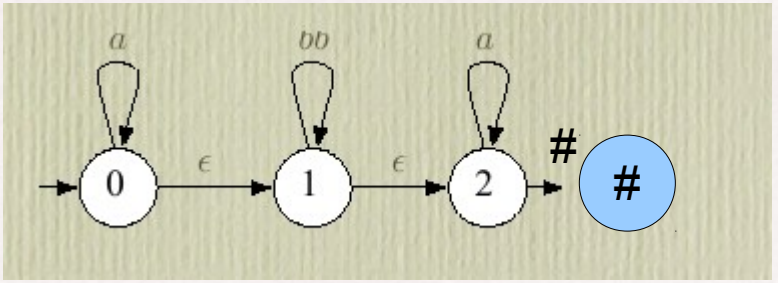
Derivaciones y cadenas aceptadas



Reglas de reescritura

$$q_0\, a \vdash q_0 \qquad q_0 \vdash q_1$$
$$q_1\, bb \vdash q_1 \qquad q_1 \vdash q_2$$
$$q_2\, a \vdash q_2 \qquad q_2\, \# \vdash \#$$

Derivaciones y cadenas aceptadas



Reglas de reescritura

$$q_0\, a \vdash q_0 \qquad q_0 \vdash q_1$$
$$q_1\, bb \vdash q_1 \qquad q_1 \vdash q_2$$
$$q_2\, a \vdash q_2 \qquad q_2\, \# \vdash \#$$

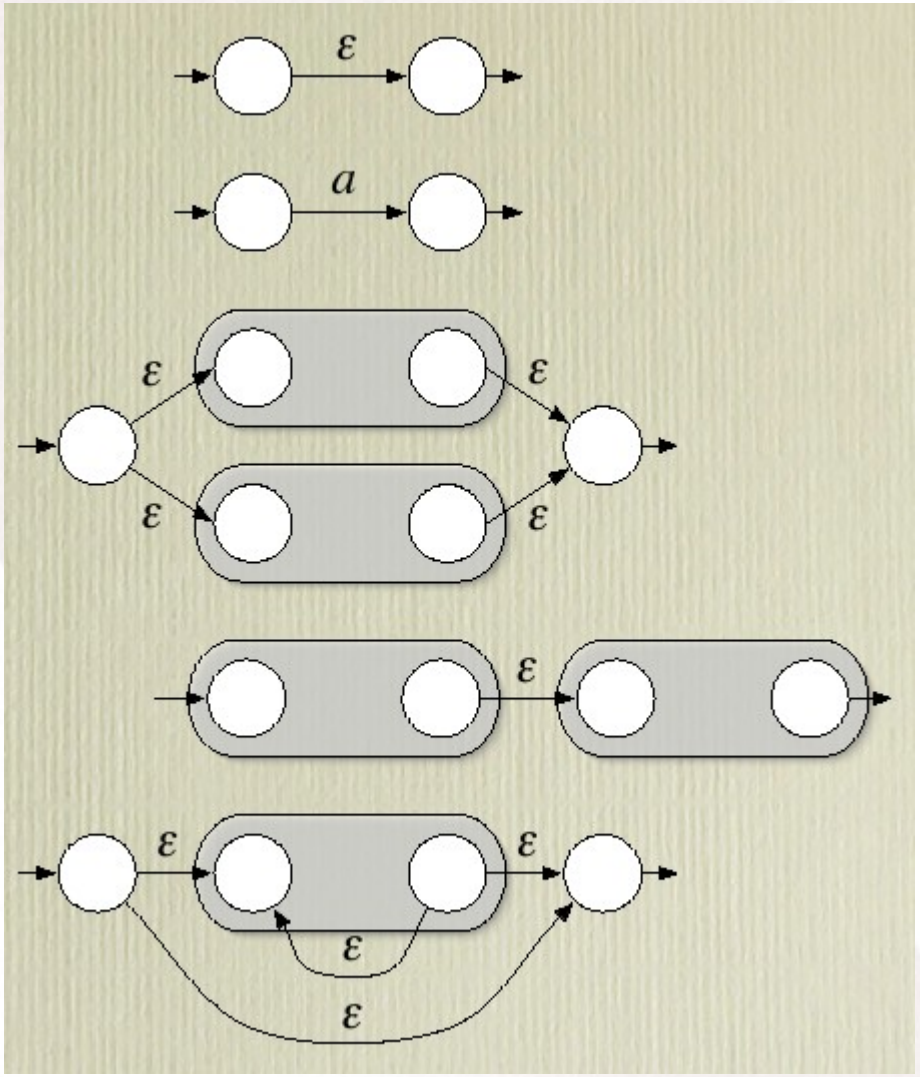
Una derivación:

$$q_0 aaabba\# \vdash q_0 aabba\# \vdash q_0 abba\# \vdash$$
$$\vdash q_0 abba\# \vdash q_1 bba\# \vdash q_1 a\#$$
$$\vdash q_2 a\# \vdash q_2\# \vdash \#$$

Def. de aceptación: aceptar w sii $q_0 w\# \vdash^* \#$

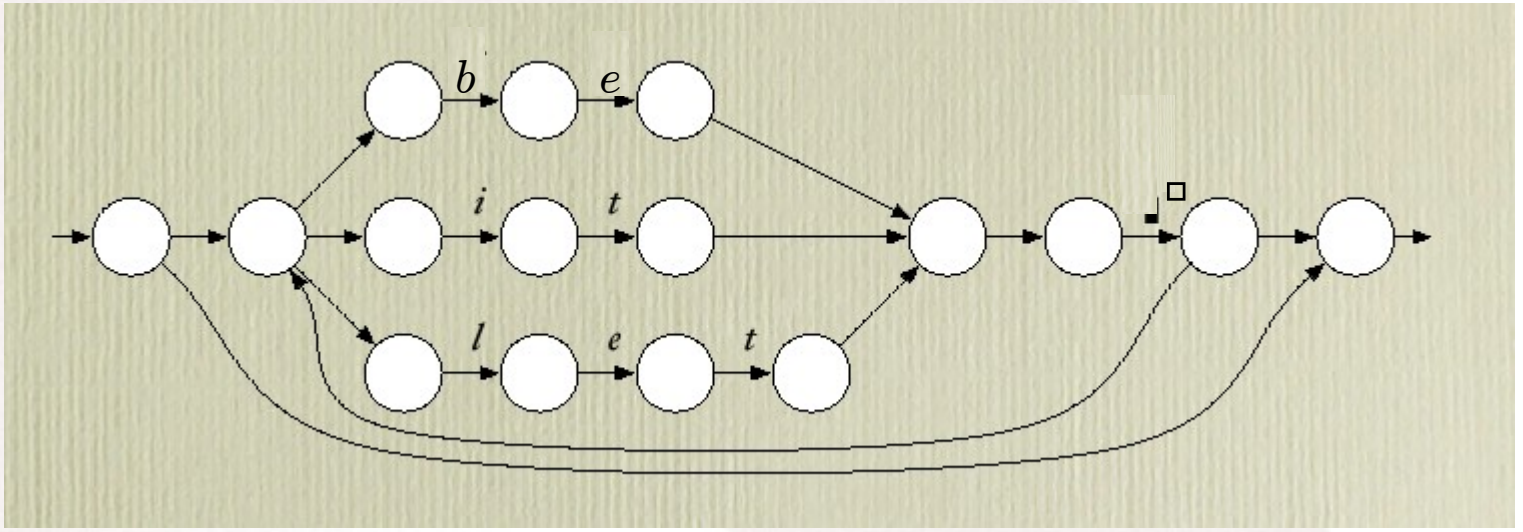
De expresiones regulares a automatas

Construcción de Thompson



Let it be

$$((be \mid it \mid let)\square)^*$$



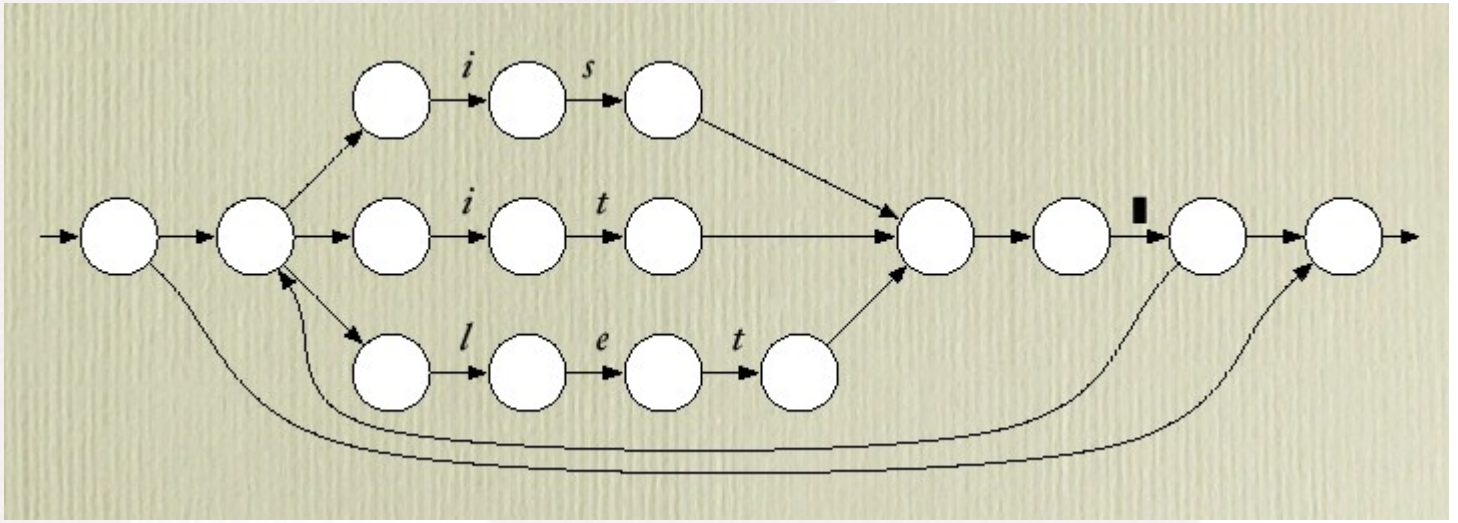
Fin del repaso

Operaciones sobre autómatas

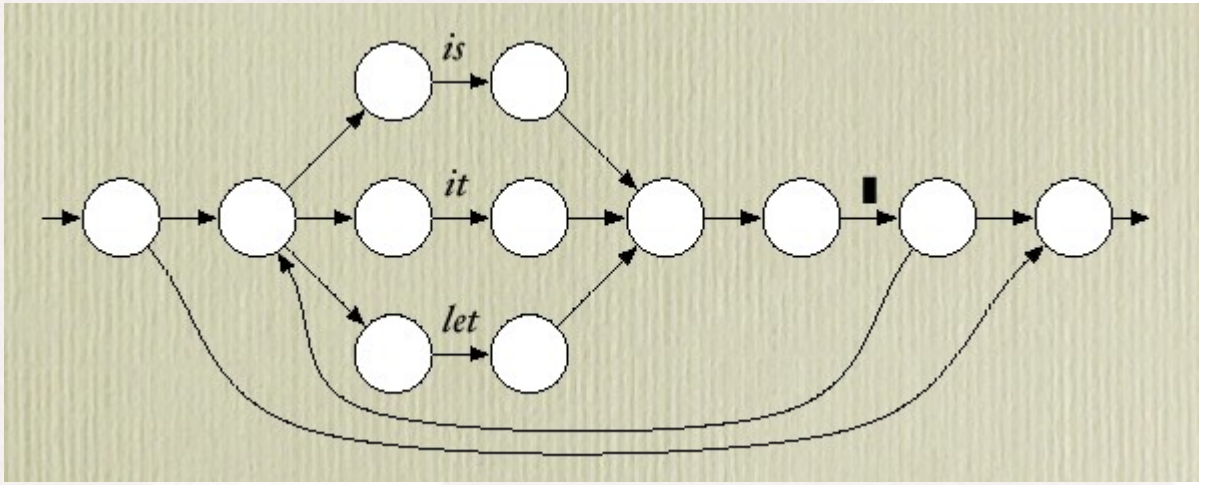
- Existen diferentes operaciones sobre autómatas que preservan el lenguaje del autómata (o lo modifican de forma controlada)
  - Granularidad
  - Eliminación de ϵ
  - Reversión derecha-izquierda
  - Determinización
  - Minimización

Granularidad

Granularidad del input

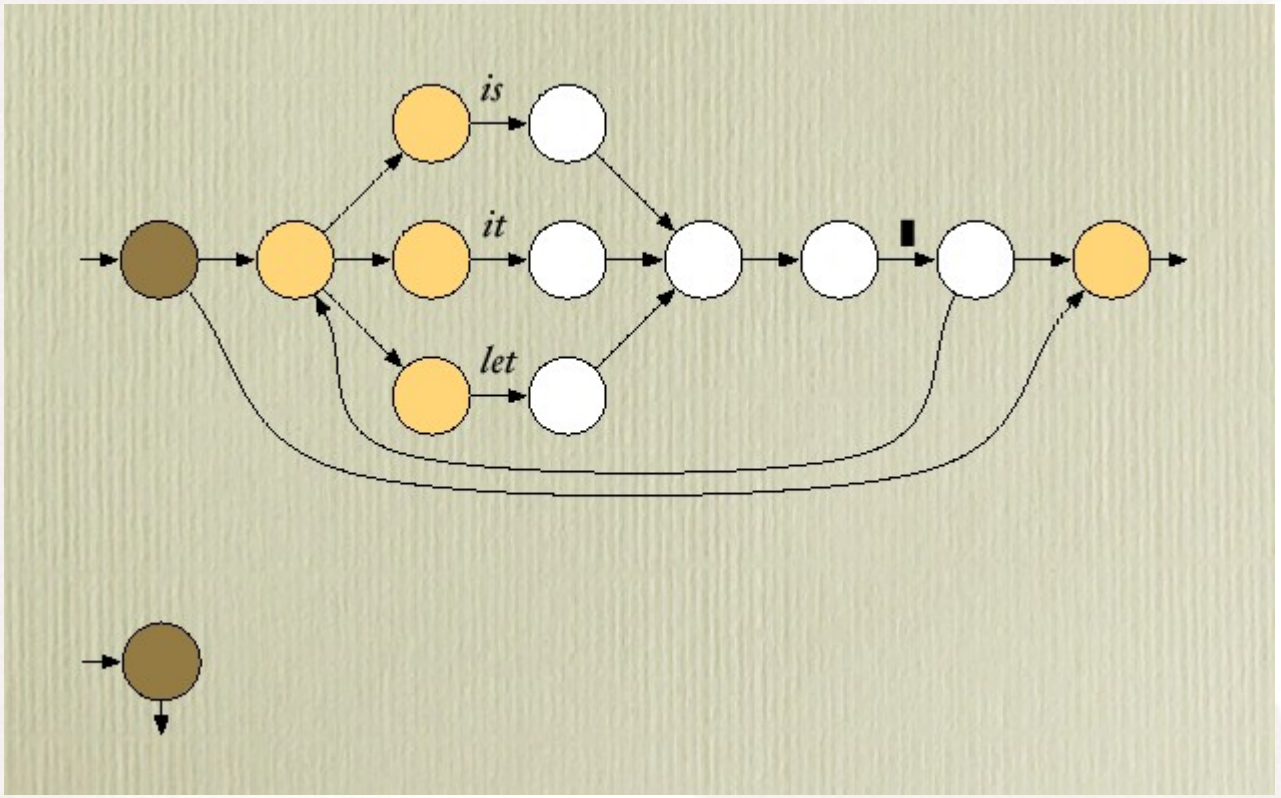


Granularidad del input

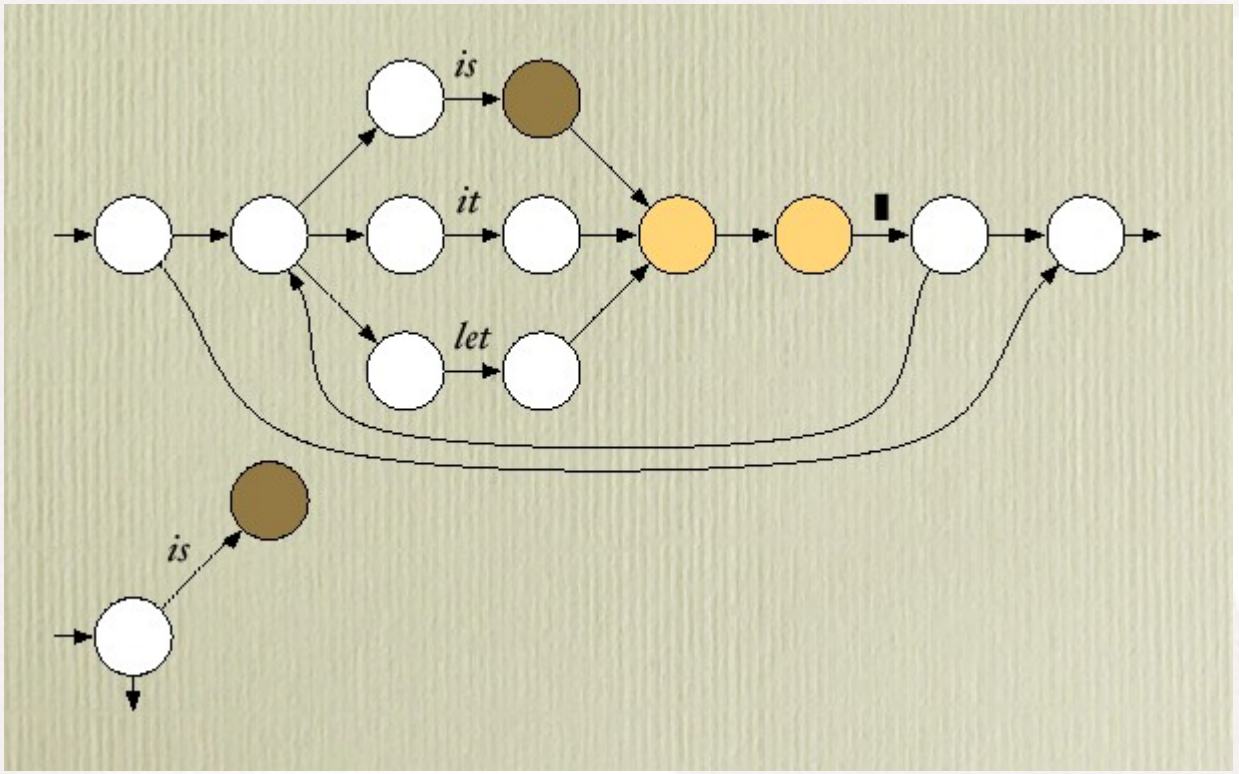


Eliminación de ϵ

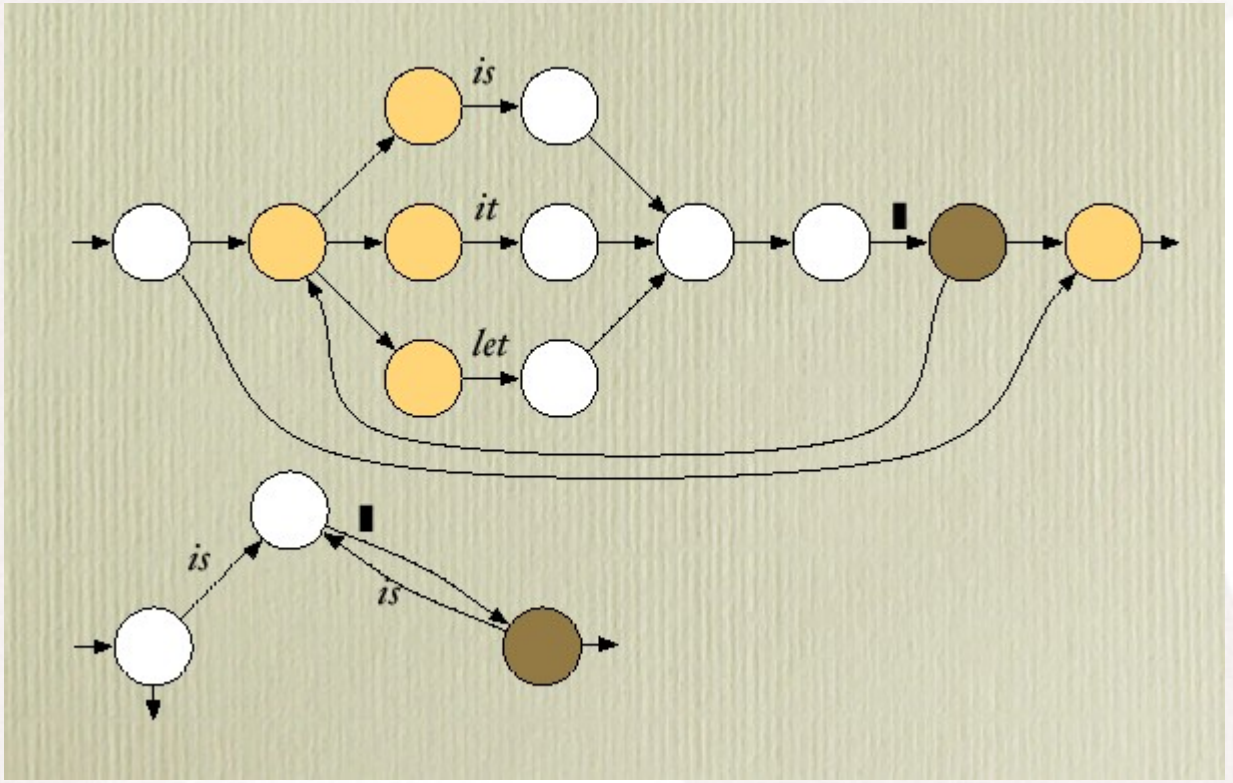
Eliminación de ϵ



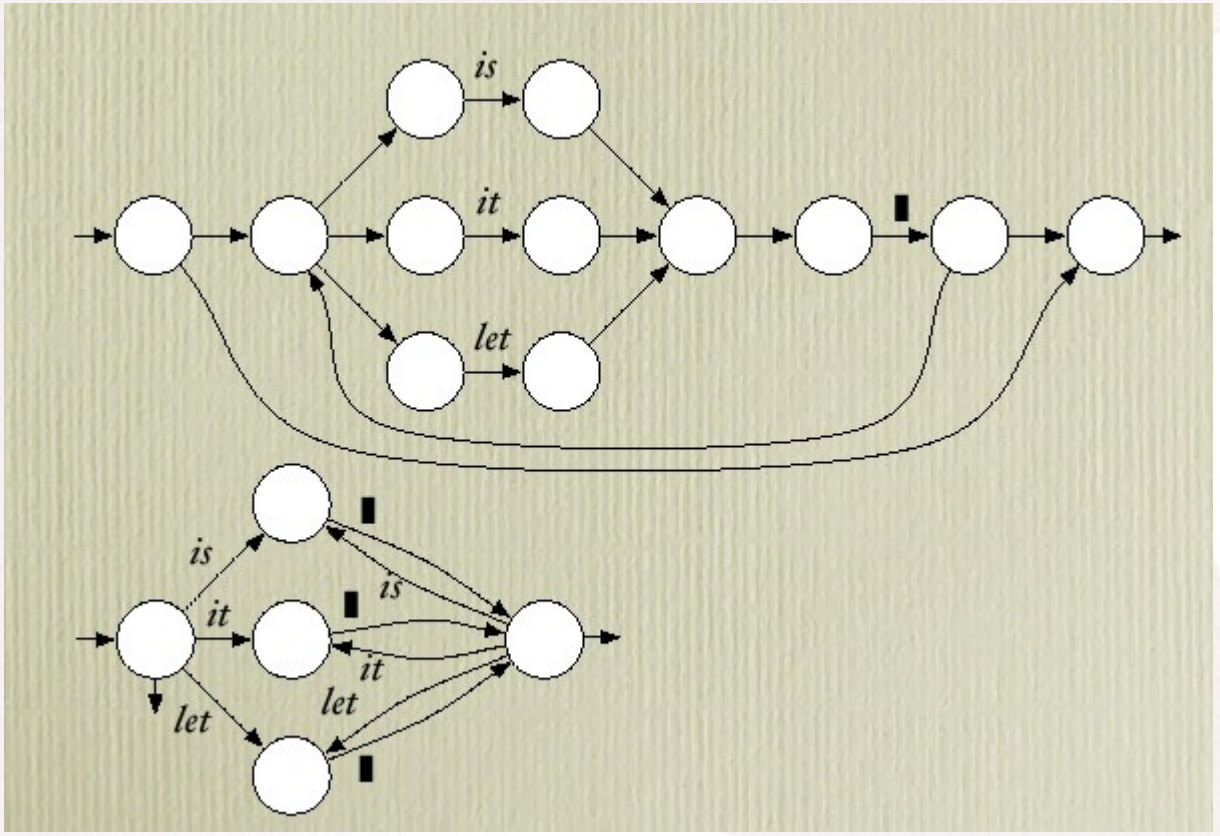
Eliminación de ϵ



Eliminación de ϵ

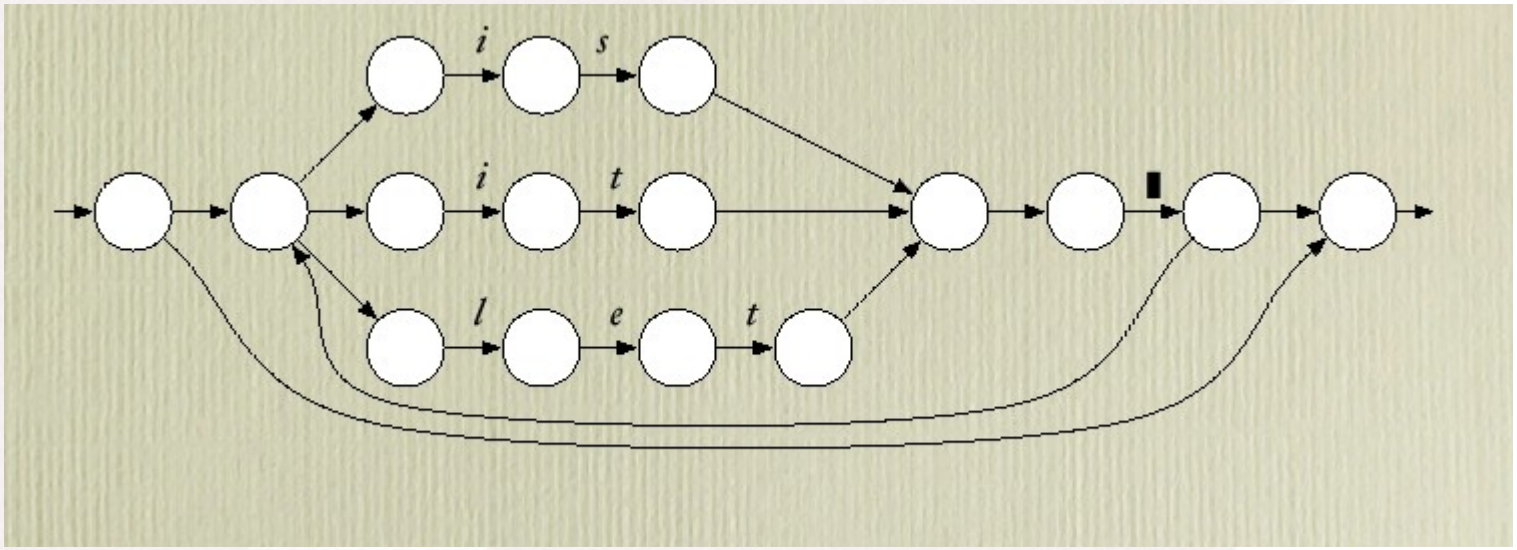


Eliminación de ϵ

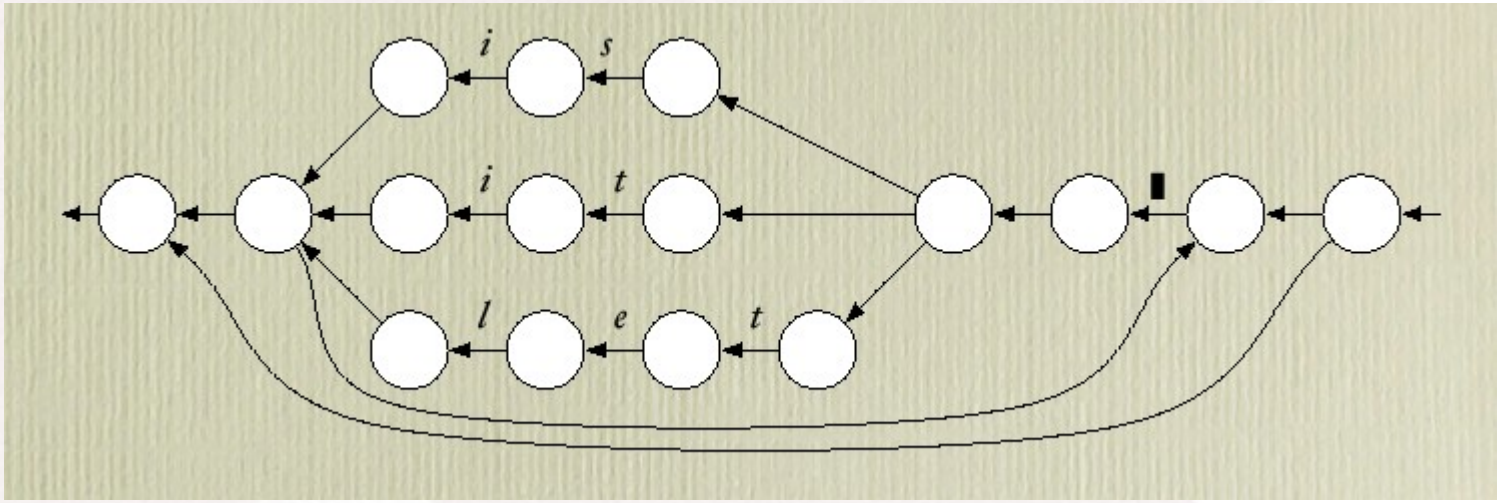


Reversión

Reversión de izquierda a derecha

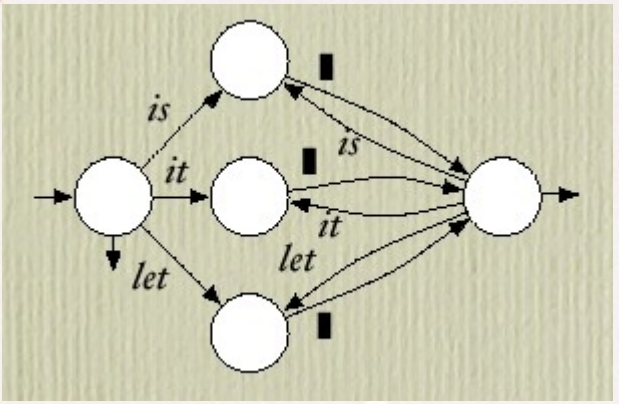


Reversión de izquierda a derecha

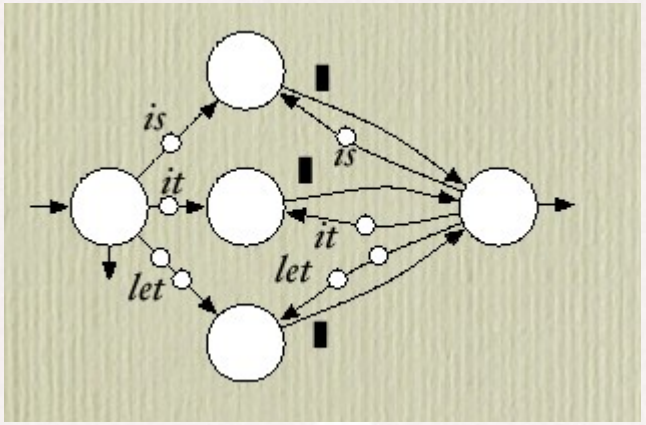


Determinización

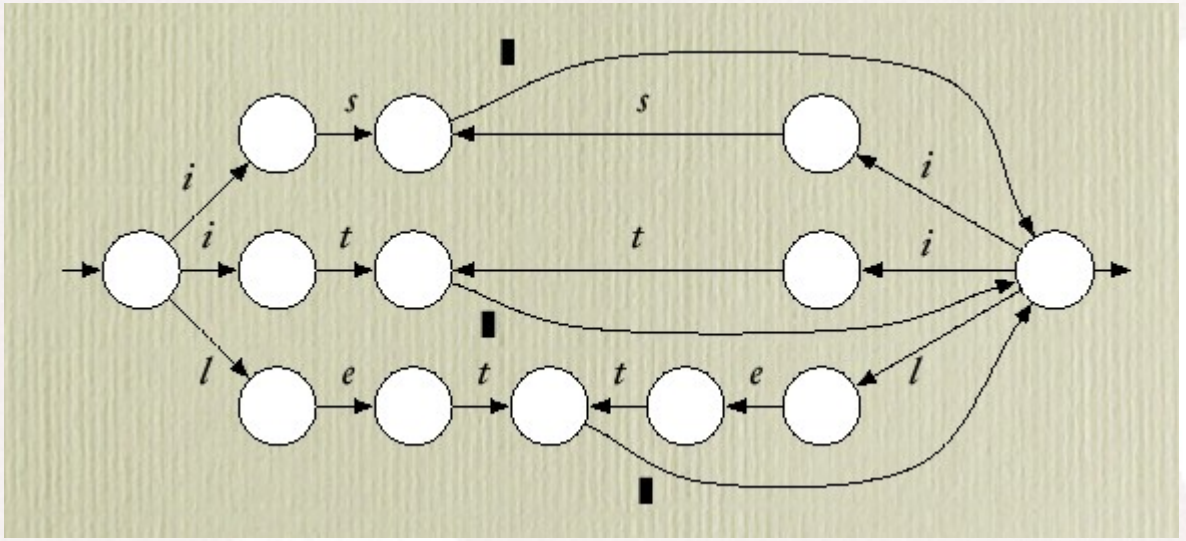
Determinización



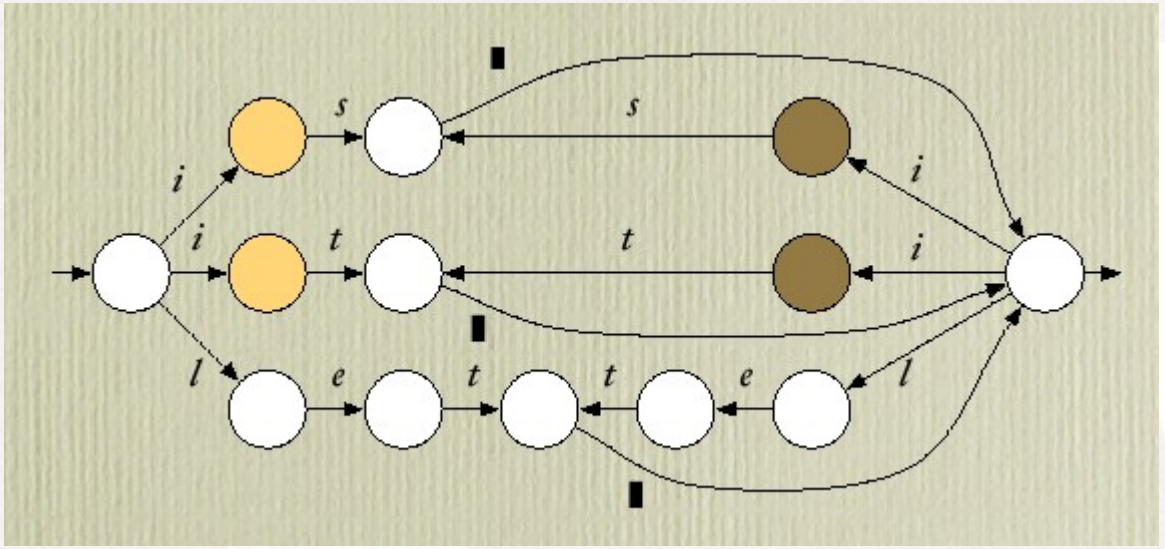
Determinización



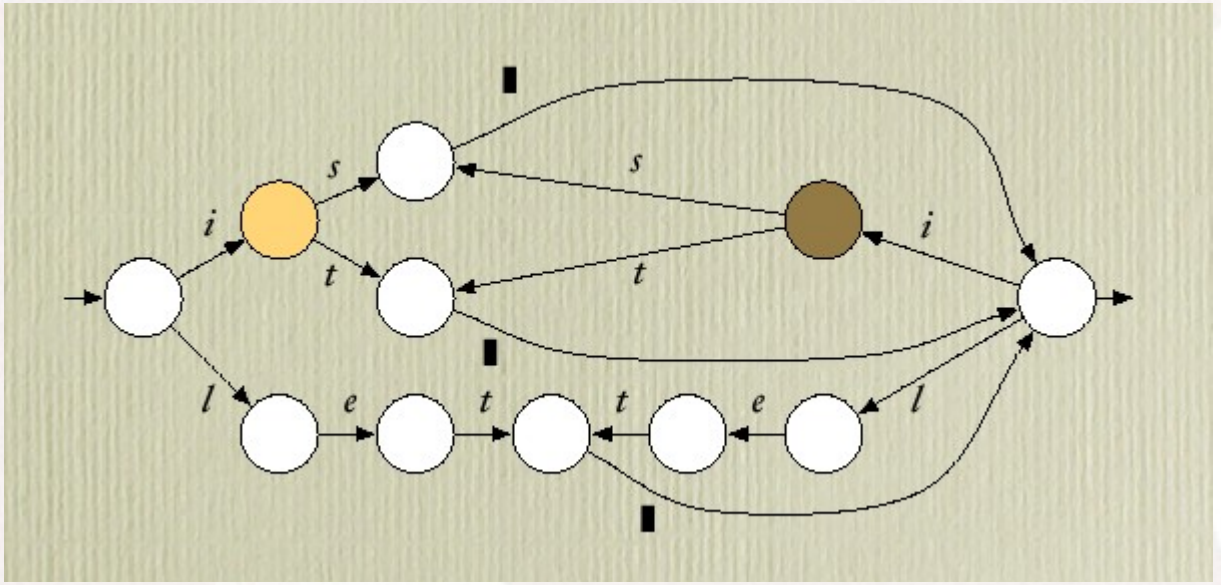
Determinización



Determinización

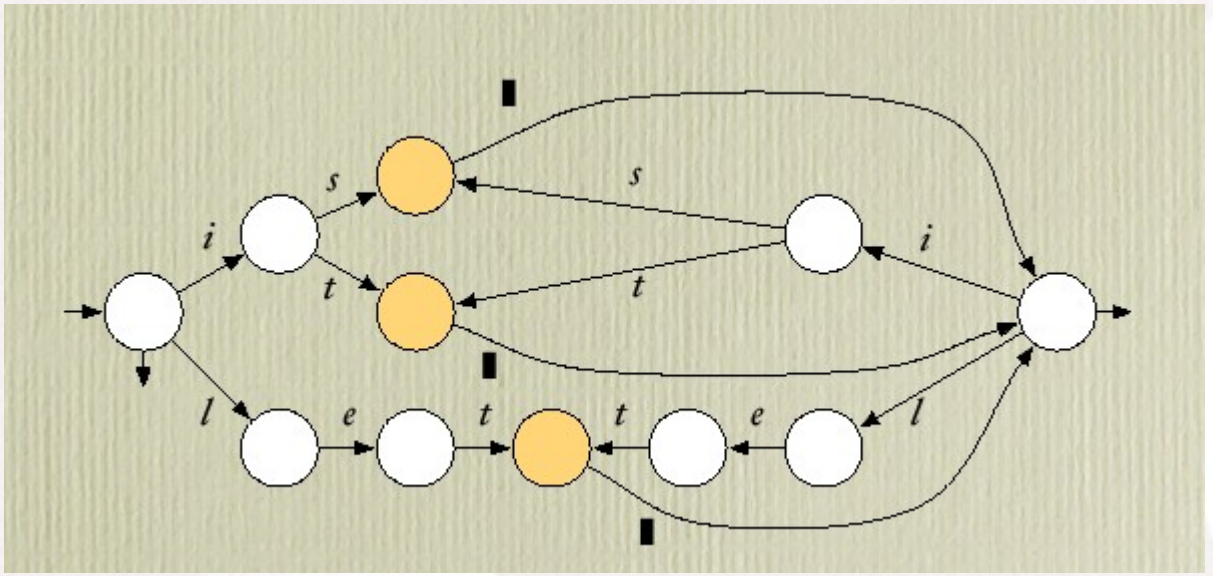


Determinización



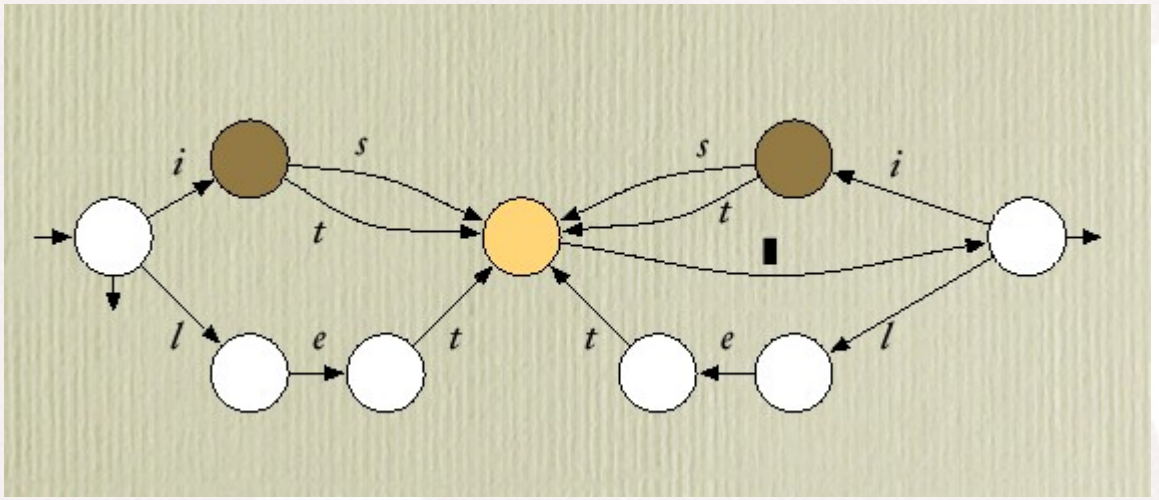
Minimización

Minimización



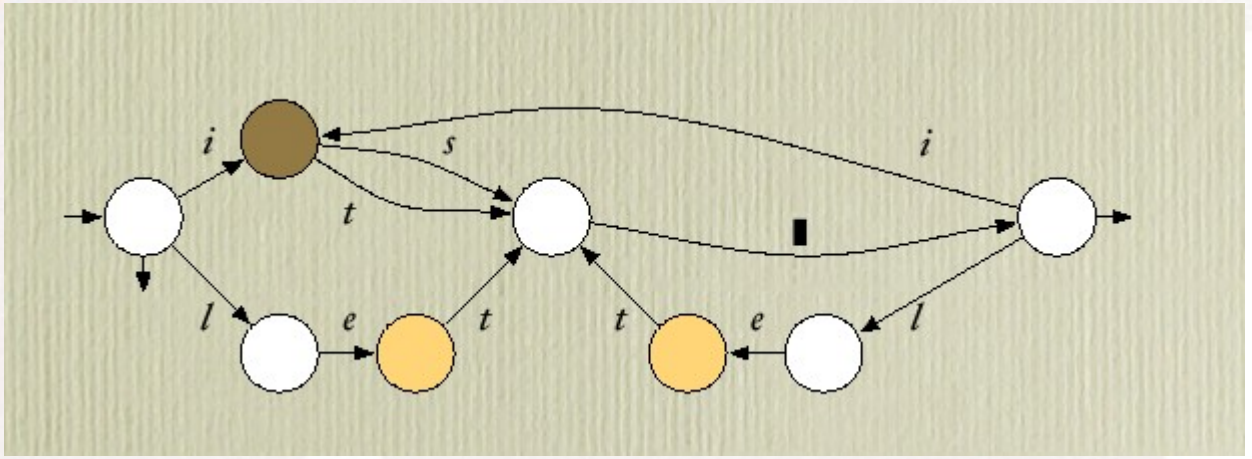
Consideramos a los estados como clases de equivalencia de sufijos. Estados que tengan los mismos sufijos pueden unirse.

Minimización



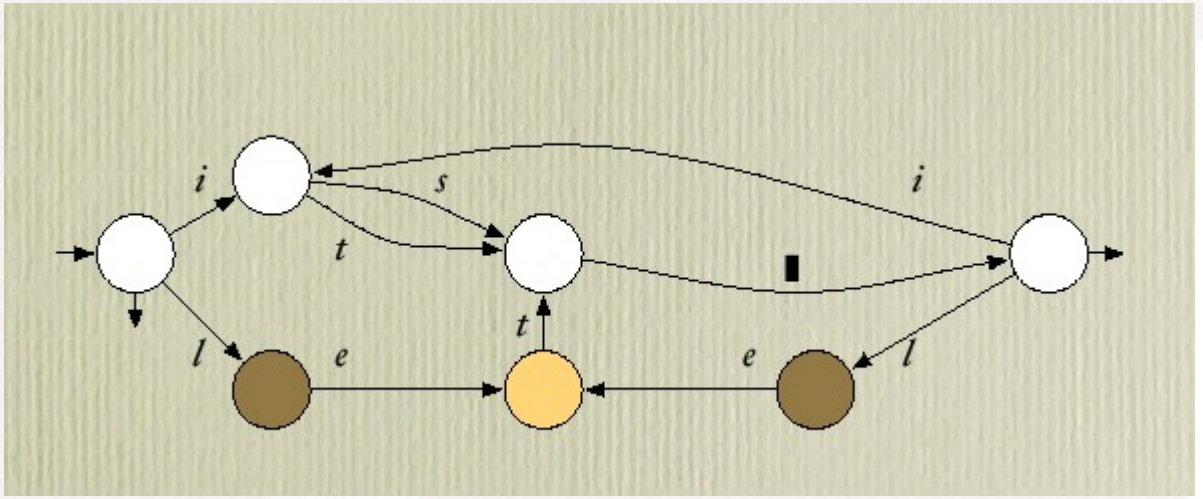
Consideramos a los estados como clases de equivalencia de sufijos. Estados que tengan los mismos sufijos pueden unirse.

Minimización



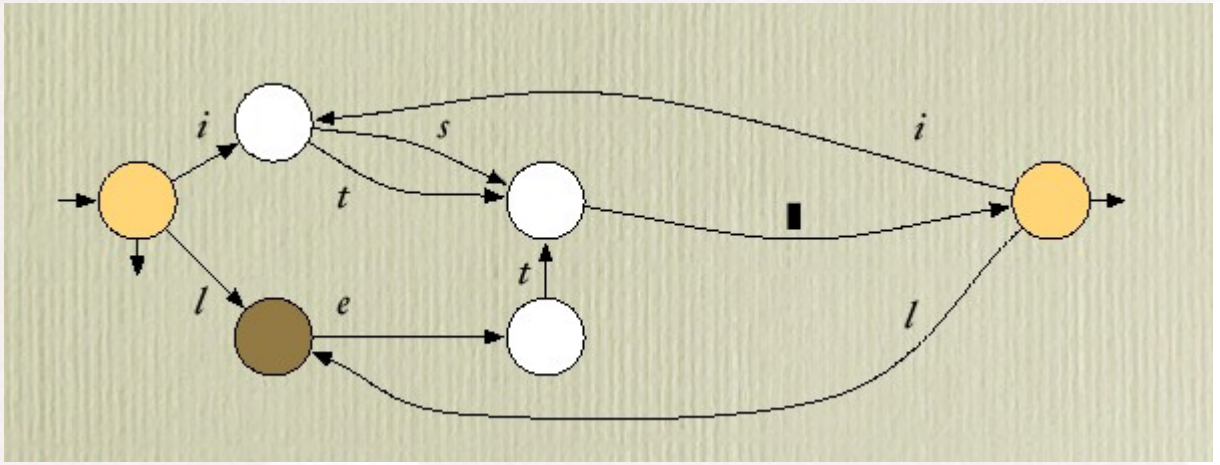
Consideramos a los estados como clases de equivalencia de sufijos. Estados que tengan los mismos sufijos pueden unirse.

Minimización



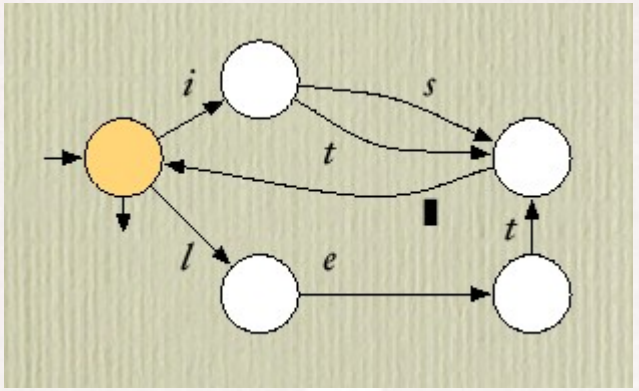
Consideramos a los estados como clases de equivalencia de sufijos. Estados que tengan los mismos sufijos pueden unirse.

Minimización



Consideramos a los estados como clases de equivalencia de sufijos. Estados que tengan los mismos sufijos pueden unirse.

Minimización



Consideramos a los estados como clases de equivalencia de sufijos. Estados que tengan los mismos sufijos pueden unirse.

Transductores de estados finitos

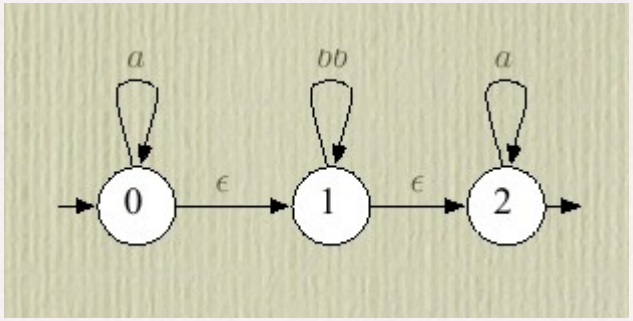
Relaciones Regulares

- Una **relación de cadenas** es un conjunto de pares de cadenas.

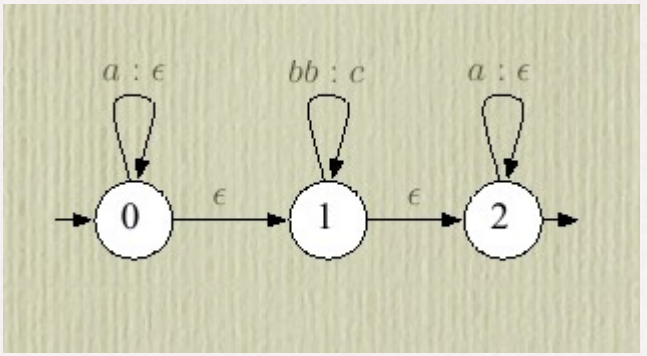
input : output

- La clase de las relaciones regulares es la clase de relaciones de cadenas más pequeña que incluye
  - La relación vacía: {}.
  - Las relaciones unitarias: {ϵ:ϵ}, {ϵ:a}, {a:ϵ}, {a:b}, ...
  - Cerradas bajo
    - Unión
    - Concatenación {(xx',yy') | (x,y) $\in$ R, (x',y') $\in$ R'}
    - Iteración

Transductores Finitos



Transductores Finitos



Función de transición

$$q_0\, a \vdash q_0 \qquad q_0 \vdash q_1$$
$$q_1\, bb \vdash c\, q_1 \qquad q_1 \vdash q_2$$
$$q_2\, a \vdash q_2 \qquad q_2\, \sharp \vdash \sharp$$

Una derivación:

$$q_0 aaabba\# \vdash q_0 aabba\# \vdash q_0 abba\# \vdash$$
$$\vdash q_0 abba\# \vdash q_1 bba\# \vdash c q_1 a\#$$
$$\vdash c q_2 a\# \vdash c q_2\# \vdash c\#$$

Def. de aceptación: aceptar $s{:}t$ sii $q_0 s\# \vdash^* t\#$

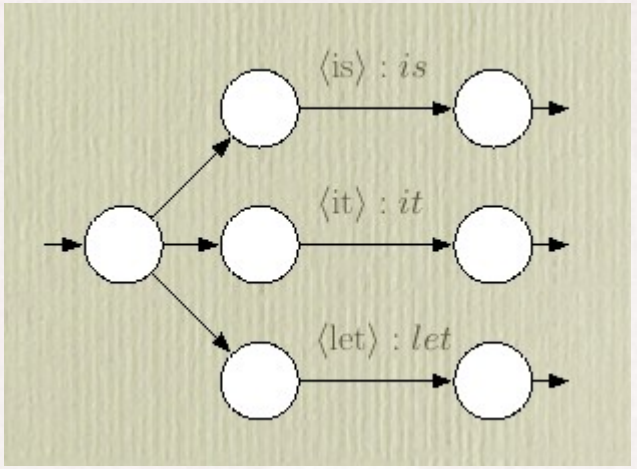
Operaciones sobre transductores

Operaciones sobre transductores

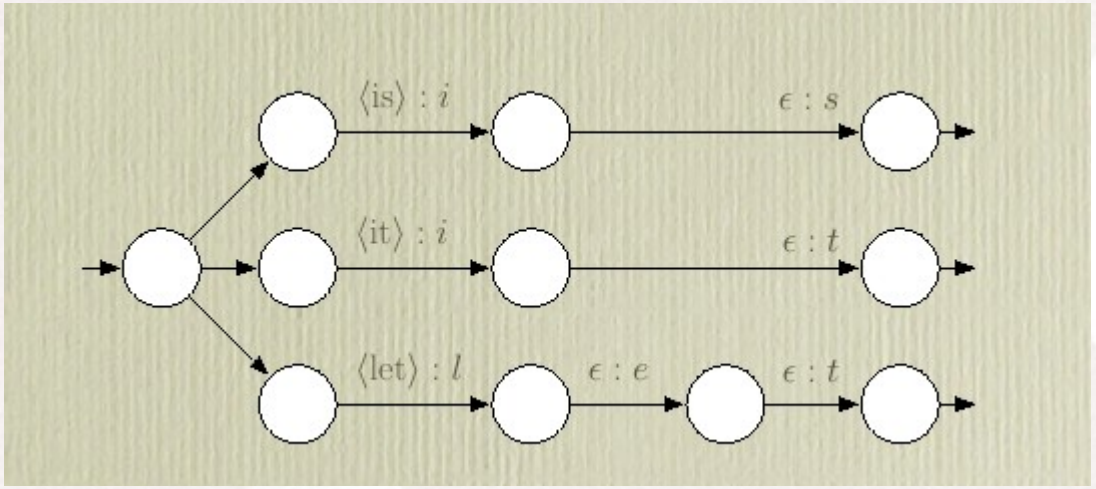
- Varias de las operaciones sobre autómatas también funcionan sobre transductores (más o menos). Y aparecen algunas nuevas:
  - Granularidad (de entrada y salida)
  - Reversión izquierda-derecha
  - Eliminación de épsilon (*)
  - ***Pushing***
  - Determinización (*)
  - Minimización (*)
  - ***Inversión***
  - ***Composición***

(*) a veces

Granularidad (output)

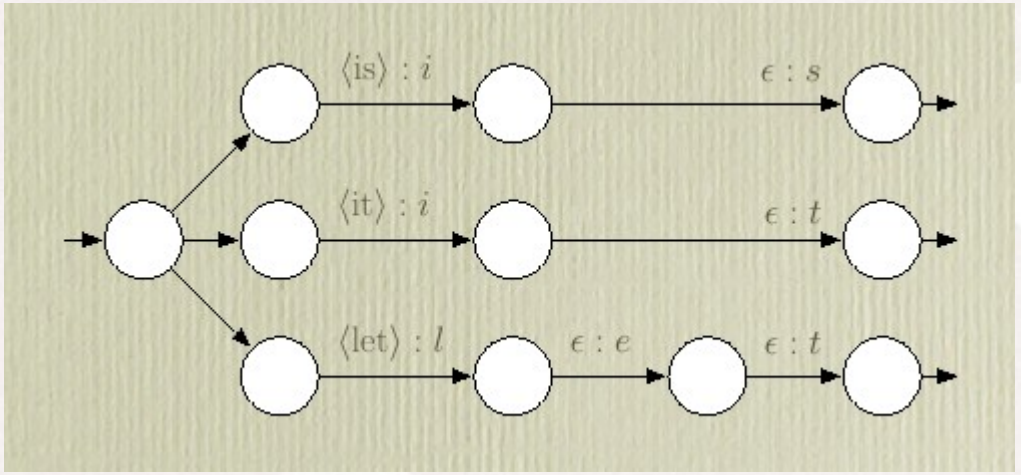


Granularidad (output)

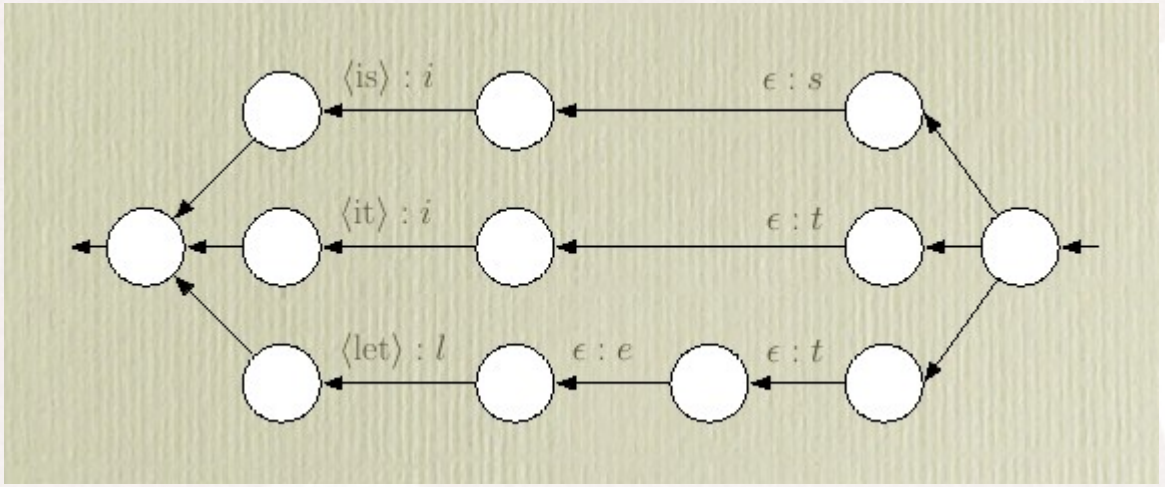


Un diccionario de deletreo.

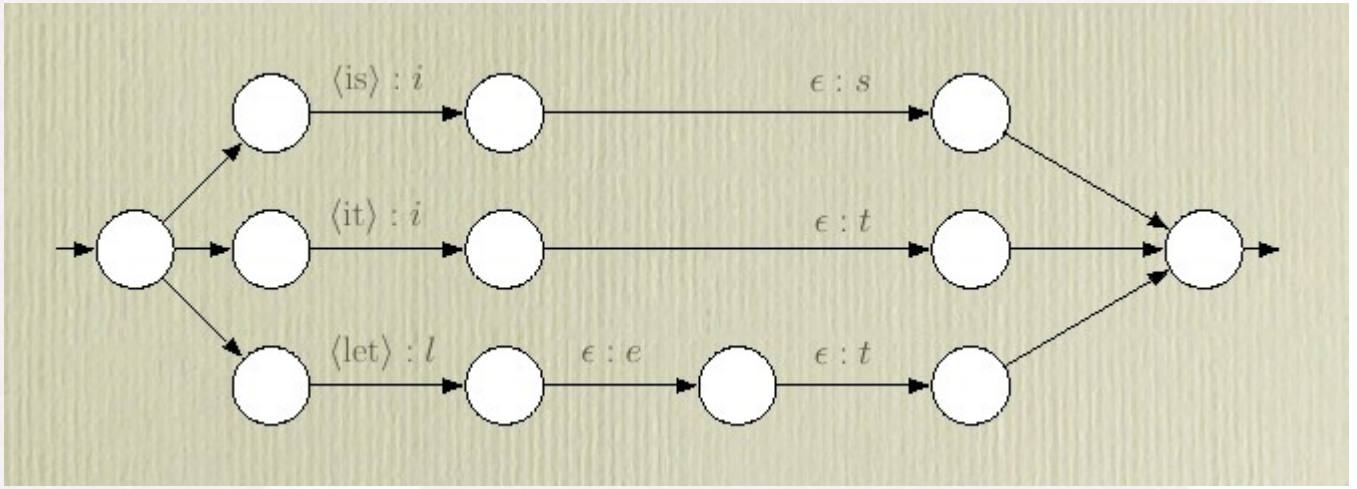
Reversión de izquierda a derecha



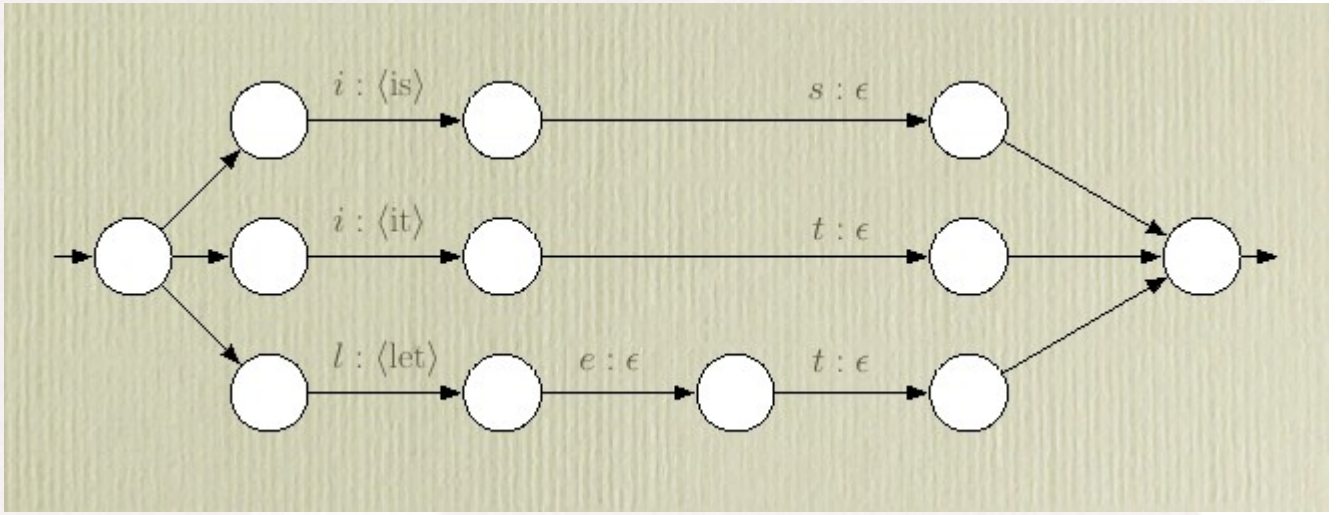
Reversión de izquierda a derecha



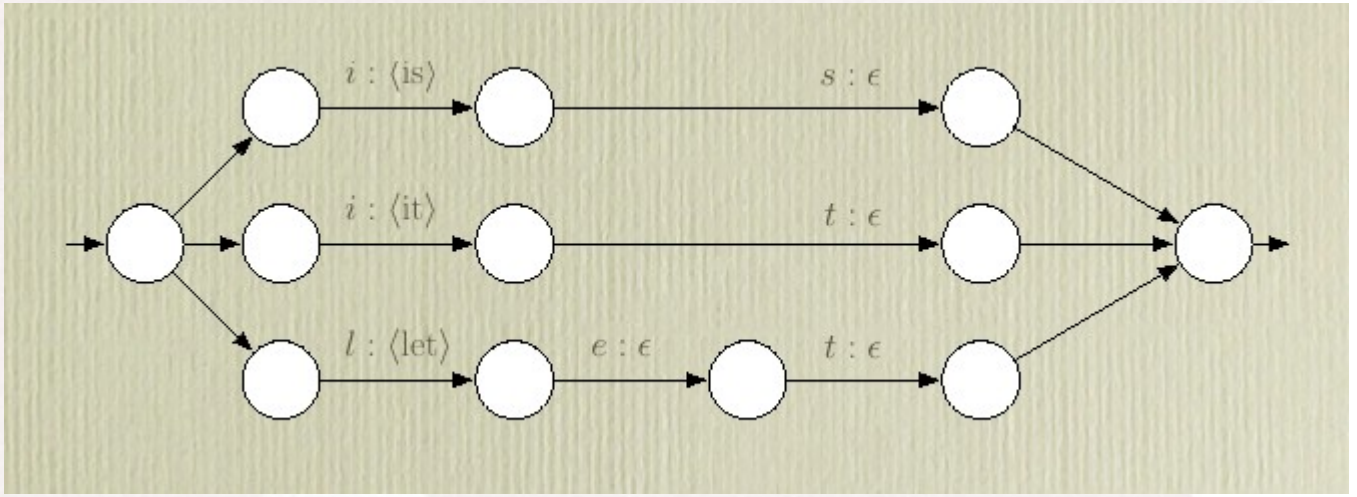
Inversión



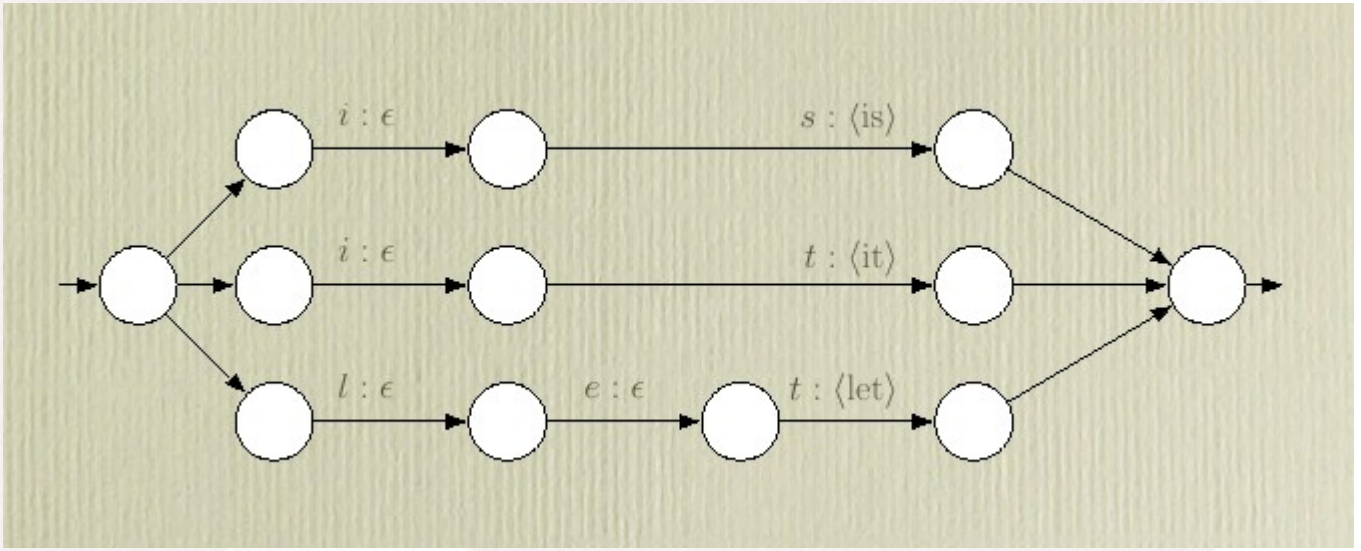
Inversión



Pushing


$i : \langle \text{is} \rangle$
$s : \epsilon$
$i : \langle \text{it} \rangle$
$t : \epsilon$
$l : \langle \text{let} \rangle$
$e : \epsilon$
$t : \epsilon$

Pushing


$i : \epsilon$
$s : \langle \text{is} \rangle$
$i : \epsilon$
$t : \langle \text{it} \rangle$
$l : \epsilon$
$e : \epsilon$
$t : \langle \text{let} \rangle$

Bibliografía

- *“Introduction to Finite-State Devices in Natural Language Processing”*, Emmanuel Roche and Yves Schabes. In E. Roche and Y. Schabes, editors, *Finite-State Language Processing*, chapter 1, pages 1-66. MIT Press, Cambridge, Massachusetts, 1997.