Contenidos

- **Introducción:**
  - Lenguajes Naturales, Análisis Léxico y Autómatas Finitos Acíclicos.
- **Implementación de Grandes Diccionarios:**
  - Modelización Compacta de un Diccionario.
  - Autómatas Finitos Acíclicos Mínimos Numerados.
- **Construcción de Autómatas Acíclicos:**
  - Algoritmo General de Minimización.
  - Minimización Basada en la Propiedad de la Altura.
  - Algoritmo de Construcción Incremental.
- **Nuevas Funcionalidades:**
  - Mejoras en el Acceso al Registro.
  - Comportamiento Dinámico de los Autómatas Acíclicos.
- **Conclusiones.**

Análisis Léxico y Autómatas Finitos Acíclicos

- **Tareas de los analizadores léxicos para lenguajes naturales:**
  - Transformar el flujo de caracteres de entrada en palabras.
  - Obtener rápida y cómodamente las etiquetas candidatas de las palabras.

Análisis Léxico y Autómatas Finitos Acíclicos

- **Tareas de los analizadores léxicos para lenguajes naturales:**
  - Transformar el flujo de caracteres de entrada en palabras.
  - Obtener rápida y cómodamente las etiquetas candidatas de las palabras.
- **Razones para comprimir un diccionario en un autómata finito acíclico:**
  - La representación del conjunto de palabras es muy compacta.
  - La búsqueda de las palabras en el diccionario es muy rápida.
  - Los autómatas acíclicos reconocen conjuntos finitos de palabras.

Análisis Léxico y Autómatas Finitos Acíclicos

- **Tareas de los analizadores léxicos para lenguajes naturales:**
  - Transformar el flujo de caracteres de entrada en palabras.
  - Obtener rápida y cómodamente las etiquetas candidatas de las palabras.
- **Razones para comprimir un diccionario en un autómata finito acíclico:**
  - La representación del conjunto de palabras es muy compacta.
  - La búsqueda de las palabras en el diccionario es muy rápida.
  - Los autómatas acíclicos reconocen conjuntos finitos de palabras.
- **Motivaciones y objetivos:**
  - Diseñar una arquitectura general para la implementación de grandes diccionarios de palabras, válida para cualquier idioma.
  - Manejar eficientemente dos diccionarios de español:
    - Diccionario GALENA:
      291.604 palabras con 354.007 etiquetaciones posibles.
    - Diccionario ERIAL:
      775.621 palabras con 993.703 etiquetaciones posibles.

Contenidos

- **Introducción:**
  - Lenguajes Naturales, Análisis Léxico y Autómatas Finitos Acíclicos.
- **Implementación de Grandes Diccionarios:**
  - Modelización Compacta de un Diccionario.
  - Autómatas Finitos Acíclicos Mínimos Numerados.
- **Construcción de Autómatas Acíclicos:**
  - Algoritmo General de Minimización.
  - Minimización Basada en la Propiedad de la Altura.
  - Algoritmo de Construcción Incremental.
- **Nuevas Funcionalidades:**
  - Mejoras en el Acceso al Registro.
  - Comportamiento Dinámico de los Autómatas Acíclicos.
- **Conclusiones.**

Modelización Compacta de un Diccionario

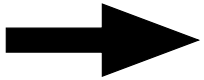
Texto Etiquetado

palabra	etiqueta	lema
...	...	...
no	Wn	no
requiere	V3spi0	requerir
conocimiento	Scms	conocimiento
sobre	P	sobre
las	Dfp	el
funciones	Scfp	función
suplementarias	Afp0	suplementario
de	P	de
...	...	...

Modelización Compacta de un Diccionario

Texto Etiquetado

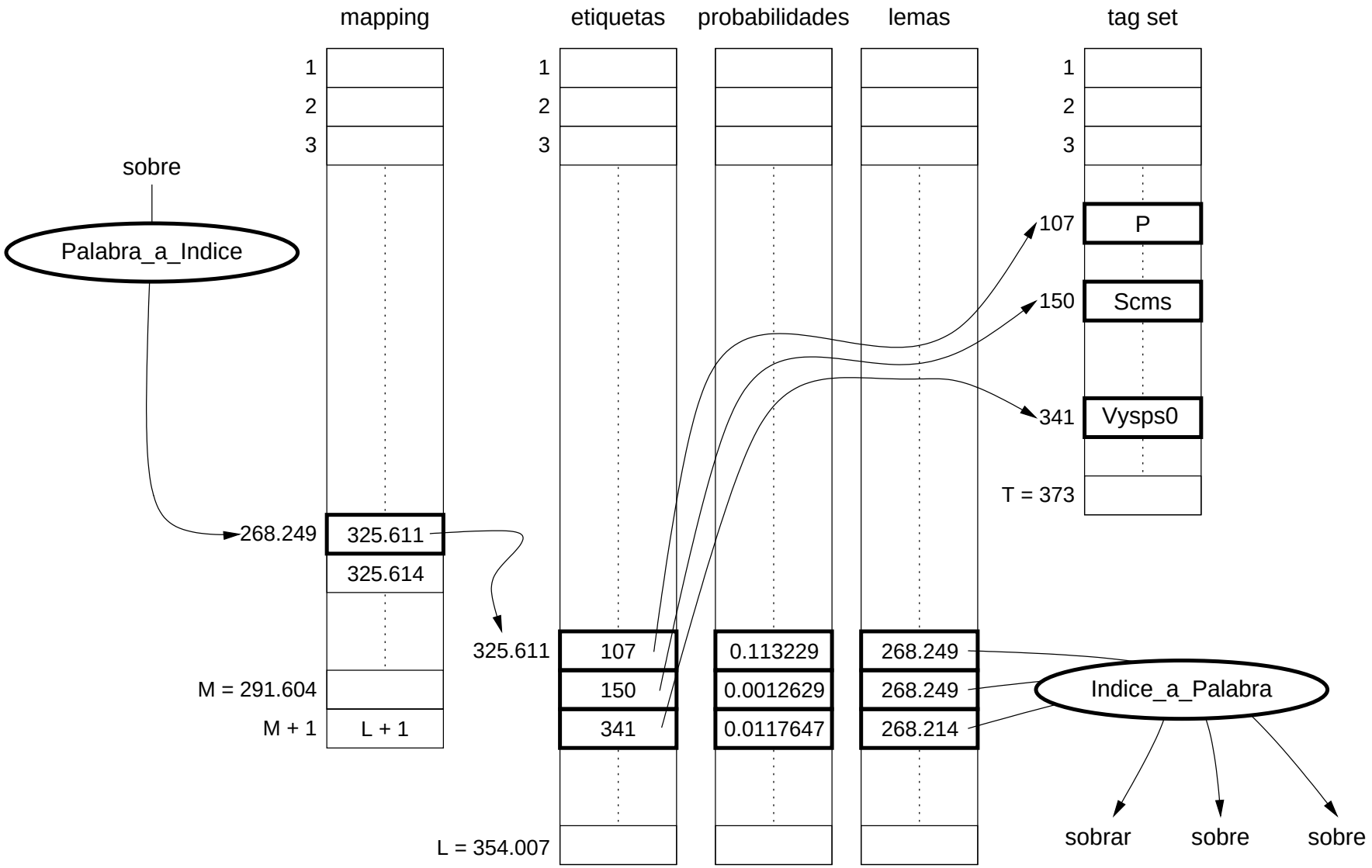
palabra	etiqueta	lema
...	...	...
no	Wn	no
requiere	V3spi0	requerir
conocimiento	Scms	conocimiento
sobre	P	sobre
las	Dfp	el
funciones	Scfp	función
suplementarias	Afp0	suplementario
de	P	de
...	...	...



Diccionario

palabra	etiqueta	lema	probabilidad
...	...	...	...
sobrasteis	V2pei0	sobrar	0.00377715
sobre	P	sobre	0.113229
sobre	Scms	sobre	0.0012629
sobre	Vysps0	sobrar	0.0117647
sobrecarga	Scfs	sobrecarga	0.00383284
sobrecarga	V2spm0	sobrecargar	0.00175131
sobrecarga	V3spi0	sobrecargar	0.000629723
sobrecargaba	Vysii0	sobrecargar	0.0026455
...	...	...	...

Modelización Compacta de un Diccionario

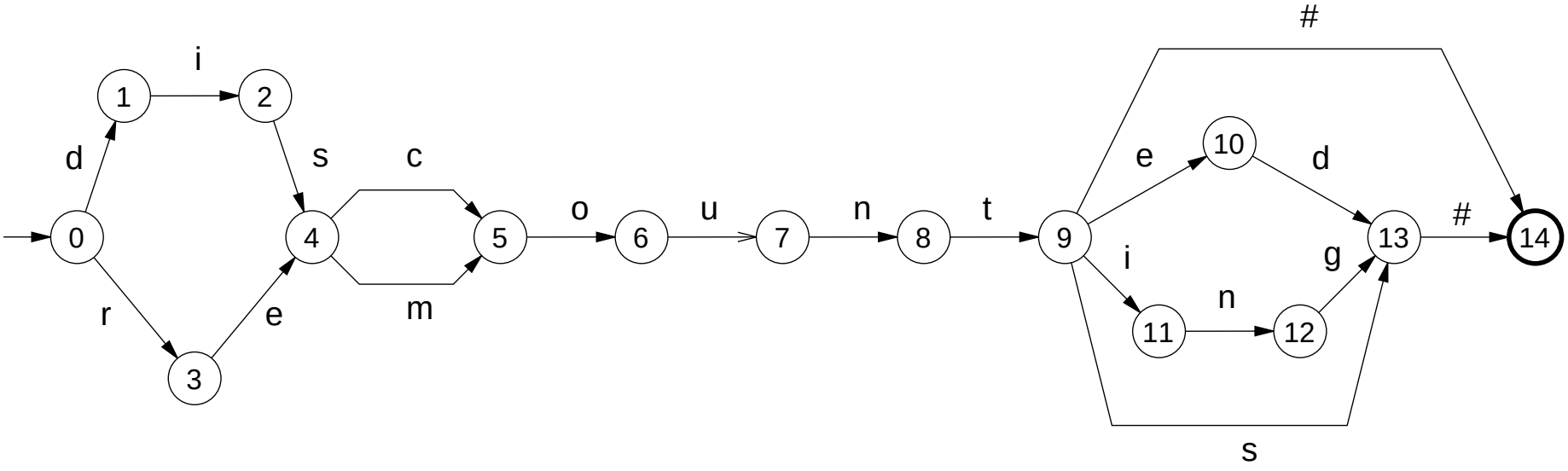


Autómatas Finitos Acíclicos Numerados

Peso de un Estado: $weight(q) =$ número de cadenas w tales que $q.w \in F$

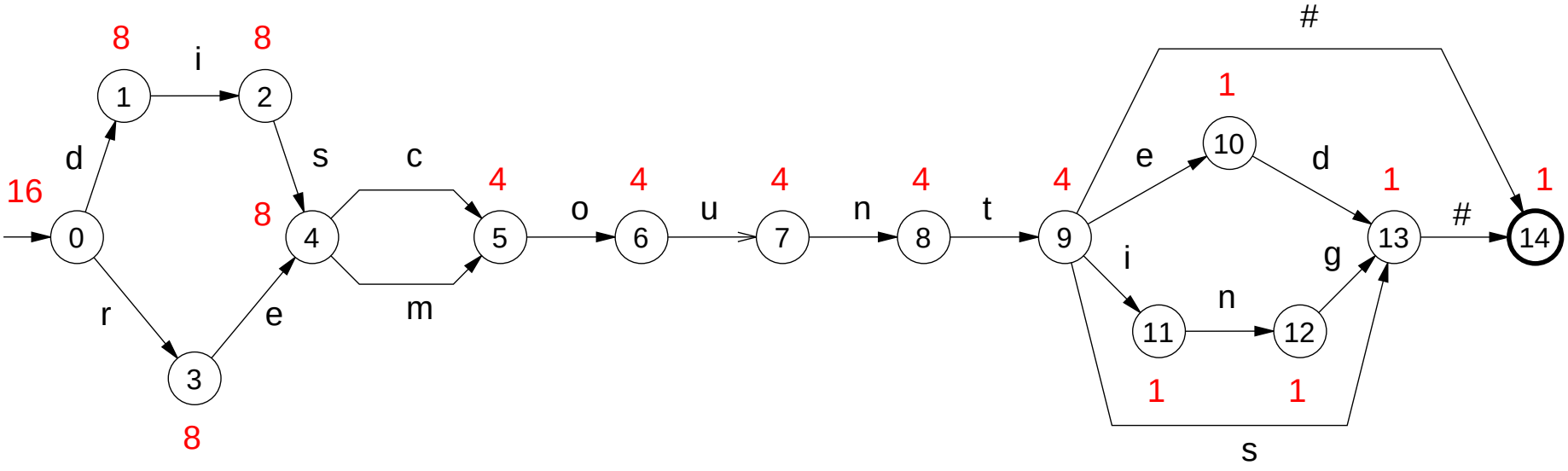
Autómatas Finitos Acíclicos Numerados

Peso de un Estado: $weight(q) =$ número de cadenas w tales que $q.w \in F$



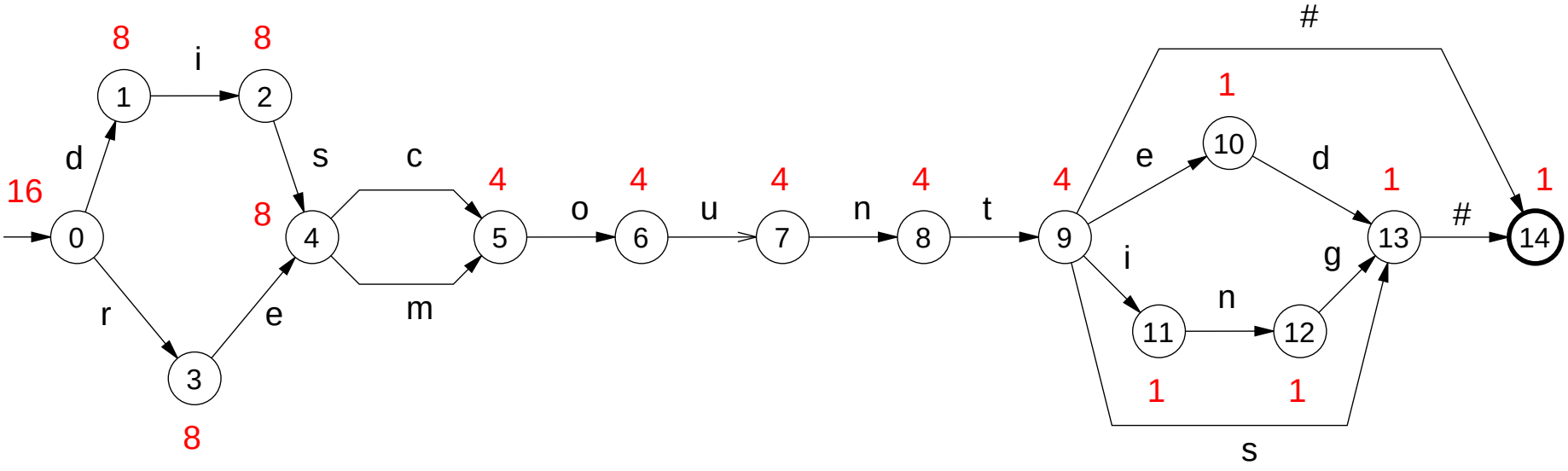
Autómatas Finitos Acíclicos Numerados

Peso de un Estado: $weight(q) =$ número de cadenas w tales que $q.w \in F$



Autómatas Finitos Acíclicos Numerados

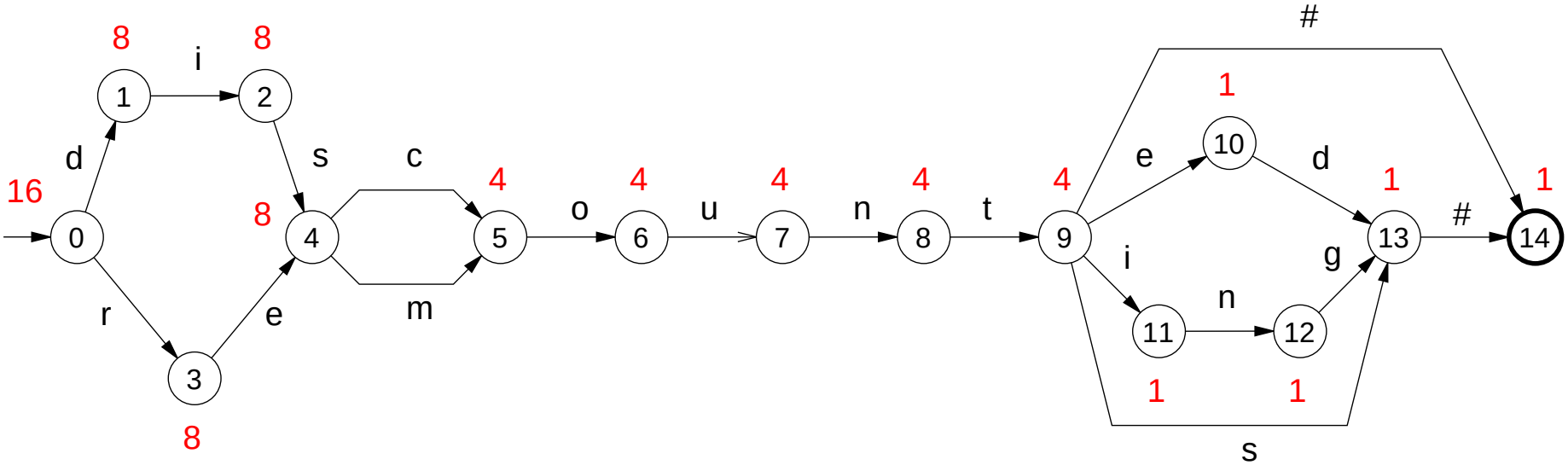
Peso de un Estado: $weight(q) =$ número de cadenas w tales que $q.w \in F$



1 = discount	2 = discounted	3 = discounting	4 = discounts
5 = dismount	6 = dismounted	7 = dismounting	8 = dismounts
9 = recount	10 = recounted	11 = recounting	12 = recounts
13 = remount	14 = remounted	15 = remounting	16 = remounts

Autómatas Finitos Acíclicos Numerados

Peso de un Estado: $weight(q) =$ número de cadenas w tales que $q.w \in F$

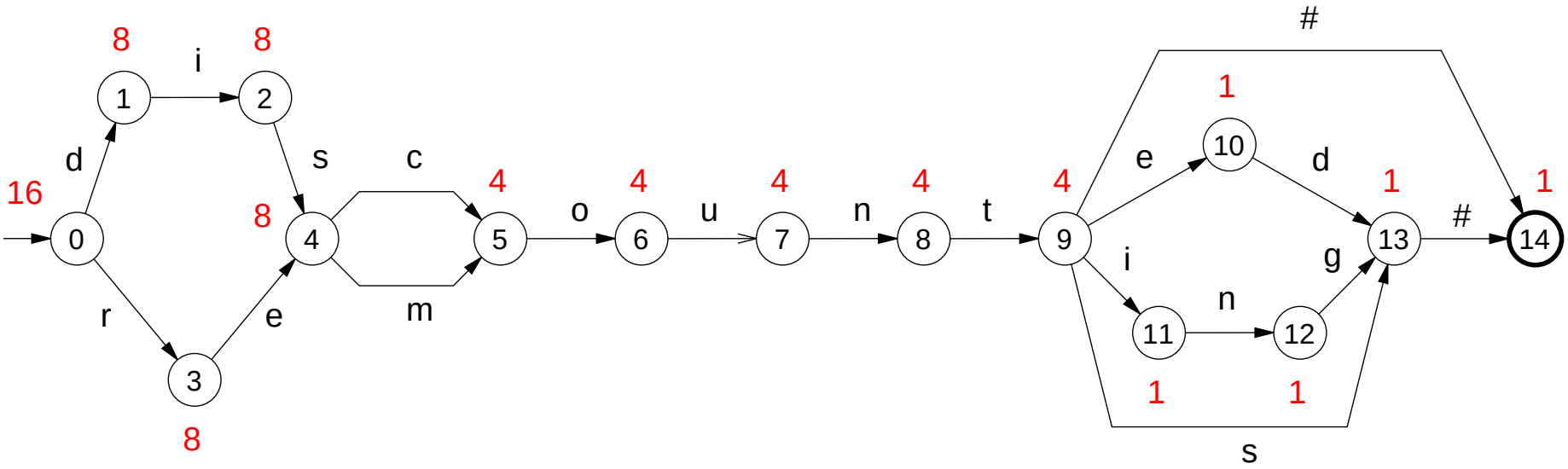


1 = discount	2 = discounted	3 = discounting	4 = discounts
5 = dismount	6 = dismounted	7 = dismounting	8 = dismounts
9 = recount	10 = recounted	11 = recounting	12 = recounts
13 = remount	14 = remounted	15 = remounting	16 = remounts

Hashing Perfecto [Lucchesi y Kowaltowski 1993]

Autómatas Finitos Acíclicos Numerados

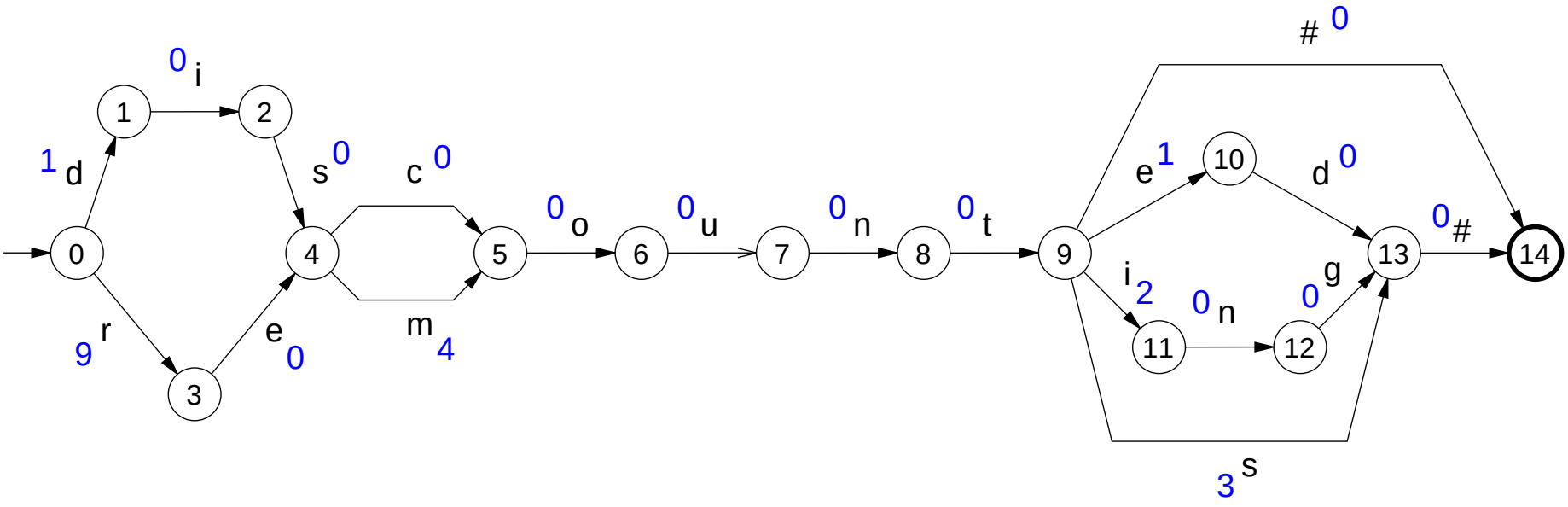
Peso de un Estado: $weight(q) =$ número de cadenas w tales que $q.w \in F$



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
34	2	d	r	1	i	1	s	1	e	2	c	m	1	o	1	u	
	16	4	8	8	6	8	10	8	10	8	13	13	4	15	4	17	
	Estado 0			Estado 1		Estado 2		Estado 3		Estado 4			Estado 5		Estado 6		

Autómatas Finitos Acíclicos Numerados

Peso de un Arco: suma de los pesos de los estados destino de los arcos precedentes.



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
			d	r		i		s		e		c	m		o		u
34	2		4	8	1	6	1	10	1	10	2	13	13	1	15	1	17
			1	9		0		0		0		0	4		0		0
		Estado 0			Estado 1		Estado 2		Estado 3		Estado 4			Estado 5		Estado 6	

Contenidos

- **Introducción:**
  - Lenguajes Naturales, Análisis Léxico y Autómatas Finitos Acíclicos.
- **Implementación de Grandes Diccionarios:**
  - Modelización Compacta de un Diccionario.
  - Autómatas Finitos Acíclicos Mínimos Numerados.
- **Construcción de Autómatas Acíclicos:**
  - Algoritmo General de Minimización.
  - Minimización Basada en la Propiedad de la Altura.
  - Algoritmo de Construcción Incremental.
- **Nuevas Funcionalidades:**
  - Mejoras en el Acceso al Registro.
  - Comportamiento Dinámico de los Autómatas Acíclicos.
- **Conclusiones.**

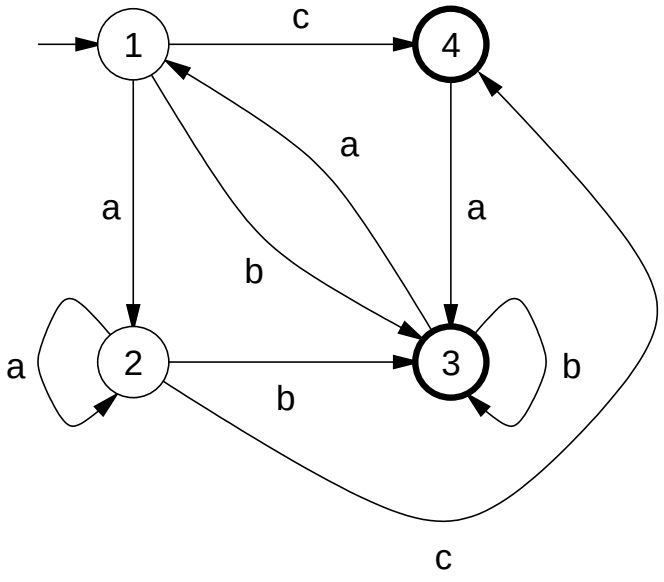
Algoritmo General de Minimización

Refinamiento Sucesivo de Particiones de Estados

Algoritmo General de Minimización

Refinamiento Sucesivo de Particiones de Estados

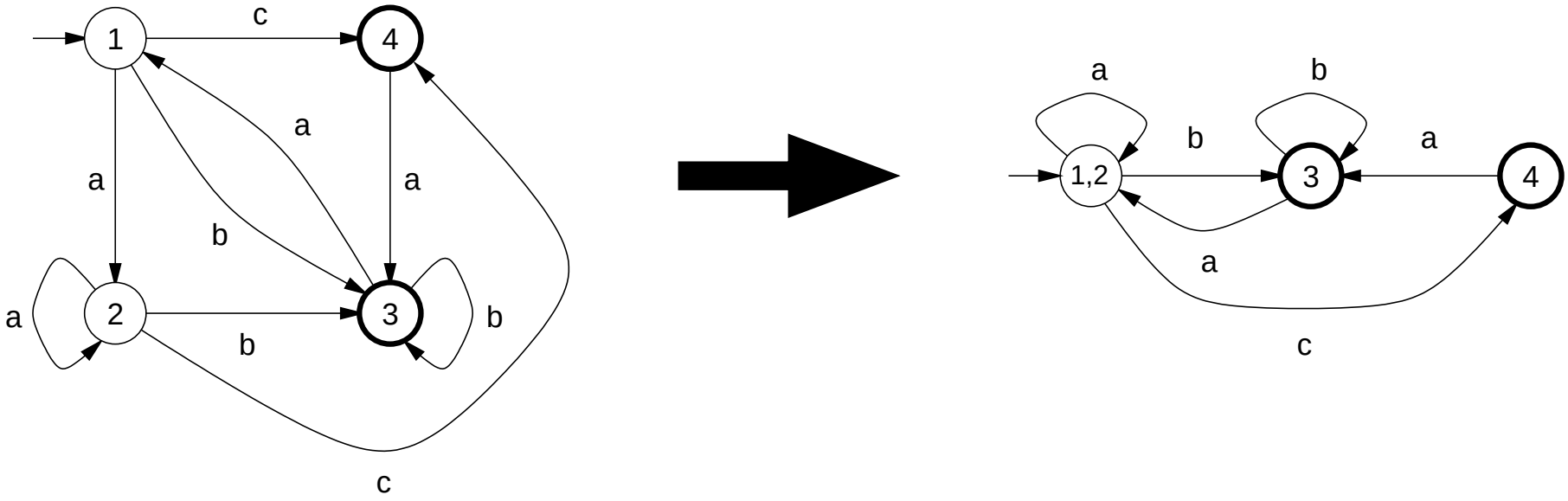
Partición Inicial: [(1,2) (3,4)] . . . Partición Final: [(1,2) (3) (4)]



Algoritmo General de Minimización

Refinamiento Sucesivo de Particiones de Estados

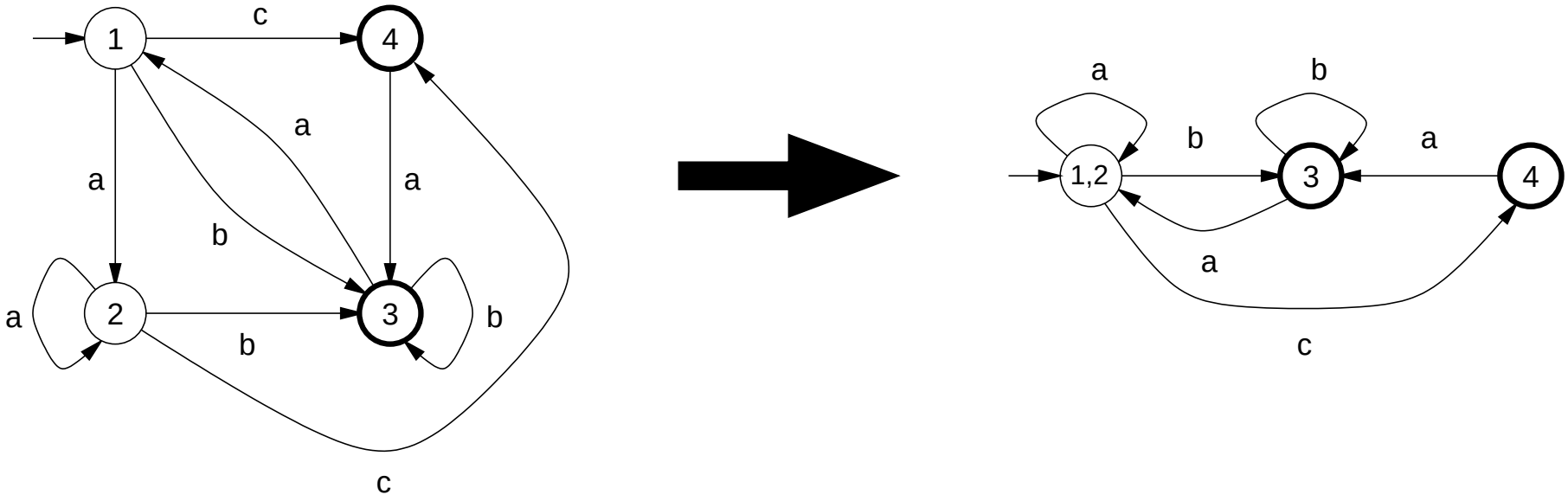
Partición Inicial: [(1,2) (3,4)] . . . Partición Final: [(1,2) (3) (4)]



Algoritmo General de Minimización

Refinamiento Sucesivo de Particiones de Estados

Partición Inicial: [(1,2) (3,4)] . . . Partición Final: [(1,2) (3) (4)]



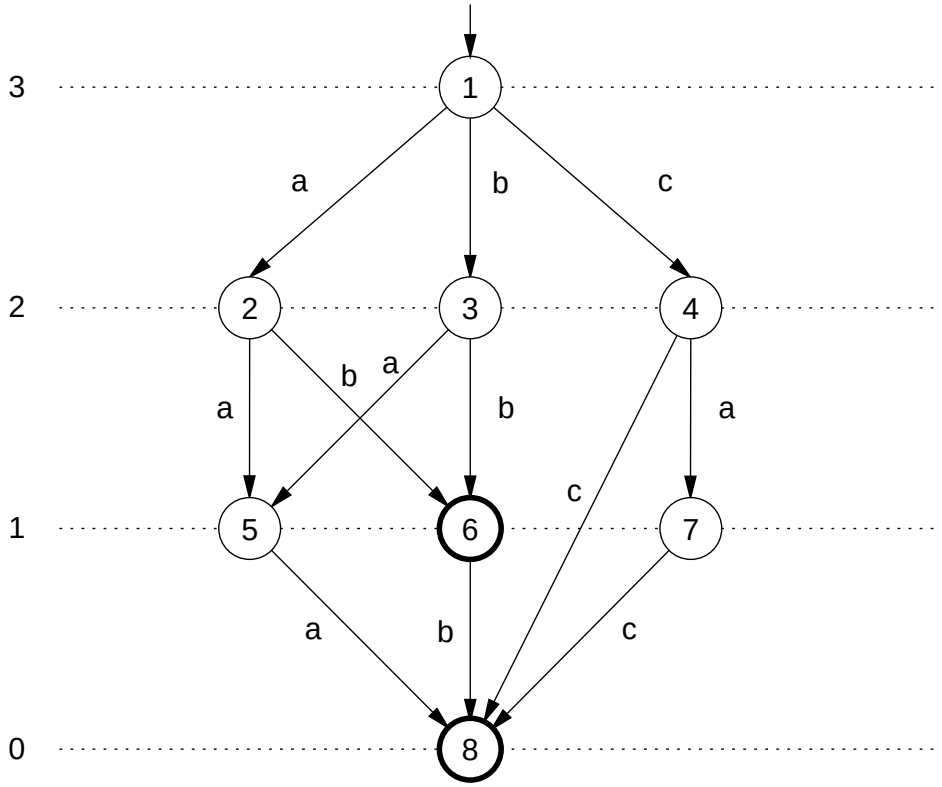
Complejidad Temporal: $\mathcal{O}(n \times \log n)$ [Hopcroft 1971]

Minimización Basada en la Altura

Altura de un Estado: $height(q) = max\{|w| \text{ tal que } q.w \in F\}$

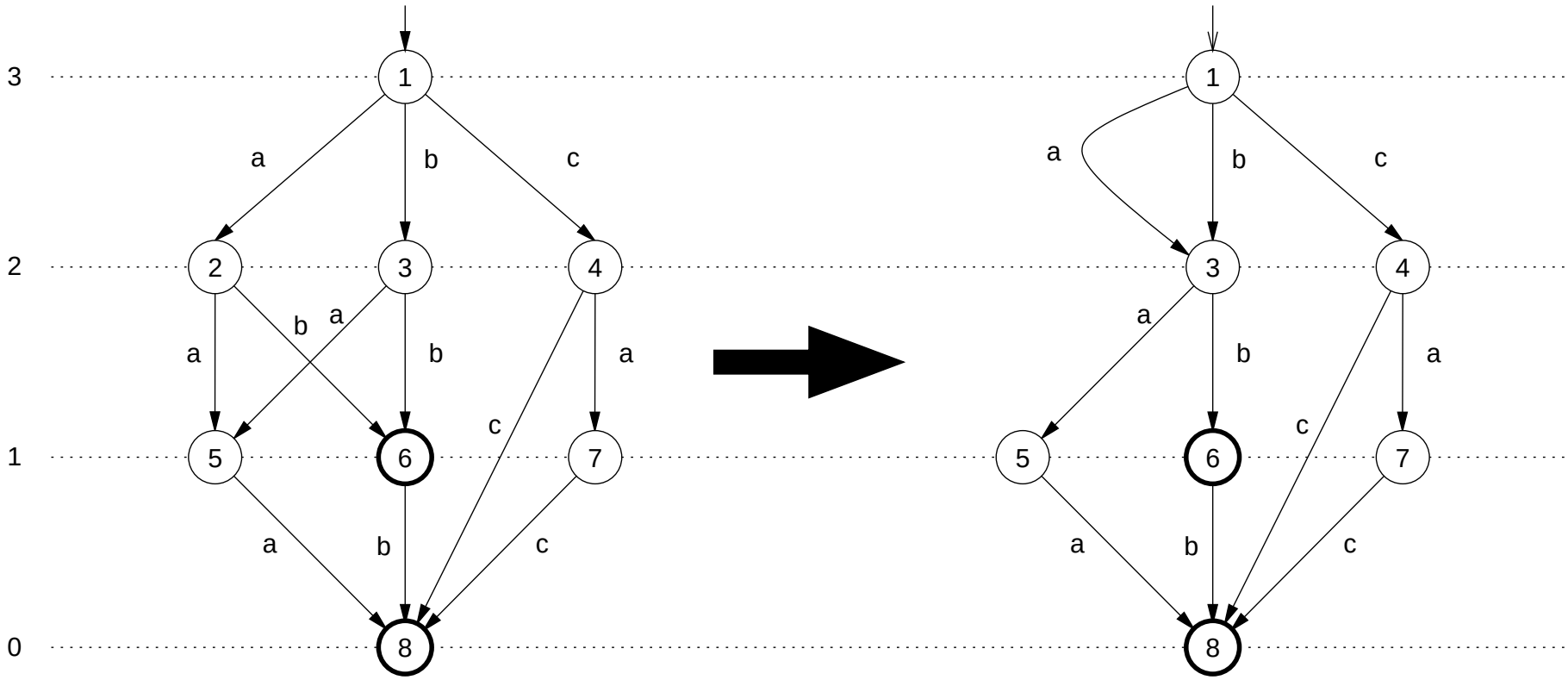
Minimización Basada en la Altura

Altura de un Estado: $height(q) = max\,\{|w| \text{ tal que } q.w \in F\}$



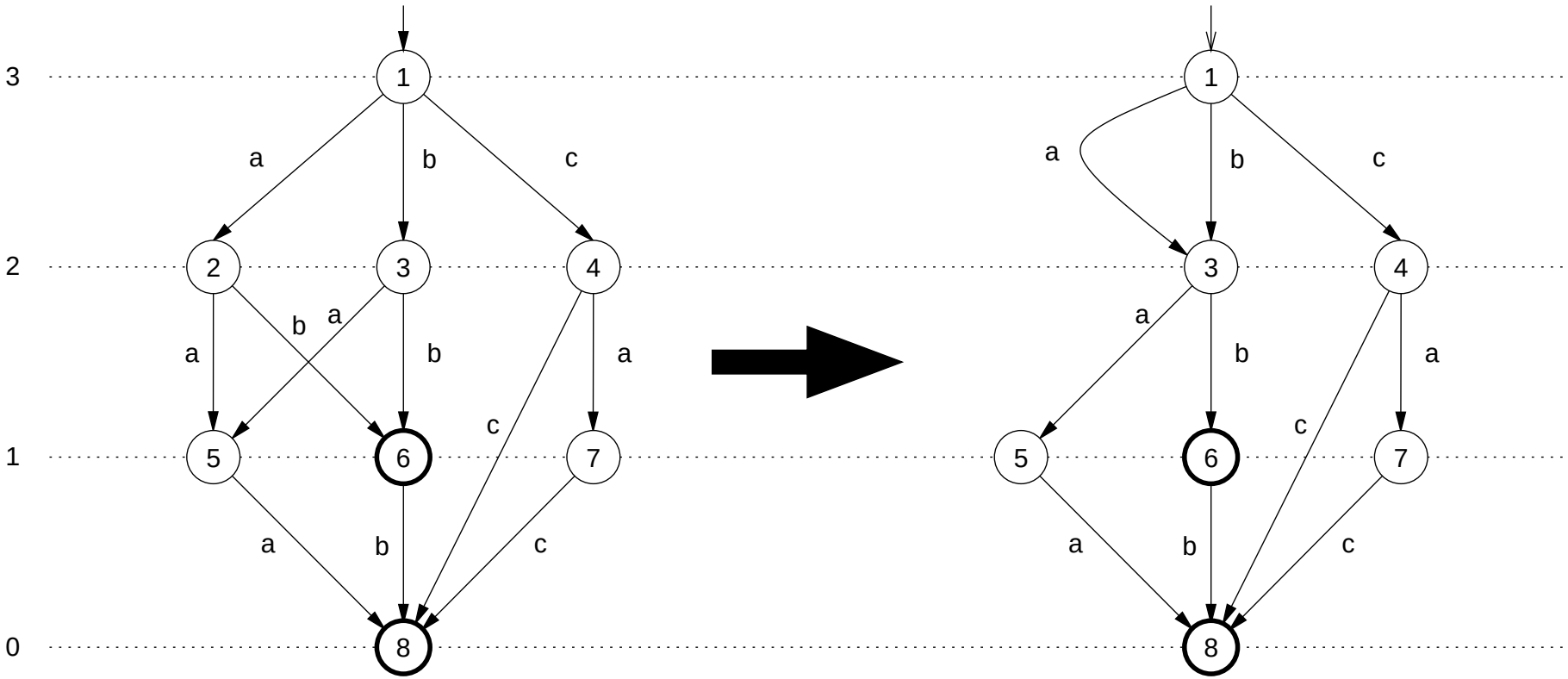
Minimización Basada en la Altura

Altura de un Estado: $height(q) = max\,\{|w| \text{ tal que } q.w \in F\}$



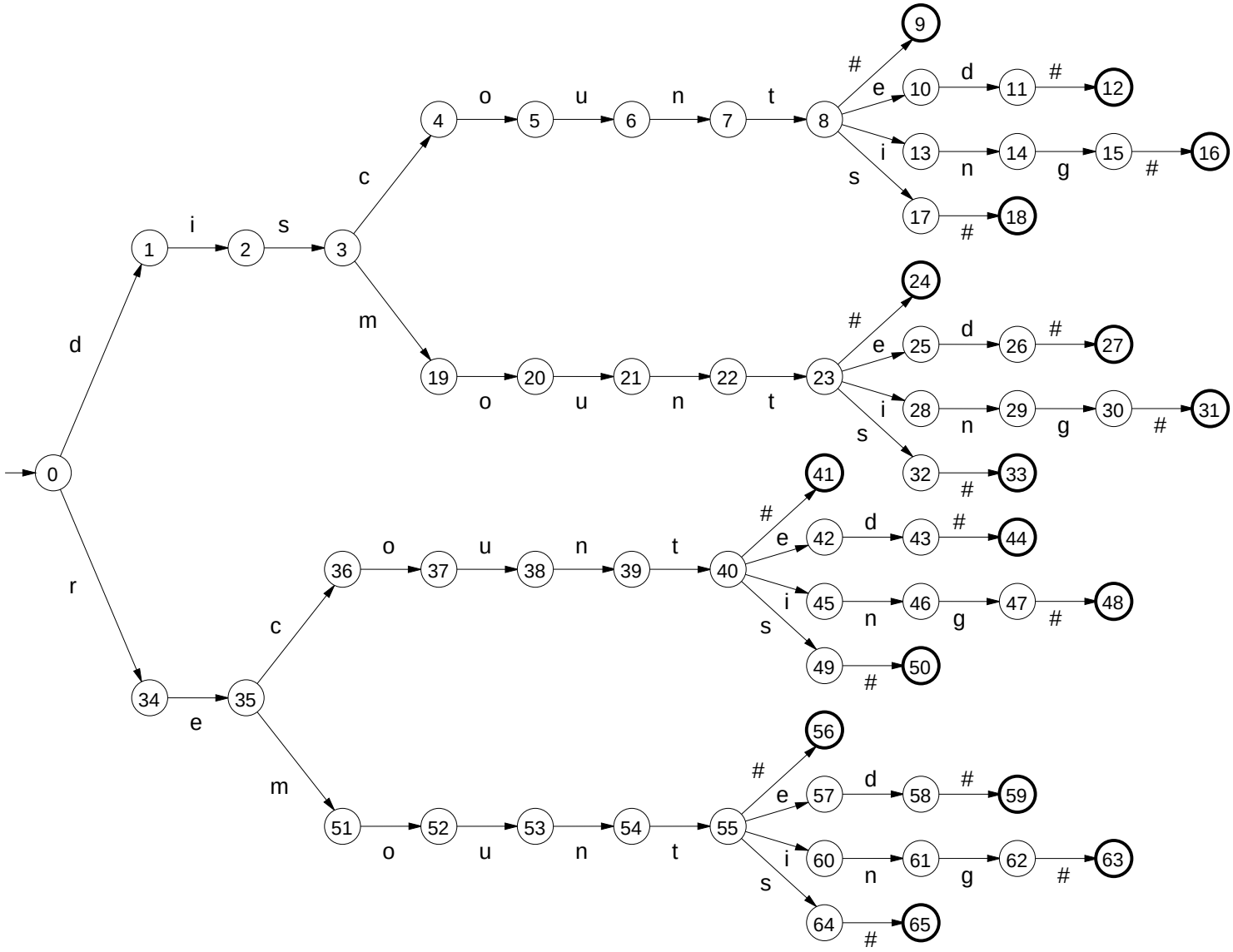
Minimización Basada en la Altura

Altura de un Estado: $height(q) = max\,\{|w| \text{ tal que } q.w \in F\}$

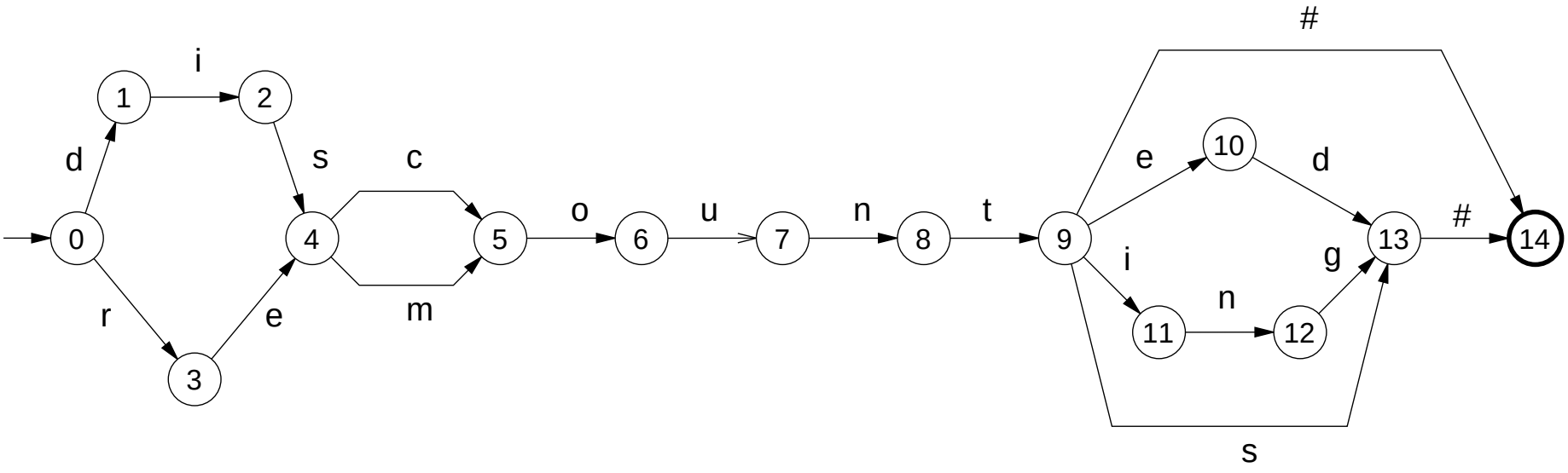


Complejidad Temporal: $\mathcal{O}(t + \sum_{i=0}^{h(q_0)} f(|\Pi_i|))$ [Revuz 1992]

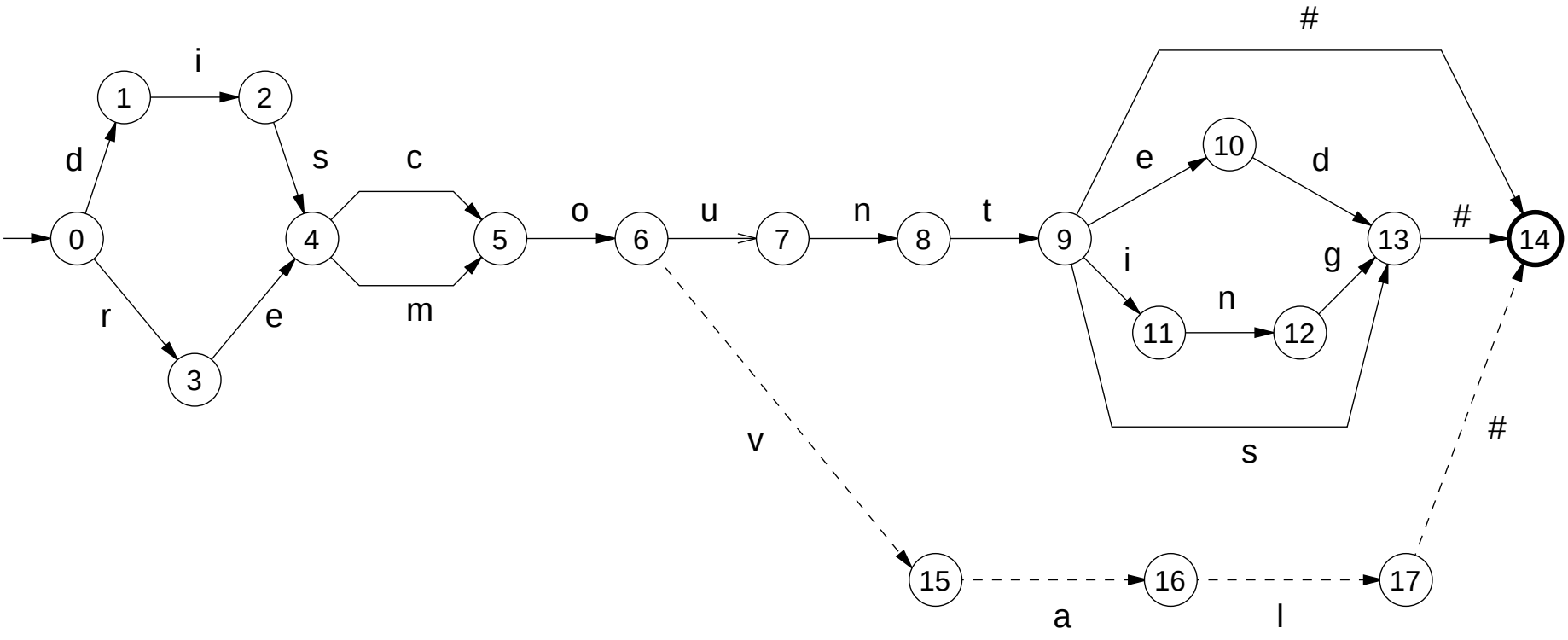
Fases de Inserción - Minimización



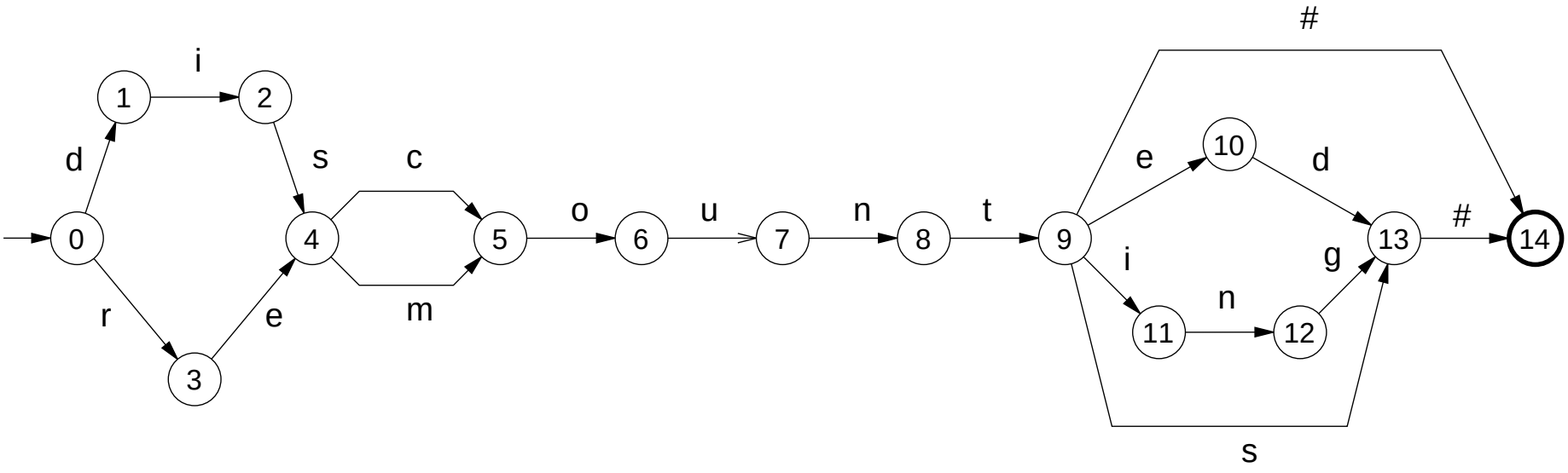
Fases de Inserción - Minimización



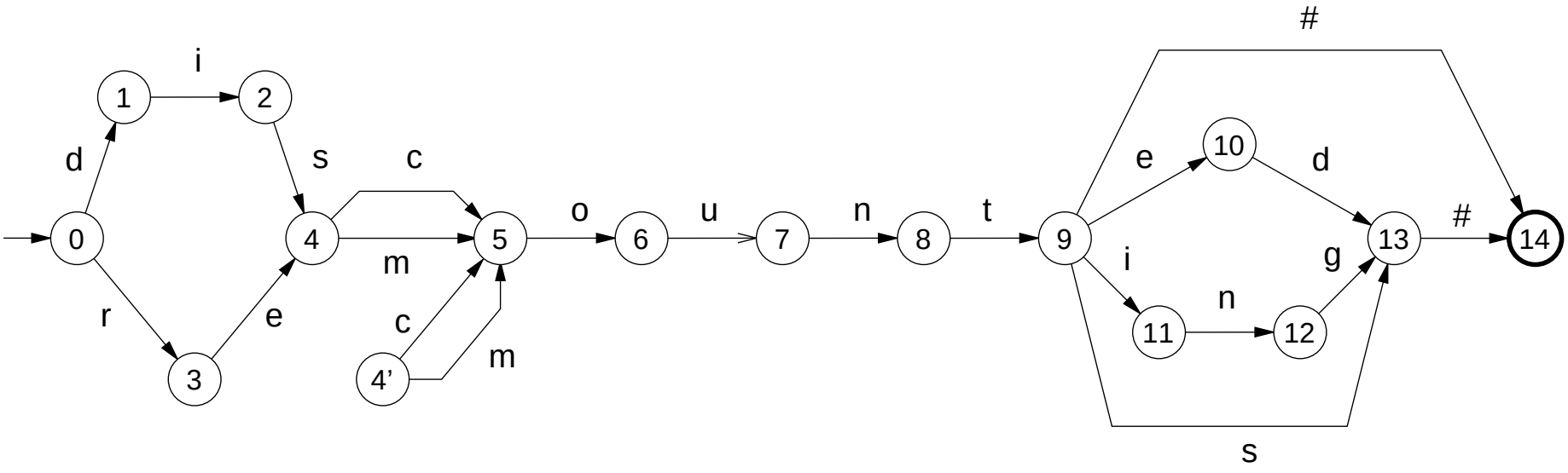
Fases de Inserción - Minimización



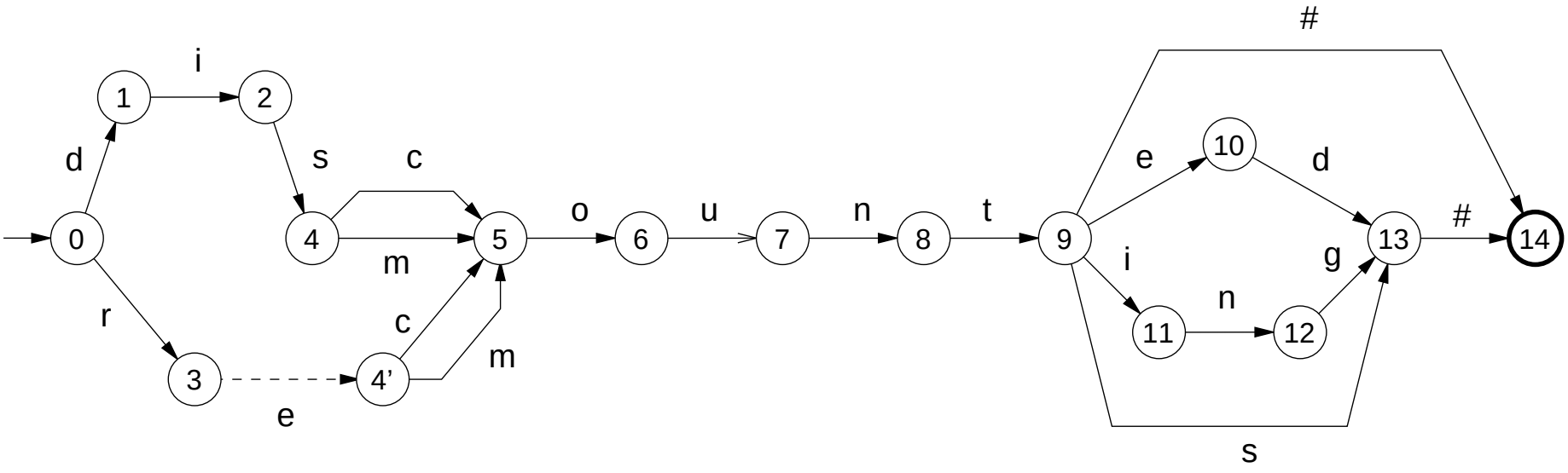
Fases de Inserción - Minimización



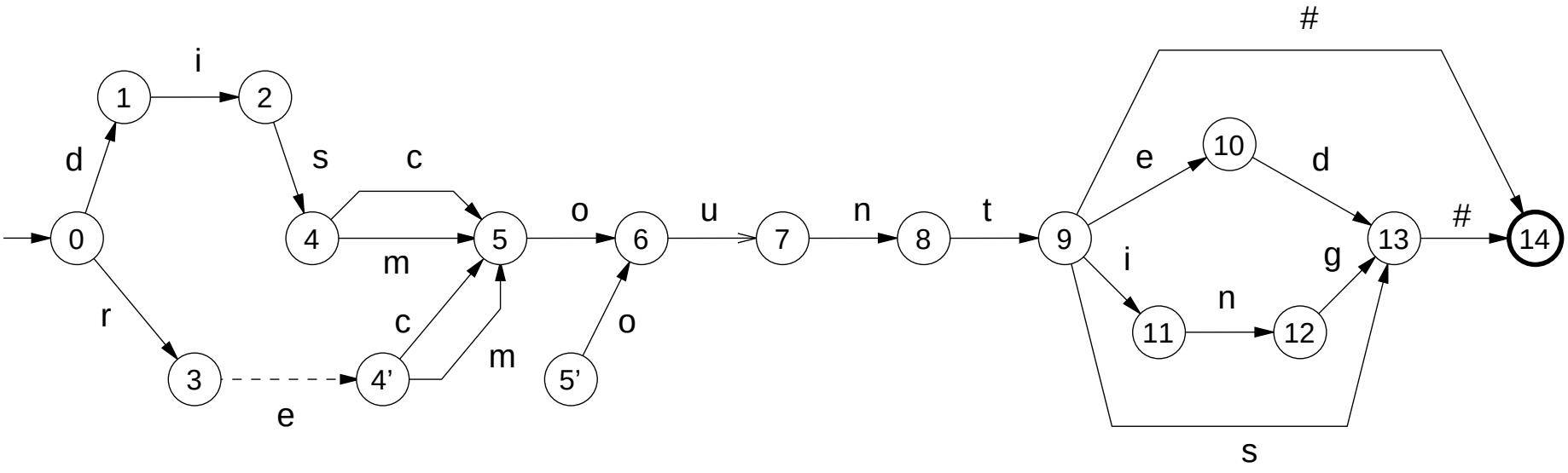
Fases de Inserción - Minimización



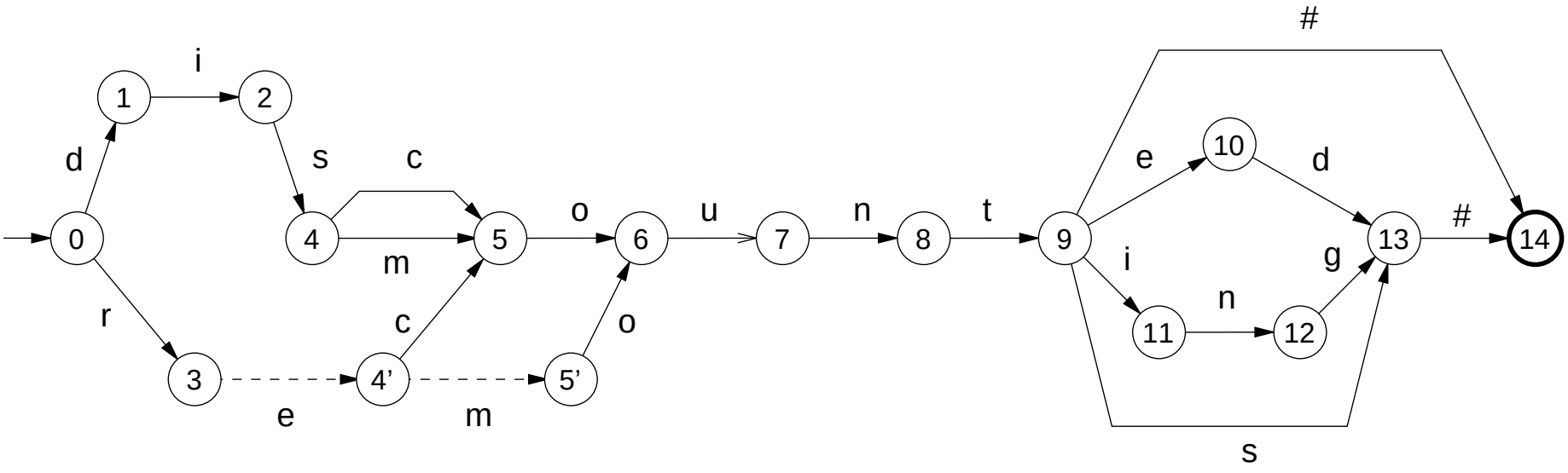
Fases de Inserción - Minimización



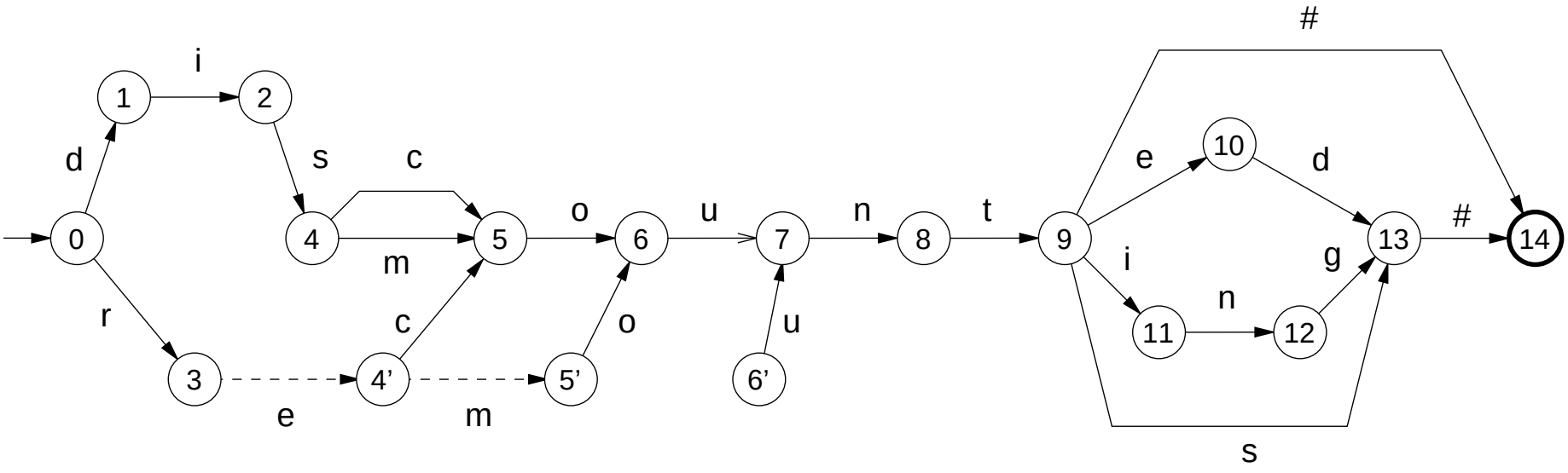
Fases de Inserción - Minimización



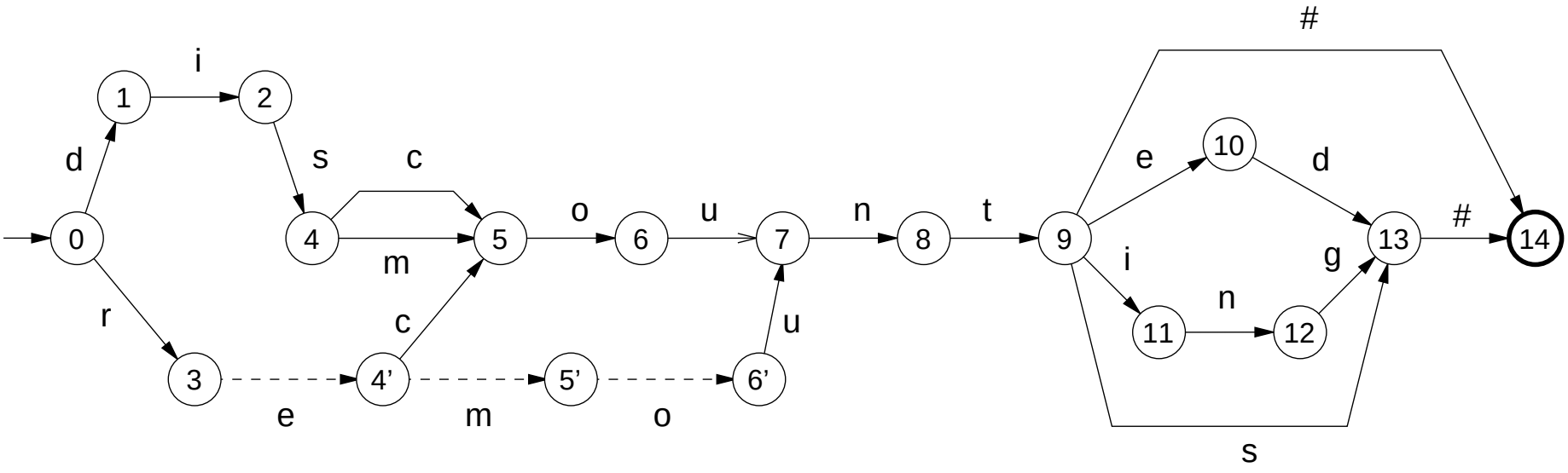
Fases de Inserción - Minimización



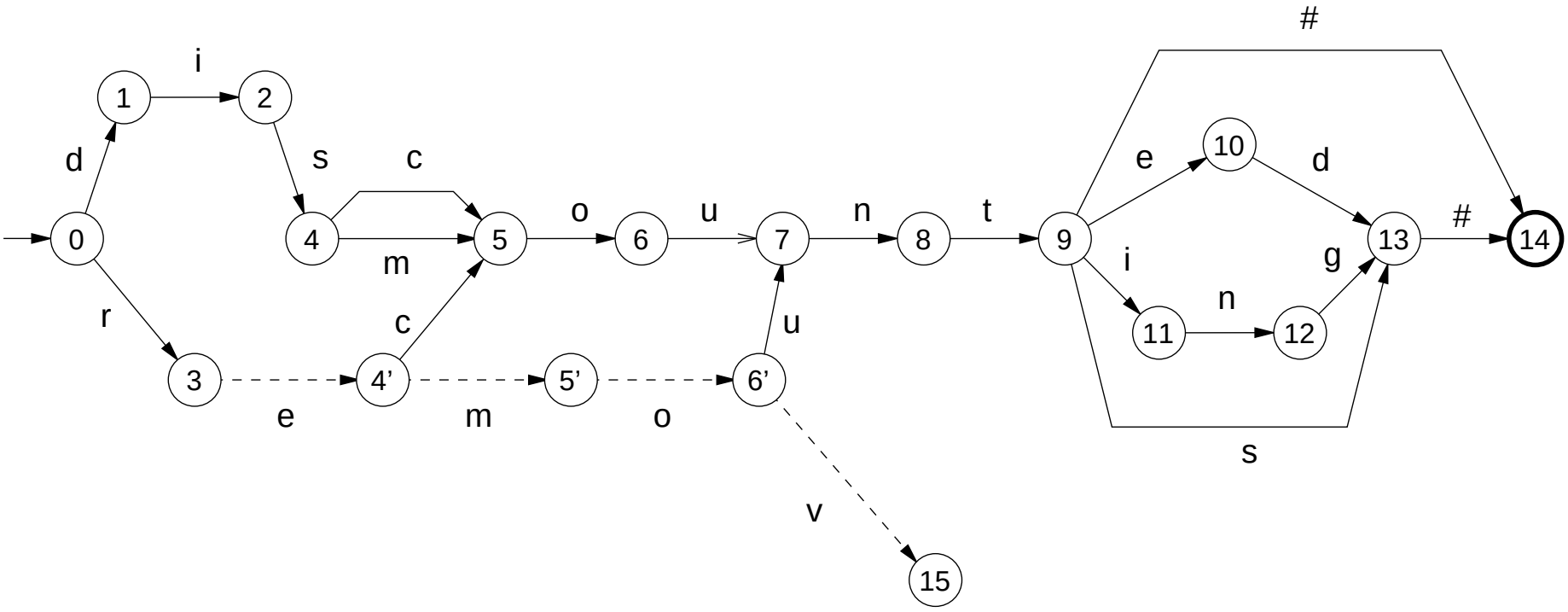
Fases de Inserción - Minimización



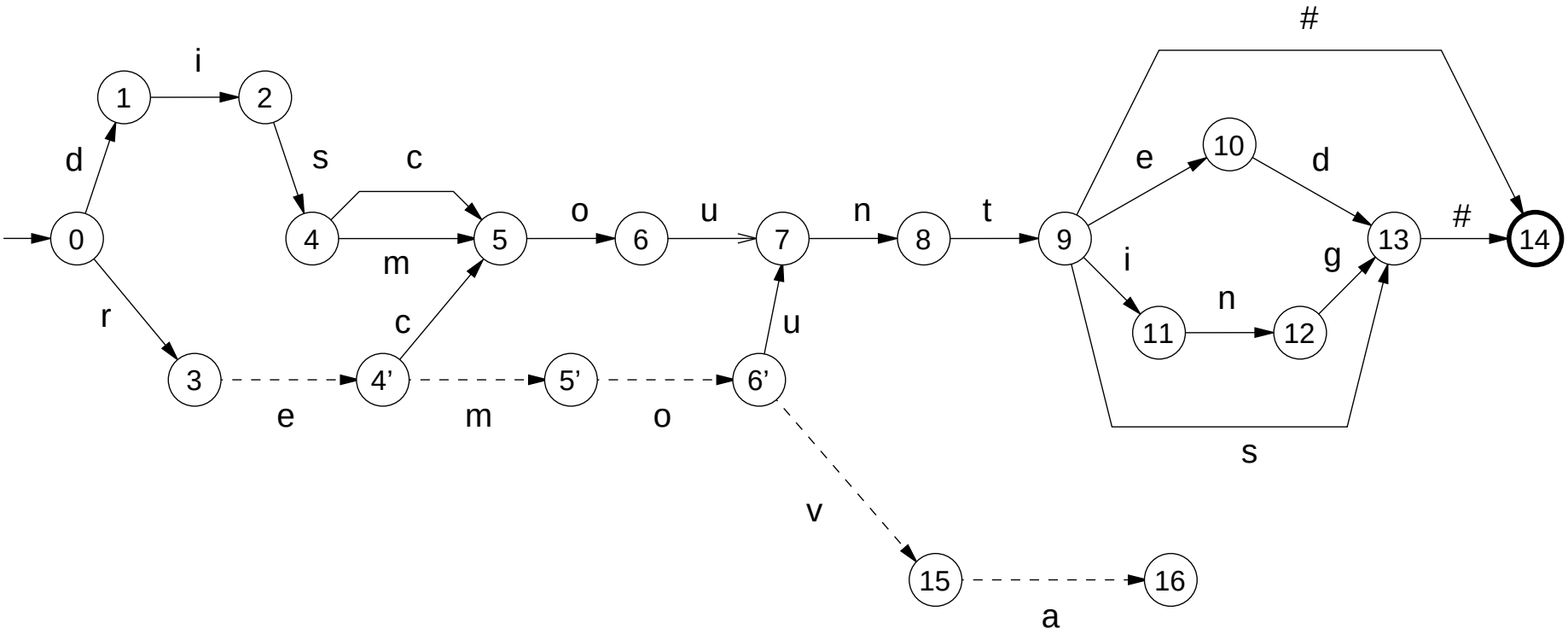
Fases de Inserción - Minimización



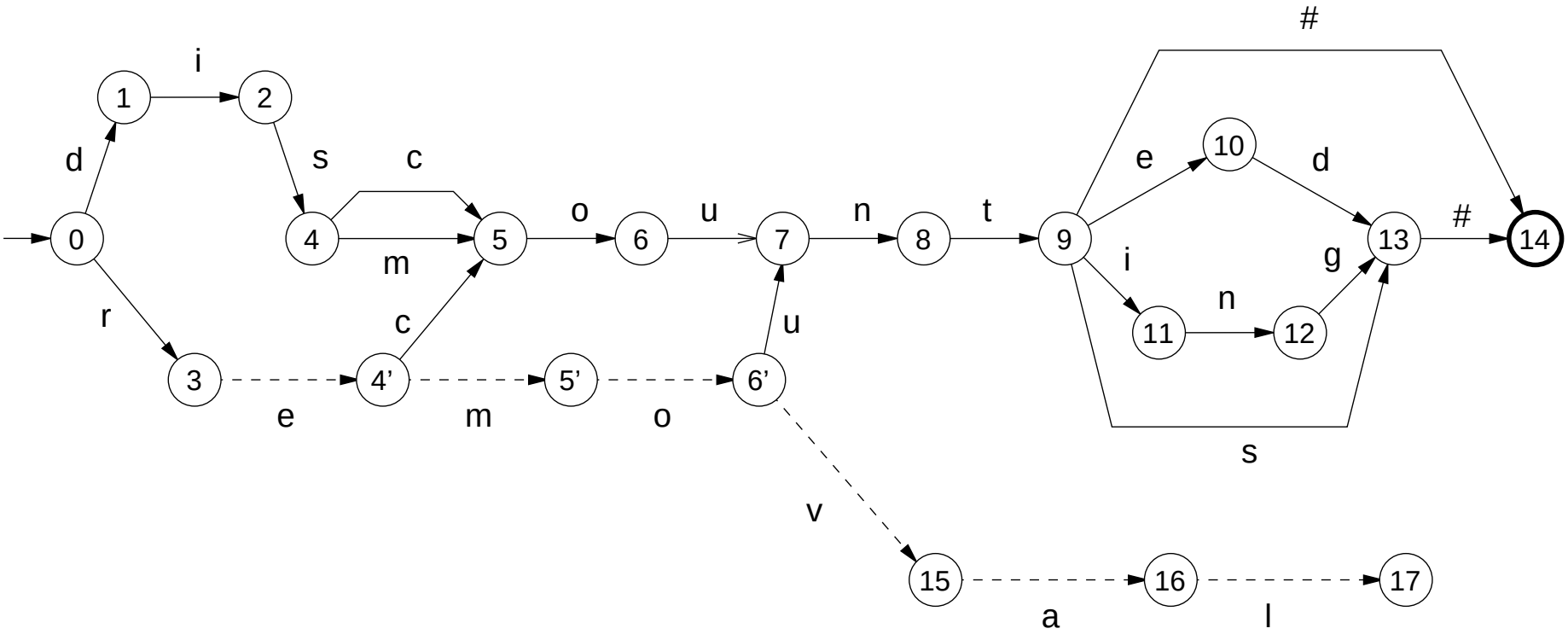
Fases de Inserción - Minimización



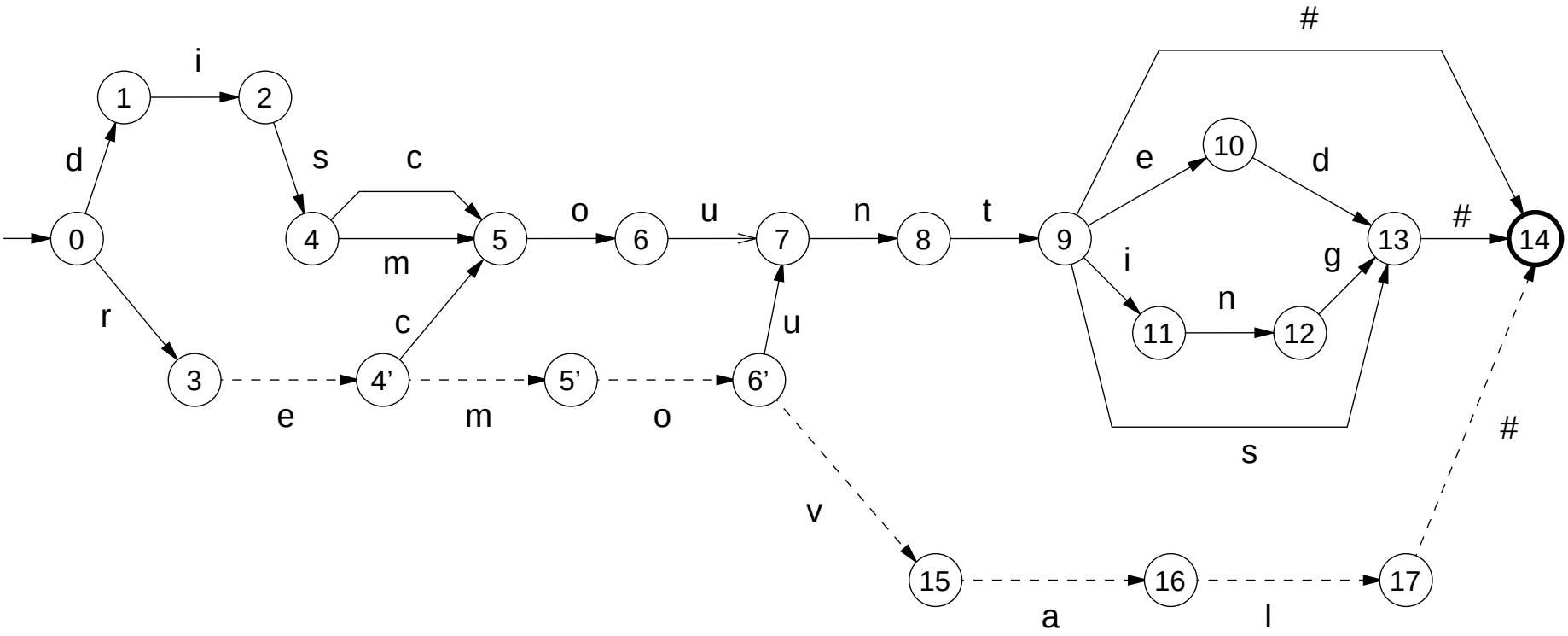
Fases de Inserción - Minimización



Fases de Inserción - Minimización



Fases de Inserción - Minimización



Fases de Inserción - Minimización

Etapa de Construcción	Palabras Insertadas	Antes de la Minimización		Después de la Minimización	
		Estados	Transiciones	Estados	Transiciones
1	34.839	65.526	100.363	2.277	5.219
2	68.356	65.527	101.986	4.641	11.431
3	100.325	65.526	104.285	6.419	16.163
4	132.094	65.530	107.043	7.377	18.560
5	163.426	65.532	108.047	8.350	21.094
6	193.368	65.525	108.211	9.858	24.877
7	223.743	65.530	110.924	10.646	27.166
8	253.703	65.527	112.007	11.221	28.850
9	283.281	65.528	112.735	11.779	30.617
10	291.604	26.987	54.148	11.985	31.258

Algoritmo de Construcción Incremental

ab

abc

abde

abdf

bb

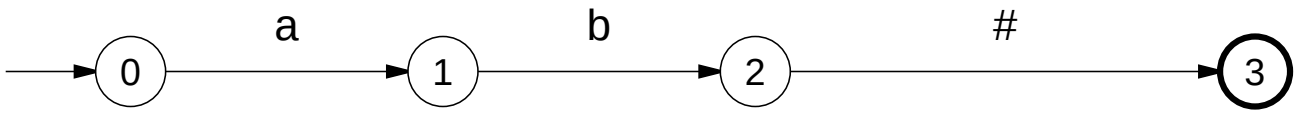
bbc

bbde

bbdf

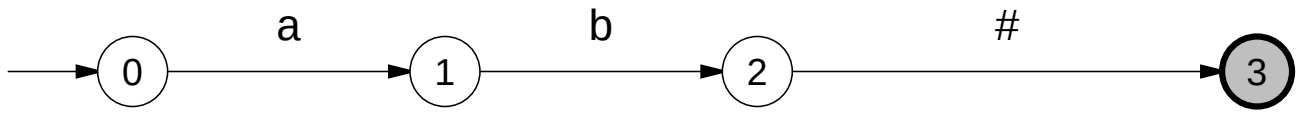
Algoritmo de Construcción Incremental

- ✓ ab
- abc
- abde
- abdf
- bb
- bbc
- bbde
- bbdf



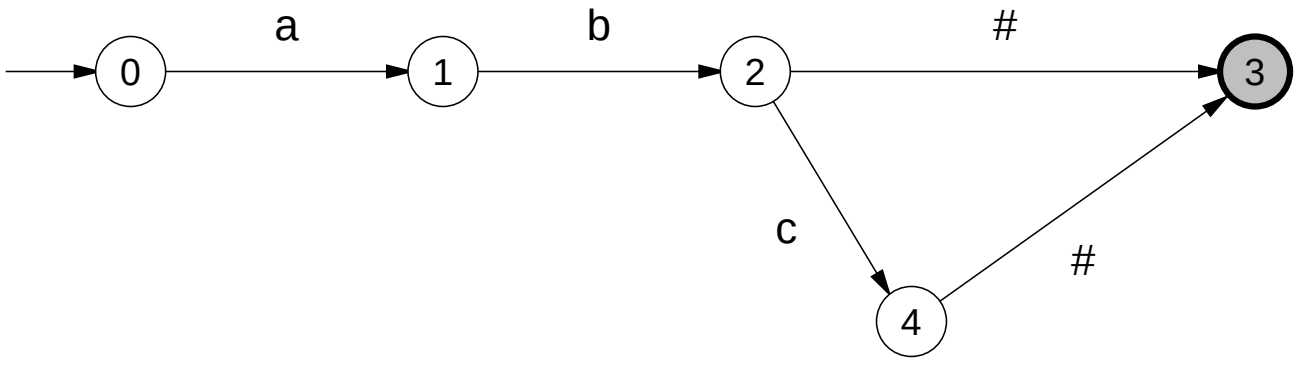
Algoritmo de Construcción Incremental

- ab
- abc
- abde
- abdf
- bb
- bbc
- bbde
- bbdf



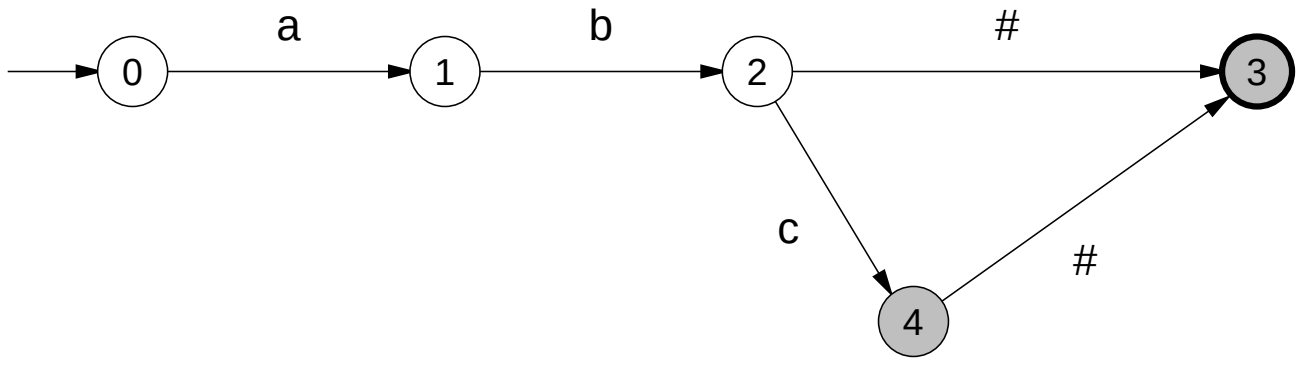
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- abde
- abdf
- bb
- bbc
- bbde
- bbdf



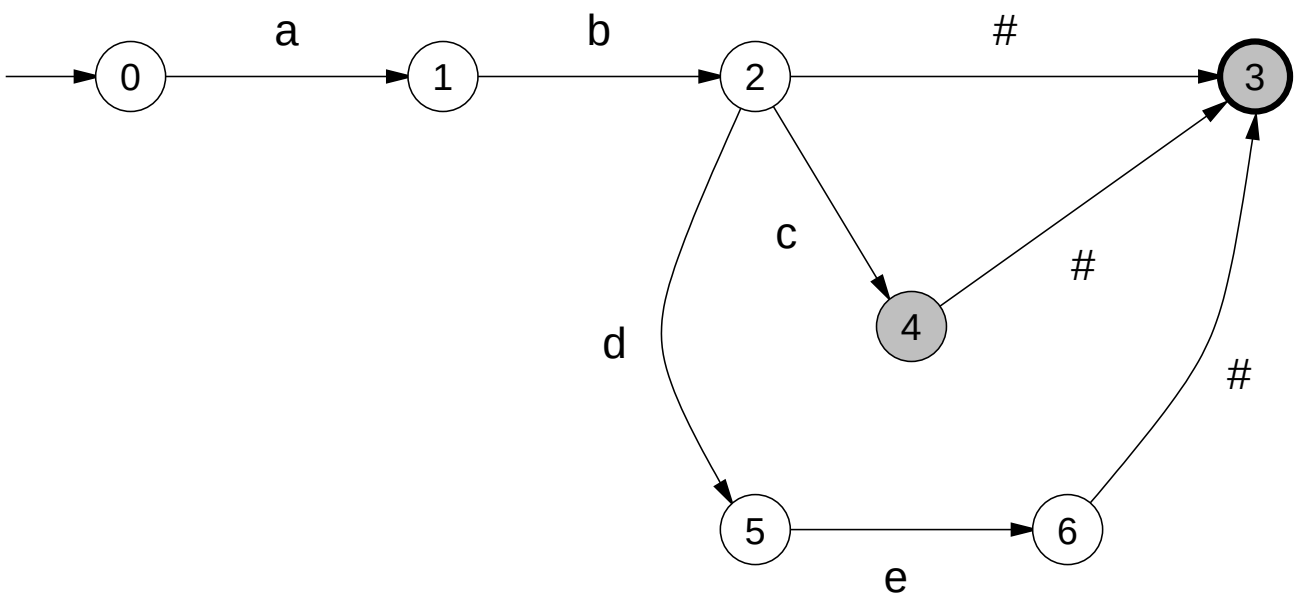
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- abde
- abdf
- bb
- bbc
- bbde
- bbdf



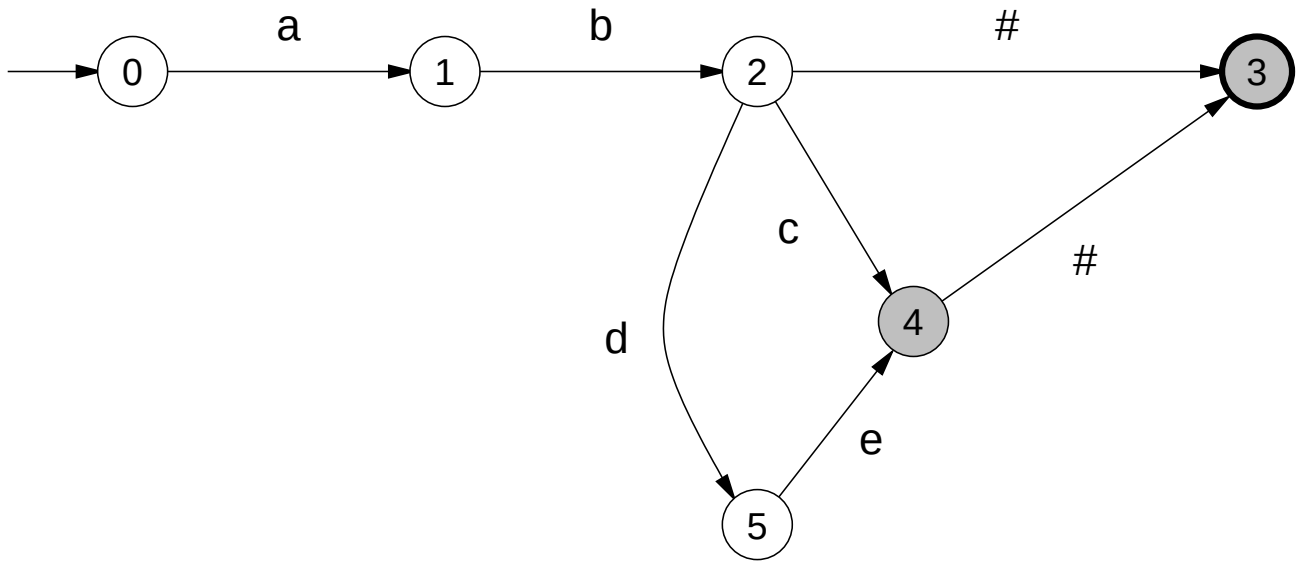
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- abdf
- bb
- bbc
- bbde
- bbdf



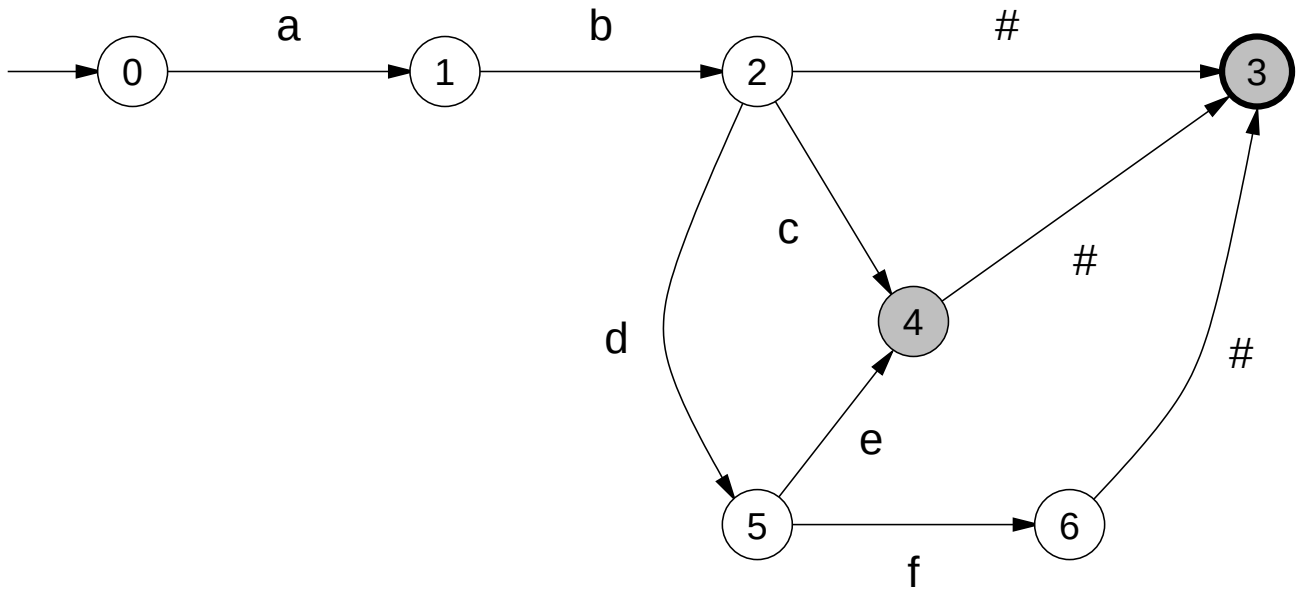
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- abdf
- bb
- bbc
- bbde
- bbdf



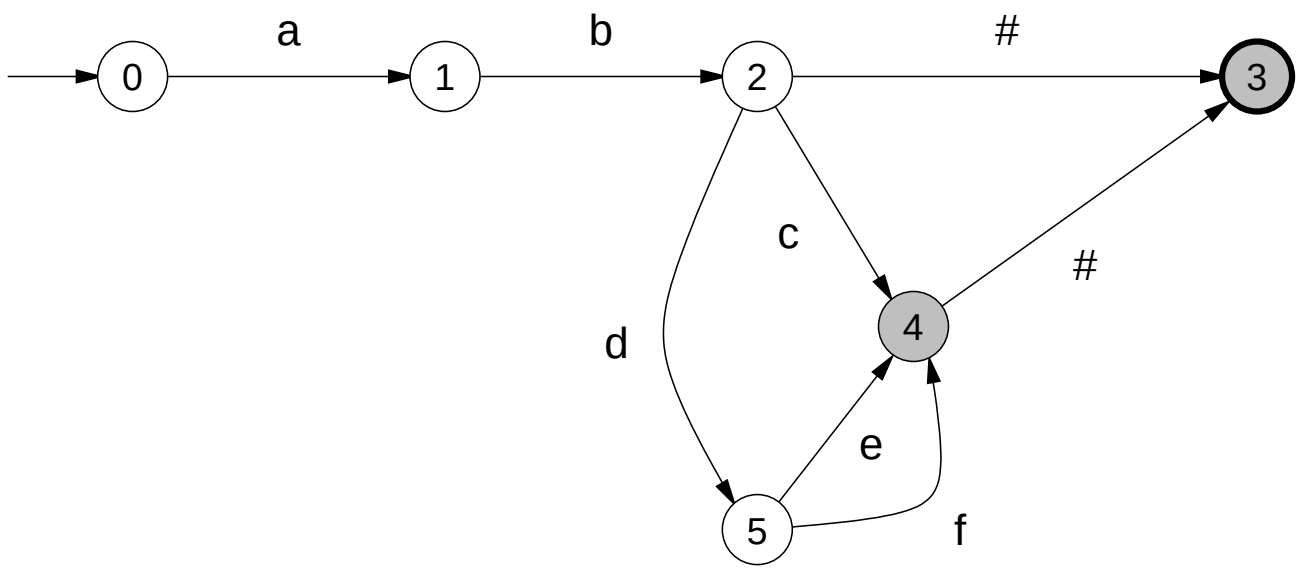
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- bb
- bbc
- bbde
- bbdf



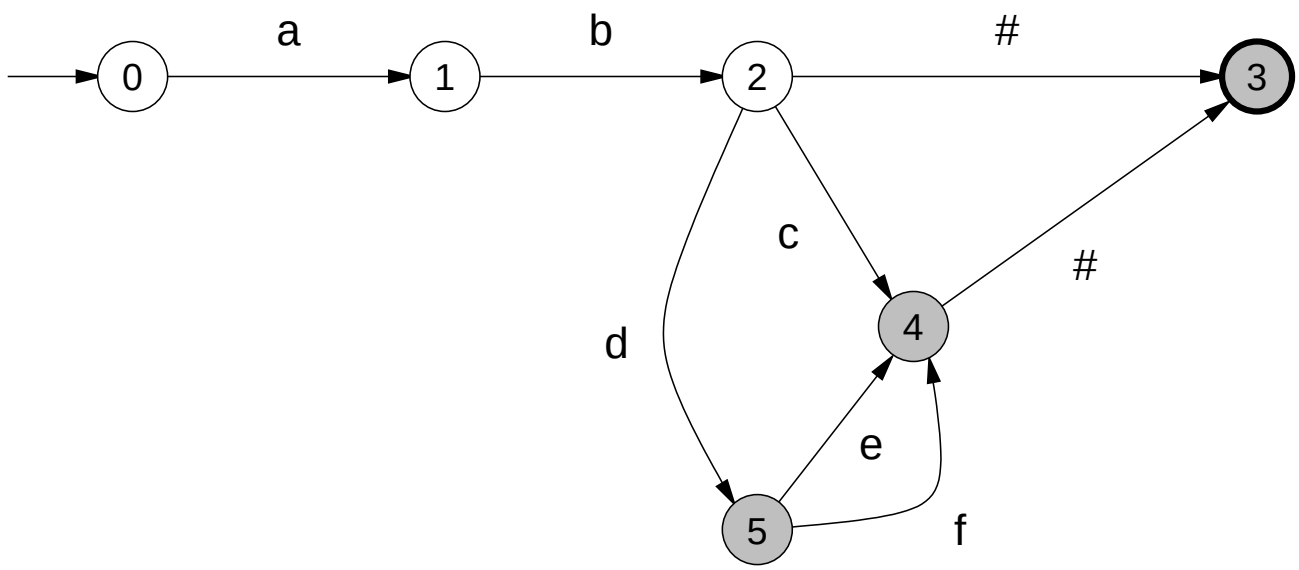
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- bb
- bbc
- bbde
- bbdf



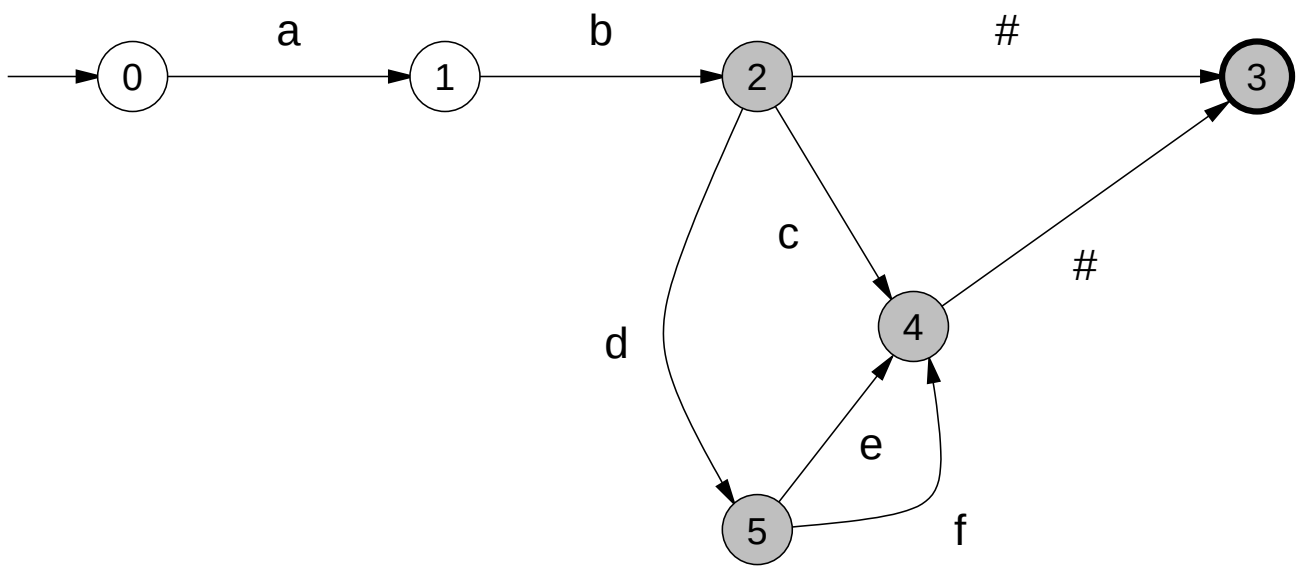
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- bb
- bbc
- bbde
- bbdf



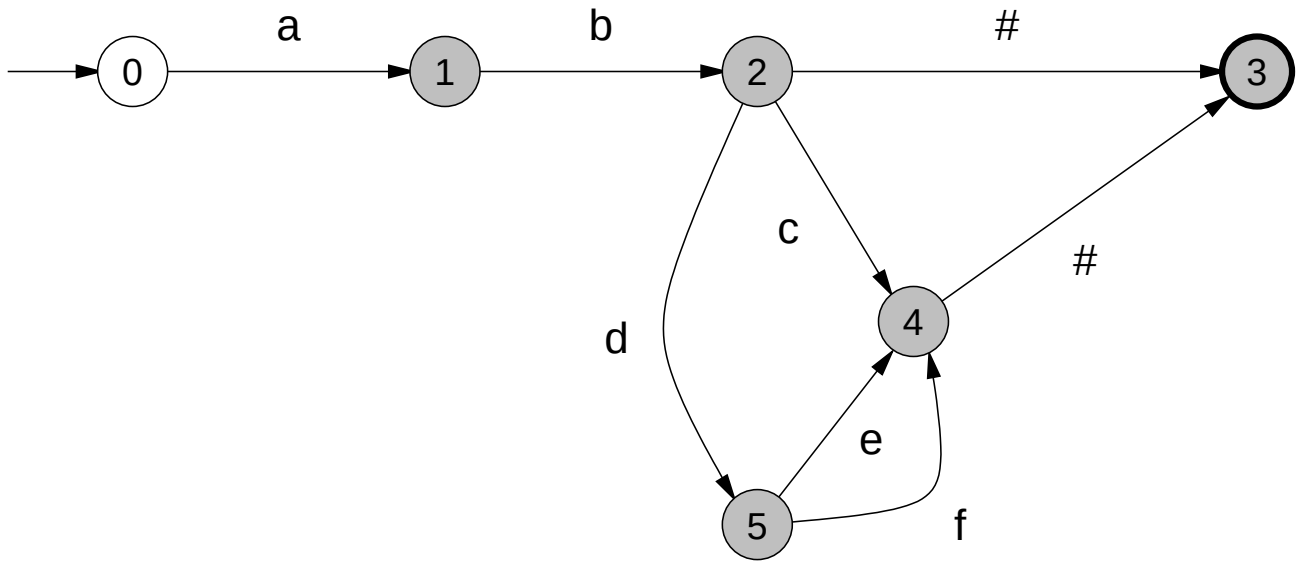
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- bb
- bbc
- bbde
- bbdf



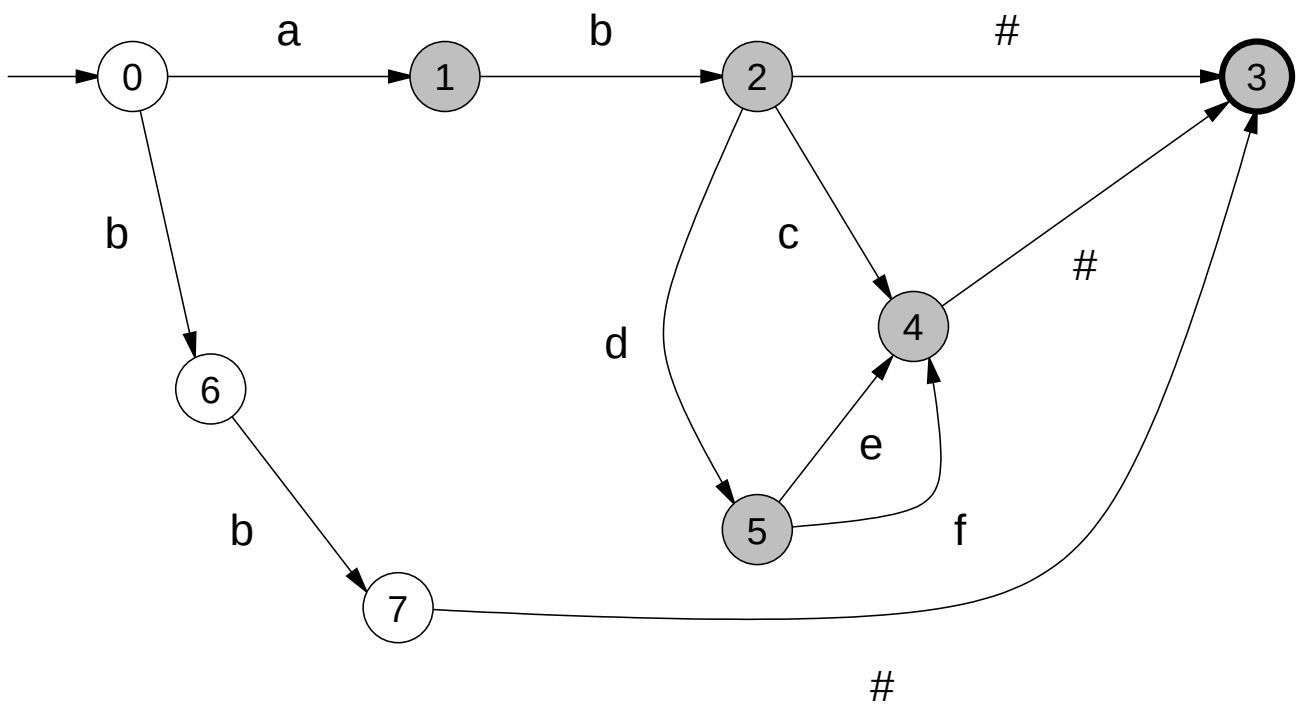
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- bb
- bbc
- bbde
- bbdf



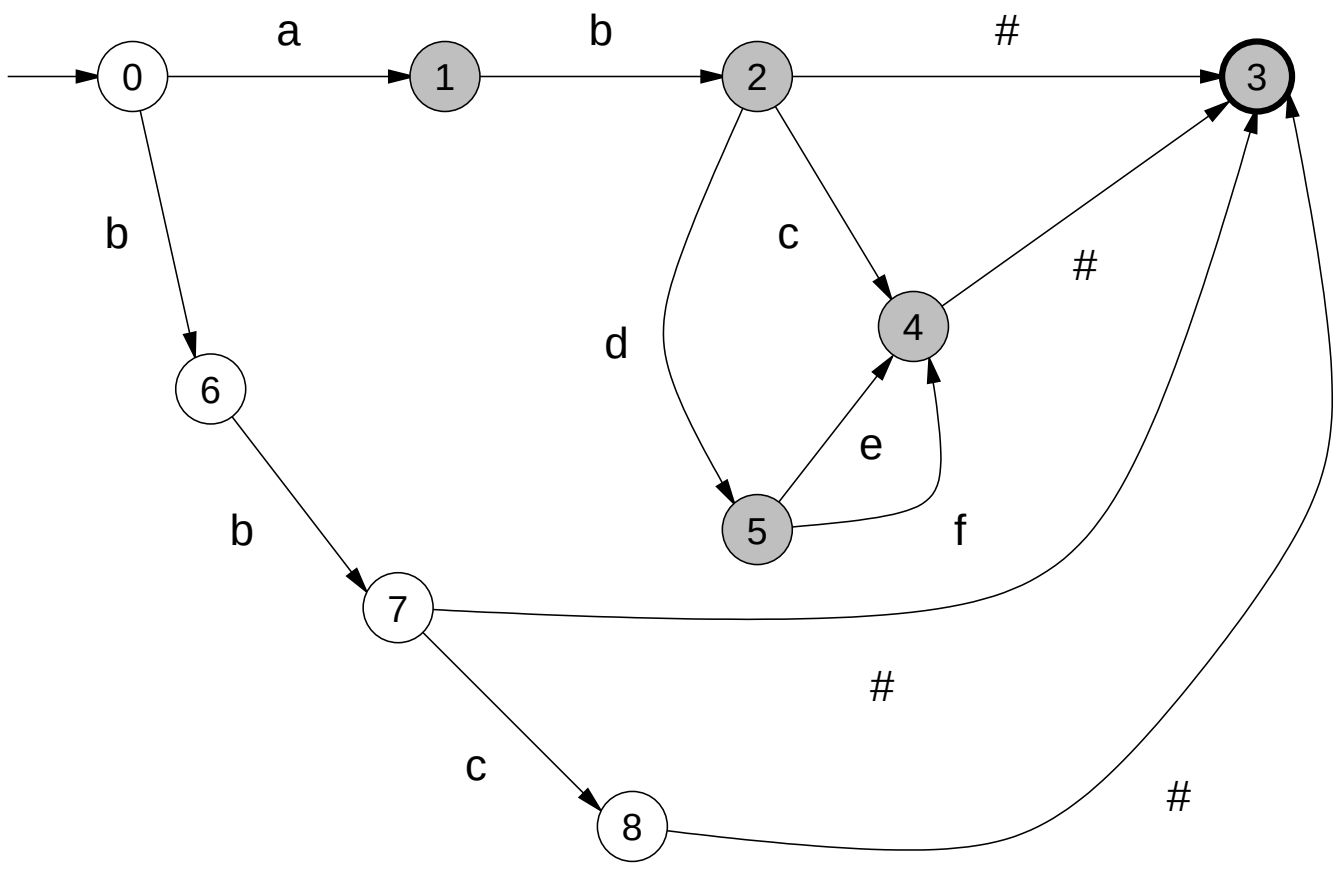
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- bbc
- bbde
- bbdf



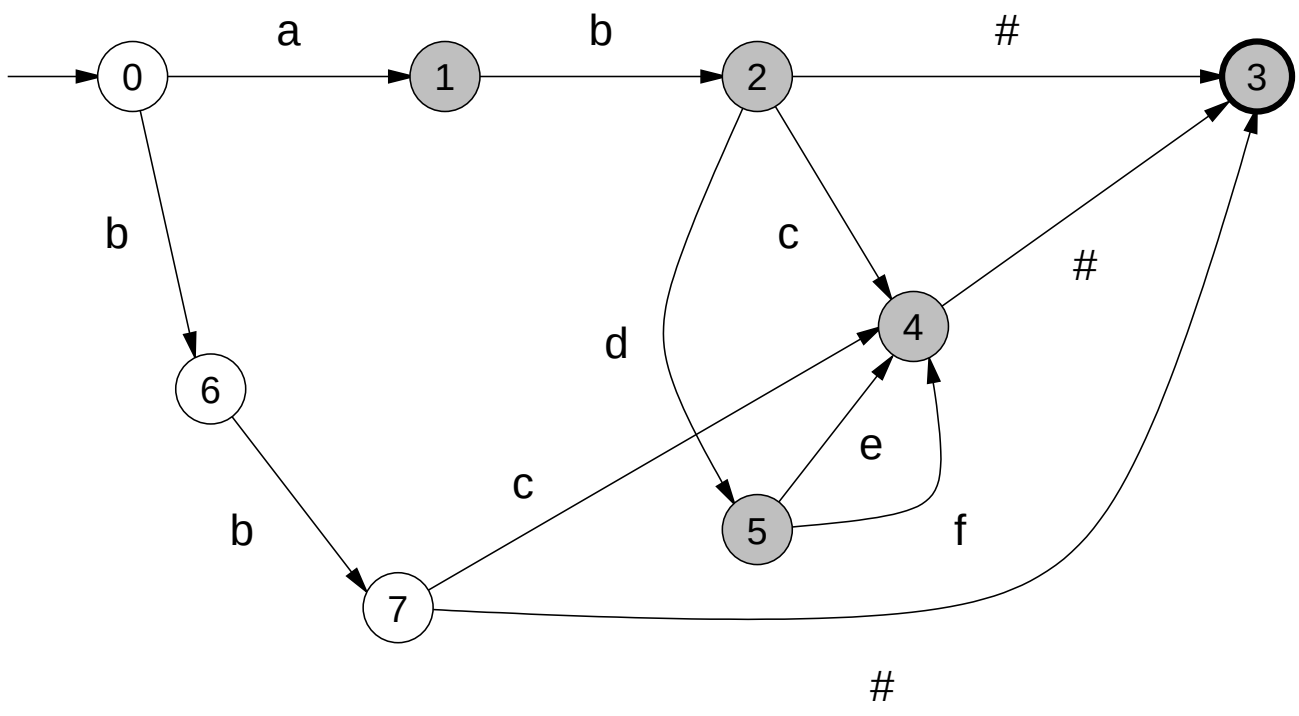
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- bbde
- bbdf



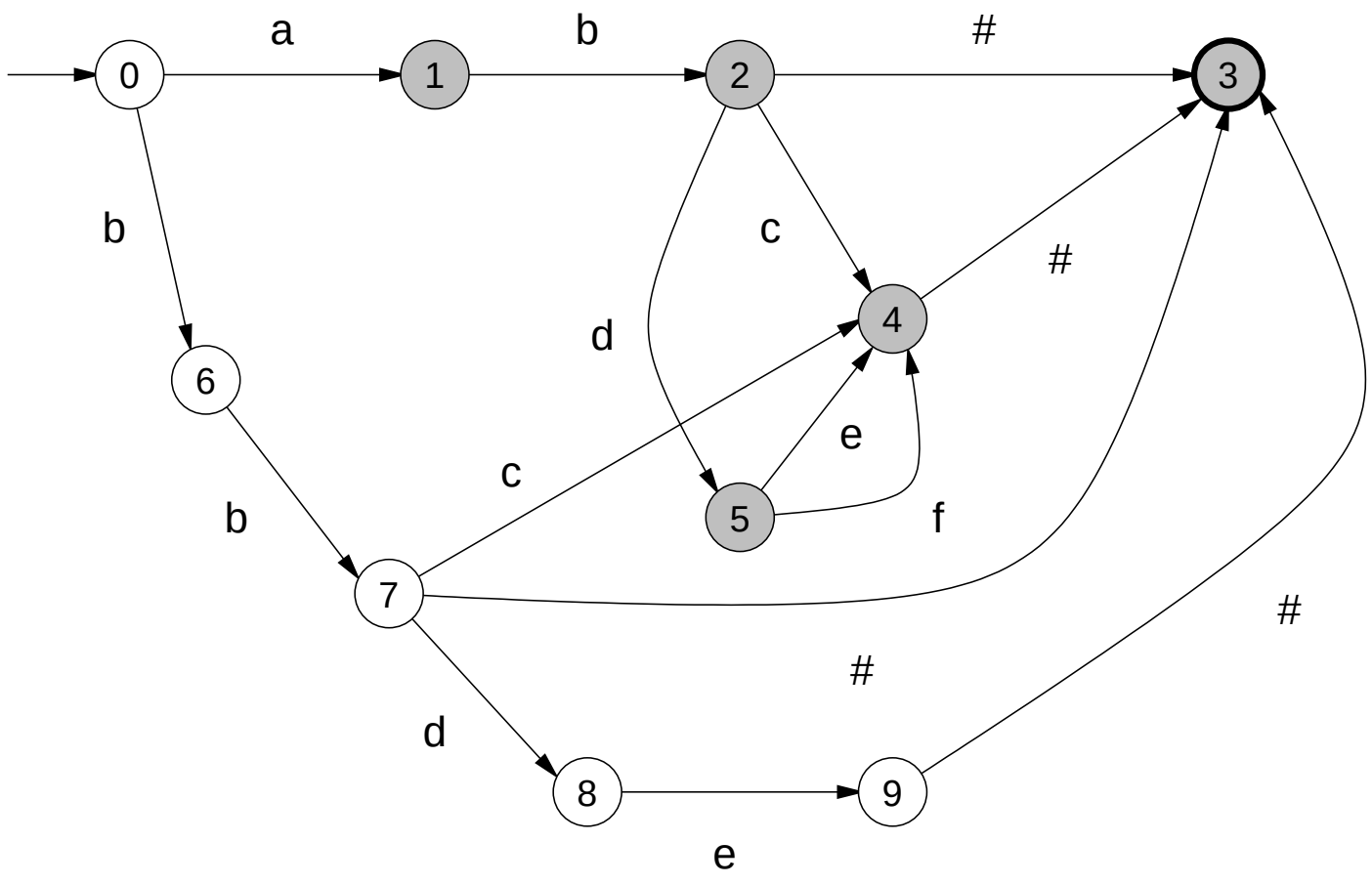
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- bbde
- bbdf



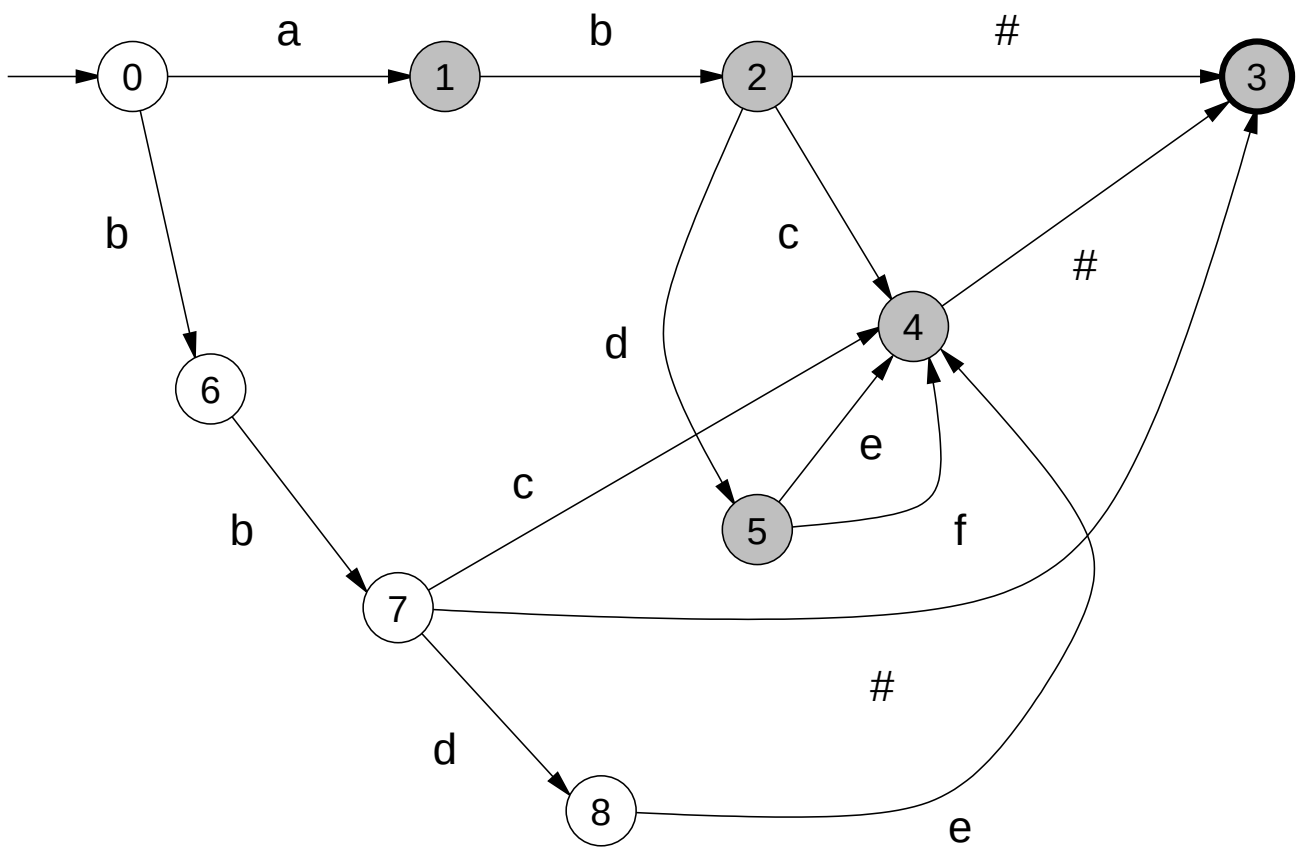
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- ✓ bbde
- bbdf



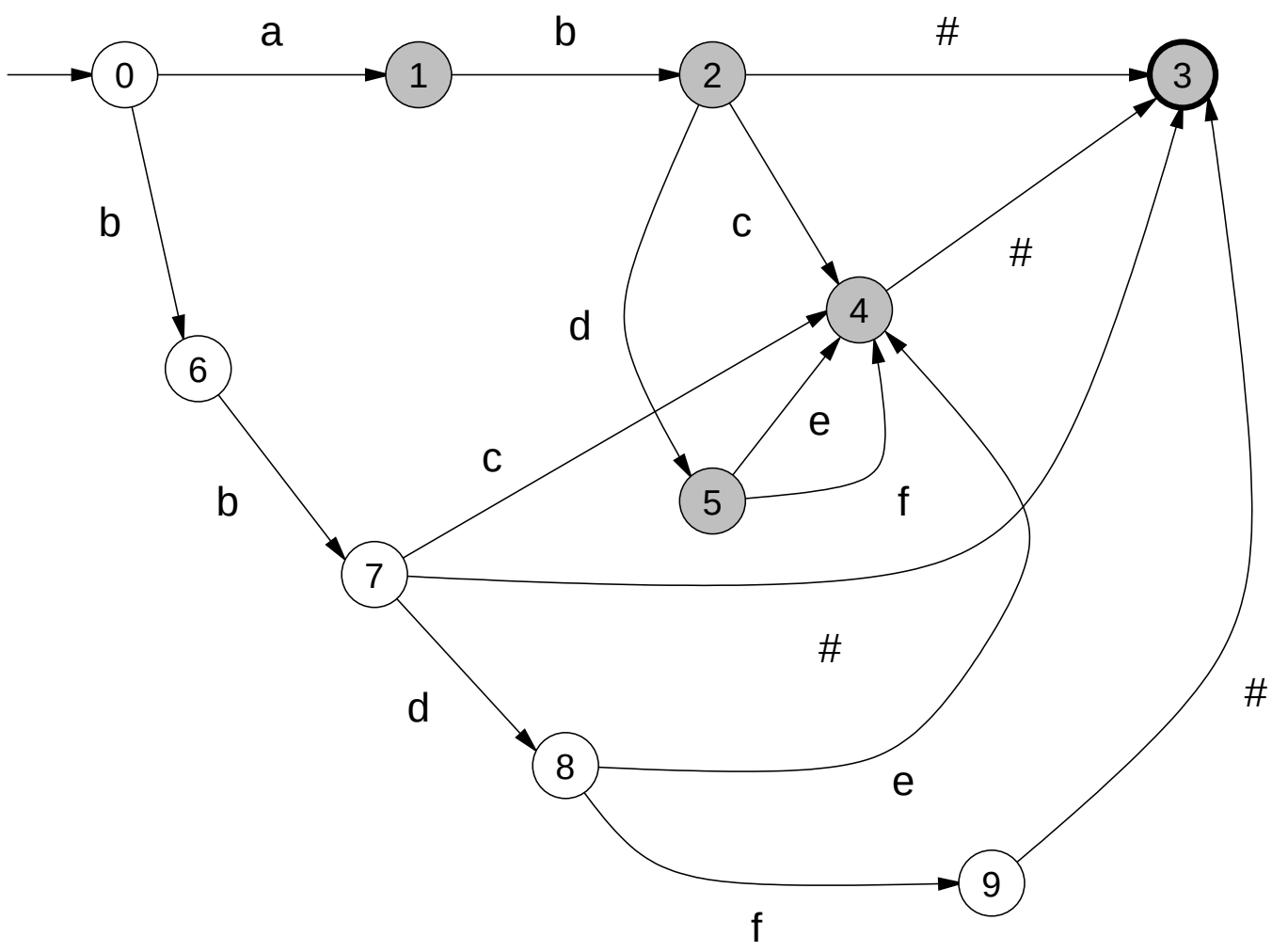
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- ✓ bbde
- bbdf



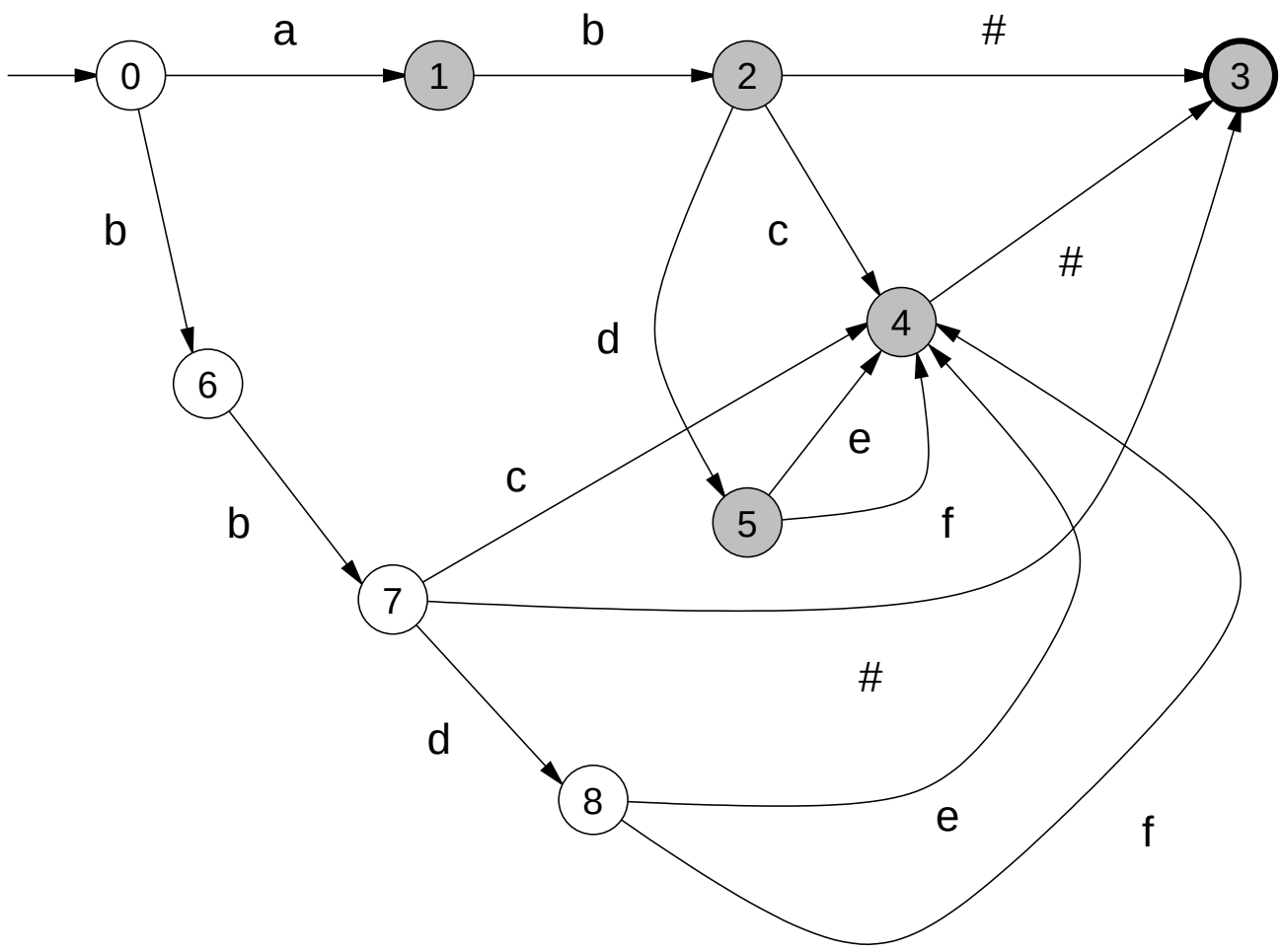
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- ✓ bbde
- ✓ bbdf



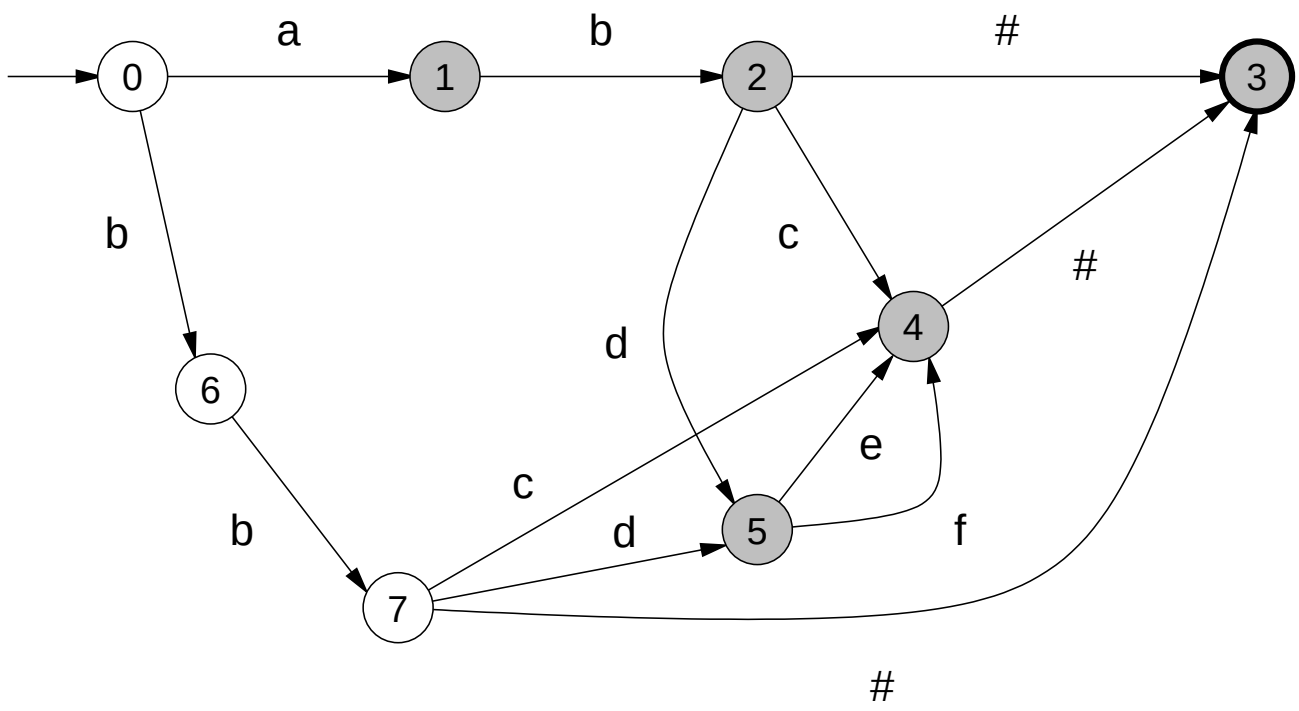
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- ✓ bbde
- ✓ bbdf



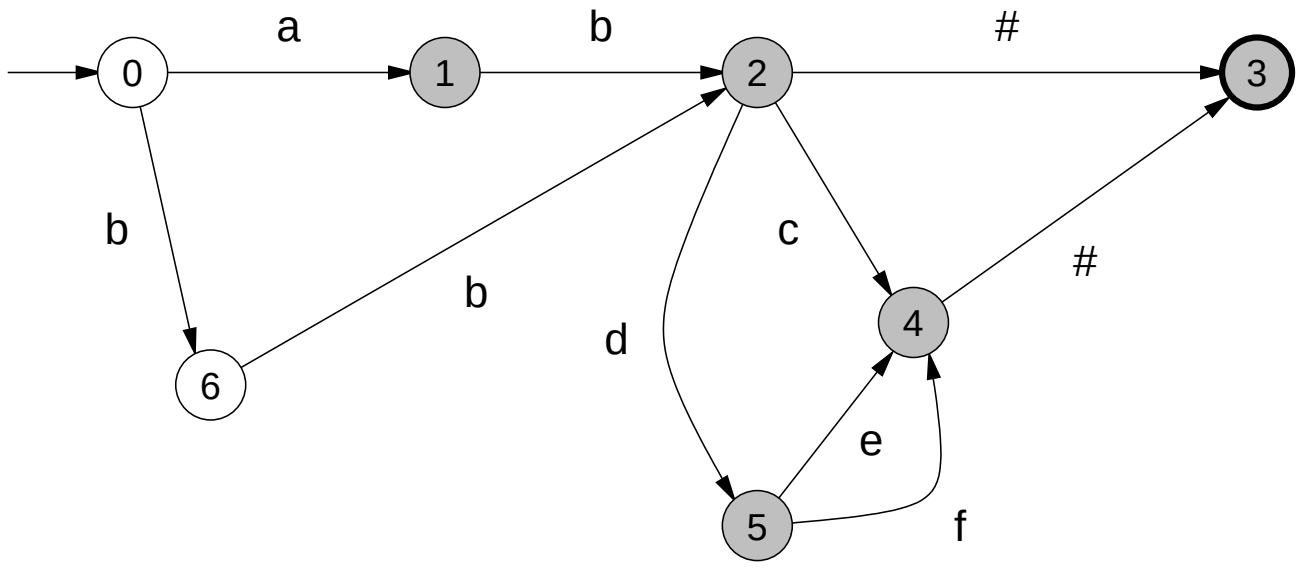
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- ✓ bbde
- ✓ bbdf



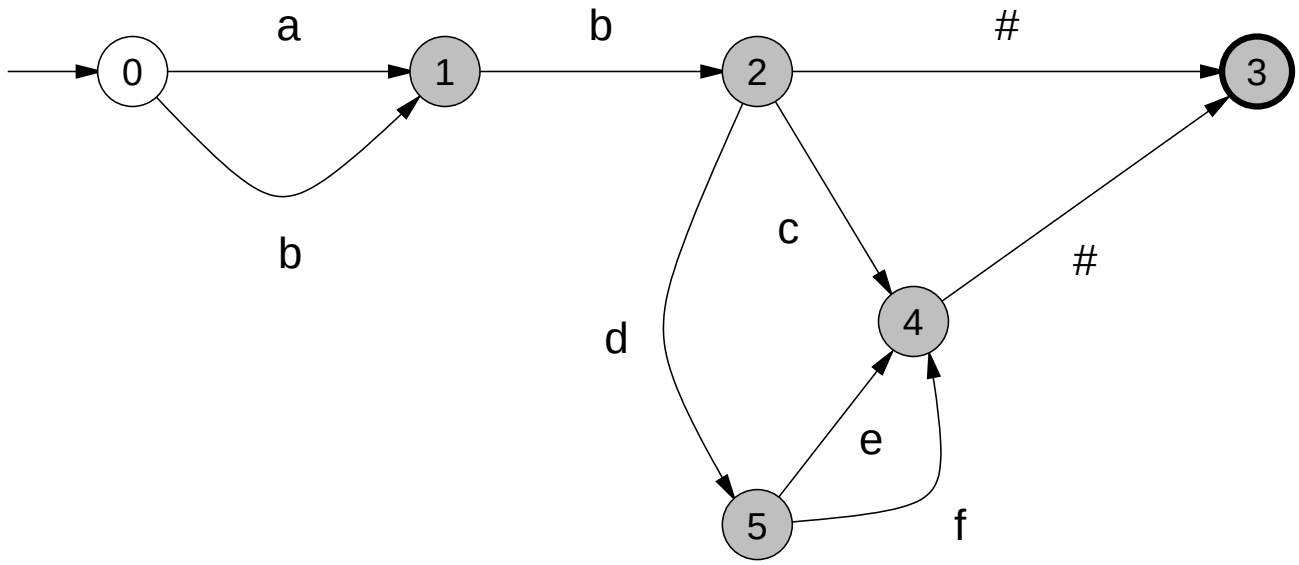
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- ✓ bbde
- ✓ bbdf



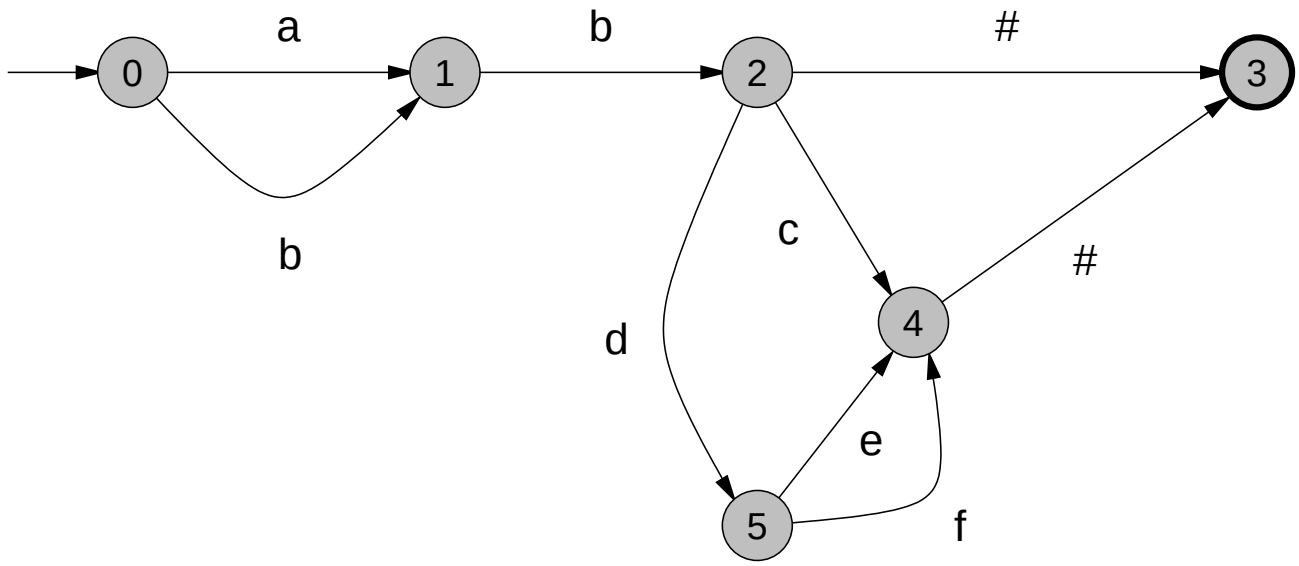
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- ✓ bbde
- ✓ bbdf



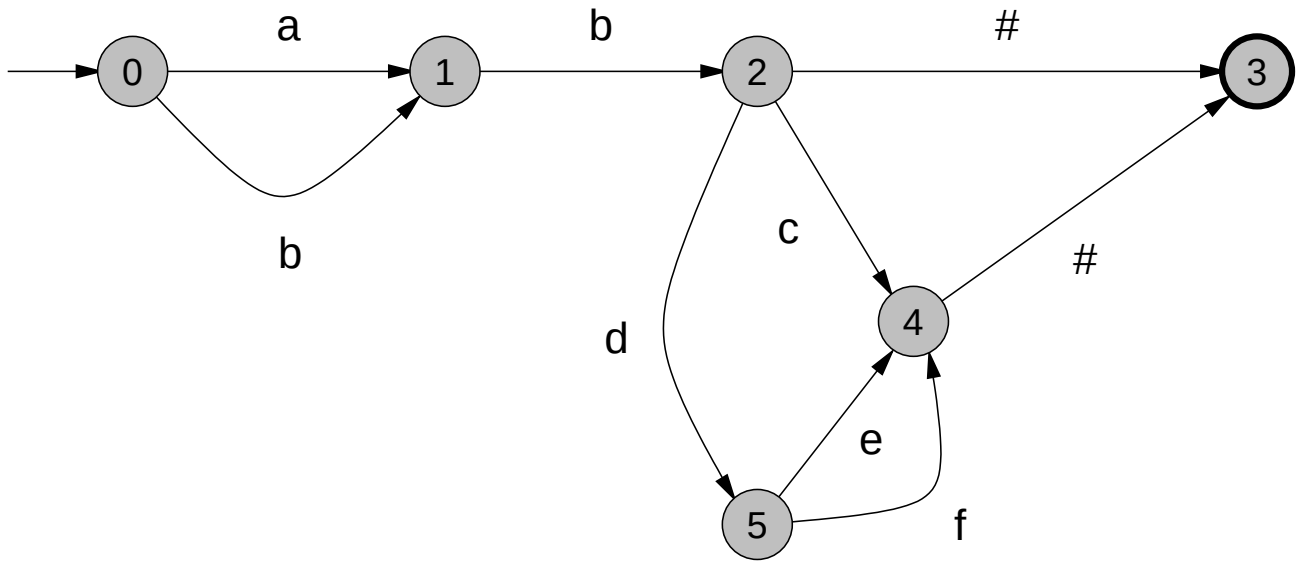
Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- ✓ bbde
- ✓ bbdf



Algoritmo de Construcción Incremental

- ✓ ab
- ✓ abc
- ✓ abde
- ✓ abdf
- ✓ bb
- ✓ bbc
- ✓ bbde
- ✓ bbdf



Complejidad Espacial: $\mathcal{O}(n)$ Complejidad Temporal: $\mathcal{O}(\log n)$ [Daciuk *et al*. 2000]

Contenidos

- **Introducción:**
  - Lenguajes Naturales, Análisis Léxico y Autómatas Finitos Acíclicos.
- **Implementación de Grandes Diccionarios:**
  - Modelización Compacta de un Diccionario.
  - Autómatas Finitos Acíclicos Mínimos Numerados.
- **Construcción de Autómatas Acíclicos:**
  - Algoritmo General de Minimización.
  - Minimización Basada en la Propiedad de la Altura.
  - Algoritmo de Construcción Incremental.
- **Nuevas Funcionalidades:**
  - Mejoras en el Acceso al Registro.
  - Comportamiento Dinámico de los Autómatas Acíclicos.
- **Conclusiones.**

Mejoras en el Acceso al Registro

¿Cuándo dos estados son equivalentes? Cada una de las posibles respuestas constituye un nuevo filtro que deja más y más estados fuera del proceso de comparación:

Mejoras en el Acceso al Registro

¿Cuándo dos estados son equivalentes? Cada una de las posibles respuestas constituye un nuevo filtro que deja más y más estados fuera del proceso de comparación:

- Para que dos estados sean equivalentes, sus alturas deben ser iguales.
  Las alturas son números bajos (entre 0 y la longitud de la palabra más larga).
  La altura de un estado es la máxima altura de los estados destino de sus arcos más 1.

Mejoras en el Acceso al Registro

¿Cuándo dos estados son equivalentes? Cada una de las posibles respuestas constituye un nuevo filtro que deja más y más estados fuera del proceso de comparación:

- Para que dos estados sean equivalentes, sus alturas deben ser iguales.
  Las alturas son números bajos (entre 0 y la longitud de la palabra más larga).
  La altura de un estado es la máxima altura de los estados destino de sus arcos más 1.

- El número de arcos salientes también debe ser igual.
  El número de arcos también es un número bajo (entre 1 y el tamaño del alfabeto).

Mejoras en el Acceso al Registro

¿Cuándo dos estados son equivalentes? Cada una de las posibles respuestas constituye un nuevo filtro que deja más y más estados fuera del proceso de comparación:

- Para que dos estados sean equivalentes, sus alturas deben ser iguales.
  Las alturas son números bajos (entre 0 y la longitud de la palabra más larga).
  La altura de un estado es la máxima altura de los estados destino de sus arcos más 1.

- El número de arcos salientes también debe ser igual.
  El número de arcos también es un número bajo (entre 1 y el tamaño del alfabeto).

- Los pesos de indexación también deben ser iguales.
  Los pesos pueden ser números muy altos (entre 1 y el tamaño del diccionario), pero se puede demostrar empíricamente que los pesos más frecuentes también son bajos.
  El peso de un estado es la suma de los pesos de los estados destino de sus arcos.

Mejoras en el Acceso al Registro

¿Cuándo dos estados son equivalentes? Cada una de las posibles respuestas constituye un nuevo filtro que deja más y más estados fuera del proceso de comparación:

- Para que dos estados sean equivalentes, sus alturas deben ser iguales.
  Las alturas son números bajos (entre 0 y la longitud de la palabra más larga).
  La altura de un estado es la máxima altura de los estados destino de sus arcos más 1.

- El número de arcos salientes también debe ser igual.
  El número de arcos también es un número bajo (entre 1 y el tamaño del alfabeto).

- Los pesos de indexación también deben ser iguales.
  Los pesos pueden ser números muy altos (entre 1 y el tamaño del diccionario), pero se puede demostrar empíricamente que los pesos más frecuentes también son bajos.
  El peso de un estado es la suma de los pesos de los estados destino de sus arcos.

Nuestra implementación del **registro** es un **array tridimensional** al que se accede mediante la **altura**, el **número de arcos salientes** y el **peso de indexación**. Cada celda contiene la lista de estados que comparten estas tres características. [Graña *et al*. 2001]

Mejoras en el Acceso al Registro

¿Cuándo dos estados son equivalentes? Cada una de las posibles respuestas constituye un nuevo filtro que deja más y más estados fuera del proceso de comparación:

- Para que dos estados sean equivalentes, sus alturas deben ser iguales.
  Las alturas son números bajos (entre 0 y la longitud de la palabra más larga).
  La altura de un estado es la máxima altura de los estados destino de sus arcos más 1.

- El número de arcos salientes también debe ser igual.
  El número de arcos también es un número bajo (entre 1 y el tamaño del alfabeto).

- Los pesos de indexación también deben ser iguales.
  Los pesos pueden ser números muy altos (entre 1 y el tamaño del diccionario), pero se puede demostrar empíricamente que los pesos más frecuentes también son bajos.
  El peso de un estado es la suma de los pesos de los estados destino de sus arcos.

Nuestra implementación del **registro** es un **array tridimensional** al que se accede mediante la **altura**, el **número de arcos salientes** y el **peso de indexación**. Cada celda contiene la lista de estados que comparten estas tres características. [Graña *et al*. 2001]

Finalmente, se verifican las etiquetas de los arcos salientes y sus estados destino.

Análisis de Resultados

Diccionario	Tamaño del Diccionario		Tamaño del Autómata	
	Palabras	Etiquetaciones	Estados	Arcos
GALENA	291.604	354.007	11.985	31.258
ERIAL	775.621	993.703	52.861	159.780

Análisis de Resultados

Diccionario	Tamaño del Diccionario		Tamaño del Autómata	
	Palabras	Etiquetaciones	Estados	Arcos
GALENA	291.604	354.007	11.985	31.258
ERIAL	775.621	993.703	52.861	159.780

Diccionario	Tiempos de Compilación (en segundos)		
	Revuz	Daciuk *et al*.	Graña *et al*.
GALENA	29,3	3,4	**2,5** (+ 4,6 = 7,1)
ERIAL	141,7	11,2	**9,2** (+ 15,6 = 24,8)

Análisis de Resultados

Diccionario	Tamaño del Diccionario		Tamaño del Autómata	
	Palabras	Etiquetaciones	Estados	Arcos
GALENA	291.604	354.007	11.985	31.258
ERIAL	775.621	993.703	52.861	159.780

Diccionario	Tiempos de Compilación (en segundos)		
	Revuz	Daciuk *et al*.	Graña *et al*.
GALENA	29,3	3,4	**2,5** (+ 4,6 = 7,1)
ERIAL	141,7	11,2	**9,2** (+ 15,6 = 24,8)

Velocidad de Reconocimiento (en palabras por segundo)	
Daciuk *et al*.	Graña *et al*.
35.000	**80.000**

Comportamiento Dinámico

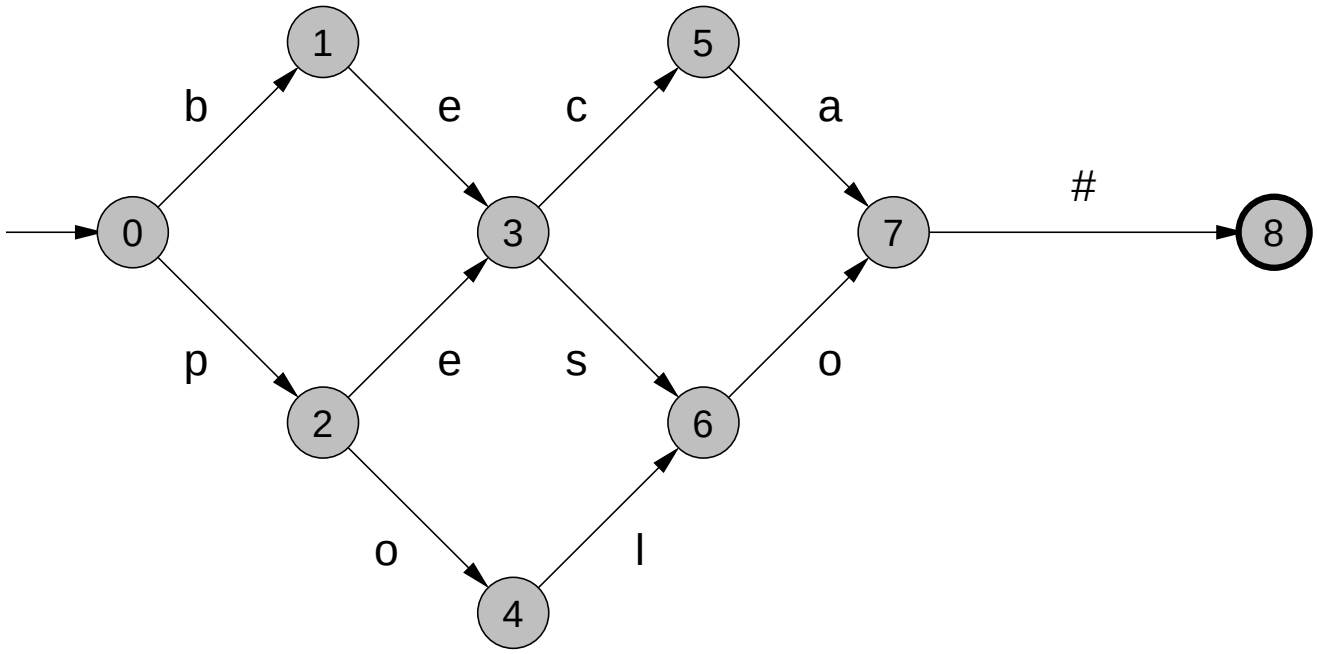
Operaciones de mantenimiento sobre autómatas finitos acíclicos:

- Inserción de nuevas palabras [Daciuk *et al*. 2000].
- Eliminación de palabras ya insertadas [Barcala y Graña 2003].

Comportamiento Dinámico

Operaciones de mantenimiento sobre autómatas finitos acíclicos:

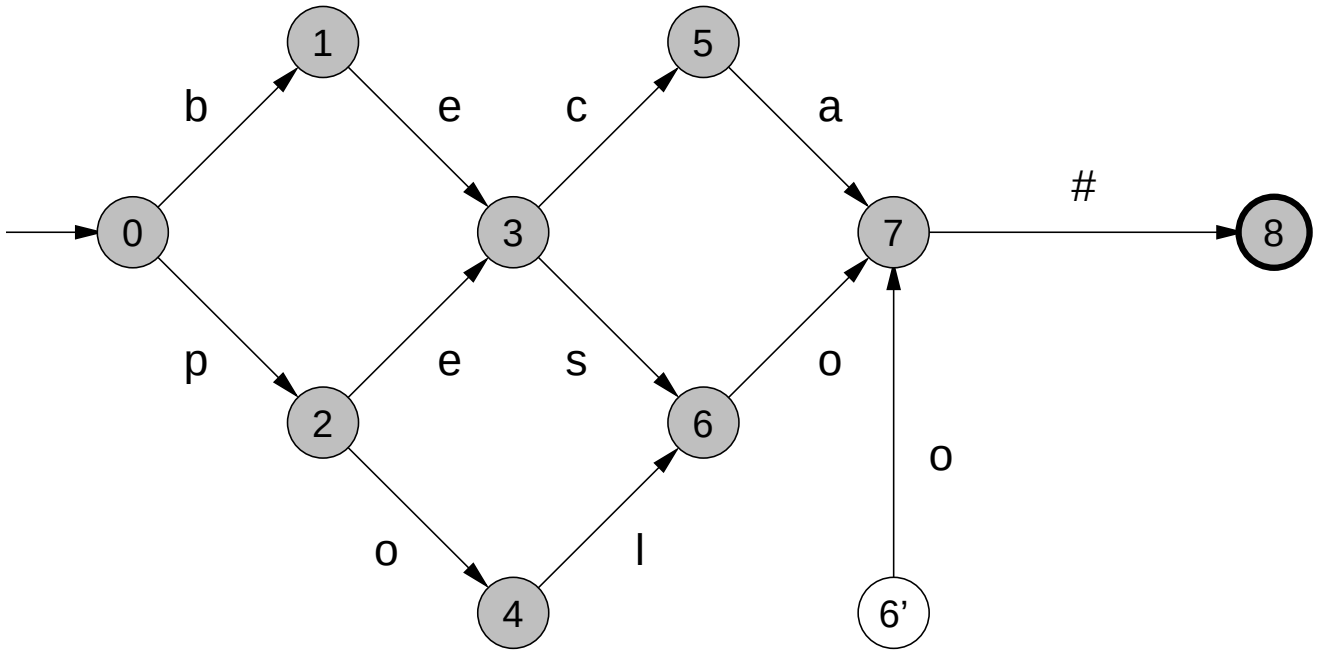
- Inserción de nuevas palabras [Daciuk *et al*. 2000].
- Eliminación de palabras ya insertadas [Barcala y Graña 2003].



Comportamiento Dinámico

Operaciones de mantenimiento sobre autómatas finitos acíclicos:

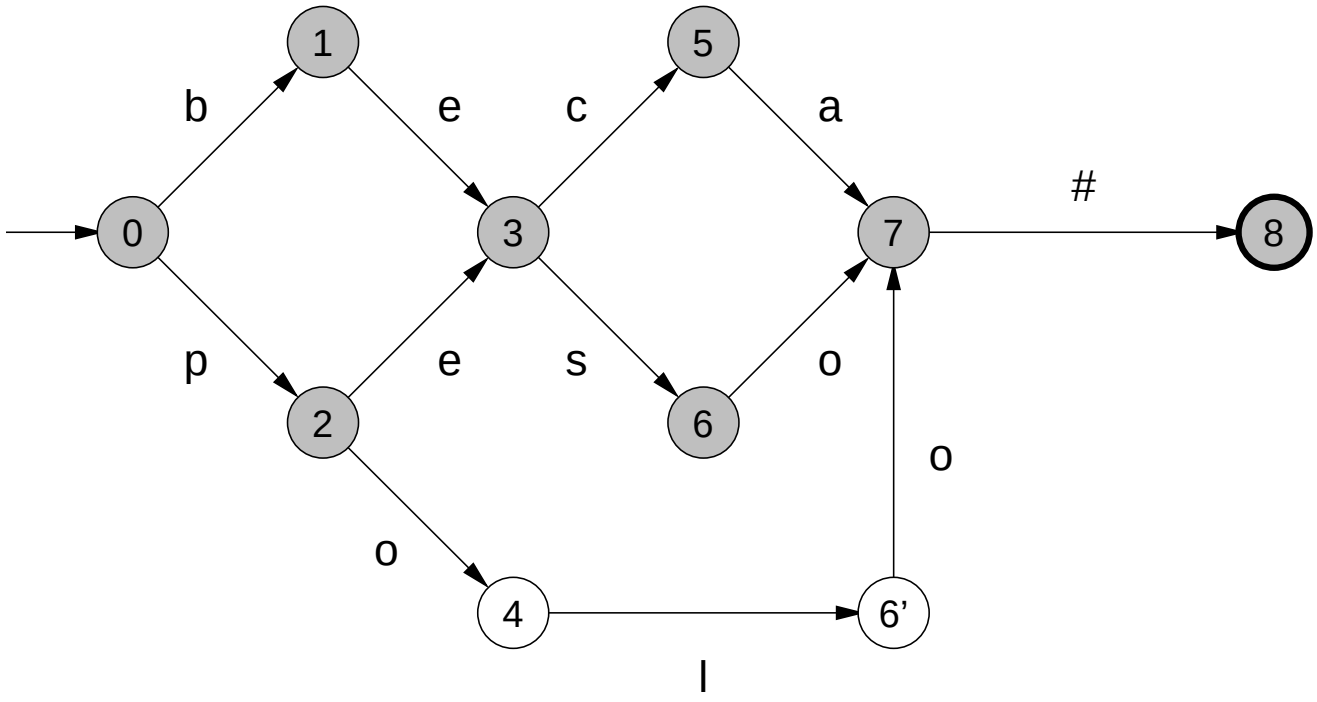
- Inserción de nuevas palabras [Daciuk *et al*. 2000].
- Eliminación de palabras ya insertadas [Barcala y Graña 2003].



Comportamiento Dinámico

Operaciones de mantenimiento sobre autómatas finitos acíclicos:

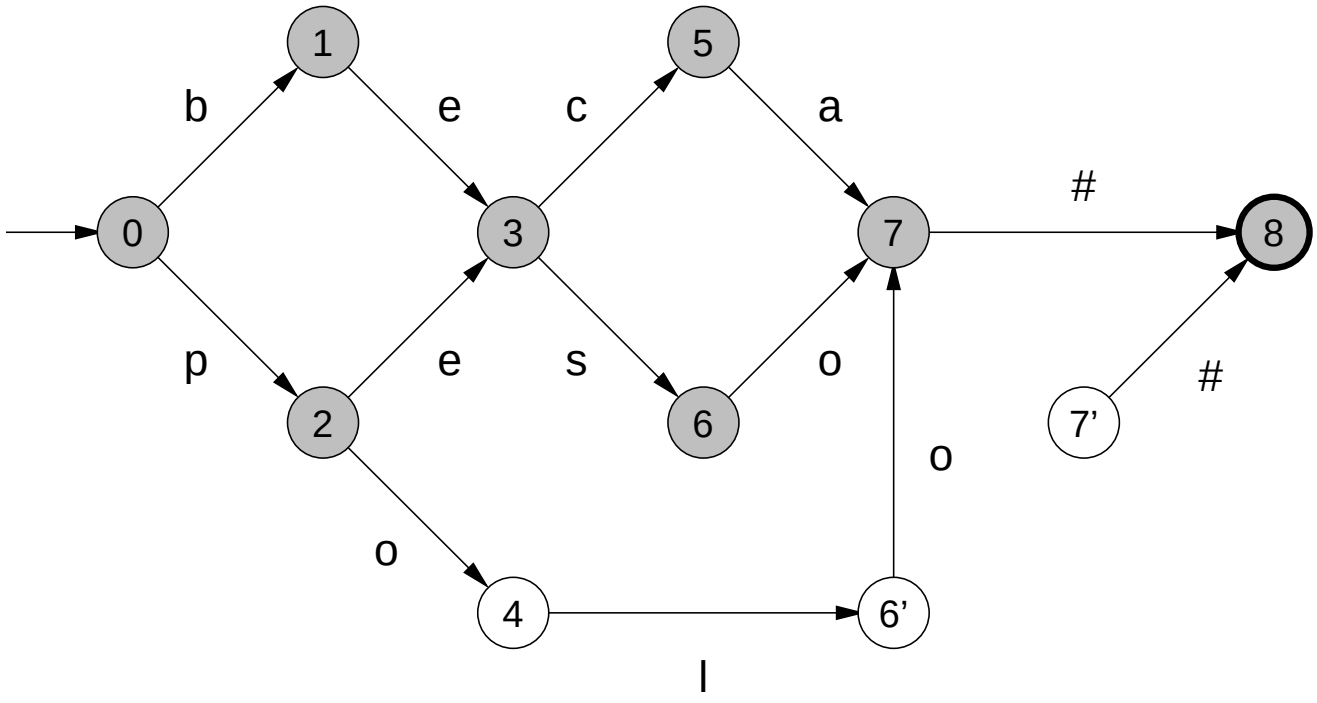
- Inserción de nuevas palabras [Daciuk *et al*. 2000].
- Eliminación de palabras ya insertadas [Barcala y Graña 2003].



Comportamiento Dinámico

Operaciones de mantenimiento sobre autómatas finitos acíclicos:

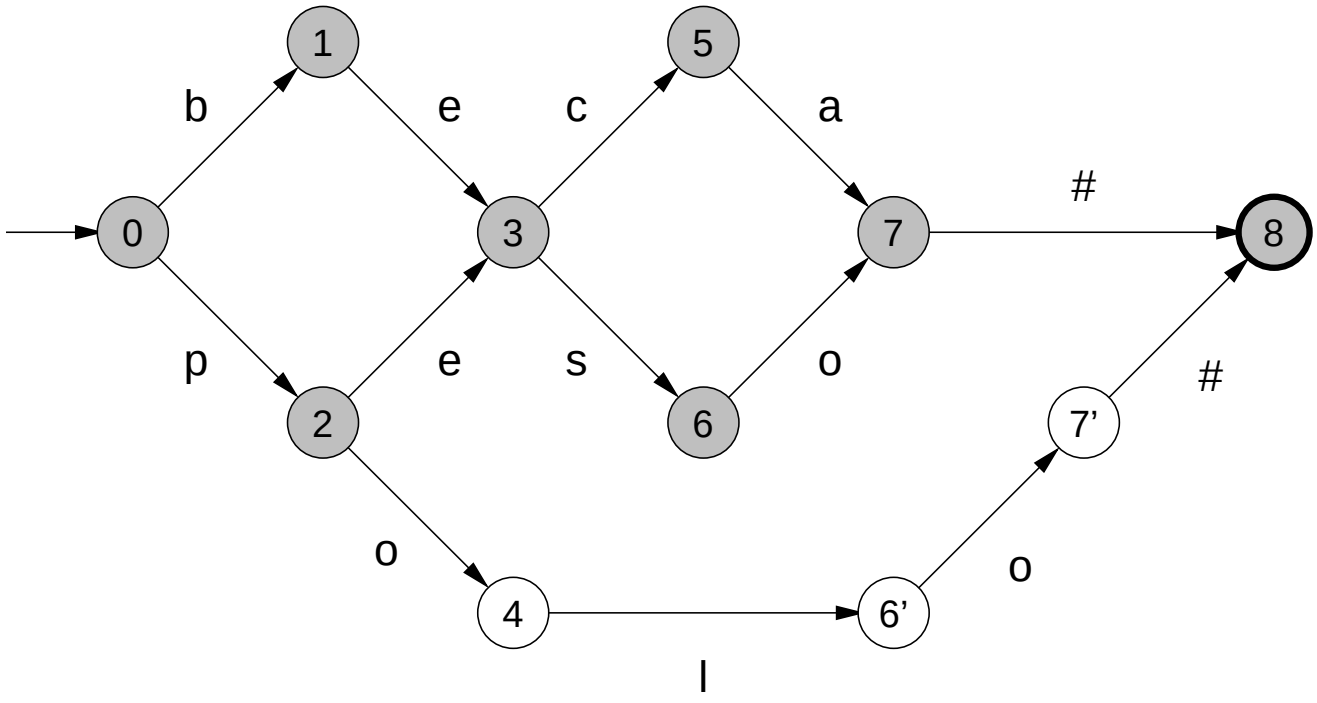
- Inserción de nuevas palabras [Daciuk *et al*. 2000].
- Eliminación de palabras ya insertadas [Barcala y Graña 2003].



Comportamiento Dinámico

Operaciones de mantenimiento sobre autómatas finitos acíclicos:

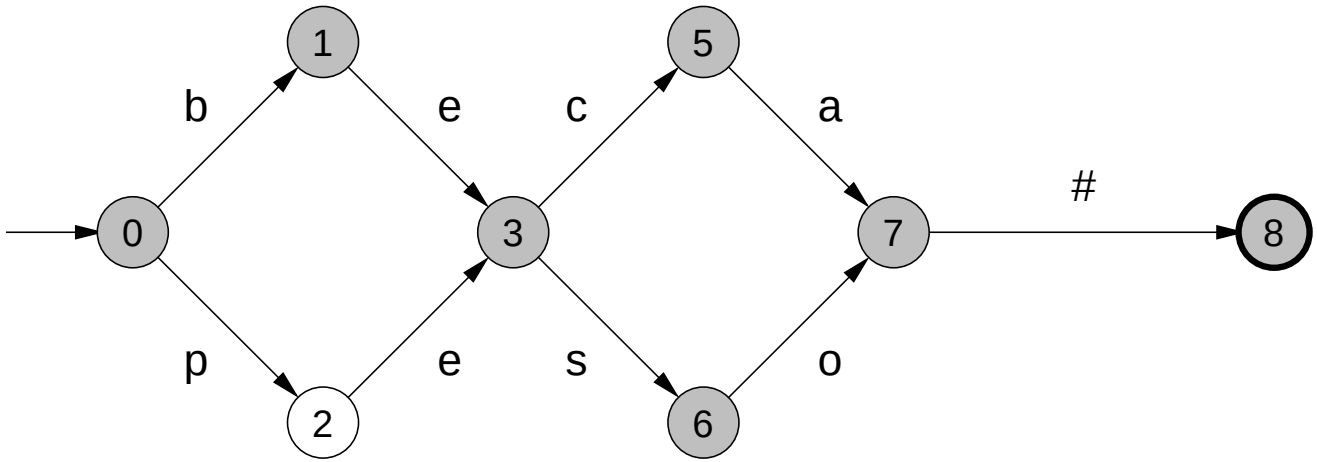
- Inserción de nuevas palabras [Daciuk *et al*. 2000].
- Eliminación de palabras ya insertadas [Barcala y Graña 2003].



Comportamiento Dinámico

Operaciones de mantenimiento sobre autómatas finitos acíclicos:

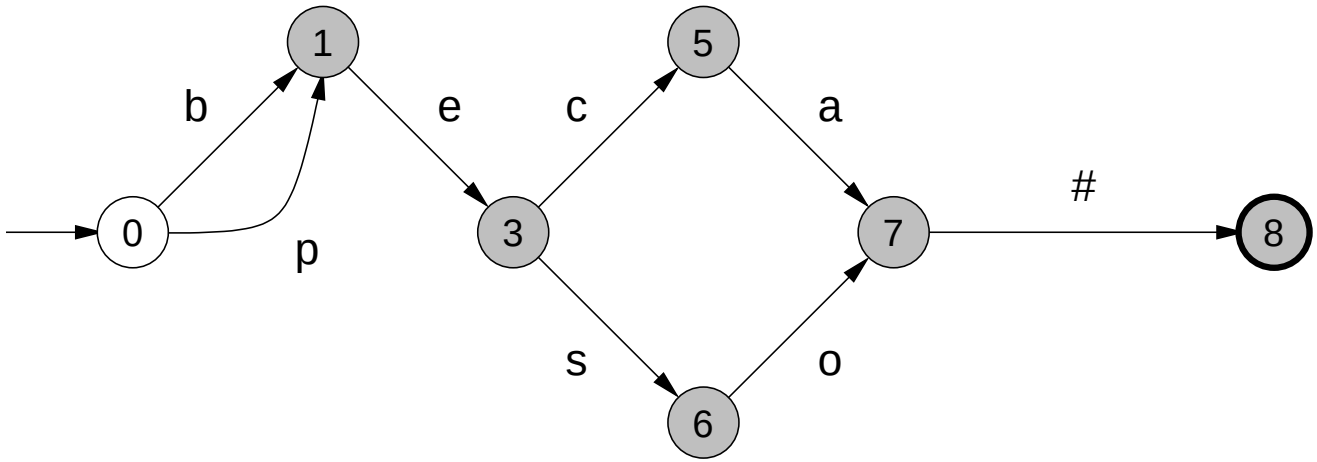
- Inserción de nuevas palabras [Daciuk *et al*. 2000].
- Eliminación de palabras ya insertadas [Barcala y Graña 2003].



Comportamiento Dinámico

Operaciones de mantenimiento sobre autómatas finitos acíclicos:

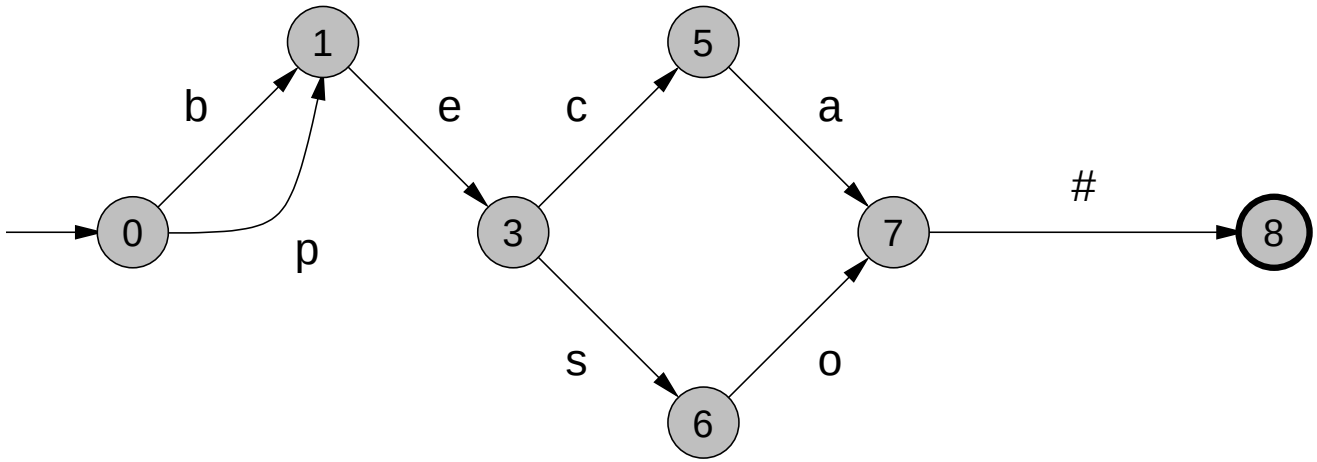
- Inserción de nuevas palabras [Daciuk *et al*. 2000].
- Eliminación de palabras ya insertadas [Barcala y Graña 2003].



Comportamiento Dinámico

Operaciones de mantenimiento sobre autómatas finitos acíclicos:

- Inserción de nuevas palabras [Daciuk *et al*. 2000].
- Eliminación de palabras ya insertadas [Barcala y Graña 2003].



Contenidos

- **Introducción:**
  - Lenguajes Naturales, Análisis Léxico y Autómatas Finitos Acíclicos.
- **Implementación de Grandes Diccionarios:**
  - Modelización Compacta de un Diccionario.
  - Autómatas Finitos Acíclicos Mínimos Numerados.
- **Construcción de Autómatas Acíclicos:**
  - Algoritmo General de Minimización.
  - Minimización Basada en la Propiedad de la Altura.
  - Algoritmo de Construcción Incremental.
- **Nuevas Funcionalidades:**
  - Mejoras en el Acceso al Registro.
  - Comportamiento Dinámico de los Autómatas Acíclicos.
- **Conclusiones.**

Reflexiones Finales

Las propuestas más relevantes de este estudio son:

- Una **arquitectura general para la implementación de diccionarios**, capaz de almacenar la gran cantidad de información léxica relativa a las palabras.

Reflexiones Finales

Las propuestas más relevantes de este estudio son:

- Una **arquitectura general para la implementación de diccionarios**, capaz de almacenar la gran cantidad de información léxica relativa a las palabras.
- Una **mejora en los métodos para la construcción incremental** de autómatas finitos acíclicos, mediante el uso de características específicas de los estados inspiradas:
  - en los mecanismos de trabajo de nuestra arquitectura de diccionarios (**pesos**),
  - en la base formal de otros algoritmos previos (**alturas**).

Reflexiones Finales

Las propuestas más relevantes de este estudio son:

- Una **arquitectura general para la implementación de diccionarios**, capaz de almacenar la gran cantidad de información léxica relativa a las palabras.
- Una **mejora en los métodos para la construcción incremental** de autómatas finitos acíclicos, mediante el uso de características específicas de los estados inspiradas:
  - en los mecanismos de trabajo de nuestra arquitectura de diccionarios (**pesos**),
  - en la base formal de otros algoritmos previos (**alturas**).
- Un conjunto de operaciones que facilita las tareas de mantenimiento de los autómatas, y que proporciona a éstos la posibilidad de **comportamiento dinámico**:
  - **inserción y eliminación de palabras** durante ejecuciones reales.

Reflexiones Finales

Las propuestas más relevantes de este estudio son:

- Una **arquitectura general para la implementación de diccionarios**, capaz de almacenar la gran cantidad de información léxica relativa a las palabras.
- Una **mejora en los métodos para la construcción incremental** de autómatas finitos acíclicos, mediante el uso de características específicas de los estados inspiradas:
  - en los mecanismos de trabajo de nuestra arquitectura de diccionarios (**pesos**),
  - en la base formal de otros algoritmos previos (**alturas**).
- Un conjunto de operaciones que facilita las tareas de mantenimiento de los autómatas, y que proporciona a éstos la posibilidad de **comportamiento dinámico**:
  - **inserción y eliminación de palabras** durante ejecuciones reales.

Aspectos a desarrollar en el futuro:

- Realizar un **estudio teórico de la complejidad temporal** de nuestras mejoras.

Reflexiones Finales

Las propuestas más relevantes de este estudio son:

- Una **arquitectura general para la implementación de diccionarios**, capaz de almacenar la gran cantidad de información léxica relativa a las palabras.
- Una **mejora en los métodos para la construcción incremental** de autómatas finitos acíclicos, mediante el uso de características específicas de los estados inspiradas:
  - en los mecanismos de trabajo de nuestra arquitectura de diccionarios (**pesos**),
  - en la base formal de otros algoritmos previos (**alturas**).
- Un conjunto de operaciones que facilita las tareas de mantenimiento de los autómatas, y que proporciona a éstos la posibilidad de **comportamiento dinámico**:
  - **inserción y eliminación de palabras** durante ejecuciones reales.

Aspectos a desarrollar en el futuro:

- Realizar un **estudio teórico de la complejidad temporal** de nuestras mejoras.
- Diseñar un **mecanismo de indexación** para sistemas de recuperación de información basado en autómatas finitos acíclicos.

Referencias Bibliográficas (I)

Métodos Generales para la Construcción de Autómatas Finitos:

- Watson, B.W. (1993).
  A Taxonomy of Finite Automata Construction Algorithms.
  Computing Science Note 93/43, Eindhoven Univ. of Technology, The Netherlands.

Referencias Bibliográficas (I)

Métodos Generales para la Construcción de Autómatas Finitos:

- Watson, B.W. (1993).
  A Taxonomy of Finite Automata Construction Algorithms.
  Computing Science Note 93/43, Eindhoven Univ. of Technology, The Netherlands.

***Hashing* Perfecto:**

- Lucchesi, C.L.; Kowaltowski, T. (1993).
  Applications of Finite Automata Representing Large Vocabularies.
  Software - Practice and Experience, vol. 23(1), pp. 15-30.

Referencias Bibliográficas (I)

Métodos Generales para la Construcción de Autómatas Finitos:

- Watson, B.W. (1993).
  A Taxonomy of Finite Automata Construction Algorithms.
  Computing Science Note 93/43, Eindhoven Univ. of Technology, The Netherlands.

***Hashing* Perfecto:**

- Lucchesi, C.L.; Kowaltowski, T. (1993).
  Applications of Finite Automata Representing Large Vocabularies.
  Software - Practice and Experience, vol. 23(1), pp. 15-30.

Algoritmos de Minimización:

- Hopcroft, J.E. (1971).
  An $n \log n$ Algorithm for Minimizing the States in a Finite Automaton.
  The Theory of Machines and Computations, Academic Press, NY, pp. 189-196.
- Revuz, D. (1992).
  Minimization of Acyclic Deterministic Automata in Linear Time.
  Theoretical Computer Science, vol. 92(1), pp. 181-189.

Referencias Bibliográficas (II)

Algoritmos de Construcción Incremental:

- Daciuk, J.; Mihov, S.; Watson, B.W.; Watson, R.E. (2000). Incremental Construction of Minimal Acyclic Finite-State Automata. *Computational Linguistics*, vol. 26(1), pp. 3-16.
- Graña Gil, J.; Barcala Rodríguez, F.M.; Alonso Pardo, M.A. (2001). Compilation Methods of Minimal Acyclic Finite-State Automata for Large Dictionaries. In *Proc. of CIAA-2001*, pp. 135-148. Pretoria, South Africa.

Referencias Bibliográficas (II)

Algoritmos de Construcción Incremental:

- Daciuk, J.; Mihov, S.; Watson, B.W.; Watson, R.E. (2000).
  Incremental Construction of Minimal Acyclic Finite-State Automata.
  Computational Linguistics, vol. 26(1), pp. 3-16.
- Graña Gil, J.; Barcala Rodríguez, F.M.; Alonso Pardo, M.A. (2001).
  Compilation Methods of Minimal Acyclic Finite-State Automata for Large Dictionaries.
  In *Proc. of CIAA-2001*, pp. 135-148. Pretoria, South Africa.

Comportamiento Dinámico de los Autómatas Finitos:

- Revuz, D. (2000).
  Dynamic Acyclic Minimal Automaton.
  In *Proc. of CIAA-2000*, pp. 226-232. London, Ontario, Canada.
- Carrasco, R.C.; Forcada, M.L. (2002).
  Incremental Construction and Maintenance of Minimal Finite-State Automata.
  Computational Linguistics, vol. 28(2), pp. 207-216.

Referencias Bibliográficas (y III)

Corrección Ortográfica:

- Oflazer, K. (1996).
  Error-Tolerant Finite State Recognition with Applications to Morphological Analysis and Spelling Correction.
  Computational Linguistics, vol. 22(1), pp. 73-89.

Referencias Bibliográficas (y III)

Corrección Ortográfica:

- Oflazer, K. (1996).
  Error-Tolerant Finite State Recognition with Applications to Morphological Analysis and Spelling Correction.
  Computational Linguistics, vol. 22(1), pp. 73-89.

Otros Métodos de Análisis Léxico (Traductores de Estado Finito):

- Koskenniemi, K. (1983).
  Two-Level Model for Morphological Analysis.
  In *Proc. of IJCAI-83*, pp. 683-685. Kalsruhe, Germany.
- Kempe, A.; Karttunen, L. (1996).
  Parallel Replacement in the Finite-State Calculus.
  In *Proc. of COLING-96*. Copenhagen, Denmark.
- Mohri, M. (1997).
  Finite-State Transducers in Language and Speech Recognition.
  Computational Linguistics, vol. 23(2), pp. 269-311.