Autómatas y Lenguajes

3er curso

1er cuatrimestre

Alfonso Ortega: alfonso.ortega@uam.es

UNIDAD 1: Modelos de cómputo y familias de lenguajes

TEMA 3: Autómatas finitos deterministas y no deterministas, expresiones regulares

Autómatas finitos

Autómatas finitos deterministas

Transiciones entre estados ante símbolos de entrada

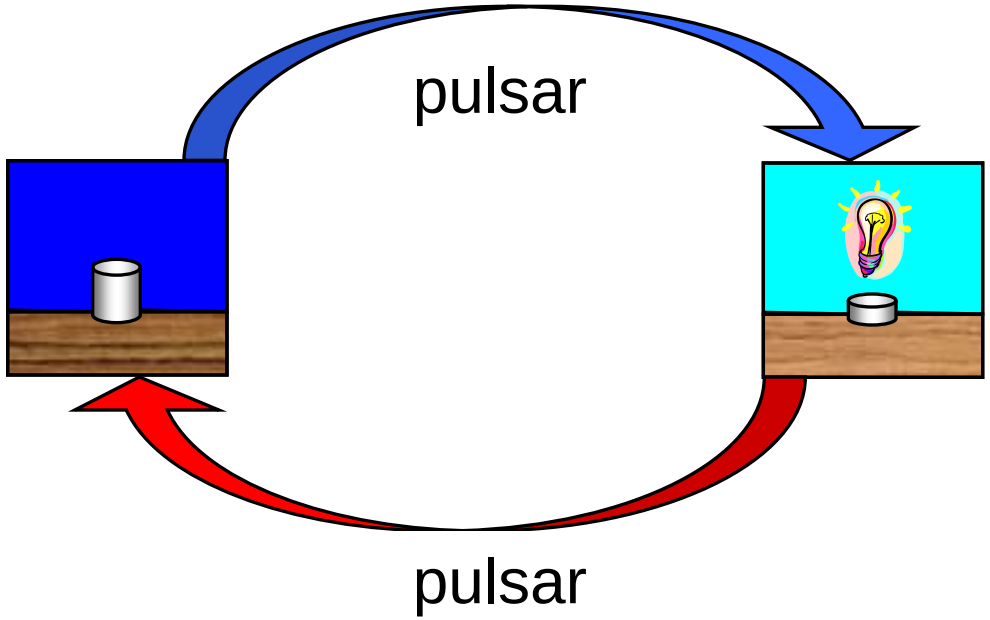
- Anteriormente se ha analizado informalmente el comportamiento de un ejemplo de autómata: un **expendedor de productos**.
- Un elemento básico de las clases de autómatas objeto del curso es la capacidad que tienen de adoptar diferentes estados y transitar entre ellos.
- Este cambio de estado a menudo se asocia a la presencia en la entrada del autómata de un símbolo.
- De esta manera el autómata reacciona a los símbolos que tiene en la entrada.
- Podemos imaginar el siguiente comportamiento de los autómatas para procesar una cadena de entrada:
  - Inicialmente el autómata se encontrará en algún estado inicial ante el primer símbolo de la palabra.
  - En cualquiera de los pasos intermedios de evolución del autómata, éste pasará del estado en el que está al que le corresponda según el símbolo que actualmente tiene en su entrada. Conoceremos este hecho como **consumir** el símbolo. Tras consumir un símbolo, el nuevo símbolo actual del autómata es el siguiente en la cadena.
  - En algún momento se llegará a consumir el último símbolo de la cadena, lo que representará el final del proceso de la cadena.
- Éste es un componente básico de los tipos de autómatas que se estudiarán este curso.
- Cada tipo se completará con otras acciones especiales que se pueden añadir al proceso anteriormente descrito.

Transiciones entre estados ante símbolos de entrada

- Comenzaremos analizando con detalle las transiciones entre estados.
  - A continuación se analizará exhaustivamente un caso sencillo (**un interruptor**) Resulta claro que el interruptor puede encontrarse en dos situaciones distintas: **apagado** o **encendido**.
  - El interruptor sólo puede recibir un estímulo exterior: **que se pulse su botón**.
  - Cuando el interruptor esté apagado, si se pulsa el botón pasa a estar encendido, en otro caso sigue apagado.
  - Cuando el interruptor esté encendido, si se pulsa el botón pasa a estar apagado, en otro caso sigue encendido.
  - Es frecuente que inicialmente el interruptor esté apagado.
  - En este caso la salida del sistema puede considerarse que es el funcionamiento del dispositivo que controle (una bombilla, por ejemplo).

Transiciones entre estados ante símbolos de entrada

- La siguiente figura muestra gráficamente la situación.



Transiciones entre estados ante símbolos de entrada

- El análisis del comportamiento de las transiciones del interruptor sugiere dividir su análisis formal en dos partes:
  - **Entradas** (externas),
    - Que representan los estímulos del exterior a los que se puede reaccionar
  - **Estados** (internos) y **transiciones**,
    - Los **estados** representan las diferentes situaciones en las que se puede encontrar. En el caso del **interruptor**: encendido y apagado
    - Las **Transiciones** representan el cambio de un estado a otro debido a las entradas.

Transiciones entre estados ante símbolos de entrada

- Los **estados** de los autómatas
  - Tienen como objetivo **recordar la historia** del sistema.
  - La **cantidad** de estados posibles es **finita** y se recuerda **sólo lo relevante**.
  - El hecho de que sea **finita** permitirá que se escriban algoritmos para **simular** los autómatas con una **cantidad de recursos acotada**, es decir, los algoritmos no necesitarán una cantidad creciente de recursos (por ejemplo memoria) que puedan hacer que el programa falle por agotamiento de los mismos.
- El **estado inicial**
  - Recoge la **situación** en la que se encuentran los autómatas **inicialmente**.
- Los **estados finales**
  - Representan las situaciones en las que, de llegar los autómatas a ellas, se considera que su funcionamiento **ha finalizado satisfactoriamente**.

Transiciones entre estados ante símbolos de entrada

- Las **entradas**
  - Representan los **estímulos** que pueden llegar desde el **exterior** y afectar el comportamiento de los autómatas.
- Las **transiciones** entre estados de los autómatas asociadas a una entrada
  - Representan la manera en la que los **estímulos externos modifican** el comportamiento de los autómatas.

Formalización de la función de transición del ejemplo 1

- Es fácil entender que, formalmente, puede definirse los conjuntos de estados, símbolos de entrada y la función de transición del interruptor de la siguiente manera:
- Supongamos
  - El **conjunto de estados** representa **encendido** con 1 y **apagado** con 0.

$$Q_1=\{0, 1\}$$

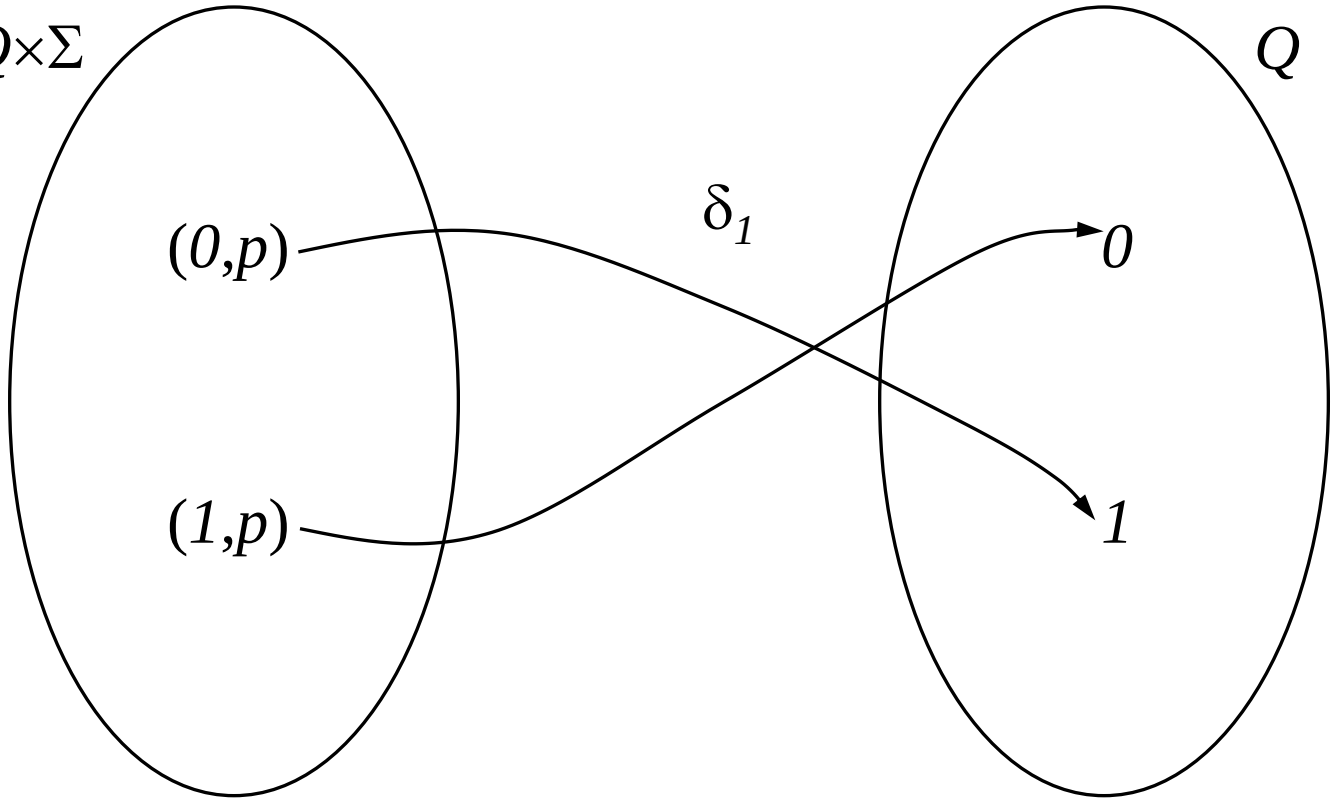
  - El conjunto de símbolos de entrada al que llamaremos **alfabeto de entrada** representa **pulsar** con p.

$$\Sigma_1=\{p\}$$

  - Entre los estados, el **estado inicial** q_0 es que esté **apagado** (0).
  - En este caso no se indican **estados finales**.
  - La **función de transición** se suele llamar δ y, a continuación, se muestra mediante la notación habitual en teoría de conjuntos.

Formalización de la función de transición del ejemplo 1

- A continuación se muestra la formalización del ejemplo del interruptor:



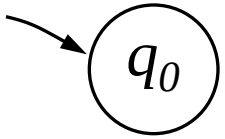
Otras notaciones formales

- La función de transición suele representarse utilizando otras notaciones formales, en algún sentido, más cómodas.
- Las más usadas son las siguientes:
  - **Diagrama de transiciones**:
    - Consiste en disponer la información en forma de **grafo**
  - **Tabla de transiciones**:
    - Consiste en disponerla en forma de tabla.

Diagrama de transición

- El diagrama de transición para una función de transición δ es un grafo dirigido que se define de la siguiente manera:
  - **Nodos**:
    - Hay uno para **cada estado** ($q \in Q$) y se representan mediante un óvalo que contiene el nombre del estado:

    - Al que contiene el **estado inicial** (q_0), se le marca de una manera especial, habitualmente mediante un arco sin origen y sin etiqueta que termina en él:



    - La línea del óvalo de los **estados finales** ($q \in F$) es doble:

Diagrama de transición

- **Arcos**:
  - Se etiquetan con los símbolos del alfabeto de entrada ($a \in \Sigma$).
  - Existe un arco del nodo p al q con la etiqueta a si y sólo si $\delta(p,a)=q$.
  - Se representan mediante una flecha que sale del nodo p y llega al q.

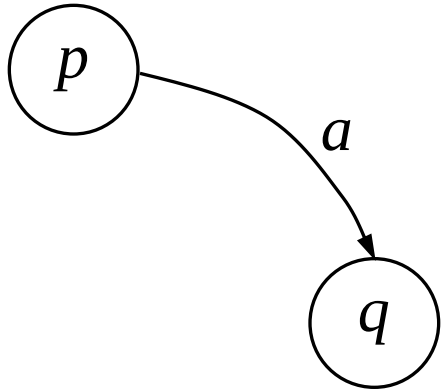


Diagrama de transición: ejemplo del interruptor

- A continuación se muestra el diagrama de la función de transición del interruptor.

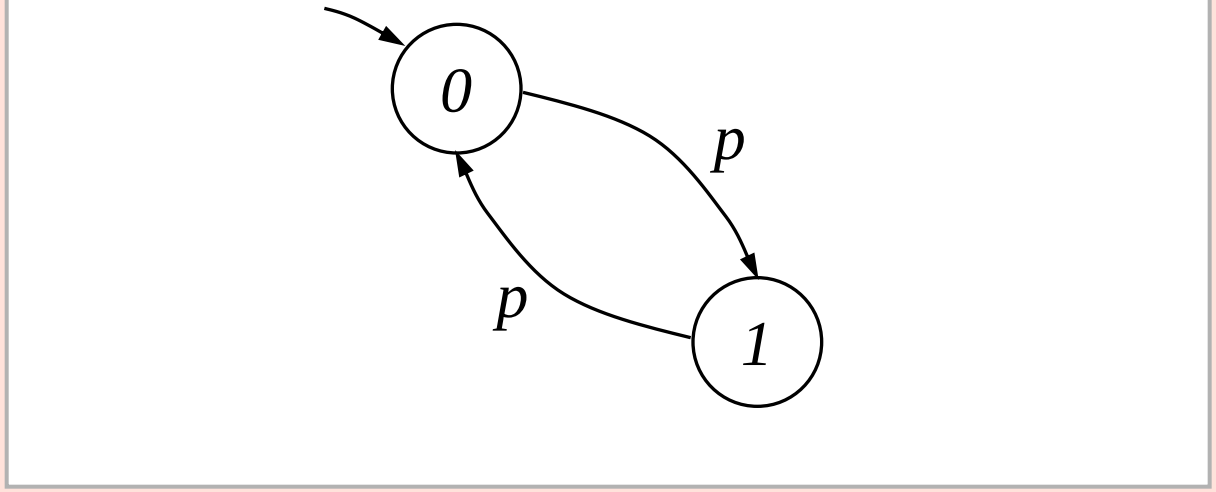


Tabla de transición

- La tabla de transición para una función de transición δ se define de la siguiente manera:
  - **Filas**:
    - Tantas como estados haya ($|Q|$)
    - Es necesario
      - Marcar la del **estado inicial** (por ejemplo mediante el símbolo $\rightarrow$)
      - Marcar la de los **estados finales** (por ejemplo mediante el símbolo *)
  - **Columnas**:
    - Tantas columnas como entradas haya ($|\Sigma|$)
  - **Contenido de las celdas**:
    - La casilla de la fila q y columna a contiene $\delta(q,a)$

Diagrama de transición: ejemplo del interruptor

- A continuación se muestra la tabla de la función de transición del ejemplo del interruptor.

	p
$\rightarrow 0$	1
1	0

Reconocimiento de palabras

- Una aplicación muy importante de los autómatas que serán estudiados a lo largo de este curso es su **capacidad de reconocer palabras**.
- El autómata reconocerá una palabra si, **tras procesar la secuencia de sus símbolos** en el mismo orden en el que aparece, **termina en un estado final**.
- En este sentido el conjunto de estados finales, que indica que el funcionamiento del autómata ha terminado de forma exitosa, realmente indica que la palabra tratada por él ha sido reconocida.

Introducción: transiciones entre estados

Ej: función de transición con estados finales para el reconocimiento de una palabra

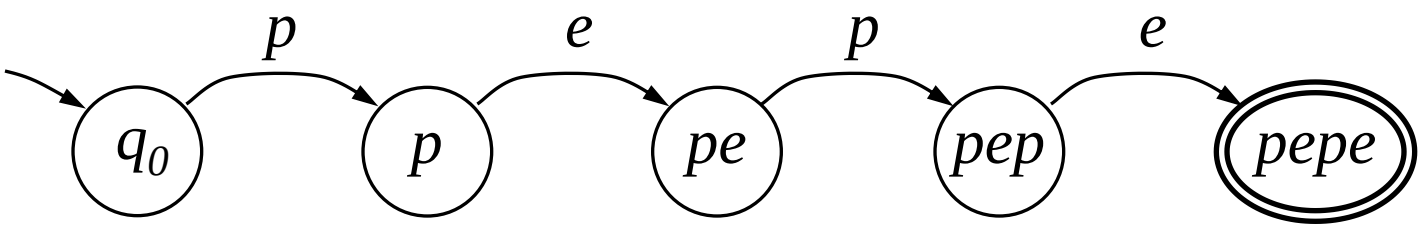
- Se desea diseñar un conjunto de estados para definir una función de transición que reconozca la palabra $pepe$:
- El alfabeto de entrada contendrá el siguiente conjunto (podría extenderse a más letras)

$$\{p, e\}$$

- Se puede analizar su funcionamiento determinando la parte relevante de su historia que tiene que corresponder a estados:
  - Se necesita un estado inicial (q_0)
  - El único cambio relevante en el estado inicial es que se reciba una letra p, en este caso debe transitarse a un estado que lleve cuenta de que de la palabra pepe sólo se ha recibido la primera letra (q_p o simplemente p)
  - En este estado el único cambio relevante es la presencia en la entrada de la letra e; con ella debe transitarse a un estado que lleve cuenta de que se han recibido las dos primeras letras de la palabra (pe)
  - Este mismo razonamiento se aplica para añadir los estados correspondientes a pep y a $pepe$.
  - Cuando se llega a este último estado, puede afirmarse que la función de transición ha reconocido la palabra completa por lo que será final.

Ej: función de transición con estados finales para el reconocimiento de una palabra

- A continuación se muestra parte del diagrama de la función de transición del ejemplo.



- En general, **siempre que tengamos que aceptar secuencias concretas de símbolos**, las funciones de transición de los correspondientes autómatas tendrán una secuencia de estados similar a ésta.

Introducción: transiciones entre estados

Ej: función de transición con estados finales para el reconocimiento de una palabra

- A continuación se muestra la tabla correspondiente a la situación estudiada.

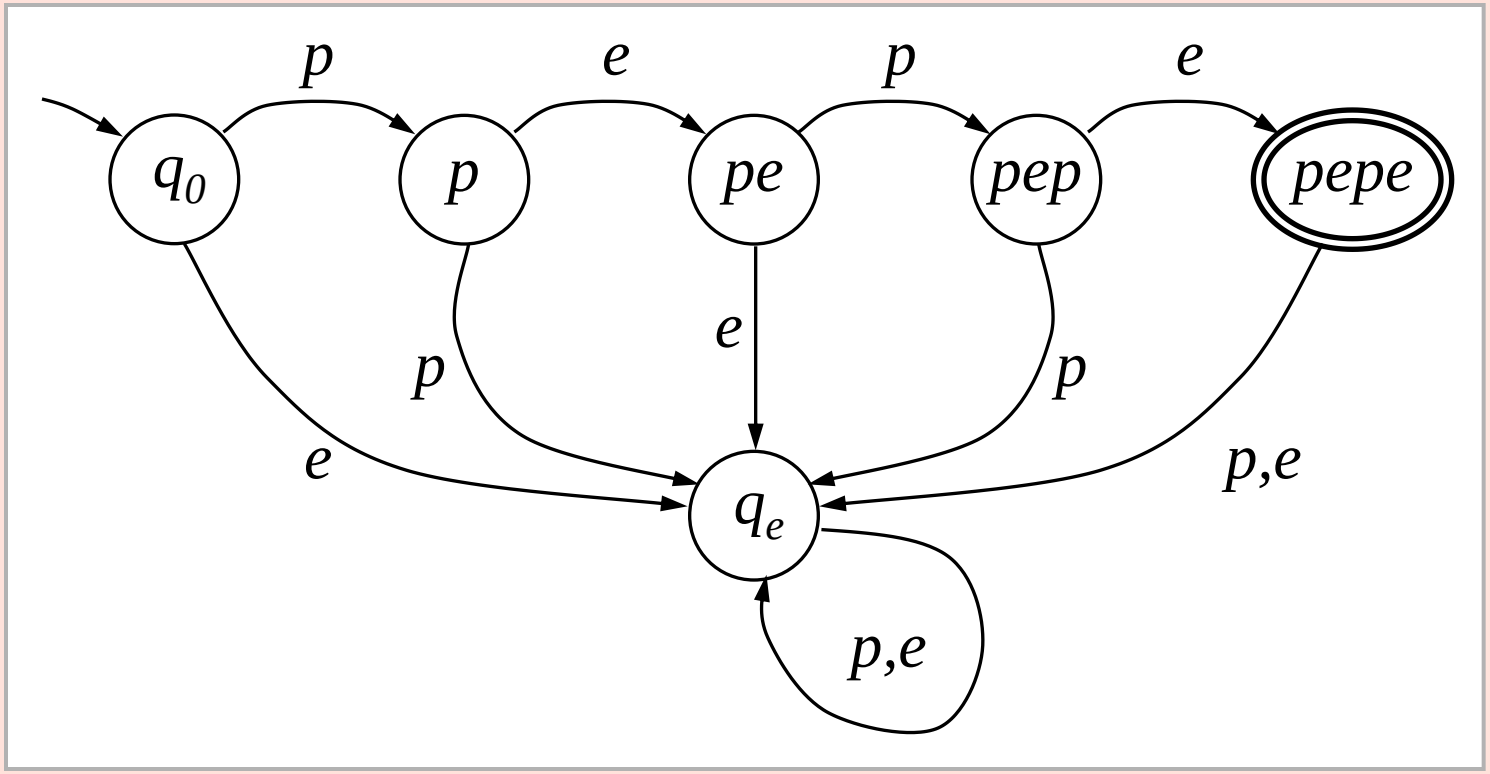
	p	e
$\rightarrow q_0$	p	
p		pe
pe	pep	
pep		$pepe$
$*pepe$		

Ej: función de transición con estados finales para el reconocimiento de una palabra

- Esto no es suficiente para definir por completo la función de transición. La función de transición debe especificar estados para todas las combinaciones posibles de estados y símbolos de entrada. La tabla mostraba 6 casillas vacías:
  - Cuando se encuentra en el estado inicial, si recibe la letra e, puede afirmar que no va a poder reconocer la palabra $pepe$ porque empieza por una letra (p) distinta a la recibida.
  - Lo mismo pasa cuando se está en el estado p y se recibe otra p, no se podrá terminar con éxito ya que $pepe$ no comienza por pp.
  - Y la situación es la misma para las casillas (pe,e) y (pep,p).
  - Desde el estado final, ya que sólo interesa reconocer la palabra $pepe$, y no las palabras que comiencen con $pepe$, cualquier otro símbolo que la siga debe implicar que no se termina con éxito.
- Se puede añadir un estado **de error** (q_e) que recoja los descritos anteriormente:
  - Este estado recoge las transiciones no exitosas de los demás.
  - También es necesario especificar qué pasa cuando desde este estado, se reciben símbolos de entrada: resulta claro que, producida una situación de error, no hay ningún símbolo que pueda hacer que el autómata termine con éxito.

Ej: función de transición con estados finales para el reconocimiento de una palabra

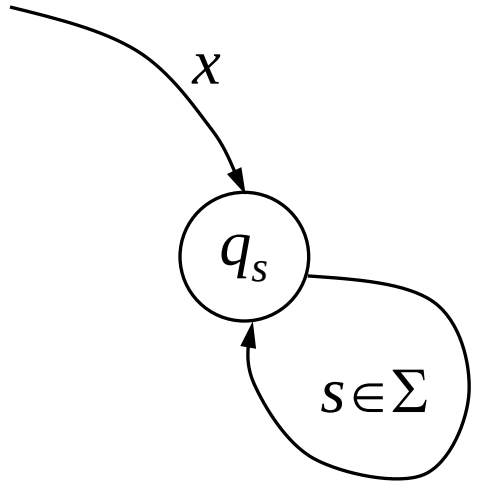
- A continuación se muestra el diagrama de la función de transición del ejemplo.



- En general, será muy frecuente que las funciones de transición tengan este tipo de estados para gestionar los errores.

Indicaciones prácticas: estados sumidero

- Se puede calificar estos estados como de **sumidero**: aquellos estados desde los que es imposible salir ya que, con cualquier posible entrada, los autómatas deben permanece en ellos.
- Es general la técnica de utilizarlos para recoger las situaciones de error.
- Si la entrada x hace que la función de transición no pueda terminar con éxito, debe incluir las siguientes (q_s es un estado nuevo):



Ej: función de transición con estados finales para el reconocimiento de una palabra

- A continuación se muestra la tabla correspondiente a la situación estudiada.

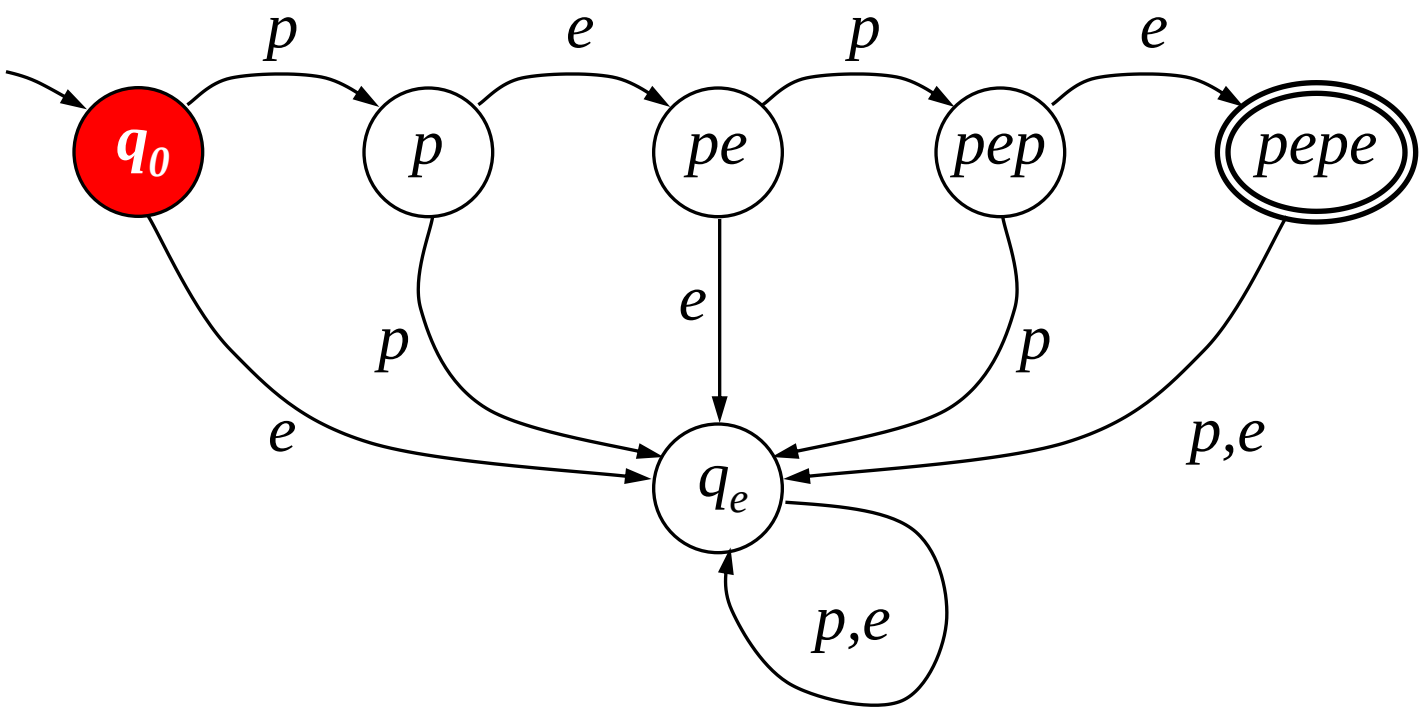
	p	e
$\rightarrow q_0$	p	q_e
p	q_e	pe
pe	pep	q_e
pep	q_e	$pepe$
$*pepe$	q_e	q_e
q_e	q_e	q_e

Ej: función de transición con estados finales para el reconocimiento de una palabra

- Puede comprobarse ahora cómo se comporta un autómata cuando recibe como entrada la secuencia de símbolos *pepe*.

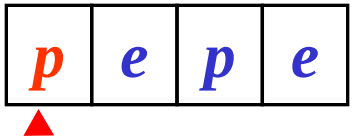
p	***e***	***p***	***e***

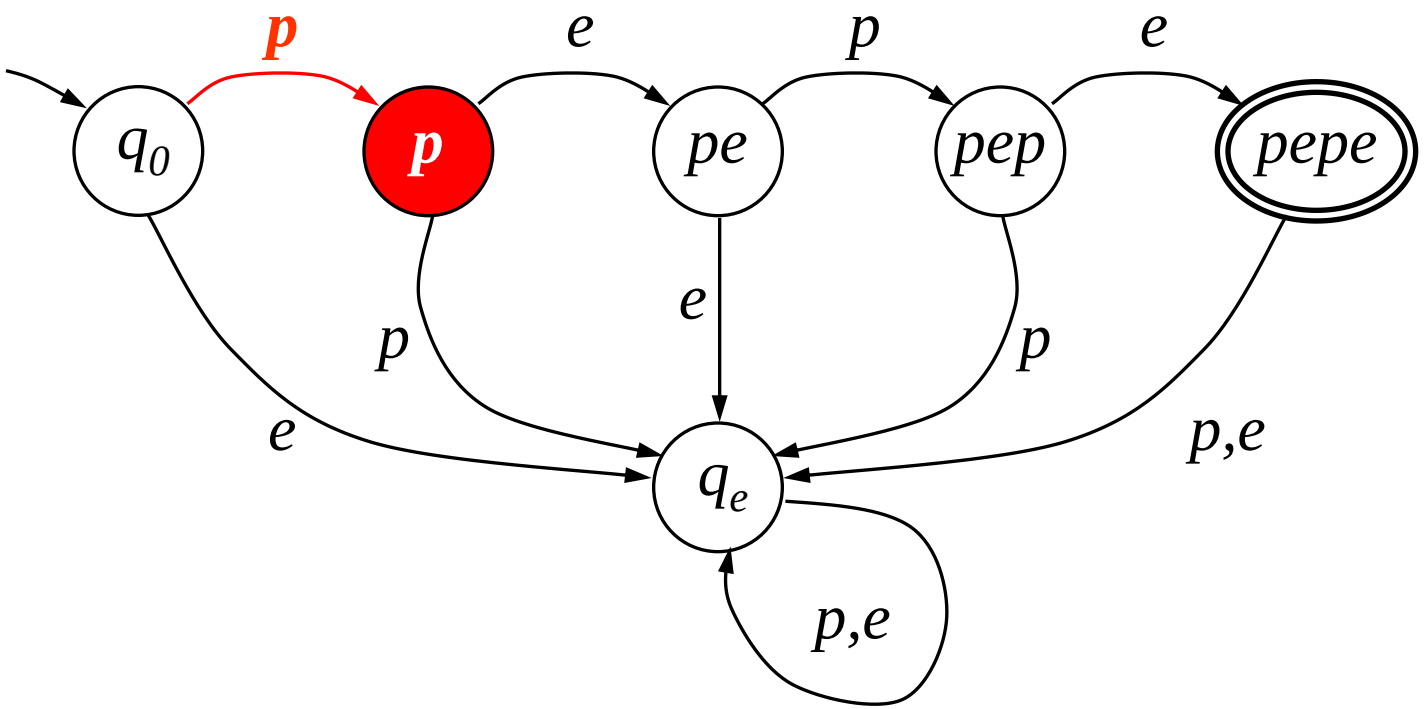
▲



Ej: función de transición con estados finales para el reconocimiento de una palabra

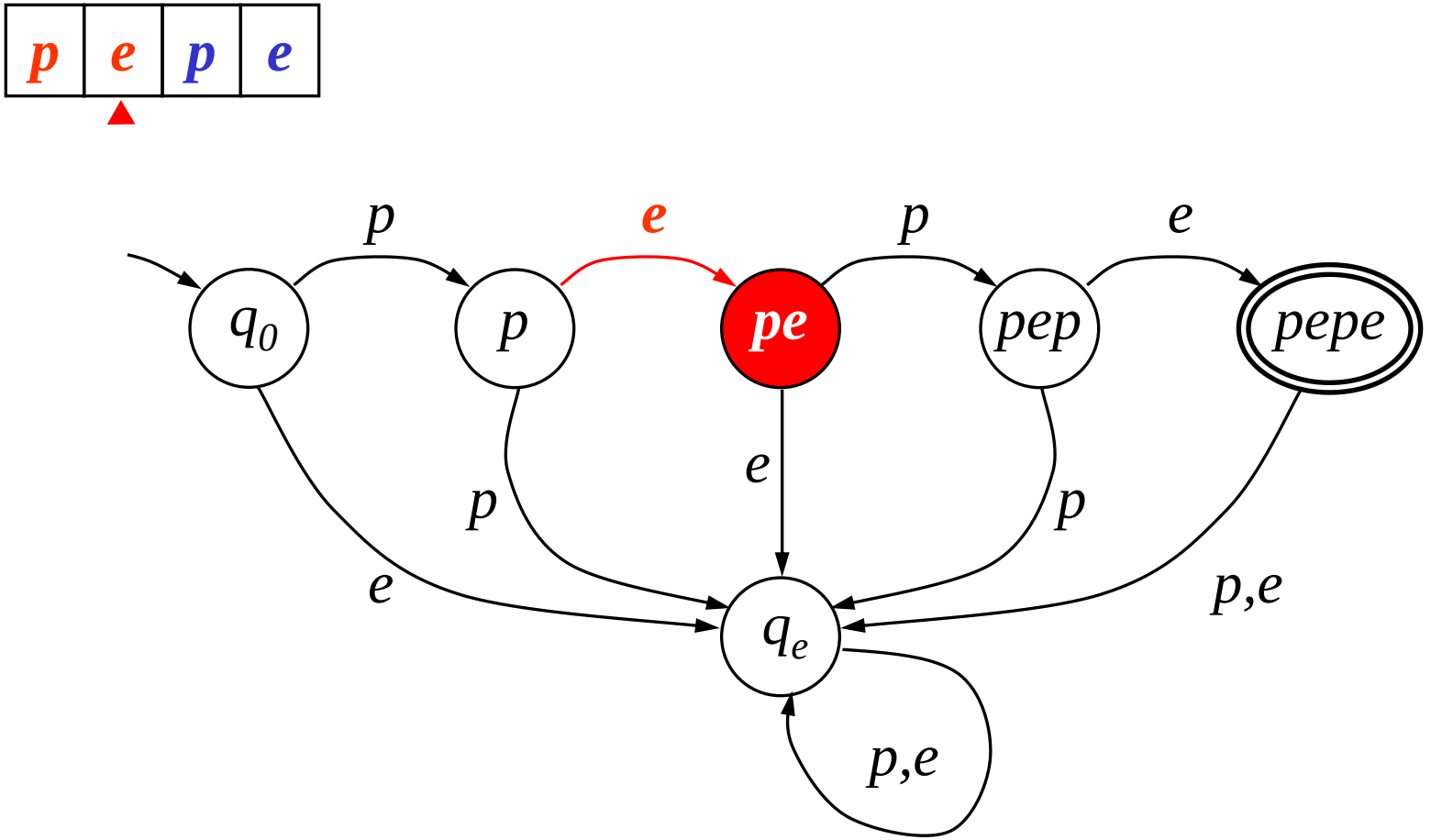
- Puede comprobarse ahora el procesamiento de la palabra de entrada *pepe*.





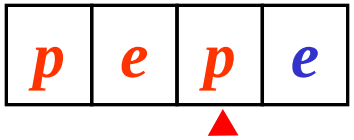
Ej: función de transición con estados finales para el reconocimiento de una palabra

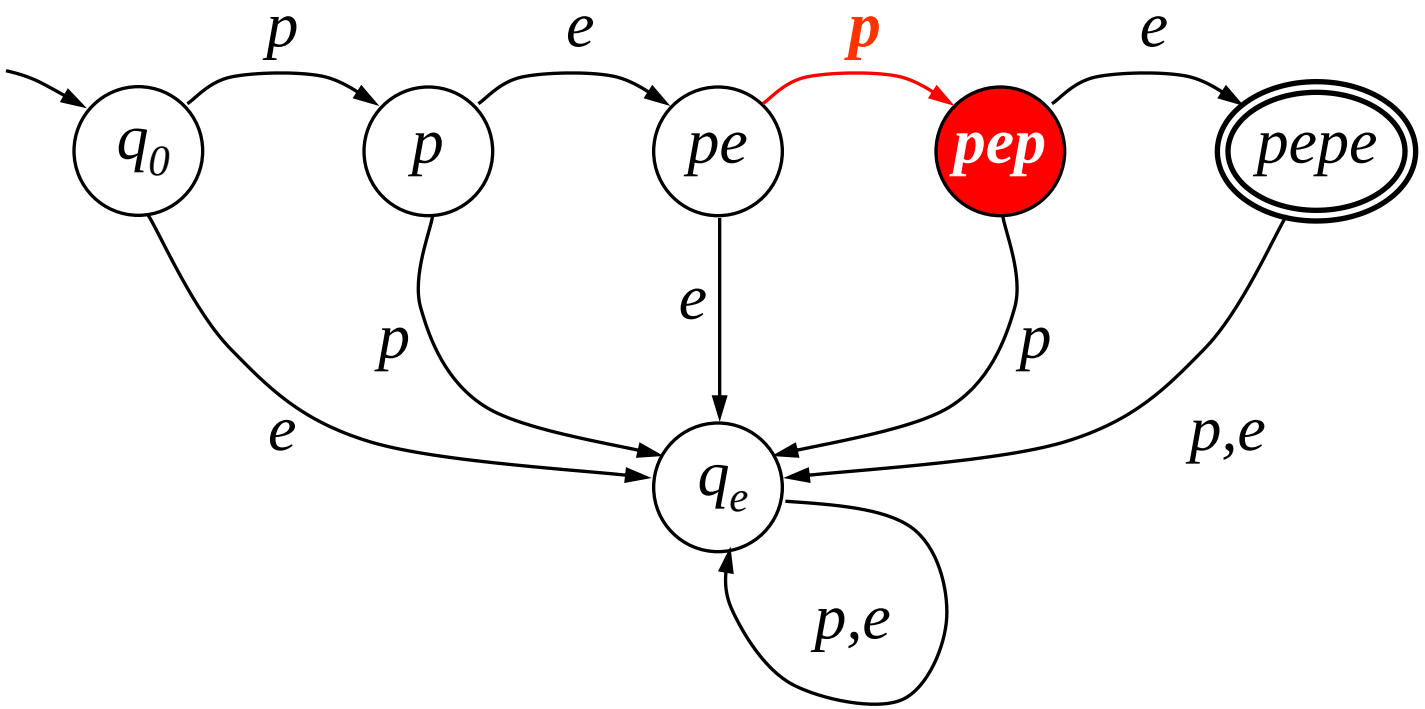
- Puede comprobarse ahora el procesamiento de la palabra de entrada *pepe*.



Ej: función de transición con estados finales para el reconocimiento de una palabra

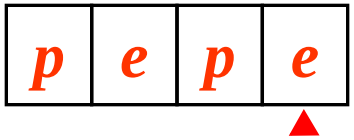
- Puede comprobarse ahora el procesamiento de la palabra de entrada *pepe*.

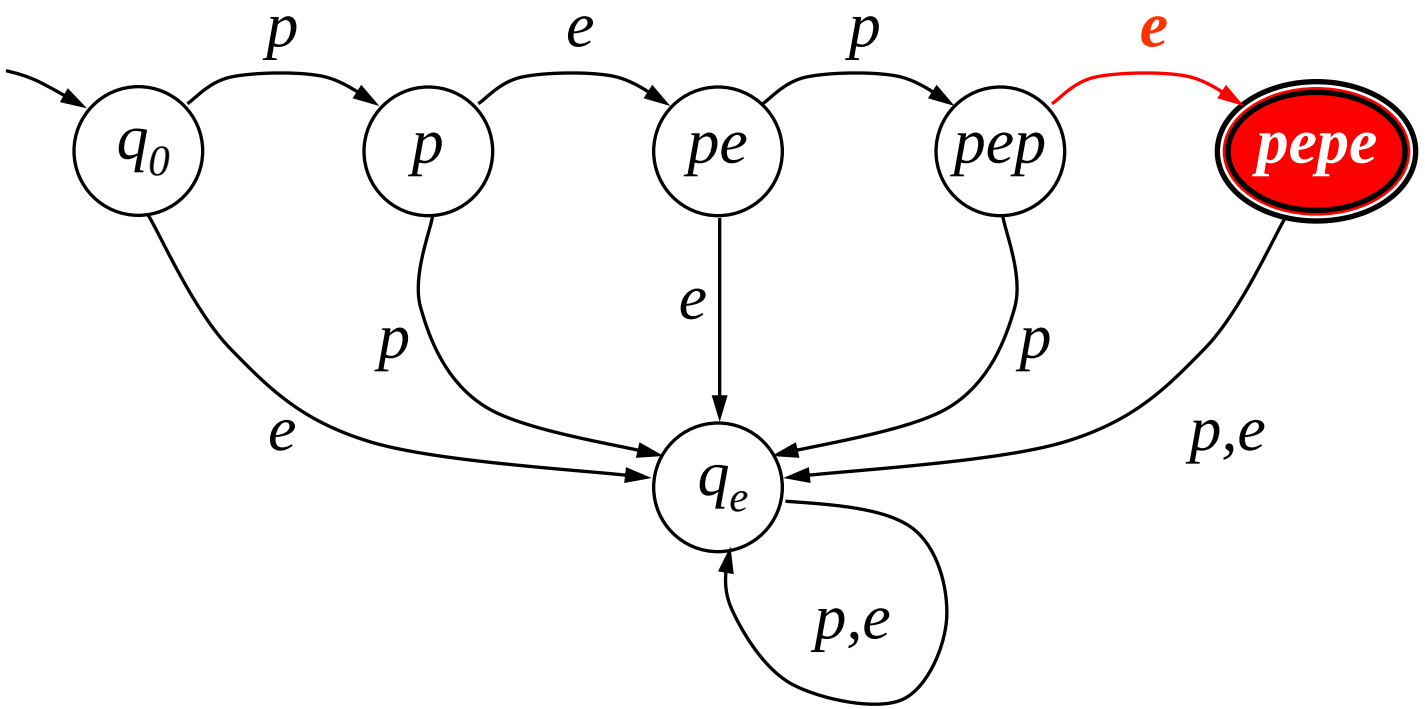




Ej: función de transición con estados finales para el reconocimiento de una palabra

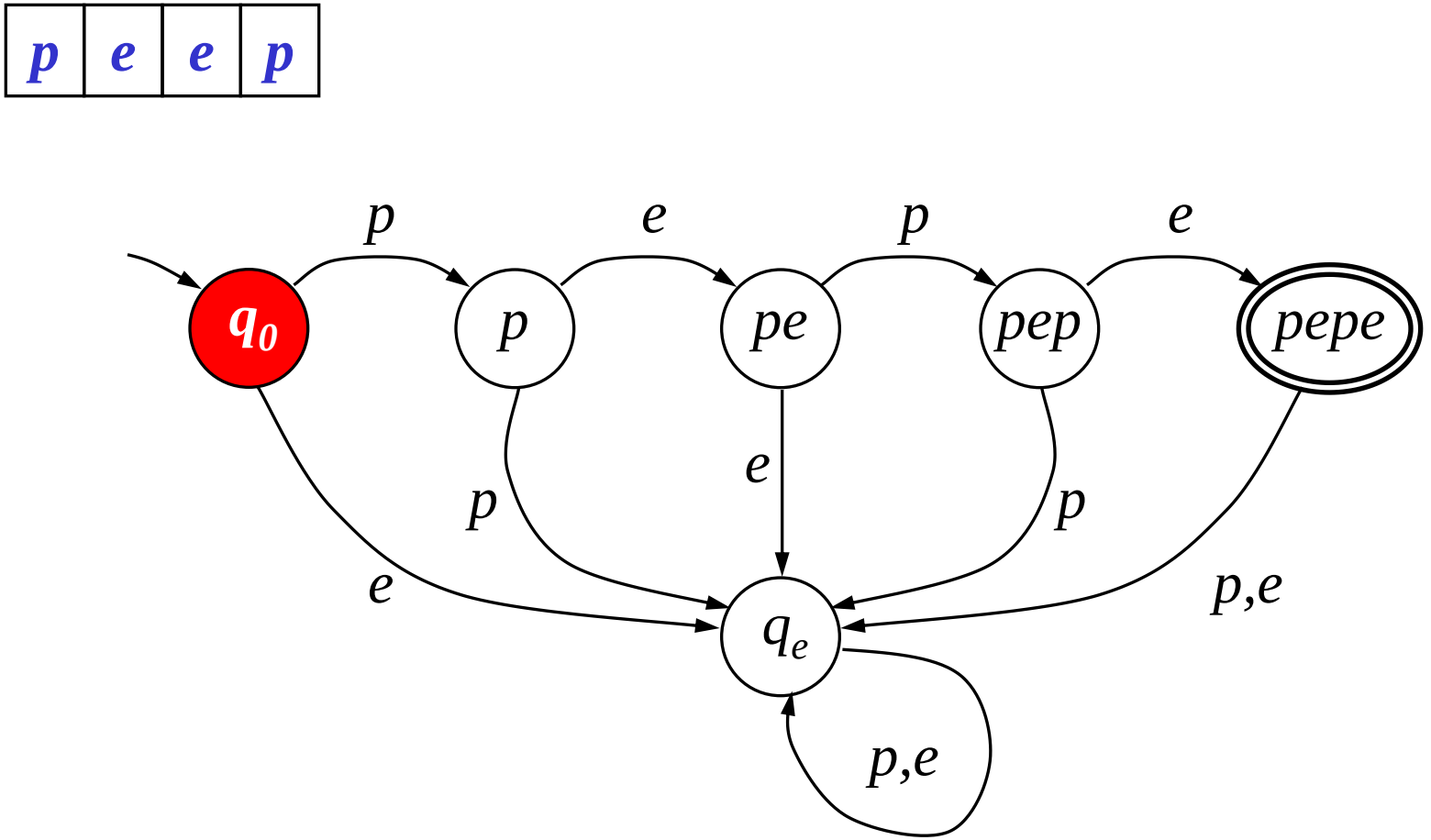
- Puede comprobarse ahora el procesamiento de la palabra de entrada *pepe*. El autómata termina en un estado final: la cadena se acepta.





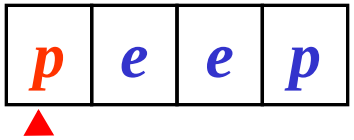
Ej: función de transición con estados finales para el reconocimiento de una palabra

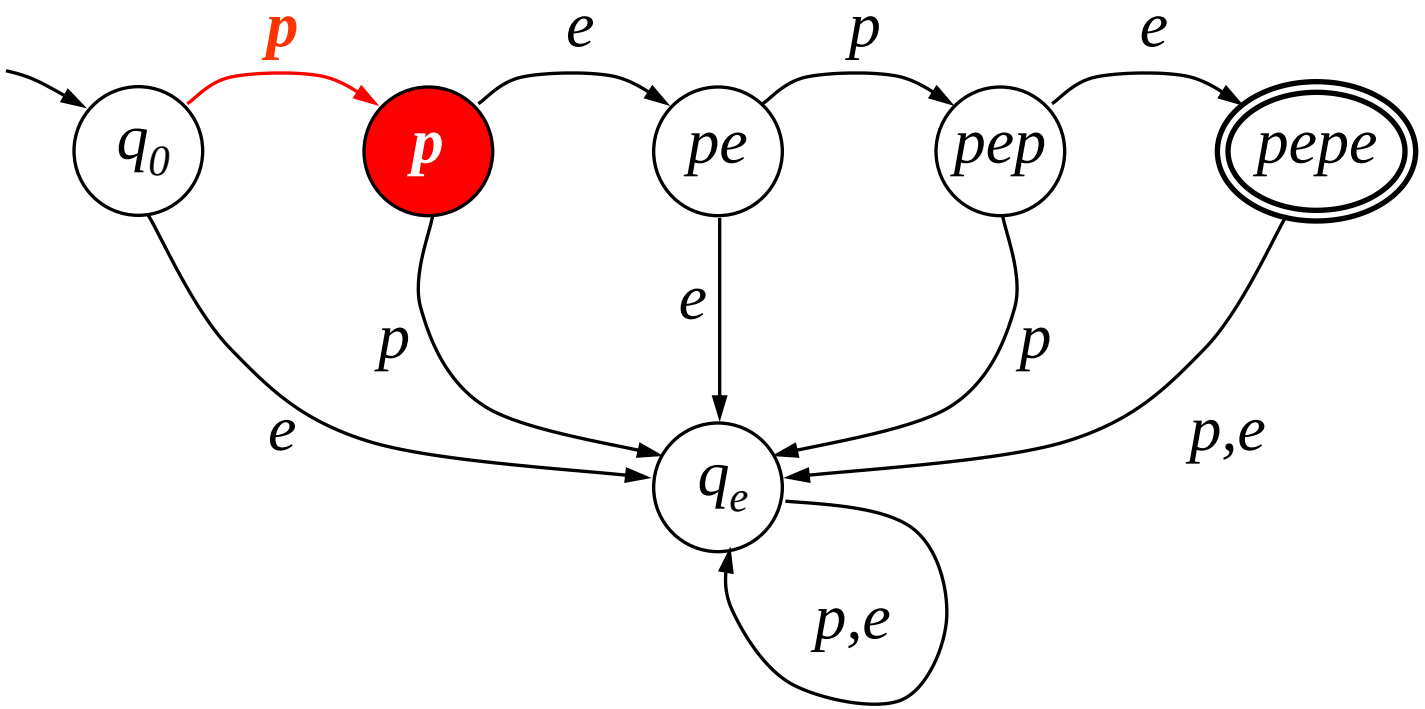
- Puede comprobarse ahora el procesamiento de la palabra de entrada *peep*.



Ej: función de transición con estados finales para el reconocimiento de una palabra

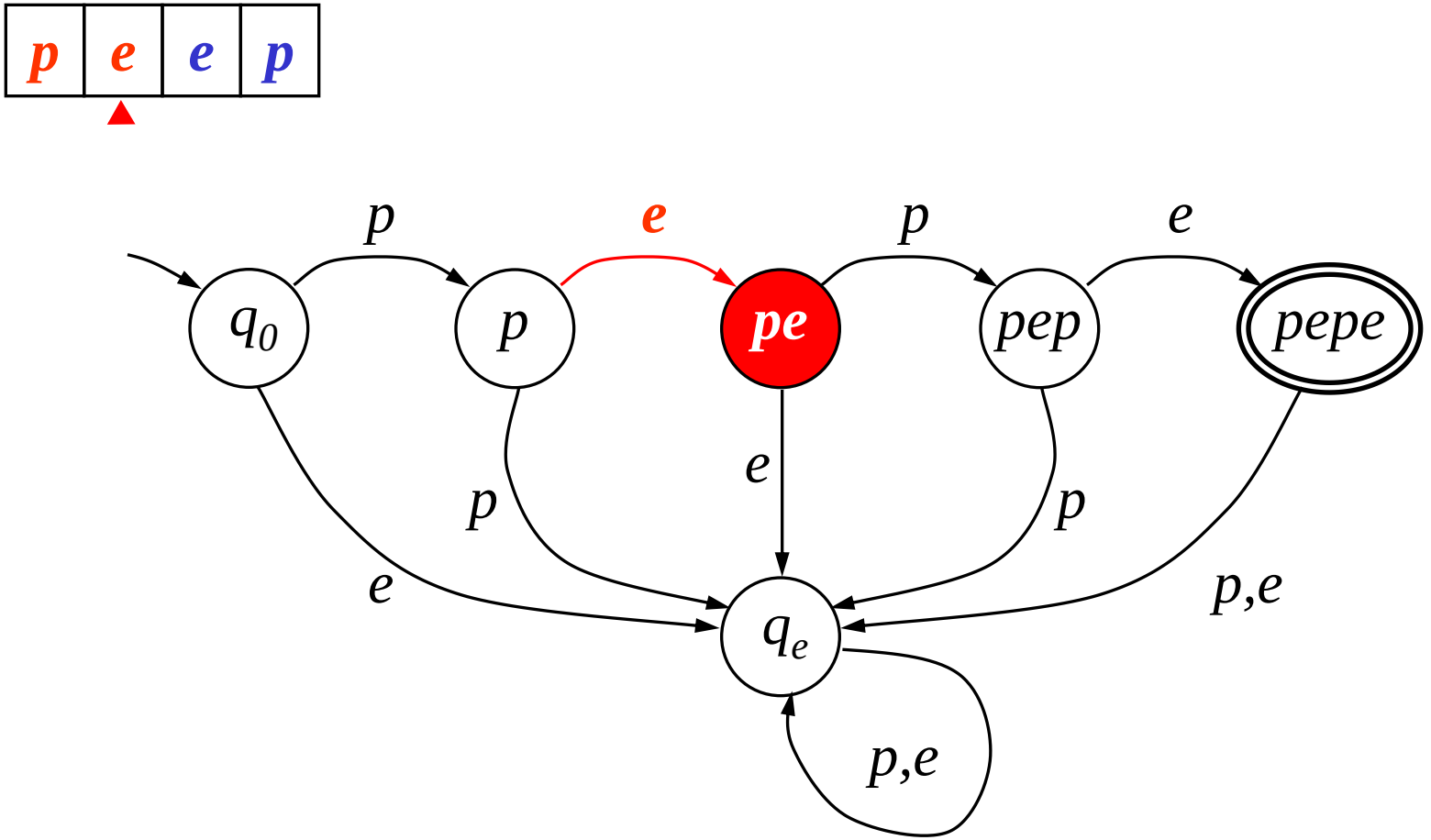
- Puede comprobarse ahora el procesamiento de la palabra de entrada *peep*.





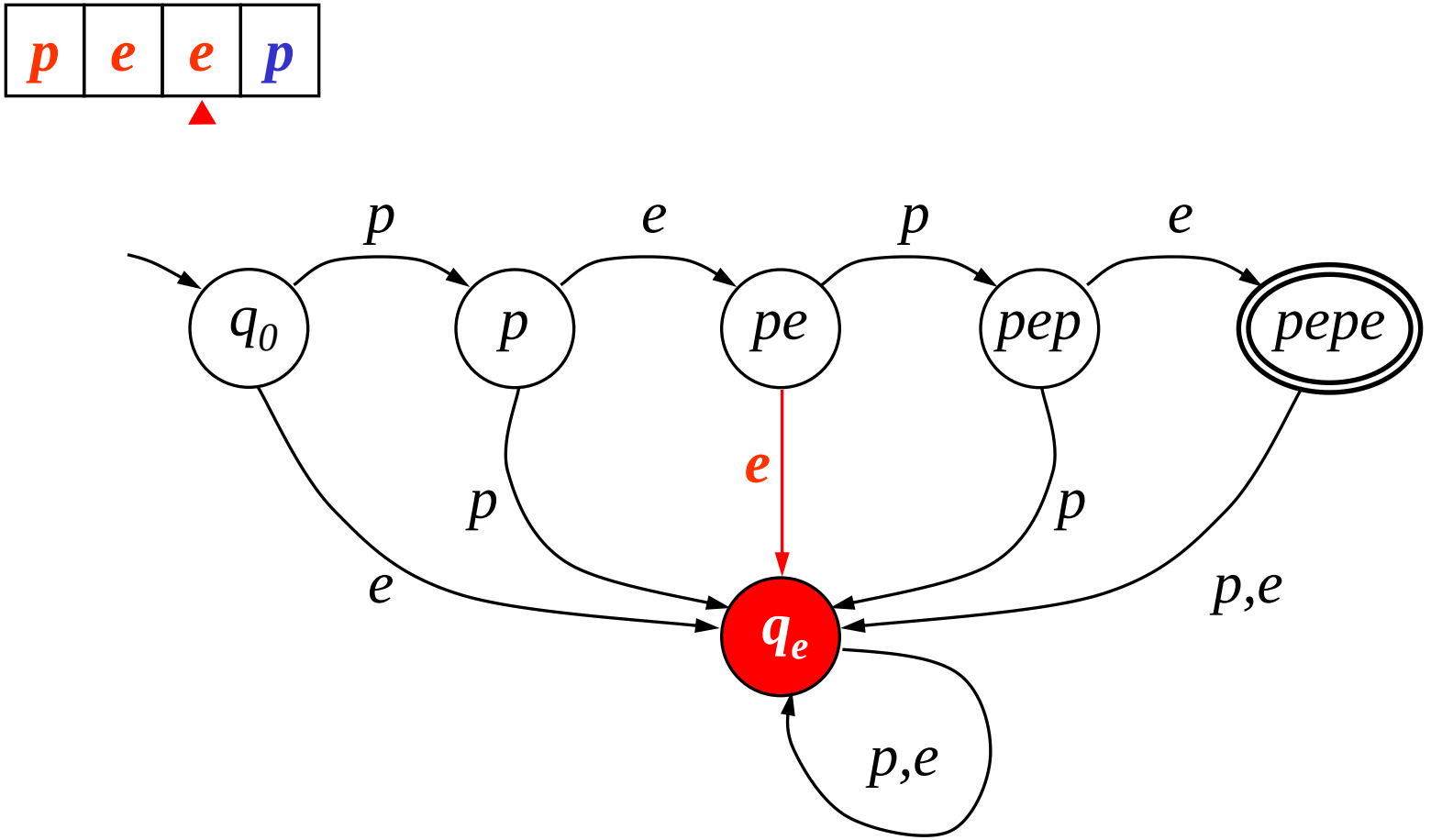
Ej: función de transición con estados finales para el reconocimiento de una palabra

- Puede comprobarse ahora el procesamiento de la palabra de entrada *peep*.



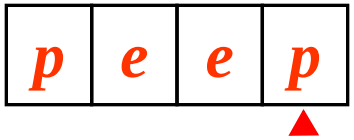
Ej: función de transición con estados finales para el reconocimiento de una palabra

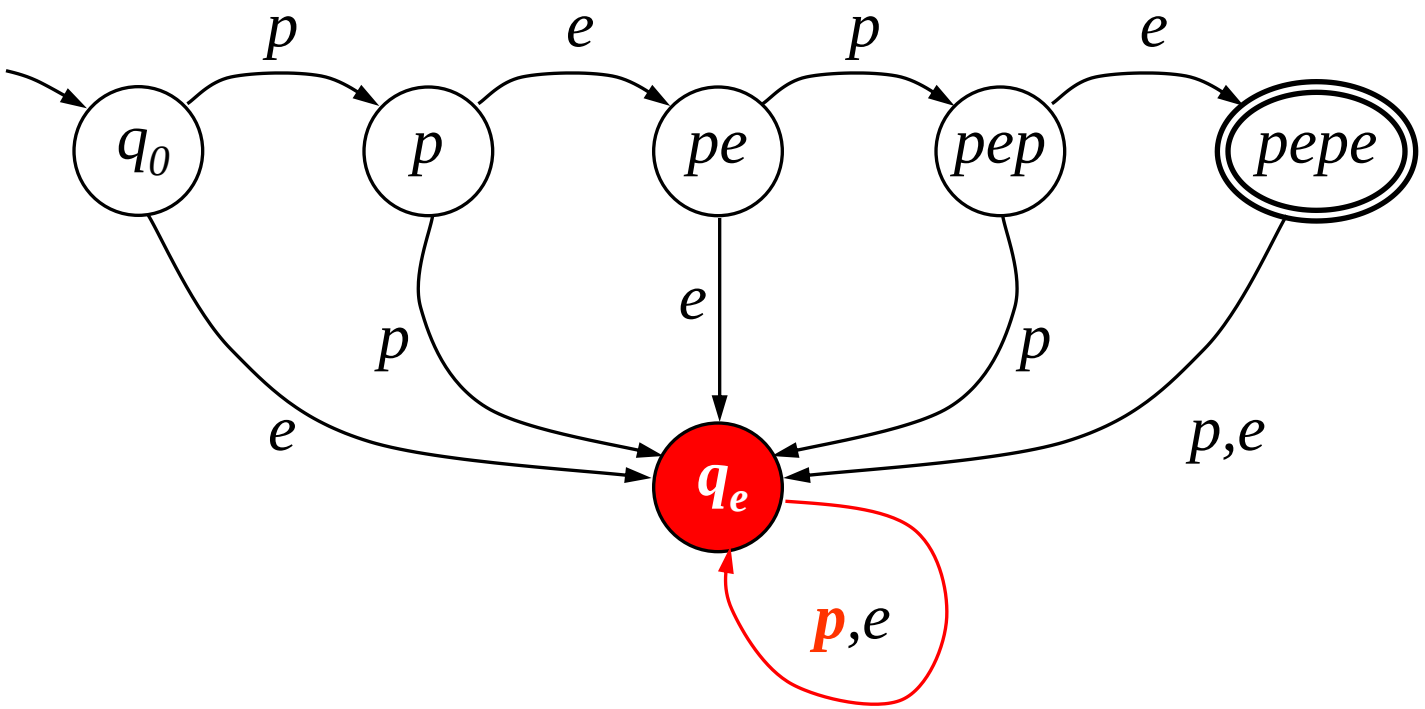
- Puede comprobarse ahora el procesamiento de la palabra de entrada *peep*.



Ej: función de transición con estados finales para el reconocimiento de una palabra

- Puede comprobarse ahora el procesamiento de la palabra de entrada *peep*. El autómata termina en un estado no final: la cadena se rechaza.





Introducción

- Tras comprender estos ejemplos de transiciones entre estados a partir de símbolos de entrada, será muy fácil comprender cuál es la estructuras de los autómatas finitos
  - Los autómatas finitos se pueden entender como un modelo de cómputo en el que sólo se dispone de la posibilidad de transitar entre diferentes estados
  - En este sentido, a continuación, veremos que los ejemplos analizados de funciones de transición
    - Interruptor
    - Reconocimiento de la palabra *pepe*.
  - Describen directamente sendos autómatas finitos (en este caso) deterministas.

Definición formal

- Un **autómata finito determinista** se puede definir como una quíntupla

$$A=(Q, \Sigma, \delta, q_0, F)$$

- Donde:
  - Q es un conjunto finito no vacío de **estados**.
  - Σ es el **alfabeto de entrada**.
  - $\delta: Q \times \Sigma \rightarrow Q$ es la **función de transición** que, a cada pareja de estados y símbolos de entrada (q,a), le asigna un nuevo estado $q' = \delta(q,a)$.
  - $q_0 \in Q$ es el **estado inicial** del autómata.
  - $F \subseteq Q$ es el **conjunto de estados de finalización** del autómata.

Definición formal: estados

- Los **estados** del autómata
  - Tienen como objetivo **recordar la historia** del sistema.
  - La **cantidad** de estados posibles es **finita** y se recuerda **sólo lo relevante**.
  - El hecho de que sea **finita** permitirá que se escriban algoritmos para **simular** los autómatas finitos con una **cantidad de recursos acotada**, es decir, los algoritmos no necesitarán una cantidad creciente de recursos (por ejemplo memoria) que puedan hacer que el programa falle por agotamiento de los mismos.
- El **estado inicial**
  - Recoge la **situación** en la que se encuentra el autómata **inicialmente**.
- Los **estados finales**
  - Representan las situaciones en las que, de llegar el autómata a ellas, se considera que su funcionamiento **ha finalizado satisfactoriamente**.

Definición formal: estados

- Las **entradas**
  - Representan los **estímulos** que pueden llegar desde el **exterior** y afectar el comportamiento del autómata.
- Las **transiciones** entre estados del autómata asociadas a una entrada
  - Representan la manera en la que los **estímulos externos modifican** el comportamiento del autómata.

Formalización del ejemplo del interruptor

- A continuación se muestra la formalización del ejemplo del interruptor:

$$A_1=(Q_1, \Sigma_1, \delta_1, 0, \varnothing)$$

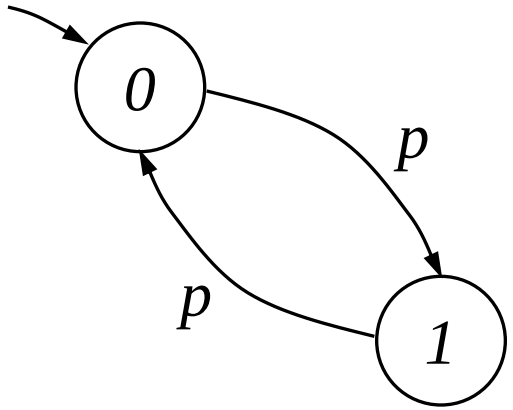
- Donde
  - El **conjunto de estados** representa **encendido** con 1 y **apagado** con 0.

$$Q_1=\{0, 1\}$$

  - El **alfabeto de entrada** representa **pulsar** con p.

$$\Sigma_1=\{p\}$$

  - El **estado inicial** q_0 es que esté **apagado** (0).
  - En este autómata no se indican **estados finales** ($\varnothing$ es el conjunto vacío).
  - La **función de transición** queda definida por el siguiente diagrama

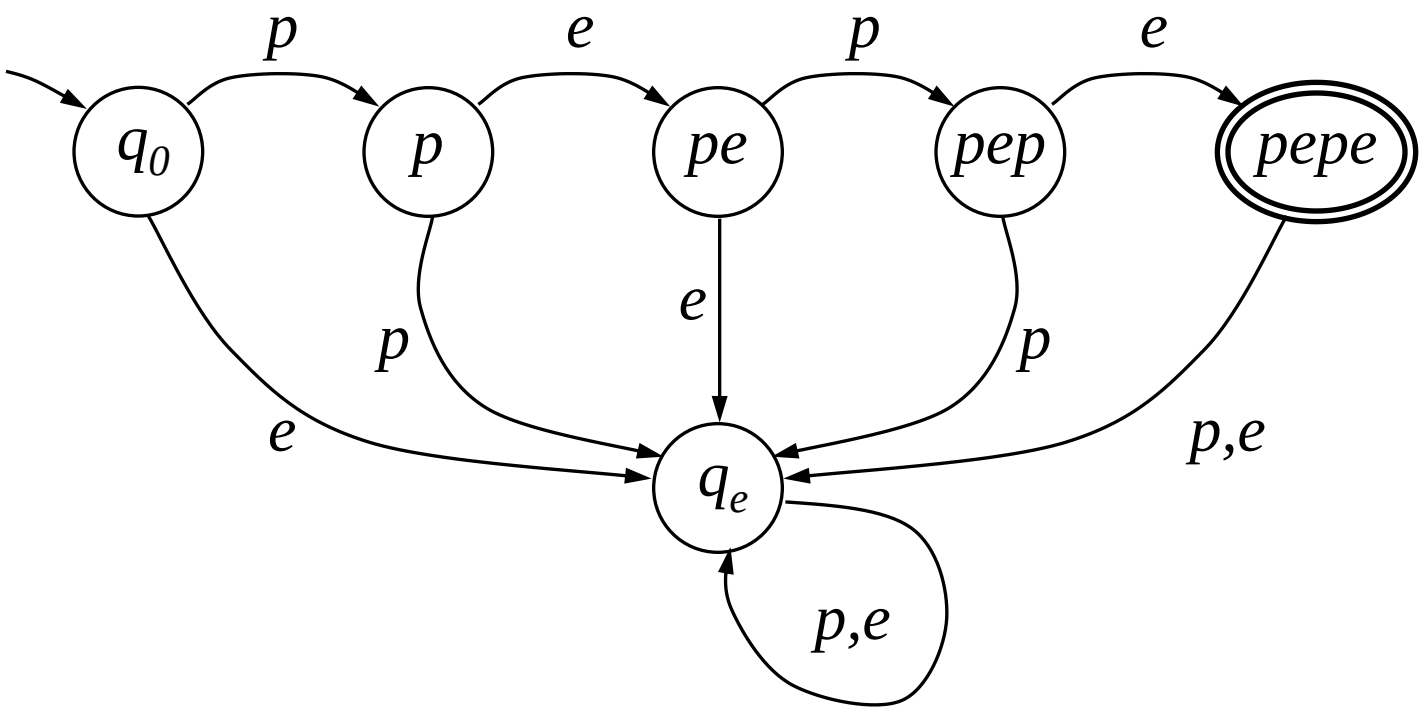


Formalización del ejemplo reconocedor de *pepe*

- Por lo que el ejemplo completo se describe formalmente de la siguiente forma:

$$A_2=(Q_2, \Sigma_2, \delta_2, q_0, \{pepe\})$$

- Donde
  - El **conjunto de estados**: $Q_2=\{q_0, p, pe, pep, pepe, q_e\}$
  - El **alfabeto de entrada**:$\Sigma_2=\{p, e\}$
  - La **función de transición** δ_2 es la del diagrama siguiente.



Tratamiento de cadenas de entrada: extensión de δ a palabras

- Se ha analizado intuitivamente al presentar las transiciones entre estados cuál es el efecto de que un dispositivo de cómputo analice una cadena de entrada.
- En el caso de los autómatas finitos (deterministas) este concepto no entraña ninguna dificultad.
- Llamaremos $\hat{\delta}$ a la función de transición extendida a palabras, es decir, aquella que genera el mismo resultado en el que queda el autómata tras procesar en el orden en el que aparecen y a partir del estado inicial, todos los símbolos de la cadena.
- La única utilidad que le daremos a este concepto en este curso será la definición del lenguaje reconocido por un autómata.

Definiciones

- Dado un autómata finito determinista

$$A=(Q, \Sigma, \delta, q_0, F)$$

- Se llama **lenguaje del autómata** A, $L(A)$ o **lenguaje aceptado por** A y se define de la siguiente manera:

$$L(A)=\{w \in \Sigma^* \mid \hat{\delta}(w, q_0) \in F\}$$

- Y es fácil comprobar que

$$epe \notin L(A)$$
$$epp \notin L(A)$$
$$epppp...p \notin L(A)$$
$$epeee...e \notin L(A)$$
$$epeppeepep\,e \notin L(A)$$

Autómatas Finitos

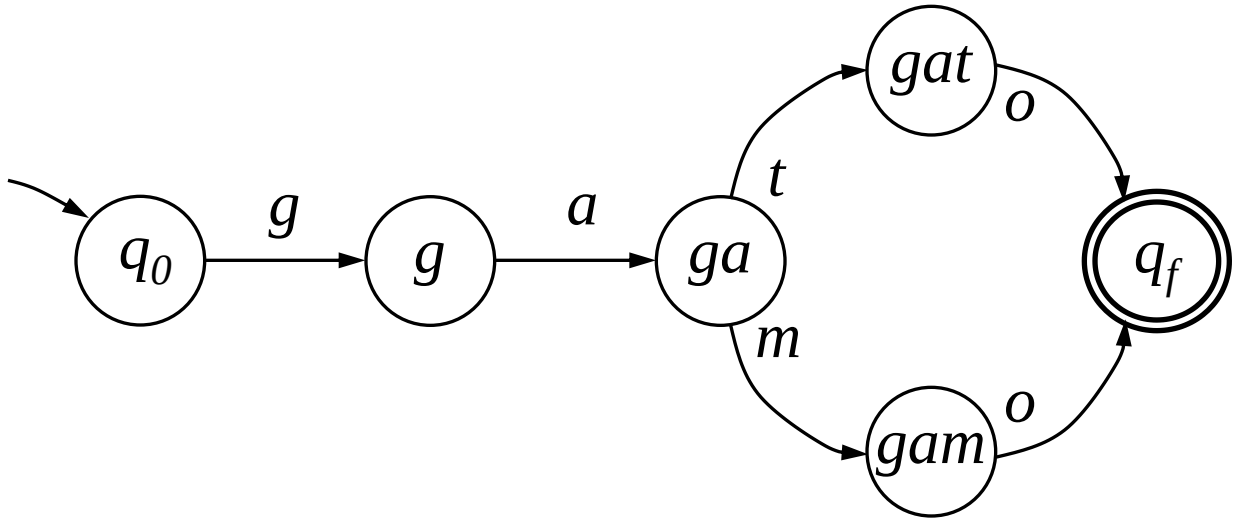
Autómatas finitos no deterministas

Descripción informal intuitiva

- Abordamos a continuación la manera en la que los autómatas finitos pueden incorporar el no determinismo.
- El concepto de no determinismo es tal vez inquietante pero fácil de analizar de manera sistemática.

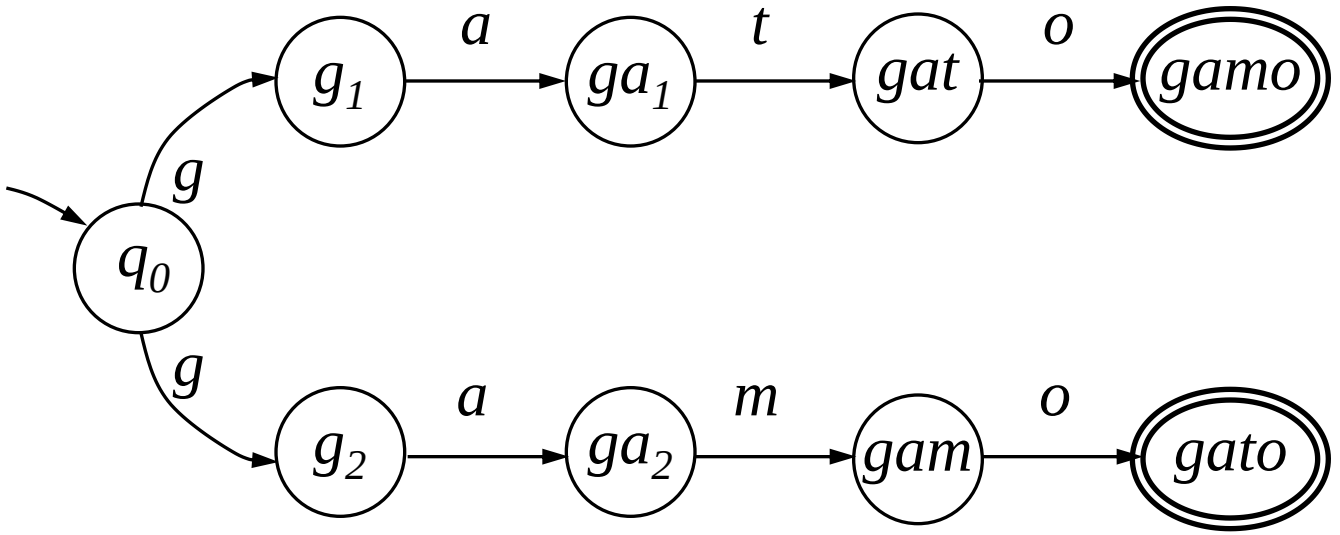
Descripción informal intuitiva

- Los autómatas finitos no deterministas pueden considerarse una *extensión* de los deterministas en los que **se permite que**, desde **un estado y ante un símbolo** de entrada, el autómata **transite a más de un estado**.
- Esto se puede interpretar como que el autómata *sigue en paralelo dos o más hipótesis respecto a la entrada*.
- Por ejemplo, supóngase que se quiere diseñar un autómata finito que reconozca el lenguaje $\{gato, gamo\}$
- Para ello puede utilizarse el siguiente diagrama de transiciones determinista, en el que se ha omitido el estado sumidero para el rechazo del resto de las transiciones:



Descripción informal intuitiva

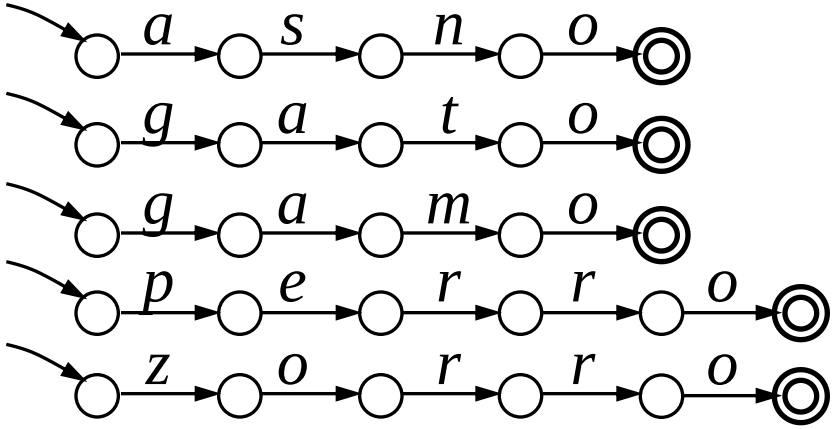
- Otra estrategia consiste en suponer que la primera g es el carácter inicial, o bien de la palabra *gato*, o bien de la palabra *gamo*, como muestra el siguiente diagrama de transiciones:



- Que:
  - Presenta el **inconveniente** de no ser determinista ya que desde q_0 con el símbolo g se pude ir a dos estados distintos: g_1 y g_2.
  - Ofrece la **ventaja** de proporcionar una técnica de diseño más sencilla: se dedica una *rama no determinista* a cada palabra que se desea reconocer.

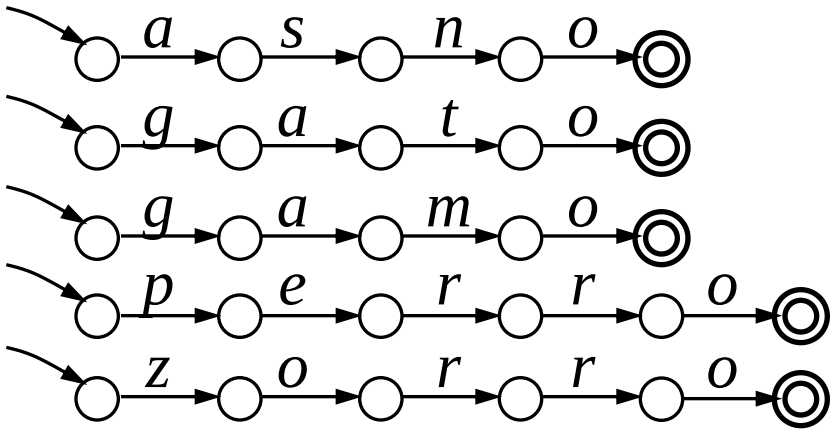
Descripción informal intuitiva: transiciones λ

- Los autómatas finitos no deterministas incorporan otra posibilidad: **cambiar de estado sin consumir ningún símbolo de entrada**.
- Así, se podría generalizar la técnica de diseño sugerida antes de la siguiente manera:
  - Supóngase que se desea diseñar un autómata finito para el reconocimiento del lenguaje {*asno*, *gato*, *gamo*, *perro*, *zorro*}
  - Es fácil diseñar los autómatas finitos deterministas para cada una de las palabras (se omiten los nombres de los estados)



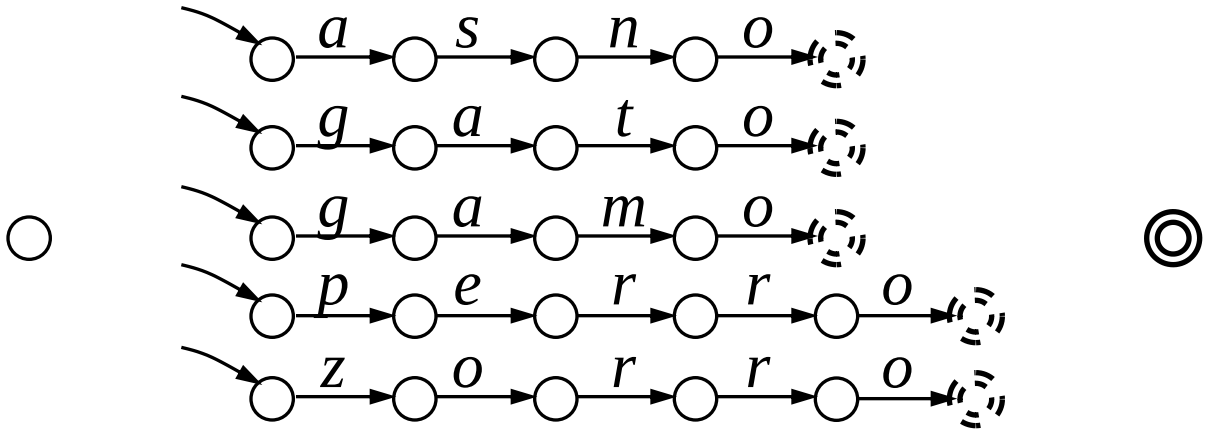
Descripción informal intuitiva: transiciones λ

- Y, si se puede transitar a un estado sin consumir entrada, es muy sencillo construir un autómata finito (no determinista) casi sin cambiar los anteriores:



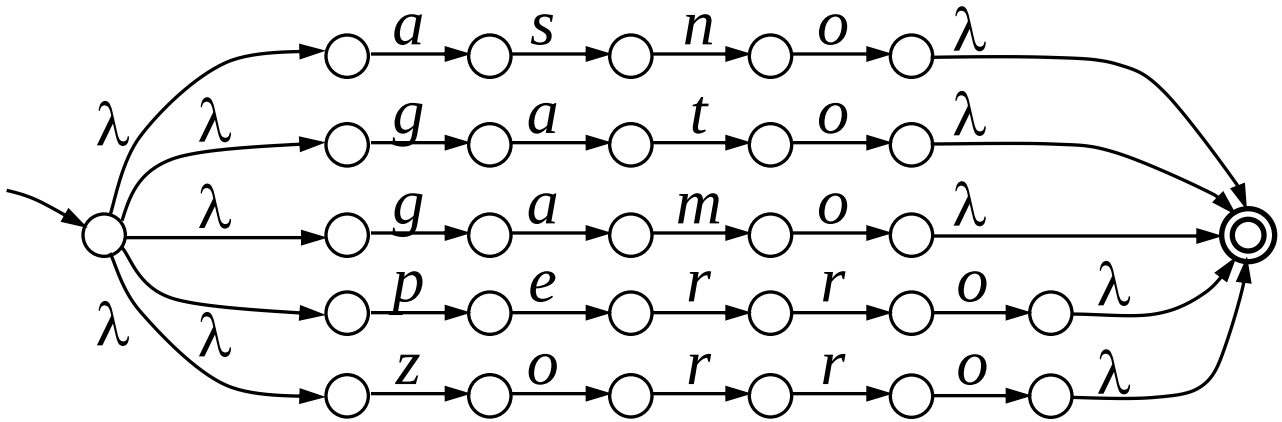
Descripción informal intuitiva: transiciones λ

- Y, si se puede transitar a un estado sin consumir entrada, es muy sencillo construir un autómata finito (no determinista) casi sin cambiar los anteriores:



Descripción informal intuitiva: transiciones λ

- Y, si se puede transitar a un estado sin consumir entrada, es muy sencillo construir un autómata finito (no determinista) casi sin cambiar los anteriores:



Definición formal

- Un **autómata finito (no necesariamente determinista)** se puede definir como una quíntupla

$$A=(Q, \Sigma, \delta, q_0, F)$$

- Donde todo tiene el mismo significado que en el caso de los autómatas finitos:
  - Q es un conjunto finito no vacío de **estados**.
  - Σ es el **alfabeto de entrada**.
  - $q_0 \in Q$ es el **estado inicial** del autómata.
  - $F \subseteq Q$ es el **conjunto de estados de finalización** del autómata.
- Excepto la función de transición:
  - $\delta: Q \times \Sigma \cup \{\lambda\} \rightarrow 2^Q$ donde
    - 2^Q es el conjunto de todos los subconjuntos del conjunto Q
    - la **función de transición** que a cada estado q le asigna el conjunto de estados al que puede transitar
      - mediante el símbolo de entrada a: $\delta(q,a)$;o
      - sin consumir ningún símbolo de entrada (**transiciones λ**): $\delta(q,\lambda)$.

Representaciones

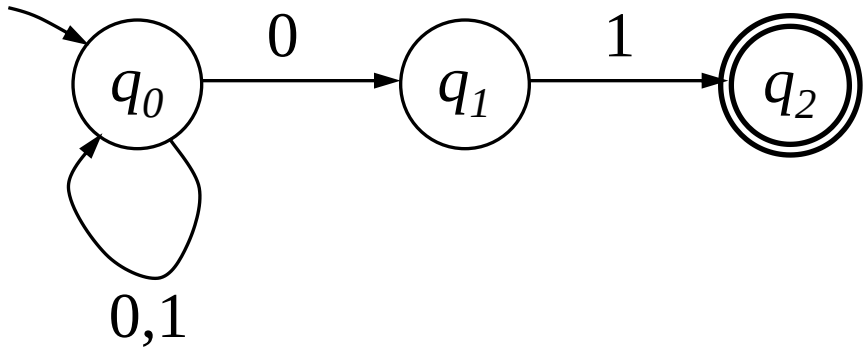
- Para la función de transición de los autómatas finitos no deterministas suelen utilizarse las mismas representaciones que para los deterministas:
  - Tablas de transición
  - Diagramas de transición
- A continuación se muestran algunos ejemplos:

Ejemplo 1

- Considérese el siguiente autómata finito no determinista:

$$A=(Q=\{q_0, q_1, q_2\}, \Sigma=\{0,1\}, \delta, q_0, F=\{q_2\})$$

  - Donde el diagrama de transiciones de δ es el siguiente:



Ejemplo 1

- Que también se puede representar mediante la siguiente tabla de transiciones:

	0	1	λ
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$	Φ
q_1	Φ	$\{*q_2\}$	Φ
$*q_2$	Φ	Φ	Φ

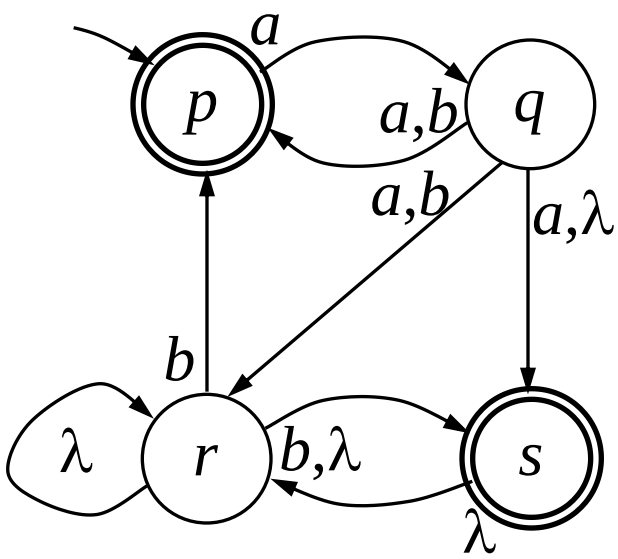
- De cuya última columna resulta claro que no hay transiciones λ

Ejemplo 2

- Considérese el siguiente autómata finito no determinista:

$$A=(Q=\{p,q,r,s\},\ \Sigma=\{a,b\},\ \delta,\ p,\ F=\{p,s\})$$

  - Donde el diagrama de transiciones de δ es el siguiente:



Ejemplo 2

- Que también se puede representar mediante la siguiente tabla de transiciones:

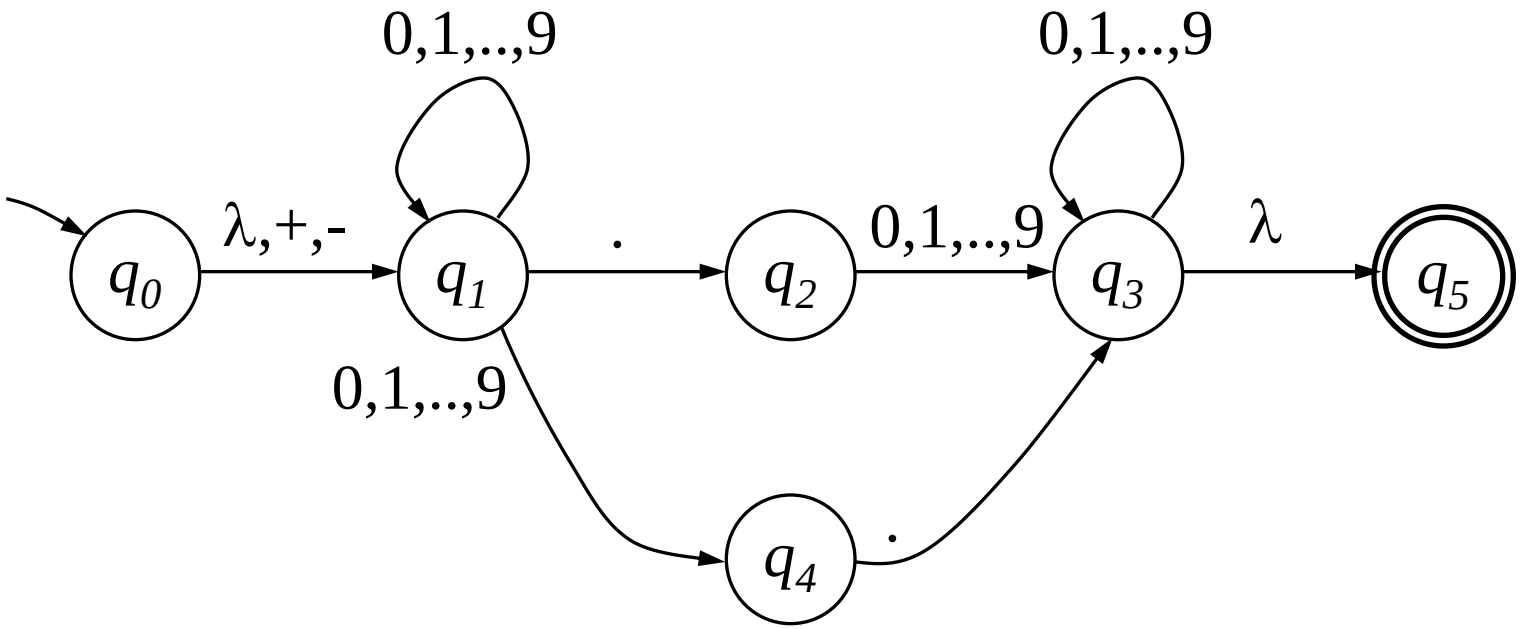
	a	b	λ
$\rightarrow *p$	$\{q\}$	Φ	Φ
q	$\{*p,r,*s\}$	$\{*p,r\}$	$\{*s\}$
r	Φ	$\{*p,*s\}$	$\{r,*s\}$
$*s$	Φ	Φ	$\{r\}$

Ejemplo 3

- Considérese el siguiente autómata finito no determinista:

$$A=(Q=\{q_0, q_1, q_2, q_3, q_4, q_5\}, \Sigma=\{0,1,...,9,+,-,.\}, \delta, q_0, F=\{q_5\})$$

  - Donde el diagrama de transiciones de δ es el siguiente:



Ejemplo 2

- Que también se puede representar mediante la siguiente tabla de transiciones (obsérvese que se han agrupado en una sola columna varias que serían idénticas; el nombre de la columna acumula todos los nombres como en +,- y 0,1,...,9, la tabla real tiene 14 columnas):

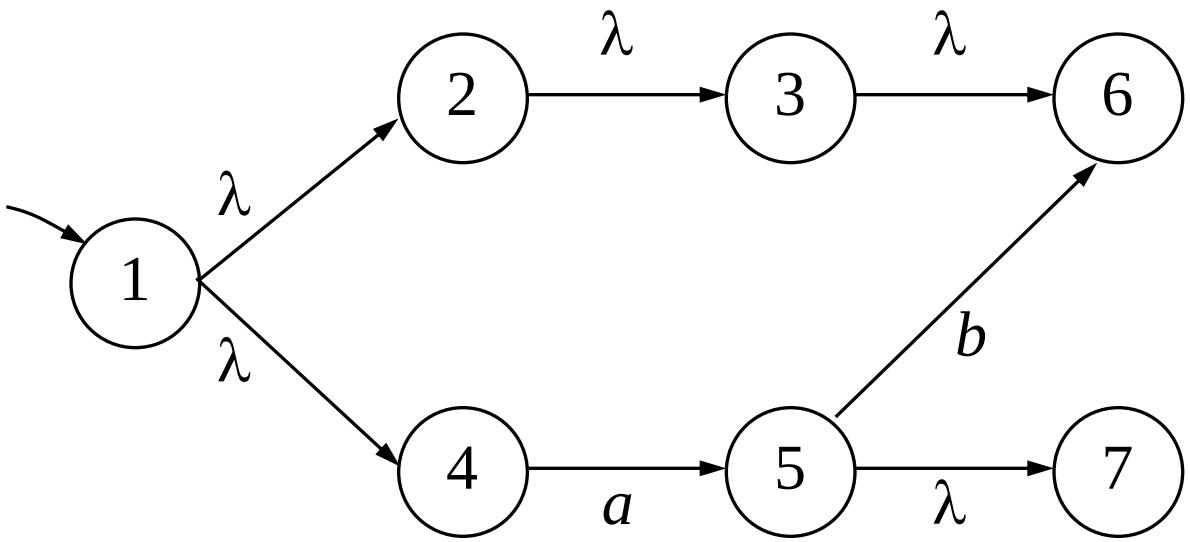
	+,-	0,1,...,9	.	λ
$\rightarrow q_0$	$\{q_1\}$	Φ	Φ	$\{q_1\}$
q_1	Φ	$\{q_1, q_4\}$	$\{q_2\}$	Φ
q_2	Φ	$\{q_3\}$	Φ	Φ
q_3	Φ	$\{q_3\}$	Φ	$\{*q_5\}$
q_4	Φ	Φ	$\{q_3\}$	Φ
$*q_5$	Φ	Φ	Φ	Φ

Ejemplo 4

- Considérese el siguiente autómata finito no determinista:

$$A=(Q=\{1, 2, 3, 4, 5, 6, 7\}, \Sigma=\{a, b\}, \delta, 1, F=\Phi)$$

  - Donde el diagrama de transiciones de δ es el siguiente:



Ejemplo 4

- Que también se puede representar mediante la siguiente tabla de transiciones:

	a	b	λ
$\rightarrow 1$	Φ	Φ	$\{2,4\}$
2	Φ	Φ	$\{3\}$
3	Φ	Φ	$\{6\}$
4	$\{5\}$	Φ	Φ
5	Φ	$\{6\}$	$\{7\}$
6	Φ	Φ	Φ
7	Φ	Φ	Φ

Observaciones: los autómatas finitos no deterministas incluyen los deterministas

- Resulta fácil comprobar el siguiente resultado

$$AF \subseteq AFN$$

- Ya que un autómata finito determinista no es más que un autómata finito no determinista en el que:
  - Las imágenes de la función de transición son siempre conjuntos unitarios.
  - No hay transiciones λ.

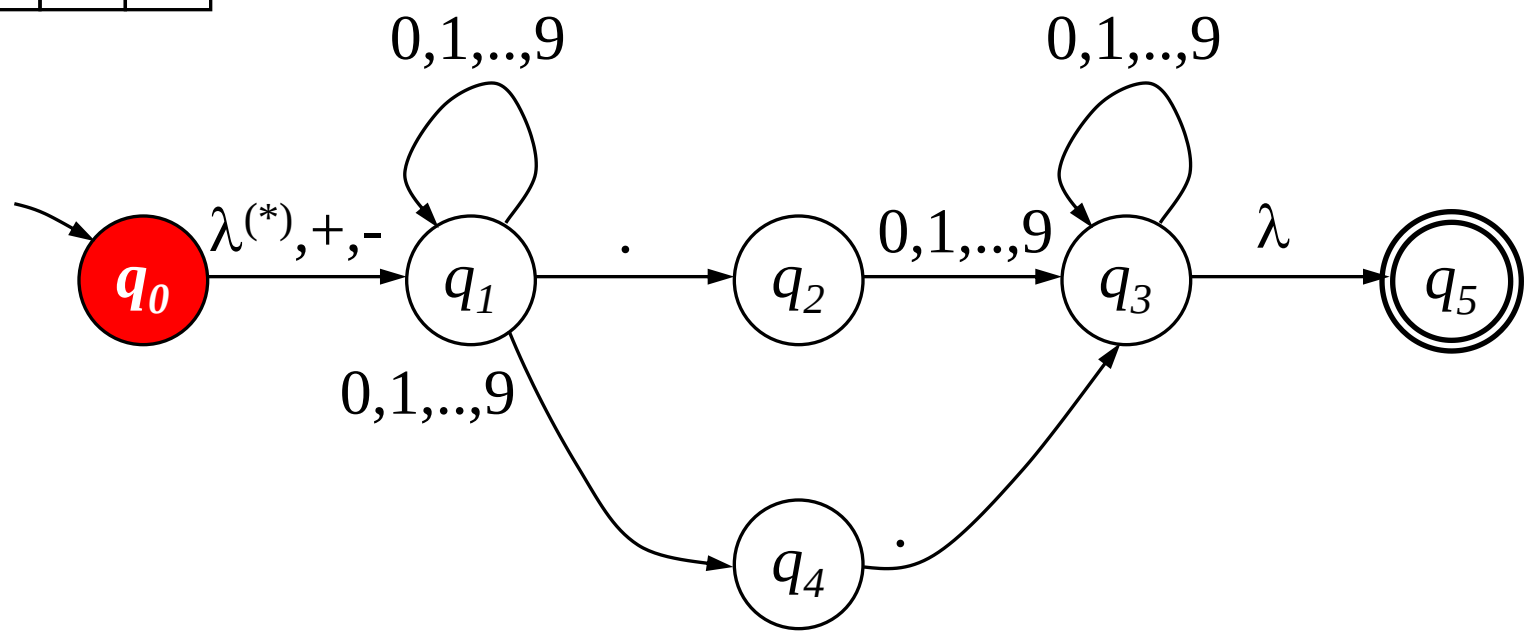
Observaciones: estados sumidero en AFN

- Los AFN suponen una alternativa de diseño de autómatas finitos **más cómoda**, en ocasiones, que los AF.
- En este sentido, una diferencia entre los AFN y los AF es que los deterministas deben especificar de forma explícita todas las transiciones posibles.
- Los **estados sumidero** son una técnica para satisfacer esa necesidad.
- La función de transición de los AFN ($\delta: Q \times \Sigma \cup \{\lambda\} \rightarrow 2^Q$) genera un conjunto que puede **ser vacío** ($\varnothing$) para los casos en los que no se desee especificar una transición, de forma que:
  - Estas transiciones no aparecen en el diagrama de transiciones del autómata
  - Y no se necesita incorporar estados adicionales como sumideros para las transiciones no deseadas.

Tratamiento de entrada, ejemplo

- Puede observarse el comportamiento del ejemplo 3 estudiado anteriormente:
  - Frente a la entrada: +3.1

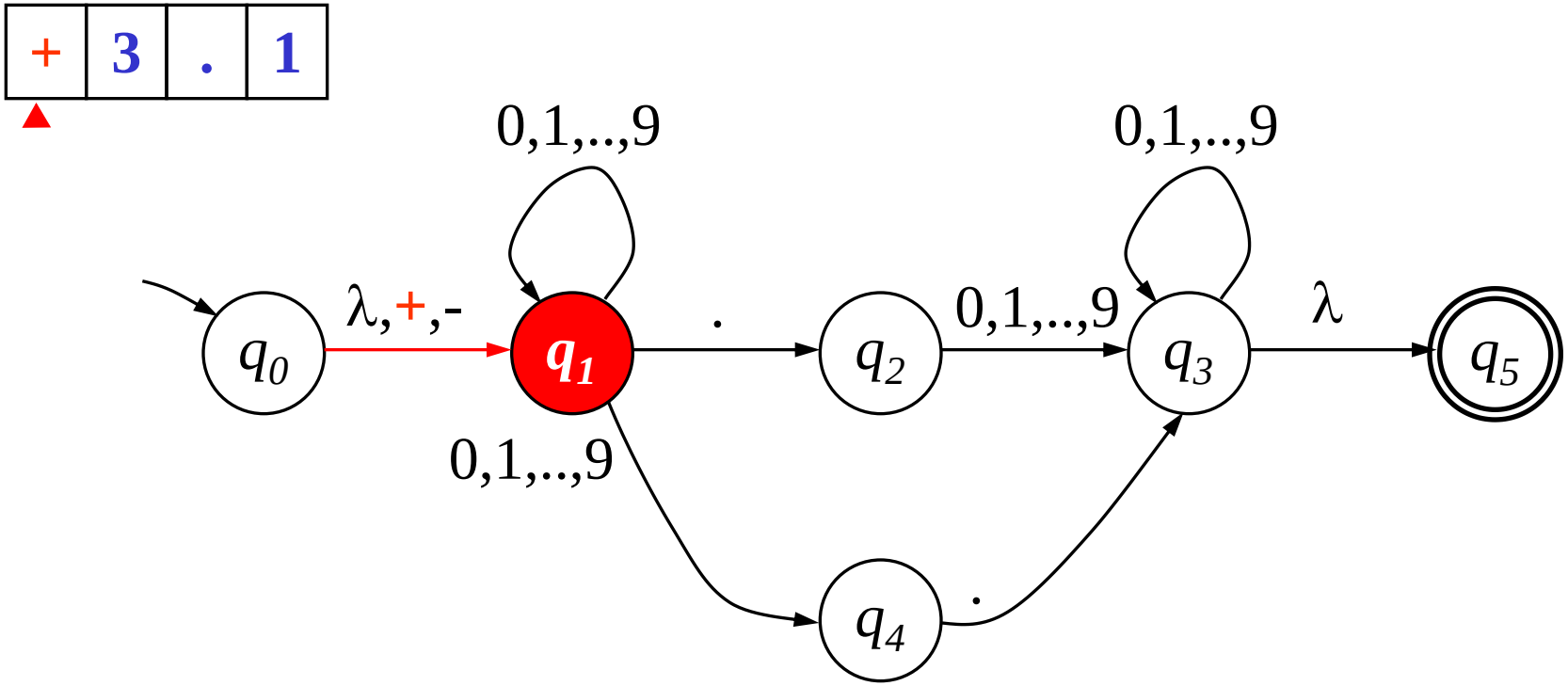
+	3	.	1



(*) Inicialmente, sin consumir entrada, esta transición λ permitiría pasar al estado q_1. Esa *rama no determinista* terminaría acto seguido ya que con el símbolo + no se puede transitar a ningún sitio desde q_1. Por esa razón, para simplificar el diagrama, se omite la transición.

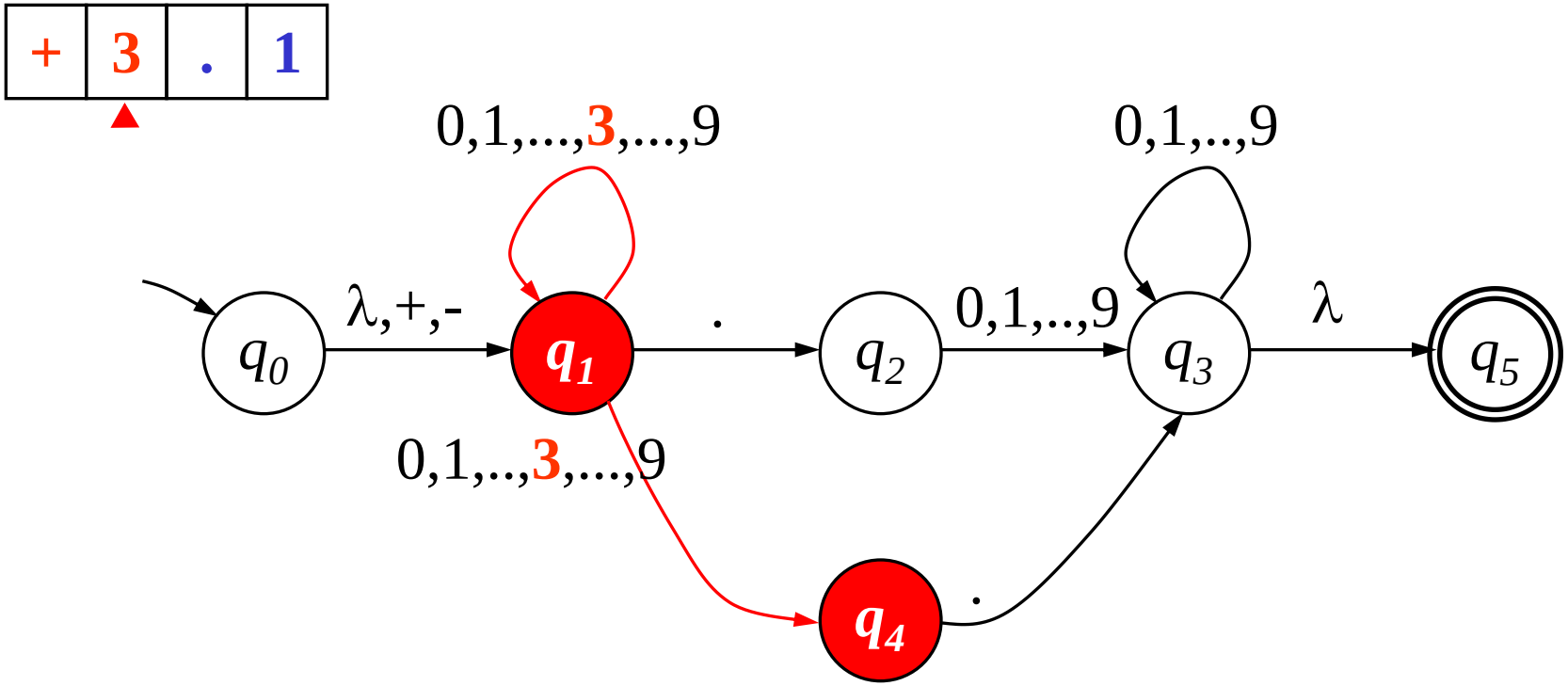
Tratamiento de entrada, ejemplo

- Puede observarse el comportamiento del ejemplo 3 estudiado anteriormente:
  - Frente a la entrada: +3.1



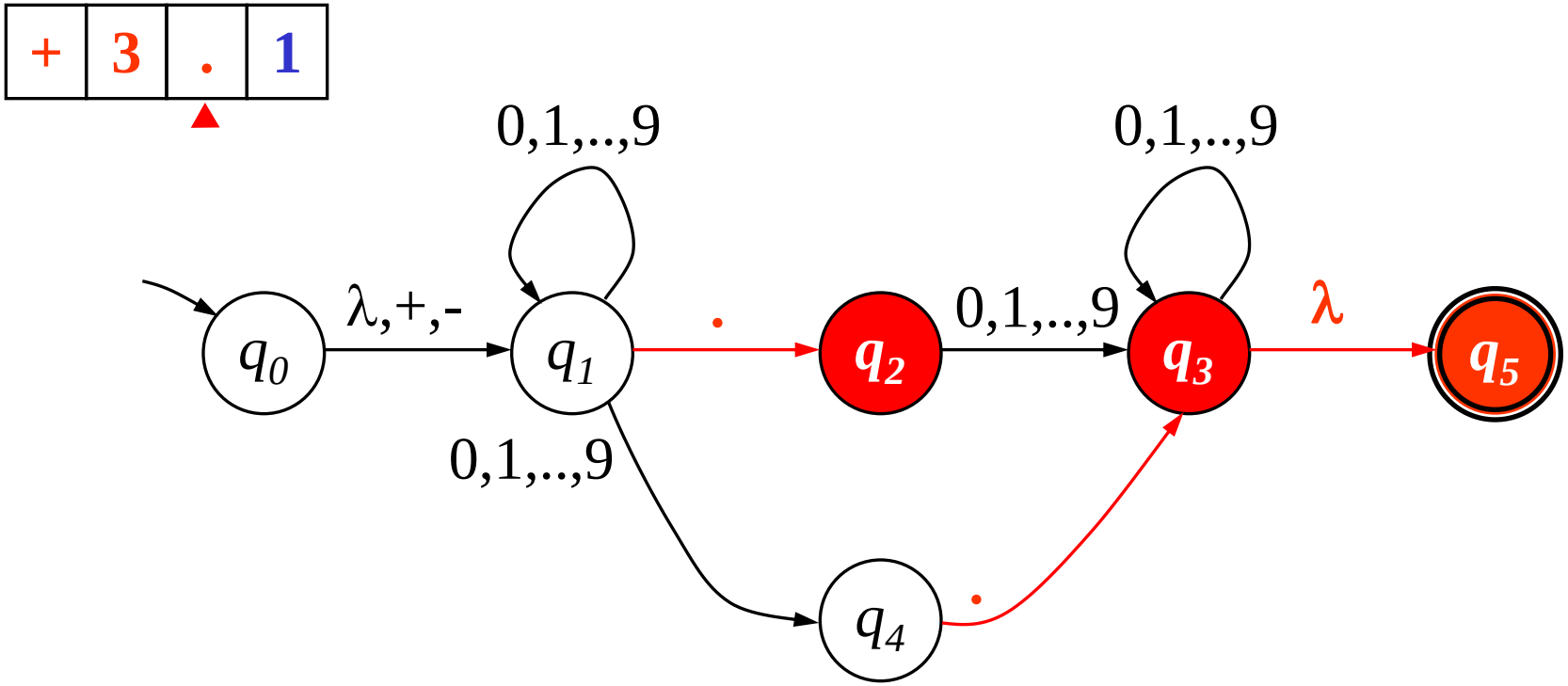
Tratamiento de entrada, ejemplo

- Puede observarse el comportamiento del ejemplo 3 estudiado anteriormente:
  - Frente a la entrada: +3.1



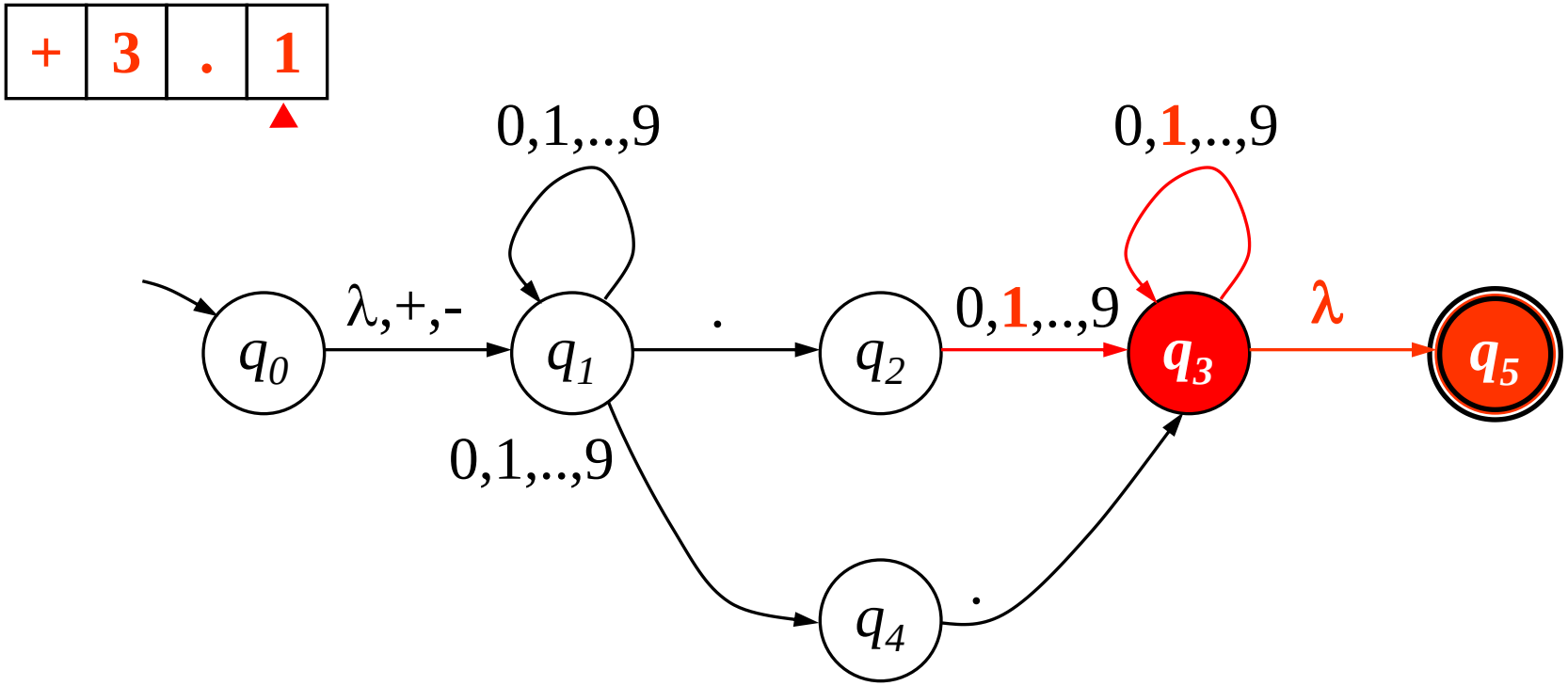
Tratamiento de entrada, ejemplo

- Puede observarse el comportamiento del ejemplo 3 estudiado anteriormente:
  - Frente a la entrada: +3.1



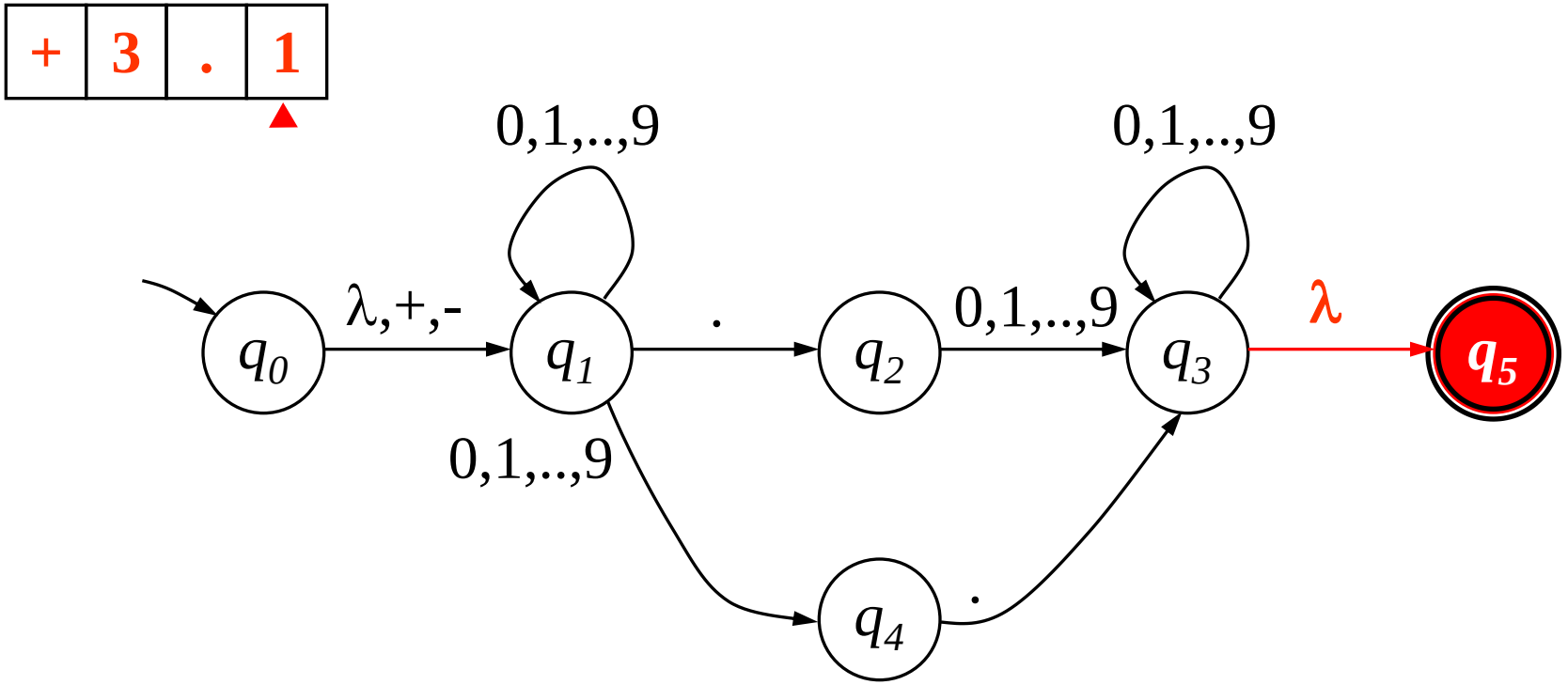
Tratamiento de entrada, ejemplo

- Puede observarse el comportamiento del ejemplo 3 estudiado anteriormente:
  - Frente a la entrada: +3.1



Tratamiento de entrada, ejemplo

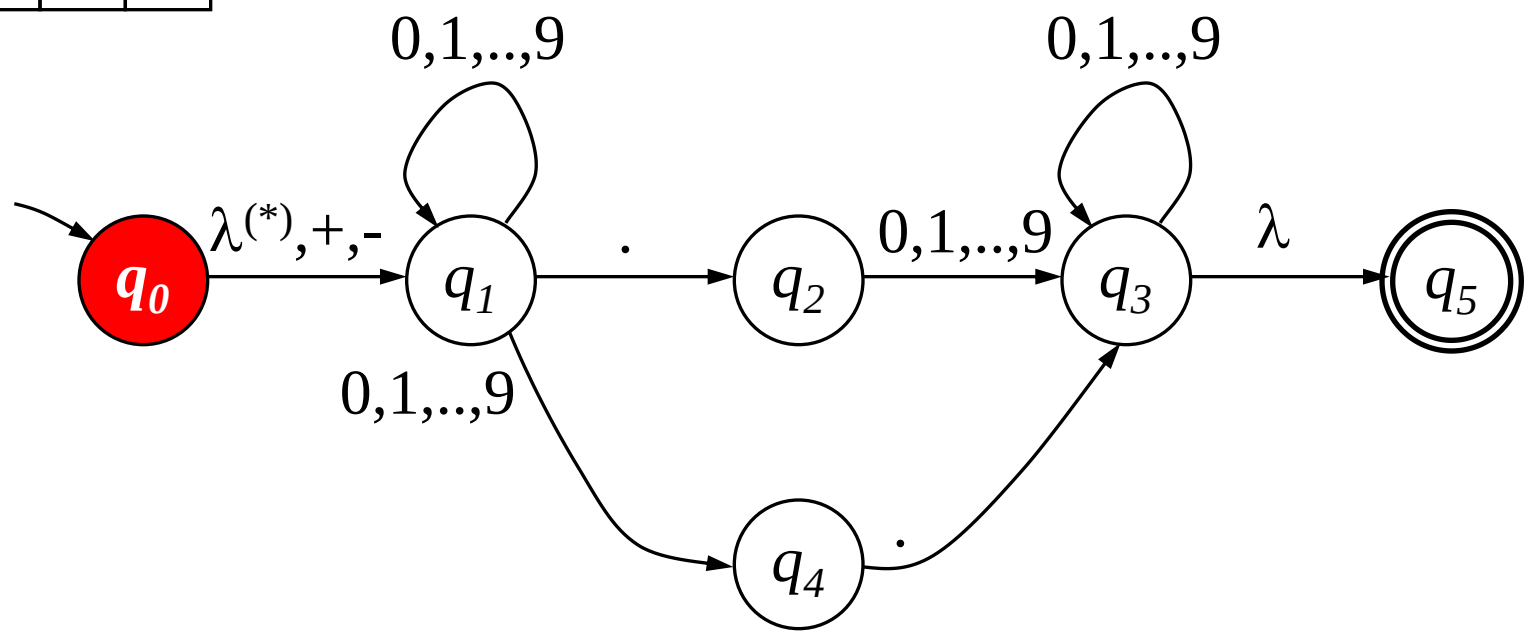
- Puede observarse el comportamiento del ejemplo 3 estudiado anteriormente:
  - Frente a la entrada: +3.1, que es aceptada ya que existe alguna línea de ejecución que termina en el estado final.



Tratamiento de entrada, ejemplo

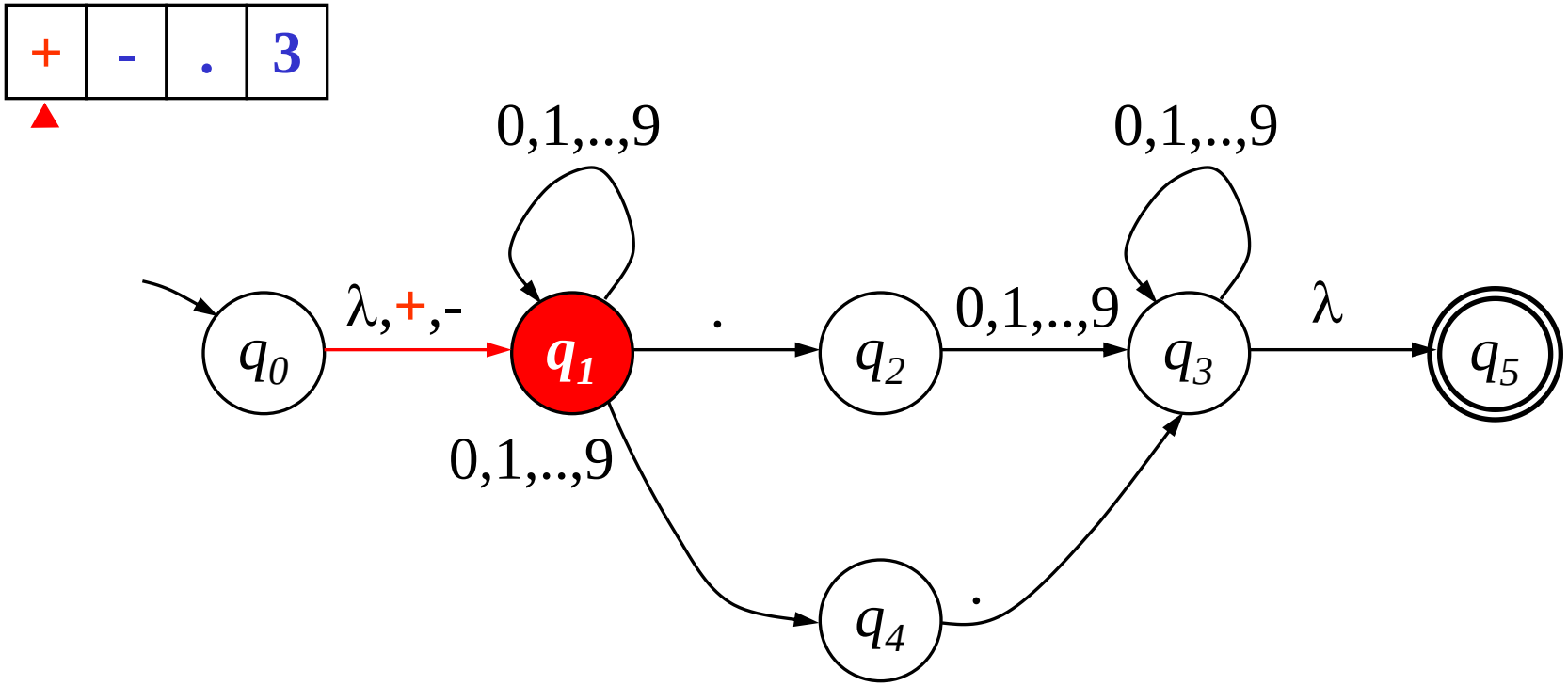
- En comparación con la situación
  - Frente a la entrada: +-.3

+	-	.	3



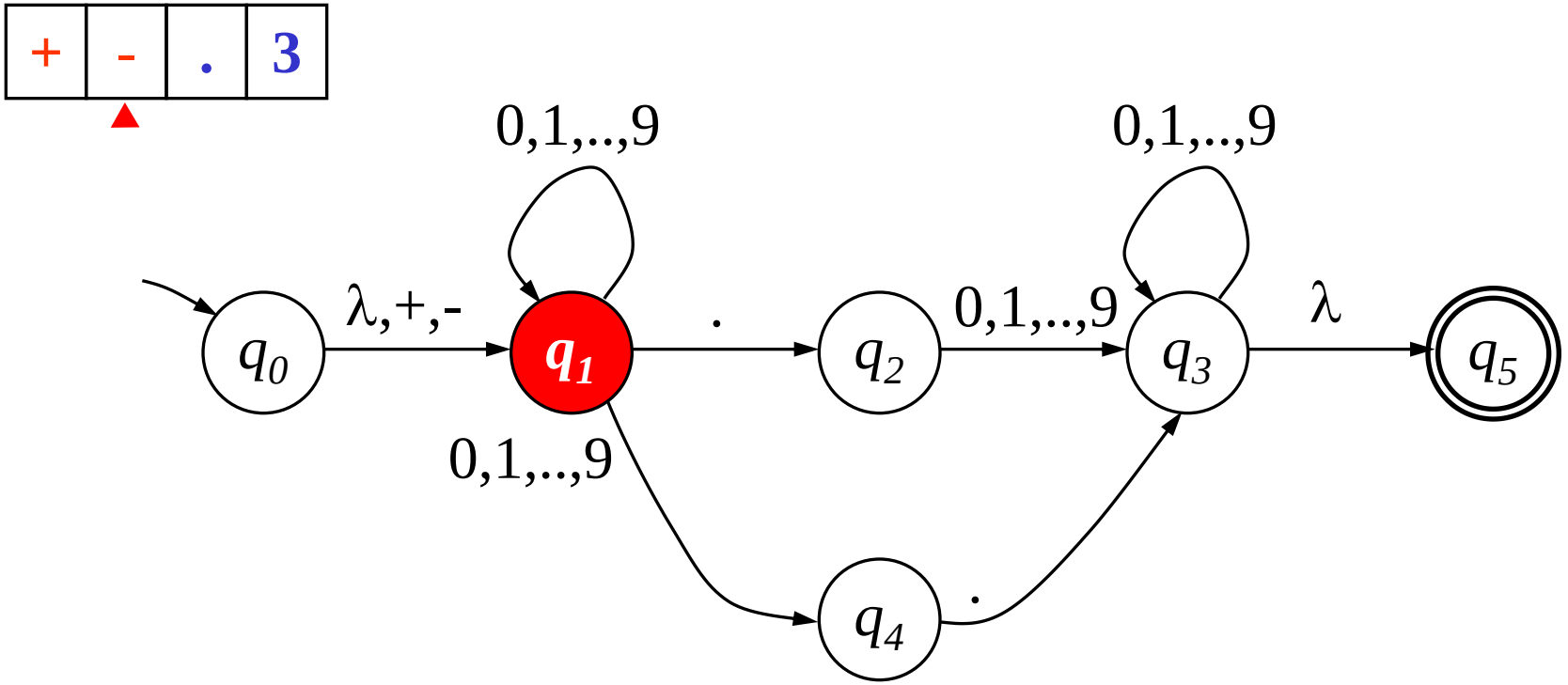
Tratamiento de entrada, ejemplo

- En comparación con la situación
  - Frente a la entrada: +-.3



Tratamiento de entrada, ejemplo

- En comparación con la situación
  - Frente a la entrada: +-.3, que es rechazada, ya que la ejecución termina sin poder transitar a ningún estado ante el siguiente símbolo de entrada.



Tratamiento de cadenas de entrada: extensión de δ a palabras

- Aunque en este curso no se realice un tratamiento formal de este concepto, cuando consideramos el comportamiento de un AFN ante una entrada realmente estamos extendiendo la función de transición a palabras.
- Para definir la extensión de δ (originalmente definida sobre símbolos “simples” de entrada) a palabras tenemos que tener en cuenta las siguientes situaciones
  - Posibilidad de más de una rama en paralelo, a causa del no determinismo.
    - Esta situación es más o menos fácil de capturar
  - El papel de las transiciones λ.
    - Aunque conceptualmente también es simple, en la práctica puede resultar más compleja (implica la idea de “cierre por transiciones de este tipo”)
- A continuación se mostrarán algunos casos para ilustrar ambos aspectos.
- Tras ello se utilizará la extensión a palabras para definir el lenguaje reconocido por un autómata finito no determinista.

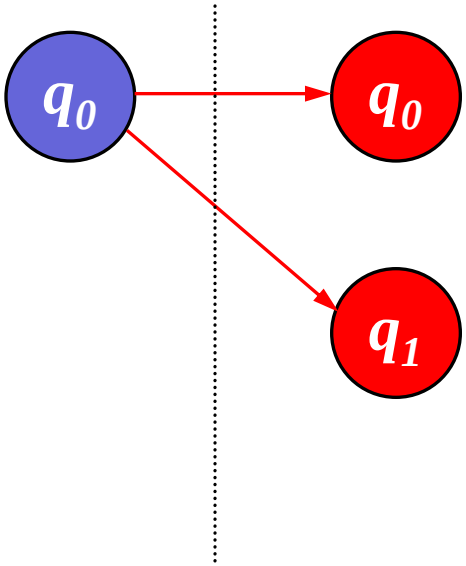
Extensión de δ a palabras sin transiciones λ: introducción

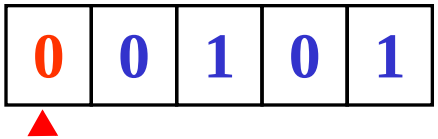
- El ejemplo anterior muestra que las ramas no deterministas en la ejecución de un autómata finito no determinista dificultan el seguimiento de su funcionamiento.
- Se considera a continuación, de nuevo, el AFN del ejemplo 1 frente a la entrada 00101 .
- La tarea es más fácil si se mantiene un árbol en el que cada rama no determinista figure de forma explícita.

0	0	1	0	1

Extensión de δ a palabras sin transiciones λ: introducción

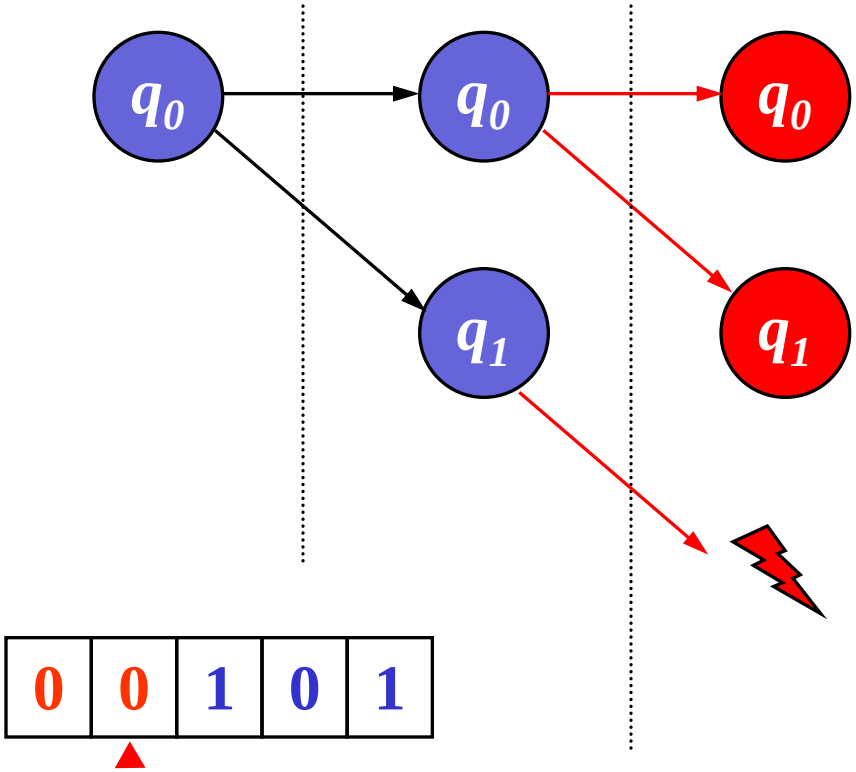
- El ejemplo anterior muestra que las ramas no deterministas en la ejecución de un autómata finito no determinista dificultan el seguimiento de su funcionamiento.
- Se considera a continuación, de nuevo, el AFN del ejemplo 1 frente a la entrada 00101 .
- La tarea es más fácil si se mantiene un árbol en el que cada rama no determinista figure de forma explícita.





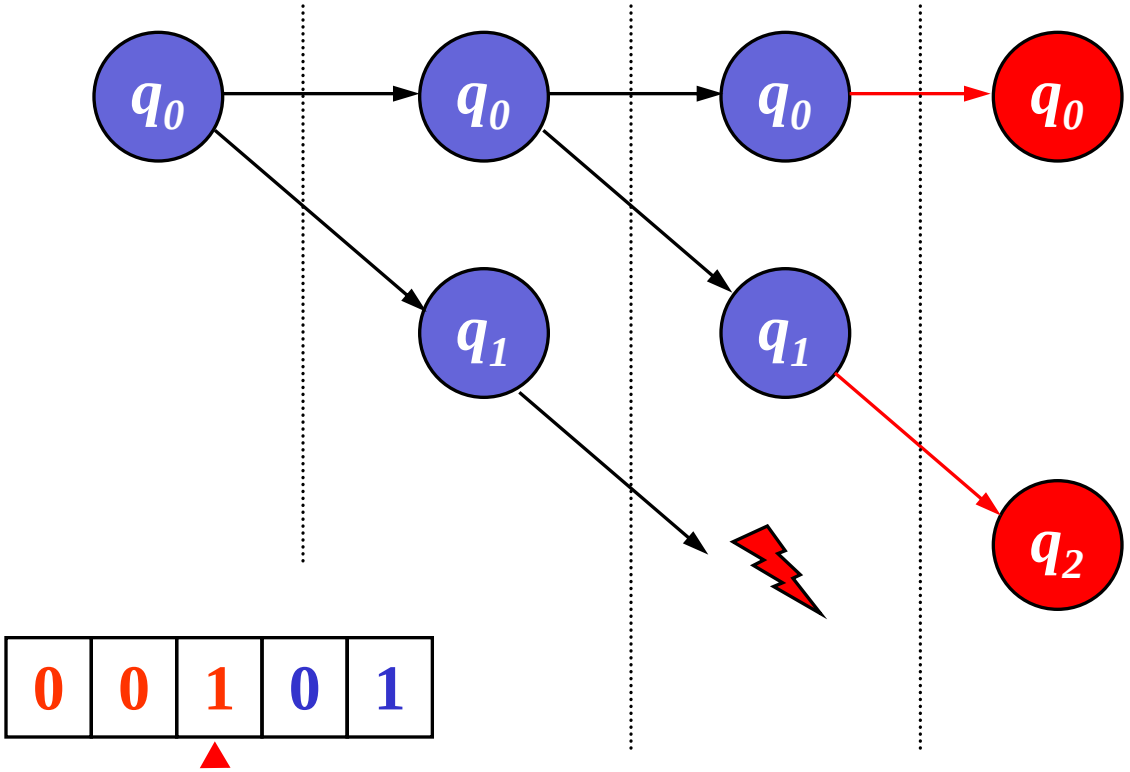
Extensión de δ a palabras sin transiciones λ: introducción

- El ejemplo anterior muestra que las ramas no deterministas en la ejecución de un autómata finito no determinista dificultan el seguimiento de su funcionamiento.
- Se considera a continuación, de nuevo, el AFN del ejemplo 1 frente a la entrada 00101 .
- La tarea es más fácil si se mantiene un árbol en el que cada rama no determinista figure de forma explícita.



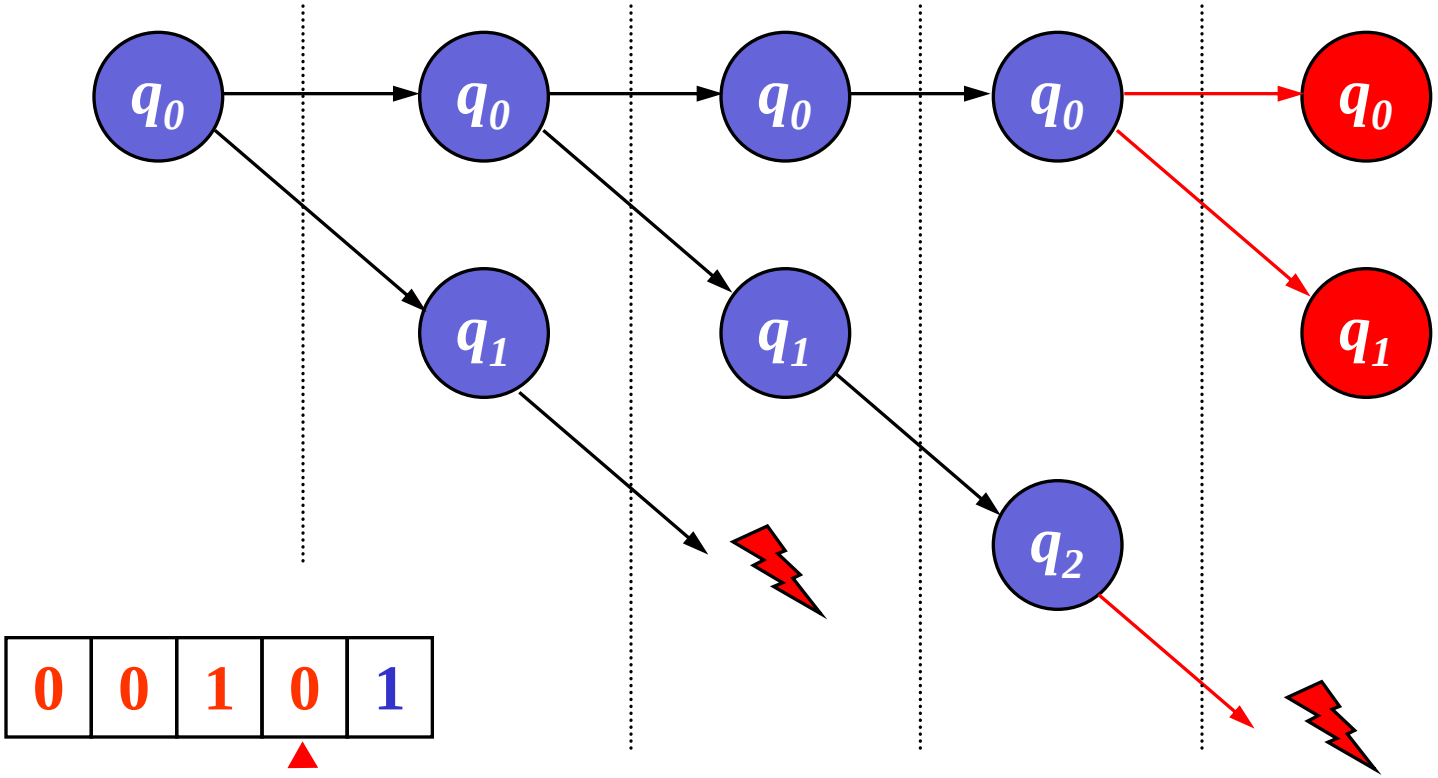
Extensión de δ a palabras sin transiciones λ: introducción

- El ejemplo anterior muestra que las ramas no deterministas en la ejecución de un autómata finito no determinista dificultan el seguimiento de su funcionamiento.
- Se considera a continuación, de nuevo, el AFN del ejemplo 1 frente a la entrada 00101 .
- La tarea es más fácil si se mantiene un árbol en el que cada rama no determinista figure de forma explícita.



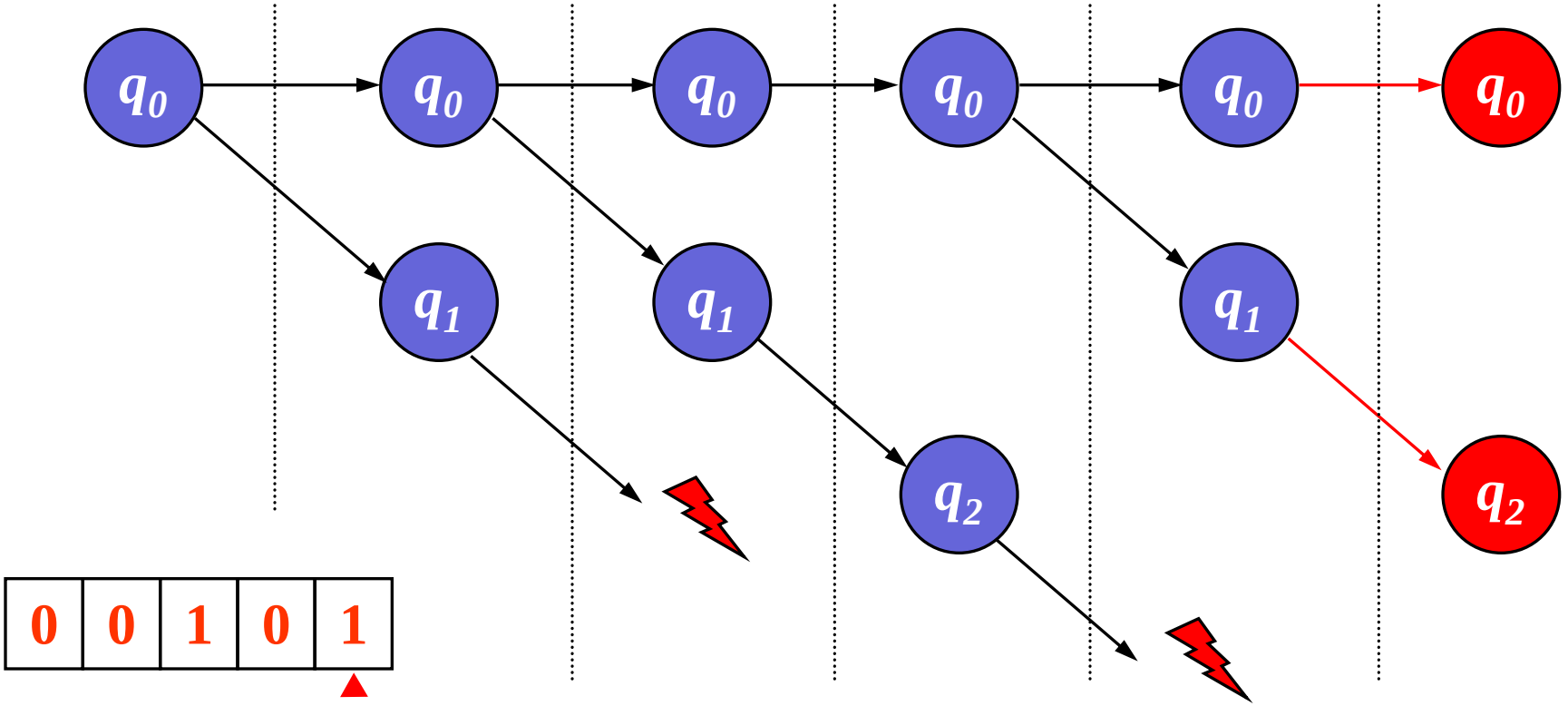
Extensión de δ a palabras sin transiciones λ: introducción

- El ejemplo anterior muestra que las ramas no deterministas en la ejecución de un autómata finito no determinista dificultan el seguimiento de su funcionamiento.
- Se considera a continuación, de nuevo, el AFN del ejemplo 1 frente a la entrada 00101 .
- La tarea es más fácil si se mantiene un árbol en el que cada rama no determinista figure de forma explícita.



Extensión de δ a palabras sin transiciones λ: introducción

- El ejemplo anterior muestra que las ramas no deterministas en la ejecución de un autómata finito no determinista dificultan el seguimiento de su funcionamiento.
- Se considera a continuación, de nuevo, el AFN del ejemplo 1 frente a la entrada 00101 .
- La tarea es más fácil si se mantiene un árbol en el que cada rama no determinista figure de forma explícita.



Extensión de δ a palabras en AFN sin transiciones λ

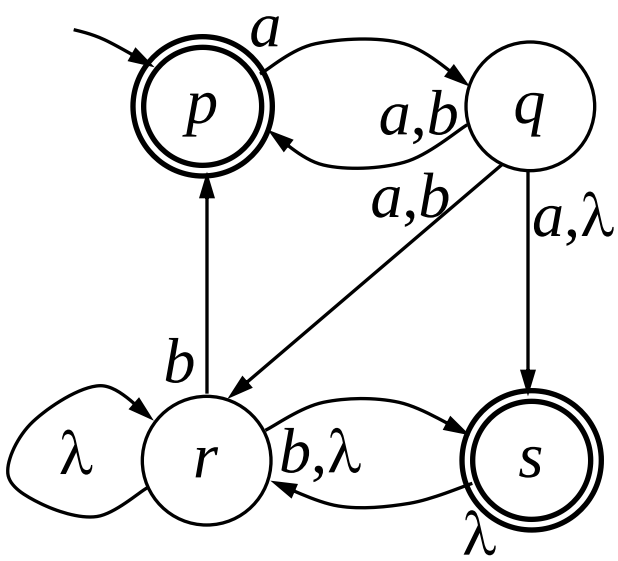
- La extensión de la función de transición δ a palabras describe los **estados a los que el AFN puede llegar a partir de un estado determinado tras procesar una cadena de entrada**.
- La única diferencia con el caso determinista consiste en que la función δ de partida, devuelve un conjunto de estados que **hay que acumular en cada paso de tratamiento de la cadena de entrada.**

Extensión de δ a palabras, caso general: introducción

- Considérese el AFN del ejemplo 2:

$$A=(Q=\{p,q,r,s\},\ \Sigma=\{a,b\},\ \delta,\ p,\ F=\{p,s\})$$

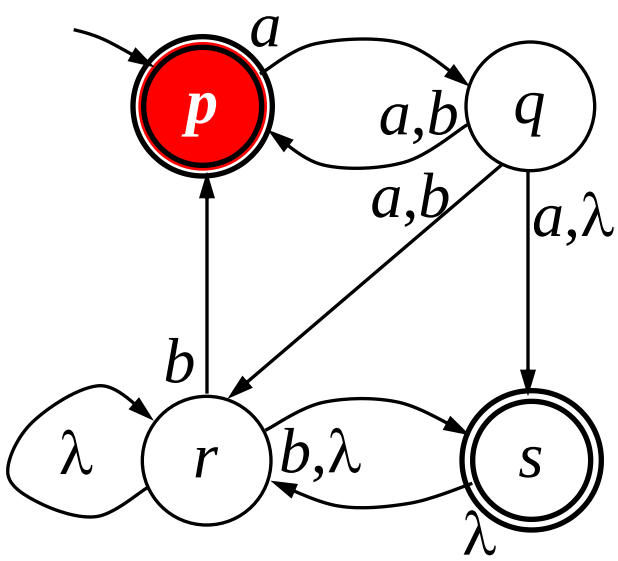
  - Donde el diagrama de transiciones de δ es el siguiente:



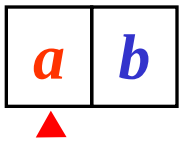
- ¿Dónde transita el autómata si recibe la entrada ab?

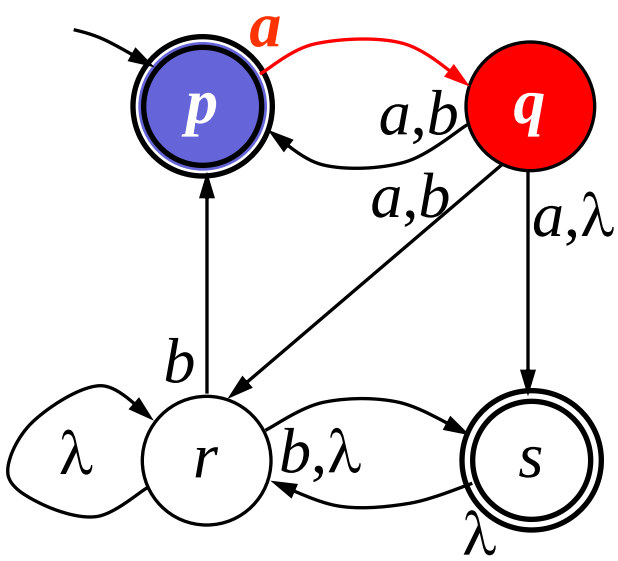
Extensión de δ a palabras, caso general: introducción

a	b



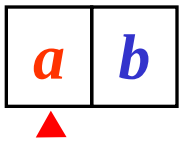
Extensión de δ a palabras, caso general: introducción

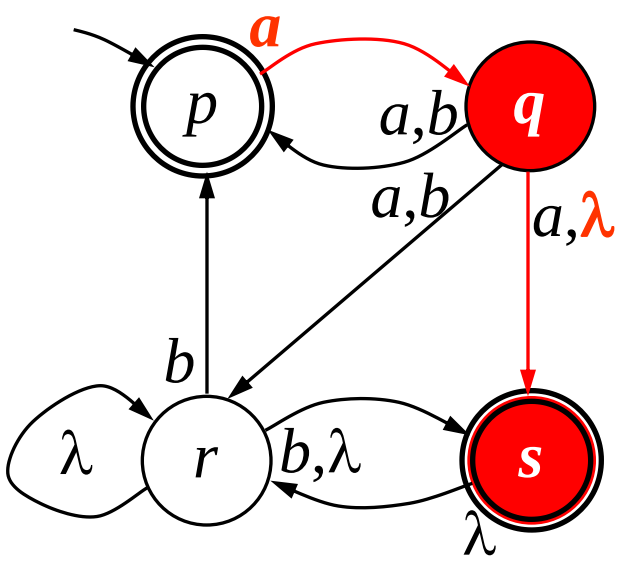




Extensión de δ a palabras, caso general: introducción

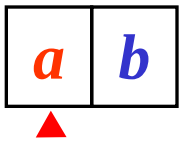
- Pero la transiciones λ permiten seguir transitando sin consumir símbolos de entrada

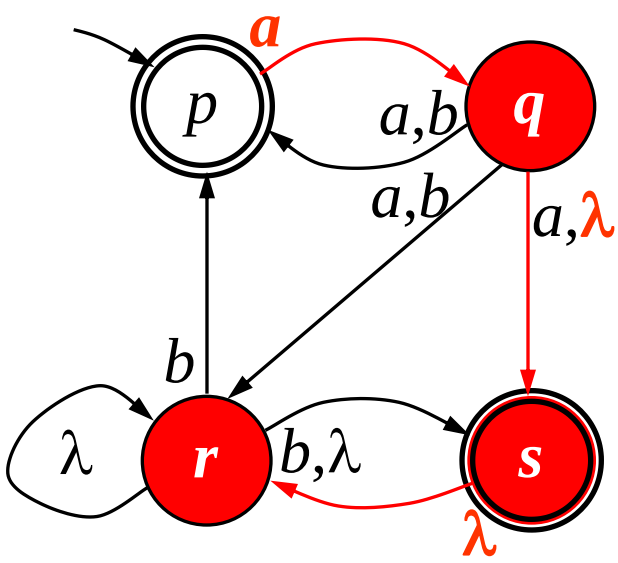




Extensión de δ a palabras, caso general: introducción

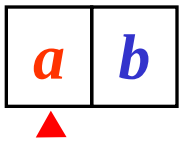
- Pero la transiciones λ permiten seguir transitando sin consumir símbolos de entrada

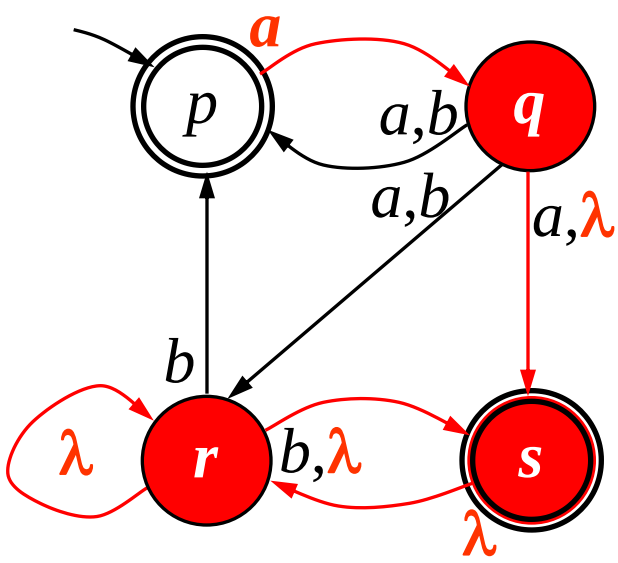




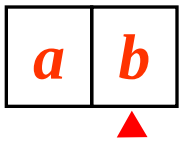
Extensión de δ a palabras, caso general: introducción

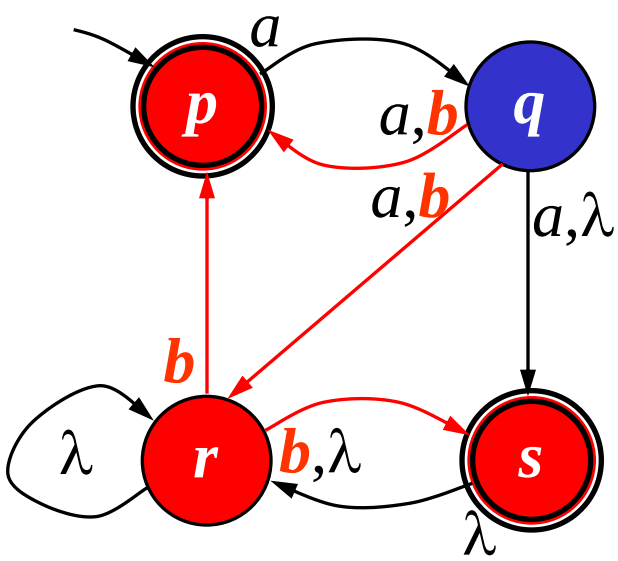
- Pero la transiciones λ permiten seguir transitando sin consumir símbolos de entrada





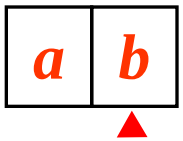
Extensión de δ a palabras, caso general: introducción

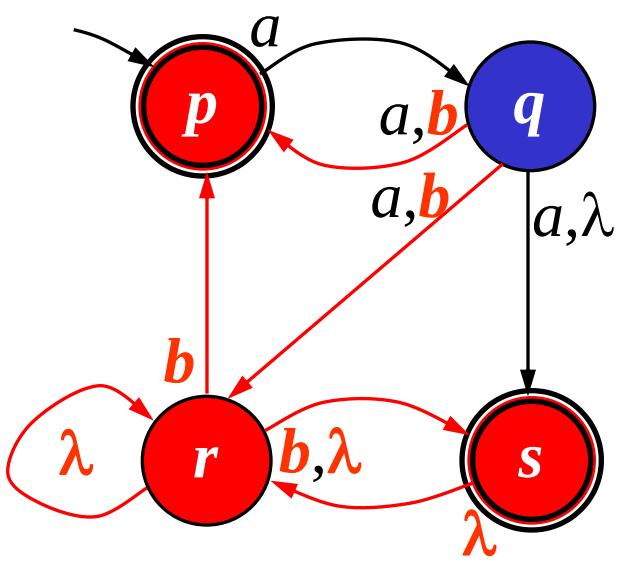




Extensión de δ a palabras, caso general: introducción

- Desde esta situación, las transiciones λ no modifican el conjunto de estados al que se llega





- Este ejemplo sugiere que la extensión de δ se defina con cuidado para el caso en el que existan transiciones λ, ya que éstas hacen que el autómata acceda adicionalmente a más estados (todos a los que se puede llegar con λ, es decir, a los que se puede transitar sin consumir símbolos de entrada).

Lenguaje de un autómata finito no determinista

Definición

- Dado un autómata finito no determinista

$$A=(Q, \Sigma, \delta, q_0, F)$$

- Se llama **lenguaje del autómata** A, $L(A)$ o **lenguaje aceptado por** A al que se define de la siguiente manera:

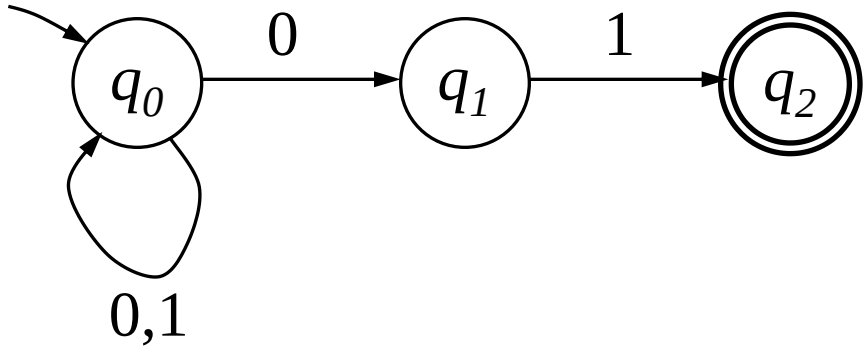
$$L(A) = \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) \cap F \neq \Phi\}$$

Lenguaje aceptado por un AFN: ejemplo 1

- Considérese al AFN del ejemplo 1:

$$A=(Q=\{q_0, q_1, q_2\}, \Sigma=\{0,1\}, \delta, q_0, F=\{q_2\})$$

  - Donde el diagrama de transiciones de δ es el siguiente:



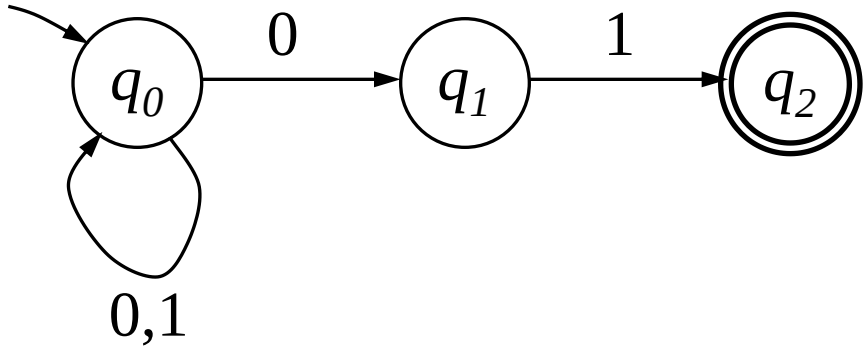
	0	1	λ
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$	Φ
q_1	Φ	$\{*q_2\}$	Φ
$*q_2$	Φ	Φ	Φ

Lenguaje aceptado por un AFN: ejemplo 1

- Considérese al AFN del ejemplo 1:

$$A=(Q=\{q_0, q_1, q_2\}, \Sigma=\{0,1\}, \delta, q_0, F=\{q_2\})$$

  - Donde δ es la siguiente:



  - Tiene como lenguaje el conjunto de los números binarios que terminan con la cadena "01"

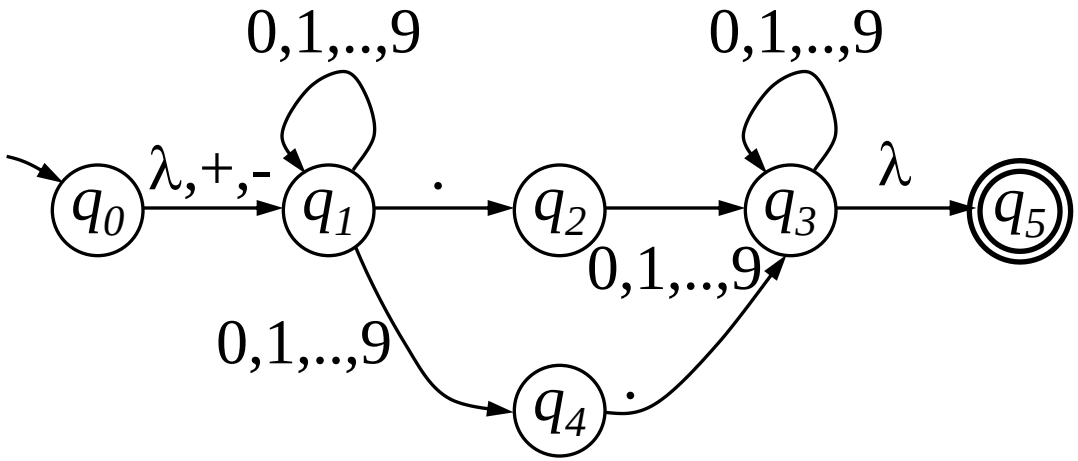
$$L(A)=\{w.01 \mid w \in \{0,1\}^*\}$$

Lenguaje aceptado por un AFN: ejemplo 2

- El autómata finito no determinista del ejemplo 3:

$$A=(Q=\{q_0, q_1, q_2, q_3, q_4, q_5\}, \Sigma=\{0,1,...,9,+,-,.\}, \delta, q_0, F=\{q_5\})$$

  - Donde δ es la siguiente:



  - Tiene como lenguaje

$L(A)$={"Números reales con signo (+,-) opcional y en notación entero.entero en los que es opcional o la parte entera o la fraccionaria"}

Bibliografía

[AlfonsecaMoriyon07] Enrique Alfonseca Cubero, Manuel Alfonseca Moreno, Roberto Moriyón Solomón. *Teoría de Autómatas y Lenguajes Formales* McGrawHill

[HopcroftUllman03] Hopcroft, J. E.; Motwani, R.; Ullman, J. D.; *Introduction to Automata Theory, Languages and Computation 2nd edition* Pearson Education.