Taller de Haskell

Paradigmas de Lenguajes de Programación

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

4 de Febrero de 2019

Entrando en calor...

Se cuenta con la siguiente representación de conjuntos, caracterizados por su función de pertenencia:

```
type Conj a = (a->Bool)
```

De este modo, si `conj1` es un conjunto y `e` un elemento, la expresión `conj1 e` devuelve `True` si `e` pertenece a `conj1`, y `False` en caso contrario.

Entrando en calor...

Se cuenta con la siguiente representación de conjuntos, caracterizados por su función de pertenencia:

```
type Conj a = (a->Bool)
```

De este modo, si `conj1` es un conjunto y `e` un elemento, la expresión `conj1 e` devuelve `True` si `e` pertenece a `conj1`, y `False` en caso contrario.

Operaciones sobre conjuntos

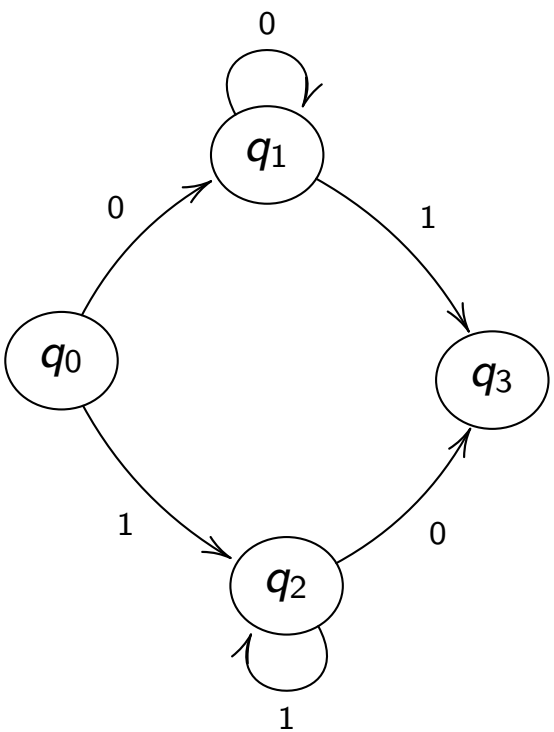
- Definir y dar el tipo de las siguientes funciones:
  - vacío
  - unión

Máquinas de Estado

Máquina de estado

Una máquina de estado se define como una 3-tupla (Q, Σ, δ) donde:

- Q: es un conjunto de estados;
- Σ es un alfabeto;
- $\delta :: Q \times \Sigma \to Q$



- $Q = \{q_0, q_1, q_2, q_3\}$;
- $\Sigma = \{0, 1\}$;
- δ :
  - $\delta\ q_0\ 0 = q_1$
  - $\delta\ q_0\ 1 = q_2$
  - $\delta\ q_1\ 0 = q_1$
  - $\delta\ q_1\ 1 = q_3$

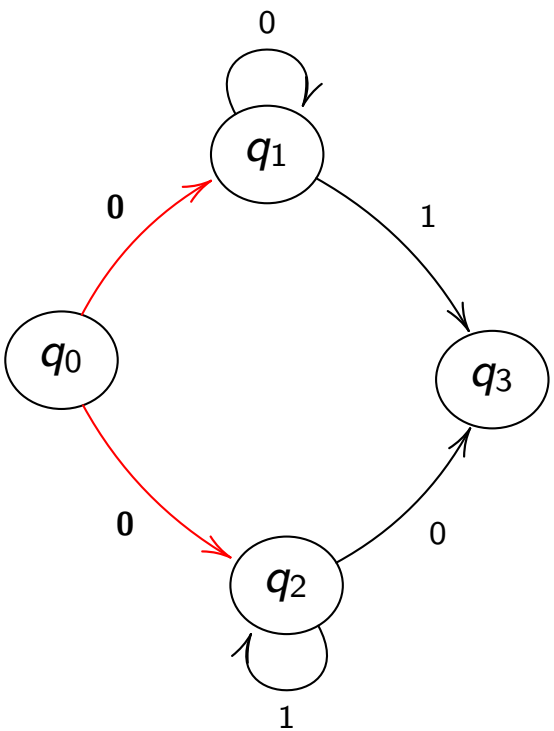
    ⋮

MEN

Máquina de estado No determinística

Una máquina de estado **no** determinística se define como una 3-tupla (Q, Σ, δ) donde:

- Q: es un conjunto de estados;
- Σ es un alfabeto;
- $\delta :: Q \times \Sigma \to \mathcal{P}(Q)$



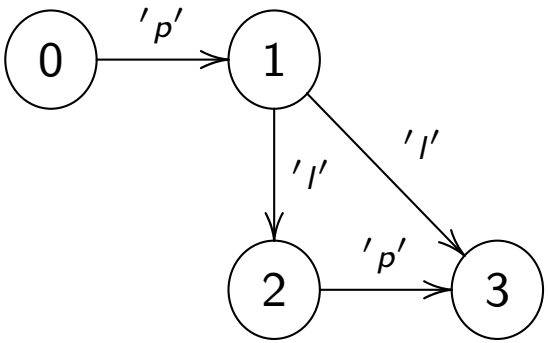
- $Q = \{q_0, q_1, q_2, q_3\}$;
- $\Sigma = \{0, 1\}$;
- δ :
  - $\delta\ q_0\ 0 = \{q_1, q_2\}$
  - $\vdots$

MEN en la práctica...

```
data MEN a b = AM {sigma :: [b], delta :: (a -> b -> [a])}
```

MEN en la práctica...

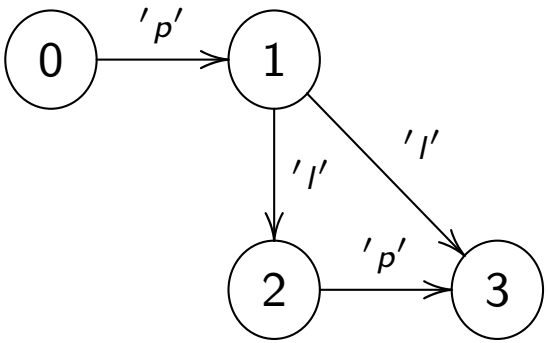
```
data MEN a b = AM {sigma :: [b], delta :: (a -> b -> [a])}
```




```
mPLP :: MEN Int Char
mPLP = AM ['l','p'] tran
 where tran q s | q == 0 && s == 'p' = [1]
                | q == 1 && s == 'l' = [2, 3]
                | q == 2 && s == 'p' = [3]
                | otherwise = []
```

MEN en la práctica...

```
data MEN a b = AM {sigma :: [b], delta :: (a -> b -> [a])}
```




```
mPLP :: MEN Int Char
mPLP = AM ['l','p'] tran
 where tran q s | q == 0 && s == 'p' = [1]
                | q == 1 && s == 'l' = [2, 3]
                | q == 2 && s == 'p' = [3]
                | otherwise = []
```

```
sigma mPLP ⇝ ['l', 'p']

delta mPLP 0 'p' ⇝ [1]

mPLP {sigma ⇝ ['a']}
```

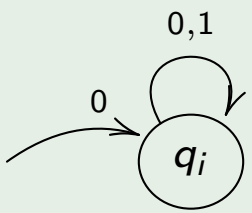
Ejercicios

Ejercicio #1

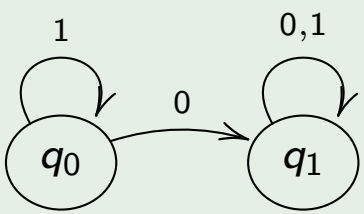
Definición de funciones para hacerse amigx del tipo de dato MEN.

Ejercicio #2

- **Estado trampa**: cuando no tiene transiciones que lo lleven a otros estados más que a sí mismo.



- **Completo**: todos los estados del autómata tienen transiciones por todos los símbolos, es decir, la función de transición es no nula para todas las combinaciones de estados y símbolos del alfabeto.



Ejercicios

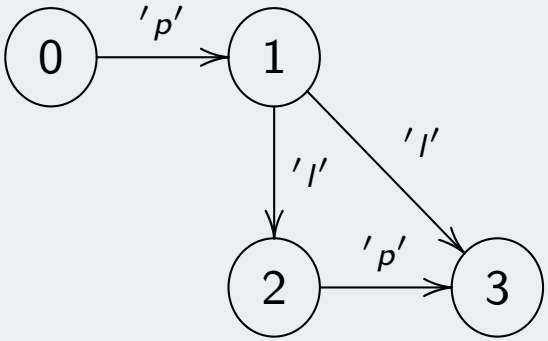
Ejercicio #3

Consumir: retorna los estados en los que se puede encontrar el autómata después de haber consumido la cadena.

Ejercicios

Ejercicio #3

Consumir: retorna los estados en los que se puede encontrar el autómata después de haber consumido la cadena.



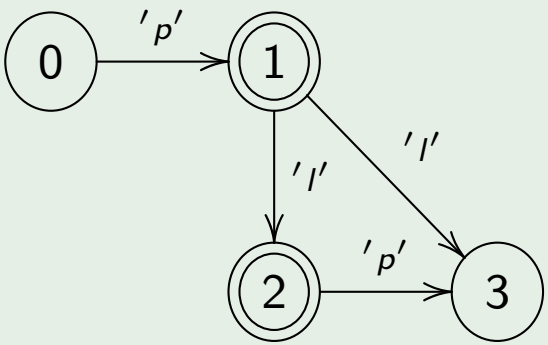

```
consumir mPLP 0 "pl'' ⇝ [2,3]

consumir mPLP 0 "plp" ⇝ [3]

consumir mPLP 1 "lp" ⇝ [3]
```

Ejercicio #4

Lenguaje: el conjunto de las palabras de símbolos del alfabeto que son aceptadas por el autómata.

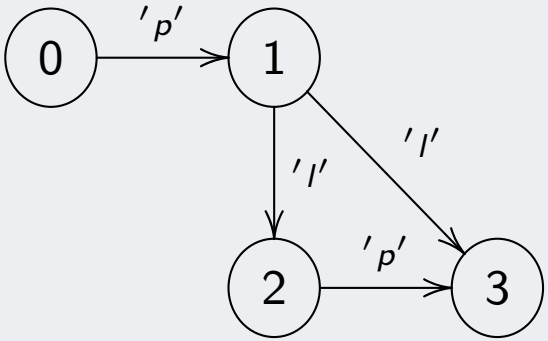



```
lenguaje mPLP 0 [1,2] ⇝ ["p","pl"]
```

Ejercicios

Ejercicio #5

Trazas: todas las cadenas de símbolos por las que una máquina de estados puede llegar desde un estado un estado q a cualquier otro estado, a partir de transiciones no nulas.




```
mPLP 0 ⇝ ["p", "pl", "pl", "plp"]
```

Ejercicio #6

Alcanzables: lista con todos los estados alcanzables desde un estado cualquiera.

```
mPLP 0 ⇝ [1,2,3]
mPLP 1 ⇝ [2,3]
mPLP 2 ⇝ [2,3]
mPLP 3 ⇝ []
```

Agenda

Taller Parte 1

- Resolución de ejercicios 1, 2.
- Armado de test con los que chequearon el funcionamiento de sus resoluciones.
- Deseable: ejercicio 3 resuelto, o al menos, tenerlo pensado.

Agenda

Taller Parte 1

- Resolución de ejercicios 1, 2.
- Armado de test con los que chequearon el funcionamiento de sus resoluciones.
- Deseable: ejercicio 3 resuelto, o al menos, tenerlo pensado.

Taller Parte 2

- Cierre del taller, ie, ejercicios 3,4,5 y 6 resueltos

Agenda

Taller Parte 1

- Resolución de ejercicios 1, 2.
- Armado de test con los que chequearon el funcionamiento de sus resoluciones.
- Deseable: ejercicio 3 resuelto, o al menos, tenerlo pensado.

Taller Parte 2

- Cierre del taller, ie, ejercicios 3,4,5 y 6 resueltos

Importante: leer sección 3 del enunciado (pautas de evaluación y entrega).