Verificación de Programas Concurrentes (2)

Ingeniería del Software 2
1er Cuatrimestre 2020

Objetivo

- Dado:
  - Un LTS M representando un programa concurrente
  - Una fórmula LTL P que queremos ver que cumple el programa concurrente
- Definir un algoritmo que retorna **true** si todas las trazas de M satisfacen P

Algoritmo (Alto Nivel)

- Entrada: LTL P, LTS M

1. Convertir la fórmula LTL ¬P a un autómata $A_{\neg P}$ que caracteriza todas las trazas que satisfacen ¬P

2. Chequear que si las trazas de M son disjuntas con las trazas de $A_{\neg P}$

3. Si la intersección es vacía, retornar True, caso contrario devolver traza como contra-ejemplo.

Algoritmo (+Detalle)

- Entrada: LTL P, LTS M

1. Convertir la fórmula LTL ¬P a un autómata de Büchi $A_{\neg P}$ que caracteriza todas las trazas que satisfacen ¬P

2. Convertir el LTS M a un autómata de Büchi A_M que caracteriza todas las trazas que contiene M

3. Hacer el producto de los autómatas de Büchi $A_{\neg P}$ y A_M

4. Si $L(A_{\neg P} \times A_M) \neq \emptyset$, entonces existe una traza de M que no cumple (i.e. la propiedad P no se cumple en el sistema M)

Algoritmo (+Detalle)

- Entrada: LTL P, LTS M

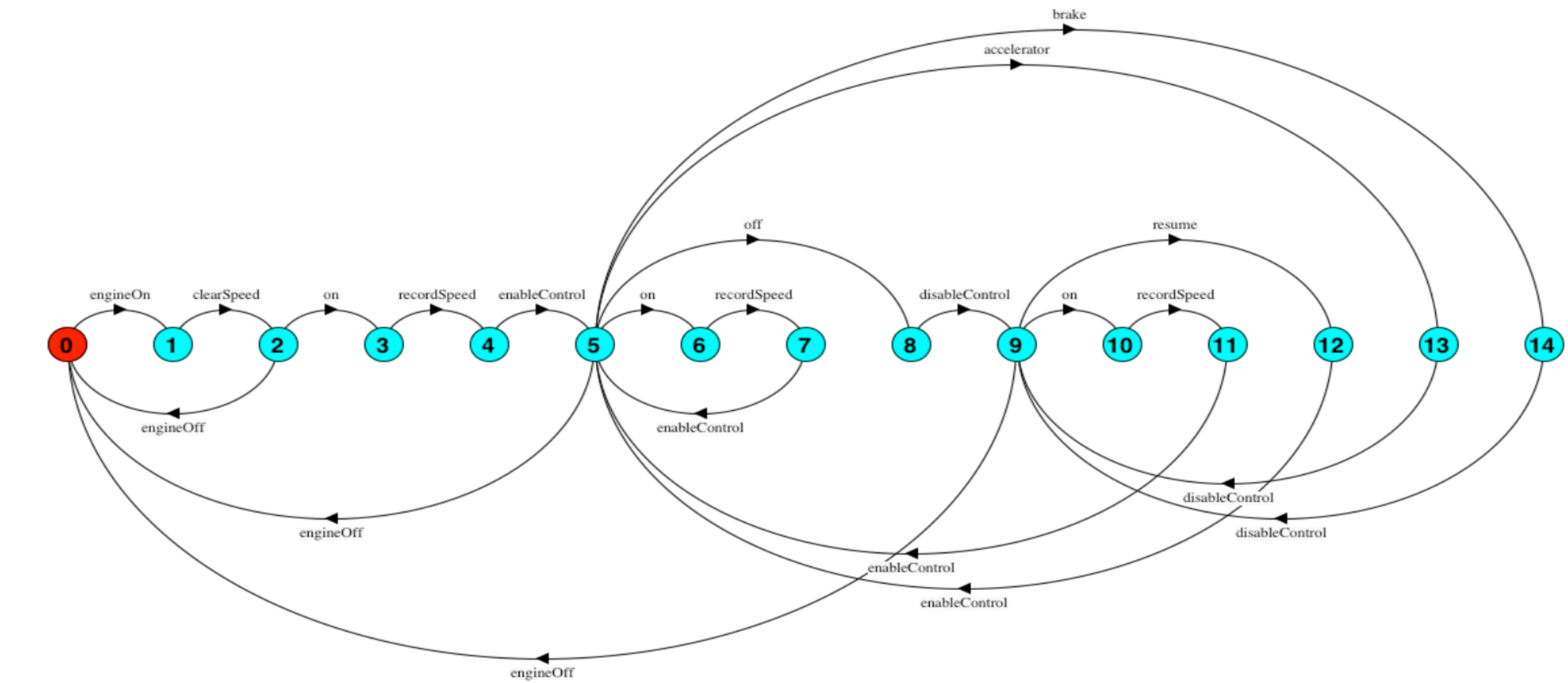
1. Convertir la fórmula LTL ¬P a un autómata de Büchi $A_{\neg P}$ que caracteriza todas las trazas que satisfacen ¬P

2. Convertir el LTS M a un autómata de Büchi A_M que caracteriza todas las trazas que contiene M

3. Hacer el producto de los autómatas de Büchi $A_{\neg P}$ y A_M

4. Si $L(A_{\neg P} \times A_M) \neq \emptyset$, entonces existe una traza de M que no cumple (i.e. la propiedad P no se cumple en el sistema M)

LTS: Labeled Transition System



LTS2Büchi

- El LTS es la descripción del comportamiento del programa concurrente
- Necesitamos transformarlo en un autómata de Büchi que represente las mismas trazas

LTS2Büchi

Definición. *(LTS)* Sea $Estados$ el universo de estados, Act el universo de acciones observables, y $Act_\tau = Act \cup \{\tau\}$. Un LTS es una tupla $P = (S, A, \Delta, s_0)$, donde $S \subseteq Estados$ es un conjunto finito, $A \subseteq Act_\tau$ es un conjunto de etiquetas, $\Delta \subseteq (S \times A \times S)$ es un conjunto de transiciones etiquetadas, y $s_0 \in S$ es el estado inicial. Definimos el alphabeto de comunicacion de P como $\alpha P = A \setminus \{\tau\}$.

LTS2Büchi

- Sea un LTS P=$\langle S, A, \Delta, s_0 \rangle$, podemos construir un autómata de Büchi A_P =$\langle \Sigma, Q, \Delta', Q_0, F \rangle$ donde:
  - Σ = Act (conjunto de acciones observables)
  - Q = S
  - $\Delta' = \Delta$ es la relación de transición
  - $Q_0 = \{s_0\}$ es el único estado inicial
  - $F = S$ son todos estados de aceptación

Algoritmo (+Detalle)

- Entrada: LTL P, LTS M

1. Convertir la fórmula LTL $\neg P$ a un autómata de Büchi $A_{\neg P}$ que caracteriza todas las trazas que satisfacen $\neg P$

2. Convertir el LTS M a un autómata de Büchi A_M que caracteriza todas las trazas que contiene M

3. Hacer el producto de los autómatas de Büchi $A_{\neg P}$ y A_M

4. Si $L(A_{\neg P} \times A_M) \neq \emptyset$, entonces existe una traza de M que no cumple (i.e. la propiedad P no se cumple en el sistema M)

Producto de Autómatas de Büchi

- Sean $A_1 = \langle \Sigma, Q_1, \Delta_1, Q_{01}, F_1 \rangle$ y $A_2 = \langle \Sigma, Q_2, \Delta_2, Q_{02}, F_2 \rangle$ dos autómatas de Büchi, el producto $A_1 \times A_2$ es un autómata de Büchi generalizado $A_1 \times A_2 = \langle \Sigma, Q, \Delta, Q_0, F \rangle$ que se define del siguiente modo:

  - $Q = Q_1 \times Q_2$
  - $Q_0 = Q_{01} \times Q_{02}$
  - $F = \{F_1 \times Q_2, Q_1 \times F_2\}$
  - ((q_1,q_2),a,(q_1',q_2')) ∈Δ sii (q_1,a,q_1')∈Δ_1 y (q_2,a,q_2') ∈ Δ_2

Lema

- Sean $A_1 = \langle \Sigma, Q_1, \Delta_1, Q_{01}, F_1 \rangle$ y $A_2 = \langle \Sigma, Q_2, \Delta_2, Q_{02}, F_2 \rangle$ dos autómatas de Büchi, $A_1 \times A_2 = \langle \Sigma, Q, \Delta, Q_0, F \rangle$ el producto de A_1 y A_2,

  - Entonces $L(A_1 \times A_2) = L(A_1) \cap L(A_2)$

Algoritmo (+Detalle)

- Entrada: LTL P, LTS M

1. Convertir la fórmula LTL ¬P a un autómata de Büchi $A_{\neg P}$ que caracteriza todas las trazas que satisfacen ¬P

2. Convertir el LTS M a un autómata de Büchi A_M que caracteriza todas las trazas que contiene M

3. Hacer el producto de los autómatas de Büchi $A_{\neg P}$ y A_M

4. Si $L(A_{\neg P} \times A_M) \neq \emptyset$, entonces existe una traza de M que no cumple (i.e. la propiedad P no se cumple en el sistema M)

Chequeo de Vacuidad de Lenguaje

- Problema: Dado un Büchi verificar si el lenguaje que acepta es vacío.
- Un Büchi acepta una palabra cuando existe una ejecución que visita un estado de aceptación infinitas veces
- Algoritmo:

1. Buscar un ciclo en que contiene un estado de aceptación y es alcanzable desde el estado inicial.
    - Buscar un componente fuertemente conexo, alcanzable, que contenga un estado de aceptación. ✔
2. Si no existe, el lenguaje que acepta el autómata es vacío.

> Existen dos algoritmos conocidos con complejidad $o(n+m)$ para calcular componentes fuertementes conexas. Uno de ellos es el algoritmo de Kosaraju y el otro es el algoritmo de Tarjan. El algoritmo de Kosaraju es un poco más simple de implementar y veremos sólo este algoritmo.

Visto en AED3

Algoritmo (+Detalle)

- Entrada: LTL P, LTS M

1. Convertir la fórmula LTL ¬P a un autómata de Büchi $A_{\neg P}$ que caracteriza todas las trazas que satisfacen ¬P

2. Convertir el LTS M a un autómata de Büchi A_M que caracteriza todas las trazas que contiene M

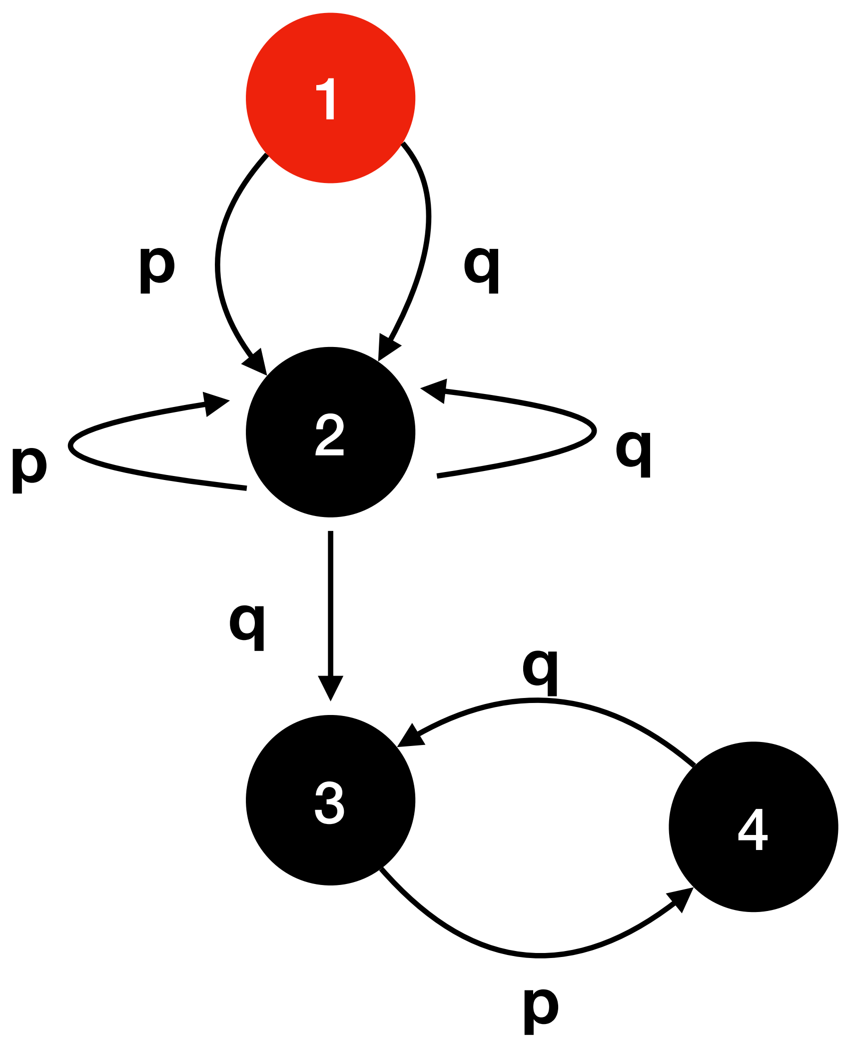
3. Hacer el producto de los autómatas de Büchi $A_{\neg P}$ y A_M

4. Si $L(A_{\neg P} \times A_M) \neq \emptyset$, entonces existe una traza de M que no cumple (i.e. la propiedad P no se cumple en el sistema M)

Un Ejemplo

Input

El Sistema
(LTS)



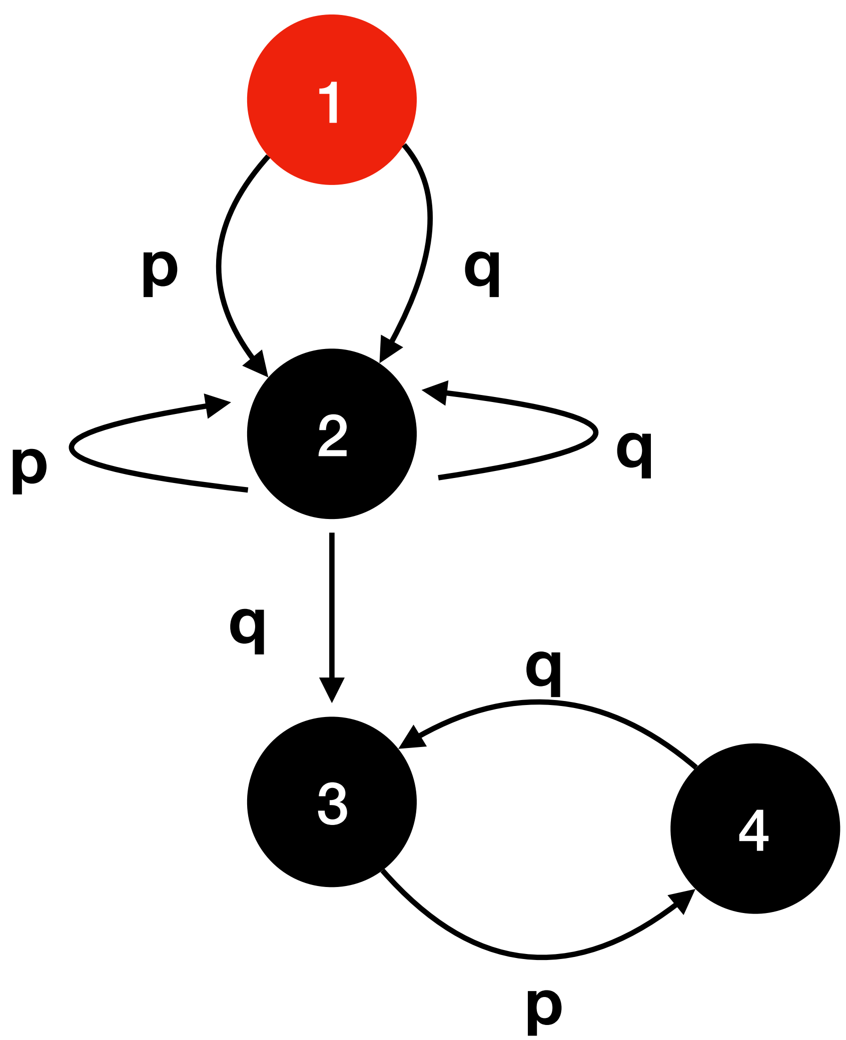
La Propiedad
(LTL)

$\Box q$

Paso 1

1.Convertir la fórmula LTL ¬P a un autómata de Büchi $A_{\neg P}$ que caracteriza todas las trazas que satisfacen ¬P

El Sistema (LTS)



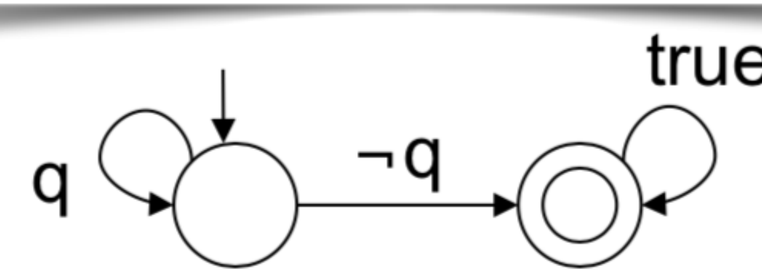
La Propiedad (LTL)

[]q

La negación de la propiedad

![]q = <>!q

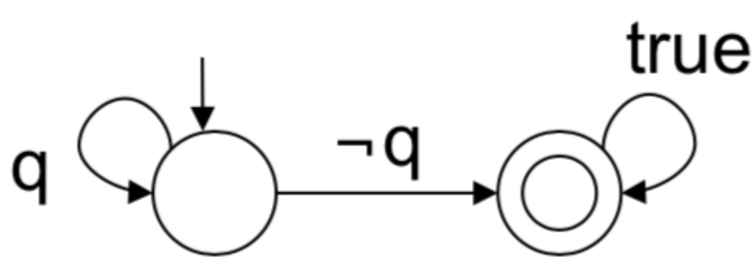
El Büchi de la negación



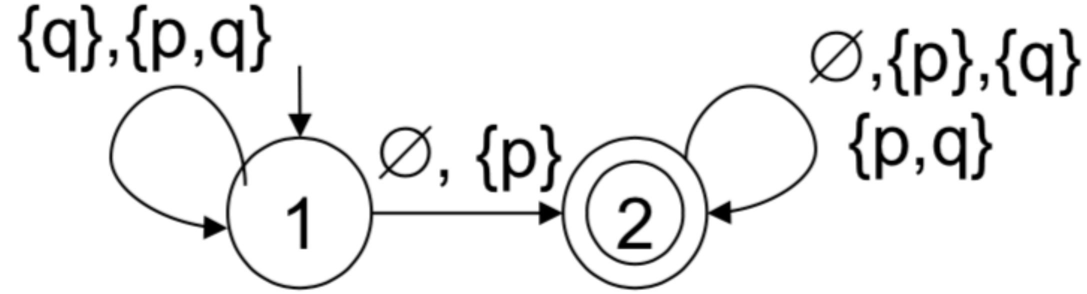
Paso 1

1. Convertir la fórmula LTL ¬P a un autómata de Büchi $A_{\neg P}$ que caracteriza todas las trazas que satisfacen ¬P

El Büchi de la negación



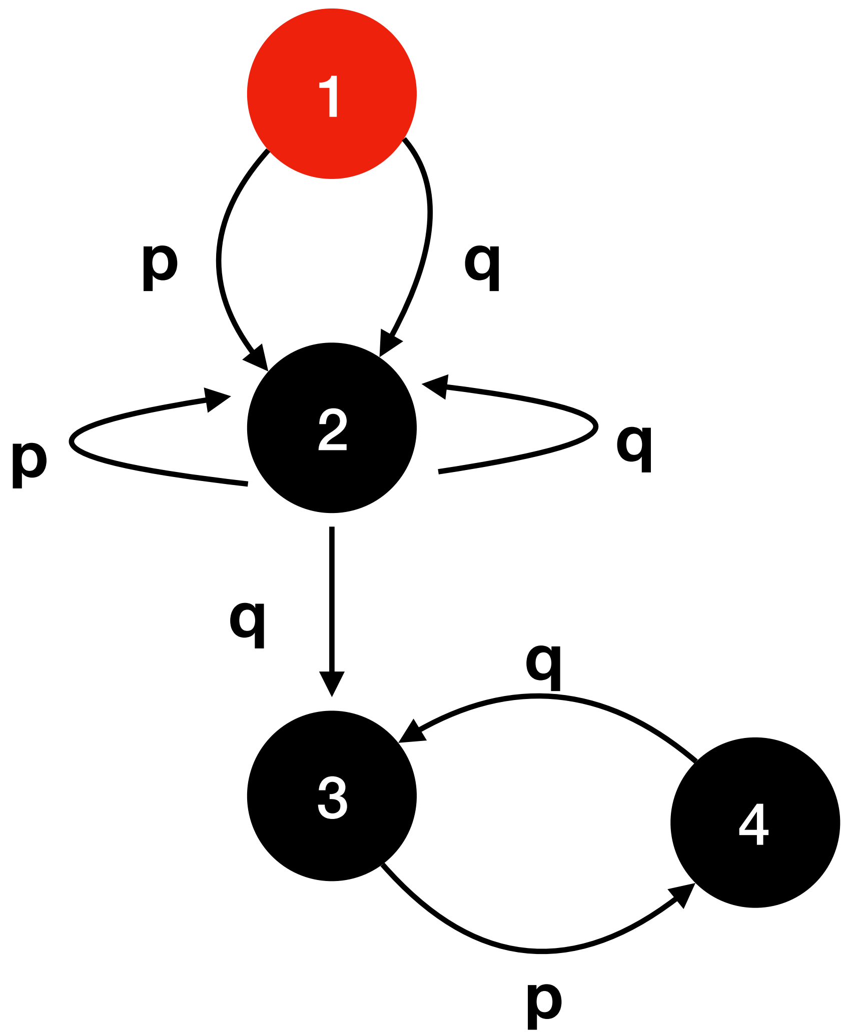
Convertimos cada fórmula proposicional (ej. "q") en las valuaciones que lo satisfacen (ej. {q}, {pq})

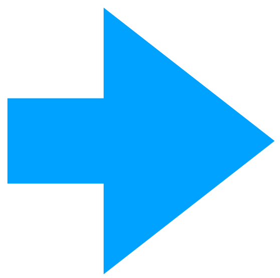


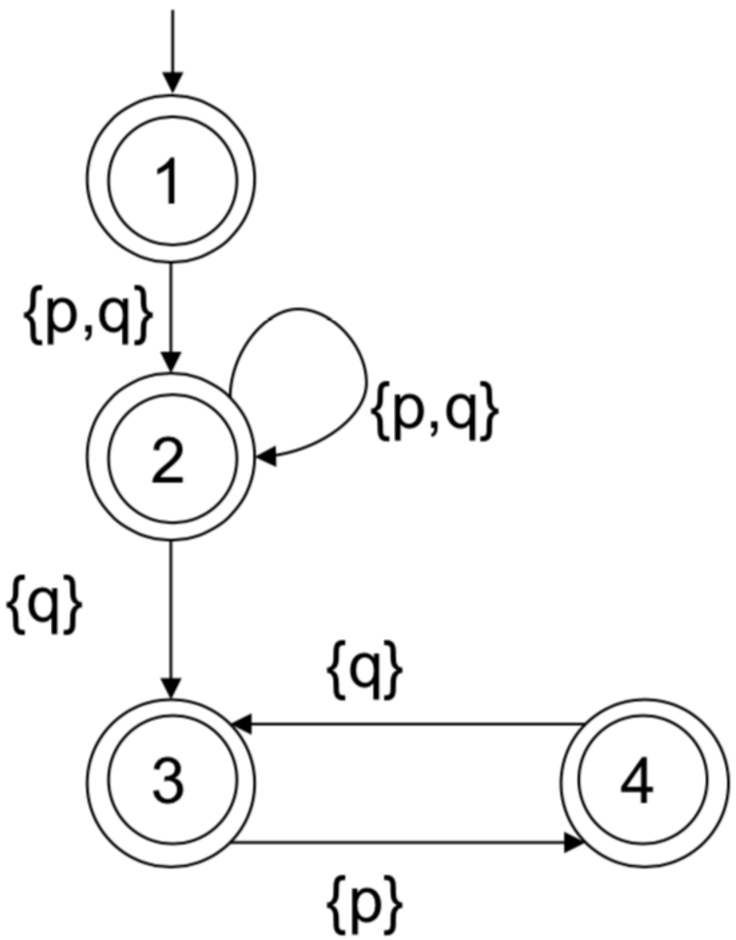
Paso 2

2.Convertir el LTS M a un autómata de Büchi A_M que caracteriza todas las trazas que contiene M

El Sistema
(LTS)





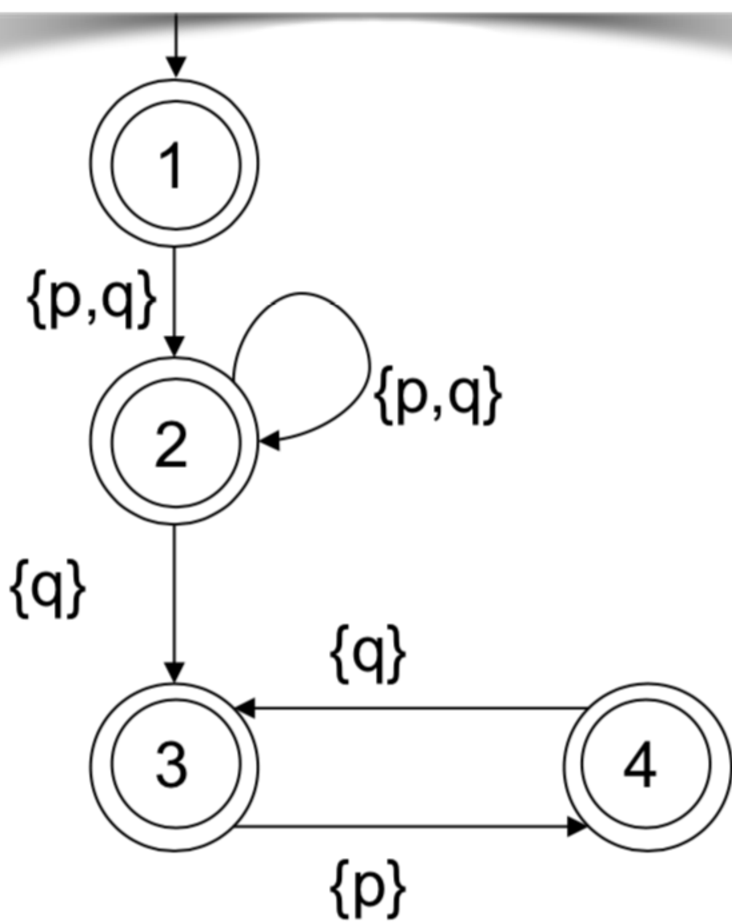


Notar que todos los estados son de aceptación

Paso 3

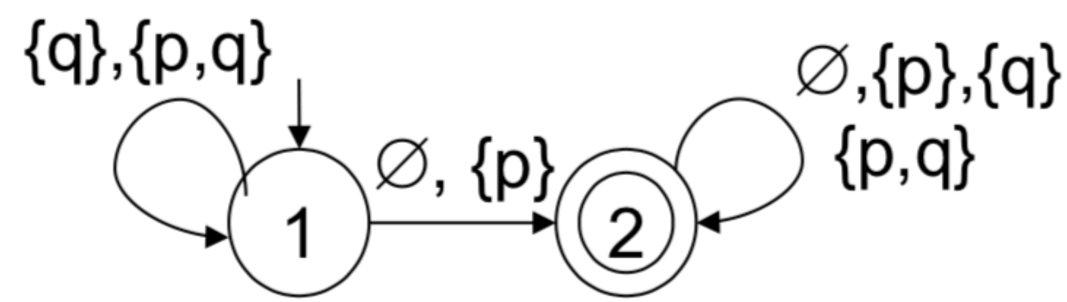
3.Hacer el producto de los autómatas de Büchi $A_{\neg P}$ y A_M

El Sistema
(Büchi)



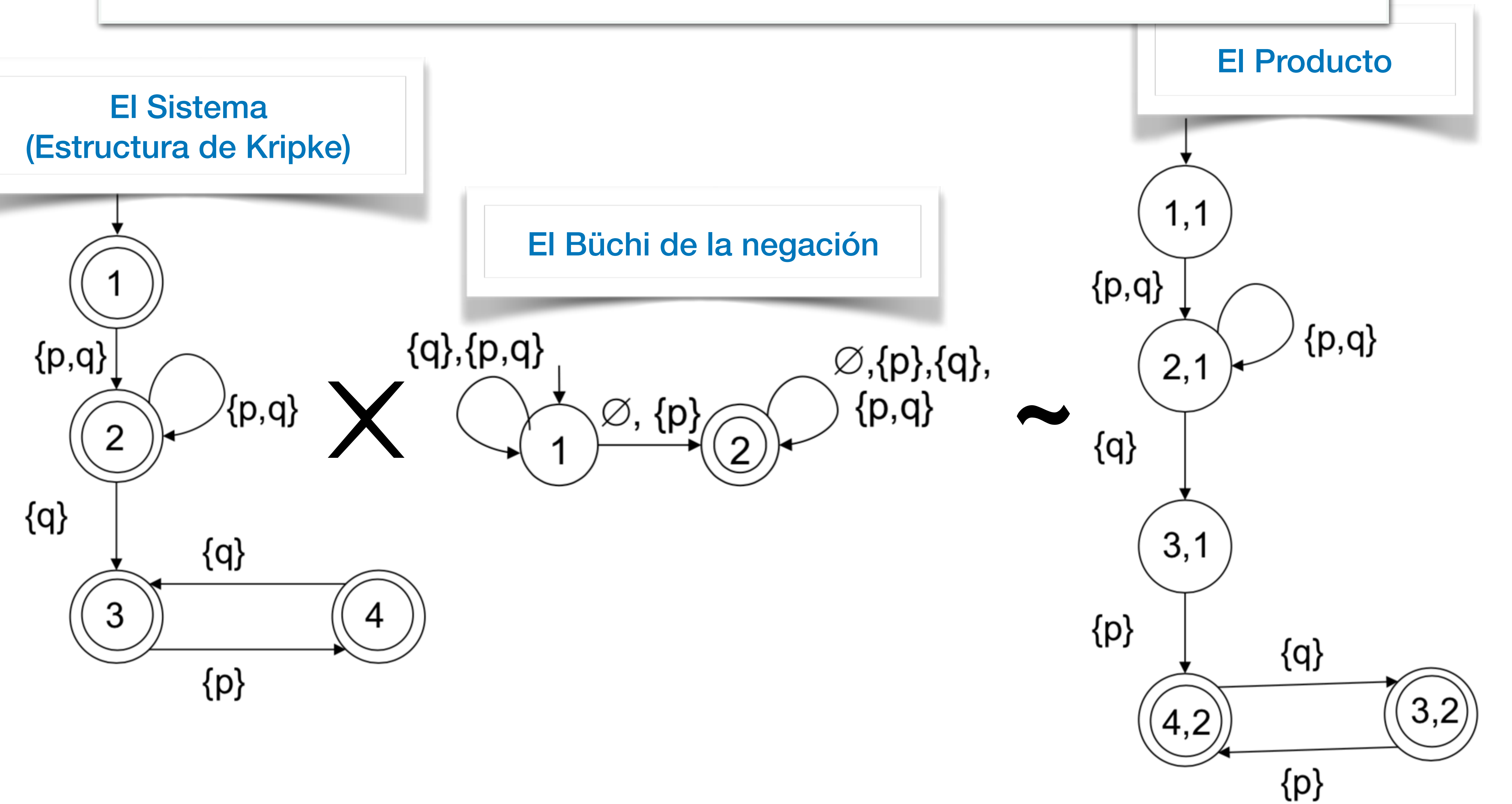
X

El Büchi de la negación



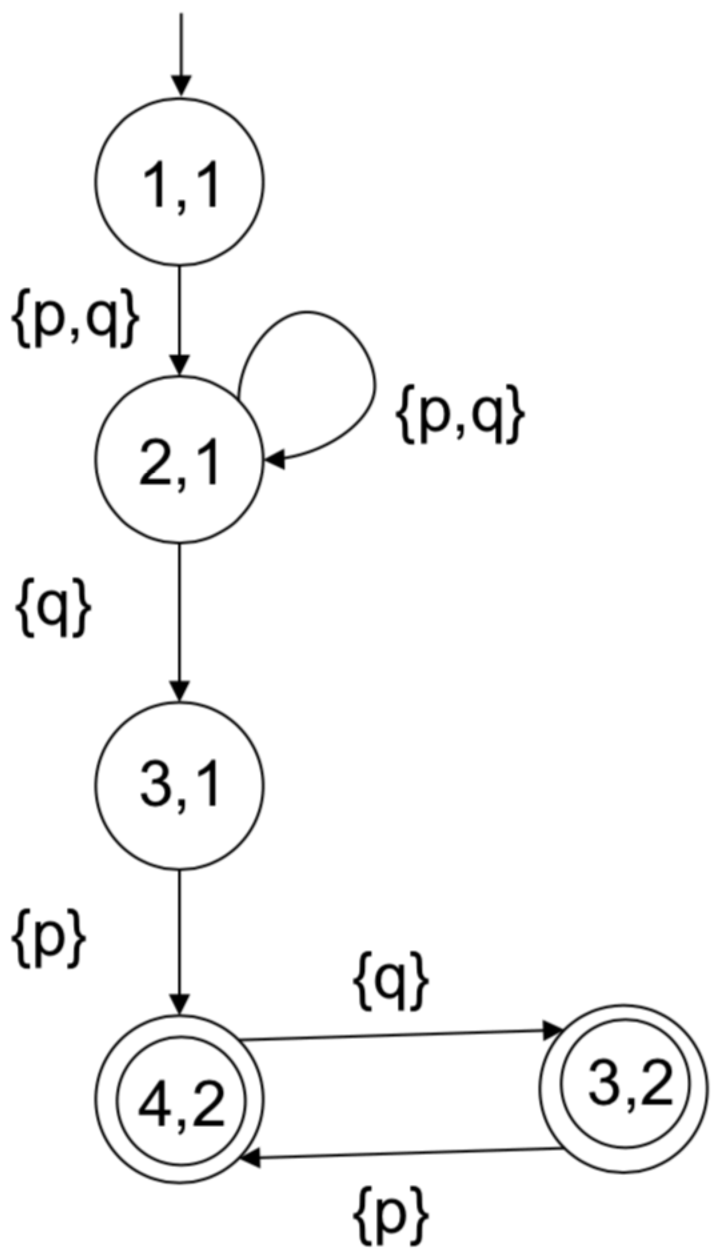
Paso 3

3.Hacer el producto de los autómatas de Büchi $A_{\neg P}$ y A_M



Paso 4

4.Si $L(A_{\neg P} \times A_M) \neq \emptyset$, entonces existe una traza de M que no cumple (i.e. la propiedad P no se cumple en el sistema M)



Componente fuertemente conexa con estado de aceptación: {(4,2), (3,2)}

$\Downarrow$

El lenguaje es no vacío.
La palabra {p,q} {q} {p} ({q} {p})$^{\omega}$ es aceptada por el camino (1,1)(2,1)(3,1)((4,2)(3,2))

$\Downarrow$

Existe una palabra aceptada por el sistema y también por la negación de la propiedad

$\Downarrow$

El sistema viola la propiedad []q

Resumen

- Dado un sistema concurrente LTS M y una propiedad LTL P, existe un algoritmo que automáticamente nos dice si la propiedad P se cumple o no en M

- De esta forma, podemos verificar automáticamente propiedades sobre los sistemas concurrentes que construimos, asegurando la ausencia de defectos.