Análisis de Refinamientos entre Sistemas de Transiciones Modales basado en SAT

Carolina Inés Dania
Director: Dr. Nazareno Aguirre

Fa.M.A.F

16 de diciembre de 2008

Introducción

- Queremos crear, diseñar y mantener software de calidad y de gran escala
- Queremos proveer garantía del correcto funcionamiento
- Tenemos metodologías de desarrollo basadas en matemática y lógica
- Los LTS constituyen un formalismo para la descripción operacional de sistemas

Introducción

- Queremos crear, diseñar y mantener software de calidad y de gran escala
- Queremos proveer garantía del correcto funcionamiento
- Tenemos metodologías de desarrollo basadas en matemática y lógica
- Los LTS constituyen un formalismo para la descripción operacional de sistemas
- En este trabajo nos enfocaremos:
  - en la verificación de la existencia de relaciones de refinamientos/implementaciones.
  - aprovechar mecanismos avanzados de análisis basado en SAT para acelerar el proceso de análisis
  - en analizar cierto tipo de "meta propiedades" de MTS.

Estructura

- Introducción a LTS y MTS.
  - Relaciones semánticas basadas en simulación
  - Refinamientos e implementaciones
- Automatización del análisis de estas relaciones mediante SAT solvers.
  - La herramienta Alloy
  - Modelado con Alloy
- Verificación de propiedades sobre refinamientos e implementaciones usando Alloy
  - Problemática en la cuantificación de alto orden
  - Soluciones propuestas
- Conclusiones y trabajo futuro

Sistemas de Transiciones Etiquetadas

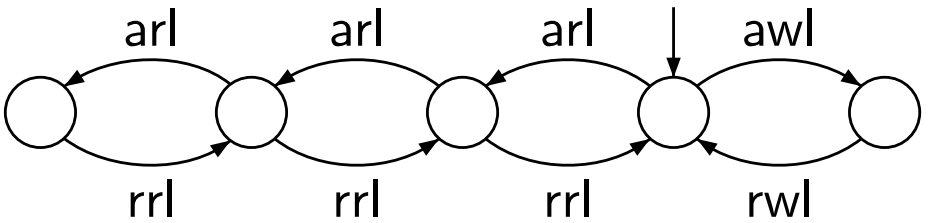
Un *sistema de transiciones etiquetadas (LTS - labelled transition systems)* es una estructura $\mathrm{P} = (S, L, \longrightarrow, s_0)$ donde:

- S es un conjunto finito de estados,
- L es un conjunto finito de etiquetas observables más una etiqueta no observable (o silenciosa) llamada τ,
- $\longrightarrow \subseteq (S \times L \times S)$ es una relación de transición entre estados, y
- $s_0 \in S$ es el estado inicial.

Sistemas de Transiciones Etiquetadas

Un *sistema de transiciones etiquetadas (LTS - labelled transition systems)* es una estructura $P = (S, L, \longrightarrow, s_0)$ donde:

- S es un conjunto finito de estados,
- L es un conjunto finito de etiquetas observables más una etiqueta no observable (o silenciosa) llamada τ,
- $\longrightarrow \subseteq (S \times L \times S)$ es una relación de transición entre estados, y
- $s_0 \in S$ es el estado inicial.



Limitaciones de los LTS

Tenemos:

- Los LTS que describen comportamiento.
- Algo ocurre o no en el sistema.
- Las descripciones son totales.

Limitaciones de los LTS

Tenemos:

- Los LTS que describen comportamiento.
- Algo ocurre o no en el sistema.
- Las descripciones son totales.

Necesitamos:

- descripciones parciales de comportamiento.
- algo ocurre, no ocurre o es factible que ocurra.

Sistemas de Transiciones Modales

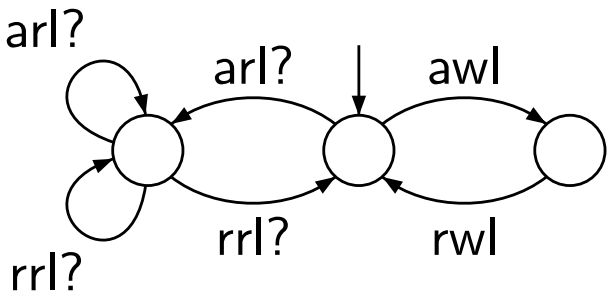
Un *sistema de transiciones modales (MTS - modal transition systems)* es una estructura $\mathrm{P} = (S, L, \longrightarrow_r, \longrightarrow_p, s_0)$ donde tenemos dos tipos de transiciones:

- transiciones requeridas $\longrightarrow_r$
- transiciones probables $\longrightarrow_p$
- satisfaciendo $\longrightarrow_r \subseteq \longrightarrow_p$.

Sistemas de Transiciones Modales

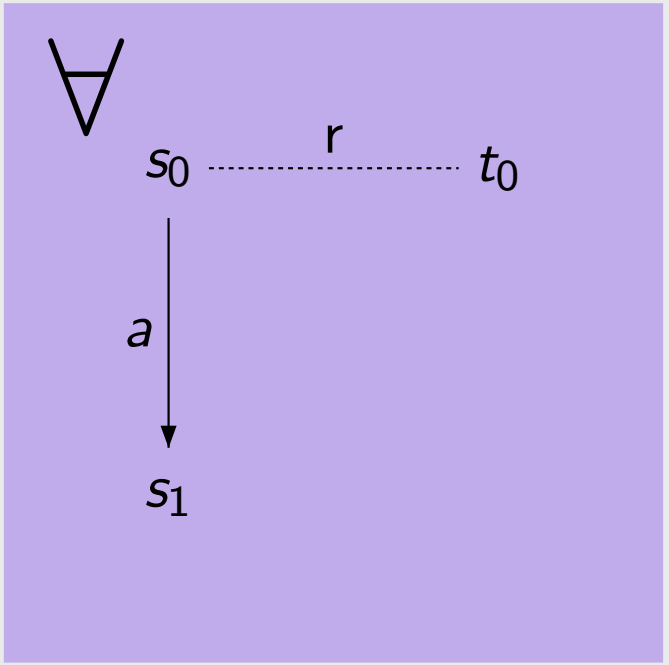
Un *sistema de transiciones modales (MTS - modal transition systems)* es una estructura $P = (S, L, \longrightarrow_r, \longrightarrow_p, s_0)$ donde tenemos dos tipos de transiciones:

- transiciones requeridas $\longrightarrow_r$
- transiciones probables $\longrightarrow_p$
- satisfaciendo $\longrightarrow_r \subseteq \longrightarrow_p$.



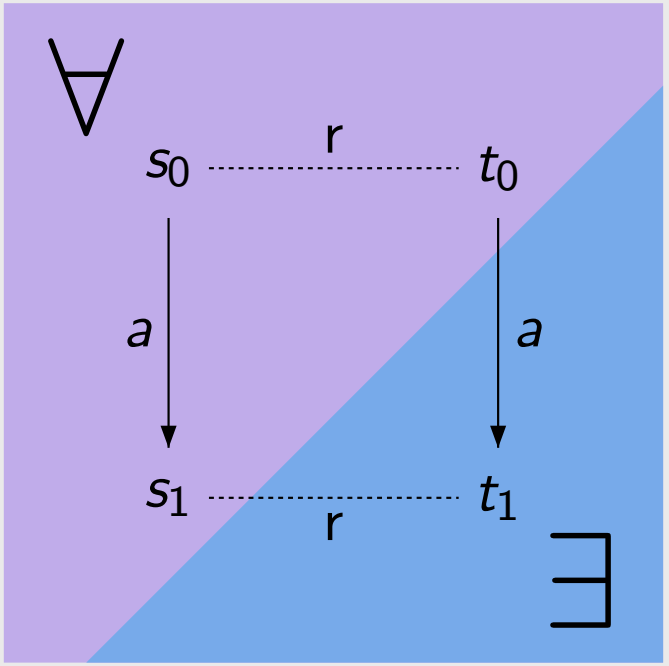
Simulación y bisimulación fuerte

Una *simulación fuerte (strong simulation)* es una relación $r \in S \times S'$ entre dos LTS, $P = (S, L, \longrightarrow, s_0)$ y $P' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in r$ y



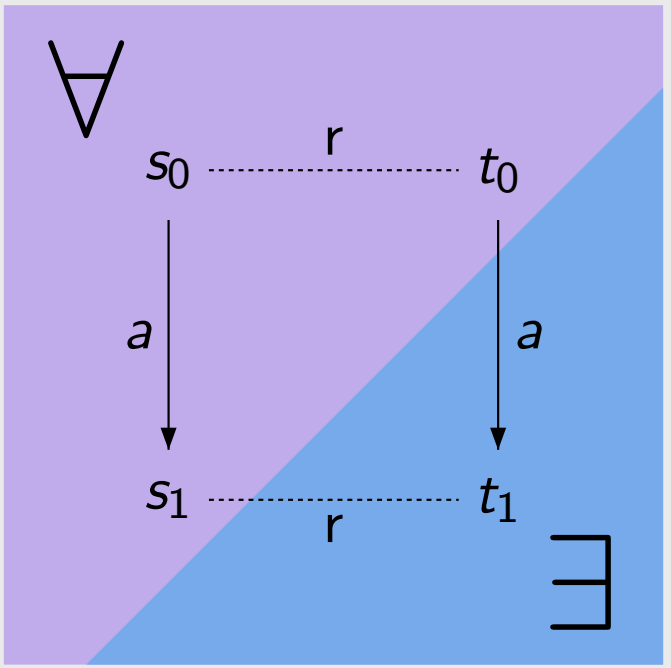
Simulación y bisimulación fuerte

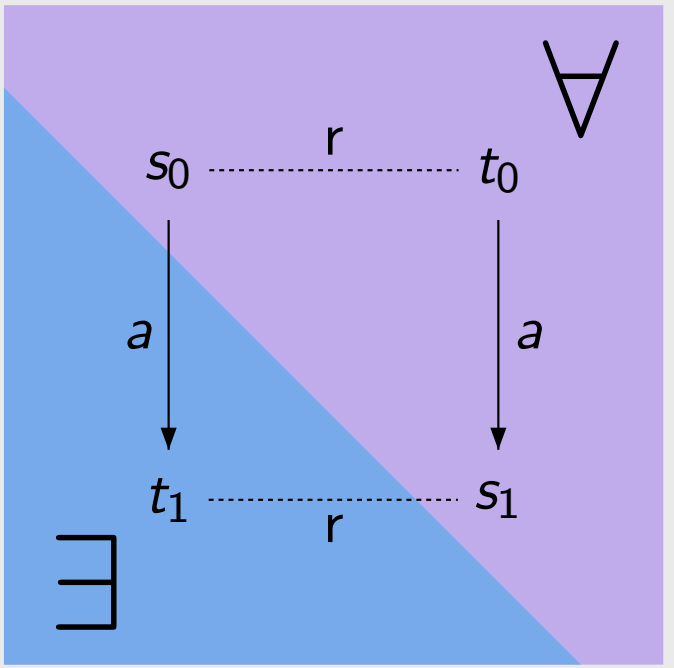
Una *simulación fuerte (strong simulation)* es una relación $r \in S \times S'$ entre dos LTS, $P = (S, L, \longrightarrow, s_0)$ y $P' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in r$ y



Simulación y bisimulación fuerte

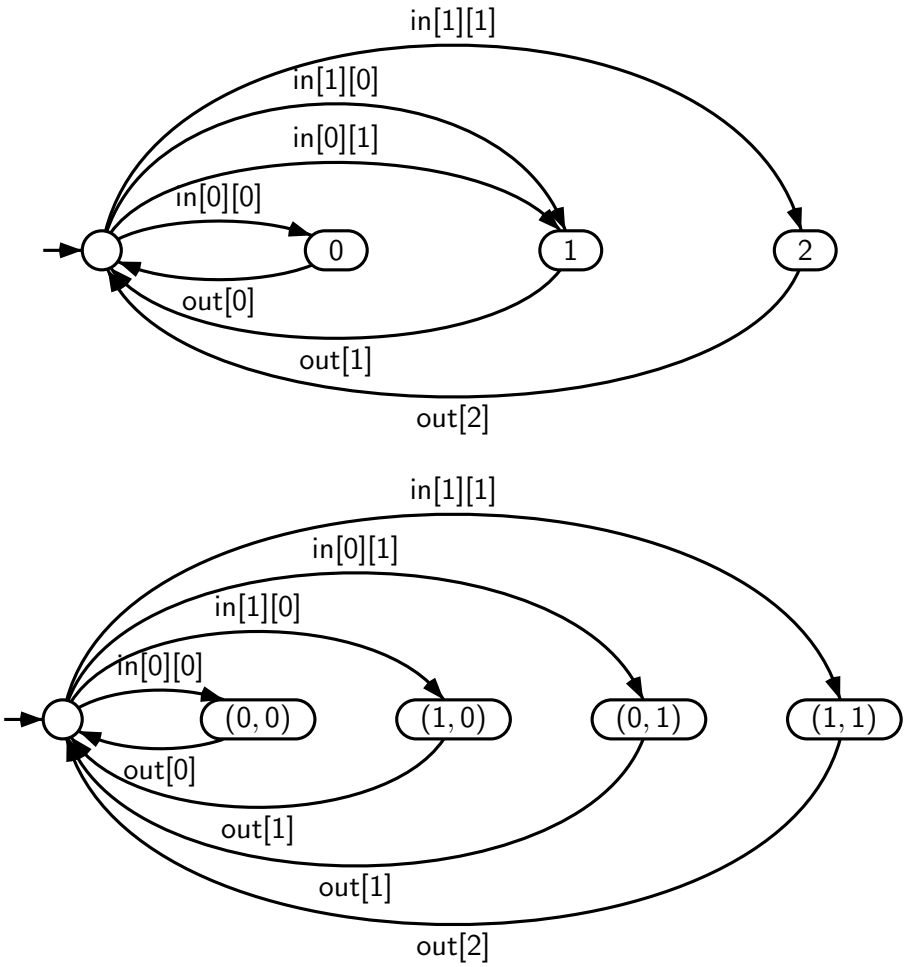
Una *simulación fuerte (strong simulation)* es una relación $r \in S \times S'$ entre dos LTS, $P = (S, L, \longrightarrow, s_0)$ y $P' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in r$ y



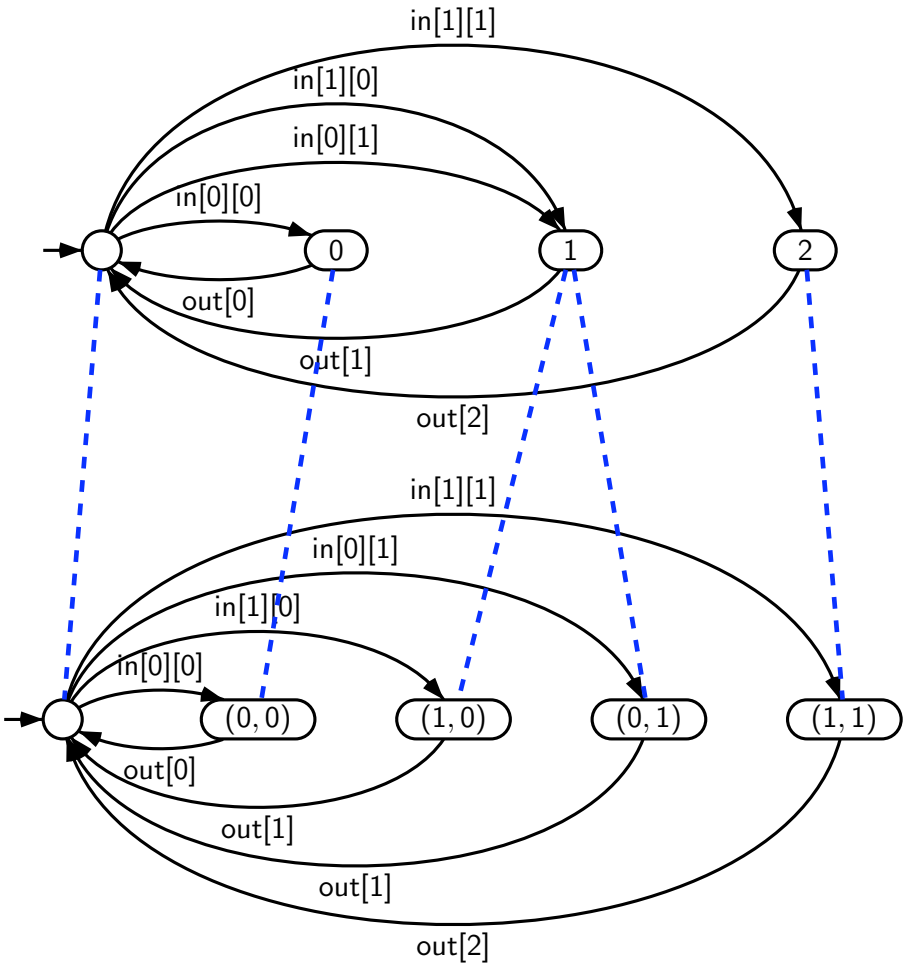


$\sim$r también es simulación

Ejemplo de bisimulación fuerte

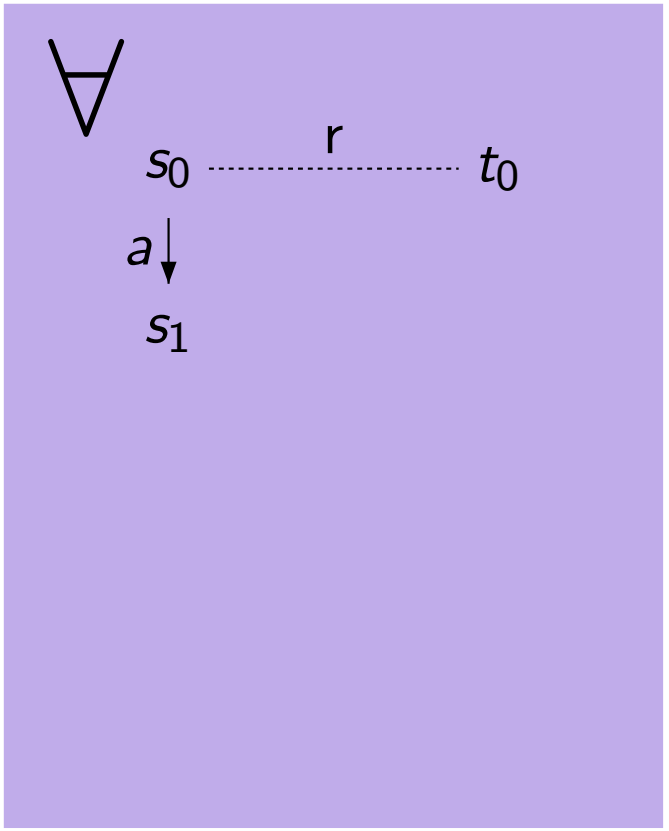


Ejemplo de bisimulación fuerte



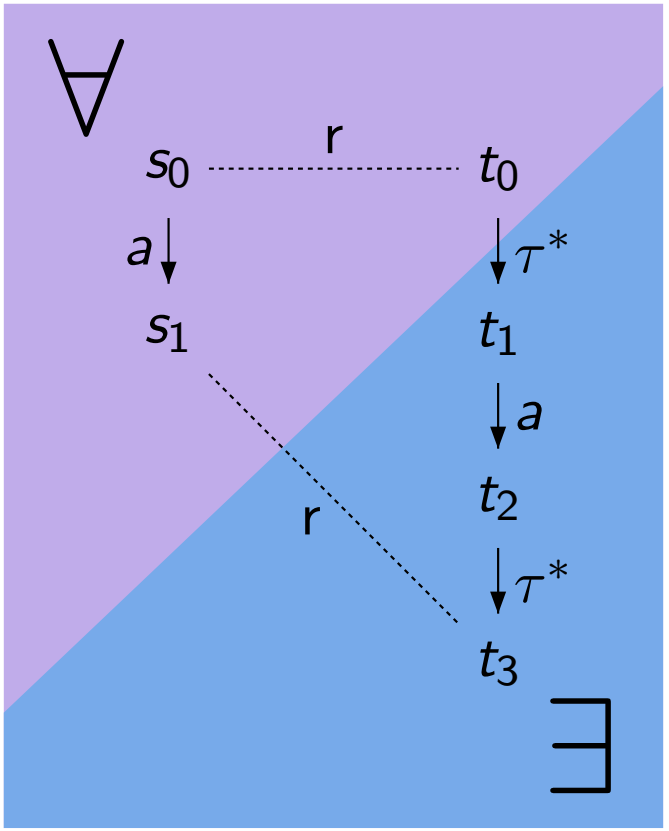
Simulación y bisimulación débil

Una *simulación débil (weak simulation)* es una relación $r \in S \times S'$ entre dos LTS, $P = (S, L, \longrightarrow, s_0)$ y $P' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in r$ y



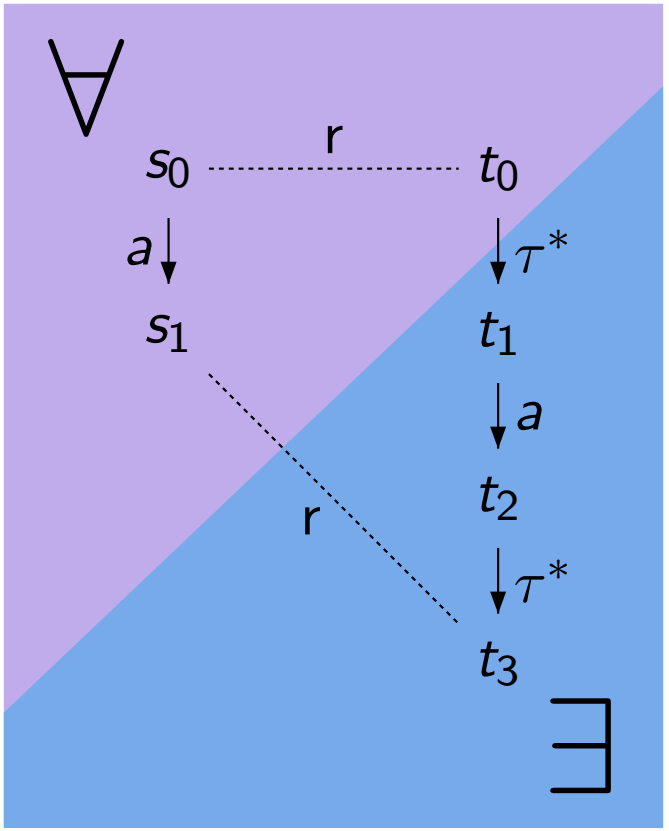
Simulación y bisimulación débil

Una *simulación débil (weak simulation)* es una relación r $\in S \times S'$ entre dos LTS, P $= (S, L, \longrightarrow, s_0)$ y P$' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in$ r y

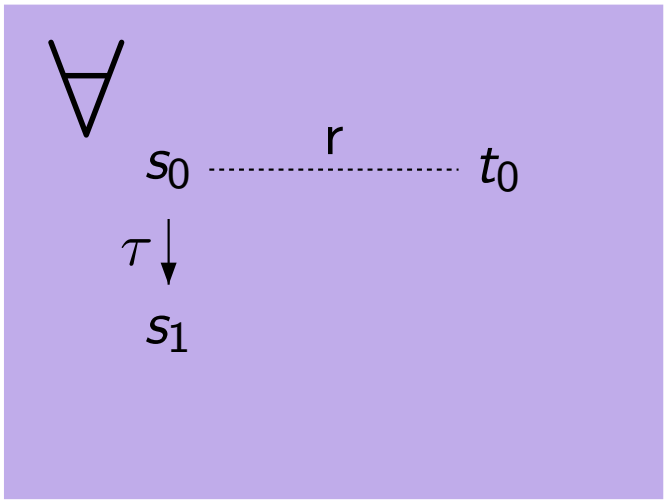


Simulación y bisimulación débil

Una *simulación débil (weak simulation)* es una relación $r \in S \times S'$ entre dos LTS, $P = (S, L, \longrightarrow, s_0)$ y $P' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in r$ y

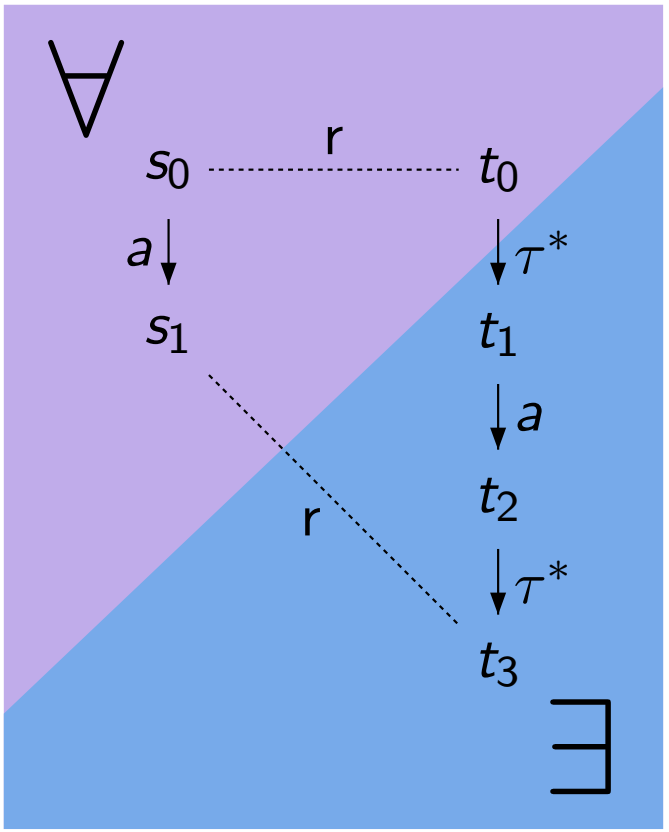


ó

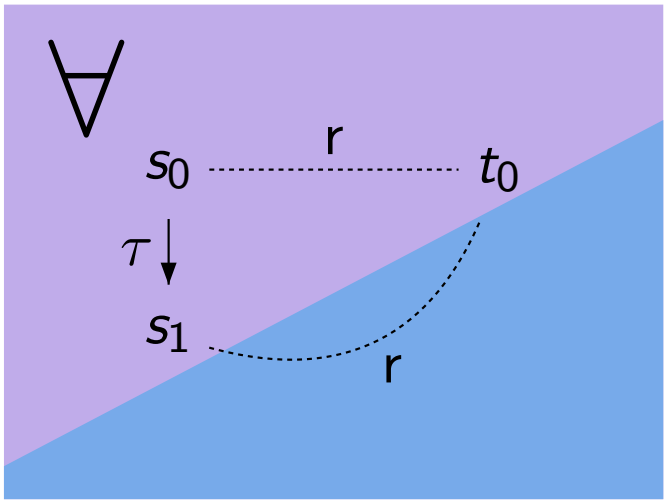


Simulación y bisimulación débil

Una *simulación débil (weak simulation)* es una relación r $\in S \times S'$ entre dos LTS, P $= (S, L, \longrightarrow, s_0)$ y P$' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in$ r y

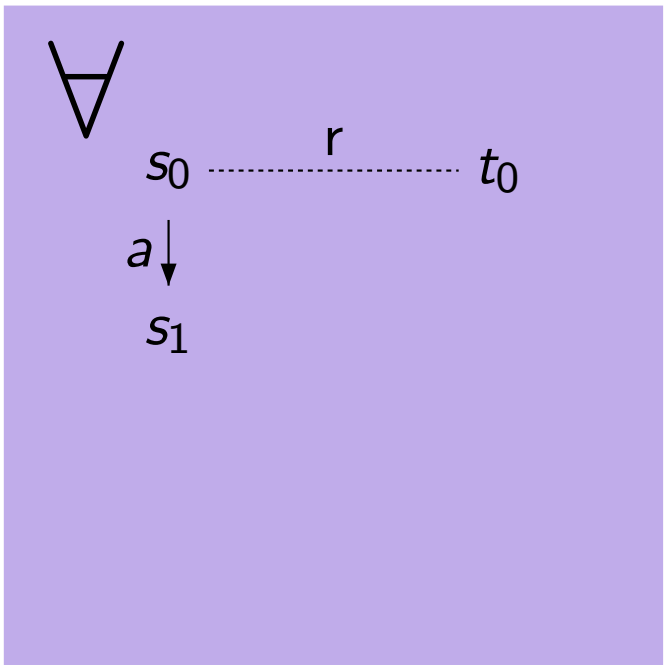


ó



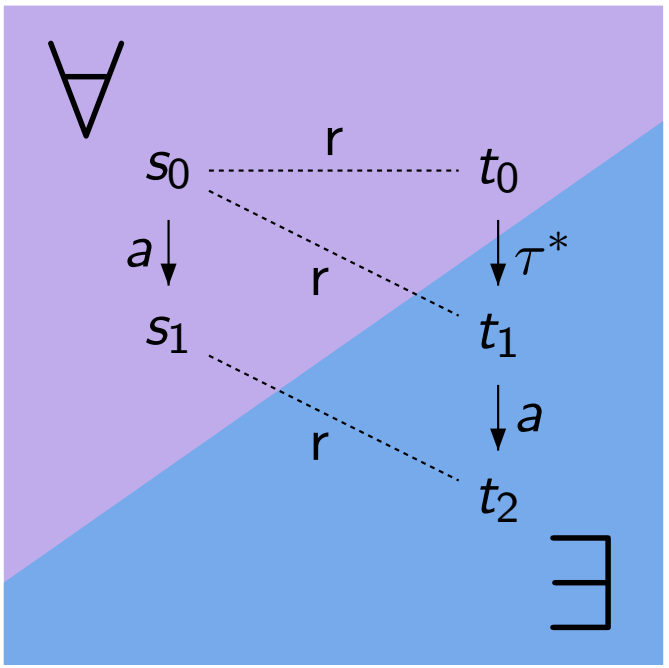
Simulación y bisimulación ramificada

Una *simulación ramificada (branching simulation)* es una relación $r \in S \times S'$ entre dos LTS, $P = (S, L, \longrightarrow, s_0)$ y $P' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in r$ y



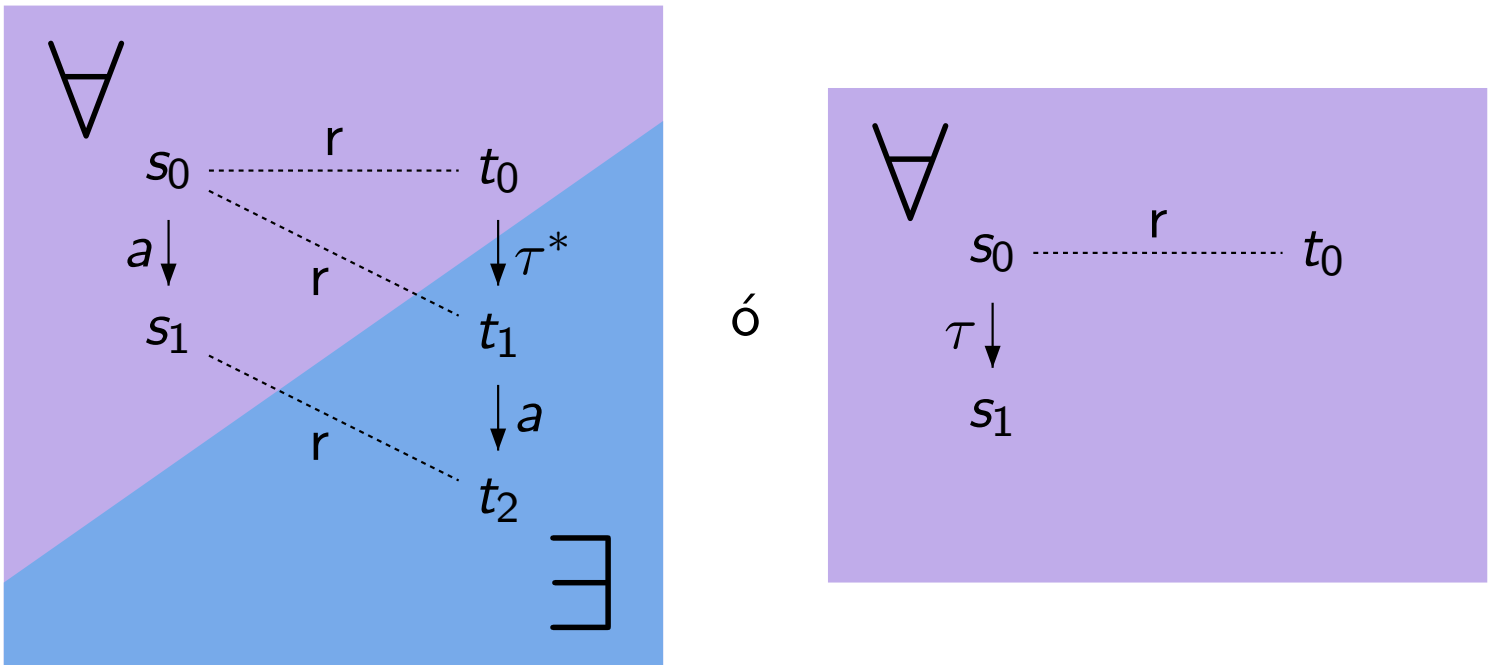
Simulación y bisimulación ramificada

Una *simulación ramificada (branching simulation)* es una relación $r \in S \times S'$ entre dos LTS, $P = (S, L, \longrightarrow, s_0)$ y $P' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in r$ y



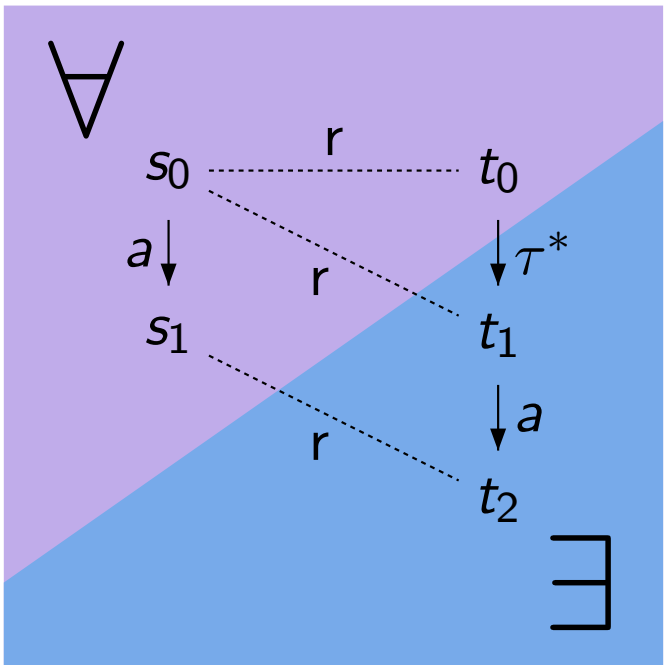
Simulación y bisimulación ramificada

Una *simulación ramificada (branching simulation)* es una relación $r \in S \times S'$ entre dos LTS, $P = (S, L, \longrightarrow, s_0)$ y $P' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in r$ y

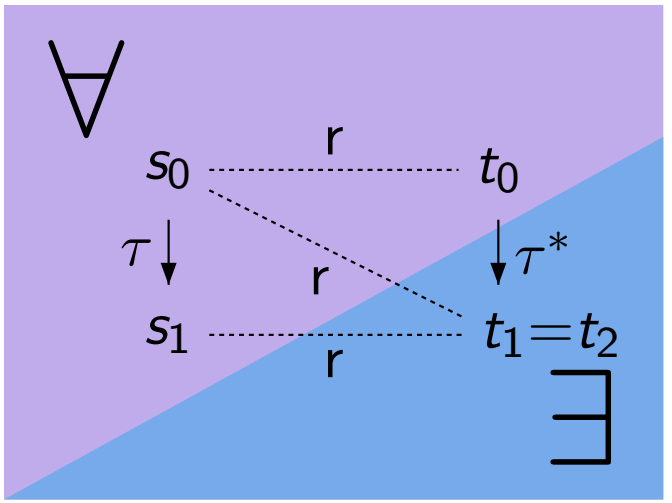


Simulación y bisimulación ramificada

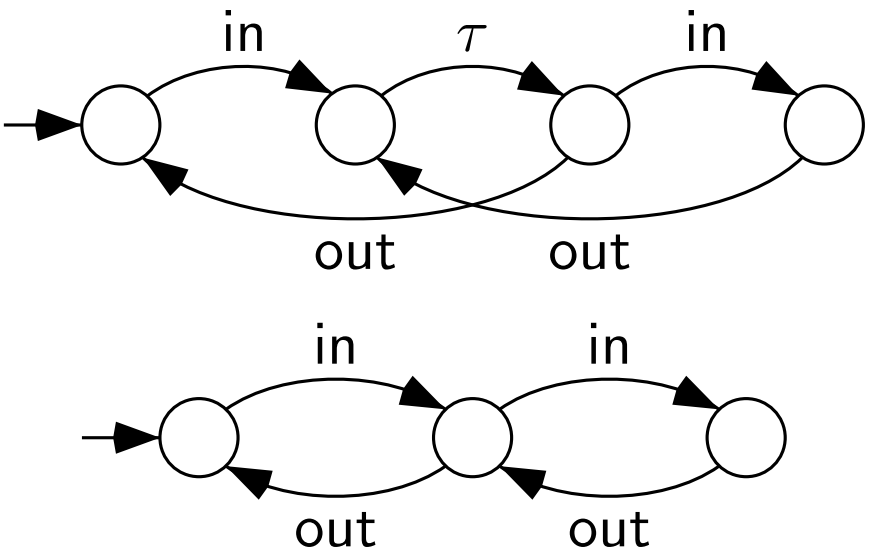
Una *simulación ramificada (branching simulation)* es una relación $r \in S \times S'$ entre dos LTS, $P = (S, L, \longrightarrow, s_0)$ y $P' = (S', L', \longrightarrow', s_0')$ donde se cumple que $(s_0, s_0') \in r$ y



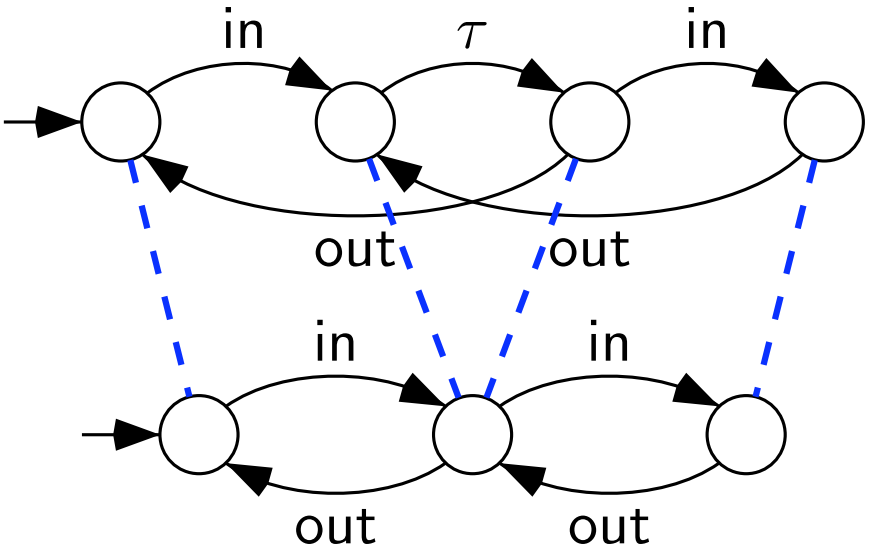
ó



Ejemplo Bisimulación débil y ramificada



Ejemplo Bisimulación débil y ramificada

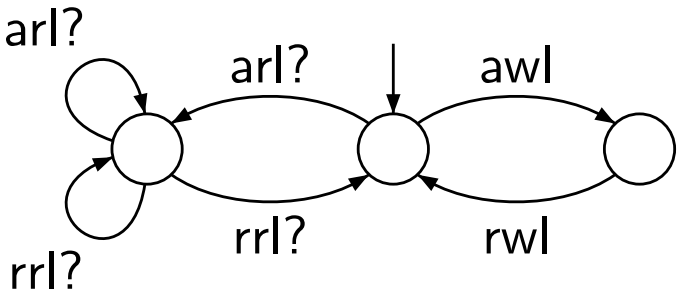


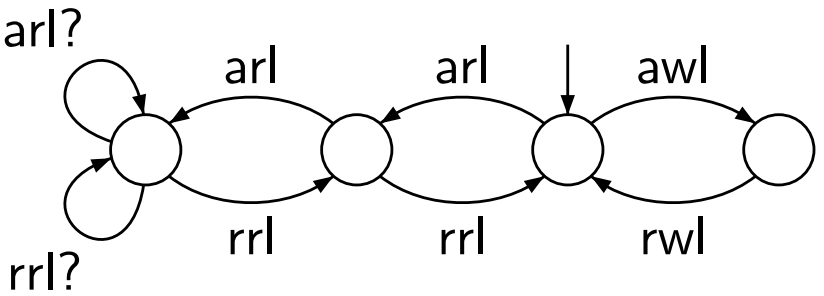
Refinamiento fuerte

- Un *refinamiento fuerte* entre los MTS P y P' es una relación r que satisface que:
  - es una simulación fuerte de P a P' teniendo en cuenta sólo las transiciones requeridas,
  - es una simulación fuerte de P' a P teniendo en cuenta sólo las transiciones probables.

Refinamiento fuerte

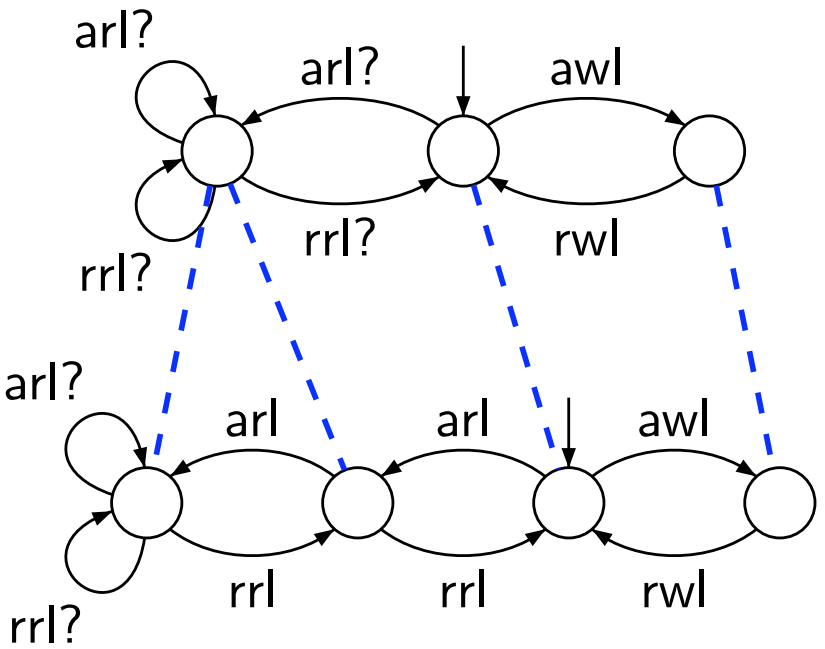
- Un *refinamiento fuerte* entre los MTS P y P' es una relación r que satisface que:
  - es una simulación fuerte de P a P' teniendo en cuenta sólo las transiciones requeridas,
  - es una simulación fuerte de P' a P teniendo en cuenta sólo las transiciones probables.





Refinamiento fuerte

- Un *refinamiento fuerte* entre los MTS P y P' es una relación r que satisface que:
  - es una simulación fuerte de P a P' teniendo en cuenta sólo las transiciones requeridas,
  - es una simulación fuerte de P' a P teniendo en cuenta sólo las transiciones probables.

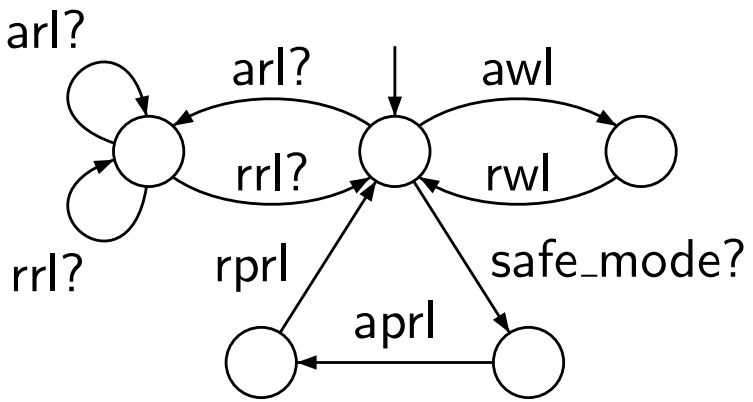


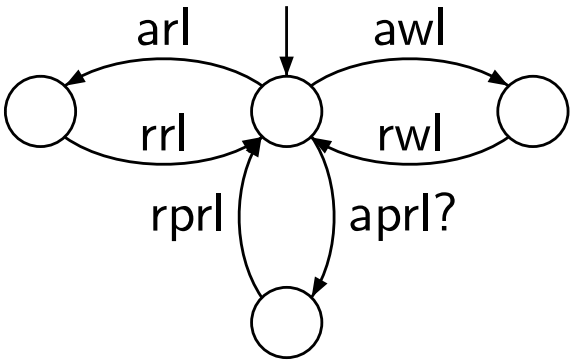
Refinamiento débil

- Un *refinamiento débil* entre los MTS P y P' es una relación r que satisface que:
  - es una simulación débil de P a P' teniendo en cuenta sólo las transiciones requeridas,
  - es una simulación débil de P' a P teniendo en cuenta sólo las transiciones probables.

Refinamiento débil

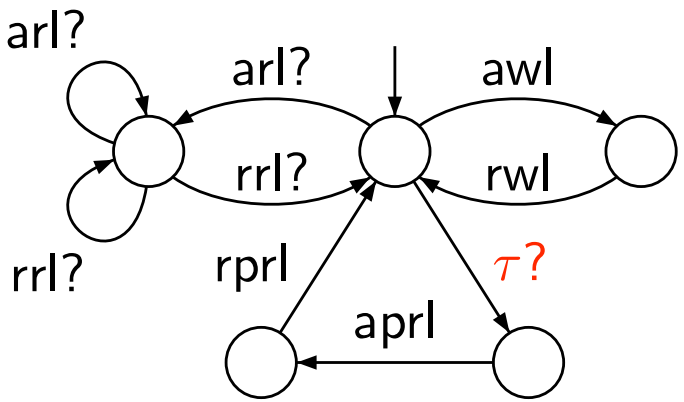
- Un *refinamiento débil* entre los MTS P y P' es una relación r que satisface que:
  - es una simulación débil de P a P' teniendo en cuenta sólo las transiciones requeridas,
  - es una simulación débil de P' a P teniendo en cuenta sólo las transiciones probables.

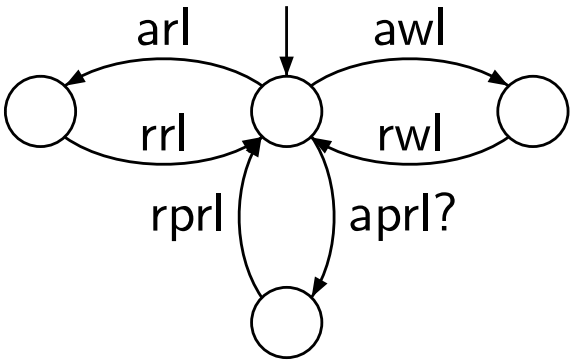




Refinamiento débil

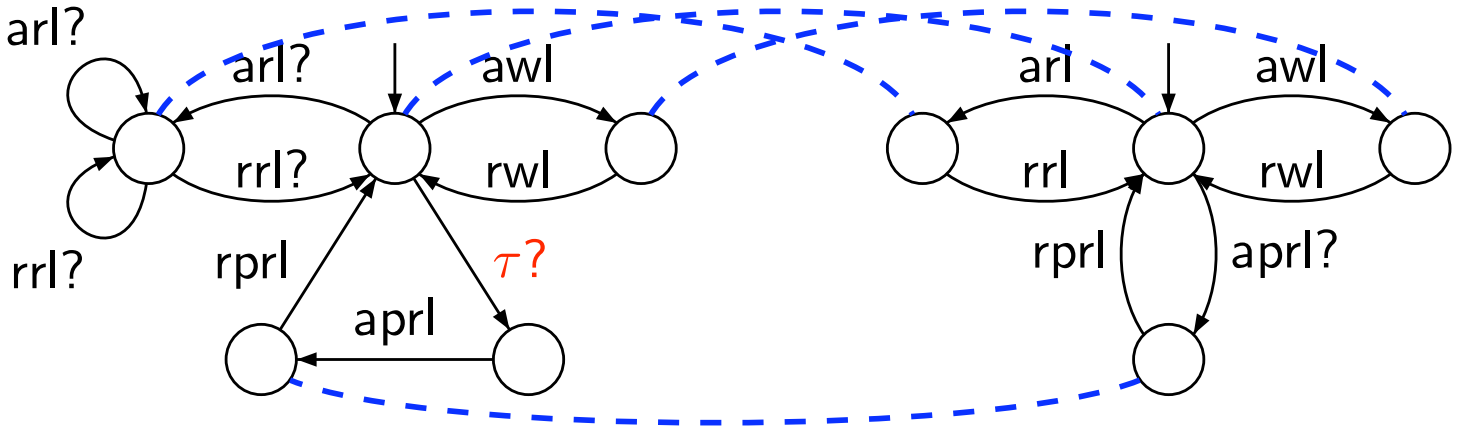
- Un *refinamiento débil* entre los MTS P y P' es una relación r que satisface que:
  - es una simulación débil de P a P' teniendo en cuenta sólo las transiciones requeridas,
  - es una simulación débil de P' a P teniendo en cuenta sólo las transiciones probables.





Refinamiento débil

- Un *refinamiento débil* entre los MTS P y P' es una relación r que satisface que:
  - es una simulación débil de P a P' teniendo en cuenta sólo las transiciones requeridas,
  - es una simulación débil de P' a P teniendo en cuenta sólo las transiciones probables.

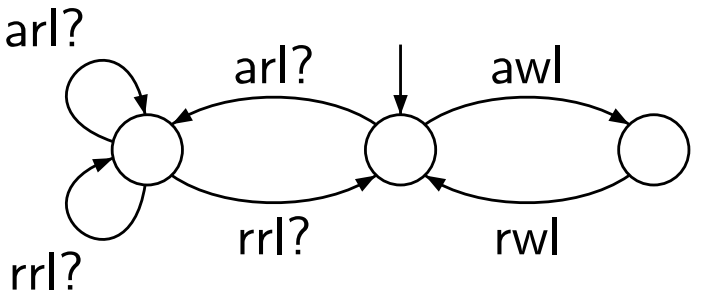


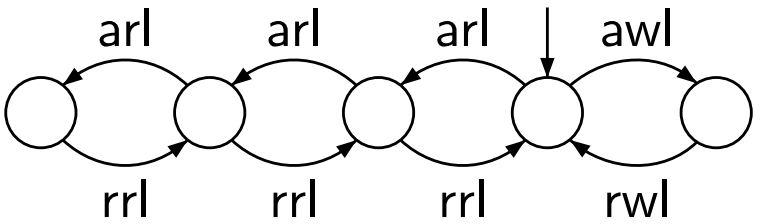
Implementación fuerte

- Una *implementación fuerte* entre el MTS P y el LTS P' es una relación r que satisface que es un refinamiento fuerte entre el MTS P y el MTS P' donde al LTS P' lo vemos como un MTS el cual tiene el mismo conjunto de transiciones probables y requeridas.

Implementación fuerte

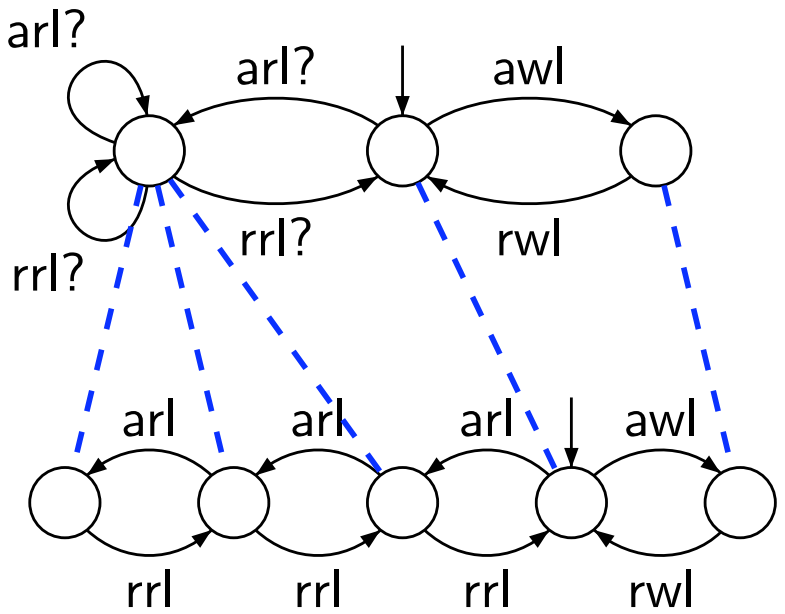
- Una *implementación fuerte* entre el MTS P y el LTS P' es una relación r que satisface que es un refinamiento fuerte entre el MTS P y el MTS P' donde al LTS P' lo vemos como un MTS el cual tiene el mismo conjunto de transiciones probables y requeridas.





Implementación fuerte

- Una *implementación fuerte* entre el MTS P y el LTS P' es una relación r que satisface que es un refinamiento fuerte entre el MTS P y el MTS P' donde al LTS P' lo vemos como un MTS el cual tiene el mismo conjunto de transiciones probables y requeridas.

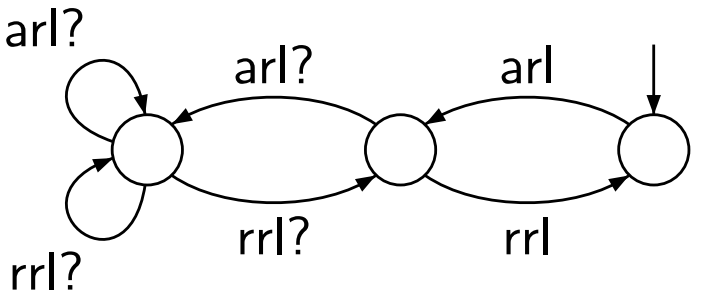


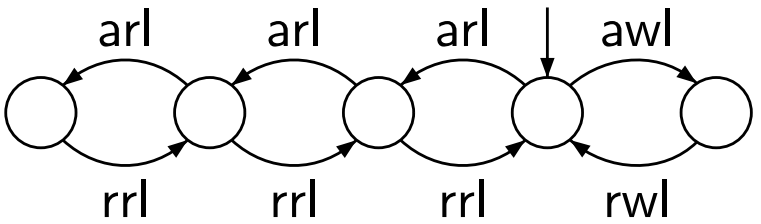
Implementación débil

- Una *implementación débil* entre el MTS P y el LTS P' es una relación r que satisface que es un refinamiento débil entre el MTS P y el MTS P' donde al LTS P' lo vemos como un MTS el cual tiene el mismo conjunto de transiciones probables y requeridas.

Implementación débil

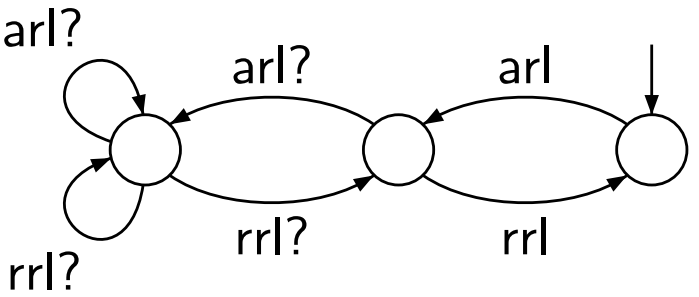
- Una *implementación débil* entre el MTS P y el LTS P' es una relación r que satisface que es un refinamiento débil entre el MTS P y el MTS P' donde al LTS P' lo vemos como un MTS el cual tiene el mismo conjunto de transiciones probables y requeridas.

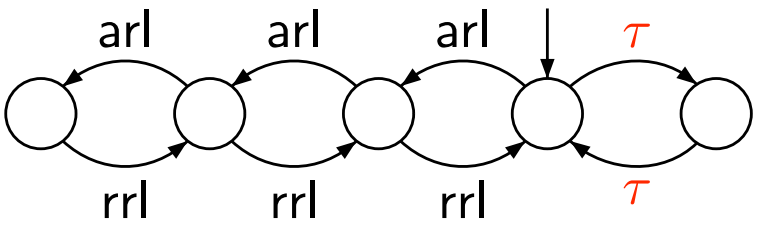




Implementación débil

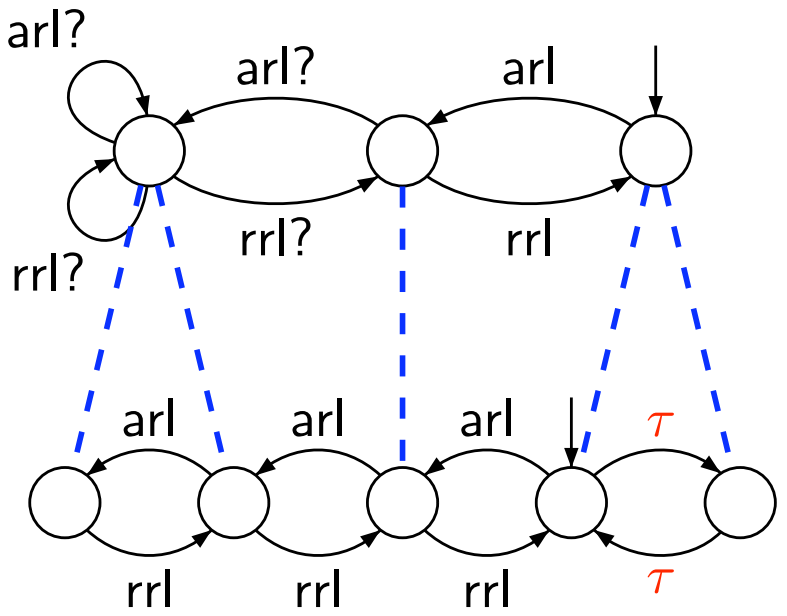
- Una *implementación débil* entre el MTS P y el LTS P' es una relación r que satisface que es un refinamiento débil entre el MTS P y el MTS P' donde al LTS P' lo vemos como un MTS el cual tiene el mismo conjunto de transiciones probables y requeridas.





Implementación débil

- Una *implementación débil* entre el MTS P y el LTS P' es una relación r que satisface que es un refinamiento débil entre el MTS P y el MTS P' donde al LTS P' lo vemos como un MTS el cual tiene el mismo conjunto de transiciones probables y requeridas.



Automatizando el análisis de dichas propiedades

Queremos:

- automatizar el chequeo de estas propiedades
- un lenguaje intermedio como modelado

Utilizaremos SAT solvers para el análisis

¿Qué es SAT? Determinar si una fórmula proposicional es satisfactible.

¿Es computable el problema SAT? Sí, existen algoritmos correctos y completos que resuelven SAT. Algunos de ellos son,

- tablas de verdad
- David Putnam

¿Qué es Alloy?

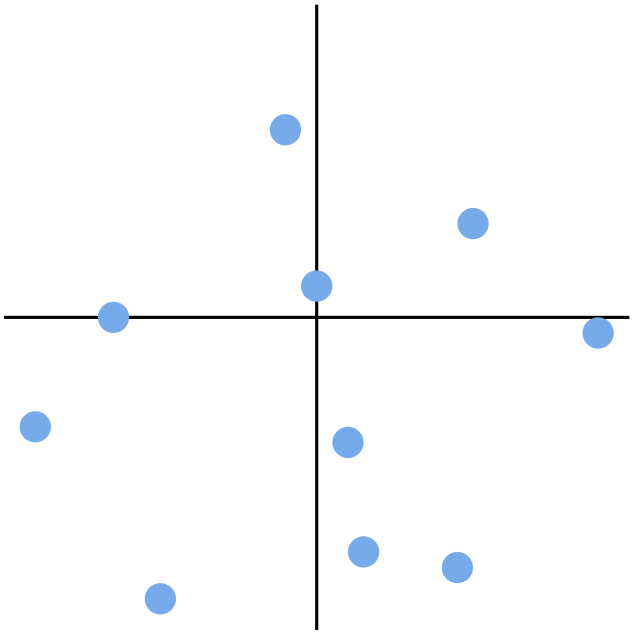
- un lenguaje para el modelado de software
  - sintaxis clara y expresiva para lógica de primer orden con relaciones
  - objetos expresados mediante signaturas
  - hechos (axiomas) sobre los distintos tipos de objetos
  - funciones, aserciones y predicados

- una herramienta de análisis
  - análisis exhaustivo y simulación con cota
  - visualización personalizable
  - API disponible para su uso en otras herramientas
  - basados en SAT solvers "sacados del estante" (off-the-shelf)

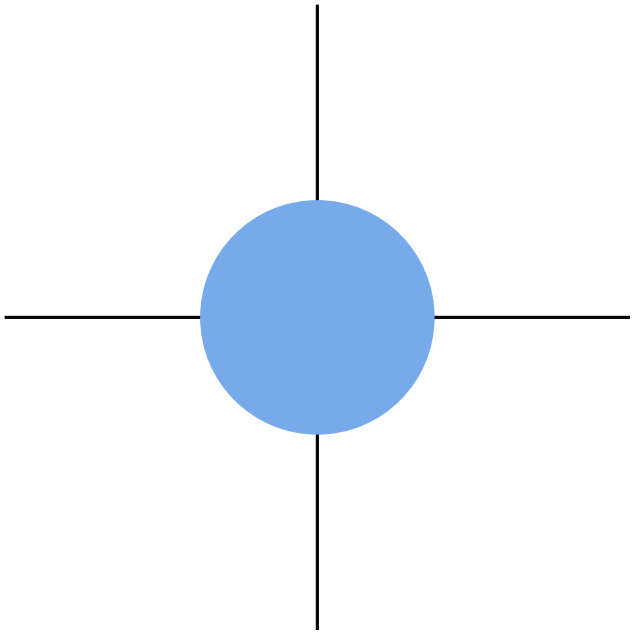
Cómo funciona Alloy? Análisis completo acotado

Observaciones sobre el análisis de diseño:

- la mayoría de las aserciones son erroneas
- la mayoría de los errores tienen contrajemplos pequeños



testing: algunos casos de tamaño arbitrario



entorno completo de todos los casos pequeños

Modelado en Alloy

- Estados

```
abstract sig State {}
sig CState extends State {}
```

- Acciones

```
abstract sig Action {}
sig CAction extends Action {}
one sig Tau extends Action {}
```

- LTS

```
abstract sig LTS {
    init: one State,
    trans: (Action -> State) -> State
}
```

- Simulación y bisimulación fuerte

```
pred StrongSimulation(r: State -> State, t, t': (Action -> State) -> State) {
    all a: Action | (~r).(t[a]) in (t'[a]).(~r)
}

pred StrongBisimulation(r: State -> State, t, t': (Action -> State) -> State) {
    StrongSimulation [r, t, t'] and StrongSimulation [~r, t', t]

}
```

Modelado en Alloy

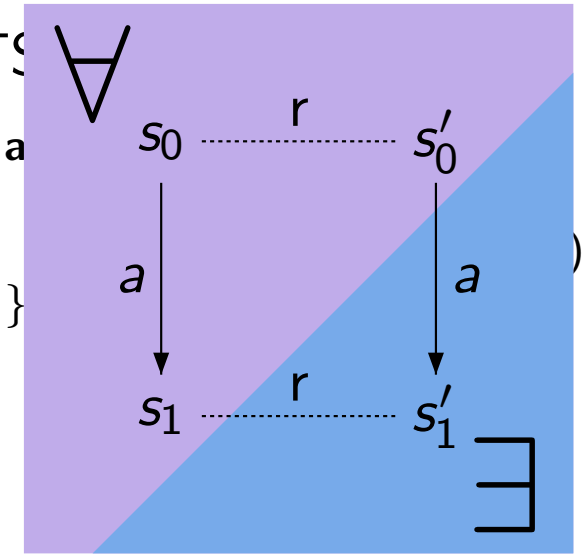
- Estados

```
abstract sig State {}
sig CState extends State {}
```

- Acciones

```
abstract sig Action {}
sig CAction extends Action {}
one sig Tau extends Action {}
```

- LTS



- Simulación y bisimulación fuerte

```
pred StrongSimulation(r: State -> State, t, t': (Action -> State) -> State) {
    all a: Action | (~r).(t[a]) in (t'[a]).(~r)
}

pred StrongBisimulation(r: State -> State, t, t': (Action -> State) -> State) {
    StrongSimulation [r, t, t'] and StrongSimulation [~r, t', t]

}
```

Verificación de algunas propiedades utilizando Alloy

- Dados dos MTS, verificar si existe un refinamiento entre ellos.
- Todo refinamiento fuerte es también refinamiento débil. Esta propiedad es válida, no así su recíproca.
- Un sistema es inconsiste, si no existe un refinamiento común entre los MTS que lo especifican.
- Verificar si existe un refinamiento común observacional entre dos MTS. Esto es, si existe una implementación que satisface que es refinamiento débil respecto de cada MTS considerando sólo las acciones correspondiente a cada uno.

Predicados de alto orden sobre MTS y LTS

Propiedad esperable:

$$N \preceq M \Rightarrow I[M] \subseteq I[N]$$

Predicados de alto orden sobre MTS y LTS

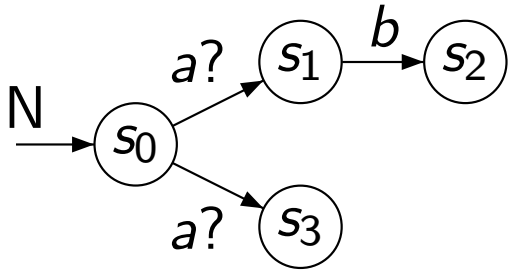
Propiedad esperable:

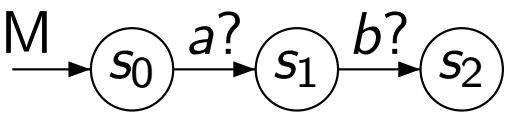
$$N \preceq M \Rightarrow I[M] \subseteq I[N]$$

Propiedad deseable:

$$I[M] \subseteq I[N] \Rightarrow N \preceq M$$

- demostrada verdadera por Huth en 2005
- pero es falsa! (Fischbein 2007)
- propiedad representativa de muchas otras variantes





Predicados de alto orden sobre MTS y LTS

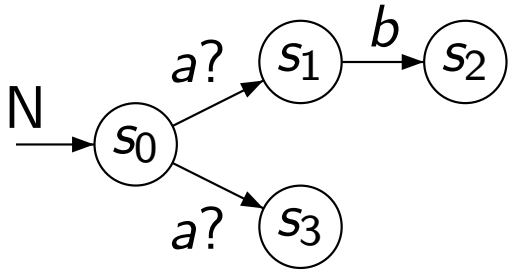
Propiedad esperable:

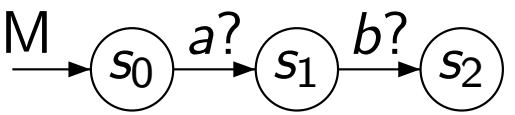
$$N \preceq M \Rightarrow I[M] \subseteq I[N]$$

Propiedad deseable:

$$N \not\preceq M \Rightarrow I[M] \not\subseteq I[N]$$

- demostrada verdadera por Huth en 2005
- pero es falsa! (Fischbein 2007)
- propiedad representativa de muchas otras variantes





Problemas

$$N \not\preceq M \Rightarrow I[M] \not\subseteq I[N]$$

Se puede expresar esta propiedad en Alloy?

Problemas

$$N \not\preceq M \Rightarrow I[M] \not\subseteq I[N]$$

Se puede expresar esta propiedad en Alloy? SI

Se puede analizar esta propiedad en Alloy?

Problemas

$$N \not\preceq M \Rightarrow I[M] \not\subseteq I[N]$$

Se puede expresar esta propiedad en Alloy? SI

Se puede analizar esta propiedad en Alloy? NO

- Problema: Alloy no puede analizar cuantificación de alto orden

  Es solucionable?

Problemas

$$N \not\leq M \Rightarrow I[M] \not\subseteq I[N]$$

Se puede expresar esta propiedad en Alloy? SI

Se puede analizar esta propiedad en Alloy? NO

- Problema: Alloy no puede analizar cuantificación de alto orden

  Es solucionable? SI

  Solución: Creación de signatura de relaciones

Problemas

$$N \not\leq M \Rightarrow I[M] \not\subseteq I[N]$$

Se puede expresar esta propiedad en Alloy? SI

Se puede analizar esta propiedad en Alloy? NO

- Problema: Alloy no puede analizar cuantificación de alto orden

  Es solucionable? SI

  Solución: Creación de signatura de relaciones

Pero esto no alcanza...

- Problema: Población de relaciones

  Es solucionable?

Problemas

$$N \not\leq M \Rightarrow I[M] \not\subseteq I[N]$$

Se puede expresar esta propiedad en Alloy? SI

Se puede analizar esta propiedad en Alloy? NO

- Problema: Alloy no puede analizar cuantificación de alto orden

  Es solucionable? SI

  Solución: Creación de signatura de relaciones

Pero esto no alcanza...

- Problema: Población de relaciones

  Es solucionable? SI

  Solución: Axioma de generación

Problemas

$$N \not\leq M \Rightarrow I[M] \not\subseteq I[N]$$

Se puede expresar esta propiedad en Alloy? SI

Se puede analizar esta propiedad en Alloy? NO

- Problema: Alloy no puede analizar cuantificación de alto orden

  Es solucionable? SI

  Solución: Creación de signatura de relaciones

Pero esto no alcanza...

- Problema: Población de relaciones

  Es solucionable? SI

  Solución: Axioma de generación

Pero esto tampoco alcanza...

Problemas (cont.)

- Problema: Capacidad limitada de instancias.

  Es solucionable?

Problemas (cont.)

- Problema: Capacidad limitada de instancias.

  Es solucionable? SI

  Cuántas relaciones tenemos para i estados?

$$\sum_{i=0}^{n} \binom{n}{i} = 2^n, \text{ donde } n = i \times i$$

  En números...

  2 estados $\longrightarrow$ 16 relaciones,

  3 estados $\longrightarrow$ 512 relaciones,

  4 estados $\longrightarrow$ 65536 relaciones

  Cuántas instancias soporta Alloy? 480!

Problemas (cont.)

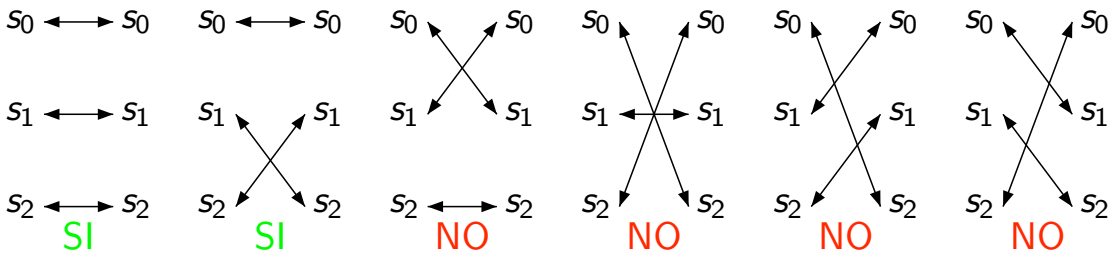
Solución: Eliminar el axioma de generación. Pasarle las instancias a Alloy.

Supondremos:

- todos los LTS/MTS tendrán como estado inicial el estado s_0
- toda relación debe contener el par (s_0, s_0)

Entonces,

- generamos las 512 relaciones, eliminar las que no contengan el par (s_0, s_0)
- encontramos las biyecciones válidas de 3 estados.



- utilizamos Alloy Analyzer para encontrar el mínimo conjunto de Relaciones.

Problemas (cont.)

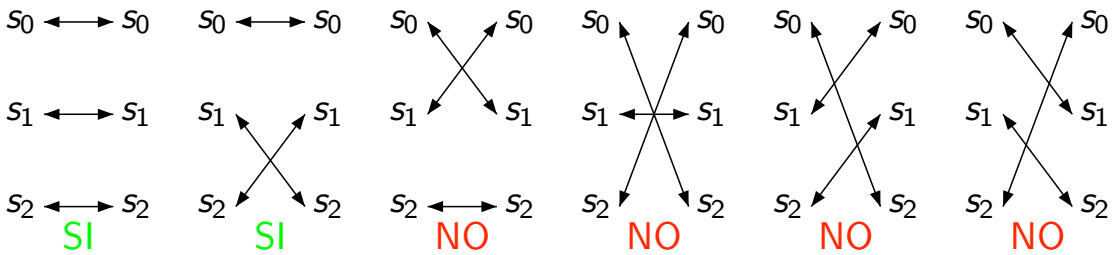
Solución: Eliminar el axioma de generación. Pasarle las instancias a Alloy.

Supondremos:

- todos los LTS/MTS tendrán como estado inicial el estado s_0
- toda relación debe contener el par (s_0, s_0)

Entonces,

- generamos las 512 relaciones, eliminar las que no contengan el par (s_0, s_0)
- encontramos las biyecciones válidas de 3 estados.



- utilizamos Alloy Analyzer para encontrar el mínimo conjunto de Relaciones.

Pero esto tampoco alcanza...

Problemas (cont.)

- Problema: Generación de MTS y LTS
- Es solucionable?

Problemas (cont.)

- Problema: Generación de MTS y LTS
- Es solucionable? SI

  Solución: Generamos los MTS, y se lo pasamos de a pares a Alloy.

  Cómo deben ser nuestros MTS candidatos?
  - tener como estado inicial s_0
  - al menos debe salir una transición del estado s_0.
  - las transiciones requeridas deben estar incluidas en las probables
  - deben ser conexos

  Una cota

$$\sum_{i=0}^{|S|^2 \cdot L} \left[\binom{|S|^2 \cdot |L|}{i} - \left\{ {|S|^2 \cdot |L| - |S| \cdot |L| \atop i} \right\} \right] \cdot 2^i \qquad \text{donde, } \left\{ {k \atop l} \right\} = \begin{cases} \binom{k}{l} & \text{si } l \leq k \\ 0 & \text{c.c.} \end{cases}$$

  S representa el número de estados y L el de acciones.

Problemas (cont.)

- Problema: Generación de MTS y LTS
- Es solucionable? SI

  Solución: Generamos los MTS, y se lo pasamos de a pares a Alloy.

  Cómo deben ser nuestros MTS candidatos?
  - tener como estado inicial s_0
  - al menos debe salir una transición del estado s_0.
  - las transiciones requeridas deben estar incluidas en las probables
  - deben ser conexos

  Una cota

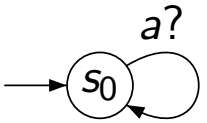
$$\sum_{i=0}^{|S|^2 \cdot L} \left[\binom{|S|^2 \cdot |L|}{i} - \left\{ {|S|^2 \cdot |L| - |S| \cdot |L| \atop i} \right\} \right] \cdot 2^i \qquad \text{donde, } \left\{ {k \atop l} \right\} = \begin{cases} \binom{k}{l} & \text{si } l \leq k \\ 0 & \text{c.c.} \end{cases}$$

  S representa el número de estados y L el de acciones.

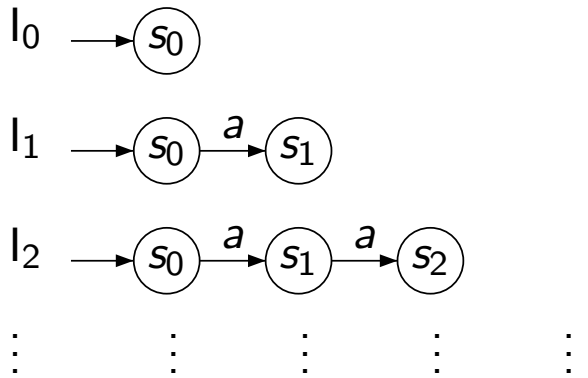
Y no, nuevamente no alcanza...

Problemas (cont.)

- Problema: Cantidad posible de implementaciones



Éste tiene infinitas implementaciones. En particular, la siguiente familia de LTS son todas implementaciones del MTS anterior.



$$I_n = (\{s_i\}^n, \{a\}, \{s_i \rightarrow s_{i+1} \mid 0 \leq i \leq n\}, s_0)$$

Solución: Analizar para todas las implementaciones de tamaño N. Donde N es una cota arbitraria. En nuestro caso, N será 3 estados.

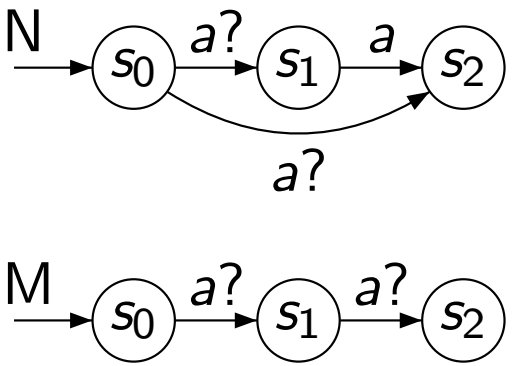
Metodología de análisis

- Tomamos 2 MTS,
- analizamos si se da el refinamiento fuerte con Alloy Analyzer.
  - si no se satisface, estos MTS son buenos candidatos, continuamos.
  - si se satisface, descartamos estos MTS y volvemos a empezar.
- analizamos el consecuente de la prop con Alloy Analyzer.
  - si no se satisface, estos MTS son buenos candidatos, continuamos.
  - si se satisface, descartamos estos MTS y volvemos a empezar.
- corroboramos a mano, si estos MTS conforman un contraejemplo real o espúreo.

Contraejemplos de la propiedad

$$I[M] \subseteq I[N] \Rightarrow N \preceq M$$

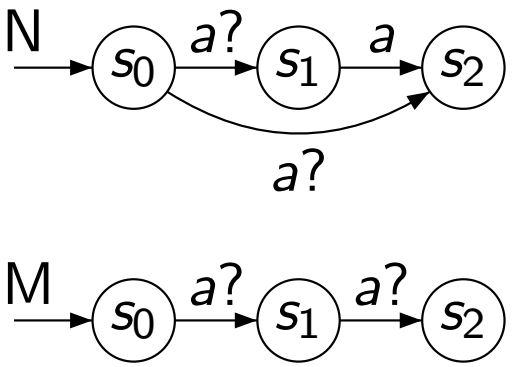
Contraejemplos reales:

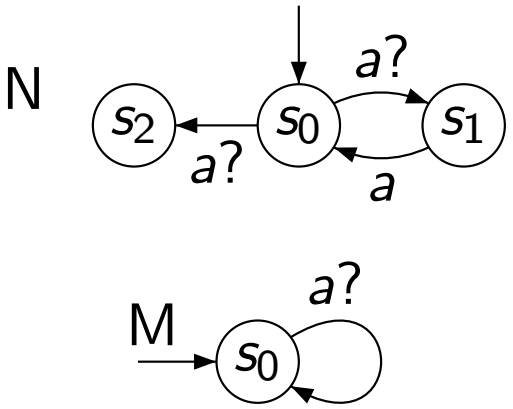


Contraejemplos de la propiedad

$$I[M] \subseteq I[N] \Rightarrow N \preceq M$$

Contraejemplos reales:

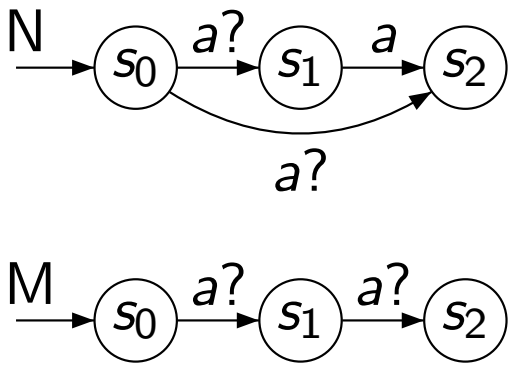


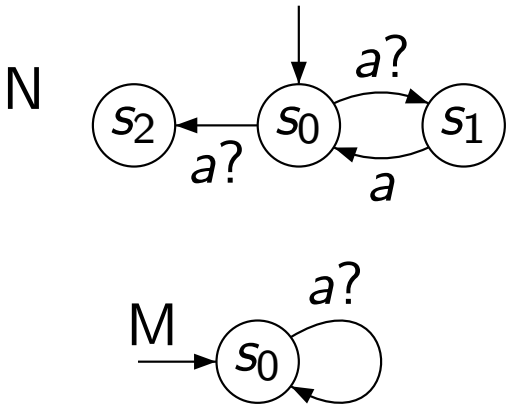


Contraejemplos de la propiedad

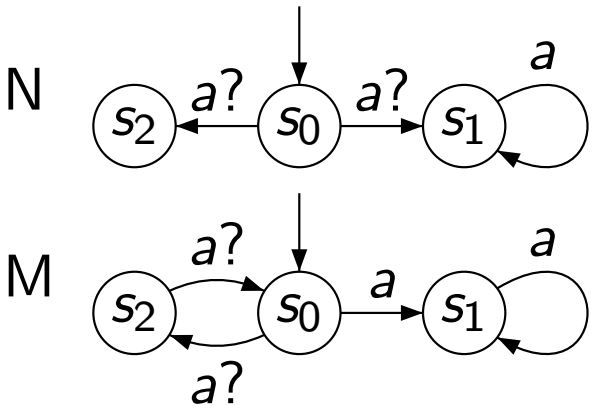
$$I[M] \subseteq I[N] \Rightarrow N \preceq M$$

Contraejemplos reales:





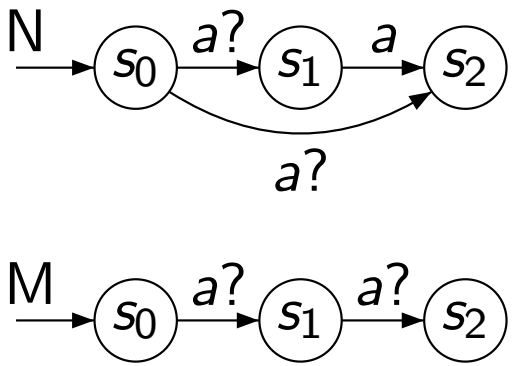
Contraejemplo espúreo:

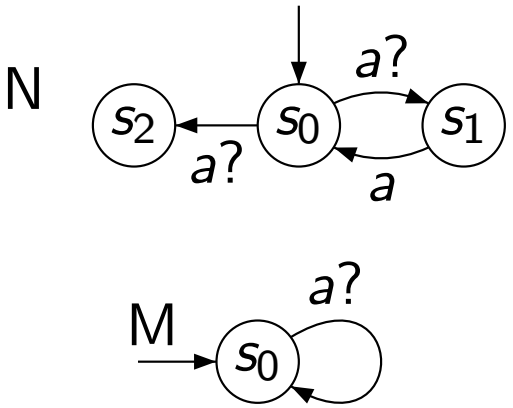


Contraejemplos de la propiedad

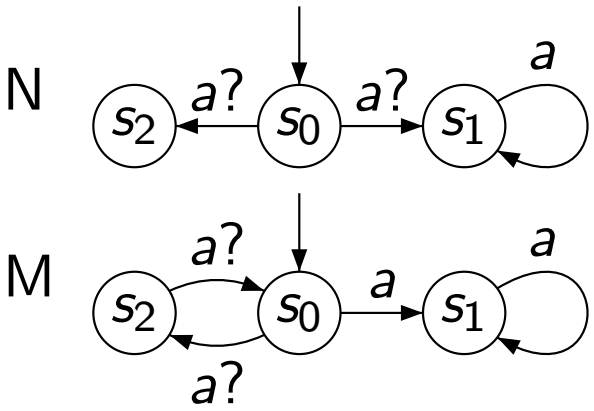
$$I[M] \subseteq I[N] \Rightarrow N \preceq M$$

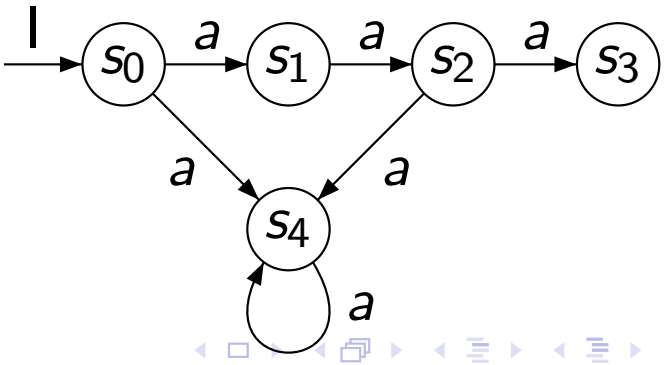
Contraejemplos reales:





Contraejemplo espúreo:





Tiempos de ejecución y otras yerbas

- Trabajamos con 3 estados y 1 acción.
- Cantidad de MTS 18.954. Cantidad de MTS útiles 14.690.
- Tenemos sólo 215.796.100 pares de MTS vs 387.420.489 que tendríamos sin las optimizaciones
- Tiempo de análisis por cada par de MTS, entre 200ms y 1s. Uso de memoria menor a 1GB.
- En tiempo, nos llevaría, 1 mes si contamos con 100 nodos.
- Actualmente contamos con 20 candidatos a contraejemplos y seguimos buscando.

Otras herramientas

- MTSChecker, desarrollada por D. Fischbein.

  Permite analizar refinamientos e implementaciones entre MTS/LTS.

  Analiza si se da o no dicha propiedad.

  Debemos pasarle las instancias de los MTS/LTS.
- MTSA, es una extensión de LTSA (Labelled Transition System Analyzer). Utiliza el motor de MTSChecker.

  Ambas herramientas son más veloces que Alloy.

  Análisis limitado a verificación de refinamientos.
- DynAlloy, es una extensión de Alloy, permite describir propiedades mediante acciones
- Kodkod, es el motor de Alloy.

  Permite un mejor manejo parcial de cotas.

Conclusiones y trabajo futuro

Conclusiones:

- SAT solving es una herramienta viable para este tipo de análisis (al menos para el tamaño de MTS de la bibliografía que hay).
- Provee versatilidad para el análisis de diferentes tipos de propiedades.
- El lenguaje Alloy permite expresar naturalmente las propiedades de nuestro interés

Conclusiones y trabajo futuro

Conclusiones:

- SAT solving es una herramienta viable para este tipo de análisis (al menos para el tamaño de MTS de la bibliografía que hay).
- Provee versatilidad para el análisis de diferentes tipos de propiedades.
- El lenguaje Alloy permite expresar naturalmente las propiedades de nuestro interés

Trabajo futuro:

- Hacer un análisis completo de la propiedad vista.
- Si la propiedad es violada. ¿Existe una cota k para el tamaño de la implementación que hacer violar la propiedad? Es decir, ¿Existe k tal que $k \geq min\{\#i | i \in I(M) \backslash I(N)\}$?
- Analizar otros tipos de propiedades, como la mínima implementación común entre dos MTS.

¿Preguntas?