NOCIONES DE TOLERANCIA A FALLAS

Lic. Nicolás Ricci

UNRC - CONICET

TOLERANCIA A FALLAS

En ciertos contextos los sistemas de software desarrollan tareas de control críticas. En tales casos una falla puede resultar catastrófica.

El objetivo es elaborar metodologías que incorporen nociones formales para asistir en la obtención de sistemas de software **robustos**.

Sistemas que puedan llevar a cabo la tarea especificada incluso en presencia de **fallas**.

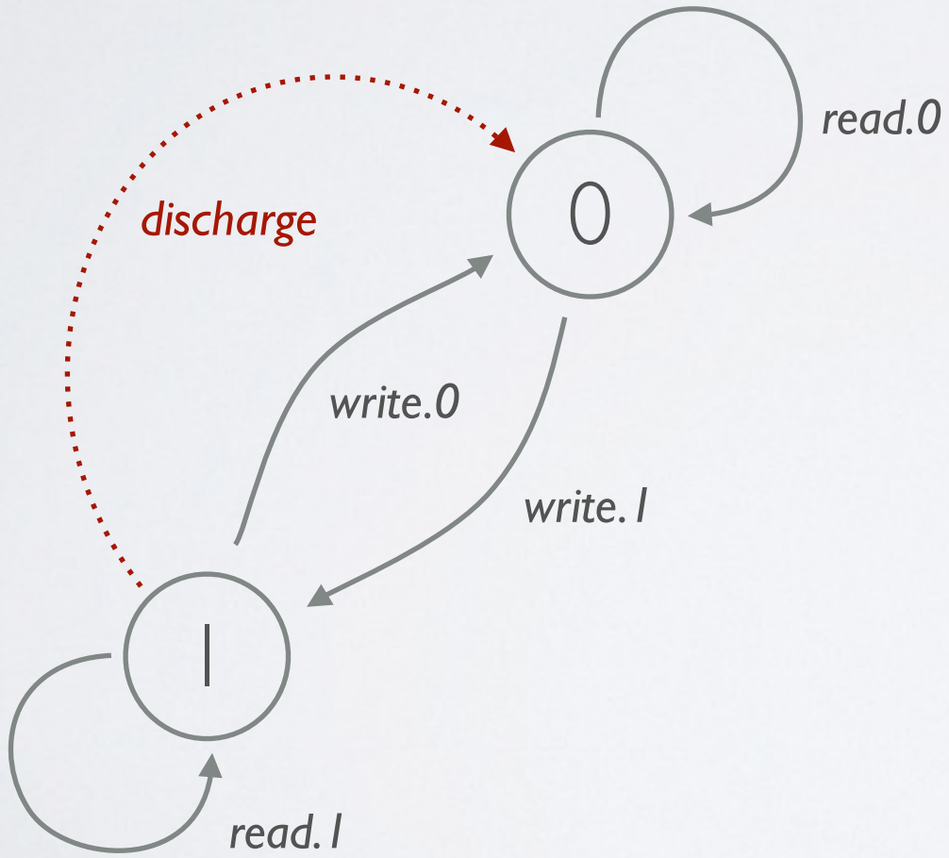
Existen varias técnicas que utilizadas:

Recovery Blocks	N-way Redundancy	Self-checking Software

EJEMPLO

Memory Cell

Consideremos una celda de memoria que almacena un bit.



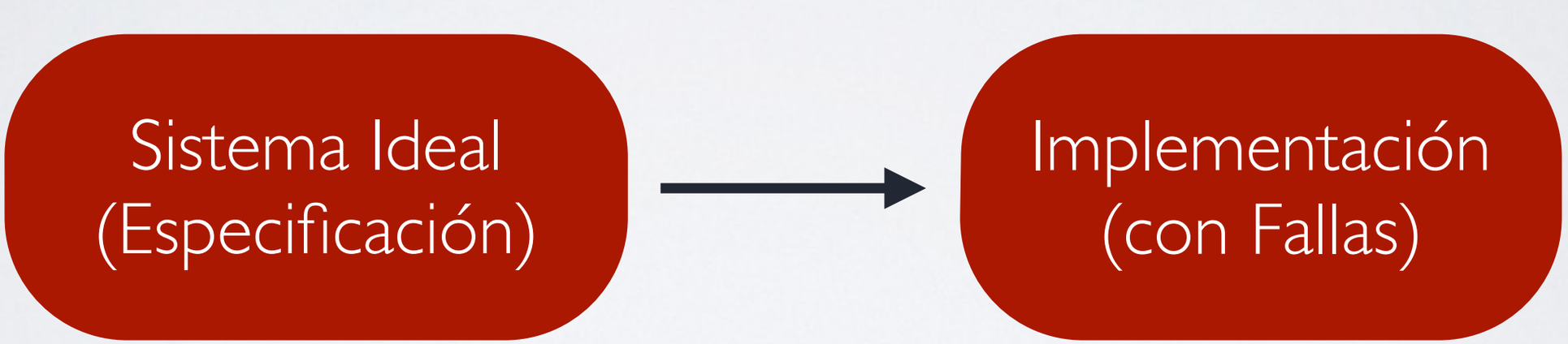
read.0 write.1 read.1 write.0 read.0

read.0 write.1 read.1 read.0

incorrecto!

ENFOQUE

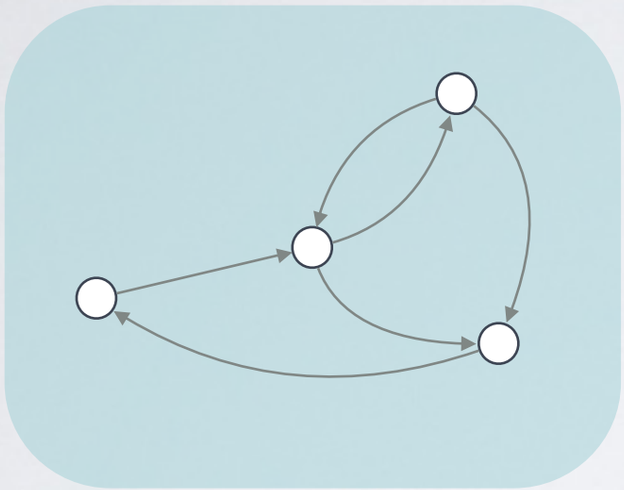
Elaborando distintos tipos de **relaciones de simulación** se puede caracterizar la **preservación** de **comportamiento normal** en presencia de fallas.



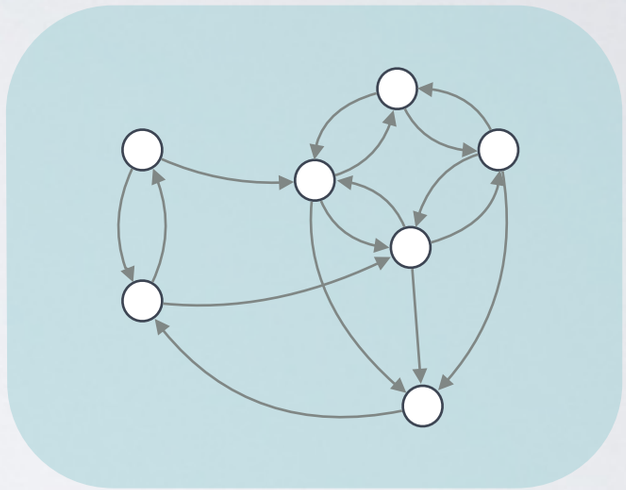
Para esto es necesario tomar un sistema como especificación. La relación entre ambos garantiza la preservación de comportamiento.

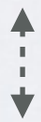
ENFOQUE

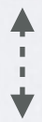
Sistemas de transición de estados etiquetados



Relación de
Simulación







$AG\neg(p \wedge q)$

Preservación de
propiedades

$AG\neg(p \wedge q)$

Propiedades de comportamiento

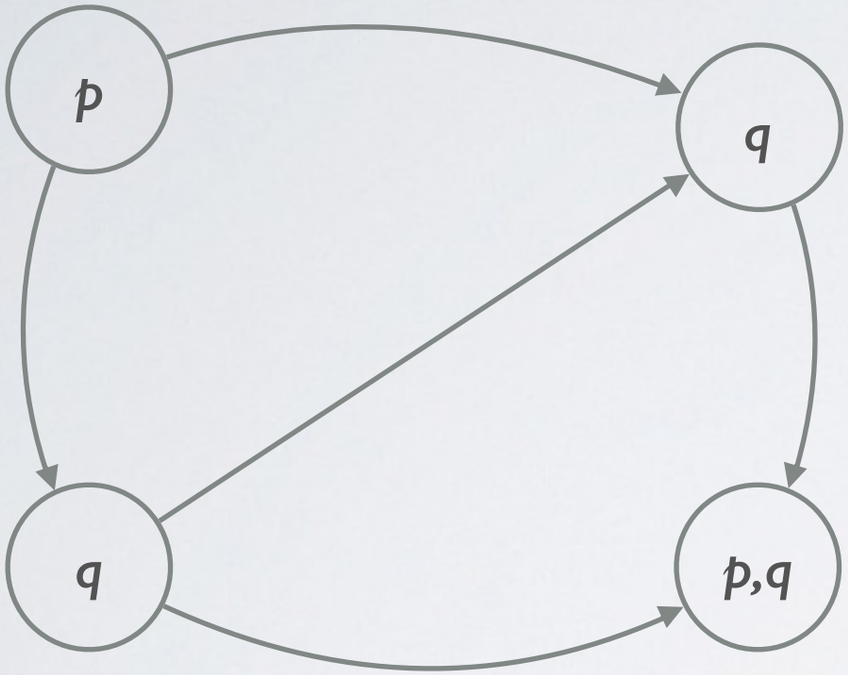
UN POCO DE LÓGICA

Para hablar sobre **sistemas reactivos** es necesario contar con una lógica apropiada. En este caso **CTL** (*Computation Tree Logic*).

CTL puede verse como una expansión de lógica proposicional con operadores temporales y cuantificación sobre caminos.

CTL

in a nutshell



Variables Proposicionales

$$p,\ q,\ r\ \ldots$$

Conectivas Proposicionales

$$\neg,\ \wedge,\ \vee\ \ldots$$

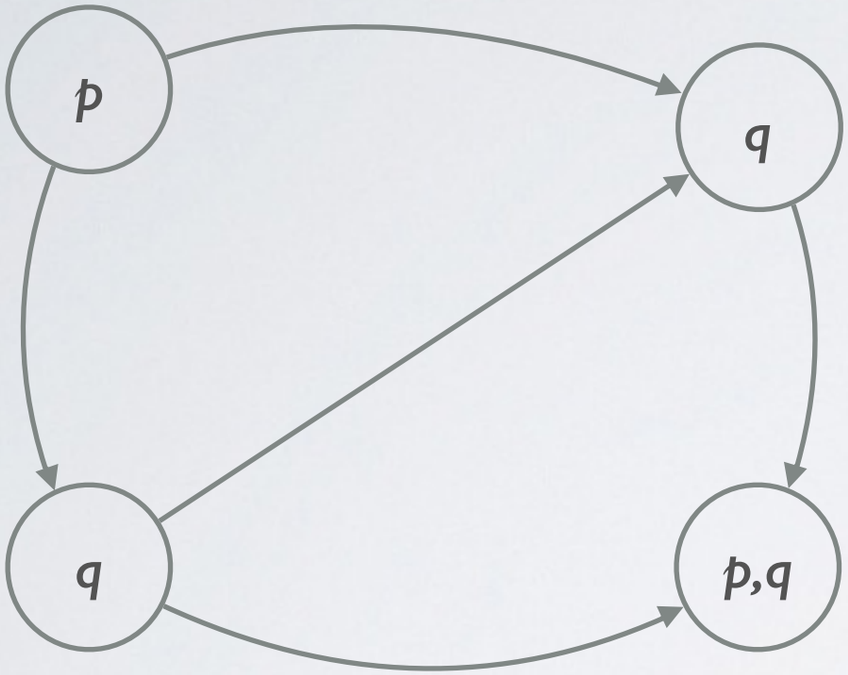
Cuantificadores

$$\mathbf{EX}\phi\ \ \mathbf{EF}\phi\ \ \mathbf{EG}\phi$$

$$\mathbf{AX}\phi\ \ \mathbf{AF}\phi\ \ \mathbf{AG}\phi$$

CTL

in a nutshell



p ✓

Variables Proposicionales

$$p,\ q,\ r\ \ldots$$

Conectivas Proposicionales

$$\neg,\ \wedge,\ \vee\ \ldots$$

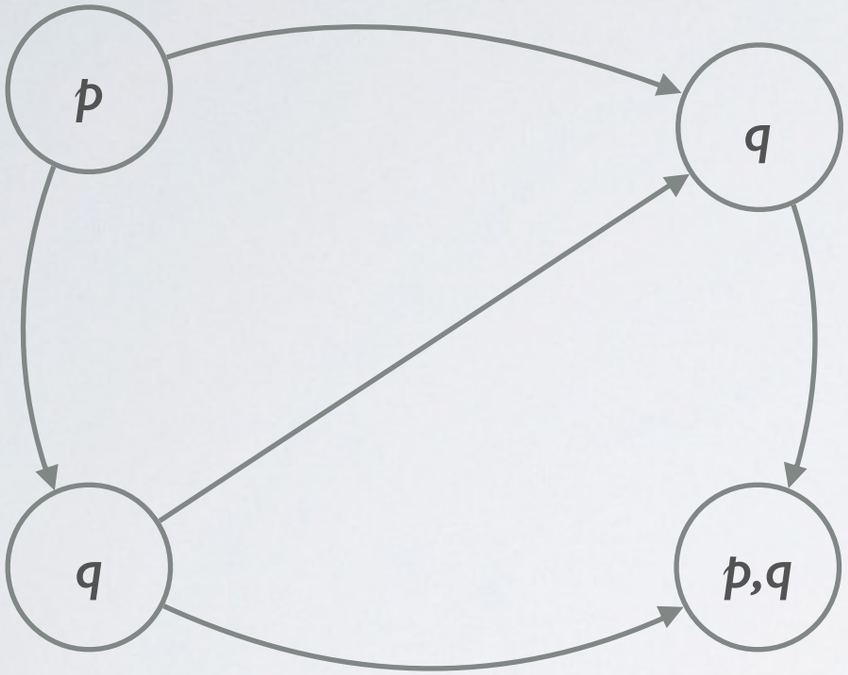
Cuantificadores

$$\mathbf{EX}\phi\ \ \mathbf{EF}\phi\ \ \mathbf{EG}\phi$$

$$\mathbf{AX}\phi\ \ \mathbf{AF}\phi\ \ \mathbf{AG}\phi$$

CTL

in a nutshell



Variables Proposicionales

$$p,\ q,\ r\ \ldots$$

Conectivas Proposicionales

$$\neg,\ \wedge,\ \vee\ \ldots$$

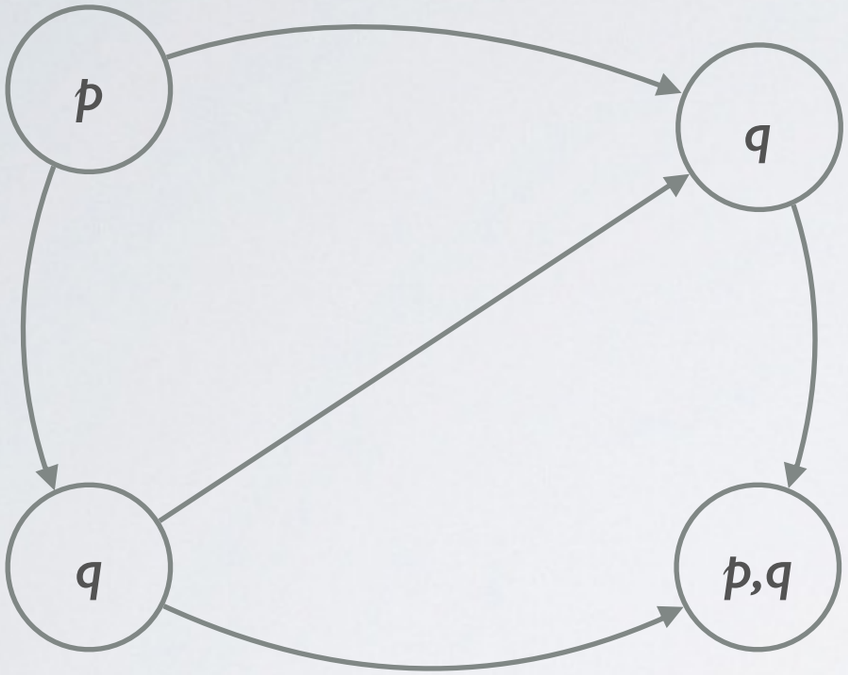
Cuantificadores

$$\mathbf{EX}\phi\ \ \mathbf{EF}\phi\ \ \mathbf{EG}\phi$$

$$\mathbf{AX}\phi\ \ \mathbf{AF}\phi\ \ \mathbf{AG}\phi$$

CTL

in a nutshell



Variables Proposicionales

$$p,\ q,\ r\ \ldots$$

Conectivas Proposicionales

$$\neg,\ \wedge,\ \vee\ \ldots$$

Cuantificadores

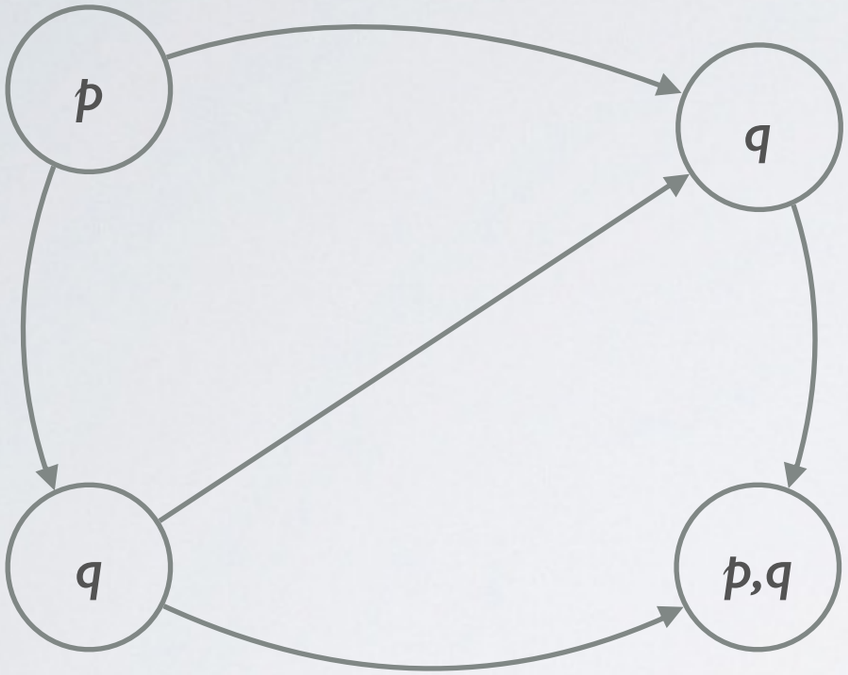
$$\mathbf{EX}\phi\ \ \mathbf{EF}\phi\ \ \mathbf{EG}\phi$$

$$\mathbf{AX}\phi\ \ \mathbf{AF}\phi\ \ \mathbf{AG}\phi$$

✗ $\mathbf{AG}p$

CTL

in a nutshell



Variables Proposicionales

$$p,\ q,\ r\ \ldots$$

Conectivas Proposicionales

$$\neg,\ \wedge,\ \vee\ \ldots$$

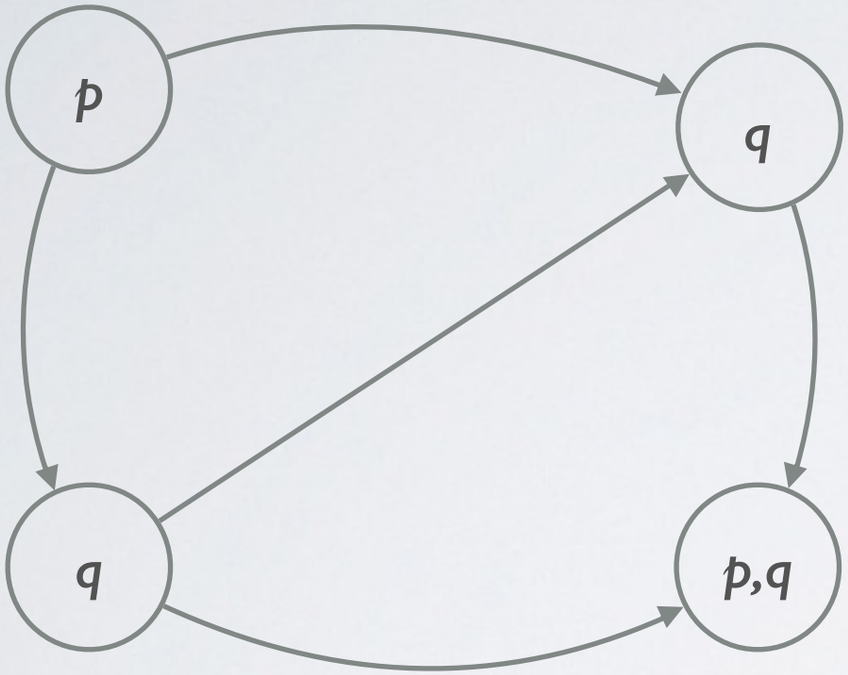
Cuantificadores

$$\mathbf{EX}\phi\ \ \mathbf{EF}\phi\ \ \mathbf{EG}\phi$$

$$\mathbf{AX}\phi\ \ \mathbf{AF}\phi\ \ \mathbf{AG}\phi$$

CTL

in a nutshell



Variables Proposicionales

$$p,\ q,\ r\ \ldots$$

Conectivas Proposicionales

$$\neg,\ \wedge,\ \vee\ \ldots$$

Cuantificadores

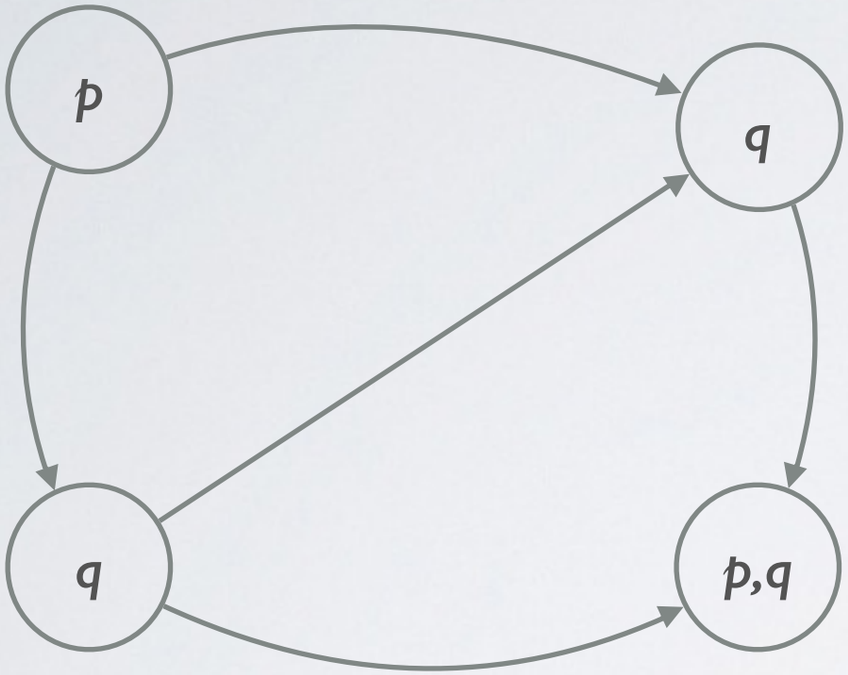
$$\mathbf{EX}\phi\ \ \mathbf{EF}\phi\ \ \mathbf{EG}\phi$$

$$\mathbf{AX}\phi\ \ \mathbf{AF}\phi\ \ \mathbf{AG}\phi$$

$$\mathbf{EF}(\mathbf{EX}p \wedge \mathbf{EX}\neg\mathbf{p})$$

CTL

in a nutshell



Variables Proposicionales

$$p,\ q,\ r\ \ldots$$

Conectivas Proposicionales

$$\neg,\ \wedge,\ \vee\ \ldots$$

Cuantificadores

$$\mathbf{EX}\phi\ \ \mathbf{EF}\phi\ \ \mathbf{EG}\phi$$

$$\mathbf{AX}\phi\ \ \mathbf{AF}\phi\ \ \mathbf{AG}\phi$$

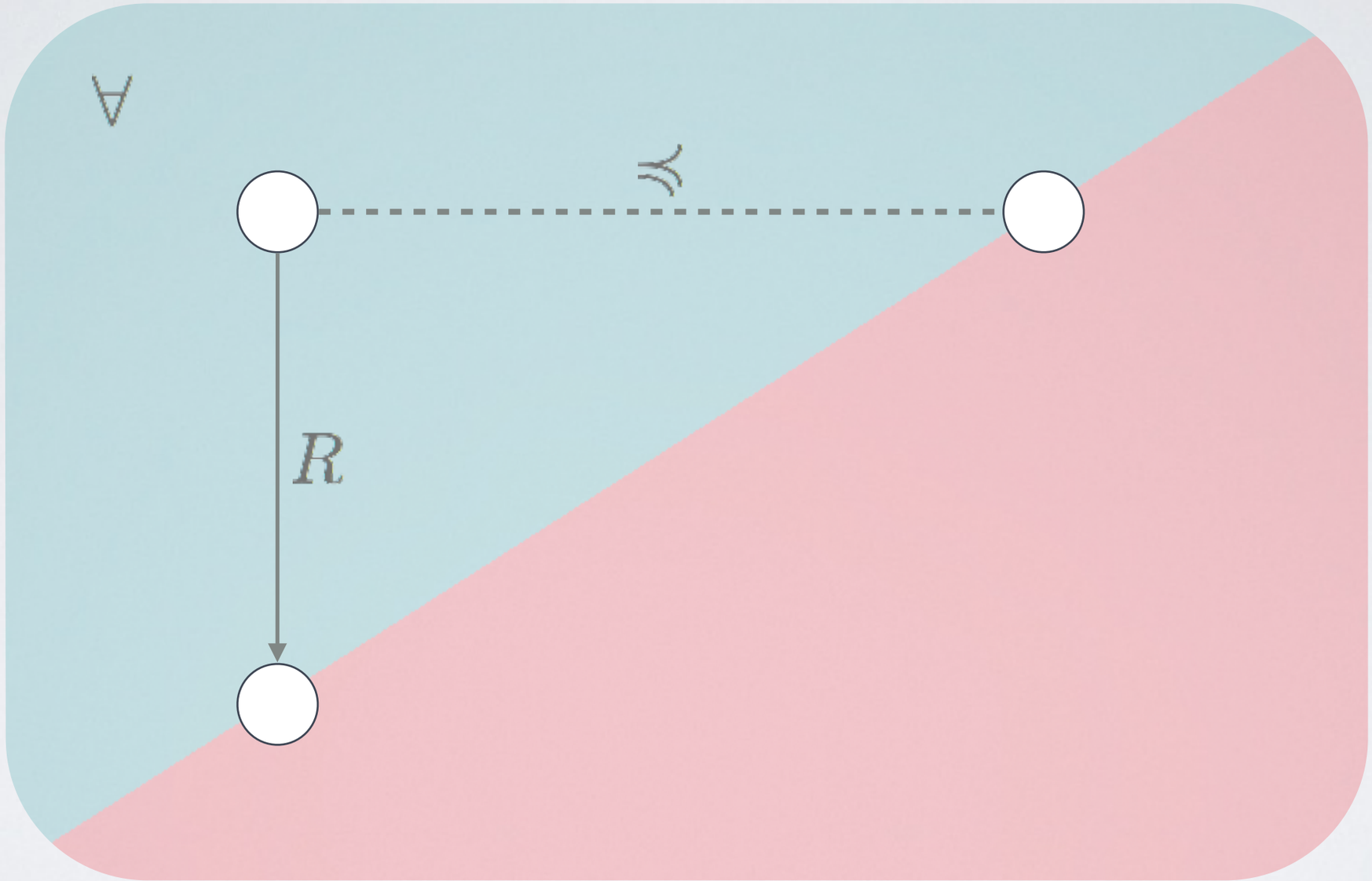
TOLERANCIA A FALLAS

Las propiedades temporales se pueden clasificar en dos grandes categorías: **Safety** y **Liveness**

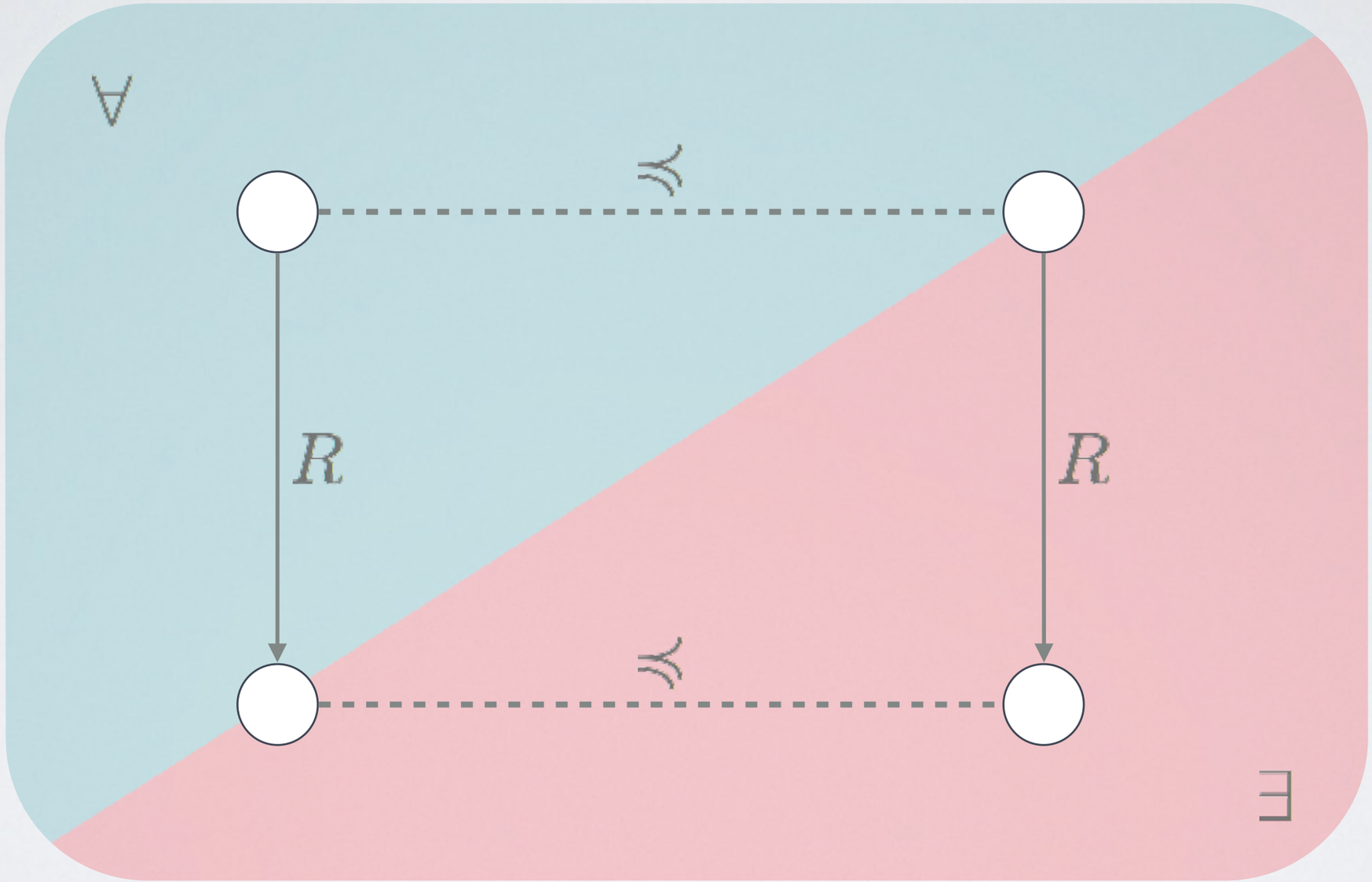
Es por esto que se diseñaron relaciones de simulación adecuadas para preservar cada tipo de propiedad.

- **Masking**: preserva safety y liveness.
- **Non-Masking**: preserva liveness
- **Failsafe**: preserva safety.

SIMULACIÓN



SIMULACIÓN



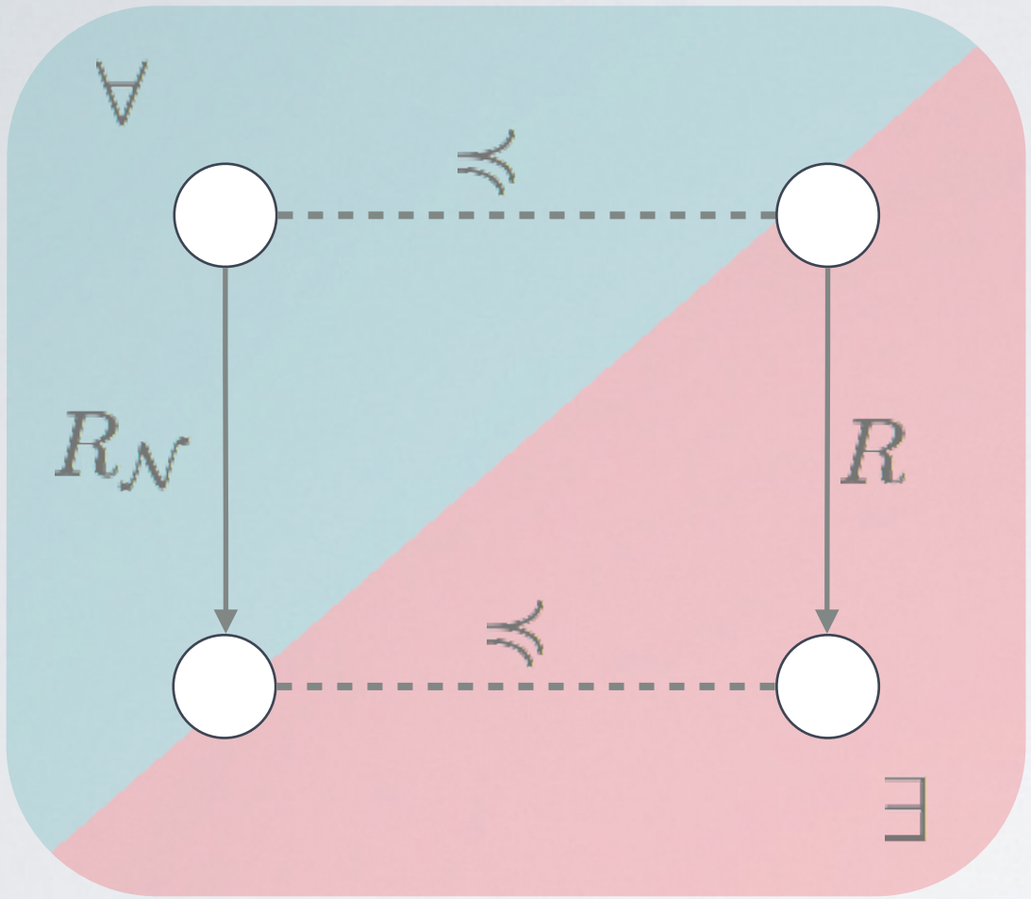
MASKING FAULT TOLERANCE

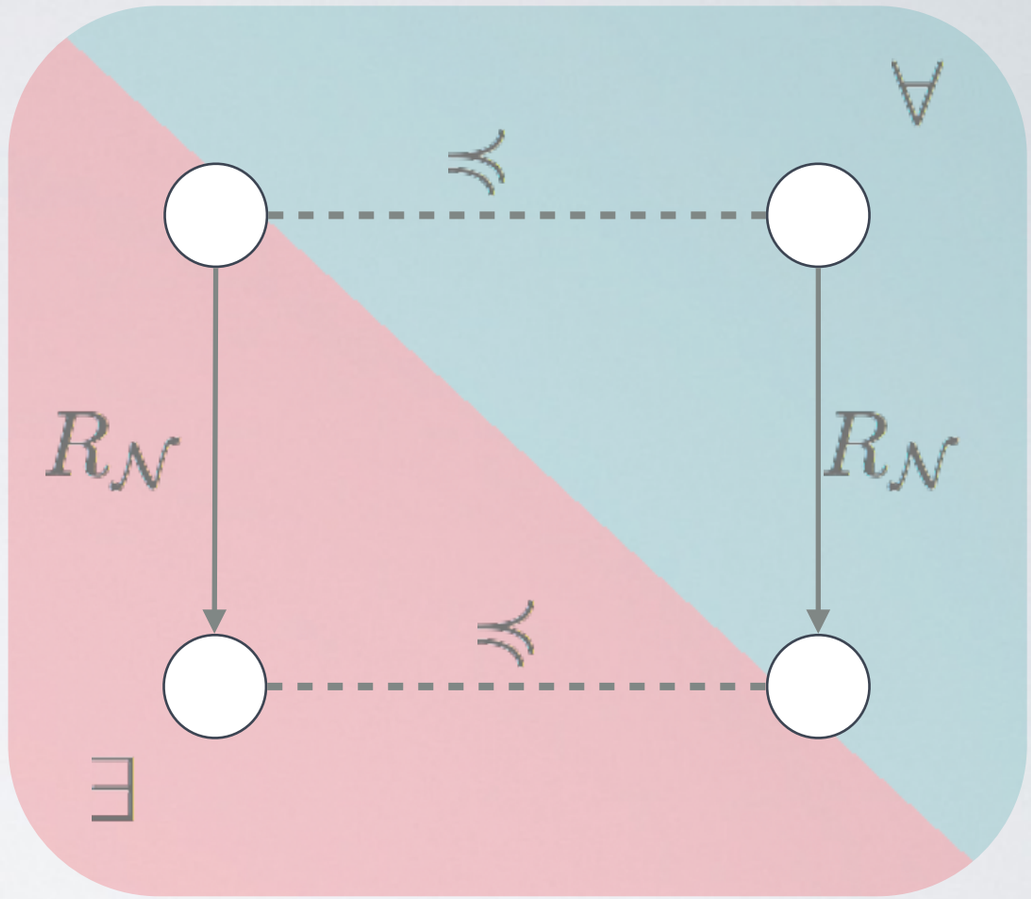
La relación de *masking* se captura la noción de que el sistema con fallas imita el comportamiento normal del sistema simulado.

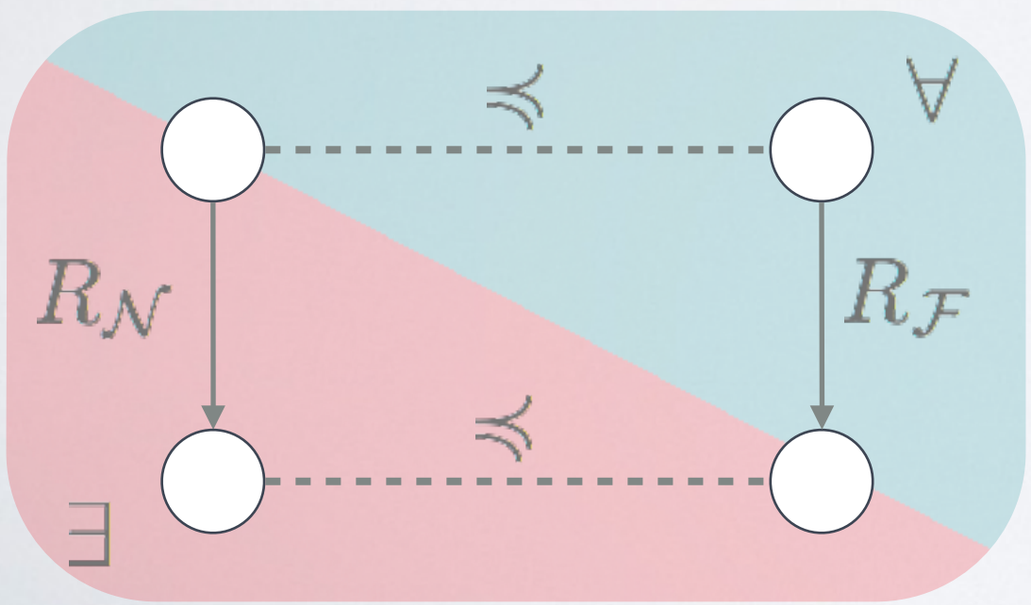
Por otro lado las transiciones fallidas deben ser "enmascaradas" por transiciones normales. Es decir, el comportamiento anormal debe ser invisible.

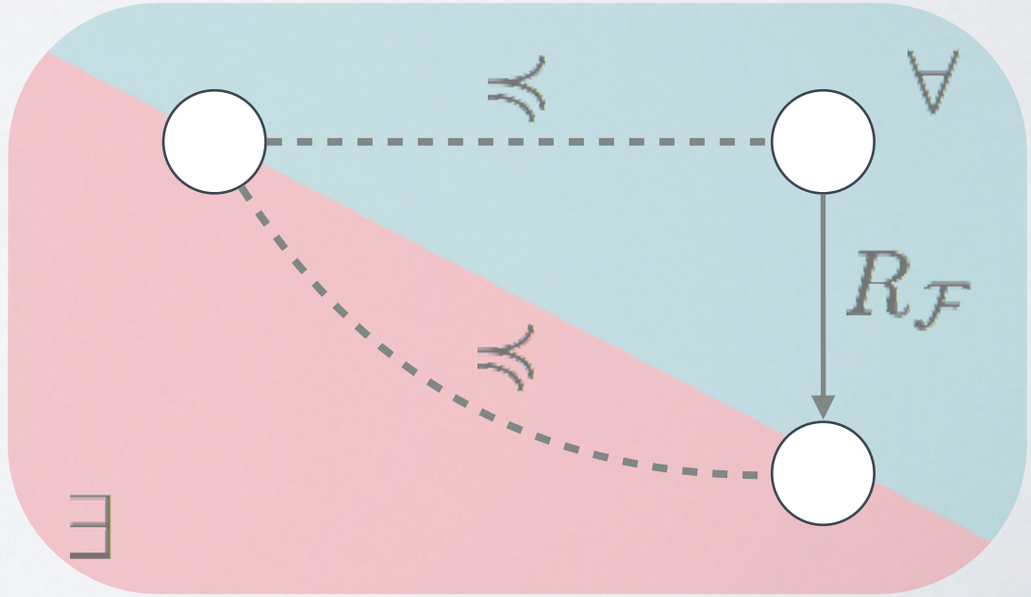
Utilizamos sistemas de transiciones etiquetadas con un conjunto de estados dividido entre estados normales y estados con fallas.

MASKING


∀
$R_{\mathcal{N}}$
R
∃


∀
$R_{\mathcal{N}}$
$R_{\mathcal{N}}$
∃

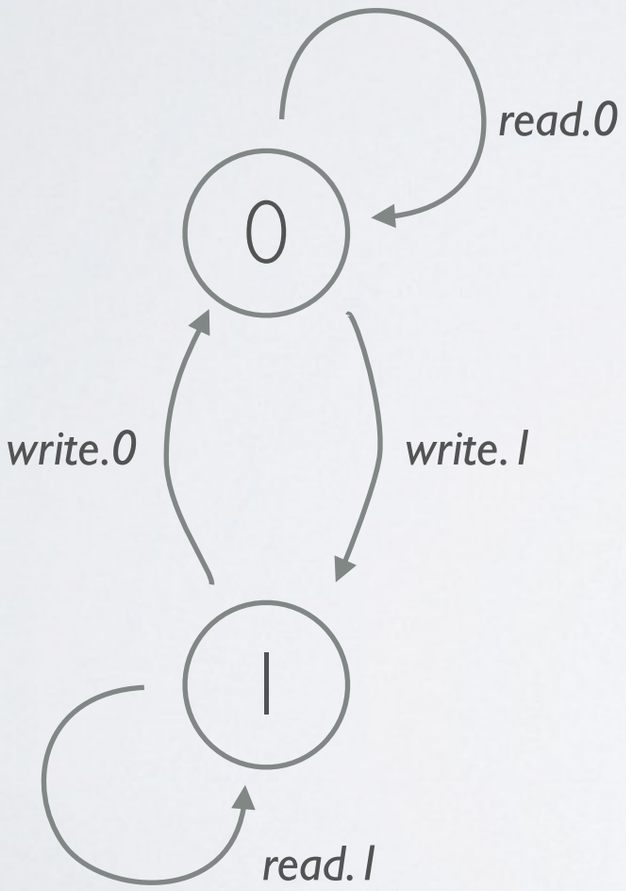

∀
$R_{\mathcal{N}}$
$R_{\mathcal{F}}$
∃

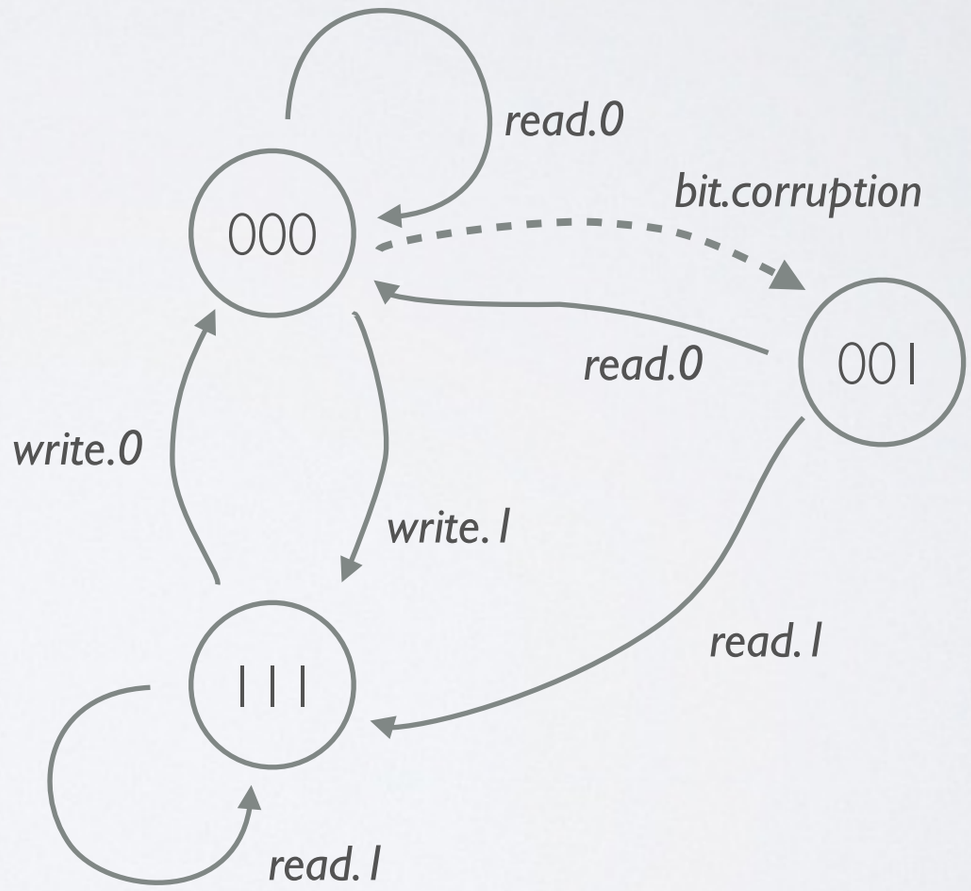

∀
$R_{\mathcal{F}}$
∃

EJEMPLO

Memory Cell

Celda de memoria correcta contra su implementación con fallas.





CONCLUSIONES Y TRABAJO FUTURO

Mediante las relaciones de *masking*, *non-masking* y *failsafe fault tolerance* es posible modelar diferentes niveles de tolerancia a fallas.

Uno de nuestros objetivos es aplicar estas nociones a síntesis de sistemas tolerantes a fallas.

Un objetivo principal es investigar como es posible migrar estas nociones al terreno probabilístico, de modo de poder ver a las fallas desde un punto de vista cuantitativo.