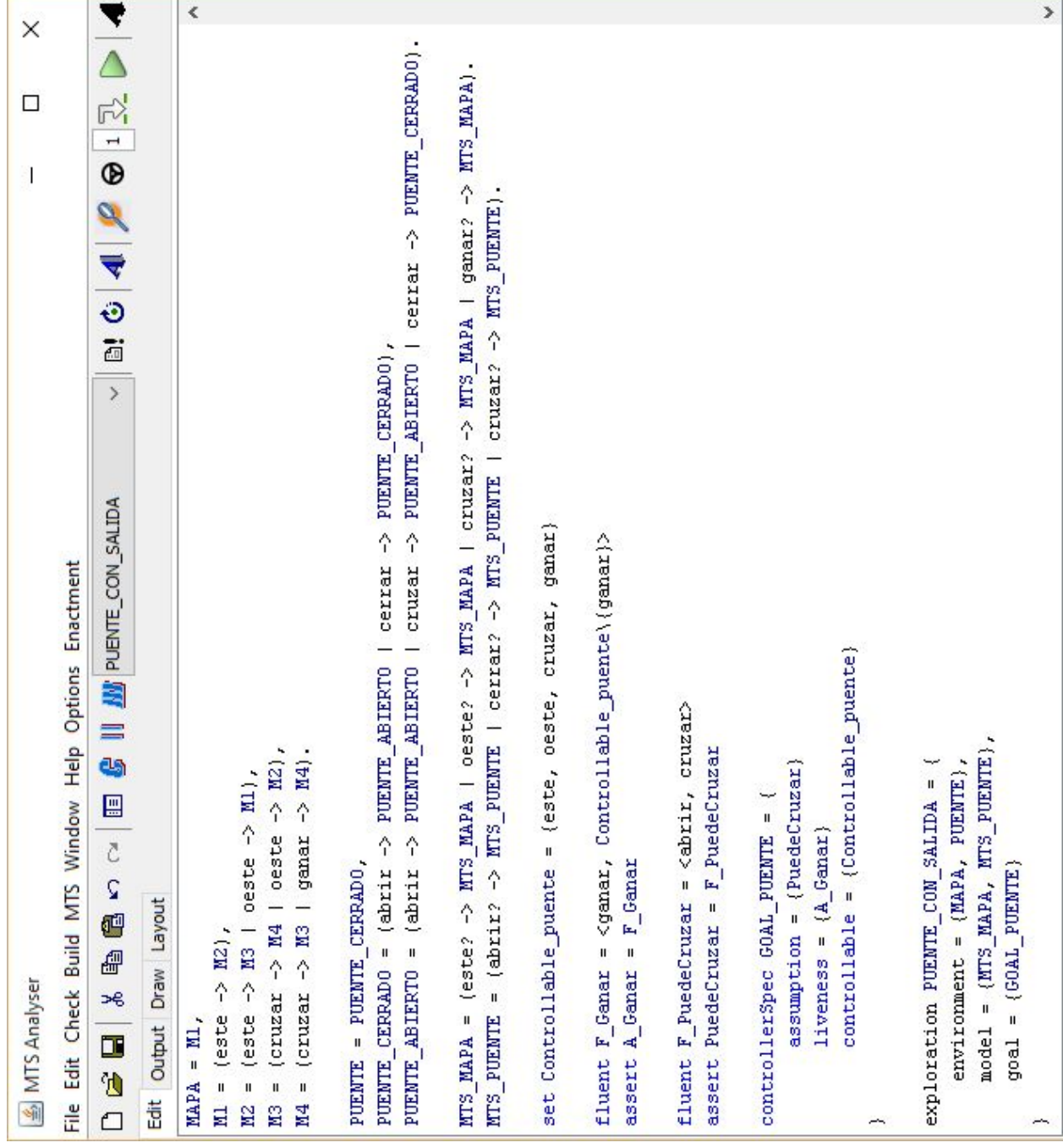


Ejemplo de exploración en MTSa



The screenshot shows the MTS Analyser application window. The title bar reads "MTS Analyser". The menu bar includes "File", "Edit", "Check", "Build", "MTS", "Window", "Help", "Options", and "Enactment". The toolbar contains icons for file operations (save, open, print), editing (undo, redo, copy, paste), and analysis (run, stop, refresh). A dropdown menu is currently set to "PUENTE_CON_SALIDA". The main text area contains the following MTS code:

```
MAPA = M1,
M1 = (este -> M2),
M2 = (este -> M3 | oeste -> M1),
M3 = {cruzar -> M4 | oeste -> M2},
M4 = {cruzar -> M3 | ganar -> M4}.

PUENTE = PUENTE_CERRADO,
PUENTE_CERRADO = {abrir -> PUENTE_ABIERTO | cerrar -> PUENTE_CERRADO},
PUENTE_ABIERTO = {abrir -> PUENTE_ABIERTO | cruzar -> PUENTE_ABIERTO | cerrar -> PUENTE_CERRADO}.

MTS_MAPA = (este? -> MTS_MAPA | oeste? -> MTS_MAPA | cruzar? -> MTS_MAPA | ganar? -> MTS_MAPA).
MTS_PUENTE = {abrir? -> MTS_PUENTE | cerrar? -> MTS_PUENTE | cruzar? -> MTS_PUENTE}.

set Controllable_puente = {este, oeste, cruzar, ganar}

fluent F_Ganar = <ganar, Controllable_puente\{ganar}>
assert A_Ganar = F_Ganar

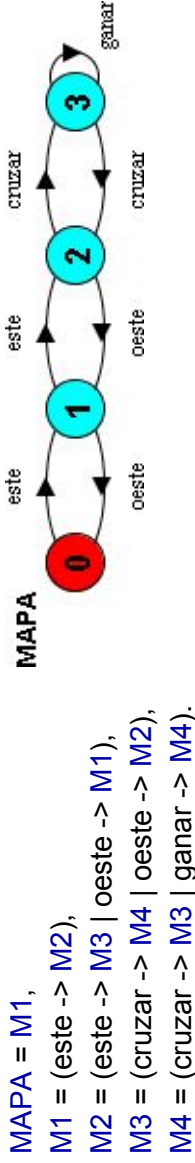
fluent F_PuedeCruzar = <abrir, cruzar>
assert PuedeCruzar = F_PuedeCruzar

controllerSpec GOAL_PUENTE = {
  assumption = {PuedeCruzar}
  liveness = {A_Ganar}
  controllable = {Controllable_puente}
}

exploration PUENTE_CON_SALIDA = {
  environment = {MAPA, PUENTE},
  model = {MTS_MAPA, MTS_PUENTE},
  goal = {GOAL_PUENTE}
}
```

Input

Mapa



En este ejemplo comenzamos en la posición 0. Nuestro robot puede moverse hacia el este o hacia el oeste entre las posiciones 0 y 2. Las posiciones 2 y 3 están separadas por un puente.

Puente



El puente comienza cerrado. Si se abre, se lo puede cruzar.

Modelos

MTS_MAPA = (este? -> MTS_MAPA | oeste? -> MTS_MAPA | cruzar? -> MTS_MAPA | ganar? -> MTS_MAPA).
MTS_PUENTE = (abrir? -> MTS_PUENTE | cerrar? -> MTS_PUENTE | cruzar? -> MTS_PUENTE).



Nuestros modelos iniciales sólo nos dicen qué acciones pueden estar disponibles en cada estado.

Objetivo

```
set Controllable_puente = {este, oeste, cruzar, ganar}

fluent F_Ganar = <ganar, Controllable_puente\{ganar}>
assert A_Ganar = F_Ganar

fluent F_PuedeCruzar = <abrir, cruzar>
assert PuedeCruzar = F_PuedeCruzar

controllerSpec GOAL_PUENTE = {
    assumption = {PuedeCruzar}
    liveness = {A_Ganar}
    controllable = {Controllable_puente}
}
```

Nuestro objetivo es ejecutar la acción ganar.

Asumimos que en algún momento se va a ejecutar la acción abrir.

Esto implica que el puente no va a quedar cerrado para siempre.

Para ganar, necesitamos movernos 3 veces hacia el este, pero la última vez hay que pasar por el puente.

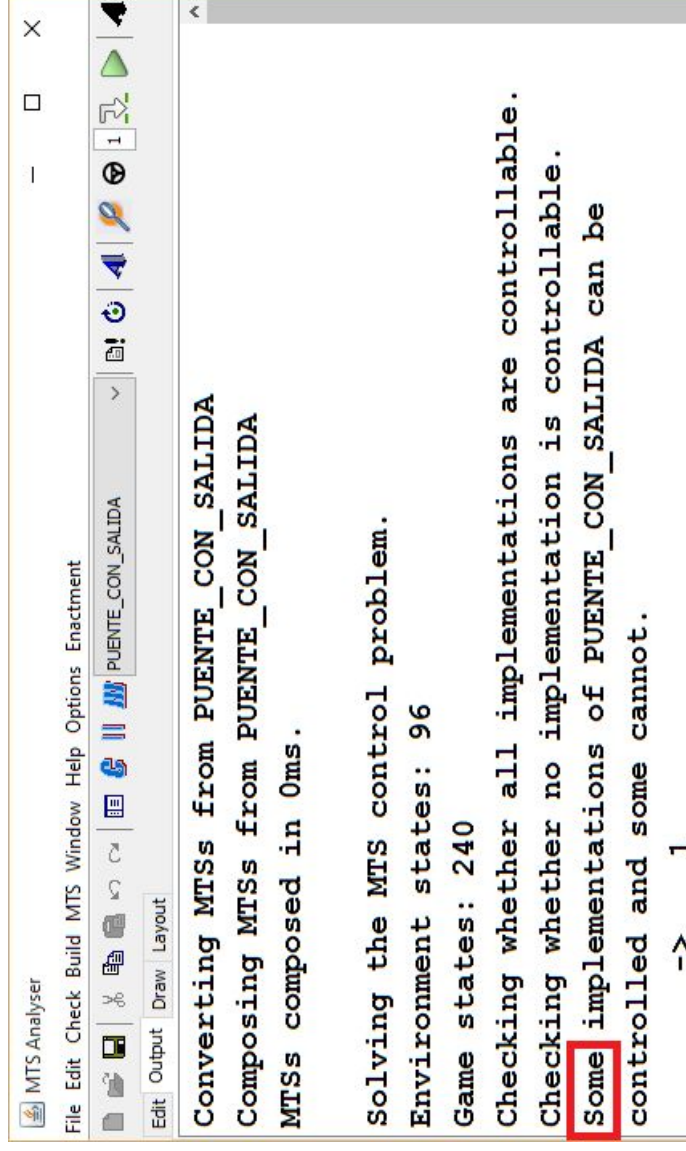
Estrategia y comportamiento del ambiente

Nuestra estrategia consiste en ejecutar siempre acciones que sean nuevas para el estado en el cual nos encontramos.

Si desde el estado actual ya ejecutamos todas las acciones disponibles, elegimos una acción que nos lleve a un estado no explorado o esperamos a que el ambiente cambie y así nos permita seguir explorando.

El comportamiento del puente será cíclico, permaneciendo cerrado 11 turnos y abierto durante 3.

Exploración



MTS Analyser
File Edit Check Build MTS Window Help Options Enactment

Output Draw Layout

Converting MTSS from PUENTE_CON_SALIDA
Composing MTSS from PUENTE_CON_SALIDA
MTSS composed in 0ms.

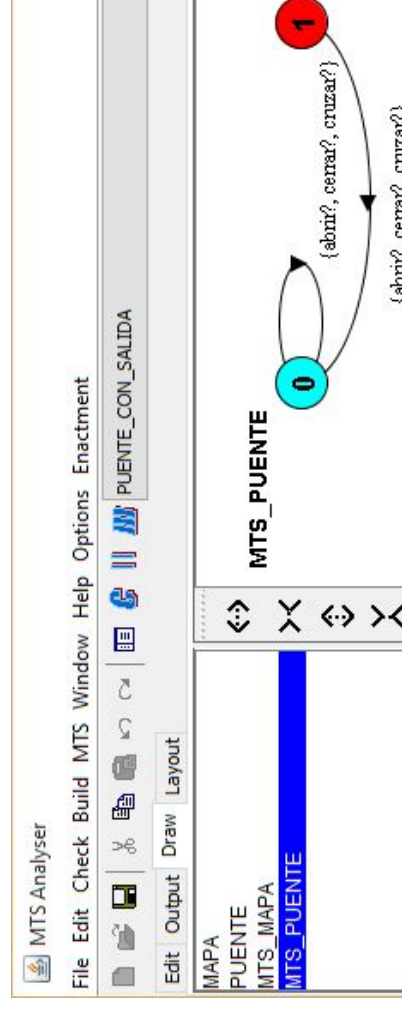
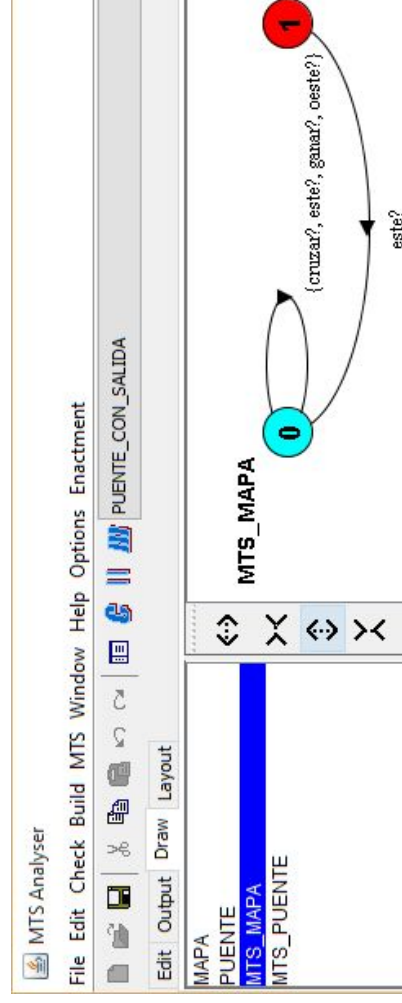
Solving the MTS control problem.
Environment states: 96
Game states: 240
Checking whether all implementations are controllable.
Checking whether no implementation is controllable.
Some implementations of PUENTE_CON_SALIDA can be controlled and some cannot.

-> 1

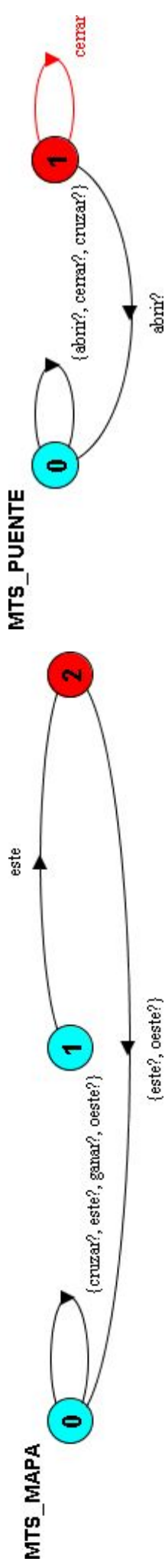
En un principio no tenemos la información suficiente para razonar sobre el objetivo.

Lo que sabemos sobre el robot es que, desde su estado actual, puede ejecutar la acción este. No sabemos a qué estado llegamos si ejecutamos dicha acción.

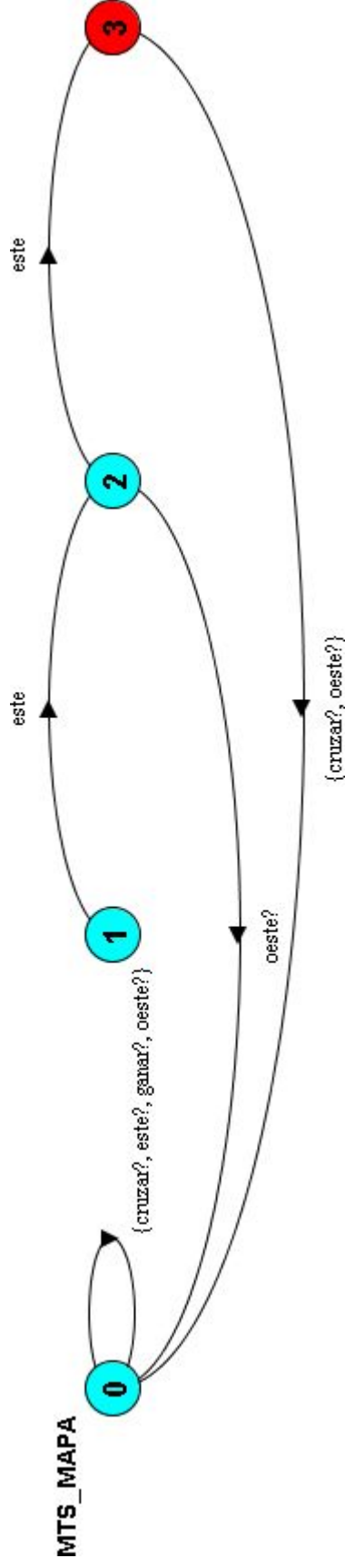
Sobre el puente sabemos que, desde su estado actual, podría tener disponibles las acciones abrir y cerrar y cruzar.



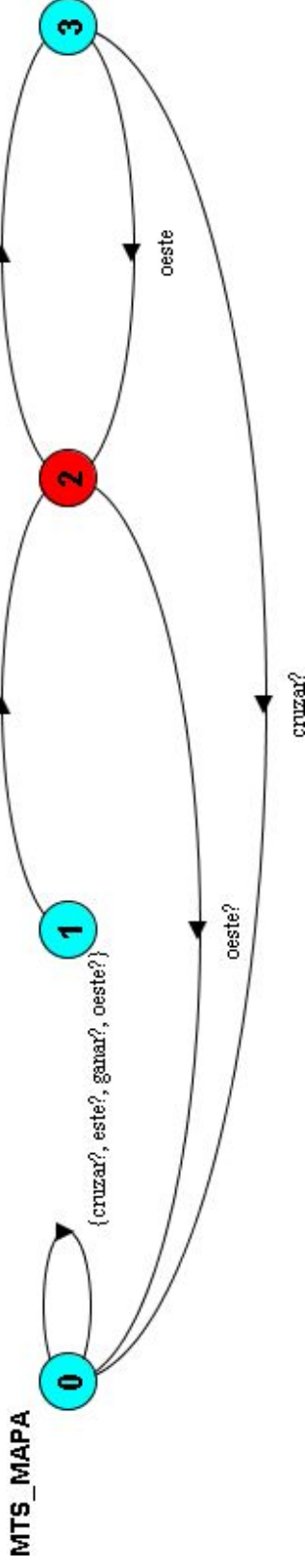
- > 1 La estrategia elige la acción este. Al ejecutarla aprendemos que desde el estado inicial, por este, llegamos a un estado
- este desde el cual podemos ejecutar las acciones este y oeste, sin saber a que estados nos llevan dichas acciones.
- cerrar El puente sigue cerrado. Aprendemos que desde el estado inicial del puente, por cerrar, llega al mismo estado.



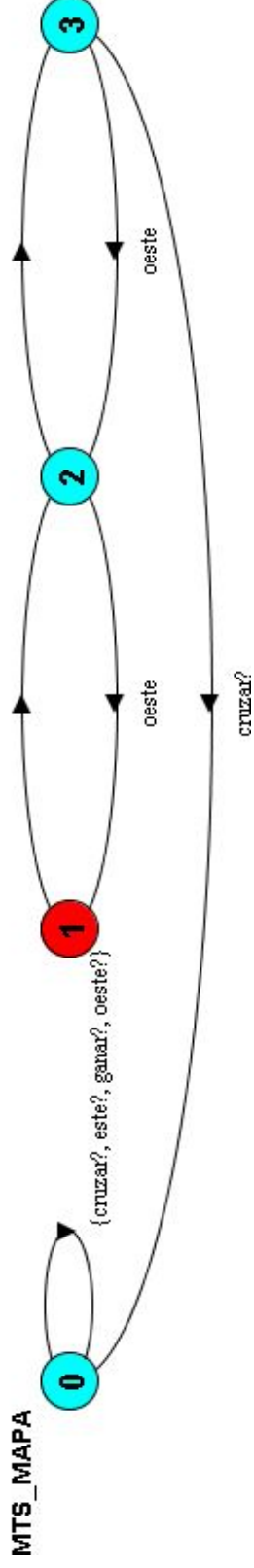
- este -> 3 Ahora la estrategia tiene disponibles para elegir las acciones este y oeste. Ambas son acciones no ejecutadas desde el estado actual. Elige la acción este.
- cerrar Al ejecutarla llegamos a un nuevo estado desde el cual podemos ejecutar las acciones cruzar y oeste. El puente sigue cerrado. Como ya había ejecutado la acción cerrar desde su estado actual no aprendemos nada nuevo.



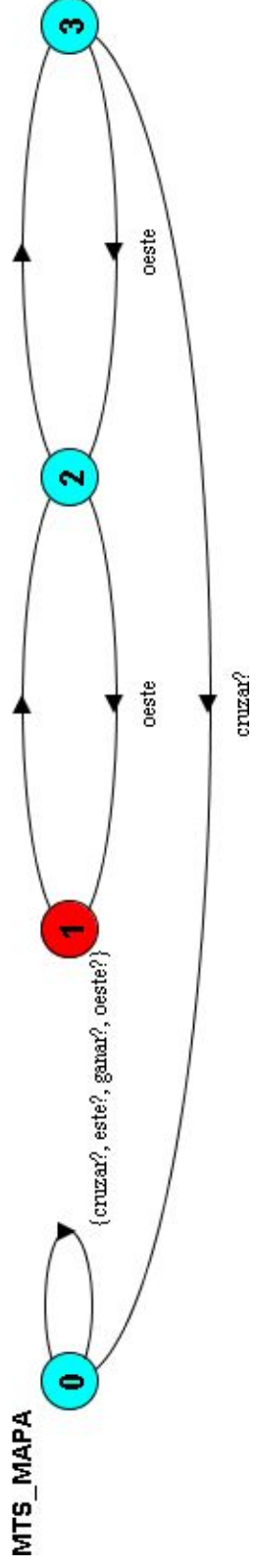
oeste -> **2** Cruzar es una acción compartida. Como no puede ser ejecutada desde el estado actual del puente, no está disponible.
cerrar Aprendemos que por oeste, desde el estado 3 volvemos al estado 2. El puente sigue cerrado.



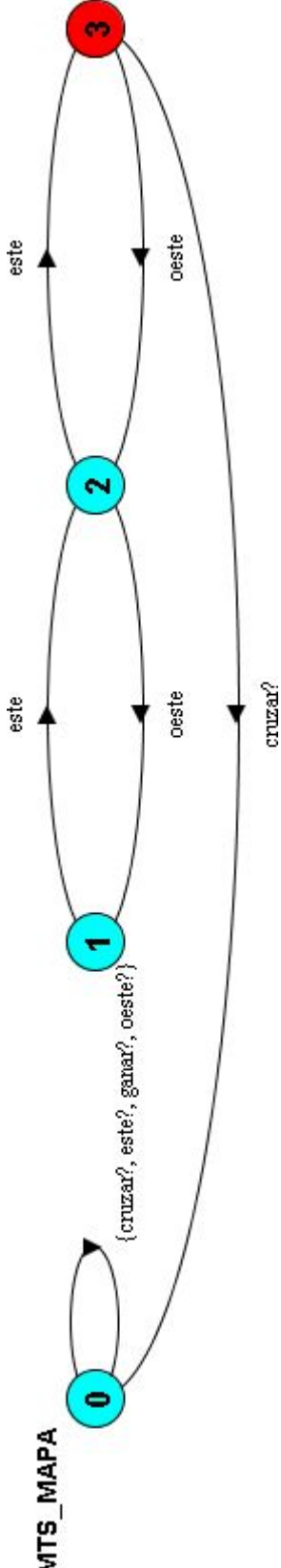
oeste -> **1** Las acciones este y oeste están disponibles. Como desde el estado actual ya habíamos ejecutado este, la estrategia elige oeste. Aprendemos que por oeste, desde el estado 2 volvemos al estado 1. El puente sigue cerrado.



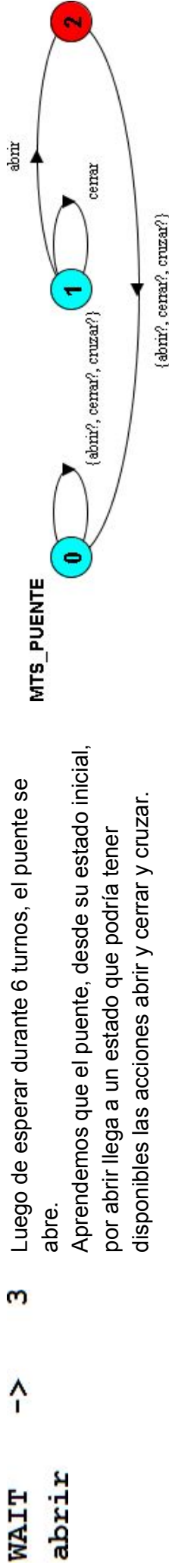
oeste -> **1** Las acciones este y oeste están disponibles. Como desde el estado actual ya habíamos ejecutado este, la estrategia elige oeste. Aprendemos que por oeste, desde el estado 2 volvemos al estado 1. El puente sigue cerrado.



este	->	2	Ahora no tenemos disponibles acciones que no hayan sido ejecutadas anteriormente desde el estado actual. La estrategia busca llegar a un estado que tenga acciones no ejecutadas, por eso ejecuta dos veces este para llegar al estado 3.
cerrar			
este	->	3	El puente sigue cerrado.
cerrar			



WAIT	->	3	La única acción que nos permite seguir adquiriendo conocimiento, cruzar, es una acción compartida. No podemos ejecutarla porque el puente no permite ejecutarla desde su estado actual, así que esperamos a que el puente cambie.
cerrar			El modelo de conocimiento del robot no cambia, ya que no se ejecutó ninguna acción. El puente sigue cerrado.



cruzar	->	4	Cuando finalmente el puente se abre, el robot puede cruzar.
abrir			
cruzar	->	3	El robot logra llegar a un estado desde el cual se puede ejecutar la acción ganar.
abrir			La información obtenida es suficiente para garantizar el cumplimiento del objetivo.
cruzar	->	4	
cerrar			
ganar	->	4	
cerrar			

All implementations can be controlled