Labelled Transition Systems for Agents' Behaviour

Annalisa Paladino, Emanuele Ruoppolo

Introduction

In this work we aim to present a solution to the *Prison Puzzle* by using Labeled Transition Systems (**LTSs**) to model agents' behavior. The problem consists in modeling, using LTSs, a protocol that $N = 3$ prisoners must agree upon in order to be freed from detention. This protocol will precisely describe how the prisoners must behave.

After establishing the protocol, no further communication is allowed. The prisoners will be taken in turn from their cells to an empty room with a light switch, according to a random but fair scheduler. Using only this light switch, they must find a way to communicate. One of the prisoners will have a counter ranging from 0 to n to count the visits, where n is selected before the whole process begins. Once they are certain that all three of them have been in the room at least once, they can inform the guards and be freed.

Protocol

To solve the puzzle we propose a *Counter-Signalers* protocol. The prisoner with the counter, that here we define as (**P1**), will act as the Counter, while the other two prisoners (**P2**, **P3**) will act as Signalers. Since the Signalers act in the same way, to avoid redundancy, we will model only P2's behavior.

The protocol we propose is:

Signalers: On the first visit they found the light off, they turn it on to signal that they have visited. After signaling once, they do nothing further.

Counter: When the light is on, he turns it off and increment the counter. When the counter reaches 2, he declares "end".

This protocol requires the counter maximum to be set to $n = N - 1 = 2$. Indeed, the Counter's duty is to ensure that the other two prisoners have visited the room at least once by counting how many times he finds the light turned on. Since he needs to enter the room to check the state of the light, he does not count his own visits but only counts up to the total number of Signalers, which is two.

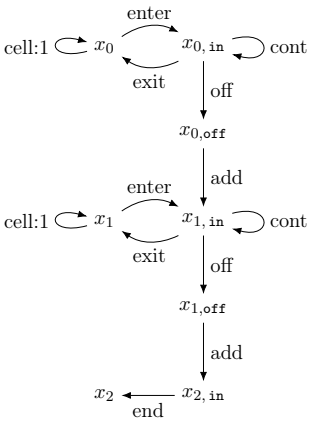
LTSs

According to the solution, we propose P1's LTS with the following states and actions:

$$S = \{x_0, x_1, x_2, x_{0,\texttt{in}}, x_{1,\texttt{in}}, x_{2,\texttt{in}}, x_{0,\texttt{off}}, x_{1,\texttt{off}}\}$$

$$A = \{\texttt{cell:1, enter, exit, cont, off, add, end}\}$$

It can be represented with its transitions as follows:



States x_0 and x_1 represent the Counter waiting in its cell. From either state, he can perform the `cell:1` action to synchronize with the scheduler and begin his turn. Although the LTS could seem redundant we considered as crucial the distinction between the states: x_0 is the initial state (zero signals counted), while x_1 is the state after one signal has been counted.

Once his turn begins, the Counter enters the room and:

If the light is on: it turns the light off and increments its internal counter.

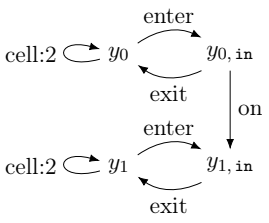
If the light is off: it performs the `cont` action followed by the `exit` action, leaving the room and returning to its cell without changing its count.

Since the goal is to count to $n = 2$, after incrementing its counter for the second time, the Counter performs the `end` action to reach the terminal state x_2.

The LTS for prisoner P2 is described below (P3's is analogous, but uses states z_i and the action `cell:3`).

$$S = \{y_0, y_1, y_{0,\texttt{in}}, y_{1,\texttt{in}}\}$$

$$A = \{\texttt{cell:2, enter, exit, on}\}$$



Similar to the Counter, P2 begins in his cell (in state y_0) waiting for his turn. Once scheduled, he enters the room, transitioning to state $y_{0,\texttt{in}}$. His primary task is to turn the light on, but only the very first time he enters and finds the light off.

After this initial check, his behavior changes. On all subsequent turns, he will simply enter and exit the room as scheduled, without interacting with the light, until the Counter performs the `end` action.

Temporal properties

We can express the property *eventually the prisoners give the correct solution* using the following recursive temporal formula:

$$X \overset{\min}{=} \langle \text{end} \rangle \text{T} \vee \Diamond X$$

This formula can be read as: "X is the set of states from which either the `end` action is immediately possible, or it is possible to perform some other actions to reach a new state that has the same properties". Here, X represents the set of states from which it is possible to eventually give the correct solution. The components of the formula are:

- The use of $\overset{\min}{=}$ expresses **reachability**, ensuring that the property must be eventually (in a finite number of steps) satisfied.
- $\langle \text{end} \rangle \text{T}$ is the base case, describing a state from which it is **possible** to take an `end` transition.
- $\Diamond X$ is the recursive step, describing a state from which it is possible to take **any** transition to a new state that, in turn, satisfies the equation.

The initial state of the composed system is:

$$r_0 \,|\, t_0 \,|\, c_0 \,|\, x_0 \,|\, y_0 \,|\, z_0$$

The property holds on this initial state primarily because the **fair scheduler** guarantees that every prisoner will eventually be selected to visit the room. The protocol then ensures progress towards the solution:

1. **A Signaler (e.g. P2) visits:** The fair scheduler will eventually select P2. Since the light is off (r_0), P2 will `enter`, turn the light `on`, and `exit`. He is now in state y_1 and will no longer interact with the switch. The system state evolves to one where the light is on

$$r_1 \,|\, t_2 \,|\, c_0 \,|\, x_0 \,|\, y_1 \,|\, z_0$$

2. **The Counter (P1) visits:** The scheduler will eventually select P1. He will find the light on (r_1), turn it `off`, increment his counter to c_1, and change his internal state to x_1. Following his exit, the light is now reset for the next signaler and the system state will be similar to

$$r_0 \,|\, t_1 \,|\, c_1 \,|\, x_1 \,|\, y_1 \,|\, z_0$$

3. **The second Signaler (P3) visits:** Eventually, P3 is selected. He will find the light off, turn it `on`, and move to his final state z_1. The system state will be similar to

$$r_1 \,|\, t_0 \,|\, c_1 \,|\, x_1 \,|\, y_1 \,|\, z_1$$

4. **The Counter (P1) visits a final time:** The scheduler will eventually select P1 again. He will find the light on, turn it `off`, and increment his counter to its final value c_2. This action moves P1 into his local state $x_{2,\text{in}}$. The global system is now in a state like

$$\dots \mid c_2 \mid x_{2,\text{in}} \mid \dots$$

From this specific state, the `end` transition is enabled for P1. Therefore, this state satisfies the base case of our formula, $\langle \text{end} \rangle \text{T}$. Because we have demonstrated a guaranteed (due to the scheduler fairness) path of transitions from the initial state to a state where the property holds, we proved that the property holds on the initial state of the composed system.

Unknown Initial Room State

If the initial state of the light is unknown, a more robust protocol is required to guarantee a correct solution. The revised protocol is as follows:

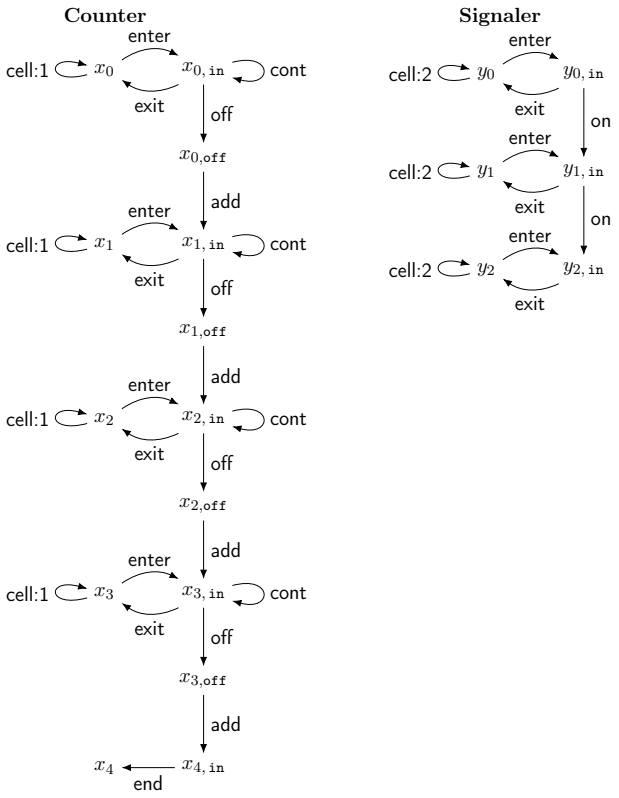
Signalers: Each signaler must turn the light on exactly twice. On any visit, if the light is off and they have not yet used both of their signals, they must turn it on. After turning the light on for the second time, they do nothing on subsequent visits.

Counter: On any visit, if the light is on, he turns it off and increments his counter. When the counter reaches 4, he is certain that both other prisoners have visited at least once and declares `end`.

The counter value is set to $n = 2\,(N-1) = 4$ because each of the two signalers must provide two distinct signals. The core idea is that the Counter's first visit serves as a **reset phase**. By turning the light off, he forces the system into a known state (light off), effectively neutralizing the initial uncertainty. We can analyze the two possible initial scenarios to prove this protocol is sound:

- **Light is initially ON:** The Counter's first visit is guaranteed by the fair scheduler. He will find the light on, turn it off, and increment his counter to 1. At this point, the room is in a known state (light off), and the Counter needs 3 more signals to reach his target. Since the two signalers will provide a total of 4 `ON` signals throughout the process, these 3 required signals are guaranteed to occur, eventually allowing the Counter to declare `end`.
- **Light is initially OFF:** In this scenario, the Counter will only increment his count when a signaler has turned the light on. To reach the target of 4, he requires 4 distinct `ON` signals. The protocol is designed for the two signalers to provide exactly 4 signals in total (two each). Therefore, once the counter reaches 4, the Counter is guaranteed that every signal from both prisoners has been registered, which implies that both have visited.

The LTSs keep the same initial logic, but need then to be expanded to account for the ulterior actions both the Counter and the Signaler must perform.



Extra: Simulation

Outside the requirements, to validate our protocols, we simulated two scenarios: one with a known initial state and one with an unknown initial state. The number of prisoners (N) was configurable, with the protocol's counter set to $n = N-1$ for the known case and $n = 2(N-1)$ for the unknown case. Each scenario was simulated 1000 times using a uniformly random scheduler to gather statistics.

The results are visually presented in the image below. They confirmed that both protocols correctly terminate in a finite number of steps. As expected, knowing the initial state leads to a faster solution; this protocol was $\approx 1.6\times$ faster on average, requiring 14 ± 5 iterations versus 22 ± 6 for the unknown-state case. The simulation and its results can be found on the project GitHub repository.

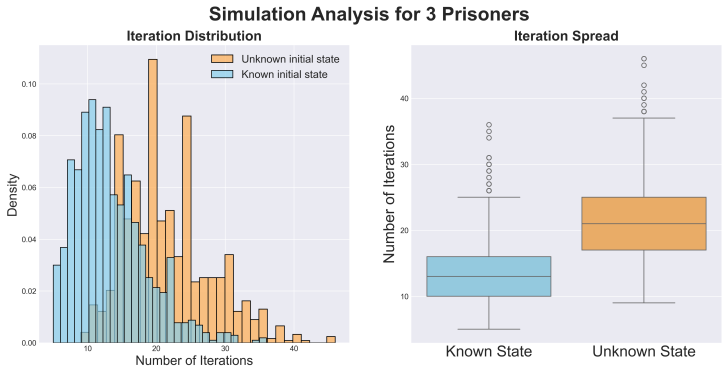


Figure 1: Results from the simulation of the two proposed protocols