1 Background

We will start with the theoretical background and then will formulate the solution using epistemic logic.

1.1 Kripke Model

Kripke models are used to represent the state of information in a multi-agent environment. It is defined as follows:
Given a finite set of agents N and a finite set of atoms P, a Kripke model is defined as a tuple $M = (S, R, V)$, where:
- S: a non-empty set of states.
- R: a set of accessibility relations for each agent. Defined as $N \to (SXS)$
- V: assigns values to atoms in each state.

1.2 Update Model

Update models are used to model the change in information. We define events that change information the same way as we define states in Kripke model. The domain of the model is defined as the set of non distinguishable events by the agents. Every event has a precondition.
For a given finite set of agents N and a language L, we define the update model as a tuple $U = (E, R, pre, post)$:
- E: A finite set of events.
- R: The set of accessibility relations for each agent. Defined as $N \to (EXE)$.
- pre: assigns a precondition to each event. $(E \to L)$.
- $post$: assigns a postcondition to each event for each atom. $E \to (P \to L)$.

An update is defined as a pair (U, e) for $e \in E$.

1.3 Execution of Update model

For a Kripke model $M = (S, R, V)$, an update model $U = (E, R, pre, post)$ and a language L of the model M. For an event $e \in E$ and a state $s \in S$ such that $(M, s) \models pre(e)$, the result of executing an update (U, e) in (M, S) is $((S', R', V'), (s, e))$, where:
- S': $\{(k, l)\}$, if $(M, k) \models pre(l)$, where $k \in S$ and $l \in E$.
- $R'(a)$: $\{((s, l), (s', l'))\}$ where $(s, s') \in R(a)$ and $(l, l') \in R(a)$.
- $V'(p)$: $post(l)(p)$, for $l \in E$ and $s \in S$.

2 Logic Description

In order to model the solution of the problem in terms of epistemic logic we need to define a set of atoms, an initial model and set of possible updates. For this analysis, we will focus on strategy 1.

2.1 Atoms and formulas

There are two types of agents: counters and non-counters. We define agent 0 to be a counter. The atoms are defined as follows:
- p: The light bulb is on.
- q_i for $1 <= i <= n - 1$: Means whether non-counter i has been interrogated previously, which implicitly also means whether non-counter i has turned on the light bulb.

We don't need a q_0, because if the counter is being interrogated then only the counter can make an announcement.
We define the following formulas:
- $\wedge_{i>0} q_i$: All non-counters have been interrogated.
- $K_0 \wedge_{i>0} q_i$: The counter knows that all prisoners have been interrogated.

2.2 Initial Model

The initial model has a single state in which all atoms $p, q_1 ... q_{n-1}$ are false. This is accessible by the counter, as it represents the initial condition before interrogation.

event	pre	post
e_i	if $\top$	then $p = q_i \rightarrow p$ and $q_i = p \rightarrow q_i$
$e_0^{\neg p}$	if $\neg p$	then ϵ
e_0^p	if p	then p = $\perp$
e_ϕ	if $\top$	then ϵ

Table 1: Description of events in Update model.

2.3 Update Model

The events of the update model are described in table 1. The informal description is as follows:
- e_i: When non-counter i is being interrogated and if the light bulb is on, then do not do anything. If the light bulb is off, then turn it on if you have not done so before, else do nothing.
- e_0^p: For the counter, if the light bulb is on turn it off.
- $e_0^{\neg p}$: For the counter, if the light bulb is off then turn it on.
- e_ϕ: Nothing happens.

The counter can only distinguish events involving himself from events that do not involve himself (and therefore involve non-counters). He cannot distinguish among events not involving himself, so there exists a relation between these events for the counter. We show the update model U for a case of 3 prisoners with 1 counter and 2 non-counters in figure 1.

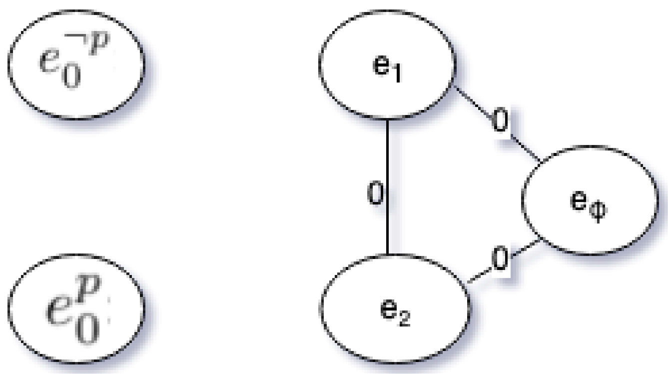

Figure 1: Update model for 3 prisoners.

2.4 Execution of Strategy 1

We make non deterministic choices out of the events $e \in E$ of update model U and execute the (U, e) on the initial model, until we reach the termination condition $K_0 \wedge_{i>0} q_i$. Repeated execution of updates until termination is shown in Figure 2.

2.5 Execution of Strategy 2

In strategy 2, every prisoner can count, so we have to model the knowledge of every prisoner. This only affects the termination condition, which will be $\vee_{j=0}^{n-1} K_j \wedge_{i>0} q_i$.

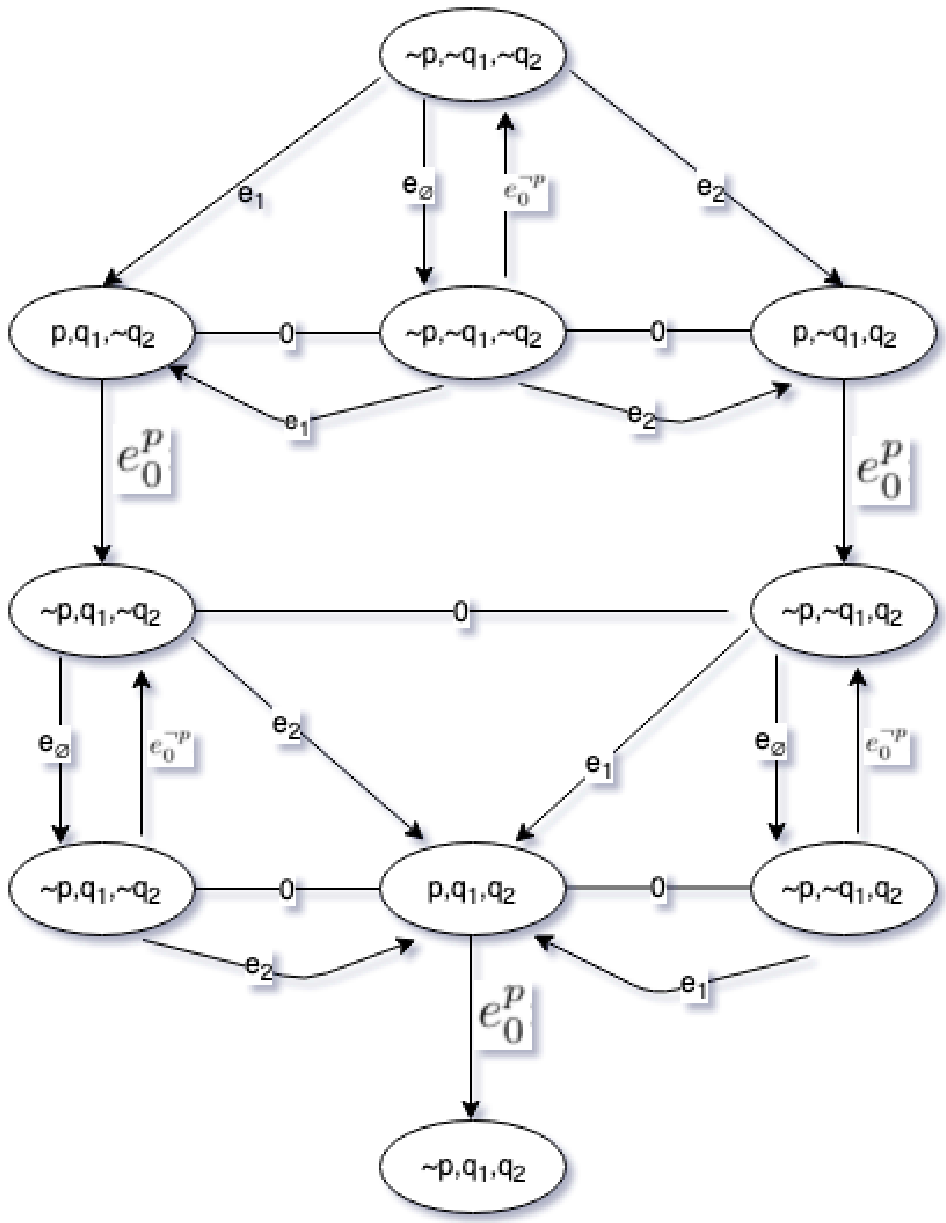

Figure 2: Execution of actions for three prisoners.