Discussion on the Muddy Children Problem

Sowmya S Sundaram
Computer Science and Engineering
Indian Institute of Technology
Madras, Chennai 600 036
Email: sowmyassundaram@gmail.com

***Abstract*—The purpose of this document is to describe the Muddy Children problem and the use of epistemic logic to solve it. It also aims to describe an implementation of the same.**

I. Introduction

The Muddy Children problem is formulated as follows. There are *N* children playing in a field. Some of their foreheads have become muddy. Each child can see the foreheads of the other children, but not her own. The father announces that atleast one of the children is muddy. After that, he asks the children if they know whether they are muddy or not. All the children are intelligent. How many times do the children have to be asked the same question before they are sure whether they are muddy or not?

It can be proved that given k children are muddy, then when the father asks the child for the kth time, they will be sure whether they are muddy or not.

A. Example

Consider the situation where there are three children A, B and C. A and B are muddy while C is not. When the father announces that atleast one of you is muddy, all three children come to the conclusion that all three of them cannot be clean. The three of them are not able to come to any conclusion. After the father asks the question whether they know if they are muddy or not, all three of the answer that they do not know. After that, A begins her reasoning. If she had not been muddy, B would've seen that A and C are clean and concluded that B herself was muddy. Since B said that she doesn't know the answer, it follows that A is muddy. B follows a similar reasoning and is sure that she is muddy. Now C still has no way of knowing whether A or B is muddy. However in the next round, when A and B answer "Yes", C knows that they could not have come to that conclusion if she herself was muddy. Hence C also knows the answer that she is not muddy.

II. Possible Worlds Model

A problem exists because there are options. In some scenarios, given a question, an intelligent agent lists out the possible solutions and eliminates them gradually by a process of sound reasoning until it arrives at the correct answer. A similar idea is captured by the possible worlds model.

In the above example, A had two possible worlds - (A-muddy, B-muddy, C-clean) and (A-clean, B-muddy, C-clean). After some reasoning, she was able to eliminate the second world and zero in on the first world. Consider another case when you are inside a closed room and it is either sunny or rainy outside. When you come out, you will be able to find the "true" world. Here, reasoning is not possible.

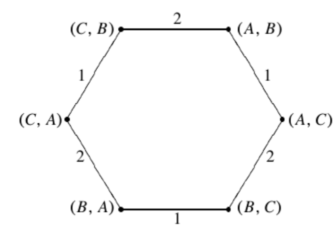

Fig. 1. Kripke Structure for Card Game [1]

As this is a discussion on what an agent knows about the world and the knowledge of other agents, we turn to a different form of logic known as modal logic.

The operator K_i for every i^{th} agent is known as the modal operator. The semantics is as follows. $K_i p$ means that agent i knows that the proposition p is true. Similarly, $K_1 K_2 p$ means that agent 1 knows that agent 2 knows p. Another example is $K_1 \neg K_2 p$ which means agent 1 knows that agent 2 does not know that p is true.

III. Kripke Structures

A Kripke Structure is a formal representation of the Possible Worlds Model. There is a fixed set of propositions p in the world. Let the Kripke Structure be denoted by M. It is a tuple (S, π, $\kappa_1, \kappa_2, ..., \kappa_n$). Here S is a set of states, which is equivalent to worlds. In each state s, the p propositions have a particular configuration of truth values defined by $\pi(s)$. Here κ_i is a binary relation defined over states. If (s,t) is related through κ_i, it means that the agent i cannot distinguish between the states s and t.

Given a Kripke structure M and the true world s, we can perform meaningful reasoning. Given (M,s) we can answer questions like $K_i p$? The Kripke structure can be described as a graph. The nodes represent the states and the edges represent the relation κ.

A. Example

Consider two agents 1 and 2 playing a game of cards. There are only 3 cards A, B and C.

The possible worlds are the permutations of the three cards with two people. Here each vertex is represented as an ordered pair (x,y) where agent 1 is holding card x and

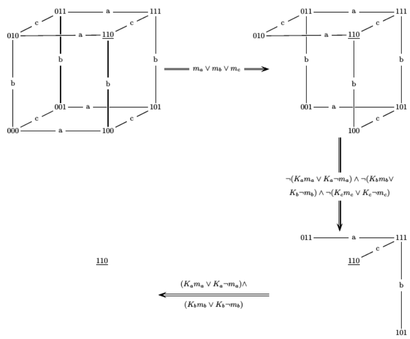

Fig. 2. Kripke Structure for Card Game [2]

agent 2 is holding card y. When agent 2 is holding a B, it cannot distinguish between (A,B) and (C,B). This is the edge represented at the top of the figure. This is the Kripke structure. Given that the true world is (C,B), the query $K_1(1C)$, i.e. whether agent 1 knows that 1 is holding the card C, can be answered. From (C,B), the worlds accessible by 1 are (C,B) and (C,A). In both of these worlds, agent 1 has C, so it knows it's holding a C. Similarly, $K_1 \neg K_2(1C)$ is also true. That is, agent 1 knows that agent 2 doesn't know that agent 1 is holding a C. This can be reasoned from the Kripke structure as follows. From (C,B), there are only two worlds accessible by 1, (C,B) and (C,A).

At (C,B), the 2 accessible worlds are (C,B) and (A,B). Since 1C is true in the first state and false in second state, agent 2 does not know 1C.

At (C,A), the 2 accessible worlds are (C,A) and (B,A). Similarly in this state also 2 does not know 1C.

Hence in both worlds agent 1 can reason that agent 2 does not know 1C. Therefore, agent 1 knows that agent 2 doesn't know 1C.

IV. Kripke Structures and Muddy Children Problem

Initially, the Kripke structure for the muddy children problem is described below. A binary encoding is used to represent the state. For example, 011 denotes that the first child is not muddy, second and third children are muddy.

Let us describe our first example where the true world can be encoded as 110. After the father's public announcement, agents a, b and c no longer consider the world 000 possible and the node is removed. Here, from the true world 110, there are a-accessible worlds, b-accessible worlds and c-accessible worlds. Hence, there is no way a,b and c can distinguish the true world from the rest. So it follows they will all answer no. After that, take the world 010 for example. If this was the true world, as there are no b-accessible worlds, b would've answered yes. Since b answered that she doesn't know, 010 is removed. Similarly, 010, 001 and 100 are removed. Now the true world 110 have no a-accessible or b-accessible worlds. Hence, a and b know they are muddy and answer yes. Since c knows that at 111 there are a-accessible and b-accessible worlds and a and b could not have come to conclusion, c concludes 110 is the true world and the process ends.

V. Implementation

The implementation is done in Java. The logic of Kripke Structures is captured by strings representing the worlds.

Input: The number of children and the true world. **Output**: The possible worlds and the answers given by the children at every round.

Algorithm:

- Create all possible worlds and store them as strings depending on number of children. The set would be (000,001,010,100,...,111) for 3 children.
- For each agent (modelled as a class), apart from all possible worlds, maintain a list of worlds with relation to the true world. For example, if the true world is 110, agent A will maintain 010 and 110 as it cannot distinguish between them.
- After the public announcement, remove 000 (in this example) from possible worlds. Every time a world is removed, it is checked if it is in any of the agent specific worlds. If so, it is removed from that as well.
- Repeat until all agents have answered they know whether they are muddy or not.
  - Check if the worlds the agent considers possible with respect to true world is just one. In that case, display that the agent knows the answer.
  - Communicate the answer to all other agents through a method.
  - When an agent i receives an answer, it checks whether some other agent j knew it was muddy or not.
  - **Case 1:** If the agent j did not know it was muddy, iterate over all the worlds agent i considers possible. Flip the bit in the j^{th} position and create a temporary world. If this world is not in the list of possible worlds, that means the current world is also not possible.
    Example: Consider agent 2 did not know the answer and agent 1 was informed of that. While iterating through the list of possible worlds, when it is considering 010, it will create a temporary state 000 by flipping the second bit. Since 000 was removed in the previous step, if 010 was indeed the true world, agent 2 would've answered yes. As it didn't, it follows that 010 can also be removed.
  - **Case 2:** If the agent j knew whether it was muddy, iterate over the worlds agent i considers possible with respect to the true world. Flip the bit in the j^{th} position and create a temporary world as before. If this world is in the list of all possible worlds, remove this world from that list. If this temporary world is not in the list of all possible worlds, then the current world is the true world and agent j can conclude it knows the answer as well.
    Example: Consider agent 3 is informed that

agent 1 knew the answer in the second round. Agent 3 considers 110 and 111 possible with respect to true world. Following the above figure we can see at that time the total possible worlds are 101, 110, 111 and 011. When agent 1 says it knows the answer, let's take 111 out of the two worlds. After flipping, 011 is the world to be checked. Since, it is in the list of possible worlds, it is removed. This is because we know agent 1 has ruled out that world. Now let's take 110. The flipped world is 010. Since this is not in the list of possible worlds, agent 3 can conclude that the true world is 110. This is because from agent 1's perspective only 010 and 110 are possible with respect to the true world.

A. *Output*

Enter the number of children: 3
Enter the true world: 110
Possible Worlds
A:[000, 001, 010, 011, 100, 101, 110, 111]
B:[000, 001, 010, 011, 100, 101, 110, 111]
C:[000, 001, 010, 011, 100, 101, 110, 111]
Public Announcement..
A:[001, 010, 011, 100, 101, 110, 111]
B:[001, 010, 011, 100, 101, 110, 111]
C:[001, 010, 011, 100, 101, 110, 111]
A:[001, 010, 011, 100, 101, 110, 111]
B:[001, 010, 011, 100, 101, 110, 111]
C:[001, 010, 011, 100, 101, 110, 111]
A: No answer
B: No answer
C: No answer
Round 1
A:[001, 010, 011, 100, 101, 110, 111]
B:[001, 010, 011, 100, 101, 110, 111]
C:[001, 010, 011, 100, 101, 110, 111]
A: No answer
B: No answer
C: No answer
A:[110]
B:[110]
C:[011, 101, 110, 111]
A: Yes
B: Yes
C: No answer
Round 2
A:[110]
B:[110]
C:[110]
A: Yes
B: Yes
C: No

VI. Conclusion

The Muddy Children problem was implemented by using the logic of Kripke Structures and possible world semantics. This has to list all possible worlds explicitly and will explode for large inputs. Also, it is very specific to this problem not for any Kripke structure.

Acknowledgment

The author would like to thank her guide Dr. Deepak Khemani for motivating her to write.

References

[1] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, Moshe Y. Vardi *Reasoning about Knowledge*, 1st ed. MIT Press, Cambridge, London 1995.

[2] Hans van Ditmarsch, van der Hoek, Barteld Kooi *Dynamic Epistemic Knowledge*, 1st ed. Springer 2008