

A New Parameter for Maintaining Consistency in an Agent's Knowledge Base Using Truth Maintenance System

Qutaibah Althebyan, Henry Hexmoor
Department of Computer Science and Computer Engineering
University of Arkansas, Fayetteville, Arkansas 72701 .
{qaltheb, hexmoor}@uark.edu

Abstract: In this paper we present a new and a novel way for maintaining the consistency in an agent's knowledge base relying on the notion of Truth Maintenance Systems. In our work, we present a new parameter which helps in determining which propositions or assertions to retract if a contradiction happens. Our new correctness parameter will be very easy to compute, at the same time, it is very effective.

1. Introduction:

When we consider the knowledge base properties of an agent, we could have in mind completeness, efficiency, consistency, and many other properties [1]. Herein, we will limit our attention to consistency. In this paper, we will concentrate on maintaining the consistency within an agent knowledge base. Particularly, we will address local knowledge base consistency in an agent, overlooking the global consistency for a while. Therefore, when we say that an agent is consistent, we mean that this agent is consistent regardless of the system as a whole. Moreover, if we have a group of agents, all the agents may be consistent locally. However, this does not provide the global consistency. In a system of agents, we can expect widely varied information. For an agent A, a proposition S will be valid according to her knowledge, but for another agent B in the same group or system, proposition S may not be valid. On the contrary, the negation of S (not S) may be valid for the agent B, leading to an inconsistent system by having a contradiction (i.e., a proposition and its negation). In our work we are using propositions and assertions [2] to mean the same. We are using both of them in our whole paper interchangeably. However, we concentrate our current work at the individual agent level, where we have a proposition and its negation in the same agent to lead to an inconsistent agent.

2. Motivation:

In our ongoing research, we are trying to build our model of trust. In this process, we divided our work into several subtasks. The first step of our work is trying to maintain each agent of the group of agent consistent. The details of this work are down in this paper. More work will be done in the future.

Our goal, as we said, is trying to build our model of trust. So, the motivation of our existing work which is done and implemented in this paper is trying to build our model of trust using both the Truth Maintenance Systems and trust as our basic blocks or concepts.

In a group of interacting agents, we are aiming to make their negotiation and interaction very trusted. So, we try to make the group of agents globally consistent. This consistency will make our goal of building the trust model much easier. In working to achieve the global consistency, we found it much easier and more effective if we maintain each agent consistent at all times. This will take care of the local consistency in each agent, and hence, we are not expecting any inconsistency in any agent. We achieve this, by trying to overcome any contradiction that may happen by retracting some of the assertions/propositions that cause this contradiction. For this purpose, we find a new and an easier way for overcoming any contradiction that may arise by presenting a new and a novel parameter which guards the process of retraction. In the following discussion, we have more details about our goal. The following example will motivate and illustrate our goal of building a trusted model

If we have the following group of interacting agents:

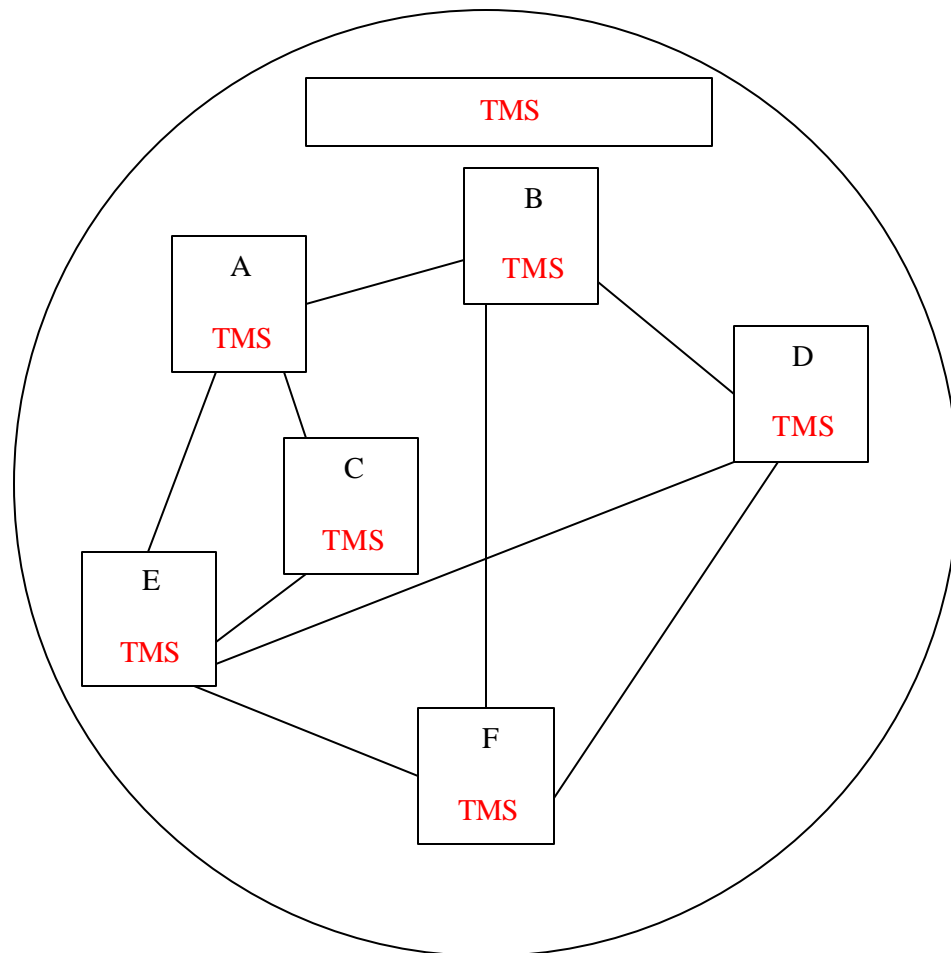


Figure1: A group of interacting agents

In **Figure 1**, we have a group of interacting agents where each agent has a local TMS that will take care of any contradiction that may arise. Also, we have a global TMS which will take care of any contradiction that may arise among the interacting agents. We need this, because it is almost impossible for interacting agents not to have some contradiction, because each agent has her own point of view of information. The difference in opinions always leads to some kind of contradictions. Also, the global TMS will take care of assigning trust values for agents either by increasing or decreasing the trust values. This is because, if a contradiction happens among agents, the global TMS will take care of removing this contradiction by retracting proposition(s) that causes the contradiction. By doing this the global TMS will realize the source of contradicting proposition(s) and hence will decrease the trust value of that agent(s). At the same time, if an agent has a trust value less or equal to the threshold value, and he proves good reputation by not introducing any false information to other agents, the global TMS will increase her trust value.

After some time of interacting and increasing/decreasing trust values of the agents, we are expecting a group of agents in which all of its members are trusted, and at the same time they are consistent, leading to a trusted and a consistent group of agents.

3. Truth Maintenance Systems:

TMSs (Truth Maintenance Systems), also called Reason Maintenance Systems, are one of the common tracking mechanisms for achieving knowledge base integrity [1]. TMSs support default reasoning, provide justification for conclusions. Also, one of the basic uses of TMSs is recognizing inconsistencies. Moreover, TMSs can remember the previously computed derivations. And finally, they support dependency-directed backtracking [3]. There are different types of TMSs, namely, Justification Truth Maintenance Systems, Assumption based Truth Maintenance Systems, Logical based Truth Maintenance Systems, etc. Henceforth, we employed a kind of justification based TMS in our work to fit our system requirements. A justification TMS, in general, is a TMS where each proposition or assertion in it has a justification and is labeled as IN (believed) or else, OUT (disbelieved) [1], meaning that it is not justified or it is invalid. TMSs can be used for achieving the knowledge base integrity in a single agent system and can be used for multiagent system negotiations as well.

(We need to add more details about TMS)

4. Specifications of our Model:

In our system, we have two types of propositions :

Premises: are the propositions sent to the agent by other agents [4]. Once an agent accepts a premise from another agent, she will consider it to be always true. The agent has no access and does not need justifications for the correctness of this premise from the sending agent once she accepts it.

Derived Propositions or (just Propositions): The derived propositions are those that are inferred or constructed in the agent's database based on the Premises she has and her previous knowledge [4].

We will have in our system a new type of information which will play a main and an important role in all our discussions. This information will be saved in the agent's Database and will be considered as the agent's previous knowledge. We will hardcode all the inference rules we have in the calculus and set them as our part of judgment. What we mean here is we take each inference rule such as $A + B \rightarrow C$ and hardcode this in the agent's database. The following table illustrates this process:

```

set op1 = operation
set operator1 = A
set operator2 = B
set operator3 = C
if( op1 eq? +)then
{
  operator1 ? operator3 ( is true)
  operator2 ? operator3 (is true)
}
else if(op1 eq? *)
{
  operator1 ? operator3 ( is false)
  operator2 ? operator3 ( is false)
}

```

We will try to hardcode as much inference rules as we can in order to cover most propositions that will be inferred in the system in the future.

All the future discussion and judgment for propositions will depend on this notion and the notion of justification. For each proposition, if it is believed, then it means that there is a set of other propositions that justify it according to the inference rules. This new justified proposition will be added to the system and will be called the descendent of the justifiers' propositions. The justifying propositions are called the ancestors of this proposition. In the following discussion, we will give more details about this notion.

5. Correctness Parameter:

We posit consider that a contradiction will not be forecasted until it occurs. Upon assertion of a proposition, it is justified. After a while, there might exist a proposition with its negation in the database of an agent, leading to a contradiction. Our focus is finding ways to keep every agent consistent at all times by not permitting any proposition and its negation to exist simultaneously. For that, we define a new *correctness* parameter, which allows us to resolve arising conflicts. This new *correctness* parameter is used to check the correctness of the derived propositions in the agent's database. We will depend on our new parameter and whether the

proposition has been justified or not in order to resolve any conflict or contradiction. By using our *correctness* parameter, as long as the fact whether a proposition is justified or not, we will have a very strong judgment in retracting any of the proposition stored in the system, lowering the chance of retracting any wrong proposition.

Correctness: is a parameter that quantifies the validity of each proposition in the agent knowledgebase. Its value is in the interval [0, 1].

This is different from the traditional labeling algorithm [6], which is used to retract some of the proposition in the agent to remove the conflict, and it is also different from the set of retraction steps proposed by John Doyle [7]. Instead, our parameter will rely on the degree of support. If a proposition is found to have high support then it will not be retracted, but if it is found to have low support it may be retracted. We have developed our model and assigned a value for our *correctness* parameter for each proposition based on our model. After applying our parameter and our criteria of retraction, the conflict will be removed with a very low likelihood of choosing a wrong proposition.

For a proposition P, we assign a correctness value for it based on both the premises in the system, and the previous knowledge of the agent. This value can be one of three values:

- < 0.5 which means that its correctness is low/(very low), and hence, this proposition can be retracted.
- > 0.5 which means that its correctness is high/(very high), and hence, this proposition cannot be retracted.
- $= 0.5$ which means that its correctness is neither high nor low, and hence, this proposition may or may not be retracted, depending on the situation. So, more knowledge is needed.

Before explaining how we got these values we need to mention the way we will base our judgment. Before checking for the correctness value of a proposition we need to check whether the TMS has been able to justify this proposition or not. Then, we will proceed to the next step, which is trying to compute the correctness value of the proposition. If the proposition can be justified, then we can expect a high value of correctness, taking into consideration that this may not be the case. If the proposition cannot be justified, then there are two situations that are to be considered. One is when the proposition can not be justified because it is invalid, and the other one is when the proposition can not be justified because there is no enough knowledge of this proposition and also there is no evidence that this proposition is invalid.

Now, we will proceed to describe the way we got the above values. And in order to understand how we got the previous values, let us consider the following example:

Suppose we have the following Premises in Agent's A Database:

$$A + B \rightarrow C \quad (1)$$

$$D * E \rightarrow F \quad (2)$$

Now, if the following proposition has been derived:

$$A \rightarrow C \quad (3)$$

Then from (1) and our previous knowledge of inference rules, we know that this is true and, hence, will be assigned a high correctness value, like $\text{corr}(P3) = 0.9$. For this proposition we can expect that it has been justified by the TMS.

We have the following components:

Premises/Assumptions: these are the propositions that are sent from other agents. In our model, we consider any proposition received from other agents as a premise, and hence, it is true at all times, and does not need a justification.

Inference Engine: this component creates new propositions. The creation of new propositions will depend on the agent's previous knowledge and the set of premises. However, some of the new created propositions may not depend on any of the previous knowledge of the agent. This does not mean that the created proposition will be true or valid either. At this stage, the agent has no guarantee of correctness or validity of any proposition. Some of the propositions/assertions may violate other propositions that exist in the agent's database. There are many reasons for this. One reason may be due to the fact that new propositions fit the requirement of the current time and state, and the old ones may be out of date at this time. Also, new requirements arise as time progresses. Moreover, the premises are considered as eternally true propositions and they do not need justifications. These premises come from different sources or agents. And since the new derived propositions partially depend on them, some contradictions may occur between these assertions or propositions in the agent due to the disparity of different sources of premises.

TMS: the TMS is the central component of our model. It has access to all propositions and premises. It is the component where the propositions are justified and then labeled as *justified*, meaning they are justified or believed; or *not-justified*, meaning that they are disbelieved. When a new proposition is created, the TMS will try to justify this proposition by relying on its previous knowledge and the set of premises the agent has. Also, it will control the system by detecting any contradiction that may happen. Once any contradiction happens, it tries to resolve this contradiction by trying to retract some of the propositions relying on the fact whether the proposition is justified or not, and the correctness parameter which will be assigned to each proposition. In the system, every proposition and all its descendents are maintained in the agent's database in a form of a graph. Hence, at the time a contradiction happens, we can keep track of the propositions, and follow the chain of propositions or assertions from the proposition we are on now back to its ancestor until getting to the set of premises. By this we can specify and choose the proposition or propositions that need to be retracted, using Dependency- Directed Backtracking [2].

Correctness Handler: this is a novel component where each proposition receives its correctness value. A correctness value will be any value between 0 and 1 inclusive. A proposition which is found to be valid will be assigned a correctness value of more than 0.5, and will be labeled as IN. Also, a proposition which is found to be invalid will be assigned a correctness value of less than 0.5, and then will be labeled as OUT. However, if a TMS fails to judge whether the proposition is valid or invalid because of not having enough information about the proposition, this proposition will be

assigned a correctness value of 0.5, and will be labeled as *pending*. The *pending* proposition means that it may be relabeled later as either IN or OUT, by having new information or evidence of validity or invalidity. If new evidence appears to support this proposition, this proposition will be relabeled as IN. Also, if any new created proposition depends on one of the *pending* propositions as its justification, then the *pending* proposition or assertion will be relabeled as IN. The new proposition will be the descendent of the relabeled proposition. However, if new evidence appears to show that some *pending* proposition is invalid, the *pending* assertion/proposition will be labeled as OUT, and its correctness value will be lowered by a value of 0.1, making its correctness value less than 0.5.

7. Conflict Management, a Scenario:

Most truth maintenance systems have some way of detecting a contradiction when happens. This can be done by adding a special proposition symbol called *contradiction*. If the truth maintenance system is able to detect a contradiction then the contradiction proposition will return **yes** [3].

But how can this work in resolving any contradiction?

Suppose we have the following set of inferences:

$$\begin{array}{l} P1 \text{ ? } P2 \text{ ? } P3 \qquad (1) \\ \text{And } q1 \text{ ? } q2 \text{ ? } q3 \qquad (2) \end{array}$$

And suppose the following correctness values 0.8, 0.5, 0.3 have been assigned to each proposition of P1, P2, and P3, respectively. And the following values 0.7, 0.6, 0.9 for q1, q2, and q3, respectively. And suppose that P3 has not been justified and q3 has been justified.

And suppose we have the situation where $P3 = \sim q3$ which causes a contradiction in the system. Then based in the correctness values we have, P3 will be retracted because it has a correctness value less than 0.5 which is less than q3's value and it is not justified, removing the conflict from the system.

Continuing with our previous example, we may consider the following situations:

- If $\text{corr}(q1) > 0.5$ and $\text{corr}(p1) < 0.5$ then retract P1 given that q1 has been justified and p1 has not been justified.
- If $\text{corr}(q1) > 0.5$ and $\text{corr}(p1) = 0.5$ then retract P1
- If $\text{corr}(q1) > 0.5$ and $\text{corr}(p1) > 0.5$ then it is not enough to check which one is greater than the other because it is very difficult to retract both of them. Also, both propositions has been justified and labeled as IN. So, this situation needs further discussion
- If $\text{corr}(q1) < 0.5$ and $\text{corr}(p1) < 0.5$ we can retract any of these two proposition, but for more accuracy we will discuss this further too.

If we have the following situation where $\text{corr}(q1) > 0.5$ and $\text{corr}(p1) > 0.5$ we consider the following solutions:

- Assign a timestamp to each proposition at the moment this proposition is inferred or constructed. Then if we have the above situation we can check the timestamp for each proposition and retract the older one. Here we consider the newest one the freshest and hence has a better chance to be

the most correct one. But this is not always the case? So, we will consider the following:

- Evaluate the correctness average starting from the last proposition P3 back to the first premise participating in constructing this proposition. So
 - $Avg(corr(P_i)) = \sum P_i / n$
 - And do the same for the q_i
 - Then compare the two averages and retract the proposition with the lowest average.

Relying on the average may not always give us the correct solution, but we hope that (On Average) it will give us the correct solution.

We apply the same scenario if we have the situation where $corr(q1) < 0.5$ and $corr(p1) < 0.5$ by considering one of the two solutions above.

(The proposition may be correct and may be not. Similarly, the proposition may be valid and may be not. We are distinguishing between the correctness and validity of a proposition. A correct proposition is a proposition which is correct in its meaning and does not violate an inference rule. Whereas, a valid proposition is a proposition which is valid according to the inference rules, but does not need to be correct in its meaning).

8. Conclusion:

In our work, we studied Truth Maintenance Systems. We introduced salient components of a TMS and then we introduced a new *correctness* parameter. By combining the TMS with our new parameter, we provide a novel and resilient conflict management scheme. Correctness Parameter is used to help in deciding which propositions/ assertions to retract if a contradiction occurred. These help the TMSs in reducing the chance of retracting wrong parameters, leading to lower the chance of wrong judgments and at the same time, enhance the correctness of the working of the Truth Maintenance System.

9. Future Work:

Our previous work will be the first step in our ongoing research. The next step will be trying to extend our work by applying our correctness parameter to a multiagent system. We are trying to maintain the consistency among a group of agents, and, hence, having the notion of global consistency. Also, we are trying to build our model of trust. In this work, we have built the first step, which is maintaining the consistency in each agent, then we will extend this notion to have a global consistency among a group of agents by enhancing the inter agent consistency. And finally, we will build our trust model. This will support and enhance our model of trust, and will make it a trusted and a reliable model. Moreover, it will make achieving trust among the agents much easier and more reliable.

References:

- [1] M. Huhns and D. Bridgeland: Multiagent Truth Maintenance. IEEE Transactions on Systems, Man, and Cybernetics, Vol. 21, No. 6, Nov/Dec 1991.
- [2] Stuart C. Shapiro: Belief Revision and Truth Maintenance Systems: an Overview and a Proposal. Technical Report 98-10, Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY, December 1998.
- [3] Kenneth D. Forbus and Johan de Kleer: Building Problem Solvers, MIT Press, 1993
- [4] M. Barbuceanu, M.S. Fox: The Information Agent: An Infrastructure Agent Supporting Collaborative Enterprise Architectures, Proceedings of 3-rd Workshop on Enabling Technologies, Infrastructure for Collaborative Enterprises, Morgantown WV, IEEE Computer Society Press, 1994.
- [5] David A. McAllester: Truth Maintenance. AAAI-90, pp. 1109-1116
- [6] J. Kleer: A general labeling algorithm for assumption-based truth maintenance, AAAI-88, pp. 188-192, 1988.
- [7] J. Doyle: Truth maintenance systems for problem solving. Fifth International Joint Conference on Artificial Intelligence, Cambridge, Massachusetts, 1977.