

Solving Sum and Product Riddle via BDD-based Model Checking

Xiangyu Luo^{1,2,†}, Kaile Su^{3,4,‡}, Abdul Sattar^{4,§} and Yan Chen^{2,¶}

¹Tsinghua National Laboratory for Information Science and Technology
Key Laboratory of Security for Information System of Ministry of Education
School of Software, Tsinghua University, Beijing 100084, China

²Department of Computer Science, Guilin University of Electronic Technology, Guilin 541004, China

³School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China

⁴Institute for Integrated and Intelligent Systems, Griffith University, Brisbane, QLD 4111, Australia

[†]xyluo@tsinghua.edu.cn, [‡]sukl@pku.edu.cn, [§]a.sattar@griffith.edu.au, [¶]lenny0519cy@gmail.com

Abstract

We model the Sum and Product Riddle in public announcement logic, which is interpreted on an epistemic Kripke model. The model is symbolically represented as a finite state program with n agents. A model checking method to the riddle is developed by using the BDD-based symbolic model checking algorithm for logic of knowledge we developed in [7]. The method is implemented by extending the model checker MCTK [7] and then the solution of the riddle is verified successfully.

1. Introduction

In 1969, H. Freudenthal first stated Sum and Product riddle in Dutch language [2], and then solved this riddle in 1970 [3]. A translation of the original formulation quoted from [8] is: J says to S and P: I have chosen two integers x and y such that $1 < x < y$ and $x + y \leq 100$. In a moment, I will inform S only of $s = x + y$, and P only of $p = xy$. These announcements remain private. You are required to determine the pair (x, y) . He acts as said. The following conversation now takes place:

- (1) P says: "I do not know it."
- (2) S says: "I knew you didn't."
- (3) P says: "I now know it."
- (4) S says: "I now also know it."

Determine the pair (x, y) .

At first appearance, it seems that these announcements are unuseful for agents to reason about the two numbers, because these announcements are just about agents' knowledge or ignorance about the two numbers, not about the two numbers

themselves. However, these announcements are actually very informative, from them rational agents are able to learn facts about the two numbers. For example, the numbers cannot be 2 and 4, because 8 is the unique product of 2 and 4 but P first announce that he do not know the numbers. The numbers cannot be two prime numbers, or else P, who knows the product, can immediately deduce them according to the uniqueness of product of two prime numbers. Besides, the sum of the numbers cannot be the sum of two prime numbers, or else S would consider it possible that P knew the numbers, but it is not fact from the second announcement of S. By using the elimination method above, S and P can finally determine the numbers is the pair (4, 13).

Sum and Product problem can be modeled as a Multi-Agent System (MAS). In the multi-agent paradigm, particular emphasis is given to the formal representation of the mental attitudes of agents, such as agents' knowledge, beliefs, desires, intentions and so on. Dynamic epistemic logic was developed to study the changes of agents' individual knowledge or group knowledge caused by communication about agents' or group knowledge. Public Announcement Logic is a kind of dynamic epistemic logic. It is very suitable for modeling the sum and product problem because it is able to describe the actions of public announcement about agents' or group knowledge. Hans P. van Ditmarsch *et al.* [9] claimed that they was the first to use an automated model checker DEMO [10] to tackle the problem. In [9], they stated that the problem cannot be tackled in two state-of-the-art symbolic model checkers for temporal logics of knowledge [1, 6], MCK 0.2.0 [4] and MCMAS 0.7 [5] because they does not support checking epistemic formulas as preconditions in their specification languages.

In this paper, we propose a symbolic model checking method for temporal logic of knowledge, not for public announcement logic, that is able to automatically tackle sum and product problem by using MCTK [7], a kind of model checker for temporal logic of knowledge developed by us. The experimental results show that the proposed method is correct and efficient.

2. Public announcement logic

Given a set of agents $A = \{1, \dots, n\}$ and a set of primitive propositions P . The language syntax of public announcement logic is inductively defined as

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid K_a\varphi \mid C_\Gamma\varphi \mid [\varphi]\psi$$

where $p \in P$, $a \in A$ and $\Gamma \subseteq A$. For $K_a\varphi$, read “agent a knows formula φ ”. For $C_\Gamma\varphi$, read “group Γ of agents commonly know formula φ ”. For $[\varphi]\psi$, read “after public announcement of φ ”, formula ψ is true.

We now give the semantics of public announcement logic under an epistemic Kripke model $M = \langle S, V, \sim_1, \dots, \sim_n \rangle$. Let L_e be a set of possible states for the environment and L_i a set of possible local states for each agent $i \in A$, we take $S = L_e \times L_1 \times \dots \times L_n$ to be the set of (global) states. $V : P \rightarrow \mathcal{P}(S)$ is a valuation function. For each $p \in P$, V gives the set of states in which p is true. We associate each agent $i \in A$ with an equivalence relation \sim_i over the set of states. We define function $l_i : S \rightarrow L_i$ to return the local state of agent i from a state $s \in S$, then for any states $s, s' \in S$, $s \sim_i s'$ iff $l_i(s) = l_i(s')$. It means that s and s' are indistinguishable for agent i . The semantics of public announcement is defined by the induction on the structure of formulas:

- $M, s \models p$ iff $s \in V(p)$, where $p \in P$;
- $M, s \models \neg\varphi$ iff $M, s \not\models \varphi$;
- $M, s \models \varphi \wedge \psi$ iff $M, s \models \varphi$ and $M, s \models \psi$;
- $M, s \models K_i\varphi$ iff $M, s' \models \varphi$ for all $s' \in S$ such that $s \sim_i s'$;
- $M, s \models C_\Gamma\varphi$ iff $M, s' \models \varphi$ for all $s' \in S$ such that $s \sim_\Gamma^C s'$, where \sim_Γ^C is the transitive closure of $\bigcup_{i \in \Gamma} \sim_i$;
- $M, s \models [\varphi]\psi$ iff $M, s \models \varphi$ implies $M|_\varphi, s \models \psi$, where $M|_\varphi = \langle S', V', \sim'_1, \dots, \sim'_n \rangle$ is also an epistemic Kripke model, in which $S' = \{s \in S \mid M, s \models \varphi\}$, $V'(p) = V(p) \cap S'$ for each $p \in P$ and $\sim'_i = \sim_i \cap (S' \times S')$ for each $i \in A$.

Given an epistemic Kripke model M and a formula φ , φ is valid on model M , notation $M \models \varphi$, if and only if $M, s \models \varphi$ for each state s in M .

3. Sum and product problem in public announcement logic

In this section we model the sum and product problem in public announcement logic. We first determine the set of propositions and the set of agents. From the formulation of the problem, the integer pair (x, y) is included in $I = \{(x, y) \in \mathbb{N}^2 \mid 1 < x < y \text{ and } x + y \leq 100\}$. Therefore, we need 14 propositional variables $\{x_6, x_5, \dots, x_0\}$ and $\{y_6, y_5, \dots, y_0\}$ to specify x and y by binary coding. We use notations E_x^i and E_y^j to represent the truth of propositions “ $x = i$ ” and “ $y = j$ ” respectively. For example, E_x^{13} stands for the propositional formula $\neg x_6 \wedge \neg x_5 \wedge \neg x_4 \wedge x_3 \wedge x_2 \wedge \neg x_1 \wedge x_0$. Compare with the encoding method in [8], our encoding method is more succinct, because in [8] they need to create at least 194 propo-

sitional letters for each assignment to the two integer variables.

Consider the set of agents, the function of agent J just like the environment of a multi-agent system, is to ensure that other agents gain the background knowledge of the problem. We don't care about J 's knowledge. Therefore, the set of agents $A = \{S, P\}$.

The proposition “ S knows that the numbers are 4 and 13” is represented as $K_S(E_x^4 \wedge E_y^{13})$. Now consider how to describe the proposition “ S knows the two numbers” in public announcement logic. First for agent S , we can use $K_Sx = \bigwedge_{0 \leq i \leq 6} (K_Sx_i \vee K_S\neg x_i)$ and $K_Sy = \bigwedge_{0 \leq i \leq 6} (K_Sy_i \vee K_S\neg y_i)$ to represent the two propositions “ S knows the integer x ” and “ S knows the integer y ” respectively. Take formula K_Sx for example, it means that agent S knows the value of all propositional variables encoding integer variable x . Therefore, the proposition “ S knows the two numbers” can be represented as $K_S(x, y) = K_Sx \wedge K_Sy$. Similarly, the proposition “ P knows that the two numbers” can be represented as $K_P(x, y) = K_Px \wedge K_Py$, where the structure of K_Px, K_Py is similar to that of K_Sx, K_Sy . Thus, the four announcements of the problem can be described in turn as:

$$\neg K_P(x, y), K_S\neg K_P(x, y), K_P(x, y) \text{ and } K_S(x, y).$$

Now, we can interpret these announcements on the epistemic Kripke model $\Theta = \langle I, V, \sim_S, \sim_P \rangle$, where $I = \{(x, y) \in \mathbb{N}^2 \mid 1 < x < y \text{ and } x + y \leq 100\}$. For any integer $x, y, x', y' \in I$, $(x, y) \sim_S (x', y')$ iff $x + y = x' + y'$, and $(x, y) \sim_P (x', y')$ iff $xy = x'y'$. Valuation function V is defined as: for each $i = 0..6$, $V(x_i) = \{(x, y) \in I \mid ((x/2^i) \bmod 2) = 1\}$ and $V(y_i) = \{(x, y) \in I \mid ((y/2^i) \bmod 2) = 1\}$, where operator “/” is integer division without rest. Valuation function V labels each state (x, y) with the set of propositional variables true in the state according to the binary code of x and y .

Finally, we can define the model validity formula as

$$\Theta \models [K_S\neg K_P(x, y)][K_P(x, y)][K_S(x, y)](E_x^4 \wedge E_y^{13}),$$

to express that the integer pair (4, 13) is the only solution of sum and product problem. Note that as the analysis in [8, 9], we consider P 's announcement (1) is superfluous for the analysis and then omit it. The ‘knew’ in S 's announcement (2) refers to the truth of that announcement in the initial epistemic state, not in the epistemic state resulting from announcement (1).

4. Model checking knowledge in MCTK

In this paper we adopt the BDD-based symbolic model checker MCTK we developed in [7], to evaluate agents' knowledge. Given an epistemic Kripke model $M = \langle S, V, \sim_1, \dots, \sim_n \rangle$, we can symbolically represent M as a *finite-state program with n agents*, a tuple $\mathcal{P}_M = (\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'), O_1, \dots, O_n)$, where $\mathbf{x} = \{x_1, \dots, x_k\}$ is a set of boolean variables. Any state in S can be encoded as

an assignment for \mathbf{x} , thus a set of states in $\mathcal{P}(S)$ can be represented as a boolean formula over \mathbf{x} or a subset of \mathbf{x} ; θ is a boolean formula over \mathbf{x} representing the set of initial states, called the *initial condition*; τ is a boolean formula over $\mathbf{x} \cup \mathbf{x}'$, called the *transition relation*, where $\mathbf{x}' = \{x'_1, \dots, x'_k\}$ is a copy of \mathbf{x} , encoding the next state in a transition relation; and for each agent $i \in \{1, \dots, n\}$, $O_i \subseteq \mathbf{x}$ is the set of agent i 's *local variables*, or *observable variables*. For any $p \in \mathbf{x}$, if state $s \in V(p)$ then $s(p) = \text{true}$, here we omit V in \mathcal{P}_M .

Now, we can compute the set of reachable global states of S in M , via Ordered Binary Decision Diagram (OBDD), as

$$G(\mathcal{P}_M) = \text{Ifp}Z \left[\theta(\mathbf{x}) \vee (\exists \mathbf{x}' (Z \wedge \tau(\mathbf{x}, \mathbf{x}')) \left(\frac{\mathbf{x}'}{\mathbf{x}} \right) \right],$$

where $\text{Ifp}Z\xi(Z)$ is a *least fixed point* of an operator ξ from the set of boolean formulas over \mathbf{x} to the set of boolean formulas over \mathbf{x} . $\psi(\frac{\mathbf{x}'}{\mathbf{x}})$ is the result of renaming variables in \mathbf{x}' by those in \mathbf{x} respectively.

Given an epistemic Kripke model M and an epistemic formula $K_i\varphi$, from Proposition 3 in [7] we can compute the set of states satisfying $K_i\varphi$ as $\forall(\mathbf{x} - O_i)(G(\mathcal{P}_M) \Rightarrow \varphi)$, where φ is a formula that does not contain any temporal modalities.

For sum and product problem we only focus on the resulting set of states after all announcements. These announcements are just about agents' knowledge or ignorance about the two numbers. For the limited space, in this paper we do not deal with common knowledge modality and temporal operators. Thus, the set of states in M (generated from \mathcal{P}_M) satisfying a formula φ that does not contain common knowledge modality and temporal operators, can be computed as the OBDD:

$$[\varphi, M] = \begin{cases} \text{The OBDD of } \varphi, & \text{if } \varphi \in \mathbf{x} \text{ of } \mathcal{P}_M; \\ \neg[\alpha, M], & \text{if } \varphi = \neg\alpha; \\ [\alpha, M] \wedge [\beta, M], & \text{if } \varphi = \alpha \wedge \beta; \\ \forall(\mathbf{x} - O_i)(G(\mathcal{P}_M) \Rightarrow [\alpha, M]), & \text{if } \varphi = K_i\alpha. \end{cases}$$

5. Modeling sum and product problem in MCTK

The input language of MCTK [7] is an extension of that of NuSMV, in which we extend the declaration language for agents' observable variables, such that in MCTK we can describe a finite-state program with n agents $\mathcal{P}_\Theta = (\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'), O_S, O_P)$ for the epistemic Kripke model of the problem $\Theta = \langle I, V, \sim_S, \sim_P \rangle$ as Fig. 1.

In Fig. 1, integer variables x and y encode the number pair. To be able to make the sum and product of x and y observable for agents S and P respectively, we define two additional integer variables `sum` and `product` to represent the sum and product of x and y respectively. The domains of `sum` and `product` are $5 \leq \text{sum} \leq 195$ and $6 \leq \text{product} \leq 9506$. Therefore, the set \mathbf{x} of \mathcal{P}_Θ is the union of binary coding variables of x, y, sum and `product`.

Line 7-8 restrict the initial value of the four integer variables, which is the initial condition $\theta(\mathbf{x})$ of \mathcal{P}_Θ . Line 9-10 define

```

1  MODULE main()
2  VAR
3    x : 2..98;      y : 2..98;
4    sum : 5..195;  product : 6..9506;
5    S : SumAgent(sum);
6    P : ProductAgent(product);
7  INIT x>1 & y>x & x+y<=100 &
8      sum=x+y & product=x*y;
9  TRANS next(x)=x & next(y)=y &
10     next(sum)=sum & next(product)=product;
11 MODULE SumAgent(Observable sum)
12 MODULE ProductAgent(Observable product)

```

Figure 1. The finite-state program with agents S and P for sum and product problem

the transition relation $\tau(\mathbf{x}, \mathbf{x}')$ of \mathcal{P}_Θ , to describe the situation that the four integer variables will keep unchangeable once their initial values are chosen non-deterministically under the restriction of $\theta(\mathbf{x})$.

Line 11-12 are the module definitions of agents S and P, in which the formal parameters `sum` and `product` are observable respectively for S and P. Line 5-6 declare agents S and P, the actual parameters of S and P are `sum` and `product` respectively, it means that the set O_S of S's observable boolean variables is the binary coding variables of integer variable `sum`, and the set O_P of P's observable boolean variables is the binary coding variables of integer variable `product`.

6. Model checking sum and product problem in MCTK

In this section we reduce the model validity formula $\Theta \models [K_S \neg K_P(x, y)][K_P(x, y)][K_S(x, y)](E_x^4 \wedge E_y^{13})$ in Section 3, to an OBDD-based computation process on the finite state program with n agents \mathcal{P}_Θ .

From Section 4 we know that the set of states in M that satisfies the epistemic formula $K_i\varphi$ can be computed as the OBDD of $\forall(\mathbf{x} - O_i)(G(\mathcal{P}_M) \Rightarrow \varphi)$. It is denoted as $[K_i\varphi, M]$. According to the semantics of $M, s \models [\varphi]\psi$ in Section 2, we learn that $M|_\varphi = \langle S', V', \sim'_1, \dots, \sim'_n \rangle$ is an epistemic Kripke model that satisfy formula φ . Obviously, it is easy to compute the set of reachable states of $M|_\varphi$ as the OBDD of $G(\mathcal{P}_M) \wedge [\varphi, M]$. Therefore, we have the following proposition.

Proposition 1. *Given epistemic Kripke models $M = \langle S, V, \sim_1, \dots, \sim_n \rangle$ and $M|_\varphi = \langle S', V', \sim'_1, \dots, \sim'_n \rangle$, and a formula ψ , the set of states in $M|_\varphi$ satisfying ψ is the following OBDD: $[\psi, M|_\varphi] =$*

$$\begin{cases} \text{The OBDD of } \psi, & \text{if } \psi \in \mathbf{x}; \\ \neg[\alpha, M|_\varphi], & \text{if } \psi = \neg\alpha; \\ [\alpha, M|_\varphi] \wedge [\beta, M|_\varphi], & \text{if } \psi = \alpha \wedge \beta; \\ \forall(\mathbf{x} - O_i)((G(\mathcal{P}_M) \wedge [\varphi, M]) \Rightarrow [\alpha, M|_\varphi]), & \text{if } \psi = K_i\alpha. \end{cases}$$

where φ and ψ are formulae that does not contain common knowledge modality and temporal operators, and \mathbf{x} is the set of boolean variables in \mathcal{P}_M .

Thus, for sum and product problem, after these announcements: $[K_S \neg K_P(x, y)][K_P(x, y)][K_S(x, y)]$, the set of reachable states is the OBDD of $[K_S(x, y), \Theta|_{K_S \neg K_P(x, y)}|_{K_P(x, y)}]$. We further have that the OBDD that represent all solutions for sum and product problem is the conjunction of the initial states and the set of reachable states after these announcements:

$$\theta(\mathbf{x}) \wedge [K_S(x, y), \Theta|_{K_S \neg K_P(x, y)}|_{K_P(x, y)}].$$

7. Experimental results

We successfully implemented the symbolic model checking method for sum and product problem in MCTK. The BDD-package exploited in MCTK is the CUDD library developed by Fabio Somenzi at Colorado University. The experiment is based on a laptop configuration Ubuntu 2.6.10-5-386 Linux system, Intel(R) Pentium(R) M 1.60 GHz processor and 512Mb RAM. The time required for the whole model checking process, i.e., both symbolic model construction and formula checking, is about 90 seconds. 236521 BDD nodes are allocated. The model checking result is

$$(\theta(\mathbf{x}) \wedge [K_S(x, y), \Theta|_{K_S \neg K_P(x, y)}|_{K_P(x, y)}]) \Leftrightarrow (E_x^4 \wedge E_y^{13}),$$

it means that it is possible to publicly announce (2), (3) and (4) in turn if and only if the initial state is (4, 13). In other words, the only solution of sum and product problem is (4, 13), which is consistent with the result of [8, 9].

To compare with the experimental result of [8, 9], on the same experimental platform we implement the sum and product riddle as Fig.1 of [8] in dynamic epistemic model checker DEMO, and verify that the unique solution is also the pair (4, 13). The Haskell interpreter for DEMO is GHC Interactive, version 6.2.2, for Haskell 98. The time required for checking solution of the interpreted version of the problem, is about 1864 seconds. Obviously, the performance of MCTK is higher than that of interpreted version of DEMO, at least for checking sum and product problem. However, We believe that the compiled version of the problem by DEMO does better than the interpreted version.

8. Conclusions

In this paper we first model the sum and product problem in public announcement logic, then describe the problem in the finite state program with agents S and P, from which the epistemic Kripke model of the problem can be automatically generated by our symbolic model checker MCTK. By adopting

the symbolic model checking method for epistemic formulas we developed in [7], we are able to reduce the model checking problem in public announcement logic to a series of BDD-based computations of the set of states that satisfies a given epistemic formula. The experimental results show that the proposed method is correct and efficient. As for future work, to tackle the problems similar to sum and product riddle, we plan to develop an agent conversation language supporting the communication of agents' mental states, then design a translation process to automatically translate the agent conversation language into the finite state program with n agents, finally improve the symbolic model checking algorithms [7] for dynamic epistemic logics.

Acknowledgements The authors would like to thank the reviewers for valuable comments. This work was partly supported by the National Natural Science Foundation grants 60763004, 60725207 and 60553002, the Young Science Foundation of Guangxi Province of China grant GuiKe-Qing0728090, GuiJiaoKeYan 2006 (No.26, D2006044), and the Innovation Project of Guangxi Graduate Education of China grant 2007105950811m19.

References

- [1] R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about knowledge*. MIT Press, Cambridge, MA, 1995.
- [2] H. Freudenthal. Formulation of the sum-and-product problem. *Nieuw Archief voor Wiskunde*, 3(17):152, 1969.
- [3] H. Freudenthal. Solution of the sum-and-product problem. *Nieuw Archief voor Wiskunde*, 3(18):102–106, 1970.
- [4] P. Gammie and R. van der Meyden. MCK: Model checking the logic of knowledge. In R. Alur and D. Peled, editors, *CAV*, volume 3114 of *Lecture Notes in Computer Science*, pages 479–483. Springer, 2004.
- [5] A. Lomuscio and F. Raimondi. MCMAS: A model checker for multi-agent systems. In H. Hermanns and J. Palsberg, editors, *Proceedings of TACAS-2006*, volume 3920 of *Lecture Notes in Computer Science*, pages 450–454. Springer, 2006.
- [6] X. Luo, K. Su, A. Sattar, and M. Reynolds. Verification of multi-agent systems via bounded model checking. In A. Sattar and B. H. Kang, editors, *Australian Conference on Artificial Intelligence*, volume 4304 of *Lecture Notes in Computer Science*, pages 69–78. Springer, 2006.
- [7] K. Su, A. Sattar, and X. Luo. Model Checking Temporal Logics of Knowledge Via OBDDs. *The Computer Journal*, 50(4):403–420, 2007.
- [8] H. P. van Ditmarsch, J. Ruan, and L. C. Verbrugge. Model checking sum and product. In S. Zhang and R. Jarvis, editors, *Australian Conference on Artificial Intelligence*, volume 3809 of *Lecture Notes in Computer Science*, pages 790–795. Springer, 2005.
- [9] H. P. van Ditmarsch, J. Ruan, and R. Verbrugge. Sum and product in dynamic epistemic logic. *Journal of Logic and Computation*, 18(4):563–588, 2008.
- [10] J. van Eijck. Dynamic epistemic modelling. Technical report CWI Report SEN-E0424, Centrum voor Wiskunde en Informatica, Amsterdam, 2004.