和与积数迷的符号化模型检测*)

骆翔宇　古天龙　董荣胜

(桂林电子科技大学计算机与控制学院　桂林 541004)

摘　要　和与积是一个著名的数迷问题。采用公告逻辑对该问题进行建模，将其 Kripke 模型符号化表示为多智能体有限状态程序，并在其上采用一种基于局部命题解释系统语义的知识逻辑符号化模型检测算法计算该问题的所有解。在时态逻辑模型检测器 NuSMV 基础上扩展实现了本文算法，然后在相同实验平台上用动态认知建模工具 DEMO 对该问题进行求解。实验表明，我们的算法不仅结果正确，而且在运行效率上与 DEMO 相比占有绝对优势。

关键词　符号化模型检测，多智能体系统，时态逻辑，公告逻辑

Symbolic Model Checking for Sum and Product Riddle

LUO Xiang-yu　GU Tian-long　DONG Rong-sheng

(Shcool of Computer&-Control, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract　The Sum-and-Product Riddle is a famous number puzzle. The riddle is modeled in a modal logic called public announcement logic, which is interpreted on multi-agent Kripke model. The model is symbolically represented as a multi-agent finite state program. Based on the semantics of the logic, a symbolic model checking approach to the riddle is developed by using the symbolic model checking algorithm for logic of knowledge under the semantics of interpreted systems with local propositions. The approach is then implemented by extending the model checker NuSMV and in which the solution of the riddle is verified successfully. The riddle is also verified by using the epistemic model checker DEMO in the same experimental environment. The comparison between the experimental results from the two model checkers show that the running efficiency of our model checking approach is much higher than that of DEMO.

Keywords　Symbolic model checking, Multi-agent system, Temporal logic, Public announcement logic

1　引言

和与积问题是一个著名的数字谜题，1969 年首次由 Freudenthal 提出[1]。该问题可描述如下：

J 向 S 和 P 说道：我有两个不同的整数 x 和 y，它们满足 $1<x<y$，并且 $x+y\leqslant 100$。我将秘密地把和 $s=x+y$ 告诉 S，而把积 $p=xy$ 告诉 P。请你们确定这两个数(x,y)是什么。

J 将和与积分别秘密告知 S 和 P 后，发生如下对话：

(1)P 说："我不知道这两个数"；

(2)S 说："我早已知道你不知道的这两个数"；

(3)P 说："我现在知道这两个数了"；

(4)S 说："我现在也知道这两个数了"。

请问这两个数(x, y)是什么？

从上述对话可看出，智能体 S 和 P 只是宣告了他们的知识或无知，并没有关于数字的信息，因此这些宣告似乎是无用的。然而事实并非如此，智能体实际上可从别人的宣告中推断出事实。例如，这两个数不可能是 2 和 4，因为只有 2 和 4 的积是 8，这样 P 可立即推断出这两个数是 2 和 4。这两个数也不可能是素数，因为两个素数的积是唯一的，所以 P 也可根据积立即推断出这两个素数。此外，这两个数也不可能是 14 和 16，因为素数 7 和 23 的和同样是 30，而素数 7 和 23 可被 P 立即推断出。因此，如果这两个数的和是 30，那么 S 应该认为 P 可能知道这两个数。然而 S 却说他知道 P 不知道这两个数。因此这两个数不可能是 14 和 16。通过上述推断，S 和 P 可逐步消减那些不可能的数对。最终他们将会推出和与积数迷问题的唯一解是(4,13)[2]。

和与积问题是一种典型的多智能体系统。而分析智能体关于其他智能体心智状态的知识，以及通信对智能体知识的影响，是解决这类问题的关键所在。动态认知逻辑(Dynamic Epistemic Logic[3])由多智能体认知逻辑扩展而来，用于研究智能体及其群体知识在通信过程中的演变。而公告逻辑(Public Announcement Logic)是一种动态认知逻辑，它可描述类似和与积问题中的公告动作。Ditmarsch 等在文献[4]中采用公告逻辑对该问题进行建模，并成功利用动态认知模型检测器 DEMO[5]对该问题进行求解。他们宣称利用时态认知逻辑的模型检测器(如 MCK 和 MCMAS)求解该问题的困难在于它们不支持在系统描述语言的前置判断条件中直接指定认知公式。因此本文将针对这一问题提出一种基于时态认知逻辑[6]模型检测方法的和与积问题模型检测算法，用于自动检测该问题的所有解，而在系统描述语言中不必指定任何认知公式。

本文的目标是将公告逻辑描述的和与积数迷的模型检测问题转化为其在时态认知逻辑模型上的模型检测问题。事实上，我们在时态认知逻辑的模型检测领域已有一些成果[7,9]，其中文献[7]提出的模型检测算法是基于有序二叉判定图(OBDD[10])的，而文献[8,9]提出的是基于命题可满足性求解

*)国家自然科学基金(60763004)、广西青年科学基金(D200716)。**骆翔宇**　博士，主研方向包括模型检测、网络安全；**古天龙**　教授，博士生导师；**董荣胜**　教授。

技术(SAT)的。我们基于文献[7]的知识逻辑模型检测方法提出了和与积问题的符号化模型检测算法,并在时态逻辑模型检测器 NuSMV 的开放源码基础上扩展实现。实验结果表明了本文方法的正确性,并且比文献[4]中采用认知模型检测器 DEMO 的方法在运行效率上占有绝对优势。

2 公告逻辑

给定一个智能体集合 $A=\{1,\cdots,n\}$ 和原子命题集合 P,公告逻辑语言的语法归纳定义如下:

$\varphi::=p|\neg\varphi|\varphi\wedge\psi|K_a\varphi|C_G\varphi|[\varphi]\psi$

其中 $p\in P$,$a\in A$ 和 $G\subseteq A$。公式 $K_a\varphi$ 表示"智能体 a 知道公式 φ",公式 $C_G\varphi$ 表示"公式 φ 是 G 中所有智能体的公共知识",而公式 $[\varphi]\psi$ 表示"公式 φ 被公告后,公式 ψ 成立"。

我们在 Kripke 模型 $M=[S,V,\sim_1,\cdots,\sim_n]$ 上解释公告逻辑的语义,其中 S 是有限状态集合。评价函数 $V:P\to S$ 给出使原子命题 $p\in P$ 为 true 的状态集合($V(p)$简写为 V_p)。$\sim_i(i=1\cdots n)$是认知可达关系。假设智能体 i 在状态 s 中的局部状态是 $l_i(s)$,则对于状态 $s,s'\in S$,$s\sim_i s'$ 当且仅当 $l_i(s)=l_i(s')$,这意味着智能体 i 根据它的局部状态不能区别状态 s 和 s'。公告逻辑在 $M=[S,V,\sim_1,\cdots,\sim_n]$ 上的语义定义如下:

$M,s|=p$ 当且仅当 $s\in V_p$;

$M,s|=\neg\varphi$ 当且仅当 $M,s|\neq\varphi$;

$M,s|=\varphi\wedge\psi$ 当且仅当 $M,s|=\varphi$ 并且 $M,s|=\psi$;

$M,s|=K_i\varphi$ 当且仅当对于任意 $s'\in S$,如果 $s\sim_i s'$,则 $M,s'|=\varphi$;

$M,s|=C_G\varphi$ 当且仅当对于任意 $s'\in S$,如果 $s\sim_G s'$,则 $M,s'|=\varphi$,其中 $\sim_G$ 是 $\bigcup_{i\in G}\sim_i$ 的传递闭包;

$M,s|=[\varphi]\psi$ 当且仅当 $M,s|=\varphi$ 蕴涵 $M|_\varphi,s|=\psi$。

$M|_\varphi=\langle S',V',\sim'_1,\cdots,\sim'_n\rangle$ 也是一个 Kripke 模型,其中

$S'=\{s\in S|M,s|=\varphi\}$,

$V'_p=V_p\cap S'$ 和

$\sim'_i=\sim_i\cap(S'\times S')(i=1\cdots n)$。

模态算子 $[\varphi]$ 的对称算子 $[\varphi]$ 语义是:$M,s|=[\varphi]\psi$ 当且仅当 $M,s|=\varphi$ 并且 $M|_\varphi,s|=\psi$。给定一个模型 M 和一个公式 φ,如果对于 M 中的任一状态 s,均有 $M,s|=\varphi$,则我们称 φ 在 M 上有效(记为 $M|=\varphi$)。

3 和与积问题在公告逻辑中的建模

本节给出和与积问题在公告逻辑中的形式规格。首先要确定这个问题的命题变量和智能体集合。令 x 和 y 是两个整数,满足 $1<x<y$ 和 $x+y\leqslant 100$。定义 $I=\{(x,y)\in N^2|1<x<y$ 并且 $x+y\leqslant 100\}$。因为 x 的最小值是 3,所以 y 的最大值为 98,我们有 $2\leqslant x,y\leqslant 98<2^7$。由此可用 14 个命题变量 $\{x_6,x_5,\cdots,x_0\}$ 和 $\{y_6,y_5,\cdots,y_0\}$ 分别对 x 和 y 进行二进制编码。我们用符号 E_x^i 和 E_y^j 分别代替表示 $x=i$ 和 $y=j$ 的命题公式。例如,如果 x 的值是 13,那么可用命题公式 $E_x^{13}\equiv\neg x_6\wedge\neg x_5\wedge\neg x_4\wedge x_3\wedge x_2\wedge\neg x_1\wedge x_0$ 进行表示。文献[4]用一个命题变量对 x 或 y 的一个值进行编码。例如,$x=13$ 用命题变量 x_{13} 表示。这样的编码方法需要用 198 个命题变量表示 x 和 y 的值。与他们的编码方法相比,我们的更简洁。

智能体 J 只是通过宣告来保证智能体 S 和 P 拥有解决这一问题的背景知识,因此我们不考虑 J 的知识。这样,智能体集合 $A=\{S,P\}$。

命题"S 知道这两个数是 4 和 13"可表示为 $K_S(E_x^4\wedge E_y^{13})$。下面考虑如何将命题"$S$ 知道这两个数"表示为公告逻辑公式。对于"S 知道整数 x"这样的命题,可表示为

$K_Sx\equiv\wedge_{0\leqslant i\leqslant 6}(K_Sx_i\vee K_S\neg x_i)$

也就是说,S 知道所有 x 的编码变量 $x_0,\cdots,x_6$ 的取值。这样,命题"S 知道这两个数(x,y)"可表示为 $K_S(x,y)\equiv K_Sx\wedge K_Sy$。同理有命题"$P$ 知道这两个数(x,y)"可表示为 $K_P(x,y)\equiv K_Px\wedge K_Py$。

S 和 P 的对话(1)-(4)可依次形式地表示为 $\neg K_P(x,y)$,$K_S\neg K_P(x,y)$,$K_P(x,y)$ 和 $K_S(x,y)$。

现在,我们可以在 Kripke 模型 $\Theta\equiv[I,V,\sim_S,\sim_P]$ 上解释上述宣告了,其中 $I=\{(x,y)\in N^2|1<x<y$ 并且 $x+y\leqslant 100\}$。智能体 S 的认知可达关系 $\sim_S$ 定义为:$(x,y)\sim_S(x',y')$ 当且仅当 $x+y=x'+y'$。智能体 P 的认知可达关系 $\sim_P$ 定义为:$(x,y)\sim_P(x',y')$ 当且仅当 $xy=x'y'$。评价函数 V 定义为 $V_{x_i}=\{(x,y)\in I|(x/2^i)\bmod 2=1\}$ 和 $V_{y_i}=\{(x,y)\in I|(y/2^i)\bmod 2=1\}$。评价函数 V 依据二进制编码规则定义了在每一全局状态(x,y)中取真值的命题变量集合。

最后,我们可用模型可满足性公式

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

表示数对(4,13)是和与积问题的唯一解。注意,该模型可满足性公式没有引入 P 的宣告(1)。这是因为 S 的宣告(2)表明 S 在系统初始状态就已经知道 P 不知道这两个数了,并不是在 P 发出宣告(1)后才知道的。而且每一宣告都是事实,因此 P 的宣告(1)是多余的。

4 符号化模型检测知识

本文采用我们在文献[7]中提出的基于局部命题的符号化模型检测算法。该算法是基于有序二叉判定图(OBDD)设计的,很好地缓解了状态爆炸的问题。给定一个 Kripke 模型 $M=[S,V,\sim_1,\cdots,\sim_n]$,我们可用一个带 n 个智能体的有限状态程序 $P_M=[X,\theta(X),\tau(X,X'),O_1,\cdots,O_n]$ 符号化地表示 M,其中

· $X=\{x_1,\cdots,x_k\}$ 是一个命题变量集合,用于编码 M 中的状态,即一个状态是 X 或其子集的一个真值赋值。这样,我们可用 X 上的命题公式表示 S 及其子集。

· 初始条件 θ 是 X 上的布尔公式,对系统可能的初始状态集合进行编码。

· 迁移关系 τ 是 $X\cup X'$ 上的布尔公式,其中 $X'=\{x'_1,\cdots,x'_k\}$ 是另一个命题变量集合。如果 $\tau(X\cup X')$ 在 X 和 X' 的赋值(即状态)s 和 s' 下为 true,则意味着系统从当前状态 s 迁移到下一状态 s'。这样,从满足 $\theta(X)$ 的初始状态开始不断依照满足 τ 的迁移关系展开获得新的状态,得到的可达状态集合就是 M 中的 S。

· 对于任一 $i\in A$,$O_i\subseteq X$ 是智能体 i 的可观察(局部)变量集合。给定一个状态 s,那么 $s\cap O_i$ 就是智能体 i 在状态 s 中的局部状态。

· 假设对于某一命题变量 x_j,如果 $s\in V_{x_j}$,则有 $s(x_j)=$ true。我们在 P_M 中忽略 V。

给定一个带 n 个智能体的有限状态程序 $P_M=\langle X,\theta(X),\tau(X,X'),O_1,\cdots,O_n\rangle$,构造表示 M 中 S 的全局可达状态集合的量化布尔公式:

$$G(P_M)=\mathrm{lfp}Z\left[\theta(X)\vee(\exists X(Z\wedge\tau(X,X')))\left(\frac{X'}{X}\right)\right]$$

其中 $\mathrm{lfp}Z\xi(Z)$是从 X 上的布尔公式到 X 上的布尔公式的运算 ξ 的最小不动点。令 ψ,ψ_0 是 X 上的布尔公式，如果 $\xi(\psi)\Leftrightarrow\psi$，则称 ψ 是 ξ 的一个不动点。如果 ψ_0 是 ξ 的一个不动点，并且对任一 ξ 的不动点 ψ 有 $\psi_0\Rightarrow\psi$，则称 ψ_0 是 ξ 的最小不动点。$\varphi\left(\frac{X'}{X}\right)$是将公式 φ 中的 X' 变量置换为对应的 X 变量后得到的布尔公式。

最后，给定一个公式 φ，根据文献[6]的命题 3，我们有

$M|=K_i\varphi$ 当且仅当 $\forall(X-O_i)(G(P_M)\Rightarrow\varphi)$

这就是符号化模型检测知识的算法。由于 OBDD 支持布尔函数的与、或、非运算以及全称和存在量化运算，而最小不动点 $\mathrm{lfp}Z\xi(Z)$可从一个表示 false 的 OBDD 开始不断迭代计算 ξ 得到，因此很容易将不含模态算子[φ]的公式在 M 中的可满足问题转化为一系列 OBDD 运算。

5 和与积问题的有限状态程序

模型检测器 NuSMV 的输入语言是有限状态机的描述语言，可描述不带智能体的有限状态程序。我们在其中扩展支持智能体的可观察变量声明，使得它可描述带 n 个智能体的有限状态程序。有关 NuSMV 输入语言的介绍，请查阅软件包的附带文档。通过 NuSMV 编译器可将和与积问题的描述语言(图 1)编译为一个带智能体 S 和 P 的有限状态程序 $P_\theta=[X,\theta(X),\tau(X,X'),O_S,O_P]$。这里忽略编译过程的描述。

第 3 行定义表示数对的整型变量 x 和 y。由于智能体 S 和 P 分别只能观察到和 $s=x+y$ 与积 $p=xy$，根据 $5\leqslant x+y\leqslant 195<2^8$ 和 $6\leqslant xy\leqslant 9506<2^{14}$，在第 4 行分别定义整型变量 s 和 p。这样，P_θ 的 X 就是 x,y,s,p 四个变量通过 NuSMV 编译得到的命题变量集合。

第 6,7 行定义 P_θ 的 $\theta(X)$，用于约束 x,y,s,p 四个变量的初始值；第 8,9 行定义 P_θ 的 $\tau(X,X')$，用于约束 x,y,s,p 的值一旦在初始条件下非确定地选择后保持不变。

第 10,11 行分别定义智能体 S 和 P 的描述模块。它们的描述很简单，分别只有一个可观察的形式参数。通过第 5 行的智能体声明，可定义 S 和 P 的可观察变量分别是 s 和 p。显然，P_θ 的 O_S 和 O_P 分别是整型变量 s 和 p 通过 NuSMV 编译得到的命题变量集合。

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

图 1 和与积问题的有限状态程序

6 和与积问题的符号化模型检测

本节将把第 3 节的模型可满足问题 $\Theta|=[K_S\neg K_P(x,y)][K_P(x,y)][K_S(x,y)](E_x^4\wedge E_y^{13})$转化为 P_θ 上的 OBDD 运算。

通过第 4 节的知识公式模型检测算法可知，M 中满足 $K_i\varphi$ 的状态集合是 $\forall(X-O_i)(G(P_M)\Rightarrow\varphi)$ 的 OBDD 表示，记为$[K_i\varphi,M]$。根据第 2 节的公告算子语义 $M,s|=[\varphi]\psi$，可知 $M|_\varphi=(S',V',\sim'_1,\cdots,\sim'_n)$是将 M 约束为满足公式 φ 的 Kripke 模型。显然，$M|_\varphi$的可达状态集合就是 $G(P_M)\wedge[\varphi,M]$。因此，$M|_\varphi,s|=K_i\psi$当且仅当

$\forall(X-O_i)((G(P_M)\wedge[\varphi,M])\Rightarrow\psi)$

该式记为$[K_i\psi,M|_\varphi]$。

这样，在和与积问题中，依次进行三个公告$[K_S\neg K_P(x,y)][K_P(x,y)][K_S(x,y)]$后得到的系统可达状态集合是$[K_S(x,y),\Theta|_{K_S\neg K_P(x,y)}|_{K_P(x,y)}]$的 OBDD 表示。进一步地，我们有

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

就是和与积问题所有解的 OBDD 表示。

我们在 NuSMV 基础上，采用 CUDD(一种高效的 OBDD 软件包)编程实现了该问题的符号化模型检测。实验平台是带 512M 内存的 IBM ThinkPad R50e 笔记本电脑，Ubuntu Linux 系统。实验总的运行时间是 90s，分配了 236521 个 BDD 节点。实验结果是

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

这意味着只有在 x 和 y 初始值分别为 4 和 13 的情况下，三个宣告才能成功进行。换句话说，和与积问题的唯一解是数对(4,13)。这一结果与文献[4]的一致。

为了与类似工作进行性能比较，我们在相同实验平台上用动态认知建模工具 DEMO 对文献[4]中该问题的描述程序进行求解，但是由于运行时间超过 2h 而被迫强行终止。

结束语 本文采用公告逻辑对和与积数迷问题进行建模，将其 Kripke 模型符号化表示为带智能体 S 和 P 的有限状态程序，并在其上采用一种基于局部命题的知识逻辑符号化模型检测算法计算该问题的所有解。我们首先在时态逻辑模型检测器 NuSMV 基础上扩展实现了本文的算法，然后在相同实验平台上用动态认知建模工具 DEMO 对该问题进行求解。实验结果表明，我们的算法不仅结果正确，而且在运行效率上与 DEMO 相比有绝对的优势。关于未来工作，我们将在带 n 个智能体的有限状态程序描述语言中进一步扩展对公告、消息发送和消息群发等通讯动作的支持，并在此基础上设计关于智能体动态知识的符号化模型检测算法，有望在密码协议和基于知识的协议安全性验证中得到广泛应用。

参 考 文 献

[1] Freudenthal H. Formulation of the sum-and-product problem. Nieuw Archief voor Wiskunde, 1969, 3(17):152

[2] Freudenthal H. Solution of the sum-and-product problem. Nieuw Archief voor Wiskunde, 1970,3(18):102-106

[3] Gerbrandy J. Dynamic epistemic logic// L. e. a. Moss, ed. Logic, Language and Information. Stanford : CSLI Publications, 1999,2

[4] van Ditmarsch H P, Ruan Ji, Verbrugge L C. Model checking sum and product // Zhang Shichao, Jarvis R, eds. Australian Conference on Artificial Intelligence. vol 3809 of Lecture Notes in Computer Science, Springer, 2005: 790-795

[5] van Eijck J. Dynamic epistemic modelling. Technical report CWI Report SEN-E0424. Amsterdam: Centrum voor Wiskunde en Informatica, 2004

[6] Fagin R, Halpern J Y, Moses Y, et al. Reasoning about Knowledge [M]. Cambridge, MA: MIT Press,1995

[7] Su Kaile , Sattar A , Luo Xiangyu . Model Checking Temporal Logics of Knowledge via OBDDs. The Computer Journal Advance Access published online on May 15, 2007. Available at: http://comjnl. oxfordjournals. org/cgi/content/full/bxm009?ijkey=joSUmdNT3JHRYz7&keytype=ref

[8] 骆翔宇，苏开乐，杨晋吉. 有界模型检测同步多智体系统的时态认知逻辑. 软件学报，2006，17(12)：2485-2498

[9] Luo Xiangyu, Su Kaile, A Sattar, et al. Verification of Multi-agent Systems via Bounded Model Checking // AI 2006: Advances in Artificial Intelligence, 19th Australian Joint Conference on Artificial Intelligence. Hobart, Australia, December 2006

[10] Bryant R E. Graph-based Algorithms for Boolean Function Manipulation. IEEE Trans Computers,1986, 35(8): 677-691