2003-

质量保证的属性分为"safety"和"liveness"属性。虽然前者说明不应该发生什么（或者换种说法，应该总是发生什么），后者说明最终应该发生的事情。

BMC的基本想法是寻找一个反例来执行，其边界长度是整数k的倍数。如果没有发现错误，边界长度则会增加 k直到发现错误（问题变得难以解决），或者达到一些预先已知的上界

BMC的独特性质：①用户必须提供应探索的周期数的界限，这意味着如果界限不够高，该方法是不完整的。②它使用 SAT 技术，而不是BDDs。

**模型检测：**

模型检测有三个基础特性：①自动化 ②检查的有限系统 ③用时间逻辑来说明系统属性

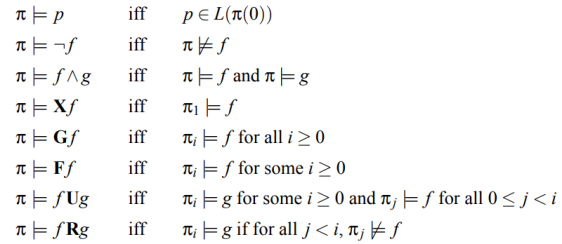
Kripke结构M是四M=（S，I，T，L），其中S是一组状态，I⊆S是初始状态集，T⊆S×S是过渡关系，L:S → P(A)是标记函数，其中A是原子命题集，P（A）表示A上的幂集。标记是将观察结果附加到系统上的一种方式：对于一个属于S的状态s，集合L(s)由包含在s中的原子命题构成。

抽象层次上每个进程有两个程序计数器位置0和1，其中1代表临界段。它可以被编码为一个长度为2的二进制向量s ∈ S = {0,1}2。 因此，S={0,1}2是系统的状态集。

转换关系由几个可能的转换组成，按照以下两个规则:①除非当前状态已经是初始状态，否则下一个状态s'是初始状态00; ②初始状态可以来回转换到01和10。 因此，转换关系T⊆S2 = {0,1}4可以用以下一组位串表示:

{0100, 1000, 1100, 0001, 0010}

根据Kripke结构的路径定义时态公式的形式语义。设π是一个Kripke结构M的无限路径，设f是一个时间公式。 当f保持在π上时，我们递归地定义π |= f：



**BMC：**

**Definition 1.**对于l≤k，当T(π(k),π(l))且π= u•vw其中u = (π(0),...,π(l −1))且v = (π(l),...,π(k))3。如果存在k ≥ l ≥ 0我们称π为一个k-loop，其中π是(k, l) -loop。

**Definition 2 (Bounded Semantics for a Loop).**设k≥0,π是k-loop。当且仅当π|= f(π|=kf )那么LTL公式f在边界为k的路径π上是有效的。

**Theorem 1.**设f为LTL公式，M为Kripke结构。那么M |= Ef如果存在k≥0 可使M|=kEf。

注：这篇论文核心内容是第3章和第4章，你好像没有重点关注？

2015-

**介绍：**

①大多数优化技术只考虑safety属性，忽略了同样重要的liveness属性。 ②许多BMC优化技术直接重用现有的显式抽象技术，通过引入额外的控制变量将他们编码到BMC实例中，只得到了性能增强。 ③关于基于多核的BMC优化，经常发生的是单片BMC实例直接交付给内部支持多核计算的SMT求解器。但是核之间交流的通常与否会明显影响到整体表现。

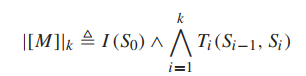
对BMC优化来说，简单胜过复杂

关键思想是利用SESE遍历目标系统的状态空间，并在每个时间步骤(即执行深度)记住可执行的转换，直到用户指定的边界

**BMC：**

系统可以用Kripke结构表示，型如M = (S，I，T，L)，S是系统状态集，I是初始状态集并I ⊆ S，T ⊆ S × S是转换关系（不完全），L是标记函数。

定义Kripke结构M，LTL公式f以及界限k，BMC的基本思想是构造命题式 |[M, f, k]|，当且仅当在k步中有一m违反了f这命题式是可满足的。  
[|M, f, k]|定义为|[M]|k ∧ |[¬ f ]|k



I(S0) 是一个状态变量的谓词，定义初始状态S0；Ti(Si−1, Si) 是M的转换关系，是一个命题公式。因此，|[M]|k表示系统从初始状态到指定边界k的所有执行路径。

2008-

**FSMActor：**

四元组 Afsm = (Qfsm, Pfsm, Parafsm, Tfsm)  
Qfsm 是状态集，其中具有参数(initial,true)的状态q0 = {φ, Pq0, Paraq0,φ} ∈ Qfsm为FSMActor的初始状态。  
Pfsm是端口集。  
Parafsm内部变量及其对应初始值的集合，包含元素形为(Vari, IniVali)。  
Tfsm是迁移集，Tfsm = {(ps, Parasd, pd)| ps ∈ Ps, pd ∈ Pd, qs = {φ, Ps, Paras, φ} ∈ Qfsm,   
qd = {φ, Pd, Parad, φ} ∈ Qfsm} qs∈Qfsm是源状态，qd∈Qfsm是目标状态。

迁移所满足性质  
当警戒条件expguard满足时，迁移会触发；

输出行为actoutput指定了目标端口和值；

设定行为actset更新了内部变量的值。

初始化：preinitialize() 和 initialize()，每次执行调用一次

执行： prefire(), fire(), 和 postfire()，每次SR导演执行一次  
显示： wrapup()，在每次执行的最后调用一次来产生结果

当我们将FSMActor转化为Kripke结构，所得的状态数量不能为|Qfsm|

guardExpression 决定了可能触发转换的变量值

setAction 决定了转换后内部变量值

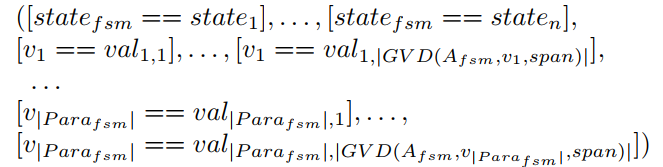
**转换FSMActor到Kripke：**

**原子命题集**

FSM所有状态并集 FSM变量count的并集

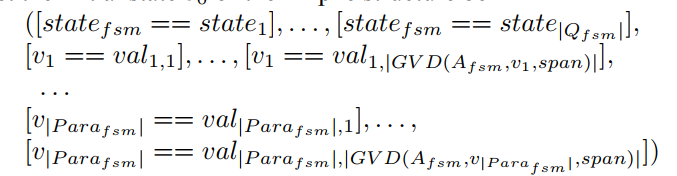
**状态集**

2的幂？



**标记函数L**

L：S→S



**迁移关系R**

对于∀r = (s, s0) ∈ S × S,若满足下述条件时，有 R = R ∪ {r}

①状态s满足其中一个关于statefsm的原子，即

∃1 i, 1 ≤ i ≤ |Qfsm| ，s.t. 在 s中,有[statefsm == statei] == 1

②状态s'满足其中一个关于statefsm的原子，即

状态 s’满足其中一个关于statefsm 的原子，即

∃1 i', 1 ≤ i '≤ |Qfsm| ，s.t. 在 s'中,有[statefsm == statei] == 1

③状态s中和s'中所有变量都各自满足其中一个关于其取值的原子：

∀j, 1≤ j ≤ |Parafsm|,∃1 k, k', 1 ≤ k, k' ≤ |GVD(Afsm, vj , span)|,s.t.

在s中,有[vj == valj,k] == 1；

在s'中，有[vj == valj,k' ] == 1,④Tfsm中存在符合下述条件的迁移，即∃t = (ps, Parasd, pd) ∈ Tfsm：  
 (a) ps ∈ Ps, pd ∈ Pd, statei = {φ, Paras, Ps, φ} ∈ Qfsm,

statei' = {φ, Parad, Pd, φ} ∈ Qfsm,

(b) ∀j, 1 ≤ j ≤ |Parafsm|, expguardj(valj,k) == 1,and actsetj(valj,k) == valj,k' ;

**SR模型转化：**

假设有两个FSMActor：Afsm1 和 Afsm2

**原子命题集** AP = AP1 ∪ AP2

**状态集** S12 = S1 S2

**标记函数** S12 → S12

**迁移关系R** 前提：相连接的两个端口的名字相同

都得满足statefsm1 和 statefsm2 中的原子