文章编号：0253-374X(2016)11-1702-07 DOI:10.11908/j.issn.0253-374x.2016.11.009

基于因果关系的列控系统模型约简方法

周庭梁[1,2]，许 婧[2]，陈小红[3]，赵时旻[1]

(1.同济大学 道路与交通工程教育部重点实验室，上海 201804；2.卡斯柯信号有限公司，上海 200071；
3.华东师范大学 上海市高可信计算重点实验室，上海 200062)

摘要：在基于安全需求对验证问题进行投影的方法基础上，针对投影出的验证子问题，提出了基于因果关系的变量约简方法，定义了环境变量间的因果关系，归纳出基本的因果关系组合，并提炼出变量约简规则，通过变量约减减少了验证问题的状态空间.采用国内某地铁线路的相关数据进行建模和验证，结果表明，该方法能够有效降低系统验证复杂度.

关键词：需求验证；变量约简；因果关系；列车运行控制系统

中图分类号：TP311 **文献标志码**：A

Automatic Train Control System Model Reduction Based on Causal Relation

ZHOU Tingliang[1,2], *XU Jing*[2], *CHEN Xiaohong*[3], *ZHAO Shimin*[1]

(1. Key Laboratory of Road and Traffic Engineering of the Ministry of Education, Tongji University, Shanghai 201804, China; 2. CASCO Signal Ltd., Shanghai 200071, China; 3. Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai 200062, China)

Abstract: Based on the previous work about verification problem projection according to the safety requirements, a variable reduction approach was proposed based on causal relation for the projected sub-problems. First, the causal relations among the environment variables of the projected sub-problems were defined. Then, the basic causal relation combination of variables and the reduction rules were concluded. Through variable reduction, the state space of the verification problem was reduced. Finally, with configuration of a domestic metro line, an experiment of modeling and verification was demonstrated to show that the variable reduction approach efficiently reduces the verification complexity.

Key words: requirement verification; variable reduction; causal relation; automatic train control system

作为一种安全攸关系统，列车运行控制系统(简称列控系统)必须经过验证.目前列车运行控制系统的建模和验证方法主要有以下3种：① 基于逻辑的方法.这类方法基于集合论和一阶逻辑，采用逻辑推理或者定理证明的方式精化系统的功能，证明系统的正确性，典型代表是Z，VDM，B和Event-B方法[1]等，其中B方法已经在法国巴黎RER线路所采用的SACEM系统[2]中得到成功应用.② 基于进程代数的方法.进程代数关注并行行为，强调对不同模块并发过程之间的交互进行建模.例如，Zou等[3]调研了如何对CTCS-3(Chinese Train Control System 3)的系统需求规约(system requirement specification，SRS)进行形式化的描述和验证.③ 基于形式化模型与规约的方法.在列控系统的需求阶段，多种类型的模型被用于系统建模和验证过程中.例如，基于有色Petri网(coloured Petri nets，CPNs)，Horste等[4]对欧洲列控系统ETCS进行了针对系统功能的形式化描述.

虽然现有的这些需求建模方法能有效避免二义性，并且能成功实现对需求规约的验证，但是这些方法并没有考虑到环境的影响.列控系统所处环境的复杂性以及环境自身的其他特性，会带来系统模型验证空间急剧爆炸的问题，导致在复杂环境中难以保证系统的可信度.因此，在需求模型的可信构造和验证过程中，需要研究如何对环境相关的约束进行调整，从而提高复杂环境下列控系统模型的可验证性.

为了解决这个问题，组合验证方法应运而生[5].组合验证方法先将系统分解为子系统，然后分别验证各子系统，最后分析集成的系统.这类方法在很多

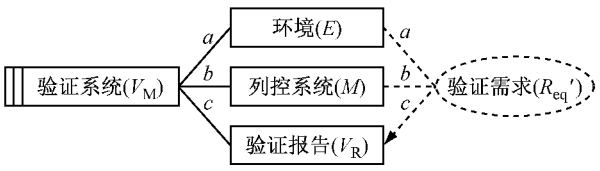
收稿日期：2016-03-29
基金项目：国家自然科学基金(91418203)
第一作者：周庭梁(1980—)，男，博士生，主要研究方向为轨道交通列车列控系统研制、测试及安全评估.E-mail:Leon_ztl@163.com

大型系统中得到成功的应用，例如 McMillan[6-7] 将该方法应用于微处理器的乱序执行单元和多处理机的缓存一致性协议. 但是传统的组合验证的分解方法只能对系统进行划分，不允许子系统中存在重叠的部分. 然而在列控系统中，由于环境共享，不可避免地存在子系统之间的部分重叠. 基于问题框架方法的投影[8]是一种有效的处理有重叠部分的分解方法. 它能够对复杂的问题通过投影进行分解，降低大型系统的复杂度.

基于问题框架方法，本文作者提出了一种基于安全需求的验证问题投影方法[9]，以列车控制验证系统的安全子属性作为投影维度，将列控系统验证问题投影为若干子问题. 该方法可以有效地降低复杂度，但还是不能完全避免验证问题状态空间过大的问题. 在研究投影出的子问题时，发现在列控系统中，变量之间存在因果关系，而变量的减少可以有效减小状态空间. 因此，本文通过找出存在因果关系的变量，并根据不同的因果关系类型对这些变量进行约简，进一步降低验证系统的状态空间. 最后采用国内某地铁线路的相关数据进行建模和验证，结果表明，该方法可以有效地降低验证系统的复杂度，降低形式化模型验证中的状态空间爆炸问题发生的可能性，并且提升了形式化验证的效率.

1　列控系统的需求模型验证问题及投影

文献[11]对列控系统的验证及需求问题进行了分析，根据问题框架理论[10]得出了列控系统验证问题图(图 1)，并将其定义为一个问题 P，通过一个 5 元组来表示：$P \stackrel{\text{def}}{=} <M, E, V_{\mathrm{M}}, I_{\mathrm{S}}, R'_{\mathrm{eq}}>$. 其中 V_{M} 是列控系统的验证系统，E 是验证过程中系统运行的环境，M 是需要验证的列控系统，R'_{eq} 是验证需求，要求 V_{M} 能根据安全需求 R_{eq} 判断 M 是否通过验证，V_{R} 是记录验证结果的报告，I_{S} 是验证时 M 与问题领域交互的集合.



$I_{\mathrm{S}}=a \cup b \cup c$　a: E! {$p_{11}, p_{12}, \cdots, p_{mn}$}　b: M! {$q_{11}, q_{12}, \cdots, q_{mn}$}
c: V_{M}! {true, false}

图 1　列控系统验证问题

Fig.1　Verification problem of automatic train control system

在这些元素中，环境(E)实际上是与列控系统进行交互的设备集合，每个设备可以用一组变量描述. 因此，本文将 E 定义为一组设备变量 D_i ($0 \leqslant i \leqslant n$) 的集合，而每个设备变量又有一组属性变量 V_{ij} ($0 \leqslant i \leqslant n, 0 \leqslant j \leqslant m$) 表示，可形式化定义为 $E=\{D_1, D_2, \cdots, D_i, \cdots, D_n\}$，其中，$D_i=\{V_{i1}, V_{i2}, \cdots, V_{ij}, \cdots, V_{im}\}$.

E 与 M 有共享的交互集合 a 和交互集合 b，其中 a 为由 E 控制的交互 p_{ij} ($0 \leqslant i \leqslant n, 0 \leqslant j \leqslant m$) 的集合，$b$ 为由 M 控制的交互 q_{ij} ($0 \leqslant i \leqslant n, 0 \leqslant j \leqslant m$) 的集合. 在列控系统问题中，因为 E 和 M 间共享的是设备的属性变量，所有的交互都是值交互，所以可以将交互定义为由设备的属性变量的取值所确定的函数值，因此将与设备 D_i 的属性变量有关的交互 p_{ij}，q_{ij} 定义为，$p_{ij}=f_{ij}(V_{i1}, V_{i2}, \cdots, V_{im})$，$q_{ij}=g_{ij}(V_{i1}, V_{i2}, \cdots, V_{im})$.

安全需求(R_{eq})描述了 M 需要满足的条件. R_{eq} 引用了交互 b，这是一个约束引用，约束环境的行为必须按照安全需求所规定的方式改变. 安全需求由一组安全属性构成，只有当每个安全属性 P_i ($0 \leqslant i \leqslant n$) 都成立时，安全需求 R_{eq} 才能成立. 安全需求定义为(其中 $\wedge$ 表示与)：$R_{\mathrm{eq}}=P_0 \wedge P_1 \wedge \cdots \wedge P_i \wedge \cdots \wedge P_n$.

在这些定义基础上，提出了基于安全需求对列控系统的验证问题进行投影[9]，将原验证问题投影成多个子问题进行验证，从而得到了新的子问题的验证系统、验证环境和验证需求. 由于安全需求 R_{eq} 与问题的 5 个描述元素都有关系，所以可将 R_{eq} 作为投影的维度. 因此，可以基于安全子属性对整个验证问题进行投影. 采用关系代数类似的表达，问题投影的形式可以表示如下：

$$\pi_{P_j}(P) = S_P \tag{1}$$

式中：P 是对系统 M 的验证问题；P_j 是安全子属性；S_P 是问题 P 的子问题. 另外，为投影出子问题，定义了 4 个辅助投影算子：环境投影算子 $\pi_{P_j}(E)$、验证系统投影算子 $\pi_{P_j}(M)$、交互投影算子 $\pi_{P_j}(I_{\mathrm{S}})$ 和验证需求投影算子 $\pi_{P_j}(R'_{\mathrm{eq}})$.

在这 4 个辅助算子的基础上，结合系统 M，可以定义列控系统形式化验证问题的投影为

$$S_P=\pi_{P_j}(P)=<M, \pi_{P_j}(E), \pi_{P_j}(V_{\mathrm{M}}), \pi_{P_j}(I_{\mathrm{S}}), \pi_{P_j}(R_{\mathrm{eq}}')> \tag{2}$$

在对投影结果进一步分析时发现，在对子问题的环境进行投影时只是将与待验证的安全需求无关的设备进行约简，保留下所有与待验证的安全需求

相关的设备参数.因此,只要与待验证的安全需求产生了交互,该设备的所有变量都会被加入到环境中,但并不是所有加入的变量都真正会对验证结果产生影响.因此,本文提出基于因果关系的约简方法,尝试找出设备内部的各个变量间可能存在的关系,并基于变量间的因果关系进一步降低系统的复杂度,从而进一步减小系统的状态空间,提高系统的验证效率.

2 基于因果关系的变量约简

2.1 列控系统的变量因果关系分析

在对列控系统验证问题进行投影后,将原验证问题投影成多个子问题进行验证,能有效地降低系统验证复杂度[11].但在投影后,只要与验证需求产生了交互,该设备的所有变量都会被加入到环境中.但事实上,并不是所有变量都会对验证结果产生影响[12],这为进一步约简验证系统提供了可能.

在未对变量进行任何约简前,环境中所有设备的属性变量都与验证系统共享.分析各变量间的关系后可以发现,在某些设备内部的变量之间存在直接的因果关系.

设 C, R 为 E 中的变量或者 E 中变量间任意的与、或组合.根据因果关系理论,因果关系共分为4种:① $C\rightarrow R$;② $\neg C\rightarrow R$;③ $C\rightarrow\neg R$;④ $\neg C\rightarrow\neg R$.

在情况①中,即当 C 存在时,R 一定存在,且 R 可由 C 推导得出,因此可设 $R=S(C)$,此时可直接用 C 来表示 R.

例如,在描述列车车尾的位置时,有两个变量:最大车尾 MaxTrainTailPos 和最小车尾 MinTrainTailPos.最大车尾和最小车尾之间的距离固定为 TailLength,所以有

$$\text{MaxTrainTailPos}=\text{MinTrainTailPos}+\text{TailLength}$$

则 MinTrainTailPos → MaxTrainTailPos 可表示为 MaxTrainTailPos = S(MinTrainTailPos).

此时,所有与 MaxTrainTailPos 相关的变量均可由 S(MinTrainTailPos)替代.

在情况②中,当 C 不存在时,R 一定存在.在情况③中,当 C 存在时,R 一定不存在.在列控系统的环境中,设备的属性变量很少出现互斥的关系,所以在列控系统中,这两种因果关系很少存在.即使存在,在情况②中,由于 C 不存在,R 无法由 C 推导得出,所以无法约简.在情况③中,由于 R 本身就不存在,所以同样无法约简.在情况④中,当 C 不存在时,R 一定不存在.这种情况下,由于 C,R 都不存在,这种因果关系对约简状态空间没有实际意义.

总结上述4种情况分析,只有 $C\rightarrow R$ 这类因果关系会对列控系统验证过程中的变量约简产生直接影响.由于系统验证时会对所有变量进行全状态空间的遍历,可用 $R=S(C)$ 替代 R,在 R 的取值范围较大的情况下可明显减小状态空间,提升系统的验证效率.由于变量的因果关系不一定是一一对应的,也可能存在多个变量之间的因果关系.设 C, R 为 E 中的变量或者 E 中变量间任意的与、或组合,由上文可知,只有 $C\rightarrow R$ 类因果关系能用于变量约简.

2.2 列控系统的因果关系定义

为了表达出所有能用于变量约简的因果关系,基于巴克斯范式(Backus-Naur Form,BNF)[13]对比类因果关系进行如下定义:

<因果关系> ::= <变量表达式> → <变量表达式>

<变量表达式> ::= <变量> | <变量表达式> <运算符> <变量表达式>

<运算符> ::= ∧ | ∨

<变量> ::= V_1 | V_2 | … | V_i | … | V_n

变量间的组合型因果关系有7种最基本的情况,设 V_i, V_j, V_k, $V_l\in E$,这7种基本情况分别为

$$\begin{cases} V_i\rightarrow V_l \\ V_i\wedge V_j\rightarrow V_l \\ V_i\vee V_j\rightarrow V_l \\ (V_i\wedge V_j)\vee V_k\rightarrow V_l \\ (V_i\vee V_j)\wedge V_k\rightarrow V_l \\ V_i\rightarrow V_j\wedge V_l \\ V_i\rightarrow V_j\vee V_l \end{cases} \tag{3}$$

(1) $V_i\rightarrow V_l$

由上文可知,这种情况下可以用 $V_l=S(V_i)$ 替代 V_l.

(2) $V_i\wedge V_j\rightarrow V_l$

需要同时知道 V_i 和 V_j,才能推导出 V_l.这种情况下 V_l 可表示为 $V_l=S(V_i, V_j)$.

(3) $V_i\vee V_j\rightarrow V_l$

只需知道 V_i 和 V_j 中任意一个变量,就能推出 V_l.这种情况下 V_l 可表示为 $V_l=S(V_i)$ 或 $V_l=S(V_j)$,从中任取一种即可.从变量约简的角度看,替换 V_l 时选择 V_i 和 V_j 没有区别,所以可以比较 $V_l=S(V_i)$ 和 $V_l=S(V_j)$,从中选择较为简单的一种表示

方法对 V_l 进行替换.

(4) $(V_i \wedge V_j) \vee V_k \rightarrow V_l$

这种情况下，需要知道 $V_i \wedge V_j$ 或者 V_l，就能推出 V_l. 则 V_l 可表示为 $V_l = S(V_i, V_j)$ 或 $V_l = S(V_k)$，从中任取一种即可. 由于 $V_l = S(V_k)$ 的表示方法更简洁，用 $S(V_k)$ 替换 V_l 会使之后的验证过程更为简单.

(5) $(V_i \vee V_j) \wedge V_k \rightarrow V_l$

这种情况下，需要知道 $V_i \wedge V_k$ 或者 $V_j \wedge V_k$，才能推出 V_l. 所以 V_l 可表示为 $V_l = S(V_i, V_k)$ 或 $V_l = S(V_j, V_k)$，从中任取一种即可.

(6) $V_i \rightarrow V_j \wedge V_l$

在这种情况下，只需知道 V_i，就可以推出 V_j 和 V_k，因此等价于 $V_i \rightarrow V_j$ 且 $V_i \rightarrow V_k$.

(7) $V_i \rightarrow V_j \vee V_l$

在这种情况下，因无法确定推出的结果是 V_j 还是 V_k，所以无法进行约简. 如果有其他信息能辅助判断推出的结果具体为哪个变量，则可以将问题转化为 $V_i \rightarrow V_j$ 或者 $V_i \rightarrow V_k$ 进行处理.

2.3　变量表达式的约减规则

在提炼出的 7 种基本情况中，情况(1)是一对一的情况，即一种原因推出一个结果；情况(2)至(5)是多对一的情况，即多种原因可以推出一个结果. 这 5 种情况中变量 V_l 都可以由其他变量推出，因此可以用其他变量进行替代. 情况(6)和(7)是一对多的情况，用一个变量能够推导出多个结果，即变量 V_j 和 V_k 都可以用变量 V_i 推导得到，因此在变量约简后，可以用更少的变量来定义交互，由此可以降低验证复杂度. 除了上述 7 种最基本的情况外，式(3)所定义的其他情况都是这 7 种情况的组合，并可以通过依次迭代最终化简为这 7 种情况中的一种，再对变量进行约简.

设 C，C_1，C_2，C_3，R，R_1，R_2 是式(3)所定义的变量表达式，通过对这 7 种最基本情况的分析，可以总结出 4 个约简规则：

$$① \ C_1 \vee C_2 \rightarrow R \Rightarrow \begin{cases} C_1 \rightarrow R \\ C_2 \rightarrow R \end{cases} \tag{4}$$

根据基本情况(3)可知，只需知道 C_1 和 C_2 中任意一个变量表达式，就能推出 R，所以原因中的 $C_1 \vee C_2$ 可以直接用 C_1 或 C_2 替换.

$$② \ C \rightarrow R_1 \wedge R_2 \Rightarrow \begin{cases} C \rightarrow R_1 \\ C \rightarrow R_2 \end{cases} \tag{5}$$

根据基本情况(6)可知，只需知道 C，就可以推出 R_1 和 R_2，因此等价于 $C \rightarrow R_1$ 且 $C \rightarrow R_2$，可以拆成这两种情况分别进行讨论.

$$③ \ (C_1 \wedge C_2) \vee C_3 \rightarrow R \Rightarrow C_3 \rightarrow R \tag{6}$$

根据基本情况(4)可知，只需要知道 $C_1 \wedge C_2$ 或者 C_3，就能推出 R. 由于用 C_3 推导 R 会更为简洁，所以 $(C_1 \wedge C_2) \vee C_3$ 直接用 C_3 替换.

$$④ \ (C_1 \vee C_2) \wedge C_3 \rightarrow B \Rightarrow \begin{cases} C_1 \wedge C_3 \rightarrow R \\ C_2 \wedge C_3 \rightarrow R \end{cases} \tag{7}$$

根据基本情况(5)可知，只需要知道 $C_1 \wedge C_3$ 或者 $C_2 \wedge C_3$，就能推出 R. 所以 $(C_1 \vee C_2) \wedge C_3$ 可用 $C_1 \wedge C_3$ 或者 $C_2 \wedge C_3$ 替换.

在对列控系统的变量进行分析时，如果变量之间存在可以用于约简的因果关系，则一定在式(4)所定义的范围内. 而式(4)所定义的因果关系，又可以通过上述 4 个化简规则转化成最基本的 7 种变量间的组合型因果关系之一进行处理，所以对所有可能情况都可以进行相应的变量约简操作. 在约简的过程中，由于交互并未发生改变，但验证过程中实际使用到的变量个数减少了，从而降低了系统验证复杂度.

3　实例分析

以卡斯柯信号有限公司的轨旁列控系统区域控制器 iZC 为例，应用基于因果关系的约简方法，降低系统验证的复杂度. 与上一代列控系统有所不同，CBTC 系统通过移动闭塞来追踪并保护列车. iZC 根据轨道上各列车的精确位置，来计算各列车之间的安全区间，并通过无线车地通信传递给列车. 这些安全区间被称为移动授权(MA). MA 属于列控系统的核心安全功能，其安全等级达到 SIL4 要求，必须通过形式化验证来确保该功能的正确性及安全性.

iZC 会在每个计算周期内为其管辖范围内的所有列车计算移动授权 MA 的范围，起点一般为列车的最小车头，计算得到的终点称之为 EOA(end of authority). EOA 是指授权列车移动的最大距离，EOA 的计算依赖于列车自身的位置、速度，同时还依赖于轨道上的信号设备和特殊区段，例如包括信号机、缓冲区、重叠区等.

根据 IEEE 1474.1 标准[14]，当遇到 8 种特殊情况时列车 EOA 计算将会终止，例如列车前方出现另外一辆 CBTC 列车、缓冲区、重叠区等等. 这些情况可能导致列车运行发生事故，因此这些情况也被称为非安全状态点. 根据这 8 种情况，本文将 EOA 分为 8 种类型. 同时，在 iZC 发给列车的 MA 报文中，

包含 EOA_TYPE 字段,主要描述选取当前点作为 EOA 的原因.

3.1 EOA 验证问题描述

EOA 模型的形式化验证问题可以表示为:

ProblemEOA=<EOA, EnvEOA, VeriEOA, IntEOA, SafeR′>

其中,EOA 为待验证的系统模型,共有 8 种类型,在验证过程中为"黑盒",不可更改. EnvEOA 为问题所处的环境,包含车、闭塞、分支、信号机、BZ(buffer zone)、OL(overlap)、TD(traffic direction)等,即

EnvEOA=<Train, Block, Branch, Signal, BZ, OL, TD>

VeriEOA 为需要创建的 EOA 验证系统. IntEOA 为 EnvEOA 与各问题领域交互的集合 Int1∪Int2∪Int3. SafeR′为 EOA 的验证需求,验证 EOA 的安全需求 SafeR= $P_0 \wedge P_1 \wedge P_2$.

针对不同的 EOA 系统类型 j,每个安全需求 P_i 仅需满足对应的安全子属性 P_{ij},因此有

SafeR=$P_{01} \wedge P_{02} \wedge \cdots \wedge P_{08} \wedge P_{11} \wedge \cdots \wedge P_{18} \wedge P_{21} \wedge \cdots \wedge P_{28}$

由于系统 EOA 不可变动,所以投影后结果不会变化. 另外的 4 个元组可以借助投影算子分别进行投影. 下面以第 8 种类型 EOA 为例进行投影,并对投影后的子问题进行约简. 该类型 EOA 表示非安全状态点为重叠区,即前方搜索到闭塞轨道边界. 根据文献[9]中的投影方法,以安全子属性 $SafeR_8$ 为投影维度,对 ProblemEOA 进行投影,投影结果为 $SProblemEOA_8 = \pi_{P_8}(ProblemEOA) = <EOA, EnvEOA_8, VeriEOA_8, IntEOA_8, SafeR'_8>$.

其中,EOA 的验证系统 VeriEOA 的投影,是一个满足安全需求 $SafeR_8$ 的验证子系统 $VeriEOA_8$. 环境投影结果为 $EnvEOA_8 = \pi_{P_8}(EnvEOA) = <Train, Block, Branch, OL>$. 交互的投影是投影后的验证系统 $VeriEOA_8$ 与其他投影后的问题领域交互的集合,$IntEOA_8 = Int1_j \cup Int2_j \cup Int3_j$. 验证需求投影为 $SafeR'_8 = \pi_{P_8}(SafeR') = P'_8 \wedge P'_{18} \wedge P'_{28}$,其中 P'_8,P'_{18},P'_{28}是验证的安全子属性.

$SProblemEOA_8$ 是由原问题 ProblemEOA 投影出的子问题. 安全需求 $SafeR_8$ 是原问题安全需求 SafeR 的子集,验证时可直接跳过与当前类型无关的安全需求的验证过程,因此,根据投影后的安全需求设计出的验证系统的规模也相应减小.

3.2 EOA 验证变量约简

列控系统一般采用两种坐标定位方式. 一种是基于闭塞(block)的,例如一组道岔的位置可以表示为一个二元组(Block_Index, OffsetOnBlock),其中 Block_Index 表示道岔所处的闭塞索引,OffsetOnBlock 表示精确的位置偏移. 另外一种是基于分支(branch)的,分支是一组闭塞连接而成的连续轨道,一组道岔的位置可以表示为一个二元组(Branch_Index, OffsetOnBranch),其中 Branch_Index 表示道岔所在的分支索引,OffsetOnBranch 表示在该分支上的精确的位置偏移. 因此,所有的基于闭塞的位置坐标都可以转化为基于分支的位置坐标.

以列车内部变量关系为例,列车位置由 4 个坐标决定:最大(MaxTrainHeadPos)/最小(MinTrainHeadPos)车头位置和最大(MaxTrainTailPos)/最小(MinTrainTailPos)车尾位置. 值域[MinTrainHeadPos, MaxTrainHeadPos]代表列车车头的可能位置. 值域[MinTrainTailPos, MaxTrainTailPos]代表列车车尾的可能位置. 同时,每辆列车被一个列车在轨道上的虚拟占用区域(VTP)包络. iZC 系统被设计用于计算 VTP 的碰撞概率,并防止这些 VTP 发生碰撞. 由于 VTP 的位置与列车的位置相互关联,VTP 的位置变量与列车的位置变量会相互影响. VTP 位置也由 4 个坐标决定:最大(MaxVTPHeadPos)/最小(MinVTPHeadPos)VTP 头位置和最大(MaxVTPTailPos)/最小(MinVTPTailPos)VTP 尾位置,如图 2 所示.

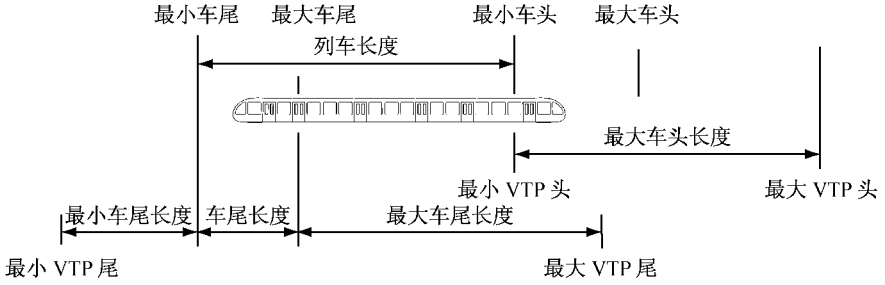


图 2 列车定位相关坐标点

Fig.2 Coordinating points related to train location

另外还有 1 个列车状态变量（TrainMonitorModes）用来表示列车当前的运行模式. 由此可知，

Train={MinVTPTailPos, MaxVTPTailPos,
MinVTPHeadPos, MaxVTPHeadPos,
MinTrainTailPos, MaxTrainTailPos,
MinTrainHeadPos, MaxTrainHeadPos,
TrainLength, TrainMonitorModes}

在问题投影过后生成的子问题 $SProblemEOA_8$ 中，列车的位置坐标点全部随机，即它们可能为任意一个 block 中的任意一个点. 但实际情况中，车的位置坐标之间存在几组等价关系，由图 2 可以推导出：

（1）最小 VTP 尾与最小车尾间距离为固定值 MinTailLength，所以

MinTrainTailPos → MinVTPTailPos

（2）最小车尾与最小车头之间距离为固定长度，所以

MinTrainTailPos ∧ MinTrainHeadPos→
TrainLength

（3）最小车头与最小 VTP 头的位置一致，所以

MinTrainHeadPos → MinVTPHeadPos

（4）最大 VTP 头与最小 VTP 头间距离为固定值 MaxHeadLength，所以

MinVTPHeadPos→MaxVTPHeadPos

又因为 MinTrainHeadPos→
MinVTPHeadPos

所以

MinTrainHeadPos→MaxVTPHeadPos

（5）最大车尾与最小车尾间距离为固定值 TailLength，所以

MinTrainTailPos → MaxTrainTailPos

（6）最大车尾与最大车头间距离为车长，所以

MaxTrainTailPos ∧ TainLength→
MaxTrainHeadPos

又因为 MinTrainTailPos → MaxTrainTailPos

MinTrainTailPos ∧ MinTrainHeadPos→
TainLength

所以

MinTrainTailPos ∧ MinTrainHeadPos→
MaxTrainHeadPos

（7）最大车尾与最大 VTP 尾间距离为固定值 MaxTailLength，所以

MaxTrainTailPos → MaxVTPTailPos

又因为 MinTrainTailPos → MaxTrainTailPos

所以

MinTrainTailPos → MaxVTPHeadPos

由此，一旦确定了最小车尾和最小车头的具体坐标，就能得到车长，另外 6 个列车的相关坐标点位置也能相应固定. 所以 Train 可以约简为

Train′={MinTrainTailPos,
MinTrainHeadPos, TrainMonitorModes}

此时，Train 中的 10 个变量可约简为 Train′中的 3 个变量，系统验证状态空间得到了极大的约简，有效降低了验证复杂度.

3.3　EOA 验证实验及分析

以图 3 所示的国内某地铁线路数据作为环境，线路中共有 6 个 block，并涵盖 2 个道岔. 在此环境下，基于 SCADE 工具直接对 iZC 系统模型进行验证.

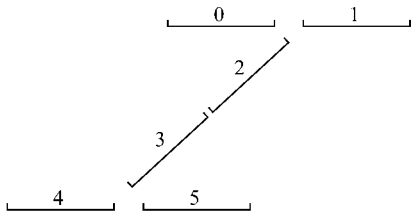


图 3　验证环境

Fig. 3　Verification environment

按照文献[5]中的投影方法，基于安全需求对验证问题进行投影后，iZC 系统顺利通过了形式化验证. 图 4 展示了投影生成的 $SProblemEOA_8$ 的验证结果. 由图 4 的结果可以看出，复杂环境下系统验证状态空间仍然较大，形式化验证耗时较长.

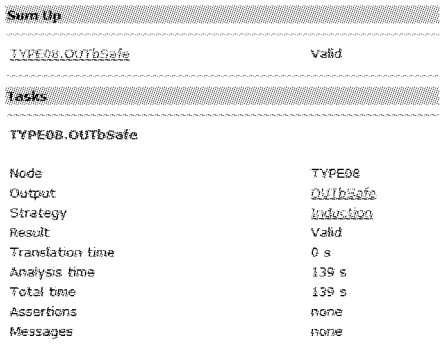


图 4　投影后系统的验证结果

Fig. 4　Verification result after projection

按照第 2 节中的约简方法，基于因果关系对变量进行约简后，环境输出的变量减少，系统的复杂度进一步降低，状态空间进一步缩小，形式化验证的效率得到提高. 由于消耗时间可能与工具运行状态有关，可能存在一定的随机性，所以每种类型在约简前后均多次验证，最终结果取多次验证所耗时间的平

均值进行分析. 变量约简前后每种类型的待验证系统进行验证所耗的时间如图5所示.

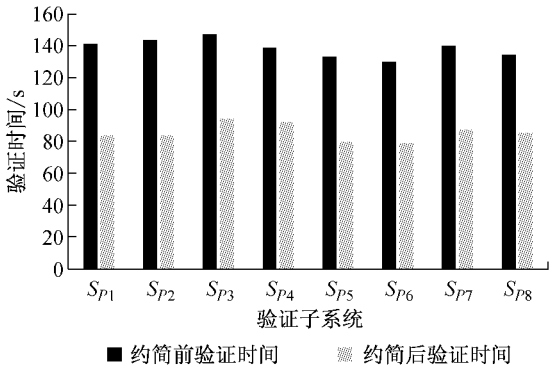


图5 变量约减前后验证效率对比

Fig.5 Comparison of verification efficiency before and after variable reduction

由实验结果可知,对于系统过于复杂,验证时间较长的iZC系统,在对每种类型的验证子系统分别进行验证时,基于因果关系对变量进行约简,使验证所需的时间缩短,验证效率得到了有效提高. 由此在对整个iZC系统进行验证时,所需的总体时间明显缩短,验证效率显著提升.

4 结论

本文针对列控系统的特点,在对列控系统进行形式化验证的过程中,提出可以利用列控系统各设备变量间的因果关系对验证系统进行化简,形成基于因果关系的列车运行控制系统模型约简方法. 通过分析变量间的因果关系,在基于安全需求的列控系统投影方法的基础上,针对投影得到的子系统,根据设备内各变量之间存在的因果关系进一步对环境的输出变量进行了约简,从而减少了验证系统的状态空间.

本文以轨旁列控系统iZC的安全功能EOA为例,进行EOA形式化验证问题的投影及变量约减,并对8种EOA类型的验证子系统分别进行验证. 实验结果表明,采用基于因果关系的列控系统模型约简方法,能够有效地提高形式化验证的效率.

参考文献:

[1] Abrial J R. Modeling in Event-B: system and software engineering[M]. New York: Cambridge University Press, 2010.

[2] DaSilva C, Dehbonei B, Mejia F. Formal specification in the development of industrial applications: subway speed control system[C]//Proceedings 5th IFIP Conference on Formal Description Techniques for Distributed Systems and Communication Protocols (FORTE'92). North-Holland: Perros-Guirec, 1993:199-213.

[3] Zou L, Lv J, Wang S, *et al*. Verifying Chinese control system under a combined scenario by theorem proving [C]// International Conference on Verified Software: Theories, Tools, Experiments (VSTTE). Berlin: Springer-Verlag, 2013: 262-280.

[4] Horste M, Hungar A, Schnieder E. Modelling functionality of train control systems using petri nets[R]. Madrid: FM-RAIL-BOK Workshop, 2013.

[5] Giannakopoulou D. Model checking for concurrent software architectures[D]. London: University of London, 1999.

[6] McMillan K L. Microarchitecture verification by compositional model checking[C/CD]//Proceedings of the 13th International Conference Computer-aided Verification. Livingston: Springer-Verlag, 2011.

[7] McMillan K L. Parameterized verification of the FLASH cache coherence protocol by compositional model checking[C/CD]// Correct Hardware Design and Verification Methods. Lecture Notes in Computer Science. Livingston: Springer-Verlag, 2001.

[8] JIN Zhi, CHEN Xiaohong, Zowghi Didar. Performing projection in problem frames using scenarios[C]//16th Asia-Pacific Software Engineering Conference. Piscataway: IEEE, 2009: 252-256.

[9] XU Jing, CHEN Xiaohong, ZHOU Tingliang, et al. Decomposing automatic train control verification system with projection [C]//22th Asia-Pacific Software Engineering Conference. Los Alamitos: IEEE Computer Society, 2015: 301-308.

[10] 陈小红,尹斌,金芝. 基于问题框架方法的需求建模:一个本体制导的方法[J]. 软件学报,2011, 22(2): 177.
CHEN Xiaohong, YIN Bin, JIN Zhi. Ontology-guided requirements modeling based on problem frames approach [J]. Journal of Software, 2011, 22(2): 177.

[11] Sanaz Yeganefard, Michael Butler. Problem decomposition and sub-model reconciliation of control systems in Event-B[C]// IEEE 14th International Conference on Information Reuse & Integration (IRI). Piscataway: IEEE, 2013: 528-535.

[12] Alebrahim Azadeh, Faβbender Stephan. Problem-based requirements interaction analysis [C]//20th International Working Conference. Cham: Springer International Publishing, 2014: 200-215.

[13] Backus J W. The syntax and semantics of the proposed international algebraic language of the Zurich ACM-GAMM Conference[C]//Proceedings of the International Conference on Information Processing. Paris: UNESCO, 1959: 125-132.

[14] IEEE. IEEE Std 1474.1 Standard for communications-based train control (CBTC) performance and functional requirements [S]. [S.l.]: IEEE, 2004.