中国科学院软件研究所

杰出青年人才发展专项计划年度进展报告书

（2010 年度）

姓名	乔颖	课题名称	面向复杂反应式系统的实时推理技术研究
资助金额	160 万	课题起止时间	2009 年 9 月—2013 年 9 月
资助类别	应用基础研究	所在部门	人机交互与智能信息处理实验室

研究工作主要进展和阶段性成果（含论文、研究生培养等）

1. 主要进展与阶段性成果

（1） 实时推理算法

本年度对面向实时反应式系统的推理算法进行了研究。传统的实时推理算法都是基于产生式的，不适用于实时反应式系统。而国内外学术界还没有对基于 ECA 规则的推理算法展开专门研究。为此，本课题提出了一种基于 ECA 规则的实时推理算法—RTEIA。

在实时反应式系统中，规则中的动作通常是具有时间约束的，也就是说，这些动作需要在其截止期前完成，这也给推理提出了实时约束。RTEIA 算法可根据给定的 ECA 规则推理出响应到达系统的外部事件所需采取的动作，并使这些动作的截止期尽可能地被满足。在该算法中，给定的一系列 ECA 规则被表示为规则图。ECA 规则图是一个有向图，其结点表示 ECA 规则中的事件、条件与动作。若两个结点间需要传递信息，则用一条有向边进行连接。规则图中入度为 0 的结点被称为入口结点，表示系统中的原子事件；出度为零的结点被称为出口结点，表示规则中的动作。推理过程实际上是一个寻找从入口结点到出口结点路径的过程。当一个出口结点被找到，该出口结点所对应的动作将被作为一个推理结果输出。

我们采用基于规则图的启发式搜索，寻找入口结点到出口结点的一条“预期路径”使得系统中的动作能够及时被推理机输出，从而使这些动作地截止期能够得到满足。为此，我们为每一个事件/条件结点定义了目标函数来指导这个搜索过程。假设 v 是一个事件结点。v 的目标函数 $H(v)$的定义如下：

$$H(v) = MIN_{1\le s\le L}\left(\sum_{v_j\in PE_s} WT(v_j) + W * LST(A_s)\right) \quad (1)$$

在这里，PT 是一个从结点 v 到所有出口结点的路径集，$PT=\{pt_1, pt_2, \ldots, pt_L\}$，其中，$L$ 是路径数。PE_s 是 PT 中第 s 条路径上的事件结点集。$LC(v_j)$是 v 的左子结点，$RC(v_j)$是 v 的右子结点。$WT(v_j)$表示 v_j 结点所表示事件的预期等待时间，即反映了 v_j 结点所表示事件需要

多长时间会出现。A_s 是 pt_s (1≤s≤L)上的出口结点。$LST(A_s)$是 A_s 的最晚开始时间。 $LST(A_s) = D(A_s) – E(A_s)$, 其中，$D(A_s)$ 是 A_s 的截止期， $E(A_s)$是 A_s 的运行时间。W 是权重。对于一个事件结点来说，目标函数值越小说明若该结点被激活加入预期路径，则及时找到出口结点，并使其所代表的动作的截止期被系统满足的可能性越大。

搜索从一个特定的入口结点开始。在搜索当中，通过计算结点的目标函数值来选择扩充预期路径所需的结点。当搜索找到了一个动作结点时，便得到了一个推理结果。同时，当一个动作被搜索找到后，还要对其它动作结点进行可行性检查，即检查当该动作结点被作为推理结果输出后，其它动作结点的截止期是否还有可能被满足。根据检查结果，对规则图进行剪枝，将截止期已肯定会被错过的出口结点从搜索目标中删除。RTEIA 算法的描述如下图所示。

1. At the beginning, the expected path, denoted as P, is null.
2. For i = 1 to K
(2.1) Activate the corresponding entrance node, denoted as Ext, for E_i.
(2.2) CurrentNode = E_i;
(2.3) Send Tokens to all parent nodes of CurrentNode;
(2.4) Compute heuristic functions H for all parent nodes of CurrentNode;
(2.5) For each parent node, get the token and save it for the child node corresponding to CurrentNode; set the flag of child node corresponding to CurrentNode to 1;
(2.6) Select one of parent node with smallest value of H; (This node is denoted as SP)
(2.7) Activate SP; check if the event represented by SP occurs according the methods listed in Table 1; If it occurs, Occ=true; otherwise, Occ = false;
(2.8) If (Occ==true)
(2.8.1) <CurrenNode, SP> is added into P;
(2.8.2) CurrentNode = SP;
(2.8.3) Create the token and send the token to all parent nodes of CurrentNode;
Else
(2.8.4) Backtrack to the previous level;
(2.8.5) Select one of parent node with the next smallest value of H at this level;
(2.9) Repeat (2.4) – (2.8), until one of following termination conditions is satisfied:
(a) find a exit node, i.e., an action node;
(b) find a condition node
(c) No backtrack is possible
(d) When the predefined reasoning deadline is reached
（2.10）If an exit node is found
(2.10.1) For all action nodes in the rule graph, perform feasibility check;
(2.10.2) Perform pruning for the rule graph;
(2.10.3) Output the action represented by this exit node

图 RTEIA 算法描述

我们在随机生成 ECA 规则集的基础上，对 RTEIA 算法进行了模拟验证，并将 RTEIA 的推理成功率与传统深度优先搜索算法的推理成功率进行了比较。结果显示，RTEIA 的推理成功率随着 ECA 规则集规模的增大（即规则条数的增大）而减少，但对规则条数的变化反应并不敏感；同时，在规则规模变化的过程中，RTEIA 的推理成功率始终要大于深度优先搜索算法的推理成功率。另一方面，随着规则中动作运行时间的增大，RTEIA 的推理成功率将减少，且在动作的运行时间足够小时，RTEIA 的推理成功率对动作运行时间的变化并不敏感，而当动作运行时间超过了某一长度，则 RTEIA 的推理成功率对动作运行时间的变化变得非常敏感。这表明 RTEIA 适用于动作运行时间较短的系统，这对于大多数的实时反应式系统来说，都是有意义的。同时，在动作运行时间变化的过程中，RTEIA 的推理成功率始终要大于深度优先搜索算法的推理成功率。

（2） 推理的最坏运行时间分析

本年度对推理的最坏运行时间分析进行了研究，提出了针对 RTEIA 算法的实时推理最坏运行时间分析方法。经典的估算最坏运行时间的方法分为两种，静态分析及动态测量。动态测量的结果不能保证安全性。而静态分析方法包括三类，基于结构、基于路径及基于整数线性规划的方法。但是，所有这些方法都只适合于分析具有简单控制流的程序，而用来分析复杂的智能推理程序准确度很低。为此，本课题在传统的线性规划的估算方法基础上，提出了基于分层结构的 ECA 规则推理算法的最坏运行时间估算方法，针对 RTEIA 算法，给出了 ECA 规则推理系统在包含N个事件的事件风暴模型下的最坏运行时间的估算方法。下图给出了最坏运行时间估算的分层框架。

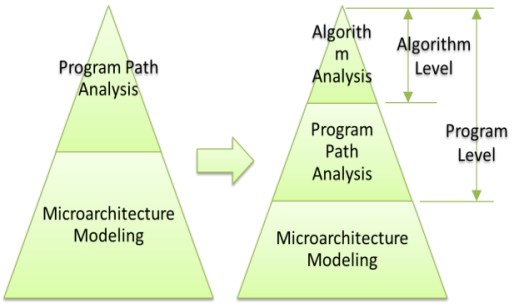

图 最坏运行时间估算框架，(a)传统框架，(b)分层框架

如上图(a)，传统的基于整数线性规划的估算方法分为程序路径分析及硬件模型分析两步。由于 ECA 规则推理程序的控制流非常复杂，在程序路径分析中难以发现线性约束影响准确度。如上图(b)，我们增加了算法分析步骤，分析程序的算法逻辑结构归纳线性约束。这一分析分为算法分析层及程序分析层。在算法分析层，估算算法逻辑模块访问次数，线性约束源于算法分析步骤；在程序路径分析层，估算程序语句执行条数，线性约束源于算法分析及程序路径分析。目前本课题尚未涉及硬件模型分析。

在算法分析层，分析的目标是估算最坏情况下的规则图中所有节点处理令牌的个数。假设节点v_i的输出的令牌个数是x_i，则所有规则图节点处理的总次数是：$\sum_{v_j \in V-V_{Ext}} \left(b_j \sum_{i \in I_j} x_i \right) + \sum_{v_i \in V_{Ext}} b_i x_i$，其中前项是非叶节点的处理次数，后项是叶节点的处理次数，V_{Ext}是叶节点集。在事件风暴模型下，输入的事件个数是N。所以，对于叶节点，所有的输入满足输入约束：$\sum_{v_i \in V_{Ext}} x_i = N$。

在规则中，节点决定如何处理令牌，不同类型的节点对令牌的处理方式不同。我们采用自动机来表示各类复合事件类型所对应的节点处理令牌的过程，并根据自动机中的“最小路径”可以归纳出算法结构约束：$x_{output} \leq Min(x_A, x_B, x_C)$。

变量x_i与$y_i{}^j$之间的关系，称之为连接约束。假设$y_i{}^{call}$是v_i的控制流图的入口访问次数，$y_i{}^{trigger}$是v_i输出令牌次数。那么，连接约束为：$y_i{}^{call}=\sum_{j\in OP_i{}^{Begin}} y_i{}^j=\sum_{j\in IP_i{}^{End}} y_i{}^j$，$\begin{cases} y_i{}^{call}=x_i, v_i\in V_{Ext} \\ y_i{}^{call}=\sum_{j\in I_i} x_j, v_i\in V-V_{Ext} \end{cases}$，$y_i{}^{trigger}=y_i{}^{jt}, m^{jt}$是创造新信元的模块，以及$y_i{}^{trigger}=x_i$。

程序的顺序、分支、循环结构，分别可以有如下约束：$y_i{}^0=y_i{}^1$，$y_i{}^0=y_i{}^1+y_i{}^2$，$y_i{}^1=y_i{}^0+y_i{}^2=y_i{}^2+y_i{}^3$。在循环结构中，应该有功能性约束，

$$\begin{cases} y_i{}^j\geq (lower\ loop\ bound+1)*\sum_{k\in IP_i{}^{j,outside}} y_i{}^k \\ y_i{}^j\leq (upper\ loop\ bound+1)*\sum_{k\in IP_i{}^{j,outside}} y_i{}^k \end{cases}$$

在此基础上，我们对以上的整数线性规划问题进行了优化，降低了求解时间，并将其进一步转化为线形规划问题，从而可以在多项式时间求得有效解。

为了验证所提出的最坏运行时间估算方法的有效性，本课题对其进行了模拟实验。在这里，我们选用不准确度$p=T_{Analysis}/T_{Real}$作为实验的指标。其中$T_{Analysis}$是估算值，T_{Real}是真实值。实验结果表明，当规则图中节点数增加时，叶节点数减少时，或者最大子节点数增加时，所提出的估算方法的不准确度基本保持稳定，并比传统的估算方法在准确度上提高了一个以上数量级。

2．国内外学术交流、研究生培养及论文发表情况

本年度课题与国外相关学者，如，与 Washington University in st. Louis 的吕晨阳教授就实时推理算法进行了交流与讨论。同时，培养硕士生两名，协助培养博士生两名。本年度发表论文两篇。

下一年度工作计划，包括国内外合作与交流计划

下一年度的研究工作计划包括：

1. 继续对实时推理算法进行完善，进一步提高推理成功率，通过实际应用对其进行进一步的验证；并开展分布式实时推理算法的研究。
2. 继续深入研究推理最坏运行时间分析方法，研究推理实时性分析通用框架，并进行原型实现。
3. 研究模糊时序事件，提出模糊规则模型，并设计图形化的模糊主动规则描述语言

下一年度的国内外合作与交流计划包括：

1. 继续与 Washington University in st. Louis 的吕晨阳教授就推理最坏运行时间分析方法以及实时模糊推理方法进行合作交流。
2. 参加 IEEE symposium on Real-time systems 会议。