一种无死锁的时间管理算法*

刘步权+, 王怀民, 姚益平

(国防科学技术大学 计算机学院,湖南 长沙 410073)

A Non-Deadlock Time Management Algorithm

LIU Bu-Quan+, WANG Huai-Min, YAO Yi-Ping

(School of Computer Science, National University of Defence Technology, Changsha 410073, China)

+ Corresponding author: Phn: 86-731-4573697, E-mail: bqliu@nudt.edu.cn

http://www.nudt.edu.cn

Received 2002-07-19; Accepted 2003-03-20

Liu BQ, Wang HM, Yao YP. A non-deadlock time management algorithm. *Journal of Software*, 2003,14(9): 1515~1522.

http://www.jos.org.cn/1000-9825/14/1515.htm

Abstract: HLA (high level architecture) is the standard for modeling and simulation put forward by the American Department of Defense. Time management is an important component of HLA while GALT (greatest available logical time) algorithm is RTI (runtime infrastructure)'s critical technology in implementing time management. An improper GALT algorithm can easily result in a deadlock so that the whole federation can not advance any more. On the basis of the GALT algorithm introduced by Frederick Kuhl, the principles of deadlock are discussed and some important results are revealed in this paper. If deadlock occurs in a simulation, all federates must have the same GALT and the same output time respectively, and GALT is also equal to output time, and a federate whose lookahead is greater than zero must suspend because of a NMR or NMRA request, other than a TAR, TARA or FQR request. Finally, a GALT algorithm without deadlock is also brought forward in this paper that is called Stature-Measuring, and this algorithm can provide reliable technology support to develop time management services of RTI.

Key words: high level architecture; time management; GALT; deadlock; stature-measuring

摘 要: 高层体系结构 HLA(high level architecture)是美国国防部提出的建模和仿真的标准,时间管理服务是其重要的组成部分,而 GALT(greatest available logical time)的计算是 RTI(runtime infrastructure)时间管理服务实现的核心技术.GALT 算法容易导致死锁,继而导致整个仿真无法推进.在 Frederick Kuhl 算法的基础上探讨了死锁产生时系统所特有的一些规律,得出了一些重要结论:如果系统处于死锁状态,则所有盟员的 GALT 和输出时间一定分别相等,并且 GALT 一定等于输出时间;所有 Lookahead 大于 0 的盟员一定处于 NMR/NMRA 推进状态

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2001AA115127 (国家高技术研究发展计划(863)); the National Grand Fundamental Research 973 Program of China under Grant No.G1999032703 (国家重点基础研究发展规划(973))

第一作者简介: 刘步权(1969－),男,江苏姜堰人,博士生,讲师,主要研究领域为分布式仿真,分布对象技术,操作系统.

而不会处于其他推进状态(TAR/TARA/FQR).最后提出了基于“身高测量法”的无死锁的 GALT 算法,“身高测量法”为 RTI 中时间管理模块的实现提供了可靠保证.

关键词: 高层体系结构;时间管理;GALT;死锁;身高测量法

中图法分类号: TP338 **文献标识码**: A

高层体系结构 HLA[1](high level architecture)是美国国防部提出的关于分布式仿真的一个最新标准,其主要目的是用于实现各类仿真的互操作和可重用.在典型的分布式仿真应用中,由于盟员(federate)(“八六三”会议对 HLA 规范中的若干术语进行了统一命名.如:federation 译作“联盟”,federate 译作“盟员”,RTI 译作“运行支撑环境”)方的计算量、处理器的计算速度以及网络延迟等因素的影响使得盟员之间的消息传递存在不确定性,而消息的先后次序对整个仿真而言是非常重要的.时间管理服务是 HLA 的重要组成部分,其主要目的就是保证盟员能够按照正确的顺序接收消息,从而保证仿真的正确执行.

HLA 支持两类时间推进机制,即保守盟员的时间推进机制和乐观盟员的时间推进机制.保守盟员可以采用 TAR(time advance request),TARA(time advance request available),NMR(next message request)(IEEE 1516 中的 NMR/NMRA 对应于 HLA 1.3 中的 NER/NERA)和 NMRA(next message request available)等 4 种 HLA 标准服务进行时间推进;乐观盟员可采用 FQR(flush queue request)服务进行时间推进.时间管理服务是基于 HLA 标准实现 RTI(runtime infrastructure)软件的重点和难点,而 GALT(greatest available logical time)(IEEE 1516 中的 GALT 相当于 HLA 1.3 中的 LBTS.IEEE 1516 去掉了 LBTS,增加了 GALT 和 LITS)的计算问题则是 RTI 中时间管理模块实现的关键.GALT 是 IEEE 1516 标准提出的新概念,在此之前,GALT 被称为 LBTS(lower bound on the time stamp);为统一起见,本文将遵循 IEEE 1516 标准,采用 GALT 术语来论述问题.早在 HLA 出现之前,GALT 就已经成为并行离散事件仿真讨论的重要话题[2,3];但是与并行离散事件仿真相比,HLA 有着不同的体系结构,因此 GALT 的计算问题也必然有所区别[4].GALT 计算应该综合考虑各种时间推进方式,然而很多论文考虑的问题都不够全面[4,5].作为 RTI 的核心技术之一,RTI 产品开发者也不可能公布完整的 GALT 关键算法.但是,在 Frederick Kuhl 等人的著作中[6],却描述了一个简单、实用的 GALT(书中使用的是 LBTS)算法.作为 HLA 的参与者和制订者,他们的算法应当具有可信性和权威性.该算法在通常情况下能够比较好地工作,但遗憾的是,在特殊情况下,该算法会造成死锁.

GALT 算法与 RTI 实现时所采用的底层通信方式密切相关,为简单起见,我们假设两个结点之间的消息传递具有管道特性,即从一个结点发送给另外一个结点的消息遵循先发送先接收的原则.在此基础上,我们首先介绍了 Frederick 算法及其死锁问题,重点分析了死锁产生时系统所特有的一些规律,从而揭示了死锁产生的根本原因.最后,对 Frederick 算法进行改进,提出了基于“身高测量法”的无死锁的 GALT 计算方法.这就表明,采用恰当的 GALT 算法,基于 HLA 的时间管理机制的联盟的推进是不会出现死锁的;当出现死锁时,如果采用不恰当的解锁算法则可能导致与 HLA 规范相冲突.

1 基本术语和算法

在介绍 GALT 算法时,需要引用一些术语.这些术语有些在 IEEE 1516 标准中已有说明,有些在 Frederick 算法中被采用了,但在仿真技术人员常见的 HLA 1.3[7]规范中几乎都没有出现过.

1.1 基本术语

定义 1. GALT 是指盟员能够推进的最大逻辑时间.

在保守时间推进机制下,HLA 规范要求一个盟员在仿真的任何时刻都不能接收到一个过时(即比盟员的当前逻辑时间小)的消息.因而在盟员推进时,RTI 必须确保盟员的时间推进不能超过 GALT 这个时间上界,否则在未来的逻辑时间里有可能接收到一个过时的消息.

定义 2. LETS(least existent time stamp)是指盟员消息队列中已有消息的最小时标.

LETS 不是 IEEE 1516 规范中的定义,而是为了叙述方便在本文中引进的.

定义 3. LITS(least incoming time stamp)是指盟员有可能接收到的最小消息的时标.

通过计算 GALT 和 LETS,RTI 能够确定盟员的 LITS.LITS 可取 GALT 和 LETS 的最小值.LITS 和 LETS 的区别在于,LETS 是那些已经接收到的放在盟员的 TSO 队列中的所有消息的最小时标;而 LITS 还可能包括那些仍在网络中传输甚至还没有发出的小时标的消息.

定义 4. 输出时间(output time)是指 RTI 计算出的盟员能够发送的消息时标的最大下界.

盟员只能发送不小于输出时间的消息,否则其他盟员有可能接收到一个过时的消息.据此,RTI 在计算盟员的 GALT 时可由其他盟员的输出时间来得到.

定义 5. 挂起(pending)是指盟员在请求时间推进时因为没有得到 RTI 的允许而处于暂停状态.

盟员可使用 5 种时间管理服务进行时间推进:TAR/TARA 为时间步进方式,NMR/NMRA 为事件驱动方式,FQR 为乐观方式.由于 FQR 请求总能够得到满足,所以 FQR 请求不会挂起.盟员只会因为其他 4 种时间管理服务而被挂起.这样,本文把盟员的状态定义为 5 种:Grant、TAR 挂起、TARA 挂起、NMR 挂起和 NMRA 挂起.“Grant”是指盟员没有请求时间推进时的状态,当 RTI 同意挂起盟员的时间推进请求时,盟员进入 Grant 状态.

定义 6. 死锁(deadlock)是指系统中有相互约束关系的盟员均处于挂起状态.

不同系统对死锁的定义会有所不同,本文谈到的死锁是由于计算 GALT 而引起的.一个盟员的时间推进状态将直接影响到其他盟员的 GALT 计算,一个处于挂起状态的盟员需要其他盟员 GALT 的增加才有可能被解除挂起状态而进入 Grant 状态.如果有相互约束关系的盟员都因为期望其他盟员能够增加 GALT 而解除挂起状态,那么这些盟员就会处于互相等待状态而构成死锁.当处于死锁状态时,整个仿真将无法推进.这里所说的“约束关系”是指盟员之间的“校准/约束(regulating/constrained)”关系.文中的所有定理和算法都以参加仿真的所有盟员之间既相互校准又相互约束为前提条件,在第 4 节将会对盟员的约束关系进行全面的探讨.

1.2 Frederick算法

Frederick 算法通过输出时间来计算 GALT,其算法非常简单.

算法 1(Frederick 算法). 对于任意盟员 i,其 GALT 计算公式为

$$GALT(i)=\min\{\mathrm{OUTPUT}(j)\};\ i\neq j.$$

Frederick 算法将盟员的 GALT 取值为其他盟员的输出时间的最小值.算法的关键在于计算盟员的输出时间,输出时间取决于盟员的当前状态、逻辑时间、Lookahead 和 LETS 等多个因素.

算法 2(输出时间). 盟员的输出时间可按照下列公式进行计算:

(1) 如果盟员处于 TAR/TARA 挂起状态,则输出时间的计算公式为

$$\mathrm{OUTPUT}(i)=T(i)+L(i),$$

$T(i)$为盟员 i 请求推进的时间,$L(i)$为盟员 i 当前的 Lookahead 值.

(2) 如果盟员处于 NMR/NMRA 挂起状态,则输出时间的计算公式为

$$\mathrm{OUTPUT}(i)=\min\{T(i)+L(i),LETS(i),GALT(i)\},$$

$T(i)$为盟员 i 请求推进的时间,$L(i)$为盟员 i 当前的 Lookahead 值,$LETS(i)$为盟员 i 的消息队列中的最小消息的时标,$GALT(i)$为盟员 i 的 GALT 值.

(3) 如果盟员处于 Grant 状态,则输出时间的计算公式为

$$\mathrm{OUTPUT}(i)=T(i)+L(i),$$

$T(i)$为盟员 i 的逻辑时间,$L(i)$为盟员 i 当前的 Lookahead 值.

在讨论 GALT 计算的诸多文献中,都仅限于 Grant/TAR/TARA 状态[6,7],所以这里需要重点讨论盟员处于 NMR/NMRA 状态时输出时间的计算公式.在 Frederick 算法中,将盟员的“逻辑时间+Lookahead”称为“最小允许的时标”(minimum allowed time stamp),这里暂记作 MATS.在 Grant/TAR/TARA 状态,MATS 与输出时间相等.当处于 NMR/NMRA 状态时,除了考虑逻辑时间和 Lookahead 以外,Frederick 算法还考虑了 LETS 和 GALT,因而 MATS 与输出时间是不相等的.

如图 1 所示,盟员的 Lookahead 为 0.5,并采用 NMR 方式推进.在墙上时间(wall clock time)$w1$ 时刻,盟员的逻辑时间为 t,处于 Grant 状态,$MATS(w1)$=逻辑时间+Lookahead=t+0.5;在 $w2$ 时刻,盟员使用 NMR 请求推进到

t+1,$MATS$(w2)=(t+1)+0.5=t+1.5;在 w3 时刻,RTI 同意盟员推进到 t+1,于是盟员处于 Grant 状态,$MATS$(w3)=t+1.5;在 w4 时刻,盟员使用 NMR 请求推进到 t+2,$MATS$(w4)=(t+2)+0.5=t+2.5;但此时其他盟员给该推进盟员发送了一个时标为 t+1.5 的时标消息,则 RTI 只允许该盟员推进到 t+1.5,而不是 t+2.假设在 w5 时刻,RTI 同意盟员推进到 t+1.5,于是盟员处于 Grant 状态,$MATS$(w5)=(t+1.5)+0.5=t+2;然而 $MATS$(w5)<$MATS$(w4),如果以 MATS 来计算盟员的 GALT 值,则盟员有可能接收到一个过时的消息.图 1 说明,在计算 GALT 时,不能仅考虑逻辑时间和 Lookahead,因此 Frederick 算法中使用输出时间这个概念来计算 GALT 是比较合理的.

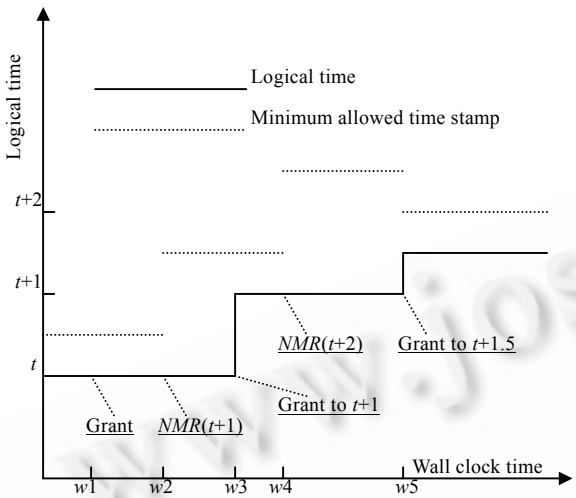

Fig.1 Event-Driven simulation
图 1 事件驱动的仿真

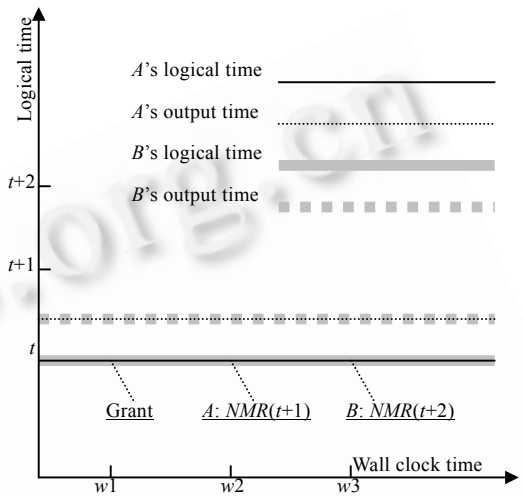

Fig.2 Deadlock of Frederick algorithm
图 2 Frederick 算法中的死锁

1.3 Frederick算法的死锁问题

Frederick 算法较好地解决了盟员使用 NMR/NMRA 请求时间推进时的 GALT 计算问题,并且在绝大多数情况下非常实用和有效,但是该算法却不能避免死锁.

假设在一次仿真中只有 A 和 B 两个盟员参加,它们的 Lookahead 均为 0.5,图 2 描述了 A 和 B 的逻辑时间均为 t 时发生的序列事件.由于只有两个盟员,所以一个盟员的输出时间就是另外一个盟员的 GALT.在墙上时间 w1 时刻,A 和 B 均处于 Grant 状态,它们的输出时间均为 t+0.5,GALT 也同样为 t+0.5;在 w2 处,盟员 A 使用 NMR 请求推进到 t+1,但因为此时它的 GALT 仍为 t+0.5,所以盟员 A 的请求被挂起;在 w3 处,盟员 B 使用 NMR 请求推进到 t+2,这时在计算 B 的 GALT 时就会出现两种情况.第 1 种情况是,由于 B 的 GALT 等于 A 的输出时间,因而可取 w3 时刻之前的 A 的输出时间 t+0.5 作为 B 的 GALT,由于 t+0.5 小于 B 的请求时间 t+2,所以 B 也被挂起,出现了死锁;第 2 种情况是,在 w3 处,重新计算 A 的输出时间,但根据 Frederick 算法,A 的输出时间依赖于 B 的 GALT,而 B 的 GALT 正是要求的结果,这样,在计算 GALT 或输出时间时算法本身也会出现死锁,即无法计算 GALT 或输出时间.

然而,如果一个盟员使用 TAR 请求,另外一个使用 TAR 或 NMR 请求,则不会出现死锁.在如图 2 所示的死锁现象中,两个盟员的 GALT 和输出时间均相等,并且每个盟员的 GALT 等于自己的输出时间.这些现象到底是偶然的,还是必然的呢? 而这正是本文在下一节所要讨论的问题.

2 死锁的规律性

一个考虑不全的 GALT 算法容易产生死锁,而死锁却是有规律可循的.要解决死锁,就必须摸索到死锁的规律性,否则一个能够解决死锁的算法也只是解决部分而不是全部死锁现象.本文正是在这方面作了积极的探索,并在此基础上提出了一个无死锁的 GALT 算法.

本文借用 Frederick 中的输出时间概念,在算法 1 和算法 2 的基础上来研究死锁问题,得到了一些重要结论.

引理 1. 如果所有盟员均因为请求时间推进而处于死锁状态,那么对于任意盟员 i,GALT 满足公式:

(1) $GALT(i)\le T(i)$;

(2) $GALT(i)\le T(i)+L(i)$;

(3) 如果盟员处于 NMR/NMRA 状态,则 $GALT(i)\le LETS(i)$;

(4) 如果盟员处于 NMR/NMRA 状态,则 $GALT(i)\le\min\{T(i)+L(i),LETS(i)\}$.

证明:用反证法证明.

(1) 假设盟员 i 的 $GALT(i)>T(i)$,则不论盟员因为何种时间推进服务而挂起,RTI 都能够满足盟员的时间推进请求,这样盟员 i 就不会挂起,与死锁矛盾.

(2) 因为 $L(i)$为非负值,由(1)即可得到证明.

(3) 假设 $GALT(i)>LETS(i)$,则盟员的请求能够满足,RTI 同意盟员推进的时间就是 $LETS(i)$,与死锁矛盾.

(4) 由(2)和(3)可得. □

引理 2. 如果所有盟员均因为请求时间推进而处于死锁状态,那么对于任意盟员 i,输出时间满足公式:

(1) 如果盟员处于 TAR/TARA 状态,则 $OUTPUT(i)\ge GALT(i)$;

(2) 如果盟员处于 NMR/NMRA 状态,则 $OUTPUT(i)=GALT(i)$;

(3) 不论盟员处于何种挂起状态,都有 $OUTPUT(i)\ge GALT(i)$;

(4) 不论盟员处于何种挂起状态,都有 $OUTPUT(i)\le T(i)+L(i)$.

由引理 1 和算法 2 易证引理 2.

定理 1. 如果所有盟员均因为请求时间推进而处于死锁状态,那么所有盟员的 GALT 相等.

证明:设系统中的盟员个数为 $n(n\ge2)$,n 个盟员的 GALT 构成一个集合 K:

$$K=\{GALT(1),GALT(2),GALT(3),\ldots,GALT(n)\}.$$

假设并非 K中的所有值均相等,不失一般性,可设 $GALT(1)$为 K中的最大值,则存在 a,使得 $GALT(1)>GALT(a)$.另外,n 个盟员的输出时间构成一个集合 $\Phi=\{O(1),O(2),O(3),\ldots,O(n)\}$.在有穷集 Φ中,一定存在最小值,分两点说明:(1) 假设 $O(1)$为最小值,则 $GALT(a)=\min(\Phi-\{O(a)\})=O(1)$.这样,$O(1)<GALT(1)$,与引理 2 中的(3)矛盾,所以,$O(1)$不可能是 Φ中的最小值.(2) 不失一般性,设 $O(2)$为 Φ中最小值,且 $O(1)>O(2)$.由算法 1 可知:$GALT(1)=\min(\Phi-\{O(1)\}=O(2)$;$GALT(2)=\min(\Phi-\{O(2)\})\ge O(2)=GALT(1)$.若 $GALT(2)>GALT(1)$,则与 $GALT(1)$为 K中的最大值矛盾,所以 $GALT(2)=GALT(1)$.对于其他任意盟员 $i(i\ne1,2)$,$GALT(i)=\min(\Phi-\{O(i)\})=O(2)=GALT(1)$,所以 n 个盟员的 GALT 值均相等. □

定理 2. 如果所有盟员均因为请求时间推进而处于死锁状态,那么所有盟员的输出时间相等.

证明:设系统中的盟员个数为 $n(n\ge2)$,用数学归纳法证明.(1) 当 $n=2$ 时,由算法 1 可知,$O(1)=GALT(2)$,$O(2)=GALT(1)$.由定理 1 可知,$GALT(1)=GALT(2)$,所以,$O(1)=O(2)$.(2) 假设当 $n=k$ 时结论成立,下面证明 $n=k+1$ 时结论也成立.$k+1$ 个盟员的输出时间构成一个集合 $\Phi=\{O(1),O(2),O(3),\ldots,O(k+1)\}$.对集合 Φ中的值按照从小到大的顺序进行排列,不失一般性,可设 $O(1)\le O(2)\le O(3)\le\ldots\le O(k+1)$,则

$$GALT(1)=\min(\Phi-\{O(1)\})=\min(\{O(2),O(3),\ldots,O(k+1)\})=O(2),$$

$$GALT(2)=\min(\Phi-\{O(2)\})=\min(\{O(1),O(3),\ldots,O(k+1)\})=O(1).$$

由定理 1 可知,$GALT(1)=GALT(2)$,所以,$O(1)=O(2)$.那么从 $k+1$ 个盟员中去掉盟员 1,不会影响其余 k 个盟员的 GALT 计算,所有这些盟员并不会因为盟员 1 的退出而不再处于挂起状态,因此定理的条件满足,由归纳法可知:$O(2)=O(3)=\ldots=O(k+1)$.所以 $n=k+1$ 时结论也成立. □

定理 3. 如果所有盟员均因为请求时间推进而处于死锁状态,那么所有盟员的输出时间等于自己的 GALT,也等于任意盟员的 GALT.

证明:设所有盟员的输出时间构成集合 Φ,则对于任意盟员 i,$GALT(i)=\min\{\Phi-O(i)\}$.所以一定存在盟员 j,使得 $GALT(i)=O(j)$,$i\ne j$.由定理 2 可知,$O(i)=O(j)$,所以 $GALT(i)=O(i)$.根据定理 2,$GALT(i)=O(k)$,k 为任意盟员. □

以上的讨论和定理适用于 Lookahead≥0 的情形,下面的讨论包括身高测量法仅限于 Lookahead>0 的情形.因为在 Lookahead=0 时多个盟员构成的系统也可能造成死锁,但这种死锁是与特定的仿真应用相关的,而与

GALT 算法无关.

定理 4. 如果所有盟员均因为请求时间推进而处于死锁状态,那么不存在盟员因为 TAR 或 TARA 请求而被挂起,即所有盟员均因为 NMR 或 NMRA 请求而被挂起.

证明:用反证法证明.假设盟员 i 处于 TAR 或 TARA 挂起状态,其输出时间为 $O(i)$,则根据算法 2 有 $O(i)=T(i)+L(i)$.另外,由定理 3 可知:$GALT(i)=O(i)$,所以由假设 $L(i)>0$ 可知,$GALT(i)=T(i)+L(i)>T(i)$,则盟员 i 的推进请求能够得到满足,与盟员 i 处于挂起状态矛盾.由于盟员在调用 FQR 请求时总能够得到满足,所以盟员不会因为 FQR 请求而被挂起,这样盟员只能因为 NMR 或 NMRA 请求而被挂起. □

定理 5. 如果系统中至少有一个盟员因为 TAR 或 TARA 请求而处于挂起状态,那么系统中至少存在一个处于 Grant 状态的盟员.

证明:用反证法证明.假设没有盟员处于 Grant 状态,则所有盟员均处于挂起状态.由定理 4 可知,系统中不存在处于 TAR 或 TARA 挂起状态的盟员,与定理的前提条件矛盾. □

由以上的定理可知,死锁产生的原因不是 HLA 规范本身所造成的,其根本原因在于不恰当的 GALT 算法导致所有挂起盟员的输出时间相等而无法推进任何一个挂起盟员.然而,如果所有盟员的 Lookahead 大于 0,则只有当所有挂起的盟员都是因为 NMR/NMRA 请求时才会出现死锁现象;由于 HLA 主要面向大规模的系统仿真,并且参加仿真的盟员可以使用多种时间推进机制,因此死锁问题一般是不会出现的.但是对于一个合格的 RTI 产品而言,必须综合考虑各种情况,尤其是死锁问题.

3 “身高测量法”

前面讨论的死锁问题本质上是由于 GALT 的算法不完善而造成的,一个好的算法是不会造成死锁的.下面我们给出一个基于“身高测量法”的无死锁的算法.

定义 7. 身高(stature)可定义为

$$H(i)=\begin{cases}T(i)+L(i), & \text{if federate in Grant/TAR/TARA state}\\ \min\{T(i)+L(i),LETS(i)\}, & \text{if federate in NMR/NMRA state}\end{cases}.$$

$H(i)$表示盟员 i 的身高;

$T(i)$表示盟员 i 的逻辑时间或请求推进的逻辑时间;

$L(i)$表示盟员 i 的 Lookahead;

$LETS(i)$表示盟员 i 消息队列中的最小消息的时标.

这里再次对 FQR 请求进行必要的说明,由于 FQR 请求总能够得到满足,因此在具体的 RTI 实现中可以把该请求看做是原子动作,盟员在请求前后均为 Grant 状态.但是在实现 RTI 时需要注意的是,FQR 请求可能导致其他盟员的 GALT 的增加,从而引起其他盟员从挂起状态进入 Grant 状态.

算法 3(身高测量法). 对于任意盟员 i,其 GALT 取决于其他盟员的身高,公式为

$$GALT(i)=\min\{H(j)\};i\neq j.$$

Frederick 算法会导致 GALT 计算的死锁,其根本原因是在计算输出时间时增加了对 GALT 的比较.身高测量法本质上是对 Frederick 算法的改进,最重要的改进之处在于用身高代替了输出时间,在身高中不再增加对 GALT 的比较.在图 2 的 $w3$ 时刻,$H(A)=(t+1)+0.5=t+1.5=GALT(B)$,$H(B)=(t+2)+0.5=t+2.5=GALT(A)$,这样,$A$ 的 GALT 大于 A 的请求时间,所以 A 的推进请求得到满足,死锁被解开.身高测量法的特点可以通过下面的定理和推论来体现.

定理 6. 如果一个盟员请求时间推进时的身高最矮,则该盟员的请求总能够得到满足.

证明:设盟员 A 的身高最矮,即 $H(A)$最小.由身高测量法可知:$GALT(A)=\min\{H(i)\}\geq H(A),i\neq A$.(1) 如果盟员的请求为 TAR/TARA,则 $GALT(A)\geq T(A)+L(A)>T(A)$,请求能够满足.(2) 如果盟员的请求为 NMR/NMRA,则 $GALT(A)\geq\min\{T(A)+L(A),LETS(A)\}$.若 $LETS(A)>T(A)$,则 $GALT(A)>T(A)$,请求能够满足.下面讨论 $LETS(A)\leq T(A)$ 的情形,此时 $H(A)=LETS(A)$.又因为对任意盟员 $i(i\neq A)$而言,RTI 同意其推进的时间一定$\geq H(A)$,因此由消息传递时的管道特性可知:盟员 i 能够发送的消息的时标$\geq H(A)+L(i)>H(A)=LETS(A)$,因此盟员 A 再也不会收到$\leq LETS(A)$

的消息,RTI 同意盟员 A 推进到 $LETS(A)$.(3) 如果盟员的请求为 FQR,则请求总能够满足. □

在一个联盟中,可能有多个身高最矮的盟员,但是只要它们处于时间推进状态,则 RTI 总能够满足其时间推进请求.

定理 7. 身高测量法不会导致死锁.

证明:用反证法证明.假设系统存在死锁,则 n 个盟员的身高构成一个集合 $\Psi=\{h(1),h(2),h(3),\ldots,h(n)\}$.在有穷集 Ψ中,一定存在最小值,即系统中一定存在身高最矮的盟员 A,由定理 6 可知,盟员 A 的请求总能够满足,与死锁矛盾. □

推论. 身高测量法总是首先推进身高最矮的盟员.

盟员可以采用 5 种时间管理服务 TAR/TARA,NMR/NMRA 和 FQR 进行时间推进,在每种服务推进的过程中,都需要计算盟员的 GALT,如果盟员自己的 GALT 得到了提升,则需要重新计算身高最矮的盟员,如果最矮盟员处于挂起状态,则会推动它前进.

4 进一步的讨论

前面讨论了参加仿真的所有盟员之间既相互校准又相互约束的情形,然而并非所有的盟员都必须采用时间管理机制.一个盟员可以采用 4 种方式参加联盟,即校准但非约束的、约束但非校准的、校准且约束的、既非校准的也非约束的.在计算 GALT 时,RTI 应综合考虑各种情况.

(1) 校准但非约束的盟员

RTI在计算一个盟员的GALT时只考虑制约该盟员的其他盟员是否具有校准特性,而不涉及约束特性.盟员在将自己设置为校准盟员时,有自己的逻辑时间和 Lookahead 值.因此,RTI 在计算 GALT 时必须考虑这样的盟员.

(2) 约束但非校准的盟员

由于约束但非校准的盟员只接收时标消息而不产生时标消息,因此对其他盟员的GALT不会产生影响,RTI在计算 GALT 时可不必考虑这样的盟员.

(3) 校准且约束的盟员

这种盟员正是前面所提的盟员,它既能够接收时标消息,也可以产生时标消息.RTI 必须考虑这样的盟员.

(4) 既非校准的也非约束的盟员

这种盟员既不接收时标消息,也不产生时标消息,对其他盟员的 GALT 不会产生影响,RTI 在计算 GALT 时不必考虑这样的盟员.

(5) 集群关系

在计算 GALT 时,除了那些不需要考虑的盟员之外,其他盟员构成既相互校准又相互约束的关系.然而,盟员之间的这种校准/约束关系如果用图来表示(点表示盟员,线表示盟员之间的关系),则这样的图通常是单个的连通图,因为参加仿真的盟员之间应当互相联系;但 RTI 也应该考虑可能分成多个不相关的连通图的情形.此时,前面的定理和推论可以分别作用于每个连通图.然而,身高测量法既可以分别作用于每个连通图,也可以同时作用于所有连通图,只不过有可能使得一个连通图中的盟员推进速度因为等待另外一个连通图中的盟员的推进而变慢.这就是说,身高测量法具有普遍性,不需要去考虑盟员之间的校准/约束关系可能构成多个连通图的情形.

5 结束语

GALT 算法是时间管理服务能否实现的关键技术,也是整个 RTI 产品的核心技术.一个不好的 GALT 算法会导致死锁,从而导致整个仿真无法向前推进.本文对死锁的规律性进行了积极的探索,指出了死锁的根本原因在于 GALT 算法使得所有挂起的盟员的输出时间甚至 GALT 均相等,这样一个盟员无法通过其他盟员的 GALT 的提升而使自己的 GALT 得到提升.最后,本文提出了一个无死锁的 GALT 算法,即身高测量法,较好地解决了 GALT 的死锁问题.

References:

[1] Simulation Interoperability Standards Committee (SISC) of the IEEE Computer Society. IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)—IEEE Std 1516-2000, 1516.1-2000, 1516.2-2000. New York: Institute of Electrical and Electronics Engineers, Inc., 2000.

[2] Fujimoto RM. Lookahead in parallel discrete event simulation. In: Briggs FA, ed. Proceedings of the 1988 International Conference on Parallel Processing. 1988. 34~41.

[3] Riley GF, Fujimoto R, Ammar MH. Network aware time management and event distribution. 2000. http://www.cc.gatech.edu/computing/pads/papers.html.

[4] Ouyang LL, Song X, Qing DZ, Hao JB, Wang J. Research of time management in HLA and simulation algorithms of PDES. Journal of System Simulation, 2000,12(3):237~240 (in Chinese with English abstract).

[5] Carothers CD, Weatherly RM, Fujimoto RM, Wilson AL. Design and implementation of HLA time management in the RTI version F.0. In: Andradottir S, Healy KJ, Withers DH, Nelson BL, eds. Proceedings of the 1997 Winter Simulation Conference. Piscataway: IEEE, 1997. 373~380.

[6] Kuhl F, Weatherly R, Dahmann J. Creating Computer Simulation Systems: An Introduction to the High Level Architecture. Prentice Hall PTR, 1999. 97~108. http://www.phptr.com.

[7] Defense Modeling and Simulation Office. RTI 1.3-Next Generation Programmer's Guide Version 4. 2001. http://www.dmso.mil.

附中文参考文献:

[4] 欧阳伶俐,宋星,卿杜政,郝江波,王锦.HLA 时间管理与 PDES 仿真算法研究.系统仿真学报,2000,12(3):237~240.

敬 告 作 者

《软件学报》创刊以来,蒙国内外学术界厚爱,收到许多高质量的稿件,其中不少在发表后读者反映良好,认为本刊保持了较高的学术水平.但也有一些稿件因不符合本刊的要求而未能通过审稿.为了帮助广大作者尽快地把他们的优秀研究成果发表在我刊上,特此列举一些审稿过程中经常遇到的问题,请作者投稿时尽量予以避免,以利大作的发表.

1. 读书偶有所得,即匆忙成文,未曾注意该领域或该研究课题国内外近年来的发展情况,不引用和不比较最近文献中的同类结果,有的甚至完全不列参考文献.

2. 做了一个软件系统,详尽描述该系统的各个方面,如像工作报告,但采用的基本上是成熟技术,未与国内外同类系统比较,没有指出该系统在技术上哪几点比别人先进,为什么先进.一般来说,技术上没有创新的软件系统是没有发表价值的.

3. 提出一个新的算法,认为该算法优越,但既未从数学上证明比现有的其他算法好(例如降低复杂性),也没有用实验数据来进行对比,难以令人信服.

4. 提出一个大型软件系统的总体设想,但很粗糙,而且还没有(哪怕是部分的)实现,很难证明该设想是现实的、可行的、先进的.

5. 介绍一个现有的软件开发方法,或一个现有软件产品的结构(非作者本人开发,往往是引进的,或公司产品),甚至某一软件的使用方法.本刊不登载高级科普文章,不支持在论文中引进广告色彩.

6. 提出对软件开发或软件产业的某种观点,泛泛而论,技术含量少.本刊目前暂不开办软件论坛,只发表学术文章,但也欢迎材料丰富,反映现代软件理论或技术发展,并含有作者精辟见解的某一领域的综述文章.

7. 介绍作者做的把软件技术应用于某个领域的工作,但其中软件技术含量太少,甚至微不足道,大部分内容是其他专业领域的技术细节,这类文章宜改投其他专业刊物.

8. 其主要内容已经在其他正式学术刊物上或在正式出版物中发表过的文章,一稿多投的文章,经退稿后未作本质修改换名重投的文章.

本刊热情欢迎国内外科技界对《软件学报》踊跃投稿.为了和大家一起办好本刊,特提出以上各点敬告作者.并且欢迎广大作者和读者对本刊的各个方面,尤其是对论文的质量多多提出批评建议.