HLA 时间管理中的死锁问题研究

刘步权, 王怀民, 姚益平
(国防科技大学计算机学院, 湖南长沙 410073)

摘 要: 研究高层体系结构中的死锁问题对于正确理解 HLA 标准中的时间推进机制、设计运行支撑平台 RTI 软件中的时间推进服务、以及开发基于逻辑时间的 HLA/RTI 仿真应用等都具有重要的意义. 论文从零前瞻值、时间推进服务、时间管理算法以及死锁的解除等方面对分布式仿真中的死锁问题进行了多方面的研究, 探讨了死锁发生时系统所具有的一系列现象, 论证了发生死锁和不可能发生死锁时的各类情形. 论文揭示了一个带有相当普遍性的原理, 即"水平面原理". 该原理表明: 当死锁发生时, 所有程序都无法向前推进而处于相对静止状态, 犹如平静的水面一样. 在 HLA/RTI 仿真中表现为处于死锁状态的所有程序具有相同最大可用逻辑时间, 此时所有程序均无法继续向前推进.

关键词: 分布式仿真; 高层体系结构 (HLA); 时间管理; 死锁; 水平面原理
中图分类号: TP391.9 文献标识码: A 文章编号: 0372-2112 (2006) 11-2038-05

Research on Deadlock in the HLA Time Management

LIU Bu-quan, WANG Huai-min, YAO Yi-ping
(*School of Computer, National University of Defense Technology, Changsha, Hunan 410073, China*)

Abstract: It is rather important to study deadlocks in the High Level Architecture (HLA) for correctly comprehending time advancing mechanism in HLA standards, designing time management services in Runtime Infrastructure (RTI), and developing HLA/RTI simulations based on logical time. This paper investigates deadlocks in the HLA time management from different aspects, including deadlocks resulted from zero lookahead, from time advance services, and from time management algorithms as well as the resolution of deadlocks. A series of interesting phenomena are also explored, and varieties of cases in which deadlocks must happen or must not happen are also demonstrated. Particularly, the paper brings forward a rather common principle called Water Surface Rule. This rule shows that all programs cannot advance further and all of them are in relatively quiet status when deadlock occurs, which seems to be the calm water surface. In HLA/RTI simulations, all programs in deadlock status have the identical Greatest Available Logical Time (GALT) so that none of them can advance logical time any further.

Key words: distributed simulation; high level architecture(HLA); time management; deadlock; water surface rule

1 引言

事件排序是分布式系统中的经典问题. 事件排序可以有两种典型方式: 物理时钟排序和逻辑时钟排序. 一方面, 由于时钟的漂移特性使得分布式系统中各个结点的物理时钟很难精确一致; 另一方面, 基于 GPS(全球定位系统) 或时统等物理设备的对时机制并不能够有效地应用于所有的分布式仿真. 例如, 并非所有的场合都能够轻易地获得这些设备; 在尽可能快的并行离散事件仿真(PDES) 中, 每个分布式进程尽可能快地向前推进仿真, 这样基于物理时钟的时间推进机制就不能够很好地适应这类应用.

早在 1978 年, Lamport 就已经指出[1]: 在分布式系统中, 有时很难确认两个事件发生的先后次序. Lamport 在论文中提出了基于逻辑时钟的事件排序机制. 从此之后, 基于逻辑时间的死锁问题就成为分布式系统中的热点问题[2~4].

分布式算法的死锁问题与具体的时间管理算法相关, 不同的分布式算法所遇到的死锁问题以及与之相关的死锁检测、避免和解除算法都具有各自不同的特点. 另外, 在并行离散事件仿真领域, 前瞻值(Lookahead) 是时间管理算法中的最常用的概念之一, 前瞻值是 Chandy 和 Misra 在文献[5] 中提出的概念, 用于解决与特定仿真应用相关的死锁问题.

高层体系结构(High Level Architecture, HLA) 于 2000 年 9 月, 被正式接纳为 IEEE1516 标准. 正如时间管理设计者所说, HLA 中的时间管理综合了以往所有仿真的时间管理机制, 能

收稿日期: 2005-10-24; 修回日期: 2006-05-31
基金项目: 国家自然科学基金(No. 60373024); 国家 973 重点基础研究发展规划(No. 2005CB321804)

够满足目前已有的各类分布式仿真应用. 然而, 并行离散事件仿真领域中的时间管理机制与 HLA 中的时间管理机制并不相同. 在 HLA 中, 由 RTI(Runtime Infrastructure) 统一负责时间管理, 时间管理机制与上层应用程序即盟员(Federate) 是相互分离的; 但在 PDES 中却没有进行时间管理的中心机构, 时间管理和时间推进均由分布式进程负责. 因此, 并行离散事件仿真领域中已有的时间管理算法很难移植到 HLA/ RTI 中来.

HLA 中的时间管理是一种基于逻辑时间的时间推进机制, 因此对于死锁问题的研究也同样适用于 HLA 中的时间管理机制. 与并行离散事件仿真类似, HLA 中的死锁与前瞻值和时间管理算法密切相关; 另外, HLA 中的死锁还与盟员所使用的时间推进服务紧密相关. 目前关于死锁问题的研究只是针对特定的算法来研究, 或者简单地讨论由前瞻值而引起的死锁问题, 缺乏对死锁问题全面而系统的研究. 例如文献[6] 讨论了前瞻值的作用和可能产生的死锁问题, 文献[7] 在提出自己的时间管理算法之后讨论了相关算法的死锁检测和解除问题, 文献[8, 9] 针对 Frederick 提出的时间管理算法对死锁问题进行了讨论, 文献[10] 对 HLA 中死锁的解除问题进行了探讨. 在这些论文讨论的基础上, 本文试图比较完整地研究 HLA 时间管理中与死锁相关的各类问题.

2　前瞻值死锁

前瞻值是保守同步机制的基础. 前瞻值的作用可以用图 1 形象地说明, 其作用可以概括为两点:

(1) 提高仿真程序的并行性;

(2) 避免死锁.

前瞻值就好比图 1 中所示的盟员之间的绳子长度:

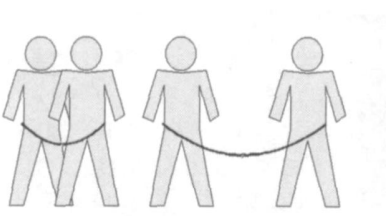


图 1　前瞻值

(1) 如果把两人紧紧地捆绑在一起, 则相当于前瞻值为零的情形, 盟员向前推进时就比较困难, 一个人不可能先于另外一人前进, 两个人只能够同步前进. 也就是说, 如果一个人不向前移动, 则另外一个人就无法前进.

(2) 如果盟员之间能够保持一定的距离, 则相当于前瞻值大于零的情形, 两个人前进时就相对自由得多; 前瞻值越大, 则盟员程序的并行性越大. 前瞻值反映了盟员程序之间的制约程度. 总之, 前瞻值越小, 则同步效果越好; 前瞻值越大, 则并发效果越好.

前瞻值引起的死锁问题与仿真应用、时间管理服务、时间管理算法等密切相关. 前瞻值为零时可能导致死锁, 但是否存在死锁与具体的仿真应用相关, 本文的定理 1 和定理 2 给出了前瞻值为零时的两类不同的死锁现象. 譬如说彼此间保持固定间距(相当于前瞻值为零) 的一列士兵, 如果没有指挥员的口令, 则谁都无法跨出第一步, 相当于系统进入死锁状态; 但在指挥员的口令下, 就可以有节奏地、有规律地向前步进, 这样的系统就不构成死锁. 前瞻值非零时也可能导致死锁, 但是否存在死锁与 RTI 中的时间管理算法以及仿真应用所采用的时间推进服务相关. 由于前瞻值为零时会导致 HLA 时间管理的复杂性, 因而在早期的时间管理服务规范中并不允许前瞻值为零, 文献[11, 12] 对此进行了描述. 如果不允许前瞻值为 0, 则有不少问题就很难解决; 因此随着分布式仿真技术的发展, 后来的 HLA 标准(包括 HLA1. 3 和 IEEE1516) 都支持前瞻值为零的情形.

3　时间服务死锁

HLA 标准 IEEE1516 中的时间管理提供了多个时间服务用于仿真盟员的时间推进, 包括 TimeAdvanceRequest (TAR)、TimeAdvanceRequestAvailable (TARA)、 NextMessageRequest (NMR)、NextMessageRequestAvailable(NMRA)、FlushQueueRequest (FQR) . IEEE1516 中的 NMR 和 NMRA 相当于 HLA1. 3 中的 NER 和 NERA. 因为按照 HLA 标准, FQR 服务的推进请求总能够得到满足, 不会导致仿真死锁. 因此, 本文只讨论其它 4 个服务. 一个盟员在请求推进逻辑时间时, 关键在于计算该盟员的最大可用逻辑时间 GALT(Greatest Available Logical Time) , 在 IEEE1516 之前 GALT 也称之为 LBTS(Lower Bound Time Stamp) . 本节研究了时间推进服务与零前瞻值之间的辨证关系.

定理 **1**　在一个仿真中, 如果至少有两个盟员的前瞻值为零, 并且使用 TAR 服务推进到同一个逻辑时间点, 则会导致死锁.

证明　假设盟员 i 和 j 的前瞻值为 0, 且使用 TAR 服务请求推进到逻辑时间 T. 若系统中仅有这两个盟员, 则按照 HLA 标准, 每个盟员的 GALT 是另外一个盟员请求推进到的逻辑时间 T. 两个盟员的请求均不能够得到满足, 因而会发生死锁. 如果系统中有其它的盟员, 例如 x. 只要不存在死锁, 则盟员 x 总可以向前推进, 因此存在某一时刻 t, 盟员 $x(x \neq i, j)$ 的逻辑时间> T. 盟员 i 和 j 就成为系统中推进最慢的盟员, i 和 j 会发生死锁, 且其它盟员 x 再不能够向前推进, 于是死锁就发生了.

定理 1 本质上是由于零前瞻值引起的. 为了解决零前瞻值问题, HLA 标准引入了 TARA 和 NMRA 服务. 在定理 1 中, 如果盟员使用 TARA/ NMRA 服务推进逻辑时间, 则不会发生死锁.

定理 2　如果一个仿真满足下列条件, 则不会发生死锁.

(1)所有盟员的前瞻值为零;

(2)所有盟员使用 TAR 服务推进逻辑时间;

(3)所有盟员交叉推进逻辑时间, 即任何时刻没有任何两个盟员会请求推进到相同的逻辑时间点.

证明　由于任何时刻没有任何两个盟员会请求推进到同一个逻辑时间点, 因此可设 i 是整个系统中请求推进最慢的盟员, 则该盟员的 GALT 为所有其它盟员请求推进到的逻辑时间的最小值, 该值> 盟员 i 请求推进到的逻辑时间. 因此盟员 i 的请求总能够得到满足.

图 2 给出了两个盟员组成的系统. 两个盟员使用 TAR 服务交叉推进逻辑时间, 因此推进最慢的盟员的请求总能够得到满足.

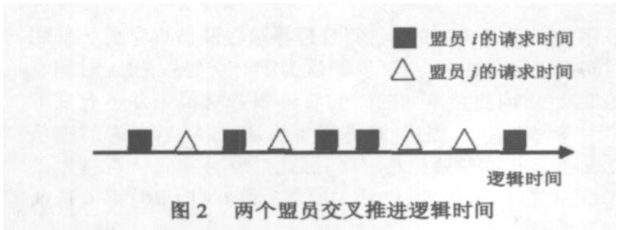


图 2 两个盟员交叉推进逻辑时间

4 时间管理算法死锁

最大可用逻辑时间的计算问题是时间管理中的关键问题之一,这里我们介绍文献[13]中描述的 GALT 算法,即 Frederick 时间管理算法,并通过该算法来揭示 GALT 算法与死锁问题的辨证关系.此部分内容本质上是对文献[8]的修正说明,具体理由可参考文献[9];但与文献[8,9,13]不同的是,本文允许盟员在仿真运行过程中动态地调用 ModifyLookahead 服务修改前瞻值,因而本文中的 Frederick 算法也相应地进行了更改,已不再是原来意义上的算法.在基于 *IEEE*1516 标准研究死锁规律时,需要考虑以下两方面内容:(1)如果盟员处于请求推进状态,则盟员调用 ModifyLookahead 服务修改前瞻值时会被 *RTI* 挂起,不论盟员通过该服务增加或减少前瞻值都会收到 RTI::InTimeAdvancingState 异常.(2)Frederick 算法中的输出时间是指 RTI 计算出的盟员能够发送的 TSO(Time Stamp Order)消息的最大下界,有时一个 TSO 消息如果按照输出时间的定义则不能够发送,但是按照 HLA 标准却能够发送;输出时间是一种相对保守的算法,不能够取代 HLA 标准中的约定.

Frederick 算法是我们所见的第一个比较全面地介绍 GALT 的时间管理算法,研究该算法与死锁的辩证关系具有典型意义.但是有必要指出,Frederick 算法是一种有效的时间管理算法,只是该算法存在死锁;在 RTI 具体运用该算法时会因为检测和解除死锁问题而导致额外的开销.Frederick 介绍的 GALT 算法由算法 1 给出.

算法 **1** Frederick 算法

对于任意盟员 i,GALT 取值为其它盟员的输出时间(OUTPUT)的最小值:

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

(1)如果盟员处于 TAR/TARA 请求推进状态:

$$OUTPUT(i)=T(i)+L(i)$$

$T(i)$ 为盟员 i 请求推进的时间,$L(i)$ 为盟员 i 的 TAR/TARA 推进请求得到 RTI 同意后的前瞻值.

(2)如果盟员处于 NMR/NMRA 请求推进状态:

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

$T(i)$ 为盟员 i 请求推进的时间,$L(i)$ 为盟员 i 的 NMR/NMRA 推进请求得到 RTI 同意后的前瞻值,$LETS(i)$ 为盟员 i 的 TSO 队列中的最小消息的时标,GALT(i) 为盟员 i 的最大可用逻辑时间.

(3)如果盟员处于 Grant 状态:

$$OUTPUT(i)=T(i)+L(i)$$

$T(i)$ 为盟员 i 的当前逻辑时间,$L(i)$ 为盟员 i 的当前前瞻值.

上述算法比较直观,对于 HLA 初学者学习和理解时间管理非常重要,但对于 RTI 设计者而言仍可以进一步完善.基于以上的 GALT 算法,会有下列一些重要的规律.

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,那么对于任意盟员 i,$T(i)$ 为盟员 i 请求推进的逻辑时间,GALT 满足公式:

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

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

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

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

证明 用反证法证明.(1)假设盟员 i 的 GALT$(i)>T(i)$,则不论盟员使用何种时间推进服务,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)即可证得.(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,与死锁矛盾.(4)由(2)和(3)可得.

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,那么对于任意盟员 i,$T(i)$ 为盟员 i 请求推进的逻辑时间,输出时间满足公式:

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

(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,有:$OUTPUT(i)\geqslant GALT(i)$;

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

由定理 3 和算法 1 易证定理 4.

定理 5 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 $n(n\geqslant 2)$,n 个盟员的 GALT 构成一个集合 K:

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

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,$a\neq 1$,使得 $GALT(1)>GALT(a)$.另外,n 个盟员的输出时间构成一个集合 $\Phi=\{O(1),O(2),O(3),\cdots,O(n)\}$.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 $GALT(a)=\min(\Phi-\{O(a)\})=O(1)$.所以 $O(1)<GALT(1)$ 与定理 4(3)矛盾,故 $O(1)$ 不可能是 Φ 中最小值.(2)不失一般性,设 $O(2)$ 为 Φ 中最小值,且 $O(1)>O(2)$.由算法 4.2 知:$GALT(1)=\min(\Phi\{O(1)\})=O(2)$;$GALT(2)=\min(\Phi-\{O(2)\})\geqslant O(2)=GALT(1)$.若 $GALT(2)>GALT(1)$,则与 $GALT(1)$ 为 K 中的最大值矛盾;所以 $GALT(2)=GALT(1)$.对于其它任意盟员 $i(i\neq 1,2)$,$GALT(i)=\min(\Phi\{O(i)\})=O(2)=GALT(1)$,所以 n 个盟员的 GALT 值均相等.

定理 5 具有一定的典型意义,该定理是基于 Frederick 算

法获得的结论; 但该结论同样适用于定理 1. 在定理 1 中, 构成死锁的所有盟员的 GALT 都等于 T. 定理 5 表明: 当死锁发生时, 所有程序都无法向前推进而处于相对静止状态, 犹如平静的水面一样, 此时所有程序都具有相同的最大可用逻辑时间.

定理 6　如果前瞻值大于 0 的盟员处于 TAR/TARA 请求推进状态, 并且该盟员的输出时间最小, 则该盟员的请求一定能够满足.

证明　假设盟员 i 处于 TAR/TARA 请求推进状态, 盟员 i 的前瞻值 $L(i)>0$, 输出时间 $O(i)$ 最小, 则根据算法 1 有: $O(i)=T(i)+L(i)\leqslant\min\{O(j)\}=GALT(i)$, $i\neq j$; 所以 $T(i)<GALT(i)$. 因此, 盟员 i 的推进请求能够满足.

定理 7　如果所有盟员均因为请求时间推进而处于死锁状态, 并且输出时间最小的盟员的前瞻值大于 0, 则该盟员一定处于 NMR/NMRA 请求推进状态.

证明　用反证法证明. 假设盟员不处于 NMR/NMRA 请求推进状态, 由于 FQR 请求总能够得到满足, 因此该盟员一定处于 TAR/TARA 请求推进状态, 由定理 6 知, 盟员的推进请求一定能够满足, 与死锁矛盾.

定理 8　如果前瞻值大于 0 的盟员处于 NMR/NMRA 请求推进状态, 并且系统中所有其它盟员的输出时间大于该盟员的输出时间, 则该盟员的推进请求一定能够满足.

证明　假设盟员 i 处于 NMR/NMRA 请求推进状态, 盟员 i 的前瞻值 $L(i)>0$. 对于任意盟员 j, $O(i)<O(j)$, $i\neq j$. 根据算法 1 有: $GALT(i)=\min\{O(j)\}>O(i)$. 由算法 1 有 $GALT(i)>\min\{T(i)+L(i), LETS(i), GALT(i)\}$. 因此, 一定有 $T(i)+L(i)<GALT(i)$ 或者 $LETS(i)<GALT(i)$, 即 $T(i)<GALT(i)$ 或者 $LETS(i)<GALT(i)$. 盟员的推进请求总能够得到满足.

定理 9　如果所有盟员均因为请求时间推进而处于死锁状态, 并且所有盟员的前瞻值大于 0, 则系统中至少存在两个输出时间最小的盟员, 并且都处于 NMR/NMRA 挂起状态.

证明　假设盟员 i 的前瞻值 >0, 且输出时间最小. 由定理 7 可知, 盟员 i 一定因为 NMR/NMRA 推进请求而被挂起. 假设盟员 i 是系统中唯一具有最小输出时间的盟员, 则任何其它盟员的输出时间一定大于盟员 i 的输出时间, 由定理 8 可知盟员 i 的推进请求能够满足, 与死锁矛盾.

定理 10　如果所有盟员的前瞻值大于 0, 并且不存在输出时间最小的盟员因为 NMR/NMRA 推进请求而被挂起, 则该系统不可能处于死锁状态.

证明　用反证法证明. 假设系统处于死锁状态, 则由定理 9 可知, 一定存在输出时间最小的盟员因为 NMR/NMRA 推进请求而被挂起, 与前提条件矛盾.

5　死锁的解除

死锁的解除有两种经典方法[7]: 一是从算法的角度进行优化, 避免死锁的发生; 二是在运行过程中动态地检测死锁, 当死锁发生时, 采用一定的规则打破死锁. 文献[7]在提出了自己的算法后, 介绍了相应算法的死锁检测和解除方法.

就 Frederick 算法而言, 文献[8]提出了一个有效的改进算法. 下面我们给出一个更为有效的算法, 该算法允许盟员在仿真推进过程中动态地更新前瞻值, 并能够有效地避免死锁. 采用该算法的 RTI 软件不需要在运行过程中动态地检测死锁的发生, 因而能够提高 RTI 时间管理的决策效率; 实际上, 该算法正是我们在 StarLink[14] 中所采用的实现方法, 该算法在文献[10, 15]中都有说明. 当然, 如果允许前瞻值为零, 则该算法仍然具有定理 1 和定理 2 的特性, 这是由零前瞻值的本质所决定的. 另外, 除了讨论与死锁相关的问题外, 一个时间管理算法还需要证明其有效性. 关于下列算法的正确性证明是一个非常复杂的过程, 我们将在相关论文中对其进行严格证明和解释. 在文献[8~10]中, 盟员的前瞻值假设保持不变; 而允许前瞻值变化将使得时间管理算法的证明过程更加复杂和具有挑战性.

算法 2　身高测量法

对于任意盟员 i, $GALT(i)=\min\{S(j)\}$; $i\neq j$.

$$S(i)=\begin{cases}T(i)+L(i); \text{如果盟员处于 } Grant/TAR/TARA \text{ 状态}\\ \min\{T(i), LETS(i)\}+L(i); \text{如果盟员处于 } NMR/NMRA \text{ 状态}\end{cases}$$

$S(i)$: 盟员 i 的身高.

$T(i)$: 当盟员 i 处于 Grant 状态时, $T(i)$ 为盟员 i 的当前逻辑时间; 当盟员 i 处于请求推进状态时, $T(i)$ 为盟员 i 请求推进的逻辑时间.

$L(i)$: 当盟员 i 处于 Grant 状态时, $L(i)$ 为盟员 i 的当前前瞻值; 当盟员 i 处于请求推进状态时, $L(i)$ 为盟员 i 的请求得到同意后的前瞻值.

$LETS(i)$: 盟员 i 的 TSO 队列中的最小消息的时标.

最后, 我们强调两点. (1) 定理 5 表明, 在死锁状态, 因为所有挂起盟员的最大可用逻辑时间相等而导致所有盟员均无法向前推进. 水通常由高处流向低处, 定理 5 就好比没有波动的平静的水面, 我们通常将该定理称之为"水平面原理". 就本文而言,"水平面原理"适合于定理 1 和定理 5 两类死锁. (2) 根据 HLA 标准, 如果盟员处于请求推进状态, 则在计算 GALT 时, 所使用的前瞻值是指盟员的时间推进请求获得 RTI 同意后的前瞻值, 该前瞻值与盟员的当前逻辑时间、当前前瞻值、请求推进的逻辑时间、请求更新的前瞻值、时间推进服务、盟员 TSO 消息队列中的最小时标 LETS 等相关, 因此我们说本文中的 Frederick 算法已不再是文献[8~10]和文献[13]中的算法, 本文中的身高测量法与文献[10]中的算法也有本质的不同, 虽然它们在形式上相同, 只是前瞻值表达的意义不同; 当 RTI 同意盟员的请求后, 计算 GALT 所使用的前瞻值为盟员的当前前瞻值. 该条件适合以上各个定理, 否则若将 Frederick 算法和身高测量法中的前瞻值始终取值为盟员的当前前瞻值, 将无法避免时序不一致性现象.

6　结论

基于逻辑时间的死锁是分布式系统中的关键问题之一. 作为分布式系统的一个重要分支, 分布式仿真也无法避免死锁. 高层体系结构 HLA 已于 2000 年 9 月被 IEEE 确定为国际

分布仿真的通用标准.时间管理服务是 HLA 标准中的重要内容,它负责处理仿真中逻辑时间的同步推进,保证分布式仿真中的消息被正确地排序和接收,从而确保事件间的因果关系.本文从四个方面讨论了时间管理中的死锁问题,包括零前瞻值、时间推进服务、时间管理算法而导致的死锁,以及死锁的解除等一系列问题.详细地讨论了各类死锁的原因,对死锁的规律性进行了积极探讨.本文揭示了一个带有普遍性意义的“水平面原理”,即当死锁发生时,所有程序都无法向前推进而处于相对静止状态,犹如平静的水面一样.

参考文献:

[1] L Lamport. Time, clocks, and the ordering of events in a distributed system[J]. Communications of the ACM, 1978, 21(7): 558- 565.

[2] M Maekawa, A E Oldehoeft, R R Oldehoeft. Operating Systems [M]. California: The Benjamin/Cummings Publishing Company, 1987.

[3] A S Tanenbaum. Distributed Operating System[M]. New Jersey: Prentice Hall PTR, 1995.

[4] A S Tanenbaum, M V Steen. Distributed Systems[M]. New Jersey: Prentice Hall PTR, 2002.

[5] K M Chandy, J Misra. Distributed simulation: a case study in design and verification of distributed programs[J]. IEEE Transactions on Software Engineering, 1979, 5(5): 440- 452.

[6] R M Fujimoto. Zero lookahead and repeatability in the high level architecture[DB/OL]. http://www.cc.gatech.edu/ computing/pads/papers.html, 2005- 08.

[7] 张龙, 尹文君, 柴旭东, 刘民. RTI 系统时间管理算法研究[J]. 系统仿真学报, 2000, 12(5): 494- 498.

[8] 刘步权, 王怀民, 姚益平. 一种无死锁的时间管理算法[J]. 软件学报, 2003, 14(9): 1515- 1522.

[9] 唐京桥, 侯朝桢. HLA 中时间管理算法死锁的规律性[J]. 计算机工程, 2005, 31(15): 27- 29.

[10] 胡依娜, 侯朝桢, 唐京桥. HLA 时间管理中死锁的解除[J]. 系统仿真学报, 2005, 17(6): 1396- 1399.

[11] R M Fujimoto. HLA time management: design document[DB/OL]. http://www.cc.gatech.edu/computing/pads/papers.html, 2005- 08.

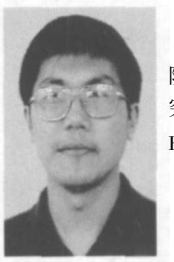
[12] C D Carothers, R M Weatherly, R M Fujimoto, A L Wilson. Design and implementation of HLA time management in the RTI version F.0[A]. Proceedings of the 1997 Winter Simulation Conference[C]. Atlanta, Georgia, USA, 1997. 373- 380.

[13] F Kuhl, R Weatherly, J Dahmann. Creating Computer Simulation Systems: An Introduction to the High Level Architecture [M]. New Jersey: Prentice Hall PTR, 1999.

[14] B Liu, H Wang, Y Yao. Data consistency in a large-scale runtime infrastructure[A]. Proceedings of the 2005 Winter Simulation Conference[C]. Orlando, Florida, USA, 2005. 1787- 1794.

[15] 刘步权. 分布式仿真运行支撑平台中时间管理服务的研究[D]. 长沙: 国防科技大学研究生院, 2004.

作者简介:



刘步权 男, 1969 年 5 月生于江苏姜堰, 国防科技大学计算机学院博士, 副研究员, 主要研究方向为分布式仿真和高性能计算.
E-mail: bqliu@nudt.edu.cn

王怀民 男, 1962 年生于江苏南京, 国防科技大学计算机学院教授, 博士生导师, 主要研究方向为分布式对象、Agent 技术、网格计算和网络安全等.

姚益平 男, 1963 年生于湖南邵东, 国防科技大学计算机学院研究员, 博士生导师, 主要研究方向为分布式仿真与虚拟现实等.