

层次式体系结构下一种似然时标系统

陈春鹏^{1),2)} 王怀民¹⁾ 姚益平¹⁾

¹⁾(国防科技大学计算机学院 长沙 410073)

²⁾(91404 部队 河北 秦皇岛 066001)

摘 要 当前,分布式系统具有越来越大的规模,而且可能层次地分布在广域网上,这些属性增加了保证分布式系统中时序关系的难度.向量时钟可以准确地探测事件间的时序关系,但是向量时钟的维度与节点数相同,就引起了效率和可扩展性问题.该文提出了层次体系结构下确定分布式系统时序关系的一种优化解决方案.首先给出了分布式系统的一种层次式体系结构,其中独立的子组通过代理进程互连,组成集群完成特定目标;在这种结构中,采用“分而治之”的策略,提出了一种似然时标系统,其特点是消息传输时的时间标签仅仅是系统时标的一部分.该时标系统能够探测大规模分布式系统中消息间的因果关系,而且具有较好的效率和可扩展性.

关键词 似然时标系统;层次式结构;代理进程;分而治之;仿真

中图法分类号 TP306

A Plausible Timestamp System in Hierarchical Architecture

CHEN Chun-Peng^{1),2)} WANG Huai-Min¹⁾ YAO Yi-Ping¹⁾

¹⁾(School of Computer Science, National University of Defense Technology, Changsha 410073)

²⁾(No. 91404 Troop, Qinhuangdao, Hebei 066001)

Abstract Currently, in distributed systems there are more and more processes which are hierarchically distributed in the wide area network, which make it hard to realize the causality in the distributed systems. In a distributed system with N sites, the precise detection of causal relationships between events can only be done with vector clocks of size N . This gives rise to scalability and efficiency problems for vector clocks. In this paper, a kind of hierarchical architecture is described in which proxy process is used to exchange messages between the subgroups and global group can be composed to achieve some objectives. A plausible timestamp system is presented using divide-and-conquer approach, in which only a part of the timestamp is attached to the message as the tag whenever the message is transmitted. The timestamp system allows to detecting causality between events in a large distributed system with high degree of accuracy, enhancing the scalability of the system and minimizing the affection because of adding or reducing federates.

Keywords plausible clock; hierarchical architecture; proxy process; divide-and-conquer; simulation

1 引 言

分布式系统由一组不同的普通进程组成,这些

进程在空间上分离,通过交互消息完成相互间的通信^[1].在分布式系统中,消息传输延迟不能像单个进程内部的事件间的传输延迟一样忽略不计^[1].实际应用中的消息延迟通常是可变的和不可预测的,并

且进程通常不能访问全局时钟,也不能访问完全同步的局部时钟。这样,在分布式系统运行过程中就有可能产生在现实世界中不可能的事件序,这是人们不希望发生的。例如,在分布式仿真运行中,可能出现结果事件看起来像是在原因事件之前发生的情况,这与现实世界中的因果关系相悖^[2]。时标系统的目标就是在分布式系统中尽量避免这类异常的发生,削弱其影响。

Lamport 基于分布式系统中一个事件先于另外一个事件的概念,首先提出了逻辑时间的思想,描述了用逻辑时钟模拟全局时钟捕获因果关系的方法^[1]。该逻辑时钟能够解决互斥问题,能够完成一致性的“快照”。但是,它不具有强一致性的,不能完全捕获并发。为了弥补这一缺陷,后来许多研究人员提出了向量时钟的概念,其中最著名的是 Fidge^[3]和 Mattern^[4]。向量时钟具有很多优良属性,例如,向量时钟能够完全捕获事件间的因果或并发关系,基于长度为 N 的向量时钟的逻辑时间系统同构。然而,Charron-Bost 证明为了维护时钟同构,向量时钟的大小必须至少等于系统中的进程数目^[5]。这样,存储和通信的额外耗费随着系统大小线性增长,使大规模分布式系统中使用向量时钟的成本高得令人望而却步。文献[6]提出了一类时标系统,将其命名为似然时标系统,该系统由一系列组件实现,而组件数目不受系统大小的影响,具有可扩展性,并且能够提供良好的排序精度。使用层次式网络解决可扩展问题是一种由来已久的惯例^[7],本文将对向量时钟进行优化,研究层次式体系结构下的似然时标系统。

文献[8]研究了一个支持成百上千进程的层次式进程组,在每个子组中,通过向量时钟之类的机制,局部的进程和代理进程支持消息的因果序提交,并且研究了所有消息间的全局因果性与子组中的消息的局部因果性间的关系。在文献[9]中,本地子组的进程使用物理或者线型时钟,而广域网上的进程使用向量时钟,并且该文献讨论了怎样利用每个子组中的局部同步机制因果性地提交消息。但是,这些方法只适应特殊的案例,同时因为该时钟仅仅具有两个层次,所以不能解决可扩展性问题。

文献[10]中提出了一种叫做层次式向量时钟的时标系统,该系统适合底层网络的层次式结构,能够探测大规模分布式系统中事件间的因果关系,并且具有高度的精确性。但是,每个进程需要维护较长的时标(包括进程所在本地子组以及所有祖先子组,直至顶层组的时标),而且处于底层的同一组内的进程

间通信,需要整个时标作为时间标签。所以,由于层次式系统的通信局部性,同一组内的进程间通信频繁,这样的时标系统将会增大总的通信负载。

本文提出了层次式分布条件下的一种似然时标系统,该时标系统采用“分而治之”的策略,不同子组的时标互不相关,不同的进程也许具有不同大小的时标,属于同一个子组的进程才会有相同大小的时标,其特点是消息传输时的时间标签仅仅是系统时标的一部分。该时标系统不仅能够探测大规模分布式系统中消息间的因果关系,而且它并不限定层次结构,完全适合各种层次式结构的底层网络,消息的额外消耗不随着通信进程的层次距离发生改变,所以能够解决效率和可扩展性问题。

2 应用背景和相关工作

当前,基于物理、地理、管理等方面的因素,分布式系统的网络体系结构具有本质的层次性,例如,多个园区的 LAN 通过骨干网互连等。而在 P2P(Peer-to-Peer)和网格中的分布式应用,几百甚至成千上万的进程广泛分布在 Internet 上相互交换信息,也因为网络安全、消息传输的额外消耗等因素,希望考虑分布式协议的多级层次结构^[11-12]。所以层次式体系结构下的时标系统具有广阔的应用前景,而我们对层次式时标系统的研究最初源自大规模分布式仿真系统中保持因果序的应用需求。

在分布式仿真领域,高层体系结构(High Level Architecture, HLA)是在分布式条件下实现计算机建模和仿真的标准体系结构,被正式接纳为 IEEE 1516 标准。HLA 的主要目的是支持仿真盟员的互操作和可重用^[13],但是现在的大规模仿真中需要在不同的级别实现不同仿真组件的互操作和可重用(例如,模型级、盟员级、联盟级)。另外,尽管国际上有支持 HLA 的各种商业 RTI(Runtime Infrastructure)实现,但是当前的分布式仿真系统可能由分布在 Internet 之类的广域网上的成百上千的盟员进行协作,所以,几乎没有一种 RTI 实现能够在一个单层联盟中满足所有系统需求。为了满足大规模仿真以及跨广域网分布式仿真的需求,需要研究层次式结构的联盟群,在不同的联盟中使用不同的 RTI 满足其局部特定需求。文献[14]首先给出了联盟群的定义:为了完成共同的目标协同工作的 RTI 与联盟的集合,并且指出提供联盟间最简便通信的方式就是盟员代理(Federate Proxy, FP)。在层次式联盟群

代理进程是层次式结构中的关键部件,在整个分布式系统中具有举足轻重的桥梁作用。代理进程的主要目的是在子组之间提供一种松耦合的、有效的、透明的互连,即代理进程应当使得在物理上、地理上分布的子组之间实现“无缝”交互。所以,代理进程的任务就是观察一个子组,然后在另一个子组中重新展现其必要的行为。代理进程本身通常没有自己独立的行为模型,逻辑内容相关的信息可以由本子组中的某些进程公布。由于代理进程的转发功能,系统中的远程进程可以通过代理进程得到这些信息。在该体系结构中,代理进程仅仅为局部子组和其上级组提供了一条消息传播的通道,兄弟子组间的信息交互可以通过其父亲组实现。由于代理进程总是严格地连接两个上下级关系的组,所以使代理进程得到简化,而使系统的体系结构规整而明确。因为代理进程表面上看起来跟其它进程一样,所以子组能够透明地加入到上层组中,实现层次性。而在运行时,同一组中,是否使用代理进程对组中的所有其它成员来说都是完全透明的。这里,代理进程不能采用循环的方式连接组。

代理进程将接收的消息按照时标排序,如果代理进程实现子组间的网络通信,消息在代理进程内部传输时去掉其时标,然后转发,如果新的子组中需要时标,则根据新的子组中的规则像在其它普通

进程中一样,在消息离开进程前添加时标。但是,消息在代理进程中必须保证其先进先出 FIFO 的管道特性,这一点至关重要。

本文研究在多级层次结构系统中的时标系统。其思想是把向量时钟的思想递归地应用到层次的每一级,该时标系统不将层次级别限定到任何固定的数目,随着数目的增加具有自然的扩展性。总之,该时标系统能够探测大规模分布式系统中事件间的因果关系,而且具有高度的精确性,其完全适合底层网络的层次式结构,随着分布式站点数目的增加能够得体地扩展。

4 系统模型

这部分主要建模分布式系统的层次式结构以及这种体系结构对时标系统的支持。

4.1 层次式结构模型

任何层次式分类方案都会产生一个树型结构^[20] (在本文中,有关树的定义和属性都来自文献[20])。在层次式组中,进程组是分支节点,普通进程作为树型结构的叶节点,互连组成子组,子组由代理进程互连,组成树型结构,如图 2 所示。因为在树中仅仅考虑节点的相对方向,而不考虑它们的顺序,所以该树可以认为是导向树。

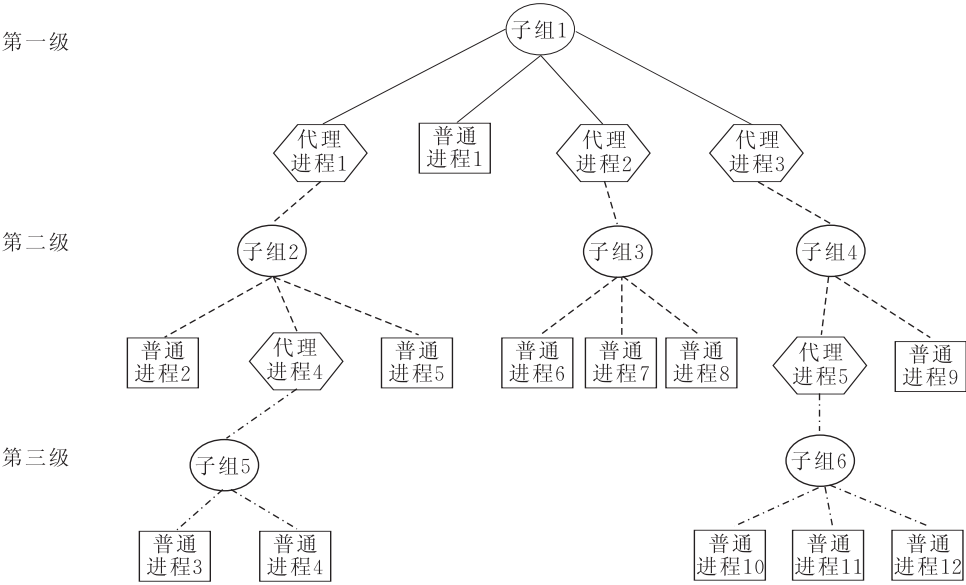


图 2 层次式系统的树型模型

在层次式组中,子组中的每一对进程直接交互数据。不管子组采用 Peer-to-Peer 或者 Client-Server 结构,其中的因果序能够得到保证。而不同子组的进程对通过代理进程间接地交互数据。这样,在全局组

中,所有局部组都可以看作具有 FIFO 属性的节点。

我们对问题进行如下建模:一个层次式系统 $C = (G, P_p, P_s, E, M)$ 是一个互连的无向图,其中, G 是子组的集合(不能为空集), P_p 是代理进程的集

合, P_s 是普通进程的集合, 这 3 个集合中的元素构成图的节点, E 是将节点对互连的边的集合, M 是节点间传输的消息的集合. 这样, 层次式系统的通信网络很明显是一个导向树, 而且至少是弱连通的.

4.2 消息传输和时标系统

在分布式系统中, 每个子组中的进程直接相互通信, 进程间的通信通道可以假定是可靠的. 为了反映时标系统的层次性、可嵌套性和可扩展性, 本文假定每个子组中的时标系统都是向量时钟. 代理进程因为处于两个不同的子组, 所以拥有两个互不相关的向量时钟. 每个进程依据消息的时标, 自主地决定消息的接收和提交顺序.

在本文中, 消息可以流动到全局组的任何位置, 而且在没有被普通进程接收前可以看作是同一个消息. 一对局部消息 m_1 和 m_2 要通过同一代理进程, 即使这两个消息在局部子组中是并发的, 代理进程根据 FIFO 规则, 按照接收的顺序将这两个消息转发出去, 此后这两个消息将有序地在系统中传输. 所以, 一对消息即使在局部子组中具有因果上的先后关系, 也可能在全局组中具有并发关系.

假定在子组 $G_i \in G$ 中的进程 $p_i \in P_s$ 向在子组 $G_j \in G$ 中的另一个进程 $p_j \in P_s$ 发送消息, G_i 和 G_j 通过代理进程 $p \in P_p$ 直接相连. 首先, 进程 p_i 向其子组的代理进程 p 发送本地消息 $m \in M$, p 将消息 m 原有的时标去掉, 然后根据 p 在 G_j 中的时钟为 m 附加新的时标, 再把消息 m 发送到 G_j 的 p_j . 在系统中, 如果消息 m 通过代理进程在子组之间传输, 都按照这样的协议推进, 直至到达目的进程.

5 解决方案

在这一部分, 我们采用分而治之的策略, 针对层次式系统的特点提出一种似然时标系统, 该系统既能够保证消息的因果序传输, 又能够提高效率, 维持系统的可扩展性. 然后, 我们给出与之相应的消息传输协议.

5.1 基本定义

在分布式系统中, 对于一个进程的内部事件, 如果没有相应的消息发出, 对其它进程来说是不可知的. 消息的发送和接收表示节点间的信息流, 建立了从发送节点到接收节点之间的因果关系. 为了简化, 在本文中使用包含事件通知的消息代替事件本身, 表示事件间的关系.

定义 1. 假定 $m_1, m_2 \in M$, 在它们从同一个节

点发出的情况下, m_2 是 m_1 的下一条消息 (在它们从不同节点发出的情况下, 接收和提交后的下一条消息), 则称 m_2 和 m_1 之间存在直接因果关系, 记作: $m_1 \rightarrow m_2$. 如果 $m_1 \rightarrow m_2, m_2 \rightarrow m_3, \dots, m_{n-1} \rightarrow m_n$ ($n > 1, n \in Nat$), 则称 m_n 和 m_1 之间存在因果关系, 记作: $m_1 \rightsquigarrow m_n$. 如果 m_2 和 m_1 是同一条消息, 记作: $m_2 = m_1$. 如果 $\neg(m_1 \rightsquigarrow m_n) \wedge \neg(m_n \rightsquigarrow m_1)$, 则称 m_n 和 m_1 之间并发, 记作: $m_1 \parallel m_n$.

这就是说, m_2 中的事件是 m_1 中事件的直接结果, m_1 中的事件是 m_2 中事件的直接原因. 很明显, 因果关系具有反自反、反对称和传递的性质, 即关系 \rightsquigarrow 是严格偏序关系.

我们的兴趣仅仅在于到达同一进程的消息之间的因果关系, 而且时标系统的最终目的就是判断收到的消息能否立即接收和提交运行.

定义 2. 在本文中, $S(m_1, u, v)$ 表示消息 m_1 从节点 u 发出流经边 $(u, v) \in E$, $R(m_1, u, v)$ 表示消息 m_1 流经边 $(u, v) \in E$ 由节点 v 接收.

通过借鉴 Lamport 的“happen before”关系^[1], 我们定义发送消息和接收消息相关活动之间的关系.

定义 3. 发送消息和接收消息的活动 a, b, c 之间的关系. (1) 如果 a 和 b 是发生在同一节点的发送消息和接收消息的活动, a 发生在 b 之前, 则称 $a \angle b$; (2) $S(m_1, u, v) \angle R(m_1, u, v)$; (3) 如果 $a \angle b$ 且 $b \angle c$, 则 $a \angle c$.

似然时标系统 (Plausible Time Stamping System, PTSS) 是一种对事件的时间调度机制, 使得基于递增时钟值的事件顺序与因果顺序一致. 下面是形式化定义^[6]:

对分布式系统的消息集合 M , 一个时标系统 (Time Stamping System, TSS) P 具有如下结构: $(\langle S, \Rightarrow \rangle, P.stamp, P.tag)$. 其中, S 是时标值的集合 (其细节不在该定义中确定); \Rightarrow 是定义在集合 S 上的具有反自反和传递性的关系, 这样 $\langle S, \Rightarrow \rangle$ 就是一个严格偏序; $P.stamp$ 是从 M 到 S 上的时标赋值函数; $P.tag$ 是附加到每个发出消息上的结构.

我们用 $P.stamp(m)$ 或简写为 $P(m)$ 来表示一个消息 m 的时标. 时间标签的格式可以与时标的格式相同, 如果能够简化, 更好. 当消息发送的时候, 将 $P.tag$ 应用到当前的逻辑时间, 生成标签. 消息 m 发送时, 带有的标签记作 $P.tag(m)$ ^[6]. 为了简化而不失一般性, 在子组 i 中消息 a 的时间标签记作 $P_i(m)$ 而不是 $P.tag(m)$.

定义 4. 一个时标系统 P 中, 对于 $m_1, m_2 \in$

$M, m_1 = m_2$ iff $P(m_1) = P(m_2)$ 且若 $m_1 \rightsquigarrow m_2$, 则 $P(m_1) \Rightarrow P(m_2)$, 那么 P 是似然 TSS^[6].

一个似然 TSS 满足弱时钟条件, 并且, 它指派的时标与事件之间是一一对应关系, 其目标是提供更好的排序精度.

在本文中, 用于标量和向量的小于关系“ $<$ ”的定义是众所周知的, 可以在相关文献中获得, 例如文献[6].

定义 5. 我们将层次式 TSS(HTSS)形式化地定义为一种结构 $P(\langle S, \Rightarrow \rangle, P.stamp, P.tag)$, 其中:

(1) S 是型如 $\langle W, V \rangle$ 的集合, 其中: W 是整数 j 和 k 的 2 维向量, j 标识层次式系统中的子组, k 标识子组中的站点, 这些站点可能是普通进程或者代理进程; V 是 n 维向量 $\langle V[1], \dots, V[i], \dots, V[n] \rangle$, n 是层次式系统中子组的个数, 向量中的元素是对应子组中的向量时钟. V 中的下标 i 表示层次式系统中每个子组的标识 ID. $V[i]$ 是消息在子组 i 中传播时附加的时标, 在不同的子组中该时标不同. 如果消息 a 并不到达子组 G_i , 则 $V[i](m_1) = \max(V[i](m_2))$ 对于所有 $m_2: m_2 \rightsquigarrow m_1$.

(2) $P.stamp$ 按照以下规则定义.

(R0) 初始值:

每个子组的初始化过程独立完成, 一般将向量时钟的每个元素设置为 0.

(R1) 子组 j 中站点 i 发送或转发消息前:

$$V[j][i] = V[j][i] + 1.$$

(R2) 子组 j 中站点 i 接收到带有时标 W 的消息时:

$$V[j][k] = \max(V[j][k], W[k]), \forall k;$$

$$V[j][i] = V[j][i] + 1.$$

(R3) 带时标 V 的消息流经代理进程时:

从消息中除掉时标 V , 消息按照 FIFO 的顺序在代理进程中传输.

(3) 假定 $\langle W_i, V_i \rangle, \langle W_j, V_j \rangle \in S$, 则

$$\langle W_i, V_i \rangle \Rightarrow \langle W_j, V_j \rangle \text{ iff } V_i < V_j,$$

$$\langle W_i, V_i \rangle = \langle W_j, V_j \rangle \text{ iff } (W_i = W_j \wedge V_i = V_j),$$

$$\langle W_i, V_i \rangle \parallel \langle W_j, V_j \rangle \text{ iff } \neg(\langle W_i, V_i \rangle = \langle W_j, V_j \rangle) \wedge$$

$$\neg(\langle W_i, V_i \rangle \Rightarrow \langle W_j, V_j \rangle) \wedge \neg(\langle W_j, V_j \rangle \Rightarrow \langle W_i, V_i \rangle).$$

很明显, \Rightarrow 是定义在元素 S 上的具有反自反和传递性的关系, 这样, $\langle S, \Rightarrow \rangle$ 是严格偏序.

(4) $V.tag(m)[i] = V[i]$. 当消息在子组 i 中流动时, 只有时标的第 i 个元素附加到该消息.

5.2 基本假定

在进一步说明问题前, 有必要声明一些有用的假定. 这些假定表示为了使用这一理论, 问题环境应满足的条件.

假定 1. $R(m_1, u, v)$ iff $S(m_1, u, v)$. $R(m_1, u, v) \angle R(m_2, u, v)$ iff $S(m_1, u, v) \angle S(m_2, u, v)$.

分布式系统必须首要考虑的是组件通信的消息通道的属性. 假定 1 意味着通信是可靠、有序的, 消息的接收顺序与发送顺序一致. 这样的通信协议随处可见, 例如, 广泛使用的 TCP/IP, FDK 中使用的 Fast Message 协议, 都能可靠、有序地提交传输的数据.

假定 1 对几乎所有采用保守同步的时标系统实现都是至关重要的. 不管在何种体系结构下, 没有这样的假定, 在没有其它附加条件的前提下, 保守同步都很难保证, 所以该假定是几乎所有保守同步的时标系统的前提^[21].

假定 2. 若 m_1 和 m_2 从同一站点 u 发出, 则 $S(m_1, u, *) \angle S(m_2, u, *)$ iff $m_1 \rightsquigarrow m_2$.

该假定是说从同一进程发出的消息必须遵循时间进化的递增顺序, 该假定保证了消息有序发出. 该假定是保持消息的相对顺序、防止系统运行中的不可能事件序的發生的基本要求.

假定 1 和假定 2 保证了源自同一进程的因果消息不会在系统中引发因果异常.

假定 3. 在平面子组中使用该时标系统能够避免因果异常的发生.

单个进程不会因为自己的消息发生因果异常, 而该时标系统在平面系统中即为向量时钟, 不会引发因果异常. 我们将该假定作为归纳前提, 在其成立的基础上, 利用层次式系统的嵌套性, 能够证明使用该时标系统, 整个层次式系统能够避免因果异常的发生.

文献[3-4]中已经证明标准的向量时间协议能够正确捕获分布式系统内部事件间的因果关系, 该假定与之一致.

5.3 基本结论

下面, 我们将证明时标系统 HTSS 是似然 TSS.

定理 1.

假定 $C = (G, P_p, P_s, E, M)$ 是使用 HTSS $P(\langle S, \Rightarrow \rangle, P.stamp, P.tag)$ 的层次式系统, 其中 $m_1, m_2 \in M, G_i \in G$.

如果 $m_1 \rightsquigarrow m_2$ 且 m_1, m_2 都被 G_i 接收, 则 $P_i(m_1) <$

$P_i(m_2)$.

证明.

假定 m_1 从进程 j 发出, $j \in P_s$. 如果 j 是 G_i 中的进程, 则很明显, m_1 先到达 G_i , 即 $P_i(m_1) < P_i(m_2)$. 如果 $i \neq j$, 则根据树的基本属性^[20], 从 j 到 i 恰有一条简单路径. 同时, 在使用相关时标系统的任何平面组中没有因果异常发生(假定 3), 所以消息能够在任何节点保持 FIFO. 考虑从 j 到 i 的路径, 对于任何消息 c , 如果 $m_1 \rightsquigarrow m_3$, m_3 不可能先于 m_1 出现在任何节点, 即 m_1 在这条路径上一定首先被接收. 所以 $P_i(m_1) < P_i(m_2)$ (定义 5). 证毕.

引理 1.

假定 $C=(G, P_p, P_s, E, M)$ 是使用 HTSS $P(\langle S, \Rightarrow \rangle, P.stamp, P.tag)$ 的层次式系统, 其中 $m_1, m_2 \in M, G_i \in G$.

如果 $m_1 \rightarrow m_2$, 则 $P(m_1) \Rightarrow P(m_2)$.

证明.

① 对于 m_1, m_2 都被接收的子组 G_i :

$P_i(m_1) < P_i(m_2)$ 定理 1

② 对于 m_2 不被接收的子组 G_i :

$P_i(m_2) = \max(P_i(m_3))$ 对所有 $m_3: m_3 \rightsquigarrow m_2$ 定义 5

$m_1 \rightarrow m_2$ 前提条件

$P_i(m_1) \Leftarrow P_i(m_2)$

③ 对于 m_2 被接收而 m_1 不被接收的进程 G_i :

如果 $\exists m_3, m_3$ 被 i 接收, 且 $m_3 \rightsquigarrow m_2$, 则

对于所有 $m_3, P_i(m_3) < P_i(m_2)$ 定理 1

$P_i(m_1) = \max(P_i(m_3)) < P_i(m_2)$ 定义 5

否则

$P_i(m_1) = 0 < P_i(m_2)$ 定义 5

④ $P(m_1) \Rightarrow P(m_2)$ 定义 5 应用到(1), (2), (3).

证毕.

定理 2.

假定 $C=(G, P_p, P_s, E, M)$ 是使用 HTSS $P(\langle S, \Rightarrow \rangle, P.stamp, P.tag)$ 的层次式系统, 其中 $m_1, m_2 \in M$.

如果 $m_1 \rightsquigarrow m_2$, 则 $P(m_1) \Rightarrow P(m_2)$.

证明.

① $m_1 \rightsquigarrow m_2$ 前提条件

② 存在 $m_1 \rightarrow m_2, \dots, m_{n-1} \rightarrow m_2. (n > 1)$

定义 1 应用到①

③ $P(m_1) \Rightarrow P(m_2), \dots, P(m_{n-1}) \Rightarrow P(m_2)$

引理 1 应用到②

④ $P(m_1) \Rightarrow P(m_2)$ 定义 5 应用到③

证毕.

引理 2.

假定 $C=(G, P_p, P_s, E, M)$ 是使用 HTSS $P(\langle S, \Rightarrow \rangle, P.stamp, P.tag)$ 的层次式系统, 其中 $m_1, m_2 \in M$.

$P(m_1) = P(m_2)$ iff $m_1 = m_2$.

证明.

由于 HTSS 的时标包含源站点的标识 ID, 所以源自不同站点的消息不可能含有相同的时标. 同时, HTSS 定义中的规则 1、2 保证了源自相同站点的不同消息不可能含有相同的时标. 所以, $P(m_1) = P(m_2)$ iff $m_1 = m_2$. 证毕.

定理 3.

假定 $C=(G, P_p, P_s, E, M)$ 是使用 HTSS $P(\langle S, \Rightarrow \rangle, P.stamp, P.tag)$ 的层次式系统, 其中 $m_1, m_2 \in M$. HTSS 是似然 TSS.

证明.

① $P(m_1) = P(m_2) \Leftrightarrow m_1 = m_2$ 引理 2

② 若 $m_1 \rightsquigarrow m_2$, 则 $P(m_1) \Rightarrow P(m_2)$ 定理 2

③ HTSS 是似然 TSS 定义 4 应用到①, ② 证毕.

因为 HTSS 是似然 TSS, 那么它将具有似然 TSS 的特点, 包括可以保证因果关系探测、具有可扩展性、可组合性等^[6].

5.4 消息分发协议

5.4.1 基本思想

我们利用“分而治之”的策略, 简化层次式系统的时标系统. 在子组中, 可以利用向量时钟, 防止因果异常. 同时, 代理进程互连的子组间的因果关系能够由 FIFO 的转换器保证. 这样, 在整个层次式系统中就不会发生因果异常. 这里采用的“分而治之”方法能够简化时标系统并加速进程间的并发运行. 同时层次式体系结构可能降低向量时钟的额外消耗, 这些消耗包括每个进程的空间需求和每条通信路径的带宽需求.

5.4.2 消息分发协议的内容

这里讲述基于层次式时标系统的消息分发过程.

消息格式

在层次式结构中, 普通进程可以发送消息到其它任何普通进程, 代理进程只是转发消息, 而不能作为消息的发送源或者消息的发送目的. 假定子组 G 中的普通进程 p 发送消息 m , 消息 m 包含下列各项:

- $m.tag$ = 时钟标签;
- $m.sp$ = 消息发送源进程的 ID;
- $m.dp$ = 消息发送目的进程的 ID;
- $m.data$ = 数据——消息内容.

在传输过程中只是 $m.tag$ 即时钟标签发生变化, 这就是说, 代理进程对消息进行转发的时候会根据新的子组中的时钟重新为消息附加 $m.tag$.

数据传输过程

子组中的每个普通进程包含下列时间变量:

$$Clock = VT \langle tm_1, tm_2, \dots, tm_n \rangle,$$

其中, n 表示子组中进程的个数(含代理进程), 如果 i 表示该进程, 则 tm_i 表示进程自身的进展, 如果 i 表示子组内的其它进程, 则 tm_i 表示该进程对进程 p_i 进展的知识, 这样, $Clock$ 则是该进程对于子组内逻辑全局时间的局部视图. 在所有进程中, 向量时钟中的每个元素初始化值为 0.

由于代理进程可以同时向本地子组和远程子组发送数据, 所以其具有向上和向下两组向量时钟. 可以将代理进程的向上、向下部分看作分别属于子组 G_1 和子组 G_2 . 因此, 代理进程包含下列时间变量:

$$Clock1 = VT \langle tm_1, tm_2, \dots, tm_m \rangle;$$

$$Clock2 = VT \langle tm_1, tm_2, \dots, tm_n \rangle.$$

当普通进程发送消息时, 子组 G 中的进程 p_i 按照下列过程发送消息 m :

$VT_i[i] := VT_i[i] + 1$, 进程 p 中的时钟值附加到 m 上, 即 $VT_m := VT_i$;

$m.sp, m.dp$ 附加到消息;

发出消息.

当普通进程接收消息时, 一个进程从子组 G 中的进程 p_j 接收到消息 m 时, 变量进行如下操作: $VT_i := \text{sup}(VT_i, VT_m)$, $VT_i[i] := VT_i[i] + 1$.

当代理进程接收到消息 m 时, 首先按照普通进程接收消息的过程接收消息, 推进消息接收子组的时钟; 然后, 推进消息转发子组的时钟, 将 $m.tag$ 用转发子组的时钟替换, 按照发送过程发送消息.

排序规则

消息按照下列规则在进程中排序因果性:

在子组 G 的一个进程 p 中, 一对消息 m_1 和 m_2 : m_1 先于 m_2 iff $m_1.tag < m_2.tag$.

5.5 属性分析

这一部分, 我们讨论层次式时标系统及其协议的属性.

属性 1. 没有因果异常.

假定 $C = (G, P_p, P_s, E, M)$ 是使用 HTSS $P(\langle S, \Rightarrow \rangle, P.stamp, P.tag)$ 的层次式系统, 其中 $m_1, m_2 \in M$. 如果 $m_1 \rightsquigarrow m_2$, 则 $P(m_1) \Rightarrow P(m_2)$.

定理 2 已经证明. 所以, 层次式系统的时标系统能够满足避免因果异常的要求, 其正确性是确

定无疑的.

属性 2. 性能提高.

除了没有因果异常的好处, HTSS 中包含独立的时间管理演算, 仅仅每个平面子组中的所有进程参加各自的时间推进. 这样, 采用“分而治之”的策略, 实现了层次式系统中进程间更加松散的同步, 因此能够得到性能上的提高. 对于该属性, 后面的实验也给出了相应的验证.

6 应用实例及性能分析

6.1 应用实例

在本节, 根据该层次式时标系统在空战仿真系统中的应用, 对其性能进行分析和评估. 该仿真应用实例描述了发生在边长为 400km 的方形区域海面上的空战. 仿真开始时, 双方飞机分别位于己方一侧四分之一方形的中心位置; 然后, 飞机根据飞机性能和控制信息执行巡航和战斗任务, 当一方飞机全部被击毁或者逃离该区域时为结束. 在该仿真应用中, 采用基于模块的方法, 将现实系统的物理结构直接映射为联盟群的逻辑结构. 图 3 展示了该实例的逻辑层次结构, 其中: 最高层联盟群中有三个普通联盟(战场环境盟员 EF、显示盟员 DisF、总控盟员 ConF)和红蓝双方两个联盟群, 其中每个联盟群由两个飞机联盟及本方的指挥盟员组成, 飞机联盟由雷达盟员、推进盟员、导航盟员、导弹盟员等组成.

每个联盟群内部利用向量时钟维护各个节点间的因果关系. 而整个仿真系统采用本文提供的消息分发协议(HTSS 方法), 能够保证每个盟员接收的所有消息间的因果关系, 例如, 飞机联盟的代理盟员中雷达搜索和导弹攻击消息之间的关系、显示盟员中导弹攻击和飞机爆炸消息之间的关系. 这样, 我们的时标系统能够解决逻辑时间需要解决的基本问题, 能够避免因网络延迟的不确定性和不可预测性造成的因果异常^[1].

6.2 性能分析

在层次式仿真实验中, 系统将 34 台 PC 机(操作系统为 Windows 2000)通过 100Mb 以太网按照层次分布在局域网中. 在单层联盟中, 仅有一个 RTI 服务器, 28 台 PC 机分布在同一 100Mb 以太网中, 为了确保盟员接收到所在实体的消息, 采用了数据分发管理技术(DDM)^[22]. 其中, 每个盟员就是位于一台主机中的模型, 分别执行其在空战仿真系统中的功能. 在对比实验中, 连续系统模型(如推进盟员

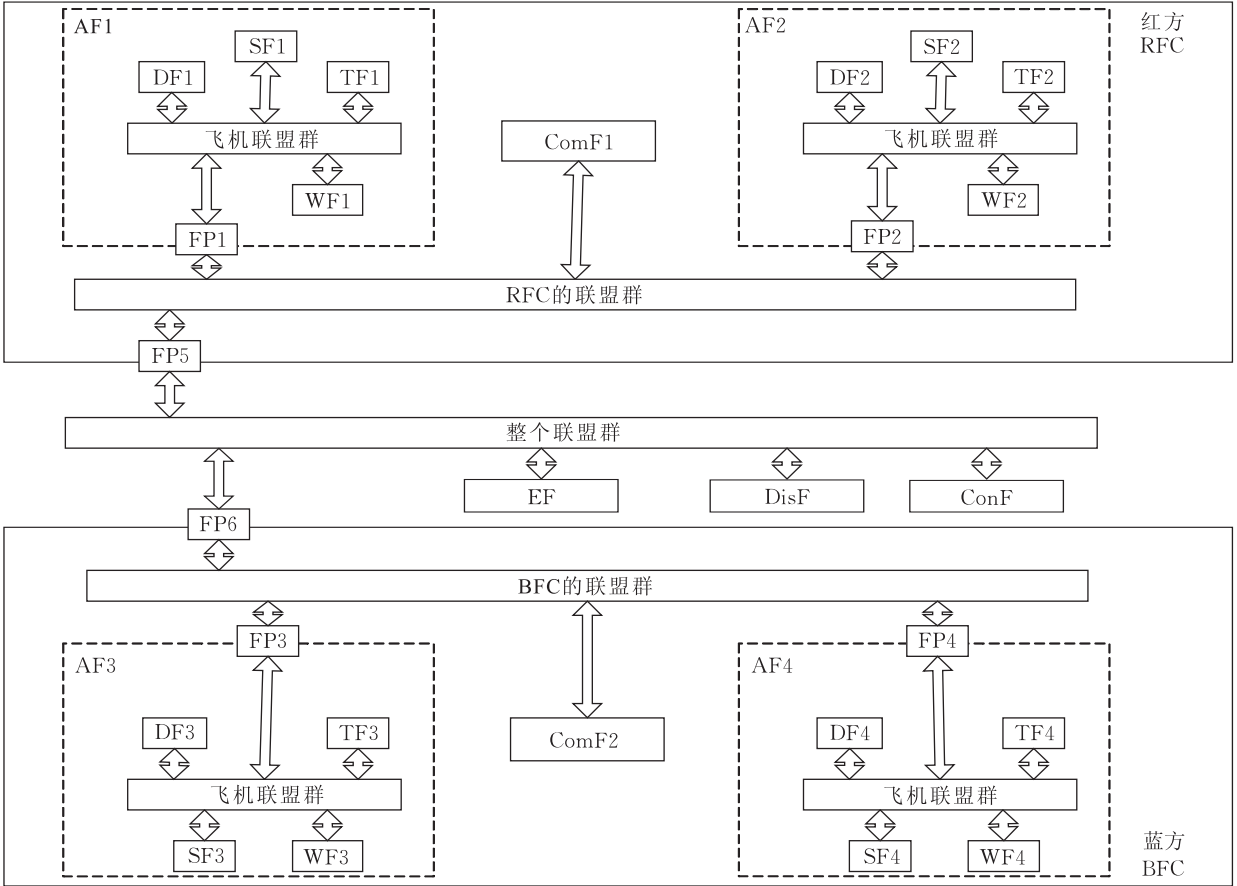


图 3 层次式系统仿真应用实例

中的模型)的仿真步长为 0.05s,保证连续模型数值计算的精确性和稳定性;同时采样周期为 1s,尽量降低连续系统消息发送的通信量。

在对比实验中,单层联盟中的仿真系统(在图 4 中用“联盟”表示)需要进行有效的 LBTS(Lower Bound on Time Stamp)演算^[2].在层次式联盟群的仿真中^[16],按照文献^[18]中的方法演算 LBTS(在图 4 中用“HFC”表示),而采用 HTSS 作为时标系统的层次式联盟群在图 4 中用“HTSS”表示.另外,不管采用何种消息序管理方式,该仿真系统的底层通信都需要采用 TCP/IP 协议,为支撑平台提供服务,保证仿真节点之间消息传递的 FIFO 关系.在对比实验的不同分组中,蓝方飞机性能确定,红方飞机性能在不同组的实验中改变.图 4 展示了三组实验

中不同消息序管理方式的平均仿真运行时间(墙上时钟)的对比,可以看到采用 HTSS 方法的层次式联盟群的性能有明显提高。

在单层联盟和层次式联盟群的仿真中,仿真系统需要进行有效的 LBTS 演算,使得盟员之间的时间推进相互影响,在层次式联盟群中,由于仿真系统具有的层次性,仅仅同一个联盟中的盟员相互影响,在不同联盟中的时间推进可以同时进行.而在采用 HTSS 方法的联盟群中,系统中的所有盟员可以完全独立地推进自己的时间,实现层次式系统中盟员之间更加松散的同步,在时间推进中的额外耗费被降低到最小,并且时间标签的长度为同一联盟中的盟员数目,减小了通信负载,因此能够得到性能上的提高。

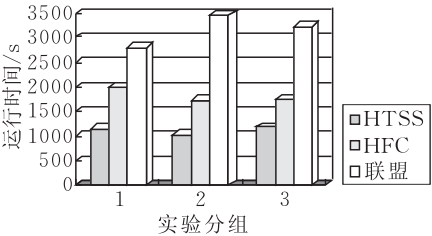


图 4 运行时间/s

7 结论与展望

本文提出了大规模分布式系统的层次式体系结构下的一种似然时标系统,该时标系统采用“分而治之”的策略,其特点是消息传输时的时间标签仅仅是系统时标的一部分.它不仅能够准确无误地排序因

果事件,又能解决效率和可扩展性问题.下一阶段,我们将进一步研究能够处理进程或者通信通道可能动态的生成或终止情况下的解决方案.

参 考 文 献

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

[2] Defense Modeling and Simulation Office. HLA Time Management; Design Document, 1996

[3] Fidge Colin J. Timestamps in message-passing systems that preserve the partial ordering//*Proceedings of the 11th Australian Computer Science Conference*. University of Queensland, Brisbane, Australia, 1988: 56-66

[4] Mattern Friedemann. Virtual time and alobal states of distributed systems//*Cosnard M et al eds. Proceedings of the International Workshop on Parallel and Distributed Algorithms*. Amsterdam: Elsevier Science Publishers, 1989: 215-226

[5] Charron-Bost Bernadette. Concerning the size of logical clocks in distributed systems. *Information Processing Letters*, 1991, 39: 11-16

[6] Torres-Rojas F, Ahamad M. Plausible TSSs: Constant size logical clocks for distributed systems//*Proceedings of the 10th International Workshop on Distributed Algorithms (WDAG 96)*. Bologna, Italy, 1996

[7] Mohamed El-Darieby, Dorina Petriu, Jerry Rolia. A hierarchical distributed protocol for MPLS path creation//*Proceedings of the 7th International Symposium on Computers and Communications (ISCC'02)*. Taormina, Italy, 2002: 920-926

[8] Taguchi Kojiro, Tomoya Enokido, Takizawa Makoto. Causality in hierarchical group communication//*Proceedings of the 23rd International Conference on Distributed Computing Systems Workshops (ICDCSW'03)*. Rhode, Island, USA, 2003: 568-573

[9] Kawanami Satoshi, Tomoya Enokido, Takizawa Makoto. A group communication protocol for scalable causal ordering//*Proceedings of the 18th International Conference on Advanced Information Networking and Application (AINA'04)*. Fukuoka, Japan, 2004: 296-302

[10] Khotimsky Denis A, Zhuklinets Igor A. Hierarchical vector clock; Scalable plausible clock for detecting causality in large distributed systems//*Proceedings of the 2nd International Conference on ATM (ICATM'99)*. University of Haute Al-

sace, Mulhouse, France, 1999: 156-163

[11] Hsiao Richard, Wang Sheng-De. Jelly: A dynamic hierarchical P2P overlay network with load balance and locality//*Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCSW'04)*. Hachioji Tokyo, Japan, 2004: 534-540

[12] Xu Zhi-Yong, Min Rui, Hu Yi-Ming. HIERAS: A DHT based hierarchical P2P routing algorithm//*Proceedings of the 2003 International Conference on Parallel Processing (ICPP'03)*. Kaohsiung, China, 2003: 187-196

[13] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) — Framework and Rules. The Institute of Electrical and Electronics Engineers, USA, 2000

[14] Myjak M D, Clark D, Lake T. RTI interoperability study group final report//*Proceedings of the Simulation Interoperability Workshop*. Orlando, Florida, USA, 1999: 99F-SIW-001

[15] Cramp A, Oudshoorn M J. Employing hierarchical federation communities in the virtual ship architecture. *Australian Computer Science Communications*, 2002, 24(1): 41-49

[16] Cai Wen-Tong, Turner Stephen J, Gan Boon Ping. Hierarchical federations; An architecture for information hiding//*Proceedings of the 15th Workshop on Parallel and Distributed Simulation*. Lake Arrowhead, Callifornia, USA, 2001: 647-674

[17] Cai Wen-Tong, Turner Stephen J, Lee Busung, Zhou Jun-Lan. An alternative time management mechanism for distributed simulations. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 2005, 15(2): 1-29

[18] Cramp A, Best J P, Oudshoorn M J. Time management in hierarchical federation communities//*Proceedings of the Fall Simulation Interoperability Workshop*. Orlando, Florida, 2002: 02F-SIW-031

[19] Snively Keith, Wilson Annette. Scalable reliable data dissemination for distributed simulations using hierarchical interconnect//*Proceedings of the Spring Simulation Interoperability Workshop*. Arlington (VA), 2004: 04S-SIW-073

[20] Knuth D M. *The Art of Computer Programming*. Ninth Printing. Addison Wesley, 2001

[21] Fujimoto R M. Distributed simulation systems//*Proceedings of the 2003 Winter Simulation Conference*. New Orleans, Louisiana, USA, 2003: 124-134

[22] Morse Katherine L, Steinman Jeffrey S. Data distribution management in the HLA//*Proceedings of the 1997 Spring Simulation Interoperability Workshop*. Orlando Florida, USA, 1997: 97S-SIW-052



CHEN Chun-Peng, born in 1974, Ph.D. candidate, engineer. His research interests include parallel and distributed simulation.

WANG Huai-Min, born in 1962, Ph.D. , professor, Ph.D. supervisor. His research interests include parallel simulation, parallel processing, modeling and software engineering.

YAO Yi-Ping, born in 1963, professor, Ph.D. supervisor. His research interests include simulation, modeling and software engineering.

Background

Currently, networks over which distributed systems running possess multilevel hierarchical structure, where certain sets of physical or logical nodes are grouped together according to physical, geographical, and/or administrative consideration. A hierarchical network is a traditional solution to the scaling problem. In the literature [Taguchi et al. 2003], a hierarchical group of processes are discussed to support hundreds of processes. In the literature [Kawanami et al. 2004], processes in local subgroups use physical and linear clocks while processes in a wide-area network adopt vector clock. However, this method is applicable in the specific case, and it can not resolve the expansibility because there are two levels in the hierarchical group.

A new time stamping system is proposed, called Hierarchical Vector Clock, which allows to detect causality between events in a large distributed system, while being ideally suited for the hierarchical structure of an underlying network and able to scale gracefully with the increasing number of distributed sites [Khotimsky and Zhuklinets 1999]. However, in this time stamping system, each process should maintain a

much longer timestamp which includes vector in the lowest peer group and vectors in the all higher peer group until in the peer group of the highest level and the whole timestamp should be attached to the messages even if the messages be transmitted in the local peer group. So, the communication overhead can be much larger with the advance of the level distance because a given process tends to interact frequently with only the processes in the same peer group. In this paper, the authors present a plausible time stamping system in the hierarchical architecture.

At the same time, in the current HLA application, Hierarchical Federation Community (HFC) and Time Management (TM) mechanism in the architecture was researched. However, it has not been proved to be correct, although the implementation of the TM services within the HFC architecture was discussed in the literature [Cramp et al. 2002]. And, the HLA TM services are not available to Hierarchical Architecture [Snively et al. 2004]. At first the protocol used in the HLA TM service is the main objective, but now it is extended to the distributed system.