分布式实时计算系统的一种调试方法

陈文宇** 桑 楠

(电子科技大学计算机学院 成都 610054)

【摘要】针对分布式实时系统的特殊性，对分布式实时语言DRTC++编写的分布式实时应用程序，利用DRTDebug的设计和调试原理，提出分布式实时计算中不确定性等问题的解决方法，得到分布式实时程序运行信息的收集模型。该模型简便、实用，可以直接用于分布式实时程序的调试中。

关 键 词 调试； 分布式实时； 不确定性； 事件； 信息收集

中图分类号 TP274+.2

随着分布式硬件技术的发展和分布式语言的使用，分布式程序的调试成为一个日益重要的问题。然而，传统的调试技术对分布式程序很难适用，其原因有以下几点：

1) 分布式程序的执行行为并非总是可再现的。在分布式系统中，任务在多个处理机结点上并发执行。处理机速度、通信延迟等诸多因素严重影响了各任务在系统中的执行次序，使每次执行均可能产生不同的结果，即Heisenberg不确定性[1]。

2) 交互的调试可干扰程序的正常执行。在顺序程序中，插入任何调试语句或断点均不会影响程序的执行行为，但在分布式系统的任务中插入断点和调试语句必然影响该任务的正常执行，从而导致整个系统中任务执行序列和结果的改变，即探针影响。

3) 分布式系统中全局状态的维护。传统的顺序调试技术的一个重要因素是利用程序执行状态的确定性，但对分布式系统而言，由于大多无精确的全局时钟，任务并发，无法同时冻结所有的执行任务，因而很难获得精确的全局状态[2]。

4) 收集调试所需数据。为了再现分布式系统上一次的运行情况，必须将分布式程序运行时的有关信息记录下来，信息收集是基于事件的，整个分布式程序的执行过程看成若干个并行执行的事件序列。在源程序中嵌入收集事件信息的原语，将有关进程创建／终止、进程间交互作用的信息收集在规定的监控缓冲区中；改源程序P成一个执行结果等同的分布式实时程序P′，在P′中增加了获取调试信息的监控进程，它直接从信息缓冲区中读信息，然后存放在相应的信息数据库中，供再现使用。分布式系统通常是既大又复杂，它运行时所产生的各类信息十分庞大。完全收集这些信息通常是不可能的，同时也是不必要的[3]。

5) 用户接口的合理化。在分布式系统中，除了各任务本身的执行状态之外，还应着重体现任务间的交互作用情况，即多线程控制和数据分布等因素，而用传统顺序调试技术的用户接口界面则无法体现。

另外，分布式实时性能调试也是传统调试工具很难解决的。本文针对分布式实时系统的特殊情况，提出调试器DRTDebug的设计思想。该调试器针对用基于对象的分布式实时语言DRTC++开发、在操作系统核心DRTK上运行的应用系统。

1 DRTDebug的设计

基干事件的调试策略的基本思想是将分布式系统的一个状态变化看成是一个事件发生，整个分布式程序的执行过程看成若干个并行执行的事件序列。分布式程序的调试则是调试每个监控事件的

2001年6月4日收稿

* 男 32岁 硕士 讲师

状态变化，从而掌握分布式程序的执行过程和执行结果。因此，DRTDebug的设计分成事件定义、收集运行信息、信息整理、运行再现和调试信息的体现及图形用户界面五个部分。事件定义用于设置调试系统所用的监控原语集，图形用户界面是为用户提供一个良好的使用环境，其他各部分则反映了整个调试过程。

DRTDebug的设计中，只监控导致分布式程序不确定性执行行为产生的全局系统状态变化，如进程创建／撤消、RPC等。相应的监控原语插入在调度程序、IPC调用等核心成分中，收集的事件信息仅包括再现执行无法恢复的部分，如进程执行次序、执行时间、接受的信息等。

1.1 调试过程

DRTDebus的调试过程分成两个阶段：1) 收集涉及分布式并发执行的事件信息并初步整理，保证调试时可确定性再现执行，此阶段内不做任何具体的调试工作，以减少调试对用户原程序执行的干扰；2) 进行程序的再现调试，根据用户的调试要求，对收集的事件信息进行加工处理，用于在模拟环境下确定性再现分布式程序的原始执行过程，并将有关调试信息告知用户，用户可在程序中设置断点、断言等，而其执行不再受分布式系统的影响，相当于顺序程序调试[4]。

1.2 不确定性问题的解决

对一道进程的执行，只要确定了进程切换时间、来自其他进程的信息及其时间、进程撤消时间等事件信息，就确定了整个进程的执行过程和结果。一旦各道进程的执行过程和结果都确定，随之而得的若干并发事件序列(依时间的偏序)就确定了整个分布式程序的执行行为。因此，不确定性问题解决的关键是事件信息是否完整反映分布式程序的执行。

在DRTDebug中，来自调度程序的事件信息反映了各结点上进程的执行次序(调度次序)，由IPC执行获取的事件信息包含了进程执行的人口数据、到达时间，因此确定了每个结点上进程的执行行为。结点间进程的执行次序，可以根据IPC和RPC调用确定大致的执行顺序和同步关系(调用先后)。这样，不必考虑全局时钟同步问题，就可使整个实时系统的执行行为确定化。

1.3 实时性能调试

根据所收集事件信息的时间特性，可以方便地判断进程或线程的执行是否满足实时性能要求。

同顺序程序的调试一样，分布式实时系统的再现调试可以分成单步执行、依断点执行、流览等多种方式。事件是再现执行的基本单元，执行过程就是事件发生的过程。

DRTDebug的调试界面是这个调试系统的控制面板。它一方面接收用户的各种预置条件，诸如断点、断言、监控对象等，另一方面将系统的运行情况和相关信息体现给用户，以便用户找出分布式实时系统中的错误。

告知用户的运行信息(事件信息)是事件发生前后系统状态变化情况，其中包括发生的时间、地点、导致哪些被监控对象发生变化等。如果多个事件同步发生，则相应的信息同时体现。此外，系统执行是否违背预置的时间约束等与时间相关的问题必须特别指出。为了方便用户理解多道进程的并发执行过程，使用时间进程图来体现多道进程的活跃周期，即依事件排序的进程执行顺序和同步情况。

2 信息收集

信息的收集是参考进程间通信的消息传递模式。通过监控通信可以获得在进程间传递的消息。首先定义几个源事件，如进程的创建、页失效、消息的传送和接受等。事件的信息包括在给定点(给定时间处)的进程状态、系统队列的状态、调度顺序以及进程间交换的信息。这些信息可以分为四种事件类型：消息接受、进程调度、主、外内存交换、页代替[5]。

定义一个标准的IPC源集，包括Send,Recv,Reply，如图1所示。

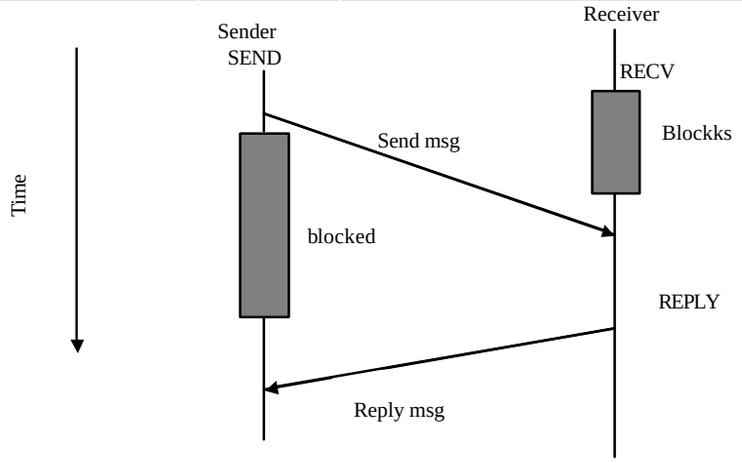

图1　IPC源集

信息的收集并不是直接在原语言上进行，要对源语言进行预处理、扩充。预处理器为各自动事件跟踪产生代码，并生成一个符号表。扩充的程序经过编译和连接成为可执行代码。预处理是通过在源代码中插入指令而完成的，指令包括trace、label、priority、break等，跟踪的事件由ETR记录，保存在数据库中，跟踪历史和符号表也保存在数据库中。

经过预处理后，可以着手对信息进行收集。进程间的通信是基于消息传递方式，消息的传递可能是同步(sender和receiver都同意交换信息时，完成消息交换)，也可能是按队列方式(直到receiver同意接收消息时，sender的消息才入队)，系统的状态包括IPS(进程内状态)和GSS(全局系统状态)。每个进程的IPS包含本地变量的值和本地程序的计数器，GSS包括系统中每个进程的状态变化，以指示进程是在运行、等待、失效或停止，还有运行的进程间相互作用表，指示发送或接受消息，或是在进行队列调度等[6,7]。

3　事件定义

事件是改变系统状态的活动。分布式程序的一个状态变化看成是一个事件发生，整个分布式程序的执行过程看成若干个并行执行的事件序列。分布式程序信息收集需要监控事件的状态变化，从而掌握分布式程序的执行过程和执行结果。

对一道进程的执行，只要确定了进程切换时间、来自其他进程的信息及其时间、进程撤消时间等事件信息，就确定了整个进程的执行过程和结果。一旦各道进程的执行过程和结果都确定，随之而得的若干并发事件序列(依时间的偏序)就确定了整个分布式程序的执行行为。

事件分为IPS和GSS事件。GSS事件包括例示/停止一个进程，发送/接受消息或队列调度。IPS事件包括变量的修改和可达标记。

所有GSS事件必须被跟踪，因为它们可用于分析和弄清并发程序的情况，为单个进程上的事件提供因果关系上的顺序。IPS事件是随使用者指令记录的，调试程序可产生代码以记录每个被跟踪变量的变化和控制到达被跟踪标号的时间。在程序运行时，任何GSS事件和被选样的IPS事件的发生均记录在调试程序的数据库中。IPS事件类型，如变量赋值和到达标号；GSS事件类型，如进程间通信(发送和接收)和进程示例或终止，这些事件类型基本上概括了程序中所发生事件的种类。每个记录的事件包括以下信息：

1) 事件类型(指语言预定义的事件类型)；
2) 进程标识(用于识别完成事件的进程的实例)；
3) 源代码的行数(位置)(用于确定对应事件在源代码的哪一行，即入口地址)；
4) 时间邮戳，给记录的事件以统一的全局顺序。

由于通信延迟，运行期间事件的报表送到监控器的顺序可能与实际发生顺序不一致，在消息传递的通信机制中，有因果关系的时间可以从进程间的事件推论而得到。

为保护事件的因果关系，引入ETR用于重新安排监控输出的事件流，使顺序适合因果关系。可能各个处理机的系统时间不一致，需要一个统一的全局时钟，根据各个处理机中事件的本地时间邮戳进行事件顺序的调整和整理，得出正确的事件时间顺序。

4 结 束 语

分布式实时系统开发环境是当前计算机学科前沿研究课题，其中分布式实时调试工具的研究尚属实验阶段。分布式实时程序运行的不确定性，为再现分布式实时程序的运行增加了很大的难度，在运行时进行信息的收集和整理，以保证能再现上一次程序运行的情况。该分布式实时程序运行信息的收集模型是一种简便、实用的方法。

参 考 文 献

1 Mcdowell C E. Debugging concurrent program. ACM computing Surveys, 1999, 21(4): 21~26

2 Baiardi E. Development of a concurrent language. ACM Sigplan Notices, 1986, 18(5):134~139

3 Leb1anc T J. Debugging paralld programs with insstant replay. IEEE Transon Computers, 1997, C-36:78~85

4 Goldszmidt G S. Interactive blockbox debugging for concurrent languages. Proc of the ACM SIGPLAN/SIGOPS Workshop on Parallel & Distributed Debugging, 2000:210~218

5 Tokuda H. A real time too1 set for the ARTS kerne1. In Proc of the 9th IEEE Real Time System Symposium. 1998: 138~143

6 Lei Hang, Xiong Guangze, Liu Jinde. The dependence of a distributed software module and a software reliability model. Journal of University of Electronic Science and Technology of China, 1995, 24(6):631~634[雷 航, 熊光泽, 刘锦德. 分布式软件模块的相关性与软件可靠性模型. 电子科技大学学报, 1995, 24(6): 631~634]

7 Li Tao, Li Hongbin. Actor: An implementation model of concurrent object-oriented programming. Journal of University of Electronic Science and Technology of China, 1996, 25(1):76~80[李 涛, 李鸿彬. Actor:实现并发面向对象程序设计的模型. 电子科技大学学报, 1996, 25(1):76~80]

Debugger of Real Time Computing System of Distributed System

Chen Wenyu Sang Nan

(Dept. of Computer Science, UEST of China Chengdu 610054)

Abstract For the speciality of distributed real-time system and the program that is written by distrbuted real-time language DRTC++ based on the on the distributed os kernel DRTK, this paper introduces the design and theorem of DRTDebug, exposes the approach of adjusting the indeterminacy and initiates the model of information collection of real-time execution of distributed system. The model is easy and useful, and can be used in debugging of distributed real-time program directly.

Key words debugging; real time of distributed sysytem; indeterminacy; event ; information collection