PVM 任务迁移协议的研究*)

The Study of PVM Task Migration Protocol

李 毅 周明天 虞厥邦

(电子科技大学570研究室 电子科技大学计算机学院 成都610054)

Abstract The work load could not be floated in PVM system, this fact limits seriously its functions of the balancing of work and dynamic scalabillity, thereby limits the improving of the parallel resource utilization and parallel running performance. Based on PVM system architecture and the scheme of message driving, this paper proposes a new protocol and its implementation of transparent and independent PVM task migration for supporting system load balancing in dynamic PVM system.

Keywords PVM, Load balance, Task migration, Scalability

1 引言

被作为MPI"事实标准"的PVM[1,2]是一个优秀的异构多机分布计算环境支持软件。出于最初系统设计的考虑,它具极好的静态可伸缩性,但是它不支持负载在系统内部浮动,严重影响了系统资源利用率和并行效率的提高。因此,为PVM增加任务迁移功能,使它成为可负载均衡的动态PVM系统是近年来的研究热点。

文[3,4]主要讨论了PVM任务调度问题,文[5]研究了SPMD计算模型的PVM任务调度,改进了PVM原来任务在主机集上平均分配的模式。但是,一个PVM任务一经启动,仍必须在原主机上一直运行到结束。文[6,7]修改了PVM任务间接通信的协议,每个pvmd用一张转发消息的"路由表"(oldtid-newtid映射表),将消息发送到目标任务的新位置,主pvmd有一张全局的路由表,用于从pvmd发送消息失败后,查找目标任务的新位置,这样可解决任务迁移期间发送给被迁移任务消息的转发问题,以及向已经迁出的任务发送消息失败的问题。文[8]沿用了文[6,7]的思想(包括路由表方法),但做了两点改变:(1)修改通信库函数,使PVM任务在进行socket I/O时,阻塞任务迁移信号,以保证通信信息的原子性;(2)每个主机(host)引入了专用于通信和(与任务迁移有关的)报文

*)本文的工作得到九五国防预研基金(DJ8.1.1)和四川省科技基金(JS05671)的资助。李 毅 博士生,主要研究方向为计算机操作系统,分布并行处理及分布对象技术。周明天 教授,博士生导师,主要研究方向为计算机网络,分布并行处理,分布对象技术和网络与信息安全。虞厥邦 教授,博士生导师,主要研究方向为电路CAD及EDA,并行分布系统及神经网络计算机。

4 Wan M, et al. A batch scheduler for the Intel Paragon with a non-contiguous nodes allocation algorithm. In: IPPS '96 Workshop on Job Scheduling Strategies for Parallel Processing. Hawaii, Apr, 1996. 29～39
5 Ali H, El-Rewini H. Task allocation in distributed systems. Journal of Combinatorial Mathematics and Combinatorial Compruting, 1993, 14. 15～32
6 Garey M, Johnson D. Computers and intractability. A guide to the theory of NP-completeness, W H, Freeman and Company, San Francisco, 1979
7 Stone H. Multiprocessor scheduling with the aid of network flow algorithms. IEEE Trans. Software Eng., 1977. 85～93
8 Casavant T, Kuhl J. A taxonomy of scheduling in general purpose distributed computing systems. IEEE Transaction on Software Engineering, 1988, 14(2)
9 Saletore V. A distributed and adaptive dynamic load balancing scheme for parallel processing of medium-grain tasks. In: Proceedings of DMCC. Portland, Oregon, 1990. 994～999
10 陈华平,计永昶,等. 分布式动态负载平衡调度的一个通用模型. 软件学报, 1998, 9(1): 25～29
11 陈华平,林洪,陈国良. 并行分布计算中的启发式任务调度. 计算机研究与发展, 1997, 34(suppl.): 74～78
12 Sarkar V. Partitioning and scheduling parallel programs for execution on multiprocessors. MIT Press, 1989
13 Kim S, Browne J. A general approach to mapping of parallel computation upon multiprocessor architectures. In: Proc. of the Intl. Conf on Parallel Processing. 1988. 1～8
14 Joosen W, Pollet J. The efficient management of task clusters in a dynamic load balancer. In: Proc. of the 1994 Intl. Conf. on Parallel Distributed Systems. Hsinchu, Taiwan, ROC, 1994. 19～21
15 Wang Fang, et al. A gang scheduling design for multiprogrammed parallel computing environments. In: IPPS '96 Workshop on Job Scheduling Strategies for Parallel Processing. Hawaii, Apr. 1996. 67～73
16 Hori A, Tezuka H, Ishikawa Y. Implementation of gang-scheduling on workstation cluster. In: IPPS '96 Workshop on Job Scheduling Strategies for Parallel Processing. Hawaii, Apr. 1996. 76～82

转发的 post 后台进程。引入 post 进程产生的问题是：(1)PVM 任务的间接消息均通过 post 与本地 pvmd 通信(再由 pvmd 转发)，增大了通信的时间和资源开销；(2)增加 post 后，必须完全重写 PVM 任务与 pvmd 的通信接口和通信协议，工作量巨大；(3)运行在每个主机上的 pvmd 后台进程的任务有二，PVM 主机及任务管理和 PVM 任务通信管理(消息转发)。一般而言前者工作量不大，并且消息转发操作时间很短，在 pvmd 上运行的通信协议操作函数完全能够胜任该工作。因此使用 post 进程解决(与任务迁移有关的)消息转发的必要性值得考虑。文[9]讨论了基于轻量级进程的 PVM 任务迁移，适用于共享内存的多处理机系统上的 SPMD 计算问题的优化。文[10]是动态 PVM 的代表作。主体上它基本同文[6,7]，pvmd 也使用路由表处理转发消息和迁移期间途中报文的递交问题。文[10]的贡献是：(1)完善了使用 checkpointing 进行 PVM 任务迁移的机制，尤其是任务重启；(2)解决了任务迁移时用于 PVM 任务直接通信的 TCP 连接的断连问题；(3)使用分布的 hometab(路由表)代替了集中的在主 host 上的全局路由表，避免主 pvmd 成为瓶颈，并且提高了效率(串行的"名字服务"处理转化为分布并行处理)。

尽管文[6,7,8,10]动态 PVM 的实现各不相同，但它们具二个基本共同点：(1)使用路由表转发消息和正确递交在迁移期间发给迁移任务的消息；(2)使用 checkpointing 机制，即核外技术进行进程迁移，以实现动态 PVM 系统的可移植目标。

用类似于蜂窝移动电话建立呼叫连接时寻找目的话机位置算法的路由表(以下路由表项即为 $T_1 \rightarrow Hy$，即 T_1任务在 Hy 主机上运行)消息转发协议，实现动态 PVM 有如下缺陷：

(1)通信延时和低层网络负载增大　①当消息源任务发送消息给 T_1任务时，其所在主机的路由表项 $T_1 \rightarrow Hx$ 与 T_1任务的 hometab(最初生成 T_1的主机 pvmd 的有关路由表项，对于文[7,8]是主 pvmd 的全局路由表项)路由表项 $T_1 \rightarrow Hy$(即 T_1已从 Hx 迁移到 Hy)不同时，会引起无效的消息发送。随后的(协议)管理消息会通知消息源主机 $T_1 \rightarrow Hy$ 路由。更遭的情况是：由于存在通信延迟，当消息发向 Hy 时，T_1已迁移到 Hz…，会出现二次、三次…无效发送及路由查询。②任务首次向 T_2任务发送消息时，主机路由表中尚无关于 T_2的路由，pvmd 会查询 T_2的 hometab 关于其路由信息，然后再发送消息给 T_2。路由表依赖 LRU 策略经常替换表项。因此，类似的情况也经常发生。③由于存在无效消息发送，因此为了区分这种情况，不仅仅下层用于 pvmd 之间通信的可靠 UDP 需要确认(对用户透明)，而且 pvmd 的高层协议中也需要确认，以确定到达目的主机的报文是否被目的任务收到。

(2)任务 tid 空间维护工作量加大　任务 T_1的第一个 tid 在 T_1的生存期不能被再分配使用，即使是 T_1已迁移到其它主机上(有新的任务号)，否则会造成消息误投。当 T 终止时，释放当前任务号和原 tid 号，会增加通信开销。

使用核外技术(不修改 OS 核心)的 checkpointing 机制实现进程迁移存在的问题是：(1)无法做到对应用程序的完全透明[10]。(2)尽管改进后的 checkpointing 可以不做磁盘 core 文件的 I/O 进行进程迁移，提高了效率，但是它却很难用于支持 elf 格式文件和 VM 的操作系统进程的迁移。这主要是因为最初在 Condor 中，它是针对 a.out 格式文件和非 VM 内存管理系统的 UNIX 进程迁移设计的。文[6,7,8,10]的 PVM 任务迁移方法没有将进程迁移部分从任务迁移协议中分离出来，致使其不能用于迁移支持 elf 文件 OS 上的 PVM 任务。这是因为尽管用户程序同 pvmlib 是静态链接的，但是其它库函数调用(包括系统功能调用的 C 接口函数)均优先同 elf 动态库链接，失败后再同 .a 文件中的库函数链接(限于篇幅，这方面的问题另文讨论)。

PVM 任务迁移分四部分(阶段)[6,10]：

(1)迁移事件发生　由动态 PVM 系统的任务调度进程根据负载分布情况和用户(机器主人)对主机的要求等条件作出迁移决策，并通知有关对象(在本文协议中，通知主 pvmd)。

(2)任务迁移准备

(3)进程迁移　将 PVM 任务进程的图像从源主机取出，安装到目的主机。

(4)任务重启

本文只研究(2)、(4)部分的协议及技术细节。它们与下层操作系统及进程迁移和上层 PVM 用户程序基本无关，仅与中间层(pvmd 和 pvmlib)有关。

本文工作的目标是：(1)任务迁移是透明的[10]。对 PVM 应用程序透明，用户不察觉 PVM 任务是否迁移；同时，系统内其它任务的正确性不受任务迁移的影响。(2)迁移是高效的。任务迁移期应尽量地短；与迁移有关的通信量不宜过大；且迁移完成后，系统的通信开销不应因发生过任务迁移而增大。(3)任务迁移与进程迁移方法无关，即实现后者从前者中完全分离。

本文基于 PVM 体系结构和消息驱动机制，通过对 PVM 系统作了少量向后兼容的扩充(增加了部分与任务迁移相关的协议操作函数)，并令同迁移关联的任务在迁移期间暂停活动且适时修改与迁移任务地址(任务号)有关的信息来实现 PVM 任务迁移。

本文介绍的PVM任务迁移协议的特征是：在任务迁移过程中，PVM系统仍能正常运行，只有相关主机pvmd、相关任务和对应于该迁移的三个迁移管理任务(mmig、smig和dmig)进行有关的任务迁移活动；在该过程中相关任务处于暂时打开的迁移态；相关主机pvmd仍具正常功能；其它任务无论在PVM系统的何处仍正常运行；并且，本迁移协议允许PVM系统同时存在多个无关任务(相关任务集不重合)的并行迁移。

2 与PVM任务迁移有关的问题

2.1 消息处理

主要解决任务迁移期间途中报文的正确递交与迁移完成后消息的正确收发。

1)消息、报文的完整性　文[8]为了实现迁移开始时刻PVM任务收发的信息以消息为单位是完整的，当任务通过socket口读写消息前，先屏蔽任务迁移信号，读写完成后开启信号屏蔽位。但该方法存在这样的问题：(1)当迁移信号发送到迁移任务时，恰好它正在接收消息，必须被迫延时一不定长时间(与剩余接收消息长度有关)才能响应迁移信号，使任务迁移的实时性遭到破坏，不能很好应用于动态负载均衡目的。(2)任务收到迁移信号时，并未通知合作任务暂停向自己发送消息。如果此时多个分布并行的合作任务，向迁移任务发送大量消息，这些大量需要重新投递的消息会积累在迁移任务的源主机。造成的直接结果是产生长时间的消息重新递交延时和高的网络通信负载。因此本文对此作了改进。UNIX信号是进程从核心态返回到用户态时响应并在用户态下运行信号处理程序的。当主pvmd产生的某个任务迁移控制任务(mmig)向迁移任务的所有合作任务(有亲缘关系的相关任务)发出迁移信号(通过相关任务所在的相关主机pvmd转发)，相关任务(包括迁移任务)从核心态返回到用户态响应迁移信号时，相关任务和迁移任务一定完成了一次socket I/O。在迁移信号处理程序中(所有)相关任务暂停全部发送活动且延时一段时间(2*最大报文传送时间)，均只接收消息(mrout(0,0,0,0))，接收到的消息缓存在任务的pvmrxlist中，然后进入暂停活动的迁移态。这样便能确保：(1)没有与迁移任务有关的通信数据在与OS socket口相关联的缓冲中；(2)根据pvmlib提供给任务的底层报文发送函数mxfer，能保证相关任务和迁移任务接收的、pvmd存储准备转发的报文以frag或pkt为单位是完整的。

2)途中报文重定向　当迁移任务及其合作任务进入迁移态后，有关协议函数在迁移管理任务mmig发送的消息驱动下，遍历相关主机pvmd的主机表和本地任务表的收发与转发报文队列，将frag、pkt及mesg的源地址和目的地址中被迁移任务号oldtid修改为新任务号newtid。如果是发向被迁移任务的报文，将会被转发到新目的地；如果是被迁移任务发出的报文，便会标识为(迁移后)新任务是消息源，便于后继通信。这一点利用的正是无连接存储转发报文可在中途任一结点处重定向的特点。

3)任务间直接通信的TCP连接的断连　被迁移任务及其相关任务完成了当前socket I/O后(从核心返回)，先后响应收到的任务迁移命令信号，此时TCP连接两端的任务均没有读写端口，随后的暂停写和延时读便能保证TCP连接中无残存通信数据。因此随后的断连不会丢失数据。任务迁移完成后，再进行任务通信时，底层函数透明地使用新任务号建连通信。如果建连失败则使用经过pvmd转发的可靠UDP的间接方式通信。

4)任务迁移后的消息正确收发　当PVM任务(oldtid)迁移后，会出现两类问题：(1)合作任务如何将原来准备发向oldtid的消息正确发向newtid；(2)迁移到目的地的PVM任务(newtid)如何在向它的合作伙伴发消息时使用正确的源地址newtid，而不是oldtid。任务号变量tid是在编写PVM源程序时写入源代码的，任务迁移时进程使用的是编译好的二进制信息。因此，不能直接修改。本文使用指针间接操作的方法来处理此问题。用户使用全局变量ntid和tid[]来标识本任务的合作任务数及合作任务号，程序中用指针来引用tid[]中的任务号。pvmlib中扩充的协议函数使用ntid和tid[]来查找oldtid，并将其修改为newtid。这样便圆满解决了前面提出的两个问题。

5)消息汇合信息的处理　pvmd中使用wait context消息链(peer队列)来管理消息汇合，每发出一个与该消息事件有关的消息，登记一个表项结点并链入peer。每接收到一个应答消息，处理过后便从peer中删除与之对应的结点。当收到应答消息时，peer长度为1时，便向发起该消息事件的消息源发总应答，这是基于消息驱动的分布式系统中1对n事件同步处理的一种方法。为使系统能正常运行，保证任务迁移的透明性，在任务迁移期间相关pvmd需遍历所有wait context表，相关任务需遍历waitlist表，进行oldtid到newtid的转换。

2.2 迁移活动管理

1)迁移活动同步　迁移主控任务(mmig)及主pvmd与多个相关主机pvmd、迁移主控任务与源迁移控制任务(smig)和目的迁移控制任务(dmig)以及相关pvmd和它管理的(在该主机上的)多个相关任务均存在1对n的消息事件同步关系。本文协议采用原PVM系统中的wait context机制，并结合阻塞接收消息和

UNIX 信号与 pause()调用等技术来解决迁移活动中的同步的问题。

2)迁移活动范围限定　首先给出相关任务和相关主机概念：迁移任务的相关任务是与迁移任务属同一用户作业的 PVM 任务，并且定义迁移任务的目的任务也是它的相关任务。迁移任务的相关主机是它的相关任务所驻留的主机，并且定义 PVM 的主 host 也是它的相关主机。

在本文的 PVM 任务迁移协议中，给定的任务迁移会产生相应于该任务迁移的主控迁移管理任务 mmig、源迁移管理任务 smig(在源主机)和目的迁移管理任务 dmig(在目的主机)；同时，该特定迁移活动仅限于与它对应的上述三个迁移管理任务和相关任务及相关主机 pvmd 范围之内。这样设计的目的是：(1)为了任务迁移活动的独立，即多个不相关的任务迁移活动可在 PVM 系统中并存；(2)为了任务迁移活动的透明，即 PVM 系统不受任务迁移活动影响，其它任务仍可正常运行；(3)使与指定任务迁移有关的三个任务迁移管理进程成为 PVM 任务的目的，是为了它们能利用 PVM 消息驱动机制和通信机构，便于实现任务迁移管理。为了便于查找相关任务，主 host 维护一个 family[]邻接表，记录整个 PVM 系统的用户作业的相关任务。修改 spawn 和 pvmd 的相关协议，使其生成任务时，主 pvmd 在 family[]中登记该任务，任务迁移后需要修改 family[]中相应项。

3)迁移活动识别　为了识别多个并存的迁移活动和 PVM 任务(迁移管理任务)tid 与其进程 pid 的映射，对于每个迁移任务它的主 host、源 host 和目的 host 的 pvmd 在其维护的有关迁移活动的映射表中有唯一的项(四元组，它是迁移活动在该主机上的唯一标识)：(oldtid，newtid，迁移管理任务的 tid，迁移管理任务的 pid)，当 pvmd 收到有关任务迁移的报文后，协议函数查找映射表，便能确定相应迁移活动和下一个有关的操作对象。

4)迁移控制消息　为基于 PVM 体系结构和消息驱动机制开发任务迁移协议，使其与原系统兼容，本文协议使用了与原系统语法上一致的协议控制消息类标识，供协议状态机识别处理。当迁移管理任务与本地 pvmd 通信，使用 TM_MIGXXX 类消息。当主 host pvmd 与相关 host pvmd 通信，使用 DM_MIGXXX 类消息。迁移管理任务之间通过 pvmd(也使用 TM_MIGXXX 消息标识)或 TCP 连接通信。

2.3　任务重启

迁移任务(oldtid)的目的任务号 newtid 在任务迁移活动开始之前就确定了。这主要是因为：(1)系统任务调度进程出于负载均衡的目的需要指定任务迁到何处及由于 PVM 任务 tid 的结构(主机号＋主机内的任务号)，需事先确定 newtid，(2)为了进行有关任务迁移的消息处理以及迁移活动管理也应该事先确定 newtid，但是，newtid 先于被迁任务迁移到目的主机成为 PVM 任务之前被指定而出现的问题是：迁移到目的地的任务含有大量原任务信息(waitlist，ttlist，pvmrxlist…)，尽管这些信息已经过一致性的消息处理，可是新任务还没有在目的主机 pvmd 中登记并与它建立 TCP 连接(topvmd)，也即 PVM 任务化。但是这项工作不能使用 PVM 系统函数 pvmbeatask()来完成，否则在 PVM 任务化的初始化过程中会清空原任务信息。本文协议用下面方法解决该问题：被迁移任务做好迁移准备时，将任务重启函数(restart())作为信号处理函数安装到信号处理函数表中后，调用 pause()，等待任务进程图像被迁移。当任务进程迁移到目的主机后，信号触发进程调用任务重启函数，任务重启函数代表迁移任务与目的主机 pvmd 建立 TCP 连接，并让 pvmd 登记自己后获得 pvmmytid(系统分配新任务号)；向 pvmd 发消息，通知它将本任务的任务号 pvmmytid 改为 newtid，即让 pvmd 用新增加的协议函数把 pvmmytid 在 pvmd locatask 表中有关表项的内容复制到 newtid 表项后，将前者删除掉。

3　PVM 任务迁移

本文 PVM 任务迁移协议是基于 PVM 系统的分布并行环境、在三个迁移管理任务控制下、相关主机 pvmd 及相关任务的协议函数在任务迁移控制消息的驱动下协同工作而实现的。下面介绍任务迁移的主要过程。

(1)动态 PVM 系统的任务调度进程向主 host pvmd 发送任务迁移请求(DM_MIGRQ)，请求将系统中用户 PVM 任务 oldtid 迁移到目的主机并成为 newtid。主 pvmd 登记本次活动，并生成相应于本任务迁移活动的迁移管理任务 mmig。

(2)mmig 从主 pvmd 获取迁移任务 oldtid 的全部相关任务 tid[]，并从中提出相关主机 id 后，通知相关主机 pvmd：命令所有相关任务进行迁移准备，即延时一段时间只接收不发送报文，并将相关任务和(相关)pvmd 自己的报文消息进行重定向、修改 waitlist 等，完成2.1小节中叙述的消息处理工作后，进入迁移暂停态。

oldtid 所在主机 pvmd(源主机)，首先登记本次迁移活动，然后生成源迁移控制任务 smig，最后令 oldtid 安装任务重启函数 restart 为信号处理函数后使其暂停(paust)。

newtid 所在主机 pvmd(目的主机)，也先登记本

次迁移活动，再生成目的迁移控制任务 dmig。

(3)dmig 首先建立一 TCP socket 端口，通过自己的信号处理函数将该端口号在适当的(与 smig 同步的)时刻发向 smig(由 pvmd 转发)，smig 获取 dmig 的 TCP 端口后，与 dmig 建立 TCP 连接。然后通过 TCP 连接将取出的 oldtid 进程的图像发送到 dmig，并由其安装，smig 和 dmig 完成任务后相继退出。

(4)mmig 由反馈消息得知迁移工作基本完毕后，向全体相关 pvmd 发消息，通知相关任务继续运行；源 pvmd 强行终止 oldtid 任务；目的 pvmd 向 newtid 进程发出任务重启触发信号；newtid 重启后，从信号处理程序(restart)返回迁移断点继续运行。

4 PVM 任务迁移的实现

(1)在 ddpro.c 中增加 dm_migxxx 类任务迁移消息协议处理函数。在 netswitch 的入口表中增加消息 tag 为 DM_MIGXXX 的 dm_migxxx 函数入口，以处理 pvmd 之间的有关任务迁移的控制消息。

(2)在 tdpro.c 中增加 tm_migxxx 类任务迁移消息协议处理函数。在 loclswitch 的入口表中增加消息 tag 为 TM_MIGXXX 的 tm_migxxx 函数入口，以处理控制迁移任务发来的有关任务迁移的控制消息。

(3)在 lpvm.c 中增加迁移信号处理函数的 stub migtidstart，该函数主要负责相关任务消息处理及迁移活动的同步。在 pvmbeatadk()函数中的消息处理函数初始化部分增加迁移活动映射表 migtaskslist 的初始化，并安装 migtidstart 函数到 mhandle 表中(restart 安装到 lpvm.c 中)，用于进程图像迁移后的任务重启。

(4)修改 spawn 函数，并增加相应的 pvmd 协议处理函数，使 spawn 一个新任务时，主 pvmd 一致性地登记该相关任务组的 family[]邻接表项。在主 pvmd 的 pvmd.c 的 main()函数中初始化 family。

(5)相关任务组中有任务处于 spawn 过程，则该相关任务组不能进行任务迁移活动。负载均衡任务在发出 DM_MIGREQ(oldtid, newtid)之前应作相应的检测。

5 需进一步研究的问题

我们在四台不同配置的 PC 机组成的 PVM 系统上对该方法做了实验，成功地进行了 PVM 任务迁移，经过一段时间的运行实践，证明本文方法是可行的，且达到了预期目的。但本文只研究了在给定条件下一般的用户 PVM 任务的迁移问题，进一步开展的工作有：

(1)组机构和 mailbox 在有关任务迁移时的相应处理技术。

(2)用户任务一般的 fork 子进程(尚未 PVM 任务化)在父进程迁移时的处理。

(3)在任务迁移过程中磁盘文件 I/O 的一般解决方法。

(4)协议的优化，以进一步提高迁移效率和实时性。

参 考 文 献

1 Sunderam V S. PVM: A framework for parallel distributed computing. Concurreney, Practice and Experience, 1990, 2(4): 315～339

2 Geist A, et al. PVM: Parallel Virtual Machine-A Users' Guide and Tutorial for Networked Parallel Computing. Massachusetts: MIT Press, 1994

3 鞠九滨，魏晓辉，徐高潮，尹玉. DPVM：支持任务迁移和排队的 PVM. 计算机学报，1997，20(10)：872～877

4 鞠九滨，王勇. 调度 PVM 任务. 计算机学报，1997，20(5)：470～474

5 张博，王纪成，周兴社. SPMD 模式 PVM 任务均衡分配研究与实现. 计算机研究与发展，1997，34(8)：588～593

6 Dikken L. DynamicPVM: Task migration in PVM. [Technical Report]. Shell Research ICS/155. 1. Nov. 1993

7 Dikken L, et al. DynamicPVM: Dynamic load balancing on Parallel systems. High-Performance Computing and Networking, 1994, 797: 273～277

8 鞠九滨，魏晓辉，郭雷. 利用检查点机制在 PVM 中实现进程迁移. 软件学报，1996，7(3)：175～179

9 Konuru R, et al. A user-level process package for PVM In. 1994 Scalable High-Performance Computing Conference. IEEE Computer Society Press, May 1994. 48～55

10 Casas J, et al. MPVM: A migration transparent version of PVM: [Technical Report, CSE-95-002]. Dept. of Computer Science and Engineering, Oregon Graduate Institute of Science & Technology, February 1995

(上接第20页)

术、度量和统计技术和波动分析技术等，对 C++/Java 语言程序进行分析和理解，显示直观，界面友好，用户可以在程序和图表间自由切换，实现了交互式浏览，达到了方便软件人员进行面向对象程序设计的目的，也是我们在探索软件工具和环境设计和开发的新思路、新方法上的新的尝试。

参 考 文 献

1 李必信，郑国梁，等. 软件理解研究与进展. 计算机研究与发展，1999，36(8)：897～906

2 李必信，郑国梁，等. 一种分析和理解程序的方法—程序切片. 计算机研究与发展，2000，37(3)

3 Weiser M. Program Slicing: Formal, Psychological and Practical Investigations of an Automatic Program Abstraction Method: [Ph. D thesis]. The University of Michigan, Ann Arbor, Michigan, 1979

4 Krishnaswamy A. Program Slicing: An Application of Object-oriented Program Dependency Graph: [Technical Report TR94-108]. Department of Computer Science, Clemson University, 1994