MPI(消息传输界面)的标准化与我国的对策

Standard and Chineses Strategy of MPI(Message-Passing Interface)

苏运霖

(广州暨南大学计算机科学系　广州510632)

Abstract The motivation of posing the message-passing interface and thereby the development of the standardization of it are surveyed in the paper. Then the goals and the contents are presented. Finally the paper discusses how would China deal with the issue.

Keywords Message, Passing, Interface

一、消息传输界面的提出

众所周知,计算机的网络化成为当今计算机科学技术发展的不可阻挡的潮流,计算机的网络化形成了并行计算和分布式计算的环境。这些计算环境,旨在大大提高计算机的工作效力和工作能力,使得原来在单机环境下无法实现或实现起来效力很低的问题,在新的环境下可以有效地解决。在并行计算和分布式计算当中,有着普遍存在的一项通用任务,这就是消息在这样一种环境中的传输,以实现在参与计算的各进程之间的全局性的同步,或者激发在每个进程中某个事件的执行,或者实现各进程掌握有其部分输入的函数的计算(即此计算是分布在上述环境的各部分同步进行的)。这些任务通常总是通过按照某种预先确定的同网络拓扑有关的方案的消息传输实现的。因此我们可以说,消息传输是实现并行计算或分布式计算的基础,是在并行计算机上,特别是在具有分布存储可伸缩的并行计算机(SPC)和在工作站的网络(NOW)上,广泛应用的一个程序设计的范例。消息传输的算法,成为吸引广泛注意的一个研究课题,并被称作波动算法。迄今为止,已被提出的波动算法,即消息传输算法,有环算法、树算法、回声算法、投票算法、阶段算法、Finn 算法、遍历算法、等等。这里仅列举较为基本的几个,实际上还有很多。消息传输,自然都涉及到发送和接收两个方面。可以想像,既然有两端而这两部分可以是异构的计算机系统,就如同使用不同语言的两个国家一样。因此消息传输就存在一个双方都可以理解,或经过一定的处理之后,可以理解的界面。消息传输界面的问题,正是在这样一个背景下提出的。

为了开发重要的并行应用,许多可伸缩的并行计算机系统中,都开发了消息传输数据库。这是实现高水平的并行计算的必由之路。只有实现了这一点,才有真正意义下的大规模并行计算可言。但由于这些程序库之间可以想像得到的差异(在不同部门,不同的机器配置,使用不同的程序语言,等等),使程序代码的可移植性受到阻碍。

与此同时,好多公开领域的系统已经揭示,可以有效地和可移植地实现消息传输系统。因此许多美国和欧洲在这领域的权威专家都认识到,现在是制定对广泛用户有用并可在广泛的计算机上有效地实现消息传输程序库的核心代码标准的适当时刻了。通过对于这个标准的语法和语义的制定,广大的计算机用户便可在共同的基础上进行沟通。因此它的意义,就如同网络通讯中的协议,或者程序设计中的程序设计语言文本一样。但是它却比它们更进一步,因为它是同最前沿的计算机科学技术——可伸缩的并行计算机和分布式计算机相关联的。以下我们将把消息传输界面简称为 MPI。

二、MPI 标准的制定

正是在上述共识的基础上起始于1992年,来自美国和欧洲的40个团体的80余人,代表着并行计算机的厂家,企业界用户,企业和国家研究实验室和大学,组成了 MPI 讨论会。在从1992年11月到1993年9月这一期间,他们每六个星期进行一次,对 MPI 标准的方方面面的问题进行广泛而深入的讨论。最终于1993年11月在"超级计算'93会议"上,提出了头一个 MPI 标准的草稿。这些专家们寻求的目标,是把标准化置于一个更形式化的立足点上,也就是说,不是选择现存的 MPI 版本作为基础,把它修定成为标准,而是吸收现存的 MPI 系统的最有吸引力的特征,来制定一个新的标准。为了集中各方面的见解,后来这些会议吸收来自高性能计算界的有兴趣的成员都来参加。形式上也不局限于亲自参加这些会议,也可以通过电子邮件来参与讨论。经过一段时间交给公众讨论之后,于1994年5月,正式形成了 MPI 的版本1.0。

从1995年春天开始,新的 MPI 讨论会又开始举行,以便对 MPI 的实现和使用所取得经验以及所发现的问题再度进行讨论。新的 MPI 讨论会既有老的成员,又有新的成员组成,并且和以前的讨论会一样,有一个对公众开放的过程,以期集思广益,充分吸收各方面的智慧和见解。到1997年7月,新的 MPI 讨论会完成了 MPI-2的定义,它包括对 MPI-1程序设计模型的重要扩充,对于 MPI-1标准的一些改进,以及对以前未明确说明的 MPI 说明书部分作了澄清。

三、MPI 标准的作用

自1994年5月 MPI 1.0标准正式提出至今,MPI 已被广泛接受和使用,它现在已经是向所有并行计算机提供的一个标准。一些实现免费提供给异构工作站网络和单个的并行机使用 MPI,也已开发出大量的科学和工程软件,包括许多免费的程序库在内。由于这样,MPI 实现了它的目标之一,即增加了并行计算的可信度。MPI 标准有以下四个作用,这也就是它的四个重要设计目标:

1. 实现在不同机器上的可移植性。MPI 所期望的,是实现诸如 Fortran, Pascal 以及 C 语言这样的程序设计语言所提供的可移植性。这意味着,相同的消息传输的源代码,只要有 MPI 程序库可利用,便可以在各种各样的机器上执行,只须对于每个系统作一些必要的调整以最好地利用各自的特

征。尽管消息传输被想象作为是在分布式内存的并行计算机的范筹内使用的，但相同的代码在共享内存的并行计算机上也能很好地运行。MPI 可以在一个工作站的网络上运行，也可作为一组进程在一个工作站上运行。意识到贯穿于广泛的计算机上存在着 MPI 的有效实现，就为程序代码的开发和调试提供了高度的灵活性，也对生产性运行提供了选择平台的灵活性。

当然，可移植性不能以牺牲性能为代价。如果是这样，则 MPI 的标准不可能被广泛接受。例如，Fortran 比汇编程序得到更广泛的使用，因为高性能的 Fortran 的编译程序几乎总是可以得到的，而它相对不可移植的汇编语言的选择，总是产生出可接受的性能。这里的一个关键之点在于，已经把 MPI 仔细地设计成能够有效地加以实现的系统，因而在形形色色的平台上，它都达到了高的性能。

2. 实现在异构系统上的兼容性。MPI 标准提供了透明地在异构系统上运行的能力。因此，一个 MPI 的实现能够跨越着样的异构初理器的聚合，同时提供掩盖了许多体系结构上差异的一个虚拟计算，用户根本无需担心代码是在类似结构的处理器之间还是不同结构的处理器之间传送消息。MPI 的实现将自动地做任何必要的数据转换以及使用正确的通讯协议。然而，MPI 并不专注于单个的和同构的系统上的实现，而且也不强迫不同的实现相互间可以操作。因此希望在一个异构系统上运行的用户必须使用专门为支持异构性而设计的 MPI 系统。

3. 实现并行处理中的可伸缩性。通过其若干设计特征，MPI 允许或支持可伸缩性。例如，一个应用可以建立一小组进程。进而，这些进程允许集体的通讯操作并把范围局限于所涉及的进程而已。采用的另一个技术是提供这样一个功能，这一功能不要求和进程的个数成比例的局部计算。例如，二维笛卡尔拓扑可以再分成一维的行或列，而不必明确列出诸进程。

4. 当然，MPI 始终不渝地追求的是在具有不同特征的机器上的有效实现。例如，MPI 总是仔细地避免明确说明一个操作做什么。结果，无论是缓冲消息是在发送者一边，或者是在接收者一边，或者全然没有缓冲的系统，MPI 都能很容易地实现。其实现可以利用各种机器的通讯子系统的一些特定特征。比如说，在具有智能通讯协议处理器的机器上，许多的消息传输协议都可以下载到这个协处理器上。而在其他系统上，大多数通讯代码都由主处理器来执行。另一个例子是使用 MPI 不透明的对象，通过掩盖 MPI 的特定对象是如何表示的细节，每个实现就可以根据各自的情况自由选择最好的方案。实现有效性的另一个设计选择是避免不必要的工作。MPI 被仔细地设计以避免对于每个消息都要求大量额外的信息，或者对于消息的标题需要复杂的加密和解密过程。MPI 也避免在关键程序上的额外计算或检测，因为这将使性能退化。鼓励对以前计算的重用，也是减少不必要工作的一个方法。MPI 通过持续一致的通讯要求以及对于通讯器的属性实现高速缓冲这样一些构造，提供了这个能力。MPI 的设计避免对数据做过多的复制和缓冲，使数据能从用户方直接传送到线路上，又可从线路上直接传送到接收方的存储器中。

MPI 鼓励通讯和计算的重迭，以便利用智能的通讯机构（代理），并且掩盖通讯的潜伏时间。这是通过使用无封锁的通讯调用实现的。这种调用把通讯的初始化和它的结束分离开来。最后，MPI 的价值在于，它定义了一个已知的消息传输实现的最低特性。这就使程序员避免了担心出现某些问题的烦恼。比如，MPI 保证消息的基础性传输过程是可靠的，因此用户就不必检查消息是否正确地被接收了。

四、MPI 标准包括的内容

MPI 标准版本1.0包括的内容主要有：

1. *点对点的通讯*　MPI 的基本通讯机制是在一对进程之间数据的传输。一方发送，一方接收。我们因而把它称作点对点。MPI 的几乎所有构造都是围绕点对点的操作来构建的。有许多不同类型的点对点操作，MPI 因而提供了一组发送和接收功能，它们允许通过一个相关联的 tag 来对各种类型（type）的数据进行通讯。这种消息的分类型对于异构的支持来说是必要的。

2. *集体性操作*　所谓集体性操作指 MPI 同时在各个进程之间进行通讯的能力，包括广播、散开，和搜集数据的能力。MPI 也能实现归约和扫描的操作，包括由用户自定义的一些操作。这些操作还包括有各种变形，它们控制某一个进程或某一些进程接收结果。全局性的归约操作包括求和，求极大值，求极小植，或者用户自定义函数等。

3. *进程的分组*　在某些应用中，希望对进程进行分组以便允许不同的进程组实现独立的工作。例如，我们可能需要有这样一个应用，其中三分之二的进程基于已经处理的数据来预测天气，而其余三分之一的进程开始处理新的数据。这样就可以使这个应用定期地完成天气预报的工作。然而，如果没有可供处理的新数据，我们可能希望所有进程都投入到天气预报的工作中，或者我们也可能把进程分组以防止在不同程序库中的消息的冲突，即使不同组的进程仅可访问在其管辖范围内的库。

4. *通讯领域*　在 MPI 中任何点对点的通讯都是在通讯领域中出现的。这样一个通讯领域由带有一致的值的一组通讯器表示，每个参与进程都有一个值。如果这个领域是组内通讯，则所有通讯器都是内部通讯器，因而全都有相同的组的属性。一个组内通讯领域由一组通讯器来确定，它们的链接形成一个完全图，而且这些链接有一致的下标。另外一种是组之间的通讯领域，对于它们，MPI 又以不同的方法来描述和表征。

5. *进程拓扑*　一个拓扑是可以赋给一个组内通讯器的额外的任选的属性，但拓扑不能赋给组间通讯器。一个拓扑可以对于一个通讯器内的一组进程提供方便的命令机制。此外，还可以帮助运行系统把进程映射到硬件上。在 MPI 的一个进程组是 n 个进程的一个聚合，组中每个进程被指定0至 n－1的一个号码。但在并行应用中，进程的线性编号不能适当地反映进程的逻辑通讯模式（这通常是由基础性的问题的几何和所用算法确定的）。诸进程经常被安排成如二维或三维栅格的拓扑模式。更一般地，逻辑进程安排是由一个图来描述的。

6. *环境管理和查询*　这涉及 MPI 和操作系统的交互以及实现的特定信息，包括如何起动和停止一个 MPI 程序或使之流产，以及 MPI 如何和操作系统的其他特性包括信号进行交互，也包括如何处理由 MPI 返回的错误信息等。

7. *形象界面*　这涉及到 MPI 用于描述 MPI 诸程序的性能的机制。为了满足 MPI 的形象界面的要求，其性能实现必须满足下列要求：

i）提供一个机制，通过它，用一个名字替换便可访问 MPI 定义的所有函数。因此，通常以前缀“MPI”开始的所有函数，应当也能通过前缀“PMPI”来访问；

ii）确保未被代替的 MPI 函数仍可被链接到一个可执行

映像而不会引起名字冲突；

iii）如果不同的语言是在彼此的顶上被分层的，则应建立MPI界面的不同语言约束的实现文档以便形象的开发者知道形象界面是否必须对于每个约束来给出实现，或者是否仅须对最低层的程序来给出实现；

iv）确保不同语言约束的实现在哪里通过分层方法实现，则包装函数（例如Fortran的约束是一组包装函数，它们调用C的实现）要和这个库分开；

v）MPI库中提供一个不可操作的程序MPI-PCONTROL。

8. *对于Fortran和C的约束* MPI-2在这个基础上加上：①动态进程管理；②输入/输出；③ 一边的操作；④对C++的约束。

五、我国的对策

人类社会生活的各个方面，包括科学活动在内，都需要有秩序地进行。各种法规，规范，标准也就为此而制定的。然而，在很多情况下，这类东西又是强者强加于弱者的。在计算机科学技术领域内，我们大体上已经经历了四次标准的制定：(1)各种程序设计语言的标准。我们由于种种原因，无缘参与这项工作；(2)操作系统标准的制定，这主要是由国外大计算机企业或研究单位完成的，我们也无资格参加；(3)通讯协议的制定。在这方面，我们或许参加了一点工作，但由于我们的客观条件的限制，我们也没有多少发言权；(4)软件工程的规范。这次，MPI标准虽然涉及的仅是并行计算或分布式计算的问题，但由于在未来发展中，这是一个极为重要的领域，我们不能再置身事外，必须认真研究我们的对策。据我个人看来，对策不外有三：

一是完全不理会国外的什么标准，我们按自己的需要自搞一套。然而，这条路走不通，我们不可能不同国外发生联系，一旦我们的并行机要和国外的并行机实现通讯，就不可避免地要用已有的MPI标准。就是我们所购买的国外的并行机（这类机器的数量在我国肯定将不断增加），也还是要使用已有的MPI标准。

二是完全用国外的标准。这条路最省事，其实我们早也就司空见惯了这样一种情况。我们这么多年来，不就是什么都用人家的标准吗？然而，随着我国的并行机的发展以及在它上面应用的扩大，而且当我们的通讯是用中文信息进行时，就势必会发现，国外的标准未必完全符合我们的要求。因此，这就决定了完全采用国外的标准不可取。

三是原则上采用国外的标准，或者说参考国外的标准，但要根据我们中文信息处理以及将来中文计算的需要，进行适当的修改。这里有以下的问题：MPI-1和MPI-2所包括的这些内容，哪些完全符合我们的需要？凡属符合我们的需要的，我们自然无需作任何改动；而哪些，对于我们的中文信息通讯或计算，是不大适当的？对此自然应作适当修改；还有哪些根本不适合我们的需要？我们应当以我们真正需要的东西来代替它（不换也可，但就是不使用它）。

事实上，MPI讨论会在这个问题上，也采取了明智的态度，即有许多特征，他们考虑暂不列入标准中。这是因为感觉到对于这些特征还缺乏足够的经验，没有明确的判断；有些特征在他们看来，其应用有限；还有一点，就是担心增加了它们，使程序员实现它们的负担增加了，但是对于用户价值又不很大，当我们来考虑自己的标准时，对于这些特征也都要充分考察。

总之，MPI标准的制定，为我们开拓我国的并行计算和分布式计算，将打下良好的基础。这也是我们参与到世界性的标准制定的一次大好机会，我们应该站在面向未来的高度，搞出一个与国际接轨又符合我国需要的MPI标准。

参 考 文 献

1 苏运霖. 分布式系统与分布式算法. 暨南大学出版社，1995
2 Tel G. Distributed Algorithms. 剑桥大学出版社，1994
3 Sair M, et al. MPI—The Complete Reference, MIT Press, 1998. 1, 2
4 Gropp W, et al. Using MPI, Portable Parallel Programming with the Message-Passing Interface. MIT Press, 1994

（上接第43页）

建立一个简单的模型，该模型描述了在规定期限内完成目标与当前查询及任务特征是如何相关联的。然后，把该模型与新增一个信息Agent的假想局势做比较，如果评估结果认为有必要新增一个信息Agent，则该Agent暂停信息服务（向中介发送停止提供服务的信息），进行自我复制。

总结和展望 信息Agent是多Agent系统的重要组成部分，本文主要介绍了信息Agent的体系结构和五种可重用行为。合理的体系结构能够支持许多复杂的自治Agent行为。行为重用意味着信息Agent的行为与特定的领域知识无关。我们可以不需要编程或只须写几行代码就可以迅速创建一个新的信息Agent，甚至，当我们对一个信息Agent进行性能改善或增加新行为时，所有其它领域的信息Agent都可以从中获益。本文讨论了五种可重用行为：定义行为、信息查询、信息监控、自我调整以及自我复制。

对于信息Agent，仍有很多问题需要进一步研究，如多媒体信息检索与服务，分布式运行的信息Agent合作问题的研究[7]，以及信息Agent学习算法[8]等。信息Agent相对于传统的搜索引擎而言，其性能有了很大的改善。信息Agent在没有用户的指导的情况下，能够对外部请求进行思考并有序地完成请求任务。为使信息Agent更加适应网上应用，为用户提供更加高质量的服务，有必要对信息Agent进行更深入的研究。

参 考 文 献

1 Takahashi K. Intelligent pages: collecting shop and service information with software agents. Applied Artificial Intelligence, 1997 (11): 489～499
2 Goldman C V, et al. Musag: An agent that learns what you mean. 1997
3 Moukas A. Amalthaea: Information discovery and filtering using a multi-agent evolving ecosystem. Applied Artificial Intelligence, 1997(11): 437～457
4 Payne T R, Edwards P. Interface agents that learn: An investigation of learning issues in a mail agent interface. Applied Artificial Intelligence, 1997(11): 1～32
5 Decker K, Pannu A, et al. Designing behaviors for information agents. In the Robotics Institute, Carnegie-Mellon University 1996
6 Gudivada V N. Information retrieval on the World Wide Web. IEEE Internet Computing, 1997, (5): 58～68
7 Oates T, Prasad M V N, et al. A distributed problem solving approach to cooperative information gathering. In AAAI Spring Symposium on Information Gathering in Distributed Environments, Stanford University, March 1995
8 Decker K S, Lesser V R. Designing a family of coordination algorithms. In: Proc. of the First Intl. Conf. on Multi-Agent systems. AAAI Press, San Francisco, June 1995