Mont-Blanc原型系统：实现高性能计算的新方法

● 刘先祥　王广益
江南计算技术研究所 无锡 214083

摘要：
为了实现百亿亿次级性能，当前的高性能计算系统不断发展，以克服日益严峻的结构性挑战。Mont-Blanc原型系统是第一个基于面向嵌入式和移动领域的商用片上系统（SoC）、存储器及网络接口卡（NIC）而构建的高性能计算系统。本文重点描述Mont-Blanc原型的体系结构，并评估比较该系统的计算节点、互连网络及整个系统的性能与能效。

关键词：高性能计算，商用部件，片上系统，嵌入式/移动处理器，能效，可扩展性

1. 引言

高性能计算（HPC）是推动科学、工业、医学和教育进一步发展的重要支柱之一，受应用需求牵引，HPC系统快速发展。从性能角度看，采用商品化技术的HPC系统不断取得里程碑式进步。ASCI Red和Roadrunner分别突破了1Tflops（每秒一万亿次浮点运算）和1Pflops（每秒一千万亿次浮点运算）性能大关。这些系统的成功充分展现了商品化技术可以很好地推动下一代HPC系统结构的发展。

在广阔的市场驱动下，商用部件比专用部件发展更快。当前，嵌入式/移动处理器已开始商品化，处理器技术发展迅速，性能不断攀升，基本具备了HPC系统所需的能力。同样，由于庞大的消费类需求推动，嵌入式市场加速发展，创新设计不断涌现。例如，低功耗双数据速率（LPDDR）内存技术最先被引入移动领域，最近又被推荐作为能量均衡服务器的存储解决方案。

Mont-Blanc项目旨在提供一种基于当前商用技术（移动芯片等）的HPC系统替代解决方案。Mont-Blanc原型系统是第一个基于面向嵌入式和移动领域的商用片上系统（SoC）、存储器，以及网络接口卡（NIC）构建的HPC系统。该项目设计制造了一台由三星Exynos 5250 SoC构成的1080节点高性能计算集群。Mont-Blanc设立了如下目标：一是基于当前的移动商用技术，设计并部署足够大的HPC系统；二是开发并优化软件栈，使其能适用于HPC；三是移植并优化一系列HPC应用，使其能在该原型系统上运行。本文重点对Mont-Blanc原型系统的体系结构进行详细描述，并通过与MareNostrum III超级计算机对比，评估该原型系统的性能和功耗。

2. Mont-Blanc原型系统简介

本节简要介绍Mont-Blanc原型系统的结构（参见图1）组成，重点描述其每一功能模块主要特点。

图1 Mont-Blanc系统视图

2.1 Mont-Blanc计算节点

Mont-Blanc计算节点采用了“模块上服务器（Server-on-Module）”结构。图2描述了Mont-Blanc节点卡（三星子板，简称SDB）及其组件。每块SDB围绕一个三星Exynos 5250 SoC制造，集成2个时钟频率为1.7GHz、共享1MB片上L2高速缓存的ARM Cortex-A15 CPU和1个533MHz移动四核ARM Mali-T604 GPU。SoC连接到板上4GBLPDDR3-1600RAM，

CPU和GPU通过两个32位存储器通道共享该RAM，峰值存储带宽为12.8GB/s。

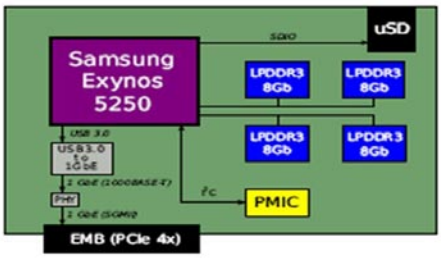


图2 Mont-Blanc节点框图

节点互连由ASIXAS88179 USB 3.0-to-1 Gb以太网网桥和一个以太网PHY提供。一个外部16GB μSD卡提供引导装载程序、操作系统映像和本地暂存器。节点通过一个专有总线，使用一个PCI-e 4x形状因子边缘连接器（以太网母板连接器）连接至刀片。

2.2 Mont-Blanc刀片

图3描述了Mont-Blanc刀片的结构。刀片含有15个Mont-Blanc节点，节点间通过一个板上1GbE交换网互连。该交换网可提供2个10GbE上行链路。此外，以太网母板（EMB）可提供管理服务、SDB功耗监测及刀片温度监测功能。EMB组件通过安装在前端的风扇进行空气冷却。

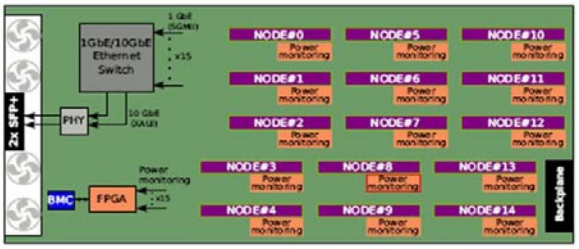


图3 Mont-Blanc刀片框图

2.3 Mont-Blanc系统

整个Mont-Blanc原型系统配置两个标准42U-19英寸机柜。每个机柜最多含有4个7U Bullx机仓，每个机仓可集成9个Mont-Blanc刀片。此外，系统还配有2个2U 10GbE思科Nexus 5596UP柜顶（TOR）交换机、1个用于原型管理的1U 1GbE交换机和2个2U存储节点。

（1）系统互连

Mont-Blanc原型系统采用两个独立的网络：GbE管理网络和10 GbE消息传递接口（MPI）网络。本文仅介绍MPI网络（参见图4）。刀片内交换由一个1GbE交换网（提供两条10GbE上行链路）实现，刀片间交换则由速率为1.92 Tbps的TOR交换机实现。机柜间直接通过四条40 GbE链路相连。

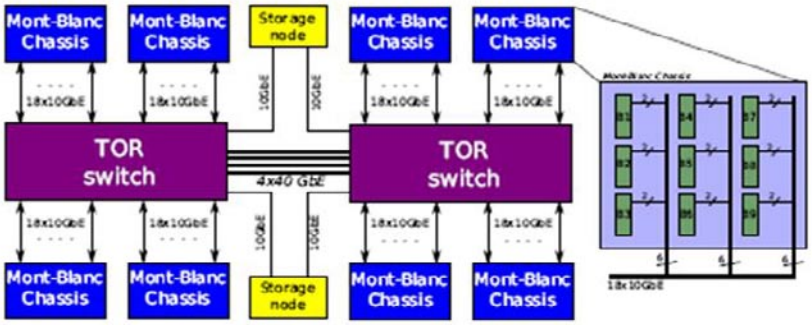


图4 Mont-Blanc系统互连框图

（2）系统存储

Lustre并行文件系统建立在一个基于x86-64架构的超微公司存储桥接坞（Storage Bridge Bay）上，总容量9.6TB，读/写带宽2-3.5GB/s（取决盘区）。该存储系统通过4个10GbE链路与柜顶交换机相连。

（3）系统冷却

Mont-Blanc原型系统的计算节点主要通过安装在系统顶部的散热器进行被动冷却，而刀片则主要通过温度控制回路中的前端变速风扇进行主动冷却。

2.4 Mont-Blanc软件栈

通过努力，Mont-Blanc项目使HPC软件栈得以成功移植到ARM架构上，使之可像其他HPC集群一样好用。Mont-Blanc原型系统的节点运行Ubuntu 14.04.1 Linux（基于定制Linaro Kernel 3.11.0版），允许用户在ARM Mali-T604 GPU上进行OpenCL编程。Mont-Blanc原型系统的软件列表参见表1。

表1 Mont-Blanc原型系统软件栈列表

编译器	*GNU、JDK、Mercurium*
科学库	*ATLAS、LAPACK、SCALAPACK、FFTW、BOOST、clBLAS、clFFT、PETSc、HDF5*
性能分析	*EXTRAE、Paraver、Scalasca*
调式工具	*Allinea、DDT*
运行时库	*Nanos++、OpenCL、OpenMPI、MPICH3*

集群管理	*SLURM、Nagios、Ganglia*
存储	*LustreFS*
操作系统	*Ubuntu*

Mont-Blanc原型系统采用由Mercurium编译器和Nanos++运行时系统提供的OmpSs编程模型。OmpSs是一种基于任务的编程模型，允许运行时系统精心安排任务的无序执行，可选择性地将任务卸载到GPU上，或是当GPU忙碌时，在CPU上运行任务。但移植至OmpSs上的应用可同时使用CPU和GPU，以保证动态地适应负载不平衡的状况。

2.5 功耗监控设施

Mont-Blanc原型系统提供了一种用于高频测量单个计算节点功耗，并可扩展至整个原型规模的基础设施。系统在通向每个SDB的电源轨道中设置了一个数字电流电压表。每个EMB上的FPGA都可通过I2C总线访问各SDB中的电源传感器，并将每1120毫秒的平均数值存储在一个FIFO缓存器中。EMB上的板级管理控制器（BMC）与FPGA通信时，先从FIFO缓存器中收集功率数据样本，然后将它们连同读取时间戳一并存入DDR2存储器。用户执行一系列定制的智能平台管理接口（IPMI）命令，通过管理以太网从BMC获取数据。

为了向应用开发人员提供其应用的功率轨迹，Mont-Blanc原型系统提供了一个定制的系统监控工具，它可自动完成功率测量与采集过程。该工具通过采用MQTT和Apache Cassandra实现了简易性和可扩展性。其中，MQTT用于轻便消息传输，Apache Cassandra则为可扩展分布式数据库，主要用于存储所获得的功率和其他监测数据。一系列命令行工具和一个特殊API可为用户提供访问原始监测数据或其他数据源信息的能力。

2.6 性能小结

表2显示了Mont-Blanc原型系统的性能指标。两个Cortex-A15核可提供27.2Gflops单精度（SP）和6.8Gflops双精度（DP）峰值性能。之所以存在性能差异，是因为代号为NEON的SIMD单元只支持SP浮点（FP）运算，从而导致DPFP指令必须在标量单元中执行。

表2 Mont-Blanc原型性能指标

计算节点		
	CPU	*GPU*
计算单元	*2 × ARM Cortex-A15*	*1 × ARM Mali-T604*
频率	*1.7 GHz*	*533 MHz*
峰值性能（单精度）	*27.2 GFLOPS*	*72.5 GFLOPS*
峰值性能（双精度）	*6.8 GFLOPS*	*21.3 GFLOPS*
存储器（共享）	*4 GB LPDDR3-800*	
刀片= 15 × 节点		
峰值性能（单精度）	*408 GFLOPS*	*1.08 TFLOPS*
峰值性能（双精度）	*102 GFLOPS*	*319.5 GFLOPS*
存储器	*60 GB*	
机仓 = 9 × 刀片		
峰值性能（单精度）	*3.67 TFLOPS*	*9.79 TFLOPS*
峰值性能（双精度）	*0.92 TFLOPS*	*2.88 TFLOPS*
存储器	*540 GB*	
系统=8× 机仓		
峰值性能（单精度）	*29.38 TFLOPS*	*78.3 TFLOPS*
总性能（单精度）	*107.7 TFLOPS*	
峰值性能（双精度）	*7.34 TFLOPS*	*23 TFLOPS*
总性能（双精度）	*30.3 TFLOPS*	
存储器	*4.32 TB*	

3. 计算节点评估

本节将Mont-Blanc原型系统中使用的三星Exynos 5250 SoC与MareNostrum III超级计算机中使用的2.6GHz八核Intel Xeon E5-2670服务器CPU进行比较。MareNostrum节点采用的是一种使用DDR3-1600存储器DIMMs的双插槽结构。详细比较情况参见表3。在评估比较中，两者皆运行Mont-Blanc基准测试程序。

（1）处理器核评估

图5显示了Mont-Blanc系统与MareNostrum超级计算机的核对核性能比较。本比较采用单线程运行，这样可以不受并行调度和同步化的影响。

表3 Mont-Blanc与MareNostrum III的节点性能比较

指标	Mont-Blanc	MareNostrum III
频率[GHz]	1.7	2.6
插口数	1	2
峰值性能[GFLOPS]	CPU：6.8；GPU：21.3	CPU：332.8；GPU：/
存储带宽[GB/s]	12.8	51.2
网络带宽[Gb/s]	1	40
插口间带宽[GB/s]	/	32

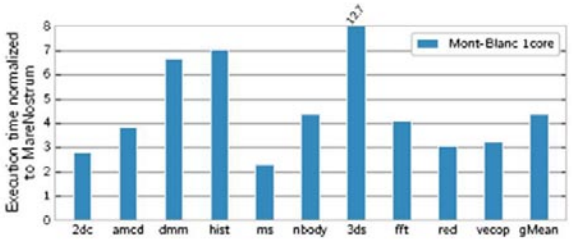
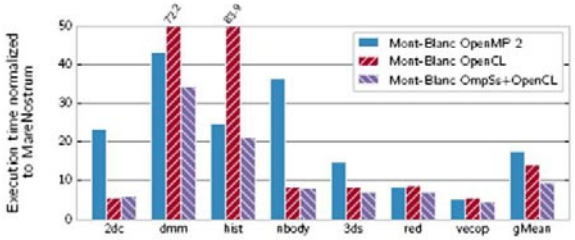


图5 Mont-Blanc基准测试：核对核性能比较

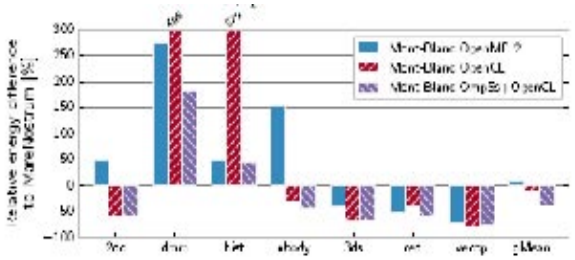
Mont-Blanc原型系统的每个核在所有基准测试程序中都相对慢了2.2~12.7倍。即Cortex-A15核逊于Intel Sandy Bridge，这是因为缺少SIMD DPFP扩展指令，且每个插口的存储带宽低（12.8对51.2GB/s），存储子系统资源有限。平均而言，Mont-Blanc原型系统的处理器核比MareNostrum III慢了4.3倍。

（2）节点性能评估

图6显示了Mont-Blanc原型系统的节点与MareNostrum III超级计算机节点的性能和能耗比较。考虑到Mont-Blanc SoC及其软件栈的特性，这里评估了用OpenMP的同构CPU计算、用OpenCL的异构CPU+GPU计算和用OmpSs的异构计算，三个不同的计算方案。



（a）性能比较



（b）能耗比较

图6 Mont-Blanc基准测试：节点对节点比较

比较仅用CPU的计算，双核Mont-Blanc节点平均比16核MareNostrum节点慢18倍；当用OpenCL把所有计算任务卸载到GPU上时，Mont-Blanc节点比MareNostrum节点慢14倍；最后，用OmpSs有效地将计算卸载到GPU和CPU上，差距则显著缩小到9倍。

在节能方面，只用CPU时，Mont-Blanc节点比MareNostrum节点多消耗7%能量。而随着性能差距逐步缩小，Mont-Blanc节点变得更节能：从只用GPU时少消耗10%能量，到GPU和CPU都用时少消耗40%能量。

比较结果显示，采用嵌入式GPU时，Mont-Blanc系统明显比MareNostrum III之类的同构集群更节能。但是，为了匹配性能，Mont-Blanc系统需要把应用扩展到10~15倍以上的节点，因此互连网的性能尤为关键。

（3）节点功耗分析

能量通常有两个维度：功耗和时间。执行时间取决于应用在基础架构上如何执行，功耗则取决于应用给计算资源、处理器物理实现及SoC电源管理施加了多大压力，Mont-Blanc原型系统中的功耗监控设施可帮助用户合理平衡这两个因素。通过比较不同计算的功耗（CPU、GPU或CPU+GPU），用户可估算补偿功耗差异所需的加速度，并在最佳能效点运行系统。图7显示了Mont-Blanc节点针对不同计算资源执行3D-stencil基准程序获得的高抽样率功耗曲线。不同计算资源是指：单CPU、双CPU、GPU和GPU+1CPU。

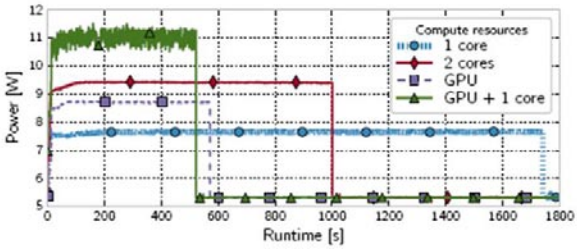


图7 不同硬件资源执行3D-stencil基准程序的功耗曲线

节点闲置时功耗是5.3W，这包括所有组件的静态功耗。在单个和2个CPU核上运行时，平均功耗分别是7.8W和9.5W，这包括SoC、存储子系统和网络接口的功耗。

在使用GPU和GPU+1CPU时，节点功耗分别是8.8W和11W。只在GPU上运行时，其中一个核依旧活跃，其作为辅助线程，同步启动GPU内核，以阻塞方式运行直到结束；在GPU+1CPU上运行OmpSs时，因其中一个核要作为GPU辅助器，另一个核则实际运行计算所需的工作线程，因此增加了额外功耗。

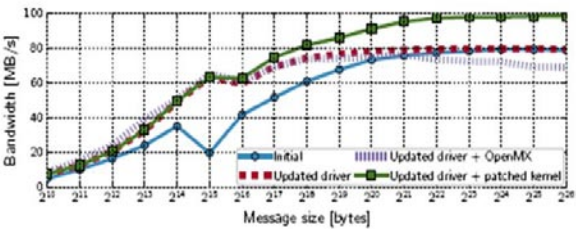
测试结果表明，OmpSs因增加了一个CPU核参与GPU计算而导致的额外功耗胜过性能提升，进而

导致在3D stencil基准测试中更高的能耗方案。针对其它测试基准来分析，节点功耗因不同工作负荷而异。当用2个CPU核执行时，最大功耗是14W，而用GPU+1CPU核执行时，最大功耗是13.7W。

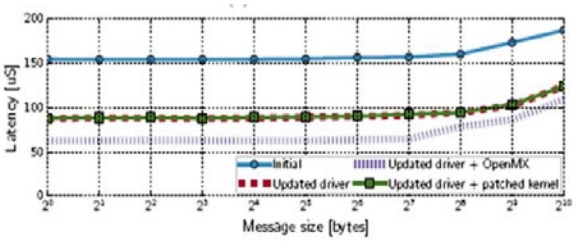
4. 互连网络评估

本节主要量化评估Mont-Blanc系统互连网络的延迟和带宽。由于Mont-Blanc系统的互连是采用一种有损耗以太网技术实现的，因此对每层加以适当调谐是极其重要的。以下讨论对该互连网不同部分所做的改进，它们自然会影响到总体互连的性能。图8展示了使用Intel MPI PingPong基准程序所测得的Mont-Blanc系统的互连带宽和延迟。每个图里有四条曲线，分别对应节点网络接口的增强改进。

测量得到Mont-Blanc原型系统的MPI吞吐量为80MB/s，延迟为156 μs，这些结果是通过内置在Linux内核的NIC驱动程序获得的。通过升级使用USB-to-GbE网桥制造商提供的一种专有版驱动程序，可显著改善其吞吐量和延迟：对于64KB以下的消息，吞吐量提高了3.4倍；对于零长度（zero-sized）消息，延迟只有88 μs。但是对于较大消息，吞吐量则没有太大变化。



（a）互连网络带宽



（b）互连网络延迟

图8 Mont-Blanc原型系统节点间互连网络带宽与延迟

此外，研究人员通过移植Linux内核补丁，提高了64KB以上消息的吞吐量，最高达100MB/s。为便于比较，在一个集成有1GbE NIC的服务器类x86_86系统上，运行相同基准程序，测得的吞吐量为112.5MB/s，延迟为46.5 μs。由此可见，Mont-Blanc系统实现了89%的潜在带宽，但延迟仍高出1.9倍。

绝大部分乒乓延迟归因于ARM Cortex-A15 CPU上运行的TCP/IP协议栈。为此，研究人员部署了Open-MX协议栈以取代TCP/IP，结果使该轻量级协议将小消息的延迟减至65 μs，并增加了32KB以下消息的带宽，但对于较大消息，它却降低了吞吐量。由于绝大多数MPI应用都需交换大消息，优化带宽相比降低延迟更重要，因此Mont-Blanc原型系统通常选择使用专有驱动程序＋打补丁USBNET内核，作为它的稳定网络配置。

5. 全系统评估

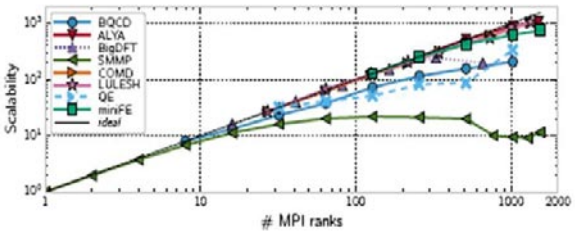
本节采用全规模、产品化的MPI应用和美国能源部国家实验室使用的三个小型参考应用来评估Mont-Blanc原型系统的性能，所有测试应用都使用OpenMPI，且只在CPU上运行。

（1）应用可扩展性

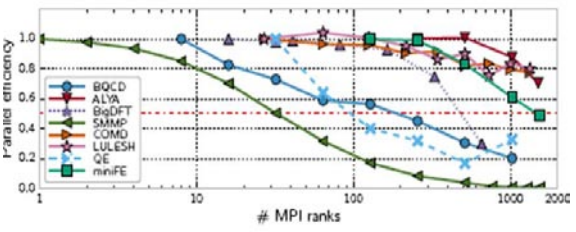
如前所述，在只用CPU核时，Mont-Blanc节点比MareNostrum-III节点慢18倍。这意味着必须把工作负载线性地分散到18倍多的计算节点中，才能二者达到同等性能。

图9显示了Mont-Blanc原型系统MPI应用的强扩展和弱扩展性。每张图都带有相应的并行效率，以更好地展示应用可扩展性。值得注意的是16个Mont-Blanc节点已超越了2个EMB刀片，32个节点则超越了3个刀片。由于4GB DRAM/每节点的限制，大多数应用都超出一个节点运行。

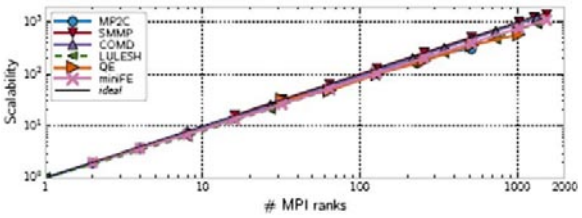
从32节点开始，SMMP强扩展性迅速下降，并行效率降到50%，性能变差，甚至在512节点还会下降。BQCD和QE也显示出强扩展性迅速降低，但在64节点时运行效率依然高于50%。其余应用可线性地扩展至数百节点，其中四个应用在全机规模运行的效率仍高于50%。



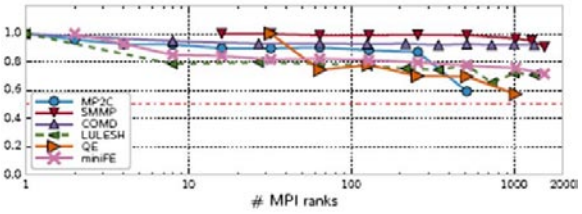
（a）强扩展性



（b）强效率



（c）弱扩展性



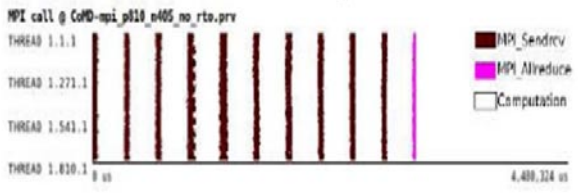
（d）弱效率

图9 Mont-Blanc原型系统的MPI应用可扩展性与并行效率

结果表明，把应用扩展到16个节点以补偿与一个MareNostrum III节点的性能差异是很合理的。但是，不是所有应用都可以进一步扩展以补偿多个MareNostrum III节点。弱扩展性的结果则好很多，大多数应用在最大问题规模时，运行效率仍高于70%。尤其是CoMD和SMMP的运行效率更高于90%，但QE和MP2C的效率则降到60%。

性能分析显示，其缺少可扩展性的原因除了低带宽/高延迟GbE网络外，还包括互连中丢失数据包（每次至少引发一个重传超时，简称RTO）和由于调度程序抢占，而导致的负载失衡的影响。

（a）有数据包丢失



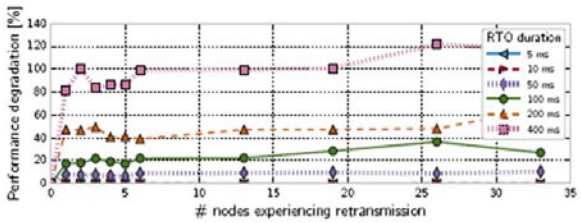
（b）无数据包丢失

图10 数据包丢失对MPI并行应用的影响

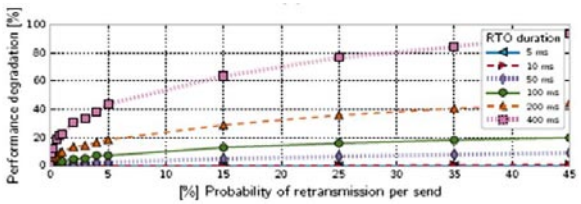
图10显示了对CoMD实际运行和模拟运行的（使用Dimemas消除网络重传）分析，两张曲线图具有相同的时间标度。结果显示，系统本机执行会有很多数据包丢失（大多数应用在通信阶段至少有一次重传），性能降低1.47倍。当然，这取决于应用类型，还要看通信模式、消息大小、通信量等。进一步分析MPI发送操作的持续时间，可以发现CoMD每个数据包都会经历多次重传，MPI发送持续时间平均为158ms（最佳为50ms），且经常会达到400ms。

图11（a）把性能下降表示为，每条被发送消息遭受重传损失节点数的函数。结果表明，该损失与重传延迟呈线性关系。但更重要的是，每当一个节点不得不重传时，那么整个应用几乎都要为这一损失付出代价。

图11（b）则把性能下降表示为，需要重传（被任何节点）消息数的函数。结果表明，该损失与重传延迟和重传概率呈线性关系。上述两个结果都显示，在整个系统中避免重传，或及时集结重传很重要。因为，一旦某节点需要重传，其他节点是否需要重传就显得并不那么重要了。例如，交换机中一个故障导致所有与之相连的节点都需重传，与某节点因NIC故障导致只有该节点需要独自重传，两者损失是相似的。



（a）特定节点的每条消息都受影响



（b）随机的消息受影响

图11 由于重传而导致的性能降级

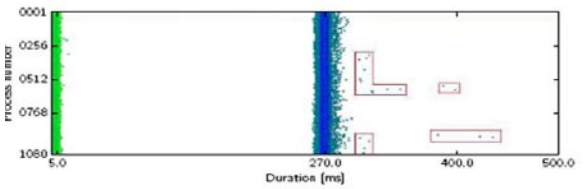


图12 计算阶段持续时间的二维柱状图

为了把重传损失降到最低限度，现将TCP/IP协议栈中的RTOmin参数从缺省值200ms减小到5ms（系统中最低）。当通过减小RTOmin参数降低重传损失时，就有望通过重传早期检测（RED）来减轻重传影响。但是，数据包丢失不仅发生在交换机缓存器中，节点也会丢失数据包。由于刀片交换机（向前

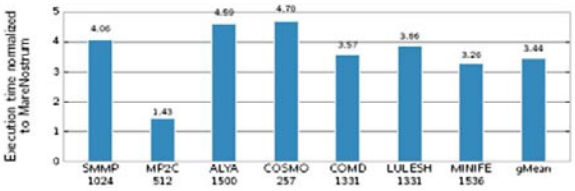
转发网络流量）不支持显式拥塞通知（ECN），因此无法控制传输速率。图12显示了CoMD在实际执行中，计算阶段持续时间的二维柱状图。

该图显示了5ms和270ms两个持续时间区域。把5ms区域与TCP/IP重传相匹配（匹配5ms RTO设置，并确认许多进程遭受重传）。剩下的时间用在270ms区域，匹配该应用的一个内部循环持续时间。在270ms区域外，出现一组明显占用更多时间的离群点。

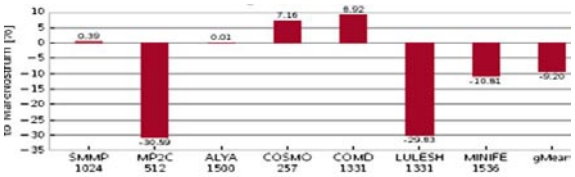
检查这些计算阶段的IPC，可确认执行时间的差异与应用的负载失衡无关，导致差异的是外部因素，其原因可归结于调度程序抢占。进一步模拟不同的噪音注入频率和噪音持续时间。结果表明，OS噪音带来的性能影响与噪音注入的概率及噪音持续时间与计算猝发时间的比率呈线性关系。也就是说，含有短计算猝发的应用比含有长计算猝发的应用，更容易受到OS噪音的影响，导致性能的降低。

图13显示了当采用的MPI行列数相同（核数相同）时，Mont-Blanc与MareNostrum III的性能和能耗的对比。由于各应用对其所能采用的MPI行列数并不是完全可控的，因此每个应用所占用的核数也不完全相同，从257到1536不等。结果显示，Mont-Blanc系统平均慢3.5倍，但运行应用的能耗会减少9%。然而，这些应用都没有经过优化以使用GPU或OmpSs。因此，可以期望一旦GPU与CPU一起使用，Mont-Blanc系统将会有更好的能效。

表4显示了在以相同执行时间为目的时，Mont-Blanc原型系统与MareNostrum III的对比。由于实验是为了测试应用在Mont-Blanc上的强扩展能力，因此研究人员保持输入设置恒定不变，增加MPI行列数，以获得应用在含有64个MPI行列的MareNostrum III上运行的相同执行时间。



（a）性能比较



（b）能耗比较

图13 Mont-Blanc与MareNostrum III的比较

（2）与传统HPC的比较

结果显示，Mont-Blanc需要多3.5-3.75倍的MPI行列才能匹配MareNostrum III的执行时间。这与常数MPI队列所观察到的3.5倍减速是一致的，显示两个系统具有相近的可扩展性；从能耗方面看，两个系统消耗大致相同的能量；从机柜空间看，Mont-Blanc需要7个机柜单元（一个BullX机仓、9个刀片x15个节点、270个核），而MareNostrum III则需8个机柜单元（4个2U节点）。

表4 相同执行时间下Mont-Blanc与MareNostrumIII的比较

	CoMD		*miniFE*	
	MareNostrum III	*Mont-Blanc*	*MareNostrum III*	*Mont-Blanc*
MPI行列数	*64*	*240*	*64*	*224*
执行时间 [s]	*70.72*	*68.05*	*71.66*	*72.19*
平均功耗 [W]	*992*	*1083*	*1065*	*1034*
能量 [Wh]	*195*	*205*	*212*	*207*
机柜单元数	*8*	*7*	*8*	*7*

由此可得出如下结论：当只用CPU计算时，Mont-Blanc与MareNostrum III的能效几乎相同，但Mont-Blanc的集成密度略高。

6. 结束语

本文详细描述了Mont-Blanc原型系统的架构，并将其与MareNostrum III超级计算机对比。结果显示，在MPI进程数相同时，Mont-Blanc比MareNostrum III慢4倍，但是通过将应用扩展至多4倍节点，就可以补偿性能差异，并且运行时间相近，能耗相同。由于应用必须扩展至更多节点，因此负载平衡成为应用的关键设计问题。通过使用MPI+OpenMPI和一种运行时动态负载平衡方法，可改善负载失衡问题。该方法可通过检测已完成任务的空闲核，使其为遭受重传影响的核分担一部分工作。

本文提倡利用为嵌入式和移动领域而开发的IP来构建专用SoC，并增加HPC所需的功能，包括无损网络、ECC存储保护和加速器等。从Mont-Blanc问