通过调研JAVA在HPC中的开发潜能，期望能并行化osv, mhe等地震数据解释Java代码。

地震数据解释，主要是加速3D数组计算(inline, crossline, time/depth coord.)，osv等已经实施了基于fork-join的多线程并行，考虑使用：

（1）OpenMP：基于omp4j (2015)

（2）CUDA/OpenCL：基于jcuda库和jocl库

（3）MPI：基于Fast MPJ (至少10年没更新了)或MPJ Express(纯Java, latest: 2015)以及Java binding to OpenMPI (推荐, 2016)

（4）MPI+X (X=OpenMP/CUDA/OpenCL)，基于jcuda和jocl库

# Guillermo L. Taboada, Sabela Ramos, Roberto R. Expósito, Juan Touri, Ramón Doallo. 2013. Java in the High Performance Computing arena: Research, practice and experience. Science of Computer Programming, 78: 425-444

本文综述了JAVA语言在HPC开发中的作用和实践。

Java for HPC可分为以下4类：

1 共享内存编程

2 Java sockets

3 Remote Method Invocation (RMI)

4 Java下的消息传递

## 1 共享内存并行

2个OpenMP-like的Java实施：JOMP和JaMP

（1）JOMP可以与MPI联合实施，相比纯MPI并行的效率提高3倍。

（2）JaMP是为Jackal环境实施的OpenMP-like并行，因此对实施环境有限制。

M.E. Kambites, J. Obdrzálek, J.M. Bull, An OpenMP-like interface for parallel programming in Java, Concurrency and Computation: Practice and Experience 13 (8-9) (2001) 793-814.

M. Klemm, M. Bezold, R. Veldema, M. Philippsen, JaMP: an implementation of OpenMP for a Java DSM, Concurrency and Computation: Practice and Experience 19 (18) (2007) 2333-2352.

上述2个项目没有开放代码！

这里没有提及一个OpenMP的Java开发项目：

Petr Belohlavek. OpenMP for Java. Faculty of Mathematics and Physics, Charles University in Prague, BACHELOR THESIS, 2015

## 2 消息传递并行

也主要有2个项目：

（1）mpiJava 1.2 API, 遵守MPI 2.0标准的接口，但限制于支持MPI 1.1，已经被MPJ Express项目代替

（2）JGF MPJ

M. Baker, B. Carpenter, G. Fox, S. Ko, S. Lim, mpiJava: an object-oriented Java interface to MPI, in: Proc. 1st Intl. Workshop on Java for Parallel and Distributed Computing, IWJPDC’99, in: LNCS, vol. 1586, San Juan, Puerto Rico, 1999, 748-762.

B. Carpenter, V. Getov, G. Judd, A. Skjellum, G. Fox, 2000 MPJ: MPI-like message passing for Java, Concurrency: Practice and Experience 12 (11):1019-1038.

从2003年以来，研发的消息传递实施的Java库有：

（1）MPJava是第1个Java消息传递库，基于Java NIO套接字实施，利用了他们的尺度化和高性能通信；缺陷：不稳定，不是线程安全的，不能通过多线程利用多核系统。被MPJ Express项目代替，MPJ Express是一个纯Java的消息传递实施，是线程安全的，可以联合纯Java的共享内存(smpdev设备)与新的IO包（Java NIO）通信（niodev设备），支持高性能的Myrinet通信库（在mxdev设备内）。

（2）其他几个都有缺陷

（3）Fast MPJ (F-MPJ)支持高速网络，通过Java Fast Sockets。目前，免费下载的源码是基于TCP/IP通信，其他通信设备使用的库需要私下联系。



MPJava （第1个针对消息传递并行的Java）

B. Pugh, J. Spacco, MPJava: high-performance message passing in Java using Java.nio, in: Proc. 16th Intl. Workshop on Languages and Compilers for Parallel Computing, LCPC’03, in: LNCS, vol. 2958, College Station, TX, USA, 2003, pp. 323-339.

F-MPJ

G.L. Taboada, J. Touri, R. Doallo, F-MPJ: scalable Java message-passing communications on parallel systems, Journal of Supercomputing (2011) doi:10.1007/s11227-009-0270-0.

F-MPJ的通信设备：



Java binding to OpenMPI

MPJ Express和F-MPJ等库尽管使用纯Java语言实现MPI通信传递，但受限于通信效率（其实已经有很多高速通信设备和协议研发了，比如Java通信 中间件、低延迟网络通信—InfiniBand or Myrinet或通过共享内存通信）；由于用纯Java编写MPI API，因此可使用的MPI API有限（也足够用了）。

Java binding to OpenMPI是在Java中绑定原始的OpenMPI的C语言库，解决了上述问题，但目前仅与OpenMPI绑定（应该也可以用于其他的MPI库，如MPICH2）。

## 实施有效评估HPC benchmarks

NAS Parallel Benchmarks (NPB) suite for MPJ (NPB-MPJ)

实施了(NPB-MPI), OpenMP (NPB-OMP), Java threads (NPB-JAV) and ProActive (NPB-PA).



为最大化NPB-MPJ效率，采用plain objects设计，降低纯面向对象设计的开销（能降低95%的开销）。因此，仅使用1个对象，而不是为问题域的每个元素定义一个对象。例如，复数就定义为2个元素的数组。

Java对多维数组的计算效率很低（N维数组在Java中定义为N-1维数组的数组，因为数据在内存中不能保证是连续存储的），显著限制了NPB-MPJ的计算效率。可以通过数组flatten优化降低这个开销，也就是将多维数组映射为1D数组。因为，C/Fortran版本的相邻单元在Java中也是连续存储的了，可以挖掘数据局部性的潜力。

NPB-MPJ利用了JVM JIT (Just-In-Time)编译优化。Bytecode的JIT编译针对重度使用的方法，其运行成本高，显著增加运行时。如：多维元素映射到1D数组。

## 支持数据并行的Java库

Java可以使用硬件加速器做数据并行化计算，如GPUs。

编译器在编程语言层级，如JCUDA

另一方面，与基加速于库的接口。如Java绑定CUDA：jcuda.org, jCUDA, JaCUDA, Jacuzzi, java-gpu

参考文献

Y. Yan, M. Grossman, V. Sarkar, JCUDA: a programmer-friendly interface for accelerating Java programs with CUDA, Proc. 15th Intl. European Conference on Parallel and Distributed Computing, Euro-Par’09, Delft, The Netherlands, 2009, 887-899. 没有源码！

jcuda.org, http://jcuda.org [Last visited: May 2011]. 德国人研发的java库，还有jocl库

jCUDA, http://hoopoe-cloud.com/Solutions/jCUDA/Default.aspx [Last visited: May 2011]. 无法访问了！

JaCuda, http://jacuda.sourceforge.net [Last visited: May 2011]. 已下载

Jacuzzi, http://sourceforge.net/apps/wordpress/jacuzzi [Last visited: May 2011]. 已下载

java-gpu, http://code.google.com/p/java-gpu [Last visited: May 2011]. 不能访问了

Java与OpenCL的绑定：

jocl.org, http://jocl.org [Last visited: May 2011].

JavaCL, http://code.google.com/p/javacl [Last visited: May 2011]. Move to https://github.com/nativelibs4java/JavaCL

以上是Java中数据并行的支持，尽管其效率比直接的CUDA/OpenCL效率低，由于Java数据在与GPU之间的移动成本。支持在Java程序中执行用户编写的CUDA代码，自动支持原始变量和多维数组的数据转移。

解决上述低效率的一个项目JCudaMP，在OpenMP框架下挖掘有效的GPU计算。

G. Dotzler, R. Veldema, M. Klemm, JCudaMP: OpenMP/Java on CUDA, in: Proc. 3rd Intl. Workshop on Multicore Software Engineering, IWMSE’10, Cape Town, South Africa, 2010, pp. 10-17. 没有源码！

直接生成GPU可执行代码（不是JNI访问CUDA/OpenCL），优化在GPU上运行Java代码效率，可使用Java编译器Jikes，可以自动并行化循环计算(A. Leung, 2011)。

A. Leung, O. Lhoták, G. Lashari, Parallel execution of Java loops on graphics processing units, Science of Computer Programming (2011) 没有源码！

## 4 效率评估