

# Using Java for Plasma PIC Simulations

Vladmir Getov<sup>a</sup>, Quanming Lu<sup>a,b</sup>, and Shu Wang<sup>c</sup>

<sup>a</sup>University of Westminster, London, United Kingdom

<sup>b</sup>University of Science and Technology, Hefei, Anhui, P. R. China

<sup>c</sup>China Netcom Corporation, ZhuHai, Guangdong, P. R. China

# High Performance Java

---

1. Can Java be used for parallel computing?
2. Can Java deliver performance comparable to Fortran/C?

Java Grande Forum since 1998

# Distributed and High Performance Group at Westminster

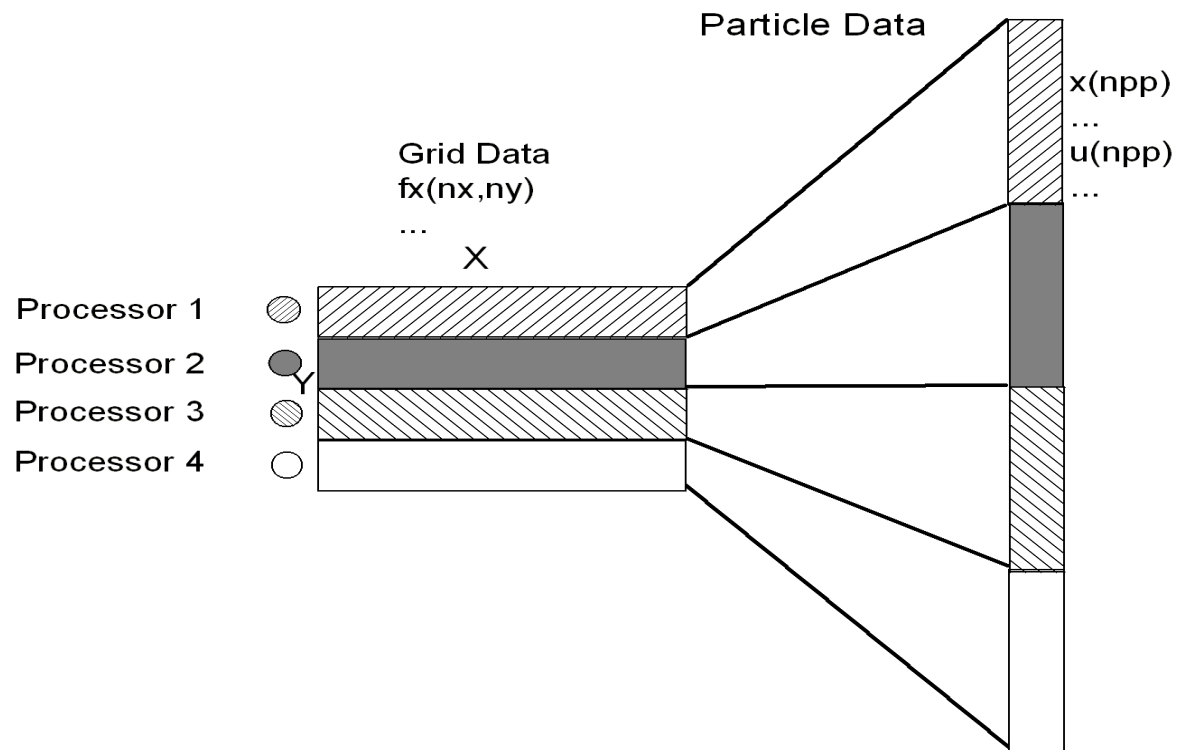
- Message passing for Java
- Automatic wrapper generator
- Wrapped existing message-passing libraries - MPI
- Wrapped scientific libraries - BLAS, BLACS, LAPACK, ScaLAPACK, etc.
- Needed an application case study

# The Skeleton plasma PIC code

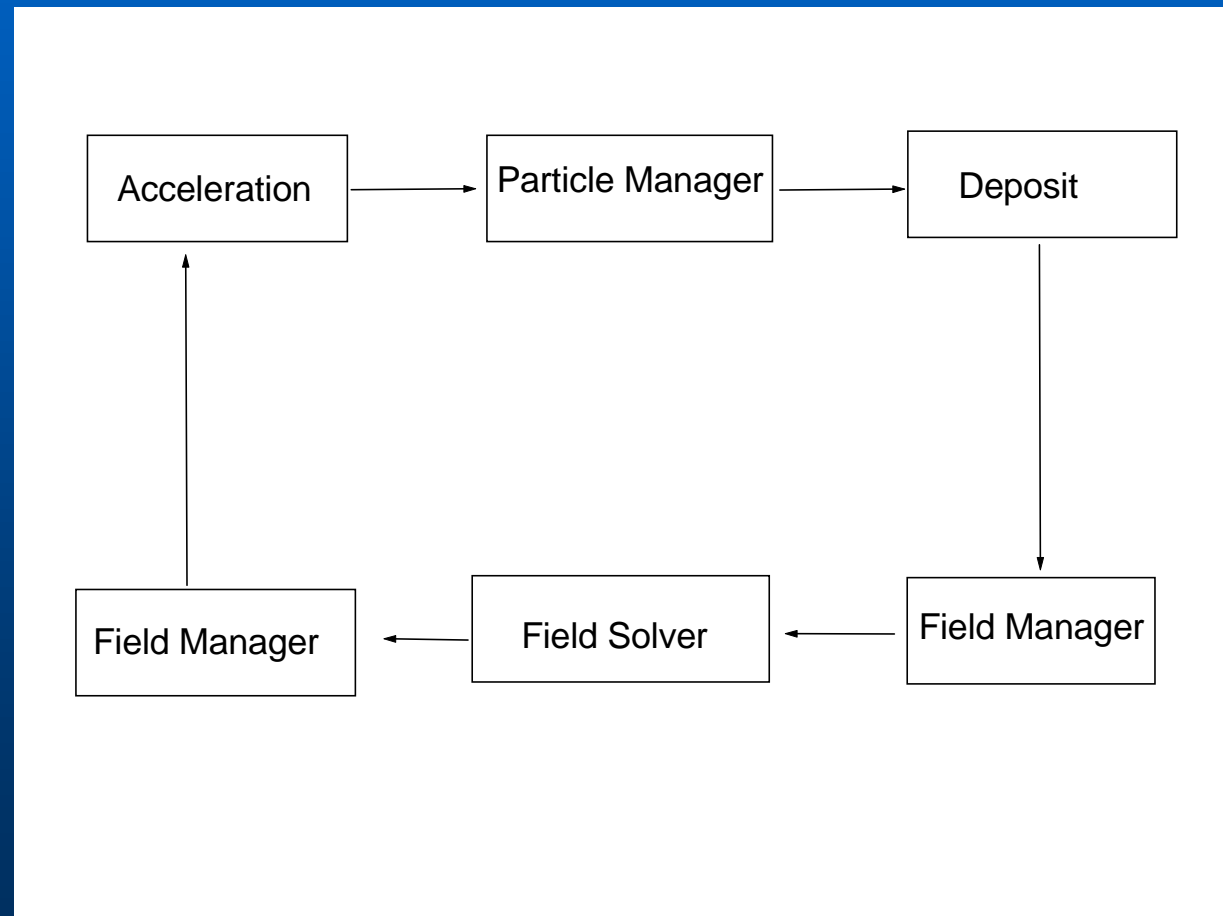
---

- Electrostatic code - move only electrons
- The physical problem is a plasma beam instability
- Quadratic spline functions for interpolation
- It is a good testbed for parallel computing

# Grid and particle partitioning



# General structure of the PIC skeleton



# Our 2D and 3D PIC codes in Java

---

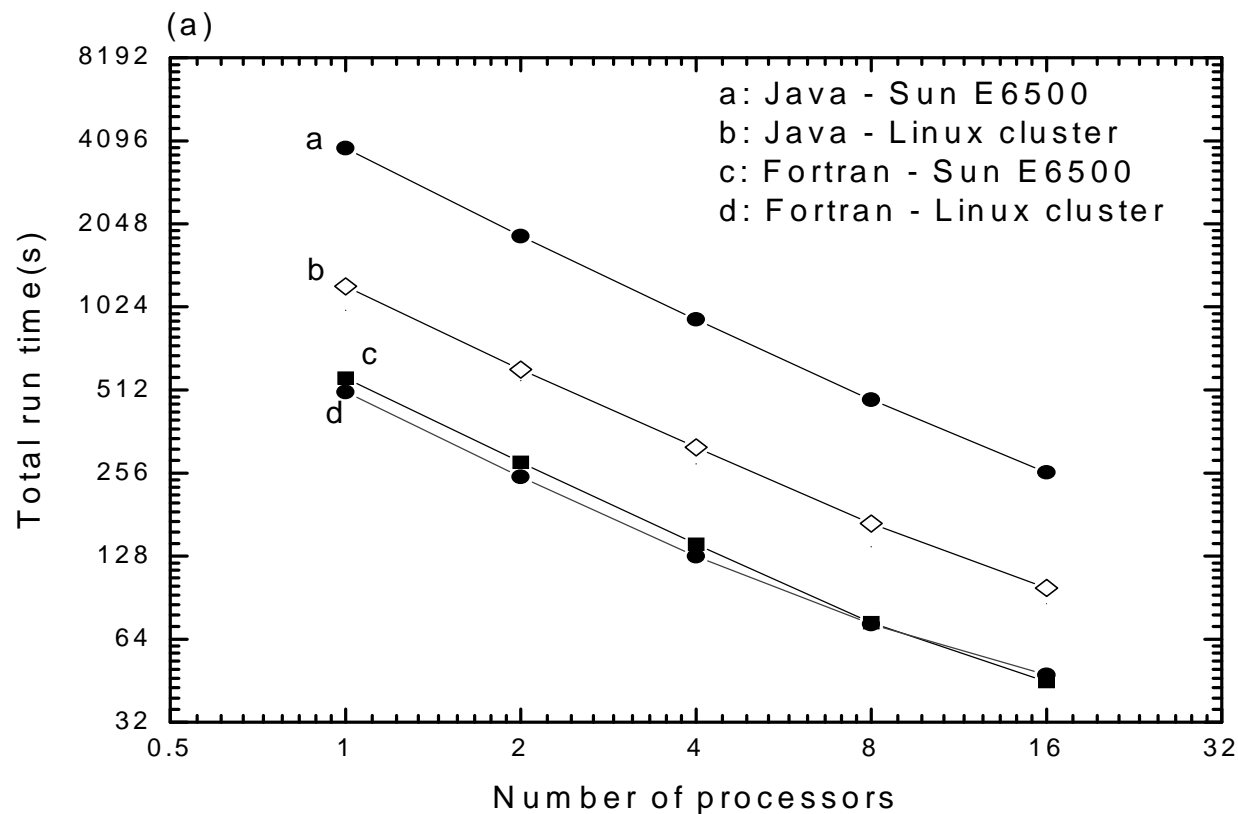
- Two main Java classes: “plasma” and “field” which are based on “PICSimulator”.
- The “PICSimulator” class defines some basic parameters.

# Performance Experiments

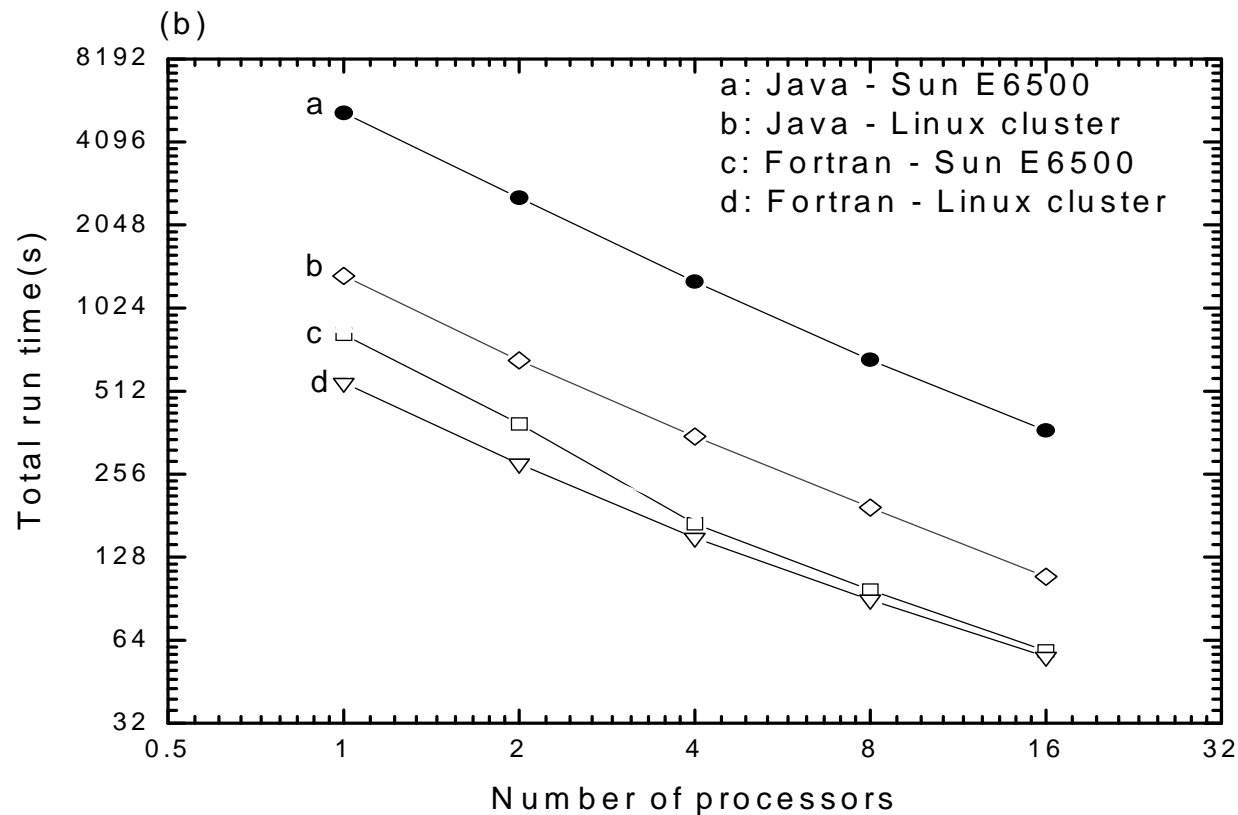
1. Codes: 2D and 3D plasma PIC codes.
2. Computers:
  - Sun E6500: 30\*336MHz Ultra Sparc 2 processors with shared memory, Solaris 2.6. Sun JDK 1.3, Sun Fortran F90 2.0., Sun MPI 2.0
  - Linux cluster: 16\*900MHz Pentium III processor, RedHat 6.2, Sun JDK 1.4, Gnu Fortran 77, MPICH1.2, 100Mb/s switch.
3. Java interface to MPI: MPJ RI (mpiJava1.2.)



# Total runtime versus number of processors - 2D PIC codes



# Total runtime versus number of processors - 3D PIC codes



# Runtime of main parts of the 2D PIC codes on Sun E6500

No. of proc	Fortran		Java	
	Particle time	Field time	Particle time	Field time
1	549.86	15.60	3780.68	69.06
2	272.33	7.73	1800.93	47.33
4	136.49	4.80	893.18	28.47
8	69.45	3.94	451.70	20.13
16	39.31	5.59	243.33	13.74

# Runtime of main parts of the 2D PIC codes on the PC cluster

No. of proc	Fortran		Java	
	Particle time	Field time	Particle time	Field time
1	488.26	15.05	1194.92	20.10
2	239.36	8.71	592.21	15.03
4	123.19	4.57	305.87	11.64
8	68.55	4.06	158.18	9.89
16	43.20	4.28	89.41	8.64

# Runtime of main parts of the 3D PIC codes on Sun E6500

No. of proc	Fortran		Java	
	Particle time	Field time	Particle time	Field time
1	794.10	27.83	5222.12	126.89
2	353.82	35.06	2567.14	116.18
4	154.02	15.34	1275.15	69.92
8	83.32	14.06	663.68	73.89
16	41.02	17.53	368.08	51.67

# Runtime of main parts of the 3D PIC codes on the PC cluster

No. of proc	Fortran		Java	
	Particle time	Field time	Particle time	Field time
1	526.64	20.57	1298.93	40.23
2	264.56	17.49	627.16	33.73
4	134.77	15.28	322.09	28.48
8	74.62	14.95	168.01	25.57
16	40.32	15.36	87.40	21.38

# Conclusions and future work

---

- 2D and 3D PIC codes successfully ported to Java - <http://perun.hscs.wmin.ac.uk>
- This first (naïve) port delivers relatively low performance - 15% of Fortran on Sun E6500 and 45% of Fortran on our Linux cluster.
- Optimizations using pure Java
- Mixed Java/Fortran code
- Comparisons with multithreading
- Porting to a Grid environment