

MPJ

Message Passing in Java

Über MPJ

- MPJ ist ein von MPI abgeleiteter 'Standard' für Message Passing in Java
- je nach Implementation nicht vollends kompatibel zu MPI



MPJ

<http://www.cs.vu.nl/ibis/mpj.html>

Vorteile

- 100% Java
- ist leicht aufzusetzen
- setzt den MPJ Standard um
- basiert auf getestetem Ibis Framework für Netzwerkkommunikation

Nachteile

- ist nicht kompatibel zu ursprünglichem MPI
- jeder Knoten muss separat gestartet werden, MPI dagegen bietet Möglichkeiten, ein Programm zentral zu starten
- diese Implementation hat einen eigenen Server, MPI nutzt Bordmittel von Betriebssystemen

Alternativen

- JavaMpi
 - nicht komplett Java - nutzt nativen Code
 - versucht zu MPI kompatibel zu bleiben
 - Projekt leider 'veraltet' und nicht aktiv
- MPJ Express
 - 100% Java
 - Kompatibel zu MPI
 - wirkte noch nicht ausgereift



MPJ Funktion

<http://www.cs.vu.nl/ibis/downloads/mpj/usersguide.pdf>

Struktur

#1



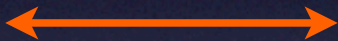
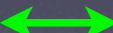
#4



#2



#3



Server (I)

- Dient zur Anmeldung von Clients / 'Nodes'
 - Kommunikation zwischen Knoten läuft eigentlich direkt
 - Server kann aber für Firewalls zwischen Knoten kompensieren.
- Ein Server kann mehrere 'Pools' verwalten
 - Jeder Pool ist ein Cluster von Knoten

Server (2)

- Server starten durch Aufruf des 'mpj-server' Skripts
- Damit der Server korrekt arbeitet, muss die Umgebungsvariable `$MPJ_HOME` gesetzt sein.
- Befehl: `$MPJ_HOME/bin/mpj-server`
- Parameter `--events`: Zeigt detaillierte Infos zum Server und den verbundenen Clients
- Beenden des Servers mit `Strg+C`

Client (I)

- Man spricht hier eher von Knoten / 'Nodes'
- Jeder Knoten wird wie bei MPI durch seinen Rang identifiziert
- Knoten kommunizieren direkt miteinander
 - bei Firewalls auch über Server, auch wenn der nur zur Anmeldung dienen sollte.

Client (2)

- Knoten starten durch Aufruf des 'mpj-run' Skripts
- Damit der Knoten korrekt arbeiten kann, muss die Umgebungsvariable \$MPJ_HOME gesetzt sein.
- export CLASSPATH=Pfad zu den %Javaklassen
% / %JAR-Archiven%
- Befehl: \$MPJ_HOME/bin/mpj-run %PARAMETER
%HAUPTKLASSE

Client (3)

- 3 Parameter sind für den Start nötig:
 - -Dibis.server.address=%IP / HOST%
 - Adresse vom Rechner mit dem Server
 - -Dibis.pool.name=%POOL / CLUSTER%
 - Knoten im gleichen Pool gehören zusammen.
 - -Dibis.pool.size=%KNOTENANZAHL%
 - Gibt an, auf wieviele Knoten im Pool der Server vor der Ausführung wartet



MPJ API

<http://www.cs.vu.nl/ibis/javadoc/mpj/index.html>

Zentrale Klasse

- Singleton Klasse MPJ
 - Funktionen meist analog zu den MPI 'Vorgängern'
 - init, getRank, getSize, finish.....
 - send, isend, bsend....
 - scatter, gather....

Java Spezifisches (I)

- Jede Java-Methode, die MPJ aufruft, kann eine MPJException werfen
 - alternativ zum Throw kann man die MPJ Befehle in try/catch Blöcke einbetten

```
void foo() throws MPJException
{
    ..
    MPJ.send(...)
    ..
}
```

Java Spezifisches (2)

- Da es in Java keine Pointer gibt, arbeiten die Send und Receive Methoden in Java mit Arrays.
- Arrays ahmen die Funktionalität von Pointern nach.

```
void foo() throws MPJException
{
    ...
    byte[] bytes = str.getBytes();
    int[] buf = new int[1];
    buf[0] = bytes.length;
    MPJ.COMM_WORLD.send(buf, 0, 1, MPJ.LONG, 1, 0);
    MPJ.COMM_WORLD.send(bytes, 0, bytes.length, MPJ.BYTE, 1, 1);
    ...
}
```