自宅で作る PC クラスタ

上川純一

Junichi Uekawa

LILO

dancer@netfort.gr.jp

dancer@debian.org

なぜ自宅に**PC**クラスタ

- 高速な計算がしたい

なぜ自宅にPCクラスタ

- 高速な計算がしたい
- 余っているパソコンがもったいない

なぜ自宅にPCクラスタ

- 高速な計算がしたい
- 余っているパソコンがもったいない
- 遊んでみたい

なぜ自宅に PC クラスタ

- 高速な計算がしたい
- 余っているパソコンがもったいない
- 遊んでみたい
- とりあえず，かっこいい(？)

なぜ自宅にPCクラスタ

- 高速な計算がしたい
- 余っているパソコンがもったいない
- 遊んでみたい
- とりあえず，かっこいい(？)
- 並列プログラミングができると彼女に自慢できる

PCクラスタとは

- 複数台のマシンを繋げて構築し，協調して動作させて一台ではできないような事をするシステム
- 頑張ったらプログラムがより高速に実行できるかもしれない
- 頑張ったらより信頼性のあるシステムが実現できるかもしれない

PCクラスタの材料

- 複数台のパソコンとネットワーク
- OS (GNU/Linux)
- パソコンを協調して動作させるためのシステム

クラスタの種類

- HA = High Availability
  落ちないシステムが作りたい
- HPC = High Performance Computing
  より高速に計算をさせたい

HPCクラスタの目的

- 大きなプログラムを高速化する
- データ交換の量 < 計算量 ならネットワークを介しても高速にできる
  - ⇒ POVRay
  - ⇒ πの計算

HPCクラスタの目的

- 大きなプログラムを高速化する
- データ交換の量 < 計算量 ならネットワークを介しても高速にできる
  - ⇒ POVRay
  - ⇒ πの計算

HPCクラスタの現実

- 従来からあるプログラムはそのままクラスタで動くわけではない．
  - 現在あまり一般人が遊べるアプリケーションが開発されていない

HPCクラスタの現実

- 従来からあるプログラムはそのままクラスタで動くわけではない．
  - 現在あまり一般人が遊べるアプリケーションが開発されていない
- じゃあ，自分で並列プログラムを書こう．
  - MPI
  - PVM など

MPIとは

- 並列用の開発環境の標準規格の一つ
- 同じプログラムが複数のマシンで動き，MPIの関数で通信
- 色々な実装がある
  - mpich
  - lam
  - MPIPro

mpich

- MPIの模範実装
- とりあえず動くまでが楽(？)

下準備

- クラスタの設定
  - /homeを共有 (NFS)
  - rshで認証無しで login
  - mpichのインストール

マスターの設定

- mpichをインストール
- NFSサーバの`/etc/exports`の設定
- rshのクライアントをインストール

mpichのインストール

- `apt-get install mpich`
- `/etc/mpich/machines.LINUX`
  にクライアントマシン名を列記

mpichのインストール

- `apt-get install mpich`
- `/etc/mpich/machines.LINUX` にクライアントマシン名を列記
- もしくはソースを取って来て，コンパイル．(時間がかかるけど)

mpichのコンパイル

自分でコンパイルする場合は,

```
http://www.mcs.anl.gov/
mpi/mpich/からダウンロード
./configure
--with-device=ch_p4
-prefix=/usr/lib/mpich
-rsh=/usr/bin/rsh &&
make && make install
```

スレーブの設定

- 頑張って計算してくれる方々
- rshのサーバをインストール
- /home等をマウント？(マウントしなくても，毎回rcpしてもOK．)
  - ⇒ マスターはスレーブに対してrshでプログラムを実行させる

rshの設定

- rshのサーバとクライアントを入れる(apt-get install rsh-server rsh-client)
- /etc/hosts.equivで互いにパスワード無しで入れるようにする.
- /etc/hostsでちゃんと互いに見えるようにする.

はまりどころ

- マスターの /etc/exportsとスレーブの/etc/fstabが整合性が取れているか
- /etc/passwdの整合性.
  - NISを導入するのが楽.
  - もしくはcronで/etc/passwdをrcp

mpichでのCプログラム

- MPI_Init(&argc, &argv)で始まる
- MPI_Comm_rankで自分のプロセス番号を知る
- MPI_Send, MPI_Recv, MPI_Reduce等で通信
- MPI_Finalize()で終了

mpichの実行

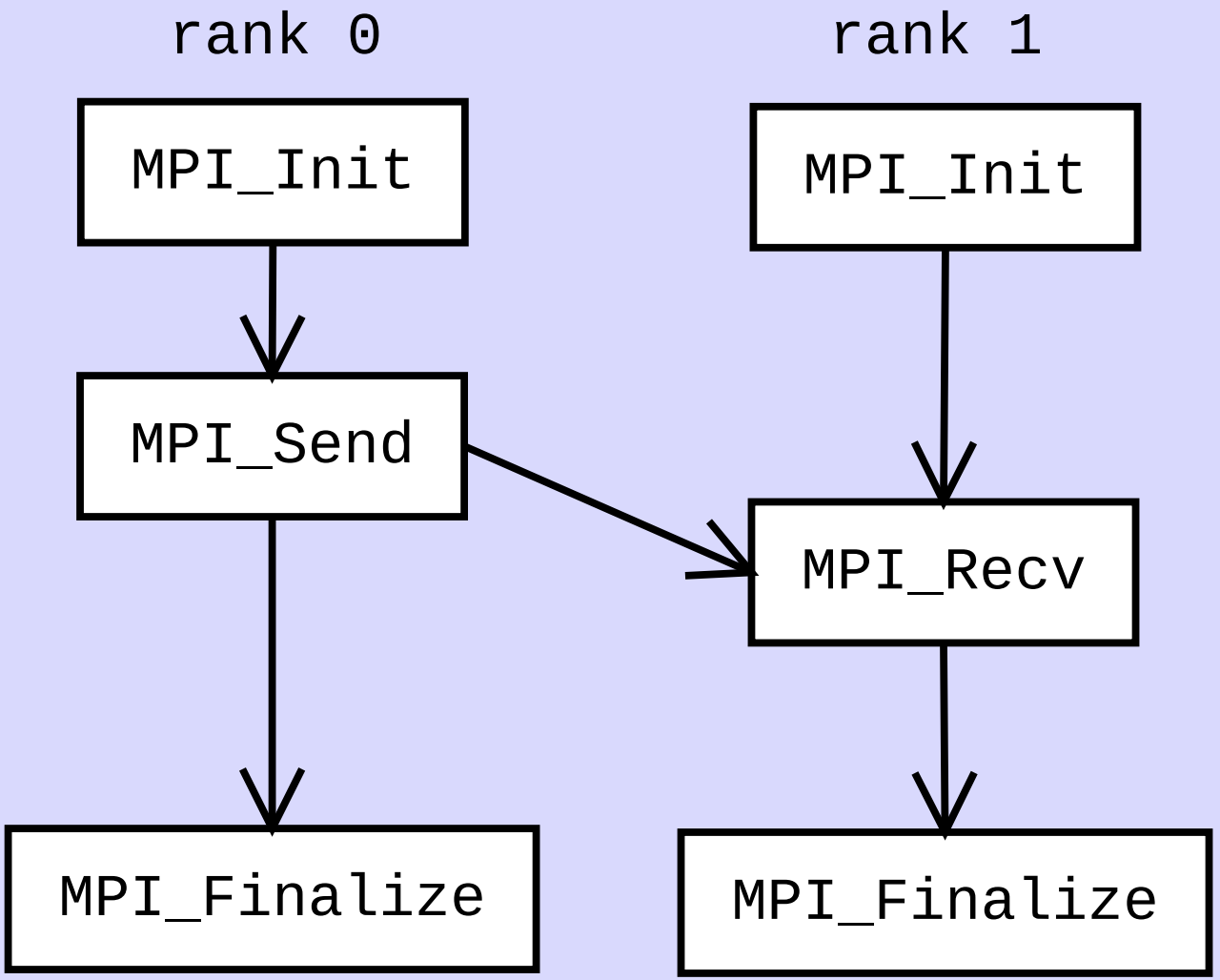
- `mpicc -o hello hello.c`
  - ⇒ MPI用にプログラムをコンパイル
- `mpirun -np 3 ./hello`
  - ⇒ 3ノードでプログラムを実行

```
#include <stdio.h>
#include <mpi.h>

main(int ac, char ** av )
{
  int hoge=0;
  int rank;

  MPI_Status Status;
  MPI_Init(&ac, &av );
  MPI_Comm_rank( MPI_COMM_WORLD, &rank);
  if (rank == 1)
    {
      printf ("first mutation. testing\n");
        MPI_Send(&hoge, 1, MPI_INTEGER, 0, 0, MPI_COMM_WORLD)
    }
  else
    {
      printf ("second mutation. testing\n");
      MPI_Recv(&hoge, 1, MPI_INTEGER, 1, 0,
              MPI_COMM_WORLD, &Status);
    }
  MPI_Finalize();
}
```

図解


rank 0
rank 1
MPI_Init
MPI_Init
MPI_Send
MPI_Recv
MPI_Finalize
MPI_Finalize

参考文献

- Debianの`mpi-doc`パッケージ (mpichに付属の説明書)
- `http://lyre.mit.edu/~powell/debian-howto`
- `http://www.mcs.anl.gov/mpi/mpich/`

同志社大学のクラスタ例

- Cambria - 256-node
  - ディスクレスのノードが 15台につきディスクありのノードが 1台のシステムが 16で 256台.
  - ネットワークは 100BASE-TX
- Gregor - 64-node 2-way (128PE)
  - Myrinet2000 で相互接続