Rainer Spurzem

Parallel Computing

1. Introduction to Scalability

Tuesday, Nov. 20, KIAA, 7:00 - 8:30 p.m.

1.1 Parallel Thinking

• The 1st lecture of Prof. Wen-mei Hwu **(partly only!)**. See:
http://silkroad.bao.ac.cn/web/index.php/conferences/iccs3

Parallel Thinking - Scalability
GPU / CPU parallelism differences
Data Parallelism
Work Parallelism

Memory Bandwidth
Amdahl's Law
Load Balance
Algorithm Complexity

If X is the fraction of a program's operations which can be parallelised, then the maximum speed-up S using a large number of parallel processors is S = 1/(1-X). Examples: X = 0.5 ; S = 2.
X = 0.99 ; S = 100. Here we have used the assumption that the number of processors is large compared to unity. There are more precise formulae for Amdahl's law with finite number of processors.

1.2 Apply to NBODY6++

Memory Bandwidth/Amdahl's Law/Load Balance

• What is NBODY6++? See: http://silk0.bao.ac.cn/nb6mpi/
(Inofficial Pre-Release 2012, Website still contains 2007 version!!)

Main Differences Between NBODY6 and NBODY6++:

- Parallel Work Structure (intgrt.F, Do-Loops)
- Contiguous Data Structure (intgrt.F, XMPI, YMPI)
- Better Load Balance (intgrt.F, no sorted time steps)
- Control average neighbour number by NNBOPT
- Future: Parallel KS-Integration/Parallel Prediction
- Input File: Parameters isernb, iserreg - serial execution if block size is smaller than isernb (for nbint) / iserreg (for regint)

Some useful but not so important differences:

- different filenames (comm.1, comm.2, conf.3, sev.82, bev.83, lagr.7, and so on)
- more output, better format (Lagrangian radii, etc)
- many more small differences (Makefile, directory structure)

Usage of .F files:

intgrt.F is pre-processed with C-Preprocessor; it evaluates so-called preprocessor directives in the source code; they start with # , for example:

```
#ifdef PARALLEL
...
#endif
```

Preprocessor directives are selected with a compiler option:
-D PARALLEL compiles code between #ifdef PARALLEL and #endif.
Without -D PARALLEL these code lines will not be used!
WARNING - never keep .f if you have .F - the preprocessor directives will fail.

1.3 Parallel Communication Schemes and Literature

NBODY6++ runs in the SPMD (Single Program Multiple Data) Scheme (note that the GPU runs in a combination of SIMD and SPMD, see later). It means when you start the parallel NBODY6++ run on n cores, n identical copies of the program start. In parallel section they share their work and communicate data with each other.

NBODY6++ uses for communication a copy algorithm (all new information is copied immediately to all nodes); other algorithms are ring algorithm or (hyper)systolic algorithm (see Dorband, Hemsendorf, Merritt, 2003 (Journ. Comp. Phys.); Makino (2002). If the number of particles per node is large enough, all algorithms scale equally well. The phiGRAPE and phiGPU codes by Berczik and others (see e.g. Harfst et al. 2007, New Astronomy) use a mixed algorithm.

The copy algorithm in NBODY6++ is implemented manually with MPI_SENDRECV. Current modern implementations of MPI_BCAST will be equally efficient.

1.4 Hands-On Experiment on parallel computer 老虎 (no GPU yet)

ssh -l studentnn@laohu.bao.ac.cn For Cluster see: http://www.science-weekly.cn/skhtmlnews/2012/10/1896.html

- Profiling for NBODY6/6++

The code measures the wall clock time used for many things:

total, regular force, irregular force, adjust, regularised, prediction, overhead for parallelisation, communication time…

Your task: Do some experiment - run on 1,2,4,8,... processors.
In nbody6-mpi.lsf you need to change:

```
#BSUB -R 'select[type==any] span[ptile=2]'
#BSUB -n 2
```

The 'ptile=2' gives the number of cores *per node* you want to use.
Variation from 1 ... 6. Not more than 6 allowed.
The -n 2 option gives the total number of cores. Current limit for student queue: 6.

Explanation of times in output (line below 'PE N'):
ttot: total treg: regint
tirr: nbint tpred, tprednb: prediction
tmov: overhead for data move
tsub,tsub2: communication time after nbint, regint
xtsub1,xtsub2: number of bytes transferred

Plot results for these times as a function of core number. What is the maximum speedup we may get? What is the prediction of Amdahl's law? Compare.

2. Introduction to Many-Core Programming (GPU's for Parallel Computing)

Thursday, Nov. 22, KIAA, 7:00 - 8:30 p.m.

- What is a GPU? Why it can be so fast?
- Profiling, Performance Optimisation for GPU
- NBODY6/GPU (Nitadori/Aarseth) versus NBODY6++/GPU or how to use GPU's in a parallel computer for NBODY?
- Hands-on Experiments on 老虎 with GPU.

Notice: After each lecture (not before) I will write up a few more informations and put it on the Internet.