中国科学院国家天文台
National Astronomical Observatories, CAS

千人計畫
RECRUITMENT PROGRAM OF GLOBAL EXPERTS

UNIVERSITÄT HEIDELBERG
Zukunft. Seit 1386.

# Introduction to Parallel Computing: Many-Core, GPU, other ideas...

University


Rainer Spurzem
Astronomisches Rechen-Inst., ZAH, Univ. of Heidelberg, Germany
National Astronomical Observatories (NAOC), Chinese Academy of Sciences
Kavli Institute for Astronomy and Astrophysics (KIAA), Peking University

the SILK ROAD PROJECT at NAOC/KIAA
丝绸之路 计划

spurzem@ari.uni-heidelberg.de
http://silkroad.bao.ac.cn

PEKING UNIVERS
北京大

# History

http://cdsads.u-strasbg.fr/abs/1960ZA.....50..184V

Astronomisches Rechen-Institut in Heidelberg
Mitteilungen Serie A Nr. 14

## Die numerische Integration des *n*-Körper-Problemes für Sternhaufen I

Von

SEBASTIAN VON HOERNER

Mit 3 Textabbildungen

*(Eingegangen am 10. Mai 1960)*

Tabelle 5. *Zahl der gegenseitigen Umläufe, Häufigkeit des Auftretens und kleinster gegenseitiger Abstand $D_m$ der engsten Paare.* (Alle engsten Paare mit mehr als zwei vollen Umläufen wurden notiert)

| Umläufe | Häufigkeit | $D_m$ |
|---|---|---|
| 2—3 | 11 | 0.0102 |
| 3—5 | 9 | 0.0177 |
| 5—10 | 5 | 0.0070 |
| 10—20 | 2 | 0,0141 |
| 20—50 | 1 | 0.0007 |
| 50—100 | 1 | 0.0035 |
| 100—200 | 1 | 0.0039 |

Astronomisches Rechen-Institut in Heidelberg
Mitteilungen Serie A Nr. 19

## Die numerische Integration des *n*-Körper-Problems für Sternhaufen, II.

Von

SEBASTIAN VON HOERNER

Mit 10 Textabbildungen

*(Eingegangen am 19. November 1962)*

http://cdsads.u-strasbg.fr/abs/1963ZA.....57...47V

S.v. Hoerner,
Z.f.Astroph. 1960, 63

Siemens 2002
N=4,8,12,16 (4 Trx)

N=16,25 (40 Trx)

# History

***Supercomputer***
***JUGENE***
***IBM Blue Gene***
***At FZ Jülich,***
***Germany***


***Opening Ceremony June 2008***

# Computational Science...

...after von Neumann...

Exaflop/s?

Petaflop/s

Teraflop/s

Gigaflop/s

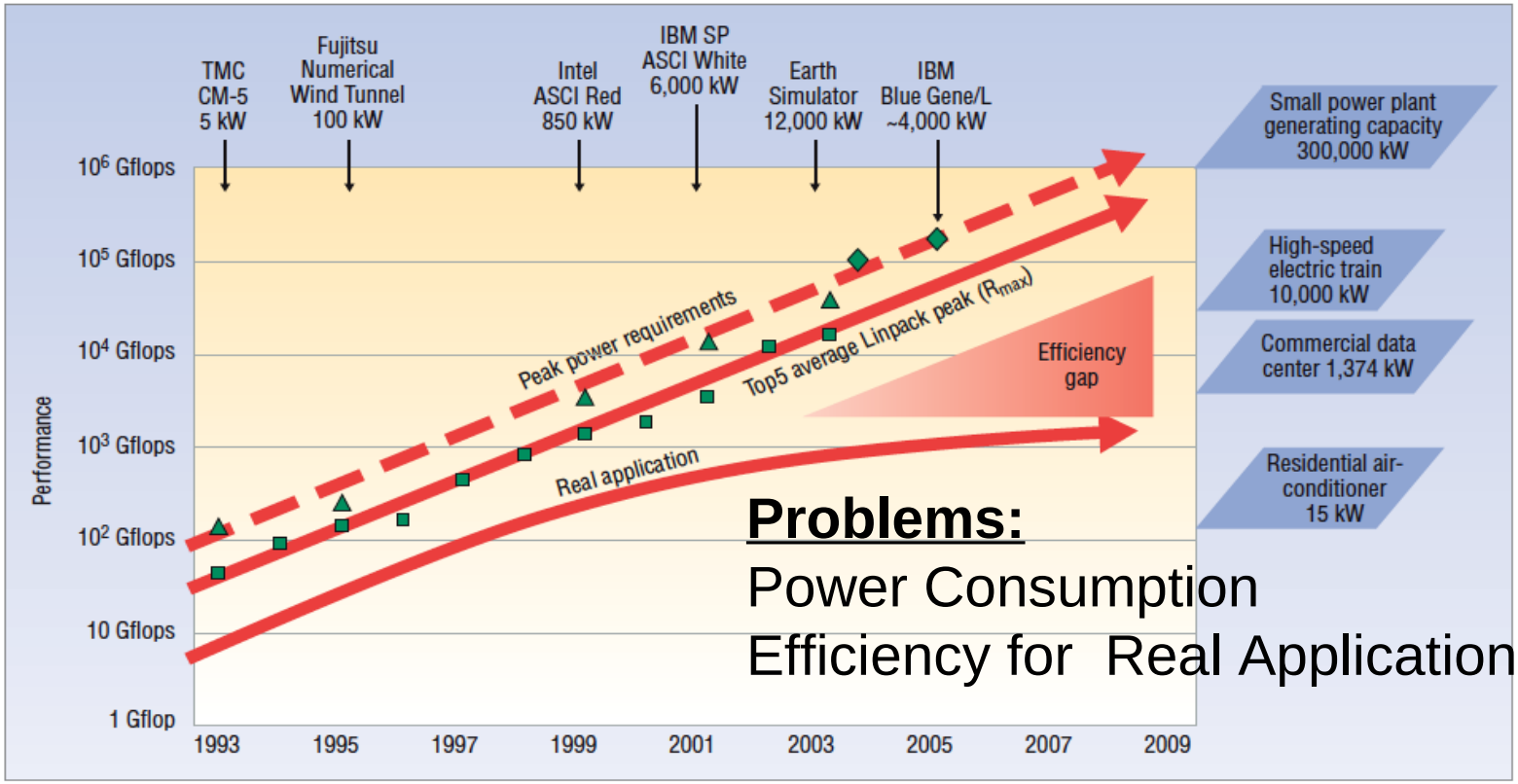



***Figure 1. Rising power requirements. Peak power consumption of the top supercomputers has steadily increased over the past 15 years.***

Thanks to Horst Simon, LBNL/NERSC for this diagram.

GPU Computing

# Special Hardware Accelerators

# SPECIAL HARDWARE

## CPUs

Central Processing Units

General Purpose oriented

1-12 Cores

Up to 4 pipes per core using Vector Units

Fully Programmable, many languages available

Very well studied

Max. 125W per processor

## GPUs

Graphic Processing Units

Graphics oriented

16-512 Cores

Massively Parallel Architecture, specialized instructions for parallel processing

Fully programmable, but limited languages

Algorithms not fully explored

Max. 400W per card

## FPGAs

Field Programmable Gate Arrays

Custom designs, best for processing streaming data

Programmable Logic, Architecture is custom-built for the required application

Requires extensive knowledge to program, development time is longer than CPUs and GPUs

Application interface is custom built on each case

Max. 60W per FPGA

## ASICs

Application Specific Integrated Circuits

Fully custom designs, built for a specific application

Not flexible, cannot be changed once it is built

Development is even more specialized than FPGAs

Power consumption varies with the application, usually best performance per Watt

Slide: Guillermo Marcus

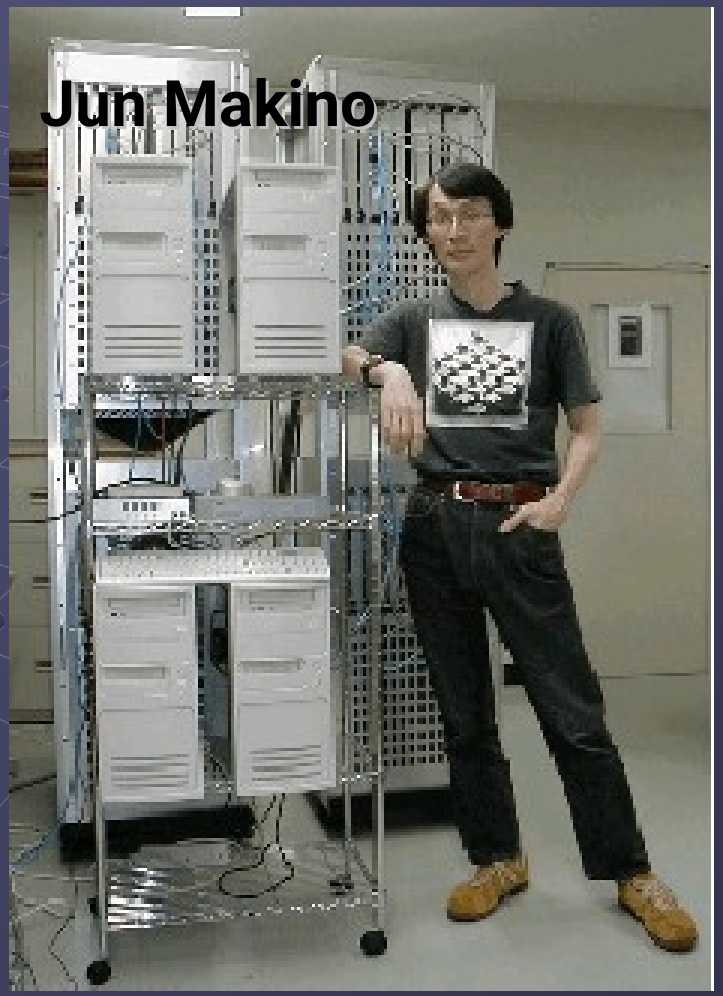
HITS

# HARDWARE

## GRAPE-6 Gravity/Coulomb Part

- G6 Chip: 0.25$\mu$ 2MGate ASIC, 6 Pipelines
- at 90MHz, 31Gflops/chip
- 48Tflops full system (March 2002)
- Plan up to 72Tflops full system (in 2002)
- Installed in Cambridge, Marseille, Drexel, Amsterdam, New York (AMNH), Mitaka (NAO), Tokyo, etc..

**New Jersey, Indiana, Heidelberg**

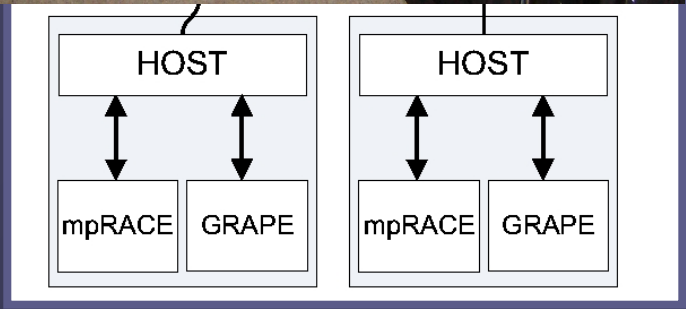


VolkswagenStiftung


## kepler1 Cluster 2006

4 Tflops (32 micro-GRAPE6)
Dual Port Infiniband
4 MPRACE-1 reconfigurable
(soon: 32 MPRACE-2)

GRAPE + MPRACE
= GRACE



# GPU: NAOC laohu cluster Beijing, China



Heidelberg

# Kepler GPU cluster

VolkswagenStiftung

Kepler GPU cluster

12 nodes = 12 x 16 = 192 CPU cores (@ 2 GHz)

12 x 64 GB = 768 GB RAM CPU memory

12 GPUs K20m = 12 x 2496 ~ 30k GPU threads

12 x 4.8 GB ~ 57 GB GPU device memory

4 x Xilinx Virtex-6 FPGA (ML 605)

since beg. 2013 operated.

Since 2019 two more nodes:
1x Quadro P6000
1x RTX 2080 Ti


# *NVIDIA Volta V100 GPU, 21 billion transistors, 5120 cores (now Ampere A100, 6920 cores, 9.7 Tflops DP, 19.5 SP)*

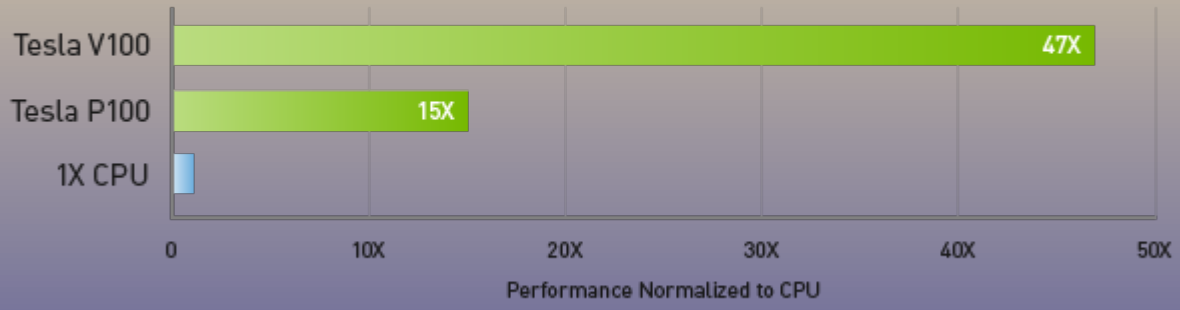
With NVLINK

Without NVLINK


| | | |
|---|---|---|
| PERFORMANCE with NVIDIA GPU Boost™ | DOUBLE-PRECISION 7.8 teraFLOPS | DOUBLE-PRECISION 7 teraFLOPS |
| | SINGLE-PRECISION 15.7 teraFLOPS | SINGLE-PRECISION 14 teraFLOPS |
| | DEEP LEARNING 125 teraFLOPS | DEEP LEARNING 112 teraFLOPS |
| INTERCONNECT BANDWIDTH Bi-Directional | NVLINK 300 GB/s | PCIE 32 GB/s |
| MEMORY CoWoS Stacked HBM2 | CAPACITY 32/16 GB HBM2 | |
| | BANDWIDTH 900 GB/s | |
| POWER Max Consumption | 300 WATTS | 250 WATTS |

# *NVIDIA Volta V100 GPU, 21 billion transistors, 5120 cores*

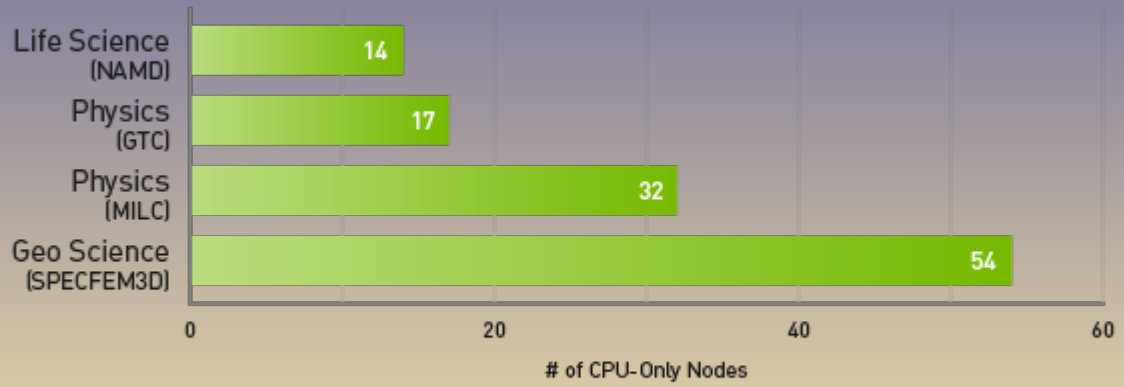
## 47X Higher Throughput Than CPU Server on Deep Learning Inference




Workload: ResNet-50 | CPU: 1X Xeon E5-2690v4 @ 2.6 GHz | GPU: Add 1X Tesla P100 or V100

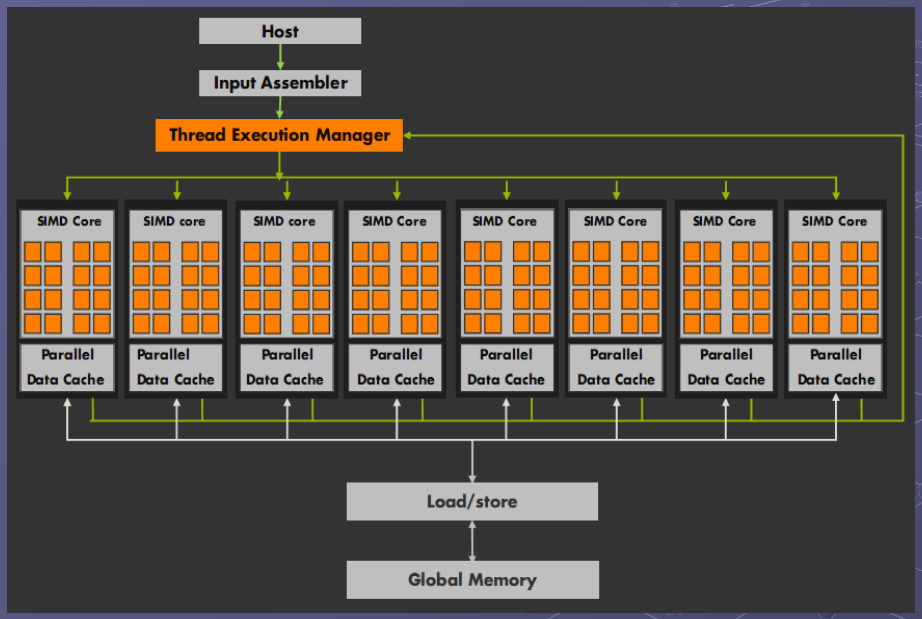
## 1 GPU Node Replaces Up To 54 CPU Nodes

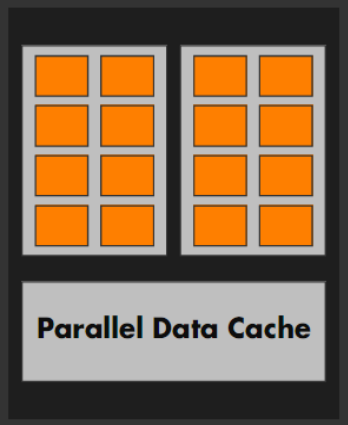
Node Replacement: HPC Mixed Workload




CPU Server: Dual Xeon Gold 6140@2.30GHz, GPU Servers: same CPU server w/ 4x V100 PCIe | CUDA Version: CUDA 9.x| Dataset: NAMD (STMV), GTC (mpi#proc.in), MILC (APEX Medium), SPECFEM3D (four_material_simple_model) | To arrive at CPU node equivalence, we use measured benchmark with up to 8 CPU nodes. Then we use linear scaling to scale beyond 8 nodes.

# Hardware around 2006




Each core

- 8 functional units
- SIMD 16/32 "warp"
- 8-10 stage pipeline
- Thread scheduler
- 128-512 threads/core
- 16 KB shared memory

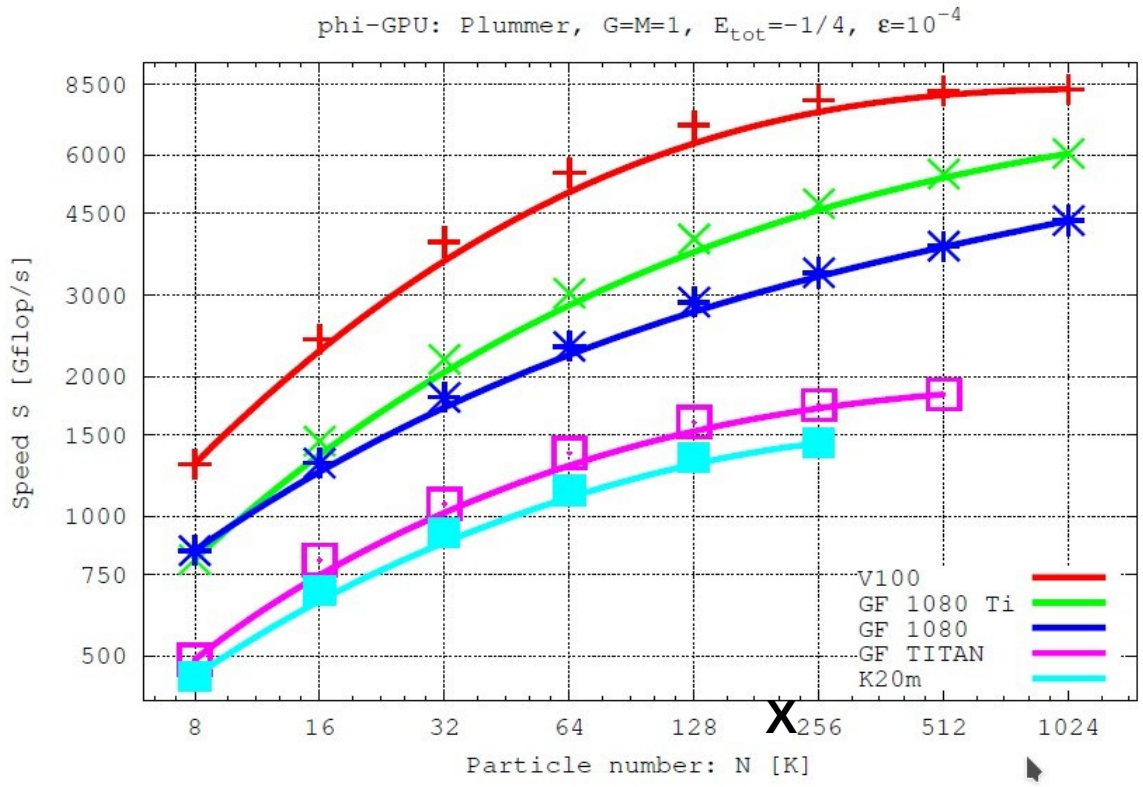
Total #threads/chip

16 * 512 = 8K

**GeForce 8800 GTX:**

575 MHz * 128 processors * 2 flop/inst * 2 inst/clock = 333 Gflops

# Kepler, Pascal, Volta, Scaling, it works...




**Volta V100**

**Pascal GF1080**

**Kepler K20m**

**Spurzem, Berczik, et al., 2013, LNCS Supercomputing, 2013, pp. 13-25, Springer. (updated unpublished)**

**Fig. 4.** Here we report a preliminary result from a benchmark test of our code on one Kepler K20 card; we compare with the performance on Fermi C2050 (used in the Mole-8.5 cluster), and the oldest Tesla C1060 GPU (used in the laohu cluster of 2009) - the latter is used as a normalization reference. We plot the speed ratio of our usual benchmarking simulation used in the previous figures, as a function of particle number. From this we see the sustained performance of a Kepler K20 would be about 1.4 - 1.5 Tflop/s.

**X = first GPU of laohu 2010**

# GPU Computing

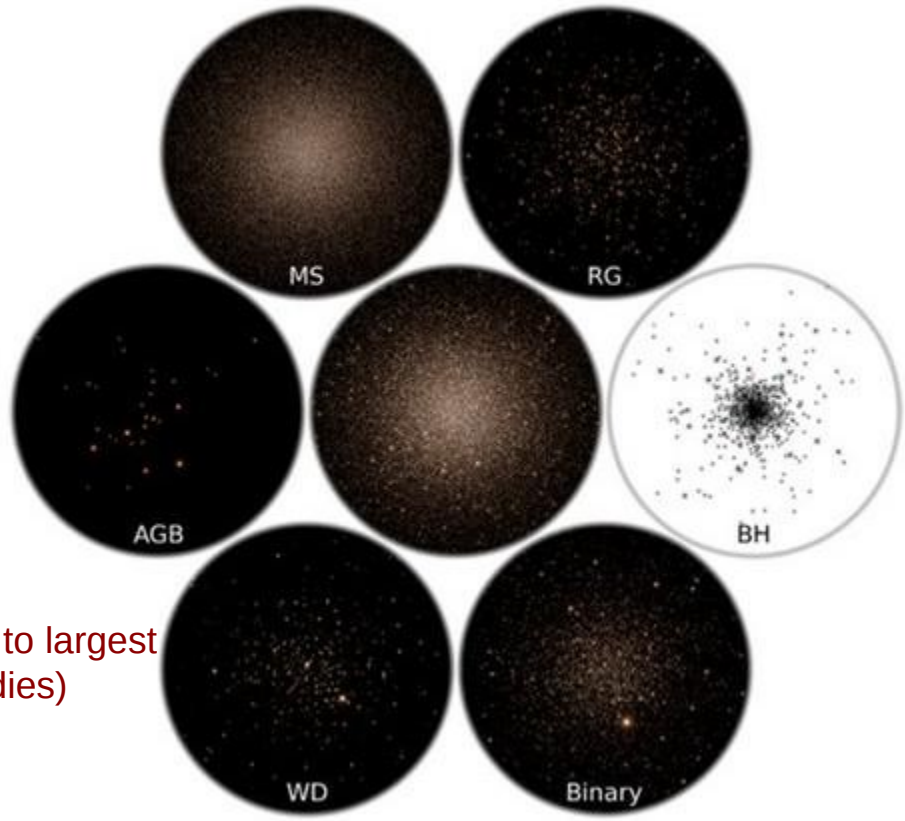
# Our NBODY6++GPU

# DRAGON Simulation

http://silkroad.bao.ac.cn/dragon/

***One million stars direct simulation,***

biggest and most realistic direct N-Body simulation of globular star clusters.
With stellar mass function, single and binary stellar evolution, regularization of close encounters, tidal field (NBODY6++GPU).
***(NAOC/Silk Road/MPA collaboration).***

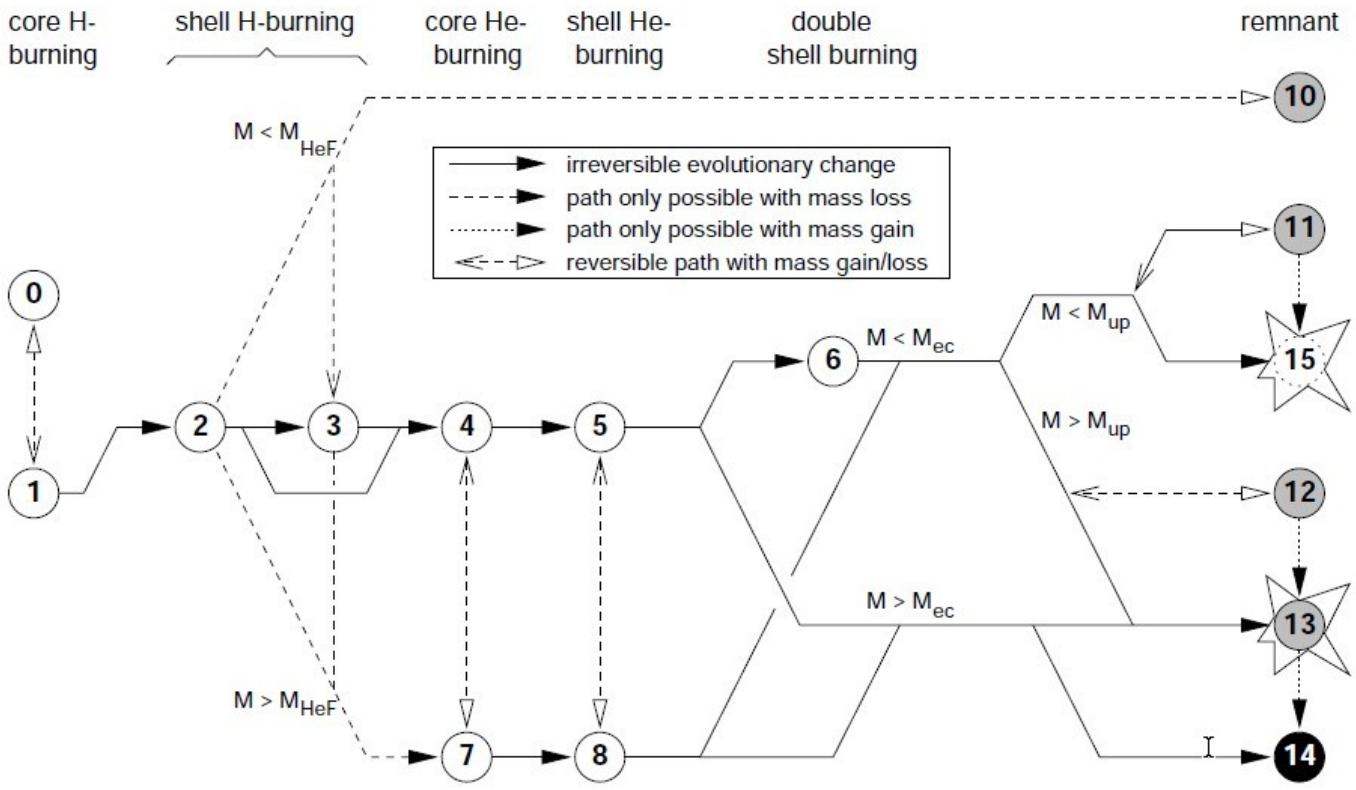



Wang, Spurzem, Aarseth, Naab et al.
MNRAS, 2015
Wang, Spurzem, Aarseth Naab, et al.
MNRAS 2016

Number of Floating Point Operations (~1M bodies) similar to largest Cosmological simulations (Millennium, Illustris, ~100M bodies)

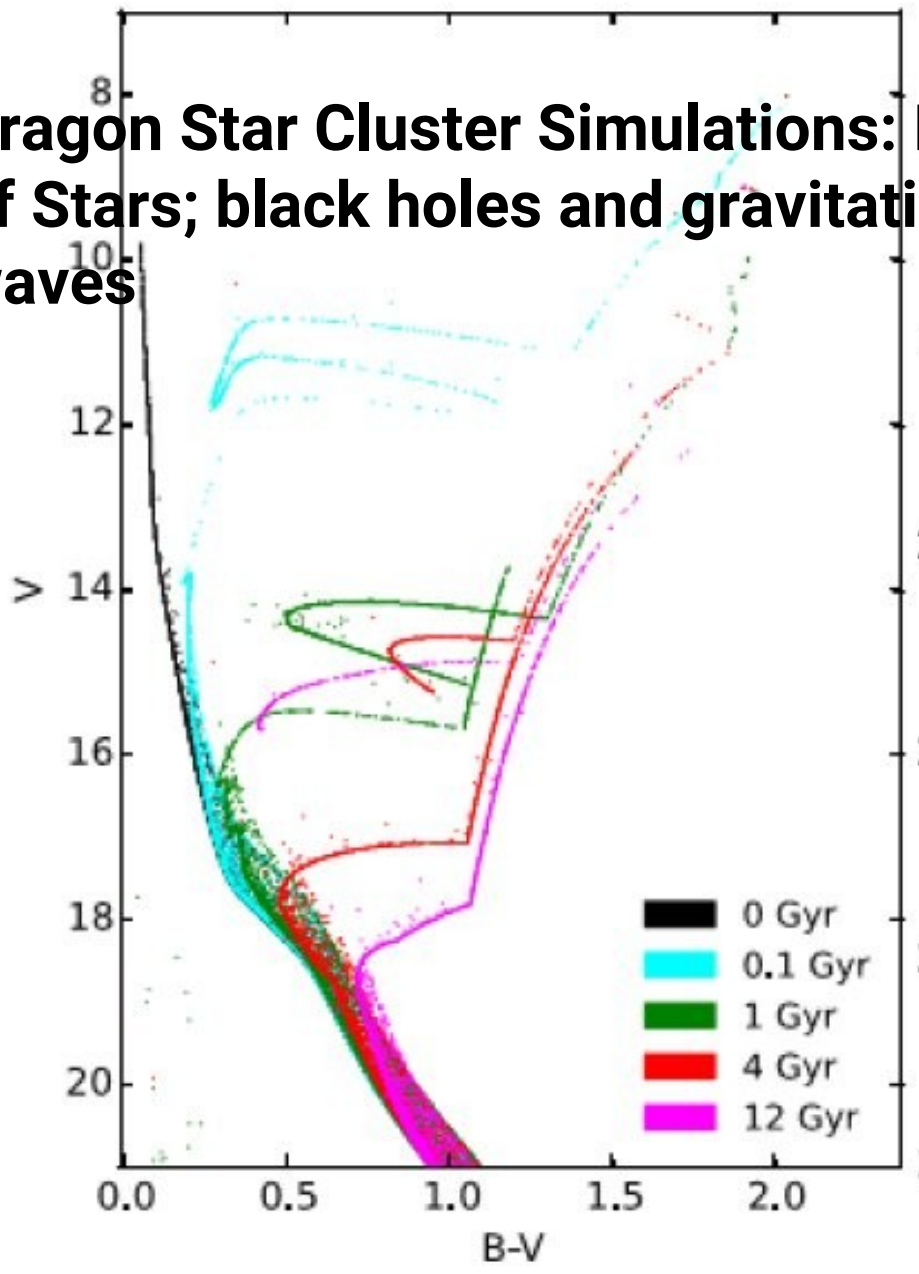
# Stellar Evolution in NBODY6++GPU

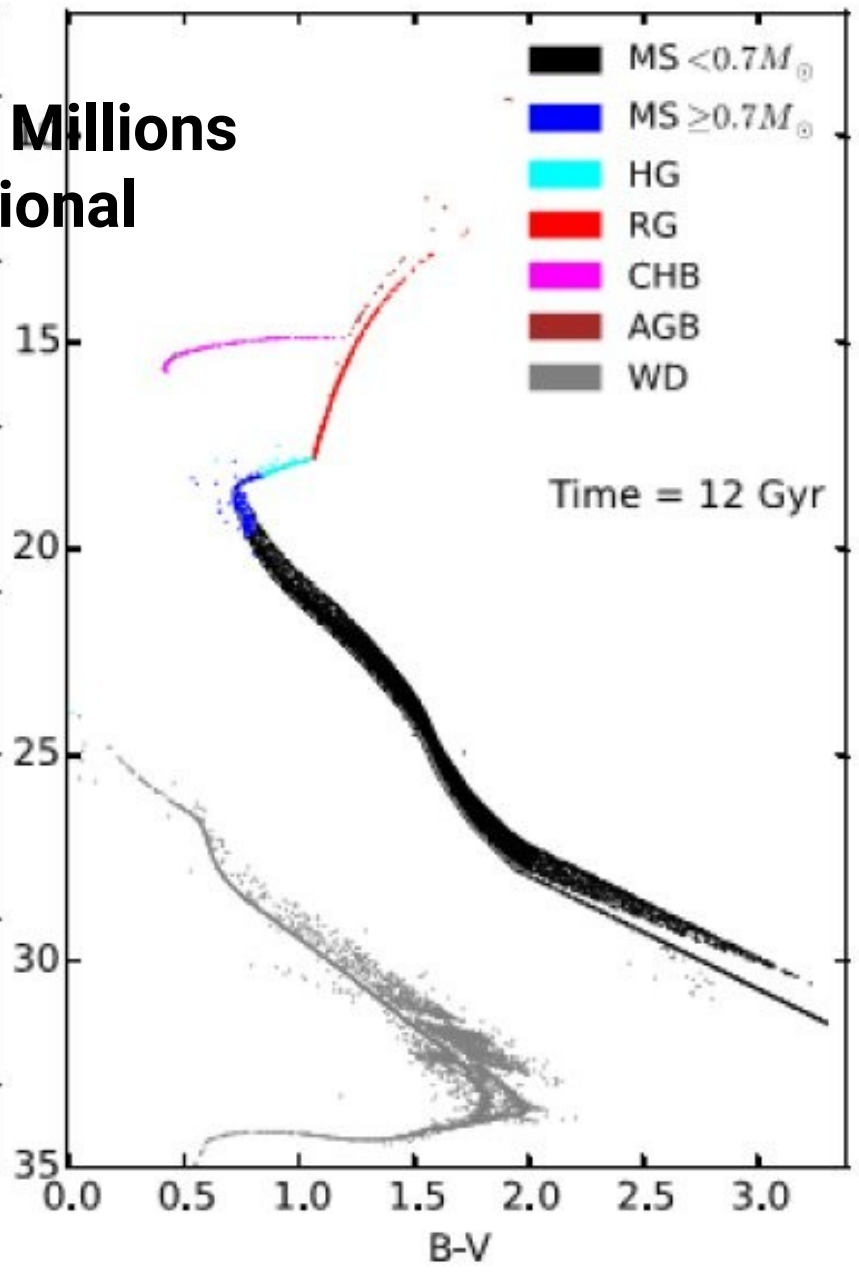
Hurley,
Ph.D. thesis




0 = main sequence M < 0.7 $M_\odot$
1 = main sequence M > 0.7 $M_\odot$
2 = Hertzsprung gap / subgiant
3 = first-ascent red giant
4 = horizontal branch / helium-burning giant
5 = early asymptotic giant / red supergiant
6 = thermally pulsating asymptotic giant
7 = naked helium main sequence
8 = naked helium (sub) giant
10 = helium white dwarf
11 = carbon/oxygen white dwarf
12 = oxygen/neon white dwarf
13 = neutron star
14 = black hole
15 = no stellar remnant

# Dragon Star Cluster Simulations: Millions of Stars; black holes and gravitational waves



Leiden

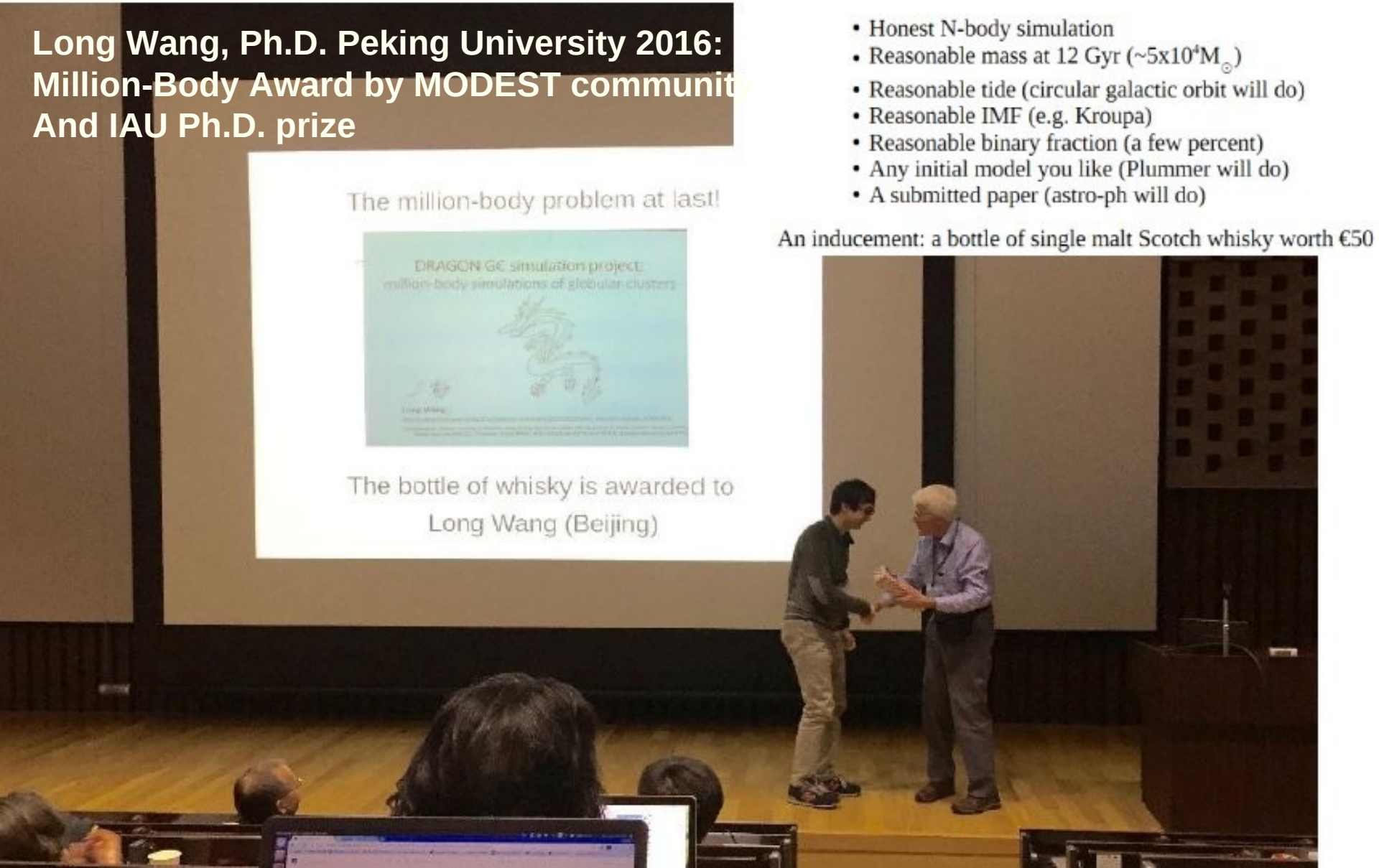
# CPU/GPU N-body6++

## Key Question 1. When will we see the first star-by-star *N*-body model of a globular cluster?

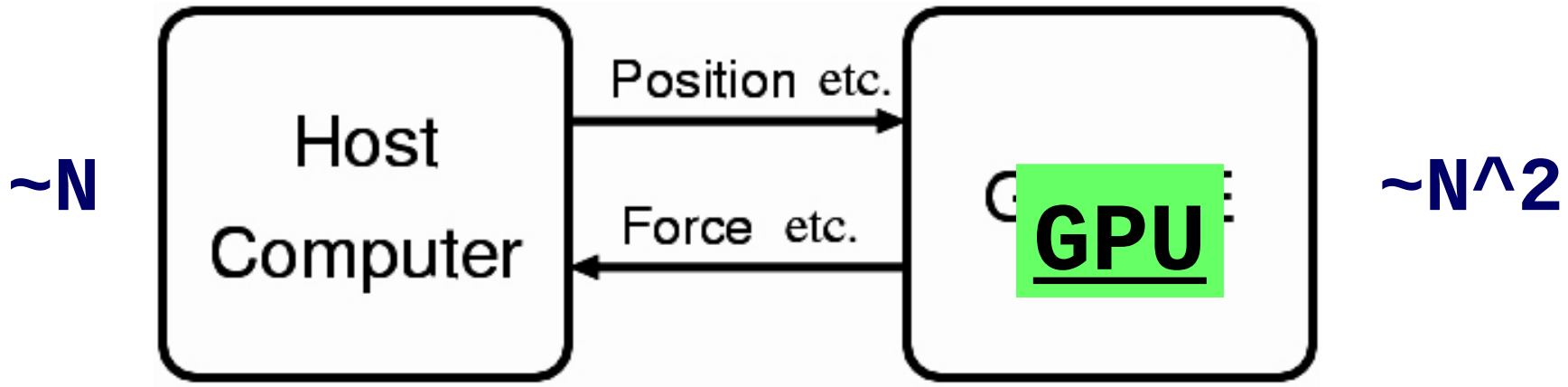
- Honest N-body simulation
- Reasonable mass at 12 Gyr (~$5x10^4 M_\odot$)
- Reasonable tide (circular galactic orbit will do)
- Reasonable IMF (e.g. Kroupa)
- Reasonable binary fraction (a few percent)
- Any initial model you like (Plummer will do)
- A submitted paper (astro-ph will do)

An inducement: a bottle of single malt Scotch whisky worth €50

**Long Wang, Ph.D. Peking University 2016:**
**Million-Body Award by MODEST communit**
**And IAU Ph.D. prize**



# Our own φGRAPE/GPU N-body code



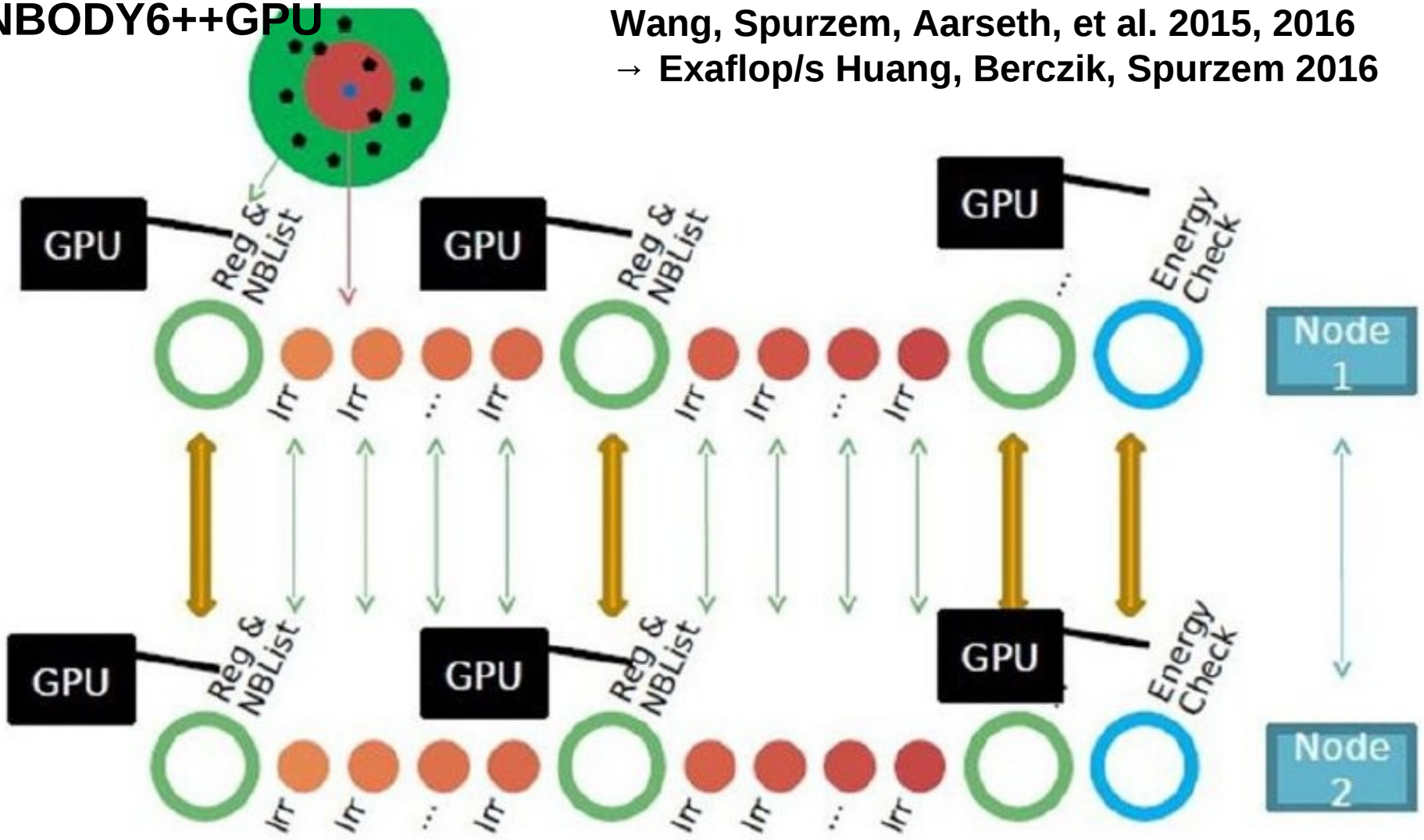

$$\vec{a}_i = \sum_{j=1;j\neq i}^{N} \vec{f}_{ij} \qquad \vec{f}_{ij} = -\frac{G \cdot m_j}{\left(r_{ij}^2 + \epsilon^2\right)^{3/2}} \vec{r}_{ij}$$

No softening for NBODY6++GPU

# Our CPU/GPU N-body (AC) code

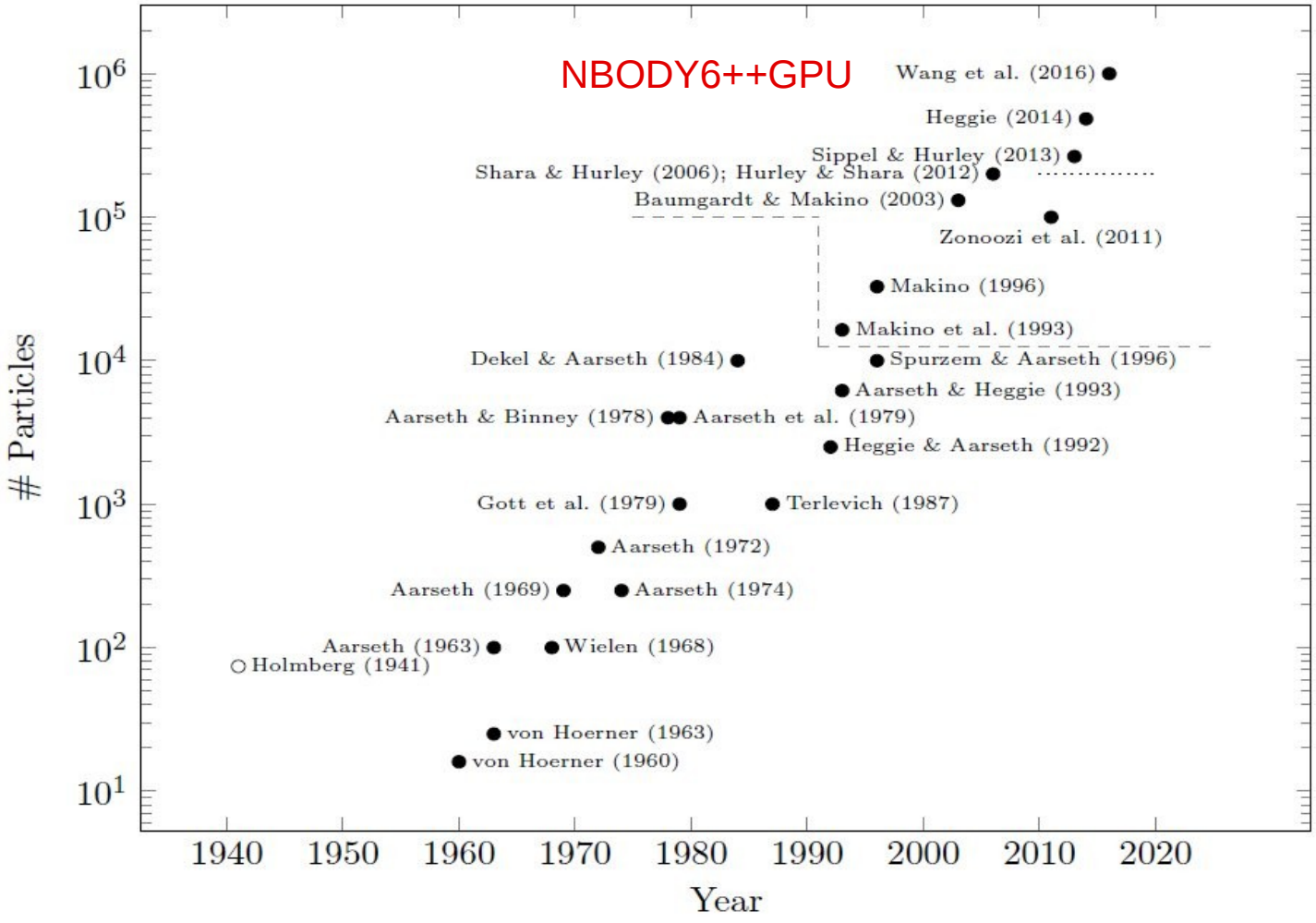
**NBODY6++GPU**

**Wang, Spurzem, Aarseth, et al. 2015, 2016**
**→ Exaflop/s Huang, Berczik, Spurzem 2016**




https://github.com/lwang-astro/betanb6pp

# “Moore's” Law for Direct N-Body




GRAPE/
GPU Clusters

GRAPE

Vector Computers

by D.C. Heggie Via added new cits. Sippel


**Table 1** Main components of NBODY6++

| Description | Timing variable | Expected scaling $N$ | Expected scaling $N_p$ | Fitting value [sec] |
|---|---|---|---|---|
| Regular force computation | $T_{\rm reg}$ | $\mathcal{O}(N_{\rm reg} \cdot N)$ | $\mathcal{O}(N_p^{-1})$ | $(2.2 \cdot 10^{-9} \cdot N^{2.11} + 10.43) \cdot N_p^{-1}$ |
| Irregular force computation | $T_{\rm irr}$ | $\mathcal{O}(N_{\rm irr} \cdot \langle N_{nb} \rangle)$ | $\mathcal{O}(N_p^{-1})$ | $(3.9 \cdot 10^{-7} \cdot N^{1.76} - 16.47) \cdot N_p^{-1}$ |
| Prediction | $T_{\rm pre}$ | $\mathcal{O}(N^{kn_p})$ | $\mathcal{O}(N_p^{-kp_p})$ | $(1.2 \cdot 10^{-6} \cdot N^{1.51} - 3.58) \cdot N_p^{-0.5}$ |
| Data moving | $T_{\rm mov}$ | $\mathcal{O}(N^{kn_{m1}})$ | $\mathcal{O}(1)$ | $2.5 \cdot 10^{-6} \cdot N^{1.29} - 0.28$ |
| MPI communication (regular) | $T_{\rm mcr}$ | $\mathcal{O}(N^{kn_{cr}})$ | $\mathcal{O}(kp_{cr} \cdot \frac{N_p-1}{N_p})$ | $(3.3 \cdot 10^{-6} \cdot N^{1.18} + 0.12)(1.5 \cdot \frac{N_p-1}{N_p})$ |
| MPI communication (irregular) | $T_{\rm mci}$ | $\mathcal{O}(N^{kn_{ci}})$ | $\mathcal{O}(kp_{ci} \cdot \frac{N_p-1}{N_p})$ | $(3.6 \cdot 10^{-7} \cdot N^{1.40} + 0.56)(1.5 \cdot \frac{N_p-1}{N_p})$ |
| Synchronization | $T_{\rm syn}$ | $\mathcal{O}(N^{kn_s})$ | $\mathcal{O}(N_p^{kp_s})$ | $(4.1 \cdot 10^{-8} \cdot N^{1.34} + 0.07) \cdot N_p$ |
| Sequential parts on host | $T_{\rm host}$ | $\mathcal{O}(N^{kn_h})$ | $\mathcal{O}(1)$ | $4.4 \cdot 10^{-7} \cdot N^{1.49} + 1.23$ |

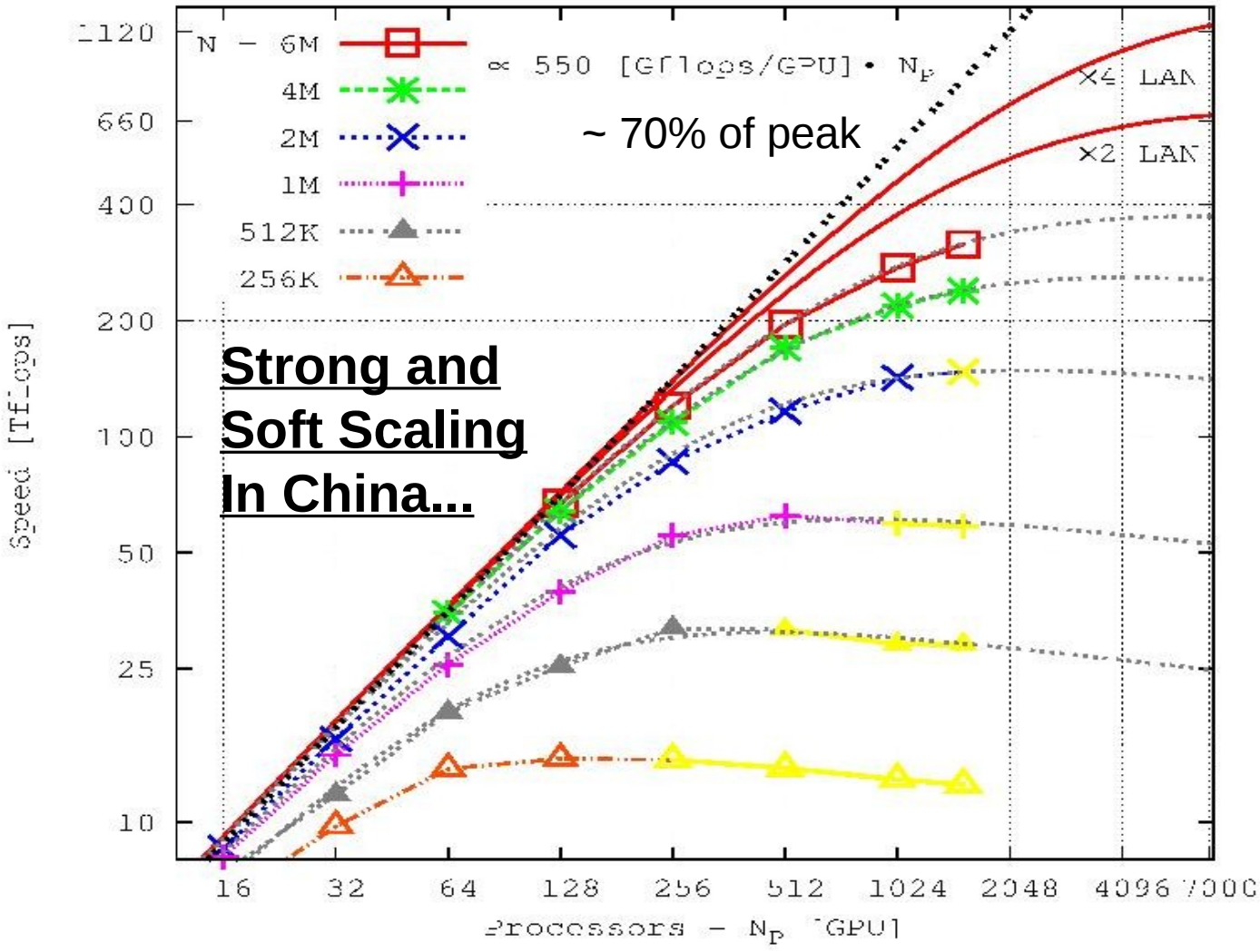
# ΦGPU – NBODY Code

中国科学院国家天文台
National Astronomical Observatories, CAS

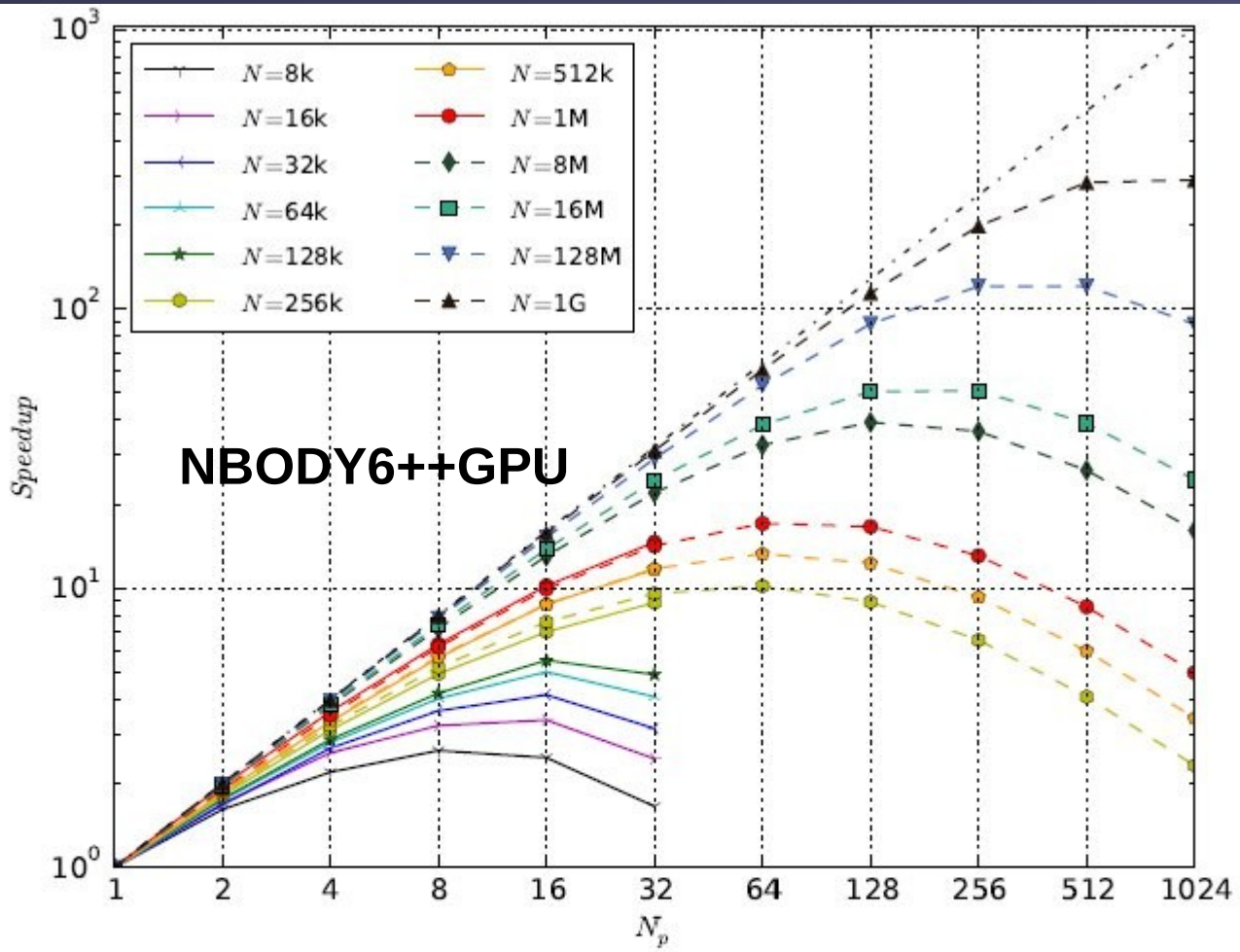
**350 Teraflop/s**
**1600 GPUs .**
**440 cores**
**= 704.000**
**GPU-Cores**

**Using**
**Mole-8.5**
**of**
**IPE/CAS**
**Beijing**

**Berczik et al.**
**2013**




中国科学院过程工程研究所
Institute Of Process Engineering, Chinese Academy Of Sciences



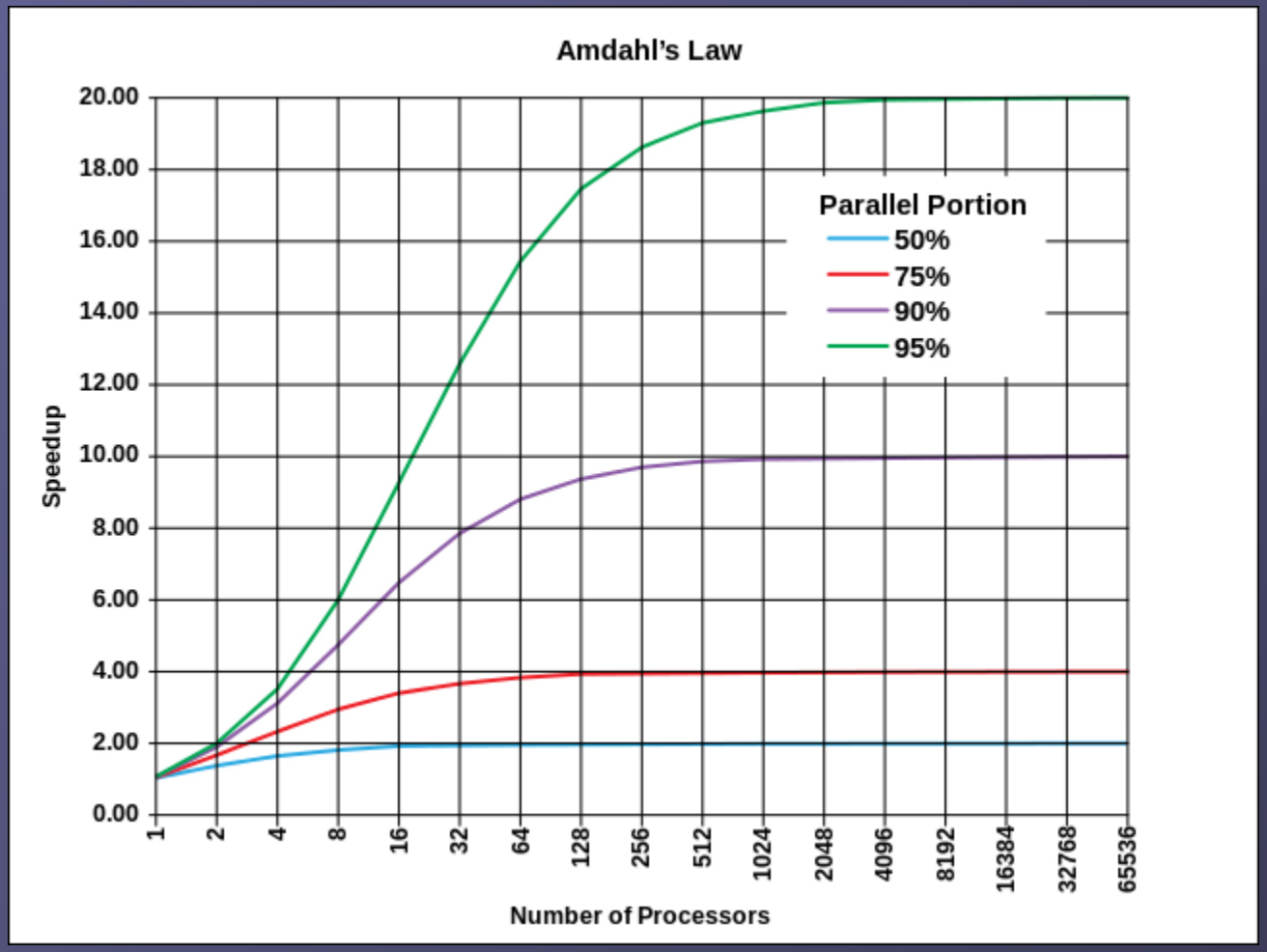

*Huang, Berczik, Spurzem, Res. Astron. Astroph. 2016, 16, 11.*

**Fig. 2** The speed-up ($S$) of NBODY6++ as a function of particle number ($N$) and processor number ($N_p$). Solid points are the measured speed-up ratio between sequential and parallel wall-clock time, dash lines predict the performance of larger scale simulations further. The symbols used in figure have the magnitudes: $1k = 1,024$, $1M = 1k^2$ and $1G = 1k^3$.

# Parallel Computing

## Some basic ideas

# Amdahl's Law (Gene Amdahl 1967)




Evolution according to Amdahl's law of the theoretical speedup of the execution of a program in function of the number of processors executing it, for different values of p. The speedup is limited by the serial part of the program. For example, if 95% of the program can be parallelized, the theoretical maximum speedup using parallel computing would be 20 times.

By Daniels220 at English Wikipedia - Own work based on: File:AmdahlsLaw.png, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=6678551

# Calculate Amdahl's Law:

Let X be the part of my program (in terms of computing time) which can be parallelised.
The sequential computing time $T_{seq}$ is normalized to unity (1), and can be expressed as:

$$T_{seq} = 1 = X + (1-X)$$

The parallel computing time Tpar under ideal conditions (ideal load balancing, ultrafast communication):

$$T_{par} = X/p + (1-X)$$

with processor number (core number) p ;
Then the speed-up of the program $S = T_{seq} / T_{par}$ :

$$S = 1 / (1-X+X/p) \quad ;$$

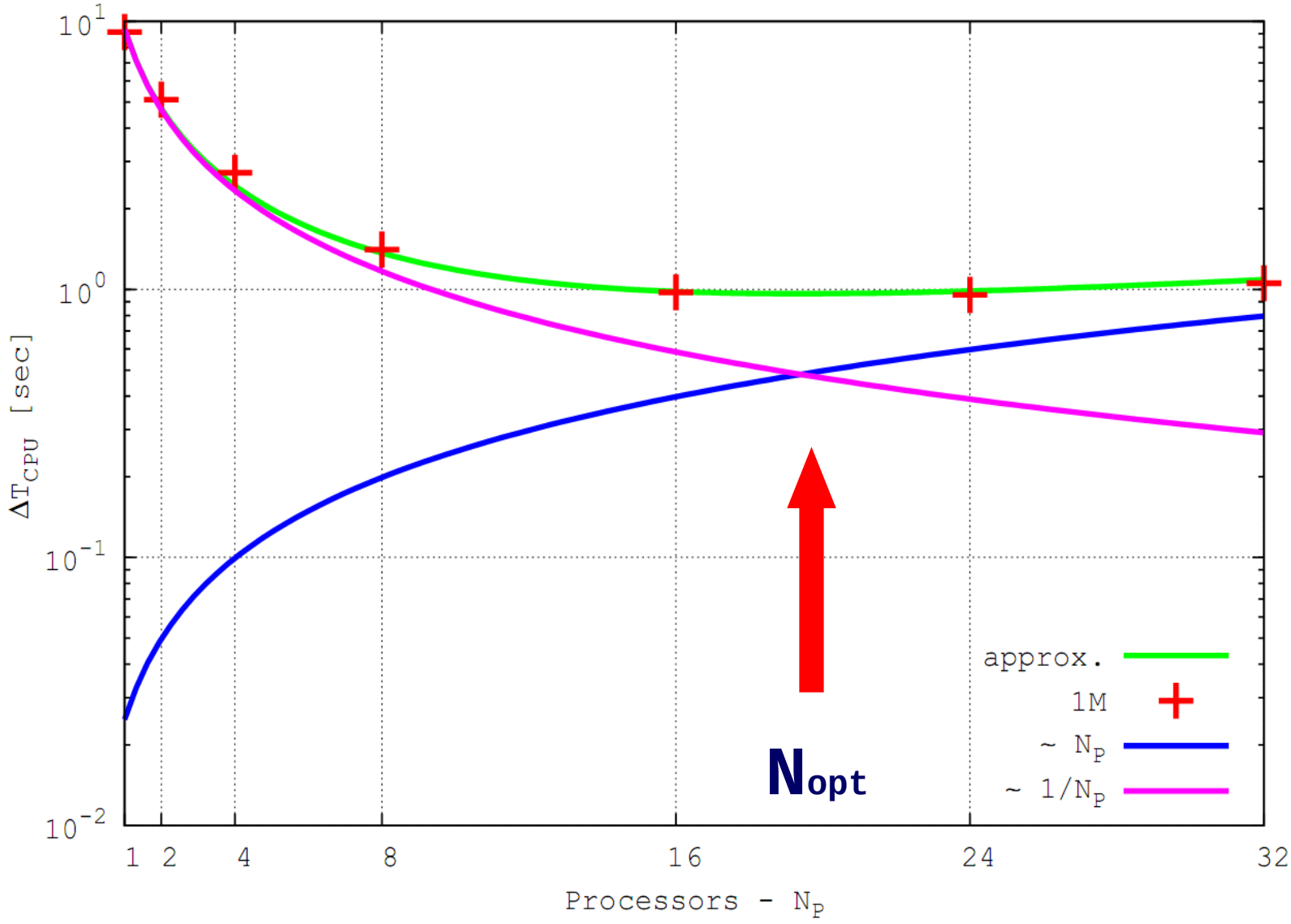
**Note: $T_{par}/T_{seq} = 1/S$ (sometimes also plotted)**

Note the limit of S for large p is very large: S = 1/(1-X). And if X ~ 1: S ~ p
With communication overhead:

$$T_{par} = X/p + (1-X) + T_{comm} \quad \rightarrow \quad S = 1 / (1-X+X/p+T_{comm})$$

If $T_{comm}$ independent of p we have for large p: $S = 1 / (1-X + T_{comm}) = \text{const.}$

# Parallel code on cluster



# Strong and Soft Scaling

- Strong Scaling: Fixed Problem size, increase p
- Soft Scaling: Increase Problem size, increase p (constant amount of work per processing element)

Ansatz for Soft Scaling:

- $T_{seq} = p\,(X + (1-X))$
- $T_{par} = X + p\,(1-X)$
- $S = T_{seq}/T_{par} = p\ /\ (X+p\,(1-X))$

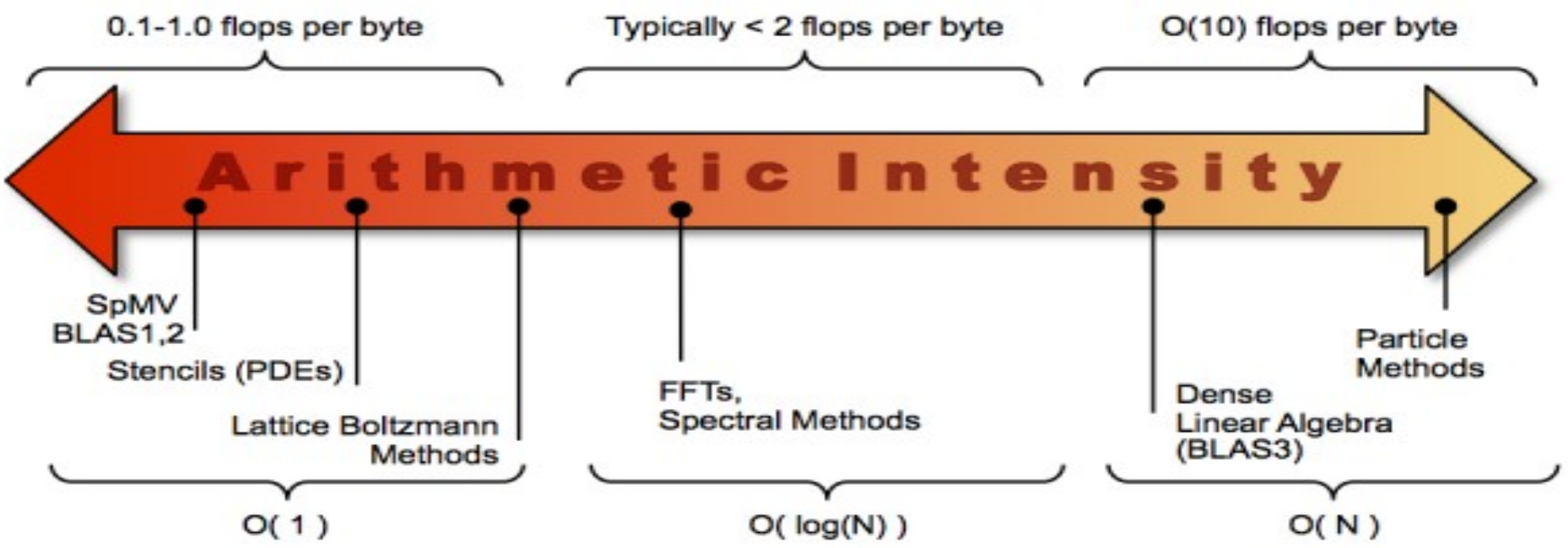
**If $X \sim 1$: $S = p$ ; $T_{par} = X = \text{const.}$**

# Roofline Performance Model (LBL)

http://crd.lbl.gov/departments/computer-science/PAR/research/roofline

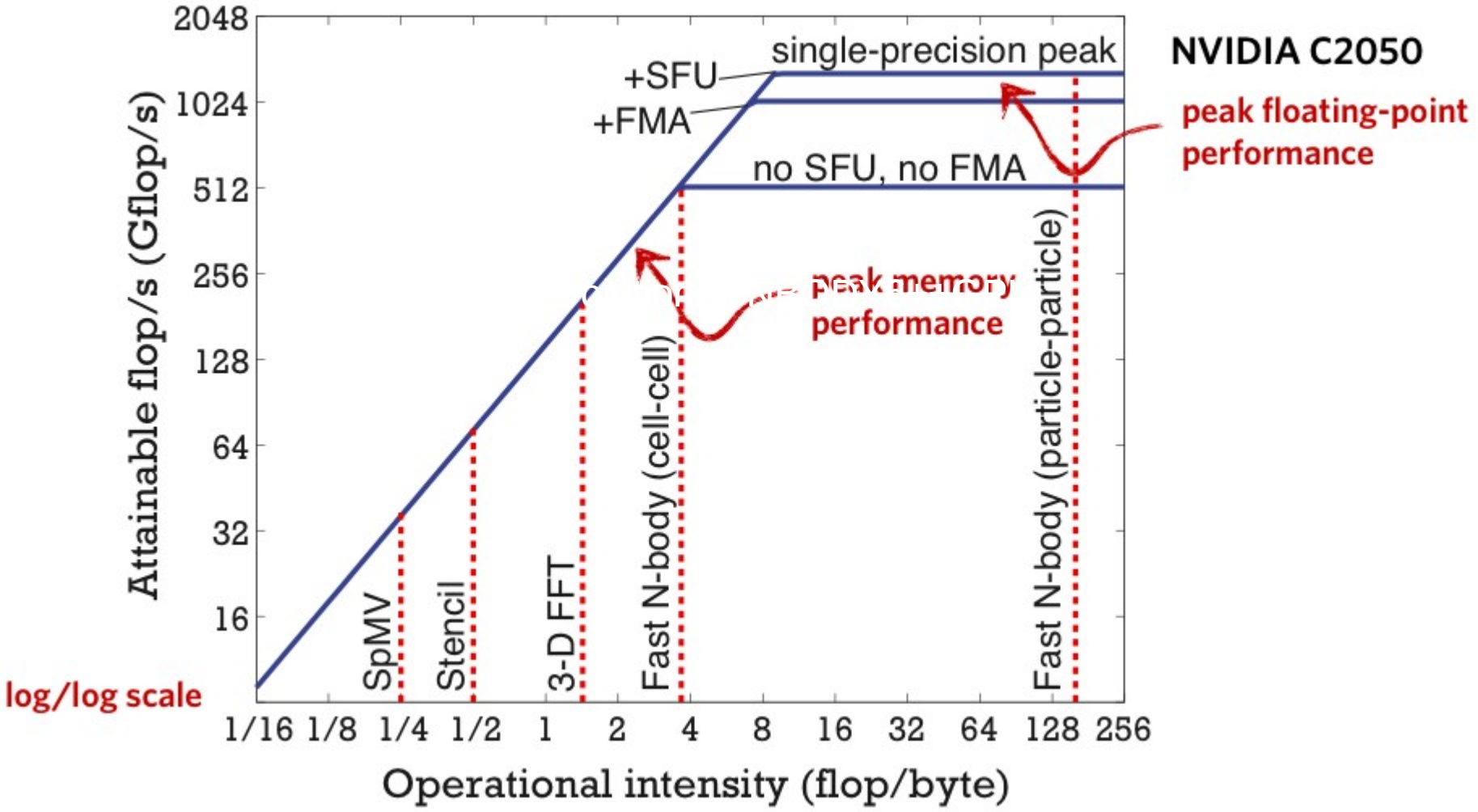
## Arithmetic Intensity

The core parameter behind the Roofline model is Arithmetic Intensity. Arithmetic Intensity is the ratio of total floating-point operations to total data movement (bytes).



# Roofline Performance Model (LBL)

http:///lorenabarba.com/wp-content/uploads/2012/01/roofline_slide.png



# More on Parallel Computing

Wen-Mei Hwu,
Univ. of Illinois
At Urbana-Champaign

# Additional deeper material:

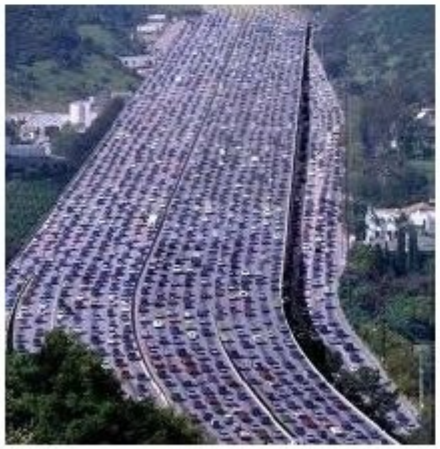
Lectures by Prof. Wen-Mei Hwu Chicago in Berkeley 2012 and Beijing 2013, see http://iccs.lbl.gov/workshops/tutorials.html
(down on page links to all lecture files, also available on request from spurzem@nao.cas.cn)

Lecture1: Computational thinking
Lecture2: Parallelism Scalability
Lecture3: Blocking Tiling
Lecture4: Coarsening Tiling
Lecture5: Data Optimization
Lecture6: Input Binning
Lecture7: Input Compaction
Lecture8: Privatization

See also:
http://freevideolectures.com/Course/2880/Advanced-algorithmic-techniques-for-GPUs/1

# Massive Parallelism - Regularity






©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# Main Hurdles to Overcome

- Serialization due to conflicting use of critical resources
- Over subscription of Global Memory bandwidth
- Load imbalance among parallel threads

©Wen-mei W. Hwu and David Kirk/NVIDIA, Berkeley, January 24-25, 2011

# Computational Thinking Skills

- The ability to translate/formulate domain problems into computational models that can be solved efficiently by available computing resources
  - Understanding the relationship between the domain problem and the computational models
  - **Understanding the strength and limitations of the computing devices**
  - **Defining problems and models to enable efficient computational solutions**

©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# DATA ACCESS CONFLICTS

©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# Conflicting Data Accesses Cause Serialization and Delays

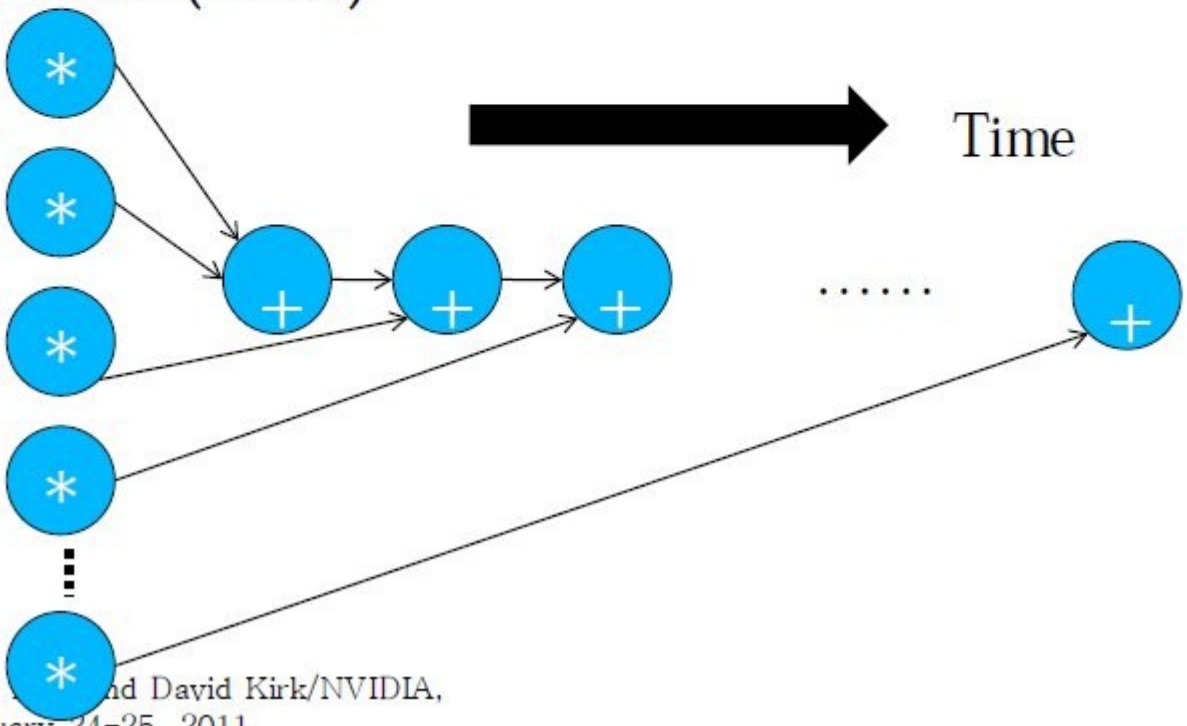
- Massively parallel execution cannot afford serialization

- Contentions in accessing critical data causes serialization


©Wen-mei W. Hwu and David Kirk/NVIDIA, Berkeley, January 24-25, 2011

# A Simple Example

- A naïve inner product algorithm of two vectors of one million elements each
  - All multiplications can be done in time unit (parallel)
  - Additions to a single accumulator in one million time units (serial)



# How much can conflicts hurt?

- Amdahl's Law
  - If fraction X of a computation is serialized, the speedup can not be more than 1/(1-X)
- In the previous example, X = 50%
  - Half the calculations are serialized
  - No more than 2X speedup, no matter how many computing cores are used

©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# GLOBAL MEMORY BANDWIDTH

©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# Global Memory Bandwidth

**Ideal**

**Reality**

©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# Global Memory Bandwidth

- Many-core processors have limited off-chip memory access bandwidth compared to peak compute throughput
- Fermi
  - 1 TFLOPS SPFP peak throughput
  - 0.5 TFLOPS DPFP peak throughput
  - 144 GB/s peak off-chip memory access bandwidth
    - 36 G SPFP operands per second
    - 18 G DPFP operands per second
  - To achieve peak throughput, a program must perform 1,000/36 = ~28 SPFP (14 DPFP) arithmetic operations for each operand value fetched from off-chip memory



©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# LOAD BALANCE

©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# Load Balance

- The total amount of time to complete a parallel job is limited by the thread that takes the longest to finish


©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

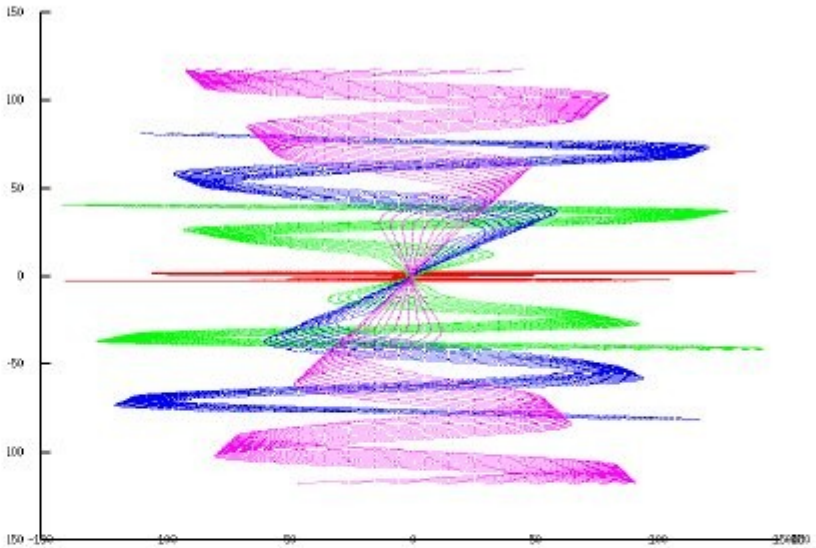
# How bad can it be?

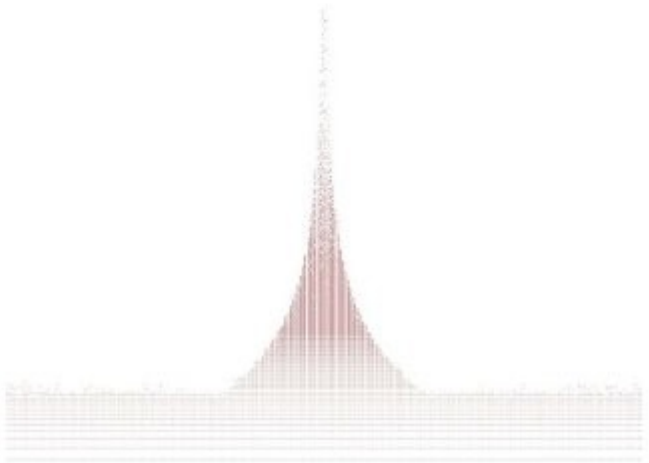
- Assume that a job takes 100 units of time for one person to finish
  - If we break up the job into 10 parts of 10 units each and have fo10 people to do it in parallel, we can get a 10X speedup
  - If we break up the job into 50, 10, 5, 5, 5, 5, 5, 5, 5, 5 units, the same 10 people will take 50 units to finish, with 9 of them idling for most of the time. We will get no more than 2X speedup.

©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# How does imbalance come about?

- Non-uniform data distributions
  - Highly concentrated spatial data areas
  - Astronomy, medical imaging, computer vision, rendering, …
- If each thread processes the input data of a given spatial volume unit, some will do a lot more work than others





© B

# Eight Algorithmic Techniques (so far)

| Technique | Contention | Bandwidth | Locality | Efficiency | Load Imbalance | CPU Leveraging |
|---|---|---|---|---|---|---|
| Tiling | | X | X | | | |
| Privatization | X | | X | | | |
| Regularization | | | | X | X | X |
| Compaction | | X | | | | |
| Binning | | X | X | X | | X |
| Data Layout Transformation | X | | X | | | |
| Thread Coarsening | X | X | X | X | | |
| Scatter to Gather Conversion | X | | | | | |

http://courses.engr.illinois.edu/ece598/hk/

©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011

# You can do it.

- Computational thinking is not as hard as you may think it is.
  - Most techniques have been explained, if at all, at the level of computer experts.
  - The purpose of the course is to make them accessible to domain scientists and engineers.

©Wen-mei W. Hwu and David Kirk/NVIDIA, Berkeley, January 24-25, 2011

# ANY MORE QUESTIONS?

©Wen-mei W. Hwu and David Kirk/NVIDIA,
Berkeley, January 24-25, 2011