



計算科学が拓く世界

スーパーコンピュータは 何故スーパーか

学術情報メディアセンター

中島 浩

<http://www.para.media.kyoto-u.ac.jp/jp/>
username=super password=computer



講義の概要

■ 目的

- 計算科学に不可欠の道具**スーパーコンピュータ**が
 - どういうものか
 - なぜスーパーなのか
 - どう使うとスーパーなのかについて**雰囲気**をつかむ

■ 内容

- スーパーコンピュータの歴史を概観しつつ
- スーパーである基本原理を知り
- どういう計算が得意であるかを学んで
- それについて**レポート**を書く



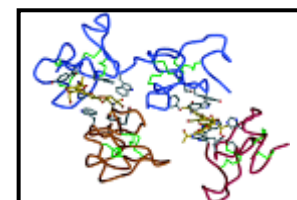
スーパーコンピュータ (スパコン) とは (1)

- パソコンの数千倍～数万倍の規模・性能を持つ
巨大な超高速コンピュータ
 - 世界最大・最高速マシン ≒パソコン × 10万
 - 京大スーパーコンピュータ ≒パソコン × 5千→パソコンで丸1日かかる計算 = 1秒～数10秒
(ただしスパコン向きの問題をうまくプログラムしたら)
- スパコンが高速な理由
 - 個々の部品(CPU, メモリなど)≒パソコン
 - 非常に多数のパソコン(のようなもの)の集合体
 - パソコン = 1~4 CPU
 - 京大スパコン = 6,656 CPU
 - 世界最高速スパコン = 663,552 CPU
 - 世界最大規模スパコン = 663,552 CPU



スーパーコンピュータ (スパコン) とは (2)

- スパコンが得意な計算 = 大量CPUによる分担計算
= 超大量のデータを対象とする計算
 - 地球全体の気象・気候・海洋現象の予測
 - 1km^2 あたり1データ
 - データ数 = **5億** (× 高さ方向)
 - 生体物質・化学物質・材料の解析
 - **膨大**な分子・原子数
 - (e.g. 水 $1\text{ml} = 3.3 \times 1$ 兆 $\times 100$ 億)
 - 自動車の空力・衝突解析
 - 1mm^2 or 1cm^3 あたり1データ
 - データ数 = **1~10億**
 - Web 文書の解析
 - (← 自動翻訳用データ作成など)
 - 文書数 = **数億~数10億**



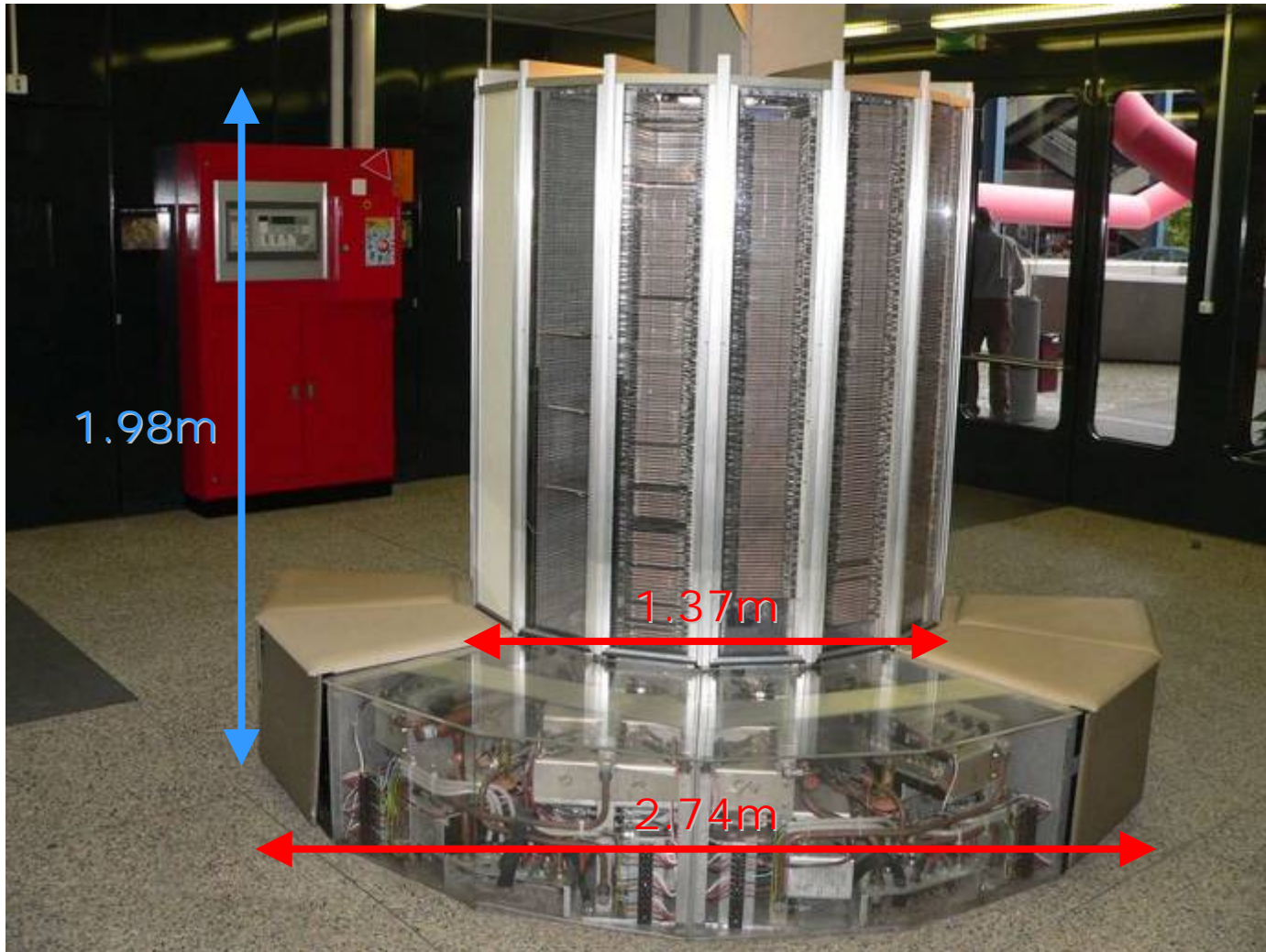


そもそもの始まり:ベクトルマシン (1)

- 1976年:最初のスパコンCray-1登場
 - 動作周波数=80MHz (< 携帯電話)
 - 演算性能=160MFlops (< 携帯電話)
 - Flops: Floating-point Operation Per Second
= 10進16桁精度の数値($10^{-308} \sim 10^{308}$)の加減乗算回数/秒
→ 160MFlops = 毎秒1億6千万回の加減乗算
 - 消費電力=115kW
 - 大量の数値データ(ベクトル)に対する同種演算が得意
- 1976年(中島=20歳)での「スーパー」度
 - 最速@京大(富士通 F230-75) < 5MFlops
 - 最速@京大情報工学科(日立 H8350) < 1MFlops
 - Intel 8086/87(1978/80) \approx 50KFlops



そもそもの始まり: ベクトルマシン (2)



source: <http://en.wikipedia.org/wiki/Image:Cray-1-p1010221.jpg>



ベクトル計算の原理 (1)

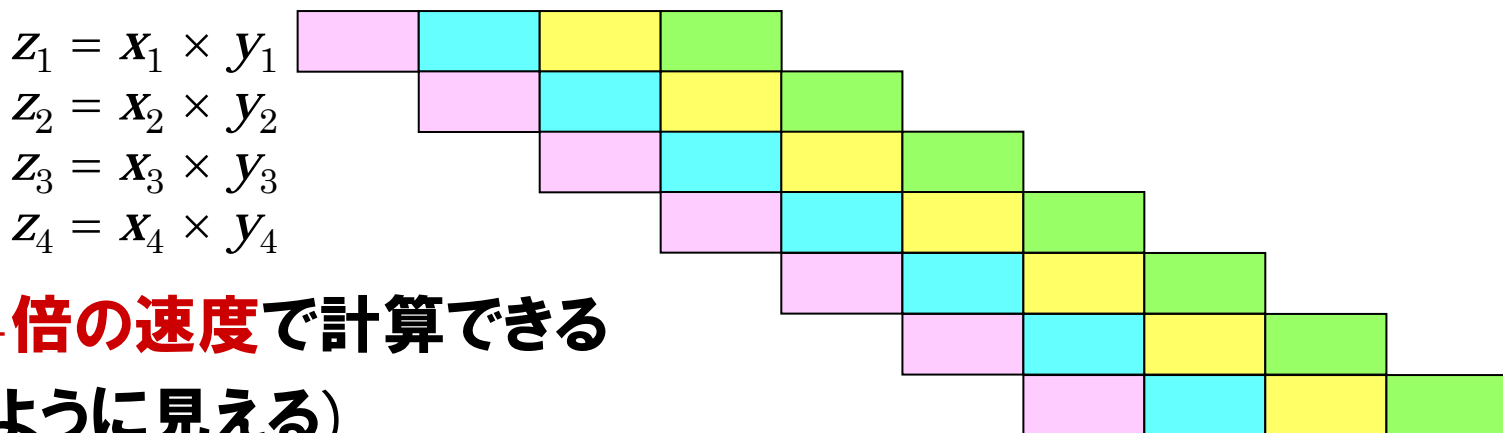
■ 大量数値データの同種演算を高速に行う方法

例: $z_i = x_i \times y_i$ ($i = 1, 2, \dots$)

- 1つの乗算をいくつか(たとえば4つ)の小さい操作に分ける

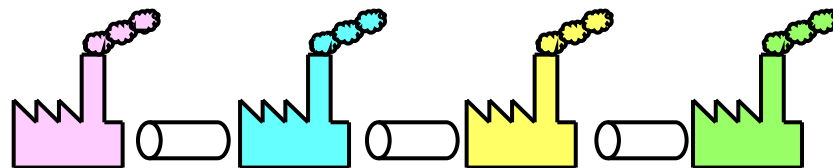
$$z_i = x_i \times y_i$$


- 多数の乗算を1小操作ずつずらして行う



→ **4倍の速度**で計算できる
(ように見える)

→ (演算) **パイプライン処理**

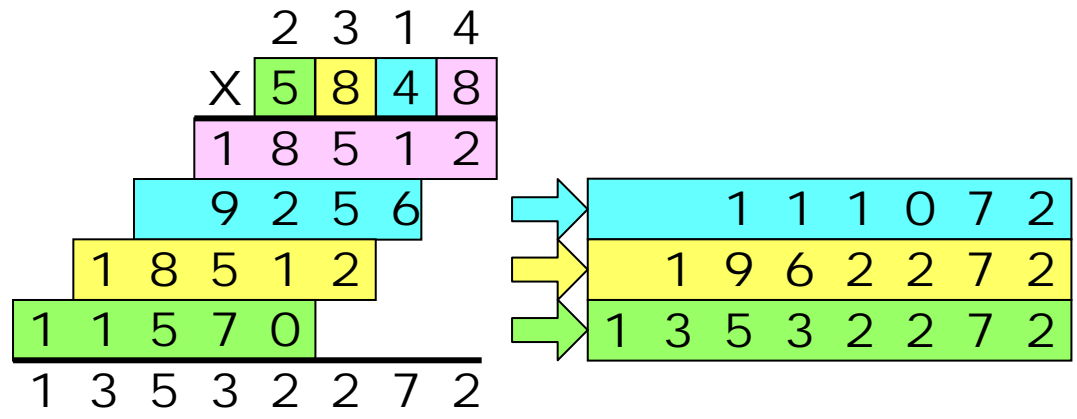




少し寄り道:スーパーコンピュータの原理

ベクトル計算の原理 (2)

- 乗算を4分割して**ずらす**考え方(たとえ話 ≠ 真実)



$2314 \times 5848 = 13532272$

$2314 \times 8 + 2314 \times 40 + 2314 \times 800 + 2314 \times 5000$

$7872 \times 6752 = 53151744$

$7872 \times 2 + 7872 \times 50 + 7872 \times 700 + 7872 \times 6000$

$1778 \times 7142 = 12698476$

$1778 \times 2 + 1778 \times 40 + 1778 \times 100 + 1778 \times 7000$

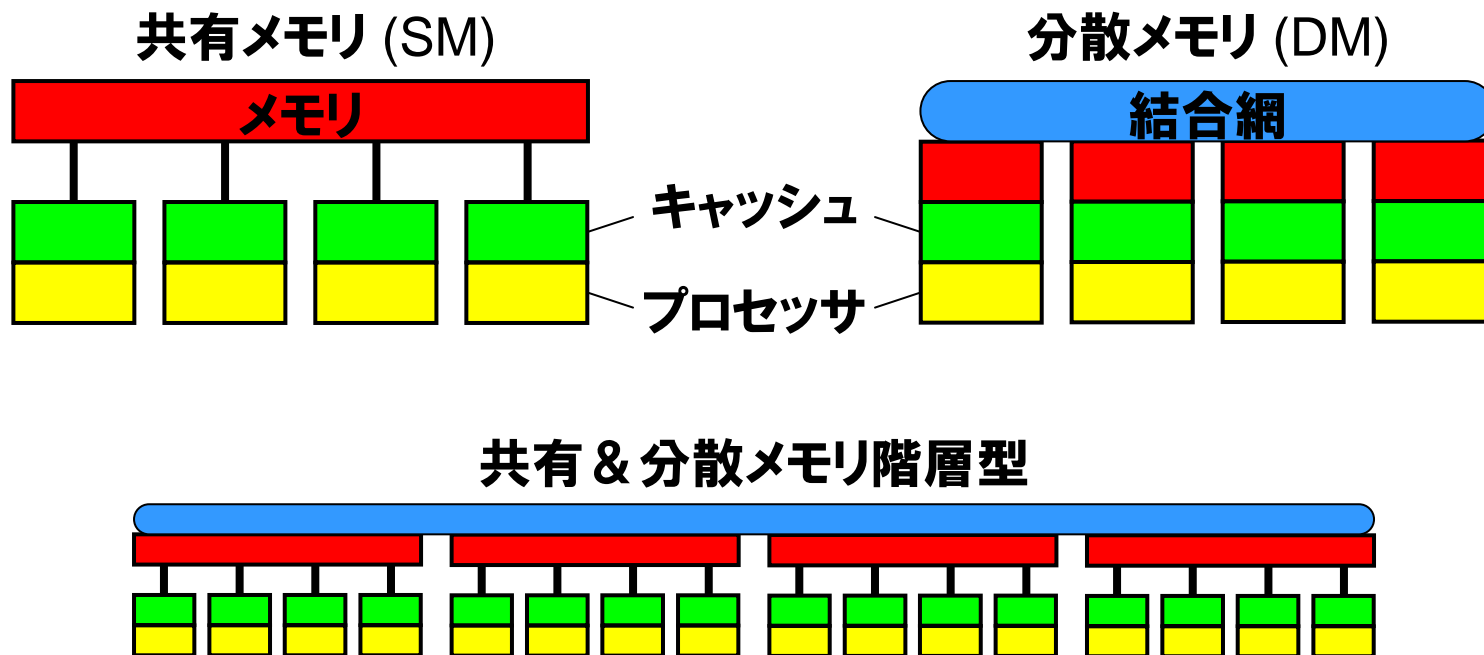
$8485 \times 1651 = 13843635$

$8385 \times 1 + 8385 \times 50 + 8385 \times 600 + 8385 \times 1000$



もう一つの方法: 並列マシン(1)

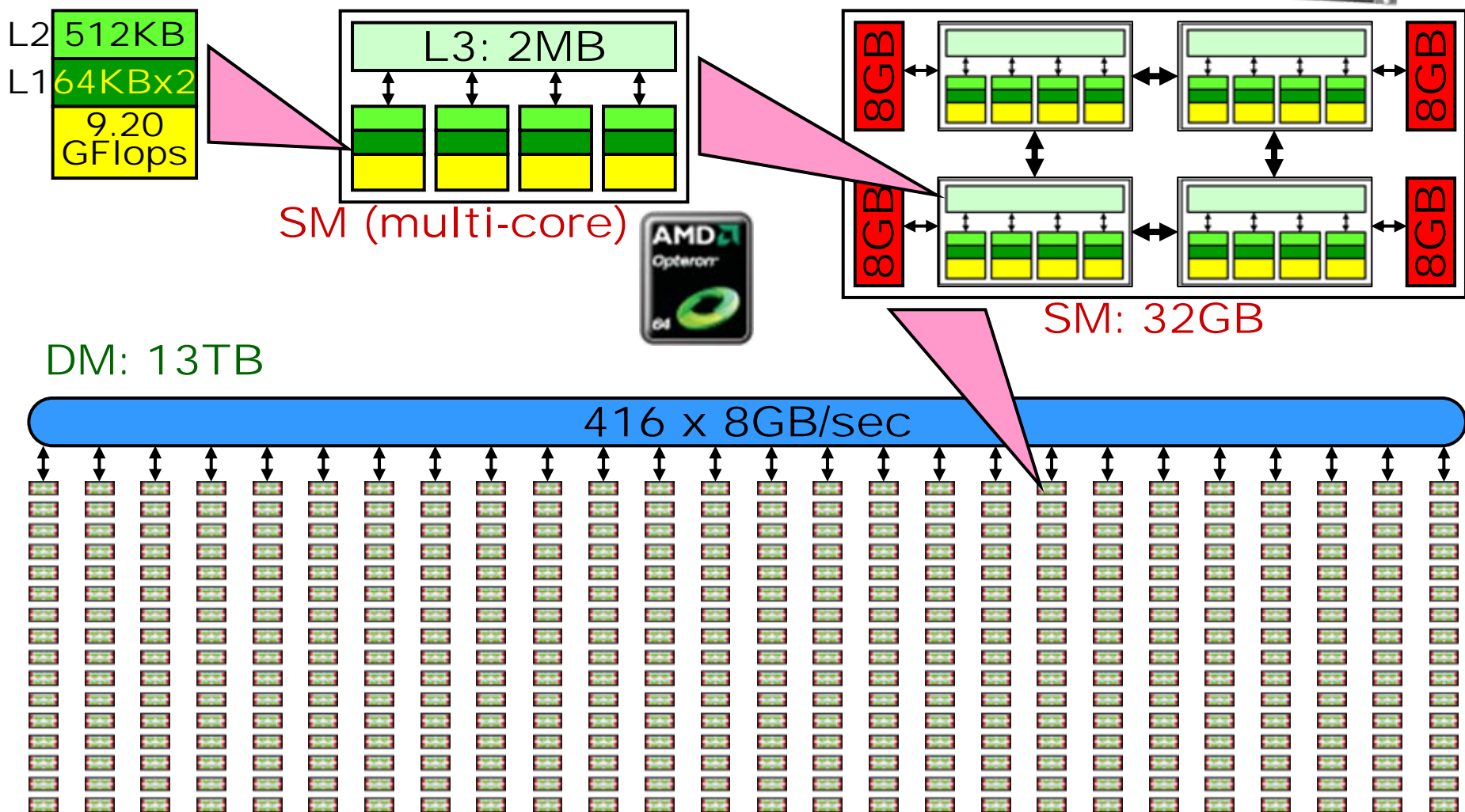
- 1980年代: **スカラーマルチプロセッサ台頭**
 - 多数のパソコン(のようなもの)の集合体
 - Sequent Balance : 20 x NS32016 ('84)
 - Intel iPSC/1: 128 x i80286 ('85)





もう一つの方法: 並列マシン(2)

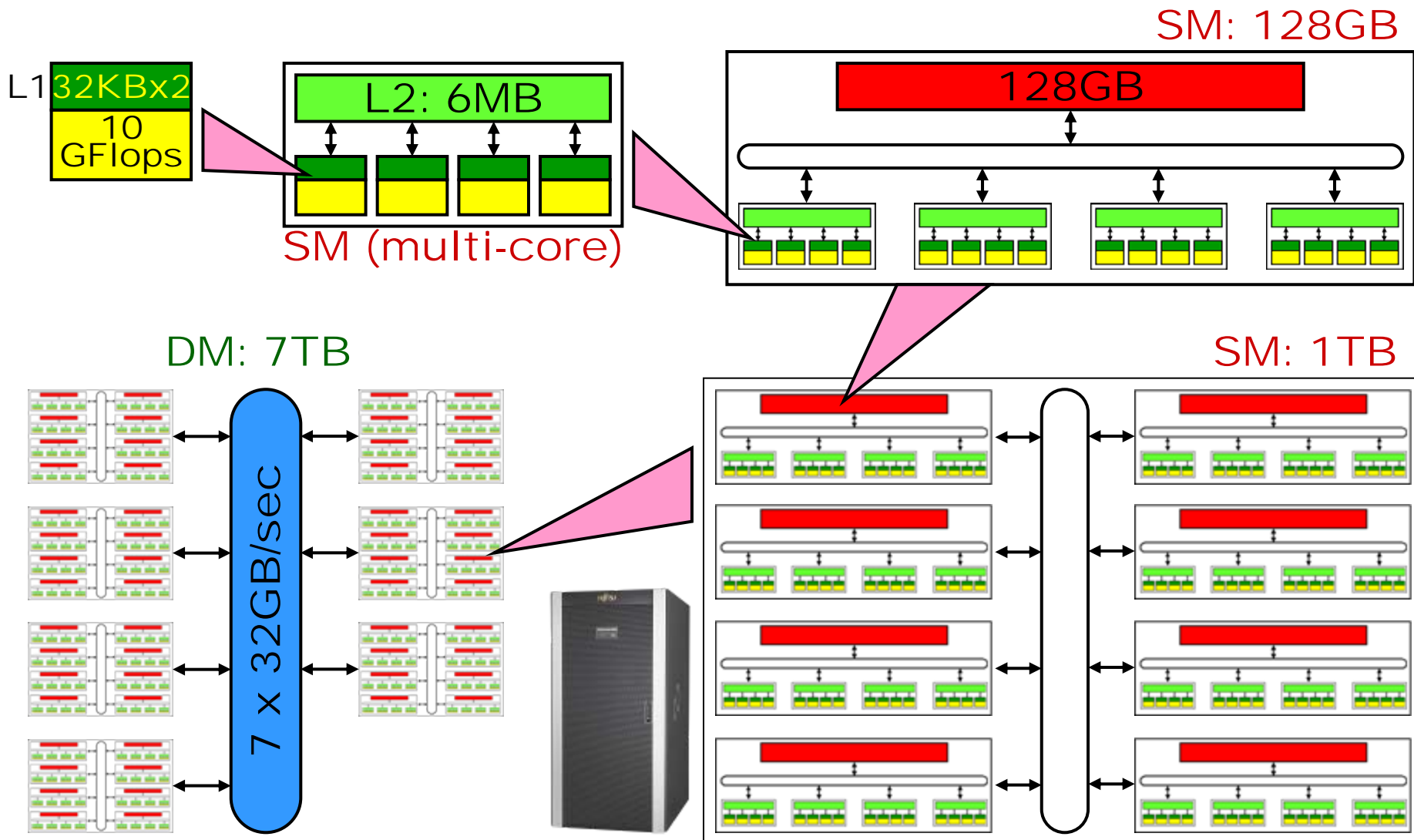
- 実例(1): 京大スパコン=富士通HX600

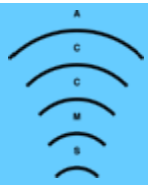




もう一つの方法: 並列マシン(3)

■ 実例(2): 京大スパコン=富士通SE M9000





もう一つの方法：並列マシン(4)

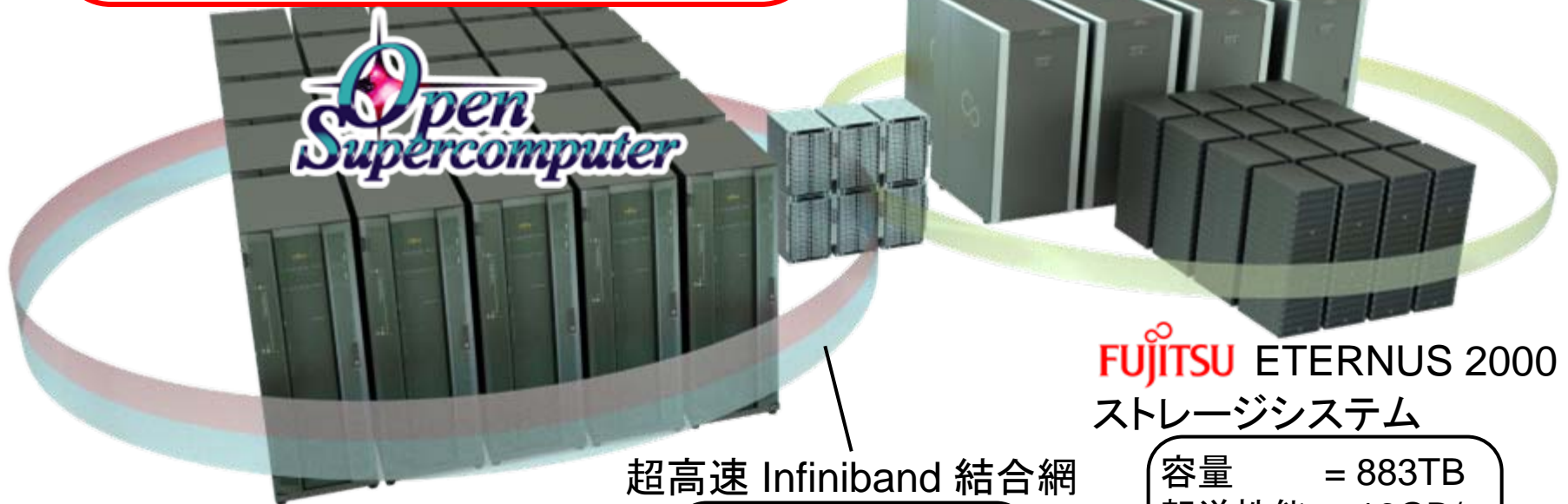
■ 京大スパコンの全体像

FUJITSU HX600 クラスタ

ノード数 = 416
 コア数 = $16 \times 416 = 6656$
 ピーク性能 = 61.2 TFlops
 Linpack 性能 = 50.5 TFlops(#308)
 メモリ容量 = 13 TB

FUJITSU SPARC Enterprise M9000 fat node サブシステム

ノード数 = 7
 コア数 = $128 \times 7 = 896$
 ピーク性能 = 8.96 TFlops
 メモリ容量 = $1\text{TB} \times 7 = 7\text{TB}$



超高速 Infiniband 結合網

通信性能 = 3.3TB/s

FUJITSU ETERNUS 2000 ストレージシステム

容量 = 883TB
 転送性能 = 16GB/s



■ 大量CPUによる同じ(ような)計算の分担方法

例: $z_i = x_i \times y_i$ ($i = 1, 2, \dots, n$)

- n 組のデータを p 個のCPUに均等に分割する
- それぞれのCPUが割当てられた計算をする
- おしまい

$$\begin{aligned}z_1 &= x_1 \times y_1 \\z_2 &= x_2 \times y_2 \\&\vdots \\z_{n/4} &= x_{n/4} \times y_{n/4}\end{aligned}$$

$$\begin{aligned}z_{n/4+1} &= x_{n/4+1} \times y_{n/4+1} \\z_{n/4+2} &= x_{n/4+2} \times y_{n/4+2} \\&\vdots \\z_{2n/4} &= x_{2n/4} \times y_{2n/4}\end{aligned}$$

$$\begin{aligned}z_{2n/4+1} &= x_{2n/4+1} \times y_{2n/4+1} \\z_{2n/4+2} &= x_{2n/4+2} \times y_{2n/4+2} \\&\vdots \\z_{3n/4} &= x_{3n/4} \times y_{3n/4}\end{aligned}$$

$$\begin{aligned}z_{3n/4+1} &= x_{3n/4+1} \times y_{3n/4+1} \\z_{3n/4+2} &= x_{3n/4+2} \times y_{3n/4+2} \\&\vdots \\z_n &= x_n \times y_n\end{aligned}$$



ベクトル VS 並列 (1)

■ 1990年代:ベクトル並列 VS スカラー並列

■ TOP-10 of @ 1993.6

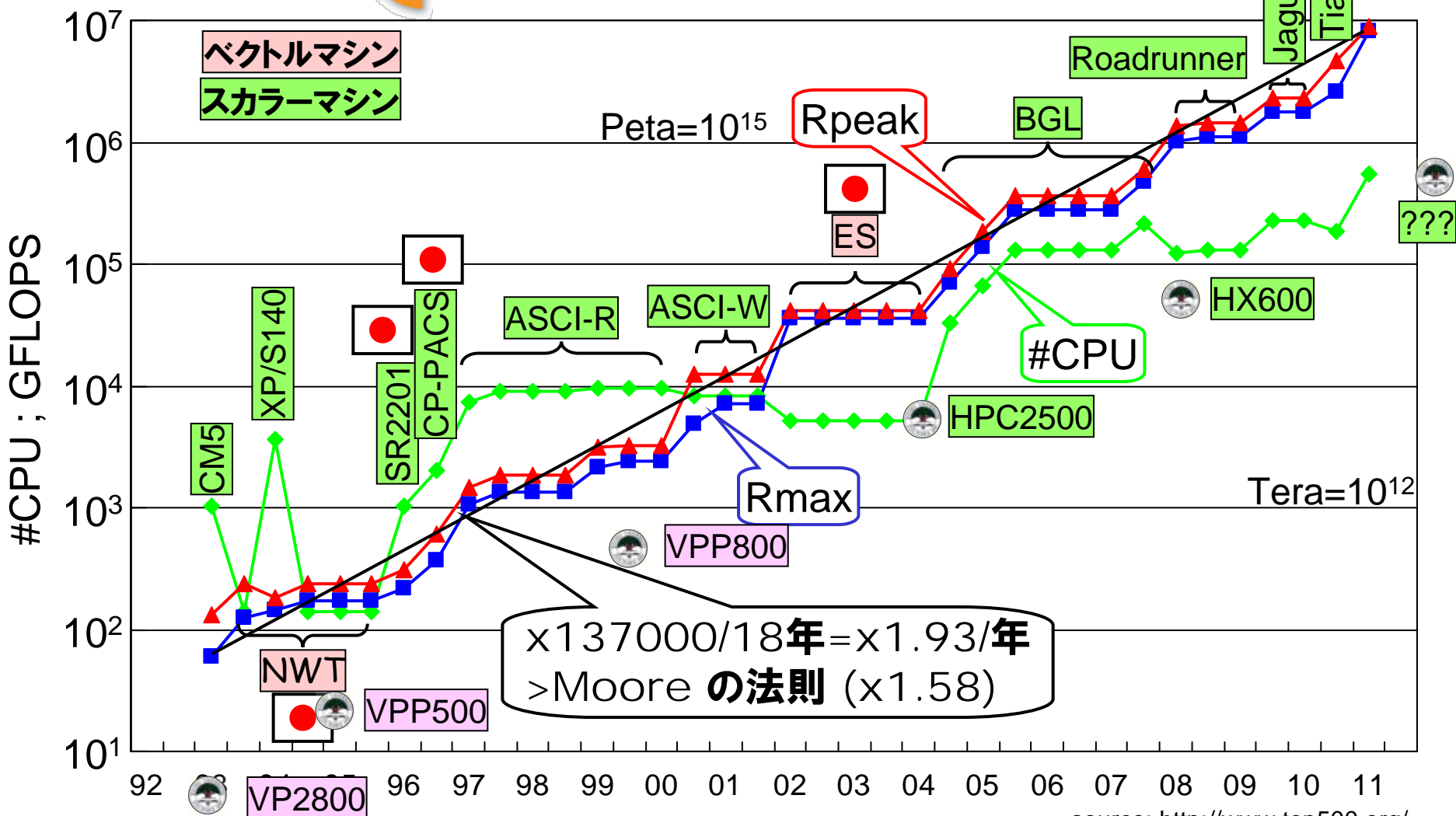
machine	#proc	Rmax	Rpeak
TMC CM-5	1024	59.7	131.0
TMC CM-5	544	30.4	69.6
TMC CM-5	512	30.4	65.5
TMC CM-5	512	30.4	65.5
NEC SX-3	4	23.2	25.6
NEC SX-3	4	20.0	22.0
TMC CM-5	256	15.1	32.8
Intel Delta	512	13.8	20.5
Cray Y-MP	16	13.7	15.2
Cray Y-MP	16	13.7	15.2

- 巨大で(>100万元)密な連立一次方程式の求解性能に基づく世界中のスパコン順位表
- 1993.6から毎年2回発表(6月&11月)
- Rmax: 求解性能
Rpeak: 理論最大性能
(単位GFlops:毎秒10億演算)



一般のスパコン: TOP500

■ #1 of **TOP500**





(いきなり&とりあえず)まとめ

■ ベクトルマシン

- 1つの演算を k 個の小さい操作に分割する
- 多数の同種演算を1小操作ずつずらして行う
 - k 倍の速度で計算できる(ように見える)
 - 大量 ($\gg k$) の同種演算が得意

■ 並列マシン

- 多数の同じ(ような)演算を p 個のCPUに分割
- それぞれのCPUが割当てられた計算をする
 - p 倍の速度で計算できる(ように見える)
 - 大量 ($\gg p$) の同じ(ような)演算が得意
- スパコンは大量の同じ(ような)演算(や処理)が得意



大量同種演算は何でも得意か？ (1)

- 超得意

$$z_i = x_i + y_i$$

- 普通に得意

$$z_i = (x_{i-1} + 2x_i + x_{i+1}) / 4$$

- 微妙に得意

$$z = x_1 + x_2 + \cdots + x_n$$

- 何とかなる

$$z_i = x_{f(i)} \text{ s.t. } z_1 \leq z_2 \leq \cdots \leq z_n$$

- 全然ダメ

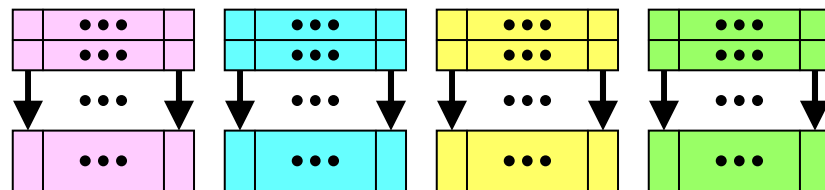
$$z_1 = f(x_1, 0), z_i = f(x_i, z_{i-1})$$



大量同種演算は何でも得意か？ (2)

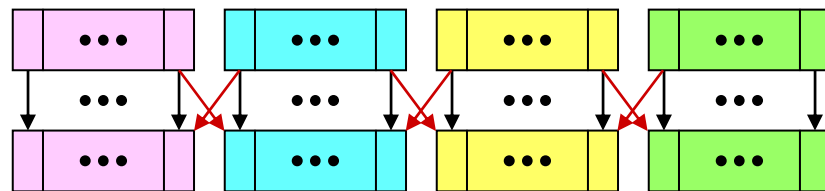
■ 超得意

$$z_i = x_i + y_i$$



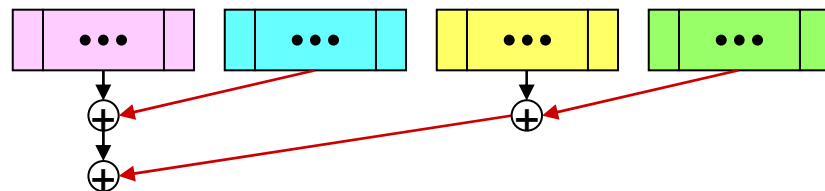
■ 普通に得意

$$z_i = (x_{i-1} + 2x_i + x_{i+1}) / 4$$



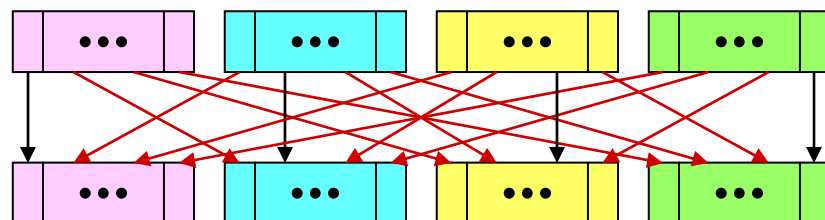
■ 微妙に得意

$$Z = x_1 + x_2 + \dots + x_n$$



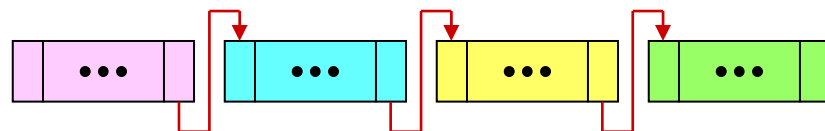
■ 何とかなる

$$z_i = x_{f(i)} \text{ s.t. } z_1 \leq z_2 \leq \dots \leq z_n$$



■ 全然ダメ

$$z_1 = f(x_1, 0), z_i = f(x_i, z_{i-1})$$



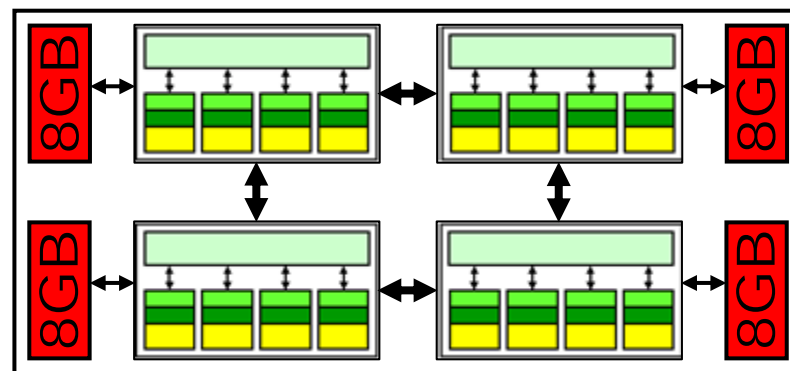
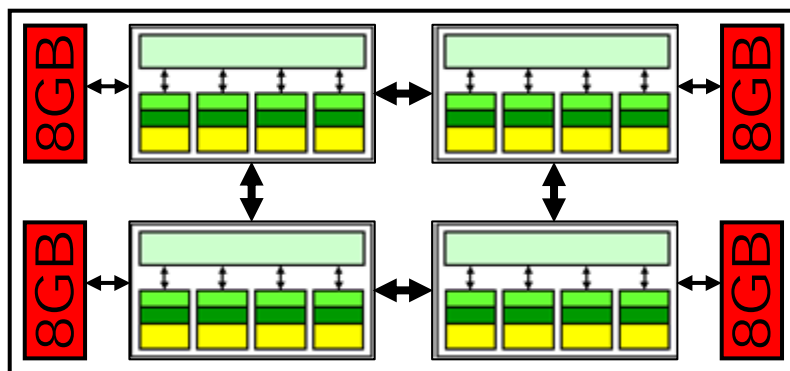


大量同種演算は何でも得意か？ (3)

■ 京大スパコンの通信速度

147GFlops

147GFlops



8GB/sec
=32Gbit/sec
=200Mbit/sec×160

3.3TB/sec=26.4Tbit/sec=200Mbit/sec×132000

- 1個の数値(8B)の通信時間 = 4.4 μ sec
= 40480回の演算時間
- 10億個の数値(8GB)の通信時間 = 1秒
= 1470億回の演算時間



まとめ & 課題

- **スーパーコンピュータは ...**
 - 大量の同じ(ような)演算(や処理)が得意
 - ただし演算どうしの**依存性が少ない**ことが必要
- ➔ **そんな都合のよい問題はあるのか？**

- **そこでレポート課題**
(できればスパコンに適する大規模な)並列計算により高い性能が期待できる**実際的な**問題を一つ挙げ、なぜその問題が並列計算に適するのかを説明せよ。