情報論理工学研究室

第1回：並列とは

1

2

3

4

5

6

7

8

9

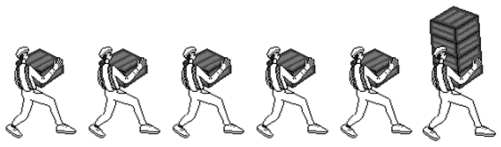
並列アルゴリズムとは

- 並列アルゴリズム(Parallel Algorithm)
  - 並列計算機で問題を解くためのアルゴリズム
    - 多数の計算機を使って高速に解く
    - 1台の計算機上で動作する通常のアルゴリズム (逐次アルゴリズム) とは異なる手法が必要

10

並列アルゴリズム(Parallel Algorithm)

1人だと時間が掛かる仕事がある
⇒10人いれば同じ仕事をもっと速くできる



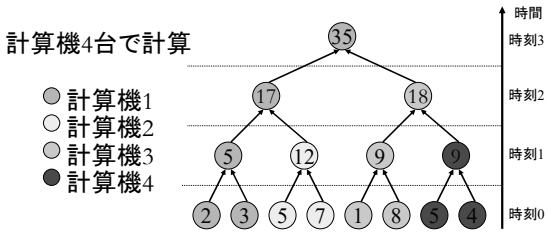
計算機1台だと時間が掛かる
⇒計算機10台でやればいい

11

並列アルゴリズムとは

- 例: 2+3+5+7+1+8+5+4

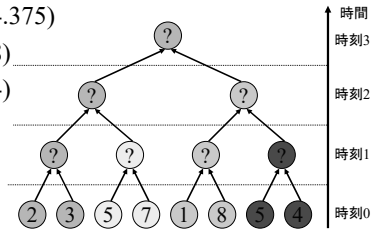
計算機4台で計算



12

並列計算の例

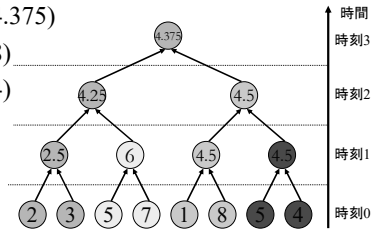
- 足し算と同様の処理が可能なのは？
  - 入力: 2, 3, 5, 7, 1, 8, 5, 4
- 平均値 (4.375)
- 最大値 (8)
- 中央値 (4)



13

並列計算の例

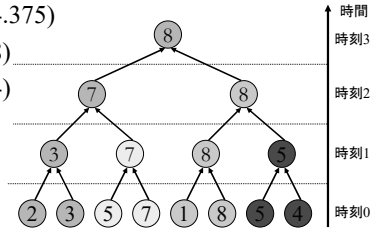
- 足し算と同様の処理が可能なのは？
  - 入力: 2, 3, 5, 7, 1, 8, 5, 4
- 平均値 (4.375)
- 最大値 (8)
- 中央値 (4)



14

並列計算の例

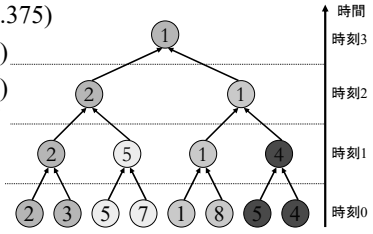
- 足し算と同様の処理が可能なのは？
  - 入力: 2, 3, 5, 7, 1, 8, 5, 4
- 平均値 (4.375)
- 最大値 (8)
- 中央値 (4)



15

並列計算の例

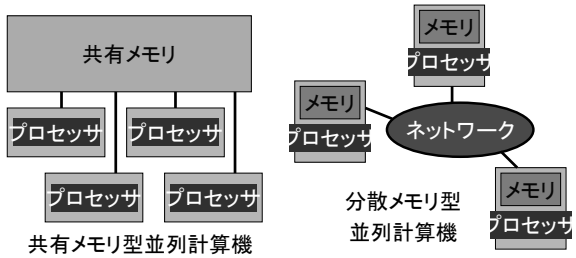
- 足し算と同様の処理が可能なのは？
  - 入力: 2, 3, 5, 7, 1, 8, 5, 4
- 平均値 (4.375)
- 最大値 (8)
- ~~中央値 (4)~~



16

並列計算機 (parallel computer)

- 複数のプロセッサを持ち高速計算が可能



共有メモリ型並列計算機

分散メモリ型
並列計算機

17

スーパーコンピュータ「京」

- 汎用京速計算機 (2012年7月完成)
  - 理化学研究所と富士通が共同開発
    - 2011年6月Top500 1位, CPU数: 68,544, 8.162 PFlops
    - 2011年11月Top5001位, CPU数: 88,128, 10.51 PFlops

スーパーコンピュータ「京」[1]

[1] スーパーコンピュータ「京」, 富士通,
http://www.fujitsu.com/jp/about/businesspolicy/tech/k/

18

スーパーコンピュータ「富岳」

- 汎用京速計算機 (2021年3月完成)
  - 「京」の後継機
    - 2020年6月Top500, HPCG, HPL-AI , Graph500 1位

スーパーコンピュータ「富岳」[2]

[2] スーパーコンピュータ「京」, 富士通, https://www.fujitsu.com/jp/about/businesspolicy/tech/fugaku/

19

京と富岳の世界順位

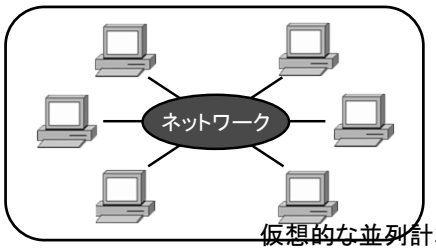
京	TOP500	HPCチャレンジクラス1			
		Global HPL	Global RA	STREAM	Global FFT
2011年6月	1位：8.162PFlop/s				
2011年11月	1位：10.510PFlop/s	1位	1位	1位	1位
2012年6月	2位：10.510PFlop/s				
2012年11月	3位：10.510PFlop/s	1位	2位	1位	1位
2013年6月	4位：10.510PFlop/s				
2013年11月	4位：10.510PFlop/s	1位	2位	1位	1位

富岳	TOP500	HPCG	HPL-AI
2020年6月	1位：415.43PFlop/s	1位：415.53PFlop/s	1位：415.53PFlop/s
2020年11月	1位：442.01PFlop/s	1位：442.01PFlop/s	1位：442.01PFlop/s
2021年6月	1位：442.01PFlop/s	1位：442.01PFlop/s	1位：442.01PFlop/s
2021年11月	1位：442.01PFlop/s	1位：442.01PFlop/s	1位：442.01PFlop/s
2022年6月	2位：442.01PFlop/s	1位：442.01PFlop/s	2位：442.01PFlop/s

20

クラスタコンピューティング

- Cluster computing
  - ネットワーク接続された計算機全体を仮想的な並列計算機とする



仮想的な並列計算機

21

クラスタコンピューティング

- クラスタコンピューティング
  - 長所
    - 安価な計算機を集めることで並列計算可能
  - 短所
    - 計算機間の通信が必要
      - 通信遅延が起きる
      - ネットワークに負荷がかかる

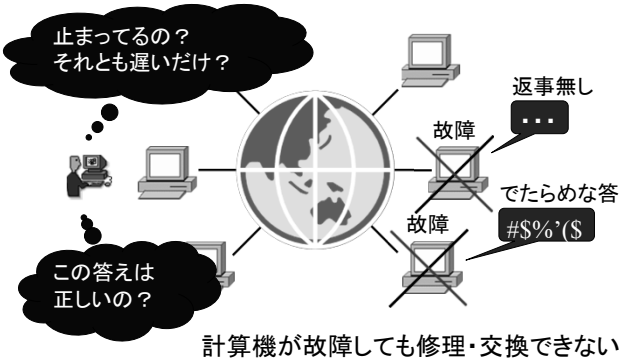
22

グリッドコンピューティング

- Grid computing
  - インターネット等を用いて広域的に計算機を連携し並列計算を行う

23

グリッドコンピューティング



計算機が故障しても修理・交換できない

24

グリッドコンピューティング

- グリッドコンピューティング
  - 長所
    - 非常に多くの計算機を利用可能
  - 短所
    - 通信遅延が非常に大きい
    - 全ての計算機を管理できない
      - 計算機が故障しても修理・交換できない

故障することを前提に
処理を行う必用がある

25

クラスタコンピューティングと グリッドコンピューティング

クラスタコンピューティングとグリッドコンピューティングの特徴

	クラスタコンピューティング	グリッドコンピューティング
規模	数十台～数百台	数千台～数万台
使用する計算機	組織内の計算機	世界中の計算機
計算機の管理	使用者が管理可能	使用者は管理不可能
通信遅延	比較的小さい	非常に大きい
計算機の種類	統一可能	統一不可能
故障への対応	使用者が対応可能	使用者は対応不可能

26

何故並列アルゴリズムが必要か？

- 並列化の利点
  - 計算時間の短縮
  - より複雑な問題が解ける
- 並列化の実現性
  - 複数の計算機が使用可能

27

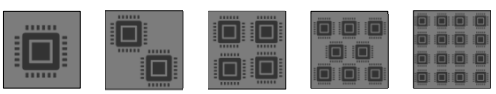
処理の高速化

- 様々な分野で複雑な問題を解く必要がある
  ⇒処理の高速化が必要
- 高速化のための手法
  1. 計算機高速化
  2. アルゴリズム改良
  3. 並列化

28

ムーアの法則

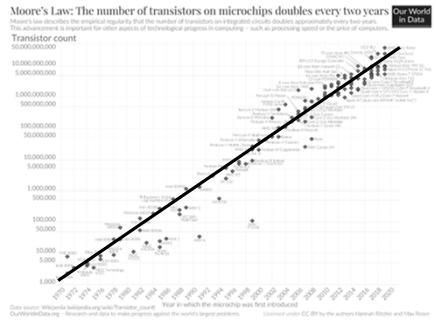
- Moore's law
  - ある大きさの計算機回路内の計算機素子の数は2年ごとに2倍になる



2年後　4年後　6年後　8年後

29

ムーアの法則



各年の単位面積辺りの計算機素子数 [1]

[1] https://upload.wikimedia.org/wikipedia/commons/0/00/Moore%27s_Law_Transistor_Count_1970-2020.png

30

ムーアの法則

ある大きさの計算機回路内の計算機素子の数は
2年ごとに2倍になる

⇒ 計算機の処理速度は2年ごとに2倍になる

2年後	4年後	6年後	8年後	10年後	12年後	14年後	16年後	18年後	20年後
2倍	4倍	8倍	16倍	32倍	64倍	128倍	256倍	512倍	1024倍

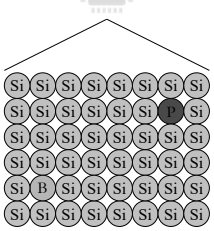
10年後	20年後	30年後	40年後	50年後	60年後	70年後
32倍	1024倍	32000倍	100万倍	3200万倍	10億倍	320億倍

今後もこのペースが続く？

31

計算機素子

計算機素子



原子(主に珪素)が
規則的に並んでいる

原子の大きさ
約1Å = 10^{-10}m = 0.0000001mm
珪素原子の大きさ
1.11Å

32

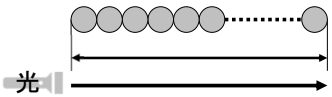
計算速度の限界

原子の大きさ:約1Å = 10^{-10}m

光 → 光速:30万km/s = 3.0×10^8 m/s

原子を通過するのにかかる時間:$\frac{10^{-10}}{3.0\times10^8} = 3.3 \times 10^{-19}$ s

原子100個で計算機素子が作れたと仮定



約100Å = 10^{-8}m

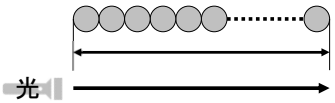
光

素子を通過するのにかかる時間:$\frac{10^{-8}}{3.0\times10^8} = 3.3 \times 10^{-17}$ s

33

計算速度の限界

原子100個で計算機素子が作れたと仮定



約100Å = 10^{-8}m

光 → 光速:3.0×10^8 m/s

素子を通過するのにかかる時間:$\frac{10^{-8}}{3.0\times10^8} = 3.3 \times 10^{-17}$ s

1秒間にできる演算:$\frac{3.0\times10^8}{10^{-8}} = 3.0 \times 10^{16}$個　30PHz

現在の計算の演算速度:10^9〜10^{10}　1GHz〜10GHz

あと100万倍〜1000万倍くらいしか速くならない

34

ムーアの法則

ある大きさの計算機回路内の計算機素子の数は
2年ごとに2倍になる

⇒ 計算機の処理速度は2年ごとに2倍になる

2年後	4年後	6年後	8年後	10年後	12年後	14年後	16年後	18年後	20年後
2倍	4倍	8倍	16倍	32倍	64倍	128倍	256倍	512倍	1024倍

10年後	20年後	30年後	40年後	50年後	60年後	70年後
32倍	1024倍	32000倍	100万倍	3200万倍	10億倍	320億倍

1000万倍
半世紀後には壁にぶつかる

35

アルゴリズム改良

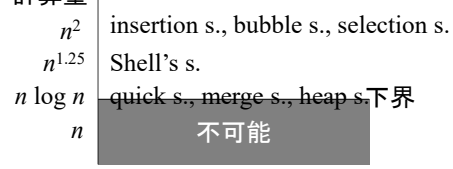
- 劇的な高速化が可能
  - 例:ソーティング
    - 挿入法: n^2
    - クィックソート: $n \log n$

	10	100	1000	1万	10万
$n \log n$	1秒	32秒	12分	3時間	3日
n^2	1秒	100秒	3時間	10日	3年
2^n	1秒	10^{19}年	10^{290}年		

36

アルゴリズム改良の壁

- 計算量には下界が存在
  - 例: ソーティングの下界 : $n \log n$
    - クィックソートの計算量: $n \log n$
    ⇒ソーティングはこれ以上の改良は不可能
  - 例: ナップサック問題の下界: おそらく 2^n (未解決)
- 並列アルゴリズムならばより低い計算量に



37

並列化の対象

- 何を並列化するか？

どんな処理でも速くできた方がいい

⇩

対象は何でもOK!

…とは言え費用対効果は考慮する必要あり

38

並列化の問題点

- 並列化の問題点
  - 並列計算機が必要
    - 並列計算機は高価
  - 並列アルゴリズムが必要
    - 並列性を考えてアルゴリズムデザインする必要あり
  - 並列化できるとは限らない
    - 並列化しにくい問題もある
  - 高速化できるとは限らない
    - 並列化しても速度の上限はある

39

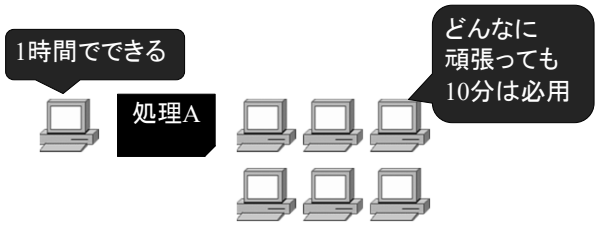
並列化の対象

- 並列化するべき対象
  - 処理速度が必要
    - リアルタイム処理が必要な場合
    - 期限が設けられている場合
  - 複雑な計算
    - 膨大なデータに対する正確な計算が必要な場合
  - 費用をかけてもする意義のある処理
    - 科学的意義や技術的意義のある重要な処理

40

並列化によるスピードの上限

- 並列化によるスピードの上限
  - どんなに頑張っても計算機台数倍まで
    - 計算機10台なら速さ10倍が上限



41

並列化の2つの目標

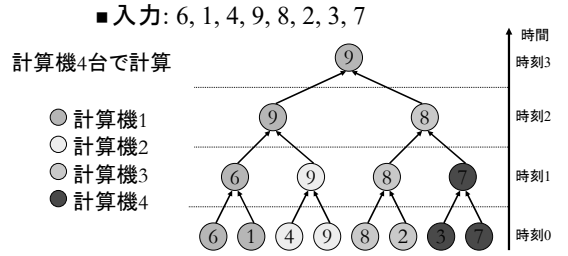
- 最速化：とにかく速くする
  - 計算機を何台使ってもいいので計算時間を減らす
    - 計算機を100台使って20倍速く
- 最適化：効率良く速くする
  - 計算機の台数分速くする
    - 計算機を10台使って10倍速く

42

計算時間の例

■ 最大値計算

■ 入力: 6, 1, 4, 9, 8, 2, 3, 7



計算機を何台使ってもいいのでもっと早くするには？

43

最速最大値アルゴリズム

■ 最速最大値アルゴリズム

☙データ数の2乗台の計算機を用いる

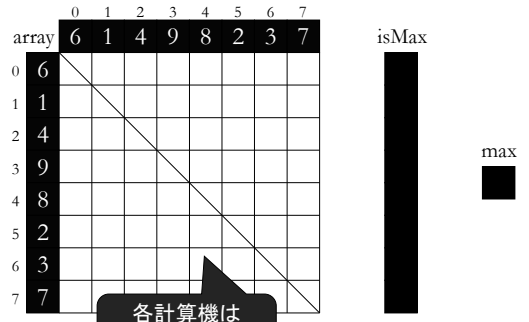
例: 6, 1, 4, 9, 8, 2, 3, 7 の最大値

データ数が8個なので

$8^2 = 64$ 台の計算機を用いる

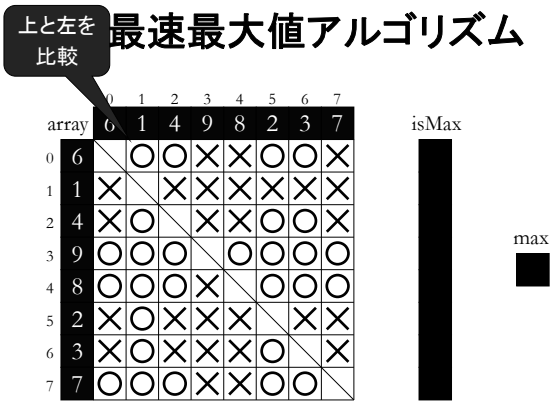
44

最速最大値アルゴリズム



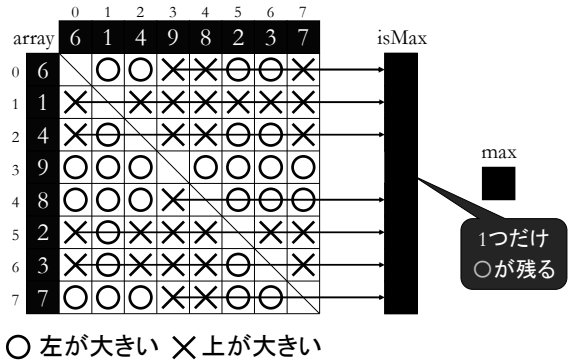
45

最速最大値アルゴリズム



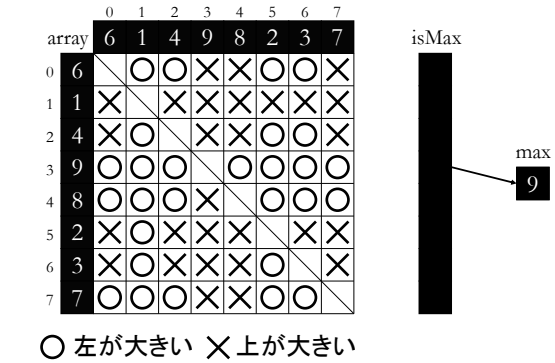
46

最速最大値アルゴリズム



47

最速最大値アルゴリズム



48

最大値計算の時間

1つのデータを見るのに1秒かかる場合

	データ数	10	100	1000	1万	10万	100万	n
1台	時間	10秒	1分40秒	20分	3時間	1日	10日	n
最速化	時間	2秒	2秒	2秒	2秒	2秒	2秒	1
	計算機	100台	1万台	100万台	1億台	100億台	1兆台	n^2
最適化	時間	4秒	7秒	10秒	14秒	17秒	20秒	$\log n$
	計算機	4台	16台	128台	1024台	8000台	6万台	$\frac{n}{\log n}$

最速化と最適化，どちらを目指す？

49

宿題:「京」と「富岳」の調査

- スーパーコンピュータ「京」および
  その後継機「富岳」について調査
  - いつ作られた？
  - 計算速度は？
  - プロセッサは何台？
  - いつ世界一になった？
  - 現在世界何位？
  - どこが開発した？
  - どこにある？
  - etc.

50