Parallel and Distributed Computing in Statistics

Diao Tianbo

School of Mathematics and Statistics
Central China Normal University

December 20, 2022

Table of contents

Table of contents

What is Parallel and Distributed Computing

Wikipedia

- Parallel computing[1]: Parallel computing is a type of computation where many calculations or the execution of processes are carried out simultaneously.
- Distributed computing[2]: A distributed system is a system whose components are located on different networked computers, which communicate and coordinate their actions by passing messages to one another from any system.

[1] Parallel computing:https://en.wikipedia.iwiki.eu.org/wiki/Parallel_computing

[2] Distributed computing:https://en.wikipedia.iwiki.eu.org/wiki/Distributed_computing

What is parallel and distributed computing

Common Explanation

- Parallel computing: Multiple processors performs multiple tasks assigned to them simultaneously. Memory in parallel systems can either be shared or distributed.
- Distributed computing: Each processor has its own private memory (distributed memory). Information is exchanged by passing messages between the processors.

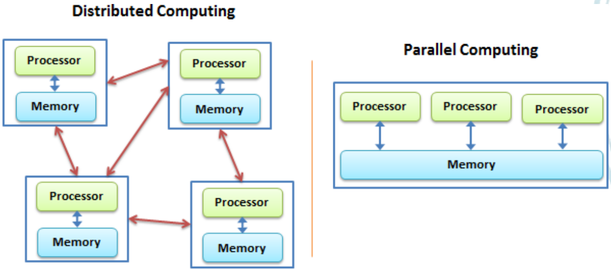


Figure 1: The difference between parallel and distributed computing

The difference

Difference between Parallel Computing and Distributed Computing:

Table 1: Comparison

Parallel Computing	Distributed Computing
1.Many operations are performed simultaneously	System components are located at different locations
2.Single computer is required	Uses multiple computers
3.Multiple processors perform multiple operations	Multiple computers perform multiple operations
4.It may have shared or distributed memory	It have only distributed memory
5.Processors communicate with each other through bus	Computer communicate with each other through message passing
6.Improves the system performance	Improves system scalability, fault tolerance and resource sharing capabilities

Why we need parallel and distributed computing

- Data World: Big data is ubiquitous today. In the era of "big data," in many applications, it is impossible to store data in a single device or central location.
- New Challenge: The big data challenges current numeric statistical and machine learning methods, visualization methods, computational methods and computational environments.

Figure 2: Are we drowning in a sea of Big Data

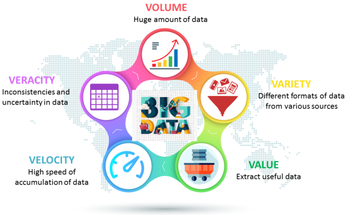

VOLUME
Huge amount of data
VERACITY
Inconsistencies and uncertainty in data
BIG DATA
VARIETY
Different formats of data from various sources
VELOCITY
High speed of accumulation of data
VALUE
Extract useful data

Figure 3: Features of Big Data

Table of contents

History and its development

- Flynn (1966; 1972) created one of the earliest classification systems for parallel (and sequential) computers and programs, known as **Flynn's taxonomy**.
- single-instruction-single-data(**SISD**)
- single-instruction-multiple-data(**SIMD**)
- Multiple-instruction-single-data(**MISD**)
- Multiple-instruction-multiple-data(**MIMD**)

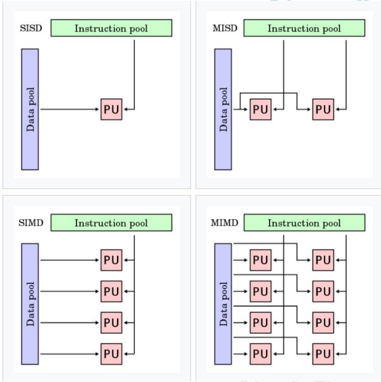


Figure 4: Flynn's taxonomy

History and its development

Amdahl's law: In parallel statistical computing, the speed-up for r processors is defined as $S_r = \frac{T_1}{T_r}$, where T_k is the running time with k processors. According to Amdahl's law, the theoretical limit of speed-up is

$$S_r \leq \frac{1}{(1-\alpha) + \frac{\alpha}{r}} \leq r$$

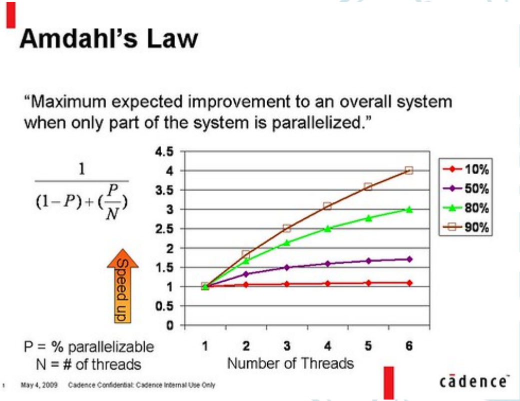
where α is the proportion of the computation that can be run in parallel, and $1 - \alpha$ is the proportion that remains serial.

History and its development

- **Amdahl's law:**

$$S_r \leq \frac{1}{(1-\alpha)+\frac{\alpha}{r}} \leq r$$

- S_r is the speed-up for r processors
- where α is the proportion of the computation that can be run in parallel



Amdahl's Law

"Maximum expected improvement to an overall system when only part of the system is parallelized."

$$\frac{1}{(1-P)+(\frac{P}{N})}$$

P = % parallelizable
N = # of threads

May 4, 2009 Cadence Confidential: Cadence Internal Use Only

cādence

Figure 5: Amdahl's law

History and its development

Parallel processing for statistics: a review

- **Exploiting parallelism for statistics is not a new idea, nor is it entirely ignored in major statistical texts.**
- Chambers (1977) mentioned the use of parallel computing in data analysis situations where the magnitude of computation makes interactive analysis impossible.
- Thisted (1988) briefly mentioned parallel computers as being an ideal method of implementing Jacobi methods for extracting eigenvalues.
- Schervish (1988) considered a variety of parallel applications-including discrete-finite inference, a computerintensive approach for the analysis of discrete data-where the dominant aspect of the computation is simple summation of large sets of data.
  (to be continued)

History and its development

Parallel processing for statistics: a review

- Skvoretz et al. (1992) employed an NCUBE/10 hypercube multiprocessor (MIMD) to assess the application of parallel processing to typical large-scale social science research.
- Schmidberger (2009) gave a classification of parallel statistical computers: multicore system,multiprocessor system,multicomputer with computing cluster(distributed computer), and multicomputer with grid computing (grid computers).
- The classification is also very popular in parallel computers. The tutorial of Creel and Goffe (2008) urged further use of these techniques by economists. Creel (2005) identified a steep learning curve and expensive hardware as the main barriers to adoption.

Table of contents

Introduction

Parallel Computing

Distributed Computing

- A Overview of Distributed Computing
- Example for Distributed Computing

Applications

A Overview of Distributed Computing

- The study of distributed computing became its own branch of computer science in the late 1970s and early 1980s.
- The first conference in the field, Symposium on Principles of Distributed Computing (PODC), dates back to 1982, and its counterpart International Symposium on Distributed Computing (DISC) was first held in Ottawa in 1985 as the International Workshop on Distributed Algorithms on Graphs.

A Overview of Distributed Computing

Distributed Architectures:

- Various hardware and software architectures are used for distributed computing.
- At a lower level, it is necessary to interconnect multiple CPUs with some sort of network, regardless of whether that network is printed onto a circuit board or made up of loosely coupled devices and cables.
- At a higher level, it is necessary to interconnect processes running on those CPUs with some sort of communication system.

A Overview of Distributed Computing

Distributed Computing in Statistics:

- With the advent of the Big Data era, distributed computing is ushering in its own spring of development in statistics.
  In recent years, more and more statistical papers on the processing of data are distributed.
- **Distributed computing is already one of the mainstream of statistics**

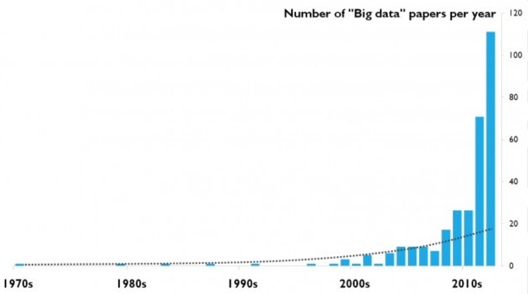


Figure 6: Fast Growing Distributed Statistics Essays

Table of contents

Example for Distributed Computing

We give an algorithm as follows:

Algorithm 1 : Distributed Communication algorithm

```
**Input**. Initial value $\theta_0$, $\varphi$, number of iterations $T$.
**For** $t = 0, 1, 2, \ldots, T-1$ :
    • Each machine evaluates $\nabla\mathcal{L}_N(\theta_n)$ and sends to the $1^{st}$ machine;
    • The $1^{st}$ machine computes $\theta_{n+1} = \theta_n - \frac{1}{\varphi}\nabla\mathcal{L}_N(\theta_n)$;
**Output**. $\theta_T$
```

Table of contents

Applications

R software runs a program in parallel

```
library(doParallel)
library(glmnet)
number_cores <- detectCores() - 1
print(number_cores)
# Initiate cluster
cl <- makeCluster(number_cores)
registerDoParallel(cl)
x <- matrix(rnorm(1e5 *100), 1e5, 100)
y <- rnorm(1e5)
system.time(cv.glmnet(x,y)) # not parallel
system.time(cv.glmnet(x,y,parallel=TRUE)) # this is parallel
stopCluster(cl)
```

Applications

R software runs a program in parallel

```
> print(number_cores)
relax [1] 7
> system.time(cv.glmnet(x,y)) # not parallel
user system elapsed
8.01 0.53 8.54
> system.time(cv.glmnet(x,y,parallel=TRUE)) # this is parallel
user system elapsed
2.07 0.74 8.76
```

Reference

1. MJ Schervish. Applications of parallel computation to statistical inference. Journal of the American Statistical Association. 1988
2. NM Adams, SPJ Kirby, P Harris, DB Clegg. A review of parallel processing for statistical computation.Statistics and Computing. 1996

Thank you!