Computer Architecture and Algorithms for High Performance Computing Through Parallel and Distributed Processing

Venkata Subba Reddy Poli
Computer Architecture and Algorithms for High Performance Computing through Parallel and Distributed Processing

P. Venkata Subba Reddy

Abstract—There is a very high need of High Performance Computing (HPC) in many applications like space science to Artificial Intelligence. HPC shall be achieved through Parallel and Distributed Computing. In this paper, Parallel and Distributed algorithms are discussed based on Parallel and Distributed Processors to achieve HPC. The Programming concepts like threads, fork, sockets and par do are discussed with some simple examples for HPC.

Index Terms—High Performance Computing, Parallel and Distributed Algorithms, Computer Architecture, Computer Programming

1 INTRODUCTION

Computer Architecture and Programming play an important role for High Performance computing (HPC) in large applications like space science to Artificial Intelligence[1]. The Algorithms are problem solving procedures and later these algorithms transform into particular Programming language for HPC. There is need to study algorithms for High Performance Computing. These Algorithms are to be designed to computer in reasonable time to solve large problems like weather forecasting, Tsunami, Remote Sensing, National calamities, Defense, Mineral exploration, Finite-element, Cloud Computing, and Expert Systems etc. The Algorithms are Non-Recursive Algorithms, Recursive Algorithms, Parallel Algorithms and Distributed Algorithms.

The Algorithms must be supported the Computer Architecture. The Computer Architecture is characterized with Flynn’s[2] Classification SISD, SIMD, MIMD, and MISD. Most of the Computer Architectures are supported with SIMD (Single Instruction Multiple Data Streams). The class of Computer Architecture is VLSI Processor, Multiprocessor, Vector Processor and Multiple Processors[1,3].

2 ALGORITHMS

There are Non-Recursive Algorithms, Recursive Algorithms, Parallel Algorithms and Distributed Algorithms[5,10]. In the following Algorithms, Computer Programming and Architectures are discussed

2.1 Non-Recursive Algorithms

Non-Recursive Algorithms are systematically applied to the problems by analyzing the efficiency. Consider the algorithm of finding maximum element in the array A (0 .. n-1)

\[
\text{maxval } \leftarrow A[0]
\]

for i\(\leftarrow 0\) to n-1 do

if A[i] > maxval

axval \(\leftarrow A[i]\)

return maxval

The problem will be analyzed as

C(n) is number of computations. Where n is input size. The number of times of operations in execution

\[
C(n) = \sum_{i=1}^{n-1} \Theta(n) = n-1
\]

2.2 Recursive Algorithms

The recursive algorithm is binary expansion whose number of executions is multiplication.

For instance n! = n*n-1*n-2*…*1

Consider the algorithm of finding n!

If n=0 return 1

Else return F(n-1)*n

The number of multiplications M(n) needs to compute

M(n)=M(n-1) +1 for n>0

M(0)=0

2.3 Parallel Algorithms

Parallel Algorithms are designed to apply on problem to compute parallel whose number of executions is independent.

For instance parallel sum of odd and even numbers up to A[n].

oddsun \(\leftarrow 0\), evensun \(\leftarrow 0\)

for i\(\leftarrow 0\) to n-1 stem 2 do

oddsun \(\leftarrow \text{oddsun } + A[i]\)

return oddsun, evensun
return oddsum
for i=1 to n-1 stem 2 do
    evensum ← evensum + A[i]
return evensum

Consider the algorithm to find parallel matrix multiplication of A[n, n] and B[n,n].

for i=0 to n-1 do in parallel
    for j=0 to n-1 do in parallel
        C[i,j] ← 0
    for k=0 to n-1 do in parallel
return C

2.4 Distributed Algorithms
Distributed Algorithms are designed to apply on problems to compute on Distributed Systems whose executions are independent and distributed.
Consider the algorithm to compute two transactions
The algorithm of the problems is designed to execute on Distributed Systems.

if (fork())

    #Transaction1;

else
    #Transaction2

3 Parallel and distributed Architecture and Programming
Parallel and distributed Computer Architecture of processors is defined through Flynn’s classification (SISD, SIMD, MIMD, MISD)

<table>
<thead>
<tr>
<th>Tightly</th>
<th>Parallel and Distributed</th>
<th>Loosely</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multicpu</td>
<td>Bus</td>
<td>Switched</td>
</tr>
</tbody>
</table>

Parallel and Distributed Computer Systems

There are four minimum number of Architecture are VLSI Processor, Multiple Processor(Multi Computer) and Vector Processors. These Architectures are discussed in the following

3.1 VLSI Processor
VLSI Chip has Computer components such as Processor arrays (Processing Elements), Memory array and large scale switching networks. Communicate the PEs for implementing Parallel Algorithms with VLSI Chip.

use threads;
$thr1= threads->new("ascending");
$thr2= threads->new("decending");
sub ascending {
    my $num=0;
do { $num=$num+1;
    print "$num
";
} while ( $num<10)
}
sub decending {
    my $num=10;
do { print "$num
";
    $num=$num-1;
} while ( $num>0)
}$thr1-> join;
$thr2->join;

3.2 Multi Processor
Multiprocessor Computer Have been modeled as n Processor and Parallel Random Access Machine (PRAM) with shared memory. The Parallel Algorithms will be implemented with PRAM.

Consider the Parallel Algorithm for computation for Multiprocessing System
oddum ← 0
for $i \leftarrow 0$ to $n-1$ stem 2 do
  oddsum $\leftarrow$ oddsum + $A[i]$
return oddsum

one of the PEs set to evensum:
evensum $\leftarrow$ 0
for $i \leftarrow 1$ to $n-1$ stem 2 do
  evensum $\leftarrow$ evensum + $A[i]$
return oddsum

The Perl Program for above problem using fork for Parallel Processing

3.3 Multi Computer System

Multi Computer involves sequence of routers and channels for as number of Computer Systems with Message Passing Interconnection Network. The Parallel Algorithms will be implemented with this Network.

Multi Computer System

Multi computer System/ Distributed system

Consider the Parallel/Distributed algorithms in Multi Computer/distributed System. These algorithms will be computed in two ways Client/Server and Remote Procedural Calls

Remote Procedural Computation

The Client request the Data from the Server and the Server sends the Data from the Server buffer. The Perl Program gets Time from Server.

Remote Procedural Computation

3.4 Vector Processors

Super Computers are model with Vector Processor. Super Computers are specified by 5-tuples

$M = \langle N, C, I, M, R \rangle$

Where

$N$=number of processors
$C$= Set of instructions
$I$ is set of instructions for parallel execution
$M$= Set of Processors
$R$= Set of routing functions
The Architecture of Vector Supercomputer

Parallel Algorithms are designed to compute big problems like Weather forecasting, Remote sensing, Mineral exploration, Oceanography etc in parallel using Super Computers.

Consider the algorithm to find Parallel Matrix Multiplication $A[nxn]$ and $B[nxn]$, where $n$ is very large.

The Perl program for above parallel algorithm for matrix multiplication is given by

```
for i=0 to n-1 do in parallel
for j=0 to n-1 do in parallel
    P[i,j] set to C[i,j]=0
for k=0 to n-1 do in parallel
    A[I,k] and B[k,j] send to P[i,j]
return C
```

The Perl program for above parallel programming for matrix multiplication using `par do` in FORTRAN is given by

```
par do 300 i = 1, n
    par do 200 j = 1, n
        par do 100 k = 1, n
            a(i,k) = a(i,k) + b(i,j) * c(j,k)
        100 continue
    200 continue
300 continue
```

4 CONCLUSION

High Performance Computing is required when large computations of the problems. HPC shall be performed through the Parallel and distributed Algorithms. The Parallel and Distributed are discussed based on Computer Architecture. The Class of Algorithms and Class of Computer Architecture are discussed. The Programming concepts like threads, fork, sockets and Par Do are discussed for HPC. Some simple examples are discussed for HPC. The examples shall be extending to large problems like Grid Computing and Cloud Computing. Usually Fortran is used for HPC. The Perl and Java Programming are also useful for HPC[11].

ACKNOWLEDGMENT

The author wishes to thank B.Tech., M.Tech., and Ph.D Students for helping me.

REFERENCES


Dr. P. Venkata Subba Reddy was Professor and Head, Department of Computer Science and Engineering, MCRIT, Udayagiri, India during 2006-07. He is currently Associate Professor in Department of Computer science and Engineering, College of Engineering, Sri Venkateswara University, Tirpathi, India working since 1992. He did Ph.D in Artificial Intelligence, 1992. Sri Venkateswara University, Tirpathi, India. He did Post Doctoral/Visiting fellowship in Fuzzy Algorithms under Prof. V. Rajaraman, SERC, IISC/JNCASR, Bangalore, India. He is actively engaged in Teaching and Research work to B.Tech., M.Tech., and Ph.D students. He is actively in doing research in the areas of fuzzy systems, database systems, Software Engineering, Expert Systems and Natural language processing. He published papers in reputed National and International journals. He is an Editor for JCSE.