Parallelizing Stock Buy and Sell

CSE 708 Parallel Algorithms Seminar
- Dheeraj Sai Gogineni
  50463786
Contents

- Problem Statement
- Practical Application
- Sequential - 1d DP Approach
- Parallel Approach
- Bash Script Explanation
- Results
- References
Problem Statement Overview

Buy low, sell high – the fundamental principle of stock trading.

Find the maximum profit by determining the best buying and selling points for a given stock over time.
# Real time Application

<table>
<thead>
<tr>
<th>Day Trading</th>
<th>Investment Research</th>
<th>Financial Planning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day traders buy and sell stocks within a single trading day to take advantage of intraday price fluctuations. Solving the stock buy and sell problem helps them make profitable trading decisions.</td>
<td>Investment analysts and researchers use similar concepts to analyze historical stock data and make predictions about future stock price movements.</td>
<td>Financial planners and advisors can use these algorithms to optimize their clients' investment portfolios, helping them achieve their financial goals.</td>
</tr>
</tbody>
</table>
Sequential 1d DP Approach

Time Complexity : $O(n)$

```java
int maxProfit = 0;
int mini = Arr[0];

for(int i=1;i<Arr.size();i++) {
    int curProfit = Arr[i] - mini;
    maxProfit = max(maxProfit, curProfit);
    mini = min(mini, Arr[i]);
}
print(maxProfit);
```

Space Complexity : $O(1)$
Sequential 1d DP Approach

Sell on day 1 $3
Sell on day 2 $2
Sell on day 3 $90

Sell on day 4 loss $2
Sell on day 5 $6
Parallel Approach Architecture

best price = max(price[i] - price[j])
where \( i > j \)

min val = minimum value of the array
Bash Script Used

#!/bin/bash
# Indicates this is a bash Script
#SBATCH --nodes=128
# Total number of nodes used
#SBATCH --ncores-per-node=1
# Number of cores used per Node
#SBATCH --constraint=IB|OPA
# Specifies the communication network
#SBATCH --time=00:10:00
# Specifies the time limit

# These lines specify the partition and quality of service (QoS) for the job
#SBATCH --partition=general-compute
#SBATCH --qos=general-compute

#SBATCH --job-name="input1000-128node-1core"
# This line sets a name for the job
#SBATCH --output=input1000-1node-1core-pl128.out
# Standed Output File Name

#SBATCH --exclusive
# This line requests exclusive node allocation, meaning that no other jobs will share the allocated nodes.

# This line loads the Intel software module, which is often used to set up the development environment with Intel compilers and libraries.
module load intel

# This line sets an environment variable related to the Intel MPI library
export I_MPI_PINNING=/opt/software/slurm/lib64/libpmi.so

# This line specifies the program file
mpicc -o compiled_file stock_buy_sell.c

# This line uses srun to run the compiled executable
srun -n 128 compiled_file
Results for Large Inputs
$1 \times 10^6, 5 \times 10^6, 9 \times 10^6$

Amdahla Law
Results for scaled constant data per node 1e5

Gustafson’s Law
Results for Sequential Approach
Comparasion of Sequential vs parallel vs scaled constant
References

• Dr. Jones Lectures on MPI

• https://carleton.ca/rcs/rcdc/introduction-to-mpi/

• https://www.geeksforgeeks.org/stock-buy-sell/
Thank You