Name Tsion Meride
Problem 1
A,

After executing the program, i have analyze the results based on your question:

1. speedups and efficiencies as p is increased with n fixed

- **Speedup**: Generally increases with more processors because the parallel runtime decreases. However, the increase may diminish due to the logarithmic overhead. At some point, additional processors provide less benefit.
- **Efficiency**: Tends to decrease as p increases. This is because the overhead from log2(p)\log_2(p)log2(p) becomes significant relative to the gains in speedup, leading to less efficient use of resources.

2. when p is fixed and n is increased

- **Speedup**: Increases as n increases because the serial execution time grows quadratically (i.e., n2n^2n2), which can lead to higher speedups in parallel execution. However, the parallel time also increases, and the overall speedup may not grow as quickly as n.
- **Efficiency**: May remain stable initially but can decrease at larger values of n. This is due to the increasing impact of the parallel runtime's logarithmic term, especially if the workload becomes too large relative to the number of processors.

The results of this analysis show how speedup and efficiency are influenced by the number of processors and the size of the problem. Understanding these dynamics is crucial for optimizing parallel applications.

B,

Case 1: Toverhead grows more slowly than Tserial

Assume:

Toverhead = O(Tserial)

This means that as n increases, Toverhead grows slower than Tserial.

1. Expression for Efficiency:

E = n^2 / (n^2 + p. Toverhead)

2. Behavior as n Increases:

Since Toverhead grows slower than n^2 , we have:

$$\lim_{n \to \infty} \text{Toverhead} / n^2 = 0$$

Therefore, as n increases, p . Toverhead becomes negligible compared to n^2 .

3. Limit of Efficiency:

Thus:

$$E \to n^2 / n^2 + 0 = 1$$

This shows that the efficiency increases towards 1 as n increases.

If Toverhead grows more slowly than Tserial, the parallel efficiency E will increase as the problem size n increases.

Case 2: Toverhead grows faster than Tserial

Assume

Toverhead = Omega Tserial

This means that as n increases, Toverhead grows faster than Tserial

1. Expression for Efficiency:

$$E = n^2 / n^2 + p . \text{Toverhead}$$

2. Behavior as n Increases:

Since Toverhead grows faster than n^2 , we have:

$$, \quad \lim_{N \to \infty} n^2 / \text{Toverhead} = 0$$

Therefore, as n increases, p .Toverhead dominates n^2 .

3. Limit of Efficiency:

Thus:

$$E \to n^2 / 0 + p.\text{Toverhead} = 0$$

This shows that the efficiency decreases towards 0 as n increases.

If Toverhead grows faster than Tserial, the parallel efficiency E will decrease as the problem size n increases.
- If Toverhead grows slower than Tserial, parallel efficiency increases with problem size.
- If Toverhead grows faster than Tserial, parallel efficiency decreases with problem size.

Problem 2

- S: Sequential execution time
- T: Number of threads
- O: Parallelization overhead (fixed for all versions)
- B: Cost for the barrier
- M: Cost for each mutex operation
- N: Number of elements to sum

(a) Execution Time of Parallel Versions

Version 2

In Version 2, the elements are divided among T threads, and each thread sums its portion. The execution time can be modeled as:

$$T2 = \frac{S}{T} + O + B$$

Here, $\frac{S}{T}$ is the time taken by each thread to sum its portion, O is the overhead, and B is the time spent on synchronization (barrier).

Version 3

In Version 3, similar to Version 2, threads sum their portions, but this version also includes additional overhead for synchronization. The execution time can be modeled as:

$$T3 = \frac{S}{T} + O + (T-1) . B$$

Here, (T - 1) . B accounts for the barriers that each thread (except the last one) may encounter during synchronization.

Version 5

In Version 5, there is additional work for thread 0. It sums its portion and then also processes the results of the other threads. The execution time can be modeled as:

$$T5 = \frac{S}{T} + O + B + \frac{S}{T} + M$$

Here, the first $\frac{S}{T}$ is the time for thread 0 to sum its portion, O is the overhead,B is the synchronization cost, and the second $\frac{S}{T}$ represents the time taken for thread 0 to process the results from the other threads. The M accounts for the mutex operations that may be required during this additional work.

(b) Model for When Parallelization is Profitable for Version 3

To determine when parallelization is profitable for Version 3, we need to compare the execution time of the parallel version to that of the sequential version.
Profitability Condition
Parallelization is considered profitable when the execution time of the parallel version is less than that of the sequential version:

$$T3 < S$$

Substituting the expression for T3:

$$\frac{S}{T} + O + (T - 1).B < S$$

Rearranging the inequality, we find:

$$\frac{S}{T} + O + (T - 1).B < S$$

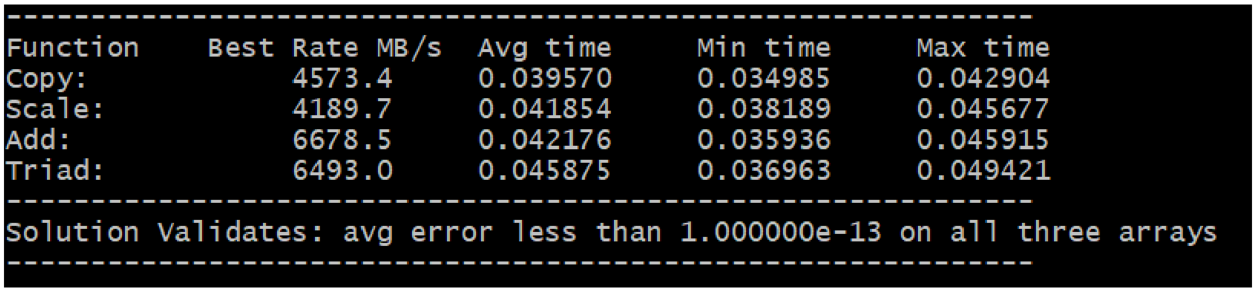
$$O + (T - 1).B < S - \frac{S}{T}$$

$$O + (T - 1).B < S(1 - \frac{1}{T})$$

$$O + (T - 1).B < \frac{S(T-1)}{T}$$

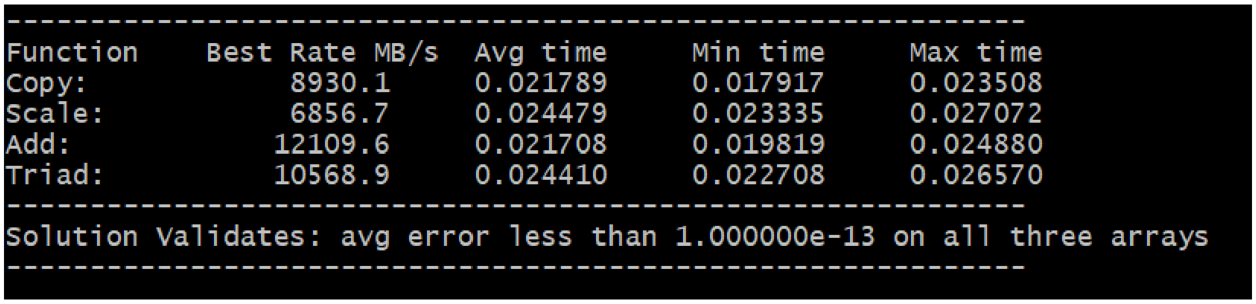
This inequality shows that parallelization is profitable for Version 3 when the overhead O and the synchronization costs (T - 1) . B are sufficiently small compared to the potential time savings from dividing the work among the threads. Specifically, as T increases, the parallel execution time decreases, but the overhead and synchronization costs must not outweigh the benefits of parallel execution for it to be worthwhile. By ensuring that O and B are minimized and the number of threads T is selected wisely, we can achieve a profitable parallel execution.

Question 3
After setting the benchmarks I saw the result and I tried to explain it in the table.
Thread 1



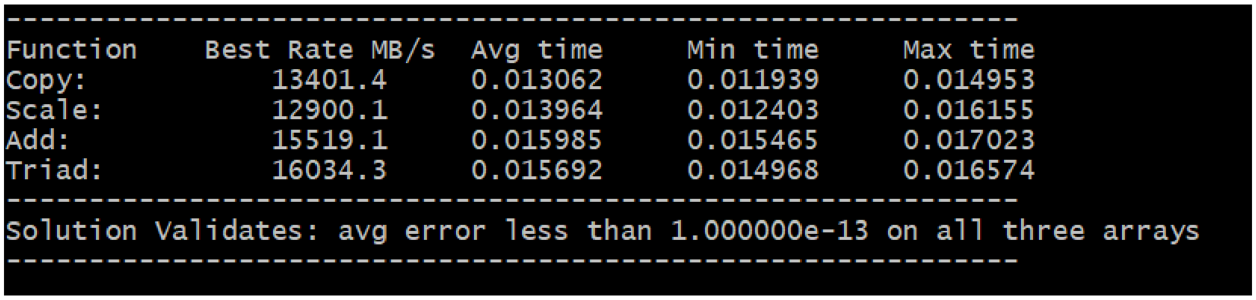
```
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:            4573.4     0.039570     0.034985     0.042904
Scale:           4189.7     0.041854     0.038189     0.045677
Add:             6678.5     0.042176     0.035936     0.045915
Triad:           6493.0     0.045875     0.036963     0.049421
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
-------------------------------------------------------------
```

Thread 2



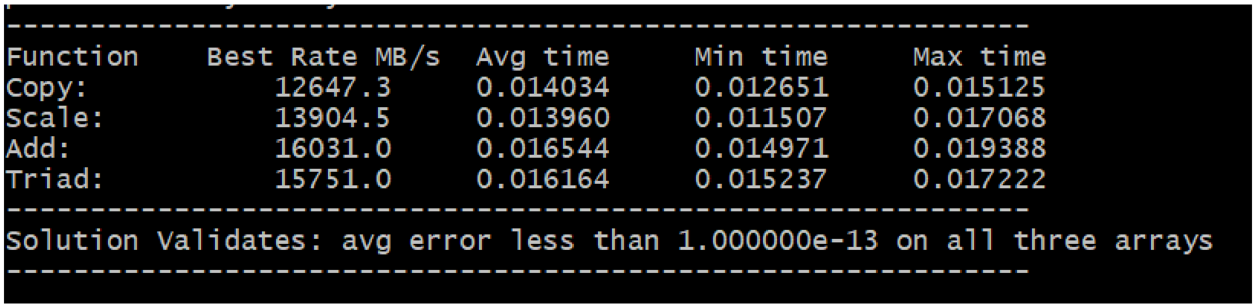
```
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:            8930.1     0.021789     0.017917     0.023508
Scale:           6856.7     0.024479     0.023335     0.027072
Add:            12109.6     0.021708     0.019819     0.024880
Triad:          10568.9     0.024410     0.022708     0.026570
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
-------------------------------------------------------------
```

Thread 4



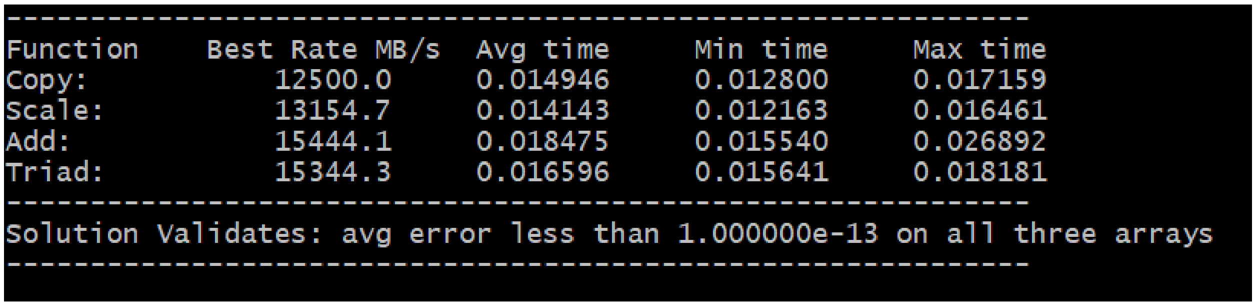
```
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:           13401.4     0.013062     0.011939     0.014953
Scale:          12900.1     0.013964     0.012403     0.016155
Add:            15519.1     0.015985     0.015465     0.017023
Triad:          16034.3     0.015692     0.014968     0.016574
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
-------------------------------------------------------------
```

Thead 8



```
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:           12647.3     0.014034     0.012651     0.015125
Scale:          13904.5     0.013960     0.011507     0.017068
Add:            16031.0     0.016544     0.014971     0.019388
Triad:          15751.0     0.016164     0.015237     0.017222
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
-------------------------------------------------------------
```

Thread 16


```
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:          12500.0     0.014946     0.012800     0.017159
Scale:         13154.7     0.014143     0.012163     0.016461
Add:           15444.1     0.018475     0.015540     0.026892
Triad:         15344.3     0.016596     0.015641     0.018181
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
-------------------------------------------------------------
```

Analysis

Scalability:- the scalability increases as the number of threads increases.

Copy:- copy increases until the thread of 4 and shows a little decrease for the reset.

Triad:- the triad increases until the triad 4 then slowly decreases.

Add:- the add increases until 4 and slowly decreases.

The average, min and max time is also presented in the fig above. The overall avg error is less than 1.00000e-13 on all three arrays.