Assignment 1 Report

1. Overview

In assignment 1, a parallelized version of Game of World (GOL) is implemented using C and MPI, and the performance of the parallel program is analyzed. Besides, two bonus features are also implemented, which are the latency hiding and the measurement of analysis of the computation and communication times.

2. Experiment Setup

- All the experiments are performed on the DAS-5 cluster.
- The parallel GOL program's performance is evaluated on two kinds of tasks: **medium task(1000x1000) and large task(2000x2000)**, with a maximum step **15000**.
- For each kind of task, the parallel GOL program's performance is tested with a different **number of processes(ranks): 2,4,8,16**
- For the scenario using X processors, the parallel GOL program's performance is tested with a different **number of nodes (1,2… X).**
- **Each task will be repeated 3 times.**
- For each individual test, the efficiency will be calculated and discussed.

Besides, two additional test are designed for the bonus feature:

- The latency hiding version of the GOL program will be tested on large tasks with 16 processes on different numbers of nodes, and compared with the original parallel GOL program.
- The computation and communication times will be measured on large tasks with different numbers of processes on one node, and the results will be reported.

3. Experiment Result and Analysis

3.1 Baseline

The sequential version of the GOL program provided by this course is run as the baseline.

Task Size	Time consumption(s) (minimum of 4 repeated run)
Medium	18.039
Large	55.564

3.2 Performance of parallel GOP program

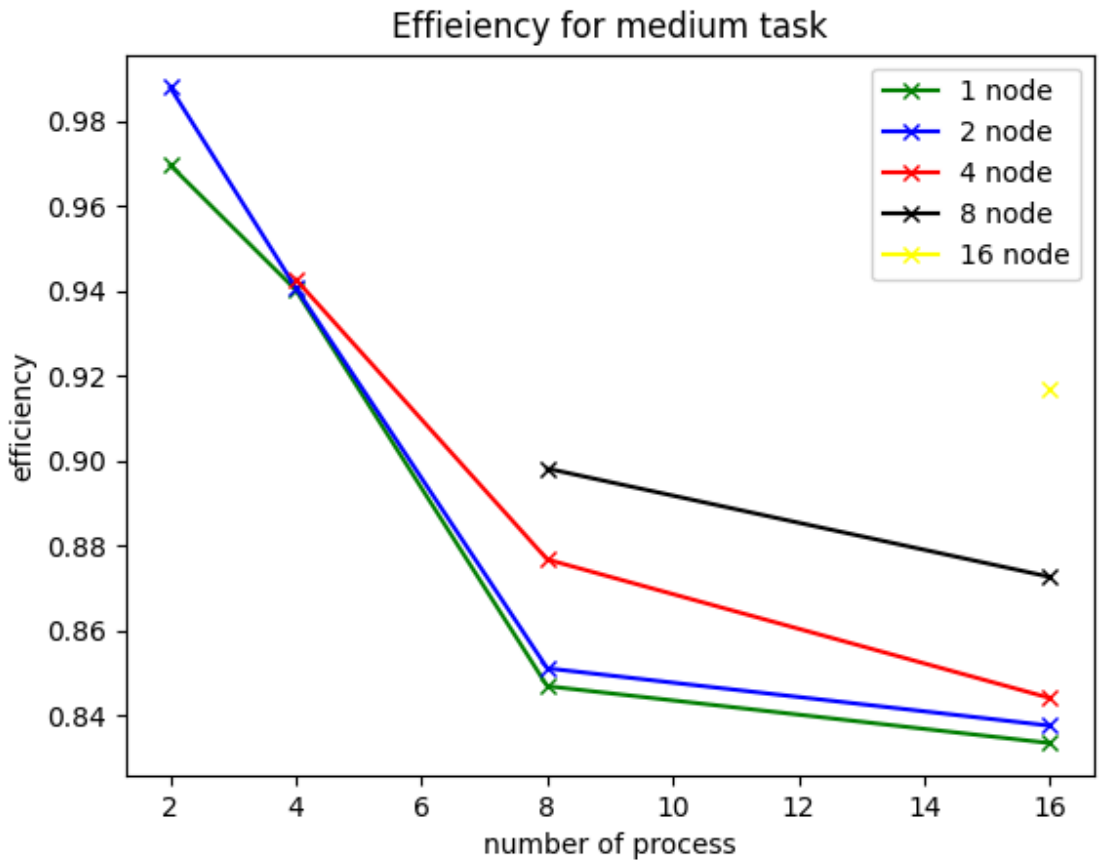
3.2.1 Performance for medium tasks

Average Time Consumption(seconds) of Parallel GOP program on Medium Task

	2 processors	4 processors	8 processors	16 processors
1 node	21.371	11.019	6.264	3.107
2 nodes	20.957	11.013	6.086	3.092
4 nodes	N/A	10.990	5.908	3.068
8 nodes	N/A	N/A	5.767	2.905
16 nodes	N/A	N/A	N/A	2.994

Efficiency of Parallel GOP program on Medium Task

	2 processors	4 processors	8 processors	16 processors
1 node	0.9696	0.9402	0.8270	0.8336
2 nodes	0.9879	0.9407	0.8512	0.8377
4 nodes	N/A	0.9427	0.8768	0.8442
8 nodes	N/A	N/A	0.8982	0.8727
16 nodes	N/A	N/A	N/A	0.8651



3.2.2 Performance for large tasks

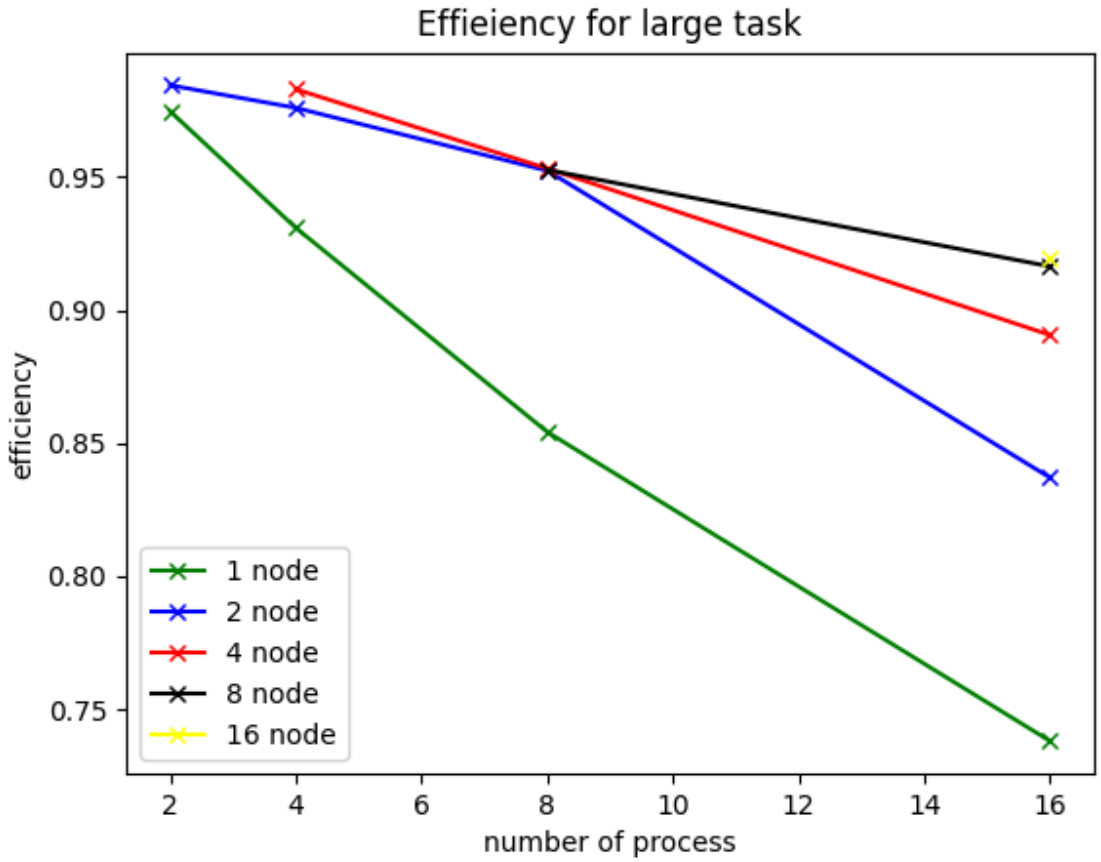
Average Time Consumption(seconds) of Parallel GOP program on Large Task

	2 processors	4 processors	8 processors	16 processors
1 node	60.677	28.980	16.531	9.566
2 nodes	57.433	28.931	14.827	8.436
4 nodes	N/A	28.728	14.815	7.928
8 nodes	N/A	N/A	14.822	7.705
16 nodes	N/A	N/A	N/A	7.679

Efficiency of Parallel GOP program on Large Task

	2 processors	4 processors	8 processors	16 processors
1 node	0.9747	0.9309	0.8542	0.7380

2 nodes	0.9845	0.976	0.9524	0.8370
4 nodes	N/A	0.983	0.9532	0.8906
8 nodes	N/A	N/A	0.9527	0.9164
16 nodes	N/A	N/A	N/A	0.9195



3.2.3 Analysis

- In both medium and large tasks, the efficiency drops when the number of processes increases. That is because that more communication overhead is generated when the number of process increases
- When the number of processes is identical, the more nodes we use, the higher efficiency we get. That is probably because of IO congestion. When all the processes are deployed more densely, the IO will be more intensive on one single machine's network interface card, causing congestion and higher communication overhead.

3.3 Performance of parallel GOP program with latency hiding

Average Time Consumption(seconds) of original and latency-hiding GOP program on Large Task with 16 processes

	16 nodes	8 nodes	4 nodes	2 nodes	1 nodes
Original	7.679	7.705	7.928	8.436	9.566
LatencyHiding	7.636	7.658	7.895	8.404	9.548

Analysis: when we use latency hiding, we can slightly increase the time consumption as well as efficiency, because latency hiding can hide some of the computational cost into communication time consumption.

3.4 Computation and Communication times

Computation and Communication Times on Medium Tasks With 1 node

#Process	16	8	4	2
Computational Cost(s)	47.69	46.80	43.44	42.86
Communication Cost(s)	5.34	3.39	2.29	0.61

Computation and Communication Times on Medium Tasks With 4 processes

#Node	4	2	1
Computational Cost(s)	41.78	43.15	42.16
Communication Cost(s)	2.04	1.65	1.93

Analysis:
1.When the number of processes increases, the communication cost increases dramatically, due to more data having to be transmitted.
2. When the number of processes increases, the computational cost also increases slightly. It is because that there are more computational tasks introduced, like copying data from message buffer, more memory allocation and initialization, as well as some calculation like merging the results from different processes.