Milestone 6

Mubanga Ben Ngosa
SFU 4th Year Student
mngosa@sfu.ca

Harwinder Dhillon
SFU 4th Year Student
hda26@sfu.ca

Abstract

In milestone 5, our focus was mainly on overcoming consistency, replication, and scalability issues. We made much progress in implementing an asynchronous system using the ZeroMQ messaging library, enhancing our slurm knowledge, and exploring dynamic node initialization. For this milestone, we have directed our efforts towards practically applying these concepts, particularly in graph processing across distributed nodes, optimizing the efficiency of our system, and ensuring correctness in distributed computations. Our work now demonstrates a more mature system capable of handling graph data in a distributed environment, marking significant progress towards actually scaling out Peregrine.

1. Progress Descriptions

1 Node Initialization and Job Distribution We refined the process of node initialization and job distribution. specifically our choice in using ZeroMQ. We chose it because ZeroMQ offers a flexible and lightweight messaging solution over something like MPI (Message Passing Interface) for our implementation mainly because of its ability to handle dynamic node creation and its compatibility with various network topologies which will allow for more straightforward integration into our existing infrastructure. in theory, our system will be able to dynamically create worker nodes based on the size and complexity of the graph being processed. This approach has strong potential for having significant improvements in processing times compared to a single-node benchmark.

2 Node Initialization and Job Distribution We achieved a major milestone in enabling worker nodes to receive job packets. The `JobPacket` struct contains the taskId, vgsCount, v, vgsi and Serielization and Deserialization methods. with this data struck we are able to send job packets to the worker nodes. We have not yet been able to process the jobs but we can confirm the jobs are being received by the worker nodes which is promising.

3 Ensuring Correctness and Avoiding Redunduncie We are yet to implement any solid methods to ensure correctness but our plan is to implement a verification system where each worker node sends back a confirmation upon completing a job, along with any relevant results or error messages. This system would help us identify and rectify any instances of job redundancy or computation errors.

Setbacks and Resolutions:

1 Time Management Challenges

Balancing project work with other obligations remains a challenge. We implemented a more structured schedule with regular sprint meetings, which significantly improved our productivity and focus.

2 SLURM Complexity

Although the complexity of SLURM initially posed a hurdle, ongoing discussions with teaching assistants and peers have provided us with valuable insights and training. We are now more proficient in utilizing SLURM for job scheduling and resource management in our setup.

For the next milestone, our goals include:

- **Performance Optimization**: We plan to focus on profiling and optimizing our code to enhance system performance, particularly in graph processing tasks.
- **Scalability Testing**: Conduct extensive testing with varying graph sizes and worker node counts to evaluate and improve the scalability of our system.
- **User Interface and Feedback Mechanism**: Develop a user-friendly interface and a feedback mechanism for users to specify scaling preferences and receive system performance data.

Acknowledgments

We extend our gratitude to our course instructors, teaching assistants, and fellow classmates for their invaluable support and insights throughout this project.

References

://www.overleaf.com/project/654efc8b2a521bac8c266ebaK. Jamshidi, R. Mahadasa. and K. Vora. ...Peregrine: A Pattern-Aware Graph Mining System...

1. Wes Kendall and Greg Lee, "MPI Hello World Tutorial," *MPITutorial.com*. Available: https://mpitutorial.com/tutorials/mpi-hello-world/https://mpitutorial.com/tutorials/mpi-hello-world/
2. Google Research, "Research Areas: Distributed Systems and Parallel Computing," *Google Research*. Available: https://research.google/research-areas/distributed-systems-and-parallel-computing/https://research.google/research-areas/distributed-systems-and-parallel-computing/
3. Thomas Weise, "Distributed Computing Examples," *GitHub*. Available: https://github.com/thomasWeise/distributedComputingExampleshttps://github.com/thomasWeise/distributedComputingExamples
4. Mikito Takada, "Distributed Systems for Fun and Profit," *Mixu's Distributed Systems Book*. Available: https://book.mixu.net/distsys/single-page.htmlhttps://book.mixu.net/distsys/single-page.html
5. ZeroMQ, "ZeroMQ - Connect Your Code," *ZeroMQ*. Available: https://zeromq.org/https://zeromq.org/
6. ZeroMQ, "ZeroMQ: Messaging for Many Applications," *YouTube*. Available: https://www.youtube.com/watch?v=UrwtQfSbrOshttps://www.youtube.com/watch?v=UrwtQfSbrOs