Forschungsprojekt

Simon König (3344789) Leon Matzner (3315161) Felix Rollbühler (3310069) Jakob Schmid (??????)

***Abstract*—In this paper we will analyze and compare the graph frameworks Galois, Ligra and Polymer in their performance. We will also set up and compare these frameworks in a distributed scenario.**

***Index Terms*—Galois, Ligra, Polymer, distributed computing, Gluon**

I. Introduction

My intro... blah blah [**?**].

II. Graph Formats

A rather big portion of our time was invested in figuring out which graph framework requires which graph formats. We thus decided to give an overview over all the formats we encountered, with explanation on how they represent the graph.

Additionally, to make life in the future a little bit easier, we wrote multiple tools to convert graphs acquired from Snap or Konect to the required formats. Information on how to use those tools is available in our repository.

A. AdjacencyList

The AdjacencyList and WeightedAdjacencyList formats[**?**] are used by Ligra and Polymer. They represent the directed edges of a graph as a number of offsets that point to a set of target nodes in the file. First the file contains the number of vertices n and edges m, followed by an offset for each vertex. This offset specifies at what point in the following list of numbers the information for a node begins. Lastly the file format contains a list of target nodes. The numbers are all separated by newlines.

$$\begin{array}{c} n \\ m \\ o_1 \\ o_2 \\ \vdots \\ o_n \\ t_1 \\ t_2 \\ \vdots \\ t_m \end{array}$$

The offsets $o_i = k$ and $o_{i+1} = k + j$ mean that vertex i has j outgoing edges, these edges are

$$(i, t_k), (i, t_{k+1}), \ldots, (i, t_{k+j-1})$$

For the WeightedAdjacencyList format, the weights are just appended to the end of the file in the same order as the edges.

B. EdgeList

The EdgeList format is probably the easiest to understand and is one of the most commonly used in the online graph repositories. The directed eges $(s_1, t_1), (s_2, t_2), \ldots$ are represented in the following way.

$$\begin{array}{c} s_1, t_1 \\ s_2, t_2 \\ \vdots \\ s_m, t_m \end{array}$$

C. Binary EdgeList

The binary EdgeList format is used by Gemini. Finding information on this format required reverse engineering of the Gemini code.

We found that Gemini requires the following input format

$$s_1 t_1 w_1 s_2 t_2 w_2 \ldots$$

where s_i, t_i have `uint32` ?? data type and the weights are `float32`. Gemini will derive the number of edges from the file size, so there is no file header or anything similar allowed.

D. ?

III. Things

Galois [**?**] is a general purpose library designed for parallel programming. The Galois system supports fine grain tasks, allows for autonomous, speculative execution of these tasks and grants control over the task scheduling policies to the application. It also simplifies the implementation of parallel applications by providing an implicitly parallel unordered-set iterator.

For graph analytics purposes a topology aware work stealing scheduler, a priority scheduler and a library of scalable data structures have been implemented. Galois includes applications for for many graph analytics problems, among these are single-source shorthest-paths (sssp) and pagerank. Both of these applications can be executed in shared memory systems and due to the Gluon integration in a distributed environment. Gluon [**?**] reduces the communication overhead needed in distributed systems for graph analysis by exploiting structural and temporal invariants.

A. Polymer

Installing Polymer is straight forward.

IV. Conclusion

The conclusion goes here.

ACKNOWLEDGMENT

We are using the graph frameworks Galois [**?**], Ligra [**?**] and Polymer [**?**].

Also we use Gluon [**?**] for the distributed setups.

Gemini [**?**]

REFERENCES

[1] K. Zhang, R. Chen, and H. Chen, "Numa-aware graph-structured analytics," in *Proceedings of the 20th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, ser. PPoPP 2015. New York, NY, USA: Association for Computing Machinery, 2015, p. 183–193. [Online]. Available: https://doi.org/10.1145/2688500.2688507

[2] J. Shun, G. Blelloch, J. Fineman, P. Gibbons, A. Kyrola, K. Tangwonsan, and H. V. Simhadri. (2020, Jun.) Problem Based Benchmark Suite. graphIO.html. [Online]. Available: http://www.cs.cmu.edu/~pbbs/benchmarks/

[3] D. Nguyen, A. Lenharth, and K. Pingali, "A lightweight infrastructure for graph analytics," in *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, ser. SOSP '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 456–471. [Online]. Available: https://doi.org/10.1145/2517349.2522739

[4] J. Shun and G. E. Blelloch, "Ligra: A lightweight graph processing framework for shared memory," in *Proceedings of the 18th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, ser. PPoPP '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 135–146. [Online]. Available: https://doi.org/10.1145/2442516.2442530

[5] R. Dathathri, G. Gill, L. Hoang, H.-V. Dang, A. Brooks, N. Dryden, M. Snir, and K. Pingali, "Gluon: A communication-optimizing substrate for distributed heterogeneous graph analytics," in *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI 2018. New York, NY, USA: Association for Computing Machinery, 2018, p. 752–768. [Online]. Available: https://doi.org/10.1145/3192366.3192404

[6] X. Zhu, W. Chen, W. Zheng, and X. Ma, "Gemini: A computation-centric distributed graph processing system," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. Savannah, GA: USENIX Association, Nov. 2016, pp. 301–316. [Online]. Available: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/zhu