Modeling Single-Source Shortest Path Algorithm Dynamics to Control Performance and Power Tradeoffs

Sara Karamati
School of Computational Science and Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: s.karamati@gatech.edu

Jeffrey Young (advisor)
School of Computational Science and Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: jyoung9@gatech.edu

Rich Vuduc (advisor)
School of Computational Science and Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332–0250
Email: richie@cc.gatech.edu

I. Introduction

In this work, we apply dynamical system modeling as a new methodology to tune the performance and power characteristics of an algorithm with irregular dynamics in order to control its runtime behavior. The need to control software behavior arises at all scales of platforms—whether they be mobile and embedded devices, data centers, or high-end supercomputers–as a way to control factors like the amount of thermal dissipation in the system or its overall operating energy cost. One major challenge in controlling the software behavior, especially in high-performance applications, occurs when the software exhibits a non-stationary, dynamic amount of parallelism (and subsequently power/core usage) during its runtime. Controlling such algorithms requires modeling the algorithmic demands on resources in each step and to accordingly tune the algorithm in real time to fit as much computation as possible within the given operational limits. However, modeling an irregular algorithm is not trivial since its dynamic behavior changes over time.

II. Overview of Our Approach

In this work, we examine one such non-trivial case, i.e. a GPU-based parallel single-source shortest path (SSSP) from the Gunrock analytics library [1]. We are motivated by an application like SSSP because it is compact enough to study in detail while also having challenging characteristics: its concurrency and resulting performance and power behavior depend on the input graph and can be highly irregular. Our approach for a tunable algorithm [2] derives from the baseline method, which is the near-far SSSP algorithm [3], a variation on delta-stepping that was developed and first implemented in the Gunrock library for graph analysis on graphics coprocessor (GPU) systems. The near-far method has a natural tuning parameter, delta, which a user must select manually and which remains fixed throughout the execution. The parallelism profile of the SSSP algorithm with fixed delta shows the variability in available parallelism is high (see Figure 1). High variability in parallelism wastes time when threads are underutilized,

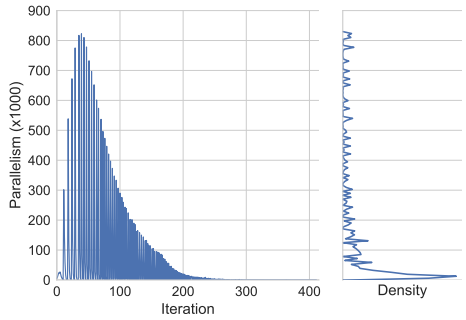
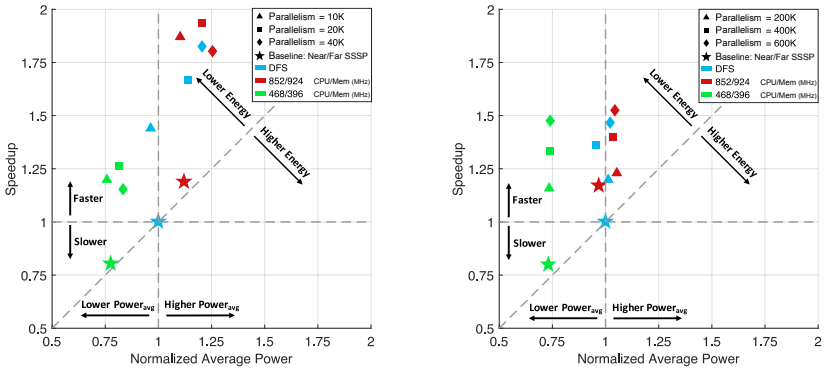


Fig. 1. Concurrency profiles for the baseline Gunrock SSSP

and wastes energy (Joules) as idle cores consume their base power. Our proposed method is an adaptive near+far SSSP algorithm with a feedback loop that dynamically adjusts delta so that the amount of available parallelism converges to values around a set-point P, which is the desired amount of parallelism. Informed by an analysis of this algorithm's runtime characteristics, we develop a mathematical model that uses online learning techniques based on stochastic gradient descent to predict the relation between available parallelism in each iteration and the tunable delta parameter. We then use this model to tune the available parallelism dynamically to meet a given target, thereby improving the average available parallelism while reducing its variability.

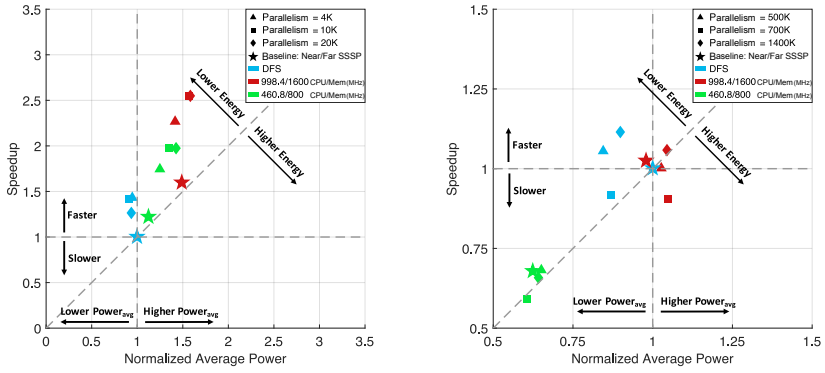
III. Experimental Results

We experimentally show that this controller can be used to limit power consumption on an NVIDIA Tegra TK1 and TX1 platform, can manage power-performance tradeoffs in software, and can even save power while increasing performance in certain cases. Such a capability would not only complement existing methods for hardware performance tuning and power capping but also would allow us to see what happens when we combine different power-control mechanisms. Our preliminary experiments show that our proposed method can augment,



(a) Cal (road network) (b) Wiki (scale-free network)

Fig. 2. Performance versus power (TK1)



(a) Cal (road network) (b) Wiki (scale-free network)

Fig. 3. Performance versus power (TX1)

and even work synergistically with, hardware-based dynamic voltage and frequency scaling (DVFS) techniques. The data sets used in our experiments are a high-diameter and low-degree road network, Cal [4], and a high-degree and low-diameter scale-free network, wikipedia-20051105 [5]. For the Cal road network, at different frequency settings, controlled parallelism improves the performance and power consumption over the baseline (see figure 2(a) and figure 3(a)). For all the frequency levels, the controller actually improves the performance when tuning to meet the parallelism set-point. In fact, most of the self-tuning points are both faster and more energy efficient than the baseline algorithm. The results demonstrate that self-tuning does not necessarily improve power usage over the baseline with DVFS, although it does enable additional speedup at the same system power. For the Wiki network, figures 2(b) and 3(b), DVFS can provide either a 25% power savings or a 20% speedup, but it cannot by itself achieve the maximum combined power savings and speedups seen with algorithmic knobs, which yield a 50% speedup with a 25% power savings.

IV. Conclusion

While we only demonstrated a concrete implementation and results of our controller idea for SSSP, we believe the same ideas are relevant to other graph implementations. Extending and testing the controller model in all such cases will be part of our future work.

References

[1] Y. Wang, A. A. Davidson, Y. Pan, Y. Wu, A. Riffel, and J. D. Owens, "Gunrock: A high-performance graph processing library on the GPU," *CoRR*, vol. abs/1501.05387, 2015. [Online]. Available: http://arxiv.org/abs/1501.05387

[2] S. Karamati, J. Young, and R. Vuduc, "An energy-efficient single-source shortest path algorithm," in *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2018, pp. 1080–1089.

[3] A. Davidson, S. Baxter, M. Garland, and J. D. Owens, "Work-efficient parallel gpu methods for single-source shortest paths," in *2014 IEEE 28th International Parallel and Distributed Processing Symposium*, May 2014, pp. 349–359.

[4] C. Demetrescu, A. V. Goldberg, and D. S. Johnson, *The Shortest Path Problem: Ninth DIMACS Implementation Challenge*. American Mathematical Soc., vol. 74.

[5] T. A. Davis and Y. Hu, "The university of florida sparse matrix collection," *ACM Trans. Math. Softw.*, vol. 38, no. 1, pp. 1:1–1:25, Dec. 2011. [Online]. Available: http://doi.acm.org/10.1145/2049662.2049663