CS 259 Fall 2019: Class Projects
Jason Cong

Some important dates

- Project selection due by (11/6 12pm); Please mark your choice at https://docs.google.com/spreadsheets/d/1iTXbYh7mpbJWf-FxKU_2e60p0Kj_oTiB6S3Ev-lUsPA/edit#gid=0 (first come first serve)
- Project background presentation due (11/13 8am)
- Project progress presentation due (12/2 8am)
- Project report due (12/12 12pm)

Project Candidate Topics

Topic 1: Graph Convolutions

Graph Processing Frameworks

- Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein, "GraphLab: A new framework for parallel machine learning," in UAI, 2010.
- R. R. McCune, T. Weninger, and G. Madey, "Thinking like a vertex: A survey of vertex-centric frameworks for large-scale distributed graph processing," ACM Comput. Surv., vol. 48, no. 2, Oct. 2015.
- Roy, I. Mihailovic, and W. Zwaenepoel, "X-stream: edge-centric graph processing using streaming partitions," in SOSP. ACM, 2013.
- X. Zhu, W. Han, and W. Chen, "Gridgraph: Large-scale graph processing on a
- single machine using 2-level hierarchical partitioning," in USENIX ATC. Santa Clara, CA: USENIX Association, 2015.
- Kyrola, G. Blelloch, and C. Guestrin, "GraphChi: Large-scale graph computation on just a pc," in OSDI. Hollywood, CA: USENIX, 2012.

Graph Convolutional Networks.

- Thomas N. Kipf and MaxWelling. Semi-supervised classification with graph convolutional networks. In 5th International Conference on Learning Representations (ICLR-17). https://arxiv.org/pdf/1609.02907.pdf (GCN Paper, ICLR 2017)
- William L. Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA, pages 1025–1035, 2017.
- Jie Chen, Tengfei Ma, and Cao Xiao. FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling. In 6th International Conference on Learning Representations (ICLR-18).

Two versions of implementation
https://github.com/tkipf/gcn (GCN Tensorflow Implementation)
https://github.com/tkipf/pygcn (GCN PyTorch Implementation)

Hardware Acceleration of Graph Processing and GCN:

- Eriko Nurvitadhi, Gabriel Weisz, Yu Wang, Skand Hurkat, Marie Nguyen, James C. Hoe, Jos´e F Mart´ınez, and Carlos Guestrin. GraphGen: An FPGA Framework for Vertex-Centric Graph Computation. In FCCM, pages 25–28, 2014.
- Zhiyuan Shao, Ruoshi Li, Diqing Hu, Xiaofei Liao, and Hai Jin. Improving Performance of Graph Processing on FPGA-DRAM Platform by Two-level Vertex Caching. In FPGA, pages 320–329, 2019.
- Guohao Dai, Tianhao Huang, Yuze Chi, Ningyi Xu, Yu Wang, and Huazhong Yang. ForeGraph : Exploring Large-scale Graph Processing on Multi-FPGA Architecture. In FPGA, New York, New York, USA, 2017. ACM Press.
- Shijie Zhou, Rajgopal Kannan, and Viktor K Prasanna. HitGraph : High-throughput Graph Processing Framework on FPGA. TPDS, 2019.
- Sang-Woo Jun, Andy Wright, Sizhuo Zhang, Shuotao Xu, and Arvind, "GraFBoost: Using accelerated flash storage for external graph analytics," 2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture.

Topic 2: Acceleration of Bayesian Networks

Bayesian Modeling

- Adnan Darwiche, "A Differential Approach to Inference in Bayesian Networks," J. of ACM, 2003.
- Adnan Darwiche, *Modeling and Reasoning with Bayesian Networks*, Cambridge University Press, 2009 (Chapter 12, electronic version free for UCLA students through library)
- Yujia Shen, Haiying Huang, Arthur Choi, Adnan Darwiche, Conditional Independence in Testing Bayesian Networks, Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.
- Arthur Choi,, Ruocheng Wang, Adnan Darwiche, On the relative expressiveness of Bayesian and neural networks, Int'l J. of Approximate Reasoning, 2019, pp 303-323.

Hardware Acceleration

- Mingjie Lin, et al., "High-Throughput Bayesian Computing Machine with Reconfigurable Hardware", FPGA'2010.
- Michael D. Linderman, et al. "High-throughput Bayesian Network Learning using Heterogeneous Multicore Computers," ICS 2010.
- I Pournara, et al. "FPGA-accelerated Bayesian learning for reconstruction of gene regulatory networks," FPL 2005.

Topic 3. System-Level Acceleration

Workload Analysis and Software Solutions

- Svilen Kanev, et al. "Profiling a warehouse-scale computer," ISCA 2015
- Skyway: Connecting Managed Heaps in Distributed Big Data Systems, Nguyen et al., ASPLOS 2018 https://dl.acm.org/citation.cfm?id=3173200
- Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks, ASPLOS'18 https://dl.acm.org/citation.cfm?id=3173177
- SpaceJMP: Programming with Multiple Virtual Address Spaces. ASPLOS'16 https://dl.acm.org/citation.cfm?id=2872366

Hardware Acceleration

- D. F. Bacon, P. Cheng, and S. Shukla, "And then There Were None: A Stall-free Real-time Garbage Collector for Reconfigurable Hardware," Commun. ACM, vol. 56, no. 12, pp. 101–109, Dec. 2013.
- Martin Maas, Krste Asanovic, John Kubiatowicz, A Hardware Accelerator for Tracing Garbage Collection, the Proceedings of the 45th ACM/IEEE International Symposium on Computer Architecture, 2018.
- Irina Calciu, et al, "Project PBerry: FPGA Acceleration for Remote Memory," HotOS '19, May 13–15, 2019, Bertinoro, Italy

Project Submission Requirements

1. Each team has two members, and needs to submit one report only.
2. Please email me your report by the project submission deadline. There will be 5% penalty for each 24-hour delay, up to 48 hours.
3. A typical report has a minimum of 10 pages (font size 11) in the form of a written report (you also need around 50+ slides for two project presentations covering the same materials below). Please make sure that your report has the following 10 sections:
   1) General description of your application or application domain for acceleration
   2) Discussion of related work on this topic (describing in detail the ones you covered in your presentation and make references of the rest covered in other presentations)
   3) Detailed discussion of the algorithm and the reference code you used, profiling result, and your decision of accelerator selection
   4) Detailed description of the micro-architecture of the accelerators that you have chosen and justification of your design
   5) Please discuss the expected performance of your accelerator, and identify the performance bottlenecks of the overall design (computation or communication? Which level of communication, at PCI-e or DRAM?, etc)
   6) Please include a flowchart of the tools you used to design and integrate your accelerator.
   7) Please discuss the verification flow that you used or plan to use (for correctness)
   8) Please discuss the 1-3 most significant challenges you encountered and how you overcome them.
   9) Please include 1-2 paragraphs for possible extensions.
   10) Please identify the contributions from each team member
   11) If you do complete the accelerator design and implementation, please discuss the code modification you made and pragma you used.
   12) A table showing the actual speedup using the selected accelerators for each input data for your application, and the final performance for each input. The geometric mean of each metric (speedup, power reduction, etc) for all inputs should be used for reporting. You need to include the details of both CPU results (e.g. if you have used multi-core for parallelization), and the FPGA results (including both the performance and area utilization of LUTs, FFs, DSPs, BRAMs, etc).

4. You will be required to give two presentations about the above contents, one in the 7-8th week, and another in the 10th week.

5. The code submission includes a directory that includes your programs, including both the source code, the executable code, input data and the results.

Your final submission should be a tarball which contains the following files:

<Your UID>.tar.gz

```
├ source_package.tar.gz
└ report.pdf
```