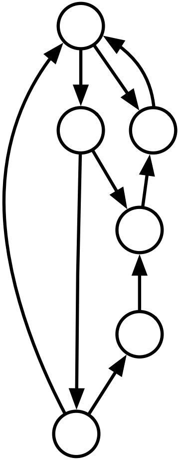
Speeding Up GPU Graph Processing Using Structural Graph Properties

Merijn Verstraaten
Advisor: Ana Lucia Varbanescu
Promotor: Cees de Laat

University of Amsterdam

Graphs Analytics

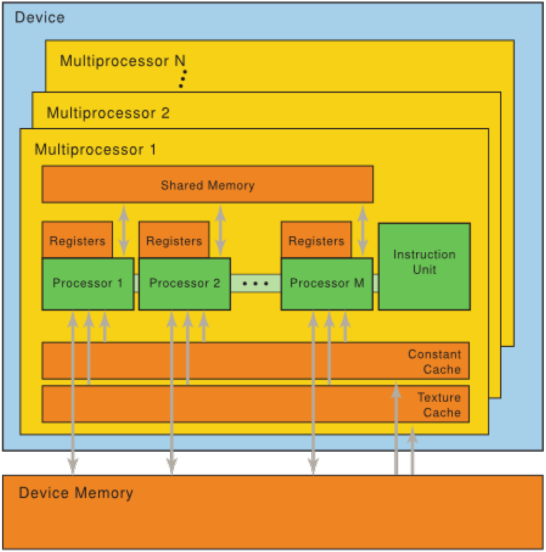


Graphs:

- Vertices
- Edges

Who cares?

General Purpose GPU Computing



My Work So Far…

Systematic Benchmarking (Graph Generator + Real World Datasets)

Analytical Modeling
- Multiple PageRank implementations
- Sequential workload model
- Parallel execution model

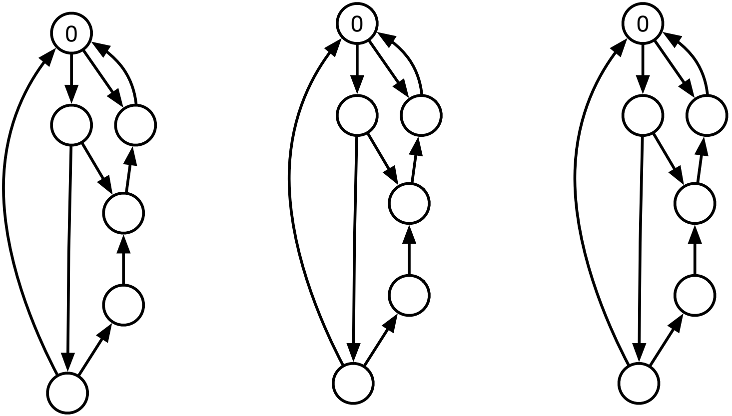
Graph Generator
- Vary individual parameters
- Evolutionary graph generator
- Scaling to graph generation to large sizes

Real World Datasets
- Many different BFS implementations
- Benchmark on SNAP & KONECT
- Different vertex/ edge orderings
- Per level timings

Machine Learning
- Determine important parameters
- Predict fastest implementation
- Implementation switching BFS

Breadth-First Search: Implementations

Edge-centric | **Vertex Push** | **Vertex Pull**



Useful Frontier Thread

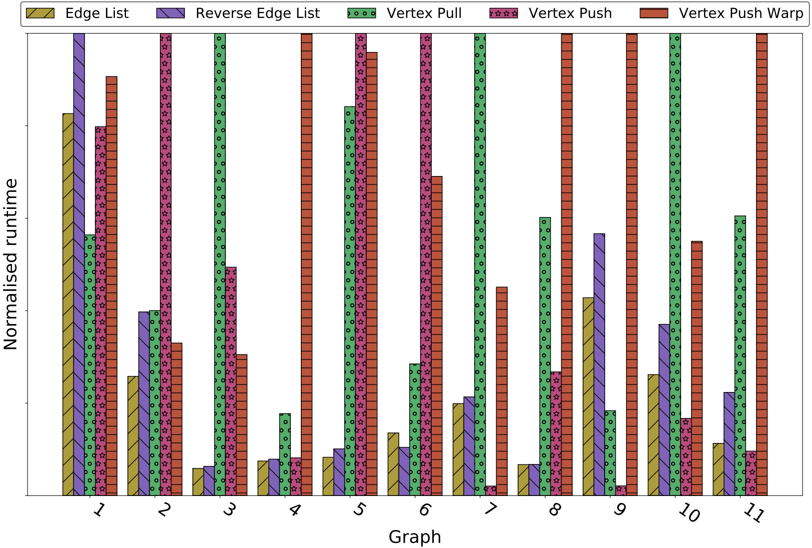
Useless Frontier Thread

Frontier Node

Updated Node

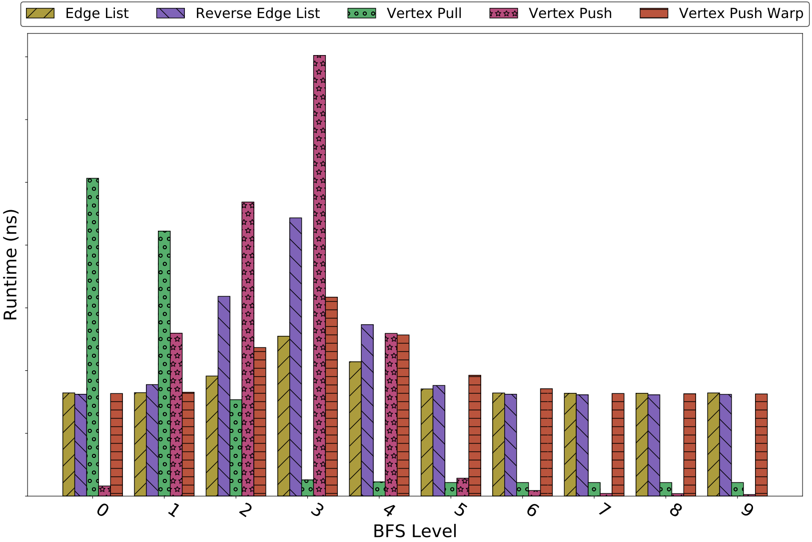
Accessed Node

Relative Performance of Implementations



There is no “best”!

Relative Performance Within a Single Traversal



Sticking to one implementation costs us!

Predicting the Best Implementation

Weapon of Choice: Decision Trees

Features:

- black-box approach
- predictive power and high accuracy
- require small number of samples

Training Parameters:

- Degree distribution
- Frontier size
- Percentage discovered
- Vertex count
- Edge count

Trained Models

Feasibility:

Accuracy:	$\sim$98%
Avg. Prediction Time:	144 ns ($\sigma = 165$ ns)
Min. BFS Step:	20 ms
(Re)loading graph representation:	Stupidly slow

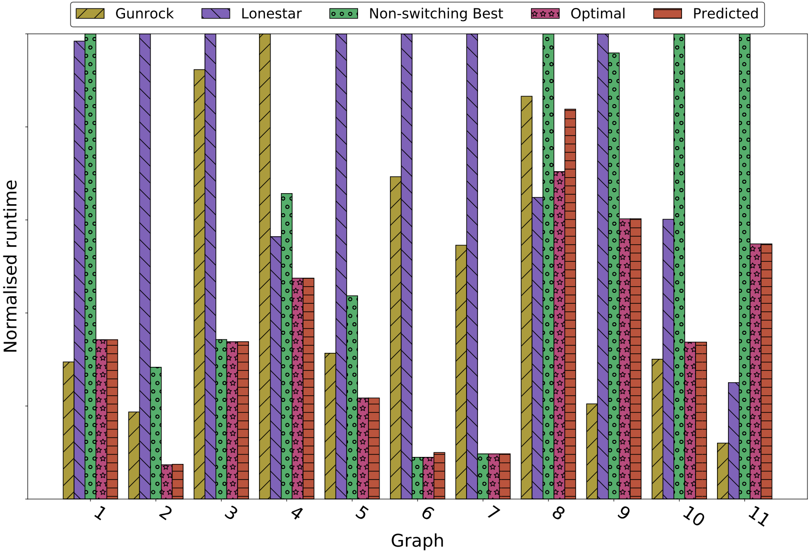
Classic time-space trade-off.

Overall Results

Algorithm	Optimal	1–2×	>5×	>20×	Average	Worst
Predicted	**56%**	**41%**	1%	0.5%	**1.40×**	236×
Oracle	23%	55%	2%	0%	1.65×	8.5×
Edge List	10%	61%	7%	0.4%	2.22×	38×
Rev. Edge List	5%	59%	15%	0.6%	2.92×	50×
Vertex Pull	0%	15%	27%	24%	38.62×	2,671×
Vertex Push	9%	15%	53%	29%	39.66×	1,048×
Vertex Push Warp	0%	0%	3%	30%	18.69×	97×

Averaged over 248 KONECT graphs.

Comparison with State-of-the-Art: Best & Worst



Even better if we include Gunrock in model?

Related Work

Single Node:
Boost Graph Library (BGL), GraphMat, Ligra

Distributed Systems:
Giraph, GraphLab, GraphX, PGX.D, Pregel

GPU Frameworks:
CuSha, Gunrock, MapGraph, Medusa, nvGraph

Hybrid Systems:
Galois, Totem

Takeaway

No single best implementation for irregular GPU algorithms

Large potential performance gains for graph algorithms

Not all machine learning leaves you clueless

Variable importance can guide analytical modelling

Questions?

Publications I

- S. Hong, S. Depner, T. Manhardt, J. Van Der Lugt, **Verstraaten, Merijn**, and H. Chafi.
  PGX.D: a fast distributed graph processing engine.
  In *High Performance Computing, Networking, Storage and Analysis, 2015 SC-International Conference for*, pages 1–12. IEEE, 2015.
- S. Hong, T. Manhardt, J. van der Lugt, **Verstraaten, Merijn**, and H. Chafi.
  Distributed graph processing system that support remote data read with proactive bulk data transfer, Apr. 3 2015.
  US Patent App. 14/678,358.
- A. Iosup, M. Capotă, T. Hegeman, Y. Guo, W. L. Ngai, A. L. Varbanescu, and **Verstraaten, Merijn**.
  Towards benchmarking IaaS and PaaS clouds for graph analytics.
  In *Workshop on Big Data Benchmarks*, pages 109–131. Springer, Cham, 2014.
- K. MacKenzie, P. K. Hölzenspies, K. Hammond, R. Kirner, V. T. N. Nguyen, I. te Boekhorst, C. Grelck, R. Poss, and **Verstraaten, Merijn**.
  Statistical performance analysis of an ant-colony optimisation application in S-Net.
  In *2nd Workshop on Feedback-Directed Compiler Optimization for Multi-Core Architectures*, 2013.

Publications II

- W. L. Ngai, A. L. Varbanescu, and **Verstraaten, Merijn**.
  Towards benchmarking IaaS and PaaS clouds for graph analytics.
  In *Big Data Benchmarking: 5th International Workshop, WBDB 2014, Potsdam, Germany, August 5-6-2014, Revised Selected Papers*, volume 8991, page 109. Springer, 2015.
- R. Poss, **Verstraaten, Merijn**, F. Penczek, C. Grelck, R. Kirner, and A. Shafarenko.
  S+Net: extending functional coordination with extra-functional semantics.
  arXiv preprint arXiv:1306.2743, 2013.
- R. Poss, **Verstraaten, Merijn**, and A. Shafarenko.
  Towards S+Net: compositional extra-functional specification for large systems, 2013.
- **Merijn Verstraaten**, A. L. Varbanescu, and C. de Laat.
  Using graph properties to speed-up gpu-based graph traversal: A model-driven approach, 2017.
- **Verstraaten, Merijn**, C. Grelck, M. W. van Tol, R. Bakker, and C. R. Jesshope.
  On mapping distributed S-Net to the 48-core Intel SCC processor.
  In *3rd MARC Symposium, Fraunhofer IOSB, Ettlingen, Germany*, 2011.

Publications III

- **Verstraaten, Merijn**, S. Kok, R. Poss, C. Grelck, et al.
  Task migration for S-Net/LPEL.
  In *2nd HiPEAC Workshop on Feedback-Directed Compiler Optimization for Multicore Architectures (FD-COMA'13), Berlin, Germany. HiPEAC*, 2013.
- **Verstraaten, Merijn** and S.-B. Scholz.
  On predicting the impact of resource redistributions in streaming applications.
  In *Proceedings of ACM SIGPLAN International Workshop on Libraries, Languages, and Compilers for Array Programming*, page 76. ACM, 2014.
- **Verstraaten, Merijn**, A. L. Varbanescu, and C. de Laat.
  Quantifying the performance impact of graph structure on neighbour iteration strategies for pagerank.
  In *European Conference on Parallel Processing*, pages 528–540. Springer, Cham, 2015.
- **Verstraaten, Merijn**, A. L. Varbanescu, and C. de Laat.
  Synthetic graph generation for systematic exploration of graph structural properties.
  In *European Conference on Parallel Processing*, pages 557–570. Springer, Cham, 2016.

Publications IV

- J. van der Lugt, **Verstraaten, Merijn**, S. Hong, and H. Chafi.
  Method of achieving intra-machine workload balance for distributed graph-processing systems, May 21 2015.
  US Patent App. 14/718,430.
- M. W. Van Tol, R. Bakker, **Verstraaten, Merijn**, C. Grelck, C. R. Jesshope, et al.
  Efficient memory copy operations on the 48-core Intel SCC processor.
  In *3rd Many-core Applications Research Community (MARC) Symposium*, volume 7598. KIT Scientific Publishing, 2011.
- A. L. Varbanescu, **Verstraaten, Merijn**, C. de Laat, A. Penders, A. Iosup, and H. Sips.
  Can portability improve performance?: An empirical study of parallel graph analytics.
  In *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering*, pages 277–287. ACM, 2015.