Bounded Asynchrony and Nested Parallelism for Scalable Graph Processing

Adam Fidel, Nancy M. Amato, and Lawrence Rauchwerger
Parasol Lab, Department of Computer Science and Engineering, Texas A&M University,
http://parasol.tamu.edu/

Parasol — Smarter computing. Texas A&M University

CERT — Center for Exascale Radiation Transport

COMPUTER SCIENCE & ENGINEERING — TEXAS A&M UNIVERSITY

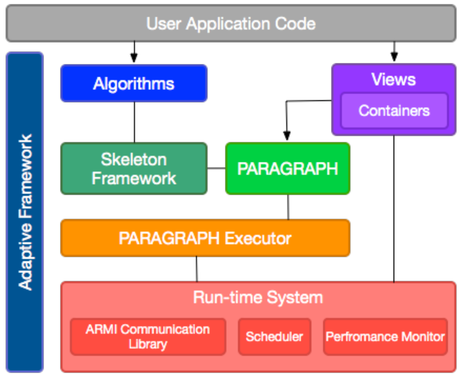
Standard Template Adaptive Parallel Library (STAPL)

A framework for developing parallel C++ code

- A library of C++ components with interfaces similar to the (sequential) C++ Standard Template Library (STL)
- Open source: http://gitlab.com/parasol-lab/stapl

Project Goals

- **Ease of use** - shared object programming model provides consistent interface across shared or distributed memory systems
- **Efficiency** - Application building blocks based on C++ STL constructs and extended, automatically tuned for parallel execution
- **Portability** - ARMI runtime system hides machine specific details and provides an efficient, uniform communication interface.



STAPL Graph Library

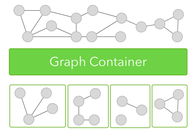
SGL Model – vertex centric programming model that emphasizes fine-grained parallelism

- **Vertex operator**: invoked on a single vertex and describes work for that particular vertex with possible neighbor visits
- **Neighbor operator**: payload for neighbor visit

```
Function VertexOperator(v)
 if v.color = GREY then
  v.color = BLACK
  VisitAllNeighbors(v, NeighborOp, v.dist+1, v.id)
  return true
 else
  return false
end
```

```
Function NeighborOperator(u, dist, parent)
 if u.dist > dist then
  u.dist ← dist
  u.parent ← parent
  u.color ← GREY
  return true
 else
  return false
end
```

Parallel Graph Container – a distributed graph with methods to access vertices and edges that provides a shared-object view to users



- **SOV** provides uniform access to data independent of physical location
- Abstracts data distribution and communication

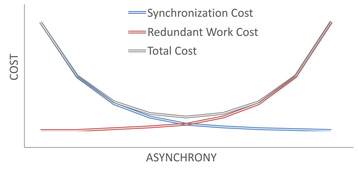
Execution Policies

- **k-level-asynchronous [PACT '14]** – generalization of level-synchronous and asynchronous
- **Hierarchical comm. reduction [PACT '15]** – Use machine hierarchy to optimize graph communication patterns
- **Out-of-core [IPDPS '15]** – Subgraph paging to disk

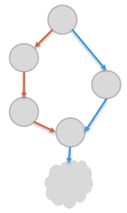
Bounded Asynchrony

k-level-asynchronous Paradigm (KLA) **[PACT'14]**

- Generalization of level-synchronous and asynchronous
  - **Level synchronous**: BSP-style iterative computation, global synchronizations
  - **Asynchronous**: point-to-point dependencies, possible redundant work
  - **KLA**: traverse up to k hops before synchronizing

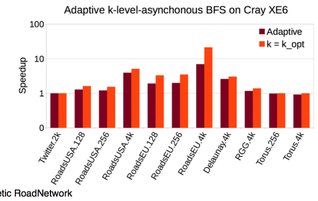


- Balances synchronization cost with redundant work
- Redundant work due to multiple paths with lack of ordering
- Applicable to a wide range of algorithms including breadth-first search, PageRank, connected components, etc.
- **Problem:** Redundant work – visiting vertices out of order (high k) will result in having to redo work

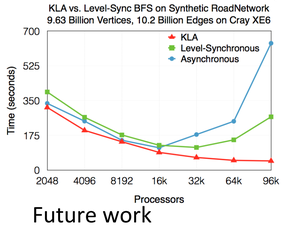


Finding the right value of asynchrony

- Optimal value of k depends on the type of graph, the machine and the algorithm
- Adaptively tune k during execution



- Adaptive selection is competitive with oracle



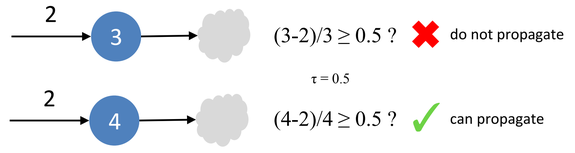
- Level-sync and async stop scaling after 32k cores
- KLA scales up to 96k cores

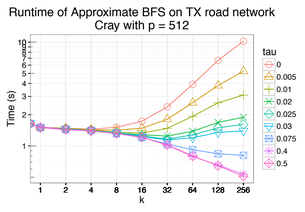
Future work

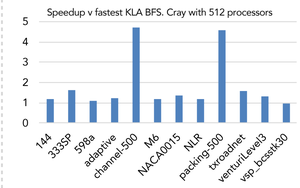
- Identify a set of graph properties, algorithm properties and architecture that enable a performance benefit with KLA
- Dynamic (streaming) graph computations
- Heterogeneous graph processing using accelerators

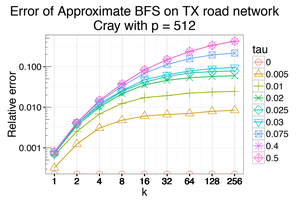
Approximation Through Asynchrony

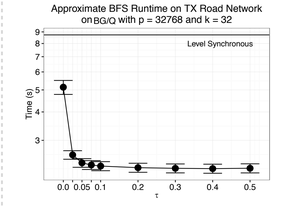
- **Idea:** Only redo work if new distance is sufficiently better
- **First approach with breadth-first search [LCPC '16]**
- Allow vertex distance to contain some error
  - Configurable parameter for tolerance τ
  - **Propagate new distance if $(d - d_{new})/d \geq \tau$**







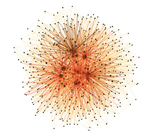


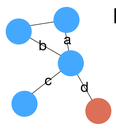


Nested Parallelism

Many real-world graphs are scale-free

- Presence of "hub" vertices connected to most of graph
- Load imbalance when processing visits
- May not fit into main memory of single machine



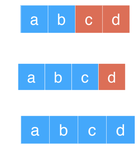


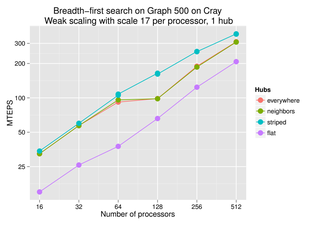
Distribute Edges + Dynamic Nested Parallelism [LCPC '15]

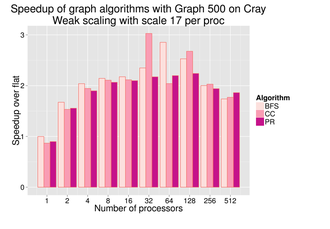
- Nested parallel visitation to process a vertex's neighbors
- Provide several strategies for distributing the edges of hub vertices, that can be seamlessly interchanged.

Hub distributions

- Randomized-balance: balanced partition across all processors
- Neighbors: place an edge (s,t) on the same processor as t
- Hierarchical: use one processor in each shared-memory node







References and Aknowledgements

This research supported in part by NSF awards CNS-0551685, CCF 0702765, CCF-0833199, CCF-1439145, CCF-1423111 CCF-0830753 IIS-0916053, IIS-0917266, EFRI-1240483, RI-1217991, by NIH NCI R25 CA090301-11, and by DOE awards DE-AC02-06CH11357, DE-NA0002376, B575363. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

- **"Fast Approximate Distance Queries in Unweighted Graphs using Bounded Asynchrony,"** Adam Fidel, Francisco Coral, Colton Riedel, Nancy M. Amato, Lawrence Rauchwerger, In Wkshp. on Lang. and Comp. for Par. Comp. (LCPC), Sep 2016.
- **"KLA: A New Algorithmic Paradigm for Parallel Graph Computations,"** Harshvardhan, Adam Fidel, Nancy M. Amato, Lawrence Rauchwerger, In Proc. IEEE Int.Conf. on Parallel Architectures and Compilation Techniques (PACT), pp. 27-38, Edmonton, AB, Canada, Aug 2014.
- **"An Algorithmic Approach to Communication Reduction in Parallel Graph Algorithms,"** Harshvardhan, Adam Fidel, Nancy M. Amato, Lawrence Rauchwerger, In Proc. IEEE Int.Conf. on Parallel Architectures and Compilation Techniques (PACT), San Francisco, CA, Oct 2015.
- **"A Hybrid Approach To Processing Big Data Graphs on Memory-Restricted Systems,"** Harshvardhan, Brandon West, Adam Fidel, Nancy M. Amato, Lawrence Rauchwerger, In Proc. Int. Par. and Dist. Proc. Symp. (IPDPS), pp. 799-808, Hyderabad, India, May 2015.
- **"Using Load Balancing to Scalably Parallelize Sampling-Based Motion Planning Algorithms,"** Adam Fidel, Sam Ade Jacobs, Shishir Sharma, Nancy M. Amato, Lawrence Rauchwerger, In Proc. Int. Par. and Dist. Proc. Symp. (IPDPS), Phoenix, Arizona, USA, May 2014.
- **"The STAPL Parallel Graph Library,"** Harshvardhan, Adam Fidel, Nancy M. Amato, Lawrence Rauchwerger, In Wkshp. on Lang. and Comp. for Par. Comp. (LCPC), Tokyo, Japan, Sep 2012.
- **"STAPL-RTS: An Application Driven Runtime System,"** Ioannis Papadopoulos, Nathan Thomas, Adam Fidel, Nancy M. Amato, Lawrence Rauchwerger, In Proc. ACM Int. Conf. Supercomputing (ICS), pp. 425-434 , Newport Beach, CA, USA, Jun 2015.
- **"The STAPL Parallel Container Framework,"** Gabriel Tanase, Antal Buss, Adam Fidel, Harshvardhan, Ioannis Papadopoulos, Olga Pearce, Timmie Smith, Nathan Thomas, Xiabing Xu, Nedhal Mourad, Jeremy Vu, Mauro Bianco, Nancy M. Amato, Lawrence Rauchwerger, In Proc. ACM SIGPLAN Symp. Prin. Prac. Par. Prog. (PPOPP), Feb 2011.
- **"The STAPL pView,"** Antal Buss, Adam Fidel, Harshvardhan, Timmie Smith, Gabriel Tanase, Nathan Thomas, Xiabing Xu, Mauro Bianco, Nancy M. Amato, Lawrence Rauchwerger, In Wkshp. on Lang. and Comp. for Par. Comp. (LCPC), Oct 2010.
- **"STAPL: Standard Template Adaptive Parallel Library,"** Antal Buss, Harshvardhan, Ioannis Papadopoulos, Olga Tkachyshyn, Timmie Smith, Gabriel Tanase, Nathan Thomas, Xiabing Xu, Mauro Bianco, Nancy M. Amato, Lawrence Rauchwerger, In Haifa Experimental Systems Conference, Haifa, Israel, May 2010.