

Corolla: GPU-Accelerated FPGA Routing Based on Subgraph Dynamic Expansion

Minghua Shen and Guojie Luo

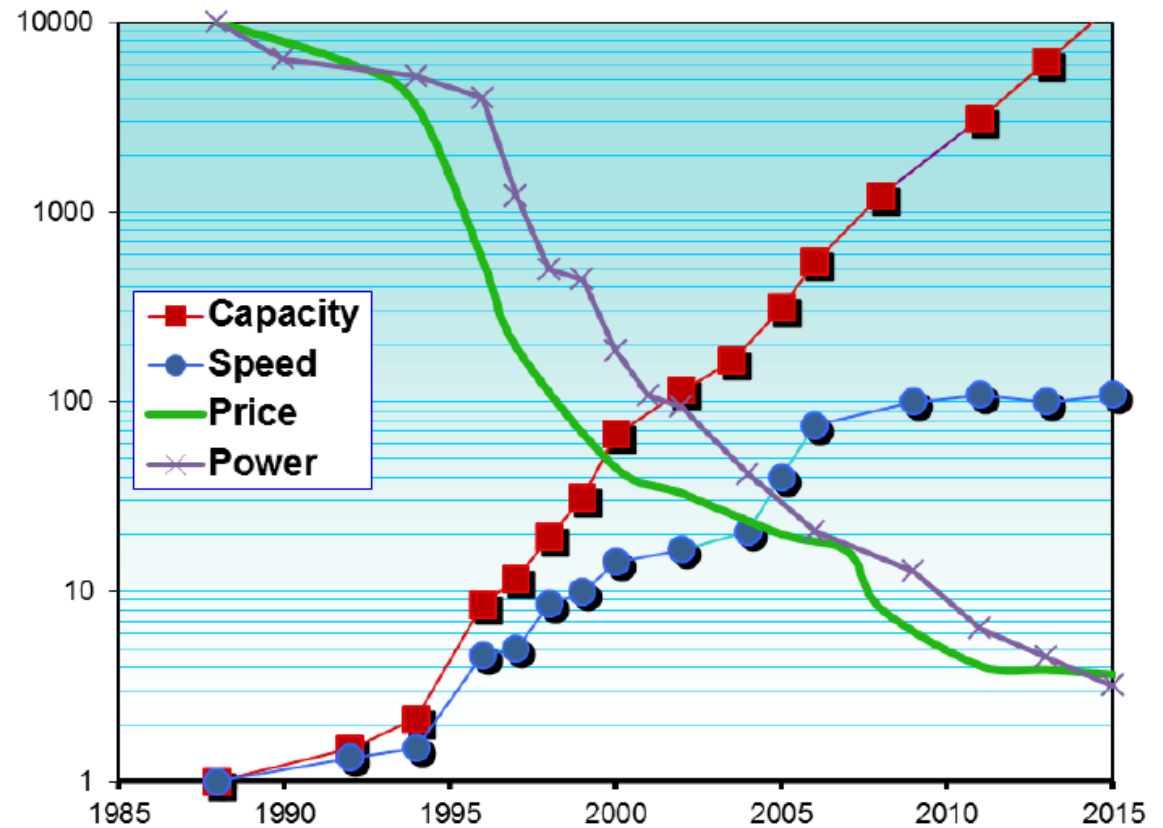
Peking University

Contents

- Motivation
- Background
- Search Space Reduction for Routing
- Parallelism Exploration for GPU-Based Routing
- Evaluation and Conclusion

High Performance FPGA

- 100x Faster
- 10,000x Lower Cost
- 5000x Lower Power



FPGA in datacenters

- Increasing interest in use of FPGAs as application accelerators in datacenters



Microsoft®
bing™



intel®
ALTERA®
now part of Intel

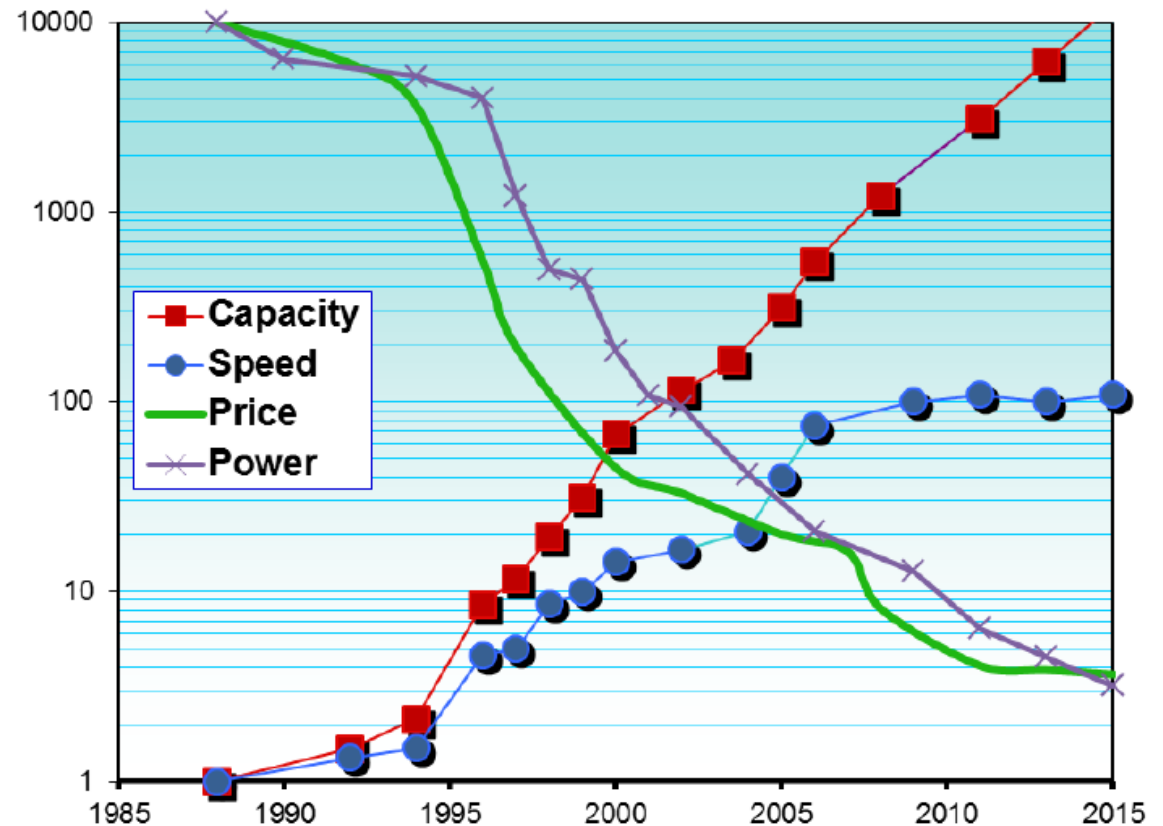


amazon
Baidu 百度

Key advantage: Performance/Watt

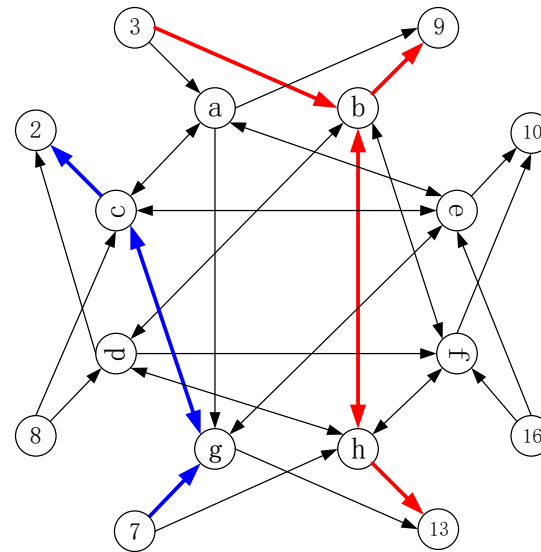
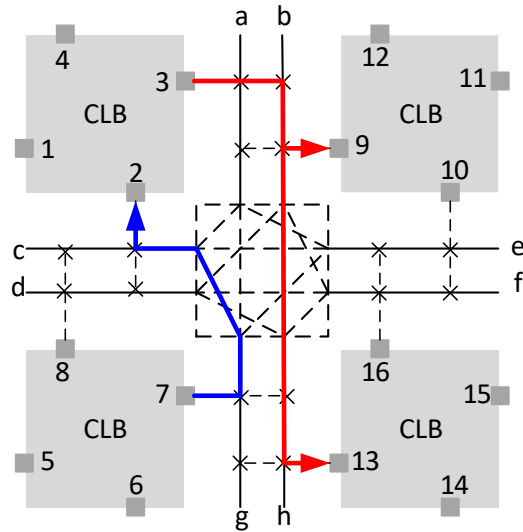
High Performance FPGA

- 100x Faster
- 10,000x Lower Cost
- 5000x Lower Power
- **10,000x More Logic**



FPGA Routing

- Find a physical path for every net in the circuit
 - Disjoint-path problem and NP-complete
 - The most complex and critical step in CAD flow



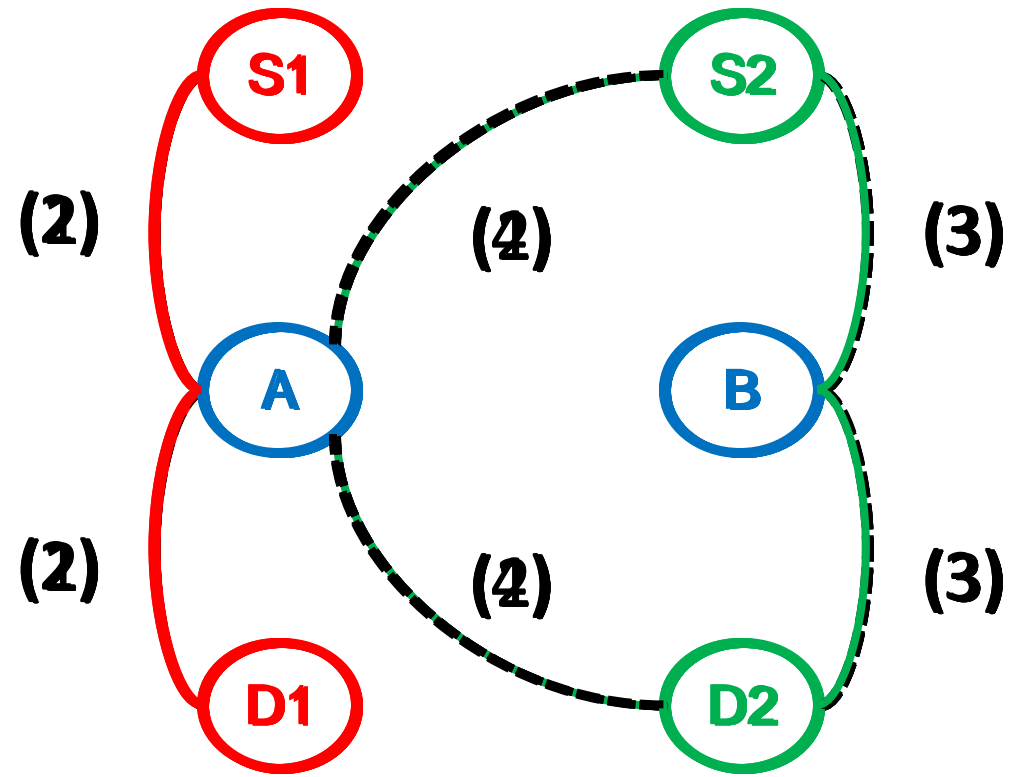
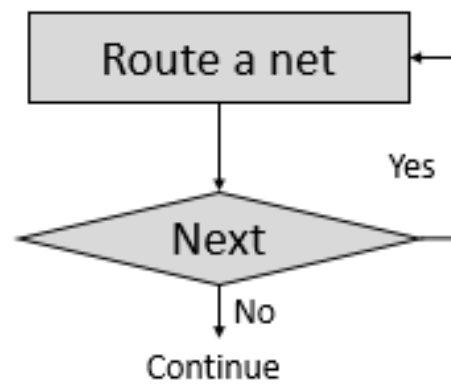
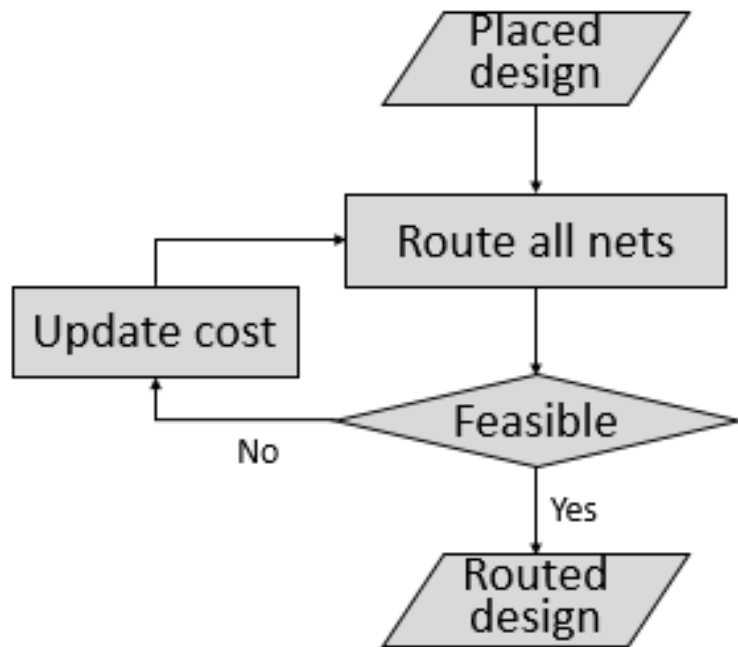
Challenge : **Time-consuming**

Contents

- Motivation
- Background
- Search Space Reduction for Routing
- Parallelism Exploration for GPU-Based Routing
- Evaluation and Conclusion



PathFinder Routing Algorithm

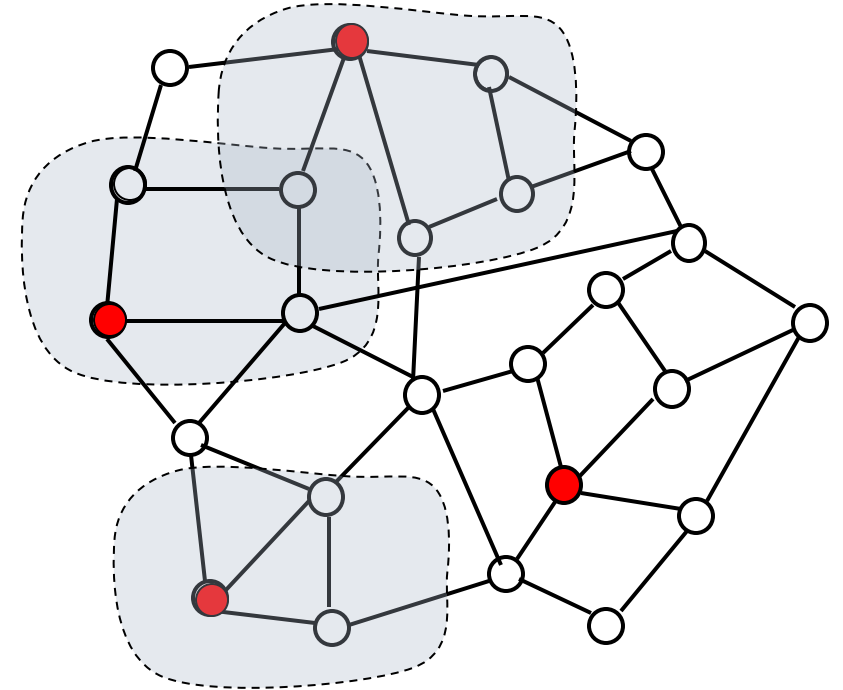
- Basis for both Xilinx and Altera commercial routers



2nd iteration, partitioned by history
 Enhanced by present congestion

Dynamic Parallelism on GPU

- Operator
 - Active node: 
 - node where computation is needed
 - Activity: 
 - Application of operator to active element
- Multiple active nodes can be processed in parallel, subject to neighborhood constraints
- Algorithm = repeated application of operator to graph

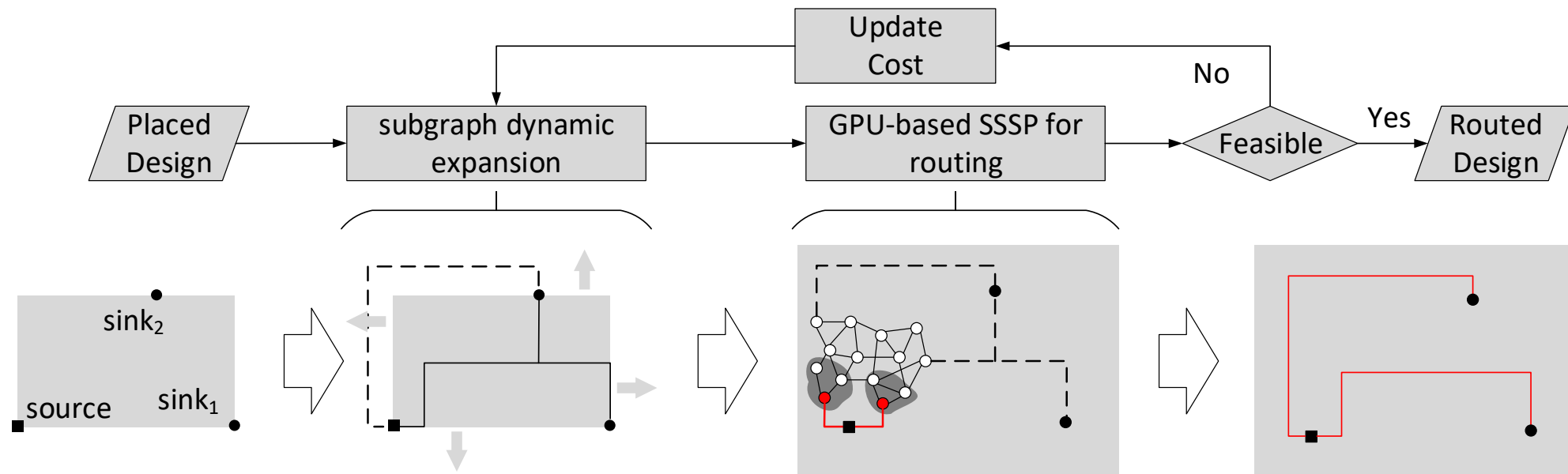


[1] Pingali et al. . “The Tao of Parallelism in Algorithms”, in PLDI 2011.

[2] Y. Muctor and P. Brisk. “Parallel FPGA routing based on the operator formulation”, in DAC 2014.

Overall Design Flow

- The iteration contains two steps:
 - Search space reduction: subgraph dynamic expansion
 - Parallelism exploration: SSSP with dynamic parallelism on GPU



Contents

- Motivation
- Background
- Search Space Reduction for Routing
- Parallelism Exploration for GPU-Based Routing
- Evaluation and Conclusion

Single-Net Routing

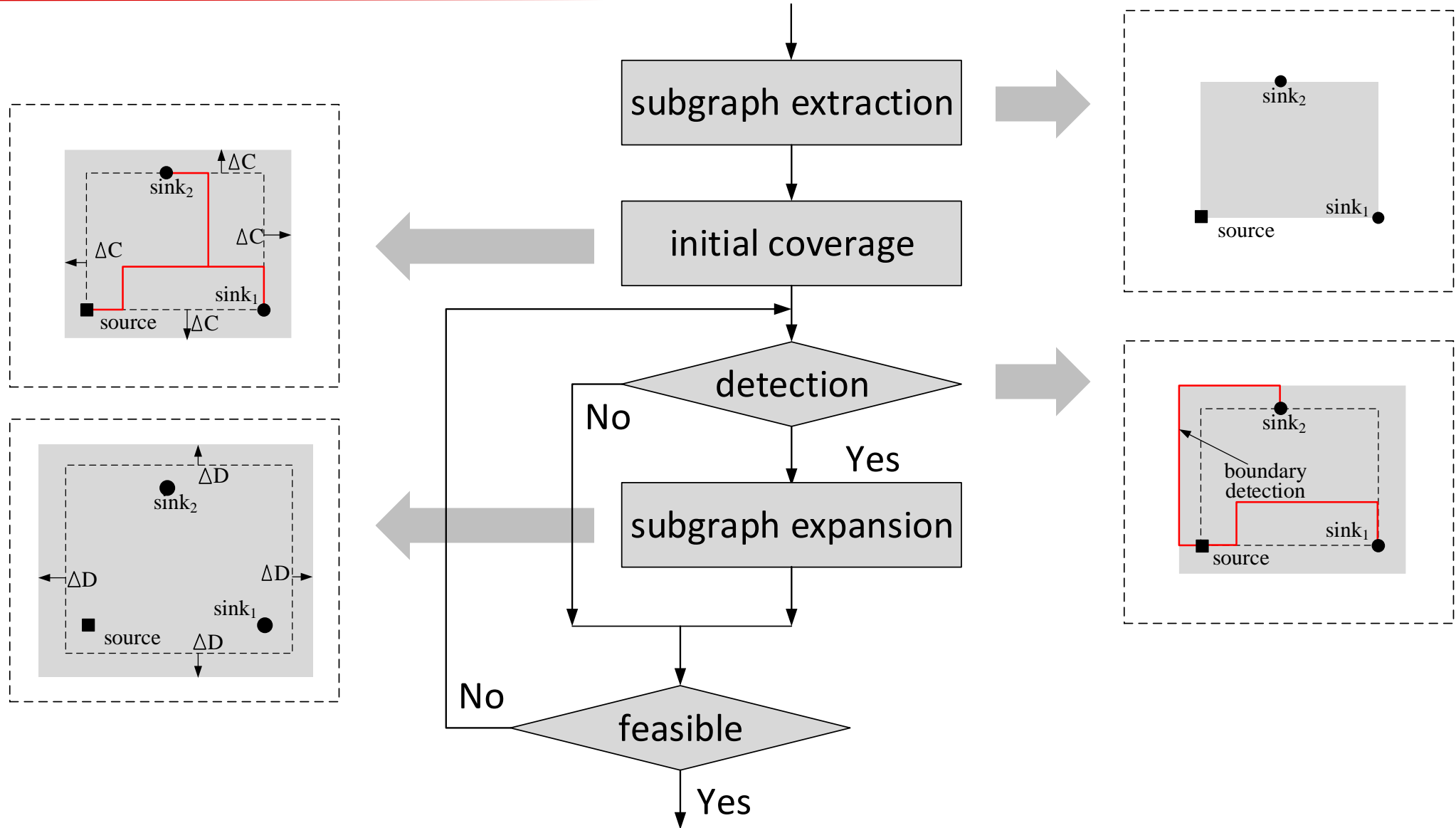
- Kernel: single source shortest path solver
 - Dijkstra -> VPR router
 - Bellman-Ford -> GPU-friendly router
 - Serial Bellman-Ford excels when running on a small graph^[1]
 - Parallel Bellman-Ford provides greatest speedup^[2]

Challenge : routing resource graph is large

[1] A. DeHon et al. “GraphStep: a system architecture for sparse-graph algorithm”, in FCCM 2006.

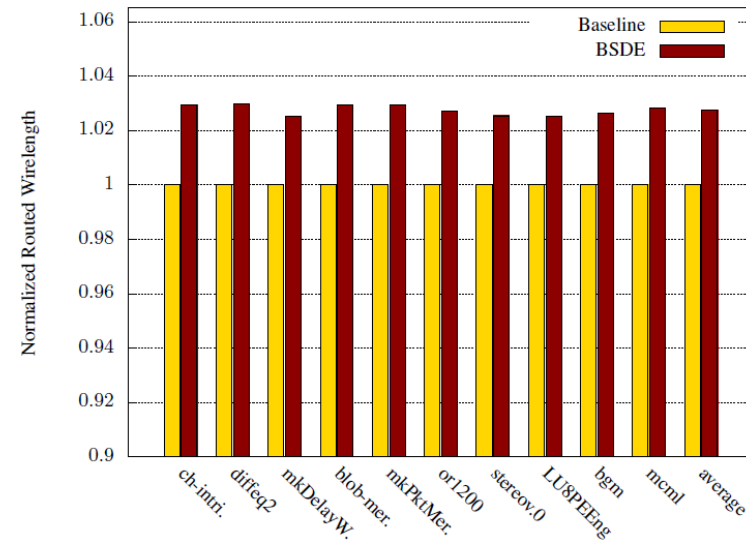
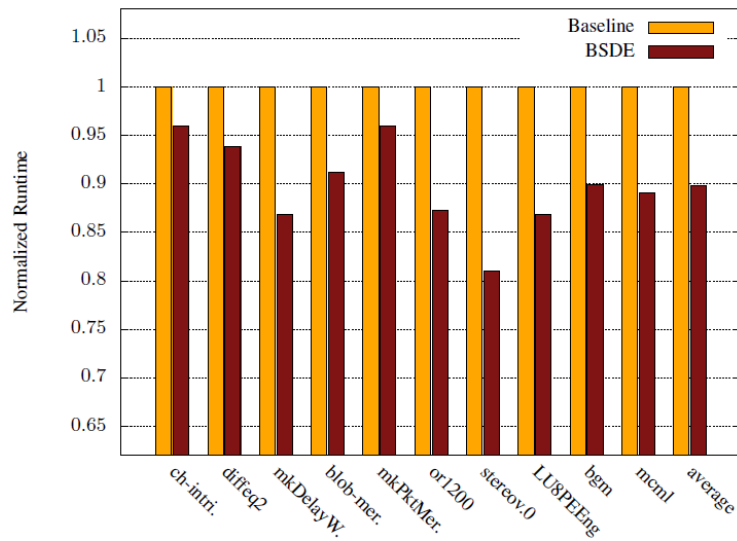
[2] F. Busato et al. “An efficient implementation of the bellman-ford algorithm for Kepler GPU architecture”, in IEEE Trans. on PDS 2016.

Subgraph Dynamic Expansion



Effectiveness

- Runtime-Quality Comparison
 - Baseline: VPR router (Dijkstra + large routing graph)
 - BSDE: GPU-friendly router (Bellman-Ford + small routing subgraph)



Runtime: **10% reduction**

Quality: **2.7% degradation**

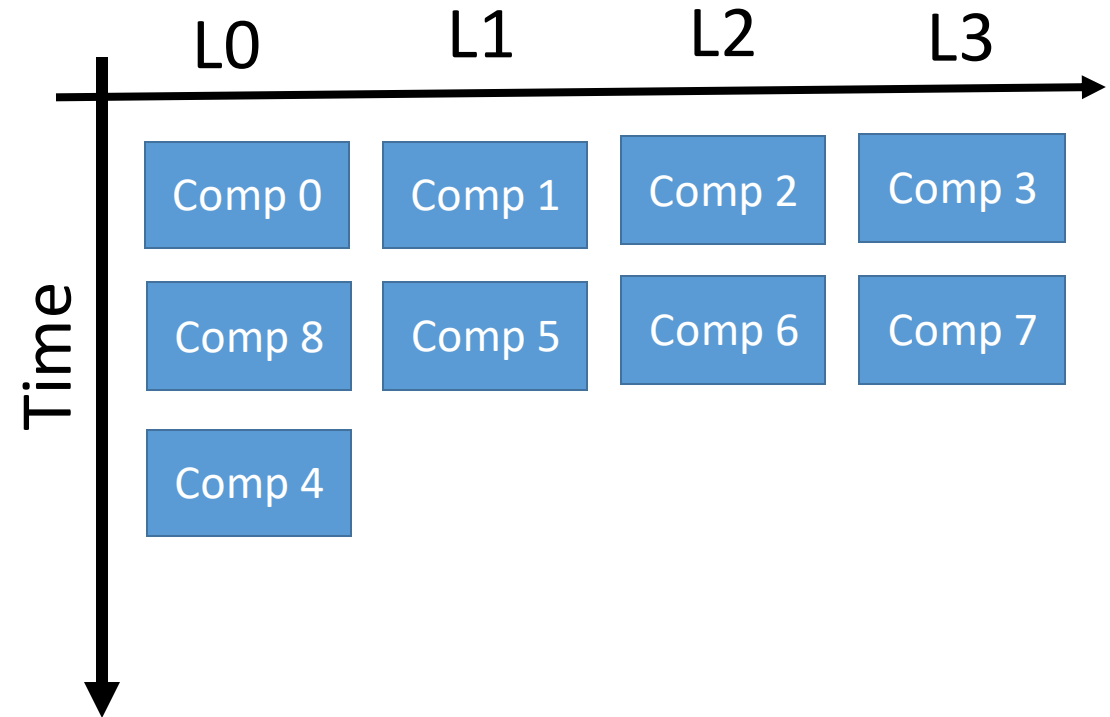
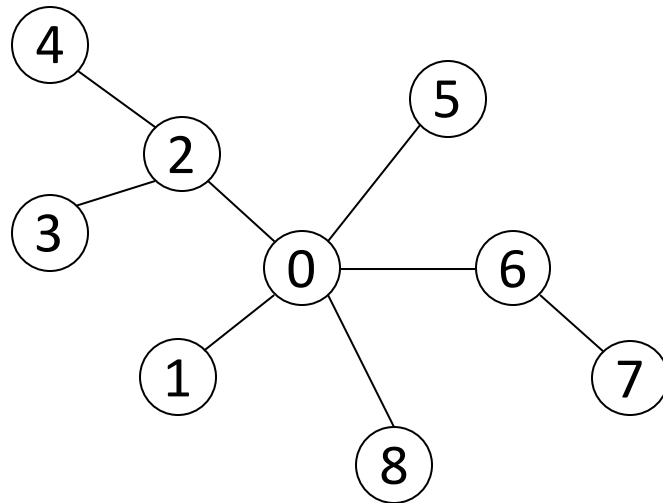
Contents

- Motivation
- Background
- Search Space Reduction for Routing
- Parallelism Exploration for GPU-Based Routing
- Evaluation and Conclusion

Single-Net Parallel Routing

- Static node-level parallelism

- Example



[1] Pingali et al. "A Quantitative Study of Irregular Programs on GPUs", in IISWC 2012.

[2] Pingali et al. "Data-Driven versus Topology-Driven Irregular Computations on GPUs", in IPDPS 2013.

Single-Net Parallel Routing

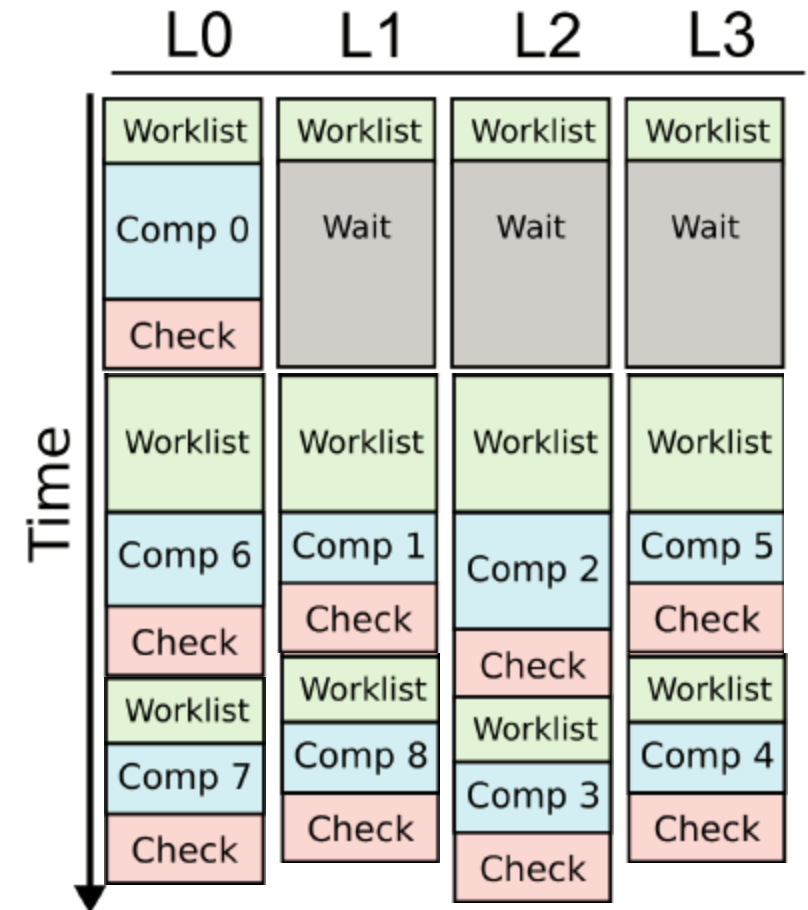
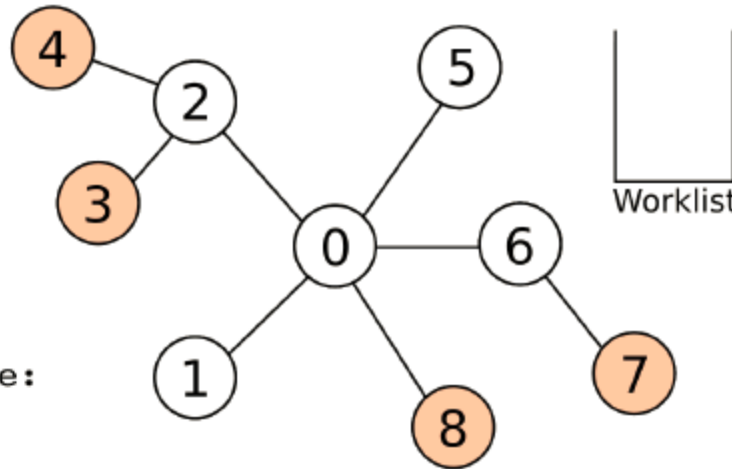
- Dynamic node-level parallelism

- Example

```

def data_driven:
    while idx = wl.pull():
        my_node = nodes[idx]
        compute( my_node )
        for all neighbors of my_node:
            if check( neighbor ):
                wl.push( idx )

def main:
    init_wl<<<N>>>( nodes, wl )
    data_driven<<<M>>>( nodes, wl )
    
```

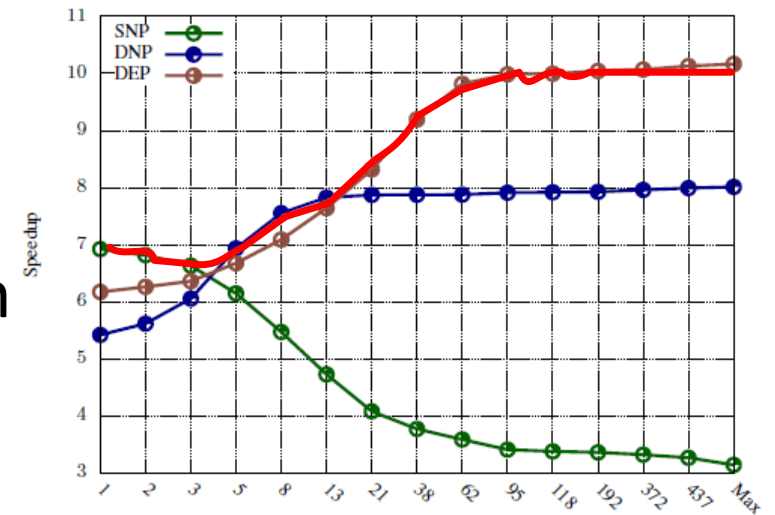


[1] Pingali et al. "A Quantitative Study of Irregular Programs on GPUs", in IISWC 2012.

[2] Pingali et al. "Data-Driven versus Topology-Driven Irregular Computations on GPUs", in IPDPS 2013.

Single-Net Parallel Routing

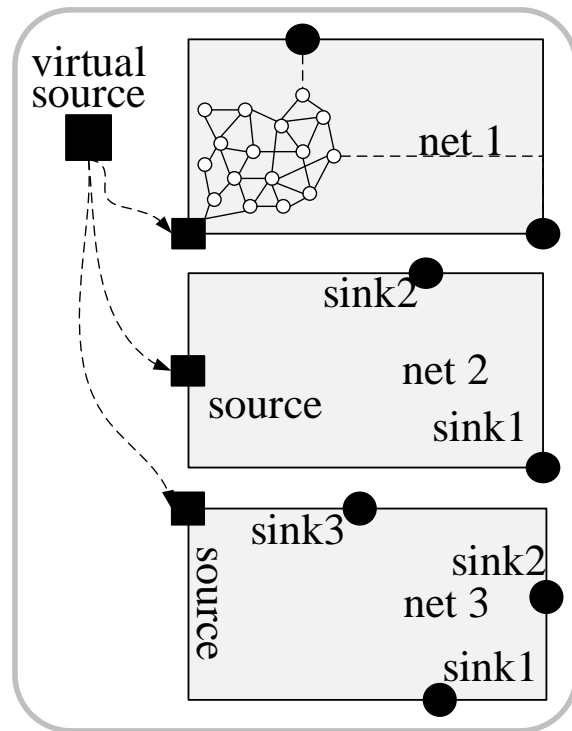
- Dynamic edge-level parallelism
 - Only for the edge
 - Similar to dynamic node-level parallelism
- Hybrid approach
 - Net: different sinks.
 - Static node-level parallelism for low-fanout net
 - Dynamic edge-level parallelism for high-fanout net



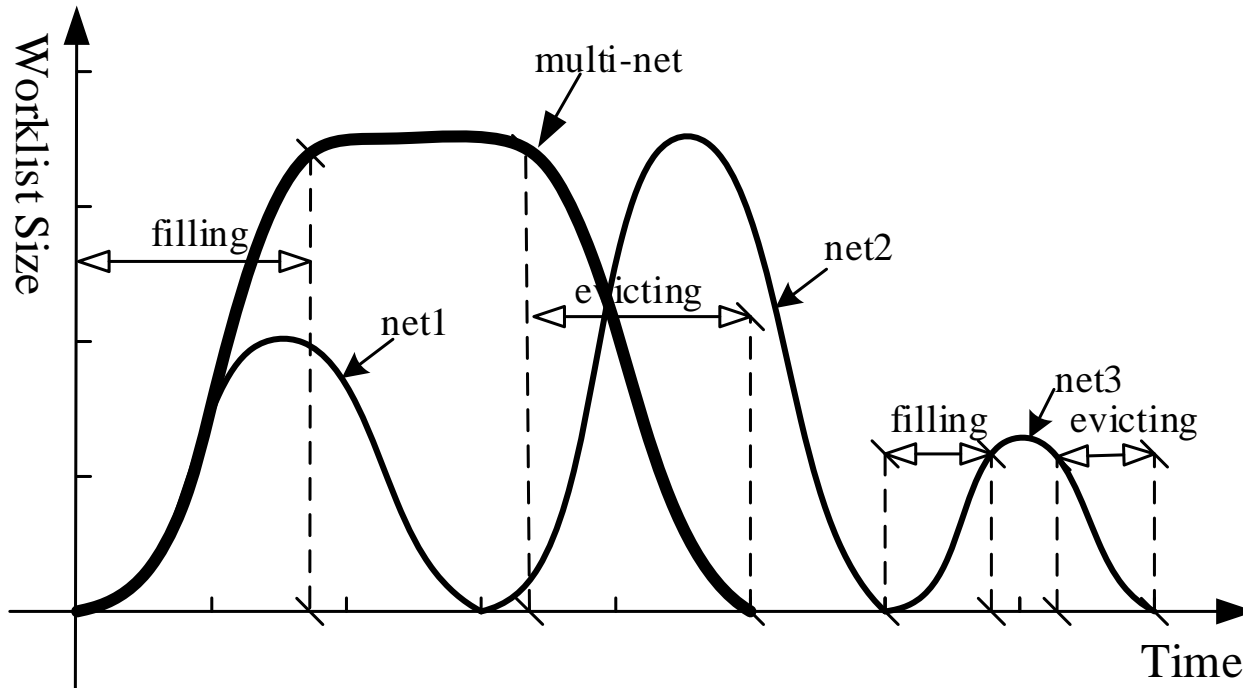
(b) Speedup with respect to the number of sinks

Multi-Net Parallel Routing

- Parallelization for multiple independent nets



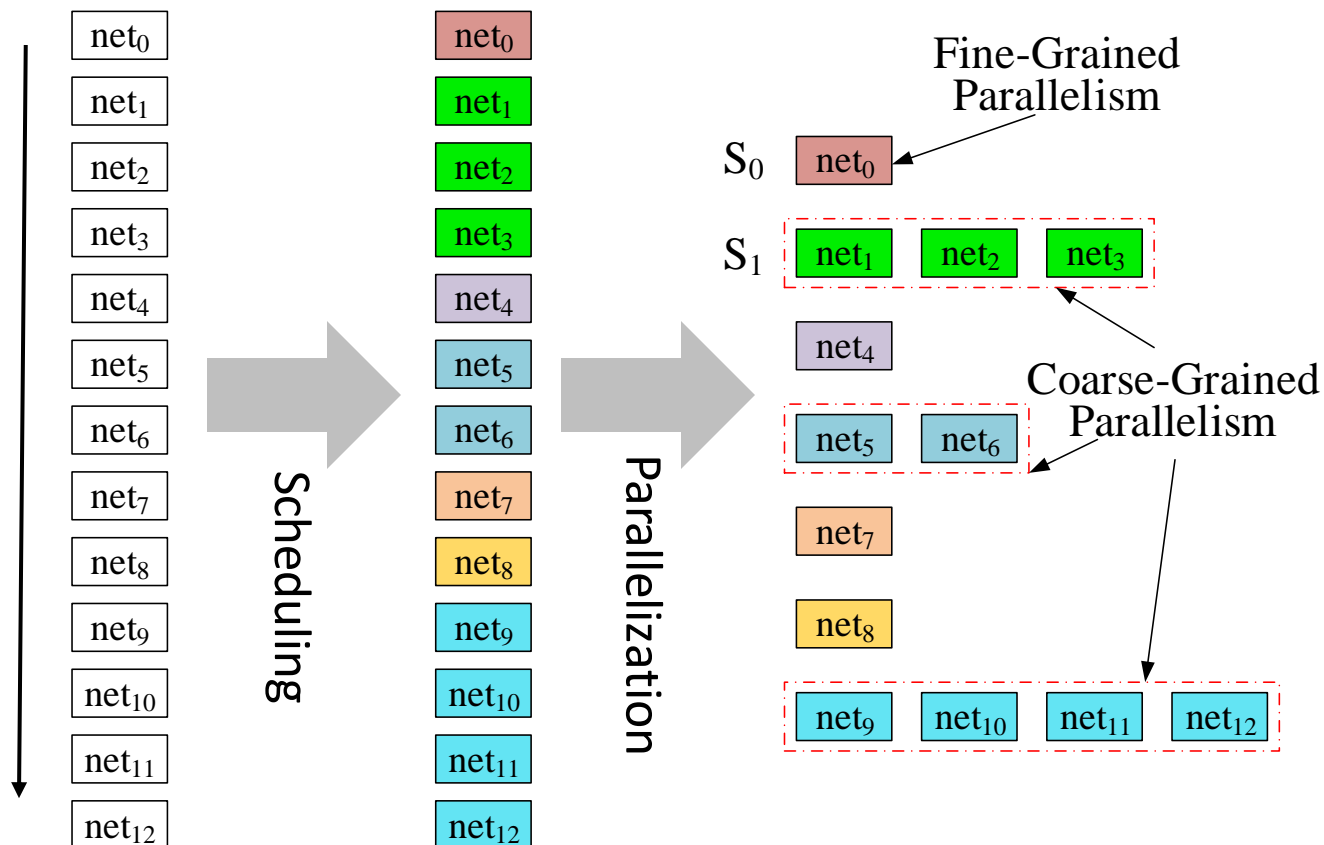
single-net parallelism
 multi-net parallelism



Multi-Net Parallel Routing

- Challenge: dependency
- Solution: extracting Independent nets for parallelization

- Example:



Contents

- Motivation
- Background
- Search Space Reduction for Routing
- Parallelism Exploration for GPU-Based Routing
- Evaluation and Conclusion

Methodology

- Experimental platform
 - Intel Xeon E5-2640 CPU + 32GB memory
 - Tesla K40c GPU + 12GB memory
- Experimental setup
 - Baseline: VPR 7.0 router
 - Channel width: 1.3x minimal channel width
 - Benchmark: VTR benchmark

Speedup

- Single-net parallel routing
 - Static node-level parallelism: 4.15x
 - Dynamic node-level parallelism: 5.82x
 - Dynamic edge-level parallelism: 9.75x
 - Hybrid approach: 10.86x
- Multi-net parallel routing
 - Hybrid approach: 18.72x

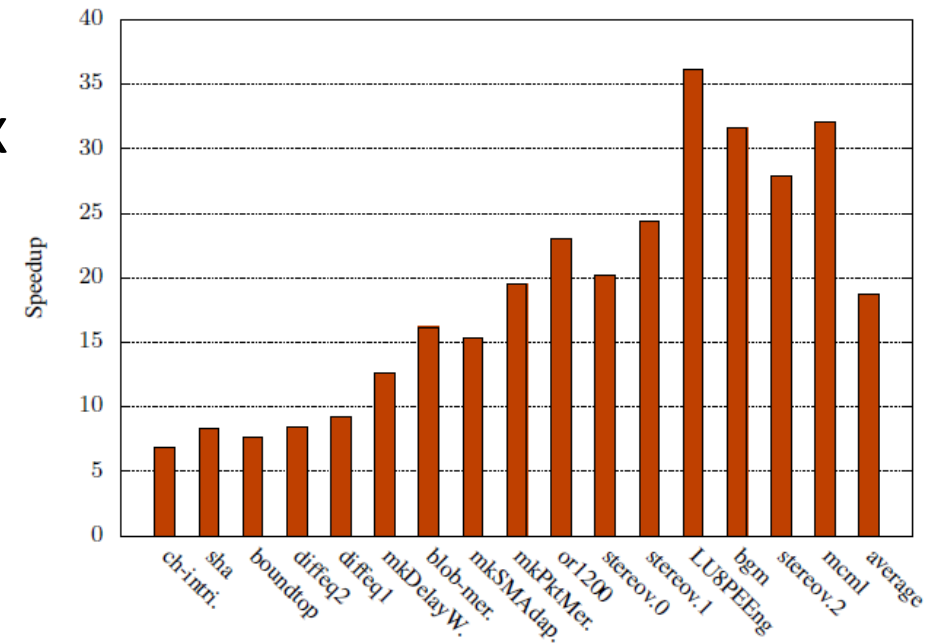
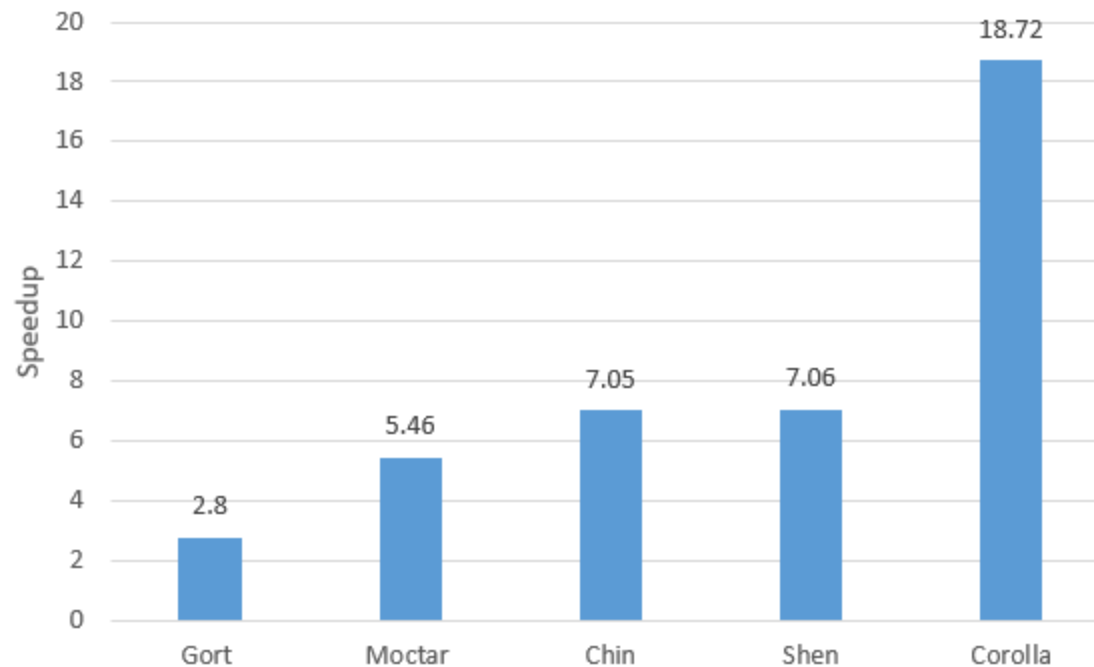


Figure 11. Multi-net acceleration on GPU using the hybrid approach.

Comparison

- Compared to the recent existing work
 - 3.43x fine-grained parallel router
 - 2.67x coarse-grained parallel router



Quality

- Wirelength: 2.7% degradation

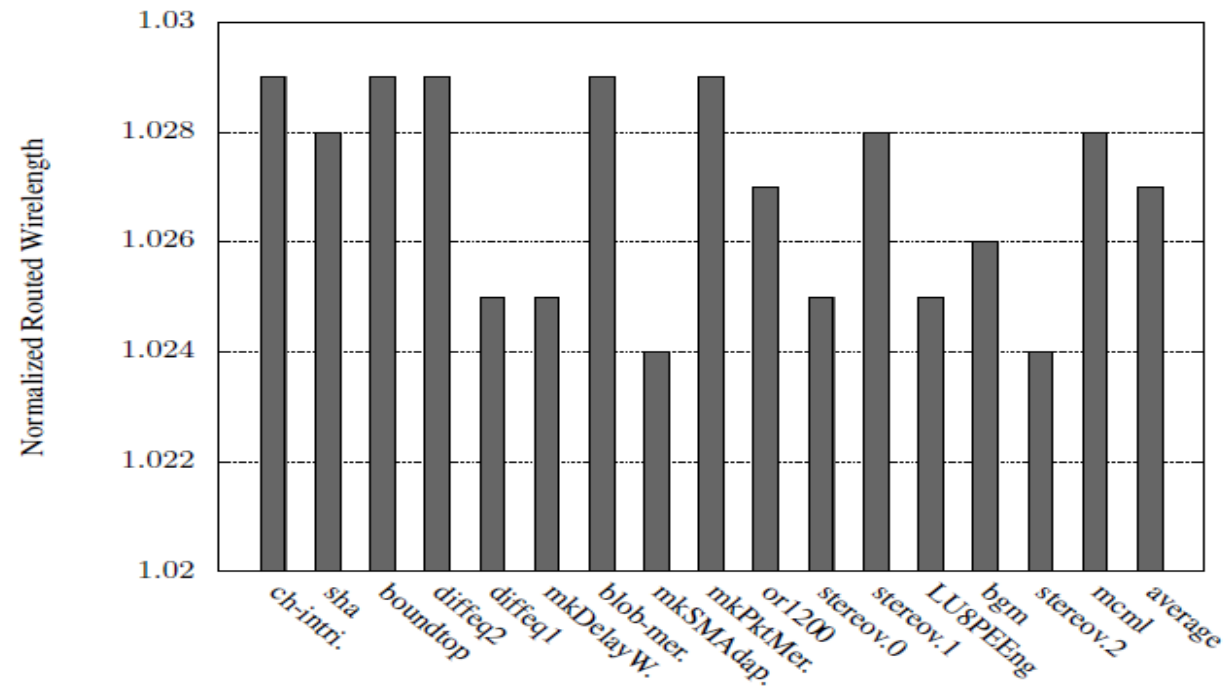


Figure 13. Impacts on the routed wirelength using the multi-net parallelization in Corolla.

Scalability

- Construct synthetic designs
 - Stretch the locations of sink and source of each net
 - FPGA size: 100x100 – 1000x1000
- Maintain a similar speedup
 - Reduce the problem size
 - Woklist that behaves similarly to a queue

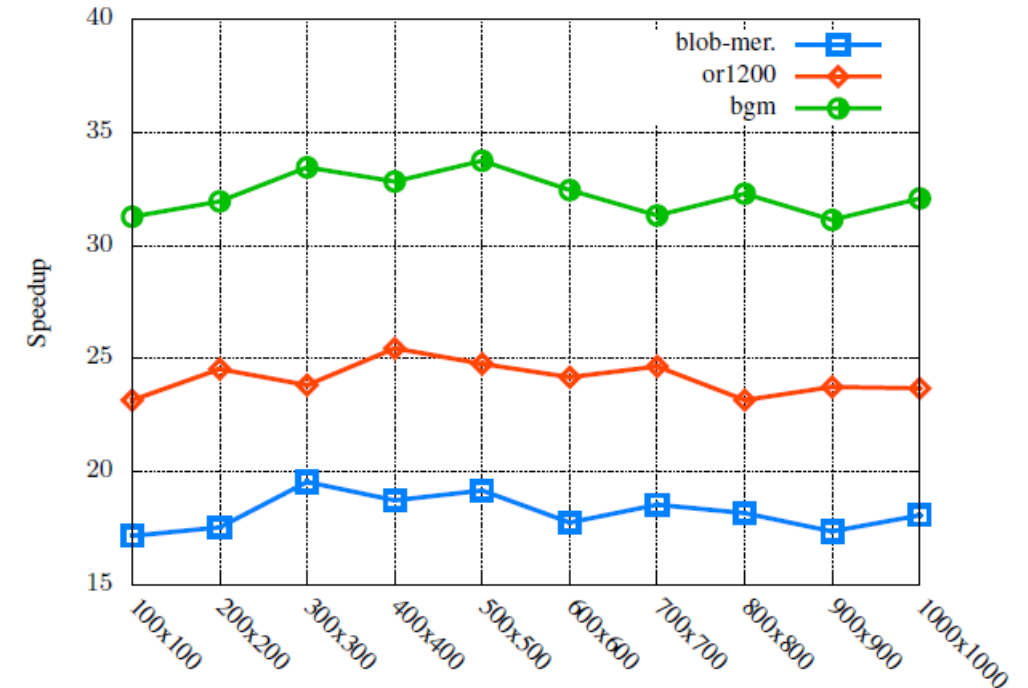


Figure 14. Scalability Analysis with different FPGA array sizes.

Take-Away Points

- Search space reduction is very effective to mitigate the time complexity of GPU-friendly routing algorithm.
- More speedup can be achieved with both node-level and net-level parallelization.

Special thanks to LonestarGPU team!

Thanks!

Question?