High Performance Computing Systems (CMSC714)

Lecture 17: Topology Aware Mapping

Abhinav Bhatele, Department of Computer Science

UNIVERSITY OF MARYLAND

Summary of last lecture

- Most HPC systems use a job/batch scheduler
- Scheduler decides what jobs to run next and what resources to allocate
  - Backfilling to use idle nodes and improve utilization
- Different quality of service metrics to evaluate schedulers

Task Mapping

- Also referred to as task placement or node mapping
- Given an allocation, decide which MPI processes are placed on which physical nodes/ cores
  - In case of task-based models, map finer-grained tasks to cores
- Goal:
  - Minimize communication volume on the network
  - Optimize "unavoidable" communication on the network

Graph embedding problem

- Inputs: Application communication graph, network topology graph (of one's job allocation)
- Output: Process-to-node/core mapping
- Most mapping algorithms do not consider that communication patterns might evolve over time

Metrics to evaluate mapping

- Hop-count

$$\sum_{(i,j} H(i,j)$$

- Hop-bytes

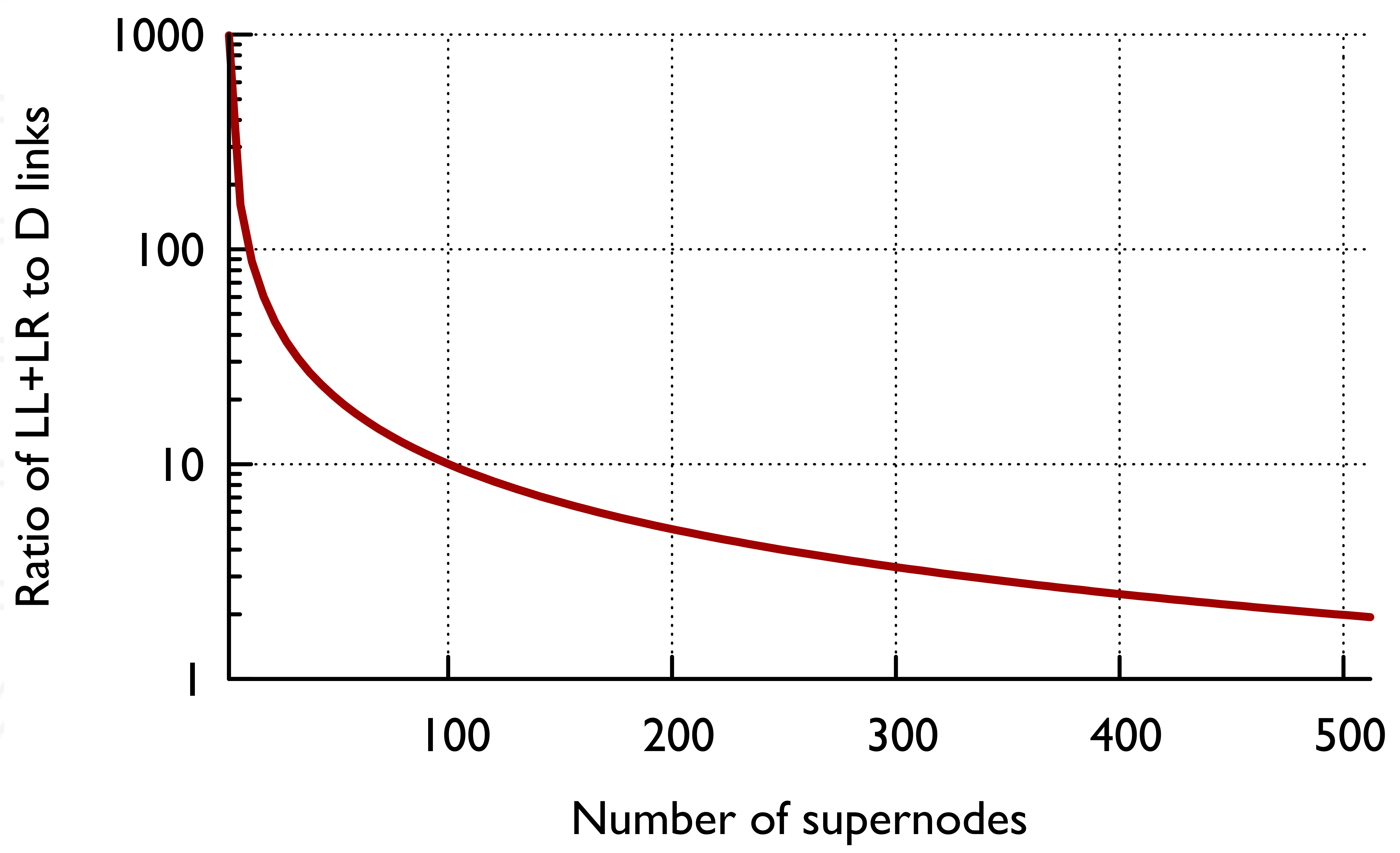
$$\sum_{(i,j} C(i,j) \times H(i,j)$$

Different techniques

- Heuristics-based
  - Recursive bi-partitioning
  - Random pairwise swaps
- Physical optimization problems
  - Simulated annealing
  - Genetic algorithms

Global link bottleneck in dragonfly systems

- Few global links when building a smaller than full-sized system



Questions

Optimizing task layout on the Blue Gene/L supercomputer

- Does the favorable performance of SA, or the proposed divide-and-conquer method, generalize to other topologies (such as the more modern dragonfly and fat-tree)?
- How well will the proposed mapping method perform in other applications, such as FMM accelerated N-body (dense interaction) problems?
- Does the topology-aware task mapping method based on SA also help improving performance in manycore processors in a single node?
- The paper is from 2005. Is the proposed method still used today? Are there more advanced techniques used nowadays?
- The authors note that their implementation of the proposed layout optimization algorithm is slow, taking several hours to run in some cases. They also claim that this could be easily parallelized for production purposes. Is it obvious that the proposed algorithm can be easily parallelized? Has somebody done this?
- Since the method takes the communication matrix as input, does this mean that we have to run the HPC application in full before the layout can be optimized? Could you extrapolate from the communication matrix of a small-scale run so that the full-scale version never has to run unoptimized?
- I'm confused about where the inverse temperature parameter comes from in the Metropolis algorithm. Do we call this "temperature" because of the analogy to free energy? This isn't actually temperature, right?

Questions

Avoiding hot-spots on two-level direct networks

- Indirect routing increases overall network traffic, so the comparable performance with RNM is a bit counter-intuitive. How can this be explained?
- Indirect routing performed slightly worse than RNM in the 64SN case but slightly better in the 300SN case. How can this be explained? (One might expect the improvement of RNM to be larger if there are more SNs)
- Time profile results are only shown for DEF and RNM. How would it look like in DFI and RDI?
- Why did the authors not try "Random Nodes with Indirect Mapping", which might be also promising?
- If the mapping schemes were implemented in a real system, where are they implemented? In the MPI runtime? Or, does the application programmer has to implement it in the application code?
- The authors experiment with indirect routing for the default and the random drawers mappings. Is there a reason to not experiment with indirect routing in the other mapping cases? Is there a reason that this type of combination wouldn't work or wouldn't make sense?
- Are there other communication patterns unexplored in this paper that are commonly used? Are the three that are explored the most common?

Questions?

UNIVERSITY OF MARYLAND

Abhinav Bhatele

5218 Brendan Iribe Center (IRB) / College Park, MD 20742

phone: 301.405.4507 / e-mail: bhatele@cs.umd.edu