CS 598: Communication Cost Analysis of Algorithms

Lecture 29: Network interconnect topologies

Edgar Solomonik

University of Illinois at Urbana-Champaign

December 7, 2016

Network interconnects

For the duration of the course, we have focused on communication on 'fully-connected' networks, implicitly assuming that

- (1) any pair of processors can exchange messages at the same speed
- (2) messages between distinct pairs do not affect one another
  - this effect is known as *network contention*
- we did this consciously, aiming general analysis of algorithms
  - there are no widely-used generic network models for algorithms
  - the connectivity structure of different networks can differ drastically
  - on real systems, algorithms execute on a subset of a network (the only type of existing large-scale architecture on which this subset is structured are the BlueGene and K-computer torus networks)
- we will first discuss torus networks and topology-aware collectives
- then we will study a few other topologies based on general metrics

Characterization of interconnect topologies

A network topology can be characterized by

- *direct* vs *indirect*: are nodes connected directly or via switches?
- *diameter*: maximum number of hops between nodes (switches)
- *degree*: number of links on each node/switch
- *link bandwidth*: bandwidth per link (often non-uniform in indirect topologies)
- *bisection bandwidth*: minimum balanced edge cut of graph with edge weights equal to link bandwidth
- *injection bandwidth*: (specific to direct topologies) how many links can a node saturate?

Mesh and torus networks

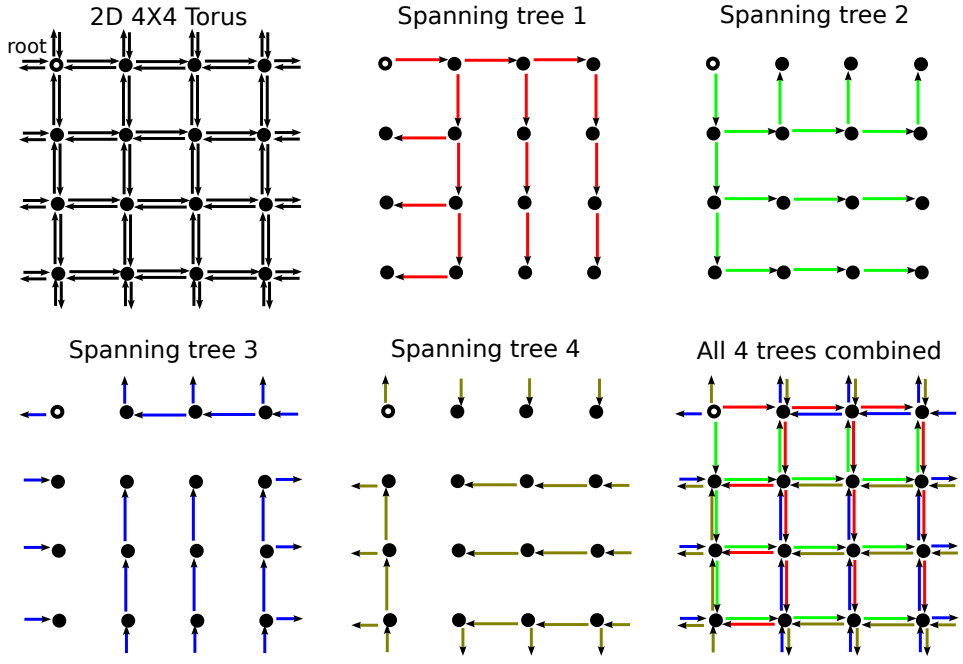
A simple direct n-node network topology is a k-dimensional grid

- a torus is distinguished from a mesh by wrap-around links
- the simplest torus is a ring
- tori are generally advantageous as all nodes are 'created equal'
- Q: how could a ring network be constructed so each link is the same physical length?
- when $P = 2^k$ a mesh topology is a hypercube
- 3D torus topologies have been popular in HPC because many applications map nicely to them
- larger k implies higher bisection bandwidth, which scales with $n^{(k-1)/k}$, so 5D torus networks have been more popular recently

Collective communication on torus networks

As a case-study consider broadcast on a torus with full injection bandwidth

- so injection bandwidth is $2k$ times link bandwidth
- BlueGene tori have full injection bandwidth, but not Cray XT/XE series or K computer
- optimal broadcast protocols are given by edge-disjoint spanning trees



- $n/2k$ data is sent along each spanning tree

Topology-aware algorithms on torus networks

On torus networks, it is worthwhile to design algorithms that map to the topology

- if arbitrary subsets of processors communicate, there will be network contention
- the benefit of topology-aware algorithms (like the above 'rectangular' broadcast) is to fully avoid network contention
- a simple example is matrix multiplication
  - on a 2D torus, SUMMA and Cannon avoid network contention
  - on a 3D torus, a 3D matrix multiplication algorithm is highly beneficial
  - on a $2k$D torus, we can used a 2D algorithm like SUMMA with each processor grid rows and columns arranged in a kD torus
- another important example is iterative methods
  - given a 3D domain/mesh, contention between ghost zone exchange passes can be avoided by mapping to 3D torus topology or 3D 'unfolding' of higher-order torus

All-to-all on torus networks

One-to-all and all-to-one collectives like broadcast, reduce, scatter, gather are done efficiently by rectangular algorithms like broadcast

- all-to-all is noticeably more difficult
- let each processor sends a total of s data, s/P to each processor
- $\Omega(sP)$ data needs to cross any balanced cut, while bisection bandwidth is $P^{(k-1)/k}$ with respect to link bandwidth
- so $\Omega(sP^{1/k})$ data must cross some link
- in practice, randomized algorithms are used for all-to-all
- a more concrete approach is to perform all-to-all along rings in each dimension (in sequence or a distinct subset of the all-to-all data along disjoint dimensional orderings)
- Q: how can we perform an all-to-all along a ring communicator?
- A: pass data going m-hops away for $m = P^{1/k} - 1$ to $m = 1$, total communication cost $\sum_{i=1}^{P^{1/k}-1} 2^i s = O(sP^{1/k})$

General routing strategies

Specialized collective routines can be designed to be very efficient on tori, but a network topology must permit any communication pattern

- even a broadcast for a random subset of nodes poses a difficulty
- bandwidth-efficiency is achieved for messages via *wormhole routing*
  - each message is subsdivided into packets
- a message can be stalled due to *head-of-line blocking*
  - multiple packets from two input links can follow the same output link
  - one line of packets must wait in a *buffer*
- deadlock can occur if route 1 goes through link a then through link b while route 2 goes through link b then through link a
- Q: can deadlock occur if no pair of routes behave like this?
- A: yes, consider route 1: $\{a, b\}$, route 2: $\{b, c\}$, route 3: $\{c, a\}$
- to prevent deadlock, routing protocol graph should not contain cycles
  - graph has edge between link a and link b if some route follows this consecutive pair of links

Virtual channels

Virtual channels provide flexibility to routing protocols

- each physical channel (e.g. fiber in a torus) is assigned multiple virtual channels
- protocols multiplex the channels, alternating between packets from different input virtual channels
- each packet is routed based on its channel label and its destination
- no deadlock can occur within a channel so long as the routing graph contains no cycles
- no deadlock can occur in the entire protocol if the *channel dependency graph* has no cycles
  - a channel dependency graph has an edge if there is a transition from one channel to another for any node and destination
- two channels suffice for deadlock-free routing on tori
- there can be trade-offs between the number of virtual channels defined and flexibility of the routing strategy

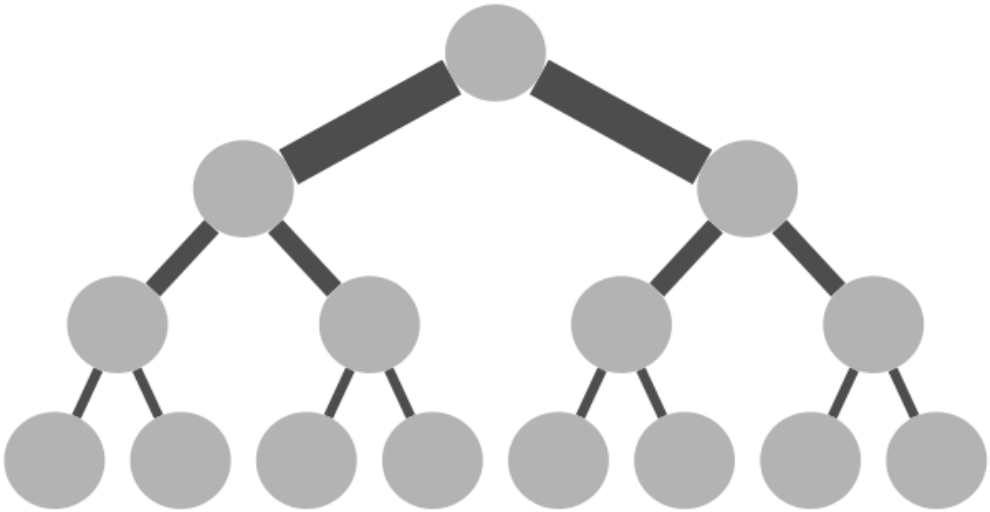
Short pause

Tree network topologies

Indirect topologies leverage routers that are not associated with any node

- typically each node is connected to a single router
- a network connects these routers, possibly using a hierarchy of routers
- Ex: a butterfly network has P nodes and $P(\log_2 P - 1)$ routers
- tree networks are a natural hierarchical construction
- Q: if the network is a binary tree with uniform link bandwidth, what is its bisection bandwidth?
- A: its bisection bandwidth is bound by the root and is equal to the link bandwidth
- fat-trees (Leiserson 1985) solve this problem by increasing link bandwidth exponentially from leaves to root

Fat-tree network topology



source: https://commons.wikimedia.org/wiki/File:Fat_tree_network.svg

Fat-tree bisection bandwidth

Fat-trees can be specified differently depending on the desired properties

- to achieve maximal bisection bandwidth, we can increase link bandwidth by factor of 2 from leaves to root
- Q: what factor of increase do we need if we have P leaves and want bisection bandwidth to be P^k times link bandwidth for $k < 1$
- A: need factor of f so that $f^{\log_2(P)} = P^k$, so $f = 2^k$
- to be able to construct a fat-tree efficiently, it makes sense to choose $k = 2/3$ and $f = 4^{1/3}$
- this choice enables the fat-tree to be embedded into 3D space (bisection bandwidth is like 3D torus)
- the construction is *universal*: no network can be constructed with the same number of components that is faster by more than a polylogarithmic factor
- key idea: use *decomposition tree* to subdivide physical space and simulate any communication pattern via fat-tree

Universality of fat-trees

We sketch the analysis of Leisersion 1985 "Fat-Trees: universal networks for hardware-efficient supercomputing"

- consider routing a message set $M \in [1, P] \times [1, P]$ (all possible interchanges of datums of unit size between processors)
- for each link l, define
  - load(M, l) to be the number of messages passing through l
  - cap(l) to be the number of messages that can pass through l simultaneously (effective bandwidth)
  - the load factor for l is

$$\lambda(M, l) = \frac{\text{load}(M, l)}{\text{cap}(l)}$$

- load factor for the whole tree $\lambda(M)$ is the max $\lambda(M, l)$ for any link l
- given any definitions of cap(l) and it is possible to decompose any M into $M = \bigcup_{i=1}^{d} M_i$ such that $\lambda(M_i) = 1$ and $d = O(\lambda(M) \log(P))$

Hardware requirements of fat-trees

Leiserson consider the VLSI volume (rather than area) necessary to implement a fat tree

- in his construction, the capacity/link-bandwidth increases exponentially from leaves to root by $f = 4^{1/3}$ to achieve bisection bandwidth of $P^{2/3}$ times link bandwidth
- the VLSI volume necessary to construct the network architecture can be embedded into a volume that grows as $O(P \log(P)^{3/2})$
- for any other network, it is assumed that $O(a)$ information can pass through an area of size a
- the fat-tree universality theorem uses this to show that if any network in volume v delivers any message set M in time t, a fat-tree in v can deliver M in time $O(t \log(P)^3)$
- the proof constructs a decomposition tree of the volume and simulates the given network using a corresponding fat tree

Low-diameter network topologies

Fat-trees are optimal within polylogarithmic factors, but hardware is designed with consideration even for small constant factors

- the current trend is towards topologies that have a low diameter (maximum path length)
- consider a definition of the diameter just in terms of the number of routers, and assume there are $O(P)$ base routers (connected to nodes)
- the latest Cray architectures leverage the Dragonfly topology [Kim, Dally, Scott, Abts, ISCA 2008]
  - define densly connected groups (cliques) of routers
  - connect a pair of routers between each group
  - resulting topology is diameter 3
- one of the latest innovations is the Slim-Fly topology [Besta, Hoefler 2014], which is diameter 2 and satisfies some optimality properties

Motivation for slim-fly

- Q: what is the simplest diameter 2 topology you can think of?
- define a 2D grid of nodes $\Pi \in [1, \sqrt{P}] \times [1, \sqrt{P}]$ connect each $(i, j) \in \Pi$ to $(i, k), (k, j) \in \Pi$ for each k
- require $2\sqrt{P}$ incoming and outgoing links per node
- Q: how many 2-hop routes are there between each pair of nodes?
- A: there are 2, which suggests that we may be able to construct a network with fewer links
- it is possible to use fewer links by relaxing the assumption that each link is bidirectional, but this is undesirable in hardware terms

A lower bound on degree

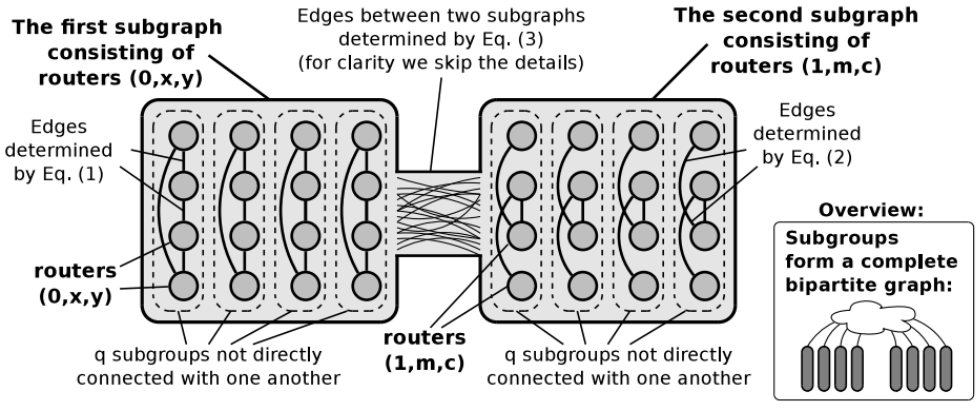
In general, we want to find an undirected graph which has minimal diameter r for a given degree d

- in graph theory, Hoffman and Singleton (1960) defined Moore graphs as having a maximal number of vertices P such that

$$P = 1 + d \sum_{i=1}^{r-1} (d-1)^i$$

- for $r = 2$ we have $P \leq 1 + d(d-1)$, so $d \geq \sqrt{P}$
- thus, for a given P, we have a lower bound
- our initial construction was within a factor of two
- some Moore graphs (ones that attain the bound) exist for $r = 2$, but the existence of general families is an open question
- McKay, Miller, and Siran 1998 provide a general construction that comes close to the Moore bound

Slim-fly construction



- source: Besta, Hoefler 2014. Slim Fly: A Cost Effective Low-Diameter Network
- a network of size $2q^2$ is constructed where q is (almost any) prime
- there are two $q \times q$ grids A, B, each node is connected to some nodes in its column and some nodes in the other grid
- given a node $(x, y) \in A$ in the first grid and $(m, c) \in B$ in the second grid, they are connected iff

$$mx + c - y \equiv 0 \mod q$$

- these links suffice to connect any pair of nodes in two columns of the same grid!

Slim-fly routing

Given (x, y) and (x', y'), there must exist (m, c) such that

$$mx + c - y \equiv mx' + c - y' \equiv 0 \mod q$$

- to route we need to determine m, c given x, x', y, y'
- we can do some modular arithmetic to determine m, c

$$mx + c - y \equiv_q mx' + c - y'$$
$$m(x - x') \equiv_q y - y'$$
$$m \equiv_q (x - x')^{-1}(y - y')$$

  where we need to find the modular multiplicative inverse (this is one of the reasons q needs to be prime)
- we also need to connect (x, y) to (m, c) by finding (x', y) and (m', c) that are connected, so $m'x' \equiv_q c - y$, which defines how nodes should be connected in columns

Slim-fly degree

Each node is connected to q nodes in the other grid

- this holds since given m, x, c there is a unique y such that

$$mx + c - y \equiv 0 \mod q$$

- within each column, the number of connections is given by half the size of the generator set for the prime number field induced by multiplication modulo q
- the smaller the generator set the fewer connections are necessary
- in any case, it suffices to connect each node to at most $q/2$ nodes within its grid column
- thus we have $d = 3q/2$ connections given $2q^2$ nodes, which is close to the Moore bound (roughly) $d \geq \sqrt{2}q$
- generalizing this construction to arbitrary diameter is an open question