OBST: A Self-Adjusting Peer-to-Peer Overlay Based on Multiple BSTs

Chen Avin (BGU)

Michael Borokhovich (BGU)

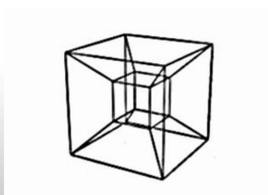
Stefan Schmid (T-Labs)

From “Optimal” Networks to Self-Adjusting Networks

- Networks become more and more dynamic (e.g., flexible SDN control)
- Vision: go beyond classic “optimal” static networks
- Example (of this paper): Peer-to-peer

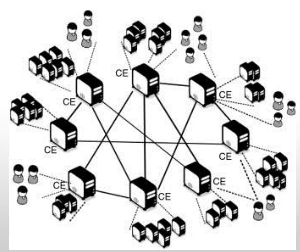
Chord, Pastry, SHELL

- Hypercubic
- Log diameter
- Log degree
- Log routing



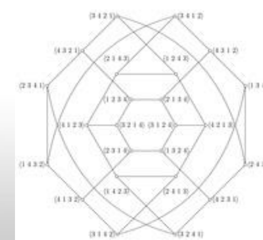
Koorde, ...

- Constant degree
- Log routing



Pancake

- Log/loglog degree and log/loglog routing



From “Optimal” Networks to Self-Adjusting Networks

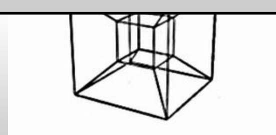
- Networks become more and more dynamic (e.g., flexible SDN control)
- Visi
- Exa

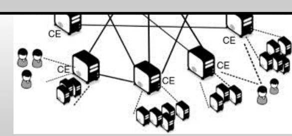
What if networks could self-adjust depending on communication pattern?

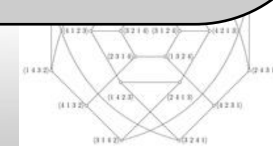
Chor

- Hyp
- Log
- Log
- Log

nd

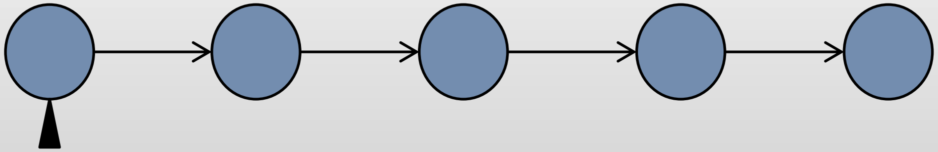




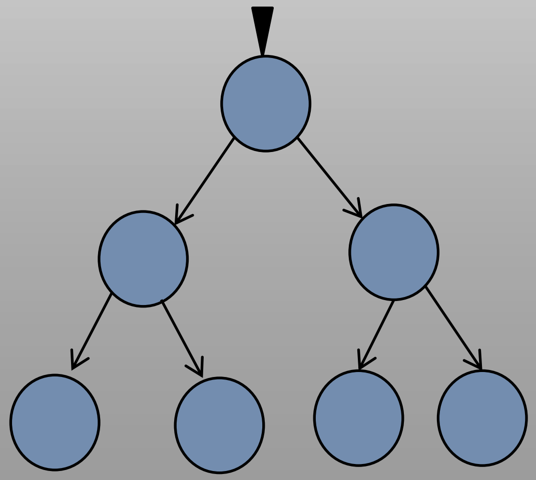


An Old Concept: Move-to-front, Splay Trees, …

- Classic data structures: lists, trees
- Linked list: move frequently accessed elements to front!



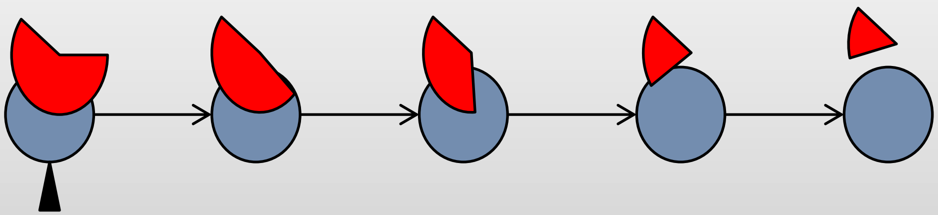
- Trees: move frequently accessed elements closer to root



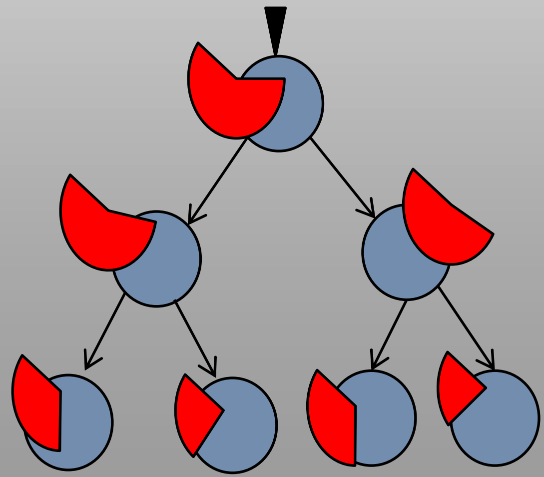
An Old Concept: Move-to-front, Splay Trees, …

- Classic data structures: lists, trees

- Linked list: move frequently accessed elements to front!



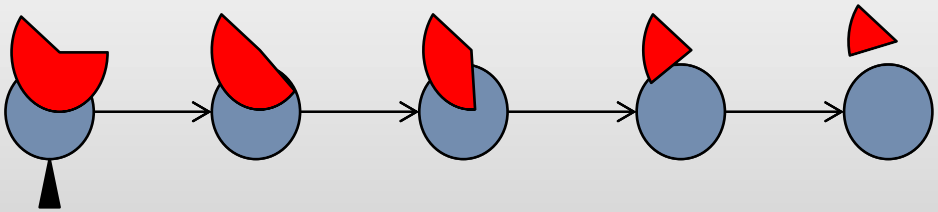
- Trees: move frequently accessed elements closer to root



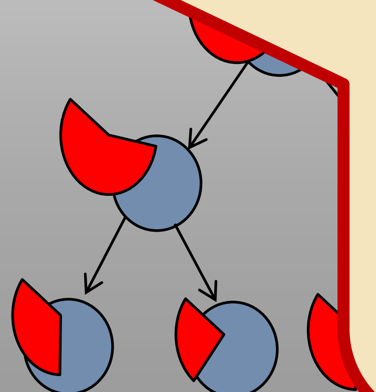
An Old Concept: Move-to-front, Splay Trees, …

- Classic data structures: lists, trees

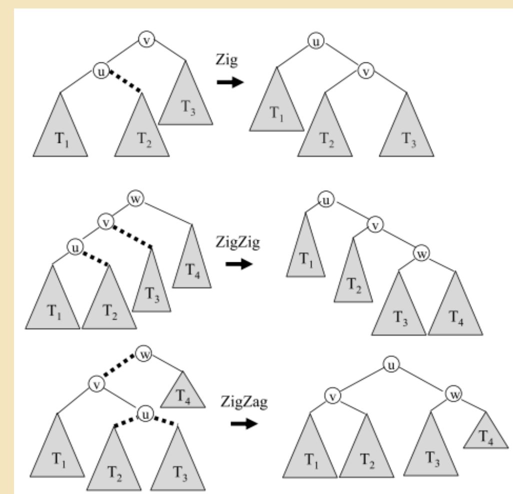
- Linked list: move frequently accessed elements to front!



- Trees: move frequently accessed el

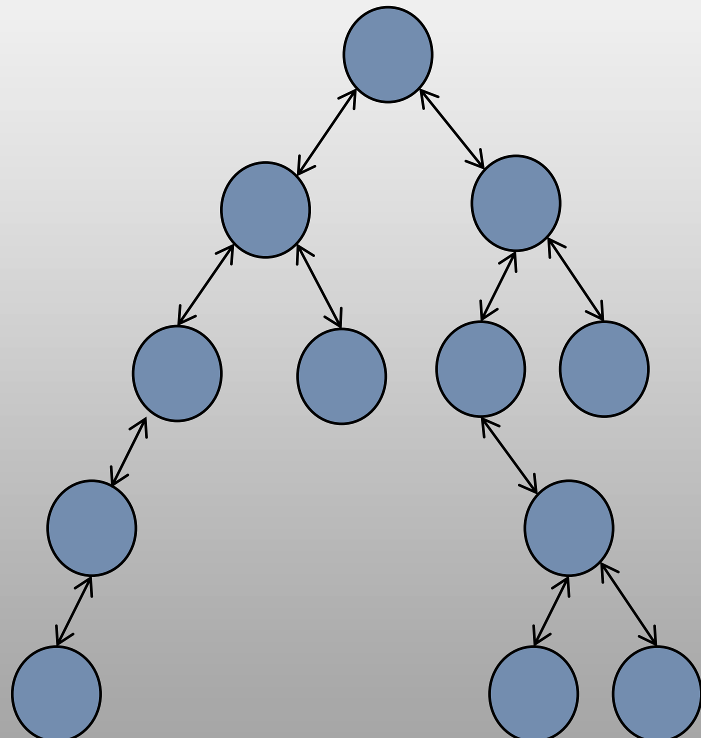


Splay Trees!



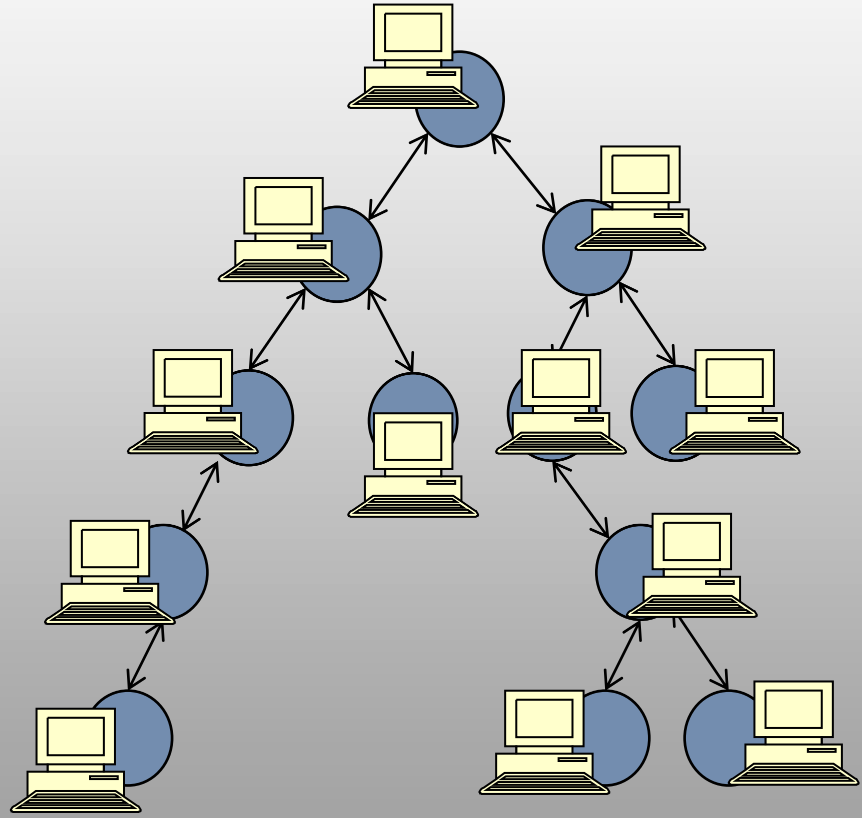
The Vision: Splay Networks ("Distributed Splay Trees")

- Most simple self-adjusting tree network: Binary Search Tree (BST)



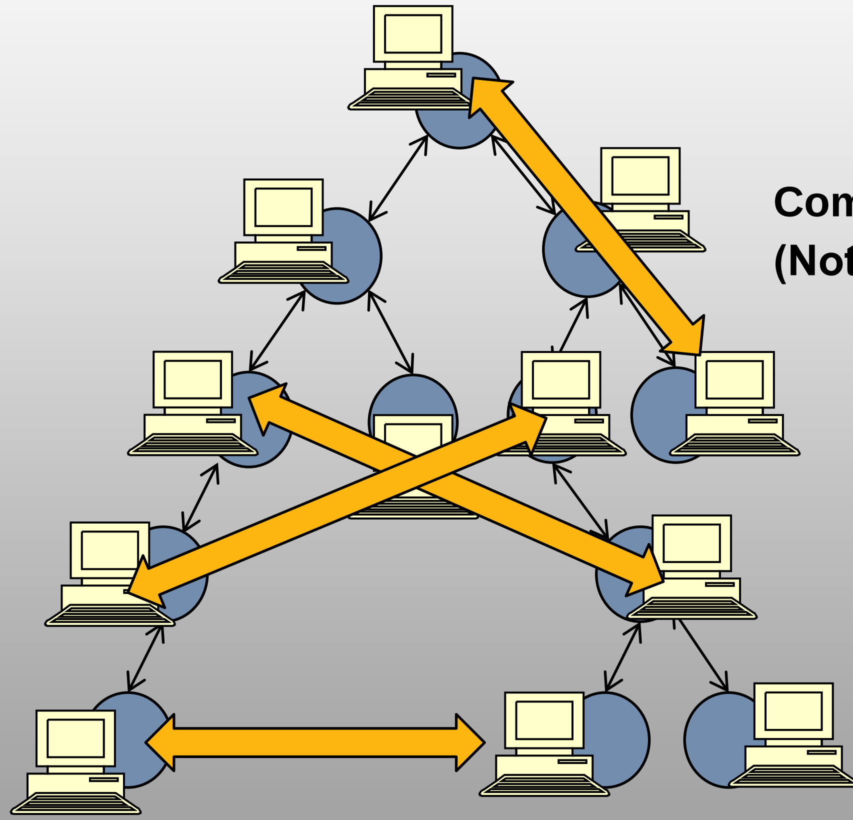
The Vision: Splay Networks (“Distributed Splay Trees”)

- Most simple self-adjusting tree network: Binary Search Tree (BST)



The Vision: Splay Networks ("Distributed Splay Trees")

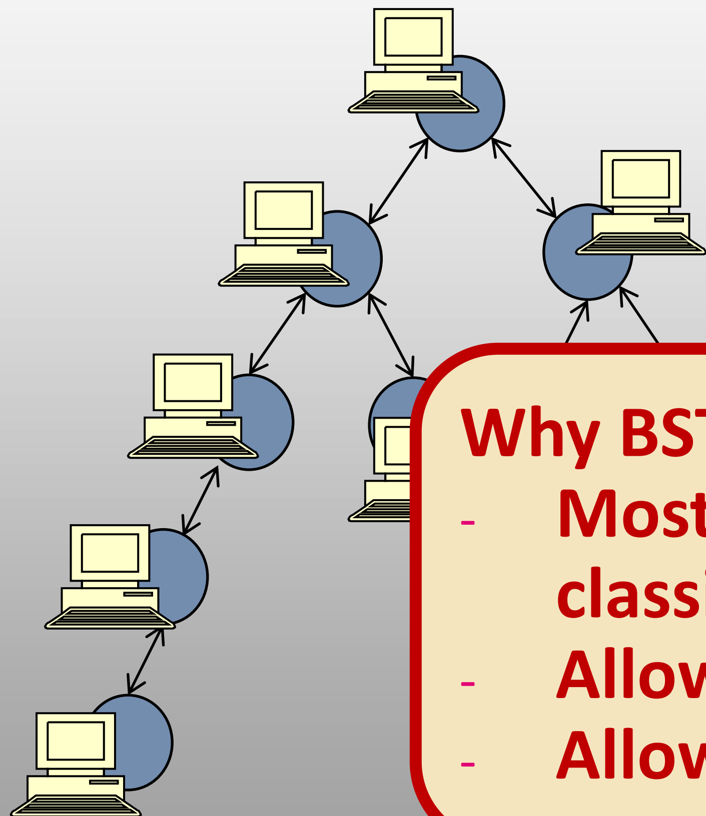
- Most simple self-adjusting tree network: Binary Search Tree (BST)



Communication between peer pairs!
(Not only lookups from root…)

The Vision: Splay Networks ("Distributed Splay Trees")

- Most simple self-adjusting tree network: Binary Search Tree (BST)



Why BST?!
- **Most simple generalization of classic data structure**
- **Allows for local routing!**
- **Allows for algebraic gossip**

Model: Self-Adjusting SplayNets

Input:

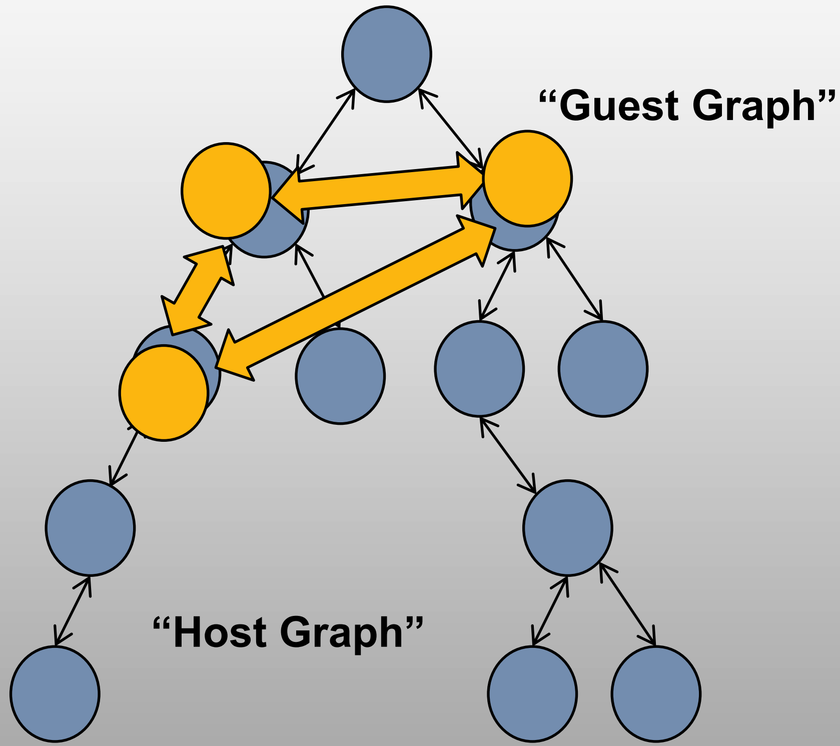
- communication pattern: (static or dynamic) graph

Output:

- sequence of network adjustments

Cost metric:

- expected path length
- # (local) network updates



Some Facts: Optimal Algorithm and Amortized Cost

Optimal Static Solution

- Dynamic program: decouple left from right!
- Polynomial time (unlike MLA!)
- So: solved M”BST”A

Dynamic Solution

- There exists self-adjusting algorithm
- Inspired by Splay trees
- E.g., optimal under product distribution: P[(u,v)]=P(u)*P(v)
- E.g., optimal under directed BST, non-crossing matching, …
- Lower bounds…

Upper Bound

A-Cost < H(X) + H(Y)

where H(X) and H(Y) are empirical entropies of sources resp. destinations

Adaption of Tarjan&Sleator

Lower Bound

A-Cost > H(X|Y) + H(Y|X)

where H(|) are conditional entropies.

Assuming that each node is the root for “its tree”

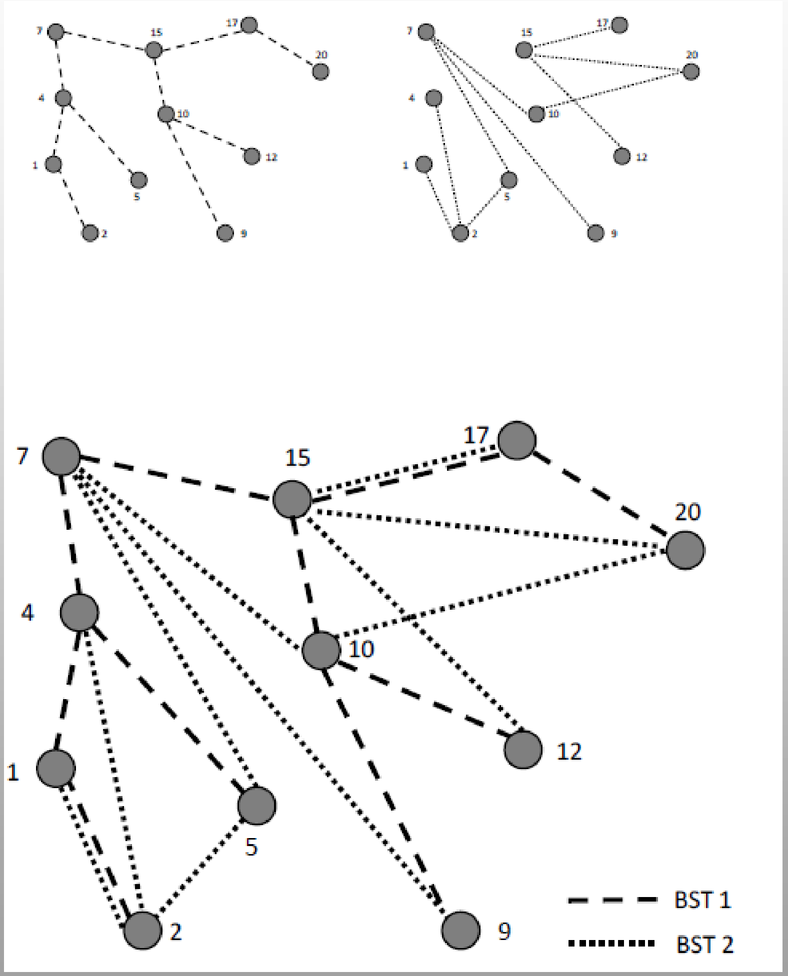
From One to Multiple BSTs

Research Question:

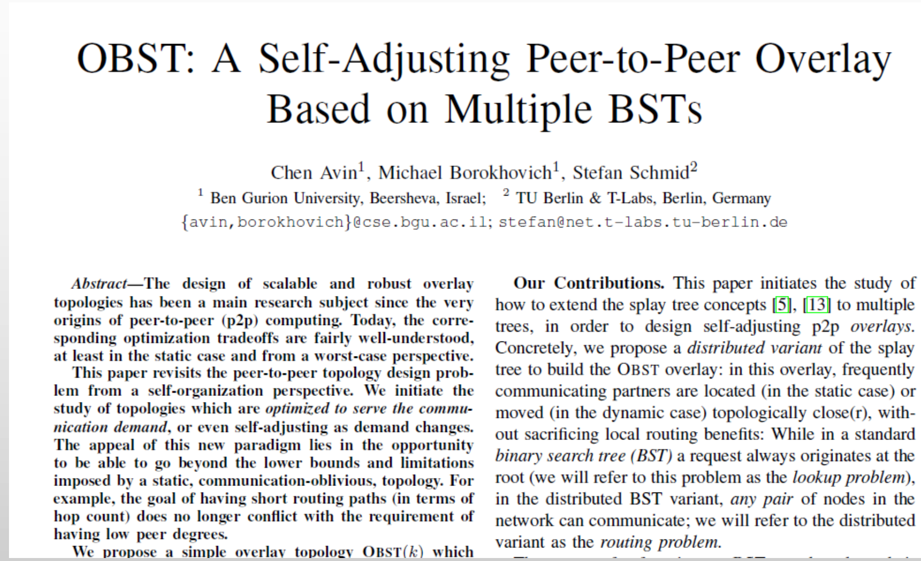
- What is the benefit of multiple BSTs?
- Focus on amortized communication cost

Two Models:

- <u>Lookup Model:</u> Classic datastructure where requests originate at root
- <u>Routing Model:</u> Peer-to-peer communication



Our Contribution



OBST: A Self-Adjusting Peer-to-Peer Overlay Based on Multiple BSTs

Chen Avin[1], Michael Borokhovich[1], Stefan Schmid[2]
[1] Ben Gurion University, Beersheva, Israel; [2] TU Berlin & T-Labs, Berlin, Germany
{avin,borokhovich}@cse.bgu.ac.il; stefan@net.t-labs.tu-berlin.de

Abstract—The design of scalable and robust overlay topologies has been a main research subject since the very origins of peer-to-peer (p2p) computing. Today, the corresponding optimization tradeoffs are fairly well-understood, at least in the static case and from a worst-case perspective.

This paper revisits the peer-to-peer topology design problem from a self-organization perspective. We initiate the study of topologies which are *optimized to serve the communication demand*, or even self-adjusting as demand changes. The appeal of this new paradigm lies in the opportunity to be able to go beyond the lower bounds and limitations imposed by a static, communication-oblivious, topology. For example, the goal of having short routing paths (in terms of hop count) does no longer conflict with the requirement of having low peer degrees.

We propose a simple overlay topology OBST(k) which

Our Contributions. This paper initiates the study of how to extend the splay tree concepts [5], [13] to multiple trees, in order to design self-adjusting p2p *overlays*. Concretely, we propose a *distributed variant* of the splay tree to build the OBST overlay: in this overlay, frequently communicating partners are located (in the static case) or moved (in the dynamic case) topologically close(r), without sacrificing local routing benefits: While in a standard *binary search tree (BST)* a request always originates at the root (we will refer to this problem as the *lookup problem*), in the distributed BST variant, *any pair* of nodes in the network can communicate; we will refer to the distributed variant as the *routing problem*.

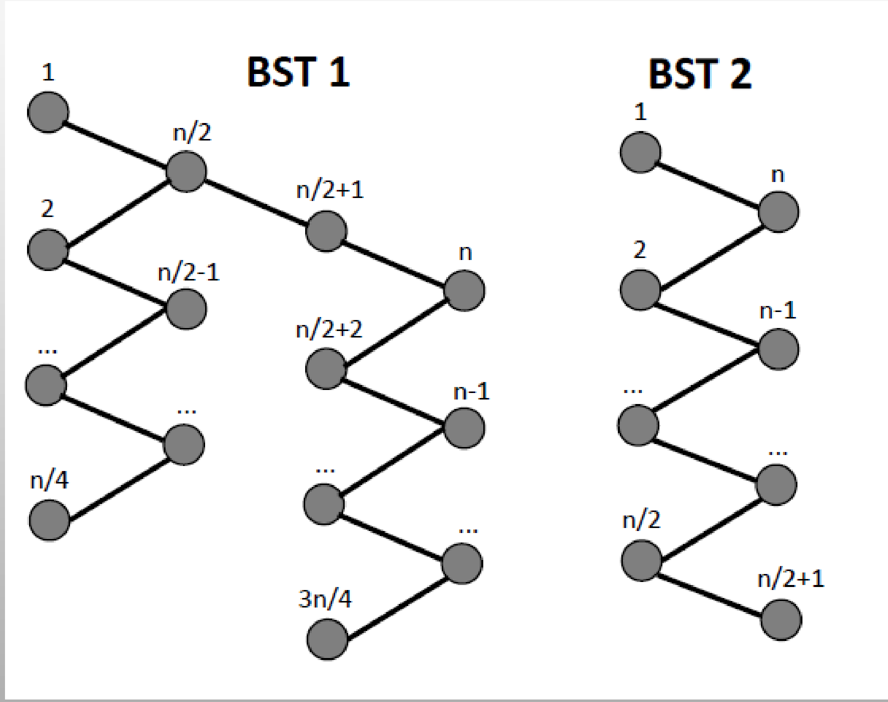
Routing:

- Static OBST:
  - A single additional BST can reduce costs from O(log n) to O(1)!
  - Entropy-based upper bounds on amortized communication costs
- Dynamic OBST:
  - Self-adjusting splay trees
  - Simulations
- Simulations

Lookup:

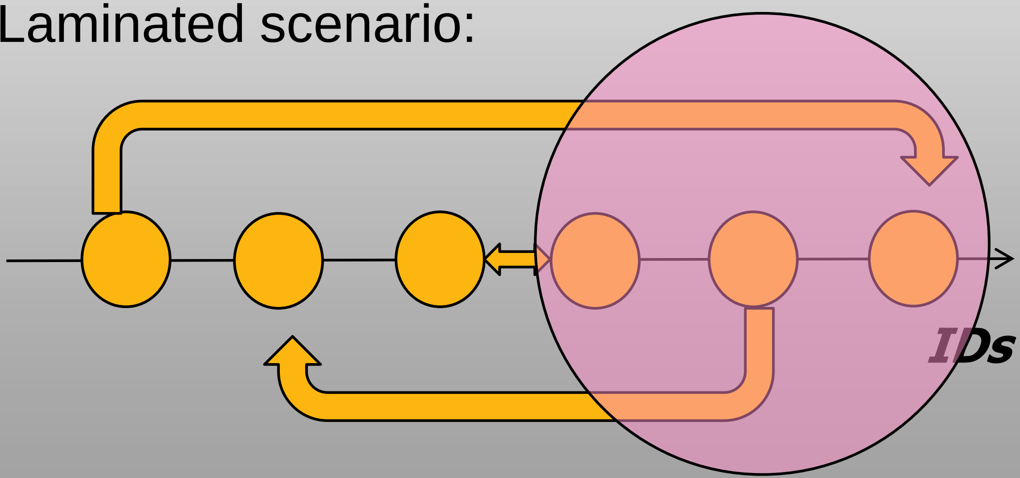
- Static OBST:
  - OBST(k) can only improve by additive –log(k) compared to OBST(1)

Routing: OBST(2) vs OBST(1)



- Easy to embed in two BSTs: one for each (cost O(1))
- Hard to embed in one BST: because large interval cut ("crossing-matching")

Laminated scenario:

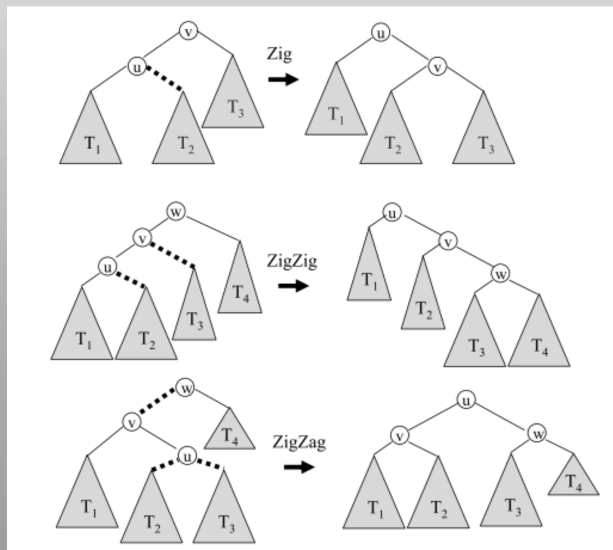


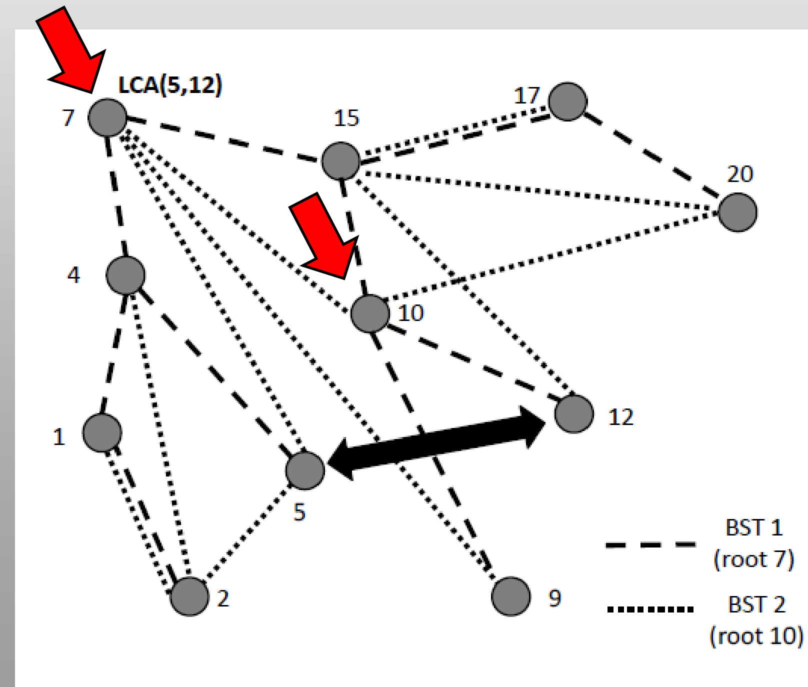
Self-Adjusting OBST

Algorithm 1 Dynamic OBST(k)

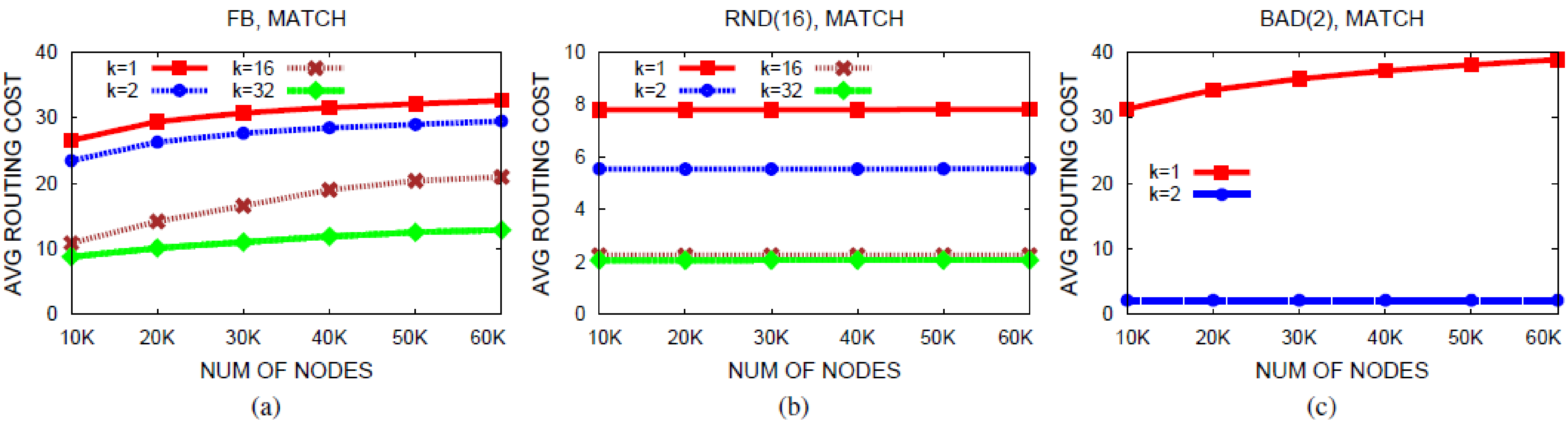
(* upon request (u, v) *)
find BST $T \in$ OBST where u and v are closest;
$w :=$ LCA$_T(u, v)$;
$T' :=$ **splay** u to root of $T(w)$;
splay v to the child of $T'(u)$;

- Splay to Least Common Ancestor (more local!)…
- … in best tree!



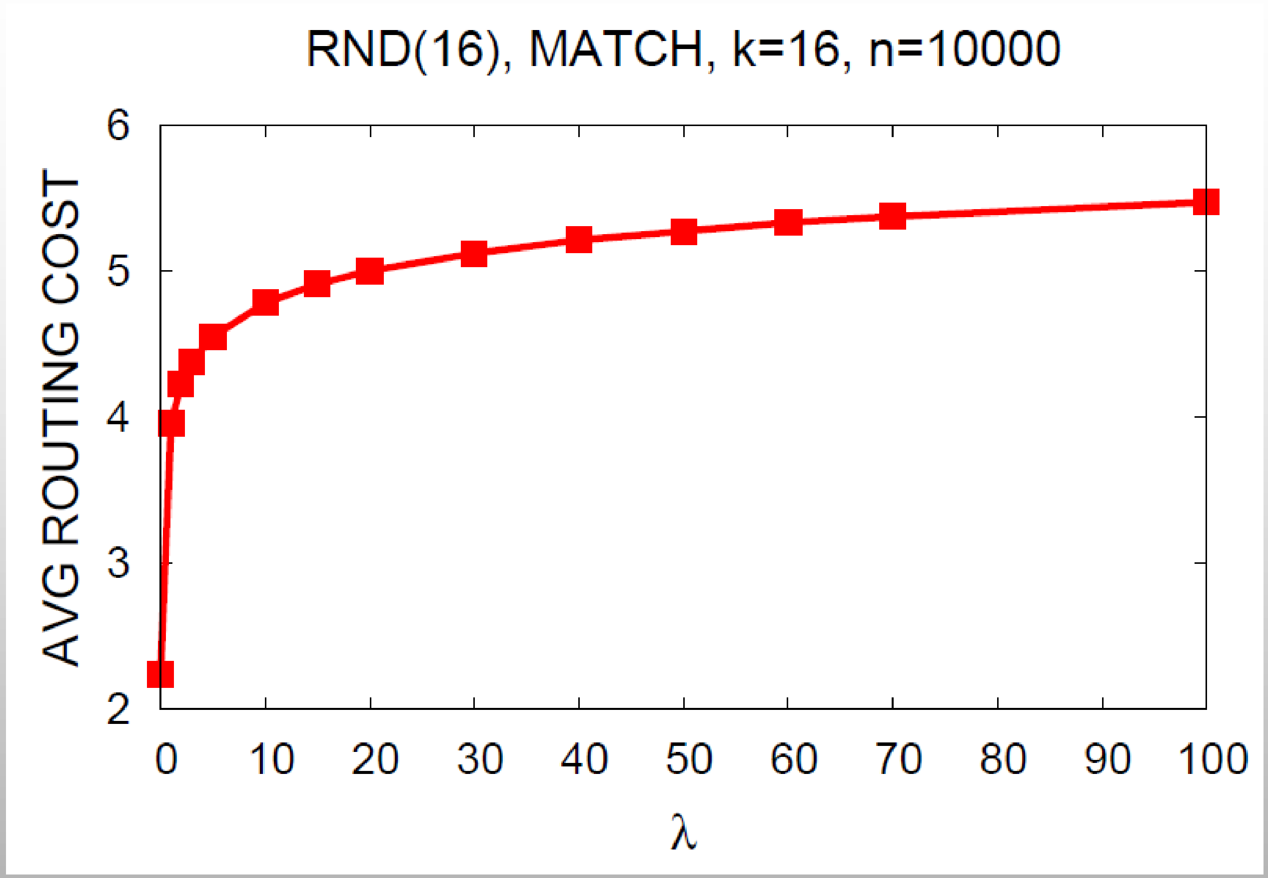


Simulation



- Initially: k independent, random BSTs
- Communication models: matching and random walk
- (a) More trees help
- (b) On Random Graph relatively stable, almost perfect convergence
- (c) OBST(2) convergence to perfect tree for “bad example”

Churn



- Routing cost under λ joins/leaves between a lookup operation
- More plots in full paper…

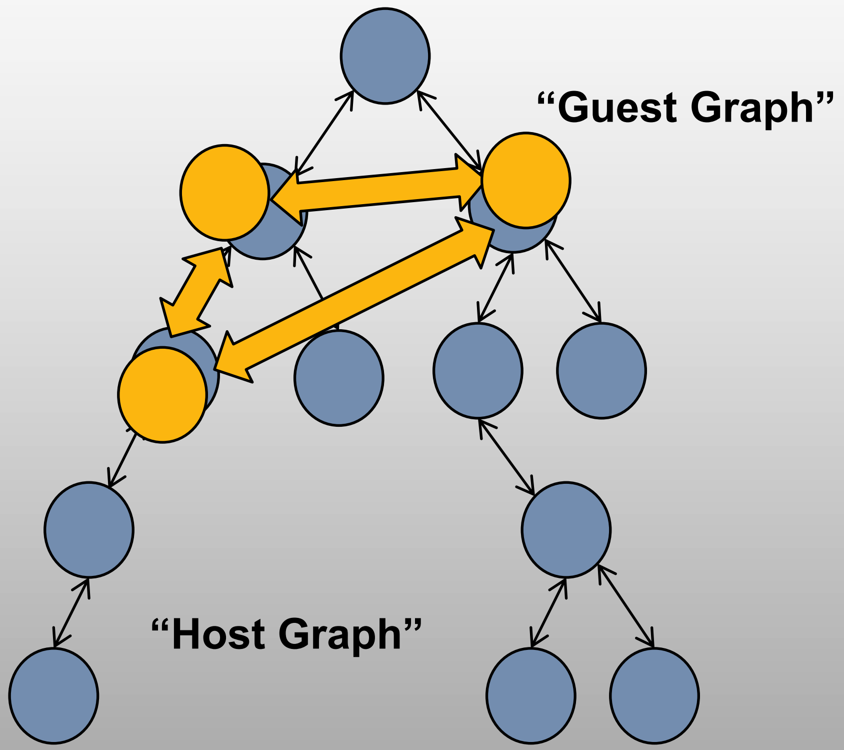
Conclusion

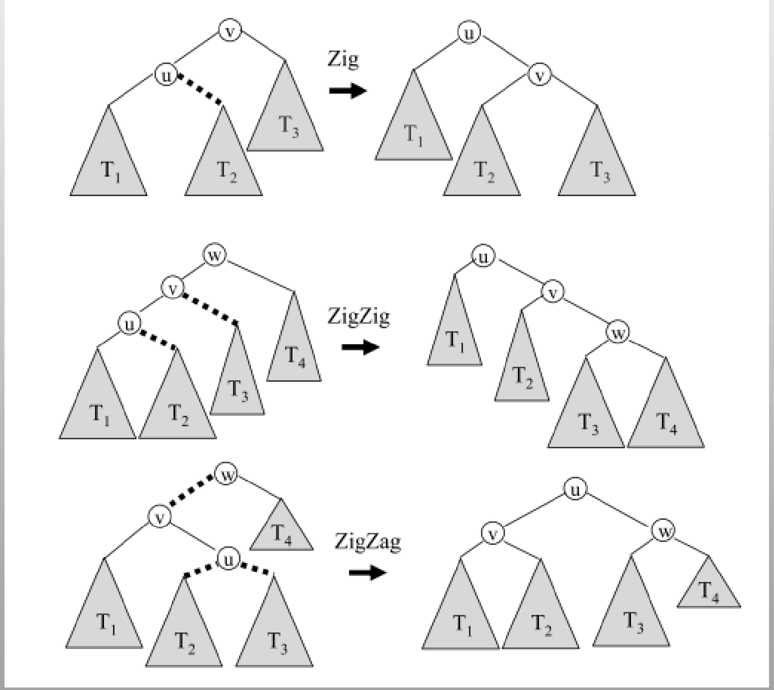
- Vision: self-adjusting networks
- Interesting generalization of Splay trees
- SplayNets
  - Formal analysis reveals nice properties
  - Amortized costs good: but tight?
  - Competitive ratio remains open
- OBST: lookup vs routing

Thank you! Questions?

Algorithm 2 Double Splay Algorithm DS

1: (* upon request (u, v) in T *)
2: $w := \alpha_T(u, v)$
3: $T' :=$ **splay** u to root of $T(w)$
4: **splay** v to the child of $T'(u)$





Advertisement

The Workshop on Distributed Cloud Computing (DCC 2013) will be co-located with IEEE/ACM Conference on Utility and Cloud Computing (UCC): December 9, 2013, Dresden, Germany

News

- [August 22, 2013] Rick McGeer, Distinguished Technologist at HP Enterprise Services, will give the keynote address
- [August 2, 2013] EasyChair submission closed
- [July 8, 2013] EasyChair submission open
- [June 1, 2013] Website online

Dates

- Submissions due: **2 August 2013 (hard!)**
- Notification of acceptance: **10 September 2013**
- Camera-ready papers due: **27 September 2013**
- Workshop: **9 December 2013**
- IEEE/ACM UCC Conference: **9-12 December 2013**

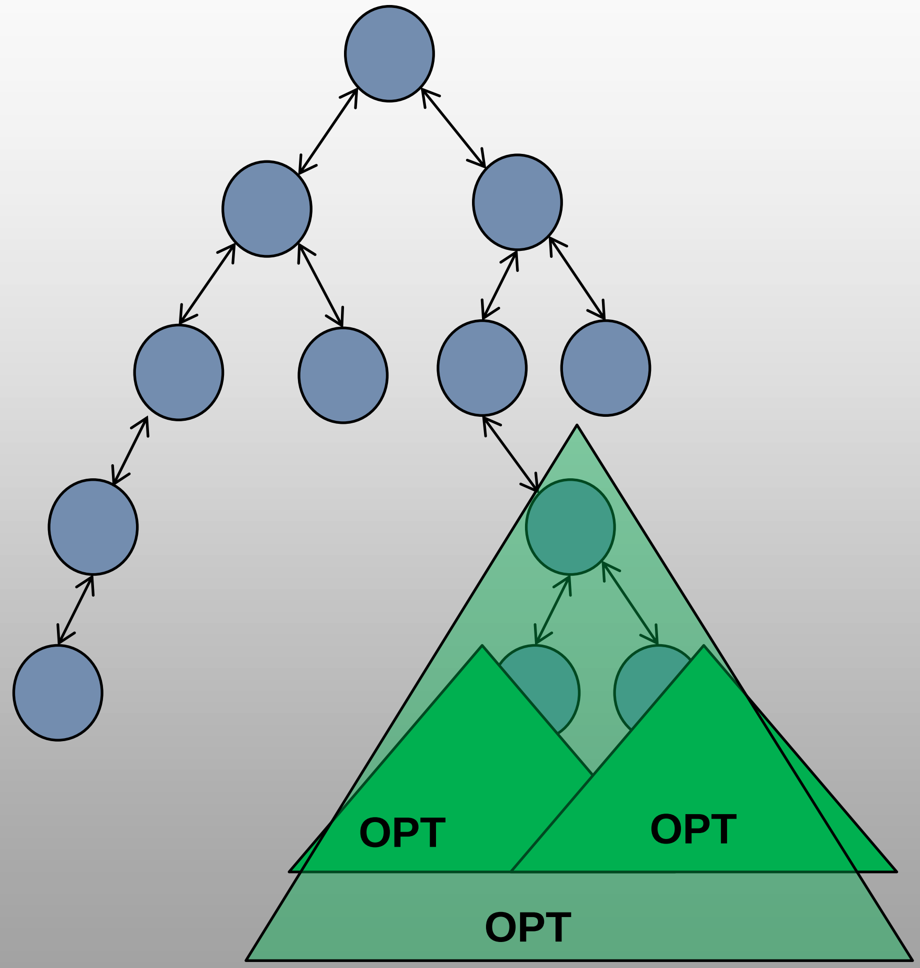
Peer-to-Peer and the Cloud!

Keynote by Rick McGeer, papers by Jen Rexford, Holger Karl, Christof Fetzer, Pietro Michardi, etc.

Backup: The Optimal Offline Solution

Dynamic program

- Binary search:
  decouple left from right!
- Polynomial time
  (unlike MLA!)
- So: solved M"BST"A



See also:

- Related problem of
  phylogenetic trees