

Fast Stochastic Exploration of Tree-based Content Distribution Architectures

Damiano Carra, Renato Lo Cigno
Department of Information and Communication
Technology (DIT)
University of Trento

Ernst W. Biersack
erbi@eurecom.fr
Institut Eurecom
France

Outline

- Homogeneous case
 - ◆ Deterministic Performance
- Heterogeneous case
 - ◆ Analytical Solution
 - ◆ Numerical (Monte-Carlo) Solution
- Summary
- Open issues

The basic "single-chain" model

We consider a cooperative approach

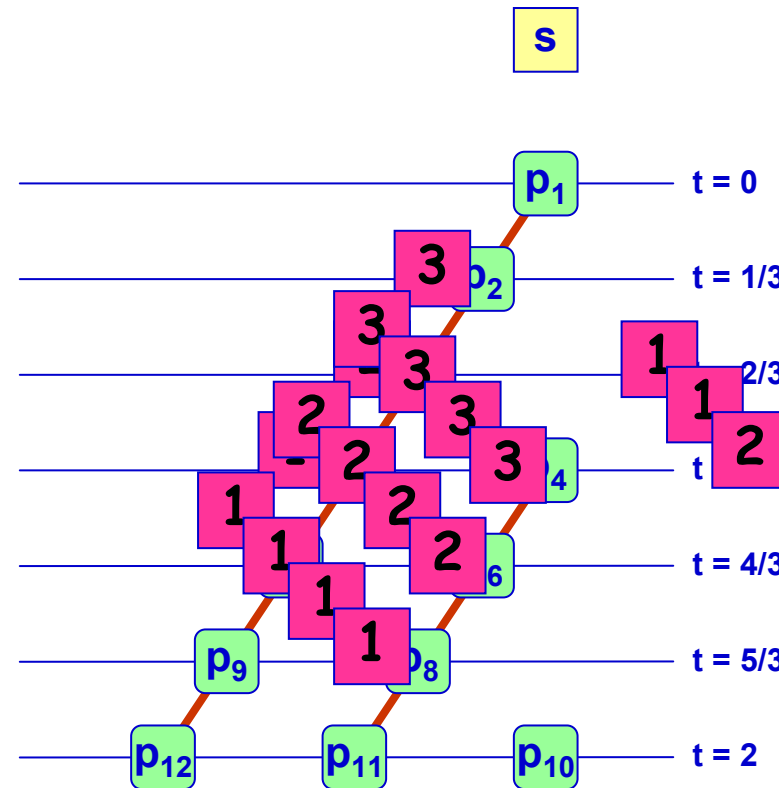
- every peer serves the whole file once to another peer and disconnects
- a peer can start serving once it has 1 chunk
- each peer can reach all the other peers in one hop
- file F divided in C independent pieces called "chunks"

C has a great influence on the performance!

- the only bottleneck is the access link

We evaluate a basic distribution architectures

- single and multiple chains

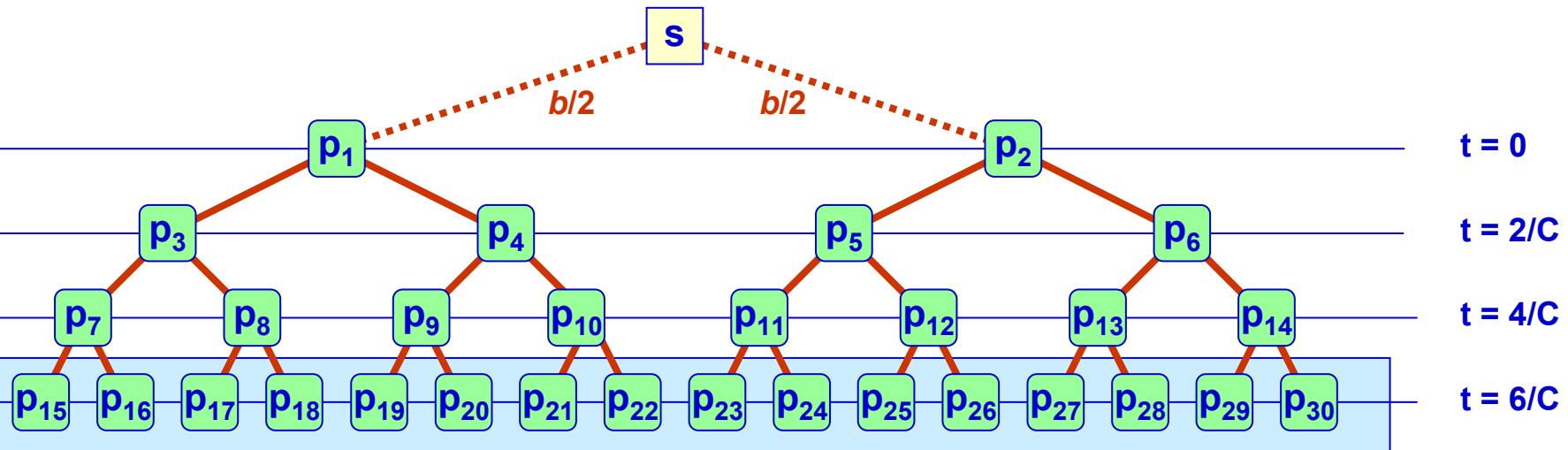


Homogeneous case:
each peer has the same capacity
($C=3$)

Tree^k

- Server uploads the file to k peers in parallel at rate b/k
- Every peer downloads the file at rate b/k in k rounds
- Non-leaf peers upload the whole file to k peers at rate b/k
 - ◆ Serves the file k times
- Linear chain when $k=1$

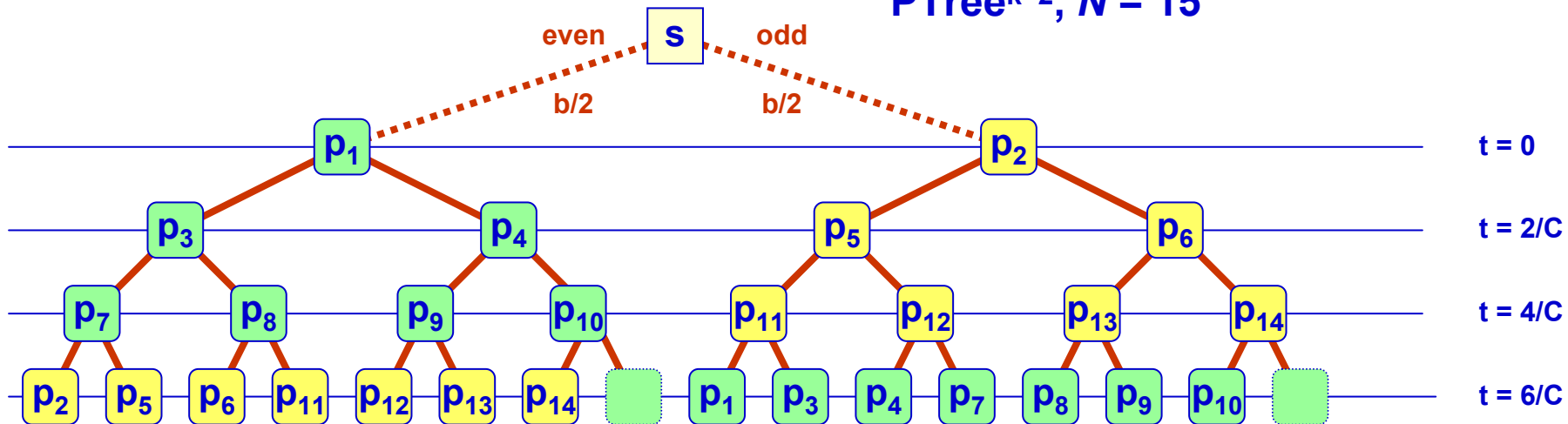
Tree^{k=2}, $C = 3$, $N = 30$



PTree^k

- Server uploads $1/k$ of the file to k peers in parallel at rate b/k
- Every peer downloads the file from k peers at rate b/k
- Every peer uploads $1/k$ -th of the file to k peers at rate b/k
 - Serves the file *once*
- Creates k parallel spanning trees, linear chain when $k=1$
 - Each peer is interior node of at most one tree \Rightarrow reliability

PTree^{k=2}, $N = 15$



Performance for Deterministic Case

		Time until all nodes have the file	Copies served	Download & upload rate
<i>Linear</i>		$\frac{C + \sqrt{C^2 + 8 \cdot N \cdot C}}{2 \cdot C}$	1	b b (- leaves)
<i>Tree^k</i>		$k + \left\lfloor \log_k \left(\frac{N}{k} \right) \right\rfloor \cdot \frac{k}{C}$	k	b/k b (- leaves)
<i>PTree^k</i>		$1 + \left\lfloor \log_k N \right\rfloor \cdot \frac{k}{C}$	1	$b/k * k = b$ b



- PTree^k performs best

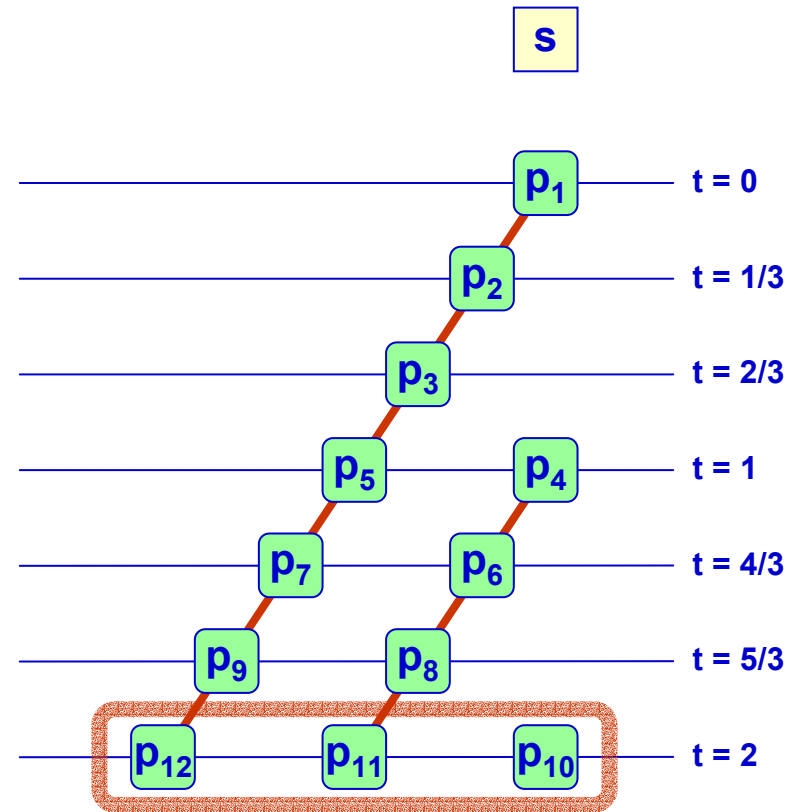
Opposite from ADSI

Simple Analytical Solutions

March 27, 2006

Linear Chain

- Cooperative approach
 - Every peer serves the whole file once to another peer
 - A peer can start serving once it has 1 chunk
- Many chunks improve scalability



After t rounds: $t+1$ chain

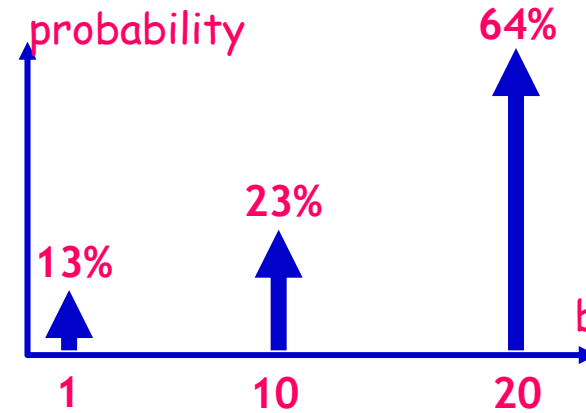
Linear, $C = 3$

Chain has $1+tC$ peers ($1+(t-1)C$ complete)

Chain: From homogeneous to heterogeneous

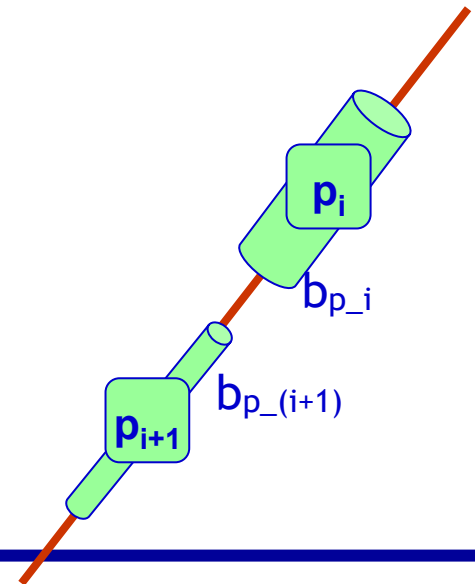
We assume that

- ◆ there are peers with different access (symmetric) bandwidths
 - ☞ the distribution of the bandwidths follows a *probability density function* (pdf),
 - considered known
- ◆ a peer has no knowledge about its neighbors capacity
 - ☞ neighbor bandwidth is a random variable (b_p)



Hypothesis and definitions

- ◆ All peers are i.i.d. \rightarrow single r.v. b_p
- ◆ Transfer bandwidth b_i $\rightarrow \min(b_{p_i}, b_{p_{(i+1)}})$
- ◆ Chunk transfer time $\rightarrow (F/C) * (1/ b_i)$

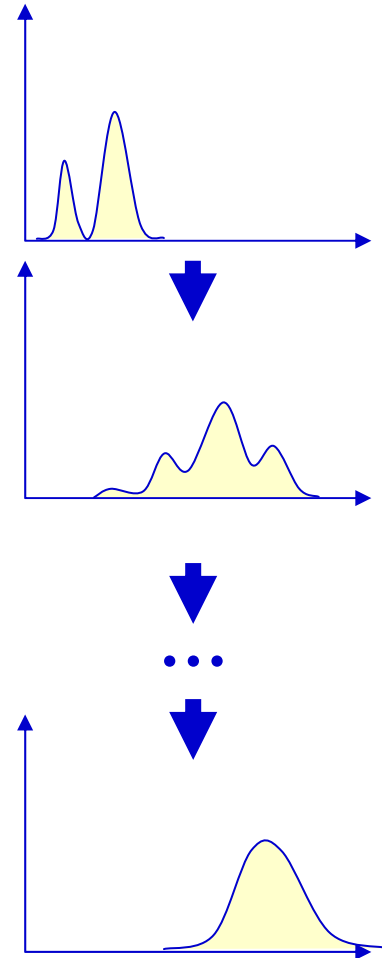


Chain: Methodology

Basic idea

- ◆ we study how the pdf evolves along the chain
 - ◆ from the pdf, we can derive the mean, quantiles, ...
- Each step is independent from the previous ones → we look for an iterative formulation of the problem
- ◆ find the pdf after reaching n -th peers from
 - ☞ the pdf after reaching $(n-1)$ -th peers
 - ☞ pdf of a peer (i.e., $pdf^{(1)}$, the starting pdf)

$$pdf^{(n)} = f(pdf^{(n-1)}, pdf^{(1)})$$



Chain: Detailed model

Total download time

- ◆ $t_{\text{total}} = t_{\text{reach}} + t_{\text{downl}}$

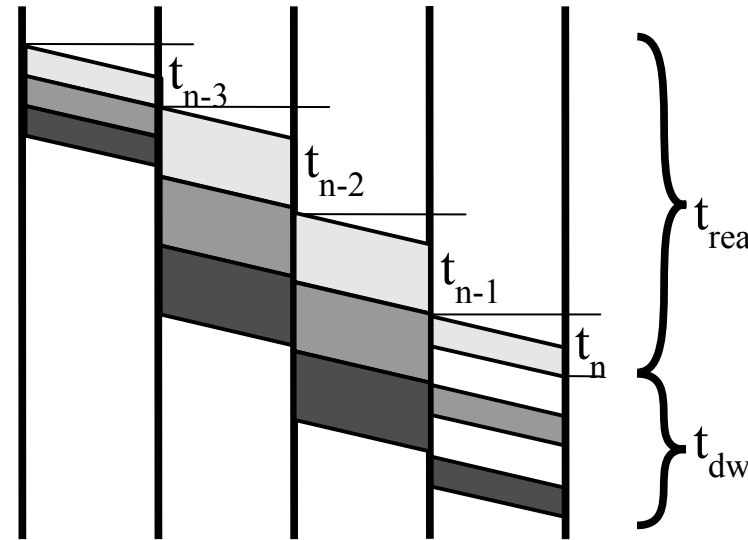
t_{reach}

- ◆ time necessary to reach the n-th peer

t_{downl}

- ◆ time necessary to download the whole file

We obtain

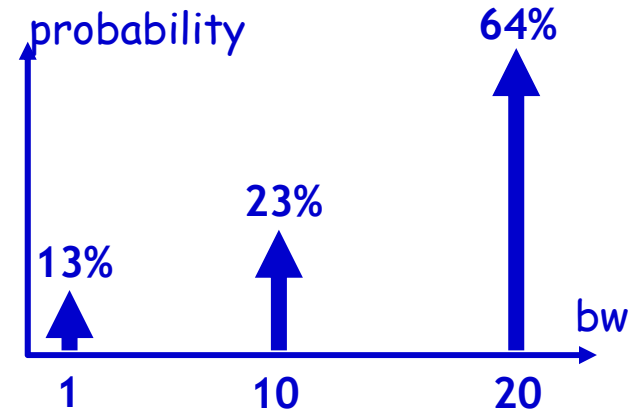


$$t_{\text{reach}}^{(n)} = t_{\text{reach}}^{(n-1)} + \frac{F}{Cb_n} = \frac{F}{C} \sum_{i=1}^n \frac{1}{b_i}$$

$$t_{\text{downl}}^{(n)} = \max \left(t_{\text{downl}}^{(n-1)}, (C-1) \frac{F}{Cb_{p_n}} \right) = (C-1) \max \left(\frac{F}{Cb_{p_1}}, \frac{F}{Cb_{p_2}}, \dots, \frac{F}{Cb_{p_n}} \right)$$

Chain: Numerical example

- We show the results with 3 classes of peers
- We define as one round the time it takes to the slowest class to download the whole file
 - ◆ $F/b_{\text{slow}} = 1$ round
- We vary the initial pdf decreasing the percentage of slow peers

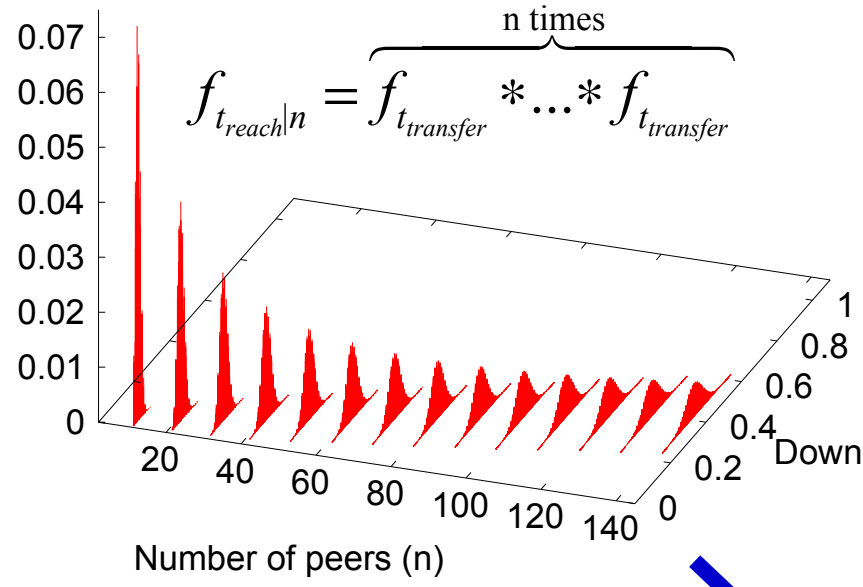


%	13%	23%	64%
bw	1	10	20
<i>time (rounds)</i>	<i>1</i>	<i>0.1</i>	<i>0.05</i>

Probability

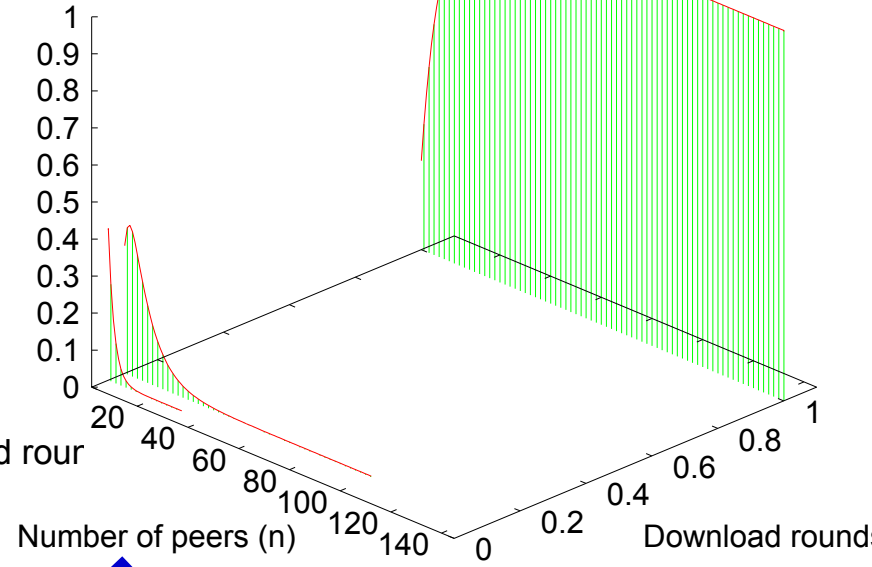
t_{reach}

$$f_{t_{reach}|n} = \overbrace{f_{t_{transfer}} * \dots * f_{t_{transfer}}}^{n \text{ times}}$$



Probability

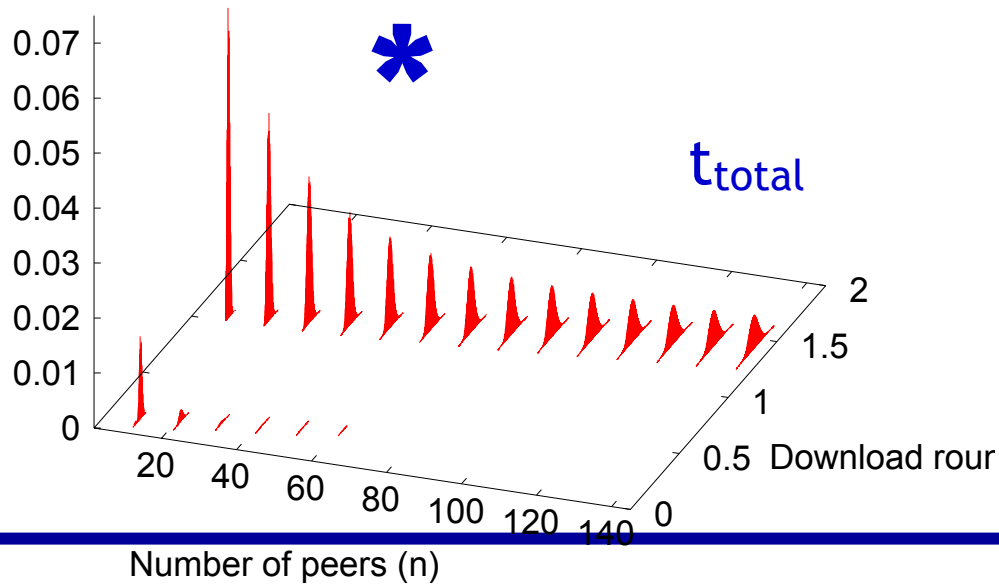
t_{down}



Probability



t_{total}



March 27, 2006



Chain: From steps to time

Aim of the model

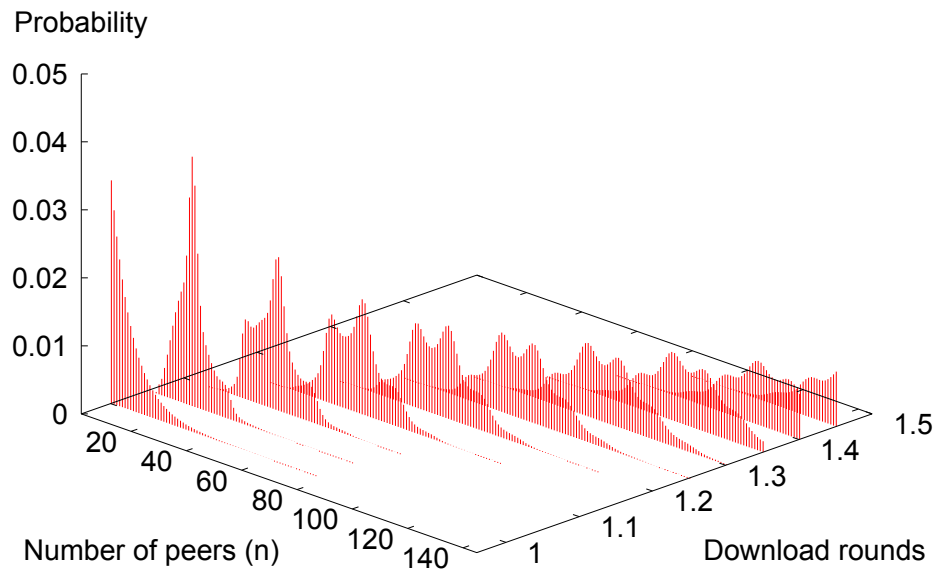
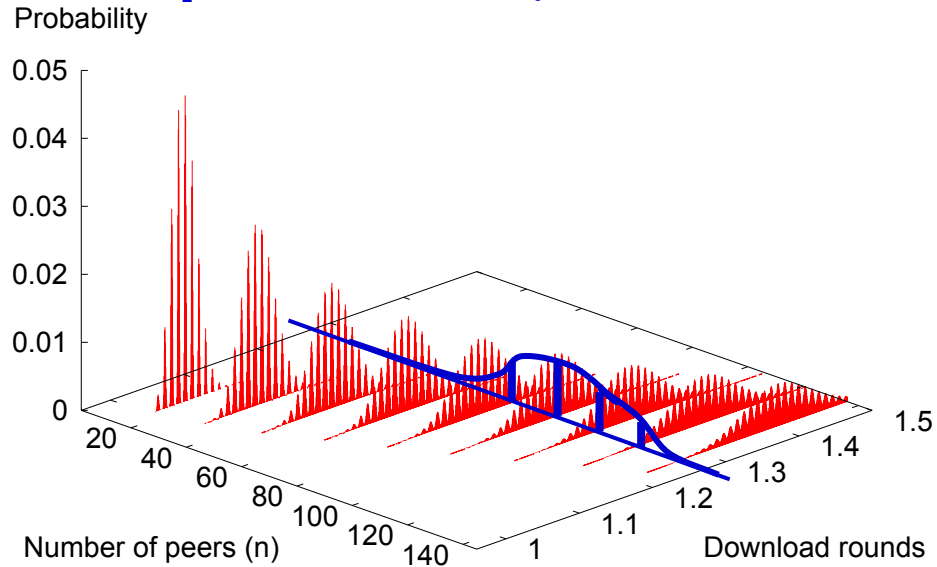
- ◆ answering to question like "Which is the mean download time after n peers?"

☞ conditional probability at after n peer

Instead, we would also

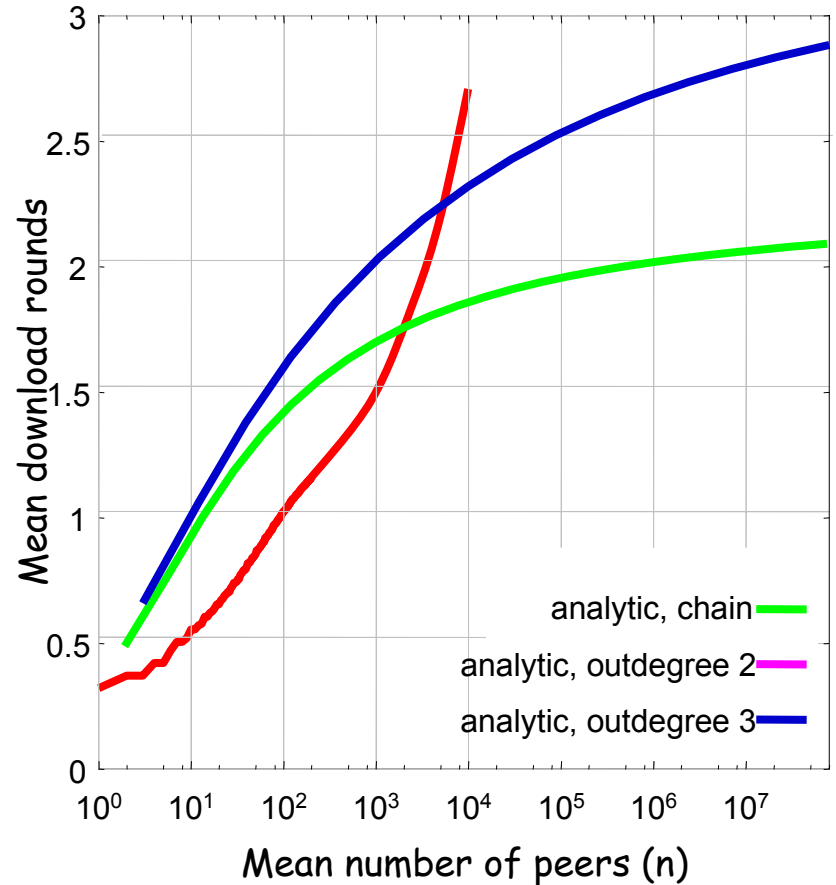
- ◆ find the pdfs at each time instant, i.e., find "Which is the mean number of peers reached at time t ?"

☞ conditional probability at time t



Results on Tree

- We consider the mean download rounds
- As the number of peers increases, the curves tend to k rounds
 - ◆ $k \rightarrow$ outdegree
- Note: short chain performs better than small trees

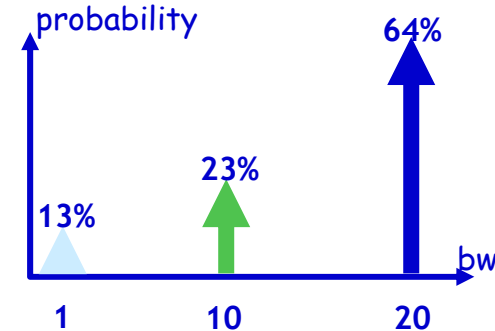
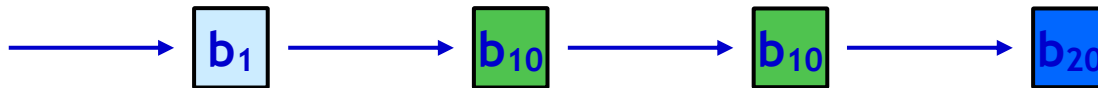


%	13%	23%	64%
bw	1	10	20
<i>time (rounds)</i>	<i>1</i>	<i>0.1</i>	<i>0.05</i>

Efficient Numerical Solution (Monte-Carlo Simulation?)

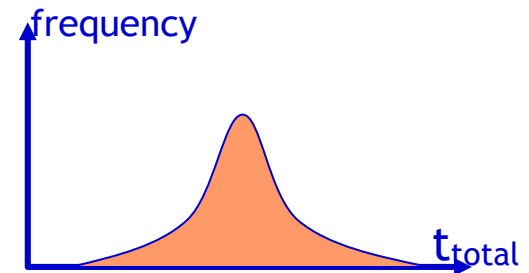
Model simulator

- Simulator to evaluate the impact of approximations made in the analysis of tree
- Monte Carlo analysis of the solution space
- At each step, the simulator
 - ◆ Chooses randomly a peer
 - ◆ Calculates the t_{reach} and t_{dwnl}
 - ◆ Registers (builds the histogram of) t_{total}



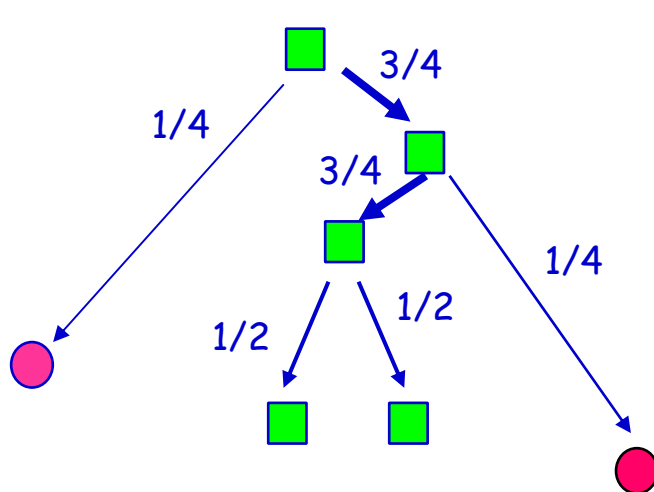
$$t_{reach}^{(n)} = t_{reach}^{(n-1)} + \frac{F}{Cb_n} = \frac{F}{C} \sum_{i=1}^n \frac{1}{b_i}$$

$$t_{dwnl}^{(n)} = (C-1) \max \left(\frac{F}{Cb_{p_1}}, \frac{F}{Cb_{p_2}}, \dots, \frac{F}{Cb_{p_n}} \right)$$



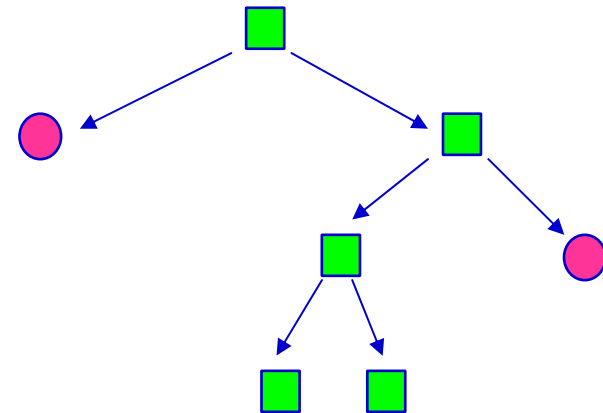
Tree: How to grow a tree

- Fast node (rate $r = 1$)
- Slow node (rate $r = 1/4$)



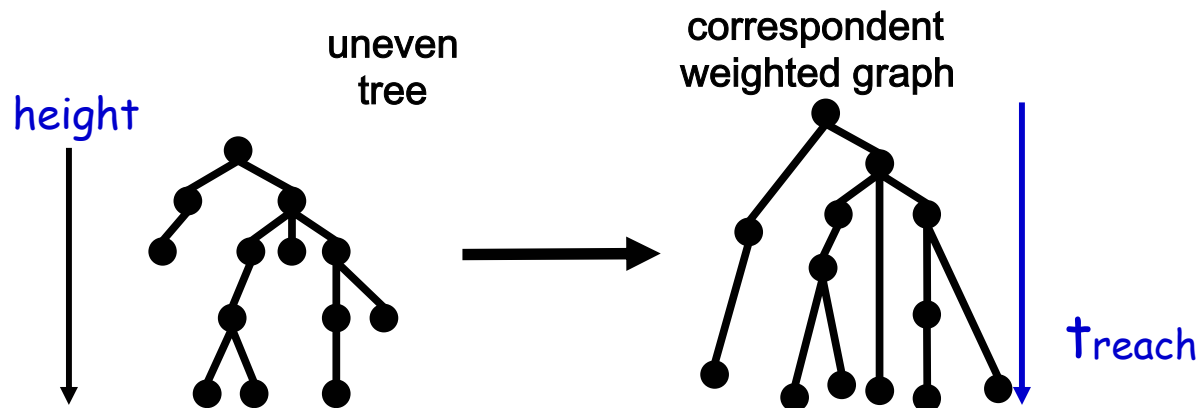
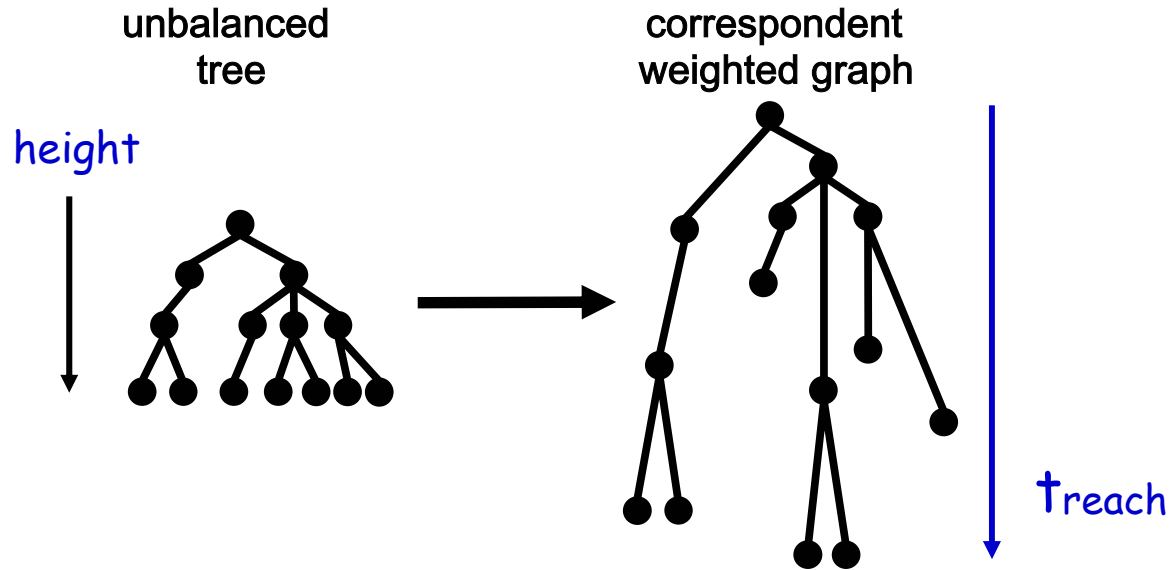
Uneven tree

†reach



Trees: Unbalanced or uneven

- Trees evolve in an uneven fashion
- Many models consider unbalanced tree...
- The **weights** are the difference of the download times between the two nodes



"Discrete event"-like simulator

- At each iteration i a **full distribution architecture is simulated**
 - ◆ The server selects its children and sets the correspondent events "first chunk uploaded"
 - ◆ Each node, when the event "first chunk uploaded" occurs, selects randomly its children and sets the correspondent events "first chunk uploaded"
 - ☞ the children selection is made randomly among the neighbor set of the node
 - ☞ the number of children and the event time are given by
 - the scenario constraints (maximum number of levels, minimum and maximum number of children)
 - the input bandwidth distribution (pdf)
 - ◆ When no more nodes can be reached, the simulator computes the statistics and makes another iteration
 - ☞ mean of all the download times $\rightarrow T_{\text{mean}}(i)$
 - ☞ update overall mean $\rightarrow (\sum T_{\text{mean}}(i))/i$
 - ☞ register the download time of a randomly chosen node T_{rand}

"Discrete event"-like simulator (2)

■ Stop criterion

- ◆ When the confidence level for the overall mean is reached, stop the simulation

■ Outputs

- ◆ Overall mean download time
- ◆ The samples T_{rand} are used to create the normalized histogram that represents the pdf of the download times

Sample path version

- At each iteration i only a path inside the distribution architecture is simulated
 - The server
 - selects its children among its neighbors
 - computes the time they take to download the first chunk
 - chooses randomly one of its children
 - The chosen children repeats exactly the same operations of the server
 - At each step, the number of children are registered and from there the total number of nodes in the tree is inferred
 - When the maximum level or the maximum time is reached, it computes the statistics and makes another iteration

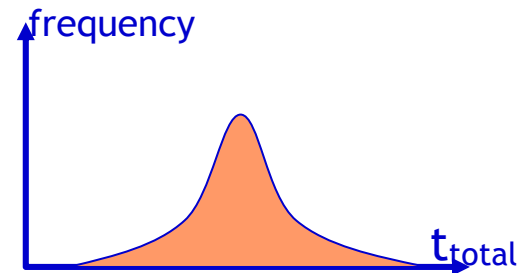
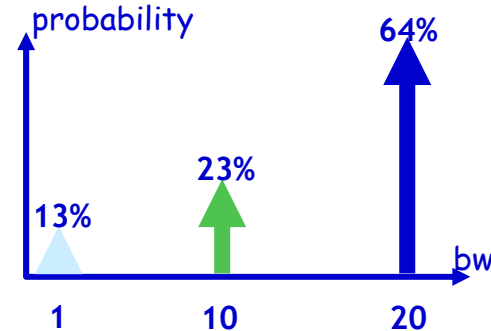
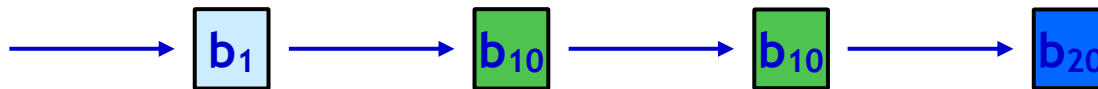
mean of all the download times

$$T_{mean}(i) = \frac{\sum_{step} T_{step_j}^{dwld} * \#nodes_{step_j}^{inferred}}{\#nodes_{global}^{inferred}}$$

- update overall mean $\rightarrow (\sum T_{mean}(i))/i$
- register the download time of a randomly chosen node
- Stop criterion and statistics are the same of the "discrete event"-like version

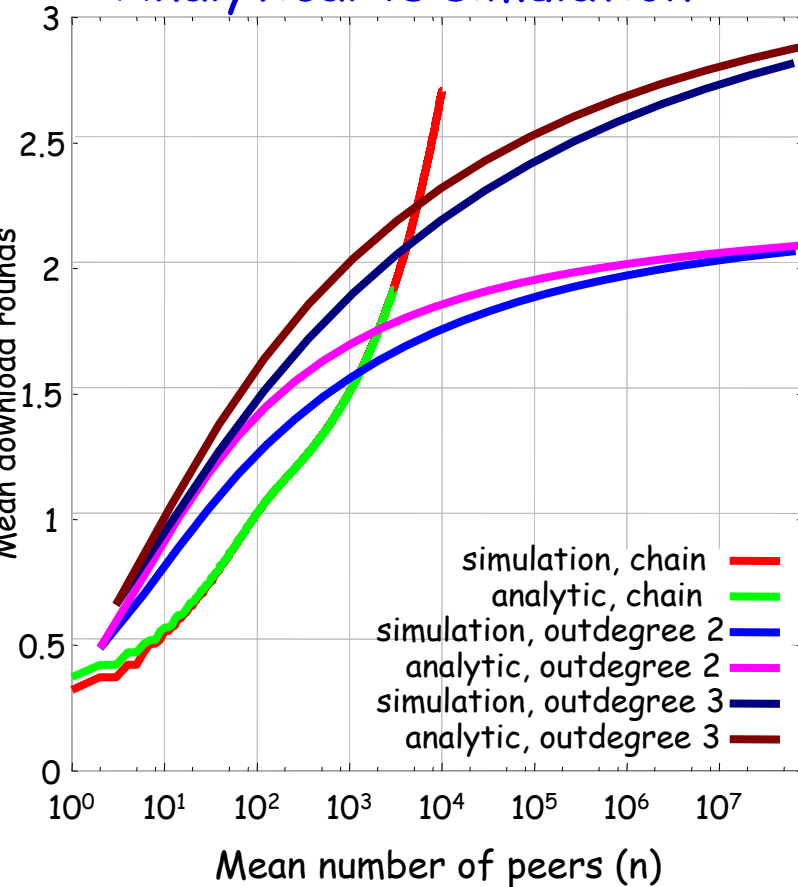
Model simulator

- We developed a simulator to evaluate the impact of approximations made in the analysis of tree
- It performs a Monte Carlo analysis of the solution space
- At each step, the simulator
 - ◆ Chooses randomly a peer
 - ◆ Calculates the t_{reach} and t_{dwnl}
 - ◆ Registers (builds the histogram of) t_{total}

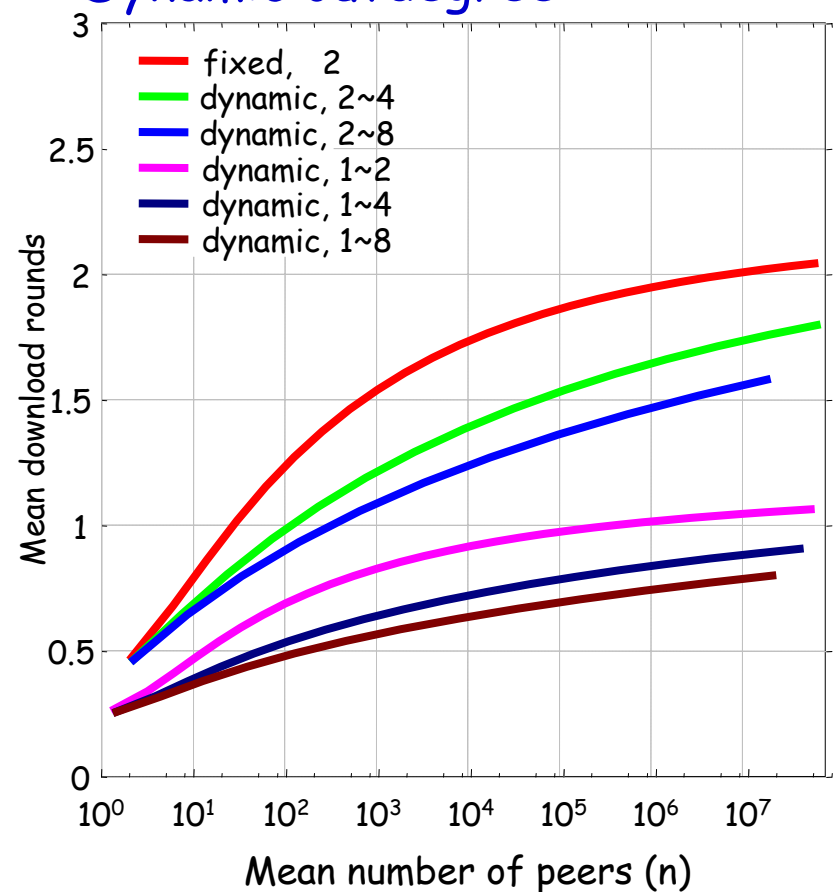


Results on Tree

Analytical vs simulation



Dynamic outdegree



- $F/b_{\text{slow}} = 1$ round; with outdegree k , trees converge to k rounds
- with dynamic outdegree we reduce the speed of this convergence
- with min outdegree = 1, trees converge to 1 round

Simulator: Performance

- We run the two different versions of the simulators on the same machine using the same set of configurations
- Comparison with respect to number of iterations and CPU time

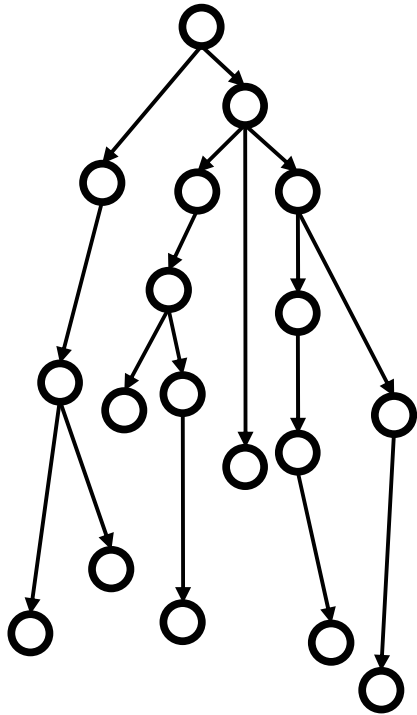
# peers	Full Tree		Sample Path	
	Iterations	CPU time (sec)	Iterations	CPU time (sec)
10^2	2000	4	10^6	19
10^3	2000	37	10^6	33
10^4	1000	372	10^6	45
10^5	500	6507	10^6	57
10^6	200	~ 20 days	10^6	71
10^7	-	-	10^6	84
10^8	-	-	10^6	112

March 27, 2006

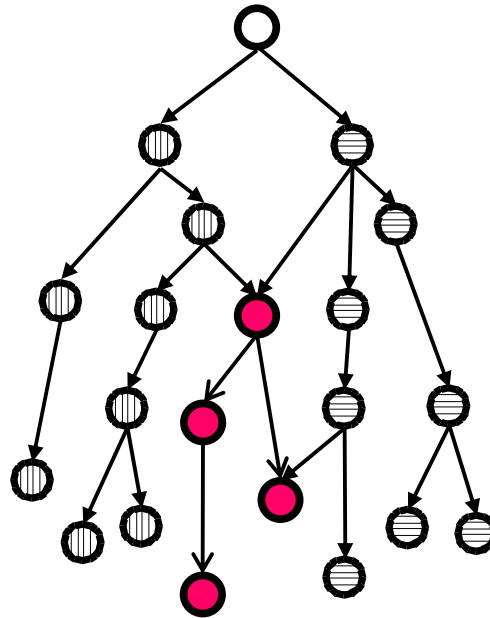
Mesh approach

- Tree based architectures have known shortcomings
 - ◆ single father
 - ◆ upload bandwidth shared among children
 - ◆ leaf capacity unused
- Mesh based approach can solve these problems
 - ◆ but introduce the issue of finding nodes with disjoint content
- The stochastic process used to build mesh networks is based on the observation that
 - ◆ At the first stage, the diffusion process is a tree
 - ◆ Then, trees start to merge

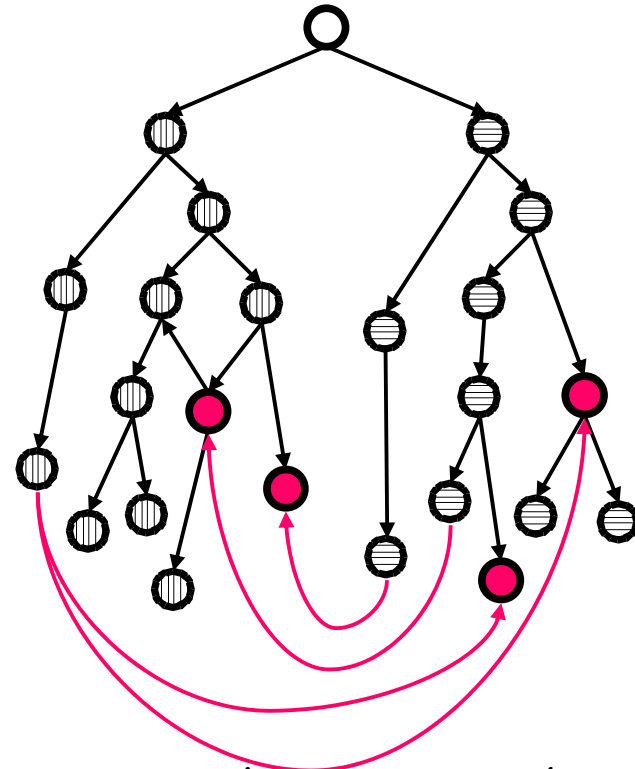
Examples of trees and meshes



diffusion tree
(a)

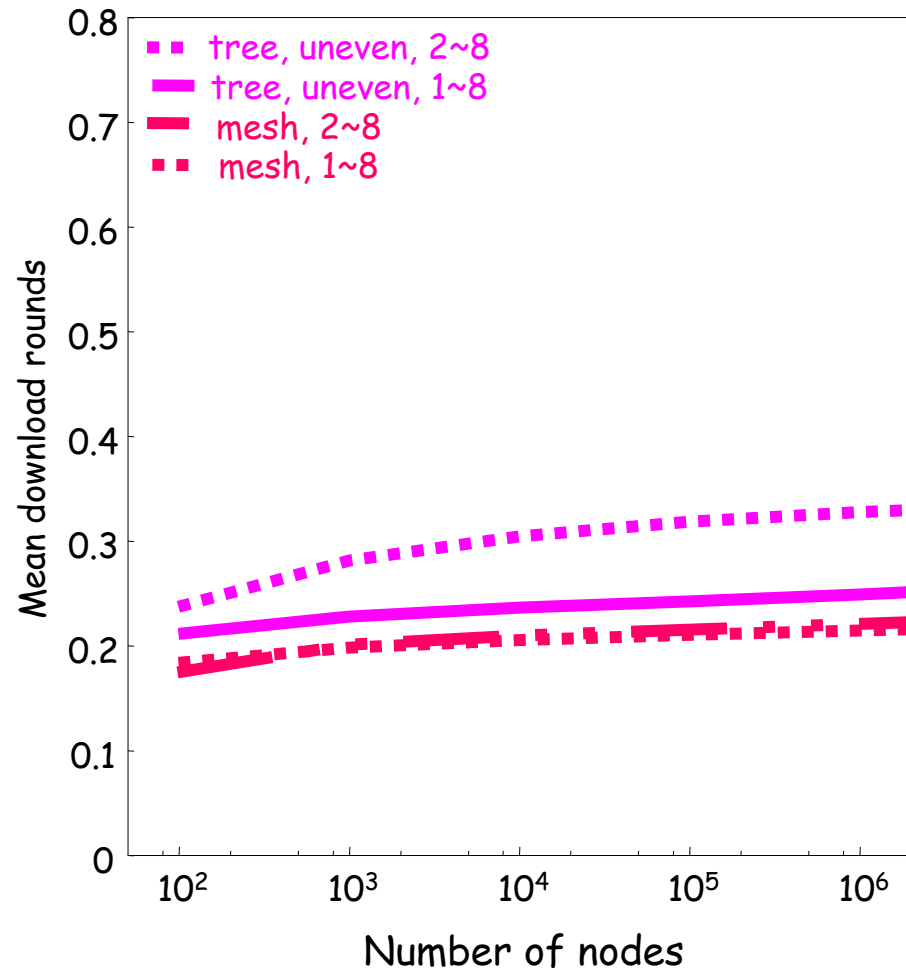
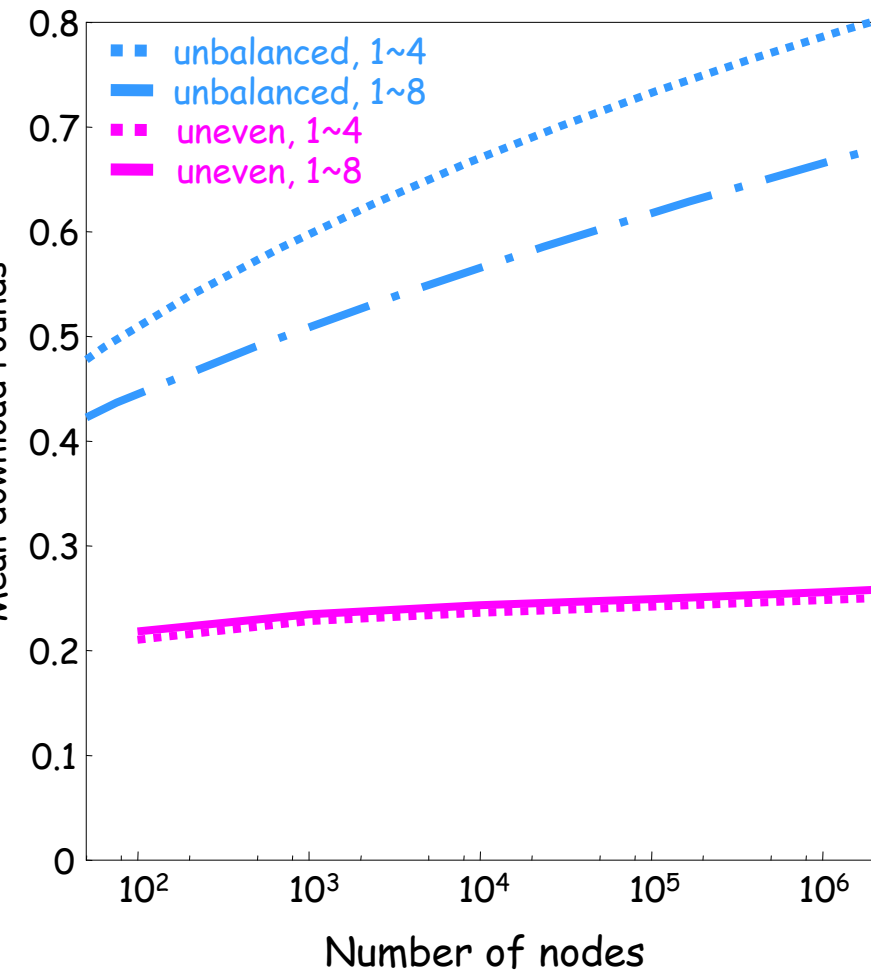


mesh: two
partially overlapped
diffusion subtrees
(b)

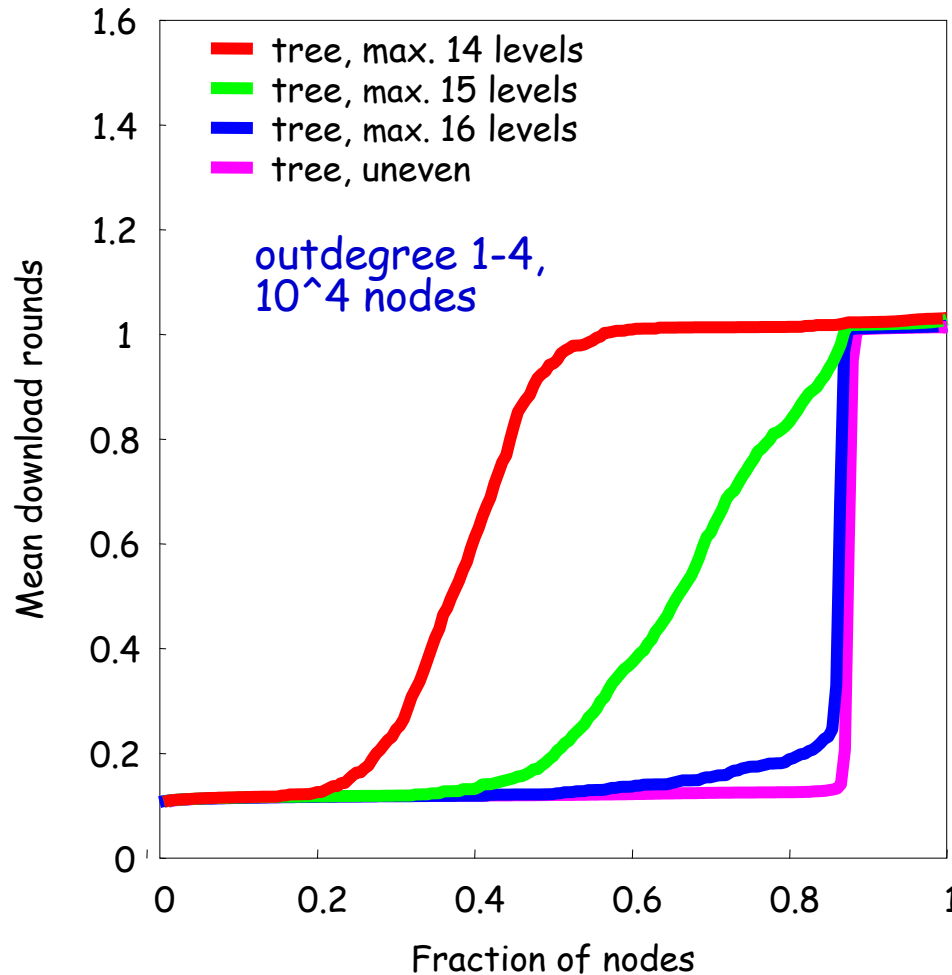


mesh: *constrained*
diffusion subtrees
(c)

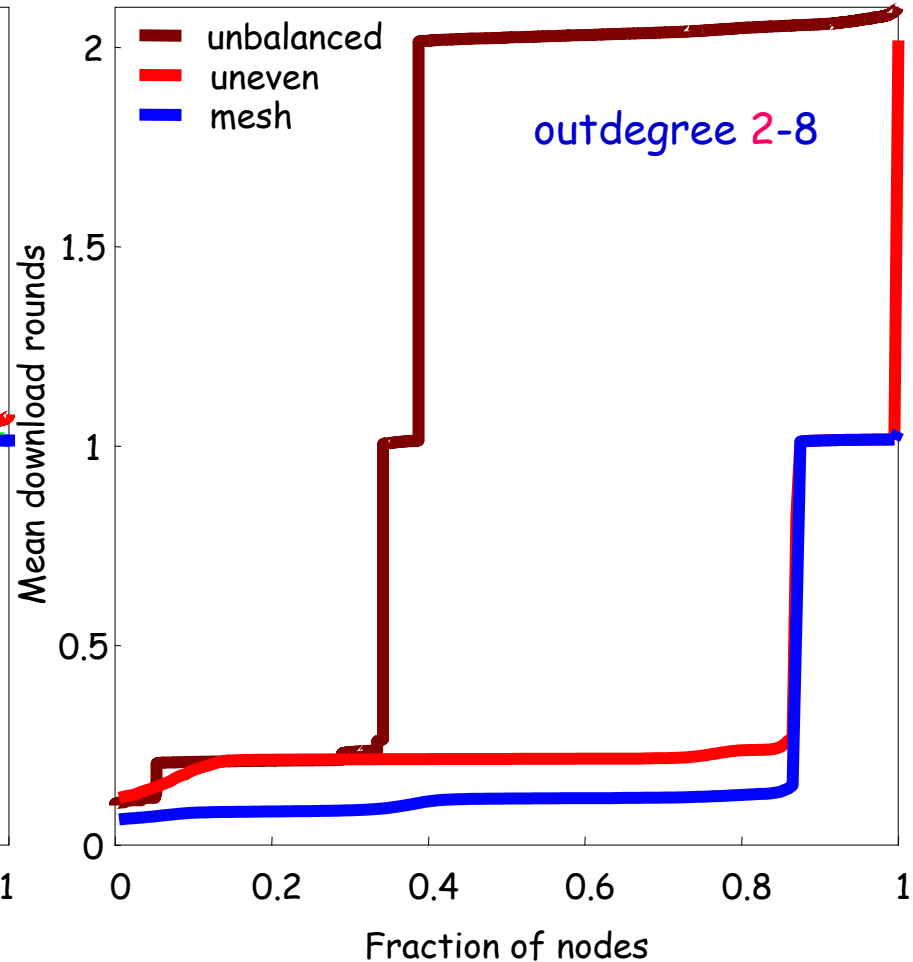
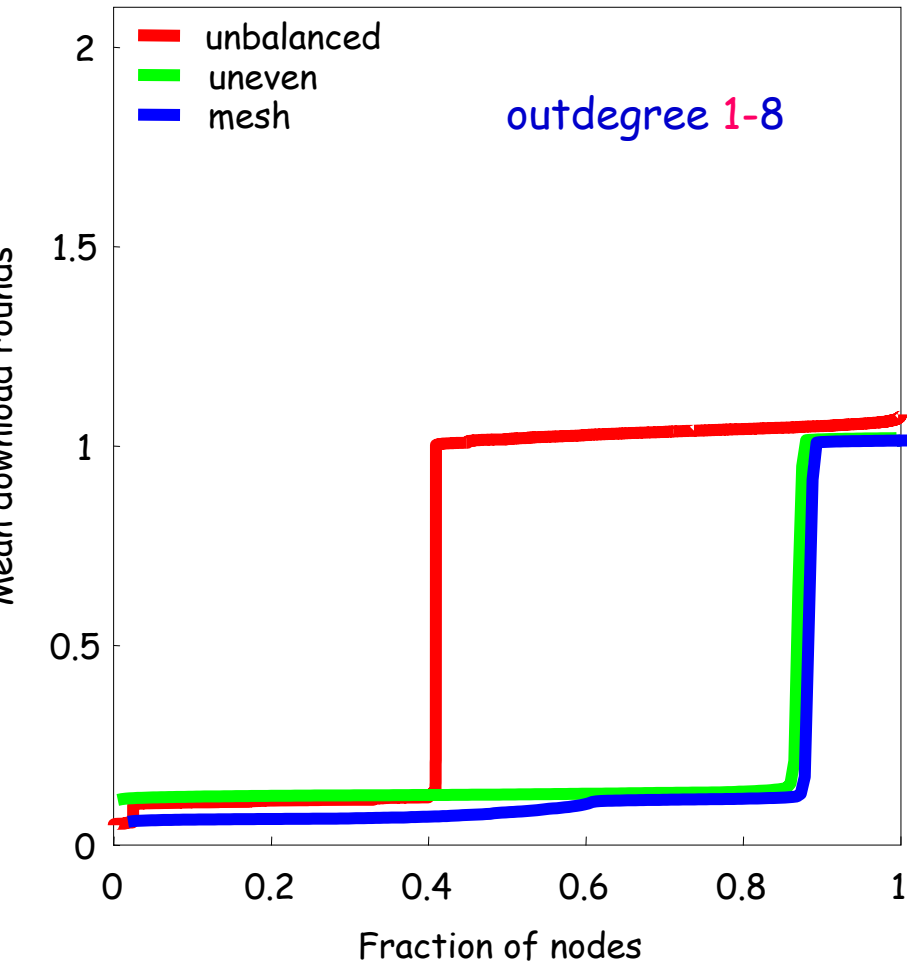
Trees (Unbalanced vs uneven) vs Mesh



Trees: From "unbalanced" to uneven



Download times for 10^5 nodes



Trees vs Mesh

			Wasted upload Bandwidth	
Outdegree	#peers	Levels	Uneven Tree	Mesh
1-8	10^5	21	46.9%	13.3%
1-8	10^6	24	47.5%	13.1%
2-8	10^5	14	66.2%	26.8%
2-8	10^6	18	68.9%	29.3%

What technique for what architecture

	Balanced tree	Unbalanced or Uneven tree	Ptree	Mesh
Analytical approach	Yes	No	<i>For fixed outdegree only</i>	No
Sample path method	Yes	Yes	Yes	TBD
Monte-Carlo simulation of full architecture	Yes	Yes	Yes	Yes

Conclusions

- Analysis of the performance in distributing a file
- Results for linear, tree, mesh distribution architectures
 - ◆ Not only mean download times but **full distribution**
- Analysis of "uneven" trees
- Interesting properties of dynamic systems with heterogeneous peers
 - ◆ A minimum outdegree of 1 can significantly improve performance
- Can also treat (not discussed)
 - ◆ The case of upload bw \neq download bw (e.g. ADSL)
 - ◆ Impact of free riders