Fundamental delay bounds in peer-to-peer chunk-based streaming systems

Giuseppe Bianchi
October 21, 2008

Joint work with the P2P research group in Roma Tor Vergata
N. Blefari Melazzi, L. Bracciale, F. Lo Piccolo, S. Salsano

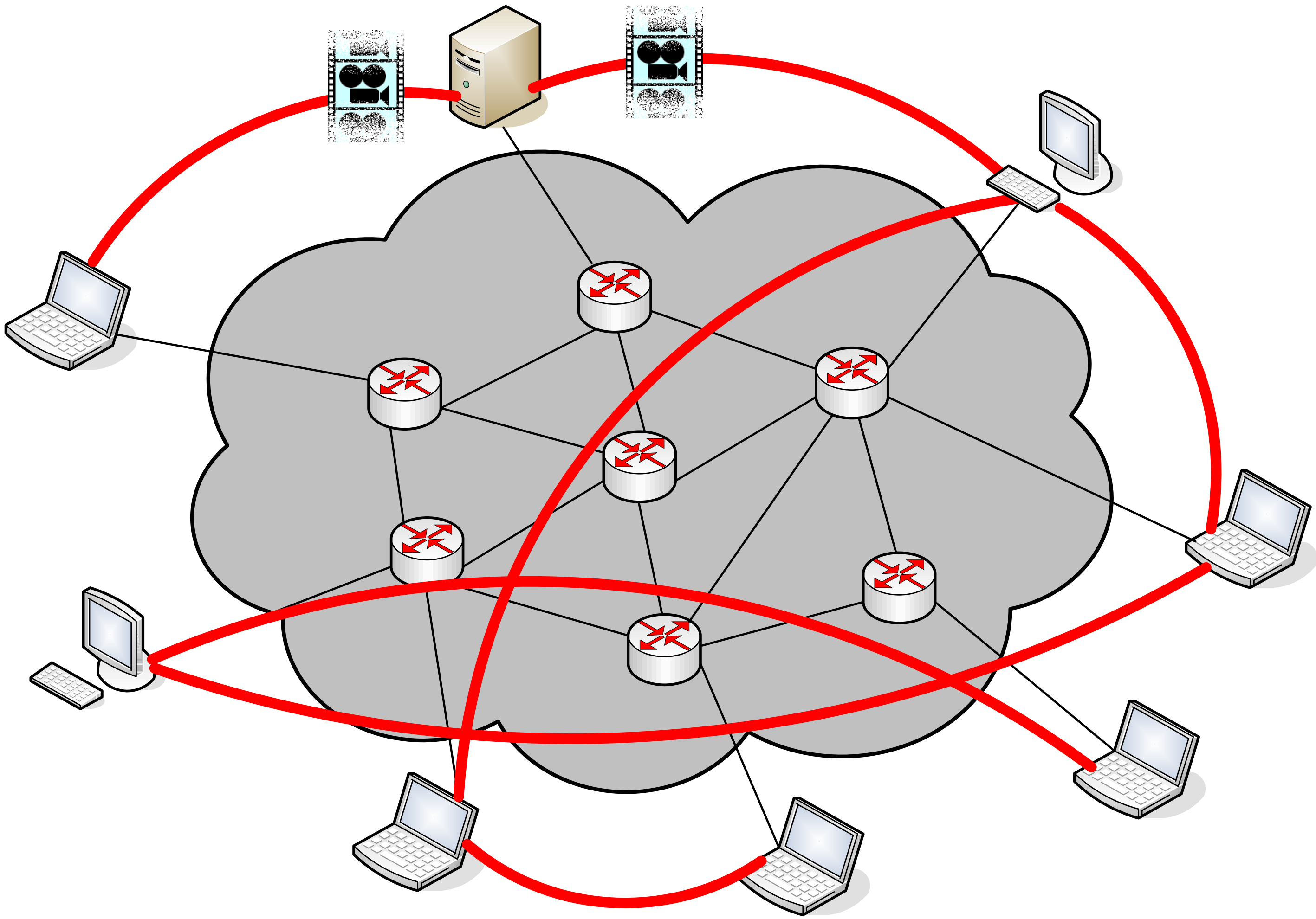
giuseppe.bianchi@uniroma2.it

Outline

- ➔ **P2P streaming in a nutshell**
- ➔ **Motivations and goals**
- ➔ **Constructive demonstration of the bounds**
- ➔ **The “tree intertwining” problem**
- ➔ **A theory-driven distribution algorithm**

What is P2P streaming?

- ➔ **P2P overlay operation for live streaming**
  - ⇨ P2PTV as new emerging trend



Deployments and numbers…

- **PPLive**
  - December 2005: more than 20 millions download
- **Gridmedia**
  - Adopted by CCTV (largest TV station in China) to broadcast Gala Evening for Spring Festival (Chinese New Year)
    - over 500.000 users attracted and 224.000 simultaneously online users in January 2006
- **Babelgum**
  - September 2007: “Babelgum Online Film Festival”
    - 7 categories of films, voting online viewers, jury of industry experts (chair: Spike Lee), winners awarded @ Cannes Film Festival
- **TVUPlayers**
  - Live Internet TV; 3.5 M monthly unique viewers in January 2008

Technical alternatives (rough)

➔Topology

- ⇨Trees explicitly maintained
  - →NICE, SplitStream, ...
- ⇨Mesh: no a priori established path; delivery driven by content availability
  - → CoolStreaming/DONET, Gridmedia, PRIME, PULSE, …

➔Data selection

- ⇨Push: sender decides
  - →E.g., all tree-based system
- ⇨Pull: receiver-driven
  - →E.g., CoolStreaming, PRIME
- ⇨Hybrid Pull-Push
  - →E.g., Gridmedia

Outline

- ➔ P2P streaming in a nutshell
- ➔ **Motivations and goals**
- ➔ Constructive demonstration of the bounds
- ➔ The "tree intertwining" problem
- ➔ A theory-driven distribution algorithm

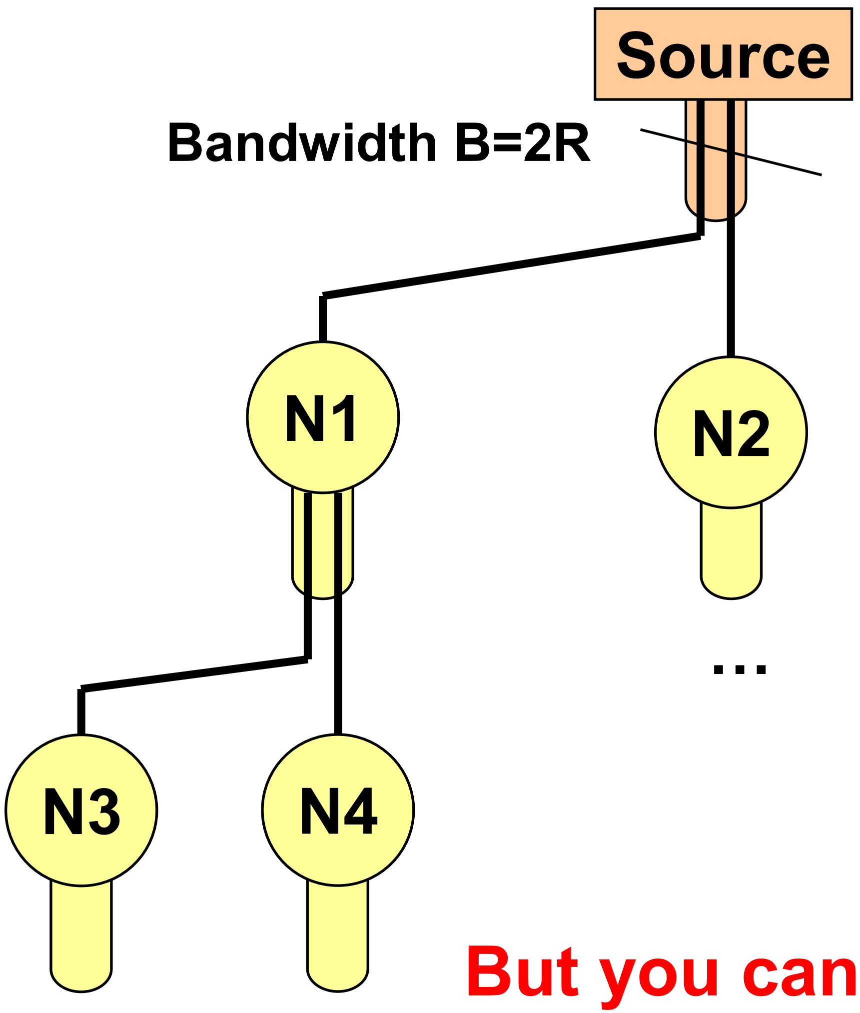
Our problem

What is the <u>minimum</u> delay achievable in a large scale <u>chunk-based</u> P2P streaming system?

And what are the best topologies emerging as a consequence?

Well…
This looks like an "usual" path cost optimization problem…
… but it is NOT, and it come out to be a NEW problem. Why?

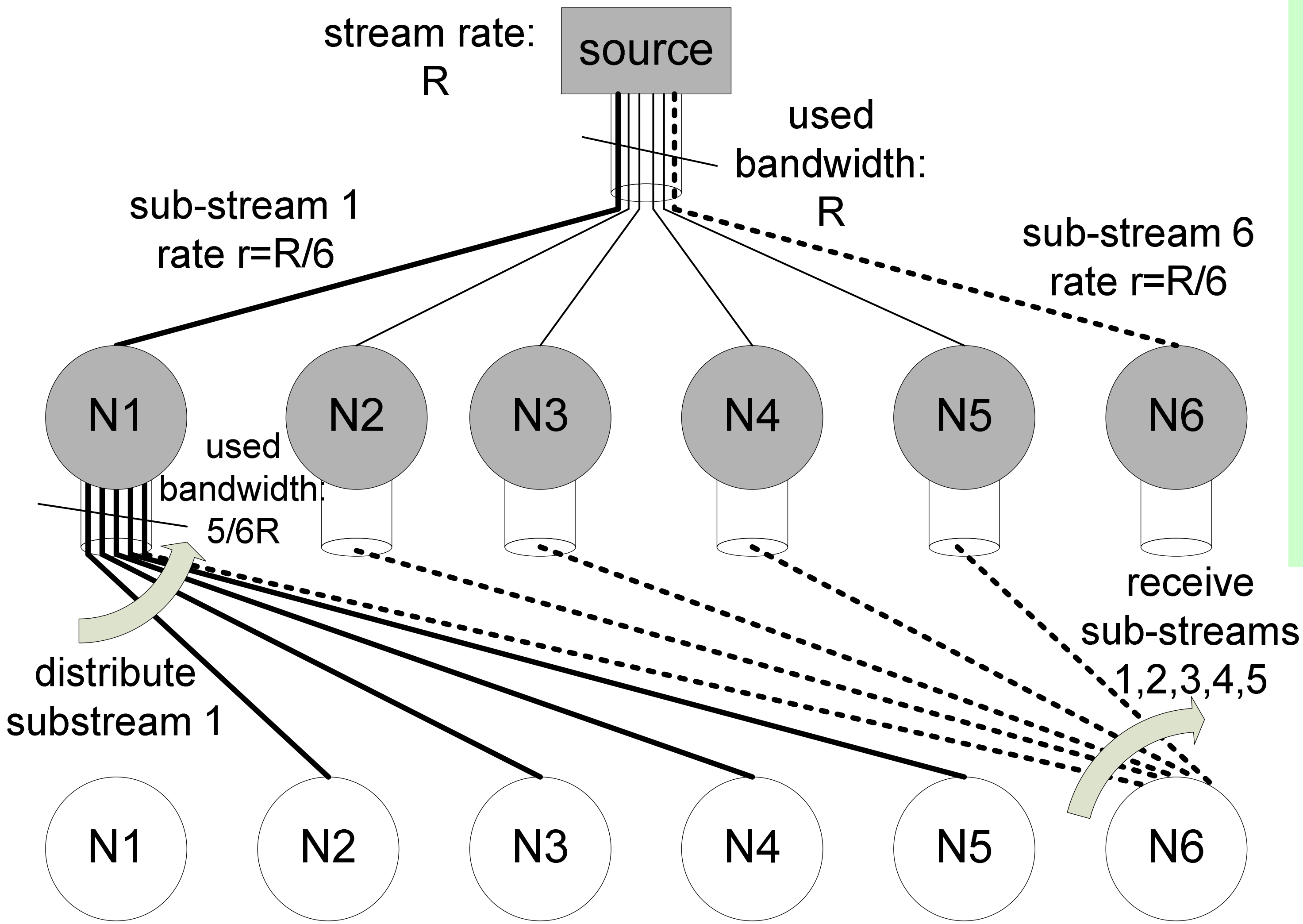
Non chunk-based systems



- **No chunks:**
  - Information continuously delivered
  - Small size IP packets is reasonable approx
- **Apparently a multicast tree problem**
  - Assign delays to each overlay path
  - Find minimum delay tree
  - Fanout depends on B/R ratio
- **Homogeneous delays → minimum depth tree**

But you can do A LOT better than this…

Exploiting sub-streams



Min tree depth = 2
(well known)

If B>=R, perf. not affected
(more general cond. were studied)

If limit m on children, perf. depend on $\log_m N$

ALL Well Known & SOLVED. SO??

Why chunk-based systems differ?

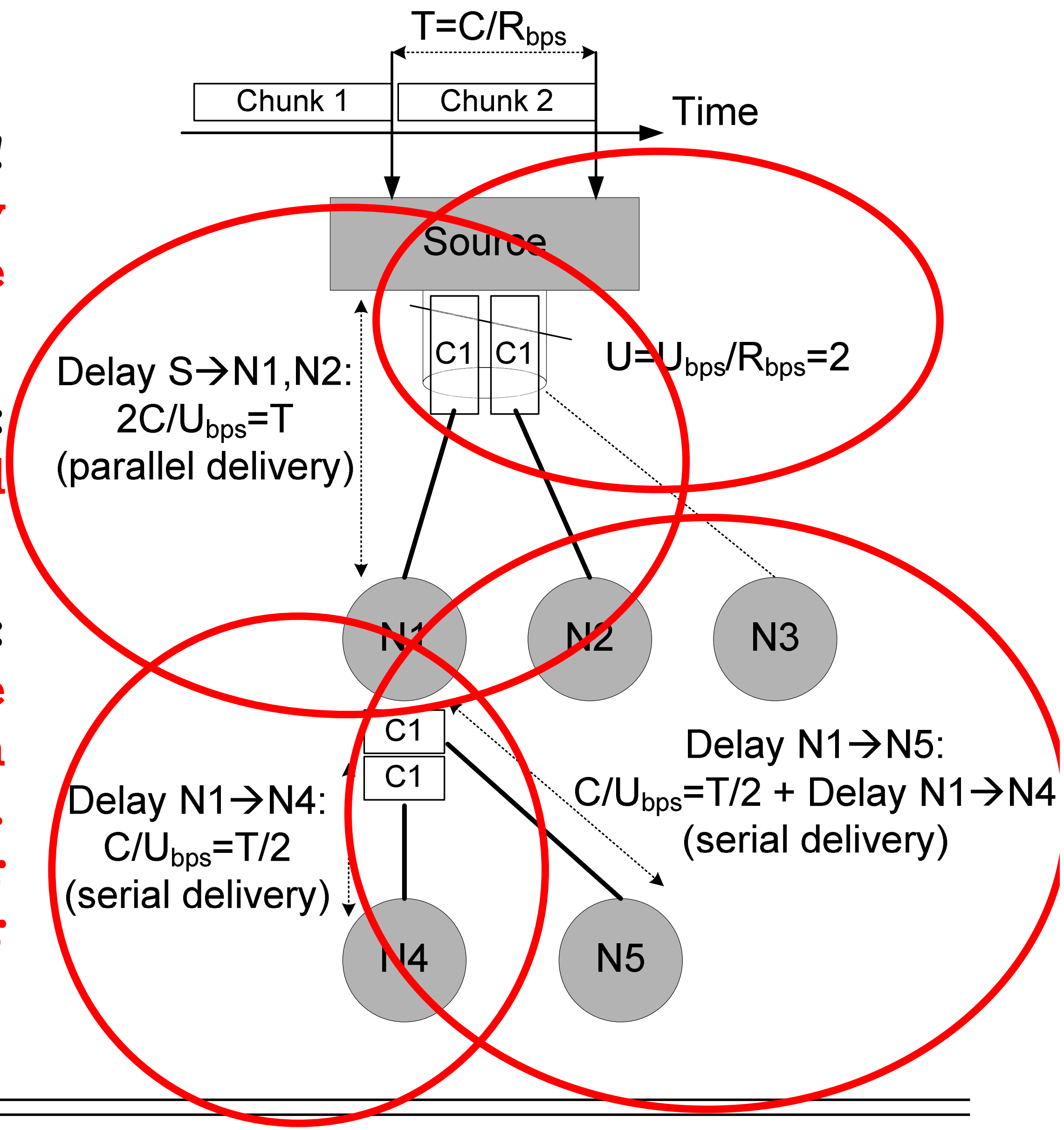
- ➔ **Chunk size >> IP packet size**
  - ⇨ 500 kb in CoolStreaming
  - ⇨ Chunk = "Atomic" transmission unit → store&forward!
- ➔ **Delay performance mostly depends on chunk transmission time**
  - ⇨ Chunk tx time much greater than overlay link delay
  - ⇨ Exactly the opposite of sub-stream-based models (tx time negligible)
- ➔ **Overall delay optimization problem is radically different!!!**
  - ⇨ You CANNOT model this as a path delay problem – see why in next slide

Why not a minimum path cost problem…

1) Bandwidth matters!
 # children constrained by stream rate and the available uplink bandwidth

2) No "per-hop" delay:
uplink bandwidth shared by multiple overlay links

3) Extra sources of delay:
delivery delay may include components other than the transmission time (e.g. time spent by supplier node in serving other nodes)



Our contribution

- ➔ **Theoretical formalization and understanding of chunk-based systems' delay performance**
  - ⇨ No prior literature (to the best of our knowledge)
    - → Perhaps the fundamental difference brought by chunk-based systems not properly captured?
- ➔ **Fundamental bound derivation**
  - ⇨ For homogeneous bandwidth nodes
  - ⇨ Does NOT start from the assumption of a topology or scheduling, and its consequent optimization
    - → Although it will be presented later on in a constructive way
- ➔ **Bound reachability**
  - ⇨ Which is the topology and chunk scheduling that allows to reach such bound
- ➔ **From theory to practice**
  - ⇨ How to design a practical P2P streaming system which takes advantage of the lessons learned from this theory

Outline

- ➔ P2P streaming in a nutshell
- ➔ Motivations and goals
- ➔ **Constructive demonstration of the bounds**
- ➔ The “tree intertwining” problem
- ➔ A theory-driven distribution algorithm

Performance metric

- ➔**Absolute Network Delay:**
  - ⇨ Worst-case delay experienced across all chunks and all network nodes

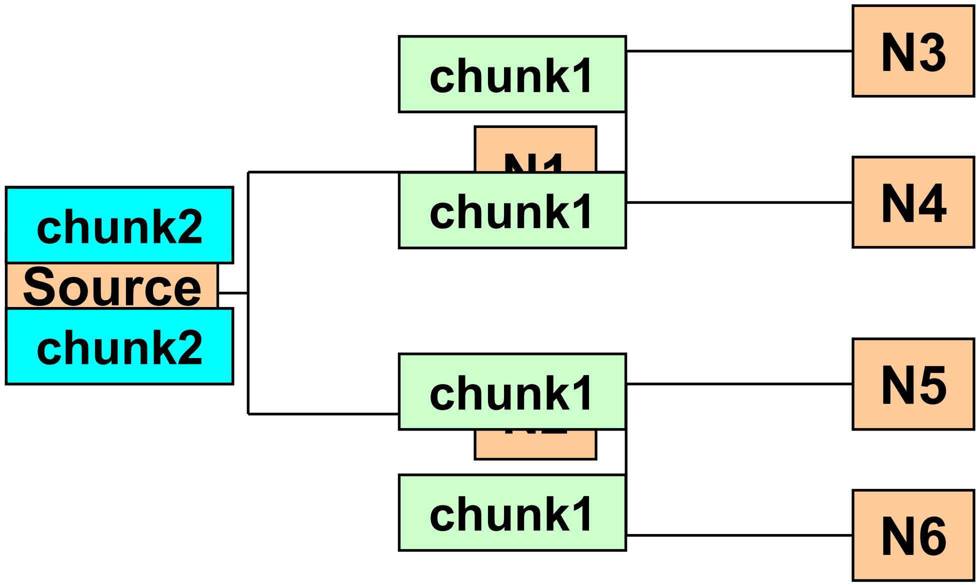
But it is not a convenient metric to deal with, hence:

- ➔**Stream Diffusion Metric N(t)**
  - ⇨ Number of nodes that receive chunks within time t
  - ⇨ "dual metric" with respect to absolute delay
    - →By maximizing N(t), Absolute delay is minimized

May handle infinite network sizes
Convenient asymptotic expressions

Balanced tree

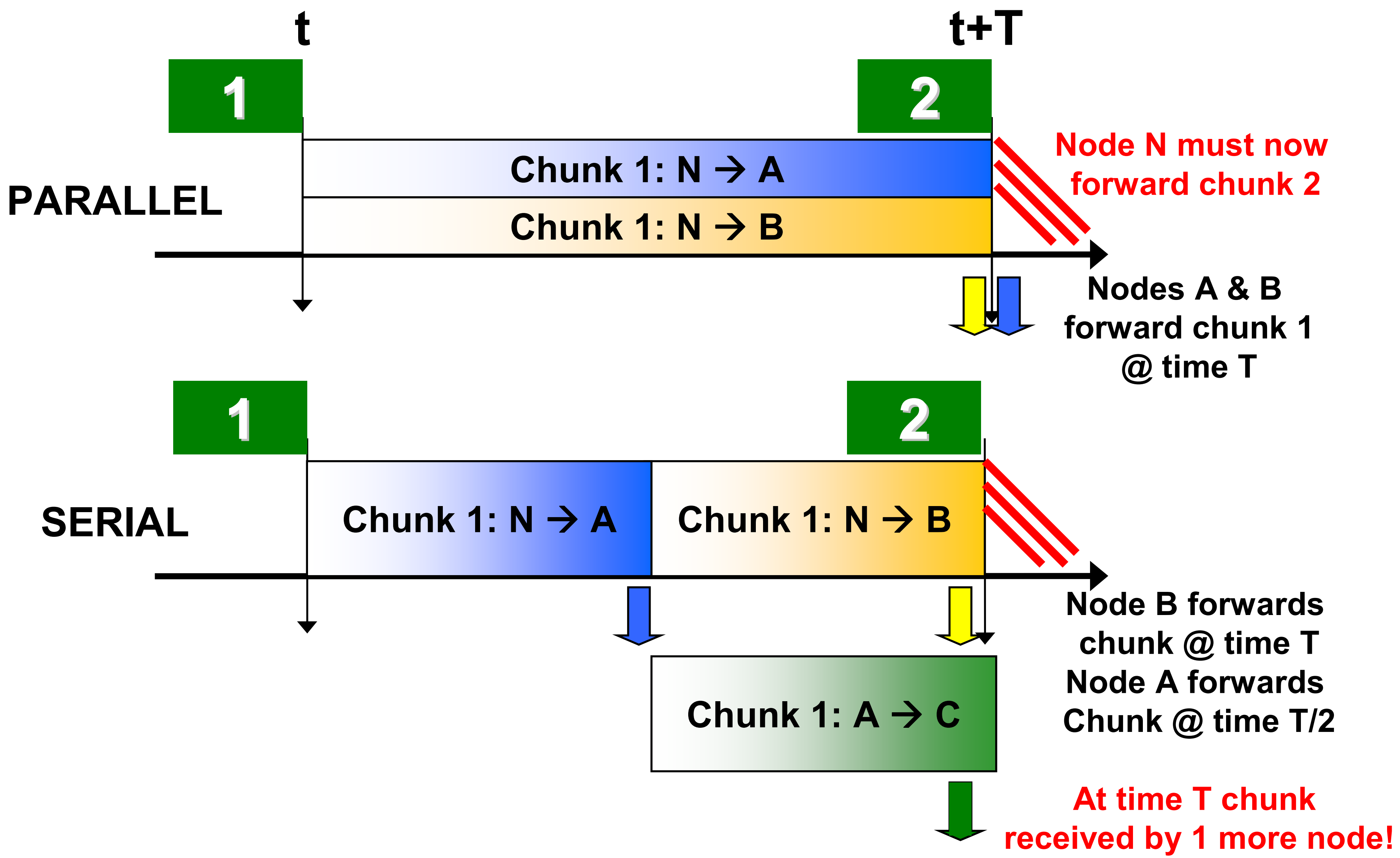
Tree fanout = B/R=U (=2 in the example)
Time unit: T = C/R (chunk interarrival time)



After 1 time unit	2 new nodes
After 2 time unit	4 new nodes + 2 prior nodes
After 3 time units	8 new nodes + 6 prior nodes
After t time units	$N(t) = 2^{t+1}-2$ → if network has 30 nodes, delay=4T

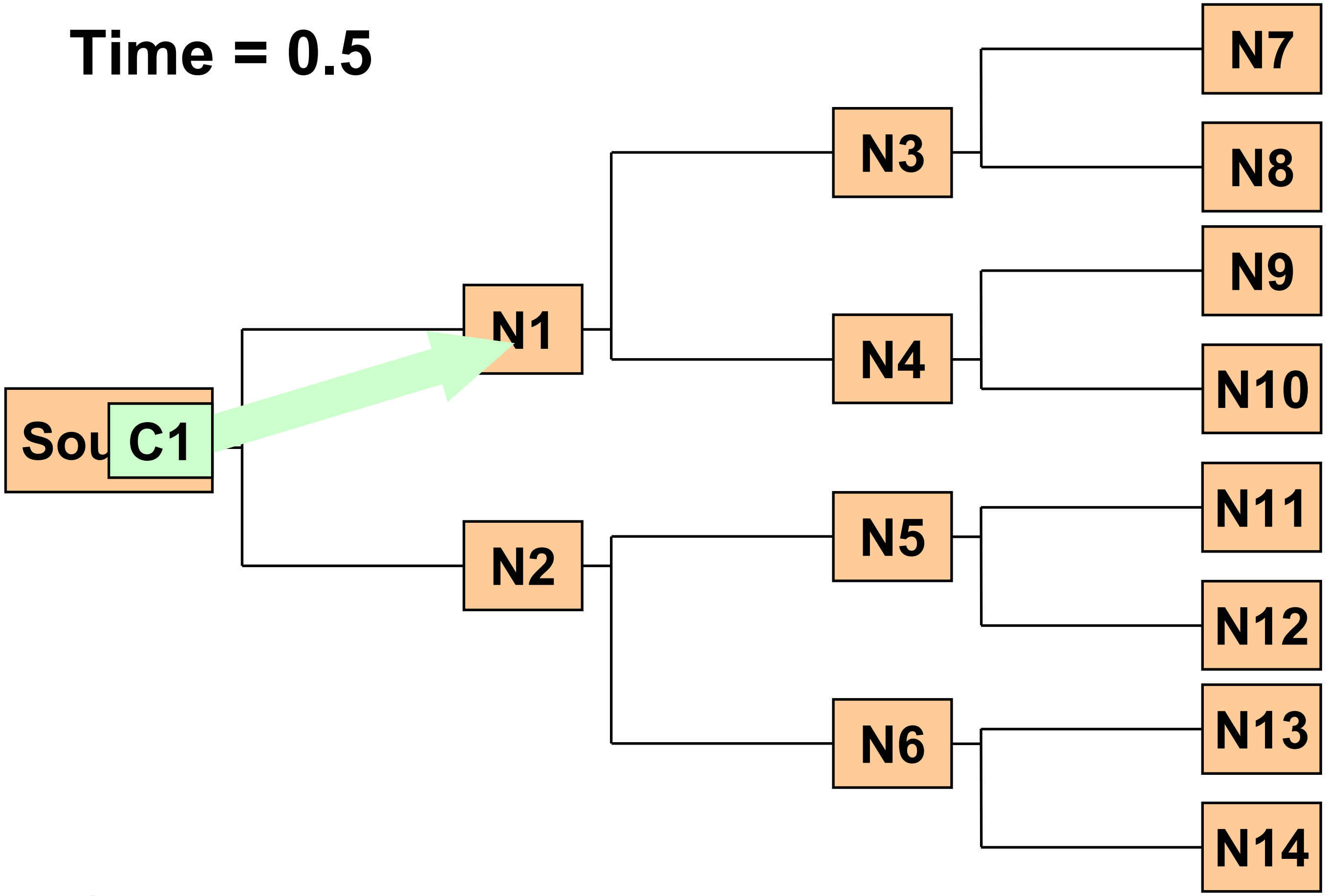
Can we do better???

Benefits of Serial transmission



Distribution through serialization

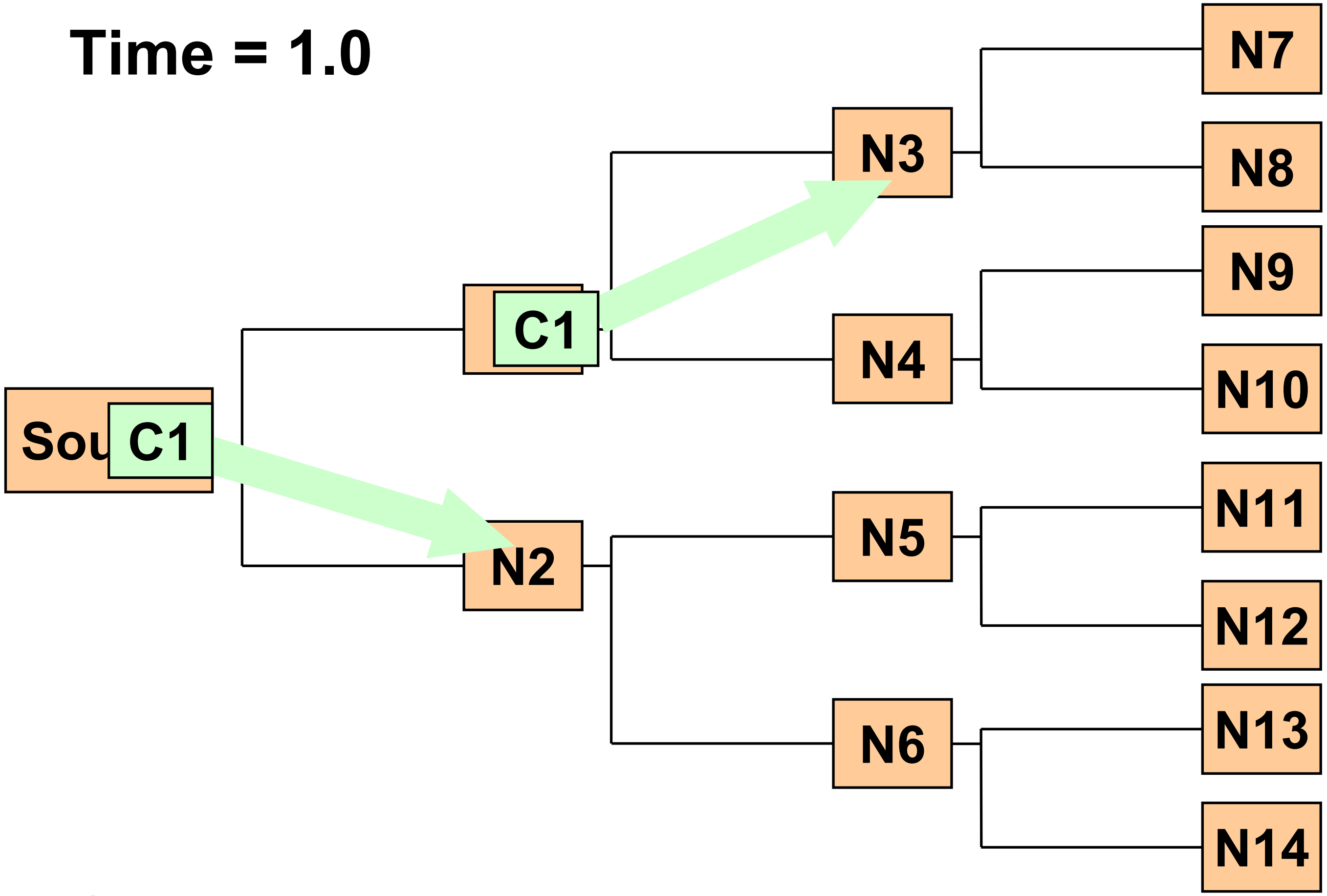
Time = 0.5



C1 at 1 node

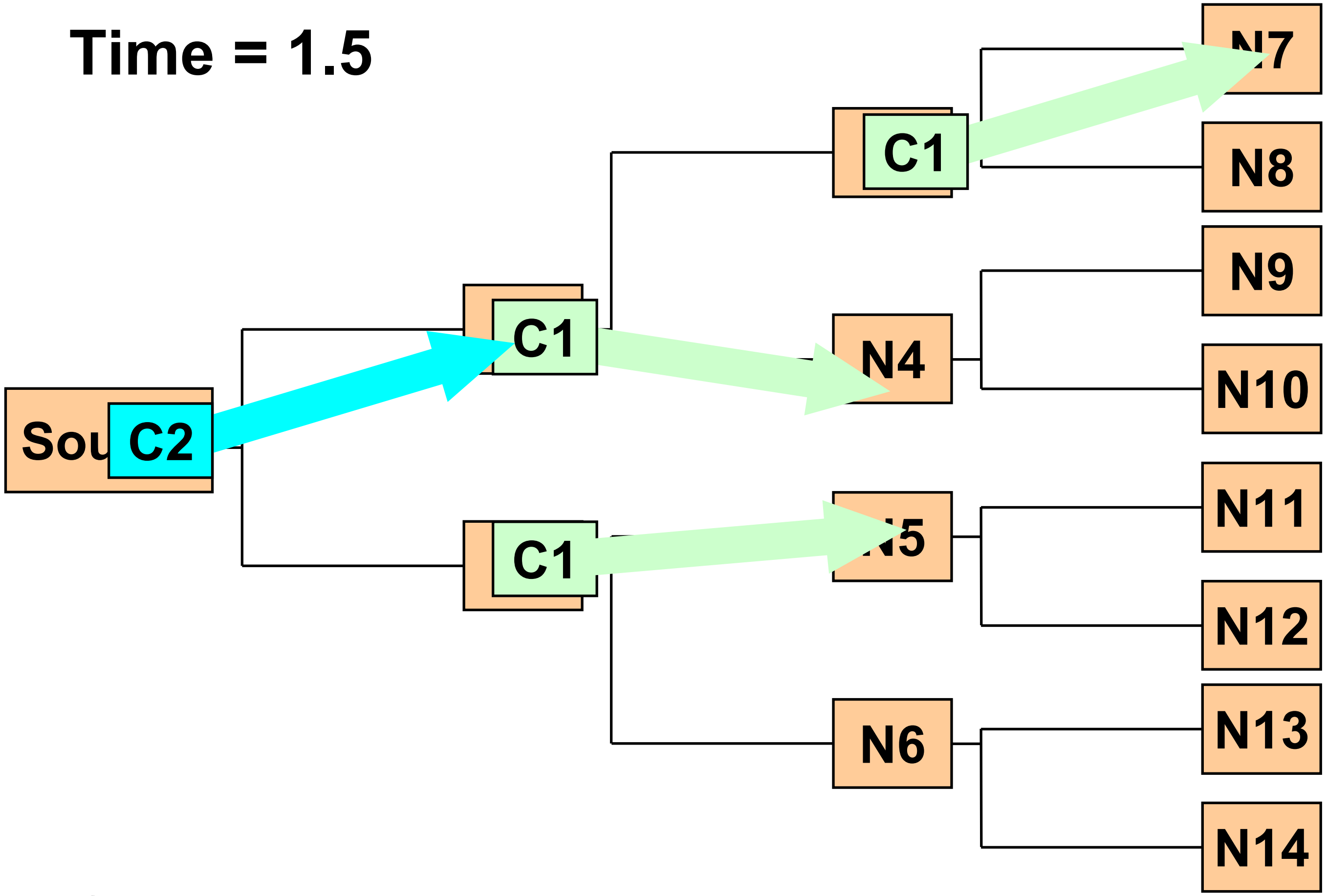
Distribution through serialization

Time = 1.0



C1 at 1 + 2 nodes

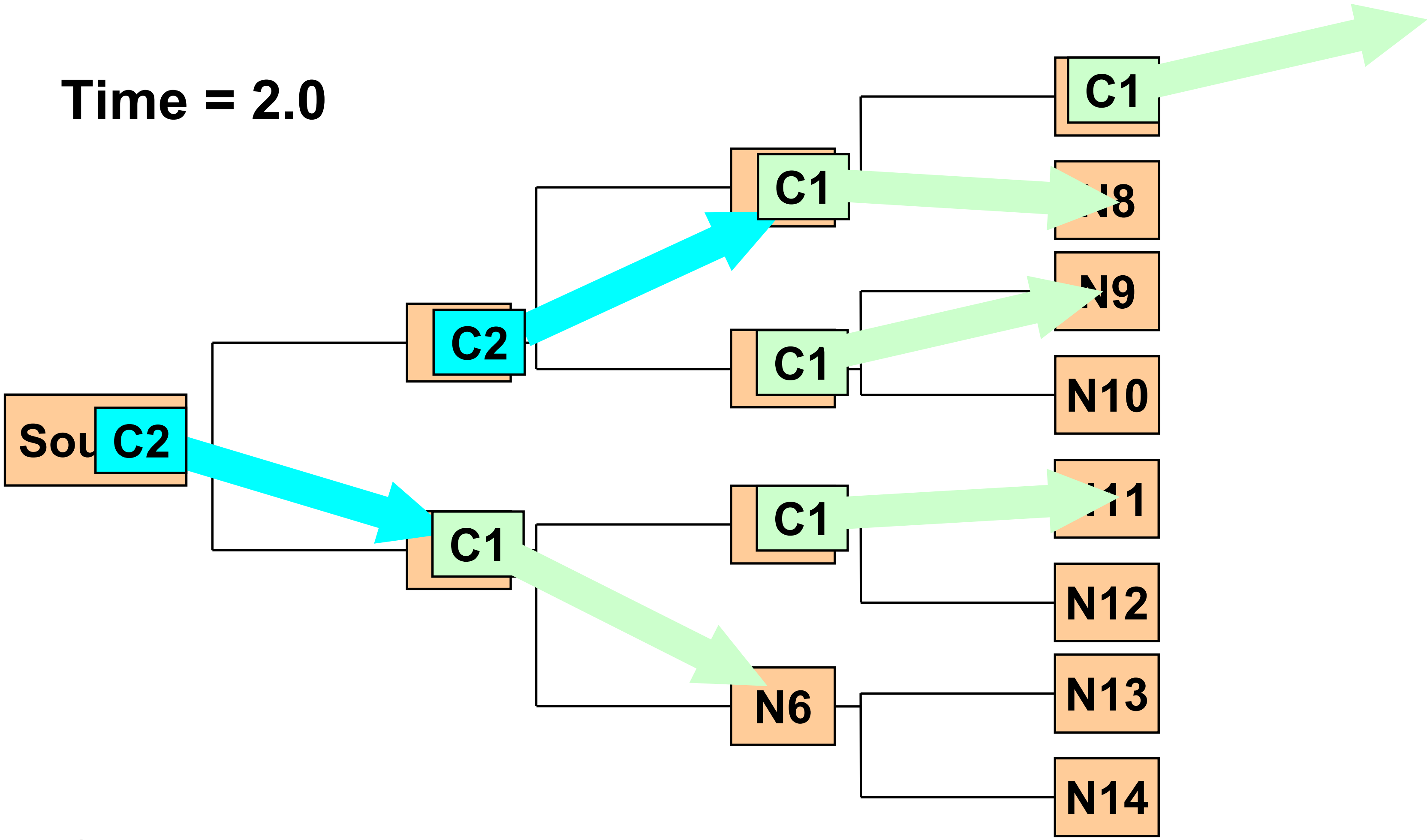
Distribution through serialization



C1 at 1 + 2 + 3 nodes

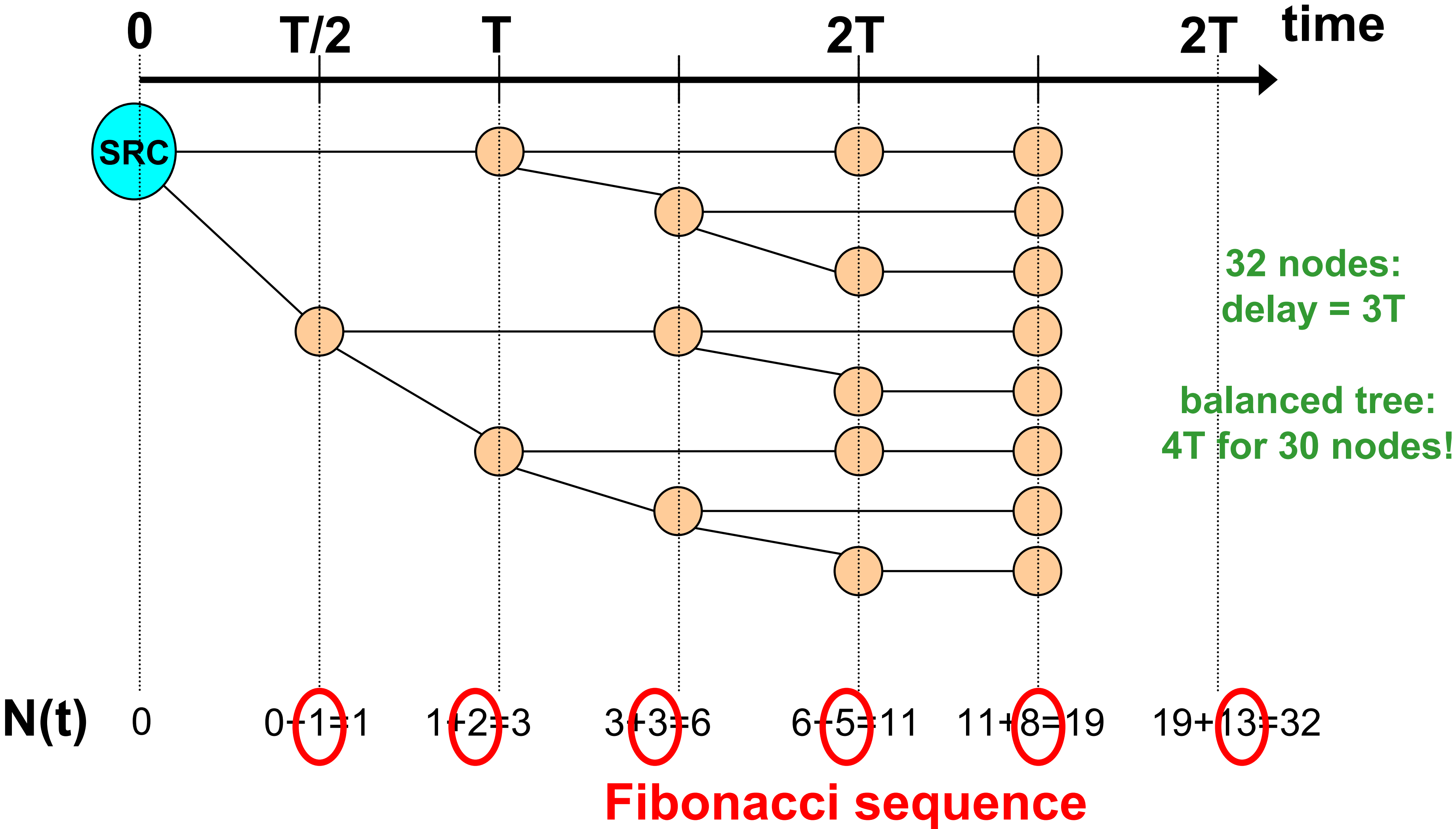
Distribution through serialization

Time = 2.0

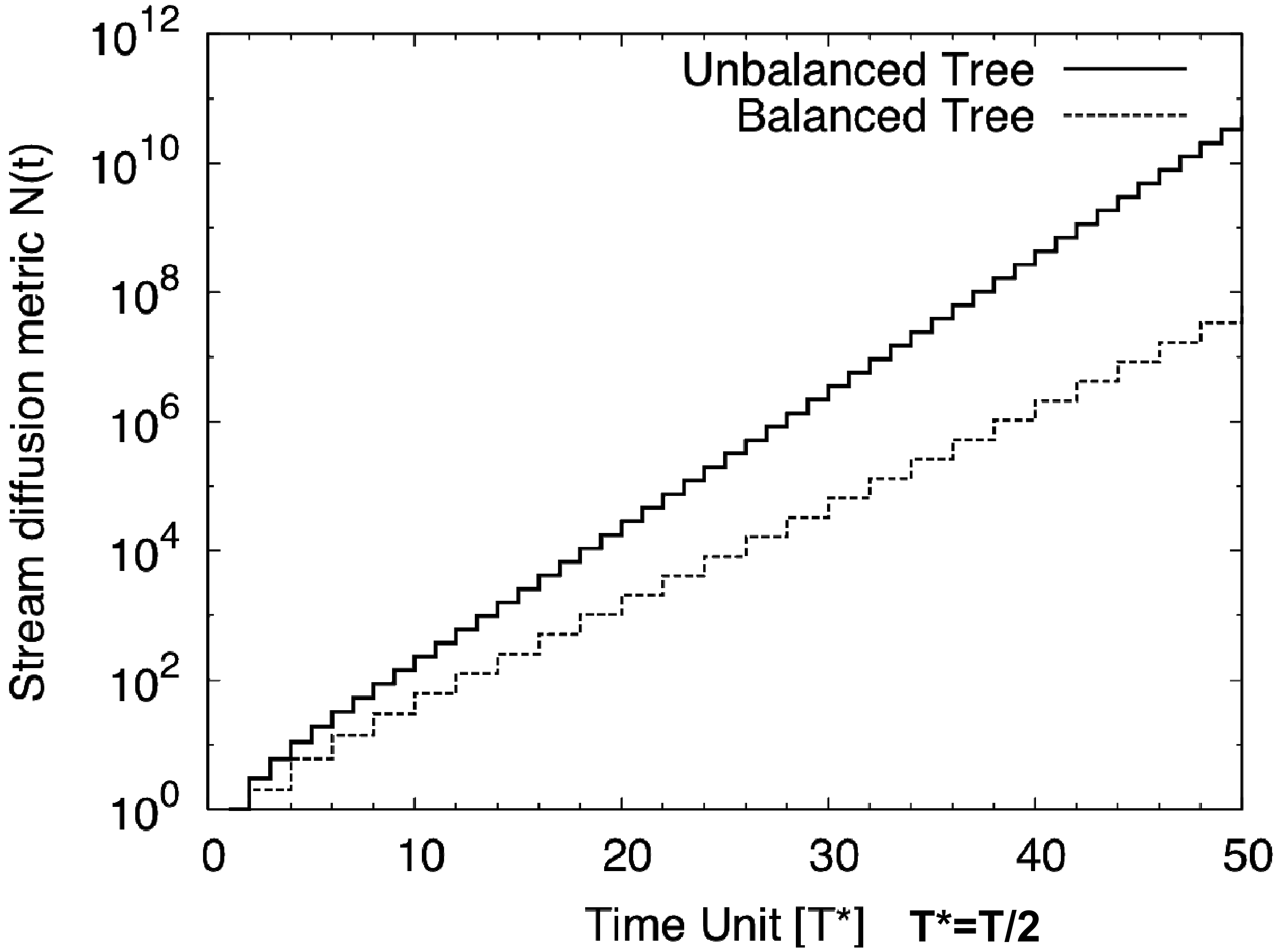


C1 at 1 + 2 + 3 + 5 nodes

Result: unbalanced tree!

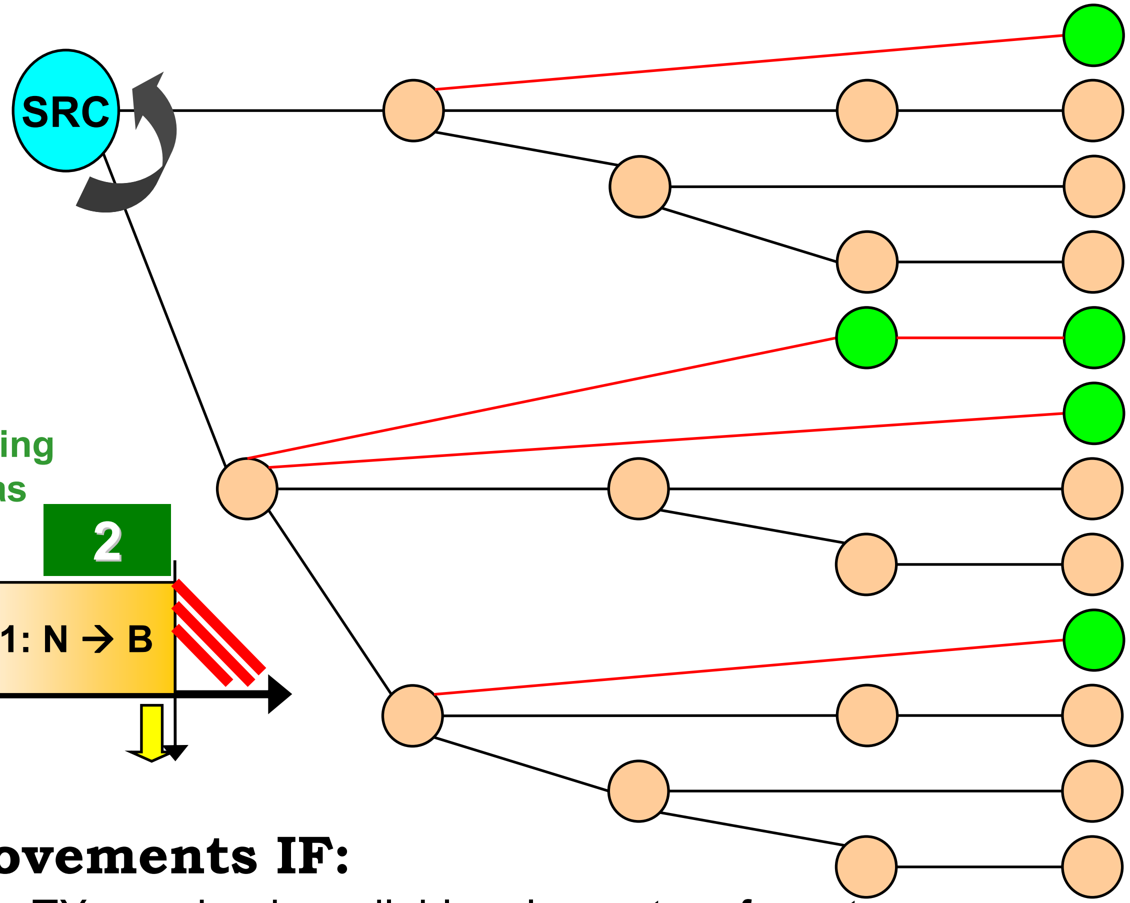


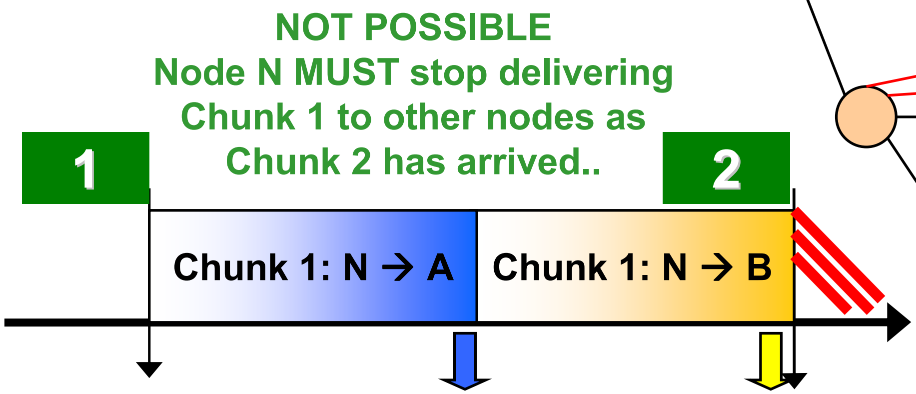
Balanced vs Unbalanced tree



T*=T/2

Can we do better?

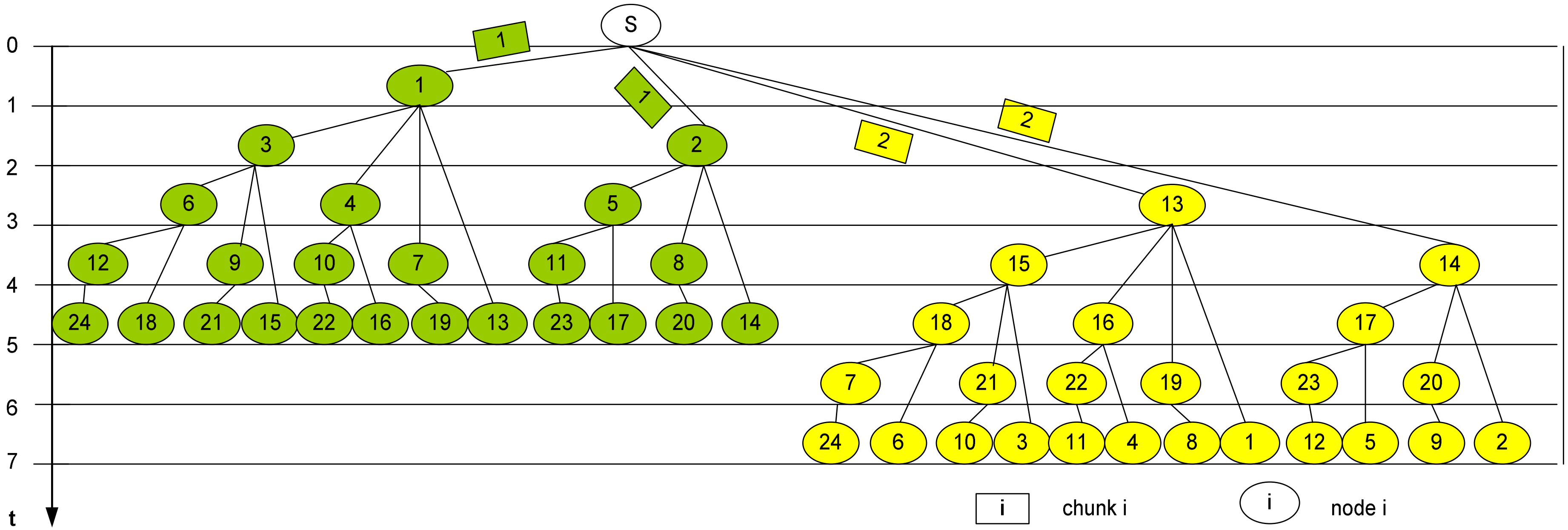




➔ **Performance improvements IF:**

- ⇨ More bandwidth =more TX per chunk available = larger tree fanout
  - →obvious
- ⇨ More time between consecutive chunks…
  - →How? **MULTIPLE TREES!**

Unbalanced Forest Topology (B = 2R case)



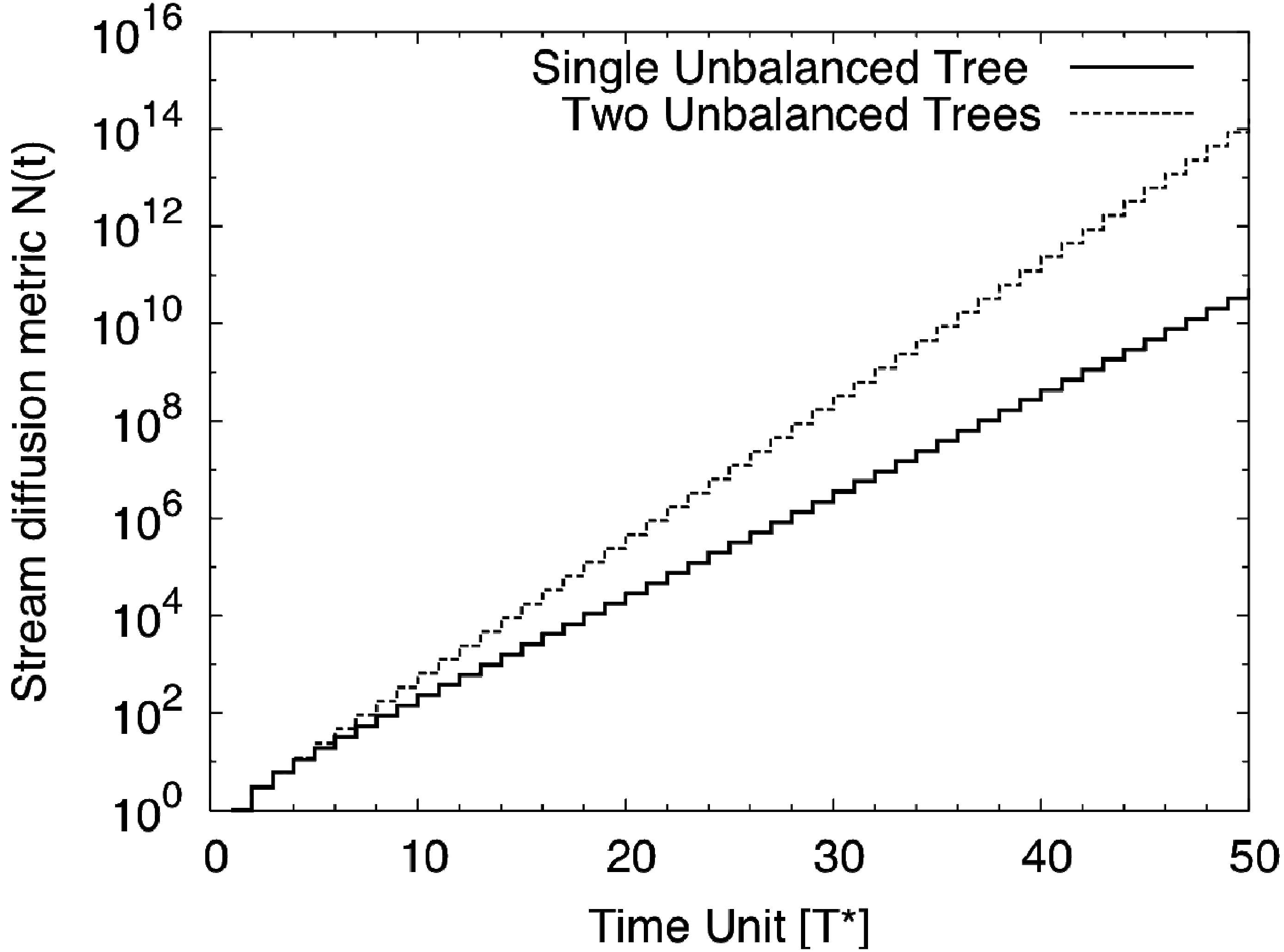
➔ **Each node receives all the chunks**

➔ **But each node uploads only half of the chunks**

→ **the other half only if it can…**

➔ **Nodes have more time (two times!) to upload chunks**

Unbalanced tree vs two Unbalanced trees



Can we do better?

NO (for same U and k)!

→ **Fundamental theorem proven - no assumption on topology:**

- ⇨ Given bandwidth B = U x R
  - → U multiple of stream rate, U=1 OK
- ⇨ Given max number of children k a node may serve
  - → k=multiple of U (non multiple would waste tx opportunities)
  - → k=infinity OK
- ⇨ Using integer time t with unit C/B (min tx time for a chunk)

$$\overline{N}(t) = \sum_{j=1}^{U} S_k(t-j+1)$$

where

K-step fibonacci sum

$$S_k(n) = \sum_{i=1}^{n} F_k(i) \qquad n > 0$$

K-step fibonacci seq.

$$F_k(i) = \begin{cases} 0 & i \le 0 \\ 1 & i = 0 \\ \sum_{j=1}^{k} F_k(i-j) & i > 0 \end{cases}$$

Closed form expressions

constant　　　　small

$$\overline{N}(t) \approx \frac{\phi_k^2\left(1-\phi_k^{-U}\right)}{Q_k(\phi_k)(\phi_k-1)^2}\phi_k^t - \frac{U}{k-1}$$

$$\overline{N}_{k=\infty}(t) = 2^t\left(1-2^{-U}\right)$$

$$\phi_2 = 1.61803 \quad Q_2(\phi_2) = 2.23607$$
$$\phi_3 = 1.83929 \quad Q_3(\phi_3) = 2.97417$$
$$\phi_4 = 1.92756 \quad Q_4(\phi_4) = 3.40352$$
$$\phi_5 = 1.96595 \quad Q_5(\phi_5) = 3.65468$$
$$\phi_6 = 1.98358 \quad Q_6(\phi_6) = 3.80162$$
$$\phi_\infty = 2 \quad Q_\infty(\phi_\infty) = 4$$

Some Fibonacci math

new results on k-step Fibonacci sums were necessary

Recursive expression

$$S_k(n) = 1 + \sum_{i=1}^{k} S_k(n-i)$$

Direct expression versus Fibonacci k-step sequence

$$S_k(n) = \frac{\sum_{i=1}^{k} (i+1-k) F_k(n+i)}{k-1} - \frac{1}{k-1}$$

Binet-like (exact) Expression – complex numbers

$$S_k(n) = \sum_{i=1}^{k} \frac{\phi_{k,j}}{(\phi_{k,j} - 1) Q_k(\phi_{k,j})} \phi_{k,j}^n - \frac{1}{k-1}$$

Approximate Expression (only real root)

$$S_k(n) \approx \frac{\phi_k}{(\phi_k - 1) Q_k(\phi_k)} \phi_k^n - \frac{1}{k-1}$$

Improvements with forest size
(number of parallel unbalanced trees)

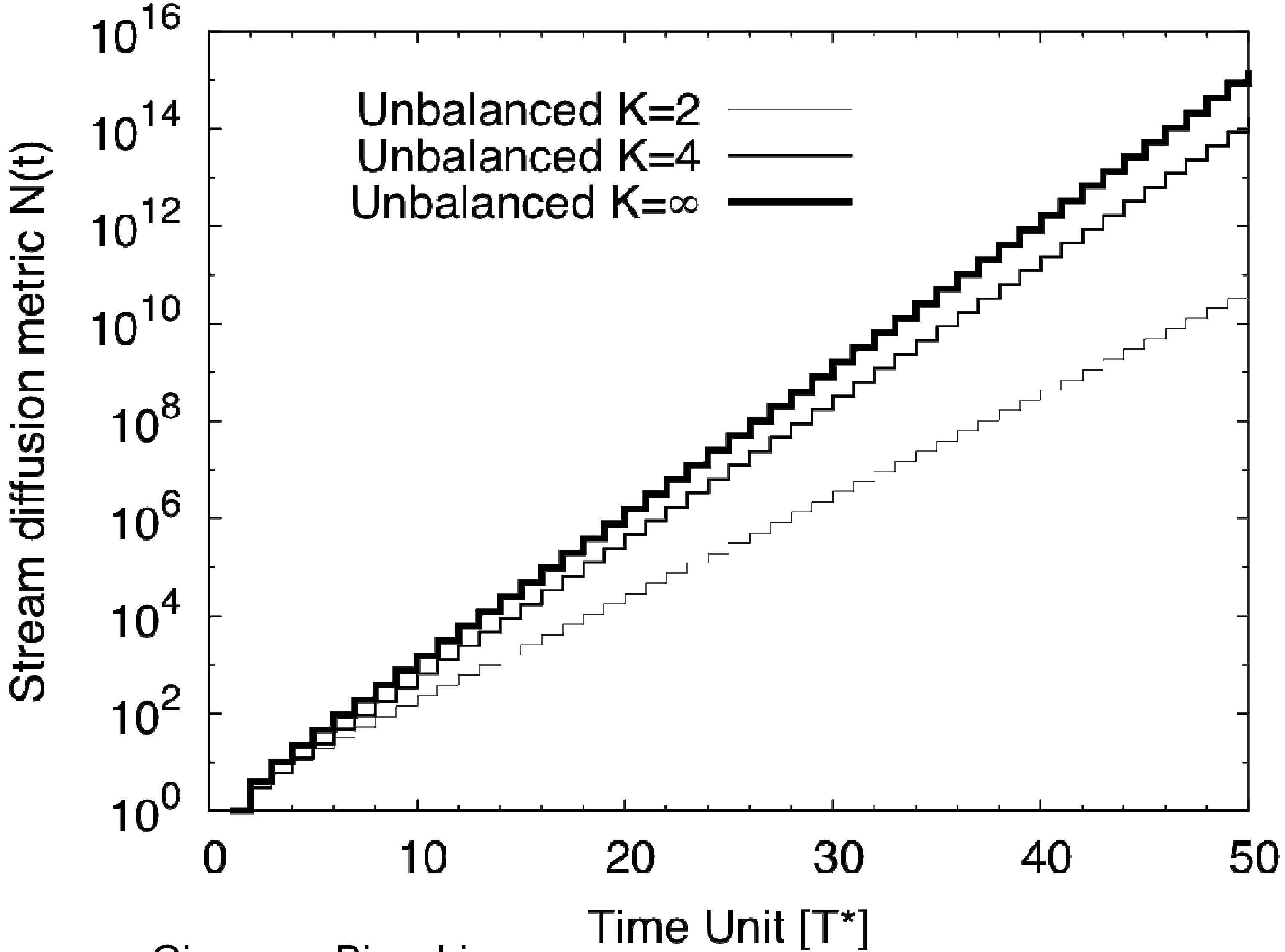
- **More trees, better performance**
  - Each node has more time to deliver chunks
  - Fibonacci "memory" k increases
  - Exponent in bound increases $N(t) \propto \varphi_k^{Ut}$
- **But…**
  - Fibonacci constants rapidly converge to 2
  - For k=4, we are already VERY close to 2
- **Good!**
  - Because more trees → more complexity!

$$\phi_2 = 1.61803$$
$$\phi_3 = 1.83929$$
$$\phi_4 = 1.92756$$
$$\phi_5 = 1.96595$$
$$\phi_6 = 1.98358$$
$$\phi_\infty = 2$$

Comparison /1
U=2

t	0	0.5	1	1.5	2	2.5	3	…	50
Balanced Tree	0	0	2	2	6	6	14		$2.25\ 10^{15}$
Serial tree	0	1	3	6	11	19	32		$1.5\ 10^{21}$
Serialize forest (two trees)	0	1	3	6	12	24	47		$2{,}93\ 10^{28}$
Bound (k=infty)	0	1	3	6	12	24	48		$9.5\ 10^{29}$

Comparison /2
U=2



Outline

- ➔ P2P streaming in a nutshell
- ➔ Motivations and goals
- ➔ Constructive demonstration of the bounds
- ➔ **The “tree intertwining” problem**
- ➔ A theory-driven distribution algorithm

Can we reach the bound???

➔ **Network nodes can always be arranged into ONE tree, as obvious**

➔ **But they might NOT be arranged into trees as REQUIRED by the bound**

Is this a problem, and where is the problem? See next…

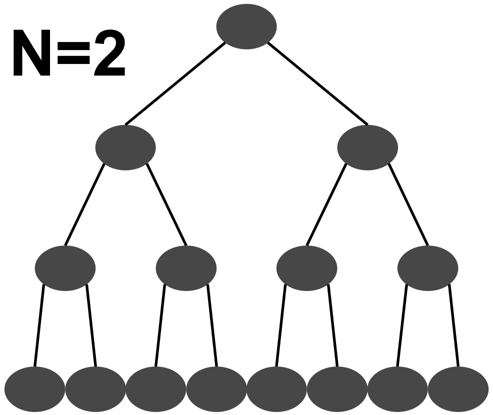
Can it be solved? Yes! … *but <u>very hard</u> to find a proof*

Property of N-ary balanced trees

$$\#_{LEAVES} = 1 + (N-1) \times \#_{INTERIOR}$$

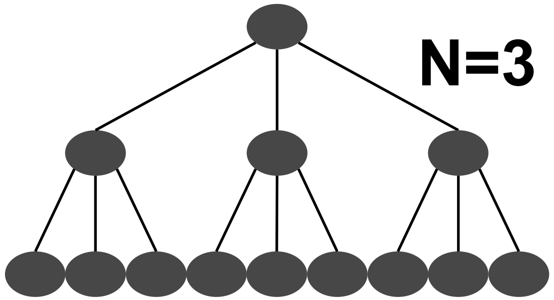
(for any tree depth)

N=2



8 leaves = 1 + (2-1) x 7 interior

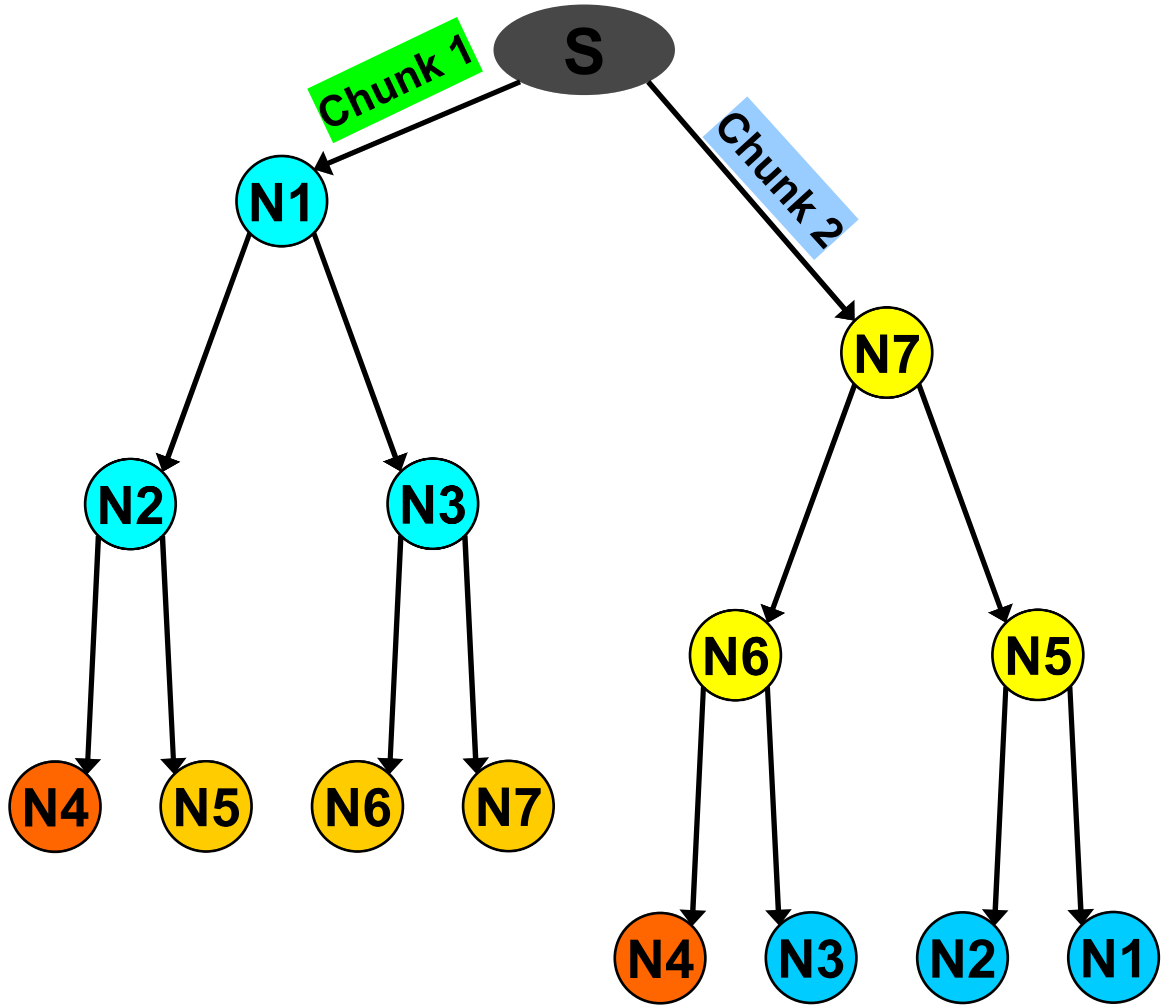
N=3



9 leaves = 1 + (3-1) x 4 interior

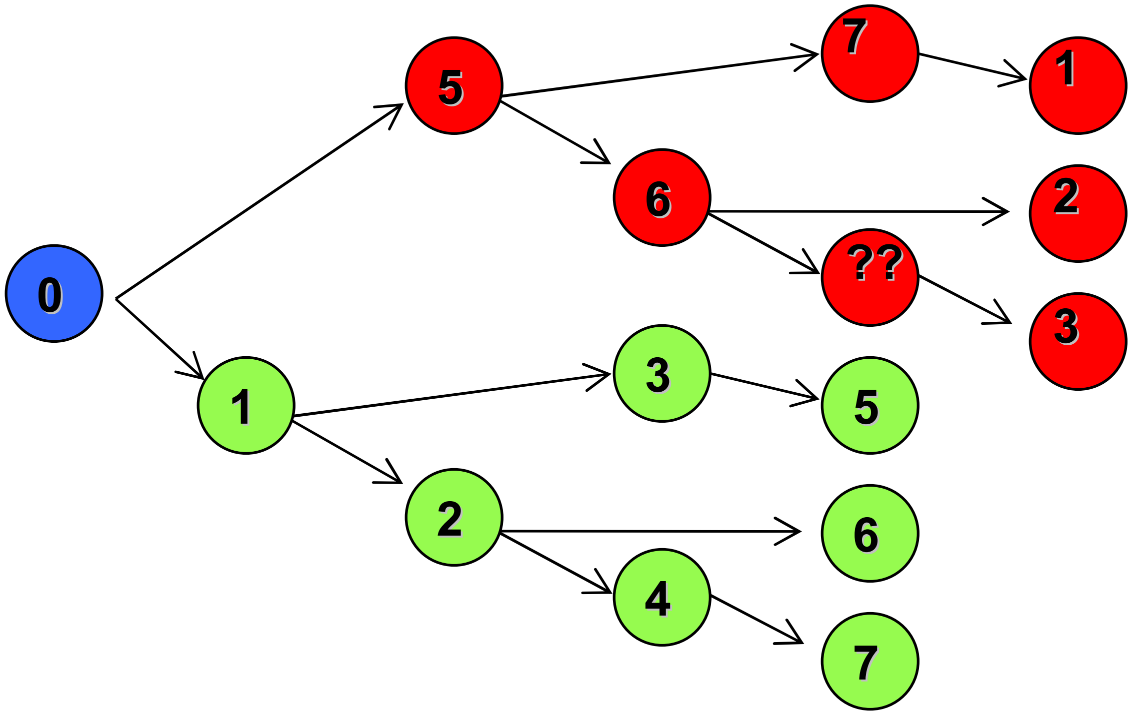
Coexistence of multiple distribution trees

- **Trivial problem**
- **Interior nodes for one tree set as leaves for other trees**
  - A node forwards chunks only for a tree



Unbalanced trees:
Tree Intertwining issue!

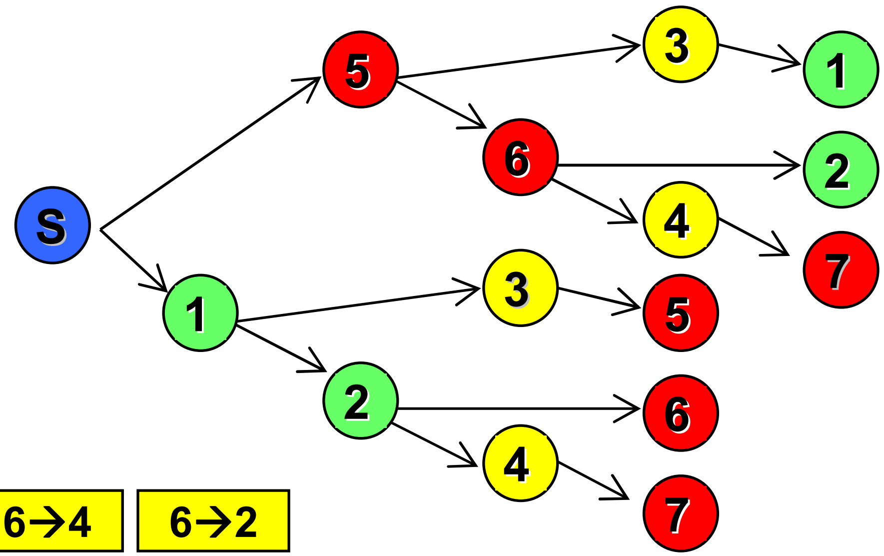
B=R (worst case)
2 trees



FOR EACH TREE:
➔ 4 interior nodes
➔ 3 leaves

3 < 4: tree intertwining problem no more a "interior" to "leaves" mapping problem: more subtle issue!

How to approach this issue



				6→4	6→2	6→4	6→2	
			5→6	5→3	5→6	5→3		
src	S→1	S→5	S→1	S→5	S→1			
N1		1→2	1→3	1→2	1→3	1→2	1→3	
N2			2→4	2→6	2→4	2→6	2→4	2→6
N3				3→5	3→1	3→5	3→1	3→5
N4				4→7	4→7	4→7	4→7	4→7

“three” classes of nodes: interior (1,2), leaves (5,6,7), 50% (3,4)

Result

- ➔ **Tree intertwining immediate to grasp & "hand-solve", proving it holds for small number of trees**
  - ⇨ Say k up to 6-8

- ➔ **Found constructive approach that proves that, given any k, the intertwining problem is feasible for such k**
  - ⇨ Cumbersome…
  - ⇨ still looking for a <u>simpler</u> proof

Outline

- ➔ P2P streaming in a nutshell
- ➔ Motivations and goals
- ➔ Constructive demonstration of the bounds
- ➔ The “tree intertwining” problem
- ➔ **A theory-driven distribution algorithm**

From theory to practice

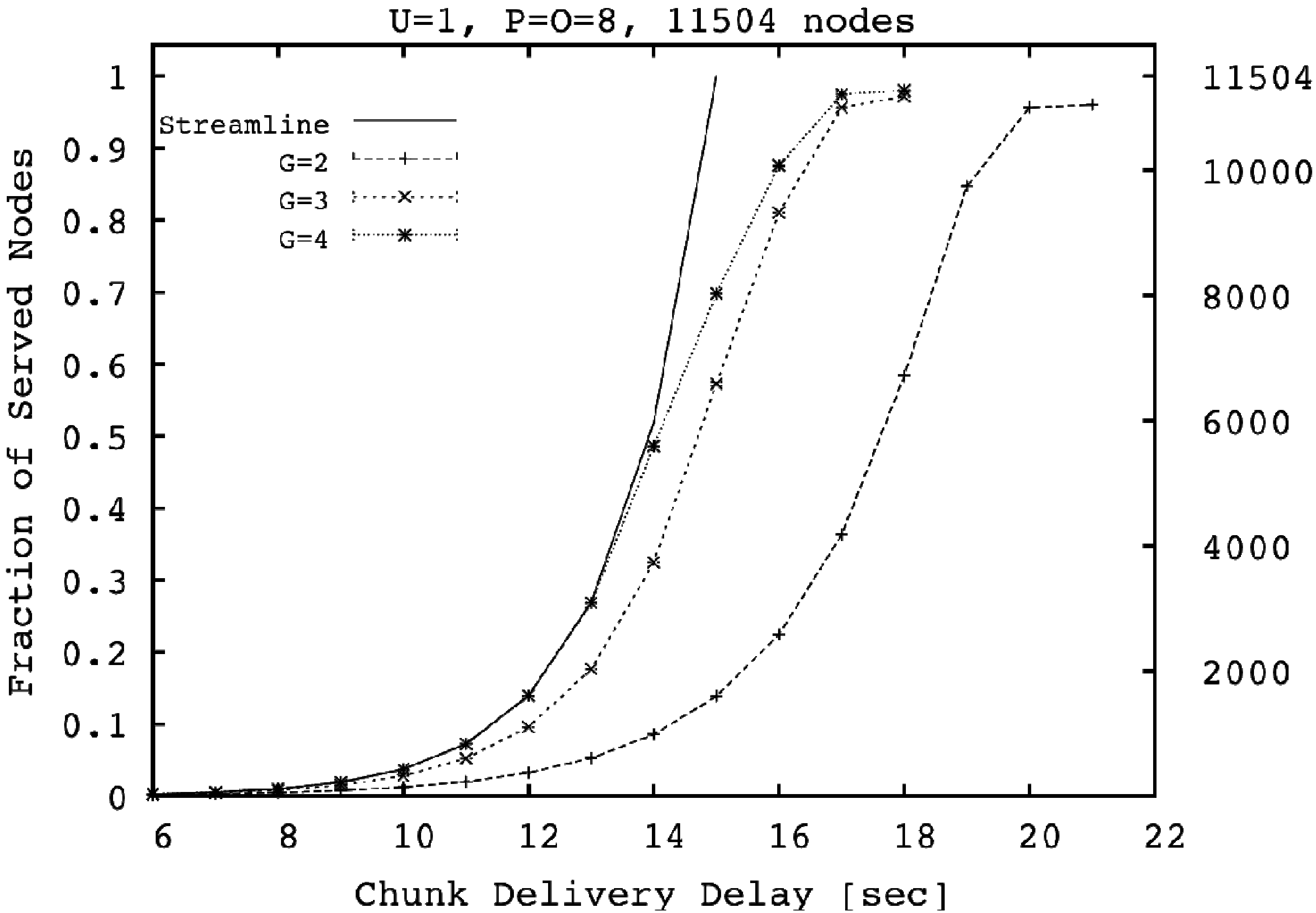
- ➔ **Theory taught us that**
  - ⇨ Serial transmission of chunks is better;
  - ⇨ Spreading chunks over multiple distribution trees is better;
- ➔ **How to design a mechanism which**
  - ⇨ Tries to mimic this in a purely distributed fashion
  - ⇨ Does not require to build and manage trees, but works on a per-chunk basis and exhibits robustness to node churn

Approach (idea)

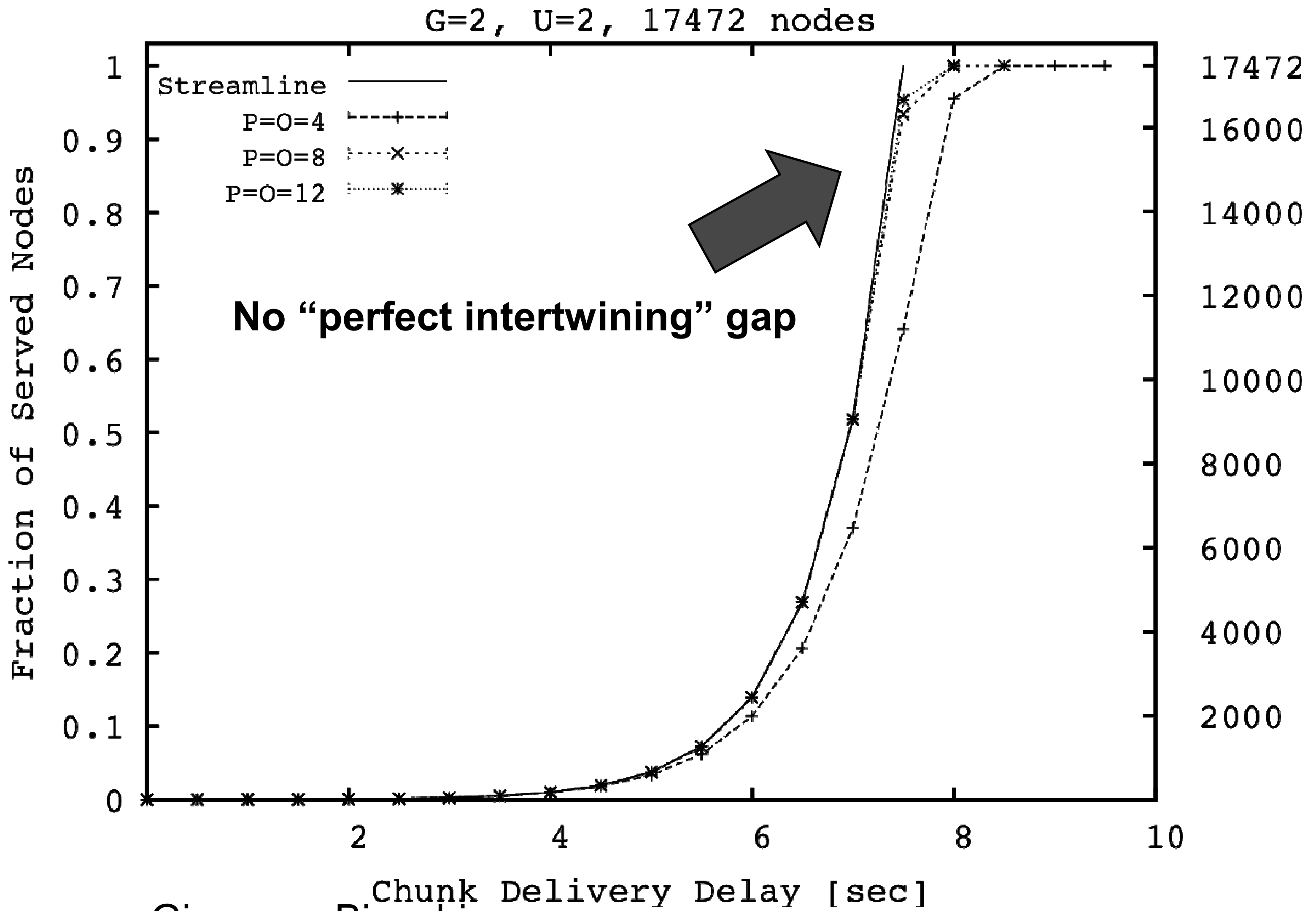
- **Divide peers in G groups**
  - Example: two groups
- **i-th chunk associated to group i mod G**
  - Example: odd/even chunks
- **Each peer has**
  - P partners belonging to its own group
  - O partners for each one of the other groups

- **Source uploads each chunk to nodes associated with the chunk group**
- **Peers try to perform up to U*G upload in series:**
  - First serving the partners of their own group that need that chunk (if any)
  - Then serving the rest of partners (if any)
- **If possible trying to maintain the same order of served nodes between different chunks**
  - To mimic the build-up of trees

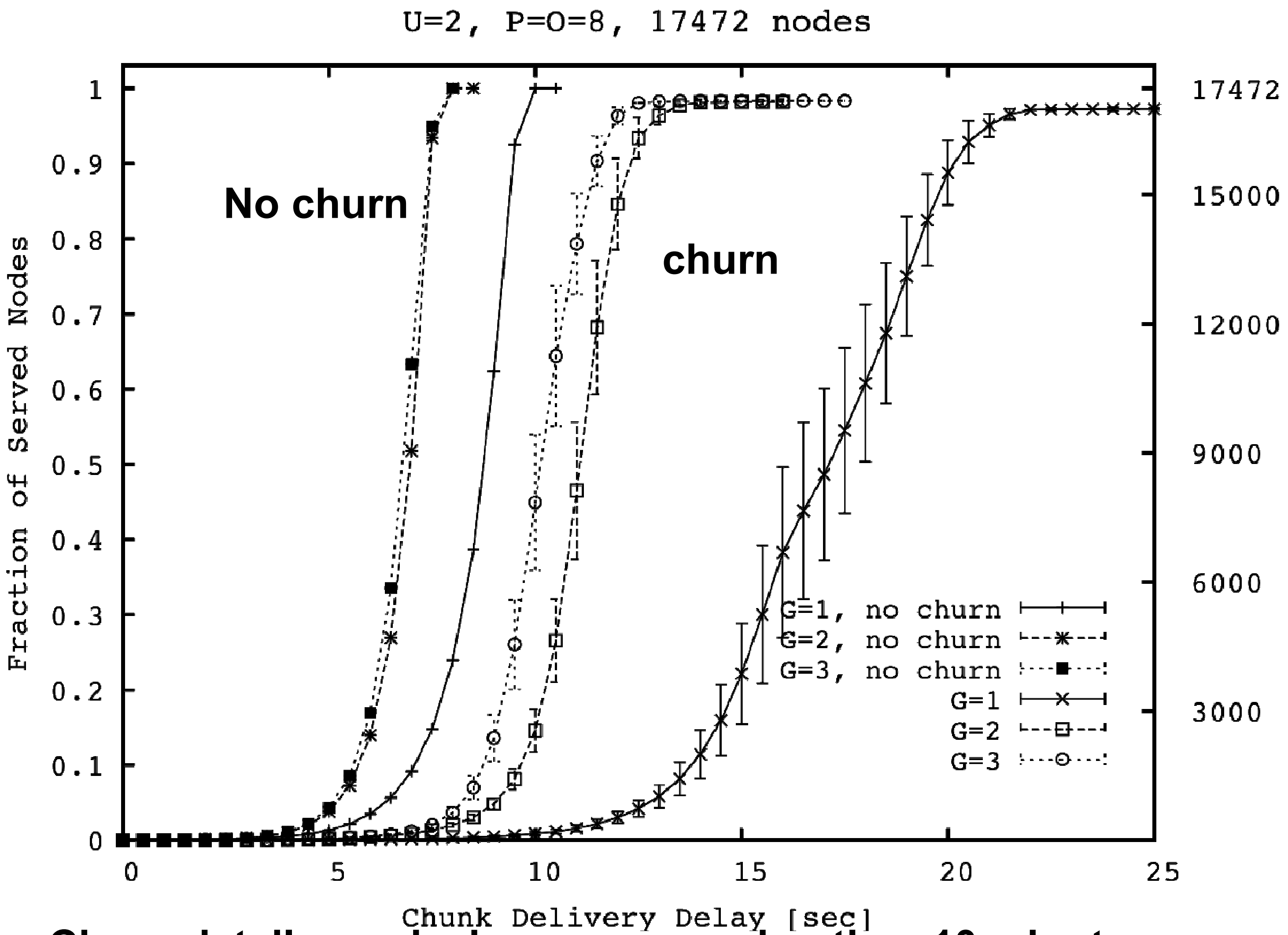
O-Streamline: Simulation Results



O-Streamline: Simulation Results



O-Streamline-Simulation Results



Churn details: nodes' average session time 10 minute