Tree-structured Data Regeneration with Network Coding in Distributed Network Storages

Jun Li[1], Shuang Yang[1], Xin Wang[1], Xiangyang Xue[1], Baochun Li[2]

[1]School of Computer Science, Furan University, China
[2]Department of Electrical Computer Engineering, University of Toronto, Canada

July 13, 2009



Distributed Storage System

Distributed storage

- save data into multiple storage peers distributively
- peer failure and data loss
- employ redundancy to guarantee data availability

Redundancy

- replica
- coded redundancy: erasure codes, random linear coding

Data regeneration

- data loss in the network
- providers $\longrightarrow$ newcomer



Distributed Storage System

Distributed storage

- save data into multiple storage peers distributively
- peer failure and data loss
- employ redundancy to guarantee data availability

Redundancy

- replica
- coded redundancy: erasure codes, random linear coding

Data regeneration

- data loss in the network
- providers $\longrightarrow$ newcomer



Improve data availability via redundancy

- replica: BitVault

Improve data availability via redundancy

- replica: BitVault
- erasure codes: OceanStore, TotalRecall



Improve data availability via redundancy

- replica: BitVault
- erasure codes: OceanStore, TotalRecall
- random linear coding:

Improve data availability via redundancy

- replica: BitVault
- erasure codes: OceanStore, TotalRecall
- random linear coding:
  - Accendanski et al. (NETCOD '05): is as good as erasure codes



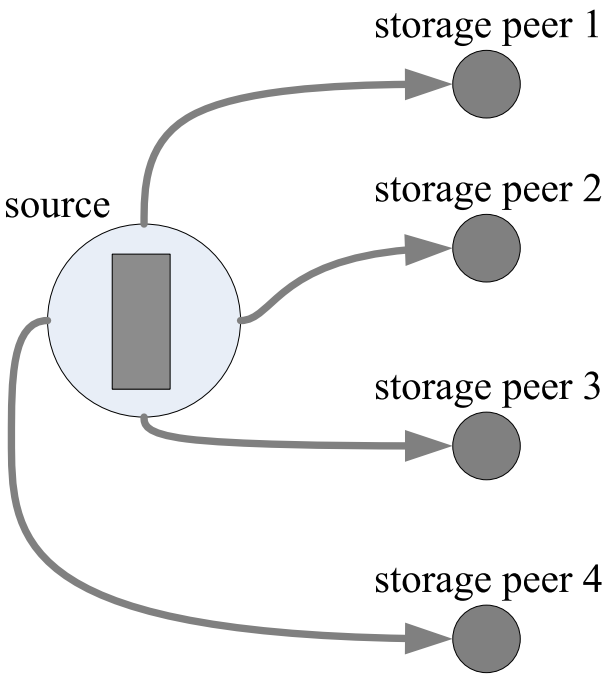
Improve data availability via redundancy

- replica: BitVault
- erasure codes: OceanStore, TotalRecall
- random linear coding:
  - Accendanski et al. (NETCOD '05): is as good as erasure codes
  - Dimakis et al. (INFOCOM '07): can reduce regeneration traffic



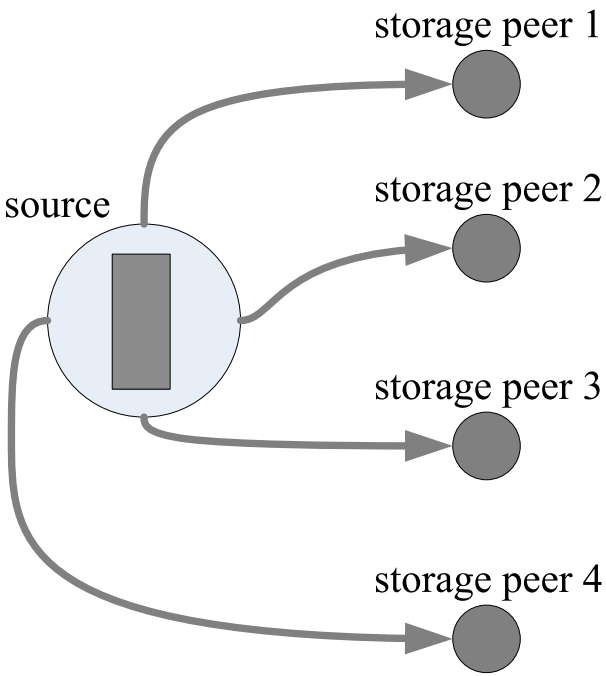
Replica vs. Coded redundancy

Replica redundancy

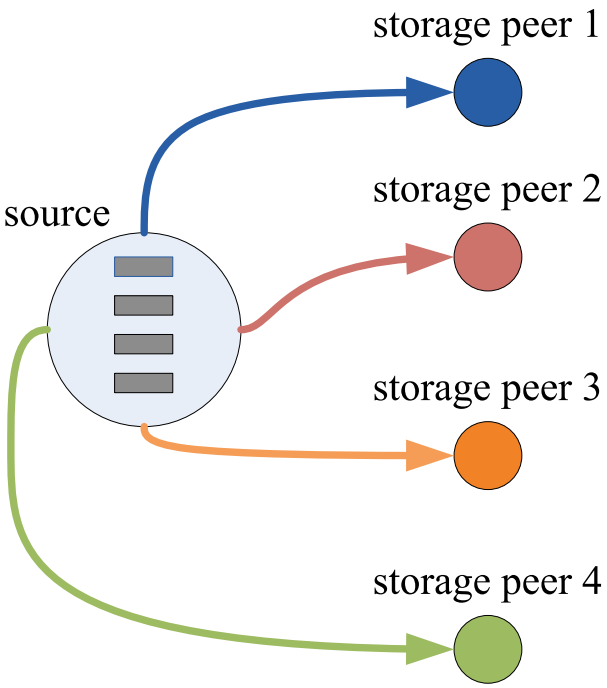


Replica vs. Coded redundancy

Replica redundancy



Coded redundancy



Distributed Storage System

Distributed storage

- save data into multiple storage peers distributively
- peer failure and data loss
- employ redundancy to guarantee data availability

Redundancy

- replica
- coded redundancy: erasure codes, random linear coding

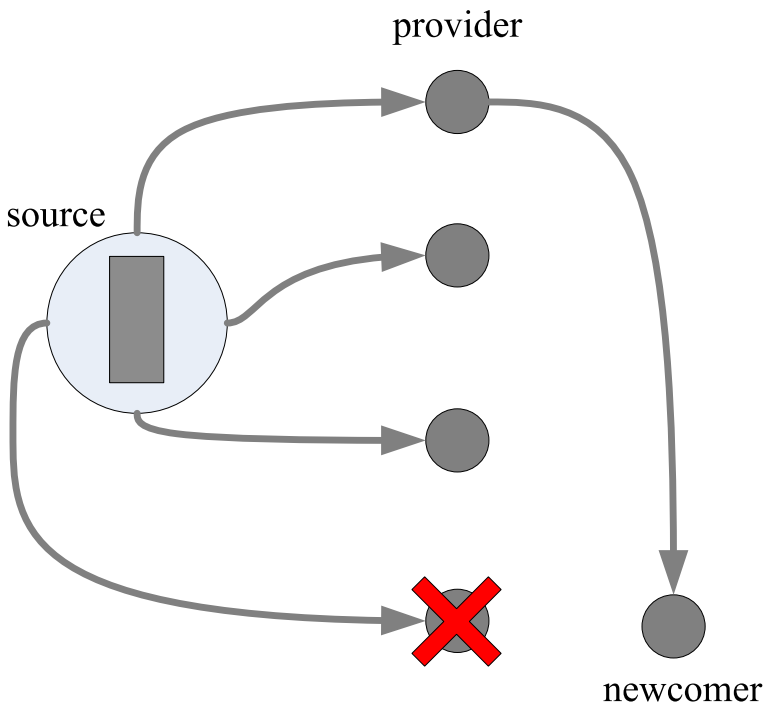
Data regeneration

- data loss in the network
- providers $\longrightarrow$ newcomer



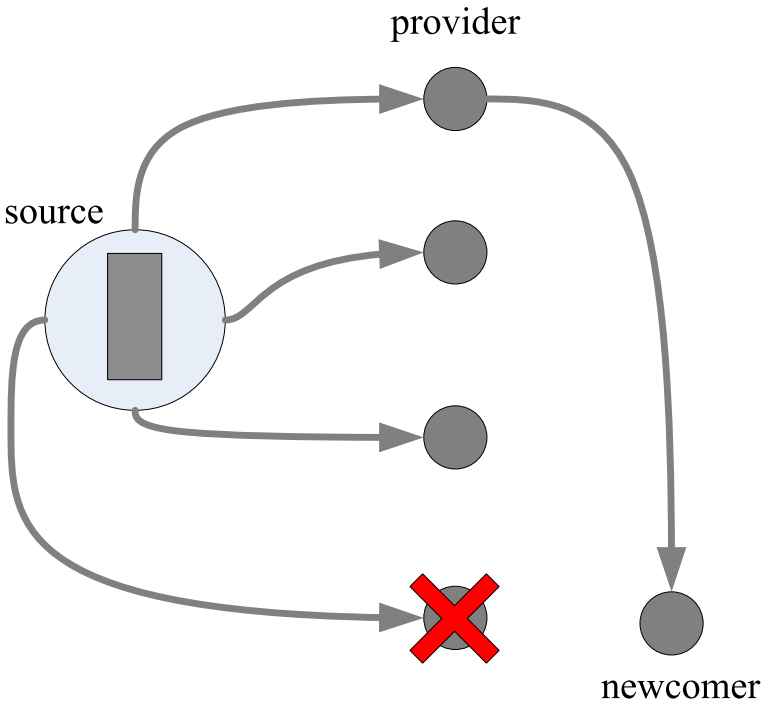
Star-structured data regeneration

Replica redundancy

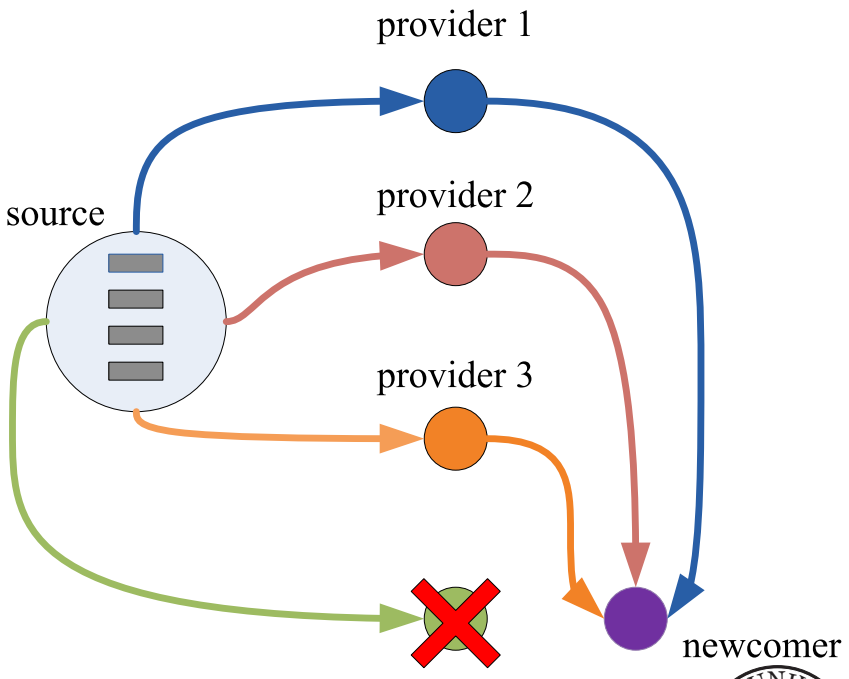


Star-structured data regeneration

Replica redundancy

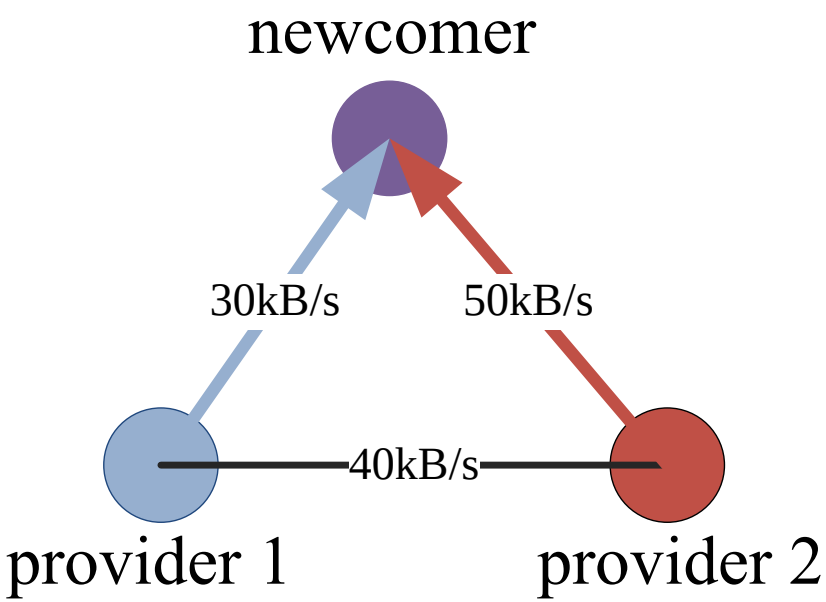


Coded redundancy

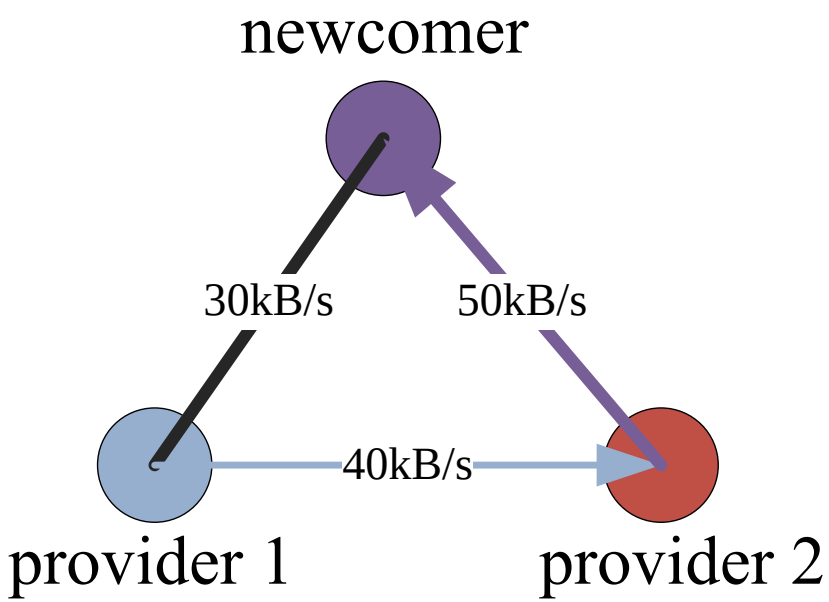


Our contribution

Tree-structured data regeneration



(a) Star-structured Regeneration Scheme



(b) Tree-structured Regeneration Scheme

Our work



Our work

1. Consider the heterogeneity of bandwidth capacity between nodes in the network and propose a tree-structured regeneration scheme.

Our work

1. Consider the heterogeneity of bandwidth capacity between nodes in the network and propose a tree-structured regeneration scheme.
2. Give the expected available bandwidth of the star-structured regeneration scheme and the tree-structured regeneration scheme.



Our work

1. Consider the heterogeneity of bandwidth capacity between nodes in the network and propose a tree-structured regeneration scheme.
2. Give the expected available bandwidth of the star-structured regeneration scheme and the tree-structured regeneration scheme.
3. Evaluate the performance of tree-structured regeneration scheme by simulations.



Model



Model

- Original file size: M bytes.

Model

- Original file size: M bytes.
- $(n, k)-$random linear coding



Model

- Original file size: M bytes.
- $(n, k)-$random linear coding
- Node set $V = \{V_0, V_1, \ldots, V_k\}$

Model

- Original file size: M bytes.
- $(n, k)-$random linear coding
- Node set $V = \{V_0, V_1, \ldots, V_k\}$
  - V_0 is the newcomer.
  - $V_1, \ldots,$ and V_k are providers.
  - Each node can receive node from multiple nodes, but can send data to only one node. (not to incur more regeneration traffic)
  - Encoding delay on each node is ignored.

Model

- Original file size: M bytes.
- $(n, k)-$random linear coding
- Node set $V = \{V_0, V_1, \ldots, V_k\}$
  - V_0 is the newcomer.
  - $V_1, \ldots,$ and V_k are providers.
  - Each node can receive node from multiple nodes, but can send data to only one node. (not to incur more regeneration traffic)
  - Encoding delay on each node is ignored.
- Edge set $E = \{(V_i, V_j) | i, j = 0, 1, \ldots, k, i < j\}$



Model

- Original file size: M bytes.
- $(n, k)-$random linear coding
- Node set $V = \{V_0, V_1, \ldots, V_k\}$
  - V_0 is the newcomer.
  - $V_1, \ldots,$ and V_k are providers.
  - Each node can receive node from multiple nodes, but can send data to only one node. (not to incur more regeneration traffic)
  - Encoding delay on each node is ignored.
- Edge set $E = \{(V_i, V_j) | i, j = 0, 1, \ldots, k, i < j\}$
- $\omega(V_i, V_j)$ denotes the bandwidth capacity between V_i and V_j.

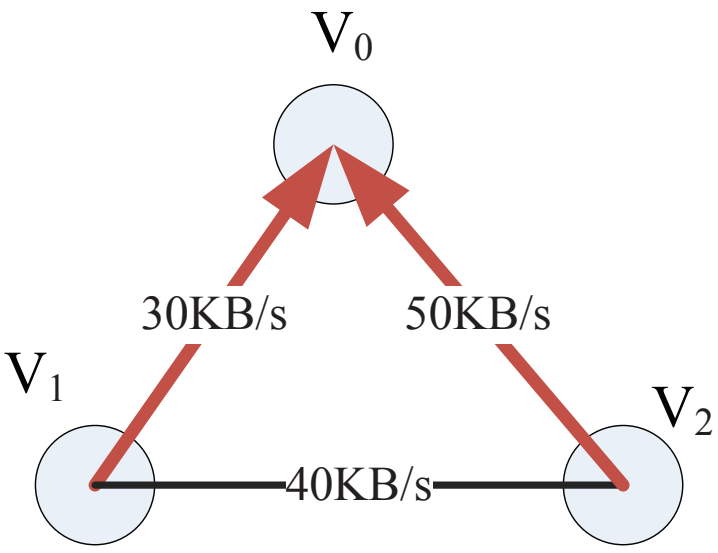


Model

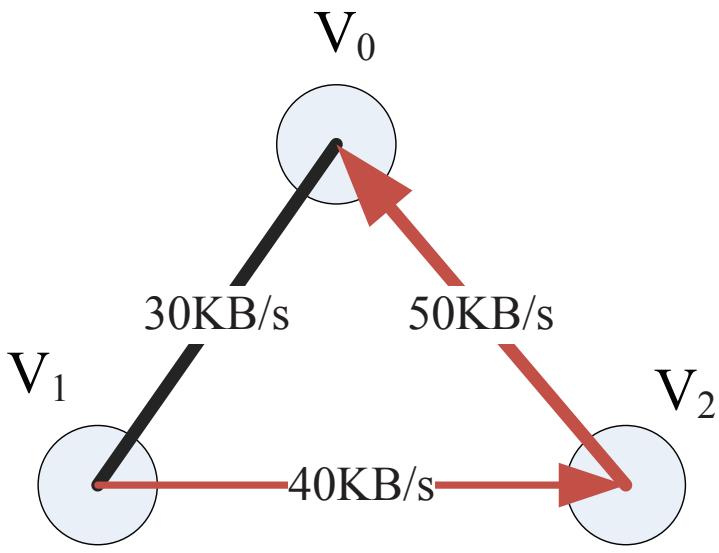
- Original file size: M bytes.
- $(n, k)-$random linear coding
- Node set $V = \{V_0, V_1, \ldots, V_k\}$
  - V_0 is the newcomer.
  - $V_1, \ldots,$ and V_k are providers.
  - Each node can receive node from multiple nodes, but can send data to only one node. (not to incur more regeneration traffic)
  - Encoding delay on each node is ignored.
- Edge set $E = \{(V_i, V_j)|i, j = 0, 1, \ldots, k, i < j\}$
- $\omega(V_i, V_j)$ denotes the bandwidth capacity between V_i and V_j.
- The weighted undirected complete graph $G_k = (V, E, \omega)$ denotes the network topology of the newcomer and the providers, where k is the number of providers in G_k, $k = |V| - 1$.



An Example



(a) Star-structured Regeneration Scheme



(b) Tree-structured Regeneration Scheme

Figure: Network model G_2.

Regeneration tree

Any spanning tree T in G_k can be a regeneration tree.

- Non-leaf provider receives data from child nodes, and sends out the linear combination.
- Pipeline transmission
- Regeneration time depends on the bottleneck edge in the regeneration tree.

The maximum spanning tree in G_k is an optimal regeneration tree, i.e. an bottleneck spanning tree.

Bottleneck bandwidth

- Sorting

  Considering $G_k = (V, E, \omega)$, let $E = \{e_1, e_2, \ldots, e_{\frac{k(k+1)}{2}}\}$, where $\omega(e_1) > \cdots > \omega(e_{\frac{k(k+1)}{2}})$.

Bottleneck bandwidth

- **Sorting**

  Considering $G_k = (V, E, \omega)$, let $E = \{e_1, e_2, \ldots, e_{\frac{k(k+1)}{2}}\}$, where $\omega(e_1) > \cdots > \omega(e_{\frac{k(k+1)}{2}})$.

- **Definition**

  MST(G_k)=r if and only if the minimal edge in the maximum spanning tree of $G_k = (V, E, \omega)$ is the r*th* maximal edge of E.

Bottleneck bandwidth

- **Sorting**

  Considering $G_k = (V, E, \omega)$, let $E = \{e_1, e_2, \ldots, e_{\frac{k(k+1)}{2}}\}$, where $\omega(e_1) > \cdots > \omega(e_{\frac{k(k+1)}{2}})$.

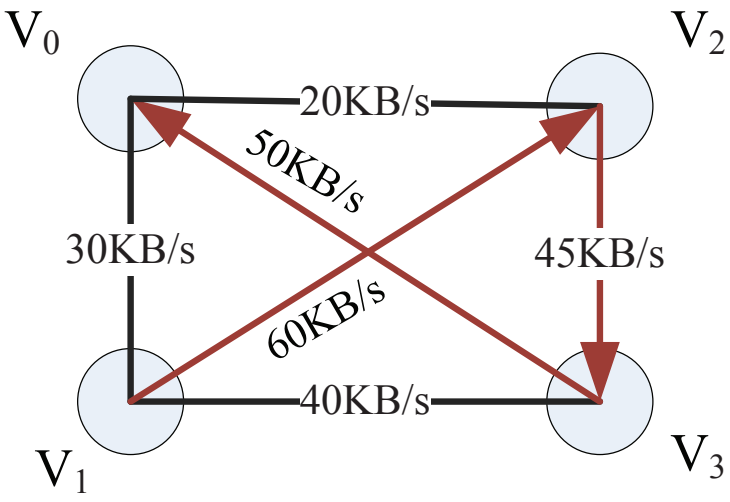
- **Definition**

  MST(G_k)=r if and only if the minimal edge in the maximum spanning tree of $G_k = (V, E, \omega)$ is the r*th* maximal edge of E.

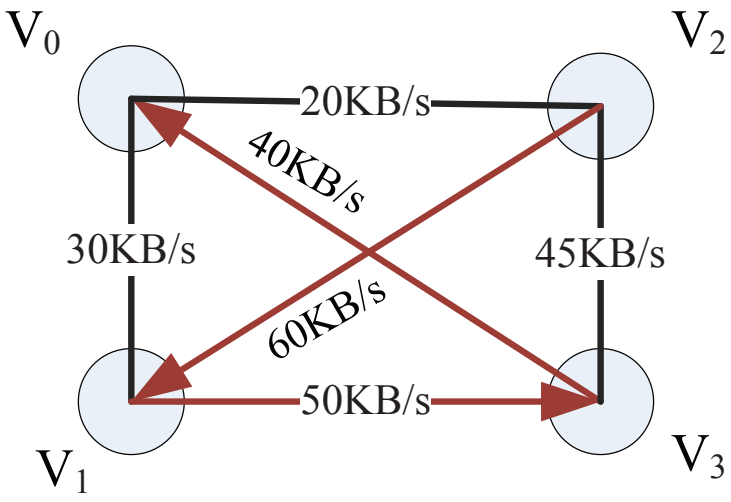
- **Property**

  $k \leq \text{MST}(G_k) \leq \frac{k(k-1)}{2} + 1$.





(a) Best case:
MST(G_3)=3



(b) Worst case:
MST(G_3)=4

Figure: Best and worst case of regeneration tree for G_3.

Theorem

Define $Q(l,j)$ to be the number of the connected graphs which contain l nodes and j edges, and $P(k+1,i)$ the probability that MST(G_k)=i. Thus in $G_k = (V, E, \omega)$,

$$p(k+1,i) = \frac{\sum_{l=1}^{k} \binom{k-1}{l-1} \sum_{j=0}^{i-1} Q(l,j) Q(k+1-l, i-1-j)}{\binom{\frac{k(k+1)}{2}-1}{i-1}},$$

and

$$Q(l,j) = \begin{cases} 0 & j < l-1 \\ \binom{l(l-1)/2}{j} \sum_{i=l-1}^{j} P(l,i) & l-1 \le j \le l(l-1)/2 \\ 0 & j > l(l-1)/2 \end{cases}$$

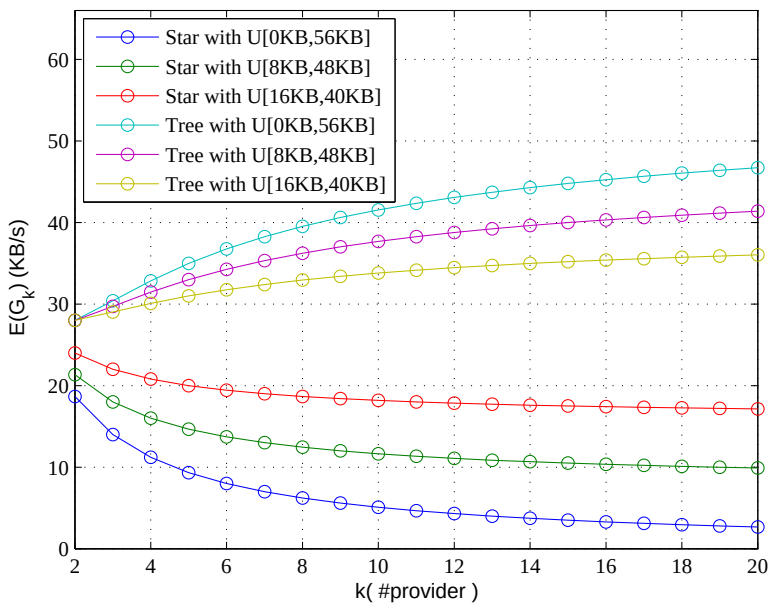


Expected available bandwidth capacity: $U[a, b]$

$$E_{(i:M_{k+1})} = \frac{(b-a)(M_{k+1}-i+1)}{M_{k+1}+1} + a.$$

$$E_{tree}(G_k) = \sum_{i=k}^{M_{k+1}-k+1} p(k+1,i) E_{(i:M_{k+1})}.$$

$$E_{star}(G_k) = \frac{(b-a)}{k+1} + a.$$



Simulation

based on trace and measured bandwidth in PlanetLab

- reduce regeneration time by 75%-82%
- improve data availability by 73%-124%

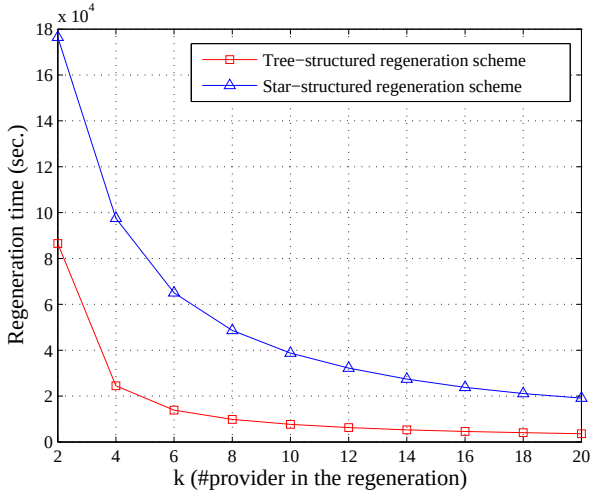


Figure: Regeneration time

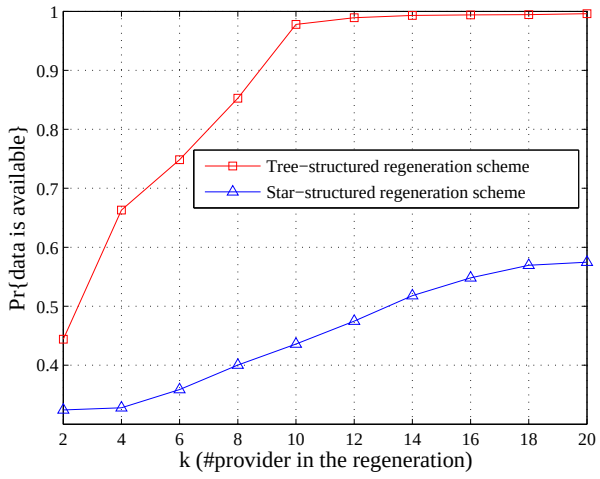


Figure: Data availability

- Conclusion
  1. Tree-structured regeneration scheme.

- Conclusion
  1. Tree-structured regeneration scheme.
  2. Expected available bandwidth capacity and the distribution of $\mathrm{MST}(G_k)$.

- Conclusion
  1. Tree-structured regeneration scheme.
  2. Expected available bandwidth capacity and the distribution of $\mathrm{MST}(G_k)$.
  3. Evaluation by simulation.

- Conclusion
  1. Tree-structured regeneration scheme.
  2. Expected available bandwidth capacity and the distribution of MST(G_k).
  3. Evaluation by simulation.
- Future Work
  1. Performance analysis of multiple tree regeneration scheme.

- Conclusion
  1. Tree-structured regeneration scheme.
  2. Expected available bandwidth capacity and the distribution of MST(G_k).
  3. Evaluation by simulation.
- Future Work
  1. Performance analysis of multiple tree regeneration scheme.
  2. Encoding scheme and regeneration scheme when #provider$>$ k.



- Conclusion
  1. Tree-structured regeneration scheme.
  2. Expected available bandwidth capacity and the distribution of MST(G_k).
  3. Evaluation by simulation.
- Future Work
  1. Performance analysis of multiple tree regeneration scheme.
  2. Encoding scheme and regeneration scheme when #provider$>$ k.
  3. Dynamic network model.



Thank you!



Thank you!

Q & A

