The Viceroy Network

Algorithms for Modern Communication Networks

Andreas Barthels

Ferienakademie 2008, Course 1
Province of Bolzano-Bozen, Italy

Prof. Dr. E. W. Mayr (TU Munich)
Prof. Dr. R. Wanka (Erlangen)

http://www.ferienakademie.eu/

September 21 - October 3, 2008

Outline

Introduction

Previous Peer2Peer Networks

- Single peer linkage cost not constant or fair, or,
- Suboptimal lookup efficiency

Viceroy Design Goals

- Uniformly distribute minimal linkage cost on every peer
- Avoid bottlenecks
- Maximize Lookup Efficiency for Huge Networks

Introduction

Previous Peer2Peer Networks

- Single peer linkage cost not constant or fair, or,
- Suboptimal lookup efficiency

Viceroy Design Goals

- Uniformly distribute minimal linkage cost on every peer
- Avoid bottlenecks
- Maximize Lookup Efficiency for Huge Networks

Outline

Network Characteristics

Network Degree deg

Maximum number of outgoing links out of a single peer

Network Diameter dia

Longest of all shortest distances between two peers

Outline

Trade-Off in Network Design

- Within a distance d of a peer there are at most deg^d peers reachable.
- The Definition of Diameter and Degree thus yield for the number of peers n:

$$deg^{dia} \geq n$$

and thus

$$\Rightarrow dia \geq \frac{\log(n)}{\log(deg)}.$$

- Optimum:

$$deg = \text{const}, \; dia \propto \log(n).$$

Trade-Off in Network Design

- Within a distance d of a peer there are at most deg^d peers reachable.
- The Definition of Diameter and Degree thus yield for the number of peers n:

$$deg^{dia} \geq n$$

and thus

$$\Rightarrow dia \geq \frac{\log(n)}{\log(deg)}.$$

- Optimum:

$$deg = \text{const}, \; dia \propto \log(n).$$

Trade-Off in Network Design

- Within a distance d of a peer there are at most deg^d peers reachable.
- The Definition of Diameter and Degree thus yield for the number of peers n:

$$deg^{dia} \geq n$$

and thus

$$\Rightarrow dia \geq \frac{\log(n)}{\log(deg)}.$$

- Optimum:

$$deg = \text{const}, \; dia \propto \log(n).$$

Trade-Off in Network Design

- Within a distance d of a peer there are at most deg^d peers reachable.
- The Definition of Diameter and Degree thus yield for the number of peers n:

$$deg^{dia} \geq n$$

and thus

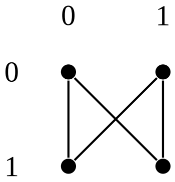
$$\Rightarrow dia \geq \frac{\log(n)}{\log(deg)}.$$

- Optimum:

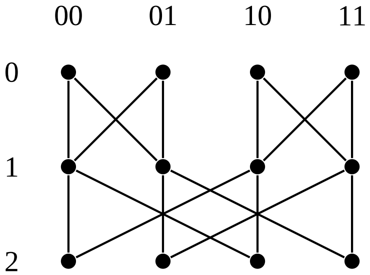
$$deg = \text{const}, \; dia \propto \log(n).$$

Outline

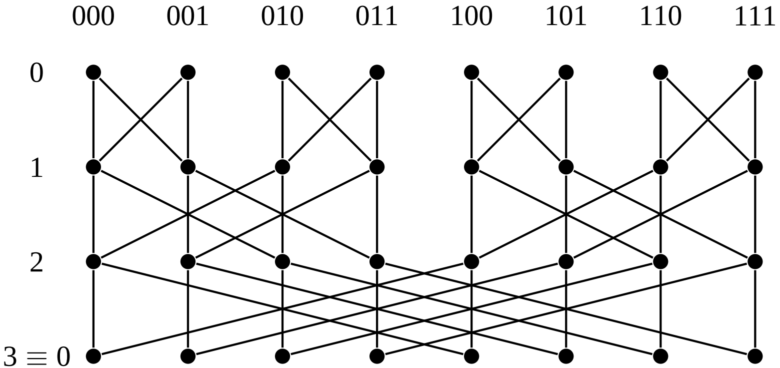
Butterfly-Networks: Level 1



Butterfly-Networks: Level 2



Butterfly-Networks: Level 3



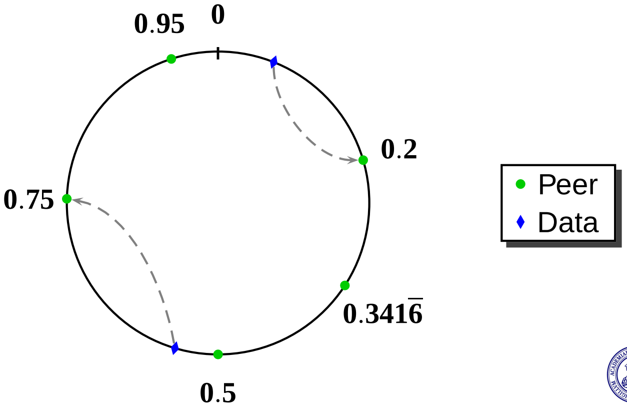
Outline

Viceroy

Limenitis archippus (Viceroy Butterfly)

Viceroy: Basics

- Viceroy implements a 1-dimensional distributed hashtable
- Keys are mapped to $[0, 1)$
- Data is assigned to the clockwise-closest successor

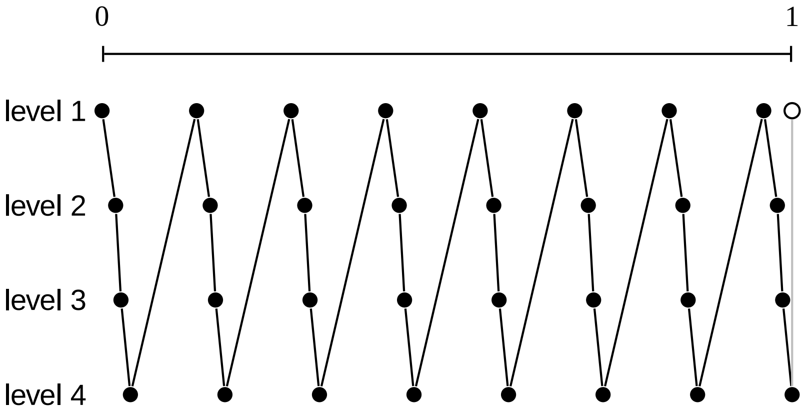


Viceroy: Network Structure

At this point, we assume that each peer s

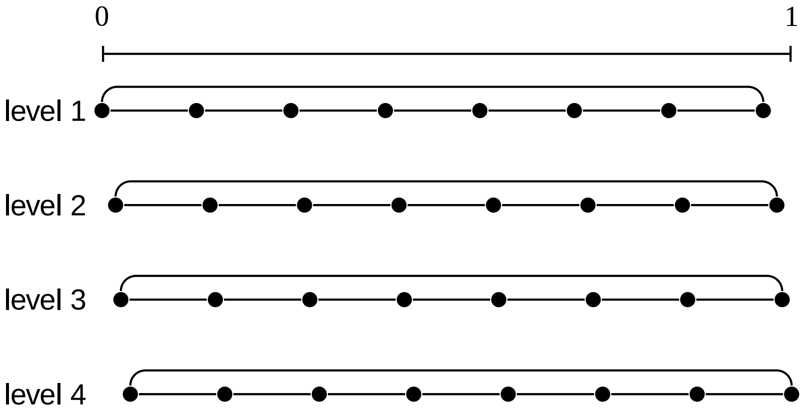
- features a $s.position$ that is determined by its ID, and,
- a level ($s.level$).
- Links are established to...

Best Case Emulation - Ring Links



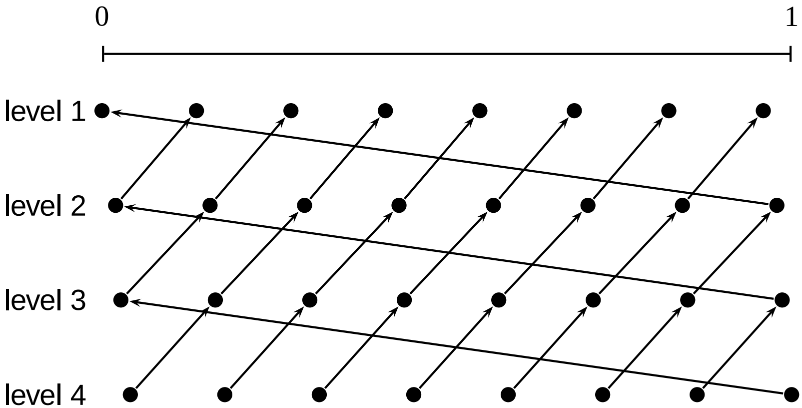
... the first successor and predecessor of s on the Interval $[0, 1)$, regardless of the level...

Best Case Emulation - Level Links



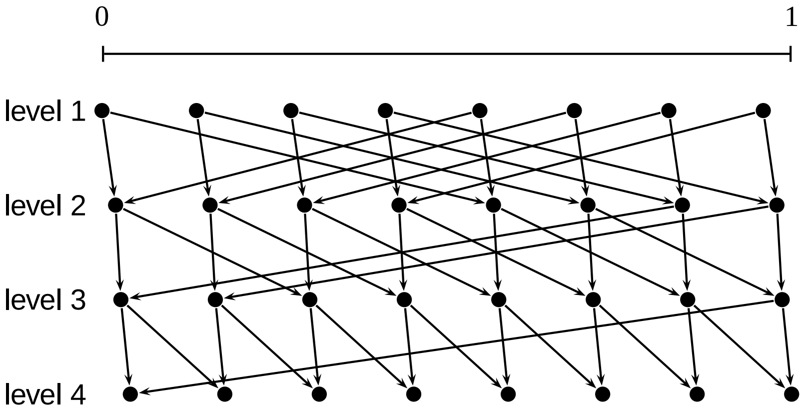
... the first successor and predecessor of s on $s.level$...

Best Case Emulation - Up Links



... the clockwise-closest peer on level $s.level - 1$, if $s.level > 0$...

Best Case Emulation - Butterfly Links



... the clockwise-closest peer on $s.level + 1$ to $s.position$ and $(s.position + (1/2)^{s.level}) \mod 1$...

Viceroy Outdegree

The outdegree is given by counting:

- 2 ring links
- 2 level links
- 1 up link
- 2 butterfly links

$\Rightarrow$ Constant network degree of 7.

Viceroy Outdegree

The outdegree is given by counting:

- 2 ring links
- 2 level links
- 1 up link
- 2 butterfly links

$\Rightarrow$ Constant network degree of 7.

Outline

Level Selection

Performance is proportional to the maximum level

Level selection is key in bounding performance:

- Assume uniform distribution of the peers
- Expected number of nodes is $\frac{k}{\text{distance to the } k\text{-th node}}$
- The estimation for the number of peers $\hat{n}$ is

$$\log\left(\frac{n}{c\log(n)}\right) \leq \lfloor \log(\hat{n}) \rfloor \leq c' \log(n)$$

with high probability.

Level Selection

Performance is proportional to the maximum level

Level selection is key in bounding performance:

- Assume uniform distribution of the peers
- Expected number of nodes is $\frac{k}{\text{distance to the } k\text{-th node}}$
- The estimation for the number of peers $\hat{n}$ is

$$\log\left(\frac{n}{c\log(n)}\right) \le \lfloor \log(\hat{n}) \rfloor \le c' \log(n)$$

with high probability.

Outline

Routing

Step 1 - Route up to level 1

Follow up-links. Their number is $O(\log(n))$.

Step 2 - Route down to target

Is the target in between the left and the right down-link, route left, else right. $O(\log(n))$

Step 3 - Traverse level and outer rings

If there are no down links anymore

1. Try level-links in direction of target
2. Route along the outer ring

$\Rightarrow$ Logarithmic complexity with high probability.

Routing

Step 1 - Route up to level 1

Follow up-links. Their number is $O(\log(n))$.

Step 2 - Route down to target

Is the target in between the left and the right down-link, route left, else right. $O(\log(n))$

Step 3 - Traverse level and outer rings

If there are no down links anymore

1. Try level-links in direction of target
2. Route along the outer ring

$\Rightarrow$ Logarithmic complexity with high probability.

Routing

Step 1 - Route up to level 1

Follow up-links. Their number is $O(\log(n))$.

Step 2 - Route down to target

Is the target in between the left and the right down-link, route left, else right. $O(\log(n))$

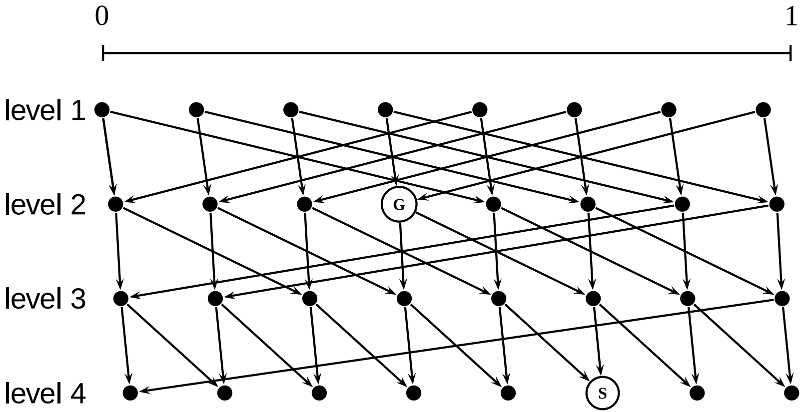
Step 3 - Traverse level and outer rings

If there are no down links anymore

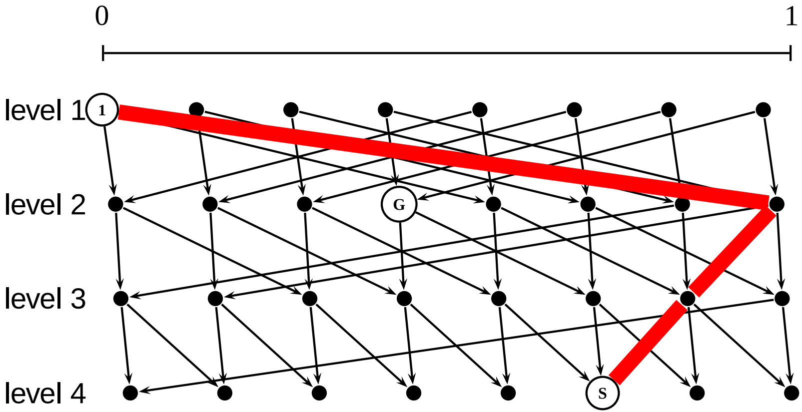
1. Try level-links in direction of target
2. Route along the outer ring

$\Rightarrow$ Logarithmic complexity with high probability.

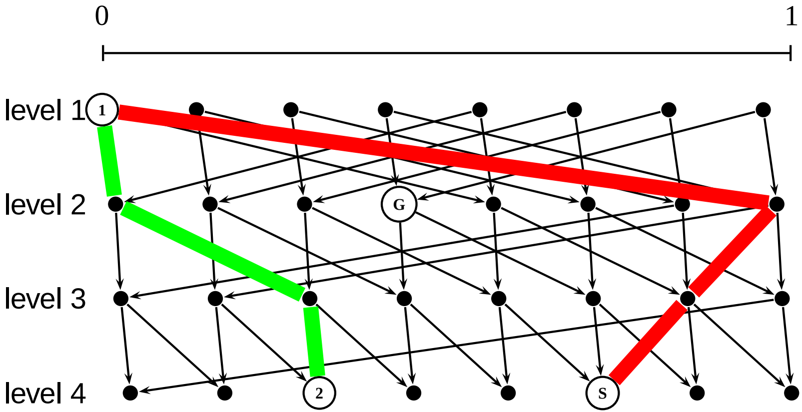
Routing Example



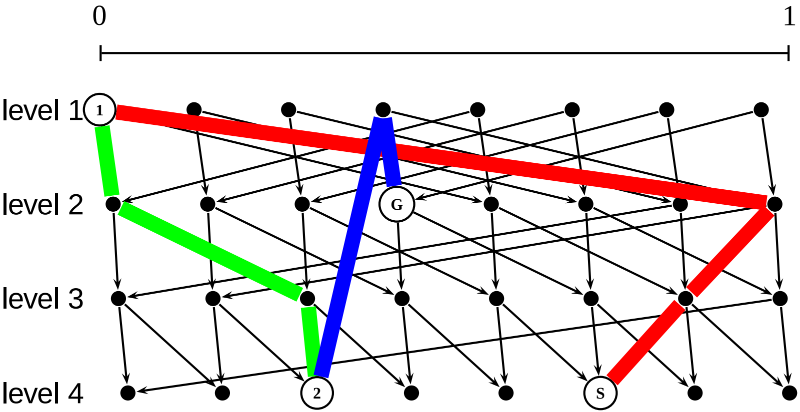
Routing Example - Step 1



Routing Example - Step 2



Routing Example - Step 3



Outline

Peer Insertion

1. Find peer responsible for SystemID. $O(\log(n))$
2. Reassign keys according to successor relationship. $O(1)$
3. Estimate number of peers $\hat{n}$ through distance to k-successing peers ($\hat{n} = \frac{k}{d_k}$). $O(1)$
4. Choose Butterfly-Level uniformly out of $1 \leq l \leq \lfloor \log(\hat{n}) \rfloor$. $O(1)$
5. Update links. $O(1)$ in expectation, $O(\log(n))$ with high probability
6. Additional to plain Butterfly, link peers on each level and create uplink. $O(1)$

$\Rightarrow$ Logarithmic complexity with high probability.

Peer Insertion

1. Find peer responsible for SystemID. $O(\log(n))$
2. Reassign keys according to successor relationship. $O(1)$
3. Estimate number of peers $\hat{n}$ through distance to k-successing peers ($\hat{n} = \frac{k}{d_k}$). $O(1)$
4. Choose Butterfly-Level uniformly out of $1 \leq l \leq \lfloor \log(\hat{n}) \rfloor$. $O(1)$
5. Update links. $O(1)$ in expectation, $O(\log(n))$ with high probability
6. Additional to plain Butterfly, link peers on each level and create uplink. $O(1)$

$\Rightarrow$ Logarithmic complexity with high probability.

Peer Insertion

1. Find peer responsible for SystemID. $O(\log(n))$
2. Reassign keys according to successor relationship. $O(1)$
3. Estimate number of peers $\hat{n}$ through distance to k-successing peers ($\hat{n} = \frac{k}{d_k}$). $O(1)$
4. Choose Butterfly-Level uniformly out of $1 \leq l \leq \lfloor \log(\hat{n}) \rfloor$. $O(1)$
5. Update links. $O(1)$ in expectation, $O(\log(n))$ with high probability
6. Additional to plain Butterfly, link peers on each level and create uplink. $O(1)$

$\Rightarrow$ Logarithmic complexity with high probability.

Peer Insertion

1. Find peer responsible for SystemID. $O(\log(n))$
2. Reassign keys according to successor relationship. $O(1)$
3. Estimate number of peers $\hat{n}$ through distance to k-successing peers ($\hat{n} = \frac{k}{d_k}$). $O(1)$
4. Choose Butterfly-Level uniformly out of $1 \leq l \leq \lfloor \log(\hat{n}) \rfloor$. $O(1)$
5. Update links. $O(1)$ in expectation, $O(\log(n))$ with high probability
6. Additional to plain Butterfly, link peers on each level and create uplink. $O(1)$

$\Rightarrow$ Logarithmic complexity with high probability.

Peer Insertion

1. Find peer responsible for SystemID. $O(\log(n))$
2. Reassign keys according to successor relationship. $O(1)$
3. Estimate number of peers $\hat{n}$ through distance to k-successing peers ($\hat{n} = \frac{k}{d_k}$). $O(1)$
4. Choose Butterfly-Level uniformly out of $1 \leq l \leq \lfloor \log(\hat{n}) \rfloor$. $O(1)$
5. Update links. $O(1)$ in expectation, $O(\log(n))$ with high probability
6. Additional to plain Butterfly, link peers on each level and create uplink. $O(1)$

$\Rightarrow$ Logarithmic complexity with high probability.

Peer Insertion

1. Find peer responsible for SystemID. $O(\log(n))$
2. Reassign keys according to successor relationship. $O(1)$
3. Estimate number of peers $\hat{n}$ through distance to k-successing peers ($\hat{n} = \frac{k}{d_k}$). $O(1)$
4. Choose Butterfly-Level uniformly out of $1 \leq l \leq \lfloor \log(\hat{n}) \rfloor$. $O(1)$
5. Update links. $O(1)$ in expectation, $O(\log(n))$ with high probability
6. Additional to plain Butterfly, link peers on each level and create uplink. $O(1)$

$\Rightarrow$ Logarithmic complexity with high probability.

Peer Insertion

1. Find peer responsible for SystemID. $O(\log(n))$
2. Reassign keys according to successor relationship. $O(1)$
3. Estimate number of peers $\hat{n}$ through distance to k-successing peers ($\hat{n} = \frac{k}{d_k}$). $O(1)$
4. Choose Butterfly-Level uniformly out of $1 \leq l \leq \lfloor \log(\hat{n}) \rfloor$. $O(1)$
5. Update links. $O(1)$ in expectation, $O(\log(n))$ with high probability
6. Additional to plain Butterfly, link peers on each level and create uplink. $O(1)$

$\Rightarrow$ Logarithmic complexity with high probability.

Outline

Ensuring Constant Indegree

Although the expected indegree is constant, it can still be logarithmic for some node.

- local coordination on “buckets” with $O(\log(n))$ peers
- position and level selection is not random any more
- good and sane distribution of peers over stretches and levels

- really complex procedure

Outline

Peer Failure / Leaving Peer

1. The successing peer has to take over the data. $O(1)$
2. Every former link has to find a replacement. $O(\log(n))$

Peer Failure / Leaving Peer

1. The successing peer has to take over the data. $O(1)$
2. Every former link has to find a replacement. $O(\log(n))$

Outline

Summary

Viceroy features

- constant degree
- logarithmic diameter
- quite complex implementation

Thank you

Additional Literature I

Peter Mahlmann and Christian Schindelhauer
Peer-to-Peer-Netzwerke.
Springer Berlin Heidelberg, 2007.
<springerlink.de>

Dahlia Malkhi and Moni Naor and David Ratajczak
Viceroy: A Scalable and Dynamic Emulation of the Butterfly
Proceedings of the twenty-first annual symposium on Principles of distributed computing, 2002.