BDCAT: Balanced Dynamic Content-Addressing in Trees

Stefanie Roos, Martin Byrenheid, Clemens Deusser, and Thorsten Strufe

INFOCOM ‘17
BDCAT in a nutshell

What do we want to do?
• *Build a secure, low-delay darknet*

Why is that difficult?
• Publish/access without fear of retribution/censorship
• -> Routing/content addr. in restricted environments

What’s this paper about?
• Content addressing on a
tree-shaped (efficient routing!) overlay
• with fair load balancing

log \quad polylog \quad log
Content Addressing 101

• Derive identifier from resources that map to a (zone within the) network address

• Register and retrieve information at respective node

| A bit more formal, in a network G=(V,E) with resources O: |
| id: V -> ID | map: O -> ID |
| store: O -> V | retrieve: ID -> O |

• *First-term CS students: Hash tables*

• *Cloud/datacenter people: memcached*

• *P2P people: DHT*

• *Everybody else: DNS*
Routing 101

Find a path of links to a destination (store, retrieve)

1. Assign node addresses and connectivity
2. Disseminate connectivity information
3. Calculate distances via neighbors (DV/LS)
4. Minimize distance to destination (greedy)

2 degrees of freedom: addresses / connections

- IP: centralized addressing, strategic connections
- DNS: centralized addressing, connect to tree
- DHT: random addresses, connect to routable structure

Information is restricted, connections are fixed! ¹

Meet Embeddings

Single degree of freedom

1. Adapt addresses
2. Estimate distances
3. Minimize distance

A bit more formal, let $ID(v)$ be address of $v \in V$. A network embedding on graph $G$ is a function $ID : V \rightarrow M$ to metric space $M$ equipped with a distance $d : M \times M \rightarrow \mathbb{R}^+$. 

(1): Approx. routable structure (greedy?), eg a tree

PIE embedding$^{1,2}$

$$d(s, t) := |s| + |t| - 2(cpl(s, t))$$

Map resources to nodes, \( \text{map} : O \rightarrow M \)

Assume max depth of tree to be \( L \):
Define metric space to be vectors (of \( L \) components):

\[
\text{map}(o) = (h_1(o), h_2(o), h_3(o), \ldots, h_L(o))
\]

distance measured as above.

Maps uniform into components \( \rightarrow \) overall namespace
Trees determined by social graph, unbalanced, static.

Task is to guarantee \((f,\delta)-balance\):

\[
\text{for all } v \in V \text{ less than } f \frac{1}{|V|} + \delta \text{ assigned.}
\]

Intervals (zones) assigned to each component:
1. Estimate \#nodes in branch \(\text{succ}(v)\)
2. Allocate zones \(z(v)\) accordingly:

assign, width: \(|z(v)| = 2^L \frac{\text{succ}(v)}{\text{succ(parent}(v))}\)

Distance metric considering zones:
- \(d(x,y)=|x|+|y|-2\text{con}(x,y)\)
- with: \(\text{con}(x,y)=\max\{i: x_j \in y_j \forall j \leq i\}\)
Churn: nodes are unreliable, *trees are delicate*

Guarantee polylog stabilization complexity (+ *balance*)

- **Tree**: Stabilize underlying branches locally
  - Join (or: depart/reconnect)
  - a: local reassignment possible
  - b: escalate to parent

**Results:**
- logarithmic $f$: $O(\log^3 n \ E(S))$
- polylog $f$: $O(\log^6 n)$

---

*Paper: slightly different algorithm, disseminating estimate of $|V|$*
Balanced Dynamic Content Addressing

Churn: nodes are unreliable, trees are delicate

Guarantee stabilization in complexity (+ balance)

- Tree: Stabilize underlying branches locally
- Join (or depart / reconnect)
  - a: local reassignment
  - b: escalate to parent

Results:
- logarithmic $f$: $O(\log^3 n)$
- polylog $f$: $O(\log^6 n)$

Standing on the Shoulders of Giants

Virtual Overlays

- Long routes
  - Voute: polylog \(^1\)
  - Others: > polylog \(^2\)

Kleinberg, Prefix Emb. \(^3,4\)

- Unfair load:

Topography-aware hashes \(^5\)

\[ h(\mathcal{G}, \mathcal{T}) \]

- Churn: Reassign everything

BDCAT

\[ (0,10/3), (10/3, 13) \]

- log / log / polylog

---

[1] Roos et al., INFOCOM 16
[2] Mittal et al., NDSS 12
References


T. Paul et al. C4PS-helping Facebookers manage their privacy settings. In International Conference of Social Informatics


