Übung zu Peer-to-Peer und Cloud Computing

Übungstermin 04: Besprechung des Übungsblattes 03

Dominik Rauh

28. November 2018

Universität Augsburg
Institut für Informatik
Lehrstuhl für Organic Computing

Beschreiben Sie Symphonys grundlegenden Aufbau!

Symphony ist ein Ring mit Fingern

ORGANIC COMPUTING

- „Symphony ist wie Chord nur mit probabilistischen Long-Distance-Links.“
- Knoten bilden Ring mit Adressbereich $[0, 1)$
  - Verbindungen zu direkten Nachbarn (Short-Distance-Links, SDLs)
  - zusätzlich Long-Distance-Links (LDLs, Finger wie in Chord)

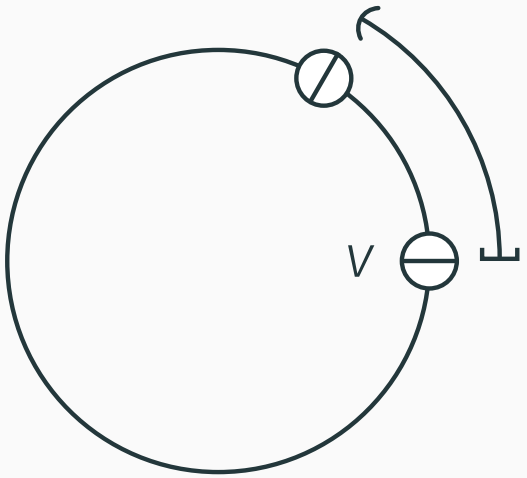
Hashing nach $[0, 1)$

ORGANIC COMPUTING

- sei h eine „klassische“ Hash-Funktion $h : \mathrm{Data} \to \mathbb{B}^m$
- es gilt $\mathbb{B}^m \simeq \mathbb{N}$
- sei $f : \mathbb{N} \to \mathbb{R}$, $f(x) = x/2^m$
- dann ist $f \circ h : \mathrm{Data} \to [0, 1)$ die in Symphony genutzte Hash-Funktion

Zuständigkeit der Knoten

Knoten *v* zuständig für Daten mit Schlüsseln im Bereich zwischen seiner und der Adresse seines *Vorgängers* im Uhrzeigersinn



Beschreiben Sie Symphonys Wahl der Long-Distance-Links!

Wahl der LDLs

Peer v_a mit Adresse a soll einen LDL aufbauen.

1. Auswahl eines zufälligen Offsets o verteilt nach

$$p_n(x) = \begin{cases} \frac{1}{x \ln n}, & x \in [\frac{1}{n}, 1] \\ 0, & \text{sonst} \end{cases}$$

   z.B. mittels der Prozedur $e^{\ln n(rand()-1)}$.

2. Aufbau eines LDLs zu Peer v_{a+o}.

Anmerkung zur Wahl der LDLs

$p(x)$ Wahrscheinlichkeitverteilung über $[0, 1)$

$\Rightarrow x \mapsto [[p(x) + a]]$ ebenfalls

Anzahl der LDLs

ORGANIC COMPUTING

- gegeben: feste Anzahl $k \geq 1$ von LDLs
- mögliche Erweiterungen: k abhängig von Netzwerkgröße/Peerleistung/...
- Peers lehnen neue Verbindungen ab, wenn sie $\geq 2k$ Verbindungen haben
  $\Rightarrow$ erneute Auswahl eines zufälligen Offsets

Beschreiben Sie Symphonys Routing-Protokoll!

Routing-Problem

ORGANIC COMPUTING

- gegeben: Schlüssel x
- gesucht: Peer, der für die Daten d zuständig ist, für die

$$(f \circ h)(d) = x$$

Routing in Symphony

ORGANIC COMPUTING

- unidirektionaler Ring: Weiterleitung, sodass *Distanz im Uhrzeigersinn* verkleinert wird
- bidirektionaler Ring: Weiterleitung, sodass *absolute Distanz* verkleinert wird
- in beiden Fällen: Pfadlänge $O(\frac{1}{k} \log^2 n)$
  aber: kleinerer konstanter Faktor im bidirektionalen Ring

Beschreiben Sie Symphonys Join-Protokoll!

Beitritt

Beitreten eines Knotens v.

1. Auswahl einer zufälligen Adresse $a \in [0, 1)$.
2. Suche des aktuell für a zuständigen Knotens (Routing-Protokoll).
3. Einordnen in den Ring zwischen a und dessen Vorgänger. (dabei: setzen deren und der eigenen SDLs)
4. Abschätzen der Anzahl an Knoten im Netzwerk (Estimation-Protokoll, $s = 3$).
5. Aufbauen der LDLs.

Beschreiben Sie Symphonys Leave-Protokoll!

Verlassen

Knoten v verlässt das Netzwerk.

1. Benachrichtigung aller Nachbarn (Short- und Long-Distance-).
2. Bisherige Short-Distance-Nachbarn von v verbinden sich.
   $\Rightarrow$ Aufrechterhaltung des Ringes
3. Bisherige Long-Distance-Nachbarn, die nun weniger als k Long-Distance-Nachbarn haben, wählen erneut zufällig welche aus.
4. Nachfolger von v führt das Estimation-Protokoll aus ($s = 3$).

Nennen Sie zwei Vorteile, die Symphony gegenüber anderen DHT-Ansätzen bietet!

Allgemeine Vorteile von Symphony

ORGANIC COMPUTING

- nur wenige Verbindungen müssen pro Knoten aufrechterhalten werden (*Low-State-Maintenance*)
- hohe Robustheit möglich
  - Daten-Redundanz einfach zu bewerkstelligen
  - Robustheit bereits nur durch SDLs (LDLs nur für Effizienz)
- gut sichtbarer Tradeoff zwischen Anzahl an Verbindungen (pro Knoten) und durchschnittlicher Suchdauer

Von welchem in der Vorlesung vorgestellten Netzwerkmodell ist Symphony inspiriert? Worin unterscheidet es sich?

Symphony als Kleinberg-Modell

ORGANIC COMPUTING

- inspiriert von Kleinberg-Modell
- im Kleinberg-Modell: *zweidimensionales* Gitter
  $\Rightarrow$ jeder Knoten vier Nachbarn, in Symphony zwei
- Symphony $\approx$ eindimensionales Kleinberg-Modell

Anmerkung zum Kleinberg-Modell

ORGANIC COMPUTING

Das Kleinberg-Modell sieht q Long-Distance-Links pro Knoten vor (Folie 3.30).

Warum ist die gewählte PDF (*Probability-Distribution-Function*) problematisch? Wie wird der resultierenden Problematik begegnet?

Problem mit der PDF

ORGANIC COMPUTING

- PDF ist gegeben als

$$p_n(x) = \begin{cases} \frac{1}{x \ln n}, & x \in [\frac{1}{n}, 1] \\ 0, & \text{sonst} \end{cases}$$

- Anzahl der Knoten im Netzwerk, n, muss bekannt sein
- aber: n ist den einzelnen Knoten unbekannt!

Estimation-Protokoll

ORGANIC COMPUTING

- Annahme: Knoten-Adressen gleichverteilt in $[0, 1)$
- betrachte s verschiedene Knoten mit Segmentlängen $l_1, \ldots, l_s$
- sei

$$X_s = \sum_{i \in \{1,\ldots,s\}} l_i$$

- dann ist $\frac{s}{X_s}$ ein guter Schätzwert für n, denn:

$$\frac{X_s}{X_{\text{Ring}}} \approx \frac{s}{n} \quad \Leftrightarrow \quad n \approx \frac{X_{Ring}}{X_s} s,$$

und da alle Segmente des Symphony-Rings immer zu 1 summieren, also $X_{\text{Ring}} = 1$:

$$n \approx \frac{s}{X_s}$$

Was soll mit dem (1-)Look-Ahead-Protokoll erreicht werden? Wie beeinflusst das die Performance von Symphony?

Look-Ahead-Protokoll von Symphony

ORGANIC COMPUTING

- Idee: Informationen über die Nachbarn der Nachbarn speichern (ohne Aufbau neuer Long-Distance-Links)
- Routing wird weniger *greedy*
- Latenz-Reduktion von bis zu 40 %
- Kosten für Look-Ahead-Listen beherrschbar: $O(k^2)$

Abschließende Anmerkung

ORGANIC COMPUTING

Achtet auf die Typen!

Wenn ihr v als *Knoten* definiert, dann ist v keine *Adresse*. Adresse von v könnte mit a_v o. ä. notiert werden.

Eine Datenadresse ist etwas anderes als die Daten, die dort liegen: Wenn Daten d, dann ist $a_d = (f \circ h)(d)$ die Adresse dieser Daten.

Terminausfall

- Der Termin nächste Woche, also am 05.12. fällt aus
- Lösungsvorschlag des 4. Blattes wird am 12.12. vorgestellt
- 5. Übungsblatt wird jetzt vorgestellt
- es hat eine Bearbeitungszeit bis 10.12.