Universität Augsburg
Institut für Informatik
Lehrstuhl für Organic Computing
Prof. Dr. Jörg Hähner

Ansprechpartner
Dominik Rauh, M. Sc.
dominik.rauh@informatik.uni-augsburg.de
Eichleitnerstr. 30, Raum 502

Wintersemester 2018/2019

Peer-to-Peer und Cloud Computing

Lösungsvorschläge zu Aufgabenblatt 3

Symphony: Grundlegendes (18 Punkte)

Lesen Sie den wissenschaftlichen Beitrag *Symphony: Distributed Hashing in a Small World* (im Digicampus verfügbar). Beantworten Sie dazu die folgenden Fragen.

1. Beschreiben Sie kurz die Eigenschaften und die Funktionsweise von Symphony:
   - grundlegender Aufbau (Netzwerkstruktur, Adressbereiche, Verbindungen, ...) (3 Punkte)

     Lösung

     – „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)
     – DHT: Daten werden gehasht nach $[0, 1)$
       * sei h eine „klassische" Hash-Funktion $h : \text{Data} \to \mathbb{B}^m$
       * es gilt $\mathbb{B}^m \simeq \mathbb{N}$
       * sei $f : \mathbb{N} \to \mathbb{R}$, $f(x) = x/2^m$
       * dann ist $f \circ h : \text{Data} \to [0, 1)$ die in Symphony genutzte Hash-Funktion
     – Knoten zuständig für Daten mit Schlüsseln im Bereich zwischen seiner und der Adresse seines *Vorgängers* im Uhrzeigersinn (siehe Abbildung 1)

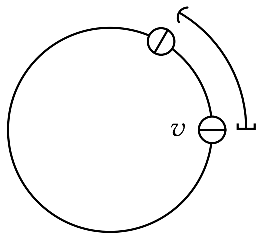

Abbildung 1: Zuständigkeit eines Knotens v.

- Wahl der Long-Distance-Links (1 Punkt)

  Lösung

  – Peer v_a mit Adresse a soll einen LDL aufbauen

    a) Auswahl eines zufälligen Offsets o verteilt nach

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

    b) z.B. mittels der Prozedur $e^{\ln n (rand() - 1)}$
    ($rand()$ gibt gleichverteilt Werte aus $[0, 1]$ zurück)

    c) Aufbau eines LDLs zu Peer v_{a+o}

  – gegeben für das Netzwerk: Anzahl k der LDLs pro Peer
  (mögliche Erweiterungen: k abhängig von Netzwerkgröße/Peerleistung/...)

  – Peers lehnen neue Verbindungen ab, wenn sie $\geq 2k$ Verbindungen haben
  $\Rightarrow$ erneute Auswahl eines zufälligen Offsets

- Routing-Protokoll (2 Punkte)

  Lösung

  Gegeben: Schlüssel x. Gesucht: Peer, der für die Daten d zuständig ist, für die

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

  – 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

- Join-Protokoll (2 Punkte)

  Lösung

  Beitreten eines Knotens v.

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

- Leave-Protokoll (2 Punkte)

  Lösung

  Knoten v verlässt das Netzwerk.

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

2. Nennen Sie zwei Vorteile, die Symphony gegenüber anderen DHT-Ansätzen bietet. (2 Punkte)

   Lösung

   - 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

3. Von welchem in der Vorlesung vorgestellten Netzwerkmodell ist Symphony inspiriert? Worin unterscheidet es sich? (2 Punkte)

   Lösung

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

4. Warum ist die gewählte PDF (*Probability Distribution Function*) problematisch? Wie wird der resultierenden Problematik begegnet? (2 Punkte)

   Lösung

   - PDF ist gegeben als
     $$p_n(x) = \begin{cases} \frac{1}{x \ln n}, & x \in [\frac{1}{n}, 1] \\ 0, & \text{sonst} \end{cases}$$
   - Problem: Anzahl der Knoten im Netzwerk, n, muss bekannt sein; *aber*: kein globales Wissen über alle Knoten!
   - Lösung: Estimation-Protokoll zum Abschätzen von n
     - 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 $X_{\text{Ring}} = 1$:
       $$n \approx \frac{s}{X_s}$$

5. Was soll mit dem (1-)Look-Ahead-Protokoll erreicht werden? Wie beeinflusst das die Performance von Symphony? (2 Punkte)

Lösung

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