Übung zu Peer-to-Peer und Cloud Computing

Übungstermin 06: Besprechung des Übungsblattes 04

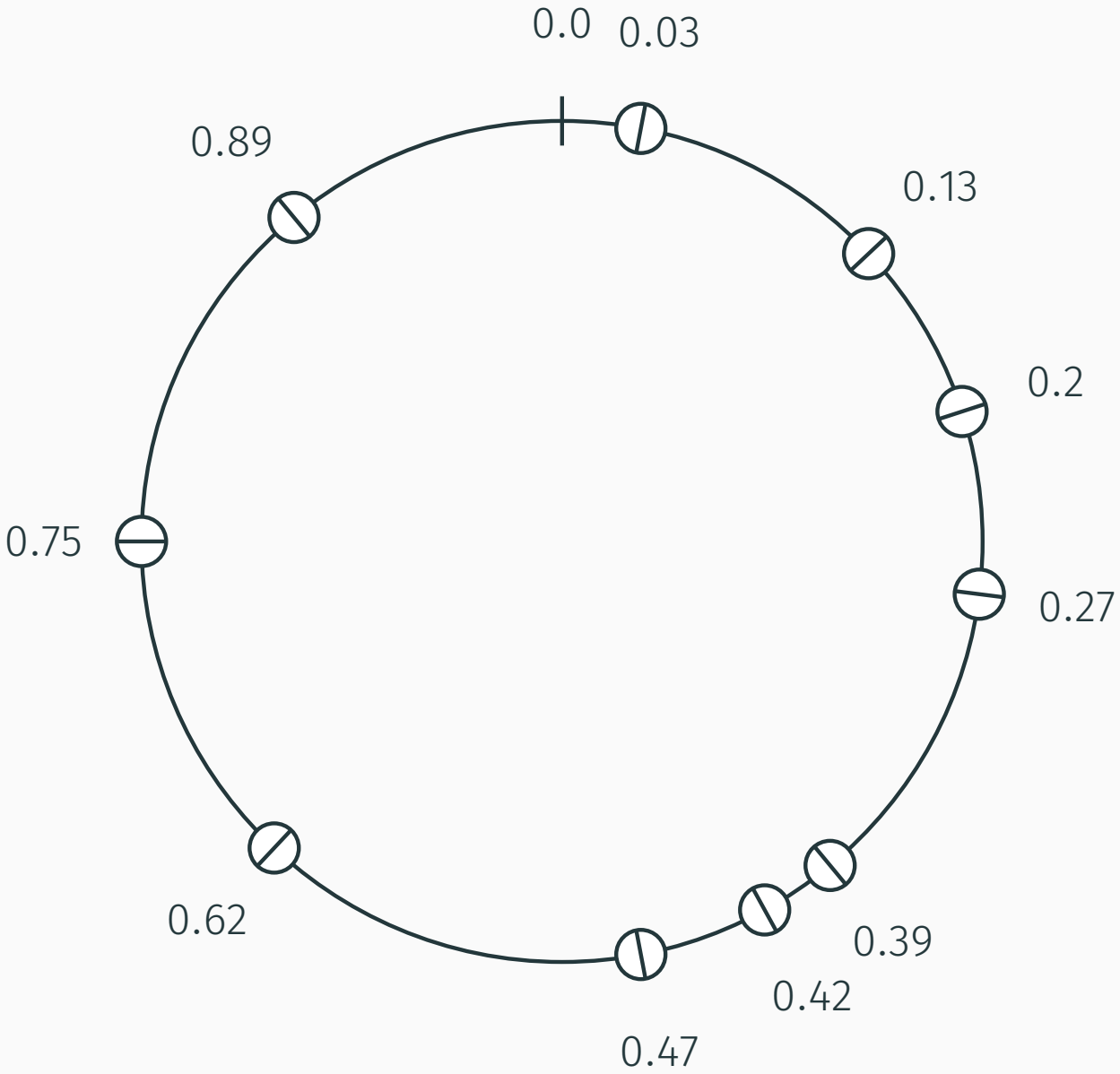
Dominik Rauh

12. Dezember 2018

Universität Augsburg
Institut für Informatik
Lehrstuhl für Organic Computing

Rechenaufgabe zu Symphony

Symphony-Netzwerk



Für welche Schlüssel ist Knoten $v_{0,42}$ zuständig?

Zuständigkeit von $V_{0,42}$

ORGANIC COMPUTING

- für alle Daten d mit $0{,}39 < (f \circ h)(d) \leq 0{,}42$
- $f \circ h$ ist die in Symphony verwendete Hashfunktion

Berechnen Sie für den Knoten $v_{0,47}$ den Schätzwert für die Anzahl von Knoten im Netz (mit $s = 3$).

Estimation-Protokoll

ORGANIC COMPUTING

$$
\begin{aligned}
\tilde{n} &= \frac{s}{X_{S_{0,47}}} \\
&= \frac{s}{\sum_{i \in 0,42;0,47;0,62} l_i} \\
&= \frac{3}{0,42 - 0,39 + 0,47 - 0,42 + 0,62 - 0,47} \\
&= \frac{3}{0,23} \approx 13,04
\end{aligned}
$$

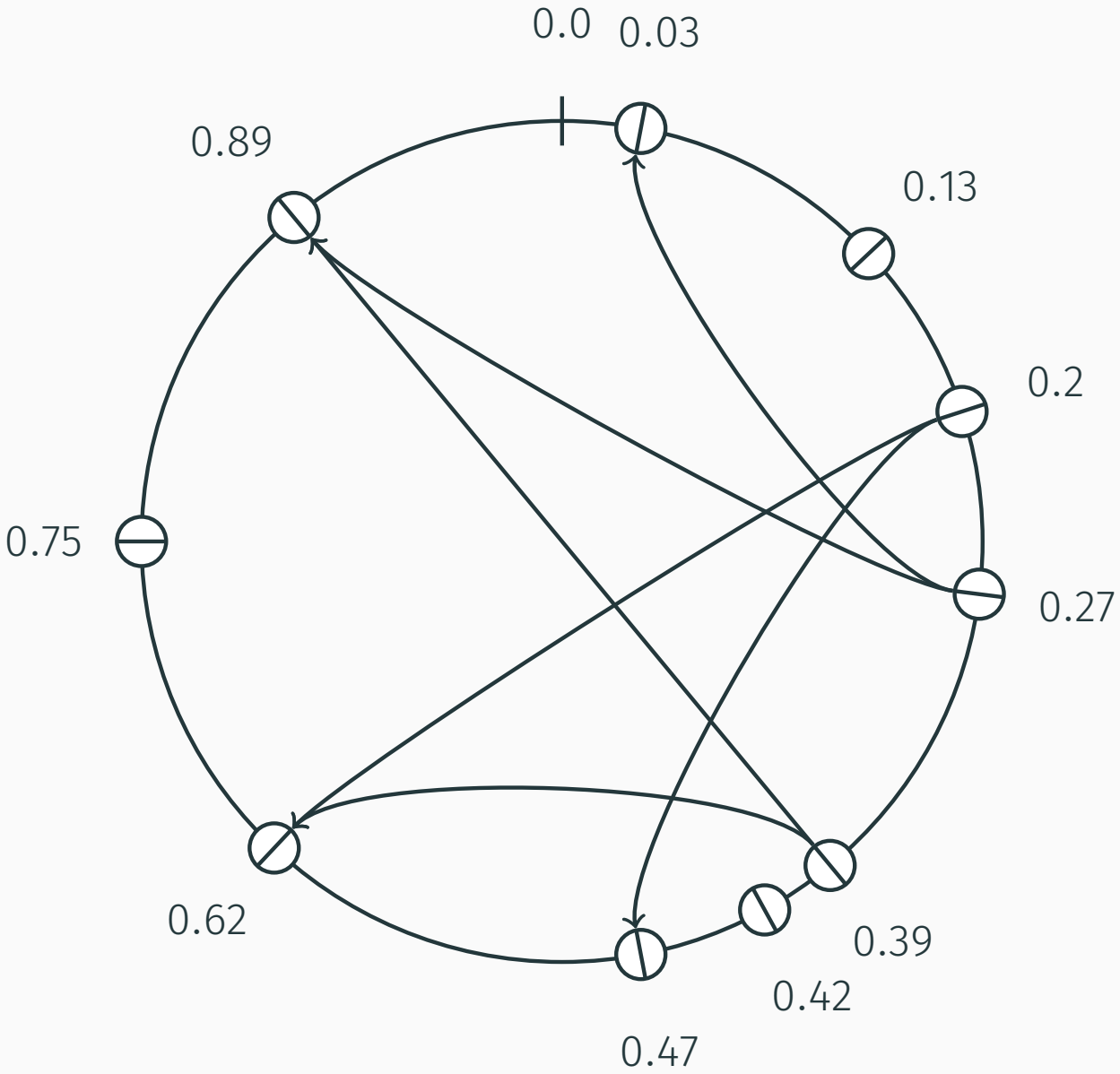
Zeichnen Sie die Long-Distance-Links unter Benutzung der gegebenen „Zufallszahlen“ ein.

Berechnung der LDL-Ziele

ORGANIC COMPUTING

$e^{\ln n * (rand() - 1{,}0)}$	genutzt von Peer	$\Rightarrow$ LDL zu Peer
0,41	0,2	0,62
0,23	0,2	0,47
0,51	0,27	0,89
0,67	0,27	0,03
0,17	0,39	0,62
0,37	0,39	0,89

LDLs



Knoten $v_{0,13}$ fordert Daten mit dem Schlüssel $0{,}88$ an. Geben Sie den vollständigen Anfragepfad an und begründen Sie ihn.

Annahme: Bidirektionaler Ring.

Suche nach Schlüssel 0,88 bei $v_{0,13}$

ORGANIC COMPUTING

$v_{0,13}$ sucht Nachbarn mit minimaler Distanz zu 0,88.

- wenn LDL zu $v_{0,89}$ $\Rightarrow$ Suche dorthin weitergeben
- aber: nichts über LDLs bekannt $\Rightarrow$ Annahme: kein LDL
- analog für $v_{0,75}$
- also: nur SDLs $\{v_{0,03}; v_{0,2}\}$ betrachten
- es gilt:

$$\underset{v_k \in \{v_{0,03}; v_{0,2}\}}{\textbf{arg min}} \; (min(|k - 0{,}88|; 1 - |k - 0{,}88|)) = v_{0,03}$$

$\Rightarrow$ Weiterleitung an $v_{0,03}$

Suche nach Schlüssel 0,88 bei $v_{0,03}$

ORGANIC COMPUTING

$v_{0,03}$ sucht Nachbarn mit minimaler Distanz zu 0,88. Es gilt:

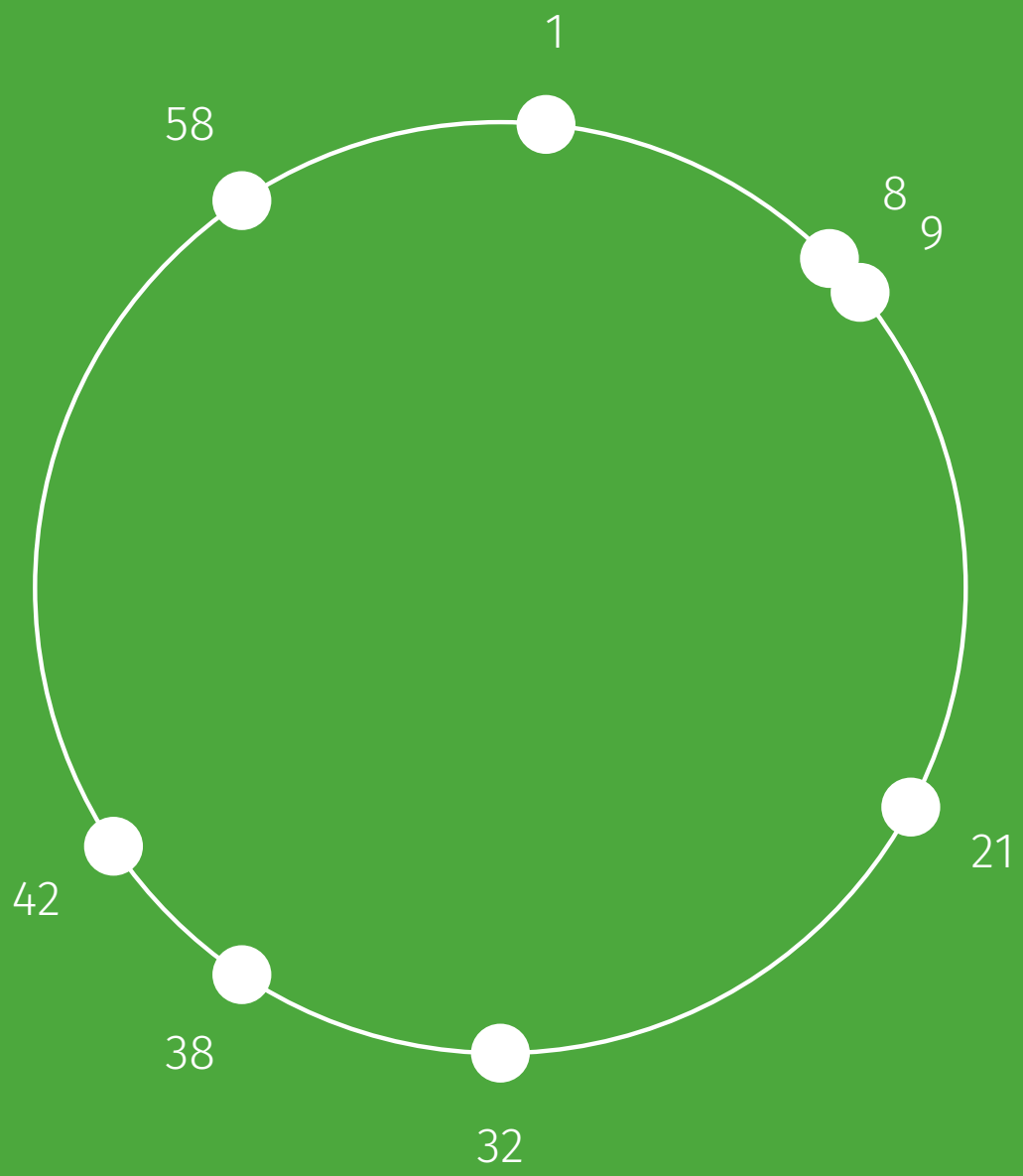
$$\underset{v_k \in \{v_{0,13}; v_{0.89}\}}{\textbf{arg min}} \; (min(|k - 0{,}88|; 1 - |k - 0{,}88|)) = v_{0,89}$$

$\Rightarrow$ LDLs spielen keine Rolle, auf jeden Fall Weiterleitung an $v_{0,89}$

Suche nach Schlüssel 0,88 bei $v_{0,89}$

$v_{0,89}$ ist Ziel der Suche, denn zuständig für $(0{,}75; 0{,}89]$.

Rechenaufgabe zu Chord


1
58
8
9
21
42
38
32

Erstellen Sie die Routingtabellen.

Routing- vs. Fingertabelle

ORGANIC COMPUTING

- Routingtabellen vereinfacht
  - im Allgemeinen: Zuordnung Ziel $\rightarrow$ Nachbar
  - z.B. Internet: Subnetz $\rightarrow$ Next hop
  - in P2P-Netzwerken: Zuordnung (Peer-)Adresse $\rightarrow$ (Nachbar-)Peer
  - nicht Teil der Routingtabelle: Auflösungstabelle für (Nachbar-)Peers mit Zuordnung Overlay-Adresse $\rightarrow$ Underlay-Adresse
- in Chord
  - Fingertabelle nur Teil der Routingtabelle
  - außerdem: Verbindungen zu Vor- und Nachfolger im Ring

Knoten v_1

		Zieladr.
$1 + 2^0 = 2$	$\Rightarrow$	8
$1 + 2^1 = 3$	$\Rightarrow$	8
$1 + 2^2 = 5$	$\Rightarrow$	8
$1 + 2^3 = 9$	$\Rightarrow$	9
$1 + 2^4 = 17$	$\Rightarrow$	21
$1 + 2^5 = 33$	$\Rightarrow$	38

Knoten v_8

		Zieladr.
$8 + 2^0 = 9$	$\Rightarrow$	9
$8 + 2^1 = 10$	$\Rightarrow$	21
$8 + 2^2 = 12$	$\Rightarrow$	21
$8 + 2^3 = 16$	$\Rightarrow$	21
$8 + 2^4 = 24$	$\Rightarrow$	32
$8 + 2^5 = 40$	$\Rightarrow$	42

Knoten v_9

		Zieladr.
$9 + 2^0 = 10$	$\Rightarrow$	21
$9 + 2^1 = 11$	$\Rightarrow$	21
$9 + 2^2 = 13$	$\Rightarrow$	21
$9 + 2^3 = 17$	$\Rightarrow$	21
$9 + 2^4 = 25$	$\Rightarrow$	32
$9 + 2^5 = 41$	$\Rightarrow$	42

Knoten v_{21}

		Zieladr.
$21 + 2^0 = 22$	$\Rightarrow$	32
$21 + 2^1 = 23$	$\Rightarrow$	32
$21 + 2^2 = 25$	$\Rightarrow$	32
$21 + 2^3 = 29$	$\Rightarrow$	32
$21 + 2^4 = 37$	$\Rightarrow$	38
$21 + 2^5 = 53$	$\Rightarrow$	58

Knoten v_{32}

		Zieladr.
$32 + 2^0 = 33$	$\Rightarrow$	38
$32 + 2^1 = 34$	$\Rightarrow$	38
$32 + 2^2 = 36$	$\Rightarrow$	38
$32 + 2^3 = 40$	$\Rightarrow$	42
$32 + 2^4 = 48$	$\Rightarrow$	58
$32 + 2^5 = 64$	$\Rightarrow$	1

Knoten v_{38}

		Zieladr.
$38 + 2^0 = 39$	$\Rightarrow$	42
$38 + 2^1 = 40$	$\Rightarrow$	42
$38 + 2^2 = 42$	$\Rightarrow$	42
$38 + 2^3 = 46$	$\Rightarrow$	58
$38 + 2^4 = 54$	$\Rightarrow$	58
$38 + 2^5 = 70$	$\Rightarrow$	8

Knoten v_{42}

		Zieladr.
$42 + 2^0 = 43$	$\Rightarrow$	58
$42 + 2^1 = 44$	$\Rightarrow$	58
$42 + 2^2 = 46$	$\Rightarrow$	58
$42 + 2^3 = 50$	$\Rightarrow$	58
$42 + 2^4 = 58$	$\Rightarrow$	58
$42 + 2^5 = 74$	$\Rightarrow$	21

Knoten v_{58}

		Zieladr.
$58 + 2^0 = 59$	$\Rightarrow$	1
$58 + 2^1 = 60$	$\Rightarrow$	1
$58 + 2^2 = 62$	$\Rightarrow$	1
$58 + 2^3 = 66$	$\Rightarrow$	8
$58 + 2^4 = 74$	$\Rightarrow$	21
$58 + 2^5 = 90$	$\Rightarrow$	32

Knoten mit ID 41 nimmt Kontakt mit Knoten v_1 auf, um ins Netzwerk aufgenommen zu werden.

Welche Schritte werden unternommen bis der neue Knoten Teil des Netzwerks ist?

1. Finden der Position des neuen Knotens

ORGANIC COMPUTING

1. Aufnahme-Anfrage an Knoten v_1.
2. v_1 sucht nach Knoten, der für Adresse 41 zuständig.
   - 2.1 v_1 checkt, ob sein Nachfolger, v_8, für 41 zuständig: Nein!
     Achtung: Dazu muss er *keine* Nachricht an v_8 schicken!
   - 2.2 v_1 schaut in seiner Fingertabelle nach, findet v_{38} als „Nächstkleineren“.
     $\Rightarrow$ Anfrage an v_{38}
3. v_{38} sucht nach Knoten, der für Adresse 41 zuständig.
   - 3.1 v_{38} checkt, ob sein Nachfolger, v_{42}, für 41 zuständig: Ja!

2. Verbinden mit dem Netzwerk

ORGANIC COMPUTING

1. v_{41} verbindet sich mit v_{42} und v_{38}.
2. v_{41} erstellt seine eigene Routingtabelle.
3. v_{41} benachrichtigt Knoten, die ihre Routingeinträge kontrollieren müssen.
4. Eventuell: Daten, für die v_{41} zuständig ist, werden an ihn übertragen.

Geben Sie die neue Routingtabelle für Knoten v_{41} an.

Routing-Tabelle von v_{41}

Fingertabelle:

		Zieladr.
$41 + 2^0 = 42$	$\Rightarrow$	42
$41 + 2^1 = 43$	$\Rightarrow$	58
$41 + 2^2 = 45$	$\Rightarrow$	58
$41 + 2^3 = 49$	$\Rightarrow$	58
$41 + 2^4 = 57$	$\Rightarrow$	58
$41 + 2^5 = 73$	$\Rightarrow$	9

Außerdem in Routing-Tabelle: v_{38}, v_{42}.

Welche Knoten müssen von v_{41} dazu aufgefordert werden, ihre Routingtabellen zu aktualisieren?

Update-Tabelle von v_{41}

Sei $f \equiv$ findPredecessor.

		Zieladr.
$41 - 2^0 = 40$	$\xrightarrow{f}$	38
$41 - 2^1 = 39$	$\xrightarrow{f}$	38
$41 - 2^2 = 37$	$\xrightarrow{f}$	32
$41 - 2^3 = 33$	$\xrightarrow{f}$	32
$41 - 2^4 = 25$	$\xrightarrow{f}$	21
$41 - 2^5 = 9$	$\xrightarrow{f}$	9

Geben Sie die aktualisierten Routingtabellen der anderen Knoten an.

Aktualisierte Routingtabellen

ORGANIC COMPUTING

- v_{42} setzt seinen Vorgänger auf v_{41}
- v_{38} setzt seinen Nachfolger auf v_{41}
- v_{21} muss keine Änderungen vornehmen
- neue Fingertabellen der Knoten v_9, v_{32} und v_{38}:

		Zieladr.
$9 + 2^0 = 10$	$\Rightarrow$	21
$9 + 2^1 = 11$	$\Rightarrow$	21
$9 + 2^2 = 13$	$\Rightarrow$	21
$9 + 2^3 = 17$	$\Rightarrow$	21
$9 + 2^4 = 25$	$\Rightarrow$	32
$9 + 2^5 = 41$	$\Rightarrow$	41

		Zieladr.
$32 + 2^0 = 33$	$\Rightarrow$	38
$32 + 2^1 = 34$	$\Rightarrow$	38
$32 + 2^2 = 36$	$\Rightarrow$	38
$32 + 2^3 = 40$	$\Rightarrow$	41
$32 + 2^4 = 48$	$\Rightarrow$	58
$32 + 2^5 = 64$	$\Rightarrow$	1

		Zieladr.
$38 + 2^0 = 39$	$\Rightarrow$	41
$38 + 2^1 = 40$	$\Rightarrow$	41
$38 + 2^2 = 42$	$\Rightarrow$	42
$38 + 2^3 = 46$	$\Rightarrow$	58
$38 + 2^4 = 54$	$\Rightarrow$	58
$38 + 2^5 = 70$	$\Rightarrow$	8

Aktualisierte Routingtabellen

ORGANIC COMPUTING

- v_9, v_{32} und v_{38} fordern jeweils ihren Vorgänger zum Kontrollieren der Routingtabellen auf
- Führt zu Update der Routingtabelle von v_8 über v_9:

		Zieladr.
$8 + 2^0 = 9$	$\Rightarrow$	9
$8 + 2^1 = 10$	$\Rightarrow$	21
$8 + 2^2 = 12$	$\Rightarrow$	21
$8 + 2^3 = 16$	$\Rightarrow$	21
$8 + 2^4 = 24$	$\Rightarrow$	32
$8 + 2^5 = 40$	$\Rightarrow$	41

Knoten v_{41} fordert Daten mit dem Schlüssel 9 an.

Welche Knoten werden von welchen Knoten in welcher Reihenfolge nach den Daten gefragt?

Ablauf der Suche nach Schlüssel 9

ORGANIC COMPUTING

1. v_{41} sucht nach Knoten, der für Adresse 9 zuständig.
   - 1.1 v_{41} checkt, ob sein Nachfolger, v_{42}, für 9 zuständig: Nein!
     Achtung: Dazu muss er *keine* Nachricht an v_{42} schicken!
   - 1.2 v_{41} schaut in seiner Fingertabelle nach, findet v_{58} als „Nächstkleineren“ nach $9 = 73 \mod 64$.
     $\Rightarrow$ Anfrage an v_{58}
2. v_{58} sucht nach Knoten, der für Adresse 9 zuständig.
   - 2.1 v_{58} checkt, ob sein Nachfolger, v_1, für 9 zuständig: Nein!
   - 2.2 v_{58} schaut in seiner Fingertabelle nach, findet v_8.
     $\Rightarrow$ Anfrage an v_8
3. v_8 sucht nach Knoten, der für Adresse 9 zuständig.
   - 3.1 v_8 checkt, ob sein Nachfolger, v_9, für 9 zuständig: Ja!
   - 3.2 v_8 liefert Kontaktdaten von v_9 an v_{41} zurück.

Welcher Knoten liefert schließlich das Ergebnis der Suche an v_{41} zurück?

Wer liefert das Ergebnis der Suche zurück?

v_8!

Knoten v_{21} fällt aus. Welche Knoten aktualisieren nun unmittelbar welche Informationen?

- Knoten bemerken über Keep-Alive-Nachrichten den Ausfall von v_{21}
- v_{32} löscht Verbindung zu v_{21} (seinem Vorgänger)
- v_9 löscht Verbindung zu v_{21} (seinem Nachfolger), setzt v_{32} als neuen Nachfolger
  $\Rightarrow$ Ringstruktur erhalten
- nicht unmittelbar, aber periodisch: Stabilisation-Protokoll repariert den Rest

Anhang

Erinnerung: Hashing nach [0,1)

- 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