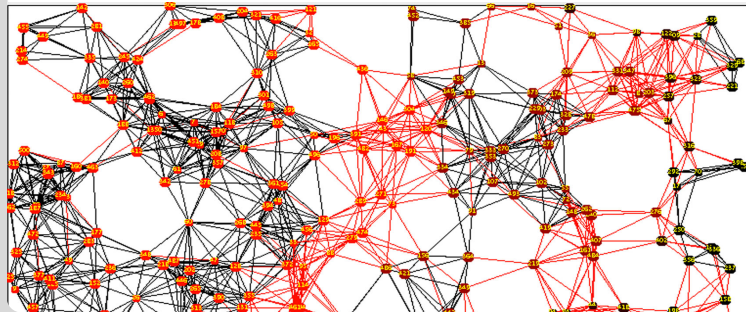


# Algorithmen für Ad-hoc- und Sensornetze

## Übung 1 – Leader Election

Fabian Fuchs | 27. Oktober 2015 (Version 1)

INSTITUT FÜR THEORETISCHE INFORMATIK - LEHRSTUHL FÜR ALGORITHMIK (PROF. WAGNER)





Simulation Control




Round: 7

Rounds to do:

Refresh rate:

View

Output

- Organisatorisches
  
- Implementierung der Algorithmen
  - Sinalgo Demo
  
- Leader Election in allgemeinen Graphen (wdh.)

## ■ Übung

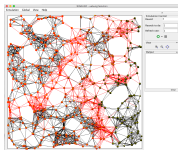
- Es sollen ausgewählte Algorithmen im Netzwerksimulator Sinalgo implementiert werden
- Heute: Übungsblatt zum warm werden
- Von den folgenden Übungsblättern soll mindestens eins in der Übung (teilweise) vorgestellt werden um zur Prüfung zugelassen zu werden.
- Übungsblätter auch digital unter: <http://illwww.iti.kit.edu/teaching/winter2015/sensornetze/index>

## ■ Bei Fragen: Email oder Sprechstunde

- Dienstag: 13:00-14:00 (kurze Anmeldung erwünscht!)
- Oder nach Vereinbarung: [fabian.fuchs@kit.edu](mailto:fabian.fuchs@kit.edu)
- Raum 317, Gebäude 50.34

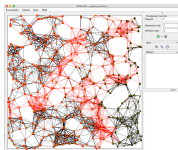
- Organisatorisches
- Implementierung der Algorithmen
  - Sinalgo Demo
- Leader Election in allgemeinen Graphen (wdh.)

- Sinalgo: Simulator für Netzwerkalgorithmen in (Drahtlos-)Netzwerken
  - Implementiert in Java, dadurch plattformübergreifend nutzbar
  - Modularer Aufbau ermöglicht parallele Projekte
  - Nachteil: Implementierung grundlegend anders als auf Sensorknoten



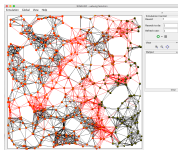
- Arduino Uno: Hands-on Sensorknoten
  - Komplexere (bzw. oftmals ungewohnte) Umgebung, daher mehr Focus auf Kennenlernen der Plattform, weniger auf Algorithmen.
  - Implementierung in leicht erweitertem C
- Aufteilung: 5 Blätter zu Sinalgo, ein Blatt zu Arduino

- Sinalgo: Simulator für Netzwerkalgorithmen in (Drahtlos-)Netzwerken
  - Implementiert in Java, dadurch plattformübergreifend nutzbar
  - Modularer Aufbau ermöglicht parallele Projekte
  - Nachteil: Implementierung grundlegend anders als auf Sensorknoten
- Arduino Uno: Hands-on Sensorknoten
  - Komplexere (bzw. oftmals ungewohnte) Umgebung, daher mehr Focus auf Kennenlernen der Plattform, weniger auf Algorithmen.
  - Implementierung in leicht erweitertem C



- Aufteilung: 5 Blätter zu Sinalgo, ein Blatt zu Arduino

- Sinalgo: Simulator für Netzwerkalgorithmen in (Drahtlos-)Netzwerken
  - Implementiert in Java, dadurch plattformübergreifend nutzbar
  - Modularer Aufbau ermöglicht parallele Projekte
  - Nachteil: Implementierung grundlegend anders als auf Sensorknoten
- Arduino Uno: Hands-on Sensorknoten
  - Komplexere (bzw. oftmals ungewohnte) Umgebung, daher mehr Focus auf Kennenlernen der Plattform, weniger auf Algorithmen.
  - Implementierung in leicht erweitertem C
- Aufteilung: 5 Blätter zu Sinalgo, ein Blatt zu Arduino



- Demonstration
  - Vom Download zur Ausführung
  - Erste Beispiele: Sample x und y
  - Wie binde ich die verschiedenen Modelle ein?
  - Wie setze ich spezifische Parameter?
- Weitere Fragen?
  - Werden nach der Übung online aktualisiert



- Organisatorisches
- Implementierung der Algorithmen
  - Sinalgo Demo
- Leader Election in allgemeinen Graphen (wdh.)

## Erinnerung: Leader Election

**Gegeben:** Symmetrischer, zusammenhängender Kommunikationsgraph  $G$ .

**Problem:** Genau ein Prozessor soll als *Leader* ausgezeichnet werden.

## Modell

- Synchroner Runden, synchroner Start
- Pro Runde kann jeder Knoten eine (potentiell unterschiedliche) Nachricht an jeden Nachbarn senden
- Verteilung: Random, Grid, Manuell, ...<sup>a</sup>
- Weiteres: UGD, keine Mobilität, keine Interference, reliable delivery, ...

---

<sup>a</sup>Bei mehreren Zusammenhangskomponenten wird jeweils ein Leader bestimmt.

# Leader Election in allg. Graphen

Leader Election, Knoten  $p_i$  kennt  $ID_i$

sende  $ID_i$  an alle Nachbarn

setze  $ID_+ \leftarrow ID_i$  und  $parent \leftarrow \perp$

**wenn**  $ID_s$  empfangen wurden **dann**

    wähle maximale empfangene  $ID$  und zugehörigen Sender  $s$

**wenn**  $ID > ID_+$  **dann**

        setze  $ID_+ \leftarrow ID$  und  $parent \leftarrow s$

        sende Nachricht „new follower“ an  $parent$

        sende  $ID_+$  an restliche Nachbarn

**wenn** „new follower“ empfangen wurde und  $parent \neq \perp$  **dann**

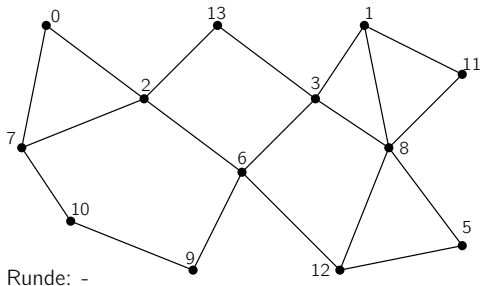
    sende Nachricht „new follower“ an  $parent^a$

---

<sup>a</sup>außer in selber Runde schon geschehen

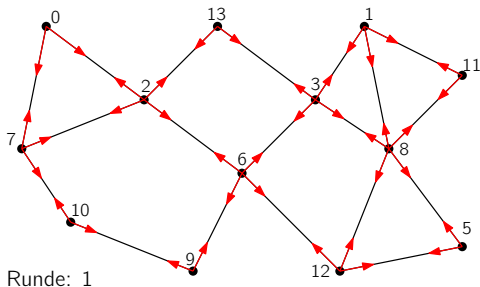
Ein Knoten mit  $parent = \perp$  darf sich zum Leader erklären nachdem zwei Runden keine höhere  $ID$  und keine new follower-Nachricht kam. Warum?

# Leader Election in allg. Graphen



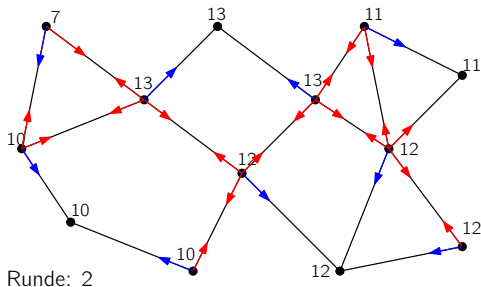
- Knotenlabels sind höchste empfangene ID (zu Beginn eigene ID)
- Knoten senden gespeicherte ID (rot), hier: Initiale Nachricht
- Knoten die eine höhere ID empfangen, leiten Sie an Nachbarn weiter, setzen parent und senden "new follower" (blau) an den parent Knoten.
- Wird eine "new follower" Nachricht empfangen wird diese an parent weitergeleitet. (Schwarze Pfeile deuten parent an, keine Nachricht!)

# Leader Election in allg. Graphen



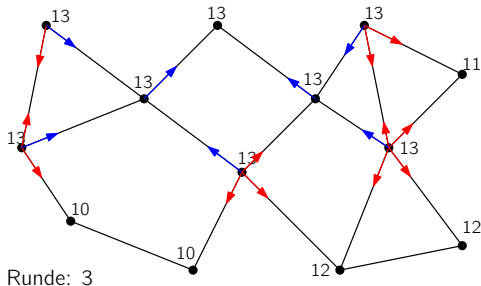
- Knotenlabels sind höchste empfangene ID (zu Beginn eigene ID)
- Knoten senden gespeicherte ID (rot), hier: Initiale Nachricht
- Knoten die eine höhere ID empfangen, leiten Sie an Nachbarn weiter, setzen parent und senden "new follower" (blau) an den parent Knoten.
- Wird eine "new follower" Nachricht empfangen wird diese an parent weitergeleitet. (Schwarze Pfeile deuten parent an, keine Nachricht)

# Leader Election in allg. Graphen



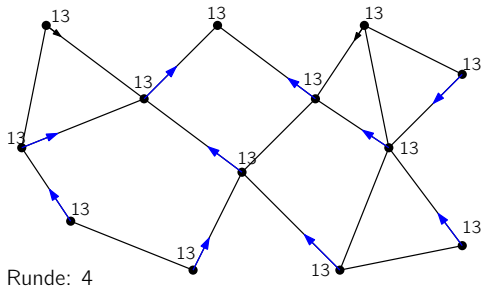
- Knotenlabels sind höchste empfangene ID (zu Beginn eigene ID)
- Knoten senden gespeicherte ID (rot), hier: Initiale Nachricht
- Knoten die eine höhere ID empfangen, leiten Sie an Nachbarn weiter, setzen parent und senden "new follower" (blau) an den parent Knoten.
- Wird eine "new follower" Nachricht empfangen wird diese an parent weitergeleitet. (Schwarze Pfeile deuten parent an, keine Nachricht)

# Leader Election in allg. Graphen



- Knotenlabels sind höchste empfangene ID (zu Beginn eigene ID)
- Knoten senden gespeicherte ID (rot), hier: Initiale Nachricht
- Knoten die eine höhere ID empfangen, leiten Sie an Nachbarn weiter, setzen parent und senden "new follower" (blau) an den parent Knoten.
- Wird eine "new follower" Nachricht empfangen wird diese an parent weitergeleitet. (Schwarze Pfeile deuten parent an, keine Nachricht!)

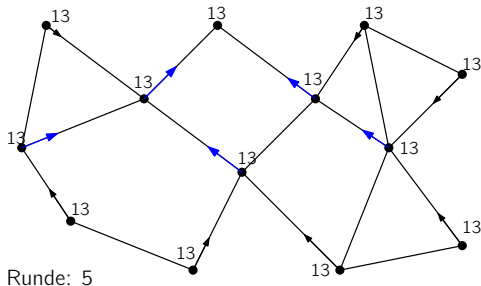
# Leader Election in allg. Graphen



- Knotenlabels sind höchste empfangene ID (zu Beginn eigene ID)
- Knoten senden gespeicherte ID (rot), hier: Initiale Nachricht
- Knoten die eine höhere ID empfangen, leiten Sie an Nachbarn weiter, setzen parent und senden "new follower" (blau) an den parent Knoten.
- Wird eine "new follower" Nachricht empfangen wird diese an parent weitergeleitet. (Schwarze Pfeile deuten parent an, keine Nachricht!)

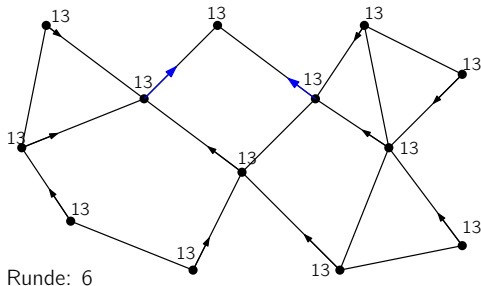


# Leader Election in allg. Graphen



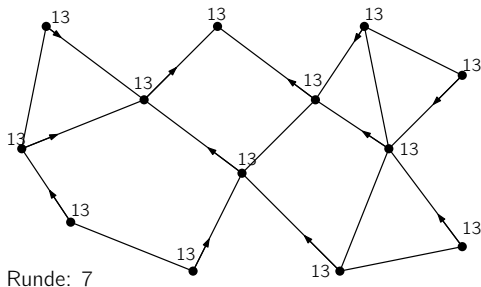
- Knotenlabels sind höchste empfangene ID (zu Beginn eigene ID)
- Knoten senden gespeicherte ID (rot), hier: Initiale Nachricht
- Knoten die eine höhere ID empfangen, leiten Sie an Nachbarn weiter, setzen parent und senden "new follower" (blau) an den parent Knoten.
- Wird eine "new follower" Nachricht empfangen wird diese an parent weitergeleitet. (Schwarze Pfeile deuten parent an, **keine** Nachricht!)

# Leader Election in allg. Graphen



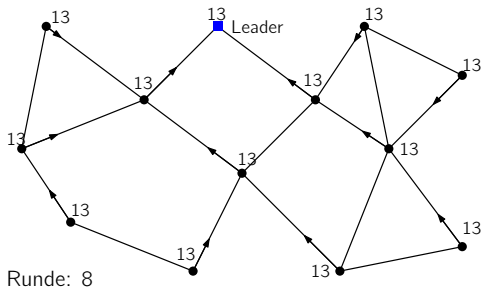
- Knotenlabels sind höchste empfangene ID (zu Beginn eigene ID)
- Knoten senden gespeicherte ID (rot), hier: Initiale Nachricht
- Knoten die eine höhere ID empfangen, leiten Sie an Nachbarn weiter, setzen parent und senden "new follower" (blau) an den parent Knoten.
- Wird eine "new follower" Nachricht empfangen wird diese an parent weitergeleitet. (Schwarze Pfeile deuten parent an, **keine** Nachricht!)

# Leader Election in allg. Graphen



- Knotenlabels sind höchste empfangene ID (zu Beginn eigene ID)
- Knoten senden gespeicherte ID (rot), hier: Initiale Nachricht
- Knoten die eine höhere ID empfangen, leiten Sie an Nachbarn weiter, setzen parent und senden "new follower" (blau) an den parent Knoten.
- Wird eine "new follower" Nachricht empfangen wird diese an parent weitergeleitet. (Schwarze Pfeile deuten parent an, **keine** Nachricht!)

# Leader Election in allg. Graphen



- Knotenlabels sind höchste empfangene ID (zu Beginn eigene ID)
- Knoten senden gespeicherte ID (rot), hier: Initiale Nachricht
- Knoten die eine höhere ID empfangen, leiten Sie an Nachbarn weiter, setzen parent und senden "new follower" (blau) an den parent Knoten.
- Wird eine "new follower" Nachricht empfangen wird diese an parent weitergeleitet. (Schwarze Pfeile deuten parent an, **keine** Nachricht!)

Wir hatten heute:

- Organisatorisches zur Übung
- Implementierung für Sinalgo bzw. Arduino
- Demo Sinalgo Framework
- Wiederholung: Leader Election für allgemeine Graphen

Weitere Fragen?

- Übungsblatt auf VL-Homepage:  
<http://illwww.iti.uni-karlsruhe.de/teaching/winter2015/sensornetze/index>
- Sinalgo Tutorial: <http://disco.ethz.ch/projects/sinalgo/tutorial/Documentation.html>
- Sinalgo Framework: VL-Homepage oder Sinalgo Tutorial Seite
- Grundgerüst für Leader Election Projekt: VL-Homepage
- Tipp: Wer schon mit Arduino rumspielen möchte:  
<https://123d.circuits.io/>