Algorithmen für offene verteilte Systeme

Stefan Schmid

TECHNISCHE UNIVERSITÄT MÜNCHEN

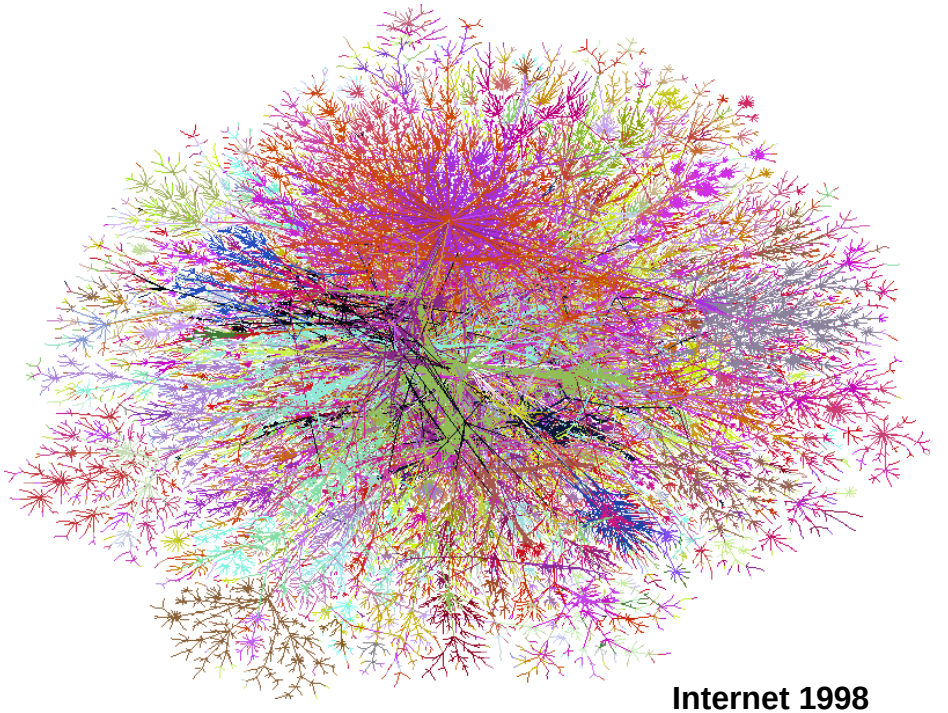
Kolloquium

Universität Karlsruhe (TH)

Verteilte Systeme (1)

- Das Internet läutete Paradigmenwechsel ein
  - Technologie um Millionen von Maschinen zu vernetzen
  - Erfolgstories: WWW, Emailsystem, usw.
  - Grosser Einfluss auf unsere Gesellschaft

- Heute: ubiquitärer Internetzugriff und tiefe Bandbreitenkosten
  - Trend zu neuen Anwendungen
  - Z.B. peer-to-peer File Sharing
  - Z.B. Social Networking Tools
  - Z.B. Streaming Anwendungen
  - Z.B. Computational Grids

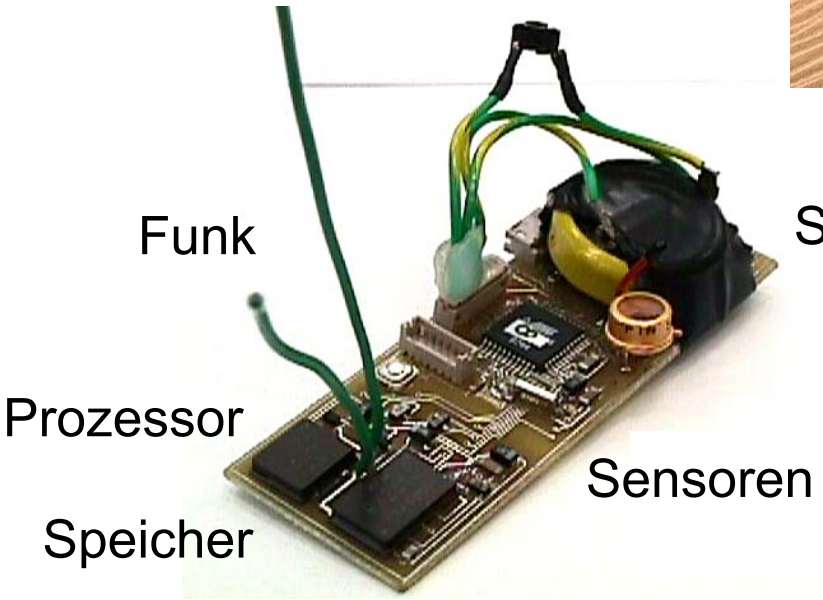


Internet 1998

© wired magazine

Verteilte Systeme (2)

- Distributed Computing ist nicht auf Internet Anwendungen beschränkt
  - Drahtlose Sensor Netzwerke
  - Multicore Computer



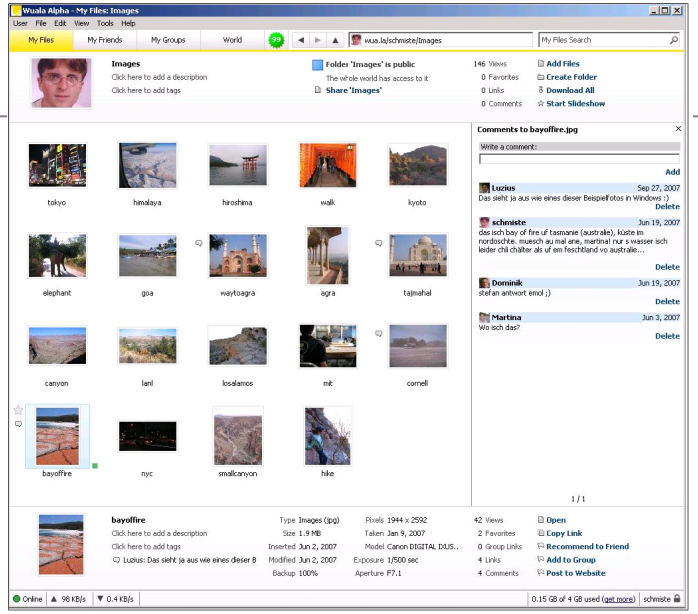
Verteilte Systeme (3)

Trotz dem Erfolg dieser Anwendungen verstehen wir viele Aspekte von heutigen verteilten Systemen noch nicht gut.

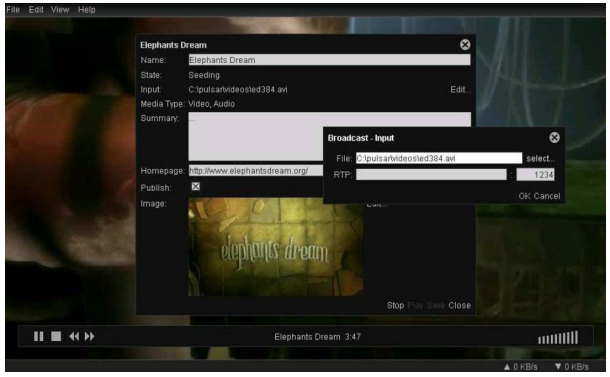
Dieser Vortrag:

- Fokus auf Herausforderungen von offenen verteilten Systemen
  - offen: „beliebige“ (Protokoll konforme) Knoten können teilnehmen
  - dynamisch, heterogen, autonome Teilnehmer
  - Beispiel: Grid Computing, P2P Computing, drahtlose ad-hoc Netzwerke
- Insbesondere: Dynamik und Kooperation in Peer-to-Peer Netzwerken
- Aktuelle (eigene) Forschungsresultate

Peer-to-Peer Technologie



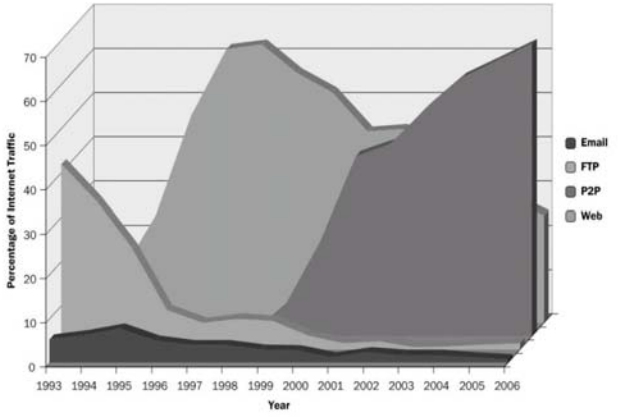
- Bekannte P2P Systeme
  - Internet Telephonie: *Skype*
  - File Sharing: *BitTorrent, eMule, ...*
  - Streaming: *Zattoo, Joost, ...*



- Andere (bekannte?) Systeme
  - Pulsar Streaming Systeme (z.B. *tilllate* clips?)
  - Wuala Onlinespeicher System

- Hauptanteil an Internet Verkehr!

  (Quelle: *cachelogic.com*)

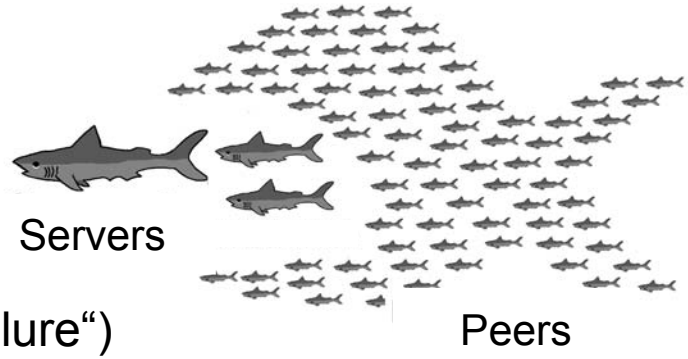


„*Hot Topic*“

- Thema in Gesellschaft / Presse / Blogs
  - Ueber Start-ups wird berichtet (z.B. Studentenprojekt Wuala, ~70,000 Users)
  - Neue Clients diskutiert in Blogs (z.B. BitThief)

Das Paradigma

- Schlüsselkonzepte
  - Maschine (Peer) im Netzwerk: Konsument und Produzent von Ressourcen („alle Maschinen sind gleich“)
  - Nutzung von dezentralisierten Ressourcen am Rande des Internet (z.B. Desktops)



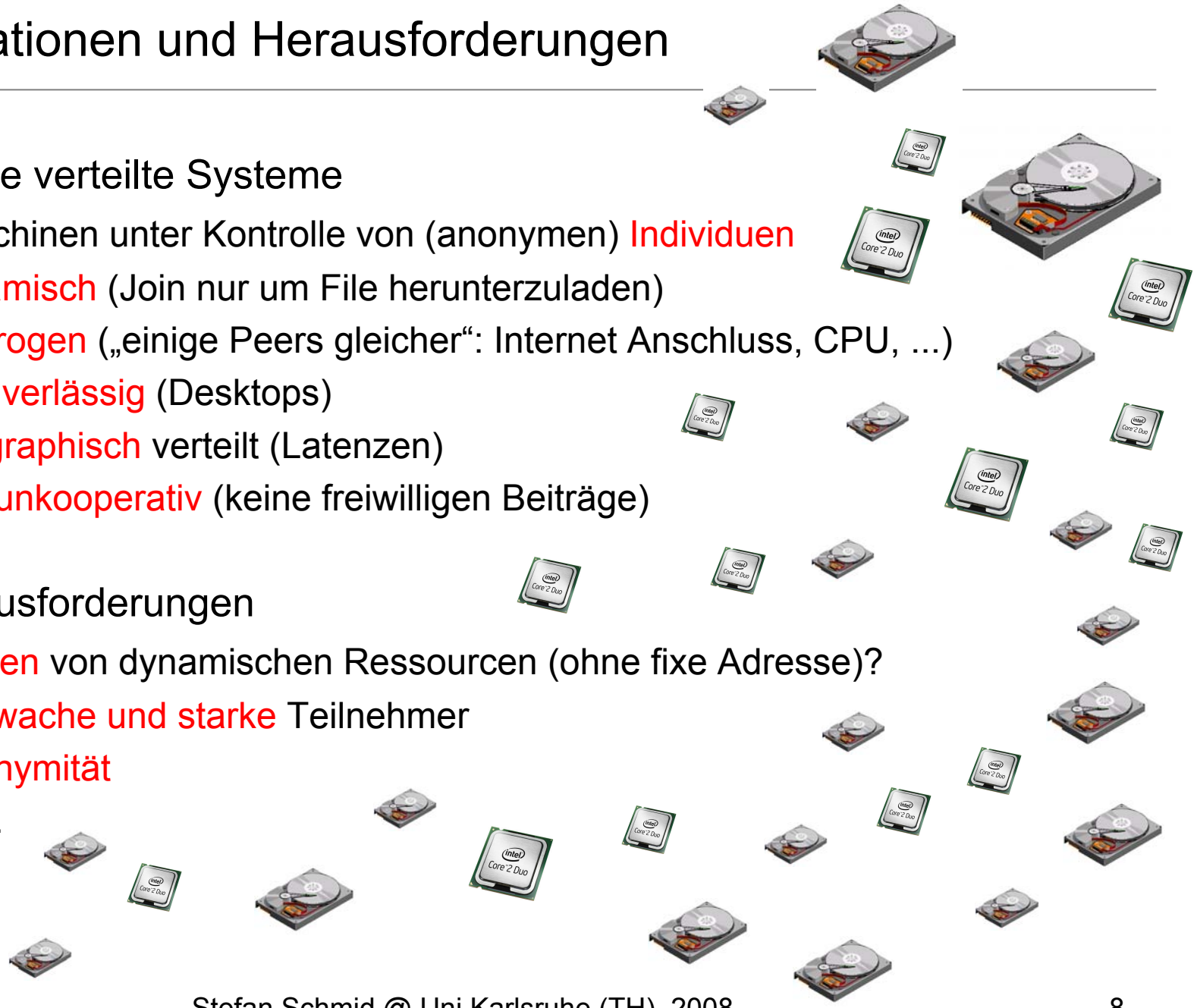
- Vorteile
  - Skalierbarkeit: Mehr Ressourcen in grösseren Netzwerken („Kuchen wächst“)
  - Robustheit: Kein Bruchpunkt („single point of failure“)
  - Kann effizienter sein als Server basierte Lösungen
  - Billig: Start-up vs Google

- Deshalb:
  - Keine teure Infrastruktur nötig
  - Demokratischer Aspekt: Jeder kann Inhalte / Reden publizieren

Implikationen und Herausforderungen

- Offene verteilte Systeme
  - Maschinen unter Kontrolle von (anonymen) Individuen
  - dynamisch (Join nur um File herunterzuladen)
  - heterogen („einige Peers gleicher“: Internet Anschluss, CPU, ...)
  - unzuverlässig (Desktops)
  - geographisch verteilt (Latenzen)
  - z.T. unkooperativ (keine freiwilligen Beiträge)

- Herausforderungen
  - Finden von dynamischen Ressourcen (ohne fixe Adresse)?
  - Schwache und starke Teilnehmer
  - Anonymität
  - usw.



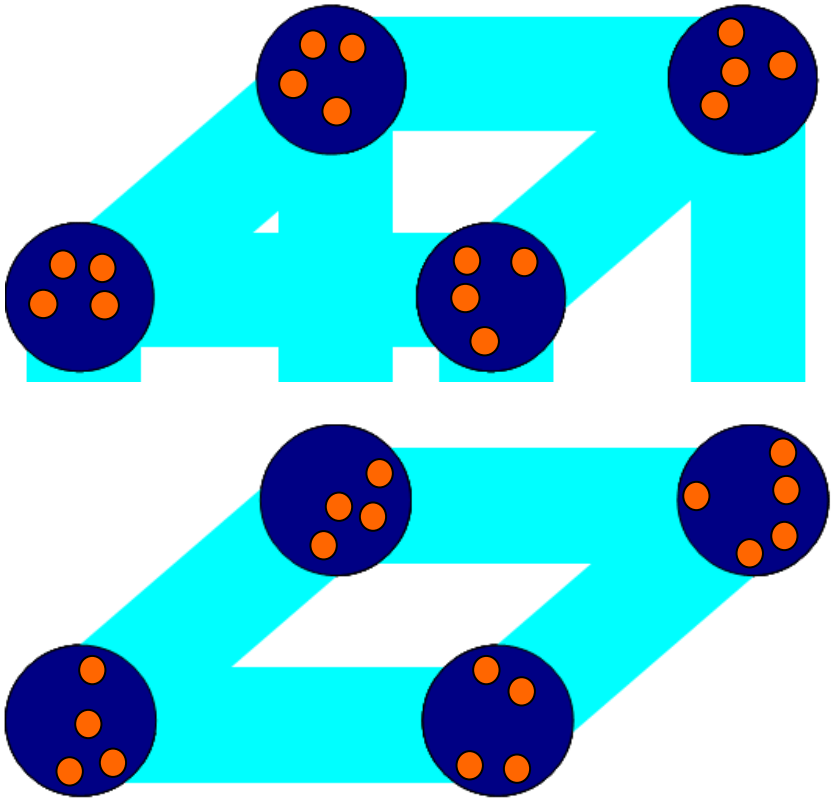
Fokus: Dynamik

IPTPS'05 & IWQoS'06

- Ein möglicher (aber limitierter) Ansatz für dynamische Systeme
- Ziel: Worst-case Resistenz bezgl. kontinuierlichen Änderungen
  - gegnerische Topologieänderungen (Joins und Leaves)
- Grund:
  - Pessimistischer Ansatz aber stärkere Garantien
  - Beinhaltet Wurm entlang Kanten oder Crawler der Topologie kennt
- Ansatz: Graphsimulation
- Resultate:
  - Trotz ADV(log n, log n, 1) wird Hyperwürfel-Topologie erhalten
  - Knotengrad und Netzwerkdurchmesser: O(log n) (asymptotisch optimal)
  - Geht auch für Pancake Graphen: statt log n dann log n / loglog n

Rezept

1. Suche Graph mit gewünschten Eigenschaften
2. Simuliere Graph: Simuliere jeden Knoten durch Menge von Peers
3. Entwickle Token Distribution Algorithmus für diesen Graphen
4. Algorithmus um totale Anzahl Peers im System zu schätzen
5. Algorithmus um Graphdimension anzupassen

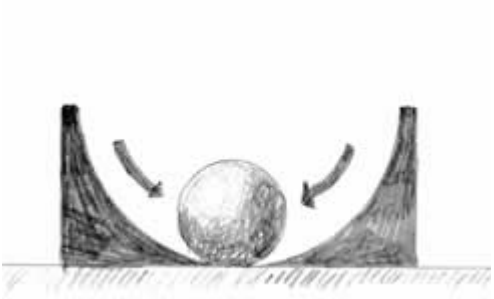


State of the Art

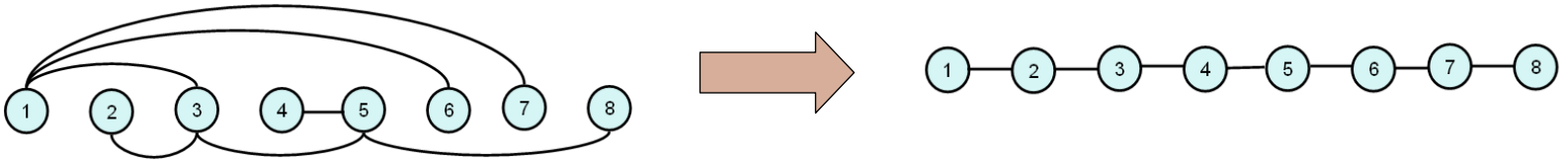
- Beweisbare Robustheit bis zu einer gewissen Grenze
  - Z.B. nicht: Bösewicht zerstört ganze Knoten auf einen Schlag
- Ziel: Topologie soll sich aus jeder Situation regenerieren können
- Wichtiges Konzept: Selbst-Stabilisierung
- Im Moment noch ungelöst für skalierbare Topologien!
  - aber wir arbeiten dran ☺
- Im Folgenden ein paar erste Erkenntnisse für die Linie
  - Warnung: work in progress

Verteilte Topologische Selbst-Stabilisierung

- **Selbst-Stabilisierung:**
  Aus beliebigem Anfangszustand konvergiert System zu „wünschenswerter“ Konfiguration



- **Topologisch:**
  Ziel gegebenen Graph zu linearisieren bezgl. IDs;
  erster Schritt für komplexere Overlays (z.B. Skip Graphen)



- **Verteilt:**
  Jeder Knote fährt sein eigenes Protokoll

Resultate

- Zwei einfache Algorithmen und ein skalierbares Modell

- Resultat Uebersicht:

Worst-case Scheduler:
LIN_{max} braucht $\Theta(n^2)$ Runden
LIN_{all} braucht $O(n^2 \log n)$ Runden

Greedy Scheduler:
LIN_{all} braucht $O(n \log n)$ Runden

Best-case Scheduler:
LIN_{max} und LIN_{all} brauchen $\Omega(n)$ Runden

Mit Gradbeschränkung (Worst-case Scheduler):
LIN_{max} braucht maximal $O(n^2)$ and LIN_{all} maximal $O(n^3)$ Runden

Algorithmus (1)

- Linearisierungsschritt über Knotentriple

- Erhält Zusammenhang

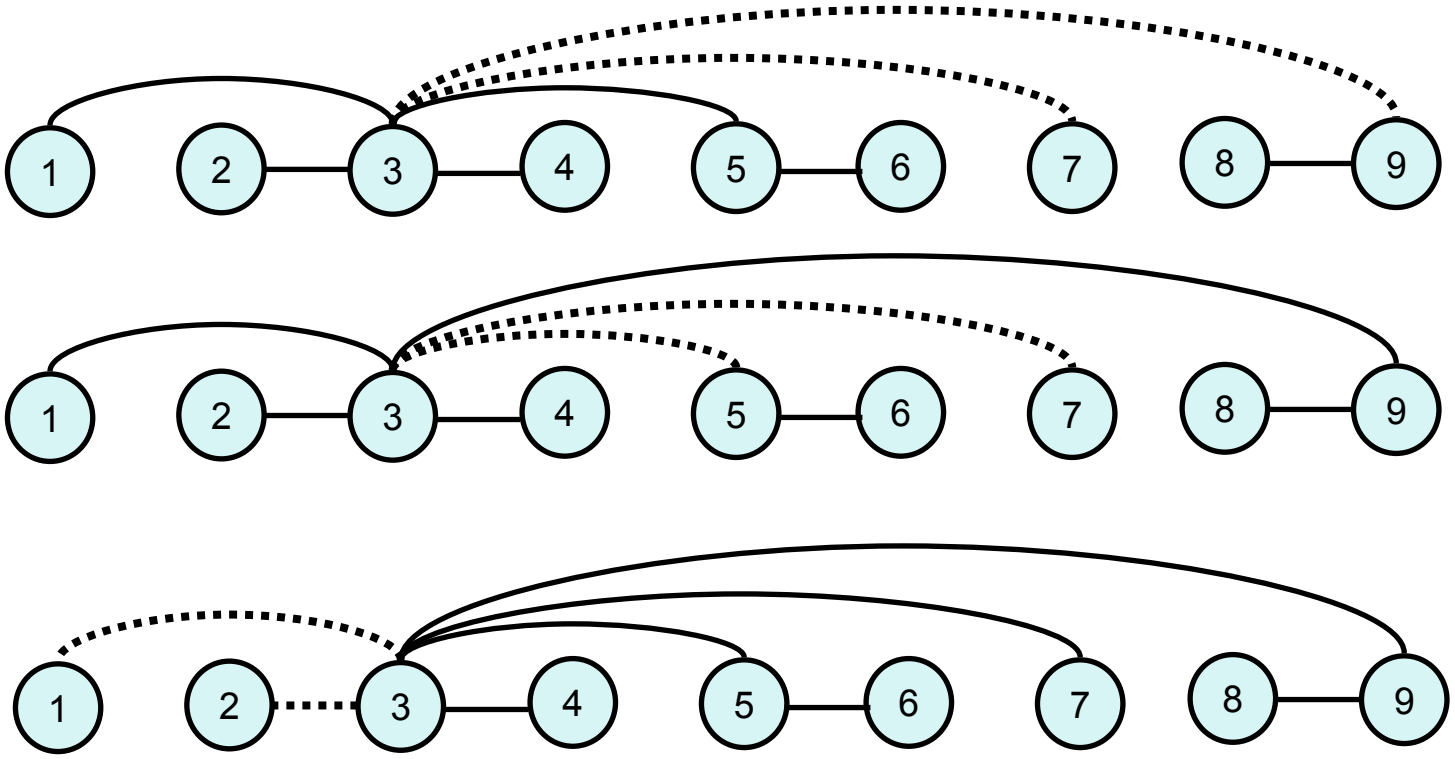
- LIN_{all} schlägt *alle* möglichen Triples dem Scheduler vor (für Knote *u*)

$$\textbf{linearize left}(v,w): \quad (v,w \in u.L \ \wedge \ w < v < u) \ \rightarrow \ e(u,w) := 0, e(v,w) := 1$$

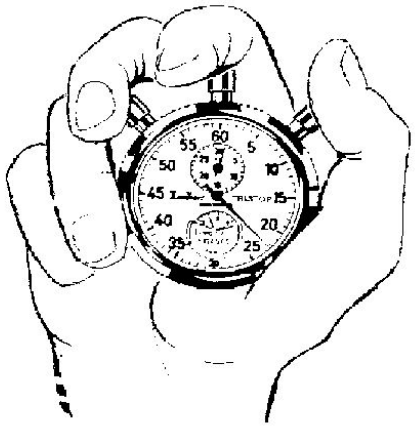
$$\textbf{linearize right}(v,w): \quad (v,w \in u.R \ \wedge \ u < v < w) \ \rightarrow \ e(u,w) := 0, e(v,w) := 1$$

Algorithmus (2)

LIN_{all} schlägt *alle* möglichen Triples dem Scheduler vor (für Knote *u*)



Time Complexity Modell

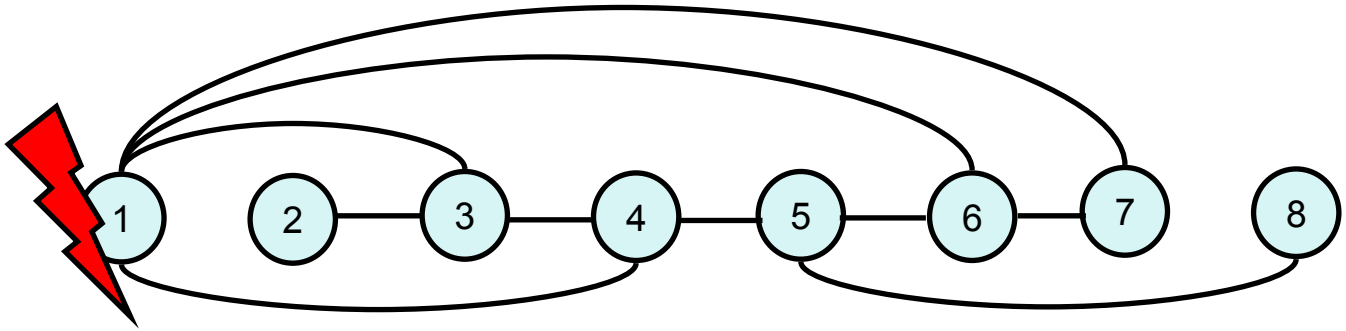


- Self-stabilisierender Algorithmus: terminiert „schlussendlich“
- Konvergenzzeit?
- Analyse von synchronem Modell
  - totale Anzahl Runden (nach letzter Änderung) = Ausführungszeit
- Was kann in einer Runde erreicht werden?

Aus Gründen der Skalierbarkeit sollte ein Knote nicht in zu viele Änderungen pro Runde involviert sein!

Ein naives Modell

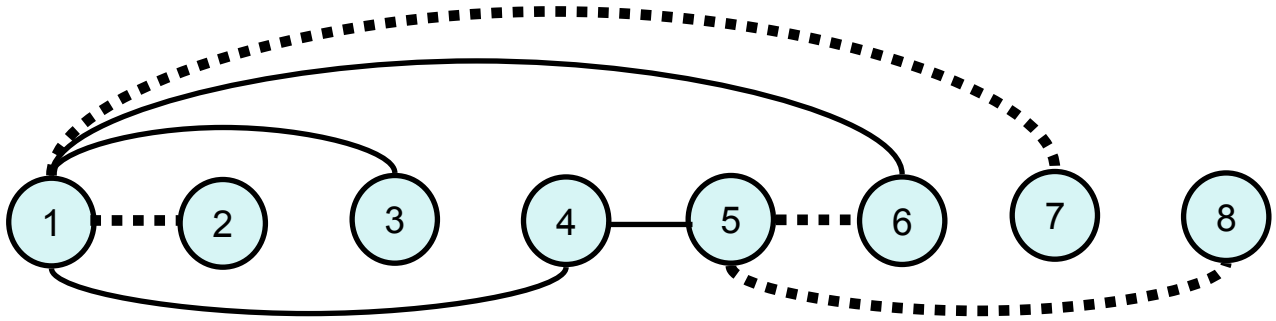
- Es gibt viele Arten, eine Runde zu definieren
- Zum Beispiel: jeder Knote darf eine Aktion feuern pro Runde
- Problem: Knoten in viele Änderungen involviert
  - Deshalb: Lösung skaliert nicht!



Ein skalierbares Modell

Wir schlagen folgendes Modell vor:

- Seien *V(A)* die Knoten die in eine Aktion A involviert sind
- Zwei Aktionen *A* und *B* sind *unabhängig* falls $V(A) \cap V(B) = \{\}$
- Nur unabhängige Aktionsmengen werden gefeuert in Runde



Schedulers (1)

Knoten schlagen verschiedene aktive Aktionen dem Scheduler vor ...

Worst-case Scheduler: Unabhängige Menge von aktiven Aktionen, die Laufzeit maximieren

Best-case Scheduler: Unabhängige Menge die Laufzeit minimiert

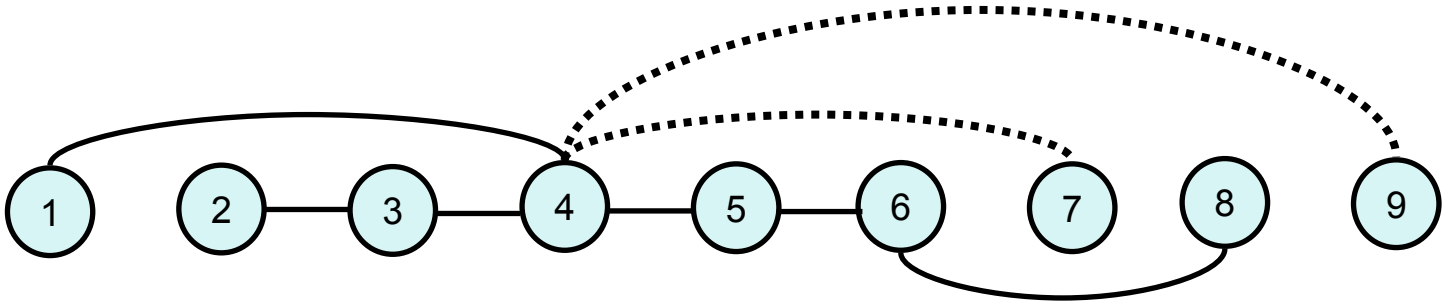
Randomisierter Scheduler: Zufällige unabhängige Menge

Greedy Scheduler: Gibt Knoten mit hohem Grad Priorität

Schedulers (2)

- Im folgenden: Beispielresultat für Greedy Scheduler

> ***Greedy Scheduler**: In jeder Runde werden Knoten bezgl. verbleibendem Knotengrad (entferne gefeuerte Triples inkl. inzidente Kanten) sortiert. Scheduler nimmt Knote v mit grösstem Grad und feuert Triple mit v und dessen zwei entferntesten Nachbarn auf grösserer Gradseite.*



Beispielanalyse (1)

Theorem: Mit Greedy Scheduler terminiert LIN_{all} in *O(n log n)* Runden.

***Greedy Scheduler**: Knoten werden in jeder Runde nach verbleibendem Grad (exklusive gefeuerte Triples und inzidente Kanten) sortiert. Scheduler nimmt Knoten v mit maximalem Grad, und linearisiert die zwei entferntesten Nachbarn von v auf grösserer Gradseite.*

Beweis.

Betrachte Potenzialfunktion $\Psi = \sum_{e \in E} \text{len}(e)$

Zu Beginn: $\psi_0 < n^3$

Am Ende: $\psi = n\text{-}1$

Wir zeigen, dass in jeder Runde das Potenzial ψ mit einem Faktor von höchstens $1\text{-}1/(24 \cdot n)$ multipliziert wird. *Daraus folgt das Theorem.*

Beispielanalyse (2)

Daraus folgt das Theorem?

Lemma: Sei Ξ eine Potenzialfunktion, Ξ_0 das Startpotenzial und Ξ_i das Potenzial nach der i-ten Runde von Algorithmus ALG. Wenn $\Xi_i \leq \Xi_{i-1} \cdot$ **(1-1/f)** und falls ALG stoppt wenn $\Xi_j \leq \Xi_{stop}$, dann ist die Laufzeit von ALG $\mathbf{O(f \cdot \log(\Xi_0/\Xi_{stop}))}$ Runden.

Beweis: Es ist $\Xi_j \leq \Xi_0 \cdot (1\text{-}1/f)^j$. Aus $j = f \cdot \ln(\Xi_0/\Xi_{stop})$ folgt

$$\Xi_j \leq \Xi_0 \cdot (1-1/f)^{f \cdot \ln \frac{\Xi_0}{\Xi_{stop}}} = \Xi_0 e^{f \cdot \left(\ln \frac{\Xi_{stop}}{\Xi_0}\right) \cdot \ln(1-1/f)} \leq \Xi_0 e^{f \cdot \left(\ln \frac{\Xi_0}{\Xi_{stop}}\right) \cdot (-1/f)} = \Xi_0 e^{-\ln \frac{\Xi_0}{\Xi_{stop}}} = \Xi_{stop}.$$

$\ln(1+x) \leq x$ for all $x > -1$

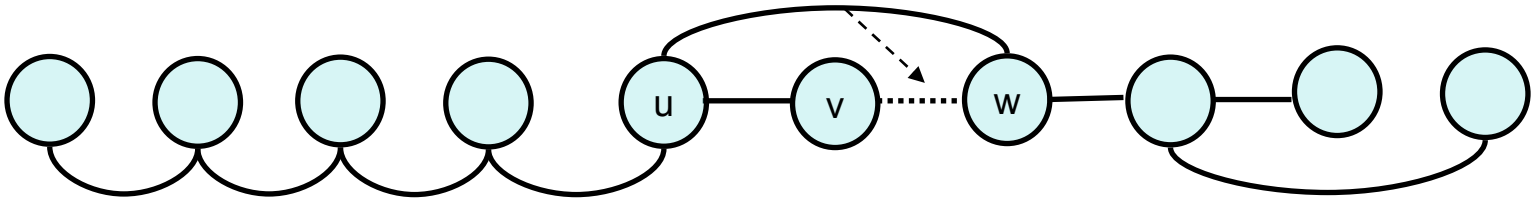
Beispielanalyse (3)

Greedy scheduler: *Knoten werden in jeder Runde nach verbleibendem Grad (exklusive gefeuerte Triples und inzidente Kanten) sortiert. Scheduler nimmt Knoten v mit maximalem Grad, und linearisiert die zwei entferntesten Nachbarn von v auf grösserer Gradseite.*

Betrachte die Potenzialfunktion $\Psi = \sum_{e \in E} \text{len}(e)$

Wir wollen zeigen, dass ψ in jeder Runde mit einem Faktor höchstens 1-1/(24· *n*) multipliziert wird. Daraus folgt das Theorem.

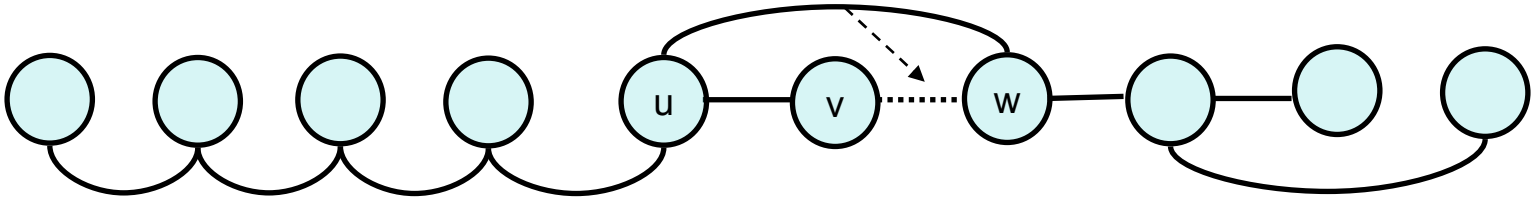
- Beobachtung: Triple feuern reduziert zwar Potenzial ψ...
- ... aber dafür sind andere Knoten nun blockiert in dieser Runde.



- Idee: Schätze blockiertes Potenzial ab.

Beispielanalyse (4)

- Betrachte folgenden Linearisierungsschritt:



- Entfernen von {*u*,*w*} und ev. Hinzufügen von {*v*,*w*} reduziert Potenzial mindestens um

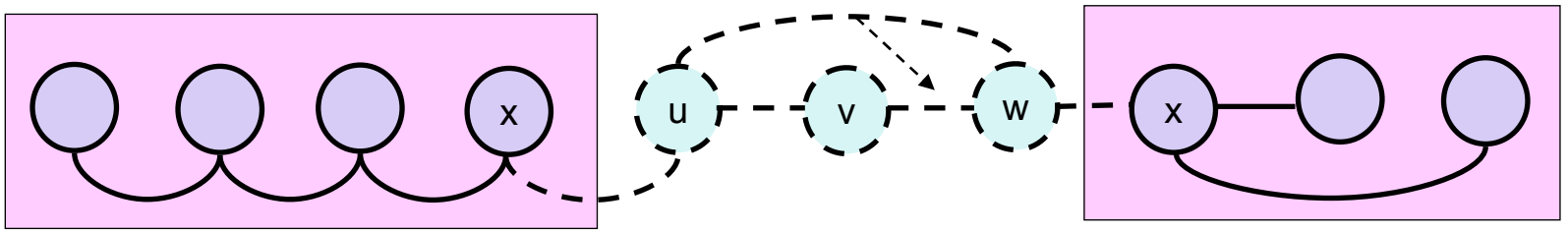
$$\text{dist}(u,w)\text{-dist}(v,w) = \text{dist}(u,v)$$

(Deshalb nehmen wir weitentfernteste Nachbarn!)

- Weil der Greedy Scheduler Seite mit höherem Grad ($\geq \deg(u)/2$) nimmt:

$$\text{dist}(u,v) \geq \deg(u)/2 - 1 \geq \deg(u)/4$$

Beispielanalyse (5)



- Also: Potenzial reduziert sich in einem Schritt um mind. *deg(u)/4*

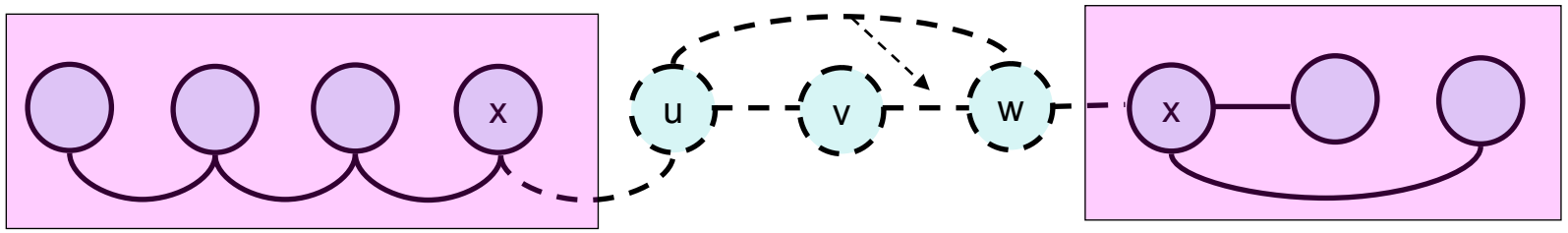
- Wie viel Potenzial ist blockiert?

- Betrachte verbleibende Komponenten (nachdem Triple entfernt)
- Betrachte Nachbar *x* von *u*, *v* oder *w*

  - falls *x* in einer sortierten Linienkomponente => blockiertes Potenzial maximal *n+n* (inzidente Kante dorthin plus Verbindungen in Komponente)

  - falls *x* in anderer Komponente => kann weiter linearisiert werden in dieser Runde (berücksichtige jenes Potenzial dann, und jetzt nur Länge des inzidenten Links: *n*)

Beispielanalyse (6)



- Blockiertes Potenzial maximal $6 \cdot \deg(u) \cdot n$

  - weil *u* grösseren Grad hat als *v* und *w* („*local control*“ genügt!),
  - und weil wir maximal $2 \cdot n$ blockiertes Potenzial pro Nachbar haben (*n* für Komponente plus *n* für Link zu diesem Nachbarn)

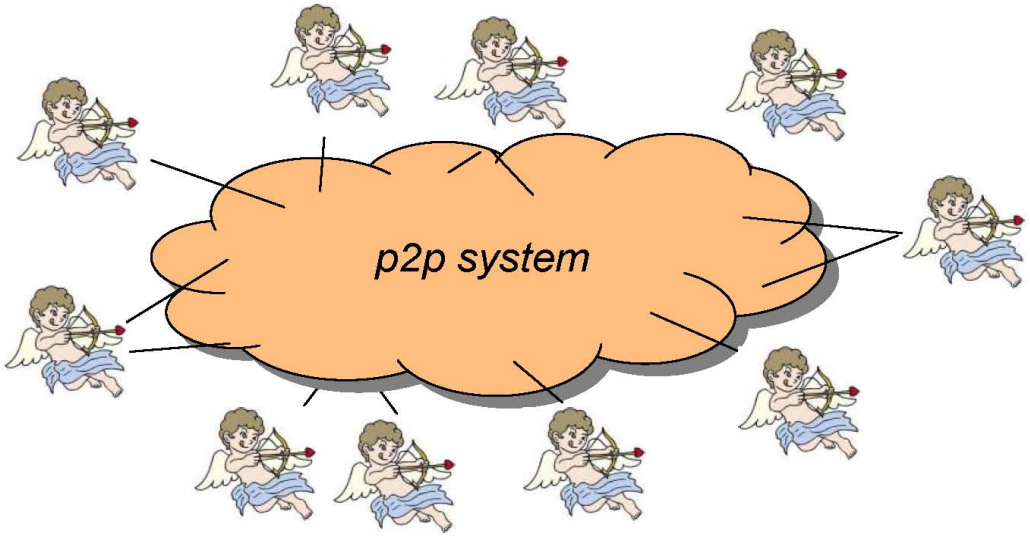
- Bereits gezeigt: gewonnenes Potenzial mindestens $\deg(u)/4$

- Deshalb: Potenzial reduziert um Faktor von mindestens $1-\Theta(1/n)$ pro Runde.

QED.

Fokus: Unkooperatives Verhalten

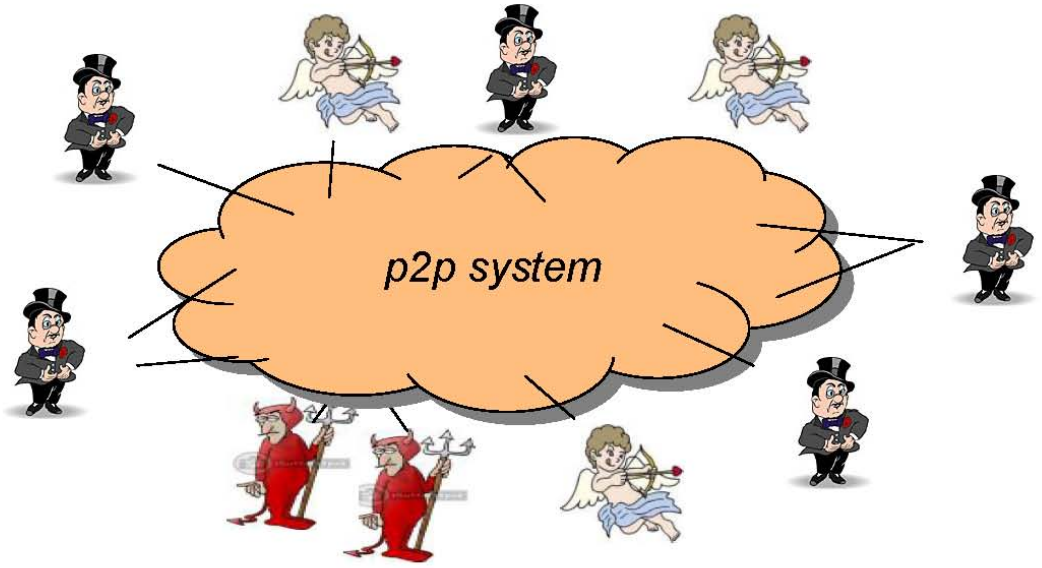
Weiteres wichtiges Problem offener verteilter Systeme!



Unkooperatives Verhalten

Teilnehmer teilen nicht unbedingt freiwillig Ressourcen

- Einige Teilnehmer wollen vielleicht dem System sogar schaden



Einflussreicher Talk von Ch. Papadimitriou (STOC 2001)

„The Internet is unique among all computer systems in that it is built, operated, and used by a multitude of diverse economic interests, in varying relationships of collaboration and competition with each other.“

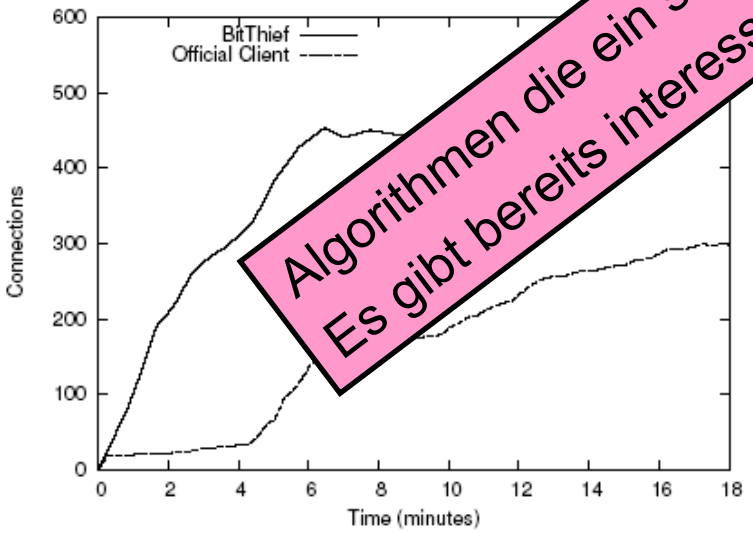
„This suggests that the mathematical tools and insights most appropriate for understanding the Internet may come from a fusion of algorithmic ideas with concepts and techniques from Mathematical Economics and Game Theory.“

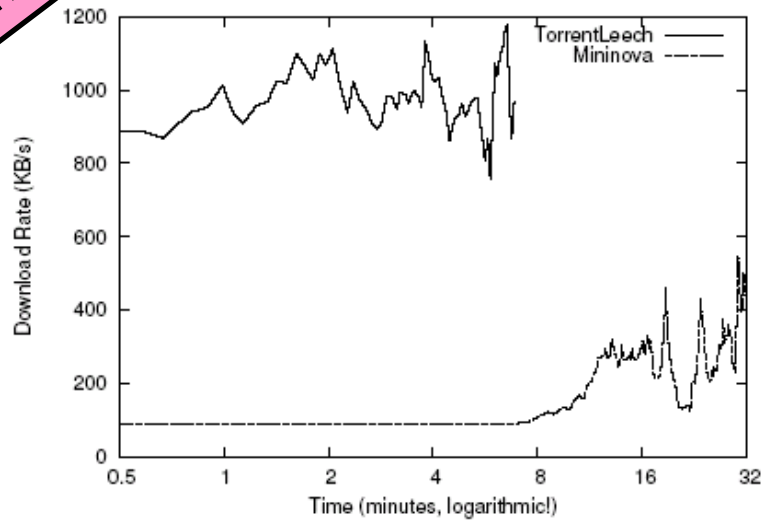
Free-riding ist heute einfach!

HotNets'06

- Fallstudie BitTorrent: BitThief (Client ohne Uploads!)
  - Resultat: Free riding heute möglich (Prinzip: viele Verb en)
  - Sogar ohne Seeders (optimistic unchoking Mecha
  - und trotz tit-for-tat Anreizsystem!
  - Unser BitThief client ist relativ schnell fall ers (*round robin*), kleine Files (*large view exploit*), oder eeders.
  - Sharing communities können au t werden.

Algorithmen die ein solches Free-riding verhindern?
Es gibt bereits interessante Vorschläge!



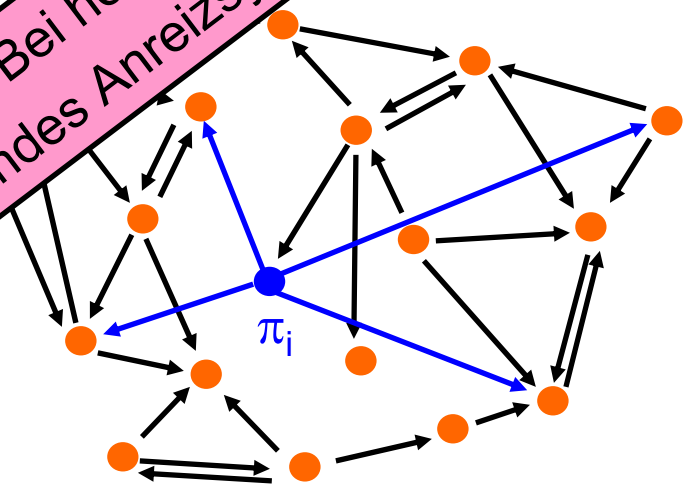


Folgen von eigennützigem Verhalten?

PODC'06

- Werkzeuge der Spieltheorie...
- Analyse eines Netzwerkformationspiels
- Resultate
  - Der Price of Anarchy ist $\Theta(\alpha,n)$
    (stark degeneriert, kein Hype
  - Eigennütziges Verhalten
    destabilisieren auch
    (keine determini n Equilibrien)
  - Es ist NP-h ilen, ob ein gegebenes
    Netzwe oder nicht
    (Reduktion speziellem SAT)

Algorithmische Implementationstheorie: Bei hohem Price of Anarchy lohnt sich ein entsprechendes Anreizsystem.

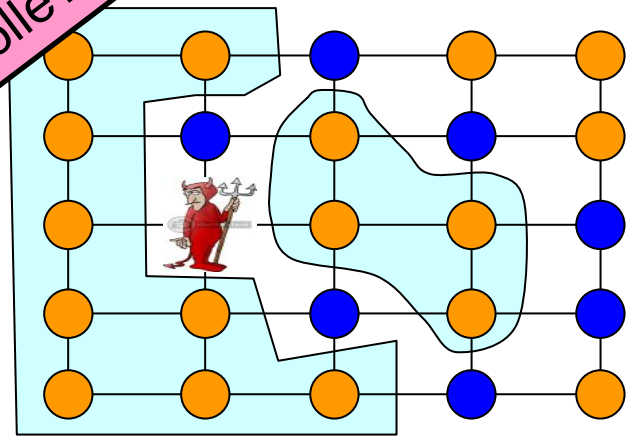


Folgen von böswilligem Verhalten?

PODC'06

- Andere Formen von unkooperativem Verhalten
- Unser Framework erlaubt Quantifizierun
  - Price of Malice (analog zu Price of An
  - Robustheit gegenüber böswilliger
- Resultate
  - Beispiel: Virus Inc ne
  - Böswillige Tei aden
  - Falls die eier risikoscheu sind, kann abe anz besser werden!

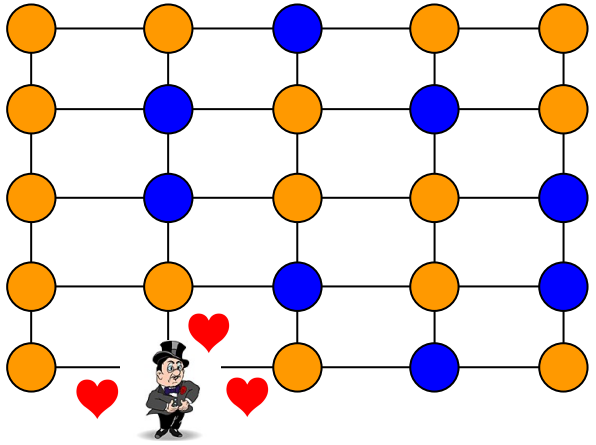
Hoher Price of Malice: Anreizsysteme schwieriger. Lösung durch Zugangskontrolle?



Folgen von sozialem Verhalten?

EC'08

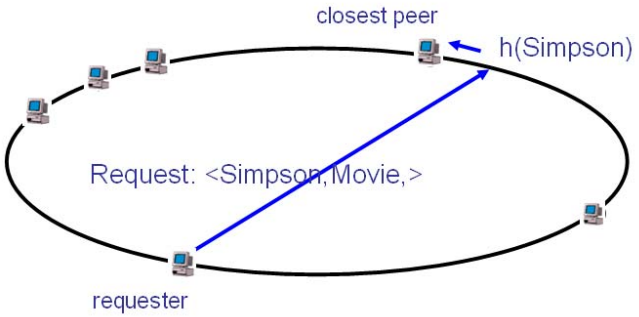
- Framework auch auf soziale Netzwerke anwendbar
- Z.B. Analyse von Impfstrategien falls Spieler sich um ihre direkten Kontakte kümmern



- Resultate
  - Beispiel: Virus Inoculation Game
  - Equilibrien sind immer mind. so gut wie in einem rein egoistischen Umfeld
  - Gewinn ist aber nicht monoton in *F* (= wie stark Spieler sich um einander kümmern)

Fallstudie Kad (1)

- Attacken im Kad Netzwerk?
  - Grosses strukturiertes P2P System
  - Kad: > 1 Mio Users gleichzeitig



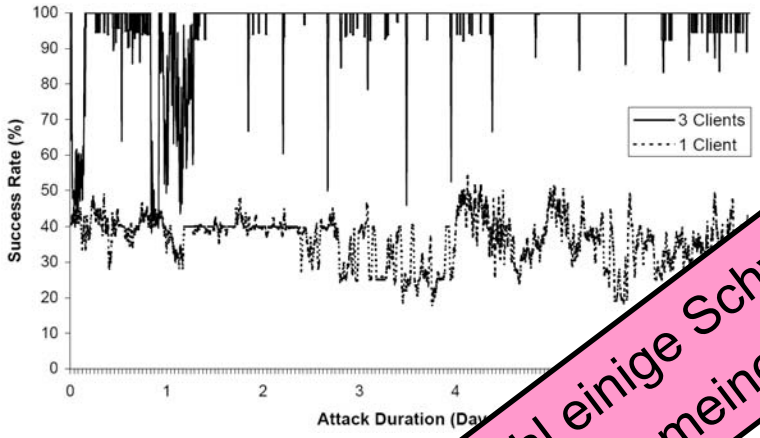
- Experimente mit richtigen Users und richtigem Inhalt

- Beispiel: Zensur im Kad Netzwerk

- Vorgehen:
  - Finde Daten-ID, füge Knoten nahe der ID ein (*node insertion attack*)
  - Fülle Indextabelle der existierenden Hosts (*publish attack*)
  - Plus: Eclipse Attacke und Denial of Service Attacke

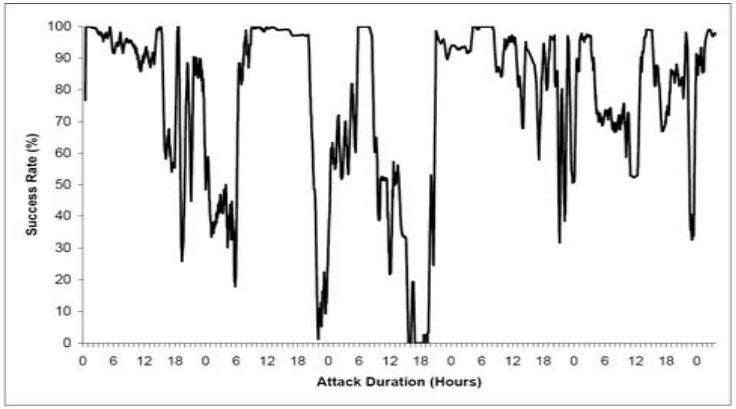
Fallstudie Kad (2)

- Resultate
  - Kad angreifbar mit verschiedenen Attacken
  - Ganze Files können entfernt werden mittels geri… …rcen / Aufwand



- N… …on Attacke auf Keyword …ns“
- …t auch Einfluss auf „Simpsons Movie“ oder „Simpsons Soundtrack“
- …ish Attacke

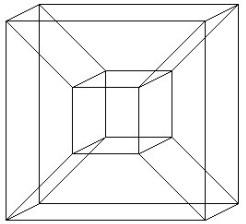
Obwohl einige Schwachstellen ev. behebbar wären, sind allgemeine algorithmische Lösungen unklar.



Work in Progress...

Dynamische Topologien: Selbst-Stabilisierung für skalierbare Topologien

[Zusammenarbeit mit TU München (Prof. Christian Scheideler, Dr. Riko Jacob, Dr. Hanjo Täubig) und Universität Arizona (Prof. Andrea Richa)]

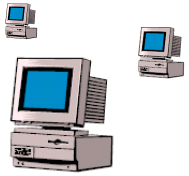


Anreizmechanismen für Kooperation

[Zusammenarbeit mit ETH Zürich (Prof. Roger Wattenhofer und Raphael Eidenbenz) und MS Research (Dr. Thomas Moscibroda)]

SHELL: Overlay für robuste (z.B. Sybil Attacken) und heterogene P2P Netzwerke

[Zusammenarbeit mit Prof. Christian Scheideler]



Chameleon: Ein DoS Attacken resistentes verteiltes Informationssystem

[Zusammenarbeit mit Prof. Christian Scheideler und Matthias Baumgart]

Fazit

Vielen Dank für die Aufmerksamkeit!

- Offene verteilte Systeme sind weit verbreitet
- Interessante Eigenschaften: sehr dynamisch, heterogen, und basieren auf Kooperation
- Interdisziplinär: Graphtheorie, Algorithmik, dynamische Systeme („control theory“), ökonomische Aspekte, usw.
- Prinzipien auch wichtig fürs „future Internet“?
- Heutige Systeme haben noch verschiedene Schwächen