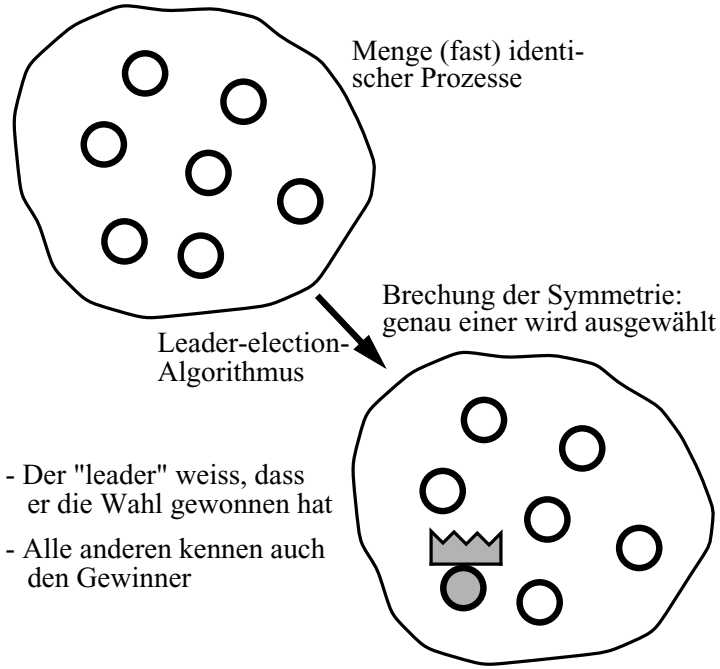
Das Election-Problem

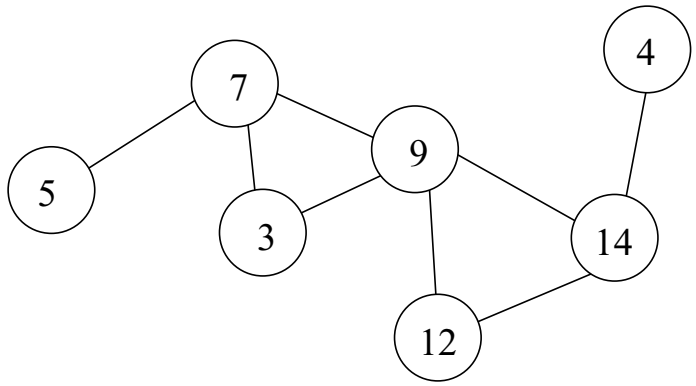


Anwendung z.B.:

- Monitorstation bei Token-Ring-LANs festlegen
- Generierung eines eindeutigen Tokens
- "Root bridge" für Spannbaum bei Ethernet-LANs
- "Symmetrisierung" anderer Algorithmen
  (Verwendung als vorgeschalteter Basisalgorithmus)

Leader-Election mit "message extinction"

- Alle Knoten haben unterschiedliche Nummern > 0
- Verteilter, symmetrischer Algorithmus zur Bestimmung der grössten Identität
- Jeder Knoten soll schliesslich grössten kennen
- Jeder Knoten darf unabhängig (gleichzeitig) den Algorithmus initiieren
- Schema der verteilten Approximation anwenden



Fragen:

- Anzahl der Nachrichten?
- Terminierung?
- Bessere Algorithmen für gleiches Problem?
- Spezielle Topologien (Ring, Baum)...?

Der Leader-Election-Algorithmus

- Nachrichtengesteuerte Spezifikation des Algorithmus ("*message driven*")
- *Atomare Aktionen* ← vereinfacht u.a. auch die Verifikation
- Jeder Prozess mit Identität p hat lokale Variable M_p
- M_p ist initial 0; am Ende enthält M_p die Identität des Masters ("leader")

Alle Prozesse verhalten sich gleich ("syntaktische Symmetrie")

Name der Aktion

assertion

I_p: $\{M_p = 0\}$
$M_p := p;$
send $\langle M_p \rangle$ **to** all neighbours

R_p: {Eine Nachricht $\langle j \rangle$ ist eingetroffen}
if $M_p < j$ **then**
$M_p := j;$
send $\langle M_p \rangle$ **to** all neighbours
fi

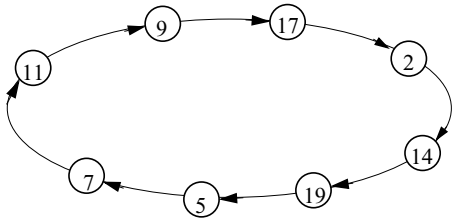
andernfalls "message exctinction"

?

T_p: {Terminierung wurde entdeckt}
if $M_p = p$ **then** *"I am the master"* **fi**

andernfalls ist M_p der Master

Leader-Election auf einem Ring



- Voraussetzung hier: unidirektionaler Ring, wobei alle Identitäten verschieden sind

1. Idee:
- Jeder Prozess wacht irgendwann auf (spätestens wenn er eine Nachricht von einem anderen erhält)
- Startet vollständigen Ringumlauf
- Meldet, ob unterwegs einen grösseren getroffen

"Bully-Algorithmus"

==> n^2 Einzelnachrichten (n = Anzahl der Prozesse)

---

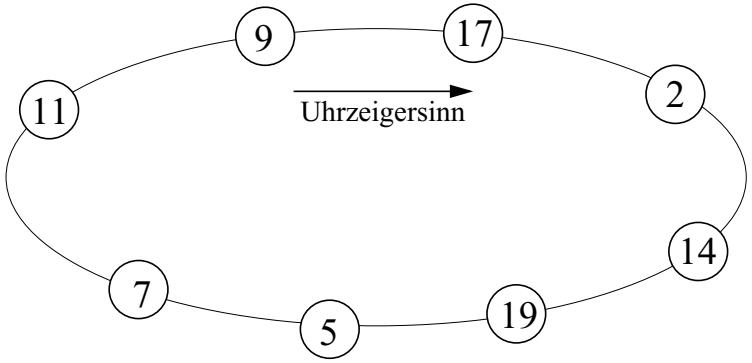
2. Idee: Message-extinction anwenden!

I_p: $\{M_p = 0\}$
$M_p := p;$
send $\langle M_p \rangle$ **to** next

R_p: {Eine Nachricht $\langle j \rangle$ ist eingetroffen}
if $M_p < j$ **then**
$M_p := j;$
send $\langle M_p \rangle$ **to** next
fi
if $j = p$ **then** *"I am the master"* **fi**

Election: Nachrichtenkomplexität

- Message-extinction-Prinzip von Chang und Roberts 1979
  - war einer der ersten verteilten Algorithmen



- *Worst-case Nachrichtenkomplexität* bei k Startern:
  $n + (n-1) + (n-2) + ... + (n-k+1) = \mathbf{nk - k(k-1)/2}$

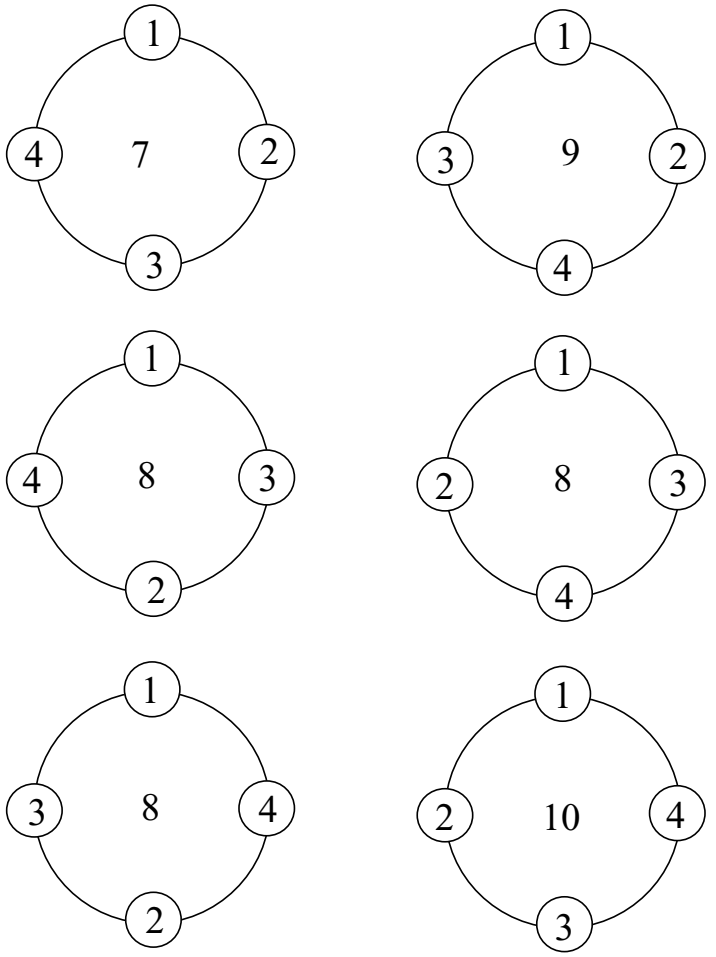
  --> $O(n^2)$ bei Ringgrösse n und k=n

- Wie hoch ist die *mittlere* Nachrichtenkomplexität?
  - bei "zufälliger" Permutation der Identitäten

- Wie hoch ist die *(Einheits)zeitkomplexität*?
  - wenn alle gleichzeitig starten?
  - beim zeitversetzten Starten?

Mittlere Nachrichtenkomplexität (1)

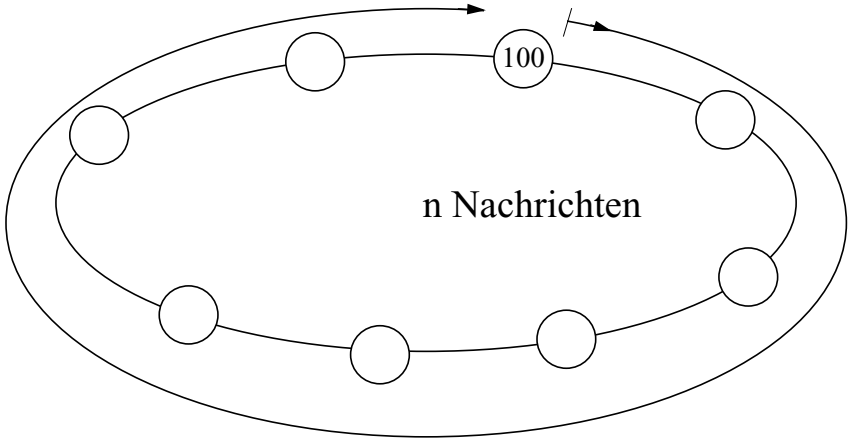
- Beispiel: Sei k = n = 4
- Über alle Permutationen mitteln (wieviele?)

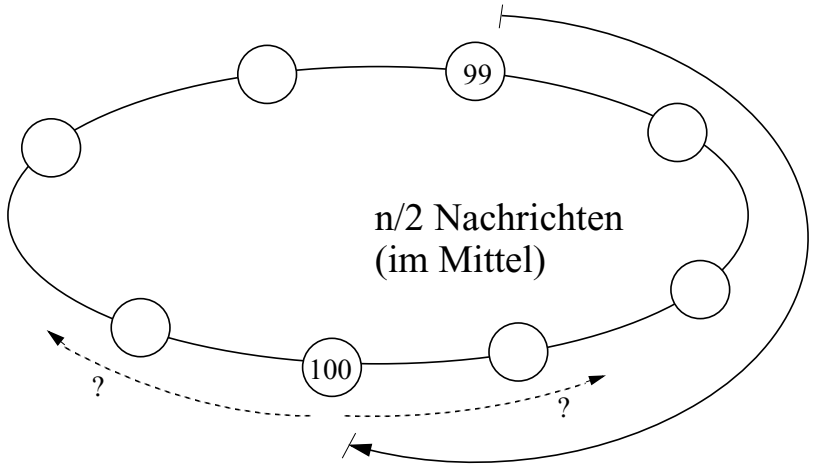


50/6 = 8.333... Nachrichten im Mittel

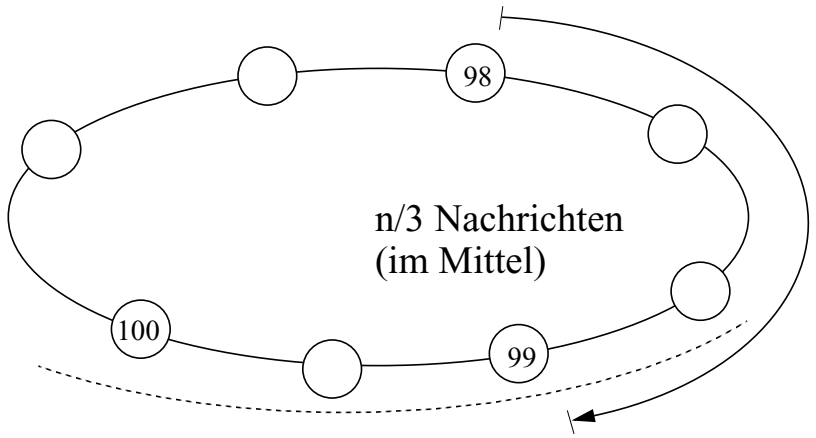
Mittlere Nachrichtenkomplexität (2)

Sei n = 100; id = 1, 2, ..., 100





Mittlere Nachrichtenkomplexität (3)



Vermutung: Drittgrösster sollte im Mittel nach n/3 Schritten auf einen grösseren Prozess (99 oder 100) treffen

- Lässt sich durch explizites Nachrechnen aller ca. n^2 Fälle auch beweisen...

- Allgemeiner Beweis für die Vermutung "n/i Schritte beim i-t grössten"?

Mittlere Nachrichtenkomplexität (4)

- Grösste Identität: n Nachrichten (immer)
- Zweitgrösste: im Mittel n/2
- Drittgrösste: im Mittel n/3 ??

  ...

- i-t grösster: im Mittel n/i ← stimmt das?

Wenn das stimmt -->

$$\frac{n}{1} + \frac{n}{2} + \frac{n}{3} + ... + \frac{n}{k} = n \sum_{i=1}^{k} \frac{1}{i} = n\, H_k$$

einfach aufaddieren?

Def. der sogenannten *harmonischen Reihe*

Was ergibt sich für n = k = 4 ?

$$4\,\frac{12+6+4+3}{12} = 25/3 = 8.333...$$

Stimmt mit früherem Wert überein!
==> *Vermutung*: $n\, H_k$ ist korrekt

- Man kann tatsächlich exakt zeigen:

  Die mittlere Nachrichtenkomplexität beträgt

$$n \sum_{i=1}^{k} \frac{1}{i} = n\, H_k \approx \mathbf{n \ln k}$$

Wir wollen dies gleich genauer herleiten...

Unabhängige Fälle?

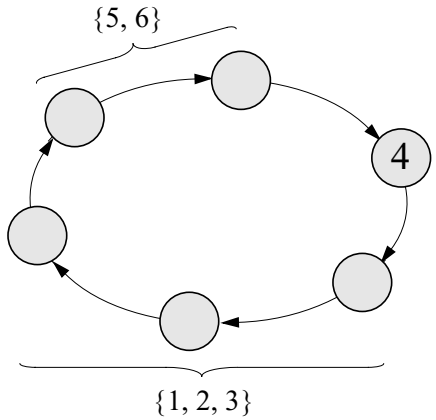
Im allgemeinen sind die "Zufallsvariablen", die die Länge der von Prozess i initiierten Nachrichtenkette repräsentieren, nicht unabhängig voneinander!

Beispiel: Gegeben 6 Prozesse 1, 2, ... 6 im Ring.
Sei *A* das Zufallsereignis "Nachrichtenkette von Prozess 4 hat die Länge 4" und *B* das Zufallsereignis "Nachrichtenkette von Prozess 5 hat die Länge 2"

Aus *A* folgt nebenstehende Situation; dann kann die von 5 initiierte Nachrichtenkette jedoch nur die Länge 1 oder 5 haben, nicht jedoch 2, wie von *B* gefordert

==> $\text{prob}(A \cap B) = 0$, obwohl $\text{prob}(A) \times \text{prob}(B) \neq 0$

==> A und B sind *nicht unabhängig* voneinander!



Allerdings (hier ohne Rechtfertigung):

- die Zufallsvariablen in obigem Beispiel sind *paarweise unkorreliert*
- *Erwartungswerte* (d.h. die mittleren Längen der Nachrichtenketten) dürfen daher *aufaddiert* werden, um den Mittelwert bzgl. der Gesamtnachrichtenzahl zu erhalten (in "Formeln": E(X+Y) = E(X) + E(Y))

Die harmonische Reihe H_n ...

Beh.: H_n divergiert

Bew.: Fasse 4 Reihenglieder ab 1/4 zusammen,
(jew. grösser als 1/8), 8 Reihenglieder ab 1/8...

Beh. (o. Bew.) $\sum_i \frac{1}{i^r}$ konvergiert gdw. $r > 1$

($r = 2$ --> $\pi^2/6$ ... Riemann'sche Zeta-Funktion)

==> Harmonische Reihe "divergiert gerade noch"

Beh. (o.Bew.) $\gamma = \lim_{n \to \infty} (H_n - \ln n) = 0.5772156649...$

Euler'sche Konstante

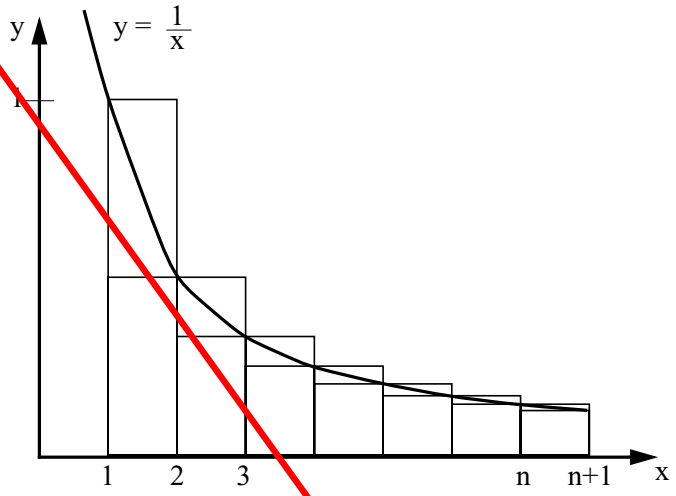
==> H_n lässt sich zumindest für grosse n durch ln n abschätzen

- diese Abschätzung lässt sich sogar noch präzisieren...

Abschätzung der harmonischen Reihe H_n

Beh.: $\ln n < H_n < 1 + \ln n$

Bew.:



- *Fläche der Obertreppe* von 1 bis n+1: H_n

- *Fläche unter Kurve* von 1 bis n+1: $\int_1^{n+1} \frac{1}{x}\, dx = \ln(n+1)$ (wg. $\ln(1) = 0$)

==> $H_n > \ln(n+1)$ ==> $H_n > \ln n$

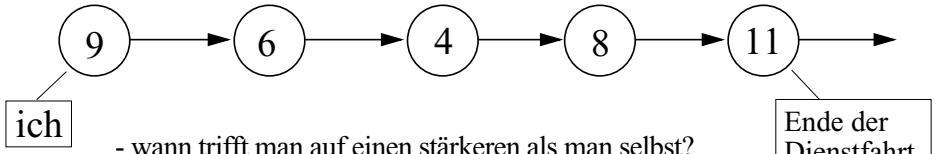
- *Fläche der Untertreppe* von 1 bis n: $H_n - 1$

- *Fläche unter Kurve* von 1 bis n: $\int_1^{n} \frac{1}{x}\, dx = \ln(n)$

==> $H_n - 1 < \ln(n)$ ==> $H_n < 1 + \ln n$

Wartezeit bis zum ersten Rekord

- Wie weit kommt die von einem "x-beliebigen" Knoten initiierte Nachrichtenkette im Mittel?



- wann trifft man auf einen stärkeren als man selbst?
- wie lange überlebt man im Mittel?

In einem Teich schwimmen n Fische verschiedener Grösse - wieviele Fische muss man im Mittel noch fangen, bis man einen fängt, der grösser als der erste gefangene Fisch ist? (Oder bis der Teich leer ist)

Lösungsansatz (aber darf man wirklich so vorgehen?):

Fängt man den grössten zuerst --> n Fänge ("Pech", Teich wird leer)
Fängt man den zweitgrössten zuerst --> n/2 Fänge im Mittel
Fängt man den drittgrössten zuerst --> ...

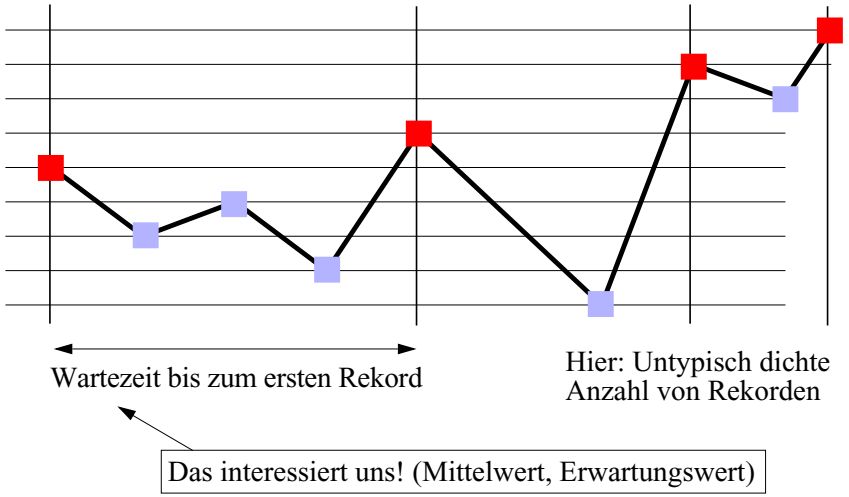
Zurückführung auf ein bek. math. Modell: *Urnenmodell ohne Zurücklegen*

Z.B. n=100, 7.grösster Fisch initial:
--> 6 schwarze und 93 weisse Kugeln
--> Sei p die Wahrscheinlichkeit, eine schwarze Kugel zu ziehen ("Treffer")
--> Beispiel: Wahrscheinlichkeit, eine weisse und dann eine schwarze zu ziehen (d.h. genau die Entfernung 2 zu schaffen): (1-p) mal Wahrscheinlichkeit für "Treffer bei 6 schwarzen und 92 weissen"
--> ... alles gewichtet aufsummieren... (aufwendig!)

Warten auf einen neuen Rekord...

- Rekord = grösserer Wert als alle vorangehenden
- "left to right maxima" (einer Zahlenfolge)

- Z.B. "heissester August seit Anfang des Jahrhunderts"



- Anwendung: z.B. Verlobungshäufigkeit bei deletion-sort

Festhalten und weitersuchen

Rekorde ...werden immer seltener

- Wieviele Rekorde gibt es eigentlich?
  - z.B. heissester August in einem Jahrhundert...
  - Annahme: keine Korrelation (d.h. zufällige Permutation vorausgesetzt)

- Betrachte Bitfolge: i-te Stelle 1 gdw. i-ter Wert ein Rekord

  Bsp: 10010110000001000000000010000000000000001
  (Rekorde traten im 1., 4., 6., 7., 14.,... Jahr auf)

- Wieviele 1en bei gegebener Länge n?

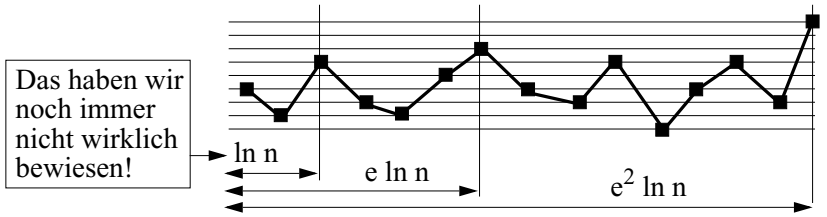
  Antwort:
  - damit i-tes Bit auf 1 steht, muss der i-te Zahlenwert der Folge ≥ alle ersten i Zahlenwerte sein
  - Wahrscheinlichkeit dafür ist 1 / i (stimmt's? wieso?)

  --> Mittlere Anzahl von 1en = 1 + 1/2 + 1/3 +...
  - darf man wirklich einfach so aufaddieren?

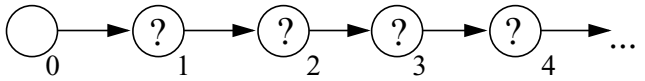
  --> Mittlere Anzahl von Rekorden ist $H_n \approx \ln n$

  - Bsp: es gibt ca. 5 heisseste Auguste im Jahrhundert
    ... und wieviele in einem Jahrtausend?

==> Für einen Rekord mehr (im Mittel) muss das Zeitintervall ca. e = 2.71828 mal länger sein



Verteilung der Lebensdauer



Zu jedem Zeitpunkt ein Schritt...

- Wahrscheinlichkeit, *genau* bis zur Position i zu gelangen (und dort besiegt zu werden)?

- p(Lebenszeit = i) = p(Lebenszeit ≥ i) - p(Lebenszeit > i)
  - Vgl.: 18% Studis ≥ 16. Semester; 13% Studis ≥ 17. Semester (= > 16. Semester)
    ==> 5% *im* 16. Semester

Wahrscheinlichkeit für "grösster unter den ersten i"

- Also: $p(\text{Lebenszeit} = i) = \frac{1}{i} - \frac{1}{i+1} = \frac{1}{i(i+1)}$ (für i<n)

- Aber: $p(\text{Lebenszeit} = n) = \frac{1}{n}$ ("erfolgreicher Durchmarsch")

Beispiel n=4:

gewichtete Summe = 25/12 --> 25/3 = 8.333...
Nachrichten im Mittel (vgl. früheres Ergebnis!)

1	$\frac{1}{1\times2} = \frac{1}{2}$	= 50%
2	$\frac{1}{2\times3} = \frac{1}{6}$	= 16.7%
3	$\frac{1}{3\times4} = \frac{1}{12}$	= 8.3%
4	$\frac{1}{4}$	= 25%

100%

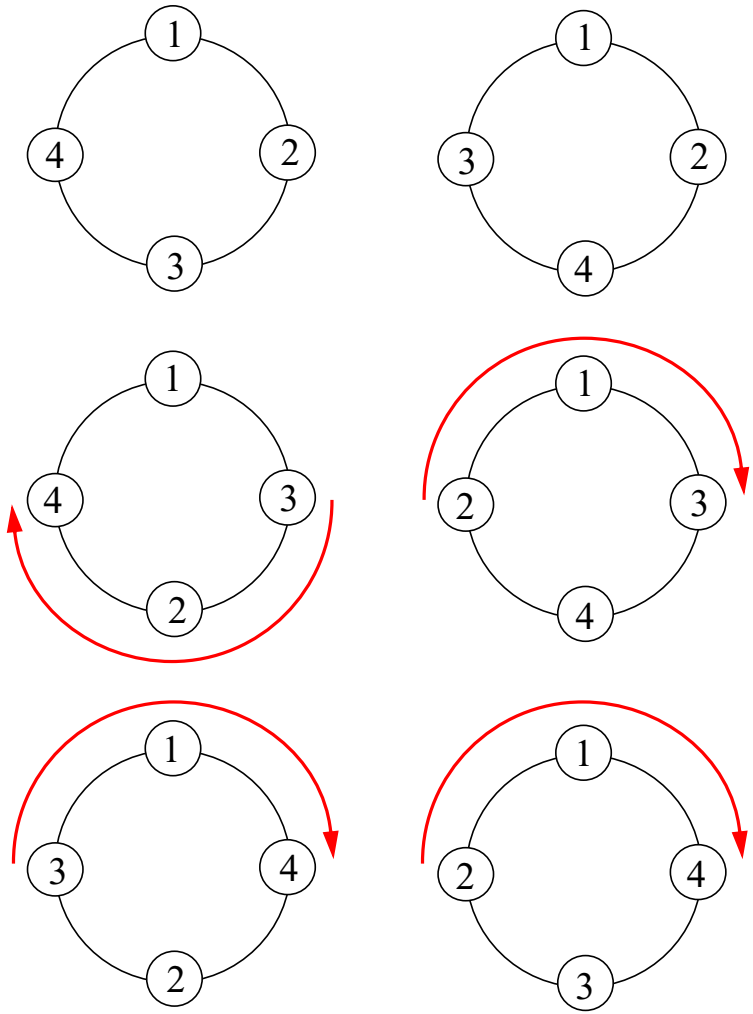
vgl. dazu nächstes Bild

Bem.: Es gilt

$$\frac{1}{n} + \sum_{i<n} \frac{1}{i(i+1)} = 1$$

(Induktionsbeweis als einfache Übung)

Lebensdauer 2 bei n=4



6 Bilder — jeweils 4 Starter

In 4 von $6 \times 4 = 24$ Situationen, also 1/6 aller Fälle (= 16.7%),
wird für einen Initiator die Lebensdauer 2 exakt erreicht

Induktionsbeweis der Summenformel

Behauptung: $\sum_{i=1}^{n-1} \frac{1}{i(i+1)} = 1 - \frac{1}{n}$

Beweis durch vollständige Induktion:

Induktionsanfang: $n = 1 \;\text{-->}\; 0 = 0$ ✓

$n = 2 \;\text{-->}\; \frac{1}{1 \cdot 2} = 1 - \frac{1}{2}$ ✓

Induktionsschritt:

$$\sum_{i=1}^{n} \frac{1}{i(i+1)} = \overbrace{1 - \frac{1}{n}}^{\text{ind. annahme}} + \frac{1}{n(n+1)}$$

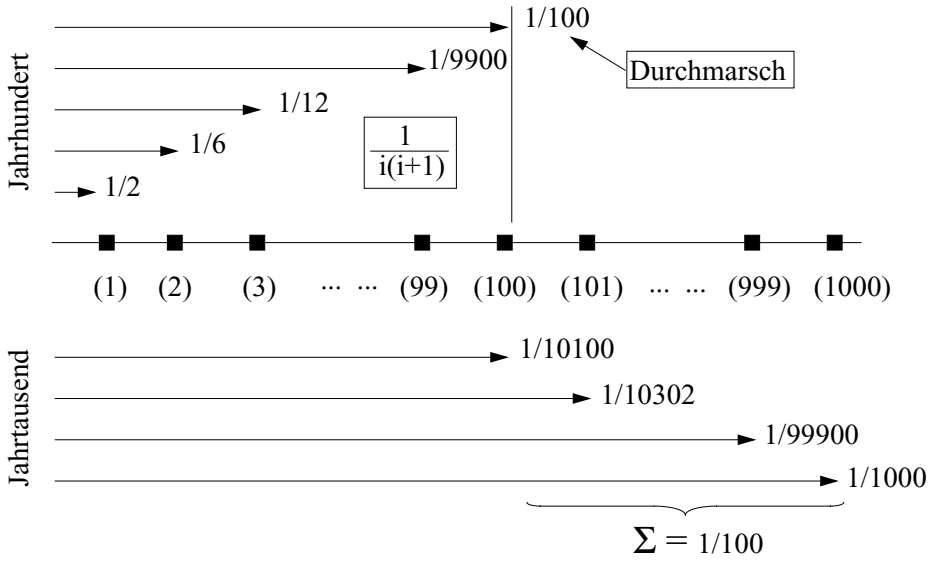
$$= 1 + \frac{-(n+1) + 1}{n(n+1)}$$

$$= 1 + \frac{-n}{n(n+1)}$$

$$= 1 - \frac{1}{n+1}$$ ✓

Rekord im Jahr i eines Jahrhunderts?

- Oder: Wahrscheinlichkeit, bei Position (i) "unterzugehen":



- "Durchmarsch" ist viel wahrscheinlicher, als an der Position davor "unterzugehen"

- Nur 50% Chance, über den ersten hinwegzukommen

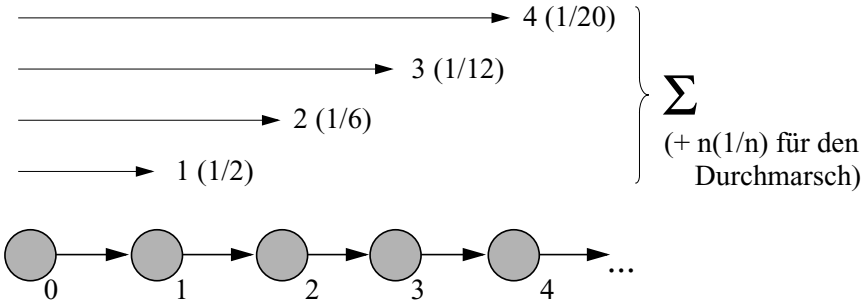
- p(*kein* echter Rekord in einem Jahrhundert) = 1/100,
- p(... in einem Jahrtausend) = 1/1000

Rekord würde jenseits liegen

Nun also: Wartezeit bis zum ersten Rekord

- Gesucht: Die *mittlere* Lebensdauer

- Dazu: Wahrscheinlichkeit mit Weglänge gewichten und alle Einzelfälle aufsummieren



- Daraus folgt allgemein für den Erwartungswert:

$$n\frac{1}{n} + 1\frac{1}{1\times2} + 2\frac{1}{2\times3} + 3\frac{1}{3\times4} + \ldots + (n-1)\frac{1}{(n-1)n} = \underline{\underline{H_n}}$$

Also: Eine von einem "x-beliebigen" Prozess gestartete Nachricht induziert im Mittel einen Weg der Länge H_n

- darf man das nun mit n multiplizieren, da jeder Prozess startet?
- ja; aber generell Vorsicht bei solchen Überlegungen!

Damit haben wir also nH_n als mittl. Nachrichtenkomplexität!

Qualität des Chang/Roberts-Algorithmus

- *Satz* (o. Bew.):

  Für unidirektionale Ringe der Grösse n und n Initiatoren ist n H_n die optimale mittlere Nachrichtenkomplexität beim *Election-Problem*

  - mit anderen Worten: es gibt keinen anderen Algorithmus, der "besser" ist!

  --> Chang/Roberts-Algorithmus ist "average case optimal"!

- *Satz* (Rotem et al., o. Bew.):

  Für die Nachrichtenzahl m des Chang/Roberts-Algorithmus gilt für alle ε: $\lim_{n \to \infty}$ prob $(m < (1+\varepsilon)\, n\, H_n) = 1$

  Interpretation: Bei grösseren Ringen werden fast nie mehr als n H_n Nachrichten benötigt

Tschebyscheff- / Chernoff-Ungleichungen

- Die Nachrichtenzahl des Chang/Roberts-Algorithmus lässt sich auch genauer abschätzen:

  - aus der *Tschebyscheff-Ungleichung* (engl.: "Chebyshev") folgt:

    $$\text{prob}\,(m \geq (1+\varepsilon)\, n\, H_n) \leq \frac{\pi^2/6}{\varepsilon^2\, H_n^{\,2}}$$

    (vgl. dazu Lehrbücher zur Wahrscheinlichkeitstheorie)

  - noch besser lässt sich dies mit der *Chernoff-Ungleichung* abschätzen:

    $$\text{prob}\,(m \geq (1+\varepsilon)\, n\, H_n) \leq \exp(-\varepsilon^2\, n H_n/3)$$

- Einige Beispiele (Wahrscheinlichkeit, dass die Nachrichtenzahl unter dem angegebenen Wert liegt):

	Tschebyscheff	*Chernoff*
ε=1, n=10	0.31	0.00046
ε=1, n=100	0.078	10^{-67}
ε=0.5, n=100	0.31	10^{-17}
ε=0.1, n=100	--	0.2

  - die Tschebyscheff-Ungleichung ist also recht konservativ!
  - auch die Chernoff-Ungleichung ist nur eine Abschätzung; *die Nachrichtenzahl liegt also fast immer sehr nahe am Mittelwert!*

Zur Chernoff-Ungleichung vergleiche man z.B.:

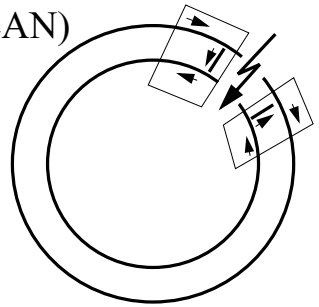
- C. Lavault: Evaluation des algorithmes distribués. Hermes, Paris 1995
- T. Hagerup. C. Rüb: A Guided Tour of Chernoff Bounds. Inform. Proc. Lett. 33, 305-308, 1990

Fehlertoleranz?

- Was geschieht bei Fehlern? ← z.B. beim Election-Algorithmus
  - was für Fehler können auftreten?
  - welche Fehlerarten betrachtet man? --> *Fehlermodell*
- Es werden typischerweise bei verteilten Systemen verschiedene *Fehlerarten* betrachtet (die unterschiedlich schwer zu behandeln sind), z.B.:
  - *crash* oder *failstop*: Prozess stoppt und bleibt gestoppt
  - *omission* (oder *lossy channel*): Nachricht geht unterwegs verloren
  - *link failure*: Kommunikationsverbindung geht (dauerhaft) kaputt
  - *byzantine*: Prozess verhält sich beliebig falsch (generiert illegale Nachrichten,...)
  - *timing* oder *performance*: bei synchronen bzw. Realzeitsystemen
- *Fehlertolerante* Algorithmen sollen robust bezüglich des betrachteten Fehlermodells sein
- Ob man überhaupt Fehler in Betracht zieht und welches Fehlemodell adäquat ist, ist ein *pragmatischer* Aspekt
  - abhängig von den Anforderungen, der Systemumgebung, Einsatzgebiet...
- Problem der *Fehlerentdeckung*: wie unterscheidet man:
  - Ausfall eines benachbarten Prozesses,
  - Ausfall der Kommunikationsverbindung,
  - extreme Nachrichtenverzögerung (länger als der timeout)?
- Fehlerentdeckung in der Praxis i.a. über *timeouts*
  - z.B. kein Acknowledge innerhalb eines bestimmten Zeitintervalls
  - dann *vermutet* man einen Fehler bzw. *verdächtigt* einen Prozess, ausgefallen zu sein (damit man sich da nicht täuscht, ist es manchmal besser, einen so verdächtigen Prozess nochmals explizit auszuschalten!)

Fehlerverhalten des Election-Algorithmus?

- Bei Ausfall einer Kommunikationsverbindung im Ring kann der Algorithmus nicht mehr funktionieren
- In der Praxis (z.B. Token-Ring-LAN) verwendet man daher meistens "Doppelringe"
  - Nachbarstationen schliessen den Ring kurz
  - zweite Unterbrechung wird so allerdings nicht toleriert!



- Konsequenz von Prozessausfällen?
  - in der Praxis hat das Ringinterface einer Station ein "Bypass-Relais" (damit wird die ausgefallene Station einfach kurzgeschlossen)
- Aber: Was geschieht bei folgendem Szenario?
  - Prozess mit der höchsten Identität sendet eine Nachricht aus
  - crasht (und wird kurzgeschlossen), bevor er "seine" Antwort vom Ring nehmen kann
  - diese kreist nun ständig weiter --> Algorithmus terminiert nicht!
- Denkübungen:
  - wie kann man obigen Fall verhindern (den Algorithmus also tolerant gegenüber diesem Fall machen)?
  - was geschieht, wenn ein anderer Prozess entsprechend ausfällt?
  - wie macht man den Algorithmus fehlertolerant gegenüber sporadisch verlorenen Nachrichten?

Praxisrelevanz

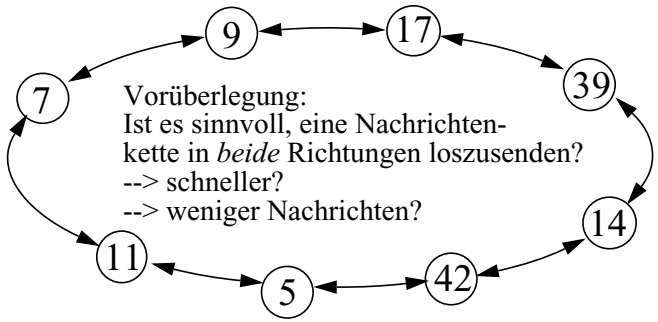
- Für die Praxis sind diverse Aspekte von Algorithmen relevant
  - z.B. Fehlertoleranz bzgl. Verlust von Nachrichten
  - oder: Crash eines Prozesses, der mitten im Senden ist (was geschieht mit Nachrichtenbruchstücken?)
  - oder: Erkennen verfälschter Nachrichten

- Denkübung: Kann beim Election-Verfahren der Fall behandelt werden, dass zwei verschiedene Stationen (versehentlich) die gleiche Identität haben?

- Der Election-Algorithmus muss wiederholt ausführbar sein
  - was geschieht, wenn in einer Ausführung des Algorithmus alte (sehr langsame) Nachrichten einer früheren Ausführung "dazwischenfunken"?
  - wie lässt sich dieses Problem behandeln?

==> Einfache algorithmische Ideen werden in der Praxis oft in komplexe Algorithmen eingebettet, um den ganzen pragmatischen Ansprüchen gerecht zu werden!

Probabilistisches Election-Verfahren für bidirektionale Ringe



- Message-extinction-Prinzip in naheliegender Weise verallgemeinern:

I_p: {M = 0}
M := p;
Wähle mit Wahrscheinlichkeit 1/2 eine Richtung;
send <M> **to** Nachbar in diese Richtung ;

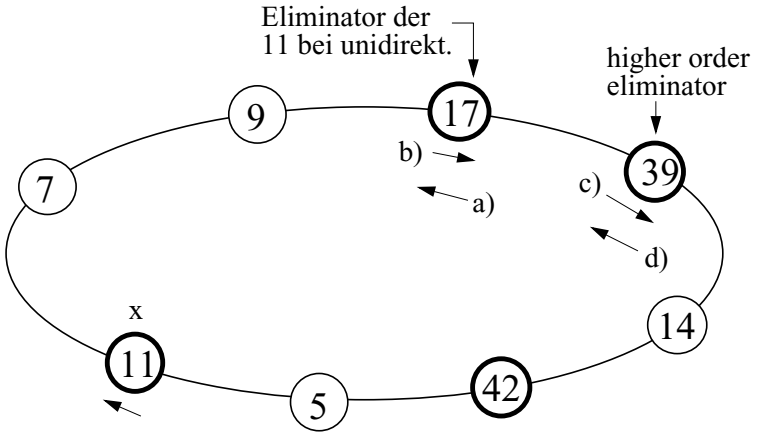
Identität des Prozesses

Uhrzeigersinn oder Gegen-Uhrzeigersinn

R_p: {Eine Nachricht <j> ist eingetroffen}
if M < j **then**
  M := j;
  send <M> **to** ... /* an anderen Nachbarn */
fi /* *weiter*senden */
if j = p **then** *"I am the master"* **fi**

- ist Wahrscheinlichkeit 1/2 eine gute Wahl?
- geht es nicht besser deterministisch als probabilistisch?
- wie hoch ist die mittlere Nachrichtenkomplexität?

Mittlere Nachrichtenkomplexität



- Annahme: Jede Einzelnachricht braucht gleich lang

- In der Hälfte aller Fälle kommt der Eliminator der Nachricht "x" auf halbem Weg entgegen
  - z.B. Fall a) bei den Knoten 11 und 17
  --> sollte 1/4 aller Nachrichten (gegenüber unidirekt. Fall) sparen (wieso?)

- Aber: höchste läuft immer ganz durch
  (jedoch: spielt für n --> ∞ eine "asymptotisch geringe" Rolle)

- Asymptotische mittlere Nachrichtenkomplexität ist *geringer* als 0.75 n ln n (Grund: "Higher order eliminators")
  - z.B. 39 verkürzt den Weg der 11 "etwas" im Fall b), d)
  - 42 als higher order eliminator würde hier aber nichts nützen!

Resultat von Lavault (Beweis schwierig!):
Asymptot. Nachrichtenkomplexität n --> ∞ ist

$$0.5\ \sqrt{2}\ n \ln n \approx 0.7071\ n \ln n$$

Als *untere Schranke* für das bidir. Election-Problem kennt man 0.5 n ln n (--> "Lücke")

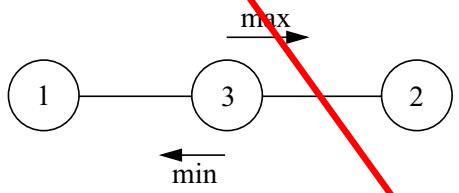
Deterministische bidirektionale Verfahren

(1) "Gerader" Prozess startet im Uhrzeigersinn, andere gegen den Uhrzeigersinn

- "common sense of orientation" vorhanden?
- aber: wenn Identifikationen keine ganze Zahlen (sondern z.B. rationale)?

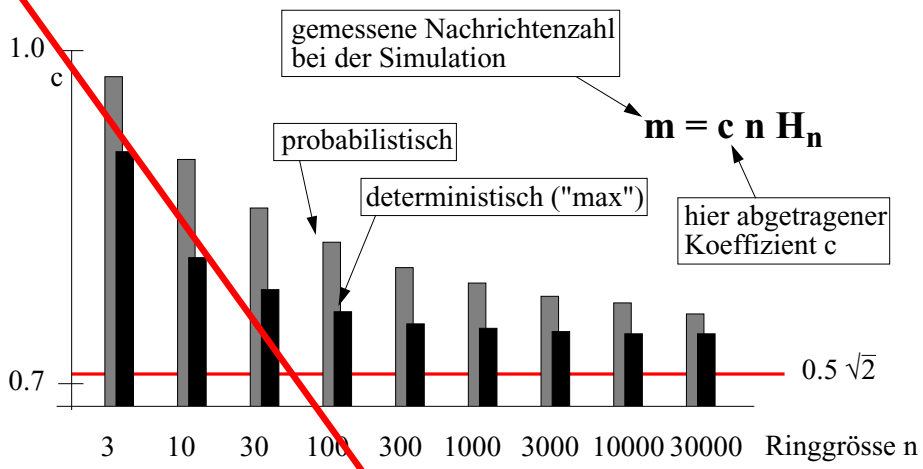
(2) Starten in Richtung des kleineren Nachbarn ("min")

- kennt man Identität der Nachbarn? (wenn nicht, was dann?)
- wieso nicht in Richtung des *grösseren* Nachbarn ("max")?
- was bringen die deterministischen Verfahren "min", "max" im Vergleich zum probabilistischen Verfahren?



Startsituation bzgl. Knoten 3

Gemessene Nachrichtenkomplexität



- Simulationsergebnisse:

  - Es stellt sich heraus, dass "min" und "max" bzgl. der mittleren Nachrichtenkomplexität etwa identisch sind (!), dass diese Varianten jedoch etwas besser sind als die probabilistische Version (und diese, wie gezeigt, besser als die unidirektionale)

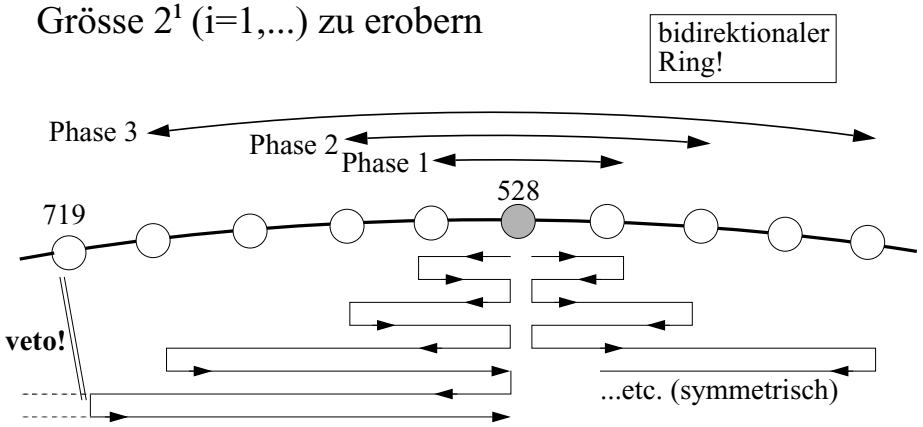
  - Simulationen zeigen auch, dass die asymptotische Nachrichtenkomplexität des probabilistischen Verfahrens lediglich ein theoretisches Ergebnis ist und der Faktor 0.7071... sehr langsam (mit steigendem n) approximiert wird

  - Ferner zeigen die Simulationen, dass die *Abweichungen vom Mittelwert* $n H_n$ bzgl. der Nachrichtenkomplexität i.a. nur sehr gering sind; 100000 Simulationen bei einer Ringgrösse von 20 lieferten z.B. stets Nachrichtenzahlen unter $2 n H_n$

  - Beachte bei *Simulationsexperimenten*: sehr viele Einzelexperimente (Varianz, statistisch relevante Ergebnisse) sowie guter Zufallszahlen-generator notwendig

Hirschberg / Sinclair-Election-Algorithmus

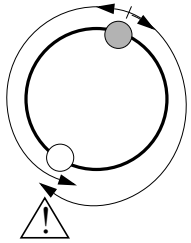
- Idee: Jeder Knoten versucht, sukzessive Gebiete der Grösse 2^i (i=1,...) zu erobern



- Ein unterwegs angetroffener grösserer Knoten legt Veto ein
  --> Initiator über Rückmeldung informieren
  --> Initiator wechselt von *aktiv* nach *passiv*

nur noch Nachrichten weiterleiten ("relay")

Gewinnsituation:



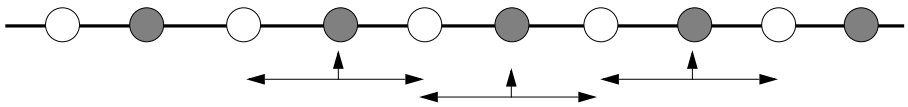
- Nachricht läuft in bereits selbst erobertes Gebiet
- *oder*: Nachricht trifft bei Initiator selbst wieder ein
- es bleibt genau ein Gewinner! (wieso?)

Komplexitätsanalyse

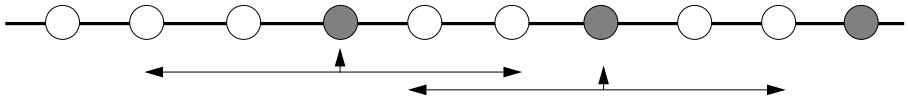
- Ein Prozess kann nur dann eine Kette der Länge 2^i starten, wenn er im Abstand 2^{i-1} in beiden Richtungen überlebt hat
  - Dichte überlebender Prozesse nimmt also exponentiell ab
- Innerhalb eines Bereiches von $1+ 2^{i-1}$ benachbarter Prozesse kann also höchstens einer eine Kette der Länge 2^i starten

---

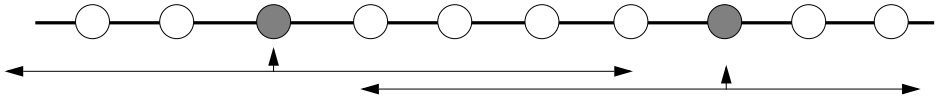
- Jeweils mind. 1 dazwischenliegender Prozess nach Phase 1:



- Jeweils mind. 2 dazwischenliegende Prozesse nach Phase 2:



- Jeweils mind. 4 dazwischenliegende Prozesse nach Phase 3:



- max. n/2 Prozesse können Ketten der Länge 2 initiieren
- max. n/3 Prozesse können Ketten der Länge 4 initiieren
- max. n/5 Prozesse können Ketten der Länge 8 initiieren

...

- max. $n/(1+2^{i-1})$ Prozesse können Ketten der Länge 2^i initiieren

Maximal 8 n $\log_2$ n Nachrichten

- Also: höchstens $n/(1+2^{i-1})$ Prozesse initiieren eine Nachrichtenkette der Länge 2^i in Phase i
- Bei jeder solchen Kette wird jede Kante max. 4 Mal durchlaufen
- In Phase i gibt es also höchstens $4 \times 2^i \times n / (1+2^{i-1}) < 8n$ Nachrichten
- Es gibt höchstens $1+\lceil \log_2 n \rceil$ Phasen

==> ca. 8 n $\log_2$ n $\approx$ 5.55 n ln n Nachrichten *maximal*

(Worst-case-Komplexität!)

- Aber: Wie hoch ist die *mittlere* Nachrichtenkomplexität?
- *Zeitkomplexität*: $2 + 4 + 8 + 16 +... + 2^i < 2^{i+1} \approx 4n$

Vergleiche dies alles mit dem Chang/Roberts-Algorithmus

- welcher Algorithmus ist "in der Praxis" besser?

Synchrone <--> asynchrone Phasen

- Die Phasen der einzelnen Initiatoren müssen nicht unbedingt synchron laufen!

- Damit der Algorithmus dann noch gut funktioniert, vereinbare folgendes:

  - anstelle von Knotenidentitäten betrachtet man das Paar (Phasennummer, Knotenidentität)

  - diese Paare werden lexikographisch geordnet (eine höhere Phasennummer hat also Priorität!)

- Konsequenz: Ein "schneller" Initiator gewinnt gegenüber einem "langsamen" Initiator mit höherer Identität

- Es gewinnt also zwar ein eindeutiger Knoten die Election, das muss aber nicht derjenige mit der grössten Identität sein!

- Unterscheide also:

  - *leader election problem*
  - *maximum finding problem*

  !

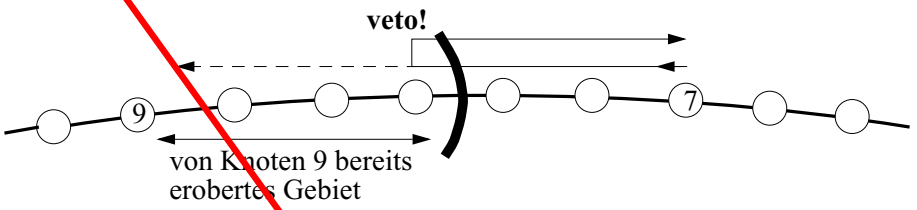
- eine Lösung des maximum finding problems ist immer auch eine Lösung des leader election problems (sofern die Knoten eindeutig numeriert sind)

- Umkehrung?

Optimierungen und Varianten

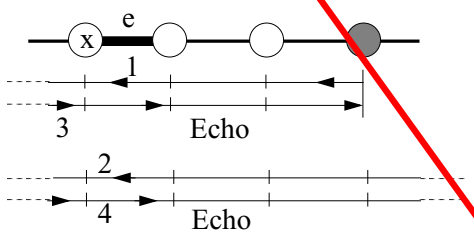
1.) *Auch passive Knoten legen Veto ein:*

(sofern ein grösserer Eroberer zuerst da war) --> verkürzt Nachrichtenketten



- hierzu muss sich jeder Knoten merken, von wem er (zuletzt) erobert wurde (inkl. der Phasennummer)

2.) *Zwei "Echos" zusammenfassen, wenn möglich:*



- Optimierung: Wenn x erst Nachricht 1 erhält, dann Nachricht 2 (vor dem "Echo" 3), dann kann x die beiden Nachrichten, die über e zurückgeschickt werden, zusammenfassen zu einer einzigen physischen Nachricht

---

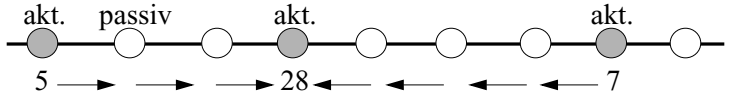
Wie wirken sich diese Optimierungen auf die Nachrichtenkomplexität aus?
(Wer simuliert den Algorithmus und ermittelt die Nachrichtenkomplexität?)

Peterson's Election-Algorithmus (1. Variante)

- Prinzip analog zum Hirschberg / Sinclair - Verfahren: Anzahl aktiver Knoten pro "Phase" mindestens halbieren
  - bidirektionaler Ring
  - anfangs sind alle *aktiv*
  - *passive* Knoten reichen nur noch Nachrichten weiter ("relay")

- Idee: Pro Phase bekommt ein Knoten die Identitäten seiner rechten und linken *noch aktiven* Nachbarn...

Vgl. dies mit iterierter Anwendung des Algorithmus für Nachbarschaftswissen!



...und überlebt nur, wenn er der grösste *aller drei* ist!

Im Unterschied zu Hirschberg / Sinclair gibt es keine Echos / Vetos!

- Ein Überlebender bleibt aktiv und startet eine neue Phase: Sendet seine Identität in beide Richtungen
  (Initial tun das alle Initiatoren)

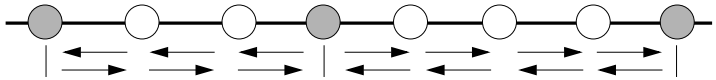
- *Gewonnen*, wenn die eigene Identität empfangen wird

Beachte: In obigem Beispiel wird die 5 von der 28 "passiviert". Bald darauf (in der nächsten Phase) erhält die 5 erneut eine Nachricht "28", um diese weiterzuleiten. Hätte die 5 nicht gleich beim ersten Mal die "28" einfach weiterleiten sollen, so dass Knoten 28 die Nachricht nicht erneut über die Strecke 28 --> 5 senden muss? (Vgl. Chang/Roberts!) Nein! Knoten 5 weiss nicht, ob die 28 die gegenwärtige Phase tatsächlich überlebt - dann wäre die Nachricht "28" fälschlicherweise weitergeleitet worden!

Zeitkomplexität des Algorithmus als Übung
(dominiert auch hier die letzte Phase?)

Nachrichtenkomplexität

- Pro Phase laufen 2 Nachrichten über *jede* Kante
  - für global *synchrone Phasen* leicht einsichtig
  - aber auch für nicht synchronisierte Phasen richtig!

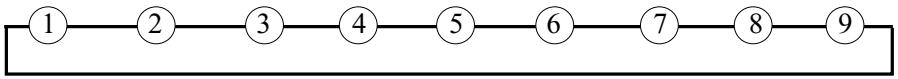


- Wenn ein Knoten Phase i überlebt, überlebt sein "linker" aktiver Nachbar diese Phase nicht!
  --> max. $\log_2 n$ Phasen
  --> max. $2 n \log_2 n$ Nachrichten

"in jeder Phase überlebt einer von *dreien*" ist *falsch* - wieso?

- wie sieht eine Anordnung aus, bei der *maximal viele* Nachrichten entstehen?

- *Sortierte Anordnung*: jeweils durch "rechten" Nachbarn eliminiert, ausgenommen grösster Knoten im Ring



--> in diesem Fall nur 2 Phasen ==> 4n Nachrichten!
(beachte Terminierungserkennung!)

- Mittlere Nachrichtenkomplexität:
  - ein Knoten überlebt eine Phase mit Wahrscheinlichkeit 1/3
  - es gibt also im Mittel $\log_3 n$ Phasen
    --> $2 n \log_3 n \approx 1.26 n \log_2 n \approx 1.82 n \ln n$ Nachrichten