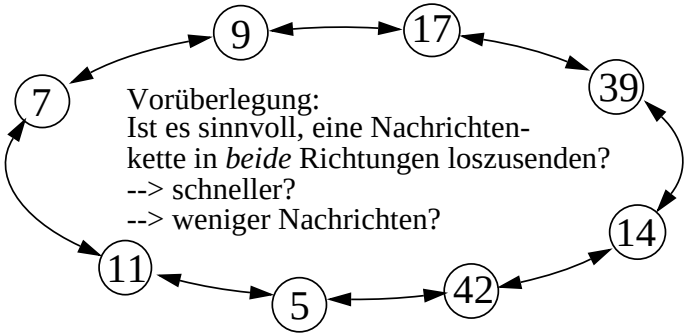
Probabilistisches Election-Verfahren für bidirektionale Ringe



- 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 ;

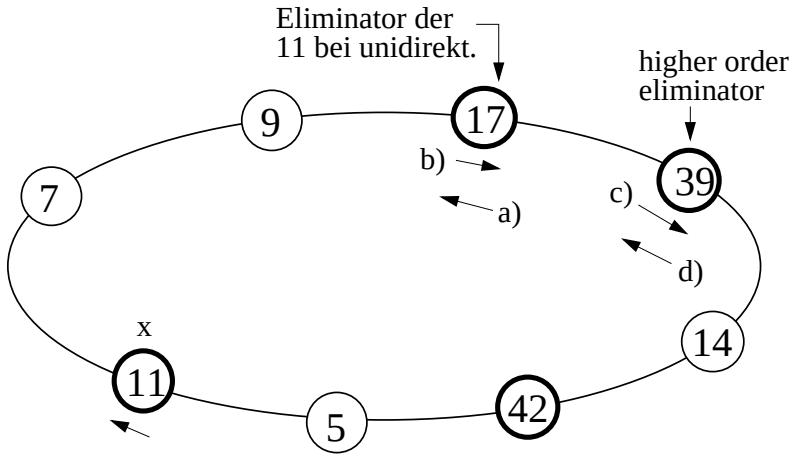
Identität des Prozesses

Uhrzeigersinn oder Gegen-Uhrzeigersinn

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

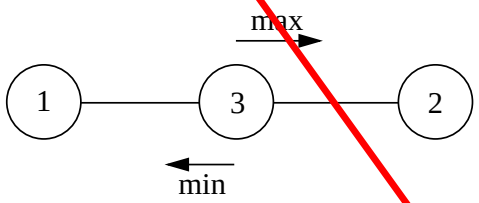
Deterministische bidirektionale Verfahren

(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)?

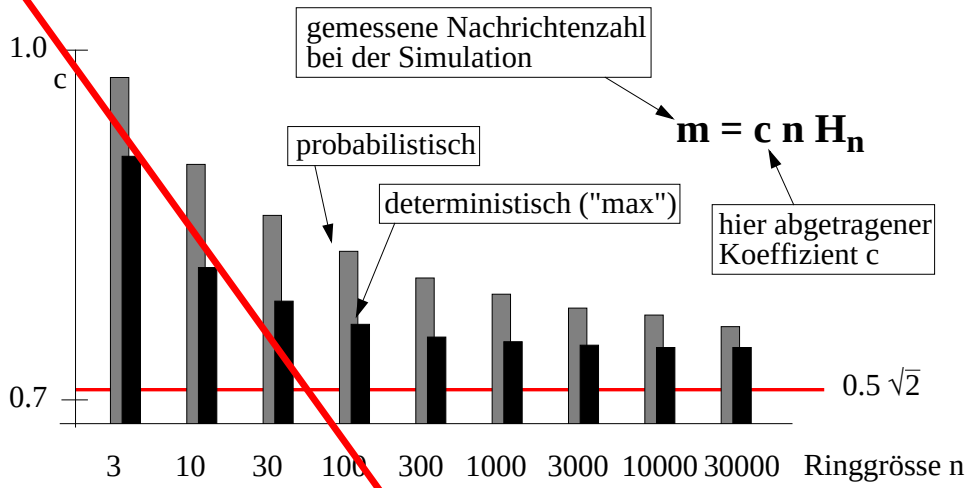
(2) Starten in Richtung des kleineren Nachbarn ("min")

- 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

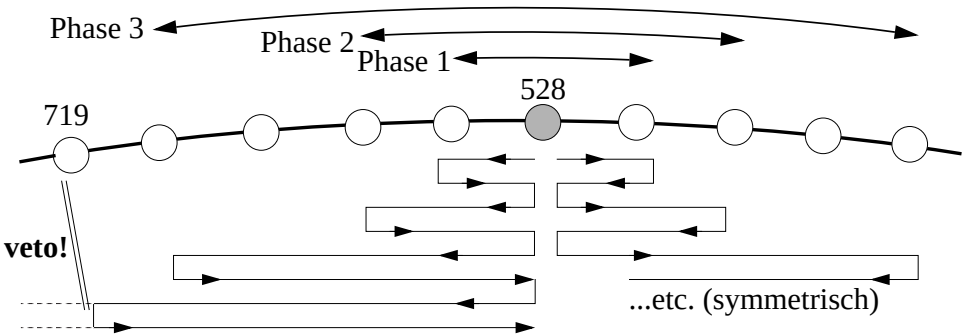
  - Ferner zeigen die Simulationen, dass die *Abweichungen vom Mittelwert* $n H_n$ bzgl. der Nachrichtenkomplexität i.a. nur sehr gering sind; 100000 Simulationen bei einer Ringgrösse von 20 lieferten z.B. stets Nachrichtenzahlen unter $2 n H_n$

  - 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

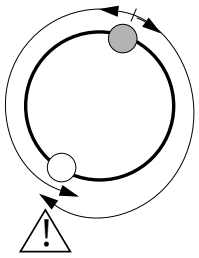
bidirektionaler Ring!



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

nur noch Nachrichten weiterleiten ("relay")

Gewinnsituation:



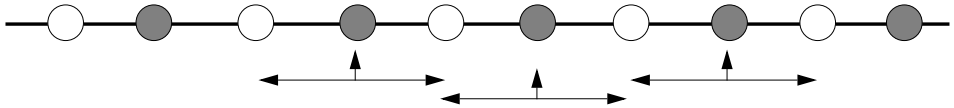
- Nachricht läuft in bereits selbst erobertes Gebiet
- *oder*: Nachricht trifft bei Initiator selbst wieder ein
- es bleibt genau ein Gewinner! (wieso?)

Komplexitätsanalyse

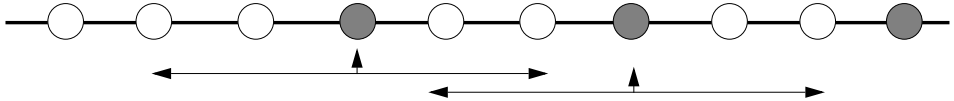
- Ein Prozess kann nur dann eine Kette der Länge 2^i starten, wenn er im Abstand 2^{i-1} in beiden Richtungen überlebt hat
  - Dichte überlebender Prozesse nimmt also exponentiell ab
- Innerhalb eines Bereiches von $1+ 2^{i-1}$ benachbarter Prozesse kann also höchstens einer eine Kette der Länge 2^i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

---

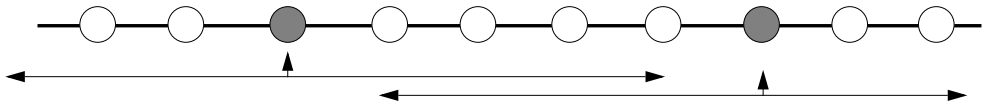
- Jeweils 1 dazwischenliegender Prozess nach Phase 1:



- Jeweils 2 dazwischenliegende Prozesse nach Phase 2:



- Jeweils 4 dazwischenliegende Prozesse nach Phase 3:



- n/2 Prozesse können Ketten der Länge 2 initiieren
- n/3 Prozesse können Ketten der Länge 4 initiieren
- n/5 Prozesse können Ketten der Länge 8 initiieren
  ...
- $n/(1+2^{i-1})$ 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*

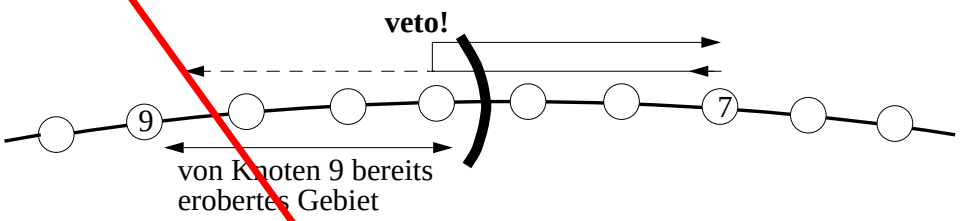
- eine Lösung des maximum finding problems ist immer auch eine Lösung des leader election problems (sofern die Knoten eindeutig numeriert sind)

- Umkehrung?

Optimierungen und Varianten

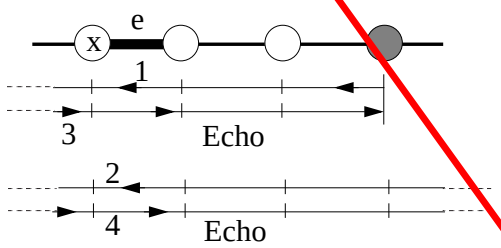
1.) *Auch passive Knoten legen Veto ein:*

(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

---

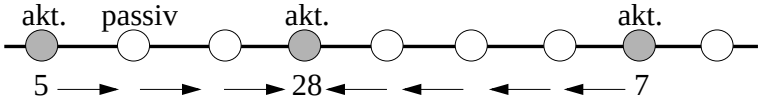
Wie wirken sich diese Optimierungen auf die Nachrichtenkomplexität aus?
(Wer simuliert den Algorithmus und ermittelt die Nachrichtenkomplexität?)

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)

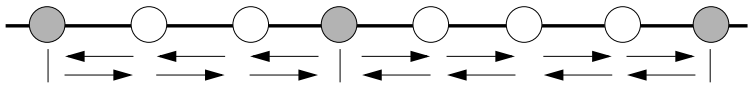
- *Gewonnen*, wenn die eigene Identität empfangen wird

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!

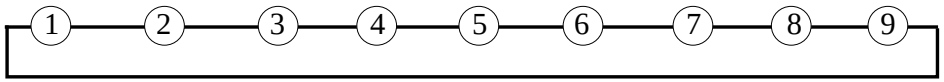
  --> max. $\log_2 n$ Phasen

  --> 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:
  - für jeden Knoten Wahrscheinlichkeit 1/3, Phase zu überleben (wieso?)
  - 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

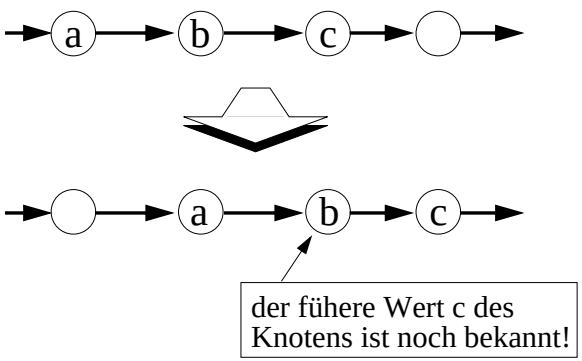
Variante für unidirektionale Ringe

- Man glaubte zunächst, dass ein Election-Algorithmus auf *unidirektionalen* Ringen mindestens $O(n^2)$ Nachrichten im Worst-case-Fall benötigt

- Das stimmt nicht: Der Peterson-Algorithmus lässt sich auf unidirektionalen Ringen *simulieren*!

---

1. "Shift" in Ringrichtung um eine Position bzgl. aktiver Knoten:



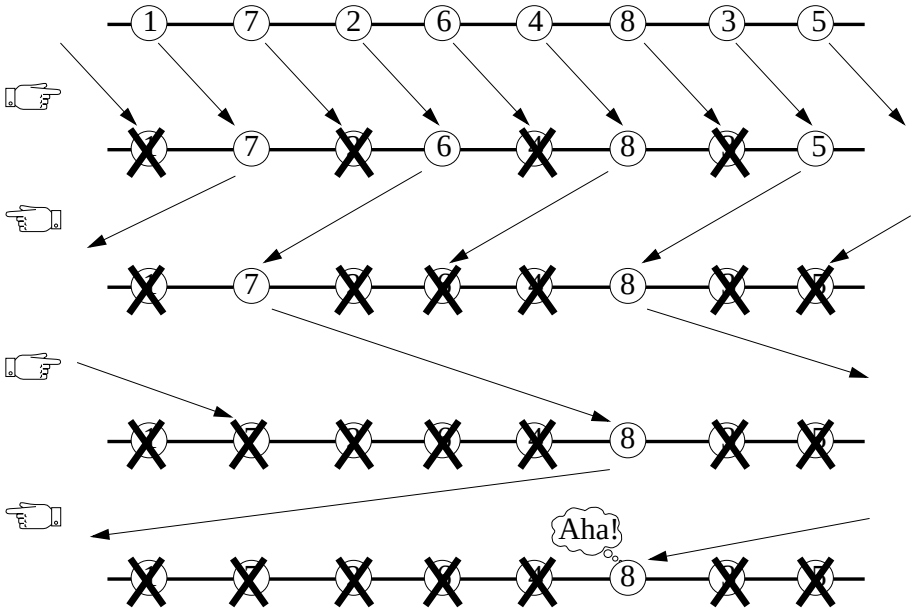
2. Nun kann (der neue) Knoten a an seinen Nachbarn b seinen Wert senden - damit kennt b sowohl a als auch c
   (als hätte b Nachrichten von a und c erhalten!)

Damit kostet eine solche Phase global auch nur 2n Nachrichten!

Peterson's Election-Algorithmus (2. Variante)

- Idee einer Optimierung:
  - anstatt sich mit beiden Nachbarn "gleichzeitig" zu vergleichen, sollte ein Knoten sich nur dann mit seinem anderen Nachbarn vergleichen, wenn er den ersten Vergleich gewonnen hat

- Phasen im / gegen den Uhrzeigersinn wechseln sich ab:



  - lässt sich auch wieder unidirektional simulieren!
  - in jeder Phase werden n Nachrichten gesendet (passive Knoten: "relay")

- Denkübungen:
  1) Wie kann man auch bei asynchronen Nachrichten und nicht gleichzeitigem Start der Knoten "eine Art" global getakteter Phasen erreichen?
  2) Man formuliere den Algorithmus aus "Sicht eines Knotens": Wie reagiert ein Knoten auf das Eintreffen einer bestimmten Nachricht?
  3) Man mache sich Gedanken zur Abschätzung der worst-case und der average-case Nachrichtenkomplexität!

Nachrichtenkomplexität

- **Behauptung:**
  Für die *Anzahl der Phasen* r gilt: $r \leq \log_{\varphi} n + O(1)$ (Basis $\varphi = (1+\sqrt{5})/2$)
  ==> Anzahl der Nachrichten $\leq 1.44\, n \log_2 n + c$

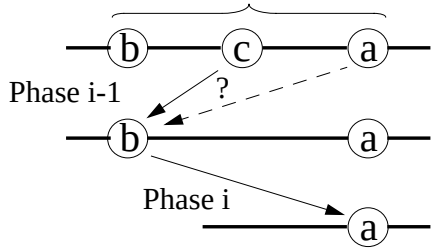
- **Lemma:** $a_i \leq a_{i-2} - a_{i-1}$ (für $i > 1$)

Def: Anzahl Überlebende von Phase i

Anzahl der "Opfer" von Phase i-1

Bew.: Betrachte zwei benachbarte Knoten a, b in Phase i

Gab es einen Knoten zwischen a und b in Phase i-1?



(1) a überlebe Phase i
==> a > b

(2) b hat Phase i-1 überlebt
==> b > c

(1) und (2) ==> $a \neq c$

Also muss es ein c geben, das in Phase i-1 Opfer wurde
Hier: a hat seinen linken Nachbarn in der vorherigen Phase verloren

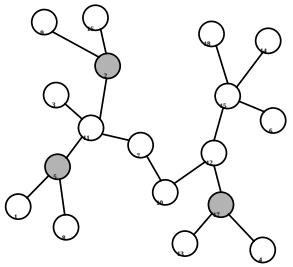
==> Für jeden Überlebenden in Phase i (hier: a) gibt es mindestens ein Opfer (hier: c) in Phase i-1 □

- Aus $a_i \leq a_{i-2} - a_{i-1}$ folgt $a_{i-2} \geq a_{i-1} + a_i$, also $a_i \geq a_{i+1} + a_{i+2}$
- Ferner gilt $a_{r-1} = 1 = Fib(2)$, $a_{r-2} \geq 2 = Fib(3)$ } $a_{r-3} \geq a_{r-2} + a_{r-1} \geq Fib(2)+Fib(3) = Fib(4)$
- Also: $n = a_0 \geq Fib(r+1)$

--> Ungleichung nach r auflösen!
Weil Fib exponentiell zur Basis φ wächst ($Fib(k) \approx \varphi^k / \sqrt{5}$), folgt die Behauptung

Election auf Bäumen

- Geht dies besser / effizienter als z.B. mit dem Message-extinction-Prinzip für allg. Graphen?
- Und im Vergleich zu den Verfahren auf Ringen?



- Explosionsphase:
  - Election-Ankündigung wird zu den Blättern propagiert

flooding!

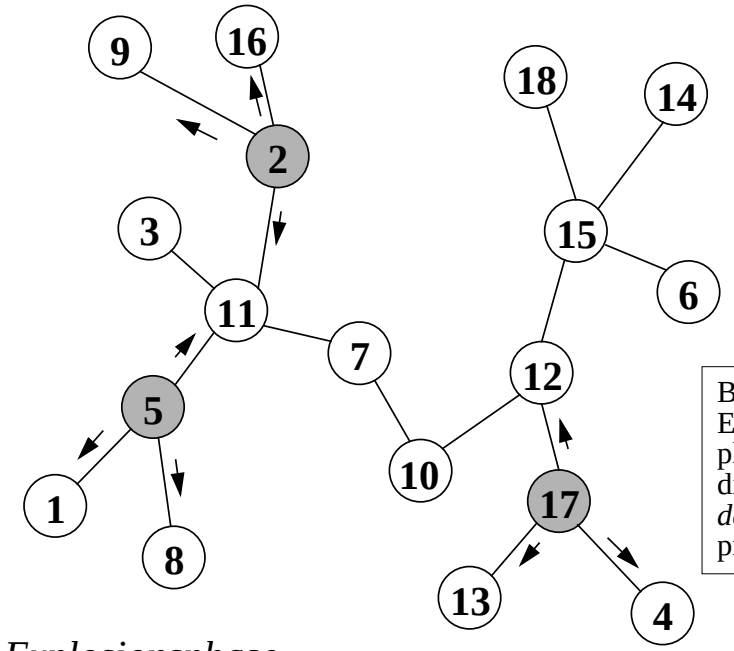
- Kontraktionsphase
  - von aussen zum "Zentrum" das Maximum propagieren

0 für unbeteiligte Knoten

- Informationsphase (notw.?)
  - Zentrum informiert alle Knoten über Gewinner

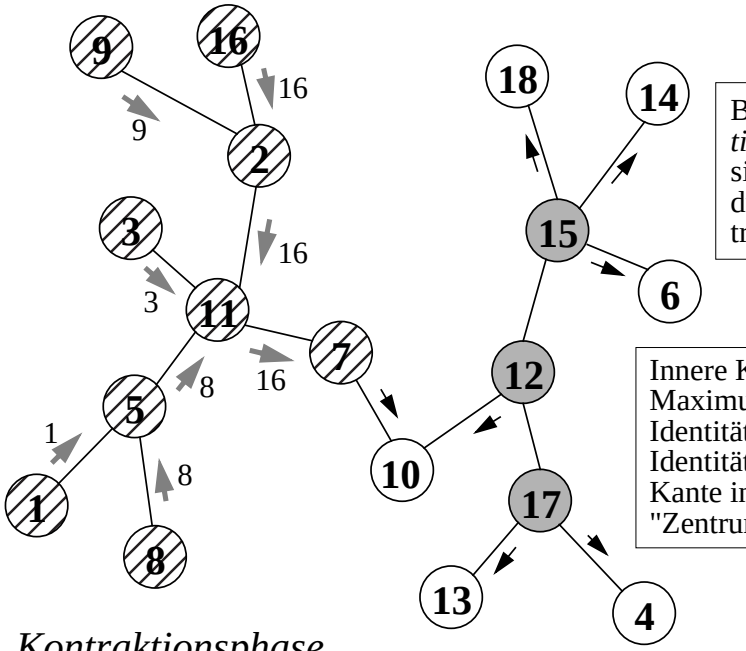
flooding!

- Explosionsphase kann an mehreren Stellen "zünden"
- "Vereinigung" der Explosionsphasen
- Ggf. Teile in Explosionsphase während andere Teile schon in Kontraktionsphase
- Zentrum ist nicht determiniert



Bei erstmaligem Erhalt einer Explosionsnachricht diese in alle *anderen* Richtungen propagieren

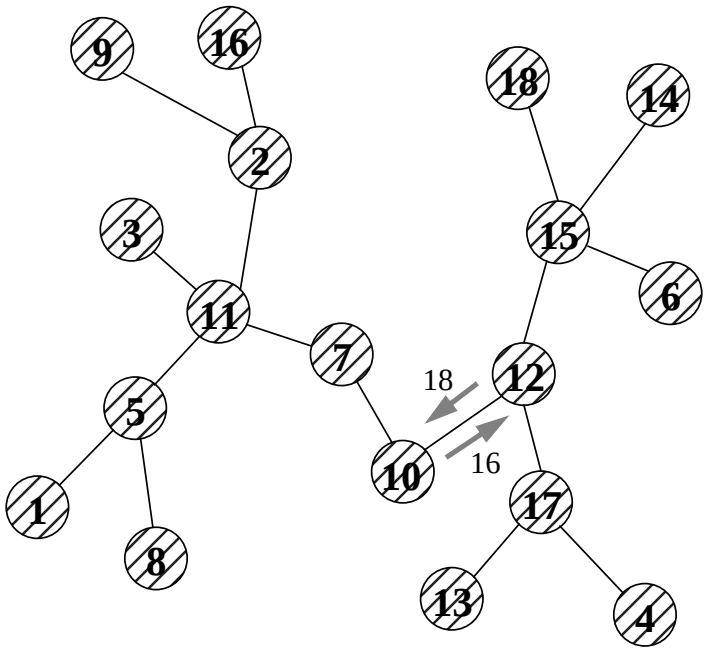
Explosionsphase



Blätter *reflektieren* Explosionsnachricht durch eine Kontraktionsnachricht

Innere Knoten senden Maximum aus erhaltenen Identitäten und eigener Identität über die "letzte" Kante in Richtung "Zentrum"

Kontraktionsphase



- Begegnung zweier Kontraktionsnachrichten auf (genau) einer Kante im Zentrum

- Die beiden Knoten wissen, dass sie nun das Maximum kennen

- Sie können dies nun ggf. per flooding verbreiten (Informationsphase)

- Terminierung der flooding-Phase (falls notwendig) einfach durch erneute Reflexion / Kontraktion ("indirektes acknowledge")

Nachrichtenkomplexität von Baumelection

Folgender Satz / Beweis ist *falsch* - wieso?

Beh.: Der Baumelection-Algorithmus hat bei *einem* Initiator und n Knoten die Komplexität m(n) = 2n - 2
(ohne Berücksichtigung der Informationsphase)

Beweis induktiv:

1) n=1 --> m(1) = 0 ✓ (offensichtlich korrekt)

2) Schritt von n auf n+1:

- Füge an einen Baum aus n Knoten ein Blatt an; über die neue Kante fliessen genau 2 Nachrichten

- Also: m(n+1) = m(n)+2 = 2n-2+2 = 2(n+1)-2 ✓

- wo genau liegt der Fehler?

- korrekter Wert der Nachrichtenkomplexität --> nächste Folie!

Nachrichtenkomplexität

(1) Explosionsphase: n-2+k

Anzahl der Initiatoren

- es gibt k-1 Begegnungskanten von Explosionsnachrichten

(2) Kontraktionsphase: n

- über alle Kanten eine Nachricht, nur über die Zentrumskante zwei

(3) Informationsphase: n-2

- keine Nachricht über die Zentrumskante

$\sum = 3n + k - 4$ (mit Information aller Knoten)

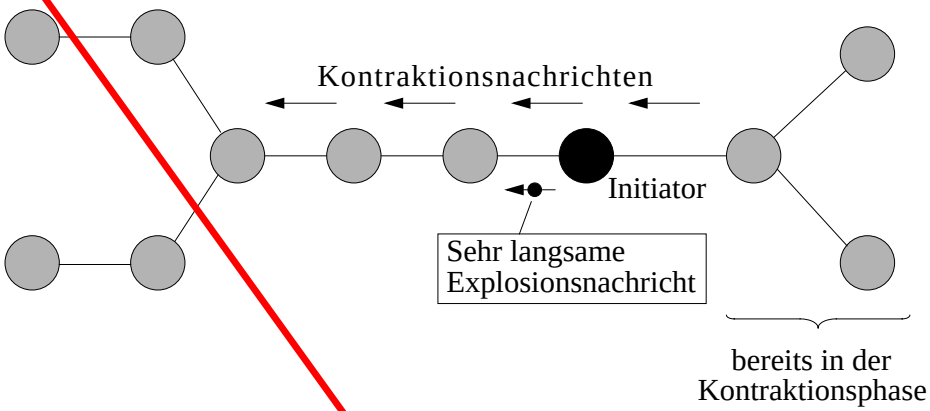
--> 3(n-1) für k=1

--> 4(n-1) für k=n

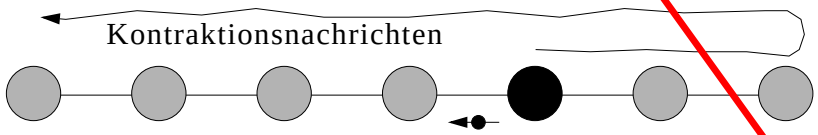
- Wesentlich effizienter als Ringe!
- Wieso? (Ringe sind symmetrischer!)
- Wieso Verfahren nicht "einfach" auf Ringe anwenden?

(eine Kante entfernen)

Schaden Nachrichtenüberholungen?



- Kann eine Kontraktionsnachricht eine Explosionsnachricht überholen?
- Muss man das vermeiden?
- Lassen sich vielleicht sogar z.T. Nachrichten durch Zusammenfassen (Explosions / Kontraktion) sparen?

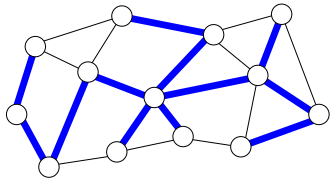


- Wie würde in diesem Fall das Ende erkannt?

Election auf allgemeinen Graphen

zusammenhängend

- Wieso versagt folgende einfache Idee für allg. Graphen?
  - verwende den Echo-Algorithmus, um einen (einzigen) Spannbaum zu konstruieren
  - führe dann Election auf diesem Baum aus



- Wie wäre es damit:
  - jeder Initiator startet seinen eigenen Echo-Algorithmus, mit dem er (über die Echo-Nachrichten) die grösste Identität erfährt
  - jeder Initiator weiss somit, ob er der grösste ist oder nicht und kennt auch den grössten
  - vgl. dies mit dem "bully-Algorithmus" für Ringe: jeder macht einen vollständigen (!) Ringdurchlauf und prüft dabei, ob er der grösste ist
  - ist das korrekt?
  - effizient?

"Echo-Election" für allgemeine Graphen

zusammenhängend mit bidirekt. Kanten

- *Generelle Idee*: Chang / Roberts-Algorithmus (also "message exctinction"), jedoch *Echo-Algorithmus* anstelle des zugrundeliegenden Ring-Verfahrens

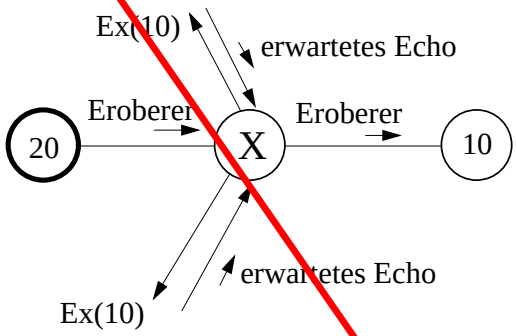
- Also:
  - Jeder Initiator startet "seinen" Echo-Algorithmus
  - Explorer und Echos führen die Identität des Initiators mit
  - Schwächere Nachrichten (Explorer und Echos) werden "verschluckt" (d.h. nicht weitergegeben)
  - Stärkste Welle setzt sich überall durch (und informiert so neben dem Gewinner auch alle Verlierer)
  - Alle anderen Wellen stagnieren irgendwo endgültig
    (zumindest der Gewinner sendet keine Echos für diese Wellen
    --> kein anderer Initiator bekommt jemals ein Echo)

- *Veranschaulichung*: "Gleichzeitiges" Einfärben des Graphen mit verschieden dominanten Farben

- Vermutung bzgl. der worst-case und der average-case Nachrichtenkomplexität?

Varianten, z.B. "Adoption"

- Idee: Stärkerer Knoten ("Eroberer") läuft nicht besiegten Explorern hinterher, sondern "adoptiert" dessen Echos



Beispiel: Knoten X wird erst von 10, dann von 20 erobert.

Eroberer-Nachrichten (= Explorer des stärkeren) laufen "direkt" in Richtung des gegnerischen Initiatorknotens

- Kommt statt erwartetem Echo dann ein fremder Eroberer:
  - Fremder > : verloren, dieser erobert nun...
  - Fremder < : Jetzt doch Explorer in diese Richtung senden, da der "Vasall" offenbar nicht stark genug war, um sich durchzusetzen

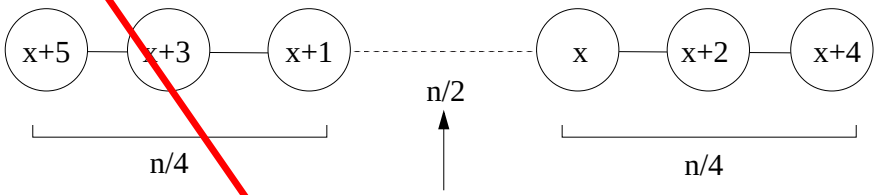
---

- *Eigenschaften* dieser Variante:
  - Nur ein einziges Echo pro Knoten (das ist oft praktisch!)
    (Interpretation: "Verschiedene" Echo-Wellen vereinigen sich)
  - Nicht jeder Knoten wird über seinen Misserfolg informiert
    (Gewinner kann aber über "seinen"Baum eine Informationswelle starten)

- Es sind noch einige andere Varianten denkbar...

Nachrichtenkomplexität von Echo-Election

- *Worst-case* Nachrichtenkomplexität ist $O(n^2)$



Strecke kann u.U. von jedem der n/2 skizzierten Initiatoren durchlaufen werden (wenn diese geeignet zeitversetzt "zünden")

- *Mittlere* Nachrichtenkomplexität ist $O(e \log k)$

Anzahl der Initiatoren

Hier nur *Beweisskizze:*

- Ein Knoten wird im Mittel $H_k \approx \log k$ mal erobert (--> Anzahl der Rekorde!)
- Eroberter Knoten sendet e/n Nachrichten im Mittel (Explorer und Echos)
  --> n H_k (e/n) = e H_k Nachrichten insgesamt.

- Bemerkungen:
  - Empirische Untersuchungen zeigen, dass die Adoptionsvariante bei typischen Graphen und mehreren Initiatoren ca. 30% - 50% derNachrichten spart. (Bei nur einem Initiator sind die Verfahren identisch!)
  - Es gibt Verfahren mit geringerer worst-case Nachrichtenkomplexität, diese sind allerdings um einiges aufwendiger!

Nachrichtenkomplexität: untere Schranke

Satz: *Die Nachrichtenkomplexität für das Election-Problem in allg. Graphen beträgt mindestens* $\Omega(e + n \log n)$

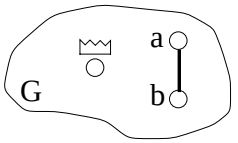
was dominiert typischerweise?

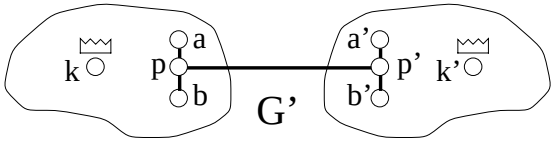
Beweis: *n log n* ist untere Grenze, da der Satz auch für Ringe gilt (vgl. dortigen Satz ohne Beweis)

Also noch zu zeigen: *e* ist untere Schranke

Widerspruchsbeweis: Angenommen, es gäbe einen Election-Algorithmus A, der weniger als e Nachrichten für einen Graphen G benötigt ==> es gibt eine Kante $\overline{ab}$, über die *keine Nachricht* fliesst

Konstruiere dann G' aus zwei Kopien von G (mit unterschiedlichen Identitäten aber der gleichen relativen Ordnung) so, dass diese mit einer Kante $\overline{pp'}$ verbunden werden (p bzw. p' werden in die unbenutzten Kanten $\overline{ab}$ bzw. $\overline{a'b'}$ neu eingefügt)





Da zwischen den beiden Teilen keine Nachricht ausgetauscht wird, gewinnen u.U. *zwei* Prozesse k, k'!

Beachte bei diesem Beweis:

- bei Anwendung von Algorithmus A auf G' kann sich jeder Knoten genauso wie der enstprechende im Graphen G verhalten
- die Knoten haben (weder in G noch in G') ein "globales Wissen": sie kennen nicht die Struktur oder Gesamtgrösse des Graphen, sie kennen nicht die Identitäten ihrer Nachbarn
- die Knoten wissen insbesondere nicht, ob Situation G oder G' vorliegt
- Knoten p und p' seien keine Initiatoren
- bzgl. der Knotenidentitäten wird nur vorausgesetzt, dass auf diesen eine lineare Ordnung definiert ist (nicht, dass es sich um Nummern handelt)