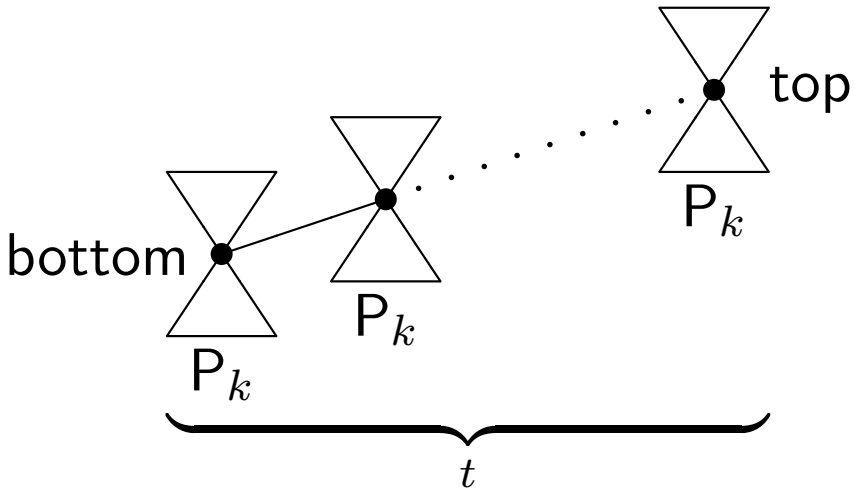
Kette von P_k's:



Gesamtzahl der Elemente:

n

$t(2k+1)$ in den P_k's

$r = n - t(2k+1)$ Rest

Wenn $r < t - 1$, dann wissen wir, dass top größer ist als

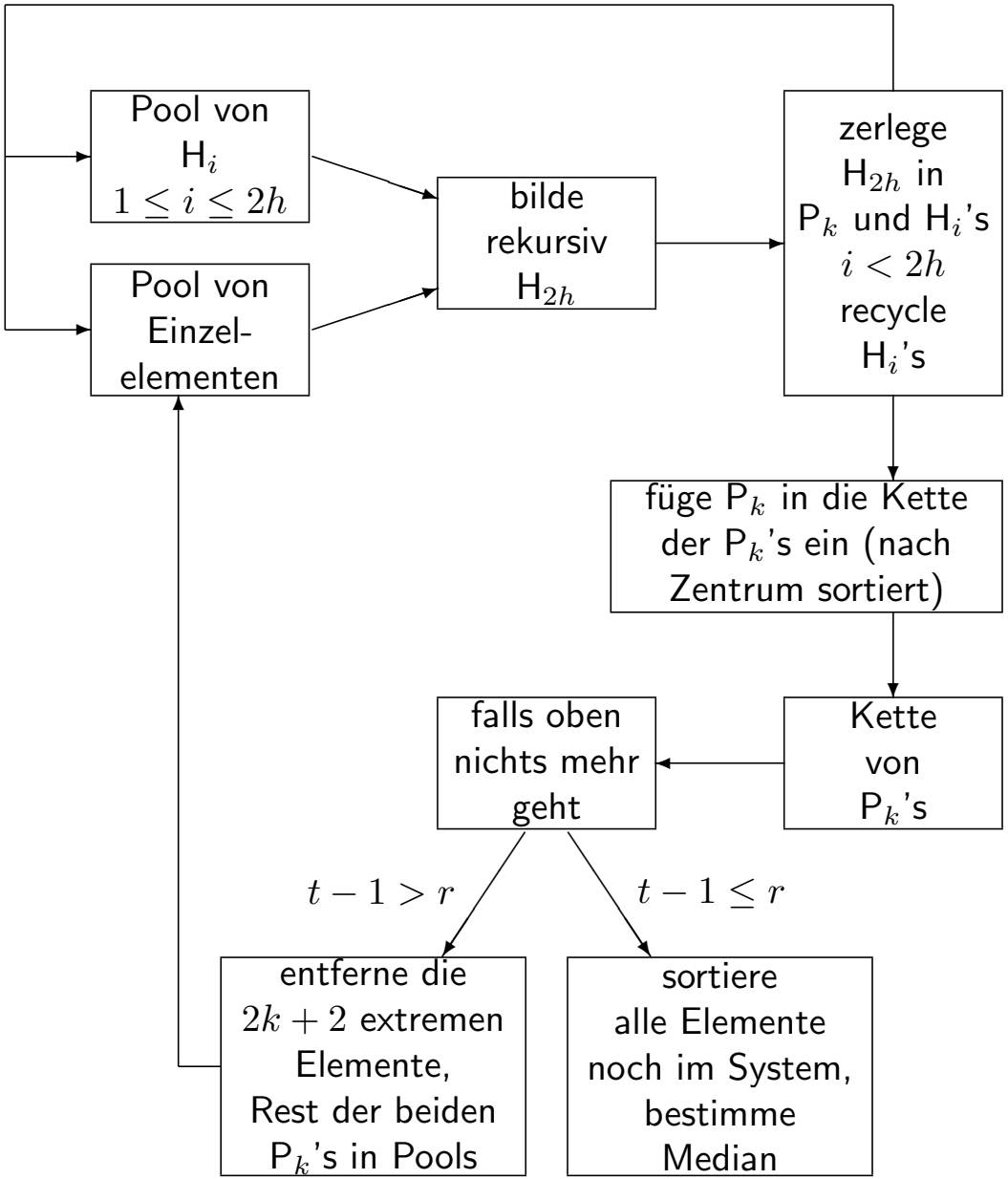
$$k + (t-1)(k+1) > k + (k+1)\left(\frac{n+1}{2k+2} - 1\right) = \frac{n-1}{2}$$

$\Rightarrow$ top $>$ Median

Setze

$$k := \left\lfloor n^{\frac{1}{4}} \right\rfloor$$

h sdg. $2^{h-1} \leq k < 2^h$


Pool von H_i $1 \leq i \leq 2h$
Pool von Einzel-elementen
bilde rekursiv H_{2h}
zerlege H_{2h} in P_k und H_i's $i < 2h$ recycle H_i's
füge P_k in die Kette der P_k's ein (nach Zentrum sortiert)
Kette von P_k's
falls oben nichts mehr geht
$t-1 > r$
$t-1 \leq r$
entferne die $2k+2$ extremen Elemente, Rest der beiden P_k's in Pools
sortiere alle Elemente noch im System, bestimme Median

Definiere $r :=$ Anzahl der noch im H_{2h}-Produktionsprozess steckenden Elemente (für jedes $i < 2h$ höchstens ein H_i, daher $r \leq \sum_{i=0}^{2h-1} 2^i = 2^{2h} - 1$).

$R :=$ Anzahl der im letzten Schritt zu sortierenden Elemente. Es gilt: $t \leq r + 1$, und damit

$$R = t(2k+1) + r \leq 2^{2h}(2k+1) + 2^{2h} - 1\,.$$

$m :=$ Gesamtzahl der im Algorithmus produzierten P_k's.

$$m = t + 2\frac{n-R}{2(k+1)} = t + \frac{n-R}{k+1}\,.$$

Gesamtzahl der vom Algorithmus durchgeführten Vergleiche =

1. Anzahl der Kanten in allen P_k's
2. $+$ Anzahl der Kanten, die gelöscht werden, um die P_k's zu formen
3. $+$ Anzahl der Kanten, die zum Schluss in übriggebliebenen H_i's, $i < 2h$, stecken
4. $+$ Anzahl der Vergleiche, um jedes Zentrum der P_k's in die (sortierte) Kette einzufügen
5. $+$ Anzahl der Vergleiche, um die zum Schluss übriggebliebenenen R Elemente zu sortieren

$$\leq \left(\frac{n-R}{k+1}+t\right)\left[\underbrace{2k}_{1}+\underbrace{3k+2h}_{2}+\underbrace{\log\frac{n}{2k+1}}_{4}\right]+\underbrace{R\log R}_{5}+\underbrace{r}_{3}\,.$$

Mit $k = \left\lfloor n^{\frac{1}{4}} \right\rfloor$, h so, dass $2^{h-1} \leq k < 2^h$ ergibt sich damit

$$\begin{aligned} r &= \mathcal{O}(k^2) \\ t &= \mathcal{O}(k^2) \text{ zum Schluss} \\ R &= \mathcal{O}(k^3), \text{ und damit die} \\ \text{Anzahl der Vergleiche } &= T(n) \leq 5n + o(n)\,. \end{aligned}$$

Verbesserte Version (besseres Zurechtschneiden, bessere Verwertung der Reste):

$$T(n) = 3n + o(n)$$

Bester bekannter Algorithmus (von Dor/Zwick):

$$2,95n + o(n)$$

Literatur:

Arnold Schönhage, Michael Paterson, Nicholas Pippenger:
Finding the median
J. Comput. Syst. Sci. **13**, pp. 184–199 (1976)

Dorit Dor, Uri Zwick:
Selecting the median
SIAM J. Comput. **28**(5), pp. 1722–1758 (1999)

5. Eine untere Schranke für die Medianbestimmung

Satz 89

Jeder (vergleichsbasierte) Medianalgorithmus benötigt im worst-case mindestens $\left\lceil \frac{3n}{2} \right\rceil - 2$ *Vergleiche.*

Beweis:

Gegenspielerargument (adversary argument)

n Elemente, o.B.d.A. n ungerade, alle Elemente paarweise verschieden. Die Menge aller Elemente wird in drei Teilmengen partitioniert: U enthält die Kandidaten für den Median, G enthält Elemente, die sicher größer als der Median sind, und L enthält Elemente, die sicher kleiner als der Median sind. Anfangs sind alle Elemente in U.

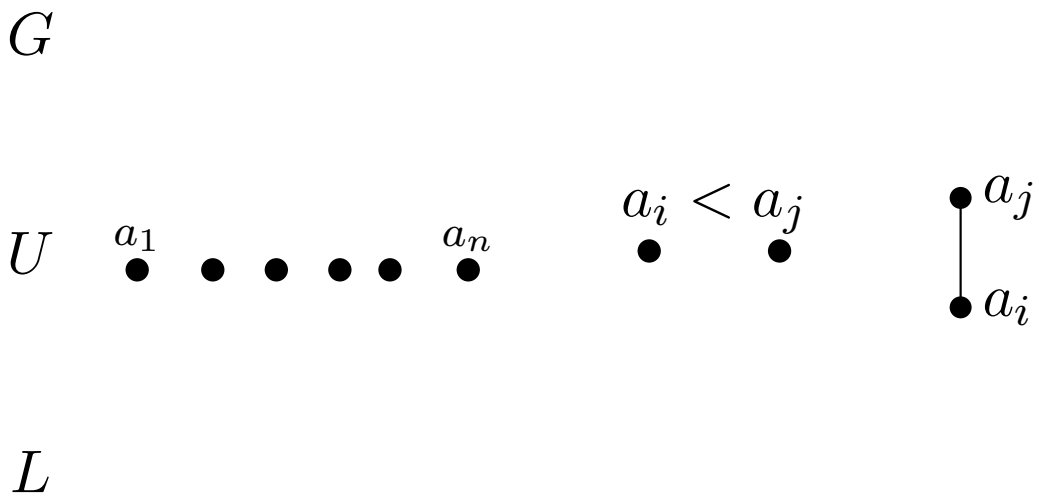
Beweis (Forts.):

Der Algorithmus stellt nun Fragen der Form $a_i < a_j$. Der Gegenspieler gibt konsistente Antworten, die den Algorithmus jedoch dazu zwingen, möglichst viele Fragen stellen zu müssen, bevor die Antwort feststehen kann(d.h. U soll möglichst ungeordnet bleiben).

Durch die (konsistenten!) Antworten des Gegenspielers auf die "$\overset{?}{<}$"-Queries des Algorithmus entsteht ein DAG. Der Gegenspieler hält diesen DAG "einfach", z.B. angenommen $y > z$ und $y > x$, dann soll y „sehr groß" sein $\Rightarrow y \to G$.

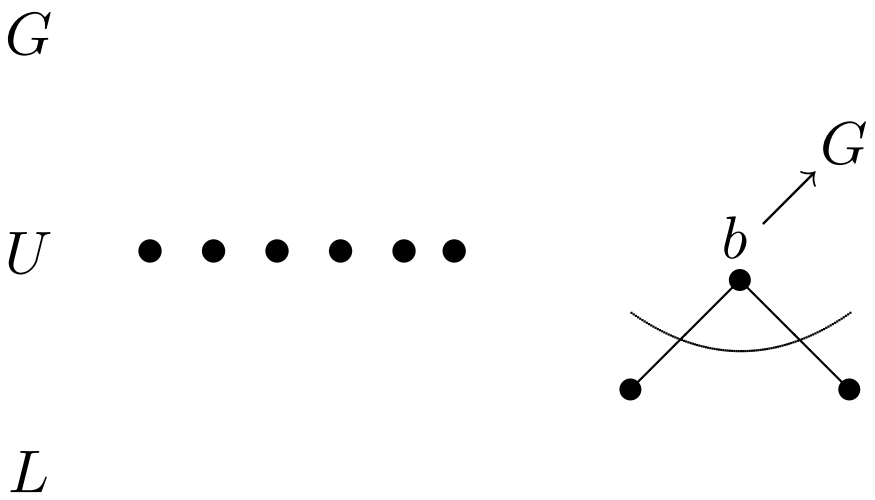
Strategie des Gegenspielers

Beweis (Forts.):



Strategie des Gegenspielers

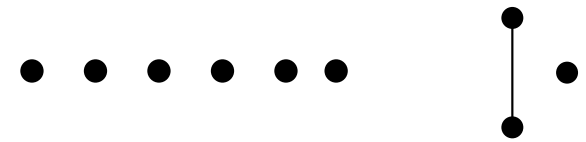
Beweis (Forts.):



Strategie des Gegenspielers

Beweis (Forts.):

$G \quad \leq \frac{n+1}{2} - 1$

U 

$L \quad \leq \frac{n+1}{2} - 1$

Solange ein Element unverglichen ist, ist nicht klar, welches der Median ist.

Beweis (Forts.):

Solange $|L|, |G| < \frac{n+1}{2}$, kann der Algorithmus annehmen, dass der Median in U ist. Solange U mindestens zwei unverglichene Elemente und keine Zusammenhangskomponente mit > 2 Elementen enthält, kann der Algorithmus den Median nicht bestimmen.

Strategie des Gegenspielers

Beweis (Forts.):

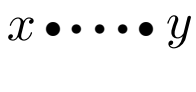
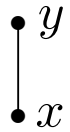
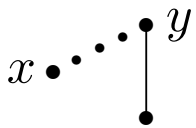
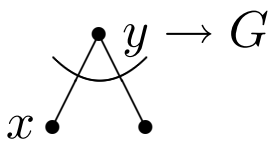
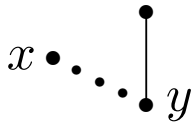
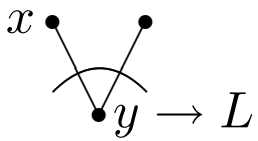
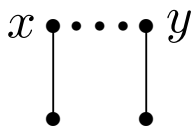
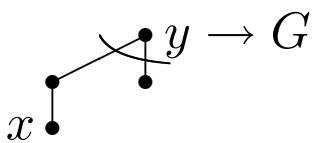
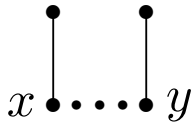
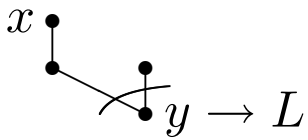
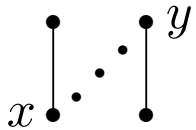
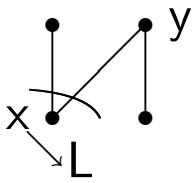
Die Strategie des Gegenspielers hat zwei Phasen.

Erste Phase: Query sei $x \overset{?}{<} y$:

i) $x, y \in G$ bzw. $x, y \in L$: irgendeine konsistente Antwort

ii) $x \in G \wedge y \in L \cup U$ (bzw. $x \in L \wedge y \in G \cup U$):
Antwort: $y < x$ (bzw. $x < y$).

iii) Sei $x, y \in U$.

	Query	Antwort des Gegenspielers	Anzahl der Paare in U	$\lvert U\rvert$	$\lvert L\rvert$	$\lvert G\rvert$
1.			$+1$	—	—	—
2.			-1	-1	0	$+1$
3.			-1	-1	1	0
4.			-1	-1	0	$+1$
5.			-1	-1	$+1$	0
6.			-1	-1	$+1$	0

Die erste Phase endet, wenn $|L| = \frac{n-1}{2}$ oder $|G| = \frac{n-1}{2}$. Während der Phase 1 enthält U mindestens zwei (in U) maximale und mindestens zwei (in U) minimale Elemente (bzgl. des DAGs). $\Rightarrow$ Während Phase 1 kann der Algorithmus den Median mit Sicherheit nicht bestimmen.

Der Gegenspieler beginnt mit Phase 2, sobald

$$|L| \text{ wird } \frac{n-1}{2} \text{ oder } |G| \text{ wird } \frac{n-1}{2}.$$

O.B.d.A.:

$$|L| = \frac{n-1}{2}$$

Der Gegenspieler zwingt nun den Algorithmus, das minimale Element in U bzgl. der gesamten totalen Ordnung zu bestimmen (da dieses unter den Vorgaben der Median ist).

Beweis (Forts.):

Laufzeitanalyse:

$C :=$ Anzahl der Vergleiche

$P :=$ Anzahl der Paare in U

i) In der Phase 1 gilt folgende Invariante: $C - P + 2|U| \geq 2n$.
Dies wird durch vollständige Induktion gezeigt:
Induktionsanfang: $0 - 0 + 2n \geq 2n$;
Induktionsschritt: Gemäß der Tabelle.

ii) Sei C die Anzahl der Vergleiche am Ende der Phase 1. In der Phase 2 werden noch $\geq |U| - 1 - |P|$ Vergleiche nötig. Die Anzahl der Vergleiche für alle Phasen ist damit

$$\geq C + |U| - 1 - |P| \,.$$

Beweis (Forts.):

Am Ende der Phase 1 gilt ja $C \geq 2n + |P| - 2|U|$ (wg. Invariante).
Damit gilt für die Anzahl der Vergleiche:

$$\geq 2n + |P| - 2|U| + |U| - 1 - |P| = 2n - |U| - 1$$

$$\geq \frac{3}{2}n - \frac{3}{2} =$$

$$= \left\lceil \frac{3}{2}n - 2 \right\rceil, \text{ da } |U| \leq \frac{n+1}{2}$$

□

6. Eine bessere untere Schranke

Satz 90

Sei T ein Entscheidungsbaum für die Bestimmung des i-kleinsten von n verschiedenen Elements, mit $i^2 \geq \log\left[\binom{n}{i}\frac{1}{n-i+1}\right] + 3$. Dann gilt, wenn

$$p := 2\sqrt{\log\left[\binom{n}{i}\frac{1}{n-i+1}\right] + 3} - 2$$

gesetzt wird,

$$\textit{Höhe}(T) \geq \log\left[\binom{n}{i}\frac{2^{n-p}}{n-i+1}\right].$$

Bemerkung: $i = \lceil \frac{n}{2} \rceil \rightarrow \binom{n}{\lceil \frac{n}{2} \rceil} = \Theta(\frac{2^n}{\sqrt{n}})$; also erhalten wir eine untere Schranke von Höhe$(T) \geq 2n - o(n)$ für die Bestimmung des Medians.

Beweis:

Der Beweis setzt sich aus einem Gegenspieler- und einem Abzählargument zusammen, um zu zeigen

„T hat viele Blätter.“

Sei A Teilmenge der Schlüssel, $|A| = i$. Wir konstruieren einen Teilbaum T_A von T, so dass alle Blätter von T_A auch Blätter von T sind, und zeigen: T_A hat viele Blätter (nämlich 2^{n-p}). Es gibt $\binom{n}{i}$ Möglichkeiten, A zu wählen, jedes Blatt von T kommt in höchstens $n - i + 1$ T_A's vor.

Beweis (Forts.):

Beobachtungen:

Jedes Blatt w von T liefert folgende Informationen:

$$\begin{aligned}
\text{answer}(w) &\xrightarrow{\text{liefert}} i\text{-kleinstes Element } x \\
\text{little}(w) &\xrightarrow{\text{liefert}} i-1 \text{ Elemente } < x \\
\text{big}(w) &\xrightarrow{\text{liefert}} n-i \text{ Elemente } > x
\end{aligned}$$

Beweis (Forts.):

Wähle A als beliebige Teilmenge der n gegebenen Schlüssel, mit $|A| = i$. Wir geben für den Gegenspieler eine Strategie an, welche dazu führt, dass wir durch Zurechtschneiden aus T einen Baum T_A konstruieren können, so dass gilt:

- T_A ist Binärbaum
- jedes Blatt von T_A ist auch Blatt von T, d.h. durch das Zurechtschneiden entstehen keine zusätzlichen Blätter
- für jedes Blatt w von T_A gilt: little$(w) \subset A$

Beweis (Forts.):

Setze $\bar{A}$ gleich dem Komplement von A, sowie

$$r := \sqrt{\log\left[\binom{n}{i}\frac{1}{n-i+1}\right] + 3}$$

$$s := r - 1$$

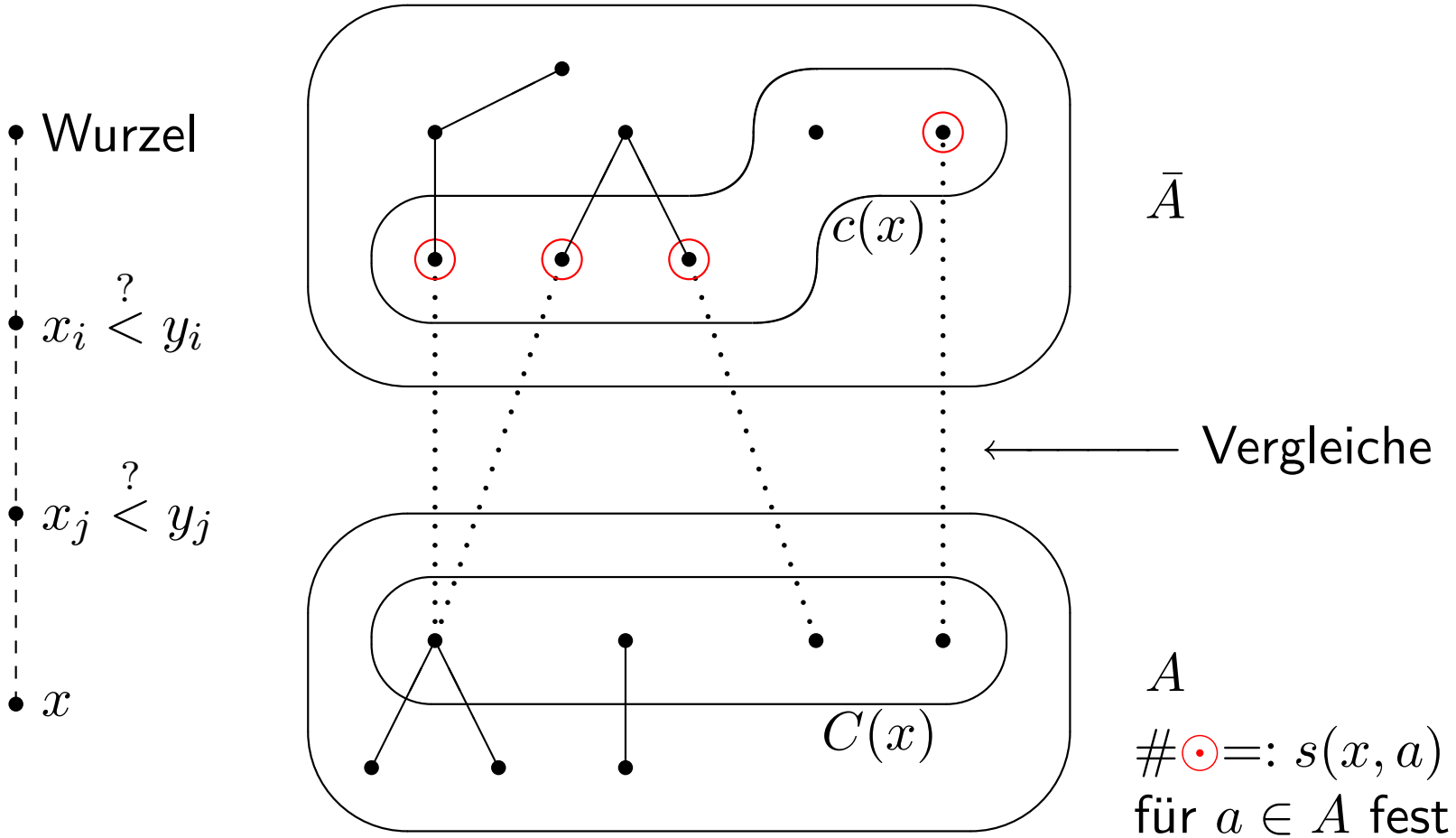
Damit gilt: $p = r + s - 1$.

Die Konstruktion (das "Zurechtstutzen") von T zu T_A erfolgt in zwei Phasen.

Beweis (Forts.):

Erste Phase: Breitensuche von der Wurzel nach unten.
Betrachte Knoten x. Definiere:

- $C(x)$ sind die Elemente $a \in A$, für die es kein $b \in A$ und keinen Knoten auf dem Pfad von der Wurzel von T zu x gibt, an dem a mit b mit dem Ergebnis $a < b$ verglichen wurde.
- $c(x)$ sind entsprechend die im Knoten x bekannten Minima in $\bar{A}$.
- $s(x, a)$ ist, für $a \in A$, die Anzahl der Elemente $\in c(x) \subseteq \bar{A}$, mit denen a auf dem Pfad von der Wurzel zu x verglichen wurde.


Wurzel
$x_i \overset{?}{<} y_i$
$x_j \overset{?}{<} y_j$
x
$c(x)$
$\bar{A}$
Vergleiche
$C(x)$
A
$\#\odot =: s(x,a)$
für $a \in A$ fest

Beweis (Forts.):

Regeln für Phase 1:

Seien a und b die Elemente, die im Knoten x verglichen werden.

1.1: Falls $a \in A$ und $b \in A$, behalte x in T_A bei.

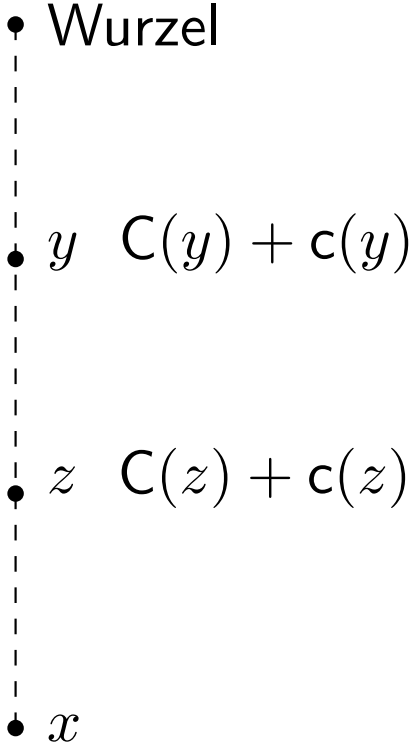
1.2: Falls $a \in \bar{A}$ und $b \in \bar{A}$, behalte x in T_A bei.

1.3: Sei nun o.B.d.A. $a \in A$ und $b \in \bar{A}$. Ersetze den Unterbaum in T mit Wurzel x mit dem Unterbaum, dessen Wurzel das Kind von x ist, das dem Ergebnis „$a < b$“ entspricht (d.h. lasse den Vergleich $a < b$ aus, da gemäß Strategie alle Elemente aus A kleiner als alle Elemente in $\bar{A}$ sind).

Phase 1 läuft, solange $|C(x)| \geq r$. Ein Knoten auf einem Pfad in T von der Wurzel, bei dem $|C(x)|$ erstmals $= r$ wird, heißt kritisch. Jeder Pfad in T von der Wurzel zu einem Blatt enthält genau einen kritischen Knoten.

Beweis (Forts.):

Betrachte in T_A einen Pfad von der Wurzel zu einem kritischen Knoten x. Sei y ein Knoten auf diesem Pfad, z sein Kind. Es gilt:



$$|C(z)| + |c(z)| \geq |C(y)| + |c(y)| - 1$$

Da $|C(\text{Wurzel})| = |A| = i$ und $|c(\text{Wurzel})| = |\bar{A}| = n - i$, müssen überhalb eines jeden kritischen Knoten x mindestens

$$i - |C(x)| + n - i - |c(x)| = n - r - |c(x)|$$

Vergleiche erfolgen. Von jedem kritischen Knoten abwärts arbeitet der Gegenspieler nach einer Strategie für Phase 2. Sei x ein solcher kritischer Knoten. Dann ist $|C(x)| = r$.