Union-Find-Laufzeitanalyse

Tobias Ludwig

1 Einleitung

1.1 Das Union-Find-Problem

Es soll eine Partition $(A_1, \cdots, A_n)$ einer endlichen Menge X verwaltet werden. Jeder Teilmenge A_i ist ein eindeutiger Repräsentant $p(A_i) \in A_i$ zugeordnet.

Operationen auf der Struktur:

- $FIND(x)\,, x \in A_i \subseteq X$ liefert den Repräsentanten $p(A_i)$, der Teilmenge, die x enthält.
- $UNION(p(A_i), p(A_j))$ vereinigt zwei Mengen, gegeben durch ihre Repräsentanten.

1.2 Datenstruktur

Die Verwaltung der Partition wird durch einen Wald F realisiert, wobei jede Teilmenge durch einen Baum dargestellt wird. Die Knoten besitzen nur einen Zeiger zu ihrem Vater. Somit werden die Wurzeln der Bäume zu den einzig sinnvollen Repräsentanten der Teilmengen.

Vorgehen bei den beiden Operationen:

- $FIND(x)$: Es wird solange den Elternzeigern gefolgt, bis die Wurzel erreicht ist. Diese wird dann zurück gegeben.
- $UNION(p(A_i), p(A_j))$: Eine der beiden Wurzeln wird zum Elternknoten der anderen, somit wurden die beiden Bäume zu einem vereint.

Daraus ergibt sich, dass die benötigte Zeit für eine *UNION*-Operation konstant ist, und die einer *FIND*-Operation proportional zur Pfadlänge ist. Daher ist es erstrebenswert, die Pfade so kurz wie möglich zu halten.

1.3 Vereinigung nach Rang

Der Rang eines Knotens gibt eine obere Schranke für die Höhe des Teilbaums unter ihm an, wobei der Rang eines kinderlosen Knotens 0 ist. Ohne Pfadkompression entspricht der Rang genau der Höhe des Teilbaums unter dem Knoten.

Fall 1: Bei unterschiedlichen Rängen wird der Baum mit dem geringeren Rang an die Wurzel des anderen gehängt und die Ränge bleiben unverändert.

Fall 2: Bei gleichen Rängen wird ein Baum an die Wurzel des anderen gehängt und deren Rang um 1 erhöht.

Daraus lassen sich folgende Eigenschaften ableiten:

1. Ein Baum dessen Wurzel Rang k hat, besteht aus $\geq 2^k$ Knoten.

   Beweis:

   IA: $k = 0$ Der Baum besteht nur aus der Wurzel: $1 \geq 2^0 = 1$ ✓

   IS: Ein Baum mit Rang $k+1$ kann nur entstehen, wenn zwei Bäume mit Rang k vereinigt werden. Diese haben beide nach IV $\geq 2^k$ Knoten. $\Rightarrow 2 \cdot 2^k \geq 2^{k+1}$ □

2. Der maximal mögliche Rang in einem Wald aus n Knoten ist $\lfloor \log_2 n \rfloor$. (folgt aus 1.)

3. Der Wald ist Rang balanciert, d.h.

   für jeden Knoten x gilt: $\forall\ 0 \leq i < Rang(x)\ \exists$ Kind x_j von x : $Rang(x_j) = i$.

   Beweis:

IA $Rang(x) = 0$: keine Kinder ✓

IS $Rang(x) = k+1$: Ein Knoten mit $Rang = k+1$ kann nur entstehen, indem ein Knoten x_1 mit $Rang(x_1) = k$ an einen anderen Knoten x_2 mit $Rang(x_2) = k$ gehängt wird. x_1 und x_2 sind nach IV Rang balanciert. Durch das Anhängen von x_1 hat x_2 nun neben den Kindern mit den Rängen $0, \ldots, k-1$ auch noch ein Kind mit Rang k erhalten. □

4. Der Rang jedes Knotens ist größer als der Rang seiner Kinder (nach Konstruktion).

2 Hauptlemma

Sei F ein Wald auf einer endlichen Menge X von Knoten.

Pfad: Folge $(x_1, \cdots, x_n)$ von Knoten, wobei x_{i+1} Vater von x_i ist.

Wurzelpfad: x_n ist eine Wurzel im Wald F

Nicht-Wurzelpfad: x_n ist keine Wurzel im Wald F, d.h. es gibt einen Vater $a(x_n)$ von x_n

Pfadkompression:

Nicht-Wurzelpfad: Jedem Knoten x_i wird $a(x_n)$ als Vater zugewiesen, daraus ergeben sich die Kosten von $n-1$.

Wurzelpfad: Jeder Knoten x_i wird zu einer Wurzel, die Kosten betragen 0.

(leere Pfade werden als Wurzelpfade angesehen, somit kostet ihre Komprimierung 0)

Sei $C = (p^{(i)})_{1 \leq i \leq M}$ eine Folgen von Pfaden, die in dem Wald $F =: F^{(0)}$ komprimiert werden. Dabei ensteht $F^{(i+1)}$ aus $F^{(i)}$ durch die Kompression von $p^{(i+1)}$. Bezeichne $cost(C)$ die auftretenden Kosten beim Komprimieren der Pfade aus C, d.h. wie oft ein Knoten einen neuen Vater bekommt.

$|C| :=$ Anzahl Nicht-Wurzelpfade in C.

Lemma 1. [1] *Sei S eine Folge von* UNION *und m* FIND-*Operationen, die auf einem Wald F aus einelementigen Bäumen auf der n-elementigen Menge X ausgeführt wird.*
Sei T die benötigte Zeit zum Ausführen von S.
Dann existiert ein Wald F' auf X und eine Folge C von höchstens[2] m Nicht-Wurzelpfaden, so dass

$$T \in O(m + n + cost(C))$$

Beweis:
Sei $F'^{(0)}$ der Wald, der durch alle *UNION*-Operationen aus S entstanden ist.
Sei S' die Menge der verbleibenden m *FIND*-Operationen aus S.
Die Folge $C = (p^{(i)})_{1 \leq i \leq m}$ von Nicht-Wurzelpfaden ergibt sich nun wie folgt:
Sei x_i der Knoten der i-ten *FIND*-Operation. Von x_i führt ein Wurzelpfad $p^{(i)}$ in $F'^{(i-1)}$ bis zur Wurzel des Baums, ohne diese ist es ein Nicht-Wurzelpfad. Durch das vorherige Ausführen aller *UNION* Operationen kann $p^{(i)}$ länger ausfallen, als wenn die i-te *FIND*-Operation in $F^{(j)}$ ausgeführt worden wäre. Genauso kann $p^{(i)}$ kürzer ausfallen. Das passiert jedoch nur, wenn bei einer vorherigen *FIND*-Operation ein zu langer Pfad komprimiert wurde, wodurch die Kosten ausgeglichen werden.
Es ist möglich, dass auf diese Weise leere Pfade entstehen. Diese werden ignoriert.
Somit ergibt sich die beschriebene obere Schranke:

$$T \in O(\underbrace{m}_{\substack{\text{letzter Schritt zur Wurzel} \\ \text{auf den Pfaden}}} + \underbrace{n}_{> UNIONs} + \underbrace{cost(C)}_{\text{Suchpfade}})$$

2.1 Zerlegung

Sei (X_b, X_t) eine Partition von X. (X_b, X_t) heißt Zerlegung von $X \iff X_t$ ist in F nach oben geschlossen, d.h. $\forall x \in X_t$: jeder Vorfahre von x ist in X_t.
Anmerkung:
Eine Zerlegung schneidet jeden Pfad p in zwei angrenzende Pfade, von denen einer leer sein kann.
Zerlegungen bleiben bei Pfadkompressionen erhalten (d.h. X_t bleibt nach oben geschlossen) .

[1] Lemma 2.1 in der Quelle[1]
[2] Die Quelle[1] fordert genau m Nicht-Wurzelpfade, was jedoch nicht immer erreichbar ist.

Lemma 2. *Hauptlemma*[3]
Sei C eine Folge von Kompressionen von Nicht-Wurzelpfaden in einem Wald F auf den Knoten X.
Sei (X_b, X_t) eine beliebige Zerlegung für F.
Dann gibt es Folgen C_b und C_t von Pfadkompressionen für $F(X_b)$ und $F(X_t)$ mit $|C_b| + |C_t| \leq |C|$ und $cost(C) \leq cost(C_b) + cost(C_t) + |X_b| + |C_t|$.

Beweis:
Sei $C = (p^{(i)})_{1 \leq i \leq M}$ die Folge von Pfaden und $(F^{(i)})_{0 \leq i \leq M}$ die daraus resultierende Folgen von Wäldern.
Sei (X_b, X_t) eine Zerlegung von $F^{(0)} := F$ und seien $F_b := F(X_b)$ und $F_t := F(X_t)$ der daraus erzeugte untere und obere Wald.

1. Die Folgen $C_b := (p_b^{(i)})_{1 \leq i \leq M}$ und $C_t := (p_t^{(i)})_{1 \leq i \leq M}$ von Pfaden ergeben sich intuitiv, indem die Knoten aus X_t bzw. X_b in C ignoriert werden.

2. $|C_b| + |C_t| \leq |C|$

   (a) $p^{(i)}$ ist ein Nicht-Wurzelpfad, dann kann höchstens einer von $p_t^{(i)}$ und $p_b^{(i)}$ ein Nicht-Wurzelpfad sein:

   Fall 1: $p_t^{(i)}$ ist leer, also ein Wurzelpfad, dann kann $p_b^{(i)}$ ein Wurzelpfad oder ein Nicht-Wurzelpfad sein.

   Fall 2: $p_t^{(i)}$ ist nicht leer, also ein Nicht-Wurzelpfad, dann muss $p_b^{(i)}$ ein Wurzelpfad sein.

   (b) $p^{(i)}$ ist ein Wurzelpfad: $p_t^{(i)}$ und $p_b^{(i)}$ sind Wurzelpfade

   Daraus folgt $|C_b| + |C_t| \leq |C|$, da nur Nicht-Wurzelpfade zur Länge der Folge beitragen.

3. $cost(C) \leq cost(C_b) + cost(C_t) + |X_b| + |C_t|$

   Fall 1: Bei der Kompression von $p^{(i)}$ erhält ein Knoten aus X_t einen neuen Vater aus X_t. Dies geschieht auch, wenn $p_t^{(i)}$ komprimiert wird, also sind die Kosten durch $cost(C_t)$ gedeckt.

   Fall 2: Genauso werden die Kosten für Knoten aus X_b, die einen neuen Vater aus X_b erhalten, durch $cost(C_b)$ gedeckt.

   Fall 3: Ein Knoten aus X_b erhält einen neuen Vater aus X_t. Dies passiert genau dann, wenn der untere Teil $p_b^{(i)}$ des Nicht-Wurzelpfades $p^{(i)}$ nicht leer ist. Alle Knoten bis auf den obersten (also die Wurzel dieses Pfades) erhalten nun zum ersten mal einen Vater aus X_t. Dies kann für jeden Knoten aus X_b höchstens einmal passieren und die Kosten werden durch $|X_b|$ gedeckt. Der oberste Knoten erhält nur dann einen neuen Vatern aus X_t, wenn $p_t^{(i)}$ nicht leer ist, also ein Nicht-Wurzelpfad ist (da $p^{(i)}$ ein nicht Wurzelpfad ist). Also werden diese Kosten durch $|C_t|$ gedeckt.
   (Da (X_b, X_t) eine Zerlegung ist, kann es nicht passieren, dass ein Knoten aus X_t einen neuen Vater aus X_b bekommt.)

   Daraus folgt die behauptete Abschätzung:

$$cost(C) \leq \underbrace{cost(C_b)}_{\substack{\text{Fall 2: Umhängen} \\ \text{innerhalb von } X_b}} + \underbrace{cost(C_t)}_{\substack{\text{Fall 1: Umhängen} \\ \text{innerhalb von } X_t}} + \underbrace{|X_b|}_{\substack{\text{Fall 3: Knoten aus} \\ X_b \text{erhält zum ersten} \\ \text{mal Vater aus } X_t}} + \underbrace{|C_t|}_{\substack{\text{Fall 3: Knoten aus } X_b \\ \text{erhält erneut Vater aus } X_t}}$$

Lemma 3. [4]*Sei F ein rangbalancierter Wald auf der Knotenmenge X mit maximalem Rang r. Sei $s \in \mathbb{N}$ und $X_{\leq s} := \{x \in X \mid rang(x) \leq s\}$ und $X_{>s} := \{x \in X \mid rang(x) > s\}$. Dann gilt:*

1. $(X_{\leq s}, X_{>s})$ ist eine Zerlegung.
2. $F(X_{\leq s})$ ist ein rangbalancierter Wald mit maximalem Rang $\leq s$.
3. $F(X_{>s})$ ist ein rangbalancierter Wald mit maximalem Rang $\leq r - s - 1$
4. $|X_{>s}| \leq |X|/2^{s+1}$

Beweis:

[3]Lemma 2.2 in der Quelle[1]
[4]Lemma 4.1 in der Quelle[1]

1. zz.: $X_{>s}$ ist nach oben geschlossen, d.h. $\forall x \in X_{>s}$: jeder Vorfahre von x ist in $X_{>s}$.

   Das ist offensichtlich erfüllt, da der Rang eines Knotens immer größer als der Rang seiner Kinder ist und somit: $x \in X_{>s} \Rightarrow Vater(x) \in X_{>s}$.

2. Von keinem Knoten aus $X_{\leq s}$ wurden Kinder entfernt, da sie einen gerigeren Rang haben, also bleibt die Eigenschaft erhalten und der maximale Rang folgt aus der Konstruktion von $X_{\leq s}$.

3. Durch das Weglassen aller Knoten mit Rang $\leq s$ hat ein Knoten mit Rang $s+1$ keine Kinder mehr. Somit ist sein Rang in $F(X_{>s})$ 0. Nun zeigt sich induktiv, dass sich der Rang jedes Knotens um $s+1$ verringert. Die Rangbalanciertheit bleibt also erhalten und es folgt der maximale Rang von $r-(s+1) = r-s-1$.

4. Jeder Teilbaum aus $X_{>s}$ besteht aus $\geq 2^{s+1}$ Knoten $\Rightarrow$ $|X| = |X_{>s}| + |X_{\leq s}| \geq |X_{>s}| \cdot 2^{s+1} \Leftrightarrow |x|/2^{s+1} \geq |X_{>s}| \square$

3 Laufzeitanalyse

Bezeichne $f(m,n,r)$ die maximalen Kosten zum Ausführen einer Sequenz von m Pfadkompressionen in einem rangbalancierten Wald aus n Knoten mit maximalem Rang r.

$\overset{\text{Lemma 11}}{\Rightarrow} T \in O(m+n+f(m,n,r))$

3.1 log *-Schranke

$f(m,n,r) \leq (r-1)\cdot n < r \cdot n =: P(r,n)$, denn jeder Knoten kann höchstens $r-1$ mal einen neuen Vater bekommen, bis er Kind einer Wurzel geworden ist.

Nun wird das Hauptlemma auf eine Zerlegung $(X_{\leq s}, X_{>s})$ mit $s = \log_2 r$ angewendet.

Durch die Wahl von s fällt der obere Wald sehr klein aus und es genügt die einfache Abschätzung durch P:

$$\begin{aligned} cost(C_t) &\leq f(|C_t|, |X_{>s}|, r-s-1) \\ &\leq P(r-s-1, |X_{>s}|) \\ &\leq (r-s-1)\cdot n/2^{s+1} \\ &\leq r \cdot n/2^{\log_2 r} \\ &= n \end{aligned}$$

Daraus ergibt sich nun mit dem Hauptlemma:

$$\begin{aligned} cost(C) &\overset{\text{Hauptlemma 2}}{\leq} cost(C_b) + \underbrace{cost(C_t)}_{\leq n} + \underbrace{|X_b|}_{\leq n} + \underbrace{|C_t|}_{\leq |C|-|C_b|} \\ &\leq cost(C_b) + 2n + |C| - |C_b| \end{aligned}$$

Dies lässt sich nun erneut mit $s' = \log_2 s = \log_2 \log_2 r$ auf die Knoten aus X_b anwenden:

$$\begin{aligned} cost(C) &\leq cost(C_b) + 2n + |C| - |C_b| \\ &\leq (cost(C_{bb}) + 2n + |C_b| - |C_{bb}|) + 2n + |C| - |C_b| \\ &= cost(C_{bb}) + 4n + |C| - |C_{bb}| \end{aligned}$$

Durch mehrmalige Anwendung erhält man Schranken der Form:

$$cost(C) \leq cost(C') + j \cdot 2n + |C| - |C'|$$

wobei C' eine Folge von Pfadkompressionen in einem Wald mit maximalem Rang $\underbrace{\log_2 \cdots \log_2}_{j \text{ mal}} r =: log^{(j)} r$ ist.

$$\log^* r := \begin{cases} 0 & , r \leq 1 \\ 1 + log^*(\log_2 r) & , sonst \end{cases}$$

Mit $j := \log^* r$ ergibt sich ein rangbalancierter Wald mit maximalem Rang 1. Dort betragen die Kosten einer Pfadkompression 0, da es nur Wurzelpfade gibt.

$$\begin{aligned} \Rightarrow cost(C) &\leq 0 + 2n\log^* r + |C| \\ &\leq 2n\log^* r + m \end{aligned}$$

$$\Rightarrow T \overset{\text{Lemma 11}}{\in} O(m + n\log^* n)$$

3.2 Eine weitere Verbesserung

Mit der nun bekannten Schranke $f(n,m,r) \leq m + 2n\log^* r$ und $s := \log\log^* r$ lässt sich für C_t folgende Schranke bestimmen:

$$
\begin{aligned}
cost(C_t) &\overset{\log^*\text{-Beweis}}{\leq} |C_t| + 2 \cdot \underbrace{|X_t|}_{\leq |X|/2^{s+1}} \cdot \log^*(s-r-1) \\
&\leq |C_t| + n/2^s \cdot \log^* r \\
&= |C_t| + n/2^{\log\log^* r} \cdot \log^* r \\
&= |C_t| + n/\log^* r \cdot \log^* r \\
&= |C_t| + n
\end{aligned}
$$

Diese führt nun mit dem Hauptlemma zu:

$$
\begin{aligned}
cost(C) &\leq cost(C_b) + cost(C_t) + |X_b| + |C_t| \\
&\leq cost(C_b) + (|C_t| + n) + n + |C_t| \\
&= cost(C_b) + 2n + 2|C| - 2|C_b|
\end{aligned}
$$

Daraus folgt nun analog zum $\log^*$-Beweis:

$$cost(C) \leq 2nL(r) + 2|C|$$

wobei L zählt, wie oft $\log\log^*$ angewendet werden kann, bis $r \leq 1$.
Das führt zu einer neuen Schranke für $f(n,m,r)$.

3.3 Die bestmögliche Schranke

Deiser Prozess lässt sich wieder und wieder wiederholen. Um dies zu formalisieren, werden zwei Funktionen definiert:
Sei $g : \mathbb{N} \to \mathbb{N}$ mit $g(n) < n \, \forall n > 0$ und
$g^\diamond(r) = \begin{cases} g(r) & , g(r) \leq 1 \\ 1 + g^\diamond(\lceil \log_2 g(r) \rceil) & , g(r) > 1 \end{cases}$

Lemma 4. *Shifting Lemma*[5]
Angenommen $\exists k \geq 0$ *und eine monoton wachsende Funktion* $g : \mathbb{N} \to \mathbb{N}$ *mit* $g(r) < r \, \forall r > 0$, *so dass*
$\forall m, n, r : \; f(m,n,r) \leq km + 2ng(r)$.
Dann gilt auch:
$\forall m, n, r : \; f(m,n,r) \leq (k+1)m + 2ng^\diamond(r)$.
Beweis per Induktion über r*:*

IA: $g(r) \leq 1 \Rightarrow g^\diamond(r) = g(r)$ ✓

IS: Sei nun r so, dass $g(r) > 1$.
Sei C eine Folge von m Pfaden, F ein rangbalancierter Wald aus n Knoten mit maximalem Rang r.
Sei $s := \lceil \log_2 g(r) \rceil$ und $(X_{\leq s}, X_{>s})$ eine Zerlegung.
$g(r) < r \Rightarrow s < r$
Der untere Wald $F(X_{\leq s})$ ist rangbalanciert und hat maximalen Rang $s < r$.

$$
\begin{aligned}
\overset{\text{IV}}{\Rightarrow} cost(C_b) &\leq (k+1) \cdot |C_b| + 2 \cdot |X_{\leq s}| \cdot g^\diamond(s) \\
&\leq (k+1) \cdot |C_b| + 2n \underbrace{g^\diamond(\lceil \log_2 r \rceil)}_{= g^\diamond(r) - 1} \\
&= (k+1) \cdot |C_b| + 2ng^\diamond(r) - 2n
\end{aligned}
$$

Der obere Wald $F(X_{>s})$ ist rangbalanciert,
hat maximalen Rang $r - s - 1$ und

[5] Lemma 4.2 in der Quelle[1]

höchsten $|X_{>s}| \leq n/2^{s+1} = n/2^{1+\lceil \log_2 g(r) \rceil} \leq \frac{n}{2g(r)}$ Knoten.

$$\begin{aligned} \Rightarrow cost(C_t) &\overset{\text{Annahme des Lemmas}}{\leq} k \cdot |C_t| + 2 \cdot \underbrace{|X_{>s}|}_{\leq n/2g(r)} \cdot g(r-s-1) \\ &\overset{g \text{ monoton wachsend}}{\leq} k \cdot |C_t| + 2 \cdot \frac{n}{2g(r)} \cdot g(r) \\ &\leq k \cdot |C_t| + n \end{aligned}$$

Das ergibt die Abschätzung:

$$\begin{aligned} cost(C) &\overset{\text{Hauptlemma2}}{\leq} cost(C_b) + cost(C_t) + |X_{\leq s}| + |C_t| \\ &\leq \big((k+1) \cdot |C_b| + 2ng^{\diamond}(r) - 2n\big) + \big(k \cdot |C_t| + n\big) + |X_{\leq s}| + |C_t| \\ &\leq (k+1) \cdot (\underbrace{|C_b| + |C_t|}_{\leq |C| \leq m}) + 2ng^{\diamond}(r) \\ &\leq (k+1)m + 2ng^{\diamond}(r) \, \square \end{aligned}$$

Mit dem Shifting Lemma ist es nun möglich, aus einer einfachen groben Schranke für $f(m,n,r)$ eine ganze Reihe gültiger Schranken abzuleiten.

Corollary 5. [6] *Sei $k \in \mathbb{N}$ und $J_k : \mathbb{N} \to \mathbb{N}$ mit*
$J_k(r) := \begin{cases} \lceil (r-1)/2 \rceil & , k = 0 \\ J^{\diamond}_{k-1}(r) & , k > 0 \end{cases}$, $J^{\diamond}_k(r) := \begin{cases} J_k(r) & , J_k(r) \leq 1 \\ 1 + J^{\diamond}_k(\lceil \log_2 J_k(r) \rceil) & , J_k(r) > 1 \end{cases}$
Dann gilt $\forall k \in \mathbb{N}: \; f(m,n,r) \leq km + 2nJ_k(r)$.

Beweis:
J_k ist monoton wachsend und $J_k(r) < r \, \forall r > 0$, was sich per Induktion zeigen lässt. Also kann das Shiftung Lemma angewendet werden:

IA $k = 0$:

In einem Wald mit maximalem Rang r kann ein Knoten höchstens $r-1$ mal einen neuen Vater bekommen, also gilt:
$f(m,n,r) \leq n \cdot (r-1) \leq 2nJ_0(r)$ ✓

IS:

$f(m,n,r) \overset{\text{I.V.}}{\leq} km + 2nJ_k(r) \overset{\text{Shiftung Lemma4}}{\Rightarrow} f(m,n,r) \leq (k+1)m + 2nJ^{\diamond}_k(r) = (k+1)m + 2nJ_{k+1}(r) \, \square$

Nun wird von den vielen Schranken, die die J_k bieten, eine besonders gute gewählt:
$\alpha_S(m,n) := min\{k \in \mathbb{N} | J_k(\lfloor \log_2 n \rfloor) \leq 1 + \frac{m}{n}\}$

Theorem 6. [7]

$$\begin{aligned} \overset{\text{Korollar 5}}{\Rightarrow} f(m,n,r) &\leq km + 2nJ_k(\lfloor \log_2 n \rfloor) \\ &\leq \alpha_S(m,n)m + 2n(1 + \frac{m}{n}) \\ &= (\alpha_S(m,n) + 2)m + 2n \end{aligned}$$

Theorem 7. [8] *Lemma 1 $\Rightarrow$ Wird auf einer n elementigen Menge eine Folge von* UNION-*Operationen und m* FIND-*Operationen unter Verwendung von* UNION *nach Rang und Pfadkompression ausgeführt, so betragen die gesamten Kosten $O(n + m\alpha_S(m,n))$.*

Die Funktion α_S verhält sich asymptotisch genauso wie die inverse Ackermannfunktion α_T:

$$\alpha_T(m,n) := min\{i \geq 1 | A(i, \lfloor \frac{m}{n} \rfloor) > \lceil \log_2 n \rceil\}$$

$A(1,j) := 2^j$	wenn $j \geq 1$
$A(i,1) := A(i-1,2)$	wenn $i \geq 2$
$A(i,j) := A(i-1, A(i,j-1))$	wenn $i,j \geq 2$

[6] Korollar 4.3 in der Quelle[1]
[7] Theorem 4.4 in der Quelle[1]
[8] Theorem 4.5 in der Quelle[1]

Literatur

[1] Raimund Seidel & Micha Sharir. Top-Down Analysis of Path Compression. http://www-tcs.cs.uni-saarland.de/Papers/unionfind.ps.gz