Datenstrukturen und effiziente Algorithmen

Markus Vieth David Klopp

15. Januar 2016

Inhaltsverzeichnis

1 Vorlesung

1.1 Priority-Queue mittels Fibonacci-Heaps

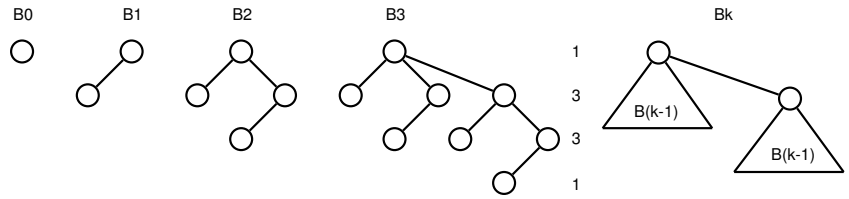

Abbildung 1.1: Binomial-Bäume

Es gilt:
Zahl der Knoten auf Level i ist $\binom{k}{i}$

$$\binom{k}{i} = \binom{k-1}{i} + \binom{k-1}{i-1}$$

Gesamtzahl aller Knoten $= 2^k$

$n = 0$:				1				
$n = 1$:			1		1			
$n = 2$:		1		2		1		
$n = 3$:	1		3		3		1	
$n = 4$:	1	4		6		4		1

1.1.1 Operationen

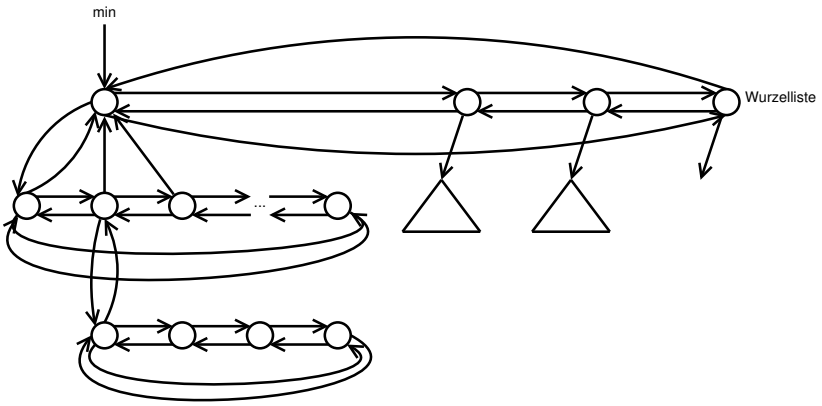

Abbildung 1.2: Aufbau

- `insert`
- `deleteMin`
- `decreaseKey`

Idee Für jeden Knoten wird gelten, dass die Zahl aller Nachfahren $\geq \Phi^k$ k =Knotengrad

`Insert`

Einzelner Knoten wird einfach in die Wurzelliste gehängt und Minimum wird aktualisiert.

`DeleteMin`

Lösche den Minimumsknoten und übernehme alle seine Kindknoten in die Wurzelliste. Konsolidiere anschließend die Wurzelliste. Nach dem Konsolidieren hat die Wurzelliste nur noch eine „kleine“ Länge und wir bestimmen das neue Minimum durch einen Durchlauf durch diese Liste.

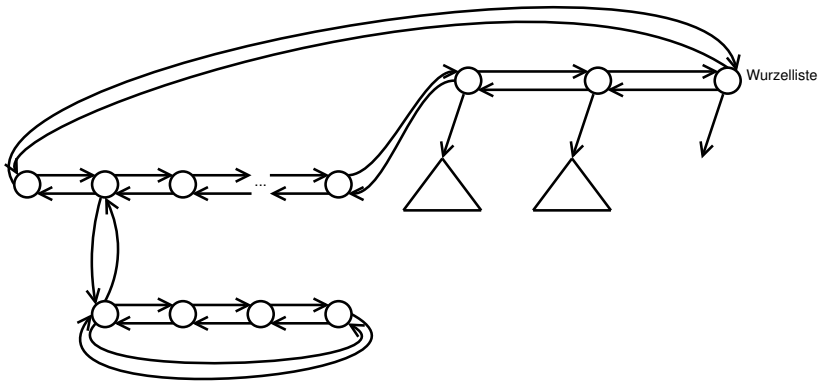

Abbildung 1.3: DeleteMin-Operation

`decreaseKey`

Wir können davon ausgehen, dass wir den betroffenen Knoten kennen. Falls der erniedrigte Schlüssel des Knotens kleiner als der Schlüssel des Vaterknotens wird, lösen wir den Knoten aus der Kindliste des Vaters und setzen ihn in die Wurzelliste. Wir markieren den Vater, dass er einen Kindknoten verloren hat. Sollte der Vater schon eine Markierung tragen, so wird auch der Vater Knoten abgelöst und in die Wurzelliste gesetzt. Dieser Prozess kann sich kaskadenartig bis zur jeweiligen Wurzel fortsetzen. Bei Aufnahme eines abgelösten Knotens in die Wurzelliste, muss auch das Minimum aktualisiert werden.

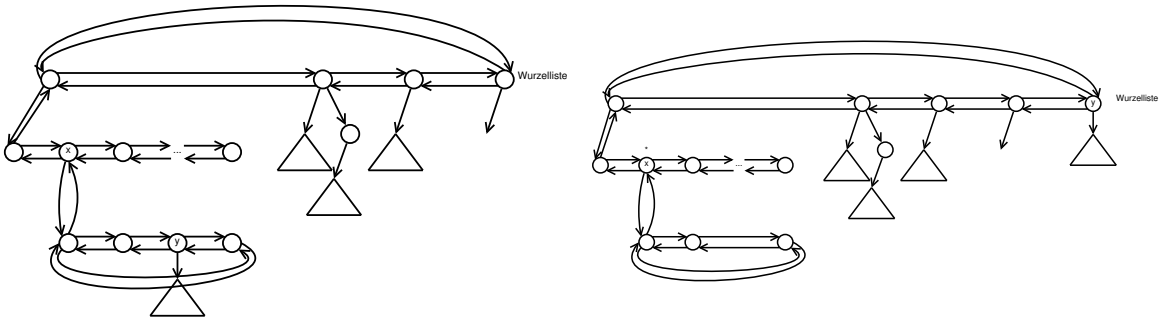

Abbildung 1.4: decreaseKey-Operation

Konsolidierung der Wurzelliste

Nach dem „Lazy Evaluation“-Prinzip wird dieser Vorgang nur nach einem `deleteMin` angestoßen, um die Wurzelliste zu verkürzen. Wir nutzen ein einfaches Feld hinreichender Größe, um temporär Knoten, entsprechend ihren Grades, zu verwalten. Wir durchlaufen die Wurzelliste. Wenn wir einen Knoten vom Grad k antreffen, schreiben wir ihn in das Feld an Position k bzw. verschmelzen ihn mit dem Knoten von Grad k, den wir dort antreffen. Dadurch entsteht gegebenenfalls ein neuer Knoten vom Grad $k+1$ der an Position $k+1$ im Feld zu setzen ist. Es kann also zu weiteren Fusionsoperationen kommen. Analogie zu der Übertragungsfortpflanzung beim Binärzähler.

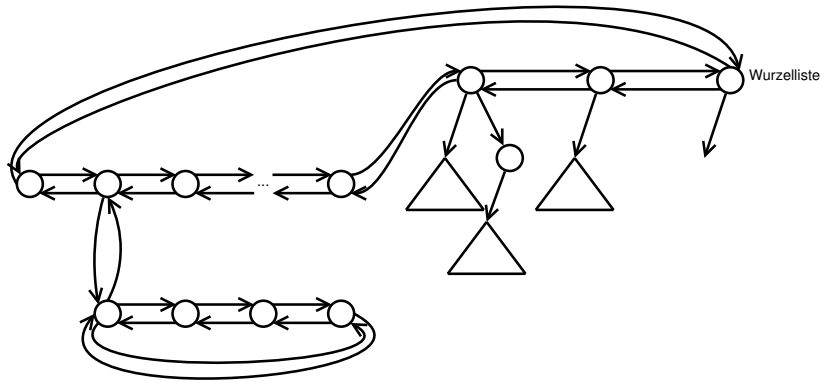

Abbildung 1.5: Konsolidierungs-Operation

2 Vorlesung

2.0.1 Lemma

Für jeden Knoten x in einem Fibonacci-Heap gilt, dass die Zahl aller Knoten im Unterbaum von x mindestens Φ^k beträgt, wobei $k = \text{grad}(x)$.

2.0.2 Beweis

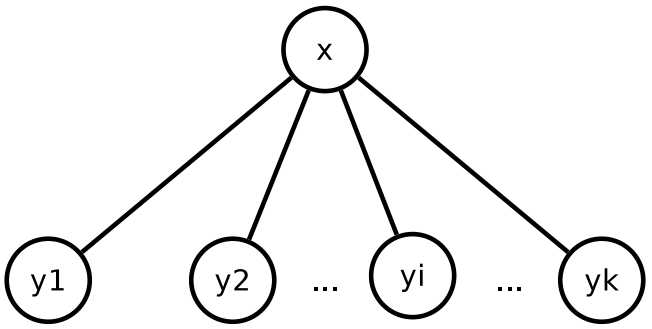

Abbildung 2.1: Schaubild zum Beweis

Seien $y_1, y_2, \ldots, y_k$ die aktuellen Kindknoten von x, nummeriert in der Reihenfolge, wie sie (letztmalig) Kindknoten von x geworden sind. Zum Zeitpunk, zu dem y_i Kindknoten von x geworden ist, existieren bereits die $i-1$ Kindknoten $y_1, y_2, \ldots, y_{i-1}$. $\Rightarrow \text{grad}(x) \geq i-1$
y_i kann nur Kind von x werden, wenn x und y_i gleichen Grad haben. $\Rightarrow \text{grad}(y_i) \geq i-1$
Da y_i im Folgenden höchstens einen Kindknoten verlieren kann, gilt:

$$\text{grad}(y_i) \geq i-2$$

Sei S_k die Mindestanzahl von knoten in einem Unterbaum eines Knoten x vom Grad k

$$S_k \geq 1 + 1 + \sum_{i=2}^{k} S_{i-2}$$

Wir zeigen

$$S_k \overset{(2)}{\geq} f_{k+2}{}^{\text{I}} \overset{(1)}{\geq} \Phi^k$$

(1)

$$f_{k+2} \geq \Phi^k$$

f_0	f_1	f_2	f_3	f_4	f_5	$\ldots$
0	1	1	2	3	5	

$$k = 0 \ : \ f_2 = 1 \geq \Phi^0 = 1$$

$$k = 1 \ : \ f_3 = 2 \geq \Phi^1 = 1,6181\ldots$$

$$f_{k+2} = f_{k+1} + f_k \geq \Phi^{k-1} + \Phi k - 2 = {}^{\text{II}}\Phi k - 2(\Phi + 1) = \Phi k - 2 \cdots \Phi^2 = \Phi^k$$

(2)

$$S_k \geq f_{k+2}?$$

$$S_k \geq 2 + \sum_{i=2}^{k} S_{i-2}$$

[I] $k-2$-te Fibonacci Zahl
[II] $\Phi^2 = \Phi + 1$

$$k = 0 \ : \ S_0 \geq f_2 = 1 \ \checkmark$$
$$k = 1 \ : \ S_1 \geq f_3 = 2 \ \checkmark$$

$$S_k \geq 2 + \sum_{i=2}^{k} f_i, \text{ da wegen Induktions-Annahme } S_{i-2} \geq f_{(i-2+2)} = f_i \text{ für } i < k$$

Zu zeigen

$$2 + \sum_{i=2}^{k} f_i \geq f_{k+2}$$

Es gilt:

$$1 + \sum_{i=1}^{k} f_i = f_{k+2}$$
$$k = 0 \ : \ 1 = f_2 \ \checkmark$$
$$k = 1 \ : \ 1 + f_1 = f_3 = 2 \ \checkmark$$
$$1 + \sum_{i=1}^{k+1} = (1 + \sum_{i=1}^{k} f_i) + f_{k+1} = f_{k+2} + f_{k+1} = f_{k+3}$$

q.e.d.

Aus dem Lemma folgt, dass in einem Fibonacci-Heap für n[III] Elemente zu keinem Zeitpunkt einen Knoten vom Grad $\log_\phi n = k$ auftauchen kann. insbesondere ist die Wurzelliste nach einer Konsolidierung auch nur $\log_\Phi n$ lang, weil dort nur Knoten unterschiedlichen Grades auftauchen.

[III] $\Phi^k \leq n$

Abbildungsverzeichnis