Info2 Uebung vom 29.4.02

Gruppe F, Thomas Briner, http://n.ethz.ch/student/brinerth/info2

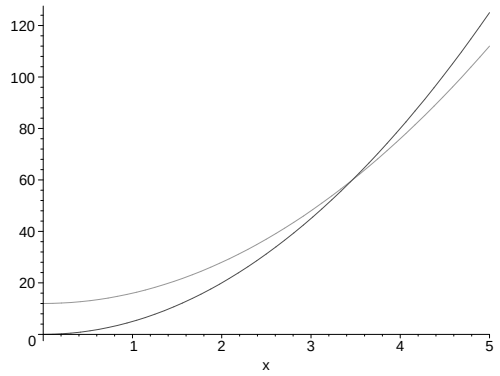
29. April 2002

1 Nachbesprechung

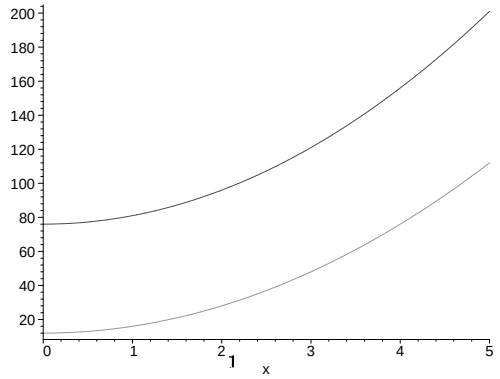
1.1 Aufgabe 2.1

Definition aus Vorlesung: $O(f(n)) = \{g(n) : \exists c_1, c_2 : \forall n : g(n) \leq c_1 f(n) + c_2\}$
Definition aus Uebung: $O(f(n)) = \{g(n) : \exists c, n_0 : \forall n \geq n_0 : g(n) \leq cf(n)\}$

```
> plot([5*x^2,4*x^2+12],x=0..5);
```




```
> plot([5*x^2+76,4*x^2+12],x=0..5);
```




```
>
```

1.2 Aufgabe 2.2b)

Umformung von $2^{log_4(n)}$:

2 Heaps

2.1 Welches der Arrays ist ein impliziter Heap?

a)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
3	9	4	27	28	6	11	33	91	46	52	10	7	88	24

b)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
4	5	12	7	6	19	21	8	14	9	10	17	33	26	40

c)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
6	19	23	36	41	42	48	56	59	83	84	87	96	101	105

d)

0	1	2	3	4	5	6	7	8	9	10	11	12
1	3	11	4	7	18	5	29	14	31	6	25	12

2.2 Die Heapeigenschaft

Die Eigenschaft eines (Min-)Heaps ist, dass jedes Element kleiner ist als seine beiden Kinder, sofern vorhanden. Das Minimum steht infolgedessen in der Wurzel.

Beim impliziten Heap wird die Baumstruktur in ein Array abgebildet. Je nachdem, ob die Numerierung bei 0 oder 1 beginnt, lässt sich diese Bedingung für einen binären Heap entsprechend übertragen in

- a[2*i] und a[2*i+1] sind grösser als a[i] (für Beginn mit 1)
- a[2*(i+1)]-1] und a[2*(i+1)] sind grösser als a[i] (für Beginn mit 0)

2.3 Operationen auf einem Heap

2.3.1 Versickern lassen

Ausgangssituation: oberstes Element im Heap ist nicht an der korrekten Stelle, der Rest der Elemente erfüllt die Heapeigenschaft

- Solange das Element noch Kinder hat:
- Welches ist das kleinere der Kinder?
- Ist das Element grösser als das kleinere der Kinder?
- Wenn ja: vertauschen und weiter im darunterliegenden Teilheap versickern
- Wenn nein: so belassen, Versickern beendet, Heapeigenschaft für ganzen Heap erfüllt.

Beispiel:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
41	9	4	27	28	6	11	33	91	46	52	10	7	88	24

Resultat:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

2.3.2 Minimum aus Heap holen:

- Wurzelelement rausnehmen
- Element mit grösstem Index in die Wurzel setzen
- Heapgrösse um eins verkleinern
- Wurzelelement im ganzen Heap versickern lassen

Beispiel:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
3	9	4	27	28	6	11	33	91	46	52	10	7	88	24

Resultat:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

2.3.3 Heap aus beliebigem Array herstellen:

Für alle Elemente, die Kinder haben, beginnend mit grösstem Index:

- Element in Teilheap, von dem es selber die Wurzel ist versickern lassen

Beispiel:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
37	6	97	5	7	32	14	16	84	61	36	31	11	19	2

Resultat:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

2.3.4 Element in Heap einfügen

- Neues Element in erste leere Feld am Schluss einfüllen
- Grösse des Heaps um eins erhöhen
- Mit Elter vergleichen; falls Elter grösser ist $\Rightarrow$ tauschen
- Solange nach oben weitergehen, bis Vertauschung nicht nötig

Beispiel: 5 einfügen

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
3	9	6	27	28	7	11	33	91	46	52	10	8	88	

Resultat:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

2.3.5 Anzahl Elemente ausgeben

- Um zu wissen, ob ein Element noch Kinder hat, muss immer die aktuelle Heapgrösse gespeichert sein ⇒ Aktuelle Heapgrösse ausgeben.

3 Projekt

Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Institut für Theoretische Informatik
Peter Widmayer
Matthias Müller
Sandra Roost

Informatik 2 **Projekt** **SS 02**

Wie bereits angekündigt, sollt Ihr ein kleines Projekt erstellen, in dem Ihr Euch mit einem selbstgewählten Thema aus dem Bereich "Algorithmen und Datenstrukturen" beschäftigt. Der Themenbereich ist bewusst sehr weit gefasst, so dass Ihr ein Thema auswählen könnt, das Euch interessiert. Ihr solltet (wenn möglich) in 3er bis 4er-Teams zusammen arbeiten.

Ziele / Motivation:

- In der Info-2-Vorlesung hört Ihr eher kurze Einführungen in viele verschiedene Themen. Einige davon vertieft Ihr in den Übungen, aber es sind immer noch viele Themen. In diesem Projekt sollt Ihr Euch für ein Thema, das Euch interessiert, zu 'Experten' machen.
- Über längere Zeit in Teams an einem Thema arbeiten. So lernt Ihr wahrscheinlich mehr als in einer Übung, wo viele aus Zeitgründen einfach weiterfahren, wenn sie etwas nicht wissen.

Mögliche Formen:
Es wäre schön, wenn Euer Thema so präsentiert würde, dass auch die anderen davon profitieren können, wenn es Sie interessiert. Deshalb empfehlen wir Euch, eine Web-Site zum gewählten Thema zu machen. Falls Ihr Euch damit nicht auskennt, werden wir Euch gerne weiterhelfen. Ausserdem wäre es gut, wenn Ihr interessante Algorithmen ausprogrammiert. Für weitere Ideen sind wir natürlich auch offen.

Beispiele:

- Sortieralgorithmen
- AVL-Baum korrekt ausprogrammieren. Oder eine andere Art Baum
- Polygon-Guard (Wächter in Museum) oder Staubsauger-freundliches Haus
- Star-Suche
- Verschiedene Arten der Leader-Election
- Graphenalgorithmen
- Selbstanordnende Listen

Rahmen:
Dieses Projekt ist freiwillig. Es wird einen Wettbewerb geben, in dem wir die besten Projekte prämieren (natürlich bewerten wir den Inhalt, nicht das Design). Ausserdem wird es Euch als Übung angerechnet, d.h. Ihr müsst dann nur 8 normale Übungen abgeben statt 9.
Falls Ihr Euch für solch ein Projekt entscheidet, meldet Euch bei einem Eurer Assistenten. Besprecht Euer Thema bis zum Montag, 13. Mai mit ihm, entweder vor/nach der Übungsstunde, per e-Mail oder an einem vereinbarten Termin. Er wird Euch sagen können, ob Ihr Euch zu viel oder zu wenig vornehmt. Wendet Euch auch mit Fragen an ihn.

Abgabe: Bis Freitag, den 21. Juni 2002 an Euren Assistenten