Theta for Dummies oder: Asymptotisches Wachstum von Funktionen

phpwutz

August 2, 2013

1 Motivation

Nachdem immer wieder Fragen bezüglich der O-, Θ- und Ω-Notation gestellt werden, versuche ich das Ganze hier mal verständlich zu erklären (Und vielleicht freut sich ja nächstes Semester jemand darüber).
Die Symbole werden dafür verwendet anzugeben, wie aufwändig der Algorithmus in Abhängigkeit von n (Anzahl der zu bearbeitenden Elemente) ist, also welche Laufzeit er in etwa benötigt.

2 Was bedeuten diese Symbole überhaupt?

Die Symbole O, Θ und Ω stehen grundlegend für eine bestimmte Menge an Funktionen. Welche Funktionen das sind, hängt davon ab, durch welche Funktion parametrisiert wird.

Beispiel $\Theta(n^2)$ sind beispielsweise alle Funktionen, die ungefähr gleich schnell wachsen wie n^2. Was das *ungefähr* bedeutet, sehen wir uns gleich an.

Man schreibt $f(n) = \Theta(g(n))$ um auszudrücken: Die Funktion $f(n)$ liegt in der Menge von $\Theta(g(n))$

- $f(n) = O(g(n))$ bedeutet, dass $g(n)$ eine **obere Schranke** von $f(n)$ ist. $f(n)$ wächst also **maximal** so schnell wie $g(n)$.
- $f(n) = \Theta(g(n))$: $f(n)$ wächst weder wesentlich langsamer, noch wesentlich schneller als $g(n)$, $g(n)$ ist also eine **obere** und gleichzeitig eine **untere** Schranke.
- $f(n) = \Omega(g(n))$: Wenn wir diese Aussage vor uns haben, wächst $f(n)$ **mindestens** so schnell wie $g(n)$, $g(n)$ ist also um eine **untere Schranke** von $f(n)$.

3 Formale Definition (mit Übersetzung auf Deutsch)

Nun also zur formalen Definition von *wächst ungefähr gleich wie...*

Θ (Theta): $\Theta(g(n)) = \{f(n)|(\exists c_1, c_2, n_0 > 0), (\forall n \geq n_0) : 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$

$\Theta(g(n))$ ist also die Menge aller Funktionen, für die wir 2 Konstanten und ein n_0 (also ein n, ab dem die Ungleichung erfüllt ist) finden können, so dass gilt:
$0 \leq c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$

O (obere Schranke) $O(g(n)) = \{f(n)|(\exists c_1, c_2, n_0 > 0), (\forall n \geq n_0) : 0 \leq f(n) \leq c_1 g(n)\}$

Im Prinzip dasselbe Spiel wie beim Theta, mit diesem Unterschied: die Funktion $f(n)$ darf **maximal** gleich schnell wachsen wie $g(n)$ um in $O(g(n))$ zu sein. Wir brauchen also n_0 und c_1, so dass gilt:
$0 \leq f(n) \leq c_1 \cdot g(n)$

Ω (untere Schranke) $\Omega(g(n)) = \{f(n)|(\exists c_1, c_2, n_0 > 0), (\forall n \geq n_0) : 0 \leq c_1 g(n) \leq f(n)\}$

Wieder (fast) dasselbe, nur dass $f(n)$ in diesem Fall **minimal** gleich schnell wachsen muss wie $g(n)$ um in $\Omega(g(n))$ zu sein. Wir brauch also n_0 und c_1, so dass gilt:
$0 \leq c_1 \cdot g(n) \leq f(n)$

3.1 Beweisen

Wenn wir nun (wie in vielen Tests nötig) beweisen wollen, dass eine gegebene Funktion in einer dieser 3 Mengen liegt, brauchen wir lediglich n_0, c_1 (für Theta zusätzlich: c_2) finden so dass die entsprechende Ungleichung erfüllt ist - et voila!
Dazu ist es oft hilfreich, Rechenoperationen auf die gesamte Ungleichung anzuwenden (dividieren durch n hat mir sehr oft geholfen), damit man leichter sehen kann, mit welchen Werten die Ungleichung erfüllt wird.

3.2 Abschätzen

Manchmal ist es nötig, schnell abzuschätzen in welcher Θ-Laufzeit eine Funktion liegt. In diesem Fall ist es schlau, alles möglichst gut zusammenzufassen, zB. $n \cdot n$ wird zu n^2 zusammengefasst, auf Summenzeichen die Gauß'sche Summenformel angewandt etc. Wonach wir suchen sind Ausdrücke, die in etwa so aussehen:

$\Theta(1) < \Theta(log(n)) < \Theta(\sqrt{n}) < \Theta(n) < \Theta(n \cdot log(n)) < \Theta(n^2) < \Theta(2^n) < \Theta(n!) < \Theta(n^n)$

Aus dieser Liste nimmt man dann das *am stärksten wachsende* was man in seiner Funktion finden kann und hat seine ungefähre Theta-Laufzeit.

4 Beispiele

4.1 Beispiel 1 - Beweisen

Wir werden hier beweisen dass die Funktion $T(n)$ in $\Theta(n^2)$ liegt, also gilt: $T(n) = \Theta(n^2)$
$T(n) = 10n + 10 \cdot \sum_{k=1}^{n} k$

Wir erinnern uns an die formale Definition der Funktionen, die in $\Theta(g(n))$ liegen und suchen Konstanten c_1, c_2 und n_0 so dass gilt:

$$0 \leq c_1 \cdot n^2 \leq 10n + 10 \cdot \sum_{k=1}^{n} k \leq c_2 \cdot n^2$$

Umformen des Summenzeichens mittels der Gauß'schen Summenformel ergibt:
$0 \leq c_1 \cdot n^2 \leq 10n + 10 \cdot \frac{n \cdot (n+1)}{2} \leq c_2 \cdot n^2$

Nochmaliges Umformen des Mittelteils:
$0 \leq c_1 \cdot n^2 \leq 10n + 5n^2 + 5n \leq c_2 \cdot n^2$

Jetzt dividieren wir die ganze Ungleichung durch n^2 und erhalten:
$0 \leq c_1 \leq \frac{15}{n} + 5 \leq c_2$

Jetzt sollte es leicht sein, Konstanten sowie ein passendes n zu finden, damit die Ungleichung erfüllt ist: (für $c_1 = 4, c_2 = 6, n_0 = 5$)

$$0 \leq 4 \leq \frac{5}{5} + 5 \leq 6$$

Bemerkung: Die Konstanten müssen nicht besonders *knapp* gewählt werden, es reicht die Tatsache aus, dass ihr zeigen könnt, dass diese Konstanten mit zugehörigem n_0 überhaupt *existieren.* Wichtig dabei ist, dass die Konstanten, die ihr verwendet, die Ungleichung **für alle** $n > n_0$ erfüllen. (wenn ihr also zB $n_0 = 100$ verwendet, müssen die Konstanten für alle $n \geq 100$ gelten).

4.2 Beispiel 2 - Theta abschätzen

Dafür benutzen wir gleich den Teil aus dem ersten Beispiel: $5n + 5n^2$
Wir suchen uns den Summanden mit dem stärksten Wachstum (Siehe Aufzählung unter *Abschätzen*) aus der Funktion und nehmen diesen als Theta-Laufzeit. Faktoren, die nicht von n abhängen, können dabei für die Θ-Notation getrost weggelassen werden. Es ergibt sich also $\Theta(n^2)$