Prof. Dr. A.K. Hartmann WiSe 2022/23, 5.04.4521
Blatt 3, 20.2.2023

Übungen zur Computerorientierten Physik

3 Maximales Teilfeld

Gegeben sei ein Array (deutsch: "Feld") $a[0] \ldots a[n-1]$ von n (ganzen) Zahlen.
Die *Teilfeld-Summe* $S(k,l)$ $(0 \le k \le l < n)$ sei definiert als

$$S(k,l) = \sum_{i=k}^{l} \mathtt{a}[i] \tag{1}$$

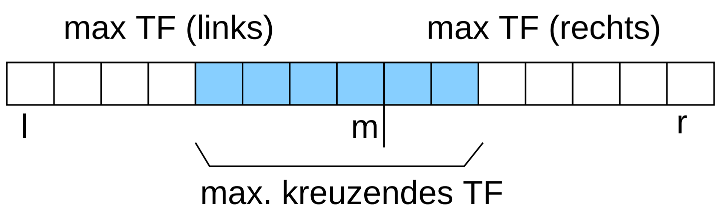
Gesucht sei das *maximale Teilfeld*, das die maximale Teilfeldsumme

$$\max_{0 \le k \le l < n} S(k,l) \tag{2}$$

ergibt.
Ein schneller Algorithmus verfolgt einen *divide-and-conquer* Ansatz. Dazu wird, für $l < r$ (rekursiv) ein (Teil-) Array $a[l] \ldots a[r]$ in zwei Teil Arrays $a[l] \ldots a[m]$ und $a[m+1] \ldots a[r]$ zerlegt (mit der "Mitte" $m = (l+r)/2$). Für die beiden Teil-Arrays wird jeweils durch rekursiven Aufruf das maximale Teilfeld ausgerechnet. Weiterhin wird dann (nicht rekursiv, durch zwei unabhängige Schleifen) das maximale "überkreuzende Teilfeld"

$$\max_{l \le j \le m} S(j,m) + \max_{m+1 \le j \le r} S(m+1,j)$$

berechnet. Das maximale Teilfeld ist dann das Maximum dieser drei Teilfelder.
Der Algorithmus wird initial mit $l = 0$, $r = n-1$ aufgerufen. Überlegen Sie sich zur Implementierung: Was macht der Algorithmus bei $l = r$?



Dieser Algorithmus hat eine asymptotische Laufzeit von $O(n \log n)$.

- Wie sieht ein einfacher "brute-force" Algorithmus aus? Welche asymptotische Laufzeit hat er als Funktion von n?
- Laden Sie sich das unvollständige Programm `max_subarry_fragment.c` vom StudIP und speichern es (z.B.) unter `max_subarry.c` ab.

- Schauen Sie sich an, was das Programm bisher macht, also insbesondere Datenstrukturen und Hauptprogramm.
- Komplettieren Sie die (nicht rekursive) Funktion `max_crossing_subarray()` die das maximale überkreuzende Teilfeld ausrechnet.

  Testen Sie die Funktion mit einfachen Arrays.
- Komplettieren Sie die Funktion `max_subarray()` die das maximale Teilfeld ausrechnet, so dass der rekursive Ansatz realisiert wird.

  Testen Sie die Funktion mit einfachen Arrays.
- Lassen Sie das Programm für $n = 10, 20, 50, 100, 200, \ldots, 500000$ laufen und schreiben Sie die mittlere Summe im größten Teilarray und dessen mittlere Länge als Funktion von n in eine Datei, z.B. `max_subarray.dat`.
- Stellen Sie das Ergebnis dar, z.B. mit `gnuplot`. Welches asymptotische Wachstum beobachten Sie (Hinweis: In `gnuplot` können Sie die Achsen mit `set logscale x` bzw. `set logscale y` logarithmisch skalieren)? Fitten Sie eine geeignete Funktion (z.B. mit `fit` in `gnuplot`). Hinweis mit `fit [from:to] f(x) "datei.dat" ...` können Sie den Bereich wählen in dem die Funktion `f(x)` gefittet werden soll.
- Wie verhält sich das ganze für die Länge des maximalen Teilarrays?