Prof. Andreas Goerdt
Andreas Berger

TU Chemnitz
Wintersemester 2002/03

Parallele Algorithmen
2. Übung

1. Aufgabe

Wir betrachten das Problem PARTITION, wie es zum Beispiel als Teilproblem innerhalb von Quick-Sort auftritt. Dabei ist ein Array $A = (a_1, ..., a_n)$ aus n Zahlen sowie eine weitere Zahl p gegeben. Die Elemente von A sollen so umgeordnet werden, dass alle Zahlen kleiner als p geschlossen am Anfang stehen, während die größeren (oder gleich großen) am Ende platziert sind. Das heißt, es muss ein Array $A' = (a'_1, ..., a'_n)$ aus den selben Elementen wie A berechnet werden, so dass es ein $1 \leq i \leq n$ gibt mit

$$\begin{aligned} j < i &\Rightarrow a'_j < p \\ j \geq i &\Rightarrow a'_j \geq p. \end{aligned}$$

Geben Sie einen optimalen Algorithmus für eine EREW-PRAM an, der dieses Problem in Zeit $\mathcal{O}(\log n)$ bewältigt!

2. Aufgabe

Gegeben sei ein Graph mit Knotenmenge $V = \{v_1, v_2, ..., v_n\}$, so dass v_1, $v_2 ... v_n$ einen Kreis bilden. Zu jedem Knoten gibt es außerdem maximal eine weitere inzidente Kante.

Entwickeln Sie einen parallelen Algorithmus, der in Zeit $\mathcal{O}(\log n)$ entscheidet, ob der gegebene Graph ohne Kreuzung zweier Kanten so gezeichnet werden kann, dass sich alle Extra-Kanten innerhalb des Kreises befinden! Ihr Algorithmus soll auf einer EREW-PRAM laufen und nicht mehr als $\mathcal{O}(n)$ Arbeit verrichten.

3. Aufgabe

Sei $A = (a_1, ..., a_n)$ ein boolesches Array der Größe n. Gesucht wird der kleinste Index i, so dass $a_i = 1$. Geben Sie einen Algorithmus für eine common-CRCW-PRAM an, der diesen Index in konstanter Zeit $\mathcal{O}(1)$ bestimmt!

4. Aufgabe

Zeigen Sie, wie man das Maximum von n Elementen auf einer CRCW-PRAM mit Arbeit $\mathcal{O}(n^{1+c})$ in konstanter paralleler Zeit bestimmen kann!

(Hinweis: die Anzahl paralleler Schritte darf natürlich von c abhängen.)

Wie schnell können Sie das Maximum mit einem (arbeits-)optimalen Algorithmus berechnen?

5. Aufgabe

Wir beschäftigen uns im folgenden mit dem Problem, die sogenannte konvexe Hülle für eine gegebene Menge von Punkten einer Ebene zu bestimmen.

Gegeben ist also eine Menge $S = \{p_1, ..., p_n\}$ von Punkten. Dabei wird jeder Punkt p_i durch seine kartesischen Koordinaten (x_i, y_i) repräsentiert.

Gesucht ist eine geordnete Liste $KH(S)$ von Punkten aus S, die die Eckpunkte des kleinsten konvexen Polygons, welches alle Punkte von S enthält, darstellt.

(a) Zeigen Sie, dass das Problem des Sortierens von n Zahlen in (sequentieller) Zeit $\mathcal{O}(n)$ auf das Bestimmen der konvexen Hülle von n Punkten reduziert werden kann! Welche Aussage über die Arbeit jedes korrekten parallelen Algorithmus zur Bestimmung von KH lässt sich daraus unter Verwendung der unteren Zeitschranke für das Sortieren ableiten?

(b) Gegeben seien Punktmenegen S_1 und S_2, so dass für je zwei Punkte $(x_1, y_1) \in S_1$ und $(x_2, y_2) \in S_2$ gilt: $x_1 \leq x_2$. Weiterhin seien die zugehörigen konvexen Hüllen $KH(S_1)$ und $KH(S_2)$ bekannt. Überlegen Sie, wie damit die konvexe Hülle $KH(S_1 \cup S_2)$ bestimmt werden kann! Geben Sie den Zeitbedarf einer sequentiellen Variante an sowie Zeit und Arbeit für eine parallele Realisierung!

(c) Verwenden sie das Verfahren aus (b), um einen sequentiellen "divide and conquer"-Algorithmus zur Bestimmung der konvexen Hülle anzugeben! Bestimmen Sie die Laufzeit des Verfahrens!

(d) Geben Sie nun eine parallele Variante des Algorithmus an! Welches PRAM-Modell wird dabei verwendet? Geben Sie Zeit- und Arbeitsbedarf an!