PHILIPPS-UNIVERSITÄT MARBURG
Fachbereich Mathematik und Informatik
Prof. Dr. R. Loogen
J.Berthold

D-35032 Marburg
Hans Meerwein Straße
Lahnberge
5.Juli 2007

11. Übung zur Vorlesung "Parallele Algorithmen", Sommer 07

Abgabe: 12.Juli 2007 vor der Vorlesung

Dies ist das letzte Übungsblatt mit schriftlichen Aufgaben. Ein weiteres Übungsblatt erscheint am 12.7 und wird mündliche Aufgaben enthalten.

Aufgaben

11.1 **8-Felder-Puzzle** 4 Punkte

1	5	2
4	8	3
7		6

Das 8-Felder-Puzzle besteht aus 8 nummerierten Teilen, die auf einem 3×3 verschiebbar befestigt sind. Ein Spielzug besteht daraus, eines der horizontal oder vertikal benachbarten Teile in das verbleibende freie Feld zu bewegen.

Das Ziel des Spiels ist, die Puzzleteile in zeilenweise sortierte Reihenfolge zu bringen. Mit Hilfe eines Suchbaums der möglichen Züge lässt sich eine minimale Zugfolge zur Lösung ermitteln.

(a) Begründen Sie, dass ein minimierender Standard-Backtracking-Algorithmus nach dem Schema der Vorlesung die Lösung wahrscheinlich *nicht* finden wird. Welche Änderungen muss man am Algorithmus vornehmen?

(b) Ein Branch& Bound-Algorithmus zur Bestimmung der minimalen Lösung kann als Abschätzung der noch nötigen Züge die sog. "Manhattan-Distanz" [1] aller Teile zu ihrer Endposition verwenden. Skizzieren Sie den vom Branch& Bound-Algorithmus durchsuchten Suchbaum für das angegebene Beispiel. Geben Sie zu jedem Knoten die Abschätzung der nötigen Spielzüge an.

11.2 **Paralleles Branch& Bound** 3 Punkte

Nennen Sie Vor- und Nachteile der folgenden Idee zur Parallelisierung eines Branch& Bound-Algorithmus im Vergleich mit der in der Vorlesung vorgestellten Parallelisierung.

> Zunächst werden alle Kindknoten des Wurzelknotens im Entscheidungsbaums bestimmt und gleichmäßig auf die verfügbaren Prozesse aufgeteilt (gibt es mehr Prozesse als Kindknoten, so werden statt dessen deren Kindknoten genommen usw.). Nun arbeiten alle Prozesse mit einem sequenziellen Branch& Bound-Algorithmus ihre Teilbäume ab. Gefundene Lösungen werden per Broadcast allen Prozessen mitgeteilt. Wenn ein Prozess eine Lösung empfängt, entfernt er alle Aufgaben aus seiner Queue, welche nicht zu einem besseren Ergebnis führen können.

---

[1] In einem n-dimensionalen Koordinatensystem wird als Manhattan-Distanz zweier Punkte x und y die Summe der Beträge aller Koordinatendifferenzen bezeichnet: $d_m(x,y) = \sum_{i=1}^{n} |x_i - y_i|$.

Bitte wenden!

11.3 Kürzeste Wege

5 Punkte

Beim „Einzelne-Quelle kürzeste-Wege"-Problem muss der kürzeste Weg von einem festgelegten Quellknoten s zu allen anderen Knoten eines gewichteten, gerichteten Graphen bestimmt werden.

Der folgende sequentielle Algorithmus wurde hierzu 1959 von E. F. Moore entwickelt:

Parameter:	n	Anzahl der Knoten
Globale Variablen:	s	Quellknoten
	distance	Array: Abstände von s zu anderen Knoten
	weight	Kostenmatrix

```
begin
   for i := 1 to n do
       INITIALISE(i)
   od
   insert s into the queue
   while the queue is not empty do
       SEARCH()
   od
end
```

`INITIALISE` initialisiert das Feld *distance* für den Quellknoten mit 0 und für jeden anderen Knoten mit ∞.

`SEARCH()`:

Local:	u	untersuchter Knoten
	v	von u über einzelne Kante erreichbarer Knoten
	new.distance	Abstand zu v bei Weg über u

```
begin
   dequeue vertex u
   for every edge (u,v) in the graph do
       new.distance := distance(u) + weight(u,v)
       if new.distance < distance(v) then
           distance(v) := new.distance
           if v is not in the queue then
               enqueue v
           fi
       fi
   od
end
```

(a) Erläutern Sie die Arbeitsweise des Algorithmus.

(b) Diskutieren Sie die Parallelisierbarkeit dieses Algorithmus.

(c) Beschreiben Sie einen parallelen Algorithmus, der die Prozedur `SEARCH` parallel ausführt. Den Prozessen stehe ein gemeinsamer Speicher zur Verfügung.

Beachten Sie, dass die Prozesse erst terminieren, wenn keine Arbeit mehr zu leisten ist, und dass beim Schreiben in den gemeinsamen Speicher, falls erforderlich, eine Synchronisation erfolgt.