Karlsruher Institut für Technologie
Institut für Theoretische Informatik

Prof. Dr. Peter Sanders
Moritz Kobitzsch, Dennis Schieferdecker

5. Übungsblatt zu Algorithmen II im WS 2011/2012

http://algo2.iti.kit.edu/AlgorithmenII.php
{kobitzsch,sanders,schieferdecker}@kit.edu

Weihnachtsblatt – mit extra vielen Aufgaben :)

Aufgabe 1 (*Pflichtaufgabe (*)*)

a) Machen Sie Ihren Übungsleitern ein schönes Weihnachtsgeschenk.

Aufgabe 2 (*Analyse: Kürzeste Wege (Wiederholung)*)

a) Betrachten Sie eine Suche mit bidirektionalem Dijkstra. Geben Sie eine Familie von Graphen an, bei der ein ausgezeichneter Knoten u eine Anzahl an `decreaseKey` Operationen erfährt, die linear in der Länge des kürzesten Weges für jede mögliche Anfrage ist.

b) Bei manchen Algorithmen kann es sinnvoll sein, unterschiedliche Potentialfunktionen zu kombinieren. Zeigen Sie in diesem Zusammenhang: Sind π_1 und π_2 gültige Potentialfunktionen, so ist auch $\pi = \frac{\pi_1+\pi_2}{2}$ eine gültige Potentialfunktion.

c) Bei der Durchführung von *Dijkstras Algorithmus* können kürzeste Wege gespeichert werden, indem Vorgängerknoten gespeichert werden. Diese werden immer dann geändert, wenn eine bessere vorläufige Distanz gefunden wird. Dieses kann auch auf einem Graphen durchgeführt werden, der negative Kantengewichte enthält. Das Stopkriterium von Dijkstras Algorithmus muss dafür durch ein schwächeres Kriterium ersetzt werden: Es wird gestoppt, sobald keine Verbesserung mehr gefunden werden kann. Zeigen Sie, wenn auf diese Art ein Kreis entsteht, so ist dieser negativ.

d) (*) Dijkstras Algorithmus ist ein Spezialfall des allgemeinen *Labeling Algorithmus*. Der allgemeine Labeling Algorithmus wählt eine beliebige Kante aus, die das Label des Zielknotens verbessert. Geben Sie einen Graphen sowie eine Reihenfolge der Kanten an, so dass bei positiven Kantengewichten eine exponentielle Zahl von Schritten ausgeführt wird.

Hinweis: Achtung, schwierige Knobelaufgabe!

Aufgabe 3 (*Analyse:* preflow-push *Algorithmus (Wiederholung)*)

Sei durch S, T ein minimaler (s, t) Schnitt gegeben. Zeigen oder widerlegen Sie folgende Eigenschaften der Distanzfunktion:

a) $\forall v \in T : d(v) < n$

b) $\forall v \in S : d(v) \geq n$

Aufgabe 4 (*Analyse: Eigenschaften spezieller Knotenmengen*)

Gegeben sei ein ungerichteter Graph $G = (V, E)$. Sei $N(v) = \{w|(v, w) \in E\}$ die Menge aller Nachbarn von Knoten v. Man definiert folgende Teilmengen der Knotenmenge V des Graphen:

- *Vertex Cover (VC).* Ein VC ist eine Teilmenge $C \in V$, so dass f.a. Kanten $(u, v) \in E$ mindestens einer ihrer Endpunkte in C enthalten ist. Üblicherweise ist ein minimales VC gesucht.
- *Dominating Set (DS).* Ein DS ist eine Teilmenge $D \in V$, so dass f.a. Knoten $v \in V$ entweder v oder mindestens ein $w \in N(v)$ in D enthalten ist. Üblicherweise ist ein minimales DS gesucht.
- *Independent Set (IS).* Ein IS ist eine Teilmenge $I \in V$, so dass für jeden Knoten $v \in I$ gilt, alle Knoten $w \in N(v)$ sind nicht in I enthalten. Üblicherweise ist ein maximales IS gesucht.

Man unterscheidet weiter zwischen einer *minimum* (*maximum*) und einer *minimalen*(*maximalen*) Teilmenge. Erstere bezeichnet ein globales Optimum, d.h. auf diesem Graphen kann keine kleinere (größere) Menge mit der geforderten Eigenschaft gefunden werden. Letztere gibt ein lokales Optimum an, d.h. man kann keinen Knoten aus der Menge entfernen (zu der Menge hinzunehmen) und die geforderte Eigenschaft erhalten.

Zeigen oder widerlegen Sie, ...

a) ein *Independent Set* ist genau dann ein *Dominating Set*, wenn es *maximal* ist.

b) ein *minimales Dominating Set* ist ein *maximales Independent Set*.

c) ist C ein *minimales Vertex Cover*, so ist V/C ein *maximales Independent Set*.

d) ein *minimales Vertex Cover* mit allen isolierten Knoten von G ist ein *Dominating Set*.

e) ein *minimales Dominating Set* ohne isolierte Knoten ist ein *Vertex Cover*.

Aufgabe 5 (*Entwurf: Paketvermittlung*)

Der Weihnachtsmann ist spät dran mit dem Verteilen der Geschenke an seine Außenlager, von wo aus sie dann an Weihnachten in alle Wohnungen geliefert werden. Es ist ein ständiges Kommen und Gehen am Nordpol und der Lagermeister hat alle Hände voll zu tun. Gerade sind die folgenden Geschenke zum Versand freigegeben worden:

- 4× Lego City Feuerwehrlöschzug
- 1× Barbies Traumhaus
- 2× Ferngesteuerter Formel 1 Rennwagen von Ferrari
- 1× Rotkäppchen Kostüm
- 3× Wii Mario Edition (Red Edition)
- 3× Kosmos electronic Baukasten
- 1× Pony mit großen Kulleraugen

Es stehen auch einige Rentierschlitten zum Abtransport bereit. Allerdings haben die lieben Tiere so ihre Eigenarten, was es dem Lagermeister nicht leichter macht, seine Aufgabe effizient zu erfüllen.

- *Rudolph* kann 10 Pakete transportieren, allerdings weigert er sich alles zu transportieren, das nicht überwiegend rot ist.
- Der alte *Claus* kann nur noch 1 Paket mitnehmen, dafür ist er nicht wählerisch, was den Inhalt des Pakets angeht. Hauptsache es gibt nach getaner Arbeit einen Sack voll Hafer.
- *Chuck* transportiert kein Mädchenspielzeug. Das passt einfach nicht zu ihm, darüber hinaus auch nicht mehr als 4 Pakete.
- *Q* ist ein Technikfreak. Er liebt alles was Strom benötigt. Leider kann er nur 2 dieser Gerätschaften mit seinem tiefergelegten Schlitten ziehen.

Wie verteilt der Lagermeister diese Geschenke optimal auf die Schlitten? Können alle Geschenke mit nur einer Beladung jedes Schlittens abtransportiert werden?
Modellieren Sie die Problem als Flussgraph und beantworten Sie die obigen Fragen.

Hinweis: Das Flussproblem kann wieder durch scharfes Hinsehen gelöst werden. Es muss kein Flussalgorithmus ausgeführt werden.

Aufgabe 6 (*Kleinaufgaben: Parallele Algorithmen*)

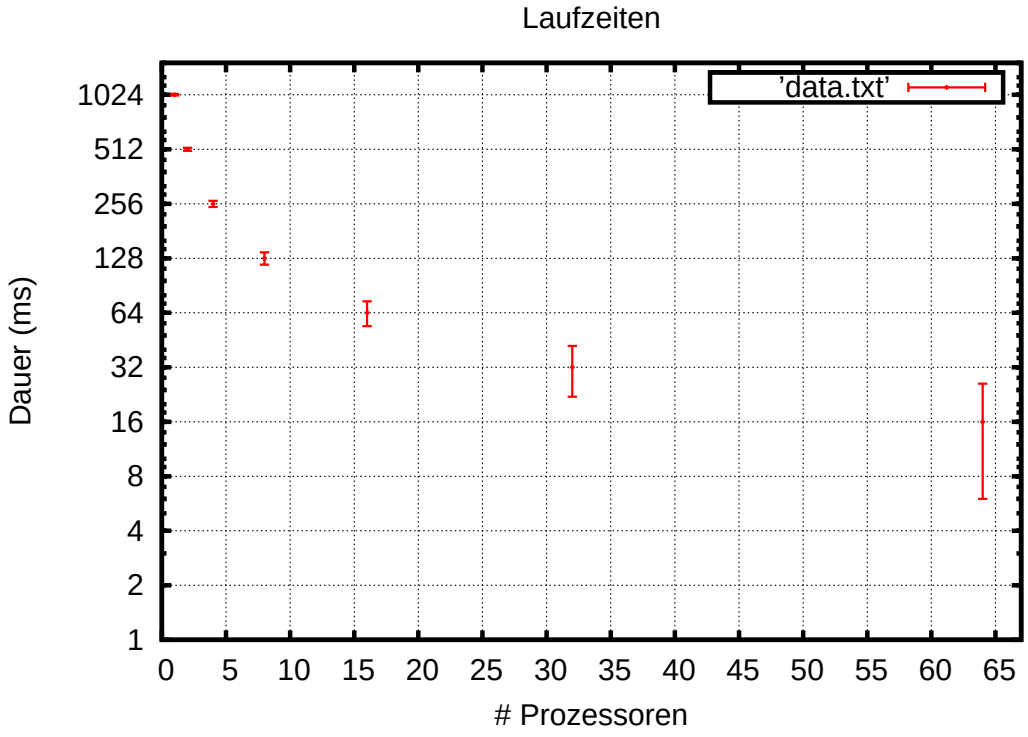
a) Gegeben sei ein paralleler vergleichsbasierter Sortieralgorithmus zum Sortieren von n komplexen Objekten auf p Prozessoren mit einer Laufzeit von

$$T(p) := \Theta(\frac{n^2 \log^2 n}{p^2}).$$

Geben Sie den absoluten *speed-up* und die *efficiency* an.

b) Wie muss in der vorherigen Teilaufgabe die Prozessorzahl p mit der Eingabegröße n wachsen, damit der absolute *speed-up* konstant bleibt?

c) Sie haben für einen parallelen Algorithmus folgende Laufzeiten bei unterschiedlicher Prozessorzahl gemessen. Was können Sie über die Skalierung dieses Algorithmus aussagen?



Aufgabe 7 (*Entwurf+Analyse: CRCW und CREW Modelle*)

Gegeben sei ein Array $a[\cdot]$ im gemeinsamen Speicher, der n Zahlen hält.

a) Beschreiben Sie einen möglichst schnellen parallelen Algorithmus, der überprüft, ob eine der Zahlen durch 7 teilbar ist. Gehen Sie von $p = n$ Prozessoren und dem CRCW *common* Modell aus. Außerdem sei die Teilbarkeit in $O(1)$ prüfbar.

b) Wie ändert sich die Laufzeit, wenn Sie nur noch $p < n$ Prozessoren zur Verfügung haben?

c) Wieviel Zeit würde Ihr Algorithmus im CREW Modell benötigen (wieder $p = n$ Prozessoren)?

d) Geben Sie den absoluten *speed-up* und die *efficiency* für die vorherigen Teilaufgaben an.

e) Im Folgenden soll berechnet werden, wieviele der Zahlen in $a[\cdot]$ durch eine andere Zahl in $a[\cdot]$ (nicht durch sich selbst!) teilbar sind. Sie haben das CRCW Modell und $p = n^2$ Prozessoren zur Verfügung.

Aufgabe 8 (*Entwurf+Analyse:* `findif`*-Anweisung*)

Gegeben sei ein Array $a[\cdot]$ im verteilten Speicher der n Objekte hält. Gesucht ist ein Algorithmus, der eine parallele `findif` Anweisung auf $a[\cdot]$ ausführt. Die Anweisung sortiert die Elemente von $a[\cdot]$ anhand eines Prädikats $pred(\cdot)$, so dass Elemente, die das Prädikat erfüllen, vorne stehen. Die relative Ordnung der Elemente untereinander soll dabei erhalten bleiben.

Bsp.: `findif( {1,4,9,7,3},` *is_bigger_than_*`3 ) = {4,9,7,1,3}`

a) Beschreiben Sie einen Algorithmus, der eine parallele `findif` Anweisung auf $a[\cdot]$ möglichst schnell ausführt. Sie haben $p = n$ Prozessoren zur Verfügung.

b) Untersuchen Sie die Laufzeit der Anweisung für den Fall, dass $p = n$ Prozessoren zur Verfügung stehen und das Prädikat in $T(n) = O(1),\ O(\log n)$ bzw. $O(n)$ ausgewertet werden kann.

c) Wie verhalten sich die Laufzeiten, wenn Sie nur noch $p < n$ Prozessoren zur Verfügung haben?

Aufgabe 9 (*Entwurf+Analyse: Assoziative Operationen*)

Gegeben sei ein Array A im gemeinsamen Speicher bestehend aus n Objekten vom Typ X. Auf den Objekten sei eine Operator $\odot$ definiert. Es sei nach dem Ergebnis von $\odot_{i=1}^{n} a_i$ gesucht.

a) Sei $X = \mathbb{R}^2$ und der Operator definiert als

$$(x_1, x_2) \odot (y_1, y_2) := (x_1 y_1, x_2 + y_2)$$

Zeigen Sie, dass der Operator $\odot$ assoziativ ist.

b) Beschreiben Sie einen schnellen parallelen Algorithmus, der $\odot_{i=1}^{n} a_i$ berechnet und geben Sie dessen Laufzeit $T(n, p)$ an.

c) Nun sei $\odot$ wie folgt definiert: X beschreibe die Menge an möglichen Zeichenketten über dem Alphabet $\{(,)\}$. Die Operation $x \odot y$ verknüpfe beide Zeichenketten und schiebe alle öffnenden Klammern nach links, alle schließenden Klammern nach rechts (Bsp.: $()(() \odot (()) = ((((()))))$.

Können Sie den selben Lösungsansatz wie in der vorherigen Teilaufgabe verwenden? Falls nein, geben Sie einen neuen parallelen Algorithmus an. Wie lange dauert die Ausführung?

Ausgabe: 20.12.2011
Abgabe: keine Abgabe, keine Korrektur