Nicole Kaiser fl Tutorium: Do. 12-14 Uhr fl Tutor: Martin Klossek

14.11.2001

Theoretische Informatik 1, WS 2001/2002

Übungsblatt 4

Die hier vorgestellten Lösungen sind nicht immer richtig.
Aufgabe 4.1

Sei n die Anzahl der Leute, die bereits in der Firma beschäftigt sind. Wenn n gerade ist, sind davon $U(n) \leq \frac{n}{2} - 1$ Menschen unehrlich und $E(n) \geq \frac{n}{2} + 1$ Menschen ehrlich. Wenn n ungerade ist, gibt es $U(n) \leq \left\lfloor \frac{n}{2} \right\rfloor$ unehrliche und $E(n) \geq \left\lceil \frac{n}{2} \right\rceil$ ehrliche Leute.

Vorgehensweise: Finde einen ehrlichen Menschen und befrage ihn über alle anderen Mitarbeiter.

1. Schritt:
Befrage alle Mitarbeiter paarweise. Also Befrage (1,2) und (2,3) und (3,4) usw. Merke dabei alle Paarkombinationen und welche Antworten sie gegeben haben. Dabei kann es drei Kombinationen von U und E geben.

Fall 1:
UU, gibt mir keine konkrete Information, weil einer oder beide lügen oder die Wahrheit sagen können.
Fall 2:
EE, jeder sagt über den anderen, dass dieser die Wahrheit sagt.
Fall 3:
UE, da E die Wahrheit sagt, bekomme ich auf jeden Fall die Information, dass einer U ist.

Nach diesen Befragungen habe ich mindestens ein Paar herausgefunden, in dem ein U ist. Wie U und E nebeneinander stehen ist nicht von Bedeutung, weil mindestens einmal der Fall eintreten muss, dass ein U neben einem E steht.

In diesem Schritt, werden $2(n-1) = 2n-2$ Fragen gestellt, da bei jeder Kombination von Paaren zwei Fragen gestellt werden.

2. Schritt:
Die Paare, die nach meinen Informationen ein U enthalten, werden gestrichen. Jetzt bilde aus den Nachbarn der gestrichenen Paare neue Paare und befrage diese. Wenn z.B. Paar (4,5) gestrichen wurde, bilde ein neues Paar aus 3 und 6.
Nun gibt es wieder mindestens ein Paar (U,E) oder (E,U), das mir die Information gibt, dass mindestens einer der Partner ein Lügner ist. Dieses Paar wird gestrichen. Bilde nun wieder neue Paar.

Schritt 2 wird höchstens $U(n)-1$ mal durchgeführt, weil im ersten Schritt schon ein U enttarnt wurde. Bei jeder Befragung werden wieder zwei Fragen pro Paar gestellt, im Worst Case also $2(U(n)-1)$ Fragen.

Da es mehr E als U gibt, ist nach den Paarsteichungen entweder ein E übriggeblieben (wenn n ungerade war) oder es ist nur noch ein Paar (E,E) da. In jedem Fall habe ich mindestens ein E gefunden.

3. Schritt:
Dieses E befrage ich nun über alle anderen $(n-1)$ Mitarbeiter. Also $(n-1)$ Fragen.

Laufzeit:
Sei $F(n)$ die Anzahl der Fragen, die bei diesem Verfahren gestellt werden.

$$F(n)=2(n-1)+2(U(n)-1)+(n-1) \;\leq\; 2(n-1)+2\left(\frac{n}{2}-1\right)+(n-1)$$

$$= \;\leq\; 2(n-1)+2\left(\frac{n}{2}-1\right)+(n-1)$$

$$=2n-2+n-2+n-1=4n-5=O(n)$$

weil $4n-5 \leq c \cdot n$ für $c \geq 4$.

Aufgabe 4.2

(a)
Idee:
Sei |T| die Anzahl der Knoten im Baum T.
Um alle Knoten aus v mit ihrer Höhe anzugeben, wird zunächst mit dem auf dem Aufgabenblatt vorgestelltem Algorithmus die Tiefe jedes Knotens berechnet. Der Knoten mit der größten Tiefe hat die Höhe 1, da er nur noch Blätter als Kinder hat. Seine Blätter haben die Höhe 0.

Für jeden unmarkierten Knoten wird nun abgefragt, ob seine Geschwister Kinder haben oder nicht. Wenn die Geschwister kinderlos sind, sind sie Blätter. Blätter und ihr Höhe werden nicht ausgegeben. Alle besuchten Knoten werden markiert. Falls die Geschwister Kinder habe, wird das Programm später nochmal für diese Kinder, die ja noch unmarkiert sind, aufgerufen.

Wenn es keine unmarkierten Knoten mehr gibt, sind alle Knoten mit ihrer Höhe ausgegeben worden.

Algorithmus:

```
Höhe (v Zeiger auf Baumknoten, Level Integer) {

    for (alle Knoten unter v) do {
        tiefe(v Zeiger auf Baumknoten, Level Integer);
        }

    sei i der Knoten mit der größten Tiefe n;
    markiere alle Kinder von i;

    Höhe(i) := 1;
    Höhe(Geschwister):=0;

    HöheV(Knoten i, Level) {
        if (i markiert) then goto Marke1;
        ausgabe(i, Höhe(Geschwister)+1);
        markiere i;

        for (alle Geschwister von i) do {
            if (Geschwister markiert) then break;
            elseIf (Geschwister.LKind = Nil) then
                Höhe(Geschwister)=0;
                markiere Geschwister;
            else
                break;

        }
        Level = Level -1;
        if Level <= 0 then goto Marke1;
        i := i.Vater;
        HöheV(Knoten i, Level);

    } // Ende Methode HöheV

    Marke1
```

```
        if (es gibt noch unmarkierte Knoten) then
                sei i der unmarkierte linkeste Knoten mit der größten Tiefe m;        // wobei m <= n
                markiere alle Kinder von i;
                HöheV(Knoten i, Level);
        else
                Programm Ende.
}
```

Laufzeit:
tiefe wird genau einmal aufgerufen und benötigt O(|T|).Bei der Abarbeitung von HöheV() und Marke1 wird jeder Knoten einmal durchlaufen, benötigt also O(|T|).
Die Laufzeit für Höhe() ist also O(|T|).