Goethe-Universität Frankfurt am Main 2. Dezember 2014
Arbeitsgruppe Theoretische Informatik, Institut für Informatik
Dr. Dominik Freydenberger, Hannes Seiwert, Prof. Dr. Georg Schnitger

Diskrete Modellierung

Wintersemester 2014/2015

Übungsblatt 7

Abgabe: *bis 9. Dezember 2014, 8.^{15} Uhr (vor der Vorlesung oder im Briefkasten zwischen den Räumen 114 und 115 in der Robert-Mayer-Str. 11–15)*

Auf diesem Blatt können durch Lösen einer Bonusaufgabe bis zu 30 Extrapunkte erreicht werden. Das heißt, Sie können bis zu 130 Punkte erreichen. Davon werden nur 100 Punkte zur maximal erreichbaren Punktzahl addiert.

Aufgabe 1: **(25 Punkte)**

Der wahnsinnige Wissenschaftler Dr. Fiesmod musste vor einigen Jahren wegen mehrerer Verstöße gegen alle Ethikrichtlinien, eine Vielzahl von Gesetzen sowie die Parkordnung seine Stelle an einer (hier ungenannt bleibenden) Universität aufgeben. Stattdessen hat er sich in einer finsteren Burg, die er geerbt hat, ein Labor eingerichtet. Dort forscht er nun – wenn er nicht gerade damit beschäftigt ist, über seine Rache nachzudenken. Bei seiner Arbeit unterstützen ihn seine fünf Handlanger Igor, Agor, Egor, Ogor und Ugor. Eines Tages beschließt Dr. Fiesmod, dass ihm die Burg zu unordentlich ist. Folgende Aufgaben müssen seiner Meinung nach sofort erledigt werden: das Entalgen des Haifischbeckens, das Polieren der Laserkanonen, das Entsorgen des Atommülls, das Ölen der Fallgruben und das Entrümpeln des Burggrabens.

Dr. Fiesmod beschließt, dass jeder der Handlanger genau eine dieser Aufgaben übernehmen soll, und befiehlt ihnen, sich auf eine Verteilung zu einigen. Die Handlanger setzen sich zusammen und stellen Folgendes fest:

(i) Igor weigert sich den Laser zu polieren,
(ii) Agor mag sich weder um den Burggraben, noch um den Atommüll, noch um das Haifischbecken kümmern,
(iii) Egor will weder den Atommüll noch den Laser übernehmen,
(iv) Ogor hat eine Abneigung gegen jede Aufgabe außer gegen das Ölen der Fallgruben,
(v) Ugor will nicht den Atommüll entsorgen und auch nicht den Burggraben entrümpeln.

(a) Stellen Sie einen *ungerichteten* Konfliktgraphen auf, dessen Knotenmenge die Handlanger und die Aufgaben repräsentiert. Eine Kante in diesem Graphen zwischen Handlanger A Aufgabe B soll dafür stehen, dass A nicht B übernehmen will.

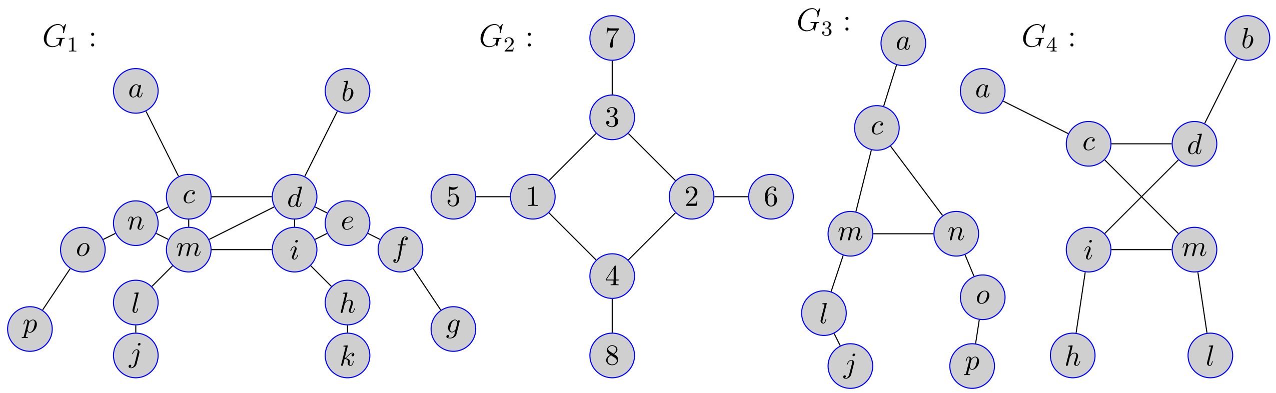
(b) Geben Sie auf der Grundlage Ihres Konfliktgraphen einen *ungerichteten* „Zufriedenheitsgraphen“ mit der gleichen Knotenmenge an. Eine Kante zwischen Handlanger A und Aufgabe B soll bedeuten, dass A einverstanden wäre B zu übernehmen.

(c) Geben sie ein Matching maximaler Größe in Ihrem „Zufriedenheitsgraphen“ an.

(d) Geben Sie eine Zuordnung zwischen Handlangern und Aufgaben an, so dass jeder Handlanger genau eine Aufgabe erhält, jede Aufgabe genau einmal zugeordnet wird und alle Handlanger mit ihrer Zuordnung zufrieden sind.

Aufgabe 2: **(25 Punkte)**

Betrachten Sie die folgenden ungerichteten Graphen:



Für welche $i, j \in \{1, 2, 3, 4\}$ gilt $G_i \cong G_j$? Falls $G_i \cong G_j$ gilt, geben Sie einen Isomorphismus von G_i nach G_j an. Ansonsten begründen Sie, warum ein Isomorphismus nicht existieren kann.

Aufgabe 3: **(25 Punkte)**

In der Vorlesung wurden "Syntaxbäume" aussagenlogischer Formeln und "Rekursionsbäume" rekursiver Programme vorgestellt (siehe auch die Folien 22,23,25 im Foliensatz "Bäume").

(a) Geben Sie den Syntaxbaum der folgenden aussagenlogischen Formel an:

$$\neg(\neg((V_1 \vee V_2) \rightarrow V_3) \leftrightarrow \neg((V_3 \wedge V_1) \wedge \neg V_2))$$

(b) Geben Sie den Rekursionsbaum des rekursiven Programms Hanoi(N, s1, s2 s3) an, das die Türme von Hanoi für N := 3 löst, d.h. 3 Ringe werden von Stab s1 nach s2 bewegt. Geben Sie zusätzlich für jeden Knoten des Rekursionsbaums die Konfiguration der Stäbe und Ringe nach Ausführung des entsprechenden rekursiven Aufrufs an.

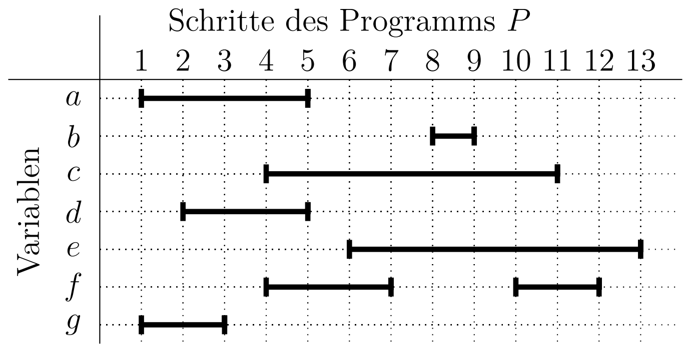
```
void Hanoi(int N,int s1,int s2,int s3)
{
  if (N==1)
    Bewege(s1,s2);
  else {
    Hanoi(N-1,s1,s3,s2);
    Bewege(s1,s2);
    Hanoi(N-1,s3,s2,s1);
  }
}
```

Die Funktion Bewege(s1, s2) bewegt hierbei einen Ring von Stab s1 auf Stab s2, ohne dabei weitere rekursive Aufrufe zu verursachen.

Aufgabe 4: **(25 Punkte)**

Die meisten Programmiersprachen vermitteln dem Programmierer den Eindruck, ihm stünden potentiell unbeschränkt viele Variablen zur Verfügung, auf die praktisch gleichzeitig zugegriffen werden könne. Allerdings muss bei der tatsächlichen Ausführung des Programms jede Variable, auf die zugegriffen wird, im Hauptspeicher verfügbar sein. Dessen Größe ist durch die Hardware begrenzt. Bei der Programmierung von eingebetteten Systemen lohnt es sich aus Platz-, Energie- und Kostengründen den verwendeten Speicher so weit zu reduzieren, dass ein gegebenes Programm gerade noch darauf ausführbar ist.

Wir stellen uns ein Programm P vor, das die sieben Variablen $a, \ldots, g$ benutzt, die jeweils eine Hauptspeicherzelle zur Speicherung benötigen. Die nebenstehende Abbildung gibt an, zu welchen Zeitpunkten der Ausführung von P welche Variablen im Hauptspeicher vorhanden sein müssen. So muss die Variable c beispielsweise in den Schritten 4 bis 11 von P im Hauptspeicher vorliegen. Zwei Variablen stehen im Konflikt miteinander, wenn sie nicht dieselbe Speicherzelle benutzen dürfen, da sie gleichzeitig im Hauptspeicher vorhanden sein müssen. Ziel der Aufgabe ist es, herauszufinden, wie viele Zellen der Hauptspeicher zur Ausführung von P haben muss, und in welche Zellen die Variablen im Verlaufe des Programms abgelegt werden können.



(a) Geben Sie den ungerichteten Konfliktgraphen $G = (V, E)$ an, der als Knotenmenge die Variablen besitzt und bei dem eine Kante für einen Konflikt zwischen zwei Variablen steht.

(b) Sei $s(P)$ die minimale Anzahl an Speicherzellen, die zur Ausführung von P benötigt wird. Beschreiben Sie den Zusammenhang zwischen $s(P)$ und der chromatischen Zahl des Konfliktgraphen $\chi(G)$.

(c) Bestimmen Sie $s(P)$.

Bonusaufgabe: **(30 Extrapunkte)**

Der Soziologe Norbert E. Twork befasst sich im Rahmen seiner Promotion mit Sozialen Netzwerken. Dabei handelt es sich um ungerichtete Graphen, deren Knoten Menschen entsprechen und deren Kanten soziale Beziehungen zwischen diesen Menschen repräsentieren. Norbert hat Kindergartengruppen untersucht und dabei festgestellt, dass in jeder Kindergartengruppe stets drei Kinder existieren, die alle miteinander befreundet sind, oder eine Gruppe von drei Kindern, die alle nicht miteinander befreundet sind. Zunächst machte Norbert sich Hoffnungen, dass seine Endeckung auf eine soziologische Besonderheit von Kindergartengruppen hindeutet, die er im Rahmen seiner Dissertation weiter untersuchen könnte. Norberts Bekannte Dana Ismod machte ihn dann allerdings auf einen Schwachpunkt seines Plans aufmerksam: Norbert hatte nur Kindergartengruppen mit mindestens 6 Kindern untersucht. Dana erklärt Norbert, dass dieses Phänomen in allen sozialen Netzwerken von mindestens 6 Menschen auftritt.

Beweisen Sie, dass Dana Recht hat.

Hinweis: Wählen sie aus einem entsprechenden sozialen Netzwerk einen Menschen aus. Was passiert, wenn dieser mit mindestens drei Menschen befreundet ist?