

AUFGABE 44 (8 Punkte):

[+++,*] Die folgende Aufgabe ist eine sehr schöne Bastelaufgabe. Sie zeigt, wie man mit Hilfe von kleinen Graphbausteinen regelrecht „programmieren“ kann. Und es gilt: Viel Text, aber vergleichsweise einfach!

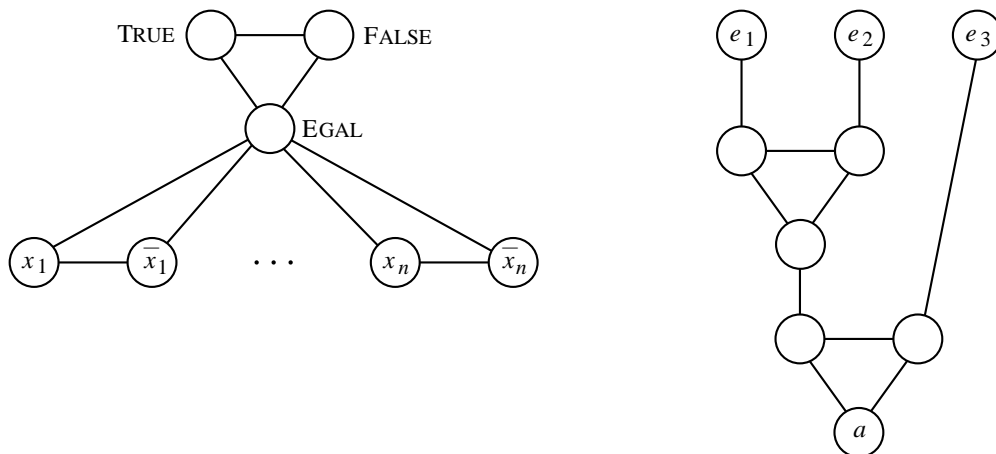
Sei $k \in \mathbb{N}$. Eine k -Färbung eines Graphen $G = (V, E)$ ist eine Abbildung $c : V \rightarrow \{1, \dots, k\}$, so daß für alle Kanten $\{u, v\} \in E$ gilt: $c(u) \neq c(v)$.

Das Dreifärbungsproblem ist die Sprache

$$3\text{COL} = \{ \langle G \rangle \mid G \text{ ist 3-färbbar} \} .$$

Wir wollen in dieser Aufgabe zeigen: $3\text{SAT} \leq_p 3\text{COL}$. Da offensichtlich $3\text{COL} \in \text{NP}$ ist, ist 3COL also NP-vollständig.

Betrachten Sie die beiden dargestellten Graphen, die als Bausteine in der Konstruktion eines größeren Graphen benutzt werden sollen.



- Sie haben die drei Farben $\{\text{TRUE}, \text{FALSE}, \text{EGAL}\}$. Was stellt der linke Graph sicher?
- Betrachten Sie nun den rechten Graphen. Welche Farbe bekommt „Ausgang“ a , wenn alle drei „Eingänge“ e_1, e_2 und e_3 die Farbe FALSE haben? Wenn einer der Eingänge die Farbe TRUE hat, kann man dann die Farbe TRUE auf den Ausgang bringen?
- Benutzen Sie die beiden Bausteine, um zur KNF Φ mit Klauseln der Länge 3 einen Graphen G_Φ zu konstruieren, so daß G_Φ genau dann 3-färbbar ist, wenn Φ erfüllbar ist (denken Sie daran, daß sie eine „genau dann, wenn“-Beziehung zeigen müssen).

Hinweis: Benutzen Sie für jede Klausel eine Kopie des rechten Bausteins. Legen Sie die Eingänge und Ausgänge auf Knoten des linken Graphen. Z. B. sollten *alle* Ausgänge auf dem Knoten TRUE liegen (Warum?).

Führen Sie ihre Konstruktion für $\Phi = (x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$ und eine erfüllende Belegung durch.

- Φ möge eine KNF über n Variablen sein und aus t Klauseln bestehen. Wieviele Knoten und Kanten hat G_Φ und wie groß ist der Grad?

Zur Erinnerung: Sei $G = (V, E)$ ein Graph. Der Grad $\delta_G(v)$ eines Knotens $v \in V$ ist die Anzahl der Kanten, die er berührt (Fachsprache: zu denen er *inzident* ist). Der Grad $\Delta(G)$ des Graphen ist der größte vorkommende Knotengrad, also $\Delta(G) = \max\{\delta_G(v) \mid v \in V\}$.

Da es vielleicht doch einige Gewöhnungsschwierigkeiten bei der Beschreibung kombinatorischer „Bastel“-Probleme durch „binäre Programme“ (vgl. auch Aufgabe 43) gibt, möchten wir hier einmal die gesamte Konstruktion aus der Reduktion von SAT auf BP an einem Beispiel vorführen. Dabei ist BP die Sprache der „binären Programme“:

$$\text{BP} = \{ \langle A, \vec{b} \rangle \mid A \text{ ist eine } (n \times m)\text{-Matrix mit ganzzahligen Einträgen, } \vec{b} \text{ ist ein Vektor mit } m \text{ ganzzahligen Einträgen, und es gibt einen 0-1-Vektor } \vec{x} \in \{0, 1\}^n \text{ mit } A \cdot \vec{x} \leq \vec{b} \}$$

„Binär“ deswegen, weil die Einträge des unbekannt(!) Vektors \vec{x} nur aus $\{0, 1\}$ sein dürfen. $\langle A, \vec{b} \rangle$ nennt man ein *binäres Programm*. Sei

$$\Phi = (x_1 \vee x_2) \wedge (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_2)$$

eine KNF über $v = 2$ Variablen und mit $k = 3$ Klauseln. Die einzige erfüllende Belegung ist $x_1 = \text{TRUE}$ und $x_2 = \text{FALSE}$. Gemäß der Reduktion der Vorlesung wird diese KNF durch das links dargestellte Ungleichungssystem beschrieben. Rechts sehen Sie die äquivalente Darstellung, bei der nur noch die \leq -Relation verwendet wird.

$$\left| \begin{array}{rcl} z_1 + z'_1 & = & 1 \\ z_2 + z'_2 & = & 1 \\ z_1 + z_2 & \geq & 1 \\ z_1 + z'_2 & \geq & 1 \\ z'_1 + z'_2 & \geq & 1 \end{array} \right| \iff \left| \begin{array}{rcl} -1 \cdot z_1 - 1 \cdot z'_1 & \leq & -1 \\ 1 \cdot z_1 + 1 \cdot z'_1 & \leq & 1 \\ -1 \cdot z_2 - 1 \cdot z'_2 & \leq & -1 \\ 1 \cdot z_2 + 1 \cdot z'_2 & \leq & 1 \\ -1 \cdot z_1 - 1 \cdot z_2 & \leq & -1 \\ -1 \cdot z_1 - 1 \cdot z'_2 & \leq & -1 \\ -1 \cdot z'_1 - 1 \cdot z'_2 & \leq & -1 \end{array} \right|$$

Eine *binäre* Lösung ist $z_1 = 1, z'_1 = 0, z_2 = 0, z'_2 = 1$, was man auch als $\vec{x}^T = (1, 0, 0, 1)$ schreiben kann (T steht für „transponiert“). Machen Sie sich den Zusammenhang zwischen der erfüllenden Belegung und dem Lösungsvektor klar! Machen Sie sich auch klar (indem Sie z. B. eine Klausel wegnehmen), daß das resultierende Ungleichungssystem mehr als eine binäre Lösung haben kann, und wie dann aus einer solchen Lösung eine erfüllende Belegung bestimmt werden kann.

Das Ungleichungssystem muß nun noch in die „richtige“ Matrix-Form gebracht werden:

$$\underbrace{\begin{pmatrix} -1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & -1 & -1 \\ 0 & 0 & 1 & 1 \\ -1 & 0 & -1 & 0 \\ -1 & 0 & 0 & -1 \\ 0 & -1 & 0 & -1 \end{pmatrix}}_{A_\Phi} \cdot \underbrace{\begin{pmatrix} z_1 \\ z'_1 \\ z_2 \\ z'_2 \end{pmatrix}}_{\vec{x}} \leq \underbrace{\begin{pmatrix} -1 \\ 1 \\ -1 \\ 1 \\ -1 \\ -1 \\ -1 \end{pmatrix}}_{\vec{b}_\Phi}$$

A_Φ ist also eine (7×4) -Matrix mit Einträgen aus $\{-1, 0, 1\}$, \vec{x} eine (4×1) -„Matrix“ (also ein 4-Vektor) und \vec{b}_Φ ein 7-Vektor mit Einträgen aus $\{-1, 1\}$. Und nach allem Gesagten gilt: $\langle A_\Phi, \vec{b}_\Phi \rangle \in \text{BP}$. Allgemein ist A_Φ eine $(2v+k) \times (2v)$ -Matrix und \vec{b}_Φ ein $(2v+k)$ -Vektor, also alles zusammen polynomiell groß in $|\langle \Phi \rangle|$.

Firmen wie die französische ILOG (<http://www.ilog.com/>), die im Januar 2009 von IBM gekauft worden ist, mit einem Jahresumsatz von über 133 Millionen US-Dollar entwickeln sog. *Solver* für solche und verwandte Probleme. Das Flaggschiff von ILOG ist das bekannte Programm *cplex* (offizieller Name: IBM ILOG CPLEX). Dieser Firmenbereich stellt konsequent Leute ein, die einen deutlich erkennbaren Hintergrund in der Theoretischen Informatik haben, und nach dem Kapitel über die NP-Vollständigkeit wird Ihnen vielleicht klar, warum das so ist.