Interstaatliche Hochschule für Technik Buchs NTB, Sy — Diskrete Mathematik
Block Informatik — 11.7.2003

Serie 40 — Test

Name | Vorname | 30 Pt. total → Note

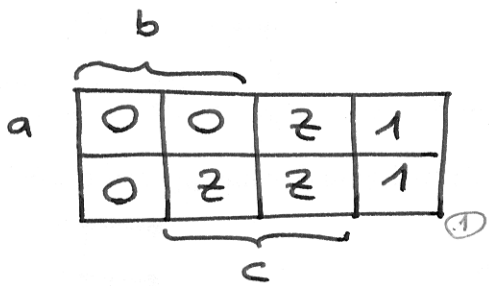
– Lösungen ohne verständliche Herleitung geben keine Punkte.
– Hilfsmittel: drei A4-Blätter, beidseitig, selber geschrieben.
– Zeit: 50 Minuten.

Aufgabe 1 (6 Pt.)

Pt. 6

Zeichnen Sie das Transistorschema nebenstehender Boolescher Funktion in CMOS-Technik. Der Z-Zustand entspricht einem hochohmigen Ausgang (*tri-state*), d.h. der Ausgang ist weder mit V_{DD} noch mit V_{SS} verbunden. Verwenden Sie möglichst wenige Transistoren, und vereinfachen Sie mit Karnaugh-Diagrammen.

c	b	a	y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	Z
1	0	1	Z
1	1	0	Z
1	1	1	0

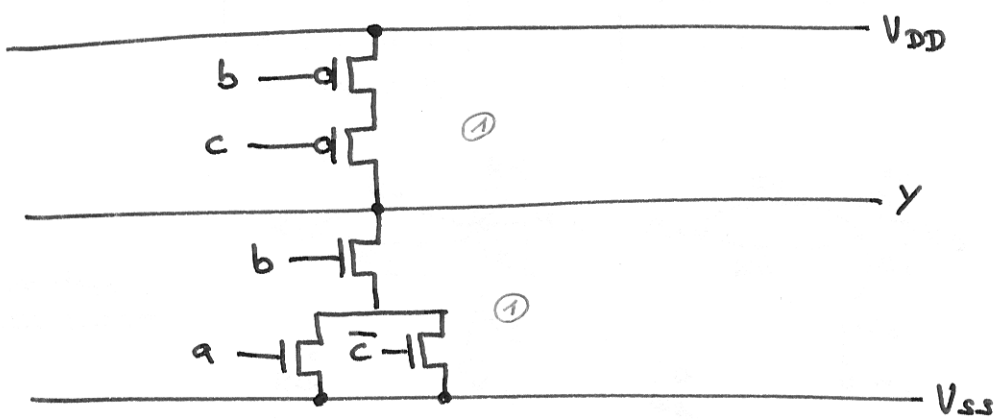


$y=1$: $\bar{b}\cdot\bar{c}$ (1)

$y=0$: $ab + b\bar{c} = b\cdot(a+\bar{c})$ (1) (1)

also:

wird vereinfacht: 6 Pt.
sonst alles richtig: 5 Pt.



6

Aufgabe 2 (2+3+2+2+3 Pt.)

12 Pt.

(a) Es gilt gemeinhin: *Hat ein ungerichteter Graf nur Knoten mit geradem Grad, so besitzt der Graf einen Euler-Zyklus.* Dieser Satz stimmt so nicht 100 %. Wo liegt der Fehler?

Der Graf muss auch zusammenhängend sein!

– praktisch nur 0/2 Pt.

2

(b) Stimmt folgende Aussage? (saubere Begründung)

Es gibt einen stark zusammenhängenden, gerichteten Grafen mit n Ecken ($n \geq 1$) und $n-1$ Kanten.

– "Ja": 0 Pt.

2 Pt. { $n \geq 2$: stimmt nicht, da zu jeder Ecke wird eine Kante hinführen muss.
– 2 Pt. nur, wenn saubere Begründung mit oben, oder minimum = n Kanten

+1 Pt. { $n = 1$: "•" Ist stark zusammenhängend und hat $1-1=0$ Kanten.

3

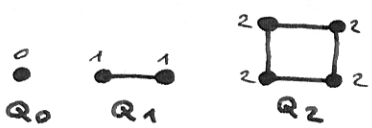
(c) Für welche Werte n hat K_n einen Hamiltonschen Kreis?

... d.h. $n \geq 3$ ($n \neq 2$) auch akzeptiert!

(c)+(d): – ohne Begründung: 0 Pt.
– unsaubere Begründung oder Detail-Fehler: 1 Pt.

2

(d) Für welche Werte n hat Q_n einen Eulerschen Kreis?



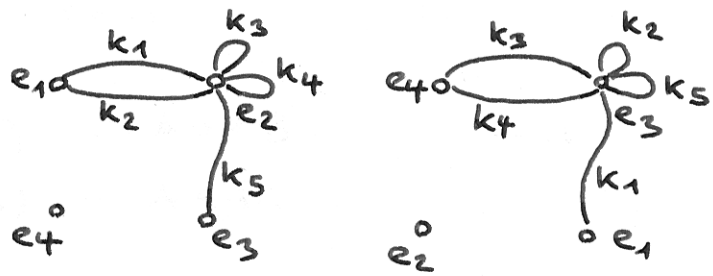
... d.h. n gerade

2

(e) Sind die Grafen mit folgenden Inzidenz-Matrizen isomorph? (saubere Begründung)

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Isomorphie:



3

– 3 Pt. nur bei korrekter Zuordnung.
– Nein: 0 Pt.

Aufgabe 3 (4 × 3 Pt.)

jeweils:
- 100% richtig: 3 Pt.
- wirkt ganz korrekt: 2 Pt.
- Ansätze: 1 Pt.

12 Pt.

(a) Was halten Sie von folgender Aussage? (saubere Begründung)

Kann man ein NP-Problem in polynomialer Zeit lösen, dann alle.

Falsch: Aussage würde gelten für NP-vollständige Probleme. Es gibt sehr viele NP-Probleme, die man in polynomialer Zeit lösen kann, nämlich alle aus P, und P ist eine Teilmenge von NP.

- ohne Antwort: 0 Pt.
- "richtig": 0 Pt.
- Konnte sehr zitiert: 1 Pt.

3

(b) Bei Problemen unterscheidet man gerne zwischen Entscheidungs-Problemen (so wie aus der Liste von Garey und Johnson) und Optimierungs-Problemen. Erklären Sie den Unterschied mit einem Beispiel beim bin-packing.

Entscheidung-Problem: Hat eine Menge von Items auf z.B. 5 Lastern Platz: ja oder nein?

optimierung-Problem: Wieviele Laster benötigt man mindestens für eine gewisse Menge von Items: Anzahl = ... ?

3

(c) Ein Programm für die Optimierung der Logistik in Ihrer Firma hat leider die Komplexität 2^n (n = Größe des Inputs in Bits). Jede Ausführung dauert 7 Stunden, was gerade noch akzeptabel ist. Sie möchten nun einen schnelleren Rechner kaufen, um größere Probleme in der gleichen Zeit bewältigen zu können. Die Nachfolgermodelle bieten 2×, 3×, ... 10× so große Geschwindigkeit. Schreiben Sie ein möglichst kleines MATLAB-Script, welches Ihnen für die Fälle 2× bis 10× berechnet, um wieviele Bits das Problem dann größer sein darf.

$k = 2, 3, \ldots, 10$: $2^u = 1 \cdot u$

neu: $2^u = \frac{1}{k} u$, d.h. $k \cdot 2^u = 1$, d.h. $2^x \cdot 2^u = 2^{(x+u)} = 1$

$x = \log_2 k$

- ohne MATLAB-Code: 0 Pt.

also:

```
for k = 2:10
    log(k) / log(2)
end
```

oder:

```
log(2:10) / log(2)
```

ultrakurz:

```
log2(2:10)
```

3

(d) Schreiben Sie ein Prolog-Programm "max", welches das Maximum zweier Zahlen bestimmt. In Prolog kann man Zahlen mit "X<Y" und "X>=Y" vergleichen. Beispiele:

```
?- max(2,3,X).
  X = 3

?- max(15,0,X).
  X = 15

?- 2<2.
  No
```

```
max(A, B, A) :- A >= B.
max(A, B, B) :- A < B.
```

3