66115
Herbst
2017

Theoretische Informatik / Algorithmen (vertieft)

Aufgabenstellungen mit Lösungsvorschlägen

Die Bschlangaul-Sammlung

Hermine Bschlangaul and Friends

Aufgabenübersicht

Die Bschlangaul-Sammlung

Hermine Bschlangaul and Friends

Eine freie Aufgabensammlung mit Lösungen von Studierenden für Studierende zur Vorbereitung auf die 1. Staatsexamensprüfungen des Lehramts Informatik in Bayern.

Thema Nr. 1

Aufgabe 2 [Kontextfreie Sprachen]

Betrachten Sie die Sprache $L_1 = L_a \cup L_b$.

- $L_a = \{ a^n bc^n \mid n \in \mathbb{N} \}$
- $L_b = \{ ab^m c^m \mid m \in \mathbb{N} \}$

(a) Geben Sie für L_1 eine kontextfreie Grammatik an.

Lösungsvorschlag

P = $\{$

$$\begin{aligned} S &\to S_a \mid S_b \\ S_a &\to aS_ac \mid b \\ S_b &\to a \mid aB_b \\ B_b &\to bB_bc \mid bc \end{aligned}$$

$\}$

(b) Ist Ihre Grammatik aus a) eindeutig? Begründen Sie Ihre Antwort.

Lösungsvorschlag

Nein. Die Sprache ist nicht eindeutig. Für das Wort abc gibt es zwei Ableitungen, nämlich $S \vdash S_a \vdash aS_ac \vdash abc$ und $S \vdash S_b \vdash aB_b \vdash abc$.

(c) Betrachten Sie die Sprache $L_2 = \{ a^{2^n} \mid n \in \mathbb{N} \}$. Zeigen Sie, dass L_2 nicht kontextfrei ist.

Lösungsvorschlag

Annahme: L_2 ist kontextfrei

$\to$ Pumping-Lemma gilt für L_2

$\to j \in \mathbb{N}$ als Pumping-Zahl

$\omega \in L_2$: $|\omega| \geq j$

Konsequenz: $\omega = uvwxy$

- $|vx| \geq 1$
- $|vwx| \leq j$
- $uv^iwx^iy \in L_2$ für alle $i \in \mathbb{N}_0$

Wir wählen: $\omega = a^{2^i}$: $|\omega| \geq j$

p $a \ldots a$

r $a \dots a$

s $a \dots a$

t $a \dots a$

q $a \dots a$

$q + r + s + t + q = 2^j$
$\Rightarrow r + t \geq 1$
$r + s + t \leq j$

1. Fall

$r + t = 2^{j-1}$
$2^{j-1} + 2^{j-1} = 2 \cdot 2^{j-1} = 2^1 \cdot 2^{j-1} = 2^{1+j-1} = 2^j$
$\omega' = uv^2wx^2y$
$p + 2 \cdot r + s + 2 \cdot t + q$
$p + s + q + 2 \cdot (r + t)$
$2^{j-1} + 2 \cdot 2^{j-1} = 3 \cdot 2^{j-1} = 2^{j-1} + 2^i \leq 2^{j+1}$
keine Zweierpotenz
$\Rightarrow \omega \notin L_2$
$\Rightarrow$ Widerspruch zur Annahme
$\Rightarrow L_2$ nicht kontextfrei

2. Fall

$r + t \neq 2^{j-1}$
$\omega' = uv^0wx^0y$
$\Rightarrow p + s + q = 2^j - (r + t)$
$(r + t) \neq 2^{j-i}$
ist keine Zweierpotenz
$\Rightarrow \omega \notin L_2$
$\Rightarrow L_2$ nicht kontextfrei

Aufgabe 3 [NAE3SAT]

Betrachten Sie die folgenden Probleme:

> **Exkurs: SAT**
> Das **Erfüllbarkeitsproblem der Aussagenlogik** SAT und K-SAT mit $k \geq 3$, $k \in \mathbb{N}$ (Satz von Cook) fragt, ob eine aussagenlogische Formel erfüllbar ist. Das Erfüllbarkeitsproblem der *Aussagenlogik* ist in exponen-

tieller Zeit in Abhängigkeit der Anzahl der Variablen mit Hilfe einer Wahrheitstabelle entscheidbar. Diese *Wahrheitstabelle* kann nicht in polynomieller Zeit aufgestellt werden.

Exkurs: NAE3SAT
Like 3-satisfiability, an instance of the problem consists of a collection of Boolean variables and a collection of clauses, each of which combines three variables or negations of variables. However, unlike 3-satisfiability, which requires each clause to have at least one true Boolean value, NAE3SAT requires that the three values in each clause are not all equal to each other (in other words, at least one is true, and at least one is false) [a]

[a] `https://en.wikipedia.org/wiki/Not-all-equal_3-satisfiability`

3SAT

Gegeben: Eine aussagenlogische Formel φ in konjunktiver Normalform (drei Literale pro Klausel).

Frage: Ist φ erfüllbar?

NAE-3SAT

Gegeben: Eine aussagenlogische Formel φ in konjunktiver Normalform (drei Literale pro Klausel).

Frage: Gibt es eine Belegung, die in jeder Klausel mindestens ein Literal *wahr* und mindestens ein Literal *falsch* macht?

Wir erlauben, dass `NAE-3SAT`-Formeln Literale der Form false haben, die immer *falsch* sind. So ist

$$(x_1 \lor \text{false} \lor \text{false}) \land (\neg x_1 \lor x_1 \lor x_1)$$

in `NAE-3SAT` (setze x_1 wahr).

(a) Zeigen Sie, dass sich `3SAT` in polynomieller Zeit auf `NAE-3SAT` reduzieren lässt.

Lösungsvorschlag

(b) Was können Sie aus a) folgern, wenn Sie wissen, dass `3SAT` NP-vollständig ist?

Lösungsvorschlag

(c) Was können Sie aus a) folgern, wenn Sie wissen, dass `NAE-3SAT` NF-vollständig ist?

Lösungsvorschlag

Aufgabe 8 [Greedy-Färben von Intervallen]

Sei X = (Ji,/2,..., eine Menge von n (geschlossenen) Intervallen über den reellen Zahlen R. Das Intervall Ij sei dabei gegeben dnrch seine linke Intervallgrenze Ij E R sowie seine rechte Intervallgrenze rj E R mit rj > Ij, ðIj = [lj,rj].

Wir nehmen in dieser Aufgabe der Einfachheit halber an, dass die Zahlen alle paarweise verschieden sind.

Zwei Intervalle Ij, 1 überlappen sich gdw. sie mindestens einen Punkt gemeinsam haben, ðgdw. falls für (o.B.d.A.) Ij < 4, auch 1 < Vj gilt. Eine gültige Färbung von X mit c e N Farben ist eine Funktion F : X (1,2,...,c) mit der Eigenschaft, dass für jedes Paar Ij,Ik von überlappenden Intervallen F(Ij) F(Ik) gilt.

Abbildung 1: Eine gültige Färbung von X

Eine minimale gültige Färbung von X ist eine gültige Färbung mit einer minimalen Anzahl an Farben. Die Anzahl von Farben in einer minimalen gültigen Färbung von X bezeichnen wir mit x(X). Wir gehen im Folgenden davon aus, dass für X eine minimale gültige Färbung F* gefunden wurde.

(a) Nehmen wir an, dass aus X alle Intervalle einer bestimmten Farbe von F* gelöscht werden. Ist die so aus F* entstandene Färbung der übrigen Intervalle in jedem Fall immer noch eine minimale gültige Färbung? Begründen Sie Ihre Antwort.

(b) Nehmen wir an, dciss aus X ein beliebiges Intervall gelöscht wird. Ist die so aus F* entstehende Färbung der übrigen Intervalle in jedem Fall immer noch eine minimale gültige Färbung? Begründen Sie Ihre Antwort.

(c) Mit uj(X) bezeichnen wir die maximale Anzahl von Intervallen in X, die sich paarweise überlappen. Zeigen Sie, dass x(A) > uj(X) ist. Wir betrachten nun folgenden Algorithmus, der die Menge X = (F,F ■ ..,In) von n Intervallen einfärbt:

- Zunächst sortieren wir die Intervalle von X aufsteigend nach ihren linken Intervallgrenzen. Die Intervalle werden jetzt in dieser Reihenfolge nacheinander eingefärbt; ist ein Intervall dabei erst einmal eingefärbt, ändert sich seine Farbe nie wieder. Angenommen die sortierte Reihenfolge der Intervalle sei Ia(i), ■ ■ ■ , F(n)-
- Das erste Intervall F(i) erhält die Farbe 1. Für 1 < i < n verfahren wir im Aten Schritt zum Färben des Aten Intervalls wie folgt:

  Bestimme die Menge Cj aller Farben der bisher schon eingefärbten Intervalle die /„(p überlappen. Färbe /„-(j) dann mit der Farbe c, = min((l,2,..., n) Cj). Fortsetzung nächste Seite!

(d) Begründen Sie, warum der Algorithmus immer eine gültige Färbung von X findet (Hinweis: Induktion).

(e) Zeigen Sie, dass die Anzahl an Farben, die der Algorithmus für das Einfärben benötigt, mindestens cü(X) ist.

(f) Zeigen Sie, dass die Anzahl an Farben, die der Algorithmus für das Einfärben benötigt, höchstens uj(X) ist.

(g) Begründen Sie mit Hilfe der o.g. Eigenschaften, warum der Algorithmus korrekt ist, ðimmer eine minimale gültige Färbung von X findet.

(h) Wir betrachten folgenden Implementierung des Algorithmus in Pseudocode:

Was ist die asymptotische Laufzeit dieses Algorithmus? Was ist der asymptotische Speicher bedarf dieses Algorithmus? Begründen Sie Ihre Antworten.

Thema Nr. 2

Aufgabe 5 [CYK mit fehlenden Zellen (T: SABC N: ab)]

Sei $G = (\{S, A, B, C\}, \{a, b\}, P, S)$ die kontextfreie Grammatik in Chomsky-Normalform und der Menge P der Produktionen:

$$P = \left\{ \begin{aligned} S &\to AB \mid BC \\ A &\to BA \mid a \\ C &\to AB \mid a \\ B &\to CC \mid b \end{aligned} \right\}$$

Sei $\omega = baaab$. Folgende Tabelle entsteht durch Anwendung des CYK-Algorithmus. Z. B. bedeutet $B \in V(3,5)$, dass aus der Variablen B das Teilwort $\omega_3\omega_4\omega_5 = aab$ hergeleitet werden kann. Drei Einträge wurden weggelassen.

(a) Bestimmen Sie die Mengen $V(1,2)$, $V(1,3)$ und $V(1,5)$.

Lösungsvorschlag

b	a	a	a	b
B	A,C	A,C	A,C	B
A,S	B	B	S,C	
-	S,C,A	B		
S,A,C	S,C			
S,C				

(b) Wie entnehmen Sie dieser Tabelle, dass $\omega \in L(G)$ ist?

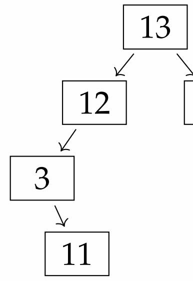
Lösungsvorschlag

In der Menge $V(1,5)$ ist das Startsymbol S der Sprache $L(G)$ enthalten.

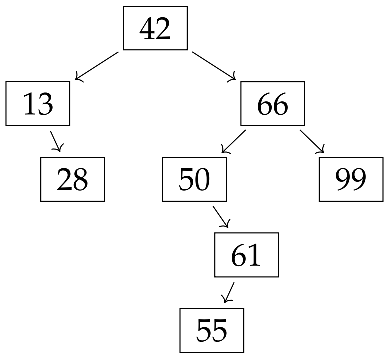
Aufgabe 8 [Binärbaum, Halde, AVL]

(a) Fügen Sie die Zahlen 13, 12, 42, 3, 11 in der gegebenen Reihenfolge in einen zunächst leeren binären Suchbaum mit aufsteigender Sortierung ein. Stellen Sie nur das Endergebnis dar.

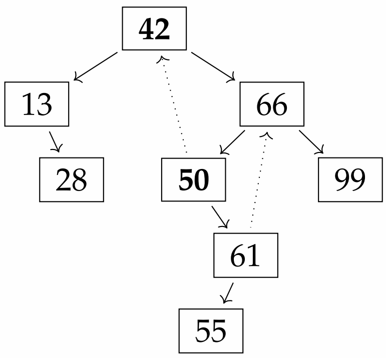
Lösungsvorschlag

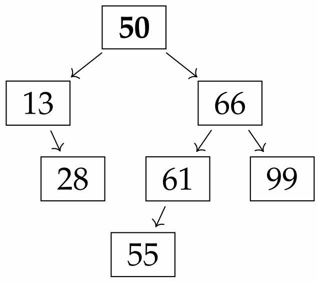


(b) Löschen Sie den Wurzelknoten mit Wert 42 aus dem folgenden *binären* Suchbaum mit aufsteigender Sortierung und ersetzen Sie ihn dabei durch einen geeigneten Wert aus dem *rechten* Teilbaum. Lassen Sie möglichst viele Teilbäume unverändert und erhalten Sie die Suchbaumeigenschaft.

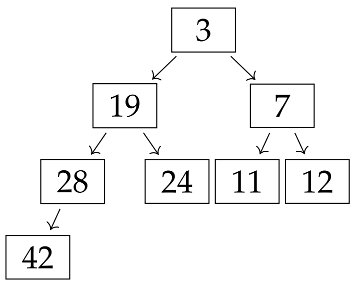


Lösungsvorschlag



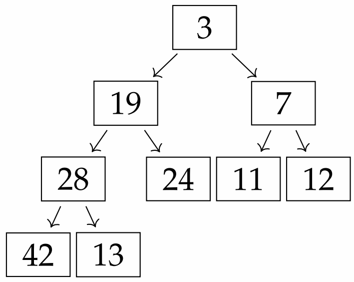


(c) Fügen Sie einen neuen Knoten mit dem Wert 13 in die folgende Min-Halde ein und stellen Sie anschließend die Halden-Eigenschaft vom neuen Blatt aus beginnend wieder her, wobei möglichst viele Knoten der Halde unverändert bleiben und die Halde zu jedem Zeitpunkt links-vollständig sein soll. Geben Sie nur das Endergebnis an.

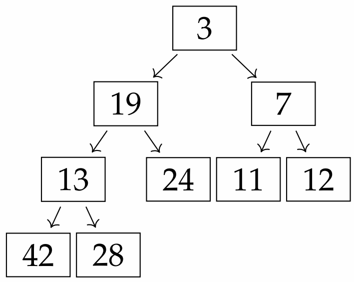


Lösungsvorschlag

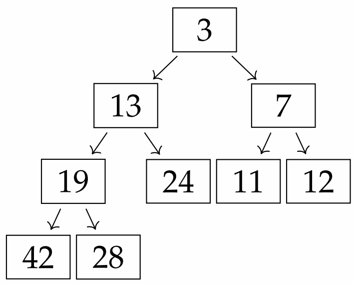
Nach dem Einfügen von „13“:



Nach dem Vertauschen von „13“ und „28“:



Nach dem Vertauschen von „13“ und „19“:

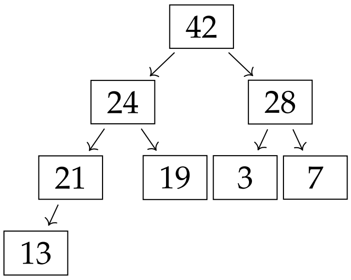


(d) Geben Sie für die ursprüngliche Min-Halde aus Teilaufgabe c)(ðohne den neu eingefügten Knoten mit dem Wert 13) die Feld-Einbettung (Array-Darstellung) an.

Lösungsvorschlag

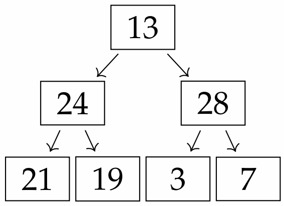
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
3	19	7	28	24	11	12	42

(e) Löschen Sie den Wurzelknoten mit Wert 42 aus der folgenden Max-Halde und stellen Sie anschließend die Halden-Eigenschaft ausgehend von einer neuen Wurzel wieder her, wobei möglichst viele Knoten der Halde unverändert bleiben und die Halde zu jedem Zeitpunkt links-vollständig sein soll. Geben Sie nur das Endergebnis an.

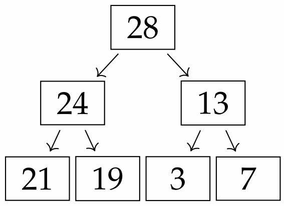


Lösungsvorschlag

Nach dem Ersetzen von „42“ mit „13“:

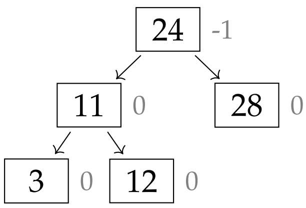


Nach dem Vertauschen von „13“ und „28“:



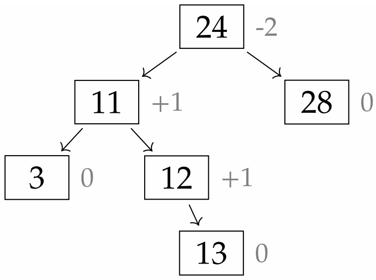
(f) Fügen Sie in jeden der folgenden AVL-Bäume mit aufsteigender Sortierung jeweils einen neuen Knoten mit dem Wert 13 ein und führen Sie anschließend bei Bedarf die erforderliche(n) Rotation(en) durch. Zeichnen Sie den Baum vor und nach den Rotationen.

(i) AVL-Baum A

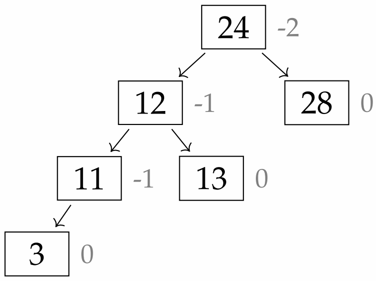


Lösungsvorschlag

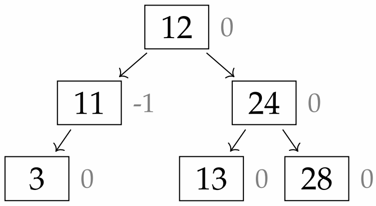
Nach dem Einfügen von „13“:



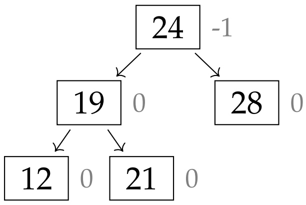
Nach der Linksrotation:



Nach der Rechtsrotation:

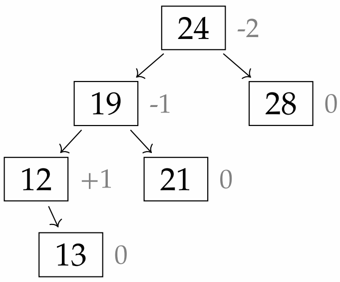


(ii) AVL-Baum B

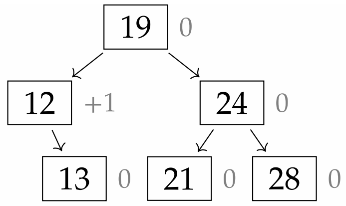


Lösungsvorschlag

Nach dem Einfügen von „13“:



Nach der Rechtsrotation:



19
0
12
+1
24
0
13
0
21
0
28
0