Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Dr. Volker Heun

Wintersemester 2001/2002
Semestralklausur
4. Februar 2002

Algorithmische Bioinformatik

Vorname	**Name**	**Matrikelnummer**
Reihe	**Platz**	**Unterschrift**

Hiermit stimme ich einer Veröffentlichung meines Klausurergebnisses dieser Semestralklausur unter Verwendung meiner Matrikelnummer im Internet zu. Ja ☐ Nein ☐

(Unterschrift)

Allgemeine Hinweise zur Semestralklausur

- Vor der Prüfung ist diese Seite mit Vornamen, Namen, Matrikelnummer, Reihe und Platz **leserlich mit Druckbuchstaben** zu versehen und zu unterschreiben.
- Bitte **nicht** in *roter* oder *grüner* Farbe bzw. mit Bleistift schreiben.
- Der Studentenausweis und ein amtlicher Lichtbildausweis ist bereit zu halten.
- Die reine Bearbeitungszeit beträgt **135 Minuten**.
- Es sind insgesamt 40 Punkte zu erreichen, zum Bestehen sind 17 Punkte nötig.

Hinweise zum Multiple-Choice-Teil

- Der rechte Rand ist frei zu lassen. Die Rückseiten und die letzten Seiten dürfen als Schmierpapier verwendet werden.
- Falls ein Kästchen versehentlich angekreuzt wurde, so ist ist dieses vollständig auszufüllen und unmittelbar rechts bzw. links daneben eine neues Kästchen zu malen: ■ ☐. Es gilt dann als nicht angekreuzt.
- Falls dieses Kästchen dennoch wieder ankreuzt werden soll, so ist das rechts bzw. links daneben gemalte Kästchen anzukreuzen: ■ ☒.
- Bei Antworten, die einen Zahlenwert verlangen, ist der konkrete Zahlenwert oder eine kurze Formel einzutragen, also z.B. 24 bzw. 4!.

Viel Glück und Erfolg!

Hörsaal verlassen von .... bis .... von .... bis ....

Vorzeitig abgegeben um ....

	Seite 2	Seite 3	Seite 4	A1	A2	A3	A4	A5	$\sum$
Erstk.									
Zweitk.									

Vorname: ______________ Name: ______________ Matrikelnummer: ____________

Aufgabe M1 (1 Punkt)

Sei $\Sigma = \{a, b\}$ ein Alphabet.

Kreuze zu jeder Aussage an, ob diese richtig [R] oder falsch [F] ist.

Jedes Wort aus Σ^+ besitzt einen eindeutigen Rand. .... ☐ R | F ☑

Für jedes Wort aus $\{a^n b^n a^m b^m \in \Sigma^+ \mid n, m \in \mathbb{N} \wedge n \neq m\}$ ist der Rand eindeutig, wobei hier $\mathbb{N} = \{1, 2, 3, \ldots\}$. .... ☑ R | F ☐

Aufgabe M2 (1 Punkt)

Betrachte $s = a\,a\,b\,a\,b\,a\,a\,b\,a\,b\,a\,a\,b\,a\,b\,a\,a\,b\,a\,b\,a\,a \in \{a, b\}^*$.

Kreuze zu jeder Aussage an, ob diese richtig [R] oder falsch [F] ist.

aab ist ein Rand von s. .... ☐ R | F ☑

$aababaa$ ist der eigentliche Rand von s. .... ☐ R | F ☑

Aufgabe M3 (1 Punkt)

Bei der Verwendung des Knuth-Morris-Pratt-Algorithmus ist bei der Suche nach der Zeichenreihe $s = s_0 \cdots s_9 = a\,b\,b\,a\,b\,b\,a\,\boxed{b}\,b\,a$ im Text $t \in \{a, b\}^*$ beim Vergleich von $s_7 = b$ (dem eingerahmten Zeichen) ein Mismatch aufgetreten.

Wie groß ist der anschließende Shift? **3**

Aufgabe M4 (1 Punkt)

Bei der Verwendung des Boyer-Moore-Algorithmus mit der (starken) Good-Suffix-Rule (aus der Vorlesung) ist bei der Suche nach $s = s_0 \cdots s_{13} = a\,b\,a\,a\,b\,b\,a\,a\,b\,a\,\boxed{a}\,b\,a\,a$ im Text $t \in \{a, b\}^*$ beim Vergleich von s_{10} (dem eingerahmten Zeichen) ein Mismatch aufgetreten.

Wie groß ist der anschließende Shift, wenn die (starke) Good-Suffix-Rule (aus der Vorlesung) verwendet wird? **6**

Aufgabe M5 (1 Punkt)

Sei $t = 100010101010100111 \in \{0, 1\}^*$ und $s = 11 \in \{0, 1\}^*$. Für die Methode von Karp-Rabin sei $V(x_0 \cdot x_1) = x_0 + 2x_1 \bmod 3$.

Wie viele Kandidaten für mögliche Treffer liefert die Methode von Karp-Rabin bei einer Suche von s in t, die nachträglich noch explizit auf Gleichheit verifiziert werden müssen? **5**

Aufgabe M6 (1 Punkt)

Kreuze zu jeder Aussage an, ob diese richtig [R] oder falsch [F] ist.

Der Suffix-Trie von $a^n b^n a^n b^n \in \{a, b\}^+$ für $n \in \mathbb{N}$ besitzt $O(n)$ Knoten .. ☐ R | F ☑

Sei $p \in \mathbb{N}$ eine Primzahl und $w \in \{0, 1\}^*$ die Binärdarstellung von p ohne führenden Nullen (z.B. für $p = 13$ ist $w = 1101$ oder für $p = 97$ ist $w = 1100001$.) Der Suffix-Baum für w hat $O(|w|)$ Knoten. .... ☑ R | F ☐

1 ☐
2 ☐

Vorname: ______________ Name: ______________ Matrikelnummer: ______________

Aufgabe M7 (1 Punkt)

Kreuze zu jeder Aussage an, ob diese richtig [R] oder falsch [F] ist.

Ist die Kostenfunktion eine Metrik, so ist die zugehörige Edit-Distanz gleich der zugehörigen Alignment-Distanz. . . . . ☑ R | F ☐

Für alle Sequenzen $s, t \in \Sigma^*$ gilt: $d(s,t) = d(s^R, t^R)$, wobei für eine Sequenz $s = s_1 \cdots s_n$ die Sequenz s^R die gespiegelte Sequenz $s^R = s_n \cdots s_1$ bezeichnet und d die Edit-Distanz ist. . . . . ☑ R | F ☐

Aufgabe M8 (1 Punkt)

Seien $s, t \in \Sigma^*$ zwei Sequenzen und σ ein Ähnlichkeitsmaß. Weiter sei $(\overline{s}, \overline{t})$ ein optimales globales Alignment, $(\hat{s}, \hat{t})$ ein optimales semi-globales Alignment und $(\tilde{s}, \tilde{s})$ ein optimales lokales Alignment von s und t.

Kreuze zu jeder Aussage an, ob diese richtig [R] oder falsch [F] ist.

Der Ähnlichkeitswert eines optimalen globalen Alignments $(\overline{s}, \overline{t})$ ist immer mindestens so groß, wie der eines optimalen lokalen Alignments $(\tilde{s}, \tilde{t})$. . . . . ☐ R | F ☑

Der Ähnlichkeitswert eines optimalen lokalen Alignments $(\tilde{s}, \tilde{t})$ ist immer mindestens so groß, wie der eines optimalen semi-globalen Alignments $(\hat{s}, \hat{t})$. . . . . ☑ R | F ☐

Aufgabe M9 (2 Punkte)

Gegeben seien die beiden Sequenzen $s = ATACG$ und $t = GTAACT$ und die folgende Kostenfunktion w für ein Distanzmaß:

$$w(x,y) = \begin{cases} 0 & \text{für } x = y \in \Sigma \quad \textit{(Match)} \\ 2 & \text{für } ((x = -) \wedge (y \in \Sigma)) \vee ((x \in \Sigma) \wedge (y = -)) \quad \textit{(Insertion/Deletion)} \\ 3 & \text{für } (x \neq y) \wedge (x, y \in \Sigma) \quad \textit{(Substitution)} \end{cases}$$

Fülle die folgende Tabelle für ein optimales **globales** Sequenzen Alignment aus.

		G	T	A	A	C	T
	0	2	4	6	8	10	12
A	2	3	5	4	6	8	10
T	4	5	3	5	7	9	8
A	6	7	5	3	5	7	9
C	8	9	7	5	7	5	7
G	10	8	9	7	8	7	8

1 ☐
2 ☐

Vorname: ______________ Name: ______________ Matrikelnummer: ______________

Aufgabe M10 (1 Punkt)

Sei $S = \{s_1, \ldots, s_k\}$ eine Menge von Sequenzen und sei M^* ein optimales mehrfaches Sequenzen Alignment für S bezüglich des Sum-Of-Pairs-Maßes.

Kreuze zu jeder Aussage an, ob diese richtig [R] oder falsch [F] ist.

Für $k = 3$ liefert die Center-Star-Methode ein mehrfaches Sequenzen Alignment, dessen Sum-Of-Pairs-Maß höchstens um den Faktor 4/3 von dem eines optimalen mehrfachen Sequenzen Alignments entfernt ist. .... ☑ R | F ☐

Es gilt immer, dass $D(M^*) \leq \sum_{i=1}^{k} \sum_{j=i+1}^{k} d(s_i, s_j)$ ist, wobei d die Alignment-Distanz und $D(M^*)$ der Wert des optimalen mehrfachen Sequenzen-Alignments M^* gemäß des Sum-Of-Pairs-Maßes ist. .... ☐ R | F ☑

Aufgabe M11 (1 Punkt)

Sei $S = \{s_1, \ldots, s_k\}$ eine Menge von Sequenzen. Sei weiter M ein mehrfaches Sequenzen Alignment für S und $\mathcal{S}_M$ der Konsensus-String von M. Ferner sei s^* der optimale Steiner-String von S.

Kreuze zu jeder Aussage an, ob diese richtig [R] oder falsch [F] ist.

Es gilt immer, dass der Alignment-Fehler von $\mathcal{S}_M$ größer gleich dem Konsensus-Fehler des optimalen Steiner-Strings für s ist, d.h. $E_M(\mathcal{S}_M) \geq E_S(s^*)$. .... ☑ R | F ☐

Es gilt immer, dass $s^* \in S$. .... ☐ R | F ☑

Aufgabe M12 (1 Punkt)

Sei $S = \{s_1, \ldots, s_5\}$ eine Menge von Sequenzen, für die ein mehrfaches Sequenzen Alignment mit Hilfe der Center-Star-Methode konstruiert werden soll. Die optimalen Alignment-Distanzen $d(s_i, s_j)$ für jedes Paar (s_i, s_j) für alle $i, j \in [1:5]$ sind in der Tabelle rechts angegeben.

d	s_1	s_2	s_3	s_4	s_5
s_1	0	3	2	3	2
s_2	3	0	2	3	4
s_3	2	2	0	2	3
s_4	3	3	2	0	4
s_5	2	4	3	4	0

Welche Sequenz wird als Zentrum gewählt? .... $\boldsymbol{s_3}$

1 ☐

2 ☐

Vorname: ______________ Name: ______________ Matrikelnummer: ____________

Aufgabe 1 (6 Punkte)

Weiter unten ist der Suffix-Trie T für das Wort $t = t_1 \cdots t_6 = aabaab$ angegeben.

a) Gebe alle Suffix-Links in T an.

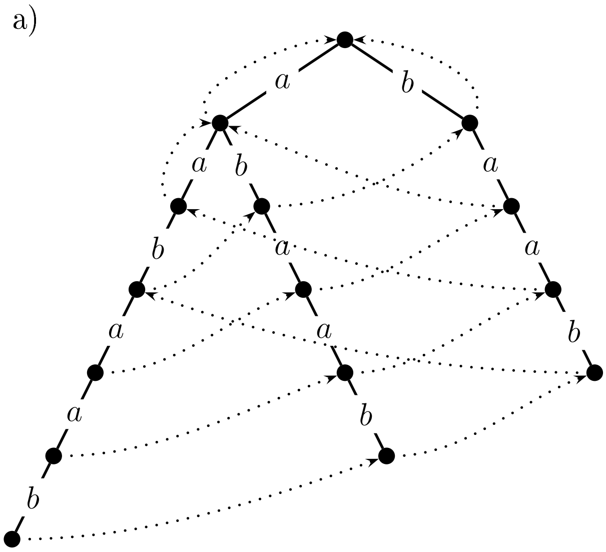
b) Gebe den zugehörigen Suffix-Baum für T an.

c) Konstruiere aus diesem mit Hilfe des Online-Algorithmus von Ukkonen schrittweise die Suffix-Tries für

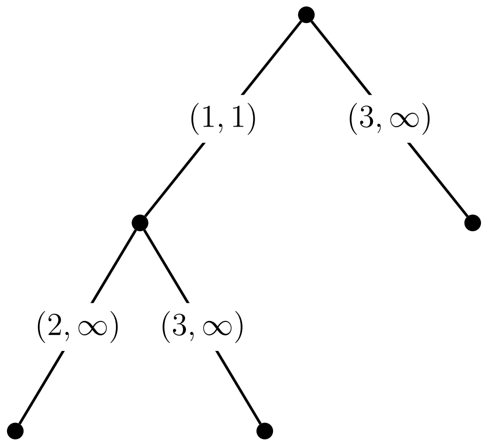
   1) $t' := t \cdot a = aabaaba$ und
   2) $t'' := t' \cdot b = aabaabab$.

   Gebe hierzu für jedes Wort t' und t'' eine eigene Zeichnung mitsamt der **verwendeten** und **neu konstruierten** Suffix-Links an.

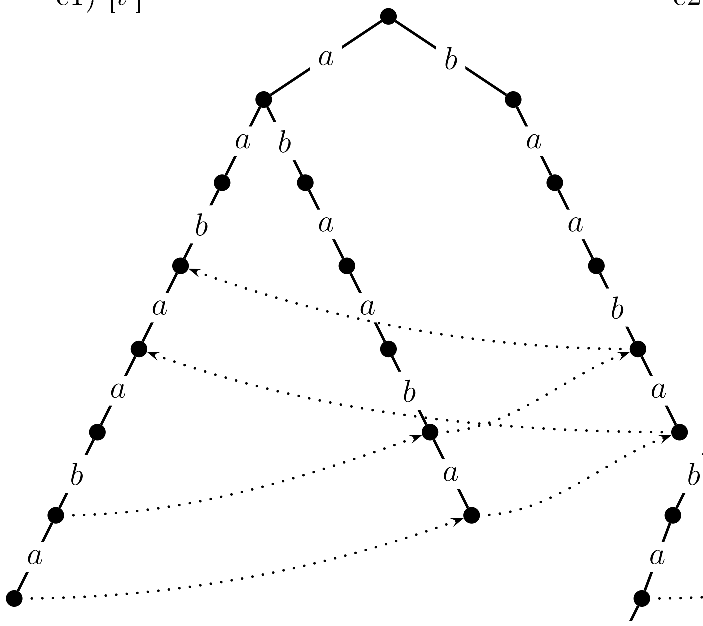
a)



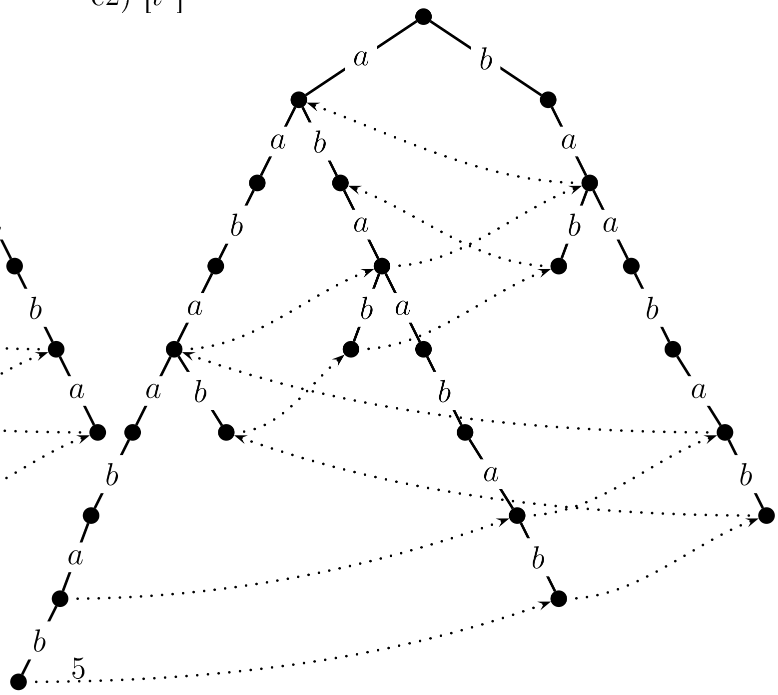
b)



c1) $[t']$



c2) $[t'']$



Vorname: ______________ Name: ______________ Matrikelnummer: ____________

Aufgabe 2 (5 Punkte)

Gegeben seien zwei Sequenzen $s, t \in \Sigma^*$ mit $|s| = n$ und $|t| = m$. Ein Alignment $(\overline{s}, \overline{t}) \in \overline{\Sigma}^* \times \overline{\Sigma}^*$ für s und t heißt *seltsam*, wenn gilt:

- $|\overline{s}| = |\overline{t}|$,
- $\exists i \in [0 : n] : \overline{s}|_\Sigma = s_1 \cdots s_i$,
- $\exists j, k \in [0, m] : \overline{t}|_\Sigma = t_j \cdots t_k$.

Anschaulich ist ein seltsames Alignment ein Alignment eines Teilwortes von t mit einem Präfix von s.

Gib einen Algorithmus mit Laufzeit $O(nm)$ an, der den Ähnlichkeitswert eines optimalen seltsamen Alignments berechnet, wobei $w : \overline{\Sigma} \times \overline{\Sigma} \to \mathbb{R}$ die Kostenfunktion für das Ähnlichkeitsmaß sei.

Begründe sowohl die Korrektheit als auch die Lauftzeit Deines Algorithmus.

Lösungsvorschlag

Da der Anfang des Wortes t wie bei einem semi-globalen Alignment nicht berücksichtigt werden muss, sind Einfügungen zu Beginn kostenlos. Hierfür muss also die erste Spalte mit 0 initialisiert werden. Da Anfänge von s berücksichtigt werden müssen, müssen Deletionen zu Beginn wie gewohnt bewertet werden. Daher gilt für die erste Spalte: $D[i, 0] = \sum_{\ell=1}^{i} w(s_\ell, -)$.

Die Rekursiongleichungen sehen wieder wie bei (semi-)globalen Alignments aus:

$$D[i, j] = \max \left\{ \begin{array}{l} D[i-1, j] + w(s_i, -), \\ D[i, j-1] + w(-, t_j), \\ D[i-1, j-1] + w(s_i, t_j) \end{array} \right\} \quad \text{für } (i, j) \in [1 : n] \times [1 : m].$$

Dies folgt aus der Tatsache, dass in einem optimalen seltsamen Alignment in der letzten Spalte entweder eine Deletion (erste Zeile der Maximumsbildung), eine Insertion (zweite Zeile der Maximumsbildung) oder ein Match bzw. Substitution steht (dritte Zeile der Maximumsbildung).

Wie findet man jetzt das optimale seltsame Alignment? Da Suffixe sowohl von s als auch von t nicht berücksichtigt werden müssen, müssen wie wie bei einem lokalen Alignment in der Tabelle D nur das Maximum bestimmen. Steht dieses an Position (i, k), so ist das optimale seltsame Alignment eines von $s_1 \cdots s_i$ mit $t_j \cdots t_k$. Wir müssen nur noch j bestimmen. Dazu verfolgen wir in der Tabelle D von der Position (i, k) einfach wieder Sieger der Maximumsbildung rückwärts. Sobald wir die erste Zeile erreicht haben, gibt die zugehörige Spalte die Position j in t an.

Da die Tabelle D $O(nm)$ Einträge hat und sich jede Maximumsbildung bzw. Initialisierung der ersten Zeile und ersten Spalte in konstanter Zeit durchführen lässt, ist der Zeitbedarf $O(nm)$. Die Rückverfolgung kostet auch nur $O(n + m) = O(nm)$.

Vorname: ______________ Name: ______________ Matrikelnummer: ______________

Aufgabe 3 (6 Punkte)

Sei $s = s_1 \cdots s_n \in \Sigma^*$ eine Sequenz. Eine Sequenz $s' = s'_1 \cdots s'_\ell \in \Sigma^*$ heißt *Oberfolge* von s, wenn es eine streng aufsteigende Folge $(k_1, \ldots, k_n) \in \mathbb{N}^n$ gibt (d.h., es gilt $k_1 < k_1 < \cdots < k_n$), so dass $s'_{k_i} = s_i$ für alle $i \in [1 : n]$ gilt.

Beispiel: Sequenzvergleich ist eine Oberfolge von *eure* und *enge* (jedoch keine kürzeste).

Seien $s = s_1 \cdots s_n \in \Sigma^*$ und $t = t_1 \cdots t_m \in \Sigma^*$ zwei Sequenzen. Konstruiere einen Algorithmus, der eine kürzeste Sequenz $u = u_1 \cdots u_\ell$ in Zeit $O(nm)$ findet, die sowohl eine Oberfolge von s als auch von t ist.

Begründe sowohl die Korrektheit als auch die Laufzeit Deines Algorithmus.

Lösungsvorschlag

Wir bemerken zuerst, dass wenn s eine Teilfolge von s' ist, dass dann s' eine Oberfolge von s ist. Wenn man also ein Alignment von zwei Sequenzen und den zugehörigen Edit-Graphen betrachtet, dann korrespondiert die Oberfolge zu einem Pfad von $(0,0)$ zu (n,m), wobei keine Substitutionen erlaubt sind. Für das zu einen solchen Pfad gehörige Alignment $(\overline{s}, \overline{t})$ gilt dann $\overline{s}|_\Sigma = s$ und $\overline{t}|_\Sigma = t$, d.h. $\overline{s}$ bzw. $\overline{t}$ ist eine Oberfolge von s bzw. t. Gibt es im Alignment $(\overline{s}, \overline{t})$ keine Substitutionen, dann ist die Folge $s' = s'_1 \cdots s'_{|\overline{s}|}$ mit $s'_i = \overline{s}_i$, falls $\overline{s}_i \neq -$ und $s'_i = \overline{t}_i$ andernfalls eine Oberfolge für beide Sequenzen.

Wie erhält man jetzt eine kürzeste Oberfolge? Matches, Insertionen und Deletionen verlängern die Sequenz um jeweils 1 (Substitutionen sind verboten). Wenn wir jetzt das Problem der kürzesten gemeinsamen Oberfolge über eine Alignment mit Hilfe der Dynamische Programmierung lösen wollen, so wählen wir am besten ein Distanzmaß mit folgender Kostenfunktion

$$w(x,y) = \begin{cases} \infty & \text{für } (x \neq y) \wedge (x \neq -) \wedge (y \neq -) \quad \text{(d.h. für eine Substitution)} \\ 1 & \text{sonst} \end{cases}$$

Mit Hilfe der Rekursionsgleichung für ein globales Alignment für $i, j > 0$

$$\begin{aligned} D[0,j] &= j \\ D[i,0] &= i \\ D[i,j] &= \min \left\{ \begin{array}{c} D[i,j-1] + w(-,t_j), \\ D[i-1,j] + w(s_i,-), \\ D[i-1,j-1] + w(s_i,t_j) \end{array} \right\} \end{aligned}$$

erhalten wir zum Schluss in $D[n,m] \geq 0$ die Länge einer kürzesten gemeinsamen Oberfolge von s und t, da wir für jeden Match, jede Insertion und jede Deletion eins hochzählen. Hierbei ist natürlich zu berücksichtigen, dass $x + \infty = \infty$ für alle $x \in \mathbb{Z}$. (Man könnte statt ∞ in einer Realisierung auch $n + m + 1$ einsetzen. Damit würden alle Werte echt größer als $n + m$ ∞ entsprechen.) Die Folge selbst erhält man wieder über die Rückverfolgung der Sieger bei der Minimusbildung. Dies lässt sich in Zeit $O(n + m)$ erledigen, wenn man sich bei der Minimumsbildung einen der Sieger gemerkt hat.

Da die Tabelle D $O(nm)$ Einträge hat und sich jede Minimumsbildung bzw. Initialisierung der ersten Zeile und ersten Spalte in konstanter Zeit erledigen lässt, ist der Zeitbedarf $O(nm)$.

Vorname: ______________ Name: ______________ Matrikelnummer: ____________

Aufgabe 4 (5 Punkte)

Konstruiere einen Algorithmus, der für zwei Sequenzen $s, t \in \Sigma^*$ mit $|s| = n$ und $|t| = m$ ein längstes gemeinsames Teilwort in Zeit $O(n + m)$ berechnet.

Begründe sowohl die Korrektheit als auch die Laufzeit Deines Algorithmus.

Lösungsvorschlag

Man konstruiere einen Suffixbaum für $v\$_1w\$_2$, wobei $\$_1, \$_2 \notin \Sigma$ zwei neue Zeichen sind. Dies geschieht in Zeit $O(n + m)$.

Anschließend geht man mit einer Tiefensuche über den Baum und löscht alle ausgehenden Kanten von $\$_1$. Somit sind alle Blätter des Suffixbaumes mit einem der beiden \$-Zeichen markiert. Diese kostet ebenfalls noch einmal Zeit $O(n + m)$.

Ein gemeinsames Teilwort von v und w ist nun ein Wort, das sowohl ein Präfix eines Suffixes von v als auch ein Präfix eines Suffixes von w ist. Damit korrespondieren gemeinsame Wörter zu Knoten des Suffixbaumes, in deren Teilbaum sowohl ein $\$_1$ als auch ein $\$_2$ auftritt. Mit Hilfe einer weiteren Tiefensuche lassen sich alle Knoten markieren, die einen Nachfolger besitzen der mit $\$_1$ markiert ist; analoges gilt für $\$_2$.Dazu liefert die Tiefensuche jeweils dem Aufrufer zurück, welche de beiden Dollarzeichen im zufgehörigen Teilbaum auftreten.

Hat man nun mit einer weiteren Tiefensuche die Level aller Knoten bestimmt, so kann man mit einer letzten Tiefensuche die Knoten auf einem tiefsten Level (mit größter Nummer) finden, die sowohl einen Nachfolger $\$_1$ als auch $\$_2$ besitzen. Diese sind dann die längsten gemeinsamen Teilwörter von v und w. Die fünf Tiefensuchen zusammen erfordern einen Zeitbedarf von $O(n + m)$.

Natürlich kann man die Anzahlen der Tiefensuchen minimieren, indem man mehrere Aufgaben bereits während einer Tiefensuche erledigt.

Aufgabe 5 (5 Punkte)

Betrachte für den Boyer-Moore Algorithmus die folgende, so genannte *weak Good-Suffix-Rule* oder *schwache Gute-Suffix-Regel*:

$$S[j] = \min \left\{ \sigma \in [1:m] : \begin{array}{c} (\sigma \leq j \wedge s_{j+1} \cdots s_{m-1} \in \mathcal{R}(s_{j+1-\sigma} \cdots s_{m-1})) \\ \vee\ (\sigma > j \wedge s_0 \cdots s_{m-1-\sigma} \in \mathcal{R}(s_0 \cdots s_{m-1})) \end{array} \right\}.$$

Hierbei bezeichnet $\mathcal{R}(s)$ die Menge aller Ränder des Wortes s.

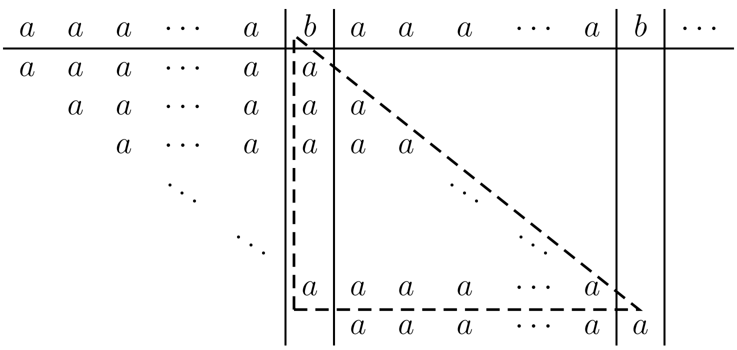
Im Gegensatz zur (strong) Good-Suffix-Rule bzw. (starken) Gute-Suffix-Regel aus der Vorlesung fehlt hier die zusätzliche Bedingung $s_j \neq s_{j-\sigma}$ in der oberen Zeile. Anschaulich ist der Unterschied zur (strong) Good-Suffix-Rule aus der Vorlesung, dass nun beim Verschieben nicht mehr sicher gestellt sein muss, dass an der Position des Mismatches nun ein anderes Zeichen aus s stehen muss als das, welches den Mismatch ausgelöst hat.

Gib für alle $n, m \in \mathbb{N}$ mit $n \geq 3m$ und $m \geq 3$ zwei Zeichenreihen s und t mit $|s| = m$ und $|t| = n$ an, so dass der Boyer-Moore-Algorithmus mit der oben genannten schwachen Gute-Suffix-Regel mindestens $\Theta(nm)$ Vergleiche ausführt.

Begründe, dass das Beispiel den Anforderungen genügt.

Lösungsvorschlag

Wir wählen $s = a^m$ und $t = (a^{m-1}b)^{\lfloor n/m \rfloor} \cdot a^{n \bmod m}$. Da wir beim Algorithmus von Boyer und Moore von rechts nach links vergleichen, findet der Mismatch beim Zeichen $t_{j \cdot m-1} = b$ im Text t statt. Schematisch sieht das wie folgt aus:



Da s nur aus a's besteht, ist im ersten Teil der Regel jeweils ein Shift um 1 zulässig: Für $1 = \sigma \leq j$ gilt $a^{m-1-j} = s_{j+1} \cdots s_{m-1} \in \mathcal{R}(s_j \cdots s_{m-1}) = \mathcal{R}(a^{m-1})$; Für $j = 0$ greift der zweite Teil, da dann für $\sigma = 1$ gilt: $a^{m-1} = s_0 \cdots s_{m-2} \in \mathcal{R}(s) = (a^m)$. Somit wird für alle j in der Minimumsbildung der Wert 1 berücksichtigt.

Damit werden für jedes b in t (bis auf das letzte) jeweils $\sum_{i=1}^{m} i = \Theta(m^2)$ Vergleiche ausgeführt (jeweils die innerhalb des oben angegebenen Dreiecks). Vergleiche durchgeführt. Da t $\lfloor n/m \rfloor$ b's enthält ist die Anzahl der Vergleich mindestens

$$\geq \left(\left\lfloor \frac{n}{m} \right\rfloor - 1 \right) m^2 \geq \left(\frac{n}{m} - 2 \right) m^2 \geq \Theta(nm).$$