Ludwig-Maximilians-Universität München
Institut für Informatik
Prof. Dr. Volker Heun

Sommersemester 2020
Semestralklausur
13. August 2020

Algorithmische Bioinformatik: Bäume und Graphen

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. **nicht** mit Bleistift schreiben.
- Der Studentenausweis und ein amtlicher Lichtbildausweis sind bereit zu halten.
- Die reine Bearbeitungszeit beträgt **120 Minuten**.
- Es sind insgesamt 40 Punkte zu erreichen, zum Bestehen sind 17 Punkte nötig.

Viel Erfolg!

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

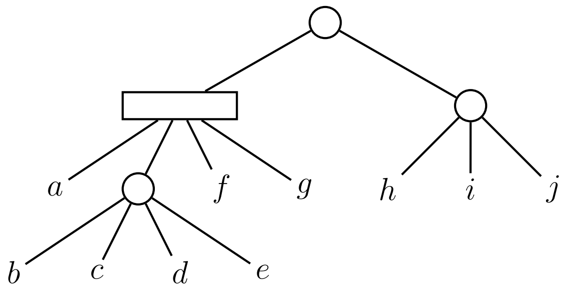
Vorzeitig abgegeben um ....

	Hz	A1	A2	A3	A4	A5	$\sum$
Erstkorrektur							
Nachkorrektur							
Zweitprüfer							

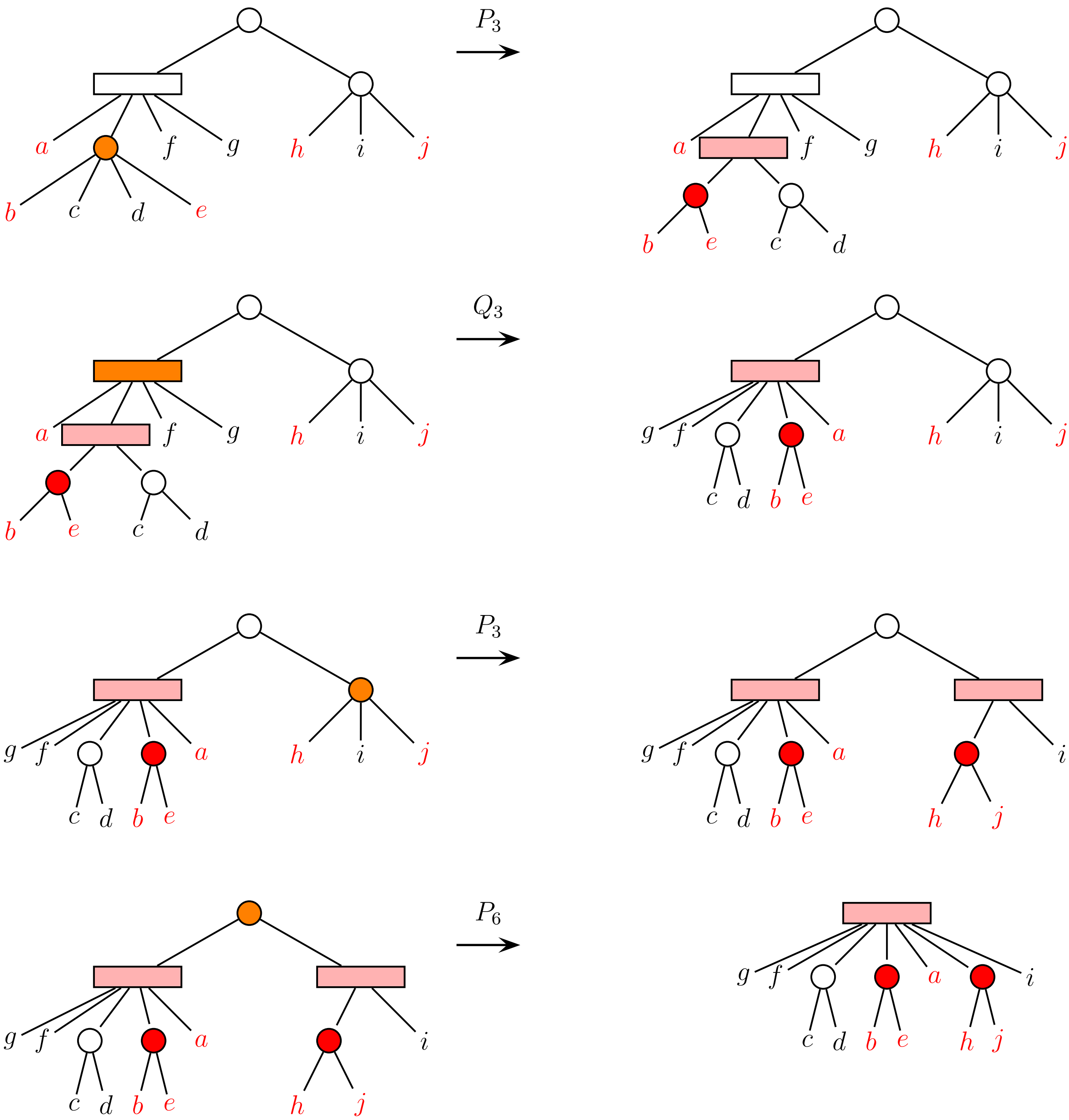
Vorname: ______________ Name: ______________ Matrikelnummer: ____________

Aufgabe 1 (8 Punkte)

Betrachte den rechts abgebildeten PQ-Baum. Modifiziere diesen PQ-Baum gemäß den in der Vorlesung angegebenen Operationen, so dass der neue PQ-Baum zusätzlich noch die Restriktion $\{a, b, e, h, j\}$ erfüllt. Dabei sind alle Zwischenschritte anzugeben und zu dokumentieren.



Lösungsskizze (nicht ausreichend für die volle Punktzahl)



Vorname: ______________ Name: ______________ Matrikelnummer: ____________

Aufgabe 2 (8 Punkte)

Entscheide mit Hilfe des in der Vorlesung angegebenen Algorithmus, ob die unten angegebene Merkmalsmatrix eine perfekte binäre Phylogenie besitzt oder nicht. Falls ja, konstruiere einen phylogenetischen Baum, ansonsten gib eine Begründung an, warum es keinen geben kann.

M	a	b	c	d	e	f
A	0	1	1	1	0	0
B	0	0	0	0	1	0
C	0	0	1	1	0	1
D	1	0	0	0	1	0
E	0	0	1	1	0	0
F	0	0	0	1	0	0

Lösungsskizze (nicht ausreichend für die volle Punktzahl)

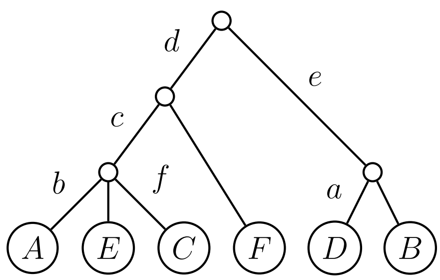
Wir sortieren erst die Spalten absteigend (interpretiert als Binärzahlen):

M'	d	c	b	e	f	a
A	1	1	1	0	0	0
B	0	0	0	1	0	0
C	1	1	0	0	1	0
D	0	0	0	1	0	1
E	1	1	0	0	0	0
F	1	0	0	0	0	0

Erstellen der Strings:

$$\begin{aligned} s_A &:= dcb\$ \\ s_B &:= e\$ \\ s_C &:= dcf\$ \\ s_D &:= ea\$ \\ s_E &:= dc\$ \\ s_F &:= d\$ \end{aligned}$$

Erstellen des kompaktifizierten Tries für $\{s_A, s_B, s_C, s_D, s_E, s_F\}$.



Alle Kantenlabels tauchen tatsächlich nur einmal auf, daher ist dies ein phylogenetischer Baum zur Matrix M.

Vorname: ________________ Name: ________________ Matrikelnummer: ______________

Aufgabe 3 (8 Punkte)

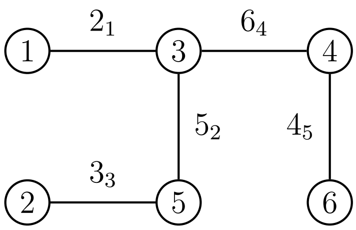
Gegeben seien die beiden folgenden 6×6-Distanzmatrizen D_ℓ und D_h. Entscheide mit Hilfe des in der Vorlesung angegebenen Algorithmus, ob es eine ultrametrische Matrix $D \in [D_\ell, D_h]$ gibt oder nicht. Gib ggf. die resultierende ultrametrische Matrix D an.

D_ℓ	1	2	3	4	5	6
1	0	4	1	3	5	6
2		0	5	5	2	3
3			0	2	4	6
4				0	4	4
5					0	3
6						0

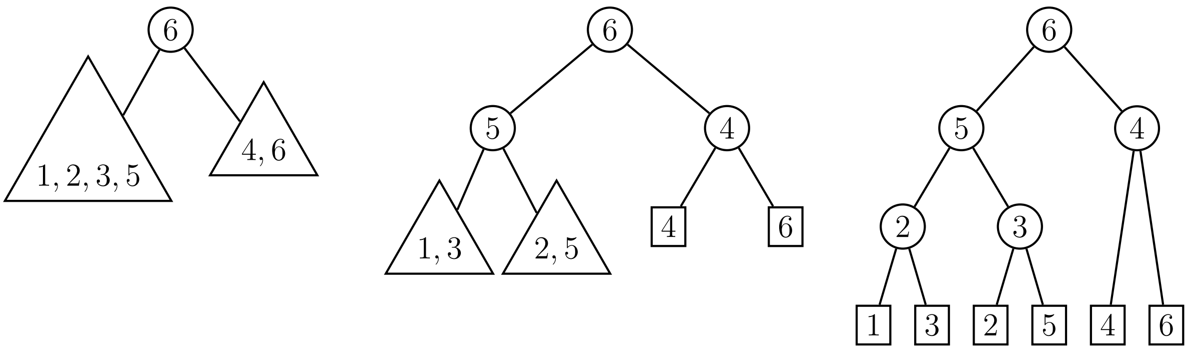
D_h	1	2	3	4	5	6
1	0	6	2	7	9	7
2		0	6	9	3	7
3			0	6	5	7
4				0	7	4
5					0	8
6						0

Lösungsskizze (nicht ausreichend für die volle Punktzahl)

Wir berechnen zuerst den minimalen Spannbaum für D_h nach Prim (die Indizes an den Kantengewichten geben die Reihenfolge an, in denen die Kanten hinzugefügt wurden).



Konstruktion des rekursiven Aufrufbaums (absteigend nach den Kantengewichten im Spannbaum, entspricht der Struktur des ultrametrischen Baums):



Damit ergibt sich die folgende ultrametrische Distanzmatrix D':

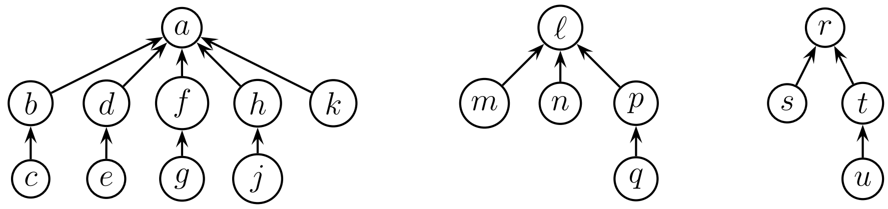
D'	1	2	3	4	5	6
1	0	5	2	6	5	6
2		0	5	6	3	6
3			0	6	5	6
4				0	6	4
5					0	6
6						0

Man prüft leicht nach, dass $D_\ell \leq D'$ gilt.

Vorname: ______________ Name: ______________ Matrikelnummer: ____________

Aufgabe 4 (8 Punkte)

Betrachte den unten angegebenen Zustand einer Union-Find-Datenstruktur:



a) Führe die folgenden Operationen (mit Pfadkompression) einzeln auf dieser aus:

$$\texttt{Union}(\texttt{Find}(p),\texttt{Find}(s)),\quad \texttt{Union}(\texttt{Find}(j),\texttt{Find}(u)).$$

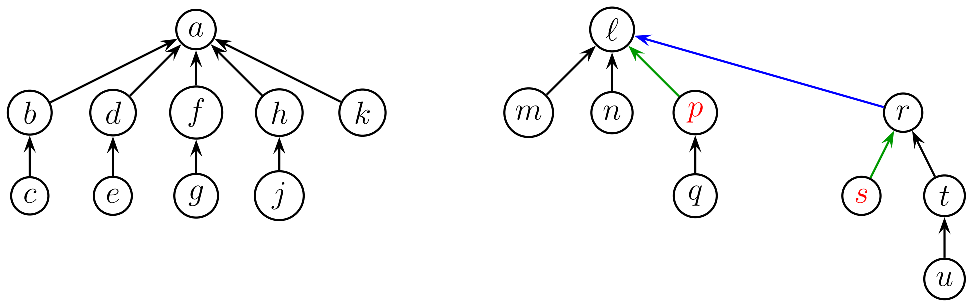
b) Gib den Rang und die Gruppe der folgenden Knoten für den finalen Zustand der Union-Find-Datenstruktur nach Ausführung der in a) angegebenen Befehle für die folgenden Knoten an:

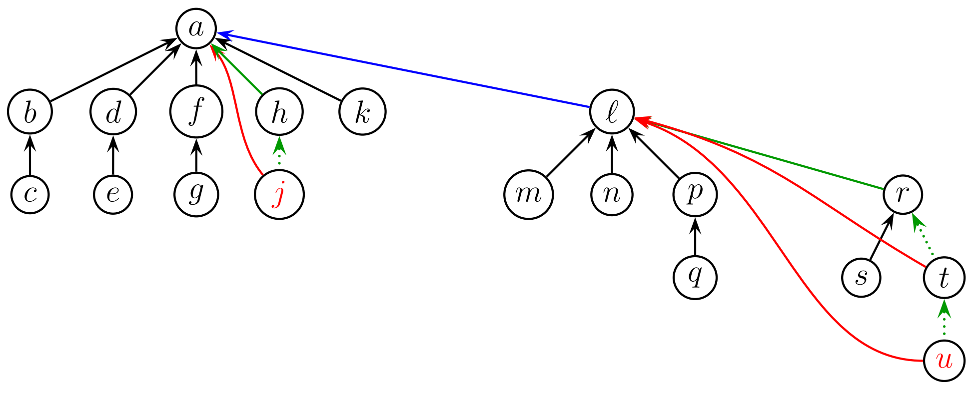
$$a,\quad h,\quad j,\quad \ell,\quad r,\quad t,\quad u.$$

Nimm hierbei an, dass der oben angegebene Zustand der Union-Find-Datenstruktur ohne Pfadkompression entstanden ist.

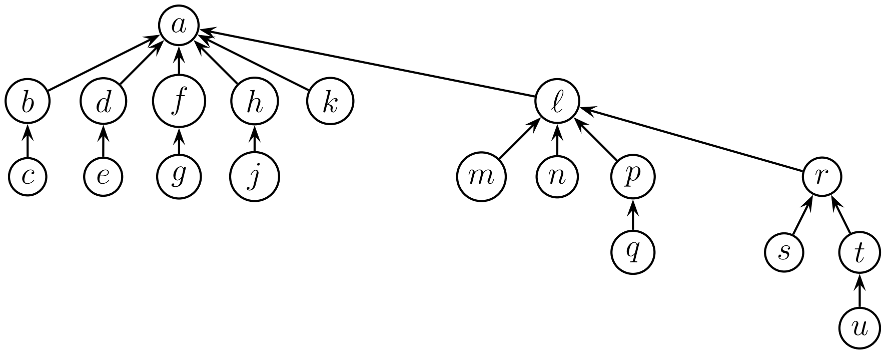
Lösungsskizze (nicht ausreichend für die volle Punktzahl)

a) Grüne Linien sind Kanten, die beim Auffinden der Wurzel eines `Find`-Befehls durchlaufen werden (gepunktet, wenn sie in der Pfadkompression entfernt wurden); rote Linien sind Kanten, die wegen der Pfadkompression eines `Find`-Befehls neu eingefügt werden; blaue Linien sind Kanten, die aufgrund eines `Union`-Befehls eingeführt werden.





b) Für die angegebenen Union-Find-Befehle erhalten wir ohne Pfadkompression den folgenden Baum:



Damit ergibt sich:

Knoten	Rang	Gruppe
a	4	2
h	1	0
j	0	0
ℓ	3	2
r	2	1
t	1	0
u	0	0

Vorname: ______________ Name: ______________ Matrikelnummer: ____________

Aufgabe 5 (8 Punkte)

Für einen ungerichteten Graphen $G = (V, E)$ mit $V = [1 : n]$ und $E \subseteq \binom{V}{2}$ sei die zugehörige *erweiterte Adjazenzmatrix* A wie folgt definiert: Es gilt genau dann $A_{i,j} = 1$, wenn $i = j$ oder $\{i, j\} \in E$.

Zeige, dass G genau dann ein echter Intervall-Graph ist, wenn A die Consecutive Ones Property erfüllt.

Hinweis: Beachte, dass bei der *erweiterten Adjazenzmatrix* eines ungerichteten Graphen auf der Diagonalen 1en statt 0en (wie in der normalen Adjazenzmatrix) stehen.

Lösungsskizze (nicht ausreichend für die volle Punktzahl)

⇒: Sei G ein echter Intervall-Graph und sei $\mathcal{I} = \{I(i) : i \in [1 : n]\}$ seine Intervall-Darstellung. Ohne Beschränkung der Allgemeinheit nehmen wir an, dass $\ell(I(i)) < \ell(I(j))$ für alle $i < j \in [1 : n]$ gilt. Da G ein echter Intervall-Graph ist, folgt dann auch, dass $r(I(i)) < r(I(j))$ für alle $i < j \in [1 : n]$ gilt.

Somit gilt für jedes $i \in [1 : n]$, dass die Menge seiner Nachbarn ein Intervall aus $[1 : n]$ bilden. Mit dieser Nummerierung besitzt A also die Consecutive Ones Property.

⇐: Seien ohne Beschränkung der Allgemeinheit die Knoten so nummeriert, dass A die Consecutive Ones Property besitzt (da A symmetrisch ist, ist es egal ob für Spalten oder Zeilen). Wir definieren nun für jeden Knoten $i \in [1 : n]$ ein Intervall $I_i := [i, r_i]$ mittels

$$r_i \;:=\; \max\{j \in [1 : n] \,:\, A_{i,j} = 1\}\,.$$

Offensichtlich ist $r_i \geq i$, da $A_{i,i} = 1$

Wir behaupten jetzt, dass genau dann $A_{i,j} = 1$, wenn $I_i \cap I_j \neq \emptyset$ gilt. Ohne Beschränkung der Allgemeinheit sei $i < j$ (für $i = j$ ist es trivial, für $i > j$ symmetrisch). Ist $A_{i,j} = 1$, dann ist $i < j \leq r_i$ und somit gilt $I_i \cap I_j \supseteq \{j\} \neq \emptyset$ (da $j \in [i : r_i] = I(i)$ und $j \in I(j)$). Ist andererseits $I_i \cap I_j \neq \emptyset$, dann ist $j \leq r_i$ (sowie $i < j$ nach Annahme) und somit ist $j \in [i : r_i]$ und $A_{i,j} = 1$ (wegen der Consecutive Ones Property von A und $A_{i,i} = 1 = A_{i,r_i}$). Somit haben wir eine Intervall-Darstellung für G gefunden.

Weiter halten wir fest, dass die Folge $(r_1, \ldots, r_n)$ eine monoton wachsende Folge ist. Für einen Widerspruchsbeweis sei $r_i > r_j$ für $i < j$. Dann existiert ein $k \in [j + 1 : n]$ mit $A_{i,k} = 1$ und $A_{j,k} = 0$. Da auf der Hauptdiagonalen 1en stehen, muss $k > j$ gelten. Da A als Adjazenzmatrix symmetrisch ist, gilt $A_{k,i} = 1$, $A_{k,j} = 0$ und $A_{k,k} = 1$ mit $i < j < k$, was ein Widerspruch zur Consecutive Ones Property ist.

Also bilden die Anfangspunkte (trivialerweise) und die Endpunkte jeweils eine monoton wachsende Folge. Für eine echte Intervall-Darstellung müssen diese streng monoton sein (die Anfangspukte sind es nach Definition). Somit kann die Intervall-Darstellung $\mathcal{I}$ ggf. durch leichtes Vergrößern des rechtes Randes r_i zu einer echten Intervall-Darstellung modifiziert werden (bspw. mittels $r'_i = r_i + \frac{i-1}{n}$).