

Prüfung (Lösung)

Datenstrukturen und Algorithmen (D-MATH RW)

Felix Friedrich, Julia Chatain, Aritra Dhar, Departement Informatik

ETH Zürich, 06.2021.

Name, Vorname: .....

Legi-Nummer: .....

Ich bestätige mit meiner Unterschrift, dass ich diese Prüfung unter regulären Bedingungen ablegen konnte und dass ich die allgemeinen Richtlinien gelesen und verstanden habe.

*I confirm with my signature that I was able to take this exam under regular conditions and that I have read and understood the general guidelines.*

Unterschrift:

Allgemeine Richtlinien:

General guidelines:

1. Dauer der Prüfung: 150 Minuten.
2. Erlaubte Unterlagen: Wörterbuch (für von Ihnen gesprochene Sprachen). 4 A4 Seiten handgeschrieben oder  $\geq 11$ pt Schriftgrösse.
3. Benützen Sie einen Kugelschreiber (blau oder schwarz) oder keinen Bleistift. Bitte schreiben Sie leserlich. Nur Ihre Resultate werden bewertet.
4. Lösungen sind direkt auf das Aufgabenblatt in den vorgesehenen Boxen zu schreiben (und direkt daneben, falls mehr Platz benötigt wird). Ungültige Lösungen sind durchgestrichen! Korrekturen bei Multiple-Choice-Aufgaben bitte unmissverständlich ankreuzen.
5. Es gibt keine Negativpunkte für falsche Antworten.
6. Störungen durch irgendjemanden sind nicht toleriert. Sie bitte sofort der Aufsichtsperson melden.
7. Wir sammeln die Prüfungen am Ende der Prüfung. Sie müssen unbedingt Ihre Prüfung abgeben und lassen Sie Ihre Unterlagen bei uns liegen. Dasselbe gilt, wenn Sie frühzeitig fertig sind. Bitte melden Sie sich lautlos, und Ihre Prüfung rechtzeitig abgeben. Vorzeitige Abgaben sind nur bis 15 Minuten vor Prüfungsende möglich.
8. Wenn Sie zur Toilette müssen, melden Sie dies einer Aufsichtsperson durch Handzeichen.
9. Wir beantworten keine inhaltlichen Fragen während der Prüfung. Kommentare zur Aufgabe schreiben Sie bitte auf das Aufgabenblatt.

1. Exam duration: 150 minutes.
2. Allowed aids: dictionary (for languages spoken by you). 4 A4 pages handwritten or  $\geq 11$ pt.
3. Do not use a pencil. Please write legibly. Only your solutions that we can read will be graded.
4. Solutions are to be written directly onto the exam sheets in the provided boxes (and directly below, if needed). Invalid solutions need to be crossed out. Provide corrections to answers to multiple-choice questions without any ambiguity!
5. There are no negative points for wrong answers.
6. Disturbance by anyone or anything, let the supervisor of the exam know immediately.
7. We collect the exams at the end. Important: You must ensure that your exam has been collected by an assistant. Do not take any exam with you, and do not leave your exam behind on your desk. The same applies when you want to finish early: Please contact us silently, and we will collect the exam. Handing in your exam ahead of time is only possible until 15 minutes before the exam ends.
8. If you need to go to the toilet, raise your hand and wait for a supervisor.
9. We will not answer any content-related questions during the exam. Please write comments referring to the tasks on the exam sheets.

**Beispielprüfung als Richtwert für den Stil der Prüfung im August. Nicht zählend. / Example exam as a rough guideline for the style of the exam in August. Does not count.**

Question:	1	2	3	4	5	6	7	Total
Points:	30	16	14	10	18	16	16	120
Score:								

Verwenden Sie die Notation, Algorithmen und Datenstrukturen aus der Vorlesung. Falls Sie eine andere Herangehensweise wählen, erklären Sie Ihre Antworten in nachvollziehbarer Weise!

*Use notation, algorithms, and data structures from the course. If you use a different approach, explain your answers in a comprehensible way!*

## Aufgabe 1: Verschiedenes (30P)

- /8P (a) Kreuzen Sie an, ob die folgenden Aussagen wahr oder falsch sind.

*Mark if the following statements are true or false.*

Wenn  $f(n) \in \Theta(g(n))$  und  $g(n) \in \Theta(h(n))$ , dann  $h(n) \in \Theta(f(n))$ .  
/ *If  $f(n) \in \Theta(g(n))$  and  $g(n) \in \Theta(h(n))$ , then  $h(n) \in \Theta(f(n))$ .*

Wahr / True

Falsch / False

Wenn  $f(n) \in O(g(n))$  und  $g(n) \in O(h(n))$ , dann  $h(n) \in \Omega(f(n))$ .  
/ *If  $f(n) \in O(g(n))$  and  $g(n) \in O(h(n))$ , then  $h(n) \in \Omega(f(n))$ .*

Wahr / True

Falsch / False

$2^{n+1} = O(2^n)$

Wahr / True

Falsch / False

$2^{2n+1} = O(2^n)$

Wahr / True

Falsch / False

Für einen ungerichteten Graphen mit negativen Kantengewichten findet der Algorithmus von Dijkstra einen kürzesten Pfad zwischen zwei Knoten immer korrekt. / *Given an undirected graph with negative edge weights, Dijkstra's algorithm always finds a shortest path between two nodes correctly.*

Wahr / True

Falsch / False

Bei der Technik des Memoisierens im Zusammenhang mit dynamischer Programmierung wird die asymptotische Laufzeit zu Lasten eines höheren Speicherplatzverbrauches verringert. / *The memoization technique in dynamic programming decreases the asymptotic runtime of an algorithm by increasing the memory consumption.*

Wahr / True

Falsch / False

Seien  $W(n)$  und  $A(n)$  die Laufzeit eines Algorithmus zur Problemgröße  $n$  im schlechtesten Fall bzw. im Durchschnitt. Dann gilt  $A(n) \in \Theta(W(n))$  / *Let  $W(n)$  and  $A(n)$  denote the worst-case and average-case running time of an algorithm executed on an input of size  $n$  respectively. Then,  $A(n) \in \Theta(W(n))$*

Wahr / True

Falsch / False

Randomisiertes Quicksort ist eine Variante des Quicksortalgorithmus, bei dem der Pivot zufällig gewählt wird. Die asymptotische Laufzeit im schlechtesten Falle für das Sortieren von  $n$  Zahlen beträgt damit  $O(n \log n)$  / *Randomized quicksort is a variant of the quicksort algorithm, where the pivot is chosen randomly. The worst-case asymptotic runtime of sorting  $n$  numbers using randomized quicksort is  $O(n \log n)$ .*

Wahr / *True*  
 Falsch / *False*

(b) Die Nebenreihenfolgeausgabe eines binären Suchbaumes sei

*Let the post-order traversal output of a binary search tree be*

/3P

1, 2, 6, 4, 12, 13, 11, 7

Finden Sie die Hauptreihenfolge.

*Find the pre-order traversal.*

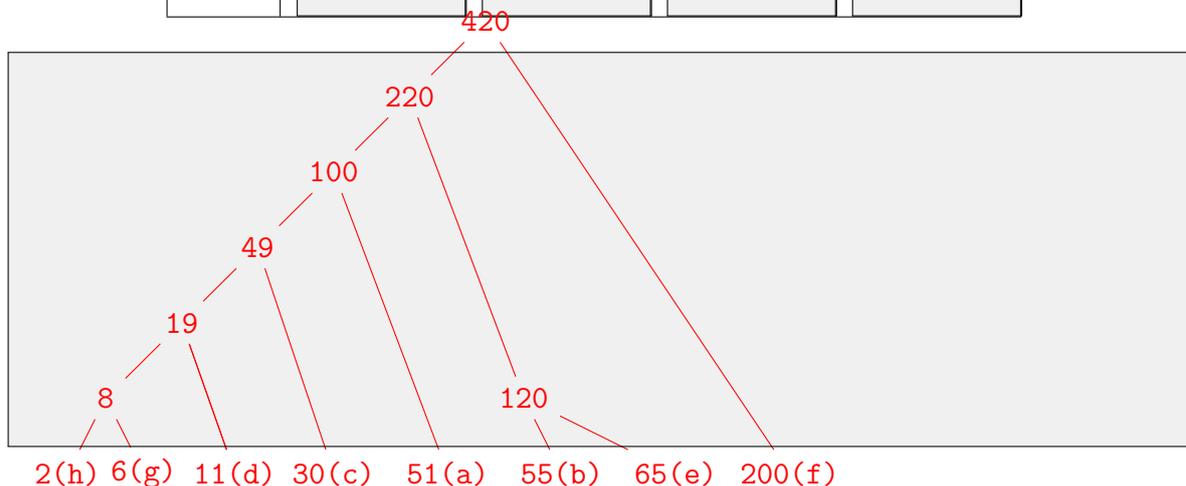
Hauptreihenfolge / *pre-order traversal*: 7, 4, 2, 1, 6, 11, 13, 12

(c) Gegeben sind acht Buchstaben mit relativer Häufigkeit (Anzahl Zugriffe) wie folgt. Erstellen Sie mit Hilfe des Huffman-Algorithmus einen optimalen Codierungsbaum. Tragen Sie den resultierenden Code in der Tabelle ein.

*Eight characters (keys) with relative frequency (number accesses) are given as follows. Using the Huffman algorithm provide an optimal code tree. Enter the corresponding code into the table.*

/3P

char	a	b	c	d
freq	51	55	30	11
Code	001	010	0001	00001
char	e	f	g	h
freq	65	200	6	2
code	011	1	000001	000000



- /2P (d) Eine Hashtabelle mit 10 Einträgen verwendet offene Adressierung mit der Hash-Funktion  $h(k) = k \bmod 10$ , mit linearer Sondierung (Sondierung geht nach rechts). Nachdem sechs Werte in die initial leere Hashtabelle eingefügt wurden, sieht die Hashtabelle wie folgt aus.

0	1	2	3	4	5	6	7	8	9
8	78		53	13				88	49

Welche der folgenden Möglichkeiten bezeichnen/bezeichnet jeweils eine Reihenfolge, in der die Schlüssel in die Hashtabelle eingefüllt werden konnten?

- (A) 88, 53, 49, 13, 8, 78  
 (B) 88, 53, 13, 49, 78, 8  
 (C) 88, 49, 8, 78, 13, 53  
 (D) 53, 49, 88, 13, 8, 78

*A hash table of length 10 uses open addressing with hash function  $h(k) = k \bmod 10$ , and linear probing (probing goes to the right). After inserting 6 values into an empty hash table, the table is as shown below.*

*Which of the following choice(s) give possible order(s) in which the key values could have been inserted in the table?*

(A, D)

- /3P (e) Sei  $T$  ein binärer Suchbaum mit 15 Knoten. Bestimmen Sie die minimal und maximal mögliche Höhe von  $T$  (wählen Sie die zugehörige Option). Geben Sie die Fälle an, in denen diese maximalen und minimalen Höhen auftreten. (Ein Baum mit nur einem Knoten habe die Höhe 0).

- (A) 4, 15  
 (B) 3, 14  
 (C) 4, 14  
 (D) 3, 15

*Let  $T$  be a binary search tree with 15 nodes. Specify the option that provides minimum and maximum possible heights of  $T$ . Specify cases when these maximum and minimum heights occur. (A tree with a single node has height 0).*

B. 3, 14

The minimum when it is a full complete tree :  $\log_2(15 + 1) - 1 = 3$   
 The maximum height of a binary search tree will be when the tree is fully skewed, max height is  $n - 1 = 14$

- (f) Sei  $G = (V, E)$  ein zusammenhängender, ungerichteter kantengewichteter Graph. Die Gewichte der Kanten in  $E$  seien **positiv** und paarweise **verschieden**. Betrachten Sie die folgenden Aussagen:

- (I) Der minimale Spannbaum von  $G$  ist immer eindeutig.  
 (II) Der kürzeste Pfad zwischen zwei beliebigen Knoten von  $G$  ist immer eindeutig.

Was stimmt?

- (A) Nur I / *I only*  
 (B) Nur II / *II only*  
 (C) I und II / *Both I and II*  
 (D) Weder I noch II / *Neither I nor II*

A. I only

*Let  $G = (V, E)$  be any connected undirected edge-weighted graph. The weights of the edges in  $E$  are **positive** and **distinct**. Consider the following statements:*

- (I) The minimum spanning Tree of  $G$  is always unique.  
 (II) The shortest path between any two vertices of  $G$  is always unique.*

*What is true?*

/2P

- (g) Gegeben sei ein ungerichteter Zufallsgraph mit acht Knoten. Die Wahrscheinlichkeit, dass zwischen einem Eckpunktpaar eine Kante liegt, betrage  $1/2$ . Was ist die erwartete Anzahl ungeordneter Kreise der Länge **drei**? Geben Sie eine kurze Begründung.

- (A)  $1/8$   
 (B) 1  
 (C) 7  
 (D) 8

A cycle of length three can be formed with three vertices. There can be a total of  $8C3$  ways to pick three vertices from 8. The probability that there is an edge between two vertices is  $1/2$ . So expected number of unordered cycles of length 3 =  $\binom{8}{3} * (1/2)^3 = 7$

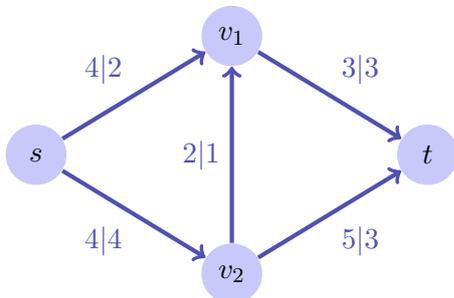
*Consider an undirected random graph of eight vertices. The probability that there is an edge between a pair of vertices is  $1/2$ . What is the expected number of unordered cycles of length **three**? Provide a short justification.*

/3P

/6P

(h) Gegeben ist das folgende Flussnetzwerk  $G$  mit Quelle  $s$  und Senke  $t$ . Die einzelnen Kapazitäten  $c_i$  und Flüsse  $\phi_i$  sind an den Kanten angegeben als  $c_i|\phi_i$ . Geben Sie den Wert des Flusses  $f$  an.

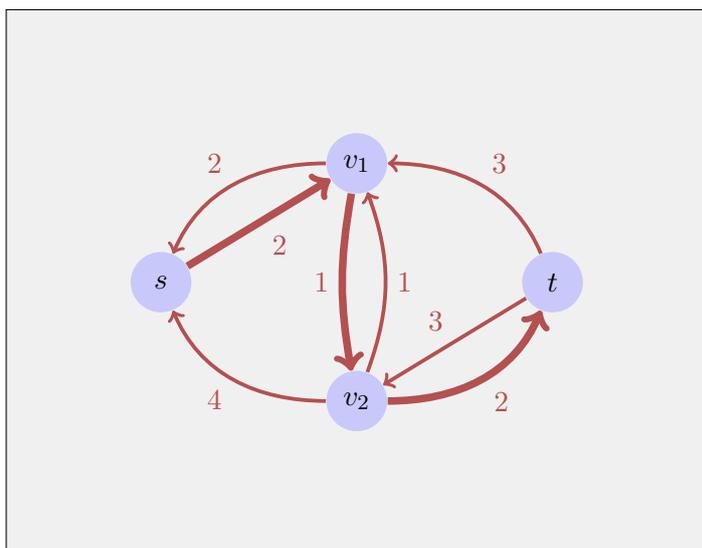
*Provided in the following is a flow network  $G$  with source  $s$  and sink  $t$ . Capacities  $c_i$  and flows  $\phi_i$  are provided at the edges as  $c_i|\phi_i$ . Provide the value of the flow  $f$ .*



$|f| =$

Zeichnen Sie nun das Restnetzwerk  $G_f$  zu obigem Fluss und markieren Sie darin einen Erweiterungspfad  $p$ . Geben Sie den Wert  $c_f(p)$  der Restkapazität des Erweiterungspfad  $p$  im Restnetzwerk  $G_f$  an.

*Draw the residual network  $G_f$  to the flow above and mark an augmenting path  $p$ . Provide the rest capacity  $c_f(p)$  of the path  $p$  in the rest network  $G_f$ .*



$|c_f(p)| =$

Woran sehen Sie, dass Sie den maximalen Fluss gefunden haben ?

*How do you see that you have found the maximum flow?*

Das Restnetzwerk hat keinen Erweiterungspfad mehr.  
 Alternativ: Identifiziere Schnitt mit  $|c(S,T)| = |f|$

## Aufgabe 2: Asymptotik (16P)

- (a) Geben Sie für die untenstehenden Funktionen eine Reihenfolge an, so dass folgendes gilt: Wenn eine Funktion  $f$  links von einer Funktion  $g$  steht, dann gilt  $f \in \mathcal{O}(g)$ .  
 Beispiel: die drei Funktionen  $n^3$ ,  $n^5$  und  $n^7$  sind bereits in der entsprechenden Reihenfolge, da  $n^3 \in \mathcal{O}(n^5)$  und  $n^5 \in \mathcal{O}(n^7)$ .

*Provide an order for the following functions such that the following holds: If a function  $f$  is left of a function  $g$  then it holds that  $f \in \mathcal{O}(g)$ .  
 Example: the functions  $n^3$ ,  $n^5$  and  $n^7$  are already in the respective order because  $n^3 \in \mathcal{O}(n^5)$  and  $n^5 \in \mathcal{O}(n^7)$ .*

/3P

$n^{2.5}$ ,  $\sqrt{2n}$ ,  $n + 10$ ,  $n^2 \log_3(n)$ ,  $10^n$ ,  $n^n$ ,  $100^n$ ,  $n^3 \log_2(n)$

$\sqrt{2n}$	$n + 10$	$n^2 \log_3(n)$	$n^{2.5}$	$n^3 \log_2(n)$	$10^n$	$100^n$	$n^n$
-------------	----------	-----------------	-----------	-----------------	--------	---------	-------

- (b) Gegeben sei die folgende Rekursionsgleichung:

*Consider the following recursion equation:*

/4P

$$T(n) = \begin{cases} T(\frac{n}{2}) + n, & n > 1 \\ 1 & n = 1 \end{cases}$$

Leiten Sie eine geschlossene (nicht rekursive), einfache Formel für  $T(n)$  her. Nehmen Sie an, dass es ein  $k \in \mathbb{N}$  gibt mit  $2^k = n$ .  
 Hinweis:  
 Für  $q \neq 1$  gilt  $\sum_{i=0}^k q^i = \frac{q^{k+1}-1}{q-1}$ .

*Derive a closed (non-recursive), simple formula for  $T(n)$ . Assume that there is some  $k \in \mathbb{N}$  for which  $2^k = n$ .  
 Hint:  
 For  $q \neq 1$  it holds that  $\sum_{i=0}^k q^i = \frac{1-q^{k+1}}{1-q}$ .*

$$\begin{aligned}
 T(n) &= T(n/2) + n \\
 &= T(n/4) + n/2 + n \\
 &= \dots \\
 &= T(n/2^k) + n/2^{k-1} + \dots + n \\
 &= 1 + n/2^{k-1} + \dots + n \\
 &= n/2^k + n/2^{k-1} + \dots + n \\
 &= n(1 + 1/2 + \dots + 1/2^k) \\
 &= n\left(\frac{1 - (1/2)^{k+1}}{1/2}\right) \\
 &= 2n\left(1 - \frac{1}{2n}\right) \\
 &= 2n - 1
 \end{aligned}$$

Geben Sie die asymptotische Laufzeit von  $f(n)$  in Abhängigkeit von  $n$  möglichst knapp und präzise an.

*Provide the asymptotic running time of  $f(n)$  as a function of  $n$  as tight and precise as possible.*

/2P (c)

```
int f(int n){
    if(n==1){
        return 1;
    } else {
        return f(n/2) + 1
    }
}
```

Asymptotische Laufzeit von  $f$  / *Asymptotic Running time of  $f$*

$\Theta(\log(n))$

/2P (d)

```
int f(int n){
    if(n<=1){
        return 1;
    } else {
        int x = f(n-1)*f(n-1);
        return x;
    }
}
```

Asymptotische Laufzeit von  $f$  / *Asymptotic Running time of  $f$*

$\Theta(2^n)$

/2P (e)

```
int f(int n){
    int q = 0;
    for (int i = 1; i<n; ++i){
        for (int j = n; j > 1; j = j/2)
            ++q;
    }
    return q;
}
```

Asymptotische Laufzeit von  $f$  / *Asymptotic Running time of  $f$*

$\Theta(n \cdot \log n)$

- (f)  $q(m)$  sei eine Funktion mit asymptotischer Laufzeit  $\Theta(m)$ , and  $A$  sei ein Array von Ganzzahlen mit  $n$  Elementen aus  $\{0, 1\}$ .

*$q(m)$  is a function whose asymptotic running time is  $\Theta(m)$ , and  $A$  is an integer array of  $n$  elements from  $\{0, 1\}$ .*

/3P

---

```
int f(int n, int[] A){
    counter = 0;
    for (i = 1; i <= n; i++) {
        if (A[i] == 1){
            counter++;
        } else {
            q(counter);
            counter = 0;
        }
    }
    return counter;
}
```

---

Asymptotische Laufzeit von  $f$  / *Asymptotic*  
*Running time of  $f$*

$\Theta(n)$

### Aufgabe 3: Spannbäume (14P)

Gegeben sei ein gewichteter, zusammenhängender, ungerichteter Graph  $G = (V, E, g)$ .

*Let a weighted, connected, undirected graph  $G = (V, E, g)$  be given.*

- /4P (a) Sei  $G$  strikt positiv gewichtet. Bei der Ausführung des Algorithmus von Dijkstra auf  $G$  zur Bestimmung aller kürzesten Wege von einem Startknoten  $s$  aus werden sukzessive Kanten hinzugenommen. Sei  $E'$  die Menge dieser Kanten. Behauptung: Der entstehende Graph  $G' = (V, E')$  ist ein Spannbaum. Zeigen oder widerlegen sie das.

*Let  $G$  be equipped with strictly positive weights. When executing Dijkstra's algorithm on  $G$  to determine all the shortest paths from a starting node  $s$ , successive edges are added. Let  $E'$  be the set of these edges. Claim: the resulting graph is  $G' = (V, E')$  is a spanning tree. Show or refute that.*

During Dijkstra edges are added to the graph that determine a shortest path from  $s$  to any other point. Because  $G$  is connected,  $G'$  will also be connected: any two points  $u$  and  $v$  are connected because  $u$  and  $s$  and  $v$  and  $s$  are connected. The graph does not contain cycles by the construction of the algorithm (each point is added only once to  $E'$ ).

- /4P (b) Ein minimaler Spanngraph zu  $G$  ist ein zusammenhängender Graph  $G'$  minimalen Gewichts. Die Kantengewichte von  $G$  seien immer noch strikt positiv. Zeigen oder widerlegen Sie: Der minimale Spanngraph  $G'$  zu  $G$  ist ein minimaler Spannbaum. Geben Sie einen Algorithmus zur Berechnung von  $G'$  mit Laufzeit an.

*A minimum spanning graph for  $G$  is a connected graph  $G'$  of minimal weight. The edge weights of  $G$  are assumed to be still strictly positive. Show or refute: The minimum spanning graph  $G'$  for  $G$  is a minimum spanning tree. Give an algorithm for the calculation of  $G'$  with running time.*

All edge weights are strictly positive. A minimum spanning graph is a minimum spanning tree if it has no cycles. A minimum spanning graph cannot have any cycles because it would remain a spanning graph if we removed an edge from each remaining cycle, which would decrease the total weight of the graph. Therefore any minimum spanning graph is a minimum spanning tree if all edge weights are strictly positive. Algorithm Kruskal or Prim-Dijkstra with running times  $O(m \log n)$

- (c) Wir nehmen nun an, dass  $G$  nicht mehr strikt positiv gewichtet ist. Wie berechnen Sie nun einen minimalen Spannbaum?

*We now assume that  $G$  is no longer strictly positive weighted. How do you calculate a minimal spanning tree?*

/2P

Same algorithm as before: Kruskal or Prim-Dijkstra also work on a graph with negative edge weights.

- (d) Wir nehmen weiterhin an, dass  $G$  nicht mehr strikt positiv gewichtet ist. Wie berechnen Sie einen minimalen Spanngraph? Beschreiben Sie Ihren Algorithmus und geben Sie die asymptotische Laufzeit an.

*We still assume that  $G$  is not strictly positively weighted. How do you calculate a minimal spanning graph? Describe your algorithm and provide its asymptotic running time.*

/4P

Start with the Graph  $G' = (G, \{e \in E : c(e) \leq 0\}, c)$ . Now apply the steps of Kruskal algorithm with the remaining edges (of positive weight). Running time is the same:  $O(m \log n)$  for sorting the edges.

## Aufgabe 4: Parallele Programmierung (10P)

- /2P (a) Die Analyse eines Programms hat eine Beschleunigung von 2 bei Ausführung auf 9 Prozessorkernen ergeben. Wie gross ist der serielle Anteil nach dem Gesetz von Gustafson?

*The analysis of a program has shown a speed-up of 2 when running on 9 processor cores. What is the serial fraction according to Gustafson's law?*

$$\begin{aligned}S_p &= p - f \cdot (p - 1) \\2 &= 9 - f \cdot 8 \\7 &= f \cdot 8 \\f &= 7/8 = 0.875\end{aligned}$$

- /2P (b) Sie messen Ihr Programm mit einer sehr großen Anzahl von Prozessorkernen. Die Messungen legen nahe, dass die Beschleunigung (bei Verwendung beliebig vieler Prozessorkerne) nach oben durch  $S_\infty = 2.5$  begrenzt wird. Wie gross ist die beste obere Schranke für die Beschleunigung bei Verwendung von 6 Prozessorkernen?

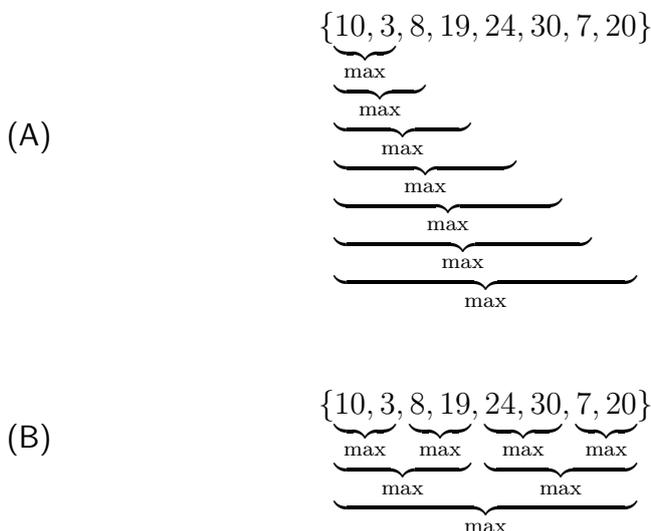
*You make a measurement of your program using a very large number of processor cores. The measurements suggest that the speed-up (using arbitrarily number of processor cores) is bounded from above by  $S_\infty = 2.5$ . What is the best possible upper bound on the speed-up using 6 cores, assuming that Amdahl's law holds for your problem?*

$$\begin{aligned}S_p &\leq \frac{1}{f + \frac{1-f}{p}} \\S_\infty &= \frac{1}{f} = 5/2 \\f &= 2/5\end{aligned}$$

$$\begin{aligned}S_p &\leq \frac{1}{f + \frac{1-f}{p}} \\S_6 &\leq \frac{1}{0.4 + \frac{0.6}{6}} \\S_6 &\leq 2\end{aligned}$$

Sie führen die Berechnung des Maximums eines Arrays der Länge 8 auf die Berechnung des Maximums zwischen zwei Zahlen zurück und implementieren es auf die folgenden zwei Arten.

*You reduce the calculation of the maximum of an array of length 8 to the calculation of the maximum between two numbers and implement it in the following two ways.*



- (c) Bestimmen Sie die Länge des kritischen Pfades für die beiden Ausführungen (A) und (B). Die Bestimmung des Maximums eines Paares von Zahlen koste einen Zeitschritt.

*Determine the length of the critical path for the two executions (A) and (B). The determination of the maximum of two pairs has a runtime cost of 1.*

/2P

$$T_{\infty}(A) = 7$$

$$T_{\infty}(B) = 3$$

- (d) Bestimmen Sie die Länge des kritischen Pfades für obige Fälle bei Anwendung der Verfahren auf ein Array der Länge  $n$ .

*Determine the length of the critical path when the methods from above are applied to an array of length  $n$ .*

/2P

$$T_{\infty}(A) = n - 1$$

$$T_{\infty}(B) = \lceil \log_2 n \rceil$$

- (e) Was ist nach dem Greedy-Scheduling Theorem die Parallelität, also der maximal erreichbare Speedup, wenn für ein Array der Länge  $n$  für die beiden Methoden A und B oben, ein Greedy-Scheduler auf beliebig vielen Prozessoren verwendet wird?

*What is, according to the greedy scheduling theorem the parallelism, i.e. the best possible speed-up when methods A and B from above are applied to the problem of size  $n$  using an arbitrarily high number of processors?*

/2P

$$S_{\infty}(A, n) = \frac{T_1}{T_{\infty}} = \frac{n-1}{n-1} = 1$$

$$S_{\infty}(B, n) = \frac{T_1}{T_{\infty}} = \frac{n-1}{\log n} = \Theta(n / \log n)$$

# Programmierfragen

Moodle-Link: <https://moodle-app2.let.ethz.ch/mod/quiz/view.php?id=601116>

Moodle-Passwort wird normalerweise zu Beginn der Prüfung bekannt gegeben. Hier lautet es Informatik / *Moodle password will normally be announced at the beginning of the exam. Here it is Informatik*

/18P

## Aufgabe 5: Dynamische Programmierung (18P)

Diese Aufgabe zum Thema Dynamische Programmierung soll am Computer gelöst werden. Sie können sich die Zeit frei einteilen. Wir **empfehlen** aber, dass Sie **nicht mehr als 30 Minuten** für diese Aufgabe aufwenden.

Lösen Sie diese Aufgabe in der Online-Umgebung (Code Expert via Moodle).

*This part on dynamic programming needs to be solved at the computer. You are free in how you divide the time. However, we recommend to spend not more than 30 minutes on that problem.*

*Solve this task in the online environment (Code Expert via Moodle).*

/16P

## Aufgabe 6: Snakes And Ladders (16P)

Diese Aufgabe zum Thema Graphen / Breitensuche soll am Computer gelöst werden. Sie können sich die Zeit frei einteilen. Wir **empfehlen** aber, dass Sie **nicht mehr als 25 Minuten** für diese Aufgabe aufwenden.

Lösen Sie diese Aufgabe in der Online-Umgebung (Code Expert via Moodle).

*This part on graphs and breadth first search needs to be solved at the computer. You are free in how you divide the time. However, we recommend to spend not more than 25 minutes on this problem.*

*Solve this task in the online environment (Code Expert via Moodle).*

/16P

## Aufgabe 7: Barriere (16P)

Diese Aufgabe zum Thema Parallele Programmierung soll am Computer gelöst werden. Sie können sich die Zeit frei einteilen. Wir **empfehlen** aber, dass Sie **nicht mehr als 25 Minuten** für diese Aufgabe aufwenden.

Lösen Sie diese Aufgabe in der Online-Umgebung (Code Expert via Moodle).

*This following part on parallel programming needs to be solved at the computer. You are free in how you divide the time. However, we recommend to spend not more than 25 minutes on that problem.*

*Solve this task in the online environment (Code Expert via Moodle).*