ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

Spring Semester 2022 Prof. R. Wattenhofer

Computational Thinking

Friday, September 2, 2022, 13:30-15:30

Do not turn over until instructed to do so

This examination lasts for 120 minutes and comprises 120 points. There is one block of questions for each lecture chapter.

You may answer in German, English, or combine German and English.

Unless explicitly stated, you do **not** have to justify your answers. Writing down your thoughts (math, text, or annotated sketches), however, might help with the understanding of your approach. This may then result in points being awarded even if your answer is not correct. Please write legibly. Unreadable answers will not be graded.

Some questions will ask you to fill in answers in a template. If you decide to start over you will find fill-in replacements at the end of the examination booklet.

Please write your name and student number on every additional sheet. Please write your name and student number in the following fields on this cover sheet.

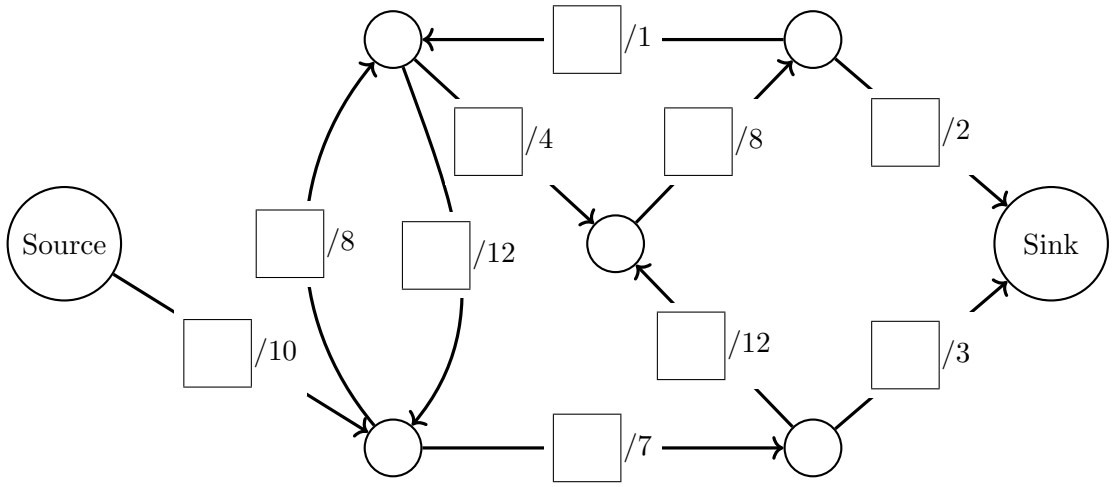
Family Name	**First Name**	**Student Number**

Task	**Achieved Points**	**Maximum Points**
1 - Algorithms		19
2 - Complexity		20
3 - Cryptography		15
4 - Data and Storage		18
5 - Machine Learning		18
6 - Neural Networks		14
7 - Computability		16
Total		**120**

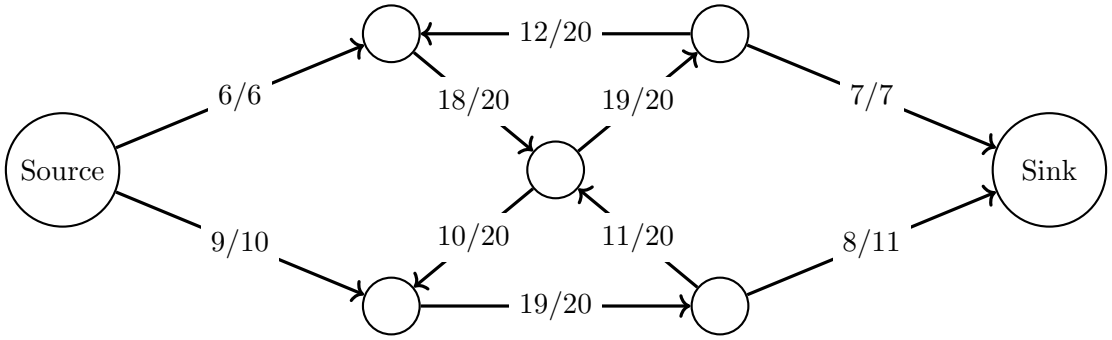
1 Algorithms (19 points)

Flows

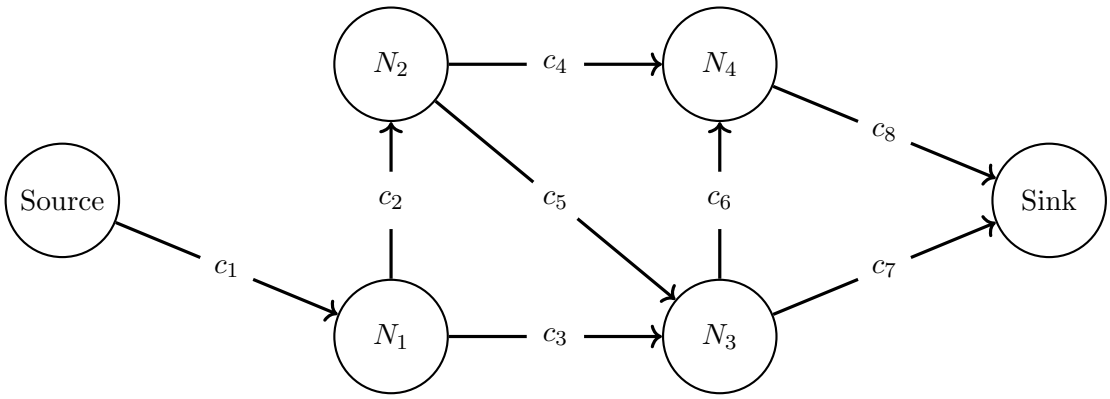
a) `[5 points]` Given is the following graph with edge capacities. For each edge, fill in the box with the flow along that edge so that the flow from source to sink is maximized.



b) `[6 points]` Given is the following graph with edge-wise flow/capacity. Adjust the flow $f(e)$ of the edges so as to minimize $\Sigma_{e \in E} f(e)$ while keeping the total flow from source to sink constant.



c) [6 points] Given is the following graph with corresponding general edge capacities c_i for $1 \leq i \leq 8$. Fill in the blanks of the linear program (LP) so that the solution of the LP is the value of the max flow. (Please fill in the boxes below. You are allowed to use equalities.)



maximize

subject to:

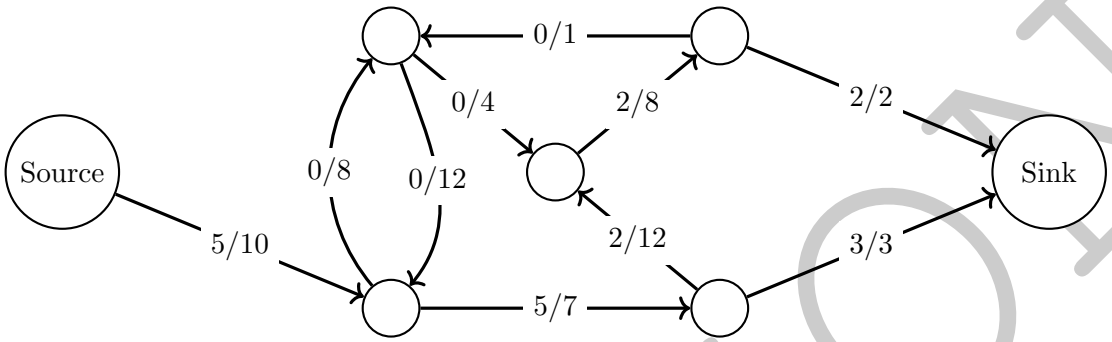
equations in this box hold **for all** i

d) [2 points] To avoid an unbalanced flow, you want to constrain the flow on edge 7 and edge 8 to differ by at most 2. Write one or more additional constraints to ensure this property in the LP.

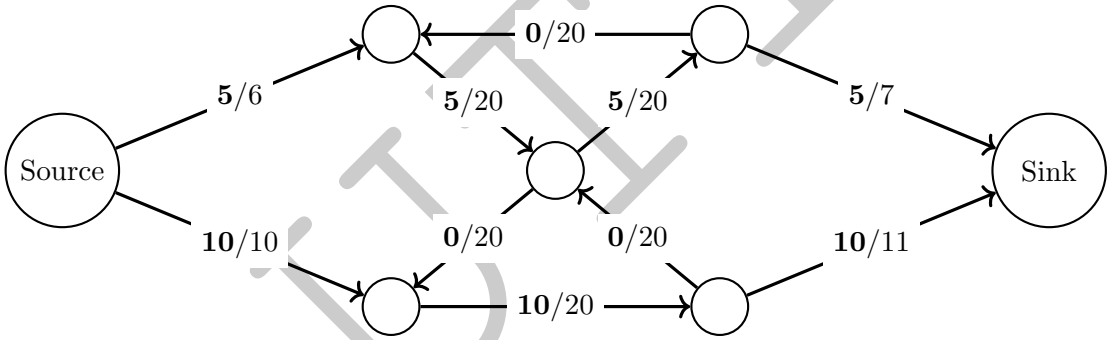
Solution

Flows

a)



b)



c)

$$
\begin{aligned}
&\text{maximize } x_1 \\
&\text{subject to: } \forall i \; x_i \geq \quad 0 \\
&\qquad\qquad \forall i \; x_i \leq \quad c_i \\
&\qquad\qquad x_1 \quad = x_2 + x_3 \\
&\qquad\qquad x_2 \quad = x_4 + x_5 \\
&\qquad\qquad x_3 + x_5 = x_6 + x_7 \\
&\qquad\qquad x_4 + x_6 = \quad x_8
\end{aligned}
$$

d)

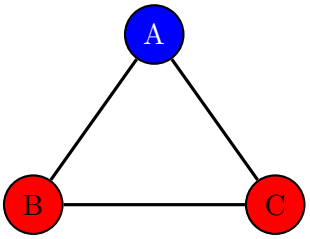
$$
\begin{aligned}
x_7 - x_8 &\leq 2 \\
x_8 - x_7 &\leq 2
\end{aligned}
$$

2 Complexity (20 points)

Coloring

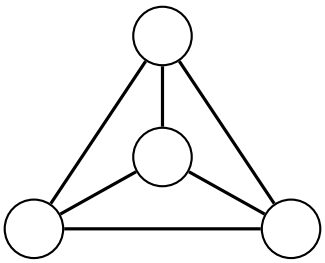
In the k-Coloring problem the input is an undirected graph $G = (V, E)$, with vertex set V and edge set E. The goal is to find a function $c : V \to \{1, 2, \ldots, k\}$, called a *coloring*, such that for all edges $(a, b) \in E$ we have $c(a) \neq c(b)$. In other words, a coloring is an assignment of colors $1, 2, \ldots, k$ to the vertices of G such that no edge has the same color assigned to its endpoints. Note that some colors might remain unused, as well as some colors might be used more than once. Sometimes, we use actual colors, like blue, red, etc., instead of $1, \ldots, k$.

a) `[2 points]` The graph on the right is colored with 2 colors: blue (A) and red (B and C). Is this a valid 2-coloring? Explain.



- ○ Yes, because:
- ○ No, because:

b) `[3 points]` Can the graph on the right be colored with 3 colors? If yes, give a 3-coloring (you can use the numbers $1, 2, 3$ for the colors). If not, explain why.



- ○ Yes, and I give a coloring in the figure to the right.
- ○ No, because:

For the remaining parts of the question, k-COL will denote the decision problem: "Given a graph G, can it be colored with k colors?" It is known that 1-COL and 2-COL are in **P**, while 3-COL is **NP**-hard. You can assume these facts and everything proved in the lectures without proof.

c) `[3 points]` For every fixed constant k, explain why k-COL is in **NP**. Then, use this to show that 3-COL is **NP**-complete.

d) [8 points] Prove that 4-COL is **NP**-complete by finding a polynomial reduction from 3-COL to 4-COL; i.e. show that 3-COL $\leq$ 4-COL. **Hint.** Try adding one node.

By a similar argument to **d)**, it can be shown that k-COL $\leq (k+1)$-COL for all numbers $k \geq 1$. In other words, 1-COL $\leq$ 2-COL $\leq$ 3-COL $\leq$ 4-COL $\leq \ldots$

e) [4 points] Consider the following two statements:

(i) Since 2-COL $\leq$ 3-COL and 3-COL is **NP**-complete, it follows that 2-COL is also **NP**-complete.

(ii) Since 1-COL $\leq$ 2-COL and 2-COL is in **P**, it follows that 1-COL is also in **P**.

For each statement, indicate whether it is true or false. If the statement is false, briefly explain why.

Solution

a) No, because nodes B and C are colored with the same color, which is not allowed.

b) No, because every node is connected to every other node, so the colors of the 4 vertices have to be pairwise distinct, which is not possible with only 3 colors, by pigeonhole.

c) Fix a constant k, then k-COL is in **NP** because a k-coloring can be used as a certificate. Checking whether a proposed k-coloring is valid can be done in polynomial time. Hence, 3-COL is in **NP**, and, since 3-COL is **NP**-hard, by definition 3-COL is **NP**-complete.

d) Given a graph $G = (V, E)$, we define a graph $G' = (V', E')$, where $V' = V \cup \{*\}$ and $E' = E \cup \{(*, v) \mid v \in V\}$. We claim that G is 3-colorable iff G' is 4-colorable:

- If G is 3-colorable, then we can 4-color G' by using the 3-coloring of G on V and assigning color 4 to $*$.
- If G' is 4-colorable, then WLOG assume node $*$ gets color 4. Since node $*$ is linked to all nodes in V, it follows that nodes in V get colors from the set $\{1, 2, 3\}$, hence describing a 3-coloring of G.

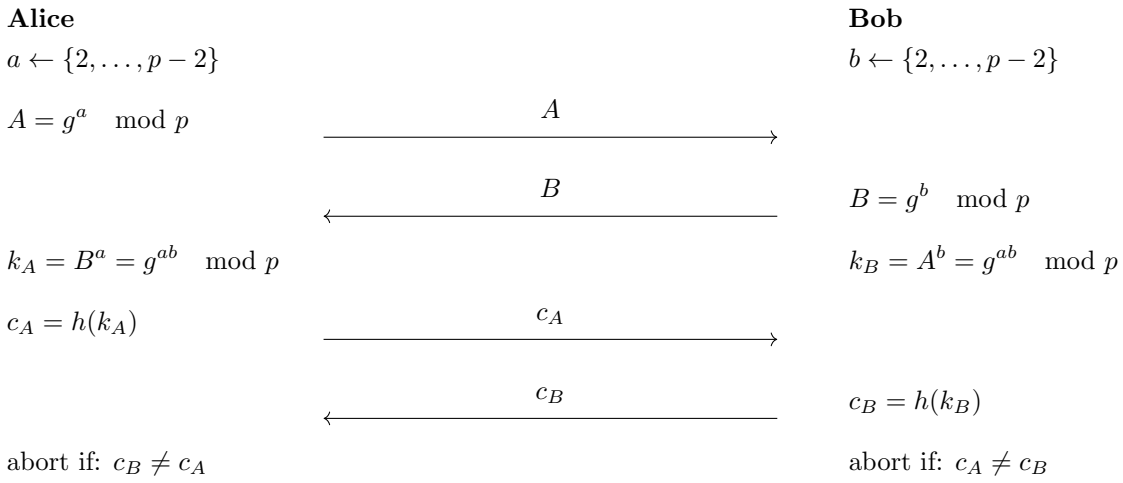
Since G' can be constructed from G in polynomial time, this completes the reduction 3-COL $\leq$ 4-COL, implying that 4-COL is **NP**-complete.

e) Statement **e)**(ii) is correct. Statement **e)**(i) is incorrect. Note that, when $A \leq B$, it means that problem A is no harder than B, so, given that 2-COL $\leq$ 3-COL; i.e. 2-COL is no harder than 3-COL; and that 3-COL is hard, it does not follow that 2-COL is also hard.

3 Cryptography (15 points)

Diffie-Hellman

a) `[7 points]` The regular (unauthenticated) Diffie-Hellman key agreement is vulnerable to man in the middle attacks. Here we show a modified Diffie-Hellman protocol. The public protocol parameters g, p, and h are the generator, prime modulus, and some cryptographic hash function respectively.
Argue whether this is still vulnerable to a similar attack.



b) [8 points] Adapt the unauthenticated Diffie-Hellman key exchange from the lecture for 3 parties, instead of just 2, by filling in the blanks below. Your protocol should ensure:

- Without active interference (man in the middle attack), in the end Alice, Bob, and Charlie share the same secret, i.e. have $k_A = k_B = k_C$.
- An eavesdropper listening in on all communication between the three cannot learn the secret.

The public protocol parameters g and p are the generator and prime modulus respectively.

Alice	**Bob**	**Charlie**
$a \leftarrow \{2, \ldots, p-2\}$	$b \leftarrow \{2, \ldots, p-2\}$	$c \leftarrow \{2, \ldots, p-2\}$
$A = g^a \mod p$	$B = g^b \mod p$	$C = g^c \mod p$
	$\xrightarrow{A}$	$\xrightarrow{B}$
	$\xleftarrow{C}$	
$m_A =$ _____	$m_B =$ _____	$m_C =$ _____
	$\xrightarrow{m_A}$	$\xrightarrow{m_B}$
	$\xleftarrow{m_C}$	
$k_A =$ _____	$k_B =$ _____	$k_C =$ _____

Solution

a) Yes, a man in the middle attacker could still perform the same attack as on the protocol from the lecture. Thus, they establish two different shared secrets, one with Alice (i.e. k_a) and one with Bob (i.e. k_b). After that, they can intercept the messages c_a and c_b and replace them with hashes over the corresponding keys instead, sending $h(k_a)$ to Alice and $h(k_b)$ to Bob.

b) This can be achieved by performing a similar exchange in a circular manner:

Alice	**Bob**	**Charlie**
$a \leftarrow \{2, \ldots, p-2\}$	$b \leftarrow \{2, \ldots, p-2\}$	$c \leftarrow \{2, \ldots, p-2\}$
$A = g^a \mod p$	$B = g^b \mod p$	$C = g^c \mod p$
	$\xrightarrow{A}$	$\xrightarrow{B}$
	$\xleftarrow{C}$	
$m_A = C^a = g^{ac} \mod p$	$m_B = A^b = g^{ab} \mod p$	$m_C = B^c = g^{bc} \mod p$
	$\xrightarrow{m_A}$	$\xrightarrow{m_B}$
	$\xleftarrow{m_C}$	
$k_A = m_C^a = g^{abc} \mod p$	$k_B = m_A^b = g^{abc} \mod p$	$k_C = m_B^c = g^{abc} \mod p$

4 Data and Storage (18 points)

School Library

We consider the SQL database of a school library. The database contains information about students, borrows and books in its tables (table keys are underlined):

students(student_id, firstname, lastname)
borrows(borrow_id, student_id, book_id, borrow_status)
books(book_id, name, pagecount, author)

The borrow_status is true, if the book is still borrowed; borrow_status is false, if the book was borrowed but has been returned.

a) [3 points] Explain what the following SQL query returns:

```
SELECT books.author, COUNT(borrows.borrow_id) AS count
FROM books
LEFT OUTER JOIN borrows ON books.book_id = borrows.book_id
GROUP BY books.author;
```

b) [3 points] What would change if we replaced the LEFT OUTER JOIN in the previous query by an INNER JOIN?

c) `[4 points]` Write a query that returns the names of all books that are currently borrowed by students.

d) `[8 points]` Write a query that returns the id of the student that has read the most pages (total number of pages of books borrowed and returned). (You can assume that no two students have read the same number of pages and that at least one book has been borrowed and returned.)

Solution

a) The query returns how many times books by each author were borrowed.

b) The query will only return the number of times books by each author borrowed, if at least one of the author's books was borrowed.

c)
```
SELECT books.name
FROM books
INNER JOIN borrows ON borrows.book_id = books.book_id
WHERE borrows.borrow_status = true;
```

d)
```
SELECT students.student_id
FROM students
INNER JOIN borrows ON students.student_id = borrows.student_id
INNER JOIN books ON borrows.book_id = books.book_id
WHERE borrows.borrow_status = false
GROUP BY students.student_id
ORDER BY SUM(books.pagecount) DESC
LIMIT 1;
```

5 Machine Learning (18 points)

Multiple Choice

a) `[4 points]` Do these methods help fighting overfitting? Select the right answers.

- Using a more complex model. ○ True ○ False
- Using a less complex model. ○ True ○ False
- Making the training dataset smaller. ○ True ○ False
- Making the validation dataset bigger. ○ True ○ False

Logistic Regression and Gradient Descent

You are given the data in Figure 1 where a blue cross means that a point belongs to class **0** and the orange square that it belongs to class **1**.

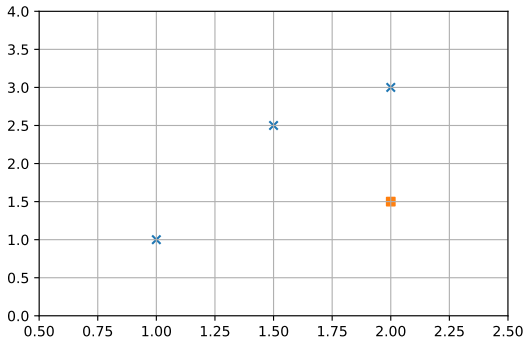


Figure 1: Test data, 3 points of class **0**, one point of class **1**.

The coordinates of the points are:

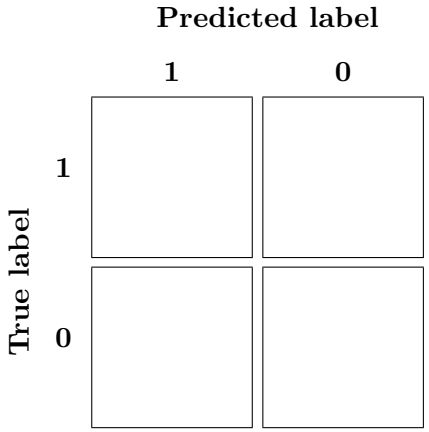
$$a = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad b = \begin{pmatrix} 2 \\ 1.5 \end{pmatrix}, \quad c = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \quad d = \begin{pmatrix} 1.5 \\ 2.5 \end{pmatrix}.$$

We want to find a model with logistic regression using Log Loss and optimize with gradient descent. For a weight vector w and input $\boldsymbol{x}$ (starting with an extra constant 1, as usual) we predict class **1** if $\sigma(w^{\mathrm{T}}\boldsymbol{x}) > 0.5$ (where σ is the sigmoid function) and class **0** otherwise. Some steps of the gradient descent algorithm were already executed and w was most recently set to $w = (-2, 0, 1)^{\mathrm{T}}$. We continue the algorithm from here.

b) `[4 points]` What are the current class predictions (**0** or **1**) for the points in the dataset?

$\hat{f}(a) =$ $\hat{f}(b) =$ $\hat{f}(c) =$ $\hat{f}(d) =$

c) [2 points] Fill in the confusion matrix for the model with the current w.



d) [2 points] Draw the current decision boundary into Figure 1.

e) [6 points] Run the next gradient descent step by using the mean of the gradients for all datapoints in one batch (see hint) and give the updated w, as well as the new predictions for the points. The learning rate α is set to $\alpha = 2$.

Hint: $\frac{\partial L}{\partial w_j} = \frac{1}{n} \sum_{(\boldsymbol{x},y)\in D} \left[\hat{f}(\boldsymbol{x}) - y\right] \cdot x_j$

$\hat{f}(a) =$ $\hat{f}(b) =$ $\hat{f}(c) =$ $\hat{f}(d) =$

$w =$

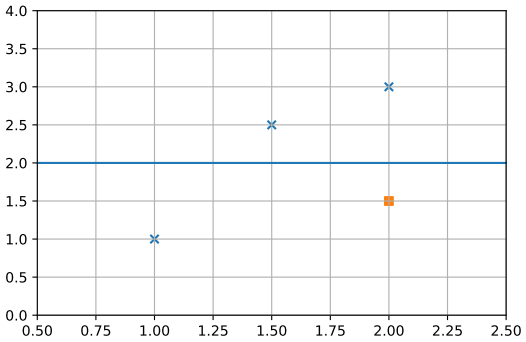
Solution

a) **Correct:** Using a less complex model, **Incorrect:** the rest

b) $\hat{f}(a) = \mathbf{0}, \hat{f}(b) = \mathbf{0}, \hat{f}(c) = \mathbf{1}, \hat{f}(d) = \mathbf{1}$

c) TP $= 0$, FN $= 1$, FP $= 2$, TN $= 1$

d) The boundary looks as follows



e) $w_j = w_j - \alpha \frac{\partial L}{\partial w_j} = w_j - \alpha \frac{1}{n} \sum_{(\boldsymbol{x},y)\in D} \left[\hat{f}(\boldsymbol{x}) - y\right] \cdot x_j$

$$\Rightarrow w = \begin{pmatrix} -2 \\ 0 \\ 1 \end{pmatrix} - \frac{1}{2} \cdot \left(0 \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - 1 \cdot \begin{pmatrix} 1 \\ 2 \\ 1.5 \end{pmatrix} + 1 \cdot \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} + 1 \cdot \begin{pmatrix} 1 \\ 1.5 \\ 2.5 \end{pmatrix} \right) = \begin{pmatrix} -2.5 \\ -0.75 \\ -1 \end{pmatrix}$$

$\hat{f}(a) = \hat{f}(b) = \hat{f}(c) = \hat{f}(d) = \mathbf{0}$

6 Neural Networks (14 points)

Multiple Choice

a) `[1 point]` The output of the attention mechanism does not depend the order of its inputs, i.e., is permutation-invariant, e.g. $(1, 2, 3) = (2, 1, 3) = \ldots$.

- ○ True
- ○ False

b) `[1 point]` The number of weights of a convolutional layer is independent of the input size.

- ○ True
- ○ False

2-D Pattern Recognition

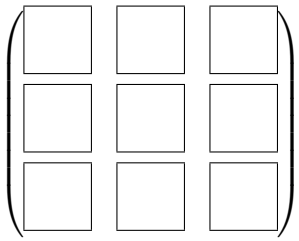
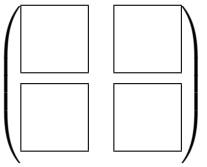
We introduce the concept of Max-Pooling. Max-Pooling is used to accumulate features. It simply describes the process of taking the maximum element of a fixed-size (in the example below 2×2) window.

$$\begin{pmatrix} 5 & 3 \\ 0 & 1 \end{pmatrix} \xrightarrow{\textit{Max-Pool}} (5)$$

Consider now the following black-and-white (entries are 0 or 1) image:

$$\boldsymbol{X} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

d) `[6 points]` Design a convolutional filter $\boldsymbol{W}$ of minimal size that detects 1px wide vertical line segments of 1s of at least 3px height with 0s on both sides, and apply it to $\boldsymbol{X}$ to compute $\boldsymbol{Y}$. The output o of the network is then computed as $o = \mathit{ReLU}(\mathit{Max\text{-}Pool}(\boldsymbol{Y}))+b$. Design $\boldsymbol{W}$ and b in such a way that the output o is positive if and only if there is such a vertical line segment in the input.

$\boldsymbol{W} =$  , $\boldsymbol{Y} = \boldsymbol{X} \circledast \boldsymbol{W} =$  , $b =$ 

e) [2 point] What would be the minimum size of **W** if the detected line segments should be **exactly** 3px high?

f) [2 points] What happens to the output of a convolutional filter if the input size doubles?

g) [2 points] Can we use only convolutional and max-pooling layers to determine the position of a single vertical line segment in the image? Justify your answer.

Solution

Multiple Choice

a) True. The value vectors get multiplied by the normalized attention scores and then added up. This adding up makes the attention mechanism permutation invariant.

b) True. If the input size changes, the output size changes with it, but the number of weights stays the same.

2-D pattern recognition

d)

$$\boldsymbol{W} = \begin{pmatrix} -1 & 1 & -1 \\ -1 & 1 & -1 \\ -1 & 1 & -1 \end{pmatrix}, \boldsymbol{Y} = \boldsymbol{X} \circledast \boldsymbol{W} = \begin{pmatrix} 3 & -5 \\ 1 & -2 \end{pmatrix}, b = -2$$

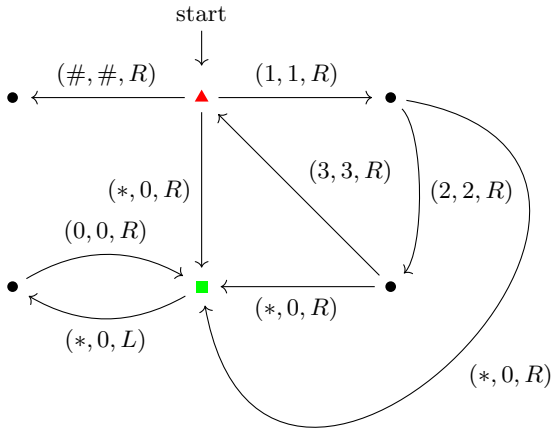
e) The size needs to be at least 5x3, as there need to be 3 rows to detect the 3px long line segment, and one row each at the top and the bottom to make sure it is not longer than that.

f) The output of the filter application scales linearly with the input size.

g) No, convolutional and max-pooling layers are not enough. Both types of layers are translation-invariant, which makes it impossible to determine the location of the line segment, as we can translate it to a different position while still getting the same output.

7 Computability (16 points)

Let A be the single-tape Turing Machine (TM) drawn below, operating on the alphabet $\mathcal{A} := \{0, 1, 2, 3, \#\}$. Recall that the triple (C, W, M) denotes the instruction *if read* C*ondition then write* W *and move* M, where M is Left, Right, or Stay. If $C = *$ then C can be any letter in $\mathcal{A}$ not already used by an incident out-transition. If the reading head of A encounters a symbol not appearing at any out-transition of the current state, A halts.

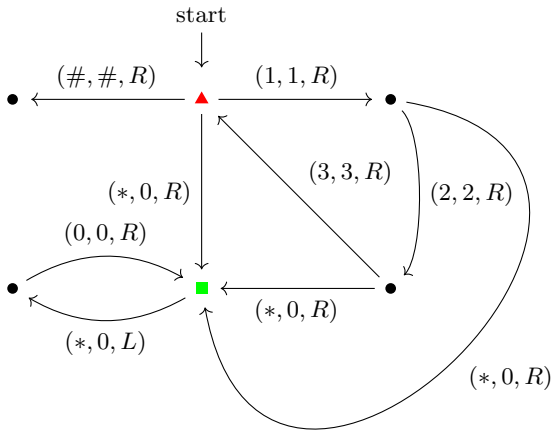


Call a word *input* if it is of finite length, formed from $\{0, 1, 2, 3\}$, and terminated by #.
Call a word *output* if it appears on the tape immediately after the terminal # once the machine has halted. Assume that the tape is blank wherever there is no symbol written on it.

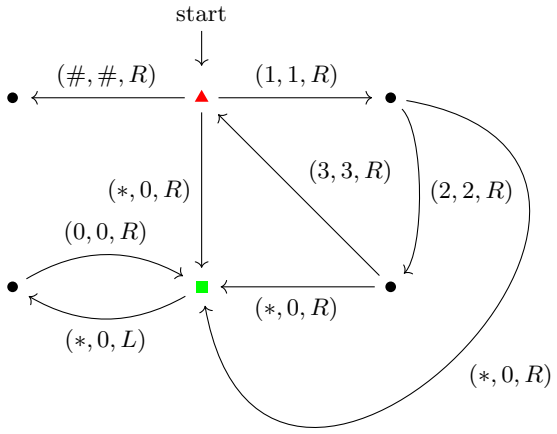
a) `[3 points]` Give two examples of inputs for which A halts and two for which it does not.

b) `[3 points]` Characterize the set of all inputs for which A halts. Can you identify a single pattern they follow?

c) [5 points] Does there exist a TM B that outputs 1 (and halts) if and only if A halts; and 0 if and only if A does not halt for the input given? Either draw an example of B or say why it cannot exist. (We reprint the original TM $\mathcal{A}$ in case you might find it useful for your solution.)



d) [5 points] Does there exist a TM C that halts if and only if A does not halt for the input given? Either draw an example of C or say why it cannot exist. (We reprint the original TM $\mathcal{A}$ in case you might find it useful for your solution.)

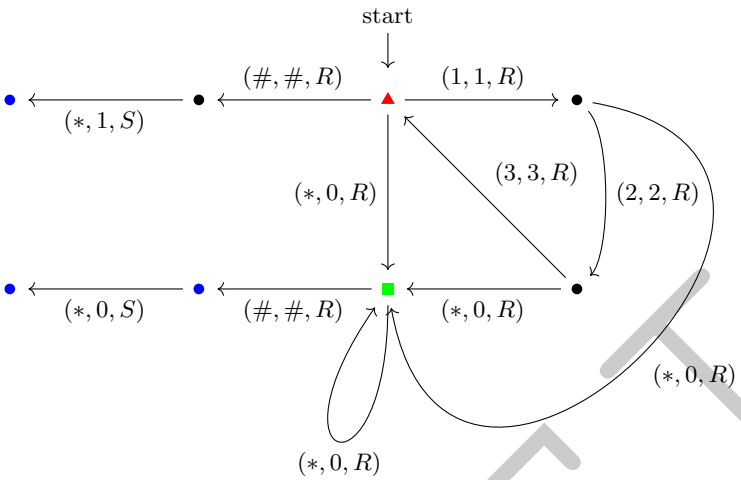


Solution

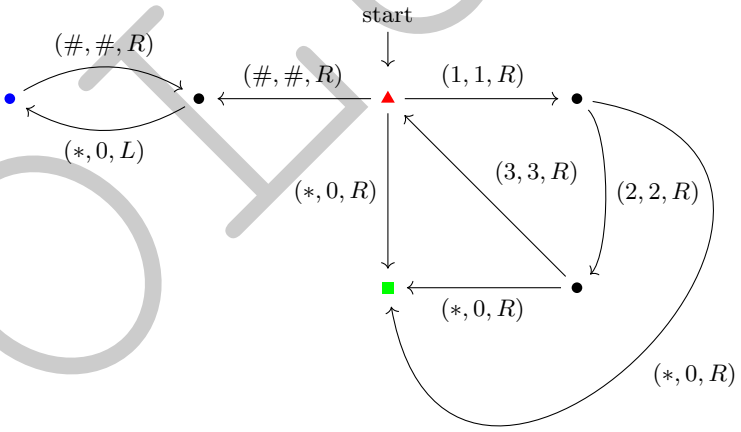
a) A halts for 123 and 123123. A does not halt for 1 or 12312.

b) A halts for any input that is a series of repetitions of 123.

c) Yes. To form an example of B, just modify the appropriate states of A (green and red) with writer sub-machines. New states are marked with blue.



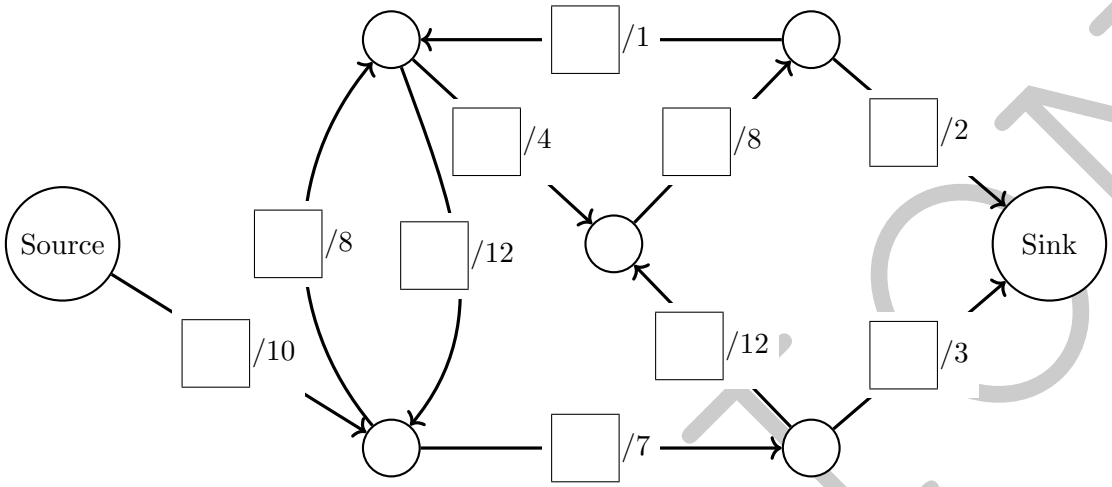
d) Yes. To form an example of C, just modify the appropriate states of A (green and red). The one new state is marked with blue.

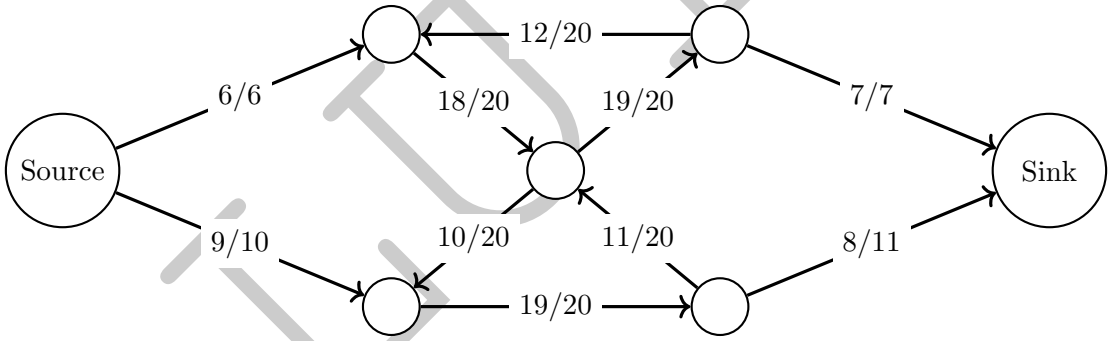


Replacements

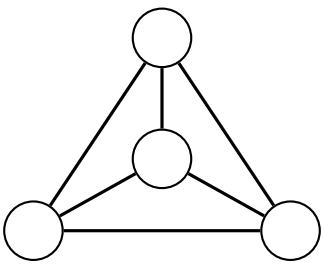
Here you can find replacement templates for the fill in tasks, in case your original solution becomes too messy. If you use these, please **clearly** indicate which one we should consider.

Question 1





Question 2



Question 3

Alice	**Bob**	**Charlie**
$a \leftarrow \{2, \ldots, p-2\}$	$b \leftarrow \{2, \ldots, p-2\}$	$c \leftarrow \{2, \ldots, p-2\}$
$A = g^a \mod p$	$B = g^b \mod p$	$C = g^c \mod p$

$A \rightarrow$ $\quad$ $B \rightarrow$

$\leftarrow C$

$m_A =$ _____ $\quad$ $m_B =$ _____ $\quad$ $m_C =$ _____

$m_A \rightarrow$ $\quad$ $m_B \rightarrow$

$\leftarrow m_C$

$k_A =$ _____ $\quad$ $k_B =$ _____ $\quad$ $k_C =$ _____

Question 5

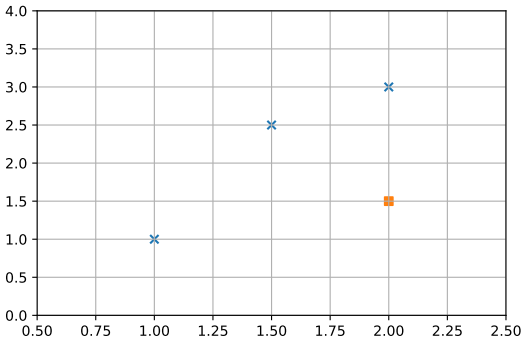
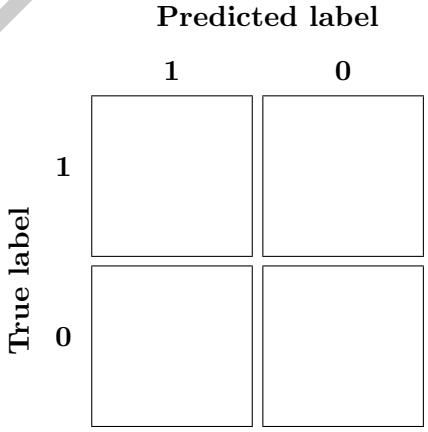
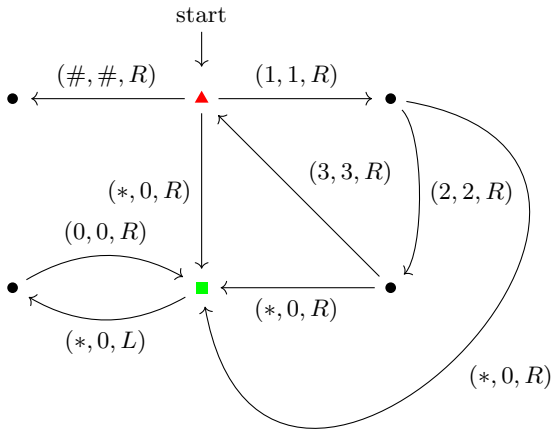
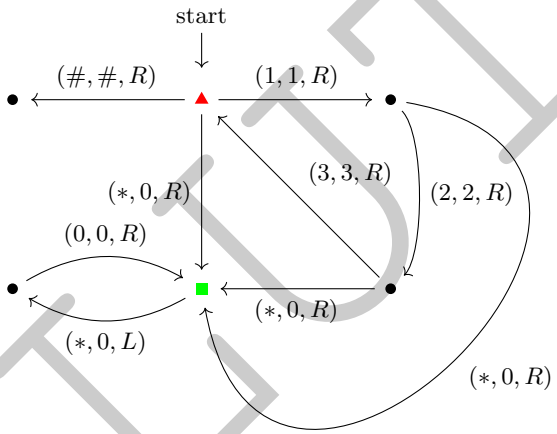


Figure 2: Test data, 3 points of class **0**, one point of class **1**



Question 7





Use this page if you need extra space.