Berlin, Februrary 22, 2024

Name: .... **Matr.-Nr.:** ....

Advanced Algorithmics Exam

(Nichterlein, Winter Term 2023/2024)

Exercise:	1	2	3	4	5	6	7	8	**Sum**
Points:	20	22	18	18	20	12	18	12	140
Achieved:									

Time limit: 120 Minutes

Max. Number of Points: 140 Points
Best grade: $\geq$ 86 Points

General hints:

- You are not allowed to use any technical aids or learning material during the exam.
- Do not use a pencil. Use a black or blue pen (non-erasable).
- Write you name and matriculation number on each sheet.
- **If not explicitly excluded in the task, then all answers must be justified! Answers without justification receive 0 points.**
- For $n \in \mathbb{N}$, $n \geq 1$, set $[n] := \{1, \ldots, n\}$.

We wish you success!

Name: .... **Matr.-Nr.: ....**

Exercise 1: **Reduction to Solvers** (20 Points)

Provide integer linear programming (ILP) formulations for the two following problems. Use only a polynomial number of variables and constraints!

Define the used variables, all constraints, and the objective function (maximization or minimization). Explain the intended meaning of your variables. *No further justification (correctness etc.) is needed.*

(a) MAXIMUM CLUSTER SUBGRAPH (10 P)

Input: An undirected graph $G = (V, E)$.

Task: Find a maximum set $S \subseteq E$ of edges such that $G' = (V, S)$ is a cluster graph, that is, every component of G' is a clique.

Hint: A graph is a cluster graph if and only if it does not contain a path on three vertices as an induced subgraph.

(b) The following fair allocation problem is also known as the *Santa Claus problem*: Santa Claus has p gifts (i.e., indivisible items) to allocate to n children (i.e., agents) having additive and non-negative preferences over the gifts. Santa Claus wants to allocate the gifts in a way that maximizes the utility of the unhappiest child (i.e., wants to find an allocation maximizing egalitarian social welfare). (10 P)

Use the following notation: $u_i(g_j)$ denotes the utility child i gives to gift j.

Name: .... **Matr.-Nr.: ....**

(You can use this paper for notes, scratches, or solutions (clearly reference the task in the latter case).)

Name: .... **Matr.-Nr.: ....**

Exercise 2: **Color Coding and Parameterized Algorithms** (22 Points)

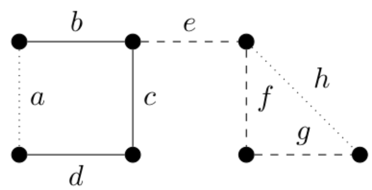
Consider the following graph problem:

MONOLABELED COMPONENTS

Input: An undirected graph $G = (V, E)$, an edge labeling $\ell \colon E \to [k]$, and an integer $r \in \mathbb{N}$.

Question: Is there a set $E' \subseteq E$ of r edges such that each connected component of $G' := (V, E')$ is *monolabeled*, i.e., each connected component contains edges of (at most) one label?

(a) Consider the following graph G with edge labels *solid*, *dotted*, and *dashed*. The edges are named $a, \ldots, h$. (2 P)



Find a largest set $E' \subseteq \{a, \ldots, h\}$ such that $G' = (V, E')$ is monolabeled (*without justification*).

(b) Provide a randomized algorithm solving MONOLABELED COMPONENTS in $f(r) \cdot n^{O(1)}$ time with some positive constant probability. (20 P)

Hint: Recall that $(1-x)^t \leq e^{-t \cdot x}$ for $0 \leq x \leq 1$. A connected subgraph with $i \in \mathbb{N}$ edges has at most $2 \cdot i$ vertices.

Name: .... **Matr.-Nr.: ....**

(You can use this paper for notes, scratches, or solutions (clearly reference the task in the latter case).)

Name: .... **Matr.-Nr.:** ....

Exercise 3: **Approximation** (18 Points)

Consider the following problem:

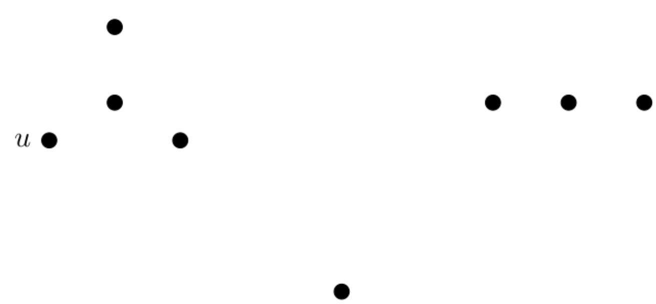
> k-Center
>
> **Input:** A set V of points and a metric distance function $d\colon V \times V \to \mathbb{R}_{\geq 0}$, i.e., $d(u,u) = 0$ for each $u \in V$, $d(u,v) = d(v,u)$ for each $u, v \in V$, and $d(u,w) \leq d(u,v) + d(v,w)$ for each $u, v, w \in V$.
>
> **Task:** Find a set $S \subseteq V$ of exactly k points (called *centers*), that minimizes $r := \max_{v \in V} d(v,S)$, where $d(v,S) := \min_{u \in S} d(v,u)$, that is, every point $v \in V$ has a center within distance at most r.

This is an (approximation) algorithm for the k-Center problem.

```
S := {u} for an arbitrary u ∈ V
while |S| < k:
    v := arg max_{v∈V} d(v, S)
    S := S ∪ {v}          // add point furthest away from any point in S to S
return S
```

Let $S^* = \{v_1^*, \ldots, v_k^*\} \subseteq V$ be an optimal solution with radius $r^* = \max_{v \in V} d(v, S^*)$ and $S = \{v_1, \ldots, v_k\}$ the solution produced by the algorithm with $r = \max_{v \in V} d(v, S)$.

(a) *(Without justification)* Consider the following point set with Euclidean distances as 3-Center instance. Mark S^* with squares and S with circles (assume the algorithm starts with point u). (2 P)



(b) Denote with $\mathcal{C}(S^*)$ the clusters induced by S^*, that is, $\mathcal{C}(S^*) = \{C_1^*, \ldots, C_k^*\}$ with $C_i^* \subset V$ being the points closest to the i^{th} center v_i^*. Formally (6 P)

$$C_i^* = \{u \in V \mid d(u, v_i^*) = d(u, S^*)\}.$$

Show that if the centers of S are in different clusters in $\mathcal{C}(S^*)$ (formally, $\forall i \in [k]\colon v_i \in C_{\sigma(i)}^*$ for a bijection $\sigma : [k] \to [k]$), then S is a 2-approximation, that is, $r \leq 2r^*$.

(c) Show that if S contains two centers from one cluster in $\mathcal{C}(S^*)$ (formally, $\exists i, i', j \in [k] : v_i, v_{i'} \in C_j^* \wedge i \neq i'$), then S is a 2-approximation, that is, $r \leq 2r^*$. (10 P)

Name: .... **Matr.-Nr.: ....**

(You can use this paper for notes, scratches, or solutions (clearly reference the task in the latter case).)

Name: .... **Matr.-Nr.: ....**

Exercise 4: **Unique Perfect Matching** (18 Points)

Let $G = (U, W, E)$ be a bipartite graph where $U = \{u_1, \ldots, u_n\}$ and $W = \{w_1, \ldots, w_n\}$.

Provide one $n \times n$ matrix A based on G that has the following three properties:

(a) The determinant[1] $\det A$ is a multivariate polynomial $f(x_1, \ldots, x_\ell)$ for some $\ell \in [n^2]$. (2 P)

(b) G has a perfect matching[2] if and only if f is not the zero polynomial (i. e. $\det A \not\equiv 0$). (6 P)

(c) Let $d \in \{0, 1\}^\ell$ be a binary vector where the i^{th} entry is defined as: (10 P)

$$d[i] := \begin{cases} 0, & \text{if } \frac{\partial \det A}{\partial x_i} = \frac{\partial f(x_1, \ldots, x_\ell)}{\partial x_i} \equiv 0 \\ 1, & \text{otherwise.} \end{cases}$$

Given d, it can be decided in $O(\ell) \subseteq O(n^2)$ time if G admits a *unique*[3] perfect matching.

Hints: This matrix A can be used with the tools of lecture about algebraic algorithms (Schwarz-Zippel lemma, Baur-Strassen theorem, ...) in a randomized algorithm deciding in $\mathcal{O}(n^\omega)$ time (where $\omega \approx 2.3716$ is the matrix multiplication constant), whether there is a unique perfect matching in G.

[1] Recall that $\det A = \sum_{\sigma \in S_n} \operatorname{sgn}(\sigma) A[1, \sigma(1)] \cdots A[n, \sigma(n)]$.

[2] A perfect matching M is a set of n edges in G such that each vertex is incident to exactly one edge in M.

[3] That is, G contains exactly one perfect matching.

Name: .... **Matr.-Nr.: ....**

(You can use this paper for notes, scratches, or solutions (clearly reference the task in the latter case).)

Name: .... **Matr.-Nr.: ....**

Exercise 5: **Online Algorithms** (20 Points)

Consider a two-level memory system that consists of a fast memory (called cache) that can hold k memory pages and an arbitrarily large slow memory.

Each memory request of an application specifies a page number in the memory system. A request is *served* if the corresponding page is in the cache. If a requested page is not in the cache, then a *cache miss* occurs and a page must be removed from the cache so that the requested page can be loaded into the free location of the cache.

Our goal is to minimize the number of cache misses when serving a sequence σ of page requests that appear in an online fashion.

Consider the following strategy:

LRU Keep track at which time each page in the cache was last accessed. On a cache miss, remove the page whose access time is the longest in the past.

(a) Show that LRU is k-competitive.[4] (12 P)

Hint: Partition the sequence σ into *phases* such that in each phase, LRU makes exactly k cache misses.

(b) Show that this analysis is tight, i.e., give a sequence σ for which LRU has k times more cache misses than OPT. (8 P)

[4]For an input sequence σ, let $A(\sigma)$ be the cost incurred by online algorithm A and let $\mathrm{OPT}(\sigma)$ be the cost incurred by an optimal offline algorithm. Then A is called *c-competitive* if there is a constant d such that, for all σ, $A(\sigma) \leq c \cdot \mathrm{OPT}(\sigma) + d$.

Name: .... **Matr.-Nr.: ....**

(You can use this paper for notes, scratches, or solutions (clearly reference the task in the latter case).)

Name: .... **Matr.-Nr.: ....**

Exercise 6: **External Memory** (12 Points)

We consider Counting Sort in the external memory setting. Consider the case where an array A contains N integers between 1 and ℓ (which will be determined later). The array A is stored in $\lceil N/B \rceil$ blocks in the external memory. Each block can store B integers.

The algorithm is as follows:

Algorithm 1: Counting Sort

Input: An array A with N integers between 1 and ℓ.
Output: Sorted A.

```
Count ← array of length ℓ initialized with all entries being 0
for i = 1 to N do  Count[A[i]] ← Count[A[i]] + 1
    // Count now stores in index i how often the integer i appears in A
j ← 1
for i = 1 to ℓ do
    while Count[i] > 0 do
        A[j] ← i
        j ← j + 1
        Count[i] ← Count[i] − 1
return A
```

For simplicity, assume the internal memory has size $M = 3B$.

Does the above algorithm achieve $O(N/B)$ I/O's if $\ell = B$? If yes, then show which data are stored in the internal memory and analyze the number of I/O's in the above algorithm. If not, then provide a counter example for A where any memory handling procedure results in $\omega(N/B)$ I/O's.

Name: .... **Matr.-Nr.: ....**

(You can use this paper for notes, scratches, or solutions (clearly reference the task in the latter case).)

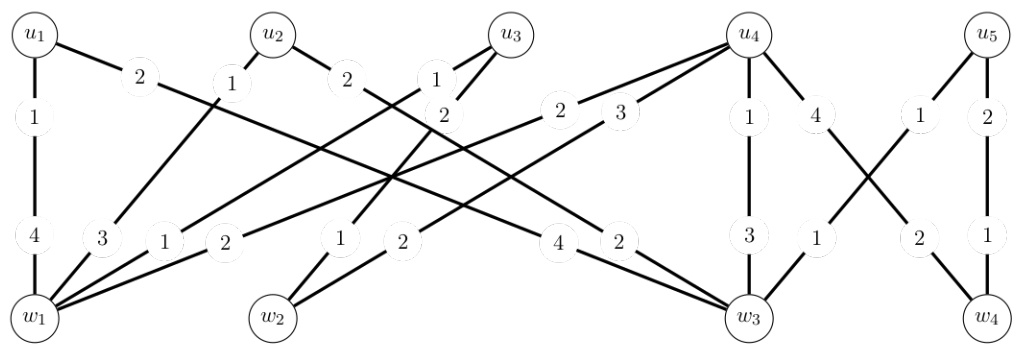
Name: .... **Matr.-Nr.: ....**

Exercise 7: **Stable Matching** (18 Points)

Consider the bipartite stable matching problem with strict preferences over the two sets $U = \{u_1, u_2, \ldots, u_5\}$ and $W = \{w_1, w_2, \ldots, w_4\}$. Consider the Gale-Shapley algorithm[5] in which the set U proposes. You can use without proof the fact that the algorithm runs on non-complete bipartite graphs in the same way as on complete ones. In particular, the algorithm returns a stable matching[6].

Call an edge *rejected* if it is proposed (by some u_i) and gets rejected (by some w_i either immediately or later when w_i accepts another proposal) during the Gale-Shapley algorithm.

(a) (*Without justification*) For the following instance[7], write down a list of all rejected edges and indicate in which round they have been rejected. (4 P)



(b) Prove that rejected edges cannot be part of any stable matching. (10 P)

(c) Use the statement in (b) to argue that the U-proposing Gale-Shapley algorithm assigns each $u \in U$ to their best possible stable partner (i.e., to their most preferred $w \in W$ who can be matched to u in any stable matching). (4 P)

[5] Recall that the algorithm works roughly as follows: Initially, no agent is matched. As long as there is an unmatched $u \in U$ that did not yet propose to all its neighbors, agent u proposes to the most-liked agent w (according to u's preference) that did not yet reject u. If w is unmatched or w is matched to u' and w prefers u over u', then match u to w and u' (if existing) becomes unmatched.

[6] A pair $u \in U$ and $w \in W$ is called a blocking pair for a matching if both agents prefer each other to their currently matched partner. A matching is stable if there are no blocking pairs.

[7] As in the lecture, the numbers on the edges indicate to position in the preferences, e. g., u_1 has w_1 as its first choice and w_3 as its second choice while u_1 itself is in the forth position in the preferences of w_1 and w_3.

Name: .... **Matr.-Nr.:** ....

(You can use this paper for notes, scratches, or solutions (clearly reference the task in the latter case).)

Name: .... **Matr.-Nr.: ....**

Exercise 8: **Distributed Algorithms and Randomized Algorithms** (12 Points)

Describe in **at most 40 words** connections between Distributed Algorithms and Randomized Algorithms.

Name: .... **Matr.-Nr.: ....**

(You can use this paper for notes, scratches, or solutions (clearly reference the task in the latter case).)