Endterm Algorithms & Data Structures (TI1520TW)

Exam created by Stefan Hugtenburg

Please read the following information carefully!

- You have 2 hours to complete this exam.
- This exam consists of 15 multiple-choice questions and 7 open questions.
- The points for the multiple-choice part of the exam are computed as $1+9\cdot\max\left(0, \frac{\text{score} - 0.25 * 15}{0.75 * 15}\right)$. This accounts for a 25% guessing correction, corresponding to the four-choice questions we use. Every multiple-choice question is worth 1 point.
- Note that the order of the letters next to the boxes on your multiple-choice sheet may **not always be A-B-C-D**!
- The grade for the open questions is computed as: $1 + 9 \cdot \frac{\text{score}}{30}$.
- The **final grade for the exam** is computed as: $0.4 \cdot \text{MC} + 0.6 \cdot \text{Open}$.
- Before you hand in your answers, check that the sheets contain your name and student number, both in the human and computer-readable formats.
- Tip: mark your answers on this exam **first**, and only after you are certain of your answers, copy them to the multiple-choice answer form.
- Read every question properly and in the case of the open questions, give **all information** requested, which should always include a brief explanation of your answer. Do not however give irrelevant information – this could lead to a deduction of points.
- This exam corresponds to all content from the course TI1520TW.
- The use of the book, notes, calculators or other sources is **strictly prohibited**.
- You may write on this exam paper and take it home.
- For the purpose of this exam, we assume P $\neq$ NP and NP $\neq$ EXP, unless stated otherwise.
- Note that the minimum score per (sub)question is 0 points.
-

Open questions:

Question:	16	17	18	19	20	21	22	Total:
Points:	2	3	7	6	3	4	5	30

This page is intentionally left blank.

Multiple-Choice questions

1. Lyra is examining an algorithm that takes two inputs, one of size n and one of size m. After much painstaking research she has managed to conclude that n is $O(m)$. What can we conclude about the run time of the algorithm $T(n,m)$?
   - A. If $T(n,m)$ is $O(n \log m)$, then $T(n,m)$ is also $\Omega(m)$.
   - B. If $T(n,m)$ is $O(n \log(nm))$, then $T(n,m)$ is also $O(m \log m)$.
   - C. If $T(n,m)$ is $O(n^2 m)$, then $T(n,m)$ is also $O(n^3)$.
   - D. If $T(n,m)$ is $O(nm^2)$, then $T(n,m)$ is also $\Omega(n^3)$.

2. Consider the following algorithm:

```
class Edge:
    a: int
    b: int

class Graph:
    edges: List[Edge]

def func(g: Graph) -> List[int]:
    res = list()
    for e in g.edges:
        if e.a not in res:
            res.append(e.a)
        if e.b not in res:
            res.append(e.b)
    return res
```

Which of the following claims about the worst-case run time $T(G)$ of this algorithm on a graph $G = (V, E)$ is **true**?
   - A. $T(G)$ is $\Theta(|V|)$
   - B. $T(G)$ is $\Theta(|E|)$
   - C. $T(G)$ is $\Theta(|E| \log |V|)$
   - D. $T(G)$ is $\Theta(|E| \cdot |V|)$

3. Consider again the algorithm from the previous question. Which of the following claims about the worst-case required space $S(G)$ of this algorithm is **true** ?
   - A. $S(G)$ is $\Theta(1)$
   - B. $S(G)$ is $\Theta(|V|)$
   - C. $S(G)$ is $\Theta(|E|)$
   - D. $S(G)$ is $\Theta(|V| \cdot |E|)$

4. Consider the following two hash functions of strings:
   - $f(s) = \texttt{len(s)} \mod 10$
   - $g(s) =$ (the number of a's in s) $\mod 10$

   Take the set of keys $K = \{$apple, pear, maya, banana, pearl, phoenix, mia, lyra, will, pantalaimon$\}$. Which of the following statements is **true** about the hashes of the keys $k \in K$?
   - A. f will result in more conflicts than g.
   - B. For two anagrams[1] only one of the functions is guaranteed to give a conflict, the other could be safe.
   - C. Taking $f(s) + g(s) \mod 10$ as the hash function would reduce the number of conflicts over using just $f(s)$.
   - D. If we change both functions to be $\mod 9$ instead of $\mod 10$, the number of hash conflicts remains unchanged for both functions.

[1]Two strings are anagrams, if they contain the same letters, but in a different order. For example 'silent' and 'listen' are anagrams.

→ 5. Consider a graph implementation that stores a list of vertex IDs as an array-based list and a list of edges as a linked list (edges are represented by tuples of two vertex IDs). Which of the following statements about this structure is **true** ?

   A. Getting all edges that are incident to a vertex v requires $O(|V|)$ time.

   B. Getting all vertices with more than two incident edges requires $O(|V| + |E|)$ time.

   C. Checking if a vertex exists in this graph requires $O(\log |V|)$ time.

   D. Checking if an edge exists in this graph requires $O(\log |E|)$ time, assuming the list of edges is sorted.

6. Consider the following state of a hash table that uses linear probing with hash function $f(k) = k \mod n$ where n is the size of the hash table. You may assume the table is of a fixed size and will not grow.

Indices:	0	1	2	3	4	5	6	7
Array:	8	1	0	17	12	18		23

Which of the following statements about this hash table is **true**?

   A. Inserting the items in the order: 0, 1, 8, 12, 17, 18, 23 would result in this hash table.

   B. Inserting the items in the order: 1, 8, 12, 23, 17, 0, 18 would result in this hash table.

   C. Inserting the items in the order: 1, 8, 23, 12, 0, 17, 18 would result in this hash table.

   D. Inserting the items in the order: 23, 18, 17, 12, 8, 1, 0 would result in this hash table.

7. Consider again the hash table from the previous question. We now perform the following sequence of insert and delete operations:

   - ~~delete 17~~
   - ~~insert 4~~
   - ~~delete 1~~
   - ~~delete 8~~
   - ~~insert 15~~
   - ~~delete 0~~
   - ~~insert 10~~

What does the resulting hash table look like? Note that we use an 'X' to represent a defunct entry.

A.

Indices:	0	1	2	3	4	5	6	7
Array:	10	15	X	X	12	18	4	23

B.

Indices:	0	1	2	3	4	5	6	7
Array:	X	15	10	X	12	18	4	23

C.

Indices:	0	1	2	3	4	5	6	7
Array:	15	10	X	X	12	18	4	23

D.

Indices:	0	1	2	3	4	5	6	7
Array:	15	X	10	X	12	18	4	23

8. Consider a min-heap containing $2^h - 1$ unique elements, where h is the height of the heap. For this question you may assume $h > 4$. Although normally considered a tree, we can also consider this to be an unweighted directed graph with edges from parents to children. Now consider a BFS traversal of the heap, starting from the root. What can we conclude with certainty about this BFS traversal?

   A. The 4th element listed is guaranteed to be smaller than the last element listed.

   B. The 5th element listed is guaranteed to be larger than the 3rd element listed.

   C. The 7th element listed is guaranteed to be larger than the first element listed.

   D. The last element listed is guaranteed to be the largest element in the heap.

9. Consider the following sorting algorithm of a list L that is guaranteed to contain no duplicate values:

   1. Create a graph $G = (V, E)$.
   2. Create a vertex $v_x \in V$ for every item $x \in L$.
   3. Create a directed edge $e \in E$ from v_x to v_y if $y > x$.
   4. Now return the topological order of G as a sorted list of the input.

   What statement about this algorithm is **true**?

   A. This algorithm does not sort the list.

   B. The run time of this algorithm is $\Theta(n \log n)$ where $n =$`len(L)`.

   C. The run time of this algorithm is $\Theta(n^2)$ where $n =$`len(L)`.

   D. This algorithm requires exponential time as the number of edges required can be exponential in the number of items to be sorted.

10. Consider a weighted directed acyclic graph G. Which of the following statements is guaranteed to be **true**?

    A. G must be strongly connected.

    B. There is exactly one minimum spanning tree for G.

    C. There is exactly one topological ordering of the vertices of G.

    D. There must be at least one vertex with less than 3 incoming edges.

11. Which of the following statements is **true**?

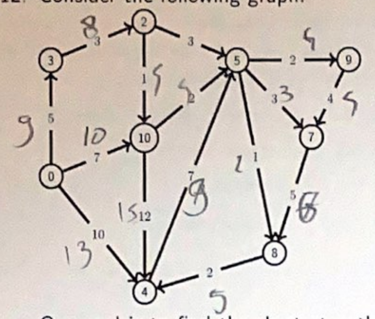
    A. Both a BFS and DFS traversal require a stack to be implemented efficiently.

    B. Both a BFS and DFS traversal can be used to list all vertices at most K edges away from a vertex v for any constant K.

    ~~C.~~ A BFS traversal starting from a vertex v in a directed acyclic graph must always be identical to a DFS traversal of that vertex v in the graph.

    ~~D.~~ A BFS traversal starting from a vertex v might discover different vertices than a DFS traversal starting from vertex v.

12. Consider the following graph:



Our goal is to find the shortest path from vertex 0 to vertex 7. We can either use Dijkstra, or use A* with the heuristic: $h(x) = |7 - x|$. So for instance for vertex 3 the heuristic would return 4.
Which of the following statements about the execution of these algorithms is **true**?

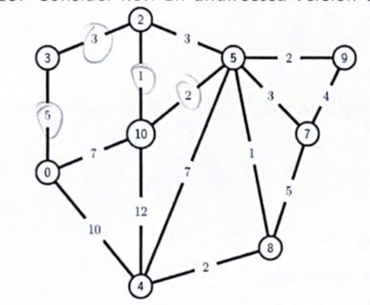
A. A* will visit vertex 10 before vertex 3, whereas Dijkstra will not.

B. A* will visit vertex 8 before vertex 7, whereas Dijkstra will not.

C. Dijkstra will visit vertex 9 before vertex 7, whereas A* will not.

D. Both A* and Dijkstra will visit the vertices in the same order with this heuristic.

13. Consider now an undirected version of the graph from question 12:



Which of the following claims about this graph is **true** ?

A. If we apply Prim's algorithm on this graph starting from vertex 0, the edge between vertices 5 and 10 will not be selected for the MST.

B. No matter what vertex we start Prim's algorithm from, the edge between 0 and 3 will always be a part of the MST.

C. If we apply Kruskal's algorithm on this graph, the edge between vertices 2 and 5 will be a part of the MST.

D. If we apply Kruskal's algorithm on this graph, the edge between vertices 5 and 7 will never be a part of the MST.

14. Which of the following statements is guaranteed to be **true**?

A. Given that a problem X can be solved by an $O(n^2)$ algorithm, that means $X \in$ NP.

B. Given that a problem X can be solved by an $O(2^n)$ algorithm, that means $X \in$ NP.

C. Given a problem $X \in$ NP, that means there is an $O(n^2)$ algorithm to find a solution to X.

D. Given a problem $X \in$ NP, that means there is an $O(n^2)$ algorithm to verify solutions to X.

15. The TAs have worked hard throughout the course and to thank them for this once more Stefan has invited them over to his house for dinner. After dinner, they decide to play a game of Geography. For this game they will form two teams (Stefan vs the TAs), which take it in turn to name a city that starts with the last letter of the previously named city. For instance the first few turns might go like this:

    - Stefan: Delft
    - Maarten: Tiel
    - Stefan: Lutjebroek
    - Yana: Kijkduin
    - Stefan: Naaldwijk
    - Yoshi: Kaag
    - Stefan: Genum
    - Kevin: Medemblik
    - etc.

    As all parties playing the game are well-aware this game can be generalised to Generalised Geography: Given a set of cities to choose from, is there a winning strategy for the team if they start with Delft as the first city? It is not known if this problem (GG for short) is in NP.
    Having determined that this game could go on for a long time, they instead decide to play a cooperative variation of the game. Which of the following variations could they efficiently solve that evening? That is, which of these do we know to be in P?

    A. GG-Delft: Given a set of n cities to choose from and a starting city x, is there a sequence of cities containing Delft?

    [✓ B.] GG-All-Of-Them: Given a set of n cities to choose from, is there a sequence of cities so that all cities are named?

    C. GG-Rally-Record: Given a set of n cities to choose from, is there a sequence of cities so that at least K cities can be named?

    [✓ D.] GG-Phoenix: Given a set of n cities to choose from, is there a sequence of cities starting with Phoenix so that the sequence also contains all other cities from the set starting with P.

Open questions

16. (2 points) Consider a hash map that uses a variation of separate chaining called alethio chaining. This version uses AVL-trees to store the key/value pairs in the different buckets. Does this change the run time of the `put` and `remove` operations of the map? If so, explain how. If not, explain why not.

17. (3 points) Use the master theorem to give a tight bound on the run time of an algorithm with recurrence equation:
$$T(n) = \begin{cases} c_0 & \text{if } n = 1 \\ 8T(n/3) + n^2\sqrt{n} & \text{else} \end{cases}.$$

18. Consider the following algorithm that should be called as `myFunc(g, len(g.nodes))`.

```
class Node:
    edges: List[Node]

class DAG:
    nodes: SinglyLinkedList[Node]

def myFunc(g: DAG, n: int) -> int:
    if n == 0:
        return 0
    else:
        x = g.nodes[n-1]
        return x.val + myFunc(g, n-1)
```

You may assume the class `DAG` represents a directed acyclic graph and that `SinglyLinkedList` implements a singly linked list data structure.

(a) (4 points) Give a recursive expression for the run time T of `myFunc`. Explain the terms in your expression by relating to specific lines of the code.

(b) (3 points) Give a recursive expression for the required space S of `myFunc`. Explain the terms in your expression by relating to specific lines of the code.

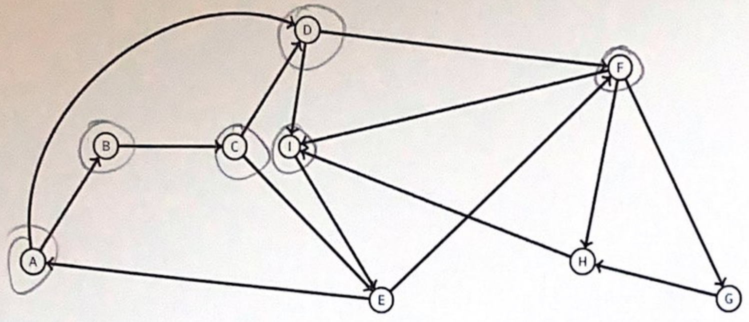
19. (6 points) Consider the following recursive expression for the run time of a recursive algorithm:
$$T(n,m) = \begin{cases} m & \text{if } n = 0 \\ 2T(n-1, m+1) + c_1 & \text{else} \end{cases}$$
Use repeated unfolding to find the closed form of this equation, then use induction to prove this is correct and finally give a tight big-Oh bound on the run time.

20. Famous archaeologist and renowned puzzle-solver Hershel Layton is chasing his nemesis Don Paolo through the cities of London. Layton will of course always abide by the rules and never cross the street when traffic lights tell him to stop, but he has to hurry and get to the theatre where Don Paolo's get-away air balloon is stationed as quickly as possible. To maximise his chances of catching Don Paolo without breaking any laws, Layton would like to know what path through London he should take to minimise the total time spent (number of traffic lights + distance travelled) getting there

(a) (2 points) Describe how we can model this problem as a graph (should it be directed or undirected, weighted or unweighted, what should be the vertices and what should be edges).

(b) (1 point) Describe what algorithm we should apply to solve the problem Layton is facing efficiently.

21. Consider the following graph $G = (V, E)$:



(a) (2 points) Give a valid order of the nodes corresponding to a BFS traversal starting from node A.

(b) (2 points) Now draw a different graph $G' = (V, E')$ so that the answer you gave to (a) is a valid DFS traversal of G' starting from node A in G'

22. A clique in a graph $G = (V, E)$ is a subset of nodes $V' \subseteq V$ such that for all $u, v \in V'$ there is an edge $\{u, v\} \in E$. Now consider the following two problems:

- CLIQUE-1: Given a graph $G = (V, E)$ is there a a clique of size $|V| - 1$?
- HALFCLIQUE: Given a graph $G = (V, E)$ is there a a clique of size $|V|/2$?

(a) (2 points) Show that both problems are in NP.

(b) (3 points) One of the problems can also be shown to be in P. Show this.

End of the exam