State Final Examination (Sample Questions)

Fall 2018

1 Automata (3 points)

1. Give a definition for a (generative) grammar, the language generated by a grammar and Chomsky hierarchy of grammars.
2. Construct a grammar generating the following language

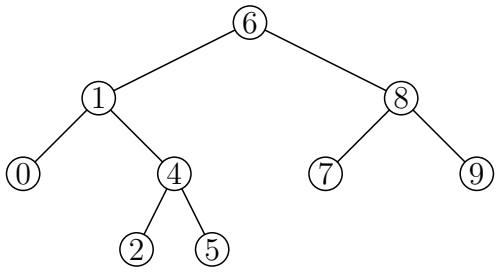
$$L = \{uvu^R \mid u \in \{a, b\}^*, v \in \{a\}^*\},$$

where u^R denotes the mirror image of the word u.

3. Show where in Chomsky hierarchy is the language L. That is, determine the weakest class of Chomsky hierarchy containing the language L and show that no grammar from a weaker class of Chomsky hierarchy can generate L.

2 Algorithms and Data Structures (3 points)

1. Define AVL trees and their basic properties (invariants).
2. Describe how to insert an element with value 3 to the following AVL tree (preserving all the invariants). Explain each step.
3. Discuss the time complexity of (1) the search for a value and (2) the insert operation, especially compared to the basic variant of a binary search tree.



3 Databases (3 points)

1. Consider the following relational schema: AUTHOR(ID, Name, DateOfBirth), BOOK(Title, NumberOfPages, AuthorID).

   Explain the term "integrity constraint". Add the missing integrity constraints to the presented schema.

   Write the following query using SQL: return the names of authors who wrote at least ten books with at least one hundred pages.

2. Explain the term "functional dependency between attributes". Extend one of the tables in the presented schema such that the table is not in the third normal form.

4 Programming Languages (3 points)

Choose an arbitrary mainstream, object-oriented, statically typed programming language (C++, C#, or Java), denote your selection in your answer. Then:

1. Implement/declare a suitable object-oriented interface (using interfaces or abstract classes) for non-oriented graphs $G = (V, E)$ in their most common form (each edge is between exactly two vertices, there is at most one edge between each pair of vertices). The main motivation is to define a common grounds for different graph implementations (representing the graph as list of neighbours, as matrix of incidence, ...), so that a programmer of graph algorithms (e.g., BFS, minimal spanning tree, ...) can write the code only once and it will still work with various graph representations.

   The interface should cover 3 basic entities (*vertex*, *edge*, and *graph*). The *vertex* provides access to all adjacent edges, the *edge* provides access to its two adjacent vertices, the *graph* provides access to the list of all vertices and list of all edges. The interface provides read-only access to this data – i.e., the graph structure is immutable.

   Each vertex and each edge has a *tag*. Vertex tag may indicate whether the vertex has been visited, for instance. Edge tag may indicate its weight. In general, different tag types may be suitable for different applications, however, a particular graph instance has fixed data types for vertex tags and for edge tags. The interface should be sufficiently generic so that the programmer may choose an arbitrary vertex tag and edge tag types for the graph instance. Furthermore, the interface will provide means to read and write the tag of each vertex and edge.

2. Consider an implementation of your interfaces where vertex tags indicate the membership of the vertex in a particular connected component of the graph ((type `int`, zero-based index) and edge tags represent their lengths (type `float`, or similar type in your language of choice). The edge tags are initially set, the vertex tags must be computed by you.

   Write the body of a function, using your interface, which will receive a graph and a real number r as arguments. Your code will compute the connected components of the graph and set the vertex membership information in each vertex tag (actual values in the tags are not important, the important thing is that vertices in the same component have the same tag). Edges which are longer than r are ignored when computing the components (i.e., as if they were not present in the graph at all).

5 Locks (3 points)

For you convenience, we start with a simplified overview of the C# language parallelism concepts sufficient to solve this question (however, if you are an experienced C# programmer and understand the C# language concepts more accurately than presented here, you can rely on your understanding instead):

Every `object` class instance contains exactly one unique lock instance. This lock primitive can be locked using the `lock` statement (see line 13 in the code below) – it is a typical lock implementation with support for recursive locking and with passive waiting for lock release. Note the opening curly brace immediately following the `lock` command is the actual lock request (to acquire the lock – see line 13), while the closing pair curly brace of the same block is the lock release (unlock) operation (see line 18).

The `Parallel.For(int fromInclusive, int toExclusive, Action<int> body)` method creates several new threads – the number of the new threads equals the number of logical processors on the target system. These worker threads then call the method passed as the `body` argument of `Parallel.For` (in the code below the worker threads call the `ProcessFilename` method) so that for every `i` from the <`fromInclusive`, `toExclusive`) interval, the `body` method is called exactly once. Which of the worker threads will call the `body` method for a specific value of `i` is not defined, the order in which the specific `i` values are used is not defined either. The `Parallel.For` method will return to the calling thread after all worker threads have finished all calls to the `body` method for all values of `i` from the defined interval.

Assume the following class written in the C# language:

```
1  class JpegCounter {
2          public JpegCounter(string[] filenames) { _filenames = filenames; }
3
4          string[] _filenames;
5          int _count;
6          object _globalLock = new object();
7
8          public int CountJpegs() {
9                  _count = 0; Parallel.For(0, _filenames.Length, ProcessFilename); return _count;
```

```
10          }
11
12          void ProcessFilename(int i) {
13                  lock (_globalLock) {
14                          string ext = Path.GetExtension(_filenames[i]).ToUpper();
15                          if (ext == ".JPG" || ext == ".JPEG") {
16                                  _count++;
17                          }
18                  }
19          }
20  }
```

1. In a multithreaded program context, briefly explain what a race condition is.
2. Decide and explain whether the use of the `_globalLock` variable in the code above is necessary for the correctness, or, if omitting may enable a race condition during the execution of the code. Is it possible that the `ProcessFilename` method without using locks (i.e. after deletion of lines 6, 13, and 18) would return a different value than the version of the method shown here above even if we provide the same input to the `JpegCounter` class?
3. Assume you are developing a program that will create a new `JpegCounter` class instance, and will provide it with 10,000,000 (ten million) file names, where roughly 10 percent of the names have the `.jpg` or `.jpeg` extension. Further assume a typical dual core processor used to execute our program. Estimate how many times the provided `CountJpegs` method implementation will be faster (or slower) than its sequential equivalent (i.e. a variant of the method with a simple `for` cycle and just sequentially doing all the calls of the `ProcessFilename` method in the `for` cycle directly in the context of the calling thread). Explain if, or ideally how, it is possible to modify the `ProcessFilename` method so that its parallel version is faster than the one provided in the code above.

6 Networking (3 points)

1. Describe (briefly) the purpose of IP protocol from the perspective of layered architecture. Which services it offers to the layer(s) above and which services it requires from the layer(s) below?
2. IPv4 protocol natively supports datagram fragmentation. Explain when such fragmentation may occur and how it can be prevented.
3. Consider application of IP over Ethernet line (1000BASE-T). Explain the principles being used to translate IPv4 and IPv6 addresses to MAC addresses for local datagram delivery.

7 Duality of LP (3 body)

1. State the strong duality theorem for linear programming.
2. We are given an undirected graph $G = (V, E)$ and its vertices $s_1, s_2, t_1, t_2 \in V$. For $i = 1, 2$, let P_i be the set of all paths between vertices s_i and t_i in G, and let $P = P_1 \cup P_2$. Consider the following linear program LP1 (there is a variable x_p for every $p \in P$):

$$\begin{aligned} & \max \sum_{p \in P} x_p \\ \text{s.t.} \quad & \sum_{p: e \in p} x_p \leq 1 \quad \text{for} \quad \forall e \in E \\ & x_p \geq 0 \quad \text{for} \quad \forall p \in P\,. \end{aligned}$$

   Formulate the dual program (use y for the vector of dual variables).
3. Consider the graph $G = (V, E)$ with $V = \{s_1, s_2, t_1, t_2, a, b, c, d\}$ and

$$E = \{(s_1, a), (s_2, a), (s_1, b), (s_2, b), (a, c), (b, d), (t_1, c), (t_2, c), (t_1, d), (t_2, d)\}.$$

   If it exists, find an optimal solution of the (primal) linear program LP1 above, and, using the strong duality theorem, prove that the solution is optimal.

8 Computational Linguistics: Morphological, Syntactic and Semantic Analysis of Natural Languages (specialization question – 3 points)

1. Explain the notion of "ontology" in the processing of natural language semantics.
2. Describe the semantic network Wordnet.
3. Explain the notion of "anaphora" and list basic categories of anaphora in texts.

9 Computational Linguistics: Formal Languages and Automata (specialization question – 3 points)

The most popular tool for morphological analysis in late eighties had been the so-called Two-level morphology of Kartunnen and Koskenniemi.

1. Identify the two levels of representation mentioned in the name of the theory.
2. Which basic formal framework has been used for morphology processing in the Two-level morphology?
3. List at least two basic ideas of this mechanism.

10 Computational Linguistics: Basic Formalisms for Description of a Natural Language (specialization question – 3 points)

1. List and describe three basic components of Chomsky's Transformational grammar.
2. Explain the notion of transformation in Chomsky's Transformational grammar – which two parts define transformations?
3. Why it is necessary to use typed feature structures in unification grammars?

11 Databases and Web: Validity of XML data (specialization question – 3 points)

1. Explain the term "valid XML schema".
2. Provide at least 3 differences between languages DTD and XML Schema.
3. Using XML Schema describe element address which contains subelements street, number, city, and postcode in any order. Can such a schema be expressed also in DTD? If so, how. If not, why.

12 Databases and Web: XSLT (specialization question – 3 points)

1. Briefly describe how the XSLT processor works. Explain the term "implicit XSLT template".
2. What is the output of application of an empty XSLT script on a document containing only element `<h1 c="blue">Hello world!</h1>`?
3. Provide an XSLT script whose output is a list of names and values of all attributes of any input XML document.

13 Databases and Web: Relational completeness, joining of tables and SQL (specialization question – 3 points)

1. Using a suitable simple example explain the difference between inner and outer join of SQL tables.

2. Explain the terms "relation" and "relationally complete language". Is the SQL relationally complete? Why?

14 Orthonormal basis (3 body)

1. Define the notion of orthonormal basis.
2. Let z_1, z_2, z_3, z_4 be an orthonormal basis of $\mathbb{R}^4$ and let $u = z_1 - z_2 + 2z_3 - 2z_4$.
   - Compute $\|u\|$.
   - Decide whether the vector $v = z_1 - z_2 + 3z_3 + 4z_4$ is orthogonal to the vector u.
   - Find the orthogonal projection of u onto the orthogonal complement of the set $\{z_1, z_2\}$.

15 Determinants and the similarity of matrices (3 body)

1. Define the notion of determinant of a matrix. Decide whether the equality $\det(A + B) = \det(A) + \det(B)$ holds in general (prove or disprove).
2. Compute the determinant of the matrix
$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$
3. Define the notion of similarity of matrices. Prove that similar matrices have equal determinants.

16 Regular matrices (3 body)

1. Define the notion of regular matrix. Prove that the set of real regular matrices of order n and the operation of matrix multiplication form a group.
2. Find the number of regular matrices in $\mathbb{Z}_2^{3\times 3}$.

17 Taylor series (3 body)

Give Taylor series of the function $\log(1 + x)$, centered at $x = 0$.

Compute the limit
$$\lim_{x\to+\infty} \left(x^2(\log(x+1) - \log x) - x\right).$$

18 Integration (3 body)

State the formula for integration (i.e., finding antiderivative) by parts.

Use it to find
$$\int e^x \sin x\,.$$

19 Graphs (3 body)

1. Define the notion of trail and closed trail in a graph.
2. State the neccessary and suficient condition for the existence of a closed trail containing all edges of a given graph.
3. Show that every graph has an orientation such that the in-degree and the out-degree of each vertex differs by at most one.

20 Generating functions (3 body)

Let $a_0, a_1, a_2 \ldots$ be a sequence of numbers defined by the following system of recurrences:

$$\begin{aligned} a_0 &= 2 \\ a_1 &= -1 \\ a_n &= 3a_{n-1} - 2a_{n-2} \text{ for } n \geq 2. \end{aligned}$$

Find the generating function of this sequence. Express the result as a formula in closed form, i. e., without infinite sums. (You are not required to find a closed-form expression for the a_n terms themselves; an expression for the generation function is enough.)

21 Normal subgroups (3 body)

Define a normal subgroup of a group.

For a positive integer n, let S_n be the symmetric group of all permutations on $\{1, \ldots, n\}$ and let A_n be the set of permutations in S_n which have sign (a.k.a. signum) equal to 1. Prove that A_n is a subgroup of S_n and prove or disprove that A_n is a normal subgroup of S_n.

22 Logic (3 points)

Write down the following two statements as formulas of predicate logic in a suitably chosen language (with unary predicates for "*wants*", "*looks for a way*", "*seeks excuses*").

1. *He who wants (to do something), looks for a way, he who does not want (to do something), seeks excuses.*
2. *He who does not look for a way, seeks excuses.*

Prove in some formal proof system (tableaux method, resolution method, Hilbert calculus) that the latter statement follows from the former.