University of Freiburg
Department of Computer Science
Prof. Dr. F. Kuhn

Exam Theoretical Computer Science - Bridging Course

Friday, September 07, 2018, 10:00-12:00

Name: .....

Matriculation No.: .....

Signature: .....

Do not open or turn until told so by the supervisor!

- Write your **name** and **matriculation number** on this page and sign the document!
- Write your name on **all sheets**!
- Your **signature** confirms that you feel physically and mentally able to write the exam and that you have answered all questions without any help.
- Write legibly and only use a pen (ink or ball point). Do **not use red**! Do **not use a pencil**!
- This is an **open book exam** therefore printed or hand-written material is allowed.
- However, **no electronic devices** are allowed.
- There are **eight tasks** (with several sub-tasks each) and there is a **total of 100 points**.
- **35 points are sufficient** in order to pass the exam. **70 points** are sufficient to get the best mark.
- Only **one solution per task** is considered! Make sure to strike out alternative solutions, otherwise the one yielding the minimal number of points is considered.
- **Detailed steps** might help you to get more points in case your final result is incorrect.
- The keywords **Show...** or **Prove...** indicate that you need to prove or explain your answer carefully.
- The keywords **Give...** or **State...** indicate that you only need to provide a plain answer.
- You may use information given in a **Hint** without explaining them.
- **Read each task thoroughly** and make sure you understand what is expected from you.
- **Raise your hand** if you have a question regarding the formulation of a task.
- **Use the space below each task and the back of the sheet for your solution.** The last two sheets of this exam are blank and can be used for solutions. If you need additional sheets, raise your hand.

Question	1	2	3	4	5	6	7	8	**Total**
Points									
Maximum	9	16	18	12	10	12	10	13	100

Task 1: Mathematical Proofs (10 Points)

(a) By mathematical induction, prove that for any positive integer n, $6^n - 1$ is divisible by 5. *(5 Points)*

Remark: Integer k is divisible by integer ℓ if there exists an integer s such that $k = s \cdot \ell$.

(b) Prove that every tree of size at least 2 is a bipartite graph. *(5 Points)*

Remark: A graph is bipartite if its set of vertices can be divided into two disjoint and independent sets. An independent set of vertices is a set of vertices in a graph, no two of which are adjacent.

Hint: You can use the fact that every tree T contains a vertex v of degree 1. Moreover, if one removes v from T, the remaining graph is still a tree.

Sample Solution

(a) Induction Base: For $n = 1$: $6^1 - 1 = 5 = 1 \cdot 5$.

Induction Hypothesis: Let us assume that $6^n - 1$ is divisible by 5.

Induction Step: $6^{n+1} - 1 = 6(6^n) - 1 = 6(6^n - 1) - 1 + 6 = 6(6^n - 1) + 5$.

As you see both terms $6(6^n - 1)$ and 5 are divisible by 5 which concludes the proof.

(b) Idea: Add nodes with even distance to the root to one set and nodes with odd distance to the other set.

Proof by induction: For sure the statement holds for trees with two nodes. Now assume the statement holds for all trees with n nodes. Let T be a tree with $n+1$ nodes and u a node with degree one, i.e., there is a node v such that v is the only neighbor of u. Let $T' := T \setminus \{u\}$. T' is a tree with n nodes, so by assumption there is a partition of T' in disjoint independent sets U and W. If $v \in U$, add u to W (and vice versa) and we obtain a partition of T in two disjoint independent sets.

Task 2: DFAs, NFAs, Regular Expressions (11 Points)

Consider the following language $\mathcal{L}$ over alphabet $\mathcal{A} = \{a, b\}$.

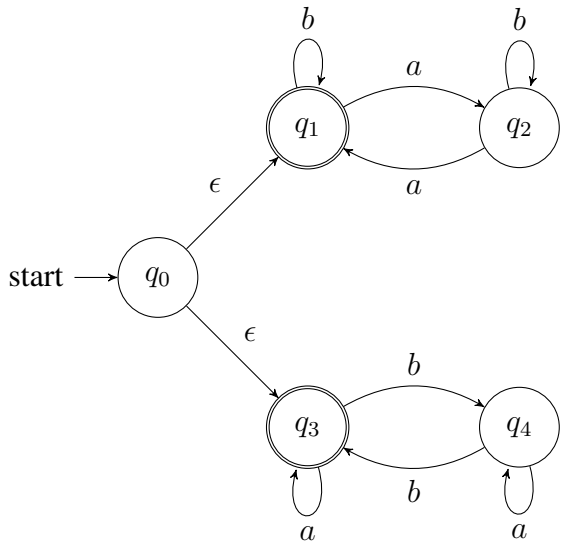
Language $\mathcal{L}$ contains all strings in which at least one of the symbols a or b occurs an even number of times. For example, $aba \in \mathcal{L}$ but $ab \notin \mathcal{L}$.

(a) Draw an NFA that recognizes $\mathcal{L}$. *(6 Points)*

(b) Provide a regular expression that recognizes $\mathcal{L}$. *(5 Points)*

Sample Solution

(a)



(b) $a^* \cup b^* \cup (b^*ab^*ab^*)^* \cup (a^*ba^*ba^*)^*$

Task 3: Context-Free Languages (20 Points)

Consider the following language over alphabet $\{a, b, c\}$.

$L = \{a^* w c^k | w \in \{a, b\}^*, \text{ and } k \text{ is the number of } a\text{'s in } w\}$

(a) Applying the pumping lemma, show that L is not a regular language. *(7 Points)*

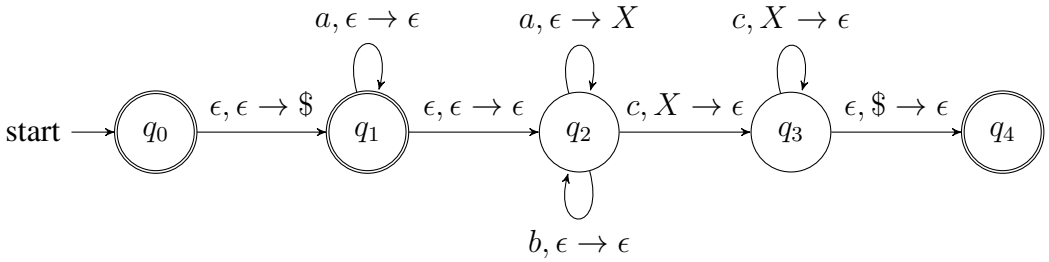
(b) Draw a pushdown automata that recognizes L. *(8 Points)*

(c) Give a context-free grammar that recognizes L. *(5 Points)*

Sample Solution

(a) Let $p \geq 1$. Choose $s = a^p c^p \in L$. Then in any partition $s = xyz$ with $|y| > 0$ and $|xy| \leq p$, y only contains as. Then xy^0z is a string with more cs than as and therefore can not be in L.

(b)



(c)

$$S_0 \to AC$$
$$A \to aA \mid \epsilon$$
$$C \to BaBCc \mid \epsilon$$
$$B \to Bb \mid \epsilon$$

Task 4: Regular and Context-Free Languages (14 Points)

(a) Let $L_1, L_2, L_3, \ldots$ be an infinte sequence of regular languages, each of which is defined over a common input alphabet $\Sigma = \{a, b\}$. Let $L = \bigcup_{i=1}^{\infty} L_i$ be the infinte union of $L_1, L_2, L_3, \ldots$.

Is it always the case that L is a regular language? If your answer is YES, give a proof. If your answer is NO, give a counterexample. Explain your answer. *(7 Points)*

(b) Consider alphabet $\Sigma = \{a, b\}$ and arbitrary strings $w_1, w_2, ..., w_{100}$ over Σ. Then, let $L_1, L_2,$ and L_3 be languages defined over Σ, where

- L_1 consists of all possible strings over Σ except the strings $w_1, w_2, ..., w_{100}$.
- L_2 is recognized by an NFA; and
- L_3 is recognized by a PDA.

Prove that $(L_1 \cap L_2) \circ L_3$ is a context-free language.

Hint: First show that L_1 and L_2 are regular. *(7 Points)*

Sample Solution

(a) The answer is NO. For $i \in \mathbb{N}$, let $L_i = \{a^i b^i\}$. Each L_i is regular because it is finite, but $\bigcup_{i=1}^{\infty} L_i = \{a^n b^n \mid n \in \mathbb{N}\}$, which is not regular.

(b) $\{w_1, \ldots, w_{100}\}$ is finite and therefore regular. L_1 is the complement of this language and therefore also regular. L_2 is regular as it is recognized by an NFA. As the intersection of regular languages is regular, $L_1 \cap L_2$ is regular and therefore context-free. L_3 is context-free as it is recognized by a PDA. As the concatenation of context-free languages are context-free, $(L_1 \cap L_2) \circ L_3$ is context-free.

Task 5: Turing machines (10 Points)

Consider alphabet $A = \{1, 2, \ldots, 9\}$. We call a string S over A a *blue* string, if and only if the string consisting of the odd-positioned symbols in S is the reverse of the string consisting of the even-positioned symbols in S. For example $S = 14233241$ is a blue string since the substring of the odd-positioned symbols is 1234 which is the reverse of the substring of the even-positioned symbols, i.e., 4321.

Design a Turing machine which accepts all blue strings over A. You do not need to provide a formal description of the Turing machine but your description has to be detailed enough to explain every possible step of a computation.

Sample Solution

On input S, first go through all symbols. If it is an odd number, reject. Else repeat the following:

Go left until you reach the first unmarked symbol, mark it, go right to the last unmarked symbol, mark it, and compare both symbols. If they are different, reject.

Accept if all symbols are marked.

Task 6: Undecidability (10 Points)

Fix an enumeration of all Turing machines (that have input alphabet Σ): $\langle M_1 \rangle, \langle M_2 \rangle, \langle M_3 \rangle, \ldots$.

Fix also an enumeration of all words over Σ: $w_1, w_2, w_3, \ldots$.

Prove that language $L = \{w \in \Sigma^* \mid w = w_i, \text{ for some } i, \text{ and } M_i \text{ does not accept } w_i\}$ is not Turing recognizable.

Hint: Try to find a contradiction to the existence of a Turing machine that recognizes L.

Sample Solution

Assume M is a turing machine recognizing L. Then there is an i such that $M = M_i$.

Assume M accepts w_i. One the one hand this implies $w_i \in L$ (as M recognizes L), on the other hand it implies $w_i \notin L$ (by the definition of L), leading to a contradiction.

Now assume M does not accept w_i. One the one hand this implies $w_i \notin L$ (as M recognizes L), on the other hand it implies $w_i \in L$ (by the definition of L), leading to a contradiction.

So in either case we get a contradiciton. Therefore such a TM can not exist.

Task 7: Complexity Theory (11 Points)

(a) Show that the following language is in class $\mathcal{P}$. *(6 Points)*

9-CYCLE $= \{\langle G \rangle \mid G$ is a graph and contains a cycle of length $9\}$

(b) Is the following statement true or false? Prove your answer. *(5 Points)*

Let L_1 and L_2 be languages in $\mathcal{NP}$, and assume $\mathcal{P} \neq \mathcal{NP}$. Then, if $L_1 \leq_P L_2$ and $L_2 \leq_P L_1$, both L_1 and L_2 are NP-complete.

Sample Solution

(a) First check if $\langle G \rangle$ is a valid graph representation (requiring linear time). Then, for all 9-tuples of nodes, check if they form a cycle. For one such tuple $(x_1, \ldots, x_9)$, you have to check for all $i = 1, \ldots, 9$ if (x_i, x_{i+1}) and (x_9, x_1) is an edge. These are 9 checks to make, each taking at most linear time (go through the list of edges or the adjacency list of a node, depending on the graph representation). So checking one 9-tuple for a cycle can be done in $\mathcal{O}(n)$. There are n^9 9-tuples to check, so checking a graph for a 9-cycle takes $\mathcal{O}(n^{10})$ steps.

(b) If we choose L_1 and L_2 from $\mathcal{P} \setminus \{\emptyset, \Sigma_1^*, \Sigma_2^*\}$, then $L_1 \leq_P L_2$ and $L_2 \leq_P L_1$. But from $\mathcal{P} \neq \mathcal{NP}$ follows that no language in $\mathcal{P}$ is NP-hard, therefore neither L_1 nor L_2 are NP-complete.

Task 8: Logic (14 Points)

(a) Consider the following propositional formula

$$\psi := (x \to y \vee z) \wedge (y \to \neg x) \wedge (x \wedge z \to y) \; .$$

(1) Transfer ψ into an equivalent formula in conjunctive normal form (CNF). *(2 Points)*

(2) Use the resolution calculus to show that ψ entails $\neg x$. *(6 Points)*

(b) Consider the following first order logical formulae

$$\begin{aligned} \varphi_1 &:= \forall x \; \neg R(x, x) \\ \varphi_2 &:= \forall x, y, z \; (R(x, y) \wedge R(y, z) \to R(x, z)) \\ \varphi_3 &:= \exists x \forall y \; (x \neq y \to R(x, y)) \end{aligned}$$

where x, y are variable symbols and R is a binary predicate. Give an interpretation

(1) I_1 which is a model of $\varphi_1 \wedge \varphi_2 \wedge \varphi_3$. *(3 Points)*

(2) I_2 which is a model of $\varphi_1 \wedge \varphi_2 \wedge \neg\varphi_3$. *(3 Points)*

***Remark**: No proof required.*

Sample Solution

(a) (1) $(\neg x \vee y \vee z) \wedge (\neg y \vee \neg x) \wedge (\neg x \vee \neg z \vee y)$

(2) ψ is equivalent to the knowledge base KB$= \{\{\neg x, y, z\}, \{\neg y, \neg x\}, \{\neg x, \neg z, y\}\}$. To prove that ψ entails $\neg x$, prove KB$\cup\{x\} \vdash_{\mathbf{R}} \square$.

$$\begin{aligned} \{x\}, \{\neg x, y, z\} &\vdash_{\mathbf{R}} \{y, z\} \\ \{y, z\}, \{\neg y, \neg x\} &\vdash_{\mathbf{R}} \{z, \neg x\} \\ \{\neg y, \neg x\}, \{\neg x, \neg z, y\} &\vdash_{\mathbf{R}} \{\neg x, \neg z\} \\ \{z, \neg x\}, \{\neg z, \neg x\} &\vdash_{\mathbf{R}} \{\neg x\} \\ \{x\}, \{\neg x\} &\vdash_{\mathbf{R}} \square \end{aligned}$$

(b) (1) $I_1 := \langle \mathbb{N}, R^{I_1} \rangle$ where $R^{I_1}(x, y) :\Leftrightarrow x <_{\mathbb{N}} y$.

(2) $I_2 := \langle \mathbb{Z}, R^{I_2} \rangle$ where $R^{I_2}(x, y) :\Leftrightarrow x <_{\mathbb{Z}} y$.