Chalmers University of Technology

Computer Science and Engineering
Elad Michael Schiller

2020/10/29

Written exam in EDA387/DIT663 Computer Networks 2020-10-29. Exam time: 4 hours.

A remote exam via Canvas.

Examiner: Elad Michael Schiller, phone: 073-6439754

Credits:	30-38	39-47	48-Max
Grade:	3	4	5
Grade (GU)	G	G	VG

1. Exam problems that require text-based answers should be written and submitted in a text document, e.g. Word or PDF.
   - Create one text document for each such exam problem.
   - Name your text document Question_YY. *Example:* Question_01.pdf
   - Submit your answers by uploading the text documents via Canvas before the due date/time.
2. Exam problem solutions involving calculations, figures, diagrams, etc. should be solved on paper as in a normal exam.
   - Make sure that each paper is clearly marked with your name, exam problem number, and the page number.
   - Scan or photograph your solutions. Make sure to have good lighting and preferably use a document scanning app, e.g. even CamScanner or Genius Scan.
   - Name your image files Question_YY_Page_XX. *Example:* Question_01_Page_02.jpg.
3. If you want, you can combine images for the same problem into a single document (e.g. Word or PDF) named Question_YY.
4. Submit your solutions by uploading the image files or documents via Canvas before the due date/time.
5. The answer must be written in English (also for Swedish students). Use proper grammar and punctuation.
6. All answers need to be motivated, unless otherwise stated. Correct answers without motivation or with wrong motivation will not be given full credit. Answer concisely but explain all reasoning. Draw figures and diagrams when appropriate. Write clearly. Unreadable or hard-to-read handwriting will not be given any credit.
7. Do not hand in anything that is not part of the solution.
8. Always write your answers in your own words, using your own diagram, etc. In case you have to use the words of others, e.g., a formal definition, please make sure to use quotation marks and clearly identify the source. Aboutsltly do not collaborate in any way or form with other people during the entire exam time.

Note that many kinds of answers were accepted to different queations.

Question 1: Networking and design criteria (32 points in total)

1.a (6 points) This part considers software-defined networks (SDNs) and routing at the Internet core. Specifically, we wonder whether current SDN technology can replace the routing infrastructure at the Internet core.

1.a.i (2 pt) In your opinion, what might be the benefits of such a replacement? Please provide a clear explanation written only in your own words.

ANSWER: a broad set of answers might make sense. For example, it is easier to manage SDN than normal routers.

1.a.ii (2 pt) Does such a replacement might violate one of the design principles that allowed the development of the Internet? Please provide a clear explanation written only in your own words.

ANSWER: several answers can be accepted. For example, is the end to end principle which talk about improving the network only at its end.

1.a.iii (2 pt) Do you see any disadvantage with the idea of this replacement? Please provide a clear explanation written only in your own words.

ANSWER: the cost of replacing the entire Internet infrastructure is huge.

1.b (8 points) Initialization of end-to-end communications. In class we learned that there is no self-stabilizing protocol that can initialize a data link, see Chapter 3.1 in the textbook.

1.b.i (2 pt) Does the same impossibility holds for transport layer algorithms? Please provide a clear explanation written only in your own words.

Yes, since one can use FIFO queue for modeling both the 2nd and 4th layers.

1.b.ii (4 pt) How does TCP/IP initialize the communication between any two peers? Please provide a clear explanation written only in your own words. You can use a diagram, if you want, as long as you draw it yourself.

In this answer, there is a need to describe the three/four-way handshakes as well as the idea behind the TIME-WAIT.

1.b.iii (2 pt) Is there a contradiction between the fact that TCP/IP does run the initialization procedure and the impossibility in Chapter 3.1? Please provide a clear explanation written only in your own words.

There is no contradiction because TCP/IP and the studied impossibility use different models. The former assumes that old packets disappear from the network within TIME-WAIT time units whereas the latter considers an asynchronous network with unbounded link capacity.

1.c (4 points) Imagine that all the installations of Domain Name System (DNS) were modified to a version that allows no RR cashing at all.

1.c.i (2 pt) In your opinion, what will be the result of this event? Is there a danger that the DNS will not work properly? Please provide a clear explanation written only in your own words.

This event will result in a greater system overload since cashing is an essential element in voiding unnecessary requests. The idea is that answers to recent requests do not change most of the time. Therefore, without cashing, many more requests will be sent --- possibly significantly more than the currently available resources.

1.c.ii (2 pt) Suppose that the opposite happens and all RR are cashed for an extremely long time. Is there a danger that the DNS will not work properly? Please provide a clear explanation written only in your own words.

This event will make it hard to update RR. As a result, for example, it would be hard to resolve the address of services that their hosts were relocated.

1 d (6 points) This question considers what is possible and not possible in computer networks.
1.d.i (3 pt) It is impossible to design deterministic self-stabilizing algorithms for vertex coloring in synchronous and anonymous networks (in which nodes have no globally unique identifiers and there is no distinguished node)? Please provide a clear explanation written only in your own words.

No. The proof is along the lines of the impossibility of self-stabilizing leader election.

1.d.ii (3 pt) Let *Alg* be a self-stabilizing algorithm works under the assumption of a central demon model in networks that have globally unique identifiers. There is always a way to make *Alg* work in asynchronous (under the assumption of a distributed demon model) and anonymous networks (in which nodes have no globally unique identifiers and there is no distinguished node). Please provide a clear explanation written only in your own words.

No. The proof is along the lines of the impossibility of self-stabilizing token circulation.

1.e (4 points)
1.e.i (2 pt) In what sense a solution for pseudo-self-stabilizing systems is inferior to a self-stabilizing one? Please provide a clear explanation written only in your own words.

It does not guarantee recovery within a bounded (or finite) time, as self-stabilizing systems do.

1.e.ii (2 pt) Should we completely avoid the use of any pseudo-self-stabilization? Please provide a clear explanation written only in your own words.

No, because sometimes there is no self-stabilizing solution but there is a pseudo-self-stabilizing one.

Question 2: self-stabilizing any-cast routing at layer two (18 points in total)

Consider a computer network that its topology is of a general graph, $G:=(P, E)$, where $E \subseteq PxP$. Assume that the network includes a set, S, of selected nodes that store some important information. We assume that the data link layer is to provide this information for the higher layer, and thus, no built-in routing services are available.

This question considers the problem of self-stabilizing routing of packets between nodes that are in S and ones that are in $P\backslash S$. It is required that node $p_i \in P\backslash S$ corresponds only with a single node in S using one of the shortest paths between them. Assume that each node is aware of whether it is in S or $P\backslash S$, but the network has no globally unique identifiers, *i.e.*, a node can locally identify their direct neighbors. Your task is to provide the entire routing scheme. Namely, you need to:

- Task 2.a Show how to construct routes between p_i and p_j.
- Task 2.b Show how node $p_i \in S$ can name the nodes $p_j \in P\backslash S$ that suppose to correspond with it.
- Task 2.c Show how to forward packets along these routes.

For each of the tasks above, the following questions are about your opinion on whether there an algorithm that we studied in class that can fit without any modification to solve task 2.x. In case you do, please solve the following blue sequence of questions. In case you do not, solve the green sequence.

2.x.i (2 points) What is the exact name of the algorithm that solves task 2.x? Which chapter in the book?
2.x.ii (2 points) Does this algorithm solves another problem in another context, model, or set of assumptions? Why there is still no need for modifications when solving task 2.x? Please explain clearly and justify your answer. You are welcome to also include small diagrams.
2.x.iii (2 points) Please provide the cost of the (perhaps modified) solution with respect to the amount of memory each node should have, the number and size of communication registers as well as stabilization time.

2.x.i (2 points) Please provide clear and exact pseudocode for a protocol that solves task 2.x.
2.x.ii (2 points) Please provide the key ideas of the correctness proof for your proposal when demonstrating convergence and closure.
2.x.iii (2 points) Please provide the cost of your solution with respect to the amount of memory each node should have, the number and size of communication registers as well as stabilization time.

For clarity, please note that your answer to this question should include the following items.
2.a.i (2 pt) The self-stabilazing algorithm for first BFS/spanning-tree construction in Chapter 2.5.
2.a.ii (2 pt) The original problem considers a single distinguished processor. Here, all the processors in S are distinguished. The rest of the answer needs to show that the same algorithm works also for this case. The idea is to rewrite the correctness proof and show each node selects a parent that is the closest to in S. Moreover, in case there is more than just one, the selection of the first one works along the same lines as in the case of the construction of the first BFS.
2.a.iii (2 pt) Same as the algorithm in Chapter 2.5.

2.b.i (2 pt) Self-Stabilizing Ranking: Converting an Id-based System to a Special-processor System. Chapter 4.3.
2.b.ii (2 pt) The problem and the settings are identical.
2.b.iii (2 pt) Same as the algorithm in Chapter 4.3.

2.c.i (2 pt) Routing to a root in S is always to address 0. Thus, the code should simply forward the packet up the tree. Any other address should be forwarded down the tree. Note that the naming algorithm of task 2.b provides internals --- is the address falls within an interval that is associated with a child in the tree, then one can route via this child.
2.c.ii (2 pt) The proof focuses on the properties of the intervals and shows that indeed the packet can be routed via this child. The proof is demonstrated using the construction of the naming algorithm for task 2.b.
2.c.ii (2 pt) No additional memory is needed since all the processing is on the fly. Same for stabilisation time.

.....

Question 3: self-stabilizing traffic reduction in anonymous networks (20 points in total)

A new kind of network was invented in which computers mostly send information to few gateways in the network and they rarely receive information. Since the network scale is huge, there is a need to preserve traffic usage. The following communication scheme was proposed. A subset of nodes, *S*, in the network is selected so that every node is either in *S* or has a direct neighbor in *S*. Whenever a node wishes to share information with the gateway, it sends it to one of the nodes in *S*. The latter aggregates a number of such messages (arriving possibly from different nodes).

The problem: Your task is to design a self-stabilizing network protocol for selecting *S* in a minimal fashion. By minimal fashion, we mean that the removal of two nodes from *S*, always violates the guarantee that every node is either in *S* or has a direct neighbor in *S*. Moreover, the task requires that as long as there is no topological change to the network, the set *S* does not change.

Assumptions: you can assume that the network topology is one of the following: general graph, undirected ring, or tree. Please assume that the network is anonymous (in which nodes have no globally unique identifiers and there is no distinguished node). That is, the tree has no predefined root, the ring has no distinguished node, p_0, etc.

3.a (2 points) Please select the network topology and define the set of legal executions. That is, specify the task requirements concise and precise way.

Let us consider the topology of a tree. During a legal execution, every configuration encodes a minimal fashion solution for vertex cover, where S is the set of covering nodes. When grading the exam, also the undirected ring and general topology, but then a solving without assuming a centered demon becomes harder.

3.b (4 points) Please provide clear and exact pseudocode for the protocol.

The idea is to rely on the algorithm that was considered in lab 2.4. The center node (or pair) is always in S. Then, starting from the center and towards the leafs, the algorithm adds to S every fourth node. Moreover, a leaf node decides to add itself in case it does not have neighbor in S. For this, the center node (or pair) needs to propagate whether their h-value is even or odd. All the other calculations can be done locally. (When grading the exam, also starting from the leaf was accepted.)

3.c (2 points) Does your solution uses the message-passing model or the shared-memory model? In case you have chosen the former, i.e., message-passing, how does your solution deal with packet failures, such as loss

or duplication? In case you have chosen the latter, i.e., shared-memory, in what way does it help to provide a simpler solution? Also, can you show how to transform your solution into a message-passing model?

Let us use shared-memory --- it makes life simpler (but also message passing can work but it requires more explanations).

3.d (6 points) Please proof the correctness of your proposed solution. That is, show that when starting from any system state (configuration), the network selects the nodes in *S* eventually.

The proof is based on the fair composition theorem. It shows that the construction is indeed a minimal vertex cover.

3.e (2 points) Can you provide a bound on the time that it takes your solution to recover from the last occurrence of a transient fault?

$O(r)$ asynchronous cycles, where r is the radius of the tree.

3.f (2 points) Does your correctness proof assume the model of central demon or distributed demon? In case you are using the former, i.e., central demon, explain how this assumption helps to avoid possible concurrency issues, if there are any. In case you are using the latter, i.e., distributed demon, explain how your solution and proof deals with possible concurrency issues, if there are any.

This solution does not require the use of a central demon.

3.g (2 points) In your own words (and only using your own words), what is the advantage of designing a self-stabilizing solution to this problem. For example, how does your self-stabilizing solution deal with topology changes?

This self-stabilizing solution guarantees automatic recovery from any transient fault, including, topology changes.