Chalmers University of Technology
Computer Science and Engineering
Elad Michael Schiller

2015-10-30

Written exam in EDA387/DIT663 Computer Networks 2015-10-30. Exam time: 4 hours.

Means allowed: Nothing except paper, pencil, pen and English - xx dictionary.

Examiner: Elad Michael Schiller, phone: 073-6439754
Note that student questions can be answered only by phone.

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

1. The answer must be written in English (even for Swedish students). Use proper grammar and punctuation.
2. All answers need to be motivated, unless otherwise stated. Correct answers without motivation or with wrong motivation will not be given full credit.
3. Answer concisely, but explain all reasoning. Draw figures and diagrams when appropriate.
4. Write clearly. Unreadable or hard-to-read handwriting will not be given any credit.
5. Do not use red ink.
6. Solve only one problem per page.
7. Sort and number pages by ascending problem order.
8. Anything written on the back of the pages will be ignored.
9. Do not hand in empty pages or multiple solutions to the same problem. Clearly cross out anything written that is not part of the solution.

Question 1: DNS Lab (6 points)

A user issued the following **dig**-command to find specific DNS information. The output of running the command is shown below.

```
C:\dig> dig mx ntnu.no @ns1.ntnu.no

; <<>> DiG 9.3.2 <<>> mx ntnu.no @ns1.ntnu.no
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1780
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

;; QUESTION SECTION:
;ntnu.no.                        IN      MX

;; ANSWER SECTION:
ntnu.no.                60       IN      MX      10 mx.ntnu.no.

;; AUTHORITY SECTION:
ntnu.no.                3600     IN      NS      ns2.ntnu.no.
ntnu.no.                3600     IN      NS      ns1.ntnu.no.

;; ADDITIONAL SECTION:
mx.ntnu.no.             600      IN      A       129.241.56.67
ns1.ntnu.no.            3600     IN      A       129.241.0.208
ns2.ntnu.no.            3600     IN      A       129.241.0.209

;; Query time: 32 msec
;; SERVER: 129.241.0.208#53(129.241.0.208)
;; WHEN: Mon Aug 17 11:52:11 2015
;; MSG SIZE  rcvd: 240
```

When answering the following sub-questions, please note that it is NOT enough with copy from the above output. Use DNS-terminology.

1a. (1 point) What did the user want to ask about? Explain by referring to the syntax of the command.

The user queried the authoritative name server `ns1.ntnu.no asking about the hostnames of the servers responsible for the electronic mail (RR type=MX Mail eXchanger) belonging to the domain ntnu.no.`

1b. (1 point) Explain the contents of the "QUESTION SECTION". What is the name of the object type that was queried?

The question section contains the domain name "`ntnu.no", the class "IN: InterNet" and the resource Record type "MX: Mail eXchanger" which the name of the object type. The requested value is the hostname of email server.`

1c. (2 points) To which DNS server (hostname and IP-address) was the query sent? Is it an authoritative server or not? Explain and point out how you are able to confirm that it is or not.
Yes it is. SERVER: ns1.ntnu.no 129.241.0.208 which is one of two authoritative name servers that are provided in the **AUTHORITY SECTION of the reply (RR type=NS). Also one can find in the flags: aa, that the answer is authoritative.**

1d. (2 points) Is there any answer in the reply? Describe in detail the information that the reply is providing to the user?

Yes there is answer as provided in the ANSWER SECTION. The answer is providing the hostname of the email server "**mx.ntnu.no" with preference value 10. The RR type is MX, the class "IN: InterNet" and TTL = 60 s. The reply is also providing the IPv4 addresses of the the email server and the two** authoritative name servers as an additional information in the **ADDITIONAL SECTION.**

Question 2: DNS (6 points)

Suppose that you are using the Chalmers network to connect your laptop to the Internet. Suppose also that you want to access the website www.tue.nl for the first time (not in cache). Explain **how and why** DNS will be involved immediately after entering the name of the site in your browser. The answer should, specifically and technically, explain the necessary operation, including:

- the interaction and communication between the different DNS resolvers and servers,
- the protocols and messages used, and
- the final outcome.

Please remember that you should use **DNS terminology.**

ANSWER: Please refer to the DNS-lecture slides 20-21 and the course books.

The following is short answer:

- The web browser needs the IP address of the server before initializing the TCP connection with the web server. Initial contact begins with local name server at Chalmers (host can learn address of DNS server from DHCP). The DNS-client starts with sending DNS-query to the local server in order to recursively resolve the hostname www.tue.nl into IP address.
(1) The local server queries a root-server to find nl TLD servers.
(2) The local server queries nl TLD-server to find tue.nl auth. Servers.
(3) The local server queries one authoritative DNS-server to get IP address(es) for www.tue.nl.

Local server caches answers (owner specifies cache timeout by including TTL in answer). The local server will reply to the DNS client by sending the answer including the IP address(es) of the server of the site.

DNS query and reply messages are transported in UDP segments.

Question 3: IPv6 Addresses and Neighbor Discovery (10 points)

These three IPv6 addresses are given with the compressed representation:

(i) 2001:6b0:2:10::1
(ii) FF02::1:ff6c:14dd
(iii)FE80::20c:f1ff:fe6c:14dd

Note: Please answer the following sub-questions (3a, 3b, and 3c) separately and in relation to the above addresses. In addition, you are allowed to use any of the given addresses as examples in your answers to sub-questions 3d and 3e.

3a. (2 points) What is the "type" of each of these IPv6 addresses? Explain what each type does imply.

Type implies which IPv6 interfaces will packets delivered (one receiver, many receivers)
(i) `2001:6b0:2:10::1 UNICAST`
(ii) `FF02::1:ff6c:14dd     MULTICAST`
(iii)`FE80::20c:f1ff:fe6c:14dd UNICAST`

3b. (1 point) Which of the given addresses **cannot** be used as valid source address in IPv6 packet? Explain why.

(ii) `FF02::1:ff6c:14dd     MULTICAST (group or set of interfaces, used only as destination address)`

3c. (2 points) What is the "scope" of each of these IPv6 addresses? Explain what each scope does imply.

(i) 2001:6b0:2:10::1 GLOBAL
(ii) FF02::1:ff6c:14dd LINK-LOCAL
(iii)FE80::20c:f1ff:fe6c:14dd LINK-LOCAL

IPv6 addresses are identifiers that are assigned to interfaces and **sets** of interfaces. The scope identifies the location of the receiver(s) of the IPv6 packets. It specifies in which part of the network the address is valid and where the packets are allowed or not to be routed to the destination. The scope is recognized by the prefix of the address and it can mainly be **local** or **global**.

3d. (2 points) What is the **main** purpose of IPv6 Neighbor Discovery? Explain **clearly** the operation.

3e. (3 points) What are the messages deployed in IPv6 Neighbor Discovery? Explain how these messages will be encapsulated and addressed in layer-3 and layer-2 PDUs (i.e. packets and frames).
Short answer for both **3d.** and **3e.**:

The main purpose of IPv6 Neighbor Discovery is to obtain the link-layer address of a neighbor, using ICMPv6 neighbor solicitation and neighbor advertisement messages sent in IPv6 packets with multicast to the solicited-node.

For complete answer please refer to slides 44-45 in IPv6 handouts.

Question 4: Dijkstra's self-stabilizing algorithm for token circulation (4 points)

Please find below Dijkstra's self-stabilizing algorithm for token circulation, as well as the proof outline, see Lemma 2.2 to 2.4 and Theorem 2.1. Please prove one of these proof elements, i.e., either Lemma 2.2, 2.3, 2.4 or Theorem 2.1.

P_1: **do** forever
  if $x_1 = x_n$ **then**
    $x_1 := (x_1 + 1) \bmod (n+1)$
$P_i (i \neq 1)$: **do** forever
  if $x_i \neq x_{i-1}$ **then**
    $x_i := x_{i-1}$

- A configuration in which all x variables are equal is a safe configuration for the set of legal execution ME (Lemma 2.2)
- For every configuration there exists at least one integer j such that for every p_i, x_i is not equal to j (Lemma 2.3)
- For every configuration c, in every fair execution that starts in configuration *c*, p_1 changes the value of x_1 at least once in every n rounds (Lemma 2.4)
- For every possible configuration c, every fair execution that starts in configuration *c* reaches a safe configuration with relation to ME within $O(n^2)$ rounds (Theorem 2.1)

Appears in the book and the slides.

Question 5: self-stabilization leader election (18 points)

5a. (1 point) Define the set of legal executions for leader election for general communication networks.

Appears in the book and the slides.

Below please find a non-self-stabilizing leader election algorithm for general networks in which processors have access to unique identifiers.

```
01 initialization
02      pulse_i := 0
03      ⟨candidate,distance⟩ := ⟨ ID(i), 0 ⟩
04 while pulse_i ≤ D do
05      forall P_j ∈ N(i) do
06         begin
07            ⟨leader_i[j],dis_i[j]⟩ := read⟨ leader_j, dis_j ⟩
08            if ((leader_i[j] < candidate) or
09               ((leader_i[j] = candidate) and (dis_i[j] < distance)) then
10                  ⟨candidate,distance⟩ := ⟨ leader_i[j],dis_i[j] + 1⟩
11         end
12      write⟨leader_i ,dis_i⟩ := ⟨candidate,distance⟩
```

5b. (2 points) Suppose that after the execution of line 03 by all processors, a transient fault occurs that change the state of a single processor. As a result, the system elects a leader that does not exist in the system, i.e., a floating identifier has been introduced to the system during the transient fault. Please define an example of a system configuration after that transient fault leads to the above result of the algorithm run.

Choose a candidate value of a lowest id processor that does not appear in the system.

The rest of this question is whether there exists a deterministic self-stabilizing leader election algorithm for general networks in which processors have access to unique identifiers.

<u>Please answer either sub-question 5c.1 or 5c.2 (not both).</u>

~~**5c.1 (15 points)** If you think that there is no such self-stabilizing algorithm, please prove your claim.~~

5c.2 If you think that there is such self-stabilizing algorithm, please design a deterministic self-stabilizing algorithm for leader election in general networks in which processors have access to unique identifiers.

This is basically the leader election self-stabilization algorithm for general graphs.

5c.2a (5 points) Write the pseudocode of your algorithm.

5c.2b (2 points) Define a safe configuration with respect to the set of legal executions (that you have defined) and the algorithm (that you have given the pseudo-code for).

5c.2c (6 points) Demonstrate that your algorithm can overcome the floating identifier problem.

5c.2d (2 points) Give the idea of the entire convergence property of your algorithm.

Question 6: self-stabilization vertex coloring (12 points)

6a. (1 points) Define the set of legal executions for the task of vertex coloring in general graphs.

Appears in the book and the slides.

Let $p_1, \ldots, p_n$ be n processors on a directed ring that Dijkstra considered in his self-stabilizing token circulation algorithm (in which processors are semi-uniform, i.e., there is one processor, p_1, that is different than other processors but there are no unique identifiers). Consider only deterministic algorithms and the task of vertex coloring in Dijkstra's directed ring. The rest of this question is whether there exists a deterministic self-stabilizing algorithm for coloring these networks.

Please answer either sub-question 6b.1 or 6b.2 (not both).

~~**6b.1 (11 points)** If you think that there is no such self-stabilizing algorithm, please prove your claim.~~

6b.2 If you think that there is such self-stabilizing algorithm, please design a deterministic self-stabilizing algorithm for vertex coloring for Dijkstra's directed ring.

In order to get at least half of the points, the solution need to identify a silent a silent self-stabilizing solution, i.e., a coloring that does not change during a legal execution. The full points requires a solution that uses O(log n) bits per processor and O(1) colors. For example, this can be done by counting the number of nodes in the ring and identifying whether there odd or even number of nodes. Both coloring schemes for even and odd number of nodes, the root processor fix to itself the 0 color. Any other processor that is not the root (or the one before the root when there are odd number of nodes), takes a color that is opposite of its previous, i.e., if the previous is 0 then it choses 1 and if it is 1 it choses 0. When there are odd number of nodes, the one before the root takes the color 2. Solutions that use O(log n) bits per processor and O(n) colors can lose some points – it mostly depends on the clarity of the presentation. For example, rank the nodes from zero to n and use that number as the node color.

6b.2a (2 points) Write the pseudocode of your algorithm.

6b.2b (1 point) Define a safe configuration with respect to the set of legal executions (that you have defined) and the algorithm (that you have given the pseudo-code for).

6b.2c (2 points) Demonstrate the closure property.

6b.2d (4 points) Demonstrate the convergence property.

6b.2d (2 points) Suppose that the number of processors, n, is an even number. Does your algorithm uses an optimal number of colors? Prove your claims. What is the minimal number of colors needed for vertex coloring Dijkstra's directed ring? How many colors does your algorithm use?

Every coloring of any graph that includes at least one edge requires the use of at least two colors. The proposed algorithm uses two. Thus, it is optimal.

Question 7: self-stabilization (digital) clock synchronization (12 points)

7a. (2 points) Define the set of legal executions for (digital) clock synchronization.
(1) The property of *agreement* means that ________ (complete this sentence).
(2) The property of *(no) adjustment* means that ________ (complete this sentence).

Appears in the book and the slides.

Let $p_1, \ldots, p_n$ be n processors on a uniform directed ring (in which there are no processors identifiers and all processors run the same program). Consider only deterministic algorithms and the task of (digital) clock synchronization in uniform directed ring. The rest of this question is whether there exists a deterministic and uniform self-stabilizing algorithm for (digital) clock synchronization for these networks ***that uses only a constant number of states per processor***.

Please answer either sub-question 7b.1 or 7b.2 (not both).

7b.1 (10 points) If you think that there is no such self-stabilizing algorithm, please prove your claim.

This proof appears in the book and the slides.

~~**7b.2** If you think that there is such self-stabilizing algorithm, please design a deterministic self-stabilizing algorithm for (digital) clock synchronization for uniform directed rings.~~

~~**7b.2a (2 points)** Write the pseudocode of your algorithm.~~

~~**7b.2b (2 points)** Define a safe configuration with respect to the set of legal executions (that you have defined) and the algorithm (that you have given the pseudo-code for).~~

~~**7b.2c (2 points)** Demonstrate the closure property.~~

~~**7b.2d (4 points)** Demonstrate the convergence property.~~