CHALMERS
EXAMINATION / TENTAMEN

Course code/ kurskod	Course name / kursnamn			
EDA387	Computer Networks			
Anonymous code Anonym kod		Examination date Tentamensdatum	Number of pages Antal blad	Grade Betyg
EDA387-10		2014-08-28	15	5

Solved task Behandlade uppgifter. No / nr		Points per task Poäng på uppgiften.	Observe: Areas with bold contour are to be completed by the teacher. Anmärkning: Rutor inom bred kontur ifylles av lärare.
1	✓	6	
2	✓	3	
3	✓	5	
4	✓	6	
5	✓	5	
6	✓	4	
7	✓	7	
8	✓	6	
9	✓	3+5	
10			
11			
12			
13			
14			
15			
16			
17			
18			
Total examination points Summa poäng på tentamen		50	

1

1a:

The Internet Domain Name System (DNS) is hierarchical and implemented as a huge distributed database. This hierarchy facilitates the autonomous handling of the names in Internet. ~~Each part of a~~ The authority for each part of the hierarchy is delegated to a designated authority so that it can handle the that part of the database for Internet names.

DNS basically consists of three-levels ~~of hierarchy~~ They are in the tree of DNS servers. They are

1. Root level servers
2. Top level domain servers (TLD)
3. Authoritative name servers (AS).

Root level servers are servers which are at the top-level of the DNS tree. These servers are handled by the IANA and IETF. There are 13 root servers globally. Each root server contains details about all the top-level domain servers.

TLD servers are responsible for handling all the domain names. One TLD server contains details about all the authoritative servers for that domain. For example a TLD server responsible for .com domain, contains details about all the .com authoritative name servers. In the internet, there are three types of top level domains.

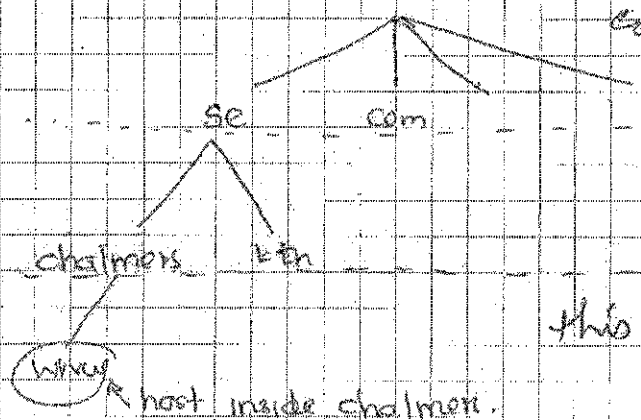
1. ccTLD - these are the country code TLDs, normally denoted by two letters.
   ex:- .se, .uk
2. gTLD - these are the generic TLDs, normally denoted by three letters
   ex:- .com, .net, .edu
3. Infrastructure TLD - this contains ~~all the~~ only the infrastructure domain arpa.

Authoritative name servers are ~~responsible~~ handled by a relevant ISP or an organization ~~for its domain~~ to list the details about the domain names which comes ~~under~~ under that organization. These authoritative name servers are configured with authoritative answers.

Let us consider the naming hierarchy with the following name

www.chalmers.se.

~~here se~~ If we consider the typical structure of this naming hierarchy it can be depicted as follows.



confusion between the naming hierarchy and the tree of servers!

- TLD.
- Authoritative

When it comes to naming, in the right most position the domain name of the TLD is written. Then before that the name of the authoritative domain is written. Like wise we can continue with the Fully qualified domain name, until it specifies the desired domain or a object inside the domain.

a 1c) When some host wants to find the IP address of www.chalmers.se host, it should first contact a root server and get the IP address of the .se TLD server. Then from this .se TLD server it should get the IP address of the authoritative name server for chalmers. Finally from chalmers authoritative name server, it can get the IP address for the host www.chalmers.se.

2

1b. Resource records specify the details about objects in the internet. A typical resource record contains the following four types of details.

~~Type~~ Resource Record (Name, Type, TTL, value)

Name describe the identifier use to address that object.
Type specifies the to which category a particular resource record belongs to.
TTL specifies the valid amount of time in seconds which a resource record can be used where?
Value specify the actual value or the content for a given name.

In a DNS database, there are different resource records maintained. four of them are as follows.

(i). Resource Record - A

Name : This is the fully qualified domain name of the host.
ex:- www.chalmers.se

Type : A

Value : IPv4 address of the host www.chalmers.se.

(ii). Resource Record - NS

Name : Domain name of an organization. ex:- chalmers.se
Type : NS
Value : Host name of the authoritative name server for that domain.

(iii) Resource Record - MX

Name : Domain name of an organization ex:- chalmers.se
Type : MX
Value : Host name of the mail server for chalmer domain.

(iv). Resource Record - PTR

Name : IP address of an organization → host → arpa
Type : PTR
Value : "Domain name" relevant to that IP address. host

3

2c.

When I enter the web site address www.tue.nl in my browser and hit enter, the DNS client software which is called the DNS resolver that resides in my machine will check the local DNS-cache to see whether there is an entry for www.tue.nl. Since this the first time I am accessing www.tue.nl there will be no entry. Therefor this resolver software will contact the DNS server inside the chalmer network. The IP address of the DNS server is learnt during the bootstrap process by my machine. Then the chalmers dns server will check its DNS database to see whether it has an entry. If it has an entry it will reply with the IP address of www.tue.nl to my resolver. But (if) it does not have that entry or my resolver demands for an authoritative answer, it then ?! then the chalmers server has to contact the name

findout

local cache-only

2

server responsible for www.tue.nl. Here if my resolver ask for an recursive answer then the DNS resolver ~~hear~~ at chalmers has to completely translate the www.tue.nl into the corresponding IP address and give it to my browser. (the case) of my ~~te~~ resolver ask for an iterative answer then the chalmers resolver will usually translate the first part of the domain and forward the IP address of the root level server to my resolver. ?!

In any of the above (cases), ~~the tra~~ the iterative or recursive case the translation happens as fellows.

First the resolver (either my resolver, if iterative or ~~chalmers~~ chalmers resolver if recursive), will contact the root server and get the IP address ~~response~~ of a TLD name server responsible for .nl domain. Then the resolver will contact that .nl TLD and get the IP address for the authoritative name server for tue.nl domain. Then from that name server, resolver will get the IP address for the www.tue.nl website. When the resolver get this IP address this will be returned to the browser and it will send a TCP connection request to the host at that address. This is the final outcome of this process

The translation uses the UDP protocol messages. ?!

!

1

1a

~~IPV6~~

IPV6 addresses allows to define multiple level of hierarchies compared to an IPV4 addresses. It consists of ~~a global level~~ three level of hierarchies. By the term 'scope' of an IPV6 destination address it means, up to which level the packet is allowed to roam inside a network.

not explained right!

0

2b

2001 : 6b0 : 2 : 10

type : unicast
scope : global

FF02 ::

type : multicast
scope : link-local

3

FE80 ::

type : unicast
scope : link-local

3a.

FE80 : a288 : b4ff : fe5c : c774.

type : unicast
scope : link-local

1

3b.

2001 : 06b0 : 2 : 10 : a288 : b4ff : fe5c : c774.

type : unicast
scope : global.

1

3c.

- In IPV6 instead of ARP protocol, it uses Neighbour Discovery ~~protocol. The per~~ operation. The purpose of this operation is to obtain the link layer address of an host when its IP address is known. So that ~~one~~ two hosts which are in the same ~~a~~ link ~~as the~~ can communicate with each other.

Neighbour Discovery operation uses two types of messages ~~that~~ send using ICMPV6 messages. The query is called the Neighbour Solicitation message which is sent using the ICMPV6 type value (135) message. This message asks for a link-layer address of a host with known IPV6 address. The response message is called the Neighbour advertisement message sent using ICMPV6 type value (136) message. This message replies with the link layer address for the asked IPV6 address. (multicast to the solicited node)

The protocol is simply as follows. Let us assume host A ~~with new address~~ ~~knows only~~ wants to find the link-layer address of host B. What host A knows about host B is only the IPV6 address of the hostB.

In the neighbour solicitation

- Host A creates a IPV6 packet with ~~her~~ source address of host A and ~~with~~ destination address of host B's solicited node multicast address.
- Then this packet is encapsulated inside an Ethernet Frame and sent as a multicast message. The source address of this frame is the ~~ad~~ link-layer address of host A and the destination address is the link-layer multicast address of hostB. ?

In the neighbour advertisement

- when hostB receives, hostA request first it creates a reply IPV6 packet. The source address of this

reply packet is the host Bs unicast address and the destination address is the host A's unicast address. Then this Ipv6 packet is encapsulated in a ethernet frame and sent as a ~~multicast~~ link-layer unicast message. The source address of the link-layer frame is the host B's link-layer address and the destination address is the host A's link layer address.

The link-layer address of host B is contained in the data section of the message.

3

4a. In the context of self-stabilization we consider the executions to be fair. We say an execution is an fair execution, if can be applie which is applicable infinitely often is executed infinitely often. Such an execution is called a fair execution. Therefore in a fair execution sequence, if the clock values are going to be changed infinitely often and if we don't have bounds then we should need a large block of memory to store the value of the clock. Lets say 64-bits. Then we can store 2^{64} different clock values. But due to the nature of a self stabilizing system, it can be started in any arbitrary configuration. Therefore if it is started in a configuration where clock value is equal 2^{64}, in the next immediate configuration a overflow occurs.
Therefore in the context of self stabilization this property is considered unattractive.

(2 - E) (6)

4b. Yes there is a deterministic self-stabilizing algorithm with bounded clock counter values.

In the above algorithm the line 05 has to be changed as follows.

clock_i := (max+1) mod ((n+1)d+1)

here n is the number of processor clocks in the system and d is the network diameter.

Here there can be two cases.
1. If all the clock values are less than (n-d) then before applying the modulo operation all the clocks values converges to a single value. So all the clock values become synchronized.

2. If all the clock values are not less than (n-d) then according to the pigeon hole principle there must be two values x and y such that y-x ≥ d+1, and there are no values in between x and y. Then if we consider the i th clock pulse after the configuration (x,y) we reach a configuration with (x+i, y+i) and there are no values in between them in this case either. Then after n+y+1 pulses the system reaches a configuration where all the clock values are less than (n-d). Then the above case holds and system reaches synchronization before the modulo is applied.

Therefore starting from any configuration ever, the algorithm converges.

Proof for lemma 2.3

For every configuration there exists at least one integer j such that for every P_i, x_i is not equal to j.

Proof:

When we consider this prog, it contains n processers and each processor P_i has got a variable x_i. Therefore at one configuration n processors can hold n different values.

When we consider the number of different values the a processor P_i can hold in a fair execution there are n+1 distinct values.

Therefore clearly, in a fair execution sequence in every configuration there exists at least one integer j such that for every P_i, x_i is not equal to j.

5

6a. vX. All of the above.

6b. iv. ✓ None of the above.

6c. iv. ✓ P1: do forever
if $x_1 = x_n$
then $x_1 = (x_1 + 1) \bmod (n+2)$
~~end~~
P_i ($i \neq 1$): do forever
if $x_i \neq x_{i-1}$
then $x_i = x_{i-1}$

6d. v. ✓ All of the above.

6e. The ✓ task of digital clock synchronization has to make sure that (a) all processors have same clock values and (b) the clock values are incremented by one in every ~~step~~ pulse.

6

7a. Safe configuration for this Leader Election task LE and the algorithm is the set of all configuration sequences in which every configuration, each processor has selected the processor with the minimum id and minimum distance in the system, as the leader. In other words for every processor pair P_i and P_j it holds that,

$$\langle leader_i, du_{ij} \rangle = \langle leader_j, du_j \rangle$$

$$\langle candidate, distance \rangle = \langle leader_j, du_j \rangle$$

↑ is that the min, id

7b. $c \xrightarrow{a_i} c'$

If c is a safe configuration in that configuration every processor has selected the same processor as the leader, which has the minimum id and minimal distance to a processor.

If we consider a computation step a_i, it consist of set of local computations followed by a single communication operation. In this algorithm there are two such steps that can be taken by a processor. One is at line 5 when reading their neighbouring registers and one is at line 10 when writing to the shared registers.

If we start in the safe configuration c, and execute line 05 everybody is going to read the details of the same processor into its local variables, that is the $\langle leader_j, dis_j \rangle$ of the actual $leader_j$. So in this step leader is not going to change.

If we consider the safe configuration c, and execute line 10, then everybody has selected the same processor as the leader by line 09. Therefore when it reaches configuration c' everybody is going to write the properties of the leader into its shared register. So in this step also leader is not changed.

Therefore since step a_i satisfies all the above two conditions, c' is a safe configuration.

$$c_{safe} \xrightarrow{a_i} c'_{safe}$$

5

8a.

The safe configuration for the maximal matching lemma and this algorithm is the set of all configuration sequences in which every configuration the value of $VF(cc) = (n, 0, 0, 0)$.

1. $pointer_i = j$ implies that $pointer_j = i$

According to the safe configuration, there are only matched and single processes in the system. If $pointer_i = j$, according to the definitions it should be a matched processor. Because P_i cannot be a single processor (without $pointer_i = null$. There since P_i is a matched processor, according to the definition of a matched processor $pointer_i = j$ implies that $pointer_j = i$.

2. if $pointer_i = null$ then there is no neighbour P_j, such that if $pointer_j = null$

If $pointer_i = null$ then it should be a single processor. When P_i is a single processor all the other neighbouring processor, P_j s should be in matched state. Therefore in this case there is no neighbour P_j such that $pointer_j = null$.

8b.

$$c \xrightarrow{a_i} c'$$

if C is safe then $VF(c) = (n, 0, 0, 0)$. In this configuration if it if the system take the step a_i, then it should execute either of lines a2, a3, a4. So by showing that in each case $VF(c) = VF(c')$ we can prove that c' is going to be safe in step a_i.

In line a2:

According to the condition of the if clause, if it is to be satisfied there should be one waiting process and one free process. But in C there are no such processes. Therefore in step a_i none of the processors changes its states.

Therefore

$\therefore VF(c) = VF(c')$

In line a3:

This condition is only satisfied if there are free processors but there are no free processors in the system. therefor in step a_i none of the processors changes its state. Therefore $VF(c) = VF(c')$.

In line 04:

This condition is satisfied if there are only chaining processes. None of the processes are chaining. ~~In configuration C. Therefore processor states are going to be same in step a1. Therefore~~ in configuration C. Therefore in step a1 none of the processes change their state. Therefore

$$VFC(C) = VFC(C').$$

Since in all cases of 02 03 04 $VFC(C) = VFC(C')$, C' is a safe configuration.

9a. In TCP TIME-WAIT state has two purposes. They are

1. Implementing TCP full-duplex termination
2. Avoid possible reincarnation of the TCP connections.

Typically the TIME-WAIT state is set to be two MSL (Maximum segment lifetime). By this setting it allows any packets dropped inside the network to die when its lifetime expires. Therefore if we shorten the TIME-WAIT state, we a host may receive a packet which belongs to a previous connection. If the TCP has allocated the resources that belongs to that connection to a new connection then this new connection will receive that packet and it may corrupt the state of the TCP connection. Therefore shortening the TIME-WAIT state should be done in a careful manner so that it will not allow any reincarnations and will terminate the connections property.

9b. Number of simultaneous socket connections depends on the application and the machine. Typically, a when a socket is created it returns a file descriptor that is unique to that socket. Since a machine has a finite amount of memory, it only allocates a limited amount of memory from the available finite memory to an application. Because of this reason the number of file descriptors are also limited.

9c. In Internet the congestion control is done through an end-to-end method.

Network assisted congestion control means the intermediately network devices such as routers has to explicity send feedback about the congestion to the senders. It should has to be achieved either to a special packet sent from router to a sender, or by marking a packet flowing from sender to receiver. Then the receiver will feedback this control message back to the sender. In practically this is expensive and it's implementation is not easy

(2)

In end-to-end congestion, the congestion is controlled by the sender totally depending on the observed Congestion on the network. That by a loss event. A loss event can be either a timeout or a receiving of 3 duplicate acks. In order to maintain control congestion, the sender can maintain a congestion window. This method is effective and practically feasible. Therefore Internet has chosen end-to-end congestion on behalf of

network assisted congestion control.

9e. TCP is a protocol that provides both error control and the congestion control.

In order to provide error control, when a packet loss occurs it has to retransmit packets. Also in order to facilate this TCP uses sequence number and acknowledgements. This guarantee reliable in order delivery. Because of this error control mechanisms TCP does not provide any time delivery guarantees.

In order to provide congestion control, TCP increases and decreases the rate at which it sends traffic in to the network. This is done through the TCP congestion control algorithm which consists of three phases slowstart, congestion avoidance and fast recovery. When TCP is in slow start mode the rate of traffic is increased exponentially. When it is in congestion avoidance phase rate of traffic is increased linearly. But in both phases, if a loss event occurs due to a timeout then the congestion window is set to 1 MSS and it starts from the slow start phase. Because of that reason there are quick and abrupt bandwidth changes in the TCP. Therefore TCP does not provide any bandwidth guarantees.

but when it comes to real-time traffic depending on the application it might require certain timelines and bandwidth.

. For example in an online game it might require as much as bandwidth quickly as possible. On the other hand like-a applications like internet telephony and multimedia apps like streaming requires smooth transition rate without abrupt changes. ~~Because of this reasons~~

Because of the above reasons, if TCP is used for real-time traffic due to error control there will be arbitrary delays in the traffic whereas due to congestion control there will be abrupt changed in the bandwidth. both of these properties ~~affects real-time~~ are undesirable for real-time traffic. So we need another protocol like UDP for real-time traffic.

Answer only one question on this page. Do not write on the back of this paper
Behandla endast en uppgift på detta blad. Skriv ej på baksidan