CS4514 HELP Session 3

Concurrent Server Using Go-Back-N

Song Wang

02/17/2004
Description

- You are supposed to implement a simple concurrent server and client having four emulated network protocol stack.
  - Application layer: Read and execute commands
  - Network layer: Message $\leftrightarrow$ Packet (send&recv)
  - Datalink layer: Packet $\leftrightarrow$ Frame and Go-Back-N sliding window protocol
  - Physical layer: TCP connection.

- Your programs should compile and work on any one of ccc.WPI.EDU.
System Overview

Note: each child process keeps a separate copy of the DB.

we do not keep data consistency for the serverbase

This is automatically done by using fork()
System Framework

Client

APP Layer

NW Layer

DLL

PHL

Server

APP Layer

NW Layer

DLL

PHL

TCP Connection
Concurrent Server (fork())

- fork() will make a child process with memory copy.
  - The initial serverbase will be copied to each child process.
  - fork() will return child pid in parent process and 0 in child process.
  - Remember to close socket after using.
Go Back N

Go-Back-4:

4 frames are outstanding; so go back 4

Out-of-sequence frames

ACKing next frame expected
How the System Works: Layer by Layer

Client

Read “scripted action” from file “scripti.txt”

Client Request:

- cmd No. [msg]
  - cmd::r / w / q [1-15]...
  - msg1::r[6]
  - msg2::w[4]Duke...
  - msg3::q
  - NO sequence# for msg

Server

Read/Write a message

nwl_send (… msg …)

nwl_recv (… msg …)

Child Process i

Stop-Wait

ACK

APP

APP

Note: The max_size of a message is 285 bytes

The number referring to tuple position is 1 to 15
How the System Works: Layer by Layer

Client

message

NWL

n_packets

Server

message

NWL

n_packets

End of Message
Can be an special packet, OR a special position in each packet, eg. 1st byte

dll_send (… pkt …)
dll_recv (… pkt …)

Tasks for NWL
Disassemble and assemble packets from Msg.
No response in this layer
No sequence no. for packets

Note: The max_size of a packet is 70 bytes

The network layer will send packets until blocked by the Data Link Layer

02/17/2004  
Computer Science  
Session 3-8
How the System Works: Layer by Layer

Clienti

packet

DLL

n_frames

End of Packet
Error Detection
Byte Stuffing
Go-Back-N

Server

packet

DLL

n_frames

phl_send (… frm …)
phl_recv (… frm …)

Note: The max_size of a frame payload is 45 bytes

Sliding window size >=3

02/17/2004
Computer Science
Session 3-9
How the System Works: Layer by Layer

Client

- frame

- PHL

- read (… data …)

- write (… data …)

Server

- frame

- PHL

- Identify client when start

- Force Single Bit Errors
  - Client: 6th frames
  - Server: 8th frames

TCP Connection
How the Functions Work: Layer by Layer

**client APP**

1. Read script file
2. Pick a command
   - q cmd?
     - No: Build Msg
       - nwl_send(…msg…)
     - Yes: nwlRecv(…ack…)
   - Yes: Build Msg
     - nwl_send(…msg…)

**server child process APP**

1. fork()
2. nwl_recv(……)
   - q cmd?
     - No: Build Msg
     - Yes: nwl_send(…msg…)

How the Functions Work: Layer by Layer

\[ \text{nwl\_send (\ldots\ text \ldots)} \]

- Split msg into pkts
- Pick a pkt
- Last pkt?
  - No: \text{ dll\_send (\ldots pkt \ldots)}
  - Yes: \text{ Chang EOM}

\[ \text{nwl\_recv (\ldots\ text \ldots)} \]

- \text{ dll\_recv (\ldots pkt \ldots)}
- Last pkt?
  - No
  - Yes: Reassemble pkts into msg
    - Return msg to APP
How the Functions Work: Layer by Layer

dll_send (... pkt ... )

- Split a packet into payloads
- Create a new frame
- Start a Timer
- Send a frame to PHL
- Wait for receiving a ACK frame
- Retransmit frames if timeout or error ACK frame!
- Receive a ACK frame correctly, then continue ...

Sliding window size =1

phl_send (...)

phl_recv (...)

Computer Science
How the Functions Work: Layer by Layer

dll_recv (… pkt … )

Receive a frame from PHL

Unstuffing the frame

Compute ED byte and check error

Drop if error detected

Drop if duplicate, else send ACK

Reassemble the packet

If EOP, forward the packet to NWL
Log Significant Events

- **Performance Timing**
- **Packet Sent**
- **Frame received in error**
  - \(i\)th log: client\(i\).log
  - \(i\)th log: server\(i\).log
Project Tips

- Sliding Window Protocol: Go-Back-N (N>3)
  - Try to implement Go-Back-1 first
  - Then implement Go-Back-N (multiple timers)
- Maybe easier to merge PHL and DLL
- How to terminate client process:
  - When the client gets the response to the quit message
  - A “clean” way to terminate the server child process?
Project Tips

- Simulate multiple timer in software
  - Approach I
    - Using link list or array
    - pp.223 on textbook()
    - Need signal()
  - Approach II
    - Using link list or array
    - Update the *struct timeval* for next select() call
Concurrent TCP Server Example

```c
pid_t pid;
int listenfd, connfd;

/* 1. create a socket socket() */
if ((listenfd = socket(AF_INET, SOCK_STREAM, 0)) < 0 )
  err_quit("build server socket error\n", -1);
/* 2. fill in sockaddr_in{ } with server's well-known port */
...
/* 3. bind socket to a sockaddr_in structure bind() */
bind (listenfd, ...);
/* 4. specify the backlog of incoming connection requests listen() */
listen (listenfd, LISTENQ);
while(1){
  connfd = accept(listenfd, ... ); /* probably blocks */
  if(( pid = fork()) == 0){
    close(listenfd); /* child closes listening socket */
    doit(connfd); /* process the request */
    close(connfd); /* done with this client */
    exit(0);
  }
  close(connfd); /* parent closes connected socket */
}
```

02/17/2004 Computer Science Session 3-18