Chương 1: Giới thiệu

Assignment for Introduction

Câu hỏi 1: Vai trò của Middleware trong Hệ Phân Tán là gì?

Câu hỏi 2: Tại sao nói Middleware thừa kế các ưu điểm của cả DOS và NOS?

Câu hỏi 3: Giải thích tính trong suốt của HPT, đưa ra các ví dụ cho các kiểu trong suốt khác

nhau.

Câu 4: Tại sao đôi khi rất khó để có thể che giấu sự xuất hiện lỗi và sự phục hồi lỗi trong HPT?

Câu 5: Tại sao không phải lúc nào cũng là tốt khi giữ độ trong suốt ở mức cao nhất có thể?

Câu 6: Thế nào là một HPT mở? Các lợi ích của tính mở đem lại là gì?

Câu 7: Những kỹ thuật nào được sử dụng để mở rộng hệ thống?

Bài làm:

Câu 1:

Vai trò của Middleware trong Hệ Phân Tán: giúp cải thiện tính trong suốt của hệ thống, giúp cho việc

cấu hình là duy nhất.

Câu 2:

Middleware thừa kế tính trong suốt của DOS, tính m ở của NOS - Th ừa kế ưu điểm của cả 2

Câu 3:

Tính trong suốt của hệ phân tán: Hệ thống là duy nhất v ới người s ử dụng.

Các kiểu trong suốt: Che giấu về quy mô, vị trí, việc điều h ướ ng, d ữ liệu...

Câu 4:

Rất khó để che dấu sự xuất hiện lỗi và phục hồi lỗi trong Hệ phân tán vì nh ư vậy hiệu năng sẽ

không được đảm bảo ở 1 mức độ nào đó.

Câu 5:

Bởi vì độ trong suốt tỷ lệ nghịch v ới hiệu năng. Nếu giữ độ trong suốt ở mức cao nhất thì hiệu năng

lại thấp nhất.

Câu 6:

Hệ phân tán mở là hệ phân tán có khả năng kết nối thêm, m ở rộng máy tính, tài nguyên mà không

quá bị đặc thì bởi kiến trúc máy tính.

Các lợi ích của tính m ở đem lại:

+ Cho phép các thành phần của nó được sản xuất t ừ nhiều nhà sản xuất khác nhau.

+ Khả chuyển, mềm dẻo, không tạo nên tính đặc quyền, dễ dàng nâng cấp hệ thống. Hệ thống sẽ

nhanh chóng hoàn thiện.

Câu 7:

Những kỹ thuật được sử dụng để mở rộng hệ thống:

- Phân tán dữ liệu

- Sao lưu

- Sử dụng ức chế bộ đệm

Labwork\_exercise: Introduction

Câu hỏi 1: Chỉ ra đường dẫn đến file html chứa nội dung của trang web mặc định của apache.

Câu hỏi 2: Cổng mặc định của giao thức http mà web server lắng nghe là cổng nào?

Câu hỏi 3: Phân quyền 755 nghĩa là thế nào?

Câu hỏi 4: Sau khi gõ 2 địa chỉ trên thì bạn thấy điều gì? giải thích.

Câu hỏi 5: bây giờ muốn cho các máy khác trong cùng mạng LAN với máy tính vừa làm việc

cũng truy cập được vào 2 địa chỉ example.com và test.com thì chúng ta phải làm gì? giải thích.

Bài làm:

Câu 1: Đường dẫn

/var/www/example.com/index.html

Câu 2: Cổng mặc định: 80

Câu 3: Phân quyền 755: set quyền cho app để hoạt động tốt trên IP

Câu 4: Sau khi gõ 2 địa chỉ trên, kết quả hiển thị như hình d ưới đây:

Câu 5:

Muốn cho các máy khác nhau trong cùng mạng Lan v ới máy tính v ừa làm việc cũng truy cập được

vào 2 địa chỉ example.com và test.com thì chúng ta cần đặt cổng mặc định cho giao th ức http và

web server

Chương 2: Kiến trúc

Assignment for Architecture

Câu hỏi 1: Khi máy client và server đặt xa nhau về mặt địa lý, điều này sẽ dẫn tới việc độ trễ

thời gian trao đổi thông tin giữa 2 máy là cao, hậu quả là sẽ làm giảm hiệu năng hệ thống. Bạn

có đề xuất gì để giải quyết vấn đề này?

Câu hỏi 2: Kiến trúc client-server 3 tầng là gì?

Câu hỏi 3: Sự khác nhau giữa phân tán dọc và phân tán ngang là gì?

Câu hỏi 4: Trong một mạng overlay có cấu trúc, các thông điệp được định tuyến dựa theo hình

trạng mạng (topology). Nhược điểm quan trọng của hướng tiếp cận này là gì?

Câu hỏi 5: Xét một chuỗi các tiến trình P1, P2, ..., Pn triển khai một kiến trúc client-server đa

tầng. Cơ chế hoạt động của tổ chức đó như sau: tiến trình Pi là client của tiến trình Pi+1, và Pi

sẽ trả lời Pi-1 chỉ khi đã nhận được câu trả lời từ Pi+1.

Vậy những vấn đề nào sẽ nảy sinh với tổ chức này khi xem xét hiệu năng yêu cầu-trả lời tới

P1?

Câu hỏi 6: Xét mạng CAN như trong hình. Giả sử tất cả các node đều biết node hàng xóm của

mình. Một giải thuật định tuyến được đưa ra đó là gửi các gói tin cho node hàng xóm gần mình

nhất và hướng đến đích. Giải thuật này có tốt không? Giải thích.

Bài làm:

Câu 1:

- Chia các đoạn mã ở client thành các đoạn nhỏ và cho chạy độc lập

- Thiết kế client sao cho nó có thể làm việc khác nhau sau khi đã g ửi 1 yêu cầu

- Thay thế trao đổi thông tin đồng bộ sang bất đồng bộ 1 chiều

Câu 2:

Kiến trúc client-server 3 tầng:

- Giao diện

- Nghiệp vụ

- Dữ liệu

Câu 3:

Sự khác nhau giữa phân tán dọc và phân tán ngang là:

- Phân tán dọc phân tầng theo chức năng

- Phân tán ngang phân tầng theo tải

Câu 4:

Nhượ c điểm:

Liên kết giữa các node chỉ là liên kết logic nên không thấy được các liên kết vật lý bên d ưới

Câu 5:

- Hiệu năng bị ảnh hưở ng bởi (n-2) tươ ng tác ở phía trên

- Nếu Pi bị treo sẽ không gửi được yêu cầu cho Pi-1

Câu 6:

Giải thuật đó chưa tối ưu:

VD: Từ nút A gửi cho nút C theo giải thuật, có thể đi t ừ A-D-C mất th ời gian h ơn so v ới A-B-C

Chương 3: Tiến trình và luồng

Assignment for Processes and Threads

Câu 1: Có cần thiết phải giới hạn số lượng các luồng trong một tiến trình server?

Câu 2: Có nên chỉ gắn một luồng đơn duy nhất với một tiến trình nhẹ?

Câu 3: Có nên chỉ có một tiến trình nhẹ đơn gắn với 1 tiến trình?

Câu 4: Bài toán này yêu cầu bạn so sánh thời gian đọc một tệp (file) của một máy chủ tập tin

(file server) đơn luồng và một máy chủ đa luồng. Phải mất tổng cộng 15 ms để nhận 1 yêu cầu

(request) và thực hiện quá trình xử lý, giả định rằng các dữ liệu cần thiết nằm ở bộ nhớ đệm

trong bộ nhớ chính. Nếu cần thiết phải thực hiện một thao tác truy cập ổ đĩa thì cần thêm 75

ms, biết rằng việc phải thực hiện thao tác này có xắc suất là 1/3. Hỏi máy chủ có thể nhận bao

nhiêu yêu cầu/giây trong 2 trường hợp: máy chủ là đơn luồng và máy chủ là đa luồng (ngoài

luồng nhận và xử lý request, sẽ có thêm 1 luồng để truy cập ổ đĩa nếu cần thiết)? Giải thích.

Câu 5: Hệ thống X chỉ định máy của user chưa server, trong khi các ứng dụng lại được coi như

client. Điều đó có vô lý không? Giải thích.

Câu 6: Giao thức thiết kế cho hệ thống X gặp phải vấn đề về tính mở rộng. Chỉ ra các giải pháp

để giải quyết vấn đề đó?

Câu 7: Với việc xây dựng một server đồng thời, hãy so sánh việc server này tạo một luồng mới

và tạo một tiến trình mới khi nhận được yêu cầu từ phía client.

Câu 8: Nếu bây giờ một webserver tổ chức lưu lại thông tin về địa chỉ IP của client và trang web

client đó vừa truy cập. Khi có 1 client kết nối với server đó, server sẽ tra xem trong bảng thông

tin, nếu tìm thấy thì sẽ gửi nội dung trang web đó cho client. Server này là có trạng thái

(stateful) hay không trạng thái (stateless)?

Bài làm:

Câu 1:

Cần giới hạn số lượ ng các luồng trong một tiến trình server vì càng tạo nhiều luồng càng tốn bộ nh ớ

chính

Câu 2:

Không phải làm

Câu 3:

Không nên chỉ có 1 tiến trình nhẹ đơn gắn v ới 1 tiến trình

Câu 4:

- Đơn luồng:

Thời gian để nhận 1 yêu cầu: 15 x 2/3 + (15+75) x 1/3 = 40 (ms)

- Số yêu cầu/giây: 1000/40 = 250 (yêu cầu)

- Đa luồng: 1000/15 (yêu cầu)

Câu 5:

Điều đó không vô lý

Câu 6:

Giải pháp:

- Cải thiện giao thức X protocol

- Nén thông điệp

Câu 7:

Tạo tiến trình tốn tài nguyên hơn so v ới luồng

Tạo tiến trình mới -Xử lý tách biệt tài nguyên - Đơn giản h ơn

Nếu muốn các collection trao đổi v ới nhau thì nên tạo ra luồng.

Câu 8:

Server này là không trạng thái (stateless) đặc trưng của webserver.

Labwork\_exercise: processes and threads

Câu hỏi 1: Thông điệp "HTTP/1.1 200 OK" là gì? Giải thích.

Câu hỏi 2: Mô tả hiện tượng xảy ra khi gõ địa chỉ như trên vào trình duyệt. Giải thích chi tiết.

Chương 4: Trao đổi thông tin

Assignment to Communication

Câu hỏi 1: Trong các giao thức phân tầng, mỗi tầng sẽ có một header riêng. Vậy có nên triển

khai một hệ thống mà tất cả các header của các tầng đưa chung vào một phần (gọi là header

chung), gắn vào đầu mỗi thông điệp để có thể xử lý chung? Giải thích.

Câu hỏi 2: Xét 1 thủ tục incr với 2 tham số nguyên. Thủ tục làm nhiệm vụ là cộng 2 tham số đó

với nhau. Bây giờ xét trường hợp chúng ta gọi thủ tục đó với cùng một biến 2 lần, ví dụ incr(i,

j). Nếu biến i được khởi tạo giá trị 0, vậy giá trị của i sẽ là bao nhiêu sau khi gọi thủ tục này

trong 2 trường hợp sau:

- Lời gọi tham chiếu

- Phương pháp sao chép-phục hồi được sử dụng.

Câu hỏi 3: Hãy mô tả quá trình trao đổi thông tin không hướng kết nối giữa client và server khi

sử dụng socket.

Câu hỏi 4: Liệu có thể áp dụng phương pháp trao đổi thông tin bất đồng bộ bền vững vào cơ

chế RPC được không? Phân tích.

Bài làm:

Câu 1:

Không nên triển khai 1 hệ thống như vậy vì nếu làm vậy thì hệ thống sẽ không còn ý nghĩa phân

tầng, các tầng cần độc lập với nhau, việc thay thế, cải tiến sẽ làm mất đi tính trong suốt của hệ

thống.

Câu 2:

- Trườ ng hợp 1: Lời gọi tham chiếu

Mỗi con trỏ đều trỏ -i, nếu i tăng 1 đơn vị -i tăng 2 lần -i = 2

- Trườ ng hợp 2: Phươ ng pháp sao chép phục hồi được sử dụng:

Sao lưu không làm tăng giá trị.

Ghi đè giá trị 2 lần -i =1

Câu 3:

Quá trình trao đổi thông tin không hướ ng kết nối gi ữa client và server khi s ử dụng socket:

Trướ c khi trao đổi dữ liệu, bên gửi và bên nhận sẽ thiết lập 1 kênh truyền chung, thống nhất v ới

nhau về giao thức sẽ sử dụng.

Câu 4:

Áp dụng phươ ng pháp trao đổi thông tin bất đồng bộ bền vững vào c ơ chế RPC:

- Tính bất đồng bộ: Thông tin không tự block mình sau khi gọi

- Tính bền vững: Có 1 bộ nh ớ trung gian để lưu tr ữ - biến trung gian

- Tính tin cậy: Coi reply như 1 thông điệp báo nhận

Đặc điểm RPC:

- Ko cần báo nhận

- Ko cần kiểm soát luồng

- Ẩn quá trình thiết lập kết nối

- Phù hợp với datagram ở tầng giao vận

Chương 5: Tên trong hệ phân tán

Assignment for Naming

Câu hỏi 1: Tại sao không thể lấy địa chỉ của Access Point để sử dụng như địa chỉ của thực

thể?

Câu hỏi 2: Với việc sử dụng Định Danh, các vấn đề gì cần phải xem xét?

Câu hỏi 3: Xét một thực thể di chuyển từ vị trí A sang vị trí B. Trong quá trình di chuyển thực

thể đó có đi qua các nút trung gian nhưng chỉ dừng lại ở đó khoảng thời gian ngắn. Khi đến B,

thực thể đó dừng lại. Chúng ta biết rằng việc thay đổi địa chỉ trong một dịch vụ tổ chức vị trí

phân cấp (hierarchical location service) là rất mất thời gian để hoàn thành, vì vậy cần tránh làm

việc này khi thực thể tạm dừng ở các nút trung gian. Hãy đề xuất một mô hình kết hợp cả dịch

vụ tổ chức vị trí phân cấp và cơ chế chuyển tiếp con trỏ (forwarding pointers) để có thể xác định

được vị trí của thực thể khi nó ở các nút trung gian.

Câu hỏi 4: Khi chúng ta thêm 1 node mới vào hệ thống Chord, chúng ta có cần phải cập nhật

toàn bộ các bảng finger?

Câu hỏi 5: Phân giải tên đệ qui có ưu điểm gì so với phân giải tên không đệ qui?

Bài làm:

Câu 1:

Vì thông tin có khả năng di chuyển từ Access point này sang Access point khác

Câu 2:

Các vấn đề:

Không gian định danh hữu hạn - Cạn kiệt định danh

Giải pháp: Sử dụng lại - Có khả năng bị trùng

Câu 3:

- Khi thông tin dịch chuyển vị trí trung gian A - B, để lại con trỏ khi đến B ghi thêm địa chỉ m ới đó -mô hình phân cấp.

- Chuỗi con trỏ trung gian bị xóa đi, địa chỉ ở A cũng bị xóa

Câu 4:

Không cần phải cập nhật toàn bộ các bảng finger, chỉ cần cập nhật cho nút sau và nút tr ước vì trong

hệ thống Chord, mỗi nút sẽ quản lý khóa của nút tr ướ c đó.

Câu 5:

Ưu điểm của phân giải có đệ quy so v ới phân giải không đệ quy:

- Phân giải tên có đệ quy xử lý nhiều ở bên server, phân giải ko đệ quy x ử lý nhiều bên client. Khi

client ở xa, phân giải không đệ quy phải gửi đi gửi lại dẫn t ới độ trễ cao.

- Phân giải tên có đệ quy các name server tổ ch ức l ưu trữ bộ đệm, server l ưu tr ữ kết quả, có thể trả

lời luôn và nhanh cho các client.

Chương 6: Đồng bộ hóa

Assignment for Synchronization

Câu hỏi 1: Trình bày 1 ví dụ để mô phỏng vấn đề gặp phải khi các máy tính/tiến trình hoạt động

trong hệ thống phân tán mà không có đồng hộ vật lý dùng chung.

Câu hỏi 2: Tại sao Lamport lại đề xuất sử dụng đồng hồ logic thay cho đồng hồ vật lý trong hệ

phân tán?

Câu hỏi 3: Đặc điểm gì của mạng không dây (wireless network) khiến cho thiết kế các giải

thuật đồng bộ khác các kiểu mạng khác?

Câu hỏi 4: Giải thuật Lamport được đưa ra để thực hiện loại trừ lẫn nhau (mutual exclusion).

Giải thuật được mô tả như sau:

Hệ thống có n tiến trình: P1, P2, ... Pn. Có 1 tài nguyên chia sẻ dùng chung gọi là SR (Shared

Resource). Mỗi tiến trình sẽ lưu trữ một hàng đợi queuei để lưu các yêu cầu của các tiến trình

khác khi chưa được thực hiện.

Khi tiến trình Pi muốn truy cập vào SR, nó sẽ quảng bá 1 thông điệp REQUEST(tsi,i) cho tất cả

các tiến trình khác, đồng thời lưu trữ thông điệp đó vào hàng đợi của mình (queuei) trong đó tsi

là timestamp của yêu cầu.

Khi 1 tiến trình Pj nhận được yêu cầu REQUEST(tsi,i) từ tiến trình Pi thì nó đưa yêu cầu đó vào

hàng đợi của mình (queuej) và gửi trả lại cho Pi thông điệp REPLY.

Tiến trình Pi sẽ tự cho phép mình sử dụng SR khi nó kiểm tra thấy yêu cầu của nó nằm ở đầu

hàng đợi queueivà các yêu cầu khác đều có timestamp lớn hơn yêu cầu của chính nó.

Tiến trình Pi, khi không dùng SR nữa sẽ xóa yêu cầu của nó khỏi hàng đợi và quảng bá thông

điệp RELEASE cho tất cả các tiến trình khác.

Khi tiến trình Pj nhận được thông điệp RELEASE từ Pi thì nó sẽ xóa yêu cầu của Pi trong hàng

đợi của nó.

Câu hỏi:

a) Để thực hiện thành công 1 tiến trình vào sử dụng SR, hệ thống cần tổng cộng bao nhiêu

thông điệp?

b) Có 1 cách cải thiện thuật toán trên như sau: sau khi Pj gửi yêu cầu REQUEST cho các tiến

trình khác thì nhận được thông điệp REQUEST từ Pi, nếu nó nhận thấy rằng timestamp của

REQUEST nó vừa gửi lớn hơn timestamp của REQUEST của Pi, nó sẽ không gửi thông điệp

REPLY cho Pi nữa.

Cải thiện trên có đúng hay không? Và với cải thiện này thì tổng số thông điệp cần để thực hiện

thành công 1 tiến trình vào sử dụng SR là bao nhiêu? Giải thích.

Câu hỏi 5: Giải thuật Szymanski được thiết kế để thực hiện loại trừ lẫn nhau. Ý tưởng của giải

thuật đó là xây dựng một phòng chờ (waiting room) và có đường ra và đường vào, tương ứng

với cổng ra và cổng vào. Ban đầu cổng vào sẽ được mở, cổng ra sẽ đóng. Nếu có một nhóm

các tiến trình cùng yêu cầu muốn được sử dụng tài nguyên chung SR (shared resource) thì các

tiến trình đó sẽ được xếp hàng ở cổng vào và lần lượt vào phòng chờ. Khi tất cả đã vào phòng

chờ rồi thì tiến trình cuối cùng vào phòng sẽ đóng cổng vào và mở cổng ra. Sau đó các tiến

trình sẽ lần lượt được sử dụng tài nguyên chung. Tiến trình cuối cùng sử dụng tài nguyên sẽ

đóng cổng ra và mở lạicổng vào.

Mỗi tiến trình Pi sẽ có 1 biến flagi, chỉ tiến trình Pi mới có quyền ghi, còn các tiến trình Pj (j ≠ i)

thì chỉ đọc được. Trạng thái mở hay đóng cổng sẽ được xác định bằng việc đọc giá trị flag của

các tiến trình khác. Mã giả của thuật toán đối với tiến trình i được viết như sau:

#Thực hiện vào phòng đợi

flag[i] ← 1

await(all flag[1..N]∈{0,1,2})

flag[i] ← 3

ifany flag[1..N]=1:

flag[i] ← 2

await(any flag[1..N]=4)

flag[i] ← 4

await(all flag[1..i-1]∈{0,1})

#Sử dụng tài nguyên

#...

#Thực hiện giải phóng tài nguyên

await(all flag[i+1..N]∈{0,1,4})

flag[i] ← 0

Giải thích ký pháp trong thuật toán:

await(điều\_kiện): chờ đến khi thỏa mãn điều\_kiện

all: tất cả

any: có bất kỳ 1 cái nào

Câu hỏi:

flag[i] sẽ có 5 giá trị trạng thái từ 0-4. Dựa vào giải thuật trên, 5 giá trị đó mang ý nghĩa tương

ứng nào sau đây (có giải thích):

- Chờ tiến trình khác vào phòng chờ

- Cổng vào được đóng

- Tiến trình i đang ở ngoài phòng chờ

- Rời phòng, mở lại cổng vào nếu không còn ai trong phòng chờ

- Đứng đợi trong phòng chờ

Bài làm:

Câu 1:

Ví dụ, 1 mạng máy tính nhận được 1 gói tin có th ời gian gửi là sau th ời gian hiện tại trên máy đó

Câu 2:

Vì quan trọng là thứ tự của yêu cầu chứ không phải thời gian cụ thể

Câu 3:

Đặc điểm mạng không dây: cần tối ưu về việc nhận và gửi thông điệp

Câu 4:

a)

(n-1) thông điệp request

(n-1) thông điệp reply

(n-1) thông điệp release

= Tổng = 3(n-1)

b)

2(n-1) < ... > 3(n-1)

Tốt nhất ... Xấu nhất=Tiết kiệm (n-1) thông điệp reply

Câu 5:

- Chờ tiến trình khác vào phòng ch ờ: 2

- Cổng vào được đóng: 4

- Tiến trình i đang ở ngoài phòng ch ờ: 1

- Rời phòng, mở lại cổng vào nếu không còn ai trong phòng ch ờ: 0

- Đứng đợi trong phòng ch ờ: 3