Câu 3: a) Định nghĩa “thông điệp” và “chuyển thông điệp” trong hệ thống phân tán

- **Thông điệp (Message)**:
  Thông điệp là **nội dung chứa thông tin** cần được **vận chuyển giữa các đối tượng** trong hệ thống phân tán. Nó có thể bao gồm dữ liệu người dùng, yêu cầu dịch vụ hoặc kết quả xử lý.

- **Chuyển thông điệp (Message Passing)**:
  Chuyển thông điệp là **phương pháp truyền dữ liệu cơ bản** giữa các tiến trình trong hệ thống phân tán thông qua mạng. Dữ liệu được đóng gói thành thông điệp và gửi qua các tầng giao tiếp đến bên nhận. Phương pháp này có thể được triển khai bằng nhiều kỹ thuật như **Socket (TCP/UDP)**, **giao diện truyền thông điệp MPI**, hoặc **hàng đợi thông điệp (message queue)** để đảm bảo tính bền bỉ.

b) Ưu điểm và nhược điểm của phương pháp chuyển thông điệp so với chia sẻ bộ nhớ

Tiêu chí	Chuyển thông điệp	Chia sẻ bộ nhớ
Tính trong suốt	- Thấp hơn (cần quản lý gửi/nhận, đồng bộ, lỗi). - Không ẩn hoàn toàn tính phân tán khỏi người lập trình.	- Cao hơn, do các tiến trình chia sẻ cùng một không gian bộ nhớ (đặc biệt trong hệ thống đơn máy).
Độ tin cậy	- **Phụ thuộc vào kỹ thuật**: TCP tin cậy hơn UDP, nhưng vẫn là **truyền thông nhất thời** (có thể mất thông điệp). - **Hàng đợi thông điệp** hỗ trợ bền bỉ, không mất thông điệp khi bên nhận chưa sẵn sàng.	- Tương đối cao nếu kiểm soát truy cập tốt. - Tuy nhiên dễ xảy ra lỗi nếu có **xung đột truy cập**, **tranh chấp tài nguyên**.
Hiệu năng	- Hiệu năng thấp hơn vì cần sao chép thông điệp, truyền qua mạng. - Độ trễ cao nếu sử dụng MPI hoặc hàng đợi thông điệp.	- Hiệu năng cao, **truy cập trực tiếp vào bộ nhớ dùng chung** nhanh hơn. - Tốt hơn trong các hệ thống tính toán song song đơn máy.

Câu 4: a) Mục đích và bối cảnh ra đời của MPI

- **Mục đích**:
  - **MPI (Message Passing Interface)** ra đời nhằm **chuẩn hóa giao diện truyền thông điệp** cho các ứng dụng **xử lý song song và phân tán**.

- Cung cấp một **giao diện lập trình thống nhất, độc lập phần cứng**, có hiệu năng cao và hỗ trợ **truyền thông nhất thời** giữa các tiến trình chạy trên nhiều máy tính.

- Tạo ra **tính khả chuyển (portable)** giữa các hệ thống phần cứng mạng khác nhau.

- **Bối cảnh ra đời**:

  - Trước MPI, nhiều hệ thống truyền thông điệp được phát triển độc lập, không tương thích nhau, đặc biệt là trên **các kênh truyền tốc độ cao** có **trình điều khiển độc quyền** từ nhà sản xuất phần cứng.

  - Năm **1991**, các nhà nghiên cứu trong cộng đồng HPC (High Performance Computing) nhận thấy nhu cầu cần có một chuẩn chung, **MPI ra đời từ sự thảo luận và hợp tác của nhiều tổ chức và công ty**.

  - Dù **chưa được tổ chức tiêu chuẩn quốc tế xác nhận**, MPI được **rất nhiều hệ điều hành, ngôn ngữ lập trình và tổ chức phần mềm mã nguồn mở hỗ trợ**.

---

b) 4 hàm nguyên thủy cơ bản trong MPI

Hàm	Đồng bộ / Không đồng bộ	Cơ chế vùng đệm	Giải thích ngắn gọn
`MPI_Send`	Đồng bộ (có thể phong tỏa)	Đợi thông điệp được **lưu trong vùng đệm bên nhận** hoặc **MPI buffer**	Tiến trình gửi bị **chặn lại** cho đến khi hệ thống xác nhận đã lưu thông điệp an toàn.
`MPI_Bsend`	Không đồng bộ	Gửi vào **vùng đệm cục bộ (local buffer)**	Tiến trình gửi **không bị chặn**, có thể tiếp tục công việc khác sau khi gửi vào buffer.
`MPI_Ssend`	Đồng bộ mạnh	Chỉ gửi khi bên nhận đã **kích hoạt thao tác nhận**	Đảm bảo **đồng bộ chính xác** giữa người gửi và người nhận trước khi truyền.
`MPI_Irecv`	Không đồng bộ	Không sử dụng đệm nếu chưa có thông điệp	Tiến trình nhận chỉ **thông báo đã sẵn sàng**, **không bị chặn** nếu chưa có dữ liệu.

a) Giải thích mô hình Stub – Skeleton trong RPC

Câu 6: Mô hình Stub – Skeleton là gì?

Stub – Skeleton là kiến trúc trung gian **che giấu sự khác biệt giữa gọi thủ tục cục bộ và gọi thủ tục từ xa**, giúp lập trình viên **sử dụng thủ tục từ xa như thể nó được gọi trong cùng máy tính**.

Vai trò của từng thành phần:

Thành phần	Vai trò
Client	Gọi thủ tục từ xa giống như gọi hàm bình thường.
Stub (Client-side)	- **Đóng vai trò như một hàm thay thế** trên máy khách. - **Mã hóa (marshal)** tên thủ tục và tham số thành thông điệp gửi đến máy chủ. - **Giải mã (unmarshal)** kết quả trả về.
Hệ điều hành máy khách	Gửi thông điệp qua mạng đến máy chủ.
Hệ điều hành máy chủ	Nhận thông điệp và chuyển đến Skeleton.
Skeleton (Server-side)	- **Giải mã thông điệp (unmarshal)** từ Stub. - Gọi hàm tương ứng trên máy chủ và nhận kết quả. - **Mã hóa (marshal)** kết quả trả về cho máy khách.
Server	Chứa thủ tục thật sự được gọi. Thực hiện xử lý và trả kết quả về.

b) Phác họa tuần tự 10 bước của một cuộc gọi RPC

Tuần tự 10 bước của một lời gọi thủ tục từ xa RPC:

Bước	Mô tả	Giải thích ý nghĩa
1	Máy khách gọi thủ tục RPC (Stub)	Giống như gọi hàm bình thường, **ẩn việc gọi từ xa**.
2	Stub đóng gói (marshal) tên hàm và các tham số vào thông điệp	Chuyển đổi tham số thành dạng **dữ liệu có thể truyền đi qua mạng**.
3	Hệ điều hành máy khách gửi thông điệp đến máy chủ	Thực hiện **gửi thông điệp RPC** đến đúng địa chỉ IP và cổng.
4	Hệ điều hành máy chủ nhận thông điệp và chuyển đến Skeleton	Truyền thông điệp đến tiến trình dịch vụ phù hợp.
5	Skeleton giải mã (unmarshal) thông điệp và gọi thủ tục cục bộ tương ứng	Chuyển đổi lại thông điệp thành **tham số thực** và **gọi hàm thật**.

6	Server thực hiện thủ tục và trả kết quả	Thực hiện tính toán, đọc/ghi dữ liệu hoặc xử lý logic nghiệp vụ.
7	Skeleton đóng gói kết quả thành thông điệp	Chuẩn bị dữ liệu để gửi trả cho máy khách.
8	Hệ điều hành máy chủ gửi thông điệp kết quả về máy khách	Truyền trả thông điệp mạng từ máy chủ sang máy khách.
9	Hệ điều hành máy khách nhận và chuyển thông điệp đến Stub	Đảm bảo thông điệp kết quả đến đúng tiến trình máy khách.
10	Stub giải mã kết quả và trả về cho hàm gọi ban đầu	Trả dữ liệu về cho đoạn mã gọi, giống như trả về từ hàm cục bộ.

Câu 8: Đặc điểm và yêu cầu của truyền thông luồng:

a) Phân biệt giữa dữ liệu rời rạc và dữ liệu luồng (continuous data stream). Nêu ví dụ ứng dụng tương ứng

1. Dữ liệu rời rạc (discrete data)

- Là **dữ liệu không phụ thuộc thời gian**, các đơn vị dữ liệu có thể được xử lý một cách **độc lập** và **không yêu cầu trình tự hay thời gian chặt chẽ** khi truyền tải hoặc hiển thị.
- **Ví dụ đặc trưng:**
    - **Văn bản (Text):** Mã hóa bằng chuẩn ASCII hoặc Unicode.
    - **Ảnh tĩnh (Static Images):** Mã hóa theo chuẩn JPEG.
- **Tính chất:**
    - Cần **đảm bảo chính xác và thứ tự hiển thị**.
    - Nhưng **không yêu cầu tính liên tục hay thời gian thực**.

2. Dữ liệu luồng (continuous data stream)

- Là dữ liệu **phụ thuộc vào yếu tố thời gian**, được tổ chức thành **chuỗi các đơn vị dữ liệu rời rạc nhưng cần được thể hiện liên tục theo thời gian**.

- **Ví dụ đặc trưng:**

    - **Âm thanh:** Mã hóa theo chuẩn G.711.

    - **Video:** Mã hóa theo chuẩn H.265 hoặc MPEG-4.

    - **Cuộc họp trực tuyến (Multimedia Conference):** Truyền video, âm thanh và phụ đề theo thời gian thực.

- **Tính chất:**

    - Cần **duy trì tính liên tục (không bị gián đoạn)** khi truyền.

    - Phải đảm bảo **tính đúng đắn về thời gian**, như độ trễ và thứ tự.

b) Tại sao các ứng dụng đa phương tiện (âm thanh, video, phụ đề) lại cần đảm bảo cả tính liên tục và tính đồng bộ giữa các luồng con?

1. Tính liên tục là gì và tại sao cần thiết?

- **Tính liên tục** đảm bảo rằng **các đơn vị dữ liệu trong luồng** (frame âm thanh/video) **được phát một cách trơn tru, không bị gián đoạn**.

- Nếu không liên tục → **âm thanh bị đứt, hình ảnh bị giật, gián đoạn hiển thị**.

2. Tính đồng bộ là gì và tại sao cần thiết?

- **Tính đồng bộ** đảm bảo rằng **các luồng con như âm thanh, hình ảnh và phụ đề** được trình chiếu **đúng thời điểm và khớp với nhau**.

- Nếu không đồng bộ:

    - Âm thanh đến trước hoặc sau hình ảnh → gây **méo thông tin**.

    - Phụ đề hiển thị lệch → gây **hiểu sai nội dung**.

3. Tình huống thực tế cần cả hai:

- **Phim ảnh**: Có thể được đóng gói thành **1 luồng phức hợp**, trong đó:

    - Âm thanh, video, phụ đề đã **được đồng bộ trong quá trình mã hóa**.

- Hệ thống chỉ cần duy trì **tính liên tục khi truyền và phát**.

- **Hội nghị truyền hình trực tuyến**:
  - Các luồng âm thanh, video, phụ đề được **thu thập riêng biệt từ thiết bị ngoại vi**.
  - Phải vừa **đảm bảo tính liên tục**, vừa **giải quyết đồng bộ giữa các luồng** trong điều kiện thời gian thực.

Câu 10:

a) Định nghĩa truyền thông theo nhóm và ví dụ ứng dụng

Định nghĩa:

Truyền thông theo nhóm (group communication) là **phương pháp truyền dữ liệu từ một thành viên đến nhiều thành viên khác** trong cùng một nhóm, nhằm phổ biến thông tin đến nhiều điểm nhận với **thời gian lan tỏa nhanh nhất và lượng thông điệp lưu chuyển ít nhất**. Phương pháp này rất phổ biến trong các hệ thống phân tán và mạng máy tính hiện đại.

Ví dụ ba ứng dụng thực tế:

1. **Phát thanh trực tiếp (Live Broadcasting):**
   Một máy chủ phát sóng chương trình radio hoặc livestream gửi dữ liệu âm thanh đến nhiều người nghe cùng lúc.
   → Cần đảm bảo tính liên tục của luồng âm thanh và độ trễ thấp.

2. **Hội thảo video (Video Conference):**
   Trong hội nghị trực tuyến, dữ liệu video, âm thanh và phụ đề được gửi từ một hoặc nhiều điểm đến tất cả các thành viên tham gia.
   → Yêu cầu tính đồng bộ giữa các luồng dữ liệu (âm thanh, hình ảnh, phụ đề) và tính thời gian thực cao.

3. **Cập nhật cấu hình hệ thống (Configuration Updates):**
   Khi cập nhật phần mềm, bản vá lỗi hoặc cấu hình mới, máy chủ gửi thông điệp cập nhật đến toàn bộ các máy khách trong nhóm.
   → Yêu cầu truyền tải chính xác và đảm bảo mọi thành viên đều nhận được bản cập nhật.

b) So sánh ưu – nhược điểm của hai mô hình: mạng phủ và lan truyền

Tiêu chí	Mạng phủ (Overlay Tree)	Lan truyền ngẫu nhiên (Gossip)

Khái niệm chính	Tổ chức các thành viên thành cây phân cấp (tree), thường có một nút gốc và các nhánh con.	Thông điệp được phát tán theo kiểu “lời đồn” – mỗi nút chọn ngẫu nhiên nút khác để truyền tin.
Ưu điểm	- Dễ kiểm soát luồng dữ liệu - Ít trùng lặp thông điệp - Hạn chế băng thông sử dụng nếu xây dựng tối ưu - Phù hợp các hệ thống yêu cầu thứ tự và đồng bộ	- Khả năng mở rộng cao (scale tốt) - Không cần thông tin toàn cục - Dễ triển khai - Hội tụ nhanh với chi phí tính toán thấp
Nhược điểm	- Dễ bị lỗi nếu nút cha hoặc nút gốc bị hỏng - Định tuyến có thể không tối ưu nếu không phản ánh cấu trúc vật lý mạng - Quá tải tại nút gốc	- Lãng phí băng thông do trùng lặp thông điệp - Không đảm bảo chắc chắn mọi nút đều nhận thông tin nếu không bổ sung cơ chế kiểm tra
Hiệu năng lan truyền	Phụ thuộc vào chiều cao cây và cách tổ chức mạng phủ, trường hợp xấu có thể là `T * (N-1)` (với T là thời gian trung bình giữa hai nút).	Trung bình chỉ cần `O(log N)` vòng để lan tỏa thông điệp đến toàn bộ hệ thống.
Khả năng chịu lỗi	Thấp – lỗi tại các nút trung gian có thể làm gián đoạn truyền thông toàn nhóm.	Cao – lỗi tại một số nút không ảnh hưởng nhiều, các nút khác vẫn có thể tiếp tục truyền thông.
Ứng dụng phù hợp	- Phát sóng nội dung đồng bộ như video streaming, IPTV - Hệ thống yêu cầu định tuyến hiệu quả và kiểm soát được luồng dữ liệu	- Hệ thống lớn, phân tán, cần lan truyền nhanh: P2P updates, Blockchain gossip, quản lý metadata phân tán