I. Câu hỏi lý thuyết

1. Mutual Exclusion (Loại trừ tương hỗ)

- Loại trừ tương hỗ trong hệ phân tán là gì? Tại sao nó quan trọng?

Loại trừ tương hỗ (Mutual Exclusion) trong hệ phân tán là cơ chế đảm bảo rằng chỉ một tiến trình được phép truy cập vào tài nguyên dùng chung (critical section) tại một thời điểm.

Tầm quan trọng:

+ Đảm bảo tính nhất quán dữ liệu: Nếu nhiều tiến trình truy cập và chỉnh sửa dữ liệu cùng lúc, có thể dẫn đến sai lệch dữ liệu.

+ Tránh xung đột tài nguyên và lỗi hệ thống.

+ Giúp hệ thống hoạt động ổn định và đúng logic.

- So sánh các thuật toán loại trừ tương hỗ:

- Thuật toán dựa trên token (ví dụ: Token Ring)

- Thuật toán dựa trên message passing (ví dụ: Ricart-Agrawala)

- Thuật toán dựa trên server tập trung

- Ưu điểm và nhược điểm của các thuật toán trên là gì?

|  |  |  |  |
| --- | --- | --- | --- |
| Tiêu chí | Thuật toán Token Ring | Ricart-Agrawala (Message Passing) | Server Tập Trung |
| Cơ chế | |  | | --- | |  |  |  | | --- | | Token duy nhất được truyền theo vòng | | Gửi yêu cầu đến tất cả các tiến trình | Một server làm trung tâm điều phối |
| Thông điệp | 1 thông điệp/token được truyền tuần tự | 2(N−1) thông điệp (request + reply) | 2 thông điệp/request (request & grant) |
| Độ trễ | Trung bình: N/2 bước để nhận token | N−1 request + N−1 reply | Tối đa 2 bước |
| Ưu điểm | |  | | --- | |  |  |  | | --- | | Đơn giản, không cần đồng hồ logic | | Không cần server trung tâm, phân tán hoàn toàn | Hiệu suất cao, đơn giản |
| Nhược điểm | |  | | --- | |  |  |  | | --- | | Token mất -> cần tái tạo; chậm nếu N lớn | |  | | Giao tiếp nhiều, độ trễ cao nếu N lớn | Server là điểm nghẽn, nếu lỗi -> hệ thống dừng |
| Khả năng chịu lỗi | Trung bình, cần thêm xử lý nếu token mất | Cao hơn, không có điểm đơn lỗi | Thấp, phụ thuộc vào server trung tâm |

2. Election Algorithms (Thuật toán bầu cử)

- Mô tả thuật toán Bully trong bầu cử leader và nêu các bước chính.

Thuật toán Bully là thuật toán bầu chọn tiến trình trưởng (leader) dựa trên ID. Tiến trình có ID lớn nhất sẽ làm leader. Nó sẽ chon thằng lớn sau nó mà không biết thằng nào lớn nhất.

Các bước:

1. Tiến trình phát hiện leader hiện tại chết (không phản hồi).
2. Nó gửi Election đến tất cả tiến trình có ID lớn hơn.
3. Nếu không có phản hồi, tiến trình đó trở thành leader.
4. Nếu có tiến trình phản hồi (ID lớn hơn), tiến trình đó bị loại và chờ thông báo leader mới.
5. Cuối cùng, tiến trình thắng cuộc gửi Coordinator đến tất cả tiến trình khác.

- Thuật toán Ring Election hoạt động như thế nào?

Nguyên tắc:

* Các tiến trình được sắp theo vòng tròn logic (logical ring).
* Một tiến trình khởi xướng election, gửi thông điệp vòng quanh vòng.
* Mỗi tiến trình đính kèm ID của mình vào thông điệp.
* Khi thông điệp quay lại người khởi xướng, tiến trình có ID lớn nhất sẽ được chọn làm leader.
* Người khởi xướng thông báo leader mới tới các tiến trình khác.

- So sánh thuật toán Bully và thuật toán Ring Election. Thuật toán nào hiệu quả hơn?

|  |  |  |
| --- | --- | --- |
| Tiêu chí | Bully Algorithm | Ring Election |
| Giao tiếp | N² trong trường hợp xấu nhất | N thông điệp trong vòng |
| Hiệu suất | Nhanh hơn nếu ít tiến trình tham gia | Tối ưu hơn nếu tổ chức theo vòng chuẩn |
| Đơn giản | Phức tạp hơn | Dễ cài đặt hơn |
| Chịu lỗi | Có thể bị lỗi nếu các tiến trình lớn hơn không phản hồi | Khả năng phục hồi tốt hơn nếu vòng còn nguyên |

- Các điều kiện nào dẫn đến cần phải tổ chức bầu cử trong hệ phân tán?

* Tiến trình trưởng hiện tại chết (timeout không phản hồi).
* Trưởng tự động thoát hoặc bị lỗi phần mềm.
* Hệ thống mới khởi động, chưa có leader.
* Leader không phản hồi yêu cầu điều phối tài nguyên.

II. Bài tập thực hành

Bài 1: Cài đặt thuật toán Ricart-Agrawala

Thuật toán Ricart-Agrawala là một thuật toán loại trừ tương hỗ dựa trên message passing. Mỗi

tiến trình khi muốn vào vùng găng (critical section) sẽ gửi request đến tất cả tiến trình khác.

Cài đặt:

Mỗi tiến trình có một timestamp (dấu thời gian logic) khi gửi yêu cầu.

Nếu một tiến trình khác không ở vùng găng, nó sẽ chấp nhận (reply) ngay.

Nếu tiến trình nhận request đang ở vùng găng, nó sẽ trì hoãn (defer) phản hồi cho đến khi thoát

ra.

Khi một tiến trình nhận được tất cả phản hồi, nó vào vùng găng.

Khi thoát vùng găng, tiến trình gửi reply cho các request bị trì hoãn.

Yêu cầu

Cài đặt thuật toán với N tiến trình.

Hiển thị log quá trình gửi và nhận tin nhắn.

Đảm bảo chỉ một tiến trình vào vùng găng tại một thời điểm.

Bài 2: Mô phỏng Bully Election

Thuật toán Bully Election giúp chọn một leader mới khi leader cũ bị lỗi.

Cài đặt:

Một tiến trình phát hiện leader bị lỗi sẽ gửi yêu cầu bầu cử đến tất cả tiến trình có ID lớn hơn nó.

Nếu một tiến trình lớn hơn nhận được yêu cầu, nó sẽ phản hồi và khởi động một cuộc bầu cử mới.

Tiến trình có ID lớn nhất sẽ trở thành leader.

Yêu cầu

Cài đặt mô phỏng với N tiến trình, mỗi tiến trình có một ID.

Hiển thị quá trình gửi tin nhắn trong quá trình bầu cử.

Xác định leader mới sau mỗi lần leader cũ bị lỗi.

Bài 3: Mô phỏng Ring Election

Thuật toán Ring Election sử dụng cấu trúc vòng tròn để bầu leader mới.

Cài đặt:

Khi một tiến trình phát hiện leader bị lỗi, nó gửi một thông báo bầu cử theo vòng tròn.

Mỗi tiến trình thêm ID của mình vào thông báo và chuyển tiếp.

Khi thông báo quay lại tiến trình ban đầu, nó chọn ID lớn nhất làm leader.

Leader mới được thông báo cho tất cả tiến trình khác.

Yêu cầu

Mô phỏng với N tiến trình, trong đó leader có thể bị lỗi.

Hiển thị các tin nhắn bầu cử qua từng bước.

Bài 4: Mô phỏng Đồng hồ Lamport

Đồng hồ Lamport giúp duy trì thứ tự sự kiện trong hệ phân tán bằng cách gán timestamp cho từng

sự kiện.

Cài đặt:

Mỗi tiến trình có một biến thời gian logic.

Khi một tiến trình thực hiện sự kiện nội bộ, nó tăng thời gian của mình.

Khi gửi tin nhắn, nó đính kèm giá trị thời gian hiện tại.

Khi nhận tin nhắn, tiến trình cập nhật thời gian của mình để đảm bảo thứ tự sự kiện hợp lệ.

Yêu cầu

Hiển thị log quá trình gửi tin nhắn giữa các tiến trình với timestamp của chúng.

Bài 5: Mô phỏng Đồng hồ Vector

Đồng hồ Vector giúp duy trì thứ tự nhân quả bằng cách sử dụng một danh sách timestamp thay vì

chỉ một giá trị.

Cài đặt:

Mỗi tiến trình có một vector thời gian.

Khi tiến trình thực hiện một hành động, nó tăng giá trị của mình.

Khi gửi tin nhắn, nó đính kèm vector thời gian hiện tại.

Khi nhận tin nhắn, tiến trình cập nhật vector của mình bằng cách lấy giá trị lớn nhất giữa hai

vector.

Yêu cầu

Hiển thị log quá trình cập nhật đồng hồ vector.