Điều khiển tương tranh bằng cơ chế khóa (lock)

Điều khiển tương tranh bằng cơ chế khóa(lock) 1. Khái niệm khóa: Lock là một đặc quyền truy xuất (access priveleg) lên các đơn vị dữ liệu của các giao tác mà bộ quản lý khóa có thể trao cho một giao tác hay thu hồi lại. Khi một giao tác đã khóa (lock) trên một đơn vị dự liệu nào đó thì các giao tác khác không được phép truy cập đến đơn vị dữ liệu đó cho đến khi nó nhả khóa(unlock). Khi một giao tác T thực hiện việc lock đơn vị dữ liệu A, ta nói T đang giữ lock A. Thông thường tại mỗi thời điểm, chỉ có một tập con các đơn vị dữ liệu bị khóa, vì vậy bộ quản lý khóa có thể lưu các khóa hiện hành trong một bảng (lock table) với các mẫu tin có dạng: (A, L, T) (giao tác T có một khóa kiểu L trên đơn vị dữ liệu A); 2. Kỹ thuật khóa đơn giản a. Giới thiệu: - Một giao tác khi có yêu cầu truy xuất đến đơn vị dữ liệu thì phải phát ra yêu cầu xin khóa (lock) trên đơn vị dữ liệu đó, nếu yêu cầu này được chấp nhận thì được quyền thao tác, và như vậy các giao tác khác sẽ không được phép truy cập đến đơn vị dữ liệu đó cho đến khi giao tác giữ khóa được unlock. b. Ví dụ: Xét 2 giao tác T1 và T2. Mỗi giao tác truy xuất 1 đơn vị dữ liệu A được giả sử là mang giá trị số nguyên, rồi cộng thêm 1 vào A. Hai giao tác này là các thực hiện của chương trình P dưới đây: P: Lock (A) Read(A) A=A+1 Write (A) Unlock (A) Nếu T1¬ bắt đầu trước, nó yêu cầu khóa trên A. Giả sử rằng không có giao tác nào đang khóa A, bộ quản lý khóa sẽ cho nó khóa này. Như vậy bây giờ chỉ có T1 mới có thể truy xuất A. Nếu T2 bắt đầu trước khi T1 chấm dứt thì T2 phải đợi. Chỉ khi T1 thực hiện lệnh unlock(A), hệ thống mới cho phép T2 tiến hành. Kết quả là T1 hoặc T2 sẽ hoàn tất trước khi giao tác kia bắt đầu, và tác dụng là cộng 2 vào A. 3. Kỹ thuật khóa đọc/ viết (readlock/ writelock) - Nếu không phân biệt khóa cho thao tác đọc hay viết thì rõ ràng sẽ có nhiều giao tác phải chờ để được quyền khóa trên một đơn vị dữ liệu. Thực tế là nhiều khi một giao tác chỉ cần lấy giá trị của một đơn vị dữ liệu nhưng không thay đổi giá trị đó. Vì vậy để giảm bớt tình huống phải chờ khi các giao tác cùng đọc dữ liệu, người ta đề nghị tách yêu cầu khóa thành 2 yêu cầu khóa riêng biệt: Khóa để đọc (hay Shared lock): một giao tác T chỉ muốn đọc một đơn vị dữ liệu A sẽ thực hiện lệnh RLOCK(A), ngăn không cho bất kỳ giao tác khác ghi giá trị mới của A trong khi T đã khóa A. Tuy nhiên các giao tác khác vẫn có thể giữu 1 khóa đọc trên A cùng lúc với T. Khóa để ghi (hay Exclusive lock): một giao tác muốn thay đổi giá trị của một đơn vị dữ liệu A đầu tiên sẽ phải lấy khóa ghi bằng cách thực hiện lệnh WLOCK(A). Khi một giao tác đang giữ một khóa ghi trên một đơn vị dữ liệu, các giao tác khác không thể lấy được khóa đọc hoặc khóa ghi trên A cùng lúc với T. Cả hai khóa dọc và khóa ghi đều được loại bỏ bằng lệnh UNLOCK. - Điều kiện để xin khóa đọc/ viết: Một yêu cầu xin RLOCK(A) chỉ được chấp nhận nếu A chưa bị khóa bởi một WLOCK trước đó. Một yêu cầu xin WLOCK(A) chỉ được chấp thuận nếu A được tự do. - Ma trận tương thích các loại khóa: Lock đang được giữ R-lock W-lock Giao tác yêu cầu lock R-lock Yes No W-lock No No - Từ ma trận tương thích, chúng ta thấy: một đơn vị dữ liệu tại một thời điểm có thể có nhiều transaction giữ Rlock nhưng chỉ có tối đa một tracsaction giữ Wlock. - Việc sử dụng cơ chế lock không đủ để đảm bảo tính khả tuần tự cho lịch giao tác. 4. Một số vấn đề trong kỹ thuật khóa a. Livelock: - Hệ thống trao và buộc khóa các đơn vị dữ liệu không thể hoạt động một cách thất thường, nếu không thì các hệ thống không mong muốn có thể xảy ra. Giả sử như ví dụ trên, khi T1 giải phóng khóa trên A, khóa này lại trao lại cho T2. Điều gì sẽ xảy ra nếu như trong khi T2 đang đợi nhận khóa, một giao tác T3 khác cũng xin khóa trên A, và T3 lại được trao khóa này trước T2. Rồi sau khi T3 được trao khóa trên A thì lại có một giao tác T4 xin khóa trên A...và rất có thể T2 sẽ phải đợi mãi và chẳng bao giờ nhận được khóa trong khi luôn có một giao

[In trang] [Đóng]