Dịch vụ CDN (Content Delivery Network) là gì?

Tại sao cần cơ chế Sao lưu và Thống nhất dữ liệu trong các hệ thống phân tán?

Trình bày về các mô hình thống nhất hướng dữ liệu và

Trình bày về các mô hình thống nhất hướng người dùng.

Đặt vấn đề

Hiện nay, một trong những bài toán thách thức nhất đối với những nhà phát triển hệ thống mạng máy tính chính là vấn đề giải quyết lưu lượng trên mạng. Khi số lượng người dùng tăng lên thì số lượng request tới server cũng tăng lên tương ứng, nếu giải quyết không tốt bài toán này sẽ dẫn đến tình trạng nghẽn mạng thường xuyên và có thể đem lại thiệt hại kinh tế không nhỏ. Sự ra đời của công nghệ CDN (Content Delivery Network) là để giải quyết tình trạng “thắt cổ chai” giữa client và server. Các nhà cung cấp dịch vụ mạng đã phát triển công nghệ CDN với mục đích chính nâng cao băng thông, ngoài ra còn có ưu điểm tạo ra các server dự phòng lập tức hoạt động nếu không may một server nào đó bị gặp sự cố.

Vậy CDN chính xác là cái gì, có bao nhiêu loại CDN, và sử dụng nó như thế nào. Trong nội dung bài tiểu luận này em sẽ tìm hiểu về các vấn đề sau:

* Khái niệm về dịch vụ CDN
* Tại sao cần cơ chế sao lưu và thống nhất dữ liệu trong các hệ thống phân tán
* Các mô hình thống nhất hướng dữ liệu
* Các mô hình thống nhất hướng người dùng

Chương 1 – Tìm hiểu về dịch vụ CDN

CDN là gì?

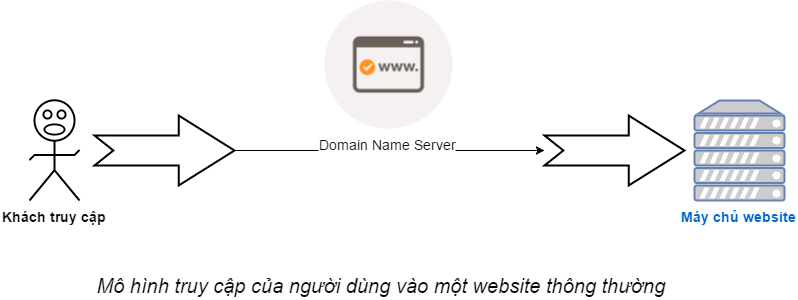


CDN là chữ viết tắt của Content Delivery Network, dịch sang tiếng Việt là “mạng phân phối nội dung”. Nghĩa là nó là một hệ thống máy chủ trên toàn cầu (số lượng tùy theo mỗi nhà cung cấp dịch vụ) làm nhiệm vụ lưu bản sao của các nội dung tĩnh bên trong website, sau đó phân tán nó ra nhiều máy chủ khác (được gọi là POP – Points of Presence) và từ đó các PoP đó sẽ gửi tới cho người dùng khi họ truy cập vào website.

Chúng ta xem qua mô hình khác nhau giữa một website không sử dụng CDN và một website có sử dụng CDN.

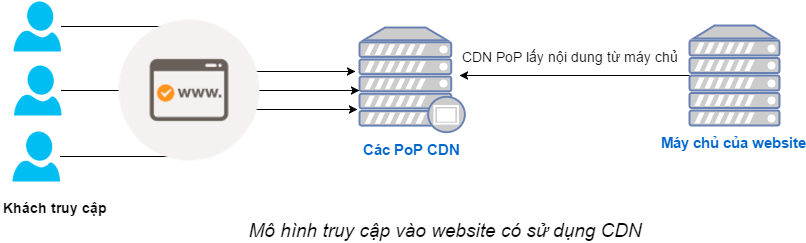
Website không sử dụng CDN

Khi người dùng xem một tập tin mà không có CDN, nghĩa là họ đã gửi một request thẳng đến máy chủ chứa website để truy cập thông tin đó.



Website sử dụng CDN

Khi một tập tin được phân phối bởi CDN, người dùng truy cập vào nó thì PoP phân phối gần nhất so với người dùng sẽ trả nội dung về cho người dùng xem. Ví dụ, khi truy cập vào một website tại Việt Nam thì PoP CDN đặt ở Việt Nam sẽ phân phối nội dung cho người dùng.



Ưu điểm khi dùng CDN

Một số ưu điểm của CDN là:

Tiết kiệm băng thông cho máy chủ gốc

Băng thông từ mạng của máy chủ gốc chỉ tốn một lần xử lý đó là chấp nhận request từ các PoP CDN, sau đó các lượt truy cập từ người dùng sẽ chỉ truy cập vào nội dung trên CDN nên máy chủ gốc sẽ không tốn thêm. Chỉ khi nào các bản lưu nội dung trên CDN bị xóa đi thì các PoP CDN sẽ tiến hành lấy lại nội dung lần nữa.

Tăng tốc lượt truy cập

Do tính chất các PoP CDN trải dài trên khắp các châu lục nên nó sẽ giúp website truy cập nhanh hơn đối với các người dùng ở xa máy chủ của website. Ví dụ một website đặt máy chủ ở Mỹ mà nếu truy cập trực tiếp để xem một hình ảnh 300KB ở Việt Nam sẽ mất khoảng 0.5 giây (500ms). Thế nhưng nếu website đó dùng CDN cho các nội dung tĩnh đó có hỗ trợ PoP tại Việt Nam thì người dùng chỉ mất 0.01 giây (10ms) để xem một tập tin, thậm chí còn nhanh hơn thế.

Tương tự với các người dùng ở quốc gia khác, CDN của ban càng có nhiều PoP ở nhiều quốc gia khác nhau thì càng có lợi trong việc tăng tốc website toàn cầu.

Tiết kiệm dung lượng

Nếu bạn có sử dụng phương thức Push CDN thì sẽ tiết kiệm được dung lượng lưu trữ cho máy chủ vì mọi thứ đã được upload lên thẳng máy chủ CDN. Tuy nhiên để an toàn bạn nên lưu lại nội dung ở một nơi nào đó đề phòng dịch vụ CDN có vấn đề.

Tiết kiệm chi phí

Chi phí tiết kiệm ở đây là chi phí băng thông. Giả sử máy chủ hoặc gói host chỉ hỗ trợ một khoảng băng thông cho phép mỗi tháng nhất định thì khi hết bạn sẽ cần mua thêm hoặc nâng cấp băng thông. Giá băng thông thấy vậy chứ không hề rẻ, giá băng thông trung bình hiện nay trên một số nhà cung cấp host là khoảng 20.000 đồng cho mỗi GB, tương đương $0.88. Nhưng các dịch vụ CDN hiện nay đa phần sẽ có giá là khoảng $0.05 hoặc rẻ hơn, ở một số PoP Châu Á nếu có đắt hơn thì cao lắm cũng khoảng $0.1 cho mỗi GB băng thông. Vậy thì thay vì phải mua thêm băng thông ở host thì dùng CDN sẽ tiết kiệm hơn rất nhiều.

Khi nào cần dùng DCN?

CDN có rất nhiều lợi ích khi sử dụng và nó là một trong những yêu cầu mà nhiều website phải sử dụng. Nhưng không phải website nào cũng cần thiết để sử dụng, mà CDN chỉ thật sự hữu ích khi:

Máy chủ của website đặt xa người dùng.

Lượt truy cập lớn tốn nhiều băng thông.

Có nhiều lượt truy cập trên nhiều quốc gia khác nhau.

Khi sử dụng kỹ thuật Load Balancing FailOver: đảm bảo cân bằng tải, sự ổn định của website; khi một node bị sự cố, giao dịch sẽ được tiếp tục phục vụ trên một node khác trong cluster mà client không phải thực hiện lại từ đầu.

Khi nào chưa cần thiết dùng CDN?

Trong nhiều trường hợp lập trình viên không cần thiết phải dùng CDN vì không những nó không giúp website nhanh hơn mà còn chậm hơn. Và một trong những lý do phổ biến nhất là máy chủ của website đặt ở gần người dùng. Ví dụ một website có máy chủ tại Việt Nam và phục vụ người dùng tại Việt Nam, nhưng nó sử dụng CDN mà nếu CDN đó không có PoP tại Việt Nam thì website đó sẽ chậm hơn vì lúc đó người dùng sẽ truy cập ở các PoP khác xa hơn so với máy chủ gốc hiện tại nên thành ra chậm hơn.

Tương tự với dịch vụ proxy CDN như CloudFlare, website đặt ở máy chủ tại Việt Nam mà dùng nó thì người dùng ở Việt Nam sẽ vào chậm hơn so với bình thường.

Những lưu ý khi lựa chọn dịch vụ CDN

Đặt bản thân vào vị trí một lập trình viên web tìm kiếm giải pháp tiết kiệm băng thông máy chủ và tăng tốc độ website, sử dụng CDN là một lời giải. Tuy nhiên, những điểm gì cần lưu ý khi xem xét các nhà cung cấp dịch vụ?

Đầu tiên, khi lập trình viên tiến hành tìm chọn một dịch vụ CDN tốt và phù hợp để sử dụng lâu dài thì nên lưu ý đến hệ thống PoP của nhà cung cấp. Tốt nhất nên ưu tiên chọn các dịch vụ CDN có hỗ trợ PoP ở quốc gia mà website đó có nhiều người dùng nhất. Ví dụ, khi website tập trung cho thị trường Việt Nam thì nên ưu tiên các dịch vụ CDN có PoP tại Việt Nam như CDNSun. Mỗi nhà cung cấp đều có mục Network để giới thiệu hệ thống PoP của họ để khách hàng tham khảo.

Cái quan trọng thứ hai là giá cả và hình thức thanh toán. Các dịch vụ CDN hiện nay hỗ trợ hai kiểu thanh toán là dùng bao nhiêu trả bấy nhiêu hoặc trả phí cố định hàng tháng để sử dụng theo gói. Nếu website không tốn nhiều băng thông thì nên chọn hình thức thanh toán thứ nhất sẽ tiết kiệm hơn và ngược lại.

Lưu ý cuối cùng chính là tốc độ và tính ổn định của server. Có thể kiếm chứng bằng cách dùng thử dịch vụ và sử dụng dịch vụ khác (ví dụ <https://asm.ca.com/en/ping.php>) định kỳ tiến hành ping để đánh giá tốc độ thực tế và tính ổn định của server 24/7.

Một số hình thức CDN

Khi sử dụng một số dịch vụ CDN (không tính CloudFlare hay các dịch vụ proxy CDN tương tự) thì họ hỗ trợ một số kiểu sử dụng CDN bao gồm:

Pull HTTP/Static

Kiểu này nghĩa là bạn khai báo tên miền của website cần sử dụng CDN hoặc IP của máy chủ. Sau đó các PoP CDN sẽ tự động truy cập tới website theo tên miền đó và tự lưu lại bản sao toàn bộ nội dung tĩnh bên trong website (các hình ảnh, tập tin CSS, tập tin Javascript, Flash, Video,….). Và sau đó bạn có thể truy cập một tập tin nào đó trên website với đường dẫn CDN mà họ cung cấp hoặc sử dụng một tên miền riêng cho CDN. Ví dụ:

Tập tin gốc: https://thachpham.com/wp-content/uploads/2015/11/with-cdn.png

Tập tin CDN: http://static.thach.io/wp-content/uploads/2015/11/with-cdn.png

POST/PUSH/PUT/Storage CDN…

Cái này có thể được gọi với nhiều cái tên khác nhau. Nhưng nó có một điểm chung là thay vì các PoP CDN sẽ tự thu thập nội dung ở website thì bạn sẽ tải thẳng các nội dung cần phân phối qua CDN lên máy chủ của họ qua các giao thức phổ biến như FTP hoặc HTTP. Thường thì hiện nay họ hỗ trợ FTP là nhiều nhất.

Và với phương thức phân phối này, bạn sẽ có thể tiết kiệm được không gian lưu trữ trên máy chủ vì không có lưu gì ở đó cả.

Streaming CDN

Mặc dù các kiểu CDN ở trên đều có hỗ trợ tập tin video nhưng nó lại không hỗ trợ phát live trực tiếp video (streaming). Vì vậy phương thức này sẽ giúp CDN phân phối nội dung streaming từ máy chủ và sau đó nó phân phối lại cho người dùng xem để tiết kiệm băng thông từ máy chủ streaming gốc. Hoặc bạn có thể lựa chọn cách tải thẳng nội dung streaming lên máy chủ CDN giống như Push CDN.

Một số dịch vụ CDN

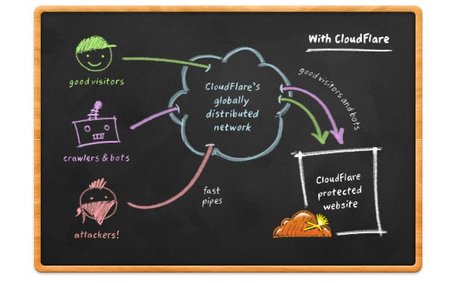
Hiện tại có nhiều dịch vụ CDN phổ biến như Amazon CloudFront, CDN.Net, MaxCDN,… trong mục này, em xin trình bày về hai dịch vụ là CloudFlare (miễn phí) và CDNSun (có hỗ trợ PoP tại Việt Nam)

CloudFlare

CloudFlare là một dịch vụ CDN và DDNS kết hợp để gia tăng tốc độ và tính bảo mật cho website.

Phương thức hoạt động

Khi một website trở thành một phần của CloudFlare, mọi truy xuất đến website đó sẽ được định tuyến qua hệ thống của CloudFlare. Tại đây CloudFlare caching các dữ liệu tĩnh như image, js, css,… để giúp tiết kiệm tối đa băng thông và qua đó tối ưu tốc độ tải trang để đạt hiệu suất cao nhất.



CloudFlare vừa đóng vai trò là DNS có trách nhiệm ánh xạ domain name thành IP, vừa đóng vai trò như một Proxy Server. Proxy này sẽ sàng lọc các truy vấn dựa vào cơ sở dữ liệu sẵn có để kiểm soát và đánh giá một yêu cầu trước khi chuyển đến điểm cuối cùng là hosting nơi chứa website. Các truy vấn bị nghi ngờ tấn công hệ thống của hacker, spam bot,… sẽ bị ngăn chặn và loại bỏ.

Cách đăng ký và sử dụng

Để sử dụng CloudFlare đầu tiên ta tạo tài khoản ở cloudflare.com , thêm tên miền vào và tiến hành cài đặt. Sau đó CloudFlare sẽ cho cung cấp cho thành viên hai nameserver dạng như bên dưới, chúng ta cứ hiểu đơn giản như là chuyển sang sử dụng dịch vụ DNS của CloudFlare cho dễ hình dung :

mary.ns.cloudflare.com

noah.ns.cloudflare.com

Ưu điểm

Cái dễ nhận ra nhất mà chúng ta thấy trước mắt là tiết kiệm băng thông, đây là do CloudFlare đã cache 1 phần website thay cho web-hosting của bạn đồng thời giảm khả năng highload cho host của bạn.

Tiết kiệm băng thông: lượng băng thông giảm hẳn chỉ còn 1/2 hoăc 1/3 so với trước khi dùng.

Cập nhật DNS: rất nhanh, chỉ trong vài phút.

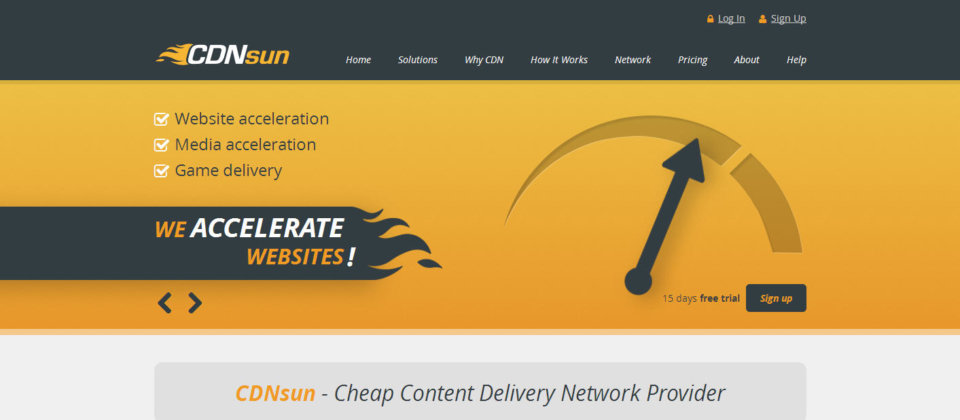
Bất lợi

Nếu website của bạn có máy chủ đặt tại Việt Nam, khách truy cập cũng chủ yếu đến từ Việt Nam thì việc sử dụng CloudFlare làm chậm đi tốc độ tải trang của bạn do ảnh hưởng bởi chất lượng đường truyền quốc tế.

Uptime website của bạn ngoài phụ thuộc vào máy chủ hosting còn bị phụ thuộc vào độ ổn định của CloudFlare.

CDNSun

Dịch vụ CDN hội tụ đủ các yếu tố như: giá thành rẻ ($0.045/GB); hỗ trợ POP tại Việt Nam; hỗ trợ POP quốc tế; tốc độ cao

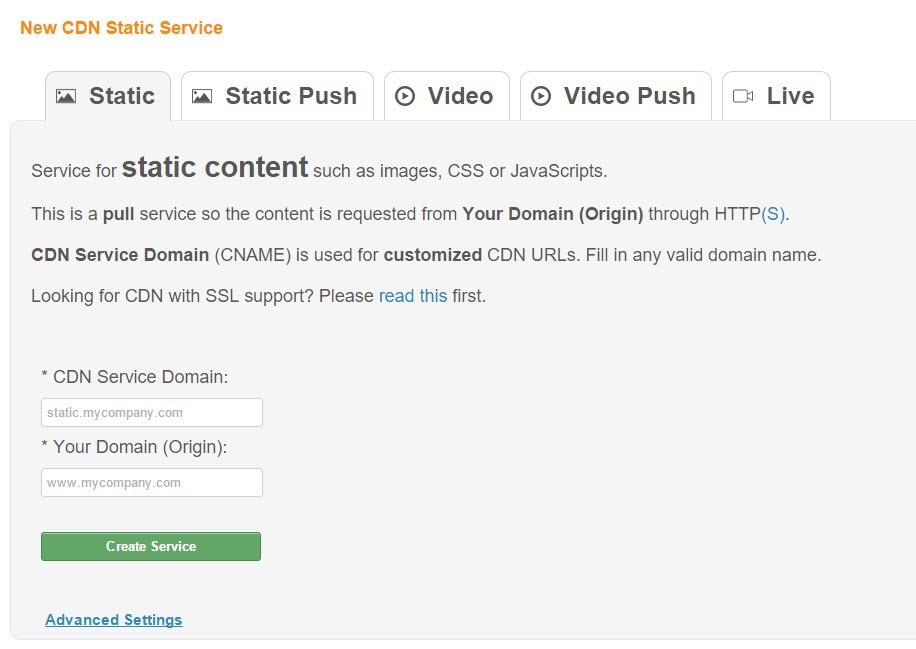


Mạng lưới PoP của CDNSun

Hỗ trợ mạng lưới trên toàn cầu với hơn 70 PoPs bao gồm 32 PoP tại châu Âu, 24 PoP tại Mỹ và 15 PoP tại châu Á (bao gồm 2 PoP tại Việt Nam)

Phương thức hỗ trợ

CDN hỗ trợ các phương thức như: Static (HTTP Pull); Static Push; Video; Video Push; Live



Ngoài ra CDNSun còn hỗ trợ các tùy chọn nâng cao, ứng dụng thống kê qua các biểu đồ, và linh hoạt quyền tùy chọn các PoP giá thành thấp để tiết kiệm chi phí.

Tham khảo

<https://thachpham.com/hosting-domain/cdn-la-gi-va-khi-nao-nen-dung-cdn-cho-website.html>

<https://viettelidc.com.vn/nhung-dieu-can-biet-ve-mang-phan-phoi-noi-dung-cdn.html>

<http://wiki.matbao.net/Ban-hieu-nhu-the-nao-ve-CloudFlare.ashx>

Chương 2. Cơ chế sao lưu và thống nhất dữ liệu trong các hệ phân tán (Consistency and Replication)

Vì sao phải thực hiện sao lưu dữ liệu?

Liên quan tới độ tin cậy: giúp tăng tính sẵn sàng của hệ thống. Khi có nhiều bản sao thì chúng có thể backup lẫn nhau trong trường hợp có bản sao bị hỏng hoặc bị treo.

Liên tới vấn đề hiệu năng: người dùng có thể chọn các bản sao gần nhất để yêu cầu phục vụ dịch vụ, từ đó khiến hệ thống có chi phí kết nối rẻ nhất có thể.

Liên quan tới khả năng co giãn: hệ thống có thể phân tải cho các bản sao để tăng khả năng phục vụ cho hệ thống. Tuy nhiên nó có nhược điểm là việc phân tải tốn thời gian và phải đầu tư cho thuật toán để thực hiện tác vụ đó.

Ví dụ về sao lưu: sao lưu giữa các server, sao lưu với client.

Yêu cầu đặt ra là tất cả các dữ liệu phải giống nhau ở các bản sao. Từ đó ta có yêu cầu về tính thống nhất đối với các bản sao trong hệ thống. Các bản sao cần có chung một dữ liệu, tuy nhiên hệ thống không thể tức thì đồng bộ dữ liệu giữa các bản sao với nhau. Câu hỏi đặt ra là đồng bộ khi nào và như thế nào?

Khái niệm về tính thống nhất mạnh và tính thống nhất yếu

Tính thống nhất yếu (lỏng): không yêu cầu tất cả bản sao lập tức phải thống nhất với nhau ngay sau khi cập nhật.

Tính thống nhất mạnh (Chặt): yêu cầu, dẫn đến tốn kém về hiệu năng.

Ví dụ: bộ nhớ đệm (cache) của trình duyệt Web. Để đảm bảo tính thống nhất, server cập nhật cache khi nội dung có sự thay đổi. Cần phải chọn một mô hình thống nhất hợp lý

Ưu và nhược điểm của Sao lưu

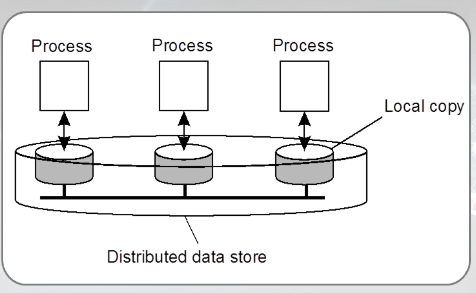
Giúp cải thiện hiệu năng về thời gian, đảm bảo tính co giãn của hệ thống và giảm băng thông sử dụng

Để đảm bảo tính thống nhất mạnh dẫn đến tốn băng thông của hệ thống mạng. Xuất hiện hiện tượng chờ đợi lẫn nhau. Tính thống nhất mạnh và yếu phụ thuộc vào sự áp dụng mô hình sao lưu nào.

Mô hình thống nhất là mô hình trừu tượng các yêu cầu thống nhất. Các yêu cầu thống nhất dựa trên nhiều tiêu chí đánh giá: giá trị thực tế các dữ liệu, thứ tự các thao tác dữ liệu…

Các mô hình thống nhất hướng dữ liệu (Data-centric consistency models)

Kho dữ liệu (data store): là mô hình bộ nhớ chia sẻ dùng chung bao gồm các thao tác đọc, ghi vào các vùng bộ nhớ. Trong hệ phân tán, dữ liệu được phân tán trên các máy tính khác nhau, tập hợp các bản sao cục bộ tạo thành kho dữ liệu.



Các thao tác trên kho dữ liệu bao gồm

+ đọc dữ liệu: không làm thay đổi dữ liệu

+ ghi dữ liệu: thay đổi dữ liệu

Mô hình thống nhất dùng để đặc tả khả năng đáp ứng của kho dữ liệu phân tán về tính thống nhất. Nó bao gồm thỏa thuận ràng buộc giữa tiến trình và kho dữ liệu. Tiến trình sử dụng dữ liệu trong kho dữ liệu, kho dữ liệu đảm bảo cung cấp dữ liệu một cách thống nhất theo mô hình thống nhất. Ví dụ, khi tiến trình đọc dữ liệu thì kết quả trả về là dữ liệu của thao tác ghi cuối cùng.

Các yêu cầu của mô hình thống nhất

Mức độ thống nhất: kho dữ liệu đảm bảo khi tiến trình đọc, dữ liệu trả về luôn là dữ liệu của thao tác ghi cuối cùng. Xác định thao tác ghi cuối cùng, nếu mức độ thống nhất lỏng thì thao tác này thực hiện cục bộ (ràng buộc thao tác đọc nhiều, dễ thực hiện), nếu mức độ thống nhất chặt thì thực hiện toàn bộ (ràng buộc thao tác đọc ít, khó thực hiện).

Mô hình thống nhất liên tục (Continuous Consistency)

Dựa trên giá trị của dữ liệu và các thông số trên bản ghi cục bộ để xác định mức độ thống nhất. Các giá trị có thể là:

Giá trị dữ liệu (biến nhiệt độ, giá cả): độ lệch trung bình không vượt quá giá trị thỏa thuận.

Thời gian cập nhật cuối cùng: chênh lệch giữa thời gian cập nhật của các bản sao cục bộ không vượt quá giá trị đã thỏa thuận.

Thứ tự thực hiện các thao tác: chênh lệch, sai khác giữa các thứ tự thực hiện các dữ liệu trên các bản sao khác nhau không vượt quá giá trị thỏa thuận.

Khi độ lệch vượt quá giá trị cho trước, thì middleware sẽ cập nhật giá trị cho phù hợp, với giá trị sai khác thì cập nhật giá trị cho trước, nếu thời gian sai khác lớn thì cập nhật lại giá trị mới, nếu thứ tự sai khác quá nhiều thì sẽ thực hiện rollback và thực hiện lại đúng với thứ tự cần thiết.

Khái niệm Conit (Consistency Unit)

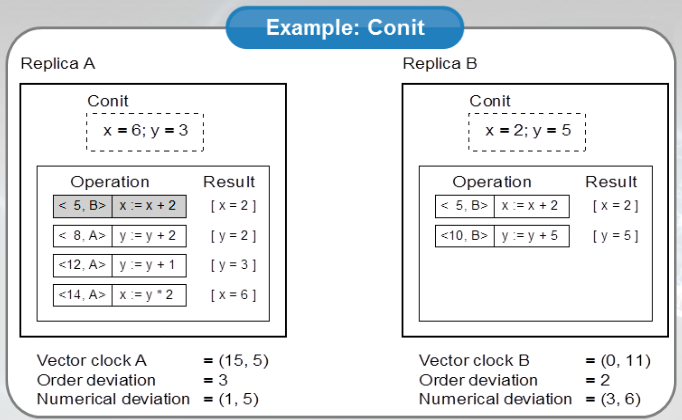
Dùng để định nghĩa mức độ không thống nhất, từ đó đánh giá độ thống nhất.

Conit là tập các đơn vị dữ liệu của kho dữ liệu phân tán. Mức độ thống nhất được xác định theo nhiều góc độ khác nhau

+ góc độ số lượng thao tác đã được thực hiện nhưng chưa hoàn thành, số lượng các thao tác chưa được thực hiện trên phần tử dữ liệu, về sai khác của dữ liệu đã hoàn thành với dữ liệu sẽ hoàn thành

+ Kich thước conit, nếu conit kích thước lớn thì sẽ làm tăng số lần cập nhật, conit kích thước nhỏ sẽ làm giảm số lần cập nhật.

Ví dụ về Conit:



Xét bản sao A và B. Conit gồm 2 phần tử (x, y). Xuất phát điểm x = y = 0, t = 0.

Tại A thực hiện bốn thao tác, trong đó có một thao tác nhận được từ B (gọi là quá trình phổ biến cập nhật). A có thời gian của B là 5, B có thời gian của A là 0.

+ Thời gian vô hướng của A là 15. Thời gian vector của A là (15, 5).

+ Tại B thực hiện 2 thao tác, không có thao tác nào từ A. Thời gian cục bộ của B là 11, thời gian vecto của B là (11, 0).

Liên quan đến thứ tự cập nhật (order deviation), ở bản sao A sẽ nhận được thao tác x = x + 2 của B. A thực hiện và ghi kết quả bền vững L. 3 thao tác đã được cập nhật ở bản sao cục bộ của A và chưa được cập nhật ở bản sao cục bộ của B. Vậy sai số về thứ tự được thực hiện ở A là 3.

Ở bản sao B, một thao tác được thực hiện ở B, một thao tác khác chờ thứ tự được thực hiện ở A, vậy sai số về thứ tự thực hiện ở B là 2.

Liên quan đến sai số về giá trị (numerical deviation), bao gồm hai giá trị, giá trị đầu tiên là sai khác lớn nhất về số thao tác thực hiện trên các phần tử khác nhau, giá trị thứ hai là sai khác lớn nhất về giá trị của các phần tử dữ liệu.

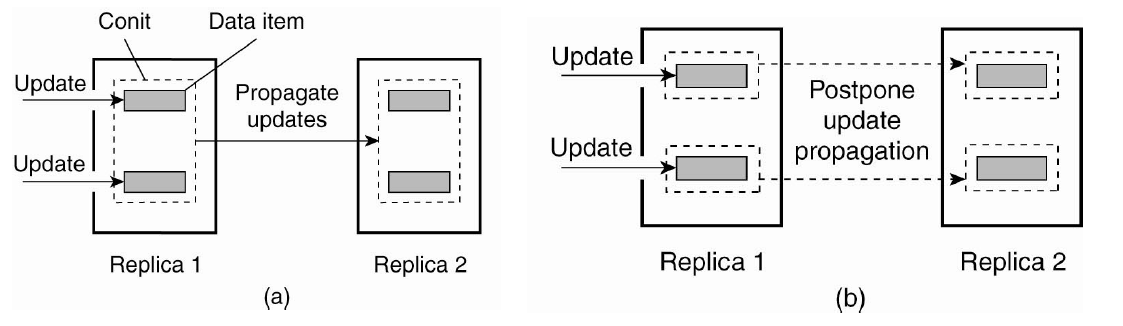
+ Dựa trên số thao tác đã thực hiện, A sẽ là 1 vì có một thao tác trên B mà A chưa nhận được. Dựa trên giá trị (x, y) ở A sẽ bằng (2, 0) trong khi thao tác được thực hiện bởi B mà A không thấy gán y = 5. Vậy giá trị ở A là (1, 5).

+ Giá trị thứ nhất ở B là 3 vì có ba thao tác trên A mà B chưa nhận được. Giá trị của x tại B là 0, tại A x là 6. Vậy B là (3, 6).

Kích thước của Conit

Kích thước của Conit cần phải được lựa chọn khi cài đặt. Số lượng Conit ít, kích cỡ Conit lớn thì khả năng xuất hiện sai lệch cao dẫn đến chi phí cao, đặc biệt không hiệu quả với các dữ liệu độc lập. Ví dụ sau khi có thao tác cập nhật thứ hai thì cần tiếp tục cập nhật bản sao 2 một lần nữa. (Event a)

Nếu số lượng Conit nhiều, kích thước Conit nhỏ thì sẽ giảm khả năng xuất hiện sai lệch, tuy nhiên chi phí quản lý các bản sao lớn. (Event b)



Mô hình thống nhất theo thứ tự thao tác (Consistent Ordering of Operations)

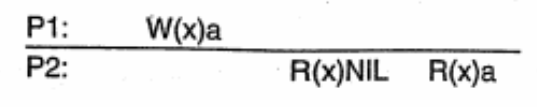
Là mở rộng của mô hình thống nhất liên tục.

Bài toán đặt ra là đạt được thống nhất đến các dữ liệu được chia sẻ, cụ thể là các dữ liệu được sao lưu. Các dữ liệu được chia sẻ sao lưu có thể là các dữ liệu tập trung hoặc phân tán. Khi thực hiện các thao tác, các bản sao phải thống nhất với nhau về một thứ tự thực hiện vậy sẽ dẫn đến thống nhất với nhau về dữ liệu.

Với các quá trình thực hiện khác nhau, tất cả các tiến trình luôn luôn cho một kết quả.

Các thao tác trên dữ liệu bao gồm:

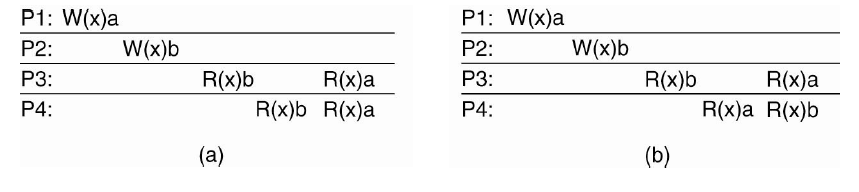
* Đọc R(x)b
* Ghi W(x)a
* Giá trị khởi tạo của các dữ liệu là NIL



Trong ví dụ trên, tiến trình P1 thực hiện ghi dữ liệu W(x)a. Thao tác này được cập nhật trên bản sao cục bộ của tiến trình P1, sau đó được phổ biến tới tiến trình P2. Tiến trình P2 đọc được dữ liệu NIL kể cả khi tiến trình P1 đã ghi vì cần có thời gian để thay đổi được cập nhật tới P2. Sau đó tiến trình P2 đọc dữ liệu a. Thao tác ghi được tất cả các tiến trình nhìn thấy theo cùng một thứ tự.

Mô hình thống nhất tuần tự (Sequential Consistency)

Các tiến trình Pi đều có một chuỗi thao tác Ci cục bộ trên một phần tử dữ liệu. Các thao tác này khi được thực hiện sẽ được tổng hợp thành chuỗi Cg trên kho dữ liệu. Nếu tất cả các Ci thuộc Cg thì sẽ thỏa mãn tính chất tuần tự.



Ở hình a, thứ tự toàn cục sẽ là W(x)b > R(x)b > W(x)a > R(x)a, thỏa mãn thống nhất tuần tự. Tiến trình P1 cần có một cơ chế chờ đợi để đảm bảo khi b được cập nhật trên P3 và P4 thì mới được thực thi.

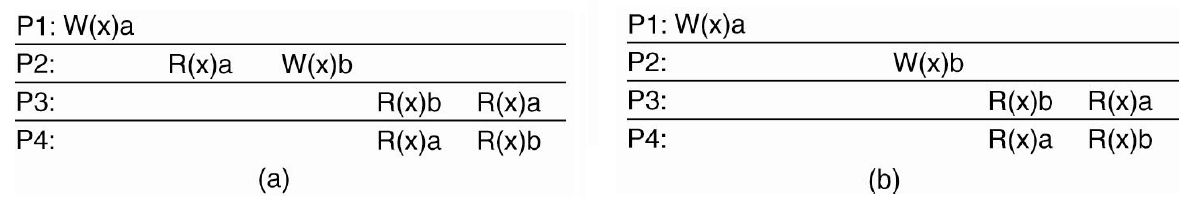
Ở hình b, không thể tìm được một thứ tự toàn cục để thứ tự cục bộ của P3, P4 được bảo toàn nên không đảm bảo được thống nhất tuần tự.

Mô hình thống nhất nhân quả (Causal Consistency)

Để đạt được thống nhất nhân quả cần tuân theo các quy tắc sau:

+ Các thao tác ghi có quan hệ nhân quả được thực hiện theo cùng thứ tự

+ Các thao tác ghi song song thì không bắt buộc thực hiện theo cùng thứ tự



Ở hình a, tiến trình P2 có thao tác R2(x)a trước thao tác W2(x)b, nên W2(x)b có khả năng phụ thuộc vào W1(x)a của tiến trình P1, vì vậy 2 thao tác ghi có quan hệ nhân quả. Tuy nhiên thứ tự đọc ở P3 và P4 lại khác nhau nên không thỏa mãn nhân quả.

Ở hình b, không còn thao tác đọc R2(x)a nên W1(x)a và W2(x)b được coi là song song (tương tranh). Thống nhân nhân quả không đòi hỏi hai thao tác tương tranh phải thống nhất toàn cục, vì vậy mô hình này thỏa mãn thống nhất nhân quả, tuy nhiên không thỏa mãn thống nhất tuần tự.

Thống nhất nhân quả yếu hơn thống nhất tuần tự.

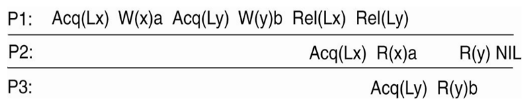
Thống nhất tuần tự và nhân quả là sản phẩm của bộ nhớ chia sẻ dùng chung, phù hợp với các quan hệ điểm điểm (point-to-point), và bắt buộc lập trình viên phải thiết kế các giao thức, trong đó một giao thức được sử dụng rộng rãi là sử dụng đoạn găng:

+ Đoạn gang được xác định cho từng thành phần dữ liệu, nên chia dữ liệu thành các mảnh để quản lý

+ Chỉ được truy cập đoạn găng khi tất cả các thao tác cập nhật đã hoàn tất

+ Chỉ được truy cập đoạn găng để ghi khi không có tiến trình nào giữ quyền truy cập

+ Trước khi truy cập đoạn găng để đọc, cần kiểm tra với chủ thể đoạn găng về tính cập nhật của dữ liệu



Tiến trình P2 yêu cầu truy cập Acq(Lx) thì phải chờ tới khi tiến trình P1 giải phóng giá trị của x Rel(Lx), khi P2 đọc giá trị R(y) sẽ có giá trị NIL.

Phân biệt tính thống nhất (Consistency) và phù hợp (Coherence)

Tính phù hợp: áp dụng cho một đơn vị dữ liệu; đảm bảo một giá trị được viết bởi một tiến trình sẽ được nhìn thấy bởi các tiến trình khác, nhưng không đảm bảo khi nào.

Tính thống nhất: áp dụng cho một tập hợp phần tử dữ liệu; đảm bảo ở những nơi khác nhau, các tiến trình khác nhau sẽ được nhìn thấy theo một thứ tự có ý nghĩa.

Mô hình thống nhất hướng người dùng (Client-centric consistency models)

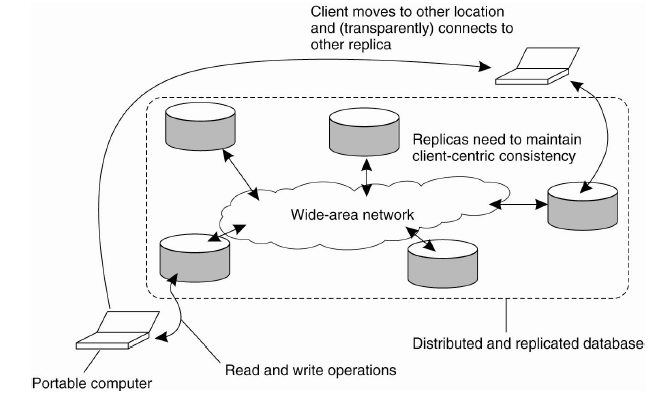
Ý tưởng: mô hình thống nhất hướng dữ liệu cung cấp một khung nhìn chung cho kho dữ liệu trên cả hệ thống. Vì vậy khi có thao tác cập nhật, thao tác này cần được phổ biến cho các bản sao khác nhau, dẫn tới chi phí tốn kém mà chưa chắc các tiến trình cần sử dụng các phần tử dữ liệu. Giải pháp đưa ra là chỉ cập nhật với các tiến trình có sử dụng phần tử dữ liệu. Đảm bảo các tiến trình có sử dụng dữ liệu có một khung nhìn thống nhất về dữ liệu là ý tưởng của mô hình thống nhất hướng người dùng.

Thống nhất cuối cùng (eventual consistency)

Thống nhất trong thời gian dài, tạm thời chưa thống nhất trong khoảng thời gian đó, khiến cho chi phí hệ thống rẻ, tuy nhiên chất lượng dịch vụ thấp, đặc biệt với các dịch vụ thời gian thực.

Ví dụ, ứng dụng database chủ yếu là đọc, rất ít khi cập nhật; web cache chỉ đọc mà không cập nhật. Xung đột ghi – ghi hầu như không xảy ra. Xung đột đọc – ghi, tiến trình ghi trong khi tiến trình khách đang đọc dẫn đến các kết quả khác nhau. Nếu dữ liệu không bị thay đổi trong thời gian đủ dài thì mô hình này sẽ đạt được thống nhất.

Giả sử, tiến trình khách luôn luôn truy cập vào một bản sao thì thống nhất luôn được đảm bảo. Nếu tiến trình khách truy cập vào nhiều bản sao khác nhau thì sẽ xuất hiện vấn đề.



Ví dụ, thiết bị di động thực hiện vài thao tác cập nhật trên một bản sao thứ nhất. Sau đó vì lí do nào đó thiết bị phải chuyển sang bản sao thứ hai, thì có thể có trường hợp các thao tác chưa được quảng bá tới bản sao thứ hai. Vì vậy cần đảm bảo thống nhất tiến trình khách. Kiểu thống nhất này không quan tâm tới các bản sao khác cũng như các tiến trình khách khác.

Các kiểu mô hình:

* Đọc đơn điệu
* Ghi đơn điệu
* Đọc kết quả đã ghi
* Ghi theo thao tác đọc

Kí hiệu

Các tiến trình truy cập vào một bản sao cục bộ thì chỉ có một tiến trình chủ được cập nhật dữ liệu (để tránh xung đột ghi – ghi)

Li: bản sao thứ i

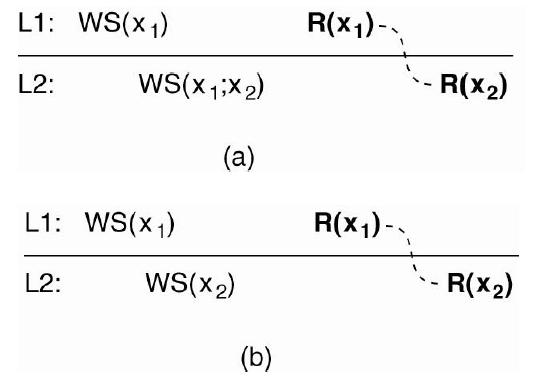
xi[t] : là phiên bản dữ liệu x ở bản sao cục bộ Li, thời gian t

WS(xi[t]): là kết quả của các thao tác ghi phần tử x tại bản sao Li trong thời gian t

WS(xi[t1], xj[t2]): tất cả các thao tác WS(xi[t1]) đã được phổ biến đến bản sao Lj, sau khoảng thời gian t2.

Thống nhất đơn điệu đọc (Monotonic Read Consistency)

Khi một tiến trình đọc giá trị phần tử dữ liệu x, tất cả các thao tác đọc tiếp theo của tiến trình đó sẽ đọc cùng một giá trị hoặc giá trị mới hơn của x. Nếu tiến trình đã đọc giá trị x, thì không bao giờ đọc giá trị cũ hơn x. Ví dụ, nếu đã nhận email ở vị trí a, thì khi sang vị trí b email vẫn nhận được.



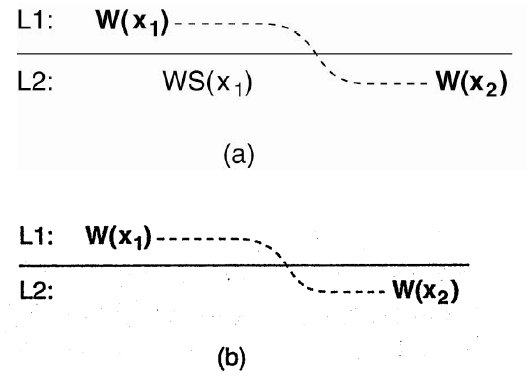
Trong hình a, thao tác R(x1) được thực hiện trước R(x2). Cú pháp WS(x1; x2) đảm bảo quá trình ghi x2 bao gồm quá trình ghi x1. Vì vậy, khi R(x2) được thực hiện thì kết quả của WS(x1) có trong thao tác.

Trong hình b, không đảm bảo khi đọc R(x2) có chứa x1

Thống nhất đơn điệu ghi (Monotonic Write Consistency)

Các thao tác ghi của một tiến trình trên dữ liệu là rời nhau, tương tự như hàng đợi queue (FIFO – First In First Out) nhưng chỉ có giá trị với một tiến trình. Các thao tác ghi của một tiến trình cần được kết thúc trước khi tiến trình đó thực hiện bất cứ một thao tác ghi nào trên cùng một phần tử dữ liệu.

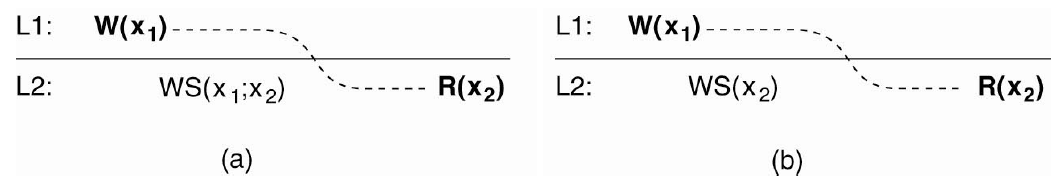
Ví dụ về đơn điệu ghi, việc cập nhật phần mềm thì phải đảm bảo luôn luôn cập nhật phiên bản mới nhất.



Trong hình a, sau khi thực hiện ghi W(x1) ở L1 thì tiến trình ghi W(x2) ở L2. Ta thấy việc ghi W(x1) đã được cập nhật trước đó nên mô hình này thỏa mãn đơn điệu ghi. Ở hình b là không thỏa mãn.

Thống nhất đọc dữ liệu ghi (Rear-your-write consistency)

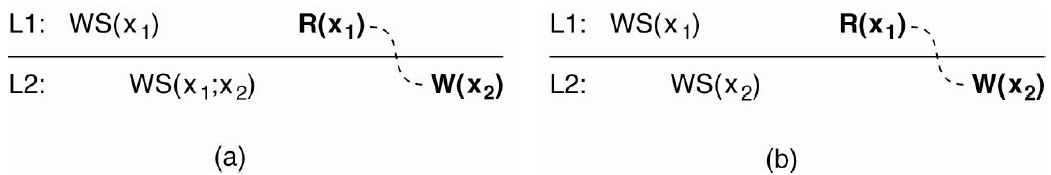
Trên một tiến trình nếu thao tác đọc xảy ra sau thao tác ghi thì thao tác đọc đó sẽ chờ và chỉ xảy ra sau khi thao tác ghi hoàn thành. Ví dụ như cập nhật nội dung trang web, thay đổi mật khẩu.



Ở hình a, tiến trình sau ghi giá trị W(x1) ở bản sao L1 rồi chuyển sang bản sao L2 để đọc giá trị R(x2). Ta thấy việc ghi x1 đã được cập nhật bằng thao tác WS(x1; x2) nên thỏa mãn mô hình đọc dữ liệu ghi. Ở hình b thì không thỏa mãn.

Thống nhất ghi sau khi đọc (Write-follow-reads consistency)

Thao tác ghi chỉ xảy ra sau thao tác đọc hoàn thành. Ví dụ, chỉ có thể trả lời sau khi đã đọc nội dung email.



Ở hình a, tiến trình đọc dữ liệu R(x1) ở bản sao L1 rồi sau đó mới ghi W(x2) ở bản sao L2. Ta thấy ở mô hình này việc ghi x1 đã được cập nhật đến L2 bằng thao tác WS(x1; x2) trước khi tiến trình đó ghi W(x2). Vì vậy mô hình này thỏa mãn thống nhất ghi sau khi đọc.

Ở hình b, việc ghi x1 chưa được cập nhật đến bản sao L2 trước khi tiến trình đó ghi W(x2), vì vậy mô hình này không thỏa mãn.

Explain how replication in DNS takes place, and why it actually works so well?

The basic idea is that name servers cache previously looked up results. These results can be kept in a cache for a long time, because DNS makes the assumption that name-to-address mappings do not change often.