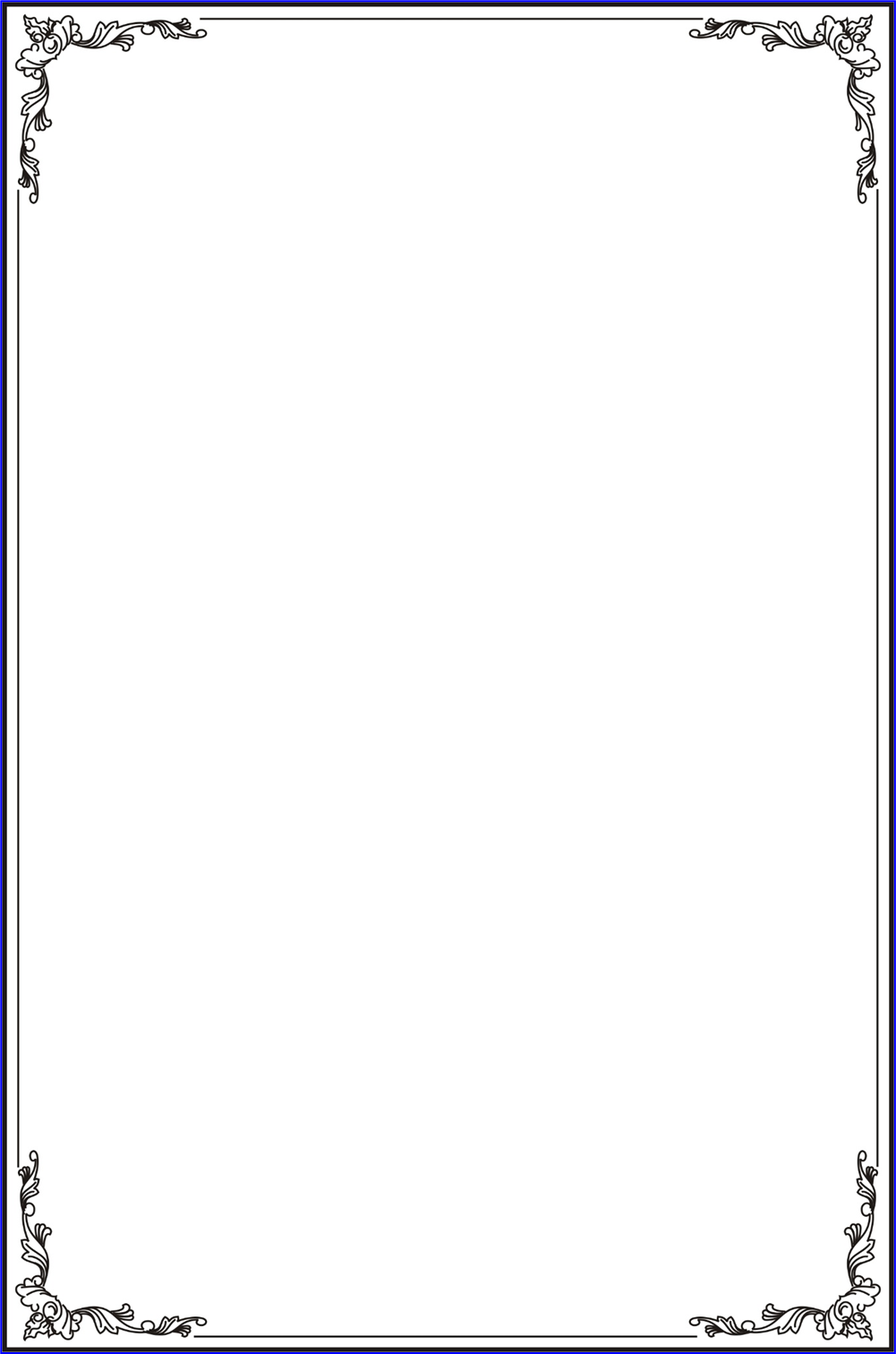
**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**🕯✡🕮🕮✡🕯**



**BÁO CÁO GIỮA KỲ**

**Môn học: BIGDATA SENSTIAL**

**Mã lớp học phần: BDES333877\_23\_2\_03CLC**

**GVHD:ThS. Lê Thị Minh Châu**

**Sinh viên thực hiện MSSV**

**Cao Thị Ngọc Phụng 21110276**

**Nguyễn Phú Thành 21110299**

**Trần Văn Tiến 21110319**

**Đinh Thị Thúy Quỳnh 21110284**

TP Hồ Chí Minh, tháng 03 năm 2024

|  |  |
| --- | --- |
| **ĐH SƯ PHẠM KỸ THUẬT TP.HCM**  **KHOA CÔNG NGHỆ THÔNG TIN**  **\*\*\*\*\*\*** | **CỘNG HÒA XHCN VIỆT NAM**  **Độc lập – Tự do – Hạnh phúc**  **\*\*\*\*\*\*** |
| **BẢNG PHÂN CÔNG CÔNG VIỆC** | |
| **Tên sinh viên** | **Công việc** |
| Đinh Thị Thúy Quỳnh | Trả lời câu a |
| Nguyễn Phú Thành | Trả lời câu b |
| Cao Thị Ngọc Phụng | Trả lời câu c |
| Trần Văn Tiến | Trả lời câu d |

***Read the original MapReduce paper by Dean and Ghemawat [1]. You can find the paper at*** [***http://research.google.com/archive/mapreduce.html***](http://research.google.com/archive/mapreduce.html)

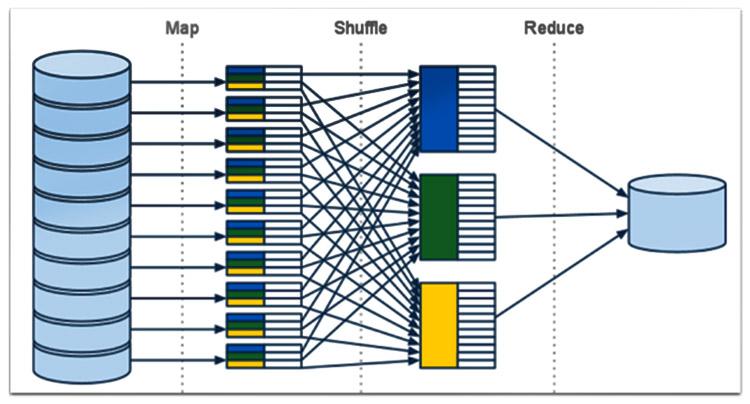
***a) How do the input keys/values, the intermediate keys/values and the output keys/values relate?***

MapReduce có 2 hàm chính là Map() và Reduce(), 2 hàm này được định nghĩa bởi người dùng và cũng là 2 giai đoạn liên tiếp trong quá trình xử lý dữ liệu của MapReduce.

Các cặp Input keys/values tạo ra bởi hàm Map và output chính là tập những cặp intermediate keys/values. Sau đó, tiến hành thông báo cho các hàm Reduce() để trực tiếp nhận dữ liệu.

Hàm Reduce sẽ tiếp nhận intermediate keys và values tương ứng với lượng keys đó. Các cặp keys/values này thường sẽ thông qua một con trỏ vị trí để đưa vào các hàm Reduce. Các cặp ouput keys/values được tạo ra bởi hàm Reduce() và là kết quả cuối cùng của MapReduce

Ở giữa Map và Reduce thì còn 1 bước trung gian đó chính là **Shuffle**. Sau khi Map hoàn thành  xong công việc của mình thì Shuffle sẽ làm nhiệm vụ chính là thu thập cũng như tổng hợp từ intermediate keys/values đã được map sinh ra trước đó rồi chuyển qua cho Reduce tiếp tục xử lý.



*Hình a.1: [1]*

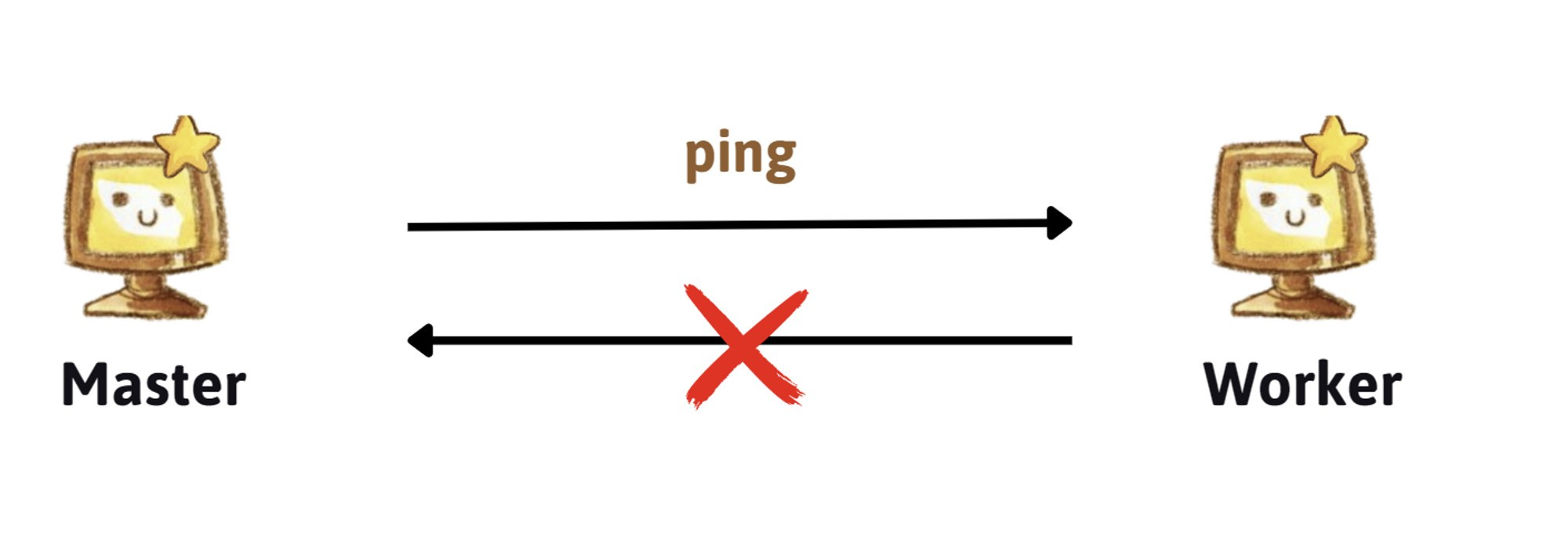
***b) How does MapReduce deal with node failures?***

Chương trình Map Reduce là một mô hình lập trình được thiết kế để xử lý song song lượng dữ liệu lớn trên hàng trăm hoặc hàng ngàn máy. Trong môi trường phân tán, việc ***node failures*** là điều hoàn toàn có thể xảy ra. MapReduce có một số cơ chế để xử lý lỗi nút và đảm bảo tính toàn vẹn của dữ liệu

***TH1: Worker failure***

*Bước 1: Xác định lỗi*

Theo định kỳ, Master sẽ ping tới tất cả các worker node. Nếu như không nhận được bất kì phản hồi nào từ worker node trong một thời gian nhất định, Master sẽ đánh dấu worker node đó bị lỗi (***node fail***)



*Hình b.1*

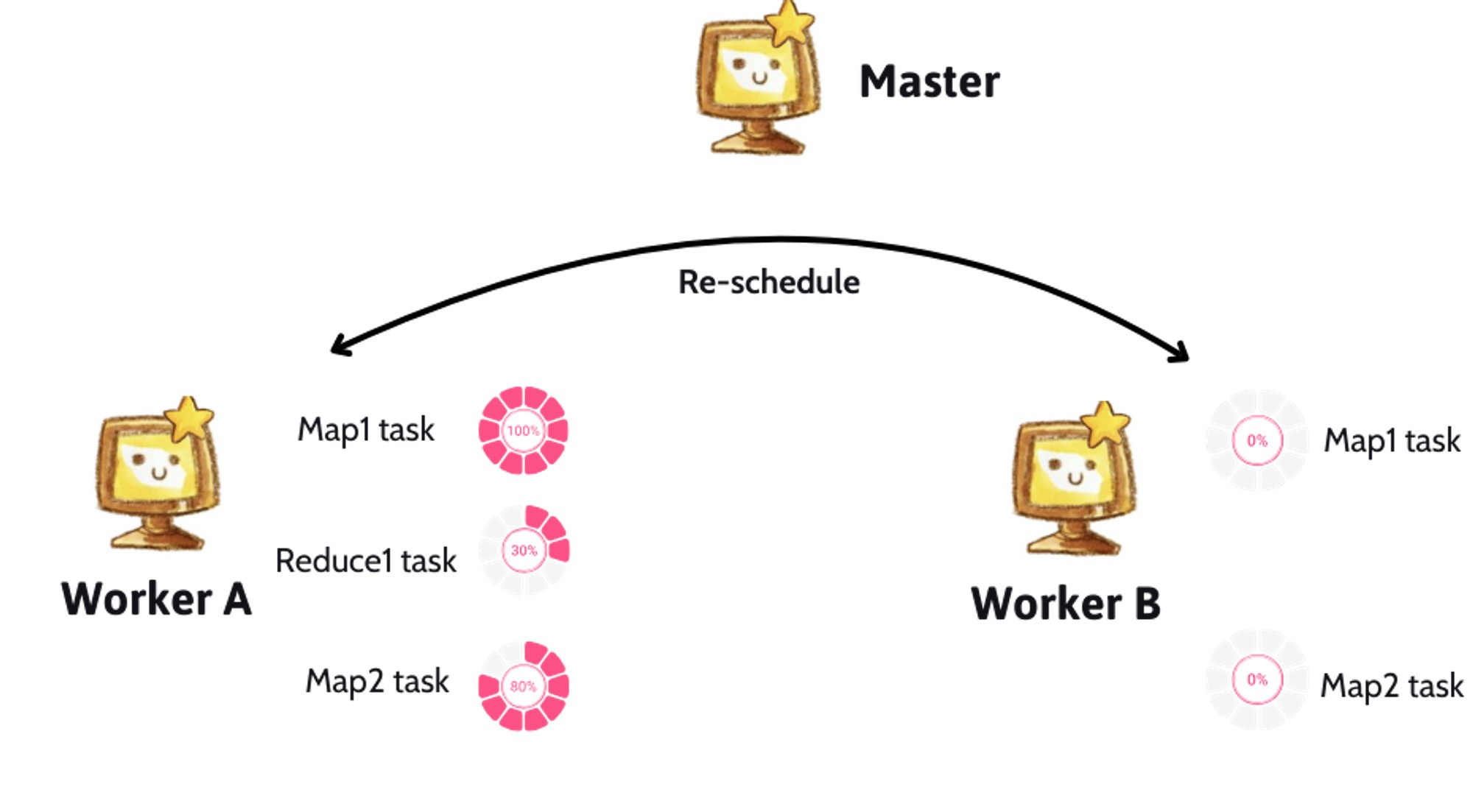
*Bước 2: Xử lý lỗi*

Khi master phát hiện worker node fails,

Master sẽ xác định các tác vụ đã được hoàn thành hoặc đang chạy trên node fail

Các map task đã được hoàn thành và các map task, reduce task đang được chạy trên node fail sẽ được đăt lại về trạng thái ban đầu

Chuyển các task đó cho một worker node đang khỏe mạnh đảm nhiệm

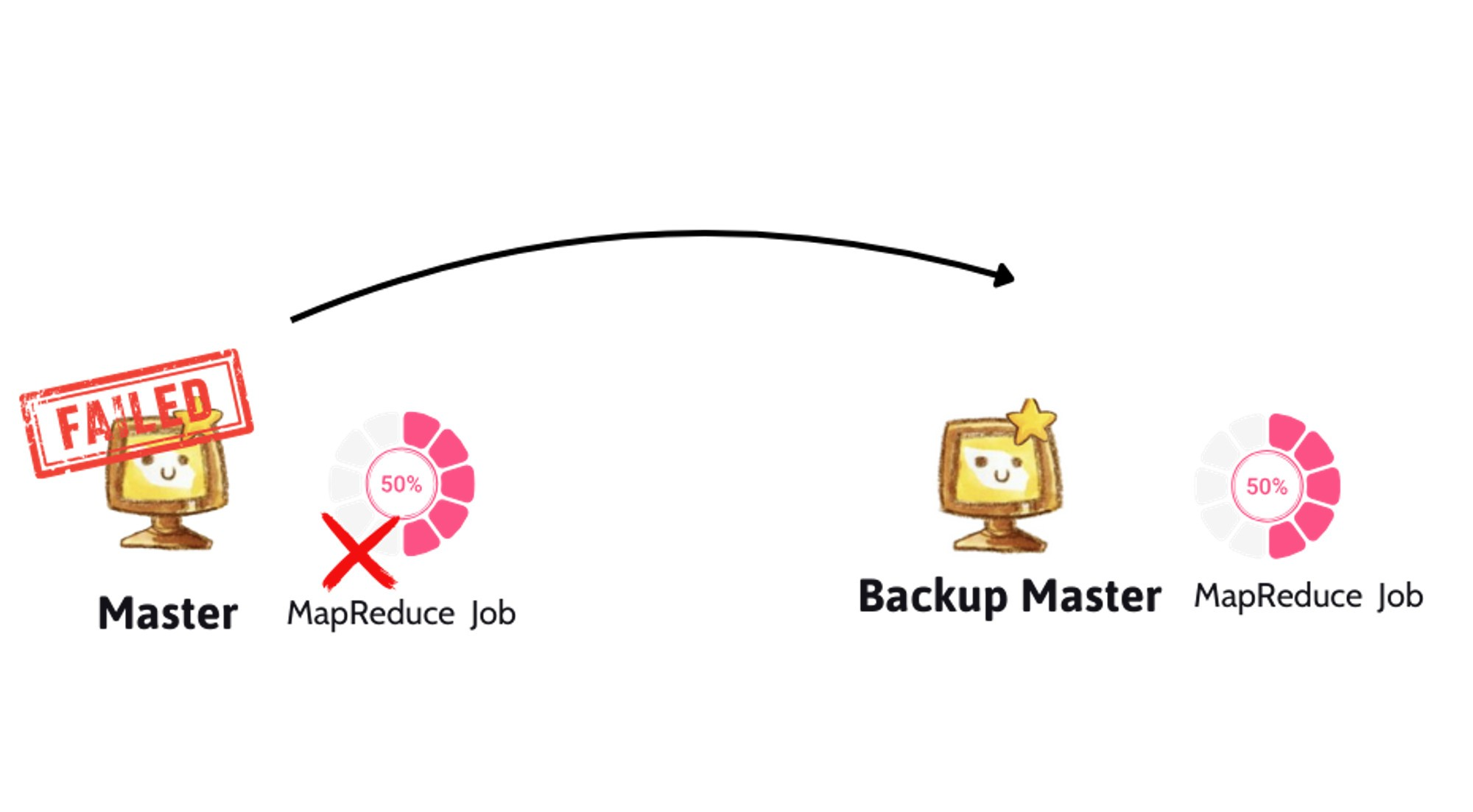


*Hình b.2*

***Lưu ý***: Các ***map task*** đã được hoàn thành trên node fail được ***thực hiện lại*** bởi node khác bởi vì output của map task đó được lưu trữ trên local disk của node fail nên không thể truy cập được. Các ***reduce task*** đã hoàn thành thì ***không cần thực hiện lại*** vì output của nó được lưu trữ ở global file system

***TH2: Master failure***

Trong Hadoop, chúng ta có thể cấu hình một hoặc nhiều Backup Master. Master chính sẽ cập nhật trạng thái, checkpoints một cách định kỳ. Khi Master chính bị lỗi (Master fail) thì Backup Master tiến hành Restart MapReduce Job từ trạng thái của Master chính trong lần kiểm tra cuối cùng

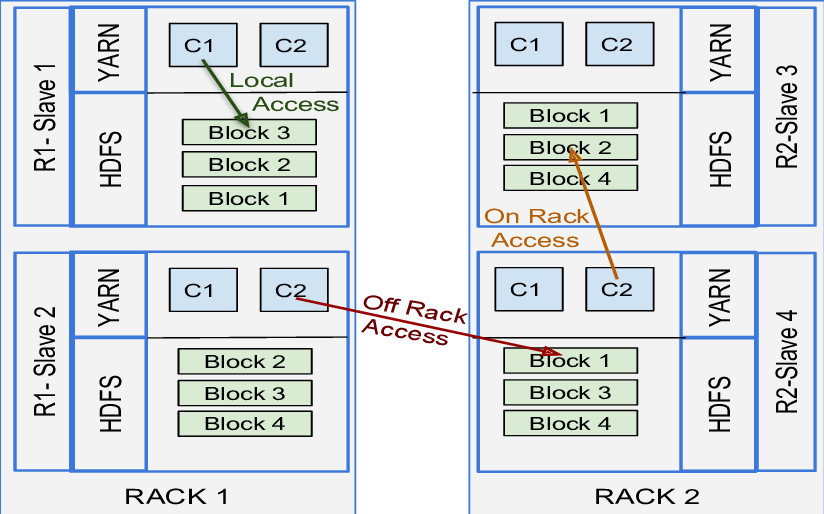
******

*Hình b.3*

***c) What is the meaning and the implication of locality? How is it used?***

"Locality" đề cập đến việc dữ liệu được xử lý ở nơi nó được lưu trữ. Việc này để giảm thiểu sự truyền dữ liệu qua mạng và tận dụng tài nguyên mạng hiệu quả hơn.

Dữ liệu đầu vào được lưu trữ trên những local disk của các máy mà chúng tạo nên cluster. Sau đó chia mỗi tệp thành những block 64 MB và lưu trữ một số bản sao của mỗi block (thường là 3 bản) trên các máy khác nhau. MapReduce master xem xét các input files và lên lịch map task trên máy có chứa bản sao của dữ liệu đầu vào tương ứng hoặc gần bản sao dữ liệu đầu vào của tác vụ đó.



*Hình c.1 [2]*

***d) Which problem is addressed by introducing a combiner function to the MapReduce model?***

*Vấn đề về tối ưu hiệu suất của việc truyền dữ liệu qua mạng*. Trong quá trình thực thi MapReduce, sau khi các nhiệm vụ map đã xử lý dữ liệu đầu vào và tạo ra các cặp key/value trung gian, dữ liệu này sẽ được chuyển đi đến các nhiệm vụ Reduce để thực hiện các phép tính cuối cùng. Trong trường hợp có nhiều bản ghi trung gian có cùng key (khóa). Điều này xảy ra khi xử lý dữ liệu có phân bố Zipf, nơi một số ít phần tử xuất hiện rất nhiều lần trong dữ liệu đầu vào (Ví dụ với input được truyền vào hàm Map là “toi yeu truong cua toi toi yeu co cua toi toi yeu ban cua toi toi yeu toi” thì sau khi qua hàm Map, sẽ có nhiều bản ghi có dang <toi, 1>). Do đó, việc truyền dữ liệu qua mạng có thể trở nên tốn kém và làm giảm hiệu suất của hệ thống do lưu lượng mạng cao và tải lớn cho các nhiệm vụ reduce.

*Hàm Combiner được giới thiệu để giải quyết vấn đề ở trên*. Hàm Combiner là một hàm tùy chọn được thực thi trên dữ liệu trung gian của mỗi node map trước khi dữ liệu được gửi đi đến các node reduce. Hàm có khả năng tổng hợp hoặc kết hợp các giá trị trung gian có cùng key thành các giá trị mới, giảm lượng dữ liệu cần phải chuyển qua mạng đến các nhiệm vụ Reduce. Khi dữ liệu được kết hợp bởi hàm Combiner trên các máy thực hiện Map, kích thước của dữ liệu cần truyền đi mạng giảm đi đáng kể, làm tăng tốc độ truyền dữ liệu và giảm áp lực trên hệ thống.

**TÀI LIỆU THAM KHẢO**

[1] https://github.com/ThanTrongHuynhNhan/Mining-Massive-Data-Sets-Apache-Spark-and-Mapreduce (truy cập vào 17:20 26/03/2024)

[2] [Data locality types in Hadoop jobs | Download Scientific Diagram (researchgate.net)](https://www.researchgate.net/figure/Data-locality-types-in-Hadoop-jobs_fig2_339363495) (truy cập vào 8:20 25/03/2024)