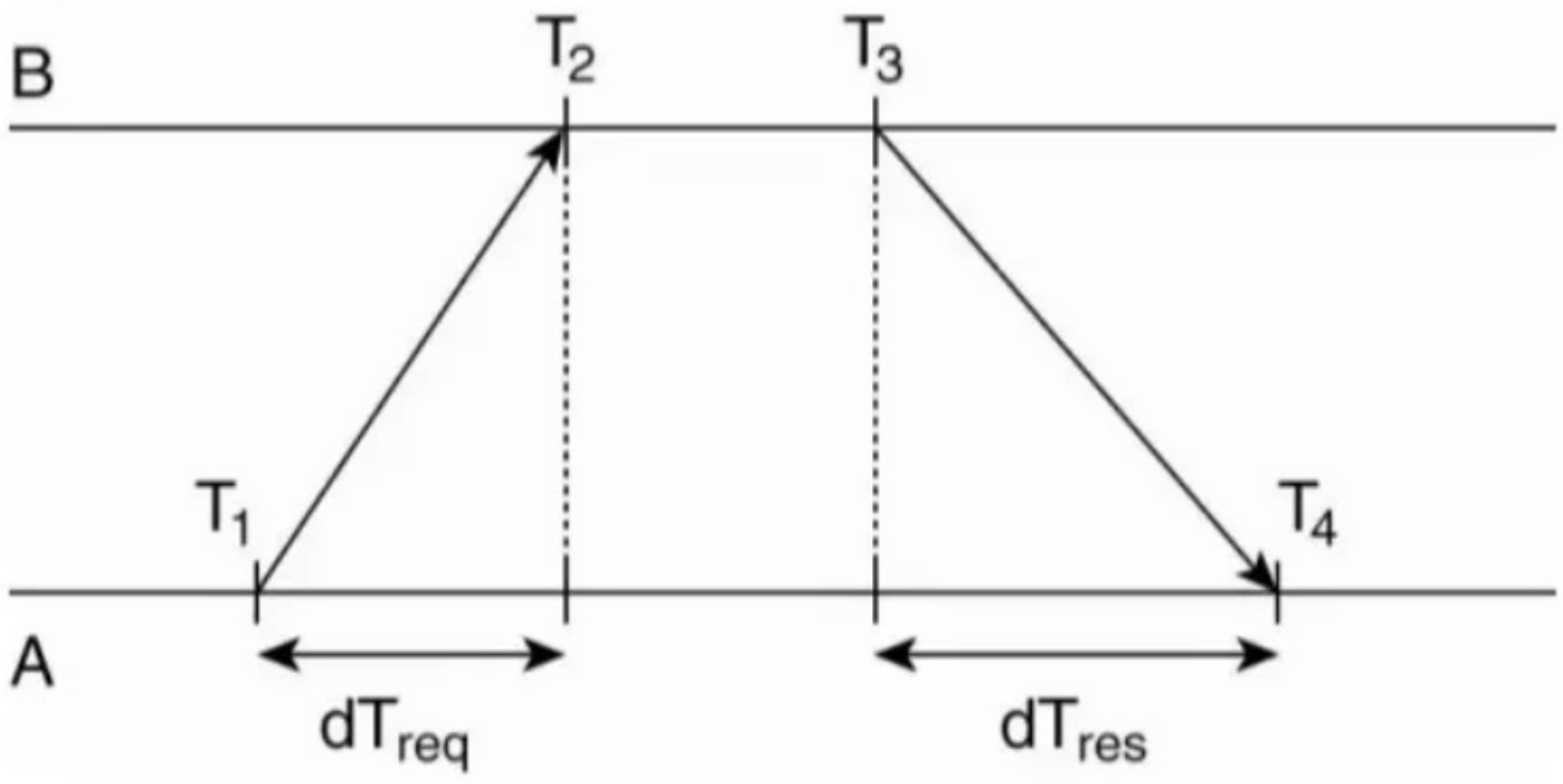
NGÂN HÀNG

CÁC HỆ THỐNG PHÂN TÁN PTIT

1. Giải thuật Cristian

Giả sử trong hệ phân tán có một máy chủ thời gian gọi là Time server và chúng ta sẽ tiến hành đồng bộ các máy khác với máy này như hình sau:



$$\theta = T_3 + \frac{(T_2 - T_1) + (T_4 - T_3)}{2} - T_4 = \frac{(T_2 - T_1) + (T_3 - T_4)}{2}$$

Getting the current time from a time server

- Client A gửi yêu cầu đến Time Server và thời gian đánh dấu đồng hồ cục bộ của A là T1.
- Server B nhận được yêu cầu đồng bộ của A tại thời điểm T2, và ghi lại thời điểm nhận là T2 theo đồng hồ của B.
- Đến thời điểm T3 theo đồng hồ của B, nó gửi cho A nhãn thời gian là T3 để A đặt lại thời gian là T3 và sẽ giữ kèm theo cả T1 và T2.
- Nhưng do độ trễ của mạng khi trả lời nên tại thời điểm T4 tính theo đồng hồ của A nó mới nhận được phản hồi của B.

- Bây giờ A có thông tin cả T1, T2, T3 và T4.

Trong hình vẽ mọi người sẽ thấy có 1 công thức. Đó là công thức tính **độ lệnh thời gian θ**. Giá trị này có thể âm hoặc dương

- Nếu θ > 0: thời gian Client tăng lên θ giây.
- Nếu θ < 0: thời gian Client chậm hơn θ giây.

2. Nhãn thời gian Lamport

Xét định nghĩa mối quan hệ "xảy ra trước" (→)

Khi có a→ b :

- a xảy ra trước b thì tất cả các tiến trình trong hệ phân tán thỏa thuận sự kiện a xảy ra trước rồi đến sự kiện b.
- Trong đó a và b là hai sự kiện của cùng một tiến trình.
- Nếu a xảy ra trước b thì a→b là đúng.
- Nếu a là sự kiện bản tin được gửi bởi một tiến trình nào đó và b là sự kiện bản tin đó được nhận bởi một tiến trình khác thì quan hệ a→ b là đúng.
- Quan hệ xảy ra trước có tính bắc cầu: a→ b, b→c thì a→c.

Nhãn thời gian

Cập nhật bộ đếm Ci cho tiến trình Pi

- (1) Trước khi thực thi sự kiên Pi thực thi Ci ←Ci+1
- (2) Khi tiến trình Pi gửi 1 thông điệp m cho Pj, nó sẽ set timestamp ts(m) băng Ci sau khi đã thực hiện ở bước trước
- (3) Khi nhận được thông điệp m, tiến trình Pj điều chỉnh bộ đếm cục bộ của chính nó là Cj ← max {Cj, ts(m)}. Sau đó nó sẽ thực hiện bước (1) và gửi thông điệp đến ứng dụng.

3. Đồng hồ vector

- Giải thuật đưa ra một vetor clocks VC(a) gán cho sự kiện a có thuộc tính là
    - nếu VC(a) < VC(b) thì sự kiện a là nguyên nhân của b.
- Trong vector clock mỗi tiến trình Pi lưu giữ một VCi với giá trị N (các tiến trình khác nhau thì N khác nhau)

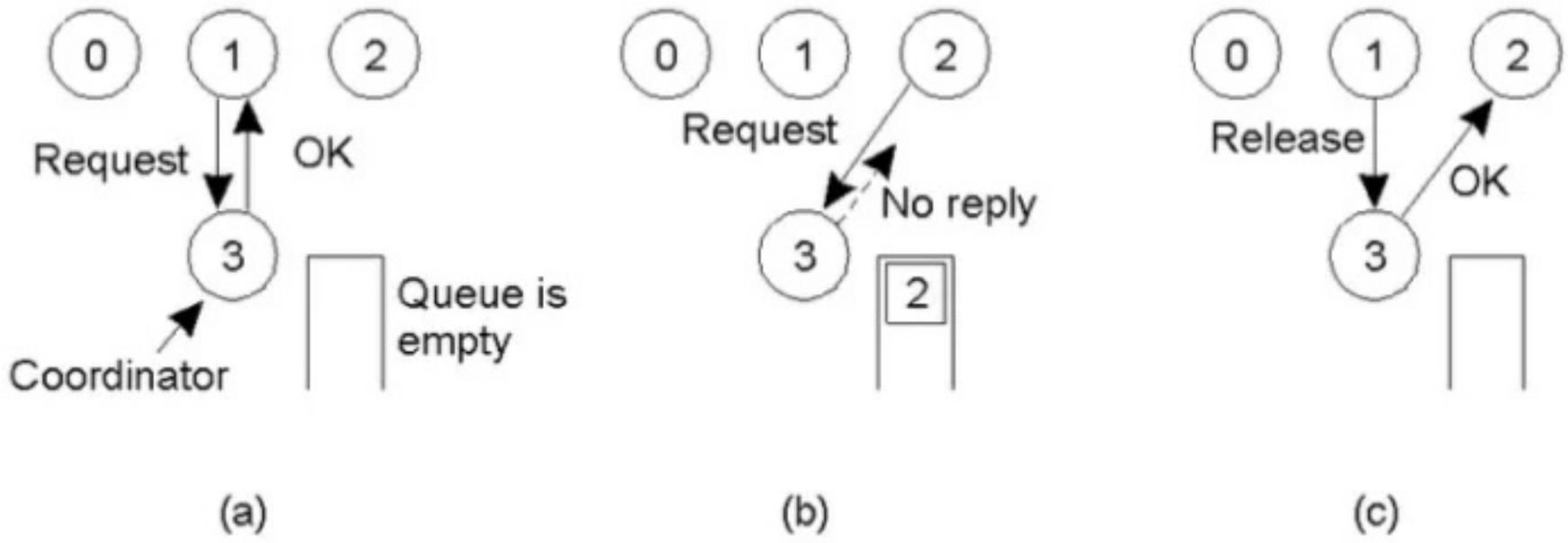
- VCi[i] là số các sự kiện đã xảy ra tại Pi
    - Nếu VCi[j] = k nghĩa là Pi biết đã có k sự kiện đã xẩy ra tại Pj
- Yêu cầu là mỗi khi có sự kiện mới xảy ra ở tiến trình Pi thì phải tăng VCi[i] và phải đảm bảo vector này được gửi cùng thông điệp suốt trong quá trình.
- Từ đó bên nhận sẽ biết được đã có bao nhiêu sự kiện xảy ra tại Pi .
- Quan trọng hơn phía nhận sẽ báo cho biết là đã có bao nhiêu sự kiện ở các tiến trình khác đã xảy ra trước khi Pi gửi thông điệp m.

Cách cập nhật vector:

- Thiết lập VCi[j] =0 với mọi j,i
- Sự kiện xảy ra ở Pi là nguyên nhân tăng VCi[i]
- Pi gắn một timestamp ts(m)=VCi vào mọi thông điệp gửi đi
- Khi Pi nhân được một thông điệp có ts(m) nó sẽ thiết lập VCi[j]=Max{Vi[j] ,ts(m)[j]} và tăng VCi[i]

4. GIẢI THUẬT TẬP TRUNG

Giả sử mỗi tiến trình có một số ID duy nhất. Tiến trình được bầu chọn làm điều phối (Coordinator) là tiến trình có số ID cao nhất.



- Khi một tiến trình nào đó cần vào vùng giới hạn nó sẽ gửi một thông điệp xin cấp quyền .
- Nếu không có một tiến trình nào đang trong vùng giới hạn thì tiến trình điều phối sẽ gửi phản hồi cho phép.
- Còn nếu có một tiến trình khác đang ở trong vùn tới hạn rồi thì tiến trình điều phối sẽ gửi thông điệp từ chối và đưa tiến trình này vào hàng đợi cho đến khi không có tiến trình nào trong vùng tới hạn nữa.
- Khi một tiến trình rời khỏi vùng giới hạn nó sẽ gửi một thông điệp đến tiến trình điều phối thông báo trả lại quyền truy cập.

- Lúc này tiến trình điều phối sẽ gửi quyền truy cập cho tiến trình đầu tiên trong hàng đợi truy cập.

Ưu điểm:

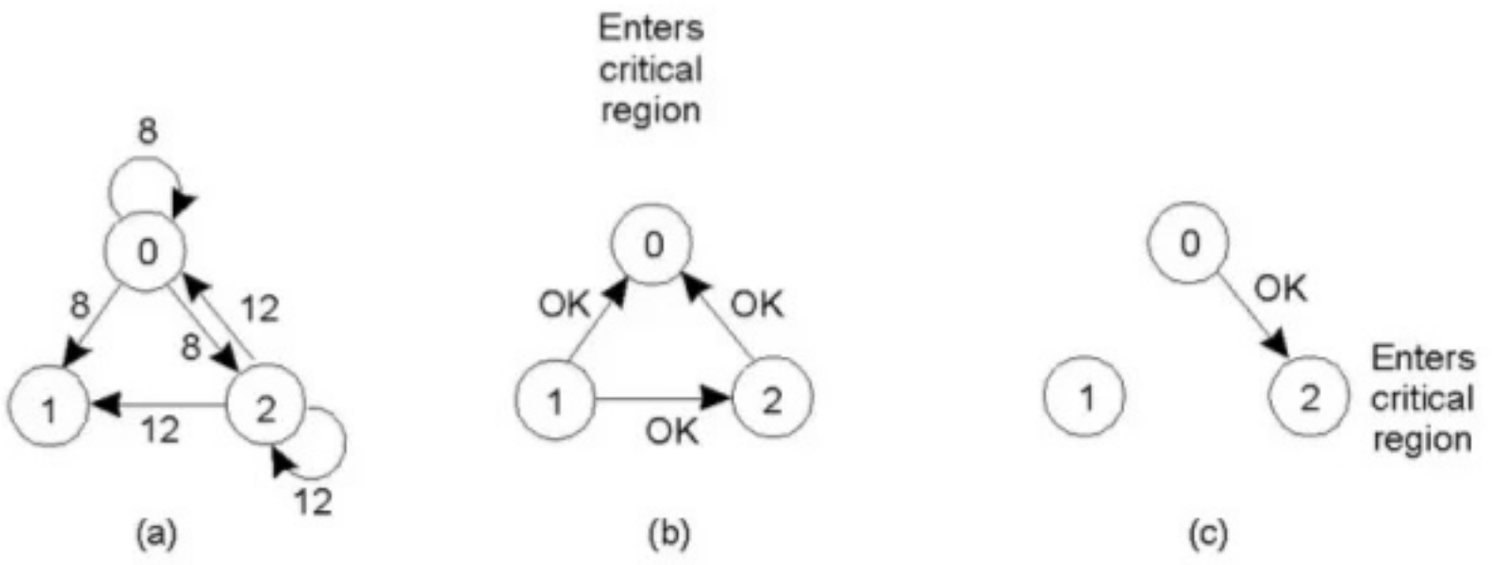
- Giải thuật đảm bảo sự loại trừ lẫn nhau: bộ điều phối chỉ để một tiến trình truy cập tài nguyên trong 1 thời điểm.
- Không có tiến trình nào phải đượi vô thời hạn.
- Sự sắp xếp này dễ thực thi và chỉ yêu cầu 3 thông điệp cho mỗi lần sử dụng tài nguyên gồm: yêu cầu truy nhập tài nguyên, sự cho phép và giải phóng tài nguyên.

Nhược điểm:

- Nếu tiến trình điều phối bị hỏng thì hệ thống sẽ sụp đổ .Vì nếu một tiến trình đang trong trạng thái Block nó sẽ không thể biết được tiến trình điều phối có bị DEAD hay không .
- Trong một hệ thống lớn nếu chỉ có một tiến trình điều phối sẽ xuất hiện hiện tượng thắt cổ chai

5. GIẢI THUẬT PHÂN TÁN

- Khi một tiến trình muốn truy cập vào một tài nguyên chia sẻ, nó tạo một thông điệp bao gồm tên của tài nguyên, số xử lý của nó và thời gian (theo logic) hiện tại.
- Sau đó nó gửi thông điệp này tới các tiến trình khác và chính nó.
- Việc gửi các thông điệp đi là đáng tin cậy.
- Các tiến trình khác sau khi nhận được thông điệp này sẽ xảy ra ba tình huống:



- (a) : Nếu bên nhận không ở trong tài nguyên chia sẻ và cũng không muốn vào vùng tài nguyên chia sẻ thì nó sẽ gửi thông điệp OK cho bên gửi.
- (b): Nếu bên nhận đang ở trong tài nguyên chia sẻ thay vì trả lời nó sẽ cho vào hàng đợi yêu cầu này.
- (c): Nếu bên nhận cũng muốn truy cập tài nguyên chia sẻ nhưng chưa được phép, nó sẽ so sánh nhãn thời gian (timestamp) của thông điệp gửi đến với timestamp

chứa trong thông điệp mà nó gửi đi cho những tiến trình khác. Nếu thông điệp đến có timestamp thấp hơn, bên nhận sẽ gửi thông điệp OK, nếu không thì nó không gửi gì cả.

Sau khi gửi các gói tin yêu cầu cho phép, một tiến trình đợi đến khi các tiến trình khác cho phép. Ngay sau khi các tiến trình cho phép, tiến trình này được truy cập tài nguyên. Khi nó kết thúc, nó gửi 1 thông điệp OK đến tất cả các tiến trình khác ở trong hàng đợi của nó và xóa nội dung hàng đợi đó.

Ví dụ: Ở hình vẽ trên:

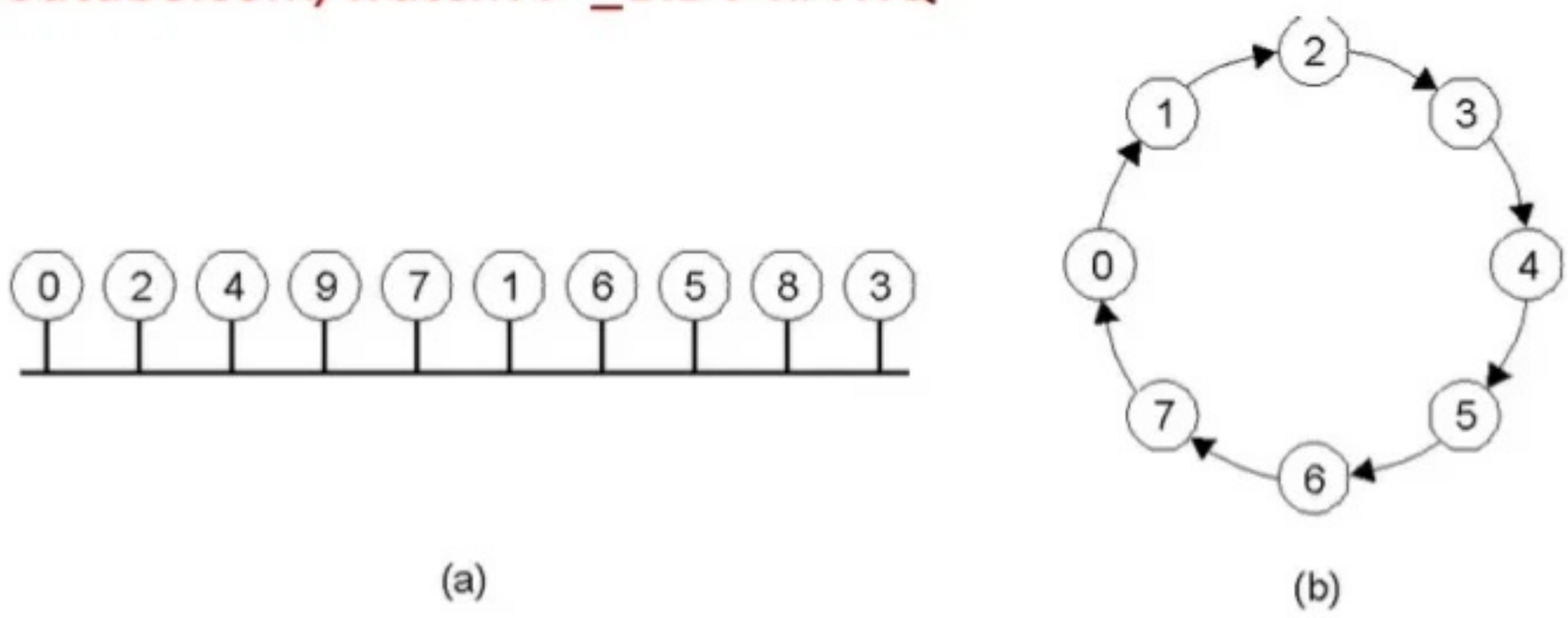
- Tiến trình 0 gửi 1 yêu cầu với timestamp = 8 đến tất cả các tiến trình.
- Trong cùng thời điểm đó, tiến trình 2 cũng làm tương tự với timestamp = 12.
- Tiến trình 1 không muốn truy cập tài nguyên, vì thế nó gửi OK đến cả 2 bên gửi.
- Cả tiến trình 0 và 2 nhận ra sự xung đột và cùng so sánh timestamp.
- Tiến trình 2 thua do có timestamp lớn hơn và nó phải gửi thông điệp OK đi.
- Tiến trình 0 truy cập vào tài nguyên và xếp tiến trình 2 vào hàng đợi của nó để xử lý và truy cập tài nguyên.
- Sau khi kết thúc, nó loại bỏ yêu cầu từ tiến trình 2 khỏi queue của nó và gửi thông điệp OK đến tiến trình 2 và cho phép nó thực hiện.

Nhược điểm:

- Khi tổng số lượng tiến trình trong hệ thống là n thì yêu cầu 2(n-1) thông điệp cho mỗi thực thể .
- Nếu bất cứ 1 tiến trình nào lỗi, nó sẽ gây lỗi thông điệp phản hồi yêu cầu và khiến toàn bộ các tiến trình tiến vào vùng giới hạn
- Thuật toán này chậm, phức tạp và chi phí đắt và ít mạnh mẽ như thuật toán tập trung

6. GIẢI THUẬT MẠCH VÒNG

Giả sử tất cả các tiến trình được sắp xếp trên một vòng tròn logic, các tiến trình đều được đánh số và đều biết đến các tiến trình cạnh nó.



- Bắt đầu quá trình truyền, tiến trình 0 sẽ được trao một thẻ bài.
- Thẻ bài này có thể lưu hành xung quanh vòng tròn logic.
- Nó được chuyển từ tiến trình k đến tiến trình (k+1) bằng cách truyền thông điệp điểm – điểm.
- Khi một tiến trình giành được thẻ bài từ tiến trình bên cạnh, nó sẽ kiểm tra xem có thể vào vùng tới hạn hay không. Nếu không có tiến trình khác trong vùng tới hạn nó sẽ vào vùng tới hạn.
- Sau khi hoàn thành phần việc của mình nó sẽ nhả thẻ bài ra và thẻ bài có thể di chuyển tự do trong vòng tròn.
- Nếu 1 tiến trình muốn vào vùng tới hạn thì nó sẽ giữ lấy thẻ bài, nếu không nó sẽ để cho thẻ bài truyền qua.

Nhược điểm:

- Vấn đề lớn nhất trong thuật toán này là thẻ bài có thẻ bị mất. Khi đó chúng ta phải sinh lại thẻ bài vì việc dò tìm lại thẻ bài là rất khó.

7. GIẢI THUẬT BẦU CHỌN

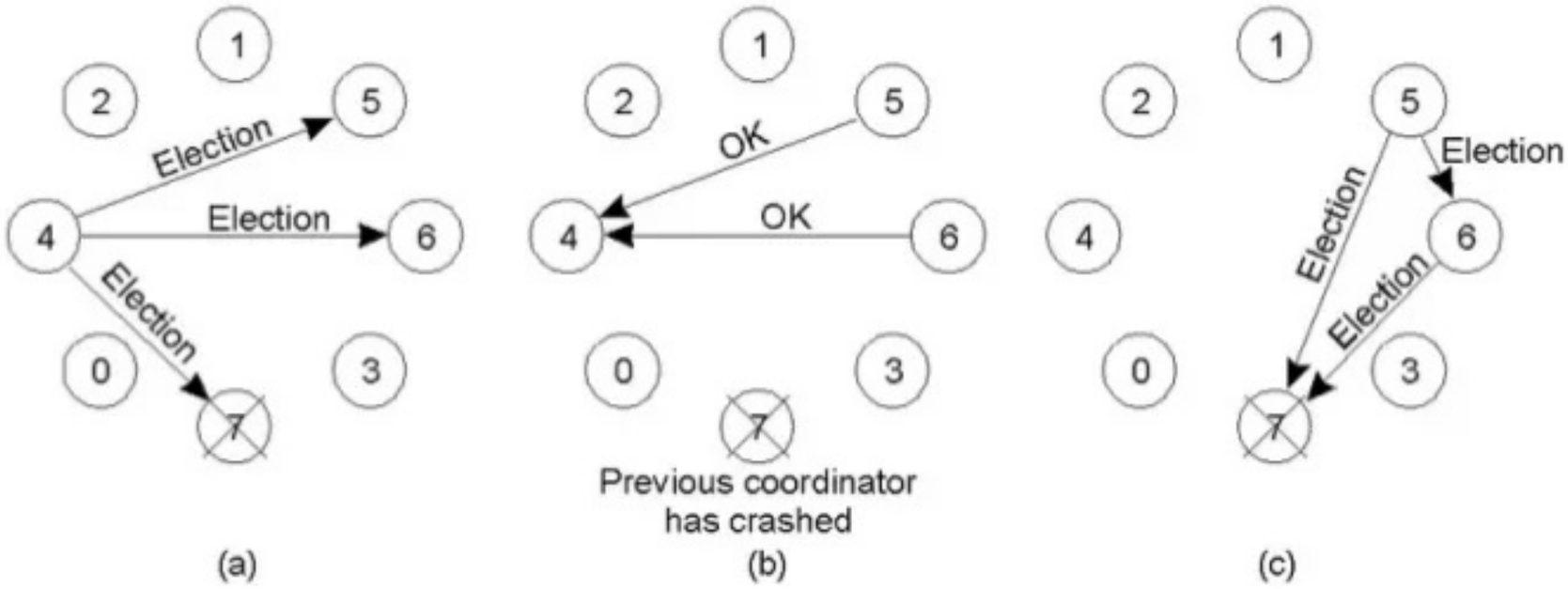
- Khi tiến trình điều phối gặp lỗi thì sẽ phải có quá trình bầu chọn để chọn ra một tiến trình khác làm điều phối thay cho nó.
- Có hai giải thuật bầu chọn hay được sử dụng là:
  - Giải thuật áp đảo (Bully Algorithm)
  - Giải thuật vòng (Ring Algorithm)

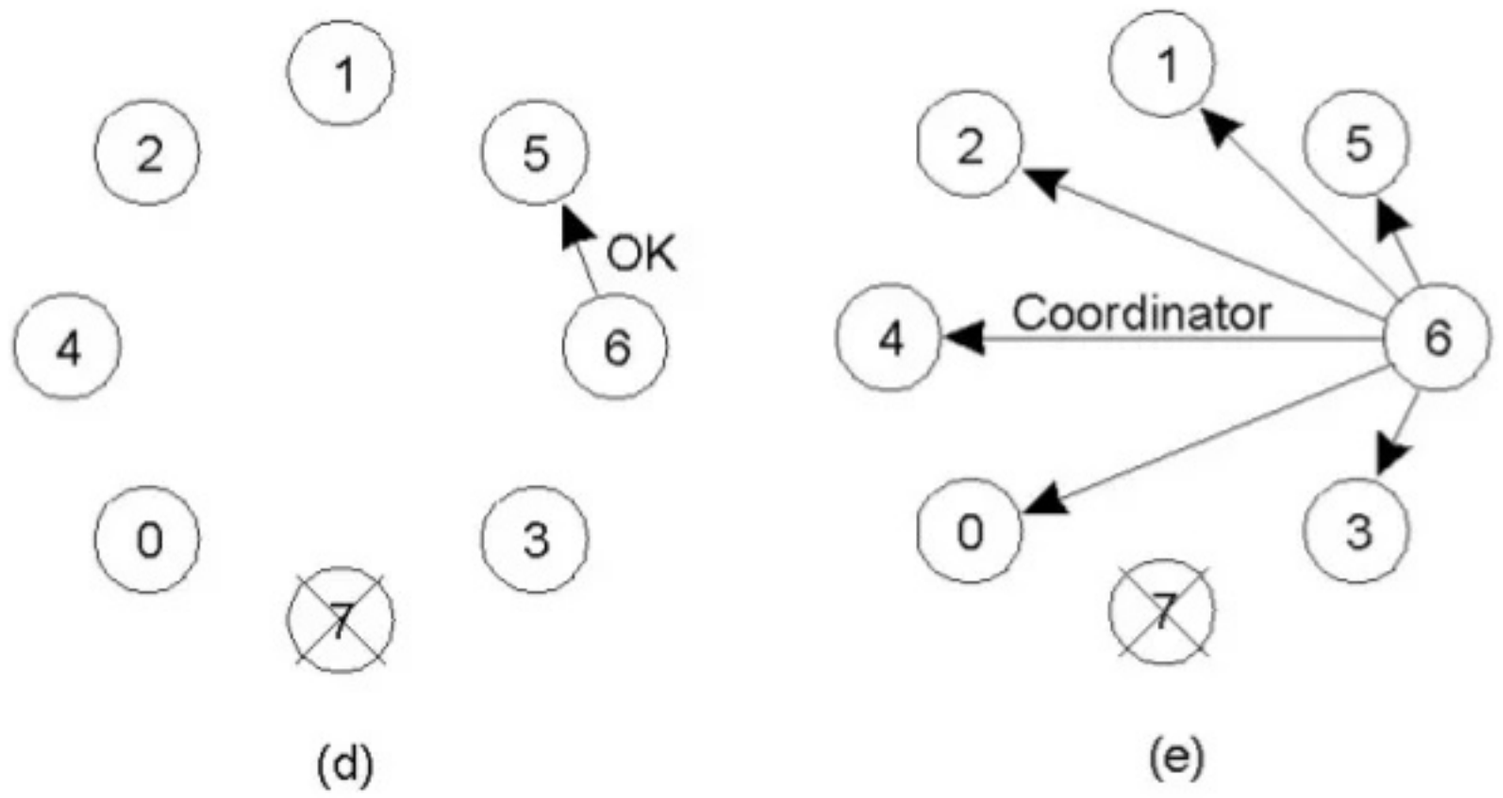
4.1. BẦU CHỌN ÉP BUỘC

Giả sử:

- Mỗi một tiến trình đều có một ID duy nhất.

- Tất cả các tiến trình khác đều có thể biết được số ID và địa chỉ của mỗi tiến trình trong hệ thống.
- Chọn một tiến trình có ID cao nhất làm khóa.
- Tiến trình sẽ khởi động việc bầu chọn nếu như nó khôi phục lại sau quá trình xảy ra lỗi hoặc tiến trình điều phối bị trục trặc.





Các bước của giải thuật:

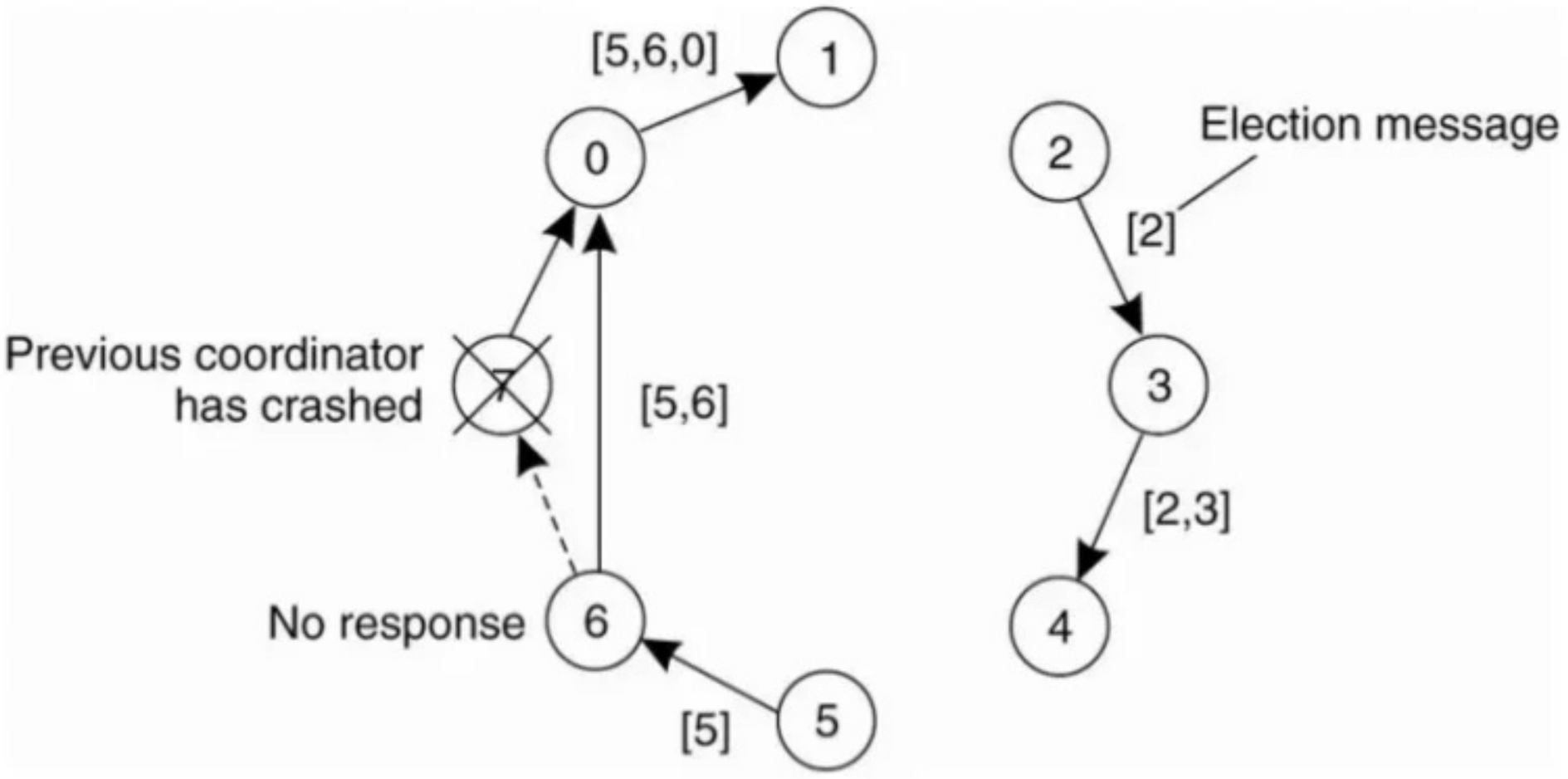
- P gửi thông điệp Election đến tất cả các tiến trình có ID cao hơn
- Nếu không có tiến trình nào phản hồi thì P sẽ trở thành tiến trình điều phối
- Nếu có một tiến trình có ID cao hơn phản hồi thì nó sẽ đảm nhiệm vai trò điều phối.

8 GIẢI THUẬT

Giả sử:

- Các tiến trình có một ID duy nhất và được sắp xếp trên 1 vòng tròn Logic.

- Mỗi một tiến trình có thể nhận biết được tiến trình bên cạnh mình.



Các bước của giải thuật:

- Một tiến trình bắt đầu gửi thông điệp Election tới các nút còn tồn tại gần nhất, quá trình gửi theo 1 hướng nhất định.
- Nó sẽ thăm dò liên tiếp trên vòng cho đến khi tìm được 1 nút còn tồn tại.
- Mỗi một tiến trình sẽ gắn ID của mình vào thông điệp gửi.
- Cuối cùng sẽ chọn ra 1 tiến trình có ID cao nhất trong số các tiến trình còn hoạt động và gửi thông điệp điều phối cho tiến trình đó.

Allahu Akbar - Marhaban Salahhudin -

DO NOT COPY - GOD WILLS IT

youtube.com/watch?v=_BlDV-xrWlQ

Câu 1 (2 điểm) : Điền giá trị của các bước thực hiện giải thuật đồng thuận phân tán cho các tiến trình sau:

	Tiến trình 1	Tiến trình 2	Tiến trình 3	Tiến trình 4
Giá trị	180	281	382	483
Trạng thái	Không lỗi	Lỗi	Không lỗi	Không lỗi

Giả thiết tiến tình lỗi nhận đượng thông điệp từ các tiến trình không lỗi nhưng không gửi được thông điệp cho các tiến trình khác. Thứ tự giá trị được lưu trữ theo số thứ tự tiến trình tăng dần.

	Tiến trình 1	Tiến trình 2	Tiến trình 3	Tiến trình 4
Kết quả lần thứ nhất	180,Nil,382,483 0 điểm	180,281,382,483 0 điểm	180,Nil,382,483 0 điểm	180,Nil,382,483 0 điểm
Kết quả lần thứ hai	Nil,Nil,Nil,Nil 0 điểm 180,Nil,382,483 0 điểm 180,Nil,382,483 0 điểm	180,Nil,382,483 0 điểm 180,Nil,382,483 0 điểm 180,Nil,382,483 0 điểm	180,Nil,382,483 0 điểm Nil,Nil,Nil,Nil 0 điểm 180,Nil,382,483 0 điểm	180,Nil,382,483 0 điểm Nil,Nil,Nil,Nil 0 điểm 180,Nil,382,483 0 điểm
Kết quả cuối cùng	180,Unknow,382,483 0 điểm	180,Unknow,382,483 0 điểm	180,Unknow,382,483 0 điểm	180,Unknow,382,483 0 điểm

Điểm số: 0.0

Lời giải:
Cái nào lỗi thì full số.
Còn các tiến trình còn lại thì sẽ bị nil ở tiến trình bị lỗi gửi

- Câu hỏi 1: Thế lần 123 là thế nào?
Kết quả lần 2. Mỗi cột là 3 tiến trình khác gửi đến

Viết theo hàng dọc
Cột thứ 1 là 2,3,4 gửi đến, 2 lỗi nên full nil, 3 và 4 là chung kết quả
Cột thứ 2 thì 1,3,4 gửi

- Câu hỏi 2: kết quả là như nào?
Cột 3 1,2,4 gửi cứ như vậy thôi: 180 nil 382 483
Dòng cuối thì so cột: Cái nào quá nil thì để unknow

- Câu hỏi 3: So cả lần 1-2 sao? Hay chỉ lấy 3 cái lần 2 ra so thôi?

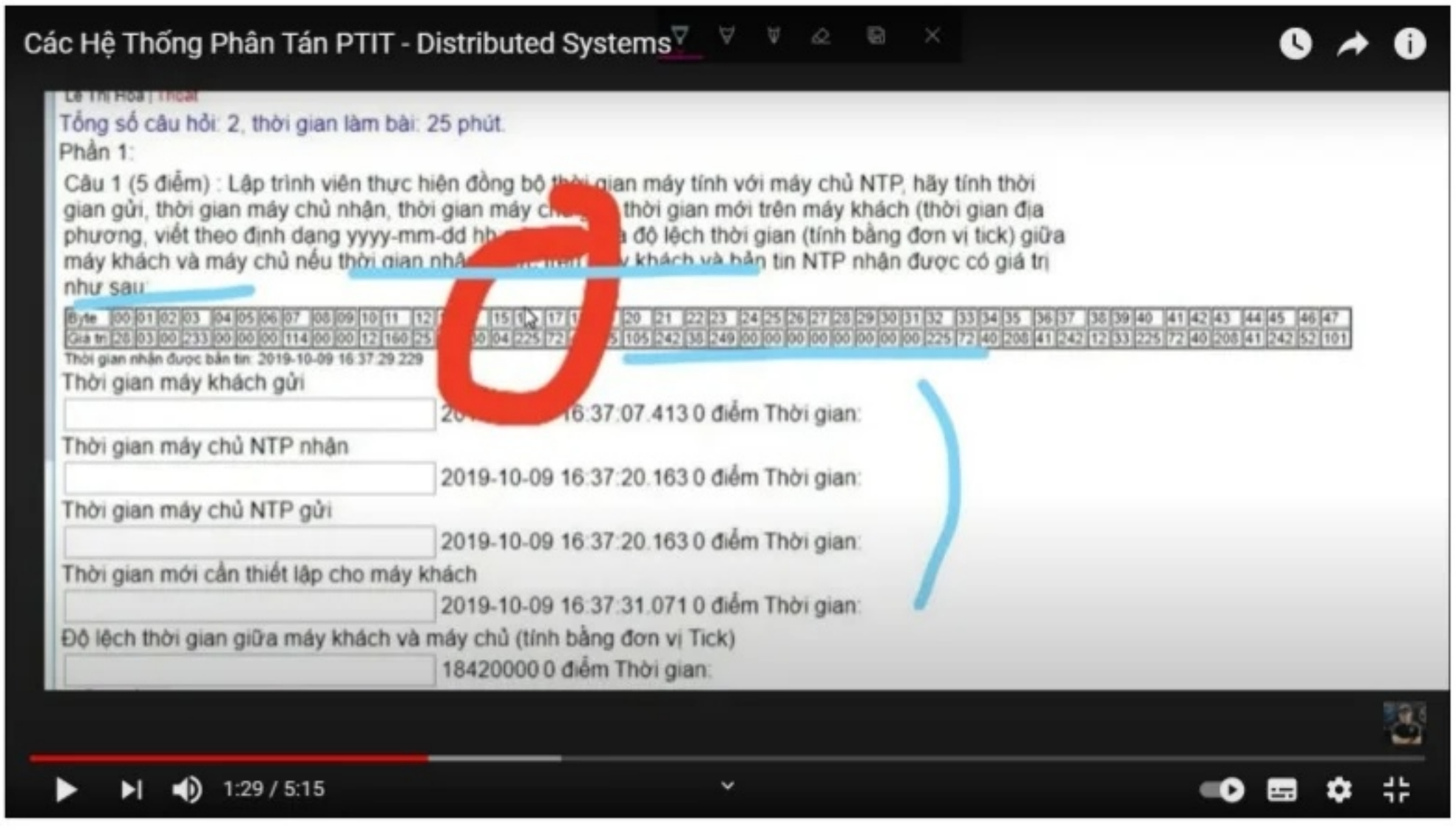
Vâng so tất cả theo trí nhớ của tôi. Ai ghi đủ kiểm tra lại cho chắc.
Kiểu còn có điều kiện của bài này là 1/3 hoặc nhiều hơn tiến trình hỏng thì may ra làm được

Nhưng yên tâm thường thầy sẽ cho 1 tiến trình hỏng thôi, ít nhất 3 tiến trình chạy được thì

Allahu Akbar - Marhaban Salahhudin -

DO NOT COPY - GOD WILLS IT

youtube.com/watch?v=_BlDV-xrWlQ



2. Làm tròn mặc định

- Math.Round(num1);
- num1: Giá trị cần làm tròn
- Làm trọn mặc định, các số từ 0,1,2,3,4,5 sẽ làm tròn về 0 và số 6,7,8,9 làm tròn về 1

VD:

- Math.Round(10.4); //kết quả: 10
- Math.Round(10.5); //kết quả: 10
- Math.Round(10.8); //kết quả: 11

Allahu Akbar - Marhaban Salahhudin -

DO NOT COPY - GOD WILLS IT

youtube.com/watch?v=_BlDV-xrWlQ

Allahu Akbar - Marhaban Salahhudin -

DO NOT COPY - GOD WILLS IT

youtube.com/watch?v=_BlDV-xrWlQ

```
using System;

namespace TestProgram
{
    class Program
    {
        static void Main(string[] args)
        {

            byte[] data =
            {
                28,03,00,233,00,00,00,114,
                00,00,12,160,25,66,230,04,
                225,72,40,195,105,242,38,249,
                00,00,00,00,00,00,00,00,
                225,72,40,208,41,242,12,33,
                225,72,40,208,41,242,52,101
            };
            DateTime T1 = Convert(data, 16);
            DateTime T2 = Convert(data, 32);
            DateTime T3 = Convert(data, 40);
            DateTime T4 = new DateTime(2019, 10, 9, 16, 37, 29,229);
            long Theta = (long)Math.Round(((T2.Ticks - T1.Ticks) + (T3.Ticks - T4.Ticks)) / 2.0);
            T4 = T4.AddTicks(Theta);
            Console.WriteLine(Theta);
            Console.WriteLine(T1.ToString("yyyy-MM-dd HH:mm:ss.fff"));
            Console.WriteLine(T2.ToString("yyyy-MM-dd HH:mm:ss.fff"));
            Console.WriteLine(T3.ToString("yyyy-MM-dd HH:mm:ss.fff"));
            Console.WriteLine(T4.ToString("yyyy-MM-dd HH:mm:ss.fff"));
```

DO NOT COPY - GOD WILLS IT