

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÁO CÁO TIỂU LUẬN MÔN
NGUYÊN LÝ
VÀ MÔ THỨC PHÁT TRIỂN HỆ PHÂN TÁN

ĐỀ TÀI: THUẬT TOÁN BẦU CHỌN TRONG HỆ PHÂN TÁN

Giáo viên hướng dẫn : TS. Trần Hải Anh

Học viên thực hiện : Ngô Minh Quân CB190232

Bùi Hoàng Nam CB190240

Lớp : 19BATTT

Hà Nội, 7 – 2020

MỤC LỤC

CHƯƠNG 1: NGHIÊN CỨU THUẬT TOÁN BẦU CHỌN TRONG HỆ PHÂN TÁN....	3
I. Các thuật toán bầu chọn truyền thống.....	3
1.1. Thuật toán bầu chọn là gì	3
1.2. Các giả thuyết đặt ra cho hệ thống phân tán sử dụng thuật toán bầu chọn	3
1.3. Thuật toán Bully	5
1.3.1. Điều kiện áp dụng.....	5
1.3.2. Giải thuật	5
1.3.3. Thuật toán Bully sửa đổi	6
1.3.4. Ví dụ minh họa	7
1.4. Thuật toán Ring	9
1.4.1. Điều kiện áp dụng:.....	9
1.4.2. Giải thuật	10
1.4.3. Ví dụ minh họa	10
1.5. Ưu nhược điểm của thuật toán bầu chọn.....	14
1.5.1. Ưu điểm:	14
1.5.2. Nhược điểm:	14
II. Thuật toán bầu chọn trong môi trường không dây	15
CHƯƠNG II. TRIỂN KHAI VÀ SO SÁNH HIỆU NĂNG CỦA CÁC THUẬT TOÁN BẦU CHỌN	17
2.1. Yêu cầu đề bài	17
2.2. Thực hiện.....	17
2.2.1. Thuật toán Bully	18
2.2.2. Thuật toán Ring	20
2.2.3. So sánh hiệu năng của 2 thuật toán	21
TÀI LIỆU THAM KHẢO	22

CHƯƠNG 1: NGHIÊN CỨU THUẬT TOÁN BẦU CHỌN TRONG HỆ PHÂN TÁN

I. Các thuật toán bầu chọn truyền thống

1.1. Thuật toán bầu chọn là gì

Các thuật toán bầu cử là việc chọn một quy trình từ nhóm các bộ xử lý để hoạt động như một người điều phối. Nếu quá trình điều phối bị treo vì một số lý do, thì điều phối viên mới sẽ được bầu chọn cho bộ xử lý khác. Thuật toán này về cơ bản xác định vị trí nên khởi động lại bản sao của điều phối viên mới

Thuật toán bầu chọn giả định rằng mọi quy trình đang hoạt động trong hệ thống đều có một số ưu tiên duy nhất. Tiến trình có mức độ ưu tiên cao nhất sẽ được chọn làm điều phối viên mới. Do đó, khi một bộ điều phối bị lỗi, thuật toán này sẽ lựa chọn tiến trình có số ưu tiên cao nhất trong các tiến trình còn lại, sau đó thông tin của tiến trình này sẽ được gửi lại cho các tiến trình khác trong hệ thống được biết.

1.2. Các giả thuyết đặt ra cho hệ thống phân tán sử dụng thuật toán bầu chọn

Loại lỗi có thể xảy ra trong hệ phân tán đóng một vai trò quan trọng trong giao thức bầu cử. Do đó, bước đầu tiên để tìm hiểu về giao thức bầu cử là hiểu loại lỗi có thể xảy ra trong hệ thống.

Cố gắng bảo vệ khỏi tất cả các hư hỏng có thể xảy ra trong hệ thống là không khả thi, vì vậy chúng ta bắt đầu bằng việc không xem xét một số hư hỏng mà tỷ lệ bảo vệ được rất thấp. Cụ thể, có 3 giả định được đặt ra như sau:

Giả định 1: Tất cả các node hoạt động và sử dụng chung thuật toán bầu chọn

Giả định 2: Thuật toán bầu chọn tại mỗi nút phải sử dụng các phần mềm cơ bản nhất định. Các phần mềm cơ bản này bao gồm hệ điều hành và trình xử lý tin nhắn. Chúng tôi giả định không có lỗi trong các phần mềm cơ bản đó.

Giả định 3: Nếu một node i nhận được thông điệp M từ node j , trong khi đó thông điệp M đã được từ node j đến node i ở một khoảng thời gian khác sớm hơn. Nghĩa là, chúng tôi giả định hệ thống con sẽ không giao tiếp 1 cách tự động

Có một số loại hư hỏng khác mà xác suất xảy ra có thể hạ xuống mức có thể

chấp nhận được bằng một số kỹ thuật tránh và sửa lỗi cơ bản. Để đơn giản hóa, chúng tôi giả định rằng những kỹ thuật này đang được sử dụng và chúng ta sẽ không gặp lại những lỗi này nữa. Cụ thể, chúng tôi có các giả định sau:

Giả định 4: Tất cả các node đều có một vài ô lưu trữ an toàn. Dữ liệu được lưu trữ trong các ô an toàn này sẽ hỗ trợ mọi sự cố của nút. Một lỗi trong quá trình cập nhật của ô an toàn không thể làm hỏng trạng thái của nó. Hoặc cập nhật được thực hiện chính xác để lại các ô cáo giá trị mới hoặc cập nhật không bao giờ được thực hiện. Vector trạng thái được sử dụng bởi thuật toán bầu chọn và sau đó được lưu giữ trong kho an toàn

Giả định 5: Khi một node xảy ra lỗi, nó sẽ tạm dừng tất cả các tiến trình đang chạy. Sau đó một khoảng thời gian, bộ xử lý tại node đó được đặt lại về một số trạng thái cố định và quá trình thực thi tiếp tục từ trạng thái cố định đó. Giả định rằng một lỗi không thể khiến một node đi chệch ra khỏi các thuật toán của nó và hoạt động theo kiểu không thể đoán trước. Mọi lỗi phần cứng khiến node không dừng hoàn toàn phải được phát hiện và chuyển thành sự cố chính thức trước khi nút đo có thể ảnh hưởng đến bất kỳ thành phần hệ thống nào khác

Giả định 6: Không có lỗi trên đường truyền. Nghĩa là nếu một bản tin M được gửi đi thì tin nhắn nhận được chính xác là M

Giả định 7: Tất cả các bản tin từ node I đến node j đều được xử lý tại node j theo cùng thứ tự mà chúng được gửi đi. Điều này không có ý rằng nếu các bản tin được gửi từ node i theo thứ tự M1, M2, M3 thì node j phải xử lý theo thứ tự M1, M2, M3. Ví dụ node j có thể xử lý M1, M3 và khai báo M2 là không nhận được

Cuối cùng, chúng tôi đưa ra 2 giả định truyền thống. Hai giả định này sẽ thực hiện một cuộc bầu chọn trong hệ thống phân tán tương tự như quan niệm trực quan của chúng ta về một cuộc bầu cử. Những giả định này cũng đơn giản hóa việc thiết kế giao thức bầu chọn.

Giả định 8: Hệ thống con giao tiếp không bị lỗi. Cụ thể, chúng tôi giả định rằng hệ thống con giao tiếp có giới hạn thời gian cố định T, trong đó nó đảm bảo rằng

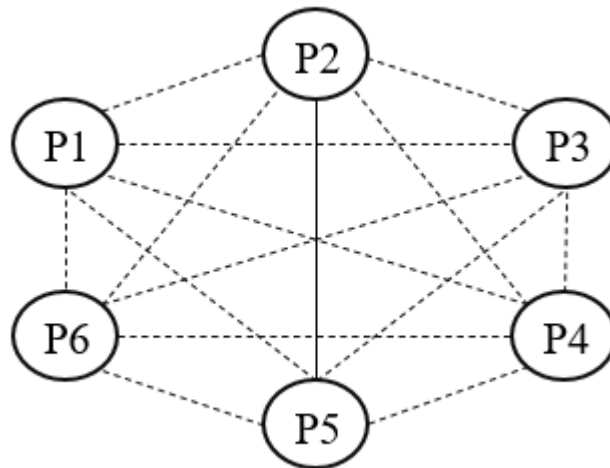
lượng dữ liệu sẽ được phân phối nếu nút đích hoạt động. Nghĩa là, nếu nút i muốn gửi một thông điệp M tới nút j , thì nút i chuyển thông điệp M cho hệ thống con giao tiếp. Nếu sau T giây mà nút i không nhận được xác nhận từ hệ thống con giao tiếp rằng M đã được nhận bởi j thì nút i biết chắc rằng nút j đã bị lỗi

Giả định 9: Tất cả các node không bao giờ dừng hoạt động và luôn phản hồi tin nhắn đến với độ trễ bằng 0 (giả định này tương tự giả định 8). Nói cách khác, không trả lời “nhẹ” cho các tin nhắn đến nó cũng coi như là một thất bại. Do đó, nếu node j trì hoãn việc nhận thông báo M và có bất kỳ khả năng nào xảy ra rằng một nút nào đó có thể đã đợi T giây mà không nghe tin về việc nhận của M thì nút j sẽ tự động chuyển trạng thái lỗi. Nghĩa là, tất cả các quá trình thực thi phải được tạm dừng, trạng thái của nút được đặt lại và khởi chạy tiến trình khôi phục

1.3. Thuật toán Bully

1.3.1. Điều kiện áp dụng

Thuật toán này áp dụng cho các hệ thống phân tán khi mà một tiến trình bất kỳ có thể gửi bản tin đến tất cả các tiến trình khác trong hệ thống. Ví dụ:



1.3.2. Giải thuật

- Giả sử tiến trình P gửi một thông điệp đến bộ điều phối
 - + Nếu tiến trình điều phối không phản hồi lại P trong một khoảng thời gian T nhất định, thì tiến trình điều phối hiện tại được coi là đã fail

+ Bây giờ, tiến trình P sẽ gửi thông báo bầu chọn leader đến tất cả các tiến trình có số ưu tiên cao hơn nó

+ Tiến trình P tiếp tục đợi trong một khoảng thời gian T, nếu không có tiến trình nào phản hồi thì tiến trình P tự chọn mình làm coordinator mới.

+ Sau đó, nó gửi một thông báo đến tất cả các quy trình có số ưu tiên nhỏ hơn nó rằng nó đã được bầu cử làm coordinator mới của chúng

+ Tuy nhiên, nếu P nhận được bất kỳ câu trả lời nào trong khoảng thời gian T, từ bất kỳ tiến trình nào khác Q:

- Tiến trình P lại đợi khoảng thời gian T' để nhận được một thông báo khác từ Q rằng nó đã được bầu chọn làm coordinator
- Nếu Q không phản hồi trong khoảng thời gian T' nghĩa là quá trình bầu chọn tại tiến trình Q không thành công và thuật toán được khởi động lại

1.3.3. Thuật toán Bully sửa đổi

Như đã nói ở phần trên, số lượng bản tin cần để vận chuyển giữa các tiến trình trong quá trình bầu chọn là khá lớn. Do đó, phương pháp này tốn khá nhiều băng thông mạng.

Để giải quyết nhược điểm này, ta sẽ trình bày phương pháp tối ưu hóa bằng cách sửa đổi thuật toán Bully, giúp giảm đáng kể số lượng thông báo cần được trao đổi. Hơn nữa, số lượng các giai đoạn chính được giảm từ 5 xuống 4. Chi tiết như sau:

- Khi tiến trình P thông báo rằng tiến trình điều phối bị lỗi, nó khởi tạo thuật toán bầu chọn

- Khi tiến trình P biết rằng tiến trình điều phối đang lỗi, nó sẽ gửi thông điệp bầu chọn cho tất cả các tiến trình có độ ưu tiên cao hơn

- Mỗi tiến trình nhận được thông điệp bầu chọn từ tiến trình P (các tiến trình này đều có độ ưu tiên cao hơn P) sẽ gửi thông điệp OK kèm theo số thứ tự ưu tiên cho P

- Nếu không tiến trình nào phản hồi cho P, nó sẽ gửi bản tin quảng bá về coordinator tới tất cả các tiến trình, khai báo nó là điều phối mới. Nếu có tiến trình

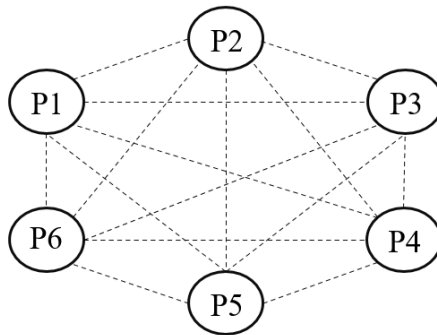
nào đó phản hồi lại P, nó sẽ so sánh các số thứ tự ưu tiên với nhau và chọn ra số lớn nhất làm coordinator và sau đó gửi bản tin GRANT

- Tại giai đoạn đó, coordinator mới cũng quảng bá tin nhắn tới tất cả các process khác và thông báo bản thân nó đã được chọn làm coordinator mới

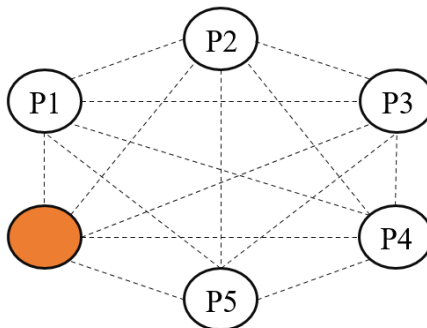
- Thuật toán mới không chỉ có tất cả các ưu điểm của thuật toán Bully mà còn không có nhược điểm mà Bully mắc phải (số lượng thông điệp truyền đi lớn). Hơn nữa, số lượng giai đoạn chính cũng giảm đi

1.3.4. Ví dụ minh họa

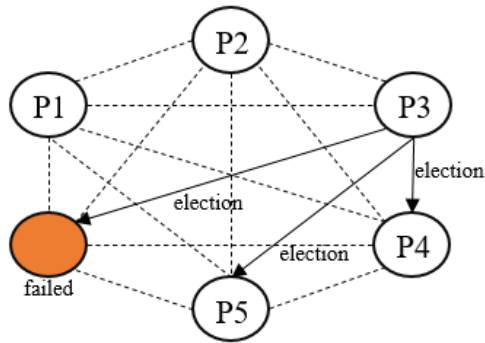
- Bước 0: Chúng ta bắt đầu với 6 tiến trình, tất cả chúng đều được kết nối với nhau. Tiến trình 6 đang là leader vì nó có số ưu tiên cao nhất.



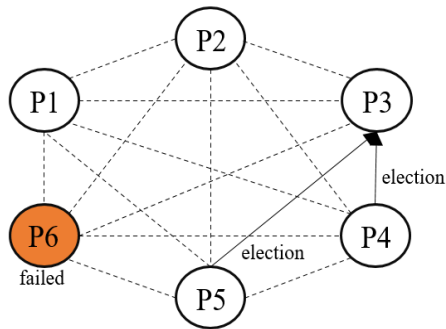
- Bước 1: Tiến trình 6 bị lỗi



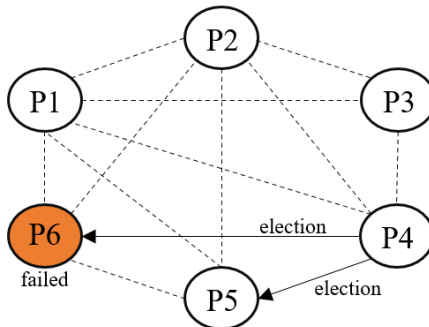
- Bước 2: Tiến trình 3 gửi thông báo cho các tiến trình 4, 5 về việc bắt đầu quá trình bầu chọn mới do tiến trình 6 không có phản hồi



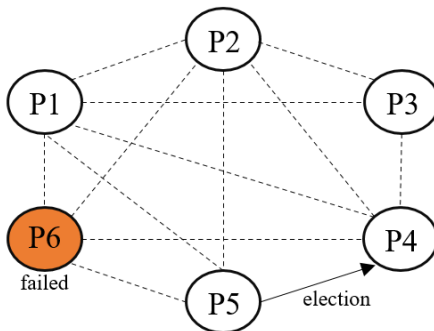
- Bước 3: Tiến trình 4 và 5 gửi phản hồi lại cho tiến trình 3, nói rằng chúng sẽ chịu trách nhiệm tiếp



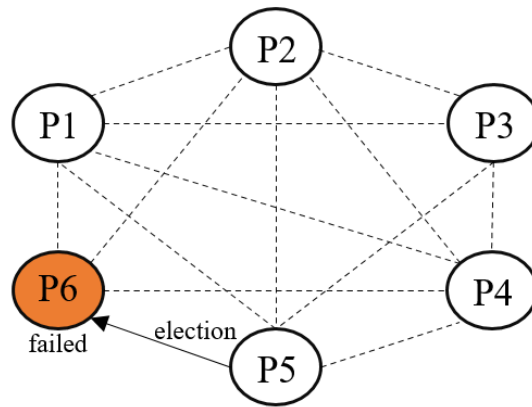
- Bước 4: Tiến trình 4 tiếp tục gửi bản tin bầu chọn cho cả tiến trình 5 và 6



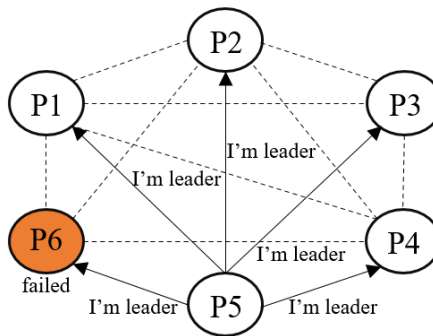
- Bước 5: Tiến trình 5 gửi trả lời tiến trình 4 rằng sẽ chịu trách nhiệm tiếp



- Bước 6: tiến trình 5 gửi bản tin bầu chọn tới tiến trình 6



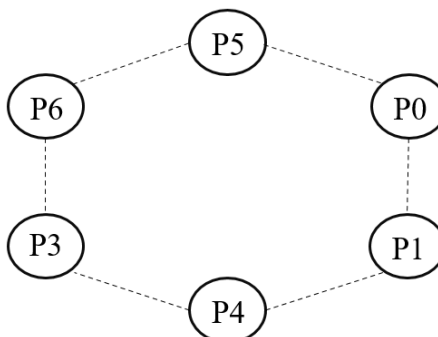
- Bước 7: tiến trình 6 không trả lời, tiến trình 5 tuyên bố nó chiến thắng trong quá trình bầu chọn



1.4. Thuật toán Ring

1.4.1. Điều kiện áp dụng:

Thuật toán này áp dụng cho các hệ thống được tổ chức theo hình vòng (về mặt vật lý hoặc logic). Trong thuật toán này, chúng ta giả định rằng liên kết giữa các tiến trình là đơn hướng và mọi tiến trình chỉ có thể gửi thông báo đến tiến trình ở bên phải của nó. Cấu trúc dữ liệu mà thuật toán này sử dụng là danh sách hoạt động (active list), là một danh sách có số thứ tự ưu tiên của tất cả các tiến trình đang hoạt động trong hệ thống.



1.4.2. Giải thuật

- Nếu tiến trình P1 phát hiện lỗi của bộ điều phối, nó sẽ tạo danh sách hoạt động mới, khởi đầu là danh sách rỗng. Nó gửi bản tin bầu chọn tiến trình hàng xóm ở bên phải nó kèm với active list và thêm số 1 vào danh sách hoạt động của nó.

- Nếu tiến trình P2 nhận được bản tin bầu chọn từ tiến trình bên trái của nó, nó sẽ trả lời theo một trong ba cách sau:

+ Nếu bản tin nhận được không chứa số 1 trong active list thì P1 thêm số 2 vào active list của nó và chuyển đi cho tiến trình tiếp theo

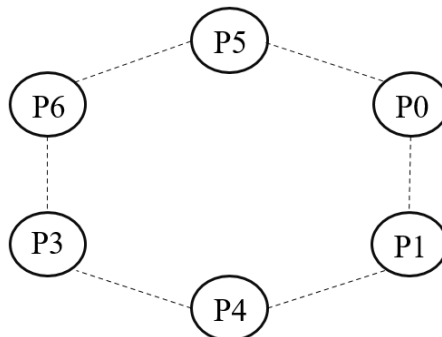
+ Nếu đây là bản tin bầu cử đầu tiên mà nó nhận được hoặc gửi đi, P1 sẽ tạo active list mới bao gồm số 1 và số 2. Sau đó nó sẽ gửi bản tin bầu cử 1 theo sau là 2

+ Nếu tiến trình P1 nhận được bản tin bầu cử 1 của chính nó thì active list cho P1 bây giờ chứa số của tất cả các quy trình đang hoạt động trong hệ thống. Bây giờ quy trình P1 phát hiện số ưu tiên cao nhất từ danh sách và chọn nó làm coordinator mới và gửi bản tin GRANT.

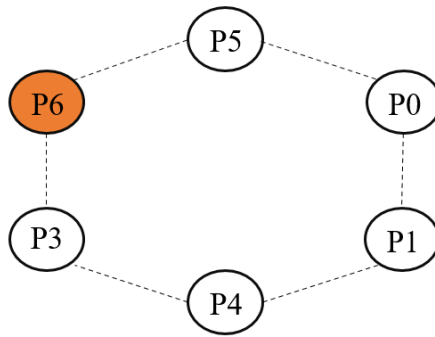
- Ở trạng thái này, tiến trình được chọn làm coordinator sẽ quảng bá bản tin cho tất cả các tiến trình khác rằng nó đã được chọn

1.4.3. Ví dụ minh họa

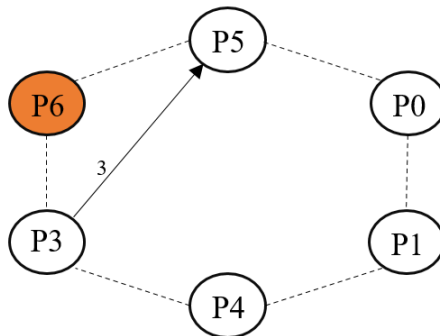
- Bước 0: Chúng ta bắt đầu với 6 tiến trình, kết nối logic theo hình vòng. Tiến trình 6 hiện đang là leader do nó có số ưu tiên lớn nhất



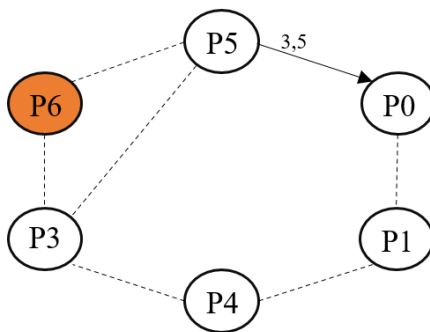
- Bước 1: Tiến trình 6 bị lỗi



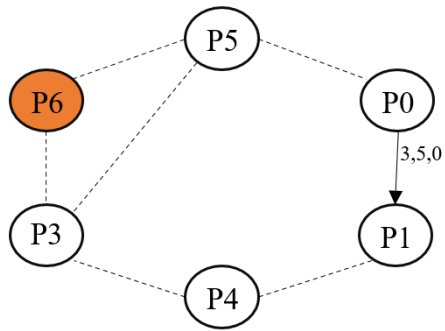
- Bước 2: Tiến trình 3 thông báo rằng tiến trình 6 không phản hồi, vì vậy nó bắt đầu quá trình bầu chọn leader mới. Nó gửi bản tin chứa ID của nó cho node bên phải của mình kèm số ID của nó



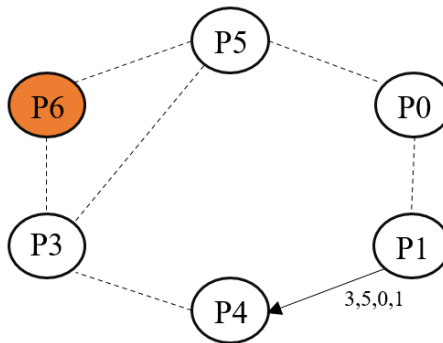
- Bước 3: Tiến trình 5 thêm ID của mình vào bản tin và gửi cho tiến trình tiếp theo



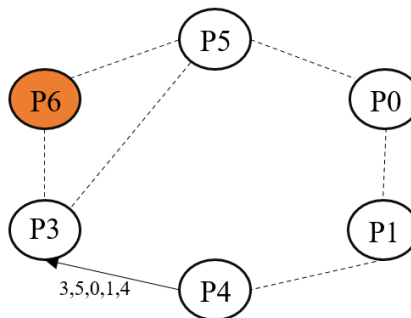
- Bước 4: Tiến trình 0 thêm ID của mình vào bản tin và gửi cho tiến trình tiếp theo



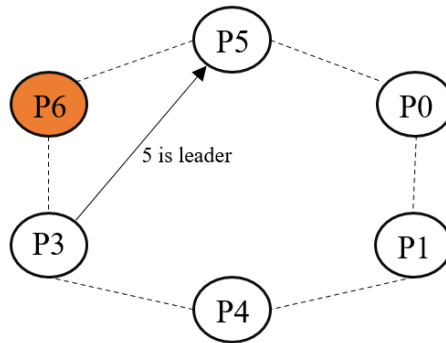
- Bước 5: Tiến trình 1 thêm ID của mình vào bản tin và gửi cho tiến trình tiếp theo



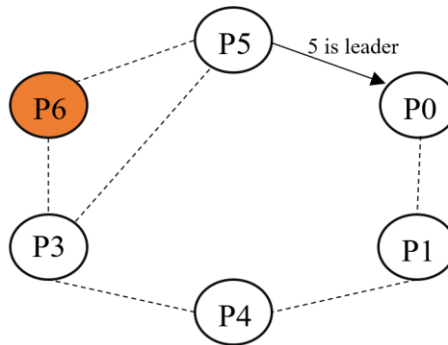
- Bước 6: Tiến trình 4 thêm ID của mình vào bản tin và gửi cho tiến trình tiếp theo



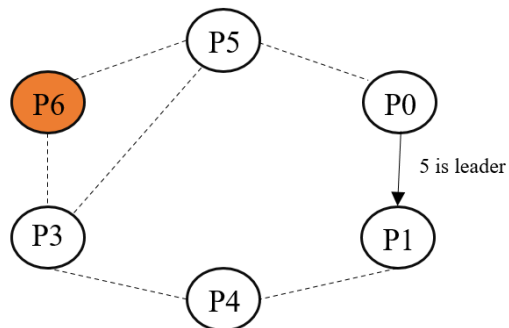
- Bước 7: Khi tiến trình 3 nhận lại được bản tin, nó biết rằng bản tin đã đi xong 1 vòng do có chứa cả ID của nó trong đó. Tiến trình 3 lấy ra ID lớn nhất trong danh sách và gửi bản tin rằng tiến trình đó được bầu làm coordinator mới



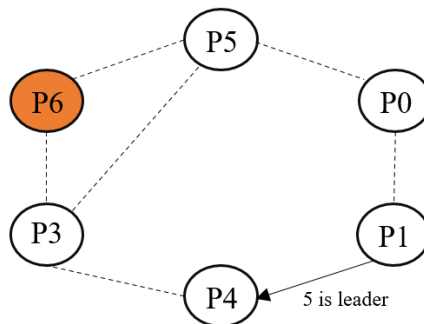
- Bước 8: Tiến trình 5 chuyển thông báo về coordinator mới



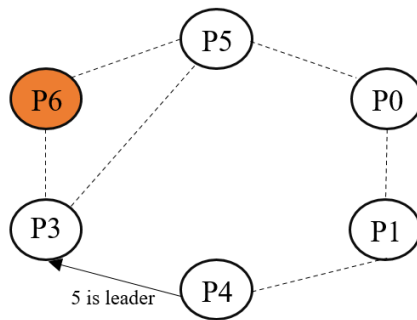
- Bước 9: Tiến trình 0 chuyển thông báo về coordinator mới



- Bước 10: Tiến trình 1 chuyển thông báo về coordinator mới



- Bước 11: Tiến trình 4 chuyển thông báo về coordinator mới



- Bước 12: Tiến trình 3 nhận lại được bản tin về coordinator mới do mình gửi lúc đầu nên dừng lại, không chuyển tiếp bản tin đó nữa.

1.5. Ưu nhược điểm của thuật toán bầu chọn

1.5.1. Ưu điểm:

- Một node coordinator duy nhất làm cho hệ thống dễ dàng quản lý hơn đối với con người. Nó đặt tất cả sự đồng thời trong hệ thống vào một nơi duy nhất, giảm các thành phần lỗi một phần và thêm một nơi duy nhất để tìm kiếm nhật ký và số liệu

- Một coordinator duy nhất có thể làm việc hiệu quả hơn. Nó thường có thể chỉ đơn giản là thông báo cho các hệ thống khác về những thay đổi, thay vì xây dựng sự đồng thuận về những thay đổi sẽ được thực hiện.

- Các coordinator đơn lẻ có thể dễ dàng cung cấp cho khách hàng sự nhất quán vì họ có thể nhìn thấy và kiểm soát tất cả những thay đổi được thực hiện đối với trạng thái của hệ thống.

- Một coordinator duy nhất có thể cải thiện hiệu suất hoặc giảm chi phí bằng cách cung cấp một bộ nhớ cache dữ liệu nhất quán duy nhất có thể được sử dụng mọi lúc.

- Viết phần mềm cho một coordinator duy nhất có thể dễ dàng hơn các cách tiếp cận khác. Coordinator duy nhất không cần phải xem xét rằng các hệ thống khác có thể đang hoạt động trên cùng một trạng thái tại cùng một thời điểm hay không.

1.5.2. Nhược điểm:

- Nếu không phát hiện được sự cố từ coordinator, toàn bộ hệ thống có thể không khả dụng với khách hàng

- Một node coordinator là một điểm tin cậy duy nhất. Nếu node đó thực hiện sai

công việc mà không có ai kiểm tra, nó có thể nhanh chóng gây ra sự cố trên toàn bộ hệ thống.

II. Thuật toán bầu chọn trong môi trường không dây

Các thuật toán truyền thống không còn phù hợp khi cấu hình mạng không ổn định hoặc khi muốn truyền các bản tin đáng tin cậy

Trong phần này, chúng ta tập trung vào các mạng không dây đặc biệt và bỏ qua tính di động của các nút

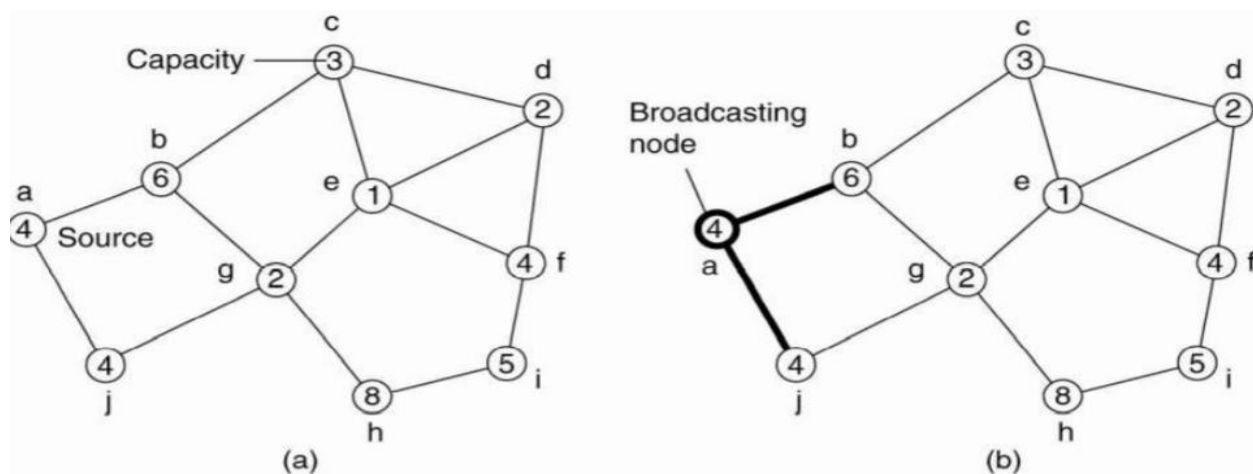
- Các nút kết nối trực tiếp với nhau, không có điểm truy cập chung và các kết nối thường là ngắn hạn

- Thường sử dụng trong các trò chơi trực tuyến nhiều người tham gia hoặc dùng cho việc truyền các tệp tin thông qua mạng internet,...

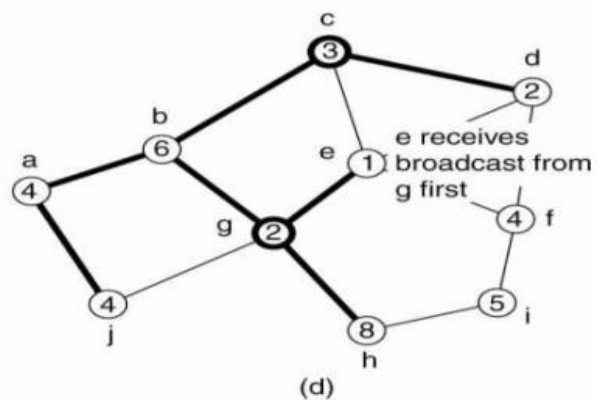
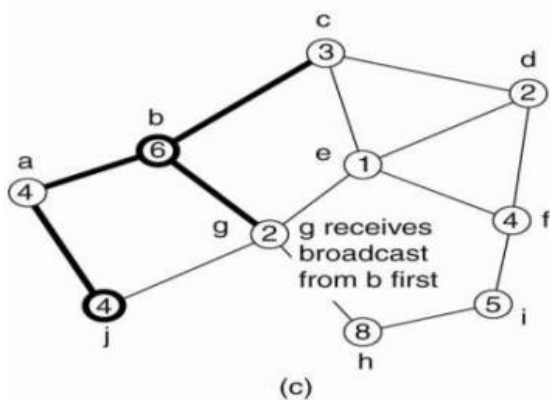
Các thuật toán không dây cố gắng tìm các nút tốt nhất để trở thành bộ điều phối trong khi các thuật toán truyền thống không hài lòng với bất kỳ nút nào

Bất kỳ nút (nguồn) nào đều có thể bắt đầu một cuộc bầu chọn lãnh đạo bằng cách gửi bản tin tới các nút lân cận (các node trong phạm vi)

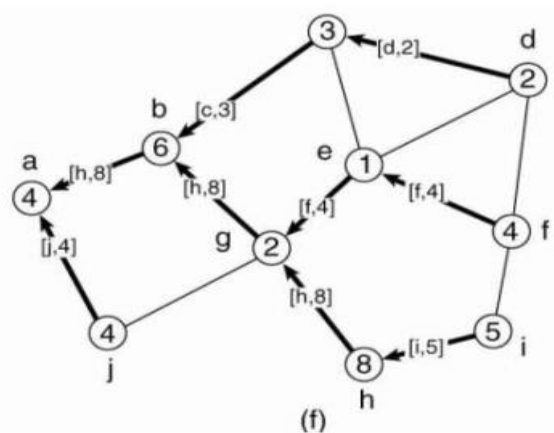
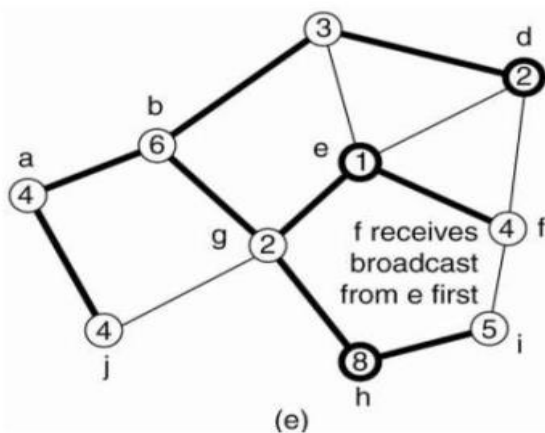
Khi một nút nhận được bản tin bầu cử đầu tiên, thì nút gửi đó sẽ thành nút cha của nó



- Node a là nguồn. Mỗi bản tin chứa ID riêng để quản lý node hiện tại.



Khi node R nhận được bản tin bầu chọn đầu tiên, nó chỉ định nguồn Q là nguồn gốc của nó và chuyển tiếp thông điệp đến tất cả các hàng xóm khác của nó ngoại trừ Q. Khi R nhận được thông báo báo từ một tiến trình ko phải tiến trình cha của nó, nó chỉ xác nhận thông báo đó



Nếu hàng xóm của R có nút cha thì R là nút lá, nếu không, nó sẽ đợi nút con của nó chuyển thông điệp đến những nút hàng xóm của nó. Khi R đã thu thập được acks từ tất cả các hàng xóm của nó, nó sẽ ghi nhận thông điệp từ Q. Một bản tin phản hồi sẽ được gửi lại về nút nguồn ban đầu

- Ở mỗi giai đoạn, nút đủ điều kiện nhất hoặc tốt nhất sẽ được chuyển từ nút con thành nút cha
- Khi nút nguồn đã nhận được tất cả các câu trả lời, nó có thể chọn bộ điều phối mới
- Khi thuật toán bầu chọn thành công, nó sẽ gửi thông báo tới tất cả các nút khác

trong mạng.

- Nếu nhiều hơn một cuộc bầu chọn được bắt đầu (nhiều nút nguồn), một nút chỉ nên tham gia vào một cuộc bầu chọn
- Thông báo bầu chọn được gắn cùng với ID của tiến trình
- Nếu một nút đã chọn được nút parent nhưng nhận thêm được thông báo bầu chọn từ một nút khác có số thứ tự cao hơn, nó sẽ loại bỏ kết quả của cuộc bầu chọn hiện tại và chấp nhận nút được đánh số cao hơn làm nút cha của nó. Điều này đảm bảo chỉ một cuộc bầu chọn duy nhất được lựa chọn

CHƯƠNG II. TRIỂN KHAI VÀ SO SÁNH HIỆU NĂNG CỦA CÁC THUẬT TOÁN BẦU CHỌN

2.1. Yêu cầu đề bài

Triển khai và chạy thử, đưa ra các kết quả thực nghiệm, so sánh hiệu năng của các thuật toán với nhau

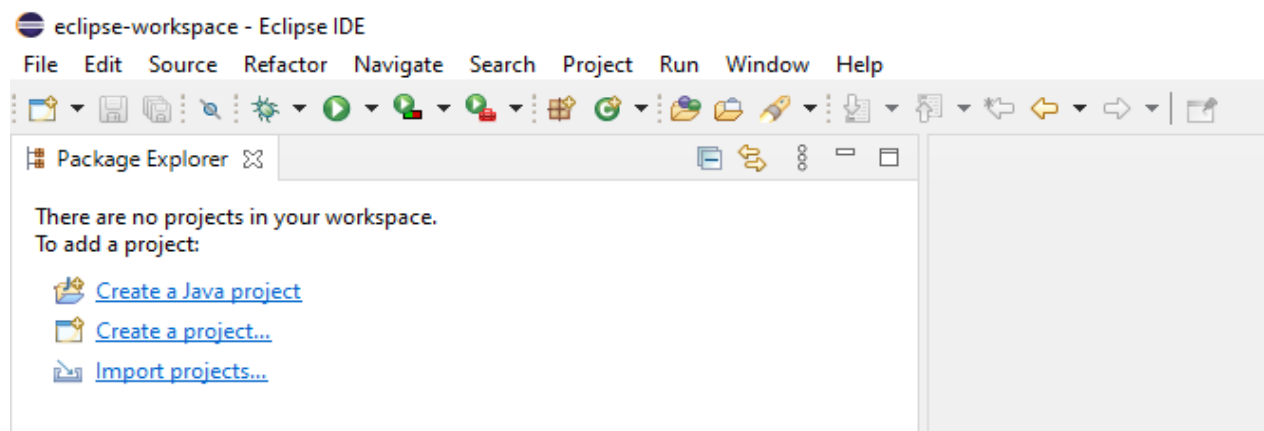
2.2. Thực hiện

Trong phần này chúng em sẽ demo 2 thuật toán bầu chọn truyền thống là thuật toán Bully và thuật toán Ring. Toàn bộ source code demo được lưu trữ tại địa chỉ: <https://github.com/minhquanptit/election>. Các phần mềm cần thiết:

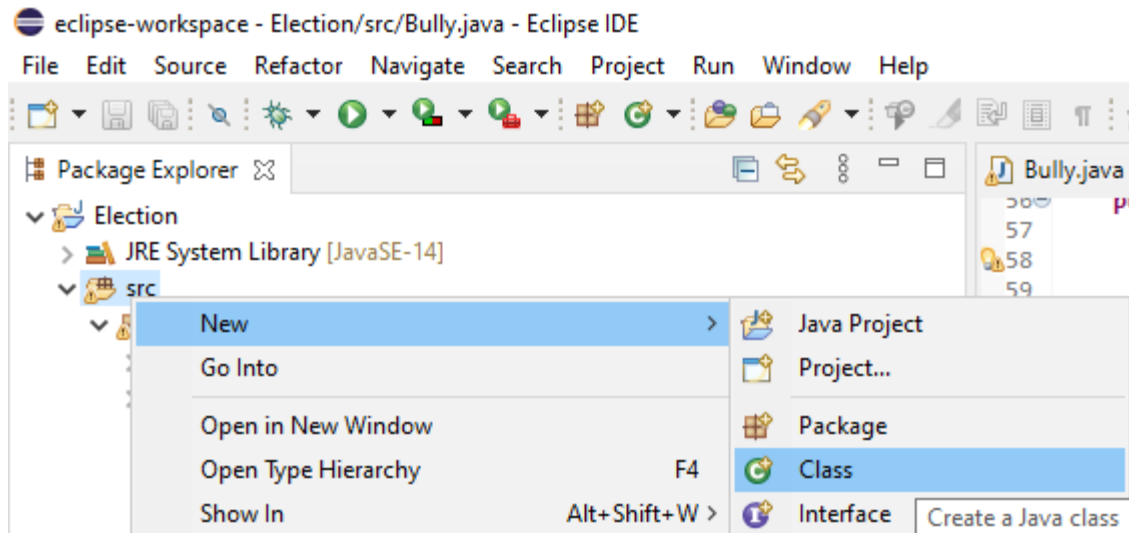
- Môi trường **java JDK**
- **Eclipse** (cần chọn IDE Java for developer trong quá trình cài đặt)

Các bước triển khai

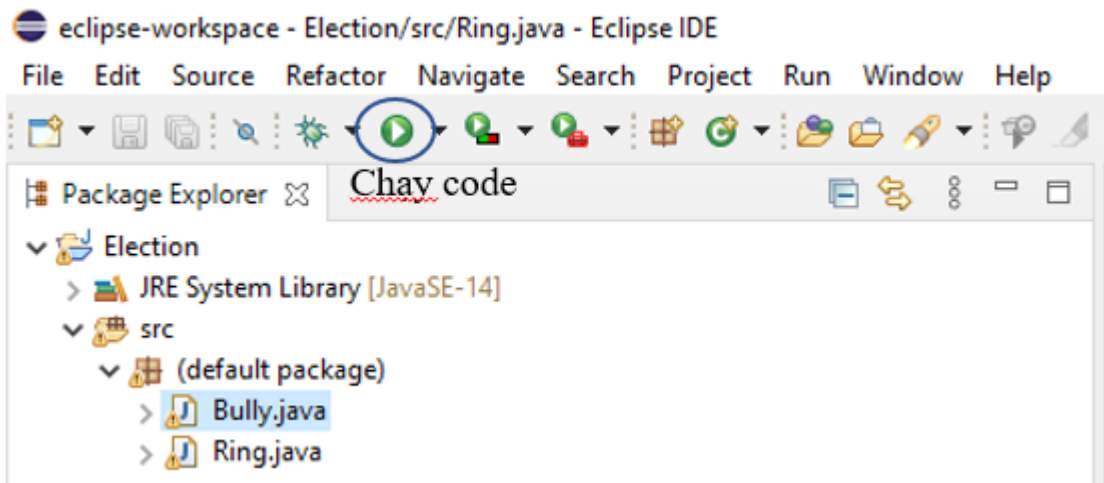
- Bước 1: Khởi chạy chương trình Eclipse



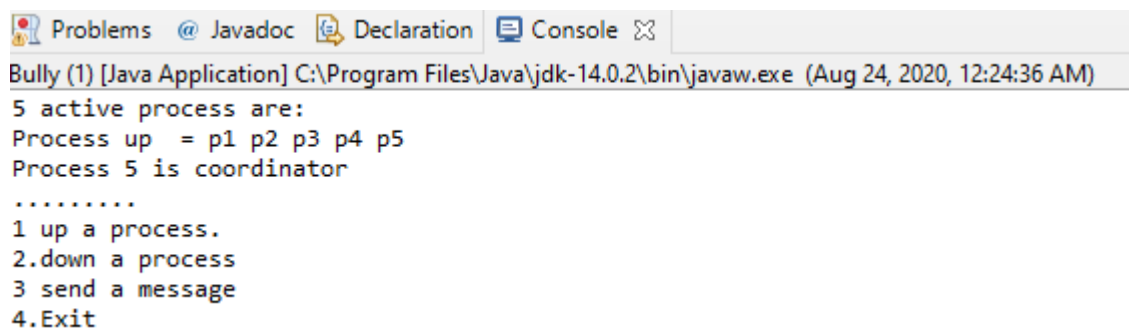
- Tạo project mới tên là **Election**, tại phần **src**, chọn **New** → **Class** và đặt tên cho class mới là **Bully**. Tiếp tục thêm 1 class khác là **Ring**



- Copy source code bully.java và ring.java vào 2 class vừa thêm mới
- Lần lượt chạy từng class vừa thêm để xem kết quả

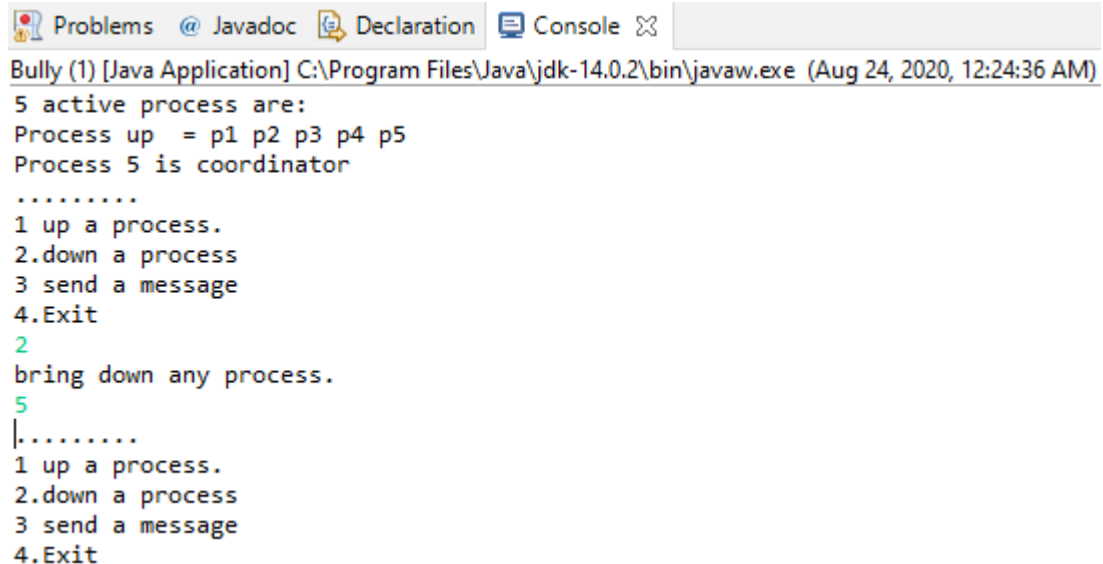


2.2.1. Thuật toán Bully



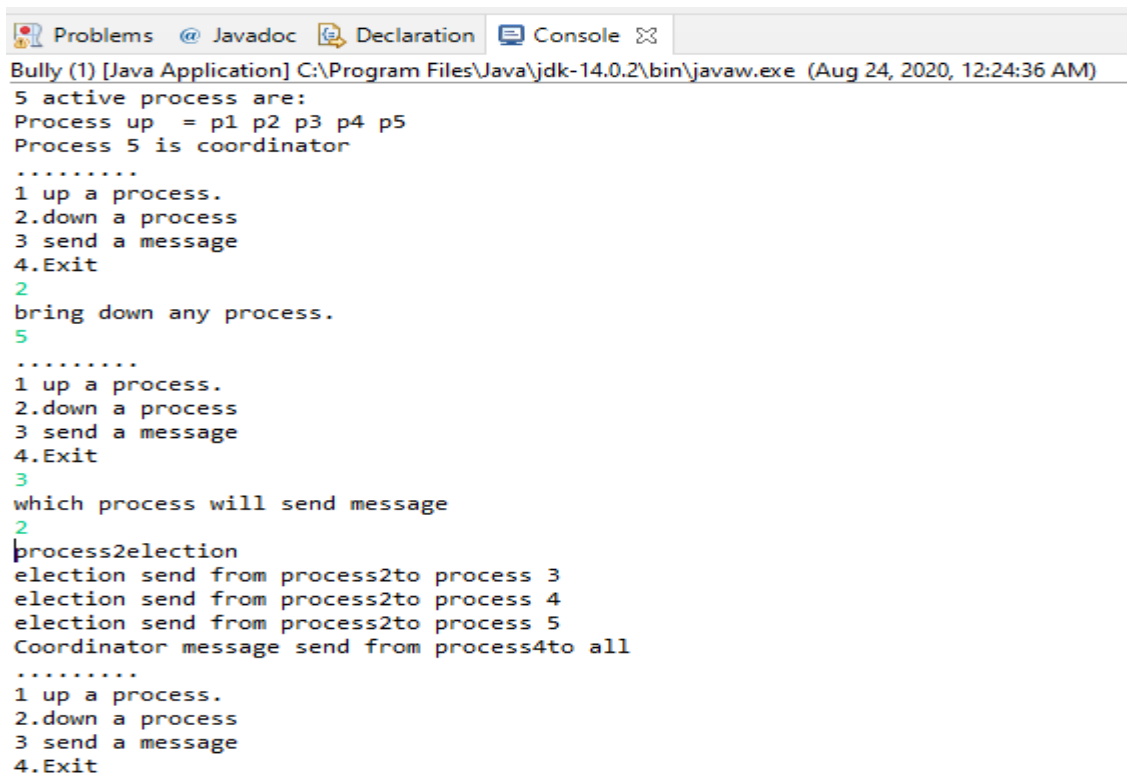
Cấu hình: Hệ thống phân tán gồm 5 tiến trình, trong đó tiến trình số 5 đang được chọn làm coordinator.

Thử down tiến trình số 5 để hệ thống tiến hành khởi chạy thuật toán bầu chọn để bầu ra coordinator mới → Chọn 2 -> chọn 5



```
Problems @ Javadoc Declaration Console
Bully (1) [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Aug 24, 2020, 12:24:36 AM)
5 active process are:
Process up = p1 p2 p3 p4 p5
Process 5 is coordinator
.....
1 up a process.
2.down a process
3 send a message
4.Exit
2
bring down any process.
5
|.
.....
1 up a process.
2.down a process
3 send a message
4.Exit
```

Chọn tiến trình khởi động quá trình bầu chọn (tiến trình này sẽ gửi message tới các tiến trình khác rằng tiến trình 5 down và hệ thống cần coordinator mới) → chọn 3 → chọn 2



```
Problems @ Javadoc Declaration Console
Bully (1) [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Aug 24, 2020, 12:24:36 AM)
5 active process are:
Process up = p1 p2 p3 p4 p5
Process 5 is coordinator
.....
1 up a process.
2.down a process
3 send a message
4.Exit
2
bring down any process.
5
.....
1 up a process.
2.down a process
3 send a message
4.Exit
3
which process will send message
2
process2election
election send from process2to process 3
election send from process2to process 4
election send from process2to process 5
Coordinator message send from process4to all
.....
1 up a process.
2.down a process
3 send a message
4.Exit
```

Tiến trình 4 lúc này đã được chọn làm coordinator mới (do có process ID lớn nhất). Nó

sẽ tiến hành quảng bá thông báo tới tất cả các nút còn lại trong hệ thống để thông báo rằng mình đã được chọn.

Thử active lại tiến trình 5 để kiểm tra hoạt động của hệ thống: Chọn 1 → chọn 5 → hệ thống bầu lại coordinator là 5 do ID của nó lớn hơn tiến trình số 4

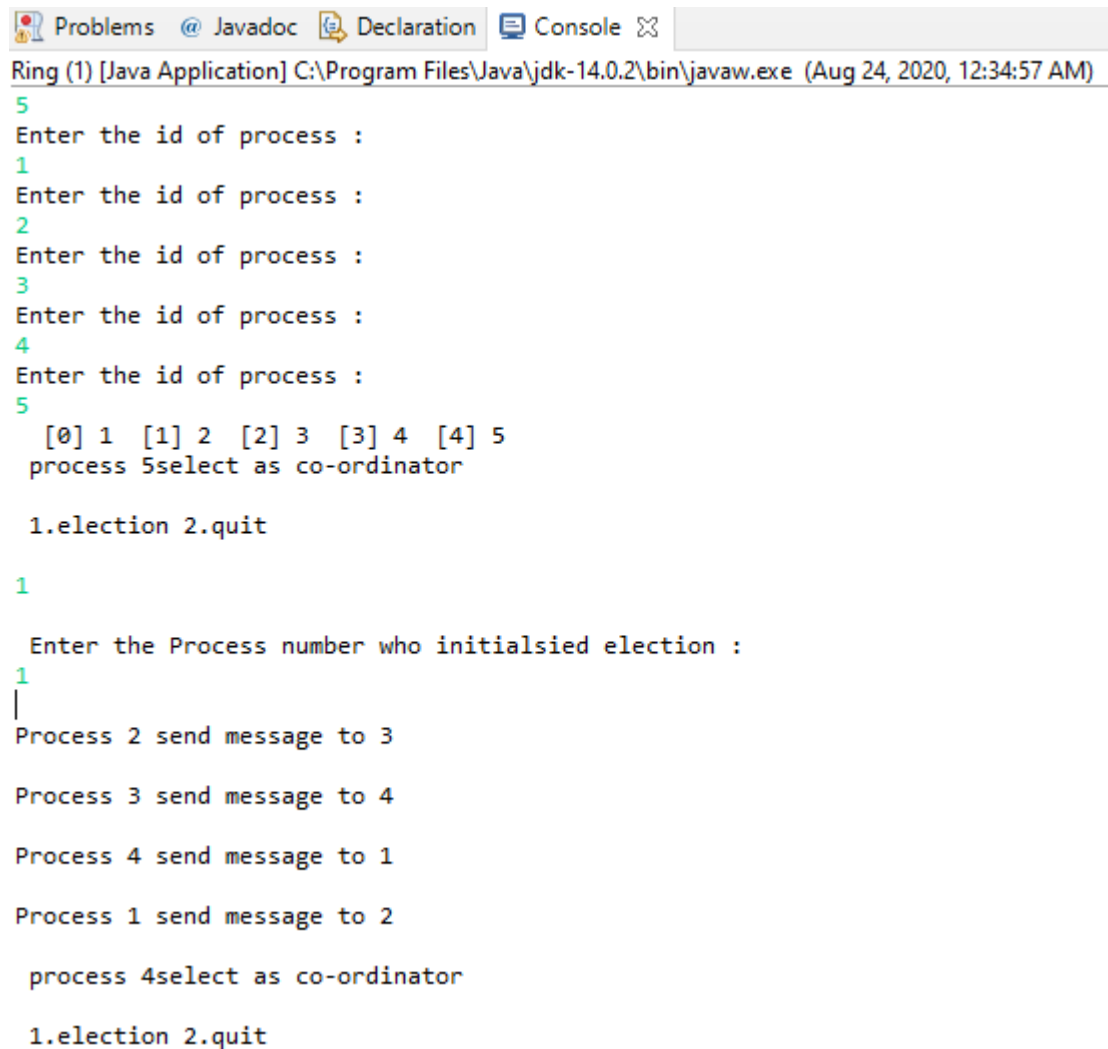
```
Problems Javadoc Declaration Console
Bully (1) [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Aug 24, 2020, 12:24:36 AM)
3 send a message
4.Exit
2
bring down any process.
5
.....
1 up a process.
2.down a process
3 send a message
4.Exit
3
which process will send message
2
process2election
election send from process2to process 3
election send from process2to process 4
election send from process2to process 5
Coordinator message send from process4to all
.....
1 up a process.
2.down a process
3 send a message
4.Exit
1
bring proces up
5
process 5 is co-ordinator
.....
1 up a process.
2.down a process
3 send a message
4.Exit
```

2.2.2. Thuật toán Ring

```
Problems Javadoc Declaration Console
Ring (1) [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Aug 24, 2020, 12:34:57 AM)
Enter the number of process :
5
Enter the id of process :
1
Enter the id of process :
2
Enter the id of process :
3
Enter the id of process :
4
Enter the id of process :
5
| [0] 1 [1] 2 [2] 3 [3] 4 [4] 5
process 5select as co-ordinator

1.election 2.quit
```

Cấu hình hệ thống gồm 5 tiến trình, được đánh số ID từ 1 đến 5. Tiến trình số 5 đang được chọn làm coordinator do có số ID lớn nhất



```
Ring (1) [Java Application] C:\Program Files\Java\jdk-14.0.2\bin\javaw.exe (Aug 24, 2020, 12:34:57 AM)
5
Enter the id of process :
1
Enter the id of process :
2
Enter the id of process :
3
Enter the id of process :
4
Enter the id of process :
5
[0] 1 [1] 2 [2] 3 [3] 4 [4] 5
process 5select as co-ordinator

1.election 2.quit

1
Enter the Process number who initialisied election :
1
|
Process 2 send message to 3
Process 3 send message to 4
Process 4 send message to 1
Process 1 send message to 2
process 4select as co-ordinator

1.election 2.quit
```

Chọn **1. Election** để bắt đầu quá trình bầu chọn (shutdown tiến trình số 5) và lựa chọn tiến trình số 1 làm tiến trình khởi động thuật toán bầu chọn. Theo thuật toán Ring, các nút sẽ gửi bản tin cho nút bên phải của mình bao gồm số ID của nó và các số ID nhận được từ nút trước đó. Sau khi bản tin được chuyển 1 vòng tới tất cả các nút trong mạng thì nút số 4 có ID lớn nhất nên được chọn làm coordinator mới

2.2.3. So sánh hiệu năng của 2 thuật toán

TÀI LIỆU THAM KHẢO

1. Garcia-Molina, Hector. "Elections in a distributed computing system." IEEE transactions on Computers 1 (1982): 48-59.
2. Kordafshari, M. S., et al. "Modified bully election algorithm in distributed systems." WSEAS Transactions on Information Science and Applications 2.8 (2005): 1189-1194.
3. Antonoiu, Gheorghe, and Pradip K. Srimani. "A self-stabilizing leader election algorithm for tree graphs." Journal of Parallel and Distributed Computing 34.2 (1996): 227-232.
4. Basu, Sandipan. "An efficient approach of election algorithm in distributed systems." Indian Journal of Computer Science and Engineering (IJCSE) 2.1 (2011): 16-21.