Parallélisme et Algorithmique Répartie :

Projet :Algorithme d'Election de Dolev Klawe-Rodeh et de Hirschberg Sinclair

Réalisé par :

Boujbair Oussamae

DATA-INE2

Supervisé par :

Pr.NAJA NAJIB

INSTITUT NATIONAL DES POSTES ET TELECOMMUNICATION

12 MARS 2022
Rabat, Maroc

Abstract :

A distributed algorithm is an algorithm designed to run on computer hardware constructed from interconnected processors. Distributed algorithms are used in different application areas of distributed computing, such as telecommunications, scientific computing, distributed information processing, and real-time process control. Standard problems solved by distributed algorithms include leader election, consensus, distributed search, spanning tree generation, mutual exclusion, and resource allocation.

Distributed algorithms are a sub-type of parallel algorithm, typically executed concurrently, with separate parts of the algorithm being run simultaneously on independent processors, and having limited information about what the other parts of the algorithm are doing.

A distributed system also called distributed is opposed to the classic model called client-server by the absence of a central server. Delete this server has beneficial consequences: the entire network no longer depends on this single entity and thus a distributed system is more robust. In a system distributed system, no entity has overall knowledge of the system and its state. It is designed to perform simple or complex tasks on multiple machines or cores of a processing unit. To cooperate between these processes, management and communication protocols must be used. effective communication which sometimes requires a coordinator. A coordinator tor is used to orchestrate processes in different nodes and organize them, however, it may fail or break down. Election Algorithms try to guarantee a high availability of a leader who maintains the coordination tasks. In this document, we will discuss the different algorithms and election methods used for a variety of problems and distinct architectures.

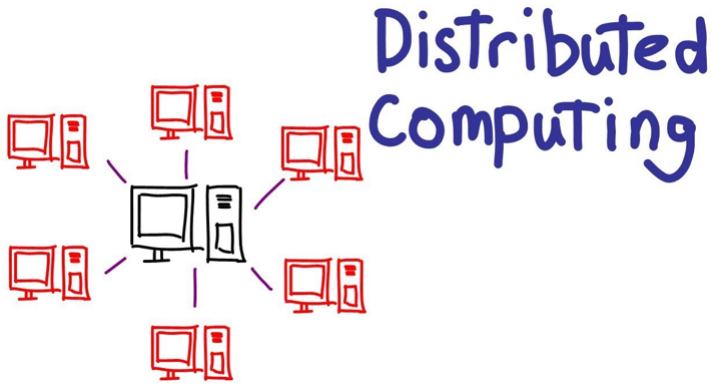
Key words: distributed algorithm, leader election, parallel algorithm, Election Algorithms

Contents

Introduction

1 Système réparti :

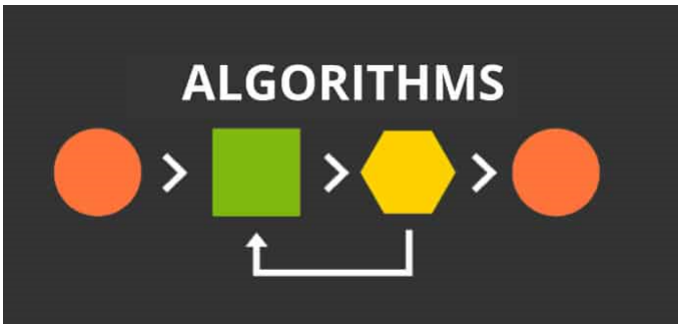
Système réparti est un ensemble de machines autonomes connectées par un réseau, et équipées d'un logiciel dédié à la coordination des activités du système ainsi qu'au partage de ses ressources.



2 Algorithme ?

Un algorithme est une suite finie et non ambiguë d'instructions et d'opérations permettant de résoudre une classe de problèmes.

Le domaine qui étudie les algorithmes est appelé l'algorithmique. On retrouve aujourd'hui des algorithmes dans de nombreuses applications telles que le fonctionnement des ordinateurs3, la cryptographie, le routage d'informations, la planification et l'utilisation optimale des ressources, le traitement d'images, le traitement de textes, la bio-informatique.



3 Algorithmes d'election :

Les algorithmes d'élection choisissent un processus parmi un groupe de processeurs pour agir en tant que coordinateur. Si le processus de coordinateur se bloque pour certaines raisons, un nouveau coordinateur est élu sur un autre processeur. L'algorithme d'élection détermine essentiellement où une nouvelle copie du coordinateur doit être redémarrée. L'algorithme d'élection suppose que chaque processus actif dans le système a un numéro de priorité unique. Le processus avec la plus haute priorité sera choisi comme nouveau coordinateur.

Par conséquent, lorsqu'un coordinateur échoue, cet algorithme choisit le processus actif qui a le numéro de priorité le plus élevé, puis ce numéro est envoyé à chaque processus actif dans le système distribué.

Un algorithme d'election satisfait les propriétés suivantes:

-Chaque processus posséde le même algorithme local.

-Il est décentralisé: il peut être initié par un sous-ensemble arbitraire de processus.

-Sureté: un seul processus est élu.

- Vivacité: un processus doit être élu en temps fini.

Objectif :

Elire un processus parmi d'autres.

Pourquoi ?

-En AD : algorithmes utilisent un coordinateur ou initiateur

-Exemple : algorithme d'exclusion mutuelle centralisé

-Le plus souvent : coordinateur = processus de plus grand numéro

-Panne du coordinateur :nommer un nouveau coordinateur → élection

Algorithmes d'Élections

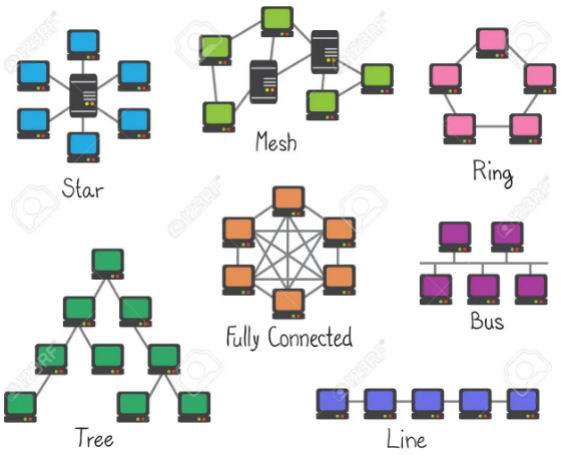
- **But :** élire un seul processus *Pi* parmi un groupe de processus *P1*…*Pn*.
- **Utilité :** élire un processus maître, un coordinateur ou un serveur central.
- Chaque processus Pi maintient une variable $\acute{E}lu_i$ (null s'il n'est pas l'élu).

- **Propriétés à satisfaire :** ∀ pi,
  - **Sûreté :** $\acute{E}lu_i$ = null ou $\acute{E}lu_i$ = Vrai **(processus élu).**

  - **Vivacité :** Pi participe et aura $\acute{E}lu_i$ = null ou $\acute{E}lu_i$ = Vrai.

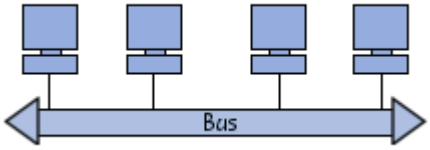
4 Topologie

La topologie d'un système distribué est l'agencement des éléments (liens, nœuds, etc.) d'un réseau de communication. La topologie du système distribué peut être utilisée pour définir ou décrire l'agencement de divers types de réseaux de télécommunication. Nous avons une variété de topologies sur lesquelles nous pouvons agir différemment en fonction de la structure du réseau.



4.1 Bus

Une topologie en bus est l'organisation la plus simple d'un réseau. En effet, dans une topologie en bus tous les ordinateurs sont reliés à une même ligne de transmission par l'intermédiaire de câble, généralement coaxial. Le mot " bus " désigne la ligne physique qui relie les machines du réseau.

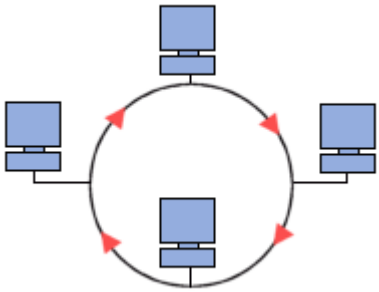


Cette topologie a pour avantage d'être facile à mettre en oeuvre et de posséder un fonctionnement simple. En revanche, elle est extrêmement vulnérable étant donné que si l'une des connexions est défectueuse, l'ensemble du réseau en est affecté.

4.2 Anneau

Une topologie en anneau est une configuration réseau dans laquelle les connex- ions de périphérique créent un chemin de données circulaire. Chaque appareil en réseau est connecté à deux autres, comme des points sur un cercle. Ensemble, les périphériques dans une topologie en anneau sont appelés réseau en anneau.

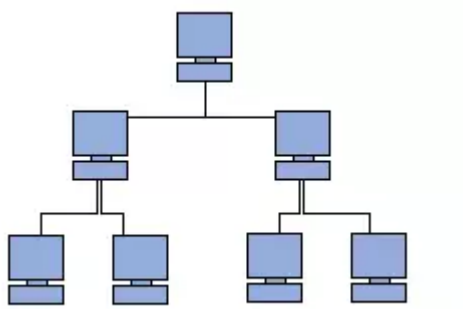
Dans un réseau en anneau, les paquets de données voyagent d'un appareil à l'autre jusqu'à ce qu'ils atteignent leur destination. La plupart des topologies en anneau permettent aux paquets de voyager uniquement dans une seule di- rection, appelée réseau en anneau unidirectionnel. D'autres permettent aux données de se déplacer dans les deux sens, appelées bidirectionnelles.



En réalité, dans une topologie anneau, les ordinateurs ne sont pas reliés en boucle, mais sont reliés à un répartiteur (appelé MAU, Multistation Access Unit) qui va gérer la communication entre les ordinateurs qui lui sont reliés en impartissant à chacun d'entre-eux un temps de parole.

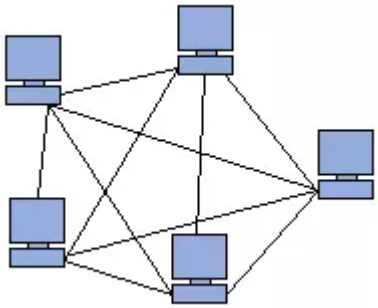
4.3 Arbre

Dans la topologie arbre, tous les ordinateurs sont connectés comme les branches d'un arbre. Dans la mise en réseau informatique, la topologie arborescente est connue sous le nom de combinaison d'une topologie de réseau Bus et Start.
Les principaux avantages de cette topologie sont une meilleure flexibilité et une meilleure évolutivité.



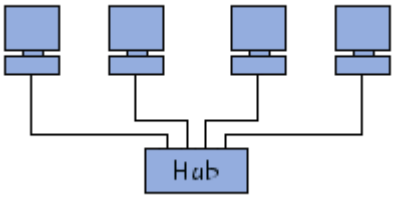
4.4 Maille

Une topologie maillée est une configuration réseau dans laquelle chaque ordina- teur et périphérique réseau est interconnecté les uns avec les autres. Cette con- figuration de topologie permet à la plupart des transmissions d'être distribuées même si l'une des connexions tombe en panne.



4.5 Etoile

Dans une topologie en étoile, les ordinateurs du réseau sont reliés à un système matériel central appelé concentrateur (en anglais hub, littéralement moyen de roue). Il s'agit d'une boîte comprenant un certain nombre de jonctions auxquelles il est possible de raccorder les câbles réseau en provenance des ordinateurs. Celui-ci a pour rôle d'assurer la communication entre les différentes jonctions.



Contrairement aux réseaux construits sur une topologie en bus, les réseaux suivant une topologie en étoile sont beaucoup moins vulnérables car une des connexions peut être débranchée sans paralyser le reste du réseau. Le point névralgique de ce réseau est le concentrateur, car sans lui plus aucune communication entre les ordinateurs du réseau n'est possible.

En revanche, un réseau à topologie en étoile est plus onéreux qu'un réseau à topologie en bus car un matériel supplémentaire est nécessaire (le hub).

5 Exemples d'algorithmes d'Election :

-Algorithme d'Election de vague Echo de Segall.

-Algorithme d'Election de Chang-Roberts.

-Algorithme d'Election de Dolev Klawe-Rodeh.

-Algorithme d'Election de Hirschberg Sinclair.

Etude de quelques algorithmes d'election :

6 Algorithme d'election de Dolev Klawe-Rodeh

6.1 Idée de l'algorithme :

L'algorithme nécessite que les canaux soient FIFO.

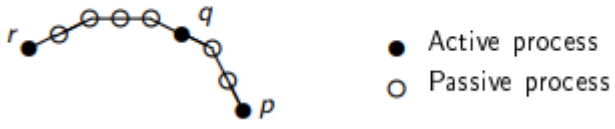
L'algorithme calcule d'abord le plus petit ID et le fait connaître à chaque processus, alors le processus avec cet ID devient leader et tous les autres sont vaincu.

L'algorithme est plus facile à comprendre si l'on le considère comme s'il était exécuté par les identifiants des processus qui fonctionnent comme des bots (proxy) autorisés par le processus correspondants pour participer à l'élections.

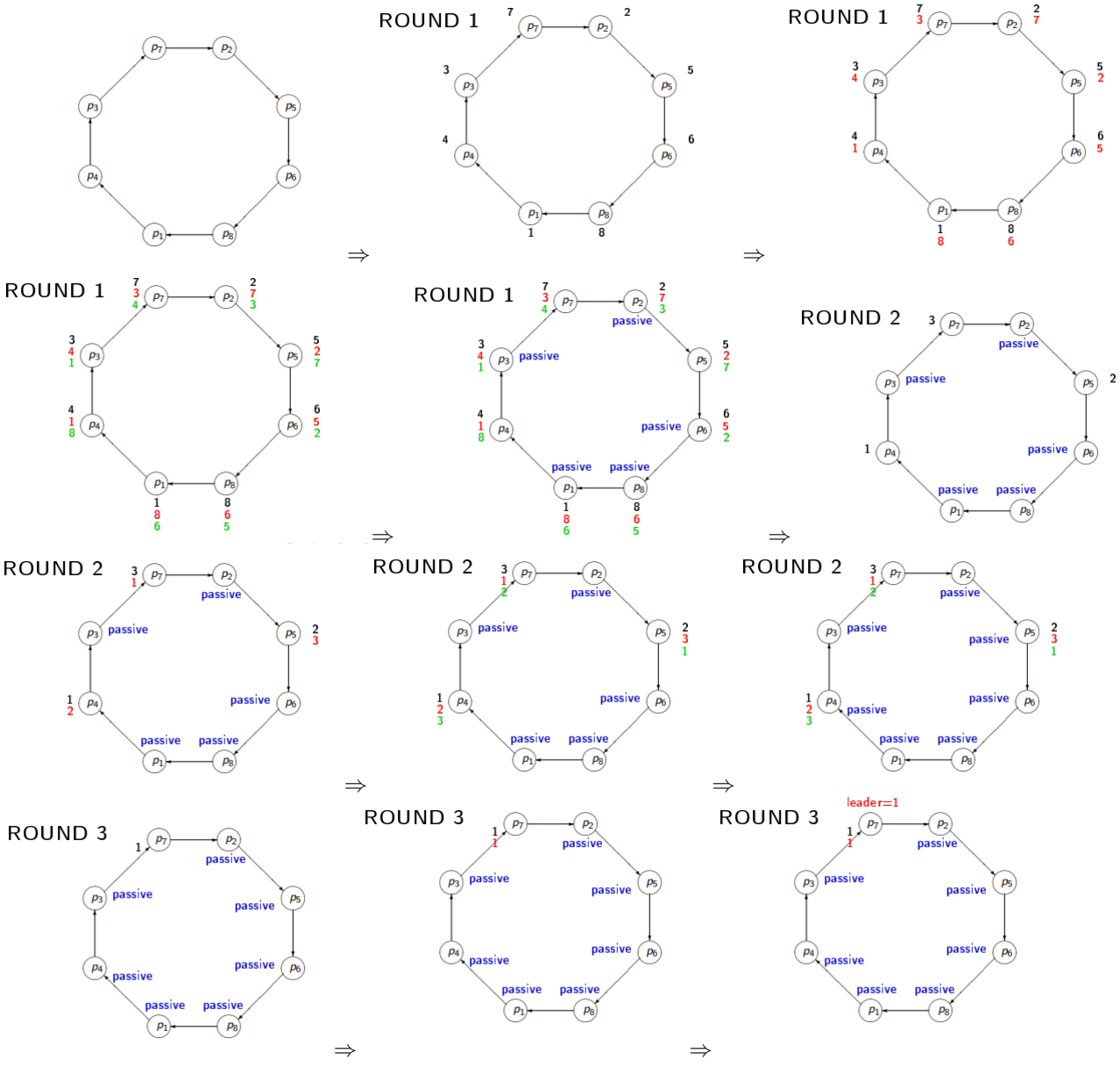
Initialement, chaque identifiant est actif , mais à chaque tour, certains identifiants deviennent passifs .

À chaque tour, un identifiant actif se compare aux deux identifiants actifs voisins dans le sens des aiguilles d'une montre et dans le sens inverse des aiguilles d'une montre ; s'il s'agit d'un minimum local, il survit au tour, sinon il devient passif .

Parce que tous les identifiants sont différents, un identifiant à côté d'un minimum local n'est pas un minimum local, ce qui implique qu'au moins la moitié des identifiants ne survivent pas au tour. Par conséquent, après au plus log N tours, il ne reste qu'un seul ID actif, qui est le gagnant.



Pour rendre possible une comparaison avec r et p, l'ID q est envoyé (en la direction de l'anneau) au processus tenant ID p et r est transmis non seulement au processus tenant q mais aussi, plus loin, au processus tenant p .



6.2 Algorithme de Dolev Klawe-Rodeh :

---

var *cip : P* ***init*** *p ;* (* Current ID of p *)

acnp : P ***init*** *udef ;* (* ID of active anticlockwise neighbor *)

winp : P ***init*** *udef ;* (* ID of winner *)

statep : (active, passive, leader, lost) ***init*** *active ;*

begin if p is initiator **then** statep := active **else** statep := passive ;

while*winp = udef* **do**

begin if statep = active **then**

```
                begin send <one, cip>; receive <one, q> ; acnp := q;
                  if acnp = cip then (* acnp is the minimum *)
                     begin send <small, acnp> ; winp := acnp ; receive <small, q>
                     end
                  else (* acnp is current ID of neighbor *)
                     begin send <two, acnp>; receive <two, q> ;
                        if acnp < cip and acnp < q
                           then cip := acnp else statep := passive
                        end
                  end
   else (* statep = passive *)
        begin receive <one, q> ; send <one, q> ;
              receive m ; send m ;
              (* m is either <two, q>, or <small, q>*)
              if m is amessage hsmall, qi then winp := q
        end
   end;
   if p = winp then statep := leader else statep := lost
   end
```

Le processus p est actif dans un tour si au début de cette tournée, il détient un identifiant cip actif. Sinon p est passif et vient de retransmettre tous les messages qu'il reçoit.

Un processus actif envoie son ID courant au prochain processus actif et obtient l'ID actuel du processus actif précédent utilisant des messages du type miel.

L'ID reçu est stocké (dans la variable acnp).

```
var cip : P init p ; (* Current ID of p *)
```

```
        acnp : P init udef ; (* ID of active anticlockwise neighbor *)

        winp : P init udef ; (* ID of the winner *)

        statep : (active, passive, leader, lost) init active ;
begin if p is initiator then statep := active else statep := passive ;

            while winp = udef do

              begin if statep = active then

                  begin send <one, cip>; receive <one, q> ; acnp := q ;

                      if acnp = cip then (* acnp is the minimum *)

                        begin send <small, acnp> ; winp := acnp ;

                            receive <small, q>

                        end

                      else (* acnp is current ID of neighbor *)

                        begin send <two, acnp>; receive <two, q> ;

                            if acnp < cip and acnp < q

                                then cip := acnp else statep := passive

                            end

                      end
```

L'ID reçu est stocké (dans la variable acnp) et si l'ID survit au tour, ce sera l'ID actuel de p au tour suivant.

Pour déterminer si l'ID acnp survit à ce tour il est comparé à la fois au cip et à l'ID actif, reçu dans le message de type htwoi .

Le processus p envoie un message htwo, message acppi à rendre cette décision possible dans le prochain processus actif.

```
    var cip : P init p ; (* Current ID of p *)

        acnp : P init udef ; (* ID of active anticlockwise neighbor *)

        winp : P init udef ; (* ID of the winner *)
```

```
        statep : (active, passive, leader, lost) init active ;
begin if p is initiator then statep := active else statep := passive ;
            while winp = udef do
              begin if statep = active then
                  begin send <one, cip>; receive <one, q> ; acnp := q;
                      if acnp = cip then (* acnp is the minimum *)
                          begin send <small, acnp> ; winp := acnp ;
                              receive <small, q>
                          end
                      else (* acnp is current ID of neighbor *)
                          begin send <two, acnp>; receive <two, q> ;
                              if acnp < cip and acnp < q
                                  then cip := acnp else statep := passive
                              end
                      end
```

An exception occurs when acnp = cip ; in this case this ID is the only remaining active one and it is announced to all processes in hsmall, acnpi message.

```
var cip : P init p ; (* Current ID of p *)
        acnp : P init udef ; (* ID of active anticlockwise neighbor *)
        winp : P init udef ; (* ID of the winner *)
        statep : (active, passive, leader, lost) init active ;
begin if p is initiator then statep := active else statep := passive ;
            while winp = udef do
              begin if statep = active then
```

```
begin send <one, cip>; receive <one, q> ; acnp := q;
    if acnp = cip then (* acnp is the minimum *)
        begin send <small, acnp> ; winp := acnp ;
            receive <small, q>
        end
    else (* acnp is current ID of neighbor *)
        begin send <two, acnp>; receive <two, q> ;
            if acnp < cip and acnp < q
                then cip := acnp else statep := passive
            end
    end
```

L'algorithme Peterson/Dolev-Klawe-Rodeh résout l'élection problème d'échanges de messages O(N log N).

---

7 Algorithme d'election de Hirschberg et Sinclair 1980 :

7.1 Définition :

L'algorithme Hirschberg – Sinclair est un algorithme distribué conçu pour le problème d'élection de leader dans un réseau en anneau synchrone . Il porte le nom de ses inventeurs, Dan Hirschberg et.

L'algorithme nécessite l'utilisation d'ID uniques (UID) pour chaque processus. L'algorithme fonctionne en phases et envoie son UID dans les deux sens. Le message sort sur une distance de 2 sauts, puis le message revient au processus d'origine. Pendant que les messages se dirigent vers la sortie, chaque processus de réception comparera l'UID entrant au sien. Si l'UID est supérieur à son propre UID, il continuera le message. Sinon, si l'UID est inférieur à son propre UID, il ne transmettra pas les informations. À la fin d'une phase, un processus peut déterminer s'il enverra des messages au prochain tour en indiquant s'il a reçu ses deux messages entrants. Les phases se poursuivent jusqu'à ce qu'un processus reçoive ses deux messages de sortie, de ses deux voisins. À ce stade, le processus sait qu'il s'agit du plus grand UID de l'anneau et se déclare le leader.

Hypothèses:

Les nœudssontcapables de s'organiser en anneaubidirectionnel
Chaque nœud possède :

-un identifiant unique dansl'anneau

-une référence succ[i] vers son successeur

-une référence pred[i] vers son prédécesseur

Aucun nœud ne connaît la taille de l'anneau
Plusieurs candidats simultanés possibles :

-Communications fiables et asynchrones

-Exécution sans faute

-Fonctionnement en rondesasynchrones

-Un nœud actif candidate

-Il émet son identifiantdans les 2 directions sur des distances croissantes

-Le vainqueur de chaque ronde est celui qui a transmissurtoute la distance

-Un identifiant qui fait le tour de l'anneau désigne le leader

7.2 Algorithme :

Valeurs locales
state = active

leader = localid

round = 0

Début de rondek (init. à 0)

if state = active

envoyer <leader, 2k> à voisins de gauche et droite

attendre retour des deux messages

k++ ; cpt =0

7.3 Idée de l'algorithme :

Sur réception de <j, TTL> venant de voisin gauche (droite)

If j = i then

if TTL > 0 then se déclarervainqueur

if j > i and TTL ≥ 1 then

state = inactive

leader = j

if TTL > 1 then

envoyer <j, TTL - 1> a voisindroite(gauche)

else

envoyer <j, 0> a voisin gauche (droit)

if TTL = 0

If (j !=i)

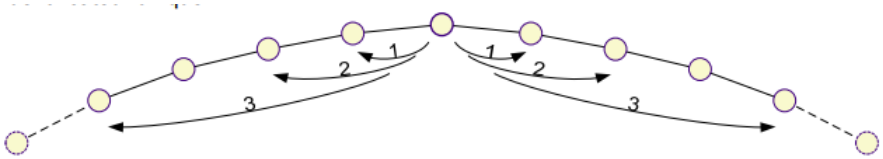
envoyer <j, 0> a voisindroite(gauche)

else

cpt++

if (cpt =2)

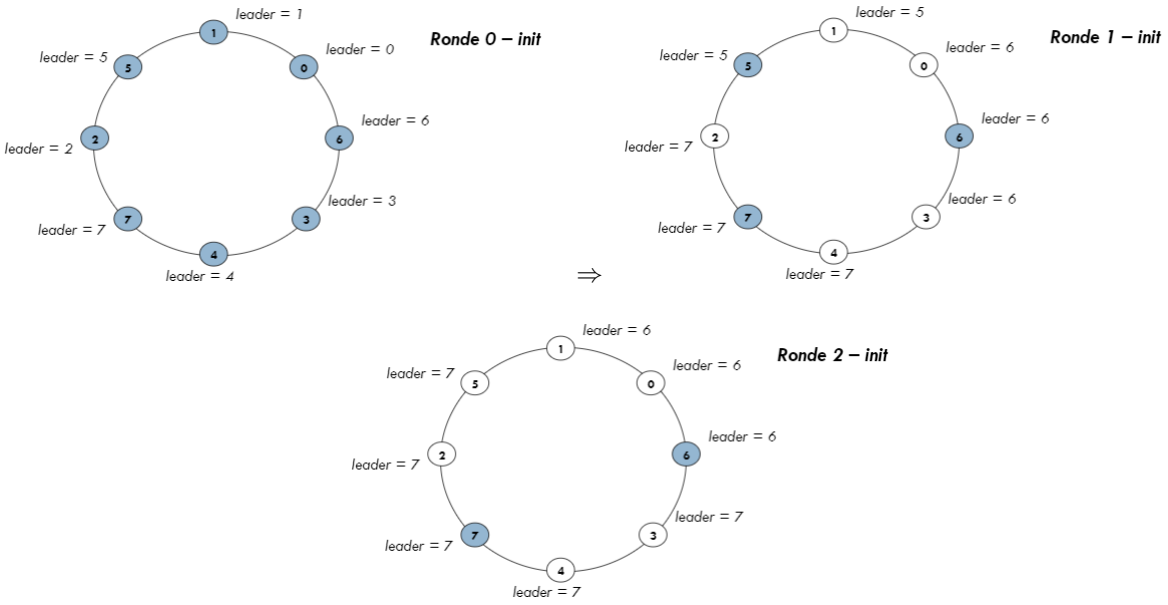
début de nouvelle étape



un noeud reconnu battu devient passif et ne fait que passer les messages qu'il recoit.

Complexité en messages : O(N log N)

7.4 EXEMPLE :



Conclusion :

Un système distribué est un ensemble d'ordinateurs. Ces ordinateurs peuvent fonctionner ensemble, mais tous les systèmes sont indépendant. L'élection d'un chef est une nécessité fondamentale pour les systèmes distribués. Lorsqu'un système est choisi comme chef de file, il doit fonctionner comme un système de gestion; prendre des décisions finales et autres. Il existe plusieurs algorithmes d'élection disponibles dans le système distribué. Dans cet article, nous avons dis- cuté du concept de certains algorithmes existants. Dans les systèmes distribués, l'élection du leader a une variété d'applications telles que: la distribution des clés, la coordination du routage, la coordination des capteurs et le contrôle général.