INFORMATIQUE
Sciences
Université Paris Cité

Leader Election

M2

Lélia Blin

lelia.blin@irif.fr

- Certains algorithmes (par exemple l'algorithme de coloration lente dans l'arbre) demandent un nœud spécial, appelé **leader**.
- Le calcul d'un leader est une forme très simple de **rupture de symétrie**.
- Les algorithmes basés sur les leaders ne présentent généralement pas un haut degré de parallélisme
    - et souffrent donc souvent d'une mauvaise complexité temporelle.
- Cependant, il est parfois utile d'avoir un leader pour prendre des décisions d'une manière simple.

Élection du leader

Le processus de choix d'un leader est connu sous le nom d'élection du leader.

Election dans un anneau

Réseaux anonymes

Définition (anonyme) Un système est anonyme si les nœuds n'ont pas d'identifiants uniques.

Définition (uniforme). Un algorithme est dit uniforme si le nombre de nœuds n est pas connu de l'algorithme (des nœuds, si vous voulez). Si n est connu, l'algorithme est dit non uniforme.

La possibilité d'élire un leader dans un système anonyme dépend de la symétrie du réseau (anneau, graphe complet, graphe bipartite complet, etc.)
ou asymétrique (étoile, nœud unique avec le plus haut degré, etc.).

Résultat d'impossibilité

- Nous allons maintenant montrer qu'il est impossible d'élire un leader anonyme non uniforme pour les anneaux synchrones.
- L'idée est que dans un anneau, la symétrie peut toujours être maintenue.

Résultat d'impossibilité

> **Lemma** Après le tour k de tout algorithme déterministe sur un anneau anonyme, chaque noeud est dans le même état s_k

Preuve par induction: Tous les nœuds commencent dans le même état. Un tour dans un algorithme synchrone consiste en trois étapes : envoi, réception, calcul local. Tous les nœuds envoient le(s) même(s) message(s), reçoivent le(s) même(s) message(s), font le même calcul local, et finissent donc dans le même état.

Résultat d'impossibilité

> **Théorème** (élection du leader anonyme). L'élection déterministe d'un leader dans un anneau anonyme est impossible.

Preuve (avec le lemme précédent) : Si un nœud décide de devenir un leader (ou un non-leader), alors tous les autres nœuds le font aussi, ce qui contredit la spécification du problème pour n > 1. Cela vaut pour les algorithmes non uniformes, et donc aussi pour les algorithmes uniformes. En outre, cela vaut pour les algorithmes synchrones, et donc aussi pour les algorithmes asynchrones.

Réseaux non anonymes

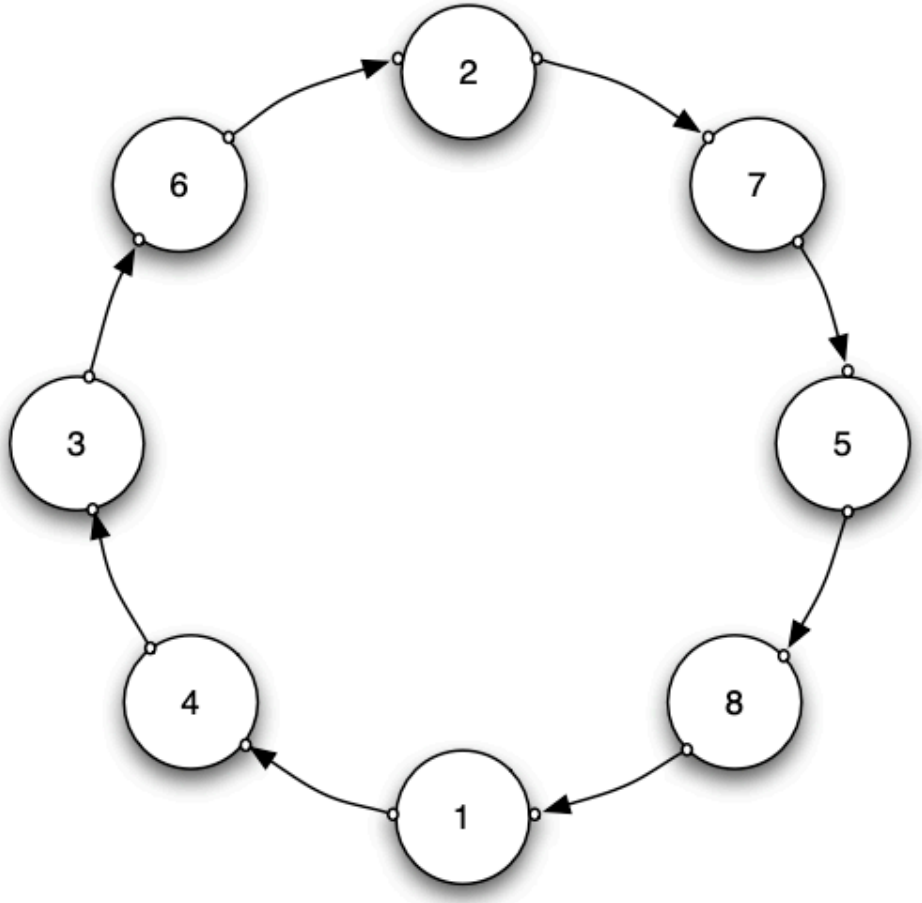
Le sens de direction

- Le sens de direction est la capacité des nœuds à distinguer les nœuds voisins dans un réseaux.
- Dans un anneau, par exemple, un nœud peut distinguer son voisin dans le sens des aiguilles d'une montre et dans le sens inverse.
- Le sens de direction n'est d'aucune utilité pour l'élection dans un réseaux anonyme.
  - réseaux symétriques (par exemple, les graphes complets, les graphes bipartites complets, . . .).

Algorithme 1

La solution la plus simple au problème de l'élection dans un cycle unidirectionnel est décrite de manière informelle et incomplète par :

- Au début, chaque nœud
  - maintient une variable id_max initialisée avec son identifiant
  - puis il envoie son identifiant à son voisin.
- Chaque fois qu'un nœud reçoit une valeur V il fait
  - $id\_max := max\{id\_max, V\}$ et
  - fait suivre V vers son autre voisin.


2
7
5
8
1
4
3
6

Questions

1. Si un seul (ou plusieurs) nœud initie l'élection, comment tous les autres nœuds peuvent participer à l'élection ? (problème du réveil).
2. Quelle est la condition à rajouter pour que l'algorithme se termine? (problème de la terminaison).
    - Qui sait qui a gagné l'élection, comment proclamer le résultat ? (problème de la prise de décision).
3. Combien de messages transitent durant cet algorithme ?
4. Que se passe t'il si
    - un message est perdu ?
    - les nœuds n'ont pas d'identifiant deux à deux distincts ?
    - un message est dupliqué ?

Questions

1. Si un seul (ou plusieurs) nœud initie l'élection, comment tous les autres nœuds peuvent participer à l'élection ? (problème du réveil).

```
Initialisation
        id_max=id
        Envoie <id> à voisin
Réveil spontané
        Intitialisation
Lors de la réception de <V>
        si Etat=endormi
                Initialisation
        si V>id_max alors id_max=V
        Envoie <V> à voisin
```

Questions

2. Quelle est la condition à rajouter pour que l'algorithme se termine? (problème de la terminaison).
    - Qui sait qui a gagné l'élection, comment proclamer le résultat ? (problème de la prise de décision).

```
Initialisation
        id_max=id
        Envoie <id> à voisin
Réveil spontané
        Intitialisation
Lors de la réception de <V>
        si Etat=endormi
                Initialisation
        si V>id_max alors id_max=V
        si V=id alors
                si id_max=id alors Etat:=Gagnant
                sinon Etat:=Perdant
        sinon Envoie <V> à voisin
```

Arguments de preuves

- On suppose que les identifiants sont 2 à deux distincts. Lorsqu'un noeud reçoit son propre identifiant cela signifie qu'il a reçu tous les identifiants présent sur l'anneau car un noeud envoie son propre identifiant avant de faire suivre les autres
- **Remarque**: cet algorithme ne fonctionne que si les liens de communications sont FIFO.
- Donc quand un noeud reçoit son propre identifiant il sait qu'il a terminé
  - il peut donc décidé si il est gagnant ou perdant
  - Il n'y a pas de proclamation de résultat

Complexités en nombre de messages

- Chaque lien de communications "voit passer" tous les identifiants soit n messages
- Dans un anneau à n noeuds il y a n liens
- **Total**: $n \times n$ messages ($O(n^2)$,$\Theta(n^2)$,$\Omega(n^2)$)

Complexité temporelle: idem

Complexités spatiale

Taille des messages

- Si les identifiants sont entre 1 et n il faut $\lceil \log_2 n \rceil$ bits pour coder un entier n
- $O(\log_2 n)$ bits

  Espace mémoire par noeud
- idem

4. Que se passe t'il si
    - un message est perdu ?
    - les nœuds n'ont pas d'identifiant deux à deux distincts ?
    - un message est dupliqué ?

Un message perdu

- le noeud dont l'identifiant est perdu ne terminera jamais.
- Si l'identifiant max est perdu un "mauvais" noeud peut être élu

les nœuds n'ont pas d'identifiant deux à deux distincts ?

- Un noeud peut terminé prématurément

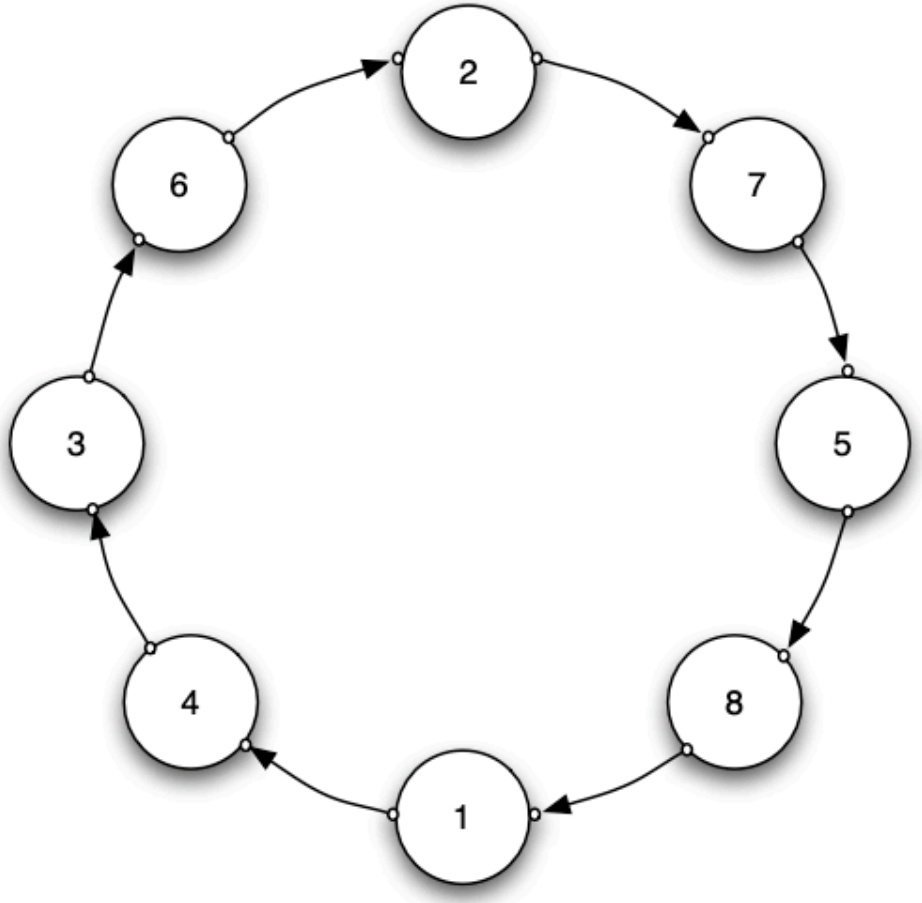
Message dupliqué

- potentiellement plus de message
- mais cela ne perturbera pas l'élection

Algorithme 2

- Nous proposons maintenant un autre algorithme pour l'élection dans un cycle unidirectionnel (sans de direction).
- Chaque nœud i maintient
  - un booléen $participant_i$ qui est initialisé à $Faux$.

```
En cas de réveil spontané
        participant := Vrai;
        Envoyer(<ELECTION,id>);
Lors de la réception d un message  <ELECTION,V>
        Si (V > id) alors
                participant:=Vrai
                Envoyer(<ELECTION,V>);
        Si (V < id) et (non participant) alors
                participant := Vrai;
                Envoyer(<ELECTION,id>);
        Si (V=id) alors Envoyer(<ELU,id>);
Lors de la réception d un message  <ELU,V>
        Le gagnant est le nœud V;
        participant:=Faux;
        si (id!=V) alors Envoyer(<ELU,V>);
```


2
7
5
8
1
4
3
6

Questions

- Qui détecte le gagnant et comment est-il propagé ?
- Supposons que tous les nœuds se réveillent simultanément. Evaluez la complexité en nombre de messages échangés dans les configurations particulières suivantes :
  - Les valeurs sont ordonnées croissantes dans le sens du cycle.
  - Les valeurs sont ordonnées décroissantes dans le sens du cycle.

Qui détecte le gagnant et comment est-il propagé ?

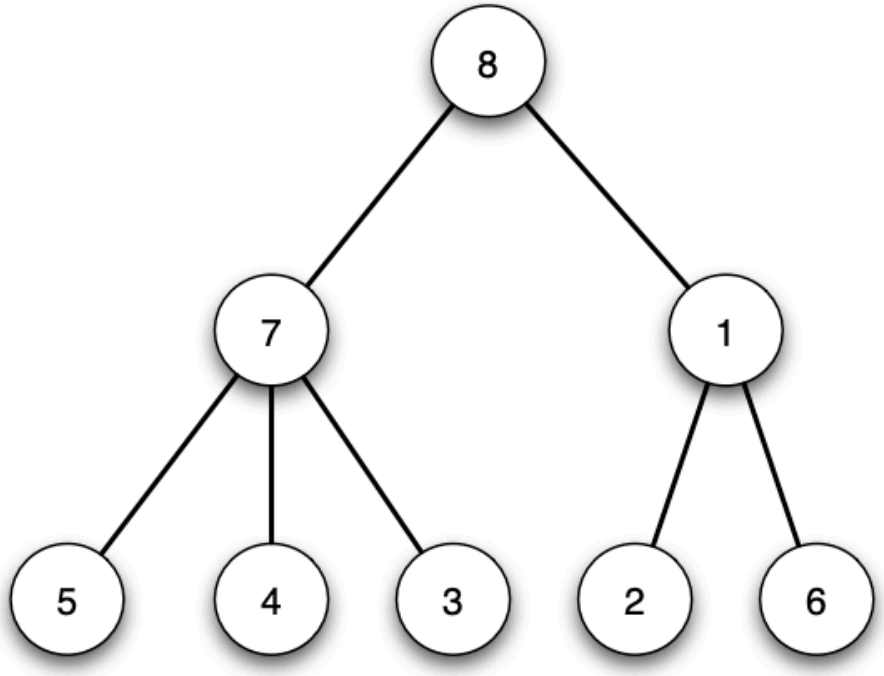
- C'est le gagnant lui même qui détecte qu'il a gagné
- Une fois cette détection faite il fait une proclamation de résultat (message ELU)

Ordre croissant

Ordre décroissant

Complexités

Election dans un arbre

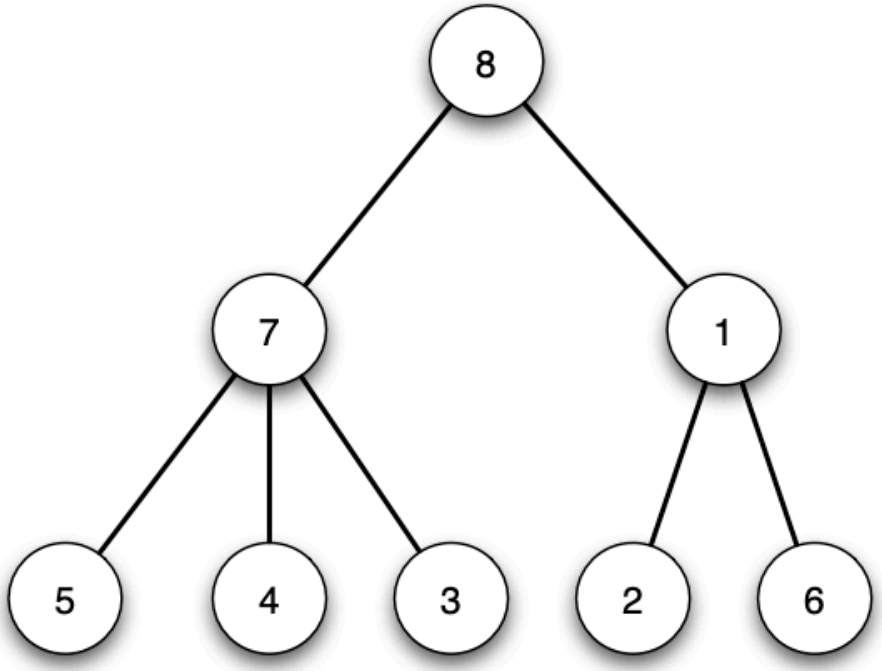


Modèle

- On suppose une topologie en arbre
- Chaque canal est **FIFO**.
- Les nœuds communiquent directement qu'avec leurs voisins
- Chaque sommet a un identifiant unique.

Ecrire un algorithme réparti qui :

- “Réveille” tous les sommets à partir d’au moins un initiateur (réveil spontané).
- Calcule le plus petit identifiant de l’arbre
  - le gagnant étant le noeud de plus petit identifiant
- Pour cela :
  - Faire un parcours des feuilles vers l’intérieur de l’arbre en calculant la plus petite étiquette au fur et à mesure.
  - Propager le résultat de l’élection de voisin en voisin.
    - Si un sommet reçoit son étiquette, il se déclare vainqueur, sinon perdant.


8
7
1
5
4
3
2
6

```
Initialisation
        Etat:=reveillé
        id_min=id
        Voisins:=N(v) //ensemble des voisins de de v
Réveil spontané
        Intitialisation
        Envoie <Reveil> à N(v)
        si |Voisins|=1 //test feuille
                Envoie <Election,id_min> à N(v)
Lors de la réception de <Reveil> envoyé par u
        si Etat!=reveillé
                Initialisation
                Envoie <Reveil> à N(v)\{u}
                si |Voisins|=1 //test feuille
                        Envoie <Election,id_min> à N(v)
```

```
Lors de la réception de <Election,m> envoyé par u
        Voisins:=Voisins\{u}
        si m<id_min alors id_min:=m
        si |Voisins|=1 //test feuille virtuelle
                Envoie <Election,id_min> à Voisins
        si |Voisins|=0
                si id_min=id alors Etat:=gagnant
                sinon Etat:=perdant
                Envoie <Elu,id_min> à N(v)
Lors de la reception de <Elu,m> envoyé par u
        Si Etat=réveillé
                id_min:=m
                si id_min=id alors Etat:=gagnant
                sinon Etat:=perdant
                Envoie <Elu,id_min> à N(v)\{u}
```

Complexités

Nombre de messages

- **Réveil**:
  - $n-1$ messages si un seul noeud se réveille spontanément
  - $2(n-1)$ si tous les noeuds se réveillent spontanément
- **Election et Elu**: n message (2 noeuds détectent la fin)
  - Total $O(n)$ messages
- **Remarque**: même complexité temporelle

Question

- Comment faire l'élection dans un graphe quelconque?