

Algorithmes classiques : élection, terminaison

Plan

Election

- Algorithme le plus simple

- Election sur un anneau

Terminaison

- Terminaison sur un anneau

- Terminaison sur un graphe quelconque

Election

- Principe :

Choisir **un et un seul leader** parmi un ensemble de processus et le faire connaître de tous.

- Hypothèses :

L'élection peut être déclenchée par un processus arbitraire ou éventuellement par **plusieurs** processus.

Leader : le processus qui a le plus grand numéro.

Propriété de l'élection

■ Propriétés :

- ▶ **Sûreté (safety)** : un seul processus doit être élu

■ Utilisation :

- ▶ Régénération d'un jeton perdu : un jeton et un seul doit être recréé.
- ▶ Dans les algorithmes de type "maître-esclave" : élire un nouveau maître en cas de défaillance du maître.

Propriété de l'élection

■ Propriétés :

- ▶ **Sûreté (safety)** : un seul processus doit être élu
- ▶ **Vicacité (liveness)** : un processus doit être élu en un temps fini.

■ Utilisation :

- ▶ Régénération d'un jeton perdu : un jeton et un seul doit être recréé.
- ▶ Dans les algorithmes de type "maître-esclave" : élire un nouveau maître en cas de défaillance du maître.

Plus précisément

Election

Algorithme le plus simple

Election sur un anneau

Algorithme Brute de Force

Hypothèses :

- ▶ Il y a aucune perte de messages
- ▶ Il y a une borne connue sur le temps de communications.

Principe :

- ▶ Les demandes d'élection sont diffusées par inondation.

Algorithme Brute de Force

Hypothèses :

- ▶ Il y a aucune perte de messages
- ▶ Il y a une borne connue sur le temps de communications.

Principe :

- ▶ Les demandes d'élection sont diffusées par inondation.
- ▶ Un processus répond à ceux de numéro inférieur au sien.

Algorithme Brute de Force

Hypothèses :

- ▶ Il y a aucune perte de messages
- ▶ Il y a une borne connue sur le temps de communications.

Principe :

- ▶ Les demandes d'élection sont diffusées par inondation.
- ▶ Un processus répond à ceux de numéro inférieur au sien.
- ▶ Un processus qui ne reçoit aucune réponse constate qu'il est élu.

Algorithme Brute de Force

Hypothèses :

- ▶ Il y a aucune perte de messages
- ▶ Il y a une borne connue sur le temps de communications.

Principe :

- ▶ Les demandes d'élection sont diffusées par inondation.
- ▶ Un processus répond à ceux de numéro inférieur au sien.
- ▶ Un processus qui ne reçoit aucune réponse constate qu'il est élu.

Complexité : $O(n^2)$ messages au pire des cas.

Plus précisément

Election

Algorithme le plus simple

Election sur un anneau

Algorithme sur un anneau

Principe :

- ▶ plusieurs sites peuvent démarrer le processus d'élection

Algorithme sur un anneau

Principe :

- ▶ plusieurs sites peuvent démarrer le processus d'élection
- ▶ Chaque site qui commence une election envoie un jeton en précisant sa valeur

Algorithme sur un anneau

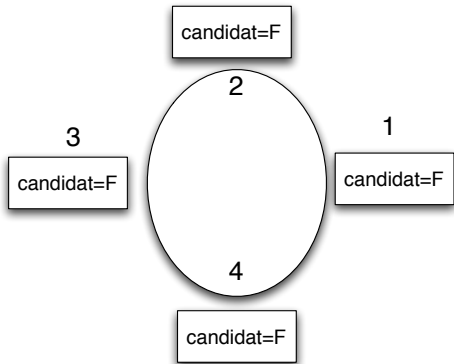
Principe :

- ▶ plusieurs sites peuvent démarrer le processus d'élection
- ▶ Chaque site qui commence une election envoie un jeton en précisant sa valeur
- ▶ Lors qu'un site reçoit un jeton de valeur inférieure, il le supprime

Plus formellement : programme sur le site i

Initialisation de l'élection :

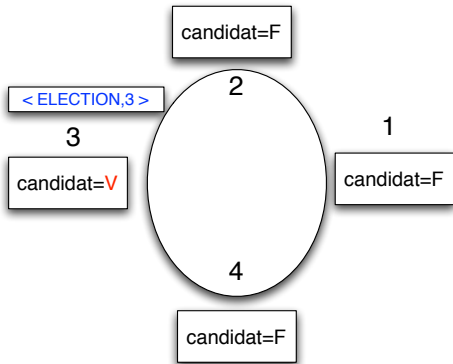
1. $candidate_i \leftarrow vrai$;
2. envoyer $\langle ELECTION, i \rangle$ au suivant ;



Plus formellement : programme sur le site i

Initialisation de l'élection :

1. $\text{candidat}_i \leftarrow \text{vrai}$;
2. envoyer $\langle \text{ELECTION}, i \rangle$ au suivant ;

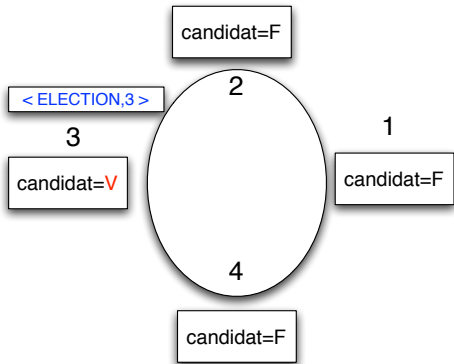


Plus formellement : programme sur le site i

Initialisation de l'élection :

1. $\text{candidat}_i \leftarrow \text{vrai}$;
2. envoyer $\langle \text{ELECTION}, i \rangle$ au suivant ;

Réception de $\langle \text{ELECTION}, \ell \rangle \rightarrow$



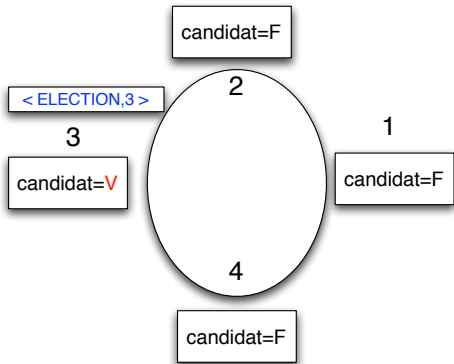
Plus formellement : programme sur le site i

Initialisation de l'élection :

1. $\text{candidat}_i \leftarrow \text{vrai}$;
2. envoyer $\langle \text{ELECTION}, i \rangle$ au suivant ;

Réception de $\langle \text{ELECTION}, \ell \rangle \rightarrow$

2. si $\ell > i$ alors envoyer $\langle \text{ELECTION}, \ell \rangle$ au suivant ;



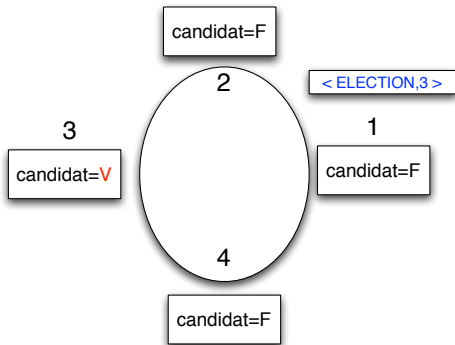
Plus formellement : programme sur le site i

Initialisation de l'élection :

1. $\text{candidat}_i \leftarrow \text{vrai}$;
2. envoyer $\langle \text{ELECTION}, i \rangle$ au suivant ;

Réception de $\langle \text{ELECTION}, \ell \rangle \rightarrow$

2. si $\ell > i$ alors envoyer $\langle \text{ELECTION}, \ell \rangle$ au suivant ;



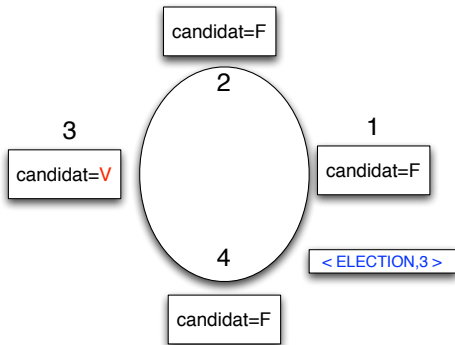
Plus formellement : programme sur le site i

Initialisation de l'élection :

1. $candidate_i \leftarrow vrai$;
2. envoyer $\langle ELECTION, i \rangle$ au suivant;

Réception de $\langle ELECTION, \ell \rangle \rightarrow$

2. si $\ell > i$ alors envoyer $\langle ELECTION, \ell \rangle$ au suivant;
3. si $(\ell < i) \wedge (candidate_i \leftarrow faux)$ alors initialiser une élection.



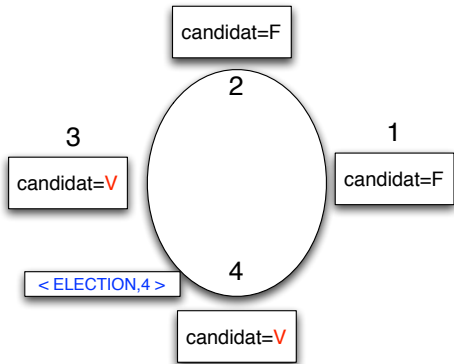
Plus formellement : programme sur le site i

Initialisation de l'élection :

1. $candidate_i \leftarrow vrai$;
2. envoyer $\langle ELECTION, i \rangle$ au suivant;

Réception de $\langle ELECTION, \ell \rangle \rightarrow$

2. si $\ell > i$ alors envoyer $\langle ELECTION, \ell \rangle$ au suivant;
3. si $(\ell < i) \wedge (candidate_i \leftarrow faux)$ alors initialiser une élection.



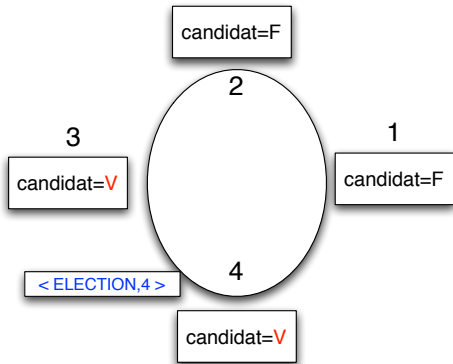
Plus formellement : programme sur le site i

Initialisation de l'élection :

1. $\text{candidat}_i \leftarrow \text{vrai}$;
2. envoyer $\langle \text{ELECTION}, i \rangle$ au *suivant*;

Réception de $\langle \text{ELECTION}, \ell \rangle \rightarrow$

1. si $\ell = i$ alors i est le leader + il diffuse le résultat de l'élection
2. si $\ell > i$ alors envoyer $\langle \text{ELECTION}, \ell \rangle$ au *suivant*;
3. si $(\ell < i) \wedge (\text{candidat}_i \leftarrow \text{faux})$ alors initialiser une élection.



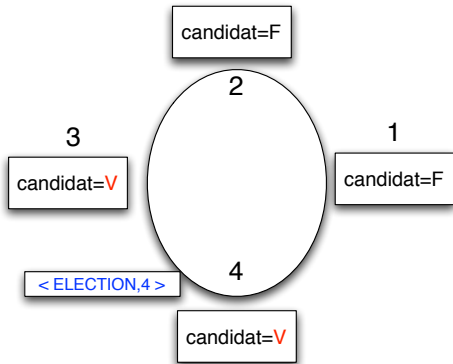
Plus formellement : programme sur le site i

Initialisation de l'élection :

1. $candidate_i \leftarrow vrai$;
2. envoyer $\langle ELECTION, i \rangle$ au suivant _{i} ;

Réception de $\langle ELECTION, \ell \rangle \rightarrow$

1. si $\ell = i$ alors i est le leader + il diffuse le résultat de l'élection
2. si $\ell > i$ alors envoyer $\langle ELECTION, \ell \rangle$ au suivant _{i} ;
3. si $(\ell < i) \wedge (candidate_i \leftarrow faux)$ alors initialiser une élection.



Complexité : $O(n^2)$ messages au pire des cas.

Plan

Election

- Algorithme le plus simple

- Election sur un anneau

Terminaison

- Terminaison sur un anneau

- Terminaison sur un graphe quelconque

Contexte

Le modèle de calcul réparti est composé

- d'un ensemble de processus communiquant par messages.

état **actif** ou état **passif**

Contexte

Le modèle de calcul réparti est composé

- d'un ensemble de processus communiquant par messages.

état **actif** ou état **passif**

- d'un programme pour chaque processus
tant que (true) faire

- 0.1 Attendre un message (état passif)
- 0.2 Exécuter un calcul local suite à la réception d'un message
- 0.3 Lors ce calcul, il peut avoir l'envoi de messages ou la terminaison pour le processus de ce calcul.

Terminaison

- Principe :

Vérifier que le calcul est fini.

Terminaison

- Principe :

Vérifier que le calcul est fini.

- C'est-à-dire :

- ▶ Tous les processus sont dans l'état passif
- ▶ Aucun message est en transit.

Plus précisément

Terminaison

Terminaison sur un anneau

Terminaison sur un graphe quelconque

Algorithme du gardien

- Principe :

1. Parcourir l'anneau dans le sens de communication

Algorithme du gardien

- Principe :

1. Parcourir l'anneau dans le sens de communication

Mécanisme de jeton.

Algorithme du gardien

■ Principe :

1. Parcourir l'anneau dans le sens de communication

Mécanisme de jeton.

2. Vérifier que tous les sites sont dans l'état passif.

Algorithme du gardien

■ Principe :

1. Parcourir l'anneau dans le sens de communication

Mécanisme de jeton.

2. Vérifier que tous les sites sont dans l'état passif.

Le jeton possède une valeur qui compte les sites non-actifs.

Algorithme du gardien

■ Principe :

1. Parcourir l'anneau dans le sens de communication

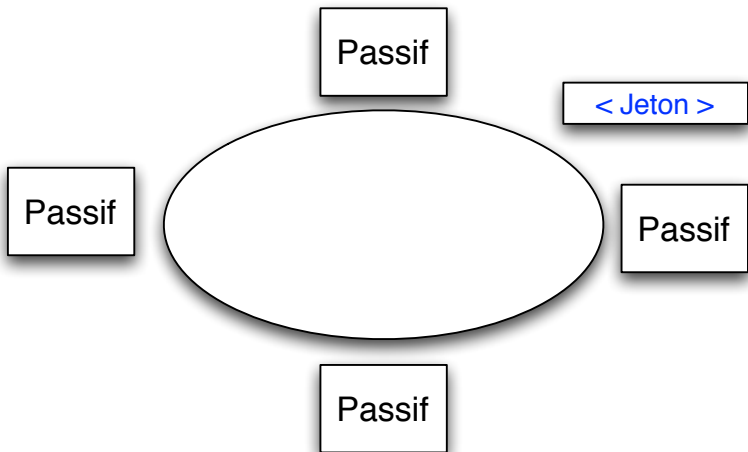
Mécanisme de jeton.

2. Vérifier que tous les sites sont dans l'état passif.

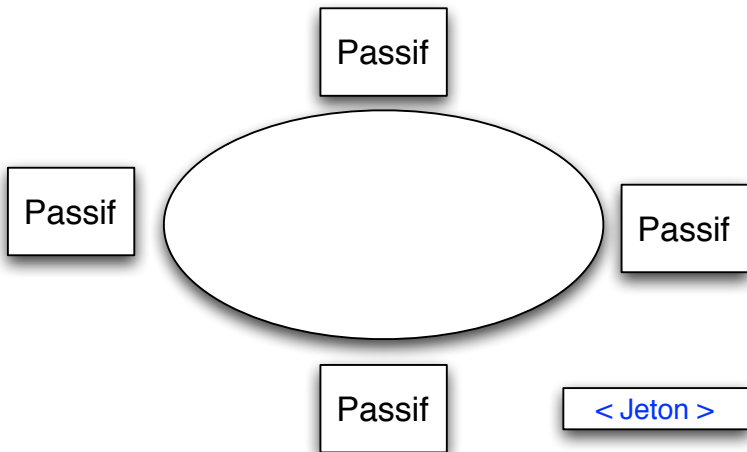
Le jeton possède une valeur qui compte les sites non-actifs.

■ Hypothèse : les sites connaissent le nombre total de sites.

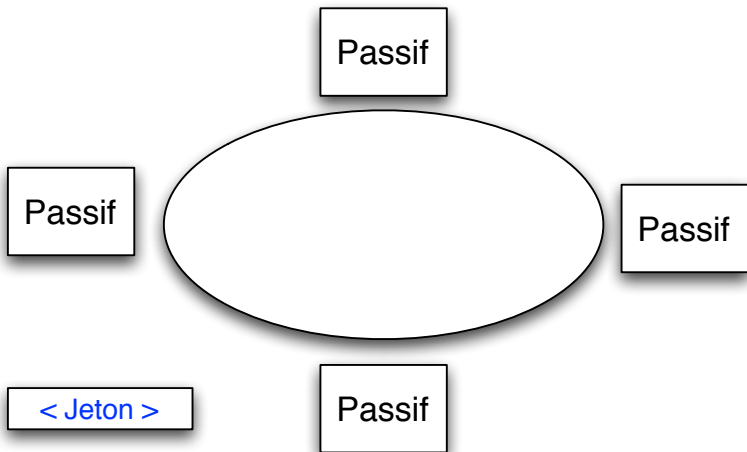
Illustration



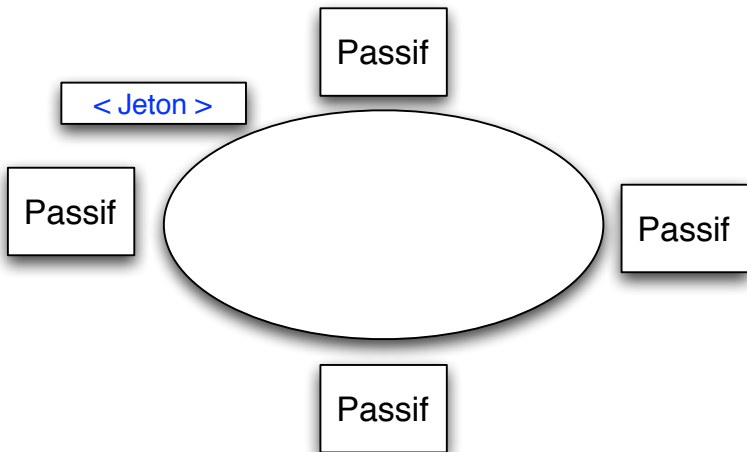
Illustration



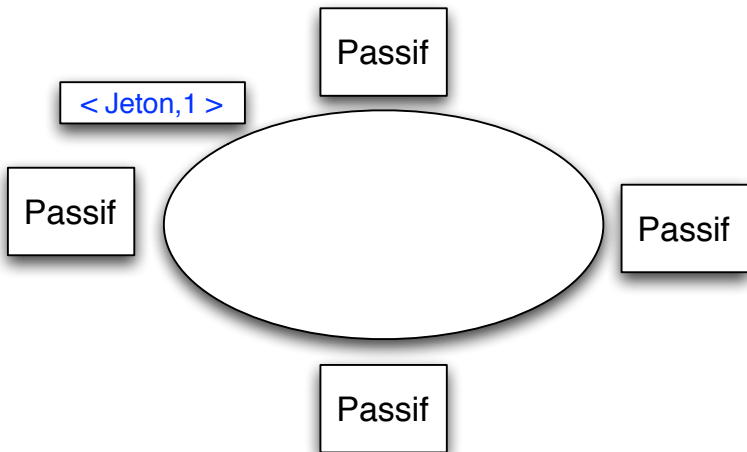
Illustration



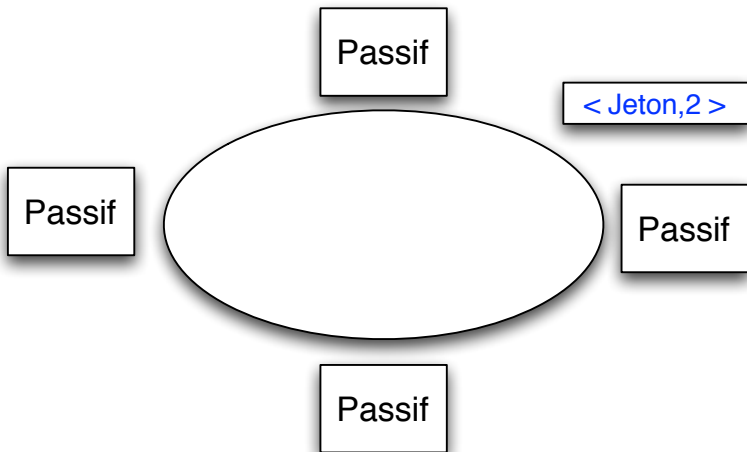
Illustration



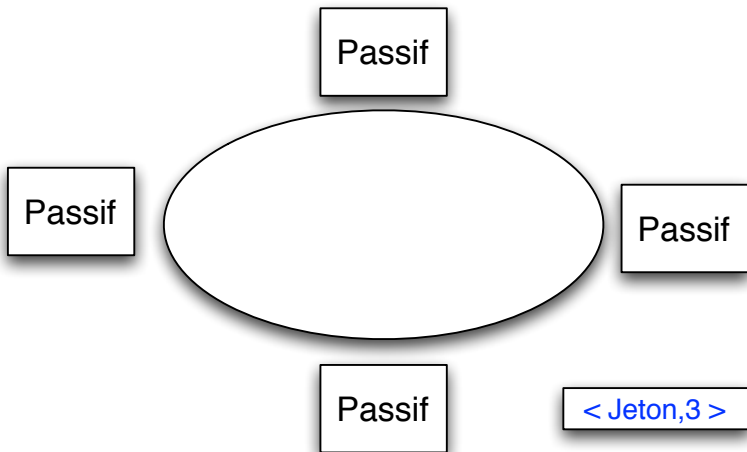
Illustration



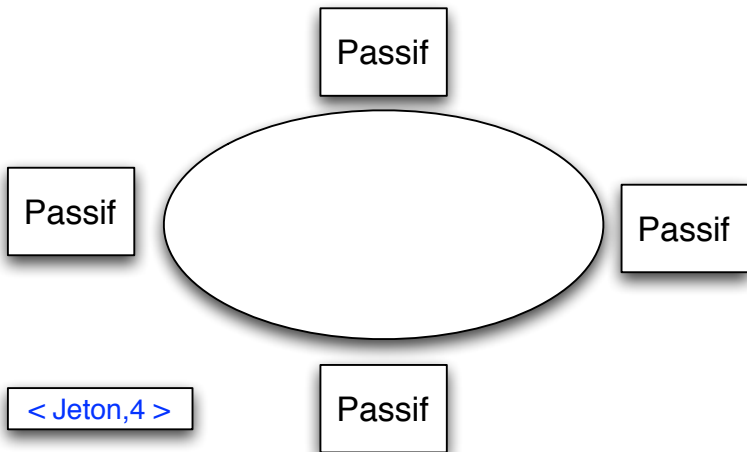
Illustration



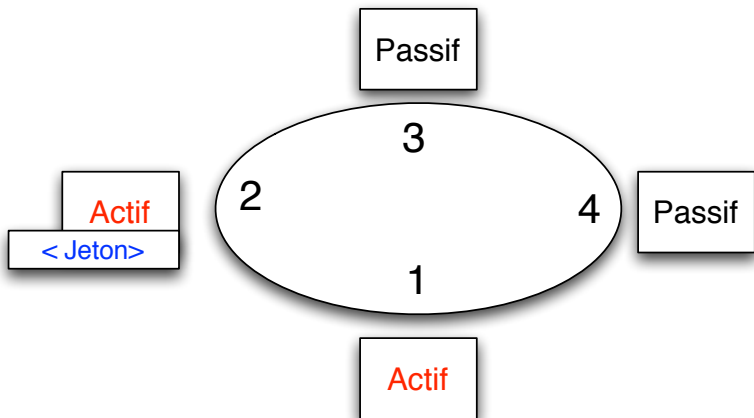
Illustration



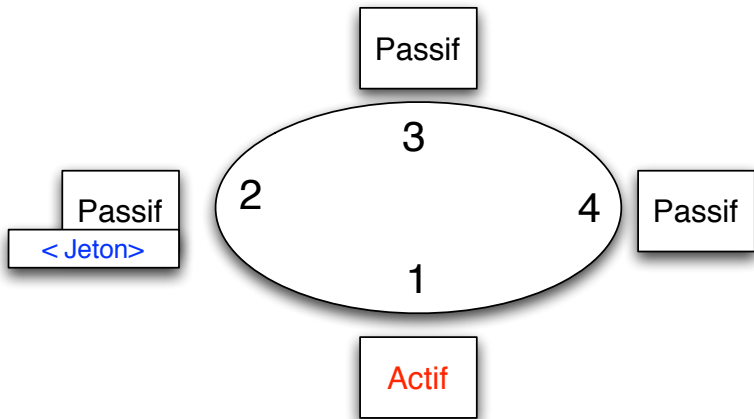
Illustration



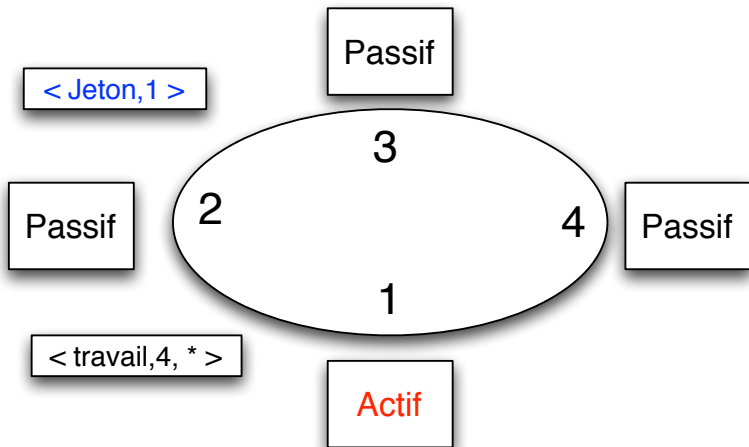
Difficulté : comment savoir qu'un message transit



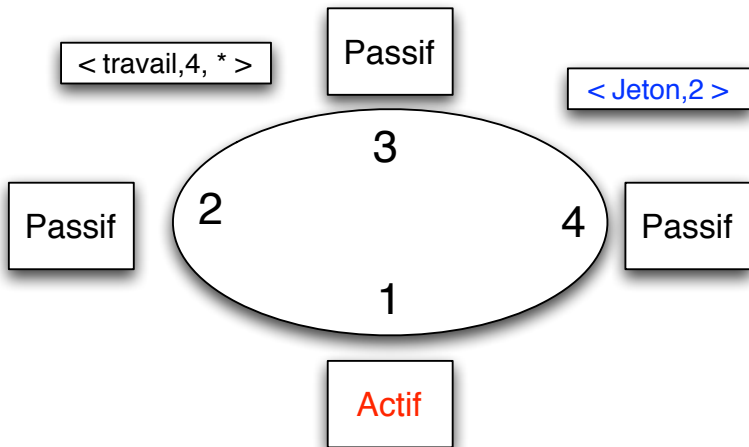
Difficulté : comment savoir qu'un message transit



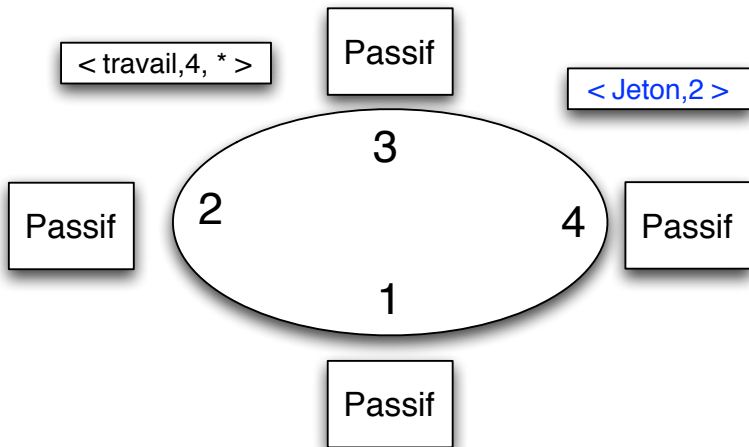
Difficulté : comment savoir qu'un message transit



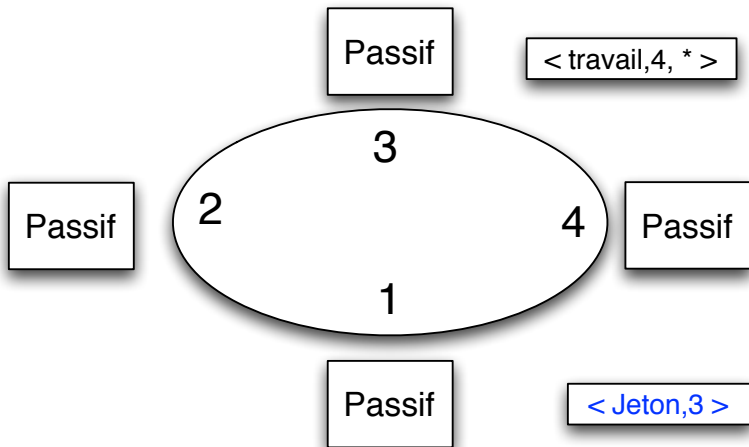
Difficulté : comment savoir qu'un message transit



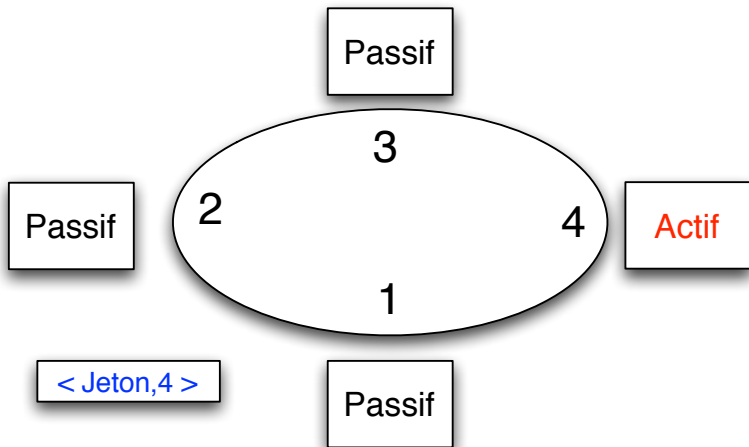
Difficulté : comment savoir qu'un message transit



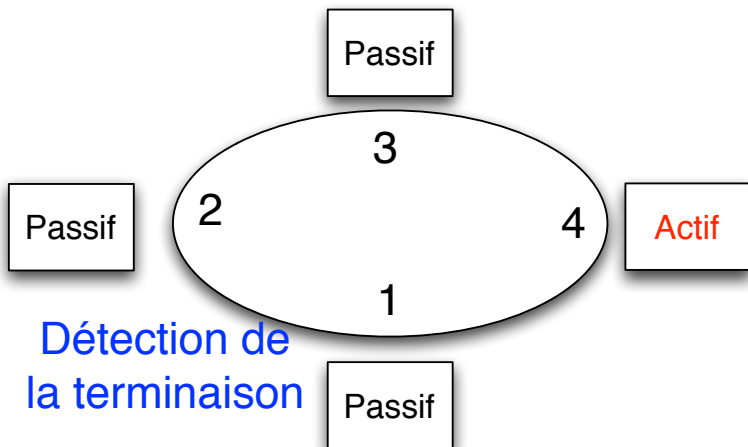
Difficulté : comment savoir qu'un message transit



Difficulté : comment savoir qu'un message transit



Difficulté : comment savoir qu'un message transit



Algorithme du gardien

Difficulté : comment savoir qu'un message est en transit

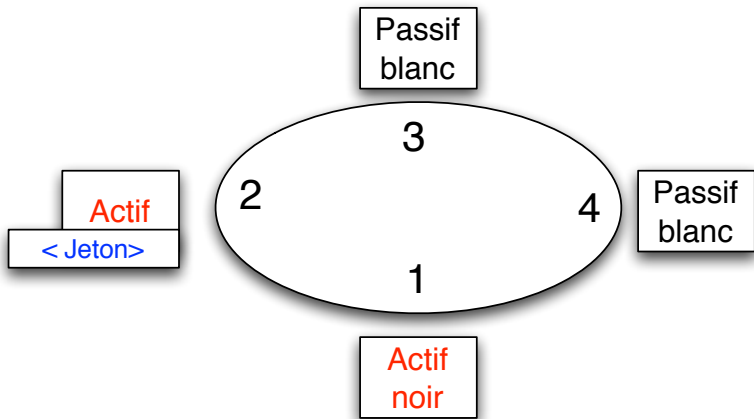
Principe :

Marqueur qui indique si un site a été actif depuis le dernier passage du jeton.

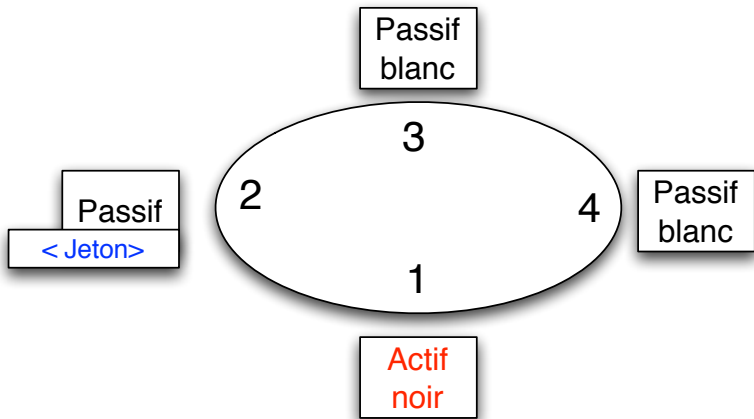
si le site i a été actif depuis le dernier passage du jeton,
alors $couleur_i = noir$
sinon $couleur_i = blanche$

Cette technique permet de savoir si **potentiellement** un message est en transit.

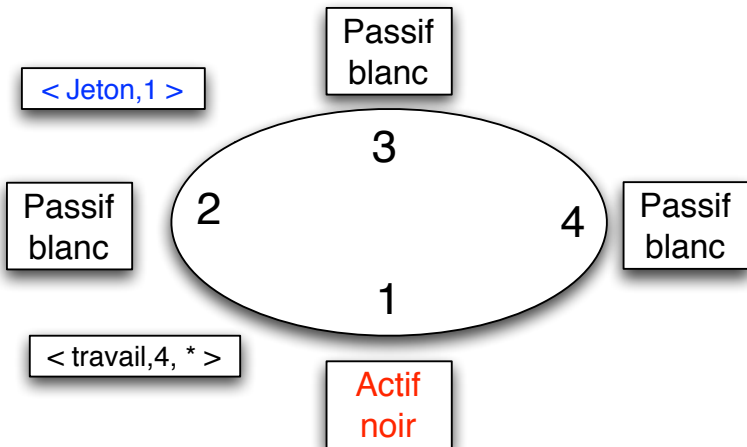
Exemple



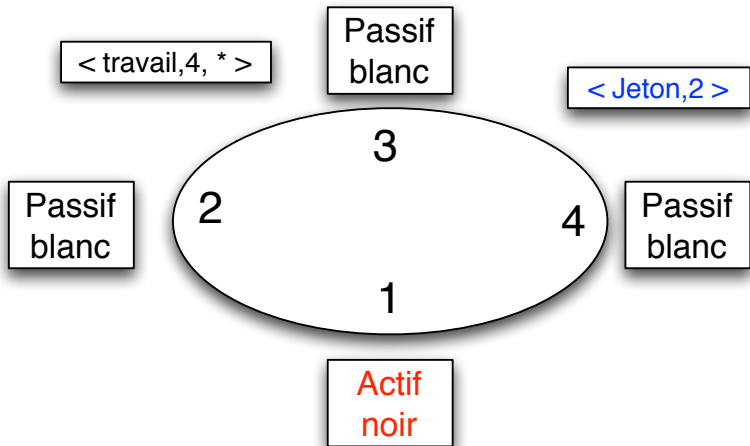
Exemple



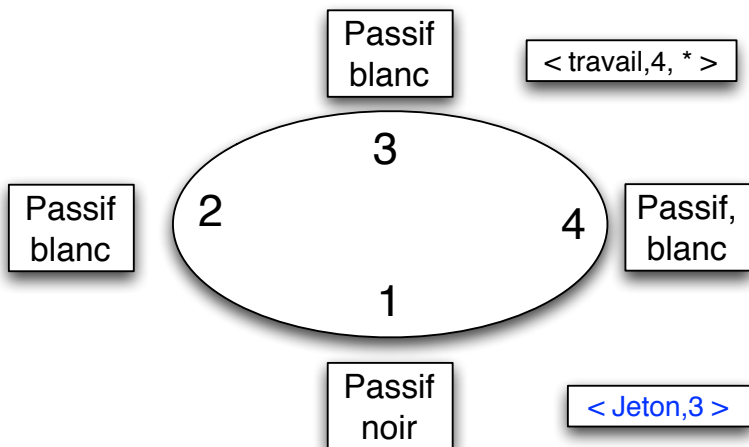
Exemple



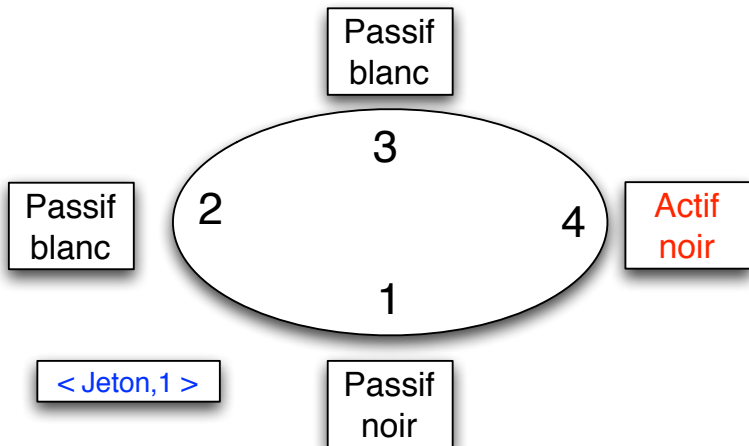
Exemple



Exemple



Exemple



Plus formellement : programme sur le site i

Messages de l'application

Plus formellement : programme sur le site i

Messages de l'application

Attente d'un message ou fin :

1. $etat_i \leftarrow passif$;

Plus formellement : programme sur le site i

Messages de l'application

Attente d'un message ou fin :

1. $etat_i \leftarrow passif$;

Lors de l'envoi d'une tâche :

1. $etat_i \leftarrow actif$;
2. $couleur_i \leftarrow noir$;

Plus formellement : programme sur le site i

Messages de l'application

Attente d'un message ou fin :

1. $etat_i \leftarrow passif$;

Lors de l'envoi d'une tâche :

1. $etat_i \leftarrow actif$;
2. $couleur_i \leftarrow noir$;

Réception d'une tâche ou initialisation :

1. $etat_i \leftarrow actif$;
2. $couleur_i \leftarrow noir$;
3. Faire le calcul

Plus formellement : programme sur le site i

Messages de l'application Circulation du jeton

Attente d'un message ou fin :

1. $etat_i \leftarrow passif$;

Lors de l'envoi d'une tâche :

1. $etat_i \leftarrow actif$;
2. $couleur_i \leftarrow noir$;

Réception d'une tâche ou
initialisation :

1. $etat_i \leftarrow actif$;
2. $couleur_i \leftarrow noir$;
3. Faire le calcul

Plus formellement : programme sur le site i

Messages de l'application Circulation du jeton

$(jeton_present_i = vrai) \wedge (etat_i = passif) \rightarrow$

Attente d'un message ou fin :

1. $etat_i \leftarrow passif$;

Lors de l'envoi d'une tâche :

1. $etat_i \leftarrow actif$;
2. $couleur_i \leftarrow noir$;

Réception d'une tâche ou initialisation :

1. $etat_i \leftarrow actif$;
2. $couleur_i \leftarrow noir$;
3. Faire le calcul

1. Si $(couleur_i == blanc)$ alors
envoyer $\langle JETON, nb_i + 1 \rangle$ au $suivant_i$
2. sinon envoyer $\langle JETON, 1 \rangle$ au $suivant_i$
3. $jeton_present_i \leftarrow faux$;
4. $couleur_i = blanc$;

Plus formellement : programme sur le site i

Messages de l'application Circulation du jeton

Attente d'un message ou fin :

1. $etat_i \leftarrow passif$;

Lors de l'envoi d'une tâche :

1. $etat_i \leftarrow actif$;
2. $couleur_i \leftarrow noir$;

Réception d'une tâche ou initialisation :

1. $etat_i \leftarrow actif$;
2. $couleur_i \leftarrow noir$;
3. Faire le calcul

$(jeton_present_i = vrai) \wedge (etat_i = passif) \rightarrow$

1. Si $(couleur_i == blanc)$ alors
envoyer $\langle JETON, nb_i + 1 \rangle$ au
suivant_i;
2. sinon envoyer $\langle JETON, 1 \rangle$ au *suivant_i*;
3. $jeton_present_i \leftarrow faux$;
4. $couleur_i = blanc$;

Réception de $\langle JETON, num \rangle$

1. $jeton_present_i \leftarrow vrai$;
2. $nb_i = num$
3. Si $(nb_i == n) \wedge (couleur_i = blanc)$
alors terminaison détectée

Plus précisément

Terminaison

Terminaison sur un anneau

Terminaison sur un graphe quelconque

Technique de vagues (Dijkstra-Scholten)

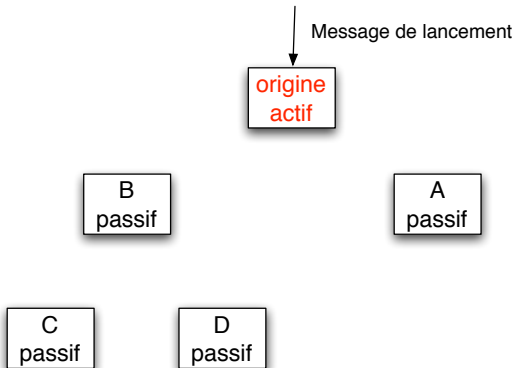
Principe

- ▶ le calcul est lancé à partir d'un nœud (origine)

Technique de vagues (Dijkstra-Scholten)

Principe

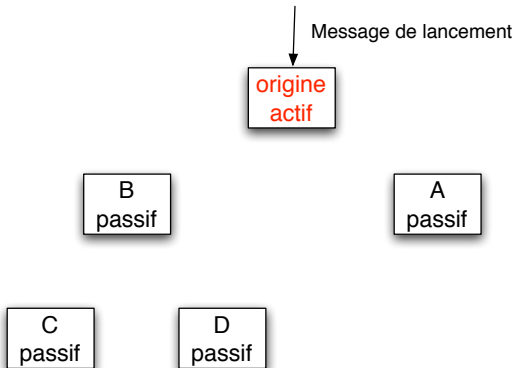
- ▶ le calcul est lancé à partir d'un nœud (origine)



Technique de vagues (Dijkstra-Scholten)

Principe

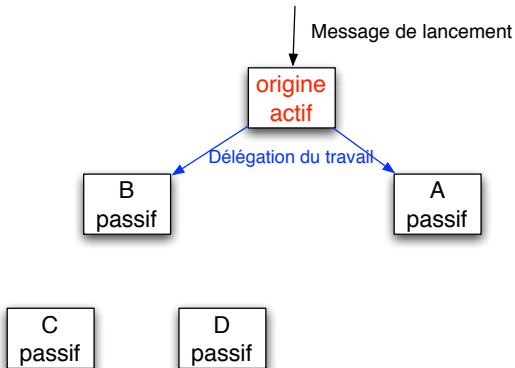
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

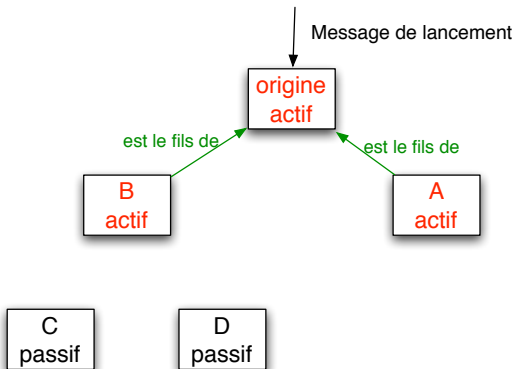
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

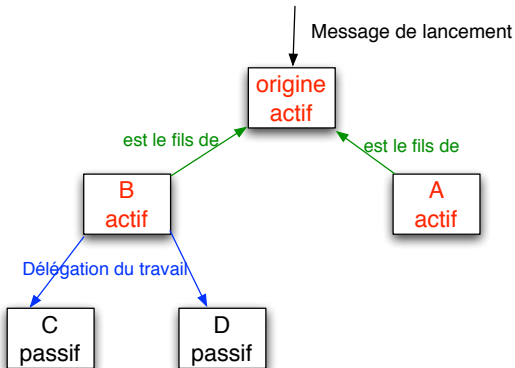
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

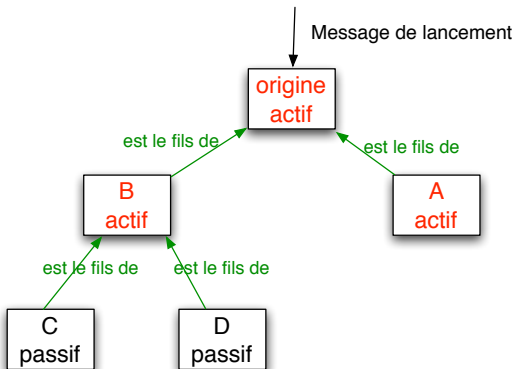
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

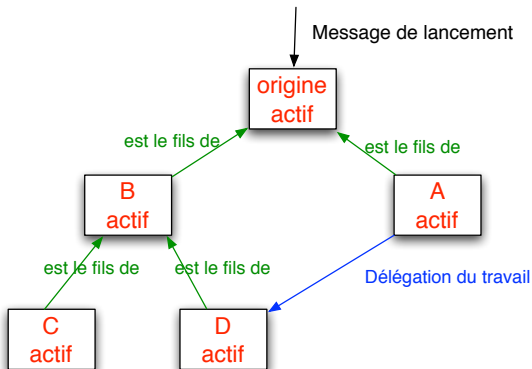
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

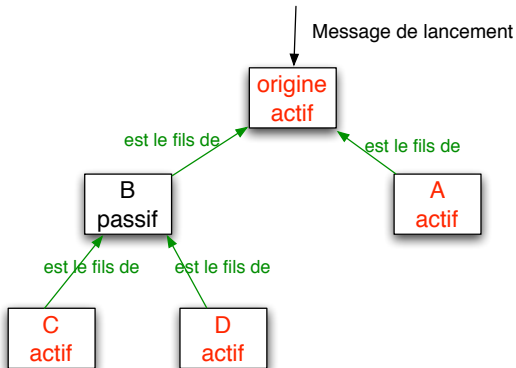
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

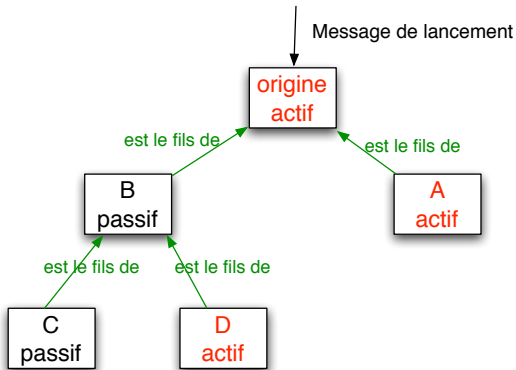
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

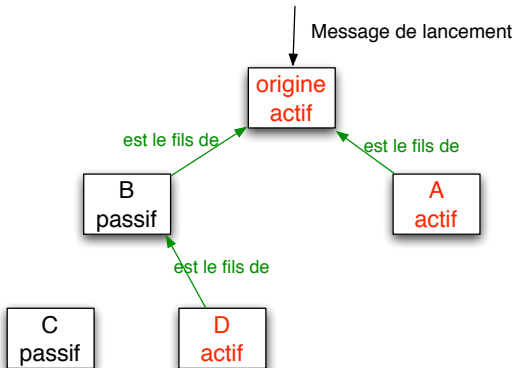
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

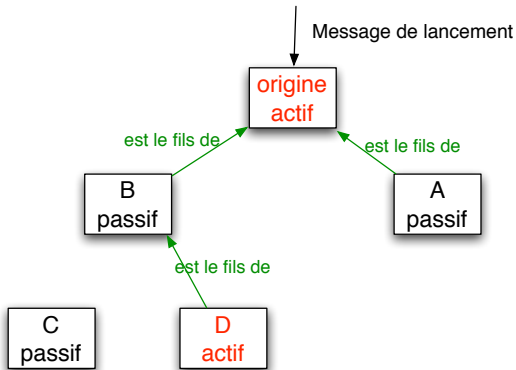
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

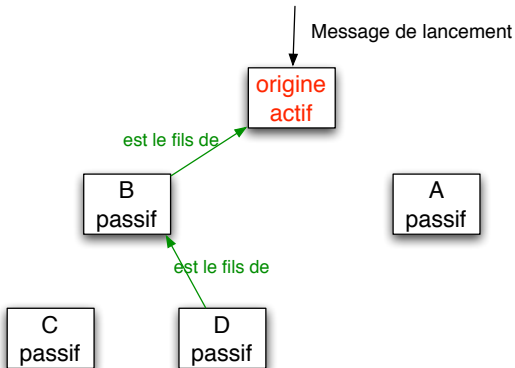
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

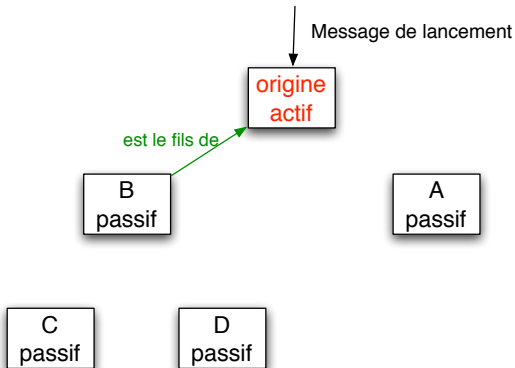
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

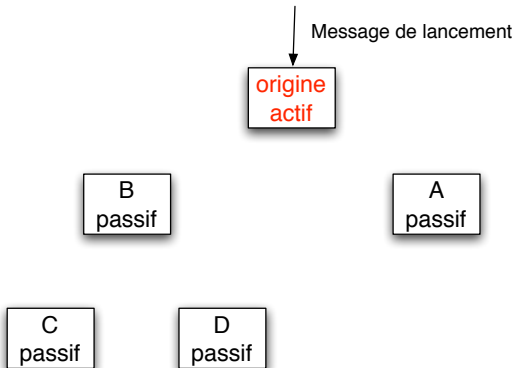
- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Technique de vagues (Dijkstra-Scholten)

Principe

- ▶ le calcul est lancé à partir d'un nœud (origine)
- ▶ A partir de ce nœud, est construit un arbre couvrant en fonction du travail.



Plus formellement : programme sur le site i

Réception d'un mes. de lancement $\rightarrow$

1. $etat_i \leftarrow actif$;
2. $pere_i \leftarrow i$;
3. lancer le calcul

fin des calculs $\rightarrow$

1. $etat_i \leftarrow passif$;

Plus formellement : programme sur le site i

Réception d'un mes. de lancement $\rightarrow$

1. $etat_i \leftarrow actif$;
2. $pere_i \leftarrow i$;
3. lancer le calcul

fin des calculs $\rightarrow$

1. $etat_i \leftarrow passif$;

Lors de l'envoi d'un travail à ℓ

1. $nb_fils_i = nb_fils_i + 1$;
2. envoyer $\langle TRAVAIL, * \rangle$ à ℓ

Plus formellement : programme sur le site i

Réception d'un mes. de lancement $\rightarrow$

1. $etat_i \leftarrow actif$;
2. $pere_i \leftarrow i$;
3. **lancer le calcul**

Lors de l'envoi d'un travail à ℓ

1. $nb_fils_i = nb_fils_i + 1$;
2. envoyer $\langle TRAVAIL, * \rangle$ à ℓ

fin des calculs $\rightarrow$

1. $etat_i \leftarrow passif$;

Réception d'un message $\langle TRAVAIL, * \rangle$ venant
 ℓ

1. Si ($etat_i = passif$) alors
 - $etat_i \leftarrow actif$;
 - 1.1 $pere_i \leftarrow \ell$;
2. sinon envoyer $\langle Pas - ton - fils \rangle$ à ℓ
3. **lancer le calcul**

Plus formellement : programme sur le site i

Réception d'un mes. de lancement $\rightarrow$

1. $etat_i \leftarrow actif$;
2. $pere_i \leftarrow i$;
3. **lancer le calcul**

Lors de l'envoi d'un travail à ℓ

1. $nb_fils_i = nb_fils_i + 1$;
2. envoyer $\langle TRAVAIL, * \rangle$ à ℓ

fin des calculs $\rightarrow$

1. $etat_i \leftarrow passif$;

Réception d'un message $\langle TRAVAIL, * \rangle$ venant ℓ

1. Si ($etat_i = passif$) alors
 $etat_i \leftarrow actif$;
 1.1 $pere_i \leftarrow \ell$;
2. sinon envoyer $\langle Pas - ton - fils \rangle$ à ℓ
3. **lancer le calcul**

Lors de la réception de $\langle Pas - ton - fils \rangle$

1. $nb_fils_i = nb_fils_i - 1$;

Plus formellement : programme sur le site i

Réception d'un mes. de lancement $\rightarrow$

1. $etat_i \leftarrow actif$;
2. $pere_i \leftarrow i$;
3. **lancer le calcul**

Lors de l'envoi d'un travail à ℓ

1. $nb_fils_i = nb_fils_i + 1$;
2. envoyer $\langle TRAVAIL, * \rangle$ à ℓ

$(etat_i = passif) \wedge (nb_fils_i = 0) \rightarrow$

1. Si $(pere_i == i)$ alors

terminaison détectée

2. sinon

envoyer à $pere_i$ le message
 $\langle Pas - ton - fils \rangle$

fin des calculs $\rightarrow$

1. $etat_i \leftarrow passif$;

Réception d'un message $\langle TRAVAIL, * \rangle$ venant
 ℓ

1. Si $(etat_i = passif)$ alors

$etat_i \leftarrow actif$;

1.1 $pere_i \leftarrow \ell$;

2. sinon envoyer $\langle Pas - ton - fils \rangle$ à ℓ

3. **lancer le calcul**

Lors de la réception de $\langle Pas - ton - fils \rangle$

1. $nb_fils_i = nb_fils_i - 1$;