

Algorithmique répartie



1 Introduction

2 Systèmes répartis

- Modélisation des réseaux
- Vocabulaires
- Noeuds
- Liens de communications

3 Notion de temps

- Réseaux synchrones
- Réseaux asynchrones

4 Algorithmes répartis

- Variables locales
- Pseudo-code
- Réveil spontané
- Un exemple

5 Complexité(s)

- Complexité séquentielle
- Nombre de messages
- Tailles des messages
- Complexité temporelle :
- Complexité spatiale :

Algorithmique Séquentielle

- Une tâche / un calcul



Algorithmique Séquentielle

- Une tâche / un calcul
- Série d'opérations élémentaires



Algorithmique Séquentielle

- Une tâche / un calcul
- Série d'opérations élémentaires
- Les unes après les autres



Algorithmique Séquentielle

- Une tâche / un calcul
- Série d'opérations élémentaires
- Les unes après les autres
- Sur une même machine





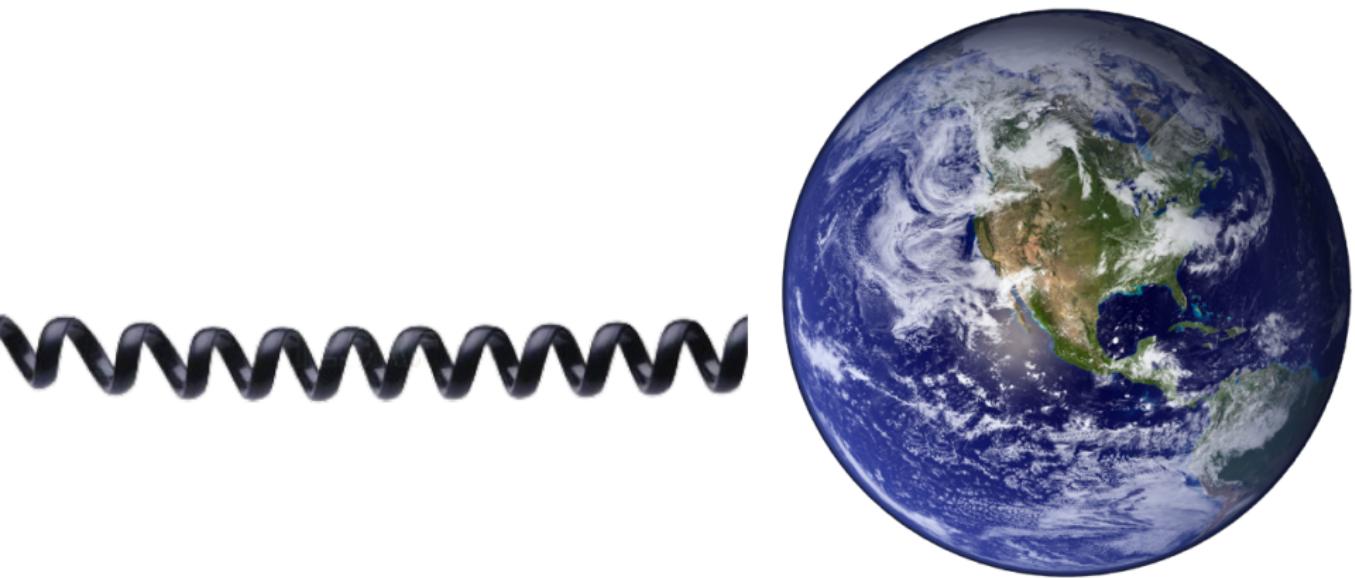
© Ron Leishman * www.ClipartOf.com/433582



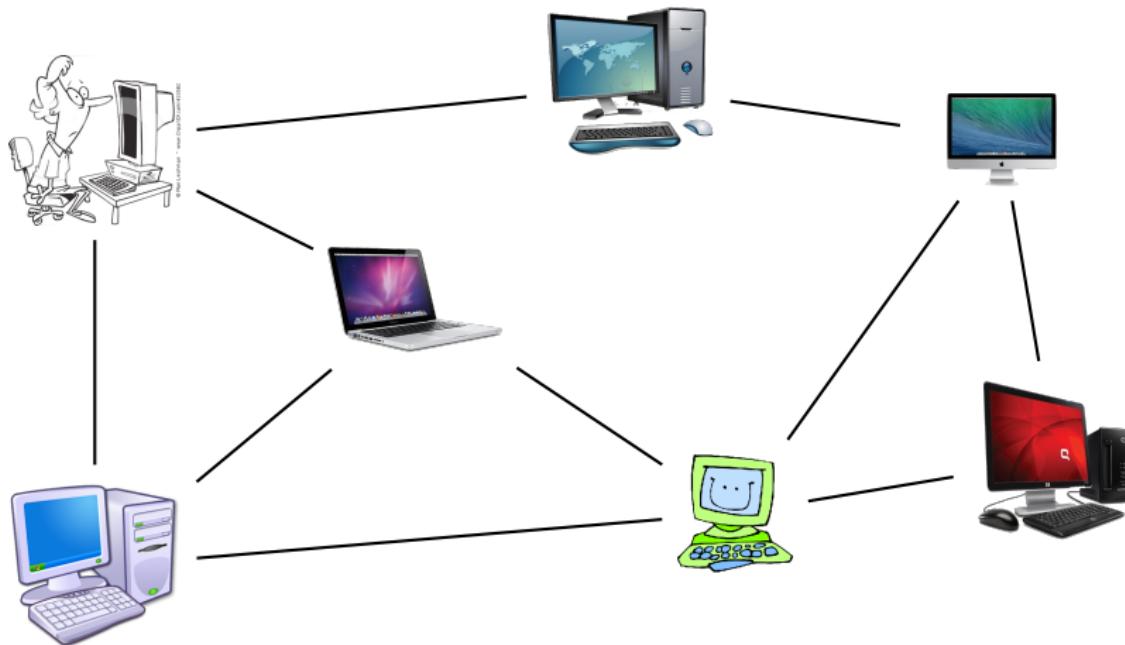
© Ron Leishman * www.ClipartOf.com/433582



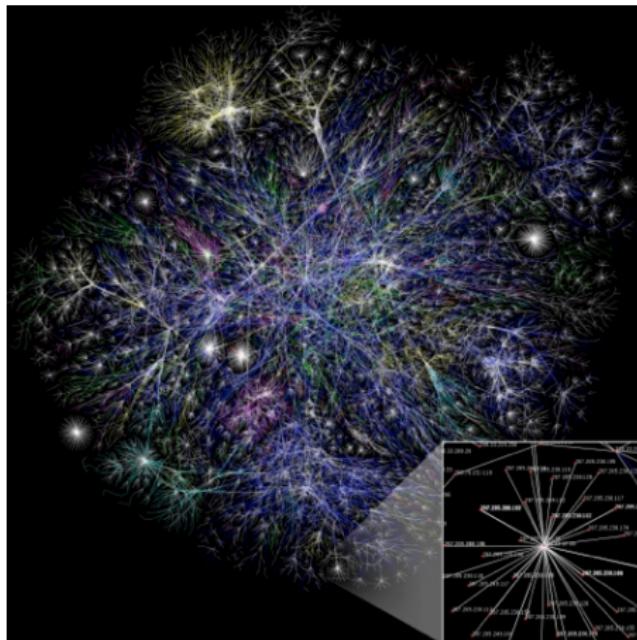




Réseaux

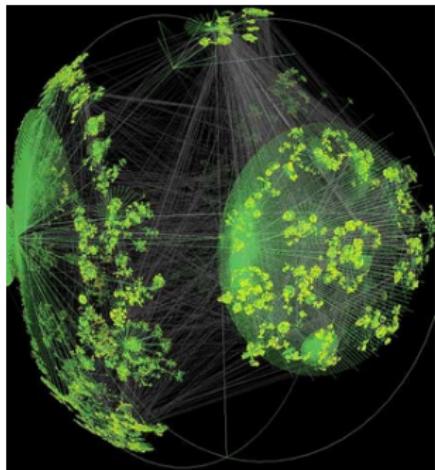


Internet



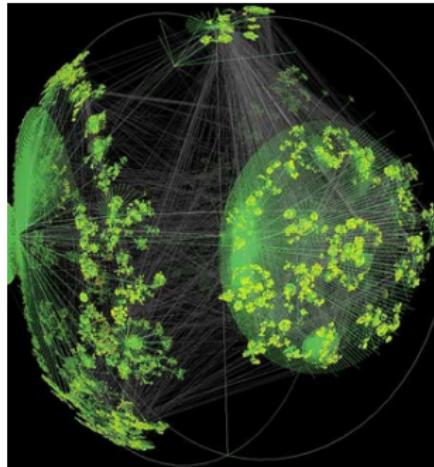
Représentation d'une partie du réseau Internet en 2005

Un réseaux c'est quoi de nos jours ?



- Très grand réseaux :
 - Explosion combinatoire du nombre de machines connecté

Un réseaux c'est quoi de nos jours ?



- Très grand réseaux :
 - Explosion combinatoire du nombre de machines connecté
- Non homogène
 - Explosion du type de machines connectées
 - Filaire ou non
 - Non statique

Non homogènes





Un réseaux pour quoi faire

- Echanger de l'information

- Emails
- fichiers textes
- fichiers son
- fichiers images
- ...

Un réseaux pour quoi faire

- Echanger de l'information

- Emails
- fichiers textes
- fichiers son
- fichiers images
- ...

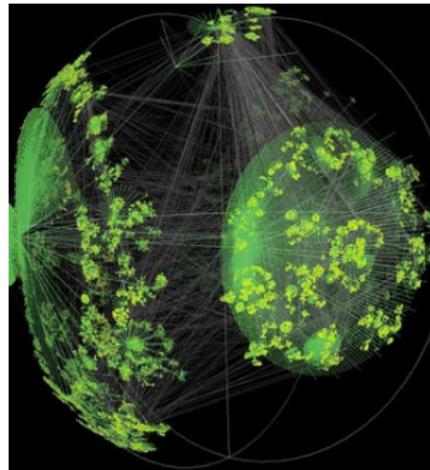
- Partager des ressources

- mémoire
- base de données
- matériels

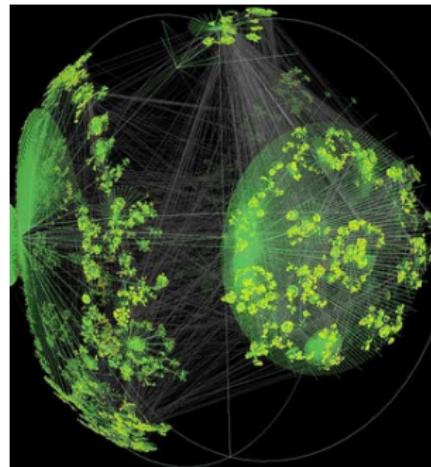
Un réseaux pour quoi faire

- Echanger de l'information
 - Emails
 - fichiers textes
 - fichiers son
 - fichiers images
 - ...
- Partager des ressources
 - mémoire
 - base de données
 - matériels
- Tout cela s'organise...

Pas de big boss



Pas de big boss





Question

Comment réaliser une tâche commune sur un réseaux à grande échelle ?

Tâche commune

Tâche : Répondre à la question :
Qui est né en Mars ?

Tâche commune

Tâche : Répondre à la question :
Qui est né en Mars ?

Approche centralisée :

Tâche commune



Tâche : Répondre à la question :
Qui est né en Mars ?

Approche centralisée :

- L'enseignant demande à tout le monde

Tâche commune

Tâche : Répondre à la question :
Qui est né en Mars ?

Tâche commune

Tâche : Répondre à la question :
Qui est né en Mars ?

Approche distribué :

Tâche commune



Tâche : Répondre à la question :
Qui est né en Mars ?

Approche distribué :

- Chaque élève demande à ses voisins
- qui demandent à leurs voisins
- ...

Systèmes et algorithmiques répartis



Systèmes répartis

C'est un réseaux qui est constitués
d'entités de calcul

- Mémoire
- puissance de calcul
- ...
- Homogène ou non



Systèmes répartis

C'est un réseau qui est constitué
de liens de communications

- filaire,
- wifi,
- radio
-



Systèmes répartis

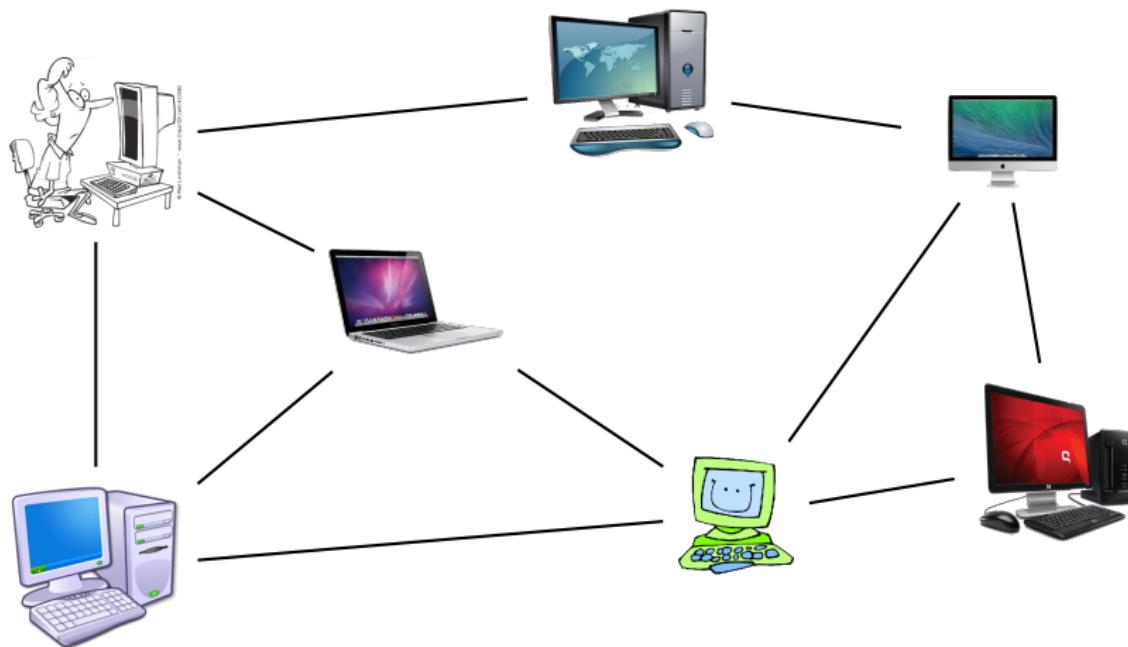
C'est un réseau qui est constitué
de liens de communications

- filaire,
- wifi,
- radio
-

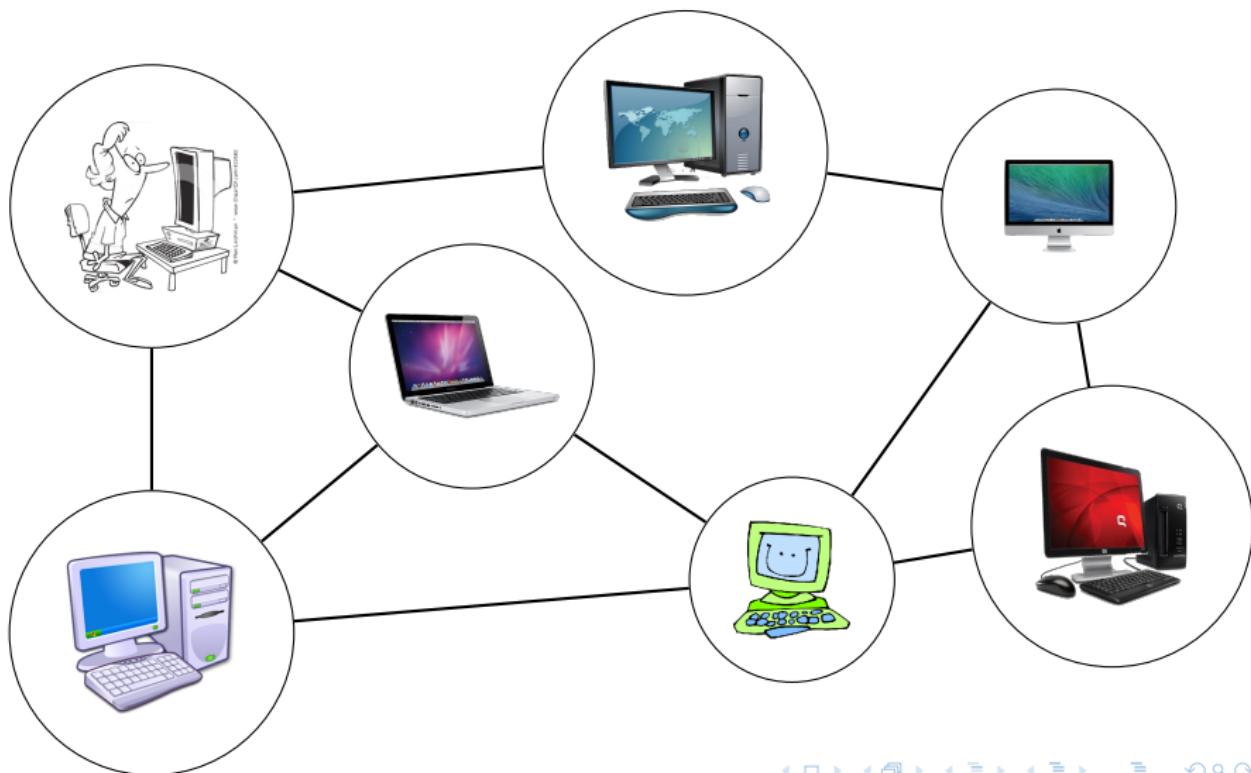
Les liens de communications

- Ils sont utilisés pour échanger de l'information.
- L'information est transportée par des **messages**.

Modélisation des réseaux



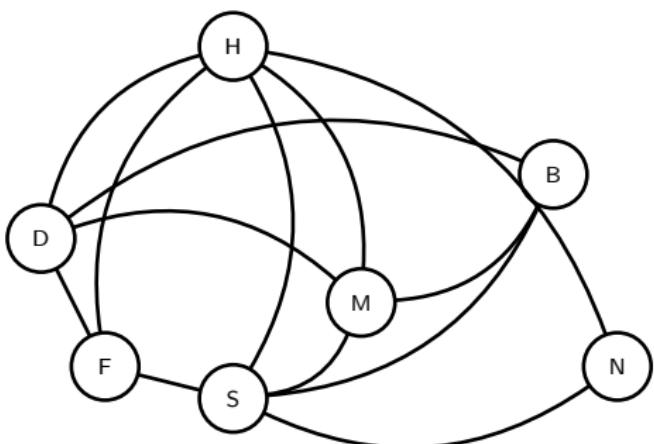
Modélisation des réseaux



Modélisation des réseaux

On modélise les réseaux par des graphes simples.

Vocabulaires : entités de calculs

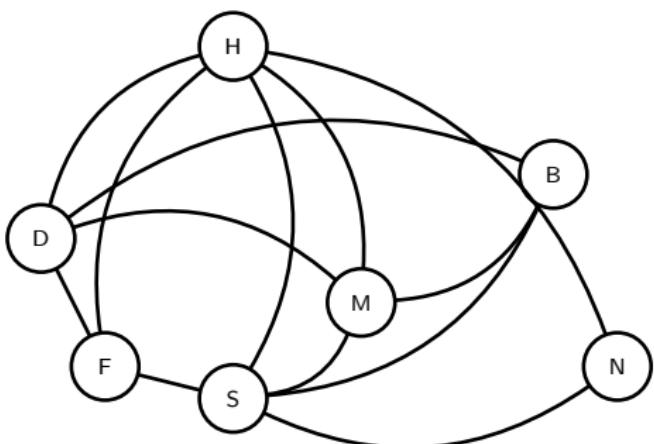


Les unités de calculs peuvent être noté de différentes façon

- Noeuds

Vocabulaires

Vocabulaires : entités de calculs

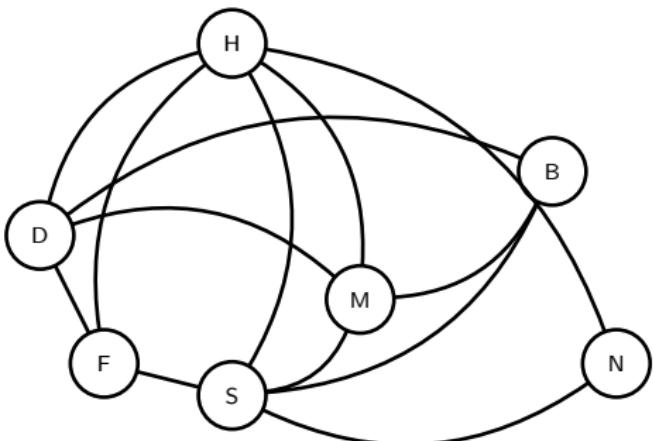


Les unités de calculs peuvent être noté de différentes façon

- Noeuds
- Sites

Vocabulaires

Vocabulaires : entités de calculs

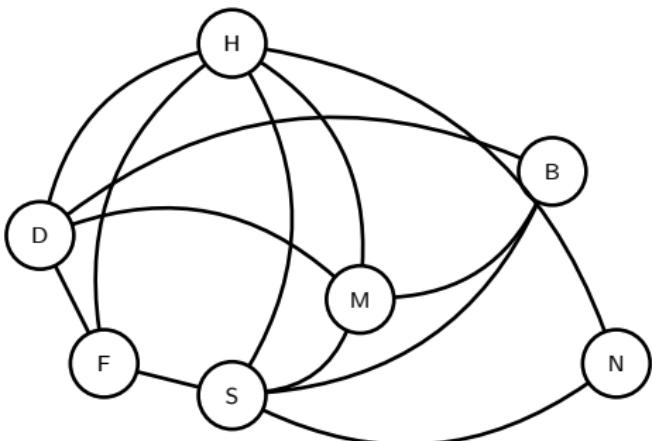


Les unités de calculs peuvent être noté de différentes façon

- Noeuds
- Sites
- Processeurs

Vocabulaires

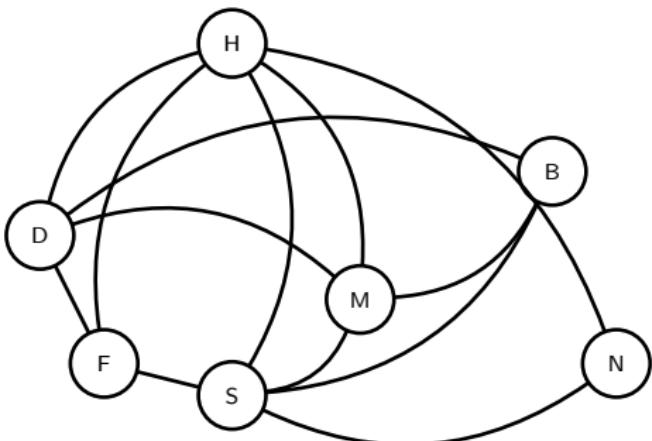
Vocabulaires : entités de calculs



Les unités de calculs peuvent être noté de différentes façon

- Noeuds
- Sites
- Processeurs
- Processus

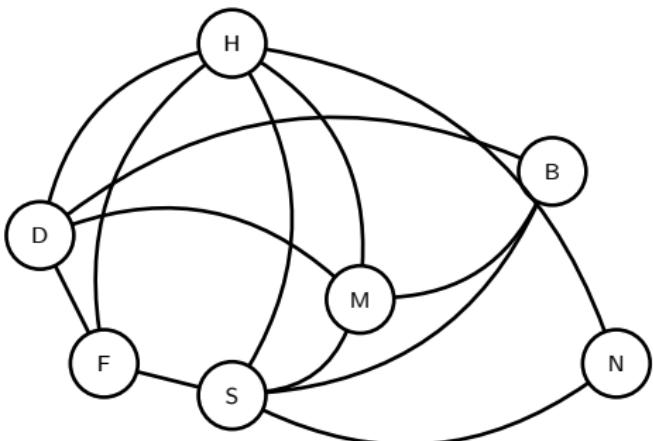
Vocabulaires : communications



Les canaux de communications peuvent être noté de différentes façon

- Canaux de communications

Vocabulaires : communications

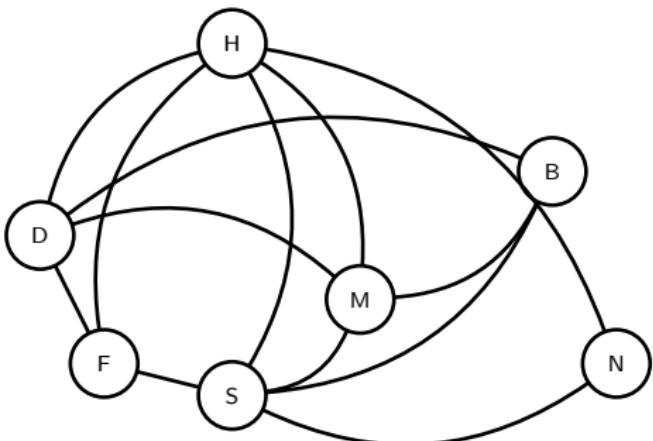


Les canaux de communications peuvent être noté de différentes façon

- Canaux de communications
- Liens

Vocabulaires

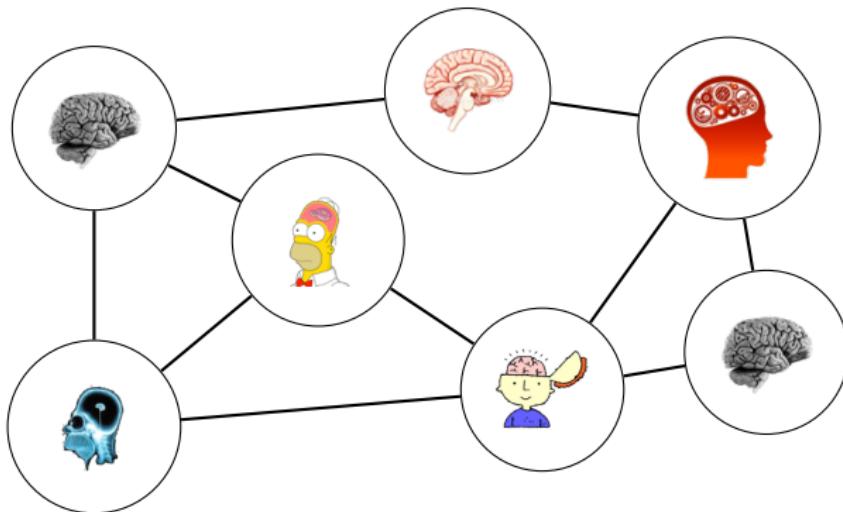
Vocabulaires : communications



Les canaux de communications peuvent être noté de différentes façon

- Canaux de communications
- Liens
- Arêtes

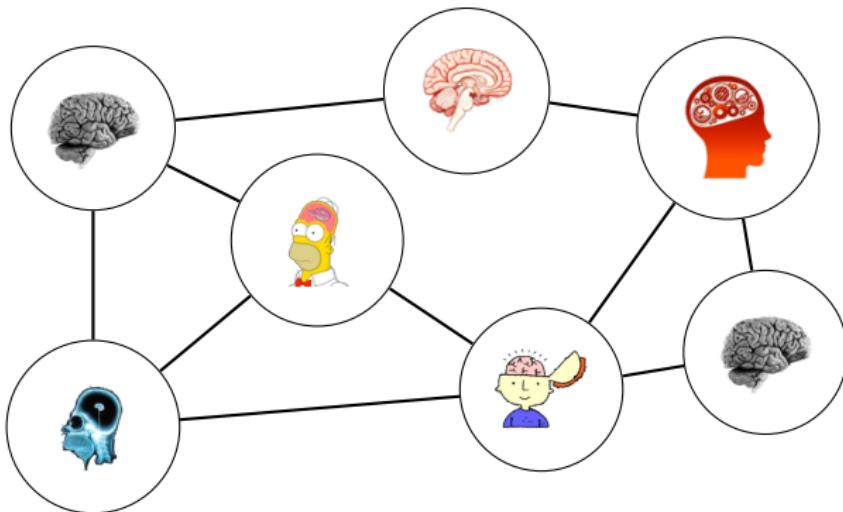
Mémoire des noeuds



Chaque noeuds a sa propre mémoire



Mémoire des noeuds

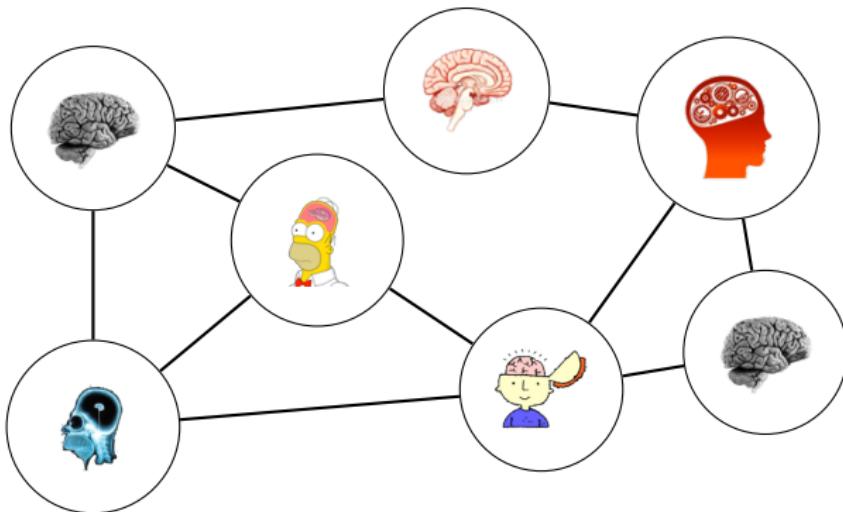


Chaque noeuds a sa propre mémoire

- chaque mémoire peut être de taille différente.



Mémoire des noeuds

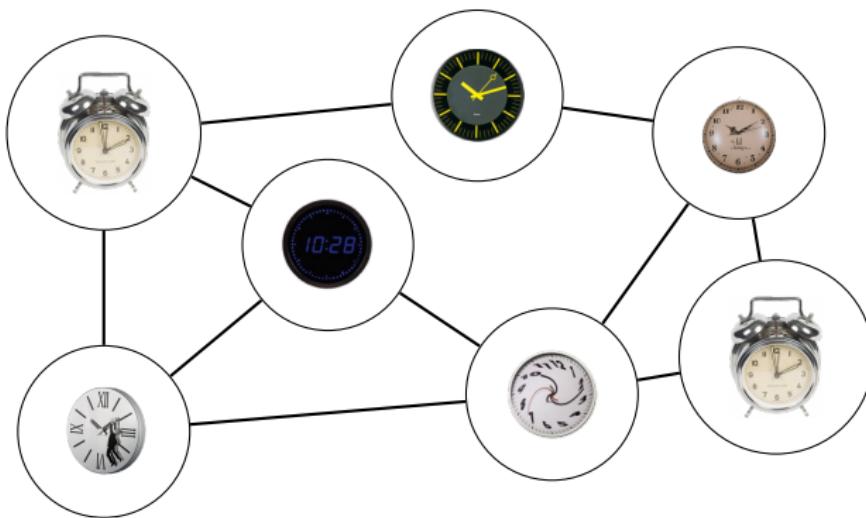


Chaque noeuds a sa propre mémoire

- chaque mémoire peut être de taille différente.
- Il n'y a pas de mémoire commune partagée.



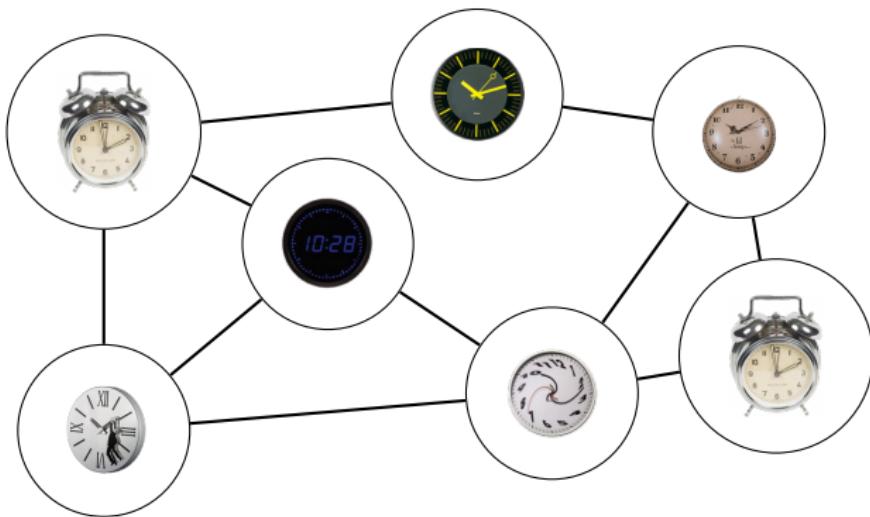
Horloge des noeuds



Chaque noeuds a sa propre horloge



Horloge des noeuds

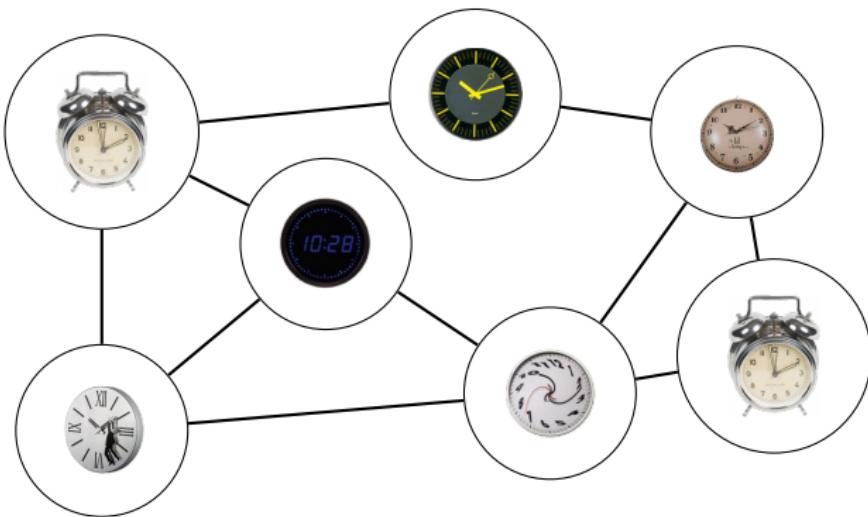


Chaque noeuds a sa propre horloge

- Les temps des horloges peuvent être différents .



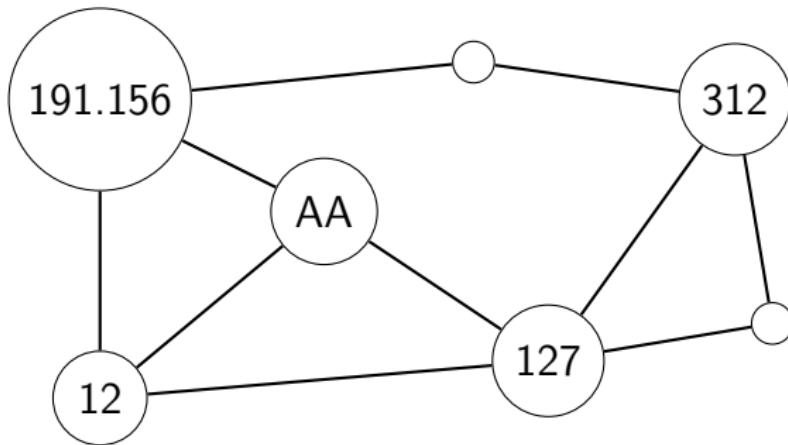
Horloge des noeuds



Chaque noeuds a sa propre horloge

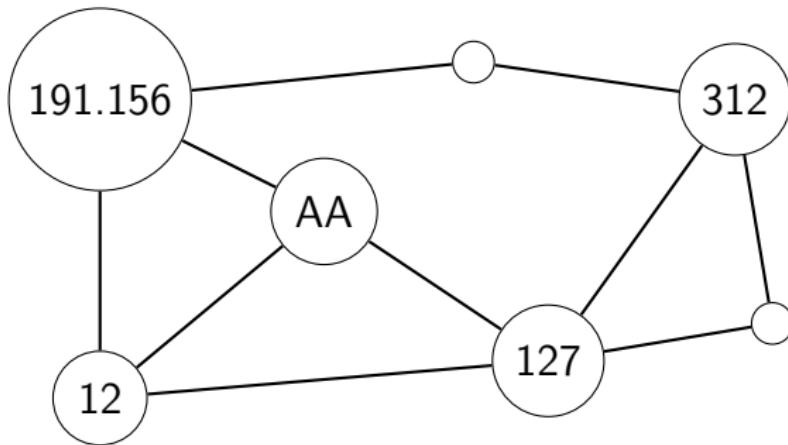
- Les temps des horloges peuvent être différents .
- Il n'y a pas d'horloge commune partagée.

Identifiants des noeuds



Identifiants

Identifiants des noeuds



Identifiants

- Chaque noeud possède ou pas un identifiant unique.

Noeud

Un noeud est une puissance de calcul qui

- Exécute un ensemble d'instructions

Noeud

Un noeud est une puissance de calcul qui

- Exécute un ensemble d'instructions
- Réagit à un événement local :

Noeud

Un noeud est une puissance de calcul qui

- Exécute un ensemble d'instructions
- Réagit à un événement local :
 - Un calcul interne

Noeud

Un noeud est une puissance de calcul qui

- Exécute un ensemble d'instructions
- Réagit à un événement local :
 - Un calcul interne
 - la réception d'un message

Noeud

Un noeud est une puissance de calcul qui

- Exécute un ensemble d'instructions
- Réagit à un événement local :
 - Un calcul interne
 - la réception d'un message
 - l'envoie d'un message

Noeud

Un noeud est une puissance de calcul qui a

- Une mémoire locale

Noeud

Un noeud est une puissance de calcul qui a

- Une mémoire locale
- Un état local

Noeud

Un noeud est une puissance de calcul qui a

- Une mémoire locale
- Un état local
- Un ensemble de données et de variables locales

Noeud

Un noeud est une puissance de calcul qui a

- Une mémoire locale
- Un état local
- Un ensemble de données et de variables locales
- Possède ou pas d'identifiant

Noeud

Un noeud est une puissance de calcul qui a

- Une mémoire locale
- Un état local
- Un ensemble de données et de variables locales
- Possède ou pas d'identifiant
- Possède peu ou pas de connaissance

Connaissance locale

Connaissance sur les liens de communications

- ① Le noeud p connaît qu'il a d liens de communications

Connaissance locale

Connaissance sur les liens de communications

- ① Le noeud p connaît qu'il a d liens de communications
- ② Le noeud p peut numérotter ses liens (numéro de ports)

Connaissance locale

Connaissance sur les liens de communications

- ① Le noeud p connaît qu'il a d liens de communications
- ② Le noeud p peut numéroté ses liens (numéro de ports)
- ③ Le noeud p connaît les identifiants de ses voisins

Connaissance globale

Connaissance

- Le noeud a aucune connaissance du réseau : le plus réaliste.

Connaissance globale

Connaissance

- Le noeud a aucune connaissance du réseau : le plus réaliste.
- Le noeud a la connaissance de la taille du réseau (nombre de noeuds) : peu réaliste (grande taille, dynamique)

Connaissance globale

Connaissance

- Le noeud a aucune connaissance du réseau : le plus réaliste.
- Le noeud a la connaissance de la taille du réseau (nombre de noeuds) : peu réaliste (grande taille, dynamique)
- Le noeud a la connaissance du diamètre du réseaux

Connaissance globale

Connaissance

- Le noeud a aucune connaissance du réseau : le plus réaliste.
- Le noeud a la connaissance de la taille du réseau (nombre de noeuds) : peu réaliste (grande taille, dynamique)
- Le noeud a la connaissance du diamètre du réseaux
- Le noeud a la connaissance de la topologie du réseaux (grille, anneaux,...)

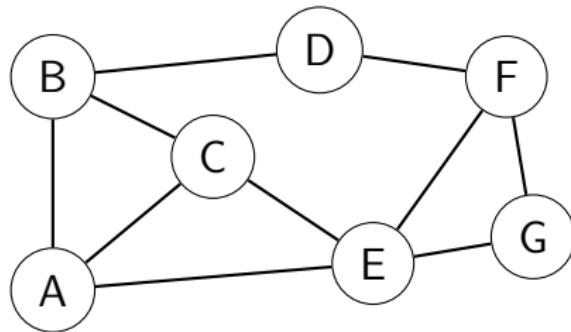
Connaissance globale

Connaissance

- Le noeud a aucune connaissance du réseau : le plus réaliste.
- Le noeud a la connaissance de la taille du réseau (nombre de noeuds) : peu réaliste (grande taille, dynamique)
- Le noeud a la connaissance du diamètre du réseaux
- Le noeud a la connaissance de la topologie du réseaux (grille, anneaux,...)
- ...

Liens de communications

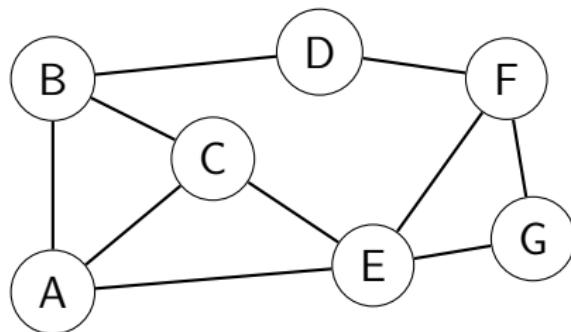
Liens de communications



communication

- Chaque noeuds envoie ou reçoit des messages à travers des liens de communications

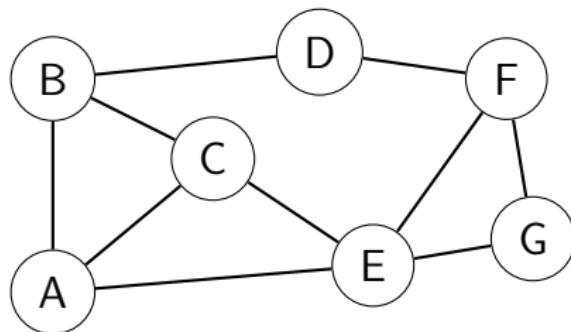
Liens de communications



communication

- Ces liens peuvent être unidirectionnel

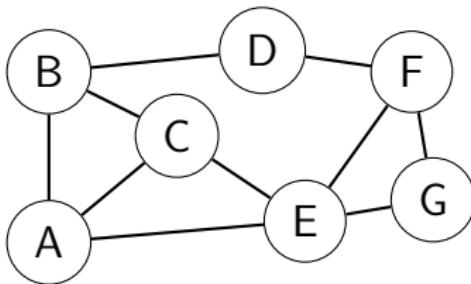
Liens de communications



communication

- Ces liens peuvent être unidirectionnel
- Ces liens peuvent être bidirectionnel

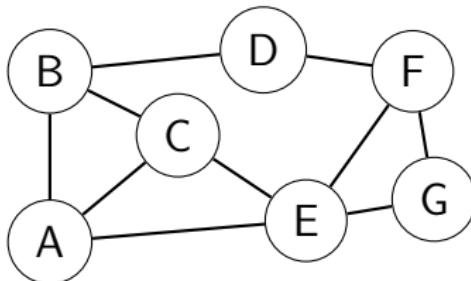
Liens de communications



Communications

- Le transit des messages à l'intérieur des liens peuvent être FIFO ou pas

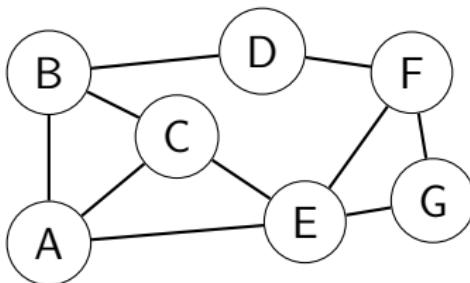
Liens de communications



Communications FIFO

- Soit deux messages m_1 et m_2 envoyer par le noeud A
- Avec m_1 envoyer avant le message m_2
 - m_1 arrivera en B avant m_2

Liens de communications

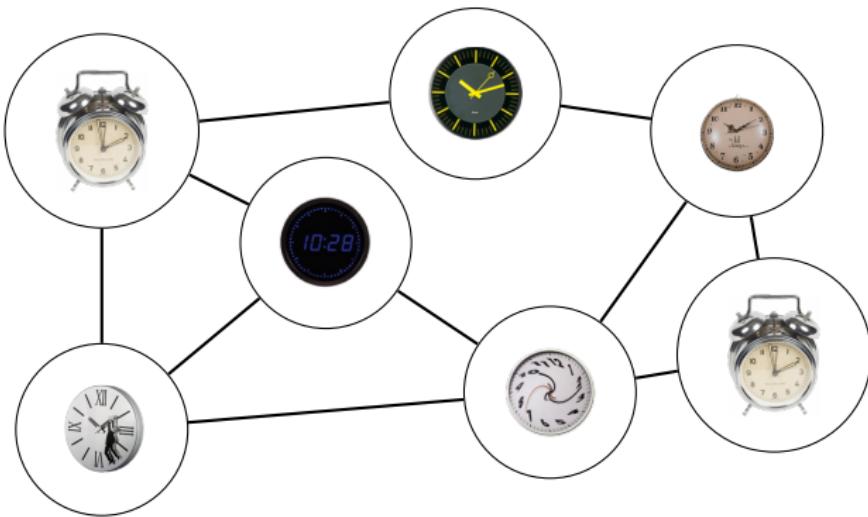


Communications non FIFO

- Soit deux messages m_1 et m_2 envoyer par le noeud A
- Avec m_1 envoyer avant le message m_2
 - m_2 peut arriver avant ou après m_1 .

Réseaux synchrones

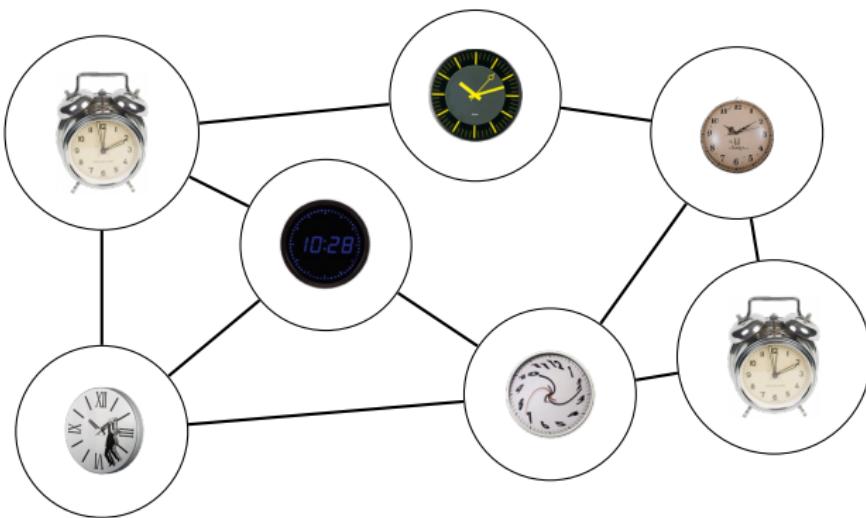
Réseaux synchrones



Définition 1



Réseaux synchrones



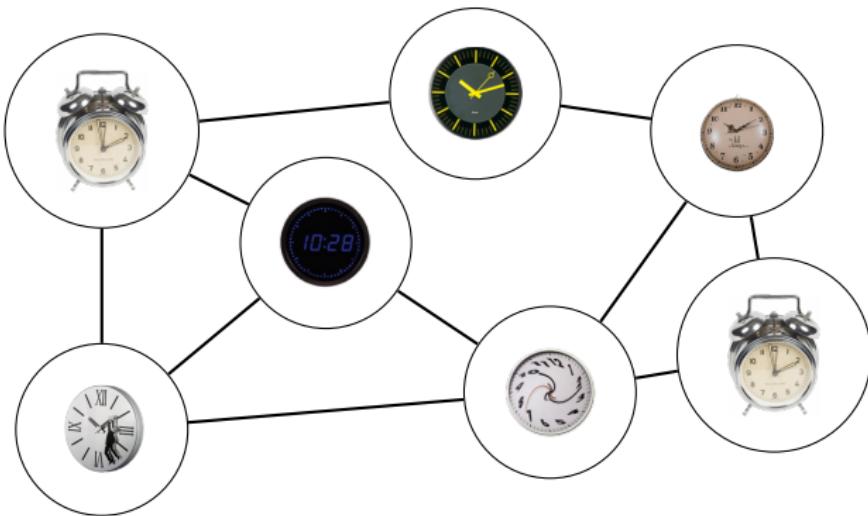
Définition 1

- Les noeuds ont des temps de calculs identiques.



Réseaux synchrones

Réseaux synchrones



Définition 1

- Les noeuds ont des temps de calculs identiques.
- Le temps de circulation des messages est identique.



Réseaux synchrones

Définition 2

Réseaux synchrones

Définition 2

- Les noeuds calculent par rondes synchrones. Dans une ronde, chaque noeud exécute les étapes suivantes :
 - ① Effectuer des calculs locaux.
 - ② Envoyer des messages à ces voisins.

Réseaux synchrones

Définition 2

- Les noeuds calculent par rondes synchrones. Dans une ronde, chaque noeud exécute les étapes suivantes :
 - ① Effectuer des calculs locaux.
 - ② Envoyer des messages à ces voisins.
 - ③ Recevoir des messages de ces voisins.

Réseaux asynchrones

Définition

Réseaux asynchrones

Définition

- Les noeuds ont des temps de calculs différents.

Réseaux asynchrones

Définition

- Les noeuds ont des temps de calculs différents.
- Le temps de circulation des messages est non borné mais fini.



Algorithmes répartis

Définition

- Tous les noeuds ont le même algorithme séquentiel.
- Cet algorithme réagit aux réceptions de messages.
- Cet algorithme envoie des messages

Variables locales

- Tous les noeuds ont le même algorithme séquentiel.

Variables locales

- Tous les noeuds ont le même algorithme séquentiel.
 - Donc tous les noeuds ont des variables locales qui porteront le même nom.

Variables locales

- Tous les noeuds ont le même algorithme séquentiel.
 - Donc tous les noeuds ont des variables locales qui porteront le même nom.
- Vision extérieure globale :

Variables locales

- Tous les noeuds ont le même algorithme séquentiel.
 - Donc tous les noeuds ont des variables locales qui porteront le même nom.
- Vision extérieure globale :
 - Pour reconnaître de quelle variable on parle on ajoutera l'identifiant du processus à la variable.

Variables locales

- Tous les noeuds ont des variables locales qui porteront le même nom.

Variables locales

- Tous les noeuds ont des variables locales qui porteront le même nom.
- Exemple :

Variables locales

- Tous les noeuds ont des variables locales qui porteront le même nom.
- Exemple :
- d_i sera la variable distance du site ayant pour identifiant i .

Variables locales

- Tous les noeuds ont des variables locales qui porteront le même nom.
- Exemple :
- d_i sera la variable distance du site ayant pour identifiant i .
- d_j sera la variable distance du site ayant pour identifiant j .

Pseudo-code

- C'est un pseudo-code classique (séquentiel) avec des tests, des boucles...

Pseudo-code

- C'est un pseudo-code classique (séquentiel) avec des tests, des boucles...
- Le pseudo code se présente par **block**

Pseudo-code

- C'est un pseudo-code classique (séquentiel) avec des tests, des boucles...
- Le pseudo code se présente par **block**
- Un bloc d'**Initialisation** pour initier les variables locales.

Pseudo-code

- C'est un pseudo-code classique (séquentiel) avec des tests, des boucles...
- Le pseudo code se présente par **block**
- Un bloc d'**Initialisation** pour initier les variables locales.
- Un bloc **par réception de message** .

Pseudo-code

- C'est un pseudo-code classique (séquentiel) avec des tests, des boucles...
- Le pseudo code se présente par **block**
- Un bloc d'**Initialisation** pour initier les variables locales.
- Un bloc **par réception de message** .
 - Lors de la réception du message $< Message1 >$ envoyer par le noeud q :

Pseudo-code

- C'est un pseudo-code classique (séquentiel) avec des tests, des boucles...
- Le pseudo code se présente par **block**
- Un bloc d'**Initialisation** pour initier les variables locales.
- Un bloc **par réception de message** .
 - Lors de la réception du message $< Message1 >$ envoyer par le noeud q :
 - instructions....

Pseudo-code

- C'est un pseudo-code classique (séquentiel) avec des tests, des boucles...
- Le pseudo code se présente par **block**
- Un bloc d'**Initialisation** pour initier les variables locales.
- Un bloc **par réception de message** .
 - Lors de la réception du message $< Message1 >$ envoyer par le noeud q :
 - instructions....
- **Attention** : il y aura autant de blocs que de types de messages.

Qui commence ?

- Les processus qui veulent faire une tâche effectueront un **réveil spontané**.

Qui commence ?

- Les processus qui veulent faire une tâche effectueront un **réveil spontané**.
- Les autres processus se réveilleront à la réception d'un premier message

Algorithme réparti de diffusion

Initialisation noeud i

- $val_i = identifiant_i$

Algorithme réparti de diffusion

Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Algorithme réparti de diffusion

Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:

Algorithme réparti de diffusion

Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
 - $val_i := val_j$

Algorithme réparti de diffusion

Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
 - $val_i := val_j$
 - Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Algorithme réparti de diffusion

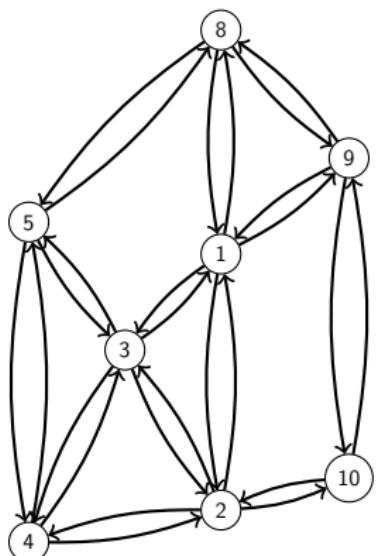
Initialisation

Quand un noeud non réveillé reçoit un message il fait le block d'initialisation avant de traiter le message.

Exemple synchrone

Dans l'exemple "pour aller plus vite", quand un noeud reçoit plusieurs messages il prend le maximum des valeurs reçues.

Exemple synchrone



Initialisation noeud i

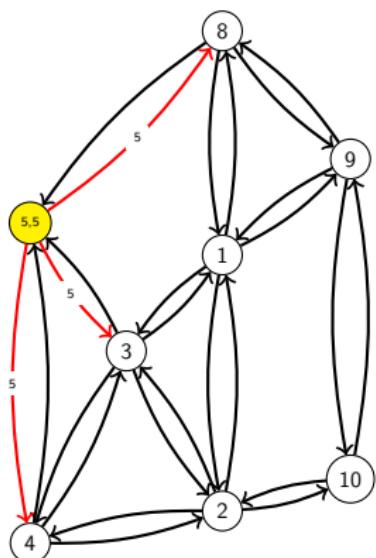
- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
- $val_i := val_j$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Un exemple

Exemple synchrone



Initialisation noeud i

- $val_i = identifiant_i$;
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

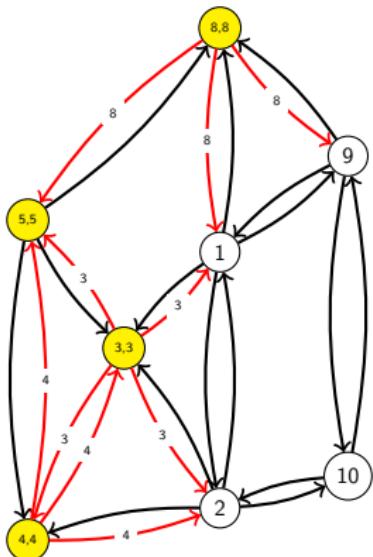
Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
 - $val_i := val_j$
 - Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Rounde 1 : Noeud actif 5

Un exemple

Exemple synchrone



Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

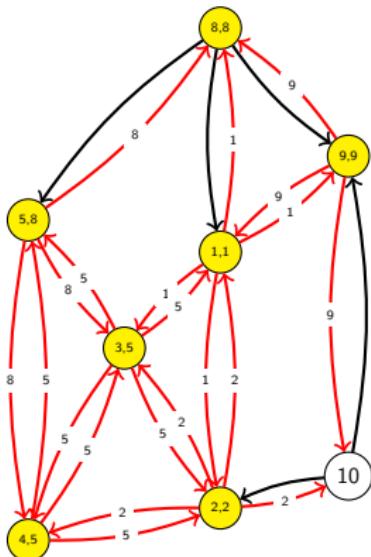
Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
 - $val_i := val_j$
 - Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Rounde 2 : Noeuds actifs
3,8,4.

Un exemple

Exemple synchrone



Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

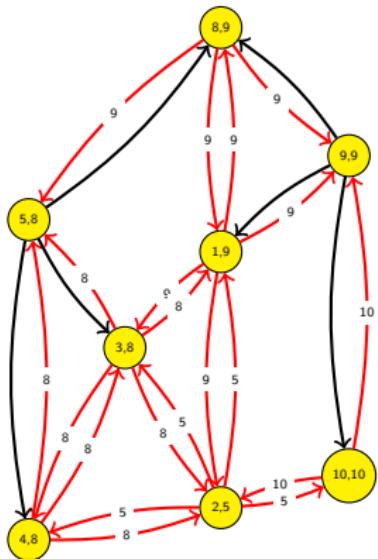
Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
 - $val_i := val_j$
 - Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Rounde 3 : Noeuds actifs 1
à 9.

Un exemple

Exemple synchrone



Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

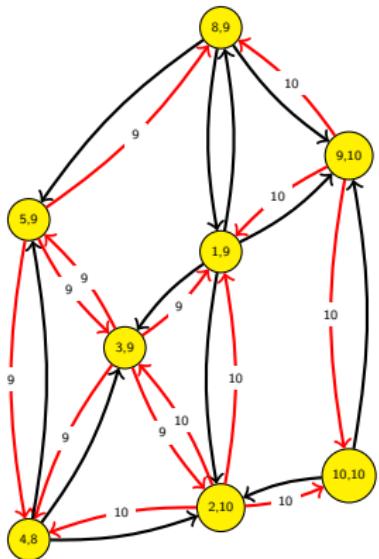
Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
 - $val_i := val_j$
 - Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Rounde 4

Un exemple

Exemple synchrone



Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

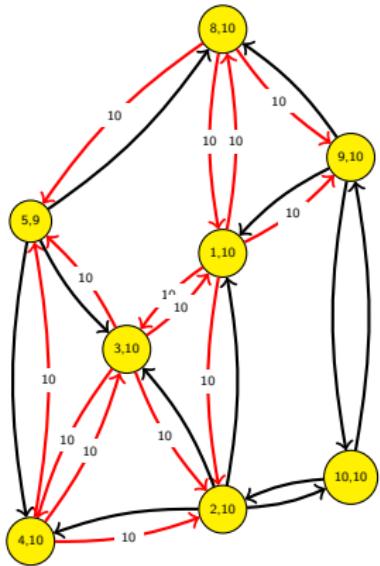
Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
 - $val_i := val_j$
 - Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Rounde 5

Un exemple

Exemple synchrone



Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

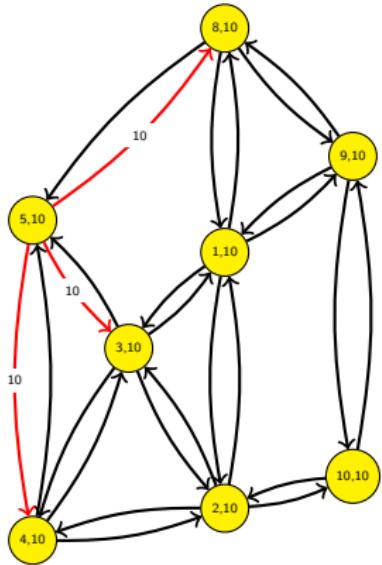
Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
 - $val_i := val_j$
 - Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Rounde 6

Un exemple

Exemple synchrone



Initialisation noeud i

- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

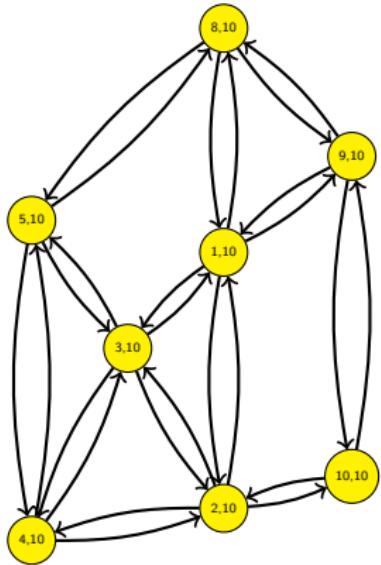
Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
- $val_i := val_j$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Rounde 7

Un exemple

Exemple synchrone



Initialisation noeud i

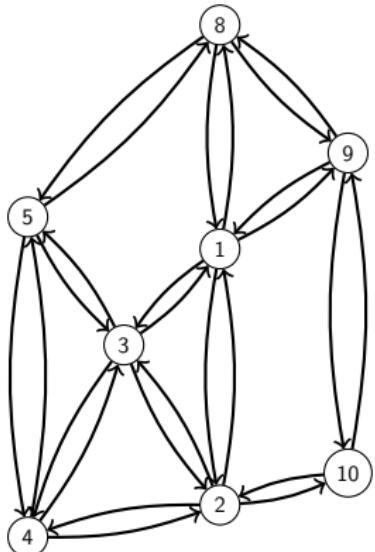
- $val_i = identifiant_i$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Lors de la réception de $< Valeur, val_j >$ envoyer par le noeud j

- if $val_i < val_j$:
- $val_i := val_j$
- Pour tout $j \in Voisins_i$ Envoyer $< Valeur, val_i >$ à j .

Rounde 8

Exemple asynchrone



Initialisation noeud i

- $val_i = \text{identifiant}_i$
- Pour tout $j \in \text{Voisins}_i$, Envoyer $\langle \text{Valeur}, val_i \rangle$ à j .

Lors de la réception de $\langle \text{Valeur}, val_j \rangle$ envoyer par le noeud j

- if $val_i < val_j$:
 - $val_i := val_j$
 - Pour tout $j \in \text{Voisins}_i$, Envoyer $\langle \text{Valeur}, val_i \rangle$ à j .

Le noeud 5 se réveille spontanément.

Un exemple

Qualités de l'algorithme

Avantage(s)

Simplicité

Un exemple

Qualités de l'algorithme

Avantage(s)

Simplicité

Inconvénient(s)

- Processeurs reçoivent plusieurs fois une même information
- Grand nombre de messages échangés

Un exemple

Qualités de l'algorithme

Avantage(s)

Simplicité

Inconvénient(s)

- Processeurs reçoivent plusieurs fois une même information
- Grand nombre de messages échangés

Question

Comment mesurer la qualité d'un algorithme réparti ?

Complexité des Algorithmes séquentiels

Notions

- Nombre d'opérations élémentaires

Complexité des Algorithmes séquentiels

Notions

- Nombre d'opérations élémentaires
- Espace mémoire occupé

Vocabulaires

Complexité des Algorithmes séquentiels

Notions

- Nombre d'opérations élémentaires
- Espace mémoire occupé

Vocabulaires

- Complexité temporelle

Question

Comment mesurer la qualité d'un algorithme réparti ?

Complexité des Algorithmes séquentiels

Notions

- Nombre d'opérations élémentaires
- Espace mémoire occupé

Vocabulaires

- Complexité temporelle
- Complexité spatiale

Question

Comment mesurer la qualité d'un algorithme réparti ?

Nombre de messages

Complexité des Algorithmes répartis

Réponse

Le nombre de message échangés.

Nombre de messages



Complexité des Algorithmes répartis

Réponse

Le nombre de message échangés.

Question

Est-ce suffisant ?

Tailles des messages

Tailles des messages

Taille d'un message

Nombre d'information que contient le message en binaire.

Tailles des messages

Tailles des messages

Taille d'un message

Nombre d'information que contient le message en binaire.

Exemples

- Un mot clé : taille constante $\leftarrow O(1)$ bits
- Un identifiant :
 - Si il y a n noeuds dans le réseaux, et que l'on note de 1 à n les identifiants des noeuds
 - Il faut $\log_2(n)$ bits pour noter l'identifiant n
 - Taille : $O(\log_2(n))$ bits

Comparaison d'algorithmes répartis

Soit A un algorithme échangeant :

- $O(n)$ messages
- de taille $O(n \log_2(n))$ bits

Soit B un algorithme échangeant :

- $O(n^2)$ messages
- de taille $O(\log_2(n))$ bits

Comparaison d'algorithmes répartis

Soit A un algorithme échangeant :

- $O(n)$ messages
- de taille $O(n \log_2(n))$ bits

Soit B un algorithme échangeant :

- $O(n^2)$ messages
- de taille $O(\log_2(n))$ bits

Question

Quel est le meilleur algorithme ?

Comparaison d'algorithmes répartis

Question

Quel est le meilleur algorithme ?

L'algorithme A échange au total :

- $n \times n \times \log_2 n$ bits
- soit $O(n^2 \log_2 n)$ bits

Comparaison d'algorithmes répartis

Question

Quel est le meilleur algorithme ?

L'algorithme *A* échange au total :

- $n \times n \times \log_2 n$ bits
- soit $O(n^2 \log_2 n)$ bits

L'algorithme *B* échange au total :

- $n^2 \times \log_2 n$ bits
- soit $O(n^2 \log_2 n)$ bits

Comparaison d'algorithmes répartis

Question

Quel est le meilleur algorithme ?

L'algorithme A échange au total :

- $n \times n \times \log_2 n$ bits
- soit $O(n^2 \log_2 n)$ bits

L'algorithme B échange au total :

- $n^2 \times \log_2 n$ bits
- soit $O(n^2 \log_2 n)$ bits

A et B sont donc équivalents en terme d'information échangées.

Taille des messages

Réseaux avec du routage IP (*Citation wikipédia : <http://fr.wikipedia.org/wiki/IPv4>*)

- Sur une interface déterminée, une trame a une taille maximale, appelée Maximum Transmission Unit ou MTU.
- Lorsque la longueur du paquet (datagramme) est supérieure, l'information sera fragmentée.
- La taille maximum supportée par IPv4 (car codée sur 16 bits) est de 64 Ko mais les réseaux ne prennent généralement pas en charge de trames de telles longueurs, en général on trouve des MTU de l'ordre de 1 500 octets (Ethernet).

Taille des messages

Conclusion :

Les messages de grosses tailles seront divisés en petits messages de petites tailles.

Taille des messages

En algorithmique répartie :

La taille de message «raisonnable» communément admise est $O(\log_2 n)$, où n est le nombre de processus.

Autres mesures de complexité

Question

Doit-on considérer

- la complexité temporelle et
- la complexité spatiale ?

Complexité temporelle :

Complexité temporelle :

Definition

Parmi toutes les executions possibles d'un algorithme distribué c'est l'execution qui maximisera le temps, qui définira la complexité temporelle dans le pire des cas.

Complexité spatiale :

Complexité spatiale :

Question :

Quel est l'intérêt de minimiser la mémoire physique ?

Caractéristiques des réseaux de capteurs :

- Batterie de durée limitée
- Peu de capacité mémoire

Complexité spatiale :

Complexité spatiale :

But des algorithmes répartis

- Faire des algorithmes pour les réseaux non homogènes.
- Pour cela il faut aussi minimiser la mémoire.

Complexité spatiale :

Complexités des algorithmes répartis

- Nombre de messages échangés
- Taille des messages (exprimé en bits)
- Temps d'exécution (exprimé en unité de temps ou en rondes)
- Espace mémoire utilisé par chaque machine