Introduction à

L'algorithmique distribuée

–

(Intelligence collective et décentralisée)

Arnaud Casteigts

–

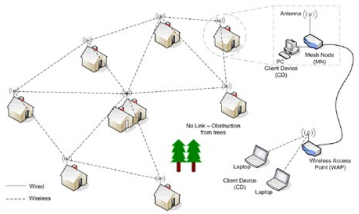
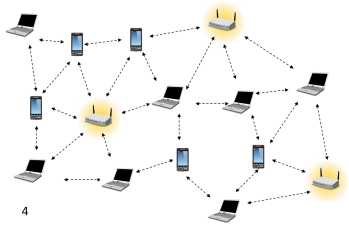
Université de Bordeaux

Dans les technologies

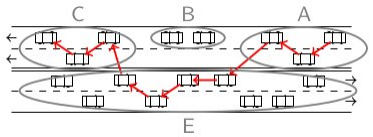
1. Internet

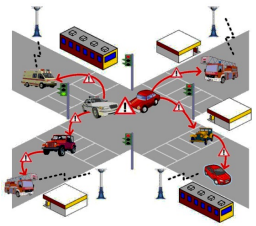


2. Réseaux maillés

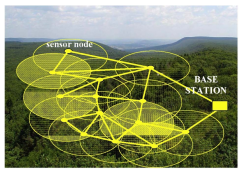



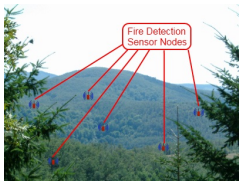
3. Réseaux véhiculaires

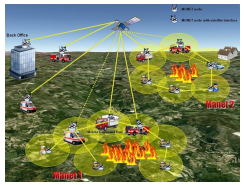





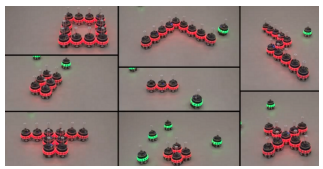
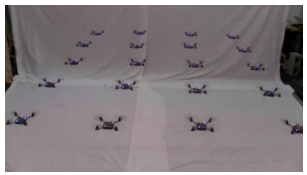
4. Réseaux de capteurs







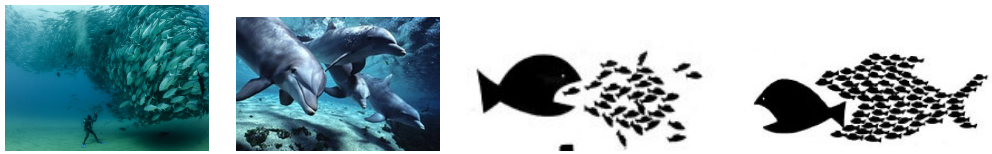
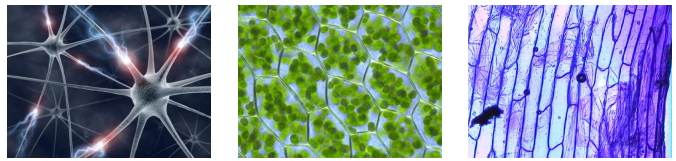
5. Robots et drones

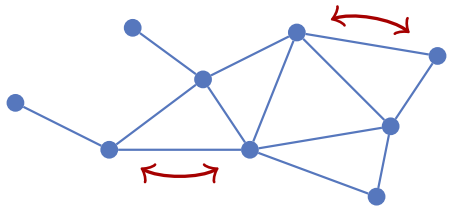
→ Tous ces contextes ont besoin d'algorithmes distribués.

Dans la nature...

Le monde a une essence décentralisée, de nombreux systèmes naturels font preuve d'intelligence collective.

1. Dans le ciel 
2. Sur terre 
3. Dans l'eau 
4. Dans les organismes 
5. En société 

Algorithmique distribuée



Concrètement ?

- Chaque entité (nœud) a sa propre capacité d'exécution
- Chaque entité interagit avec ses voisins directs
- L'intelligence du système est distribuée/décentralisée ($\neq$ parallélisme)
- Les entités ne connaissent pas l'ensemble du réseau

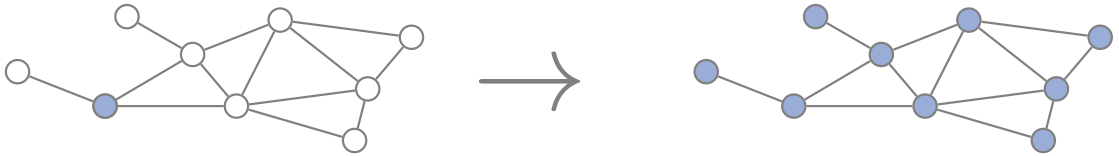
À partir de ces interactions locales, on veut résoudre des problèmes globaux.
(think globally, act locally)

Quels problèmes étudie-t-on ?

Problèmes classiques les plus étudiés :

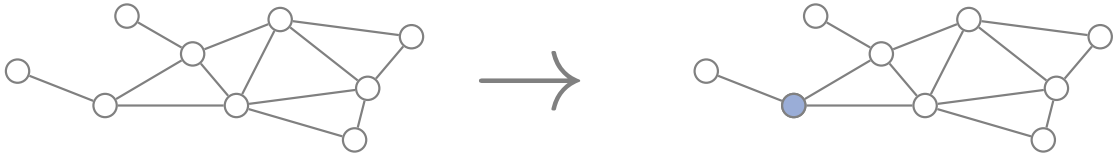
Diffusion d'information

(propager une information d'une entité vers toutes les autres)



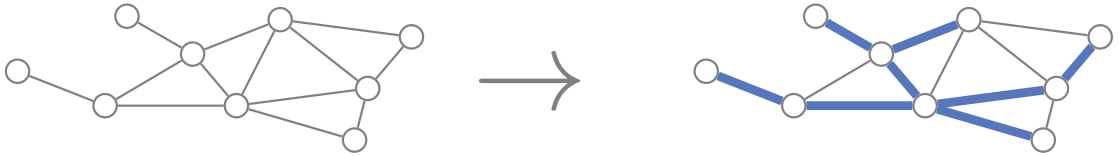
Élection

(choisir exactement une entité pour créer de la centralisation)



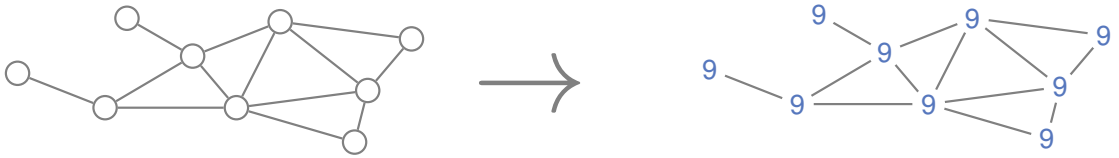
Arbre couvrant

(selectionner un ensemble d'arêtes sans cycle qui relie toutes les entités)



Comptage (ou énumération)

(déterminer combien de participants il y a)



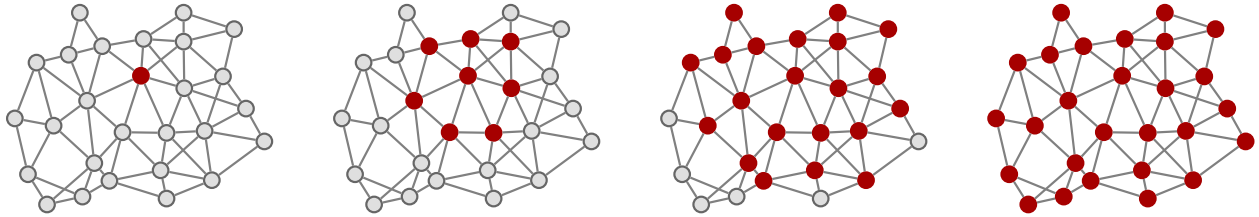
Et bien d'autres : consensus, nommage, routage, cartographie, équilibrage de charge, collecte d'info, ...

→ Les problèmes réels peuvent se réduire à une ou plusieurs de ces briques de bases.

Deux exemples basiques

Prototype d'algorithme de diffusion d'information :

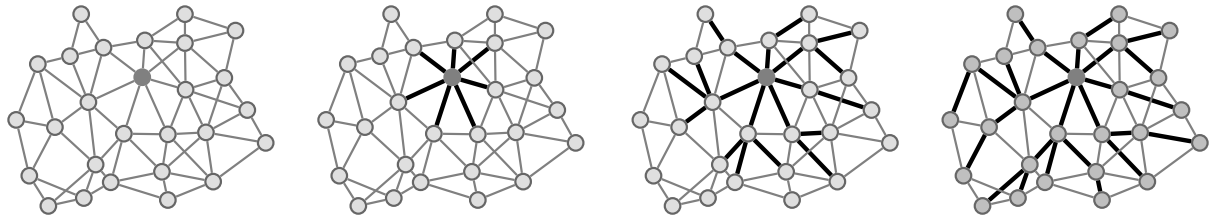
onStart():
```
if (informed)
  sendAll(message)
```
onMessage(message):
```
if (¬ informed)
  informed ← true
  sendAll(message)
```



initialement | un peu plus tard | encore plus tard | finalement

Construction d'un arbre couvrant :

onStart():
```
if (root)
  sendAll(message)
```
onMessage(message):
```
if (parent == null)
  parent ← message.sender
  sendAll(message)
```



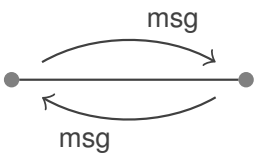
initialement | un peu plus tard | encore plus tard | finalement

Dans ces exemples, l'algorithme est le même partout, mais l'état initial est différent.

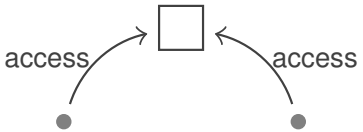
→ on parle d'initialisation non-uniforme (ou uniforme si l'état initial est le même partout).

Un peu de contexte

Modèles de communication les plus utilisés :



Passage de messages



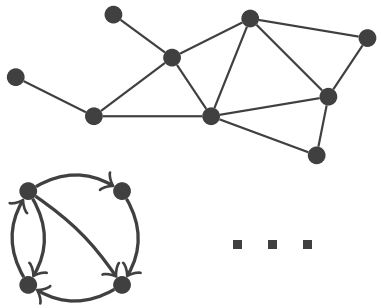
Mémoire partagée

Mais aussi : registres sur les arêtes, agents mobiles, transformation de graphes, et encore (dans la nature) communication acoustique, tactile, visuelle, olfactive, *etc.*

→ Dans ce cours : passage de messages (surtout) et transformations de graphes.

Représentation du réseau :

→ Graphe dont les arêtes représentent les communications possibles entre entités (les sommets, noeuds).



Le graphe peut être orienté ou non-orienté.

Un peu de vocabulaire

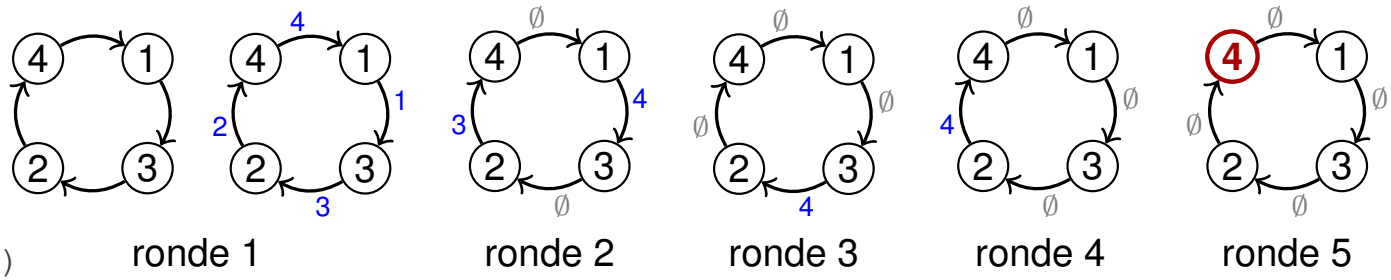
Il est souvent supposé que les noeuds ont des identifiants uniques (p.ex. adresses IP ou mac), mais pas toujours → réseau anonyme.

Si tous les noeuds démarrent dans le même état, on parle d'initialisation uniforme. Mais il est souvent utile de faire démarrer un ou plusieurs noeud dans un état différent (initialisation non-uniforme, p.ex. la future racine d'un arbre).

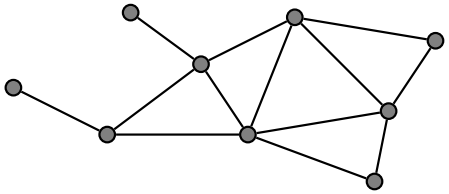
Une exécution est dite synchrone si l'envoi et la réception des messages se produit au sein de rondes (rounds) de communication bien cloisonnés (ex. ci-dessous).

Voici un exemple d'algorithme avec identifiants, initialisation uniforme et qu'on supposera synchrone (ici). Il s'agit de l'algorithme de Le Lann, Chang et Roberts (LCR), qui élit le noeud ayant l'identifiant le plus grand dans un anneau orienté :

onStart():
```
send(myID)
```
onMessage(otherID):
```
if (otherID > myID)
  send(otherID)
else
  if (otherID == myID)
    ELECTED ← true
```

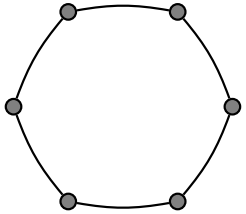


Quelques topologies fréquentes

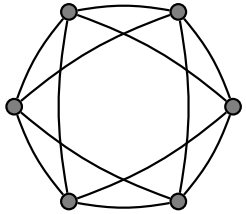


Arbitraire

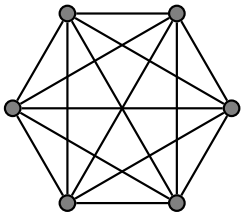
Bus



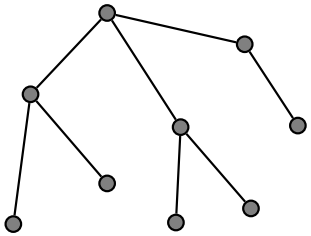
Anneau



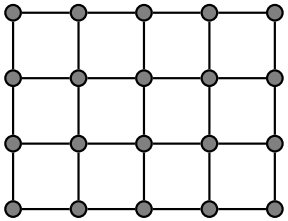
Anneau avec raccourcis



Graphe complet



Arbre



Grille

\+ Variantes orientées.