Systèmes Distribués I

Alain Cournier et Stéphane Devismes

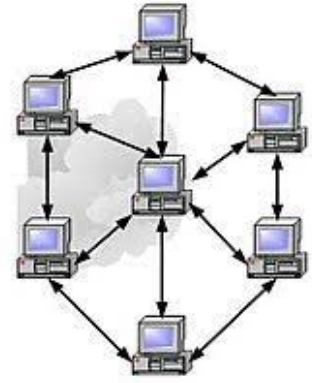
Université de Picardie Jules Verne

15 septembre 2022

Contenu

Algorithmique distribuée $\approx$ Algorithmique des réseaux

Applications : Internet, Cloud, réseaux de capteurs sans fils, réseaux sociaux, Blockchain, ...



Différences avec l'algorithmique classique :

Calcul réparti : Plusieurs entités de calcul autonomes (processus)

Données réparties : Chaque entité n'a qu'une vue partielle du système

Échange d'informations : Communication asynchrone par message

Absence de temps global : *e.g.*, pas d'horloge commune

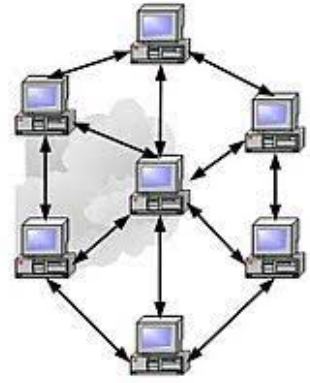
Problèmes considérés : briques de bases pour les réseaux

- élection, allocation de resource, diffusion, *etc.*

Contenu

Algorithmique distribuée $\approx$ Algorithmique des réseaux

Applications : Internet, Cloud, réseaux de capteurs sans fils, réseaux sociaux, Blockchain, ...



Différences avec l'algorithmique classique :

Calcul réparti : Plusieurs entités de calcul autonomes (processus)

Données réparties : Chaque entité n'a qu'une vue partielle du système

Échange d'informations : Communication asynchrone par message

Absence de temps global : *e.g.*, pas d'horloge commune

Problèmes considérés : briques de bases pour les réseaux

- élection, allocation de resource, diffusion, *etc.*

Google Drive

Dropbox

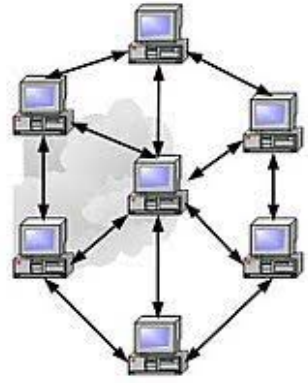
Gnutella

BitTorrent

Contenu

Algorithmique distribuée $\approx$ Algorithmique des réseaux

Applications : Internet, Cloud, réseaux de capteurs sans fils, réseaux sociaux, Blockchain, ...



Différences avec l'algorithmique classique :

Calcul réparti : Plusieurs entités de calcul autonomes (processus)

Données réparties : Chaque entité n'a qu'une vue partielle du système

Échange d'informations : Communication asynchrone par message

Absence de temps global : *e.g.*, pas d'horloge commune

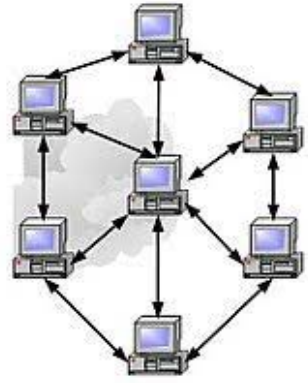
Problèmes considérés : briques de bases pour les réseaux

- élection, allocation de resource, diffusion, *etc.*

Contenu

Algorithmique distribuée $\approx$ Algorithmique des réseaux

Applications : Internet, Cloud, réseaux de capteurs sans fils, réseaux sociaux, Blockchain, ...



Différences avec l'algorithmique classique :

Calcul réparti : Plusieurs entités de calcul autonomes (processus)

Données réparties : Chaque entité n'a qu'une vue partielle du système

Échange d'informations : Communication asynchrone par message

Absence de temps global : *e.g.*, pas d'horloge commune

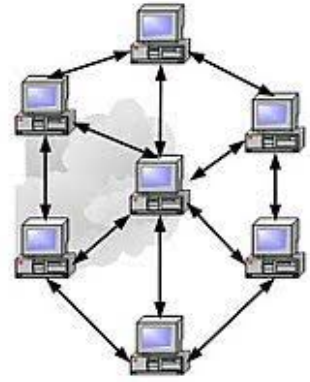
Problèmes considérés : briques de bases pour les réseaux

- élection, allocation de resource, diffusion, *etc.*

Contenu

Algorithmique distribuée $\approx$ Algorithmique des réseaux

Applications : Internet, Cloud, réseaux de capteurs sans fils, réseaux sociaux, Blockchain, ...



Différences avec l'algorithmique classique :

Calcul réparti : Plusieurs entités de calcul autonomes (processus)

Données réparties : Chaque entité n'a qu'une vue partielle du système

Échange d'informations : Communication asynchrone par message

Absence de temps global : *e.g.*, pas d'horloge commune

Problèmes considérés : briques de bases pour les réseaux

- élection, allocation de resource, diffusion, *etc.*

Google Drive

Dropbox

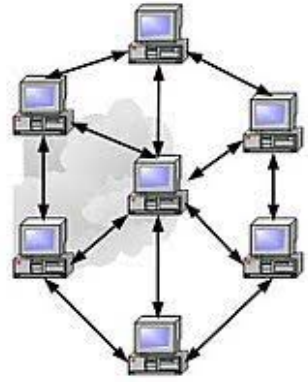
Gnutella

BitTorrent

Contenu

Algorithmique distribuée $\approx$ Algorithmique des réseaux

Applications : Internet, Cloud, réseaux de capteurs sans fils, réseaux sociaux, Blockchain, ...



Différences avec l'algorithmique classique :

Calcul réparti : Plusieurs entités de calcul autonomes (processus)

Données réparties : Chaque entité n'a qu'une vue partielle du système

Échange d'informations : Communication asynchrone par message

Absence de temps global : *e.g.*, pas d'horloge commune

Problèmes considérés : briques de bases pour les réseaux

- élection, allocation de resource, diffusion, *etc.*

Objectifs

1. **Savoir écrire des algorithmes distribués**
   - 1.1 Être capable de spécifier formellement un problème
   - 1.2 À partir d'un ensemble d'hypothèses sur le système (topologie, identification, *etc.*) être capable d'écrire un algorithme distribué réalisant la spécification.

2. **Savoir prouver la correction et analyser la complexité d'algorithmes distribués**

3. **Savoir déployer sur un simulateur les algorithmes vus en cours et TD**

   Utilisation du simulateur évènementiel SINALGO (plug-in JAVA sous *Eclipse*).

Objectifs

1. **Savoir écrire des algorithmes distribués**
   - 1.1 Être capable de spécifier formellement un problème
   - 1.2 À partir d'un ensemble d'hypothèses sur le système (topologie, identification, *etc.*) être capable d'écrire un algorithme distribué réalisant la spécification.

2. **Savoir prouver la correction et analyser la complexité d'algorithmes distribués**

3. **Savoir déployer sur un simulateur les algorithmes vus en cours et TD**

   Utilisation du simulateur évènementiel SINALGO (plug-in JAVA sous *Eclipse*).

Objectifs

1. **Savoir écrire des algorithmes distribués**
   - 1.1 Être capable de spécifier formellement un problème
   - 1.2 À partir d'un ensemble d'hypothèses sur le système (topologie, identification, *etc.*) être capable d'écrire un algorithme distribué réalisant la spécification.

2. **Savoir prouver la correction et analyser la complexité d'algorithmes distribués**

3. **Savoir déployer sur un simulateur les algorithmes vus en cours et TD**

   Utilisation du simulateur évènementiel SINALGO (plug-in JAVA sous *Eclipse*).

Objectifs

1. **Savoir écrire des algorithmes distribués**
   - 1.1 Être capable de spécifier formellement un problème
   - 1.2 À partir d'un ensemble d'hypothèses sur le système (topologie, identification, *etc.*) être capable d'écrire un algorithme distribué réalisant la spécification.

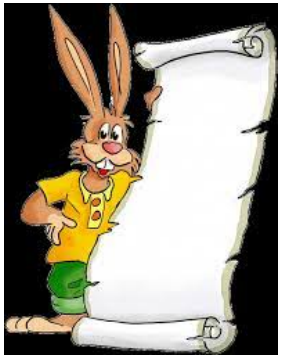
2. **Savoir prouver la correction et analyser la complexité d'algorithmes distribués**

3. **Savoir déployer sur un simulateur les algorithmes vus en cours et TD**

   Utilisation du simulateur évènementiel SINALGO (plug-in `JAVA` sous *Eclipse*).

Programme détaillé

30H : CTD, TD et TP



1. Rappel L3 (modèle et circulation de jeton)
2. Présentation du simulateur SINALGO
3. Les horloges logiques (Lamport, Mattern, ...)
4. Propagation d'Information avec Retour
5. Tables de routage
6. Prise d'instantanées et détection de terminaison
7. Impossibilité du consensus en présence de pannes (FLP'85)

Questions ?