Autour du pair-à-pair

Fabien Mathieu

Habilitation à diriger des recherches, 11/02/2009

Plan

Plan

Compréhension usuelle

Pour l'individu λ, le P2P, c'est des logiciels :

Partage de fichiers BitTorrent, eMule,...

Diffusion de vidéos TVants, PPLive,...

Téléphone par Internet Skype

Compréhension usuelle

Pour l'individu λ, le P2P, c'est des logiciels :

Partage de fichiers BitTorrent, eMule,...

Diffusion de vidéos TVants, PPLive,...

Téléphone par Internet Skype

D'autres logiciels ne sont pas P2P :

Non-réseau Office, Photoshop,...

Réseau Apache, SSH, navigateurs...

Une définition légale ?

Pour lutter contre les usages illégaux, le P2P est souvent défini comme logiciel permettant le partage et la diffusion de contenu.

Une définition légale ?

Pour lutter contre les usages illégaux, le P2P est souvent défini comme logiciel permettant le partage et la diffusion de contenu.

- Couvre les logiciels P2P de partage (fichiers ou vidéos)
- Ne couvre pas Skype
- Couvre des applications non P2P : Apache, SSH. . .

Définition réseau

La plupart des applications réseaux obéissent à une dichotomie client/serveur :

- D'un côté, un (des) serveur(s) capables de fournir des ressources,
- De l'autre, des clients qui demandent les ressources

Définition réseau

La plupart des applications réseaux obéissent à une dichotomie client/serveur :

- D'un côté, un (des) serveur(s) capables de fournir des ressources,
- De l'autre, des clients qui demandent les ressources

En réseau, le P2P se définit souvent comme l'abandon du modèle client/serveur : chaque entité (pair) peut être à la fois client (récepteur) et serveur (émetteur).

Définition réseau

La plupart des applications réseaux obéissent à une dichotomie client/serveur :

- D'un côté, un (des) serveur(s) capables de fournir des ressources,
- De l'autre, des clients qui demandent les ressources

En réseau, le P2P se définit souvent comme l'abandon du modèle client/serveur : chaque entité (pair) peut être à la fois client (récepteur) et serveur (émetteur).

Cette définition satisfait tous les exemples précédents

Skype est-il vraiment P2P ?

Skype est très souvent donné en exemple de P2P. Pourquoi ?

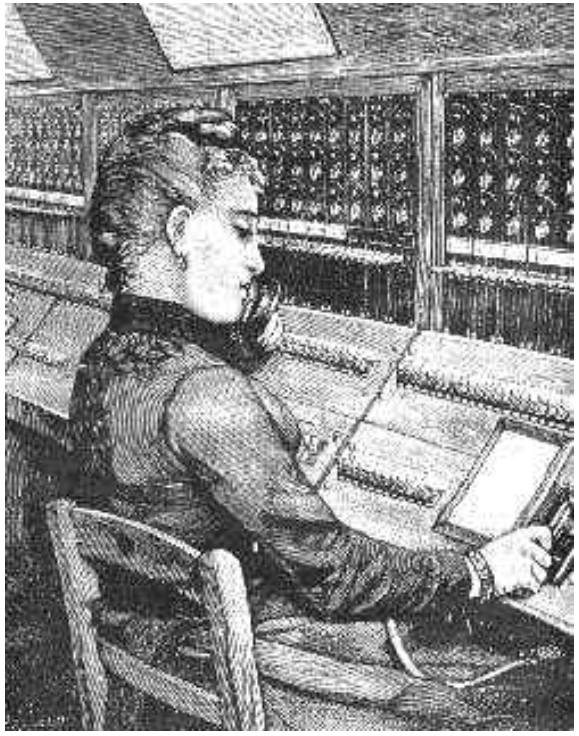
- Interaction de pair-à-pair ?

Existe depuis très longtemps

Skype est-il vraiment P2P ?

Skype est très souvent donné en exemple de P2P. Pourquoi ?

- Interaction de pair-à-pair ?
- Interactions multiples de pair-à-pair ?



Existe depuis assez longtemps

Skype est-il vraiment P2P ?

Skype est très souvent donné en exemple de P2P. Pourquoi ?

- Interaction de pair-à-pair ?
- Interactions multiples de pair-à-pair ?
- Développé par les programmeurs de KaZaA et Joost ?

Retour au sens commun

Skype est-il vraiment P2P ?

Skype est très souvent donné en exemple de P2P. Pourquoi ?

- Interaction de pair-à-pair ?
- Interactions multiples de pair-à-pair ?
- Développé par les programmeurs de KaZaA et Joost ?
- Une propriété non-triviale ?

Proposition : définition par la structure

L'approche structurelle

Structure : agencement, entre eux, des éléments constitutifs d'un ensemble, considéré comme une caractéristique durable de cet ensemble (Larousse).
Concrètement, dans un réseau d'informations, la structure est ce qui permet de répondre à la question

Qui récupère quoi de la part de qui ?

Systèmes clients/serveurs, téléphone,. . . sont caractérisés par des structures triviales.
Proposition : le P2P, c'est quand la structure d'un réseau est non-triviale.

Définitions du pair-à-pair : récapitulatif

Sens commun Médias généralistes

DADVSI Si ça sert à télécharger des divX

De pair à pair N'est pas de type client/serveur

Structure Quand ce n'est pas trivial

Définition	Apache	Téléphone	Skype	BitTorrent
Sens Commun	Non	Non	Oui	Oui
DADVSI	Oui	Non	Non	Oui
~~Client/Serveur~~	Non	Oui	Oui	Oui
Structurelle	Non	Non	Parfois	Oui

Bestiaire du P2P

Les sous-domaines P2P peuvent s'analyser

- Objectif de la structure : chercher, récupérer ?
- Méthode : quelle type de structure utiliser ?
- Réalisation/décision : global/local, réactif/pro-actif ?

Bestiaire du P2P

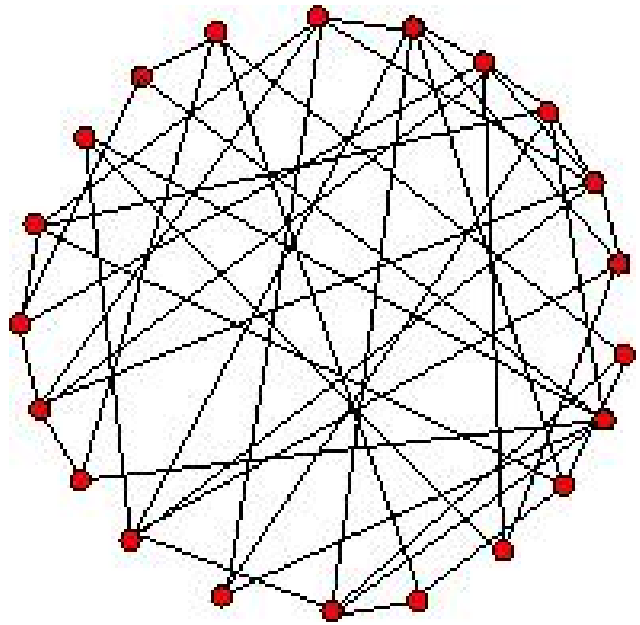
Les sous-domaines P2P peuvent s'analyser

- Objectif de la structure : chercher, récupérer ?
- Méthode : quelle type de structure utiliser ?
- Réalisation/décision : global/local, réactif/pro-actif ?

Fil conducteur : des structures explicites déterministes aux auto-structures aléatoires.

Comment représenter une structure ?

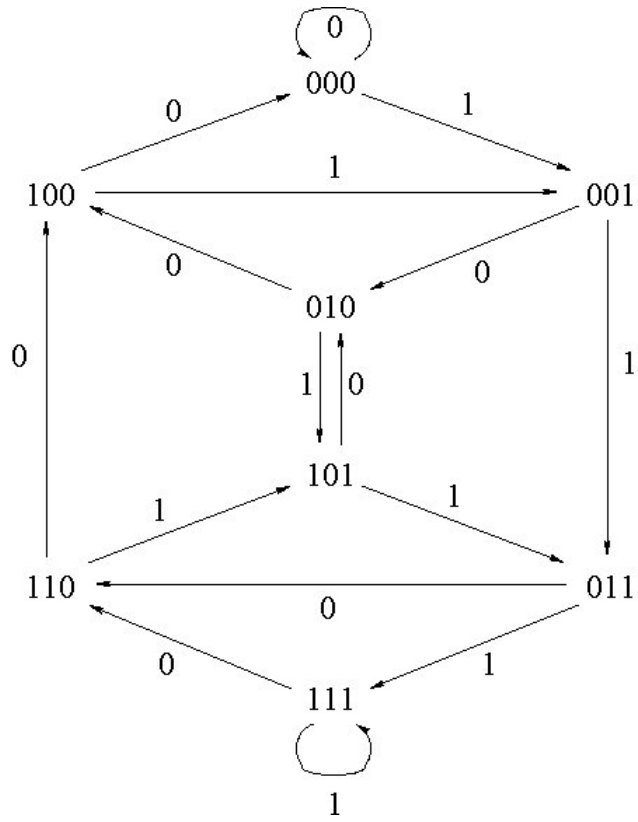
- Il s'agit de définir les interactions entre les constituants d'un système.
- Souvent représentée par un (plusieurs) graphe.



  - **Erdös-Rényi**

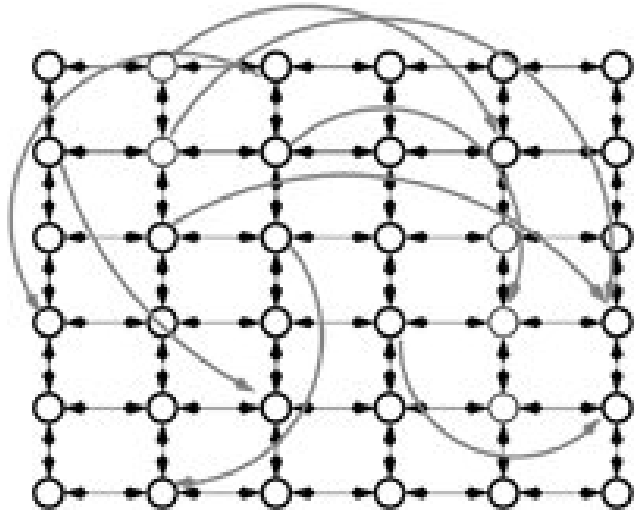
Comment représenter une structure ?

- Il s'agit de définir les interactions entre les constituants d'un système.
- Souvent représentée par un (plusieurs) graphe.
  - Erdös-Rényi
  - **de de Bruijn**



Comment représenter une structure ?

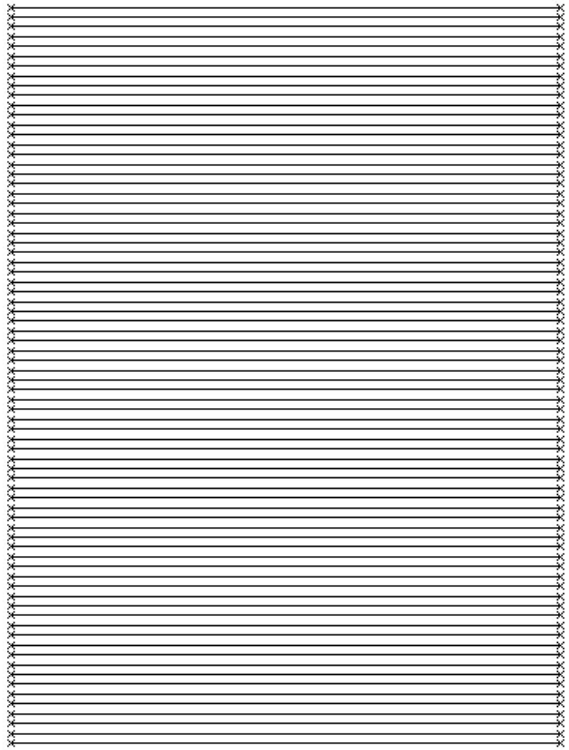
- Il s'agit de définir les interactions entre les constituants d'un système.
- Souvent représentée par un (plusieurs) graphe.



  - Erdös-Rényi
  - de de Bruijn
  - **Small World**

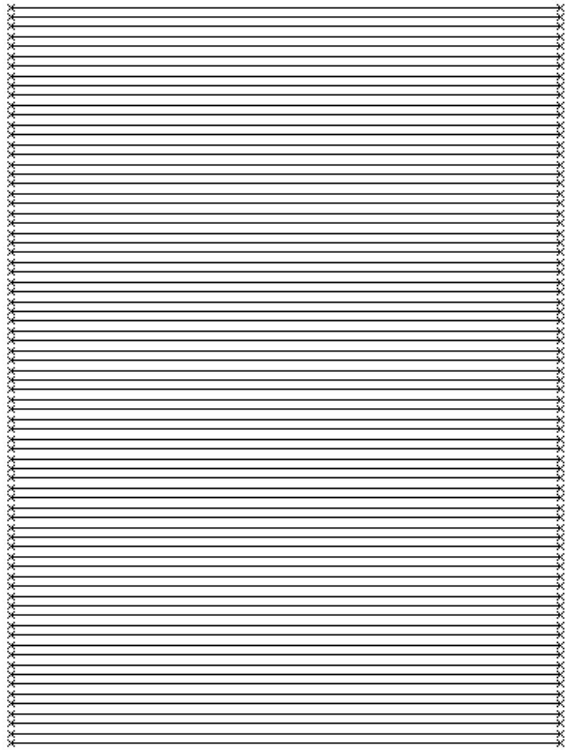
Comment représenter une structure ?

- Il s'agit de définir les interactions entre les constituants d'un système.
- Souvent représentée par un (plusieurs) graphe.
- Mais ce n'est pas toujours suffisant.
  - Erdös-Rényi
  - de de Bruijn
  - Small World
  - **Rat : couplage parfait. . .**



Comment représenter une structure ?

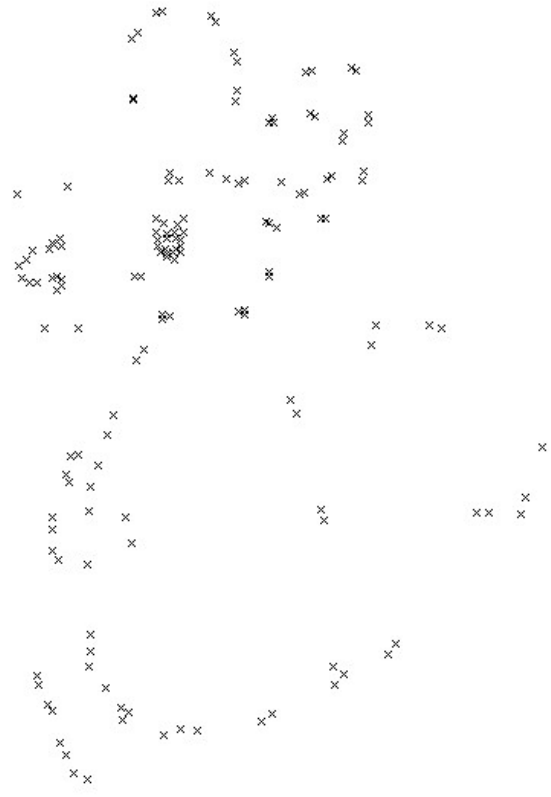
- Il s'agit de définir les interactions entre les constituants d'un système.
- Souvent représentée par un (plusieurs) graphe.
- Mais ce n'est pas toujours suffisant.
  - Erdös-Rényi
  - de de Bruijn
  - Small World
  - **Rat :** couplage parfait. . .
  - **. . . plus topologie spécifique.**



Structure explicite

La plupart des réseaux dits structurés fonctionnent ainsi :

- Les pairs ;



Structure explicite

La plupart des réseaux dits structurés fonctionnent ainsi :

- Les pairs ;
- Le graphe ;

- $1 \leftrightarrow 2$,
- $3 \leftrightarrow 4$,
- ...
- $151 \leftrightarrow 152$,

Structure explicite

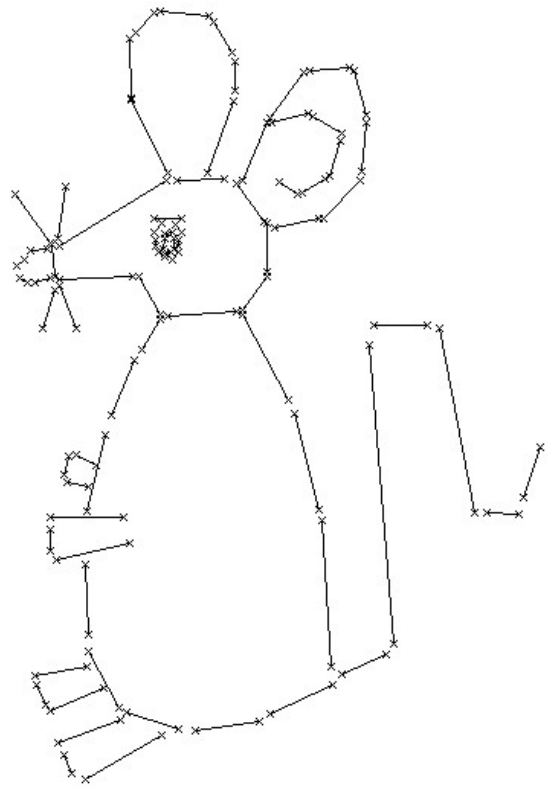
La plupart des réseaux dits structurés fonctionnent ainsi :

- Les pairs ;
- Le graphe ;
- Les pairs numérotés ;

Structure explicite

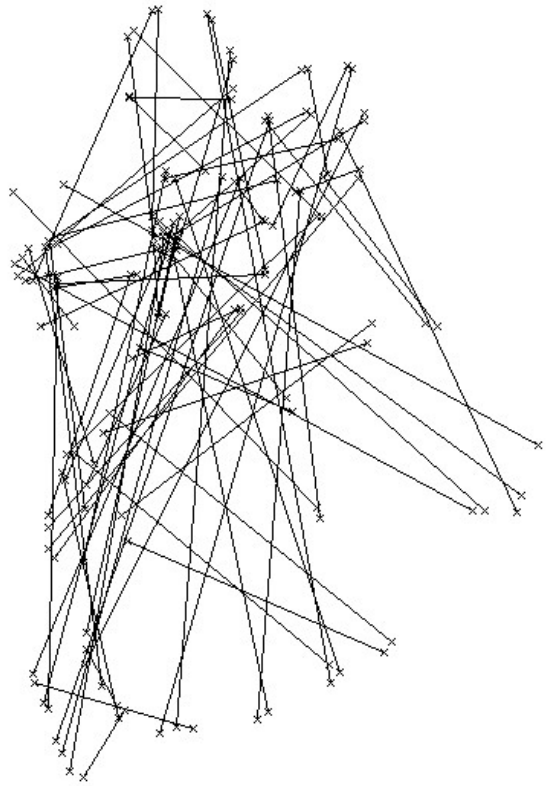
La plupart des réseaux dits structurés fonctionnent ainsi :

- Les pairs ;
- Le graphe ;
- Les pairs numérotés ;
- La structure.



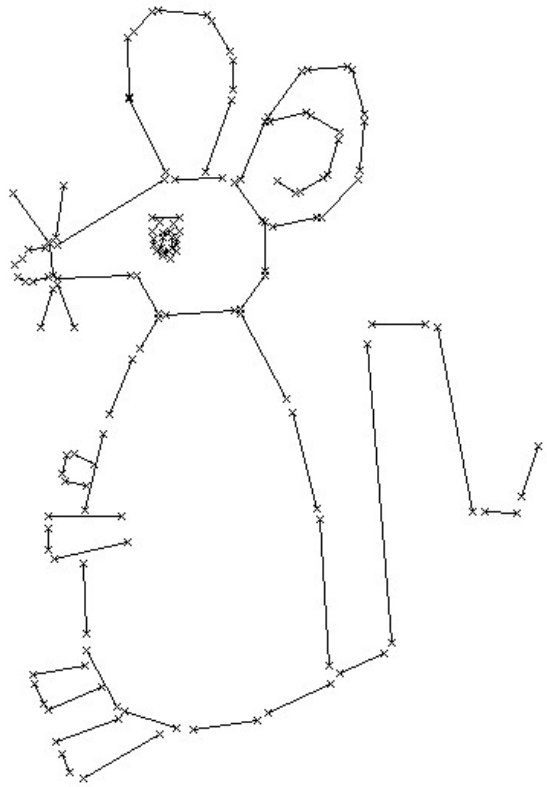
Limite des structures explicites

- Difficile de garantir que le résultat respectera la structure voulue, surtout si c'est plus qu'un simple graphe.



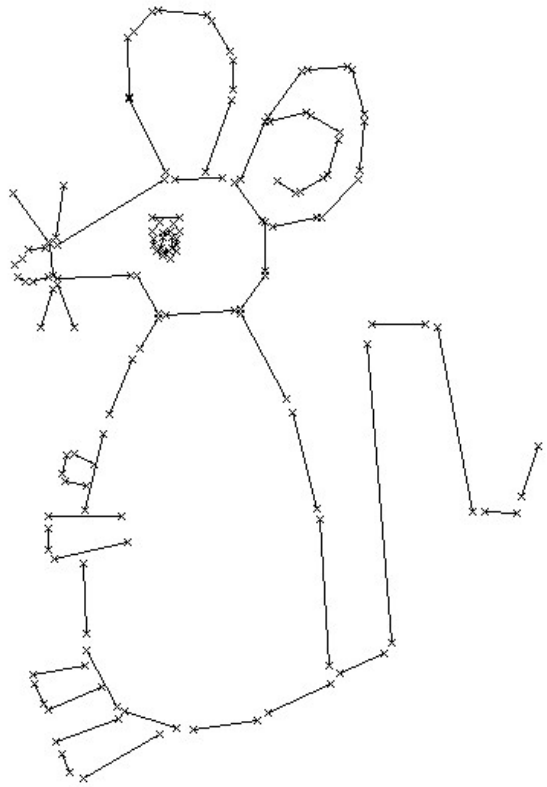
Limite des structures explicites

- Difficile de garantir que le résultat respectera la structure voulue, surtout si c'est plus qu'un simple graphe.
- Solution : renforcer le contrôle, imposer des contraintes supplémentaires (ré-étiqueter) ;



Limite des structures explicites

- Difficile de garantir que le résultat respectera la structure voulue, surtout si c'est plus qu'un simple graphe.
- Solution : renforcer le contrôle, imposer des contraintes supplémentaires (ré-étiqueter) ;
- Inconvénient : coût



Auto-structure : principe

- Laisser les pairs faire des choix locaux
  - Connaissance limitée
  - Compenser l'ignorance par le hasard

Auto-structure : principe

- Laisser les pairs faire des choix locaux
  - Connaissance limitée
  - Compenser l'ignorance par le hasard
- Prendre la structure qui résulte de ces choix
- Avantages : robuste et léger
- Inconvénient : contrôle indirect et affaibli

Auto-structure : principe

- Laisser les pairs faire des choix locaux
  - Connaissance limitée
  - Compenser l'ignorance par le hasard
- Prendre la structure qui résulte de ces choix
- Avantages : robuste et léger
- Inconvénient : contrôle indirect et affaibli
  - Obtient-on une structure convenable ?

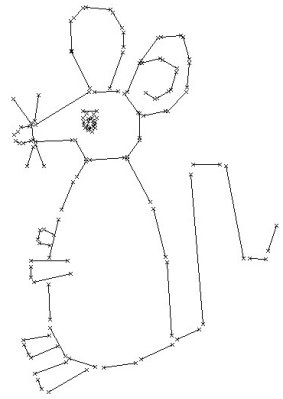
Auto-structure : principe

- Laisser les pairs faire des choix locaux
  - Connaissance limitée
  - Compenser l'ignorance par le hasard
- Prendre la structure qui résulte de ces choix
- Avantages : robuste et léger
- Inconvénient : contrôle indirect et affaibli
  - Obtient-on une structure convenable ?
  - Obtient-on même une structure ?

Explicite ou auto structures : récapitulatif

Structure explicite :

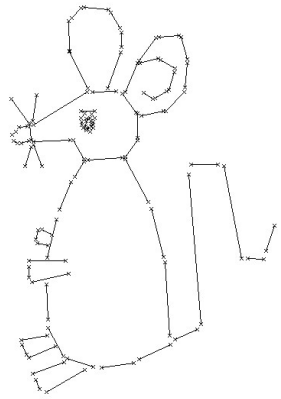
- Toujours possible d'avoir exactement ce qu'on veut
- Coût potentiellement important
- Peu de flexibilité



Explicite ou auto structures : récapitulatif

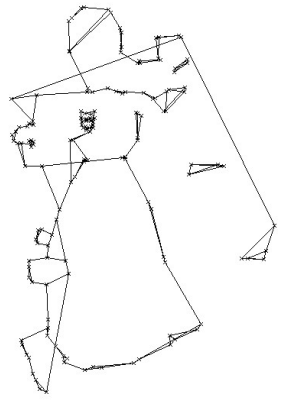
Structure explicite :

- Toujours possible d'avoir exactement ce qu'on veut
- Coût potentiellement important
- Peu de flexibilité



Auto-structure :

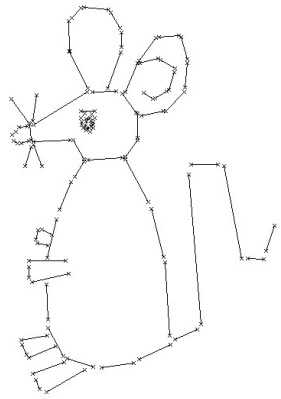
- Vision incomplète autorisée
- Flexibilité
- Si trop relaxé, risque de structure faible



Explicite ou auto structures : récapitulatif

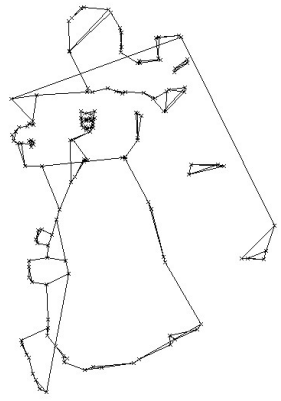
Structure explicite :

- Toujours possible d'avoir exactement ce qu'on veut
- Coût potentiellement important
- Peu de flexibilité



Auto-structure :

- Vision incomplète autorisée
- Flexibilité
- Si trop relaxé, risque de structure faible



Il faut trouver le bon compromis

Plan

Distribution de contenu

On distingue généralement trois principaux modes de distribution de contenu.

Téléchargement Distribution générique

- Seul importe la récupération du contenu
- Critères de qualité : disponibilité, temps de téléchargement

Distribution de contenu

On distingue généralement trois principaux modes de distribution de contenu.

Vidéo-à-la-demande Pouvoir regarder ce qu'on veut quand on veut

- Comme en téléchargement, le contenu est connu
- Il faut minimiser le start-up delay quelle que soit la demande

Distribution de contenu

On distingue généralement trois principaux modes de distribution de contenu.

Diffusion en léger différé

Regarder une chaîne de télévision

- Tout le monde veut voir la même chose en même temps
- Minimiser le différé

Distribution de contenu

On distingue généralement trois principaux modes de distribution de contenu.

Méthode de distribution	Téléch.	Vidéo à-la-demande	Léger différé
Streaming ?	Non	Oui	Oui
Qualité requise	Minimum	N'importe quoi, n'importe quand	petit décalage
Avantage technique	Pas de QoS	Contenu connu	Comportement homogène
Difficulté technique	Minimum	Comportement hétérogène	Anticipation impossible

Distribution de contenu

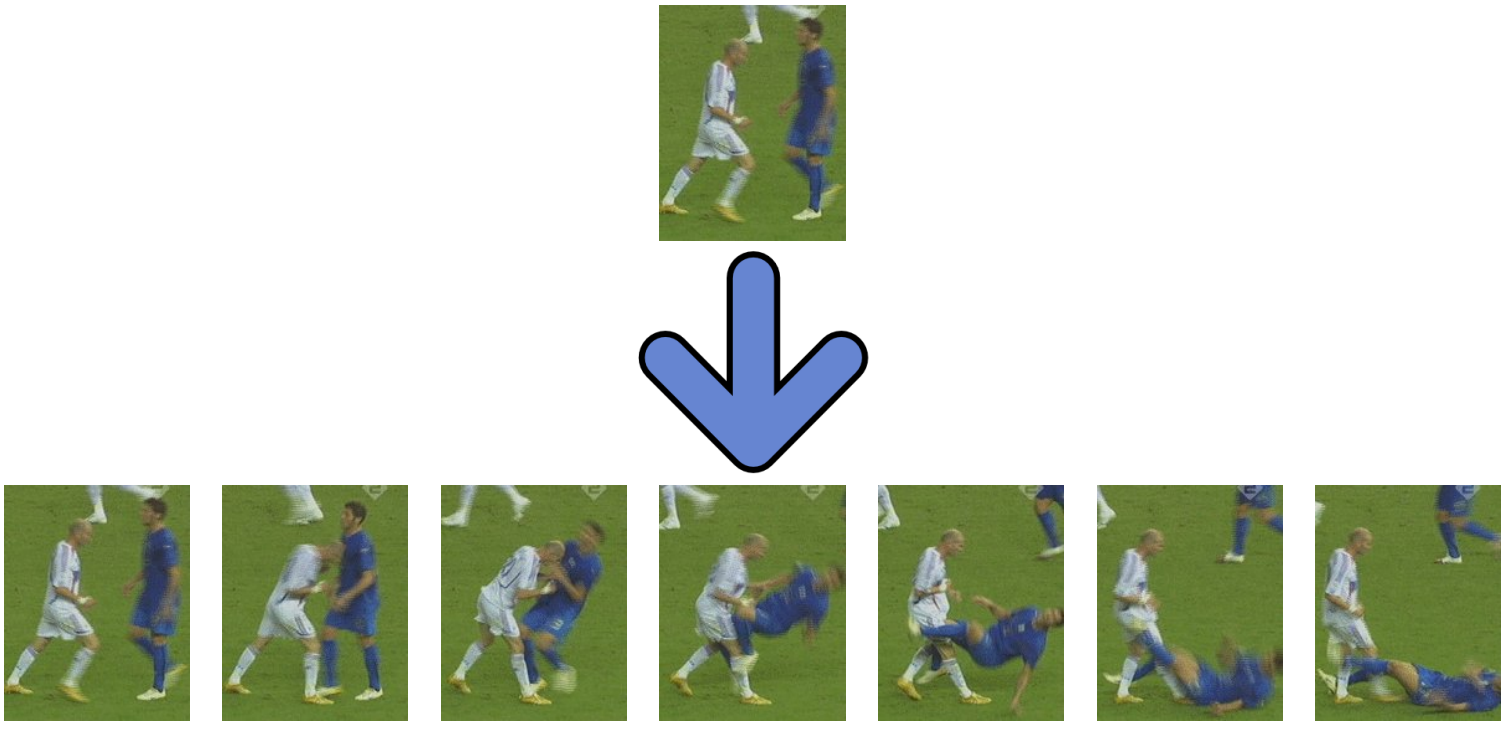
On distingue généralement trois principaux modes de distribution de contenu.

Méthode de distribution	Téléch.	Vidéo à-la-demande	**Léger différé**
Streaming ?	Non	Oui	**Oui**
Qualité requise	Minimum	N'importe quoi, n'importe quand	**petit décalage**
Avantage technique	Pas de QoS	Contenu connu	**Comportement homogène**
Difficulté technique	Minimum	Comportement hétérogène	**Anticipation impossible**

Cas d'école

- Un émetteur unique veut émettre un flux multimedia
  - Flux continu émis en direct, de durée inconnue
  - L'émetteur ne peut transmettre qu'une seule copie du flux
- Un nombre fixé *n* de spectateurs
  - Tous les spectateurs veulent voir le flux
  - Chaque spectateur a la capacité de ré-émettre un flux
- Comment faire marcher tout ça ?

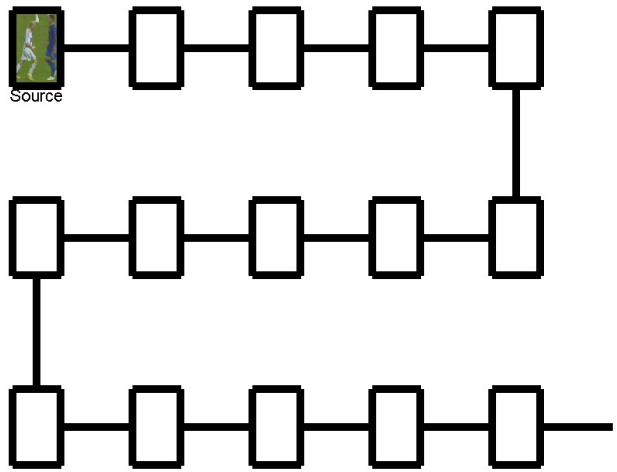
Découpage du flux



- Il est plus pratique de découper le flux en petits morceaux (chunks)
- La source injecte le flux chunk par chunk
- Les spectateurs peuvent relayer les chunks déjà reçus

Solution 1 : transmission linéaire

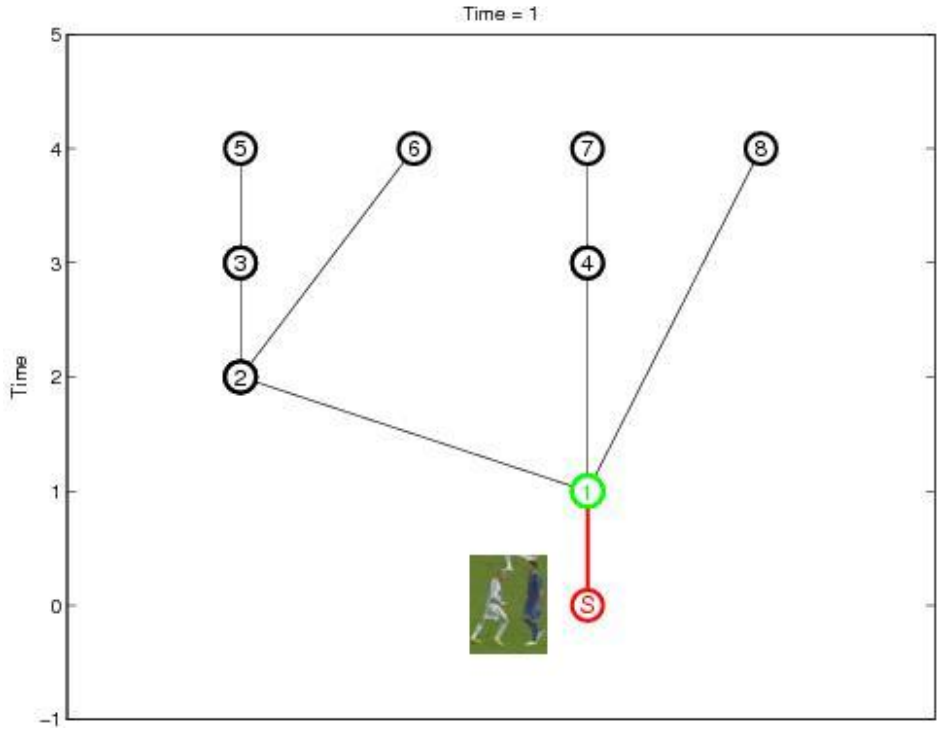
- Les chunks sont transmis à la queue leu-leu
- Tout les spectateurs arrivent à voir le flux
- Décalage très important (proportionnel à n)
- Robustesse très faible (50% de pertes en moyenne en cas de panne simple)



Transmission optimale simple

Comment transmettre un chunk le plus vite possible ?

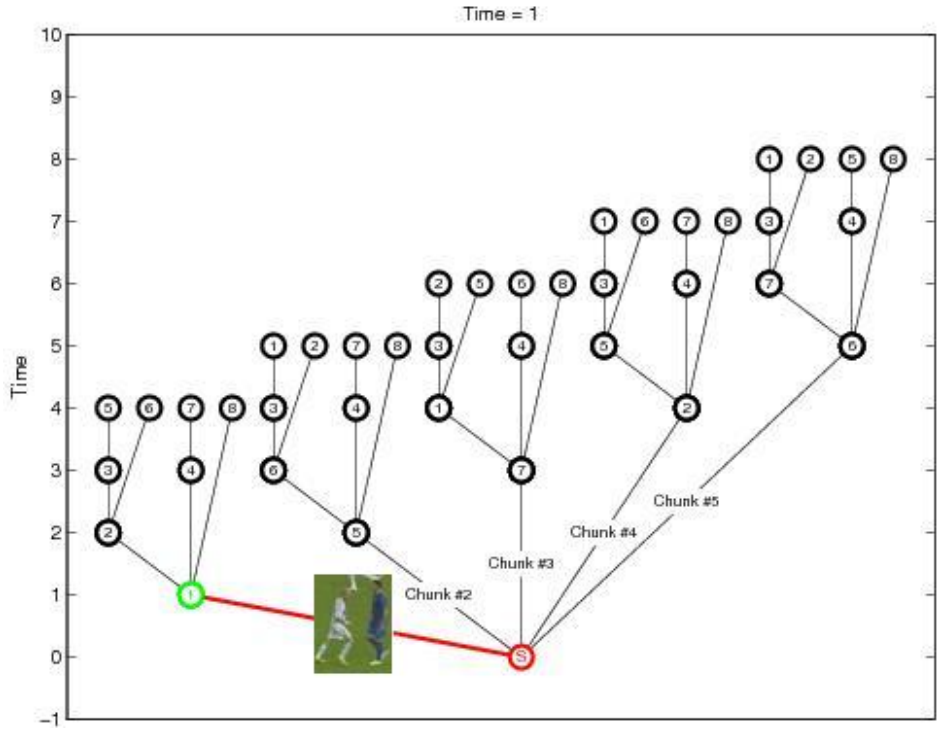
- Un des spectateur reçoit le chunk ($t = 1$)
- Ceux qui ont transmettent à ceux qui n'ont pas
- C'est optimal :
  - Le chunk est transmis à tout le monde en temps $\log_2(n)$,
  - Une panne simple affecte en moyenne $\log_2(n)$ spectateurs.



Transmission optimale multiple

Il y a maintenant une succession de chunks

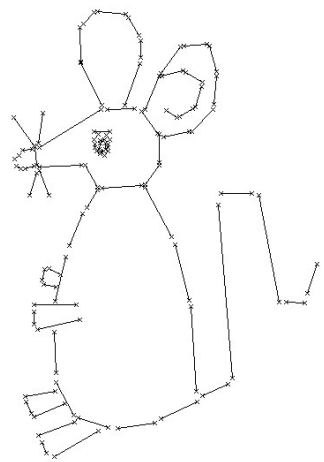
- Pour être optimal, il faut faire des permutations de la transmission simple
- Risque de conflits entre transmissions (collisions)
  - à l'émission (rouge)
  - à la réception (vert) si capacité limitée
- Il faut un peu réfléchir, mais ça marche



Transmission optimale multiple

Il y a maintenant une succession de chunks

- Pour être optimal, il faut faire des permutations de la transmission simple
- Risque de conflits entre transmissions (collisions)
  - à l'émission (rouge)
  - à la réception (vert) si capacité limitée
- Il faut un peu réfléchir, mais ça marche
- On a une structure optimale !



Diffusion épidémique

- Principe : créer une auto-structure en définissant le comportement des émetteurs
- Approche dite *push* (*pull*$\rightarrow$ récepteur)
- Concrètement, on va définir une fonction locale de diffusion
  - Entrée : un émetteur prêt à émettre, les informations dont il dispose
  - Sortie : quel chunk émettre vers quel récepteur
  - Simplification : choisir d'abord le récepteur, puis le chunk, ou l'inverse

Fonctions de diffusion élémentaires

Il suffit de définir des briques

- Choix du récepteur
  - Hasard (rp)
  - Hasard utile (up)
  - Plus nécessiteux (dp)
  - ...
- Choix du chunk
  - Plus récent (lb)
  - Plus récent utile (lu)
  - Hasard utile (ru)
  - ...
- Il suffit ensuite de combiner : rp/lb (= lb/rp), dp/lu, dp/ru, lb/up, lu/up, lu/dp...

Efficacité des fonctions épidémiques

- Métrique : sur-délai, pertes, contrôle
- Étude : théorie si possible, banc d'essai (simulations) sinon
- Résultat : compromis sur les métriques
  - Peu de contrôle $\rightarrow$ sur-délai et/ou pertes limités mais non nuls (rp/lu)
  - Fonctions optimales $\rightarrow$ énorme contrôle
- Remarques :
  - On peut parfois échanger du sur-délai contre des pertes (ex : rp/lu)
  - Certaines fonctions sont comparables entre elles (dp/lu et dp/ru)
  - Sans pondération des métriques, il est cependant difficile de donner un gagnant absolu

Plan

Principe

- Partir d'un modèle abstrait qui semble intéressant *a priori*
  $\rightarrow$ théorie des mariages stables (Gale & Shapley, 1962)
- Adapter le modèle au contexte (ici le P2P)
  $\rightarrow$ réseaux à préférences (Mathieu & al., 2006)
- Étudier les propriétés
- Comprendre l'éclairage que cela apporte

Réseaux à préférences : définitions

Un réseau à préférences, c'est :

- Un ensemble de n pairs,
- Un graphe d'acceptabilité G (interactions possibles),
- Un vecteur b de quotas sur les interactions simultanées,
- Des préférences : chaque pair classe ses voisins.

L'ensemble des interactions effectives est appelé configuration

- c'est un sous-graphe acceptable qui respecte les quotas.
- Les pairs essaient de trouver de meilleurs voisins, modifiant la configuration

Réseaux à préférences : dynamique

L'action élémentaire individuelle est l'initiative,

- elle consiste à tester un voisin.
- Si les deux sont intéressés, une interaction est créée,
- des interactions peuvent être rompues à cause des quotas.

Les initiatives sont définies par leur nature et leur « rythme » :

- Nature : compromis entre le hasard et l'explicite
  - Test aléatoire,
  - Cherche le meilleur voisin possible,
  - Recherche cyclique dans le voisinage
- Rythme : activité des pairs
  - Round-Robin,
  - Poissonnien,
  - Adversarial

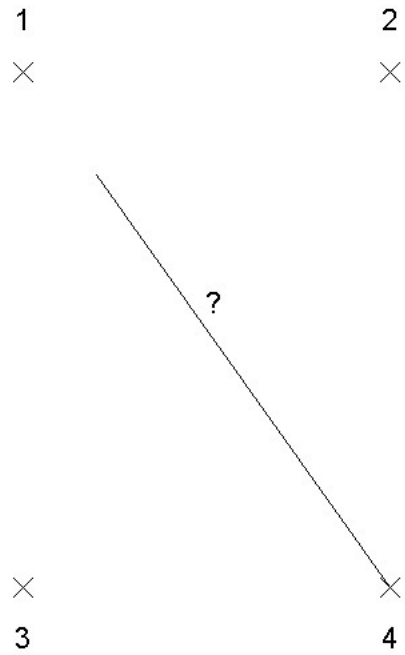
Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

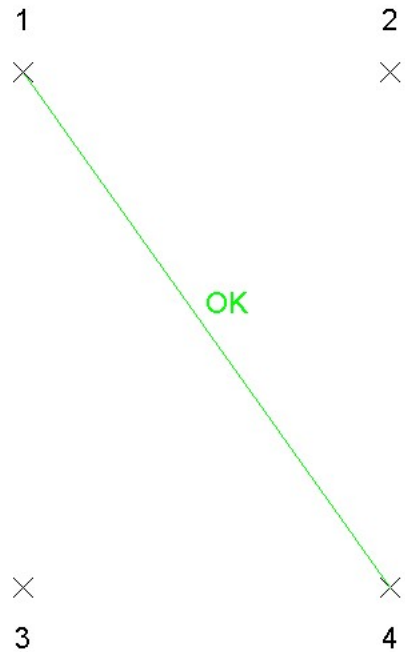
4 prend une initiative et demande à collaborer avec 1



Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

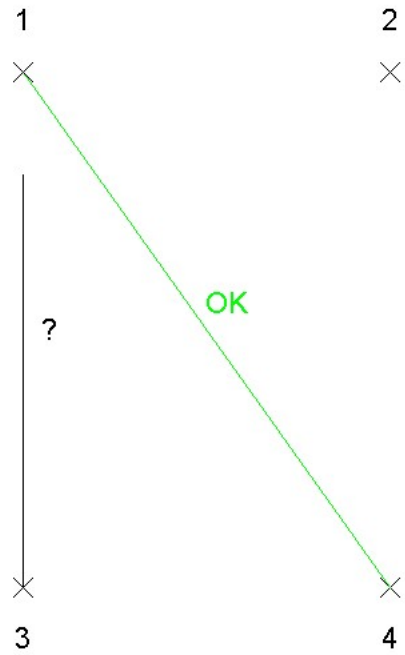
1 accepte, 1 et 4 collaborent



Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

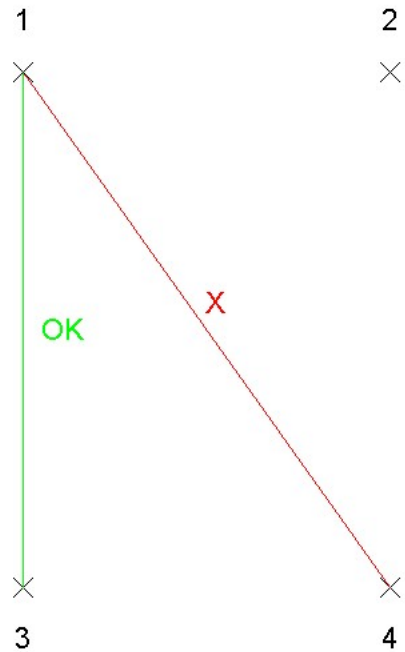
3 prend une initiative et demande à collaborer avec 1



Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

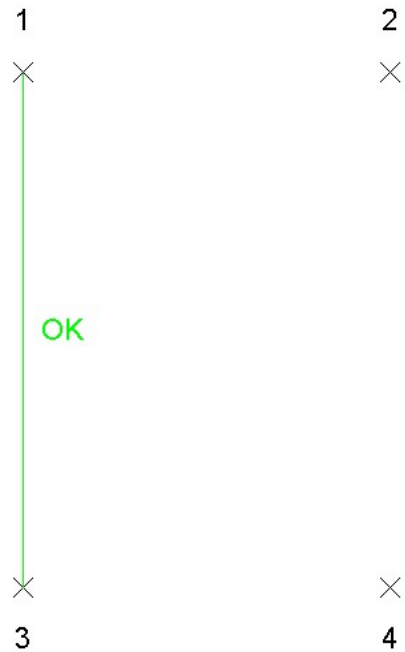
1 accepte, 1 et 3 collaborent



Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

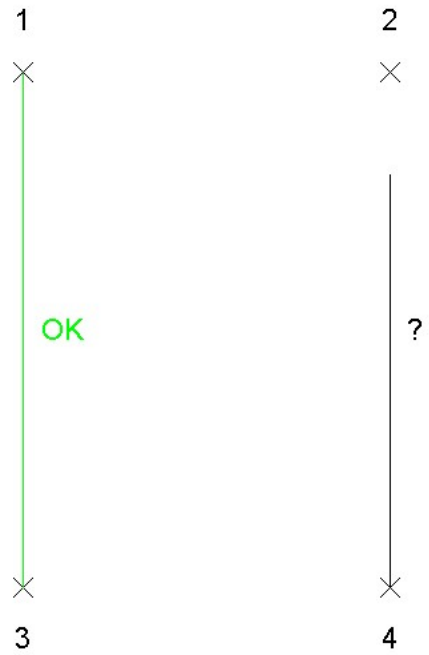
4 perd sa collaboration avec 1



Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

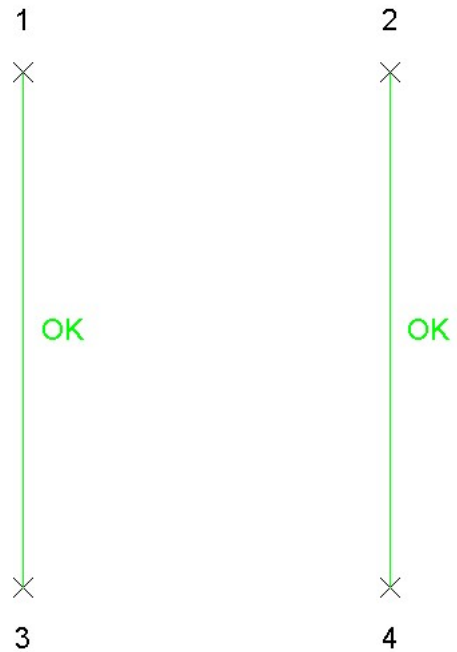
4 prend une initiative et demande à collaborer avec 2



Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

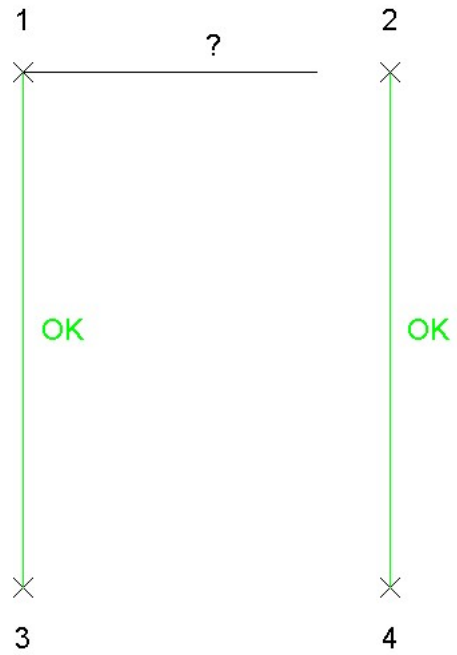
2 accepte, 2 et 4 collaborent



Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

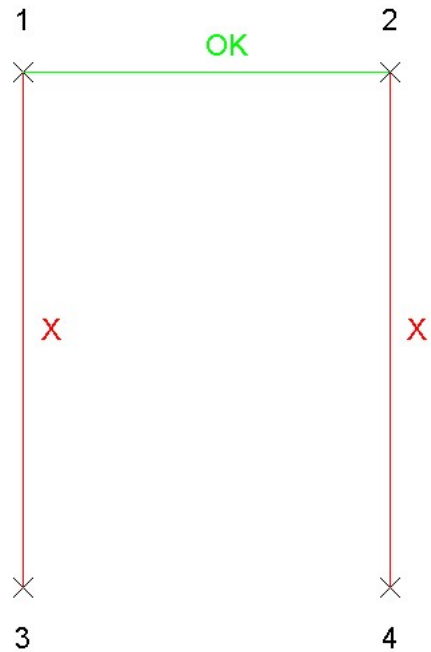
1 prend une initiative et demande à collaborer avec 2



Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

2 accepte, 1 et 2 collaborent



Un exemple

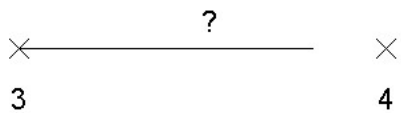
- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

3 et 4 perdent leurs collaborations avec 1 et 2

Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

3 prend une initiative et demande à collaborer avec 4



Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

4 accepte, 3 et 4 collaborent

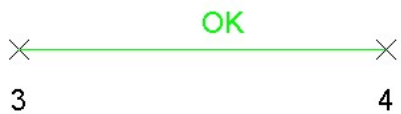
Un exemple

- Quatre pairs 1, 2, 3, 4 qui se connaissent, $b = 1$
- 1 préfère 2, puis 3 puis 4
- 2 préfère 1, puis 3 puis 4
- 3 préfère 1, puis 2 puis 4
- 4 préfère 1, puis 2 puis 3

Bilan :

- 1 collabore avec 2, 3 avec 4
- Plus aucune initiative ne peut modifier la configuration
- On est dans une

  configuration stable



Grand théorème des préférences acycliques

- Un cycle de préférences est un cycle de $k \geq 3$ pairs tel que chaque pair préfére son successeur à son prédécesseur :
  - i_1 préfère i_2 à i_k,
  - i_2 préfère i_3 à i_1,
  - $\ldots$,
  - i_k préfère i_1 à i_{k-1}.
- Théorème : un réseau à préférences acycliques admet une, et une seule, configuration stable. De plus, il est auto-stabilisant par initiatives.
- Un réseau à préférences acycliques est donc naturellement auto-structuré

Intérêt des préférences acycliques en P2P

Beaucoup de mesures conduisent à des préférences acycliques

Qualités intrinsèques bande passante, stockage, CPU, uptime...

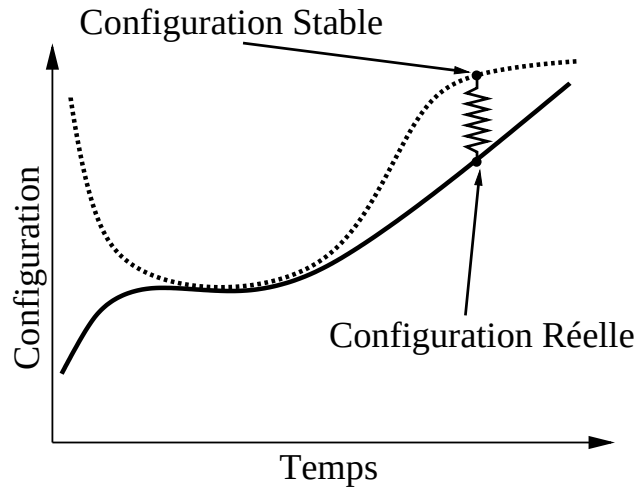
Qualités réciproques (symétriques) RTT, distances, co-uptime...

Qualités complémentaires complémentarité de contenu, d'uptime,...

Combinaisons de qualités précédentes

Étude des préférences acycliques : méthodologie

La configuration réelle va toujours chercher à s'approcher de la configuration stable.



Étude des préférences acycliques : méthodologie

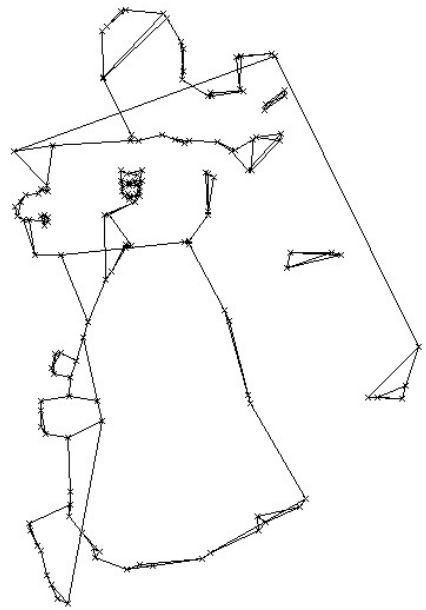
La configuration réelle va toujours chercher à s'approcher de la configuration stable.

- À quelle vitesse ?
- Si la convergence est assez rapide, la configuration stable donnera un bon indice du comportement réel

Étude des préférences acycliques : méthodologie

La configuration réelle va toujours chercher à s'approcher de la configuration stable.

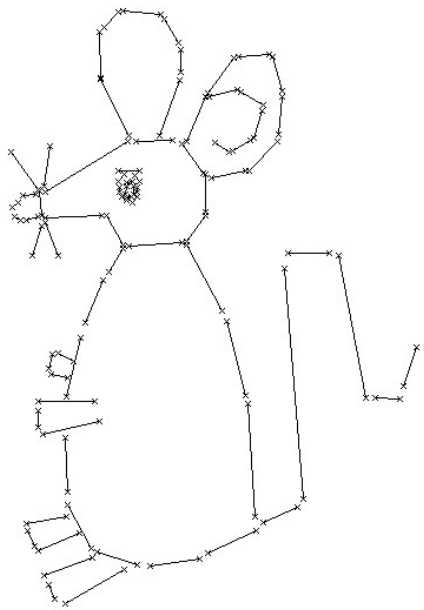
- Quelles propriétés a la configuration stable ?
- Décrit-elle une structure explicite ?
- Avec quelle marge d'erreur ?
- A-t-elle un intérêt ?



Exemple 1 : stratification

Caractéristique des préférences globales

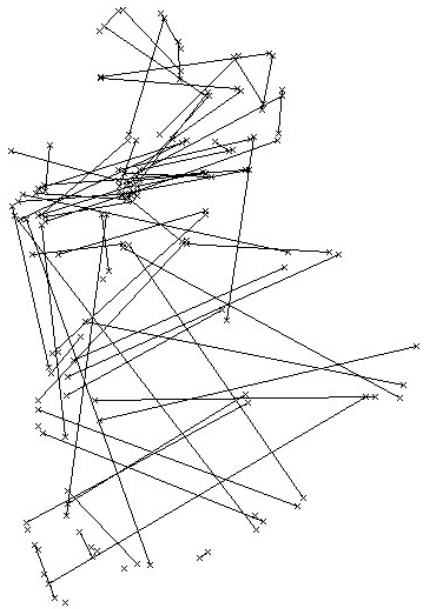
- Avec une connaissance parfaite, la configuration stable est explicite
- La structure obtenue est la plus équitable possible



Exemple 1 : stratification

Caractéristique des préférences globales

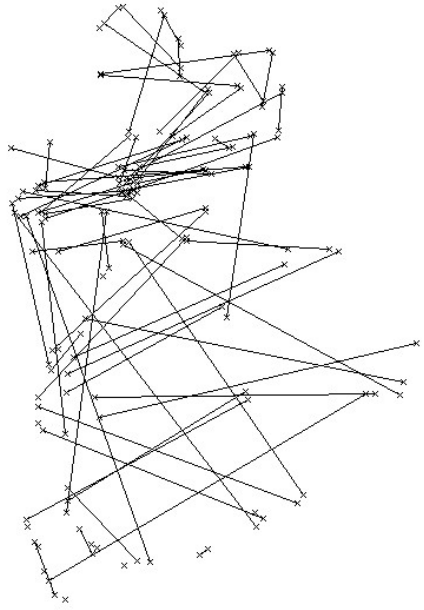
- Avec une connaissance parfaite, la configuration stable est explicite
- La structure obtenue est la plus équitable possible
- Mais si la connaissance est imparfaite...
- ...l'équité le devient aussi
- C'est le phénomène de stratification



Exemple 1 : stratification

Caractéristique des préférences globales

- Avec une connaissance parfaite, la configuration stable est explicite
- La structure obtenue est la plus équitable possible
- Mais si la connaissance est imparfaite...
- ...l'équité le devient aussi
- C'est le phénomène de stratification
- Permet de comprendre le protocole BitTorrent



Exemple 2 : Petit-mondisation

- Caractéristique des préférences géométriques
- Rappel : un petit-monde est un graphe avec certaines propriétés :
  - Petit diamètre
  - Grand clustering
- Sous certaines conditions, des réseaux à préférences géométriques peuvent produire des petits-mondes
- Applications
  - Générateur naturel de petits-mondes
  - Notion de dimension au sens des préférences
  - Possibilité de routage à la Milgram

Plan

Conclusion

- En P2P, la notion de structure est omni-présente
- Au fil des mes recherches, je me suis rapproché des auto-structures
- Approche pratique
  - Considérer un problème concret
  - Réfléchir à la structure optimale
  - Trouver une auto-structure moins gourmande et plus robuste
- Approche théorique
  - Considérer un modèle d'auto-structure
  - Réfléchir à ses propriétés
  - Trouver à quoi ça peut servir

Perspectives de l'approche pratique

- Apprivoiser plus finement les structures optimales
  - Capacités hétérogènes
  - Modèles plus réalistes
- Pondérer les métriques
- Trouver des auto-structures gagnantes
- Aller au prototype ?

Perspectives de l'approche théorique

- Approfondir la compréhension du modèle
- Injecter les résultats dans l'approche pratique
- Trouver de nouveaux domaines d'application

Perspectives du pair-à-pair

- L'avenir est incertain, pour des raisons légales, mais aussi techniques
- Mais la recherche en P2P n'est pas condamné pour autant

Définition	BitTorrent	Green C.	Cloud C.	Ad-Hoc	CDNs
Sens Commun	Oui	Non	Non	Non	Non
DADVSI	Oui	Non	Parfois	Non	Oui
~~Client/Serveur~~	Oui	Non	Non	Oui	Non
Structurelle	Oui	Oui	Oui	Oui	Oui

Questions