Contributions à l'expérimentation sur les systèmes distribués de grande taille

Lucas Nussbaum

Soutenance de thèse
4 décembre 2008

Contexte scientifique :

Expérimentation sur les systèmes distribués

Systèmes distribués ?

Définition

Ensemble de nœuds indépendants, communiquant via un réseau, coopérant pour fournir un service.

- Systèmes pair-à-pair (DHT, diffusion de fichiers, ...)
- Desktop computing (BOINC)
- Calcul à haute performances, grilles de calcul
- ...

Systèmes distribués : propriétés recherchées

Les systèmes distribués doivent :

- permettre le passage à l'échelle
- supporter l'hétérogénéité (nœuds, réseau)
- être tolérant aux pannes
- permettre d'obtenir les performances souhaitées

Exemple emblématique : les systèmes P2P

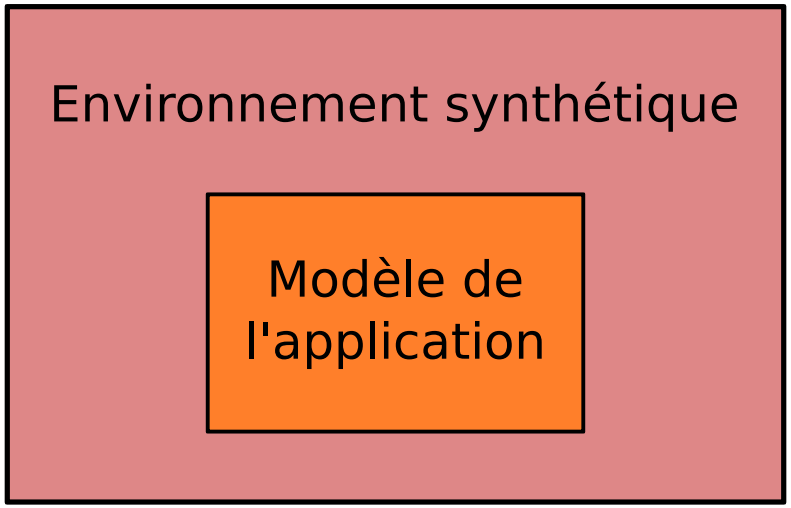
$\Rightarrow$ Besoin de pouvoir évaluer et prédire les performances

Évaluation des performances des syst. distribués

Différentes approches :

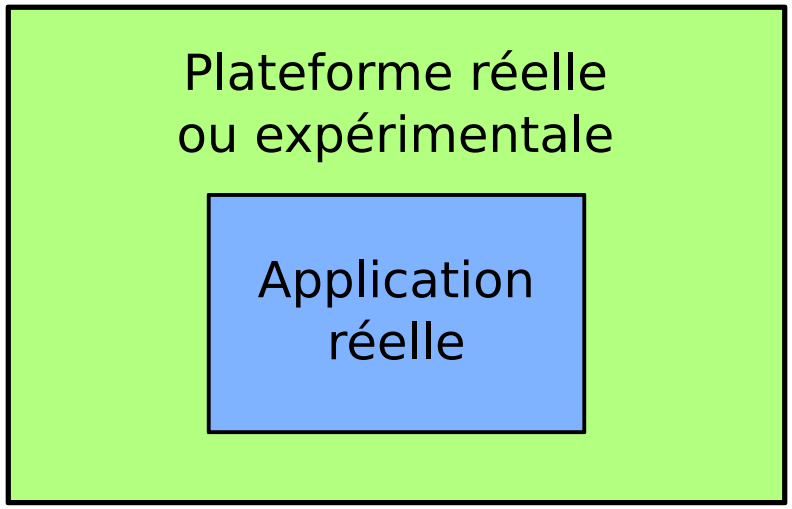
- Théoriques :
  - Modélisation, approche analytique, ...
- Expérimentales :
  - Simulation
  - Exécution sur plateformes réelles ou expérimentales
  - Émulation

Simulation des systèmes distribués



- Basée sur un modèle, pas sur l'application réelle
- Compromis précision du modèle / temps de simulation
- Permet d'obtenir rapidement des résultats

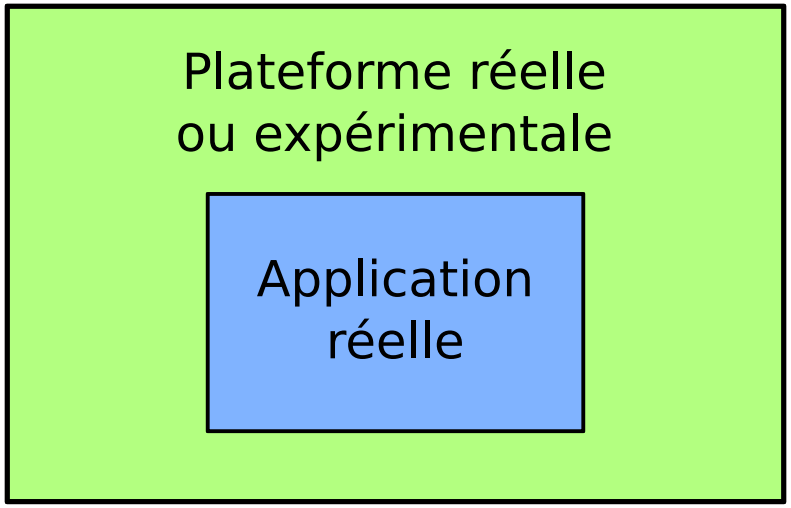
Plateformes réelles ou expérimentales



PlanetLab :

- 900 machines connectées à Internet (réseaux académiques)
- expériences sur protocoles et applications pour l'Internet du futur

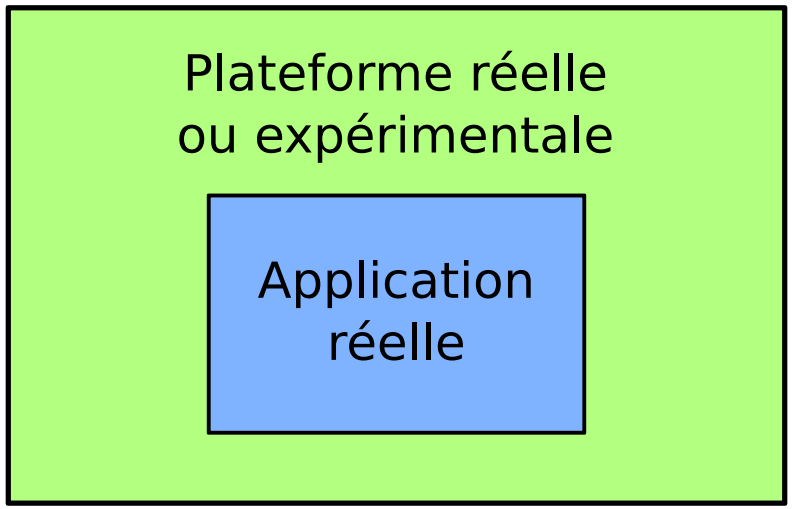
Plateformes réelles ou expérimentales



Grid'5000 :

- 2000 nœuds dans 9 sites en France
- réseau d'interconnexion dédié à 10 Gbps
- reconfigurable

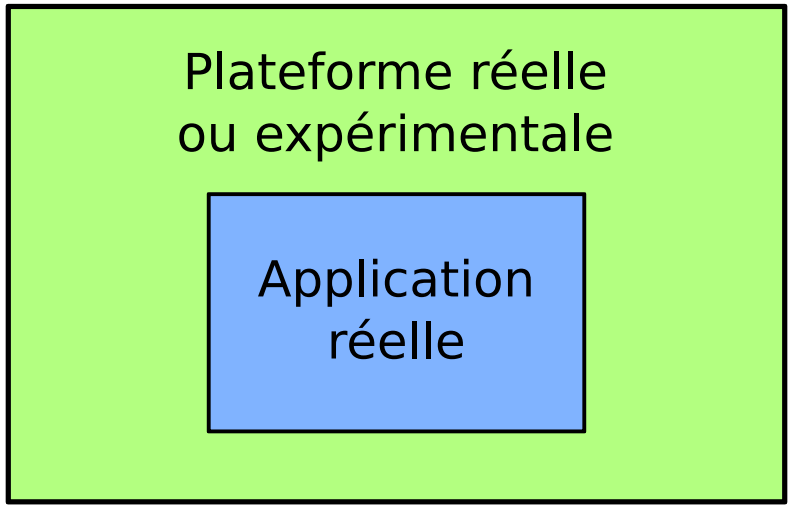
Plateformes réelles ou expérimentales



DSLLab :

- 40 machines reliées à l'ADSL
- expériences sur le *Desktop computing* et les systèmes P2P

Plateformes réelles ou expérimentales



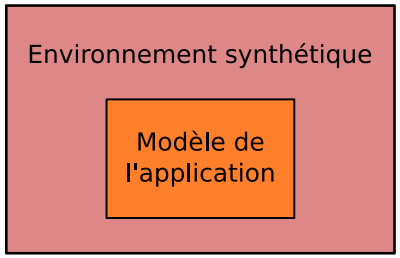
Limites des plateformes réelles et expérimentales :

- plateformes de taille limitée
- représentativité des conditions expérimentales ?
- reproductibilité des conditions expérimentales ?

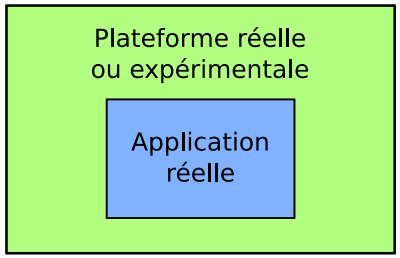
L'émulation

Une approche intermédiaire :

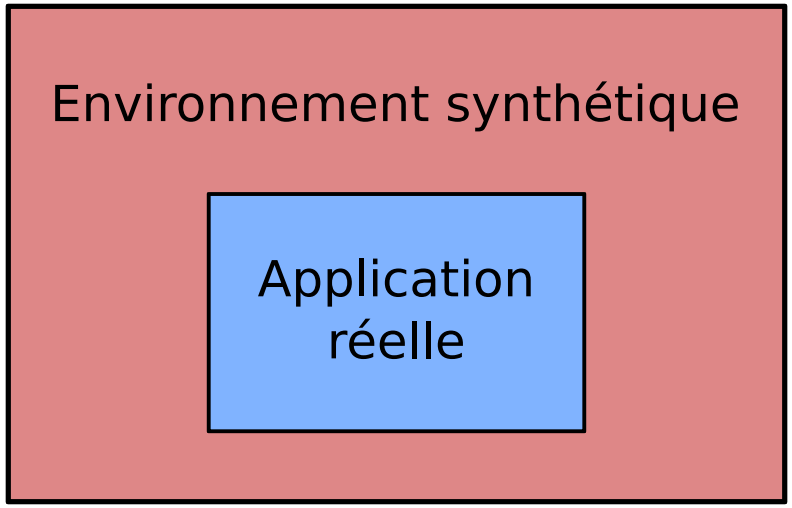
Simulation :



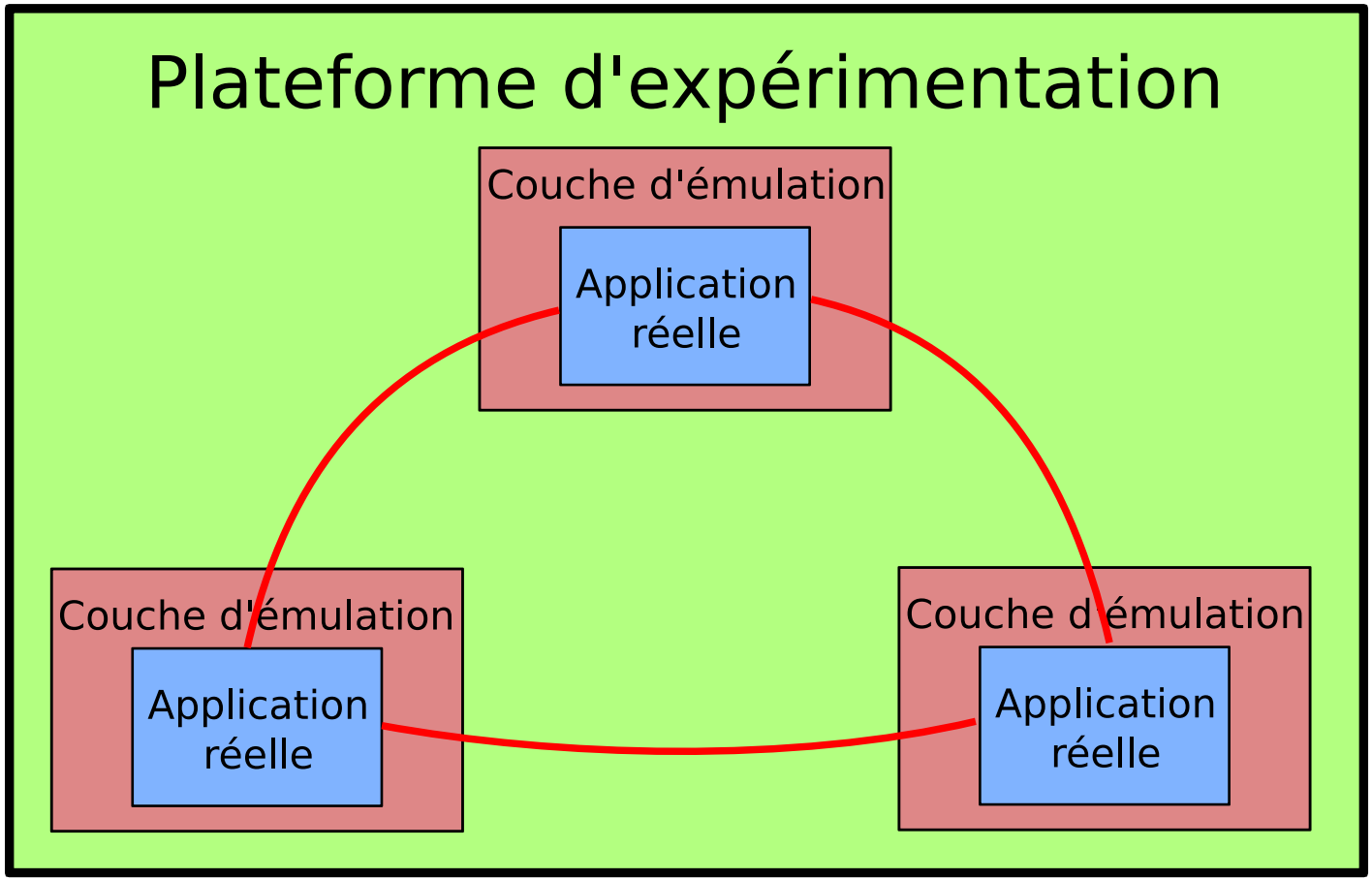
Plateformes d'expérimentation :



Émulation :



L'émulation, en pratique



Expériences à grande échelle

Simulation ou exécution sur plateformes réelles ou expérimentales pas toujours satisfaisantes

En particulier : systèmes pair-à-pair

- Systèmes de très grande taille
- Exécutés dans des environnements peu disponibles pour les expériences
- Souvent difficiles à simuler

Émulation :

- Approche intermédiaire
- Pourrait permettre des expériences à grande échelle sur des systèmes pair-à-pair

Comment concevoir un émulateur adapté à l'étude des systèmes pair-à-pair à grande échelle ?

Plan

1. Contexte de travail : expérimentation sur les systèmes distribués
2. Anatomie d'un émulateur de systèmes distribués
3. P2PLab : une plate-forme pour l'émulation des systèmes pair-à-pair
4. Conclusion et perspectives

Anatomie d'un émulateur de systèmes distribués

Plusieurs composants importants :

- Émulation du réseau
- Partage/virtualisation des ressources matérielles

Paramètres à prendre en compte :

- Passage à l'échelle
- Compromis réalisme / coût (CPU, machines)

Émulation du réseau

Objectif :
Reproduire artificiellement les caractéristiques d'un réseau

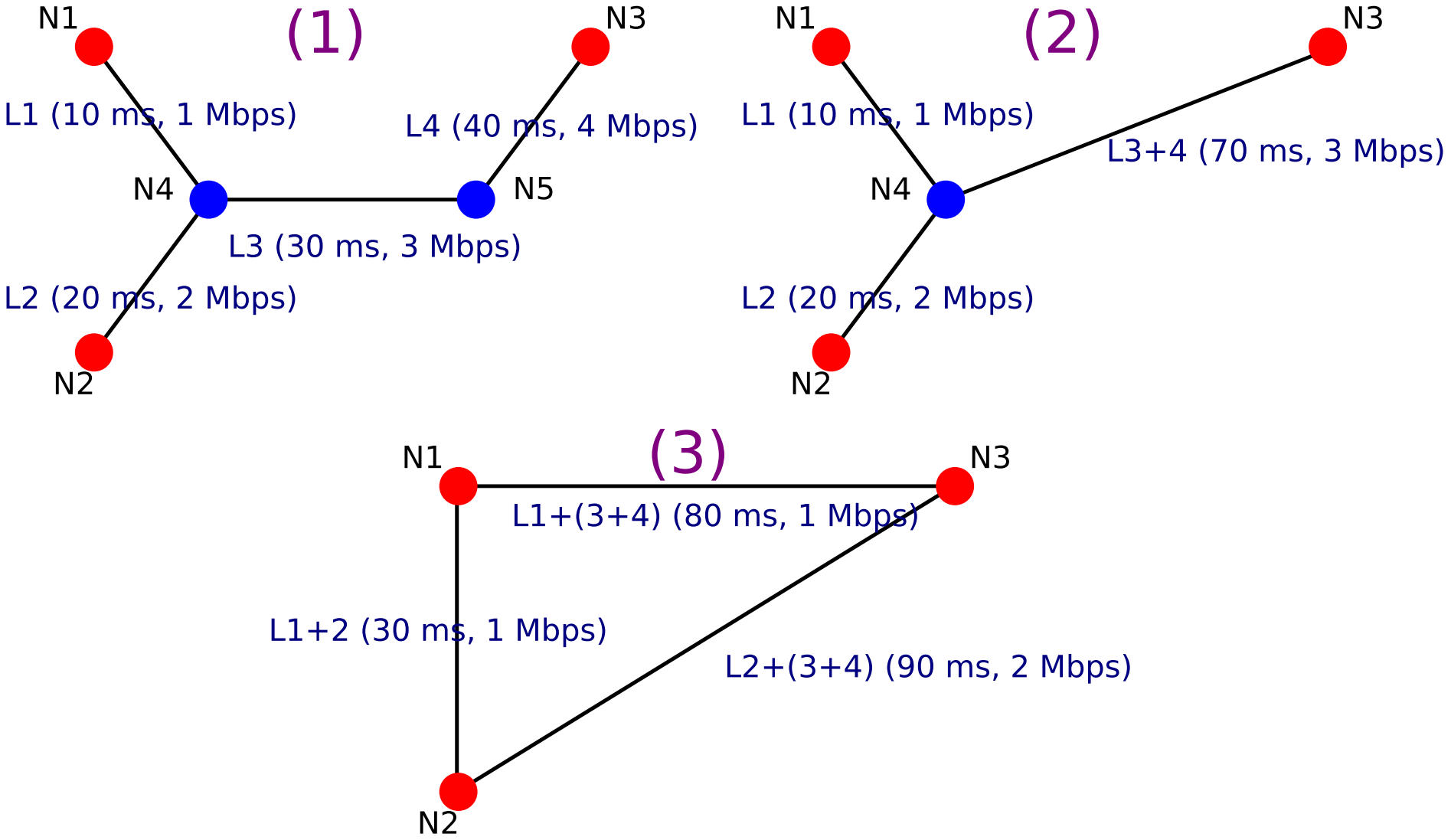
Reproduire totalement une topologie réseau complexe ?

- Possible **[Emulab, MicroGrid]** mais coûteux

Reproduction *simplifiée*
(= certaines contraintes supprimées)

- réseau en étoile **[eWAN, V-DS]**
- simplification progressive **[Modelnet : distillation]**
  $\Rightarrow$ compromis réalisme / nombre de machines utilisées

Modelnet : distillation du réseau



Plan

Émulation et passage à l'échelle

2 approches :

- plateforme de taille suffisante (difficile !)
  - $N_{noeuds} > 10000$
- partager les ressources physiques : virtualisation
  = partager une ressource entre plusieurs instances

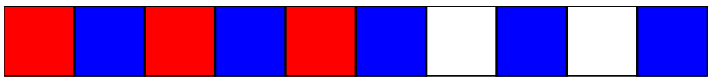
Virtualisation de ressources

Intuition :
partager entre *n* instances = rendre la ressource *n* fois plus lente pour chaque instance

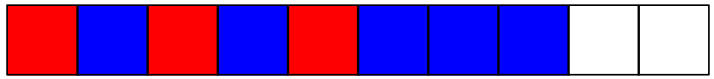
Faux si :

- le partage n'est pas équitable
- la ressource n'est pas utilisée tout le temps
  (les ordonnanceurs utilisés couramment sont *work-conserving*)

ordonnancement souhaité :



work-conserving :



Virtualisation et équité : solutions

2 approches :

- Ignorer le problème **[Emulab (récemment)]**

  Les ressources ne sont pas utilisées en permanence
  ⇒ leur partage n'affecte pas les performances

  - à vérifier en pratique pendant l'expérience

- Dilatation temporelle **[DieCast]**

  la ressource devient n fois plus lente, donc on ralentit l'écoulement du temps n fois pour que les conditions soient réalistes

  - permet des conditions irréalisables autrement
  - nécessaire d'émuler toutes les ressources

Synthèse

Solutions existantes peu adaptées :

- À l'expérimentation à grande échelle
  - Souvent : une émulation précise à petite échelle
  - Problèmes de passage à l'échelle
- Des systèmes pair-à-pair
  - Émulation précise du coeur du réseau

$\Rightarrow$ **Proposition d'une plateforme pour l'émulation à grande échelle des systèmes pair-à-pair**

Plan

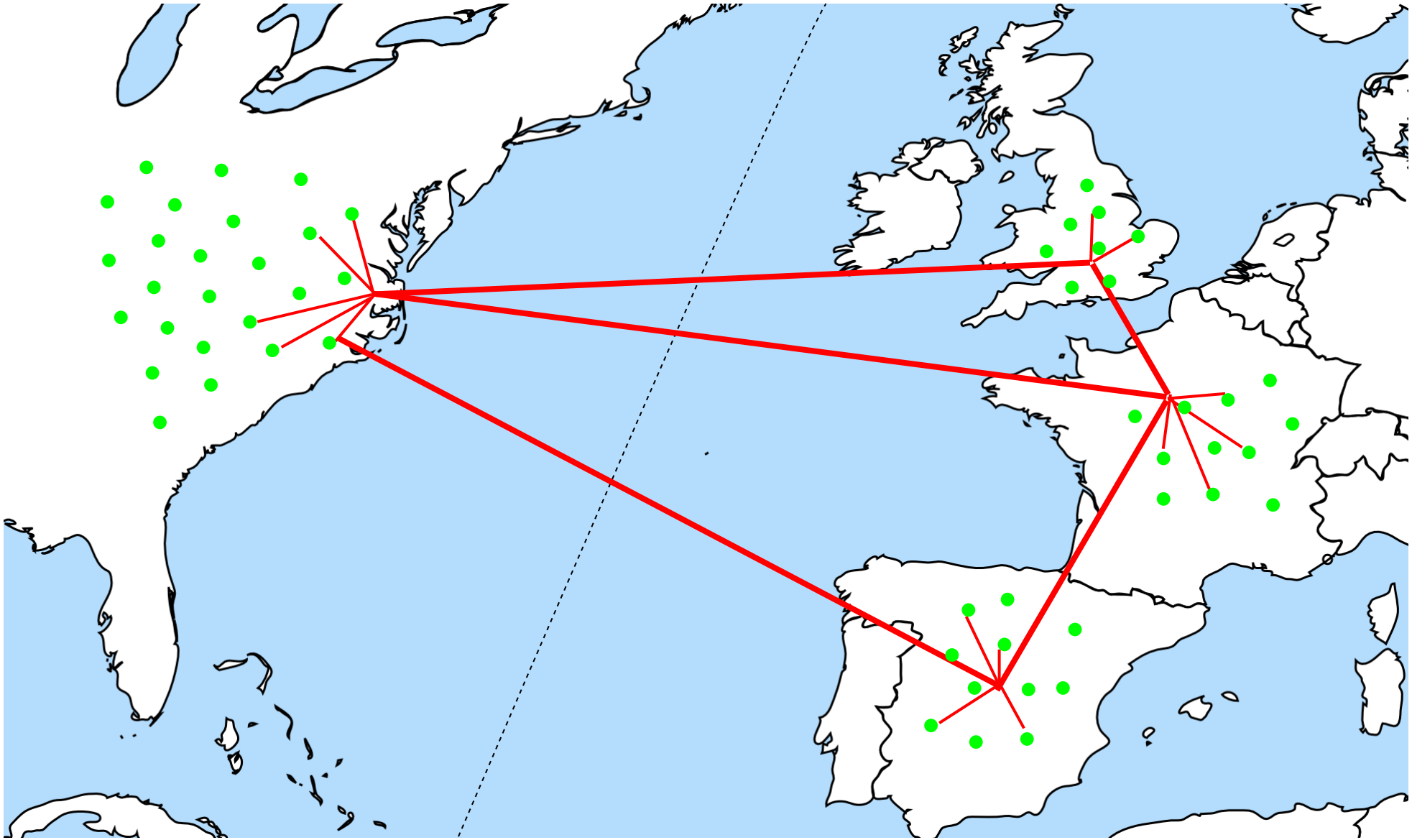
P2PLab

Plateforme pour l'émulation des systèmes pair-à-pair

- Restreint aux applications P2P centrées sur le réseau
  - exemple : diffusion de fichiers

  Cette restriction permet des compromis
- Objectifs :
  - Passage à l'échelle : plus de 10000 nœuds
  - Générique (pas spécifique à un intergiciel)
  - Simplicité

P2PLab - émulation du réseau



P2PLab - émulation décentralisée

Émulation sur les machines exécutant l'application

- Pas de machines dédiés à l'émulation réseau
- Utilisation d'un émulateur déjà disponible

$\Rightarrow$ Facteur limitant le passage à l'échelle :
nombre de règles dans l'émulateur sur chaque machine

Proposition : groupes de nœuds hiérarchiques
$\Rightarrow$ Règles par groupe, pas par nœud

P2PLab - émulation de la congestion dans le cœur

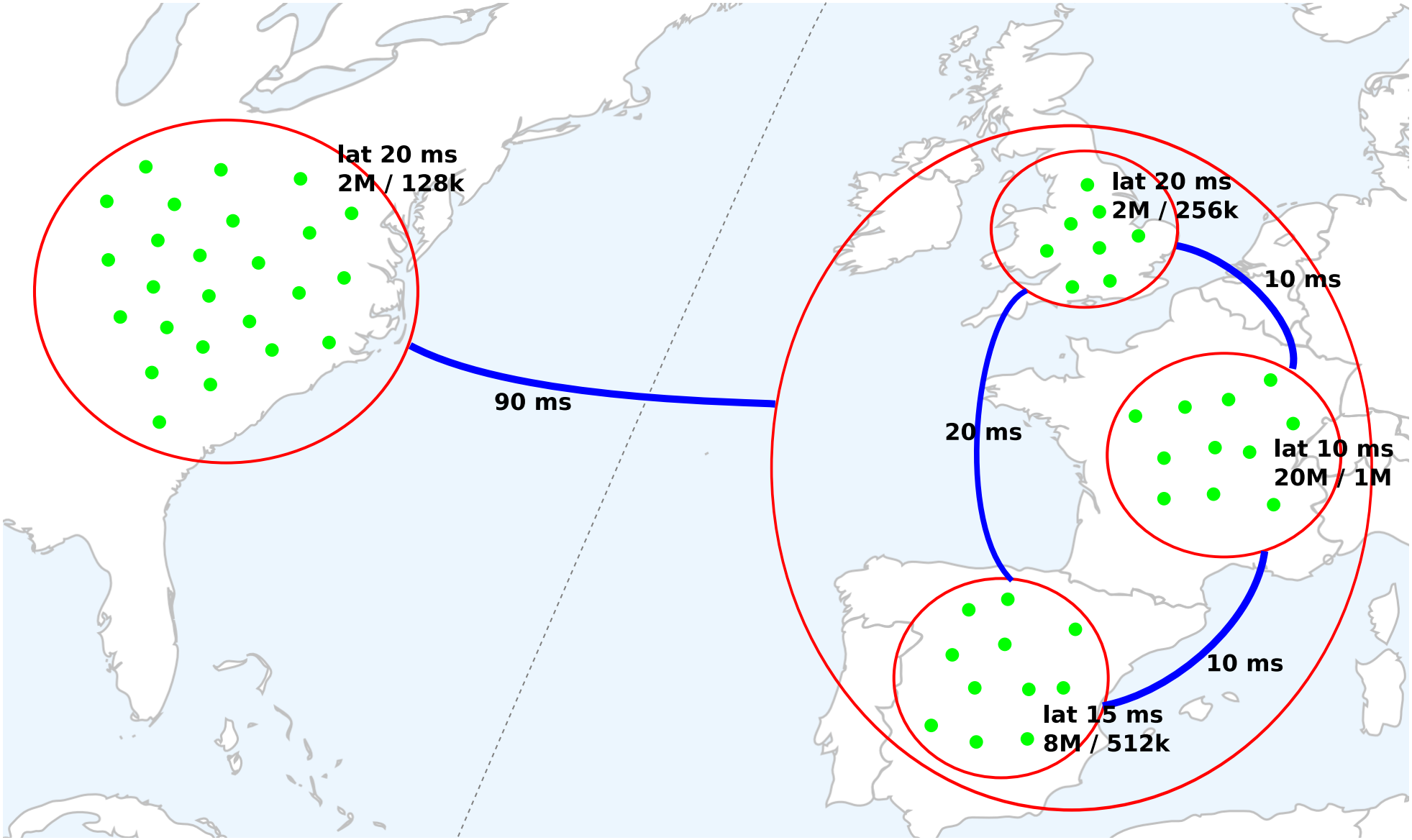
Avec nos applications cibles :
Goulot d'étranglement : lien utilisateur - FAI

Hypothèse : la congestion dans le cœur du réseau n'influence pas significativement les performances des applications P2P

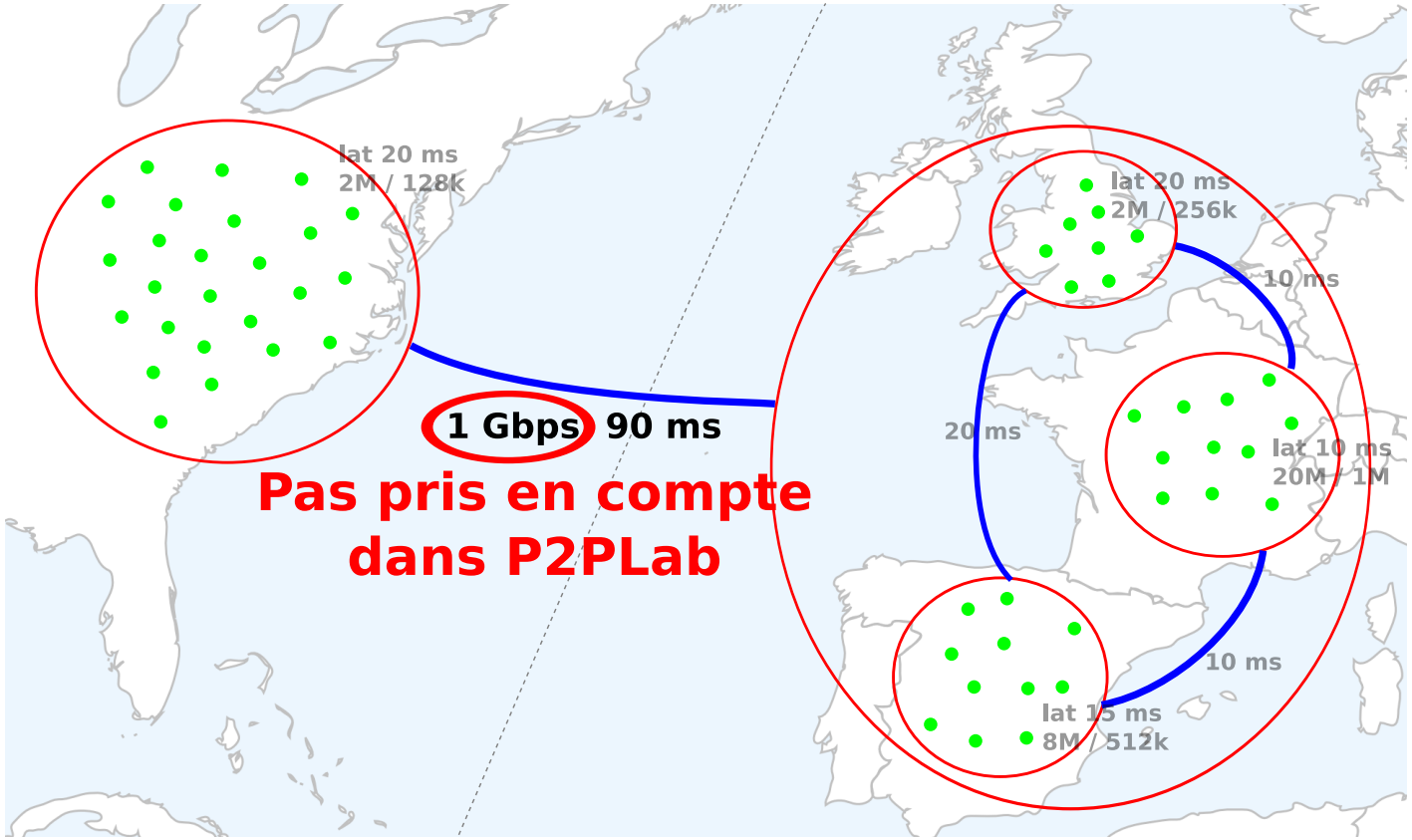
$\Rightarrow$ Dans le cœur du réseau :

- Uniquement émulation de la latence
- Pas d'émulation de la bande passante

P2PLab - topologie du réseau



P2Plab - limites du modèle de topologie

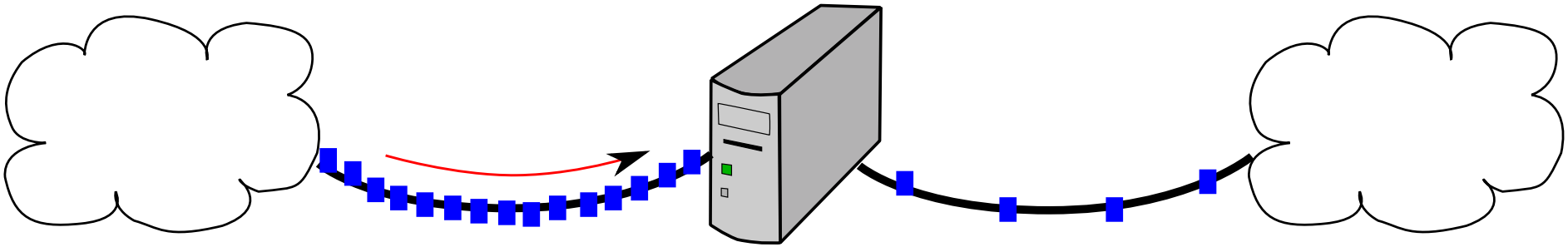


Dans le manuscrit :

Autre modèle d'émulation de topologies, permettant d'émuler correctement une bande passante limitée dans des liens intérieurs (mais avec moins bon passage à l'échelle)

Plan

Quel émulateur de liens réseaux ?



Principe :
Retarder/rejeter les paquets entrants/sortants d'une machine en fonction de paramètres (débit, latence, taux de perte, ...)

- De nombreux émulateurs réseaux disponibles et utilisés
  - Dummynet (FreeBSD), NISTNet et TC/Netem (Linux), ...
- Aussi utilisés dans des émulateurs de topologies
- Mais jamais évalués ou comparés

Lequel choisir pour P2PLab ?

- Fonctionnalités ?
- Précision / performances ?

Fonctionnalités

Tous les émulateurs :
émulation de latence, limitation de bande passante, pertes

Traitement des paquets entrants et sortants :

- Dummynet : oui
- NISTNet : entrants seulement
- TC/Netem : sortants seulement*

Réordonnancement, duplication, corruption :

- Dummynet : non
- NISTNet et TC/Netem : oui

Précision de l'émulation

Dépend de la source de temps utilisée pour programmer l'envoi des paquets :

- Dummynet et TC/Netem : interruptions horloge du système
  Depuis Linux 2.6.24 pour TC/Netem : *High Resolution Timers*
- NISTNet : horloge différente

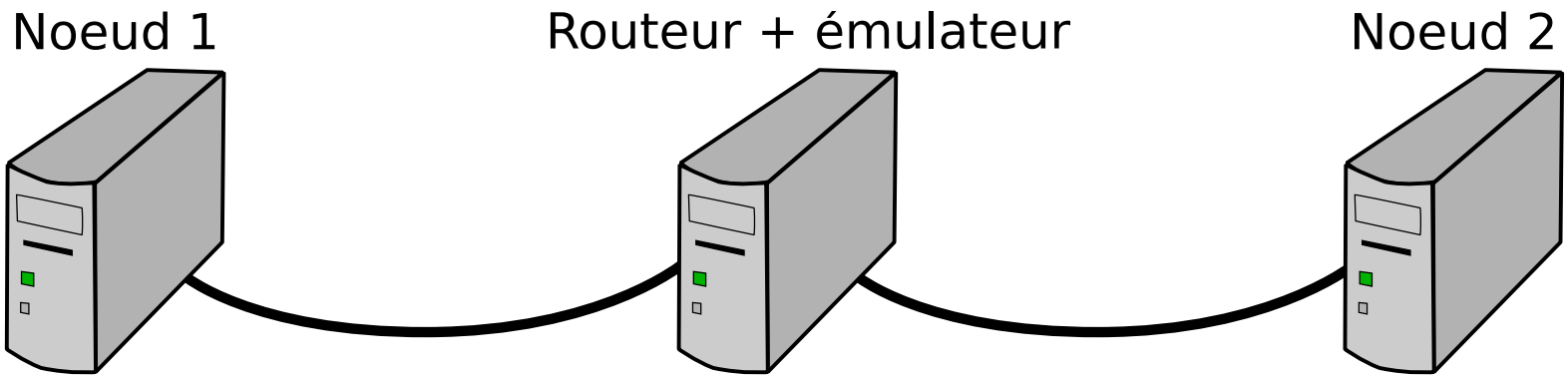
Si manque de précision :

- paquets relâchés trop tard / trop tôt
- paquets relâchés en rafale
  - non réaliste
  - peut avoir des incidences sur les résultats :
    - au niveau réseau, paquets rejetés par des nœuds intermédiaires
    - au niveau applicatif, dimensionnement des tampons

Précision de l'émulation - expérience

Expérience : évolution de la latence au cours du temps mesurée avec des pings à une fréquence très élevée (> 10 kHz)
Latence paramétrée dans l'émulateur : 10 ms

Configuration expérimentale :



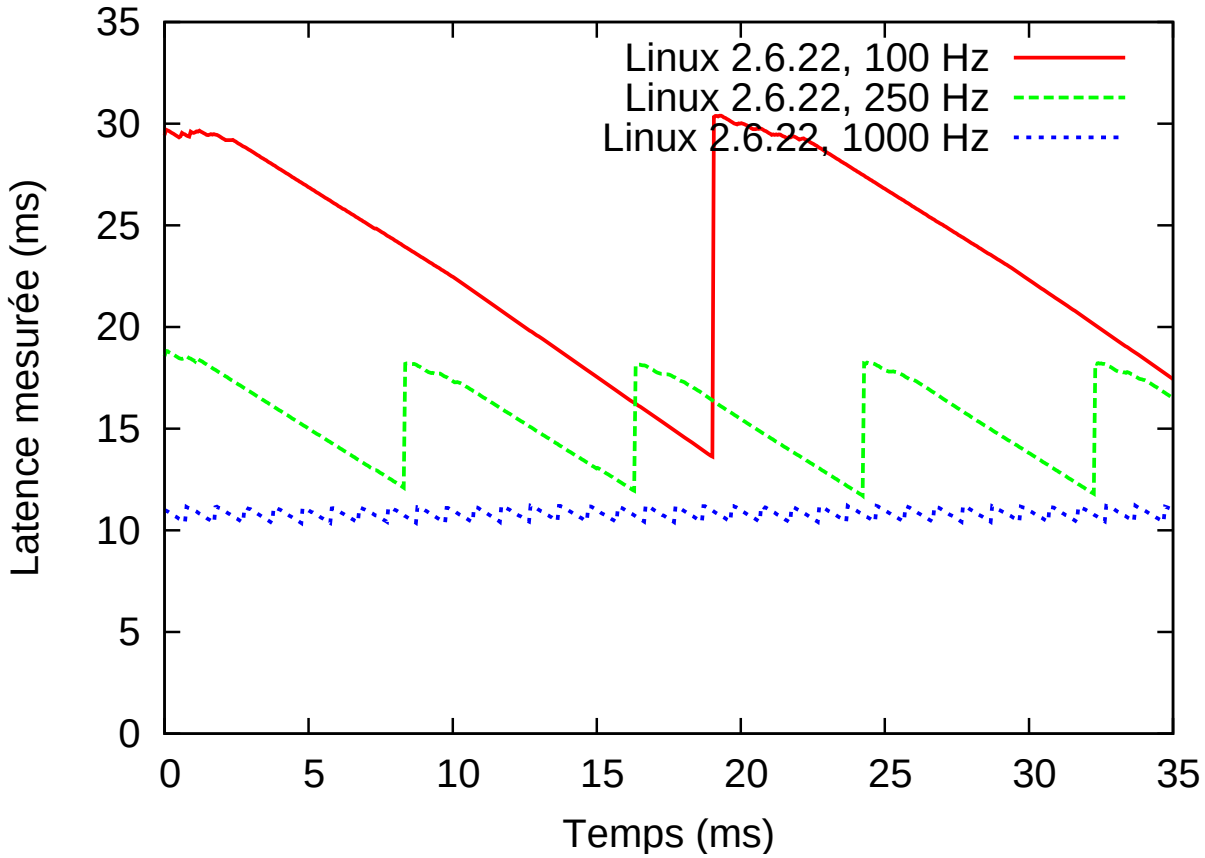
- 3 nœuds netgdx (2 cartes Gigabit Ethernet)
- Paramètres d'émulation appliqués sur le routeur

Précision de l'émulation - expérience

De nombreuses configurations à tester :

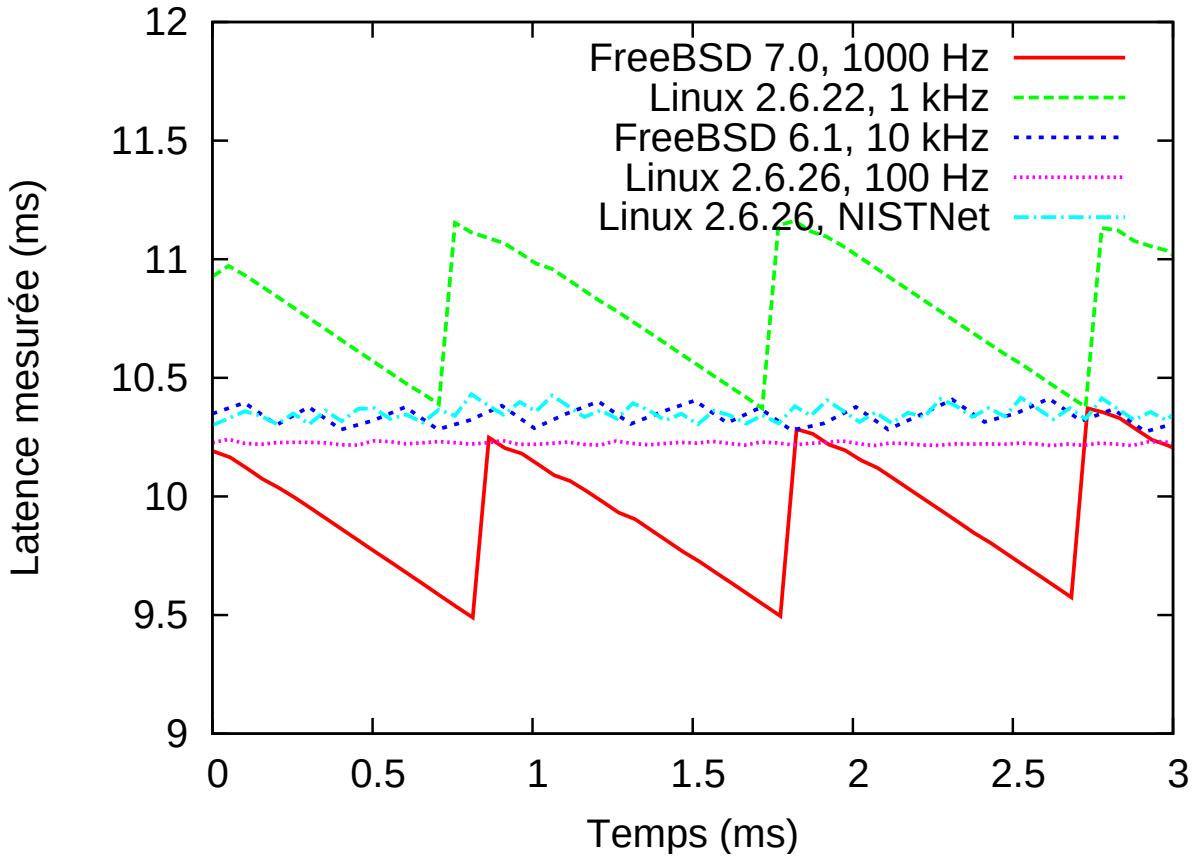
- Linux < 2.6.24, différentes fréquences d'horloge
- Linux $\geq$ 2.6.24 (*High Resolution Timers)*
- Linux + NISTNet
- FreeBSD, différentes fréquences d'horloge

Précision de l'émulation - résultats



- La latence varie au cours du temps
- Augmenter la fréquence de l'horloge améliore la précision
- Résultats similaires avec Dummynet (FreeBSD)

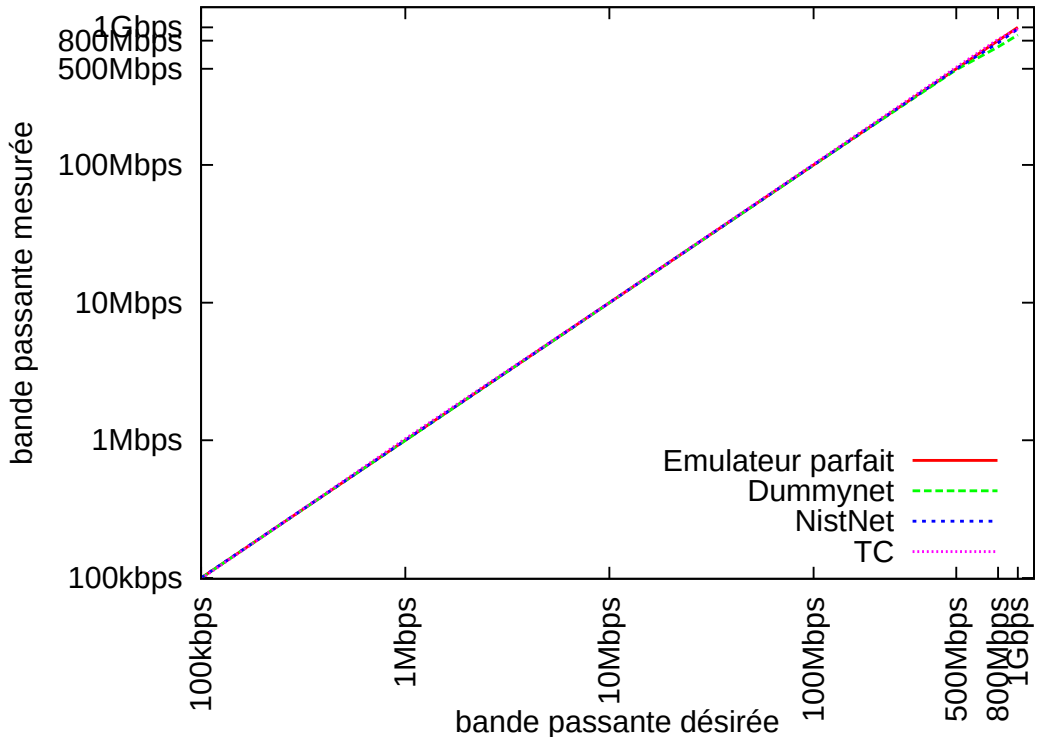
Précision de l'émulation - résultats



- Meilleure précision obtenue avec FreeBSD 6.1 (10 kHz), Linux 2.6.26 et NISTNet
- Mais attention au surcoût causé par le traitement des interruptions horloge

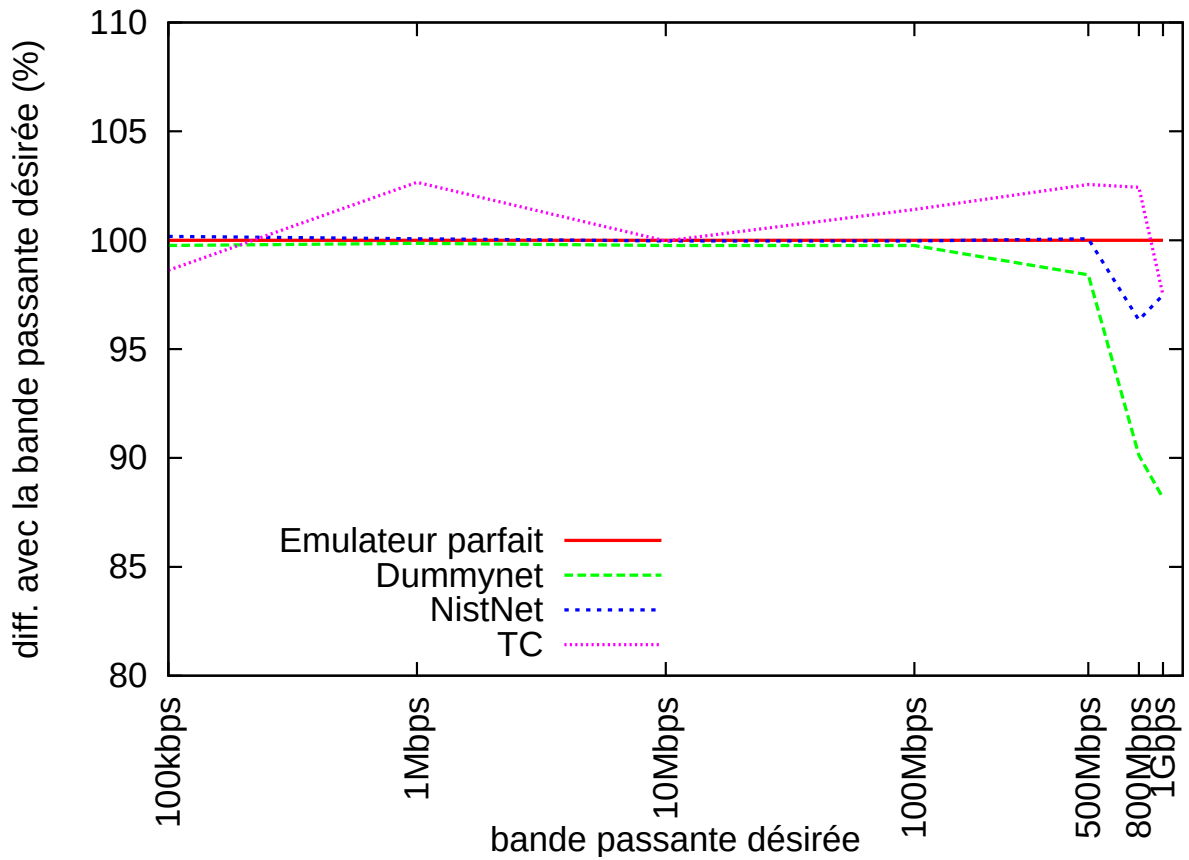
Limitation de bande passante

- NISTNet et Dummynet : simple calcul de retard
- TC : algorithme de qualité de service
  - plus difficile à paramétrer



⇒ Tous corrects dans l'ensemble

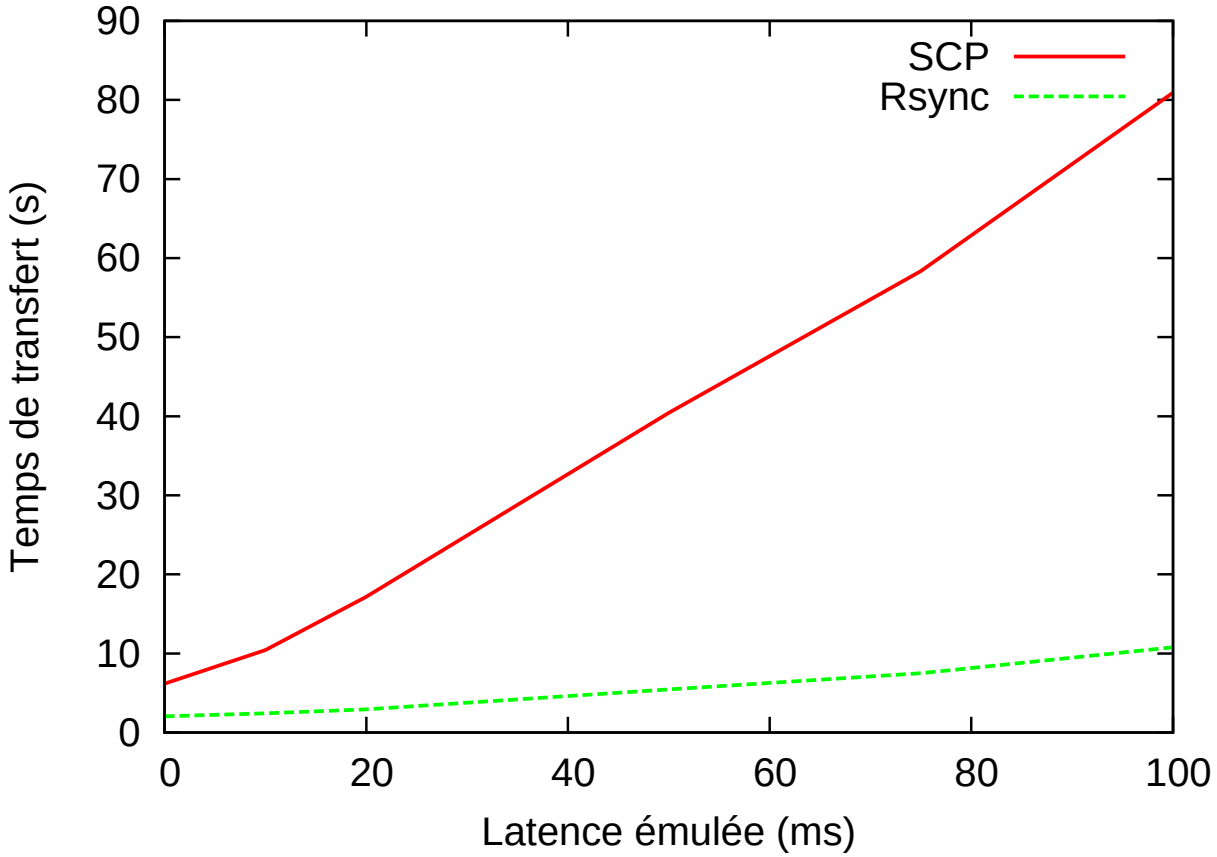
Limitation de bande passante



Dummynet ne permet pas d'émuler des débits importants

Illustration : SCP vs Rsync

Transfert de 120 fichiers (total : 2.1 Mo) avec SCP et Rsync
Bande passante fixée à 10 Mbps avec TC



Permettent d'obtenir des résultats quantitatifs

Émulateurs de liens réseaux - synthèse

3 configurations utilisables :

- Linux $\geq$ 2.6.24 avec Netem
  - HR timers, mais problème du filtrage en entrée
- Linux + NISTNet
  - pas de filtrage en sortie, attention au surcoût
- FreeBSD 6 + Dummynet, avec horloge à 10 kHz
  - attention au surcoût

Dans P2PLab : choix de FreeBSD 6 + Dummynet

Attention :

- Dangereux de considérer les émulateurs comme des boîtes noires : important de vérifier la bonne implémentation des paramètres

Plan

Virtualisation dans P2PLab

Applis P2P généralement auto-contenues :
$\Rightarrow$ Permet de virtualiser au niveau des processus

- Uniquement l'identité réseau du processus
- Transparent pour l'application (modification de la *libc*)

Partage du processeur avec l'ordonnanceur de FreeBSD

- Équité du partage validée *(cf manuscrit)*

Pas de dilatation temporelle

- On considère que le partage des ressources (CPU, mémoire, disque) n'affecte pas les résultats des expériences
  - On le vérifie avant/pendant l'expérience

Plan

P2PLab - Validation expérimentale

Questions :

- La virtualisation sans dilatation temporelle est-elle une méthode réaliste ?
  - Quel rapport de repliement (nb machines virt. / nb machines phys.) peut-on atteindre ?
- Combien de nœuds virtuels peut-on émuler avec P2PLab ?
- Peut-on expérimenter avec des implémentations différentes du même protocole ?

$\Rightarrow$ Réponses pas absolues : fonction de l'expérience.
Ici : expériences autour de BitTorrent

Plateforme : GridExplorer (jusqu'à 163 nœuds)

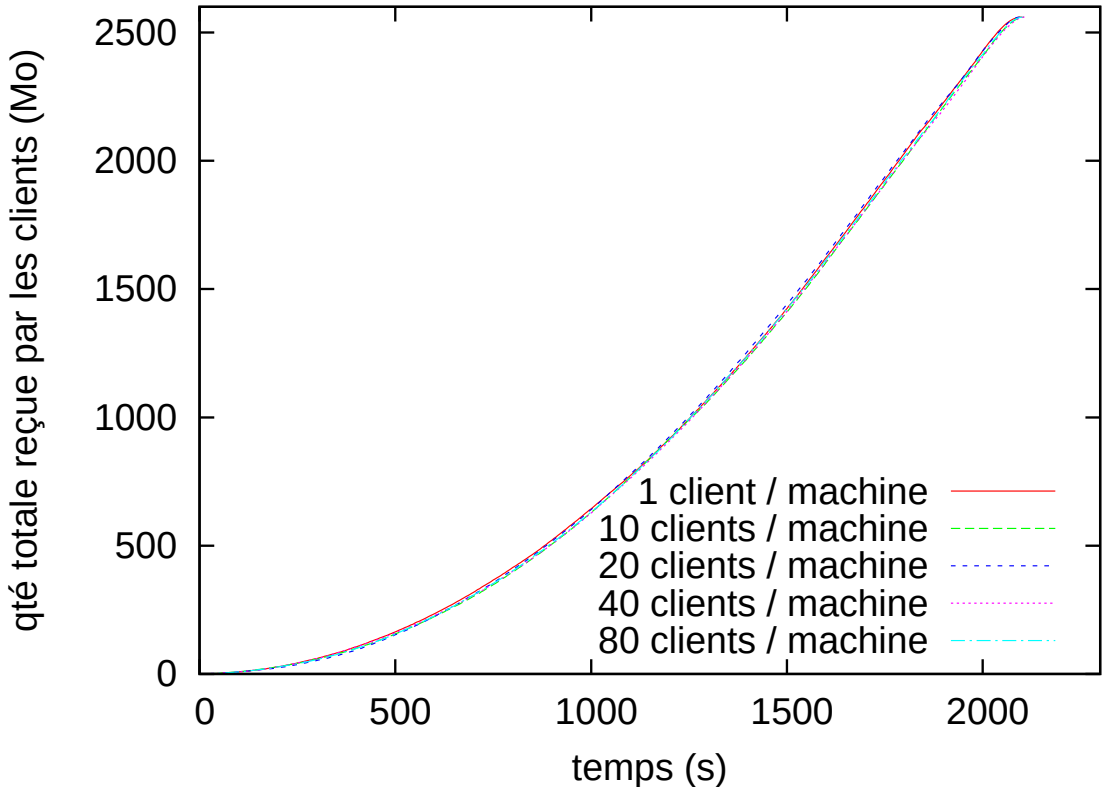
Rapport de repliement

160 nœuds

1 nœud par machine
puis 10, 20, 40, 80

paramètres réseau :
2 Mbps / 128 kbps,
latence 30 ms

(réseau pas saturé
même avec 80 nœuds
par machine)



⇒ Aucune influence visible du repliement sur ces résultats
Premier facteur limitant : utilisation mémoire (*swap*)
(nœuds avec 2 Go de RAM)

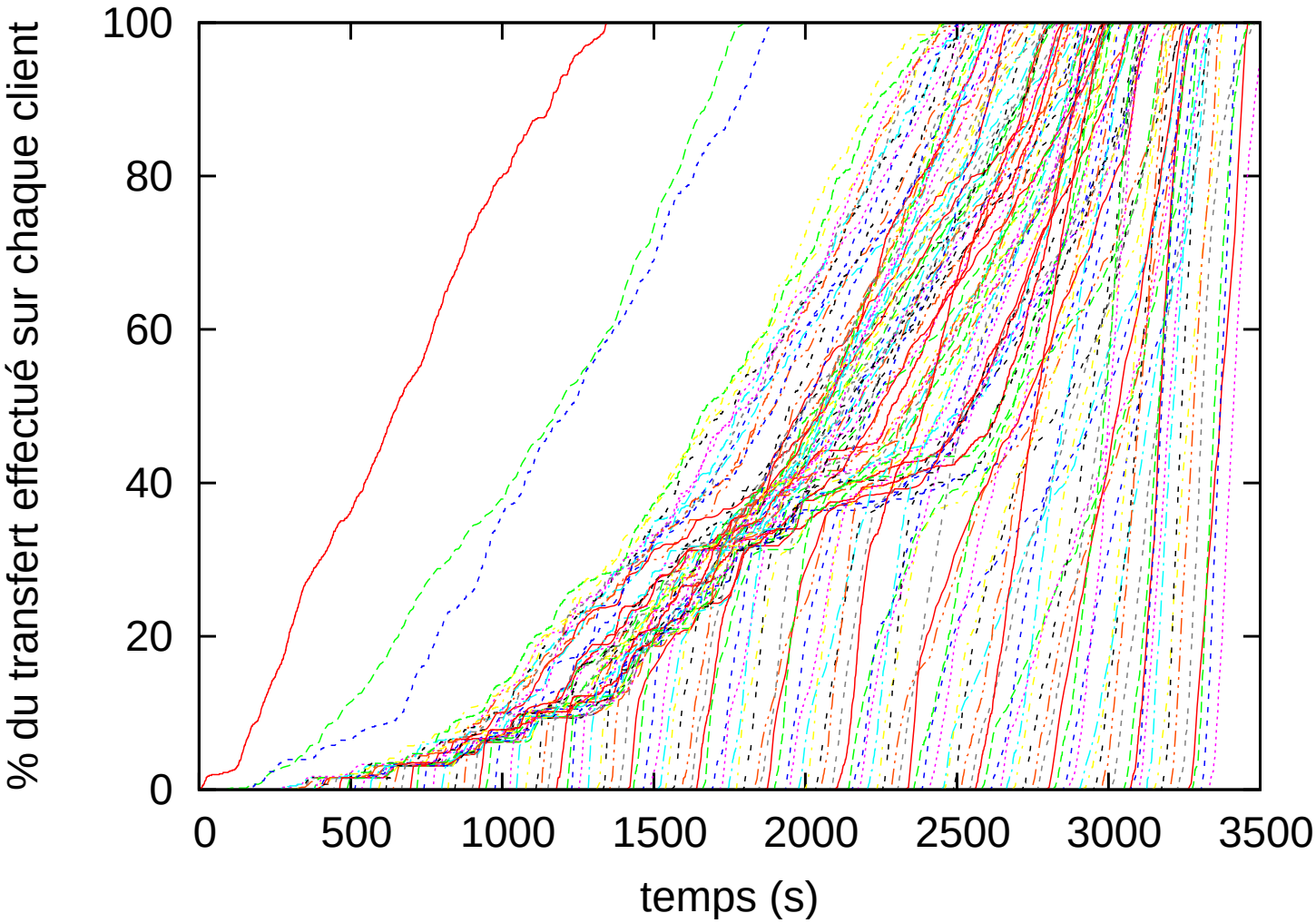
Passage à l'échelle

13040 nœuds répartis sur 163 machines (80 NV / MP)

2 types de clients :

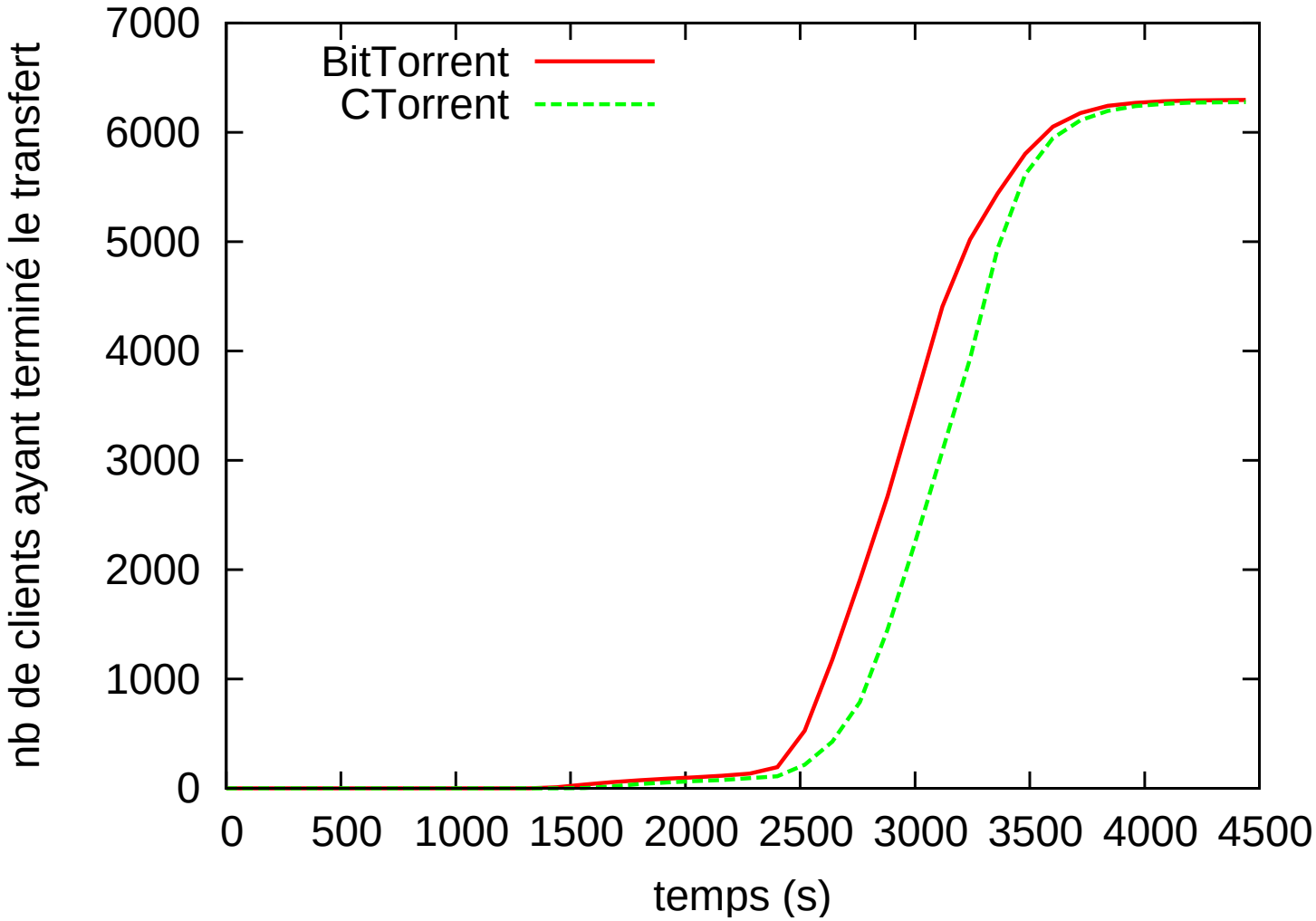
- 50% des clients utilisant BitTorrent (Python, client "officiel")
- 50% des clients utilisant CTorrent (C)

Passage à l'échelle



⇒ Obtention d'une vue complète d'un transfert sur 13k nœuds

Passage à l'échelle



⇒ Comparaison quantitative de deux implémentations différentes

Conclusion

Conception et développement d'un émulateur pour les systèmes pair-à-pair

- Approche incrémentale
- Émulation réseau distribuée, adaptée au domaine d'étude
- Virtualisation au niveau processus
- Validation des composants de l'émulateur
  - Étude comparative des émulateurs de liens réseaux, Virtualisation
- Validation des caractéristiques de l'émulateur
  - Rapport de repliement, Passage à l'échelle

Perspectives

Amélioration de la remontée d'information (sondes système)
$\Rightarrow$ Détecter plus facilement la présence de biais

Travail d'ingénierie autour de P2PLab :
Vers un émulateur intégré à Grid'5000 ?

Convergence avec d'autres travaux
$\Rightarrow$ Émulateur générique pour les systèmes distribués

- Intégration d'injecteurs de fautes, de générateurs de trafic
- Virtualisation légère ou lourde **[V-DS]**
- Différents modèles de topologie

Réévaluer les différentes solutions :
Domaine en constante évolution

Contributions à l'expérimentation sur les systèmes distribués de grande taille

Lucas Nussbaum

Soutenance de thèse
4 décembre 2008