Déploiement et contrôle d'applications parallèles sur grappes de grandes tailles

Cyrille Martin

Directrice : Brigitte Plateau

Co-encadrant : Jacques Briat


Laboratoire ID-IMAG

http://www-id.imag.fr


BULL

http://www.bull.fr

Projet Lips Action DYADE BULL-INRIA

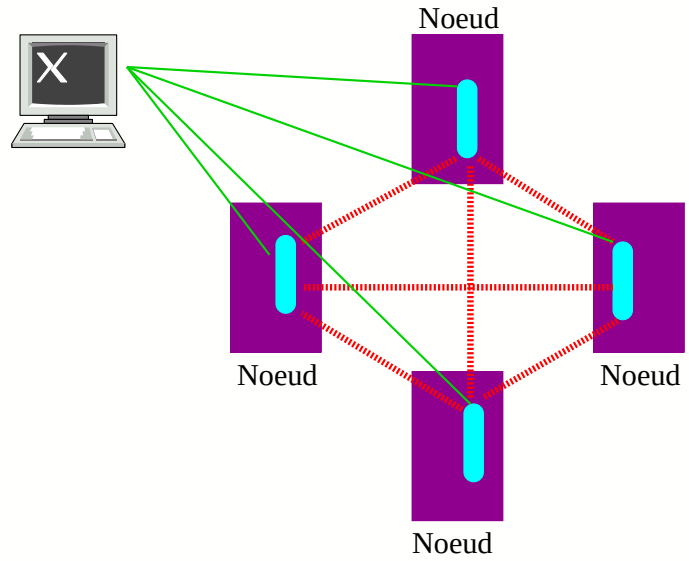
Pascale Rosse (BULL)

Contexte : Grappes de calcul

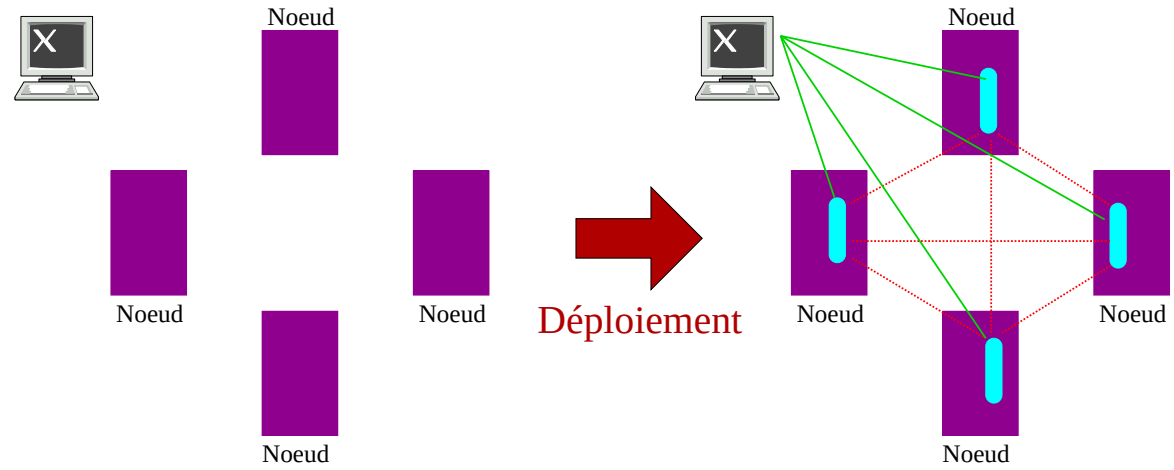
- Machines usuelles:
  Stations de travail
  Serveurs SMP
- Réseaux d'interconnexion :
  + / - rapides

Programmes parallèles

- Ensemble distribué de programmes (processus)
- Collaboration : échange de données via réseau



Déploiement



- Création des processus
- Interconnexion logique
- Services: communication, contrôle

Problématique

- Déploiement efficace d'applications sur des grappes de grandes tailles ($\sim$1000 noeuds)
- Deux domaines d'application :
  - Applications de calcul parallèle

    - Tps d'exécution (heures, jours) $>>$ Tps de déploiement (secondes)

    - *phase de développement*
  - Outils d'exploitation

    - Tps de déploiement (secondes) $>>$ Tps d'exécution (milli-secondes)

    - *opérations de maintenance : robustesse*

Performances $\Rightarrow$ Interactivité

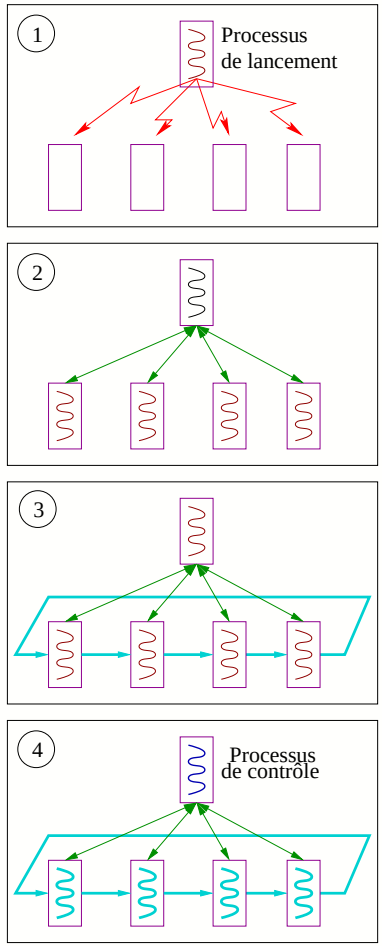
Plan

- Contexte & Problématique
- Etat de l’art
- Notre approche
  - Concepts algorithmes
  - Bibliothèque Taktuk
- Projets reposant sur Taktuk
- Conclusion et perspectives

Etat de l'art

- Déploiement = Créer un ensemble distribué de processus
- Brique de base: Création distante d'un processus
  - Protocoles standards : rsh, ssh
    - ⊕ Répandus
    - ⊕ Sécurité
    - ⊖ Coût : rsh 100 ms, ssh 350 ms
  - Protocoles dédiés :
    - ⊕ Faible coût : $\sim$ latence réseau
    - ⊕ Niveau de sécurité adapté
    - ⊖ Dédiés (pas de solution générique)

Approche naïve de déploiement



1. création des processus
2. interconnexion de contrôle
3. interconnexion applicative
4. programme parallèle

Limites de cette approche "simple"

- Approche séquentielle $\Rightarrow$ Passage à l'échelle limité
- Tps total = N $\times$ tps opération unitaire

$\longrightarrow$ Coût linéaire

Exemple:

ssh 100 noeuds : $\sim$ 35 secondes

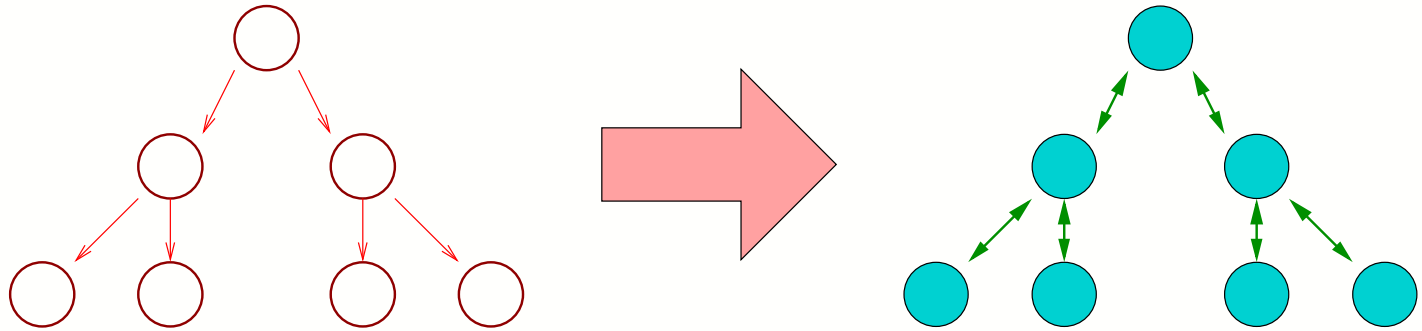
$\longrightarrow$ ls sur grappe : 35 secondes

Améliorations 1/3

- Réseaux rapides : Myrinet, Quadrics
  Cplant, Storm
  - ⊕ très performant (opérateurs matériels de diffusion)
  - ⊖ dépendant du matériel
- Logiciel : protocoles dédiés
  rexec
  - ⊕ performant
  - ⊖ implantation spécifique
  - ⊖ centralisé (coût linéaire)

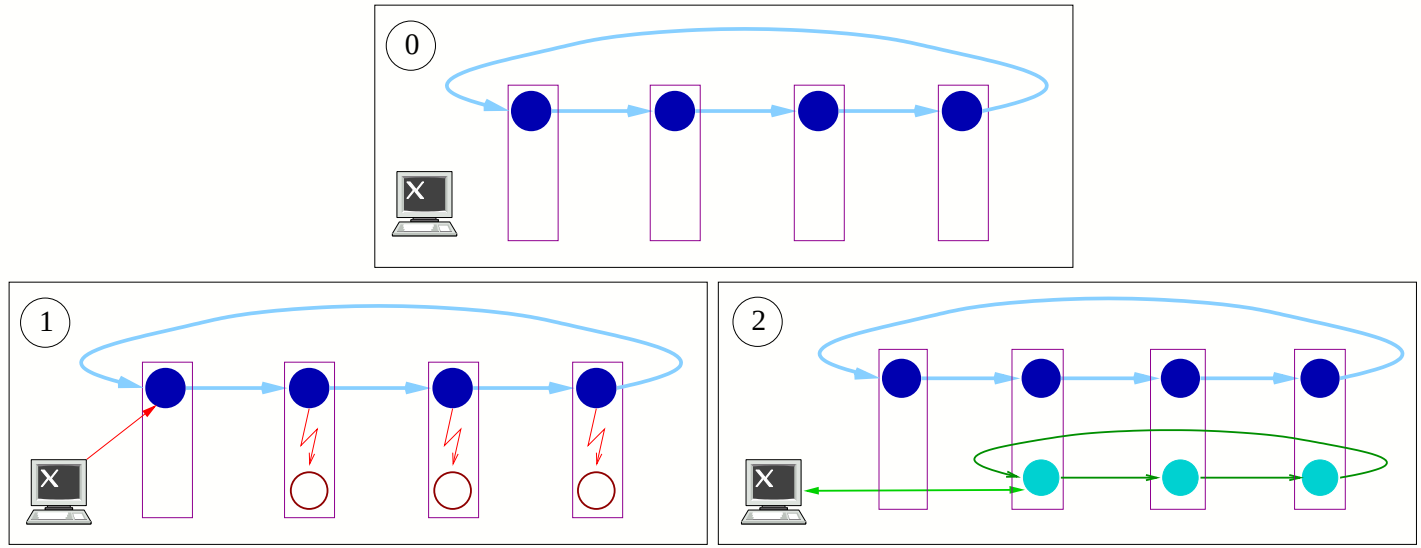
Améliorations 2/3

- Diviser pour régner :
  distribution récursive en arbre
  $\Rightarrow$ coût logarithmique
  Ksix



Améliorations 3/3

- Factorisation : *mpich, mpilam, score*
  - ⊕ performant
  - ⊕ générique, niveau utilisateur
  - ⊖ robustesse de l'implantation (mode connecté)



Bilan sur les outils existants

- Grande diversité d'outils (1 par centre de calcul)
- Approches dédiées : (outils d'exploitations)

  - hautes performances (interactivité), passage à l'échelle (milliers de noeuds)

  - déploiement d'applications parallèles (trivial avec ces outils)
- Approches génériques : (calcul parallèle)

  - passage à l'échelle $\rightarrow$ factorisation

  - outils d'exploitation (robustesse, peu utilisé)

Bilan sur les méthodes existantes

Deux méthodes :

- Améliorations "locales"
  - matériel
  - logiciel
- Améliorations "globales"
  - distribution
  - "factorisation"

Implantations génériques existantes :
Pas de combinaison de ces méthodes

Plan

- Contexte & Problématique
- Etat de l'art
- Notre approche
  - Concepts algorithmes
  - Bibliothèque Taktuk
- Projets reposant sur Taktuk
- Conclusion et perspectives

Parallélisation "globale"

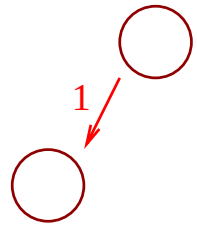
- Paralléliser le déploiement : protocoles standards
- Distribution récursive :

Parallélisation "globale"

- Paralléliser le déploiement : protocoles standards
- Distribution récursive :

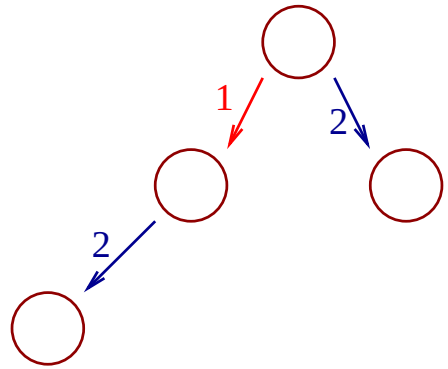
Parallélisation "globale"

- Paralléliser le déploiement : protocoles standards
- Distribution récursive :



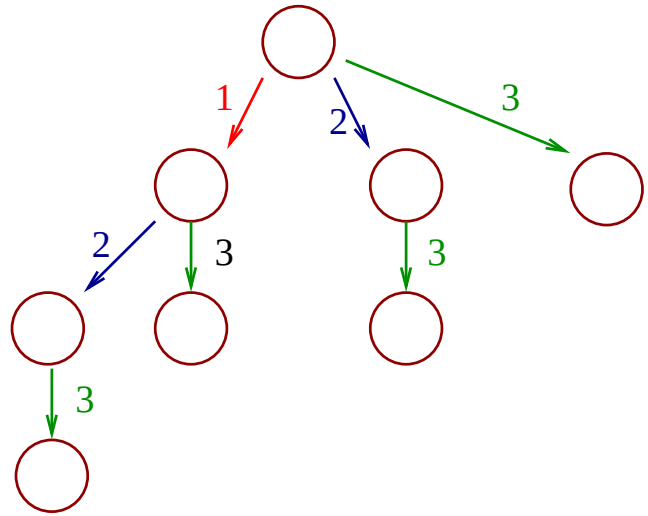
Parallélisation "globale"

- Paralléliser le déploiement : protocoles standards
- Distribution récursive :



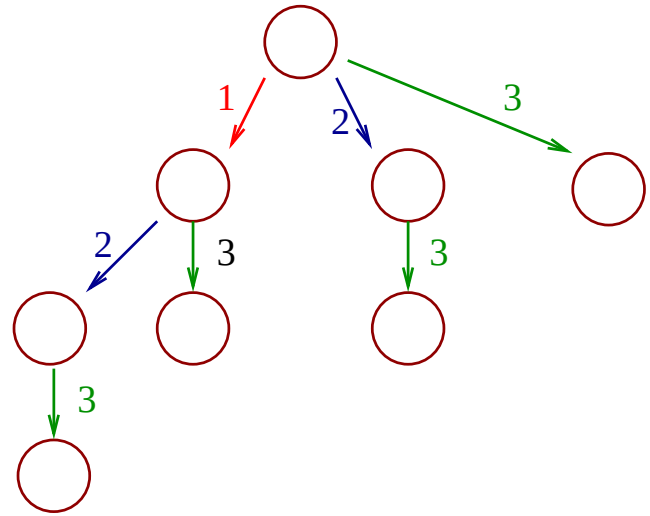
Parallélisation "globale"

- Paralléliser le déploiement : protocoles standards
- Distribution récursive :



Parallélisation "globale"

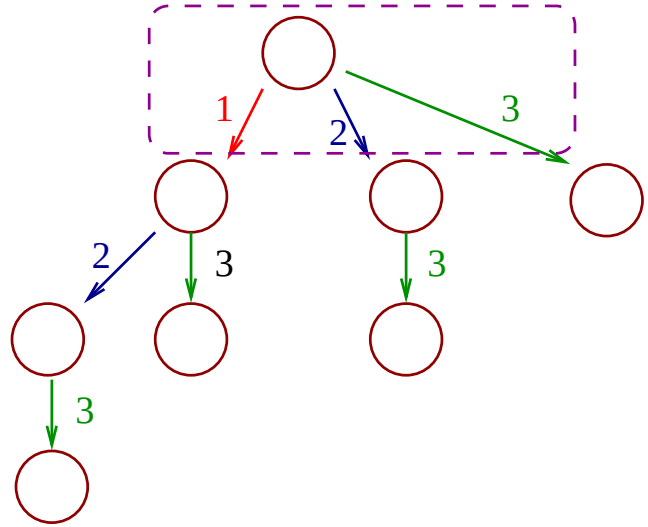
- Paralléliser le déploiement : protocoles standards
- Distribution récursive :



Coût logarithmique ⇒ Passage à l'échelle

Parallélisation "globale"

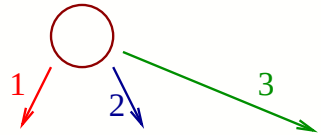
- Paralléliser le déploiement : protocoles standards
- Distribution récursive :



Coût logarithmique ⇒ Passage à l'échelle

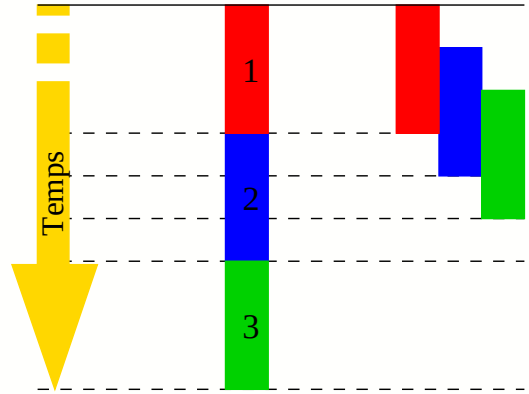
Parallélisation "locale"

- Opérations de base:

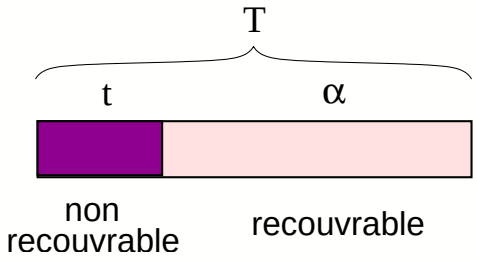


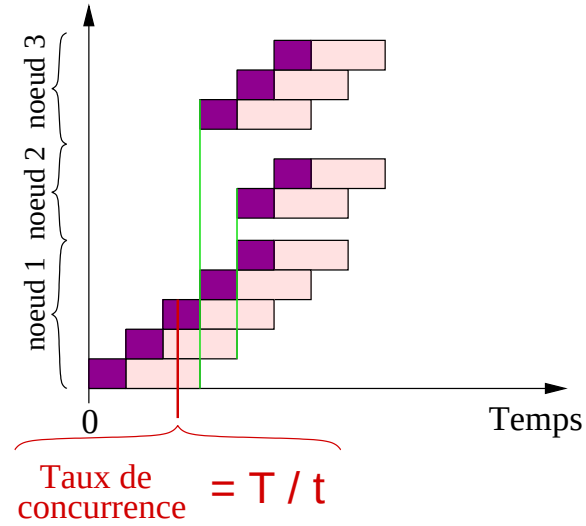
Un noeud initie N processus distants.

- Optimisation locale:
  Appels concurrents : recouvrement



Appels concurrents et distribution





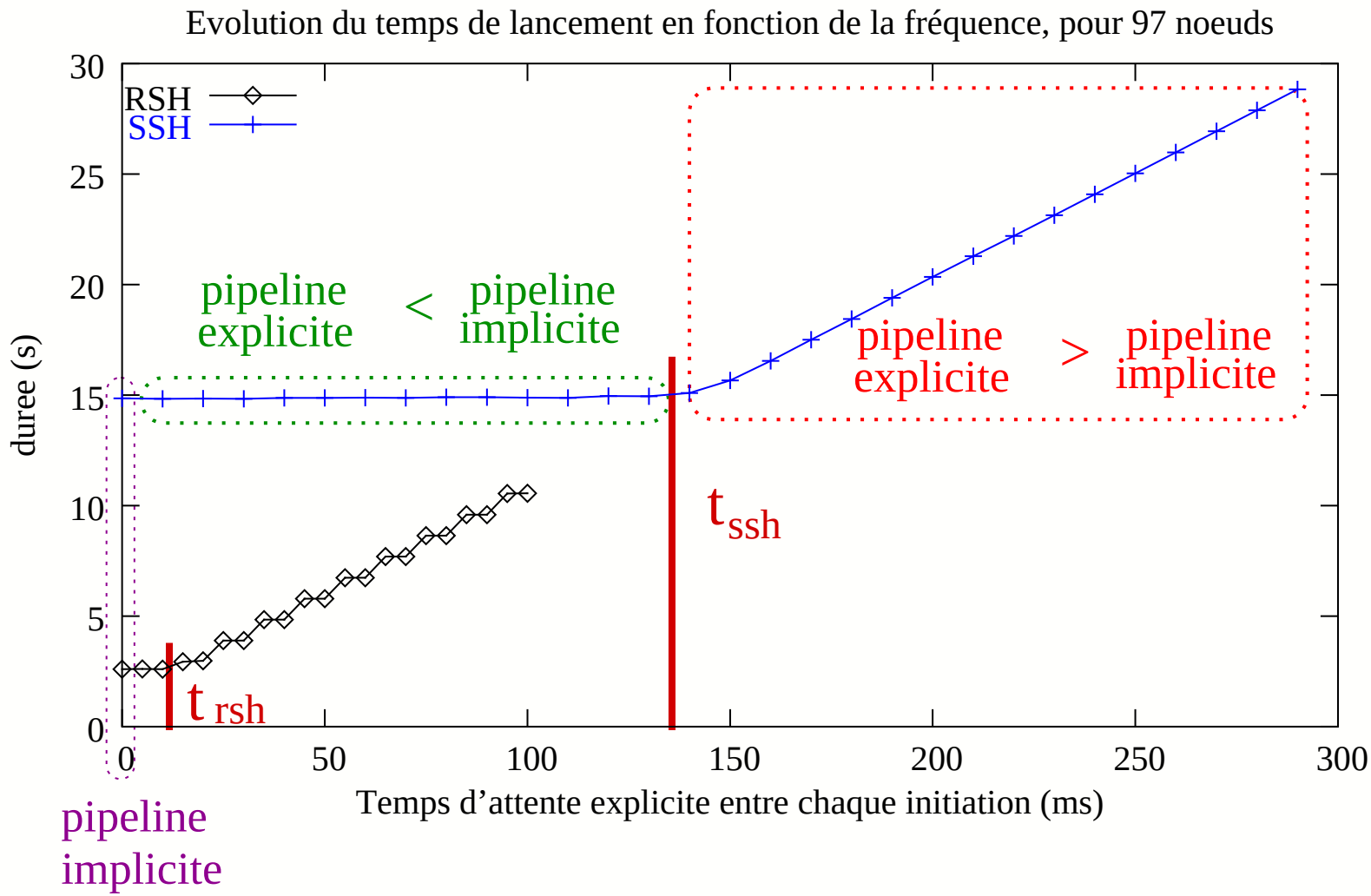
- Répartition effi cace :

  Déterminer le taux de concurrence
- Sur-estimation :

  ↘ performances (surcharge "locale")
- Sous-estimation :

  ↘ performances (capacité "locale" mal exploitée)

Mesure du taux de concurrence



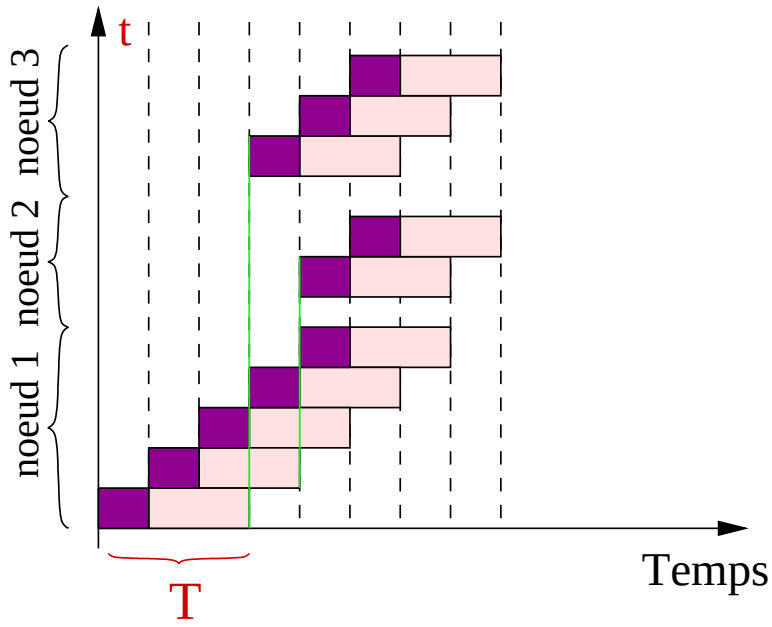
Modélisation

- Modèle :
  - création d'un processus = T
  - Tps non recouvrable t entre 2 initiations successives
  - diffusion en arbre
- Equivalent au problème *de diffusion d'un message*
  dans le modèle Postal [BarKip92]
  $\Rightarrow$ Ordonnancement optimal :

"Stratégie au plus tôt"

Stratégie "au plus tôt"

Stratégie "au plus tôt" en fonction de T_{proto} et t_{proto} :



Expérimentations

- Méthode statique :
  si t_{proto} et T_{proto} connus $\rightarrow$ trivial
- Méthode dynamique :
  Algorithme de vol de travail

Méthode dynamique

Vol de travail hiérarchique ("glouton")

Coût d'une communication $<<$ Coût d'un appel distant

Méthode dynamique

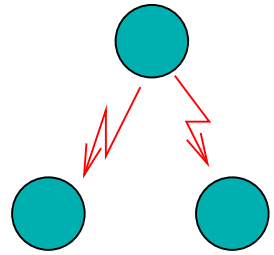
Vol de travail hiérarchique ("glouton")

Coût d'une communication $<<$ Coût d'un appel distant

Méthode dynamique

Vol de travail hiérarchique ("glouton")

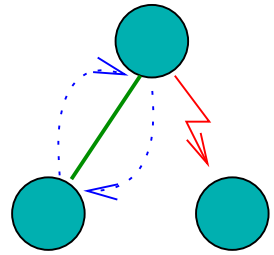
Coût d'une communication $<<$ Coût d'un appel distant



Méthode dynamique

Vol de travail hiérarchique ("glouton")

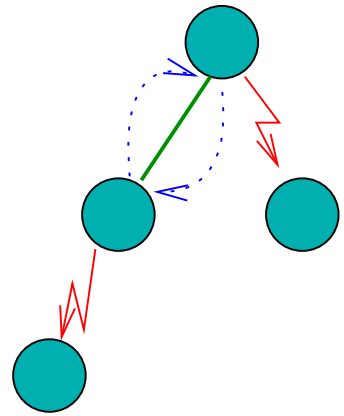
Coût d'une communication << Coût d'un appel distant



Méthode dynamique

Vol de travail hiérarchique ("glouton")

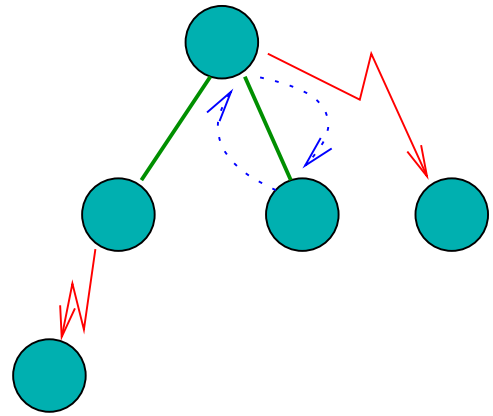
Coût d'une communication << Coût d'un appel distant



Méthode dynamique

Vol de travail hiérarchique ("glouton")

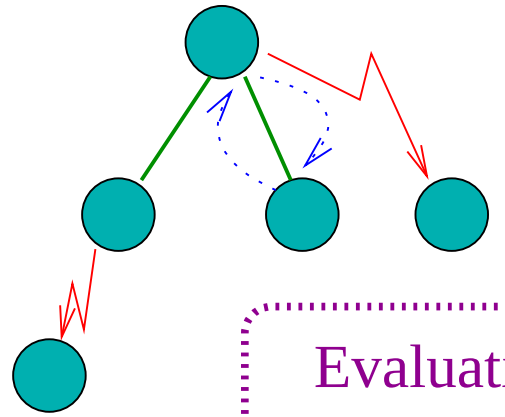
Coût d'une communication $<<$ Coût d'un appel distant



Méthode dynamique

Vol de travail hiérarchique ("glouton")

Coût d'une communication << Coût d'un appel distant



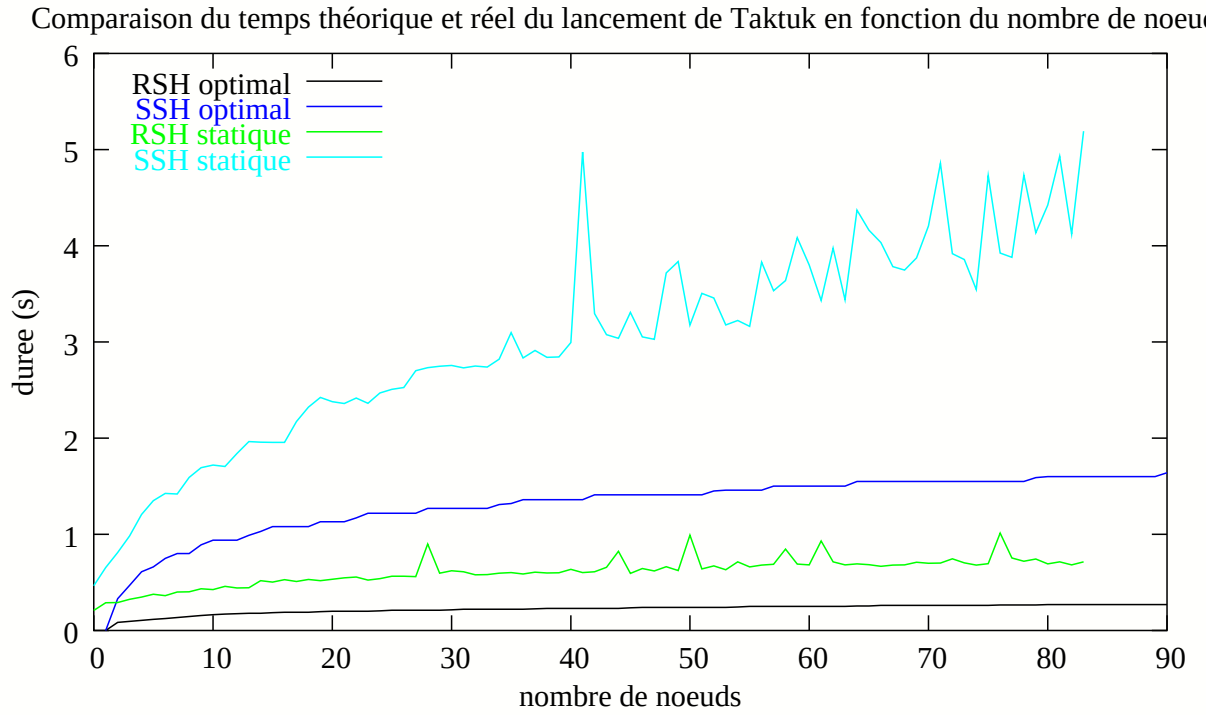
Evaluation dynamique
du taux de concurrence

Evaluation du taux de concurrence

- Problème difficile :
  - CPU, E/S disque, communications réseau,...
  - Quels paramètres pertinents pour le déploiement ?
- Solution naïve : indice de charge du système
  - ⊕ Prise en compte CPU, disque, réseau, nb processus
  - ⊖ Réactivité (moyenne amortie), précision
  - ⊕⊕ Prend en compte l'état du noeud

Evaluation méthode statique

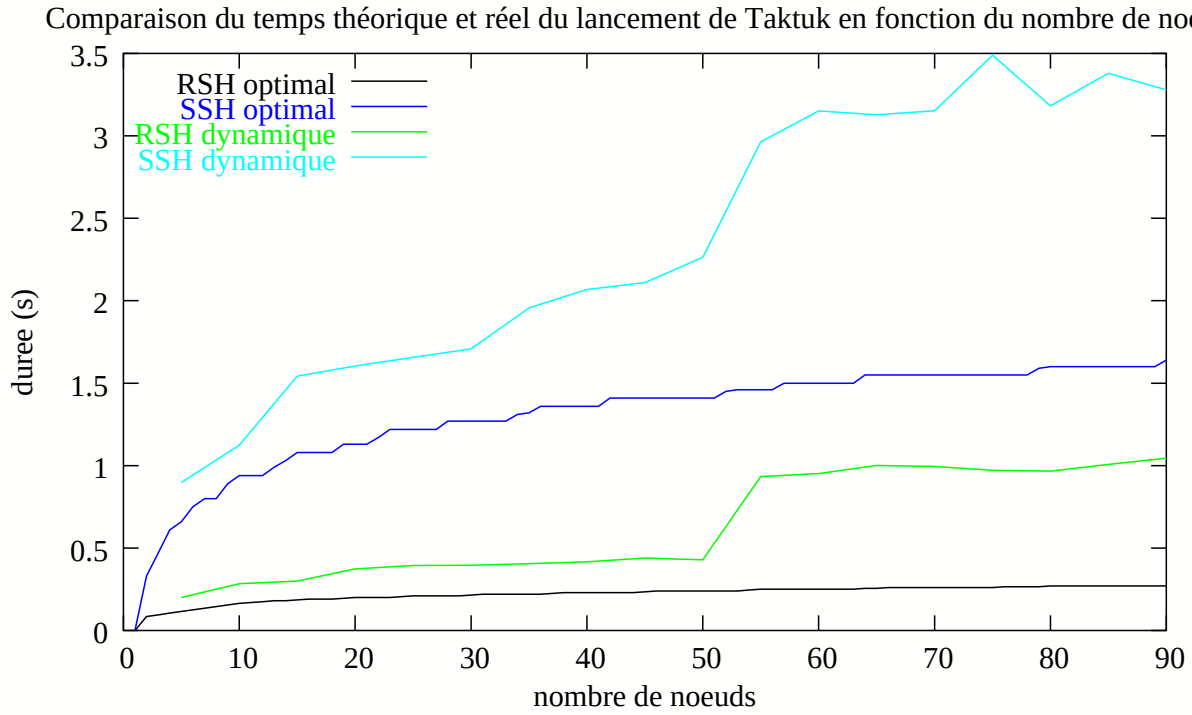
Icluster : 100 Pentium 733mhz, Ethernet 100 Mb/s commuté



- Paramètres T et t
- Homogénéité de la plate-forme

Evaluation méthode dynamique

Icluster : 100 Pentium 733mhz, Ethernet 100 Mb/s commuté



- Proche optimal
- Homogénéité de la plate-forme

Bilan

- Performances

  3s avec ssh sur 100 noeuds (ssh = 350ms)

- Passage à l'échelle

  coût logarithmique

- Robuste

  approche dynamique

- Générique

  indépendant d'un protocole

Plan

- Contexte & Problématique
- Etat de l'art
- Notre approche
  - Concepts algorithmes
  - Bibliothèque Taktuk
- Projets reposant sur Taktuk
- Conclusion et perspectives

Taktuk

Objectifs :

- Performance / passage à l'échelle
- Générique / flexible
- Bibliothèque de programmation

Contraintes :

- Robustesse
- Adaptabilité
- Installation / utilisation niveau utilisateur

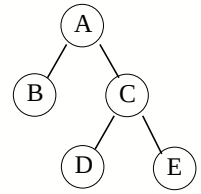
Taktuk : Services offerts

- Auto-propagation (déploiement)
- Contrôle (arrêt, démarrage, etc...)
- Redirection Entrées / Sorties
- Communication
  - routage logique (arbre)
  - messages actifs

Taktuk: Description

Méthode statique :

- Langage de description de l'arbre de diffusion
- Taktuk indépendant de l'algo d'ordonnancement
- Langage parenthésé :

A (B, C (D,E)) $\longrightarrow$ 

- Interpréter récursivement

Méthode dynamique :

- Liste de machines

Taktuk: Flexibilité

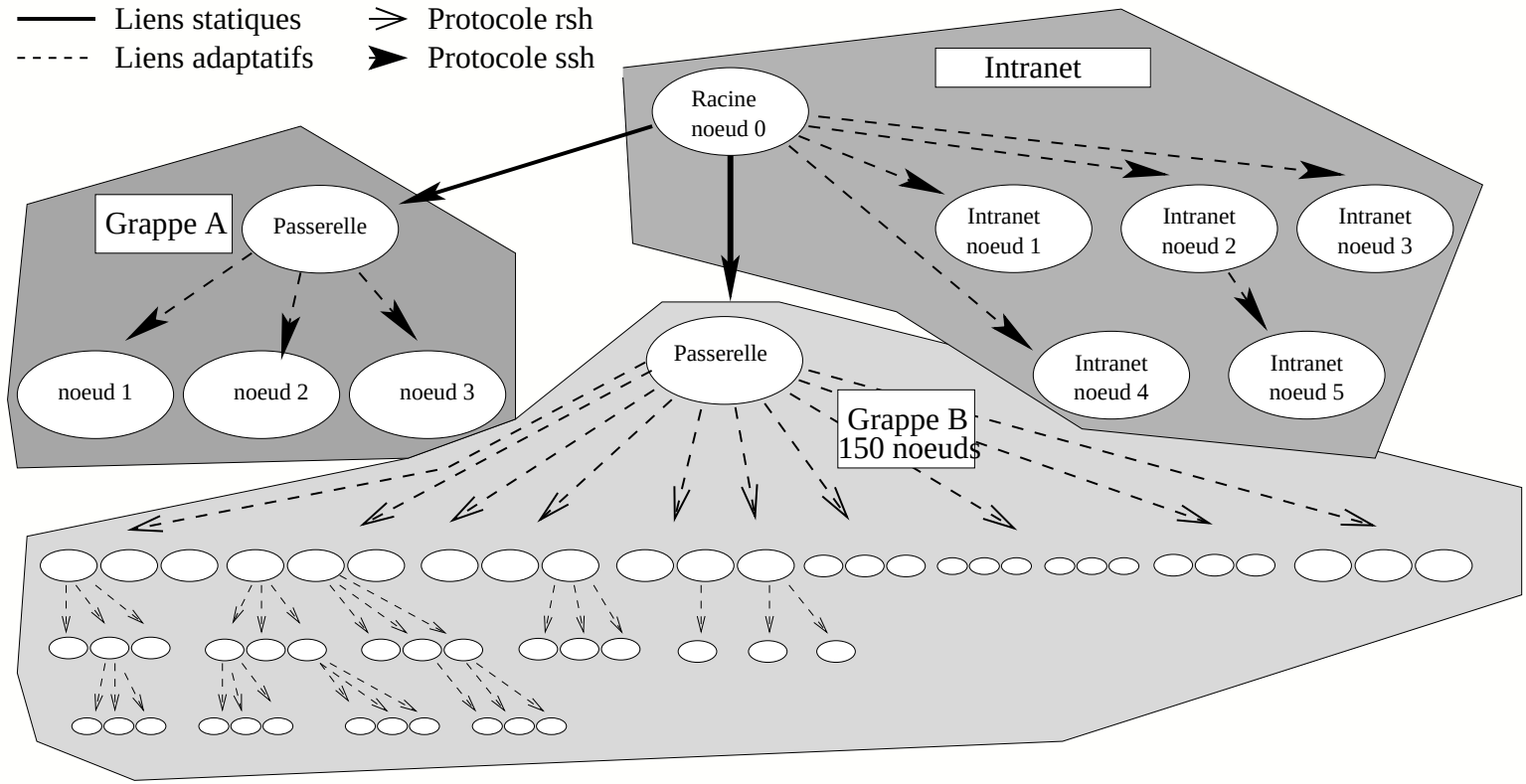
Enrichissement du langage
Spécification d'attributs pour un noeud ou un groupe de noeuds :

- protocole (rsh, ssh, autre ...)
- passerelles
- nom d'utilisateur
- temps limite de détection de défaillances
- arguments applicatifs

Facilement extensible

Bull 

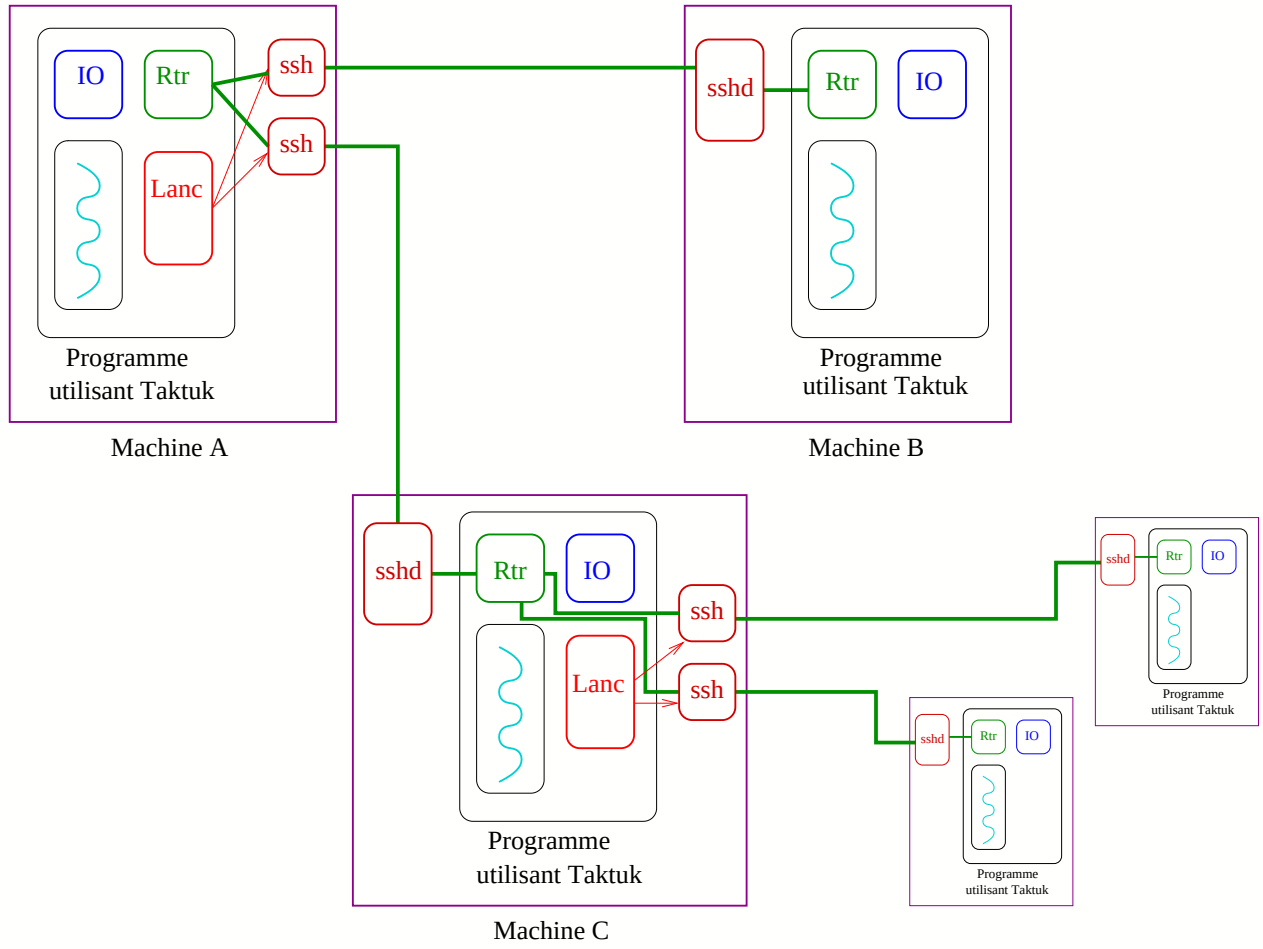
Exemple: "grille légère"



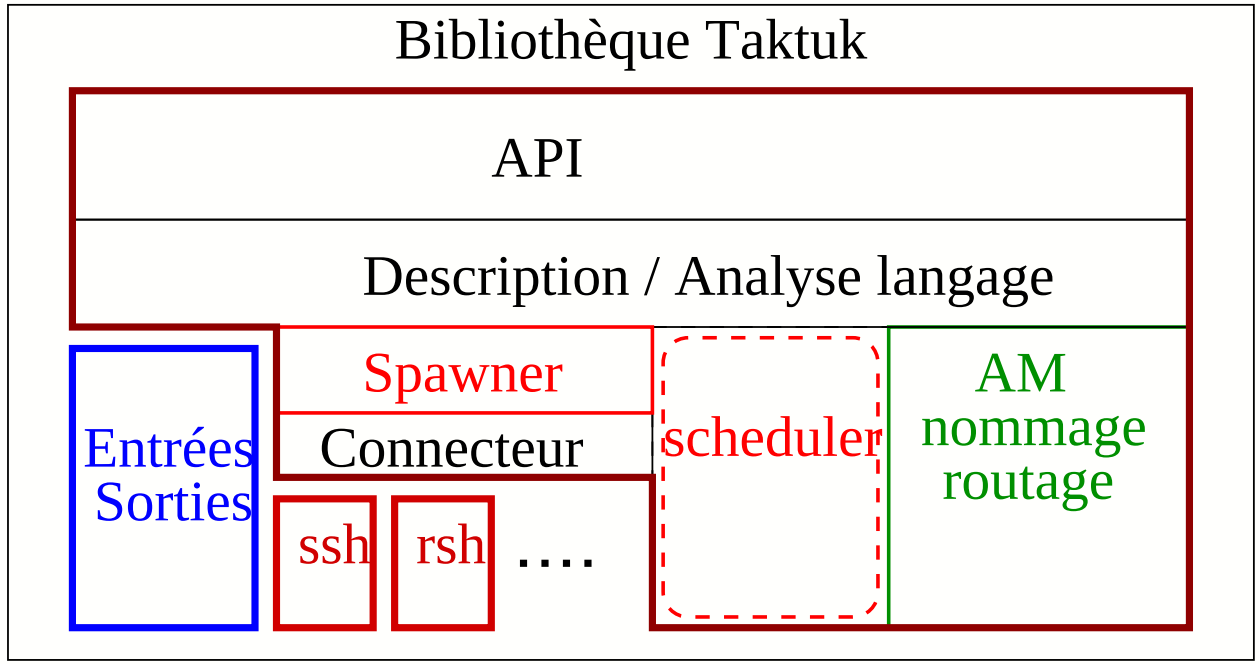
Ligne de commande :

Taktuk -cssh -m passerelle_G_B -[-c rsh -d -f Liste_machines -] -m passerelle_G_A -[-c ssh -d -f Liste_machines -] -d -f Liste_intranet

Taktuk : Architecture fonctionnelle



Taktuk: Achitecture logicielle



Taktuk : Bilan

- Interactif
- Générique (connecteurs virtuels)
- Flexible (langage de description)
- Robuste (approche dynamique)
- En exploitation depuis 1 an [W. Billot]

Plan

- Contexte & Problématique
- Etat de l'art
- Notre approche
  - Concepts algorithmes
  - Bibliothèque Taktuk
- Projets reposant sur Taktuk
- Conclusion et perspectives

Inuktitut

- Intergiciel de communication
- Multi-réseaux / multi-protocoles
- Messages actifs / écriture distante
- C++

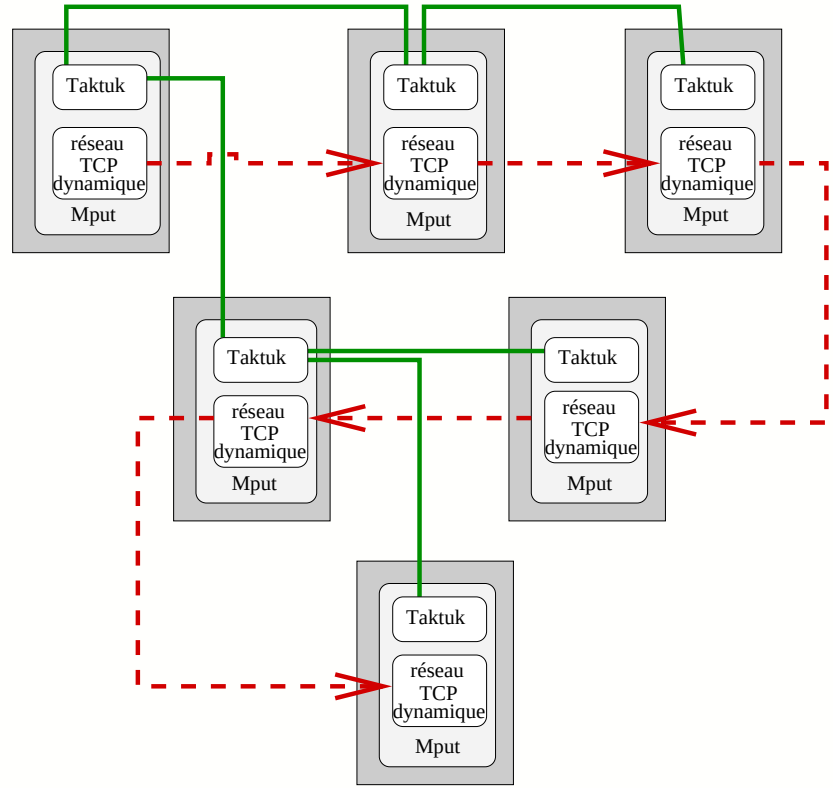
- Fort couplage avec Taktuk
- Utilisé par Athapascan (grappes et grilles légères)

KaRun

- Ensemble de commandes Unix parallèles
- Rshp : invite de commandes parallèle
- Mput : diffusion de fichiers
- Utilisé et distribué dans la distribution Mandrake CLIC (Projet BULL-INRIA) :

  urpmi : installation de packages sur grappes (mput + rshp)

KaRun : Mput



Mput < 3sec pour diffuser 16Mo sur 200 noeuds (Icluster PIII + Ether 100Mb)

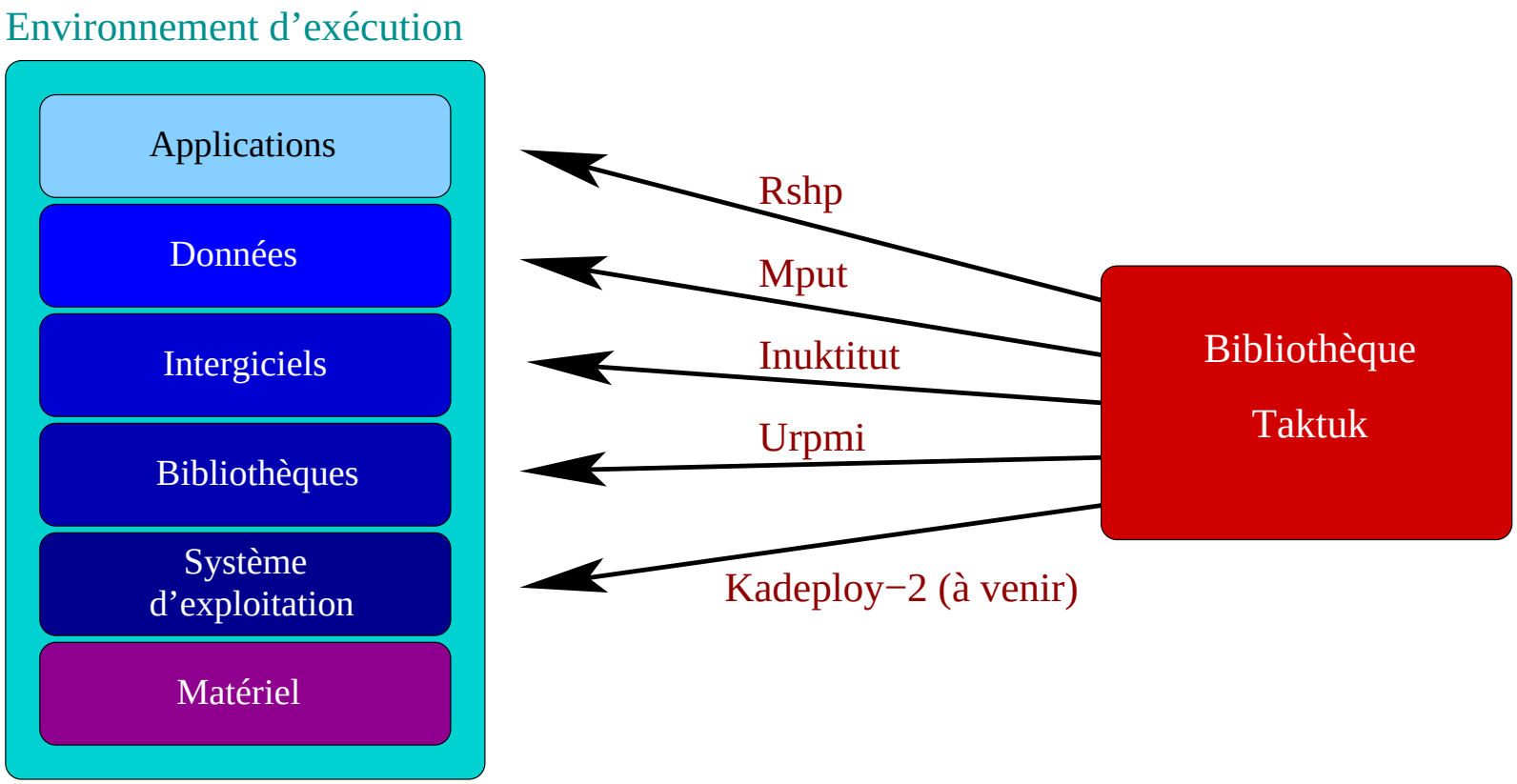
Storm 110ms pour diffuser 12M0 sur 64 noeuds (Quadrics + Alpha 866Mhz)

OAR

Gestionnaire de ressources pour grappes ("batch scheduler")
Taktuk :

- Vérifier l'état des noeuds (niveau applicatif)
- "Découverte" de noeuds
  (grappes dynamiques IDPOT)
- Traitements de pré ou post réservation
- Soumission de travail interactive

Bilan sur l'utilisation de Taktuk



Conclusion

Taktuk : ordonnanceur d'appels d'exécution distante

- Passage à l'échelle
- Performant
- Robuste
- Installation et utilisation simple
- Portable et générique
- Disponible sous la forme d'une bibliothèque
- Utilisé et fonctionnel

Améliorations

- Réduction du surcoût de Taktuk
- Expérimentations sur SMP
  - IDPOT (48 bi-xéons, Gb/s)
  - Icluster2 (104 bi-IA64, Gb/s et Myrinet)

Perspectives (1/2)

- Estimateur de concurrence (SMP)
- Approche statique / résultats théoriques
- Approche optimiste
  - Combinaison approche statique et dynamique
  - Pré-répartition ↘ coût du vol

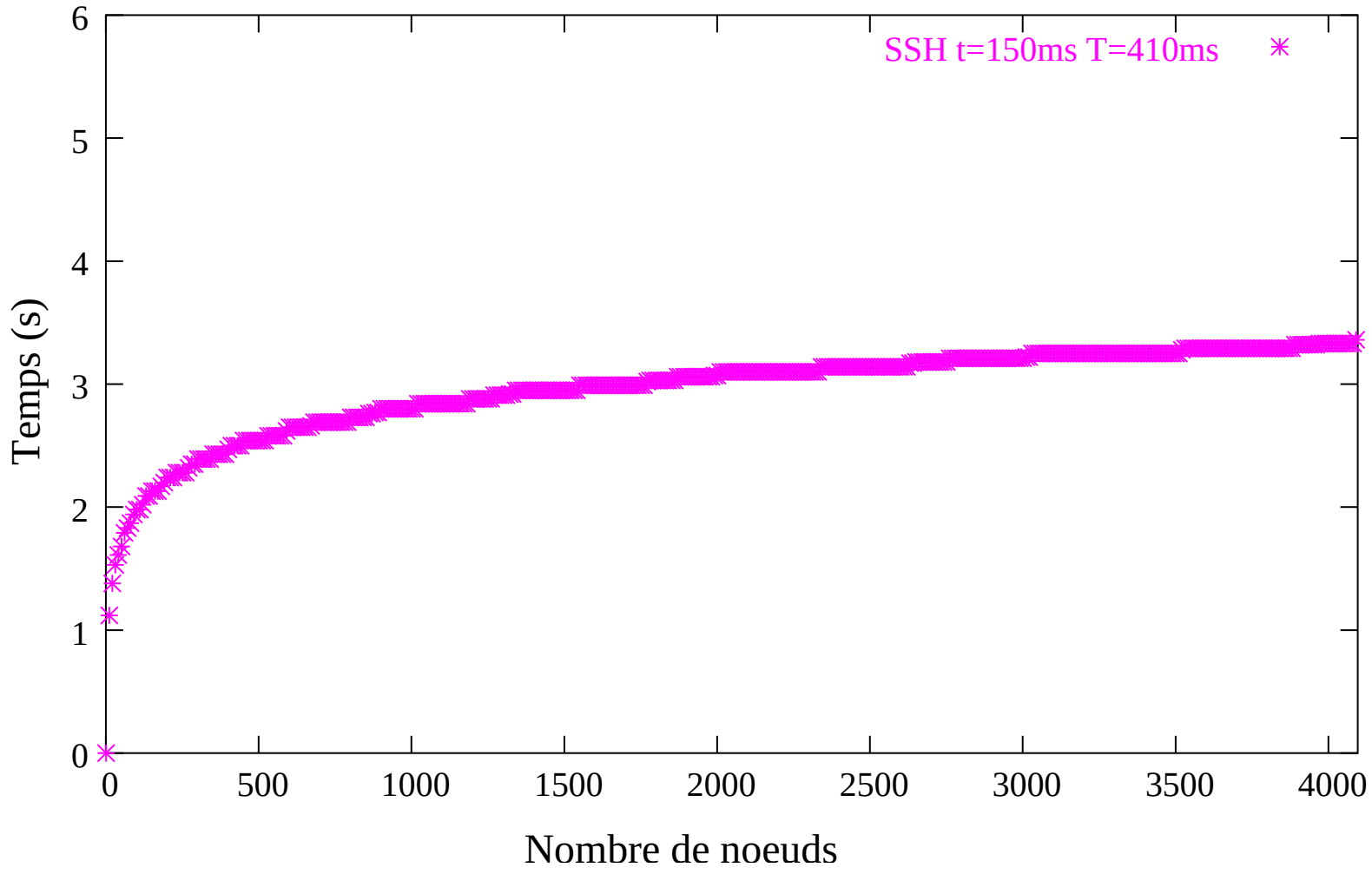
Perspectives (2/2)

- Diffusion de fichiers (Mput + Inuktitut)
  - Topologie variable du réseau de diffusion
  - Exploitation d'un réseau physique performant
- Ajout / retrait dynamique de noeuds
- "Découverte" de ressources
  - Arrêt de découverte sur critères (Tps, Nb ressources)

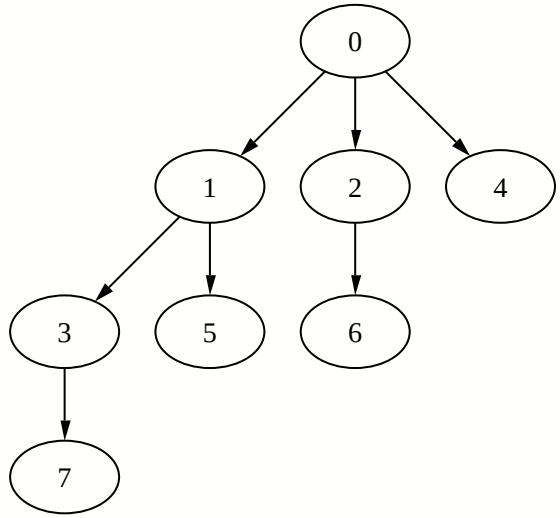
Questions ?

Projet	Temps de déploiement
C3 (rsh)	2.57 sec 200 noeuds (PIII 700 Mhz icluster)
C3 (ssh)	14.52 sec 200 noeuds (PIII 700 Mhz icluster)
Taktuk (rsh)	0.7 sec 100 noeuds (PIII 700 Mhz icluster)
Taktuk (ssh)	3.5 sec 100 noeuds (PIII 700 Mhz icluster)
Storm	0.11 sec 64 noeuds + diffusion 12Mo (Alpha 833 Mhz + Quadrics)
Mput (rsh)	2.5sec 200 noeuds + diffusion 16Mo (PIII 700 Mhz icluster)

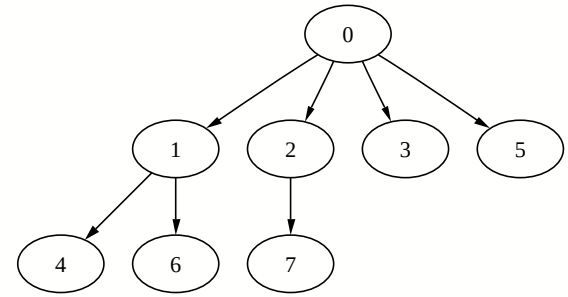
Prévisions


SSH t=150ms T=410ms
Temps (s)
Nombre de noeuds
0
1
2
3
4
5
6
0
500
1000
1500
2000
2500
3000
3500
4000

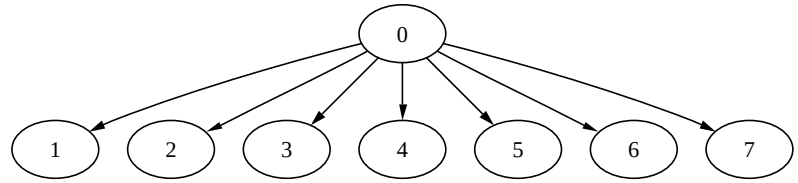
Exemples en fonction de λ



Arbre de diffusion binomial avec $\lambda = 1$

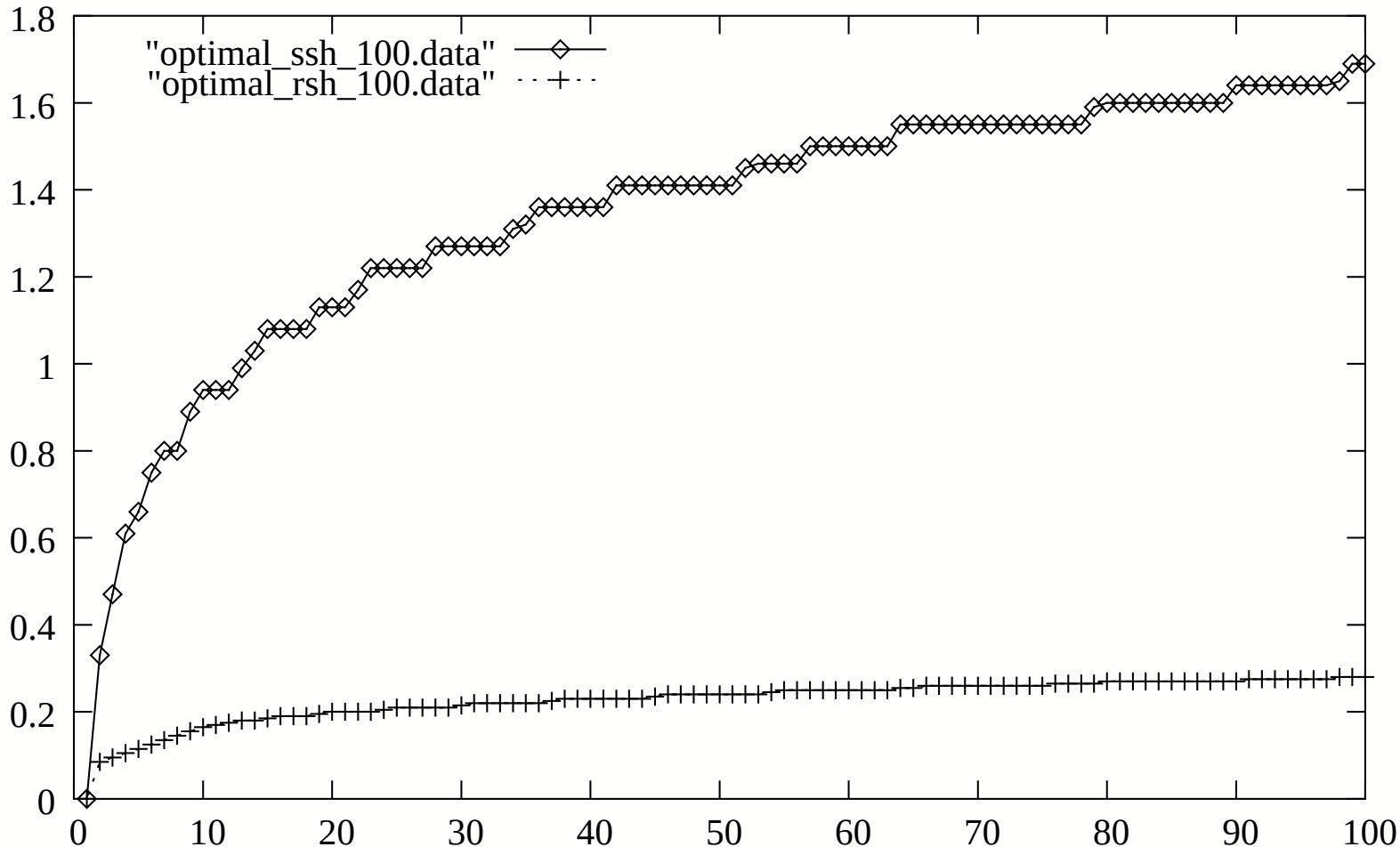


Arbre de diffusion avec $\lambda = 2$



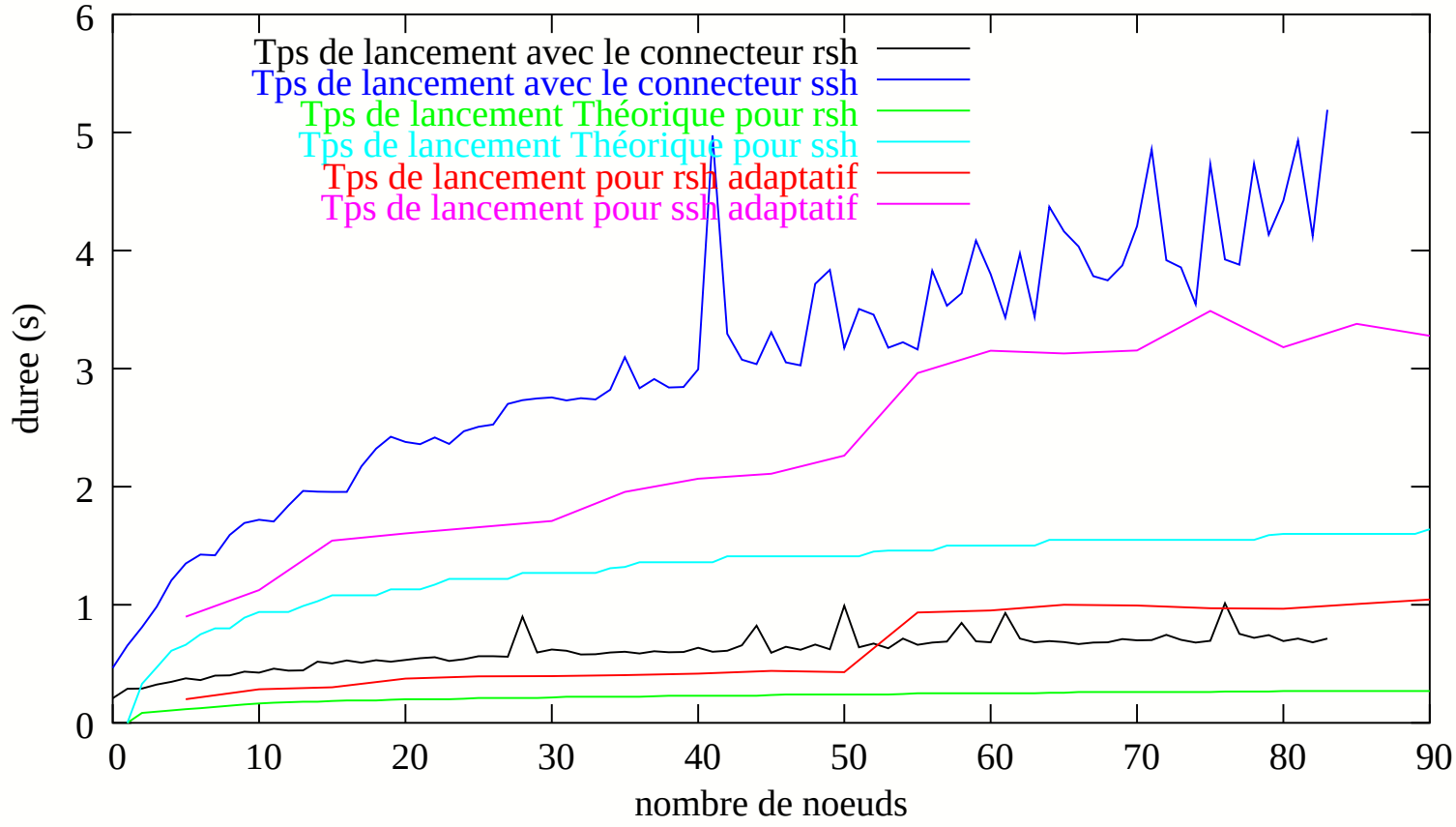
Arbre de diffusion avec $\lambda = 10$

Ordonnancement: Borne inférieure



Théorique Optimal Dynamique

Comparaison du temps théorique et réel du lancement de Taktuk en fonction du nombre de noeuds



Taktuk: Exemple d'utilisation

```
Taktuk.init(...)
print("bonjour tout le monde")
Taktuk.terminate()
```

$\Rightarrow$:

```
>./bonjour -v -cssh -m machineA -m machineB -m machineC
<machine_locale> [rank:0] :->:bonjour tout le monde
<machineA> [rank:2] :->:bonjour tout le monde
<machineB> [rank:1] :->:bonjour tout le monde
<machineC> [rank:3] :->:bonjour tout le monde
```