Découverte de services dans un environnement pair-à-pair pour les grilles de calcul

Cédric Tedeschi

ACI GDS - 06 juillet 2005

Plan

Plan

Contexte

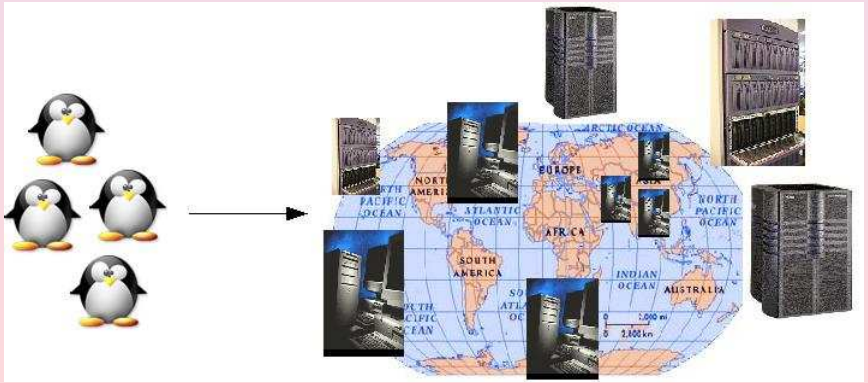
- Constats
  - explosion des besoins en puissance de calcul
  - coût important des supercalculateurs
- Apparition de nouvelles technologies
  - réseaux (très) haut débit
  - démocratisation des grappes de machines
  - $\Rightarrow$ émergence des grilles de calcul
- Problématique des grilles
  - agréger les ressources
  - les rendre accessibles
  - $\Rightarrow$ intergiciel de grille

Contexte

- Constats
  - explosion des besoins en puissance de calcul
  - coût important des supercalculateurs
- Apparition de nouvelles technologies
  - réseaux (très) haut débit
  - démocratisation des grappes de machines
  - $\Rightarrow$ émergence des grilles de calcul
- Problématique des grilles
  - agréger les ressources
  - les rendre accessibles
  - $\Rightarrow$ intergiciel de grille

Contexte

- Constats
  - explosion des besoins en puissance de calcul
  - coût important des supercalculateurs
- Apparition de nouvelles technologies
  - réseaux (très) haut débit
  - démocratisation des grappes de machines
  - $\Rightarrow$ émergence des grilles de calcul
- Problématique des grilles
  - agréger les ressources
  - les rendre accessibles
  - $\Rightarrow$ intergiciel de grille



Problématique

- Services ≡ ensemble des applicatifs disponibles
  - bibliothèques de calcul, exécutables
  - préinstallés sur la grille (serveurs)
- Découverte (par les clients)
  - multicritères (`nom`, `système`, `architecture`, `adresse`)
  - complétion automatique
  - rapide (minimisation des temps de communication)
- Environnement
  - statique (UDDI, Globus, GridRPC)
  - dynamique à large échelle(P2P)
    - tolérance aux pannes
    - répartition de la charge

Problématique

- Services ≡ ensemble des applicatifs disponibles
  - bibliothèques de calcul, exécutables
  - préinstallés sur la grille (serveurs)
- Découverte (par les clients)
  - multicritères (`nom`, `système`, `architecture`, `adresse`)
  - complétion automatique
  - rapide (minimisation des temps de communication)
- Environnement
  - statique (UDDI, Globus, GridRPC)
  - dynamique à large échelle(P2P)
    - tolérance aux pannes
    - répartition de la charge

Problématique

- Services ≡ ensemble des applicatifs disponibles
  - bibliothèques de calcul, exécutables
  - préinstallés sur la grille (serveurs)
- Découverte (par les clients)
  - multicritères (`nom`, `système`, `architecture`, `adresse`)
  - complétion automatique
  - rapide (minimisation des temps de communication)
- Environnement
  - statique (UDDI, Globus, GridRPC)
  - dynamique à large échelle(P2P)
    - tolérance aux pannes
    - répartition de la charge

Plan

Réponses des technologies pair-à-pair (1)

- Définitions
  - stricte (Gnutella, DHT)
  - élargie (Napster)
- Propriétés
  - répartition de la charge
  - tolérance aux pannes (environnement dynamique)
- Pair-à-pair non structuré (Gnutella, KaZaA)
  - inondation
  - non exhaustivité
  - surcharge du réseau
- Pair-à-pair structuré : DHT (CAN, Chord, Pastry, Tapestry)
  - routage en fonction de la clef (toto.mp3, 140.77.23.58)
  - exhaustivité
  - routage logarithmique
  - ignorance de la topologie du réseau physique
  - interrogation rigide

Réponses des technologies pair-à-pair (1)

- Définitions
  - stricte (Gnutella, DHT)
  - élargie (Napster)
- Propriétés
  - répartition de la charge
  - tolérance aux pannes (environnement dynamique)
- Pair-à-pair non structuré (Gnutella, KaZaA)
  - inondation
  - non exhaustivité
  - surcharge du réseau
- Pair-à-pair structuré : DHT (CAN, Chord, Pastry, Tapestry)
  - routage en fonction de la clef (toto.mp3, 140.77.23.58)
  - exhaustivité
  - routage logarithmique
  - ignorance de la topologie du réseau physique
  - interrogation rigide

Réponses des technologies pair-à-pair (1)

- Définitions
  - stricte (Gnutella, DHT)
  - élargie (Napster)
- Propriétés
  - répartition de la charge
  - tolérance aux pannes (environnement dynamique)
- Pair-à-pair non structuré (Gnutella, KaZaA)
  - inondation
  - non exhaustivité
  - surcharge du réseau
- Pair-à-pair structuré : DHT (CAN, Chord, Pastry, Tapestry)
  - routage en fonction de la clef (toto.mp3, 140.77.23.58)
  - exhaustivité
  - routage logarithmique
  - ignorance de la topologie du réseau physique
  - interrogation rigide

Réponses des technologies pair-à-pair (1)

- Définitions
  - stricte (Gnutella, DHT)
  - élargie (Napster)
- Propriétés
  - répartition de la charge
  - tolérance aux pannes (environnement dynamique)
- Pair-à-pair non structuré (Gnutella, KaZaA)
  - inondation
  - non exhaustivité
  - surcharge du réseau
- Pair-à-pair structuré : DHT (CAN, Chord, Pastry, Tapestry)
  - routage en fonction de la clef (toto.mp3, 140.77.23.58)
  - exhaustivité
  - routage logarithmique
  - ignorance de la topologie du réseau physique
  - interrogation rigide

Réponses des technologies pair-à-pair (1)

- Définitions
  - stricte (Gnutella, DHT)
  - élargie (Napster)
- Propriétés
  - répartition de la charge
  - tolérance aux pannes (environnement dynamique)
- Pair-à-pair non structuré (Gnutella, KaZaA)
  - inondation
  - non exhaustivité
  - surcharge du réseau
- Pair-à-pair structuré : DHT (CAN, Chord, Pastry, Tapestry)
  - routage en fonction de la clef (toto.mp3, 140.77.23.58)
  - exhaustivité
  - routage logarithmique
  - ignorance de la topologie du réseau physique
  - interrogation rigide

Réponses des technologies pair-à-pair (1)

- Définitions
  - stricte (Gnutella, DHT)
  - élargie (Napster)
- Propriétés
  - répartition de la charge
  - tolérance aux pannes (environnement dynamique)
- Pair-à-pair non structuré (Gnutella, KaZaA)
  - inondation
  - non exhaustivité
  - surcharge du réseau
- Pair-à-pair structuré : DHT (CAN, Chord, Pastry, Tapestry)
  - routage en fonction de la clef (toto.mp3, 140.77.23.58)
  - exhaustivité
  - routage logarithmique
  - ignorance de la topologie du réseau physique
  - interrogation rigide

Réponses des technologies pair-à-pair (2)

- Localité physique dans les DHT (DHT hiérarchiques)
  - approche des **supernœuds** (Brocade, Jelly, etc.)
  - approche des **nœuds repères** (CAN, ECAN, Hieras, etc.)
- Système d'interrogation complexe
  - description semi-structurée (INS/Twine, etc.)
  - requête sur un ensemble de clefs numériques
  - Squid
    - autocomplétion
    - recherche multicritères
    - perte de la répartition uniforme
    - surcouche des DHT → pire cas $O(N \log(N))$
    - nombre fixe de critères
    - pas de localité physique

Réponses des technologies pair-à-pair (2)

- Localité physique dans les DHT (DHT hiérarchiques)
  - approche des **supernœuds** (Brocade, Jelly, etc.)
  - approche des **nœuds repères** (CAN, ECAN, Hieras, etc.)
- Système d'interrogation complexe
  - description semi-structurée (INS/Twine, etc.)
  - requête sur un ensemble de clefs numériques
  - Squid
    - autocomplétion
    - recherche multicritères
    - perte de la répartition uniforme
    - surcouche des DHT → pire cas $O(N \log(N))$
    - nombre fixe de critères
    - pas de localité physique

Réponses des technologies pair-à-pair (2)

- Localité physique dans les DHT (DHT hiérarchiques)
  - approche des **supernœuds** (Brocade, Jelly, etc.)
  - approche des **nœuds repères** (CAN, ECAN, Hieras, etc.)
- Système d'interrogation complexe
  - description semi-structurée (INS/Twine, etc.)
  - requête sur un ensemble de clefs numériques
  - Squid
    - autocomplétion
    - recherche multicritères
    - perte de la répartition uniforme
    - surcouche des DHT → pire cas $O(N \log(N))$
    - nombre fixe de critères
    - pas de localité physique

Réponses des technologies pair-à-pair (2)

- Localité physique dans les DHT (DHT hiérarchiques)
  - approche des **supernœuds** (Brocade, Jelly, etc.)
  - approche des **nœuds repères** (CAN, ECAN, Hieras, etc.)
- Système d'interrogation complexe
  - description semi-structurée (INS/Twine, etc.)
  - requête sur un ensemble de clefs numériques
  - Squid
    - autocomplétion
    - recherche multicritères
    - perte de la répartition uniforme
    - surcouche des DHT $\rightarrow$ pire cas $O(N \log(N))$
    - nombre fixe de critères
    - pas de localité physique

Réponses des technologies pair-à-pair (2)

- Localité physique dans les DHT (DHT hiérarchiques)
  - approche des **supernœuds** (Brocade, Jelly, etc.)
  - approche des **nœuds repères** (CAN, ECAN, Hieras, etc.)
- Système d'interrogation complexe
  - description semi-structurée (INS/Twine, etc.)
  - requête sur un ensemble de clefs numériques
  - Squid
    - autocomplétion
    - recherche multicritères
    - perte de la répartition uniforme
    - surcouche des DHT $\rightarrow$ pire cas $O(N \log(N))$
    - nombre fixe de critères
    - pas de localité physique

Plan

Description d'un service

Val = { nom_service,
système,
architecture,
adresse}

(clef, valeur)
(nom_service, val)
(système, val)
(architecture, val)
(adresse, val)

Exemple

Val = { DGEMM,
Linux Debian 3.0,
PowerPC G5,
ble.ens-lyon.fr}

(clef, valeur)
(DGEMM, val)
(Linux Debian 3.0, val)
(PowerPC G5, val)
(ble.ens-lyon.fr, val)

Table de Placement Lexicographique Distribuée

- Similaire à une DHT
- Arbre lexicographique des clefs des services
- Gestion de l'arrivée **dynamique** des services
- 1 nœud logique $\equiv$ 1 clef
- Deux types de clefs
  - clef **réelle**
  - clef **virtuelle**
- Exemple
  - DGEMM
  - DTRSM
  - DTRMM
  - DGEMM

Table de Placement Lexicographique Distribuée

- Similaire à une DHT
- Arbre lexicographique des clefs des services
- Gestion de l'arrivée **dynamique** des services
- 1 nœud logique $\equiv$ 1 clef
- Deux types de clefs
  - clef **réelle**
  - clef **virtuelle**
- Exemple
  - DGEMM
  - DTRSM
  - DTRMM
  - DGEMM

DGEMM

Table de Placement Lexicographique Distribuée

- Similaire à une DHT
- Arbre lexicographique des clefs des services
- Gestion de l'arrivée **dynamique** des services
- 1 nœud logique $\equiv$ 1 clef
- Deux types de clefs
  - clef **réelle**
  - clef **virtuelle**
- Exemple
  - DGEMM
  - DTRSM
  - DTRMM
  - DGEMM

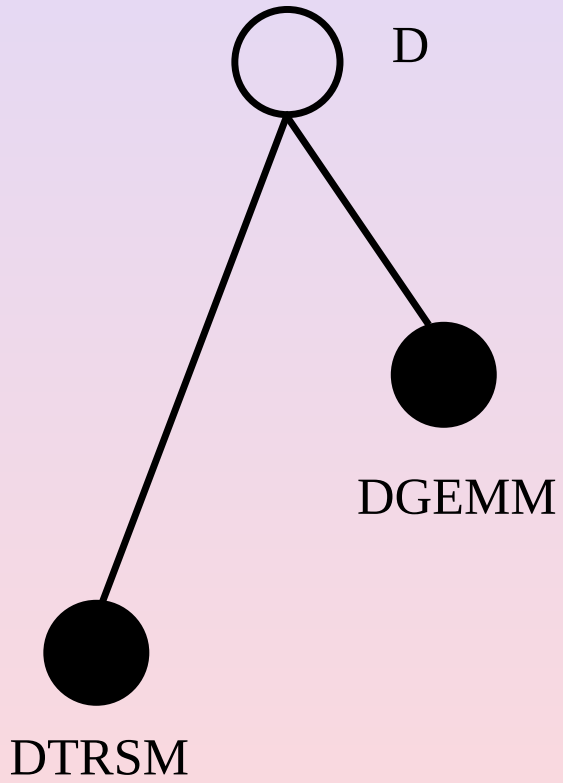


Table de Placement Lexicographique Distribuée

- Similaire à une DHT
- Arbre lexicographique des clefs des services
- Gestion de l'arrivée **dynamique** des services
- 1 nœud logique $\equiv$ 1 clef
- Deux types de clefs
  - clef **réelle**
  - clef **virtuelle**
- Exemple
  - DGEMM
  - DTRSM
  - DTRMM
  - DGEMM

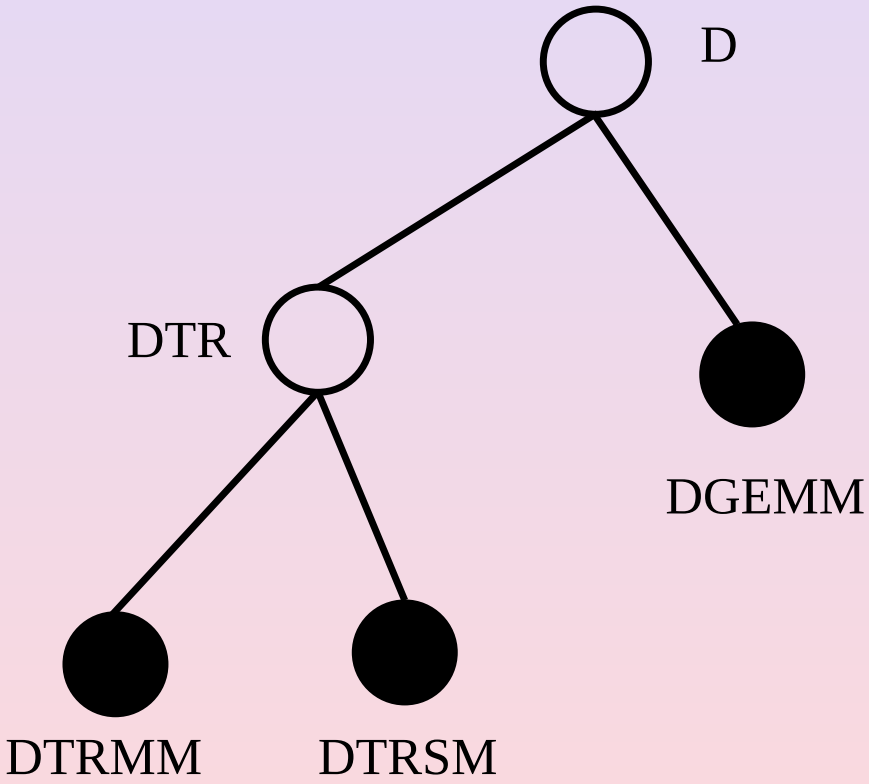
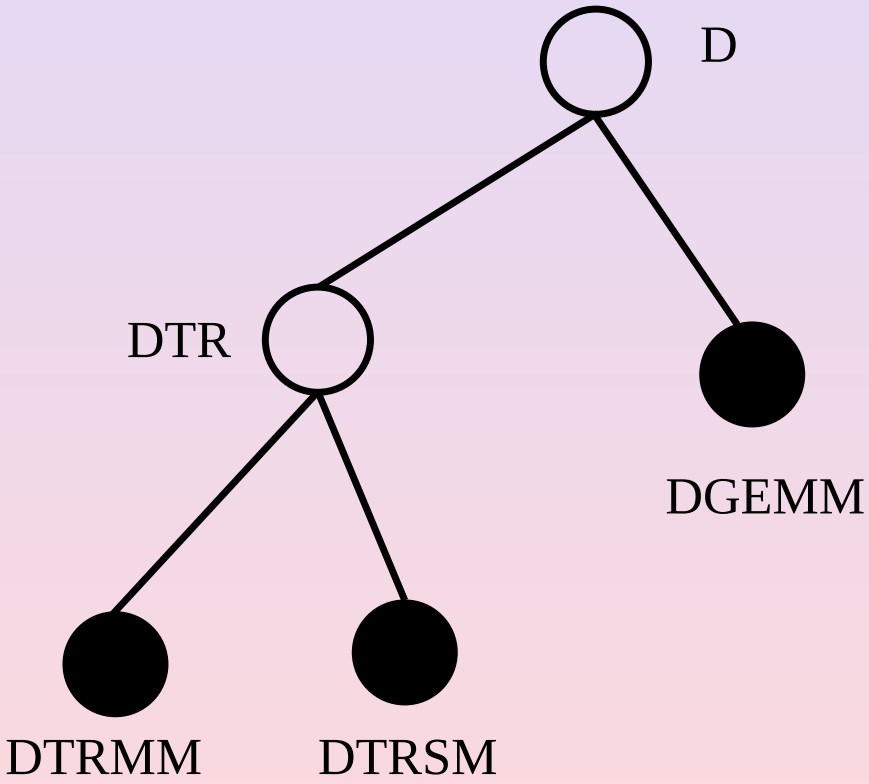


Table de Placement Lexicographique Distribuée

- Similaire à une DHT
- Arbre lexicographique des clefs des services
- Gestion de l'arrivée **dynamique** des services
- 1 nœud logique $\equiv$ 1 clef
- Deux types de clefs
  - clef **réelle**
  - clef **virtuelle**
- Exemple
  - DGEMM
  - DTRSM
  - DTRMM
  - DGEMM

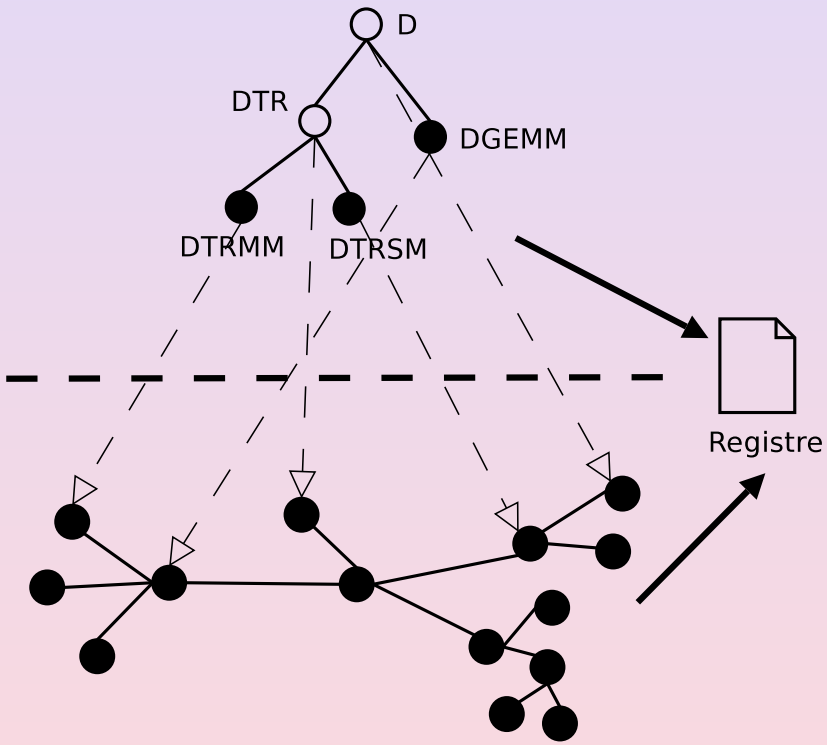


Placement des nœuds logiques

- Placement des nœuds logiques sur les **pairs**
- Deux approches
  - semi-centralisée
  - totalement distribuée

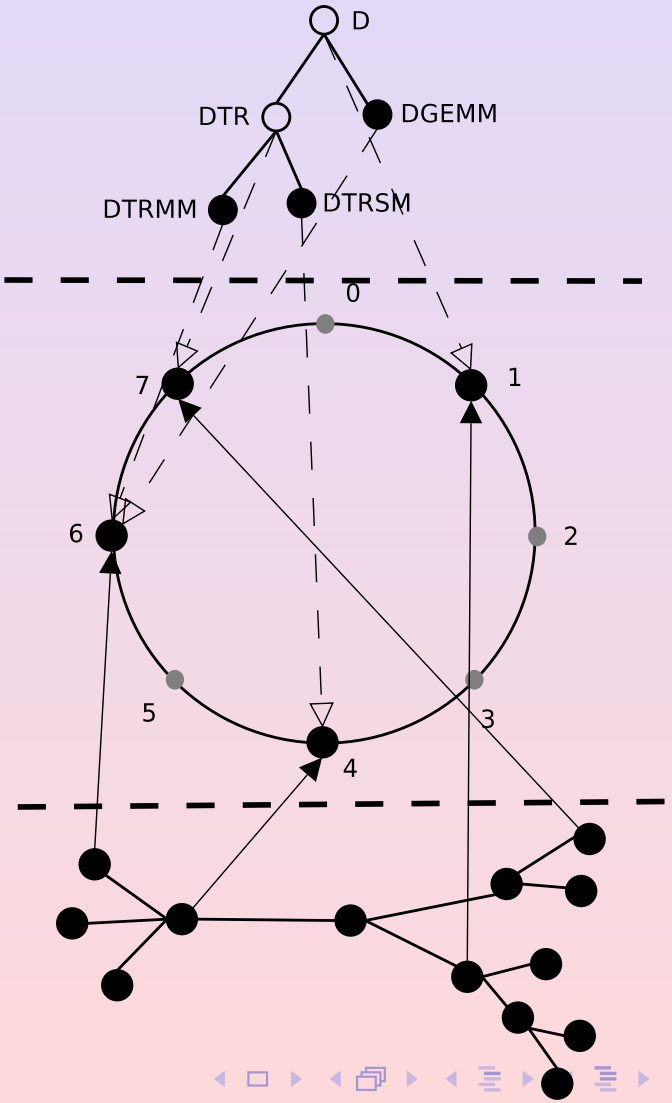
Placement des nœuds logiques

- Placement des nœuds logiques sur les **pairs**
- Deux approches
  - semi-centralisée
  - totalement distribuée



Placement des nœuds logiques

- Placement des nœuds logiques sur les **pairs**
- Deux approches
  - semi-centralisée
  - totalement distribuée



Tolérance aux pannes et arbre couvrant

- Prévention par réplication
  - k duplications des nœuds et des liens
  - répartition uniforme de la charge
    - registre
    - k fonctions de hachage aléatoires uniformes
- Minimisation des temps de communication (arbre couvrant)
  - heuristique gloutonne
  - minimiser localement pour minimiser globalement
  - intégration au processus de réplication
  - sans en modifier la complexité (linéaire)

Tolérance aux pannes et arbre couvrant

- Prévention par réplication
  - k duplications des nœuds et des liens
  - répartition uniforme de la charge
    - registre
    - k fonctions de hachage aléatoires uniformes
- Minimisation des temps de communication (arbre couvrant)
  - heuristique gloutonne
  - minimiser localement pour minimiser globalement
  - intégration au processus de réplication
  - sans en modifier la complexité (linéaire)

Algorithmes distribués

- Insertion (dans un arbre éventuellement répliqué)
  - contact
  - routage
  - insertion
- Réplication
  - lancement périodique depuis la racine
  - réplication de la racine
  - descente asynchrone dans l'arbre
- Mis en place par un ensemble de messages

Algorithmes distribués

- Insertion (dans un arbre éventuellement répliqué)
  - contact
  - routage
  - insertion
- Réplication
  - lancement périodique depuis la racine
  - réplication de la racine
  - descente asynchrone dans l'arbre
- Mis en place par un ensemble de messages

Algorithmes distribués

- Insertion (dans un arbre éventuellement répliqué)
  - contact
  - routage
  - insertion
- Réplication
  - lancement périodique depuis la racine
  - réplication de la racine
  - descente asynchrone dans l'arbre
- Mis en place par un ensemble de messages

Exemple : insertion

Arrivée consécutive des services

- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult, chol

Exemple : insertion

Arrivée consécutive des services

- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult_chol

Exemple : insertion

Arrivée consécutive des services

- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult, chol

Exemple : insertion

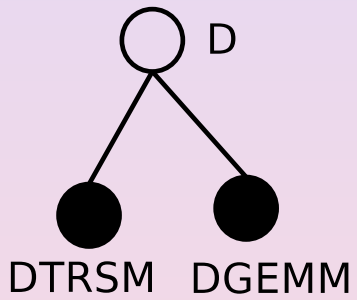
Arrivée consécutive des services

- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult, chol

Exemple : insertion

Arrivée consécutive des services

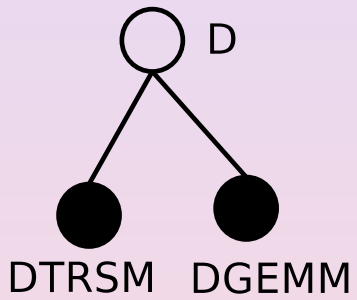
- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult_chol



Exemple : insertion

Arrivée consécutive des services

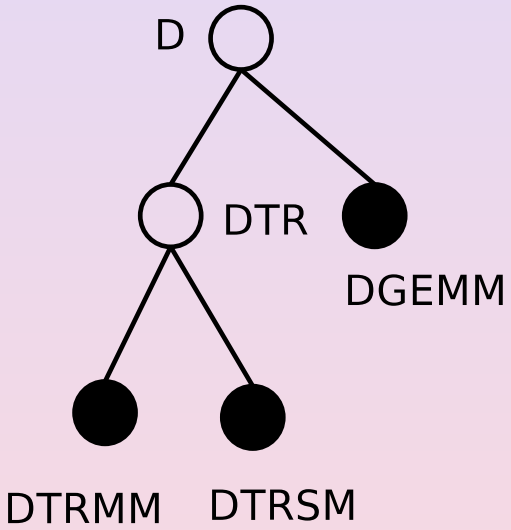
- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult_chol



Exemple : insertion

Arrivée consécutive des services

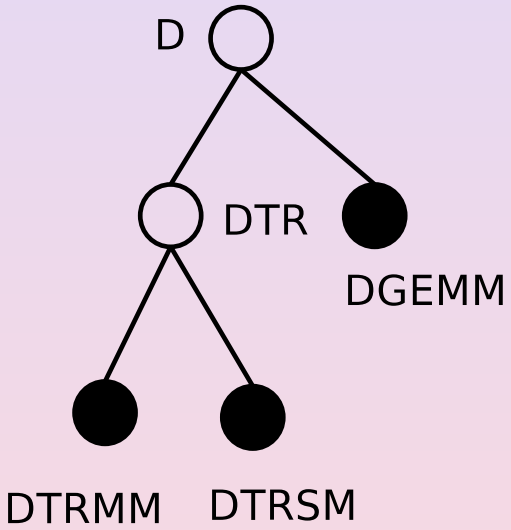
- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult, chol



Exemple : insertion

Arrivée consécutive des services

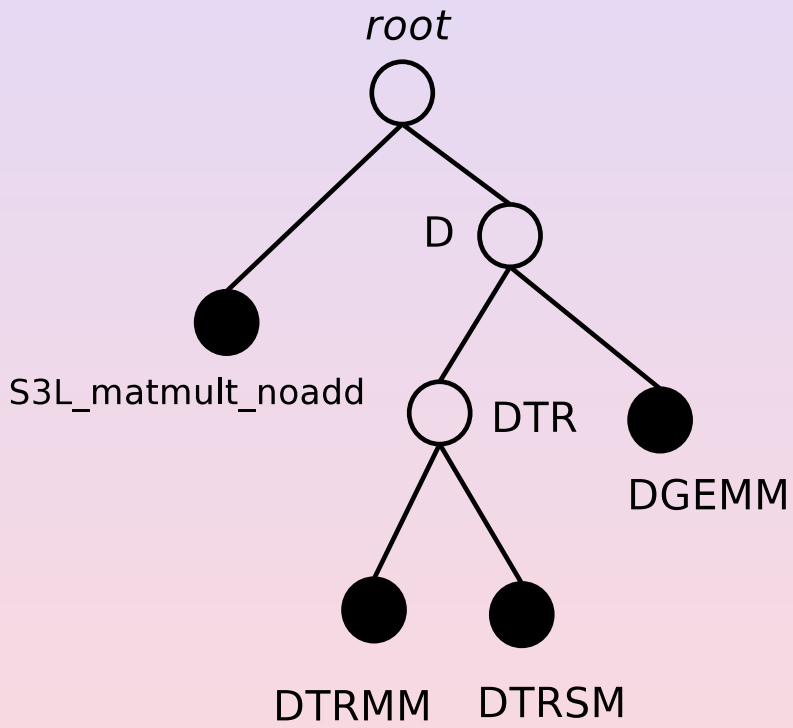
- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult, chol



Exemple : insertion

Arrivée consécutive des services

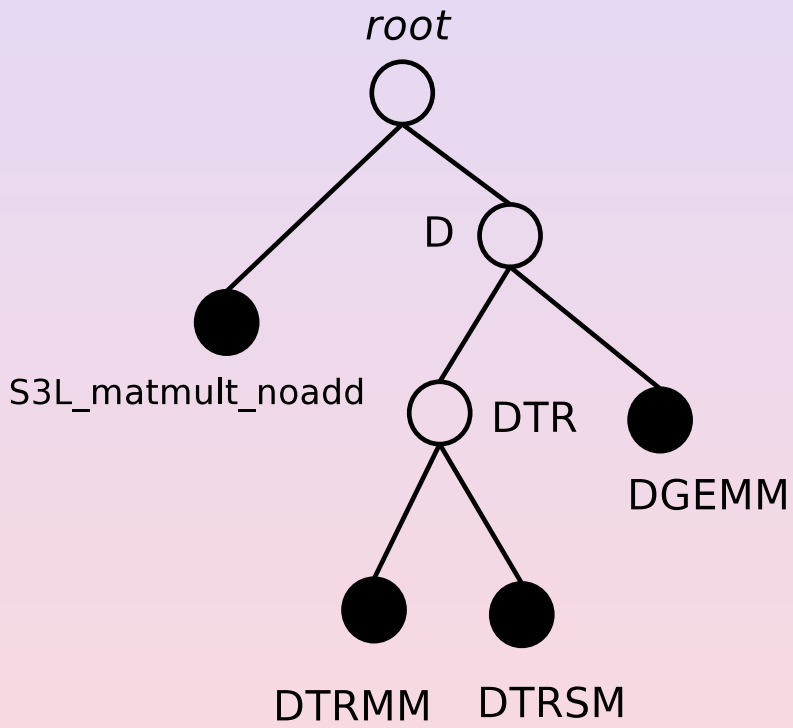
- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult, chol



Exemple : insertion

Arrivée consécutive des services

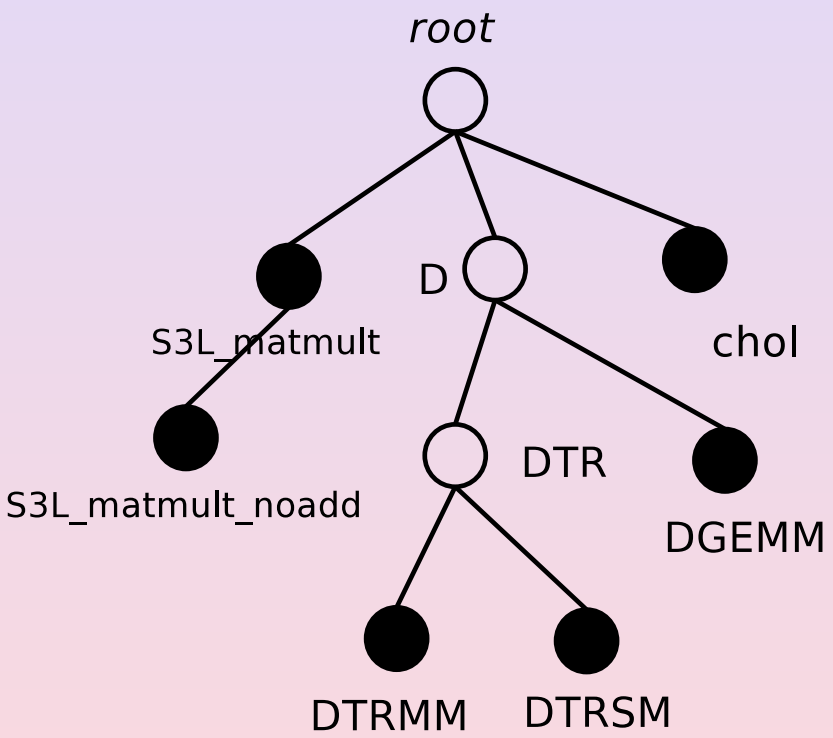
- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult, chol



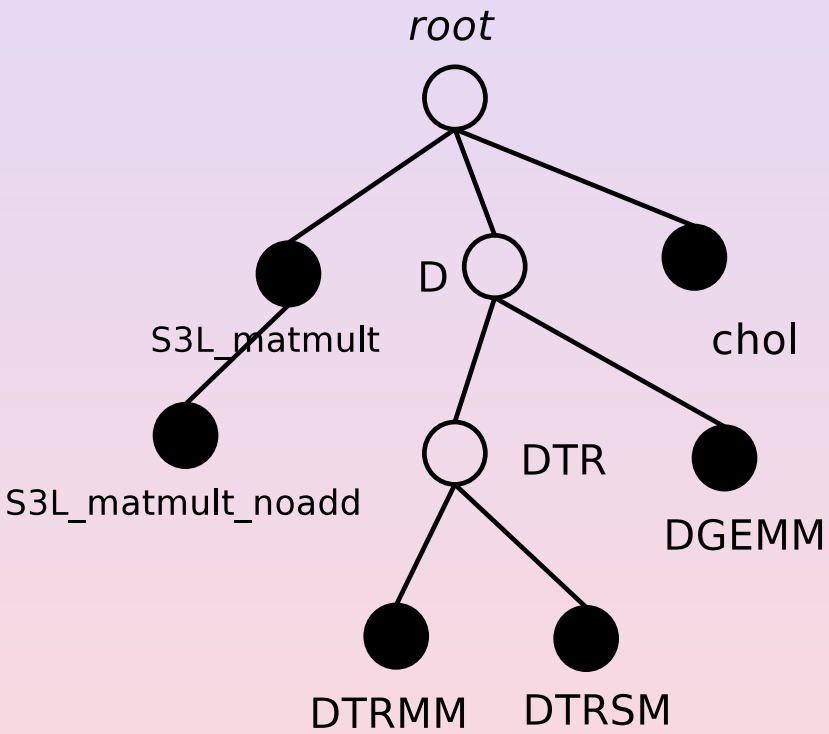
Exemple : insertion

Arrivée consécutive des services

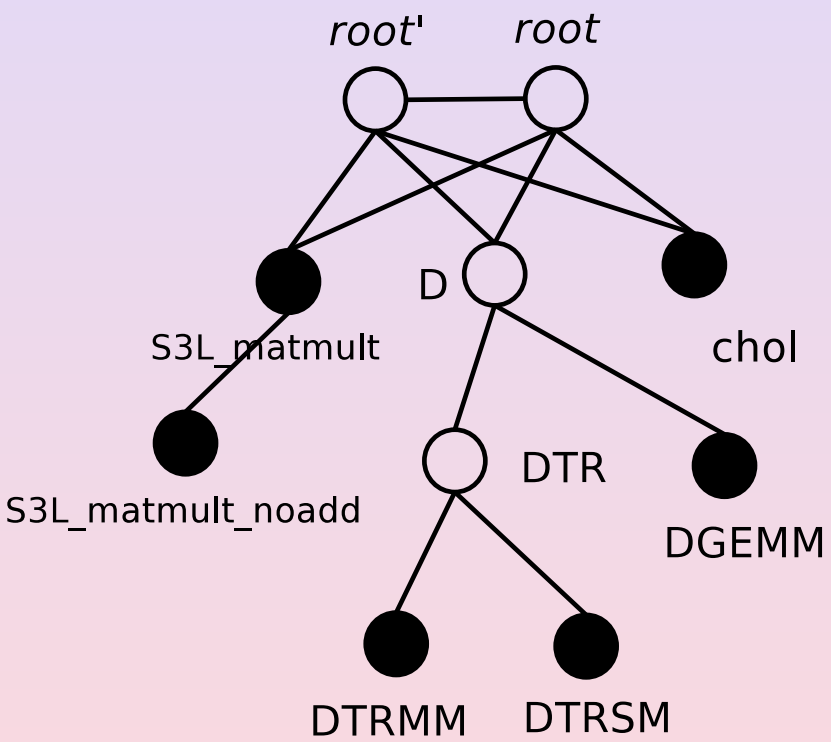
- DGEMM
- DTRSM
- DTRMM
- S3L_matmult_noadd
- S3L_matmult, chol



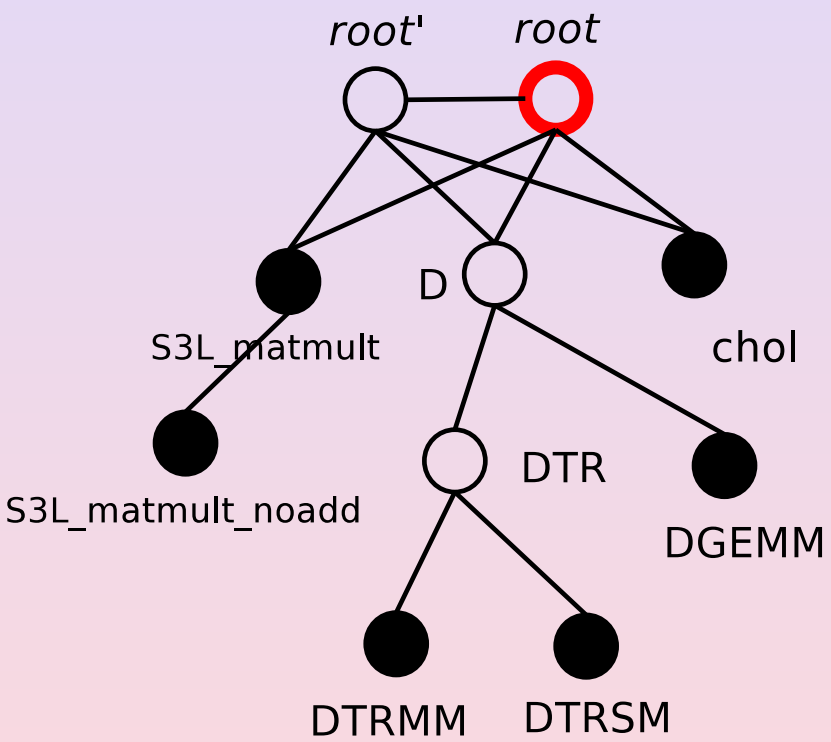
Exemple : réplication et arbre couvrant



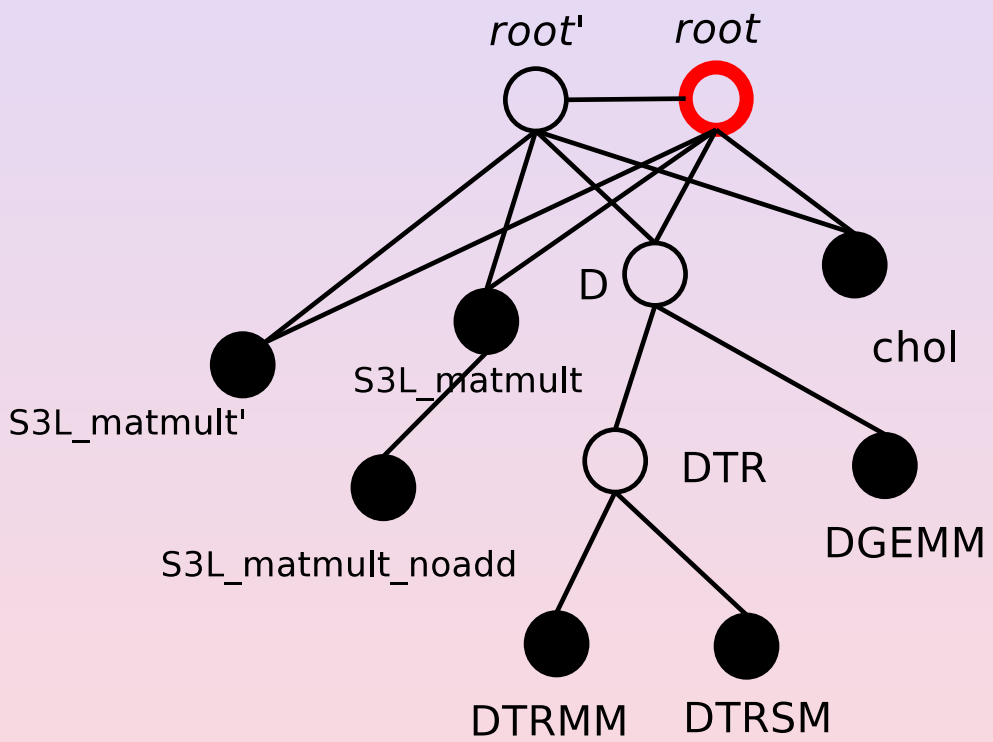
Exemple : réplication et arbre couvrant



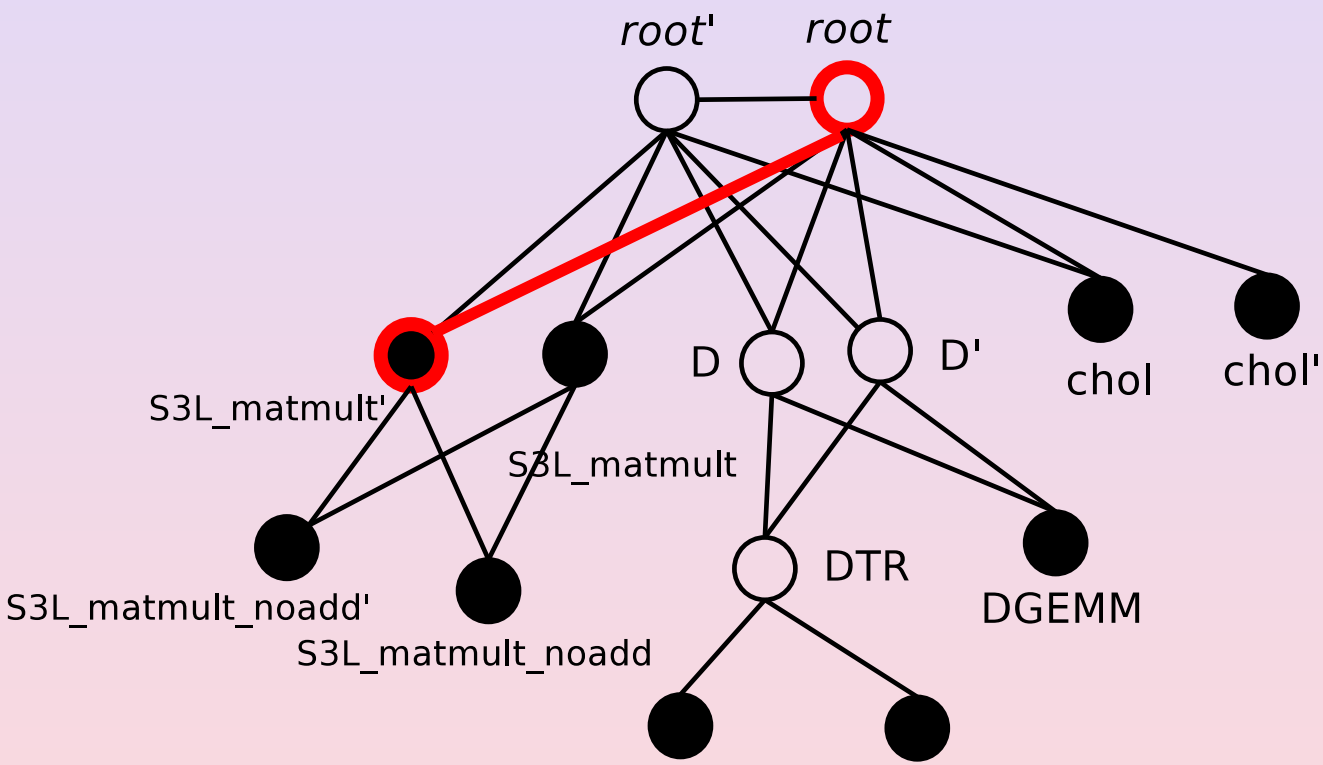
Exemple : réplication et arbre couvrant



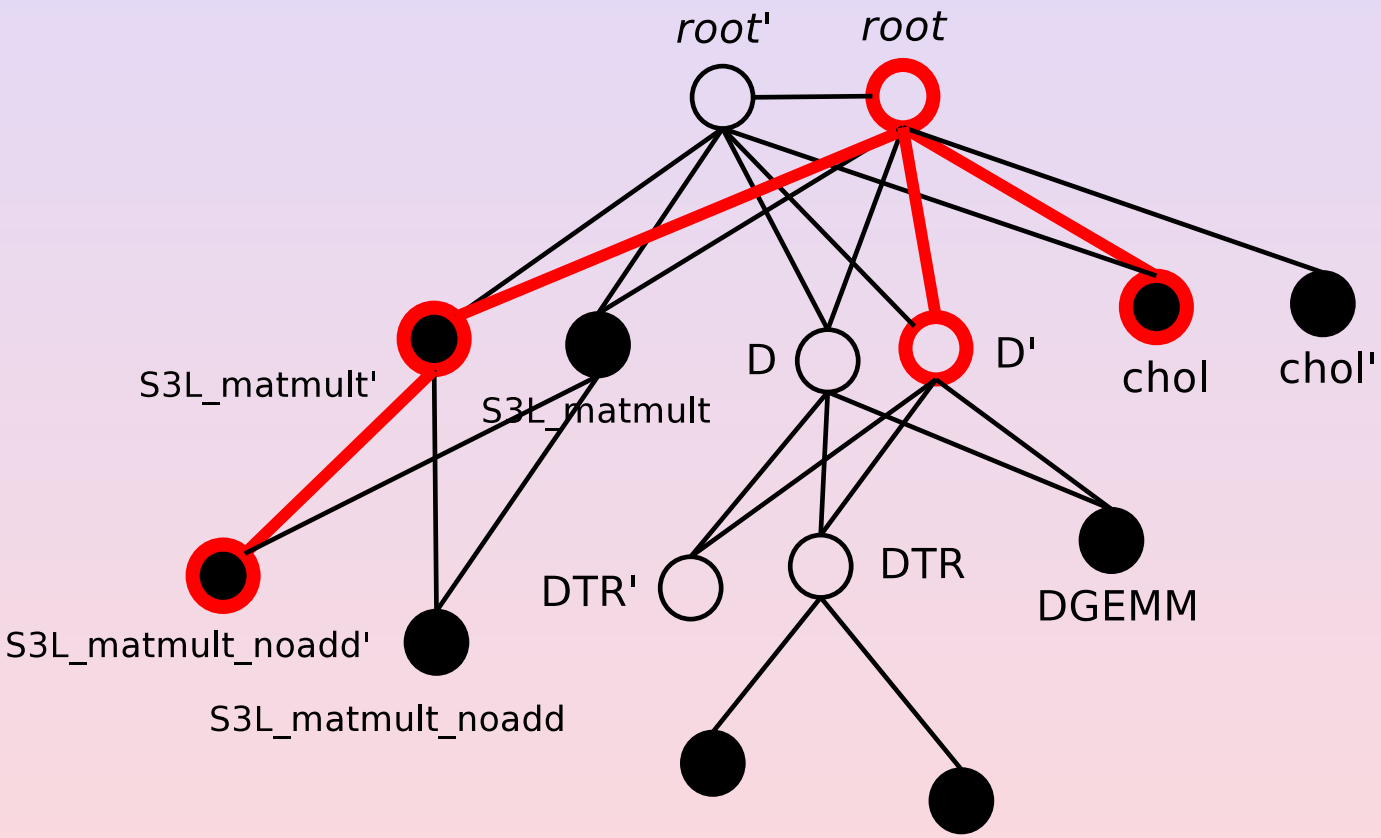
Exemple : réplication et arbre couvrant



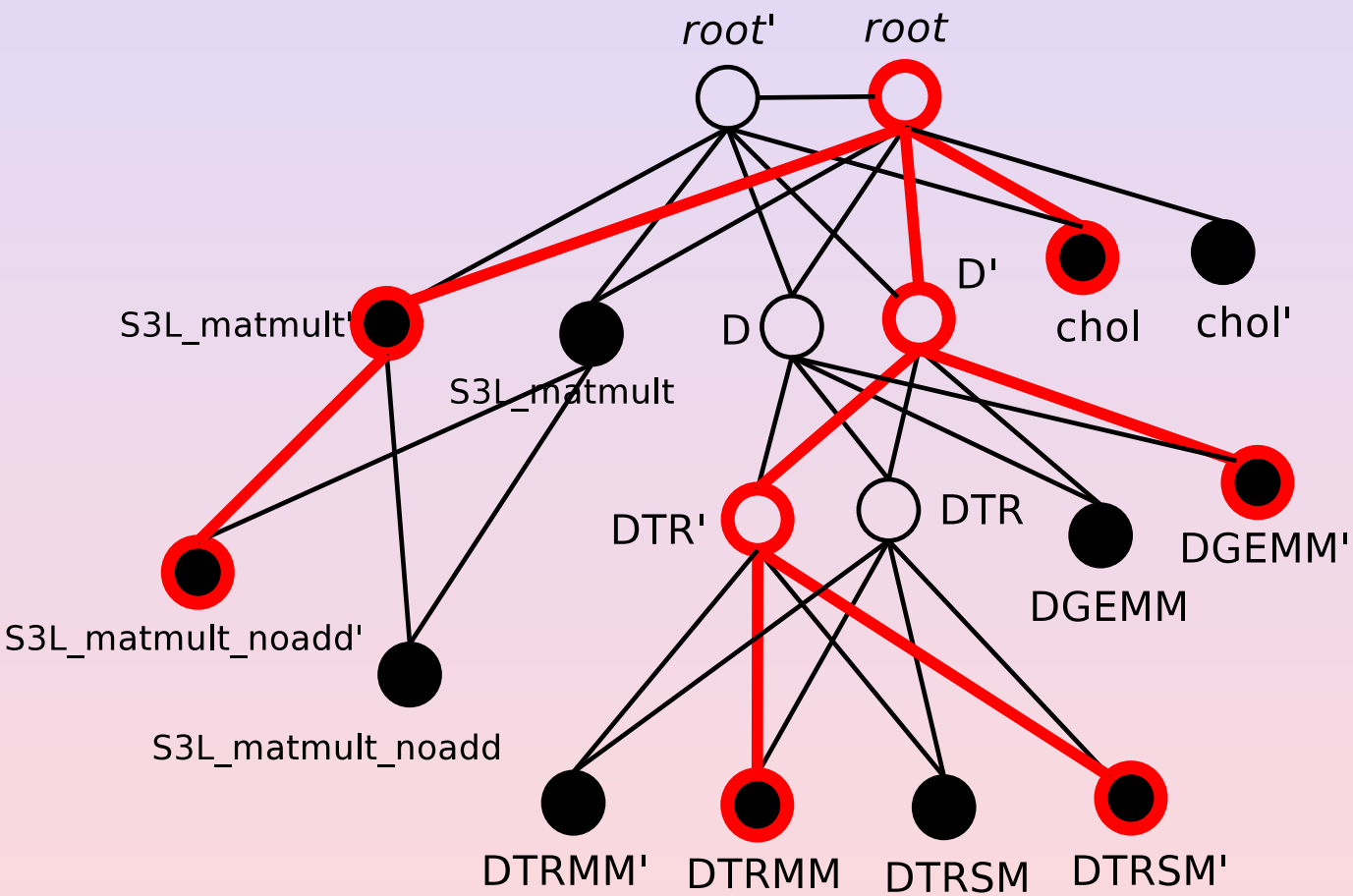
Exemple : réplication et arbre couvrant



Exemple : réplication et arbre couvrant

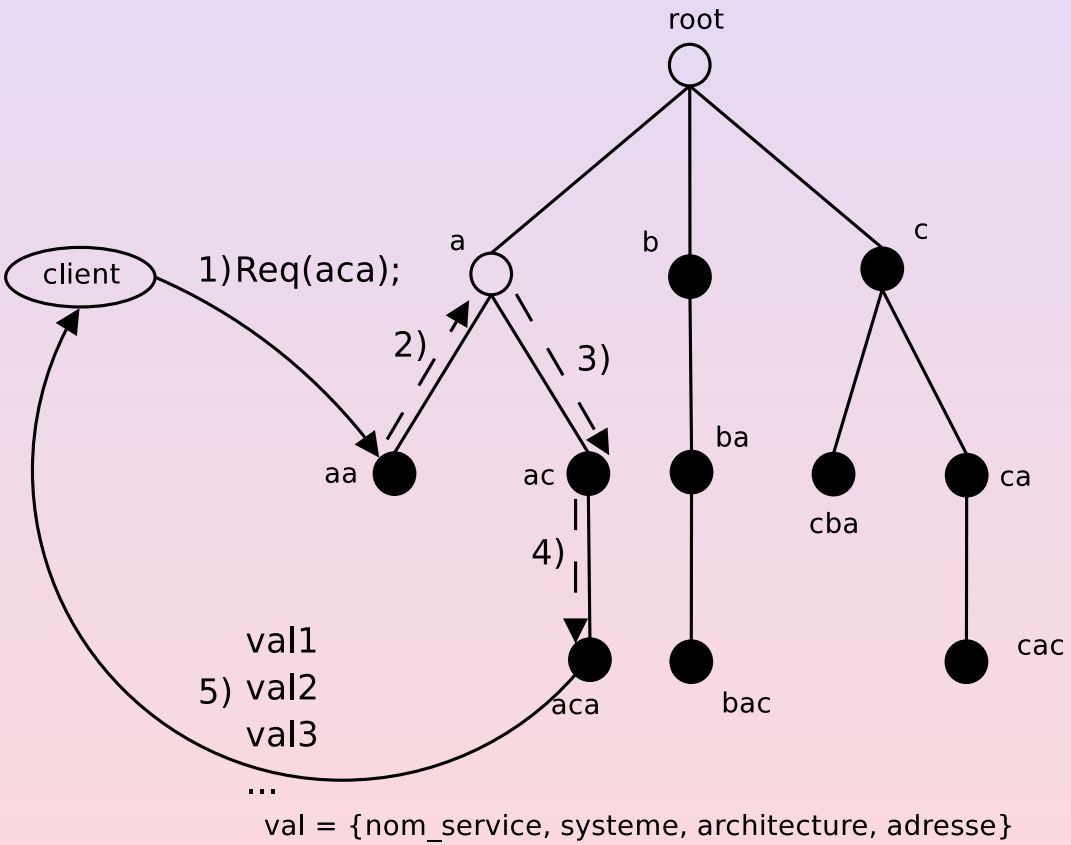


Exemple : réplication et arbre couvrant



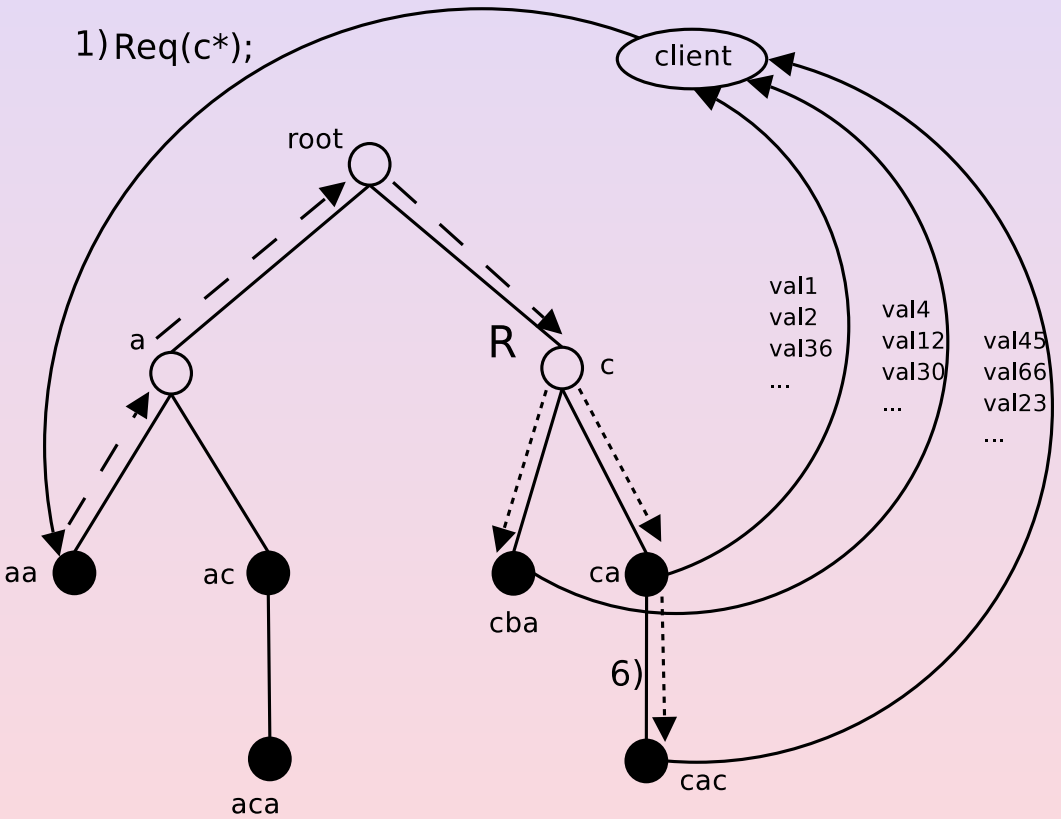
Mécanismes d'interrogation de la TPLD

- Recherche sur chaîne complète
- Recherche sur chaîne partielle
- Recherche multicritères



Mécanismes d'interrogation de la TPLD

- Recherche sur chaîne complète
- Recherche sur chaîne partielle
- Recherche multicritères

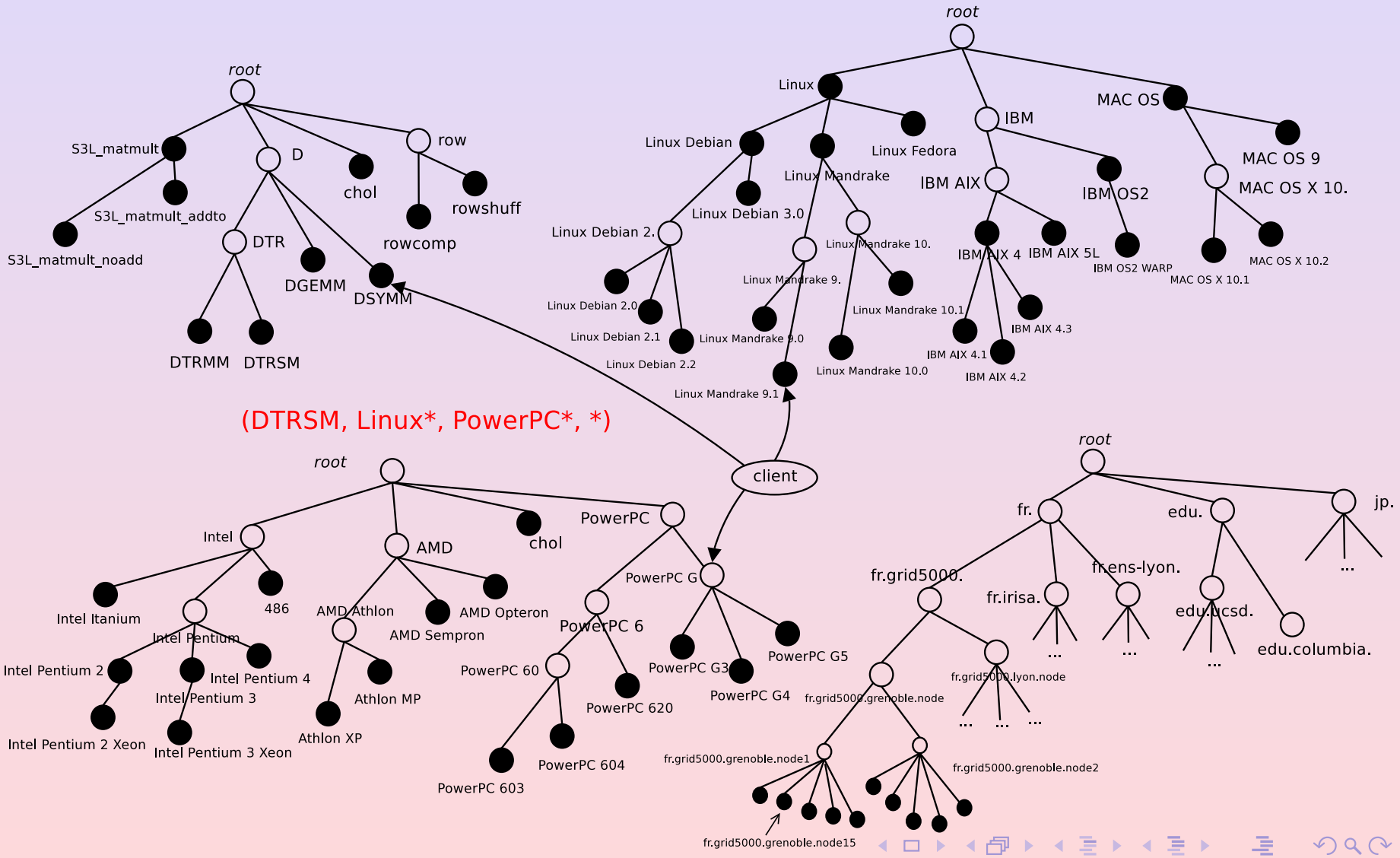


val = {nom_service, systeme, architecture, adresse}

Mécanismes d'interrogation de la TPLD

- Recherche sur chaîne complète
- Recherche sur chaîne partielle
- Recherche multicritères

Mécanismes d'interrogation de la TPLD



Mécanismes de cache

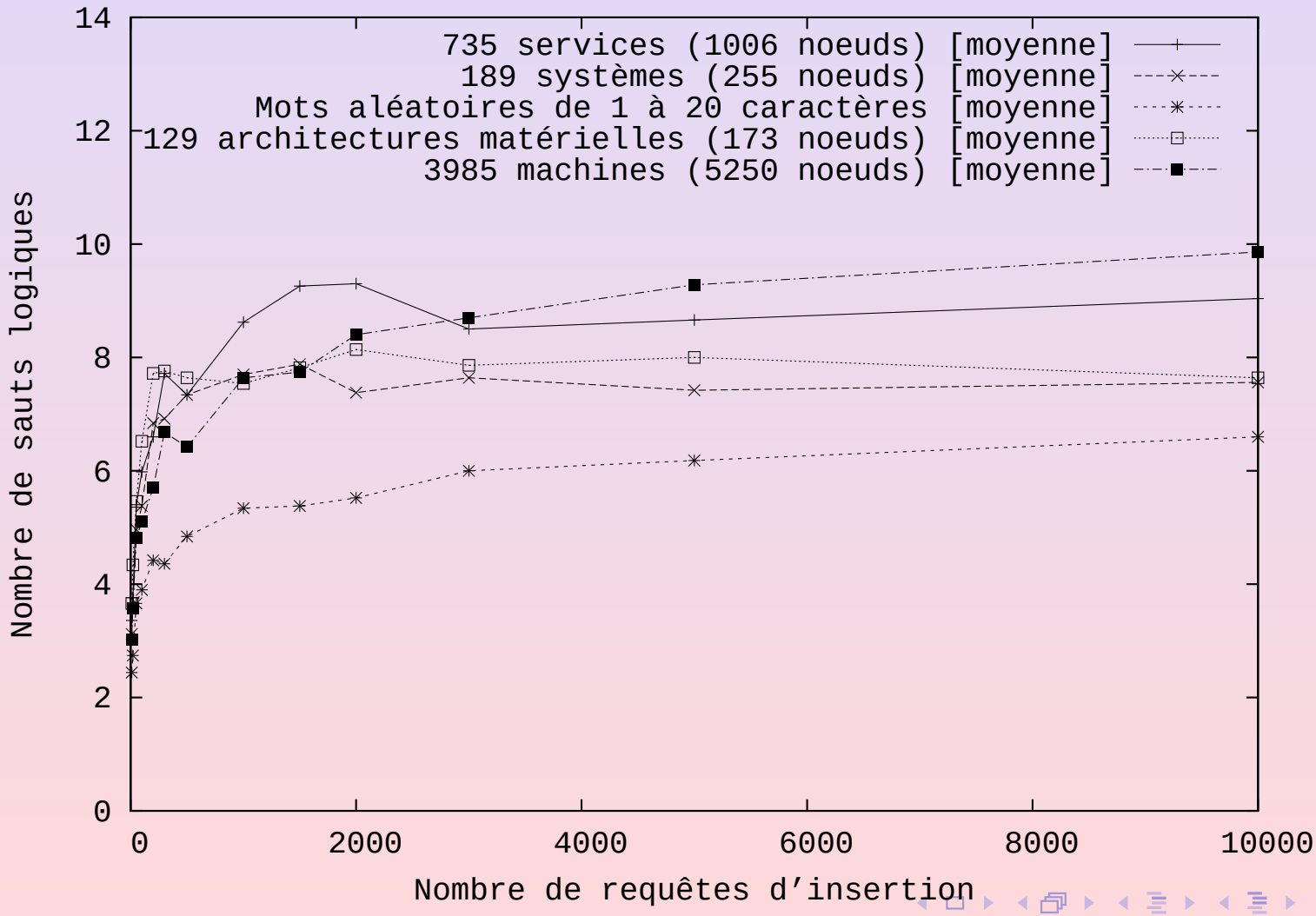
- A la demande
- Cache lors d'une requête sur chaîne partielle
  - réplication sur la racine du sous-arbre concerné
  - $\Rightarrow$ réponse au client dès l'arrivée sur la racine
- Cache lors d'une requête sur chaîne complète
  - problème des clefs populaires
  - sur la route de la requête
  - routage en arrière
- Politique LRU par pair

Plan

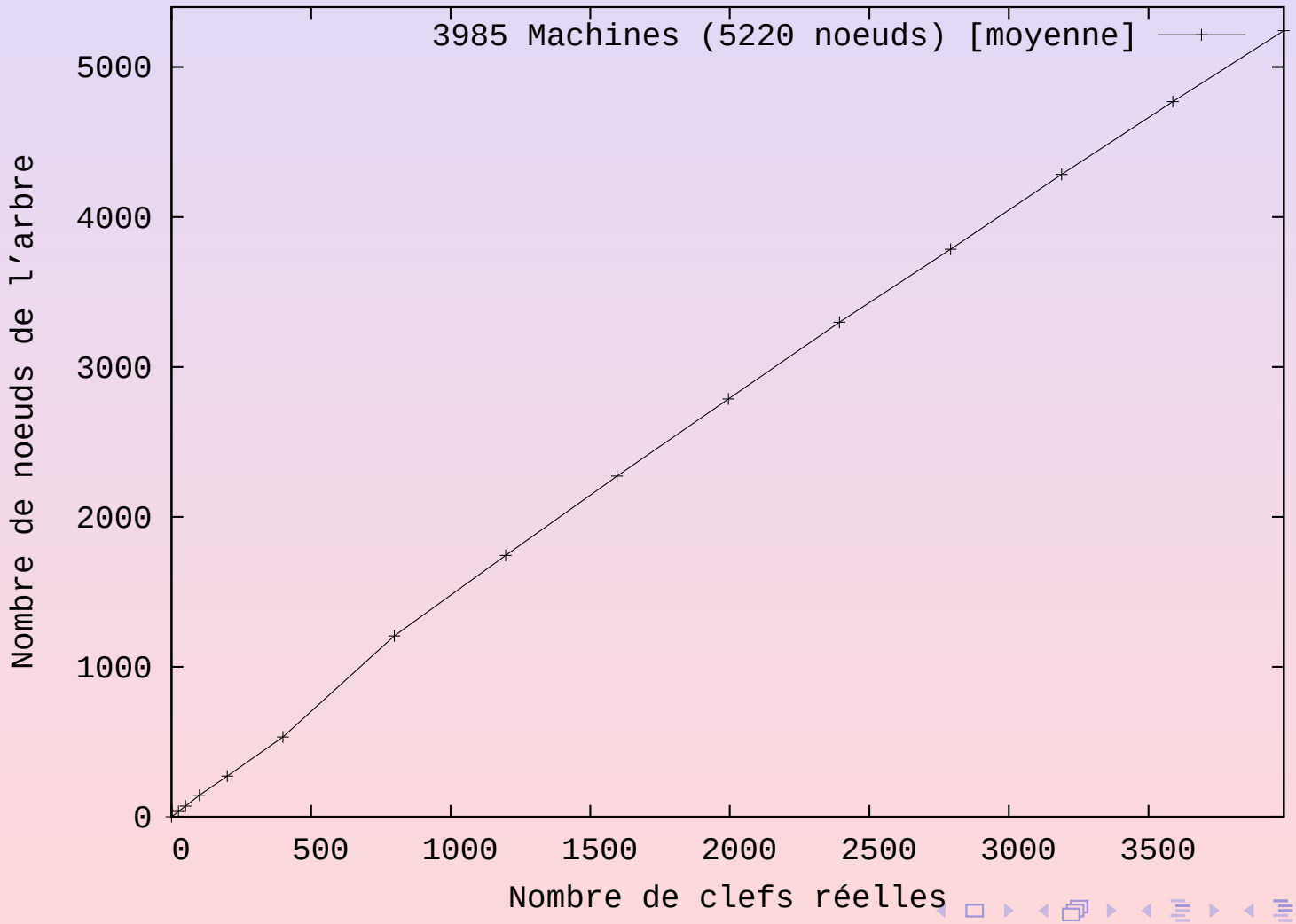
Complexités de la TPLD

- Arbre lexicographique
  - T : taille max des mots
  - A : alphabet utilisé
- Complexité du routage
  - bornée par $2T$, indépendant de la taille du réseau
  - en $O(log(N))$ pour le réseau le plus grand possible
- Taille de la table de routage
  - bornée par $Card(A)$, indépendant de la taille du réseau
  - en $O(\sqrt{N})$ pour le réseau le plus grand possible
- Choix local du routage
  - une case de tableau par caractère possible de A
  - en $O(1)$

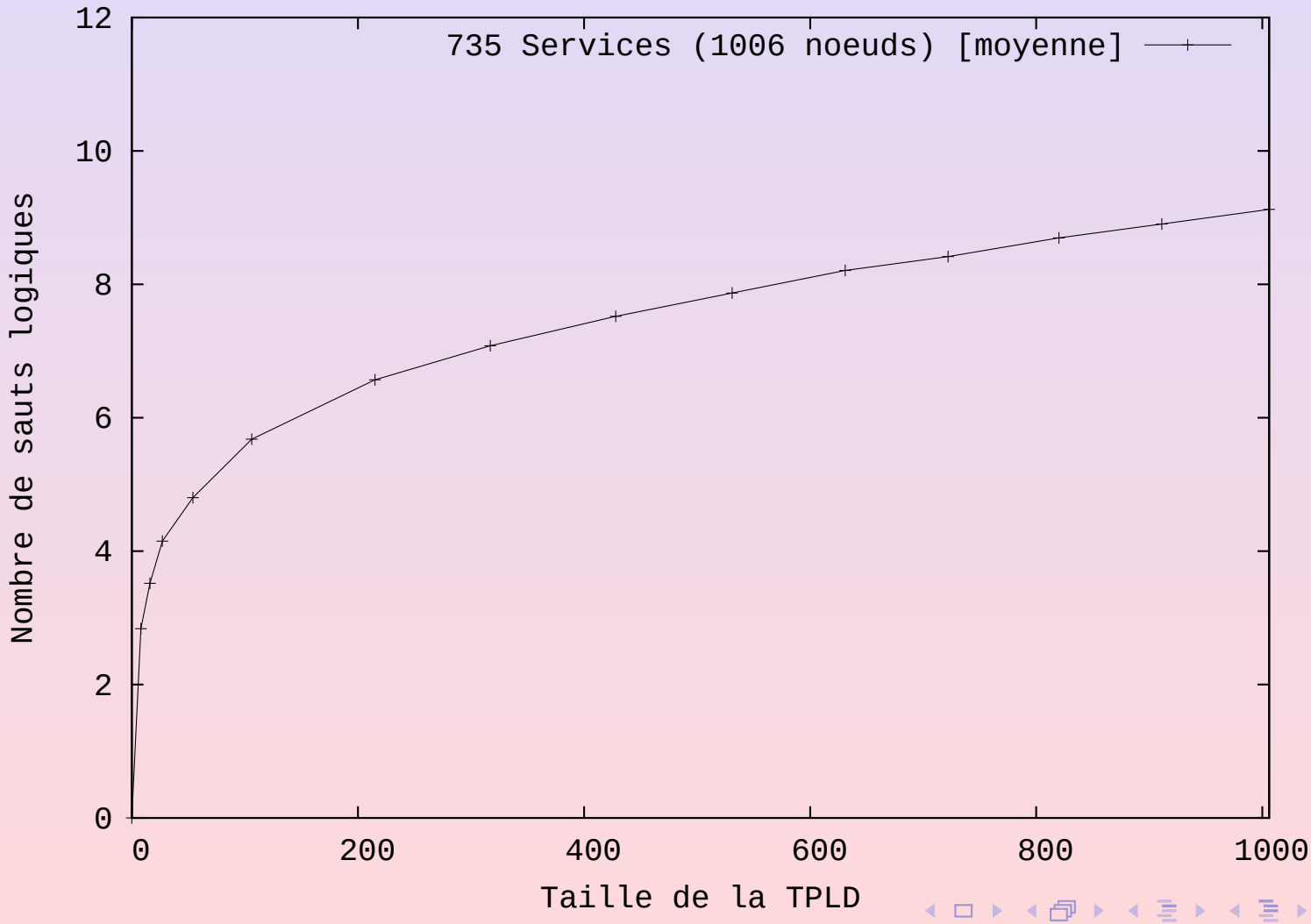
Taille de l'arbre / nombre d'insertions



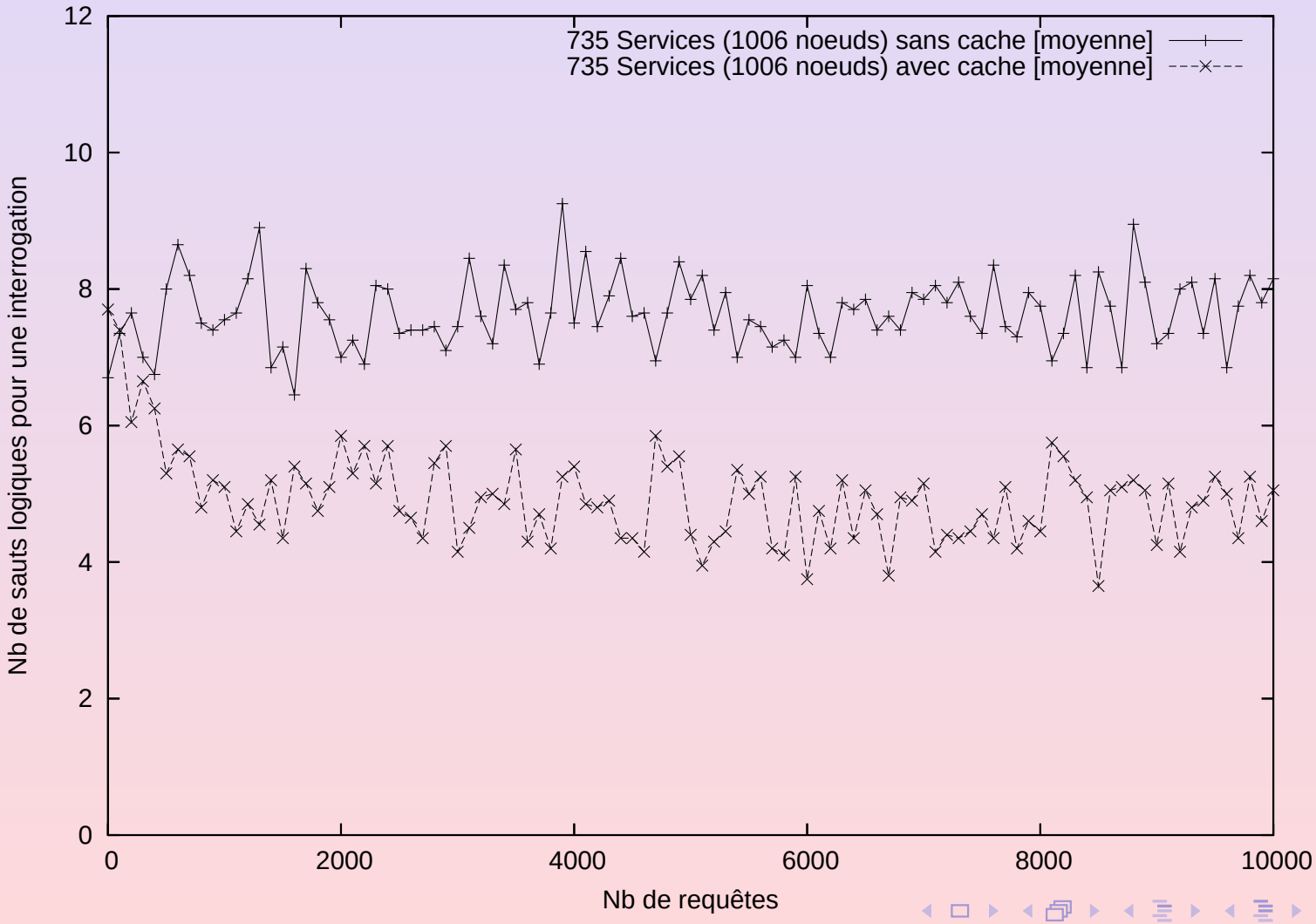
Proportionnalité clefs réelles / taille de la TPLD



Routage d'une requête d'interrogation



Cache (requête sur chaîne complète)



Complexités (TPLD/DHT)

DHT	Taille table routage	Taille routage	Décision routage
CAN	$O(\log(N))$	$O(\log(N))$	$O(\log(N))$
Chord	$O(\log(N))$	$O(\log(N))$	$O(\log(N))$
Pastry/Tapestry	$O(\log(N))$	$O(\log(N))$	$O(1)$
TPLD	$O(Card(A))$	$O(T)$	$O(1)$

Fonctionnalités (multi-TPLD/Squid)

-	SQUID	Multi-TPLD
Requêtes multicritères	X	X
Chaînes partielles	X	X
Nombre variable de dimensions	-	X
Adaptation au réseau physique	-	X
Optimisation par cache	-	X

Plan

Conclusion

- Contribution : TPLD
  - architecture distribuée pour la découverte de services dans un environnement dynamique à large échelle
  - dynamique
  - tolérante aux pannes (par k-réplication)
  - s'adaptant aux performances du réseau (arbre couvrant)
  - optimisations (cache)
- Perspectives
  - ordonnancement
  - gestion de données
  - implantation