Le projet ReMiX et ses applications en génomique

F. Raimbault [1] D. Lavenier [2]

[1]UBS / VALORIA

[2]IRISA / Symbiose

Journées ASIM des 27 et 28 Mai 2004 à Nantes

Le projet ReMiX

- réponse à l'appel à proposition ACI Masses de Données 2003
- septembre 2003 à septembre 2006
- 4 équipes participantes
- recherche par le contenu dans de grandes masses de données
- avec indexation pour limiter l'espace de recherche
- lever le verrou technologique de la taille de la mémoire
  1. conception d'une mémoire *spécialisée* de *très grande taille*
  2. réalisation d'un environnement de programmation ad-hoc
  3. validation par de nouveaux algorithmes sur :
     - les bases de données multi-média
     - les bases de données semi-structurées XML
     - les bases de données génomiques

Vue d'ensemble

Présentation du projet ReMiX

La recherche par le contenu et indexation

- caractéristiques des applications visées
  - volume des données conséquents (plusieurs *Go*)
  - requête de recherche par le contenu
  - extraction d'un ensemble de données similaires
  - calcul de distance (coûteux)
- utilisation d'un index
  - association de plusieurs données à chaque entrée
  - calcul d'une (ou plusieurs) entrée(s) à partir de la requête
  - limitation de l'espace de recherche aux données indexées
- contribution de ReMiX :
  associer mémoire d'index et opérateurs de calcul pour l'extraction rapide des données pertinentes

Verrou technologique de la mémoire

- taille de l'index supérieure au volume des données ($\times 10$)
  ✍ l'index ne tient pas en totalité dans la RAM
- accroissement du volume des données supérieur à l'accroissement des capacités d'intégration des RAM
- la mémoire secondaire est 1000 fois plus lente
- prise en compte des niveaux de hiérarchie mémoire (registres, cache niv.1, cache niv.2, RAM, disques, ...)
  - complexification des algorithmes
  - pas de solution générale (portabilité faible)
  - performances aléatoires
    - coût de lecture non constant
    - influence des accès antérieurs

La solution explorée

- Conception d'un système matériel, dédié à l'indexation, basé sur une mémoire vive de très grande taille ($[100\,Go, 1\,To]$) et sur des composants reconfigurables
- Contributions nécessaires dans plusieurs domaines :
  - architecture de machine spécialisée
  - environnement de programmation et compilation
  - algorithmique de l'indexation
    - en indexation d'images
    - en recherche documentaire
    - en génomique

Architecture de machine

grande mémoire monolithique physiquement impossible

- une mémoire physiquement distribuée
  - extensibilité pour suivre la croissance des données
  - parallélisation des accès
- exploitable si
  - opérateurs performants et en nombre suffisant

    composants ▸ reconfigurables
  - bande passante suffisante

    coupler étroitement une RAM et un FPGA
- architecture développée pour ReMiX
  - carte ▸ RMEM : (FPGA + connecteur PCI) $\oplus$ mémoire FLASH
  - ▸ système ReMiX : réseau de PC dotés de carte(s) RMEM

Environnement de programmation

- système parallèle hétérogène, partiellement spécialisable
  - cluster de PC
  - composants reconfigurables
- outils de niveau d'abstraction très différents
  - langages de programmation (C, C++, Java, ...)
  - bibliothèques MPI, OpenMP, RMI ...
  - outils de CAO et HDL
- domaines d'application distincts, fonctionnalités similaires
  - mémoire d'index répartie
  - opérateur de calcul de distance

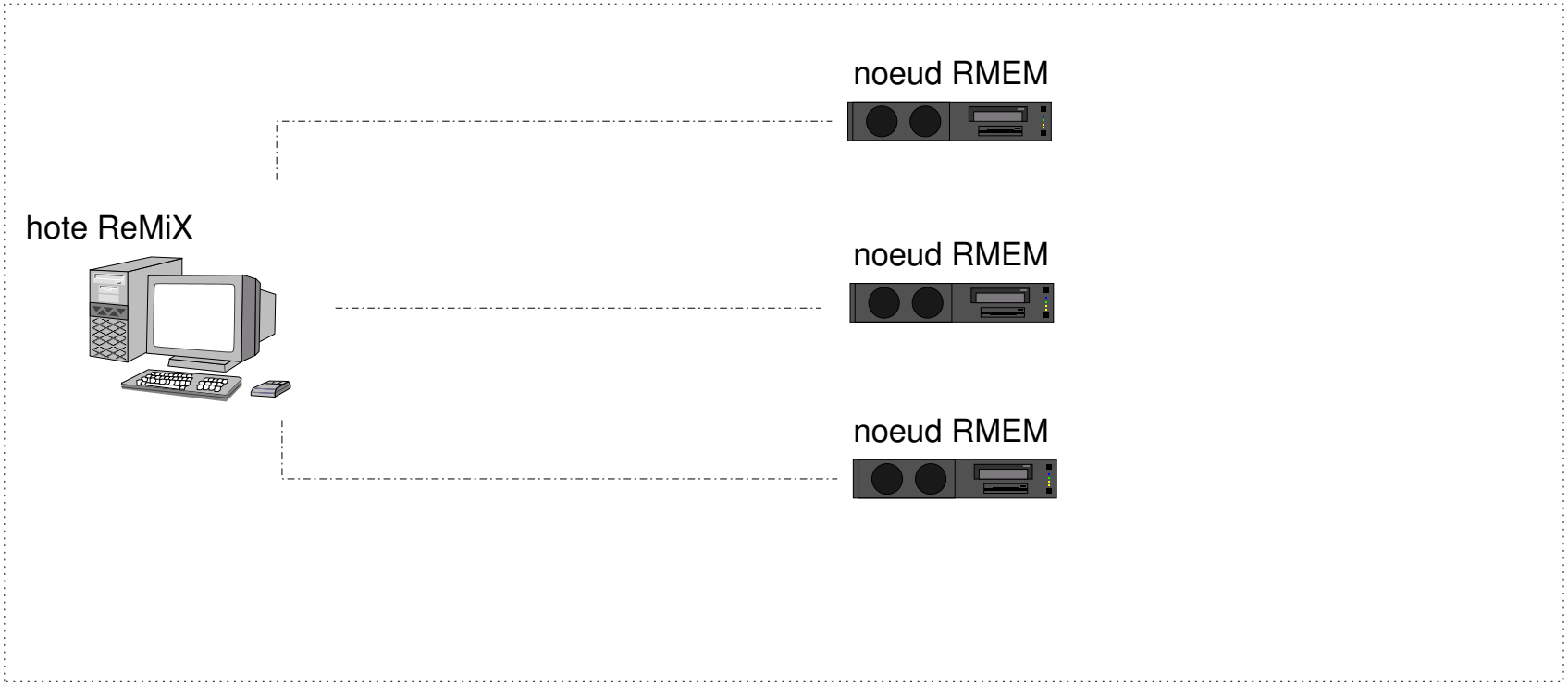
✌ approche *framework*

Cadre de conception

- ni un nouveau langage, ni une bibliothèque : sorte de programme à trous
- un ensemble de composants
  - adaptables aux domaines d'applications visés
  - dont les interactions totalement prédéfinies

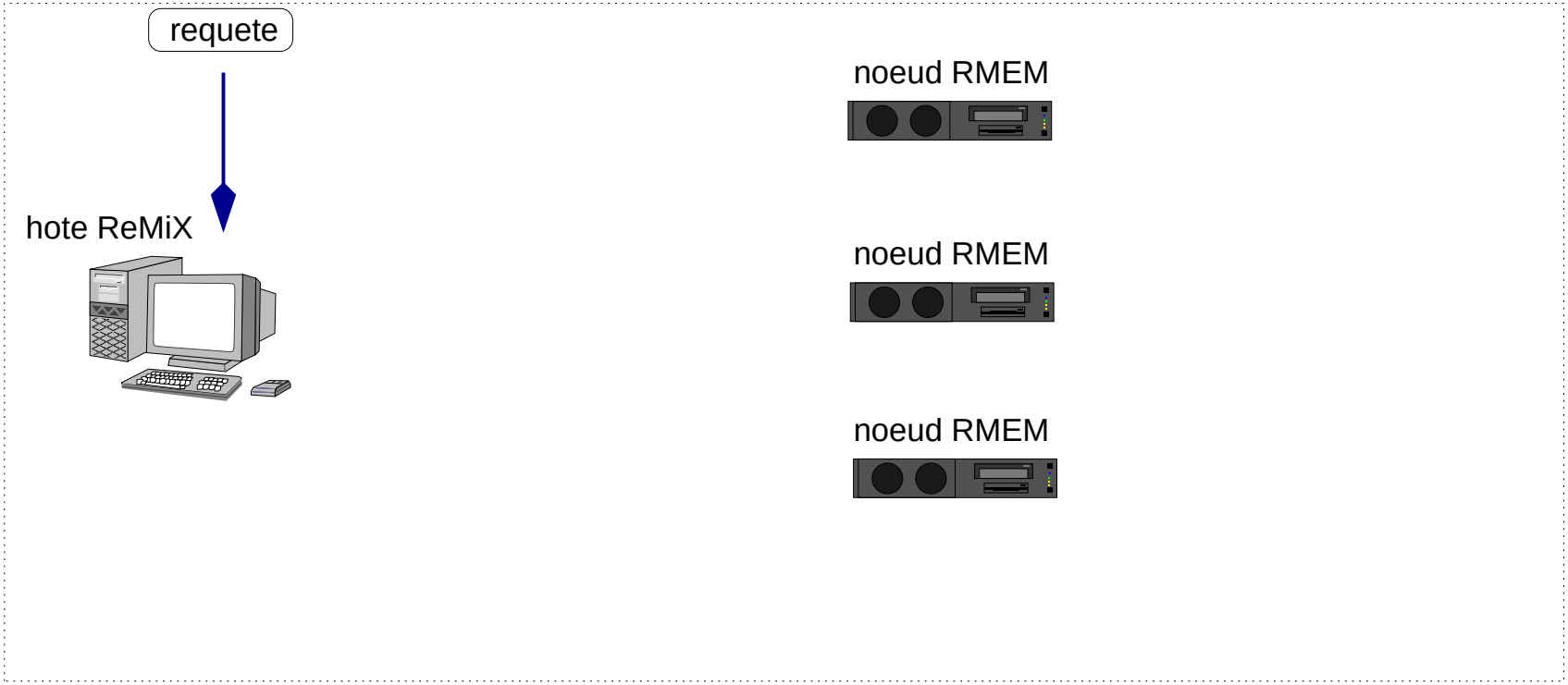
  ✌ modèle de fonctionnement *serveur de requêtes*
- terminologie :
  - requête: donnée pour laquelle on recherche des informations similaires dans une BD
  - sous-requête : partie d'une requête adressant une entrée
  - résultat partiel : résultat d'une sous-requête
  - résultat final : fusion des résultats partiels

Modèle serveur de requêtes



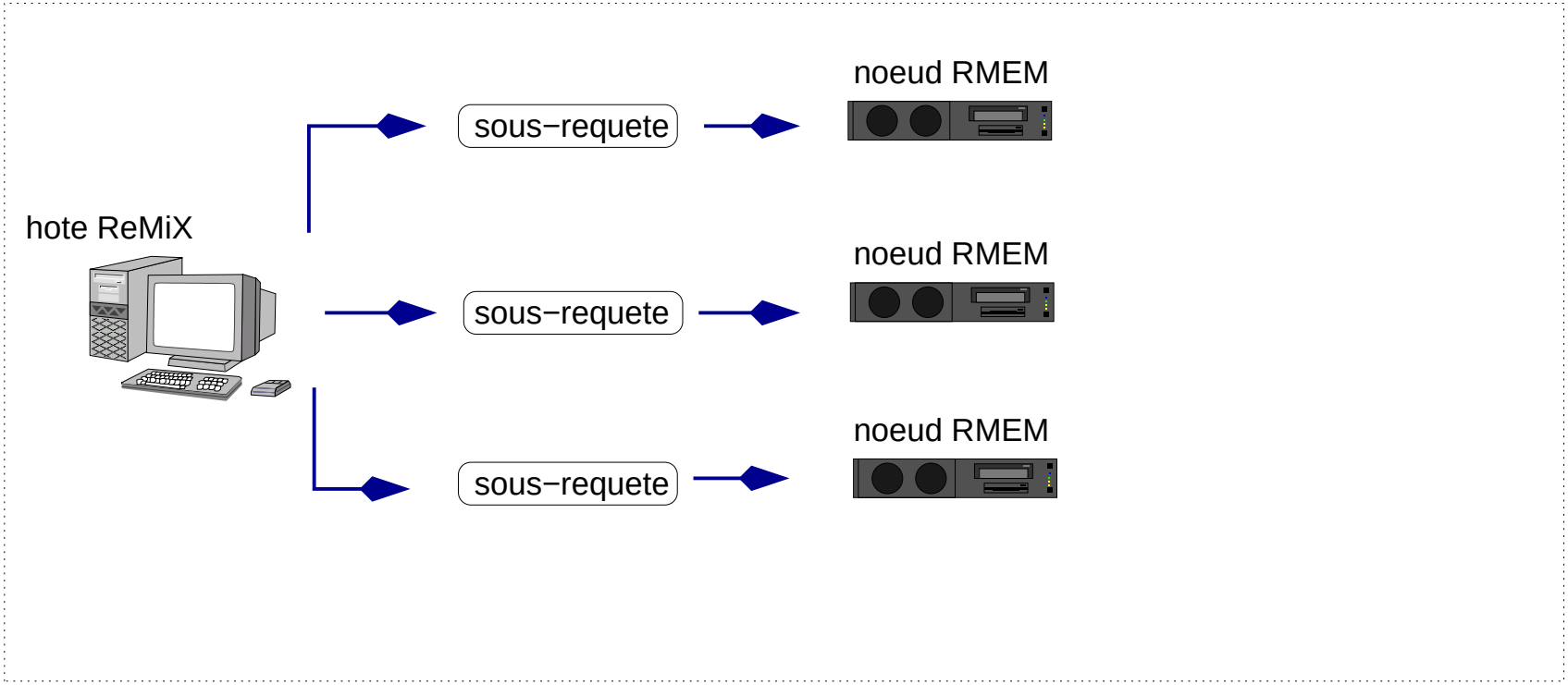
répartition ou duplication de l'index sur les noeuds

Modèle serveur de requêtes



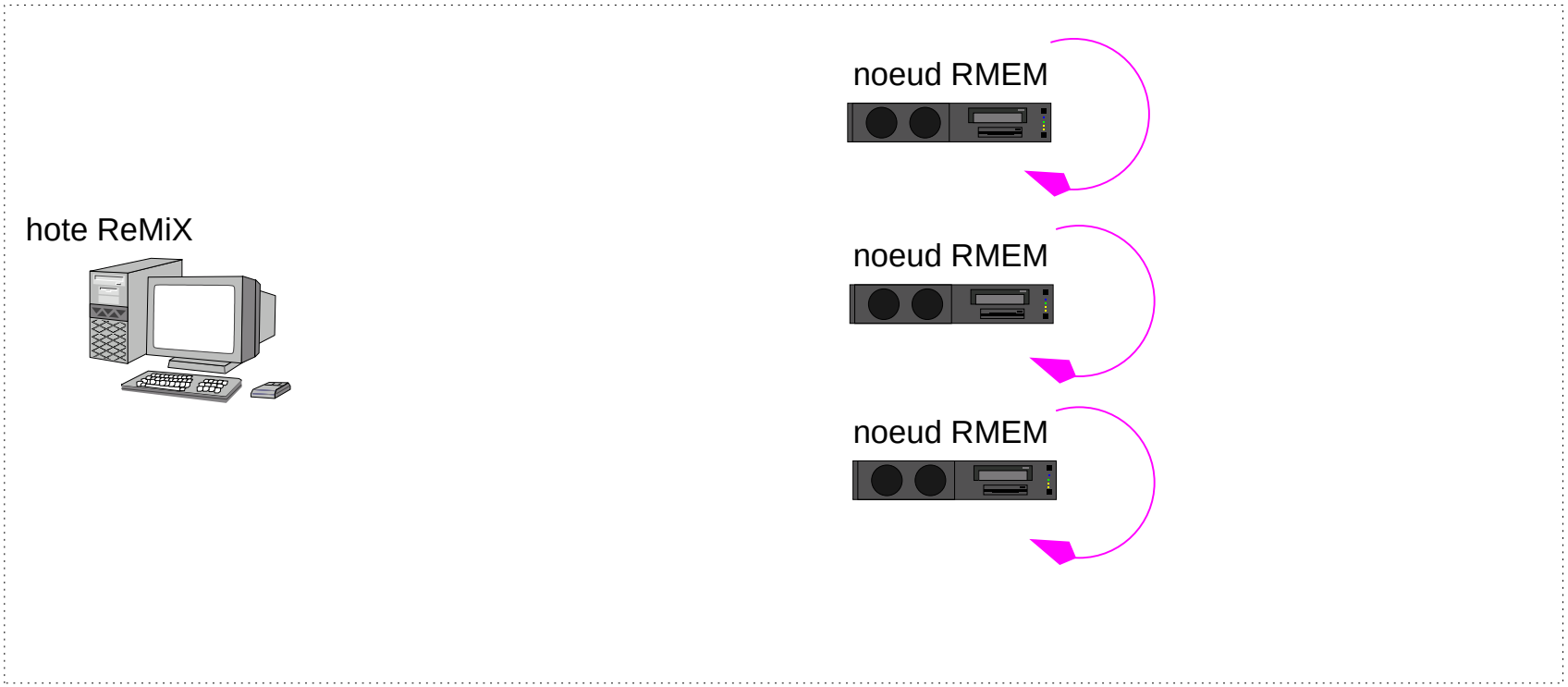
✍ sur l'hôte: carte de la répartition de la mémoire d'index

Modèle serveur de requêtes



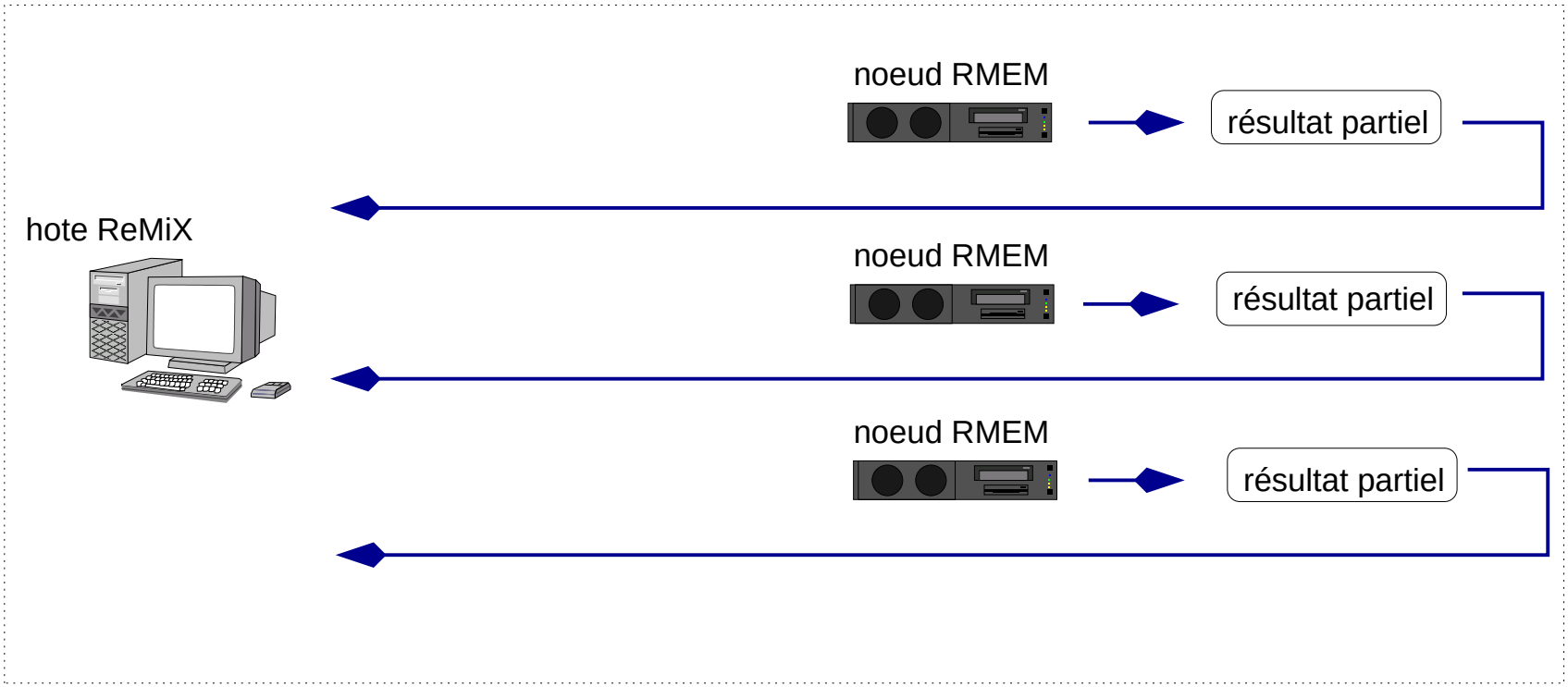
adressage des sous-requêtes basée sur la répartition

Modèle serveur de requêtes



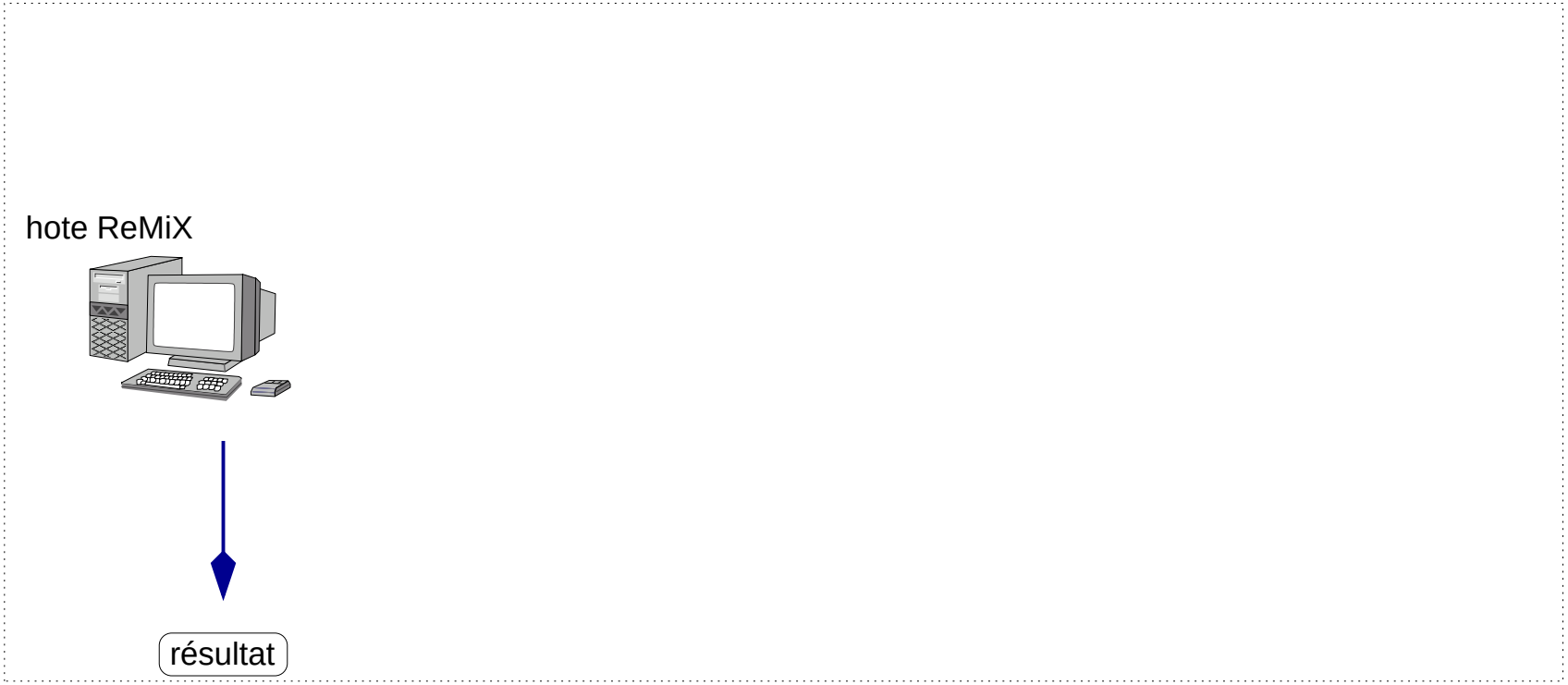
traitement des sous-requêtes

Modèle serveur de requêtes



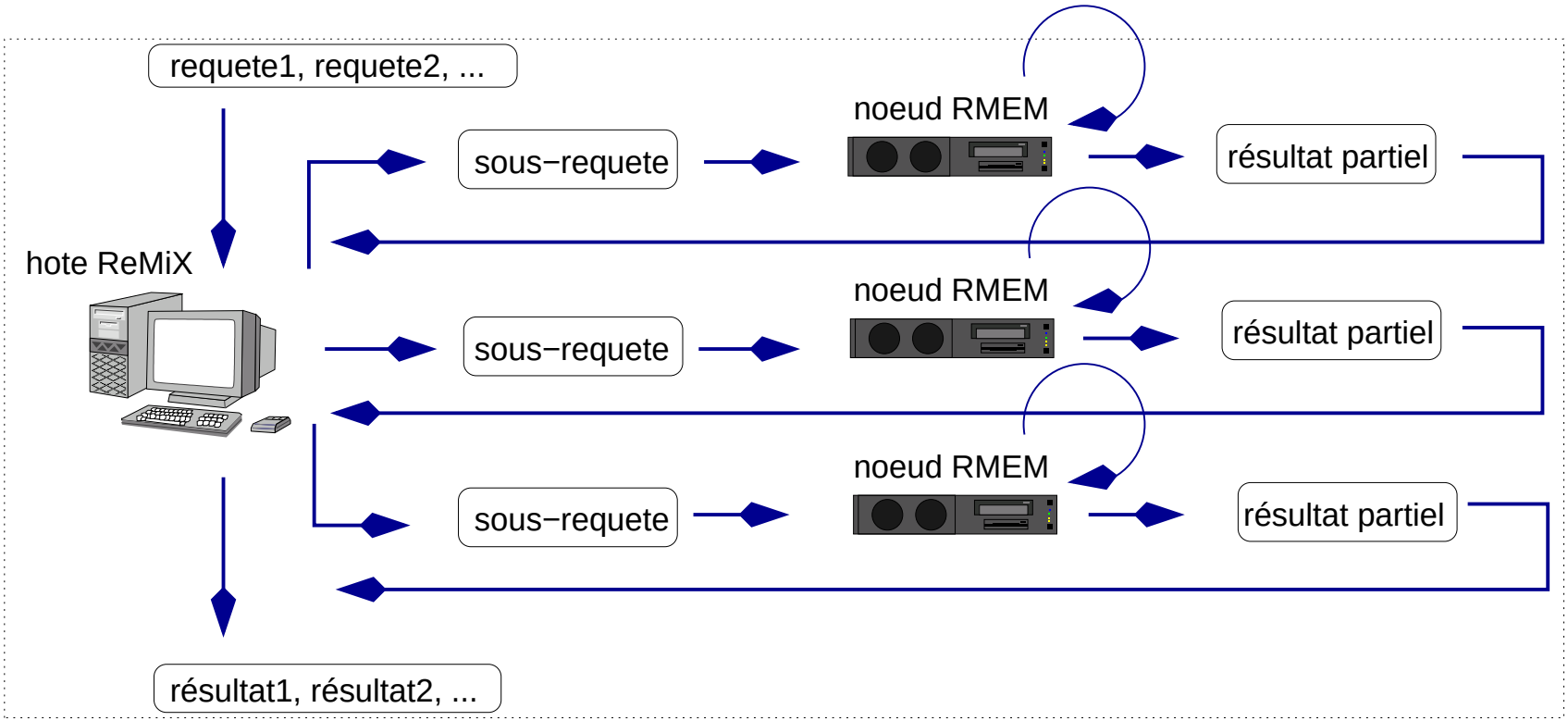
un ou plusieurs résultats partiels par sous-requête

Modèle serveur de requêtes



résultat final généré sur l'hôte

Modèle serveur de requêtes



fonctionnement en continu et en recouvrement

Ce que spécifie le programmeur

Ce qui est spécifique à chaque application :

- la construction et la répartition de l'index
- le contenu et le découpage d'une requête
- le contenu et le traitement d'une sous-requête
- traitement d'un résultat partiel
- la rétro-action d'un résultat partiel sur une requête
- la fusion des résultats partiels
- le contenu et le traitement du résultat complet

Ce que fournit le framework

Ce qui est commun à toutes les applications :

- le serveur de requête
- les serveurs de sous-requêtes et celui des résultats partiels
- les communications et les synchronisations entre serveurs
- la gestion des requêtes, sous-requêtes, résultats partiels
- la construction de la carte de la mémoire

▸ package  `remix` indépendant de l'architecture cible

Algorithmique de l'indexation

- type de données
  - indexation d'images : 1000 descripteurs / image
  - recherche documentaire : 1 arbre et 1 liste inverse / document
  - génomique : 1 chaîne de caractères / séquence
- repenser l'indexation
  - organisation des données
    - accès direct (aléatoire vs séquentiel)
    - accès parallèle (duplication vs concurrence)
    - structures précalculées
  - traitements
    - parallélisation (par les données)
    - spécialisation (des tâches)

Application à la génomique

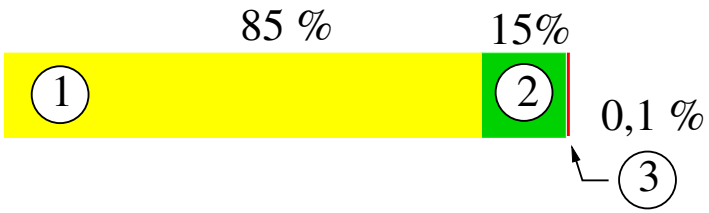
BLAST

- pour se faire la main sur une application simple et utile . . .
- *Basic Local Alignment Search Tool.*
  S.F. Altschul, W. Gish, W. Miller, E. W. Myers et al.
  Journal of Molecular Biology, 215:403-10, 1990
- 100 000 utilisations en ligne par jour via le web de NCBI
- BLASTN: recherche de similarité entre une séquence et celles d'une banque d'ADN.
- recherche séquentielle *exhaustive* d'alignements
  1. recherche de points d'ancrage
  2. extension pour des alignements sans insertion/omission
  3. extension pour des alignements avec gap
- ▸ exemple

Expérimentations avec (NCBI) BLASTN

- requête de taille $[10K, 1M]nt$ sur banque de $1,16Gbp$
- mesure du temps passé dans les différentes phases :

1. ancrage
2. extension sans gap
3. extension avec gap



✌ accélérer les phases 1 *et* 2 de BLASTN

1. indexation de tous les points d'ancrage possibles
2. opérateurs spécialisés de comparaison de séquence ▸ Loi d'Amdahl

iBlast

- iBlast $\equiv$ BLASTN avec index
- exemple:
  - banque publique du génome humain ($3.2Gbp$)
  - point d'ancrage de taille 10 et
    calcul de distance sans gap sur les $16nt$ suivants
- indexation de toutes les séquences de $10nt$
  $\Rightarrow 4^{10} \approx 1M$ d'entrées dans l'index
- valeurs associées : les $16nt$ suivants et leur position
  $\Rightarrow \approx 4000$ valeurs par entrées
- pour une requête de taille $1000nt$:
  - BLASTN: $3.2G$ nt scannés sur disque
  - iBLAST: $4M$ nt scannés en RAM

iBlast sur ReMiX

- Construction de l'index
  - taille de l'index pour le génome humain : $32Go$
  - répartition des index sur les 16 noeuds de manière cyclique
- Accès aux point d'ancrage
  - requête: séquence S
  - sous-requête: tous les sous-séquences de S de taille 26
  - distribution des sous-requêtes en fonction des 10 premiers nt
- Comparaison sans gap
  - sur chaque serveur de sous-requête
  - opérateur matériel dans le FPGA pour le calcul de distance
- Comparaison avec gap
  - (très) faible quantité de résultats partiels renvoyés
  - opération finale d'alignement réalisée sur l'hôte

Conclusion

Résumé

le projet ReMiX

- exploitation des grandes masses de données
- conception d'un système matériel d'indexation pour accélérer
  - l'accès aux données pertinentes
  - *et* les traitements à réaliser sur les données lues
- réalisation d'un environnement de programmation
  - parallélisme explicite mais implicitement géré
  - développement des algorithmes avant la livraison de ReMiX
  - utilisable sur d'autres architectures (clusters de PC)
- démonstration sur de vraies applications
  - en indexation d'images
  - en recherche documentaire
  - en génomique

Appel à participation

- si vous avez un algorithme d'indexation dont
  - la mémoire d'index requise est collossale
  - les traitements impliquent un calcul de distance coûteux
  - vous voulez tester le passage à l'échelle
- nous pouvons
  - envisager ensemble son portage sur ReMiX
  - vous fournir un outil de programmation dédié
  - l'évaluer (pour l'instant) sur un cluster de PC

☞ mailto:
{frederic.raimbault|dominique.lavenier}@irisa.fr

Questions ?

Participants du projet ReMiX

Équipe Symbiose (IRISA)

- D. Lavenier (DR)
- F. Raimbault (MCF)
- S. Rubini (MCF)
- J. Xianyang (Post-doc)

Équipe TexMex (IRISA)

- L. Amsaleg (CR)

Équipe R2D2 (IRISA)

- F. Charot (CR)
- S. Derrien (MCF)
- G. Georges (ING)

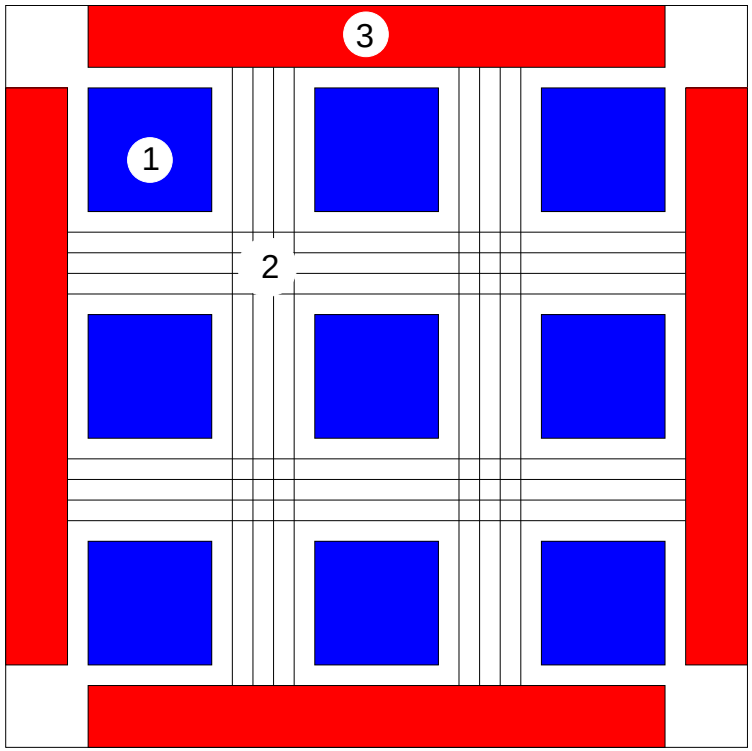
Équipe APRIM (VALORIA)

- P-F. Marteau (PR)
- G. Ménier (MCF)
- E. Popovici (DOC)

◂ introduction

Les composants reconfigurables

- circuit intégré contenant des fonctions élémentaires
  - de calcul et mémorisation
  - de routage
  - d'E/S
- technologie FPGA actuelle de $10M$ portes logiques
- reconfigurable (par programmation)



☞ accélération de plusieurs ordre de grandeur par rapport à un microprocesseur

◂ architecture

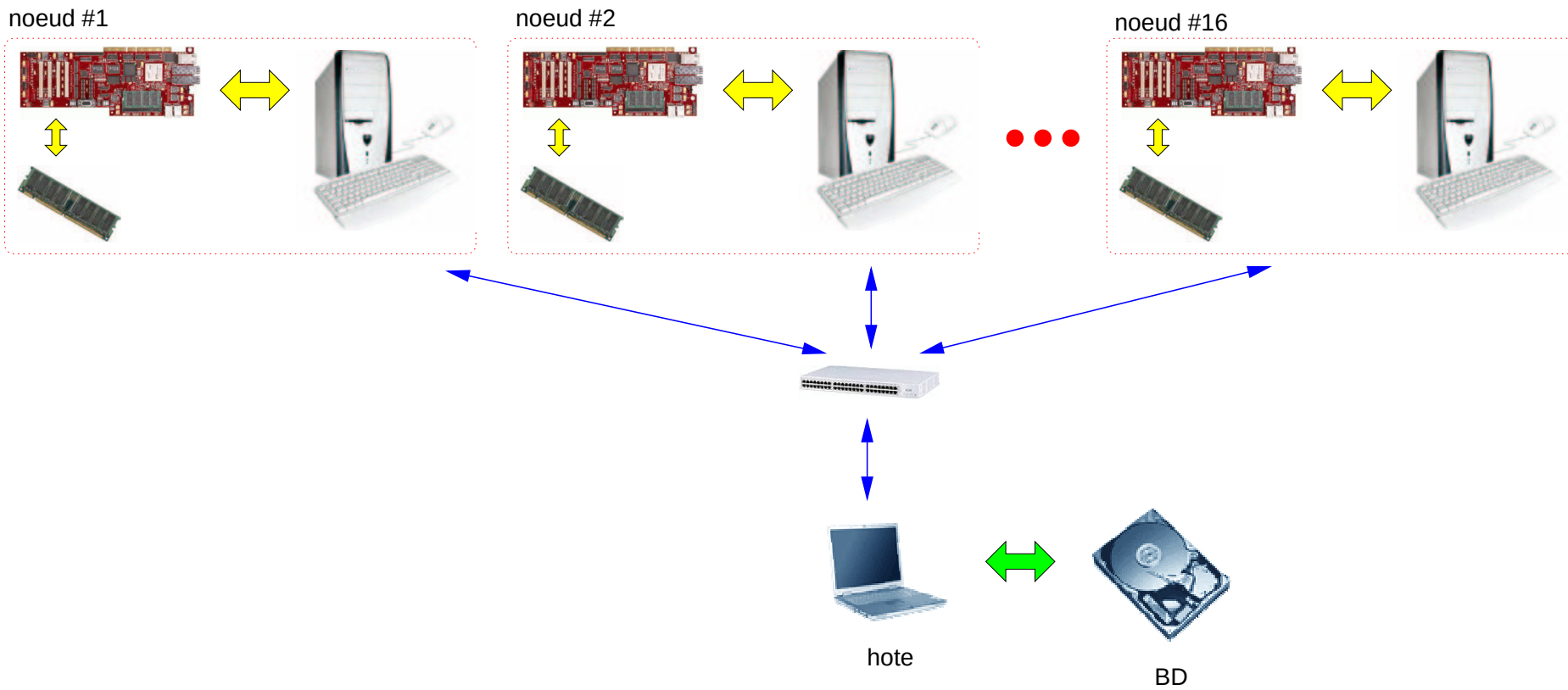
Exemple de carte RMEM

- basée sur une carte au format PCI contenant un FPGA

- conception d'une carte mezzanine contenant 16*Go* de mémoire FLASH

◂ architecture

Architecture du système ReMiX



architecture

Exemple d'alignement

BD: g t a c a t t g g t a c g t c c

S: a t t g a t c c t

recherche d'un point ancrage (de taille 4)

◂ blastn

Exemple d'alignement

BD:	g	t	a	c	a	t	t	g	g	t	a	c	g	t	c	c
S:	a	t	t	g	a	t	c	c	t							

recherche d'un point ancrage

◂ blastn

Exemple d'alignement

BD: g | t a c a | t t g g t a c g t c c

S: | a t t g | a t c c t

recherche d'un point ancrage

◂ blastn

Exemple d'alignement

BD: g t a c a t t g g t a c g t c c

S: a t t g a t c c t

recherche d'un point ancrage

◂ blastn

Exemple d'alignement

BD: g t a **c a t t** g g t a c g t c c

S: **a t t g** a t c c t

recherche d'un point ancrage

◂ blastn

Exemple d'alignement

BD: g t a c **a t t g** g t a c g t c c

S: **a t t g** a t c c t

recherche d'un point ancrage : terminé

◂ blastn

Exemple d'alignement

BD: g t a c **a t t g** g t a c g t c c

S: **a t t g** a t c c t

extension sans gap (au plus 1 substitution)

◂ blastn

Exemple d'alignement

BD:	g	t	a	c	a	t	t	g	g	t	a	c	g	t	c	c
S:					a	t	t	g	a	t	c	c	t			

extension sans gap : 1 substitution

◂ blastn

Exemple d'alignement

BD:	g	t	a	c	a	t	t	g	g	t	a	c	g	t	c	c
S:					a	t	t	g	a	t	c	c	t			

extension sans gap : terminé

◂ blastn

Exemple d'alignement

BD:	g	t	a	c	a	t	t	g	g	t	a	c	g	t	c	c
S:					a	t	t	g	a	t	c	c	t			

extension avec gap (au plus 1 insertion)

◂ blastn

Exemple d'alignement

BD:	g	t	a	c	a	t	t	g	g	t	a	c	g	t	c	c
S:					a	t	t	g	a	t		c	c	t		

extension avec gap : 1 insertion

blastn

Exemple d'alignement

BD:	g	t	a	c	a	t	t	g	g	t	a	c	g	t	c	c
S:					a	t	t	g	a	t		c	c	t		

extension avec gap : terminé

◂ blastn

Exemple d'alignement

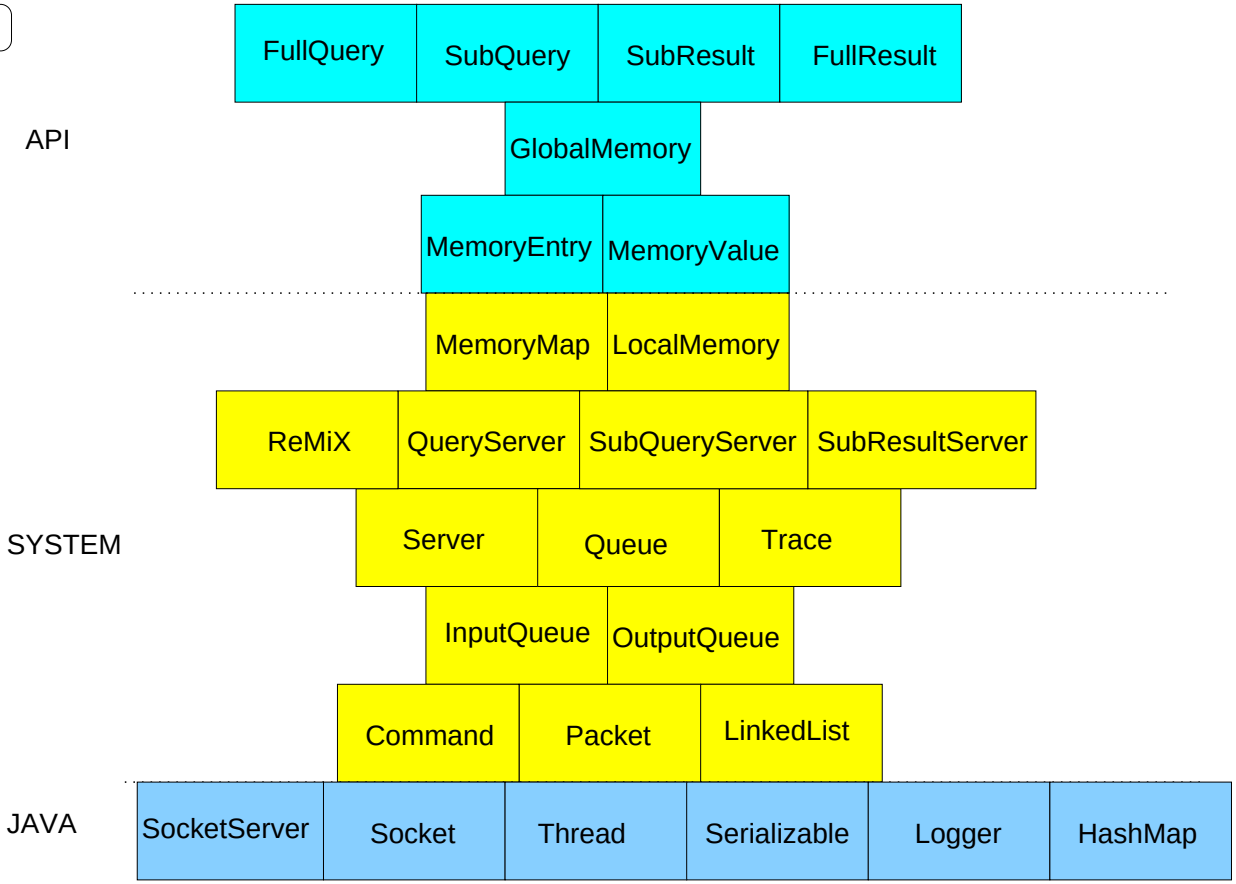
BD:	g	t	a	c	a	t	t	g	g	t	a	c	g	t	c	c
					|	|	|	|		|		|				
S:					a	t	t	g	a	t	-	c	c	t		

alignement : terminé

◂ blastn

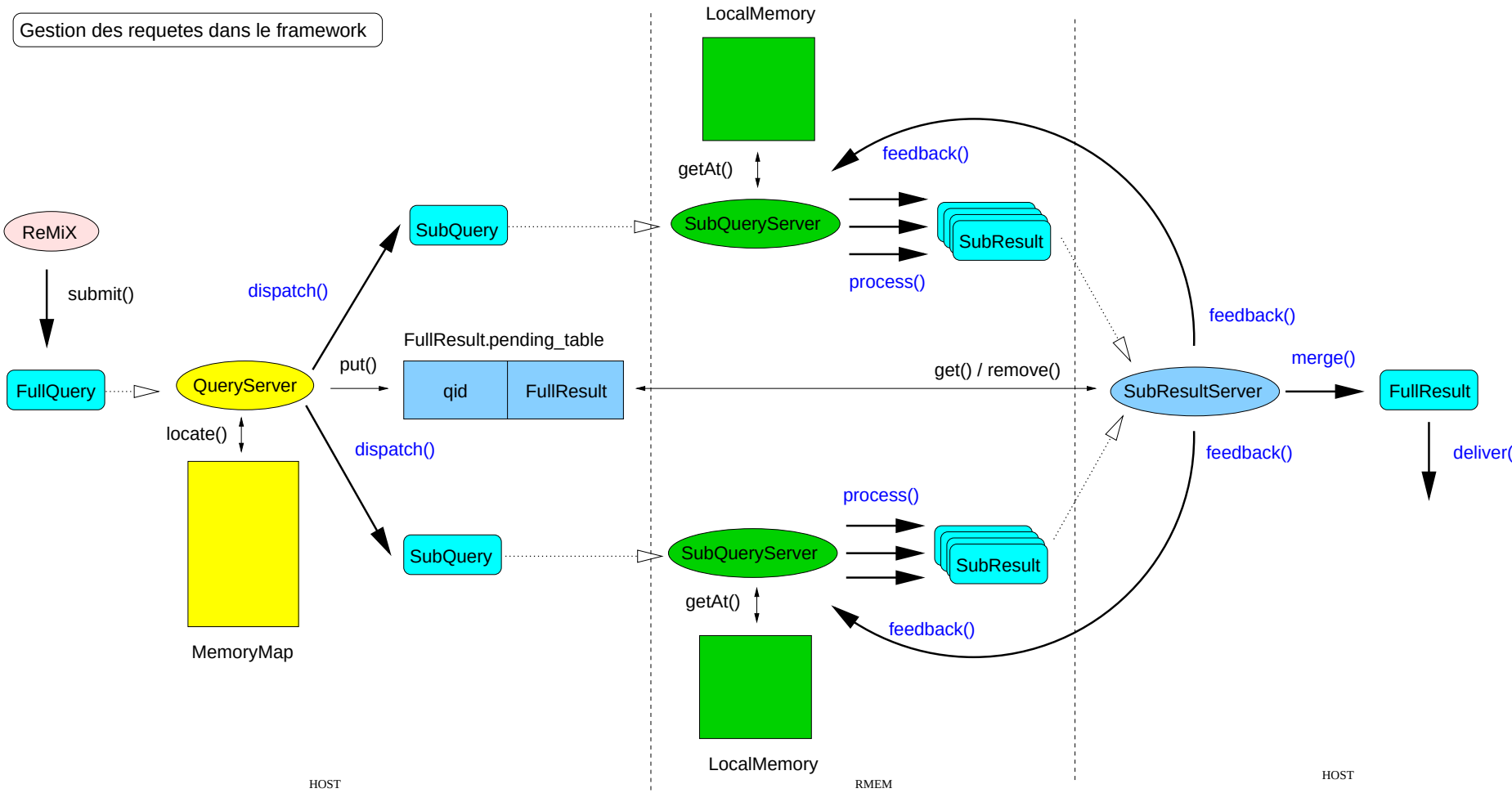
Package remix

Design du framework



framework

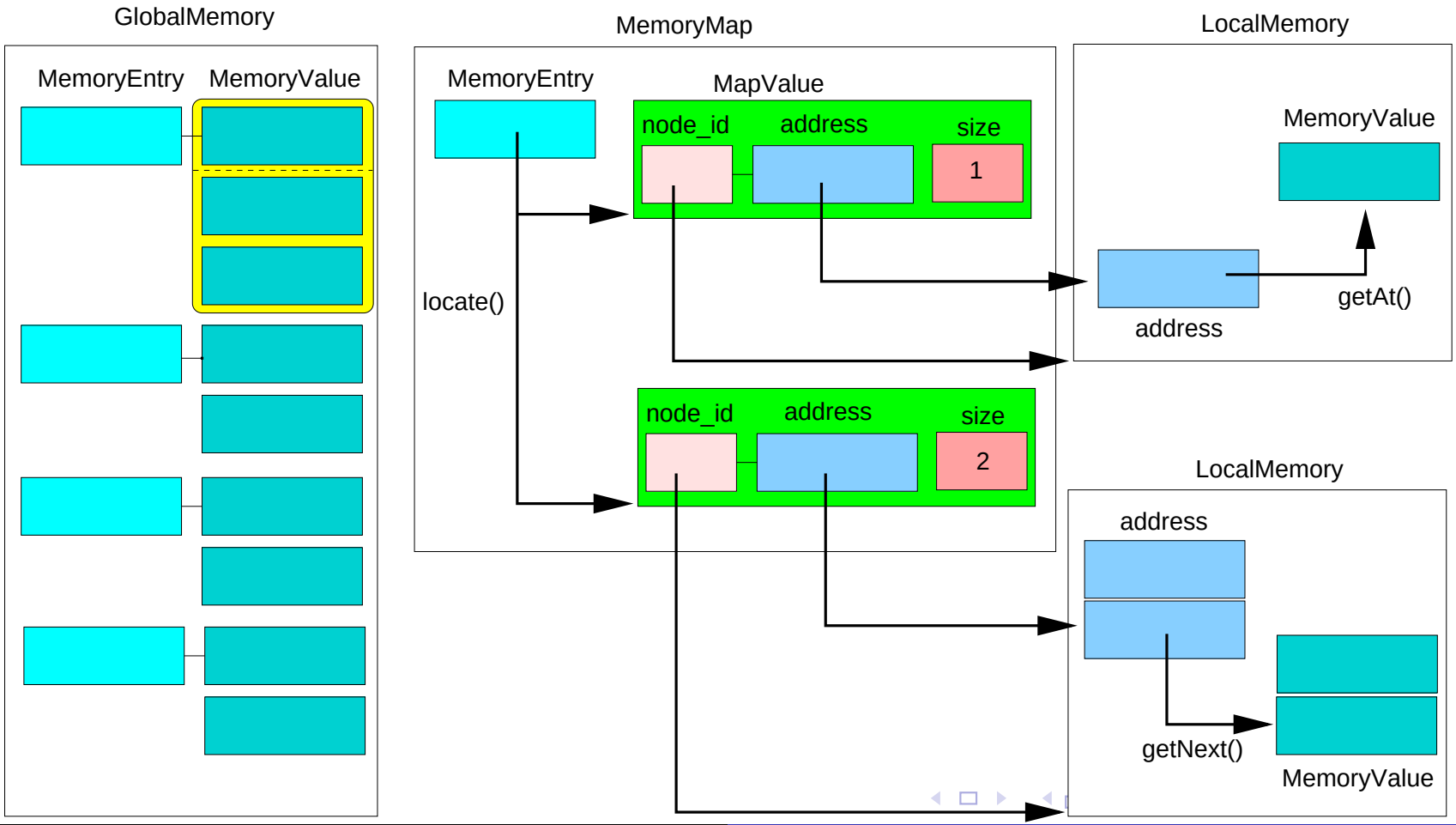
Package remix



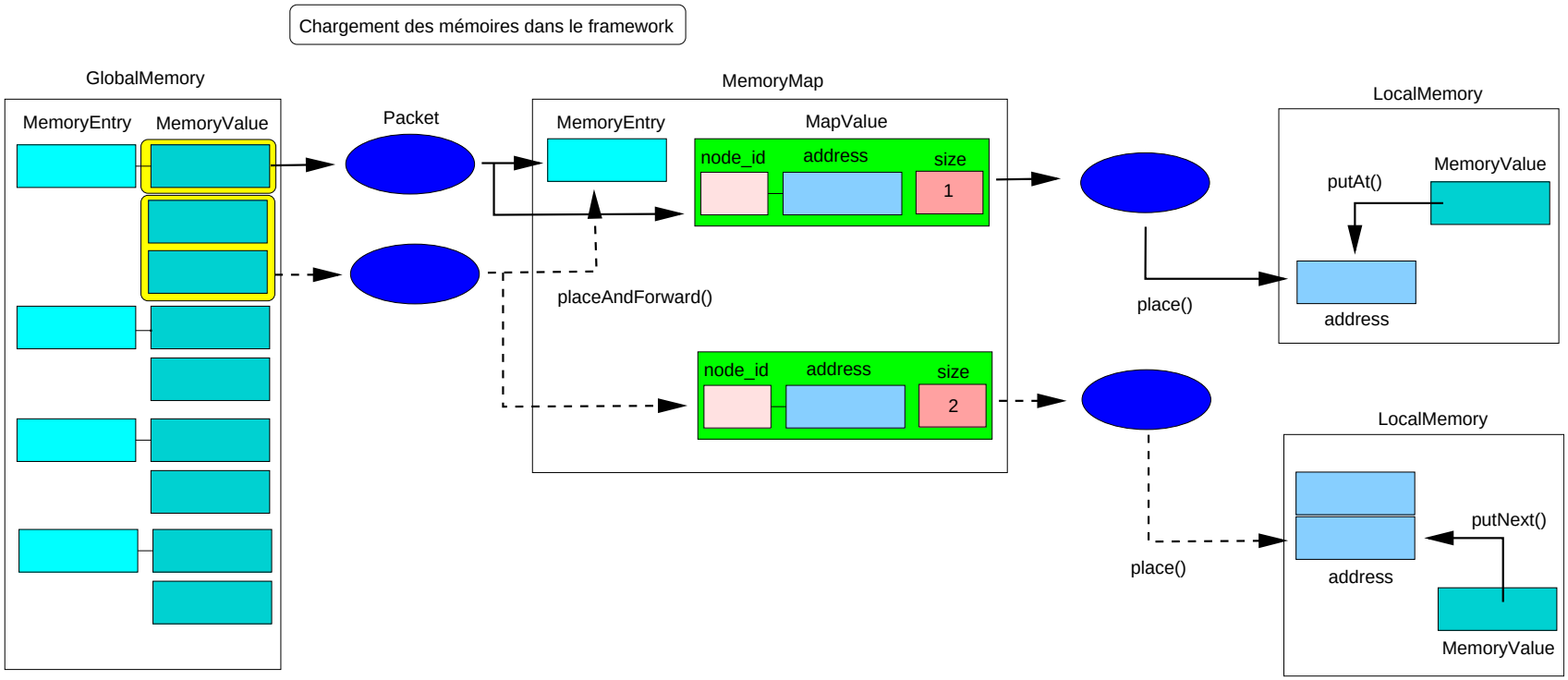
framework

Package remix

Organisation des mémoires du framework



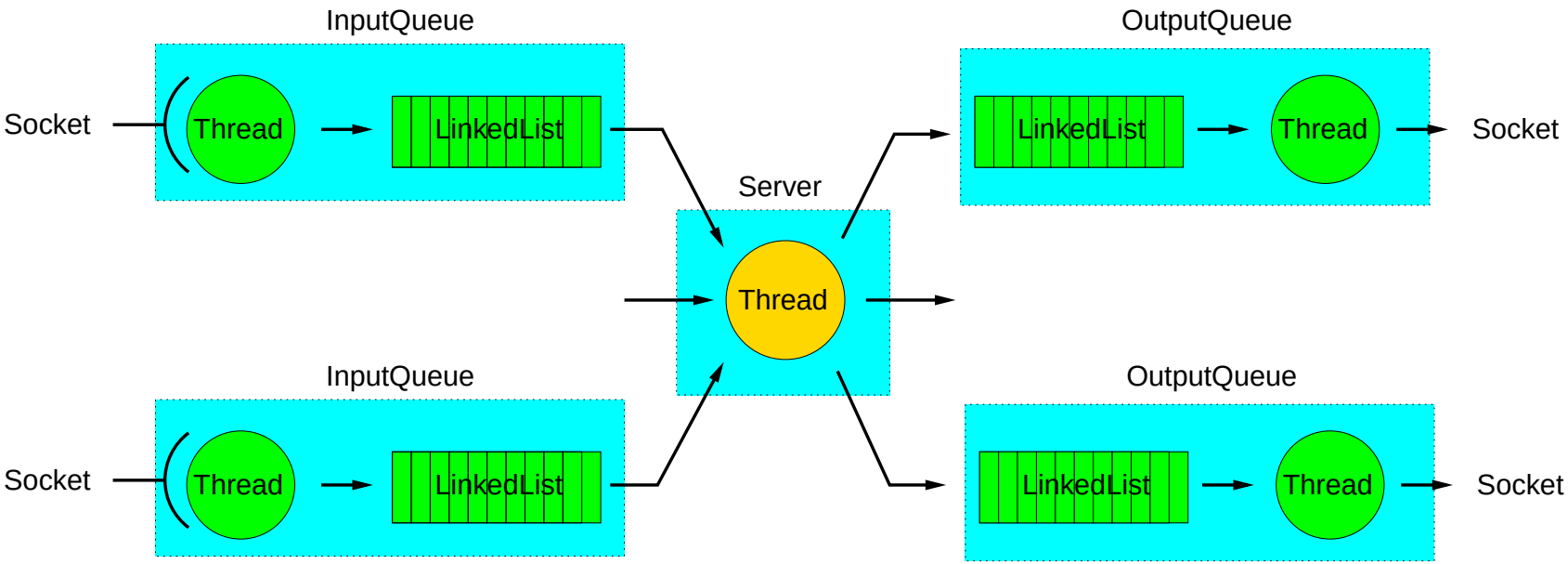
Package remix



framework

Package remix

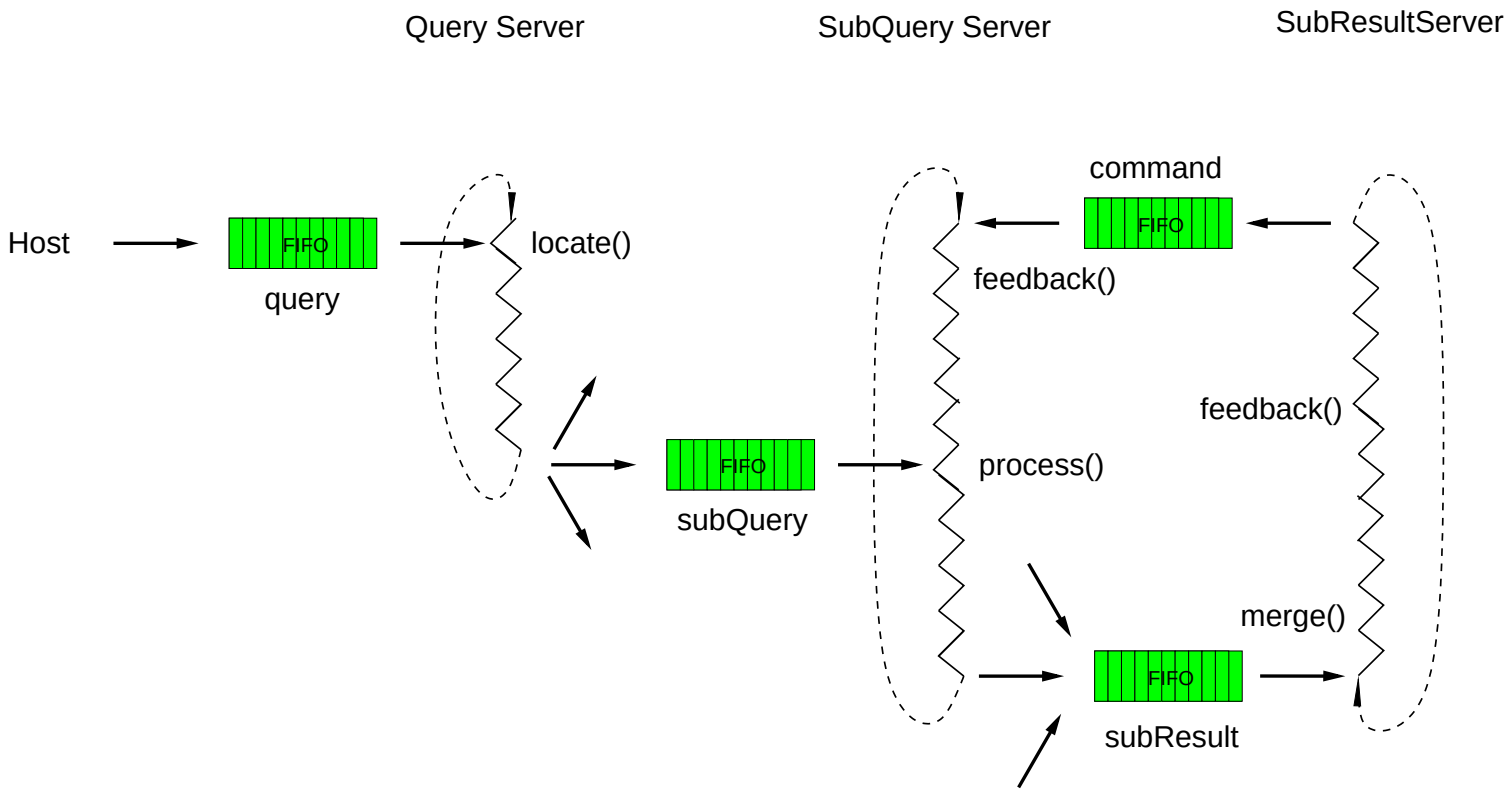
Mise en oeuvre d'un serveur



framework

Package remix

Boucles des serveurs



framework

La loi d'Amdahl appliquée à BLASTN

- G.M. Amdahl, *Validity of the single-processor approach to achieving large scale computing capabilities*, 1967.
- partie séquentielle $= s$
- partie parallélisable $= p$ $\Rightarrow$ accélération $= \frac{T_s + T_p}{T_s + \frac{T_p}{N}}$
- nombre de processeurs$= N$
- $(N = \infty, T_s + T_p = 1)$ $\Rightarrow$ $\boxed{\text{accélération maximale} = \frac{1}{T_s}}$
- pour (NCBI) BLASTN
  - si parallélisation de la phase ❶ seule : accélération max. $= 6,6$
  - si parallélisation des phases ❶ et ❷ : accélération max. $= 1000$

◂ BLASTN