Parallélisation d'un logiciel de simulation de croissance des plantes dans un environnement Java

Pascale Launay

VALORIA, Vannes

Pascale.Launay@univ-ubs.fr

Le projet Concerto

- Objectif :
  - définir une infrastructure répartie qui inclut la description globale d'une application et permet sa reconfiguration dynamique.
- Applications visées :
  - simulation numérique (croissance des plantes, mécanique des fluides
  - …
- Contexte :
  - parallélisme et distribution
  - aspects dynamiques
  - grappes, grilles

Le projet Concerto

- Besoins particuliers :
  - exploitation des réseaux haut débits
  - besoins d'adaptation
  - interconnexion
- Moyens :
  - description du parallélisme, aspects dynamiques
  - exécutif adapté aux réseaux haut-débit
  - mécanismes d'observation et d'adaptation
- Travaux préliminaires :
  - Do!
  - Expresso
  - Projet RASC

Action de validation applicative iHPerf'98

- « Parallélisation d'un logiciel de simulation de croissance des plantes dans un environnement Java »
- Objectif :
  - évaluer la faisabilité de l'exécution sur une grappe de PCs d'un code de simulation de la croissance des plantes, en utilisant notre plateforme
- Moyens :
  - environnement Do! : description du parallélisme ; accès transparent aux objets
  - exécutif Expresso : transfert de graphes d'objets

Plan de la présentation

- Présentation de l'application
- L'environnement Do!
- L'exécutif Expresso
- Points-clé du développement
- Perspectives

Simulation de croissance des plantes

- Logiciel *AMAPpara*, conçu par le CIRAD, Montpellier

- Modélisation de phénomènes de croissance des arbres de manière très fine :
  - étudier le développement d'un arbre ou d'une forêt en fonction de l'environnement.
- Applications :
  - étudier les meilleures conditions de plantation pour optimiser le rendement ou la qualité du bois
  - prévoir l'aspect visuel d'un arbre pour des applications de paysagisme.

Simulation de croissance des plantes

- Particularités :
  - complexité de la structure (arbre)
  - couplage : simulation de plusieurs phénomènes physiques différents (éclairement, contraintes mécaniques,...)
- Exécution :
  - temps de calcul importants
  - espace mémoire très grand

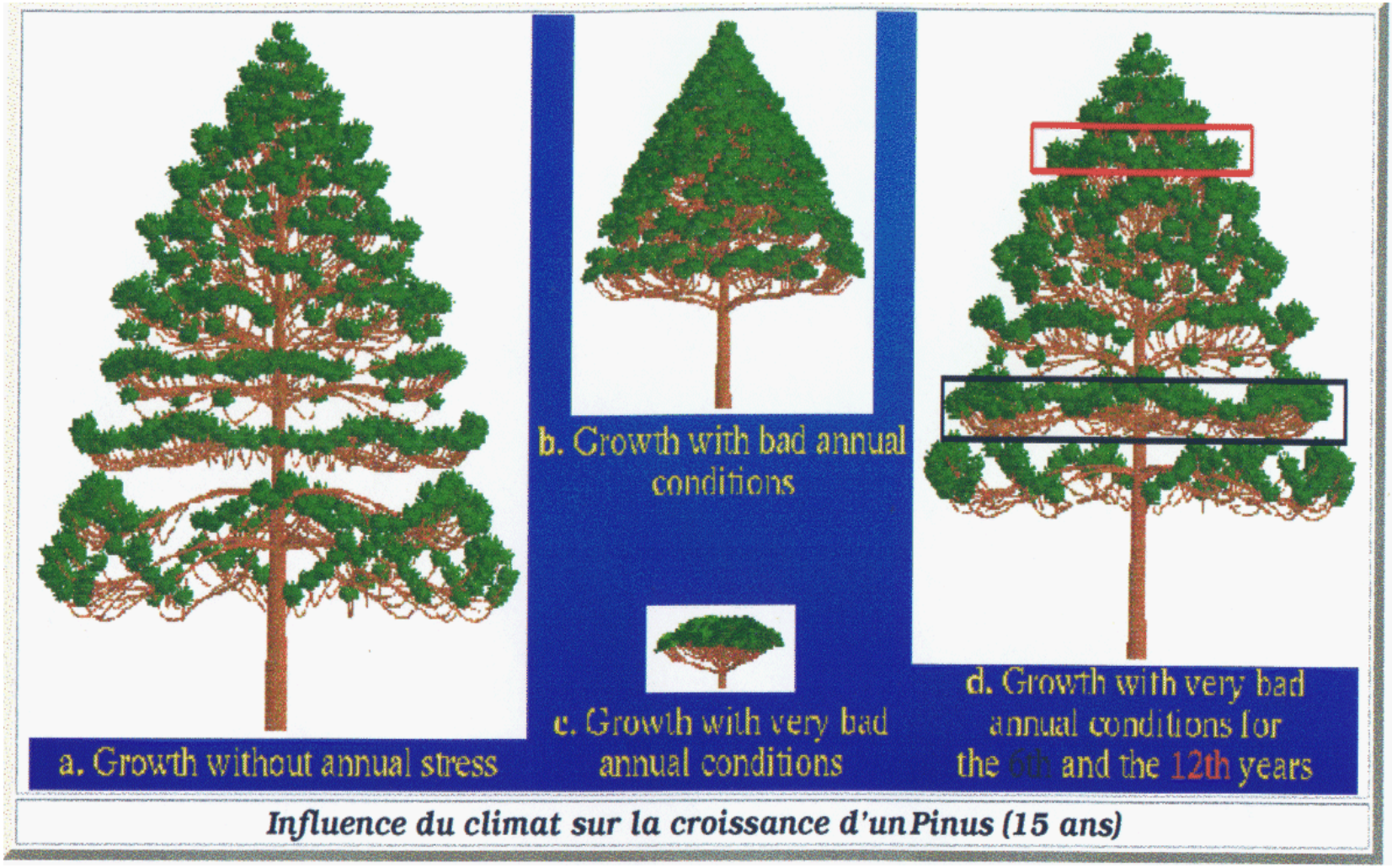
↳ Objectif : augmenter la taille des problèmes traités

Simulation de croissance des plantes

- moteur de croissance (physiologie)
- conditions du milieu
  - lumière, vent
  - hygrométrie, compétition,
- simulation mécanique

Recherche de la lumière

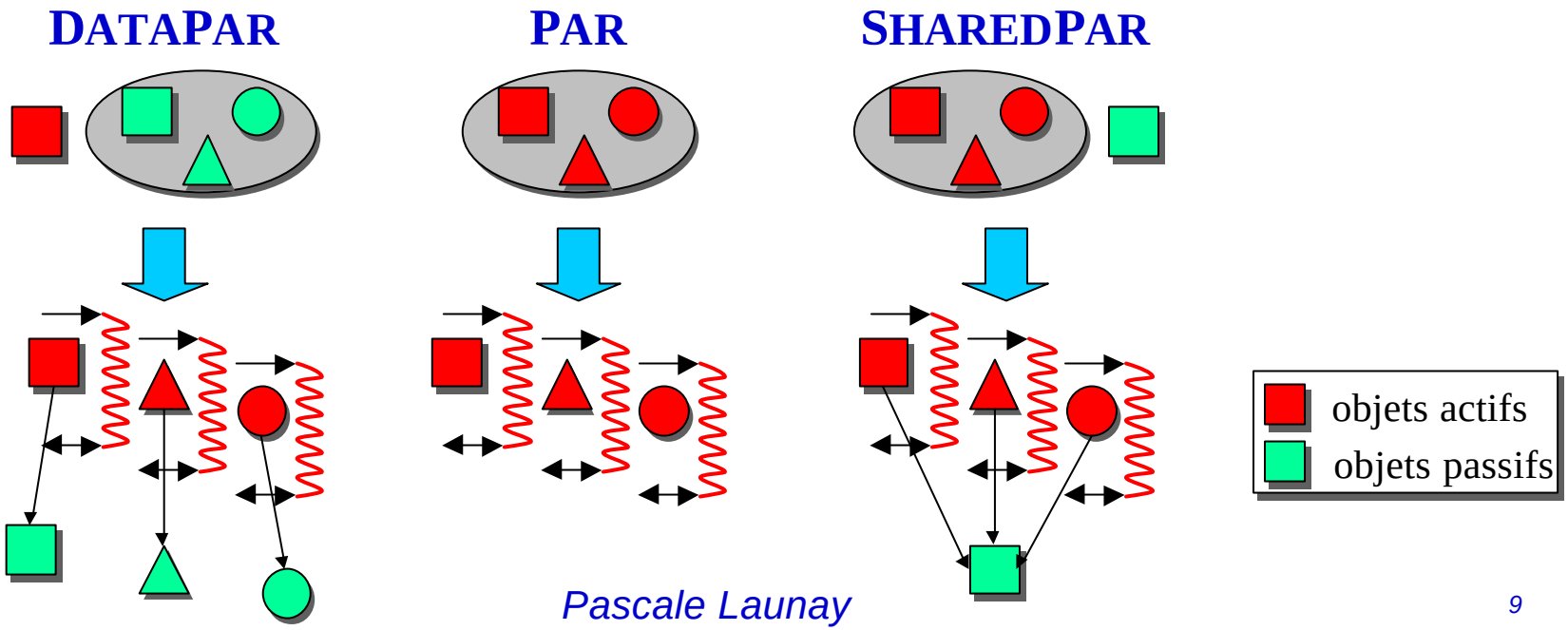
Exemple : croissance d'un Pinus



Influence du climat sur la croissance d'un Pinus (15 ans)

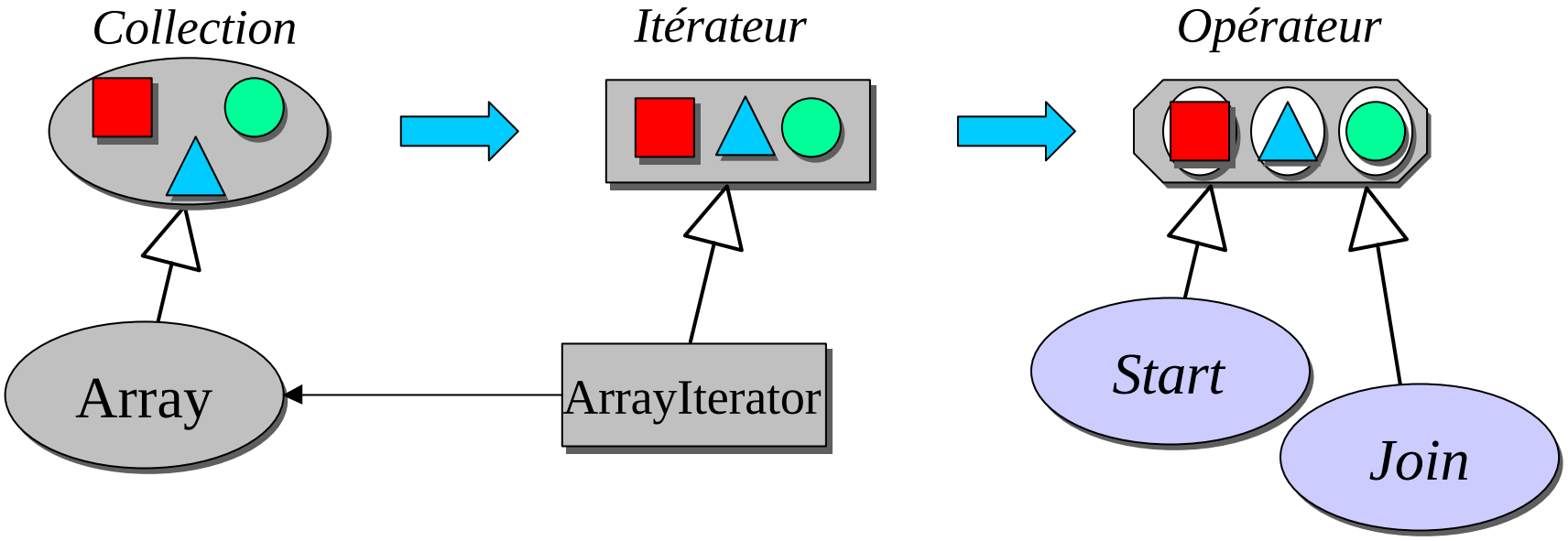
Description du parallélisme et de la distribution

- Cadre unifié pour l'expression du parallélisme et de la distribution par l'utilisation de collections
- Constructeurs parallèles appliqués à des collections d'objets actifs (tâches) ou passifs (données)



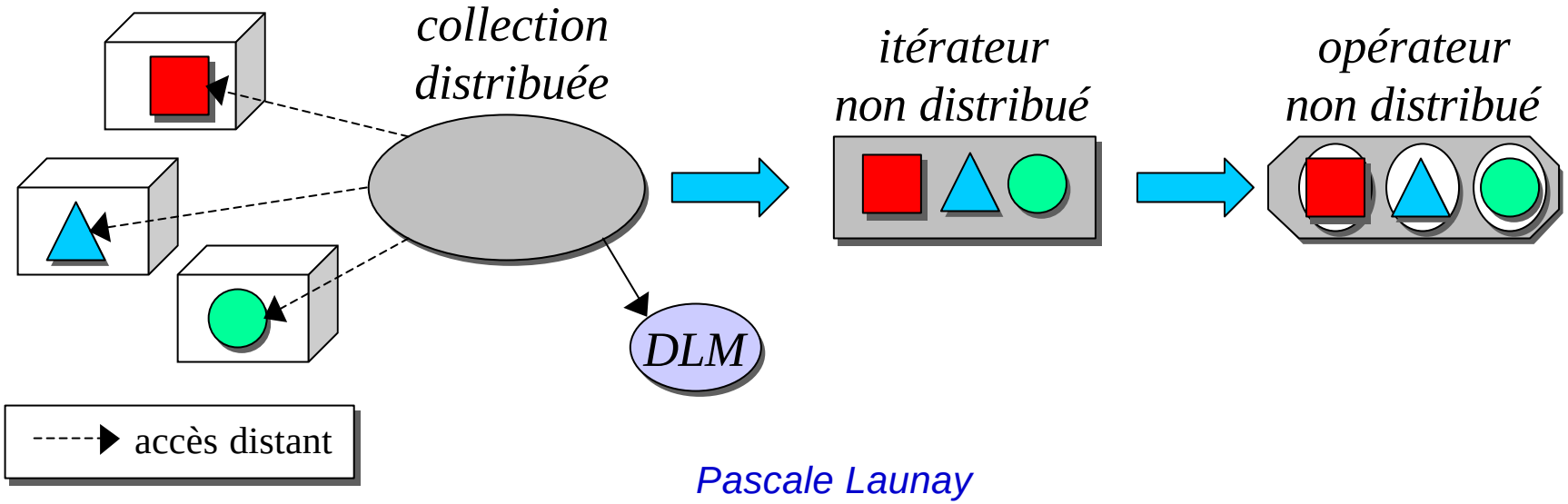
Description du parallélisme

- Parcours des collections : itérateurs *robustes*
- Traitement sur les collections : opérateurs *réactifs*

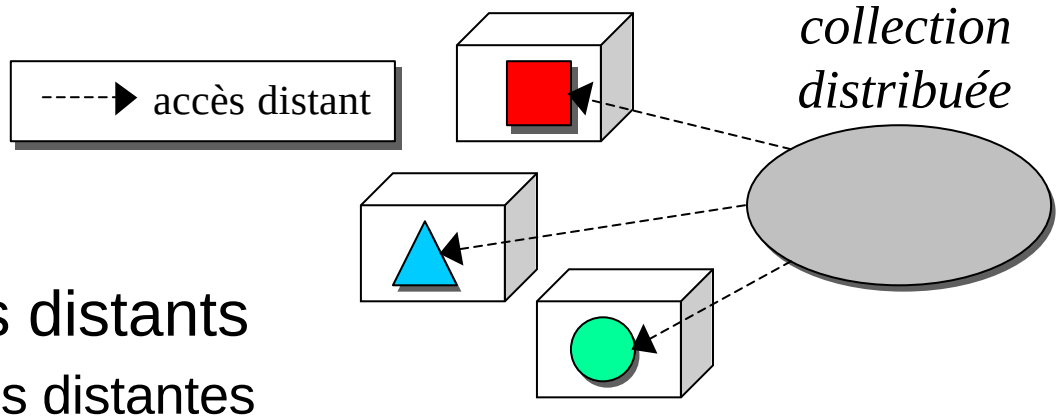


Description de la distribution

- Collection distribuée : regroupe des éléments placés sur des nœuds distincts
- Gestion de la distribution : DLM (*Distribution Layout Manager*)
  - transformation de clés
  - nœud propriétaire d'un élément



Accès transparent aux objets



- Accès aux éléments distants
  - placement : créations distantes
  - accès : RMI (invocations de méthodes à distance)

- Transformations de code (objets *accessibles*)
  - localisation transparente des objets (placement et accès uniformes)

Expresso

- Objectif : étude de l'impact des réseaux à haute performance sur les supports d'exécution pour systèmes à objets
  - mises en œuvre classiques mal adaptées aux réseaux à haute performance
- Prototype de support pour Java (Kaffe)
  - brique de base : transfert de graphe d'objet (avec partages et cycles)
  - réseau local de type Myrinet
  - mise en œuvre originale
    - performance de la sérialisation faible

Bibliothèque Expresso

- Caractéristiques :
  - regroupement d'objets (clusters d'objets)
  - contrôle de l'allocation mémoire par la JVM
  - transfert direct de zones mémoire (1 cluster = 1 message)
- Modèle :
  - création des objets dans des clusters
  - communication de clusters
  - utilisation transparente des objets émis et reçus
- Interface : package Java
  - clusters iso-adresse (pas d'empaquetage / dépaquetage)
  - clusters relogeables (plus souple)

Exemple : envoi - réception d'un arbre

```
Cluster.init(argv);
if (Cluster.myNode == NODE0) {
    ClusterISO clu = new ClusterISO(3);
    clu.setCurrent();
    Btree arbre = (Btree)Cluster.newObject(Btree.class);
    arbre.fg = (Btree)Cluster.newObject(Btree.class);
    arbre.fd = (Btree)Cluster.newObject(Btree.class);
    ...
    clu.setRootObject(arbre);
    clu.send(NODE1);
} else {
    ClusterISO clu = new ClusterISO(3);
    clu = ClusterISO.recv(NODE0,3);
    arbre Btree = (Btree)clu.getRootObject();
    arbre.affiche();
}
Cluster.quit();
```

Mise en oeuvre

- JVM Kaffe, JNI
- Communication sur GM-MPICH
- Transfert zéro-copie
- Surcoût d'empaquetage / dépaquetage faible, voire nul
- Amélioration drastique des performances / sérialisation

Nbre d'objets	Serializable	Externalizable	ClusterRELOC	ClusterISO
29	25	3	0,30	0,28
355	145	25	1,07	0,81
1760	1384	210	4,30	2,82
3479		683	8,75	5,77

Temps de transfert d'un graphe (carte routière) en ms

Portage de l'application

- Réécriture de l 'application en Java
  - code
  - structures de données
- Découpage et parallélisation
- Adaptation à l 'environnement Do!

  ↳ Distribution

Adaptation de Do! à Expresso

- Support de communication actuel : RMI Java
  - ↳ particulièrement inefficace
- Objets *accessibles* : le schéma de transformation ne dépend pas du support exécutif

↳ Modifier le préprocesseur pour générer du code pour Expresso
  - invocation de méthodes : **`send/recv`**
  - création des objets : Expresso doit pouvoir en contrôler l'allocation.

Adaptation d' Expresso à Do!

- Extension de la bibliothèque Expresso :
  - services de communication de base
  - appel de méthode à distance
  - création d'objets à distance
  - prise en compte des *threads*

Perspectives

- Action iHPerf :
  - évaluer la faisabilité de l'exécution sur une grappe de PCs d'un code de simulation
  - finaliser et valider l'environnement Do! et l'exécutif Expresso en réalisant leur interconnexion

- Construction d'une plateforme répartie plus générale
  - support exécutif adapté aux réseaux haut-débit → grappes et grilles
  - encapsulation du parallélisme et de la distribution au sein de composants parallèles
  - mécanismes d'expression des besoins, d'observation et d 'adaptation