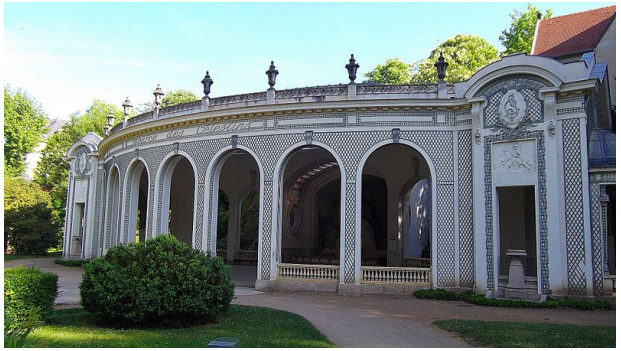
Une dérivation du paradigme de réécriture de multiensembles pour l'architecture de processeur graphique GPU



Gabriel Antoine Louis Paillard

Ce travail a eu le soutien de la **CAPES,** agence brésilienne pour la recherche


VENEZUELA
GUYANA
SURINAME
FRENCH GUIANA
PANAMA
COLOMBIA
ECUADOR
PERU
BRAZIL
BOLIVIA
PARAGUAY
CHILE
South Pacific Ocean
ARGENTINA


ORDEM E PROGRESSO

Sommaire

- Travaux de recherches
- Le formalisme *Gamma*
  - Métaphore de la Réaction Chimique
  - Quelques aspects concernant Gamma
  - Exemples de Programmes Gamma
- Architecture GPU
- Dérivation de Gamma

Travaux de Recherche

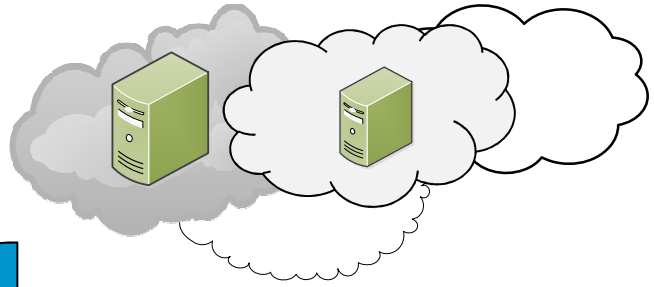
- Dévellopement d´un nouveau algorithme distribué pour le calcul des nombres premiers;

- Emploi du nuage pour le calcul à haute performance;

- Emploi du nuage pour les VLE (Virtual Learning Environment).

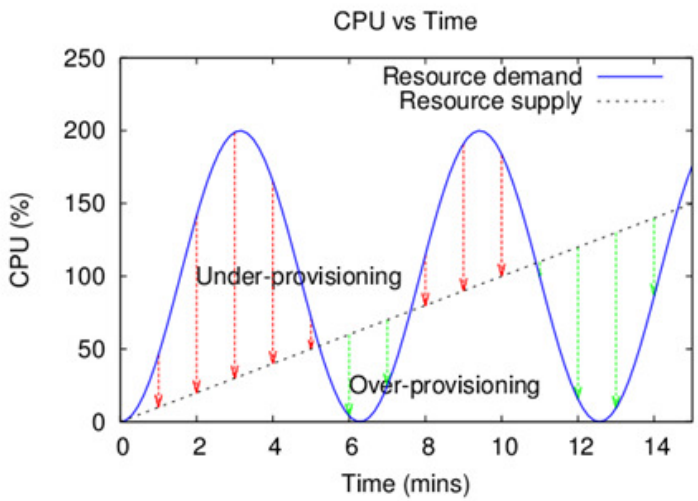
EATIS2014 VALPARAISO

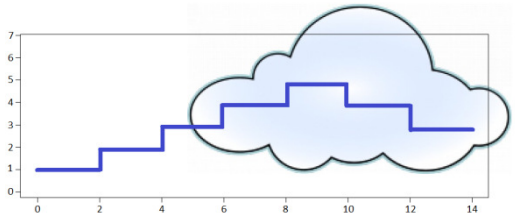
Emploi du nuage pour les VLEs

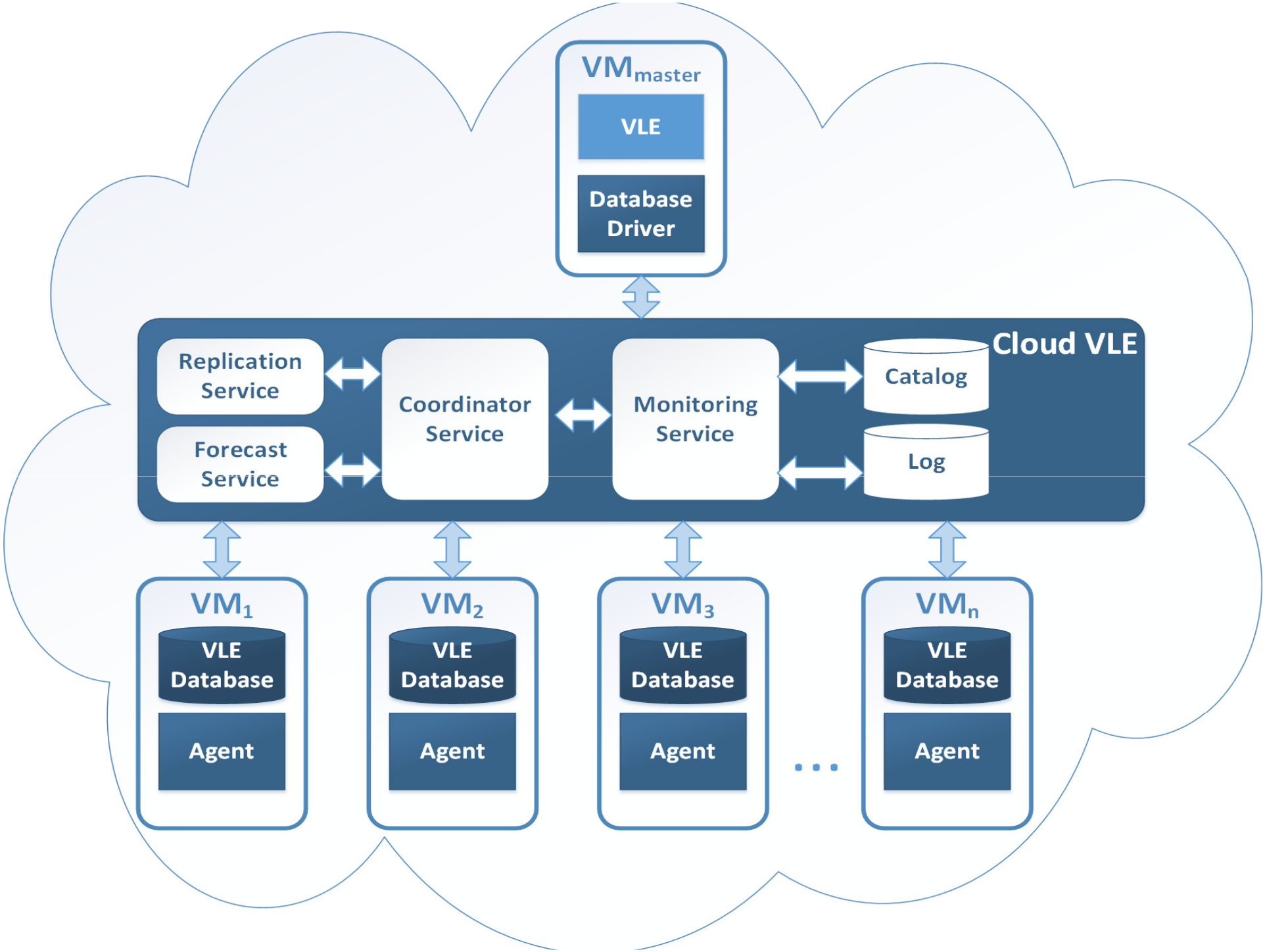


VLE

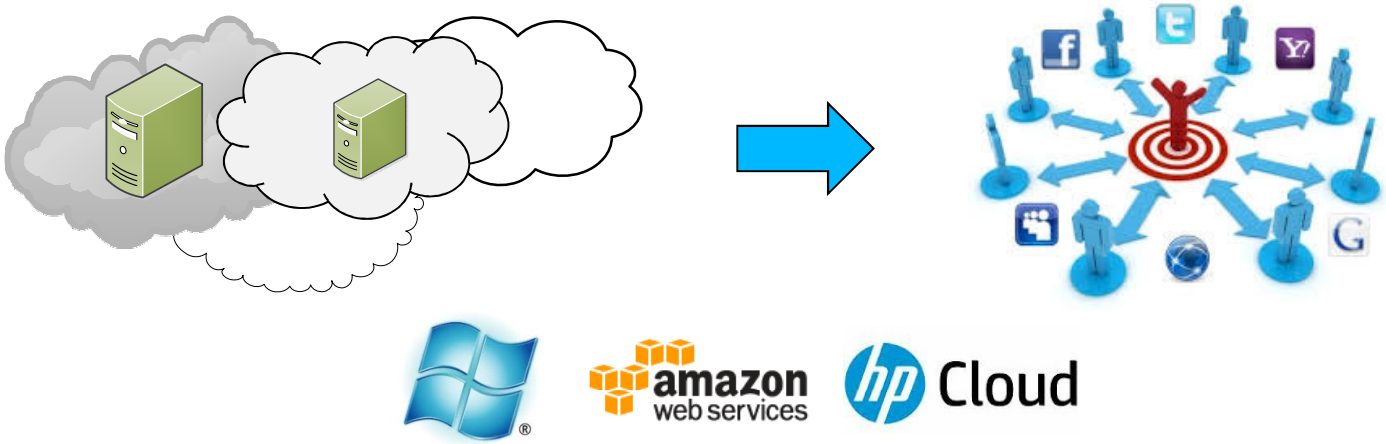
QoS





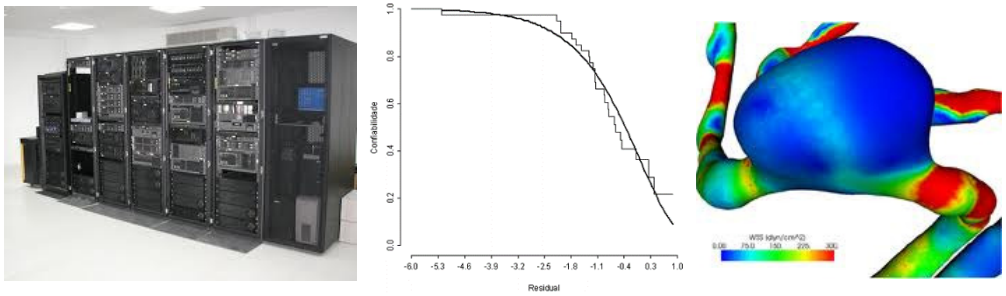

VM master
VLE
Database Driver
Cloud VLE
Replication Service
Forecast Service
Coordinator Service
Monitoring Service
Catalog
Log
VM1
VM2
VM3
VMn
VLE Database
Agent
VLE Database
Agent
VLE Database
Agent
VLE Database
Agent

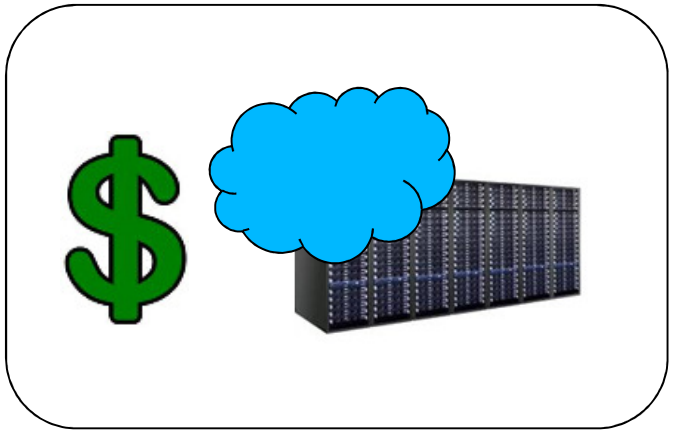
Emploi du nuage pour le HPC



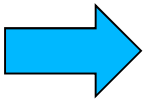
Temp d´éxécution des programmes

HPC





Emploi du nuage pour le HPC

HPC  CC

Analyse de performance

Objective:

Démontrer la possibilité d´éxécution des applications HPC em CC

Générer des données pour voir l´éxécution de benchmarks sûr plusieur dispositions

8th Euro American Association on Telematics and Information Systems

Distributed SystemsGrid Computing
Peer-to-peer computing
Cloud computing
Cooperative computing
Service computing
Green computing
Multi-agent systems

Gamma

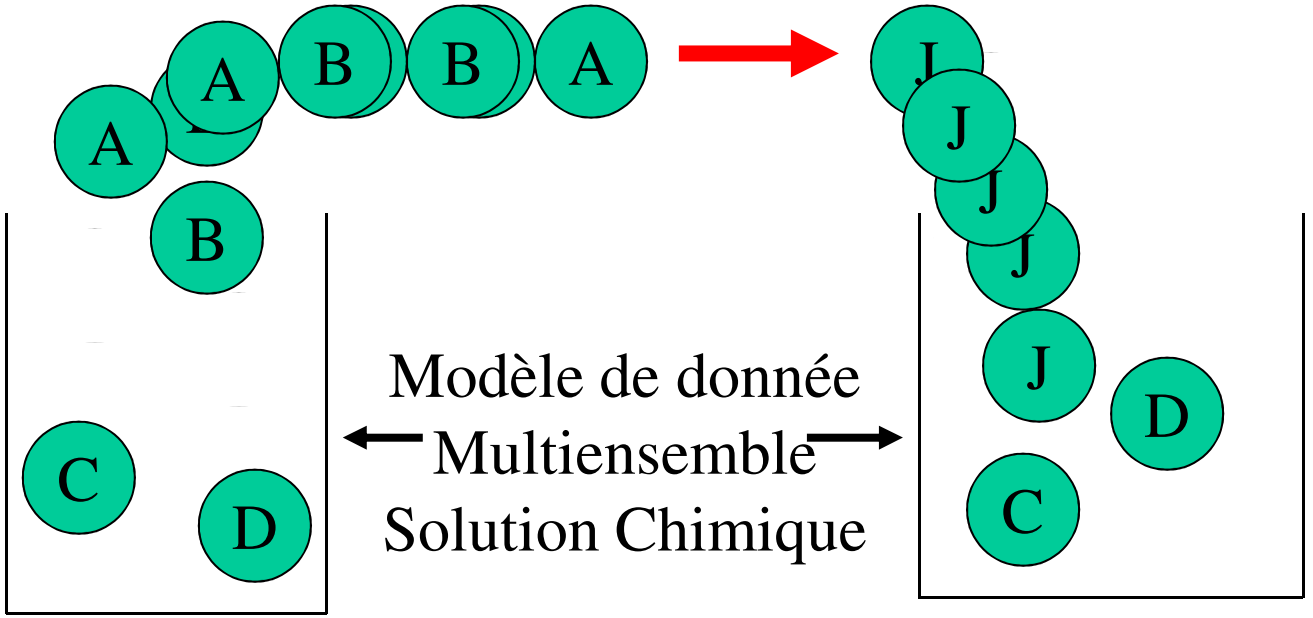
- *GAMMA* (**G**eneral **A**bstract **M**odel for **M**ultiset m**A**nipulation) fût créer em 1986 comme un formalisme pour la spécification de programmes;

- Façon d´abstraire seulement sûr le problème à traiter.

Gamma

MÉTAPHORE DE LA RÉACTION CHIMIQUE

Les élements sont remplacés par le résultat de l´action



Gamma

Le seul modèle de données en Gamma est le multiensemble qui peut être perçu comme une solution chimique où les données sont des réagents de cette solution.

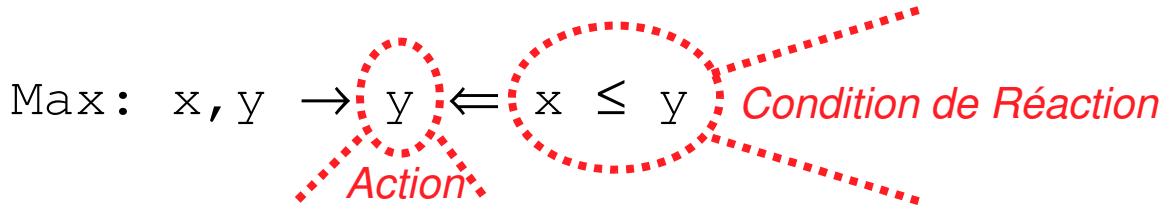
Un simple programme est une paire qui représente une action qui sera executé sûr les élements du multiensemble:

Action  **Condition de reaction**

Gamma

Exemple

- Programme Gamma qui calcule le plus grand élement



Un stade stable fût trouvé, c.a.d., aucune reaction peut avoir lieu. **{13}**

Aspects de Gamma

Libre interactions entre les éléments du multiensemble

Les programmes Gamma sont implicitement exécuté em parallèle

Si la condition de réaction est satisfaite par quelques sousensembles disjoints

Les reactions peuvent être operées simultanements et independaments

Aspects de Gamma

La possibilité de se débarasser de la séquencialité en Gamma rapporte deux conséquences:

• Permettre que le programmeur décrive les programmes de façon abstraite. Cela rend Gamma désirable comme langage intermédiaire dans le processus de dérivation de programmes;

• Comme Gamma n´emploie pas la séquencialité comme paradigme, le langage mène naturellement à des constructions de programmes parallèles.

Gamma

Definition formelle de Gamma:

$\Gamma((R_1,A_1),..,(R_m,A_m))(\mathrm{M})=$
if $\forall$ i $\in$ [1,m], $\forall$ $x_1,..x_n \in$ M, $\neg$R($x_1,..x_n$)
then M
else let $x_1,..x_n \in$ M, let i $\in$ [1,m] such that $R_i(x_1,..,x_n)$ in
$\Gamma((R_1,A_1),..,(R_m,A_m))((\mathrm{M}-\{(x_1,..,x_n\}) + \mathrm{A_i}(x_1,..,x_n))$

Gamma

Si la condition de réaction est satisfaite par quelques sousensembles disjoints, les réactions peuvent avoir lieu indépendamment et simultanément;

Cette propriété est la raison par laquelle les programmes Gamma possèdent un parallélisme potentiel.

Gamma

Ex.: prime_numbers(N) = Γ((R,A)) ({2,..,N}) where

R(x,y) = multiple(x,y)
A(x,y) = y

Comme conséquence du manque de compromis avec un ordre particulier d´éxécution, un programme Gamma peut être implémenté de façon parallèle.

Gamma

Le seul modèle de données en Gamma est le multiensemble.

Les éléments du multiensemble peuvent êtres des valeurs simples ou composés (même des multiensembles).

Um programme ne consiste plus en une séquence d´instructions modifiant un état, ou en une fonction apliquée à ses arguments mais en un transformateur de multiensembles opérant les données simultanément.

Gamma

<u>Un probléme numérique:</u>

fact(n) = Γ ((R,A)) {1,..n} where

R(x,y) = true
A(x,y) = x*y

Gamma

Problème de trie (sort):

Sort(Vetor) = Γ ((R,A)) Vetor where

R(i,v),(j,w) = (i>j) and (v<w)
A(i,v),(j,w) = {(i,w),(j,v)}

Gamma

Chemin plus court:

Shortest_path(G) = Γ ((R,A)) (G) where

$R(v_1,v_2,c_{12}),(v_2,v_3,c_{23})$, $(v_1,v_3,c_{13})= c_{13} > c_{12}+c_{23}$

A $(v_1,v_2,c_{12}),(v_2,v_3,c_{23})$, $(v_1,v_3,c_{13})=$

$\{(v_1,v_2,c_{12}),(v_2,v_3,c_{23})$, $(v_1,v_3,c_{12}+c_{23})= \}$

Gamma

Dîner des Philosophes:

philosophers= Γ $((R_1,A_1)(R_2,A_2))(\{F_0, F_1, F_2, F_3, F_4\})$ where

$R_1(F_i,F_{(i+1)mod5}) = True$
$A_1(F_i,F_{(i+1)mod5}) = \{P_i\}$
$R_2(P_i) = True$
$A_2(P_i) = \{F_i,F_{(i+1)mod5}\}$

Architecture GPU

Durant les années soixante-dix ont été développés et mis à la disposition les premiers processeurs accélérateur graphique à deux dimensions.

Dans les années 90, apparurent les premiers accélérateurs graphiques en trois dimensions qui avaient encore comme force motrice l'industrie des jeux videos,

Architecture GPU

Depuis le début des années 2000 il est possible de profiter de la puissance de traitement graphique des unités de traitement graphique grâce à la possibilité d'un modèle de programmation générale GPGPU (General-Purpose Graphics Processing Units) qui a permis l'accès à certaines unités de GPU et les employer pour d'autres fins que ses objectifs de programmation originales.

Architecture GPU

Depuis 2003, l'industrie des semi-conducteurs a mis en place deux chemins pour la conception de microprocesseurs:

- L'utilisation de plusieurs cœurs (multicore);
- L'utilisation de plusieurs noyaux (plusieurs noyaux).

Architecture GPU

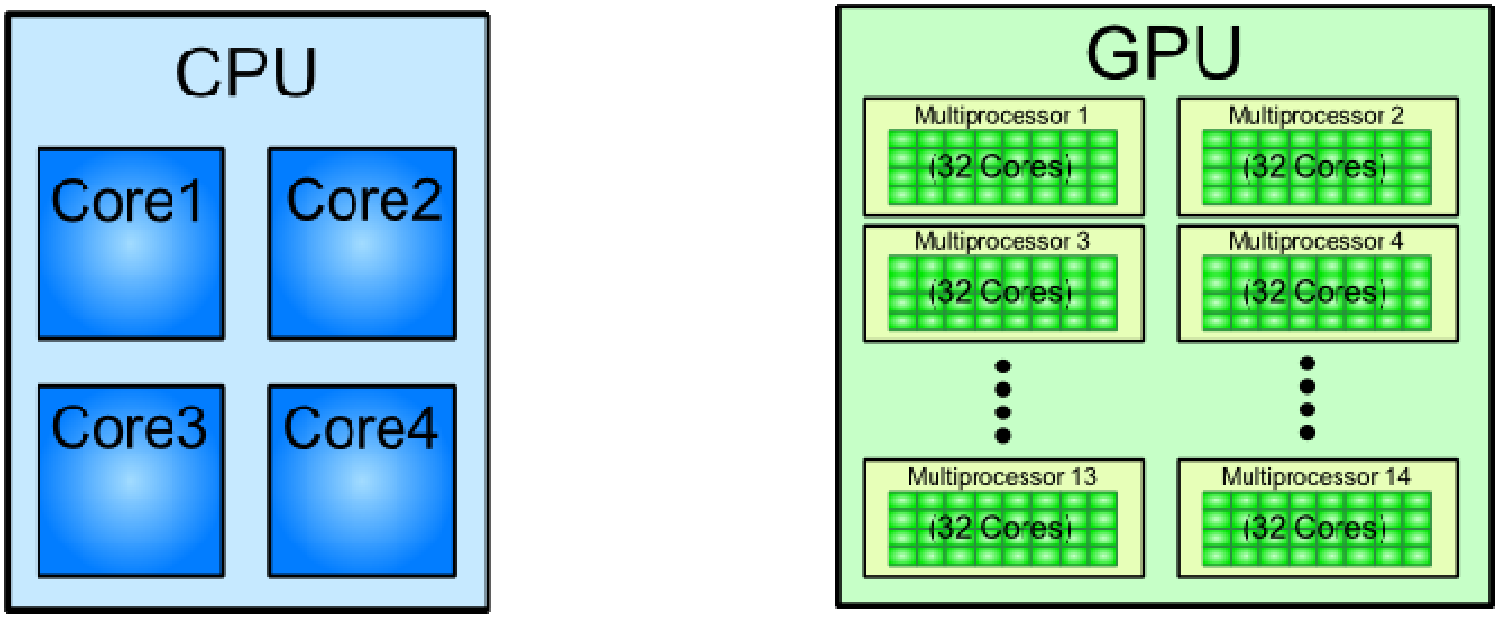
Architecture GPU

Nous pouvons visualiser le fonctionnement de ces processeurs comme un coprocesseur à l'unité centrale de calcul qui est idéal pour le traitement en parallèle d'un grand nombre de données qui peuvent être résolues d'une façon indépendante.

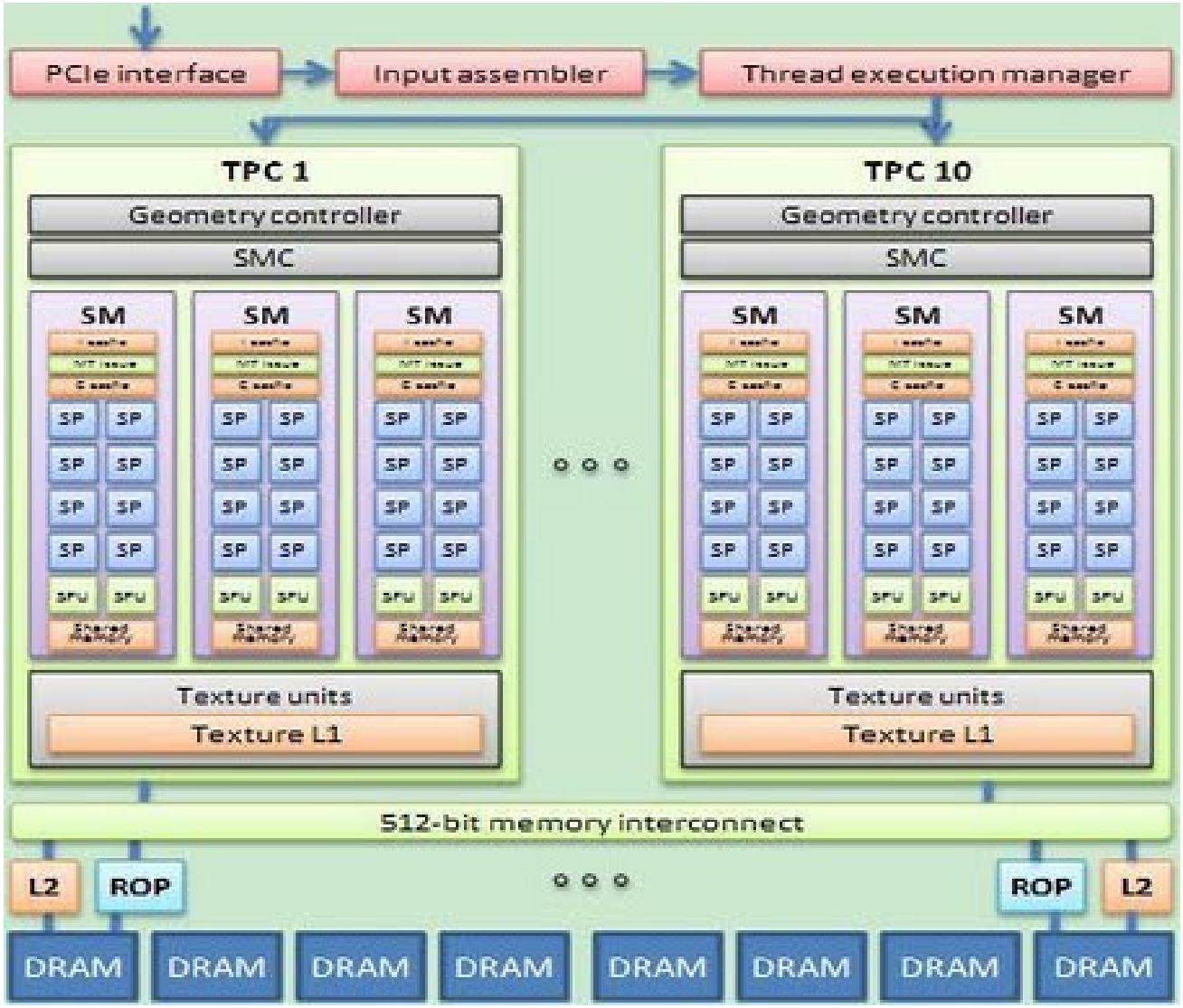
Il est à noter que les ordinateurs avec plus de puissance de calcul figurant dans le top 500 ont une architecture hybride, comprenant CPU's et GPU.

Architecture GPU

CPU/GPU Architecture Comparison



Architecture GPU



Dérivation de Gamma

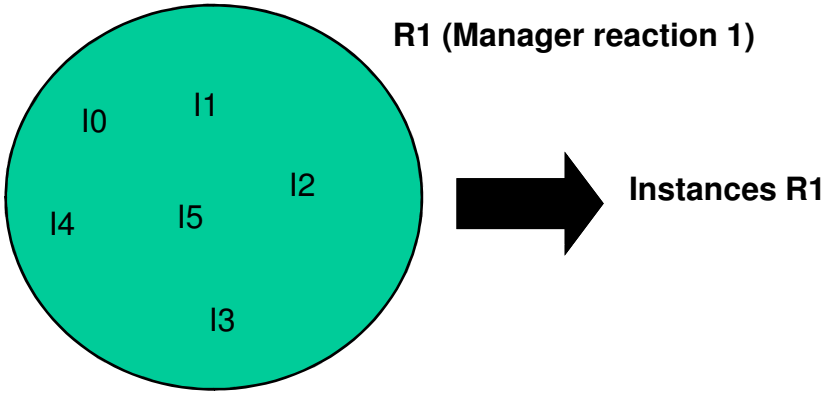
Motivations:

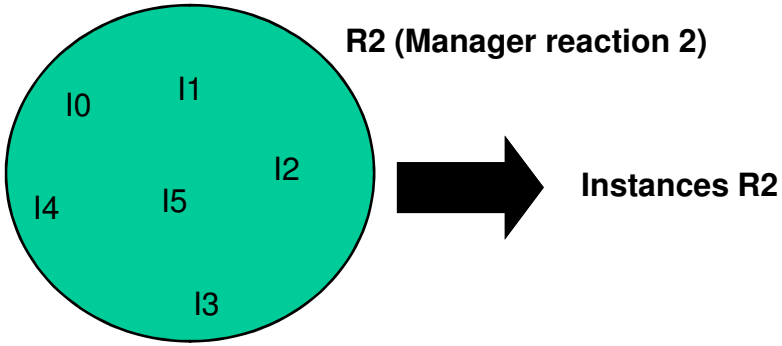
1. Dificulté de programmation de ses dispositifs;

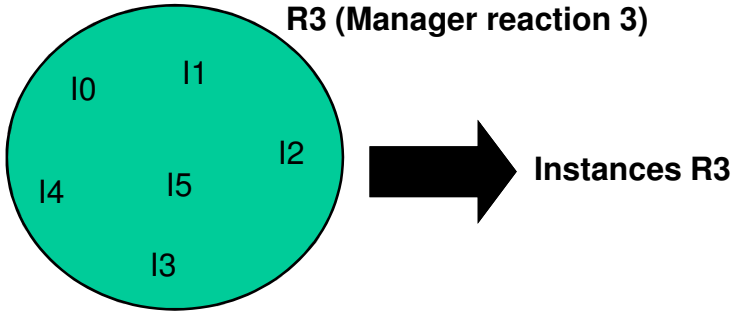
2. Caractéristique parallèle liès encore au paradigme impérative de programmation;

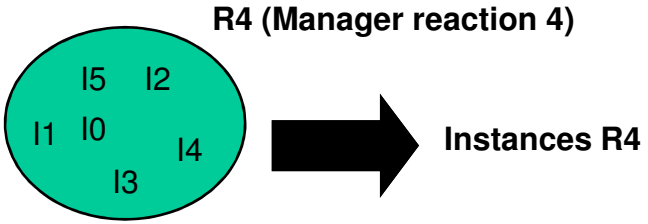
3. L'adéquation de l'architecture GPU pour le traitement des données est idéal pour l'application d'un langage qui suit le paradigme de réécritere de multiensembles.

Dérivation de Gamma









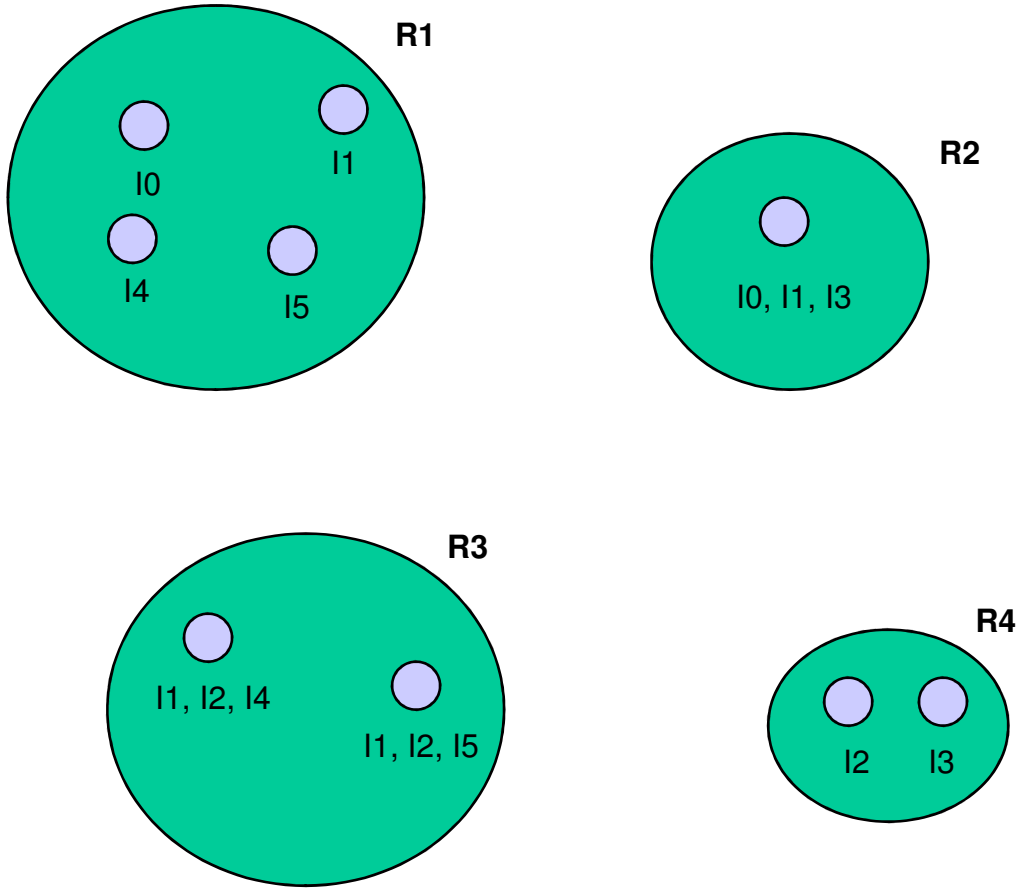
Dérivation de Gamma

R1 = i0 + i1 + i4 + i5;

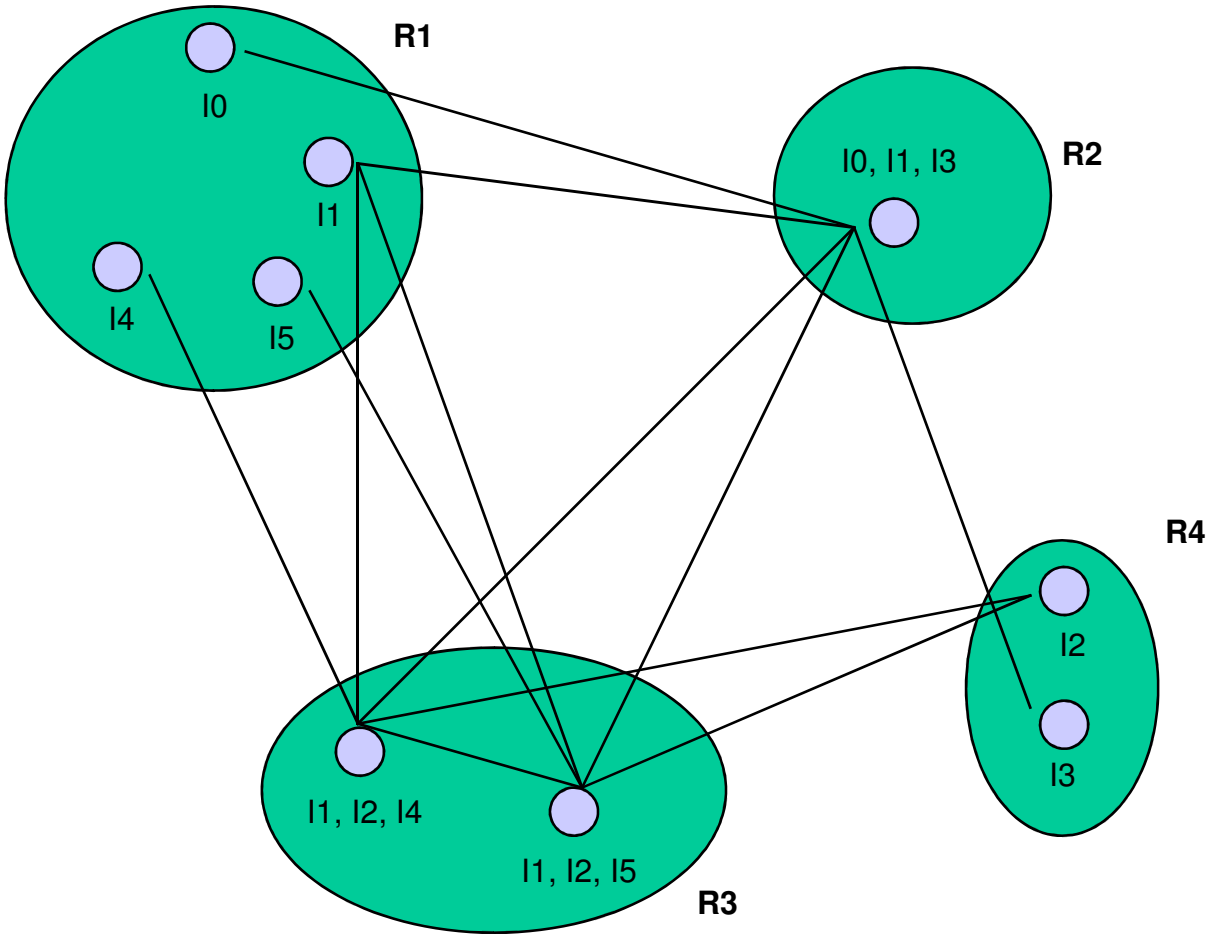
R2 = i0.i1.i3 + ... ;

R3 = i1.i2.i4 + i1.i2.i5 + ...;

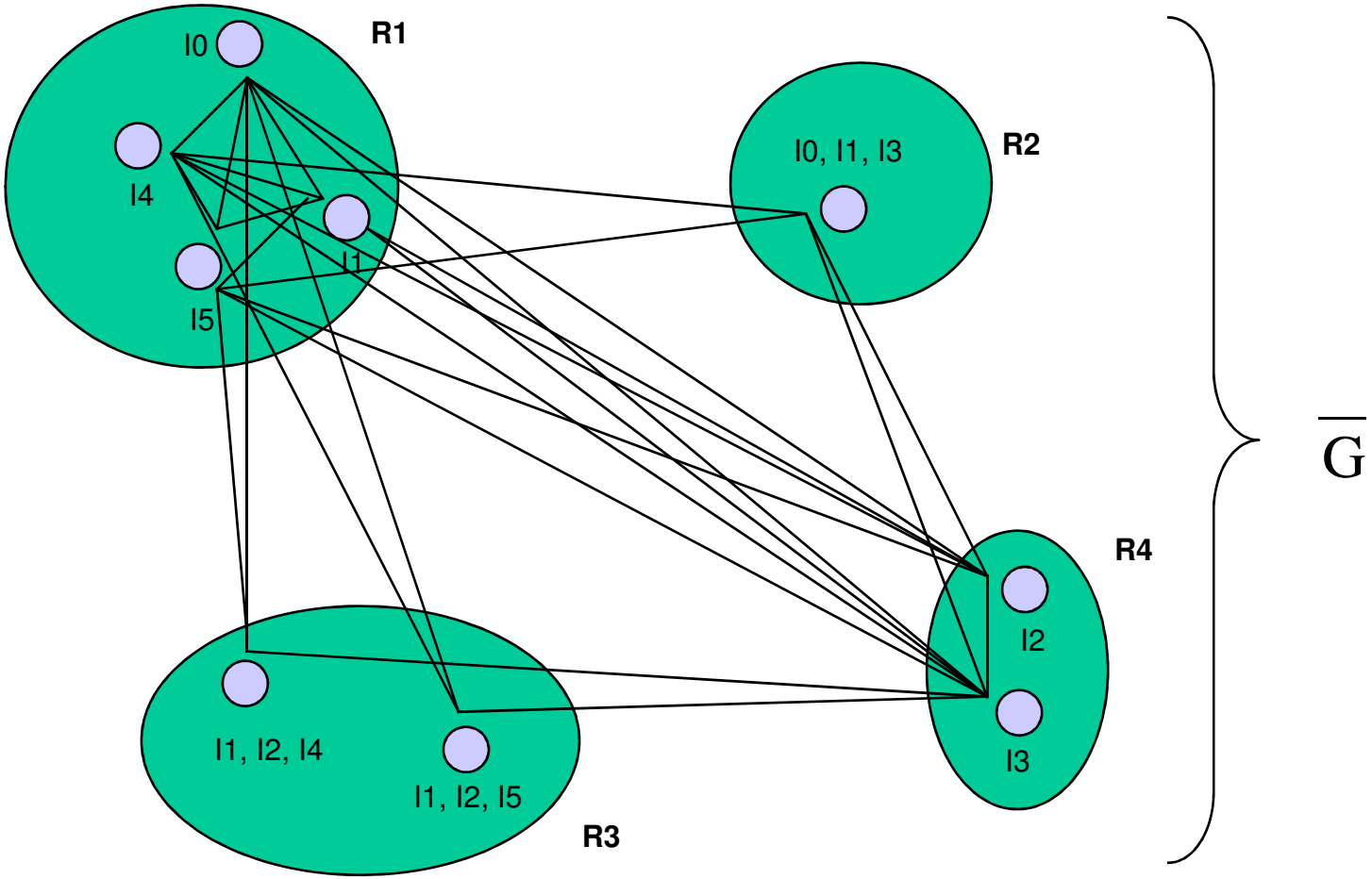
R4 = i2 + i3 ;



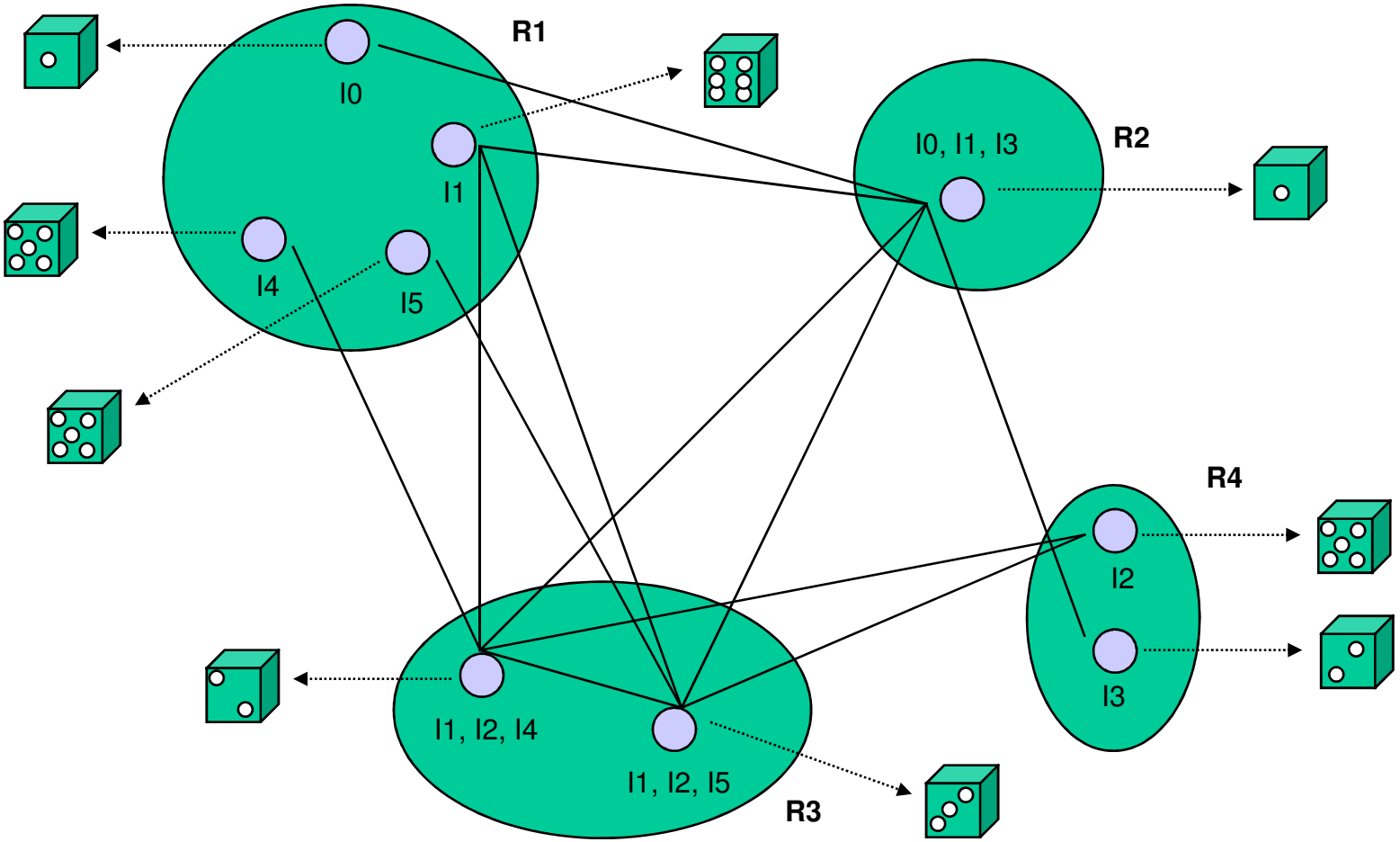
Dérivation de Gamma



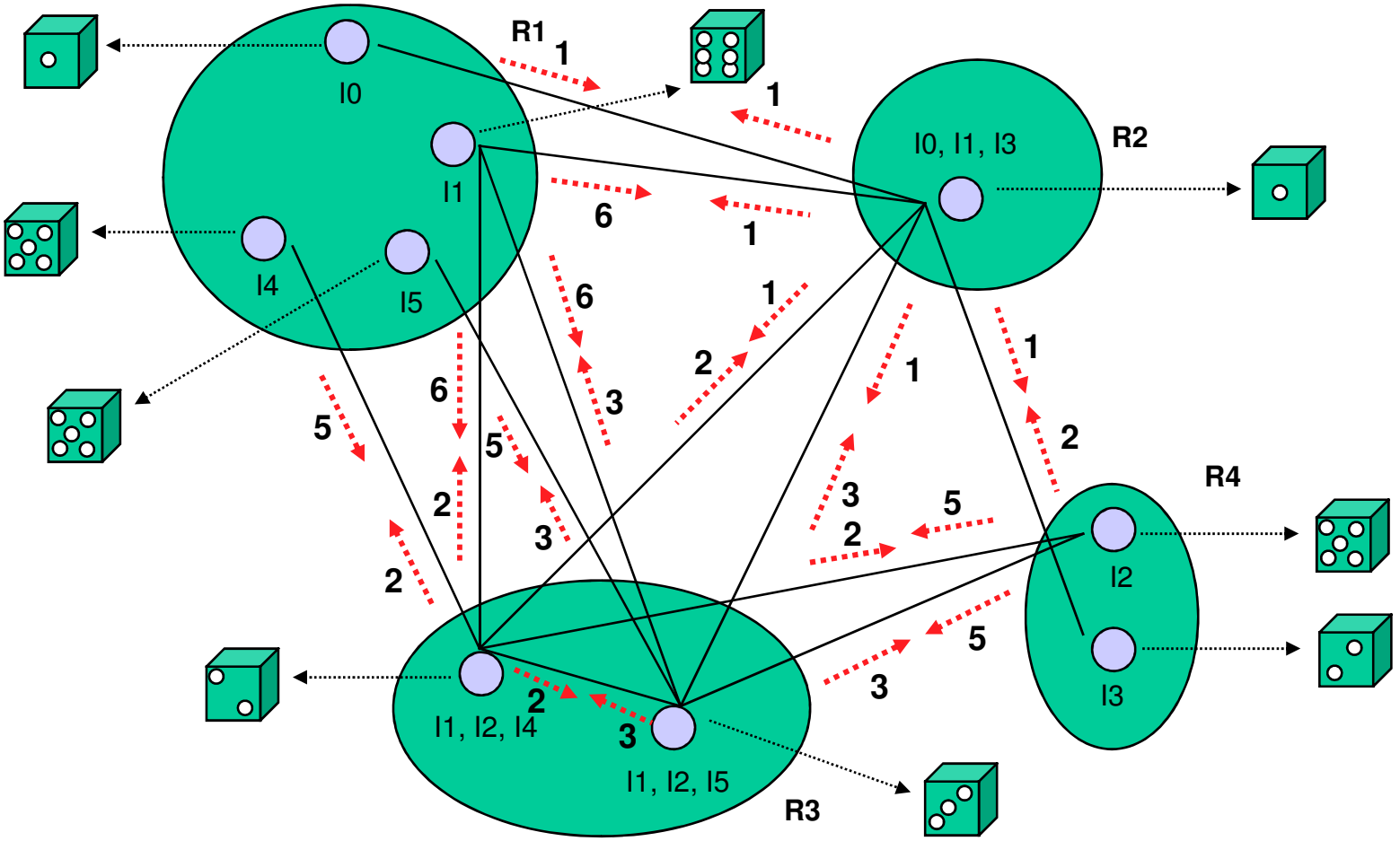
Dérivation de Gamma



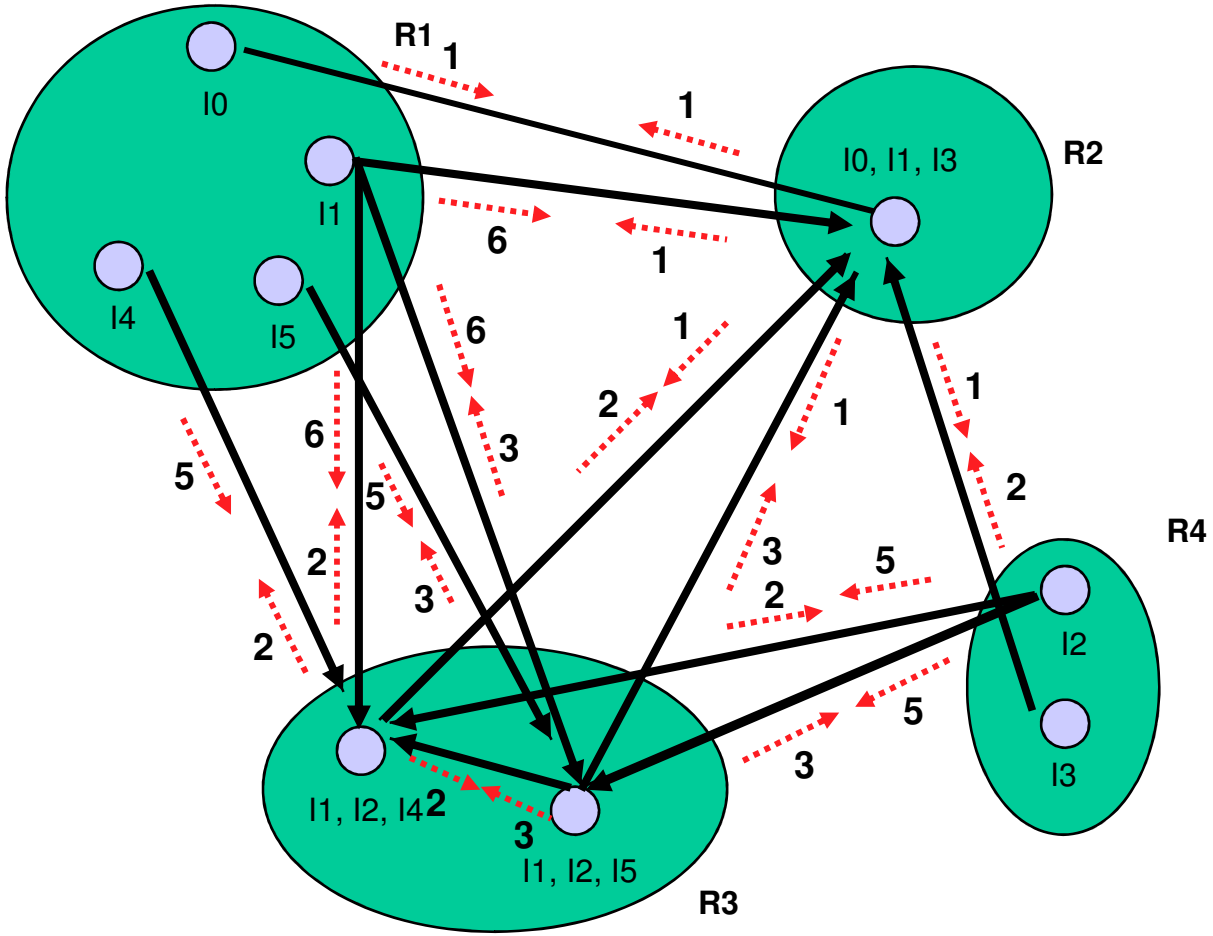
Dérivation de Gamma



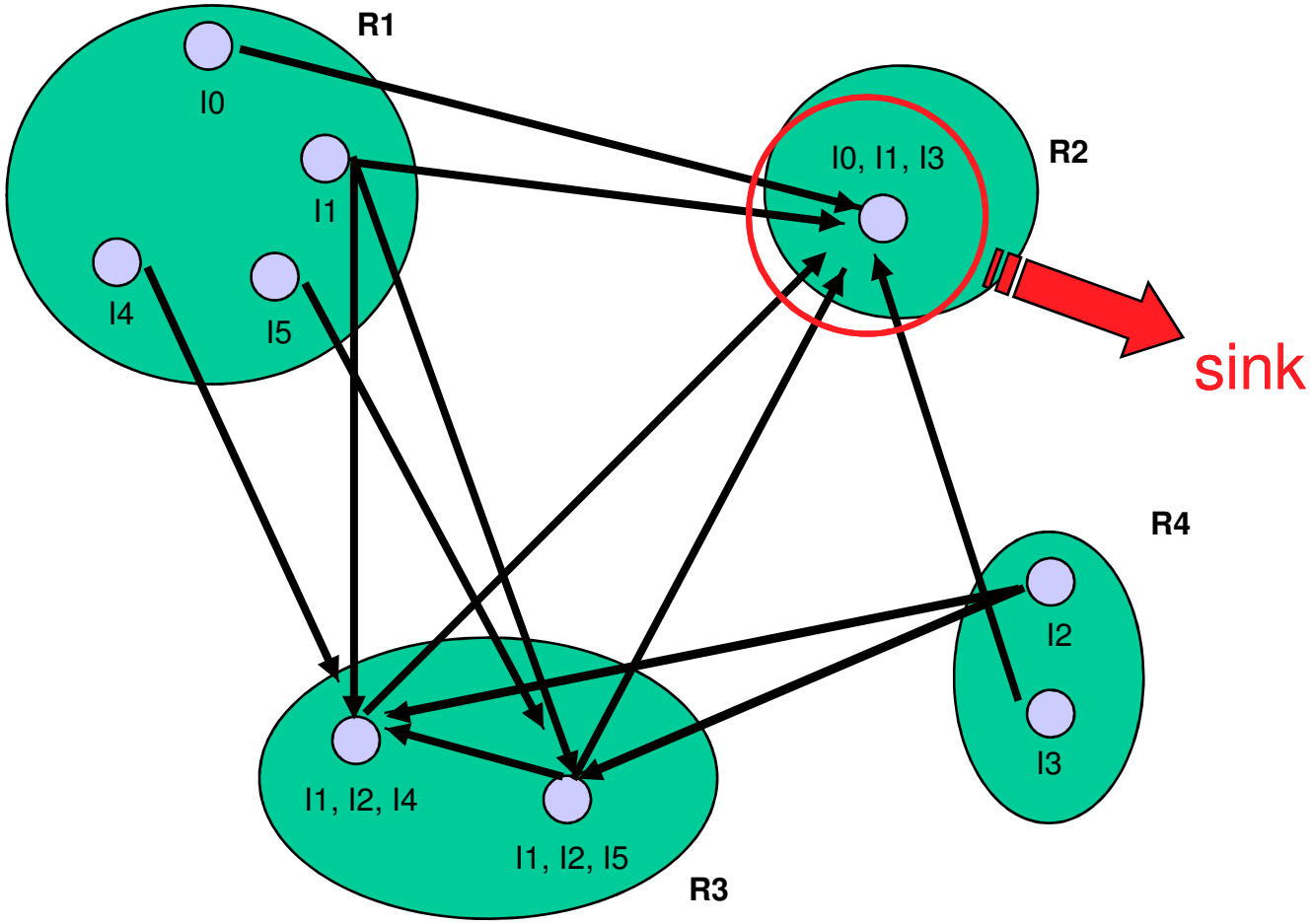
Dérivation de Gamma



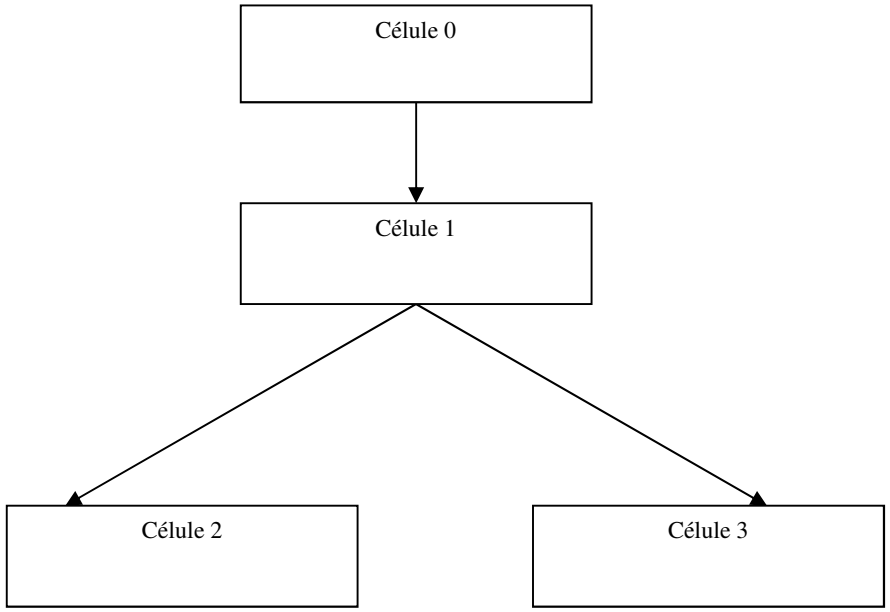
Dérivation de Gamma



Dérivation de Gamma



Dérivation de Gamma



La célule 0 est chargé de distribuer les multiensembles entre les unités de traitement où il y aura une célule, illustrée ici par la célule 1, qui sera responsable par la distribution des éléments du multiensemble aux *trheads*.

Conclusions et suite...

- Vérifier l´ordonnanceur pour les tâches Gamma;
- Implementer le langage sûr l´architecture CUDA;
- Gamma Structurée?