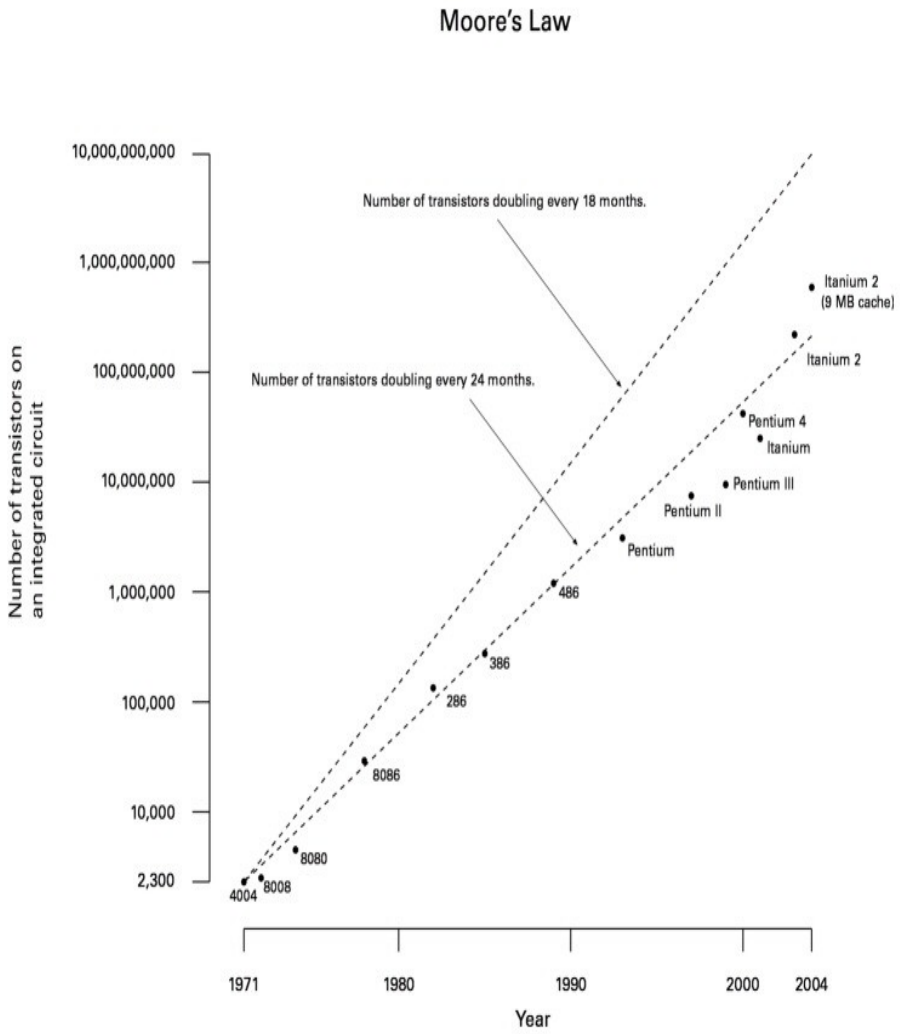
Alchemy, Inria, Orsay

Algorithms, **l**anguages and **c**ompilers to **h**arness the **e**nd of **M**oore's **y**ears

Pérenniser la loi de Moore?

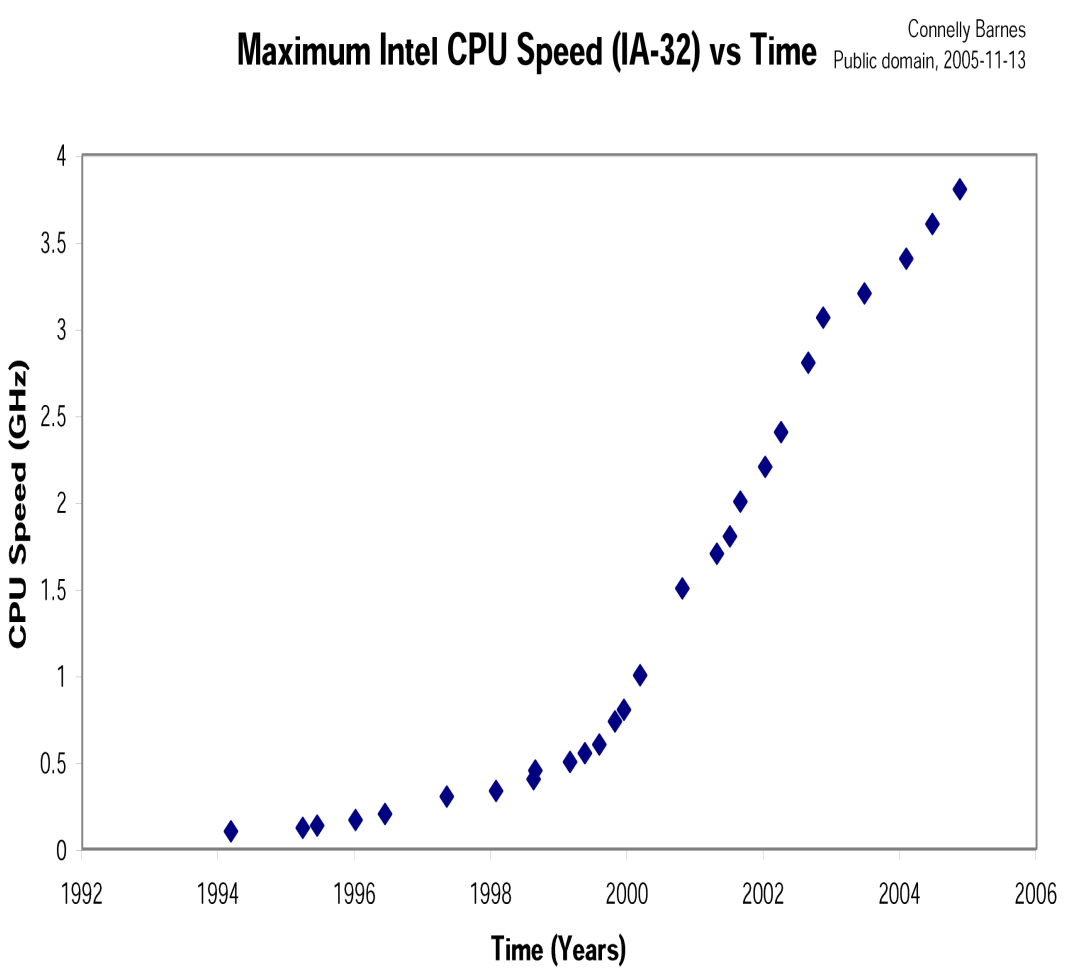
Contributions de Cédric Bastoul, Hugues Berry, Albert Cohen, Benjamin Dauvergne, Marc Duranton, Christine Eisenbeis, Grigori Fursin, Sylvain Girbal, Daniel Gracia-Perez, Frédéric Gruau, Yves Lhuillier, Claire Pagetti, David Parello, Florence Plateau, Louis-Noël Pouchet, Marc Pouzet, Olivier Temam, Sid Ahmed Ali Touati, et al.

Intégration, performance



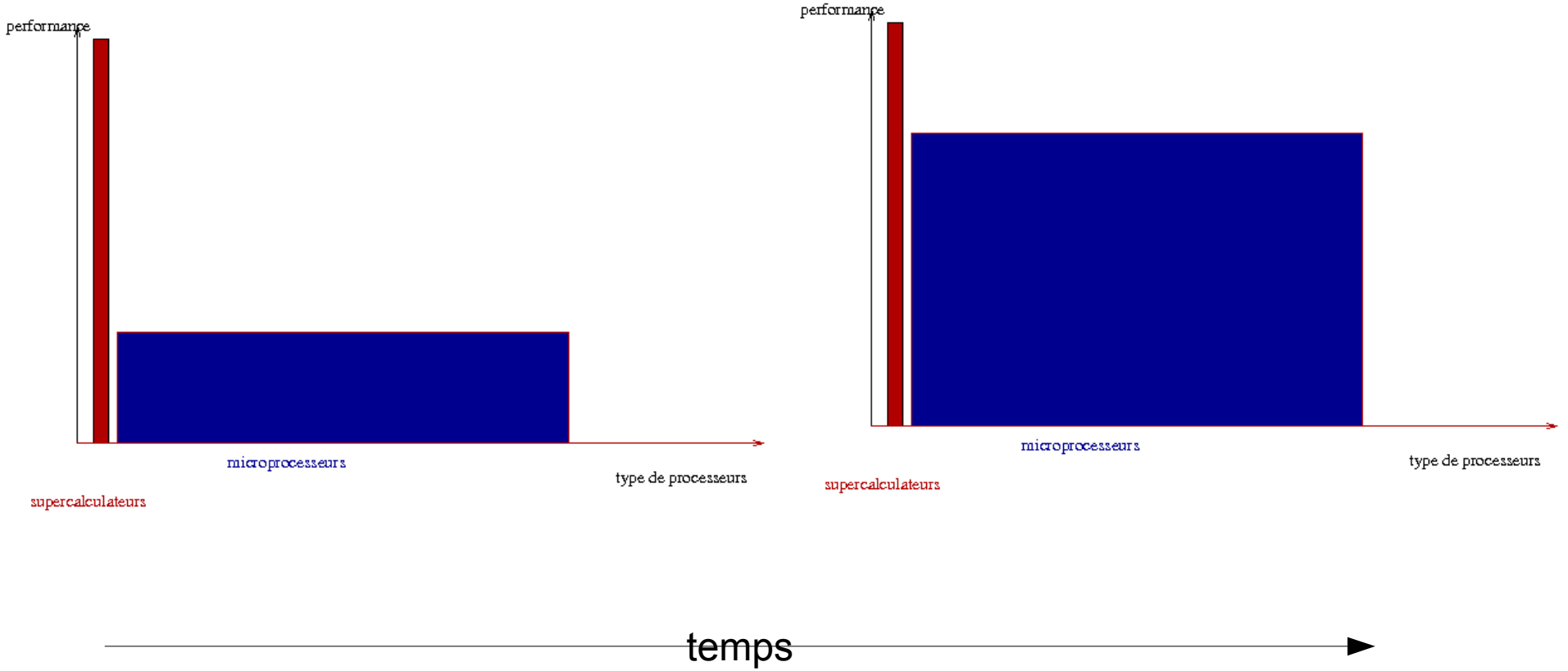
- conséquences:
- faire tenir l'équivalent d'un Cray des années 80 dans une Playstation
- un mainframe d'hier dans une carte d'anniversaire?
- plus de performance?

intégration, performance

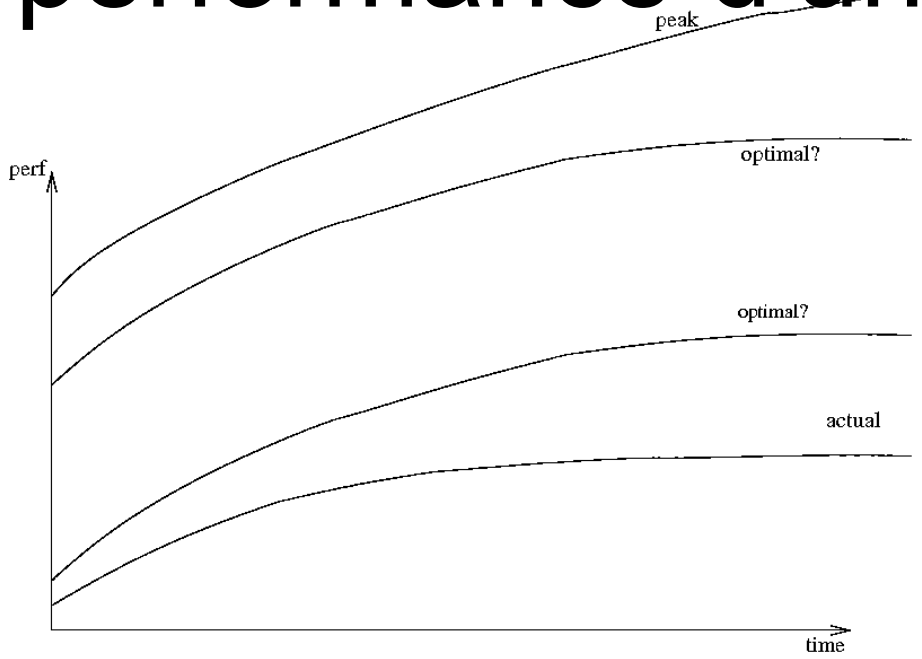


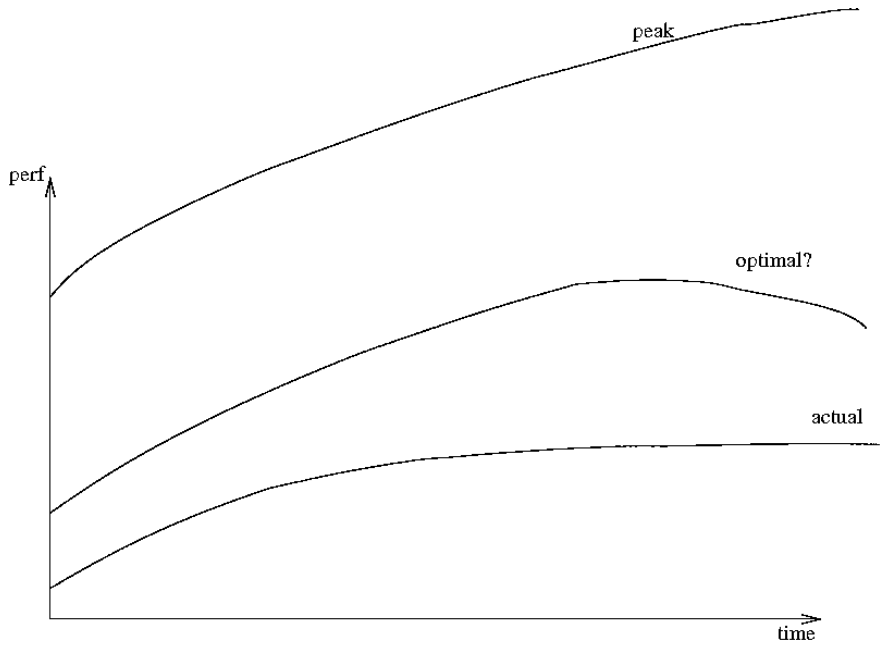
- aymptote verticale? singularité de Vinge et al.
- renversement de tendance (intégration de 30 nm = combien d' atomes??)
- chaleur dissipée, énergie nécessaire
- Intel: arrêt de l'augmentation de la fréquence

« mise à niveau » des microprocesseurs



performance d'un microprocesseur?





- écart croissant entre performance « maximale » et performance réelle
- « c'est la faute au compilateur »
- quelle performance « optimale »?
- est-il possible d'avoir déjà atteint un maximum?

un cas de performances en recul

- GUPS (Global Updates/sec, vitesse d'accès aléatoires à la mémore)
- Architecture (Year) GUPS (4 GB Memory)
  - Cray Y-MP (1988) 0.16
  - Cray C90 (1991) 0.96
  - Cray T90 (1995) 3.2
  - Cray SV1 (1999) 0.7
  - Cray T3E (1996) 2.2
  - Symmetric multiprocessors (2000) 0.35-1.00
  - COTS clusters (2000) 0.35-1.00

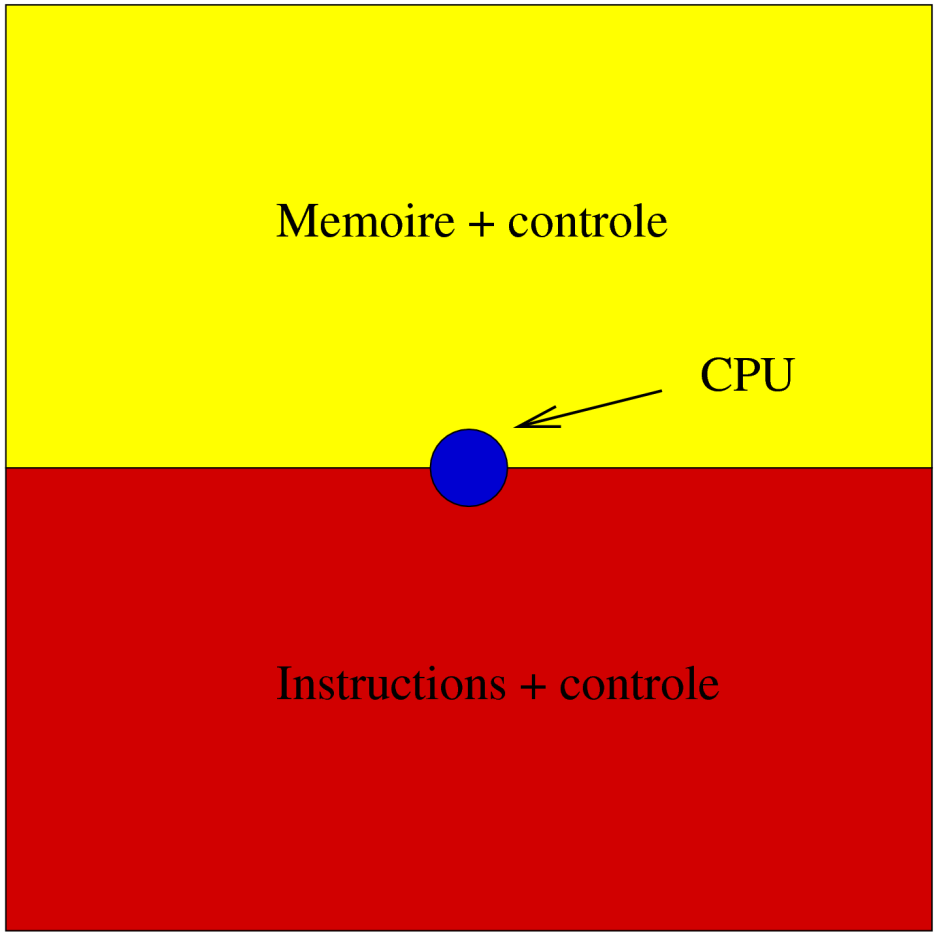
Pérenniser la loi de Moore

- la loi de Moore a donné à un microprocesseur d'aujourd'hui la puissance d'un supercalculateur d'hier
- par l'augmentation de la complexité, et la perte du contrôle par le programmeur/compilateur, quelle marge de manoeuvre?
- pour aller plus loin, il faut reconquérir l'espace de calcul
  - contrôle statique, + de contrôle
  - d'autres paradigmes de calcul, - de contrôle
- revenir aux applications et aux fondamentaux

Pérenniser la loi de Moore

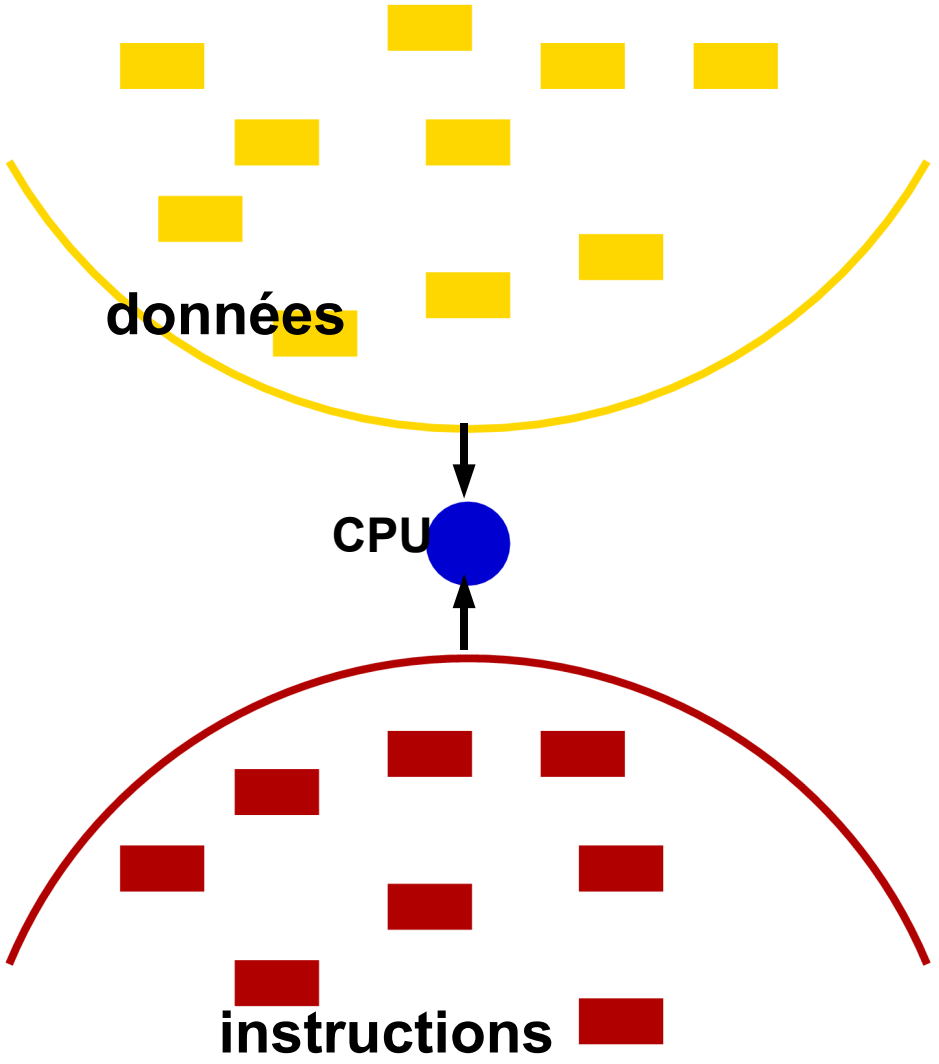
- la loi de Moore a donné à un microprocesseur d'aujourd'hui la puissance d'un supercalculateur d'hier
- **par l'augmentation de la complexité, et la perte du contrôle par le programmeur/compilateur, quelle marge de manoeuvre?**
- pour aller plus loin, il faut reconquérir l'espace de calcul
  - contrôle statique, + de contrôle
  - d'autres paradigmes de calcul, - de contrôle
- revenir aux applications et aux fondamentaux

intégration



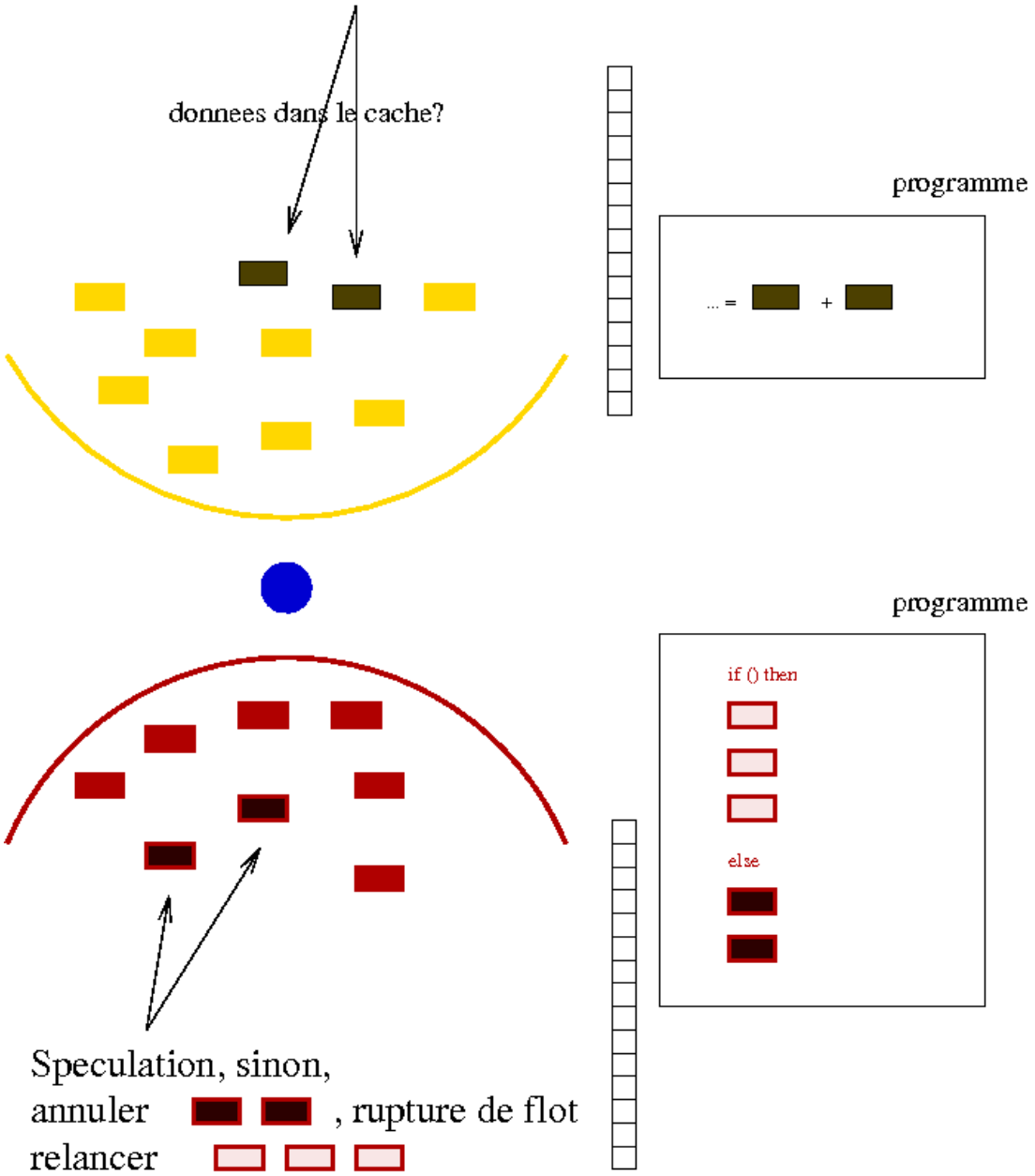
- parallélisme d'instructions (pipeline, multi-lancement simultané d'instructions) dans le CPU
- l'essentiel de la place est consacrée au contrôle
- le coeur de calcul prend de moins en moins de place sur le chip
- processus extrêmement complexes

quel contrôle? (dynamique)



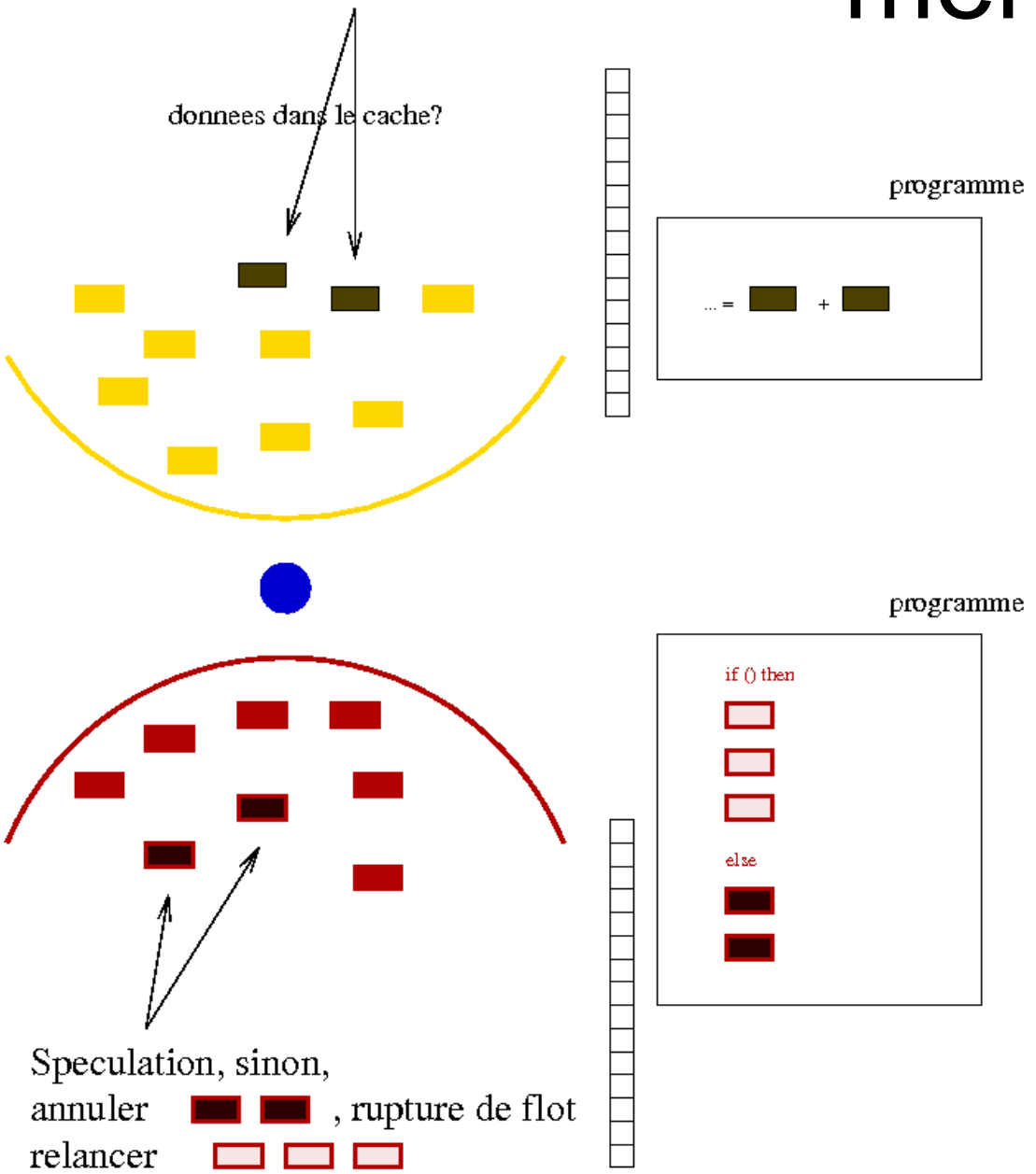
- gestion à l'exécution des transferts de données de la mémoire vers le cache,
- gestion de la parallélisation des instructions, du transfert automatique des données depuis la mémoire

défauts très coûteux



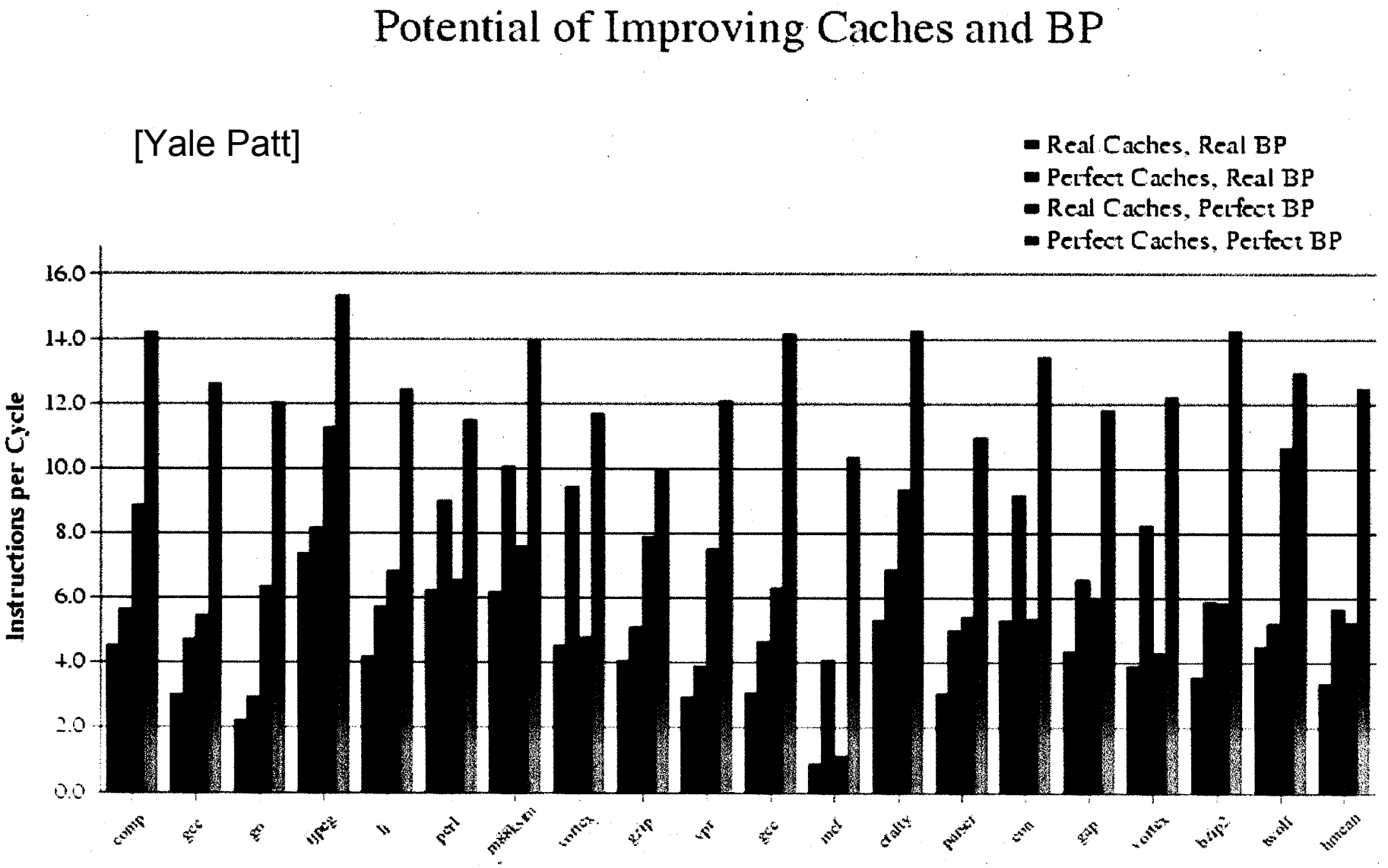
- pénalité d'un défaut de cache = durée d'un accès à la mémoire, de plus en plus long en nombres de périodes d'horloge
- pénalité d'une mauvaise prédiction = taille du pipeline d'instructions (chargement, décodage, exécution), de plus en plus long aussi

défauts très coûteux, le mur de la mémoire

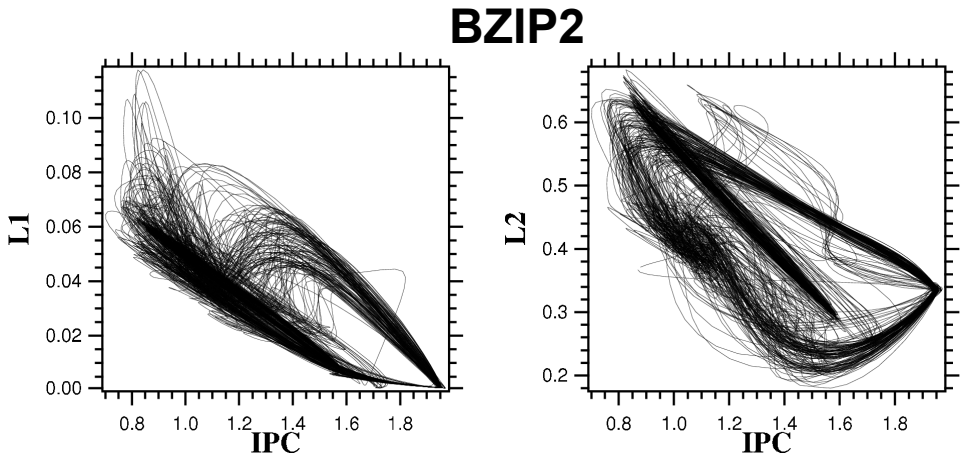


- « en 2020, en attendant qu'un accès mémoire se termine, on pourra exécuter 800 lectures [en cache] et 90000 opérations! » [Snir, Graham]

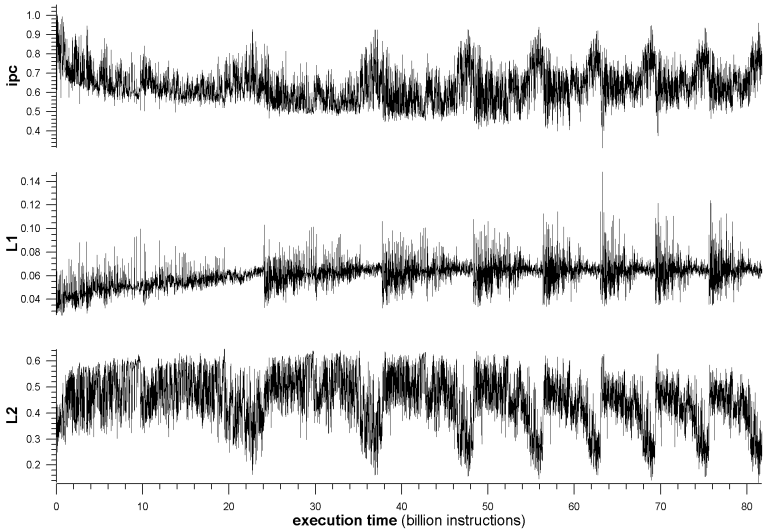
gain par une meilleure connaissance? - architecture



régularité des programmes



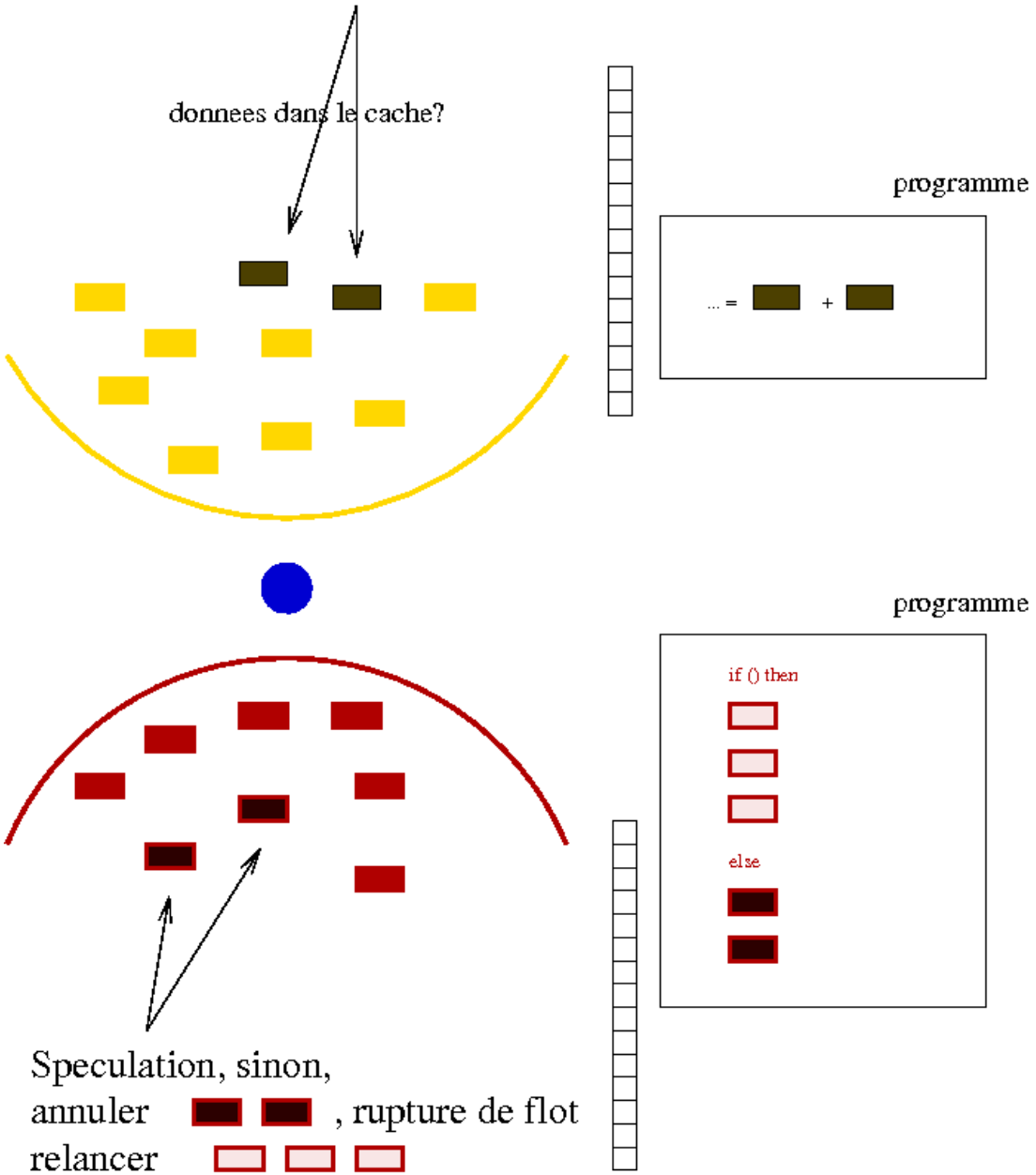
[Berry et al].



VPR

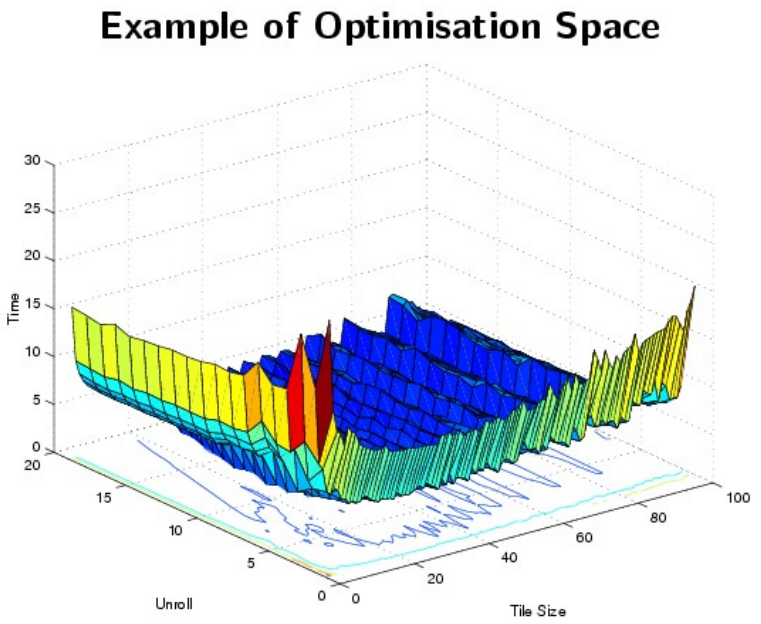
- bzip2, irrégulier, chaotique
- vpr, stochastique?
- comportement impossible à prédire/contrôler
- vision de l'extérieur/observation

défauts très coûteux - compilation



- pénalité d'un défaut de cache = durée d'un accès à la mémoire, de plus en plus long en nombres de périodes d'horloge
- conserver les données réutilisées en mémoire cache, calcul par blocs, préchargement, accès à des données contigües (nécessite une connaissance précise des données)
- pénalité d'une mauvaise prédiction = taille du pipeline d'instructions (chargement, décodage, exécution)accès mémoire
- privilégier de grandes séquences d'instructions sans branchement, dérouler les boucles

rôle du compilateur



Min: Unroll = 3, Tile = 57. Original 4.99 secs, Min 0.56 secs

[Edinburgh University]

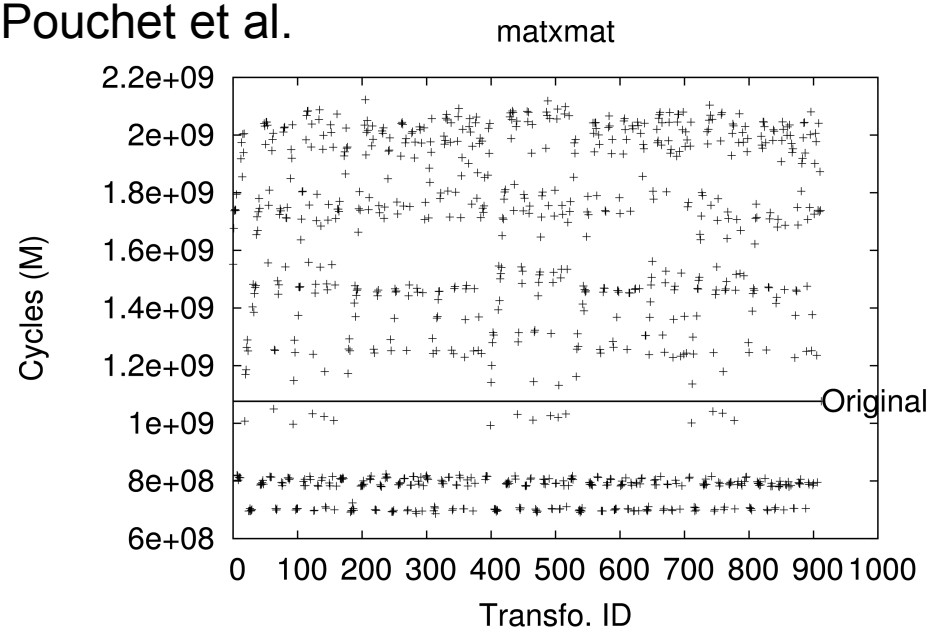
- Loi de T. Proebsting: « les performances des compilateurs doublent tous les 18 ... ans ... »
- pour optimiser, doit pouvoir statiquement prédire le comportement à l'exécution ...??? hors de portée!
- recherche combinatoire, exhaustive, aléatoire, par apprentissage, algorithmes génétiques
- arguments du compilateur, trouver la meilleur combinaison

compilation itérative (statique)

```
for (k=0; k<N; k++)        for (I=0; I<M; I++)
for (I=0; I<M; I++)        for (k=0; k<N; k++)
D[k] += E[I][k]*A[I]       D[k] += E[I][k]*A[I]g
```

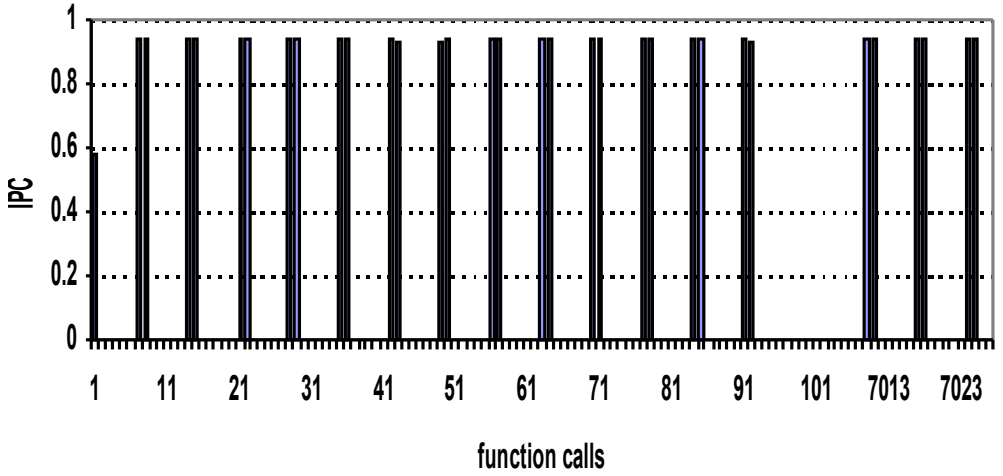
(interchange)

Pouchet et al.

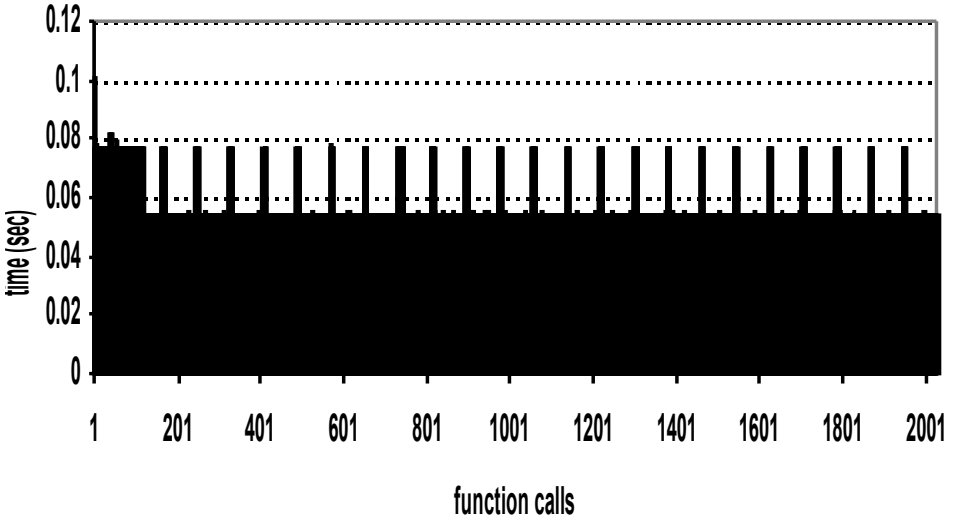


- le compilateur doit comprendre ce qu'il a à faire pour entremêler les instructions
- recherche parmi un ensemble de transformations.
- le nombre de transformations possibles dépend de la connaissance du programme
- champ d'applicabilité?

rôle du programmeur, optimisation dynamique

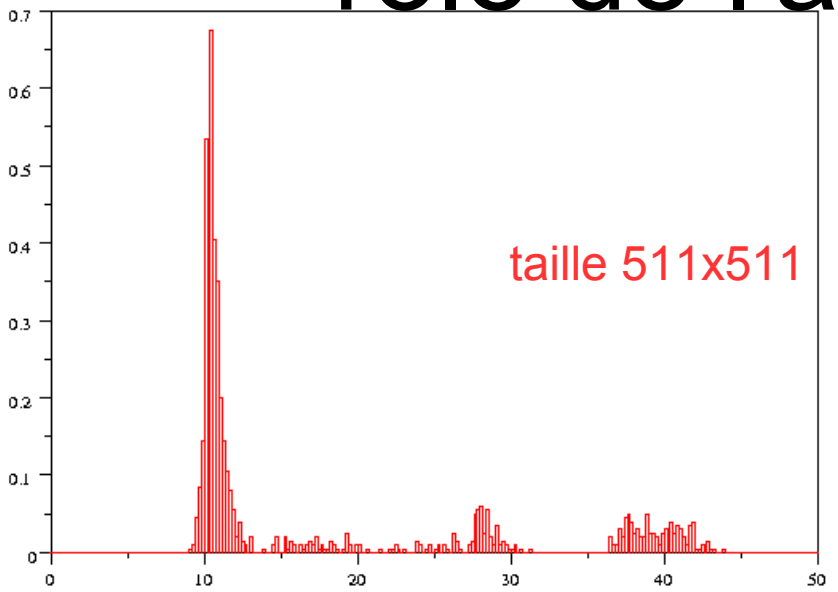


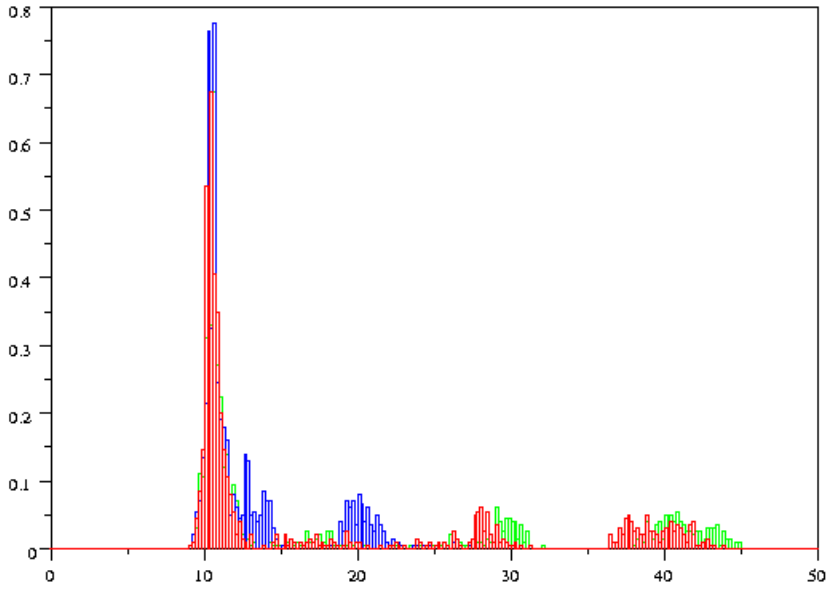
[Fursin et al.]



- rechercher des régularités, stabilités, périodicités dans les performances
- évaluer différentes versions du code dans les premières itérations
- vérifier que les optimisations sont stables en temps

rôle de l'algorithmique



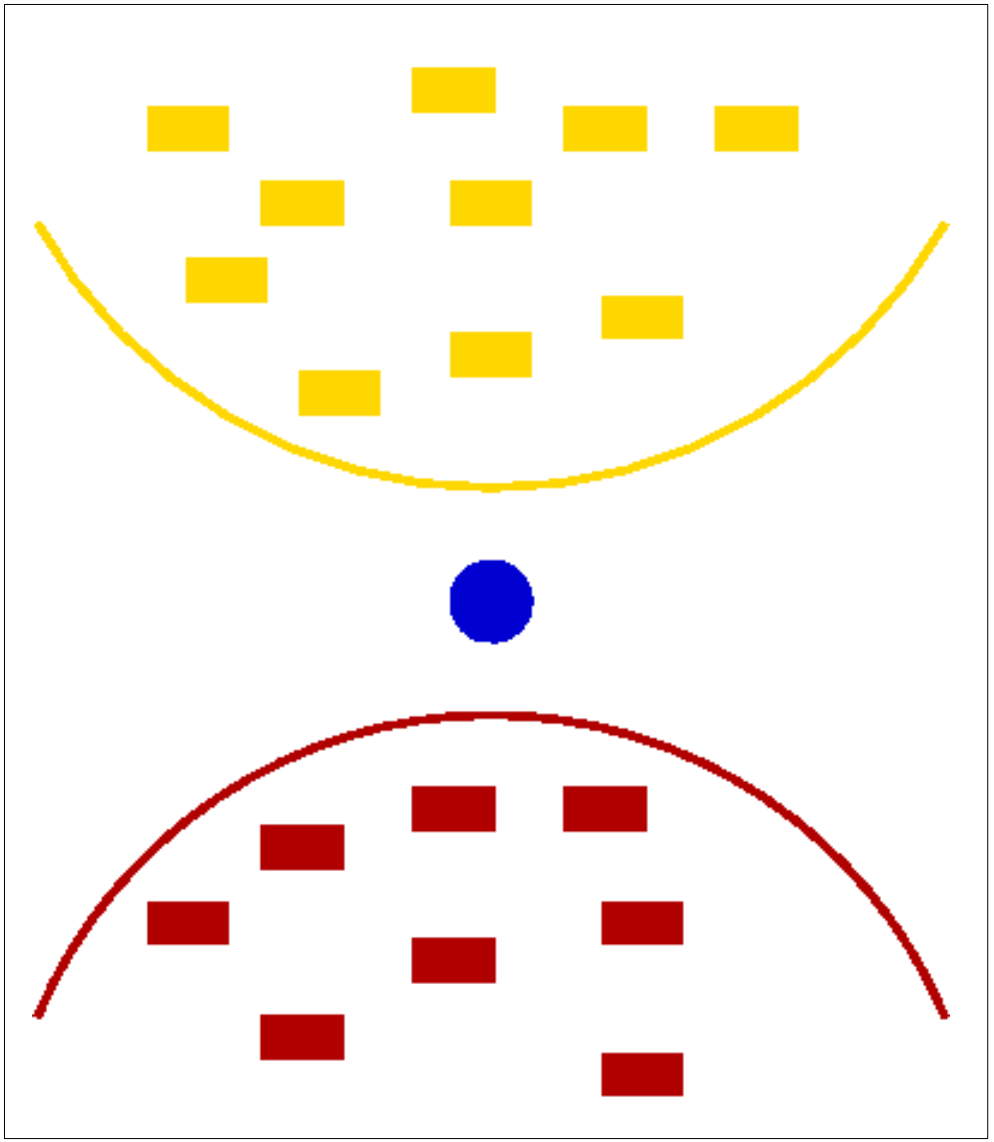


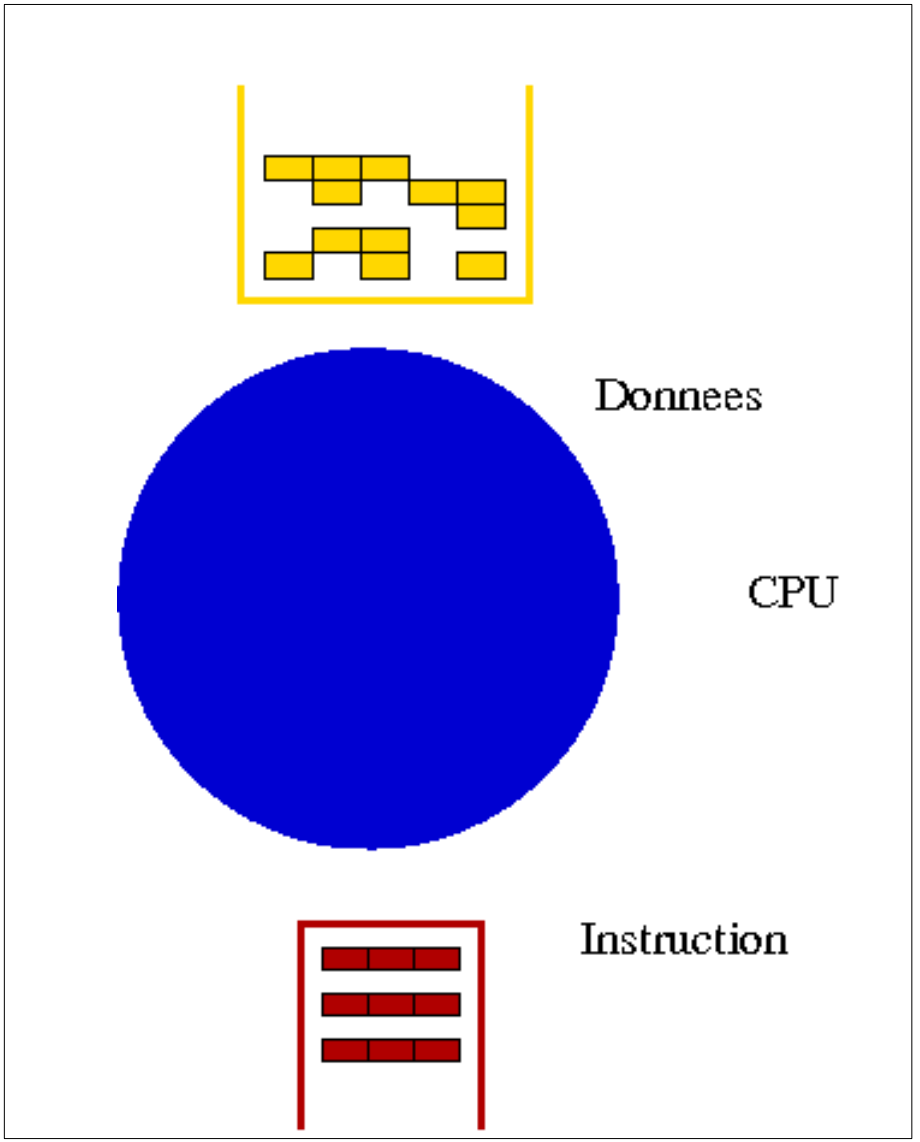
- le hasard fait bien les choses
- décomposition récursive et aléatoire par bloc du produit matrice-matrice
- tirage au sort
  - découpage oui/non
  - si oui tirer les tailles p/q du découpage + récursion
  - si non tirer la forme du produit (i,j,k), (i, k, j), (j, i, k) ...

Pérenniser la loi de Moore

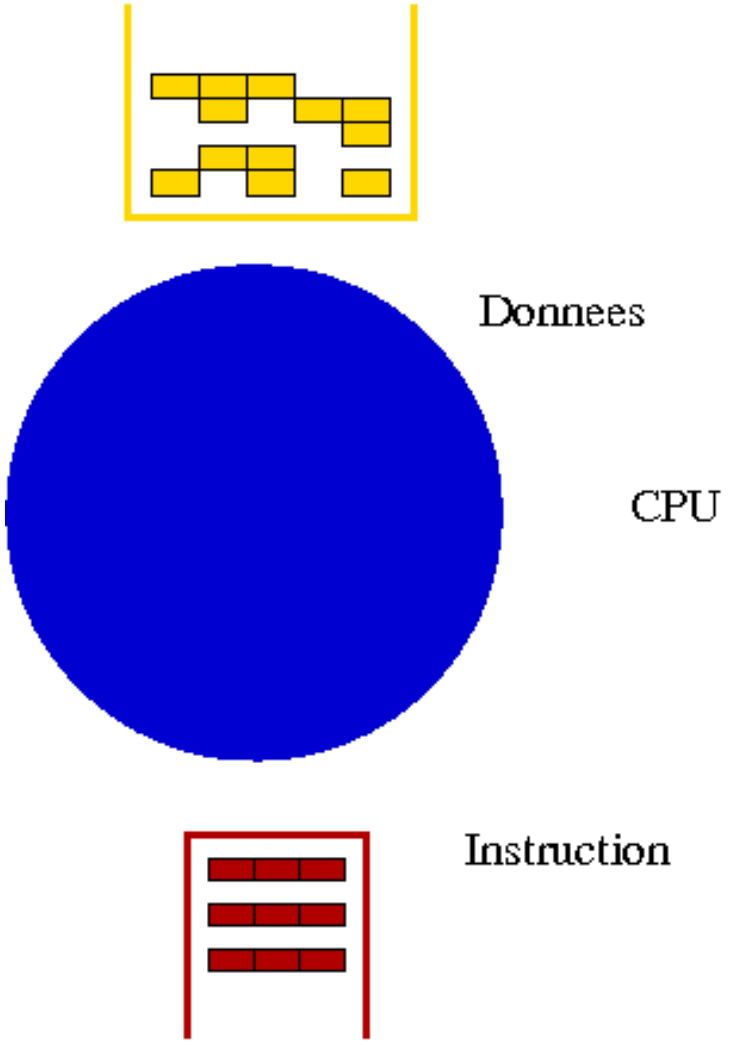
- la loi de Moore a donné à un microprocesseur d'aujourd'hui la puissance d'un supercalculateur d'hier
- par l'augmentation de la complexité, et la perte du contrôle
- pour aller plus loin, il faut reconquérir l'espace de calcul
  - contrôle statique, + de contrôle
  - d'autres paradigmes de calcul, - de contrôle
- revenir aux applications et aux fondamentaux

dynamique/statique





(re-conquête de l'espace) contrôle statique

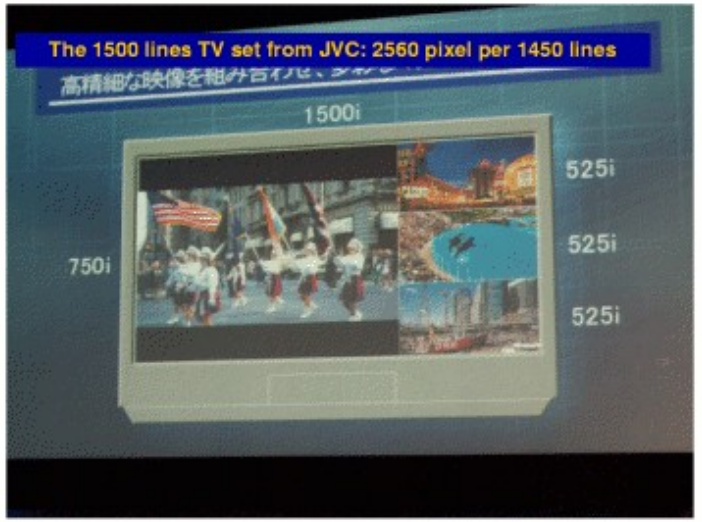


exemples : Trimédia (Philips), ST-200 (St-MicroElectronics, Crusoë (Transmeta), ...

- tout est prévu à la compilation
- pas de contrôle dynamique coûteux
- temps prédictible
- beaucoup plus efficace
- encore extensible
- nécessite une très bonne connaissance du programme
- en pratique = gestion d'une mémoire statique délicate
- gestion des if par prédicats

processeur vidéo embarqué à hautes performances

application



algorithme

entrée et sortie des images: 30 Hz

HD pixels 30 x 1920 x 1080 = 62,208,000Hz

SD pixels 30 x 720 x 480 =10,368,000Hz

(6 fois plus lentes)

HF: 8:3 – filtre horizontal

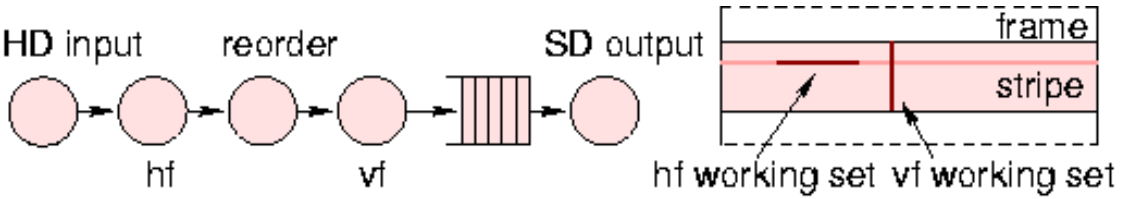
transposition vertical->horizontal par bande de 6 lignes

VF: 9:4 – filtre vertical

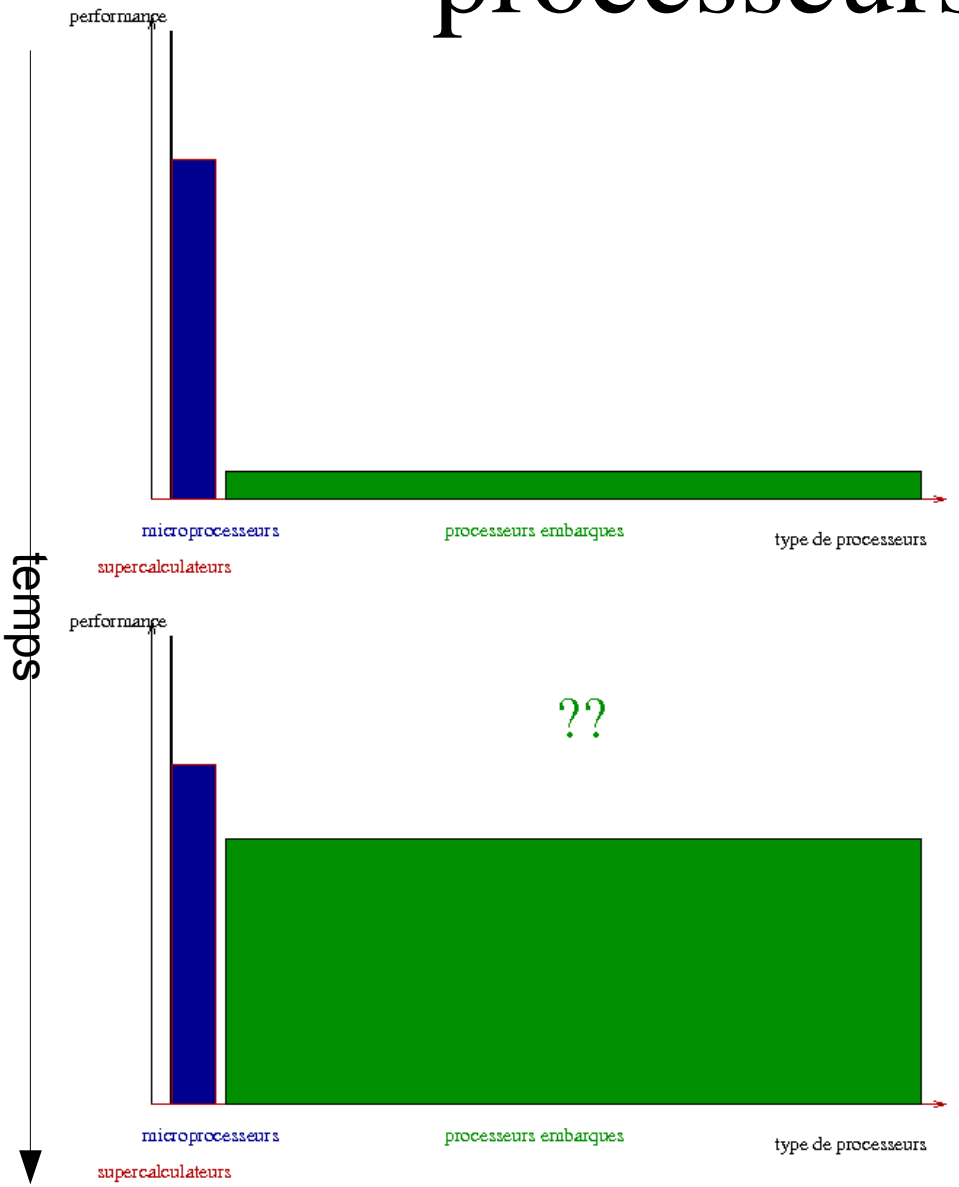
matériel

- calcul intensif + temps réel + embarqué: requis = 150 Gops/Watt
- -> parallélisme à grande échelle, statique
- **contrôle des ressources dans le langage de programmation (temps, buffers de données, CPU)**

modèle de programmation

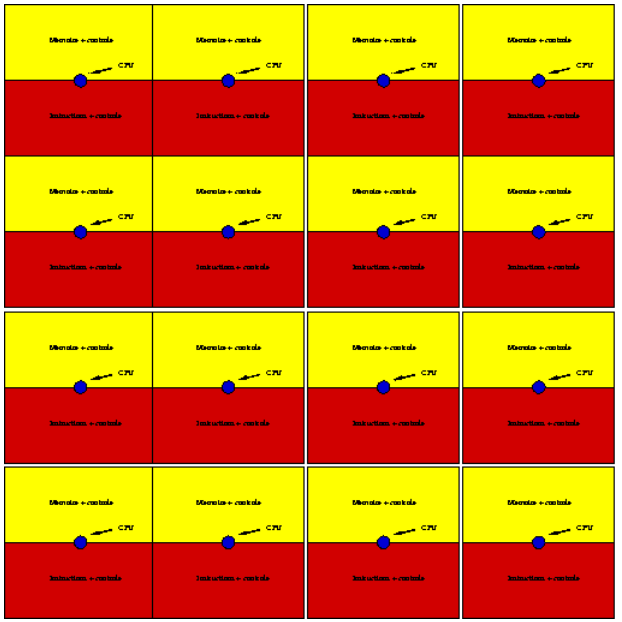


processeurs embarqués



- « niche » économique
  - programmes spécialisés comme dans les super -calculateurs (convergence?)
  - mais contraintes plus fortes, taille, mobilité, temps réel, température, consommation, robustesse, ...

(re-conquête de l'espace) parallélisme multi-coeurs

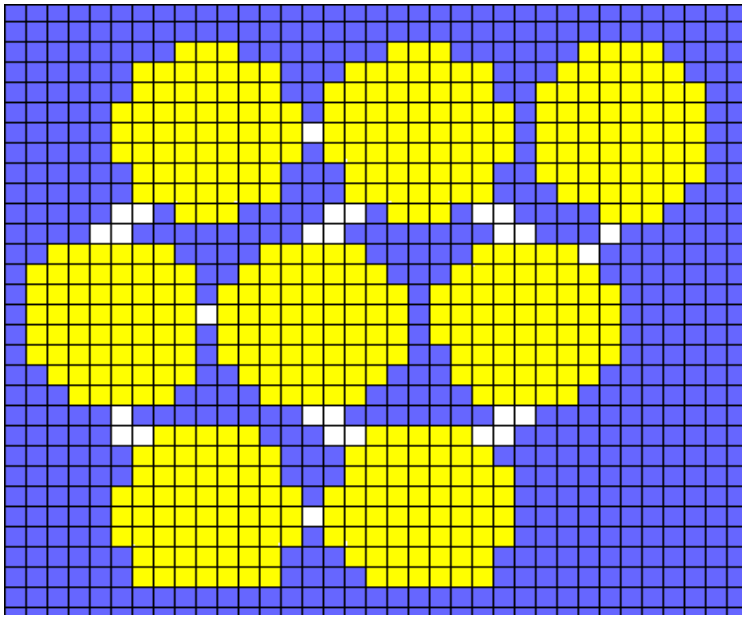


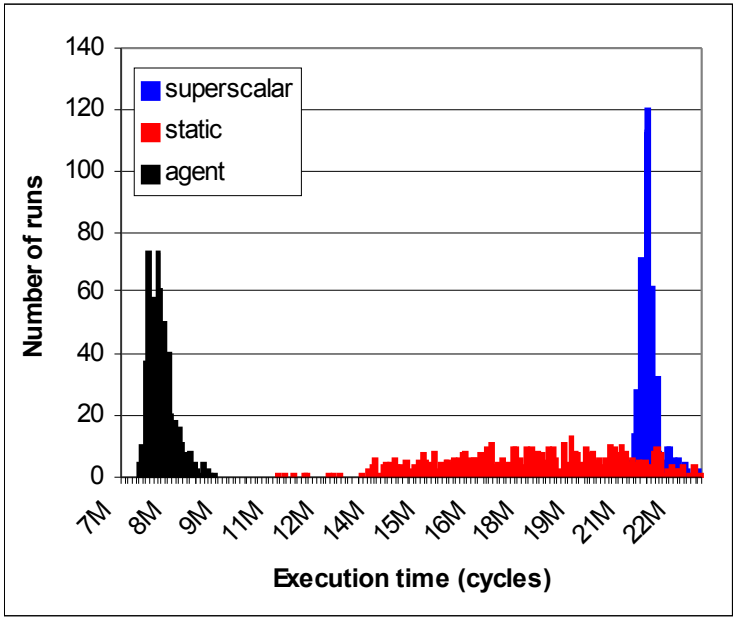
- parallélisme sur le chip (CMP)
- contrôle?
- communication?
- efficacité?
- quoi de neuf?
- réalité
- multiprogrammation

Pérenniser la loi de Moore

- la loi de Moore a donné à un microprocesseur d'aujourd'hui la puissance d'un supercalculateur d'hier
- par l'augmentation de la complexité, et la perte du contrôle par le programmeur/compilateur, quelle marge de manoeuvre?
- pour aller plus loin, il faut reconquérir l'espace de calcul
  - contrôle statique, + de contrôle
  - d'autres paradigmes de calcul, - de contrôle
- revenir aux applications et aux fondamentaux

quel contrôle?

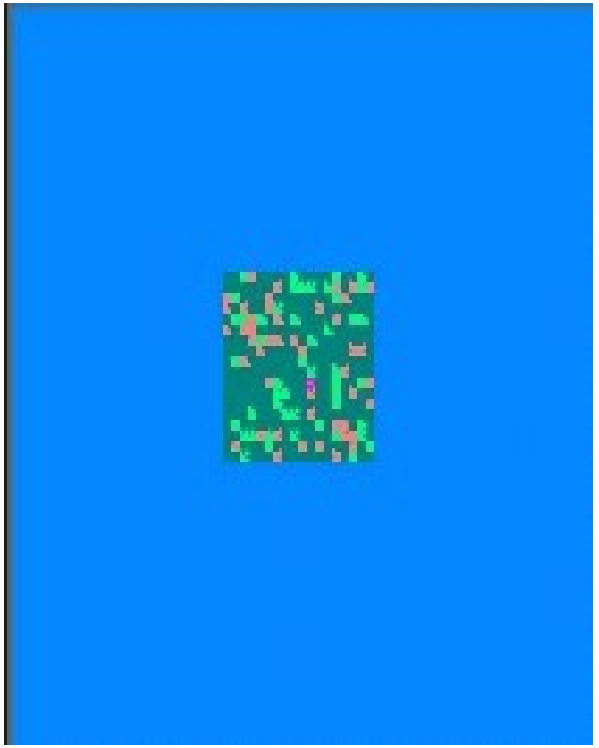




- changer le paradigme de programmation
- renoncer au contrôle global
- vers du contrôle délocalisé, des architectures moins complexes
- application à moyen terme aux SMT (Simulatenous Multi Threading architectures) et CMP(Chip Multi Processing)
- gestion automatique du parallélisme/utilisation des ressources

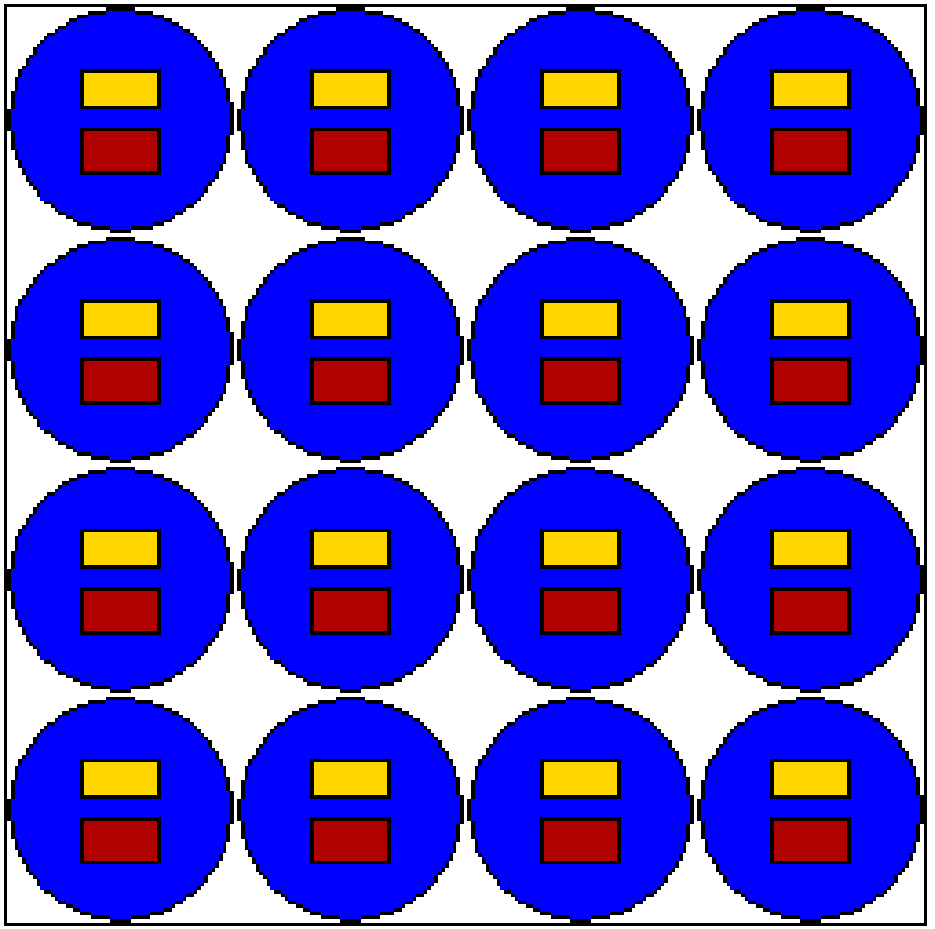
Blob computing

[Gruau et al.]



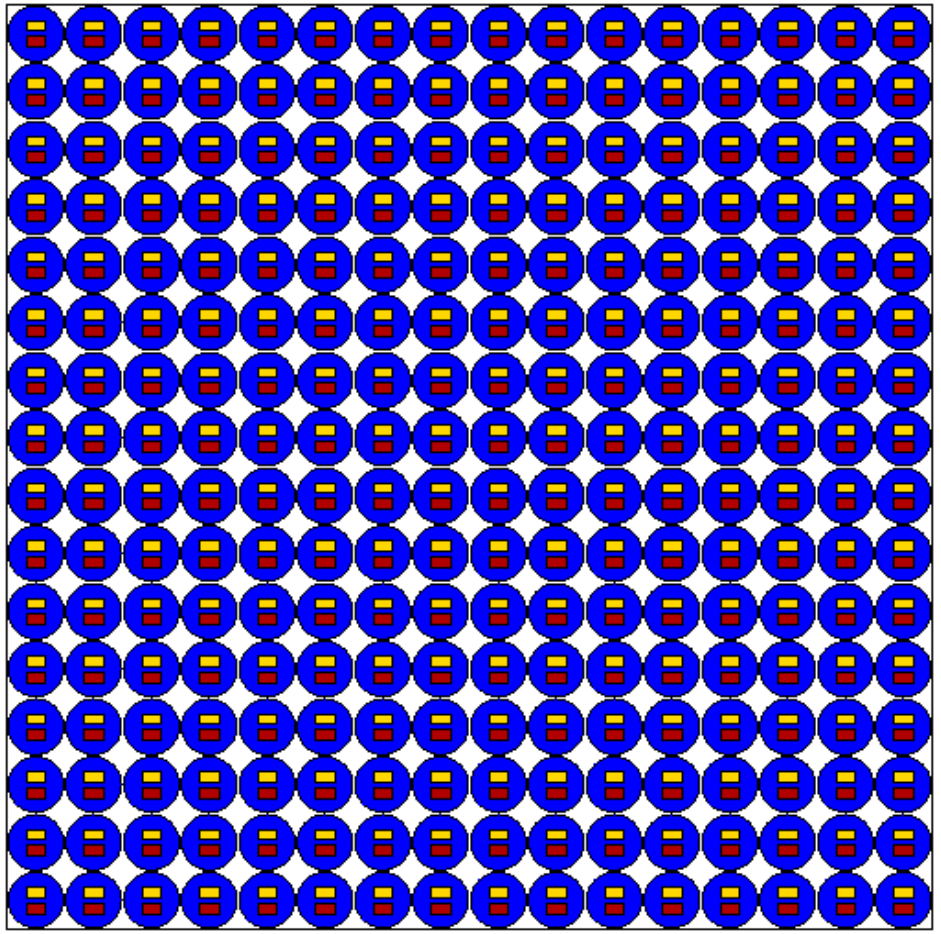
- l'utilisateur décrit le dépliement de son graphe de calcul/processus à l'aide de quelques primitives
- le placement (par exemple sur une grille d'automates cellulaires) se fait par simulation de processus physiques

calcul spatial



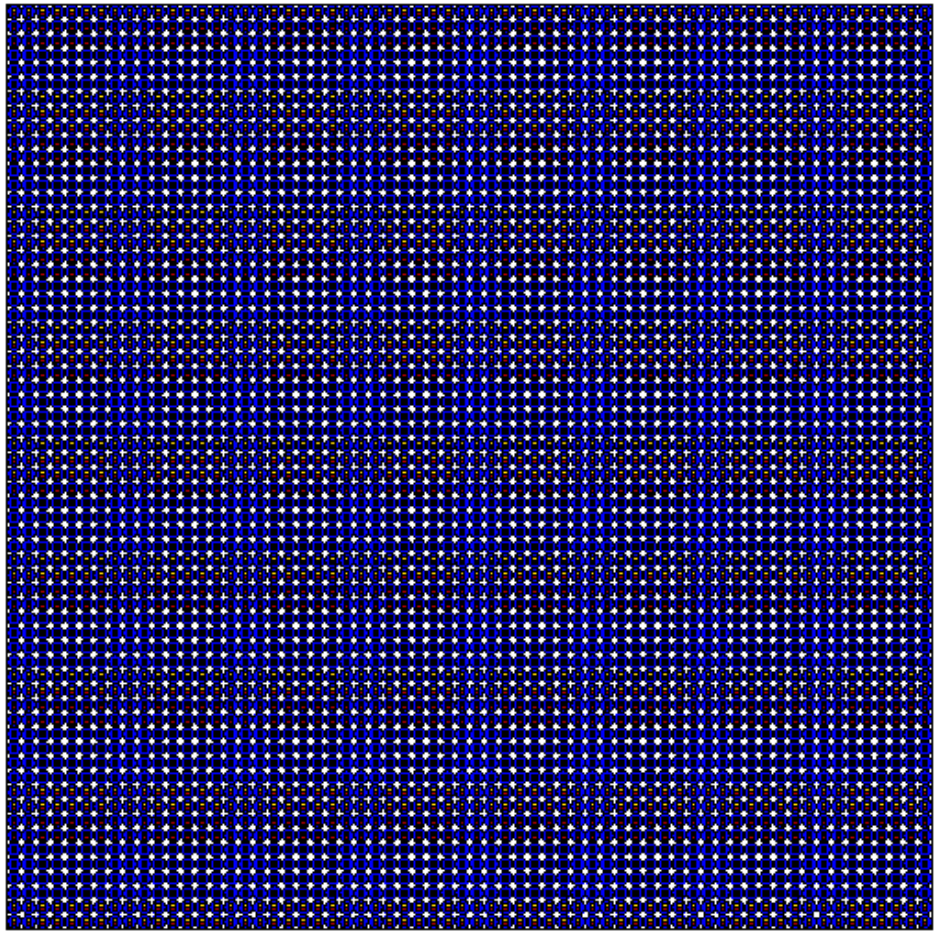
- délocaliser le contrôle
- nouveaux paradigmes
- régularité du réseau?
- nouveau? ...

calcul spatial



- délocaliser le contrôle
- nouveaux paradigmes
- régularité du réseau?
- nouveau? ...
- réseau systolique, automates cellulaires
- extensible?

calcul spatial



- délocaliser le contrôle
- nouveaux paradigmes?
- régularité du réseau?
- nouveau? ...
- réseaux systoliques, automates cellulaires
- pour quelles applications?

Pérenniser la loi de Moore

- la loi de Moore a donné à un microprocesseur d'aujourd'hui la puissance d'un supercalculateur d'hier
- par l'augmentation de la complexité, et la perte du contrôle par le programmeur/compilateur, quelle marge de manoeuvre?
- pour aller plus loin, il faut reconquérir l'espace de calcul
  - contrôle statique, + de contrôle
  - d'autres paradigmes de calcul, - de contrôle
- revenir aux applications et aux fondamentaux

à la conquête de l'espace

- multi-coeurs sur puce, un système encore plus complexe!
- spatial computing, un nouveau mot pour « parallélisme »?
- un nouveau paradigme?
- comportement local/comportement global, système émergent
- « résultat »? entrées-sorties?
- reprendre les systèmes physiques, biologiques, directement dans leurs fondamentaux? [phénomènes locaux-> edp -> discrétisation -> placement]
- quelles applications?

le calcul et la nature

- calcul quantique
- calcul sur ADN
- l'ADN calcule
- le cerveau calcule
- une pierre calcule [JL Giavitto, Dagstuhl, septembre 2006] ???

Pérenniser la loi de Moore

- la loi de Moore a donné à un microprocesseur d'aujourd'hui la puissance d'un supercalculateur d'hier
- par l'augmentation de la complexité, et la perte du contrôle par le programmeur/compilateur, quelle marge de manoeuvre?
- pour aller plus loin, il faut reconquérir l'espace de calcul
  - contrôle statique, + de contrôle
  - d'autres paradigmes de calcul, - de contrôle
- revenir aux applications et aux fondamentaux

Un mur de performances présent depuis longtemps!

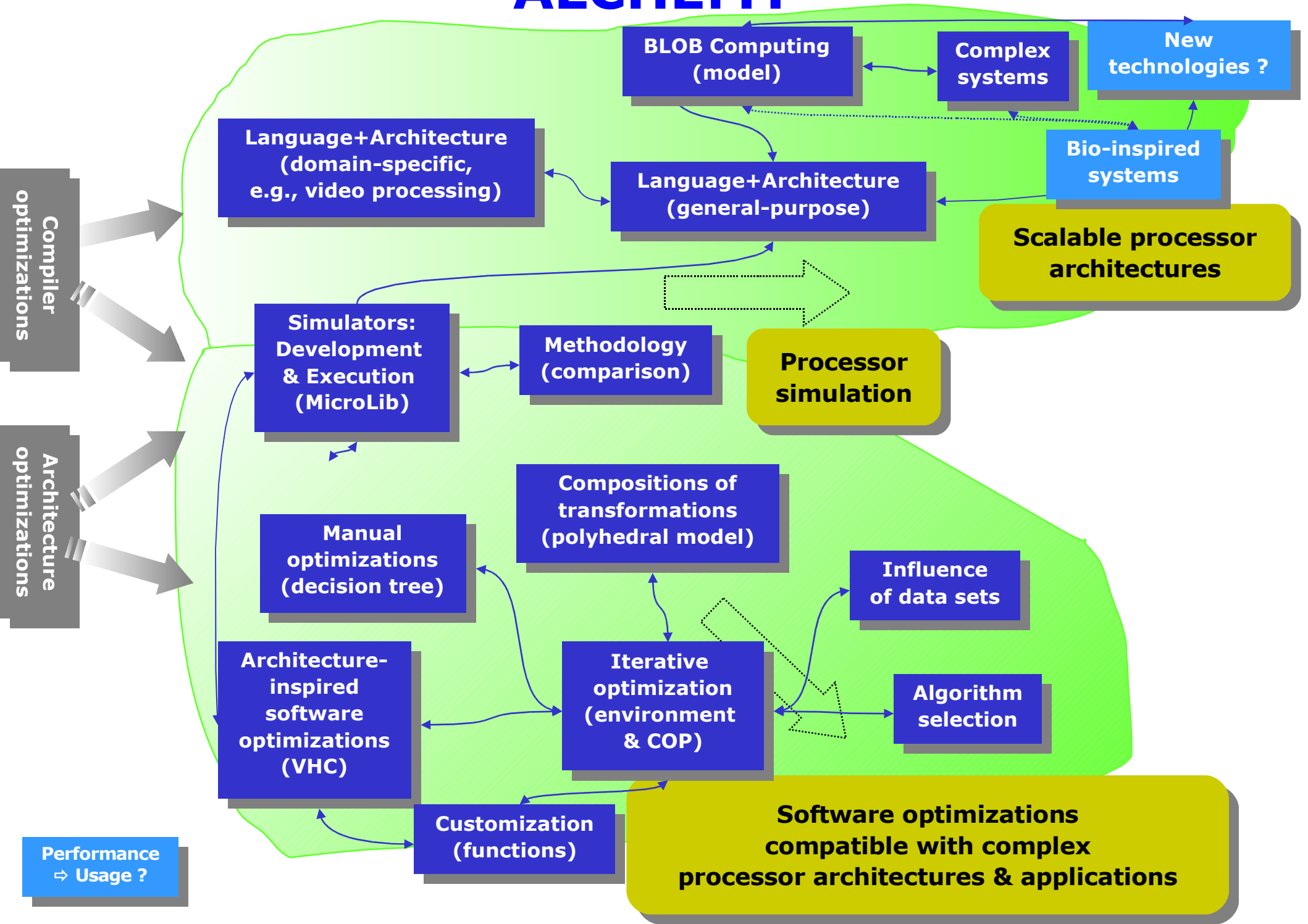
Pérenniser la loi de Moore

- la loi de Moore a donné à un microprocesseur d'aujourd'hui la puissance d'un supercalculateur d'hier
- par l'augmentation de la complexité, et la perte du contrôle par le programmeur/compilateur, quelle marge de manoeuvre?
- pour aller plus loin, il faut reconquérir l'espace de calcul
  - contrôle statique, + de contrôle
  - d'autres paradigmes de calcul, - de contrôle
- revenir aux applications et aux fondamentaux

Un mur de performances présent depuis longtemps!

Like a lawyer, an optimizing compiler will be able to do a better job if it is told the whole story.” Toffoli

ALCHEMY


BLOB Computing (model)
Complex systems
New technologies ?
Language+Architecture (domain-specific, e.g., video processing)
Language+Architecture (general-purpose)
Bio-inspired systems
Scalable processor architectures
Compiler optimizations
Simulators: Development & Execution (MicroLib)
Methodology (comparison)
Processor simulation
Architecture optimizations
Compositions of transformations (polyhedral model)
Manual optimizations (decision tree)
Influence of data sets
Architecture-inspired software optimizations (VHC)
Iterative optimization (environment & COP)
Algorithm selection
Customization (functions)
Software optimizations compatible with complex processor architectures & applications
Performance ⇨ Usage ?