# 7. Délocaliser les calculs en mémoire : le concept « processing-in-memory »

Dominique Lavenier

Le cœur d'un ordinateur est principalement composé de deux grandes parties : une unité de calcul et une mémoire, suivant le modèle proposé par Von Neumann vers le milieu des années 1940. Les données transitent à grande vitesse entre ces deux organes qui, en général, sont implémentés dans des composants différents. Pratiquement, un ordinateur portable, un serveur de calcul et un nœud d'un super calculateur sont bâtis sur le même schéma : ils sont composés d'un processeur connecté à une ou plusieurs barrettes mémoires. L'ensemble est relié au reste du système pour acquérir les données et restituer les résultats des calculs.

## Une mémoire sous pression

Les technologies des processeurs et des mémoires sont différentes et n'ont pas évolué au même rythme au cours des dernières décennies. La puissance de calcul des processeurs (le nombre d'opérations effectuées par seconde) augmente plus vite que le débit de la mémoire (le nombre de données qui peuvent être transférées de/vers la mémoire par seconde). Cet écart se creuse au fil des années. Il a pour effet de dégrader les performances des systèmes, les processeurs passant de plus en plus de temps à attendre les données en provenance de la mémoire. Pour limiter cet effet, appelé « *memory wall* », les processeurs intègrent un dispositif de mémoire cache. Ce sont des mémoires de faible capacité mais rapides d'accès, directement intégrées sur la puce du processeur, et qui sont organisées en une hiérarchie complexe pour fluidifier l'accès aux données. L'efficacité des mémoires cache repose sur le principe de localité. Un programme œuvre souvent sur des structures de données « régulières », par exemple des tableaux. Le comportement du programme est donc assez prévisible, ce qui permet d'anticiper le chargement de blocs de données dans les mémoires cache avant leur utilisation, et ainsi éviter des cycles d'attente au niveau du processeur. Le traitement des images, représentées par des tableaux de pixels, en est un exemple typique. Mais en dépit de ce dispositif efficace et de la stagnation des fréquences d'horloge* des processeurs depuis le milieu des années 2000, le problème persiste. La puissance de calcul des processeurs ne cesse de croître. L'évolution continue des densités d'intégration (nombre de transistors par unité de surface) incite à multiplier les unités de calcul (les cœurs) sur une même puce. La pression sur la mémoire s'intensifie d'autant plus que le nombre de cœurs est important. Le volume de données qui peut être échangé entre le processeur et le mémoire par unité de temps (bande passante) devient donc critique. C'est un facteur limitant qui freine significativement l'exécution des programmes. Cela se ressent plus particulièrement sur les applications qui gèrent d'importants volumes de données et pour lesquelles l'accès aux données n'est pas régulier, ce qui met en défaut le principe de localité et rend le dispositif de mémoire cache moins efficace. Par ailleurs, la consommation énergétique des systèmes est un autre problème à résoudre. Comme nous l'avons déjà indiqué, l'écart entre la vitesse d'exécution d'une instruction et de son accès à la mémoire ne cesse de grandir. Le même phénomène apparaît d'un point de vue énergétique. Les accès mémoire deviennent toujours plus coûteux, notamment pour les applications qui ne peuvent tirer parti des mémoires cache en raison d'un accès aléatoire aux données par exemple. Cela peut représenter jusqu'à 60 % de l'énergie totale

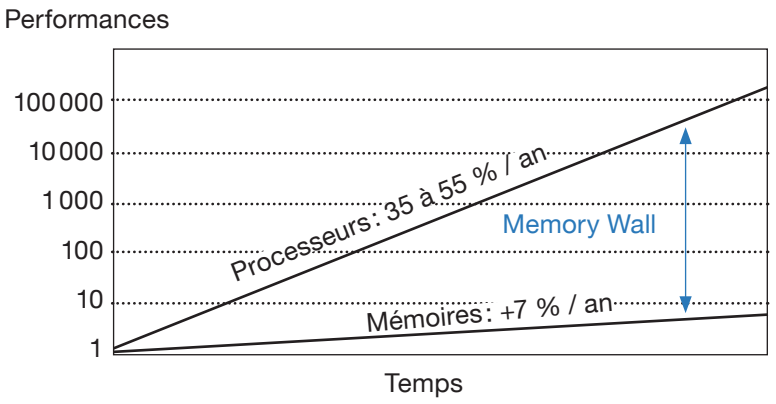


Fig. 1 – **L'effet « Memory Wall ».** Les performances des processeurs et des mémoires n'évoluent pas à la même vitesse. L'écart se creuse au fil des années. L'alimentation en données des processeurs est de plus en plus problématique, ce qui est particulièrement pénalisant pour les applications centrées sur les données où les échanges entre le processeur et la mémoire sont importants ■

consommée. Le concept de « *Processing-in-Memory* » (PiM) a pour objectif principal de contrer l'effet « memory wall » tout en exploitant les gains énergétiques liés à la réduction des transferts mémoire. L'idée générale est de rapprocher les calculs des données, voire d'effectuer les calculs directement dans les cellules mémoire, afin de limiter les allers-retours entre le processeur et la mémoire. Bien sûr, au niveau de la mémoire, tous les types de calculs ne sont pas possibles, comme des multiplications sur des nombres réels. Mais des opérations simples et répétitives qui génèrent d'importants flux de données peuvent être effectuées localement et alléger ainsi le trafic entre le processeur et la mémoire (figure 1). Le paradigme PiM n'est pas nouveau. Les premières idées sont apparues dans les années 1970, mais elles n'ont été véritablement explorées que 20 à 25 ans plus tard, lorsqu'il a été possible d'envisager une intégration commune d'un processeur et d'une mémoire conséquente sur la même puce. Le projet Berkeley Intelligent RAM (1996-2004) illustre une des premières tentatives. Aujourd'hui, les technologies sont disponibles et de nombreux domaines applicatifs tournés vers la gestion de grandes masses de données sont en attente de solutions matérielles efficaces et peu consommatrices d'énergie.

## Faire tomber le mur de la mémoire

Le concept de *Processing-in-Memory* est générique et ne fait pas référence à une architecture matérielle unique. Il existe deux grandes catégories : le calcul dans la mémoire (*in-memory computing*) et le calcul près de la mémoire (*near-memory computing*). La première intègre l'idée que des calculs puissent se réaliser directement au sein de chaque cellule mémoire. La seconde considère de multiples petites unités de calcul, activables simultanément au niveau d'un module mémoire qui est l'élément constitutif d'un composant mémoire. La nature des calculs est donc différente. Dans les deux cas, cependant, il faut spécifier les opérations à réaliser au niveau de la mémoire et il n'y a pas d'échanges de données avec le processeur.

Le calcul « dans la mémoire » nécessite des cellules mémoire complexes, une organisation inter-cellules différente et des circuits périphériques adaptés. Dans cette catégorie, deux implémentations, sont possibles. La première est centrée sur les points mémoire, chaque point intégrant des éléments matériels permettant des opérations logiques simples avec les points voisins. Le résultat du calcul modifie (ou pas) le contenu de la cellule mémoire. Il n'y a pas de transfert de données à proprement parler. Le parallélisme est très massif, un grand nombre de calculs pouvant s'exécuter simultanément sur l'ensemble des points mémoire. La seconde implémentation utilise les circuits périphériques d'entrées/sorties (notés E/S sur la figure 2) qui sont normalement dédiés à la gestion des commandes d'écriture et de lecture et d'aiguillage des données. Dans le cas présent, ils sont modifiés pour effectuer en plus des opérations arithmétiques et logiques. Lors d'un calcul, les données transitent sur les bus internes du banc mémoire, mais n'en sortent pas. Suivant l'organisation matricielle de la mémoire, plusieurs opérations peuvent s'exécuter simultanément. Par rapport à l'architecture précédente, le parallélisme est moins élevé, mais reste quand même important. L'architecture IMI (*In-Memory-Intelligence*), proposée par la société Micron (2017), illustre très bien cette seconde approche. Une mémoire IMI de 8 Go est bâtie autour d'une mémoire DRAM (mémoire vive classique) répartie en 8 bancs. Chaque banc est découpé en 64 sous-bancs correspondant à des matrices de 1024 lignes de 16 Kbits (colonnes). À chaque colonne est associé un petit bloc logique capable d'effectuer quelques opérations élémentaires. Un banc intègre donc 1 048 576 éléments de calcul (64 x 16 384), soit plus de 8 millions pour l'ensemble de la mémoire. Le mode d'exécution est SIMD*, avec un contrôleur au niveau de chaque banc

qui diffuse les instructions à l'ensemble des blocs logiques. Chaque banc est autonome et peut donc exécuter un traitement différent.

Le principe du calcul « près de la mémoire » est d'associer une unité de traitement à chaque banc du composant mémoire. Les données vont donc sortir du banc, être traitées, puis réécrites dans ce même banc, sans pour autant sortir de la puce. Le parallélisme est moindre comparé aux deux autres architectures, mais les traitements peuvent être plus complexes. Les architectures PiM qui exportent le traitement à l'extérieur des bancs mémoire diffèrent selon la complexité de l'unité de calcul. Celle-ci peut se résumer à quelques circuits logiques capables d'effectuer des opérations arithmétiques et logiques élémentaires. Elle peut, à l'opposé, s'incarner sous la forme d'un processeur complet, dont le banc mémoire joue le rôle de mémoire principale. La technologie « *3D stacked* », apparue au début des années 2010, se prête bien à une implémentation de cette approche. Cette technologie peut être vue comme un circuit intégré en 3 dimensions réalisé à partir d'un empilement de plusieurs couches technologiques hétérogènes. La majorité des couches est constituée de modules DRAM*. La couche inférieure est une couche logique sur laquelle peuvent être intégrées des unités de calcul. Les couches communiquent entre elles par des liaisons verticales, ce qui réduit les distances des communications et améliore ainsi la latence et l'efficacité énergétique. La DRAM et la logique étant construites sur des couches séparées, elles utilisent des technologies distinctes, chacune optimisant ses propres critères : la vitesse de calcul pour la couche logique, les coûts et l'efficacité énergétique pour les couches mémoire. L'architecture « *Hybrid Memory Cube* » a été la première architecture à exploiter cette technologie. La couche logique est responsable du séquençage de la DRAM, de son rafraîchissement, de l'acheminement des données, de la correction des erreurs et de l'interconnexion à grande vitesse avec l'unité centrale. Cette séparation en couches technologiques n'est cependant pas obligatoire. La société UPMEM propose une mémoire DRAM qui associe un processeur de type RISC à chaque banc de 64 Mo. Le processeur est conçu avec la technologie DRAM et est intégré sur le même support.

## Le défi des nouvelles architectures de mémoire

Beaucoup de conditions sont aujourd'hui réunies pour que les architectures PiM trouvent leur place sur le marché des mémoires. Elles proposent d'abord un paradigme qui évite le goulot d'étranglement du modèle von Neumann, dont la rapidité des calculs est limitée par le temps d'accès à la mémoire, ce qui est une limitation fondamentale pour de nombreuses applications centrées sur un usage massif des données. Elles répondent ensuite au besoin de réduire la consommation électrique des systèmes numériques. Enfin, elles ont atteint un bon niveau de maturité technologique confirmé par divers prototypes et produits industriels. L'adoption à grande échelle du paradigme PiM dépend toutefois de la capacité à créer un écosystème dans lequel plusieurs approches pourront être testées, évaluées ou comparées. Ceci afin que ce concept soit assimilé et accepté par un ensemble d'acteurs tels que les architectes système, les ingénieurs du logiciel ou les utilisateurs finaux qui verront directement le bénéfice sur la diminution des temps de traitement. La création de cet écosystème nécessite une conception conjointe sur l'ensemble de la chaîne, depuis la technologie, le matériel, le système, le logiciel et l'environnement de programmation, jusqu'au développement des applications. Un point clé est sans aucun doute un accès aisé à leur programmation. Mais tout est encore à construire, pour compiler et paralléliser automatiquement du code tenant compte de ces mémoires. Un bon scénario serait que les modifications à apporter aux codes existants soient aussi minimes que possible, indépendantes de l'architecture matérielle et s'appuient sur des normes génériques de haut niveau. En attendant une standardisation du modèle PiM, il faut bâtir un cadre de programmation qui reflète ce paradigme, mais avec un minimum d'exposition aux détails de mise en œuvre. Cela suppose un effort logiciel et système important. Mais une architecture matérielle, aussi performante soit-elle, ne sera vraiment adoptée que si elle offre un environnement de programmation convainquant et performant.

## Références bibliographiques


• S. KRAKOWIAK – *Le modèle d'architecture von Neumann*, Interstices, 2011.

• K. ASIFUZZAMAN, N. R. MINISKAR, A. R. YOUNG, F. LIU et J. S. VETTER, *A survey on processing-in-memory techniques: Advances and challenges*, Memories- Materials, Devices, Circuits and Systems, vol. 4, 2023.

• D. FUJIKI, X. WANG, A. SUBRAMANIYAN et R. DAS – *In-/near-Memory Computing, Synthesis Lectures on Computer Architectures*, Morgan & Claypool Publishers, 2021.