UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

École Doctorale Sciences Technologie Santé

Rapport français

Pour obtenir le grade de:

Docteur de l'Université de Reims Champagne-Ardenne

Discipline : **Informatique**

Spécialité : **Calcul Haute Performance**

Présentée et soutenue par:

Julien LOISEAU

le 18 Mars 2018

Le choix des architectures hybrides, une stratégie réaliste pour atteindre l'échelle exaflopique

---

Sous la direction de :
Michaël KRAJECKI, Professeur des Universités

JURY

Pr. Michaël Krajecki	Université de Reims Champagne-Ardenne	Directeur
Dr. François Alin	Université de Reims Champagne-Ardenne	Co-directeur
Dr. Christophe Jaillet	Université de Reims Champagne-Ardenne	Co-directeur
Pr. William Jalby	Université de Versailles Saint-Quentin	Rapporteur
Dr. Christoph Junghans	Ingénieur au Los Alamos National Laboratory, USA	Rapporteur
Pr. Laurent Philippe	Université de Franche-Comté	Rapporteur
Pr. Françoise Baude	Université de Nice Sophia Antipolis	Examinateur
Guillaume Colin de Verdiere	Ingénieur au CEA	Invité

Centre de Recherche en Sciences et Technologies de l'Information et de la Communication de l'URCA EA3804
Équipe Calcul et Autonomie dans les Systèmes Hétérogènes

Table des matières traduite en français

Cette traduction de la table des matières a été réduite aux Parties, Chapitres et Sections de la thèse.

Introduction

Le monde du calcul haute performance (HPC) va prochainement atteindre une puissance de calcul inégalée avec l'échelle exaflopique. Les États-Unis d'Amérique et l'Europe devraient l'atteindre aux horizons 2020-2021, mais la Chine pourrait proposer une telle machine dès 2019. Ces superordinateurs seront 100 fois plus rapides que l'estimation actuelle des performances du cerveau humain avec 10^{16} calculs à virgule flottante par seconde (FLOPS)[Kur10] et atteindre une puissance sans précédent d'un milliard de milliard (10^{18}) de FLOPS. Cette aventure a commencé avec les premiers ordinateurs à tube à vide et les besoins de la balistique militaire. De nos jours les supercalculateurs étendent leur domaine d'application et sont des éléments phares pour représenter la puissance d'une nation. Ils sont maintenant utilisés dans tous les secteurs de la science et de la technologie.

Depuis 1962, et en considérant le Cray CDC 6600 tel que le premier superordinateur, la puissance de calcul des machines n'a fait qu'augmenter en suivant une observation faite par le co-fondateur de l'entreprise Intel, Gordon Moore. Cette loi de 1965 mieux connue sous le nom de "Loi de Moore" explique que, considérant l'évolution constante de la technologie, le nombre de transistors sur un circuit intégré va doubler environ tous les deux ans pour un prix similaire. Ceci va permettre une augmentation de la puissance de calcul que peuvent fournir les ordinateurs et superordinateurs. Plus important encore, comme "*L'argent est le nerf de la guerre*", le prix des puces pour de meilleures performances va diminuer.

Cette théorie de Gordon Moore peut être observée sur le benchmark du TOP500[1], classement mondial des supercalculateurs avec l'évolution de la puissance des supercalculateurs. Présentée sur la figure 1, la Loi de Moore se poursuit et reste vraie après des décennies d'évolution du matériel informatique.

La diminution de la taille des semi-conducteurs avec des transistors de plus en plus petits n'est pas la seule raison de cette évolution linéaire. Les premiers processeurs étaient bâtis autour d'un unique cœur de calcul (CPU), avec une augmentation du nombre de transistors et une meilleure fréquence d'opération. Ils ont vite rencontré une limitation pour atteindre des fréquences toujours plus élevées du fait de l'énergie requise, mais aussi de la chaleur générée devant être canalisée. C'est pourquoi, au début du vingtième siècle, IBM proposa le premier processeur multi-cœur, le Power4. Les constructeurs ont commencé à créer des puces avec plus d'un cœur pour augmenter la puissance de calcul tout en continuant à réduire la taille des composants. Cela a permis de répondre à la constante demande en puissance de calcul et la Loi de Moore fut conservée. Bien entendu cette nouvelle architecture demandait de nouveaux paradigmes et se confronta à plusieurs limitations : un coût supplémentaire en synchronisation entre les cœurs pour l'accès à la mémoire, le partage des tâches mais aussi une complexité des algorithmes parallèles. De nos jours un CPU classique propose de deux à des dizaines de cœurs de calcul sur un seul processeur.

Pour atteindre une puissance de calcul encore plus importante et dépasser les con-

[1] https://www.top500.org

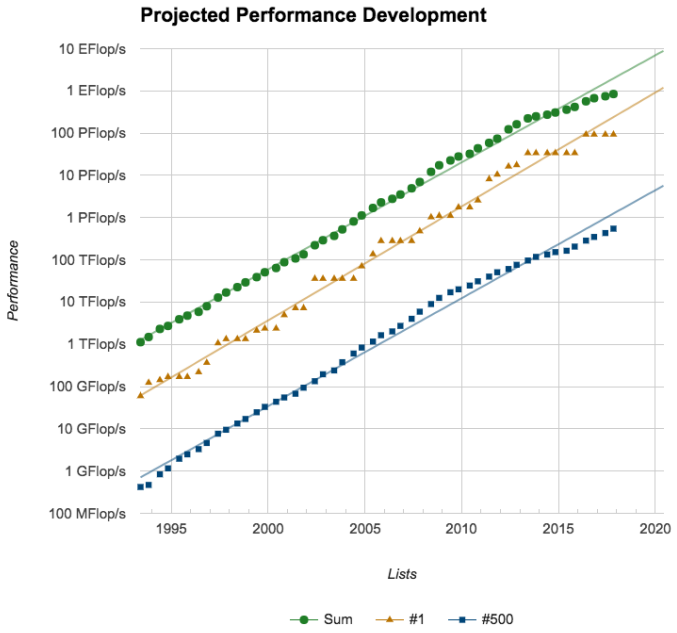


Figure 1: Évolution de la puissance de calcul des supercalculateurs, liste du TOP500

traintes énergétiques, les chercheurs se sont intéressés aux approches many-coeurs. Ces machines utilisent des centaines, voire des milliers, de cœurs de calcul "simples". Leur fréquence de fonctionnement est plus basse, ce qui réduit la consommation mais ils doivent s'exécuter de manière synchrone. Ce synchronisme d'exécution ajoute une complexité lors du développement pour pouvoir atteindre les meilleures performances. Ces architectures sont généralement couplées à un CPU qui gère l'envoi des données et les exécutions de fonctions appelées "Kernels". On notera que certaines architectures se placent entre les architectures multi-coeurs et many-coeurs proposant près de cent cœurs de calcul utilisant un réseau de processeurs multi-coeurs. Le Xeon Phi d'Intel est un parfait exemple de ce modèle. Les architectures many-coeurs sont généralement appelées accélérateurs, ils augmentent sensiblement la puissance de calcul du CPU auquel ils sont associés. Les accélérateurs les plus connus sont les *General-Purpose Graphics Processing Unit* (GPGPU), les Xeon Phi, les *Field Programmable Gate Array* (FPGA) ou encore les *Application-Specific Integrated Circuit* (ASIC). L'architecture de calcul basée sur un processeur "Host" avec un ou plusieurs "Device" apparait et est appelée architecture hybride. Un supercalculateur peut être classique, avec uniquement des processeurs multi-cœurs, hybride avec des accélérateur du même type ou encore hétérogène, avec des nœuds présentant différents types d'architectures ou d'accélérateurs.

Depuis 2013 et 2014 beaucoup d'entreprises, comme celle de Gordon Moore avec Intel, pensent que la loi de Moore n'est plus valable. Pour illustrer ce fait la figure 1 présente l'évolution de la puissance des supercalculateurs. On peut en effet voir sur la partie droite du graphique que l'évolution n'est plus linéaire mais tend à décroître dans le temps. Ceci peut s'expliquer par deux facteurs. Tout d'abord la taille minimale des transistors est peu

à peu atteinte et induit des effets de bord difficiles à gérer. De plus, l'augmentation de la surface de silicium, le grand nombre de transistors et les fréquences élevées augmentent sensiblement la consommation énergétique.

Malgré les technologies modernes, les supercalculateurs actuels sont confrontés à plusieurs limitations dans leur conception et utilisation. Les trois principales limitations à la puissance de calcul sont : la consommation énergétique, la communication et la mémoire. On trouve plusieurs sous-problématiques avec le réseau d'interconnexion, la résilience et la complexité d'utilisation de ces machines contenant des milliards de composants dédiés au calcul.

Dans cette période de questionnement par rapport aux futures architectures du calcul haute performance, cette étude propose plusieurs points de vue. Nous pensons que le futur du calcul haute performance sera bâti autour des architectures hybrides et des accélérateurs utilisant des *API* et *framework* adaptés. Nous considérons que les benchmarks classiques, tel que le fameux TOP500, ne sont pas représentatifs des problématiques actuelles et des défis du calcul haute performance. Les applications scientifiques, telles que la physique, l'astrophysique, la chimie ou encore la biologie, nécessitent des benchmarks basés sur des problèmes lourds en calcul, en communication mais surtout au comportement irrégulier qui sont à même de représenter des codes de production.

Dans cette étude nous proposons notre métrique pour extraire les principales problématiques du calcul haute performance et les appliquer aux architecture hybrides. Nous montrons comment tirer avantage de ces architectures malgré un groupe de problèmes qui ne se prête pas à leur modèle d'exécution. Notre métrique est bâtie sur deux problèmes caractéristiques puis un troisième problème, combinant tous les aspects limitants identifiés précédemment. Les deux premiers ciblent le calcul et la communication avec un comportement très irrégulier avec un grand nombre d'accès mémoire. Cette première métrique est bâtie en utilisant un problème combinatoire académique et le benchmark du Graph500. La dernière étape utilise un problème de calcul scientifique qui couvre l'ensemble des problématiques précédentes, considéré difficile à implémenter sur des architecture hybrides. Les résultats montrent le net avantage des architectures hybrides et confortent notre thèse.

Cette thèse se décompose en trois parties. La première partie explore l'état de l'art du calcul haute performance depuis les lois fondamentales jusqu'aux machines actuelles. Nous présentons les lois d'Amdahl et de Gustafson, ainsi que les notions d'accélération et d'efficacité. Les processeurs classiques, les GPGPU et autres accélérateurs sont présentés en compléments des modèles d'exécution et de mémoire. Les principales méthodes de classement des supercalculateurs et les problématiques inhérentes sont détaillées. Nous montrons que ces classements ne sont pas suffisants pour représenter les problématiques actuelles.

Dans la seconde partie nous présentons notre métrique. Le problème de Langford est décrit sous son aspect lourd en calcul et irrégulier. Ce premier élément montre l'avantage des accélérateurs, dans notre cas les GPGPU, sur ce type de problème ne

mettant pas en jeu de communications. Ce travail a mené à la publication d'un article de journal[KLAJ16b], ainsi qu'à plusieurs conférences, présentations et posters[DJK+14, LJAK16, JDK+14]. Notre implémentation de ce problème nous a permis de battre un record en termes de temps de calcul sur les dernières instances.

Le problème du Graph500 est ensuite proposé pour cibler la problématique des communications conjointement à l'irrégularité. Nous présentons notre implémentation et la logique utilisée pour obtenir des résultats sur GPGPU bien supérieurs aux architectures classiques. Ce travail a mené à la publication d'un papier de conférence[KLAJ16a] ainsi que plusieurs présentations et posters[LAJK15, LJAK15].

Dans la troisième et dernière partie nous considérons un problème confrontant à la fois la puissance de calcul et la communication avec un comportement hautement irrégulier. Nous proposons une analyse de ce problème et montrons qu'il combine tous ces aspects. Nous détaillons ensuite notre méthode pour implémenter ce problème sur les architectures modernes et en particulier hybrides. Notre choix, explicité dans l'analyse, s'est porté sur la méthode Smoothed Particle Hydrodynamics (SPH). Cette application commence avec le développement du framework FleCSI au Los Alamos National Laboratory (LANL) pour répondre à la demande des scientifiques des domaines appliqués. Le but est de mettre en place un outil efficace pour les physiciens et les astrophysiciens appelé FleCSPH. Ce travail a mené à la publication de plusieurs conférence[LLB+18, LAJK18], de présentations ainsi que de posters[DBGH+16, LLMB17].

La dernière partie résume l'ensemble du travail et les résultats obtenus. Elle montre l'avantage certain des architectures hybrides dans la course à l'échelle exaflopique sur l'ensemble des limitations abordées dans notre métrique.

1 Modèles du HPC

Cette partie présente l'état de l'art de la théorie et des technologies qui font le calcul haute performance moderne. Elle décrit les outils nécessaires à la compréhension de la problématique et de notre méthodologie scientifique. Le calcul haute performance ne trouve pas de définition stricte. Nous le définissons au travers de trois chapitres avec la théorie, les technologies et enfin les logiciels, API et framework les plus performants.

Le premier chapitre présente la machine de Von Neumann, base de toutes les machines actuelles, et des premiers processeurs mono-cœurs. Ces processeurs ont ensuite évolué vers une approche parallèle avec une augmentation de la fréquence et du nombre de cœurs de calcul. Ceci nous permet d'introduire la taxonomie de Flynn et en particulier les modèles d'exécution SIMD et SIMT qui représentent les architectures massivement parallèles et en particulier les GPGPU, utilisés dans notre étude. Les modèles de mémoires sont ensuite introduits avec le NUMA, CC-NUMA et NoRMA. Nous présentons ensuite la notion de performance avec les FLOPS, opérations à virgule flottante. Nous définissons enfin la scalabilité et l'efficacité en HPC. Cela nous améne aux lois de Amdahl et Gustafson permettant de définir le *Strong* et le *Weak scaling*.

Le second chapitre propose une présentation chronologique des technologies qui composent les supercalculateurs d'aujourd'hui. Nous présentons en détail les architectures multi-cœurs avec le calcul *out-of-order*, le *pre-fecthing*, etc. Les architectures many-coeurs sont présentées avec les GPGPU de NVIDIA et AMD, les PEZY ainsi que les FPGA. Une fois les composants de base des supercalculateurs introduits, nous présentons la topologie d'interconnexion et les principaux supercalculateurs mondiaux par rapport au classement TOP500. Nous présentons enfin le supercalculateur sur lequel l'ensemble des tests de cette étude ont été réalisés, le méso-centre ROMEO de l'Université de Reims Champagne-Ardenne.

Le dernier chapitre de cette première partie nous permet de mettre en avant les logiciels, *API* et *framework* dédiés au HPC que nous avons utilisés pour mettre en place nos implémentations multi-CPU et multi-GPU. Nous faisons un lien entre les modèles parallèles et distribués comme PRAM, *Fork-Join* et BSP et les outils de développement les plus performants.

Tous ces éléments nous permettent de présenter les benchmarks actuels des supercalculateurs. Le plus connu est le TOP500, utilisé comme classement mondial des machines et contenant les 500 plus puissants supercalculateurs publics au monde. Nous montrons que ce benchmark, et les autres, ne sont pas représentatifs des besoins et des problèmes actuels. Les problèmes de production ont en effet une irrégularité sous-jacente non représentée dans ces benchmarks. Il faut donc mettre en place une métrique permettant de déterminer le comportement des architectures dans les conditions représentatives de calcul, de communication et d'irrégularité.

2 Métrique de problème complexes

Cette seconde partie détaille notre travail méthodologique et notre métrique, adaptée aux problématiques actuelles. Nous rappelons tout d'abord les différents murs auquel le calcul haute performance est confronté et nous décidons de cibler les deux principales limitations, le calcul et la communication, tout en gardant des exemples représentatifs de l'irrégularité à la fois de calcul, de communication mais aussi de l'utilisation mémoire. Il convient de comparer sur ces problèmes les architectures classiques et hybrides à leur meilleur niveau d'optimisation.

2.1 Calcul: Problème de Langford

Pour cibler l'aspect de calcul sans communication et en environnement irrégulier, nous avons choisi le problème académique de dénombrement combinatoire de Langford. Ce problème est présenté sous deux approches, l'une généralisable à l'ensemble des problématiques de *Constraint Satisfaction Problem* (CSP) et l'autre dédiée à la résolution de ce problème.

- Une résolution arborescente, appelée méthode de Miller, nous permet de mettre en place un premier travail sur les parcours d'arbre, irréguliers par nature.
- La méthode algébrique, dite de Godfrey, nous montre l'avantage des architectures hybrides avec un comportement irrégulier induit par l'utilisation de grands entiers.

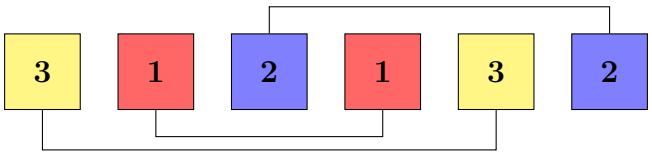


Figure 2: Exemple de solution pour l'instance L(2,3) du problème de Langford

C. Dudley Langford a donné son nom à un problème combinatoire classique[Gar56, Sim83]. En observant son fils jouer avec des cubes de couleurs différentes, il a remarqué des arrangements spécifiques de trois paires de cubes (jaune, rouge et bleu) : la paire rouge - 1 - était séparée par un cube, la paire bleue - 2 - séparée par deux cubes et enfin la paire jaune - 3 - séparée par trois cubes, tel que représenté sur la figure 2.

Ce problème a ensuite été généralisé à un ensemble de n couleurs et un nombre s de cubes de la même couleur. $L(s, n)$ consiste en la recherche du nombre de solution au problème de Langford à une symétrie près.

Méthode de Miller

La méthode de Miller propose une résolution arborescente du problème de Langford en explorant l'ensemble des cas possibles. Les branches impossibles de l'arbre sont éliminées au fur et à mesure du parcours. Avec l'explosion combinatoire le nombre de branches à explorer devient vite bien trop grand et ne permet pas de résoudre les dernières instances du problème. Néanmoins, elle fournit un parfait exemple de problème irrégulier et lourd en calcul qui ne semble pas s'adapter apriori à l'architecture SIMD des accélérateurs, du fait de l'irrégularité du parcours.

Notre implémentation propose un découpage de l'arbre de recherche par niveaux. Un premier programme génère des tâches qui sont transmisent à l'ensemble des programmes esclaves. Une fois leur calcul terminé ils transmettent le résultat et se mettent en attente d'une nouvelle tâche de travail. Nous proposons et détaillons la mise en place de cette distribution sur architectures classiques utilisant MPI-OpenMP, ainsi qu'avec l'utilisation de GPGPU avec MPI-OpenMP-CUDA. Nous exposons les choix d'implémentation pour la répartition et l'utilisation des accélérateurs. Pour la résolution des tâches nous proposons une approche régulière, explorant chaque branche et feuille, mais aussi l'approche *backtrack* classique.

Les résultats sont très bons et nous ont permis d'aborder des instances du problème de Langford au-dessus de l'instance $L(2, 19)$, encore non calculées avec cette méthode Les résultats sont présentés sur les figures 1a et 1b. Avec la méthode régularisée qui réalise donc un parcours complet de l'arbre, le GPU apporte une accélération de 1.6 fois plus rapide en comparant au *backtrack* sur CPU. Ce test a été réalisé sur 20 nœuds du supercalculateur ROMEO de l'Université de Reims Champagne-Ardenne, soit 40 CPU-GPU. Dans ce cas le GPU réalise une exploration de 200000 fois plus de nœuds mais est tout de même plus rapide que le CPU.

Dans le cas *backtrack*, les tests ont été réalisés sur 129 nœuds du supercalculateur ROMEO, un total de 258 CPU-GPU. Malgré cette méthode, qui ne semble pourtant

n	CPU (8c)	GPU (4c) + CPU (4c)
15	2.5	1.5
16	21.2	14.3
17	200.3	120.5
18	1971.0	1178.2
19	22594.2	13960.8

(a) Regularized method (seconds)

n	CPU (8c)	GPU (4c) + CPU (4c)
17	29.8	7.3
18	290.0	73.6
19	3197.5	803.5
20	–	9436.9
21	–	118512.4

(b) Backtrack (seconds)

Table 1: Comparaison du temps de calcul CPU et GPU avec la méthode régularisée et la méthode *backtrack*

pas s'adapter à l'architecture synchrone du modèle SIMD du GPU, nous observons une accélération de 15 fois plus. Nous avons repoussé les limites de la méthode en résolvant les instances $L(2, 20)$ en moins de trois heures et $L(2, 21)$ en 33 heures.

Méthode Godfrey

Nous avons ensuite travaillé sur la méthode la plus rapide, actuellement connue pour dénombrer les arrangements du problème de Langford. La méthode algébrique de Godfrey propose une sommation en explorant un sous-ensemble des combinaisons possibles du problème avec :

$$\sum_{(x_1,\ldots,x_{2n})\in\{-1,1\}^{2n}} \left(\prod_{i=1}^{2n} x_i\right) \prod_{i=1}^{n} \sum_{k=1}^{2n-i-1} x_k x_{k+i+1} = 2^{2n+1} L(2,n) \tag{1}$$

Plusieurs optimisations sont possibles et permettent de réduire le nombre de calculs à réaliser, rendant cette méthode bien plus intéressante que l'approche classique arborescente. Nous donnons le détail de ces optimisations ainsi que notre implémentation de la méthode. L'idée reste la même que pour la méthode de Miller, les tâches sont cette fois-ci basées sur le positionnement de paires représentées au format binaire. Chaque CPU ou GPU travaille sur une tâche spécifique et les résultats sont ensuite rassemblés. L'irrégularité avec cette résolution vient de l'utilisation de grands entiers, nécessitant une représentation sur plusieurs mots mémoires et des propagations de retenues qui ne peuvent être anticipées. La thèse présente le détail de nos choix et implémentation de cette méthode.

Les résultats sont très bons et montrent une nouvelle fois l'avantage des architectures hybrides confrontées au calcul irrégulier, sans communications. Cette implémentation nous a permis de recalculer les dernières itérations du problème, $L(2, 27)$ et $L(2, 28)$ dans un temps record, avec respectivement 1.9 jours et 23 jours avec une stratégie de répartition Best-Effort sur le supercalculateur ROMEO (environs 70% du cluster). Cette stratégie nous permet de bénéficier des nœuds de calcul non utilisés par les chercheurs mais nécessite des redémarrages et interruptions de tâches. Les résultats avec une répartition classique de l'ensemble du cluster auraient été encore meilleurs.

Ce premier élément de notre métrique, confrontant les architectures au problème de calcul dans un environnement irrégulier, montre nettement l'avantage des accélérateurs. Ils permettent une montée à l'échelle dans les deux méthodes abordées et s'adaptent même à des problématiques qui ne semblent pas nativement adaptées à leur architecture.

2.2 Communication: Graph500

La deuxième partie de notre métrique cible la problématique de communication en conservant le comportement irrégulier. Nous avons choisi le benchmark du Graph500[2]. Ce problème de parcours en largeur d'un très grand graphe est utilisé comme base pour le classement des supercalculateurs avec une métrique spécifique, le nombre d'arêtes du graphe traversées par seconde, les TEPS.

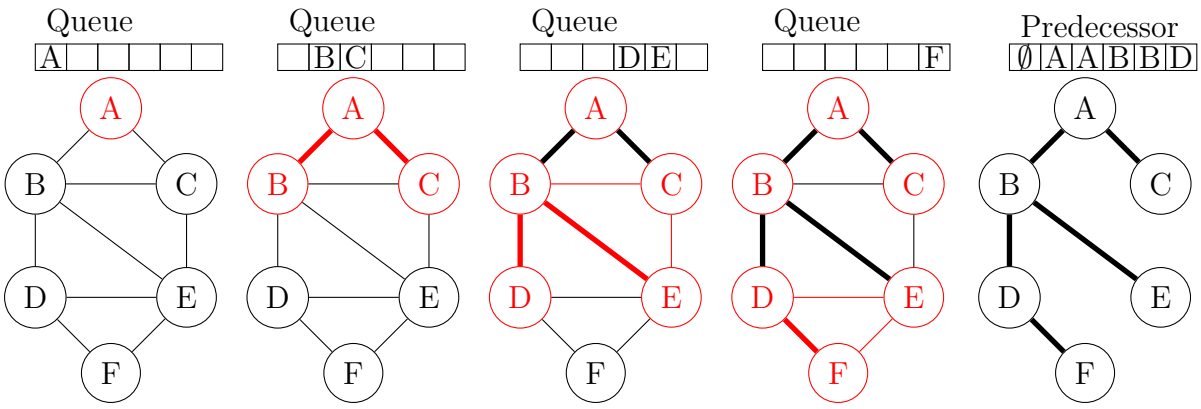


Figure 3: Exemple de parcours en largeur sur un graphe non-orienté

L'algorithme d'un parcours en largeur sur un graphe est détaillé sur la figure 3. Il comprend beaucoup de phases de communications pour échanger les frontières et réaliser des réductions entre les processus. La parcours du voisinage de chaque sommet ne comprend pas de calcul mais uniquement des accès mémoire irréguliers et des tests avec copie.

Nous proposons une implémentation multi-CPU et multi-GPU de ce benchmark. Notre travail se base sur les travaux de NVIDIA avec Merill [MGG15], ainsi que le meilleur algorithme distribué avec le code utilisé sur les architectures BlueGene/P et BlueGene/Q [CPW+12] de IBM.

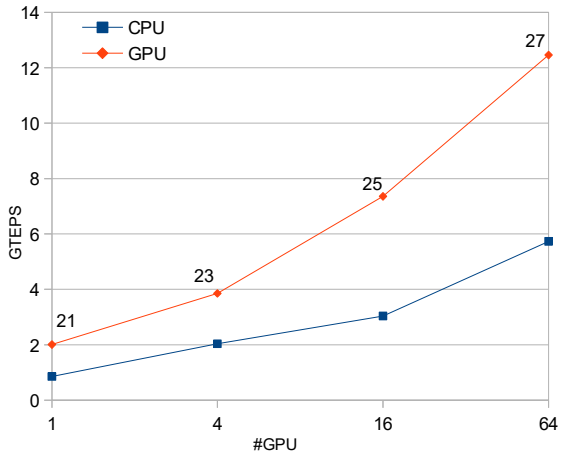
Notre approche propose une représentation compressée de la matrice d'adjacence appelée *Compressed Sparse Row* et *Compressed Sparse Column* en complément d'une représentation binaire des listes de frontière et d'exploration.

Les communications de la version GPU se font directement dans la mémoire de ceux-ci en utilisant la technologie GPU-Direct de MPI et en évitant les transferts temporaires sur la mémoire *Host*.

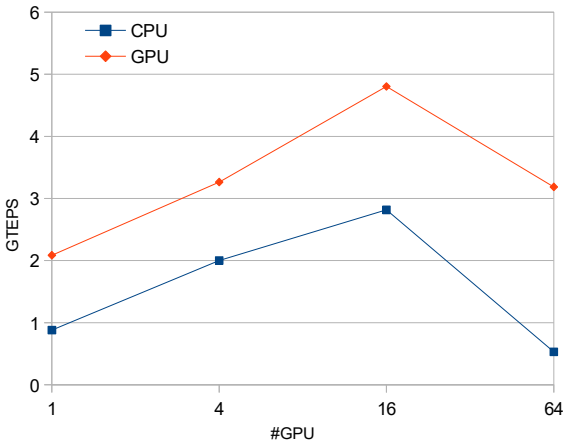
Notre implémentation montre une nouvelle fois l'avantage des architectures hybrides face aux architectures classiques.

Les résultats présentés en figure 4a et figure 4b montrent respectivement le *weak scaling* et le *strong scaling* de nos deux versions. La version CPU se base sur le meilleur

[2]https://www.graph500.org



(a) CPU et GPU *weak scaling*. Le nombre de CPU est le même que le nombre de GPU. Le *SCALE* est présenté sur la courbe GPU



(b) CPU et GPU *strong scaling*. Le nombre de CPU est le même que le nombre de GPU

Figure 4: Weak and Strong scaling between CPU and GPU

algorithme fournis par l'organisation Graph500 et optimisé pour notre architecture de supercalculateur.

Nous arrivons à obtenir une accélération deux fois supérieure en temps de calcul avec l'utilisation de GPGPUs sur l'ensemble du cluster ROMEO. Nous avons aussi réalisé des tests sur différents types d'architectures GPU, tels que des cartes graphiques grand public, la GTX980Ti, et des GPU embarqués comme le NVIDIA TX1. Ce travail montre une nouvelle fois l'efficacité des accélérateurs sur une problématique uniquement basée sur la communication et le parcours aléatoire de la mémoire. Nous avons été à même de classer le supercalculateur ROMEO, datant de 2014, à la 105ème place au classement Graph500 dans la liste de novembre 2016.

Cette section, avec les deux cas spécifiques que nous avons choisi pour cibler différentes problématiques du calcul haute performance, montrent clairement l'avantage des architectures hybrides. Cela est mis en avant avec respectivement un contexte lourd en calcul, lourd en communication et avec dans chaque cas une utilisation irrégulière de la mémoire et des calculs. La scalabilité de ces accélérateurs est aussi mise en avant avec des technologies de communication avancées, comme GPU-Direct ou encore l'arrêt et le redémarrage de tâches avec l'utilisation Best-Effort sur le cluster ROMEO.

3 Application

Dans ce dernier chapitre nous présentons une application mettant en jeu tous les aspects présentés précédemment : calculs lourds, communications coûteuses et irrégularités. Cette application se devait d'être représentative des problématiques réelles pour valider notre thèse. Après deux visites et de nombreux échanges avec les physiciens et astrophysiciens du laboratoire national de Los Alamos (LANL) aux États-Unis, nous avons choisi un problème de simulation adapté aux fluides et aux évènements astrophysiques. Nous

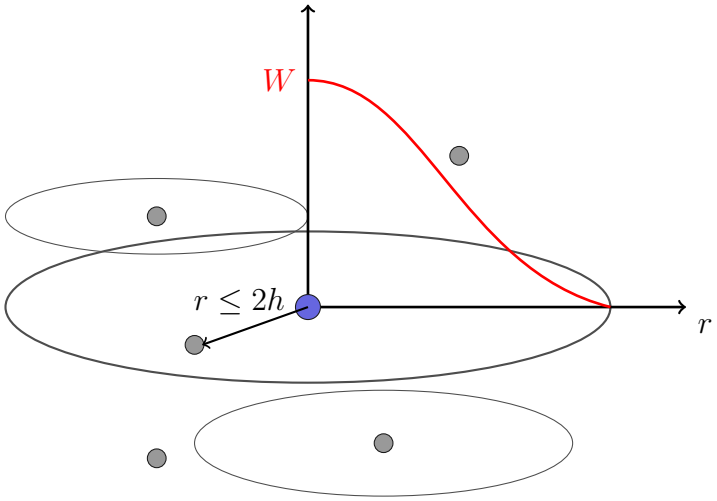


Figure 5: SPH kernel W and smoothing length h representation

présentons la méthode *Smoothed Particle Hydrodynamics* (SPH), complémentée avec la méthode *Fast Multipole Method* (FMM) pour le calcul des interactions gravitationnelles.

La méthode Lagrangienne SPH est résumée sur la figure 5. Le calcul des grandeurs physiques d'une particule se fait par l'intermédiaire des contributions de ses particules voisines à l'intérieur d'une surface de lissage h et par l'action d'une fonction de lissage W. Cette méthode se prête parfaitement à une représentation sous forme arborescente des particules avec des outils de décomposition de domaine, tels que les courbes de Morton ou Hilbert. Ceci permet une complexité de l'ordre de $O(Nlog(N))$ au lieu de $O(N^2)$ pour les calculs N-body et la recherche de voisinage. Elle fournit, de plus, à notre étude un parcours irrégulier d'un arbre.

Pour envisager des calculs avec un très grand nombre de particules sur des évènements astrophysiques majeurs nous avons besoin d'une manière rapide de calculer les interactions gravitationelles.

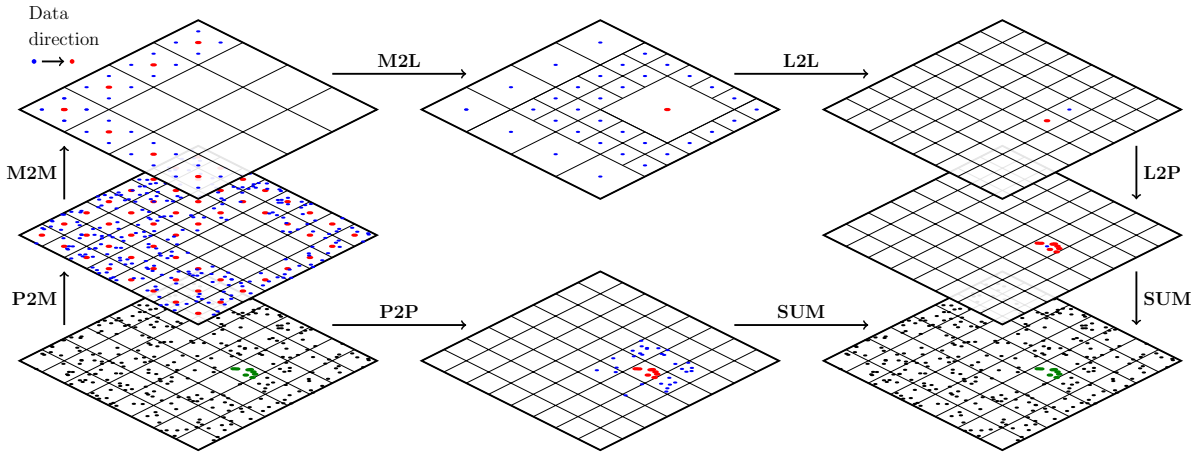


Figure 6: Schéma de la Fast Multipole Method. Particles to Multipole (P2M), Multipole to Multipole (M2M), Multipole to Particles (M2P), Multipole to Local (M2L), Local to Local (L2L) et Particles to Particles (P2P). Schéma inspiré de [YB11]

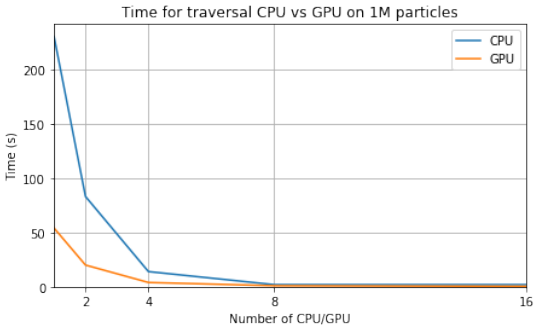


Figure 7: CPU vs GPU time per iteration

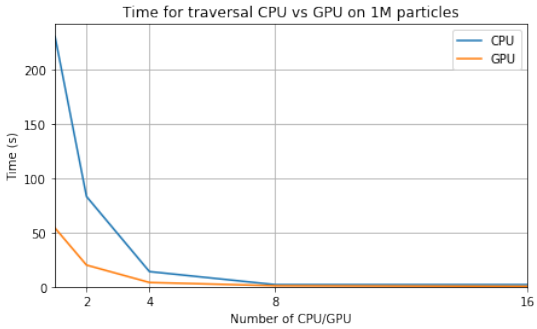


Figure 8: CPU vs GPU time per traversal

Nous avons pour cela mis en place la *Fast Multipole Method* (FMM) détaillée sur la figure 6. Cette méthode se décompose en une succession de sous-étapes avec une approximation selon la distance des particules concernées et des centres de masses. Cette méthode nativement séquentielle a été adaptée en mode distribué pour notre application.

Cette partie a nécessité un lourd travail de compréhension des phénomènes et des équations physiques et astrophysiques nécessaires pour l'implémentation Ex Nihilo de ces simulations.

Notre implémentation se base sur le *framework* FleCSI développé au LANL qui fournit un ensemble d'outils pour les simulations multi-physiques. Il fournit différentes topologies et va gérer pour l'utilisateur la répartition des données, des tâches et du calcul. Il propose déjà des topologies de *mesh*, graphes, et structures arborescentes. Notre programme, FleCSPH, propose de mettre en place la stratégie de distribution des représentations arborescentes. Dans un second temps, nous avons mis en place une version GPU de ce programme. La répartition du travail entre le CPU et les GPU se fait localement. Une liste d'interaction par groupe de particules est créée et déléguée au GPU pour un calcul en $O(N^2)$ sur un petit nombre de particules.

Les résultats de la version hybride, comparée à la version CPU, sont très encourageants et sont juste 3,5 fois plus rapide sur un million de particules pour une fusion d'étoiles à neutrons. Ils sont présentés sur la figure 7 et 8 et avec respectivement le temps global d'une itération de calcul et du calcul du parcours de l'arbre pour la recherche de voisinage.

Notre code propose une topologie d'arbre pour une, deux et trois dimensions qui se valide par les résultats de nos modèles physiques. Nous avons réalisé des simulations sur les cas génériques d'hydrodynamique avec le *shock Sod tube* et le *Sedov blast wave*. Nous avons ensuite testé notre implémentation sur un grand nombre de particules, pour des simulations de fluides utilisant des conditions limites propre à la méthode SPH. Enfin, pour rejoindre notre objectif principal, nous avons simulé la fusion d'étoiles à neutrons.

Les résultat des simulations sont présentés pour un fluide et une fusion d'étoile à neutrons respectivement sur les figures 9 et 10.

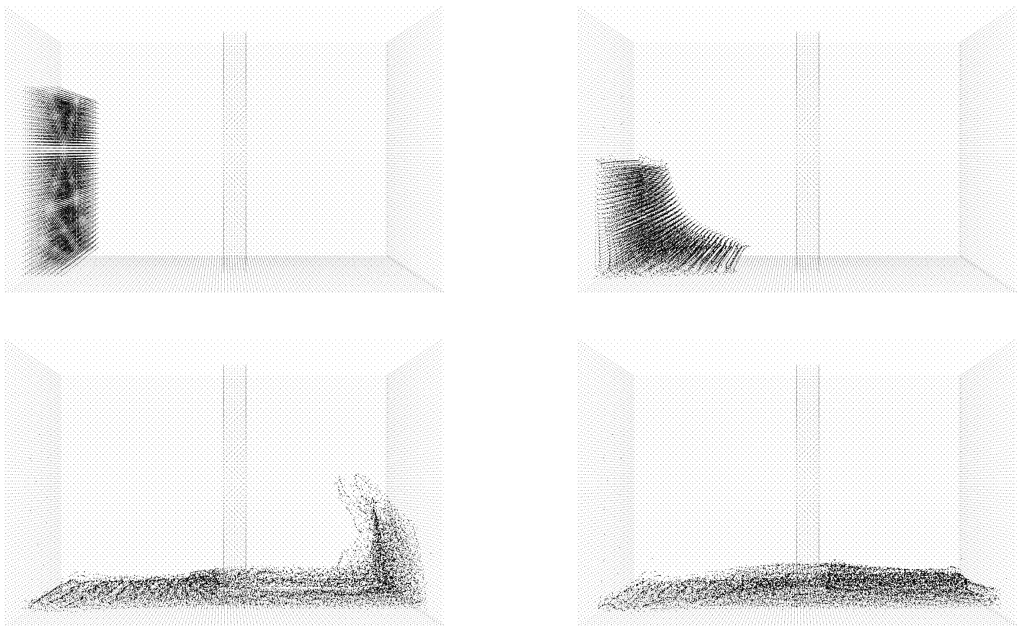


Figure 9: Écoulement de fluide, le barrage. Pour $t = 0$, $t = 0,4$, $t = 0,8$ et $t = 1$ secondes

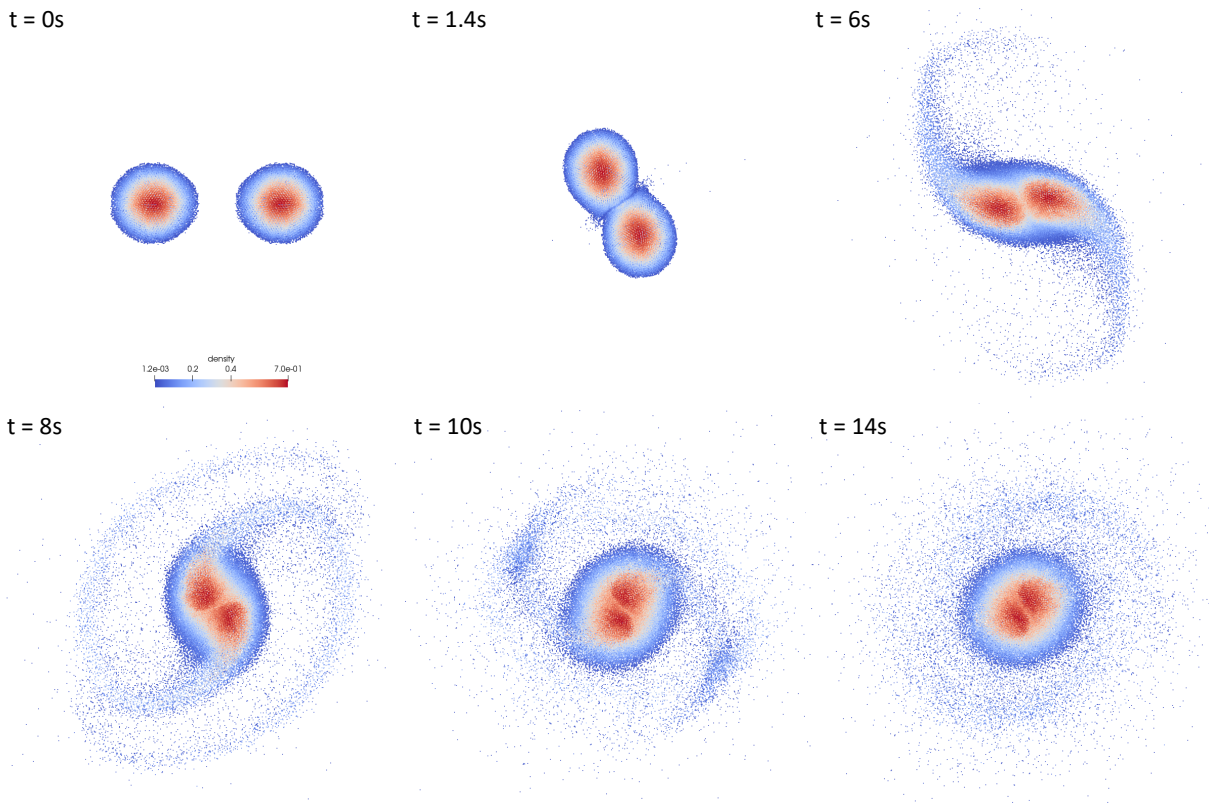


Figure 10: Coalescence d'étoiles à neutrons avec 40.000 particules.

Conclusion

La première partie de cette étude décrit les outils et la théorie nécessaire pour comprendre et atteindre la performance dans le calcul haute performance. Nous avons présenté plusieurs architectures et leurs avantages en présentant l'objectif d'un supercalculateur à échelle exaflopique aux environs 2020. Nous pensons que la solution actuelle réside dans l'utilisation d'architectures hybrides munies de machines many-coeurs tels que les GPGPU ou les FPGA.

Nous avons défini une métrique ciblant les principales limitations du calcul haute performance avec le calcul, la communication et le comportement irrégulier. L'objectif de cette métrique est de confronter les architectures classiques et hybrides à des problèmes représentatifs des problématiques actuelles.

Cette métrique est séparée en deux parties. Dans un premier temps nous avons visé des problèmes classiques, académiques et même un benchmark. Nous avons comparé les architectures classiques aux hybrides sur un premier problème lourd en calcul et un second lourd en communications. Dans les deux cas nous voulions retrouver une spécificité des codes appliqués : l'irrégularité de calcul, communication et accès mémoire.

Le premier problème choisit est le problème de dénombrement combinatoire de Langford. Nous avons étudié ce problème avec deux méthodes différentes, respectivement un parcours arborescent et une méthode algébrique. La résolution arborescente montre de très bons résultats sur accélérateurs, avec jusque 80 pourcents du travail réalisé par les GPGPU avec une stratégie de répartition des tâches efficace. La solution algébrique propose des calculs régularisés mais des accès mémoires aléatoires, irréguliers, par l'utilisation de grands entiers. Dans ce cas le GPU est capable de gérer jusque 65 pourcents de l'effort global de calcul. Notre version basée sur les architectures hybrides nous a permis de battre un record de temps pour le calcul des dernières instances, en utilisant le supercalculateur ROMEO de l'Université de Reims Champagne-Ardenne en mode Best-Effort.

Le second problème pour notre métrique est le benchmark du Graph500. C'est un candidat parfait pour considérer uniquement les communications, sans calculs. En effet, les seules opérations nécessaires sont des tests et des copies de zone de mémoire qui se réalisent de manière irrégulière. Ce problème respecte aussi la problématique d'irrégularité. Nous proposons une implémentation basée sur les derniers travaux à la fois de NVIDIA, mais aussi BlueGene/Q et BlueGene/P. Ce travail nous a permis de classer le supercalculateur ROMEO à la 105ème place du Graph500 dans la liste de novembre 2016. Cela montre le haut niveau de parallélisme et de scalabilité qui peut être obtenu avec les architectures hybrides.

Ces deux premières approches montrent un grand avantage aux architectures hybrides malgré des problèmes qui semblent inadaptés au modèle d'exécution SIMD.

Pour compléter notre métrique, nous avions besoin de confronter les deux types d'architectures à un problème présentant à la fois les problématiques de calcul et de communication avec un comportement irrégulier. Nous avons choisi une application de simulation complexe. Ce problème répondant aux problématiques est la méthode Smoothed Particle Hydrodynamics (SPH), ainsi que le calcul de la gravitation appliqué à des simulations d'astrophysique. Nous avons développé, en partenariat avec le Los Alamos National Laboratory, un framework appelé FleCSPH et dédié aux topologies arborescentes utilisées pour les simulations de physique et d'astrophysique. Nous montrons avec cette dernière métrique les avantages des architectures hybrides. Même en conservant une approche de code de production, pas entièrement porté sur GPU, les performances restent bien supérieures que sur les architectures classiques.

Cette étude a montré que, dans tous les cas étudiés, les architectures hybrides sont la meilleure solution actuelle pour atteindre l'échelle exaflopique. Les limitations sont résolues en utilisant des stratégies qui peuvent être généralisées à d'autres problèmes et de nouvelles méthodes de résolution doivent encore être trouvées.

Les architectures hybrides semblent être la meilleure approche pour bâtir des machines exaflopiques mais l'Informatique est une science évoluant très rapidement. L'arrivée d'une nouvelle technologie pourrait changer la donne et permettre un meilleur rapport puissance de calcul et consommation énergétique. La principale autre solution réside pour le moment dans l'utilisation des architectures ARM. Le projet MontBlanc, que nous avons présenté en partie I, est le projet Européen de supercalculateur implémentant ces processeurs à jeu d'instruction réduit. Le projet Japonais avec RIKEN va proposer le post-K[3], successeur du K Computer. Il sera bâti sur des processeurs ARMv8 et utilisera la même topologie que son prédécesseur avec une topologie réseau de tore à six dimensions.

Une autre solution pour des problèmes spécifiques semble émerger avec l'Informatique Quantique. Les *bits* des processeurs classiques sont remplacés par des *qbits*, quantums bits, qui permettent de représenter plus de deux valeurs simultanément. Ces machines sont basées sur des probabilités et ne sont utilisables que sur une sous-catégorie de problèmes. Cette technologie est toujours en développement mais dans certains centres de calcul, comme ROMEO à l'Université de Reims Champagne-Ardenne, il est d'ores et déjà possible de travailler sur simulateur pour en comprendre le comportement.

Cette étude s'est intéressée aux moyens d'atteindre la puissance nécessaire à exaflopique. Les principales limitations sont la puissance de calcul, la communication et la demande énergétique que nous avons déjà abordés dans cette thèse. Mais une fois l'exaflop atteint, d'autres problèmes vont survenir. Nous avons cité le problème du réseau d'interconnexion, qui va augmenter lorsque les supercalculateurs vont contenir des mil-

[3]http://www.fujitsu.com/global/Images/post-k-supercomputer-overview.pdf

lions de composants, rendant difficile la conservation d'une bande passante suffisante entre eux. Une autre problématique est l'augmentation des erreurs de calculs due aux influences environnementales. Elles sont de l'ordre d'une erreur par semaine sur un calculateur petaflopique mais vont atteindre de l'ordre d'une par heure pour des machines exaflopiques. Il faudra alors inventer de nouvelles méthodes de vérification à la volée. De la même manière les algorithmes devront être résistants aux défaillances matérielles. La problématique de résilience, le rétablissement après la perte d'un élément de calcul, nécessitera la mise en place d'algorithmes robustes et de parallélisme de tâches.

La dernière problématique sera la complexité. Les Informaticiens ont besoin d'inventer de nouvelles manières de penser les algorithmes parallèles. D'une part la mise en place de nouveaux *framework* et *API* pour gérer et cibler tous les types de topologies et d'architectures. D'autre part la production de nouvelles méthodes non héritées du calcul séquentiel pour repenser les algorithmes et les calculs.

Bibliographie

[CPW+12] F. Checconi, F. Petrini, J. Willcock, A. Lumsdaine, A. R. Choudhury, and Y. Sabharwal. Breaking the speed and scalability barriers for graph exploration on distributed-memory machines. In *High Performance Computing, Networking, Storage and Analysis (SC), 2012 International Conference for*, pages 1–12, Nov 2012.

[DBGH+16] Nicola De Brye, Daniel George, Glenn Hordemann, Hyun Lim, Julien Loiseau, Jonah Miller, and Johnathan Sharman. Domain partitioning and problem space representations for compact binary mergers. In *Poster at the 2016 Co-Design Summer School, Los Alamos.*, 2016.

[DJK+14] Hervé Deleau, Christophe Jaillet, Michaël Krajecki, Julien Loiseau, Luiz Angelo Steffenel, François Alin, and Lycée Franklin Roosvelt. Towards the parallel resolution of the langford problem on a cluster of gpu devices. In *CSC14: The Sixth SIAM Workshop on Combinatorial Scientific Computing*, page 66, 2014.

[Gar56] Martin Gardner. *Mathematics, magic and mystery.* Dover publication, 1956.

[JDK+14] Christophe Jaillet, Hervé Deleau, Michaël Krajecki, Luiz-Angelo Steffenel, François Alin, and Julien Loiseau. Langford problem: Massively parallel resolution on a multigpu cluster. In *CSC14: The Sixth SIAM Workshop on Combinatorial Scientific Computing*, 2014.

[KLAJ16a] Michaël Krajecki, Julien Loiseau, François Alin, and Christophe Jaillet. Bfs traversal on multi-gpu cluster. In *2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES)*, 2016.

[KLAJ16b] Michaël Krajecki, Julien Loiseau, François Alin, and Christophe Jaillet. Many-core approaches to combinatorial problems: case of the langford problem. *Supercomputing Frontiers and Innovations*, 3(2):21–37, 2016.

[Kur10] Ray Kurzweil. *The singularity is near.* Gerald Duckworth & Co, 2010.

[LAJK15] Julien Loiseau, François Alin, Christophe Jaillet, and Michaël Krajecki. Parcours de grands graphes sur architecture hybride cpu/gpu. In *Modèles et*

Analyses Réseau: Approches Mathématiques et Informatiques (MARAMI) 2015. 14-16 Octobre Nîmes. Journée Fouille de Grands Graphes (JFGG), 2015.

[LAJK18] Julien Loiseau, François Alin, Christophe Jaillet, and Michaël Krajecki. Flecsphg: A gpu accelerated framework for physics and astrophysics simulations. In *Latin America High Performance Computing Conference (CARLA 2018)*, jul 2018. Proceedings.

[LJAK15] Julien Loiseau, Christophe Jaillet, François Alin, and Michaël Krajecki. Massively parallel resolution of combinatorial problems on multigpu clusters. In *GTC Technology Conference 2015.*, 2015.

[LJAK16] Julien Loiseau, Christophe Jaillet, François Alin, and Michaël Krajecki. Résolution du problème de langford sur architecture massivement parallèle cpu/gpu. In *Compas' Conference 2016. Lorient, June 2016, France*, 2016.

[LLB+18] Julien Loiseau, Hyun Lim, Ben Bergen, Nicholas Moss, and François Alin. Flecsph: a parallel and distributed smoothed particle hydrodynamics framework based on flecsi. In IEEE, editor, *The 2018 International Conference on High Performance Computing & Simulation (HPCS 2018)*, jul 2018. Accepted.

[LLMB17] Julien Loiseau, Hyun Lim, Nick Moss, and Ben Bergen. A parallel and distributed smoothed particle hydrodynamics implementation based on the flecsu framework. In *SuperComputing conference, Denver CO*, 2017.

[MGG15] Duane Merrill, Michael Garland, and Andrew Grimshaw. High-performance and scalable gpu graph traversal. *ACM Transactions on Parallel Computing*, 1(2):14, 2015.

[Sim83] James E. Simpson. Langford sequences: perfect and hooked. *Discrete Math*, 44(1):97–104, 1983.

[YB11] Rio Yokota and Lorena A Barba. Treecode and fast multipole method for n-body simulation with cuda. In *GPU Computing Gems Emerald Edition*, pages 113–132. Elsevier, 2011.