Travaux Pratique 2

Contexte : Les données indiquées dans ce rapport ont été obtenues avec un Processeur 6 cœurs et 12 processeurs logiques.

Exercice 1 :

En entrée on utilise un tableau de float. Celui-ci est alors converti (2 fois) en __m256 permettant d'effectuer des opérations vectorielles. On réalise l'addition (et la racine carré) en une seule instruction, qui est appliqué aux 8 réels stockés dans le __m256. Ensuite, avec la structure du Convertor, on peut récupérer les résultats sans conversion explicite, grâce à l'union.

Exercice 2 :

Taille	Tseq (µs)	Tpar (µs)	Accélération	Efficacité
4096	48	6	8	1
40960	411	55	7,47272727	0,93409091
409600	4243	1567	2,70772176	0,33846522
4096000	42695	16412	2,60145016	0,32518127

On voit assez clairement que les instructions AVX sont bien plus rapides mais que l'efficacité décroit de manière inversement proportionnelle à la taille des données. On a une efficacité proche de 1 pour un nombre d'AVX assez bas mais celle-ci décroit rapidement.

Exercice 3 :

Comparaison des algorithmes séquentiels

Avec 4 threads

Taille	Tseq	Tpar	Accélération	Efficacité
4096	48	171	0,28070175	0,07017544
40960	411	275	1,49454545	0,37363636
409600	4243	1354	3,13367799	0,7834195
4096000	42695	14527	2,93901012	0,73475253

Avec 12 threads

Taille	Tseq	Tpar	Accélération	Efficacité
4096	48	446	0,107623318	0,00896861
40960	411	447	0,919463087	0,07662192
409600	4243	1022	4,151663405	0,34597195
4096000	42695	14827	2,879544075	0,23996201

Pour ces 2 tableaux, Tseq correspond aux temps séquentiels obtenus dans l'exercice 2 et Tpar correspond aux temps relevés avec des instructions séquentielles pour un pool de threads.

On remarque que l'efficacité est très faible au début mais croit avec l'augmentation du nombre d'AVX à traiter. On peut mettre cela sur le coût d'instanciation et de lancement des threads qui n'est amorti que lorsqu'ils travaillent suffisamment longtemps. De plus, on voit clairement que l'algorithme n'est pas forcément plus efficace avec plus de threads, ce qui conforte l'hypothèse précédente.

Comparaison des algorithmes vectoriels

Avec 4 threads

Taille	Tseq	Tpar	Accélération	Efficacité
4096	6	162	0,03703704	0,00925926
40960	55	165	0,33333333	0,08333333
409600	1567	608	2,57730263	0,64432566
4096000	16412	14208	1,15512387	0,28878097

Avec 12 threads

Taille	Tseq	Tpar	Accélération	Efficacité
4096	6	414	0,014492754	0,00120773
40960	55	452	0,121681416	0,01014012
409600	1567	644	2,433229814	0,20276915
4096000	16412	14821	1,107347682	0,09227897

Pour ces 2 tableaux, Tseq correspond aux temps vectoriels obtenus dans l'exercice 2 et Tpar correspond aux temps relevés avec des instructions vectorielles pour un pool de threads.

En vectoriel, on observe des résultats similaires mais on constate que l'efficacité diminue lorsqu'on prend un nombre important d'AVX.

Comparaison des algorithmes avec threads

Avec 4 threads

Taille	Tseq	Tpar	Accélération	Efficacité
4096	171	162	1,05555556	0,13194444
40960	275	165	1,66666667	0,20833333
409600	1354	608	2,22697368	0,27837171
4096000	14527	14208	1,02245214	0,12780652

Avec 12 threads

Taille	Tseq	Tpar	Accélération	Efficacité
4096	446	414	1,077294686	0,13466184
40960	447	452	0,988938053	0,12361726
409600	1022	644	1,586956522	0,19836957
4096000	14827	14821	1,000404831	0,1250506

Pour ces 2 tableaux, Tseq correspond aux temps relevés avec des instructions séquentielles pour un pool de threads et Tpar correspond aux temps relevés avec des instructions vectorielles pour un pool de threads.

Tout d'abord, on constate que l'utilisation des AVXs n'est pas très efficace par rapport au pool de threads séquentiels. De plus, on note une baisse d'efficacité en augmentant le nombre de threads.

On peut conclure que l'utilisation des threads n'est pas un plus lorsqu'on utilise des instructions AVX.

Exercice 4 :

Taille	Tseq	Tpar	Accélération	Efficacité
4096	83	19	4,368421053	0,54605263
40960	826	199	4,150753769	0,51884422
409600	8331	1963	4,244014264	0,53050178
4096000	83809	20079	4,173962847	0,52174536

L'efficacité de l'algorithme parallèle est assez stable, aux alentours de 0,5.

On observe que le calcul séquentiel est faux dès qu'on a un grand nombre de données (Au-dessus de 2097152 AVX de mon côté). Je suppose que c'est dû à la représentation imprécise des réels qui génèrent des approximations et donc des erreurs de calculs.

Exercice 5 :

Taille	Tseq (µs)	Tpar (µs)	Accélération	Efficacité
16	42	10	4,2	0,525
160	4199	995	4,220100503	0,52751256
1600	423000	103000	4,106796117	0,51334951

L'accélération et l'efficacité de l'algorithme parallèle sont assez stables, aux alentours de de 4 et 0,5.