COMP52315 Coursework
Fast Finite Elements

CIS Username: bgtt65

1 Introduction

Two bake-off problems proposed by the Center for Efficient Exascale Discretizations (CEED) are described in this report. With the experimental data from a series of benchmarking in four different backends and MPI, the performance of these implementations is also compared and discussed.

2 Bake-off Problems

The finite element method is a numerical technique used to solve partial differential equations, which can be applied in various industries. To solve the finite element problem efficiently, CEED provides an interface for matrix-free operator representation called libCEED. The sequence of operations is shown in Fig. 1 (simplified from CEED website). The computation starts from the degrees of freedom (DOFs) on the global domain, restricts to the DOFs on subdomains, then goes to the DOFs on elements. Next, it transits to quadrature points and completes pointwise integration. After that, the computation moves back from quadrature points to DOFs on the global domain.

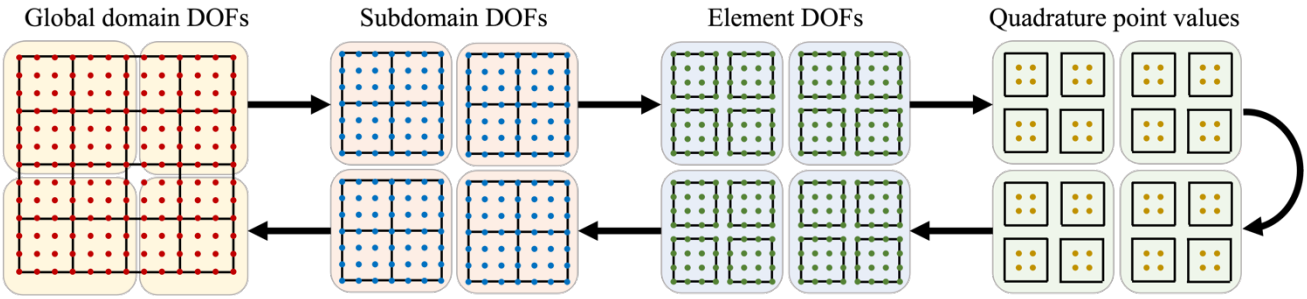

Figure 1: The operations in libCEED

The libCEED library provides several bake-off problems to test and compare the performance. In this study, the bake-off problems BP1 and BP2 are chosen to perform the benchmarking and the performance comparison. Both BP1 and BP2 are set to solve the finite element problem $Au = f$ (where A is the mass matrix) using a diagonally preconditioned conjugate gradient (PCG) method. The difference between them is that BP1 applies scalar PCG, whereas BP2 utilise vector PCG.

3 Benchmark of Backends

To assess the benchmark, the DOFs/sec in conjugate gradient (CG) of four backends that are serial reference, blocked reference, serial AVX and blocked AVX were measured on the Hamilton par.7 partition. In this study, two benchmarks are set based on the polynomial degree and the problem size.

For the first benchmark, three different implementations ran the test on a range of polynomial degrees, from 1 to 16 degrees. And the problem size 10^3, 10^4, 10^5 were chosen to compare this benchmark in the small, medium and large size. In addition, for the second benchmark, the DOFs/sec in CG of three implementations were recorded on a range of problem size. Three intervals of problem size were selected to present the different properties. Small problem sizes in the first interval are $2^0 \cdot 10^3$, $2^1 \cdot 10^3$, $2^2 \cdot 10^3$ and $2^3 \cdot 10^3$; medium problem sizes in the second interval are $2^0 \cdot 10^4$, $2^1 \cdot 10^4$, $2^2 \cdot 10^4$ and $2^3 \cdot 10^4$; and large problem sizes in the third interval are $2^0 \cdot 10^5$, $2^1 \cdot 10^5$, $2^2 \cdot 10^5$ and $5 \cdot 10^5$. Then, compare this benchmark in the small, medium and large number of polynomial degrees (including 2^0, 2^2, 2^4).

3.1 BP1 Results

The two benchmarks for BP1 are shown in Fig. 2, 3 and Table 1. In Fig. 2, the DOFs/sec of serial AVX performs well throughout three different problem sizes, sharing the same trend that there is a significant increase from polynomial degree 1 to 7 and then a few small fluctuations till the end. On the other hand, the performance of serial reference is bad in that the values are quite low in the three chosen cases. For the blocked backends, a unique behaviour is observed that the DOFs/sec goes up in the beginning and reaches a peak, then remains steady before a noticeably drop at the problem size 10^3 and 10^4. At problem size 10^5, blocked backends show a similar trend without a dramatic fall. Overall, the performance of blocked AVX is not as well as serial AVX at small and medium problem sizes, but it is efficiently improved at the large problem size, especially for low polynomial degrees.

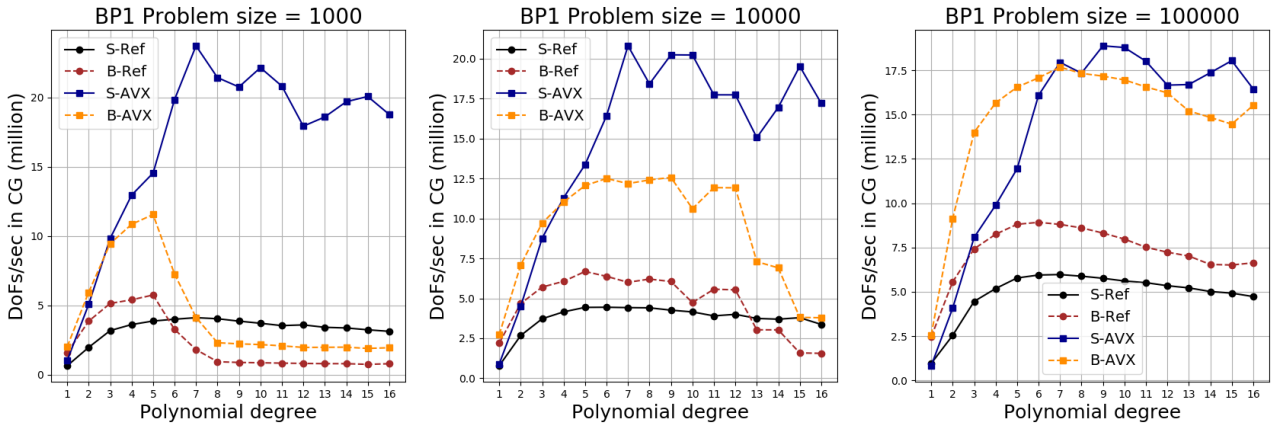

Figure 2: BP1 results, Polynomial degree vs. DOFs/sec in CG at fixed problem size

As for Fig. 3, both blocked backends show good performance at polynomial degree 1. However, as the polynomial degree rises, the serial AVX performs better and exceeds the blocked AVX in all three intervals of problem size. Furthermore, an interesting behaviour of blocked backends is found that a growth in the second interval becomes clear when the polynomial degree increases.

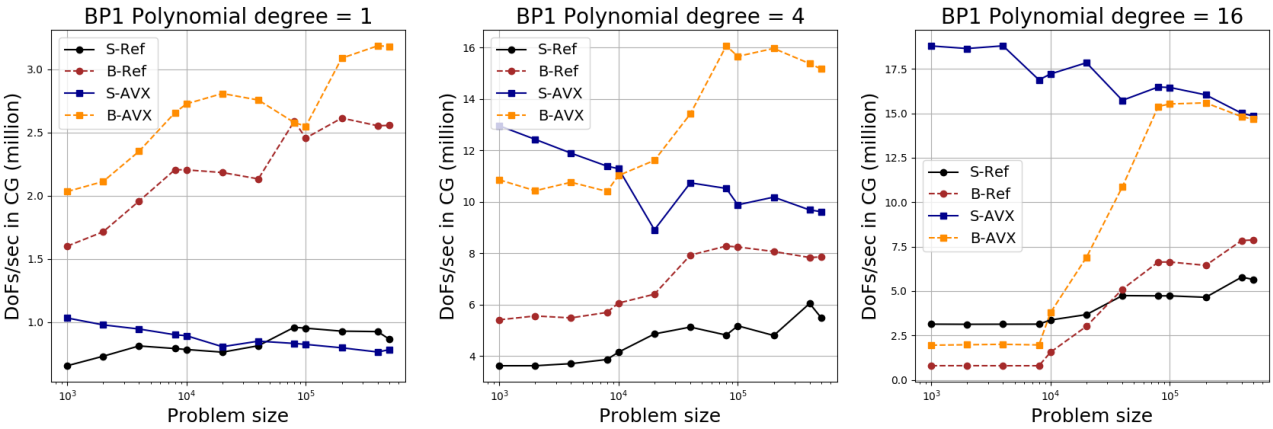

Figure 3: BP1 results, Problem size vs. DOFs/sec in CG at fixed polynomial degree

From the results, there is no doubt that the serial reference has the worst performance in BP1. For blocked reference and blocked AVX, they have a better performance if the problem size goes up but there is a fall after a particular polynomial degree (problem size 10^3 and 10^4), which means they are sensitive to both problem size and polynomial degree. On the other side, the serial AVX is more sensitive to the polynomial degree. Although its behaviour is less than blocked backends at small polynomial degree, it performs well in all the ranges of problem size when the polynomial degree improves. Therefore, the serial AVX is the best choice in BP1.

Polynomial degree	DOFs/sec in CG (million)											
Backends:	Serial reference			Blocked reference			Serial AVX			Blocked AVX		
Problem size:	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5
1	0.66	0.78	0.95	1.60	2.20	2.46	1.03	0.89	0.82	2.03	2.73	2.55
4	3.63	4.16	5.18	5.41	6.06	8.25	12.96	11.29	9.89	10.86	11.03	15.66
16	3.14	3.37	4.73	0.80	1.56	6.64	18.79	17.21	16.45	1.96	3.79	15.52

Table 1: BP1 results, DOFs/sec in CG for critical polynomial degree and problem size

3.2 BP2 Results

Fig. 4, 5 and Table 2 show the results of two benchmarks for BP2. Looking at Fig. 4, the DOFs/sec of blocked backends has the same behaviour (with a peak for all chosen problem sizes and dramatic drop for small and medium problem sizes) that have been observed in BP1 results. The blocked reference remain almost the same results compared to BP1, whereas the results of blocked AVX reach a high peak (over 20 million) in three cases. In terms of the serial backends, there is a similar performance for the serial reference between BP1 and BP2, and the serial AVX is always standing at second but doesn't perform pretty well compared to its results in BP1.

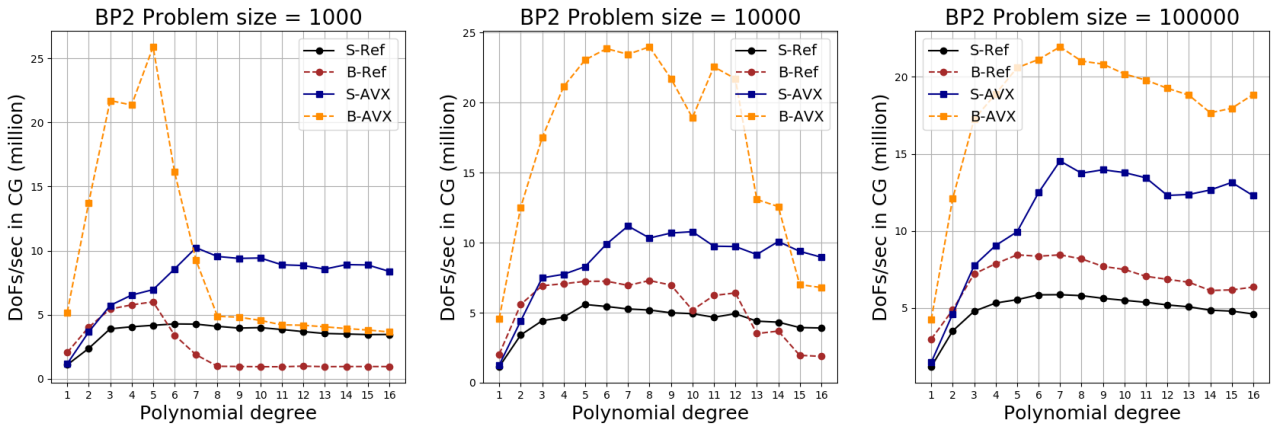

Figure 4: BP2 results, Polynomial degree vs. DOFs/sec in CG at fixed problem size

In Fig. 5, the blocked AVX has a good performance at three different polynomial degrees except for the first interval of problem size at polynomial degree 16. Similar to BP1, a significant improvement for blocked backends is found in the second interval at polynomial degree 16. The serial backends remain steady in a whole range of problem size, in addition, the results of serial AVX goes up along the increase of polynomial degree.

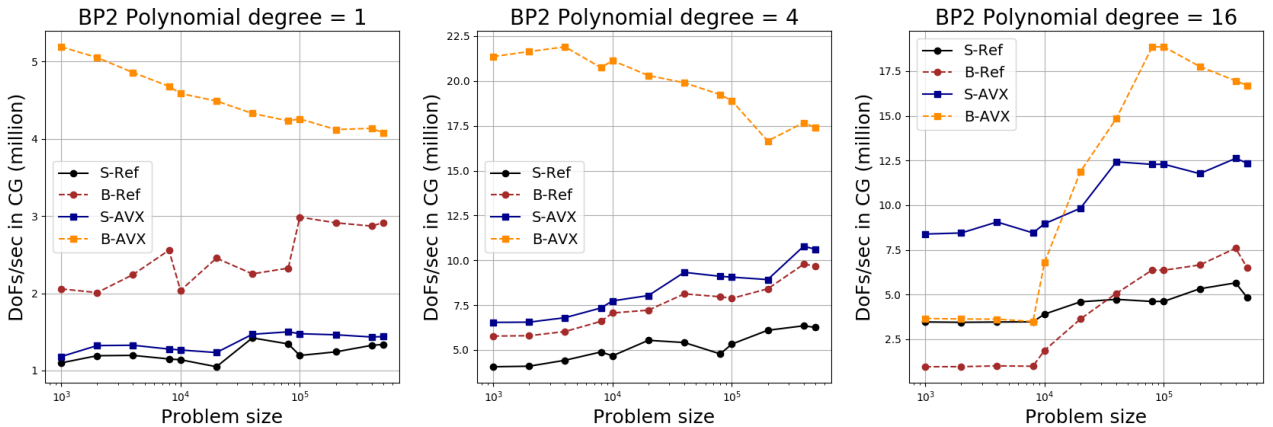

Figure 5: BP2 results, Problem size vs. DOFs/sec in CG at fixed polynomial degree

Through the observation from two benchmarks, the serial reference is the worst choice in BP2 due to the low values of DOFs/sec in every case. Different from the trend of BP1, both polynomial degree and problem size have an influence on the performance of the serial AVX in BP2. For blocked AVX, it is obviously more sensitive to the polynomial degree. Even though the bad performance in the small problem size at a larger polynomial degree, the high peak of DOFs/sec in every problem size makes blocked AVX is the best choice in BP2.

Polynomial degree	**DOFs/sec in CG (million)**											
Backends:	Serial reference			Blocked reference			Serial AVX			Blocked AVX		
Problem size:	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5
1	1.10	1.14	1.19	2.06	2.04	2.99	1.18	1.27	1.48	5.19	4.59	4.26
4	4.06	4.67	5.33	5.78	7.06	7.88	6.54	7.74	9.06	21.36	21.14	18.90
16	3.47	3.91	4.62	0.96	1.88	6.36	8.39	8.96	12.29	3.66	6.79	18.86

Table 2: BP2 results, DOFs/sec in CG for critical polynomial degree and problem size

4 Benchmark of Parallelism

To assess a benchmark of parallelism, the DOFs/sec in CG of four backends (see Section 3) were measured on the Hamilton par.7 partition. For this benchmark, three different implementations ran the test on a range of processes, from 2^0 to 2^7 processes. The polynomial degree was set as 16 and the problem size 10^3, 10^4, 10^5 were chosen to compare this benchmark in the small, medium and large size.

4.1 BP1 Results

The benchmark of parallelism for BP1 is shown in Figure 6 and Table 3. In Fig. 6, an obvious trend of four backends can be found that they are with multiple growth (twice) from 2^0 to 2^4 processes. After that, the performance of each backend remains steady or waves till the end. At the small problem size, two serial backends perform better than the other two blocked backends. However, along with the increase of problem size, the result of the blocked AVX is on the first, followed by the serial AVX.

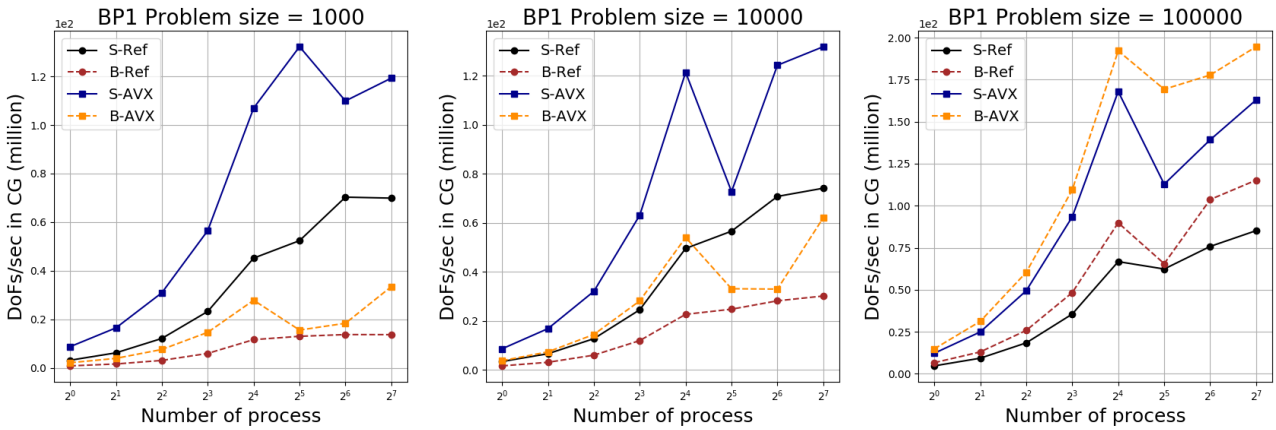

Figure 6: BP1 results, Number of process vs. DOFs/sec in CG at fixed problem size

Num. process	**DOFs/sec in CG (million)**											
Backends:	Serial reference			Blocked reference			Serial AVX			Blocked AVX		
Problem size:	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5
2	6.2	6.5	9.3	1.6	3.0	13.0	16.5	16.8	24.9	3.9	7.3	31.2
16	45.2	49.5	66.7	11.6	22.5	89.8	107.0	121.3	167.9	27.8	54.0	192.1
128	69.9	74.1	85.2	13.7	30.1	115.2	119.3	131.8	163.0	33.4	62.0	194.6

Table 3: BP1 results, DOFs/sec in CG for critical number of process and problem size

4.2 BP2 Results

Figure 7 and Table 4 show the results of the benchmark in MPI for BP2. Looking at Fig. 7, all the behaviour of four backends are quite similar to that in BP1, with a multiple growth in the beginning then few fluctuations or a slight improvement. Different from the results in BP2 without parallelism, the parallel performance of serial AVX is good throughout three different problem sizes at fixed polynomial degree 16. Therefore, the serial AVX has a major improvement through parallelism.

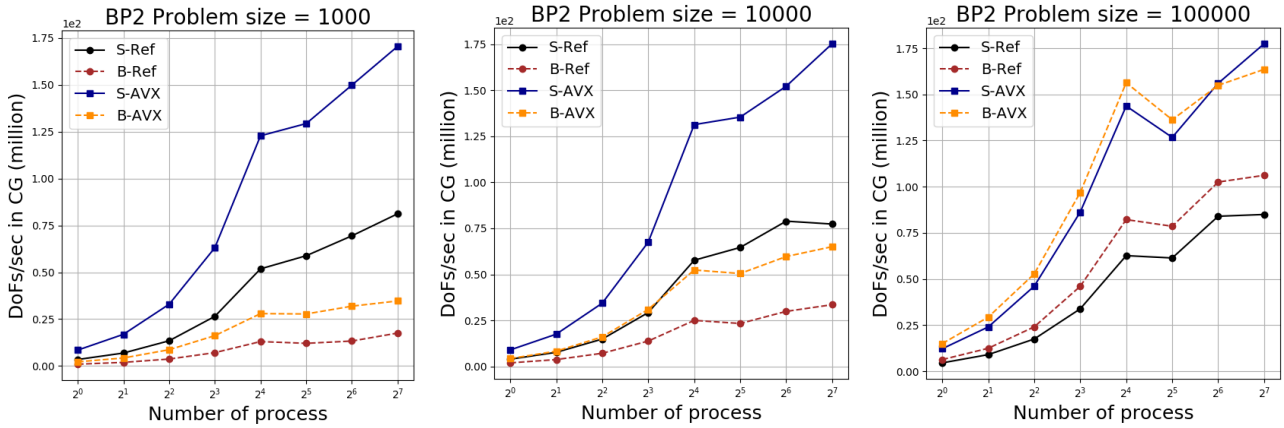

Figure 7: BP2 results, Number of process vs. DOFs/sec in CG at fixed problem size

Num. process	DOFs/sec in CG (million)											
Backends:	Serial reference			Blocked reference			Serial AVX			Blocked AVX		
Problem size:	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5	10^3	10^4	10^5
2	6.9	7.6	9.1	1.9	3.7	12.5	16.9	17.5	24.1	4.3	8.3	29.4
16	51.8	57.7	62.7	13.0	25.0	82.2	122.8	131.3	143.7	27.9	52.4	156.3
128	81.2	77.3	85.0	17.6	33.5	106.2	170.5	175.3	177.5	34.7	65.0	163.6

Table 4: BP2 results, DOFs/sec in CG for critical number of process and problem size

5 Conclusion

From the benchmark of backends, the best and the worst backend based on all the performance with different polynomial degrees and problem sizes in each problem were chosen. However, as the blocked case traverse the data in an order that is aware of the cache, it can obviously perform well when the problem size becomes large. Since the finite element problem is always complex and with a large problem size in the reality, the blocked AVX might be a better choice for both BP1 and BP2 in this situation. For the dramatic drop of blocked backends shown in Fig. 2 and 4, a possible reason is that using a higher polynomial degree to solve a simple problem increases the computation complexity for data traversing. Therefore, finding an appropriate polynomial degree to solve different problem sizes in finite element problem is important.

As for the benchmark of parallelism, the result will be expected to have linear growth along with the increase of process. However, in this study, the MPI parallelism is restricted to a single Hamilton node. As the memory bandwidth is a saturating resource that is competed by CPU cores, the benchmark result might show saturating performance. Based on this statement, the behaviour of the four backends that have been discussed in Section 4 is reasonable. In both Fig. 6 and 7, it should be noted that the memory bandwidth is still enough from 2^0 to 2^4 processes and become tight after that. Looking at the result of the large problem size, the higher value in the fluctuation range can be described as the performance limit. Therefore, the performance limit of AVX backends is higher than that of reference backends and should be closer to the hardware performance limit.