

Randomization of Sparse Matrix by Vector Multiplication

ABHISHEK JAIN, ISMAIL BUSTANY, and PAOLO D'ALBERTO

A sparse matrix by vector multiplication (SpMV) is simplified by the matrix non-zero elements and how we store them. There are many SpMV applications, many matrix storage formats, and thus algorithms. However, there is no optimality without considering the architecture: for example, the CPU is only one among ... many.

By nature, randomization is resilient to counter techniques, thus suitable to avoid worst case scenarios, improve performance on average, and reduce performance variance; however, it does to the best case the same thing it does to the worst case, it can nudge it off. Like preconditioning, randomization is advantageous when the matrix is reused or a constant such as in the power method, Krilov's space, or convolutions for image classifications. Randomization is also an optimization that any architecture may take advantage although in different ways.

We shall present cases where we can improve by 15% performance for general purpose architectures and by 8x for custom architectures.

ACM Reference Format:

Abhishek Jain, Ismail Bustany, and Paolo D'Alberto. 2020. Randomization of Sparse Matrix by Vector Multiplication . 1, 1 (May 2020), ?? pages.

1 INTRODUCTION

B.S. Goes here.

2 BASIC NOTATIONS

Let us start by describing the basic notations so we can clear the obvious (or not). A Sparse-matrix vector multiplication $SpMV$ on an (semi) ring based on the operations $(+, *)$ is defined as $\mathbf{y} = \mathbb{M}\mathbf{x}$ so that $y_i = \sum_j M_{i,j} * y_j$ where $M_{i,j}=0$ are not even represented and stored. Most of the experimental results in Section 9 are based on the classic addition $(+)$ and multiplication $(*)$ in floating point precision using 32 or 64bits (i.e., single and double floating point precision). SpMV based on semi-ring $(\min, +)$ is a short path algorithm based on an adjacent matrix of a graph, and using a Boolean algebra we can check if two nodes are connected, which is slightly simpler.

We identify a sparse matrix \mathbb{M} of size $M \times N$ as having $O(M + N)$ non-zero elements, number of non zero nnz . Thus the complexity of $\mathbb{M}\mathbf{x}$ is $O(M + N) = 2nnz$. Of course, the definition of sparsity may vary. We represent the matrix \mathbb{M} by using the Coordinate COO or and the compressed sparse row CSR ¹ format. The COO represents the non-zero of a matrix by a triplet (i, j, val) , very often there are three identical-in-size vectors for the ROW, COLUMN, and VALUE. The COO format takes $3 \times nnz$ space and two consecutive elements in the value array are not bound to be neither in the same row nor column. In fact, we know only that $VALUE[i] = M_{ROW[i], COLUMN[i]}$.

¹a.k.a. Compressed row storage CRS.

Authors' address: Abhishek Jain; Ismail Bustany; Paolo D'Alberto.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2020 Association for Computing Machinery.

Manuscript submitted to ACM

Manuscript submitted to ACM

The CSR stores elements in the same row and with increasing column values consecutively. There are three arrays V , COL , and ROW . The ROW is sorted in increasing order, its size is M , and $ROW[i]$ is an index in V and COL describing where row- i starts (i.e., if row i exists). We have that $M_{i,*}$ is stored in $V[ROW[i] : ROW[i + 1]]$ and the column are at $COL[ROW[i] : ROW[i + 1]]$ and sorted increasingly. The CSR takes $2 \times nnz + M$ space and a row vector of the matrix can be found in $O(1)$.

The computation as $y_i = \sum_j M_{i,j} * x_j$ is a sequence of dot products and the CSR representation is a natural:

$$Index = ROW[i] : ROW[i + 1]$$

$$y_i = \sum_{i \in Index} V[i] * x_{COL[i]}$$

The matrix row is contiguous (in memory) and contiguous rows are contiguous. The access of the (dense) vector x could have no pattern. The COO format could use a little preparation: For example, we can sort the array by row and add row information to achieve the same properties of CSR; however transposing a COO matrix is just a swap of the array ROW and COL . Think about matrix multiply. As today, each dot product achieves peak performance if the reads of the vector x are streamlined as much as possible and so the reads of the vector V . If we have multiple cores, each could compute a sub set of the y_i and a clean data load balancing can go a long way. If we have a few functional units, we would like to have a constant stream of independent $*$ and $+$ operations but with data already in registers: that is, data pre-fetch will go a long way especially for $x_{COL[i]}$, which may have an irregular pattern.

3 RANDOMIZATION

We refer to *Randomization* as row or column permutations of the matrix M (thus a permutation of y and x) and we choose these by a pseudo-random process. Why we want to introduce uncertainty? The sparsity of our matrix M has a pattern representing the nature of the original problem; such a pattern may exploit the wrong computation for an architecture; we could break such a pattern so that the only property left is a uniform distribution (of some sort). We must avoid the worst case and we would opt for an average case instead and we could do this to a class of M . This is the gist.

If we know the matrix M and we know the architecture, preconditioning must be a better solution. Well, it is. If we run experiments long enough, we choose the best permutations for the architecture, permute M , and go on testing the next. On one end, preconditioning exerts a full understanding of both the matrix (the problem) and how the final solution will be computed (architecture). This is the culminating point of knowing and we must strive to it. On the other end, the simplicity of a random permutation requires no information about the matrix, the vector, and the architecture. Such a simplicity can be exploited directly in HW. We are after an understanding when randomization is just enough: we want to let the hardware do its best with the least effort, or at least with the appearance to be effortless. Also we shall show there are different flavors of random.

Interestingly, this work stems from a sincere surprise about randomization efficacy and its application on custom SpMV. Here, we want to study this problem systematically so that to help future hardware designs. Intuitively, if we can achieve a uniform distribution of the rows of matrix M we can have provable expectation of its load balancing across multiple cores. If we have a uniform distribution of accesses on x we could exploit column load balancing and exploit better sorting algorithms: in practice the reading of $x_{COL[i]}$ can be reduces to a sorting and we know that different sparsity may require different algorithms. This is a lot to unpack but this translates as better performance of the sequential algorithm without changing the algorithm.

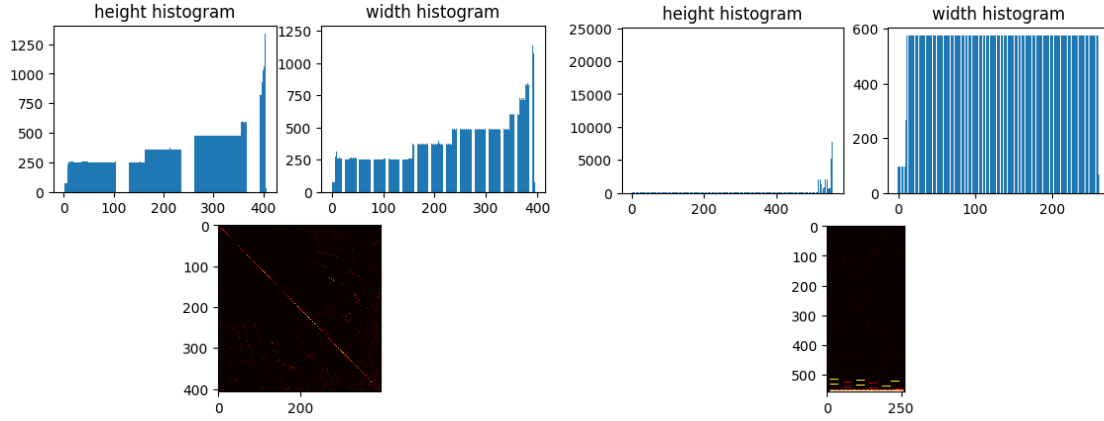


Fig. 1. Left: OPF 3754. Right: LP OSA 07. These are histograms where we represent normalized buckets and counts

We will show that (different) randomness affects architectures and algorithms differently making it a suitable optimization especially when the application and hardware are at odds. We want to show that there is a randomness hierarchy that we can distinguish as global and local; there are simple-to-find cases where the sparsity breaks randomness and the matrix has to be split into components. We want to show that this study uses common tool, open software tools and sometimes naive experiments; however, we can infer properties applicable to proprietary and custom solutions.

4 ENTROPY

Patterns in sparse matrices are often visually pleasing, see Figure 1 where we present the height histogram, the width histograms and a two-dimensional histogram as heat map. We will let someone else using AI picture classification. Intuitively, we would like to express a measure of uniform distribution and here we apply the basics: *Entropy*. Given an histogram $i \in [0, M - 1]$ $h_i \in \mathbb{N}$, we define $S = \sum_{i=0}^{M-1} h_i$ and thus we have a probability distribution function $p_i = \frac{h_i}{S}$. The *information* of bin i is defined as $I(i) = -\log_2 p_i$. If we say that the stochastic variable X has PDF p_i than the entropy of X is defined as.

$$H(x) = - \sum_{i=0}^{M-1} p_i \log_2 p_i = \sum_{i=0}^{M-1} p_i I(i) = E[I_x] \quad (1)$$

The maximum entropy is when $\forall i, p_i = p = \frac{1}{M}$; that is, we are observing a uniform distributed event. There is no conceptual difference when the PDF represents a two dimensional distribution. Thus our randomization should aim at higher entropy numbers.

The entropy for matrix LP OSA 07 is 8.41 and for OPF 3754 is 8.39. A single number is satisfying because concise.

5 UNIFORM DISTRIBUTION

We know that we should **not** compare the entropy numbers of two matrices because there entropy does not use any information about the order of the buckets. By construction, the matrices are quite different in sparsity, ins shapes and their entropy numbers are so close. To appreciate their difference, we should compare their distributions by Jensen-Shannon measure (which is a symmetric). Or we could use a representation of a hierarchical 2d-entropy, see Figure

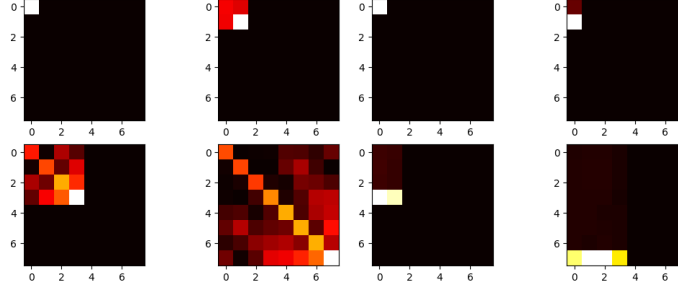


Fig. 2. Hierarchical 2D entropy for OPF 3754 (left) and LP OSA 07 (right).

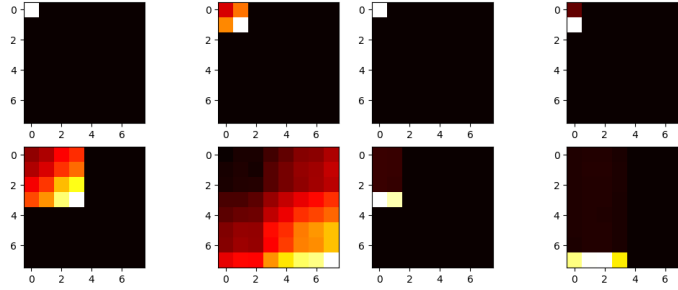


Fig. 3. Hierarchical 2D entropy after row and column random permutation for OPF 3754 (left) and LP OSA 07 (right).

2, where the entropy is split into 2x2, 4x4 and 8x8 (or fewer if the distribution is not square). We have a hierarchical entropy heat maps.

We can see a more granular entropy measure summarizes better the nature of the matrix. In this work, the entropy vector is used mostly for visualization purpose more than for comparison purpose. Of course, we can appreciate how the matrix LP OSA 07 has a few very heavy rows and they are clustered. This matrix will help up in showing how randomization need some tips. Now we apply row and column random permutation once by row and one by column: Figure 3: OPF has now entropy 11.27 and LP 9.26. The numerical difference is significant. The good news is that for entropy, being an expectation, we can use simple techniques like bootstrap to show that the difference is significant or we have shown that Jensen-Shannon can be used and a significance level is available. What we like to see is the the hierarchical entropy heat map is becoming *more* uniform for at least one of the matrix.

In practice, permutation need some help especially for relatively large matrices. As you can see, the permutation affects locally the matrix. Of course, it depends on the implementation of the random permutation (we use numpy for this) but it is reasonable a slightly modified version of the original is still a random selection but unfortunately they seem more likely than they should. We need to compensate or help the randomization so that this current implementation does not get too lazy.

If we are able to identify the row and column that divide high and low density, we could use them as pivot for a shuffle like in a quick-sort algorithm. We could apply a sorting algorithm but its complexity will the same of SpMV. We use a gradients operations to choose the element with maximum steepness, Figure 4 and 6

LP achieves entropy 8.67 and 9.58 and OPF achieves 10.47 and 11.40.

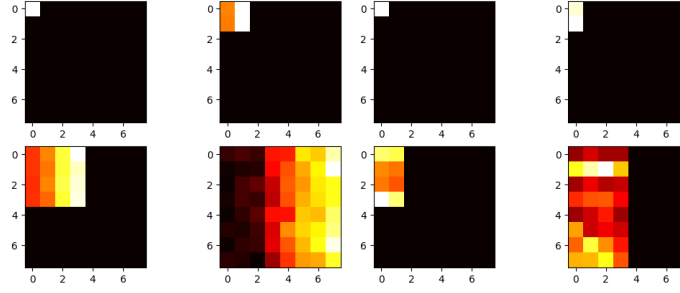


Fig. 4. Hierarchical 2D entropy after height gradient based shuffle and row random permutation for OPF 3754 (left) and LP OSA 07 (right).

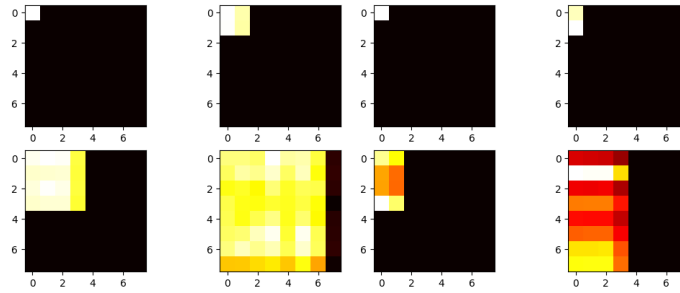


Fig. 5. Hierarchical 2D entropy after height and width gradient shuffle and row and column random permutation for OPF 3754 (left) and LP OSA 07 (right).

If the goal is to achieve a uniform matrix sparsity, it seems that we have the basic tool to compute and to measure such a sparsity. We admit that we do not try to find the best permutation. But our real goal is to create a work bench where randomization can be tested on different architectures and different algorithms.

6 MEASURING THE RANDOMIZATION EFFECTS

Whether or not this applied to the reader, when we have timed execution of algorithms we came to expect variation. The introduction of randomization may hide behind the ever present random behavior, after all these are algorithms on *small* inputs and small error can be comparable to the overall execution time. Here, we must address this concern even before describing the experiments.

First, every algorithm is run between 1000 and 5000 times. The time of each experiments is in the seconds, providing a granularity we are confident that error in measuring time (per se) is under control. Thus, for each experiment we provide an average execution time: we measure the time and we divide by the number of trials. Cold starts, the first iteration, are still accounted. To make the measure portable across platform we present GFLOPS, that is, Giga (10^9) floating operations per second: $2 * nnz$ divided by the average time in seconds.

Then we repeat the same experiment 32 times. Permutations in *numpy* Python use a seed that time sensitive and thus every experiment is independent from the previous. The number 32 is an old statistic trick and it is a minimum number of independent trials to approximate an normal distribution. In practice, they are not but the number is sufficient for most of the cases and it is an excellent starting point.

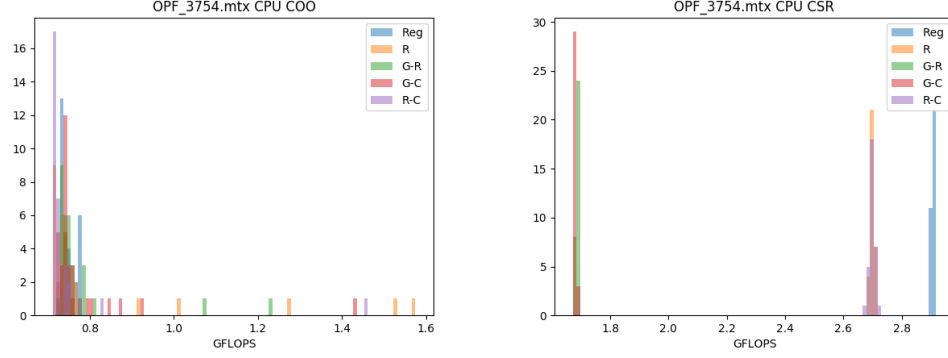


Fig. 6. CPU COO (left) and CPU CSR (left) for OPF 3754

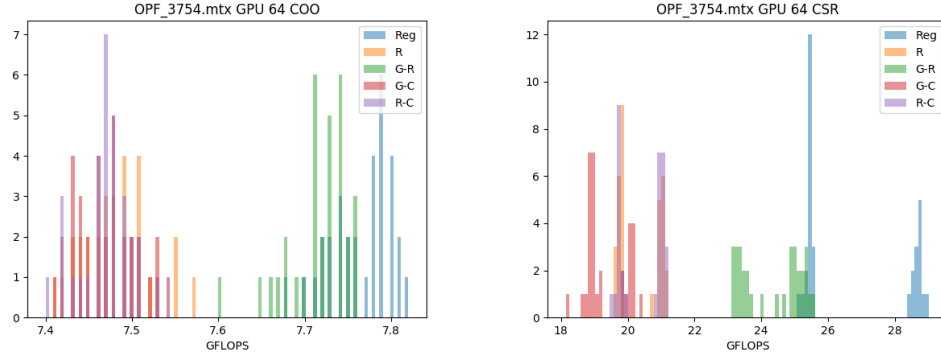


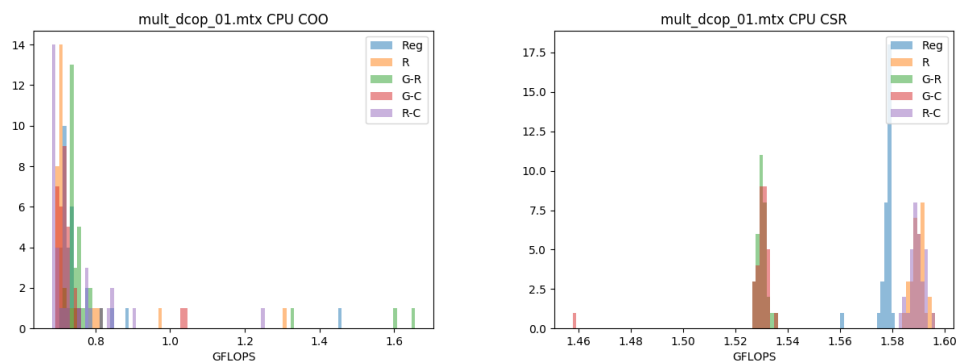
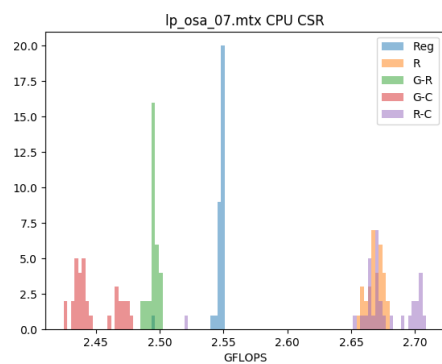
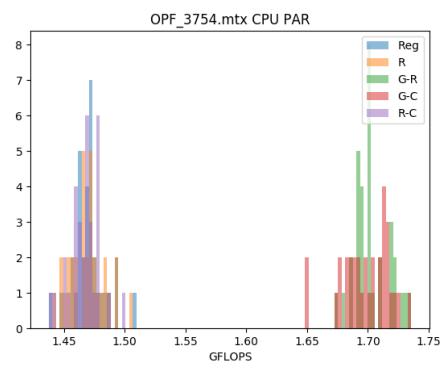
Fig. 7. GPU 64bits COO (left) and GPU CSR (left) for OPF 3754

A short legend: **Reg** is the matrix without any permutation and thus is the regular; **R** stands for random Row permutation; **G-R** stands for gradient-based row shuffle and random row permutation; **G-C** stands for gradient-based column shuffle and random column permutation; **R-C** stands for random row and column permutation. Gradient based approach shall be clarified further in the experimental results section 9. Intuitively, we help the random permutation by a quick targeting of high and low volume of the matrix.

In Figure 6, We show CPU performance using COO and CSR SpMV algorithms for the matrix OPF 3754. We can see that the CSR algorithms are consistent and the Regular (i.e., the original) has always the best performance. For the COO, permutations introduce a long tails. In Figure 7, Randomization is harmful to the GPU implementation. If the load balance is fixed (i.e., by dividing the matrix by row and in equal row), randomization is beneficial.

For matrix LP OSA 07, randomization helps clearly only for CPU CSR as we show in Figure 9

An example, the matrix MULT DCOP 01, is where randomization is useful for the CPU, GPU, and the parallel version Figure 10, 11, and 12.



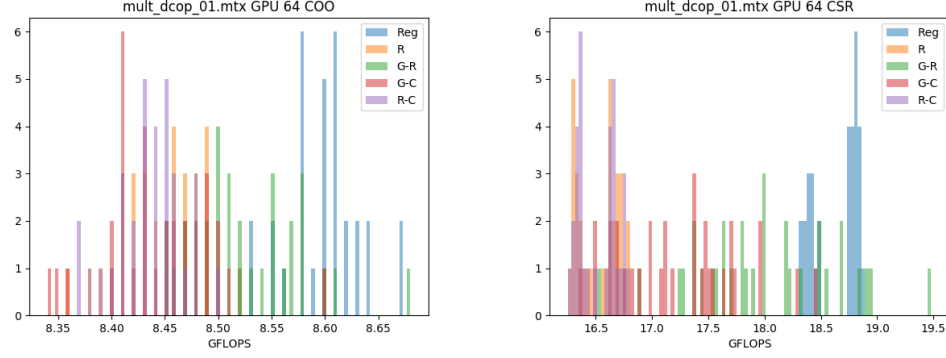


Fig. 11. GPU 64bits COO (left) and GPU CSR (left) for MULT DCOP 01

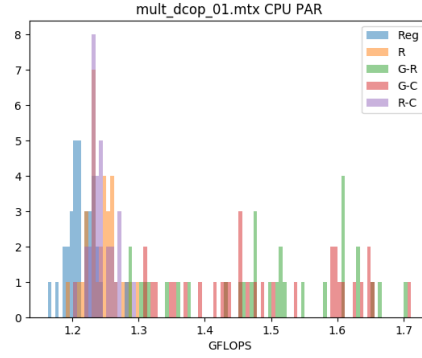


Fig. 12. Parallel CPU CSR (left) for MULT DCOP 01

7 WORKLOADS

In the previous sections, we defined what we mean for randomization and we present our tools of tricks for the measure of the effects of randomization. Here we describe the work loads, the applications, we use to test the effects of the randomization.

7.1 Python COO and CSR algorithms

The simplicity to compute the SpMV by the code $z = A * b$ in Python is very rewarding. By change of the matrix storage format, $AC = A.tocsr(); z = AC * b$, we have a different algorithm. The performance exploitation is moved to the lower level. The CSR implementation is often two times faster but there are edges cases where the COO and COO with randomization can go beyond and be surprisingly better: MUL DCOP 03 is an example where COO can do well.

Intuitively, Randomization can affect the performance because the basic implementation is a sorting algorithm and it is a fixed algorithm. There are many sorting algorithms and each can be optimal for a different initial distribution. If we knew what is the sorting algorithm we could tailor the input distribution. Here we just play with it.

7.2 Parallel CSR using up to 16 cores

Python provides the concept of Pool to exploit a naive parallel computation. We notice that work given to a Pool are split accordingly to the number of elements to separate HW cores. We also noticed that the work load can move from a core to another, thus may not be ideal. Also we notice that Pool introduce a noticeable overhead: a Pool of 1, never achieves the performance of the single thread $z = AC * b$. Using Pool allows us to investigate how a naive row partitioning without counting can scale up with number of cores. Randomization goal is to distribute the work uniformly: a balanced work distribution avoid the unfortunate case where a single core does all the work.

7.3 GPU COO and CSR algorithms

In this work, we use AMD GPUs and *rocSPARSE* is their current software. The software has a few glitches but overall can be used for different generation of AMD GPUs. We use the COO and CSR algorithms and when possible or useful we provide performance measure for single and double precision (mostly double precision). The ideas of using different GPUs is important to verify that the randomization can be applied independently of the HW. We are not here to compare performance with other GPUs or even between CPUs and GPUs.

The performance of the CSR algorithm is about two time faster than the COO. Most of the algorithms use the CSR format to count the number of sparse elements in a row and thus they can decide the work load partition accordingly. Counting give you an edge but without changing the order of the computation there could be cases where the work load is not balanced and a little randomization could help and it helps.

7.4 Randomization sometimes works

For the majority of the cases we investigated and reported in the following sections, Randomization does not work and it affects the performance negatively. However, there are cases that do work and do work for different algorithms and architectures. If you are in the business of preconditioning, permutations are pretty cheap. Of course, permutation changes the computation order and this may affect precision: for low precision matrices such as half floating point (fp16) or smaller we may re-evaluate. For the semiring (min,+) and for integer arithmetic the computation order does not matter.

8 CALL FOR A DIFFERENT STRATEGY

We want to find out randomization techniques that are suitable for custom hardware but also what are the most common and simple heuristics that can justified for any hardware.

9 EXPERIMENTAL RESULTS

Plots and pots.

10 VEGA VII

mult_dcop_03.mtx

Regular

CPU COO	min	0.728	max	0.880	mean	0.757
CPU CSR	min	1.563	max	1.581	mean	1.577
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.540	max	8.670	mean	8.619
CSR	min	18.320	max	18.930	mean	18.620
CPU PAR	min	1.170	max	1.269	mean	1.226
H	min	9.689	max	9.689	mean	9.689

Row-Premute

CPU COO	min	0.710	max	0.845	mean	0.724
CPU CSR	min	1.549	max	1.597	mean	1.589
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.360	max	8.540	mean	8.442
CSR	min	16.260	max	16.780	mean	16.551
CPU PAR	min	1.205	max	1.319	mean	1.263
H	min	10.737	max	10.742	mean	10.740

Row-Gradient

CPU COO	min	0.706	max	1.603	mean	0.806
CPU CSR	min	1.493	max	1.534	mean	1.528
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.430	max	8.610	mean	8.527
CSR	min	17.070	max	18.970	mean	18.115
CPU PAR	min	1.331	max	1.695	mean	1.513
H	min	10.576	max	10.585	mean	10.580

Column-Gradient

CPU COO	min	0.694	max	1.632	mean	0.797
CPU CSR	min	1.491	max	1.534	mean	1.529
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.350	max	8.520	mean	8.429
CSR	min	15.970	max	18.180	mean	17.124
CPU PAR	min	1.321	max	1.728	mean	1.514
H	min	10.826	max	10.840	mean	10.833

Row-Column-Permute

CPU COO	min	0.688	max	0.757	mean	0.696
CPU CSR	min	1.490	max	1.595	mean	1.584
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.380	max	8.500	mean	8.445
CSR	min	16.230	max	16.780	mean	16.513
CPU PAR	min	1.192	max	1.274	mean	1.237
H	min	10.737	max	10.742	mean	10.740

mult_dcop_01.mtx

Regular

CPU COO	min	0.710	max	1.453	mean	0.761
CPU CSR	min	1.561	max	1.581	mean	1.578
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.520	max	8.670	mean	8.597
CSR	min	18.320	max	18.870	mean	18.636
CPU PAR	min	1.163	max	1.246	mean	1.212
H	min	9.689	max	9.689	mean	9.689

Row-Premute

CPU COO	min	0.699	max	1.305	mean	0.745
CPU CSR	min	1.585	max	1.597	mean	1.590
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.360	max	8.520	mean	8.446
CSR	min	16.260	max	16.780	mean	16.528
CPU PAR	min	1.192	max	1.298	mean	1.242
H	min	10.738	max	10.742	mean	10.740

Row-Gradient

CPU COO	min	0.709	max	1.656	mean	0.819
CPU CSR	min	1.527	max	1.535	mean	1.530
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.450	max	8.680	mean	8.527
CSR	min	16.520	max	19.480	mean	17.984
CPU PAR	min	1.280	max	1.704	mean	1.485
H	min	10.572	max	10.585	mean	10.581

Column-Gradient

CPU COO	min	0.698	max	1.042	mean	0.737
CPU CSR	min	1.458	max	1.536	mean	1.528
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.340	max	8.600	mean	8.443
CSR	min	16.360	max	18.450	mean	17.247
CPU PAR	min	1.307	max	1.712	mean	1.494
H	min	10.823	max	10.841	mean	10.835

Row-Column-Permute

CPU COO	min	0.683	max	1.247	mean	0.749
CPU CSR	min	1.583	max	1.595	mean	1.590
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.370	max	8.500	mean	8.435
CSR	min	16.250	max	16.780	mean	16.518
CPU PAR	min	1.206	max	1.291	mean	1.243
H	min	10.738	max	10.742	mean	10.740

mult_dcop_02.mtx

Regular

CPU COO	min	1.615	max	1.677	mean	1.652
CPU CSR	min	1.539	max	1.579	mean	1.575
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.530	max	8.700	mean	8.614
CSR	min	18.290	max	18.890	mean	18.597
CPU PAR	min	1.120	max	1.248	mean	1.211
H	min	9.689	max	9.689	mean	9.689

Row-Premute

CPU COO	min	0.684	max	0.780	mean	0.705
CPU CSR	min	1.558	max	1.596	mean	1.588
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.360	max	8.490	mean	8.433
CSR	min	16.240	max	16.750	mean	16.552
CPU PAR	min	1.182	max	1.277	mean	1.242
H	min	10.737	max	10.742	mean	10.740

Row-Gradient

CPU COO	min	0.704	max	1.373	mean	0.790
CPU CSR	min	1.518	max	1.535	mean	1.529
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.420	max	8.590	mean	8.517
CSR	min	16.680	max	19.550	mean	17.907
CPU PAR	min	1.328	max	1.713	mean	1.484
H	min	10.572	max	10.585	mean	10.581

Column-Gradient

CPU COO	min	0.697	max	1.460	mean	0.742
CPU CSR	min	1.517	max	1.534	mean	1.527
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.330	max	8.490	mean	8.420
CSR	min	16.020	max	18.390	mean	17.303
CPU PAR	min	1.321	max	1.709	mean	1.557
H	min	10.823	max	10.843	mean	10.835

Row-Column-Permute

CPU COO	min	0.691	max	0.746	mean	0.698
CPU CSR	min	1.568	max	1.595	mean	1.587
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	8.350	max	8.500	mean	8.436
CSR	min	16.250	max	16.780	mean	16.517
CPU PAR	min	1.187	max	1.280	mean	1.228
H	min	10.739	max	10.743	mean	10.740

lp_fit2d.mtx

Regular

CPU COO	min	0.774	max	0.804	mean	0.793
CPU CSR	min	2.538	max	2.550	mean	2.547
GPU 32 COO	min	0.000	max	0.000	mean	0.000
CSR	min	0.000	max	0.000	mean	0.000
GPU 64 COO	min	7.060	max	7.170	mean	7.101
CSR	min	15.650	max	18.700	mean	18.031
CPU PAR	min	1.537	max	1.645	mean	1.590
H	min	11.109	max	11.109	mean	11.109

Row-Premute	CPU COO	min	0.740	max	0.776	mean	0.746	CPU COO	min	0.670	max	1.024	mean	0.706
	CPU CSR	min	3.302	max	3.328	mean	3.317	CPU CSR	min	2.199	max	2.340	mean	2.326
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000
	GPU 64 COO	min	7.040	max	7.180	mean	7.098	GPU 64 COO	min	6.880	max	6.980	mean	6.933
	CSR	min	15.690	max	18.580	mean	16.732	CSR	min	15.610	max	16.900	mean	16.227
	CPU PAR	min	1.327	max	1.482	mean	1.422	CPU PAR	min	1.598	max	1.668	mean	1.632
	H	min	11.098	max	11.105	mean	11.101	H	min	11.025	max	11.032	mean	11.029
Row-Gradient	CPU COO	min	0.739	max	2.092	mean	1.091	CPU COO	min	0.715	max	1.798	mean	0.885
	CPU CSR	min	2.539	max	2.546	mean	2.543	CPU CSR	min	2.495	max	2.551	mean	2.547
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000
	GPU 64 COO	min	7.040	max	7.150	mean	7.100	GPU 64 COO	min	7.650	max	7.790	mean	7.718
	CSR	min	15.520	max	18.560	mean	17.547	CSR	min	16.390	max	18.350	mean	17.093
	CPU PAR	min	1.401	max	1.661	mean	1.525	CPU PAR	min	0.963	max	1.012	mean	0.995
	H	min	11.109	max	11.109	mean	11.109	H	min	8.412	max	8.412	mean	8.412
Column-Gradient	CPU COO	min	0.726	max	2.065	mean	1.011	CPU COO	min	0.720	max	2.078	mean	1.104
	CPU CSR	min	2.539	max	2.550	mean	2.546	CPU CSR	min	2.656	max	2.679	mean	2.669
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000
	GPU 64 COO	min	6.800	max	7.140	mean	7.080	GPU 64 COO	min	7.610	max	7.690	mean	7.647
	CSR	min	15.480	max	18.560	mean	16.866	CSR	min	15.910	max	17.210	mean	16.750
	CPU PAR	min	1.391	max	1.737	mean	1.563	CPU PAR	min	0.890	max	0.940	mean	0.918
	H	min	11.329	max	11.333	mean	11.331	H	min	9.255	max	9.258	mean	9.256
Row-Column-Permute	CPU COO	min	0.746	max	0.782	mean	0.754	CPU COO	min	0.725	max	2.078	mean	1.041
	CPU CSR	min	3.310	max	3.324	mean	3.318	CPU CSR	min	2.487	max	2.502	mean	2.495
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000
	GPU 64 COO	min	7.030	max	7.160	mean	7.100	GPU 64 COO	min	7.570	max	7.730	mean	7.655
	CSR	min	15.730	max	18.530	mean	17.362	CSR	min	15.370	max	18.100	mean	16.803
	CPU PAR	min	1.340	max	1.451	mean	1.401	CPU PAR	min	1.435	max	1.796	mean	1.592
	H	min	11.099	max	11.104	mean	11.102	H	min	8.637	max	8.678	mean	8.672
bloweya.mtx Regular	CPU COO	min	0.727	max	1.815	mean	0.892	CPU COO	min	0.724	max	1.990	mean	1.000
	CPU CSR	min	2.867	max	2.936	mean	2.917	CPU CSR	min	2.425	max	2.477	mean	2.448
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000
	GPU 64 COO	min	0.000	max	0.000	mean	0.000	GPU 64 COO	min	7.510	max	7.660	mean	7.596
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	14.410	max	16.290	mean	15.267
	CPU PAR	min	1.680	max	1.751	mean	1.719	CPU PAR	min	1.238	max	1.774	mean	1.534
	H	min	7.205	max	7.205	mean	7.205	H	min	9.447	max	9.603	mean	9.576
Row-Premute	CPU COO	min	0.678	max	1.483	mean	0.746	CPU COO	min	0.738	max	1.950	mean	1.071
	CPU CSR	min	2.311	max	2.326	mean	2.320	CPU CSR	min	2.522	max	2.709	mean	2.675
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000
	GPU 64 COO	min	6.840	max	7.270	mean	6.930	GPU 64 COO	min	7.600	max	7.690	mean	7.641
	CSR	min	15.650	max	16.800	mean	16.233	CSR	min	15.820	max	17.190	mean	16.572
	CPU PAR	min	1.649	max	1.730	mean	1.682	CPU PAR	min	0.891	max	0.944	mean	0.924
	H	min	11.026	max	11.031	mean	11.029	H	min	9.255	max	9.258	mean	9.256
Row-Gradient	CPU COO	min	0.708	max	1.209	mean	0.779	CPU COO	min	0.732	max	1.837	mean	1.076
	CPU CSR	min	1.648	max	1.735	mean	1.709	CPU CSR	min	2.563	max	2.586	mean	2.577
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000
	GPU 64 COO	min	6.920	max	7.080	mean	7.015	GPU 64 COO	min	11.340	max	11.860	mean	11.441
	CSR	min	16.950	max	19.500	mean	17.794	CSR	min	36.010	max	40.960	mean	38.048
	CPU PAR	min	1.497	max	1.743	mean	1.608	CPU PAR	min	2.019	max	2.204	mean	2.130
	H	min	10.298	max	10.304	mean	10.301	H	min	8.228	max	8.228	mean	8.228
Column-Gradient	CPU COO	min	0.709	max	1.536	mean	0.817	CPU COO	min	0.718	max	0.751	mean	0.732
	CPU CSR	min	1.705	max	1.753	mean	1.735	CPU CSR	min	2.488	max	2.507	mean	2.498
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000
	GPU 64 COO	min	6.800	max	7.120	mean	6.865	GPU 64 COO	min	10.810	max	11.090	mean	10.949
	CSR	min	15.480	max	17.710	mean	16.470	CSR	min	24.860	max	26.410	mean	25.527
	CPU PAR	min	1.446	max	1.718	mean	1.591	CPU PAR	min	1.978	max	2.290	mean	2.135
	H	min	10.880	max	10.886	mean	10.883	H	min	11.836	max	11.840	mean	11.838
Row-Column-Permute														

Column-Gradient	CPU COO	min	0.722	max	1.794	mean	0.769	
	CPU CSR	min	2.407	max	2.421	mean	2.416	
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	11.210	max	11.480	mean	11.317	
	CSR	min	31.920	max	34.690	mean	33.246	
	CPU PAR	min	2.184	max	2.302	mean	2.232	
	H	min	10.742	max	10.757	mean	10.748	
	Row-Column-Permute	CPU COO	min	0.720	max	0.916	mean	0.742
		CPU CSR	min	2.395	max	2.410	mean	2.402
GPU 32 COO		min	0.000	max	0.000	mean	0.000	
CSR		min	0.000	max	0.000	mean	0.000	
GPU 64 COO		min	10.840	max	11.070	mean	10.946	
CSR		min	24.340	max	26.140	mean	25.393	
CPU PAR		min	2.184	max	2.272	mean	2.223	
H		min	11.873	max	11.882	mean	11.878	
brainpc2.mtx Regular		CPU COO	min	0.707	max	0.748	mean	0.714
		CPU CSR	min	2.458	max	2.511	mean	2.506
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	10.880	max	11.070	mean	10.957	
	CSR	min	24.890	max	26.490	mean	25.642	
	CPU PAR	min	2.209	max	2.282	mean	2.240	
	H	min	11.834	max	11.840	mean	11.838	
	Row-Gradient	CPU COO	min	0.732	max	0.751	mean	0.744
		CPU CSR	min	2.885	max	2.916	mean	2.909
GPU 32 COO		min	0.000	max	0.000	mean	0.000	
CSR		min	0.000	max	0.000	mean	0.000	
GPU 64 COO		min	0.000	max	0.000	mean	0.000	
CSR		min	0.000	max	0.000	mean	0.000	
CPU PAR		min	1.276	max	1.299	mean	1.286	
H		min	7.478	max	7.478	mean	7.478	
Column-Gradient		CPU COO	min	0.702	max	1.626	mean	0.844
		CPU CSR	min	1.327	max	1.374	mean	1.364
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	6.920	max	7.210	mean	7.030	
	CSR	min	0.000	max	15.260	mean	14.279	
	CPU PAR	min	1.283	max	1.531	mean	1.385	
	H	min	10.572	max	10.595	mean	10.590	
	Row-Preempt	CPU COO	min	0.727	max	0.855	mean	0.736
		CPU CSR	min	2.385	max	2.411	mean	2.397
GPU 32 COO		min	0.000	max	0.000	mean	0.000	
CSR		min	0.000	max	0.000	mean	0.000	
GPU 64 COO		min	8.120	max	8.410	mean	8.206	
CSR		min	18.670	max	19.960	mean	19.536	
CPU PAR		min	1.293	max	1.340	mean	1.314	
H		min	9.809	max	9.813	mean	9.811	
Row-Gradient		CPU COO	min	0.696	max	1.546	mean	0.785
		CPU CSR	min	1.361	max	1.420	mean	1.411
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	8.190	max	8.550	mean	8.302	
	CSR	min	18.700	max	21.000	mean	19.890	
	CPU PAR	min	1.435	max	1.666	mean	1.549	
	H	min	9.721	max	9.727	mean	9.723	
	Column-Gradient	CPU COO	min	0.698	max	1.467	mean	0.746
		CPU CSR	min	1.377	max	1.423	mean	1.414
GPU 32 COO		min	0.000	max	0.000	mean	0.000	
CSR		min	0.000	max	0.000	mean	0.000	
GPU 64 COO		min	8.110	max	8.290	mean	8.187	
CSR		min	18.090	max	20.190	mean	19.217	
CPU PAR		min	1.345	max	1.681	mean	1.518	
H		min	10.369	max	10.372	mean	10.370	
Row-Column-Permute		CPU COO	min	0.698	max	1.390	mean	0.788
		CPU CSR	min	2.387	max	2.410	mean	2.399
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	8.120	max	8.260	mean	8.191	
	CSR	min	18.530	max	19.960	mean	19.307	
	CPU PAR	min	1.295	max	1.347	mean	1.319	
	H	min	9.809	max	9.813	mean	9.811	
	Column-Gradient	CPU COO	min	0.693	max	1.521	mean	0.772
		CPU CSR	min	1.287	max	1.305	mean	1.296
GPU 32 COO		min	0.000	max	0.000	mean	0.000	
CSR		min	0.000	max	0.000	mean	0.000	
GPU 64 COO		min	5.920	max	6.000	mean	5.962	
CSR		min	16.810	max	18.410	mean	17.561	
CPU PAR		min	1.378	max	1.485	mean	1.429	
H		min	11.061	max	11.069	mean	11.064	

case9.mtx Regular	Row-Column-Permute	CPU CSR	min	1.291	max	1.302	mean	1.297	Row-Gradient	CPU CSR	min	2.450	max	2.543	mean	2.525
		GPU 32 COO	min	0.000	max	0.000	mean	0.000		GPU 32 COO	min	0.000	max	0.000	mean	0.000
		CSR	min	0.000	max	0.000	mean	0.000		CSR	min	0.000	max	0.000	mean	0.000
		GPU 64 COO	min	5.900	max	6.060	mean	5.960		GPU 64 COO	min	7.200	max	7.320	mean	7.268
		CSR	min	16.620	max	18.330	mean	17.592		CSR	min	17.420	max	18.540	mean	18.102
		CPU PAR	min	1.372	max	1.464	mean	1.409		CPU PAR	min	1.474	max	1.595	mean	1.546
		H	min	11.127	max	11.135	mean	11.130		H	min	10.042	max	10.046	mean	10.044
		CPU COO	min	0.704	max	1.503	mean	0.875		CPU COO	min	0.712	max	0.926	mean	0.750
		CPU CSR	min	2.447	max	2.468	mean	2.459		CPU CSR	min	1.819	max	1.846	mean	1.832
		GPU 32 COO	min	0.000	max	0.000	mean	0.000		GPU 32 COO	min	0.000	max	0.000	mean	0.000
case9.mtx Regular	Row-Gradient	CSR	min	0.000	max	0.000	mean	0.000	Column-Gradient	CSR	min	0.000	max	0.000	mean	0.000
		GPU 64 COO	min	5.880	max	5.980	mean	5.931		GPU 64 COO	min	7.210	max	7.370	mean	7.298
		CSR	min	17.550	max	19.140	mean	18.227		CSR	min	17.550	max	20.740	mean	19.089
		CPU PAR	min	1.639	max	1.743	mean	1.704		CPU PAR	min	1.256	max	1.554	mean	1.495
		H	min	11.028	max	11.035	mean	11.030		H	min	9.666	max	9.704	mean	9.690
	Row-Column-Permute	CPU COO	min	0.721	max	1.800	mean	1.177		CPU COO	min	0.710	max	1.690	mean	0.791
		CPU CSR	min	3.021	max	3.046	mean	3.036		CPU CSR	min	1.813	max	1.836	mean	1.830
		GPU 32 COO	min	0.000	max	0.000	mean	0.000		GPU 32 COO	min	0.000	max	0.000	mean	0.000
		CSR	min	0.000	max	0.000	mean	0.000		CSR	min	0.000	max	0.000	mean	0.000
		GPU 64 COO	min	0.000	max	0.000	mean	0.000		GPU 64 COO	min	7.130	max	7.310	mean	7.211
		CSR	min	0.000	max	0.000	mean	0.000		CSR	min	16.550	max	18.690	mean	17.617
		CPU PAR	min	1.508	max	1.605	mean	1.573		CPU PAR	min	1.385	max	1.539	mean	1.506
		H	min	7.380	max	7.380	mean	7.380		H	min	10.611	max	10.659	mean	10.634
	Row-Gradient	CPU COO	min	0.724	max	1.100	mean	0.765		CPU COO	min	0.709	max	1.531	mean	0.963
		CPU CSR	min	2.581	max	2.626	mean	2.609		CPU CSR	min	2.506	max	2.648	mean	2.622
		GPU 32 COO	min	0.000	max	0.000	mean	0.000		GPU 32 COO	min	0.000	max	0.000	mean	0.000
		CSR	min	0.000	max	0.000	mean	0.000		CSR	min	0.000	max	0.000	mean	0.000
		GPU 64 COO	min	7.170	max	7.340	mean	7.253		GPU 64 COO	min	7.140	max	7.330	mean	7.244
		CSR	min	17.360	max	18.500	mean	18.014		CSR	min	17.410	max	18.520	mean	18.148
		CPU PAR	min	1.494	max	1.607	mean	1.558		CPU PAR	min	1.466	max	1.574	mean	1.528
		H	min	10.043	max	10.047	mean	10.044		H	min	10.041	max	10.046	mean	10.044
	Row-Column-Permute	CPU COO	min	0.716	max	1.701	mean	0.804		CPU COO	min	0.714	max	0.731	mean	0.720
		CPU CSR	min	1.824	max	1.840	mean	1.832		CPU CSR	min	2.667	max	2.770	mean	2.720
		GPU 32 COO	min	0.000	max	0.000	mean	0.000		GPU 32 COO	min	0.000	max	0.000	mean	0.000
		CSR	min	0.000	max	0.000	mean	0.000		CSR	min	0.000	max	0.000	mean	0.000
		GPU 64 COO	min	7.220	max	7.510	mean	7.303		GPU 64 COO	min	12.310	max	12.550	mean	12.425
		CSR	min	17.540	max	20.710	mean	19.302		CSR	min	39.860	max	43.770	mean	42.075
		CPU PAR	min	1.384	max	1.593	mean	1.526		CPU PAR	min	1.735	max	1.945	mean	1.845
		H	min	9.681	max	9.706	mean	9.694		H	min	8.799	max	8.799	mean	8.799
	Row-Gradient	CPU COO	min	0.711	max	1.029	mean	0.746		CPU COO	min	0.689	max	0.710	mean	0.695
		CPU CSR	min	1.817	max	1.834	mean	1.827		CPU CSR	min	2.358	max	2.413	mean	2.392
		GPU 32 COO	min	0.000	max	0.000	mean	0.000		GPU 32 COO	min	0.000	max	0.000	mean	0.000
		CSR	min	0.000	max	0.000	mean	0.000		CSR	min	0.000	max	0.000	mean	0.000
		GPU 64 COO	min	7.110	max	7.270	mean	7.193		GPU 64 COO	min	11.430	max	11.770	mean	11.549
		CSR	min	16.530	max	18.590	mean	17.574		CSR	min	24.470	max	25.580	mean	24.785
		CPU PAR	min	1.390	max	1.574	mean	1.511		CPU PAR	min	1.758	max	1.896	mean	1.829
		H	min	10.612	max	10.659	mean	10.634		H	min	11.872	max	11.877	mean	11.875
	Row-Column-Permute	CPU COO	min	0.719	max	1.391	mean	0.756		CPU COO	min	0.716	max	0.775	mean	0.739
		CPU CSR	min	2.546	max	2.625	mean	2.611		CPU CSR	min	1.651	max	1.689	mean	1.675
		GPU 32 COO	min	0.000	max	0.000	mean	0.000		GPU 32 COO	min	0.000	max	0.000	mean	0.000
		CSR	min	0.000	max	0.000	mean	0.000		CSR	min	0.000	max	0.000	mean	0.000
		GPU 64 COO	min	7.190	max	7.320	mean	7.248		GPU 64 COO	min	12.100	max	12.410	mean	12.205
		CSR	min	17.500	max	18.640	mean	18.040		CSR	min	31.670	max	34.910	mean	33.370
		CPU PAR	min	1.465	max	1.573	mean	1.533		CPU PAR	min	2.079	max	2.286	mean	2.207
		H	min	10.041	max	10.046	mean	10.044		H	min	11.111	max	11.116	mean	11.113
	Row-Gradient	CPU COO	min	0.705	max	0.734	mean	0.718		CPU COO	min	0.715	max	1.021	mean	0.743
		CPU CSR	min	3.028	max	3.052	mean	3.045		CPU CSR	min	1.655	max	1.674	mean	1.666
		GPU 32 COO	min	0.000	max	0.000	mean	0.000		GPU 32 COO	min	0.000	max	0.000	mean	0.000
		CSR	min	0.000	max	0.000	mean	0.000		CSR	min	0.000	max	0.000	mean	0.000
		GPU 64 COO	min	0.000	max	0.000	mean	0.000		GPU 64 COO	min	11.340	max	11.560	mean	11.463
		CSR	min	0.000	max	0.000	mean	0.000		CSR	min	23.770	max	25.470	mean	24.489
		CPU PAR	min	1.528	max	1.602	mean	1.568		CPU PAR	min	2.056	max	2.172	mean	2.118
		H	min	7.380	max	7.380	mean	7.380		H	min	12.040	max	12.047	mean	12.043
	Row-Column-Permute	CPU COO	min	0.733	max	1.640	mean	0.777		CPU COO	min	0.677	max	0.785	mean	0.687
		CPU CSR	min	2.325	max	2.434	mean	2.369		CPU CSR	min	2.325	max	2.434	mean	2.369

OPF_3754.mtx Regular	GPU 32 COO	min	0.000	max	0.000	mean	0.000	Column-Gradient	GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000		CSR	min	0.000	max	0.000	mean	0.000
	GPU 64 COO	min	11.450	max	11.650	mean	11.538		GPU 64 COO	min	9.380	max	9.660	mean	9.464
	CSR	min	24.330	max	25.560	mean	25.008		CSR	min	15.770	max	19.090	mean	18.037
	CPU PAR	min	1.631	max	1.776	mean	1.709		CPU PAR	min	1.775	max	1.924	mean	1.868
	H	min	11.873	max	11.877	mean	11.875		H	min	10.205	max	10.233	mean	10.219
	CPU COO	min	0.726	max	0.774	mean	0.747		CPU COO	min	0.715	max	0.926	mean	0.757
	CPU CSR	min	2.898	max	2.919	mean	2.908		CPU CSR	min	1.729	max	1.802	mean	1.791
	GPU 32 COO	min	0.000	max	0.000	mean	0.000		GPU 32 COO	min	0.000	max	0.000	mean	0.000
	CSR	min	0.000	max	0.000	mean	0.000		CSR	min	0.000	max	0.000	mean	0.000
Row-Premute	GPU 64 COO	min	7.680	max	7.820	mean	7.766	GPU 64 COO	min	9.080	max	9.270	mean	9.158	
	CSR	min	25.070	max	29.030	mean	26.756	CSR	min	13.980	max	15.780	mean	14.938	
	CPU PAR	min	1.437	max	1.508	mean	1.471	CPU PAR	min	1.751	max	1.906	mean	1.846	
	H	min	8.393	max	8.393	mean	8.393	H	min	11.213	max	11.232	mean	11.222	
	CPU COO	min	0.714	max	1.574	mean	0.817	CPU COO	min	0.732	max	1.598	mean	0.785	
	CPU CSR	min	2.686	max	2.711	mean	2.699	CPU CSR	min	2.594	max	2.602	mean	2.599	
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	7.410	max	7.570	mean	7.484	GPU 64 COO	min	9.340	max	9.460	mean	9.394	
	CSR	min	19.600	max	21.190	mean	20.307	CSR	min	19.950	max	21.500	mean	20.544	
Row-Gradient	CPU PAR	min	1.443	max	1.505	mean	1.469	CPU PAR	min	1.326	max	1.374	mean	1.354	
	H	min	11.267	max	11.272	mean	11.269	H	min	10.059	max	10.062	mean	10.061	
	CPU COO	min	0.723	max	1.232	mean	0.775	CPU COO	min	0.759	max	0.795	mean	0.780	
	CPU CSR	min	1.672	max	1.691	mean	1.685	CPU CSR	min	2.479	max	2.565	mean	2.557	
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	7.600	max	7.760	mean	7.716	GPU 64 COO	min	5.490	max	5.650	mean	5.552	
	CSR	min	23.160	max	25.590	mean	24.304	CSR	min	16.700	max	19.460	mean	18.004	
	CPU PAR	min	1.675	max	1.736	mean	1.703	CPU PAR	min	1.456	max	1.523	mean	1.492	
	H	min	10.463	max	10.472	mean	10.468	H	min	7.132	max	7.132	mean	7.132	
Column-Gradient	CPU COO	min	0.726	max	1.431	mean	0.778	CPU COO	min	0.695	max	0.943	mean	0.726	
	CPU CSR	min	1.671	max	1.685	mean	1.679	CPU CSR	min	2.480	max	2.488	mean	2.485	
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	7.410	max	7.530	mean	7.467	GPU 64 COO	min	5.410	max	5.490	mean	5.453	
	CSR	min	18.140	max	20.350	mean	19.315	CSR	min	15.700	max	17.520	mean	16.678	
	CPU PAR	min	1.650	max	1.736	mean	1.699	CPU PAR	min	1.422	max	1.514	mean	1.474	
	H	min	11.393	max	11.401	mean	11.397	H	min	10.959	max	10.966	mean	10.963	
	CPU COO	min	0.711	max	1.458	mean	0.751	CPU COO	min	0.723	max	2.029	mean	0.990	
	CPU CSR	min	2.678	max	2.717	mean	2.700	CPU CSR	min	2.411	max	2.427	mean	2.421	
Row-Column-Permute	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	7.400	max	7.540	mean	7.471	GPU 64 COO	min	5.490	max	5.560	mean	5.534	
	CSR	min	19.560	max	21.150	mean	20.453	CSR	min	16.350	max	19.560	mean	17.784	
	CPU PAR	min	1.440	max	1.499	mean	1.467	CPU PAR	min	1.441	max	1.509	mean	1.477	
	H	min	11.266	max	11.272	mean	11.269	H	min	9.512	max	9.526	mean	9.520	
	CPU COO	min	0.754	max	1.829	mean	1.204	CPU COO	min	0.721	max	1.802	mean	0.871	
	CPU CSR	min	2.610	max	2.624	mean	2.618	CPU CSR	min	2.393	max	2.408	mean	2.404	
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000	
c-47.mtx Regular	GPU 64 COO	min	9.530	max	9.870	mean	9.640	GPU 64 COO	min	5.410	max	5.480	mean	5.453	
	CSR	min	23.990	max	25.910	mean	24.992	CSR	min	15.680	max	17.870	mean	16.540	
	CPU PAR	min	1.311	max	1.380	mean	1.357	CPU PAR	min	1.429	max	1.488	mean	1.468	
	H	min	8.364	max	8.364	mean	8.364	H	min	10.931	max	10.945	mean	10.938	
	CPU COO	min	0.740	max	0.885	mean	0.755	CPU COO	min	0.728	max	1.646	mean	1.037	
	CPU CSR	min	2.574	max	2.611	mean	2.597	CPU CSR	min	2.472	max	2.488	mean	2.480	
	GPU 32 COO	min	0.000	max	0.000	mean	0.000	GPU 32 COO	min	0.000	max	0.000	mean	0.000	
	CSR	min	0.000	max	0.000	mean	0.000	CSR	min	0.000	max	0.000	mean	0.000	
	GPU 64 COO	min	9.320	max	9.510	mean	9.397	GPU 64 COO	min	5.410	max	5.480	mean	5.449	
	CSR	min	19.960	max	21.190	mean	20.696	CSR	min	15.760	max	17.560	mean	16.654	
Row-Premute	CPU PAR	min	1.303	max	1.371	mean	1.345	CPU PAR	min	1.428	max	1.513	mean	1.474	
	H	min	10.059	max	10.062	mean	10.061	H	min	10.959	max	10.967	mean	10.963	
	CPU COO	min	0.723	max	0.984	mean	0.753	CPU COO	min	0.737	max	1.977	mean	1.431	
	CPU CSR	min	1.781	max	1.809	mean	1.803	CPU CSR	min	2.674	max	2.688	mean	2.681	

Row-Premute	GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 5.900 max 6.000 mean 5.954 CSR min 13.650 max 15.410 mean 14.657 CPU PAR min 1.468 max 1.521 mean 1.491 H min 9.234 max 9.234 mean 9.234	Row-Column-Permute	CPU COO min 0.712 max 0.971 mean 0.737 CPU CSR min 2.732 max 2.751 mean 2.743 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 17.720 max 18.070 mean 17.911 CSR min 29.600 max 30.500 mean 29.961 CPU PAR min 0.827 max 0.954 mean 0.913 H min 10.776 max 10.778 mean 10.777
Row-Gradient	CPU COO min 0.740 max 2.048 mean 1.121 CPU CSR min 2.777 max 2.798 mean 2.790 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 5.910 max 5.970 mean 5.944 CSR min 13.700 max 15.370 mean 14.541 CPU PAR min 1.468 max 1.546 mean 1.502 H min 10.250 max 10.255 mean 10.252	aft01.mtx Regular	CPU COO min 0.735 max 2.079 mean 1.069 CPU CSR min 3.132 max 3.154 mean 3.145 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 6.390 max 6.610 mean 6.457 CSR min 19.990 max 23.250 mean 21.820 CPU PAR min 1.746 max 1.865 mean 1.812 H min 7.811 max 7.811 mean 7.811
Column-Gradient	CPU COO min 0.743 max 1.991 mean 0.981 CPU CSR min 2.620 max 2.654 mean 2.646 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 5.840 max 5.910 mean 5.885 CSR min 13.130 max 17.040 mean 15.008 CPU PAR min 1.477 max 1.619 mean 1.569 H min 9.939 max 9.955 mean 9.948	Row-Premute	CPU COO min 0.714 max 1.648 mean 0.840 CPU CSR min 2.864 max 2.892 mean 2.883 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 6.280 max 6.380 mean 6.329 CSR min 17.980 max 19.700 mean 19.105 CPU PAR min 1.729 max 1.850 mean 1.782 H min 11.162 max 11.168 mean 11.165
Row-Column-Permute	CPU COO min 0.742 max 2.010 mean 1.124 CPU CSR min 2.789 max 2.800 mean 2.795 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 5.900 max 5.980 mean 5.941 CSR min 13.640 max 15.410 mean 14.556 CPU PAR min 1.462 max 1.540 mean 1.504 H min 10.250 max 10.253 mean 10.252	Row-Gradient	CPU COO min 0.735 max 1.806 mean 0.878 CPU CSR min 2.706 max 2.744 mean 2.726 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 6.390 max 6.500 mean 6.433 CSR min 19.780 max 22.870 mean 20.936 CPU PAR min 1.710 max 1.865 mean 1.785 H min 10.251 max 10.267 mean 10.257
Maragal_6.mtx Regular	CPU COO min 0.725 max 0.741 mean 0.729 CPU CSR min 2.345 max 2.409 mean 2.372 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 18.200 max 18.770 mean 18.357 CSR min 38.310 max 40.240 mean 39.477 CPU PAR min 0.789 max 0.813 mean 0.797 H min 9.930 max 9.930 mean 9.930	Column-Gradient	CPU COO min 0.728 max 1.792 mean 0.986 CPU CSR min 2.521 max 2.720 mean 2.703 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 6.280 max 6.370 mean 6.327 CSR min 18.000 max 19.720 mean 19.040 CPU PAR min 1.649 max 1.741 mean 1.702 H min 11.113 max 11.121 mean 11.117
Row-Premute	CPU COO min 0.709 max 0.779 mean 0.715 CPU CSR min 2.675 max 2.715 mean 2.696 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 17.810 max 18.030 mean 17.935 CSR min 29.650 max 30.580 mean 30.109 CPU PAR min 0.857 max 0.940 mean 0.904 H min 10.777 max 10.779 mean 10.778	Row-Column-Permute	CPU COO min 0.714 max 1.525 mean 0.957 CPU CSR min 2.876 max 2.892 mean 2.884 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 6.280 max 6.370 mean 6.322 CSR min 17.960 max 19.670 mean 18.670 CPU PAR min 1.667 max 1.754 mean 1.710 H min 11.162 max 11.168 mean 11.165
Row-Gradient	CPU COO min 0.710 max 1.566 mean 0.755 CPU CSR min 2.042 max 2.159 mean 2.120 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 18.460 max 18.960 mean 18.665 CSR min 25.650 max 27.330 mean 26.549 CPU PAR min 2.257 max 2.612 mean 2.416 H min 11.251 max 11.301 mean 11.285	TSOPF_RS_b39_c7.mtx Regular	CPU COO min 0.771 max 0.793 mean 0.780 CPU CSR min 3.219 max 3.232 mean 3.227 GPU 32 COO min 0.000 max 0.000 mean 0.000 CSR min 0.000 max 0.000 mean 0.000 GPU 64 COO min 11.070 max 11.200 mean 11.142 CSR min 37.050 max 42.100 mean 39.040 CPU PAR min 1.910 max 2.027 mean 1.982 H min 7.304 max 7.304 mean 7.304
Column-Gradient	CPU COO min 0.711 max 0.743 mean 0.725 CPU CSR min 2.036 max 2.161 mean 2.110 GPU 32 COO min 0.000 max 0.000 mean 0.000	Row-Premute	CPU COO min 0.701 max 0.722 mean 0.707 CPU CSR min 2.931 max 2.952 mean 2.942 GPU 32 COO min 0.000 max 0.000 mean 0.000

Row-Gradient	CSR min 0.000 max 0.000 mean 0.000				
	GPU 64 COO	min	10.860	max 11.030	mean 10.928
	CSR min 28.730 max 30.880 mean 29.483				
	CPU PAR	min	1.760	max 1.922	mean 1.851
	H	min	10.537	max 10.541	mean 10.539
	CPU COO	min	0.747	max 0.808	mean 0.757
	CPU CSR	min	2.606	max 2.648	mean 2.624
	GPU 32 COO	min	0.000	max 0.000	mean 0.000
	CSR min 0.000 max 0.000 mean 0.000				
Column-Gradient	GPU 64 COO	min	10.850	max 11.120	mean 10.999
	CSR min 33.910 max 37.600 mean 35.909				
	CPU PAR	min	2.154	max 2.245	mean 2.203
	H	min	9.636	max 9.646	mean 9.642
	CPU COO	min	0.718	max 1.693	mean 0.802
	CPU CSR	min	2.502	max 2.585	mean 2.547
	GPU 32 COO	min	0.000	max 0.000	mean 0.000
	CSR min 0.000 max 0.000 mean 0.000				
	GPU 64 COO	min	10.700	max 10.990	mean 10.804
Row-Column-Permute	CSR min 27.230 max 29.380 mean 28.488				
	CPU PAR	min	2.128	max 2.227	mean 2.172
	H	min	11.131	max 11.222	mean 11.208
	CPU COO	min	0.709	max 0.726	mean 0.716
	CPU CSR	min	2.917	max 2.958	mean 2.940
	GPU 32 COO	min	0.000	max 0.000	mean 0.000
	CSR min 0.000 max 0.000 mean 0.000				
	GPU 64 COO	min	10.840	max 11.030	mean 10.930
	CSR min 28.780 max 30.810 mean 29.578				
	CPU PAR	min	1.757	max 1.834	mean 1.792
	H	min	10.537	max 10.540	mean 10.539