Analysis of the construction of similarity matrices on multi-core and many-core platforms using different similarity metrics

Uxía Casal, Jorge González-Domínguez, and María J. Martín

Universidade da Coruña, CITIC, Computer Architecture Group, Campus de Elviña, 15071 A Coruña, Spain {uxia.casal.baldomir,jgonzalezd,mariam}@udc.es

Abstract. Similarity matrices are 2D representations of the degree of similarity between points of a given dataset which are employed in different fields such as data mining, genetics or machine learning. However, their calculation presents quadratic complexity and, thus, it is specially expensive for large datasets. MPICorMat is able to accelerate the construction of these matrices through the use of a hybrid parallelization strategy based on MPI and OpenMP. The previous version of this tool achieved high performance and scalability, but it only implemented one single similarity metric, the Pearson's correlation. Therefore, it was suitable only for those problems where data are normally distributed and there is a linear relationship between variables. In this work, we present an extension to MPICorMat that incorporates eight additional metrics for similarity so that the users can choose the one that best adapts to their problem. The performance and energy consumption of each metric is measured in two platforms: a multi-core platform with two Intel Xeon Sandy-Bridge processors and a many-core Intel Xeon Phi KNL. Results show that MPICorMat executes faster and consumes less energy on the many-core architecture. The new version of MPICorMat is publicly available to download from its website: https://sourceforge.net/projects/mpicormat/

Keywords: Similarity Matrix · High Performance Computing · Intel Xeon Phi · Performance Evaluation · Energy Consumption

1 Introduction

The construction of similarity matrices is a fundamental step for many applications of different areas such as bioinformatics, data mining, text mining or machine learning. For instance, they are usually necessary when constructing gene co-expression networks, as they can represent the similarity between genes. However, the calculation of these 2D matrices is highly time-consuming due to its quadratic complexity. In the Big Data era the size of datasets is continuously increasing in many fields, and thus finding fast and scalable solutions is a highly important task.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

We have recently developed the tool MPICorMat [3], a High Performance Computing (HPC) framework that accelerates the construction of similarity matrices with Message Passing Interface (MPI) and OpenMP routines. It presents high performance and scalability on multi-core clusters, but has the following limitations:

- It only includes Pearson's correlation as similarity metric, as it is suitable for data with linear relationship (very common in genetic scenarios, the focus of the previous work). However, there are in literature a number of metrics to build similarity matrices and the choice of the best one depends on the application field and the input data. For instance, if the linearity hypothesis cannot be assumed, Spearman's or Kendall's tau-b correlations may be more useful than Pearson's because they identify both linear and non-linear relationships [13].
- The calculation of the similarity metric for each pair of attributes relies on the GNU Scientific Library [4], which forces the users to install and tune that library in their systems.
- The experimental evaluation is limited to performance measures in traditional multi-core clusters, without taking into account energy consumption and/or the use of many-core platforms.

This work overcomes these limitations by presenting a new version of MPICorMat (v3) that includes eight additional metrics. The choice of the metric is made by the users through a command line parameter in order to adapt the execution to the characteristics of their data. The eight metrics were implemented using C++, MPI and OpenMP in order to avoid any dependency with external libraries and thus improve the portability of the tool. We also provide a highly detailed experimental evaluation that compares the performance of the different metrics on a multi-core platform and an Intel Xeon Phi Knights Landing (KNL) many-core system, not only in terms of performance but also of energy consumption.

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 describes the behavior of the extended application and the metrics included. Section 4 shows the results of the experimental evaluation in both the multi-core and many-core systems. Finally, conclusions are discussed in Section 5.

2 Related work

There are in the literature a number of works than compare the behavior of different similarity measures for the reconstruction of gene co-expression networks [5, 15, 17]. These studies show that the choice of the most appropriate measure depends on the nature of the gene interactions to be analyzed.

On the other hand, similarity matrices also play a crucial role in other fields such as clustering, image retrieval, or recommending systems. For instances, in [11] authors compare seven popular similarity measures for the clustering of

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

patients. They include the Euclidean distance, as well as the Pearson, Spearman and Kendall correlations, among others. Authors conclude that an absolute best similarity measure does not exist, but it strongly depends on data.

However, most of the available software tools for the calculation of similarity matrices on parallel architectures focus on only one similarity metric. For instance, TINGe [20] and CUDA-MPI [12] are parallel approaches based on Mutual Information (MI) for clusters (implemented with MPI), and GPUs (implemented with CUDA), respectively. In [8] TINGe is adapted for the first generation of the Intel Xeon Phi (KNC) architecture. The construction of Pearson's correlation-based similarity matrices was addressed for MPICorMat [3], LightPCC [7] and FastGCN [6] for multicore clusters (implemented with MPI and OpenMP), the Intel Xeon Phi KNC coprocessor and NVIDIA GPUs, respectively.

MPICorMat_v3, in contrast, allows the user to choose among several similarity metrics to better adapt to the characteristics of the problem in hand. Moreover, up to our knowledge, our performance evaluation is the first one focused on using the KNL generation of Intel Xeon Phi to accelerate the construction of similarity matrices.

3 MPICorMat version 3

As previously explained, MPICorMat is a parallel tool to accelerate the construction of similarity matrices on HPC systems. It receives as input a file that contains a 2D matrix with dimensions $n \times m$, where n is the number of attributes and m the number of samples. It returns a file with an $n \times n$ similarity matrix with the similarity values for each pair of attributes.

The third version of this tool increments its usefulness by including eight additional similarity metrics (besides the Pearson's correlation already available in the previous versions of the tool). The users should indicate its desired metric using a command line parameter. Information about the input parameters, as well as installation and execution instructions, are available in the reference manual of the tool.

The implementations of the nine metrics have been integrated into the parallel approach already available in previous versions of MPICorMat, as it was proved efficient in our previous work [3]. MPICorMat follows a hybrid parallel approach with MPI and OpenMP that is able to exploit the computational capabilities of multi-core clusters, with hybrid distributed/shared memory architecture. As will be shown in Section 4, the focus of the experimental evaluation of this work consists in testing the suitability of this parallel implementation in the Intel Xeon Phi KNL many-core processor, compared to an Intel multi-core system. Only the OpenMP parallelization is necessary for both platforms, as they are shared-memory machines. OpenMP is a parallel programming interface based on compiler directives that follows a fork-join model, where a master or parent thread creates a number of slaves or children that are able to access to the same shared memory and perform different tasks to complete a work.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

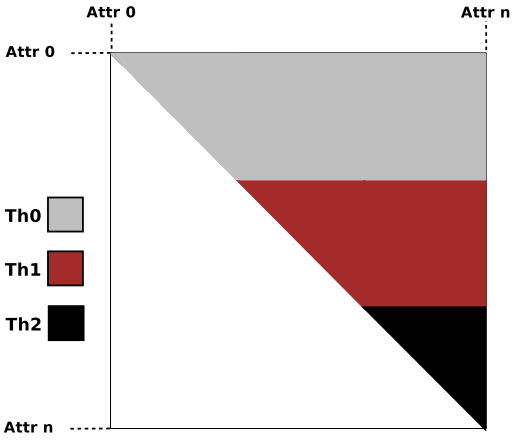

Fig. 1. Not efficient distribution of pairs among threads.

As the similarity metric must be calculated for all gene pairs, the MPICorMat workload can be seen as a 2D matrix, where each point represents one pair of attributes. Only half of the matrix (upper or lower triangular) must be calculated as all metrics are symmetric. Concretely $\frac{n \cdot (n-1)}{2}$ pairs. MPICorMat divides the workload (pairs) among the threads, which do not need any synchronization as computation is completely independent among pairs. Pairs are assigned by rows (the whole row to the same thread) in order to reuse data (one attribute is repeated in all the pairs of the row). However, due to the triangular nature of the problem, the most intuitive static block distribution, with the same number of rows per thread, would lead to unbalanced workload (some rows have more pairs than other, as can be seen in Figure 1). Instead, MPICorMat uses a dynamic OpenMP distribution, where each (still not computed) row is assigned to a thread once it has finished all its previously assigned work. We refer to [3] for more information.

Algorithm 1 shows the pseudocode of the parallel OpenMP implementation in MPICorMat, the one tested in the experimental evaluation of Section 4. After reading the file with the input values for each attribute and sample (Line 1) and allocating memory for the output similarity matrix (Line 2), several OpenMP threads are launched to parallelize the loop that traverses the rows (Lines 3 and 4). As previously explained, each thread is in charge of a complete row. The thread starts calculating the position of the row in the output triangular matrix (Line 5). Then, it calculates the metric for all the pairs of the row (Lines 6 and 7). Next, the parallel region finishes and the output is written by the master thread (Line 8).

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

Algorithm 1: Pseudo-code of the OpenMP parallelization in MPICorMat.

```
1 Read input matrix M with the values of the attributes
2 Initialize matrix of scores S
3 #pragma omp parallel for schedule(dynamic)
4 for each row i from 0 to n do
5 |   rowPos = CalculatePos(i)
6 |   for each column j from i to n do
7 |   |   S[rowPos + j − i] := CalculateMetric(i, j)
  |   end
  end
8 Write S in the output file
```

3.1 Similarity metrics included in MPICorMat_v3

A different implementation of `CalculateMetric()` (Line 7 in Algorithm 1) is performed for each metric. The following nine similarity metrics are available for MPICorMat users since its third version.

Pearson's correlation. It measures the strength of the linear relationship between two random variables. The value of the correlation is between -1 and 1. A correlation close to 1 or -1 indicates that the relationship is almost perfectly linear while a value close to 0 indicates that the two variables are uncorrelated. The Pearson's correlation assumes the data are normally distributed and there is a linear relationship between the two variables. It is sensitive to outliers and requires the data to be measured on interval or ratio scale. Assume that X and Y are two random variables with n observations (x_i, y_i with $i = 1, 2, ..., n$) and $\overline{x}$ and $\overline{y}$ are the means of X and Y, respectively. Then, Pearson's correlation is defined as:

$$\frac{\sum_{i=1}^{n}(x_i - \overline{x}) \cdot (y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2 \cdot \sum_{i=1}^{n}(y_i - \overline{y})^2}} \tag{1}$$

Spearman's correlation. It is equal to the Pearson's correlation between the rank values of the variables, being the rank value of an observation its relative position within all the values of the variable. While Pearson's correlation assesses linear relationships, Spearman's correlation assesses monotonic relationships (whether linear or not). It takes values between -1 and 1. A positive correlation implies that the ranks of both variables increase together, while a negative correlation implies that the rank of one variable increases as the rank of the other decreases. A definition of the Spearman's correlation able to deal with tied ranks (elements that have the same rank value) is:

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

$$\frac{\frac{1}{n} \cdot \sum_{i=1}^{n} (R(x_i) - \overline{R(x)}) \cdot (R(y_i) - \overline{R(y)})}{\sqrt{(\frac{1}{n} \cdot \sum_{i=1}^{n} (R(x_i) - \overline{R(x)})^2) \cdot (\frac{1}{n} \cdot \sum_{i=1}^{n} (R(y_i) - \overline{R(y)})^2)}} \tag{2}$$

where $R(x_i)$ and $R(y_i)$ are the ranks of the observation i in the variables X and Y, respectively, while $\overline{R(x)}$ and $\overline{R(y)}$ are the means of the ranks. The procedure to calculate the ranks usually consists in sorting the observations of the variable.

Euclidean distance. This is probably the simplest metric, indicating the straight-line distance between two points in Euclidean space. The Euclidean distance between two attributes X and Y, with x_i and y_i denoting their value for sample i, is measured as:

$$\sqrt{\sum_{i=1}^{n} ((x_i - y_i)^2} \tag{3}$$

Mutual information (MI). It quantifies the amount of information that one random variable provides about another. MI can only take positive values. High MI indicates a large reduction in uncertainty, while low MI indicates a small reduction in uncertainty, and MI equal to 0 means that the variables are independent. MI is a metric that only works over discrete values. If the input data are real values (either in simple or double precision), a preliminary step that discretizes the values, grouping similar elements into the same bucket, is required. The number of buckets is indicated by the user as an argument of the application through the command line. The accuracy of the metric usually increases with the number of buckets, but also its complexity. Mi is defined as:

$$\sum_{i=1}^{n} \sum_{i=j}^{n} p(x_i, y_j) \log_2 \frac{p(x_i, y_j)}{p(x_i) \cdot p(y_j)} \tag{4}$$

where $p(x_i)$ and $p(y_j)$ are the probabilities of the buckets that contain the values x_i and y_i, and $p(x_i, y_j)$ is the joint probability of the buckets associated to x_i and y_j.

Kendall's tau-b. It is a non-parametric metric of association based on the number of concordances and discordances in paired observations. It is an alternative method to Spearman's correlation, i.e., it also identifies monotonic relationships. Suppose two pairs (x_i,y_i) and (x_j,y_j), they are concordant if they are in the same order with respect to each variable. That is, if $x_i < x_j$ and $y_i < y_j$, or if $x_i > x_j$ and $y_i > y_j$. Otherwise, they are discordant. The value of this coefficient ranges from -1 (one ranking always reverses the other) to 1 (the ranks of the two attributes are the same). If the two variables are independent,

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

the value is approximately equal to 0. Assume that P is the number of concordant pairs, Q is the number of discordant pairs, X_0 the number of tied pairs on X and Y_0 the number of tied pairs on Y. Then, Kendall's tau-be is defined as:

$$\frac{P-Q}{\sqrt{(P+Q+X_0)\cdot(P+Q+Y_0)}} \tag{5}$$

Goodman & Kruskal gamma coefficient (G&K). It is another widely-used rank-based coefficient that ranges between -1 and 1. As Kendall's tau-b, a value -1 indicates 100% perfect inversion, value 1 indicates 100% perfect agreement, and value 0 indicates the absence of association. It is defined as:

$$\frac{P-Q}{P+Q} \tag{6}$$

Maximal information correlation (MIC). It is based on the idea that if a relationship between two variables exists, then a grid that partitions the data to encapsulate that relationship can be drawn on the scatterplot of the two variables [10]. Its value ranges between 0 and 1 and it takes the value 0 if the variables are independent. The MIC for two attributes X and Y is defined as:

$$\frac{MI(X,Y)}{H(X)} \tag{7}$$

where $MI(X,Y)$ is the mutual information between the variables X and Y and it can be obtained from Equation 4, and $H(X)$ is the entropy of the attribute X, which can be calculated as follows (being $p(x_i)$ the probability of the bucket that contains the variable x_i):

$$-\sum_{i=1}^{n} p(x_i)\cdot\log_2(p(x_i)) \tag{8}$$

Hoeffding D test. This metric approximates a weighted sum over observations in order to test the independence of two datasets. In this work, each attribute is seen as a dataset. The statistic D is defined as:

$$30\cdot\frac{(n-2)\cdot(n-3)\cdot D_1+D_2-2\cdot(n-2)\cdot D_3}{n\cdot(n-1)\cdot(n-2)\cdot(n-3)\cdot(n-4)} \tag{9}$$

where:

$$D_1=\sum_{i=1}^{n}(Q_i-1)\cdot(Q_i-2) \tag{10}$$

$$D_2=\sum_{i=1}^{n}(R(x_i)-1)\cdot(R(x_i)-2)\cdot(R(y_i)-1)\cdot(R(y_i)-2) \tag{11}$$

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

$$D_3 = \sum_{i=1}^{n} (R(x_i) - 2) \cdot (R(x_i) - 2) \cdot (Q_i - 1) \tag{12}$$

being $R(x_i)$ and $R(y_i)$ the ranks as in Spearman's correlation and Q_i the bivariate rank, which refers to the number of points j ($j = 1, 2, ..., n$) with both x_j and y_j values lower than the ith point. Hoeffding's D lies on the interval [-0.5,1], with larger values indicating a stronger relationship.

Weighted rank correlation. It is a variation of the Spearman's rank correlation but giving weight to the distance between two ranks by using a linear function of those ranks (more weight to higher ranks than to lower ones). Assume that $R(x_i)$ and $R(y_i)$ are the ranks as in Spearman's correlation, then the weighted rank correlation metric can be calculated as:

$$1 - \frac{90}{g(n)} \cdot \sum_{i=1}^{n} (R(x_i) - R(y_i))^2 \cdot (2 \cdot (n+1) - (R(x_i) + R(y_i)))^2 \tag{13}$$

where:

$$g(n) = n \cdot (n-1) \cdot (n+1) \cdot (2 \cdot +1) \cdot (8 \cdot n + 11) \tag{14}$$

4 Experimental evaluation

Three datasets, with a different number of attributes and samples, were used in the evaluation of the nine metrics included in the third version of MPICorMat. The datasets were downloaded from the Geo Expression Omnibus (GEO) Dataset Browser available at the National Center for Biotechnology Information (NCBI) website [9]. Table 1 shows their characteristics. As they contain genetic information, the attributes represent genes of a population.

Although MPICorMat includes support for MPI parallelization, all the experiments were carried out with only one MPI process and several OpenMP threads, as the two platforms are shared memory architectures. The scalability of the hybrid MPI/OpenMP parallel approach has not been tested again in a multi-core cluster as it has not been modified since [3].

4.1 Performance evaluation on an Intel Xeon Phi KNL

Knights Landing (KNL) is the code name for the second-generation Intel Xeon Phi product family [14]. It is a many-core processor that delivers massive thread and data parallelism with high memory bandwidth. Concretely, it provides features such as four threads per core, deeper out-of-order buffers, higher cache bandwidth, new instructions, better reliability, larger translation look-aside buffers (TLBs), and larger caches. Additionally, it introduces the new Advanced Vector Extensions instruction set, AVX-512 [19], in order to fully exploit its 512-bit

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

vector registers, which can hold 16 single precision or 8 double precision floating-point numbers. In this project, we used the Intel Xeon Phi KNL processor 7210. It has 64 active cores at 1.30 GHz, allows up to four threads per core (256 total threads) and it is configured in the quadrant clustering [16] and the flat memory modes [1]. MPICorMat has been compiled with the Intel ICPC compiler version 18.0.3 activating the automatic vectorization with Intel AVX-512 instructions (`-xMIC-AVX512` flag). Remark that all the runtimes shown in this section were obtained with the many-core system in exclusive mode, i.e., no other works were executed at the same time.

Table 1. Characteristics of the datasets used for evaluation.

Name	Number of attributes	Number of samples
GDS5037	41,000	108
GDS3795	61,170	160
GDS3244	54,675	200

Figure 2 shows the runtime for the three datasets, the nine metrics and different number of threads (from 64 threads or one thread per core to 256 threads or four threads per core). 32 buckets are used for MI and MIC. The first conclusion that can be obtained is that the runtime heavily depends on the metric. Pearson's correlation and Euclidean distance are the simplest metrics, while Kendall's tau-b is the most complex one. The results shown in these graphs also indicate that hyperthreading is beneficial for MPICorMat. Using two threads per core reduces the runtime on average 1.41, 1.47 and 1.45 times compared to the single-thread execution with the GDS5037, GDS3795 and GDS3244 datasets, respectively. This average speedup increases to 1.71, 1.77, 1.80 if fully exploiting the hyperthreading, with four threads per core. 256 threads will be used from now on for all the experiments in the Intel Xeon Phi KNL, as this configuration obtains the best runtime for all scenarios (combination of dataset and metric).

As previously mentioned, the runtimes shown in the graphs of Figure 2 were obtained by activating the automatic vectorization with the Intel AVX-512 instructions. Table 2 shows the speedups compared to an execution with 256 threads but without vectorization (`-no-vec` flag in the compiler). Its impact depends on the characteristics of the metrics, being especially beneficial for the ranking procedure necessary for Spearman's correlation, Hoeffding D test and weighted rank correlation (see Section 3.1).

4.2 Performance and energy consumption comparison between Intel architectures

The experimental evaluation has also been performed in an Intel multi-core platform in order to compare its performance to the Intel Xeon Phi KNL many-core. Concretely, a machine with two eight-core Intel Xeon E5-2660 Sandy Bridge-EP

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

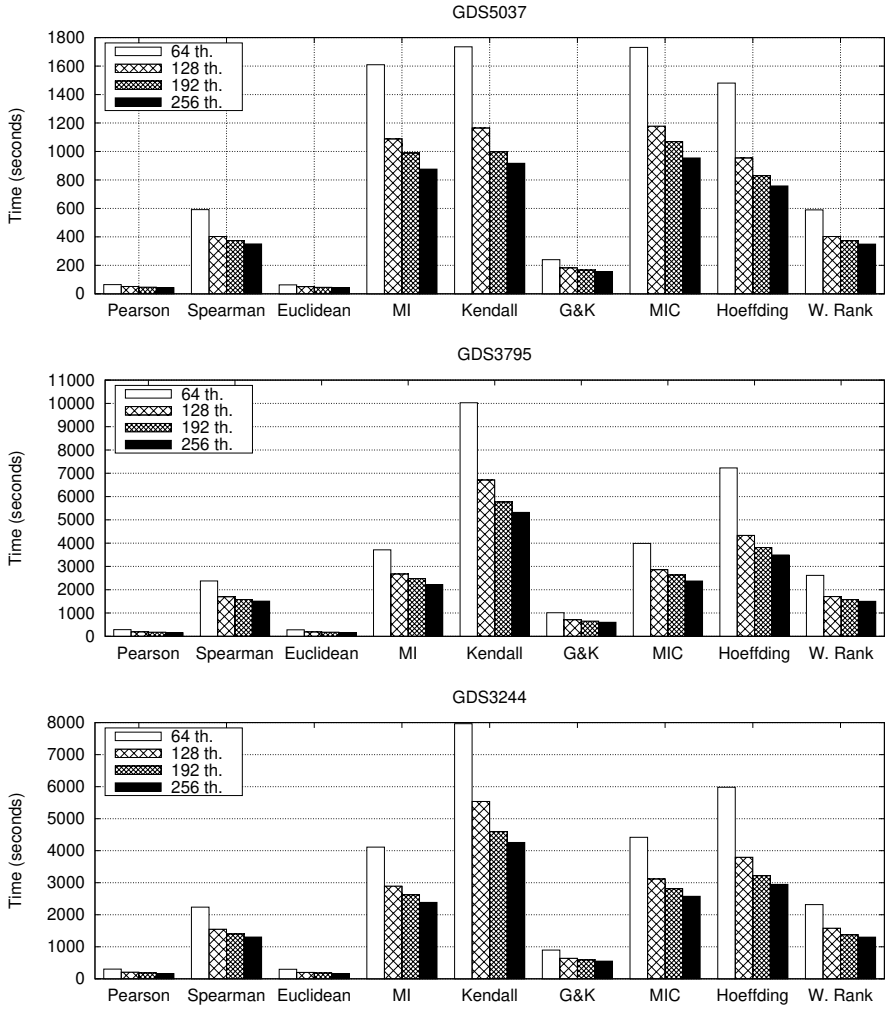

Fig. 2. Runtime of MPICorMat_v3 on the Intel Xeon Phi KNL for different metrics and number of threads. Automatic vectorization with Intel AVX-512 instructions has been used.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

Table 2. Speedup obtained thanks to the use of automatic vectorization with Intel AVX-512 instructions in the Xeon Phi KNL, compared to not vectorized versions of the metrics. All executions are carried out with 256 threads.

	GDS5037	**GDS3795**	**GDS3244**
Pearson	1.50	1.43	1.41
Spearman	10.59	14.65	13.63
Euclidean	1.36	1.31	1.30
MI	3.66	4.36	4.26
Kendall	1.46	1.49	1.49
G&K	1.56	1.90	1.81
MIC	3.55	4.27	4.15
Hoeffding	7.33	9.52	9.06
Weighted rank	10.59	14.71	13.66

Table 3. Speedup of the execution of MPICorMat_v3 in the Xeon Phi KNL for each metric (with 256 threads and automatic vectorization using Intel AVX-512 instructions) compared to the execution on the Sandy Bridge-based multi-core platform (with 16 threads and automatic AVX 256-bit vectorization).

	GDS5037	**GDS3795**	**GDS3244**
Pearson	1.54	1.13	1.15
Spearman	3.92	4.94	4.74
Euclidean	1.73	1.07	1.34
MI	5.63	6.79	6.59
Kendall	2.97	2.91	2.82
G&K	1.56	1.50	1.47
MIC	5.41	6.58	6.36
Hoeffding	4.36	5.56	5.39
Weighted rank	3.93	4.96	4.75

processors (i.e., a total of 16 cores) and 64 GB of memory. The Intel ICPC compiler has also been used in this machine (in this case, version 18.0.1) activating the automatic vectorization with AVX instructions. Remark that the impact of vectorization should be lower than in the Intel Xeon Phi KNL as the length of the vector registers is 256 bits, instead of 512 bits as in the many-core.

The execution in the Intel Xeon Phi KNL is faster than in the Sandy Bridge-based multi-core system using 16 threads (one per core) for all combinations of dataset and metric. Table 3 shows the speedup for each scenario. The highest the speedup, the fastest the execution in the many-core compared to the multi-core systems (speedup equal to 1 would mean same execution time). The magnitude of the benefit thanks to running on the Intel Xeon Phi KNL depends again on the metric. Speedups are higher for those metrics that require a ranking of the data (such as Spearman's correlation, Hoeffding D test and weighted rank correlation, with average speedups of 4.53, 5.10 and 4.55, respectively), as well as for those based on probabilities (MI and MIC, with average speedups of 6.34 and 6.12, respectively).

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

Nowadays, reduction of energy consumption is key in order to develop and maintain large HPC infrastructures. In this sense, many-core systems are expected to accelerate the execution at the same time that save energy. The Performance API (PAPI) analysis library [2, 18], together with the Running Average Power Limit (RAPL) of the Intel architectures, has been used to measure and report energy values on both platforms when calculating the similarity matrices with different metrics. Figure 3 shows the energy consumption (in Joules) for each metric and platform using the GDS5037 dataset. On average, the Intel Xeon Phi KNL consumes 4.46 times less energy than the multi-core platform, reaching factors of 5.31 and 5.85 for MI and MIC, respectively.

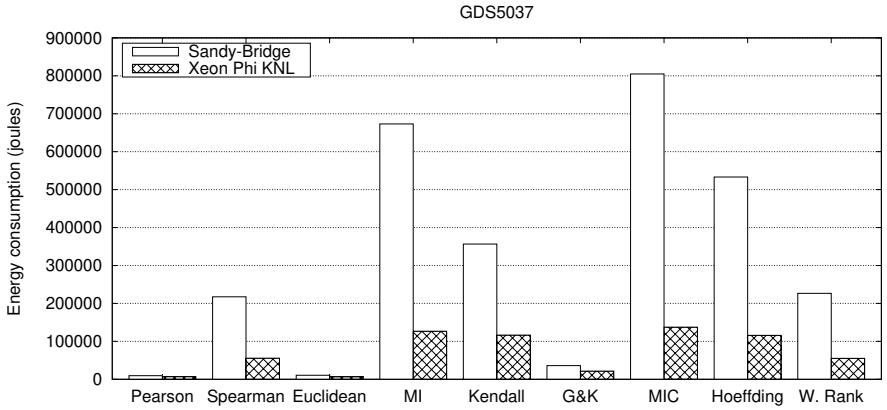

Fig. 3. Energy consumed by MPICorMat_v3 on the Intel Xeon Phi KNL (with 256 threads and automatic vectorization using Intel AVX-512 instructions) and the Sandy Bridge-based multi-core machine (with 16 threads and automatic AVX 256-bit vectorization) for different metrics.

5 Conclusions

The construction of similarity matrices is a bottleneck for many algorithms of different areas due to its quadratic complexity with the number of attributes. MPICorMat is a publicly available tool that helps to alleviate this problem by efficiently exploiting HPC resources. However, previous versions of this tool were only able to calculate similarity matrices based on Pearson's correlation, which limited its interest for many researchers. In this work, we have presented a new version of MPICorMat that includes a total of nine different similarity metrics so that the users can choose the one most suitable for their applications. The implementations of the new metrics were integrated into the framework of MPICorMat so all of them can benefit from the parallel implementation.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

The experimental evaluation has focused on testing the adequacy of the metric implementations to the hardware characteristics of the Intel Xeon Phi KNL, as the scalability in multi-core clusters had been effectively tested in a previous work [3]. The use of this and other kind of many-core accelerators (such as GPUs) is gaining popularity in the last years as they provide high performance at low power consumption. Our experimental evaluation using three datasets from genetic scenarios with different characteristics has led to several conclusions:

- The best performance is obtained in all cases with four threads per core (256 threads per Intel Xeon Phi KNL). For instance, the runtime of applying Kendall's tau-b metric to the GDS3422 dataset is reduced from around 29 minutes with only one thread per core, to around 15 minutes when hyperthreading with four threads per core is used.
- Automatic vectorization with Intel AVX-512 instruction should be applied in order to improve performance. The magnitude of this performance improvement depends on the metric, varying from an overall speedup of 1.32 for Euclidean distance to 12.99 for weighted rank correlation.
- Execution times in the Intel Xeon Phi KNL are lower than in a multi-core platform with two octa-core Sandy Bridge processors for every combination of metric and dataset. The overall performance improvement is 3.74, being more significant for metrics based on probabilities such as MI and MIC, with an overall speedup of 6.34 and 6.12, respectively.
- The energy consumption is lower in the many-core architecture for all the experiments, needing on average 4.46 times less energy.

As future work, we plan to implement a GPU version of the code and compare the results with the ones obtained in the Intel Xeon Phi architecture.

Acknowledgments

This work was supported by the Ministry of Economy, Industry and Competitiveness of Spain and FEDER funds of the European Union [grant TIN2016-75845-P (AEI/FEDER/UE)], as well as by Xunta de Galicia (Centro Singular de Investigacion de Galicia accreditation 2016-2019, ref. EDG431G/01).

References

1. Asai, R.: MCDRAM as High-Bandwidth Memory (HBM) in Knights Landing Processors: Developers Guide. Colfax Research (2016)
2. Browne, S., Dongarra, J., Garner, N., Ho, G., Mucci, P.: A Portable Programming Interface for Performance Evaluation on Modern Processors. The International Journal of High Performance Computing Applications **14**(3), 189–204 (2000)
3. González-Domínguez, J., Martín, M.J.: Fast Parallel Construction of Correlation Similarity Matrices for Gene Co-Expression Networks on Multicore Clusters. In: 17th International Conference on Computer Science (ICCS'17). vol. 108, pp. 485–494. Zurich, Switzerland (2017)

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13

4. Gough, B.: GNU Scientific Library Reference Manual - Third Edition. Network Theory Ltd. (2009)
5. Kumari, S., Nie, J., Chen, H.S., Ma, H., Stewart, R., Li, X., Lu, M.Z., Taylor, W.M., Wei, H.: Evaluation of Gene Association Methods for Coexpression Network Construction and Biological Knowledge Discovery. PloS one **7**(11), e50411 (2012)
6. Liang, M., Zhang, F., Jin, G., Zhu, J.: FastGCN: a GPU Accelerated Tool for Fast Gene Co-Expression Networks. PloS one **10**(1), e0116776 (2015)
7. Liu, Y., Pan, T., Aluru, S.: Parallel Pairwise Correlation Computation on Intel Xeon Phi Clusters. In: 28th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD'16). pp. 141–149. Los Angeles, CA, USA (2016)
8. Misra, S., Pamnany, K., Aluru, S.: Parallel Mutual Information Based Construction of Genome-Scale Networks on the Intel Xeon Phi Coprocessor. IEEE/ACM Transactions on Computational Biology and Bioinformatics **12**(5), 1008–1020 (2015)
9. National Center for Biotechnology Information (NCBI): Geo Expression Omnibus (GEO) Dataset Browser. https://www.ncbi.nlm.nih.gov/sites/GDSbrowser (Last visited: December 2018)
10. Reshef, D.N., Reshef, Y.A., Finucane, H.K., Grossman, S.R., McVean, G., Turnbaugh, P.J., Lander, E.S., Mitzenmacher, M., Sabeti, P.C.: Detecting Novel Associations in Large Data Sets. Science **334**(6062), 1518–1524 (2011)
11. Serra, A., Greco, D., Tagliaferri, R.: Impact of Different Metrics on Multi-View Clustering. In: 2015 IEEE International Joint Conference on Neural Networks (IJCNN). pp. 1–8 (2015)
12. Shi, H., Schmidt, B., Liu, W., Müller-Wittig, W.: Parallel Mutual Information Estimation for Inferring Gene Regulatory Networks on GPUs. BMC Research Notes **4**(1), 189 (2011)
13. de Siqueira Santos, S., Takahashi, D.Y., Nakata, A., Fujita, A.: A Comparative Study of Statistical Methods Used to Identify Dependencies between Gene Expression Signals. Briefings in bioinformatics **15**(6), 906–918 (2013)
14. Sodani, A., Gramunt, R., Corbal, J., Kim, H.S., Vinod, K., Chinthamani, S., Hutsell, S., Agarwal, R., Liu, Y.C.: Knights Landing: Second-Generation Intel Xeon Phi Product. IEEE Micro **36**(2), 34–46 (2016)
15. Song, L., Langfelder, P., Horvath, S.: Comparison of Co-Expression Measures: Mutual Information, Correlation, and Model Based Indices. BMC Bioinformatics **13**(1), 328 (2012)
16. Vladimirov, A., Asai, R.: Clustering Modes in Knights Landing Processors: Developers Guide. Colfax International (2016)
17. Wang, Y.R., Huang, H.: Review on Statistical Methods for Gene Network Reconstruction Using Expression Data. Journal of Theoretical Biology **362**, 53–61 (2014)
18. Weaver, V.M., Johnson, M., Kasichayanula, K., Ralph, J., Luszczek, P., Terpstra, D., Moore, S.: Measuring Energy and Power with PAPI. In: 41st International Conference on Parallel Processing Workshops (ICPPW'12). pp. 262–268 (2012)
19. Zhang, B.: Guide to Automatic Vectorization with Intel AVX-512 Instructions in Knights Landing Processors. Colfax International (2016)
20. Zola, J., Aluru, M., Sarje, A., Aluru, S.: Parallel Information-Theory-Based Construction of Genome-Wide Gene Regulatory Networks. IEEE Transactions on Parallel and Distributed Systems **21**(12), 1721–1733 (2010)

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_13