*Int. J. Appl. Math. Comput. Sci., 2019, Vol. 29, No. 3, 467–476*
DOI: 10.2478/amcs-2019-0034

# EFFICIENT ASTRONOMICAL DATA CONDENSATION USING APPROXIMATE NEAREST NEIGHBORS

SZYMON ŁUKASIK [a,b,*], KONRAD LALIK [a], PIOTR SARNA [a], PIOTR A. KOWALSKI [a,b], MAŁGORZATA CHARYTANOWICZ [b,c], PIOTR KULCZYCKI [a,b]

[a] Faculty of Physics and Applied Computer Science
AGH University of Science and Technology, al. Mickiewicza 30, 30-059 Cracow, Poland
e-mail: {slukasik,pkowal,kulpi}@agh.edu.pl

[b] Systems Research Institute
Polish Academy of Sciences, ul. Newelska 6, 01-447 Warsaw, Poland
e-mail: {slukasik,pakowal,kulpi,mchmat}@ibspan.waw.pl

[c] Faculty of Electrical Engineering and Computer Science
Lublin University of Technology, ul. Nadbystrzycka 38D, 20-618 Lublin, Poland
e-mail: m.charytanowicz@pollub.pl

Extracting useful information from astronomical observations represents one of the most challenging tasks of data exploration. This is largely due to the volume of the data acquired using advanced observational tools. While other challenges typical for the class of big data problems (like data variety) are also present, the size of datasets represents the most significant obstacle in visualization and subsequent analysis. This paper studies an efficient data condensation algorithm aimed at providing its compact representation. It is based on fast nearest neighbor calculation using tree structures and parallel processing. In addition to that, the possibility of using approximate identification of neighbors, to even further improve the algorithm time performance, is also evaluated. The properties of the proposed approach, both in terms of performance and condensation quality, are experimentally assessed on astronomical datasets related to the GAIA mission. It is concluded that the introduced technique might serve as a scalable method of alleviating the problem of the dataset size.

**Keywords:** big data, astronomy, data reduction, nearest neighbor search, kd-trees.

## 1. Introduction

Recent decades have been characterized by an unprecedented burst of new data being generated in various fields of science and engineering. In essence, it can be useful to generate new insights, which lead to new discoveries and improve our standard of living. However, the toolbox of contemporary data science, though broad and reinforced by unconventional methods of artificial intelligence, does not contain algorithms which cope well with the challenges of the so-called big data. This term encompasses a set of problematic properties of data sets stored in present-day computer systems. Besides the obvious obstacle of data volume, variety (which relates to diverse data types and structures of the datasets), velocity (which refers to the speed of new data generation) and veracity (which corresponds to data quality/uncertainty), can also be named here (Grandinetti *et al.*, 2015).

Astronomy, among other fields of science, is nowadays strongly affected by big data problems. This is due to the fact that it currently possesses a variety of data acquisition tools. In reality, the sizes of catalogs of astronomical objects reach petabytes, and they may contain billions of instances described by hundreds of parameters (Łukasik *et al.*, 2016). Even the seemingly simple task of visualizing such datasets becomes a serious challenge. In such a case, along with significant computing power, sophisticated data preprocessing algorithms are required.

The aim of this paper is to provide an efficient method of data condensation that selects the most

*Corresponding author

representative objects (data prototypes) in the data in a way that preserves the data density. Such data instances can later be used for visualization, as well as for other data mining procedures. For this purpose we propose a modified fast density-based multiscale data condensation algorithm (Mitra *et al*., 2002). Our variant utilizes a kd-trees based nearest-neighbor technique together with parallel processing. Furthermore, we suggest to optionally use approximation of $k$ nearest neighbor calculation to achieve additional performance gains. The proposed approach is thoroughly evaluated on astronomical datasets related to the GAIA mission (GAIA, 2018). The preliminary version of this paper, containing only partial results, focusing purely on performance and assuming only exact calculation of nearest-neighbors, was presented by Łukasik *et al*. (2019).

The paper is organized as follows. First, in the next section, we provide methodological preliminaries, introducing data reduction and examples of related techniques in astronomy, as well as the problem of efficient nearest neighbors calculation. Section 3 overviews the proposed approach and is followed by a discussion on the results obtained for real astronomical data, presented in Section 4. Finally, general remarks regarding characteristic features of the introduced approach and planned further studies are discussed.

## 2. Methodological preliminaries

**2.1. Data reduction and its use in astronomy.** Data preprocessing techniques used in astronomy ought to deal with large datasets, also in real-time mode. This is a consequence of the rapid development of new instruments and new data gathering schemes. It effectively means that the volume of the data generated doubles every year (Szalay and Gray, 2001). A practical illustration of this problem is the amount of objects captured by sky surveys over the last fifty years, as demonstrated in Table 1.

Consequently, data reduction is typically introduced as close as possible to the instrumentation level, i.e., at the signal/image processing phase. Its goal is to reduce the size of transferred data. Such reduction in most cases involves removing noise, signatures of the atmosphere/instrument and other contaminating factors (Freudling *et al.*, 2013; Schirmer, 2013). Typically, such reduction is a part of the data processing pipeline and is performed in online mode (Freudling and Romaniello, 2016).

When object-based data are already available, their reduction can be performed with random sampling, probabilistic modeling, algorithms based on information theory or clustering. Representative techniques implementing all of these strategies will be covered in subsequent paragraphs.

Sampling methods represent a typical approach to the problem of data reduction (Chung *et al*., 2016). Uniform sampling techniques with or without replacement are the most widely used approach also in astronomy (e.g., Dutta *et al*., 2005; Rocke and Dai, 2003). Stratified sampling, as the one preserving the ratio of objects present in different classes, is also used (e.g., Abraham *et al*., 2012).

As an alternative to statistical sampling, the aforementioned, more sophisticated procedures employing probabilistic modeling could be considered. In the work of Wang *et al*. (2009) the dataset is clustered into hyper-balls with predetermined radii. Each of them is associated with a kernel and a weight, in such way that the mixture exposes the local data distribution. Another approach of this class, presented by Zhang *et al*. (2018), involves condensation of the dataset obtained by the well-developed kernel density estimators: the reduced set density estimator or the fast reduced set density estimator. It is followed by identifying representative data points by means of the so-called exemplar score (ES), which ensures that a reduced set consists of high-density samples.

Information theory can also be a useful tool to develop efficient methods of data reduction. In the work of Huang and Chow (2006) an application of entropy to measure the quality of newly obtained representatives is presented. This method consists of two stages. First a set of representatives of a desired size is built, and then its elements are eliminated and replaced on the basis of two criteria: representative entropy or weighted representative entropy. Both measure representative performance of a single element with regards to the original set. The latter includes an additional weighting scheme which is used to make sure that data elements close to the representative have a greater effect on the entropy measure than those far from it.

Representative data instances can be also located using distance-based criteria. In this approach, a reduced sample contains iteratively added furthest items from all the already selected ones. The algorithm implementing this paradigm called DIDES (distance-and density-based sampling) requires only one parameter (granularity) which indirectly determines the number of reduced data points. At the same time this method was also found to be insensitive to initialization and noise (Ros and Guillaume, 2017). Clustering likewise represents a natural method for data reduction. In such a case data are clustered with desired granularity and then a set of representatives from each cluster is selected. Running the standard $k$-means algorithm and selecting cluster centers as reduced set represent the most typical approach based on this concept. The more complex OSC (object selection by clustering) method, presented by Olvera-López *et al*. (2010), selects border instances from each cluster (while retaining also some central ones). The method can be built upon any clustering technique as it essentially

Table 1. Selected sky surveys as reported by Łukasik *et al*. (2016).

| Survey | Institution | Number of objects | Type | Time frame |
|---|---|---|---|---|
| Hipparcos | European Space Agency | 0.12M | Optical | 1989-1993 |
| Tycho-2 | European Space Agency | 2.5M | Optical | 1989-1993 |
| DPOSS | Caltech | 550M | Optical | 1950-1990 |
| 2MASS | Univ. of Massachusetts, Caltech | 300M | Near-IR | 1997-2001 |
| GAIA | European Space Agency | 1000M | Optical | 2013- |
| SDSS | Astrophysical Research Consortium | 470M | Optical | 2000- |
| LSST | LSST Corporation | 4000M | Optical | 2019- |

proposes only an alternative scheme of representatives selection. After clustering, border instances are located in non-homogeneous clusters (border regions) and interior ones in homogeneous clusters. This is based on the assumption that border objects provide useful information allowing discrimination between different regions.

The paragraphs above described general, problem independent, methods of data reduction. As astronomy relies heavily on advanced visualization, many procedures of data reduction were developed to deal with the problem of data abundance in visual analytics (Hassan and Fluke, 2011). They are mainly based on creating new data contexts containing only selected data points, which makes data visualization less complex. Selection of such a reduced set is performed either manually Burgess *et al*. (2015) or using distance from the observer. In addition to that, data reduction is frequently build upon other underlying data exploration techniques, using supervised learning (which sometimes, in the context of data reduction, is named the wrapper approach). This means that the result of the reduction procedure is iteratively evaluated and improved over a set of runs of an auxiliary data mining procedure (e.g., classification). As a representative of this approach, an algorithm by Czarnowski and Jedrzejowicz (2017) can be given. A population of agents is used therein to find the most effective classification model based on parallel data reduction, which guarantees maximization of the classification quality. Data prototypes are selected from clusters of instances obtained by fuzzy clustering, under a condition that each representative point is derived from a single cluster.

For a more extensive overview of data reduction schemes in astronomy, along with use-case examples, one could refer to Łukasik *et al*. (2016). Here we will discuss and use the general data condensation technique proposed by Mitra *et al*. (2002). We already positively evaluated it for elementary data reduction tasks present in the preprocessing of astronomical sky surveys (Łukasik *et al*., 2016). In particular, it was found that it is very competitive in terms of preserving the original data density for an artificially generated dataset.

The condensation algorithm discussed here relies on iteratively finding points with the closest $k$-nearest neighbor (the distance to which is denoted by $r_k$) and then adding them to the reduced dataset $E$, which is initially empty. In case of ties at this stage the decision which point is selected is based on the distance to the subsequent (that is, $k+1$) neighbor. At the same time, other points lying within a disc of radius $2 \times r_k$ are eliminated (not included in the set $E$). Algorithm 1 describes the main steps of the data reduction algorithm.

It can be seen that the algorithm requires utilizing both nearest neighbor search and the so-called radius search, finding points situated in the hyper-sphere of given radius.

**Algorithm 1.** Density-based data condensation.

1: Denote by $B = \{x_1, x_2, \ldots, x_N\}$ the initial dataset. Set condensation ratio $k$.
2: Prepare empty reduced dataset $E$.
3: For each data point $x_i \in B$ calculate the distance $d_k(x_i)$ to its $k$ nearest neighbor.
4: Pick the point $j$ with the smallest value of $d_k(x_j)$, i.e., $x_j = \arg\min_{i=1,\ldots,N} d_k(x_i)$.
5: Insert $x_j$ into reduced dataset $E$, denote by $r_k$ the distance to its neighbor $k$.
6: Remove all points from $B$ which are situated within $2 \times r_k$ from $x_j$.
7: Repeat Steps 3–6 until $B$ is not empty.

**2.2. Fast nearest neighbors search.** Finding nearest neighbors in data constitutes a very important issue, as a plethora of data mining algorithms are built on

**Algorithm 2.** Exhaustive $k$-nearest neighbor search.

1: Calculate all pairwise distances $d(x_i, x_j)$, $i, j = 1, \ldots, N$.
2: For each query point $x_i$ sort distances in ascending order.
3: For each query point $x_i$ find a set of $k$ closest neighbors.

this component. It includes outlier detection (Breunig *et al*., 2000), classification (Li *et al*., 2008) and clustering (Bubeck and von Luxburg, 2009). That is why fast $k$-nearest neighbor calculation is crucial to data analysis, especially when it is performed on large datasets.

Naive, so-called brute-force or exhaustive, $k$-nearest neighbor search involves steps listed as Algorithm 2.

The time complexity of such procedures—taking into account that the neighborhood is to be identified for all query points—is $O(N^2) + (N^2 \log N)$. It becomes prohibitive for most practical applications (Arefin *et al*., 2012).

Three major classes of algorithms have been identified to speed-up the process of locating nearest neighbors: kd-trees or other tree-based partitioning structures, hashing techniques and neighboring graph approaches. The first approach is based on building a hierarchical structure partitioning the data recursively, e.g., along the dimension of maximum variance. Examples of approaches based on this paradigm include the use of kd-trees (Eastman and Weiss, 1982) and vp-trees (Yianilos, 1993). The second approach is based on hashing points in a similarity preserving way, i.e., by putting it them in the buckets grouping similar items (as in locality sensitive hashing (Zhang *et al*., 2013)). An example of the third strategy can be found in the work of Wang *et al*. (2013), where a random $k$-NN graph approximation, updated in each step of the algorithm, is used.

For a more extensive discussion on fast nearest-neighbor strategies, one could refer to Muja and Lowe (2014). In the subsequent part of the paper we will discuss how fast nearest neighbor calculation, based on kd-trees, can be included within the parallel scheme of data reduction strategy to make its execution feasible, even for large datasets.

## 3. Proposed approach

The proposed scheme of efficient data condensation involves first building a tree structure using a standard kd-tree algorithm. It should be then distributed among nodes which are subsequently used for $k$-nearest neighbor calculation. Each node is responsible for locating a set of nearest neighbors for its assigned part of the dataset. The search process itself is again parallelized at a multi-core level. It is followed by a sequential radius search, locating points situated within $2 \times r_k$ from the one with the smallest value of $d_k(x_j)$ and removing them from the dataset. Evidently, each removal operation requires an update of the kd-tree data structure. The process is repeated until the initial dataset is completely pruned. The algorithmic summary of the whole process is presented in Fig. 1.

It can be observed that parallelization is used at the most often repeated step of locating nearest neighbors. It is not needed at the stage of radius search, as it is executed only once for one point in each data pruning step.

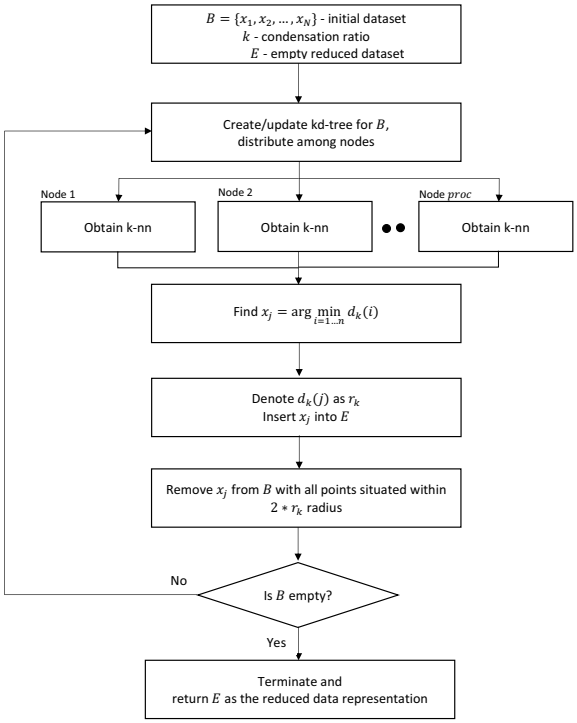


Fig. 1. Data condensation with parallel nearest neighbors calculation based on kd-trees.

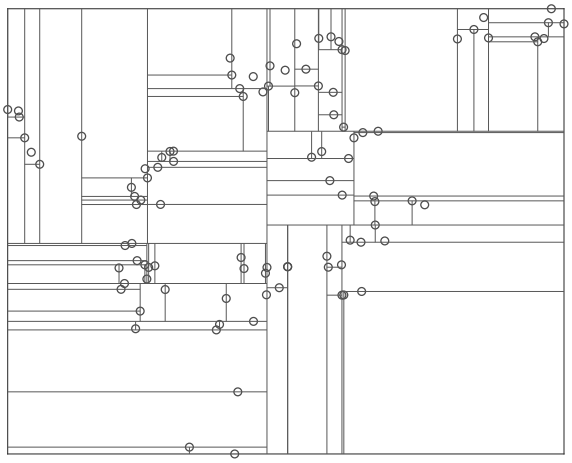

Fig. 2. kd-Tree for a normalized portion of GAIA data.

Using kd-trees as a tool for efficient nearest-neighbor search allows us to implement additional approximation. The tree is generally searched, starting from the root node, by moving down the tree recursively, i.e., it goes left or right depending on whether the point is lesser or greater than the current node in the split dimension. At each

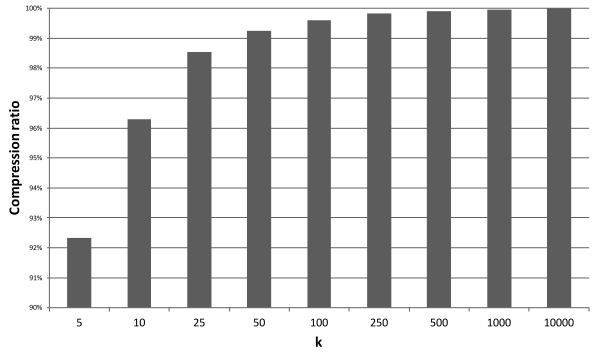


Fig. 3. Compression ratio for a varying value of $k$.

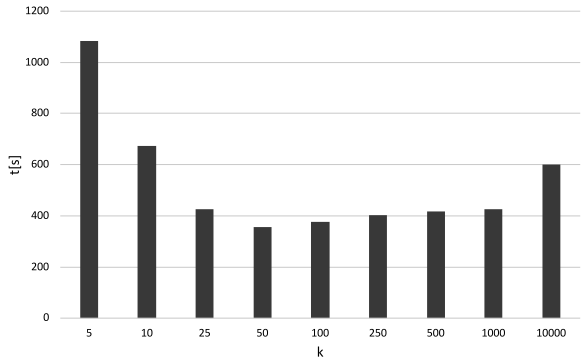


Fig. 4. Running times for a varying value of $k$.

traversed node a check is done to store nearest-neighbors. To make the algorithm run faster (at the cost of accuracy), an upper bound (denoted as $MaxNode$) on the number points to examine in the tree can be set.

In the subsequent part of the paper we will evaluate this approach and study its properties.

## 4. Experimental results

To evaluate the performance of the proposed data condensation approach, we set up an experiment involving reducing the dataset size for a portion of the GAIA Data Release 1 snapshot (GAIA, 2018). We used only spatial information, namely, transformed 3D coordinates of astronomical objects. Figure 2 demonstrates the result of kd-tree search on a normalized portion of GAIA for the first two coordinates.

As a distributed computing paradigm, MPI was used (MPI Forum, 2015). At the same time we employed OpenMP (OpenMP Architecture Review Boards, 2015) as a model for multi-core parallelization of nearest-neighbor search on one node. For the measure of time performance of a newly developed parallel algorithm, speed-up, understood as

$$\text{Speedup} = \frac{t_{\text{parallel}}}{t_{\text{sequential}}} 100\%, \tag{1}$$

i.e., running time of the parallel algorithm divided by the running time of its sequential variant, was selected.

When examining the properties of the proposed approach, we first explored the impact of neighborhood size $k$ on the resulting reduced dataset size, assuming that the initial dataset had 250 000 objects. Figure 3 contains the results of this experiment. It can be seen that the reduction provides a very compact representation of the analyzed snapshot, even for a small number of neighbors we obtain only a few percent of the initial sample.

It is important to note that the intensity of data reduction does not change with the initial dataset size—it is therefore inherently associated with geometrical properties of the dataset. This feature is demonstrated by the compression ratios shown in Table 2.

The process of data reduction is usually performed to achieve the assumed size of the resulting dataset. At the same time it was established that it is not possible to estimate it on the basis of the value of $k$. Still, as the compression ratio depends on the geometrical properties of the data, it is enough to estimate it on the small data sample, testing different values of $k$, and use the number of neighbors resulting with the desired reduction ratio for the whole dataset.

Secondly, we studied the running times for the algorithm with varying values of condensation ratio $k$. The initial dataset size was set again to 250 000 elements. One node with 4 threads running concurrently was used for this experiment. Figure 4 demonstrates the results of this test.

The analysis of the results leads us to conclusion that the profit resulting from parallelization within one node increases with the size of $k$. This is due to the frequent need of synchronization, with small values of the number $k$, which reduces the efficiency of parallel processing. Furthermore, higher values of $k$ result in more intensive reduction, requiring fewer steps of data pruning. It was observed, however, that for the large number of neighbors the impact of identifying neighboring points becomes more tangible, and it consequently deteriorates the algorithm's time performance.

We also studied obtained speed-up values for parallel calculation of nearest neighbors for 100 000 objects using 4 nodes, with one parallel thread running on each of them. The results are reported in Fig. 5. It was established that parallelization is more beneficial for a high number of neighbors. While $k$ increases, the speed-up becomes asymptotically linear. The growing overhead of communication and synchronization operations was not yet observed in this case.

Table 2. Compression ratios for different initial dataset sizes.

| | $k$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $N$ | 5 | 10 | 25 | 50 | 100 | 250 | 500 | 1000 | 10000 |
| 50000 | 90.88% | 95.57% | 98.22% | 99.08% | 99.52% | 99.78% | 99.88% | 99.93% | 99.99% |
| 100000 | 91.41% | 95.88% | 98.38% | 99.17% | 99.57% | 99.80% | 99.89% | 99.94% | 99.99% |
| 250000 | 92.34% | 96.29% | 98.54% | 99.24% | 99.59% | 99.83% | 99.90% | 99.95% | 99.99% |

The effect of using different parallel calculations models was also under investigation. We have studied speed-up values for a varying number of neighbors and 3 configurations:

- 4 nodes with 1 parallel thread running on each of them,
- 2 nodes with 2 parallel threads running on both of them,
- 1 node with 4 threads.

The results for all of the above setups are provided in Table 3. It was observed that the best performance was obtained purely on multi-core computation model (i.e., only using OpenMP-based parallelization). For small values of $k$ configuration with 2 nodes, 2 threads each were over-performed by the algorithm run only as single threads on 4 nodes. However, it was found to be faster for large values of $k$. This may be explained by the impact of the time cost of the kd-tree distribution, which for less computationally demanding tasks becomes an important factor. The configuration with many nodes is also more prone to network jitter, which might lead to anomalies in the obtained speed-up (as demonstrated by its decrease for $k = 10$ and 4 nodes in Table 3).

We also studied the possibility of using approximate nearest neighbor calculation to achieve additional boost in terms of calculation time. For that purpose we assumed (as indicated in the previous section) that the kd-tree can be evaluated only partially. We studied the relations between the maximum number of visited nodes and both the speed-up and the approximation error 'Error'. Errors here correspond to the relative number of initial dataset elements present both in the reduced dataset with and without approximation, and calculated in the following way:

$$\text{Error} = \left(1 - \frac{|E \cap E_{\text{approx}}|}{|E|}\right) 100\%, \qquad (2)$$

where $E$ is represents reduced representation of the dataset $B$, $E_{\text{approx}}$ is reduced dataset $B$ obtained with approximation.

Tables 4 and 5 provide a summary of the obtained results for $k = 1, 2, \ldots, 5$ (with dataset size 10 000) and $k = 5, 10, \ldots, 25$ (with dataset size 50 000) . It should be noted that maximum number of visited nodes was set as greater than or equal to $k$. For a smaller number of nodes, the error was unacceptable, and if $MaxNode$ was set close to $k$ the error of approximation was found to be still significant. With an increasing number of the neighbors considered, this error decreases considerably (so does the speed-up). The gain from approximation was not large due to the small scale of experimental problems; some minor anomalies (e.g., speed-up value for $k = 5$ and 50 nodes) related to the effect of background processes were also observed. It should be noted that the impact of tree reconstruction is higher than the search itself. Still, when a proper value of $MaxNode$ is selected, an improvement in computation time, with no approximation error, can be observed.

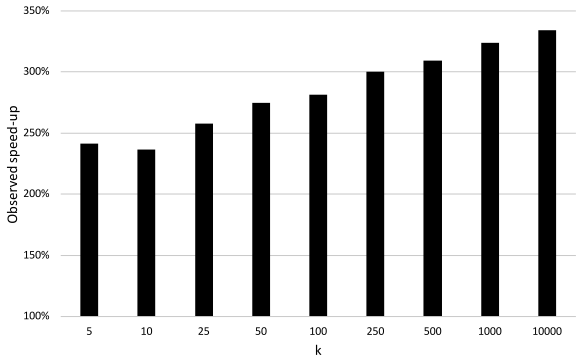


Fig. 5. Observed speed-up for 4 nodes with one thread running vs. single node configuration.

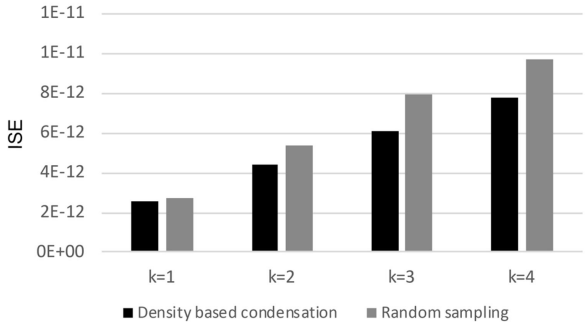


Fig. 6. ISE of probabilistic density calculated for datasets reduced with density-based condensation and random sampling.

Table 3. Observed speed-up for different parallel configurations.

| | $k$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Configuration | 5 | 10 | 25 | 50 | 100 | 250 | 500 | 1000 | 10000 |
| 1 node, 1 thread | 116 sec. | 71 sec. | 49 sec. | 44 sec. | 45 sec. | 57 sec. | 65 sec. | 81 sec. | 97 sec. |
| 2 nodes, 2 threads | 193% | 263% | 272% | 293% | 300% | 317% | 325% | 338% | 346% |
| 1 node, 4 threads | 283% | 284% | 288% | 314% | 321% | 335% | 342% | 352% | 373% |
| 4 nodes, 1 thread | 242% | 237% | 258% | 275% | 281% | 300% | 310% | 324% | 334% |

Table 4. Speed-up and error for different levels of approximation ($k < 5$).

| | | Max. visited nodes | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Configuration and performance | | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| $k = 1$ | Speed-up | 111% | 109% | 107% | 105% | 104% | 102% | 102% | 102% | 102% |
| | Error | 48.7% | 30.4% | 17.0% | 8.1% | 3.5% | 1.2% | 0.4% | 0.2% | 0.0% |
| $k = 2$ | Speed-up | – | 120% | 114% | 110% | 109% | 107% | 106% | 105% | 104% |
| | Error | – | 31.5% | 14.3% | 7.7% | 3.6% | 2.0% | 0.7% | 0.2% | 0.0% |
| $k = 3$ | Speed-up | – | – | 129% | 119% | 114% | 111% | 109% | 107% | 107% |
| | Error | – | – | 43.8% | 18.3% | 8.7% | 4.0% | 2.1% | 0.3% | 0.0% |
| $k = 4$ | Speed-up | – | – | – | 139% | 122% | 116% | 114% | 111% | 109% |
| | Error | – | – | – | 51.4% | 24% | 9.9% | 5.5% | 2.0% | 1.0% |

Finally, the quality of condensation, in terms of preserving data density, was under investigation. For that purpose we used the ISE (integrated square error) of estimator $\hat{f}$ of probabilistic density function $f$, which in general is expressed as follows:

$$\mathrm{ISE}(\hat{f}(x)) = \int (\hat{f}(x) - f(x))^2 \, \mathrm{d}x. \tag{3}$$

In this case, we assume that $f$ is an empirical density constructed for the initial sample. At the same time, $\hat{f}(x)$ corresponds to the estimator constructed for the reduced dataset. The estimator values will be calculated for the points of the initial sample. Thus the problem of calculating the integrated square error is defined by

$$\mathrm{ISE}(\hat{f}(x)) = \sum_{i=1}^{m} (\hat{f}(x_i) - f(x_i))^2, \tag{4}$$

with $x_i$ being a sample element obtained from the original dataset. The values of the ISE were examined for the GAIA subsample consisting of $m = 50000$ elements. As a point of reference, random sampling (uniformly distributed) of the initial dataset (to obtain its reduced representation) was used.

Density estimates were calculated using the kernel density estimator

$$\hat{g}(x) = \frac{1}{mh^n} \sum_{i=1}^{m} w_i K\Big(\frac{x - x_i}{h}\Big). \tag{5}$$

For the technique covered in this paper, additional weights $w_i$, equal to the number of points removed for each "prototype", were used. The experiments presented here involved using a Gaussian kernel. The value of smoothing parameter $h$ was calculated with the commonly used Silverman "rule of thumb" (Kulczycki, 2008). As random sampling is of non-deterministic nature, we used 30 replicates and report the mean ISE. The results for $k = \{1, 2, 3, 4\}$ are given in Fig. 6.

It can be seen that the proposed algorithm preserves the landscape of the probabilistic density function much better. The advantage over random sampling in this area is more significant for the cases with intensive data reduction (e.g., for $k = 2 - 4$ ISE values are approximately 20% smaller).

## 5. Conclusion

The paper presented an application of fast $k$-nearest neighbor search, based on kd-trees and parallel processing, in data condensation strategies. Our experiments demonstrate that using improved nearest-neighbor strategies may allow us to tackle large astronomical datasets. This is due to the use of efficient data representation and parallelization of data reduction scheme. We also evaluated the performance of data condensation with varying error of nearest-neighbor approximation. It was discovered that additional improvement in data reduction efficiency can be achieved in this way. Finally, we also evaluated the preservation of data density for both the proposed algorithm and random sampling. The superiority of the former approach was clearly observed.

Practical significance of proposed technique stems mainly from the fact that it can be used for dynamic large-scale visualization of astronomical objects. The

Table 5. Speed-up and error for different levels of approximation ($k \geq 5$).

| Configuration and performance | | Max. visited nodes 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 |
|---|---|---|---|---|---|---|---|---|---|---|
| $k = 5$ | Speed-up | 112% | 106% | 104% | 101% | 101% | 101% | 101% | 101% | 101% |
| | Error | 1.9% | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $k = 10$ | Speed-up | – | 114% | 114% | 109% | 104% | 104% | 104% | 104% | 100% |
| | Error | – | 14.4% | 0.8% | 0% | 0% | 0% | 0% | 0% | 0% |
| $k = 15$ | Speed-up | – | – | 118% | 118% | 108% | 108% | 108% | 108% | 100% |
| | Error | – | – | 42.3% | 13.1% | 3.9% | 0% | 0% | 0% | 0% |
| $k = 20$ | Speed-up | – | – | – | 126% | 118% | 110% | 108% | 105% | 103% |
| | Error | – | – | – | 67.4% | 26.1% | 6.5% | 0% | 0% | 0% |
| $k = 25$ | Speed-up | – | – | – | – | 130% | 120% | 116% | 112% | 109% |
| | Error | – | – | – | – | 69.8% | 33.3% | 14.3% | 0.0% | 0.0% |

intensity of data reduction can be conveniently controlled with a value of $k$, determining data granularity. Furthermore, the obtained reduced data representation is composed of data clusters corresponding to spatially concentrated groups of objects. Together with the application of other strategies, like deep learning (Castro-Ginard *et al*., 2018), it can lead to the discovery of open clusters. It is in contrast to the traditional data a reduction performed with random sampling. In this case a reduced dataset is composed of nondeterministically selected data points with no additional interpretation.

Further work in this area will concern studying the properties of other methods of nearest-neighbor approximation, e.g., techniques based on random kd-trees or hierarchical clustering. Additional experiments using the GPU computing paradigm are also planned. They will be aimed at providing production environment for large-scale astronomical data reduction. Subsequent steps include modifying the algorithm for GAIA data interface and the ability to process and merge large data hyper cubes. Additional work also needs to be done in the area of convenient data representation after their within reduction. As the reduction procedure requires significant amount of time, its result should be precomputed and stored for further use.

## Acknowledgment

This work was partially financed under the AGH UST Faculty of Physics and Applied Computer Science statutory tasks within a subsidy of the Ministry of Science and Higher Education. The study was also supported in part by PL-Grid Infrastructure.

The authors would like to express their gratitude to Profs. Rita A. Ribeiro, António Falcão and André Moitinho, who provided insight and expertise that greatly assisted the research.

## References

Abraham, S., Philip, N.S., Kembhavi, A., Wadadekar, Y.G. and Sinha, R. (2012). A photometric catalogue of quasars and other point sources in the Sloan Digital Sky Survey, *Monthly Notices of the Royal Astronomical Society* **419**(1): 80–94, DOI: 10.1111/j.1365-2966.2011.19674.x.

Arefin, A.S., Riveros, C., Berretta, R. and Moscato, P. (2012). GPU-FS-kNN: A software tool for fast and scalable kNN computation using GPUs, *PLoS One* **7**: e44000, DOI: 10.1371/journal.pone.0044000.

Breunig, M.M., Kriegel, H.-P., Ng, R.T. and Sander, J. (2000). LOF: Identifying density-based local outliers, *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, SIGMOD'00, Dallas, TX, USA*, pp. 93–104, DOI: 10.1145/342009.335388.

Bubeck, S. and von Luxburg, U. (2009). Nearest neighbor clustering: A baseline method for consistent clustering with arbitrary objective functions, *Journal of Machine Learning Research* **10**: 657–698.

Burgess, R., Falcão, A.J., Fernandes, T., Ribeiro, R.A., Gomes, M., Krone-Martins, A. and de Almeida, A.M. (2015). Selection of large-scale 3d point cloud data using gesture recognition, *in* L. Camarinha-Matos *et al.* (Eds), *Technological Innovation for Cloud-Based Engineering Systems*, Springer International Publishing, Cham, pp. 188–195, DOI: 10.1007/978-3-319-16766-4_20.

Castro-Ginard, A., Jordi, C., Luri, X., Julbe, F., Morvan, M., Balaguer-Núñez, L. and Cantat-Gaudin, T. (2018). A new method for unveiling open clusters in Gaia: New nearby open clusters confirmed by DR2, *Astronomy and Astrophysics* **618**: A59.

Chung, Y.-Y., Tirthapura, S. and Woodruff, D.P. (2016). A simple message-optimal algorithm for random sampling from a distributed stream, *IEEE Transactions on Knowledge and Data Engineering* **28**(6): 1356–1368, DOI: 10.1109/TKDE.2016.2518679.

Czarnowski, I. and Jedrzejowicz, P. (2017). Learning from examples with data reduction and stacked generalization, *Journal of Intelligent & Fuzzy Systems* **32**(2): 1401–1411.

Dutta, H., Giannella, C., Borne, K. and Kargupta, H. (2005). Distributed top-K outlier detection from astronomy catalogs using the DEMAC system, *2007 SIAM International Conference on Data Minning, Minneapolis, MN, USA*, pp. 473–478, DOI: abs/10.1137/1.9781611972771.47.

Eastman, C. and Weiss, S. (1982). Tree structures for high dimensionality nearest neighbor searching, *Information Systems* **7**(2): 115–122.

Freudling, W. and Romaniello, M. (2016). Delivering data reduction pipelines to science users, *SPIE Proceedings* **9910**: 99101U, DOI: 10.1117/12.2232734.

Freudling, W. Romaniello, M., Bramich, D.M., Ballester, P., Forchi, V., Garcia-Dablo, C.E., Moehler, S. and Neeser, M.J. (2013). Automated data reduction workflows for astronomy. The ESO Reflex environment, *Astronomy and Astrophysics* **559**: A96, DOI: 10.1051/0004-6361/201322494.

GAIA (2018), GAIA Mission, `https://www.cosmos.esa.int/gaia`.

Grandinetti, L., Joubert, G., Kunze, M. and Pascucci, V. (2015). *Big Data and High Performance Computing*, Advances in Parallel Computing, IOS Press, Amsterdam.

Hassan, A. and Fluke, C.J. (2011). Scientific visualization in astronomy: Towards the petascale astronomy era, *Publications of the Astronomical Society of Australia* **28**(2): 150–170.

Huang, D. and Chow, T.W.S. (2006). Enhancing density-based data reduction using entropy, *Neural Computation* **18**(2): 470–495, DOI: 10.1162/089976606775093927.

Kulczycki, P. (2008). Kernel estimators in industrial applications, *in* B. Prasad (Ed.), *Soft Computing Applications in Industry*, Springer, Berlin/Heidelberg, pp. 69–91, DOI: org/10.1007/978-3-540-77465-5_4.

Li, L., Zhang, Y. and Zhao, Y. (2008). k-Nearest neighbors for automated classification of celestial objects, *Science in China G: Physics, Mechanics and Astronomy* **51**(7): 916–922, DOI: 10.1007/s11433-008-0088-4.

Łukasik, S., Lalik, K., Sarna, P., Kowalski, P.A., Charytanowicz, M. and Kulczycki, P. (2019). Efficient astronomical data condensation using fast nearest neighbor search, *in* P. Kulczycki *et al.* (Eds), *Information Technology, Systems Research and Computational Physics*, Advances in Intelligent Systems and Computing, Vol. 945, Springer, Berlin/Heidelberg, pp. 107–115.

Łukasik, S., Moitinho, A., Kowalski, P.A., Falcão, A., Ribeiro, R.A. and Kulczycki, P. (2016). Survey of object-based data reduction techniques in observational astronomy, *Open Physics* **14**(1): 64, DOI: 10.1515/phys-2016-0064.

MPI Forum (2015). *MPI: A Message-passing Interface Standard: Version 3.1*, High-Performance Computing Center, Stuttgart, `https://www.mpi-forum.org/docs/mpi-3.1/mpi31-report.pdf`.

Mitra, P., Murthy, C. and Pal, S.K. (2002). Density-based multiscale data condensation, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(6): 734–747, DOI: 10.1109/TPAMI.2002.1008381.

Muja, M. and Lowe, D.G. (2014). Scalable nearest neighbor algorithms for high dimensional data, *IEEE Transactions on Pattern Analysis & Machine Intelligence* **36**(11): 2227–2240, DOI: 10.1109/TPAMI.2014.2321376.

Olvera-López, J., Ariel Carrasco-Ochoa, J., Martnez-Trinidad, J.F. and Kittler, J. (2010). A review of instance selection methods, *Artificial Intelligence Review* **34**(2): 133–143.

OpenMP Architecture Review Boards (2015). *OpenMP 4.5 Complete Specifications*, `https://www.openmp.org/wp-content/uploads/openmp-4.5.pdf`.

Rocke, D.M. and Dai, J. (2003). Sampling and subsampling for cluster analysis in data mining: With applications to sky survey data, *Data Mining and Knowledge Discovery* **7**(2): 215–232, DOI: 10.1023/A:1022497517599.

Ros, F. and Guillaume, S. (2017). DIDES: A fast and effective sampling for clustering algorithm, *Knowledge and Information Systems* **50**(2): 543–568, DOI: 10.1007/s10115-016-0946-8.

Schirmer, M. (2013). THELI: Convenient reduction of optical, near-infrared, and mid-infrared imaging data, *The Astrophysical Journal Supplement Series* **209**(2): 21, DOI: 10.1088/0067-0049/209/2/21.

Szalay, A. and Gray, J. (2001). The world-wide telescope, *Science* **293**(5537): 2037–2040.

Wang, D., Shi, L. and Cao, J. (2013). Fast algorithm for approximate k-nearest neighbor graph construction, *2013 IEEE 13th International Conference on Data Mining Workshops, Dallas, TX, USA*, pp. 349–356, DOI: 10.1109/ICDMW.2013.50.

Wang, X., Tino, P., Fardal, M.A., Raychaudhury, S. and Babul, A. (2009). Fast Parzen window density estimator, *2009 International Joint Conference on Neural Networks, Atlanta, GA, USA*, pp. 3267–3274, DOI: 10.1109/IJCNN.2009.5178637.

Yianilos, P.N. (1993). Data structures and algorithms for nearest neighbor search in general metric spaces, *Proceedings of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA'93, Austin, TX, USA*, Vol. 93, pp. 311–321.

Zhang, Y., Chung, F. and Wang, S. (2018). Fast reduced set-based exemplar finding and cluster assignment, *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **49**(5): 1–15.

Zhang, Y.-M., Huang, K., Geng, G. and Liu, C.-L. (2013). Fast kNN graph construction with locality sensitive hashing, *in* H. Blockeel *et al.* (Eds), *Machine Learning and Knowledge Discovery in Databases*, Springer, Berlin/Heidelberg, pp. 660–674, DOI: 10.1007/978-3-642-40991-2_42.

**Szymon Łukasik** is an assistant professor both at the Systems Research Institute of the Polish Academy of Sciences and the Faculty of Physics and Applied Computer Science of the AGH University of Science and Technology. His scientific interests revolve around various aspects of data analysis, nature-inspired algorithms and innovative applications of collective intelligence in distributed knowledge discovery.

**Konrad Lalik** is an MSc graduate from the Faculty of Physics and Applied Computer Science at the AGH University of Science and Technology. Currently he works as a .NET developer at Forte_ Digital.

**Piotr Sarna** is a BSc graduate from the Faculty of Physics and Applied Computer Science at the AGH University of Science and Technology. Currently he works as a software engineer at Comarch.

**Piotr A. Kowalski** is an assistant professor with the Systems Research Institute, Polish Academy of Sciences, and with the Faculty of Physics and Applied Computer Science, AGH University of Science and Technology, Kraków. His current research interests include information technology and are focused on intelligent methods, such as neural networks, fuzzy systems, and nature inspired algorithms, with applications to complex systems and knowledge discovery.

**Małgorzata Charytanowicz** is an assistant professor both at the Systems Research Institute of the Polish Academy of Sciences and the Faculty of Electrical Engineering and Computer Science of the Lublin University of Technology. Her research interests are in programming methods, data analysis and its applications for medicine.

**Piotr Kulczycki** is a professor at the Systems Research Institute of the Polish Academy of Sciences (and the head of its Center for Information Technology for Data Analysis Methods), as well as at the AGH University of Science and Technology. The area of his scientific activity covers application-oriented aspects of information technology as well as data mining and analysis, mostly connected with the use of modern statistical methods and fuzzy logic in diverse issues of contemporary systems research and control engineering.

Received: 11 November 2018
Revised: 14 February 2019
Accepted: 18 March 2019