Computers and Geosciences 119 (2018) 1–8

Contents lists available at ScienceDirect

Computers and Geosciences

journal homepage: www.elsevier.com/locate/cageo

A fast approach for unsupervised karst feature identification using GPU

Luis C.S. Afonso[a,*,1], Mateus Basso[b,2], Michelle C. Kuroda[b,2], Alexandre C. Vidal[b,2], João P. Papa[c,1]

[a] UFSCar - Federal University of São Carlos, Department of Computing, São Carlos, Brazil
[b] UNICAMP - University of Campinas, Institute of Geosciences, Campinas, Brazil
[c] UNESP - São Paulo State University, School of Sciences, Bauru, Brazil

ARTICLE INFO

Keywords:
Self-organizing map
Paleokarst
Graphics processing unit
Campos basin

ABSTRACT

Among the geological features, karst is the one that has received special attention in oil and gas exploration for being a strong indicator of the potential existence of hydrocarbon reservoirs. The integration of automatic pattern recognition methods and Graphics Processing Units (GPU) provides a powerful tool to help geological interpretation of seismic data. In order to provide insightful information for interpreters, this work investigates the usage of GPUs in addition to image segmentation by means of unsupervised classification for the identification of karst features in 3D seismic data. For this purpose, an implementation of the robust Self-Organizing Map for GPUs (SOM/GPU) is provided, and a comparison against a Central Processing Unit (CPU)-based SOM (SOM/CPU) is performed to assess the speeding-up provided by GPU. Experiments have shown promising results for geological interpretation using seismic data.

1. Introduction

At least 40% of the recoverable hydrocarbons are trapped in stratigraphic unconformities such as the ones originated from karstification (Sayago et al., 2012). Among the fields originated from karst, one can refer to the Lower Ordovician Puckett and the Permian Yates in west Texas (Loucks, 1999), the Upper Devonian Grosmont Formation in Alberta, Canada (Luo et al., 1994; Buschkuehle et al., 2007) and the Lower Ordovician Lunnan field in the Tarim Basin of China (Zhao et al., 2014, 2015).

The karstification process creates, in most cases, geomorphological features that are barely continuous and randomly spatially distributed in the seismic data. Also, they have seismic responses that occur in subtle ways that can be easily misunderstood with other geological features, seismic noise or simply not be noticed by the interpreter (Maoshan et al., 2011). In addition, the exploration to identify karst features is a time-consuming task since it requires the analysis of huge volumetric seismic data. The volume of data can have its size significantly increased if more information is added (i.e., computing and analyzing multiple attributes) in order to help out the interpretation.

The requirements to solve this problem can be met by employing graphics processing units (GPUs) that tackle both the problem of amount of data and computing time by providing a low-cost device with parallel architecture that enables the processing of high amounts of data simultaneously in a short time. Despite not having been fully investigated (Jeong et al., 2006), this powerful resource has been largely employed in a wide range of applications including simulations in geosciences research (Rubio et al., 2014; Lacasta et al., 2015), and acceleration of calculations, especially for reservoir characterization (Liu et al., 2009; Komatitsch et al., 2010). In some applications, the GPU parallel implementation reached high speed-up values (Tahmasebi et al., 2012; Cheng, 2013; Li et al., 2014).

This work explores the usage of GPUs for the application of unsupervised karst identification in 3D seismic data using multi-attribute data. The purpose of this study is to explore the programming challenges and the potential benefits of embedded computing using commodity hardware components. Due to the lack of well data from our study area and complexity of the seismic data, we decided to apply an unsupervised classification approach using the Self-Organizing Map (SOM) algorithm. SOM has been the practice in a large number of different applications (Chang et al., 2002; Ersoy et al., 2007; Kuroda et al., 2012; Mojarab et al., 2014), this study applies the technique to better interpret karst features in 3D seismic data.

The main contributions of this work are:

* Corresponding author.
E-mail addresses: sugi.luis@gmail.com, sugi.luis@ufscar.br (L.C.S. Afonso), mbstraik@gmail.com (M. Basso), mckuroda@ige.unicamp.br (M.C. Kuroda), acvidal@ige.unicamp.br (A.C. Vidal), papa@fc.unesp.br (J.P. Papa).
[1] Responsible for studying the algorithm to be used in the application as well as performing the classification experiments.
[2] Responsible for studying the most reliable attributes for this application as well as aiding in the geological evaluation of the experimental results.

https://doi.org/10.1016/j.cageo.2018.06.004
Received 8 June 2017; Received in revised form 29 May 2018; Accepted 7 June 2018
Available online 15 June 2018
0098-3004/

- a fast approach for karst feature identification in huge seismic volumetric data;
- the proposal of an unsupervised GPU-based karst feature identification approach;
- the karst feature identification using multi-attribute data;
- to provide insightful information by identifying potential karst features;
- to reduce the amount of data to be analyzed and time spent in the interpretation.

The remainder of this paper is organized as follows: Section 2 provides an overview of karst features. The theoretical background of the SOM is presented in Section 3, and a brief history of GPUs and the architecture of a Compute Unified Device Architecture (CUDA)-enabled GPU are presented in Section 4. The parallel implementation employed in the experiments is introduced in Section 5. The methodology adopted and the case of study used for the evaluation are presented in Sections 6 and 7, respectively. Conclusions regarding the experimental results are stated in Section 8.

2. Overview of karst features

Karst is described as a landscape that contains caves and extensive underground water systems that develops on soluble rocks such as limestone, marble and gypsum (Ford and Willians, 1989). Karst is formed from the subaerial exposure of carbonate rocks, recognizable by features produced by dissolution, precipitation, erosion, sedimentation and collapse (Esteban and Wilson, 1992). It can also be the result of corrosion caused by hydrothermal processes and differential CO^2 regimes or fluids containing H^2S (Immenhauser and Rameil, 2011).

Near-surface karstification creates a terrain with distinct geomorphological and geological elements (Fig. 1). Well-developed drainage systems predominate in karst terrains, having a high degree of connectivity with subsurface hydrology. Common features found in karst terrains are deep canyons, large valleys associated with sinkholes, and a large number of channels, plus less developed features associated with intermittent flows or with less erosive power, such as ravines and gullies.

However, the most common surface features are sinkholes or dolines, being defined by Waltham and Fookes (2003) as karst elements formed by a closed circular depression eroded around an internal drainage point into the underlying limestone. These features mark the main relationship between surface geomorphology and the subsurface, and are usually generated by dissolution and collapse processes.

The flow rates and aggressiveness of the surface water will dictate the construction of the underground karst, creating caves, channels, conduits, passages, and chambers. The burial compaction and diagenesis of this system will result in a paleocave system that is an important class of carbonate reservoirs (Loucks, 1999).

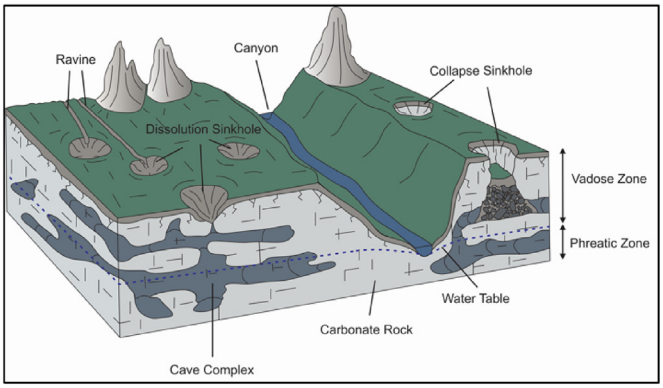

Fig. 1. Block diagram of an epigenic karst terrain, including the main features observed in exokarstic, epikarstic and endokarstic domains.

3. The Self-Organizing Map

The Self-Organizing Map (SOM) is inspired by the human brain in which each region is responsible for a specific task (Kohonen, 1990). The main feature of SOM is the competition among nearly neurons to be activated given an input sample. The competition process results in a spatially organized "internal representation" of various features of input signals and their abstractions, in which similar ones are located close to each other (Haykin, 1998). The final representation is a two-dimensional topological map composed of $N \times M$ cells (neurons) which are tuned to selectively respond to input patterns. By doing the mapping of the input samples, SOM approximates the neurons' weight vector to the input sample and creates a topological map such that similar samples will be located closer to each other in the map. Such process allows to visually identify potential clusters and their spatial organization. The mapping is conducted in the same way as in a supervised training.

The SOM learning process or pattern mapping process is divided into three steps: competition, cooperation, and adaptation. Let $\mathbf{x} \in \Re^n$, and a **j**-th neuron with synaptic weight vector $\mathbf{w}_j \in \Re^n$. In the competition process, the winner neuron is defined as the one that provides the highest level of similarity to the input sample $\mathbf{x}$. This step can be summarized in Equation (1), in which the similarity can be obtained by minimizing the Euclidean distance as follows.

$$winner(x) = \arg_j \min \|\mathbf{x} - \mathbf{w}_j\|, j = 1,2, \ldots, N \times M. \quad (1)$$

The result $winner(\mathbf{x})$ is the index of the neuron in the map whose distance to the input sample $\mathbf{x}$ is the smallest one.

The cooperation process defines a topological neighborhood centered at the winning neuron found through Equation (1). The topological neighborhood reproduces the evidence of lateral interaction among a set of neurons. It was also observed the strength of lateral interaction decays smoothly with lateral distance having its maximum strength at the winning neuron. The responses of this activity can be modeled by a Gaussian function:

$$h_{i,j}(\mathbf{x}) = \alpha(t) exp\left(-\frac{d_{i,j}^2}{2\sigma^2}\right), t = 0,1,2, \ldots, T, \quad (2)$$

in which $h_{i,j}$ stands for the topological neighborhood centered on winning neuron i with lateral distance defined through σ, $d_{i,j}^2$ represents the distance between i and activated neuron j, and T is the maximum number of iterations (Kohonen, 2013). Also, the lateral interaction σ decreases with time, which is given by:

$$\sigma^t = \sigma^0 exp\left(-\frac{t}{\tau_1}\right), \quad (3)$$

in which σ^0 is the value of σ at time 0, τ_1 is a time constant (Haykin, 1998).

Finally, the adaptation process performs the update of the neurons' weight vector based on Hebb's postulate (Hebb, 1949). The final weight vector's update function can be defined as follows:

$$\mathbf{w}_j^{t+1} = \mathbf{w}_j^t + \eta^n h_{i,j}^t (\mathbf{x} - \mathbf{w}_j^t), \quad (4)$$

in which $\mathbf{w}_j^{t+1}$ is the updated weight vector of the **j**-th neuron at time $t + 1$, and η is a learning rate parameter that decreases over time according to:

$$\eta^t = \eta^0 exp\left(-\frac{t}{\tau_2}\right), \quad (5)$$

in which η^0 is the initial value of the learning rate and τ_2 is another constant of time. Kohonen (1990) still divides the adaptive process into two phases: ordering and convergence. In the ordering phase, the topological ordering of the neurons' weight vectors takes place. The

convergence phase is related to tuning the neural network in order to provide an accurate representation of the input samples.

4. Graphics processing unit (GPU) and CUDA

The Compute Unified Device Architecture (CUDA) is a platform created by NVIDIA that enables general-purpose applications to use the power of a graphic processing unit that provides a huge increase in computational performance. Applications using CUDA go from bioinformatics (McArt et al., 2013), computational chemistry (Anthopoulos et al., 2013) to medical imaging (Shi et al., 2012), weather and climate (Michalakes and Vachharajani, 2008; Brown et al., 2015), including seismic analysis (Liu et al., 2009; Komatitsch et al., 2010).

GPUs were first designed as graphics accelerators but, in the late 1990s, their usage for general-purpose applications had significantly increased especially by research that took advantage of their great floating point performance. However, developing such applications was really difficult even for those who had knowledge of graphics programming languages such as OpenGL (OpenGL, 2016). Later on, Ian Buck and his research team from Stanford University developed the Brook compiler and runtime system (Che et al., 2008), which upgraded GPU to a general-purpose processor in a high-level language, making the programming task easier. Ian Buck and NVIDIA further developed a solution that would enable to run C programs on GPU. In 2006, NVIDIA released CUDA, in which hardware and software solutions were coupled (NVIDIA, 2010).

Fig. 2 shows the memory architecture of a CUDA-enabled GPU. The terms “host”and “device” refer to CPU and GPU, respectively. The data to be processed can be stored in the “global memory”, which can be accessed by any GPU thread or in the “shared memory” where data access is restricted to a few GPU threads. Threads are the units that execute kernels, i.e., functions written for CUDA. Threads are organized in blocks and are identified by a unique index given by its position in the block and the identification of its block. Blocks can assume up to 3D-dimension format and have a group of threads given by their dimension. A block of size $3 \times 3 \times 3$ will have 9 threads, for example.

5. GPU parallel implementation

The GPU parallel implementation used in the experiments relies on the Somoclu (Peter et al., 2015), which is implemented over Thrust (NVIDIA, 2016b), that is a C++ template library for CUDA

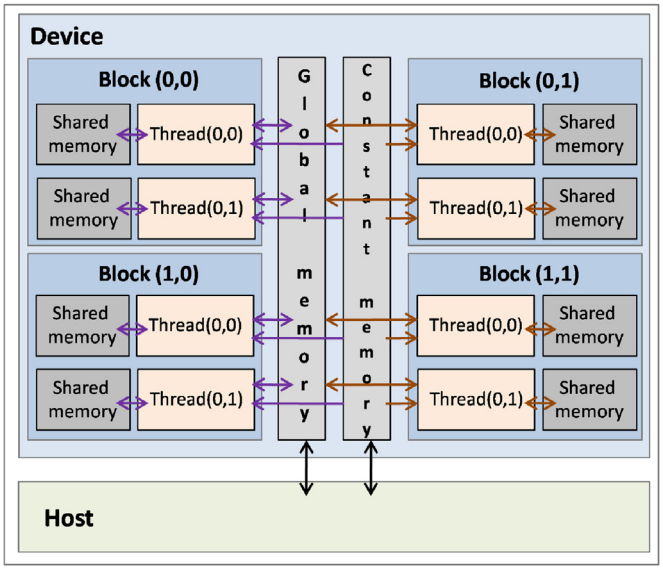

Fig. 2. Memory architecture in a CUDA-enabled GPU (Adapted from (Kirk and Hwu, 2010)).

Table 1
Structure dimensions used in the application.

	Dimension
Grid size	$\frac{SomX*SomY+BLOCK.DIM-1}{BLOCK.DIM} \times \frac{nSamples+BLOCK.DIM-1}{BLOCK.DIM} \times 1$
Block size	$32 \times 1 \times 1$
# threads	$32 \times 1 \times 1$

applications. Some operations using vectors and matrices are performed using the cuBLAS (NVIDIA, 2016a), which is an implementation of BLAS (Basic Linear Algebra Subprograms) on top of the CUDA. The program can still make use of the Message Passing Interface (MPI) (Open-MPI, 2016), that is a communication protocol used for parallel applications. The memory organization used by the Somoclu follows the values presented in Table 1, where $SomX$ is the dimension in x-axis, $SomY$ is the dimension in y-axis, and $BLOCK.DIM$ is the block size.

The parallelization in the SOM algorithm is used in the tasks that compute the best match and neuron weight vector's update, defined by Equations 1 and 3, respectively. In the first case, it is clear that finding the distance between a sample $\mathbf{x}_i$ and a neuron $\mathbf{w}_j$ is independent of finding the distance between a sample $\mathbf{x}_k$ to the same neuron $\mathbf{w}_j$, for instance. Therefore, this task is parallelized in such a way that multiple calculations can be done simultaneously by a predefined number of threads.

The neuron weight's vector update in Somoclu is performed at the end of an epoch. During an epoch, each allocated thread computes the result for Equation (3), stores the result in its local variable, and waits for the other threads to finish it. Once all threads have finished their job, their local results are all summed up and the neuron map is updated. Since the neuron map is a critical section (i.e., more than one thread accessing the same variable), a synchronization is performed during the update to keep the final result consistent.

6. Methodology

The experiments are conducted following the workflow depicted in Fig. 3. The first step computes a set of five attributes from the amplitude seismic data which will describe each sample $\mathbf{x}$. The outcome is a 5-dimensional representation, in which each dimension stands for a different attribute. The set of attributes was selected based on the features they highlight and is comprised of:

- Most positive (Fig. 4a) and most negative curvature (Fig. 4b): the curvature measures the deformation of a surface at a point. The larger the deformation, the larger the curvature is. The curvature value is positive if the feature is anticlinal, and negative if it is synclinal (Chopra S., 2007). The curvature is computed by fitting a quadratic surface on the surface patches of a given size. The most positive and most negative curvature values can be derived from a search over all possible normal curvatures Roberts (2001).
- Second derivative of the amplitude (Fig. 4c): the amplitude second derivative provides a measure of the sharpness of the amplitude peak. This attribute is an effective discriminator for bright spots, sequence boundaries, major changes in a depositional environment, and lithologic variations, among others (OpendTect, 2002). This attribute is computed from the second derivative of the envelope $E(t) = \sqrt{S(t) + H(t)}$ where $S(t)$ is the seismic trace, $H(t)$ stands fo the Hilbert's transform and t is time.
- Envelope-weighted frequency (Fig. 4d): it is the instantaneous frequency weighted by the envelope over a given time window. This attribute can be used for hydrocarbon indicator by low-frequency anomaly, fracture zone indicator, and bed thickness, among others (OpendTect, 2002). The instantaneous frequency is defined as the time derivative of the instantaneous phase.
- Isopach (Fig. 4e): this attribute gives the variation of lateral

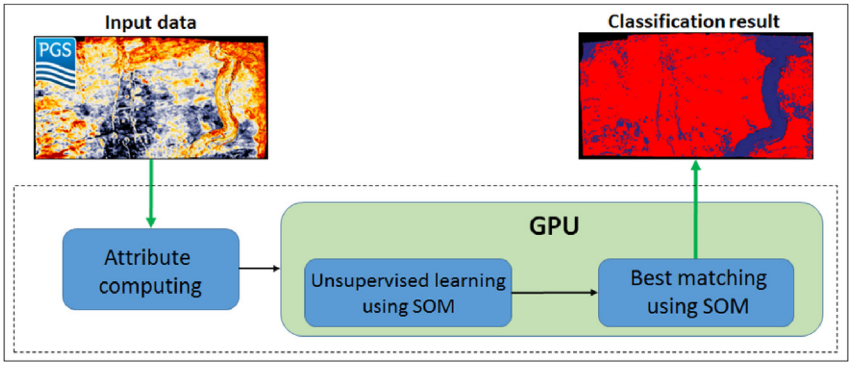

Fig. 3. Main workflow.

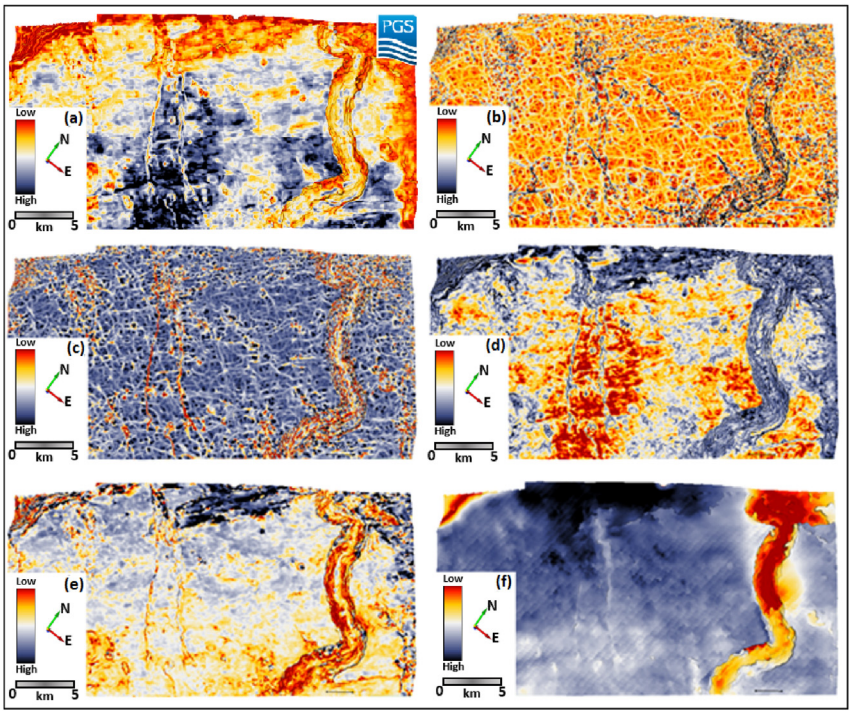

Fig. 4. The (a) data in amplitude and the set of five attributes shown at the Macaé top: (b) most positive and (c) most negative curvatures, (d) amplitude second derivative, (e) envelope-weighted frequency and (f) isopach.

Table 2
Parameters used to setup the SOM algorithm.

Parameter	Value
Neuron map size	20 × 20
Neuron map type	planar
Neighborhood function	Gaussian
Epochs	100
Initial radius	10
Final radius	1
Learning rate cooling strategy	linear
Radius cooling strategy	linear

Table 3
Additional parameters for the parallel implementation.

Parameter	Value
# nodes	1
# processes per node	16

thickness of a bed, formation or stratigraphic interval. It highlights characteristics of a basin, such as the location of buried ridges, and position of shorelines, among others (Maltman, 2000). The isopach map used in the experiments was computed using the top and basin of the Macaé formation, which are the horizons of highest acoustic impedance variation.

Due to the number of samples and calculations, the second step takes advantage of the powerful processing capability of the GPU to speed-up the unsupervised learning of the multi-attribute seismic data generated in the previous step. The unsupervised learning will map the seismic data into a 2D topological map by means of the SOM algorithm using both CPU and GPU implementations for performance comparison purposes. Both implementations have their neuron maps randomly initialized and their common parameters are listed in Table 2. Except for the neuron map size that was chosen empirically, the remaining parameters are the default values in the Somoclu library.

The GPU implementation still has some additional parameters as listed in Table 3, which were set according to the computing environment settings. The computing environment used in the experiments has the following configuration:

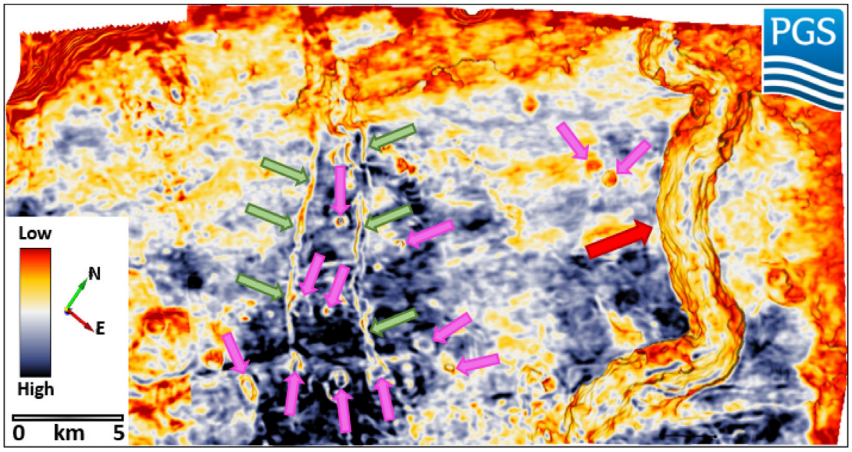

Fig. 5. Top of the Macaé formation and some of the identified geological features in the amplitude data: ravines (green arrows), a wide and sinuous canyon (red arrow) and sinkholes (pink arrows). (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

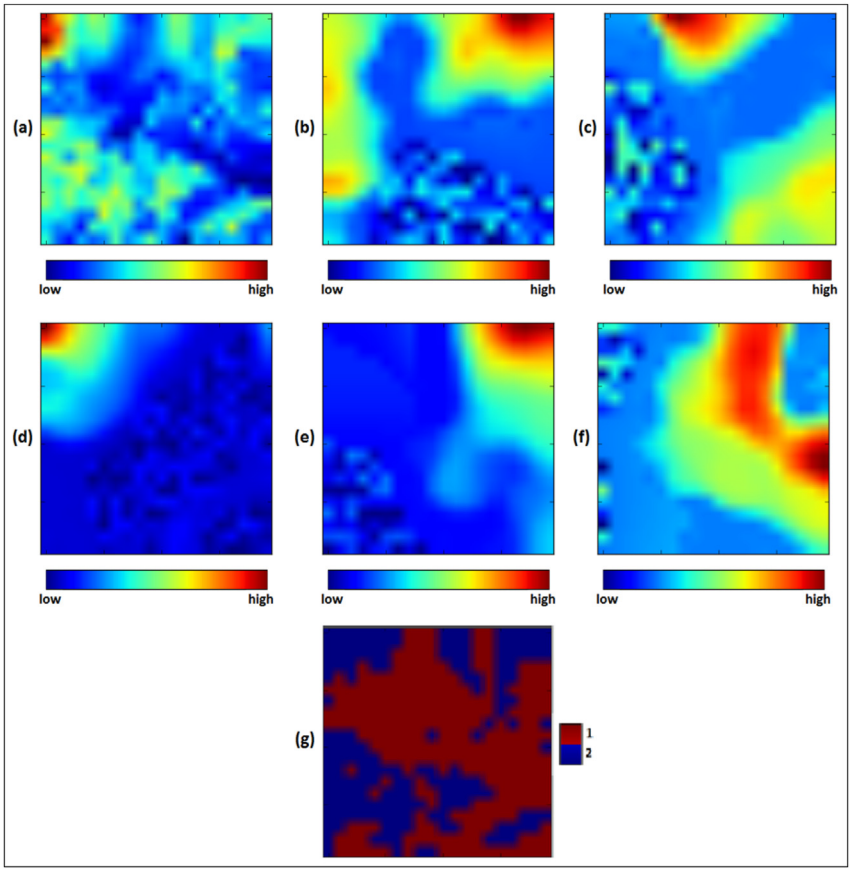

Fig. 6. Maps of distribution: (a) U-matrix, (b) amplitude second derivative, (c) most negative and (d) most positive curvature, (e) envelope-weighted frequency, (f) isopach distribution maps, and (g) labeled neuron map.

- Intel Xeon E5-2630 v3 of 2.40 GHz–8 cores/16 threads.
- 48 GB of RAM memory.
- Nvidia Titan Xp graphic card – 12 GB GDDR5X of dedicated memory, 11.4 GHz of memory speed and 547.7 GB/s of memory bandwidth.

As displayed in Table 3, the experiment itself makes use of only one node. The resulting neuron map should preserve some topological distribution and shows clusters that can be visualized through the U-matrix (Ultsch and Siemon, 1990). In unsupervised classification, the neurons have no label assigned since the input data is unlabeled. Typically, a clustering algorithm is applied in order to partition the neuron map into C clusters and associate a label l to each cluster. In our application, we applied the K-means algorithm (Jain, 2010) to create two clusters (karst and non-karst features). The K-means is one of the simplest clustering algorithms that partitions data in an iterative fashion using k-centroids being k defined a priori. The centroids are defined in the first iteration, but they change along the iterations to the mean point of its cluster.

Giving the labeled neuron map, the classification is performed by assigning the label of the best matching neuron to the input sample. In summary, the best matching neuron is obtained by finding the most

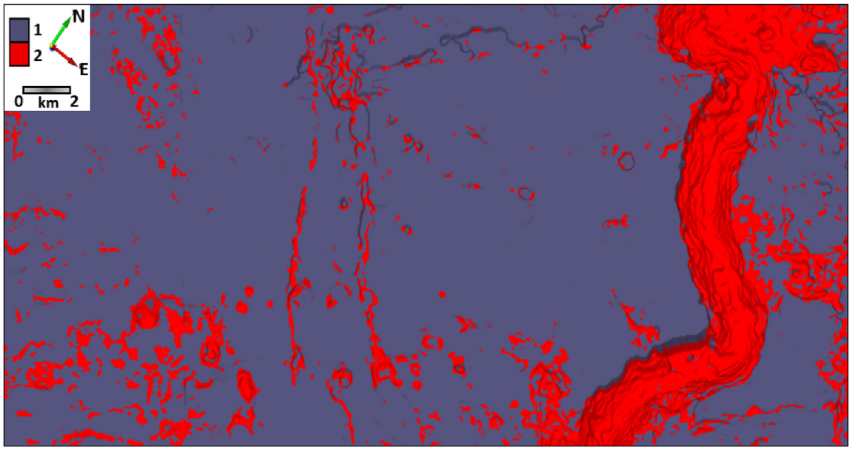

Fig. 7. Classification result at the Macaé top.

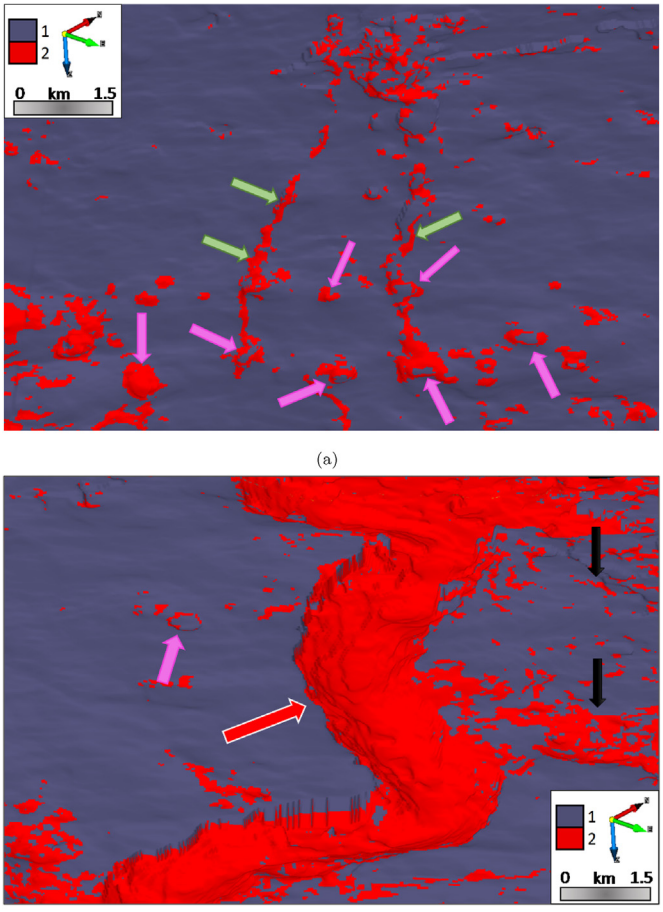

Fig. 8. Classification result in details: (a) ravines and karst features (green and pink arrows, respectively), and (b) canyon (red arrow) and false-positive features (black arrows). (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

Table 4
Computing time comparison.

Algorithm	Computing time (seconds)	Speed-up
SOM/CPU	9549	1.00
SOM/GPU	304	**31.41**

similar neuron to the input sample (*i.e.*, smallest Euclidean distance). As in the unsupervised learning, this step is also executed in the GPU, in which multiple GPU threads will compute simultaneously the best matching neuron for the entire dataset.

7. Case of study: Macaé formation

The seismic data have mapped the top of the Macae Formation, which corresponds to Albian carbonates. The selected study area is located in the southern area of the Campos Basin in Brazil, which is approximately 500 km^2 of area, 14.6 km long (SW-NE) and 34 km wide (NW-SE). Each bin represents a section of size 12.5 × 18.75 m^2 in size, sampled of 4 ms and record length of 5 s. The seismic traces are characterized by 1250 samples, frequency spectrum ranging from 0 to 125 Hz, and 35 Hz as dominant frequency. The working volume has a size of 2.8 GB.

The top of the Macaé formation (Fig. 5) is characterized by closed circular depressions interpreted as sinkholes (pink arrows), a wide and sinuous canyon (red arrow), and less developed erosive features classified as ravines (green arrows), which comprise the features of interest to be identified in the experiment.

7.1. Experiments and discussion

The U-matrix resulting from the unsupervised learning is shown in Fig. 6a. Fig. 6b–f depict the distribution of the weight values in the neuron map respective to each attribute. As aforementioned, the application is treated as a binary classification problem whose samples will be either karst or non-karst features. Given that, the clustering and labeling of the neuron map through the K-means algorithm results in the partitioned map depicted in Fig. 6g. Therefore, neurons are labeled as class 1 (red) or class 2 (blue).

The classification using the best matching approach gives the result shown in Fig. 7. There, we have that class 1 represents the non-karst features and class 2 represents the features of interest. Fig. 8 shows in detail some of the mapped features in the final result.

The delimitation of elements provided by the extraction of class 2 facilitates the process of geological interpretation of a paleohorizon. Examining Fig. 8a, at least 7 sinkholes (pink arrows) can be easily identified and analyzed under a morphometric point of view, having diameters between 70 and 600 m, and depths from 5 to 60 m. In addition, two well-developed ravines (green arrows) show rectilinear pattern and low continuity. These features end in sinkholes, defining connection points between the surface and subsurface hydrology. Class 2 also shows the occurrence of a canyon, highlighting its sinuous pattern (Fig. 8b). This feature has steep-sided walls indicating a preferably vertical development vector. Also, the high symmetry of the canyon

edges becomes evident. The canyon has dimensions ten times greater than the ravines with average width and depth of 1,200 and 100 m, respectively. A few false-positive features can be observed pointed by black arrows in Fig. 8b.

As aforementioned, this work deals with an unsupervised application. The fundamental problem in unsupervised learning is to find clusters, such that samples within each cluster should share some level of similarity. Therefore, it was computed the Silhouette coefficient (SC) Sarle (1991) to provide a qualitative evaluation of the classification result. The SC provides a score of clustering quality, which ranges from -1 to $+1$, being -1 incorrect clustering, 0 the existence of overlapping clusters, and $+1$ highly dense clusters (best result). The experimental result achieved an SC score of 0.41.

Under the computational point of view, the application successfully dealt with the large volume of seismic data, being capable of working with the entire data at once in all steps of the workflow. Regarding the computing time performance, the usage of a GPU provided a huge speed-up, thus reaching a gain of over 31 times compared to the CPU implementation (Table 4). The computing time comprises the time of mapping the seismic data, clustering the neuron map, and classification.

The gain comes from changes made in how the best matching neuron is computed in both training and classification phases, and in the neuron weight vector update. In a non-parallel code, the distance from each input sample $\mathbf{x}$ to each neuron $\mathbf{w}_j$ in the map is computed one by one. In the GPU implementation, the same distance can be computed for all neurons at the same time. This is possible because multiple threads are allocated, being each one responsible for the computing of one distance.

8. Conclusions

The properties of the karst features contained in seismic data make the task of karst identification a very difficult one. Additionally, the great amount of volumetric data that has to be analyzed makes the interpretation very much time-consuming.

Nowadays, applications need to provide useful information in the lowest time. GPU comes as a low-cost option to tackle the problems of volume of data and processing time through its parallel architecture. Allied with this powerful tool, we may add pattern recognition methods that can retrieve insightful information in situations such as the application presented in this paper.

This work explored the potential of GPUs in an application that usually requires high computational resources and achieved promising results. Considering the geological results, the application was able to successfully identify the features of interest on the top of our study area. The computing performance has also achieved satisfactory and important results. The application successfully processed the entire dataset in any step and achieved a huge speed-up against a CPU implementation.

Acknowledgments

We would like to thank Statoil for the financial support and for allowing us to publish this study. We are grateful for the multiclient seismic data provided by PGS. Sinochem is also acknowledged for the permission of this publication. The authors thank the National Center for High Performance Computing (CENAPAD/SP – Brazil) for providing the resources used in the experiments. We also thank both OpendTect and Petrel for the software used in this work.

Appendix A. Supplementary data

Supplementary data related to this article can be found at http://dx.doi.org/10.1016/j.cageo.2018.06.004.

References

Anthopoulos, A., Grimstead, I., Brancale, A., 2013. GPU-accelerated molecular mechanics computations. Journal of Computaional Chemistry 34, 2249–2260.
Brown, N., Lepper, A., Weiland, M., Hill, A., Shipway, B., Maynard, C., 2015. A directive based hybrid met office nerc cloud model. In: Proceedings of the Second Workshop on Accelerator Programming Using Directives. WACCPD '15, pp. 7:1–7:8.
Buschkuehle, B.E., Hein, F.J., Grobe, M., 2007. An overview of the geology of the upper devonian grosmont carbonate bitumen deposit, northern Alberta, Canada. Bull. Can. Petrol. Geol. 16, 3–15.
Chang, H.C., Kopaska-Merkel, D.C., Chen, H.C., 2002. Identification of lithofacies using kohonen self-organizing maps. Comput. Geosci. 28 (2), 223–229.
Che, S., Boyer, M., Meng, J., Tarjan, D., Sheaffer, J.W., Skadron, K., 2008. A performance study of general-purpose applications on graphics processors using CUDA. J. Parallel Distr. Comput. 68 (10), 1370–1380 (General-Purpose Processing using Graphics Processing Units).
Cheng, T., 2013. Accelerating universal kriging interpolation algorithm using CUDA-enabled GPU. Comput. Geosci. 54, 178–183.
Chopra, S.K.M, 2007. Seismic curvature attributes for mapping faults/fractures, and other stratigraphic features. Canadian Society of Exploration Geophysicists (9), 32.
Ersoy, O., Aydar, E., Gourgaud, A., Artuner, H., Bayhan, H., 2007. Clustering of volcanic ash arising from different fragmentation mechanisms using kohonen self-organizing maps. Comput. Geosci. 33 (6), 821–828.
Esteban, M., Wilson, J.L., 1992. Introduction to karst systems and paleokarst reservoirs. Paleokarst Related Hydrocarbon Reservoirs 1–9.
Ford, D.C., Willians, P.W. (Eds.), 1989. Karst Geomorphology and Hidrology, first ed. Springer, Netherlands.
Haykin, S., 1998. Neural Networks: a Comprehensive Foundation, second ed. Prentice Hall PTR, Upper Saddle River, NJ, USA.
Hebb, D.O., 1949. The Organization of Behavior: a Neuropsychological Theory. Wiley New ed ed. New York.
Immenhauser, A., Rameil, N., 2011. Interpretation of ancient epikarst features in carbonate successions — a note of caution. Sediment. Geol. 239 (1–2), 1–9.
Jain, A.K., 2010. Data clustering: 50 years beyond k-means. Pattern Recogn. Lett. 31 (8), 651–666.
Jeong, W.K., Whitaker, R., Dobin, M., 2006. Interactive 3d Seismic Fault Detection on the Graphics Hardware.
Kirk, D.B., Hwu, W.M.W., 2010. Programming Massively Parallel Processors: a Hands-on Approach, first ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
Kohonen, T., 1990. The self-organizing map. Proc. IEEE 78, 1464–1480.
Kohonen, T., 2013. Essentials of the self-organizing map. Neural Network. 37, 52–65. http://dx.doi.org/10.1016/j.neunet.2012.09.018. https://doi.org/10.1016/j.neunet.2012.09.018.
Komatitsch, D., Erlebacher, G., Göddeke, D., Michéa, D., 2010. High-order finite-element seismic wave propagation modeling with mpi on a large GPU cluster. J. Comput. Phys. 229 (20), 7692–7714.
Kuroda, M.C., Vidal, A.C., de Carvalho, A.M.A., 2012. Interpretation of seismic multiattributes using a neural network. J. Appl. Geophys. 85, 15–24.
Lacasta, A., Juez, C., Murillo, J., García-Navarro, P., 2015. An efficient solution for hazardous geophysical flows simulation using GPUs. Comput. Geosci. 78, 63–72.
Li, X., Huang, T., Lu, D.T., Niu, C., 2014. Accelerating experimental high-order spatial statistics calculations using GPUs. Comput. Geosci. 70, 128–137.
Liu, G.F., Liu, H., Li, B., Liu, Q., Tong, X.L., 2009. Method of prestack time migration of seismic data of mountainous regions and its GPU implementation. Chin. J. Geophys. 52 (6), 1381–1388.
Loucks, R.G., 1999. Paleocave carbonate reservoirs; origins, burial-depth modifications, spatial complexity, and reservoir implications. AAPG (Am. Assoc. Pet. Geol.) Bull. 83 (11), 1795–1834.
Luo, P., Machel, H.G., Shaw, J., 1994. Petrophysical properties of matrix blocks of a heterogeneous dolostone reservoir - the upper devonian grosmont formation, Alberta, Canada. Bull. Can. Petrol. Geol. 42, 465–481.
Maltman, A., 2000. Main types of geological maps: purpose, use and preparation French oil and gas industry association, technical committee publisher oxford and ibh publishing, New Delhi and editions technip, paris 1997 (348 pp) ffr. 560 isbn 2-7108-0622-3 and 81-204-0839-x. J. Quat. Sci. 15 (5), 558–559.
Maoshan, C., Shifan, Z., Zhonghong, W., Zhang, H., Li, L., 2011. Detecting carbonate-karst reservoirs using the directional amplitude gradient difference technique. In: 2011 SEG Annual Meeting.
McArt, D.G., Bankhead, P., Dunne, P.D., Salto-Tellez, M., Hamilton, P.,S.-D.,Z., 2013. cudaMap: a GPU accelerated program for gene expression connectivity mapping. BMC Bioinf. 14, 1–6.
Michalakes, J., Vachharajani, M., 2008. GPU acceleration of numerical weather prediction. Parallel Process. Lett. 18 (04), 531–548.
Mojarab, M., Memarian, H., Zare, M., Morshedy, A.H., Pishahang, M.H., 2014. Modeling of the seismotectonic provinces of Iran using the self-organizing map algorithm. Comput. Geosci. 67, 150–162.
NVIDIA, 2010. CUDA parallel computing platform. http://www.nvidia.com/object/cuda_home_new.html#sthash.Zjq5maqO.dpuf.
NVIDIA, 2016. Cublas. http://docs.nvidia.com/cuda/cublas/#axzz4T7JMBIUH.
NVIDIA, 2016. Thrust. http://docs.nvidia.com/cuda/thrust/#axzz4T1V7AD9R.
Open-MPI, 2016. Open mpi: open source high performance computing. https://www.open-mpi.org/.
OpendTect, 2002. Appendix a - Attributes and Filters: Instantaneous. http://doc.opendtect.org/5.0.0/doc/od_userdoc/content/app_a/inst.htm.
OpenGL, 2016. Opengl api Documentation Overview. https://www.opengl.org/

documentation/.
Peter, W., Shi, C.G., Ik, S.L., Li, Z., 2015. Somoclu: an efficient parallel library for self-organizing maps. http://arxiv.org/abs/1305.1422.
Roberts, A., 2001. Curvature attributes and their application to 3d interpreted horizons. First Break 19 (2), 85–100.
Rubio, F., Hanzich, M., Farrés, A., de la Puente, J., Cela, J.M., 2014. Finite-difference staggered grids in GPUs for anisotropic elastic wave propagation simulation. Comput. Geosci. 70, 181–189.
Sarle, W.S., 1991. J. Am. Stat. Assoc. 86 (415), 830–832. http://www.jstor.org/stable/2290430.
Sayago, J., Di Lucia, M., Mutti, M., Cotti, A., Sitta, A., Broberg, K., Przybylo, A., Buonaguro, R., Zimina, O., 2012. Characterization of a deeply buried paleokarst terrain in the loppa high using core data and multiattribute seismic facies classification. AAPG (Am. Assoc. Pet. Geol.) Bull. 96 (10), 1843–1866.
Shi, L., Liu, W., Zhang, H., Xie, Y., Wang, D., 2012. A survey of GPU-based medical image computing techniques. Quant. Imag. Med. Surg. 2, 188–206.
Tahmasebi, P., Sahimi, M., Mariethoz, G., Hezarkhani, A., 2012. Accelerating geostatistical simulations using graphics processing units GPU. Comput. Geosci. 46, 51–59.
Ultsch, A., Siemon, H.P., 1990. Kohonen's self organizing feature maps for exploratory data analysis. In: Proceedings of International Neural Networks Conference (INNC). Kluwer Academic Press, Paris, pp. 305–308.
Waltham, A., Fookes, P., 2003. Engineering classification of karst ground condition. Q. J. Eng. Geol. Hydrogeol. 36 (10), 101–118.
Zhao, T., Jayaram, V., Roy, A., Marfurt, K.J., 2015. A comparison of classification techniques for seismic facies recognition. Interpretation 3 (4), SAE29–SAE58.
Zhao, W., Shen, A., Qiao, Z.J.Z., Wang, X., 2014. Carbonate karst reservoirs of the tarim basin, northwest China: types, features, origins, and implications for hydrocarbon exploration. Interpretation 2, SF65–SF90.