Efficient Unsupervised Distance Learning through Rank Correlation Measures on Heterogeneous Systems

César Yugo Okada, Daniel Carlos Guimarães Pedronette

[1]Department of Statistics, Applied Mathematics and Computing
São Paulo State University (UNESP), Rio Claro, Brazil

{okada,daniel}@rc.unesp.br

Abstract. *The huge growth of image collections have demanded methods capable of conducting effective and efficient image searches. Among the most promising approaches, the Content-Based Image Retrieval (CBIR) systems have established as an alternative for automatically taking into account the visual information. Despite the important results achieved, retrieving relevant images (effectiveness) in minimal time (efficiency) remains a challenge task. Recently, unsupervised learning algorithms have been proposed to improve the effectiveness of CBIR systems by exploiting similarity and ranking information. Such algorithms does not require any user information, but often demand high computational efforts. On the other hand, parallel and heterogeneous approaches constitute a feasible solution for high performance computing. In this paper, we discuss a parallel and accelerated solution for computing the RL-Sim* Algorithm, a recently proposed unsupervised image re-ranking approach. The proposed algorithm uses the OpenCL standard, exploiting both CPU and GPU devives in an Accelerated Processing Unit (APU). The experimental evaluation demonstrated that significant speedups were achieved when compared with the original approach.*

1. Introduction

Advances in social media and image acquisition devices have been triggering, every day, a huge growth of images available in a unstructured way. Considering this scenario, Content-Based Image Retrieval (CBIR) systems have been establishing as a consistent solution for conducting image searches. The main objective of CBIR systems is to organize an image collection taking into account their visual content [Datta et al. 2008]. In general, given a query image, a CBIR system ranks the collection images in decreasing order of similarity according to the query.

However, despite of significant advances of CBIR systems in last decades, effectively measuring the similarity among images remains a challenging problem in image retrieval tasks. In this way, various research efforts have been put in methods for improving the accuracy of retrieval results without the need of user interventions. Actually, various unsupervised learning methods [Yang et al. 2009, Pedronette and da S. Torres 2013] have been achieved significant effectiveness gains. In general, such methods replaces pairwise measures by more global measures which consider underlying contextual information of dataset. A relevant advantage of such methods consists in its capability of improving the effectiveness of retrieval task without requiring any user intervention.

Although effective, these approaches are often computationally expensive and designed to run off-line, since they analyze the relationships among all collection images. Graph approaches based on diffusion process [Yang et al. 2009], for example, presents high complexity, what can turn unfeasible its use in some scenarios. More recently, rank-based unsupervised learning approaches [Pedronette and da S. Torres 2013] have been attracted a lot of research attention due to the smaller computational efforts required.

In addition, heterogeneous and parallel architectures have presented consistent advances. With the evolution of Central Processing Units (CPUs), which currently have various cores, and the Graphics Processing Units (GPUs) used as general-purpose processors (GPGPUs), multi tasks architectures are able to execute hundreds of operations per cycle. Parallel standards, as OpenCL, have also been proposed mainly for exploiting parallel resources available in different devices. Such heterogeneous and parallel architecture advances can be exploited to accelerate unsupervised learning methods, becoming it suitable to on-line applications.

In this paper, we discuss the acceleration of a recent effective distance learning algorithm for image retrieval. The RL-Sim* [Okada et al. 2015] algorithm exploits the similarity between ranked lists through rank correlation measures for improving the image retrieval accuracy. The objective is to accelerated the algorithm in order to allow its use in real time applications. The contributions of this paper are threefold: (*i*) an accelerated RL-Sim* algorithm is proposed, including optimizations for efficient serial execution; (*ii*) a parallel solution is discussed for computing the algorithm using the OpenCL standard; (*iii*) the algorithm is evaluated in heterogeneous environments, considering an Accelerated Processing Unit (APU). Experiments were conducted in three public image datasets. The experimental results demonstrated that the proposed approach improves the efficiency of the algorithm without significant losses of effectiveness.

This paper is organized as follow: Section 2 briefly describes the original RL-Sim* algorithm [Okada et al. 2015]. Section 3 presents the proposed accelerated RL-Sim* algorithm. Section 4 discuss the experimental results. Finally, Section 5 draws the conclusions.

2. RL-Sim* Algorithm

The RL-Sim [Pedronette and da S. Torres 2013] algorithm was proposed based on the conjecture that ranked lists encode useful contextual information for improving the effectiveness of image retrieval tasks. Ranked lists represent a relevant source of contextual information, since users do not consider pairs of images, but the ranked list as a whole. In addition, if two images are similar, their ranked lists should be similar as well.

In this way, the RL-Sim algorithm computes a new distance/similarity score between images by analyzing the similarity between their respectively ranked lists considering the top-k positions. Formally, given two images img_i, img_j, a rank correlation measure $d(\tau_i, \tau_j, k)$ is computed between their respective ranked lists τ_i, τ_j. Based on the rank correlation measure, a new ranked lists can be computed and the process can be iteratively repeated. Figure 1 [Pedronette and da S. Torres 2013] illustrates the evolution of ranked lists along the iterations. The query image is illustrated in green borders and the wrong results in red borders.

The most significant effectiveness gains are obtained at initial positions of ranked

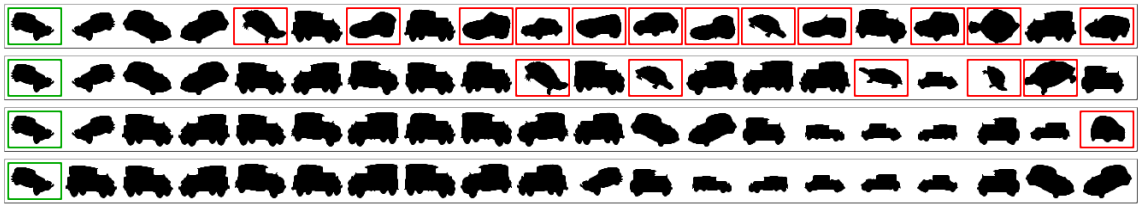

Figure 1. Iterative evolution of ranked lists - RL-Sim Algorithm.

lists. It occurs since is very unlikely to found similar images at the end of ranked lists. Therefore, the distances are redefined considering the rank correlation measure $d(\tau_i, \tau_j, k)$ for the first L positions of the each ranked list, such that $L \in \mathbb{N}$ and $k \leq L \ll N$. For images in the remaining positions of the ranked lists, the new distance is redefined based on the current distances (or rank positions). As a result, this step of the algorithm depends only on a constant L, and not on the collection size N.

Although this approach allows for decreasing the demanded computational costs, it still presents a limitation. Since the rank correlation measures are computed considering the top-k positions their accuracy tends to be low when there is no overlap between the ranked lists being compared at top positions. For measures based on intersection analysis, it is still more critical, producing the same distance values for all pairs of images without overlap at top-k positions. In theses situations, the effectiveness of distance can be worsened.

Based on this observation, the *RL-Sim* Algorithm* [Okada et al. 2015] was proposed. The *RL-Sim* Algorithm* [Okada et al. 2015] computes a different distance when there is no overlap between top-k positions. In this way, considering a query image img_i the ranked list τ_i is divided in three segments. Let $\mathcal{N}(i)$ be the set of k most similar images to img_i and let $\tau_i(j)$ denotes the position of img_j in the ranked list τ_i, the segments are defined as follows:

(i) First segment: this segment contains an image img_j if $(\tau_i(j) < L) \wedge (\mathcal{N}(i) \cap \mathcal{N}(j) \neq \emptyset)$. For these cases the new distance between img_i and img_j is computed by the rank correlation measure.

(ii) Second Segment: if the img_j appears at top-L positions of τ_i, but there is no overlap between top-k positions $(\mathcal{N}(i) \cap \mathcal{N}(j) \neq \emptyset)$, the new distance should segmented from that computed by the rank correlation measures. It is done by incrementing 1 to the current distance.

(iii) Remaining Images: the remaining images, i.e., images after top-L positions are also segmented adding a constant value (defined as 2) to current distance.

As a result, the distance computed based on the segmented rank list presents a higher retrieval accuracy. Other relevant contribution of the RL-Sim* Algorithm [Okada et al. 2015] consists in the use and evaluation of several rank correlation measures.

3. Accelerated RL-Sim*

In this work, we aim at exploiting some characteristics of the RL-Sim* Algorithm [Okada et al. 2015] for proposing an accelerated and parallel re-ranking algorithm. While studies were conducted about the efficiency of the RL-Sim Algorihtm [Pedronette et al. 2013],

the RL-Sim* Algorithm still lacks such analysis. The objective is to analyse the trade-off between effectiveness and efficiency, discussing optimizations which present high impact on efficiency, without significant loss in effectiveness.

This section is organized as follows: Section 3.1 discusses rank correlation measures and its impact on efficiency aspects. Section 3.2 presents the parallel model and Section 3.3 discusses the proposed optimizations.

3.1. Rank Correlation Measures

The computing of rank correlation measure represents the most costly step of the algorithm. Therefore, it impacts severely the execution time and efficiency aspects. Acceleration strategies of the RL-Sim Algorihtm [Pedronette et al. 2013] considered only the intersection measure. In this work, we use the Jaccard measure, which presents a significant lower computational cost. Both rank correlation measures are discussed in the following.

3.1.1. Intersection Measure

The intersection measure [Fagin et al. 2003] captures the extent of overlap between τ_i and τ_j. This measure considers not only the overlap at depth k, but also the cumulative overlap at increasing depth. For each depth $d \in \{1 \ldots k\}$, it is computed the overlap at d, and then those overlaps are averaged to derive a similarity measure. The measure assigns higher weights to the first positions of top k lists, which are considered many times. Equation 1 formally defines the intersection similarity measure ψ.

$$\psi(\tau_i, \tau_j, k) = \frac{\sum_{d=1}^{k} \mid \mathcal{N}(i,d) \cap \mathcal{N}(j,d) \mid}{k} \tag{1}$$

3.1.2. Jaccard

The Jaccard coefficient is a well-known distance between sets. Given two non-empty sets, it measures the probability that an element of at least one of two sets is an element of both, and thus is a reasonable measure of similarity or overlap between the two. The Jaccard Coefficient is defined as follows:

$$J(\tau_i, \tau_j, k) = \frac{|\mathcal{N}(i,k) \cap \mathcal{N}(j,k)|}{|\mathcal{N}(i,k) \cup \mathcal{N}(j,k)|}, \tag{2}$$

3.2. Parallel RL-Sim* Algorithm

The RL-Sim* Re-Ranking [Okada et al. 2015] Algorithm presents great potential for parallelism. A large number of comparisons between ranked lists are executed at each iteration and they do not depend on each other for processing. The re-ranking step, which is given by the re-sort of ranked lists, is also completely independent and can also be computed in parallel. Next sections discusses the parallel solution proposed.

3.2.1. OpenCL

The OpenCL [Stone et al. 2010] (Open Computing Language) is a language for task-parallel and data-parallel programs on heterogeneous platforms, like CPUs, GPUs and others architectures. The OpenCL can be used to improve the performance of several applications in different areas, such as games, medical or academic research.

A program is divided into kernels, which are functions declared in OpenCL language. Parallel executions of a kernel are invoked by a command and can be executed different devices. Each instance of a kernel running on a device is called work-item. The work-items are organized in dimension spaces and execute the same kernel on different data independently.

3.2.2. Parallel Model

The proposed parallel model divides the original RL-Sim* Re-Ranking Algorithm [Okada et al. 2015] in OpenCL kernels. The organization in kernels are also relevant for a correct execution order, ensuring the correctness of data dependences between different steps. For example, the re-sort of ranked lists can only be computed when the distance among images are re-computed based on rank correlation measures. The kernels which composes the proposed parallel model are:

(i) Update of Distances (Up. Dists): this kernel computes new distances among images, based on a rank correlation measure (Jaccard measure). The kernel is executed in 2 dimensions of $N \times L$ work-items. Each work-item computes the distance between a pair of images. Only the top-L images of each dataset have their distances updated. This kernel presents the more costly step of the algorithm.

(ii) Sorting Ranked Lists (Sort RkLists): once the update of distances among images are complete, a new set of ranked lists can be computed. The ranked lists need to be re-sorted based on the new distance. This kernel is execute by N work-items (one for each ranked list)

(iii) Initializing/Normalizing Distances (Init/Norm. Dists): this step is needed to initialize the data structures and normalize distances among images. In previous acceleration strategies of the RL-Sim Algorihtm [Pedronette et al. 2013], this step is serially executed, requiring several OpenCL memory transfers. In this work, aiming at avoiding the intermediary memory transfers, the normalization is not executed and the initializing of structures are made in parallel way, using N work-items.

Figure 2 illustrates a comparison between previous acceleration strategies [Pedronette et al. 2013] and the proposed parallel model. In Figure 2(a), both the initialization and normalization steps serially executed. In Figure 2(b), we can observe that the initialization is executed in parallel requiring no memory transfers between iterations. The normalization step is discarded in the proposed model, which can affects the effectiveness of the algorithm. However, as discussed in experimental evaluation (Section 4), the losses in effectiveness are very small.

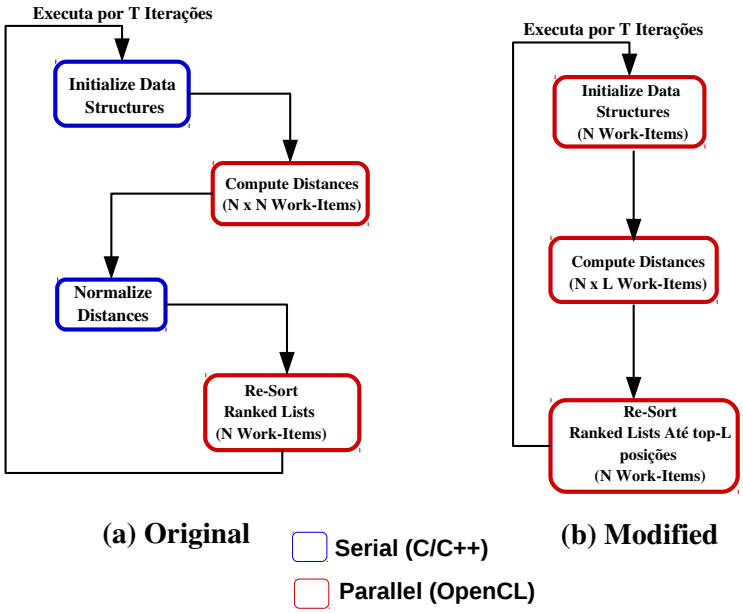

Figure 2. Parallel Model proposed for the RL-Sim* Algorthm.

3.3. Algorithm Optimizations

In addition to the parallel model, other optimizations were also proposed for the acceleration of RL-Sim* Algorithm. Such optimizations aims at minimizing the computational cost of the algorithm, without significant losses in effectiveness. The next sections presents the proposed optimizations.

3.3.1. Re-Ranking and Storage of Ranked Lists until top-L positions

The input of the RL-Sim* Algorithm is a set of ranked lists, which are processed through a ranking correlation measure for computing new distances among images. Previous accelerations strategies [Pedronette et al. 2013] compute and storage a new distance among all images in the dataset, resulting in a squared $N \times N$ distance matrix.

The first optimization proposed for acceleration of RL-Sim* algorithm consists in the storage and computing of only the top-L positions of ranked lists. Instead of $N \times N$ storage requirements, it becomes only $N \times L$. Therefore, not only the use of memory is reduced but also the time required for memory transfers.

After the update distances among images, the ranked lists need to be re-sorted. Once only the top-L positions of ranked lists are stored, the ranked lists are also re-sorted only until the top-L positions. In this way, the accelerated RL-Sim* Algorithm obtain a significant efficiency gain, since the sorting step represents a relevant workload. Figure 2 also illustrated the proposed accelerated strategy in comparison with previous approaches.

3.3.2. Removing Normalization Step and Memory Transfers

The distances computed by the RL-Sim and RL-Sim* algorithms are not symmetric, implying that the distance from img_i to img_j can be different from img_j to img_i. An initial

solution for this drawback is a normalization step, which defines both distances equal to the smaller distance. This process was serially computed, requiring intermediary OpenCL memory transfers [Pedronette et al. 2013].

We proposed a different accelerated solution to optimize this step. Once the distance between img_i and img_j is computed, the algorithm also defines the distance between img_j and img_i with the same value, dispensing the normalization process. In this way, the intermediary OpenCL memory transfers are also avoided. Despite the gains in efficiency, it also affects effectiveness, as discussed in experimental evaluation (Section 4).

3.3.3. Initializing Distances

Before each iteration of the algorithm, it is necessary to initialize the distances among images. In the proposed algorithm, it was implemented using new kernel exclusive for this task. In previous work [Pedronette et al. 2013], this step was performed in serial mode, requiring the transference of the distances from the OpenCL device (parallel) to the host (serial) and back to the parallel code. To optimize this step, it was implemented using new kernel with N work-items. Besides the acceleration obtained by the parallelization, efficiency gains are obtained since memory transfers are avoided. As illustrated in Figure 2, notice that all steps of the algorithm are parallelized.

4. Experimental Evaluation

This section presents the experimental evaluation conducted for assessing the efficiency and effectiveness of the proposed approach. Section 4.1 presents the experimental setup. Section 4.2 presents the results focused in efficiency aspects and Section 4.3 discusses the impact on effectiveness.

4.1. Experimental Setup

For the hardware environment, the experiments were conducted considering the AMD A10-6800K APU, which combines 4 CPU cores and an AMD Radeon HD 8670D. For the software environment, a Linux Ubuntu 14.04 with OpenCL 2.0 AMD-APP is considered. The algorithms was compiled with g++ 4.8.4 using the flag "-O3".

The experimental evaluation was conducted considering four datasets with different characteristics and size ranging from 280 to 10,200 images. Diverse descriptors which consider shape, color and texture were used. Table 1 presents details about the datasets and descriptors. For the effectiveness evaluation, all images of each dataset are considered as query images. For most of datasets, the Mean Average Precision (MAP) is used as effectiveness measure. Only for the N-S dataset uses the N-S score [van de Weijer and Schmid 2006].

The efficiency evaluation of the proposed algorithm considers the execution time of serial and OpenCL parallel implementation, considering CPU and GPU devices. The experiments considered the execution time of different kernels and memory transfers. The comparison with previous acceleration strategies [Pedronette et al. 2013] are also reported. Regarding parameters, we used T =3 and K =15 for all datasets, except for

UKBench [Nistér and Stewénius 2006] dataset which used T =1 and K =5. Each experiment was executed 10 times and the average results are reported besides the confidence intervals at 95%.

Table 1. Datasets and images descriptors used in the experimental evaluation.

Size	Type	Description	Descriptors	Effectiv. Measure
		Soccer [van de Weijer and Schmid 2006]		
280	Color Scenes	Dataset composed of images from 7 soccer teams, containing 40 images per class	Border/Interior Pixel Classification (BIC) [Stehling et al. 2002], Auto Color Correlograms (ACC) [Huang et al. 1997], and Global Color Histogram (GCH) [Swain and Ballard 1991]	MAP
		MPEG-7 [Latecki et al. 2000]		
1,400	Shape	A well-known dataset composed of 1400 shapes divided in 70 classes. Commonly used for evaluation of unsupervised distance learning approaches	Segment Saliences (SS) [da S. Torres and Falcão 2007], Beam Angle Statistics (BAS) [Arica and Vural 2003], Inner Distance Shape Context (IDSC) [Ling and Jacobs 2007], Contour Features Descriptor (CFD) [Pedronette and da S. Torres 2010], Aspect Shape Context (ASC) [Ling et al. 2010], and Articulation-Invariant Representation (AIR) [Gopalan et al. 2010]	MAP
		UKBench [Nistér and Stewénius 2006]		
10,200	Objects/ Scenes	Composed of 2,550 objects or scenes. Each object/scene is captured 4 times from different viewpoints, distances and illumination conditions	ACC [Huang et al. 1997], BIC [Stehling et al. 2002], Color and Edge Directivity Descriptor (CEED) [Chatzichristofis and Boutalis 2008a], Fuzzy Color and Texture Histogram (FCTH) [Chatzichristofis and Boutalis 2008b], Joint Composite Descriptor (JCD) [Zagoris et al. 2010], Scale-Invariant Feature Transform (SIFT) [Lowe 1999]	N-S score

4.2. Efficiency of Accelerated RL-Sim *

Experiments were conducted in different datasets aiming at evaluating the efficiency of the accelerated *RL-Sim* Algorithm*. Table 2 presents the results obtained by the accelerated algorithm considering CFD [Pedronette and da S. Torres 2010] descriptor on MPEG-7 [Latecki et al. 2000] dataset. The results obtained by previous accelerations initiatives [Pedronette et al. 2013] of the RL-Sim algorithm are reported for comparison purposes. We can observe that the optimizations implemented on original algorithm results in a significant gain in efficiency for all kernels executed in parallel or serial mode.

The update distances kernel, which consists in the major computational effort of the algorithm, obtained a significant efficiency gain and the smaller execution time required only 0.4339s (OpenCL CPU). The speedup of this kernel is 1.66× when compared with the original RL-Sim algorithm (0.7186s) and 1.93× when compared with the accelerated serial implementation (0.8376s). The best speedup is 5.99×, which occurs considering the sort of ranked lists for original (0.0575s) and accelerated (0.0096s) algorithm on CPU.

Table 3 presents the results considering ACC [Huang et al. 1997] descriptor on Soccer [van de Weijer and Schmid 2006] dataset. We can observe that the accelerated algorithm obtained a speedup of 2.68× in serial mode and 1.63× in CPU, when compared with the original algorithm. In addition, the update of distances kernel of the accelerated algorithm obatined a speedup of 2.89× considering the time execution on GPU (0.0198s) and on serial (0.0573s).

Table 2. Efficiency Evaluation on MPEG-7 dataset (CFD descriptor).

		Serial	*OpenCL-CPU*			*OpenCL-GPU*		
	Kernel	**Exec.Time**	**Exec.Time**	**Mem. Transf**	**Total**	**Exec.Time**	**Mem. Transf**	**Total**
RL-Sim*	**Up. Dists**	2.0582 ±0.0033	0.6739 ±0.0017	0.0430 ±0.0011	0.7186 ±0.0017	0.6537 ±0.0009	0.0458 ±0.0016	0.7002 ±0.0019
	Sort. RkLists	0.0820 ±0.0003	0.0254 ±0.0008	0.0288 ±0.0003	0.0575 ±0.0009	1.1690 ±0.0009	0.03478 ±0.001377	1.2080 ±0.0019
	Init./Norm. Dists.	0.0182 ±0.0003	0.0187 ±0.0003	0.0000 ±0.0000	0.0187 ±0.0003	0.0269 ±0.0015	0.0000 ±0.0000	0.0269 ±0.0015
	Total Time	2.1585 ±0.0032	0.7181 ±0.0023	0.0718 ±0.0011	0.7949 ±0.0021	1.8500 ±0.0017	0.0806 ±0.0008	1.9350 ±0.0026
Accel. RL-Sim*	**Up. Dists**	0.8376 ±0.0003	0.4338 ±0.00104	0.0000 ±0.0000	0.4339 ±0.0010	0.5977 ±0.0024	0.0000 ±0.0000	0.5981 ±0.0024
	Sort. RkLists	0.0240 ±0.0001	0.0074 ±0.0000	0.0001 ±0.0000	0.0096 ±0.0001	0.6236 ±0.0007	0.0028 ±0.0013	0.6305 ±0.0018
	Init./Norm. Dists.	0.0183 ±0.0001	0.0115 ±0.0002	0.0001 ±0.0000	0.0117 ±0.0002	0.1105 ±0.0001	0.0014 ±0.0003	0.1122 ±0.0002
	Total Time	0.8802 ±0.0004	0.4527 ±0.0011	0.0001 ±0.0000	**0.4553 ± 0.0012**	1.3320 ±0.0027	0.0042 ±0.0013	1.3410 ±0.0039

Table 3. Efficiency Evaluation on Soccer dataset (ACC descriptor).

		Serial	*OpenCL-CPU*			*OpenCL-GPU*		
	Kernel	**Exec.Time**	**Exec.Time**	**Mem. Transf**	**Total**	**Exec.Time**	**Mem. Transf**	**Total**
RL-Sim*	**Up. Dists**	0.1627 ±0.0002	0.0414 ±0.0008	0.0017 ±0.0001	0.0436 ±0.0009	0.0687 ±0.0004	0.0037 ±0.0009	0.0730 ±0.0001
	Sort. RkLists	0.0046 ±0.0001	0.0017 ±0.0004	0.0013 ±0.0001	0.0038 ±0.0001	0.1837 ±0.0001	0.0030 ±0.0008	0.1879 ±0.0002
	Init./Norm. Dists.	0.0007 ±0.0002	0.0007 ±0.0003	0.0000 ±0.0000	0.0007 ±0.0003	0.0011 ±0.0005	0.0000 ±0.0000	0.0011 ±0.0005
	Total Time	0.1682 ±0.0002	0.0438 ±0.0009	0.00297 ±0.0002	0.04815 ±0.0008	0.2536 ±0.0001	0.0067 ±0.0001	0.2620 ±0.0002
Accel. RL-Sim*	**Up. Dists**	0.0573 ±0.0005	0.0258 ±0.0017	0.0000 ±0.0000	0.0259 ±0.0017	0.0195 ±0.0001	0.0000 ±0.0000	0.0198 ±0.0001
	Sort. RkLists	0.0043 ±0.0002	0.0017 ±0.0002	0.0008 ±0.0002	0.0024 ±0.0003	0.1903 ±0.0002	0.0004 ±0.0003	0.1916 ±0.0002
	Init./Norm. Dists.	0.0000 ±0.0000	0.0009 ±0.0005	0.0001 ±0.0000	0.0011 ±0.0006	0.0054 ±0.0006	0.0002 ±0.0000	0.0059 ±0.0007
	Total Time	0.0626 ±0.0006	0.0285 ±0.0019	0.0001 ±0.0000	**0.0295 ± 0.0020**	0.2153 ±0.0003	0.0006 ±0.0003	0.2173 ±0.0003

Table 4. Efficiency Evaluation on UKBench dataset (ACC descriptor).

		Serial	*OpenCL-CPU*			*OpenCL-GPU*		
	Kernel	**Exec.Time**	**Exec.Time**	**Mem. Transf**	**Total**	**Exec.Time**	**Mem. Transf**	**Total**
RL-Sim*	**Up. Dists**	0.3116 ±0.0006	0.0634 ±0.0022	0.0085 ±0.0004	0.0740 ±0.0019	0.0611 ±0.0005	0.0082 ±0.0009	0.0738 ±0.0010
	Sort. RkLists	0.0105 ±0.0007	0.0046 ±0.0004	0.0018 ±0.0008	0.0145 ±0.0005	0.1212 ±0.0005	0.0037 ±0.0001	0.1332 ±0.0002
	Init./Norm. Dists.	0.0027 ±0.0003	0.0044 ±0.0001	0.0000 ±0.0000	0.0044 ±0.0001	0.0032 ±0.0007	0.0000 ±0.0000	0.0032 ±0.0007
	Total Time	0.3250 ±0.0007	0.0724 ±0.0023	0.01973 ±0.000	0.0929 ±0.0020	0.1855 ±0.0007	0.0242 ±0.0010	0.2102 ±0.0009
Accel. RL-Sim*	**Up. Dists**	0.0900 ±0.0004	0.0453 ±0.0047	0.0001 ±0.0002	0.0454 ±0.0047	0.3398 ±0.0093	0.0019 ±0.0007	0.3420 ±0.0093
	Sort. RkLists	0.0100 ±0.0001	0.0039 ±0.0001	0.0003 ±0.0005	0.0040 ±0.0001	0.2884 ±0.0002	0.0007 ±0.0001	0.2920 ±0.0018
	Init./Norm. Dists.	0.0028 ±0.0000	0.0043 ±0.0008	0.0000 ±0.0000	0.0043 ±0.0008	0.0030 ±0.0007	0.0000 ±0.0000	0.0030 ±0.0007
	Total Time	0.1028 ±0.0004	0.0535 ±0.0047	0.0004 ±0.0000	**0.05383 ± 0.005**	0.6312 ±0.0098	0.0026 ±0.0006	0.6370 ±0.0102

The proposed algorithm was also evaluated considering ACC [Huang et al. 1997] descriptor on the UKBench dataset [Nistér and Stewénius 2006]. The superiority of accelerated algorithm, when compared with original algorithm can be observed if we consider the best total time execution between the proposed (0.0538s) and original (0.0929s) algorithm. The most efficient execution occurs in CPU, presenting a speedup of 1.91× in relation of serial code (0.1028s). The efficiency of the proposed algorithm is evident in this experiment because the time execution presented is very low considering the size of the dataset (10,200 images).

4.3. Effectiveness of Accelerated RL-Sim *

In this section, we evaluate the effectiveness of the accelerated RL-Sim* and the impacts caused by efficiency optimizations. The main variations in effectiveness scores occurs mainly because of the removal the normalization step. The effectiveness experiment was conducted considering different datasets and features (shape, color, texture). In the experiments, the RL-Sim* Algorithm was evaluated with Intersection and Jaccard measure and the accelerated RL-Sim* with Jaccard measure. The choose of Jaccard to accelerated algorithm is due to the low computational efforts and little effectiveness loss when compared with Intersection measure.

Table 5 presents the effectiveness of the original algorithm and the accelerated algorithm. The proposed algorithm presents a very significant gains in most of descriptors when compared with the original MAP of each descriptor. We can also observe that the proposed method obtained similar results when compared with de original method.

Table 6 presents the score obtained by the accelerated and original algorithms con-

sidering the UKBench dataset and N-S score as effectiveness measure. Each image is used as query and the N-S score [Nistér and Stewénius 2006] between 1 and 4 is computed. The score corresponds to the number of relevant images among the first four image returned (the highest achievable score is 4). The proposed algorithm a significant gain when compared with the original score and obtained similar scores to original algorithm.

Table 5. Effectiveness evaluation on Soccer and MPEG-7 datasets (MAP score).

			RL-Sim*		Accelerated RL-Sim*	
Descriptor	**Type**	**Initial MAP**	**Intersection**	**Jaccard**	**Jaccard**	**Relative Gain (%)**
SS	Shape	37.67	44.10	45.49	44.12	+ 17.12
BAS	Shape	71.52	76.05	74.87	74.51	+ 4.18
IDSC	Shape	81.70	87.38	87.03	87.10	+ 6.61
CFD	Shape	80.71	90.15	89.51	89.03	+10.31
ASC	Shape	85.28	89.96	89.54	89.46	+ 4.90
AIR	Shape	89.39	96.17	97.72	97.59	+ 9.83
GCH	Color	32.24	33.99	33.43	33.06	+ 2.54
ACC	Color	37.23	45.19	45.63	46.20	+ 24.09
BIC	Color	39.26	45.42	44.56	44.99	+ 14.60

Table 6. Effectiveness evaluation on UKBench dataset (N-S score).

		RL-Sim*		Accelerated RL-Sim*
Descriptor	**Initial Score**	**Intersec.**	**Jaccard**	**Relative Gain (%)**
ACC	3.36	3.54	3.47	+ 3.27
BIC	3.04	3.20	3.13	+ 2.96
CEED	2.61	2.75	2.68	+ 2.68
FCTH	2.73	2.84	2.77	+ 1.46
JCD	2.79	2.92	2.85	+ 2.15
SIFT	2.54	2.81	2.77	+ 9.05
Average	2.84	3.01	2.94	-

Figure 3 presents a joined analysis of effectiveness and efficiency, comparing the RL-Sim and the accelerated RL-Sim* algorithms. The analysis consider the MPEG-7 and CFD shape descriptor. The position of algorithms in the graph is given by the effectiveness score and the run time, such that an algorithm with high effectiveness and low run time is positioned at the top-left corner of the graph. Notice that the proposed approach (in red) occupies very good positions.

5. Conclusions

In this paper, we present a parallel and accelerated model for the RL-Sim* Algorithm, a recently re-ranking approach for CBIR applications. The proposed parallel solution uses the OpenCL standard, GPUs and multi-core CPUs to accelerate the computation of the RL-Sim*. The proposed algorithm was evaluated considering both effectiveness and efficiency aspects.

The accelerated algorithm obtained a similar efficiency performance when compared with the original algorithm. Considering the original retrieval performance of different image features, the proposed approach presents significant gains in MAP and N-S score. On other hand, the proposed accelerated algorithm achieved a speedup of 1.66×

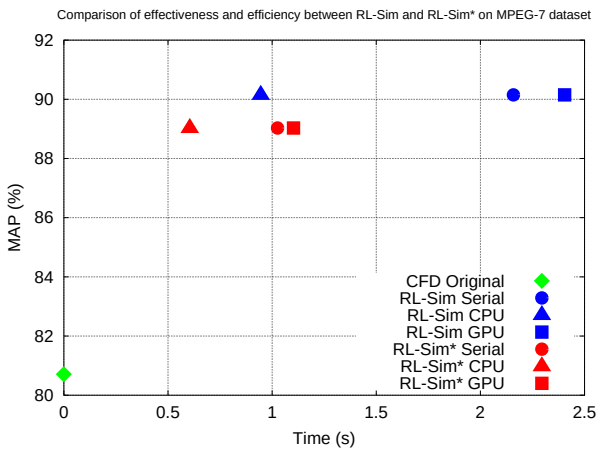

Figure 3. Effectiveness and Efficiency evaluation on MPEG-7 dataset.

in update distance and 5.66× in re-sort ranked lists. The experiment conducted in N-S dataset demonstrated that the proposed approach yields significant effectiveness gains requiring very a small time execution, even for 10,200 images.

Future work includes extending the proposed algorithm to divide the workload simultaneously on CPU and GPU. We also intend to further evaluate the proposed approach, including comparisons with other parallel APIs as OpenMP and executing it in larger datasets to analyze the scalability and other compromises between effectiveness and efficiency.

6. Acknowledgments

The authors are grateful to PROPe/UNESP and São Paulo Research Foundation - FAPESP (grant 2013/08645-0).

References

Arica, N. and Vural, F. T. Y. (2003). BAS: a perceptual shape descriptor based on the beam angle statistics. *Pattern Recognition Letters*, 24(9-10):1627–1639.

Chatzichristofis, S. A. and Boutalis, Y. S. (2008a). Cedd: color and edge directivity descriptor: a compact descriptor for image indexing and retrieval. In *Proceedings of the 6th international conference on Computer vision systems*, ICVS'08, pages 312–322.

Chatzichristofis, S. A. and Boutalis, Y. S. (2008b). Fcth: Fuzzy color and texture histogram - a low level feature for accurate image retrieval. In *WIAMIS '08*, pages 191–196.

da S. Torres, R. and Falcão, A. X. (2007). Contour Salience Descriptors for Effective Image Retrieval and Analysis. *Image and Vision Computing*, 25(1):3–13.

Datta, R., Joshi, D., Li, J., and Wang, J. Z. (2008). Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):5:1–5:60.

Fagin, R., Kumar, R., and Sivakumar, D. (2003). Comparing top k lists. In *ACM-SIAM Symposium on Discrete algorithms (SODA'03)*, pages 28–36.

Gopalan, R., Turaga, P., and Chellappa, R. (2010). Articulation-invariant representation of non-planar shapes. In *ECCV'2010*, volume 3, pages 286–299.

Huang, J., Kumar, S. R., Mitra, M., Zhu, W.-J., and Zabih, R. (1997). Image indexing using color correlograms. In *CVPR*, pages 762–768.

Latecki, L. J., Lakmper, R., and Eckhardt, U. (2000). Shape descriptors for non-rigid shapes with a single closed contour. In *CVPR*, pages 424–429.

Ling, H. and Jacobs, D. W. (2007). Shape classification using the inner-distance. *PAMI*, 29(2):286–299.

Ling, H., Yang, X., and Latecki, L. J. (2010). Balancing deformability and discriminability for shape matching. In *ECCV*, volume 3, pages 411–424.

Lowe, D. (1999). Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157.

Nistér, D. and Stewénius, H. (2006). Scalable recognition with a vocabulary tree. In *CVPR*, volume 2, pages 2161–2168.

Okada, C. Y., Pedronette, D. C. G., and Torres, R. d. S. (2015). Unsupervised distance learning by rank correlation measures for image retrieval. In *ICMR '15*.

Pedronette, D. C. G. and da S. Torres, R. (2010). Shape retrieval using contour features and distance optmization. In *VISAPP*, volume 1, pages 197 – 202.

Pedronette, D. C. G. and da S. Torres, R. (2013). Image re-ranking and rank aggregation based on similarity of ranked lists. *Pattern Recognition*, 46(8):2350–2360.

Pedronette, D. C. G., da S. Torres, R., Borin, E., and Breternitz, M. (2013). Image Re-ranking Acceleration on GPUs. In *SBAC-PAD*.

Stehling, R. O., Nascimento, M. A., and Falcão, A. X. (2002). A compact and efficient image retrieval approach based on border/interior pixel classification. In *CIKM*, pages 102–109.

Stone, J. E., Gohara, D., and Shi, G. (2010). OpenCL: A parallel programming standard for heterogeneous computing systems. *Computing in Science Engineering*, 12(3):66 –73.

Swain, M. J. and Ballard, D. H. (1991). Color indexing. *International Journal on Computer Vision*, 7(1):11–32.

van de Weijer, J. and Schmid, C. (2006). Coloring local feature extraction. In *ECCV*, pages 334–348.

Yang, X., Koknar-Tezel, S., and Latecki, L. J. (2009). Locally constrained diffusion process on locally densified distance spaces with applications to shape retrieval. In *CVPR*, pages 357–364.

Zagoris, K., Chatzichristofis, S., Papamarkos, N., and Boutalis, Y. (2010). Automatic image annotation and retrieval using the joint composite descriptor. In *14th Panhellenic Conference on Informatics (PCI)*, pages 143–147.