A New Division Operator to Handle Complex Objects in Very Large Relational Datasets

André S. Gonzaga
University of São Paulo - ICMC
São Carlos, Brazil
asgonzaga@usp.br

Robson L. F. Cordeiro
University of São Paulo - ICMC
São Carlos, Brazil
robson@icmc.usp.br

ABSTRACT

In Relational Algebra, the operator Division ($\div$) is an intuitive tool used to write queries with the concept of "for all", and thus, it is constantly required in real applications. However, the division does not support many of the needs common to modern applications, particularly those that involve complex data analysis, such as processing images, audio fingerprints, and many other "non-traditional" data types. The main issue is intrinsic comparisons of attribute values, which, by the very definition, are always performed by *identity* ($=$) in the division, while complex data must be compared by *similarity*. Recent works focus on supporting similarity comparison in relational operators, but our work is the first one to treat the division. This paper presents the new **Similarity-aware Division ($\hat{\div}$)** operator. Our novel operator is naturally well suited to answer queries with an idea of "candidate elements and exigencies" to be performed on complex data from real applications of high impact. For example, it can support agriculture, as we demonstrate through a case study in this paper.

CCS Concepts

•Information systems → **Query operators;** **•Theory of computation** → **Database query processing and optimization;** **•Applied computing** → *Agriculture;*

Keywords

Relational Division; Similarity Comparison; Complex Data

1. INTRODUCTION

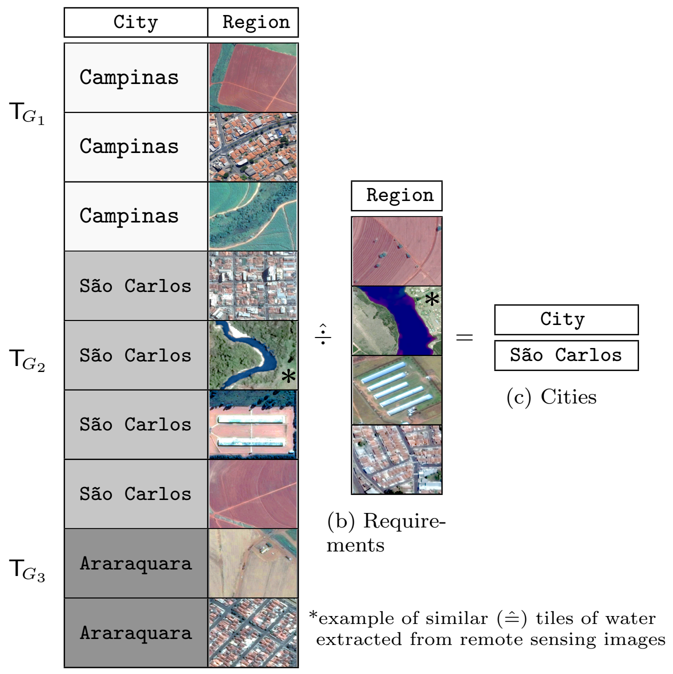
The Relational Algebra [3] defines a number of operators to express queries on relations.The Division ($\div$) has an important role in this context because it is the simplest and most intuitive way to represent queries with the idea of "for all", besides being the **only** algebraic operator that directly corresponds to the Universal Quantification ($\forall$) from the Relational Calculus [3]. As a consequence, the division is required in queries that are commonly performed by real applications. For example, it answers the queries as follows: (a) "*What products have **all** requirements of the industrial quality control?*", (b) "*What bank clients paid **all** bills of their loans?*", (c) "*What cities have **all** the requirements to produce a given type of crop?*". Unfortunately, the division is restricted to work with traditional data only.

In this paper, we identify severe limitations on the usability of the division to "non-traditional" data types that are commonly used in modern applications. Today, many real data sets include, besides the traditional numeric values and small texts, more complex data objects such as images, audio, videos, time series, long texts, fingerprints, and many others [8, 6]. One central distinction between traditional and complex data is that the latter must be compared by *similarity*, since comparisons by *identity* are in most cases senseless or unfeasible for data of a more complex nature.

Let us use the query of selecting cities well-suited to produce a given type of crop to exemplify the limitations of the division. Figure 1 illustrates a toy dataset in which the concept of division is required, but the existing operator cannot handle. Relation `CityRegions` describes three cities, i.e., the candidates to produce the crop, each one represented by a set of satellite images taken from regions of the city. For example, the city of Campinas contains regions with bare soil, urban areas, silos and vegetation. Relation `Requirements` describes the needs to produce a given type of crop. In this particular case, we assume that bare soil, water, silos and urban areas are required. The result of dividing `CityRegions` by `Requirements` is relation `Cities`. It contains the list of cities considered appropriate for the crop, that is, those cities that have an image *similar* to each image in `Requirements`. In this particular case, only the city of `São Carlos` satisfies all requirements.

Many researchers have been proposing strategies to support similarity comparison in Relational Database Systems [2], commonly by extending Relational Operators. For example, recent works focus on the Join [5], Selection [4], Grouping [7] and Union [1]. However, to the best of our knowledge, no one focuses on the Division. Here, we tackle the problem by presenting the new **Similarity-aware Division ($\hat{\div}$)** operator. Our main contributions are:

1. **Operator Design and Usability** – we present the first division operator that is well suited to answer queries with an idea of "candidate elements and exigencies" to be performed on complex data from real applications.
2. **Formal Definition and Algorithm** – we formally



(a) CityRegions

(b) Requirements

(c) Cities

Figure 1: Example of the similarity-aware division used to spot cities suited to produce a crops by analyzing images.

defined the new operator, and carefully designed a fast and scalable algorithm for it;

3. **Experiments** – we validate our proposals by analyzing remote sensing images to support agriculture, following our motivational query with cities and crops. We also performed experiments to show that our algorithm is fast and scalable.

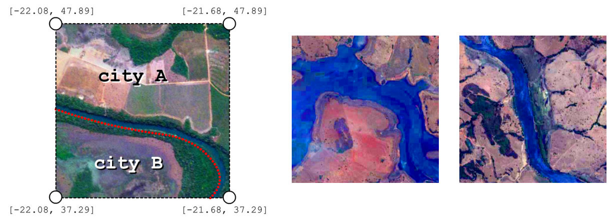
2. RELATIONAL DIVISION

The relational division is expressed by $T_1\ [L_1 \div L_2]\ T_2 = T_R$. Relations, T_1, T_2 and T_R refer to the dividend, the divisor and the quotient, respectively. L_1 and L_2 are lists of attributes from T_1 and T_2, in that order. Both lists must have the same number of attributes, and each attribute in L_1 must be union-compatible with its counterpart in L_2. The quotient relation T_R has all the attributes of T_1 except for those ones listed in L_1. That is, the schema of T_R is given by the relative complement $\overline{L_1}$ of L_1 with respect to $Sch(T_1)$, i.e., $Sch(T_R) = \overline{L_1} = Sch(T_1) - L_1$.

Remember that the quotient in the *arithmetic* operator of division for *integer numbers* is the largest integer that, multiplied by the divisor, defines a value smaller than or equal to the dividend, i.e., $\texttt{quotient} * \texttt{divisor} \leq \texttt{dividend}$. The remainder is the difference between the dividend and the result of multiplying the quotient by the divisor, i.e., $\texttt{dividend} - \texttt{quotient} * \texttt{divisor}$. The relational division is defined in a very similar manner: the quotient relation T_R is the subset of $\pi_{(\overline{L_1})}(T_1)$ with the largest possible cardinality, such that $T_R \times T_2 \subseteq T_1$. The remainder relation is given by $T_1 - T_R \times T_2$.

3. THE SIMILARITY-AWARE DIVISION

To include comparison by similarity in the Relational Division, we must cover all cases of identity comparisons that are performed intrinsically by the operator. The first case consists in altering the operator to combine the tuples of T_1 with *similar* values in the attributes of $\overline{L_1}$ to form a possible candidate that might be in the result set. The second case lies in relaxing the validation of the requirements by considering as satisfied those requirements in T_2 with values that are similar to their counterparts in T_1. For the first comparison case, consider again the example of selecting cities well-suited to produce a given crop using remote sensing images. One possible approach for this query is shown in Figure 2a. In this case, the images of regions must be grouped by similarity of their latitude and longitude coordinates in order to form candidate cities for evaluation and validation of the given crop requirements. However, a single region could be shared by two different cities. Thus, this tuple containing the image might be assigned to the group of city `A`, of city `B`, of both or, even, of none of them. The second comparison case is shown in Figure 2, where, given the image representing a crop production requirement, its satisfaction should be evaluated through comparison by similarity of the images of regions from the cities.



(a) Example of a region shared by two different cities.

(b) Image representing a region of the city.

(c) Image representing a crop requirement.

Figure 2: Example of comparisons that could be validated in the similarity-aware division shown Figure 1.

3.1 Formal Definition

This section defines formally the new operator involving the concept "for all" based on Relational Division, appending comparisons by similarity between attribute values — called as Division by similarity ($\hat{\div}$).

Definition 1. Two relations T_1 and T_2 are **Union compatible** if and only if they both have the same number of attributes and each attribute from T_1 has the same domain of its counterpart in T_2. We consider A_i to be the i^{th} attribute in the schema $Sch(T)$ of a relation T. The domain of A_i is $Dom(A_i)$. Any two relations T_1 and T_2 are Union compatible if and only if:

$$\begin{aligned} &(\,|Sch(T_1)| = |Sch(T_2)|\,) \land \\ &(\forall A_i \in Sch(T_1), \forall A_j \in Sch(T_2),\ i = j : \\ &\qquad Dom(A_i) = Dom(A_j)) \end{aligned} \tag{1}$$

Definition 2. The **similarity of attribute values** ($\hat{=}$) is represented as $a_1 \hat{=} a_2$, in which $a_1 \in A_1$ and $a_2 \in A_2$. Attributes A_1 and A_2 must follow the same metric space $M = \langle \mathbb{S}, d \rangle$ with $\mathbb{S}$ being the data domain and $d : \mathbb{S} \times \mathbb{S} \rightarrow \mathbb{R}^+$

the distance function, so that $\mathbb{S} = Dom(A_1) = Dom(A_2)$ and $a_1, a_2 \in \mathbb{S}$. For a threshold ξ, values a_1 and a_2 are similar if and only if:

$$a_1 \hat{=} a_2 \Leftrightarrow d(a_1, a_2) \leq \xi \tag{2}$$

Definition 3. The **similarity of tuples ($\triangleq$)** is represented as $t_1 \triangleq t_2$, in which t_1 and t_2 are tuples from relations T_1 and T_2, respectively. One tuple is similar to another if and only if their home relations are Union-compatible, and each attribute of the former has a value that is similar to its counterpart in the latter. We consider $t[A_i]$ to be the value of an attribute A_i for a tuple t. Formally, the similarity of tuples is defined by:

$$t_1 \triangleq t_2 \Leftrightarrow \forall A_i \in Sch(T_1), \forall A_j \in Sch(T_2), i = j : t_1[A_i] \hat{=} t_2[A_j] \tag{3}$$

Definition 4. The **set membership by similarity** ($\hat{\in}$) is represented as $t \hat{\in} T_1$, in which T_1 is a relation and $t \in T$ is a tuple. T_1 and T must be Union-compatible. Tuple t is an element of T_1 by similarity if and only if there exists at least one tuple $t_j \in T_1$ that is similar to t. Formally, we have:

$$t \hat{\in} T_1 \Leftrightarrow \exists\, t_j \in T_1 : t \triangleq t_j \tag{4}$$

Following the same idea, t is not an element of T_1 by similarity if and only if there is no tuple $t_j \in T_1$ that is similar to t. Formally, it is given by:

$$t \hat{\notin} T_1 \Leftrightarrow \nexists\, t_j \in T_1 : t \triangleq t_j \tag{5}$$

Definition 5. The **Subset by similarity** ($\hat{\subseteq}$) is represented as $T_1 \hat{\subseteq} T_2$, in which T_1 and T_2 are Union-compatible relations. Relation T_1 is a subset of T_2 by similarity if and only if every tuple $t_i \in T_1$ is also an element of T_2 by similarity. Formally, its is defined by:

$$T_1 \hat{\subseteq} T_2 \Leftrightarrow \forall\, t_i \in \mathsf{T}_1 : t_i \hat{\in} \mathsf{T}_2 \tag{6}$$

Definition 6. The **Difference by similarity** ($\hat{-}$) is a binary operation represented as $T_1 \hat{-} T_2 = T_R$, in which T_1 and T_2 are Union-compatible relations. The resulting relation T_R has all tuples of T_1 that are not members of T_2, by similarity. Formally, we have:

$$T_R = \{t_i : t_i \in T_1 \ \wedge\ t_i \hat{\notin} T_2\} \tag{7}$$

Definition 7. A **Group of similars** T_{G_k} is a subset of a given relation T_1, such that each of its tuples is similar to at least one other tuple in the group, taking into account only a subset of attributes $L \subseteq Sch(T_1)$. Relation T_{G_k} is also considered to be a group of similars if $|T_{G_k}| = 1$. Formally, T_{G_k} is a group of similars if and only if:

$$(T_{G_k} \subseteq T_1) \wedge ((\forall\, t_i \in T_{G_k} : (\exists\, t_j \in T_{G_k}, i \neq j : t_i[L] \triangleq t_j[L])) \vee (|T_{G_k}| = 1)) \tag{8}$$

Definition 8. One **Similarity grouping** T_G is the set of all groups of similars extracted from a relation T_1, taking into account a subset of attributes $L \subseteq Sch(T_1)$. The number of groups is $\kappa = |T_G|$. Formally, we have:

$$T_G = \{T_{G_i} : T_{G_i} \text{ is a group of similars from } \mathsf{T}_1 \text{ regarding } L\} \tag{9}$$

The restriction as follows applies:

$$\mathsf{T}_1 = \bigcup_{k=1}^{\kappa} T_{G_k} \tag{10}$$

Definition 9. The **Similarity-aware division** ($\hat{\div}$) is a binary operation represented as $\mathsf{T}_1\ [L_1 \hat{\div} L_2]\ \mathsf{T}_2 = \mathsf{T}_R$, in which T_1, T_2 and T_R are relations that respectively correspond to the dividend, the divisor and the quotient. $L_1 \subseteq Sch(\mathsf{T}_1)$ and $L_2 \subseteq Sch(\mathsf{T}_2)$ are lists of attributes, so that relations $\pi_{(L_1)}\mathsf{T}_1$ and $\pi_{(L_2)}\mathsf{T}_2$ are Union-compatible. The schema of T_R is defined as $Sch(\mathsf{T}_R) = \overline{L_1} = Sch(\mathsf{T}_1) - L_1$. The instance of T_R is the union of $\pi_{(\overline{L_1})} T_{G_k}$ for the largest possible number of groups of similars $T_{G_k} \in T_G$, such that $\mathsf{T}_R \times \mathsf{T}_2 \hat{\subseteq} \mathsf{T}_1$. Formally, the quotient T_R is defined as:

$$\mathsf{T}_R = \bigcup_{k=1}^{\kappa} \begin{cases} \pi_{(\overline{L_1})} T_{G_k}, & if\ \forall t_j \in \pi_{(L_2)}(T_2) : \\ & (\exists\, t_i \in \pi_{(L_1)}(T_{G_k}) : t_i \triangleq t_j) \\ \varnothing, & \text{otherwise.} \end{cases} \tag{11}$$

The remainder of the similarity-aware division is given by:

$$\mathsf{T}_1 \hat{-} \mathsf{T}_R \times \mathsf{T}_2 \tag{12}$$

3.2 Algorithm

This section presents a novel algorithm that we carefully developed for the similarity-aware division. It has five input parameters: T_1, L_1, T_2, L_2 and T_G. Parameters T_1 and T_2 are the dividend and the divisor. L_1 and L_2 respectively identify the attributes of relations T_1 and T_2 to be compared with each other, thus defining how to validate the candidate groups with regard to the requirements. For each pair of attributes for comparison, it must be informed the metric to be used to measure similarity and the similarity threshold ξ. Finally, parameter T_G identifies the group T_{G_k} (or the groups) that each tuple of relation T_1 belongs to.

```
SimilarityDivision($\mathsf{T}_1$, $L_1$, $\mathsf{T}_2$, $L_2$, $T_G$);
Result: IDs of the valid groups.
begin
    // all groups are valid at the begining
    $Gv_{id} = \{1,\ 2, \ldots\ |T_G|\}$;
    foreach tuple $t_j \in \pi_{(L_2)}\mathsf{T}_2$ do
        $\mathsf{T} = IndexRangeQuery(\mathsf{T}_1, L_1, t_j)$;
        $Gq_{id} = \varnothing$;
        foreach tuple $t_i \in \mathsf{T}$ do
            $Gq_{id} = Gq_{id}\ \cup\ t_i.groupIDs$;
        end
        $Gv_{id} = Gv_{id}\ \cap\ Gq_{id}$;
    end
    return $Gv_{id}$;
end
```

Algorithm 1: Similarity-aware division.

Algorithm 1 is the pseudocode of the algorithm that we propose. We assume that relation T_1 has indexes known as Metric Access Methods (MAM) ready to be used for the attributes in L_1. The algorithm works by iteratively updating a set Gv_{id} with valid groups' identifiers. All candidate groups are considered to be valid at the beginning, starting with $Gv_{id} = \{1,\ 2, \ldots\ |T_G|\}$. Then, for each requirement t_j from T_2, we perform a range query in T_1 using the requirement itself as the query center and taking advantage of the existing indexes to speed-up the execution. The query finds every tuple $t_i \in \mathsf{T}_1$ such that $t_i[L_1] \hat{=} t_j$. The next step is to build a set Gq_{id} with all identifiers of the groups that satisfy requirement t_j, that is, the groups of the tuples returned by the query. Then, set Gv_{id} is updated with the intersection of the valid groups from the previous iteration and the groups that meet the current requirement. At the end of the execution, only identifiers of candidate groups that meet all the requirements remain in Gv_{id}.

Time complexity: Algorithm 1 performs $|\mathsf{T}_2|$ range queries in relation T_1. State of the art MAM indexes allow us to perform each range query in $O(log\ |\mathsf{T}_1|)$ time. Therefore, the total runtime of Algorithm 1 is $O(|\mathsf{T}_2|\ log\ |\mathsf{T}_1|)$.

4. EXPERIMENTS

We performed a case study with remote sensing images to allow a semi-automatic identification of cities well-suited to produce particular types of crops. The scheme adopted to perform this study was the same of our example from Figure 1. The remote sensing images come from collaborators of the Centre of Meteorological and Climate Research Applied to Agriculture (CEPAGRI), Brazil, and the Brazilian Agricultural Research Corporation (EMBRAPA). The remote sensing images were already grouped by city limits. The pre-processing started with the segmentation of each original city image, and then we divided it into rectangular region tiles. The dataset contains imagery from five Brazilian cities, i.e., Sapezal-MT, Sorriso-MT, Anhanguera-GO, Catolândia-BA and Volta Redonda-RJ.

For the image comparison between the crop requirements and the region tiles, we use the Earth mover's distance with a similarity threshold of 0.05. The requirements were set as in Figure 3, where four needs for a crop production were given, i.e., (a) silo infrastructure, (b) bare soils, (c) water and (d) urban area to support the production. The recognition of these patterns, was performed through the segmentation of the remote sensing images into six features, i.e., urban area, water, dense vegetation, sparse vegetation, bare soil and silo. After runing the query, only one city had successfully satisfied all these requirements, the city of Sorriso-MT. The other cities were able to satisfy some of the needs, but just this city accomplished all the four requirements. The result of the division algorithm can be validated visually through Figure 4, where we illustrate the city of Sorriso-MT with all requirements and one of the other cities that miss at least one requirement. We also report that our algorithm scaled linearly or even sub-linearly in experiments performed with synthetic data, varying the sizes of the input and output relations up to millions of tuples. Space limitations prevent us from detailing these results.

5. CONCLUSION

In this paper we identified severe limitations on the usability of the Relational Division to process complex data, and tackled the problem by extending it into the new Similarity-aware Division ($\hat{\div}$) operator. We formally defined the new operator and designed a fast and scalable algorithm for it. To validate our proposals, we performed a case study on the support of agriculture. Provided that our algorithm is fast and scalable, we argue that the new similarity-aware division is potentially useful to analyze very large amounts of complex data, even in real-time. For example, it is potentially useful to support agriculture, as presented in this paper; to support genetic analyses, selecting animals that satisfies all the genetic conditions required, and even to help hiring personnel and identifying new clients in enterprises.

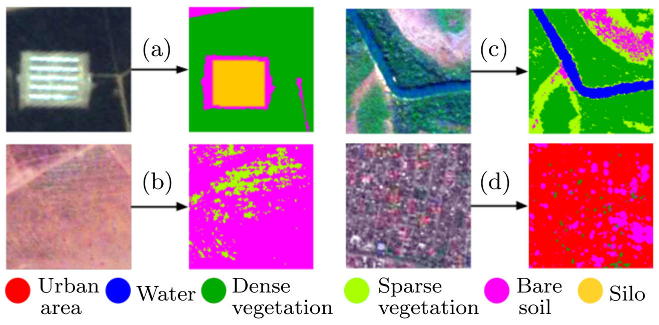
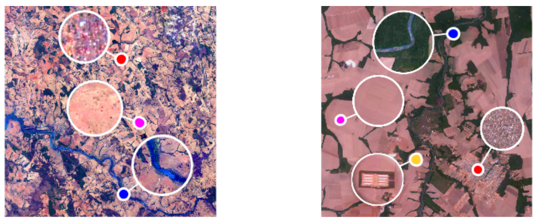


Figure 3: The images selected for the divisor in the case study, representing the requirements for a crop production.



(a) City of Anhanguera-MT missing the silo requirement. (b) City of Sorriso-MT with all the requirements.

Figure 4: Example of the result over two cities.

6. REFERENCES

[1] W. J. Al Marri, Q. Malluhi, M. Ouzzani, M. Tang, and W. G. Aref. The similarity-aware relational database set operators. *Inf. Syst.*, 59(C):79–93, July 2016.

[2] P. Budíková, M. Batko, and P. Zezula. Query language for complex similarity queries. In *ADBIS*, 85–98, 2012.

[3] E. F. Codd. *The Relational Model for Database Management: Version 2*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.

[4] Y. Silva, W. Aref, P.-A. Larson, S. Pearson, and M. Ali. Similarity queries: their conceptual evaluation, transformations, and processing. *The VLDB Journal*, 22(3):395–420, 2013.

[5] Y. Silva, S. Pearson, J. Chon, and R. Roberts. Similarity joins: Their implementation and interactions with other database operators. *Inf. Syst.*, 149-162, 2015.

[6] Y.Silva, A. Aly, W. G. Aref, and P.-A. Larson. Simdb: a similarity-aware database system. In *SIGMOD Conference*, 1243–1246, 2010.

[7] M. Tang, R. Y. Tahboub, W. G. Aref, M. J. Atallah, Q. M. Malluhi, M. Ouzzani, and Y. N. Silva. Similarity group-by operators for multi-dimensional relational data. *IEEE TKDE*, 28(2):510–523, 2016.

[8] P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search - The Metric Space Approach*, volume 32 of *Advances in DBMS*. Kluwer, 2006.