# Handling Commutative Operators in Visual Math Search

Parag Mali

**Abstract** This report presents a technique for retrieval of mathematical expressions using visual information while handling the commutative properties of the operators. It uses general Line of Sight (LOS) graph representation. Edges in this graph define symbol pair tuples which are used as the entries for the inverted index. The symbol identities are given by the probability distribution from OCR output. Retrieval is a three-step process. In the first step, all the query edges formed in the LOS graph are looked up in the inverted index. In the second step, all the candidates are aligned with the query while handling commutative operators and in the third step, matches are scored based on the symbol identities and relative positions. We discuss qualitative results as well as quantitative results obtained by testing on the NTCIR-12 Wikipedia Formula Browsing Task.

## 1 Introduction

Text-search engines with their current form are not able to perform well in the math search. These engines can not effectively index the structure of the math expressions using indexing strategies used for text. So, most of the modern math search engines use symbolic formula retrieval model. They use either LaTeX expressions or MathML for formula search. These search engines use different techniques like span logic, semi-operator trees, lexicographical ordering of operands to handle commutativity of operators. On the other hand, there is no work published yet in image-based formula retrieval which can handle commutative operators.

In this report, we extend the Tangent V [3] image based formula retrieval model to handle commutative operators. Tangent V uses general Line of Sight graphs over symbols detected using OCR to represent the symbol pairs. We use the exact same indexing strategy as used in Tangent V but modify the retrieval technique to allow symmetric matches for the commutative operators. Proposed retrieval technique handles many cases of matching expressions where symbols are permuted. This technique does not use any symbol layout trees, operator trees or grammars for finding the required structure. The bpref and precision values are lower than the Tangent V model when tested on the NTCIR-12 Wikipedia Formula Browsing Task benchmark. One of the possibilities of the lower score can be Tangent VC does not make use of spatial distortion cost used by Tangent V which helps in avoiding a match which is inconsistent with the query layout. For Tangent VC layout of the candidate, matches can be inconsistent with queries to allow commutative matches. It is required to modify spatial distortion cost before using it with Tangent VC.

## 2 Related Work

### 2.1 Mathematical Information Retrieval

Mathematical Information Retrieval (MIR) is relatively new field [27]. MIR has received increasing attention in last decade because of the widespread availability of optical character recognition and text search engines. Four important problems of pattern recognition- classification, segmentation, parsing and machine learning- all come into play while designing MIR systems. For example, while recognizing handwritten math expression, it is required to perform symbol segmentation and classification. Parsing step defines the relationships

between these symbols. For classification, segmentation and parsing machine learning is used to predict the classes.

Each MIR system should try to achieve eight basic objectives described by Youssef [26]: math awareness, a natural math query language, fine granularity of searchable and retrievable information units, perfect recall, perfect precision, perfect relevance ranking, useful highlighting, and minimum hit redundancy. Youssef envisions that beyond the conventional search for documents, math search can fulfill higher-level roles like discovering the similarity between fields, computer-aided proving, education, and research. MIR engines can be great tools for researchers who wish to find technical papers which define a particular function or for a student who wants to search textbooks or lecture notes for a particular expression.

Text-based search engines can be used for MIR, but the results will be weak. These search engines do not have a model for the mathematical structure and the meaning of math expressions also lies in their structure. Also, it is hard to encode them as a string for matching with other expressions. Furthermore, there is equivalence relationship between multiple math expressions and it is not possible to store all of them in the thesaurus the way it is done for the traditional text search engines. It is observed in the study by Zhao et al. [32] that as the MIR research advances, users will continue to use a combination of general-purpose search engines along with specialized MIR systems for their mathematical information needs. They observed that expert users are satisfied with the state of the art on screen equation editors, but general users find the keyword search most effective and did not feel inputting equation was easy.

In the following sections symbolic and image-based formula retrieval are described.

### 2.1.1 Symbolic Formula Retrieval

The symbolic formula retrieval refers to the retrieval of formula when symbols and structure of formula are known as in the case of LaTeX expressions or MathML. Content MathML provides enough metadata to create an operator tree and hence it is considered as a semantic representation. Presentation MathML and LaTeX provides only appearance information. The semantic meaning of math expressions is preserved in semantic representations whereas appearance-based representations (for example, Presentation MathML) do not provide such information which makes them ambiguous and correct interpretation of math expressions becomes context dependent. That makes semantic representations theoretically better for MIR but most published work in this area uses appearance based representations. It is because not all expressions can be represented by the semantic representations. Few such examples where Content MathML can not be used are provided in [11]. Figure 1 shows different representations of math expressions.

Approaches to formula retrieval can be classified based on the primitives used for the indexing as tree-based, text-based and spectral [30].

1. **Text based**: Mathematical content is represented by a sequence of tokens obtained by flattening it. See Figure 1c for example of flattened string. Some methods use canonicalization to simplify expressions and to identify commutative operators and equivalences [14, 22]. It is also common to enumerate identifiers to support generalized variable matching and/or unification [6, 22, 25]. A major advantage of using text-based approaches is, it allows the use of existing optimizations in text search engines, such as ranking by TF-IDF [17], topic modeling, and word embedding [25]. Common text search engines used by MIR systems include Apache-Lucene[1], Solr[2] and ElasticSearch.[3]

2. **Tree based**: These approaches index formulas as complete SLTs or OPTs. See Figure 1h and 1i for examples of SLT and OPT. The common approach is to apply tokenization and use multiple text tokens to represent each tree. The hierarchical structures in formulae are mapped directly and organized within tree-based indexing structures [8, 10, 33]. In these approaches, all subexpressions in formulae are indexed to support partial matching. Common subexpressions are labeled and shared to reduce index sizes [10].

3. **Spectral**: Features extracted from trees are used as retrieval primitives. Simpler primitives allow more partial matches, increasing recall. Path-based methods store sets of paths from the root to internal

[1] http://lucene.apache.org/
[2] http://lucene.apache.org/solr/
[3] https://www.elastic.co/

nodes [7] and leaves [33]. Paths may reflect operator commutativity by inserting symbols [33] or using unordered paths [13]. Some use hashing to encode subtrees [13, 21]. Tangent formula retrieval model [24] use symbol pairs retrieved from SLT. In Tangent 3 SLT symbol pairs along with their relative paths are used to index math expressions [5, 30]. Tangent S uses both SLT and OPT at the same time for formula retrieval. They distinguish commutative operators and non-commutative operators and ignore the order of children for commutative operators. All children of commutative operators in the operator tree have same label [3].

### 2.1.2 Image Based Formula Retrieval

Image-based formula retrieval deals with the retrieval of mathematical formula directly from images when symbols and structure of formula are not known. As compared to symbolic formula retrieval there are few very methods proposed in this area. These methods are very different from each other. Methods described in this section avoid recognition because the recognition rates of handwritten mathematical formula are still very low [20].

In their work [15], Marinai et al. proposed an approach which exploits shape contexts (SC) encoded with a bag of visual words method to describe the shape of the mathematical symbol. This method uses topologically ordered clusters computed by means of Self Organizing Maps (SOMs) and then occurrences of features in the clusters are accumulated in a vector of visual words. In a more recent work [16], they use Earth Mover's Distance (EMD) as a similarity measure in the mathematical symbol retrieval task. This method is designed for isolated math symbols, not complete formula.

Zanibbi and Yu have proposed a method for locating mathematical expressions in document images without the use of optical character recognition in [28]. In this method, they produce recursive X-Y cuts for each page in the corpus. The images of handwritten expressions are used as queries. During retrieval, the query is looked up in the document region index using features of its X-Y tree, producing a set of candidate regions. Dynamic Time Warping technique is used to rank candidate regions based on pixel projection profiles of upper/lower image halves of both query and page region image. Zanibbi and Yuan [29] segment expressions into connected components. Components in the query are matched using contour and density features, aspect ratios, and relative positions with components in each expression in the collection.

Chatbri et al. described a method based on connected component matching [2]. They extract connected components from the query and document. Each component is described using Contours Points Distribution Histogram (CPDH) shape descriptor. 'K' voting groups which match with query image are extracted from the document image. After filtering voting groups based on the scale consistency and global matching of the group with query image, remaining voting groups show relevant occurrences of query image in the document.

The Tangent V system described in [1] uses a sketch based image retrieval method. The Line-of-Sight graphs are used to create a rough representation of the structure of a document. Math symbols in each document are classified using a probabilistic math symbol classifier. Each expression is indexed using multiple combinations of potential labels for pairs of identified symbols from the graphs. This method allows retrieving expressions even if the real classes for some symbols are not matched with very high confidence as long as there are enough symbols that can be matched between the query and a given candidate document. Tangent V has achieved four out of eight math search objectives described in section 2.1: math awareness, a natural math query language, fine granularity of searchable and retrievable information units and useful highlighting. One of the important steps towards accomplishing other math search objectives is the ability to handle properties of mathematical operators. For example, to get better precision and recall, handling commutativity and associativity of mathematical operators is required. Tangent VC system proposed in this work is an extension of Tangent V and is capable of handling commutative operators for simpler math expressions. Next section describes methods used by current MIR systems for handling commutativity in formula retrieval.

## 2.2 Commutativity in Formula Retrieval

Commutativity is a very common property of mathematical operators. Hence, flexibility in the order of operands should be provided by any math search system. A binary operation $*$ on a set S is called commutative if $x * y = y * x \ \forall x, y \in S$.

$x^2 + y^2$

(a) Expression Image

```
x^2 + y^2
```

(b) Latex

add-pow-x-2-pow-y-2

(c) Flattened

add-pow-x-2-pow-y-2
pow-x-2
pow-y-2
x
2
y
2

(d) Tokens

add-pow-id1-const1-pow-id2-const2
pow-id1-const1
pow-id2-const2
id1
const1
id2
const2

(e) Generalized Tokens

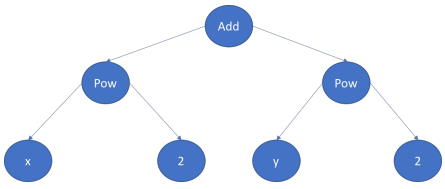
```
<Math>
    <m:math>
       <m:mrow>
          <m:msup>
             <m:mi>x</m:mi>
             <m:mn>2</m:mn>
          </m:msup>
          <m:mo>+</m:mo>
          <m:msup>
             <m:mi>y</m:mi>
             <m:mn>2</m:mn>
          </m:msup>
       </m:mrow>
    </m:math>
</Math>
```

(f) Presentation MathML

```
<Math>
    <m:math>
       <m:apply>
          <m:plus/>
          <m:apply>
             <m:csymbol>superscript</m:csymbol>
             <m:ci>x</m:ci>
             <m:cn type="integer">2</m:cn>
          </m:apply>
          <m:apply>
             <m:csymbol>superscript</m:csymbol>
             <m:ci>y</m:ci>
             <m:cn type="integer">2</m:cn>
          </m:apply>
       </m:apply>
    </m:math>
</Math>
```

(g) Content MathML


(h) Symbol Layout Tree (SLT)




(i) Operator Tree (OPT)

Figure 1: Different representations of math expressions. (a) is the original expression. (b) and (c) are string representations. (d) and (e) are sample token based representations. (f) and (g) are xml based representations. (h) and (i) are tree based representations. This figure is based on the Figure 2.2 in [1]

In their work [18], Miner et al. use span logic to provide commutativity and associativity. Span logic restricts the search space to only those documents where sub-query objects appear within a given proximity of one another defined by a span scope. For example, a span query "numerator: (x, y){4}" would match those documents where x and y appear in the numerator field within a distance of 4. So, this query will match x + z + y. The span scope can be calculated based on the length of the query and weights of each element in the query. The span logic can miss few matches for which the operands are not within span scope.

Many systems use ordering algorithms to deal with commutativity and associativity in math search. EgoMath search engine indexes different representations of same mathematical formula [19]. Also, it uses an ordering algorithm which orders symbols such that it gives the same order for all expressions which have same operands and operators but in a permuted order. DLMF [17] use sorted parse tree where children of any node are sorted from left to right whenever changing the order of children does not alter the mathematical meaning. Math Indexer and Searcher [23] uses ordering based on nodes of Presentation MathML. WikiMirs [9] system generates a presentation tree and uses a normalizer for variables and constants. The normalizer unifies different formulae with the same meaning so as to ensure the high recall of relevant formulae. Variables, constants and the order of operands are all taken into account in the process.

Lin et al. [14] use semi-operator trees. In this system, if a node of a semi-operator tree is a commutative operator then children are ordered by lexicographical order. The variables and constants are tagged and their exact values are not considered while matching. MathWebSearch [12] mark commutative operators in the query and ignore the order of operands while matching. To handle commutative operators, [33] use operator tree transformations such that if operator tree node has a parent which is also commutative then that node will pass its children to parent and delete itself.

All approaches discussed in this section come under symbolic formula retrieval. To our knowledge, there is no work done yet to handle commutativity in the image based formula retrieval. None of the systems described in the section 2.1.2 handle commutative operators. In the next section, we describe how Tangent V can be extended to handle commutative operators.

# 3 Methodology

Tangent VC is an extension of Tangent V system described in [4]. Tangent V does not rely on the full recognition of the mathematical expression rather it uses Line of Sight (LOS) graphs for construction of a rough representation of the structure of the expression. The difference between Tangent V and Tangent VC is that later allows commutative operator matching to some extent. This model cannot handle all the cases of commutative operators but handle simple cases. Similar to Tangent V, Tangent VC model uses a three-layer procedure for retrieval of math notations from images. In the first layer, it selects candidates based on the symbol pair matches. In the second layer, detailed matching procedure merges initial candidate matches to form larger unique matches. In the final layer, matches are scored on based on symbol identities and relative positions. Next sections describe indexing and retrieval in detail.

## 3.1 Indexing

Indexing step is exactly same as Tangent V as described in [4]. For indexing binary images, recognition based methods are avoided and more general line of sight graph-based method is adapted. To create a line of sight graph, convex-hull of the contours of each connected component is found and then a directed line of sight between the center of the observer and each point in the convex-hull of each of the remaining connected components are defined. After that, prune edges between symbols that are more than twice the median distance apart. This pruning reduces both the index size and retrieval times.

Each symbol in a binary image is considered as a connected component. Probabilistic math symbol classifier is used to recognize math symbols in the given image. This classifier predicts confidences of top classes for each symbol which is represented as a node in the LOS graph. The symbols connected by an edge in the LOS form a symbol pair. For each symbol pair $(u, v)$ in the LOS graph, a 3D unit vector $\langle dx, dy, dz \rangle$ is defined. $\langle dx, dy \rangle$ is nothing but the difference between the centers of the bounding boxes of two symbols $u$ and $v$. The third dimension is useful when one node partially or completely contains the other node. $dz$ defines the closeness of node centers. For a given node $u$, with width $w_u$ and height $h_u$, a sphere radius $r_u$ is defined as,

$$r_u = \frac{min(w_u, h_u)}{2} \tag{1}$$

For nodes u and v, $d_z$ is calculated as,

$$d_z = \begin{cases} 0, & \text{if } |\langle d_x, d_y \rangle| \geq r_s \\ \sqrt{r_s^2 - |\langle d_x, d_y \rangle|^2}, & \text{otherwise} \end{cases} \tag{2}$$

Final 3D vector is obtained by dividing $\langle d_x, d_y, d_z \rangle$ by its norm. Once all the spatial relationships between the nodes are defined in the next step index pairs based on top classes for each symbol pair is generated. For a given node $u$, minimum set of top $k$ class labels $L_u = \{l_u^1, l_u^2, \cdots, l_u^k\}$ with their corresponding confidences $C_u = \{c_u^1, c_u^2, \cdots, c_u^k\}$ is formed such that,

$$\sum_{i=1}^{k} c_u^i \geq c^{min} \vee (k = l^{max}) \tag{3}$$

We have set $c^{min} = 0.8$ and $l^{max} = 3$. The final set of symbol pairs are given by,

$$SortedPair(l_u^a, l_v^b) = \begin{cases} (ID(u,v), l_u^a, c_u^a, l_u^b, c_u^b, \langle dx, dy, dz \rangle, 1), \text{if } l_u^a \leq l_v^b \\ (ID(u,v), l_u^a, c_u^a, l_u^b, c_u^b, \langle -dx, -dy, dz \rangle, -1), \text{otherwise} \end{cases} \tag{4}$$

$$Pairs(u,v) = \{SortedPair(l_u^a, l_v^b) | (l_u^a, l_v^b) \in L_u \times L_v\} \tag{5}$$

The final inverted index has entries with $l_u, l_v$ of given raw pairs and corresponding posting list. Entry in a posting list has following structure,

$$(pair_{ID}, c_u, c_v, \langle dx, dy, dz \rangle, dir) \tag{6}$$

where, $pair_{ID}$ is a unique global identifier generated for nodes $u$ and $v$, $c_u$ and $c_v$ are the confidences for labels $l_u$ and $l_v$ respectively, and the value $dir \in \{1, -1\}$ indicates if the unit vector was inverted from its original values. For example, for expression $x^2$, the 3D unit vector will point from the center of $x$ towards the center of 2. This is due to 2 being considered a smaller label than x in their lexicographic order. Hence, the dir value for this vector will be -1.

## 3.2 Retrieval

The retrieval procedure in Tangent VC considers commutativity of operators while matching. For that, it introduces the concept of node type. For a given node $u$, minimum set of top $k$ class labels $L_u = \{l_u^1, l_u^2, \cdots, l_u^k\}$ with their corresponding confidences $C_u = \{c_u^1, c_u^2, \cdots, c_u^k\}$ is known. Different node types are described below.

- **Commutative Operator (C)**: If first three highest confident labels contain any of the commutative operators, then the current node is considered as a commutative operator. Formally, if $S_c$ is a set of commutative operators, then node type is 'C' if $S_c \cap L_u \neq \emptyset$
- **Non-Commutative Operator (N)**: If first three highest confident labels contain any of the non-commutative operators, then the current node is considered as a non-commutative operator. Formally, if $S_n$ is a set of non-commutative operators, then node type is 'N' if $S_n \cap L_u \neq \emptyset$
- **Other (O)**: All other nodes are considered to be of the other type.

Node type information is useful while matching symbol pairs. A node of a particular type can be matched to another node of that type. For example, it allows us to match commutative operators to commutative operators and non-commutative operators to non-commutative operators. Next, the three-layer retrieval process is described.

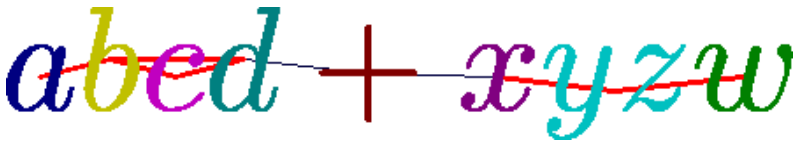
**Layer 1: Lookup** The query symbol pair labels are used to find all corresponding posting entries from the inverted index. The cosine similarity between each posting entries displacement unit vector $D^i$ and query's unit displacement vector $D^q$ is calculated by performing dot product of these vectors. There are constraints on how much angle difference between these two vectors is tolerated. Currently, $-45^\circ \leq \theta \leq 45^\circ$ and $135^\circ \leq \theta \leq 225^\circ$ for symbols pairs which have one node with of type commutative operator and $-45^\circ \leq \theta \leq 45^\circ$ for others. Allowing $\theta$ to be in this range allows symmetric matches required for the commutative operators.

The final score and direction of matching a index pair $T_i = (pair^i_{ID}, c^i_u, c^i_v, \langle dx^i, dy^i, dz^i \rangle, dir^i)$ with query pair $T_q = (pair^q_{ID}, c^q_u, c^q_v, \langle dx^q, dy^q, dz^q \rangle, dir^q)$ is given by,

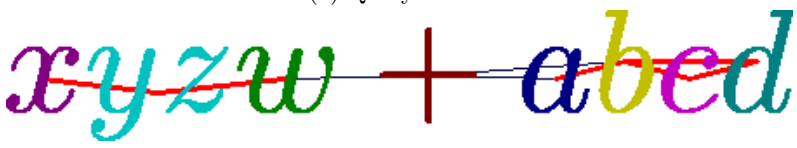
$$\begin{aligned} Score(T^i, T^q) &= (c^i_u c^i_q)(c^i_v c^q_v)(D^i \cdot D^q) \\ Direction(T^i, T^q) &= (dir^i_u dir^i_q) \end{aligned} \tag{7}$$

We might find multiple matches between unique index pairs and unique query pairs based on all the consistent label assignments between their aligned nodes. Therefore, we need to aggregate these matches by unique combinations of $(pair^i_{ID}, pair^q_{ID})$. The aggregation process computes the sum of the scores and directions for all the initial pairs. After that for each aggregated pair, aligned matches will be generated. An aligned match is a triplet $m_x = (G^q_x, G^i_x, f_x)$, where $G^q_x$ is a subset of nodes from the graph $G^q$ ($G^q_x \subseteq G^q$) , $G^i_x$ is a subset of nodes from a binary image graph $G^i$ from the index ($G^i_x \subseteq G^i$), and f is a graph isomorphism mapping nodes from $G^q_x$ to $G^i_x$. For a given query pair defining the subset $G^q_x = \{u^q, v^q\}$ and a matching index pair defining subset $G^i_x = \{u^i, v^i\}$, a positive value indicated that the mapping should be $f(u^q) = u^i \wedge f(v^q) = v^i$, while a negative direction value indicates that the mapping should be $f(u^q) = v^i \wedge f(v^q) = u^i$.

**Layer 2: Grow matches** In this step, initial pairwise aligned matches are greedily merged together to form larger matches. Two matches are merged together if there is a common node aligned in both the matches. At the end of this step, we will get the subexpressions. Once we know the subexpressions, with what operators these subexpressions are connected in the query and candidate. All those expressions which are connected with the same operator in the query, as well as the candidate, are merged. Then we find the operator in the graph with which these subexpressions are connected. It is possible that there are more than one operators of the same type. For example, + can occur more than once in the query and subexpression. In the next step, disjoint subexpression matches are merged. How operators are merged is explained in detail with an example.



(a) Query Match



(b) Candidate Match

Figure 2: This figure shows how query $abcd + xyzw$ is matched with candidate $xyzw + abcd$ using proposed method. The red edges are the matched Line of Sight graph edges. Symbols which are aligned to each other are in the same color.

See figure 2 for how query $abcd + xyzw$ is matched with candidate $xyzw + abcd$ .
The symbol pairs $S_q$ and $S_c$ for query $abcd + xyzw$ and candidate $xyzw + abcd$ are given below.

$$S_q = \{(a,b),(b,c),(b,d),(c,d),(+,d),(+,x),(x,y),(y,z),(w,z)\}$$
$$S_c = \{(a,b),(b,c),(b,d),(c,d),(+,a),(+,w),(x,y),(y,z),(w,z)\}$$

Note that there is no symbol pair with operator common in $S_q$ and $S_c$. So, operator node will be merged with the match in next step and it will not be connected with any subexpression. If there was a symbol pair with operator common between $S_q$ and $S_c$, then it would have been added in this step only and it will be connected to the subexpression. For example, in a case where query and candidate both are $abcd + xyzw$ then there will be a common symbol pair with the operator and hence it will be matched in this step. This type of matching allows us to rank $abcd + xyzw$ higher than $xyzw + abcd$ because edges with subexpression are not matched in the latter case. Let initial symbol pair matches be denoted by M,

$$M = S_q \cap S_c = \{(a,b),(b,c),(b,d),(c,d),(x,y),(y,z),(w,z)\}$$

Now merge all those symbol pairs which have at lease one common node. Let these subexpressions extracted from M be denoted by C,

$$C = \{(a,b,c,d),(x,y,z,w)\}$$

We find how each subexpression is connected in the given expression. Let $C_q$ define this connection for query and $C_c$ define this connection for candidate,

$$C_q = \{((a,b,c,d),(+,+_i)),((x,y,z,w),(+,+_i))\}$$
$$C_c = \{((a,b,c,d),(+,+_k)),((x,y,z,w),(+,+_k))\}$$

The meaning of entry $((a,b,c,d),(+,+_i))$ is that $(a,b,c,d)$ is connected with + operator which has node id $+_i$ in the LOS graph.

As both subexpressions are connected with the same operators then these subexpressions can be merged to form a larger match $C^*$,

$$C^* = \{(a,b,c,d,x,y,z,w,+)\}$$

This shows that entire query is matched with the candidate.

Let us consider another example. Refer figure 3c and 3d for query and candidate matches. The horizontal line of a fraction sign is represented by '/'. $S_q$ and $S_c$ represents symbol pairs in query and candidate respectively.

$$\begin{aligned}S_q = \{&(1,/),(2,/),(1,x),(2,x),(2,y),(/,1),(p,1),(p,4),(y,-),\\ &(2,-),(y,+),(2,+),(1,-),(4,-),(1,+),(2,+),(p,+),(/,+),(/,-),(x,+)\}\\ S_c = \{&(1,/),(2,/),(1,x),(2,x),(2,y),(/,1),(p,1),(p,4),(y,-),\\ &(2,-),(1,-),(4,-),(4,+),(1,+),(2,+),(1,+),(p,+),(/,+),(/,-),(x,+)\}\end{aligned}$$

$C$ represents matched subexpressions,

$$C = \{(y,2),(1,/,4,p),(1,2,x)\}$$

$C_q$ and $C_c$ represents how subexpressions in $C$ are connected in the query and candidate expression.

$$C_q = ((y,2),(+,+_i)),((1,/,4,p),(-,-_i)),((1,2,x),(+_i))$$
$$C_c = ((y,2),(+,+_i)),((1,/,4,p),(-,-_i)),((1,2,x),(+_i))$$

And, finally we get $C^*$ after merging subexpressions,

$$C* = \{(y,2,1,/,4,p,1,2,x)\}$$

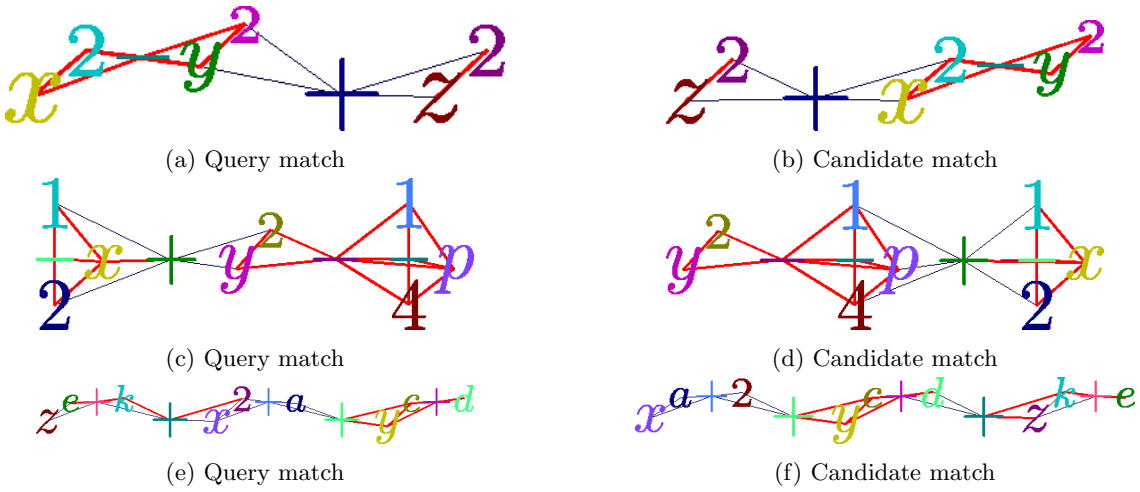
**Layer 3: Ranking** The scoring function MLS(M) used in Tangent VC is Maximum Line of Sight similarity [4]. Let a matched symbol pair be represented by $(Q, C)$ where $Q = ((\Omega_{q_1}, q_1), (\Omega_{q_2}, q_2))$ and $C = ((\Omega_{c_1}, c_1), (\Omega_{c_2}, c_2))$. $p(\omega|k)$ represents the conditional probability of symbol class $\omega$ given visual features for symbol $k$. $\Omega_{q_1}$ is a set of possible symbol identities for $q_1$. Let $\vec{q}$ and $\vec{c}$ represent the query and candidate displacement vectors. It is possible that there is no edge between subexpression and operator as explained in layer 2 and if present edges to commutative operators are treated exactly the same as other edges. If $M_s$ contains corresponding query and candidate symbol pairs in match M, and $|N_Q|$ and $|E_Q|$ represents numbers of nodes and edges in the line of sight graph, then MLS is given by,

$$
\begin{aligned}
S_{\angle} &= \vec{q} \cdot \vec{c} \\
R(M_s) &= \frac{1}{|N_Q|} \sum_{(q,c) \in M_s} \sum_{\omega \in \Omega_q \cap \Omega_c} p(\omega|q) p(\omega|c) \\
R_{\angle} &= \frac{1}{|E_Q|} \sum_{(Q,C) \in M} S_{\angle}(Q, C) \\
MLS(M) &= 2 \frac{R(M_s) \cdot R_{\angle}(M)}{R(M_s) + R_{\angle}(M)}
\end{aligned}
$$

where $R(M_s)$ is probabilistically weighted symbols recall and $R_{\angle}$ is the cosine similarity weighted edge recall. Note that in Layer 2 we match operators even if there are no symbol pairs with operators common between queries and candidates. The edges from these operators to subexpressions are not considered for scoring. This allows us to rank perfect match higher than the match found using commutative property i.e. if the relative position of subexpressions in the candidate is different than query it will be scored lesser.

# 4 Results and Discussion

Qualitative, as well as quantitative results, are provided. Qualitatively, Figure 3 and 4 respectively provides examples of matches where proposed approach works and where it does not work. Observe that in Figure 3 the subexpressions in the candidate expressions is a permutation of the query expression.Figure 3b and 3d also has non-commutative operators once in index and once in baseline. Note that, in 3f the power of $x$ and $z$ are also reordered and yet the query and candidate are perfectly matched.



(a) Query match (b) Candidate match

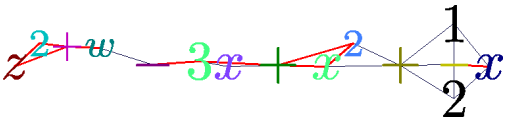
(c) Query match (d) Candidate match

(e) Query match (f) Candidate match

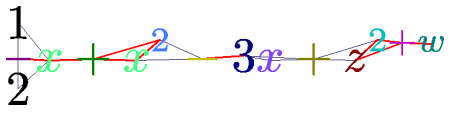
Figure 3: Examples where proposed method can handle commutativity. Query and candidate matches are placed side by side.

From figure 4b you can see that $-3$ is aligned to $/x$ as they are similar looking as / sign is a fraction sign in the actual expression. This approach generates invalid matches when symbols pairs are matched

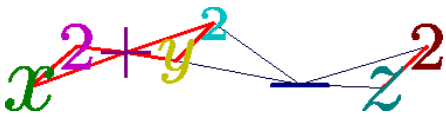
incorrectly to other similar looking symbols.In the figure 4d, it can be seen that $y^2$ is matched to $z^2$ and $-$ sign of the baseline is matched to $-$ sing on the index. It causes whole $x^{2+y^2}$ of the query to match $z^2+x^2$ of the candidate. In the visual domain, we are not using any symbols layout tree or operator tree to figure out which nodes should be matched to each other and that makes it difficult to create valid matches. Despite this, let us see how this approach performs quantitatively.



(a) Query match



(b) Candidate match



(c) Query match

(d) Candidate match

Figure 4: Examples where proposed method can not handle commutativity. Query and candidate matches are placed side by side.

**Benchmark and Protocol.** We test our methods using NTCIR-12 MathIR Wikipedia Formula Browsing Task [31]. There are total 40 queries in total: the first 20 queries are concrete without wildcards, and the remaining 20 queries contain wildcards. Queries 21-40 are produced from the first 20 queries by deleting subexpressions and/or replacing subexpressions with wildcards. The proposed approach does not support wildcards and hence, we replace wildcards with a dummy symbol ('X'). Both Precision@K and BPref are computed using the official competition relevance judgments for this task, and the `trec_eval`[4] tool. One of the queries contains only one symbol which can not be handled by current symbol pair based model resulting in a 0 score for all metrics.

**Indexing and Retrieval.** We rendered about 328,658 unique formula in PNG format and then create an index for each. There are 4185 inverted index entries with 31,726,284 postings. The inverted index itself is 6.5G and size of rendered PNG files is 2.7G.

Table 1: Average Precision@K values per topic for NTCIR-12 MathIR Wikipedia Formula Browsing Task.

| | Relevant(%) | | | Partially Relevant (%) | | |
|---|---|---|---|---|---|---|
| | P@5 | P@10 | P@20 | P@5 | P@10 | P@20 |
| Tangent S | 44.00 | 31.50 | 21.62 | 70.00 | 60.75 | 51.12 |
| Tangent V | 30.26 | 21.28 | 15.51 | 42.56 | 33.85 | 27.05 |
| Tangent VC | 17.95 | 13.33 | 11.62 | 23.59 | 19.74 | 15.51 |

Table 2: Average BPref values per topic for NTCIR-12 MathIR Wikipedia Formula Browsing Task. Results shown for all queries (40) & queries with/without wildcards (20/20).

| | Relevant(%) | | | Partially Relevant (%) | | |
|---|---|---|---|---|---|---|
| | All | No Wildcards | With Wildcards | All | No Wildcards | With Wildcards |
| Tangent S | 55.30 | | | | | |
| Tangent V | 47.54 | 59.70 | 35.99 | 60.51 | 64.86 | 56.38 |
| Tangent VC | 43.00 | 54.52 | 32.06 | 43.08 | 53.81 | 32.88 |

[4]http://trec.nist.gov/trec_eval

Table 1 and 2 gives average precision and average bpref values per topic for NTCIR-12 MathIR Wikipedia Browsing Task. As you can see the precision and bpref values are lower than Tangent V. It can be because while indexing we used the $l^{max}$ value of 3 instead of 10. We obtained the 31,726,284 posting entries instead of 39,466,717 as reported in [4].

**Comparison with Tangent V.** Surprisingly Tangent VC performs better than Tangent V on 'NTCIR-MathWiki-39' query in terms of bpref score. Table 3 compares top four results returned by the Tangent V and Tangent VC for 'NTCIR-MathWiki-39' query. This query is given below:

$$H_{*1*} = \left[\frac{\partial^2 *2*}{*3*}\right]$$

Table 3: Comparison of Top-4 results returned by Tangent V and Tangent VC for the 'NTCIR12-MathWiki-39' query

| **Rank** | **Tangent V** | **Tangent VC** |
|---|---|---|
| 1 | $\vec{u} = \begin{bmatrix} u^1 \\ u^2 \\ u^3 \end{bmatrix} = \frac{d\vec{x}}{dt} = \begin{bmatrix} \frac{dx^1}{dt} \\ \frac{dx^2}{dt} \\ \frac{dx^3}{dt} \end{bmatrix}$. | $\frac{d^2x^\lambda}{dT^2} = -\frac{dx^\nu}{dT}\frac{dx^\alpha}{dT}\left[\frac{\partial^2 X^\mu}{\partial x^\nu \partial x^\alpha}\frac{\partial x^\lambda}{\partial X^\mu}\right]$ |
| 2 | $\varphi = \varphi_b + (z+h)\left[\frac{\partial\varphi}{\partial z}\right]_{z=-h} + \frac{1}{2}(z+h)^2\left[\frac{\partial^2\varphi}{\partial z^2}\right]_{z=-h} + \frac{1}{6}(z+h)^3\left[\frac{\partial^3\varphi}{\partial z^3}\right]_{z=-h} + \frac{1}{24}(z+h)^4\left[\frac{\partial^4\varphi}{\partial z^4}\right]_{z=-h} + \cdots,$ | $\mathrm{E}\left[\frac{\frac{\partial^2}{\partial\theta^2}f(X;\theta)}{f(X;\theta)}\middle\vert\,\theta\right] = \cdots = \frac{\partial^2}{\partial\theta^2}\int f(x;\theta)\,\mathrm{d}x = \frac{\partial^2}{\partial\theta^2}\,1 = 0.$ |
| 3 | $\dot{\hat{x}} = \left[\frac{\partial H(\hat{x})}{\partial x}\right]^{-1} M(\hat{x})\ \mathrm{sgn}(V(t) - H(\hat{x}))$ | $\Delta\lambda = \left[\frac{2\delta n_0 \eta}{\pi}\right]\lambda_B$ |
| 4 | $\dot{\hat{x}} = \left[\frac{\partial H(\hat{x})}{\partial x}\right]^{-1} M(\hat{x})\,\mathrm{sgn}(V(t) - H(\hat{x})) + B(\hat{x})u.$ | $\dot{\hat{x}} = \left[\frac{\partial H(\hat{x})}{\partial x}\right]^{-1} M(\hat{x})\,\mathrm{sgn}(V(t) - H(\hat{x})) + B(\hat{x})u.$ |

Table 4 compares top five results returned by the Tangent V and Tangent VC for 'NTCIR-MathWiki-5' query. This query is given below:

$$1 + \cfrac{1}{2 + \cfrac{1}{5 + \cfrac{1}{5 + \cfrac{1}{4 + \ddots}}}}$$

The Tangent V performs better than Tangent VC for this query when bpref scores are compared. Observe the third result returned by the Tangent VC has a commutative match ($\ddots$ is on the left side of operator +)

Table 4: Comparison of Top-5 results returned by Tangent V and Tangent VC for the 'NTCIR12-MathWiki-5' query

| Rank | Tangent V | Tangent VC |
|---|---|---|
| 1 | $1+\cfrac{1}{2+\cfrac{1}{5+\cfrac{1}{5+\cfrac{1}{4+\ddots}}}}$ | $\frac{1}{2}+\frac{1}{3}+\frac{1}{5}+\frac{1}{7}+\frac{1}{11}+\frac{1}{13}+\cdots\to\infty.$ |
| 2 | $1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{5+\cfrac{1}{1+\cfrac{1}{4+\ddots}}}}}$ | $\frac{1}{2}=\frac{1}{3}+\frac{1}{9}+\frac{1}{27}+\frac{1}{81}+\cdots$ |
| 3 | $e=2+\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{4+\ddots}}}}}$ | $q=a_0+\cfrac{1}{a_1+\cfrac{1}{a_2+\cfrac{1}{a_3+\cfrac{1}{\ddots+\cfrac{1}{a_k}}}}}=[a_0;a_1,a_2,\ldots,a_k]$ |
| 4 | $1+\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{1+\ddots}}}}}$ | $\cfrac{1}{3+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{3+\cfrac{1}{9+\ddots}}}}}$ |
| 5 | $\sqrt{2}=1+\cfrac{1}{2+\cfrac{1}{2+\cfrac{1}{2+\cfrac{1}{2+\ddots}}}}$ | $\sqrt{1+x}=1+\frac{1}{2}x-\frac{1}{8}x^2+\frac{1}{16}x^3-\frac{5}{128}x^4+\frac{7}{256}x^5-\cdots$ |

# 5 Conclusion

We have presented a novel method to handle commutativity and associativity of mathematical operators in visual math search which uses Line of Sight graphs. This method introduces node types in the existing Tangent V system and allows symmetric matches for the nodes of type commutative. Although this method can not match all the permutations of given expression with operands, it matches many cases. This method may be benefited if edge types are also considered while matching symbol pairs. Edge types will let us differentiate between the subscripts, superscripts and restrict us to match symbols on different levels in the expression. For example, match of type shown in 4c and 4d can be avoided. Also, currently it uses a greedy algorithm for growing matches which can be improved by using heuristics while merging alignments or trying multiple combinations and choosing the best. Moreover, it is also necessary to consider the size of symbols and avoid scoring symbol pairs of inconsistent size ratios high. These modifications will improve proposed method but still, it will be challenging to handle properties of mathematical operators like operator range,

dominance, and, precedence given only visual information.

# References


[1] Kenny Davila Castellanos. *Symbolic and Visual Retrieval of Mathematical Notation using Formula Graph Symbol Pair Matching and Structural Alignment*. PhD Dissertation, Rochester Institute of Technology, 2017.

[2] Houssem Chatbri, Paul Kwan, and Keisuke Kameyama. "An application-independent and segmentation-free approach for spotting queries in document images". In: *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE. 2014, pp. 2891–2896.

[3] Kenny Davila and Richard Zanibbi. "Layout and semantics: Combining representations for mathematical formula search". In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM. 2017, pp. 1165–1168.

[4] Kenny Davila and Richard Zanibbi. "Visual Search using Line-of-Sight Graphs: Application to Math Images". In: *Proceedings of the 41st International ACM SIGIR conference on Research and Development in Information Retrieval*. 2018.

[5] Kenny Davila et al. "Tangent-3 at the NTCIR-12 MathIR Task." In: *NTCIR*. 2016.

[6] Liangcai Gao et al. "The Math Retrieval System of ICST for NTCIR-12 MathIR Task." In: *NTCIR*. 2016.

[7] Hiroya Hagino and Hiroaki Saito. "Partial-match Retrieval with Structure-reflected Indices at the NTCIR-10 Math Task." In: *NTCIR*. 2013.

[8] Radu Hambasan, Michael Kohlhase, and Corneliu-Claudiu Prodescu. "MathWebSearch at NTCIR-11." In: *NTCIR*. 2014.

[9] Xuan Hu et al. "Wikimirs: a mathematical information retrieval system for wikipedia". In: *Proceedings of the 13th ACM/IEEE-CS joint conference on Digital libraries*. ACM. 2013, pp. 11–20.

[10] Shahab Kamali and Frank Wm Tompa. "Structural similarity search for mathematics retrieval". In: *International Conference on Intelligent Computer Mathematics*. Springer. 2013, pp. 246–262.

[11] Kevin Kofler, Peter Schodl, and Arnold Neumaier. *Limitations in Content MathML*. Tech. rep. 2009.

[12] Michael Kohlhase and Ioan Sucan. "A search engine for mathematical formulae". In: *International Conference on Artificial Intelligence and Symbolic Computation*. Springer. 2006, pp. 241–253.

[13] Giovanni Yoko Kristianto. "G. Topic, and A. Aizawa. MCAT math retrieval system for NTCIR-12 MathIR task". In: *Proc. NTCIR-12* (2016), pp. 323–330.

[14] Xiaoyan Lin et al. "A mathematics retrieval system for formulae in layout presentations". In: *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM. 2014, pp. 697–706.

[15] Simone Marinai, Beatrice Miotti, and Giovanni Soda. "Mathematical symbol indexing using topologically ordered clusters of shape contexts". In: *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*. IEEE. 2009, pp. 1041–1045.

[16] Simone Marinai, Beatrice Miotti, and Giovanni Soda. "Using earth mover's distance in the bag-of-visual-words model for mathematical symbol retrieval". In: *Document Analysis and Recognition (ICDAR), 2011 International Conference on*. IEEE. 2011, pp. 1309–1313.

[17] Bruce R Miller and Abdou Youssef. "Technical aspects of the digital library of mathematical functions". In: *Annals of Mathematics and Artificial Intelligence* 38.1-3 (2003), pp. 121–136.

[18] Robert Miner and Rajesh Munavalli. "An approach to mathematical search through query formulation and data normalization". In: *Towards Mechanized Mathematical Assistants*. Springer, 2007, pp. 342–355.

[19] Jozef Mišutka and Leo Galamboš. "Extending full text search engine for mathematical content". In: *Towards Digital Mathematics Library. Birmingham, United Kingdom, July 27th, 2008* (2008), pp. 55–67.

[20] Harold Mouchère et al. "ICFHR2016 CROHME: Competition on Recognition of Online Handwritten Mathematical Expressions". In: *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*. IEEE. 2016, pp. 607–612.

[21] Shunsuke Ohashi, Giovanni Yoko Kristianto, Akiko Aizawa, et al. "Efficient Algorithm for Math Formula Semantic Search". In: *IEICE TRANSACTIONS on Information and Systems* 99.4 (2016), pp. 979–988.

[22] Michal Ruzicka, Petr Sojka, and Martin Líska. "Math Indexer and Searcher under the Hood: Fine-tuning Query Expansion and Unification Strategies." In: *NTCIR*. 2016.

[23] Petr Sojka and Martin Líška. "Indexing and searching mathematics in digital libraries". In: *International Conference on Intelligent Computer Mathematics*. Springer. 2011, pp. 228–243.

[24] David Stalnaker and Richard Zanibbi. "Math expression retrieval using an inverted index over symbol pairs". In: *Document recognition and retrieval XXII*. Vol. 9402. International Society for Optics and Photonics. 2015, p. 940207.

[25] Abhinav Thanda et al. "A Document Retrieval System for Math Queries." In: *NTCIR*. 2016.

[26] Abdou Youssef. "Roles of math search in mathematics". In: *International Conference on Mathematical Knowledge Management*. Springer. 2006, pp. 2–16.

[27] Richard Zanibbi and Dorothea Blostein. "Recognition and retrieval of mathematical expressions". In: *International Journal on Document Analysis and Recognition (IJDAR)* 15.4 (2012), pp. 331–357.

[28] Richard Zanibbi and Li Yu. "Math spotting: Retrieving math in technical documents using handwritten query images". In: *Document Analysis and Recognition (ICDAR), 2011 International Conference on*. IEEE. 2011, pp. 446–451.

[29] Richard Zanibbi and Bo Yuan. "Keyword and image-based retrieval of mathematical expressions". In: *Document Recognition and Retrieval XVIII*. Vol. 7874. International Society for Optics and Photonics. 2011, p. 78740I.

[30] Richard Zanibbi et al. "Multi-stage math formula search: Using appearance-based similarity metrics at scale". In: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM. 2016, pp. 145–154.

[31] Richard Zanibbi et al. "NTCIR-12 MathIR Task Overview." In: *NTCIR*. 2016.

[32] Jin Zhao, Min-Yen Kan, and Yin Leng Theng. "Math information retrieval: user requirements and prototype implementation". In: *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*. ACM. 2008, pp. 187–196.

[33] Wei Zhong and Hui Fang. "OPMES: A similarity search engine for mathematical content". In: *European Conference on Information Retrieval*. Springer. 2016, pp. 849–852.