SciMath: A Mathematical Information Retrieval System using Signature Based B Tree Indexing

Sourish Dhar, Abhijit Biswas, Nidhi Singh

Abstract*: *Given a mathematical query, traditional text retrieval systems are not very effective in retrieving mathematical information from scientific documents. This paper presents the design and implementation of a new mathematical information retrieval (MIR) system: SciMath, which can take a mathematical formulae as a query and retrieve the relevant scientific documents consisting the relevant mathematical contents based on a B-Tree indexing scheme. The proposed system is then compared with two classical math-aware search engines to prove its effectiveness.

***Keywords*: Mathematical Information Retrieval (MIR), Ranking, Structure Encoded String (SES), B-Tree indexing.**

I. INTRODUCTION

Information Retrieval (IR) is the technique of presentation, storage, organization of and access to information items. The information should be organized and represented in such a manner that it can be accessed efficiently by the users according to their information need [1]. But dealing with mathematical expressions presents a distinct set of challenges altogether for conventional search engines or IR systems which include[2,3]:

- How any mathematical expression/formulae shall be formulated as a query?
- How to represent the document in the system (internally or externally) for effective and relevant retrieval?
- What indexing scheme and similarity measure can be used for faster retrieval?

As we often come across, in many scientific documents mathematical formulae are embedded in order to explain the theoretical concepts about the scientific documents. Therefore, mathematical information retrieval (MIR) systems are required to find the relevant formulae. MIR systems are formulae based search engines which help us in retrieving information in those documents that contains mathematics. Mathematical retrieval system is of significant use both in technical fields that use mathematics frequently (e.g. Science, Technology, Engineering & Mathematics) [4].

Being highly structured in nature, mathematical formulae are becoming extensively available. In order to store these mathematical formulae or expressions, a MIR tool is required which can index these documents efficiently. Distinct communities use various conventions for naming variables (e.g. using variance vs. v) making it difficult for naive users to interpret it automatically. Due to specific characteristics of formulae, conventional search engines do not work well for the following reasons such as:

Manuscript published on 30 September 2019.
*****Correspondence Author(s)
Sourish Dhar , Department of CSE, Assam University, Silchar, India.
Abhijit Biswas*, Department of CSE, Assam University, Silchar, India.
Nidhi Singh, , Department of CSE, Assam University, Silchar, India.

A. Normalization

Normalization is the process of reducing/removing of notational ambiguity among mathematical expressions as a formulae can be written having the same structure but different variable and constant list[2,3]. For e.g.

$$a^2 + b^2 \quad vs \quad x^2 + y^2$$

B. Indexing

Indexing refers to the data structure for the purpose of storing the key features and terms of a document collection in order to enhance speed and performance in finding relevant documents for a search query[5] . Mathematical expressions are diificult to index using the popular inverted index scheme as the structure of mathematical expression may be non-linear .

C. Ranking and Similarity

An efficient ranking measure must be applied in order to rank structurally similar math formulae whereas similarity measure is still an open challenge for scientific documents having mathematical expressions.

Taking into account of the challenges mentioned above, we are proposing a mathematical retrieval system, **SciMath** with B-Tree indexing scheme for mathematical documents which would facilitate faster retrieval with improved precision. The proposed system SciMath converts the data into linear format using the concept of structure encoded string (SES) before parsing and tokenizing. For indexing, B-Tree data structure is employed because of its effectiveness as it takes *O (log n)* for insertion, deletion and searching operations. To evaluate the effectiveness of SciMath we use standard evaluation measure of IR systems namely precision and discounted cumulative gain (DCG) and further compare it with two other state-of-the art systems i.e. Math Indexer and Searcher (MIaS) and Wikimirs. The remains of the paper is categorized as follows: Related works which are based on text-based technique of indexing is discussed in section II. In Section III , we discuss our proposed approach and its working. Section IV describes and interprets about the experimental results of SciMath and in section V, we conclude our argument.

II. RELATED WORK

Ever since 2003, the field of mathematical information retrieval has been researched , and many prototypes of MIR system are developed till date. The module of a complete mathematical retrieval system consists of mainly Tokenizer, Indexer, Ranker and Interface.

The indexing of mathematical retrieval system is done mainly using two techniques: text-based and tree-based [5]. As our approach is based on text-based indexing technique, we examine the admissible approaches based on text-based indexing technique as well as the encoding scheme of mathematical formulae presentation, indexing scheme and ranking techniques. The system "QUALIBETA" as described in [6] at the NTCIR-11 Math 2 Task by Pinto et. al. (2014) is a combination of two mechanisms: the first mechanism includes the feature extraction process of the math formulae. The features extracted from the math formulae includes the sets of the identifiers, formulae type, constants and operators. The second mechanism included the representation of the formula model collection in a sentence level which described the formulae. Content MathML was used as formulae encoding scheme. For indexing the math formulae and query processing, Elastic Search[1] was used. The presence of variables in the formulae was handled incorrectly while feature extraction phase. Relying only on content MathML degraded the performance of their system and resulted in below average among their competitors as their system was unable to return relevant documents after combining the context and the formula features . Radu Hambasan et. al .(2014) introduced MathWebSearch (MWS) system [7] and depicted its results in the Math-2 task in the NTCIR-11 Information retrieval challenge. The central idea of the system aimed at generating persistence index, amalgamating formulae and keyword search. The index of the system was created using Apache Solr-ElasticSearch (Eso). The formulae were encoded in using content MathML. The amalgamation of keyword search, scalability and index persistency can be enhanced . Authors Giovanni Yoko Kristianto et. al. (2016), in their approach, introduced "MCAT" system [8] with three primary modules which included generation of dependency graph of math expressions which improved precision of their system, score normalization which also resulted good impact on the search performance of the system, formulae unification which further enhanced their system and cold start weight was evaluated. Presentation MathML [9] was used as the encoding format for formulae presentation and for searching, the authors have used Apache Solr[2] for their system. The cold start weight evaluated using multiple linear regression outperformed because of the negative weights obtained for many database fields resulting in decreasing the similarity score of the retrieved documents. The tf-idf scoring measure failed in handling the extremely common textual terms. Michal Ruzicka et al (2016), extended Math Indexer and Searcher (MIaS) [10, 11] by adding new abilities to the system which included MathML structural unification and operator unification. They further combined the features of the system by adding the functionality of canonicalization which normalized the different notations in MathML resulting in optimization of similarity search. But normalization also failed in preserving the full semantic information of the original formulae. Authors Martin Liska et al (2013), based their approach on the unevenness of Math Indexer and Searcher (MIaS) [10] system. The system carried out the preprocessing steps which included canonicalization, sub-formulae searching and tokenization of the expressions followed by extracting the sub trees. Adding with generalization techniques for variables of unification, number constants unification and also keeping track of fonts typeface as well. Lucene is employed as the searching library of their system. To increase the precision of the system performance, they rank the indexed expressions based on its distance with the raw math formulae, which are not tokenized, assigning them a weight. The user interface of MIaS can take query encoded both in LaTeX [12] as well as MathML. The scores of MIaS resulted in a slight increase whereas other systems score doubled when compared, taking the parameters of partially relevant and relevant results. The primary cause of not retrieving partially relevant results of MIaS, might be because their system failed to tokenize the queries formulated by the user. It also failed to extract the sub expressions from the queries with higher complexities. And so, it resulted in searching for the query expressions as a whole. Therefore MIaS, resulted in the retrieving the query as whole, i.e., it either resulted exact match results or with no results [11]. Another Mathematical Information Retrieval system for Wikipedia "Wikimirs", by Xuan Hu et al (2013) includes modules such as Preprocessor, Tokenizer, Indexer and Ranker[13]. They first parsed their Wikipedia dataset in order to extract the Latex markups, tagged under <math> and </math>. To avoid the collision of typesetting variations, normalization is carried out. Before tokenization, they generalize the mathematical formulae for simplifying the complicated formulae to short and simple one. The Latex markups are then parsed into an internal presentation tree and tree are constructed. The presentation is traversed in depth first search manner which is further normalized. For query processing, inverted index is used as the indexing scheme. The similarity score is evaluated by summing the matches in different levels of the tree and matches between generalized terms. Wikimirs system failed to retrieve the formulae which were semantically equivalent to the query formulated by the user as mathematical expression themselves are totally vague. Table I summarizes all the systems discussed in this section

[1] Elastic Search: https://www.elastic.co/products/elasticsearch

[2] Apache Solr: https://lucene.apache.org/solr/

TABLE I: Indexing & Ranking scheme of existing Math aware system

System Name	Tokenizer	Indexing	Ranking	Format
IFISB QUALIBETA	Text-based	Inverted-index	/	Content MathML
MathWebSearh (MWS)	Text-based	Inverted-index	/	Content MathML
MCAT	Text-based	Inverted-index	tf-idf	Presentation MathML
MIRMU	Tree-based	/	/	Presentation MathML, Content MathML
MIaS	Tree-based	Inverted-index	/	Presentation MathML, Content MathML
Wikimirs	Tree-based	Inverted-index	tf-idf	LaTeX

Most of the system discussed in Table I although having very good recall but suffers from low precision. The reasons for this limitation can be manifold. Like in indexing most of the systems used inverted index which was not able to capture the semantic information conveyed by a mathematical expression. Again lack of a consistent normalization and unification scheme also contributed towards irrelevant results while similarity measures (Euclidean or Non-Euclidean) still need to be examined in greater depth for MIR systems. We have employed B-Tree data structure as the indexing scheme for our proposed system SciMath along with the concept of structure encoded string (SES) for the process of normalization and bit vector signature mapping for unification. To compare SciMath, we have chosen two classical MIR systems MIaS and Wikimirs because of the availability of the dataset and similarity of the approach .

III. PROPOSED SYSTEM

A. *Overview*

The objective of our proposed approach is to design a MIR system called SciMath using signature based B Tree indexing scheme for efficient storage and fast retrieval resulting in retrieving relevant scientific documents consisting of relevant math formulae depending on their similar structures. The overview of our proposed system is illustrated in Fig 1. The work flow of our approach is divided into two modules: “Offline Indexing and Online Retrieval” . Offline Indexing further consists of four modules: Math Extractor, SES Convertor, Bit Vector Generator And B-Tree Indexer. In online retrieval, a mathematical expression or formulae as a query is considered as an input formulated by the user searched in the index.

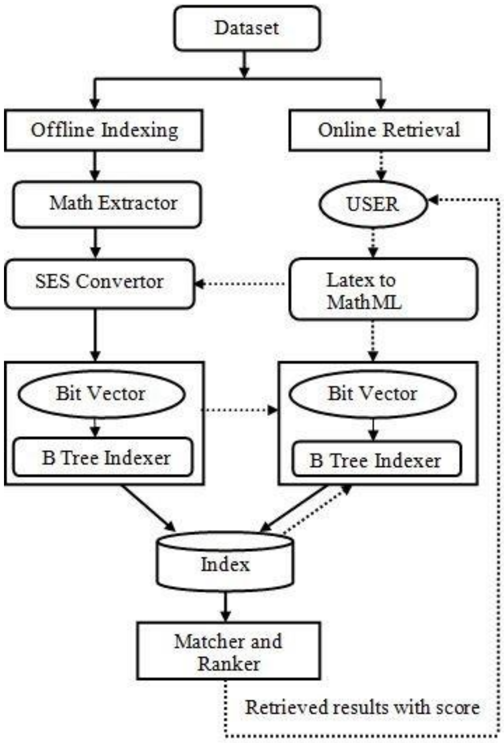


Fig 1: Proposed System: SciMath (Solid line represents Offline Indexing and dotted line represents Online Retrieval)

B. *Document Preprocessing*

Dataset of NTCIR12 MathIR Wiki Corpus v2.1.0 is used in our work where the math formulae are represented using Presentation MathML . The document set contains total 319,689 articles and 592,443 (tagged) formulae. The corpus has been grouped into 160 parts, consisting around 2000 articles each, located in subdirectories with the prefix wpmath. Each part contains articles in the form of .html, stored in the Article and Article LaTeXML Errors sub-directories. Some additional book-keeping data about the archive contents and conversion process are included in the conversion data subdirectories. We initially discarded a few tags like <mtext>, <mspace>,<mstyle> etc. as these tags does not have much contribution regarding semantic of the mathematical structure of mathematical notation. Then, we represented the math formulae in a string format using SES representation [14].

C. *Math Extractor*

We extract all the mathematical expressions from all documents in the dataset through math extractor. This extraction is carried away by parsing of these documents by identifying the markup written in between <math> and </math> tags. At the time of parsing, we have discarded certain tags of MathML such as <maction>(used for binded actions for sub-expressions), <mglyph>(used for non-standard symbols), <mlongdiv>(used for long division notation),<mphantom>(used for invisible content with reserved space),

<mpadded>(used for Space around content),
<mstyle>(used for text styling),
<ms>(used for string literal) etc.
As these extraneous tags plays no significant role in the structural meaning of math formulae. The removal of these extraneous symbols was taken into account because it may degrade or lead to inconsistent result while matching and ranking of the documents based on a query.

D. *Structure Encoded String (SES)*

As our approach is based on text-based indexing technique, we have transformed the mathematical formulae or expressions into strings in order to index them by using the concept of structure encoded string as described in [14] . The authors have proposed a string matching algorithm, in which they have transformed the recognition output and ground truth in Latex form to structured encoded strings (SES) [14]. As explained in [14] Mathematical symbols or expressions are surrounded with six regions mainly. These six regions are named as top-left, above, top-right, bottom-left, below and bottom-right regions, and the top-most part including top-left, above, top-right regions are named as the northern regions represented as N. The region bottom-left, below and bottom- right these whole sub expressions are named as the southern regions represented by S, which is illustrated in Fig. 2. For the northern region, we use NS for starting of northern sub expression and NE for ending of the northern sub expression. For the southern region, we use SS for starting of southern sub expression and SE for ending of the southern sub expression.

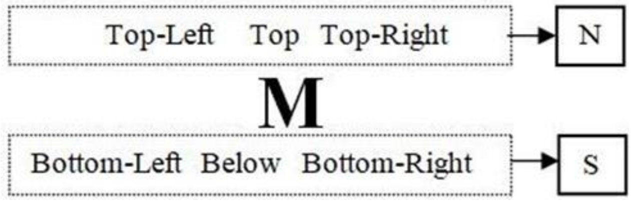


Fig. 2: Spatial regions around a mathematical symbol M

To illustrate the process let’s take the example of a quadratic equation as given below:

$$ax^2 + bx + c = 0$$

Considering the above equation a,x,+,b,c,,=,0 represents the base of mathematical symbol (M) and superscript of x i.e. 2 is in the northern region represents top right (TR).

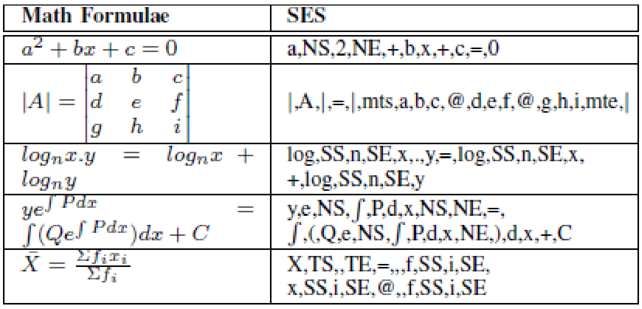
The SES representation of the given equation is generated as:

a, x, NS,2,NE,+,b,x,+,c,=,0

NS represents start of the northern region and NE encodes the end of the northern region. After extracting the mathematical expressions from the documents, we generated the equivalent SES. This conversion is completed by analyzing the tag of superscript and subscript section representation of mathematical notation. Table II shows the meaning of spatial regions around a mathematical symbol and Fig. 3 illustrates few examples of mathematical formulae and its corresponding SES Conversion.

Table- II: Meaning of Spatial regions

Symbols	Meaning of Symbols
M	Mathematical Symbol (Base)
N	Northern Region
S	Southern Region
TL	Top Left
A	Above
TR	Top Right
BL	Bottom Left
B	Below
BR	Bottom Right

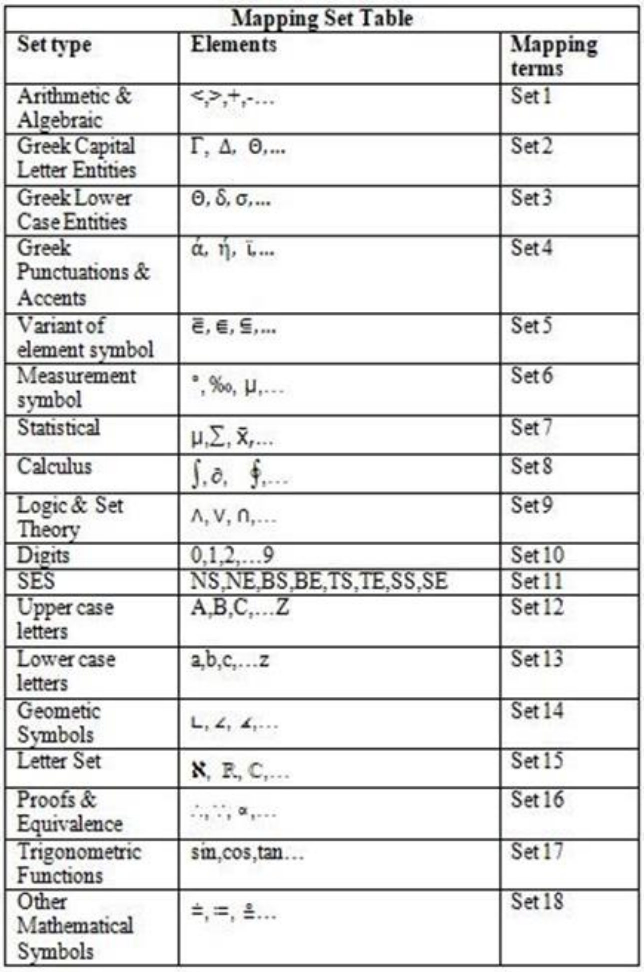


Math Formulae	SES
$a^2 + bx + c = 0$	a,NS,2,NE,+,b,x,+,c,=,0
$\|A\| = \begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix}$	|,A,|,=,|,mts,a,b,c,@,d,e,f,@,g,h,i,mte,|
$log_n x.y = log_n x + log_n y$	log,SS,n,SE,x,.,y,=,log,SS,n,SE,x, +,log,SS,n,SE,y
$ye^{\int Pdx} = \int(Qe^{\int Pdx})dx + C$	y,e,NS,∫,P,d,x,NS,NE,=, ∫,(,Q,e,NS,∫,P,d,x,NE,),d,x,+,C
$\bar{X} = \frac{\Sigma f_i x_i}{\Sigma f_i}$	X,TS,,TE,=,,,f,SS,i,SE, x,SS,i,SE,@,,f,SS,i,SE

Fig.3. SES Conversion of some mathematical expressions.

E. *Mapping of SES and bit vector (Signature) generation*

In this step, we have created a mapping table of all possible mathematical symbols. We have created sets of mathematical symbols corresponding to its particular category, falling under a specific set. Like +,-,* falls under the category of arithmetic operators given with set name as Set 1. Similarly we have created around 18 sets of all possible math symbols. A snapshot of the list is illustrated in Fig.4.



Mapping Set Table

Set type	Elements	Mapping terms
Arithmetic & Algebraic	<,>,+,-…	Set 1
Greek Capital Letter Entities	Γ, Δ, Θ,…	Set 2
Greek Lower Case Entities	θ, δ, σ,…	Set 3
Greek Punctuations & Accents	ά, ή, ΐ…	Set 4
Variant of element symbol	⋹, ∈, ⋸,…	Set 5
Measurement symbol	°, ‰, μ,…	Set 6
Statistical	μ,Σ, x̄,…	Set 7
Calculus	∫, ∂, ∮,…	Set 8
Logic & Set Theory	∧, ∨, ∩,…	Set 9
Digits	0,1,2,…9	Set 10
SES	NS,NE,BS,BE,TS,TE,SS,SE	Set 11
Upper case letters	A,B,C,…Z	Set 12
Lower case letters	a,b,c,…z	Set 13
Geometic Symbols	∟, ∠, ∡,…	Set 14
Letter Set	ℵ, ℝ, ℂ,…	Set 15
Proofs & Equivalence	∴, ∵, ∝,…	Set 16
Trigonometric Functions	sin,cos,tan…	Set 17
Other Mathematical Symbols	≐, ≔, ≙…	Set 18

Fig. 4: Example of Mapping Set Table

Mathematically mapping function can be defined as:

$$\sum_{i=0}^{n} f(w_i) = \{1, when\ there is\ a\ match\ ;0 \quad otherwise\}$$

where i is the token in the mathematical expressions and
n is the length of the tokens.

To create a bit vector signature of our SES , all the tokens of particular SES is matched against the mapping table where we generate document signature of SES associated to a bit vector which may take value 0 when there is no match for a particular symbol in the mapping set table and 1 when there is a match for a particular symbol in the mapping set table. Each bit of the bit-vector is sequentially scanned for matches.

F. *Indexing*

B-Tree data structure is employed for indexing of our dataset because of its productive features as the operations of insertion, deletions and searching are done in O(log n) time complexity. In B tree, the height is maintained to be as low as possible as the overall disk entries for insertion, deletion as well as search operations are remarkably minimized in comparison to other balanced binary search trees as such AVL tree, B+ trees,Red-Black tree, etc. B Tree of order m

Algorithm 1: Mapping Algorithm for SES string

```
Input  : SES string,Mapping Set Table
Output: Mapping string
Initialize a HashMap
$key- > string$
$values- > Array[strings]$
Initializing addvalues(string,string)
call addvalues(parameters:string key,string value)
Initialize and set list to NULL
If (hashmap contains key)
List=get the value corresponding to the key from
  hashmap
if(list==NULL)
create new List
Add value to list
end if
else
create new list Add value to list
hashmap.put(key,list)
// This is how to put:
$Hash < string, string > m$
$= newHashmap < string, string > ();$
m.put("a","b");
end else
```

satisfies the following properties:

- In B Tree, each node consists maximum of m children.
- The internal nodes has minimum of | m| children.
- The root node must have minimum of two children if it is non-leaf node.
- A non-leaf node with k children must have k-1 keys. The child between two keys k1 and k2 contains all keys in the range from k1 and k2.
- All the leaf nodes must be at same level.
- All the values as keys in a node must be sorted in Ascending Order

Unlike Binary Search Tree, B Tree grows as well as shrinks towards the root whereas, Binary Search Trees grows down- ward and also shrinks from downward. Like other balanced Binary Search Trees, time complexity of B Tree to perform search, insert and delete operations is O(log n) [15]

During our study of already developed MIR systems, we found that almost all the MIR systems are built on Lucene, Elastic search etc. where the indexing is done using inverted index data structure. In our system, we employed B Tree data structure for indexing, as the objective of our proposed work is to design an MIR system for fast and efficient retrieval. When compared with other data structures like B+ Tree, which stores redundant data hence resulting in increasing usage of space but when it comes to B Tree, it banishes the duplicate storage of search key values. And resulting in the advantage of space, as repetition of data storage does not occurs and can be used for large indices. B Tree can be faster for retrieving results when we are looking for a search key as there are fewer comparisons involved, to obtain the data we are looking for[16]. On the other hand, though in B+ Tree,as the extreme nodes are chained with linked list which enables much easier and higher performing to do a full scan searching. But the total disk accesses of B Tree for most of the operations including insert,delete and searching are minimized remarkably in comparison to other balanced Binary Search Trees like B+ Tree, AVL Tree, Splay Tree etc.[16].

G. *Online Retrieval*

In the online Retrieval phase, a LaTeX query string is considered as an input. This LaTeX query then is converted to Presentation MathML (P-MML) using SnuggleTex convertor [17]. The converted MathML query is then converted to SES string by the SES conversion process. The converted SES string is mapped to the mapping set table for the generation of bit-vector with signature based bit-vector representation. The bit pattern generated by the bit-vector is then fed to the B Tree indexer will give the relevant documents ranked in descending order using Jaccard similarity measure[18,19].

$$Score = \frac{A \cap B}{A \cup B}$$

SciMath uses Jaccard similarity measure[**19**] for matching the query and retrieved documents. Jaccard similarity measure is used to evaluate the likeness between two sets, sets A and B, given by the following expression.The numerator represents the commonality between A and B, and the denominator represents Union of A and B. The implementation of Jaccard distance works at token level, where it compares two strings by first tokenizing them and then dividing the number of tokens shared by the strings by to the total number of tokens.

$$Rank(q,d) = \frac{Similarity\ Match\,(q_i, d_j)}{Total\ Distinct\ Token\ in\ (q, d_j)}$$

$$where\ q_i\,(query) \in Q(Query\ Collection),$$

$$d_j \in D(Document\ Collection)$$

And finally we retrieve top k documents in descending order based on their score . If two or more documents gets the same rank,they are ordered on First Come First Serve basis.

IV. RESULTS AND DISCUSSION

A. *Dataset*

Dataset of NTCIR12 MathIR Wiki Corpus v2.1.0 [20] is used in our work where the math formulae are represented using Presentation MathMLThe. set of documents in total con- sists of 319,689 articles and 592,443(tagged) formulae. The corpus has been grouped into 160 parts, consisting around 2000 articles each, located in subdirectories with the prefix wpmath. Each part contains articles in the form of .html, stored in the Article and Article LaTeXML Errors sub-directories. Some additional book-keeping data about the archive contents and conversion process are included in the conversion data subdirectories.

The dataset provided by NTCIR12 ia available publicly as MathIR Wiki Corpus v2.1.0 which includes the math tagged mathematical formulae under the directory MathTagArticles and text articles under TextArticles directory.[20]

B. *Evaluation Measures*

The performance of our system is evaluated based on two parameters: Precision and Discounted Cumulative Gain(DCG) which is briefly discussed in the below sections.

Precision: Precision (p) is the fraction of relevant document among the retrieved document [21,22].

$$\Pr ecision(P) = \frac{\text{Re}levant\ Items\ \text{Re}trieved}{Total\ \text{Re}trieved\ \ Items}$$

Precision @ k : Precision @ k is the fraction of recommended items in the top-k set that are relevant. Its interpretation is as follows. Suppose that my precision at 10 in a top- 10 recommendation problem is 80%. This means that 80% of the recommendation I make are relevant to the user[21,22].

Mathematically precision@k is given by :

$$\Pr ecision\,@\,k = \frac{\text{Re}trieved\ \ Items\ @\,k\ that\ are\ relevant}{\text{Re}trieved\ \ Items\ @\,k}$$

Discounted Cumulative Gain (DCG):

The standard of ranking is estimated using Discounted cumulative gain (DCG)[21-24]. In information retrieval, DCG is frequently employed to evaluated the productiveness of web search engine scale of documents in a search-engine result set, DCG measures the usefulness, or gain, of a document based on its algorithms or related applications [21-24]. The gain is assembled from the top of the result list to the bottom, with the gain of each result discounted at lower ranks.

DCG is calculated by :

$$DCG\ \ @\,K = \sum_{i=1}^{k} \frac{rel_i}{\log_2 (i+1)}$$

We then compared the performance of our system, SciMath based on the evaluation metrics, Precision and DCG, with two other Math Search Engines (MSE) known as Math Indexer and Searcher (MIaS) and Wikimirs. The Testing or experimental result for our system is done using 100 queries which is tabulated in the appendix.

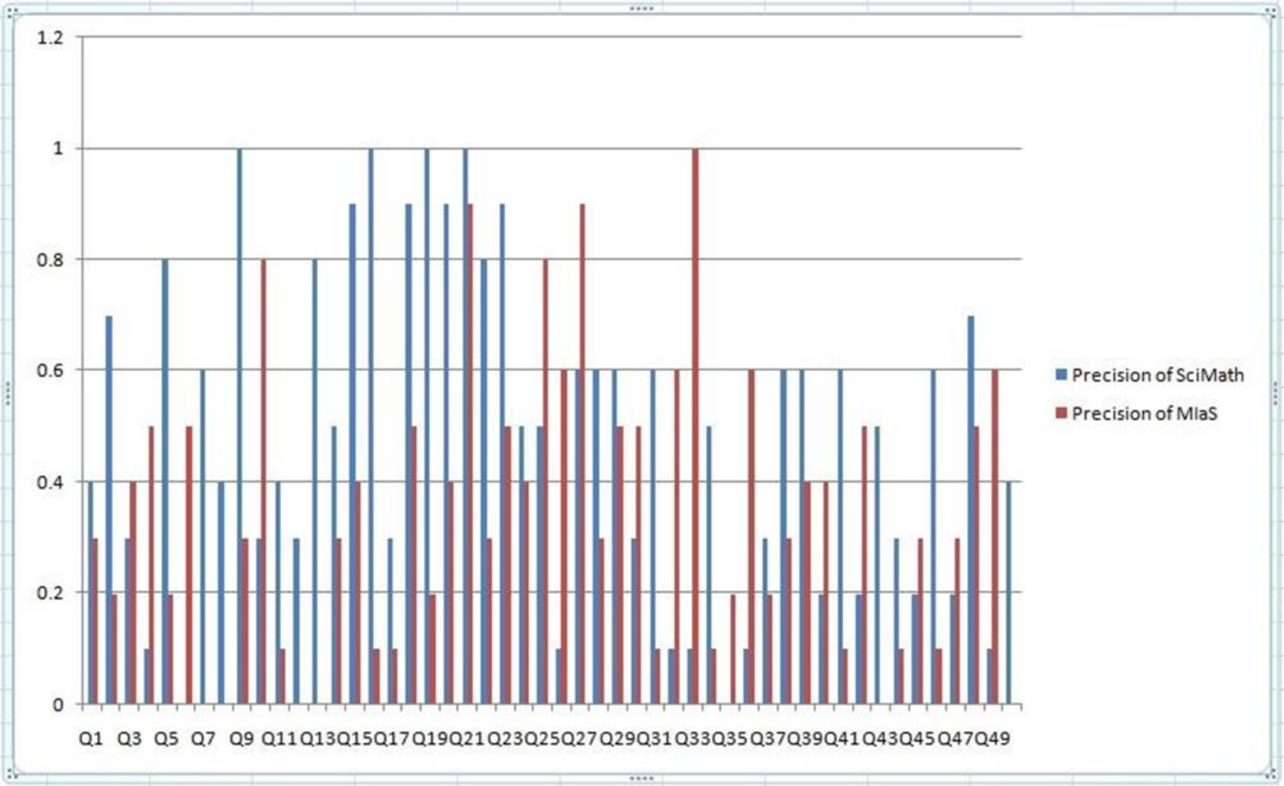


Fig. 5: Precision of SciMath vs MIaS

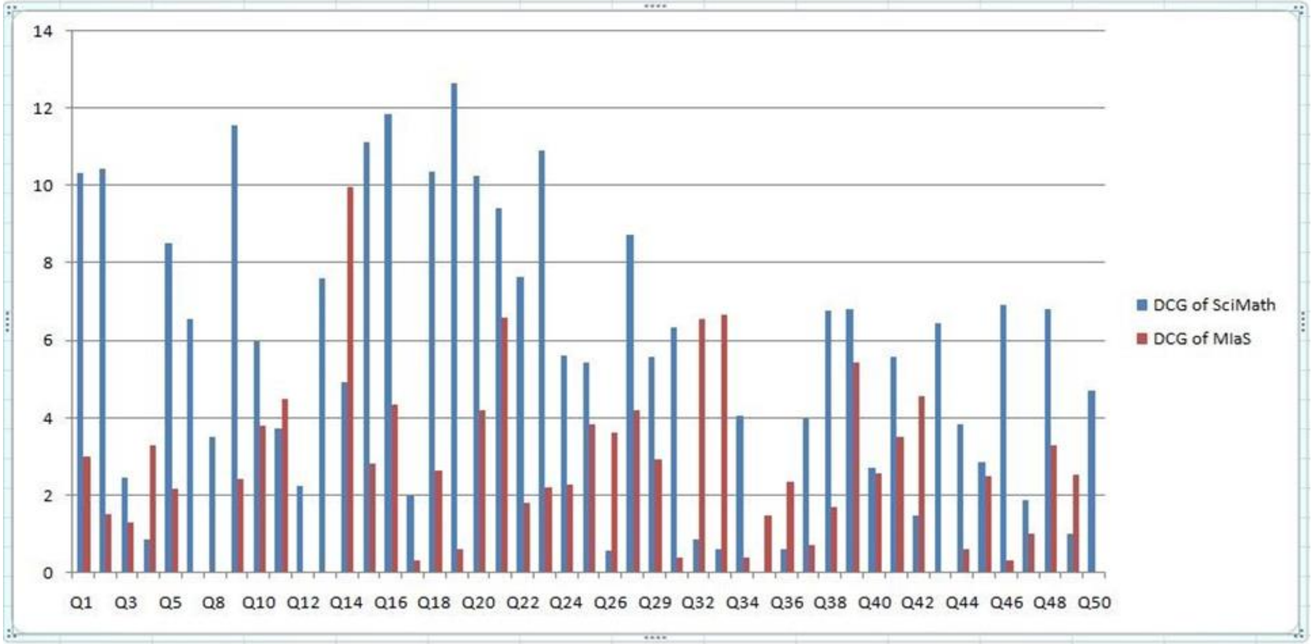


Fig. 6: DCG Comparison of SciMath vs MIaS

Retrieval Number: K12980981119/19©BEIESP
DOI: 10.35940/ijitee.K1298.0981119
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

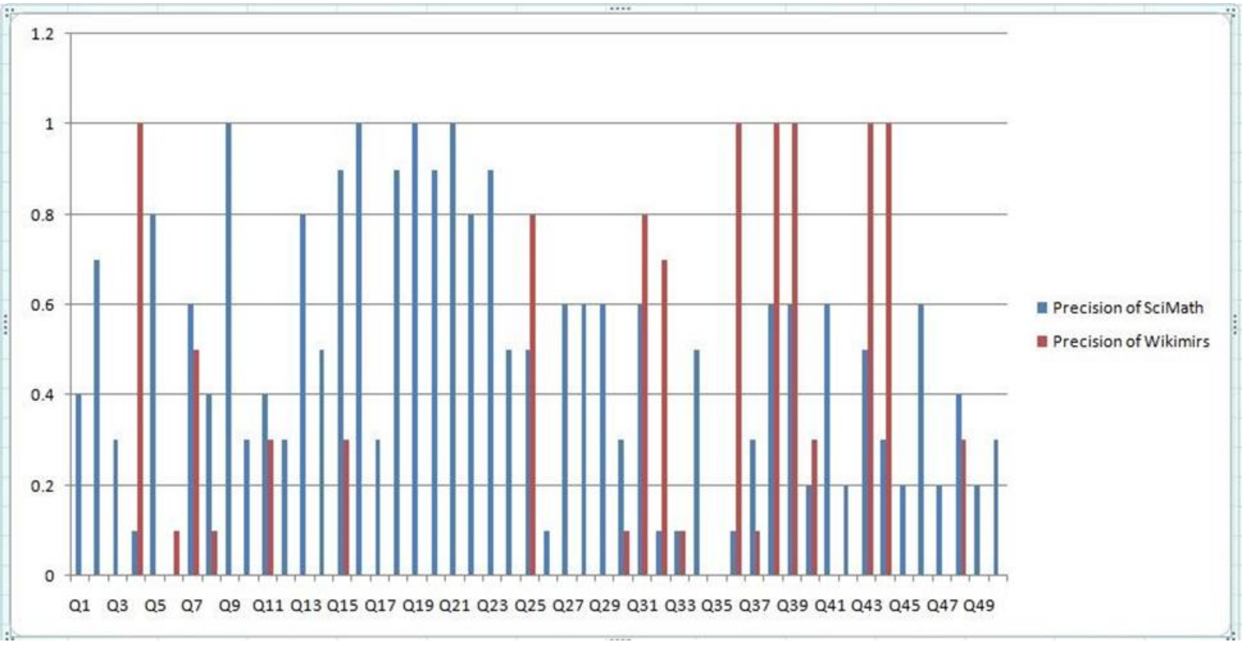


Fig. 7: Precision of SciMath vs Wikimirs

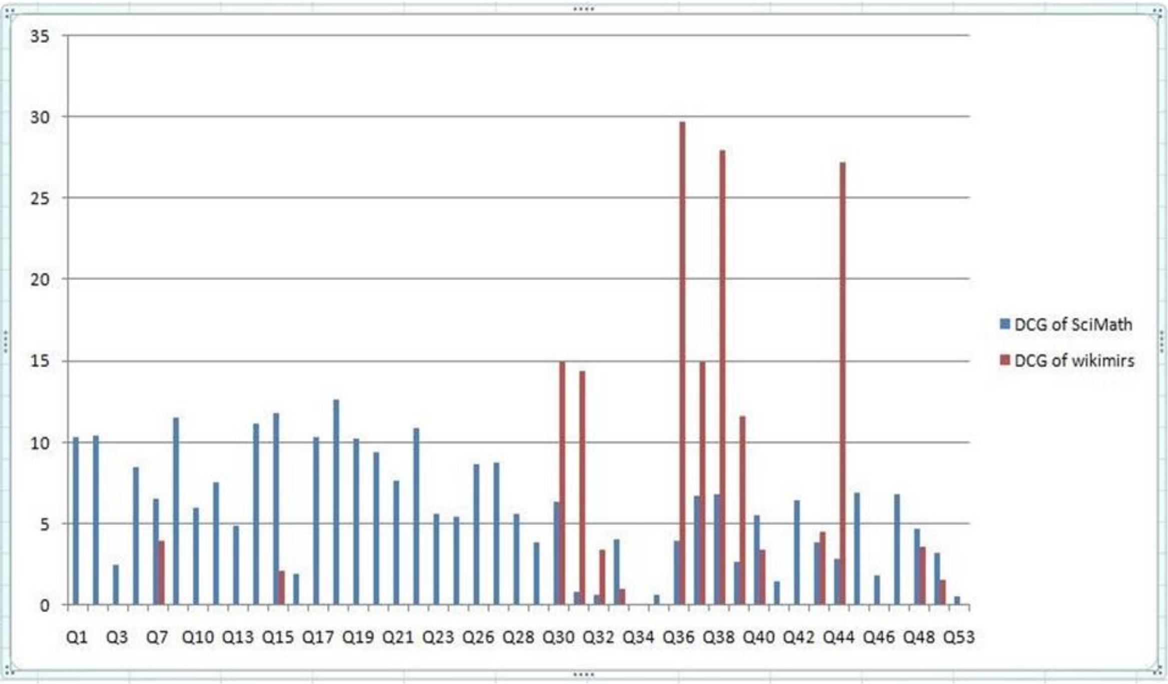


Fig. 8: DCG of SciMath vs Wikimirs

We have illustrated the comparison of our system i.e SciMath and MIas along with wikimirs Fig 5,6,7 and 8. We have illustrated the experimental result considering the top 50 queries, for the lack of space for plotting the graph. Our system, SciMath generated better precision and DCG when compared to MIaS as illustrated in the Fig 5 and 6.

We tested our system with the 100 queries and SciMath yielded an improved precision and increased DCG than MIaS because:

- SciMath resulted in retrieving both partially relevant and exact results as well.
- MIas failed in retrieving partially relevant results.
- MIaS also failed to extract the sub expressions from the queries with higher complexities, i.e., it either resulted exact match results or with no results.

From Fig 7 and Fig 8, we can infer that the precision of our system, outperformed WikiMirs as well. As we found that for many queries WikiMirs did not even retrieved any results.

Wikimirs failed to retrieve any result for many queries among the 100 considered queries. Wikimirs also failed to retrieve results which were semantically equivalent to the query formulated by the user.

For the queries with query id Q1 to Q3, Q13 to Q25, Q27 to Q30, Q56 to Q58, Q79 to Q81and Q89 to Q91 it resulted with either no results or totally irrelevant results. Hence, Wikimirs resulted in poor precision because it failed to retrieve irrelevant results for many queries.

V. CONCLUSION AND FUTURE SCOPE

This paper demonstrated a mathematical information retrieval system SciMath which uses a signature based B tree indexing scheme for efficient indexing and fast retrieval of scientific documents based on a formula query representing an information need. The proposed system can take a LaTeX query string as an input and provide the scientific documents that contains the relevant mathematical formulae or expressions. Internally the system uses P-MML data format which is converted to a string using the notion of Structured Encoded String (SES) which not only converts the given MathML document into a string but also retain its structural information. Further this SES string is then converted into a signature of bit vectors using a mapping table and is indexed using the B-Tree data structure. Our system performed fair enough for queries which included math equation with exponentiations, polynomials etc. but yielded a poor result for queries consisting of trigonometric functions. These can be fixed in near future by improvising the mapping set table with all the required math symbols. We have used Jaccard similarity measure for ranking purpose in our system, SciMath. Different perspective of ranking and similarity measures such as Levenshtein Edit Distance, Cosine Similarity, Hamming Distance, Dice Coefficient etc. can be applied which can strengthen the system providing a better result according to the users information need. Also some novel weighting schemes can also be explored to improve the precision furthermore.

REFERENCES

1. Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to Information Retrieval. New York, NY, USA: Cambridge University Press, 2008. ISBN: 0521865719, 9780521865715.
2. Richard Zanibbi and Dorothea Blostein. “Recognition and Retrieval of Mathematical Expressions”. In: Int. J. Doc. Anal. Recognit. 15.4 (Dec. 2012), pp. 331–357. ISSN: 1433-2833. DOI: 10.1007/s10032- 011-0174- 4. URL: http://dx.doi.org/10.1007/s10032-011-0174-4.
3. Ray R. Larson, Chloe Reynolds, and Fredric C. Gey. “The Abject Failure of Keyword IR for Mathematics Search: Berkeley at NTCIR-10 Math”. In: Proceedings of the 10th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-10, National Center of Sciences, Tokyo, Japan, June 18-21, 2013.
4. NTCIR-12. MathIR. http : / / ntcir - math . nii . ac . jp / introduction/. Accessed on June 1, 2018.
5. Ferruccio Guidi and Claudio Sacerdoti Coen. “A Survey on Retrieval of Mathematical Knowledge”. In: Mathematics in Computer Science 10.4 (2016), pp. 409–427. ISSN: 1661-8289. DOI: 10.1007/s11786-016-0274-0.
6. Pinto, Jos Mara Gonzlez, Simon Barthel, and Wolf-Tilo Balke. ”QUALIBETA at the NTCIR-11 Math 2 Task: An Attempt to Query Math Collections.” In: Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, Tokyo, Japan, pp.103-107, December 9-12, 2014.
7. Hambasan, Radu, Michael Kohlhase, and Corneliu-Claudiu Prodescu. ”MathWebSearch at NTCIR-11.” In: Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, Tokyo, Japan, December 9-12, 2014.
8. Kristianto, Giovanni Yoko, Goran Topic, and Akiko Aizawa. ”MCAT Math Retrieval System for NTCIR-12 MathIR Task.” In:Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies, Tokyo Japan, June 7-10, 2016
9. Mathematical Markup Language (MathML). https://en.wikipedia.org/wiki/MathML. Accessed June 1, 2018.
10. Liska, Martin, Petr Sojka, and Michal Ruicka. ”Similarity search for mathematics: Masaryk university team at NTCIR-10 math task.” Proceedings of the 10th NTCIR Conference on Evaluation of Information Access Technologies, 2013
11. Ruzicka, Michal, Petr Sojka, and Martin Liska. ”Math indexer and searcher under the hood: Fine-tuning query expansion and unification strategies.” In:Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies, Tokyo Japan, June 7-10, 2016.
12. LaTeX – A document preparation system. https://www.latex-project.org. Accessed June 15, 2018.
13. Hu, Liangcai Gao, Xiaofan Lin, Josef B. Baker. ”Wikimirs: a mathematical information retrieval system for wikipedia.” Proceedings of the 13th ACM/IEEE-CS joint conference on Digital libraries. ACM, 2013.
14. Kumar, P. Pavan, Arun Agarwal, and Chakravarthy Bhagvati. ”A string matching based algorithm for performance evaluation of mathematical expression recognition.” Sadhana 39.1 (2014): 63-79.
15. Cormen, Thomas; Leiserson, Charles; Rivest, Ronald; Stein, Clifford Introduction to Algorithms (Second ed.), MIT Press and McGraw-Hill, pp. 434–454,2001. ISBN 0-262-03293-7.
16. Knuth, Donald , Sorting and Searching, The Art of Computer Programming, Volume 3 (Second ed.), Addison-Wesley, 1998. ISBN 0-201-89685-0.
17. SnuggleTeX (1.2.2) homepage. https://www2.ph.ed.ac.uk/snuggletex/documentation/overview-and-features.html. Accessed on June 1, 2018.
18. Sven Kosub. “A note on the triangle inequality for the Jaccard distance”. In:CoRR abs/1612.02696 (2016).
19. J.; Naenudorn E.; Wanapu S. Niwattanakul S.; Singthongchai. “Using of JaccardCoefficient for Keywords Similarity”. In: International MultiConference of Engineers and Computer Scientists;2013, Vol. 1 (2013).
20. NTCIR-12 MathIR Task Data: http://ntcir-math.nii.ac.jp/data/

C. J. Van Rijsbergen. Information Retrieval. 2nd. Newton, MA, USA: Butterworth-Heinemann, 1979. ISBN: 0408709294.

21.Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. Modern Information Retrieval. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1999. ISBN:020139829X.

22.David A. Grossman and Ophir Frieder. Information Retrieval: Algorithms and Heuristics. Norwell, MA, USA: Kluwer Academic Publishers, 1998. ISBN: 0792382714.

23.Joydip Datta. A Mtech Seminar Report : Ranking in Information Retrieval. Department of Computer Science and Engineering,Indian Institute of Technology, Bombay, 2013.

APPENDIX : QUERY SET

Query ID	Latex Query	Mathematical Notation
Q1	(\neg q \vee \neg q) \leftrightarrow (p \to \neg q)	$(\neg q \vee \neg q) \leftrightarrow (p \to \neg q)$
Q2	(a-b)2 = a2+b2-2ab	$(a-b)^2 = a^2+b^2-2ab$
Q3	(x \oplus y) = (x+7)(x+4) - 6k	$(x \oplus y) = (x+7)(x+4)-6k$
Q4	\int \frac\sin xxdx	$\int \frac{\sin x}{x} dx$
Q5	\neg(p \vee q) \vee (\neg p \wedge q) = \neg p	$\neg(p \vee q) \vee (\neg p \wedge q) = \neg p$
Q6	a \equiv b \pmod n	$a \equiv b \pmod n$
Q7	6.7n - 2.3n	$6.7^n - 2.3^n$
Q8	((p \to (q \to r)) \to ((p \to q) \to (p \to r)))	$((p \to (q \to r)) \to ((p \to q) \to (p \to r)))$
Q9	(a2+b2+c2)2 = 2(a4+b4+c4)	$(a^2+b^2+c^2)^2 = 2(a^4+b^4+c^4)$
Q10	(n-m) \mid (nk-mk)	$(n-m) \mid (n^k-m^k)$
Q11	(p \leftrightarrow q)	$(p \leftrightarrow q)$
Q12	(p\vee q) \wedge (p\to r) \wedge (p \to r)\to r	$(p \vee q) \wedge (p \to r) \wedge (p \to r) \to r$
Q13	(p-1)! \equiv -1 (\mod p)	$(p-1)! \equiv -1(\mod p)$
Q14	(x+y)\times\fracab	$(x+y) \times \frac{a}{b}$
Q15	(x2+1)2	$(x^2+1)^2$
Q16	(z + y + x)2	$(z+y+x)^2$
Q17	1+\tan2x	$1+\tan^2 x$
Q18	1+x+x2+x3+\ldots \:	$1+x+x^2+x^3+\ldots$
Q19	1+x+x2+x3=y	$1+x+x^2+x^3=y$
Q20	1.1!+\dots+n.n! = (n+1)!	$1.1!+\cdots+n.n! = (n+1)!$
Q21	10 \sin2(\theta) - 13 \sin(\theta) - 3 = 0	$10\sin^2(\theta)-13\sin(\theta)-3=0$
Q22	2+4+\dots+2n=n(n+1)	$2+4+\cdots+2n=n(n+1)$
Q23	3x2 + 2x	$3x^2+2x$
Q24	7x + 5y = 3 \pmod 4	$7x+5y=3 \pmod 4$
Q25	F_n2-(F_n+1)(F_n-1)=(-1)n-1	$F_n^2-(F_{n+1})(F_{n-1})=(-1)^{n-1}$
Q26	O(n!) \in O(nn̂)	$O(n!) \in O(n^n)$
Q27	T(n) = T(n-1) + T(n-2)	$T(n)=T(n-1)+T(n-2)$
Q28	X2̂ + Y2̂ = Z2̂	$X^2+Y^2=Z^2$
Q29	X2̂ = 10X̂	$X^2=10^X$
Q30	Xn̂ + Yn̂ = Zn̂	$X^n+Y^n=Z^n$
Q31	\arctan(x)=x-\fracx3̂3+\fracx5̂5-\fracx7̂7+\\cdots	$\arctan(x)=x-\frac{x^3}{3}+\frac{x^5}{5}-\frac{x^7}{7}+\ldots$
Q32	\beginvmatrix x & a & b \\ b & x & a \\ a & b & x \endvmatrix	$\begin{vmatrix} x & a & b \\ b & x & a \\ a & b & x \end{vmatrix}$
Q33	\cos x \sin x	$\cos x \sin x$
Q34	\cos(y + x) =	$\cos(y+x)=$
Q35	\exp \sum_t \lambda f(t, Q, t)) \frac1z_\lambda (Q	$\exp \sum_t \lambda f(t,Q,t))\frac{1}{z_\lambda(Q}$
Q36	\frac\tan \fracA-B2 \tan \fracA+B2	$\frac{\tan\frac{A-B}{2}}{\tan\frac{A+B}{2}}$
Q37	\fracddx\sqrt4-x2̂ dx	$\frac{d}{dx}\sqrt{4-x^2}dx$
Q38	\fracx2̂-1x2̂+1-\frac2ixx2̂+1	$\frac{x^2-1}{x^2+1}-\frac{2ix}{x^2+1}$
Q39	\fracx2̂x+1	$\frac{x^2}{x+1}$
Q40	\ \frac(x)(1+sin x)	$\frac{x}{1+\sin x}$
Q41	\int \exp(-x2̂)dx	$\int \exp(-x^2)dx$
Q42	\int \sinx̂\cos2̂x\;dx	$\int \sin^x \cos^2 x\, dx$
Q43	\int e-\fracr2̂kdr	$\int e^{-\frac{r^2}{k}} dr$
Q44	\int e-x2̂dx	$\int e^{-x^2} dx$
Q45	\int x\;dx	$\int x\, dx$
Q46	\int x\;dx = 1	$\int x\, dx = 1$
Q47	\int x2̂ f(x)dx	$\int x^2 f(x)dx$
Q48	\int x3̂(1-x2̂)1̂/2	$\int x^3(1-x^2)^{1/2}$
Q49	\int_ 0 2̂\pi \cos (2x)/\exp (x) dx	$\int_0^{2\pi} \cos(2x)/\exp(x)dx$
Q50	\int_-\infty^\infty \exp -r2̂ dr	$\int_{-\infty}^{\infty} \exp -r^2 dr$
Q51	\int_01̂ x \cdot (1-x)n̂-1 dx	$\int_0^1 x \cdot (1-x)^{n-1}dx$
Q52	\lim_x\to \infty\left(\fracx+1x\right)=1	$\lim_{x\to\infty}\left(\frac{x+1}{x}\right)=1$
Q53	\log (x+3) +\log x	$\log(x+3)+\log x$
Q54	\log\fracz(1-z)2̂	$\log\frac{z}{(1-z)^2}$
Q55	\neg (p \leftrightarrow q) = \neg q \leftrightarrow p	$\neg(p \leftrightarrow q) = \neg q \leftrightarrow p$
Q56	\phi (V1V̂2)	$\phi(V1^{V2})$
Q57	\sin2̂ x + \cos2̂ x	$\sin^2 x+\cos^2 x$
Q58	\sqrt(a2̂ + b2̂)	$\sqrt{(a^2+b^2)}$
Q59	\sqrt2+\sqrt2+\sqrt2	$\sqrt{2+\sqrt{2+\sqrt{2}}}$
Q60	\sqrta2̂ + b2̂	$\sqrt{a^2+b^2}$
Q61	\sqrtx+1	$\sqrt{x+1}$
Q62	\sum \frac1i2̂	$\sum \frac{1}{i^2}$
Q63	\sum _i=0^\inftya_iXî	$\sum_{i=0}^{\infty} a_i X^i$
Q64	\sum_n=1^\infty \\:\frac3n+2n\left(n+1\right)	$\sum_{n=1}^{\infty} \frac{3n+2}{n(n+1)}$
Q65	\sum_i=1n̂ xî	$\sum_{i=1}^n x^{i!}$
Q66	\sum_i=1n̂ik̂	$\sum_{i=1}^n i^k$
Q67	\sum_i=1^\infty P(X>i) ¡ \infty	$\sum_{i=1}^{\infty} P(X>i) < \infty$
Q68	\sum_i=1n̂ 2î	$\sum_{i=1}^n 2^i$
Q69	\sum_i=1n̂ \fracn!i! 2î	$\sum_{i=1}^n \frac{n!}{i!}2^i$
Q70	\sum_j=1n̂ jk̂ n \choose j	$\sum_{j=1}^n j^k \binom{n}{j}$
Q71	\sum_k=05̂k+1	$\sum_{k=0}^5 k+1$
Q72	\sum_m=1n̂ \frace-am2̂m2̂	$\sum_{m=1}^n \frac{e^{-am^2}}{m^2}$
Q73	\sum_n=0 \fracf̂(n)(a)n! (x-a)n̂	$\sum_{n=0} \frac{f^{(n)}(a)}{n!}(x-a)^n$
Q74	\sum_n=0\infty zn!	$\sum_{n=0}^{\infty} z^{n!}$
Q75	\sum_n=0^\infty\frac(-1)n̂x2̂n(2n)!	$\sum_{n=0}^{\infty} \frac{(-1)^n x^{2n}}{(2n)!}$
Q76	\sum_n=1^\infty (1n)	$\sum_{n=1}^{\infty}(1/n)$
Q77	\sum_x=1^\infty \frac1x	$\sum_{x=1}^{\infty} \frac{1}{x}$
Q78	\tan(x) = 5/12	$\tan(x)=5/12$
Q79	an̂+1=bn̂	$a^n+1=b^n$
Q80	ab=c\pmod n	$ab=c \pmod n$
Q81	ax2̂ + bx + c	ax^2+bx+c
Q82	dr_t = a(b-r_t) dt + \sigma dW_ t	$dr_t = a(b-r_t)dt+\sigma dW_t$
Q83	ex̂ > xê	$e^x > x^e$
Q84	eî=\cos \left(x\right)+i\sin \left(x\right)	$e^{ix} = \cos(x)+i\sin(x)$
Q85	eî=\cos \left(φ \right)+i\sin left(φ \right)	$e^{i\varphi} = \cos(\varphi)+i\sin(\varphi)$
Q86	f(x)=\left(1+\frac1x\right)x̂	$f(x)=\left(1+\frac{1}{x}\right)^x$
Q87	f_n+12̂ + f_n2̂ = f_2n+1	$f_{n+1}^2+f_n^2=f_{2n+1}$
Q88	g(n)=o(mn̂)	$g(n)=o(m^n)$
Q89	n!-(n-1)!=(n-1)(n-1)!	$n!-(n-1)!=(n-1)(n-1)!$
Q90	p(xy—z)=p(x—z)p(y—xz)	$p(xy \mid z)=p(x \mid z)p(y \mid xz)$
Q91	poly(n)	$poly(n)$
Q92	x2̂ +1	x^2+1
Q93	x2̂-y2̂6̂	$x^2-y^{2^6}$
Q94	y = -x\sin\theta + y\cos\theta	$y=-x\sin\theta+y\cos\theta$
Q95	z \tanz	$z\tan z$
Q96	n \choose k = 1 + \sum_j=0k̂-1 \sum_i=0n̂-k-1 i+j \choose j	$\binom{n}{k}=1+\sum_{j=0}^{k-1}\sum_{i=0}^{n-k-1}\binom{i+j}{j}$
Q97	γ =\lim _n\to \infty \:\left(1+\frac12+\frac13+\frac14+\ldots \:+\frac1n-\ln \left(n\right)	$\gamma = \lim_{n\to\infty}\left(1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\ldots+\frac{1}{n}-\ln(n)\right).$
Q98	\frac00	$\frac{0}{0}$
Q99	\fracddx[kf (x)] = \fracddx [\phi(x)]	$\frac{d}{dx}[kf(x)]=\frac{d}{dx}[\phi(x)]$
Q100	f: A \to B ,y = f(x)	$f: A \to B, y=f(x)$

AUTHORS PROFILE

SOURISH DHAR working in the capacity of an Assistant Professor in the Department of CSE, Assam University (Central University), Silchar(India) for the last 10 years. He completed his PhD. in 2019, and his research interest includes NLP, Information Retrieval and Data Science..

ABHIJIT BISWAS is currently working as an Assistant Professor in the department of CSE, TSSOT, Assam University since October 2009. He is a alumni of NIT Bhopal and have received his M.Tech and PhD degrees from NERIST in the year 2009 and 2018 respectively. His research interest includes, network routing, Network – On – Chip topologies and Data Science.

Nidhi Singh completed her M.Tech(CSE) from the Department of CSE, Assam University (Central University), Silchar(India) . Her research interest includes NLP, Information Retrieval and Data Science.. She is currently gearing up for her PhD.

Retrieval Number: K12980981119/19©BEIESP
DOI: 10.35940/ijitee.K1298.0981119
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)