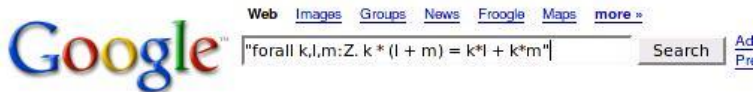


Searching for Distributivity



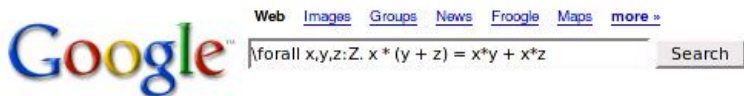
Tip: Try removing quotes from your search to get more results.

Your search - **"forall k,l,m:Z. k * (l + m) = k*l + k*m"** - did not match any documents.

Suggestions:

- ◆ Make sure all words are spelled correctly.
- ◆ Try different keywords.
- ◆ Try more general keywords.

Searching for Distributivity



Searching for Distributivity



Web

[Mathematica - Setting up equations](#)

Try "Reduce" rather than "Solve" and use "ForAll" to put a condition on x , y , and z . In[1]:=

```
Reduce[ForAll[{x, y, z}, 5*x + 6*y + 7*z == a*x + b*y + c*z], ...
```

www.codecomments.com/archive382-2006-4-904844.html - 18k - Supplemental Result -

[Cached](#) - [Similar pages](#)

[\[PDF\] arXiv:nlin.SI/0309017 v1 4 Sep 2003](#)

File Format: PDF/Adobe Acrobat - [View as HTML](#)

7.2 Appendix B. Elliptic constants related to $gl(N, \mathbb{C})$ 1 for all $s \leq j$. (4.14). The first condition means that the traces (4.13) of the Lax operator ...

www.citebase.org/cgi-bin/fulltext?format=application/pdf&identifier=oai:arXiv.org:nlin/0309017 -

Supplemental Result - [Similar pages](#)

[\documentclass{article} \usepackage{axiom} \usepackage{amssymb ...](#)

```
i+1) bz:= (bz - 2**i)::NNI else bz:= bz + 2**i z.bz := z.bz + c z x * y == z ... b,i-1]] be := reduce("...", ml)
```

```
c = 1 => be c::Ex * be coerce(x): Ex == tl ...
```

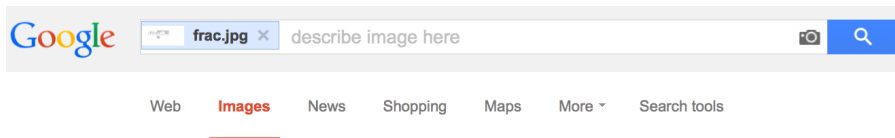
wiki.axiom-developer.org/axiom-test-1/src/algebra/CliffordSpad/src - 20k - Supplemental Result -

[Cached](#) - [Similar pages](#)

Does Image Search help?

- ▶ Math formulae are visual objects, after all

(let's try it)



A small image of the quadratic formula:
$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Image size:
133 × 61

No other sizes of this image found.

Tip: Try entering a descriptive word in the search box.

Your search did not match any documents.

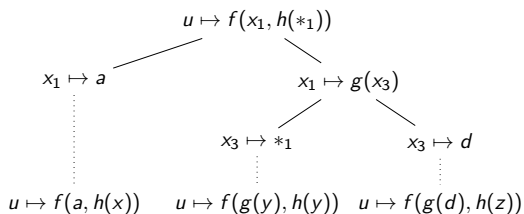
Suggestions:

- Try different keywords.

Of course Google cannot work out of the box

- ▶ **Formulae are not words:**
 - ▶ $a, b, c, k, l, m, x, y,$ and z are (bound) variables. (do not behave like words/symbols)
 - ▶ where are the word boundaries for “bag-of-words” methods?
- ▶ **Formulae are not images either:** They have internal (recursive) structure and compositional meaning
- ▶ **Idea:** Need a special treatment for formulae (translate into “special words”) Indeed this is done ([**MilYou:tadImf02**; **MunMin:MathFind06**; **LibMel:marmca06**; **MisGal:egoMath11**]) ... and works surprisingly well (using e.g. Lucene as an indexing engine)
- ▶ **Idea:** Use database techniques (extract metadata and index it) Indeed this is done for the Coq/HELM corpus ([**AGSTZ:ContMathSearchWhelp04**])
- ▶ **Our Idea:** Use Automated Reasoning Techniques (free term indexing from theorem prover jails)
- ▶ **Demo:** MathWebSearch on Zentralblatt Math, the arXiv Data Set

Substitution Tree [Graf '94]

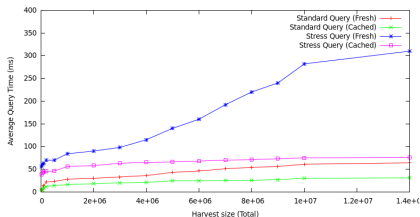


- ▶ Variant of abstraction trees that indexes **Substitutions** (Nodes labeled with **Substitutions**)
- ▶ includes **Variable renaming** ($*_i \hat{=} i^{\text{th}}$ variable)
- ▶ less redundant than abstraction trees
- ▶ allows $n : m$ indexing

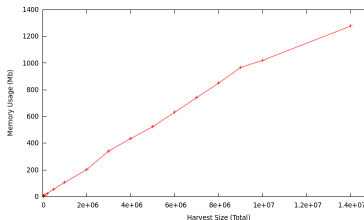
Index statistics

- ▶ **Experiment:** Indexing the arXiv (1M documents, $\sim 10^8$ non-trivial formulae)
- ▶ **Results:** indexing up to 15 M formulae on a standard laptop

Query Times



Memory Footprint



- ▶ query time is constant (~ 15 ms) (as expected; goes green depth \times symbols)
- ▶ memory footprint seems linear ($\sim 500 \frac{B}{\text{formula}}$) (expected more duplicates)
- ▶ So we need ca. 100G B RAM for indexing the whole arXiv.
- ▶ Can index all published Math ($\hat{=} 5 \times$ arXiv) on a large server (.5T B RAM). (ZBL $\hat{=} 3.5$ M art.)

Formula/Text Search Combination?

- ▶ **Observation:** MathWebSearch is similar to a one-word IR algorithm, except ... unification directly matches one search term against lots of search terms.
- ▶ **Idea:** combine unification indexing with the vector space model for a "bag-of-formulae" (instead of standard IR's "bag-of-words") method ...
- ▶ **at Indexing time:** when we index a math document D ,
 - ▶ insert the formulae into the MathWebSearch index (remember dbid)
 - ▶ replace all formulae in D with their dbid to get D'
 - ▶ index D' in a bag-of-words index (e.g. Elastic Search or Terrier)
- ▶ **At query time:** (essentially query expansion)
 - ▶ query Q consists of a set Q_f of formulae and a set Q_w of words.
 - ▶ run Q_f through MathWebSearch to get set I_f of matching dbids.
 - ▶ run $Q' = Q_w + I_f$ through nutch to get a set R of document fragments URIs.
- ▶ we return R together with the fragments of D they point to.
- ▶ we can even inherit the ranking mechanisms from nutch. (see if they help)