Document tag query problem Thoughts

Assuming there is a main variable vault storing list of designed data structure

Basic data structure

Class d – has 2 variables

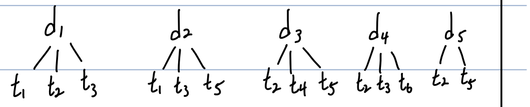
- name – identifier of the document
- ~~context – String, context of the document (use int for simplicity)~~ (not needed)
- tags – [String], tags in string.

Brute force

Let q be the tags list we are searching for

```
result = [] # initialize the result list
for d in documents:
    in_subset = True
    for tag in d.tags:
        if not (tag in q):
            in_subset = False
            break # no point to continue since d needs to have all tags in q
    if in_subset:
        result.append(d) # pass all tag 'tests' which mean this d's tags ⊆ q
return result
```

E.g. : docs = $\{d_1, d_2, d_3, d_4, d_5\}$ tags = $\{t_1, t_2, t_3, t_4, t_5, t_6\}$



$q = \{t_1, t_2, t_3, t_5\}$ $|q| > |d_i\text{'s tags}|$

process:

d_1?	d_2?	d_3?	d_4?	d_5?
$t_1 \subseteq q$? ✓	$t_1 \subseteq q$? ✓	$t_2 \subseteq q$? ✓	$t_2 \subseteq q$? ✓	$t_2 \subseteq q$? ✓
$t_2 \subseteq q$? ✓	$t_3 \subseteq q$? ✓	$t_4 \subseteq q$? X	$t_3 \subseteq q$? ✓	$t_5 \subseteq q$? ✓
$t_3 \subseteq q$? ✓	$t_5 \subseteq q$? ✓		$t_6 \subseteq q$? X	
d_1 ✓	d_2 ✓	d_3 X	d_4 X	d_5 ✓

Ans : $[d_1, d_2, d_5]$

Complexity: $O(n^2)$

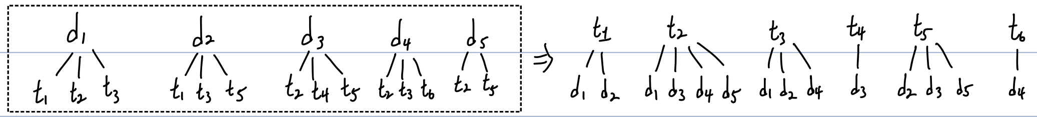
Smarter? (look at the data structure in another way where class is tag) Failed

Class t - has 1 variable • (docs) - set of Class doc, doc has 1 variable • name - identifier of the document

The magic happens during the creating of document (insertion)

Idea: instead of thinking documents with tags attached, it's each tag contain what document

Back to E.g.: docs = $\{d_1, d_2, d_3, d_4, d_5\}$ tags = $\{t_1, t_2, t_3, t_4, t_5, t_6\}$



Then to find $d_?$ with $q = \{t_2, t_3\}$, convert those tags list to tag sets and find intersection of q's elements

$$\Rightarrow t_2 \cap t_3 = (d_1, d_3, d_4, d_5) \cap (d_1, d_2, d_4) = (d_1, d_4)$$

Complexity: $O(n)$ should be Problem: Consume lots of space & unique identifier for documents required

This solution is assuming $|q| < |d_i\text{'s tags}|$ which is not the problem suggests in the first place

(misread the problem ... oops)

Smarter 2? (using the fact of prime factorization ???) elementary number theory Worked

Idea : $\overset{\text{using}}{P_i}$ - primes (e.g. 2,3,5, 7, 11, 13, ...) $V_p(a)$ - highest power of p that divides a

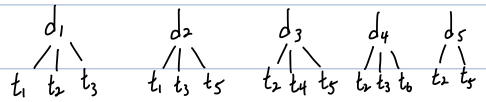
LCM of a & b = $\prod_p p^{\max(V_p(a), V_p(b))}$ or if a & b are primes, LCM of a & b is ab Since $[a,b] = \prod_p p^{\max(V_p(a), V_p(b))} = a^1 \cdot b^1 = ab$

the other primes has v(p) is 0 ⇒ $p^0 = 1$

Let say 6, 2|6, 3|6 , or 12 = 1 ,$\overset{\text{divisors}}{2}$,3,4,6,12 = $2^2 \times 3$ Hmm, find the prime factorization of a number or using sieve theory $O(n \log(\log n))$???

So, consider each tag is a prime number, then each document is a product of those primes

E.g. : docs = $\{d_1, d_2, d_3, d_4, d_5\}$ tags = $\{t_1, t_2, t_3, t_4, t_5, t_6\}$
2 3 5 7 11 13



⇒ $d_1 = 2 \times 3 \times 5 = 30$ $d_4 = 3 \times 5 \times 13 = 195$

$d_2 = 2 \times 5 \times 11 = 110$ $d_5 = 3 \times 11 = 33$

$d_3 = 3 \times 7 \times 11 = 232$

$q = \{t_1, t_2, t_3, t_5\}$ $|q| > |d_i\text{'s tags}|$

$q = 2 \times 3 \times 5 \times 11 = 330$

$d_1 | q$ $d_2 | q$ $d_3 \nmid q$ $d_4 \nmid 195$ $d_5 | q$

⇒ ans = $[d_1, d_2, d_5]$

Idea : Every tag is a prime number, when inserting a document, compute & insert LCM of its tags
later on just see which doc is divisible by LCM of q

Complexity : $O(n)$ less space needed ✓