

## 华东师范大学数据学院上机实践报告

课程名称：信息检索

年级：2018

上机实践成绩：

指导教师：张蓉

姓名：孙秋实

上机实践名称：索引与查询

学号：10185501402

上机实践日期：2021/10/21

### Part 1

#### 实验目的

- (1) 为文档构建倒排索引
- (2) 设计搜索系统，实现布尔查询

### Part 2

#### 实验任务

- (1) 使用一般方法为文档数据集构建索引
- (2) 在单机伪分布式下使用 Spark 为文档数据集构建倒排索引
- (2) 完成对已构建索引的文档的 OR 与 AND 两种查询方式
- (3) 对结果进行分析和展示

### Part 3

#### 使用环境

- (1) Google Colab
- (2) Python 3.7
- (3) Spark 3.2.0
- (4) Scala 2.12
- (5) Idea 2020.02.03

### Part 4

#### 实验过程

### Section 1

#### 倒排索引简介

倒排索引是一种索引方法，被用来存储在全文搜索下某个单词在一个文档或者一组文档中的存储位置的映射，是检索系统中用于实现“单词-文档矩阵”的一种具体存储形式。

以 Wikipedia 上的一个例子进行简单说明

假设有以下英文文本

- $T_0 = 0$  'it is what it is'
- $T_1 = 1$  'what is it'
- $T_2 = 2$  'it is a banana'

构建索引，即建立单词与文档的关系

- 'a': 2
- 'banana': 2
- 'is': 0, 1, 2
- 'it': 0, 1, 2
- 'what': 0, 1

在检索时候，使用'and' 或'or' 逻辑进行布尔查询，如：

'what', 'is', 'it' 将对应集合:  $\{0, 1\} \cap \{0, 1, 2\} \cap \{0, 1, 2\} = \{0, 1\}$

存储时，形式可以是 Posting List，如图 1 所示

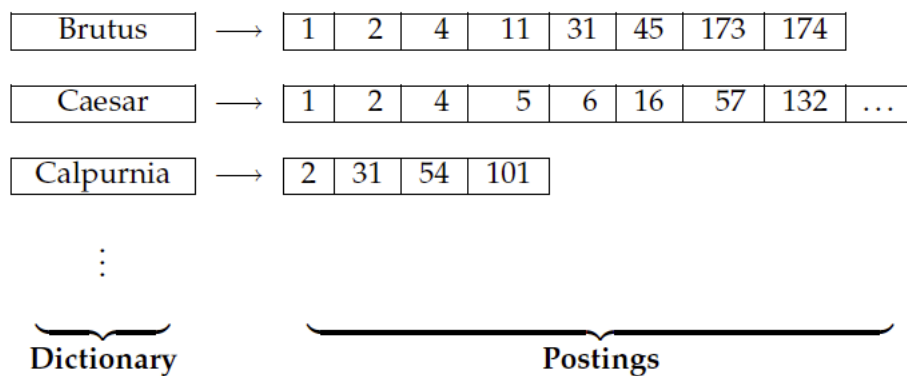


图 1: Posting List

检索时，相当于在使用共现矩阵，如图 2 所示

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
Antony	1	1	0	0	0	1	
Brutus	1	1	0	1	0	0	
Caesar	1	1	0	1	1	1	
Calpurnia	0	1	0	0	0	0	
Cleopatra	1	0	0	0	0	0	
mercy	1	0	1	1	1	1	
worser	1	0	1	1	1	0	
...							

图 2: A term-document incidence matrix

\* 上述图例来自 *Introduction to Information Retrieval*

**Section 2**

## 倒排索引算法的实现 (内存)

首先是算法核心部分

```
if doc_term != dic[0]:
    Document_id = sorted(Document_id, key=lambda s: int(s))
    len_id = len(Document_id)
    ans_temp.append((doc_term, len_id, Document_id))
    # add doc term

    doc_term = dic[0]
    # use array
    Document_id = [dic[1]]
else:
    if Document_id[-1] != dic[1]:
        Document_id.append(dic[1])
```

注意，建立好的索引需要进行排序

```
ascending_ans = sorted(ans_temp,
                        key=lambda s: int(s[1]),
                        reverse=False)

# 升序or降序
descending_ans = sorted(ans_temp,
                        key=lambda s: int(s[1]),
                        reverse=True)
```

随后，需要去除索引中大量重复出现的词（阈值为 100）

```
exceed_count=0
for term in ascending_ans:
    if (len(term[2])>100): # split
        exceed_count+=1
        print('词项: '+str(term[0]), '出现次数: '+str(len(term[2])))

print('共'+str(exceed_count)+'个词项出现超过100次')
```

**Section 3**

## 倒排索引算法的实现 (Spark)

接下来使用 Scala 语言完成基于 Spark 的倒排索引构建，其逻辑与一般方法类似，**只是以 RDD 变换的形式进行构建而已**，但是需要注意 Spark 版本与 Idea 提供的 Scala 版本是否对应。

\* 本次试验使用的是 *Spark3.2.0 + Scala2.12*

```
object ReadGetIndex {
// val writer = new PrintWriter(new File("spark_index_test.txt" ))
def main(args: Array[String]): Unit = {
```

```
//get Spark Session
val spark = SparkSession.builder().appName("Information-Retrieval-Project2")
    .master("local")
    .getOrCreate()
//Get Dir
val document_source = spark.sparkContext.wholeTextFiles("src/data-2500")
//Get File Name
val res = document_source.flatMap{ x =>
    val doc = x._1
    .split(s"/")
    .last // get file name.txt
    .split("\\.")
    .head // get file name
    x._2.split("\r\n") //seperate -> line -> blank
    .flatMap(_.split(" ").map{ y => (y, doc)})}
    .groupByKey.map{case(x,y)=>(x,y.toSet.mkString(","))} // mkstring即用指定字符串分隔
// pattern matching
// 此处需要注意partition!
res.coalesce(1).saveAsTextFile("spark_index_test.txt")
res.foreach(println)
}
}
```

简单起见，展示一下运行结果

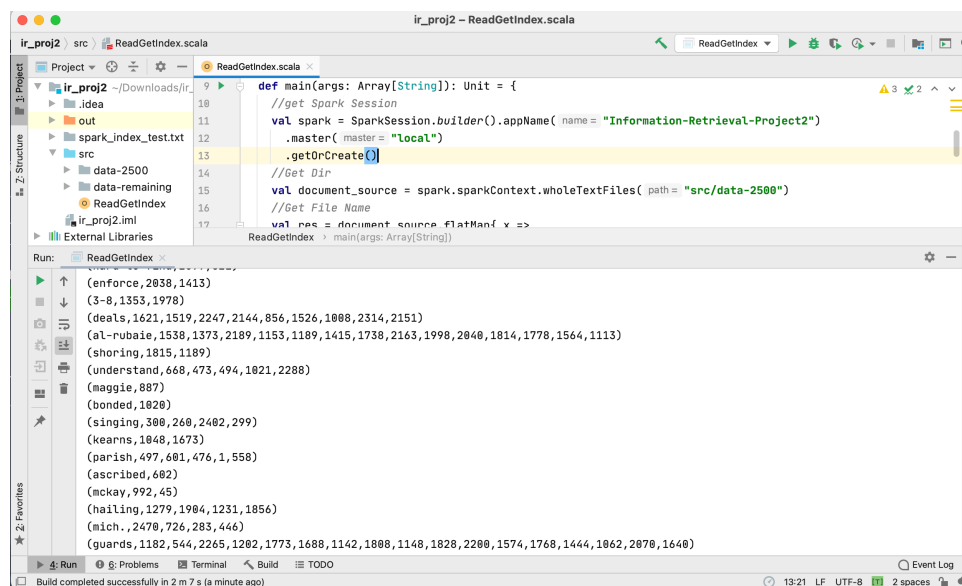


图 3: Spark 建立倒排索引

随后将其存为.txt 格式即可。

## Section 4

### 搜索系统的构建

需要对错误的输入进行处理，即

- 输入的逻辑词错误
- 输入的 query 长度错误

所以构建以下函数

---

```
def not_available_check(query):  
    if len(query) > 3 or len(query)==2 or (query[1] != 'or' and query[1] != 'and'):  
        print("Illegal Command")  
        print('---NEXT---')  
        return True # false query
```

---

其次，搜索的内容可能在文档中没有出现过，也需要进行如下处理

---

```
def not_in_dic(query_1,query_2,dic):  
    if query_1 not in dic or query_2 not in dic:  
        print("DOC NOT Found")  
        return True
```

---

先进行单关键词查询

---

```
def search_single_key(query,dic):  
    if (query[0] not in dic):  
        print("\nNOT Found")  
        continue  
    for i in dic[query[0]][1]:  
        print(i, end=" ")  
    print('\n---NEXT---')
```

---

随后分别考虑不同关键词下的查询，首先以”and”查询为例  
简而言之，文档编号不同，则移动当前位置较小的指针

---

```
def search_and(query1, query2):  
    and_search_ans = []  
    i,j=0,0  
    lth1 = len(query1)  
    lth2 = len(query2)  
    # print(lst1,lst2)  
    while i < lth1 and j < lth2:  
        if (query1[i] < query2[j]):  
            i += 1  
        elif (query1[i] > query2[j]):  
            j += 1  
        else:  
            and_search_ans.append(query1[i]) #lst1=lst2  
            i,j = i+1,j+1  
    return and_search_ans
```

---

其次是”or”查询，这种情况下需要考虑到查询文本的长度

---

---

```
def search_union_equal(query1, query2):
    answer = [] # 或者用set
    i,j=0,0
    lth1 = len(query1) # calc length
    lth2 = len(query2)
    if (i == lth1):
        answer.append(query2[j])
        j += 1
        continue
    if (j == lth2):
        answer.append(query1[i])
        i += 1
        continue
```

---

最后，使用连续查询相当于起一个死循环（加上适当的输出 break），实现逻辑如下

---

```
def exit_search(query):
    if query == '' or query == "stop":
        return True
```

---

主循环：

---

```
while True:
    query = input('Input your query:').split()
    # case 1 只输入了一个单词
    if len(query) == 1:
        search_single_key(query,dic)
    # 错误
    elif exit_search(query):
        break
    # case2:
    # OR查询或AND查询
    else:
        if (not_available_check(query)):
            continue
        # check format
        else:
            if not_in_dic(query[0],query[2],dic):
                continue
            # return to the start of the loop
            l1 = dic[query[0]][1]
            l2 = dic[query[2]][1]
            if query[1] == 'and':
                # or 查询
                for i in (search_and(l1, l2)):
                    print(i, end=" ")
                print('\n---NEXT---')
```

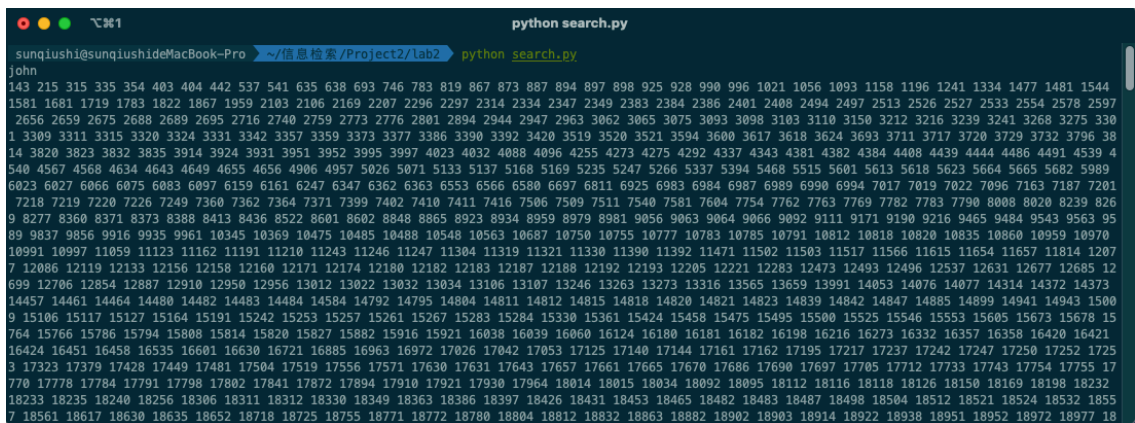
```
# 依然是分类讨论
elif len(lst1) == len(lst2):
    for i in (search_union_equal(l1, l2)):
        print(str(i), end=" ")
    print('\n---NEXT---')
else:
    for i in (search_union_not_equal(l1, l2)):
        print(str(i), end=" ")
    print('\n---NEXT---')
```

## Section 5

### 查询演示

在此部分做查询的演示

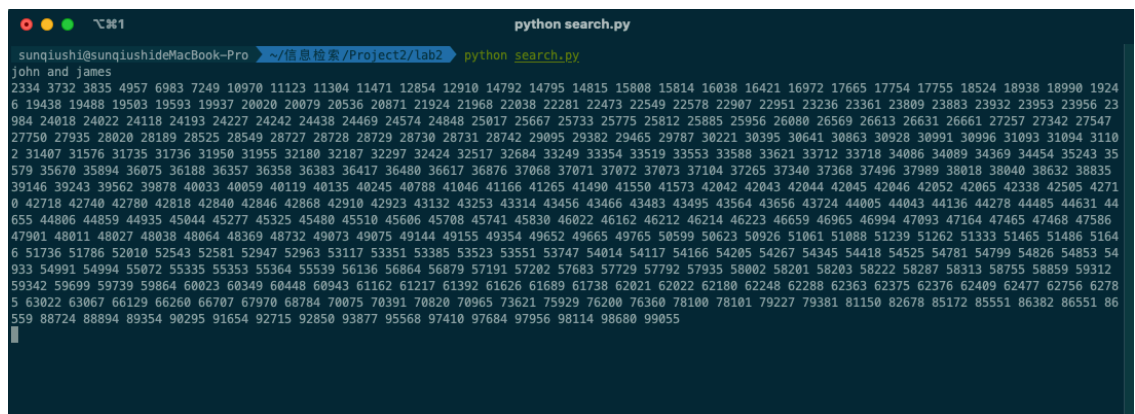
- 实现单个关键词查询



```
python search.py
john
143 215 315 335 354 403 404 442 537 541 635 638 693 746 783 819 867 873 887 894 897 898 925 928 990 996 1021 1056 1093 1158 1196 1241 1334 1477 1481 1544
1581 1681 1719 1783 1822 1867 1959 2103 2106 2169 2207 2296 2297 2314 2334 2347 2349 2383 2384 2386 2401 2408 2494 2497 2513 2526 2527 2533 2554 2578 2597
2656 2659 2675 2688 2689 2695 2716 2740 2759 2773 2776 2801 2894 2944 2947 2963 3062 3065 3075 3093 3098 3103 3110 3150 3212 3216 3239 3241 3268 3275 330
1 3309 3311 3315 3320 3324 3331 3342 3357 3359 3373 3377 3386 3390 3392 3420 3519 3520 3521 3594 3600 3617 3618 3624 3693 3711 3717 3720 3729 3732 3796 38
14 3820 3823 3832 3835 3914 3924 3931 3951 3952 3995 3997 4023 4032 4088 4096 4255 4273 4275 4292 4337 4343 4381 4382 4384 4408 4439 4444 4486 4491 4539 4
540 4567 4568 4634 4643 4649 4655 4656 4906 4957 5026 5071 5133 5137 5168 5169 5235 5247 5266 5337 5394 5468 5515 5601 5613 5618 5623 5664 5665 5682 5989
6023 6027 6066 6075 6083 6097 6159 6161 6247 6347 6362 6363 6553 6566 6580 6697 6811 6925 6983 6984 6987 6989 6990 6994 7017 7019 7022 7096 7163 7187 7201
7218 7219 7220 7226 7249 7360 7362 7364 7371 7399 7402 7410 7411 7416 7506 7509 7511 7540 7581 7604 7754 7762 7763 7769 7782 7783 7790 8008 8020 8239 826
9 8277 8360 8371 8373 8388 8413 8436 8522 8601 8602 8848 8865 8923 8934 8959 8979 8981 9056 9063 9064 9066 9092 9111 9171 9190 9216 9465 9484 9543 9563 95
89 9837 9856 9916 9935 9961 10345 10369 10475 10485 10488 10548 10563 10687 10750 10755 10777 10783 10785 10791 10812 10818 10820 10835 10860 10959 10970
10991 10997 11059 11123 11162 11191 11210 11243 11246 11247 11304 11319 11321 11330 11390 11392 11471 11502 11503 11517 11566 11615 11654 11657 11814 1207
7 12086 12119 12133 12156 12158 12160 12171 12174 12180 12182 12183 12187 12188 12192 12193 12205 12221 12283 12473 12493 12496 12537 12631 12677 12685 12
699 12706 12854 12887 12910 12950 12956 13012 13022 13032 13034 13106 13107 13246 13263 13273 13316 13565 13659 13991 14053 14076 14077 14314 14372 14373
14457 14461 14464 14480 14482 14483 14484 14584 14792 14795 14804 14811 14812 14815 14818 14820 14821 14823 14839 14842 14847 14885 14899 14941 14943 1500
9 15106 15117 15127 15164 15191 15242 15253 15257 15261 15267 15283 15284 15330 15361 15424 15458 15475 15495 15500 15525 15546 15553 15605 15673 15678 15
764 15766 15786 15794 15808 15814 15820 15827 15882 15916 15921 16038 16039 16060 16124 16180 16181 16182 16198 16216 16273 16332 16357 16358 16420 16421
16424 16451 16458 16535 16601 16630 16721 16885 16963 16972 17026 17042 17053 17125 17140 17144 17161 17162 17195 17217 17237 17242 17247 17250 17252 1725
3 17323 17379 17428 17449 17481 17504 17519 17556 17571 17630 17631 17643 17657 17661 17665 17670 17686 17690 17697 17705 17712 17733 17743 17754 17755 17
770 17778 17784 17791 17798 17802 17841 17872 17894 17910 17921 17930 17964 18014 18015 18034 18092 18095 18112 18116 18118 18126 18150 18169 18198 18232
18233 18235 18240 18256 18306 18311 18312 18330 18349 18363 18386 18397 18426 18431 18453 18465 18482 18483 18487 18498 18504 18512 18521 18524 18532 1855
7 18561 18617 18630 18635 18652 18718 18725 18755 18771 18772 18780 18804 18812 18832 18863 18882 18902 18903 18914 18922 18938 18951 18952 18972 18977 18
```

图 4: 单关键词查询

- 实现”and” 查询



```
python search.py
john and james
2334 3732 3835 4957 6983 7249 10970 11123 11304 11471 12854 12910 14792 14795 14815 15808 15814 16038 16421 16972 17665 17754 17755 18524 18938 18990 1924
6 19438 19488 19503 19593 19937 20020 20079 20536 20871 21924 21968 22038 22281 22473 22549 22578 22907 22951 23236 23361 23809 23883 23932 23953 23956 23
984 24018 24022 24118 24193 24227 24242 24438 24469 24574 24848 25017 25667 25733 25775 25812 25885 25956 26080 26569 26613 26631 26661 27257 27342 27547
27750 27935 28020 28189 28525 28549 28727 28728 28729 28730 28731 28742 29095 29382 29465 29787 30221 30395 30641 30863 30928 30991 30996 31093 31094 3110
2 31407 31576 31735 31736 31950 31955 32180 32187 32297 32424 32517 32684 33249 33354 33519 33553 33588 33621 33712 33718 34086 34089 34369 34454 35243 35
579 35670 35894 36075 36188 36357 36358 36383 36417 36480 36617 36876 37068 37071 37072 37073 37104 37265 37340 37368 37496 37989 38018 38040 38632 38835
39146 39243 39562 39878 40033 40050 40119 40135 40245 40788 41046 41166 41265 41490 41550 41573 42042 42043 42044 42045 42065 42338 42505 4271
0 42718 42740 42780 42818 42840 42846 42868 42910 42923 43132 43253 43314 43456 43466 43483 43495 43564 43656 43724 44005 44043 44136 44270 44485 44631 44
655 44806 44859 44935 45044 45277 45325 45480 45510 45606 45708 45741 45830 46022 46162 46212 46214 46223 46659 46965 46994 47093 47164 47465 47468 47586
47901 48011 48027 48038 48064 48369 48732 49073 49075 49144 49155 49354 49652 49665 49765 50599 50623 50926 51061 51088 51239 51262 51333 51465 51486 5164
6 51736 51786 52010 52543 52581 52947 52963 53117 53351 53385 53523 53551 53747 54014 54117 54166 54205 54267 54345 54418 54525 54781 54799 54826 54853 54
933 54991 54994 55072 55335 55353 55364 55539 56136 56864 56879 57191 57202 57683 57729 57792 57935 58002 58201 58203 58222 58287 58313 58755 58859 59312
59342 59699 59739 59864 60023 60349 60448 60943 61162 61217 61392 61626 61689 61738 62021 62022 62180 62248 62288 62363 62375 62376 62409 62477 62756 6278
5 63022 63067 66129 66260 66707 67970 68784 70075 70391 70820 70965 73621 75929 76200 76360 78100 78101 79227 79381 81150 82678 85172 85551 86382 86551 86
559 88724 88894 89354 90295 91654 92715 92850 93877 95568 97410 97684 97956 98114 98680 99055
```

图 5: 双关键词”and” 查询

可以进行连续查询，接下来执行”or” 查询

## 华东师范大学数据科学与工程学院学生实验报告

```
python search.py

john and james
2334 3732 3835 4957 6983 7249 10970 11123 11304 11471 12854 12910 14792 14795 14815 15808 15814 16038 16421 16972 17665 17754 17755 18524 18938 18990 19246 19438 1
9488 19503 19593 19937 20020 20079 20536 20871 21924 21968 22038 22281 22473 22549 22578 22907 22951 23236 23361 23809 23883 23932 23953 23956 23984 24018 24022 24
118 24193 24227 24242 24438 24469 24574 24848 25017 25667 25733 25775 25812 25885 25956 26080 26569 26613 26631 26661 27257 27342 27547 27750 27935 28020 28189 285
25 28549 28727 28728 28730 28731 28742 29095 29382 29465 29787 30221 30395 30641 30863 30928 30991 30996 31093 31094 31102 31407 31576 31735 31736 31950 3195
5 32180 32187 32297 32424 32517 32684 33249 33354 33519 33553 33588 33621 33712 33718 34086 34089 34369 34454 35243 35579 35670 35894 36075 36188 36357 36358 36383
36417 36480 36617 36876 37068 37071 37072 37073 37104 37265 37340 37368 37496 37989 38018 38040 38632 38835 39146 39243 39562 39878 40033 40059 40119 40135 40245
40788 41046 41166 41265 41490 41550 41573 42042 42043 42044 42045 42046 42052 42065 42338 42505 42710 42718 42740 42780 42818 42840 42846 42868 42910 42923 43132 4
3253 43314 43456 43466 43483 43495 43564 43656 43724 44005 44043 44136 44278 44485 44631 44655 44806 44859 44935 45044 45277 45325 45480 45510 45606 45708 45741 45
830 46022 46162 46212 46214 46223 46559 46965 46994 47093 47164 47465 47468 47586 47901 48011 48027 48038 48064 48369 48732 49073 49075 49144 49155 49354 49652 496
65 49765 50599 50623 50926 51061 51088 51220 51262 51333 51465 51486 51646 51786 51786 52010 52543 52581 52947 52963 53117 53351 53385 53523 53551 53747 54014 5411
7 54166 54285 54287 54345 54418 54525 54781 54799 54826 54853 54923 54991 54994 55072 55335 55353 55364 55539 56136 56864 56879 57191 57202 57683 57729 57792 57935
58002 58201 58283 58222 58287 58313 58755 58859 59312 59342 59699 59739 59864 60023 60349 60448 60943 61162 61217 61392 61626 61689 61738 62021 62022 62180 62248
62288 62363 62375 62376 62409 62477 62756 62785 63022 63067 66129 66260 66707 67970 68784 70075 70391 70820 70965 73621 75929 76200 76360 78100 78101 79227 79381 8
1150 82678 85172 85551 86382 86551 86559 88724 88894 89354 90295 91654 92715 92850 93877 95568 97410 97684 97956 98114 98680 99055

john or james
143 169 170 215 315 335 354 403 404 433 442 477 498 537 541 598 600 602 635 638 693 746 783 819 821 858 860 862 867 869 873 887 894 897 898 925 928 931 990 996 100
8 1021 1056 1093 1106 1131 1152 1158 1196 1216 1217 1241 1261 1334 1420 1477 1481 1539 1544 1546 1581 1681 1719 1732 1756 1778 1783 1822 1842 1867 1886 1959 2046 2
103 2106 2165 2169 2171 2207 2296 2297 2298 2300 2302 2304 2314 2334 2347 2349 2393 2384 2386 2387 2398 2401 2408 2448 2459 2470 2483 2494 2497 2513 2526 2527 2530
2533 2540 2554 2578 2597 2610 2621 2632 2645 2656 2659 2675 2688 2689 2692 2695 2702 2716 2740 2759 2773 2776 2801 2894 2940 2944 2947 2963 2992 3060 3062 3065 30
67 3075 3093 3098 3103 3110 3150 3212 3216 3239 3241 3268 3275 3301 3309 3311 3315 3318 3320 3324 3331 3342 3357 3359 3373 3377 3386 3390 3392 3420 3449 3519 3520
3521 3564 3566 3594 3600 3617 3618 3624 3693 3711 3717 3728 3729 3732 3768 3796 3814 3820 3823 3832 3835 3871 3914 3924 3931 3951 3952 3969 3988 3995 3997 4023 403
2 4083 4088 4086 4123 4135 4204 4255 4273 4275 4292 4327 4337 4343 4381 4382 4384 4408 4439 4444 4486 4491 4539 4540 4567 4568 4622 4634 4643 4649 4655 4656 4673 4
716 4721 4724 4737 4779 4822 4834 4839 4893 4906 4956 4957 5026 5033 5037 5040 5053 5071 5101 5133 5137 5168 5169 5235 5247 5266 5337 5342 5393 5394 5468 5515 5584
5587 5601 5613 5618 5623 5664 5665 5680 5682 5989 5995 5999 6023 6027 6066 6075 6083 6097 6159 6161 6171 6247 6278 6347 6362 6363 6371 6409 6448 6452 6494 6553 65
66 6580 6590 6697 6704 6804 6811 6818 6918 6925 6932 6983 6984 6987 6989 6990 6994 7006 7017 7019 7022 7096 7115 7143 7163 7171 7184 7187 7201 7209 7218 7219 7220
7226 7247 7249 7356 7360 7362 7364 7371 7399 7402 7410 7411 7416 7428 7463 7464 7465 7506 7509 7511 7540 7581 7604 7657 7754 7762 7763 7769 7782 7783 7790 7936 797
2 8006 8008 8020 8028 8037 8221 8239 8269 8277 8307 8316 8360 8371 8373 8388 8413 8436 8473 8522 8523 8601 8602 8848 8865 8923 8926 8934 8959 8979 8981 9016 9056 9
063 9064 9066 9092 9111 9138 9171 9190 9216 9465 9484 9511 9543 9563 9589 9837 9856 9883 9916 9935 9961 10275 10315 10345 10369 10475 10485 10488 10548 10563 10687
10750 10754 10755 10760 10771 10777 10783 10785 10790 10791 10796 10807 10812 10818 10820 10835 10860 10959 10970 10991 10997 11059 11123 11162 11191 11210 11243
11246 11247 11304 11319 11321 11330 11390 11392 11471 11502 11503 11517 11566 11615 11654 11656 11657 11814 12025 12077 12086 12119 12133 12156 12158 12160 12171 1
2174 12180 12182 12183 12187 12188 12192 12193 12205 12209 12221 12242 12283 12448 12473 12493 12496 12537 12631 12677 12685 12699 12706 12854 12887 12892 12905 12
910 12950 12956 13012 13022 13032 13034 13106 13107 13164 13180 13246 13263 13273 13316 13331 13371 13380 13435 13439 13461 13478 13487 13494 13561 13565 13659 137
27 13728 13748 13799 13925 13991 14053 14076 14077 14158 14185 14210 14314 14372 14373 14420 14424 14457 14461 14464 14480 14482 14483 14484 14488 14534 14584 1464
6 14663 14792 14795 14804 14811 14812 14815 14818 14820 14821 14823 14839 14842 14847 14885 14886 14899 14922 14941 14943 14959 14999 15009 15011 15106 15117 15127
15138 15164 15191 15210 15216 15210 15223 15242 15253 15257 15261 15267 15283 15284 15304 15304 15330 15332 15361 15424 15458 15495 15500 15535 15546 15553 15605
15672 15678 15720 15764 15766 15786 15794 15800 15814 15820 15827 15830 15837 15882 15916 15921 16038 16039 16060 16111 16112 16124 16149 16180 16181 16182 16198 1
6216 16273 16392 16357 16398 16417 16420 16421 16424 16425 16433 16451 16458 16535 16536 16557 16601 16630 16634 16672 16721 16868 16885 16963 16972 17026 17042 17
053 17125 17140 17144 17161 17162 17195 17217 17234 17237 17242 17247 17250 17252 17253 17315 17323 17379 17410 17426 17428 17449 17481 17482 17504 17519 17549 175
50 17556 17564 17571 17591 17625 17631 17637 17643 17657 17661 17665 17668 17670 17686 17690 17697 17705 17712 17727 17728 17733 17743 17745 17754 17755 1776
2 17764 17770 17778 17784 17791 17798 17802 17841 17848 17868 17870 17872 17894 17910 17921 17930 17934 17961 17964 17968 18006 18014 18015 18034 18064 18092 18094
18095 18106 18107 18112 18116 18118 18120 18126 18146 18150 18153 18169 18198 18204 18206 18232 18233 18235 18240 18256 18306 18311 18312 18330 18337 18349 18363
```

图 6: 连续查询与”or” 查询