HW 布尔检索

BSBI索引构建

目录一（块一）： [文档一：你 好]

目录二（块二）： [文档二：你 好 再 见]

BSBI算法步骤一：

首先创建两个IdMap类（等同于字典），self.term_id_map， self.doc_id_map 得到词到id，文档到id的映射关系。

self.term_id_map ---> {'你':1,'好':2,'再':3,'见':4}

self.doc_id_map ---> {'文档一':1,'文档二':2}

BSBI算法步骤二：

得到全局的映射关系后，开始分块处理，使用BSBIIndex类下的**parse_block**方法得到termID-docID对。 然后为每个块创建小的索引，使用BSBIIndex类下的**invert_write**方法由termID-docID对构建倒排表。

首先，我们处理块一中的文档，读取文档一，可以得到term-doc对--->[(1,1),(2,1)]

归纳为不同term的posting_list---> 1:[1], 2:[1]

处理块二中的文档，读取文档二，可以得到term-doc对--->[(1,2),(2,2),(3,2),(4,2)]

归纳为不同term的posting_list---> 1:[2], 2:[2]， 3:[2], 4:[2]

使用**InvertedIndexWriter**类的append方法记录postings_dict，并将posting_list写入磁盘。

BSBI算法步骤三：

合并：

索引一： 1:[1], 2:[1]

索引二： 1:[2], 2:[2]， 3:[2], 4:[2]

合并---> 1:[1,2] 2:[1,2] 3:[2] 4:[2]

联合查询

现在我们有以下内容：

self.term_id_map ---> {'你':1,'好':2,'再':3,'见':4}

self.doc_id_map ---> {'文档一':1,'文档二':2}

Index---> 1:[1,2] 2:[1,2] 3:[2] 4:[2]

postings_dict---> {termId:(start_position_in_index_file, number_of_postings_in_list, length_in_bytes_of_postings_list)}

目录一（块一）： [文档一：你 好]

目录二（块二）： [文档二：你 好 再 见]

对于联合查询 （你 见） 我们得到查询词的ID分别为 self.term_id_map['你']=1 self.term_id_map['见']=4

通过postings_dict找到记录在索引中的位置三元组信息

三元组1=postings_dict[1] 三元组2=postings_dict[4]

根据三元组信息找到Index中的记录 1:[1,2] 4:[2]

求交集，最终得到文档集合[2]，对应文档二，返回结果。