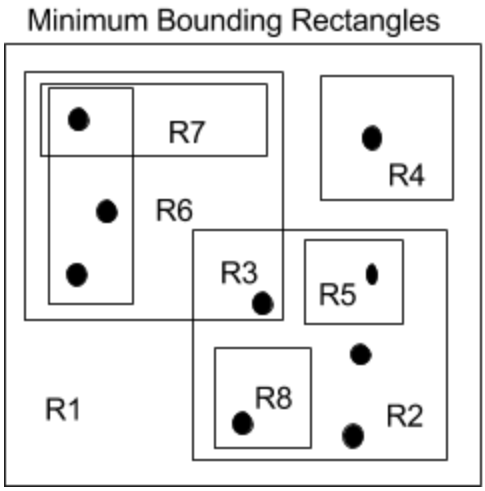
Avneet Kaur -2014027

1) Spatial Inverted Index

Inverted Indexes can only be used to store and retrieve textual documents. Certain situations arise when there is a need to store the spatial locality of a document as well because the queries can be both spatial+textual. For example, in case of an emergency, we might not only want to know the nearest hospital(spatial) but also whether there are icu's available, ambulance available, facilities etc (textual query). This calls for the need of a spatial inverted index. We use the idea of nearest neighbour search retrieval where as soon as we find k nearest neighbour pertaining to k relevant documents, we stop our search. Also, we later examine the problem of examining what steps to do first? Calculate distances and then merge postings or the other way round?

Our main approach consists of a spatial inverted index, consisting of an **R-Tree** built upon every **inverted list** of a particular document. It is based on the idea of Information Retrieval R-Trees and **Signature files** used to create a **Spatial Inverted Index**. It uses the concepts of both. Each point is considered as a location in 2-D space and each point is related to a document containing various keywords.First let us define a few terms. We consider the location associated with a document to be a point in 2-D space, and this point is mapped to a document. So, basically we have many such points, each mapping to documents. Then we define the query point to be consisting of document containing the keywords that we need to look for.



The first part of our Spatial inverted index consists of signature files which are nothing but words which have a particular hash encoded string of a particular length. So a particular document containing many words basically can be simply encoded by taking the conjunction of the encoding of the words contained within the document. If we find relevant documents for the query document then we check whether that the encoded query hash-string is present within the document strings or not. This is very essential part of the containment test, that is , if the query string is absent within a particular document then for sure, the query is not present, relevant documents cannot be found. If on the other hand, the any document contains the query string

then we very well need to check for false positives. Finally, an R-Tree is built for each inverted list.
The second part of our SI Index contains the idea of Building R-Trees. The main intuition behind using R-Trees is the concept of the Minimum bounding rectangles. Minimum Bounding regions are basically regions containing other MBR's further containing spatial data points corresponding to documents. Now the R-Trees in upper levels store the minimum bounding regions and the pointer to their constituent MBR's while at the leaf levels , the R-Trees store the actual point objects. So one point is contained within a particular BR, which may be contained within a bigger MBR, which further may be contained within another MBR and so on till we reach the root node. Each document may be contained within one or more MBR's. So basically if we define a query q and it is not contained within a particular MBR, then we most definitely do not need to check the documents contained within that particular MBR. If however , in another case, there is a case of overlapping MBR's, then we need to check both the MBR's for that particular query.
Thirdly we consider on what basis are we going to search first? That is are we going to calculate the query distance first and then merge the postings list for finding relevant documents after looking up the search tree or are we first going to look up the search tree for relevant document and merge the postings and then calculate the distances for finding the nearest neighbours? For this the answer is simple, we use the idea of distance computation before merging the postings, as this has the advantage of only returning the relevant documents that are nearby the query string, which is what we finally want. Then we find the relevance according to merging the postings list accordingly based on our retrieval from the R-Tree and the signature files combined as described above. This reduces our search space.

Extra Credit Answer :
Using the system described above, we can have one document tagged with multiple locations. This is easily established by the fact that one document can be represented by different points in 2-D space. The contents of the document remain exactly the same only its spatial orientation changes. Consider 2 points A and B having coordinates (i,j) and (k,l). They can essentially represent the same document. Basically if we map it to the R-Tree, each of the points A and B can be constituents of either different Minimum Bounding Rectangles or same Minimum Bounding Rectangles. In either case, their spatial locality will be different therefore implying that they will be completely different leaf nodes even though they are representatives of exactly the same document. So, in either case, the query document can lie in the MBR of one or two or both, based on similarity of context or content (keywords in query and document) but both points would be separate leaf nodes as there x and y coordinates are different in 2-D space, therefore we find the relevant document by computing distance measures like the euclidean distance, based on closeness we rank the relevant documents accordingly.

Q2) a) Precision:0.632 Recall:0.886 NDCG: 0.722. Only answers which have a contextual basis were used. One word answers, yes, no, yeah, and null answers were removed.

b) Precision: 0.723 Recall:0.811 NDCG: 0.832. Tf-idf method was used was used to get top K results and rank them according to cosine similarity measure.

c) Yes, the answers do verify, but not all, in certain cases when there are descriptive answers pertaining to a particular question, and there are more than one answers of a particular type on the same topic, then, sometimes, the answers are interchanged or that document is ranked highly which has a more descriptive answer than the other.