Abordagem

Foram abordadas duas soluções para a implementação do indice invertido, tabela hash com lista ligada e trie aninhada.

Implementação com tabela Hash

Utilizando uma tabela hash cujas listas de ocorrências para cada palavra mapeada na hash é implementada por meio de uma lista duplamente encadeada foi implementado o índice invertido. Foi por tando implementada uma hash table baseada em um vetor de tamanho fixo e a função de hashing escolhida. As chaves da tabela hash foram as palavras indexadas, para cada palavra foi associada a esta entrada uma lista duplamente ligada que contém as ocorrências da palavra em cada documento indexado.
O tempo necessário para realizar a indexação inviabiliza a utilização desta técnica, ainda que o consumo de memória seja aparentemente viável.

Implementação com TRIE

Utilizando uma estrutura TRIE R-way, onde cada caractere da chave é utilizado na construção de uma árvore cada palavra-chave indexada é usada como chave e a entrada conrrespondente é outra TRIE que contém as ocorrências para cada documento, a chave dessa segunda TRIE é o nome do documento, também foi utilizada uma função auxiliar que percorre a TRIE extraindo uma lista duplamente ligada com todas as entradas contidas na TRIE, isto foi necessário para um tratamento uniforme do problema, tando com a utilização de uma tabela Hash quanto com a estrutura TRIE.
A memória necessária para realizar a indexação inviabiliza a utilização desta técnica, ainda que o tempo de indexação seja viável.

Experimentos

Por conta das limitações de implementação foi adotada idexação de 1000 linhas da base dados em português para avaliação final.

Hash:

tempo de indexação: 3.98 segundos
memória consumida: 17088 KB
tempo de busca 100 palavras: 0.11 segundos
tempo de busca 1000 palavras: 0.18 segundos
tempo de busca 10000 palavras: 0.23 segundos

TRIE:

tempo de indexação: 0.70 segundos
memória consumida: 165036 KB
tempo de busca 100 palavras: 0.03 segundos
tempo de busca 1000 palavras: 0.08 segundos
tempo de busca 10000 palavras: 0.12 segundos

Tabela de comparação com diferentes números de linhas

Tabela HASH	*1000 linhas*	*10000 linhas*	*20000 linhas*
Tempo de indexação	3.98 segundos	3 minutos e 17.16 segundos	12 minutos e 42.43 segundos
Memória consumida	17.088 MB	140.516 MB	274.772 MB

TRIE	*1000 linhas*	*10000 linhas*	*20000 linhas*
Tempo de indexação	0.70 segundos	5.20 segundos	9.44 segundos
Memória consumida	165.036 MB	1.156360 GB	2.086876 GB

Conclusão

Ambas as técnicas possuem um ponto fraco e forte, tabelas Hash apresentam economia de memória mas consomem bastante tempo na construção do índice, já estruturas TRIE aninhadas são necessariamente rápidas mas consumem muita memória na construção do indice. A principal vantagem a ser considerada na utilização da TRIE é que esta não apresenta problemas de colisão e a complexidade de busca é definida pelo tamanho da chave e não pelo número de elementos no índice, sendo esta uma solução mais estável apesar do alto consumo de memória.