

Data Structure for efficient indexing of files

Mateo Restrepo Sierra

Nicolas Restrepo López

Medellín, Date of the oral presentation

Data Structure design

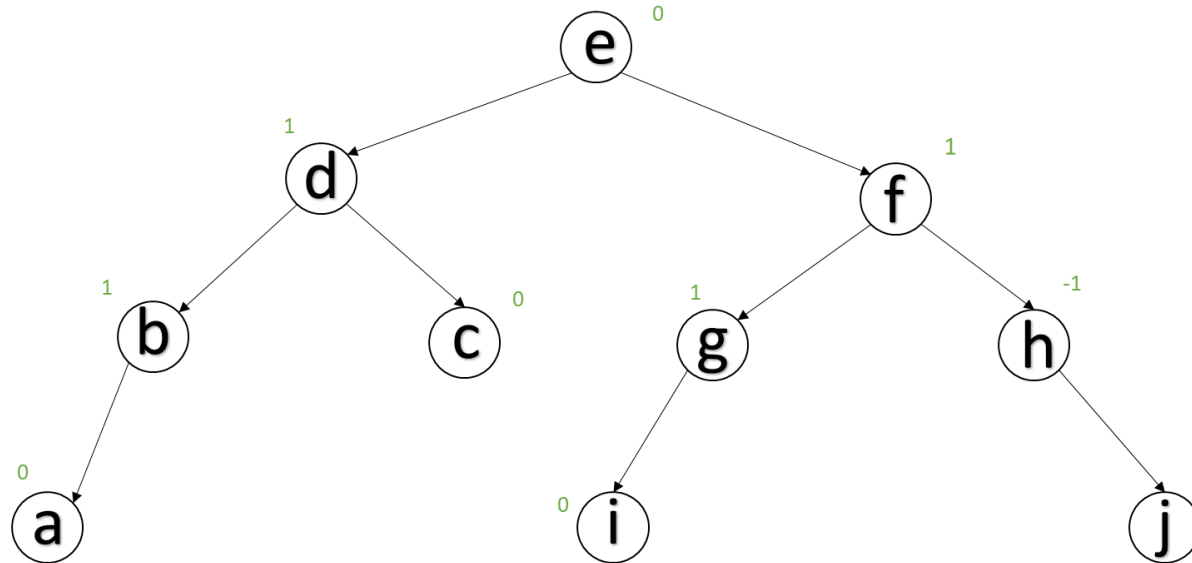


Figure 1: this is a simple example of an AVL tree. the numbers above the tree are the balanced factor

Data Structure Operations

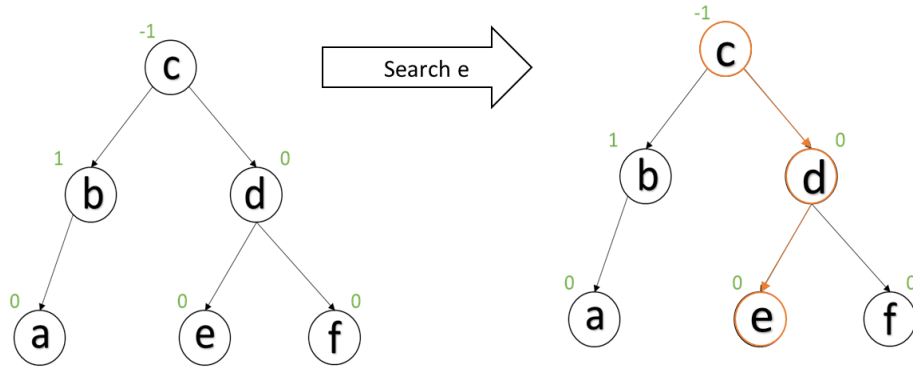


Figure 1: image of an operation of searching

Tabla de complejidades

Método	Complejidad
Búsqueda Fonética	$O(1)$
Imprimir búsqueda fonética	$O(m)$
Insertar palabra búsqueda fonética	$O(1)$
Búsqueda autocompletado	$O(s + t)$
Insertar palabra en TrieHash	$O(s)$
Añadir búsqueda	$O(s)$

Figure 2: image of an operation of searching

Tabla 1: Complejidad de las operaciones de la estructura de datos

Design Criteria of the Data Structure

- To solve the problem, we should use, extensively, the operation of searching.
- The Trees are the most ideal to solve this problem
- Searching has a memory complexity of $O(\log n)$
- The AVL Tree is one of the most efficient Data Structures, because it has shorter height in the subtrees.
- AVL Tree provides faster look-ups.

Time and Memory Consumption

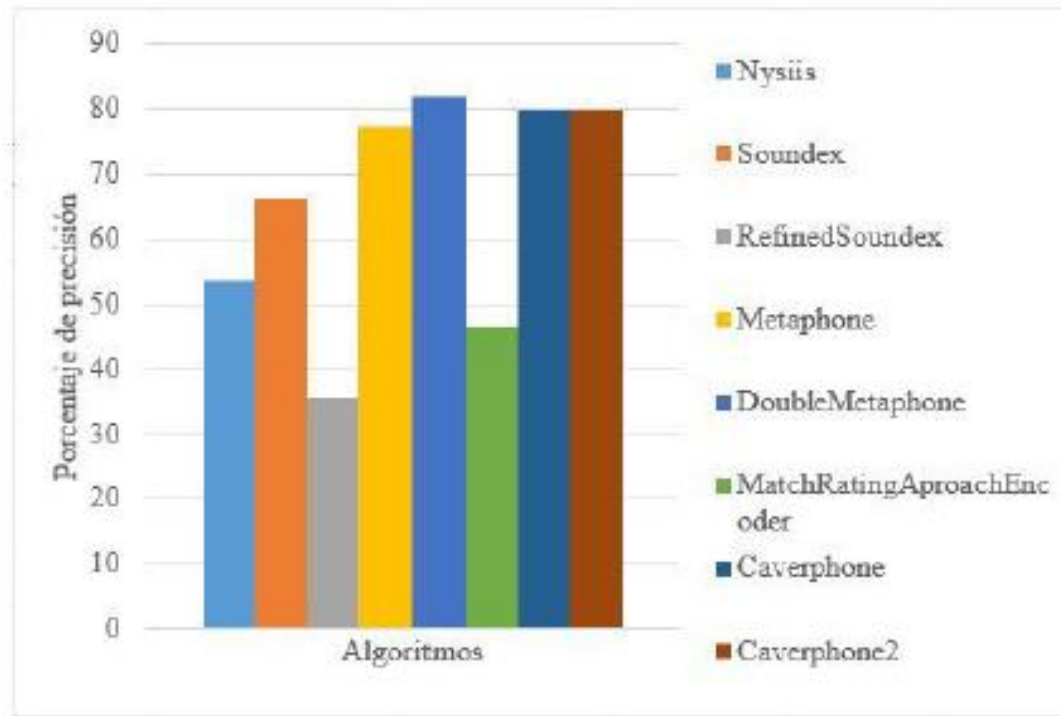


Gráfico 1: Comparación de algoritmos fonéticos de búsqueda

Implementation

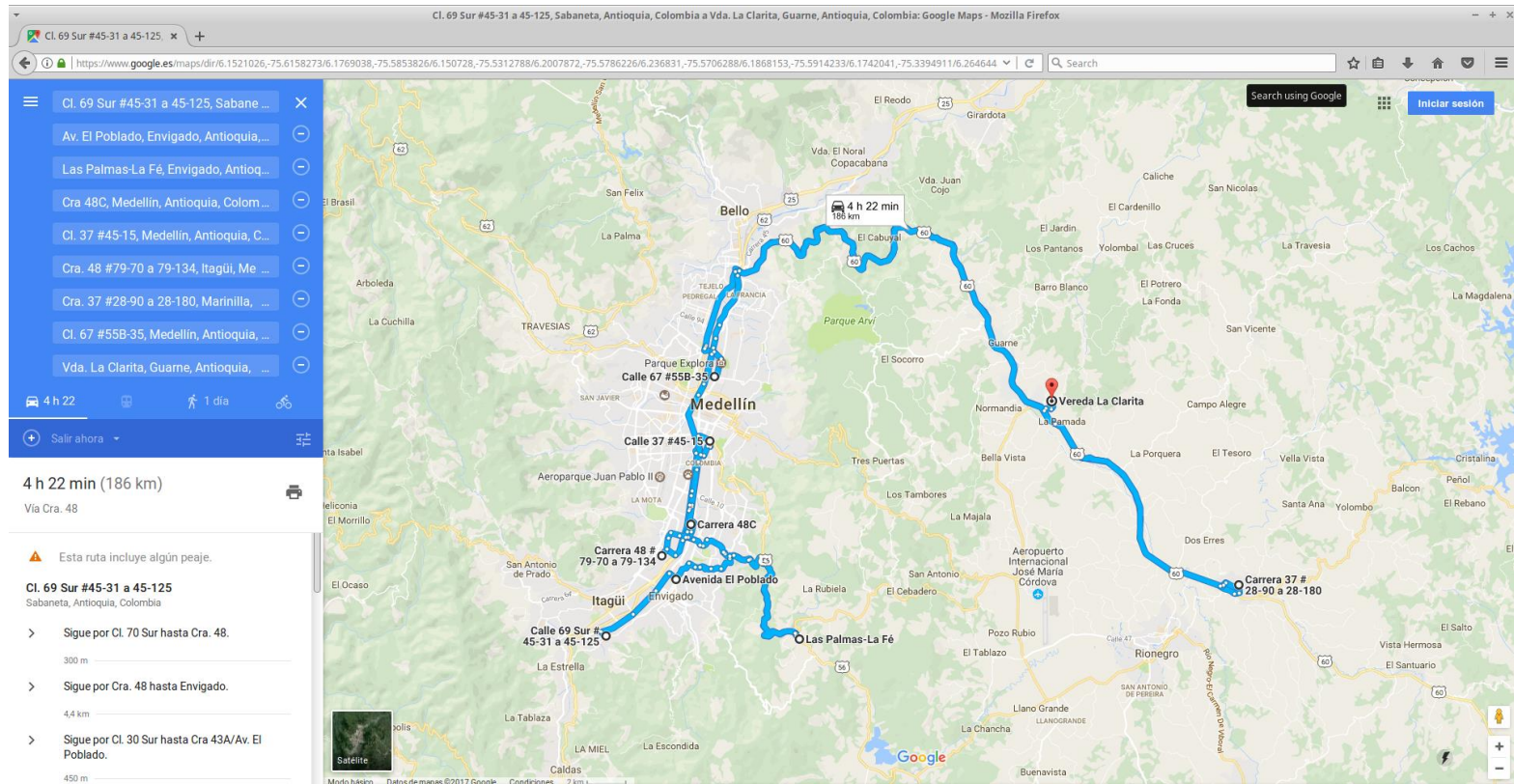



Gráfico 4: Sistema de planificación óptima de domicilios

Report in arXiv



Keep this title

C. Patiño-Forero, M. Agudelo-Toro, and M. Toro. Planning system for deliveries in Medellín. ArXiv e-prints, Nov. 2016. Available at: <https://arxiv.org/abs/1611.04156>



*Insert the URL and
information of your report
accepted in arXiv.*