DESIGN OF A DATA STRUCTURE THAT OPTIMIZES SEARCH EFFICIENCY IN A DIRECTORY (SEARCH SPACE).

Alejandro Murillo González
Juan Pablo Vidal Correa
Medellín, October 31,2017

Designed Data Structure

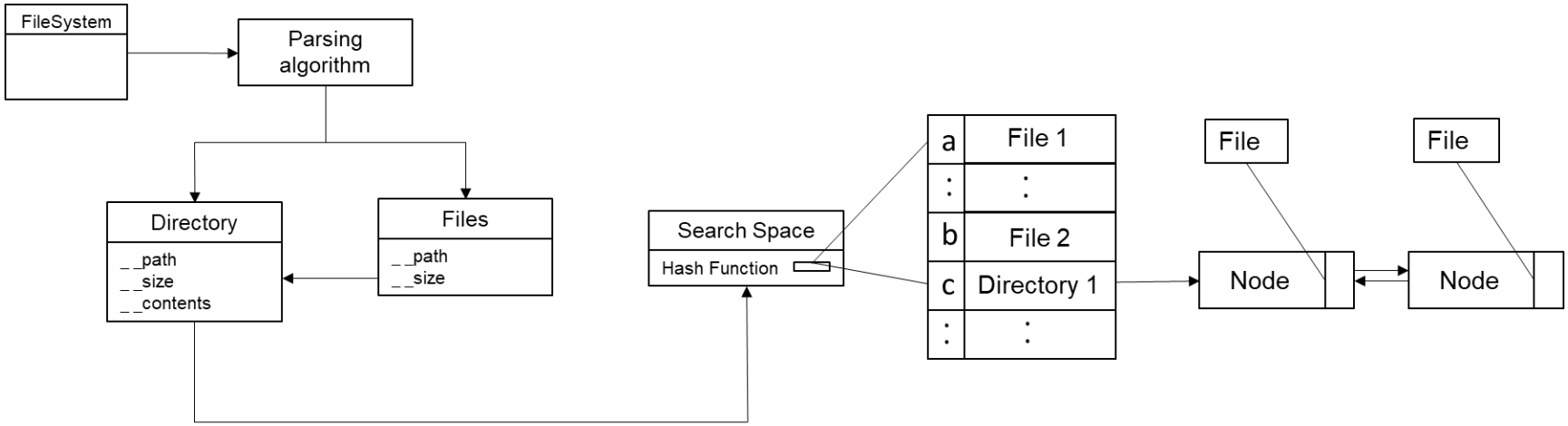

Figure 1: Search space structure.

Data Structure Operations

Search:

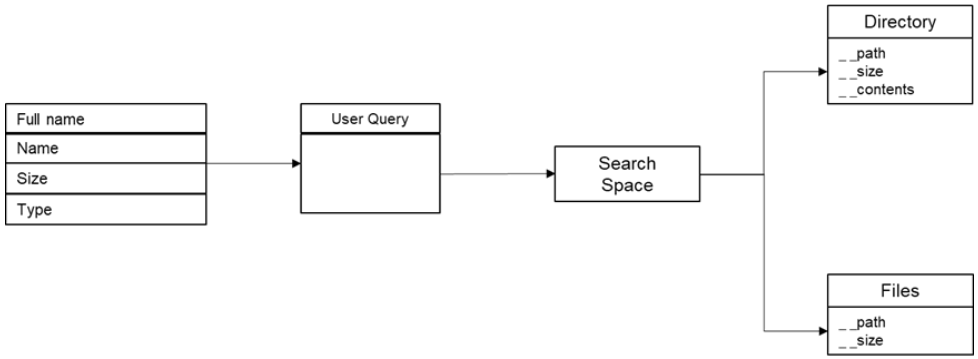

Figure 2: Search Operation.

Insert:

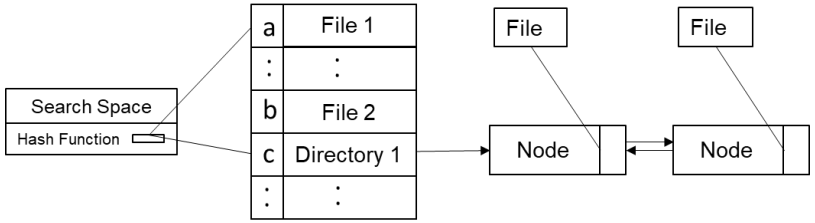

Figure 3: Insert Operation.

Operation	Complexity
Read data	*O(1)*
Insertion (hash table)	*O(1)*
Search	*O(1)*
Print search	*O(n)*

Table 1: Table to report complexity analysis of the data structure

Design criteria of the data structure:

- The design of the data structure considered access time and memory consumption combined with the search parameters that can be used to find a certain file; these might be: name, full name, type, size and path.

- Hash tables' access time compares favorably against other data structures -and considering industry success stories such as Amazon Web Services' use of NoSQL databases in its widely used product DynamoDB.

- The structure of hash tables allows to obtain very short access times with a consumption of memory proportional to the time, in addition the hash tables implemented in python have a complexity of time O (1).[1]

Consumption of Time and Memory

Execution time:

	DataSet 1	**DataSet 2**
Create	0.0028356753674414946 *ms*	0.07583519934008941 ms
Search	1.9451e-11 *ms*	5.688884393961750e-06 *ms*

Memory Usage:

	DataSet 1	***DataSet 2***
Memory consumption	13,5 *MB*	28 *MB*

The DataSet 1 has 3 directories, 18 files files and the DataSet 2 has 425 directories and 3225 files. [3]

Table 2:. Execution time of the data structure's operation for each data set.

Table 3: Memory used by each operation of the data structure in different data sets.

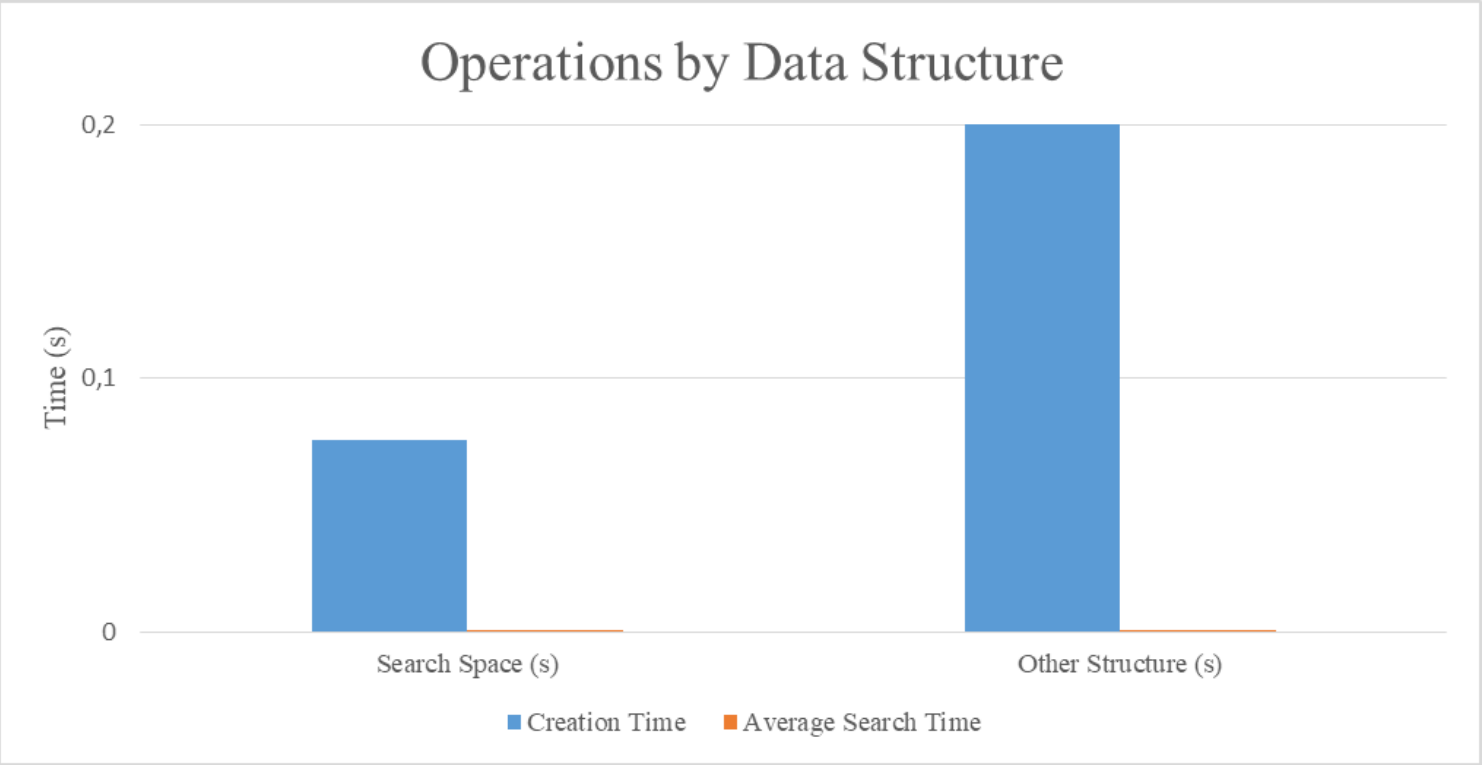

Figure 4: Comparison between access times with other types of structures (Nash table). [2]

*We do not compare memory consumption because the results are very dependent on the equipment in which they are tested.

Data Set 1	Search Space (s)	Other Structure (s)
Creation Time	0,075835199	0,213355
Average Search Time	5,69E-06	2,60E-05

Table 4: Comparison between access times with other types of structures (Nash table). [2]

Implementation

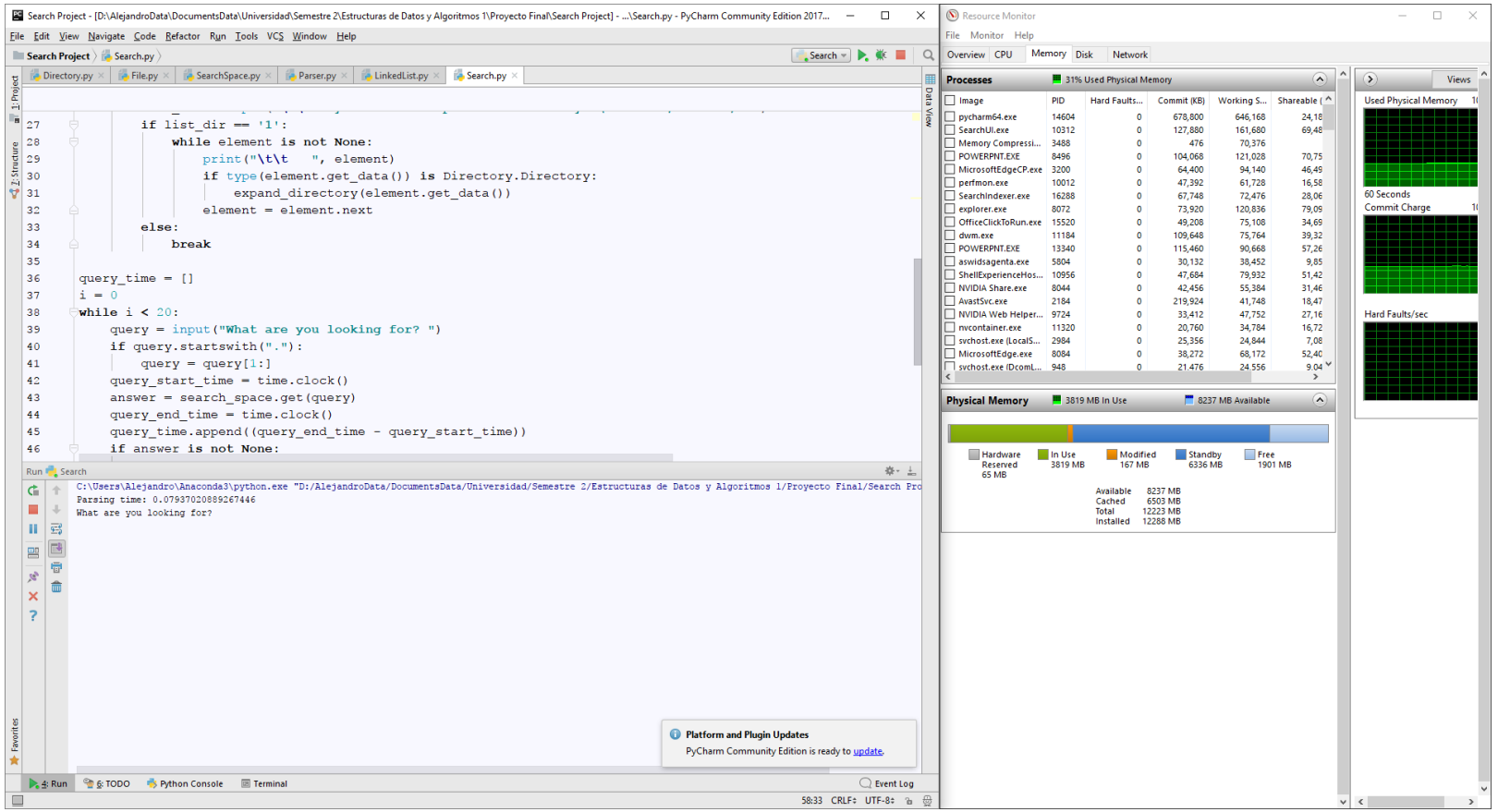

Figure 5: Implementation of the structure

```
C:\Users\Alejandro\Anaconda3\python.exe "D:/AlejandroData/DocumentsData/Universidad/Semestre 2/Estructuras de Datos y Algoritmos 1/Proyecto Final/Search Project/Search.py"
Parsing time: 0.07919993742227166
What are you looking for? pdf
    Nothing was found.
What are you looking for? colors
     [1.4K] colors.xml
     [4.0K] colors
        Do you want to expand the directory? (1 = Yes / 0 = No): 0
     [841] 40.colors
     [3.0K] Oxygen.colors
     [1.6K] Rainbow.colors
     [3.0K] Royal.colors
     [2.5K] Web.colors
```

Figure 6: Implementation of the structure.

Report in kaban table

Member	Date	Complete	Doing	To do
Murillo	15/08/2017	First presentation of the project	Planning the first implementation of the structure	bring ideas for the project
Vidal	22/08/2017	Present ideas for the project: use of hash tables	Developing the idea	Design the structure in python
Murillo	24/09/2017	Check Vidal's implementation	Check Vidal's implementation	Second presentation of the project
Vidal	15/10/2017	Meet with Murillo to plan the final structure	Part of the final delivery	Implement the last installment with murillo

References

1. Hartley, J. TimeComplexity, *Python.* Retrieved October 11, 2017 from Python Wiki: https://wiki.python.org/moin/TimeComplexity.

Github of the other structure:

2. Cardenas,J.S. and Plazas,D.Daplas,Github.Retrieved October 29, 2017 from Github: https://github.com/Daples/ST0245-032

Github of the DateSet:

3. Toro,M.MauricioToro.Github. Retrieved October 29, 2017 from Github: https://github.com/mauriciotoro/ST0245-Eafit