

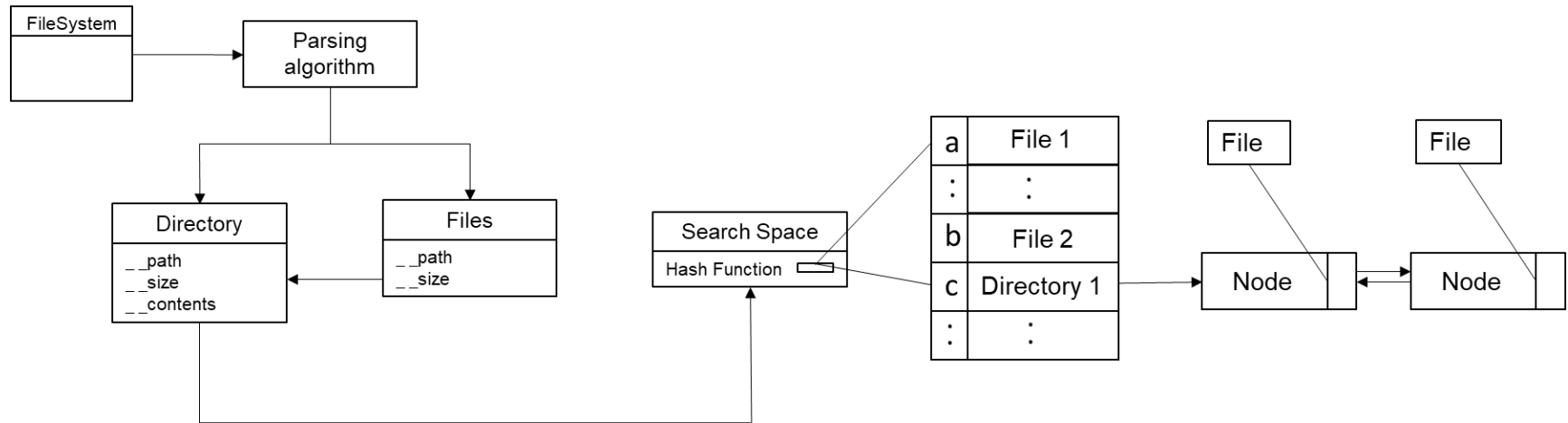
# ***DESIGN OF A DATA STRUCTURE THAT OPTIMIZES SEARCH EFFICIENCY IN A DIRECTORY (SEARCH SPACE).***

***Alejandro Murillo González***

***Juan Pablo Vidal Correa***

***Medellín, October 31, 2017***

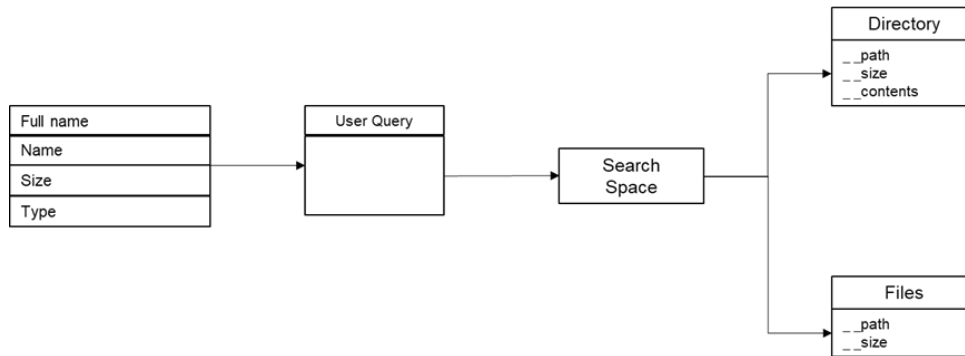
# Designed Data Structure



**Figure 1:** Search space structure.

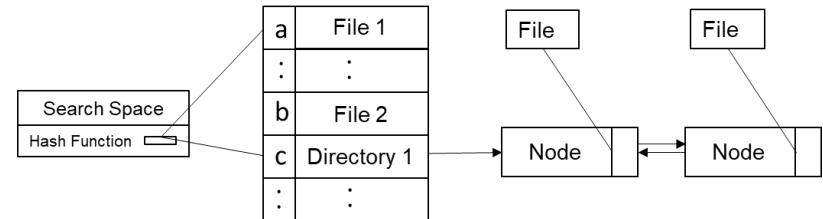
# Data Structure Operations

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:

	<b>DataSet 1</b>	<b>DataSet 2</b>
Create	0.0028356753674414946 <i>ms</i>	0.07583519934008941 <i>ms</i>
Search	1.9451e-11 <i>ms</i>	5.688884393961750e-06 <i>ms</i>

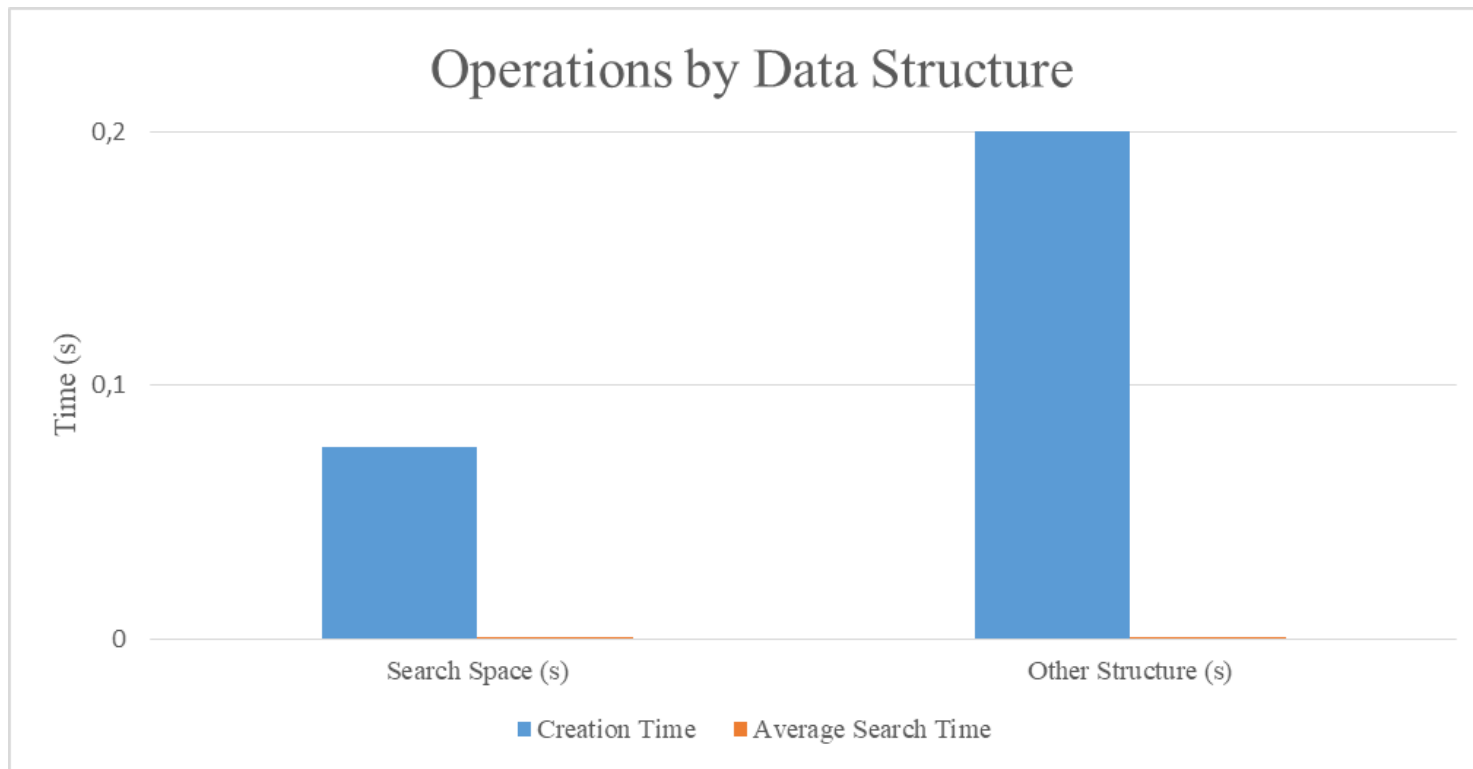
## Memory Usage:

	<b>DataSet 1</b>	<b>DataSet 2</b>
<i>Memory consumption</i>	13,5 <i>MB</i>	28 <i>MB</i>

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

The screenshot displays the PyCharm IDE interface. The main editor shows a Python script with the following code:

```
27 if list_dir == '1':
28     while element is not None:
29         print("\t\t", element)
30         if type(element.get_data()) is Directory.Directory:
31             expand_directory(element.get_data())
32             element = element.next
33     else:
34         break
35
36 query_time = []
37 i = 0
38 while i < 20:
39     query = input("What are you looking for? ")
40     if query.startswith("."):
41         query = query[1:]
42     query_start_time = time.clock()
43     answer = search_space.get(query)
44     query_end_time = time.clock()
45     query_time.append((query_end_time - query_start_time))
46     if answer is not None:
```

The Run window at the bottom shows the command prompt output: "What are you looking for?". The Resource Monitor on the right shows system performance metrics. The Processes tab indicates 31% physical memory usage. The Physical Memory tab shows 3819 MB in use and 8237 MB available.

Process	PID	Hard Faults...	Commit (KB)	Working S...	Shareable (...
Image					
pycharm64.exe	14604	0	678,800	646,168	24,16
SearchUI.exe	10312	0	127,800	161,680	69,48
Memory Compress...	3488	0	476	70,376	
POWERPMT.EXE	8496	0	104,068	121,028	70,75
MicrosoftEdgeCP.exe	3200	0	64,400	94,140	46,49
perfmom.exe	10012	0	47,392	61,728	16,58
SearchIndexer.exe	16288	0	67,748	72,476	28,06
explorer.exe	8072	0	73,920	120,836	79,09
OfficeClickToRun.exe	15520	0	49,208	75,108	34,69
dmv.exe	11184	0	109,648	75,764	39,32
POWERPMT.EXE	13340	0	115,460	90,668	57,26
aswidsagenta.exe	5804	0	30,132	38,452	9,85
ShellExperienceHos...	10956	0	47,684	79,932	51,42
NVIDIA Share.exe	8044	0	42,456	55,384	31,46
AvastSvc.exe	2184	0	219,924	41,748	18,47
NVIDIA Web Helper...	9724	0	33,412	47,752	27,16
nvcontainer.exe	11320	0	20,760	34,784	16,72
svchost.exe (LocalS...	2984	0	25,356	24,844	7,08
MicrosoftEdge.exe	8084	0	38,272	68,172	52,40
svchost.exe IDromL...	948	0	21,476	24,556	9,04

Physical Memory: 3819 MB In Use, 8237 MB Available. Hardware Reserved: 65 MB. Available: 8237 MB, Cached: 6503 MB, Total: 12233 MB, Installed: 12288 MB.

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. Mauricio Toro. Github. Retrieved October 29, 2017 from Github: <https://github.com/mauriciotoro/ST0245-Eafit>