COFFEE LEAF RUST SOLVING: WIRELESS SENSOR NETWORK, DATA STRUCTURES AND ALGORITHMS

Miguel Angel Correa Manrique
Pablo Buitrago Jaramillo

Designed data structure: Table

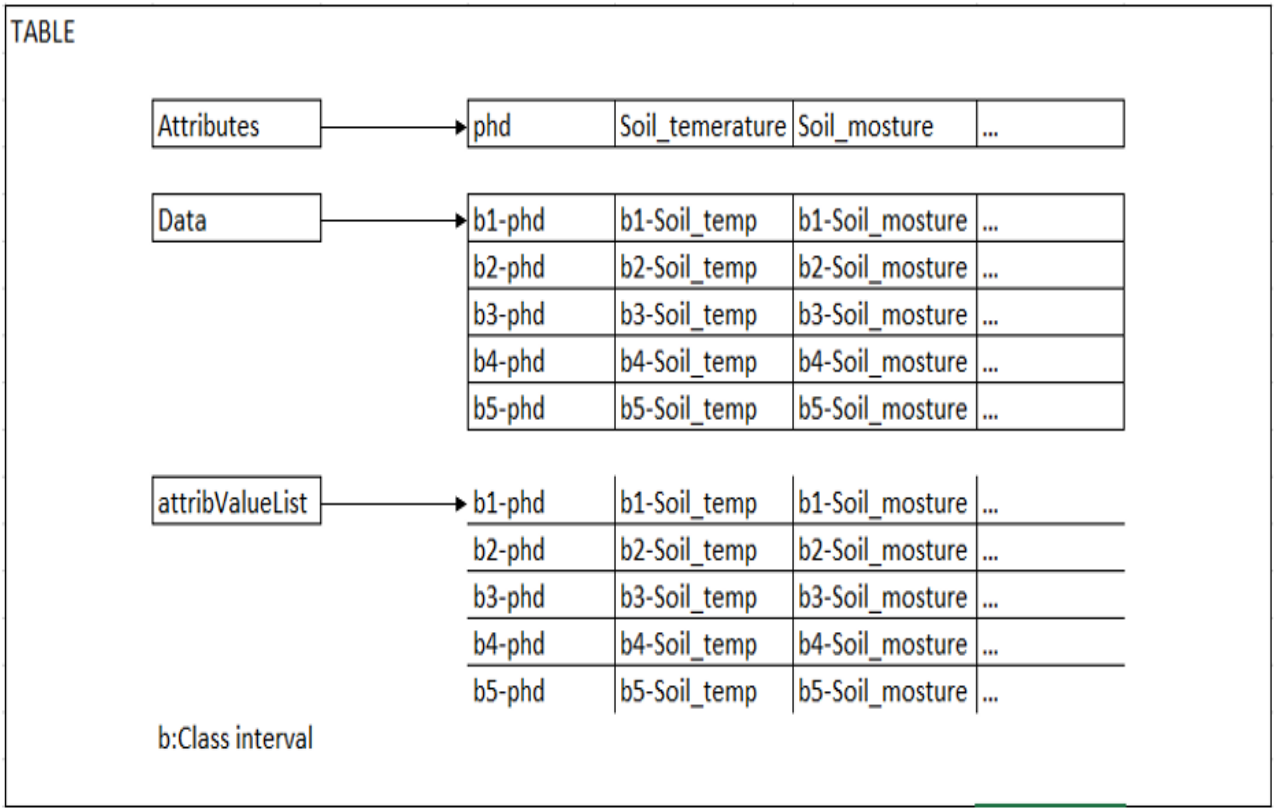


Figure 1: On the abstraction every data attribute represents a column and every data in the structure is allocated in a vector of vectors that represents the rows of the matrix

Data Structure Operations

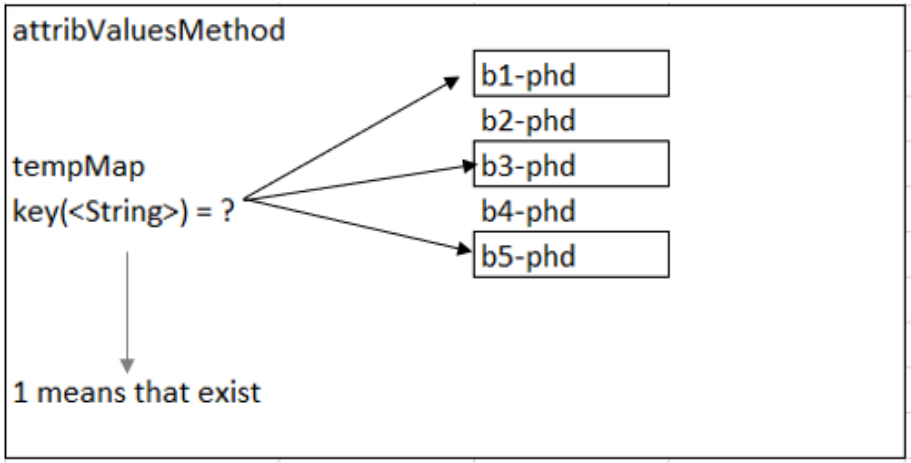


Figure 2: the attribValue Method contains only existent class intervals. It gives the categories on each column of the data set (already discretized).

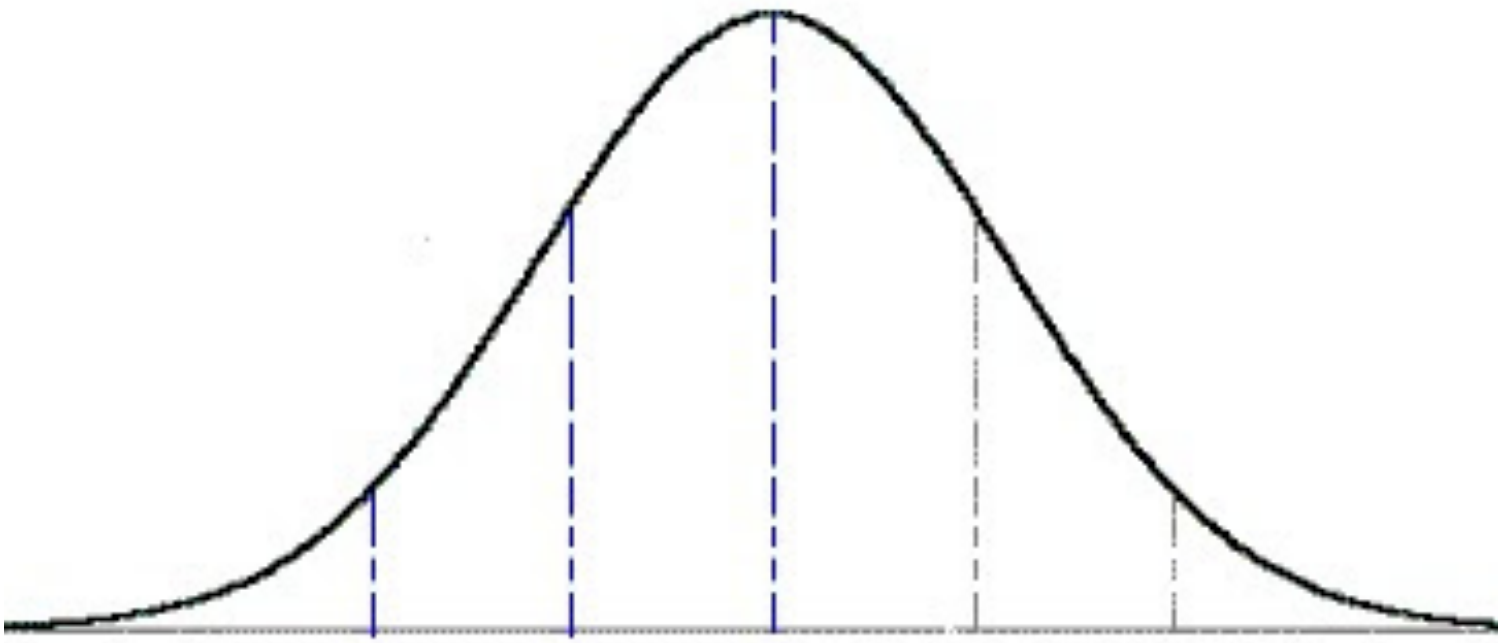
Figure 3 : Demonstrates how the data is discretized graphically.

discretizeData Method

phd	discretization	result
210,1		b1-phd
201,1		b2-phd
301,3		b1-phd
400,5		b1-phd
500,3		b3-phd
302,3		b2-phd

iluminance	discretization	result
11403		b1-illuminance
4000		b3-illuminance
2300,1		b2-illuminance
2000,1		b2-illuminance
4050		b3-illuminance
6000		b4-illuminance

Quartiles



Complexity Table

Method	Complexity
AttribValues()	O(n*s)
discretizeData()	O(n)

n : Amount of rows of the table

s : Amount of attributes

Table 1 : Complexity of the methods presented on the data structure.

Design Criteria of the Data Structure

Attributes: As string vectors.

Data: As vectors of string vectors.

Attribute List: As vectors of string vectors.

- The data values are stored in ArrayList to access the data in constant time and because its insert and delete qualities.
- The time complexity to access on ArrayList is O(1)
- ArrayList is suited to the way we want to address the problem
- On the abstraction the columns are represented by “Attributes” and the data categorization is represented by “Attribute List”

Time and Memory Consumption

	Data Set with Len 300	Data Set with Len 373	Data Set with Len 457	Data Set with Len 673
Creation	0.00062 8721s	0.00074 2888s	0.00089 0782s	0.00133 364s

Table 2 : Execution time of the operations of the data structure for each data set

Table 3 : Memory used for each operation of the data structure and for each data set data sets.
Note that the memory usage is an approximation given by the amount of data received multiplicated by the bytes of the string data type in C++ language times the number of rows presented on the data set.

	Data Set with Len 300	Data Set with Len 373	Data Set with Len 457	Data Set with Len 673
Memory Usage	0.0672 MB	0.08332 MB	0.10214 MB	0.15052 MB

Designed Data Structure: BQPT

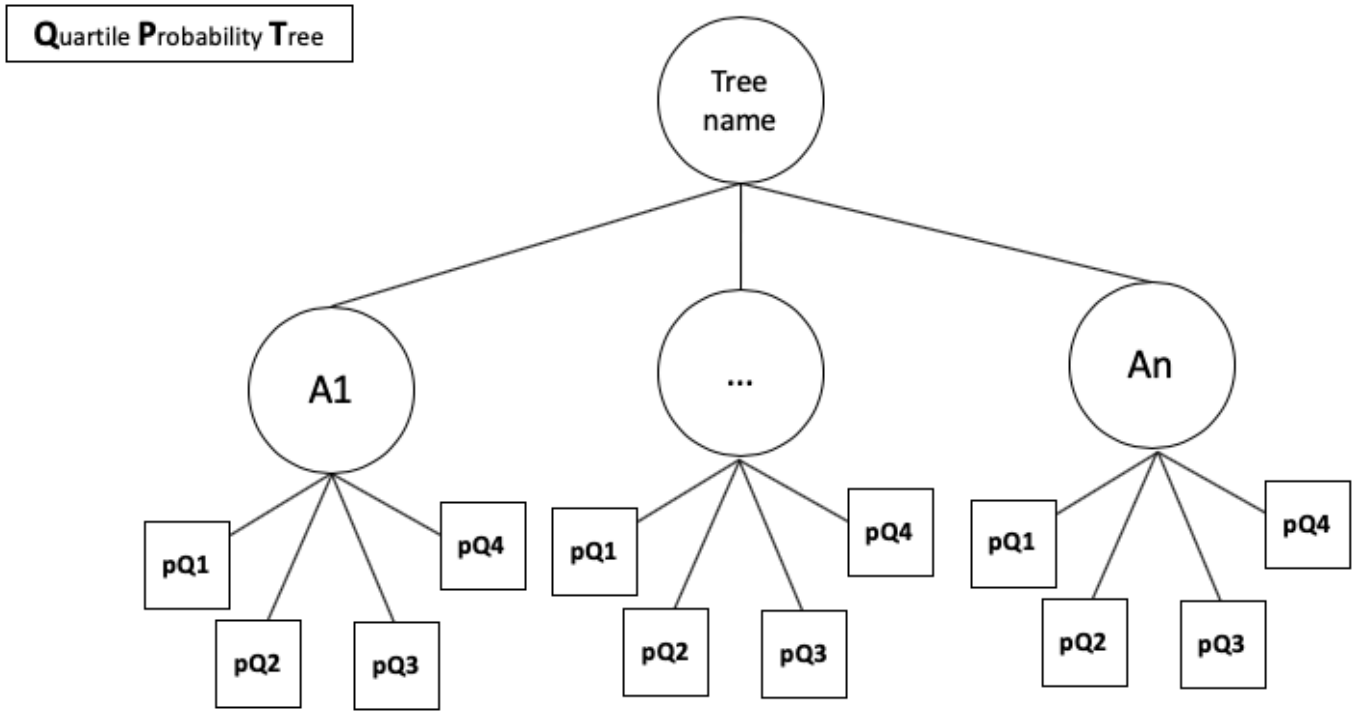
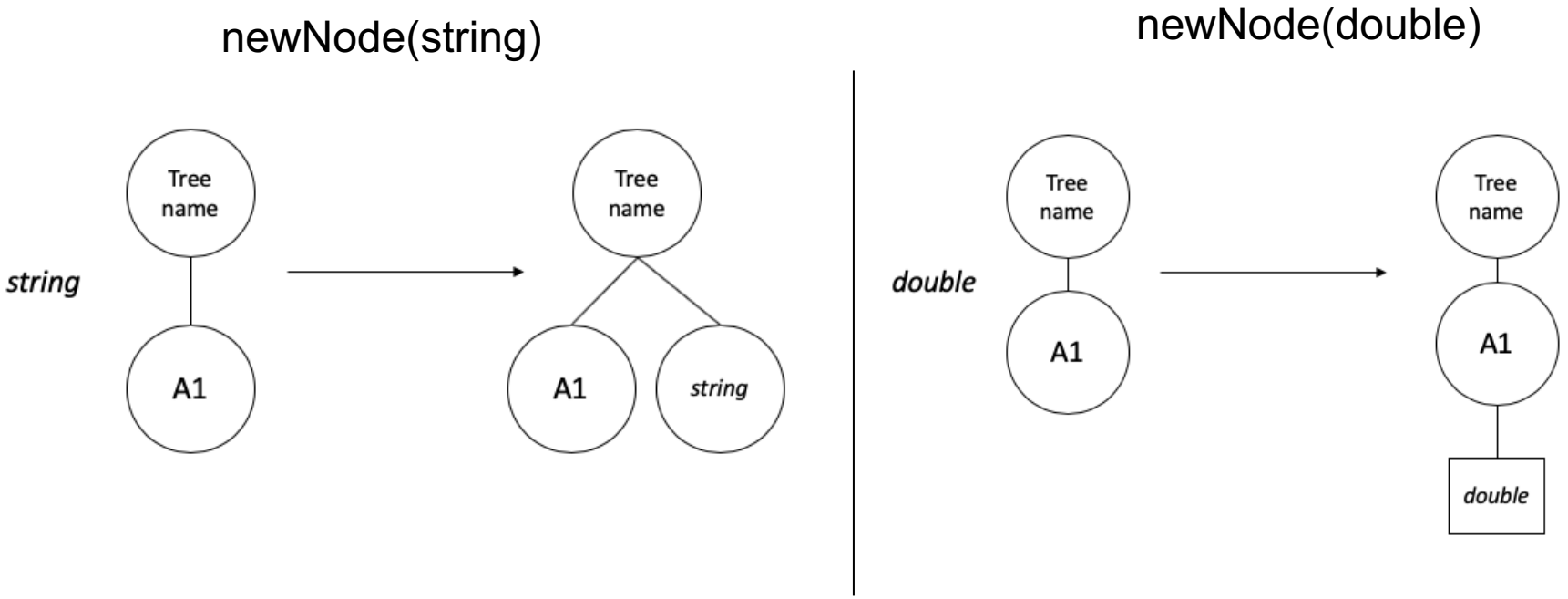


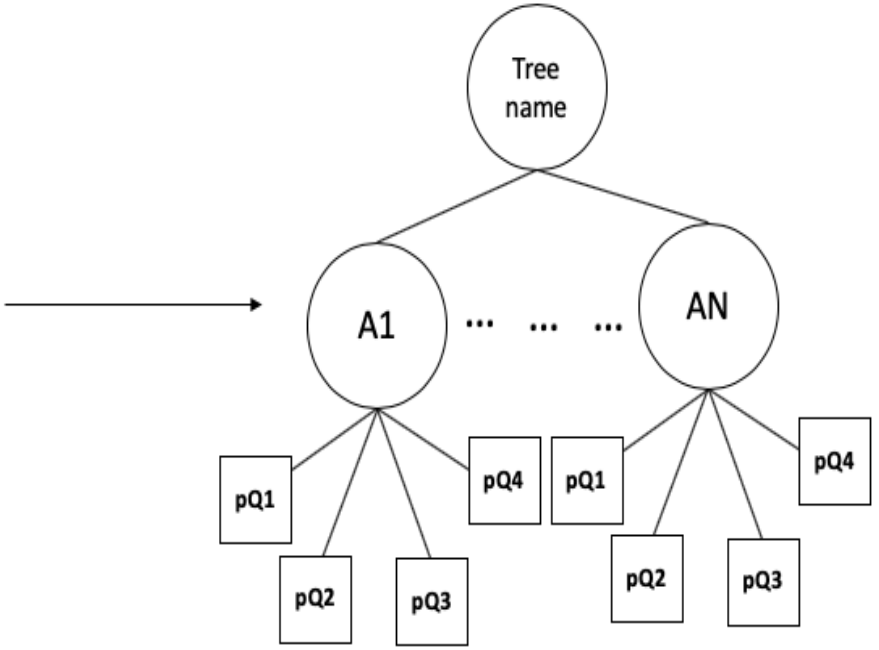
Figure 4: Abstraction of a Quartile probability tree. Each node represents a data attribute, all of them has 4 leaves; a leaf lodges a probability for the data in the class that the leaf represents

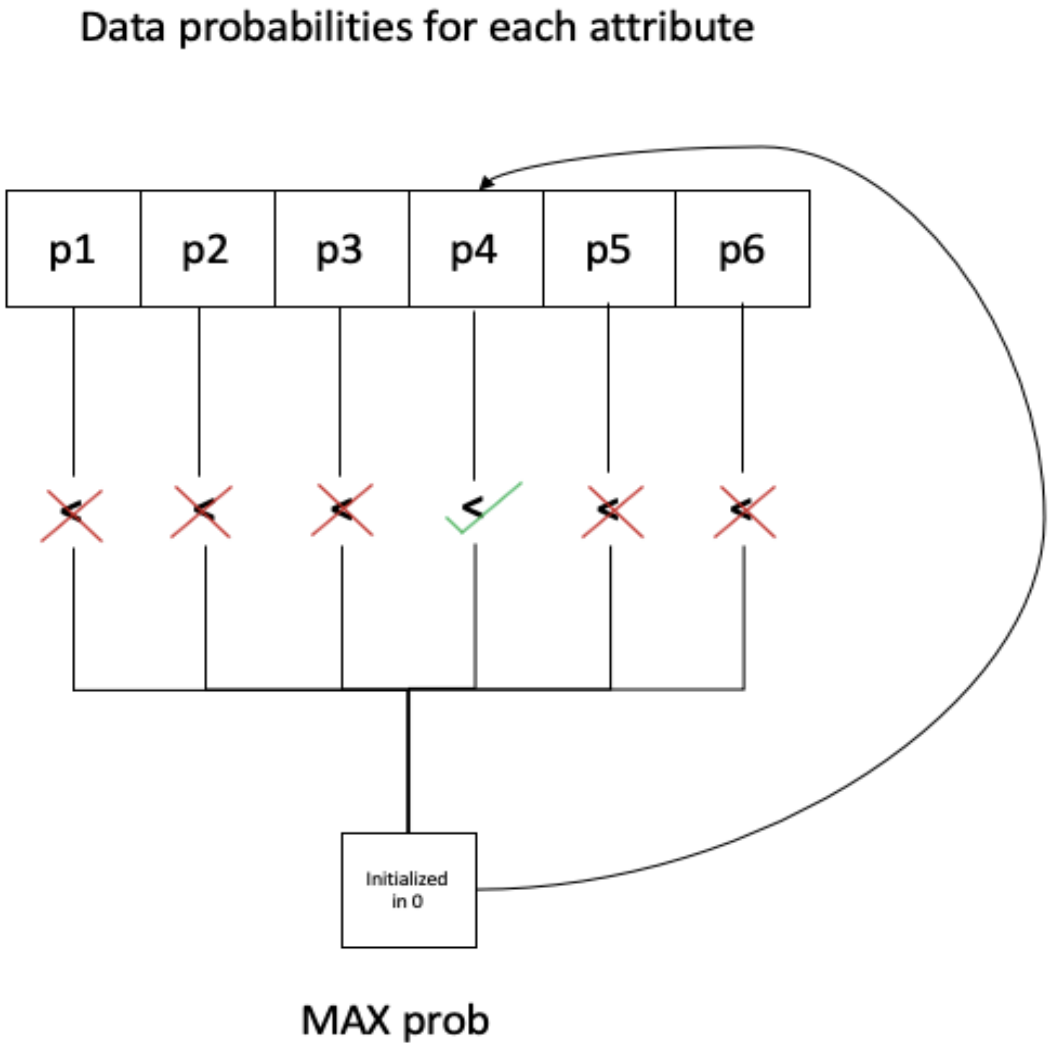
Data structure operations



make(DataFrame)

Attribute1	Attribute2	...	AttributeN
data1_A1	data1_A2	...	data1_AN
data2_A1	data2_A2	...	data2_AN
...	...	...	...
dataN_A1	dataN_A2	...	dataN_AN





decision(string)

Complexity Table

Method	Analysis
make()	O(crlog(r))
decision()	O(c2*r2)

c : Data frame columns used to train the tree

r : Data frame rows used to train the tree

c2 : Data frame columns used to predict

r2 : Data frame rows used to predict

Design Criteria of the Data Structure

- DataFrame:
  - Labels as string vectors
  - Data as vectors of double vectors
- Node:
  - Childs as vectors of nodes
  - Attribute as a string
  - Probability as double

- We need a structure that works as a tree
- Works for every dataset that can be a matrix n*s
- The distribution of the data in the class intervals using quartiles

Time and Memory Consumption

	DataSet1(300.csv)	DataSet2(373.csv)	DataSet3(457.csv)	DataSet4(673.csv)
make()	0.000131085s	0.000161801s	0.000197267s	0.000286038s
decision()	0.00299638s	0.00373048s	0.00449591s	0.00693759s

Table 4 : Execution time of the operations of the data structure BQPT for each data set

Table 5: Memory used for the implementation of the data structure and for each dataset

	DataSet1 (300.csv)	DataSet2(373.csv)	DataSet3(457.csv)	DataSet4(673.csv)
Memory Usage	3.2 MB	3.3 MB	3.6 MB	4 MB

Result Analysis

BQPT	Vectors	LinkedList
Creation	0.003011 - 0.00302 (s)	0.003011 - 0.00302 (s)
Memory Usage	3.2 - 4 (MB)	3.0 - 3.9 (MB)
Decision	0.0073 - 0.0016 (s)	0.0073 - 0.0016 (s)