Oregon State University

Intro to Databases - CS 340

Assignment 3 - Due: 11:59PM Aug. 10

Instructor:
Ben McCamish

July 27, 2018

Overall Assignment

This assignment you will be creating a bitmap index from a data file containing information about pets and whether they are adopted or not. Once you create the bitmap, it needs to be compressed using WAH with 32 and 64 Bit words. The data file will also need to be sorted and then another bitmap created over that. The sorted bitmap index will then need to be compressed using WAH with 32 and 64 bit words.

For this assignment, output all the bitmaps as characters. A character on most architectures consists of a single byte. If the file says, for example, that the size is 8 Bytes, then it would actually contain 8 bits. This is fine for our purposes, but in practice bitmaps are created and queried using actual binary files and bits to increase their speed and reduce on size. **You may not use any libraries for bitmap creation or compression**

Sample Of Database

There are three attributes in the data file, *Animal*, *Age*, and *Adopted*, in that order. I have include one file that you will perform all of your compression and bitmap index creation on titled 'animals.txt'. The other file, 'animals_test.txt' is a smaller file to test your code on, but do not turn in any of the bitmap indexes created from this.

Animal	Age	Adopted
Cat	12	True
Dog	68	False
Dog	33	False

The attributes have the following domains:

- **Animal:** ['cat', 'dog', 'turtle', 'bird']
- **Age:** [1,100]
- **Adopted:** [True, False]

Create a Bitmap Index (20%)

Create two different bitmap indexes of the data file. I will be comparing your bitmap to mine to see if it was created correctly.

Unsorted Bitmap Index

The first bitmap you should make should have the same order as the data file. Your bitmap index should have a column for each possible animal, in the order represented in its domain. Following the four animal columns, the bitmap should have 10 columns that represent the bins for the *Age* attribute. Each bin should be 10 long. For example, 1-10, 11-20, 21-30, etc. The last two columns in your bitmap should represent True and False in that order for the *Adopted* attribute.

Sorted Bitmap Index

Now sort the data file lexicographically and create the bitmap over it again. You should now have two bitmap indexes over the dataset, one created over the unsorted data file and another created over the data file that has been sorted.

Compress the Bitmap Index 32-bit Words (25%)

Compress both your bitmap indexes using the Word Aligned Hybrid method as discussed in class. Use 32-bit words for your compression. In your writeup specify whether the output of your compressed bitmap is column or row oriented.

Compress the Bitmap Index 64-bit Words (25%)

Compress both your bitmap indexes using the Word Aligned Hybrid method as discussed in class. Use 64-bit words for your compression. In your writeup specify whether the output of your compressed bitmap is column or row oriented.

Writeup and Comparison (30%)

- Writeup a summary of all the files that you included. This includes a small description for each file stating what is included in it.
- Then you must compare the size of the 6 bitmap indexes that you have. Write an analysis on why you think they are different size. Did sorting help with the compression and by how much? Did different word sizes have different compression ratios and why do you think that is?
- In addition to your analysis, include the number of fill words and literal words that were compressed for each file.
- Your write up must be in LaTeX . Include all .tex source files and the compiled PDF.

What to turn in (in a zip on Canvas):

- All code (well commented)
- Two bitmap indexes one over the lexicographically sorted data file, the other not
- Two bitmap indexes compressed using WAH 32-bit, one sorted one not
- Two bitmap indexes compressed using WAH 64-bit, one sorted one not
- README.txt on how to run your code (detailed if required).