SAFFRON: Targeting Efficiently

Rucha Kolhatkar

Graduate Mentor: Amirhossein Reisizadeh

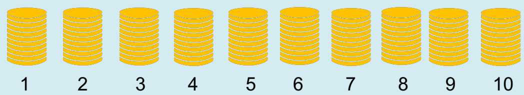
Faculty Advisor: Professor Ramtin Pedarsani

Department of Electrical and Computer Engineering, University of California, Santa Barbara

Introduction

The earliest known application of group testing took place during World War Two, when soldiers' blood samples were tested in pools rather than individually to determine which soldiers were infected with syphilis.

Group testing is a procedure that tests pools, or groups, or items within a large test group, rather than testing each item individually, in order to determine which items are defective, or in the soldiers' case, infected. To demonstrate this procedure, here is a visualization. There are ten stacks of ten coins each, labeled from one to ten, as shown. Nine stacks contain coins that are ten grams each, and one stack contains coins that are nine grams each.

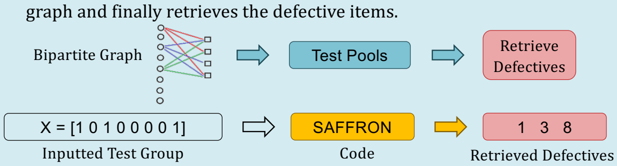


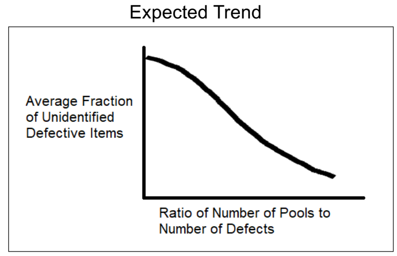
In order to find out which stack contains coins that are nine gram each by conducting only one test, we select a group, or pool, such that it contains one coin from stack 1, two coins from stack two, three coins from stack three, and so on, totaling 55 coins. If all coins in all stacks were ten grams each, we would have a total of 550 grams. If the lighter stack is stack one, we would get 550-1=549 grams, since we selected one coin from stack one. If the lighter stack is stack two, we would get 550-2=548 grams, since we selected two coins from stack two, and so on.

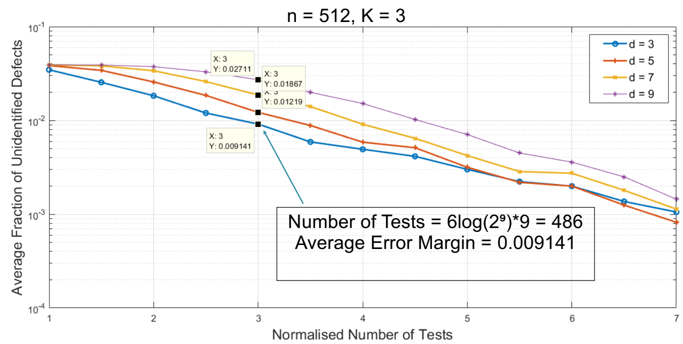
Current research is centred around deriving and implementing a more efficient algorithm by minimising the number of pools. The most recent group testing algorithm, known as SAFFRON, guarantees the identification of defective items with high probability. SAFFRON, with a runtime of $Klog(n)$, is faster compared to the previous algorithm, with a runtime of $K^2 \log(n)$. This project focuses on the implementation of SAFFRON in MATLAB.

Program Design

I learned the existing algorithm while applying concepts from linear algebra, algorithm complexity, and coding theory. From there, I developed a design for the code for SAFFRON, implemented it in MATLAB, and ran it with randomly generated data. SAFFRON takes the inputted test group that contains zeros (not defective) and ones (defective) based on a specified degree of left nodes, or items from the test group, randomly assigns each left node to that number of right nodes or pools (bundle of tests), thus generating a bipartite graph. SAFFRON then performs a series of operations on the bipartite graph and finally retrieves the defective items.







Notation

n = total number of items in test group

M = total number of pools

d = degree of left nodes (items)

K = number of defective items in test group

Analysis

The expected trend of the normalised number of tests, or the number of pools required per known number of defects, versus the average fraction of unidentified defective items is decreasing. The program implementation of SAFFRON produces a similar graph for d = 3, 5, 7, and 9, d being the number of pools an item is assigned to, suggesting that the algorithm has been implemented correctly. As the number of items assigned to each pool increases for any number of pools, the average fraction of unidentified defects decreases.

Applications

Group testing can be applied to various real-life scenarios, from DNA library screening, in which group testing is used to determine which DNA samples contain the complement of a certain gene, to Active Neighbour Discovery, a process in which group testing is applied to find out which devices are using a wireless network and which are not. By applying group testing, time can be saved.

Literature Cited

K. Lee, R. Pedarsani, and K. Ramchandran. Saffron: A fast, efficient, and robust framework for group testing based on sparse-graph codes. In Information Theory (ISIT), 2016 IEEE International Symposium on, pages 2873–2877. IEEE, 2016

Acknowledgements

Thank you to mentor Amirhossein Reisizadeh, faculty advisor Ramtin Pedarsani, Samantha Davis, the EUREKA Program, and CSEP, for providing invaluable guidance throughout the research process. Funding for the project came from the University of California, Santa Barbara.