Implementing Approximation Algorithms: Theory vs. Practice

- Final Report

CREU Final Reports 2015

Student Researchers: Christine Harbour
Jordan King

Advisor: Barbara Anthony

Institution: Southwestern University

Webpage: https://sites.google.com/a/southwestern.edu/creu-approximation-algorithms/

Goals of the Project

In this work, we sought to study how well approximation algorithms do in practice for certain problems. Based on the empirical analysis, we sought to recommend improved approximation algorithms for possibly restricted instances of the problems or provide additional insight into the algorithms beyond their theoretical bounds.

Process

The three of us met twice a week throughout the academic year, with the students sometimes meeting at other times, along with typical communication via email, a shared github repository, and files on Google Drive. We started in August by considering the well-known 2-approximation algorithm for the k-center problem, implementing both that and a brute force solution to calculate the optimal value (which is feasible for small instances). Given the limited range between optimal and the theoretical guarantee, the results on the studied data sets (including random, pre-clustered, and uniformly distributed around a circle) were not surprising. We did observe that sometimes the addition of a single center could make the performance of the algorithm go from close to one extreme to the other, and that the placement of the first center could noticeably impact the performance, leading to suggested heuristics for first center placement.

We then moved on to the online bottleneck matching problem, focusing on three algorithms (Greedy, Balance, and Permutation) and two frameworks (the standard model, and resource augmentation where the algorithm has access to twice as many servers as the offline optimal solution). We implemented each of these algorithms. Greedy and Balance were the easiest, while Permutation was the most complex to implement, and a brute force method for calculating the optimal solution is, not surprisingly, computationally intractable in general. Data sets studied in the empirical evaluation included random data in two dimensions, random data on a line, and specific data constructed from real-world applications. The line is interesting both for the theoretical guarantees that hold true even in that restricted instance and because an offline optimal solution is efficient to compute for the line by assigning servers to requests from left to right (though that assignment does not carry over to the resource augmentation variant).
Throughout the year, the students (and sometimes the faculty member) had to learn new tools, including the well-known git and R, as well as two libraries (Lemon and PAAL) that were specific to the problems being studied. As always, the various aspects of writing (abstracts, papers, conference presentations, good documentation) sometimes took more iterations and time than originally anticipated, but working on them well in advance and in conjunction with the other aspects of the project proved useful. Because of the diverse nature of the audiences to which the work was being presented (including a presentation to the campus community at large), it was incredibly useful to have motivating examples for the problems that were specific to the campus, and to practice explaining many concepts that a general CS audience would be assumed to have.

Conclusions and Results

While theoretical guarantees are likely to remain a major focus of research in approximation algorithms, there has been some recent work in empirical evaluation, and our work continues to convince us of the value of this type of study. From the k-center problem, for certain types of data (uniformly distributed points around a circle), the symmetry and structure allow us to conjecture when the approximation algorithm will do well and when it will do poorly with respect to the optimal solution and the theoretical guarantee. In other cases, particularly for random data, a heuristic for placing the first center, rather than choosing one arbitrarily, can make a marked difference in performance. Finally, in some practical problems, the number of centers to be placed may have some flexibility, and the empirical evaluation indicates that it may often be advantageous to try multiple k values, as increasing the number of centers by 1 can sometimes dramatically improve the approximation.

For the online bottleneck matching problem, despite Greedy having the worst theoretical guarantee, the constructed instances that cause Greedy to perform poorly are likely not representative of real-world problems. In general, Greedy in fact does quite well, and thus its ease of implementation and speed make it a good candidate for this problem in practice. Just as the simplex algorithm is efficient in practice, but has instances where it can have an exponential runtime, our empirical evaluation suggests that Greedy for the online bottleneck matching problem is a good approximation algorithm in practice, though it can do exponentially poorly on contrived instances. Balance, which also uses the greedy paradigm, is similarly easy to implement, but requires the choice of a cost penalty c, and while the theoretical bounds for Balance are based upon a particular value of c, the choice of c greatly impacts the performance of the algorithm in empirical evaluation. In essence, the c cannot be too large with respect to the distances or it will prevent the use of secondary servers, but if it is too small, it will be no different from greedy with resource augmentation. Permutation, the most complex to implement, could sometimes tie Greedy in its performance, but often did worse, and thus may not be worth the implementation challenges. Yet all three approximation algorithms may be worth considering in some situations, particularly when compared to the runtime of a brute force solution. Even with a brute force solution discarding options that are worse than the best known solution thus far, calculating the result on even 50 servers and requests can take days, whereas the approximation algorithms run in minutes on far larger instances.

Presentations and Publications

“An empirical evaluation of a 2-approximation for the k-center problem”, presentation at Southwestern University Research and Creative Works Symposium, Georgetown, TX, April 2015
