Advanced Algorithms (CS/DB 297R) Project
(Submission Deadline: 15/06/2013)

Prof.G.Srinivasaraghavan

February 2, 2013

This document is about the project that you need to submit as part of the Advanced Algorithms course. This is expected to be a group project — in groups of three or less — working on some realistic problem that requires you to use one or more of the methods/techniques that would be discussed as part of the Advanced Algorithms course.

A few rules of the game:

- Each team works on a different topic.
- All documents by default must be typeset in LaTeX.
- The events and dates mentioned below during the course of the project must be strictly adhered to.
- All teams must follow proper version control and source code management using CVS/SVN. Except projects 3 and 4, source code for all other projects must be published under GNU APL and hosted in sourceforge. Those taking the railways projects please set up a version control repository in the CSL Lab with Prof.Prasanna's permission.
- Each group is expected to make a presentation of its findings, the schedule for these presentations will be announced later. The quality of these presentations will be an important part of the evaluation for this course.

Milestone	***Deadline***	***Deliverables***
Project Selection	February 4	Project Name and Groups
Overall Plan	February 16	Plan Document
Key Strategy / High-Level Algorithms	March 9	Strategy / Architecture Document
Detailed Review	March 30, May 15	Demo and/or Partial Results
Final Submission	June 15	Final Report and Implementation Sources

1 Project Titles

This section has some suggested topics for the project. Each team must pick a topic and let me know asap. Feel free to come up with any other topic of your choice if you prefer that - in that case please do discuss with me before you start working on it.

1. **Sampling-Based Motion Planning**: Sampling based algorithms have been successfully employed in several domains. These are in general easier to implement and give reasonable results with limited resources. Refer [10] for a survey of sampling based motion planning algorithms in robotics. Based on the ideas in this paper and some deeper exploration, come up with a sampling based motion planning algorithm and implement it in our robotics lab.

2. **Mobile Robot Global Localization (MRGL)**: MRCS is basically about using local features such as walls to localize the position of a robot. Complications can however arise with noise, moving objects and sensor limitations. There has been work related to bit parallel algorithms for maximum clique applied to robot localization. See Section 4 of [14] for example. Explore one or more of these algorithms and implement it with/without a variant of your own.

3. **Indian Railway Network Scheduling-1**: We are working with Indian Railways to try helping them in managing the railway network better. Indian Railways consists of a massive network of railway tracks with nearly 63000 Km of tracks across India, several thousands of trains and a passenger density (passengers carried per annum per line) of nearly 100000. The idea is to use deep algorithmic ideas to deal with problems like scheduling, optimizing the infrastructure, reducing operating costs, maximizing capacity utilization, etc. under multiple, possibly conflicting, constraints. We are a starting with a pilot in which we will deal with a single line say Delhi (Hazrat-Nizamuddin) to Mugalsarai - this is part of the busy trunk line that goes from Delhi to Kolkata.

   This project must as a deliverable come up with a formulation of the problem as an optimization problem to be able to objectively answer questions like

   - is the current infrastructure saturated (I cannot schedule any more trains given the constraints)?
   - given two schedules, which one is better — in other words which one leaves more 'room' for adding more trains later on?
   - given a timetable how can I 'improve' it?

   The formulation must include at the least

   (a) specification of the data formats needed to represent the data,

   (b) specification of the data structures needed to represent the data externally on the disk and/or in memory,

(c) formulation of the optimization problem taking into account all the relevant constraints that railways would be interested in — line capacity, platforms available, must have trains and schedules, lead and lag times for trains etc.

(d) a menu of algorithms that can possibly be used for the optimization.

4. **Indian Railway Network Scheduling-2**: This project will also be in the context of Indian Railways as described above. However here the focus will be on developing algorithms for finding a new schedule and detecting schedule conflicts. The deliverables must include at the least:

    - representation of the railway timetable (or a subset of it) — the data structures involved, storage criteria etc.
    - given a timetable and a new train schedule, an algorithm to detect conflicts if any between the given schedule and the timetable.
    - assuming the above test says there is no conflict, an algorithm to compute a train schedule that fits into the existing timetable.

    There is a lot of work that has happened on train scheduling as in [15][7][12][11]. The reports published by ARRIVAL project [3] can be a good source of reference and can serve as a good starting point.

5. **Euclidean Traveling Salesman Implementation**: There is a well known geometric algorithm for Euclidean Travelling Salesman Problem that was proposed by Sanjiv Arora [2] that promises, at least in theory, a $(1 + \frac{1}{c})$-factor approximation algorithm for any constant $c > 1$. There have been attempts, as in [4], at trying implement this algorithm efficiently. Study the algorithm and the earlier attempts at implementing it and suggest your own variations as necessary on the original algorithm proposed by Sanjiv Arora that will make the implementation simpler. Implement and/or prove bounds on the performance of your variation/implementation.

6. **Recommender Systems**: Build a recommender system (algorithms and implementation) for movies using the data available in [13] as your test data. You can additionally refer to [6] for a general introduction to recommender systems and [16] for a more technical overview. Preferably use a low-rank approximation scheme as part of your algorithm.

7. **Building an External Memory Algorithm Library**: Design and build the core of a library of algorithms that are *memory-hierarchy aware*. Pick any broad algorithmic domain — sorting and searching, string algorithms, graph algorithms, dynamic programming, etc. — and implement a few generic I/O-optimal algorithms that explicitly exploit the entire memory hierarchy. These [21][1][8][17][5][18][22] should serve as good starting points.

8. **Website Tag Propagation**: This project is about assigning a descriptor tag (metadata) with any given website. Assume we have a 'small' seed of websites that have been tagged manually. Develop an algorithm that propagates these tags to other websites and finally when presented with

a query website, must return the list of say 10 most appropriate tags for the website. Let's say we have the link graph (possibly implicitly). At every incremental time step tags propagate from an already tagged site to its neighbours. Tags can also change, except for those that are tagged manually at the beginning. Explore the algorthmic aspects of this problem including convergence. Also implement with a reasonably sized (say 100000+ — use a crawler for this) corpus of websites. An example of how such a propagation can happen and how linear algebra can help in establishing convergence of such schemes is in one of the earliest papers on Link Analysis by Kleinberg [9] and the more recent work [19][20] relating non-linear dynamical systems to link analysis on the web.

9. **Monte-Carlo Tree Search**: (**Prof. Srinivas Prasanna**) This project will explore the use of Monte Carlo Tree Search in game playing. We will design an iChess player which will use MCTS to strategise based on the capability to explore a game tree from an arbitrary specified start state, going up to games of at least 20 (may be 40) moves. The performance of the resulting player will be compared with the existing ichess player, with some enhanced features. It will be especially interesting to compare the performance of the resulting player on start state that it has not been exposed to.

10. **Automatic Feature Selection for OCR Classification**: (**Prof. Srinivas Prasanna**) We wish to distinguish Indian scripts using a high speed pattern recognizer. The diversity of scripts makes it difficult to have one set of features for each, and even to decide what features are best. This project will use a gray scale representation of Akshara's in Indian Lipis using a 6×7 pixel bitmap and automatically come up with a minimal feature set and associated (linear) classifier for that Lipi. This will be tested on Kannada and Devanagari. The project should handle composite characters (samyukaaksharas) and vowel-signs (maatras).

References

[1] Akshat Verma, S. S. Algorithmic ramifications of prefetching in memory hierarchy.

[2] Arora, S. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *Journal of the ACM 45*, 5 (September 1998), 753–782.

[3] Arrival project technical reports. http://arrival.cti.gr/index.php/Documents/ListWP.

[4] Barbara Rodeker, M. Virginia Cifuentes, L. F. An empirical analysis of approximation algorithms for euclidean tsp.

[5] Barsky, M. *Suffix Trees for Very Large Inputs*. PhD thesis, University of Victoria, Canada, 2010.

[6] Francesco Ricci, Lior Rokach, B. S. Introduction to recommender systems handbook. In *Recommender Systems Handbook*, B. S. Francesco Ricci, Lior Rokach, Ed. Springer Science+Business Media, 2011.

[7] FUCHSBERGER, M. Solving the train scheduling problem in a main station area via a resource constrained space-time integer multi-commodity flow. Master's thesis, Institute for Operations Research, ETH Zurich, 2006/2007.

[8] KANGHO ROH, MAXIME CROCHEMORE, C. S. I., AND PARK, K. External memory algorithms for string problems. *Fundamenta Informaticae 21* (2001), 1001–1017.

[9] KLEINBERG, J. M. Authoritative sources in a hyperlinked environment. *Journal of the ACM 46*, 5 (September 1999), 604–632.

[10] LAVALLE, S. M. Sampling-based motion planningr. Cambridge University Press, 2006.

[11] MALACHY CAREY, I. C. Scheduling trains on a network of busy complex stations. *Transportation Research Part B*, 41 (2007), 159–178.

[12] MATTHIAS MULLER-HANNEMANN, E. A. Timetable information: Models and algorithms. Tech. Rep. TR-0015, ARRIVAL Project, 12 2006.

[13] Movie lens ratings data. http://www.grouplens.org/node/73.

[14] PABLO SAN SEGUNDO, D. R.-L., AND ROSSI, C. Recent developments in bit-parallel algorithms. http://www.intechopen.com.

[15] PETERSON, B. K. *Transportation Scheduling Methods*. PhD thesis, Tepper School of Business, Carnegie Mellon University, 2010.

[16] PREM MELVILLE, V. S. Recommender systems. In *Encyclopedia of Machine Learning*. Springer-Verlag Berlin Heidelberg, 2010, ch. 00338.

[17] REZAUL ALAM CHOWDHURY, V. R. Cache-oblivious dynamic programming. SODA.

[18] SANDEEP TATA, RICHARD A. HANKINS, J. M. P. Practical suffix tree construction. VLDB Conference.

[19] TSAPARAS, P. Application of non-linear dynamical systems to web searching and ranking. In *WWW 2003*.

[20] TSAPARAS, P. *Link Analysis Ranking*. PhD thesis, University of Toronto, 2004.

[21] VITTER, J. S. External memory algorithms and data structures: Dealing with massive data. *ACM Computing Surveys 33*, 2 (June 2001), 209–271.

[22] ZEH, N. I/o-efficient graph algorithms.