MADALGO
CENTER FOR MASSIVE DATA ALGORITHMICS


Oren Weimann
Massachusetts Institute of Technology

Accelerating Dynamic Programming

What

Dynamic Programming (DP) is a fundamental problem solving technique that has been widely used for solving a broad range of search and optimization problems. While DP can be invoked when more specialized methods fail, this generality often incurs a cost in efficiency.

Problem

Devise a unifying framework for speeding up dynamic programming algorithms.

Solution

A toolkit of acceleration tricks such as text-compression, total monotonicity, partial DP tables, utilizing sparsity, fractional subproblems, fast min-plus matrix multiplication, and bounded treewidth algorithms.

Usage

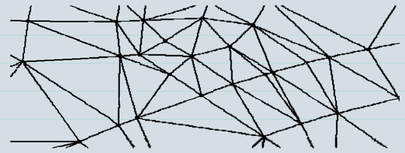
Our speedups apply to many fundamental computational problems such as shortest paths, matrix multiplication, edit distance, HMM decoding and training, array searching, tree searching, and problems on graphs of bounded treewidth.

Acceleration via Text-Compression

We use compression in order to identify repeats in the table that imply a redundant computation.
We present the first provable speedup of the celebrated Viterbi algorithm (1967) that is used for the decoding and training of Hidden Markov Models (HMMs). Our speedup relies on the compression of the HMM's observable sequence.

Totally Monotone Matrices

A wide variety of DPs can be reduced to the problem of finding row minima in totally monotone matrices. We introduce this scheme in the context of planar graph problems. We show that problems such as shortest paths, feasible flow, bipartite perfect matching, and replacement paths can be accelerated by DPs that exploit a total-monotonicity property of the shortest paths.

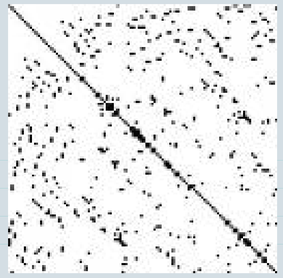


Partial Dynamic Programming Tables

In typical DP settings, a table is filled in its entirety. In some cases, by changing the DP, it is possible to compute asymptotically less cells of the table. We show that n^3 cells are both necessary and sufficient for computing the similarity between two trees. This improves all known solutions and brings the idea of partial tables to its full extent.

Utilizing Sparsity

We can use the sparsity inherent to some problems such as *tree LCS.* We find for each point of the sparse problem a geometric region of the DP table in which that point can influence the values of other points.



Fractional Subproblems

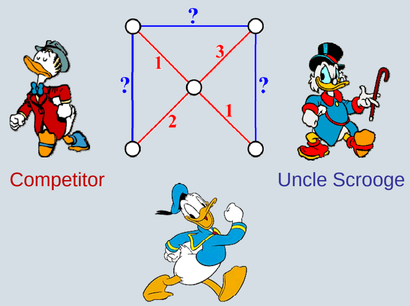
We suggest a method of reusing only parts of a subproblem's data x½xxstructure. In some cases, such fractional parts remain unchanged when constructing the data structure of larger subproblems. We show that this can be used for finding the optimal tree-searching strategy (i.e. binary searching a tree) and for file-system synchronization.

Combining Compression and Total Monotonicity

We introduce a method for accelerating string edit distance computation by combining compression and totally monotone matrices.
In the heart of this method are algorithms for computing the edit distance between two straight-line programs. These enable us to exploits the compressibility of strings, even if each string is compressed using a different compression scheme.

Bounded Treewidth Graphs

We present the first dynamic programming solution for an NP-hard two-player game on bounded treewidth graphs. Namely, the *Stackelberg Minimum Spanning Tree Game.*



Min-Plus Matrix Multiplication

Many dynamic programming problems reduce to computing the min-plus product of two matrices. We explore various problems that admit restricted matrices whose min-plus product can be computed efficiently.

$$A = B \oplus C$$

$$A_{i,j} = \min\{B_{i,k} + C_{k,j}\}$$