

CS138 Set #3

Chris Beck

May 15, 2009

11.2

To see the $n^{O(1/\epsilon)}$ bound on the number of valid visits, observe that there are $4m$ portals into a square, and if there are a total of r paths amongst them that are nonintersecting, then the valid visit information is characterized by a selection of $2r$ end points to restrict attention, out of the $4m$ choices, as well as a pairing of these $2r$ end points into pairs that “enclose” each other without intersecting – this notion is intuitively equivalent to a valid parenthetization using $2r$ symbols, and thus is equal to the $2r$ 'th Catalan Number, which is bounded by $2^{2r} = n^{O(1/\epsilon)}$. Multiplying by $4m$ choose $2r$ does not upset this bound, and adding together these terms as r varies from 0 to $2m$ still yields $n^{O(1/\epsilon)}$.

We need to prove a similar bound on the number of ways of refining a valid visit for a square S to a collection of four valid visits for each of its immediate subsquares. We have again $4m$ points on the lines cutting S' in half at right angles which define the boundaries of the subsquares S , and each portal may be used in one of three ways – passage counterclockwise, clockwise, or not used. Thus we have only $3^{4m} = O(n^{1/\epsilon})$ configurations. Once we've picked a configuration, the only ways of extending it to a full valid visit in each subsquare is to pick a paranthesization for the portal uses in each subsquare, so the Catalan bound proved above gives us what we need here as well. We conclude that there are $O(n^{1/\epsilon})$ ways of refining a valid visit to a useful square S .

The algorithm works by dynamic programming. For each useful square S , it tabulates all possible valid visits to this square ($n^{O(1/\epsilon)}$ many), and records what was the optimal length way of achieving that visit and what the length was. Once all consituent squares of a larger square S' have been handled, we begin to build up the table for S' as follows. , and we may consider, for any valid visit to S' , all possible ways of refining it to include the intersections with these two lines at portals. Only naive combinatorial methods are needed here, such as iterating over all valid parenthetizations, and the bound proved above shows that we will not take more than $n^{O(1/\epsilon)}$ time at any square. Once we've refined a valid visit for S' to a valid visit for each of its immediate subsquares, we look up the stored data for each of those valid visits, add the lengths, and store some information to remember how to reconstruct the path. After we've iterated over all refinements, we choose the most efficient one and save that and its information as the data associated to his valid visit of S' , and proceed to the next square.

Some care is needed at the root of the recursion. Obviously we will never enter or leave the largest square, so we will only consider refinements of the empty valid visit which never touches the boundary of the largest square. Of the possible refinements to valid visits of the subsquares, we will quickly examine the graph formed by the entrance and exit pairs, and check that it is connected, i.e. the cycle formed is truly a cycle and not a disjoint union of cycles. If not, we will disqualify that refinement, and so the result that we end up with will be a tour, and in fact, the optimal well-behaved tour with limited crossings.

1 12.7

Via the hint on the website / wikipedia, incidence matrices for Graphs are totally unimodular. It is clear that taking the transpose preserves total unimodularity, since this preserves determinants, as does adjoining an identity matrix, since if we take a submatrix with some rows or columns drawn from the identity matrix,

we may employ laplace expansion along these rows or columns, there will only be one nonzero entry since it is the identity matrix, and we may continue until all these rows are eliminated. The submatrix we are left with will be in the original matrix, so its determinant will be 0, +1, -1 by assumption; the laplace expansion incurs a +1 or -1 factor, thus we have what we wanted.

This proves that the constraint matrix (12.6) is totally unimodular. Now, we consider an extreme point solution, which lies in $R^{|E|+|V|}$, and which by a different hint in the book must satisfy at least $|E| + |V|$ linear independent inequalities with equality. Let v be an extreme point solution, and then select $|E| + |V|$ equalities as above (these are rows of the constraint matrix), and consider the square submatrix A consisting of those rows, with c the column of constants corresponding to those equalities (which in fact has all zeros except for a single 1), so that we have

$$vA = c ,$$

where v is written a row vector. Now, using Cramer's Rule, the i 'th entry v_i of v may be computed by calculating the ratio of $\det(A)$ to the matrix formed from A with c substituted for the i 'th column - since c is all zeros save a single one, this is the same by laplace expansion as the determinant of a submatrix of A . By unimodularity, this submatrix has determinant 0, 1, or -1. A does as well, but is known to be nonsingular since its rows are linearly independent. Thus Cramer's rule gives $v_i = 0, 1, \text{ or } -1$. By nonnegativity constraints and feasibility of v , we in fact have $v_i = 0$ or 1.

2 12.8

Some people did 12.8 instead, so I sketch a proof here.

Using techniques similar to the proof above, or the sufficient conditions hinted to on Wikipedia, it can be shown that LP (12.8) has totally unimodular constraint matrix, and must have integer solutions; the specific constraints themselves then show that the solutions must be 0/1.

The dual is easily seen to be the LP-relaxation of minimum vertex cover, as claimed in the problem, since the constraint matrix is simply the (unsigned) incidence matrix of the graph.

Then, strong LP duality yields the theorem.