6.851 Advanced Data Structures (Spring'12)

Prof. Erik Demaine TAs: Tom Morgan, Justin Zhang

$\boxed{\text{Problem 10}}$ *Sample solution*

Compact balanced parentheses.

1. Construct a graph G with a vertex for each block, and an edge between the the block containing each parenthesis and the block containing its match. Observe that G is a simple graph, since if two pioneer parenthesis lie in the same block, their matching parens cannot also lie in the same block. We construct the following planar embedding for this graph: place the vertices in order along a horizontal line, and draw the edges below this line such that the edges do not cross. This is possible because of the nesting property that the parenthesis pairs must obey. Since a planar embedding exists, G is a planar graph and thus the number of edges is at most three times the number of vertices. There are $|S|/B$ vertices, and there is a bijection between the edges and pioneer parenthesis, thus there are at most $3|S|/B = O(|S|/B)$ pioneer parenthesis, as desired. For the tight bound of $2|S|/B - 3$, observe that this graph is also an outerplanar graph (as shown by the same embedding).

2. - Divide S into blocks of size $B = 1/2 \lg |S|$.
   - Make a table giving the block containing the matching parenthesis of each pioneer parenthesis.
   - Make a rank data structure over the bit vector of pioneer parenthesis (1 for pioneer, 0 otherwise).
   - Make a rank data structure over the open parens in S, and another over the close parens in S.
   - Make a lookup table T_1 which given a block and an index into the block ($\sqrt{|S|} * 1/2 \lg |S|$ possibilities) returns its matching paren if it is in the block, or say it isn't there.
   - Make a lookup table T_2 which given a block and an integer k between $-B$ and B ($\sqrt{|S|} * \lg |S|$ possibilities) return the first closing paren in the block such that the number of open parens minus the number of closing parens in the block up to that paren equals k.

   To do query on a paren p, first check T_1 for its block, if its closing paren is there, then return it. Otherwise, use rank over pioneers to look into the pioneer table to find which block b p's matching paren is in. Then use T_2 for b with k equal to the difference between the number of open parens and the number of close parens up to p minus difference between the number of open parens and the number of close parens up to b (computed using rank over open paren and rank over close paren structures).