CSE 599S Proof Complexity Fall 2020

Exercise: Due Friday, October 23

3. In class we proved that PHP_n^{n+1} requires resolution refutations of size at least $2^{n/20}$. While this is exponential, the truth is probably somewhat larger still. This problem is about getting a much larger lower bound for *tree* resolution proof size by a much simpler argument. In this case, it is simpler to work with the *functional* pigeonhole principle directly (and a lower bound for that implies the same lower bound for PHP_n^{n+1}).

   (a) Use the following general idea to prove that the tree resolution size for $function$-PHP_n^{n+1} is at least 2^n:

   - View a tree resolution refutation of $function$-PHP_n^{n+1} as a DPLL refutation where we have removed the unit clause rule and each node v is associated with a partial assignment α_v that falsifies the tree-resolution clause labeling v.
   - Call a branch step on variable x_{ij} a *splitting step* if neither child is a leaf that falsifies a Hole Clause or Function Clause of $function$-PHP_n^{n+1}.
   - Argue that if node v is reached after t splitting steps from the root, then there is restriction associated with a partial matching of pigeons to holes of size at most t that implies all the values set in α_v.
   - Argue that every non-leaf node must reach a node labeled by a Pigeon Clause.
   - Prove that all root-leaf paths to Pigeon Clauses must be have $\geq n$ splitting steps and use this to argue that the tree-resolution proof has at least 2^n leaves that are labeled by Pigeon Clauses.

   (b) Describe the most efficient tree-resolution refutation of PHP_n^{n+1} that you can think of. How close can you get to size 2^n?