Description of Manthan

Manthan takes $\exists F(X, Y)$ formula as an input and returns Skolem function $F(X, \Psi(X))$ as output. It takes a qdimacs formula as input where X variables are specified as "a", and Y variables are specified as "e".

Sampling Using an adaptive weighted strategy, Manthan first generates the required number of samples, which is fixed according to the cardinality of Y variables. You can also provide the number of samples using *–samples –maxsamp* $\langle int \rangle$.

Learning Candidates Manthan learns the candidate function for each y_i variable using X and $\hat{Y}$ as feature set, where $\hat{Y}$ represents the subset of Y variables that do not depend on y_i. You can use *–showtrees* to dump the learned decision trees. It will store the learned decision tree of y_i variables as $y_i.png$.

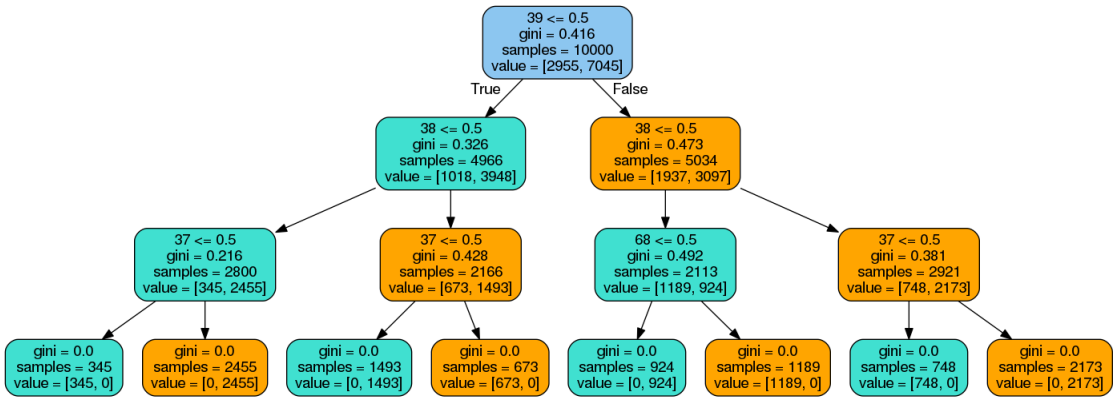

Figure 1: Represents the learned decision tree for an output variable, say 70. It will be stored as 70.png

Figure 1 represents the learned decision tree for an output variable. Each node has information about the decision variable, the impurity (Gini index) of the node, the remaining samples to divide, and the value array. In the value array, entry at value[0] (resp. 1) represents the number of samples with output label 0 (resp. 1). The left path (resp. right path) from a node represents

the decision variable being zero (resp. one). A candidate function will be a disjunction of all the paths with leaf node label 1. For the case of impure node, if the value[1] > value[0] then the path to that leaf node will also be considered. For Figure 1, the learned candidate function for variable 70 is $\psi_{70} = (\neg 39 \wedge \neg 38 \wedge \neg 37) \vee (\neg 39 \wedge 38 \wedge \neg 37) \vee (39 \wedge \neg 38 \wedge \neg 68) \vee (39 \wedge 38 \wedge 37)$, that is, if it encounters any of the path with the leaf node label 1, ψ_{70} will take value 1, else 0.

For matrix multiplication instances, Manthan will learn candidate function for C, P, X variables.

Repair Manthan repairs the candidate functions using an unsatcore-based approach. In some sense, with each repair iteration, the approximate candidate function vector moves closer the Skolem function vector. You can provide the maximum repair iteration using – *maxrepair* ⟨*int*⟩. If the repair iteration is not restricted, Manthan will terminate with a Skolem function vector. As an output Manthan returns a circuit in a verilog format, stored as *filename_skolem.v*. This circuit has X variables as input and Y as output. Each output variable y_i is assigned to ψ_i. For example, for matrix multiplication instances, the output for variable 19 is as follows:

```
assign i30 = i3 & i16;
assign i29 = i2 & i13;
assign n94 = ¬ i1 & ¬i29;
assign i28 = i1 & i10;
assign n96 = ¬ i29 & ¬ i28;
assign n97 = i29 & i28;
assign i57 = ¬ n96 & ¬ n97;
assign n99 = ¬ n94 & i57;
assign n100 = ¬ i30 & ¬ n99;
assign n101 = i13 & i28;
assign n102 = i2 & n101;
assign n103 = ¬ n96 & ¬ n102;
assign n104 = i30 & n103;
assign i19 = ¬ n100 & ¬ n104;
```

Variables starting with "n" are wired variables (intermediate variables), and the variables starting with "i" either belong to A, B or C, P, X.