Algorithm Learning

Learning Regular Expressions from Both Positive and Negative Examples

Learning regular expression from the set of strings that were matched by it is an easy problem with a trivial solution (just do a disjunction of the alternatives). The problem becomes slightly more involved if we insist that the learned expression has to be concise, in which case we call the subset construction algorithm and linear programming to our help. But what if cutting corners when trying to get rid of verbosity in our expression forces us to make assumptions about inputs that we have not seen? Perhaps we are even justified to do so, but what if we need to control this process for a handful of exceptions that we want the expression to be aware of?

Goal

To design an algorithm capable of learning regular expressions from both positive (strings that were matched) and negative examples (those not matched), given size and complexity constraints.

Structure

Solving the principal problem is not unlikely to turn out to be not too hard. Various possible extensions thus come to mind:

1. learning regular expressions from group match data;
2. learning finite-state transducers (FSTs);
3. translating the learned result to temporal logic.

Who is this for? Bachelor’s or master’s students interested in approaching the problem of learning regular expressions from the optimisation perspective. Familiarity with linear (integer) programming will be helpful but can also be picked up on the way.

Interested? Please reach out to us for more details.

Point of contact: Peter Belcak, email pbelcak@ethz.ch, ETZ G63.