Inspire at Inductive Logic Programming Competition:
Fine-grained Cost-based Hypothesis Generation*

Peter Schüller
Computer Engineering Department
Faculty of Engineering
Marmara University, Turkey
peter.schuller@marmara.edu.tr

Abstract. The Inspire system is an Inductive Logic Programming system based on an ASP encoding for generating hypotheses with a cost from the mode bias, and a transformation of hypotheses and examples to an ASP optimization problem that has the smallest hypothesis covering all examples as solutions. Inspire attempts to learn a hypothesis on single examples while increasing maximum hypothesis cost. As underlying ASP solver we use Clingo.

Disclaimer: for space reasons, this note only describes design decisions and the core idea of the Inspire system. Literature review, comparison and analysis of the system with other systems, and details about hypothesis cost, are omitted.

1 Design Decisions

Our system uses Answer Set Programming (ASP) [4–6, 9] for generating, optimizing, and verifying hypotheses.

Hypotheses are learned on an example-by-example basis, never from multiple examples. This is due to (i) low available resources (2 GB memory and 30 sec time), (ii) simplified handling of infinite relations like time/1 in the background knowledge, and (iii) because single given examples in the given example datasets were complex enough to fully specify a hypothesis.

Algorithm 1 shows the main idea of the Inspire system: the mode bias is used to blindly generate hypotheses using an ASP program, based on a fine-grained cost function and an incrementally increasing cost limit. Finding a hypothesis for a single example triggers an (efficient) test of the hypothesis over all examples and in case more examples are covered than by prior hypotheses, produces an output attempt for all test traces (unlimited attempts were permitted in the competition). The cost of hypotheses is parameterized as indicated in Figure 1.

Inspire is implemented in Python and uses ASP systems Gringo [7] and Clasp [8]. We use both unsat-core optimization [2] and stratification [1, 3].

* This work has been supported by Scientific and Technological Research Council of Turkey (TUBITAK) Grants 114E430 and 114E777.
Algorithm 1: INSPIRE-ILP(KB $bk$, Examples $e$, Mode-bias $m$, Test-traces $t$)

1. Sort $e$ by length of example trace // smaller examples first
2. $bestquality := 0$
3. for $c\text{limit} \in \{4, \ldots, 15\}$ do
   4. $hcand := \text{hypothesis rule candidates obtained from } \text{AS}(P_h(\text{costlimit} = c\text{limit}))$
   5. for $(\text{trace}, \text{label}) \in e$ do // go over sorted examples
      6. Find optimal $hr \subseteq hcand$ s.t. $\text{label} \in \text{AS}(bk \cup hr \cup \text{trace})$
      7. if $hr$ does not exist (unsat) then // cannot cover example
         8. Exit loop and continue with next $c\text{limit}$
      else
         9. $quality := |(\text{etrace}, \text{elabel}) \in e | \text{elabel} \in \text{AS}(bk \cup hr \cup \text{etrace})|$
         10. if $quality > bestquality$ then
             11. $bestquality := quality$
             12. Output $\#\text{attempt}$ // best-effort output
      13. for $t\text{trace} \in t$ do
         14. if $valid \in \text{AS}(bk \cup hr \cup t\text{trace})$ then Output VALID
         15. else Output INVALID
         16. if $quality = |e|$ then
            17. Exit program // on $e$ we predict correctly

2 Properties

The main idea of our approach is to obtain a fine-grained hypothesis spaces of slowly increasing complexity. Simple hypotheses are found fast, while all hypotheses have a chance to be found eventually.

The approach has the advantage that there is no need to make abduction of required facts, then induction of potential rules, then generalization of these rules, then a search for the smallest hypothesis (as done in the XHAIL [11] system) while the obvious disadvantage is, that hypothesis search is blind (similar as in the ILASP [10] system).

References