Predicate invention and reuse

Andrei Diaconu [1] , **Andrew Cropper**[1] , **Rolf Morel** [1]
[1]University of Oxford
{emails}@ox.ac.uk

Abstract

Abstract here.

1 Introduction

Inductive programming (IP) [Gulwani *et al.*, 2015] - also known as program synthesis or example based learning - is a field that lies at the intersection of several computer science topics (machine learning, artificial intelligence, algorithm design) and is a form of automatic programming. IP tries to find a target program starting with an incomplete specification which it tries to generalize. Usually, that incomplete specification is represented by examples, so we can informally define inductive programming to be the process of creating programs from examples using a limited amount of background information. Inductive logic programming (ILP) represents IP in the context of logic programming, the target language usually being Prolog. For example, a problem commonly solved by ILP system is planning the route of a robot in a maze. As suggested by Russel (...). In this paper we focus on two techniques, namely predicate invention and reuse, and investigate their. While the effectiveness of predicate invention has been documented, to the best of our knowledge there is no work that empirically demonstrates that predicate reuse is useful, nor any work discussing when it may be useful.

*** Talk about how reuse is not vital.

References

[Gulwani *et al.*, 2015] Sumit Gulwani, José Hernández-Orallo, Emanuel Kitzelmann, Stephen H. Muggleton, Ute Schmid, and Benjamin G. Zorn. Inductive programming meets the real world. *Commun. ACM*, 58(11):90–99, 2015.

[Inoue *et al.*, 2016] Katsumi Inoue, Hayato Ohwada, and Akihiro Yamamoto. Inductive logic programming: Challenges. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 4330–4332. AAAI Press, 2016.