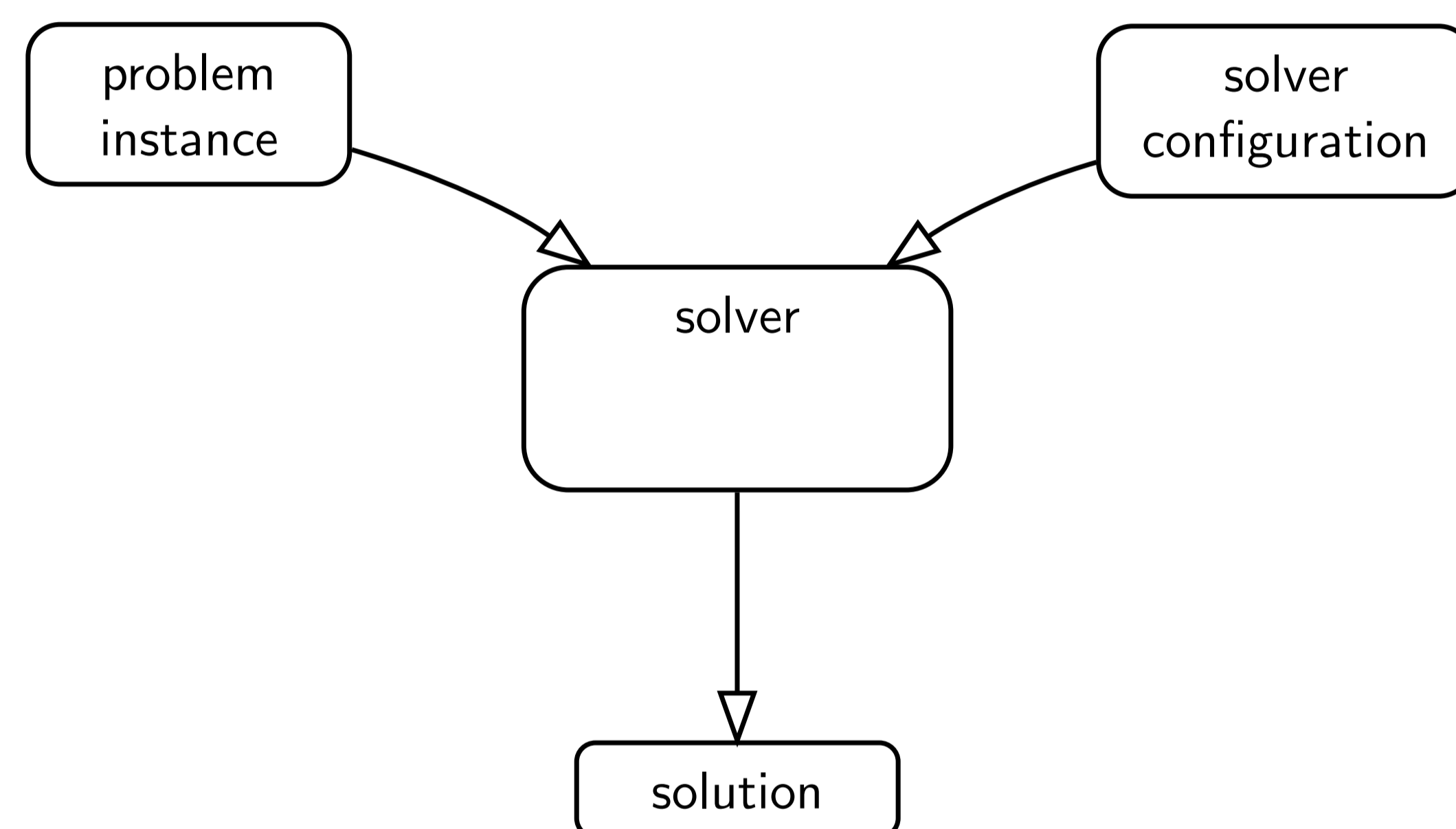


Lars Kotthoff (larsko@cs.st-andrews.ac.uk)

Work supervised by Ian Miguel and Ian Gent. Lars Kotthoff is supported by a SICSA prize studentship.

Algorithm selection problem

- ▷ given a problem and a set of algorithms, select the one which will perform best [7]
- ▷ instead of a general problem solver with a **fixed** configuration, **reconfigure** the solver for each problem instance

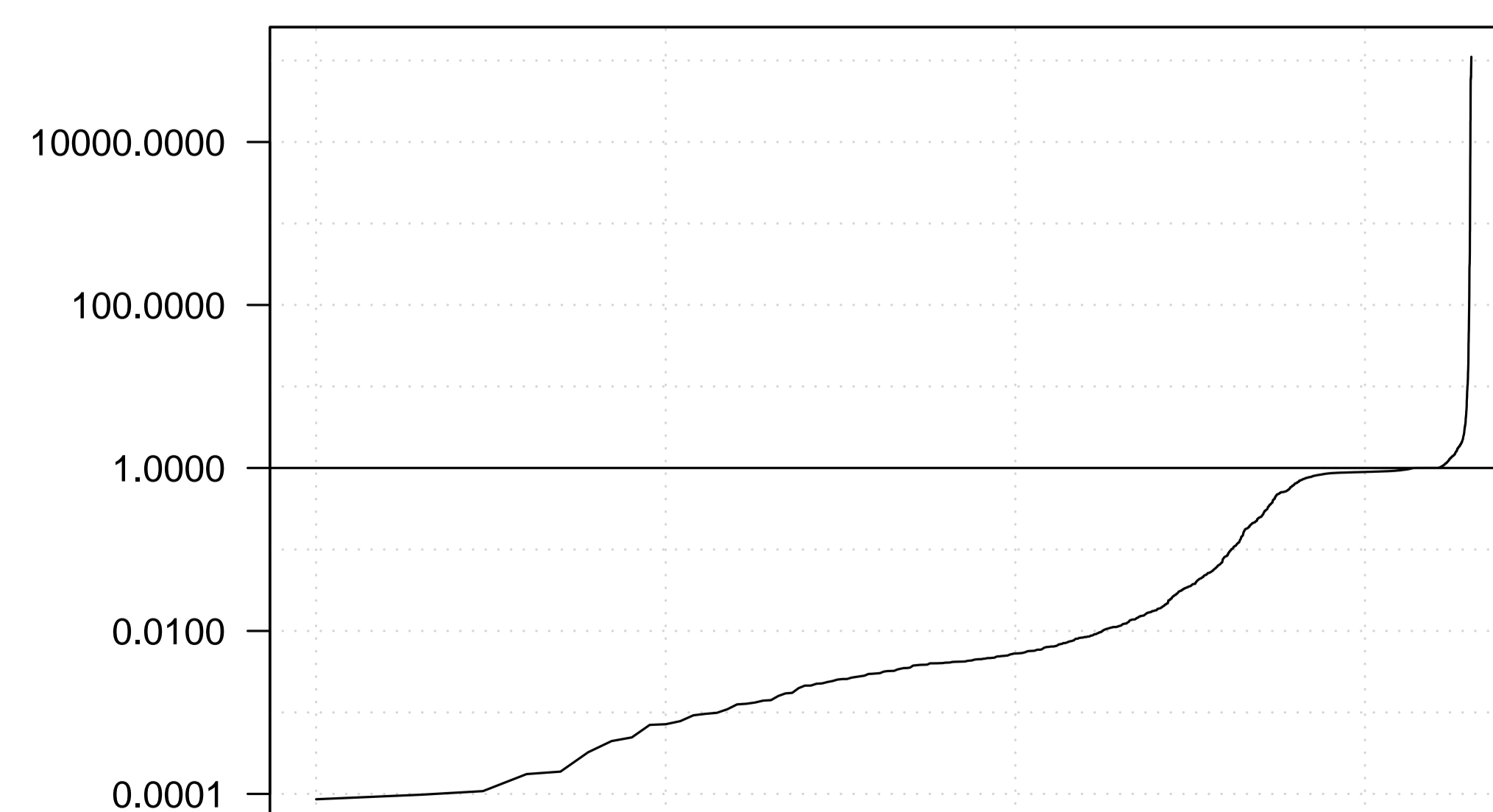


- ▷ need to **choose** which configuration to use
- looks like a **machine learning** problem

Motivation

- ▷ for many practical applications, we can choose a **sensible default**
- ▷ will **perform well** in most cases
- ▷ **bad performance** in some cases
- ▷ **huge gains** to be had by switching in a few cases

speedup over default variant

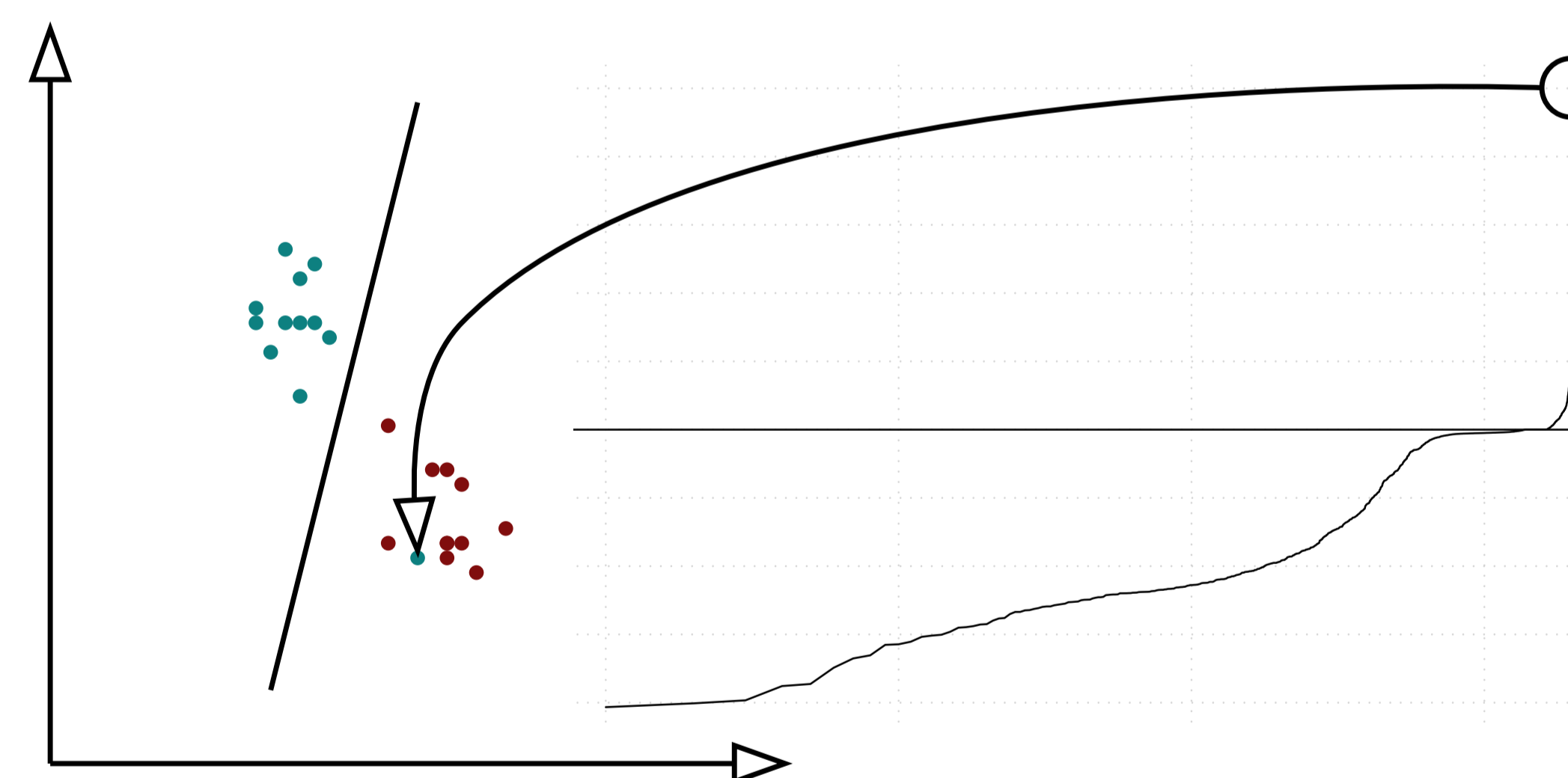


Machine learning classification

- ▷ given a set of training **problems**, **features** for each problem and a **classification** label, learn how the features affect the classification
- ▷ construct a **classifier** that, given an unknown problem, predicts the label
- ▷ performance of a classifier is characterised by the **accuracy**, i.e. how many problems are classified correctly and how many are misclassified

Machine learning for algorithm selection

- ▷ standard approaches yield **poor performance**
- ▷ normal performance metric **unsuitable**
- ▷ problems where the **potential gain** of selecting the best algorithm is large need to be classified correctly
- ▷ don't care about problems where the difference between the different algorithms is small



- ▷ left side shows problems and the classifier separating them into classes
- ▷ right side shows potential gain
- ▷ only one problem misclassified, but this is the one with the **highest gain**
- ▷ correctly classified problems don't actually matter because the gain is **minimal**
- ▷ this makes the learned classifier essentially **useless**
- need **different approach** for training and evaluating classifiers

Classification revisited

- ▷ consider **misclassification penalty** for evaluating classifier performance – how much more time do we have to spend because of selecting a suboptimal algorithm [8]?
- ▷ consider **maximum misclassified penalty** for learning a classifier
- ▷ initial results **promising** [3, 4]
- ▷ depending on the algorithm selection problem, using a dynamic reconfiguration approach can improve **orders of magnitude** over “sensible default” approach

Algorithm selection revisited

- ▷ which **machine learning algorithm** is suitable for this?
- ▷ which **parameters** for an algorithm should be chosen [6]?
- ▷ not all take **instance weights** into account
- ▷ **ensemble classification** [2] as a temporary solution – combine the decisions of many different classifiers
- ▷ shown to **improve** robustness and sometimes overall results of classification [5]
- ▷ new **challenges** – how to combine predictions?
- ▷ simple **voting** appears to be promising [1]

References

- [1] Eric Bauer and Ron Kohavi, *An empirical comparison of voting classification algorithms: Bagging, boosting, and variants*, Mach. Learn. 36 (1999), no. 1-2, 105–139.
- [2] Thomas G. Dietterich, *Ensemble methods in machine learning*, Proceedings of the First International Workshop on Multiple Classifier Systems, Lecture Notes In Computer Science, vol. 1857, Springer-Verlag, 2000, pp. 1–15.
- [3] Ian P. Gent, Chris Jefferson, Lars Kotthoff, Ian Miguel, Neil C.A. Moore, Peter Nightingale, and Karen Petrie, *Learning when to use lazy learning in constraint solving*, ECAI, 2010.
- [4] Ian P. Gent, Lars Kotthoff, and Ian Miguel, *Using machine learning to make constraint solver implementation decisions*, SICSA PhD conference, 2010.
- [5] Lars Kotthoff, Ian Miguel, and Peter Nightingale, *Ensemble classification for constraint solver configuration*, submitted to CP, 2010.
- [6] N. Lavesson and P. Davidsson, *Quantifying the impact of learning algorithm parameter tuning*, AAI, 2006, pp. 395–400.
- [7] John R. Rice, *The algorithm selection problem*, Advances in Computers 15 (1976), 65–118.
- [8] Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown, *SATzilla: portfolio-based algorithm selection for SAT*, J. Artif. Intell. Res. (JAIR) 32 (2008), 565–606.