Using Machine Learning to Focus Iterative Optimization

Best Presentation Award at ACM/IEEE Code Generation and Optimization 2006

Felix Agakov, Edwin Bonilla, John Cavazos, Bjoern Franke, Michael O'Boyle, John Thomson, Marc Toussaint, Chris Williams

`John.Thomson@ed.ac.uk`

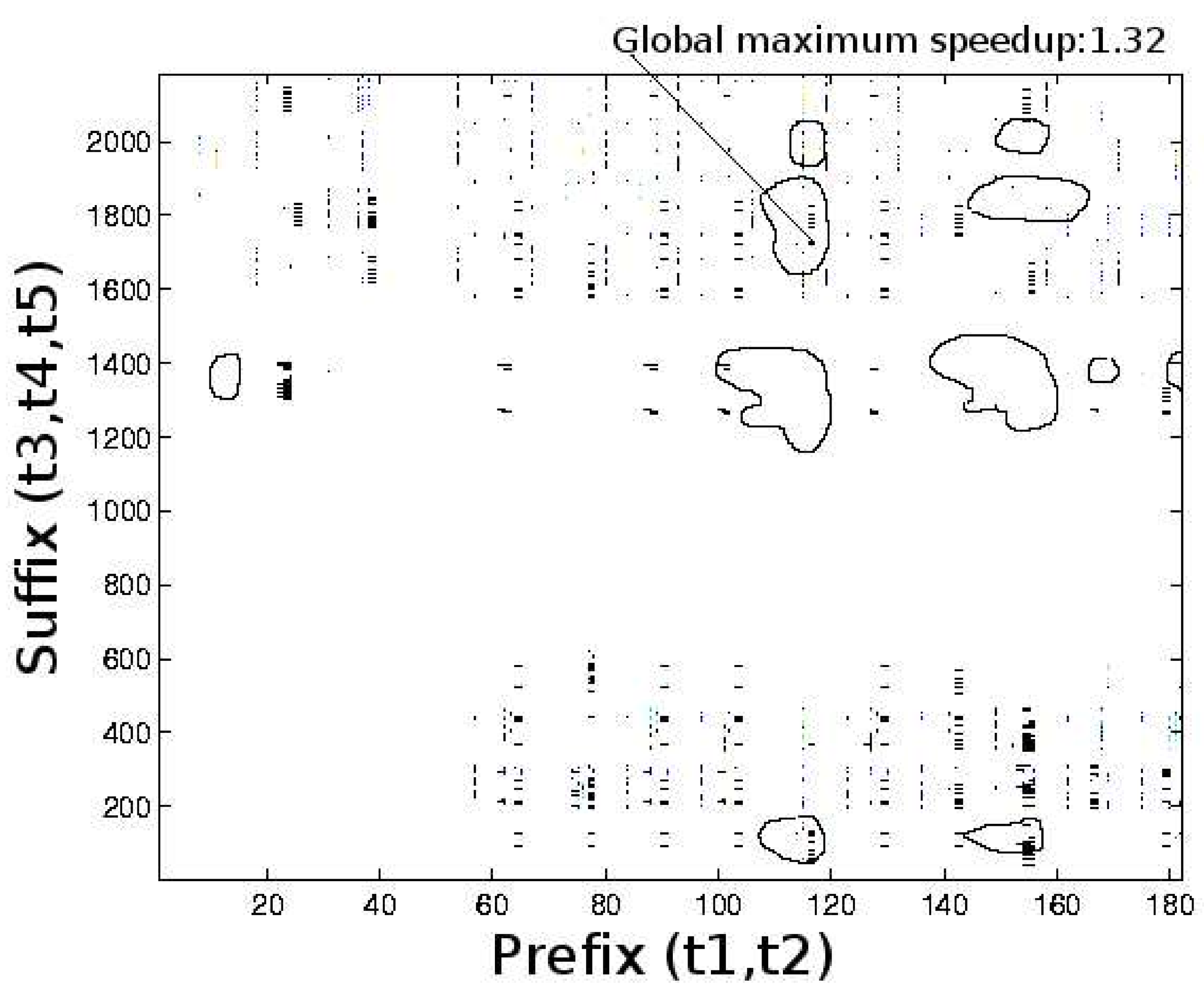
`http://www.anc.ed.ac.uk/machine-learning/colo/`

Introduction

Iterative compiler optimization has been shown to outperform static approaches, but at the cost of large numbers of evaluations of the program. This work develops a new methodology to reduce this number and hence speed up iterative optimization, using predictive modelling.

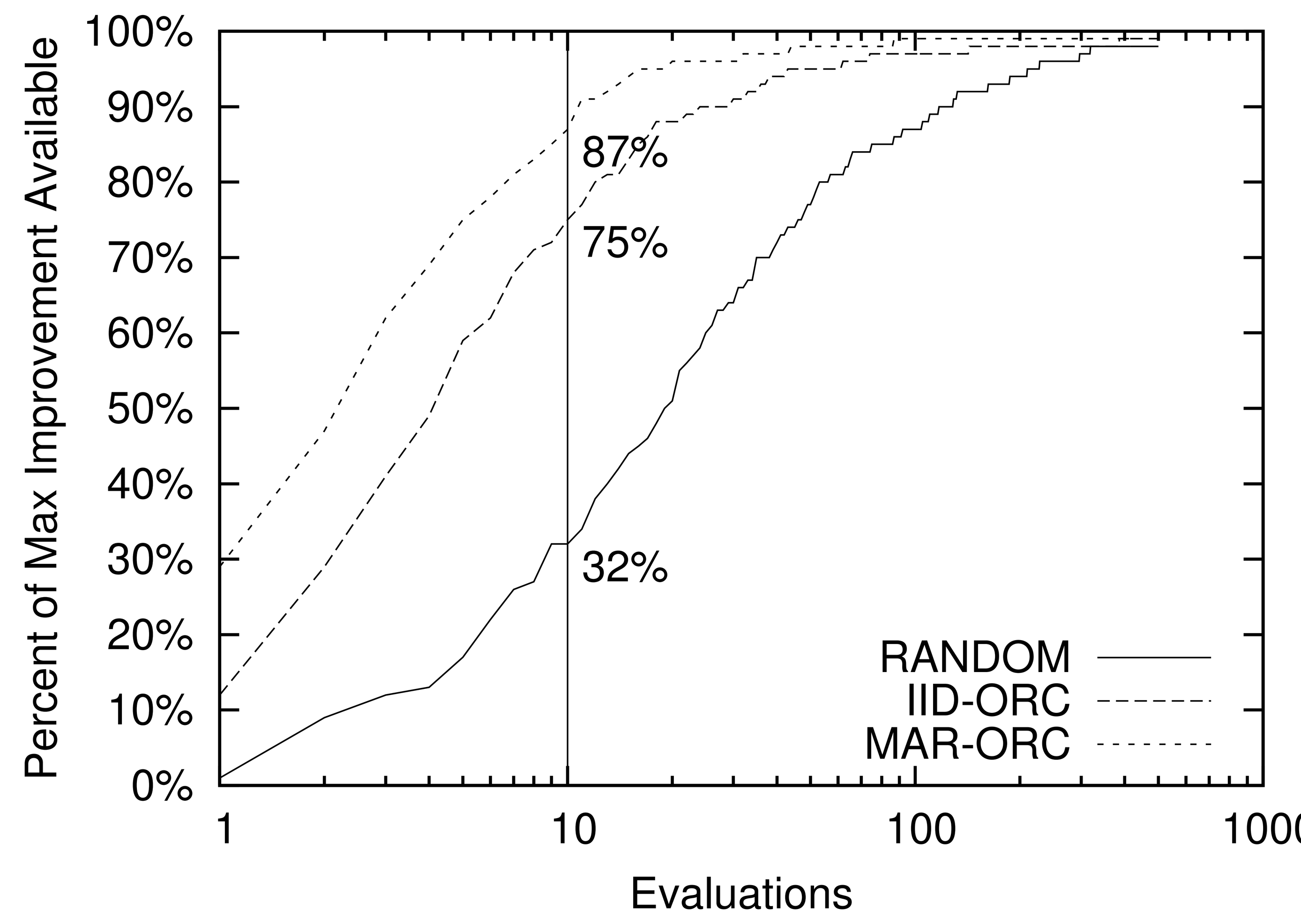
Off-line, a training set of programs is iteratively evaluated and the shape of the spaces and program features are modelled, This is used to automatically focus search on profitable areas for new, unseen, programs.

We show that such learnt models can speed up iterative search on large spaces by an order of magnitude. This translates into an average speedup of 1.26 on the TI C6713 and 1.27 on the AMD Au1500, the two embedded processors used, in just 2 evaluations.



A Highly Non-linear Space – Points corresponding to those transformation sequences whose performance is within 5 per cent of the optimum for adpcm *on the TI C6713. The contour is the predicted area for good optimizations.*

Models



TI C6713: Random search versus IID-oracle and Markov oracle.

We construct models using previously results from an exhaustive enumeration of a 14^5 transformation space. We employ two types of model in this work:

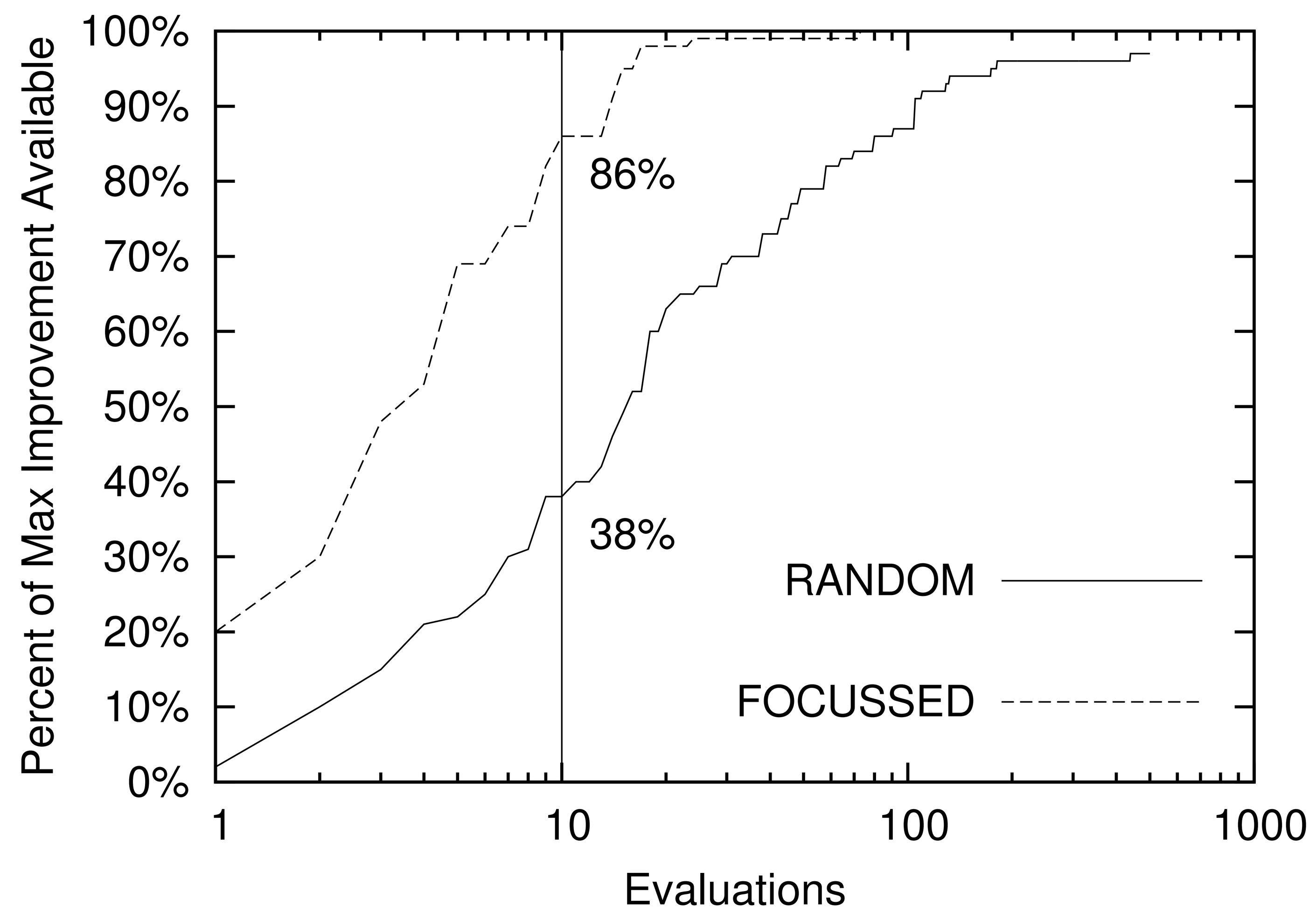
- Independent identically distributed (IID) model
  An IID model is a very simple model which assumes that all transformations are mutually independent, neglecting the effect of interactions among transformations.

$$P(s_1, s_2, \dots, s_L) = \prod_{i=1}^{L} P(s_i). \qquad (1)$$

- Markov model
  In this highly non-linear space, there can exist transformations which only yield good performance when a different transformation has already been applied. We can attempt to capture this using a Markov model, in which the probability of a transformation being applied depends on the transforamtions which have been applied before.

$$P(\mathbf{s}) = P(s_1) \prod_{i=2}^{L} P(s_i | s_{i-1}).$$

Evaluation



How close to the best performance random and focused search achieve for each program evaluation for adpcm *on the TI C6713*

TI	2 Evaluations			5 Evaluations			10 Evaluations			50 Evaluations		
Benchmark	R	M	I	R	M	I	R	M	I	R	M	I
fir	1.18	1.66	1.67	1.25	1.66	1.83	1.37	1.66	1.85	1.70	1.85	1.85
iir	1.14	1.20	1.19	1.18	1.23	1.19	1.19	1.23	1.21	1.19	1.23	1.23
adpm	1.08	1.33	1.17	1.18	1.33	1.18	1.25	1.35	1.24	1.28	1.43	1.28
com	1.10	1.25	1.26	1.15	1.26	1.30	1.19	1.27	1.32	1.29	1.32	1.35
edg	1.08	1.13	1.27	1.15	1.13	1.28	1.21	1.13	1.28	1.25	1.13	1.29
lpc	1.09	1.05	1.13	1.10	1.05	1.16	1.10	1.10	1.18	1.24	1.12	1.27
spe	1.01	1.10	1.15	1.03	1.17	1.16	1.05	1.17	1.16	1.07	1.17	1.18
AVG	1.10	1.25	1.26	1.15	1.26	**1.30**	1.19	1.27	1.32	**1.29**	1.32	1.35

Results on a larger space of 82^{20}. Excellent results on a much larger space show the scalability of the technique, allowing for practical use of iterative compilation.

This work is part funded by the EPSRC COLO grant, EU MilePost STREP and EU HiPEAC NoE.