Using Machine Learning to Focus Iterative Optimization

F. Agakov, E. Bonilla, J.Cavazos, B.Franke, G. Fursin,
M.F.P. O'Boyle, J. Thomson, M. Toussaint, C.K.I. Williams

Institute of Computer Systems Architecture
Institute of Adaptive and Neural Computing

School of Informatics
University of Edinburgh
UK

March, 2006

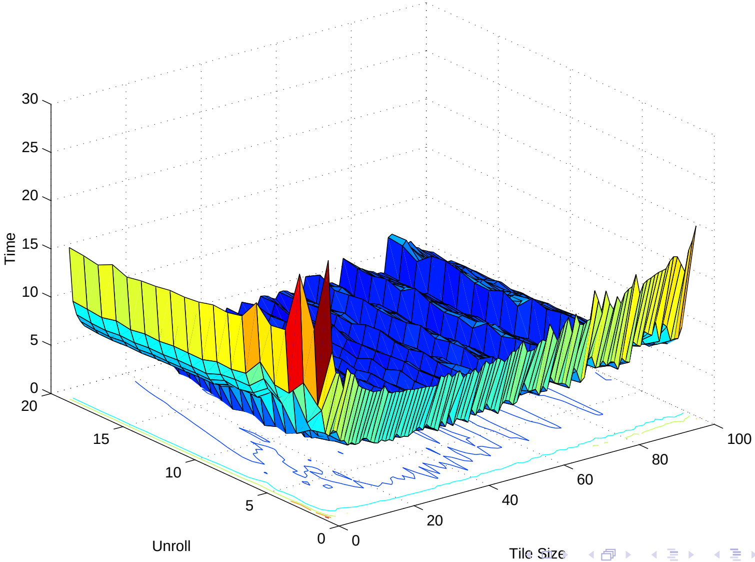
Overview

- Background + Motivation
  - Embedded applications where performance is critical
  - Using predictive modelling to guide search/global optimisation
- Models to focus search
  - Examined two standard search algorithms: Random + GA
  - Propose two models: IID and Markov to focus search
  - Learning model using nearest neighbour classification
- Evaluation
  - An exhaustively enumerated small space 14^5
  - A large test space 80^{20}
- Summary and Future work

Focused Iterative Search: Background

- Compilers are unable to effectively exploit hardware resources
  - Fundamentally this is due to the complexity of the architecture
- Static analysis based approaches try to model the space with simple models/heuristic on a piecemeal basis
  - Experiments show that the optimisation space is massively non-linear.
  - Furthermore architectures evolve faster compiler writers can react
- Try a new approach iterative compilation: try different optimisations, run them - select the best.

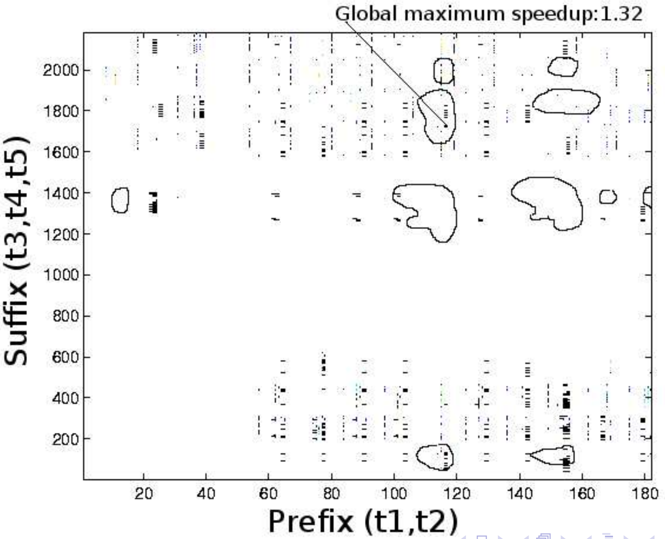
UltraSparc for mm $N = 512$.



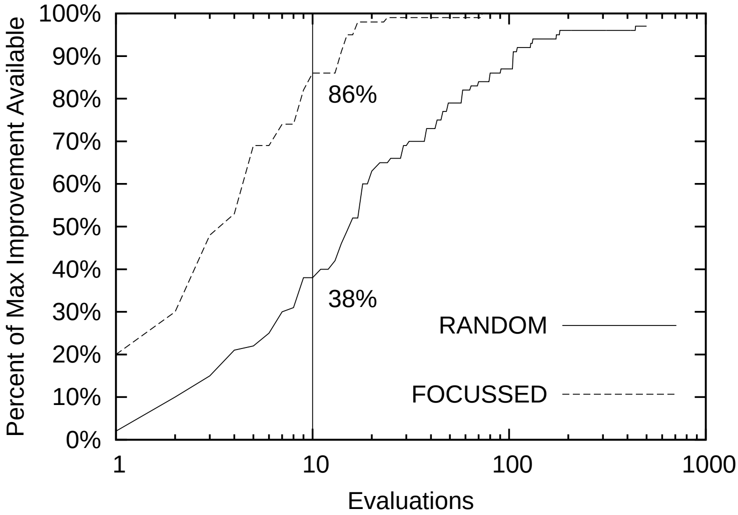
Focused Iterative Search: Motivation

- Iterative compilation is now a well known technique:
  - Search the space using random, GA, hill climbing techniques
  - It gives good results but takes a long time -a barrier to use in general purpose setting
- Basic idea is focus search on areas of space likely to be good
- We determine these areas by learning form other programs.
  - So, if my program A is similar to previously searched program B, can I use knowledge of its space to focus my search?

Focus reduces search: Adpcm on TI C6713



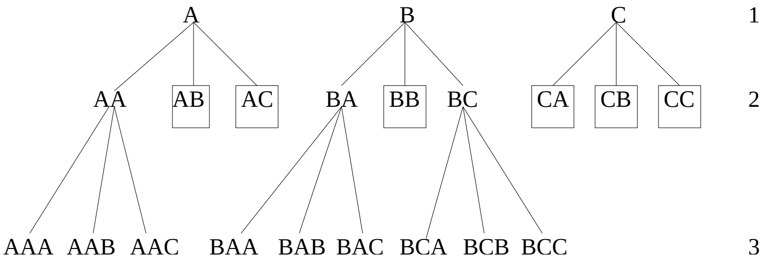
How learning helps: TI C6713



Optimization Space 14^5

- Embedded system application
  - UTDSP benchmarks: Compute intensive DSP
  - **AMD** Au1500 - gcc 3.2.1 -O3, **TI** C6713 v2.21-O3
- Exhaustively enumerated an interesting search space
  - 14 transformations selected.
  - All combinations of length 5 evaluated
- Allows comparison of techniques
  - How near the minima each technique approaches
  - Rate of improvement
  - Characterization of the space

Generating an Exhaustive Space

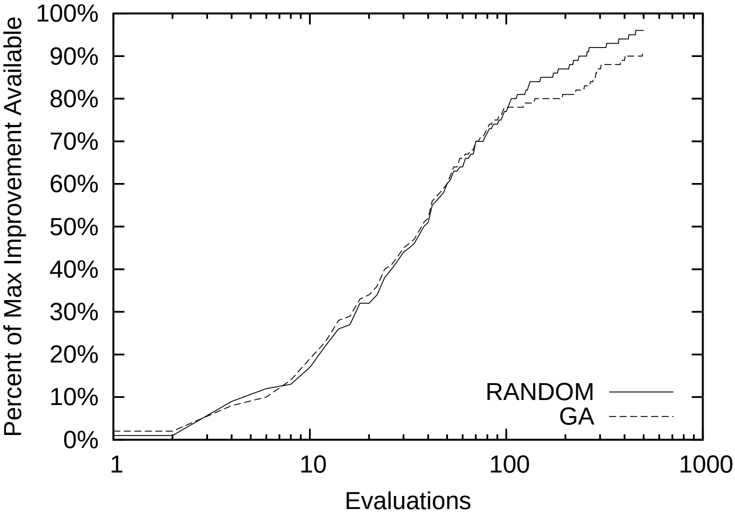


- Generate length 1 to 5 in order.
- If $ST = S$, record and prune subtree

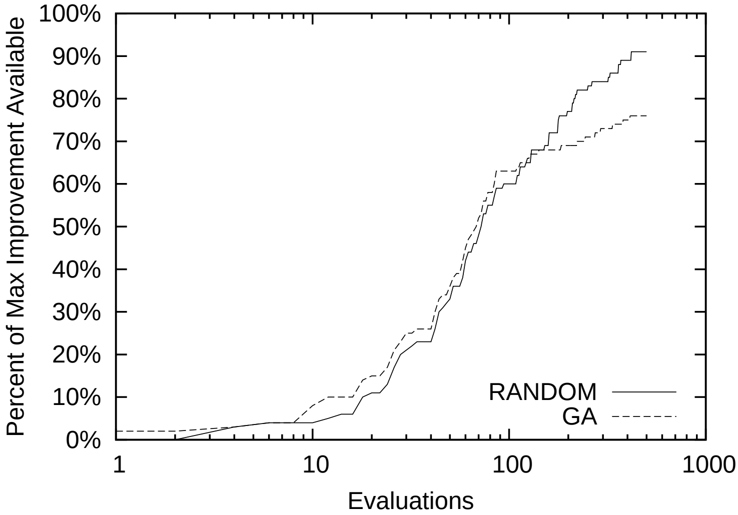
Exhaustive enumeration: 14^5

	TI		AMD	
Prog.	**Improv.**	**Seq.**	**Improv.**	**Seq.**
fft	3.64%	{3nm}	4.49%	{4hns}
fir	45.5%	{4}	26.7%	{3}
iir	16.3%	{3h}	29.5%	{h4}
latnrm	0.34%	{nsch}	27.1%	{csh4}
lmsfir	0.39%	{1s}	30.3%	{s3}
mult	0.00%	{}	30.5%	{4}
adpcm	24.0%	{1ish}	0.75%	{ism}
compress	39.1%	{4s}	24.0%	{hs4}
edge	5.06%	{3}	23.1%	{ch4}
histogram	0.00%	{}	24.7%	{4}
lpc	10.7%	{sn2}	6.01%	{h4cnm}
spectral	7.46%	{n4}	8.53%	{sh4}
Average	12.7%	-	13.8%	-

How does blind search perform? TI



How does blind search perform? AMD



Two models to help focus search

We want models that summarise the space that

- Can be applied to similar programs to focus search
- Are cheap to learn and don't overfit

Examine two basic models

- Identically independent distribution - very naive
    - Just note how often a transformation occurs in a good sequence
- Markov model - slightly smarter.
    - Considers limited interactions

Models: IID and Markov

- IID: Does not consider interactions

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

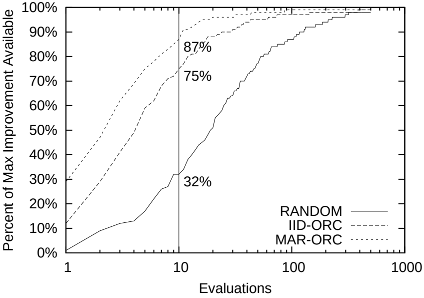
- Markov: Considers previous transformation. Not location aware

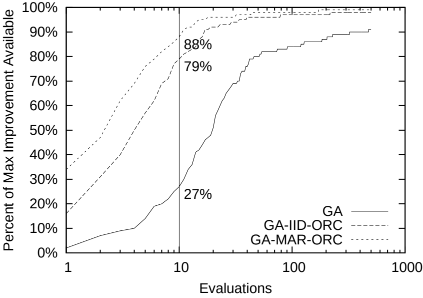
$$P(s_1, s_2, \ldots, s_L) = P(s_1) \prod_{i=2}^{L} P(s_i | s_{i-1}).$$

Models as oracles

- Want to check they are useful before trying to learn them
  - So we exhaustively enumerated space to learn each model
  - 14 transformations upto 5 in length - 14^5
- IID has a 14 element vector.
  - One probability value per transformation
- Markov a 14×14 matrix.
  - For each transformation what is the probability of the next one.
- Used the model of the space to guide search of this space as a sanity check
  - Similar to hardware oracles

Oracle vs blind search



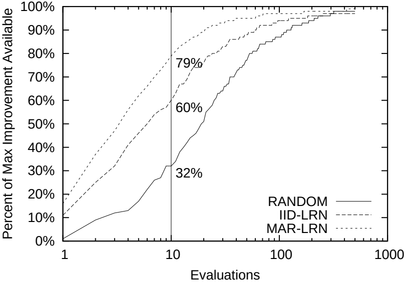


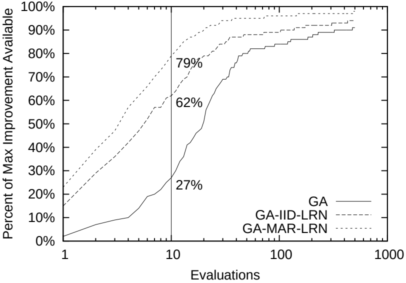
Average results on TI. Markov better than IID

Learning

- Used over 30 features to characterise programs
  - Then PCA to see which were relevant
  - Reduced to 5
- Used nearest neighbour as learning mechanism
- Evaluated mechanism on small and large 80^{20} space
  - Learnt on large space using 1000 training examples
- Compared against random over first 50 evaluations.

Performance on small space





TI: Markov is best

Performance on large space

	2 Evaluations			5 Evaluations			50 Evaluations		
	R	M	I	R	M	I	R	M	I
TI	1.10	1.25	1.26	1.15	1.26	**1.30**	**1.29**	1.32	1.35
AMD	1.08	1.24	1.27	1.17	**1.33**	1.31	**1.32**	1.41	1.44

- Significant improvement in first 2 evaluations (1.26,1,27)
  - R = Random, M = Markov, I = IID
- Focus gives an order of magnitude improvement.
  - Greater performance after 5 evaluations vs 50 of random
- IID outperforms Markov on large space
  - Learning an 80 element vector vs 80 $\times$ 80 matrix with 1000 samples

Summary and Future Work

- Learning models to focus search works
  - More sophisticated models need more training data
- For continuous optimisation switch models as certain point
- Can be used with other work to reduce cost of each evaluation.
  - Automatically choose the space for self-tuning
- Ultimate goal is to use ML to make iterative compilation as cheap as profile-directed schemes

Additional Material

- Best single transfromation across 14^5
  - `himc3` on AMD - speedup 1.11
  - No single best on TI
- Effective space: measure of pruning
  - Varies: 0.85% on histogram, 15.4%
- PCA: 5 vectors of 26 wieghts. Account for over 95% of variance
  - No clear feature dominating.
  - Loop structure important

Nearest Neighbour

Benchmark	Nearest Neighbor
fft	lpc
fir	compress
iir	fir
latnrm	iir
lmsfir	iir
mult	compress
adpcm	fir
compress	fir
edge	iir
histogram	fir
lpc	spectral
spectral	lpc