The Predictability of Data Values

Yiannakis Sazeides and James E. Smith
Department of Electrical and Computer Engineering
University of Wisconsin-Madison

yanos@ece.wisc.edu, jes@ece.wisc.edu

Micro-30, December 2, 1997

Introduction

- Use Prediction to overcome Dependences
- A variety of program information can be predicted (branches, addresses, data values, dependences)

  Branch prediction receives most attention

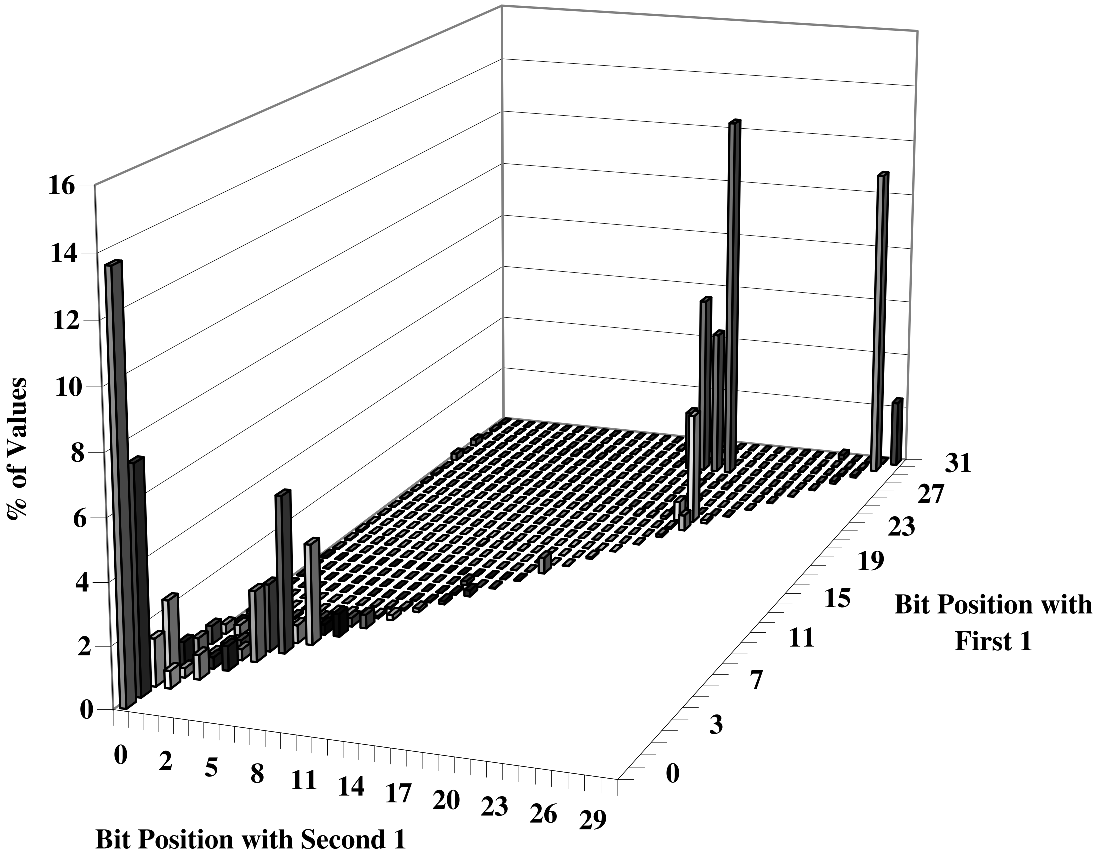
  Also important to predict *Data Values*
- Is it possible? Large range of values not 0/1

  Values exhibit “locality” (Lipasti AsplosVII)
- This talk: *Data Value Predictability*

  Framework for studying value prediction

  Simulation results, idealized study

Motivation



- Value space is very sparse. Predictable?

Value Sequences & Prediction Models

- Informal Classification of Value Sequences:

Constant (C)	5 5 5 5 5 5 5 ...
Stride (S)	1 2 3 4 5 6 7 8 ...
Non-Stride (NS)	28 -13 -99 107 23 456 ...

- Important sequences are formed by composing stride and non-stride sequences:

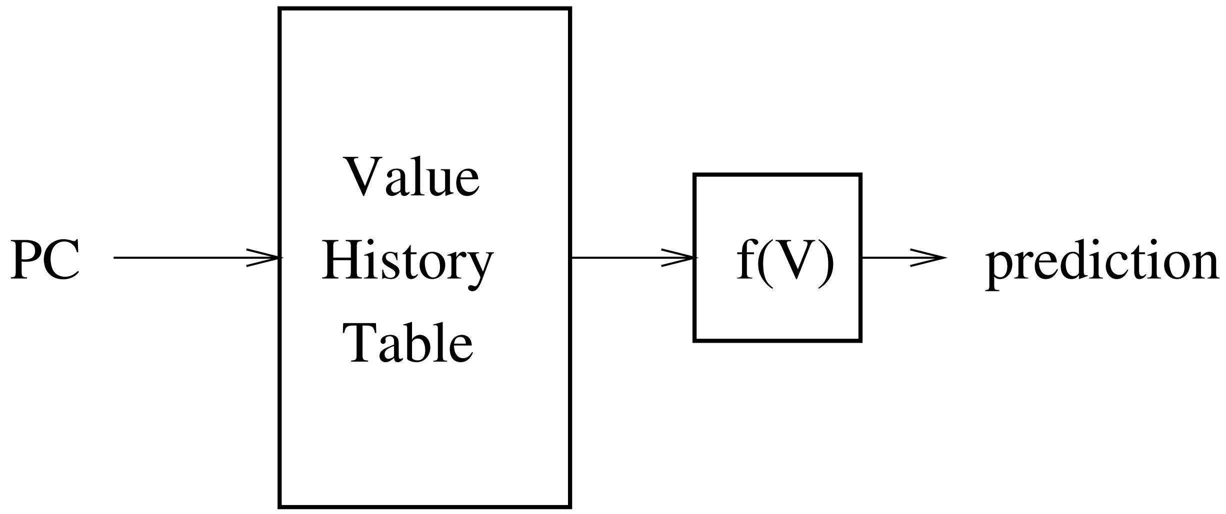
Repeated Stride (RS)	1 2 3 1 2 3 1 2 3 ...
Repeated Non-Stride (RNS)	1 -13 9 17 1 -13 9 17 ...

- Two types of prediction models:

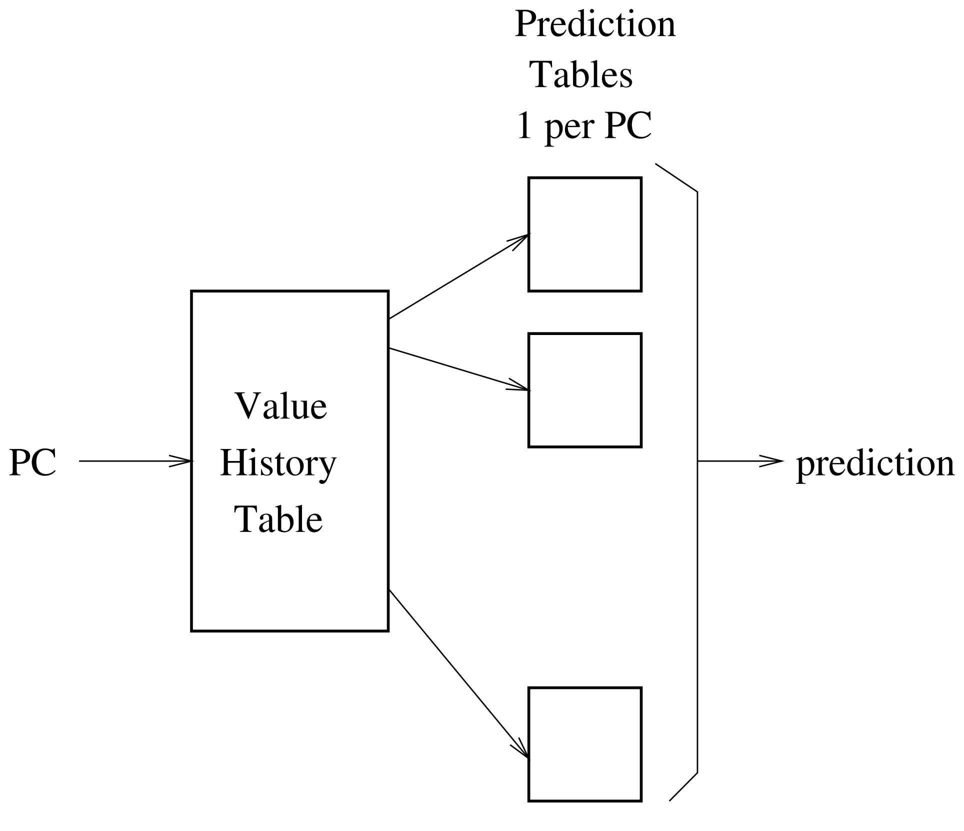
  Computational predictors make a prediction by performing a computation on previous values

  Context based predictors learn the value(s) that follow a particular *context* and predict one of the values when the same context repeats

Computational Predictors



- **Last Value Predictors** if previous value is **v** then prediction is **v**
- **Stride Predictors** if $\mathbf{v}_{n-1}$ and $\mathbf{v}_{n-2}$ are the two most recent values, then the predictor computes $\mathbf{v}_{n-1}$ **+ (**$\mathbf{v}_{n-1}$ **-** $\mathbf{v}_{n-2}$**)**
- **Replacement hysteresis**
  Saturating counters, 2-delta



- **Finite Context Method Predictors (fcm)**
  predict the next value based on a finite number of preceding values

- An **order k** fcm predictor uses k preceding values

Sequence: a a a b c a a a b c a a a ?

0th order Model

a	b	c
9	2	2

Prediction: a

1st order Model

	a	b	c
a	6	2	0
b	0	0	2
c	2	0	0

Prediction: a

2nd order Model

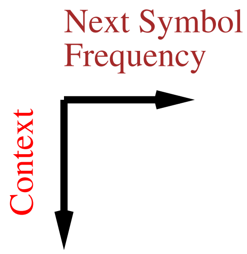
	a	b	c
a a	3	2	0
a b	0	0	2
a c	0	0	0
b a	0	0	0
b b	0	0	0
b c	2	0	0
c a	2	0	0
c b	0	0	0
c c	0	0	0

Prediction: a

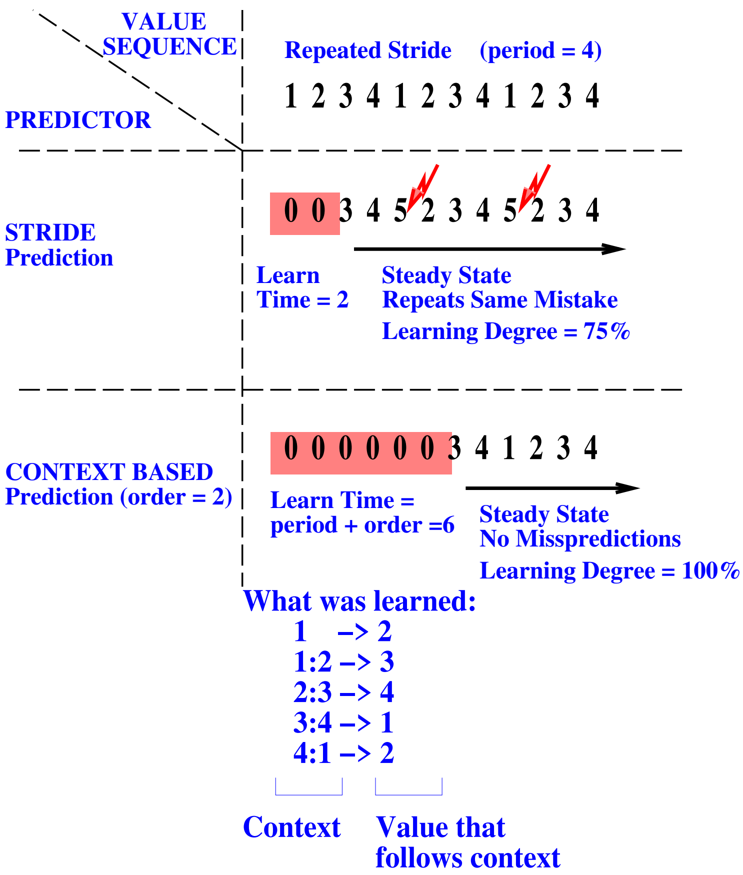
3rd order Model

	a	b	c
a a a	0	2	0
a a b	0	0	2
a b c	2	0	0
b c a	2	0	0
c a a	2	0	0

Prediction: b



- The combination of more than one prediction model is known as *blending*



- Computation learns faster
- Context learns *better*

Simulation Methodology

- Idealized Performance Study

- Three value predictors are considered
  - Last Value, (Lipasti ASPLOS VII)
  - Stride 2-delta, (Eickemeyer IBM R&D, 7/93)
  - Fcm order 1, 2 and 3

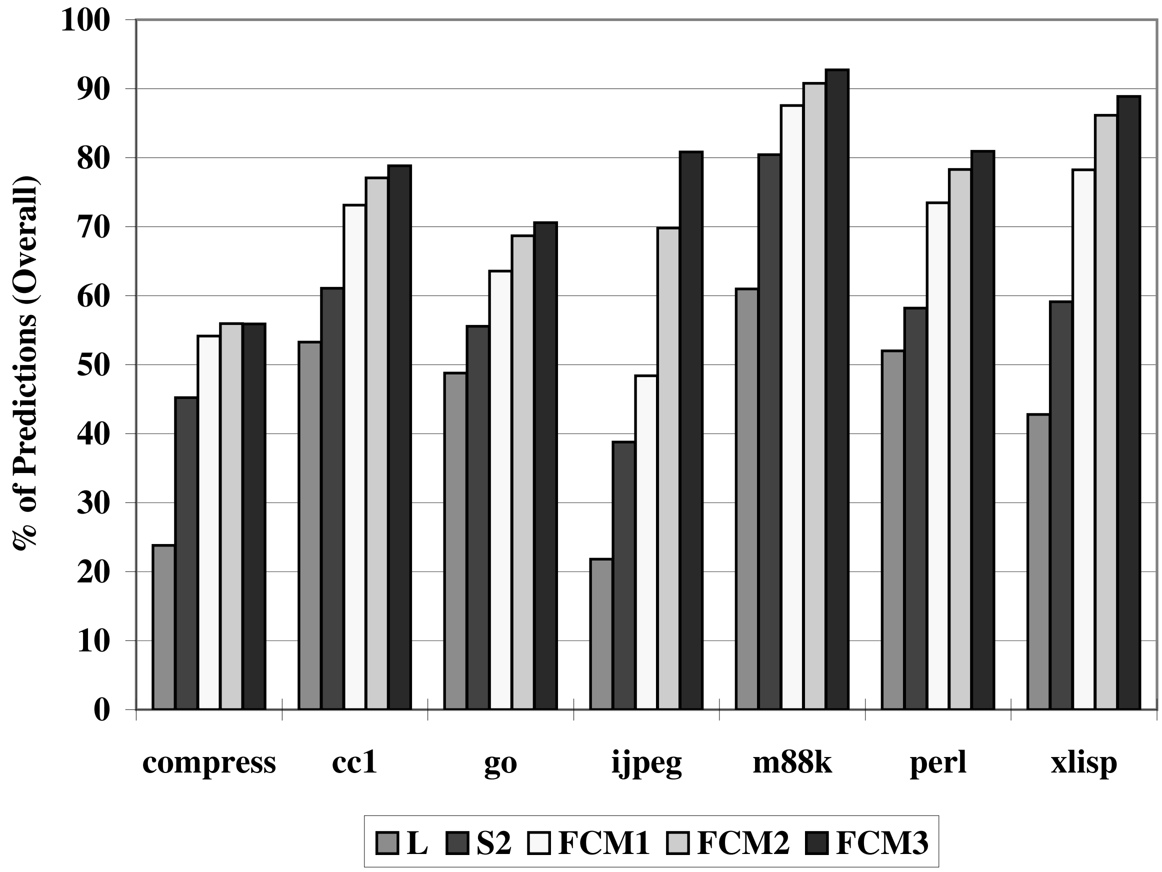
- Fcm predictor uses full concatenation of history values and blending

- Predictors accessed based on PC only

- No table aliasing

- Trace driven simulation SPECINT95

Predictability

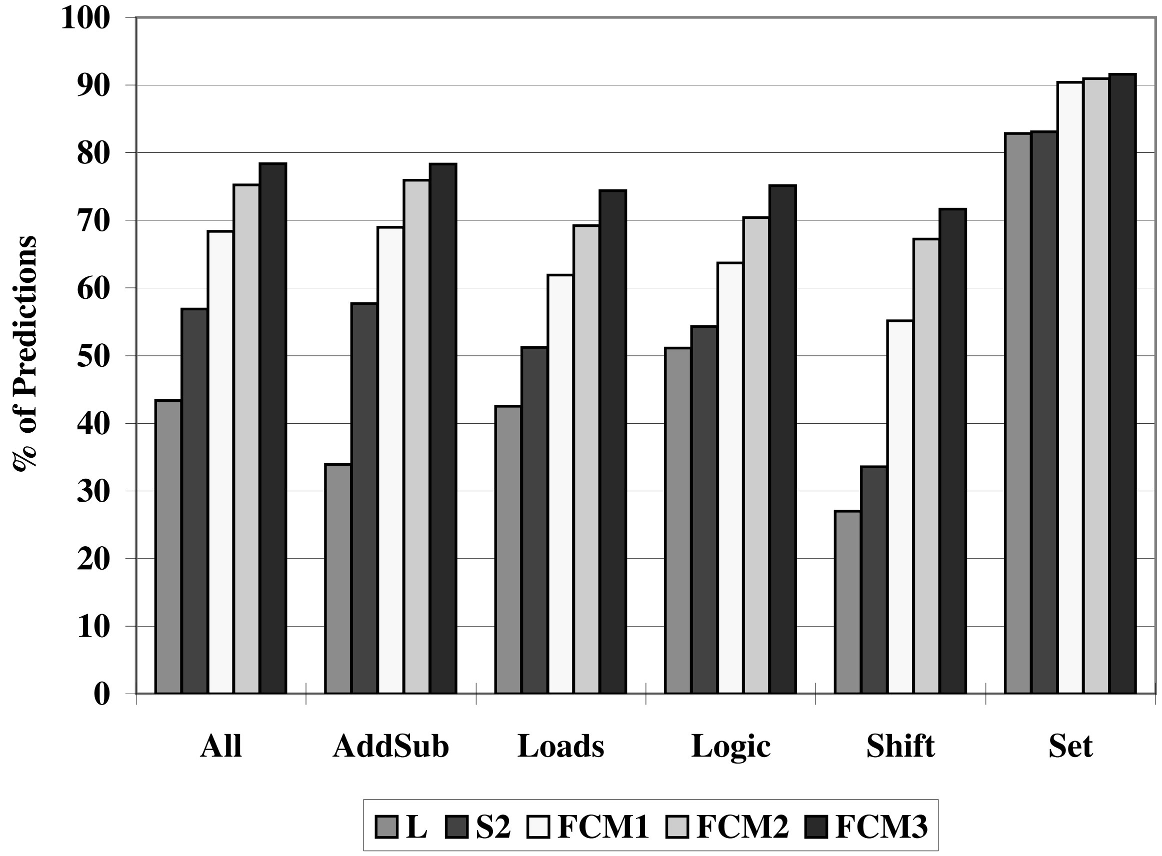


- Last Value < Stride < FCM

- Few previous values sufficient to predict well

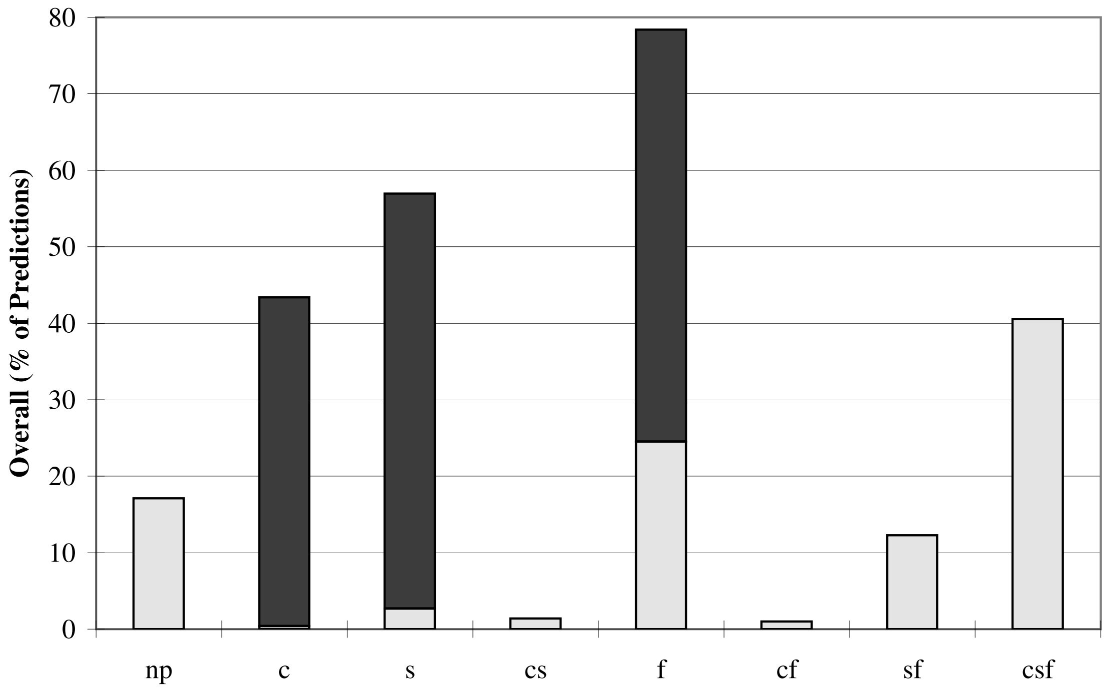
- Fcm improves accuracy with increasing order – however diminishing returns

Predictability, cntd



- Computational prediction varies significantly among instruction types of the same benchmark
- Fcm performance varies less – ability to capture any repeating sequence
- Stride does very well for add/subtract – predictor matches operation of predicted instruction. Generalize such an approach?

Correlation of Predicted Sets

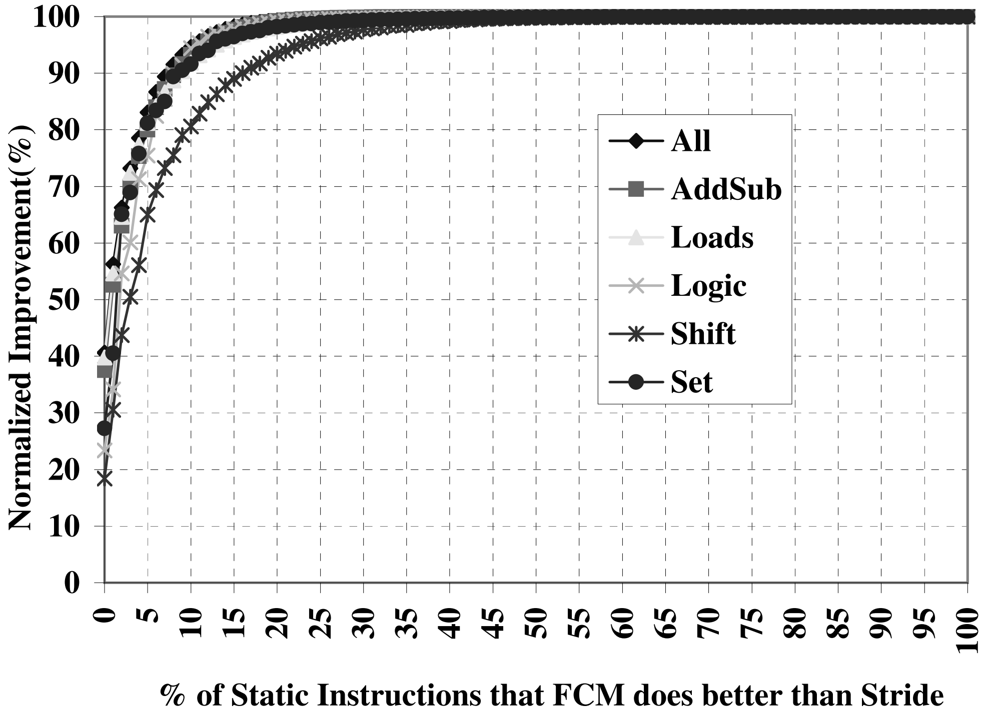


- A small number, close to 18%, of values are not predicted correctly by any predictor

- A significant fraction, over 20%, of correct predictions is only captured by fcm

- A large portion, around 40%, of correct predictions is captured by all predictors

Context Based vs Stride

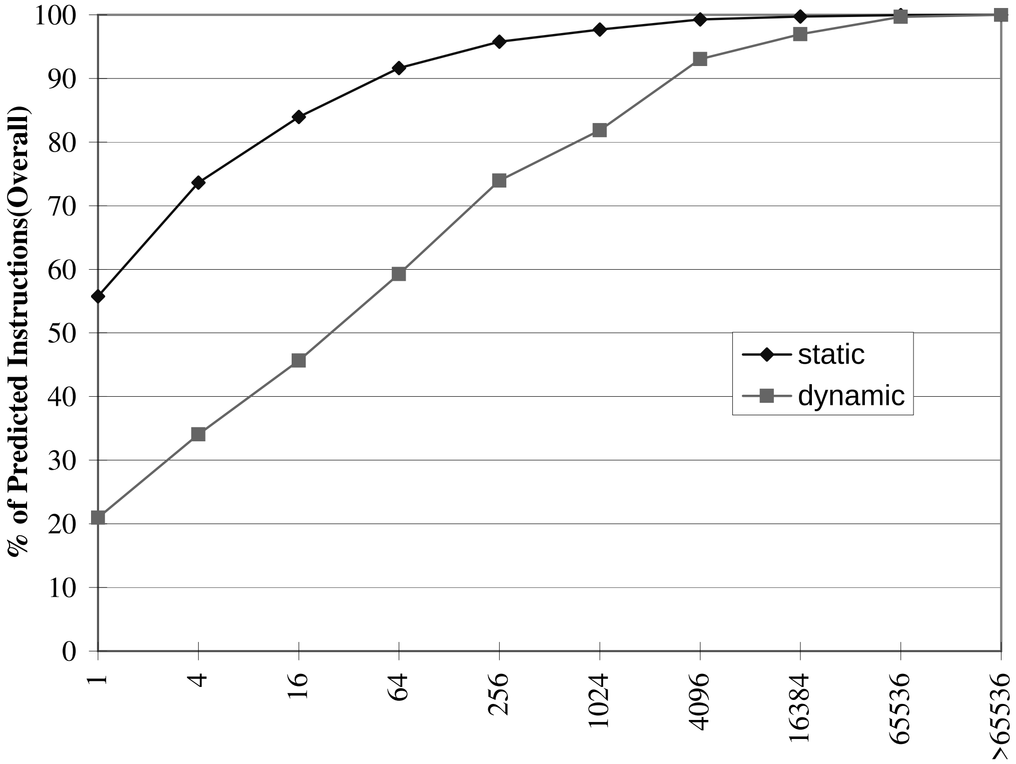


- About 10% of the static instructions account for about 90% of the total improvement

- A hybrid fcm-stride predictor with choosing may be a good approach.

- Different types of instructions have similar behavior

Value Characteristics



- A large number, $\geq$50%, of static instructions generate only one value

- The majority, $\geq$50%, of dynamic instructions correspond to static instructions that generate fewer than 64 values

Sensitivity to Input Data and Flags

- Input Data

File	Predictions (mil)	Correct (%)
jump.i	106	76.5
emit-rtl.i	114	76.0
gcc.i	137	77.1
recog.i	192	78.6
stmt.i	372	77.8

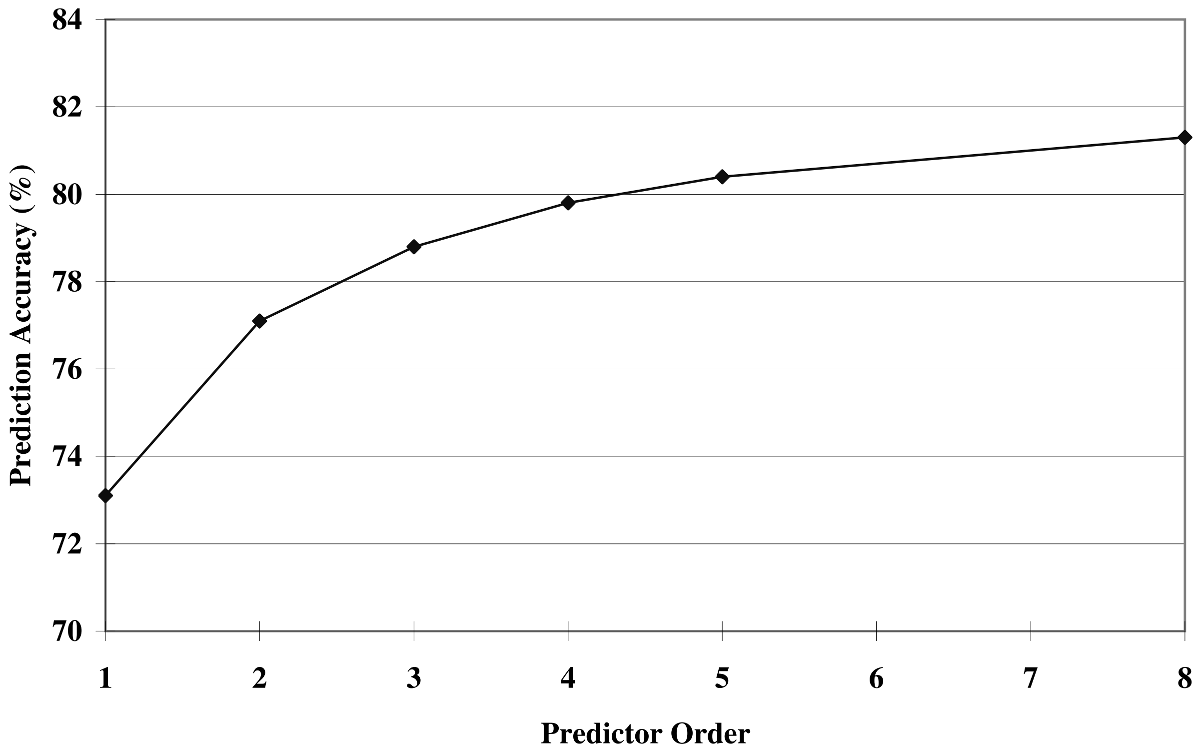
- Small variation across the different input files - unbounded tables not affected by different data set

- Input Flags

Flags	Predictions (mil)	Correct (%)
none	31	78.6
-O1	76	75.3
-O2	121	76.9
ref flags	137	77.1

- Small variation across the different compilation flags

Sensitivity on the Order



- Inreasing order translates to better accuracy – returns diminish with increasing order (large granularity of values)

Conclusions

- Data values are highly predictable
- Context based prediction outperforms previously proposed computational predictors
- Context based prediction needs to be used for high prediction accuracy - alone or in hybrid
- Few static instructions that generate relatively few values are responsible for the majority of improvement of Fcm over Stride prediction
- Instructions in general do not generate many unique values

Current and Future Work

- Fundamental questions
  - How predictable are data values?
  - Why are instructions predictable?
  - What is the behavior of predictability in programs?
  - How can predictability be exploited?
- Predictor Implementation Issues
  - Value predictor organizations
  - Choice of context
  - Efficient hash functions
  - Confidence mechanisms
  - Timing issues
  - Bandwidth considerations
- Software