Data dependent energy models: A worst case perspective

James Pallister, Steve Kerrison, **Jeremy Morse** and Kerstin Eder

University of Bristol

May 8, 2015

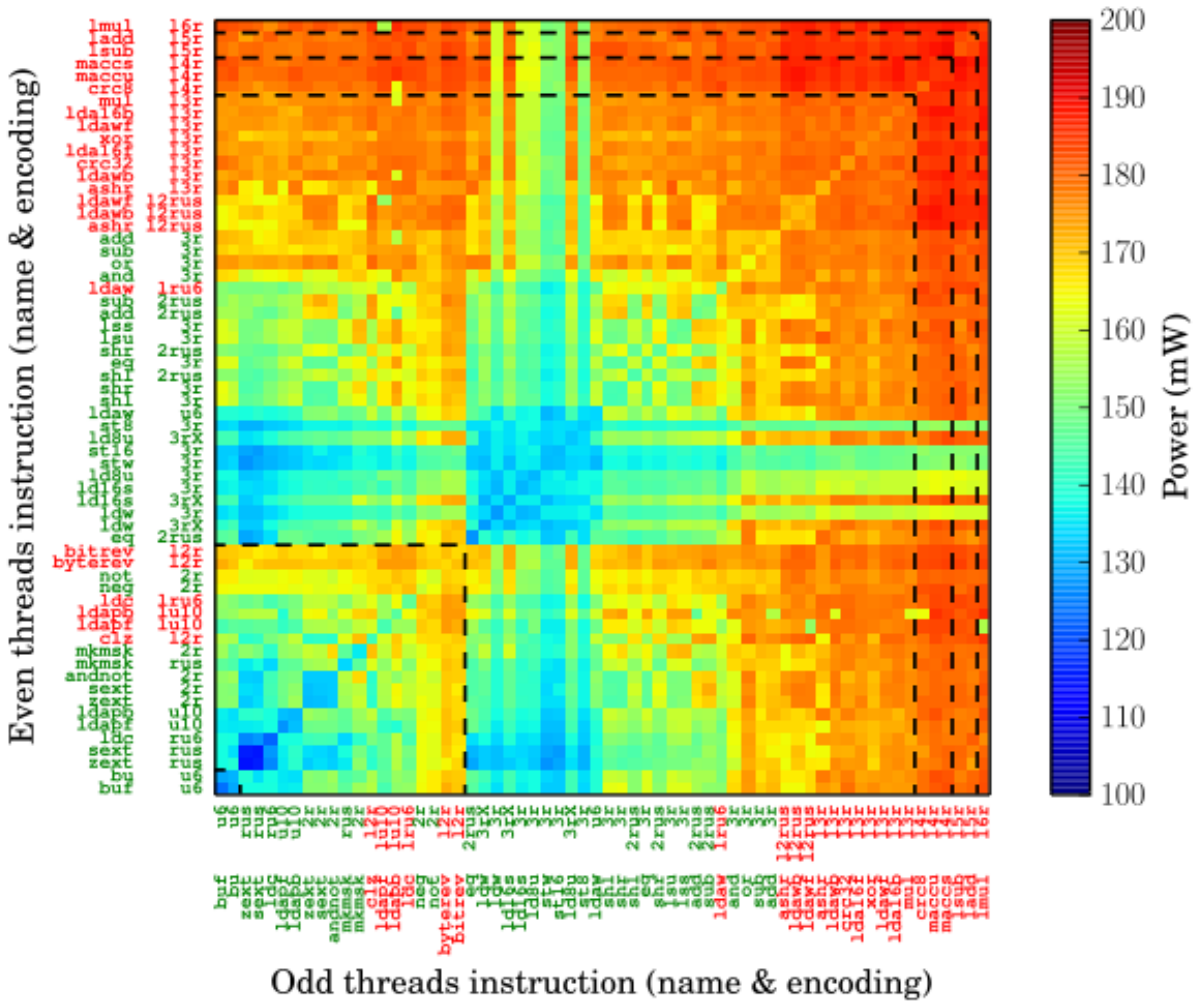
This talk

- Our existing energy model for the XCore
- How much energy is data dependent, and how do we model for it
- Examining energy distributions
- Instruction specific energy distributions
- Conclusions and future work

The Tiwari energy model

- Instruction level energy model
- Computes the base cost of executing any particular instruction
- Additionally, the cost of transitioning between two instructions
- "Extra" costs such as caches and branch prediction (not present on the XCore)

The existing Tiwari XS1 energy model



Currently implements an average-case energy model

- Instruction costs are averaged over a large number of executions
- Only coarse-grained consideration is given to the amount of data path switching
- Difficult to say that the cost for a sequence of instructions is best or worst, with this data

What could a data-dependent energy model tell us?

Within the confines of dynamic power,

- The amount of energy caused by switching in an instruction, caused by data
- The most common (average) cost of an instruction
- The most costly and least costly inputs/output for an instruction
- Ideally, for a sequence of instructions, what the worst case consumption is, and what data pattern triggers such consumption

Challenges

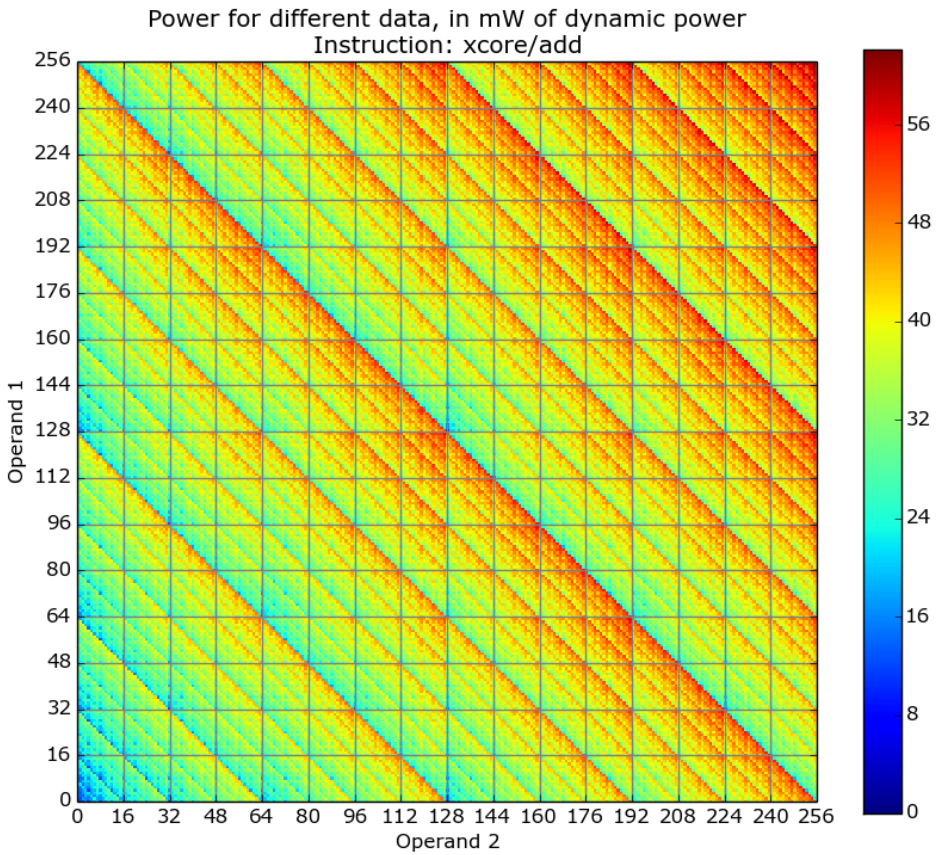
- How do we determine the best / worst / average cost of a particular instruction
- How do we compose such costs for a sequence of instructions?

Instruction specific dynamic power

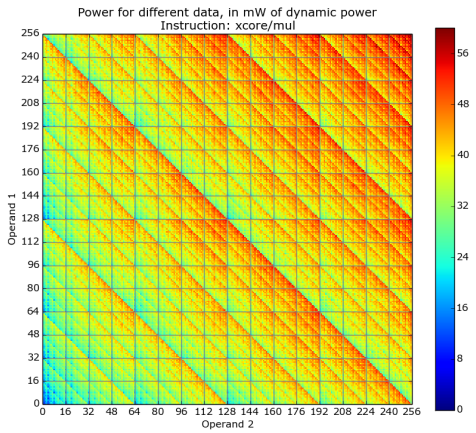
- Switching cost of charging/discharging circuits during instruction execution
- Instruction decode and functional unit activation covered by the inter-instruction effects measured in the Tiwari energy model
- Cost of data paths, register writeback, and other data dependent operations are not

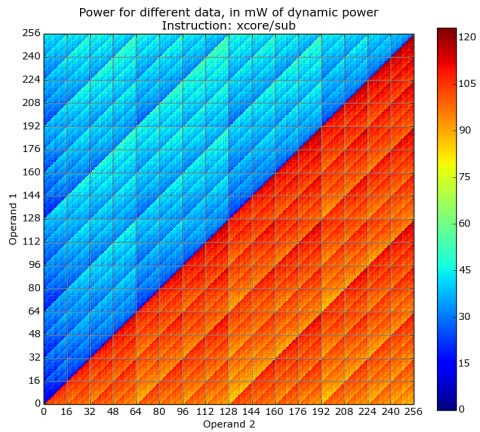
To address this we explored a small portion of the possible operand space on XCore

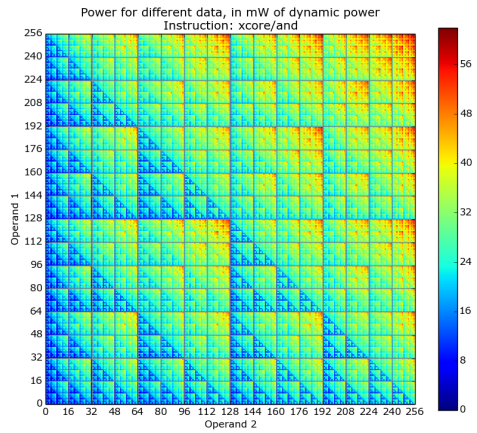
Dynamic power with a range of Add operands

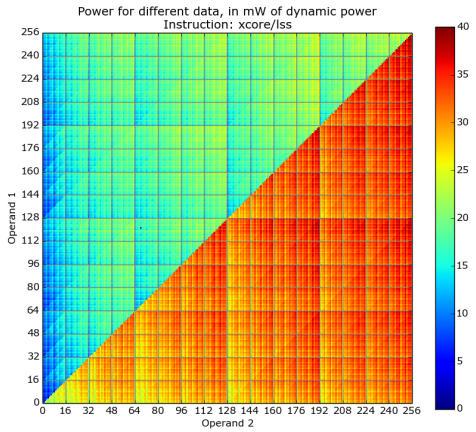


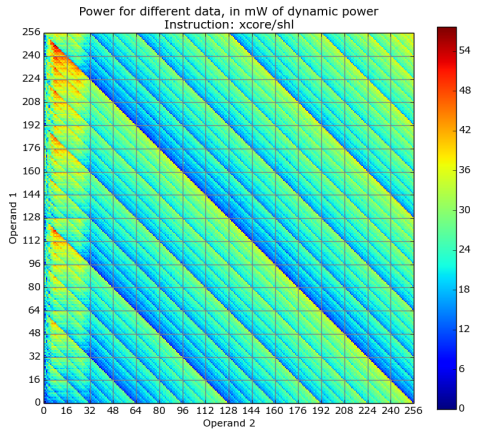
Additional power images

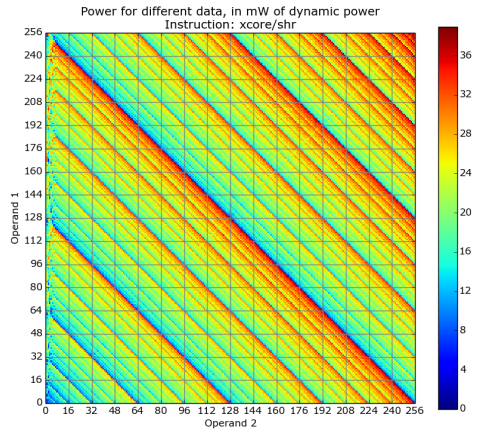












Limitations of this data

- It covers a tiny amount of the available state space (the "bottom left" of a larger square)
- We only have data for two input-operand instructions
- Models switching in the XCore *pipeline*, which is not necessarily the same as between instructions in the same thread

It does show, however, that a considerable portion of overall power (up to 25%) can be attributed to data switching

Problem 2: composing instruction costs

- Assuming we knew a perfect instruction cost for any input, how do we compose them?
- All instructions transform the distribution of inputs into a different output distribution, often non-linearly

What's the maximum switching for this sequence:

```
add     r0, r1, r2
mul     r0, r0, r3
shl     r0, r0, r4
sub     r0, r5, r0
```

Which instruction do you maximise switching for, and does that minimise others?

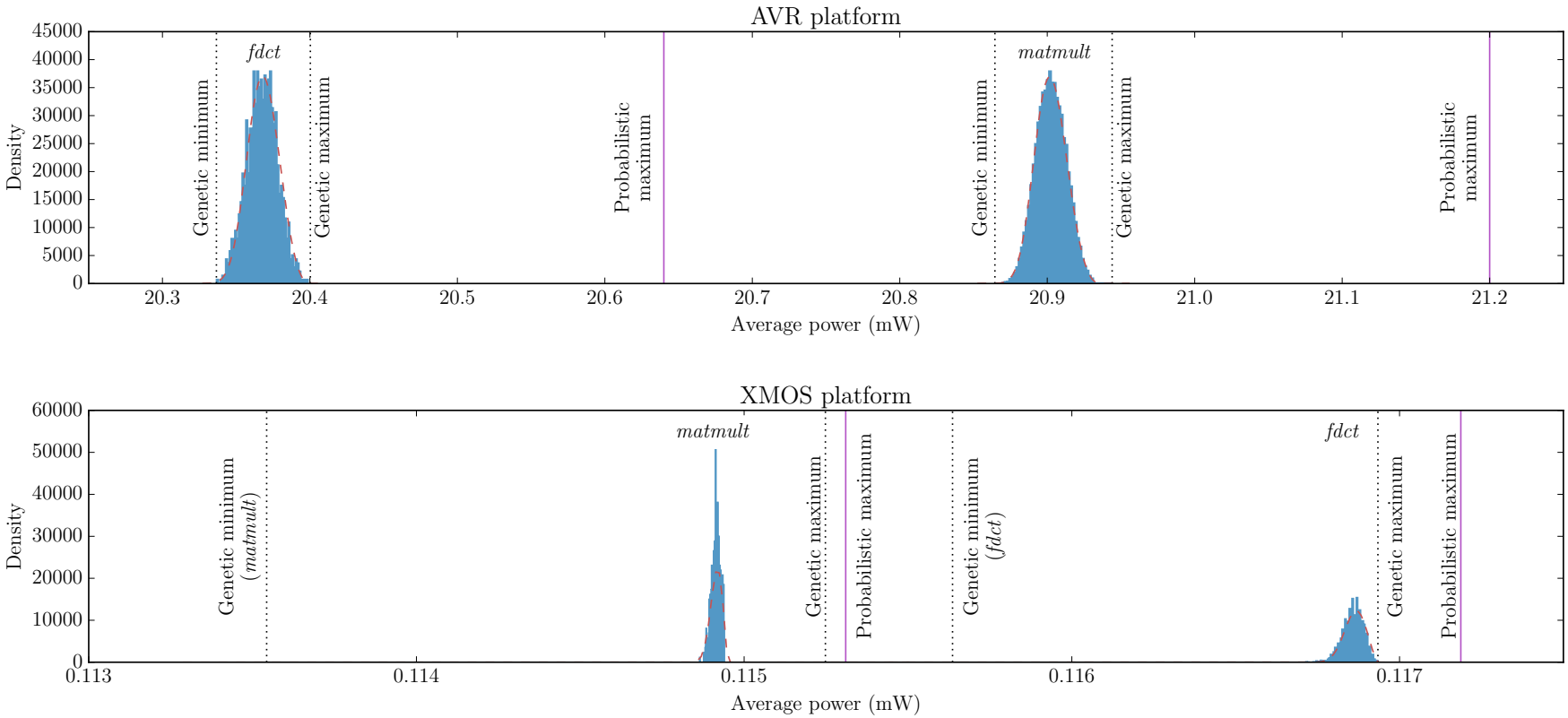
Is this feasible?

- One might need to enumerate a large amount of state space to discover the maximum amount of switching between instructions:
- This is probably NP-complete
- Assuming the maximum amount of switching possible for each instruction every time will not lead to tight bounds

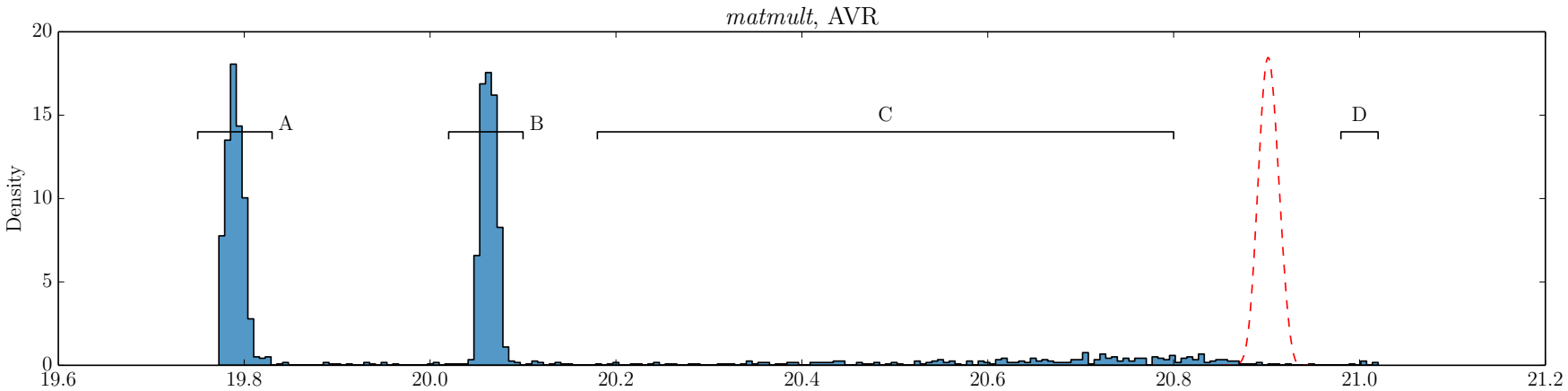
Instead: perform an analysis of energy distributions

- If instructions transform one distribution of inputs to another distribution, can we characterize energy consumption as such a distribution?
  - Do real world benchmarks yield a distribution of energy?
  - Can we estimate energy consumption of an arbitary instruction sequence in this way, by composition?
- We have:
  - Attempted this with two embedded benchmarks (FDCT and a matrix multiplication)
  - Characterized the results as a Weibull distribution
  - Used a (generic) genetic algorithm to search for high energy / low energy inputs

AVR and XMOS applied to FDCT (8x8) and MatMult (20x20) benchmarks



Contrived outliers



- Program energy appears to form a roughly normal distribution
- Certain patterns of data may cause extremely low or high power:
  - A: All zero (or very sparse) data
  - B: Most elements set to the value 1
  - C: Strided (repeating data)
  - D: Identical input data with high bits set
- Most of these biases reduce energy consumption

Can we build these distributions for arbitrary instruction sequences?

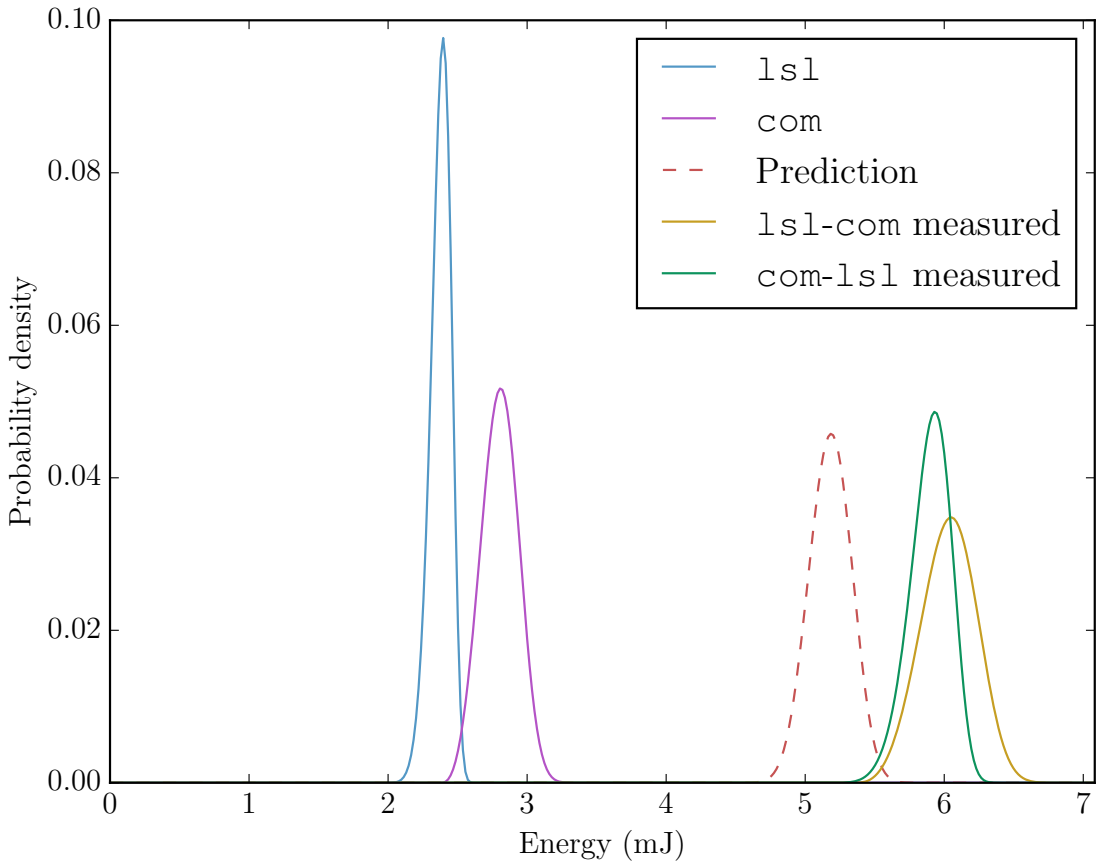
- The next goal is to try and build such a distribution from single instruction distributions
- Convolve instruction distributions together
- Resulting distribution should estimate the actual distribution of the sequence
- Such a distribution will also yield a probable upper bound

Test harness (AVR)

```
mov r1, X
loop_restart:
mov r0, Y
lsl r0, r1
mov r0, Z
...
```

Individual instructions harness: repeated operation on random values, then reset of destination operand. Similar approach for pairs of instructions. `mov` characterized and deconvolved separately.

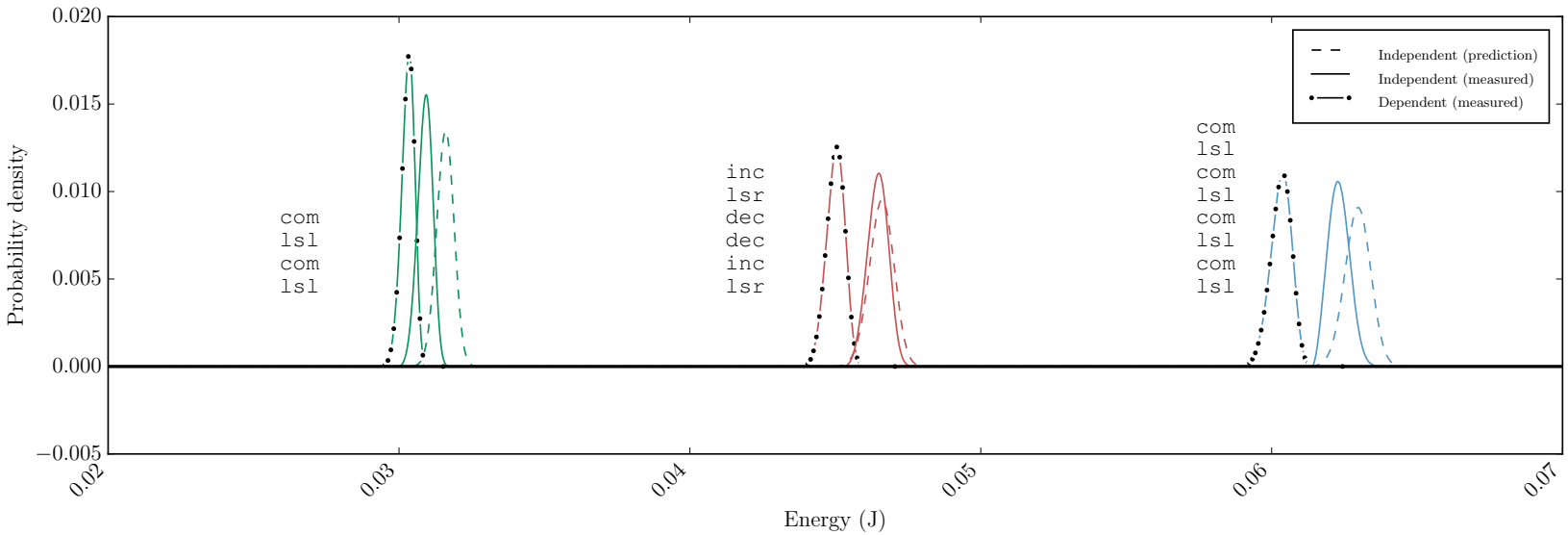
AVR distributions for LSL / COM



Difficulties in composition

- Convolution of both instruction underestimates actual energy consumption
- Additionally, cost of same instructions in different order not equal!
- Possibly due to differing inter-instruction costs between the two
- Produce instruction transition costs instead and try again

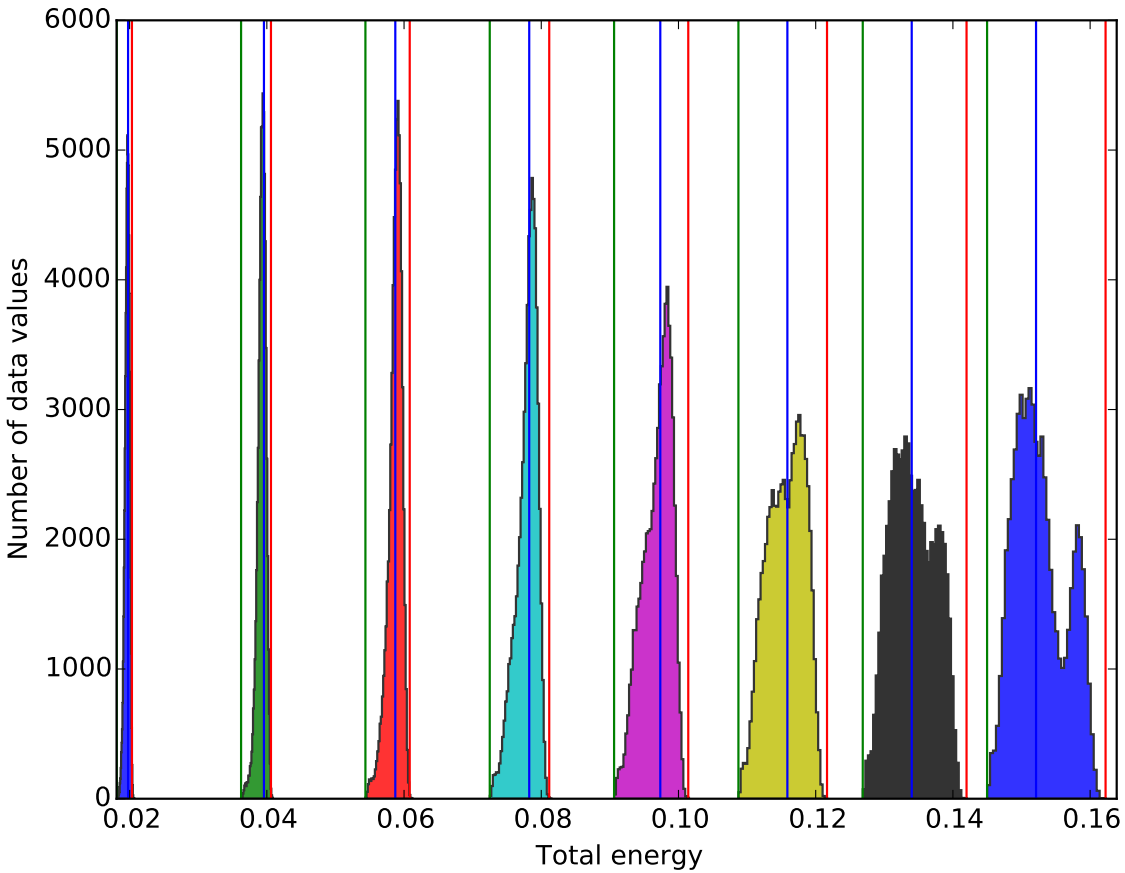
Energy distribution using instruction transitions



Once composed, some instructions do not make perfect distributions

- Repeatedly composing instructions does not always lead to characterizable distributions
- Large numbers of multiply operations on limited amounts of input tend to overflow then latch at zero
- The result is a bimodal distribution, which does not fit our previous analysis

Bimodal multiply power



Conclusions

- We may be able to calculate the worst case cost of individual instructions, however,
- Even with instruction worst case costs, we likely would not be able to compose them to find an effective and sound upper bound
- Programs can have their energy consumption characterized as probability distributions
- This leads to a probable upper bound
- Instruction transition distributions can predict the distribution of instruction sequences
- Not all distributions are perfect

Future work

- Can we use correlations / biases in the input data distributions to further refine the worst case estimate?
- Can "best" case consumption be estimated?
- Is there an effective way of modelling what the "general" or "average" case is?

Thank you for your time

- Questions?

Chaotic distribution