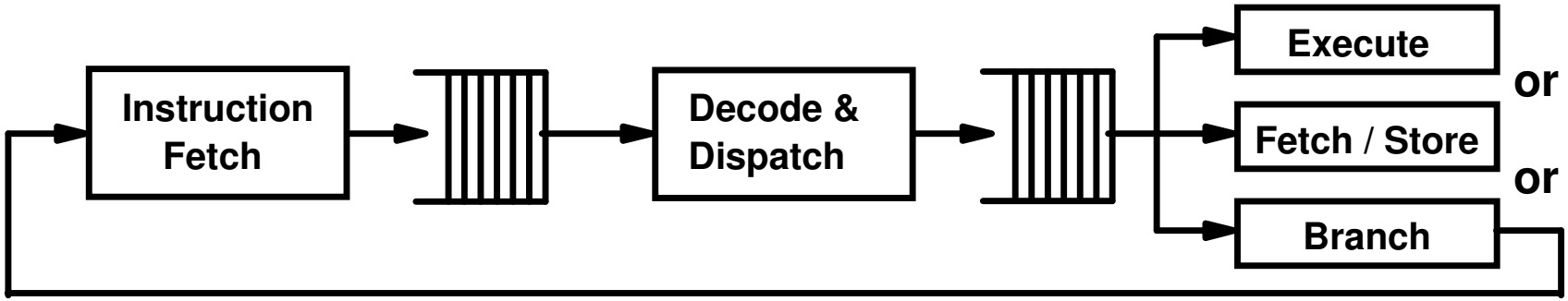
Branch Prediction:
Caveats and Second-Order Effects

Making a prediction is not an event...
... it's a process.

12/5/04

Phil Emma
pemma@us.ibm.com

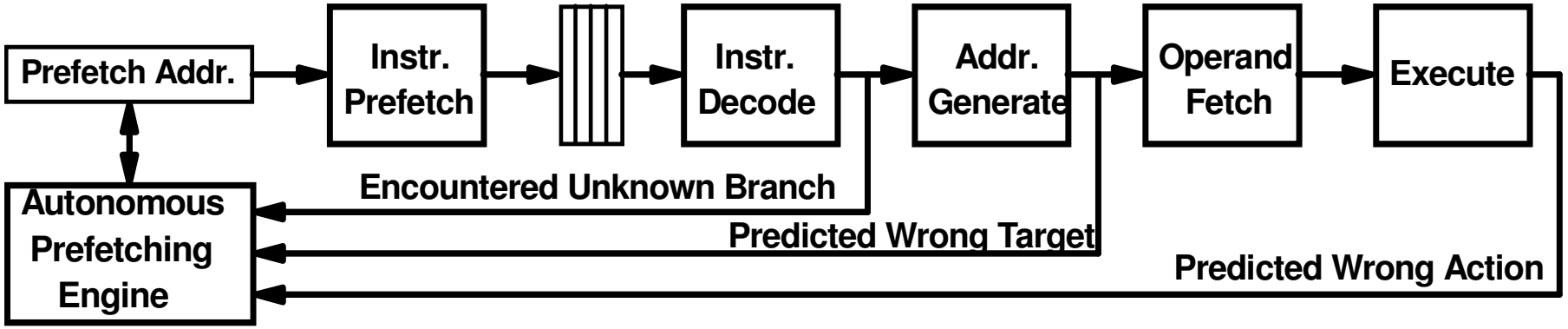
Q: WHY predict branches?



A1: To make ALL phases of processing run smoothly

A2: Because there are a lot of them

WHEN and WHAT to predict?



Prefetch Time: Is there a branch?

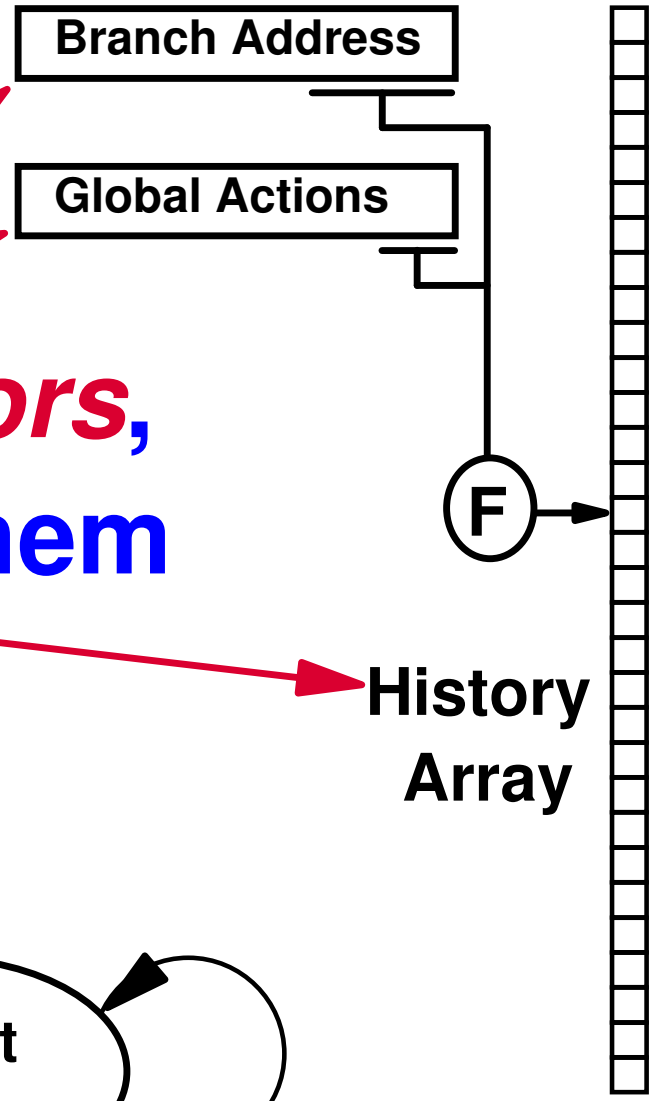
Is it taken?

To where?

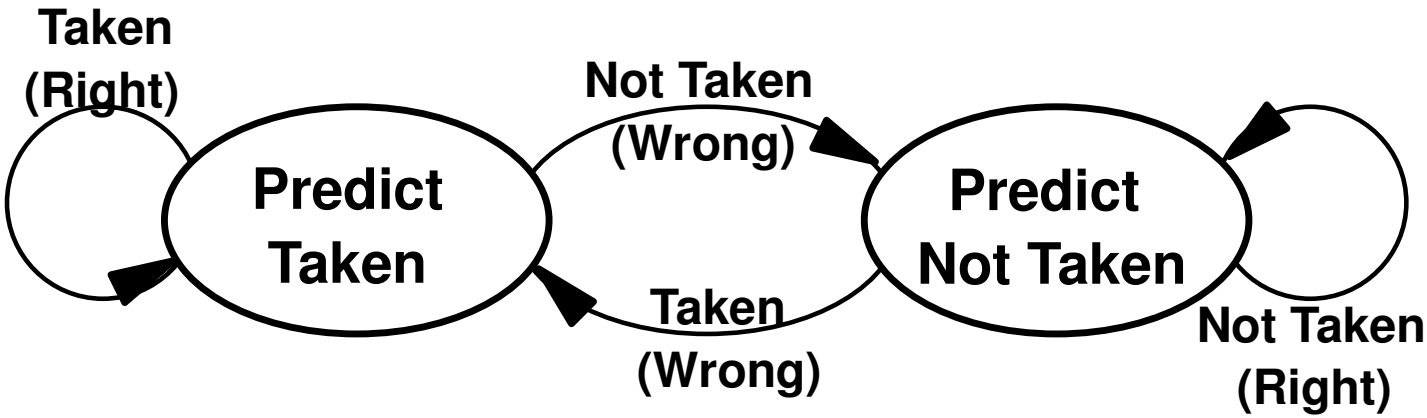
Decode Time: Is it taken?

Note: We can be wrong about THREE different things at THREE different times.

Decode-Time Prediction



What are *discriminators*, and *how* do we use them to predict branches?



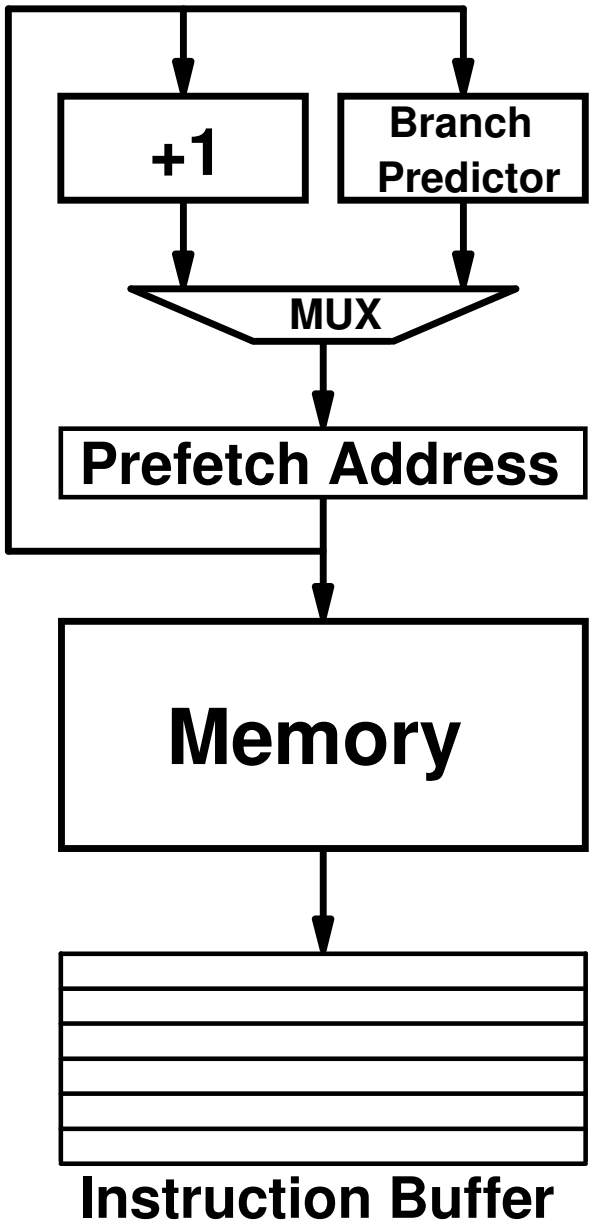
Why are discriminators fallible?

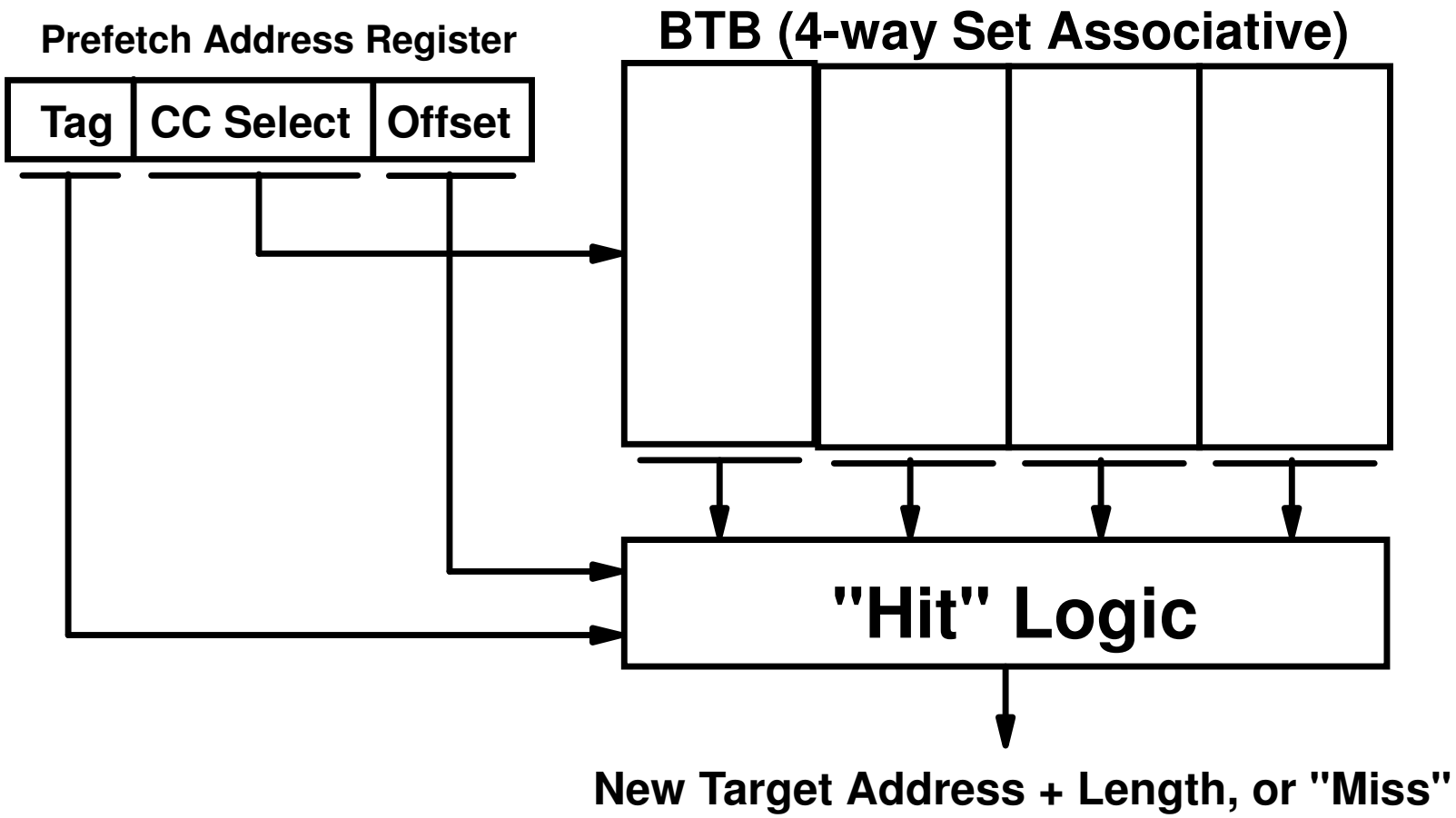
- Paths vary
- Individual branches vary
- We use VIRTUAL addresses, and are insensitive to page overlays
- Hashing functions cause collisions
- Collisions between threads

Prefetch-Time Prediction

Why is this harder?

- We do not get to see the instruction stream
- We have to predict the target address as well
- We are fetching blocks, not single instructions
- We must make our prediction in one cycle!





Q: How much context can the BTB hold?

(e.g.,relative to the I-cache?)

BTB Entry

Valid	Action	Tag	Offset	Target Address	Length	Special T	Checkbits

The Superscalar Conundrum

Q: At what issue rate do we require the prediction of multiple branches per cycle (with the corresponding multiplicity of I-fetches)?

Q: How large is the (compound) BTB entry needed to facillitate this, and ***now*** how much context can the BTB hold?

Q: Are Trace Caches more efficient in this context?

Implementation Concerns (1)

Predictors that need to know the outcome of the "last" n branches may not work in exactly the way that was anticipated.

Why?

In a deep pipeline, the "last" n branches have generally not executed by the time that the prediction is needed

Implementation Concerns (2)

Predictors that need to record lots of events (e.g., every branch) can run into problems because of the excessive bandwith required.

Why?

Searches and updates must be prioritized. Anomalous behavior will result no matter how this is done.

Implementation Concerns (3)

Predictions are usually urgent. We tend to need to know what we are going to do within a single cycle.

Why?

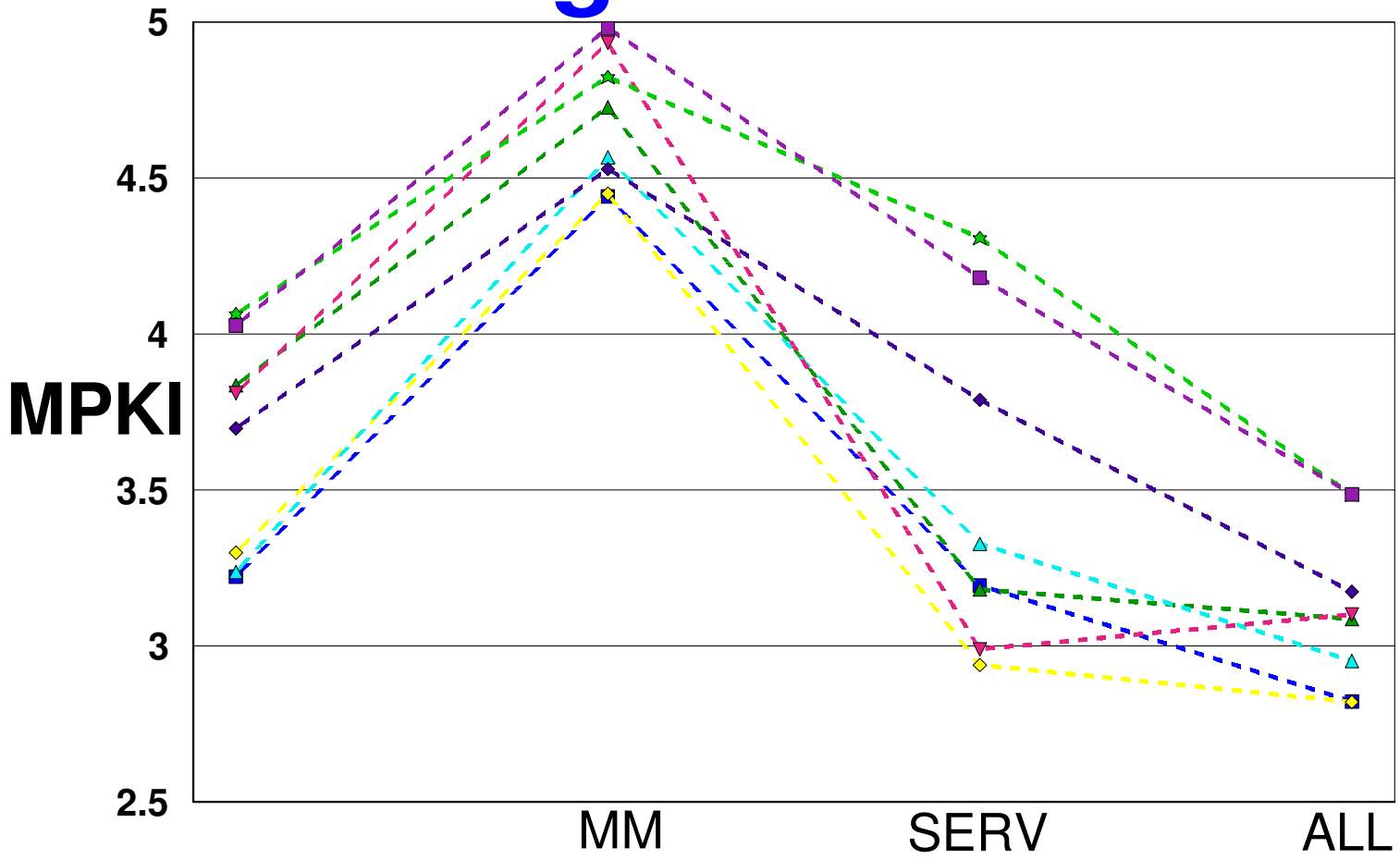
While complicated predictions can be pipelined and staged ahead, the crisis occurs following the first wrong guess.

Groundrules and Results

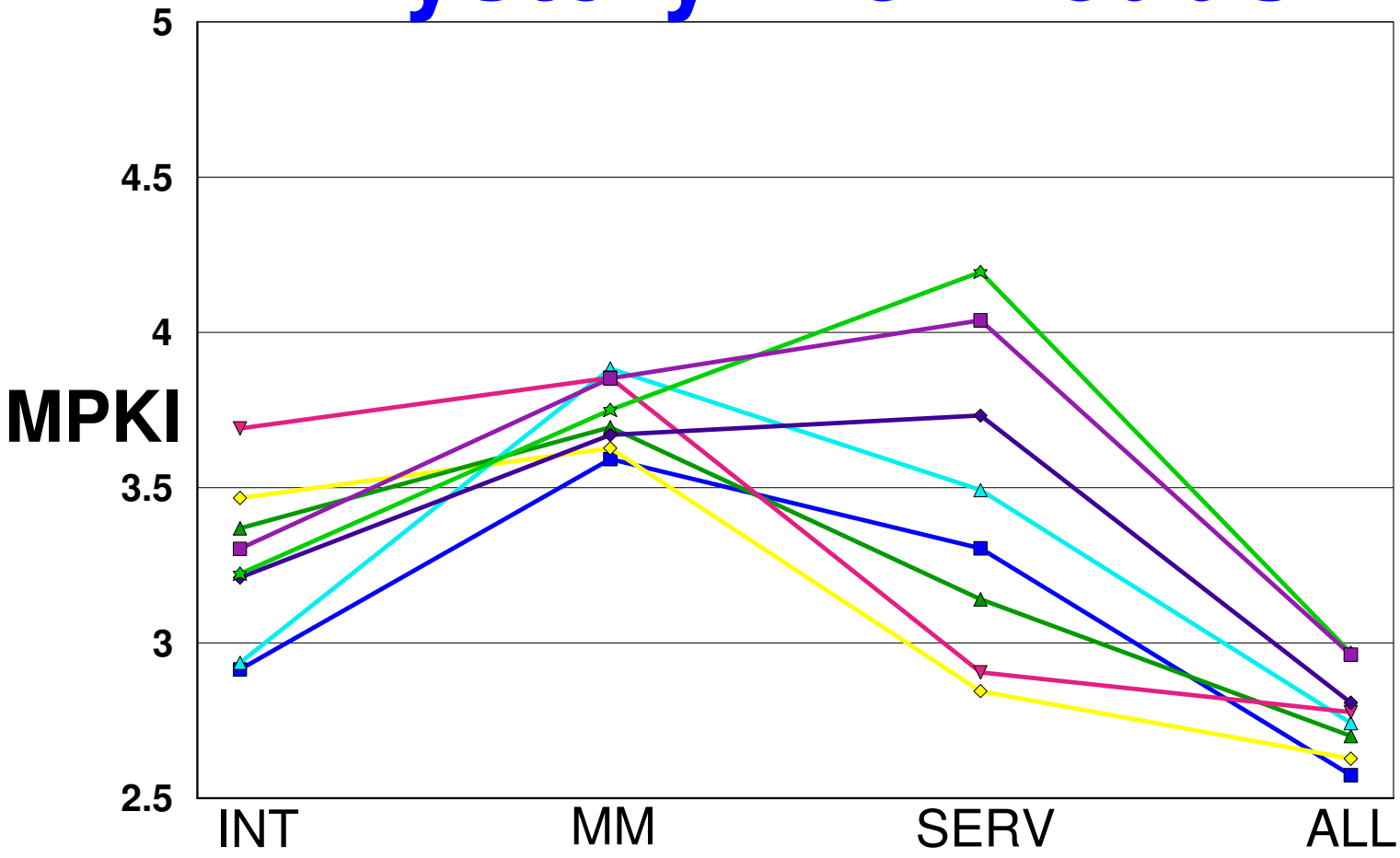
1. The only constraint was space.
2. The predictor can "see" the instructions.
3. Prediction of branch action at decode time.
4. No limit on complexity:
   - We ***DO*** know the last n branches.
   - We ***CAN*** record any and all information.
   - We ***DO NOT*** have to be timely.

The Panel Has Decided to Award a Special Prize.
But first ...

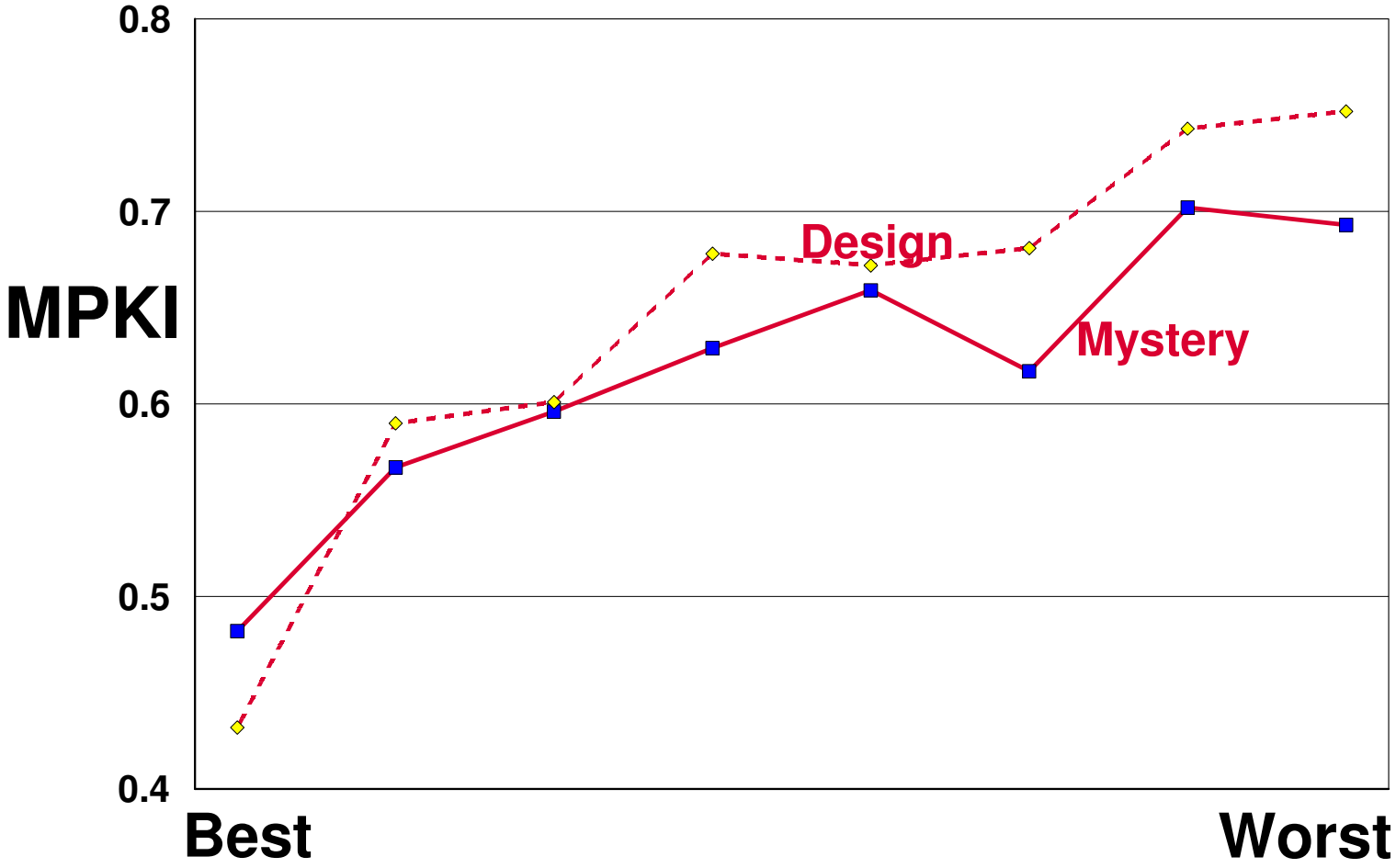
Design Workloads



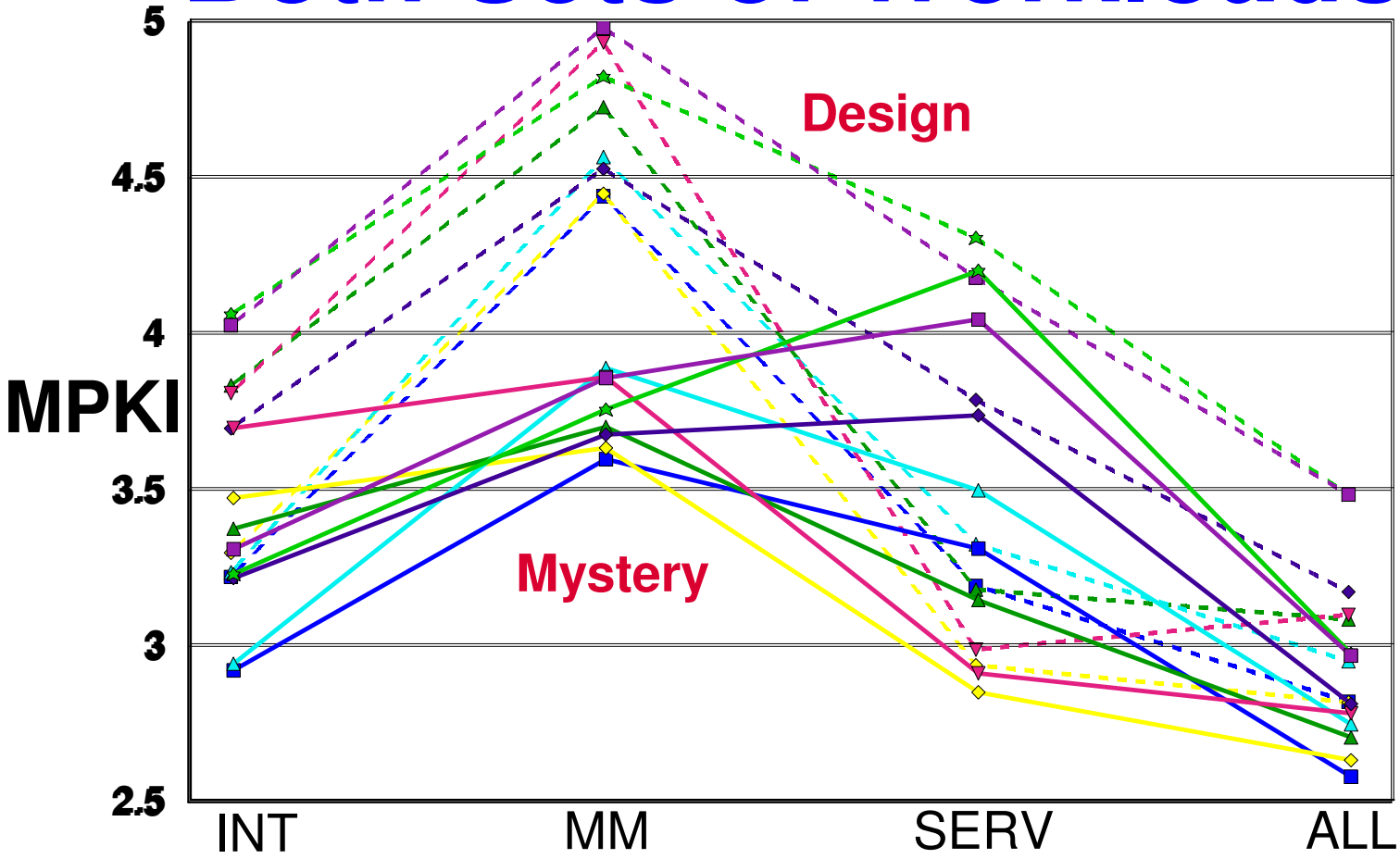
Mystery Workloads



Floating Point Workloads



Both Sets of Workloads



Special Award

for

"Best Practices"

Andre' Seznec

The O-GEHL Branch Predictor