

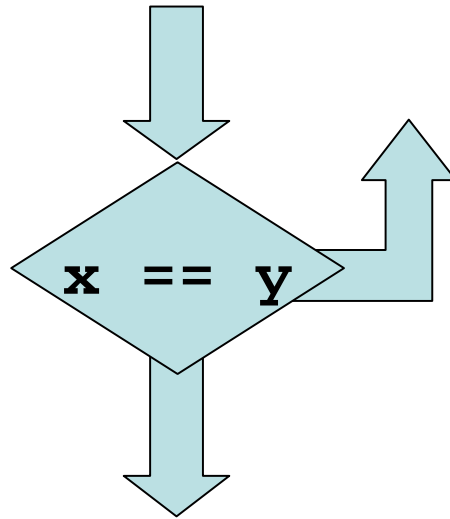
Branch Prediction with Bayesian Networks

Jeremy Singer, Gavin Brown and
Ian Watson

Prediction

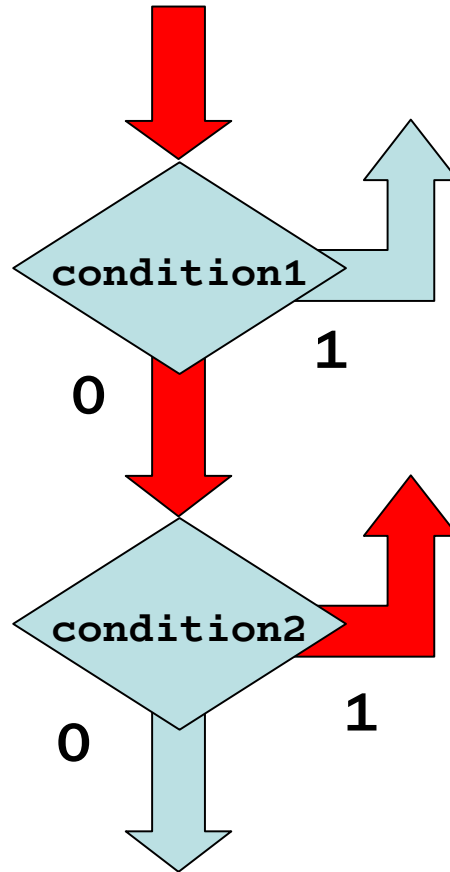
- Use the past to forecast the future
- Justification for History Departments
- Important in computer systems problems
 - *Branch prediction*

Branch Instructions



- Control flow goes one of two ways
- Boolean representation (1=taken, 0=not)
- Correct prediction => go faster

Global Branch History



Standard Predictors

- Table of all possible past outcomes
 - to some history depth
- Prediction of what should come next
 - what came next last time?

hist	next
00	0
01	1
10	1
11	1

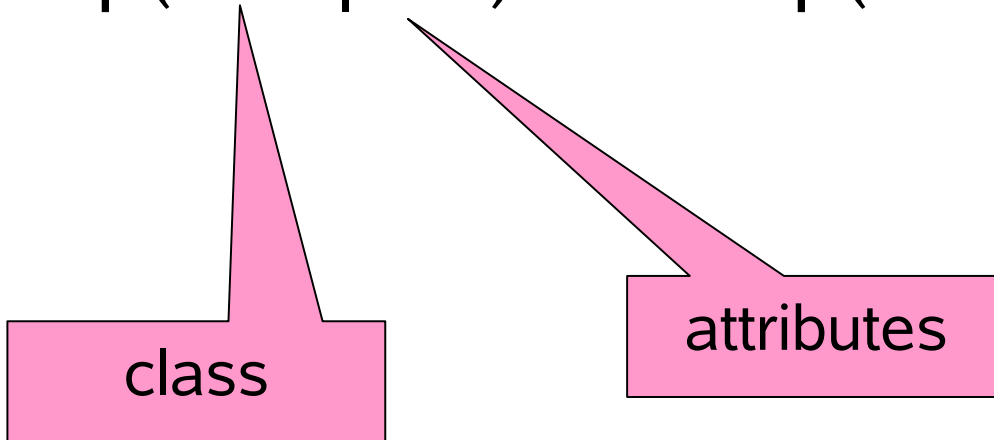
Increasing Accuracy

- Longer histories
 - Increases storage *exponentially!*
- Context of branch instruction (PC)
 - Hash with history (gshare)
 - Reduce aliasing in table

Bayesian Predictor

- Given probabilities of previous outcomes, compute probability of next outcome

- $p(\text{next}|\text{hist}) \propto p(\text{hist}|\text{next}) \cdot p(\text{next})$



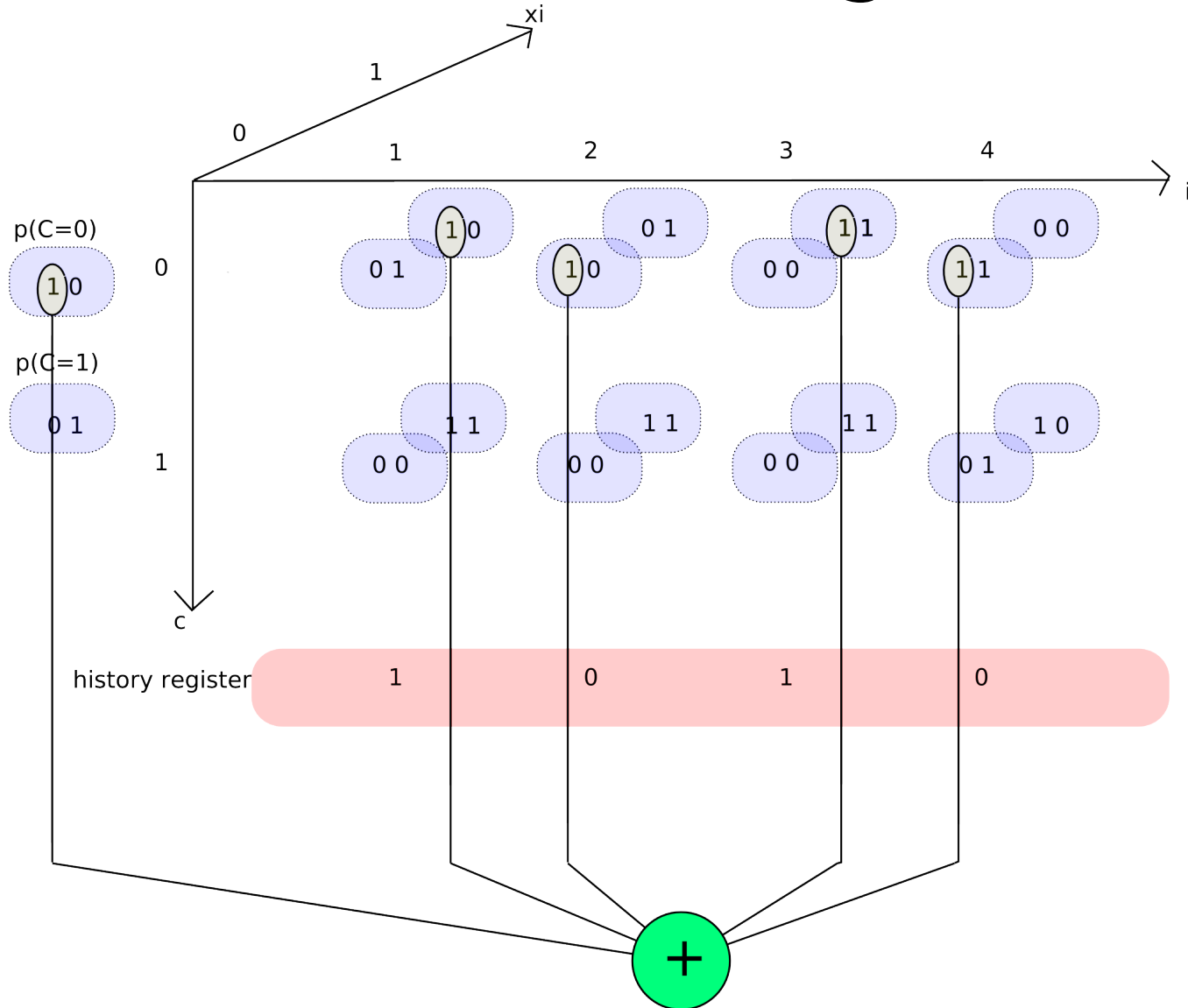
Independence Assumptions

- Split history attributes into individual bits (previous branch outcomes) $x_1 \dots x_N$
- Assume there is no correlation between these values (Naïve Bayes)
- $p(\text{hist}|\text{next}) = p(x_1|\text{next}) \cdot \dots \cdot p(x_N|\text{next})$
- Storage increases *linearly* with history

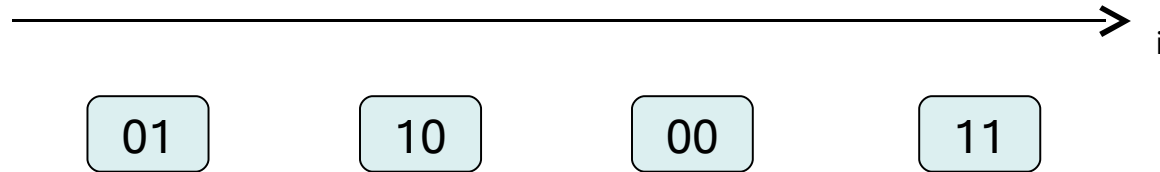
Simplifications

- Need to simplify Bayesian Predictor for hardware implementation
- Time constraints of branch prediction
- No floating point
 - Discretize probability ranges
- No multiplication
 - Count MSBs

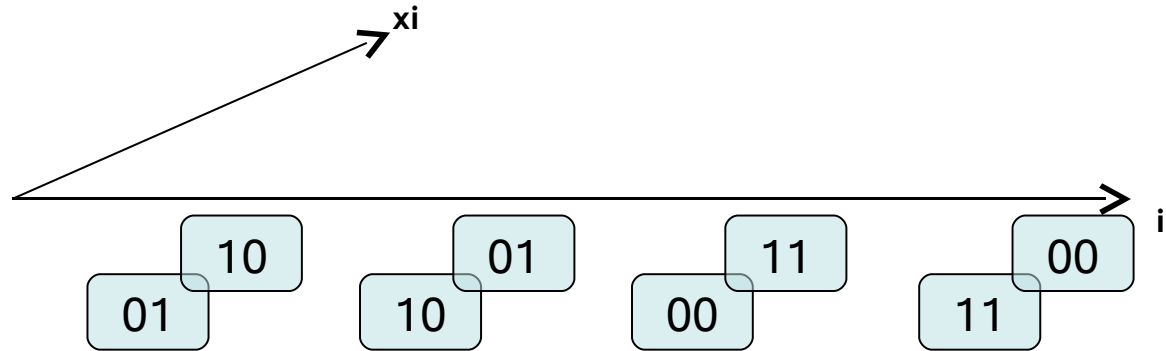
Schematic Diagram



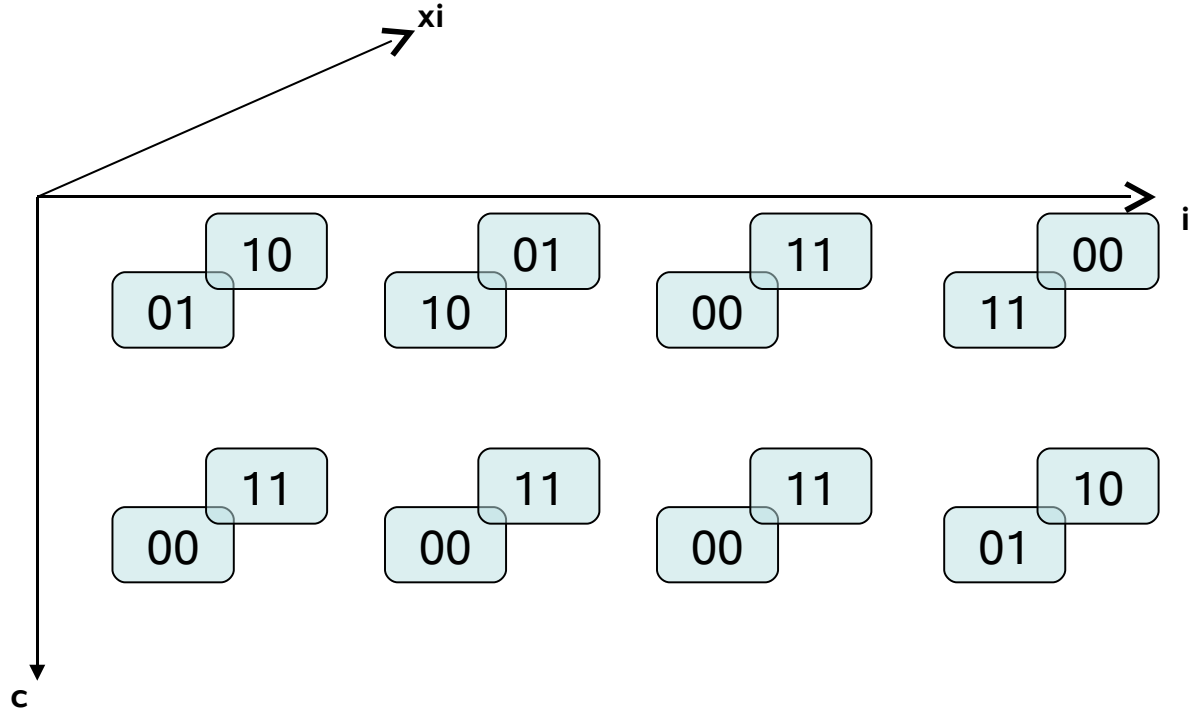
per-hist bit probability



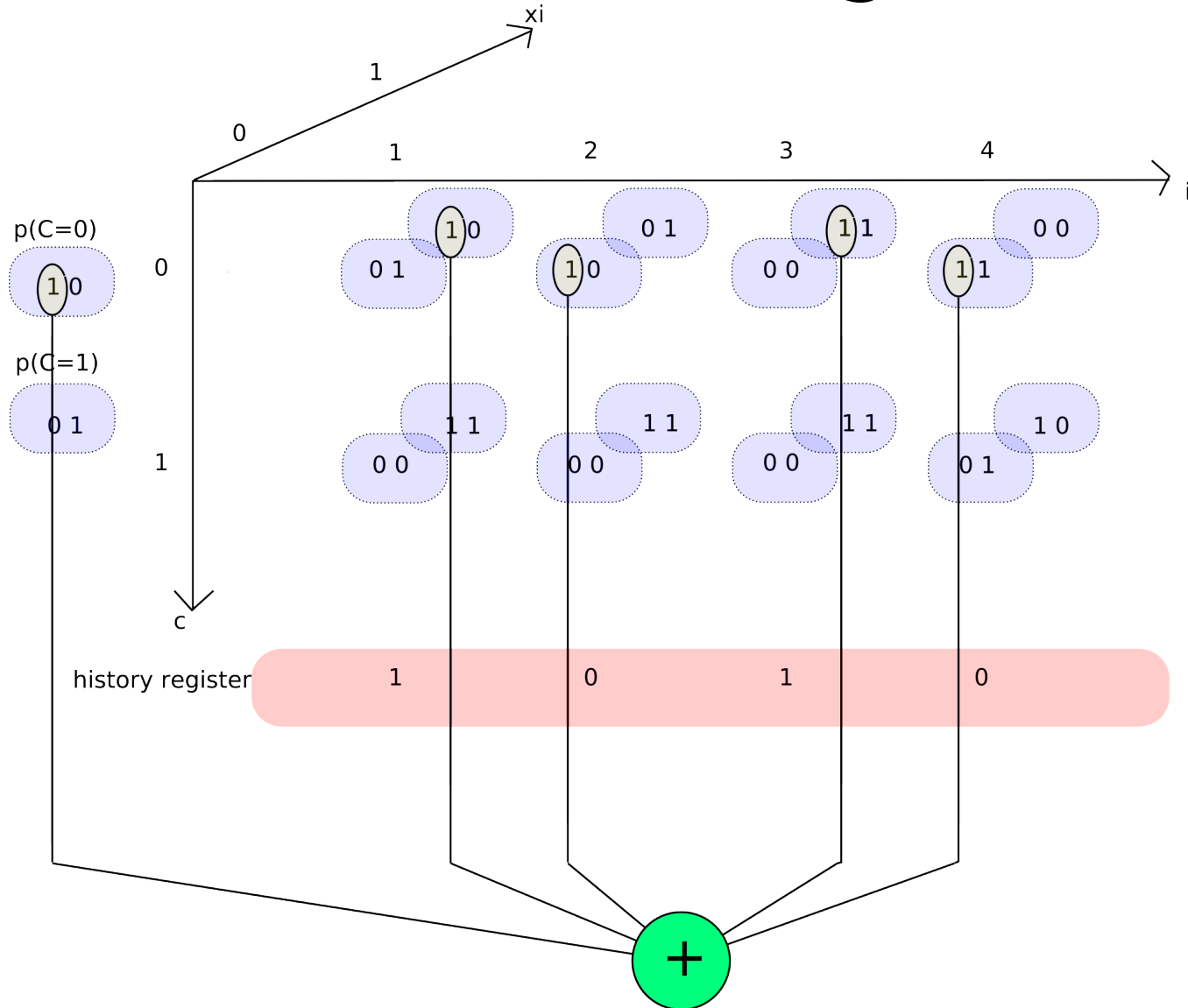
for both values of hist bit



for both possible outcomes



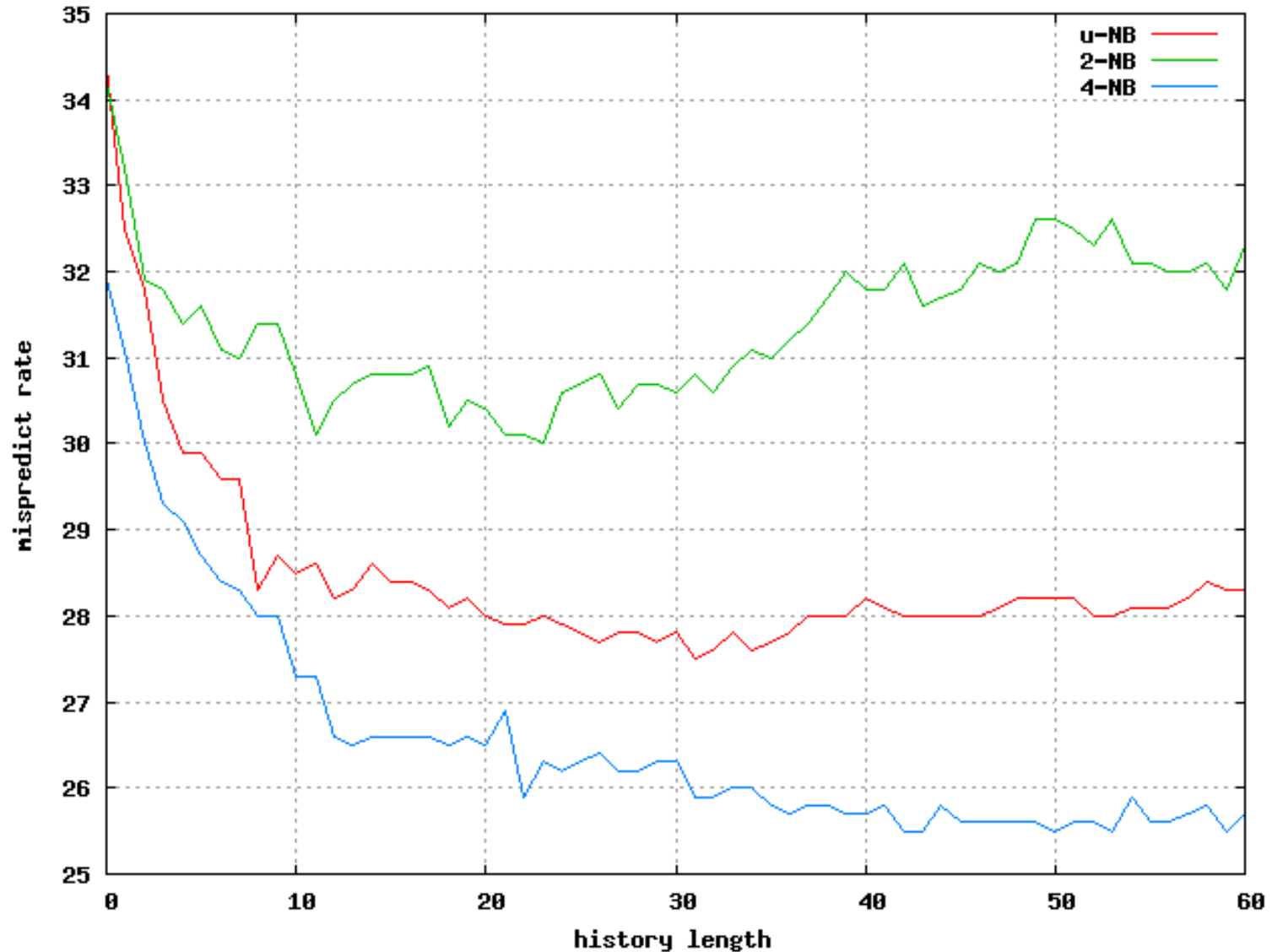
Schematic Diagram



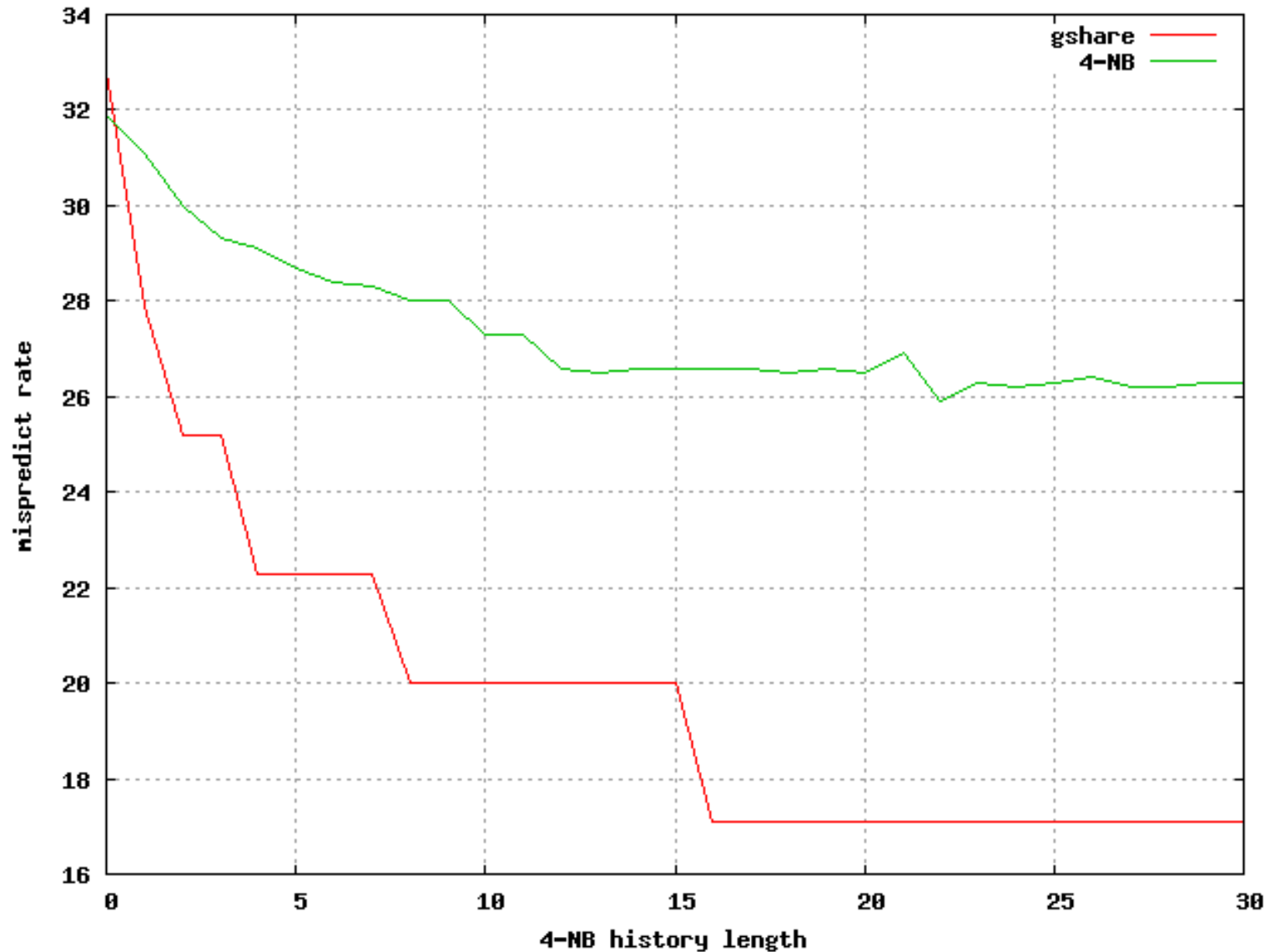
Evaluation Method

- CBP infrastructure
 - Traces from standard industrial benchmarks
 - Software predictor model
- Mispredict rate
 - % of incorrectly predicted branches
- Series of experiments
 - to explore performance of Bayesian model
 - Contrast with ‘equivalent’ gshare

Expt 1: diff size counters

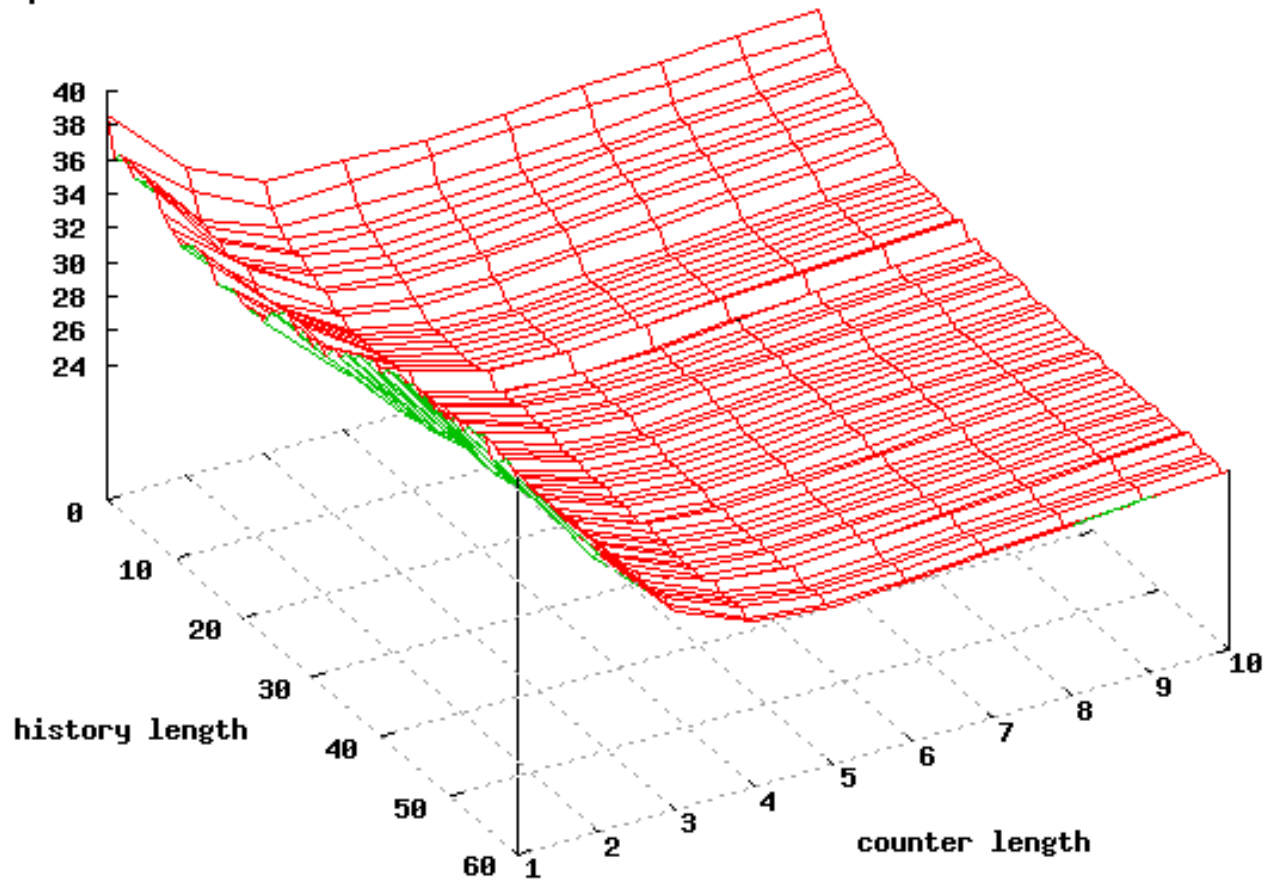


Expt 2: Bayes vs qshare

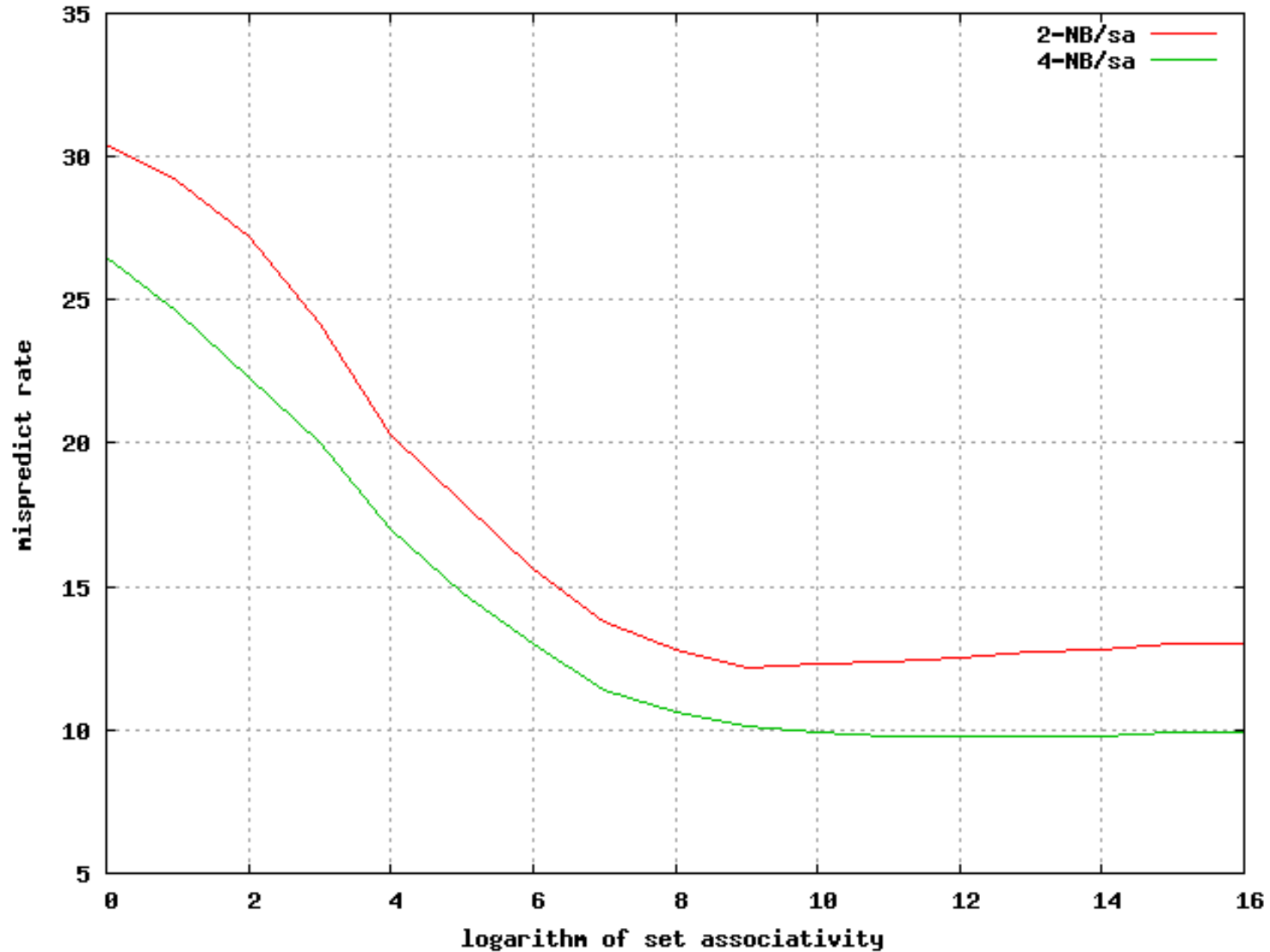


Expt 3: Vary hist and ctr size

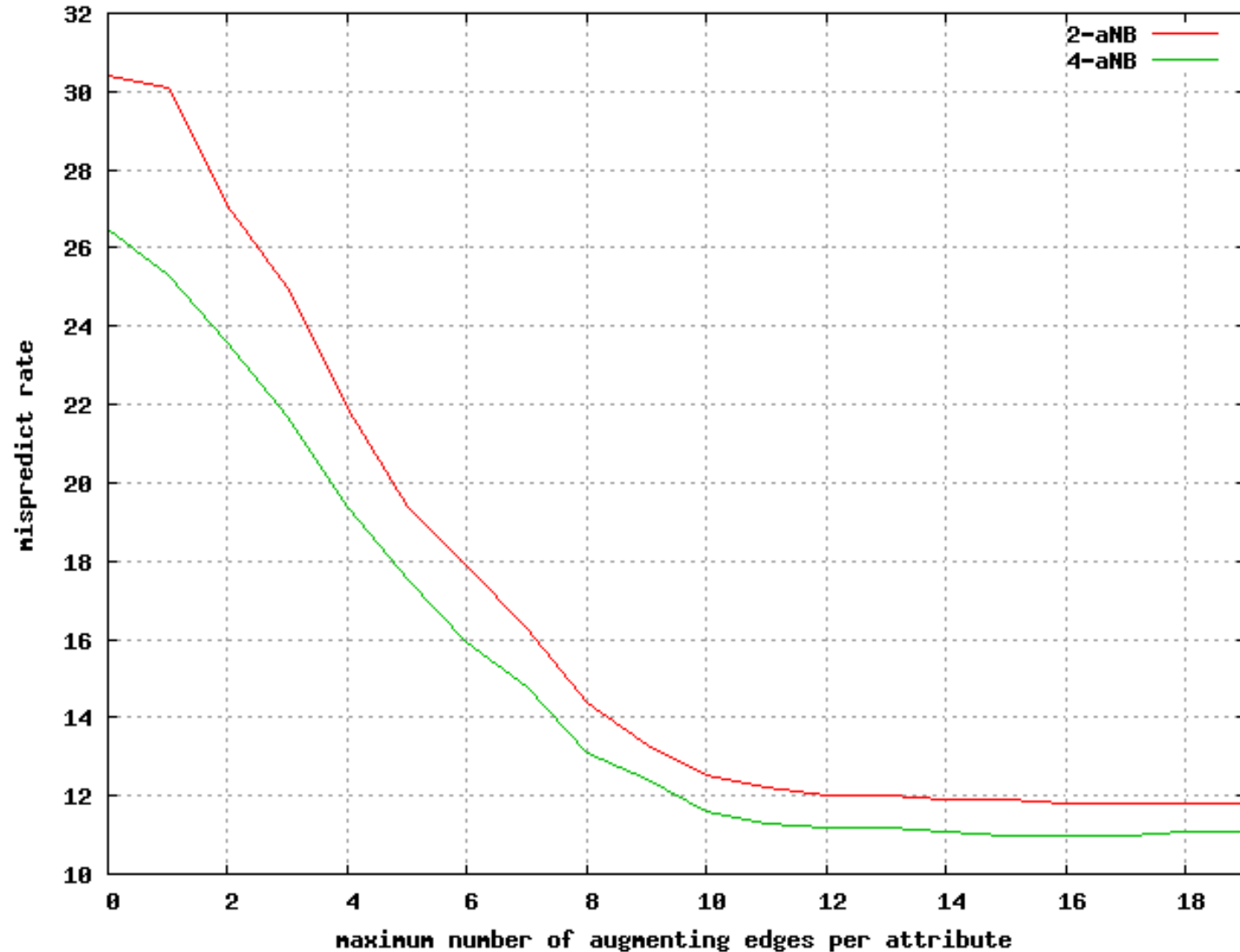
mispredict rate



Expt 4: Vary set-associativity



Expt 5: Vary dependences



Expt 6: Bayes++ vs gshare

- Combine all these independent ‘improvements’ to make a monster Bayes predictor
- 4-bit counters, 31-bit hist, 256-way set-associativity, 15 dependences.
- Equivalent to gshare with 19-bit hist
- Bayes: 4.7%, gshare: 4.0%

Conclusions

- Need to explore Bayes predictor model parameter space fully
- Implement machine learning algorithms in hardware
 - requires simplification
 - Other applications (GC, value prediction)
- Will this ever make it into real hardware?