Final Project

Advanced Computer Architecture

Álvaro Galisteo

Bachelor degree in computer engineering

TU Wien Erasmus 19/20

Implementation

Agree predictor

The Agree Predictor: A Mechanism for Reducing Negative Branch History Interference
Eric Sprangle, Robert S. Chappell, Mitch Alsup and Yale N. Pat

Perceptron predictor

Dynamic Branch Prediction with Perceptrons
Daniel A. Jiménez and Calvin L.

Agree predictor

AgreeBP class

Pattern History Table

Biasing Bit Storage

Branch history register

Agree predictor

AgreeBP class

Pattern History Table

Prediction based on agreement with bias
2-bit saturating counters
Variable number of entries
Variable counter size

Increase when bias was correct
Decrease when bias was incorrect

Biasing Bit Storage

Bias bit + tag
Indexed by branch address
Variable number of entries

Only updated the first time a branch appears

Branch history register

Outcome of k last branches
Variable length
PHT indexed by
BHR ⊕ branch address

Speculatively updated (need restore on squash)

Perceptron predictor

PerceptronBP class

Perceptron

Perceptron class

Perceptron table

Branch history register

Perceptron predictor

PerceptronBP class

Perceptron

Perceptron class

Compute y value as sum of product of bits and weights

Train weights based on y, outcomeand threshold value

Perceptron table

Indexed by branch address
Variable number of entries

Trains selected perceptron on update

Branch history register

Outcome of k last branches
Variable length
Variable initial n bit bias
Fed into perceptron for prediction

Speculatively updated (need restore on squash)

Analysis

PARSEC Benchmarks

blackscholes, bodytrack, canneal, dedup, ferret, fluidanimate, freqmine, x264

One for each category

Agree predictor

Table size and history length

Why?

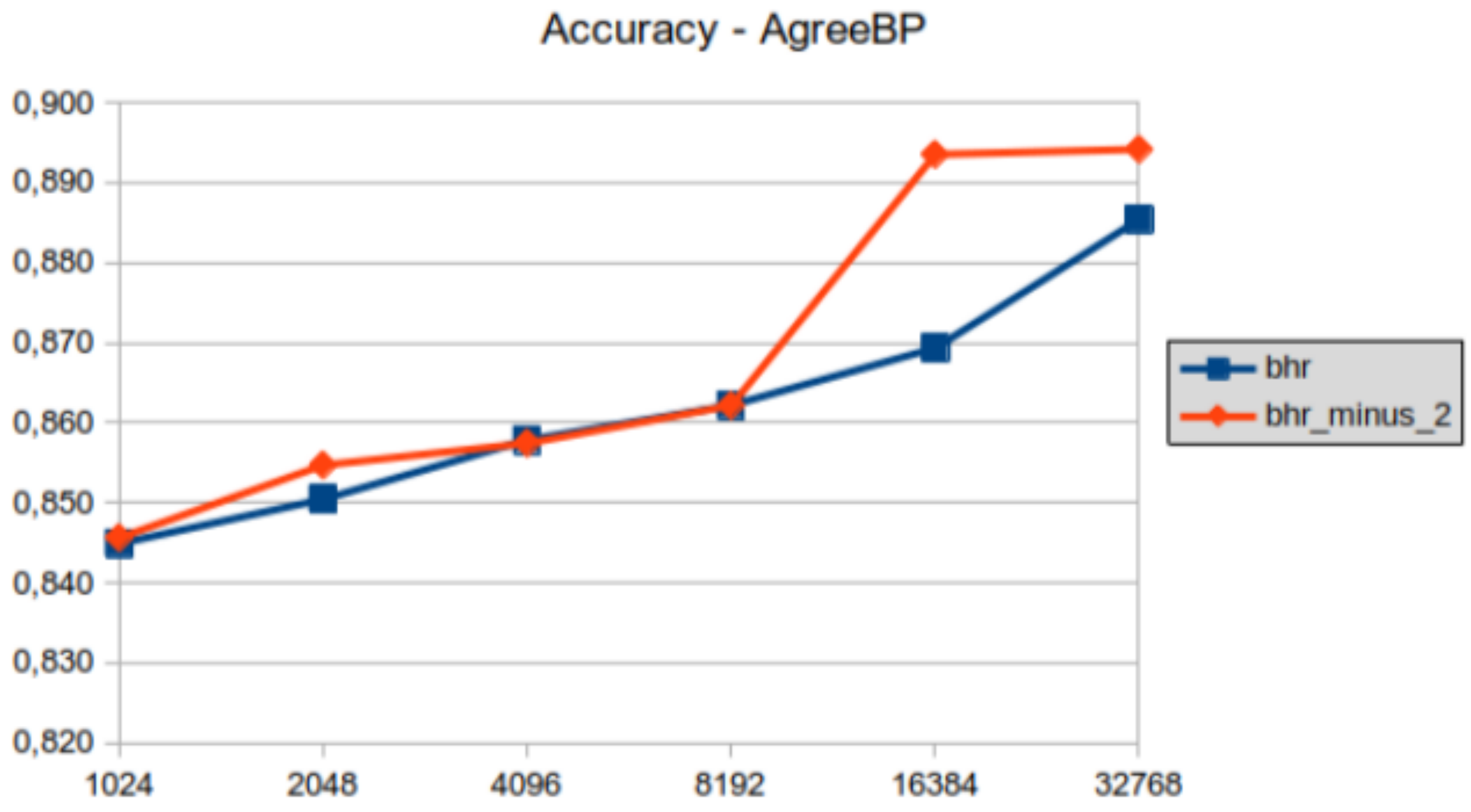
Which parameters perform the best based on hardware budget
The effect on the PHT indexing and branch address influence

Hypotheses:

1. The larger the tables, the greater the accuracy.
2. The branch history register has the same influence on selecting the entry in the pattern table as the branch direction.

Agree predictor

Table size and history length



Accuracy increases when size increases
Irregular accuracy trend line with different indexing

Agree predictor

Saturating counter size

Why?

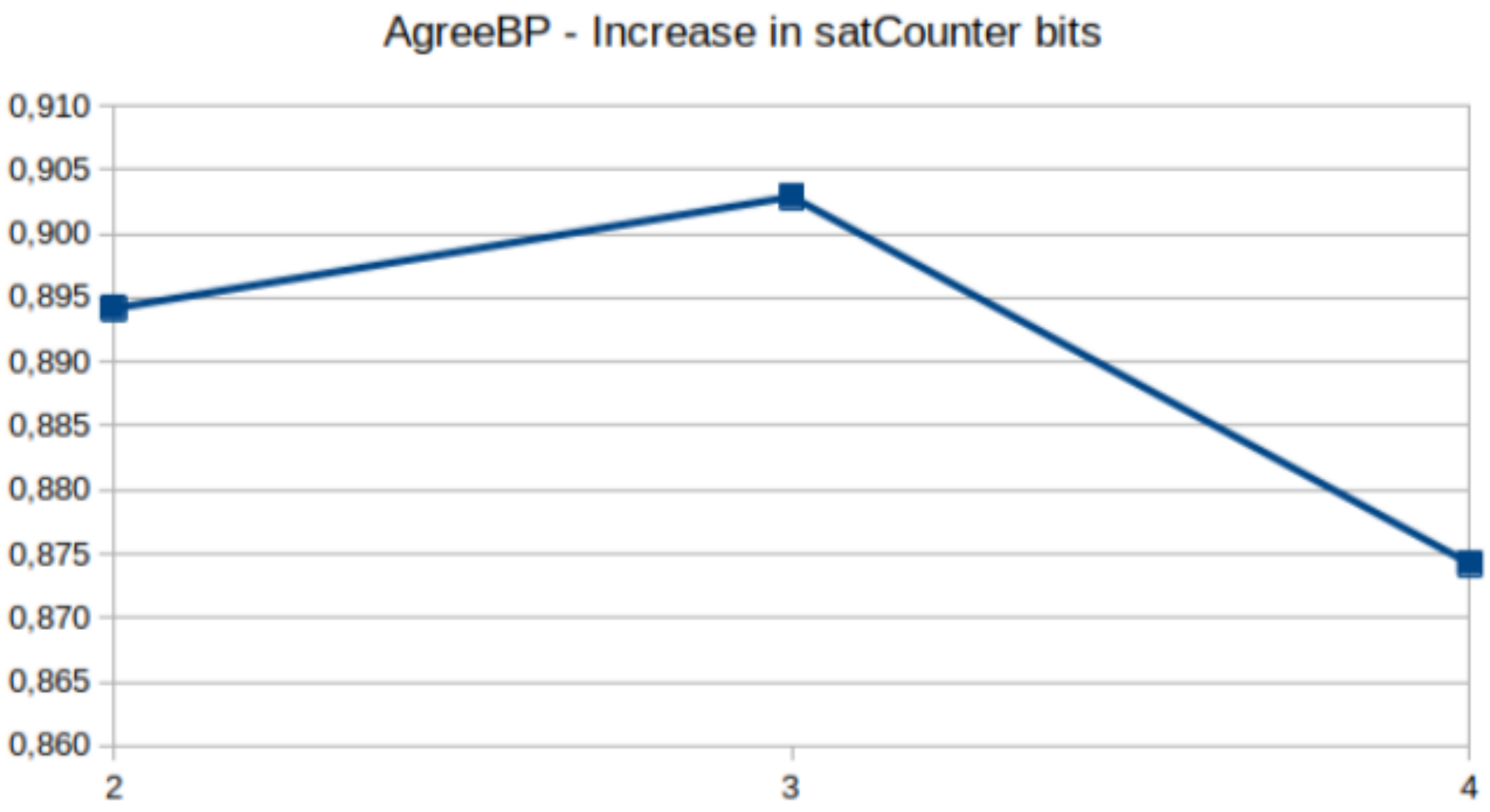
How a more "reluctant" agreement affects prediction

Hypothesis:

1. The higher the number of bits in the saturating counter, the higher the accuracy.

Agree predictor

Saturating counter size



Increased accuracy but up to an equilibrium point
More "reluctancy" negatively affects accuracy

Perceptron predictor

Table size and history length

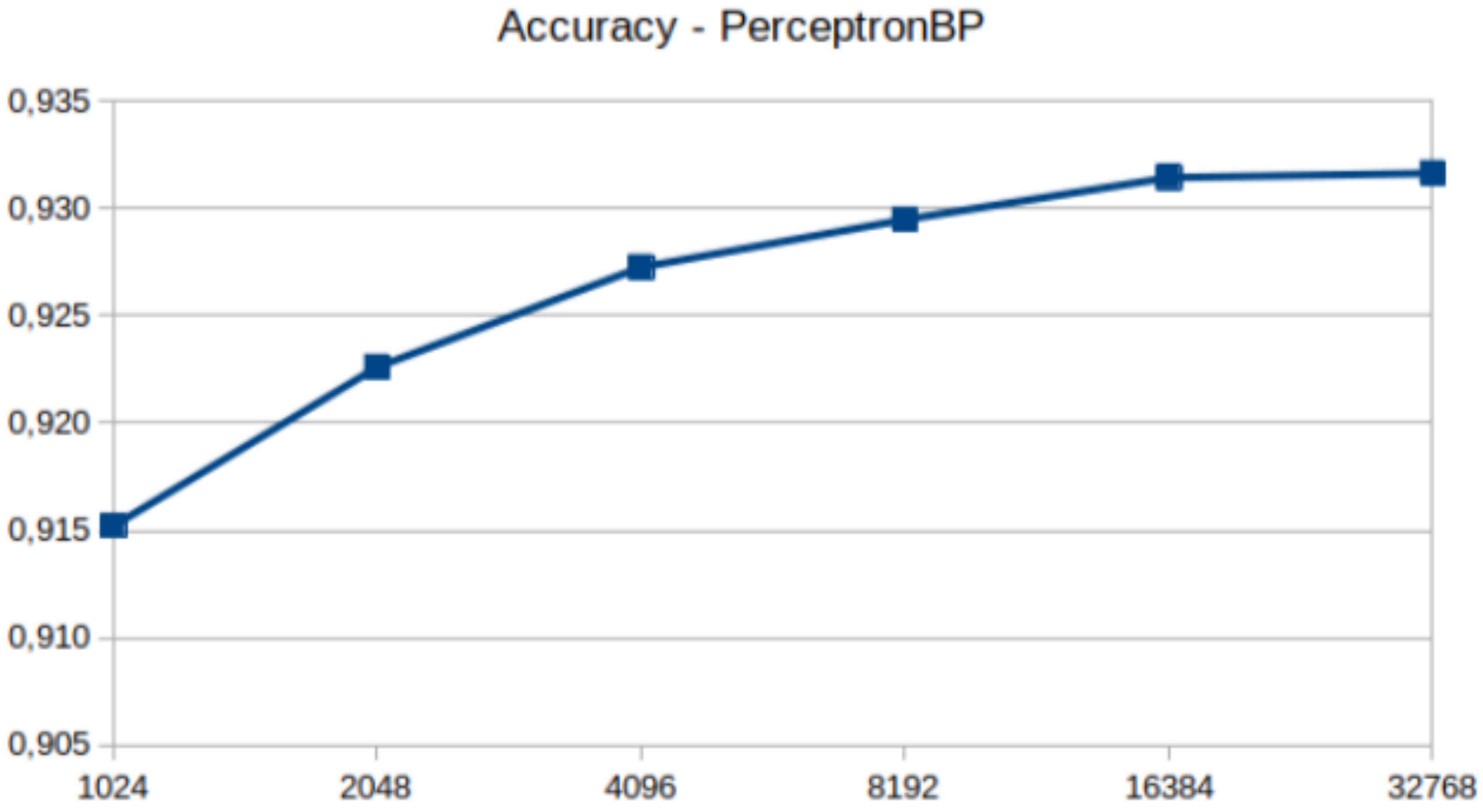
Why?

Which parameters perform the best based on hardware budget

Hypothesis: 1. The larger the tables, the greater the accuracy.

Perceptron predictor

Table size and history length



Increased accuracy

More entries and length increase training times, slowing down accuracy improvement

Perceptron predictor

Bias bits

Why?

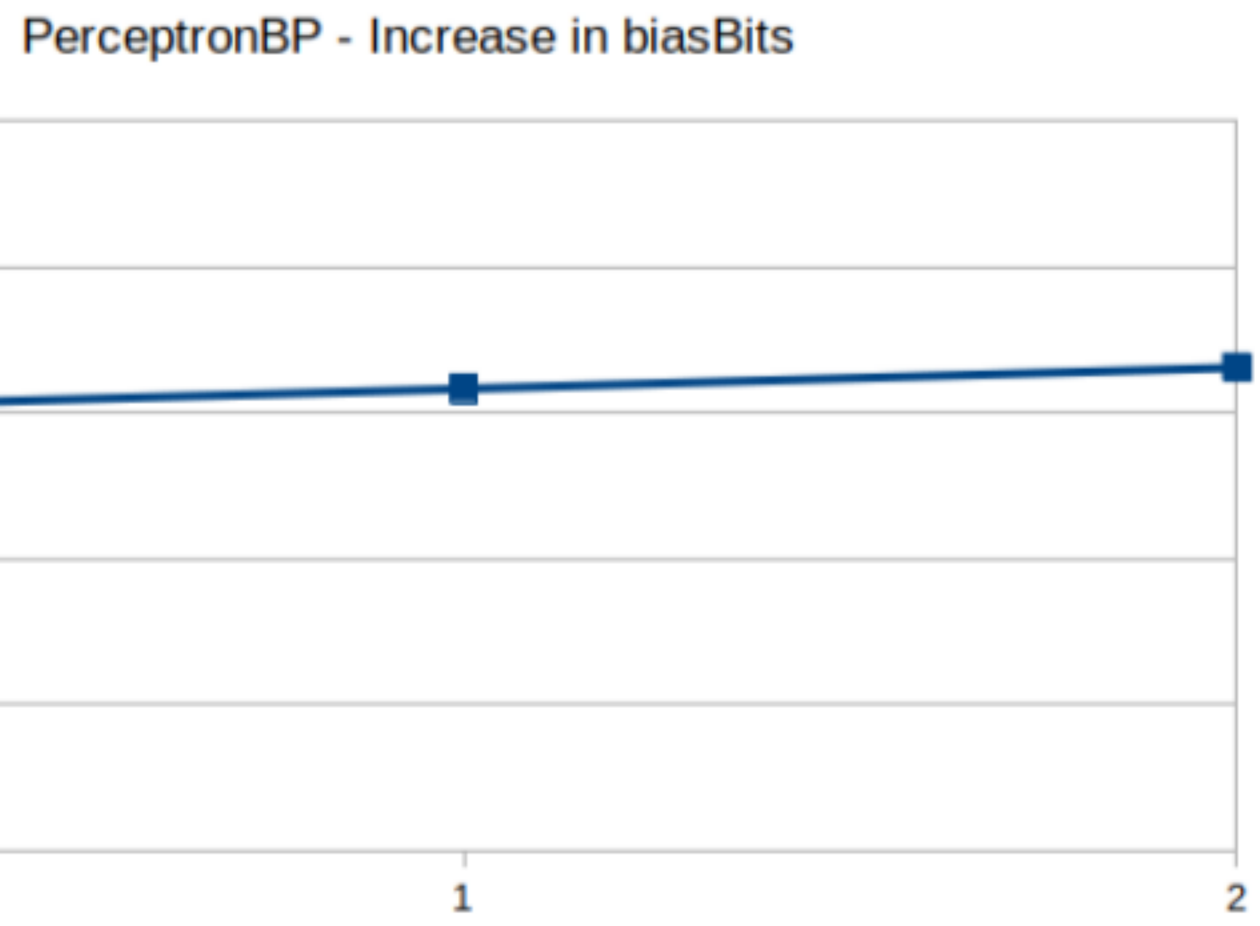
Is bias really needed? How much info is needed to learn correlations?

Hypothesis:

1. Increasing the number of biasing bits lead to better accuracy.

Perceptron predictor

Bias bits



Practically no influence

Equilibrium may appear where too much information is ommited and accuracy decreases

Comparison

Agree, Local, BiMode, Tournament, Perceptron and L-TAGE

Cronological order

Comparison

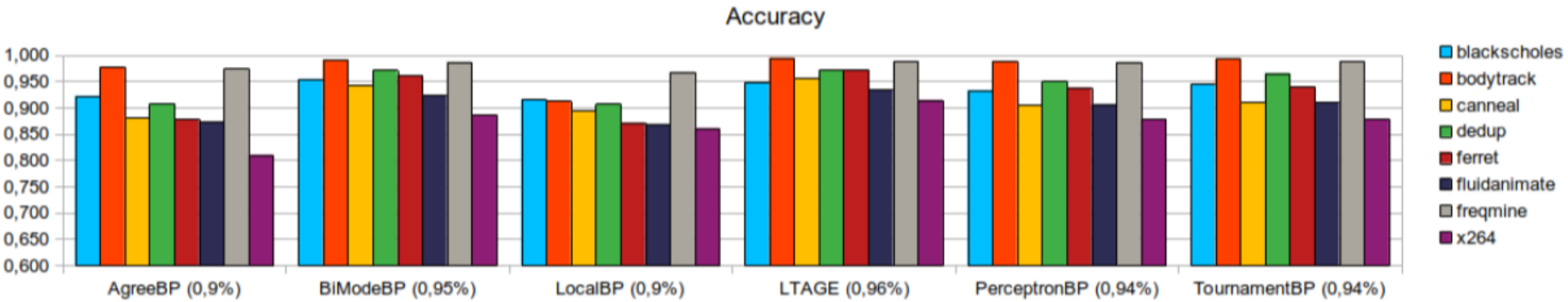
Agree, Local, BiMode, Tournament, Perceptron and L-TAGE

Cronological order

Main algorithms. Test performance/improvements over the years

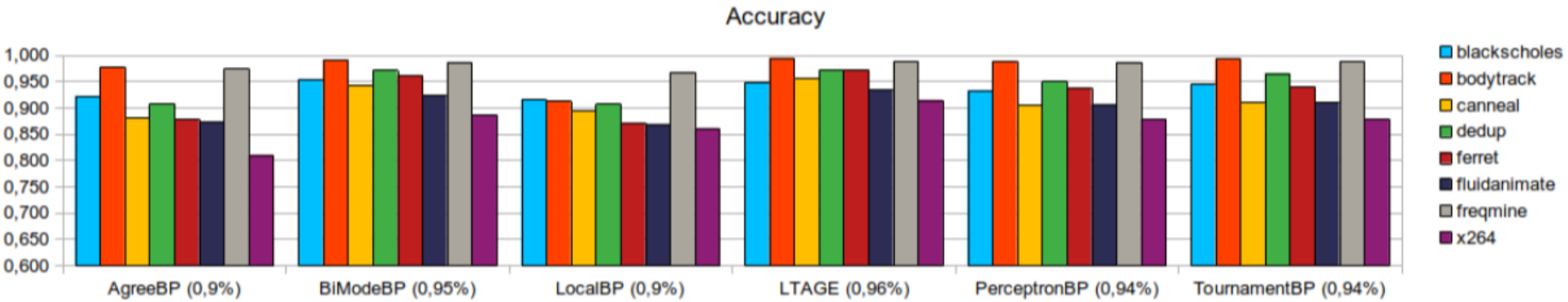
Comparison

Results



Comparison

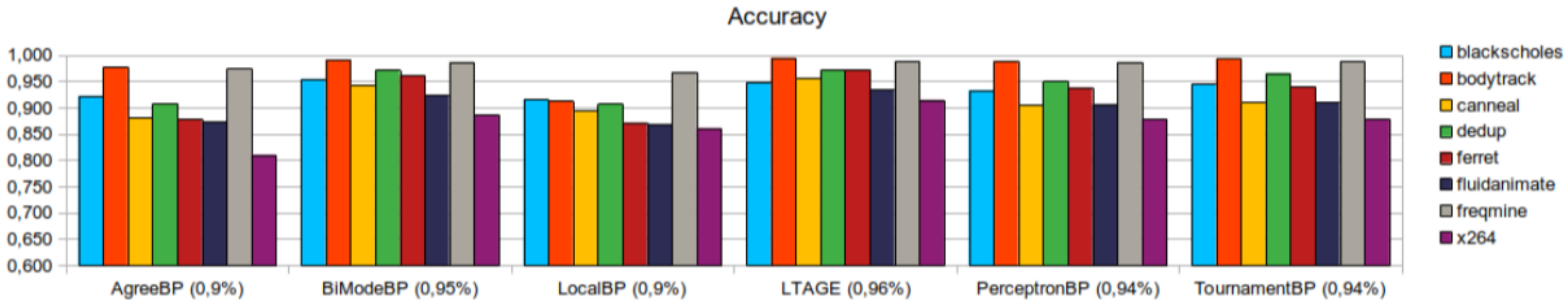
Agree vs Local



Agree predictor performs better than Local predictor
Not much average difference (implementation might need improvement?)

Comparison

All vs All



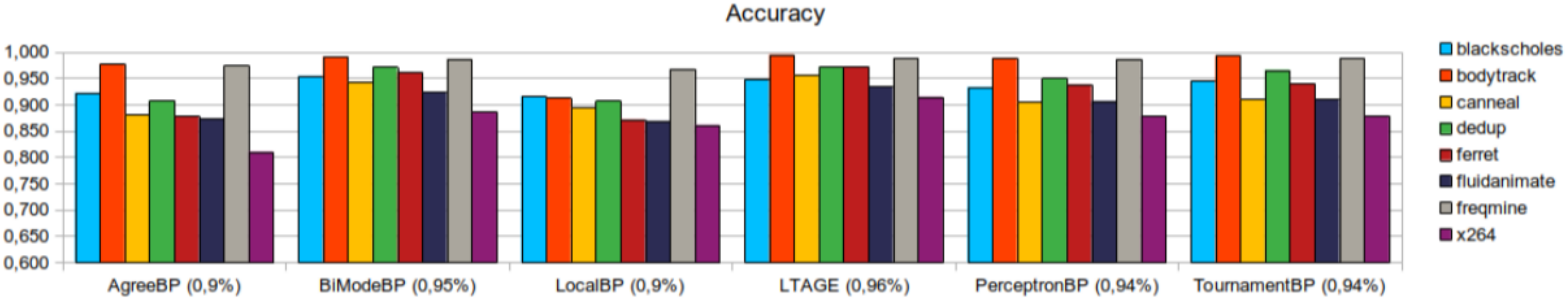
In some cases, new predictors outperform older ones

Increasing complexity leads to slightly longer prediction times

Perceptron predictor sometimes falls behind

Comparison

Best for every benchmark



BiMode is the best for blackscholes and dedup
LTAGE is the best for the rest

Improvements

Agree predictor

Better indexing function
(like hash function, mainly different to XOR)

Function with parameter autotuning
(perceptron for indexing function. Maybe overkill and inefficient?)

Different state machine for saturating counter
(as it's more a decision state instead of counter)

Improvements

Perceptron predictor

Longer branch history register lengths

Better training function
(taking hardware complexity into account)

Indexing function for addressing perceptron table
(instead of just branch address LSBs)

Thank you!

See project files and source code here:

https://github.com/SrGMC/tu-aca-project