Performance Modeling of the IBM Cell Broadband Engine Using Neural Networks

Tomofumi Yuki, Sanjay Rajopadhye, Chuck Anderson, Gautam Gupta

April 26, 2008

The Problem

Optimizations like tiling or loop unrolling have tuning parameters.

It is expensive to search the entire parameter space for the best set of parameters.

Performance models can be used to narrow down the search space.

Modeling the Cell BE is difficult because of its unique architecture. (heterogeneous, vector units, explicitly managed cache, ring-bus)

Using Performance Models

N = Problem Size, T = Tile Size
P = Performance, Y = Predicted Performance

Derive a function $f(N, T) = Y$, that predicts the performance given the problem and tile size.

Then find T that maximize Y.
If Y is accurately predicting P, the T found should have good performance.

Analytical Models

We tried analytically modeling the data using linear regression.

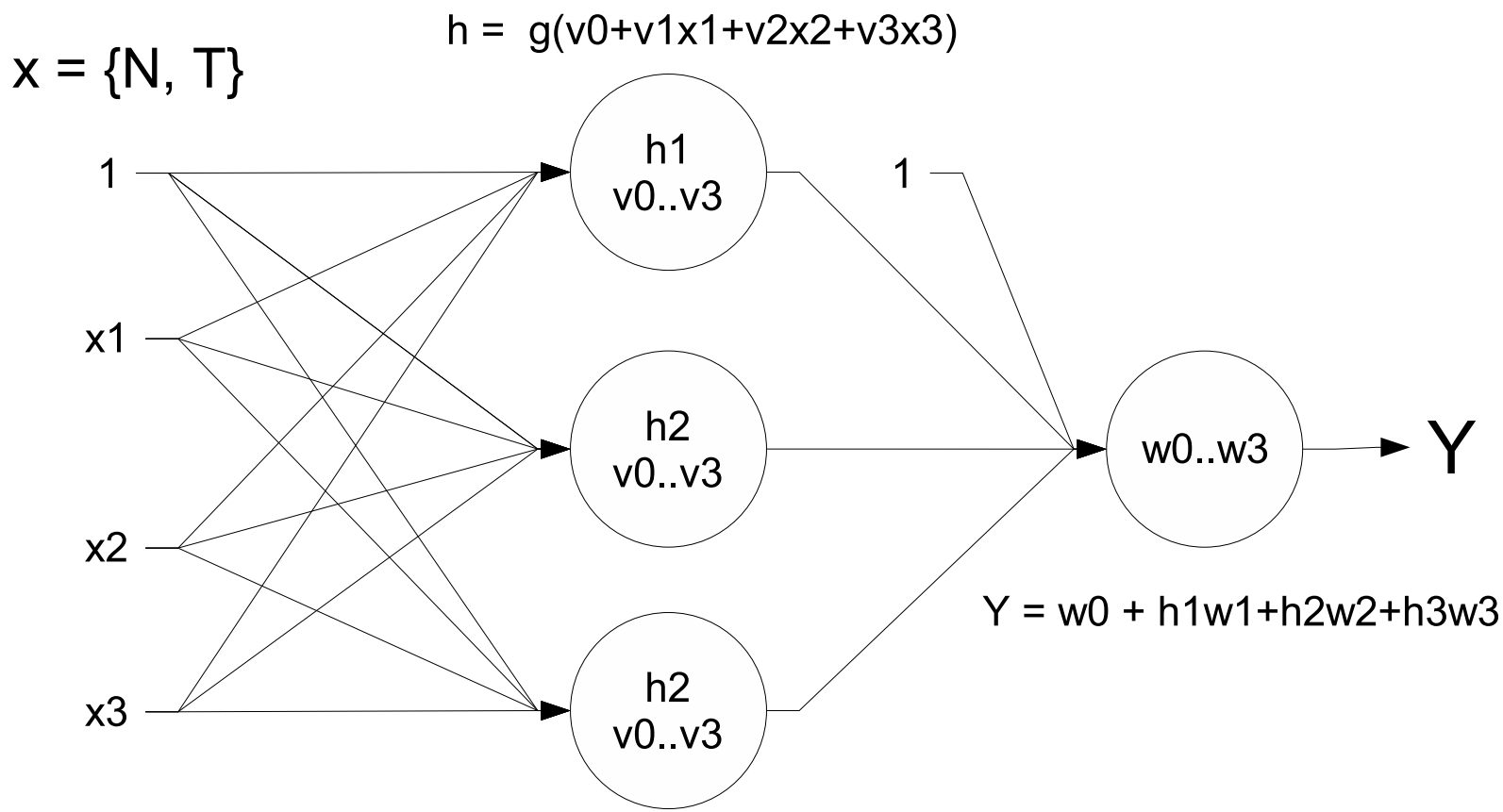
The program was modeled using N and T, and then linear regression was used to find the weights of each term that minimize the error.

$$X_i = \phi_i(N, T)$$
$$Y = w_0 + w_1 X_1 + \ldots w_n X_n$$

We were not able to find models that fit the data well even for matrix multiplication of fixed problem size.

Neural Network

Using N, T, P of the training data, minimize the difference between Y and P.



Local Search

Maximize the function $f(N, T)$ given N, by changing T.

Start from a T and climb the hill to find local maxima.

There could be many local maximas.

- Start from multiple places
- Post process Ts found to make it a multiple of 32
  - Need to be multiple of 8 to run
  - Smaller set of unique T
- Try all Ts found and pick the best

Changing The Target

The function tried to predict the GFLOPS achieved by the given parameters.

The target can be different, as long as the peak of the function matches the peak of the data.

Changing The Target

The function tried to predict the GFLOPS achieved by the given parameters.

The target can be different, as long as the peak of the function matches the peak of the data.

Normalize $(P_N / max(P_N))$

- Smaller difference between problem sizes.

Changing The Target

The function tried to predict the GFLOPS achieved by the given parameters.

The target can be different, as long as the peak of the function matches the peak of the data.

Normalize $(P_N/max(P_N))$

- Smaller difference between problem sizes.

Take the Square

- Emphasize the peak

Matrix Multiplication

Provided executable with tile size parameterized was used.
No access to the source code.

- Loop unroll factor was known because the tile sizes had to be a multiple of unroll factors.

3 problem size parameters, 3 tile size parameters.
Neural networks were trained using half of the data collected from 216 problem sizes, and tested with the remaining half.

SONY Play Station 3 was used for experimentation.

- Only 6 SPEs available

Evaluation

- Predict tile sizes for a N using trained network
- Measure its actual performance on PS3
- Compare against the observed maximum for that N
- Repeat for all N in the test data
- Count the number of Ns with predicted performance higher than 90/80% of the observed maximum.

$P/max(P_N)$	T	T^2	$normalize(T)$	$normalize(T)^2$
> 0.9	58.33%	64.81%	73.77%	90.74%
> 0.8	80.56%	99.07%	98.46%	100%

- Normalizing is important
- Taking the square seems to help as well

Conclusions and Future Work

Optimal tile sizes predicted by neural networks show good performance for problem sizes not used for training.

Extending the model

- More levels of tiling
- Multiple programs
  - How to express problem sizes of different programs?

Local Search

- Better search
- Can we find the global maximum?