Performance Analysis of Machine Learning Algorithms With Different Thread and Data Affinity Approaches

Arthur M. Krause, Eduardo H. M. Cruz, Matheus S. Serpa, Phillipe O. A. Navaux

Universidade Federal do Rio Grande do Sul

{*amkrause, ehmcruz, msserpa, navaux*}*@inf.ufrgs.br*

Porto Alegre, September 4th, 2017

Overview

Motivation

Data and thread affinity is a powerful way of increasing performance in parallel workloads. We want to know the impact of those techniques on popular machine learning algorithms.

Machine Learning Algorithms

k-Nearest Neighbors

A point receives the label most frequently occurring between its k nearest neighbors.

Backpropagation Neural Network

Algorithm to train a neural network, changing the weights of each neuron until error is low enough.

Machine Learning Algorithms

k-Means

Clusters a data set in k clusters in multiple iterations.

Streamcluster

Finds clusters centers in a data set in order to reduce total distance between points and centers.

Thread Affinity

Compact

Placing neighbor threads as close as possible within the processor.

Scatter

Placing threads as far as possible.

Memory Affinity

Interleave

Memory is allocated using round robin on nodes.

Membind

Allocate only from one node.

Automatic NUMA balancing

Automatic NUMA balancing from the Linux kernel.

Testing Environment

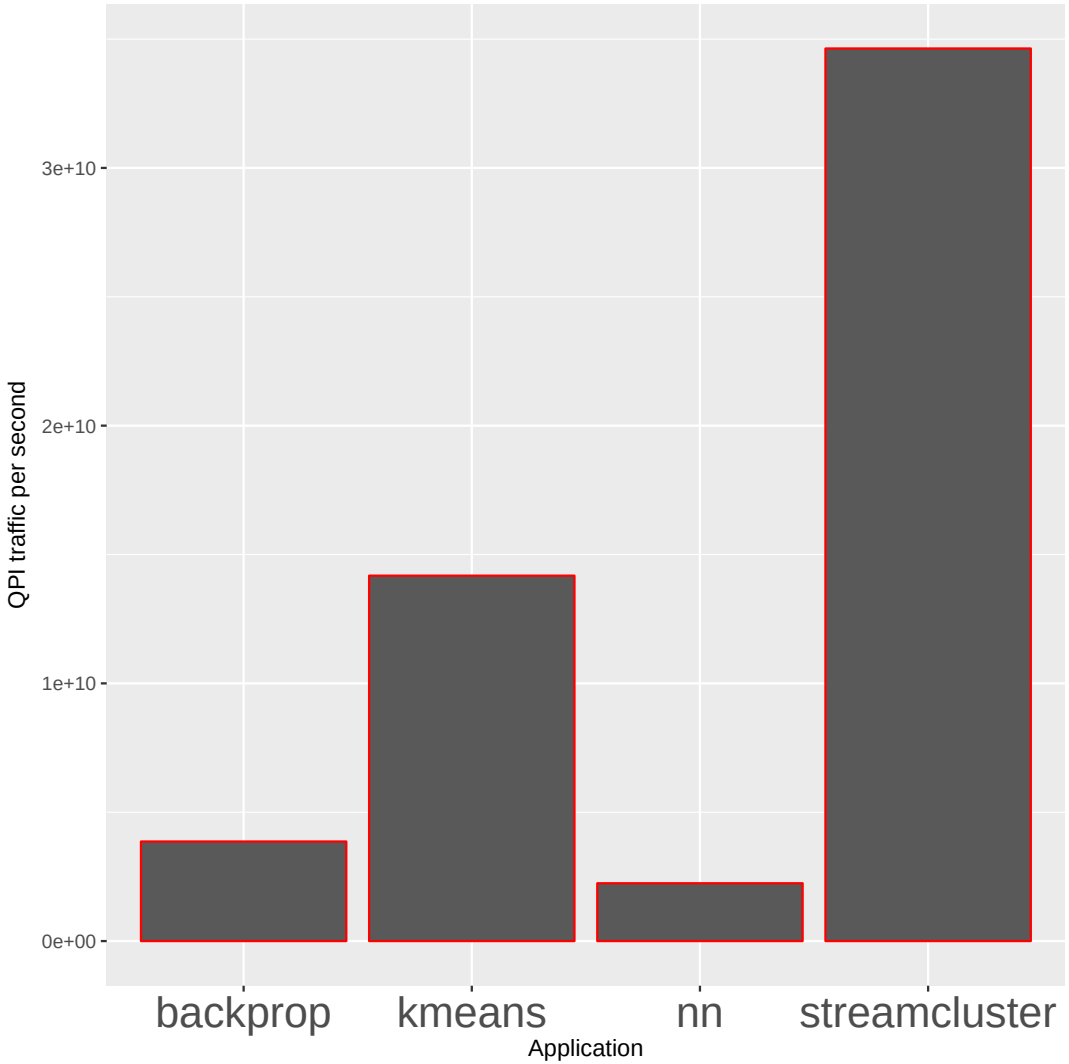
Turing

- 4x Intel Xeon X7550
  - 8 núcleos Nehalem EX @ 2.0GHz
  - 32KB de cache L1
  - 256KB de cache L2
  - 18MB de cache L3
- 31GB DDR3 per socket
- Ubuntu 16.04.3 LTS
  - Linux 4.4.0-83-generic

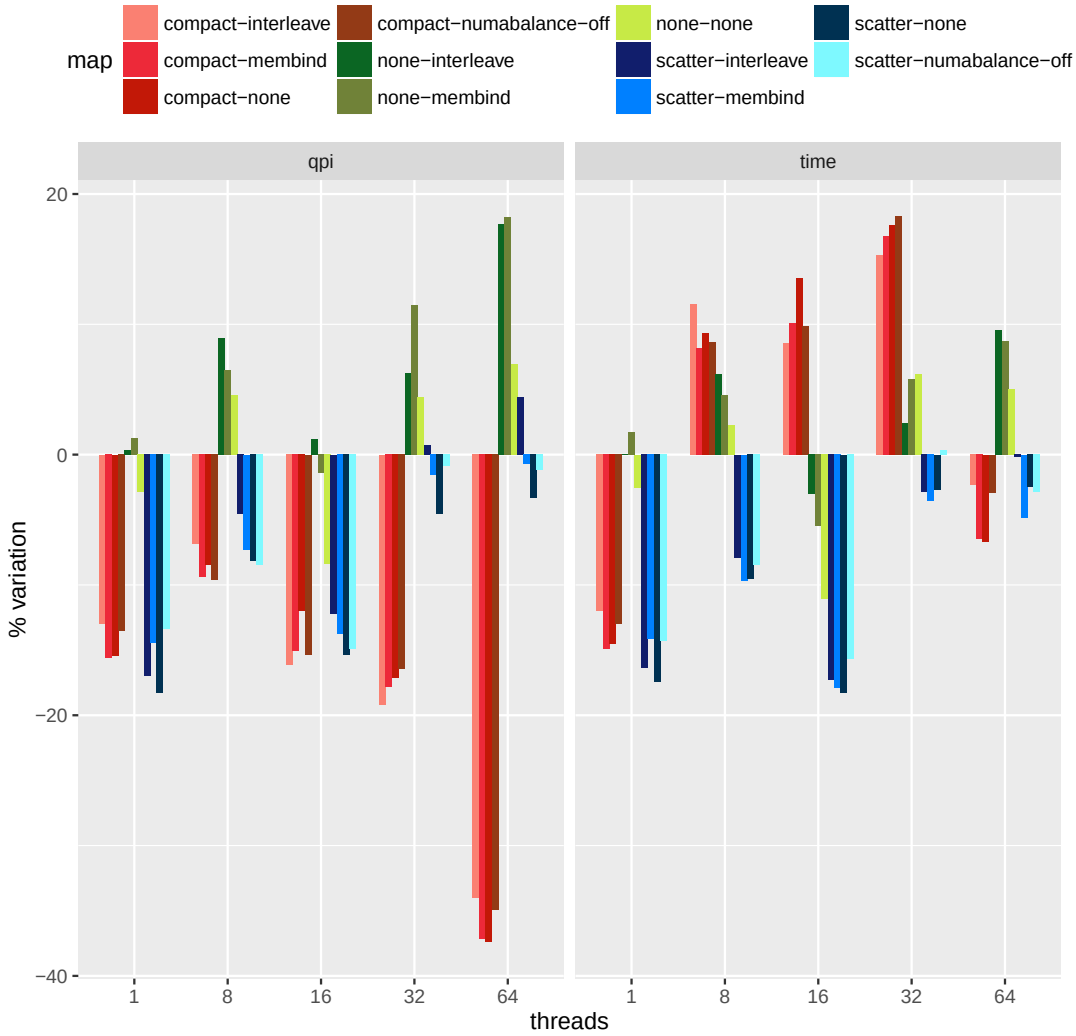
Measured Data

- PCM
  - Execution time
  - QPI traffic

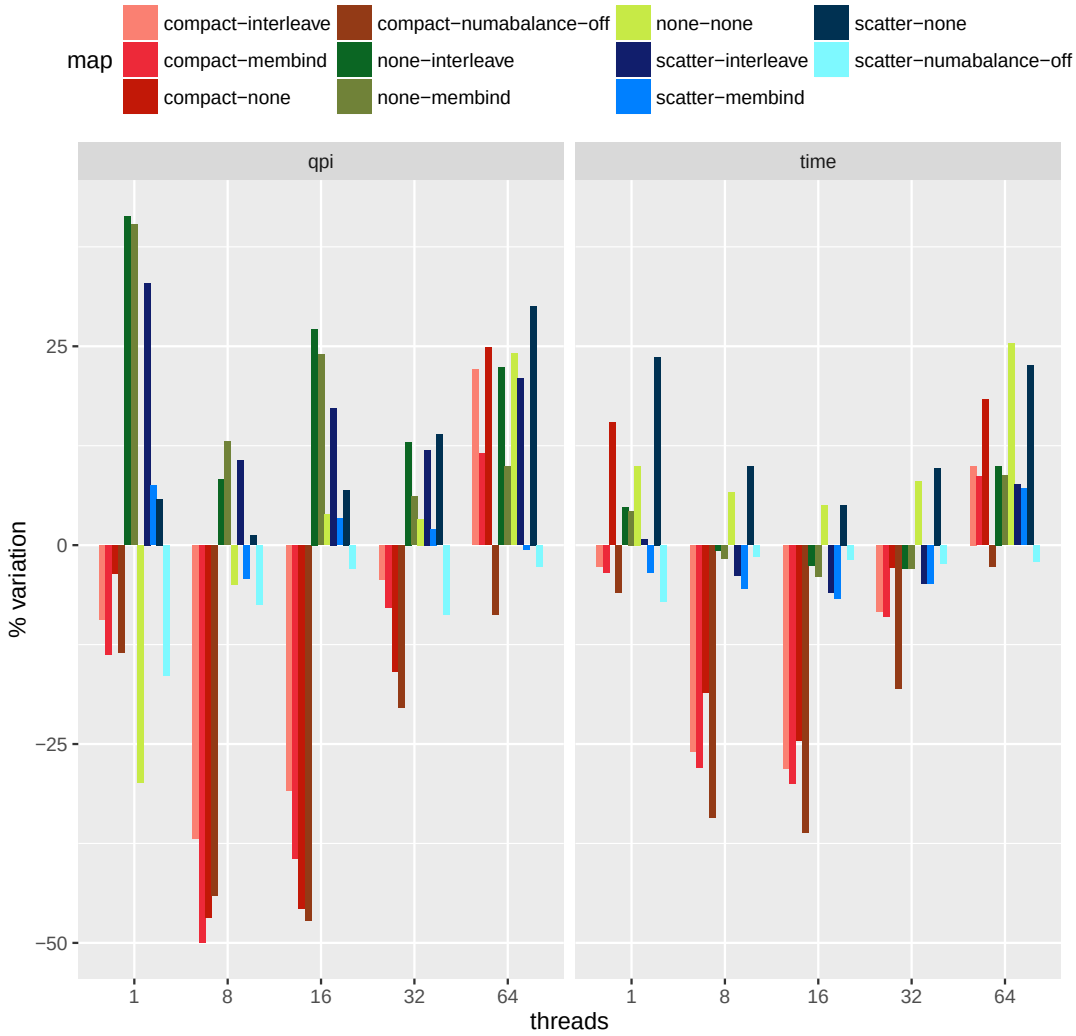
QPI traffic per second



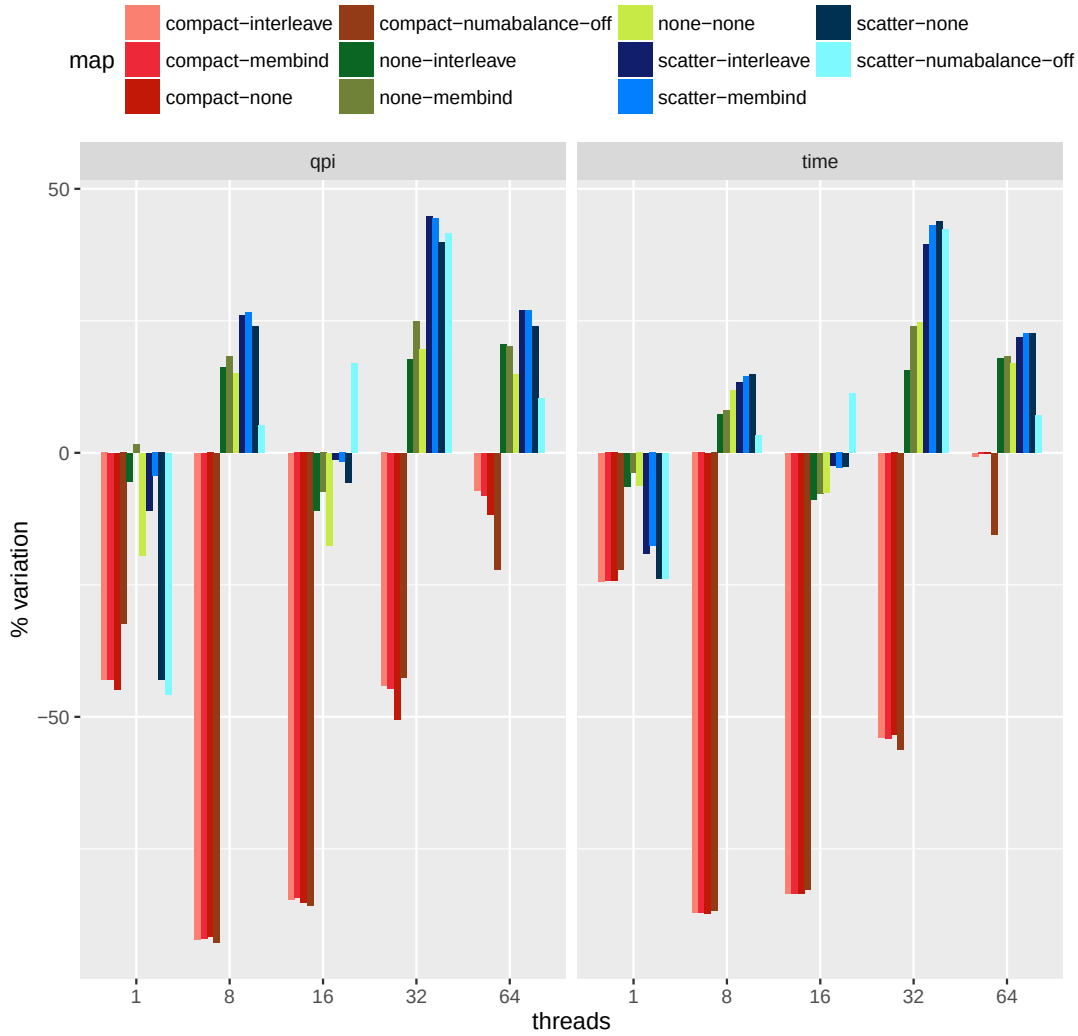
k-Nearest Neighbors



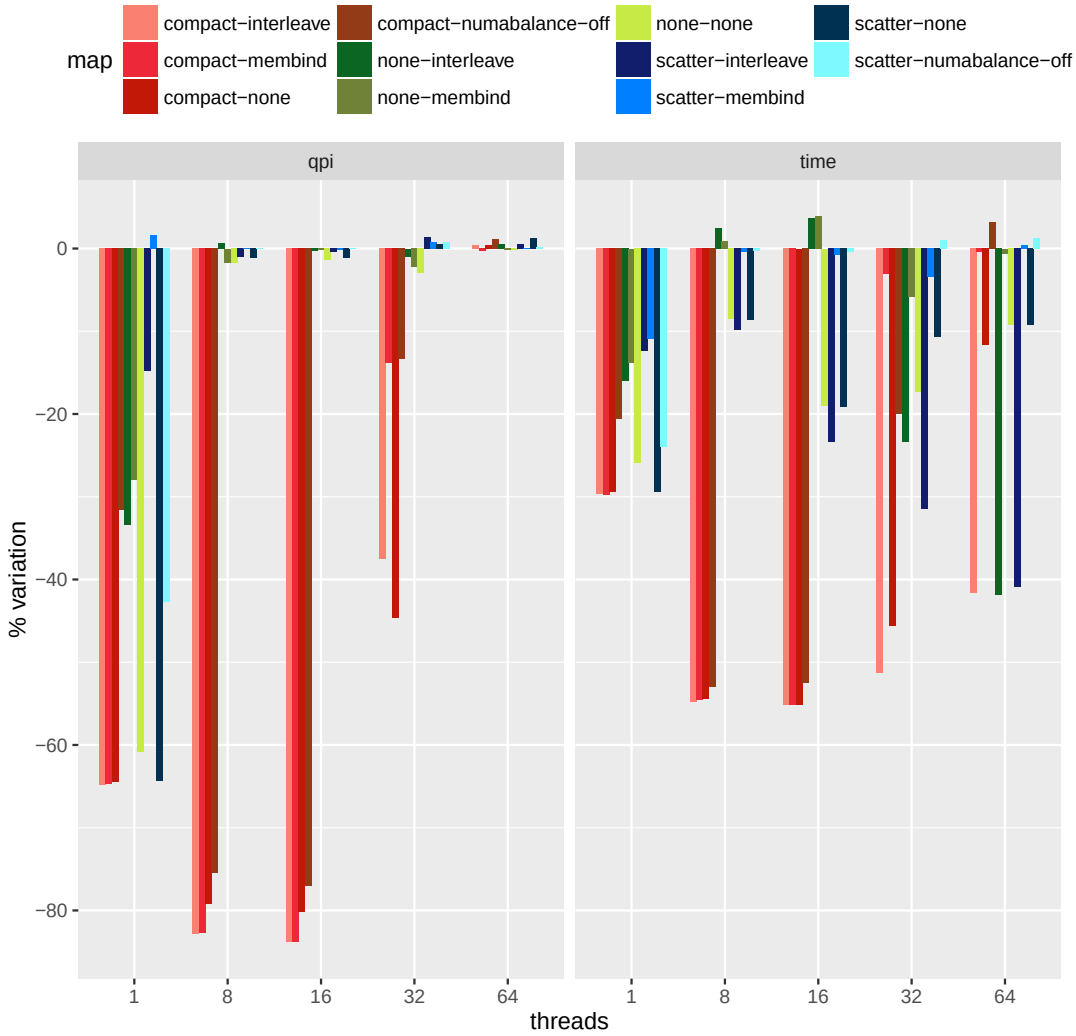
Backpropagation



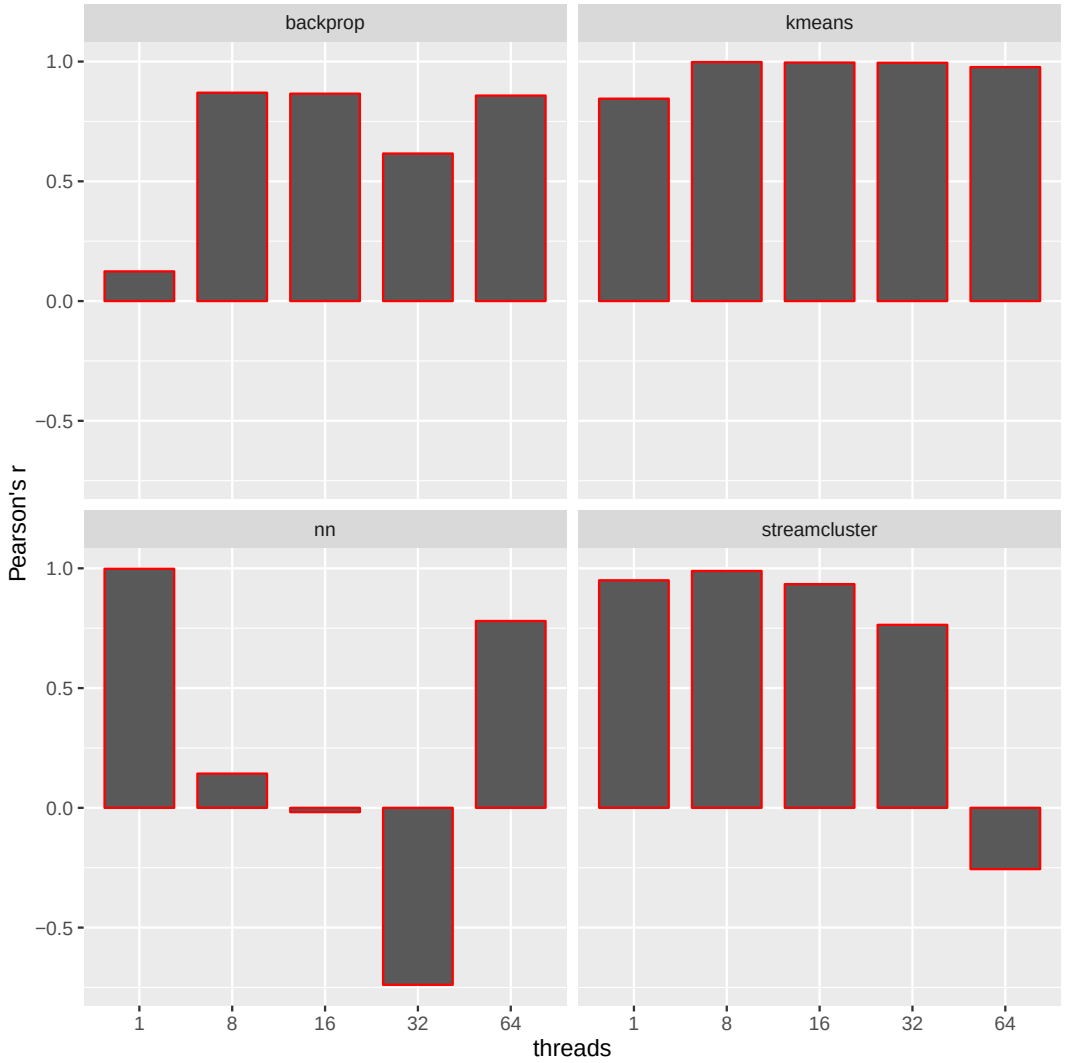
k-Means


map
compact−interleave
compact−membind
compact−none
compact−numabalance−off
none−interleave
none−membind
none−none
scatter−interleave
scatter−membind
scatter−none
scatter−numabalance−off
qpi
time
50
0
−50
% variation
1
8
16
32
64
threads

Streamcluster



Pearson's r


backprop
kmeans
nn
streamcluster
Pearson's r
1.0
0.5
0.0
−0.5
1
8
16
32
64
threads

Conclusion

1. We can achieve up to 87% execution time reduction with thread affinity in some cases.
2. Most performance gains are due to reduced communication.
3. More than 40% reduction in time with memory interleaving on Streamcluster.

amkrause@inf.ufrgs.br
msserpa@inf.ufrgs.br
ehmcruz@inf.ufrgs.br
navaux@inf.ufrgs.br