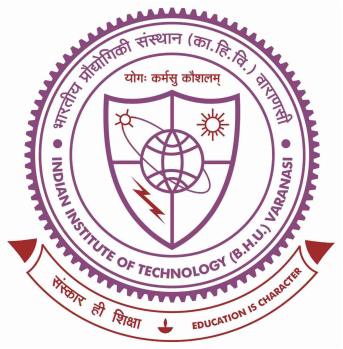
Some Methods for Performance Improvement of Multi-core clusters



Thesis submitted in partial fulfillment

for the Award of Degree

DOCTOR OF PHILOSOPHY

by

Navin Mani Upadhyay

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY

(BANARAS HINDU UNIVERSITY),

VARANASI-221 005

Roll No: 17071023

August 18, 2024

I dedicate this thesis to my beloved family, whose unwavering support and encouragement have been my constant source of strength and motivation throughout this academic journey. Their love and sacrifices have been the cornerstone of my achievements, and I am deeply grateful for their enduring faith in my pursuits. . .

Certificate

It is certified that the work contained in this thesis entitled "Some Methods for Performance Improvement of Multi-core clusters" by "Navin Mani Upadhyay" has been carried out under my supervision and that it has not been submitted elsewhere for a degree.

It is further certified that the student has fulfilled all the requirements of Comprehensive Examination, Candidacy and SOTA for the award of Ph.D. Degree.

19/8/24

Dr. Ravi Shankar Singh

Associate Professor

Department of Computer Science and Engineering

Indian Institute of Technology (Banaras Hindu University),

Varanasi-221 005

DECLARATION BY THE CANDIDATE

I, Navin Mani Upadhyay, certify that the work embodied in this thesis is my own bonafide work and carried out by me under the supervision of Dr. Ravi Shankar Singh from July 2017 to August 18, 2024, at the Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi. The matter embodied in this thesis has not been submitted for the award of any other degree/diploma. I declare that I have faithfully acknowledged and given credits to the research workers wherever their works have been cited in my work in this thesis. I further declare that I have not willfully copied any other's work, paragraphs, text, data, results, etc., reported in journals, books, magazines, reports, dissertations, theses, etc., or available at websites and have not included them in this thesis and have not cited as my own work.

Date : 19 August, 2024.

Place : Varanasi

Navin Mani Upadhyay

Signature of the Student

(Navin Mani Upadhyay)

CERTIFICATE BY THE SUPERVISOR

It is certified that the above statement made by the student is correct to the best of my/our knowledge.

19/8/24

Signature of Supervisor

(Dr. Ravi Shankar Singh)

पर्यवेक्षक/Supervisor
संगणक विज्ञान एवं अभियांत्रिकी विभाग
Department of Computer Sc. & Engg
भारतीय प्रौद्योगिकी संस्थान
Indian Institute of Technology
(काशी हिन्दू विश्वविद्यालय)
(Banaras Hindu University)
वाराणसी/Varanasi-221005

19/8/24

Signature of Head of Department

विभागाध्यक्ष/HEAD
संगणक विज्ञान एवं अभियांत्रिकी विभाग
Department of Computer Sc. & Engg
भारतीय प्रौद्योगिकी संस्थान
Indian Institute of Technology
(काशी हिन्दू विश्वविद्यालय)
(Banaras Hindu University)
वाराणसी/Varanasi-221005

COPYRIGHT TRANSFER CERTIFICATE

Title of the Thesis : **Some Methods for Performance Improvement of Multi-core clusters**

Name of the Student : **Navin Mani Upadhyay**

Copyright Transfer

The undersigned hereby assigns to the Indian Institute of Technology (Banaras Hindu University) Varanasi all rights under copyright that may exist in and for the above thesis submitted for the award of the "DOCTOR OF PHILOSOPHY".

Date : August 18, 2024
19

Place : Varanasi

Navin Mani Upadhyay.
Signature of the Student

(Navin Mani Upadhyay)

Acknowledgements

Foremost, I would like to thank my advisor **Dr. Ravi Shankar Singh**, for his continued support of my PhD studies and research. Whatever patience, inspiration, enthusiasm and immense knowledge I feel today is only the result of his constant support and inspiration that I have received every day. With his guidance, I have been able to do this research and writing.

Besides my advisor, I would like to thank the rest of my thesis committee (Research Progress Evaluation committee): **Prof. S.K. Singh (Head, CSE)**, **Dr. Bhaskar Biswas**, and **Dr. Ashok Ji Gupta**, for their encouragement, insightful comments, and hard questions, which incented me to widen my research from various perspectives. I also express my sincere thanks to all the faculty members and staff members of the department.

I thank my fellow labmates of Research Lab, and High Performance Computing Lab: **Dr. Shashank S. Singh, Dr. Vishal Srivastava, Dr. Ajay Kumar, Dr. Shivansh Mishra**, and **Dr. Swati Gupta**, for the stimulating discussions. Also, I thank my friends in IIT (BHU): **Dr. Medara Rambabu, Dr. Yaman Dua, and Mr. Vinod Kumar** for their help and cooperation.

In particular, I am grateful to my seniors **Dr. Dharmendra P. Mahato (Asst. Professor, CSE, NIT-Hamirpur) and Dr. Shri Prakash Dwivedi (Asst. Professor, IT, G.B.P.T.U.-Uttrakhand)** for enlightening me at the first glance of the research. They motivated me a lot and pushed me up at many stages on this journey toward my Ph.D. Their advice, encouragement and criticism were always the source of inspiration. This dissertation would not have been possible without their invaluable suggestions and persistent help.

Most importantly, my family is deeply grateful for their constant support, inspiration, guidance, and sacrifices. They were my constant source of motivation and inspiration. Their affection and guidance were instrumental in choosing Engineering and eventually continuing to my Ph.D. My special thanks to my parents, **Shri Jagdish Upadhyay** and **Smt. Gayatri Devi**, for giving birth to me in the first place and supporting me spiritually throughout my life.

Last but not least, I sincerely thank all of them who contributed to helping me see the light at the end of every scary tunnel during my Ph.D.

- Navin Mani Upadhyay

ABSTRACT

This thesis delves into various performance improvement techniques for multi-core clusters, which are critical for advancing High-Performance Computing (HPC). Multi-core clusters are fundamental in supporting complex computational tasks, and optimizing their performance can significantly enhance computational efficiency and system reliability.

The research begins by identifying key challenges in the HPC environment, particularly focusing on memory congestion, resource prediction, and load balancing. It introduces an innovative scheme designed to mitigate memory congestion, thereby improving data throughput in multi-core systems. This scheme employs advanced parallel data mining techniques on public datasets, allowing for the development of predictive models that can forecast CPU performance. These models are instrumental in facilitating more efficient scheduling and resource allocation.

A significant portion of the thesis is dedicated to exploring load balancing mechanisms. It evaluates various algorithms and their effectiveness in heterogeneous cloud environments, offering a comprehensive survey of existing techniques and proposing new methodologies. The proposed solutions are validated through extensive simulations and real-world applications, demonstrating substantial improvements in computational efficiency.

Additionally, the research addresses dependency prediction of long-term resource usage in HPC environments. By employing machine learning techniques, the study develops models that can predict resource usage patterns, thus enabling better resource management and allocation strategies.

The thesis also examines the impact of different scheduling algorithms on system performance. It introduces an effective scheme for memory congestion reduction and

evaluates its performance through detailed experimental analysis. The results indicate a significant reduction in memory congestion, leading to enhanced system performance and reliability.

Moreover, the research explores the integration of advanced load balancing techniques with existing HPC infrastructures. It proposes a hybrid approach that combines multiple algorithms to optimize load distribution across multi-core clusters. This approach is shown to improve overall system performance, reduce processing time, and enhance system scalability.

In conclusion, this thesis makes significant contributions to the field of HPC by providing scalable and adaptive methods for improving the performance of multi-core clusters. The findings pave the way for the development of more robust and efficient computing infrastructures, which are essential for meeting the growing demands of complex computational tasks. These advancements are expected to have a profound impact on the future of HPC, enabling more efficient and reliable computational environments.

Contents

Certificate ii

Abstract v

Acknowledgements vii

Contents ix

List of Figures xiii

List of Tables xv

Abbreviations xvii

Symbols xxi

1 Introduction 1
- 1.1 Motivation and Challenges . . . 2
- 1.2 Research Objectives . . . 5
- 1.3 Research Methodology . . . 5
- 1.4 Thesis Overview . . . 6
  - 1.4.1 Capability-Based Classification . . . 7
    - 1.4.1.1 Resource-Based Clusters . . . 7
    - 1.4.1.2 Software-Based Clusters . . . 8
    - 1.4.1.3 Infrastructure-Based Clusters . . . 8
  - 1.4.2 Performance-Based Classification . . . 9
    - 1.4.2.1 Selector-Based Scheduling . . . 9
    - 1.4.2.2 Predictor-Based Scheduling . . . 9
- 1.5 Challenges of multicore clustering research Work . . . 10
- 1.6 Classification of Related Work . . . 10

1.7 Contribution and Impact 11
1.8 Chapter Organization 13

2 Related Work **15**
2.1 Introduction 17
2.2 Memory Congestion in Multi-Core Systems 21
2.2.1 Overview 21
2.2.2 Memory congestion Problem 21
2.2.3 Challenges in Memory Management 21
2.2.4 Memory Congestion Reduction Techniques 22
2.2.5 Cache Partitioning Algorithms 25
2.2.5.1 Static Cache Partitioning 25
2.2.5.2 Both Static and Dynamic Cache Partitioning 25
2.2.5.3 Pseudo-Partitioning 26
2.2.5.4 Both Pseudo and Strict Partitioning 26
2.2.5.5 Way-Based Partitioning 26
2.2.5.6 Set/Color-Based Partitioning 26
2.2.5.7 Block-Level Real-Coded Genetic Algorithm (GA) . . . 27
2.2.6 Optimization Techniques 27
2.2.6.1 Regression and Curve Fitting Optimization 27
2.2.7 Dynamic Programming 27
2.2.8 Feedback Control Theory 28
2.2.9 Gradient-Descent Algorithm 28
2.2.10 Machine Learning Techniques 28
2.2.10.1 Machine Learning 28
2.2.11 Real-Time Capabilities 29
2.3 Resource Allocation in Multi-Core Cluster Environments 29
2.3.1 Overview 29
2.3.2 Challenges in Resource Allocation 30
2.3.3 Advanced Resource Allocation Strategies 31
2.3.4 Mining on CPU Datasets 32
2.3.5 Performance Prediction 33
2.3.6 Resource Allocation and Workload Characterization 34
2.4 Load Balancing in Multi-Core Cluster Environments 38
2.4.1 Overview 38
2.4.2 Load Balancing Techniques 41
2.4.3 Dynamic Load Balancing 43
2.5 Integration of Memory Congestion, Resource Allocation, and Load Balancing 47
2.6 Summary 49

3 Performance Prediction of Multi-core Systems **51**
3.1 Introduction 52

3.2 Proposed Framework . . . 54
3.2.1 Proposed EM-based Clustering Algorithm . . . 58
3.2.2 Applying EM-based Clustering Algorithm . . . 61
3.3 Result and Discussion . . . 62
3.3.1 Prediction Accuracy: Case Study . . . 62
3.3.1.1 Case 1: Prediction for new SKUs . . . 63
3.3.1.2 Case 2: Self-prediction . . . 65
3.3.1.3 Case 3: Cross-prediction . . . 69
3.3.2 Ranking of SKUs based on selected Cases . . . 70
3.3.2.1 Case 2: Self-prediction . . . 70
3.3.2.2 Case 3: Cross-prediction . . . 71
3.3.3 Similarity measurement and Validation . . . 73
3.3.3.1 Micro-architecture . . . 73
3.3.3.2 Statistical Analysis of Benchmarks . . . 75
3.3.3.3 Benchmark subsets Validation . . . 76
3.3.4 Statistical Test and Comparison . . . 77
3.4 Summary . . . 79

4 Memory Congestion Reduction in Multi-core Systems 81
4.1 Introduction . . . 81
4.2 Proposed Algorithm . . . 84
4.2.1 Gathering the node information and communication pattern . . . 84
4.2.2 Reducing congestion with load balancing (RCLB) Algorithm . . . 86
4.3 Applying The Algorithm . . . 88
4.3.1 Gathering the node information and communication pattern . . . 88
4.3.2 Reducing congestion with load balancing (RCLB) Algorithm . . . 89
4.4 Simulation and Result Analysis . . . 90
4.4.1 Results and Evaluation . . . 91
4.4.1.1 Simulation setup . . . 92
4.4.1.2 Simulation Verification . . . 92
4.4.2 Performance comparison of simulated results . . . 93
4.4.3 Performance comparison of NPB kernels . . . 95
4.4.4 Performance measure based on locality and congestion . . . 99
4.5 Summary . . . 100

5 Dependency Prediction of Long-Term Resources 101
5.1 Introduction . . . 102
5.2 Proposed Algorithm . . . 103
5.2.1 Overview of Ensemble Algorithm . . . 104
5.2.2 Performance Evaluation of Proposed Algorithms . . . 110
5.3 Experimental Setup and Simulation . . . 114
5.3.1 Phold . . . 115

5.3.2 Social Opinion Systems (SOS) 116
5.3.3 Data Set Description 118
5.4 Results and Discussions 119
5.4.1 Experimental Setup 119
5.4.2 Experimental Results and Analysis 121
5.5 Summary 125

6 Conclusion and Future Directions **127**
6.1 Conclusion 127
6.2 Future Research Directions 129

Bibliography **131**

A List of Publications **145**

List of Figures

2.1 Taxonomy of HPC based on multi-core clusters 15
2.2 Classification of load balancing techniques based on HPC environment . . 39

3.1 Connection-topology of used machines. 52
3.2 Working methodology for designing of virtual cluster. 55
3.3 Standard mean of EM algorithm based on six data clusters. 57
3.4 Standard deviation of EM algorithm based on six data clusters. 57
3.5 Stock Keeping Units (SKUs) breakdown in the Standard Performance Evaluation Corporation (SPECs) dataset 64
3.6 Stock Keeping Units (SKUs) breakdown in Geekbenchs datasets. 64
3.7 The Performance comparison of DNN and LR algorithms in terms of mean absolute error (MAE) for selected SPEC. 65
3.8 The Performance comparison of DNN and LR algorithms in terms of mean absolute error (MAE) for selected Geekbench 66
3.9 The Mean Absolute Error of selected SKU based on DNN for Geekbench (Self prediction after adding 50 SKUs). 66
3.10 The Mean Absolute Error of selected SKU based on DNN for SKUs (Self prediction after adding 50 SKUs for case-2). 67
3.11 Principal Component Analysis (PCA) of Standard Performance Evaluation Corporation (SPECs) workload performance based on SPEC FP and SPEC Int. 68
3.12 Cross-prediction of SPEC workloads, predicted with the Geekbench dataset 68
3.13 Cross-prediction of Geek-bench workloads, predicted with the SPEC dataset. 69
3.14 SKUs ranking results of self prediction (case-1) for Standard Performance Evaluation Corporation (SPEC) 71
3.15 SKUs ranking results of self prediction (case-1) for Geekbench. 71
3.16 Ranking comparison based on existing test approaches over Standard Performance Evaluation Corporation (SPEC) for self (case-1) and cross prediction (case-3.) 72
3.17 Ranking comparison based on existing test algorithms over Geekbenchs for self (case-1) and cross prediction (case-3.) 73
3.18 Relative execution time of selected SKUs. 77

4.1 Execution time (in ms) ratio of selected kernels and proposed algorithm (3) RCLB. 91
4.2 Result verification of selected kernels and proposed algorithm for memory congestion effect for MG Kernel. 91
4.3 Execution time ratio of RCLB on Conjugate Gradient (CG) kernel. 93
4.4 Execution time ratio of RCLB on Fault-Tolerance (FT) kernel. 93
4.5 Execution time ratio of RCLB on Lower Upper (LU) factorization Kernel 95
4.6 Execution time ratio of RCLB on Multi-Grid (MG) Kernel 95
4.7 Average execution time(in sec) of the NPB kernels for class C problems . 95
4.8 Average execution time (in sec) of the NPB kernels for class D problems . 95
4.9 Normalized execution times of the NPB kernels for class C problems . . . 97
4.10 Normalized execution times of the NPB kernels for class D problems . . . 97

5.1 Flow diagram of dataset processing and passing through the CPU cores. . 104
5.2 Feature distribution based on number of SPECs (CGs) and time domain (loading time) 105
5.3 Sequential execution of uniformly distributed datasets over selected algorithms (SVM, KNN, DT, GBT, and Naive Bayes) 113
5.4 Sequential execution of randomly distributed datasets over selected algorithms (SVM, KNN, DT, GBT, and Naive Bayes) 113
5.5 Parallel execution of uniformly distributed datasets over selected algorithms (SVM, KNN, DT, GBT, and Naive Bayes) 113
5.6 Parallel execution of randomly distributed datasets over selected algorithms (SVM, KNN, DT, GBT, and Naive Bayes) 113
5.7 Phold structure with n simulation objects 115
5.8 Social opinion system structure 118
5.9 Long-term running resource prediction structure. 120
5.10 Performance comparison of intelligent ensemble algorithms with k-feature dimensions (ARE, Accuracy, F-1 Score). 122
5.11 Feature dimension selection core based on RSME. 122
5.12 Absolute root mean error of proposed algorithm 122

List of Tables

2.1 Characteristics of the category of the algorithm in the literature for memory congestion reduction technique . . . 24
2.2 Related Work on Parallel and Multicore Data Mining Systems . . . 32
2.3 Comparison of resource allocation and workload distribution across various hardware accelerator architectures . . . 36
2.4 Comparison of Load Balancing Techniques in Multi-Core Cluster Environments . . . 42
2.5 Summary of Load Balancing Techniques . . . 46
2.6 Summary of Cluster-Based Load-Balancing Techniques . . . 48

3.1 Result based on the mean and standard deviation of EM algorithm based on six data clusters. . . . 56
3.2 Workloads used in our experiments. . . . 63
3.3 Standard list of used SPEC CPUs . . . 75
3.4 CPU configuration specifications . . . 77
3.5 The Friedman test on Mining erasable item-set MEI and MEIs based on multi-core processors pMEI based method for running time . . . 78
3.6 The estimation of p-value based on Holm procedure (unadjusted) for post-hoc analysis . . . 78

4.1 Mapping of processes of used placement methods for MG kernel in the order of (node:core) . . . 96
4.2 Traffic congestion effects over selected approaches (Packed and Socket-span) 99

5.1 Comparison of different normalization algorithms, based on accuracy, precision, recall and F1-score for SPEC dataset. . . . 112
5.2 Benchmark with the number of instances used for simulation . . . 115
5.3 A detailed configuration of selected multi-cores for virtual cluster . . . 119
5.4 Performance comparison of the proposed algorithm with other selected algorithms from different benchmark . . . 123
5.5 Performance comparison between major resource prediction methods and proposed method . . . 124

Abbreviations

ALU	**A**rithmetic and **L**ogical **U**nit
AMD	**A**dvanced **M**icro **D**evices
ARE	**A**bsolute **R**oot **M**ean **E**rror
ARRE	**A**ccuracy and **R**elative **R**untime **E**rror
BFS	**B**est **F**irst **S**earch
BN	**B**ayesian **A**pproach
CACH	**C**ache **M**emory (in KBs)
CG	**C**onjugate **G**radient Kernel
CHMAX	**M**aximum **C**hannels (in Groups)
CHMIN	**M**inimum **C**hannels (in Units)
CPU	**C**entral **P**rocessing **U**nit
DIV	**D**ivision
DMA	**D**irect **M**emory **A**ccess
DNN	**D**eep **N**eural **N**etwork
DT	**D**ecision **T**ree
EM	**E**xpectation **M**aximization
FBCA	**F**eature **B**ased **C**apatibility **A**lgorithm
FCFS	**F**irst **C**ome **F**irst **S**erve
FFT	**F**ast **F**ourior **T**ransformation
FNN	**F**uzzy **N**eural **N**etwork
FP	**F**loating **P**oint
FT	**F**ault **T**olerance Kernels

FU	**F**unctional **U**nits
GBT	**G**radient **B**oosting **T**echniques
GEM	**G**aussian **E**xpectation **M**aximization
HPC	**H**igh **P**erformance **C**omputing
I	**I**nteger based
IPC	**I**nter **P**rocess **C**ommunication
KNN	**K** **N**earest **N**eighbor
L1	**L**ayer-**1**
L3	**L**ayer-**3**
LLC	**L**ast **L**ayer **C**ache
LP	**L**inear **P**rogramming **M**odel
LR	**L**ogistic **R**egression
LU	**L**ower **U**pper Kernel
MAE	**M**ean **A**bsolute **E**rror
MCI	**M**emory **C**ontroller **I**mbalancing
MEI	**M**ining **E**rasable **I**tem-set
MG	**M**ulti **G**rid Kernel
MMAX	**M**aximum **M**ain **M**emory (in KBs)
MMIN	**M**inimum **M**ain **M**emory (in KBs)
MPI	**M**essage **P**assing **I**nterface
MUL	**M**ultiplication
MYCT	**M**achine **C**ycle **T**ime (in Sec)
NAS	**N**umerical **A**erodynamic **S**imulation
NIC	**N**ode **I**nformation **C**ommunication
NPB	**N**AS **P**arallel **B**enchmark
NUMA	**N**on-**U**niform **M**emory **A**rchitecture
PC	**P**ersonal **C**omputer
PCA	**P**rinciple **C**omponent **A**nalysis
PDF	**P**robabiltiy **D**enisity **F**unction

PEDS	**P**arallel and **D**iscrete **S**imulation **E**nvironment
PMEI	**M**ulti-core based **M**ining **E**rasable **I**tem-set
QoS	**Q**uality **o**f **S**ervices
RCLB	**R**educing **C**ongestion **W**ith **L**oad **B**alancing
RMSE	**R**oot **M**ean **S**quare **E**rror
SD	**S**tandard **D**eviation
SKU	**S**tock **K**eeping **U**nits
SMPI	**S**imulator for **M**essage **P**assing **I**nterface
SOS	**S**ocial **O**pinion **S**ystem
SPEC	**S**tandard **P**erformance **E**valuation **C**orporation
SVM	**S**upport **V**ector **M**achine

Symbols

M	Memory Bandwidth
i	Number of interconnections
$S(x)$	processing speed of a processor
$t(x)$	Execution time of problem
X	Total Performance of associate processors in a cluster
x	Individual Performance of interconnected processor
N	Number of Nodes
W	Total Data Transfer Direction
θ	Data Transfer Direction of a single Processor
μ	Ratio between performance of fast and slow CPU
c	Numeric coefficient for scaling the system
$A_{\mu}^{W}(N)$	Performance advantage of all Nodes (achievable gain)
L_p	Master fraction of Load and Processor
α, β	Normalize values
S_0	All possible solutions or slices
$L_{balance}$	Load balance
L_{ik}	Load on processor p
W_i	Weighted sum of load resources
I_1, I_2	Individuals to share the information.
M_{info}	Mutual Information
H_I	Entropy

D_I	Normal distribution
$Norm()$	Normalized Information
I_{contri}	Contribution of I candidate
f_{cap}	Feature capabilities
I_{select}	Selected feature
SD	Standard Deviation
μ	Mean
$E(I_{contri})$	Evaluation of contribution of I candidate
PBM	Predicted Base Model