Relating Reported Speedup to Attainable Speedup in HPC Applications Based on a Programmer's Expertise

Sarker Shakiur Rahman Shuvo[a], Waseem Ahmed[b]

[a] *sshuvo@stu.kau.edu.sa*, [b] *wabdalkayom@kau.edu.sa*
[a,b] *Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia*

Abstract

High Performance Computing (HPC) refers to the processing of complex computations using enhanced computing power for quick results and better accuracy. The parallel computing features of supercomputers provide an application program to obtain significant speedup over the serial implementation of certain problems in scientific computing. An HPC application largely relies on human expertise i.e., knowledge of the underlying architecture, algorithm used to solve the problem, his expertise in parallelizing the sequential code efficiently. Hence the speedup is largely dependent on the programmer's expertise. Therefore, an obtained speedup of a certain HPC application, may not always be the best possible speedup that is potentially possible on that supercomputer. So, there is always a need of finding out if the obtained speedup is the highest possible speedup for a given problem and on a given supercomputer architecture. That's why it's important to objectively rate a programmer's ability to relate it to his capability to parallelize a given code. Although, online judge platforms have given ratings to programmer's when they are tied to online judges (such as- Topcoder, Codeforce and Codechef), types of contests (long, short, national-level, international level and based on problem categories), and the programmers' sustained interest in participating (frequency of participation), problems solved, problems attempted, maturity (in years and capability) and other factors. The ranking could also be misleading in some cases; for example, in solving a difficult problem, a relatively new programmer may climb to a top position because of his familiarity with the category of the problem although he may have less submissions and experience. On the other hand, an experienced programmer can see a drop in his rank if he under-performs in one or more contests. Hence, there is no generalized way to rank a programmer based on a rating provided by an online judge platform. Therefore, we need a more sophisticated and reliable ranking model that can help us to evaluate the actual rating of a programmer. This work will involve researching existing ranking algorithms and develop a model that establishes a relationship between Programmers grade or level and the Maximum Attainable Speedup in a particular HPC application.

Keywords: Data Analysis; Machine Learning; Online Judge Platforms; High Performance Computing

1. Introduction

High Performance Computing (HPC), also known as super-computing, is the ability to process data and perform complex calculations faster using parallel processing which provides a higher speedup. Speedup in HPC can be defined as the ratio of the time taken by serial execution and the time taken for parallel execution. In our research, there are two kinds of speedups that will be dealt with: Reported Speedup and Maximum Attainable Speedup.

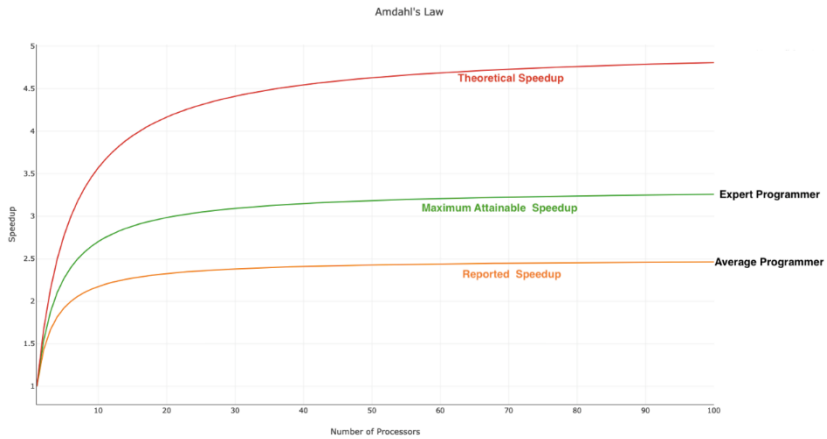

Figure (1): Correlation between Speedup and programmer's expertise

Reported speedup is the speedup as reported by the researcher (programmer). The highest point of speedup which can possibly be achieved on a particular system is the Maximum Attainable speedup which is the speedup that an expert programmer (best) can attain. Reported speedup is lower than or equal to the Maximum Attainable speedup. These speedups are different from theoretical speedup, which is defined by the Amdahl's Law [1].

A reported speedup of a particular problem executed in a HPC using a certain computing architecture may vary from programmer to programmer. The reported speedup can be considered as the maximum attainable speedup only if the program is run by a top programmer. So, there is a correlation between the maximum attainable speed and the level of the programmer. The research focuses on establishing this correlation by forecasting the programmers' expertise based on rating prediction from Codeforce's API, an online judge platform for programming contest.

The major outline of the paper: section 2 will be covering the related research on ranking techniques and predication algorithms used in machine learning followed by the methodology in section 3, Results and Analysis in Section 4 and conclusion and future work in the final Section.

2. Literature Review

An overview of related and relevant research on existing research papers in speedups in HPC (in section 2.1), ranking models and techniques (in section 2.2) and stock price predication and forecasting (in section 2.3) will be presented in the following sections.

2.1 Speedups in High Performance Computing

One of the main challenges of developing applications in the High-Performance Computing (HPC) is parallelization of serial code. There are many issues contributing to this specific challenge of HPC such as- the extent off parallelism in a program or application, sparse matrix multiplication (SpMV)[28], the

granularity or overhead in the partitioning of the processors and locality of communication and computation between processors and/or memories.

The performance of HPC applications is restricted by the sequential part even-if all the parallel parts are speedup perfectly. This is famously known as the Amdahl's Law. The parallel processing challenge of overhead caused by parallelism is known as granularity. This is one of the top barriers of getting the maximum attainable speedup for any HPC applications. The parallelism overhead may be caused by the time-taken for starting a thread or process, communicating shared data and synchronization. The location of data in the memory hierarchy has a significant impact on the speedup. This issue largely deals with load balancing, insufficient parallelism, high- latency on single process and so on. [2]

Moreover, in HPC, developing accurate computer application codes are among highly important issues. There are many factors directly affecting the advancement of HPC applications codes such as- parallel programming language and algorithm, domain knowledge, deep learning [29,30] and so on. The growth in the complexity of computer architecture due to massive parallelization has also contributed to the challenges of the HPC applications. With programs and data distributed across a huge number of different processors and separated memories, organizing the exchange of data and the order computations requires very complex logic, a lot of specialized programming and the best algorithm for maximum speedup. [3]

2.2 Ranking Models and Techniques

In recent years, we have witnessed the success of machine learning approaches to the document ranking problem, known as learning to rank (e.g., [3,4]). Moreover, we have seen the success of PageRank proposed and implemented by Google [5]. There are a number of significant ranking algorithms used in different sports and games such as ATP ranking in Lawn Tennis [6], ELO Rating in chess [7], ODI rating system in cricket [8,9] and so on.

For a successful rating of programmers, an expert ranking system is needed which is the core issue of expert information retrieval. Taking into consideration the complexity of feature redundancy in traditional dense list-wise Learning to Rank method and local optimum in parameter learning. The Expert list- wise ranking algorithm can be a great approach of ranking programmers as the feature dimension reduction can be achieved by the feature threshold from the loss-control function of sparse learning algorithm [10]. List-wise Neural Raking Models can be a unique approach for the coder's ranking. Conventional learning-to-rank models using pointwise or pairwise loss functions have generally shown lower performance compared to those using list- wise loss functions. In research, a list-wise neural ranker outperforms a pairwise neural ranking model [11].

Online recommender system can give us some important insights regarding programmer's ranking. Recommender systems allow rapid and automated customization and personalization of e-commerce sites. They allow the sites to generate more sales by tailoring to the needs of the visitors and turning them into consumers, up-selling extra products by bundling closely related things together and increasing customer loyalty. The input to a Recommender System depends on the type of the employed filtering algorithm [12].

Ranking methodology used various sports can also be taken as point of inspiration for this research. Sports like Tennis has a very sophisticated ranking technique. The ranking method is based on linear algebra, and one computes a score for each player by solving a certain linear system of equations – from these scores one

finds the ranking. The input is a set of matches, and weights representing the importance of the matches; this is represented by a weighted directed graph. The programmers may not be like tennis players, but the ranking technique used here can a unique way to come out with a new kind of Ranking methodology [13,14,15].

2.3 Stock Price Prediction

Stock price prediction is a huge research domain when it comes to predicting the programmer's rank. There are many attributes of this research that coincides with stock market forecasting. The stock prices cab be considered as the rating of the programmers. The stock opening price is like the initial rating of the contestants and closing price is the current ranking. The concept of slopes (gradient) is very crucial in both research domains since they show the ups and downs (trends) on the stock price as well as rating.

The conventional way to predict stock market is to use different machine learning and deep learning models using historic datasets. The predication models are first trained and then tested based on default and derived parameters. The most common machine learning techniques used for stock predication is time series predication model by autoregressive integrated moving average (ARIMA) [16], Regression and Classification Algorithms [17], Facebook Prophet [18], Random Forest [21]. Deep learning techniques such as Artificial Neural Network (ANN) [19,21], Long Short-Term Memory (LSTM) and Support Vector Regression (SVR) [20] are also commonly used for the stock predication.

In this research, the deep learning model implementation wasn't feasible since the dataset size wasn't large enough to apply ANN and other DL models. Although time-series models were successfully used in stock market predication but in our research of predicting programmer's ranking time-series had some fundamental implications. Since time-series models deals with stock market that changes in micro-timeframes i.e., in minutes or hours or days but the ranking change occurs after every event which happens in larger timeframes i.e., in months or even years. The change in stock price values (slope) is very minimal whereas the slope for the rating possesses substantial value. For these reasons, instead of time-series models, the conventional machine learning models, and techniques such as Liner Regression (LR), Random Forest (RF), Decision Tree (DT), Gradient Boosting Regression (GBR) were used for programmer's rating predication.

3. Methodology

This section describes the methodology used for ranking predication implementing several machine learning approaches.

Summary of Methodology is as followed:

- The dataset was collected from Codeforce Application Programming Interface (API)
- The dataset was preprocessed by data cleaning and transformation in Order to facilitate machine learning
- Important features were extracted from the dataset and were used to create new features
- The whole dataset was divided into training (80 percent) dataset and testing (20 percent) dataset
- The training dataset was used to train the ML models
- The testing dataset was used for predicting the rating

- Different error metrics were used measure the accuracy of the model

The overview of the research methodology is depicted through this block diagram:

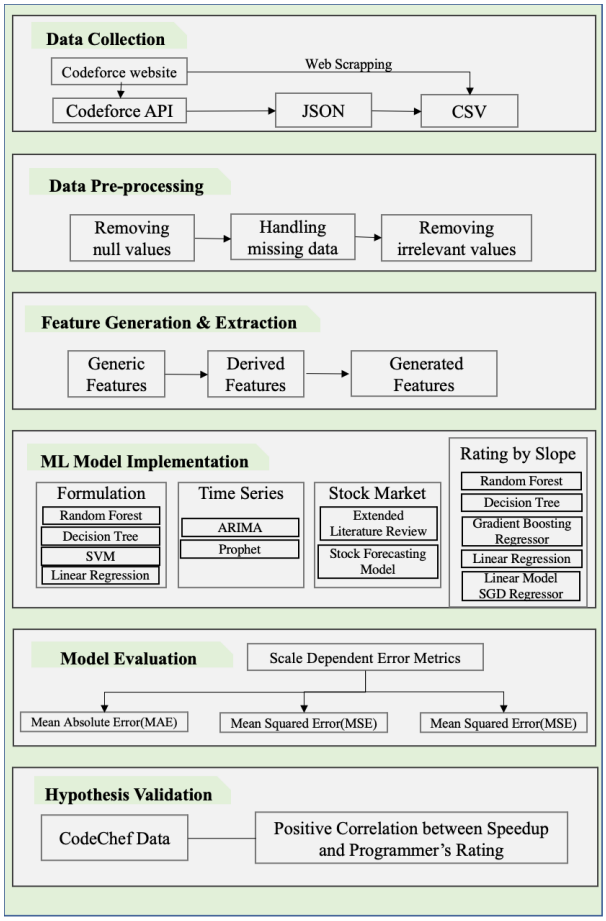

Figure (2): Block Diagram of the Methodology

The following sub-sections will describe the methods used in this research in details:

3.1 Data Acquisition

The primary source of the date used in this research is the Official API from Codeforce which are publicly available to access and download. The API provided the dataset in machine-readable JSON (JavaScript Object Notation) format which was converted into CSV (Comma Separated Values) format. [22]

3.2 Data Preprocessing

This is an integral preprocessing step where data is cleaned by filling the missing values, flattening the noisy data, and correcting, repairing, or removing incorrect or irrelevant data from the dataset. These are some of the techniques implemented in this research:

- **Handling Missing Data:** A specific row having less than 25 percent important data, was deleted from the dataset. For some specific cases, if a user has some missing key features such as current rating or number of participations, the entry was eliminated. The missing data wasn't filled using mean, mode, or median values of respective columns since all participants are unique and their expertise vary from contest to contest.
- **Removing Irrelevant Data**: A large quantity of irrelevant meaningless data are present in the main file. Since our research focus was programmer's performance based on current rating and frequency of participation, information regarding user's age, country of origin, institute they work for, gender, email address and such data has no value. So, the whole column of this sort of data was deleted from the main csv file.

3.3 Feature Creation

To implement the dataset for training and testing, new features were introduced. Individual user's list of current rating against the events was fetched from the API. The minimum participation of each user was five (5) since a user's true judgement of programming expertise can't be measured by first 2-3 contests.
These are three types of the features used:

- **Generic Feature:** The feature directly acquired from the original dataset from Codeforce API.
- **Derived Feature:** The features which were obtained from the generic features are called derived feature.
- **Generated Feature:** The features which are generated from scratch using the generic and derived features, will be considered as newly generated feature.

This is a table of features used in the research:

Feature	Nature	Remarks
Current Rating	Generic	Received from the Codeforce API
Max Rating	Derived	Taken in unchanged form from the original Data-set using Max( )
Average rating	Derived	Taken in unchanged form from the original Data-set using Mean( )
Average Slope	Generated	Average Slopes of Rating from 10th to 40th Contes
Max Rating50	Generated	Maximum Rating on the first 50th contests
Rated Slope	Generated	(MaxR-1600)/50 [1600: Initial Rating given to new programmer in 1st contest]
Average Slope	Generated	(Avg R-1600)/50 [Avg rating is a derived Parameter]

Table (1): Table of Features used

3.4 Validation of Generated Features

Three new features were created in order to predict the rating of a programmer. These newly made features were also used to train and test the dataset. These are some reasons behind the choice of the “New” features:

- **Average Slope:** The slope is defined as the "change" of rating with respect to the time. Since we are considering the first 50 contests, the average slope calculation will only confined between 10th to 40th contest. This is due to the fairness of the contest ratings since most of the contestants are not serious regarding the contests at the very beginning. And also the initial rating of 1400/1600 is by default for all contestants.
- **Max Rating50:** Since our training experiment is based on the first 50th contests, the maximum rating of the 1st first 50 contests has significant effect on the overall rating of a programmer participating in Codeforce.
- **Rated Slope:** This new feature demonstrates the ratio between the rating difference of overall Max Rating of a programmer and the default rating of 1600 and fixed number of participated contests for data training i.e. 50. This feature shows the actual change of the ratings of a programmer in terms of "Slope" in respect of a fixed time frame.
- **Average Slope:** This is a similar kind of feature as Rated Slope where instead of Max Rating , the overall Average Rating of the programmer will be used. This feature portraits the relationship of the rating changes with respect to the average rating.

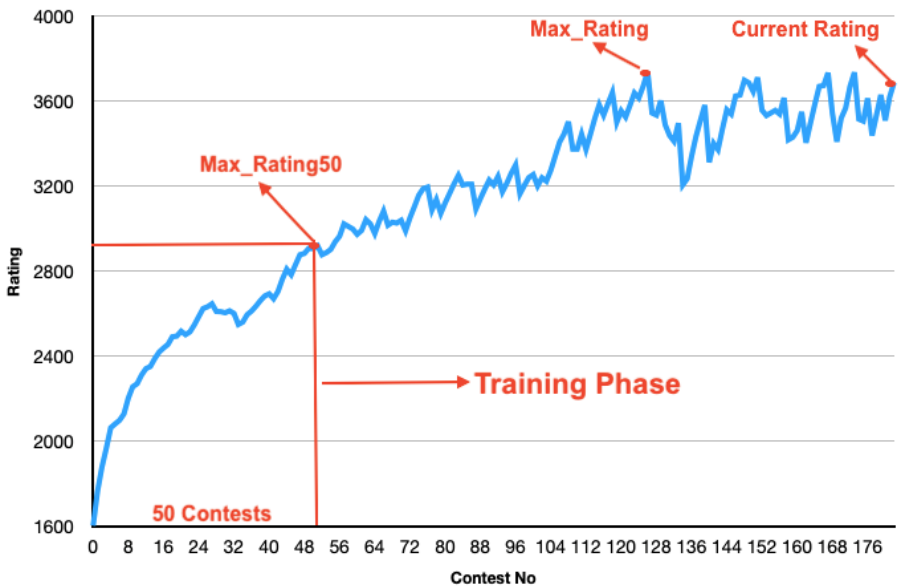

Figure (3): The Feature Creation

4. Implementation

In this section, the Machine learning will be executed to find out the predicted rating of a programmer using the generic, derived and newly constructed features.

In this research, the ML models were implemented in three distinctive stages. The stages along with detailed description is following: -

4.1 Formulation Stage:

In this initial stage of research, a new feature was introduced named Programmer's Grade. The concept was based on using generic parameter to figure out a grade out of 10 (1: worst , 10 : best).A formula was formed for finding grade using required parameters and weighted value. The top programmer's were assigned 10 and the rest of programmer's grade were assigned accordingly.

The Following graph shows relationship of Programmer's grade and Global Rating:

Figure (4): Programmer's Grade vs Rating

An hypothetical formulation was made for calculate the Programmer's Grade. The formula is the following:

$$Grade = [\frac{n-p1}{n} * 2 + \frac{p2}{100} * 3 + \frac{p3}{4000} * 5 + (\frac{(p4+p5)}{8000}) + \frac{p6}{p7*48}] * 100]$$

Let's discuss the parameters (P(n)) used on this formula: -

P(n)	Name	Weighted Value	Remarks
P1	Global Rank	2	Highly Important Feature
P2	Rank Title	3	Legendary Grandmaster (100) to Newbie (00)
P3	Present Rating	5	Main focus in predicting ranks

P4	Max rating	1	Highest level of Performance
P5	Min rating	1	Lowest indication of performance
P6	Participation	1	Finding Participation Frequency
P7	Years Joined		

Table (2): Parameter used in formulation

4.2 Time-series Stage:

In this stage of research, we tried to implement the time series algorithms such ARIMA and Prophet. They are both forecasting tools for regression analysis that shows the relative strength of the dependent variable against multiple variables. Here we were trying to predict the future rating of a programmer using multiple variables e.g., Current rating, participation, maxRating, minRating and so on.

The following diagram shows output of this stage:

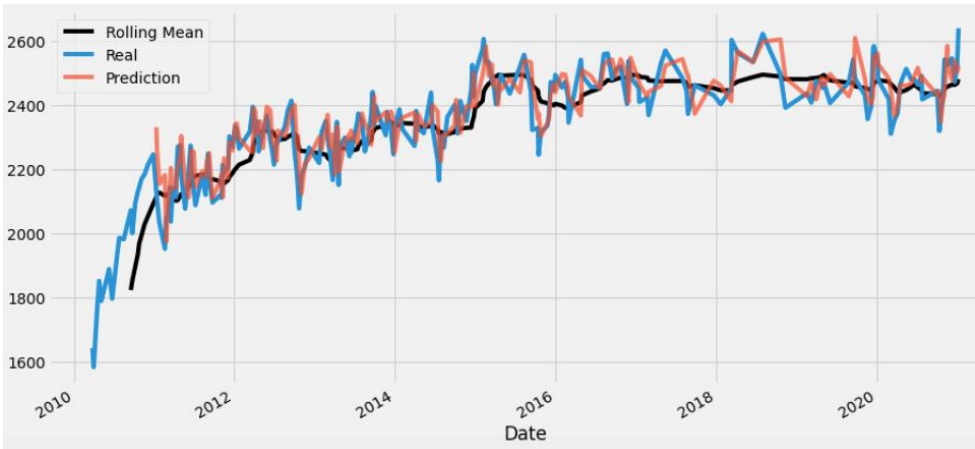

Figure (5): Implementation of ARIMA Algorithm

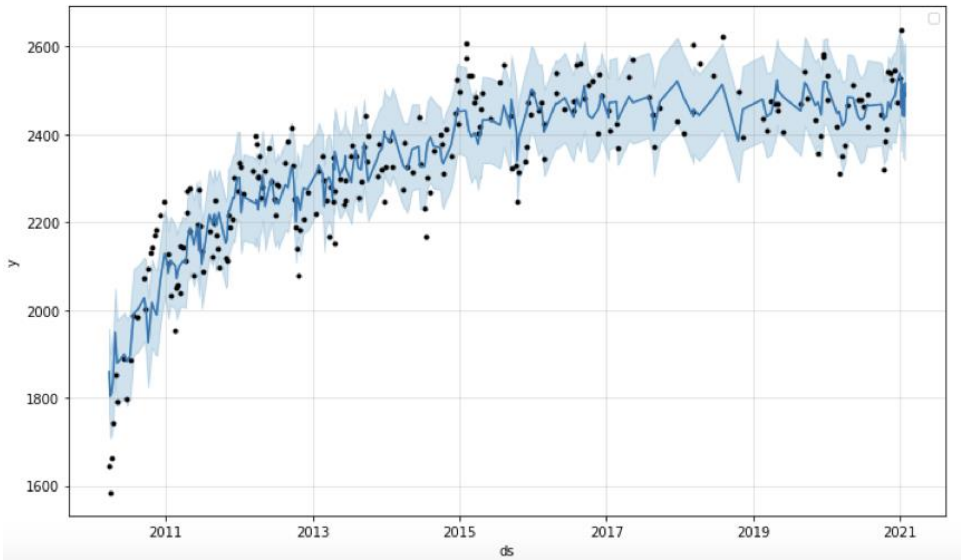

Figure (6): Implementation of Prophet Algorithm

4.3 Stock Prediction Stage:

While researching the current prediction trends, we came across the stock market value predication strategies. The techniques and tools used for stock prediction has some similarities with our research for rating prediction. A detailed methodological analysis was conducted as follows:

Paper	Methodology	Usability
[24]	Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE) and Mean Bias Error (MBE)- are calculated using Artificial Neural Network (ANN) and Random Forest	Couple of new parameters will be introduced: maxRating – minRating 1.Rating's moving average 2.Rating's standard deviation 3.Opening Rating (month/year) 4.Closing Rating (month/year) 5.Closing -Opening Rating 6.Frequency of Participation These parameters will be used in ANN & RF to predict rating
[25]	A hybrid model is used where two appending linear regression models' output of first block is fed to the input of second linear regression model. Indicator Used: MAE, MSE, RMSE	We can consider to hybrid two ML models in our research and compare the results with the single model
[26]	They have used following methods over DSE's data- 1. Auto Regressive Integrated Moving Average (ARIMA)	The following models can be implemented in our research since they share similar kind of parameters

	2. Feed- Forward Neural Network (FFNN). 3. Linear model Holt- 4. Winters model Holt- 5. Winters exponential smoothing model These prediction algorithms were applied over polynomial and actual trend data.	
[27]	The regression models that the dataset will be applied on are as follows: 1. Simple Linear Regression 2. Polynomial Regression 3. Support Vector Regression (SVR) 4. Decision Tree Regression 5. Random Forest Regression	The Regression and Classification models used in this paper can be implemented in my research since the parameters are similar in nature

Table (3): Stock Market Analysis

The knowledge and insights acquired from this analysis was later used in the final stage of our research.

4.4 Implementation of Slope Stage

In this final stage of the research, we initiated the several ML models on training and testing using the following parameters:

- Current Rating
- Max-rating
- Avg-rating
- Avg-Slope
- Max-Rating50
- Rated-Slope

The followings are the workflow in this stage:

1. **Importing Dataset:** The csv file of all programmers who have minimum participation of 5 contests was imported first. The dataset included almost 40K entries.
2. **Sorting Dataset based on 80 Participation:** In this stage, we sorted the dataset in order to keep only those participants who have taken part in at least 80 programming contests in Codeforce platform. This out training set will be of 50 contests. The new entry was around 3.5K.
3. **Populating The New Datatable:** Here we added the new features from the help of generic and derived features.

This is how the new Dataset looks:

	rating	MaxR	MinR	initialR	minS	maxS	avgS
0	3697	3783	1602	1602	-290	162	10.086735
1	3583	3675	1279	1404	-218	224	19.362069
2	3522	3597	1625	1625	-259	187	10.622093
3	3467	3668	1413	1448	-230	207	17.822034
4	3466	3648	1733	1733	-207	283	10.407609
...	...	...	...	...	...	...	...
8611	368	1371	327	1371	-115	280	-12.537500
8612	356	1443	87	1443	-124	514	-8.591837
8613	305	1382	214	1382	-106	245	-10.114943
8614	208	1366	-21	1366	-114	435	-14.846154
8615	170	908	-23	375	-97	313	3.661290

8616 rows × 7 columns

Figure (7): The Final Data table for ML Implementation

4. **Machine Learning Phase:** In this important stage, we have used several ML Models in order to find the predicted rating of the programmers.

   These following ML Models (Algorithms) were used:

   - Random Forest
   - Decision Tree
   - Linear Regression
   - Gradient Boosting Regressor
   - Linear Model SGD Regressor

   These are the following error metrics used for evaluation of the machine learning model:

   a) **Mean Absolute Error:** Mean Absolute Error(MAE) calculates the mean of absolute difference between predicted value and actual value.It's error evaluation metric used in almost all the regression models.
   b) **Mean Squared Error:** Mean squared error(MSE) is the mean of squared difference between predicted values and actual value. It can also be evaluated as the measure of the quality of a parameter .
   c) **Root-mean-square Error:** Root mean squared error (RMSE) is simply the square root of the mean of the square of all of the error. It's also known as the standard deviation of the predicted values from the actual value.

5. Result and Analysis

The following section will discuss the results acquired by the four different stages of implementation along with the findings and analysis.

5.1 Formulation Stage

The following shows the ML model used in this stage along with the Error Measures:

Predication Models	**Error Metrics**		
	MAE	MSE	RMSE
Random Forest	0.18	0.04	0.20
Decision Tree	0.19	0.05	0.22
SVM	0.29	0.13	0.36
Linear Regression	0.00218	0.00018	0.00003

Table (4): Error Metrics for Formulation Stage

Findings:

- All the prediction models show outstanding results as well as extremely low error
- Linear Regression shows the best results with error almost zero.

Limitations:

- The accuracy derived from the Machine Learning is almost 100 percent.
- Data over-fitting.
- No significant use of Predication Models

5.2 Time-series Stage

In this stage, the ARIMA could only be applied for single users only. A detailed analysis of a programmer called "ecnerwala" is given below:

This is called SARIMAX result analysis which stands for Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors.

SARIMAX Results

Dep. Variable:	Rating	No. Observations:	110
Model:	SARIMAX(1, 1, 1)x(0, 1, 1, 12)	Log Likelihood	−484.349
Date:	Tue, 05 Oct 2021	AIC	976.697
Time:	00:07:51	BIC	986.372
Sample:	0	HQIC	980.584
	− 110		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.8856	0.043	20.546	0.000	0.801	0.970
ma.L1	−1.0000	28.392	−0.035	0.972	−56.647	54.647
ma.S.L12	−1.0008	28.346	−0.035	0.972	−56.559	54.557
sigma2	5277.4208	0.005	9.8e+05	0.000	5277.410	5277.431

Ljung-Box (L1) (Q):	0.19	Jarque-Bera (JB):	12.61
Prob(Q):	0.67	Prob(JB):	0.00
Heteroskedasticity (H):	1.43	Skew:	−0.58
Prob(H) (two-sided):	0.35	Kurtosis:	4.52

Figure (8): Results of ARIMA

The basic information are as follows:

- Dep. Variable - What we're trying to predict i.e., Rating

- Model - The type of model we're using. AR, MA, ARIMA.
- Date - The date we ran the model
- Time - The time the model finished
- Sample - The range of the data
- Number of Observations - The number of observations used in this case

Analysis:

- The dependent variable is the close, which is what we're trying to predict.
- The independent variables the constant Beta, the error term is Sigma2 in model equation.
- The ar. L1, ar. L2, and ar. L3 are the lag variables.
- The P value for each of the lag variables must be less than 0.05 in order to consider them as significant.
- In this case, ar. L1 and sigma2 has p value less than 0.05. So, we can consider their standard error for our experiment.

Limitations:

- ARIMA is only used for single user predication whereas our research focused on a general model for all users together.
- In case of stock prediction, ARIMA is effective since it only considers the stock price in the basis to single time series but the core component in our research was to put all possible parameters into consideration while rating prediction and more.

5.3 Implementation of Slope Stage

The following table shows the Error Metrics:

ML Algorithms	**Error Metrics**		
	MAE	MSE	RMSE
RF	154.16	40621.98	201.54
DT	213.20	75857.92	275.42
LR	143.75	35319.95	187.93
GBR	145.05	36476.58	190.99
SDG	144.22	35505.52	188.42
LMBR	143.75	35317.54	187.93

Table (5): Error Metrics in Slope Stage

Findings:
As we can see from the table, The Linear Regression (LR) and Linear Model Bayesian Ridge (LMBR) has the least Mean Absolute Error(MAE) and Root-mean-square Error (RMSE).

Limitations:
Due to the unpredictability of the programmer's rating, it's tough to choose the point of slopes correctly. Since a slope of rating indicates the rise and fall of a programmer, it's difficult to predict the right points which will show the true nature of the programmer's expertise.

5.4 Correlation between the Programmer's Rating and Speedup:

The following table derived from [23], shows the Maximum Execution Time along with Reported Execution Time of a specific programming problem by different programmers having individual rating.

The formula of Speedup is following:-

$$Speedup = [\frac{Max\ Execution\ Time}{Reported\ Execution\ Time} * 100]$$

Rating Score	Maximum Execution Time	Reported Execution Time	Speedup (%)
2000	0.02	0.02	100
1944		0.03	67
1923		0.04	50
1875		0.05	40
1866		0.055	36
1777		0.06	30
1634		0.12	15
1543		0.19	11

Table (6): CodeChef Data of a Specific Programming problem

The following graph constructed by plotting The Rating against Speedup from the table 6.

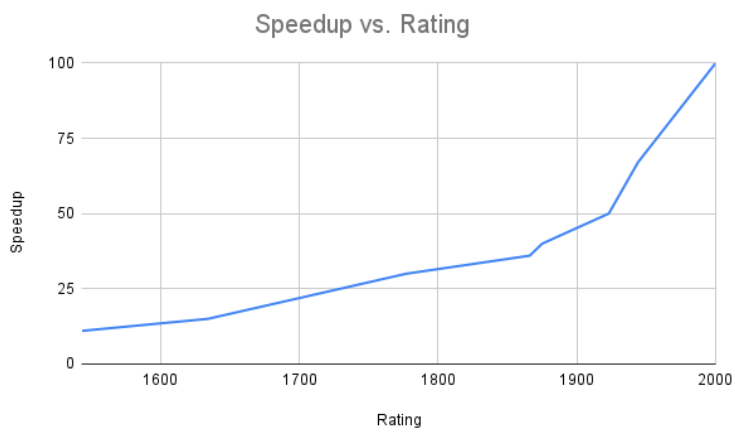

Figure (9): Correlation Between Speedup and Programmer's Rating

As we can see that, there is a "Positive" correlation between programmer's rating and Speedup.

6. Conclusion

The Speedup in HPC application is one of the main research topics in the field of parallel computing in supercomputers. The significance of programmer's expertise is huge on enhancing the speedups for

a certain HPC application. This research is based on establishing the correlation between rating and speedup. The maximum attainable speedup can be achieved by the top-rated programmers whereas the normal coders can reach up-to the average level of reported speedup. Initially the research focused on understanding the programmer's rating predication based on four different methodologies. In the first stage, a parameter named "Programmer's grade" was formed which was calculated by generic and derived features from the CodeForce API dataset. Later a different approach of Time-Series technique using ARIMA, and Prophet was executed for better rating prediction. Meanwhile an extensive literature review discovers an inter-connected feature concept between the "Stock Market" prediction and "Programmer's Rating" Prediction. The last step demonstrated a inclusion of several generated parameters based on the rise and fall of rating i.e. slope. All these programmer's data were trained and tested for a number of Machine Learning Models e.g., Random Forest, Decision Tree, Linear Regression Gradient Boosting Regressor, SVM and Linear Model SGD Regressor. All these models were evaluated using their significant Error Metrics MAE, MSE and RMSE which shows that The Linear Regression (LR) and Linear Model Bayesian Ridge (LMBR) has better prediction models compared to other ML algorithms. Finally, a CodeChef dataset containing the Ratio of two different execution times measurements from a specific contest problem, provided the Speedup which has a "positive" correlation with the corresponding rating of the individual programmer. The future work may include the execution of Programming contest in online judge platforms using HPC application and parallel programming.

References

[1] G. M. Amdahl, "Computer Architecture and Amdahl's Law," in Computer, vol. 46, no. 12, pp. 38-46, Dec. 2013, doi: 10.1109/MC.2013.418.
[2] S. Tang, B. Lee and B. He, "Speedup for Multi-Level Parallel Computing," 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops \& PhD Forum, 2012, pp. 537-546, doi: 10.1109/IPDPSW.2012.72.
[3] G. M. Amdahl, "Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities, Reprinted from the AFIPS Conference Proceedings, Vol. 30 (Atlantic City, N.J., Apr. 18–20), AFIPS Press, Reston, Va., 1967, pp. 483–485, when Dr. Amdahl was at International Business Machines Corporation, Sunnyvale, California," in IEEE Solid-State Circuits Society Newsletter, vol. 12, no. 3, pp. 19-20, Summer 2007, doi: 10.1109/N-SSC.2007.4785615.
[4] Dheeraj Malhotra and O. P. Rishi. IMSS-P: "An intelligent approach to design \& development of personalized meta search \& page ranking system". J. King Saud Univ. - Comput. Inf. Sci., (xxxx), 2018.
[5] Jiafeng Guo, Yixing Fan, Liang Pang, Liu Yang, Qingyao Ai, Hamed Za- mani, Chen Wu, W. Bruce Croft, and Xueqi Cheng. "A Deep Look into neural ranking models for information retrieval". Inf. Process. Manag., (June):102067, 2018.
[6] Geir Dahl. "A matrix-based ranking method with application to tennis". Linear Algebra Appl., 437(1):26–36, 2019.
[7] Prashant Premkumar, Jimut Bahan Chakrabarty, and Shovan Chowdhury. Key Performance Indicators for Factor based Ranking in ODI Cricket. IIMB Manag. Rev., 2019.
[8] J. Berkhout, "Google's PageRank algorithm for ranking nodes in general networks," 2016 13th International Workshop on Discrete Event Systems (WODES), 2016, pp. 153-158, doi: 10.1109/WODES.2016.7497841.
[9] N. Veček, M. Črepinšek, M. Mernik and D. Hrnčič, "A comparison between different chess rating systems for ranking evolutionary algorithms," 2014 Federated Conference on Computer Science and Information Systems, 2014, pp. 511-518, doi: 10.15439/2014F33.
[10] S.D. Suzuki, M. Ohue, Y. Akiyama, PKRank: a novel learning-to-rank method for ligand-based virtual screening using pairwise kernel and RankSVM, Artif. Life Robot. 23 (2) (2018) 205e212, https://doi.org/10.1007/s10015-017-0416- 8.
[11] W. Zhang, L. Ji, Y. Chen, K. Tang, H. Wang, R. Zhu, W. Jia, Z. Cao, Q. Liu, When drug discovery meets web search: learning to Rank for ligand-based virtual screening, J. Cheminf. 7 (1) (2015) 5, https://doi.org/10.1186/s13321-015- 0052-z.
[12] S. Jain, A. Grover, P. S. Thakur and S. K. Choudhary, "Trends, problems and solutions of recommender system," International Conference on Computing, Communication \& Automation, 2015, pp. 955-958, doi: 10.1109/CCAA.2015.7148534.
[13] Xiaoyan Li, Wei Li and Xiang Qi, "Research and develope of badminton sports video retrieval system," Proceedings of 2011 International Conference on Computer Science and Network Technology, 2011, pp. 1855-1858, doi: 10.1109/ICCSNT.2011.6182331.
[14] T. S. N. Peiris and R. M. Silva, "Player Ranking in Taekwondo: A Bayesian Elo Rating System," 2020 From Innovation to Impact (FITI), 2020, pp. 1-5, doi: 10.1109/FITI52050.2020.9424891.
[15] F. Ali and S. Khusro, "Player Ranking: A Solution to the Duckworth/ Lewis Method Problems," 2018 14th International Conference on Emerging Technologies (ICET), 2018, pp. 1-4, doi: 10.1109/ICET.2018.8603602.
[16] A. A. Ariyo, A. O. Adewumi and C. K. Ayo, "Stock Price Prediction Using the ARIMA Model," 2014 UKSim-AMSS 16th International Conference on Computer Modelling and

Simulation, 2014, pp. 106-112, doi: 10.1109/UKSim.2014.67.
[17] S. Ravikumar and P. Saraf, "Prediction of Stock Prices using Machine Learning (Regression, Classification) Algorithms," 2020 International Conference for Emerging Technology (INCET), 2020, pp. 1-5, doi: 10.1109/INCET49848.2020.9154061.
[18] A. Garlapati, D. R. Krishna, K. Garlapati, N. m. Srikara Yaswanth, U. Rahul and G. Narayanan, "Stock Price Prediction Using Facebook Prophet and Arima Models," 2021 6th International Conference for Convergence in Technology (I2CT), 2021, pp. 1-7, doi: 10.1109/I2CT51068.2021.9418057.
[19] X. Kan, M. Miao, L. Cao, T. Xu, Y. Li and J. Jiang, "Stock Price Prediction Based on Artificial Neural Network," 2020 2nd International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), 2020, pp. 182-185, doi: 10.1109/MLBDBI51377.2020.00040.
[20] G. Bathla, "Stock Price prediction using LSTM and SVR," 2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC), 2020, pp. 211-214, doi: 10.1109/PDGC50313.2020.9315800.
[21] Mehar Vijh, Deeksha Chandola, Vinay Anand Tikkiwal, Arun Kumar, “Stock Closing Price Prediction using Machine Learning Techniques”, Procedia Computer Science, Volume 167, 2020, Pages 599-606, ISSN 1877-0509,
[22] Codeforce API from: https://codeforces.com/api/user.rating?handle= .
[23] CodeChef website: https://www.codechef.com/
[24] Mehar Vijh, Deeksha Chandola, Vinay Anand Tikkiwal, Arun Kumar, Stock Closing Price Prediction using Machine Learning Techniques, Procedia Computer Science, Volume 167, 2020, Pages 599-606, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2020.03.326.
[25] S. Vazirani, A. Sharma and P. Sharma, "Analysis of various machine learning algorithm and hybrid model for stock market prediction using python," 2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE), 2020, pp. 203-207, doi: 10.1109/ICSTCEE49637.2020.9276859.
[26] M. M. R. Majumder, M. I. Hossain and M. K. Hasan, "Indices prediction of Bangladeshi stock by using time series forecasting and performance analysis," 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), 2019, pp. 1-5, doi: 10.1109/ECACE.2019.8679480.
[27] S. Ravikumar and P. Saraf, "Prediction of Stock Prices using Machine Learning (Regression, Classification) Algorithms," 2020 International Conference for Emerging Technology (INCET), 2020, pp. 1-5, doi: 10.1109/INCET49848.2020.9154061.
[28] Nazmul Ahasan Maruf and Waseem Ahmed, “Evaluate Metadata of Sparse Matrix for SpMV on Shared Memory Architecture” International Journal of Advanced Computer Science and Applications (IJACSA), 10(11), 2019. http://dx.doi.org/10.14569/IJACSA.2019.0101182.
[29] Istiak Ahmad, Fahad AlQurashi, Ehab Abozinadah and Rashid Mehmood, “A Novel Deep Learning-based Online Proctoring System using Face Recognition, Eye Blinking, and Object Detection Techniques” International Journal of Advanced Computer Science and Applications (IJACSA), 12(10), 2021. http://dx.doi.org/10.14569/IJACSA.2021.0121094.
[30] Ahmad, I.; Alqurashi, F.; Abozinadah, E.; Mehmood, R. Deep Journalism and DeepJournal V1.0: A Data-Driven Deep Learning Approach to Discover Parameters for Transportation. Sustainability 2022, 14, 5711. https://doi.org/10.3390/su14095711.