Intelligent Jobs Scheduling in palladium Cloud using Machine learning

- By - Harshit Monish <harshitm@cadence.com>; Prashant V Agarwal <agarwalp@cadence.com>; Animesh Kumar Sinha <animeshk@cadence.com>; Raghav Mahajan <raghavm@cadence.com>; Shivangi Singhal < ssinghal@cadence.com>

Introduction

Cadence palladium cloud-based verification is already a reality today. The current problems will pinch us more. Allocation of jobs on palladium cloud is a tedious task and an essential and most important part in a cloud computing environment. Due to the unreasonable resource scheduling in cloud computing, some of the cloud servers paralyze easily, while other servers are in the idle state. So, the resource scheduling is the key to the cloud computing research. The jobs scheduling mainly focuses to enhance the efficient utilization of resources and hence reduction in jobs completion time.

Jobs scheduling improves the efficient utilization of resource and yields less response time so that the execution of submitted tasks takes place within a possible minimum time. Since the scheduler is unaware of the compile time of a design, hence scheduler uses a novel approach of scheduling the jobs, which conservatively can lead to longer waiting time for small compile time design jobs. The application of ML techniques in compiler framework has become a challenging research area.

The key advantage of learning techniques is their ability to find relevant information in a high-dimensional space, thus helping us understand and control a complex system. To best of our knowledge, no approach targets jobs scheduling using compile time in cloud, using ML techniques. Several new approaches target communication, power-aware task scheduling, priority-based heuristics for concurrent architectures (Though without ML techniques).

Proposed solution

In this paper we put forward a job scheduling algorithm with the goal of the minimum completion time, maximum load balancing degree using an improved deferential algorithm in cloud computing. Our proposed solution is to use machine learning (ML) techniques, especially supervised learning techniques to learn and predict the compile time of a design at the parser stage of the palladium frontend compiler.

The learning algorithm is a function that gives us a relation between the various design attributes of a design and the time it will take for palladium to compile it. It is a supervised learning technique because the correct information is known using the existing designs.

Process

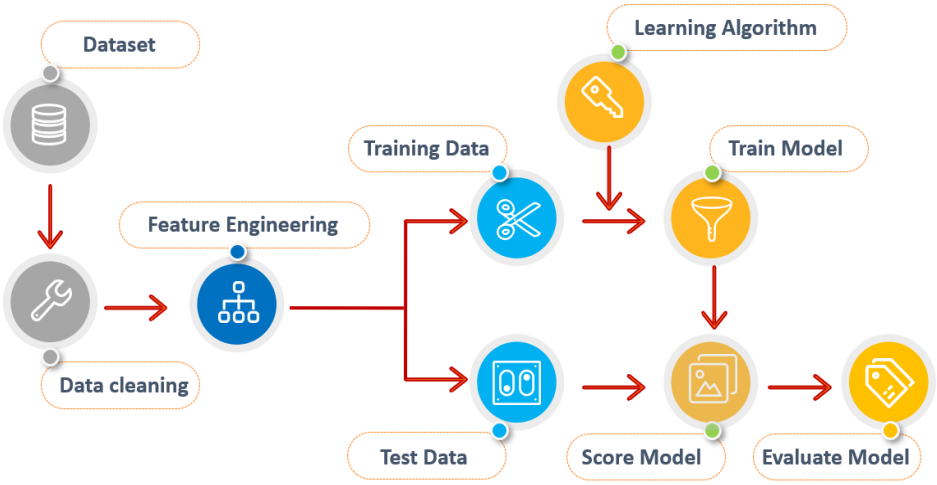

Fig. 1

Model Training:

To train our model, we collect design features via static analysis through profiling from a comprehensive suite of designs. This dataset is further refined and removed any redundancy in the data if found. The data is classified into two sets, i.e. training set and test set. The training set is fed into a supervised learning algorithm, which is intended to find a statistical model (or more precisely, a hypothesis function) that represents the relationship of static design features to compile time. Test data is used to minimize error (root mean square error) and increase accuracy and efficiency of the model. The learning algorithm strives for single goal.

Key points:

- Data collection is done by static analysis of design attributes i.e. number of I/O, processes, Assertions, Genblk, Instances, etc. and the compile time of existing designs. Followed by data cleaning.
- The learning algorithm is a regression algorithm trying to predict the compile time of the design, taking the design attributes as the input in the model.
- Data was nonlinear hence we explored various algorithms suitable for predicting the nonlinear dataset and implemented them.
- Using the model prediction on the compile time, at the early parser stage, we can schedule multiple design jobs on palladium cloud. Hence for a job that requires less compile time can be given priority over the jobs taking more compile time.

Use model:

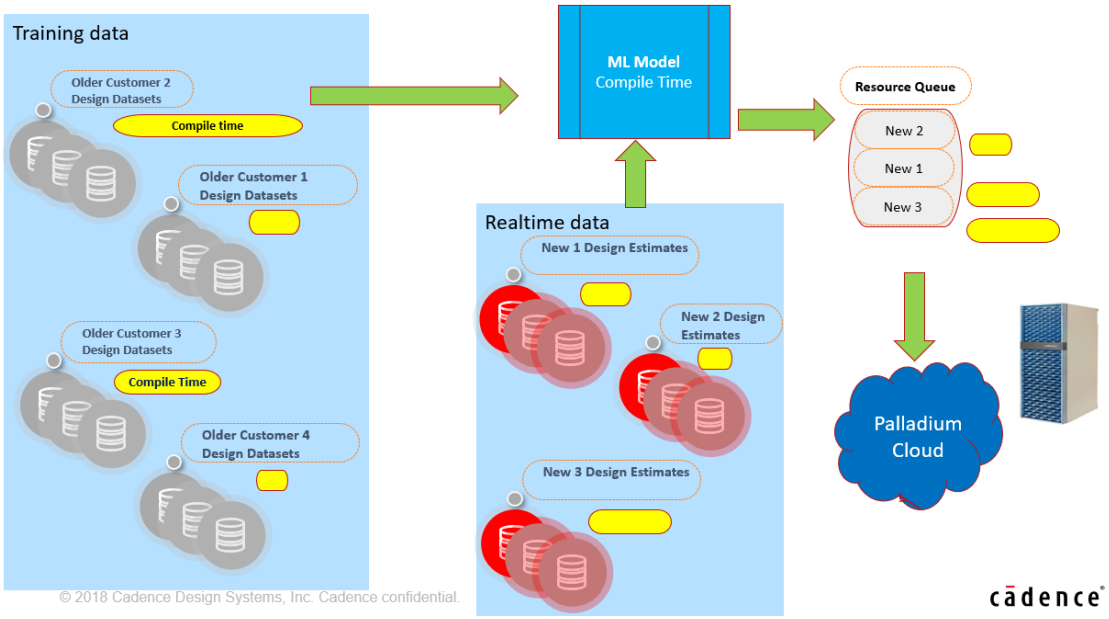

Fig. 2

The above figure depicts the whole use model. First, train our model then we predict the compile time of a design job on palladium cloud using our trained Machine learning model on the real time data. Second, we use the predicted compile time information to schedule design jobs on palladium cloud efficiently without performance degradation.

The jobs can be scheduled based on the compile time information, hence for a small design the compile time will be less therefore will be given priority in the scheduling.

Experimental results:

Details to be published as part of paper.

Conclusion/Summary

The paper demonstrates a novel and scalable approach to predict the compile time and schedule the jobs on palladium based on the compile time of these design jobs. Our proposed learning algorithm relates the static design attributes to the design compile time.

Who benefits from this:

- This solution helps customers as it reduces the turnaround time of the design jobs of customers on cadence palladium. Also, will improve the quality of the cadence palladium and will act as a differentiation factor for cadence palladium business with good results.

Future Work:

The model can be further extended to predict other verification aspects as well, i.e. the design size, static power analysis. Can also be scaled to Protium, Vulcan.