JCS&T Vol. 10 No. 3 October 2010

Thesis Overview

Parallel Application Signature for Performance Prediction

Alvaro Wong*
Universitat Autònoma de Barcelona
Computer Architecture and Operating System Department (CAOS) Barcelona, SPAIN
PhD Thesis in High Performance Computing
Advisor: Prof. Emilio Luque†
*alvaro@caos.uab.es,†emilio.luque@uab.es

In order to measure the performance of a parallel machine, a set of application kernels as benchmarks have often been used. However, it is not always possible to characterize the performance using only benchmarks, given the fact that each one usually reflects a narrow set of kernel applications at best. Computers show different performance indices for different applications as they run them. Accurate prediction of parallel applications' performance is becoming increasingly complex and the time required to run it thoroughly is an onerous requirement; especially if we want to predict for different systems.

In production clusters, where throughput and efficiency of use are fundamental, it is important to be able to predict which system is more appropriate for an application, or how long a scheduled application will take to run, in order to have the foresight that will allow us to make better use of the resources available.

We have created a tool [4], which we dubbed Parallel Application Signature for Performance Prediction (PAS2P) to characterize message-passing parallel applications. PAS2P instruments and executes applications in a parallel machine, and produces a trace log. The data collected is used to characterize computation and communication behaviour. To obtain the machine-independent application model, the trace is assigned a logical global clock according to causality relations between communication events, through an algorithm was inspired by Lamport. Once we have the logical trace, we identify and extract the most relevant event sequences (phases) and assign them a weight from the number of times they occur. Afterwards, we create a signature defined by a set of phases selected depending on their weight. This is the signature through whose execution in different target systems allows us to measure the execution time of each phase, and hence to estimate the entire application's run time in each of those systems. We do this by extrapolation of each phase's execution time using the weights we have obtained.

As shown in Figure 1, there is a sequence of stages that are necessary to obtain the relevant portions (phases) and their weights. With this information, we can proceed to create a completely machine-independent signature for each application that we can then execute in other systems in a shorter amount of time, since the execution time of the signature will always be a small fraction of the whole application's runtime. Finally, in the last stage, we predict the full execution time of the parallel application by adding the execution time of all the phases multiplied by their weights.

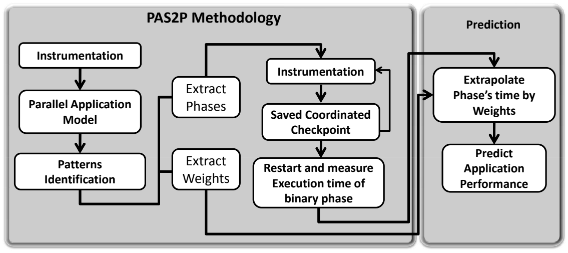
PAS2P Methodology
Prediction
Instrumentation
Parallel Application Model
Patterns Identification
Extract Phases
Extract Weights
Instrumentation
Saved Coordinated Checkpoint
Restart and measure Execution time of binary phase
Extrapolate Phase's time by Weights
Predict Application Performance

Fig. 1. PAS2P methodology

To instrument the parallel applications, we need to collect communication events and the computational time. Afterwards we define:

Event: The action of sending or receiving a message.

Extended Basic Block (EBB): A generalization of the Basic Block concept for parallel systems. We define it as a segment of a process whose beginning and end are defined by occurrences of messages, either sent or received. We may also say that it is a "computational time" segment bounded by communication actions.

The synchronization between computing nodes, which is absent in sequential applications, becomes necessary. To solve this, we have to move from multiple physical, local clocks to a single logical, global clock. In [1], we showed a logical clock based on the order of precedence of events accross processes as defined by Lamport. We found that the quality of prediction falls, because there is a non-deterministic ordering of receptions.

To solve the non-deterministic events (receptions) problem, we have decided to introduce a new algorithm [3] inspired by Lamport's. Through this algorithm, we define a new logical ordering, in which, if one process Sends a message in a Logical Time (LT), its reception will be modeled to arrive in a LT + 1 and never afterwards.

Once all events have been assigned an LT, we create a logical trace where we insert all events depending on its logical time and type of communication (Send or Recv). Finally, once we have located each event, we divide the logical trace into more logical times, that is, there can only be one event for each process in a logical time.

Now, we introduce two new concepts:

Tick: Logical time unit.

Parallel Basic Block (PBB): The set of Extended Basic Blocks delimited by two ticks. The first tick defined as Entry Point has at least one event, and the second tick defined as Exit Point also has at least one event.

To find the repetitive behavior of an application we have created in the previous section a logical trace that allows us to create the Parallel Basic Block. Now, we need to compare if the behavior of each PBB is similar to another. To do this, we search for similarity between two Parallel Basic Blocks based on the three main components of its structure:

1) Communication type: Each of the assigned values of the entry points and each of the assigned values of the exit points should be the same.
2) Communication Volume: each of the values of the entry and exit points must be similar, and can accept a difference of up to 5%.
3) Computational Time: each computational time allows for a difference of up to 5%.

In order to identify the most relevant portions (phases) of the parallel application, we are proposing a technique that searches for similarity between Parallel Basic Blocks. Another method [2] is being proposed to extract these phases which consists of creating them directly from the logical trace. Unlike the previously proposed algorithm for the detection of phases, we have identified as similar phases that were not, and we have created a robust similarity algorithm thus generating less phases, with which the quality of prediction increases. The quality of the PAS2P methodology is measured by the significantly lower value of the prediction error. We compare the results obtained with the proposed algorithm (similarity between Parallel Basic Blocks vs. similarity between phases), it can be seen that we have increased from having a 96% to a 98.7% of quality prediction.

Once we have identified phases of the application, we proceed to create its weight vector and define the relevant phases.

Weight Vector: This vector will be given by the frequency in which each phase repeats.

Relevant Phase: A phase is relevant when the weight vector multiplied by the execution time is representative of the total runtime of the application. We have considered that this ”representativeness” will be given if the phase stands for 1% or more of the total execution time of the whole application.

To build the Parallel Application Signature, the last step is to re-run the application to create the coordinated checkpoints before each relevant phase happens. The checkpoint operation is taken before the starting point of the specific phase, in order to guarantee a correct warm-up time for the machine’s components (cache, TLB’s, etc).

PAS2P gives us the phases and where they occur, it also tells us how many times the phase has repeated (weight vector for each phase). To run the signature means to execute its constituent phases. This is done using the coordinated checkpoint obtained for the different phases restarting from the saved state and start measuring from the point a phase begins until it ends. We repeat this method and proceed to execute all constituent phases.

To evaluate the quality of the prediction and validate the proposed methodology, we have executed on two target machines with multicore (dual y quad) architecture as shown in Fig. 2 with different applications with 64 processes to predict the execution time with an average accuracy above 98%.

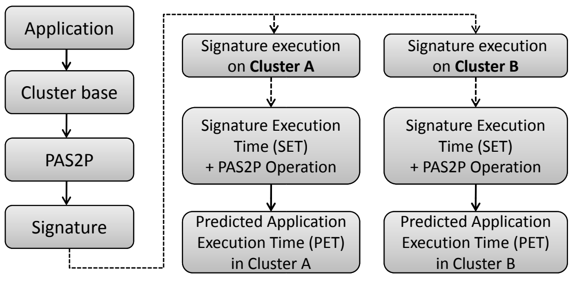
Application
Cluster base
PAS2P
Signature
Signature execution on Cluster A
Signature execution on Cluster B
Signature Execution Time (SET) + PAS2P Operation
Signature Execution Time (SET) + PAS2P Operation
Predicted Application Execution Time (PET) in Cluster A
Predicted Application Execution Time (PET) in Cluster B

Fig. 2. Experimental methodology

Program	SET (Sec)	SET vs AET(%)	PET (Sec)	PETE (%)	AET (Sec)
		Prediction on Cluster A			
CG	5.39	0.22%	2413.01	0.01%	2412.70
Sweep3D	1.94	0.15%	1257.03	0.27%	1260.32
POP	15.48	1.95%	758.31	4.33%	792.56
		Prediction on Cluster B			
CG	8.42	0.29%	2793.42	1.90%	2847.42
Sweep3D	3.01	0.22%	1328.04	0.40%	1322.62
POP	22.79	1.41%	1608.85	0.17%	1611.59

PETE: Prediction Execution Time Error

PAS2P methodology allows us to generate a model of a parallel application, and subsequently, extract its most significant behavior (phases) automatically in order to create a Parallel Application Signature, that by its execution, lets us predict the application’s performance on different parallel computers.

Acknowledgments.

This research has been supported by the MEC-MICINN Spain under contract TIN2007-64974, Appreciation to CONACYT-MEXICO for the support provided to Alvaro Wong.

References

[1] A. Wong, D. Rexachs, and E. Luque. Parallel application signature. In *Cluster Computing and Workshops, 2009. CLUSTER ’09. IEEE International Conference on*, pages 1 –4, 31 2009-sept. 4 2009.

[2] A. Wong, D. Rexachs, and E. Luque. Extraction of parallel application signatures for performance prediction. *High Performance Computing and Communications, 10th IEEE International Conference on*, 0:223–230, 2010.

[3] A. Wong, D. Rexachs, and E. Luque. Parallel application signature for performance prediction. In *In International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2010) CSREA Press.*, volume Vol 2, 2010.

[4] A. Wong, D. Rexachs, and E. Luque. Pas2p tool, parallel application signature for performance prediction. In *Para 2010: State of the Art in Scientific and Parallel Computing, Accepted*, 2010.