Compas'2019

HARDWARE ASSISTED VIRTUAL MACHINE PAGE TRACKING

Stella Bitchebe[1], Alain Tchana[1], Laurent Réveillère[2]

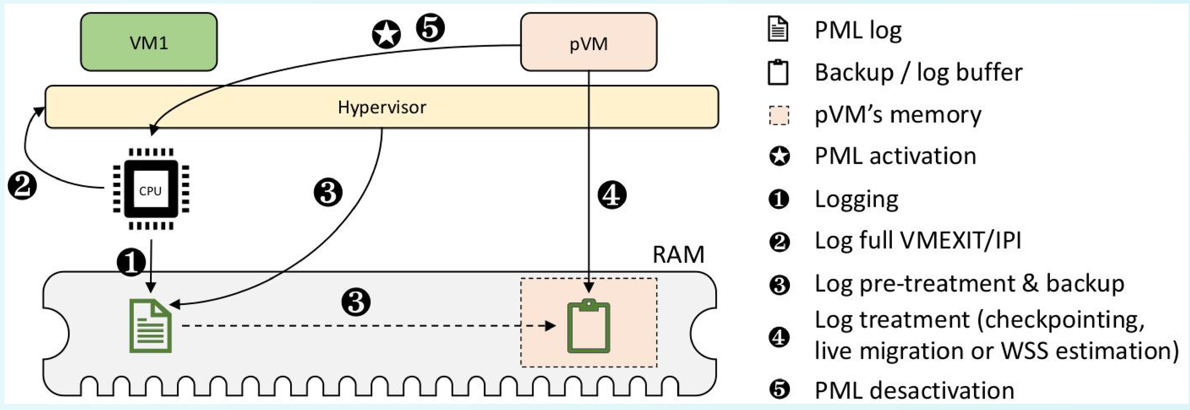
[1]Université Cote d'Azur – I3S, [2]Université de Bordeaux – LaBRI

Introduction

- Memory page tracking is a key demand in data centers because it is at the heart of several essential tasks such as checkpointing, live migration and ***working set size*** (**WSS**) estimation.
- The widely used approaches for achieving page tracking are write protection and present bit invalidation, which lead to performance degradation especially when a signigicant amount of memory need to be tracked (like in WSS). Alternatives approaches such as working on a random sample of pages (like VMWare solution) are proven to be inacurrate.
- Following the ***hardware assisted virtualization*** (**HAV**) trend, Intel started to release since 2015, processors with a novel virtualization feature for tracking virtual machine (VM) memory pages, called ***Page Modification Logging*** (**PML**).
- We studied PML from three angles: energy consumption, efficiency and performance impact on user applications. We observed that PML reduces down to 10.18% both VM live migration and check pointing, PML slightly reduces, down to 0.95%, the performance degradation incurred by those operations. We also observed that PML does not allow accurate WSS estimation.
- Using the Gem5 simulator, we implemented a smart extension of PML, called ***Page Reference Logging*** (**PRL**), and a WSS estimation system that relies on it. Evaluation results validate the accuracy of PRL and show that it incurs no energy overhead and no performance degradation for user appliations.

PML Overview

Description



Basic utilization of PML for improving a virtualization operation (live migration, checkpointing and WSS estimation).

- When the user VM (green) modifies a page during its execution, the CPU logs the guest physical address (GPA) of that page inside a buffer.
- When the log buffer is full (512 addresses), the CPU raises a VMExit which traps inside the hypervisor.
- The handler of that VMExit does a certain task (e.g. copy the content of the log buffer inside a larger one), then the PML index is reset to 511, the VM resumes and the process restart.

Limits

Soft limit - Unfairness for cloud users and their VMs

- Due to VMExits imputed to the VM when the *PML log buffer* is full.

Hard limits - Inaccurate estimation of WSS

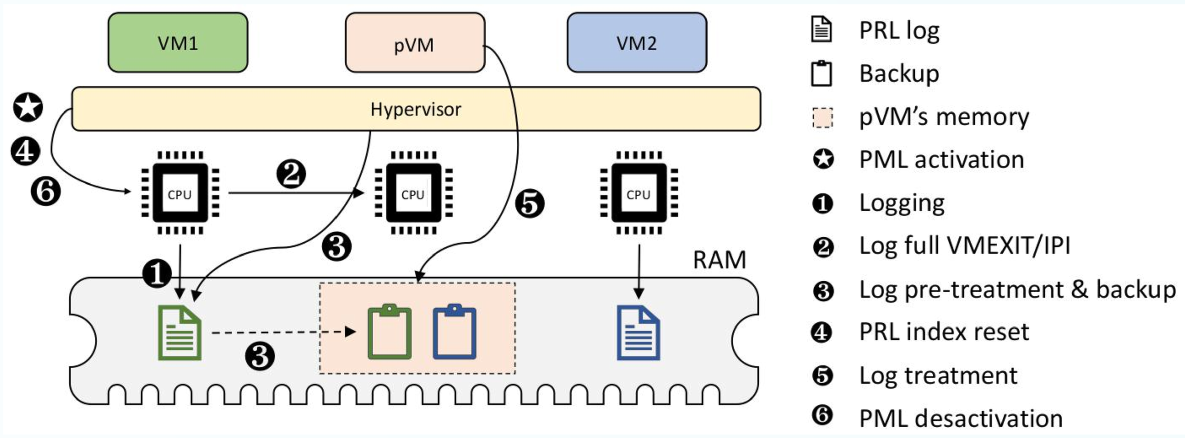
- Due to the fact that only accessed pages are not logged and hot pages cannot be tracked.

Contribution

Hardware

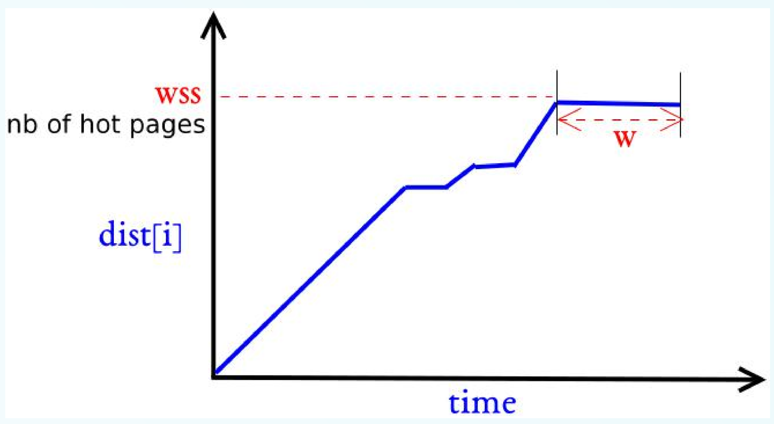
Conceptually, PRL includes two innovations:

- The capability to track both read and write pages
- The redirection of log full events to pVM's (privileged VM) CPUs, instead of user VMs



PRL, the design that we propose. It works in two exclusive modes: PRL PML and PRL PAML. The latter mode is only presented on this figure.

Software



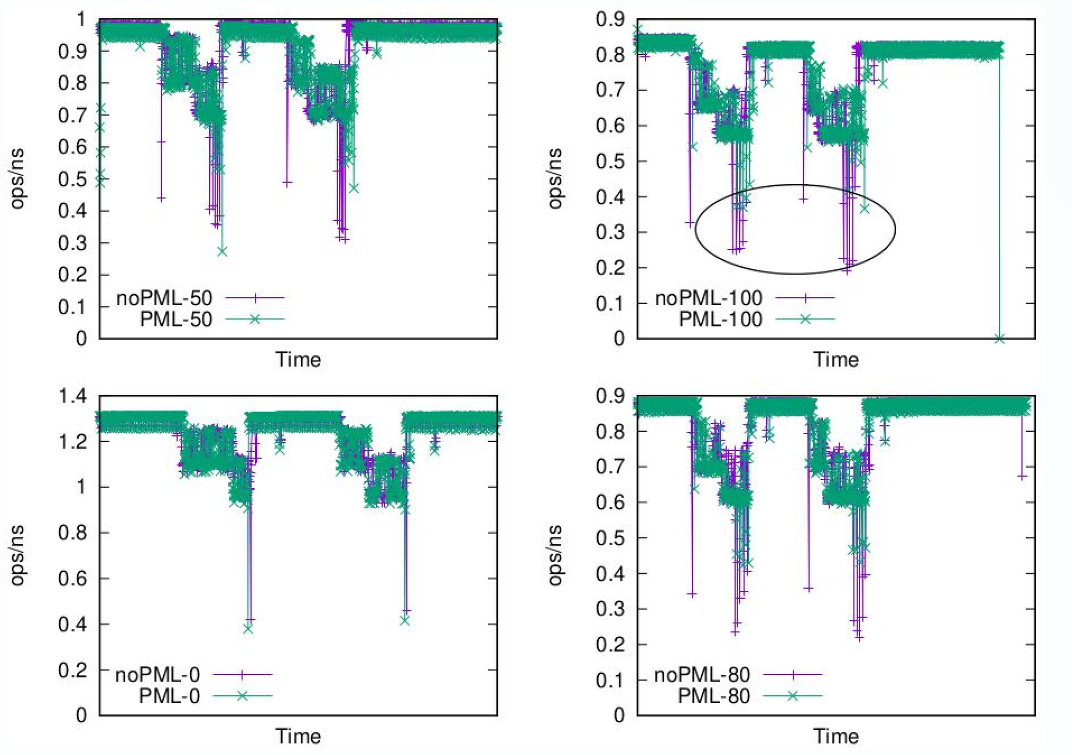
- w: stability duration
- t: treshold to consider a page as hot
- £: kernel footprint
- M: WSS estimation
- dist[i]: number of GPAs in @buff logged more than t times

M = wss x sizeof_a_page + £

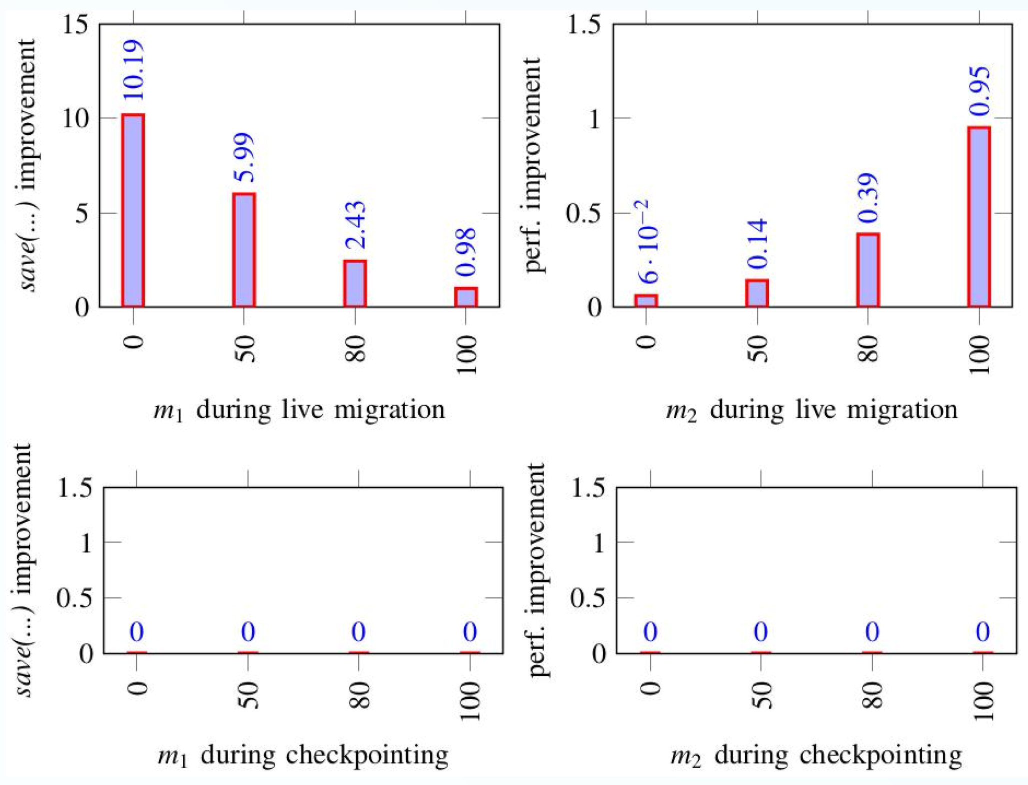
PRL-based WSS estimation algorithm

Experimental Results

PML benefit for live migration and checkpointing

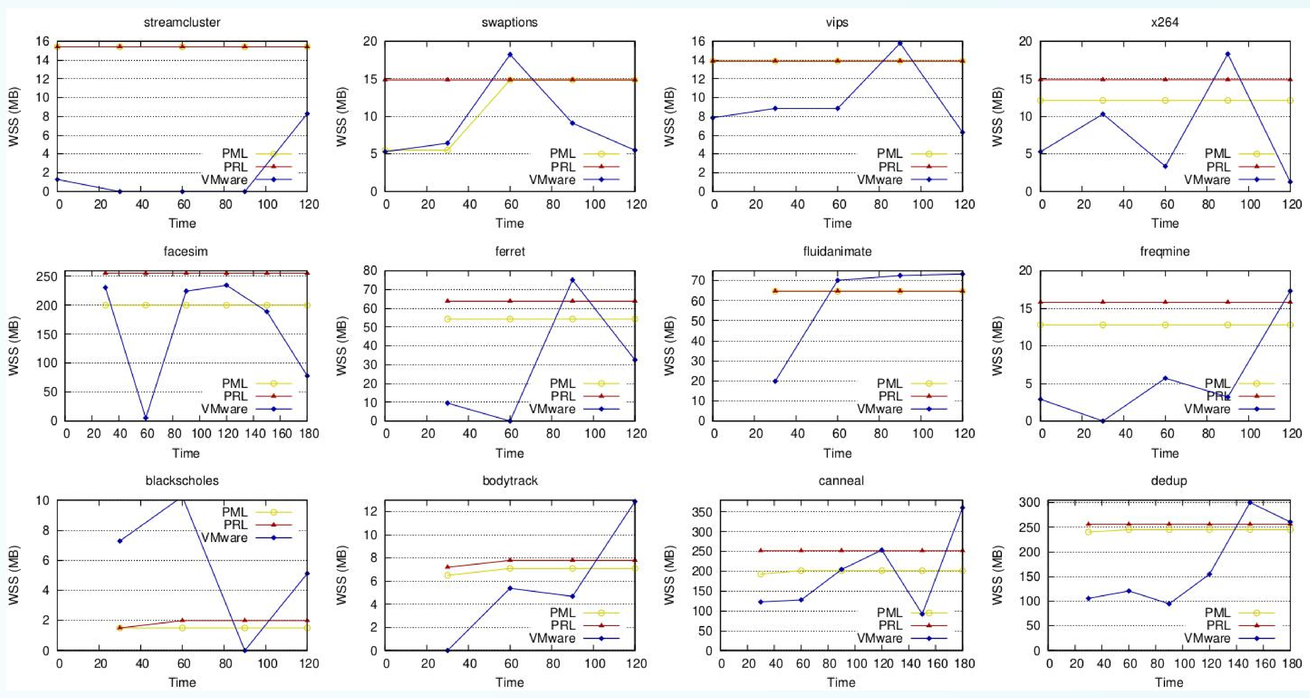


Number of operations per nanosecond during live migration for different write intensity values (0%, 50%, 80%, 100%)



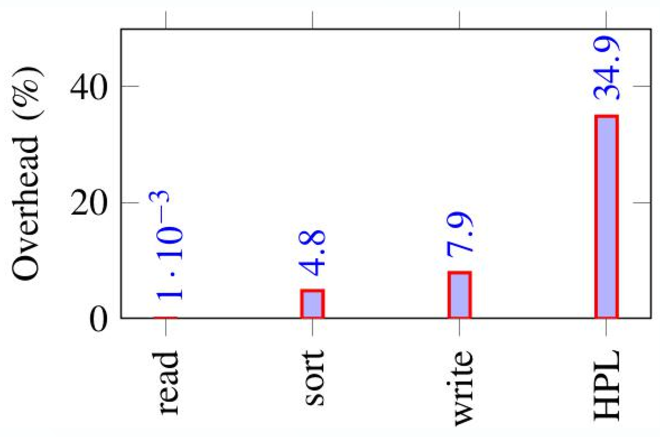
m_1 the execution time of save(...) method and m_2 the performance of the user application during checkpointing/migration. m_1 tells whether PML accelerates checkpointing/migration while m_2 allows to check whether PML reduces or increases the negative impact of these operations on the application.

WSS estimation

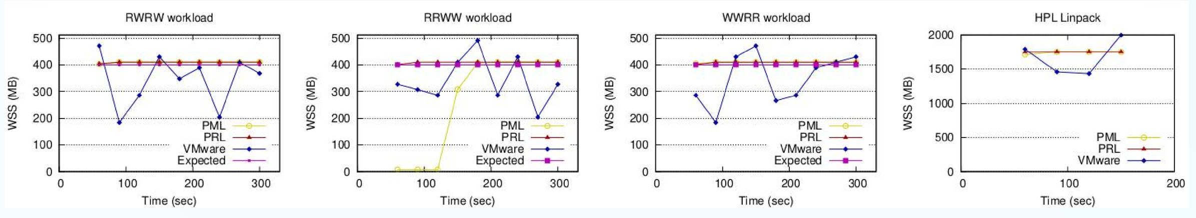


WSS estimation of PARSEC applications.

PML overhead



PML impact (for HPL Linpack the problem size is 29184 while for BigDataBench applications, the dataset is 10GB).



Accuracy of PRL compared with PML and VMware, on synthetic workloads. (R stands for read operation, and W for write operations).