

TMO: Transparent Memory Offloading

Johannes Weiner, Niket Agarwal, Dan Schatzberg, Leon Yang, Hao Wang, Blaise Sanouillet,
Bikash Sharma, Tejun Heo, Mayank Jain, Chunqiang Tang (Meta)
and Dimitrios Skarlatos (CMU)

Overview

1. Problem
2. PSI
3. Senpai
4. Evaluation
5. Advantages and Limitations

Key Problem

- Memory is a bottleneck in data centers today
- However, memory is not efficiently used by the applications in the datacenter
 - Applications are bloated
- It is hard to find the correct (least) memory allocation for application that do not degrade performance

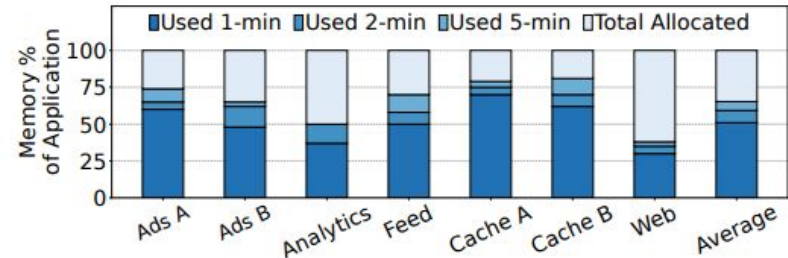


Figure 2: Application's recently used memory in 1 min, 2 min, and 5 min intervals, and memory not used in the last 5 min.

Cost of memory in datacenters

Memory cost has risen greatly while SSD cost is comparatively stable

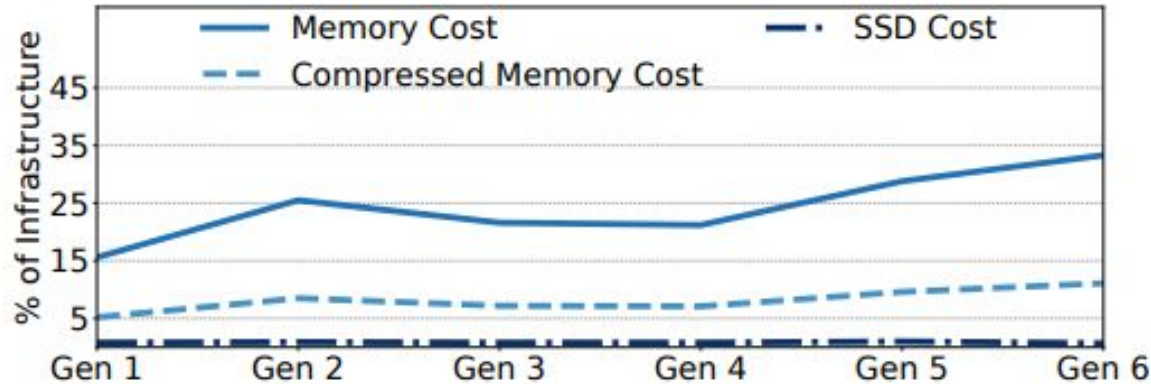


Figure 1: Cost of memory, compressed memory, and SSDs as a percentage of compute infrastructure across hardware generations.

Memory “tax”

The memory tax is the memory of additional processes and services needed to run the application in a datacenter (like logging, analytics) and to run them as microservices (like additional memory for docker, controllers etc)

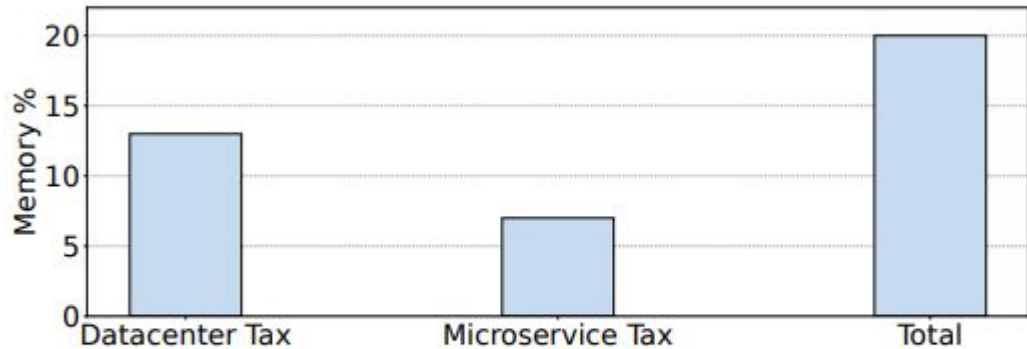
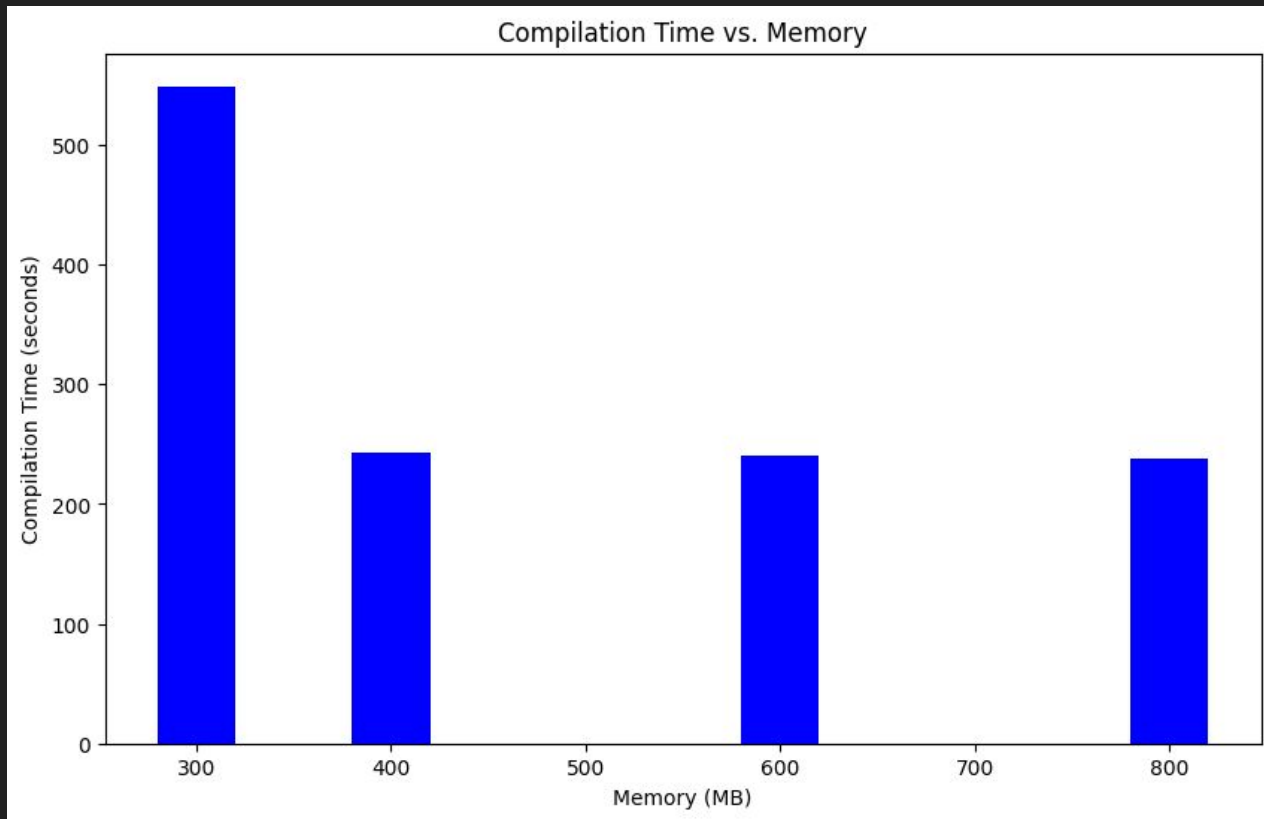


Figure 3: Datacenter and microservice memory tax.

Motivating Example

- This example is taken from the paper's github repo
- The time to compile a project is measured
 - `time make -j4 -s`
- Without any memory constraints, the process consumes around 800 MB of memory and takes 3m 58s to compile
- Reducing the process memory to 600MB and 400MB only increases the compilation time by 2s and 5s respectively
- However, reducing the memory to 300MB more than doubles the compilation time

Motivating Example



Motivating Example

Conclusion:

- The increase in latency $< 10\%$ when the memory is reduced to 400MB
- However the increase in latency $> 100\%$ when memory is reduced to 300MB
- Thus, the “correct” memory allocation is between 400MB and 300MB
 - Correct here is defined as the least memory allocable with a “reasonable” performance degradation
- Finding this correct allocation is not easy using current methods/tools dynamically

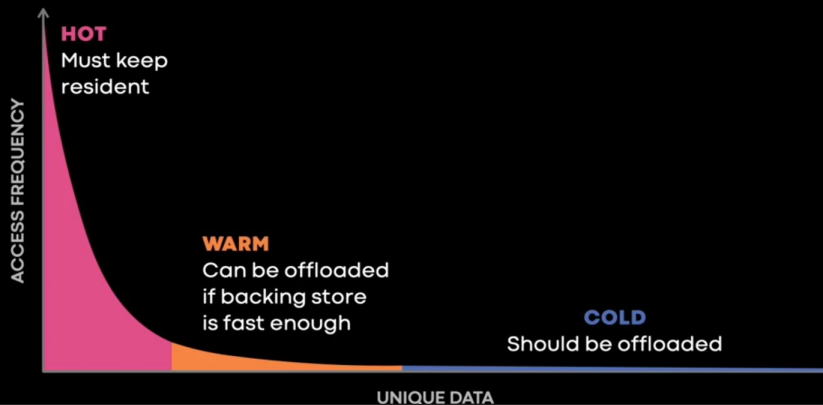
TMO

The paper aims to solve the problem of finding the correct allocation by:

- Introducing new metrics called **PSI** to measure slowdown due to resource contention
 - CPU, memory and storage contention
- Introduces **Senpai**, that uses the PSI metrics and informs the kernel how much memory should be reclaimed
- Evaluates their tool on the Meta datacenter

TMO

Memory Use Is Really a Probability Density Function



Two Pieces of Technology



PSI



SENPAI

PSI (Pressure Stall Information)

(Using the cgroups2 info instead of paper)

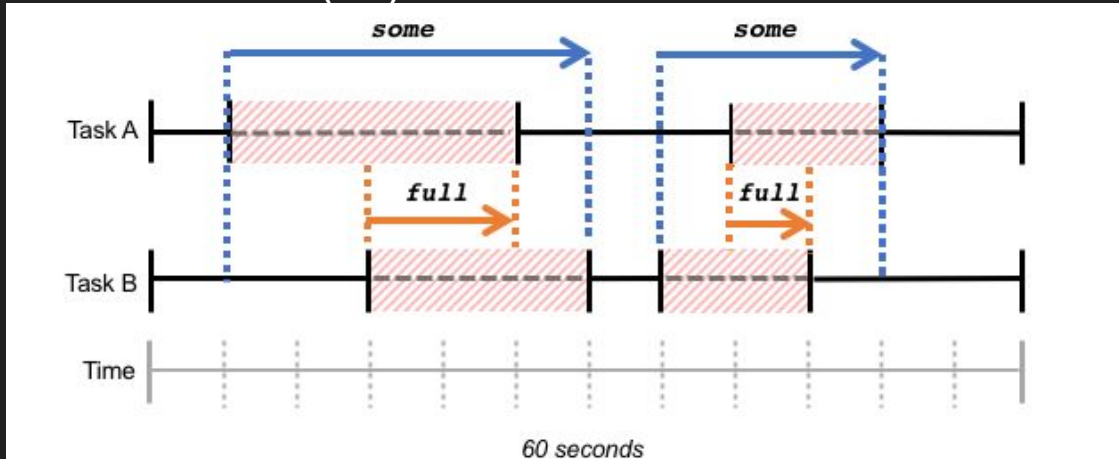
Pressure is defined as the time stalled waiting on a resource

Cgroups2 tracks some and full PSI for CPU, memory and disk.

In cgroups2, PSI is measured in microseconds (us)

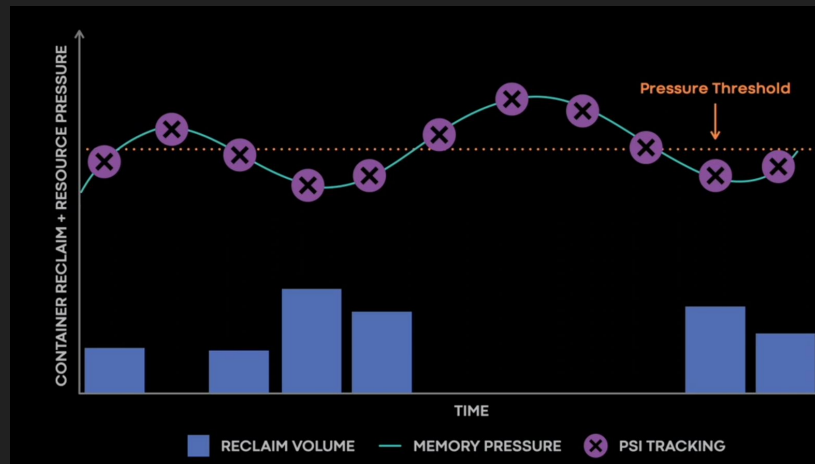
But the paper measures it in %:

% of time stalled per second



Senpai

- The core idea is to reclaim memory when the memory PSI is low
- For most cases*, reclaiming memory should put more pressure, slowing down the process and increasing PSI
- When a certain threshold is reached, reclaiming is stopped to prevent performance degradation



Senpai

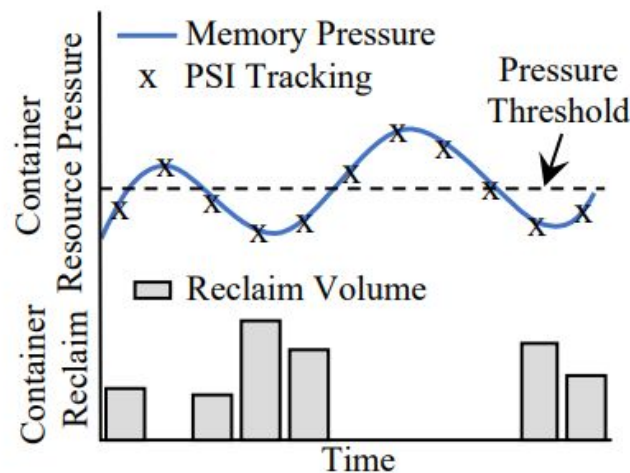


Figure 8: Example of Senpai PSI tracking and reclaim volume tuning.

Figure 8 shows a high-level overview of Senpai's operations. Once every few seconds, Senpai calculates for each cgroup the amount of memory to reclaim as follows:

$$\text{reclaim_mem} = \text{current_mem} \times \text{reclaim_ratio} \times \max(0, 1 - \frac{PSI_{\text{some}}}{PSI_{\text{threshold}}})$$

Senpai code differences

Based on the Senpai github repo, there are some differences between the paper and the implementation:

- Senpai uses both memory and disk PSI threshold, and stops reclaim when either of them reach their assigned threshold
- Senpai runs a cycle every second and measures the PSI in the last second to determine the reclaim

Reclaiming Memory

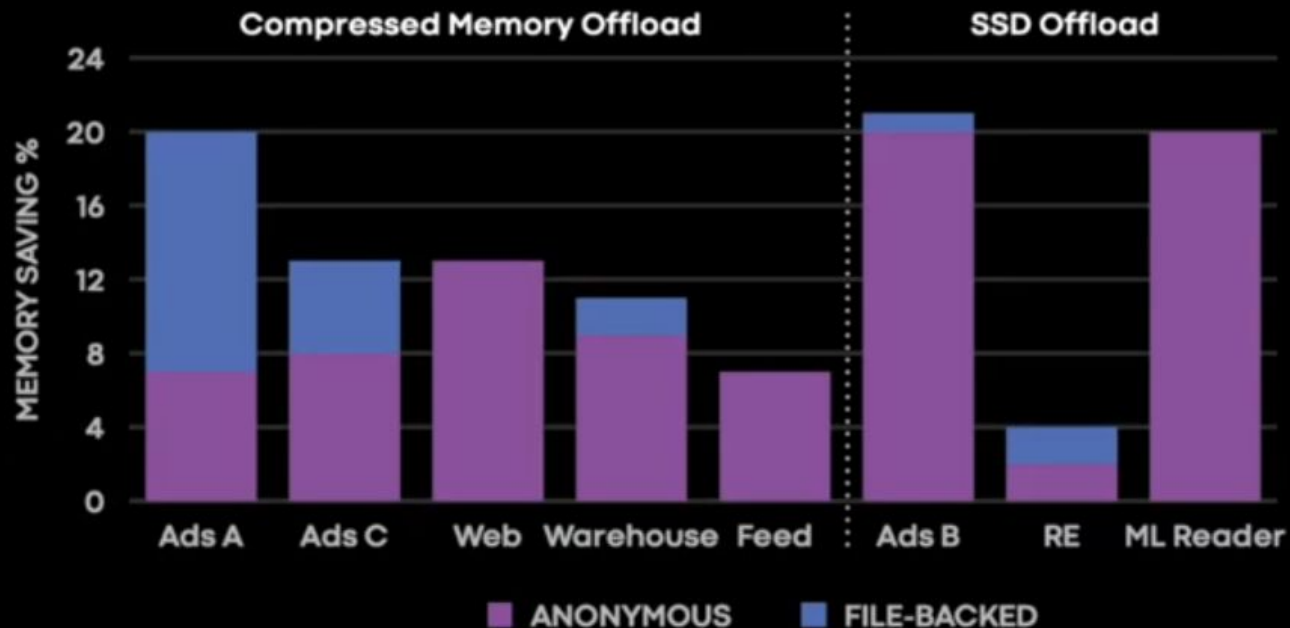
The primary source of reclamation is through evicting file caches.

When TMO detects there are a lot of refaults on the “working set” of file caches, It switches to either swapping or compressing memory.



Evaluation: Memory Savings

Overall Fleetwide Results



Evaluation

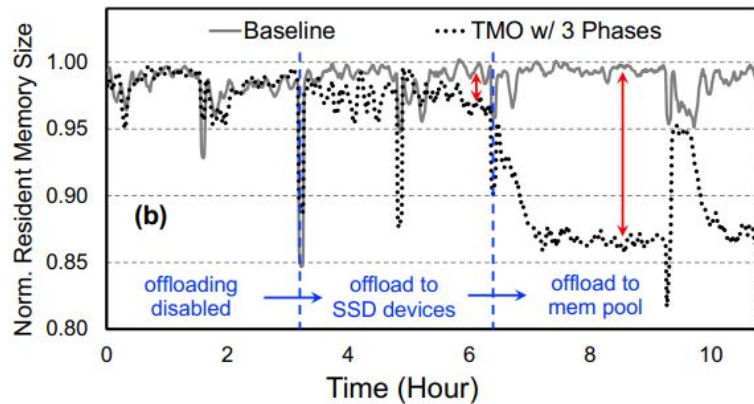
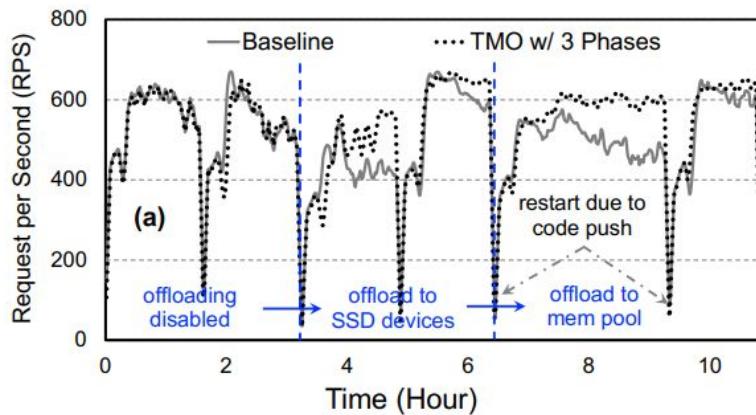


Figure 11: Performance improvement and memory saving for the Web application on memory-bound hosts. We set up two tiers: started with identical configuration without any swap enabled and later enabled SSD & then compressed memory offloading on the 2nd tier.

Evaluation: Memory Tax

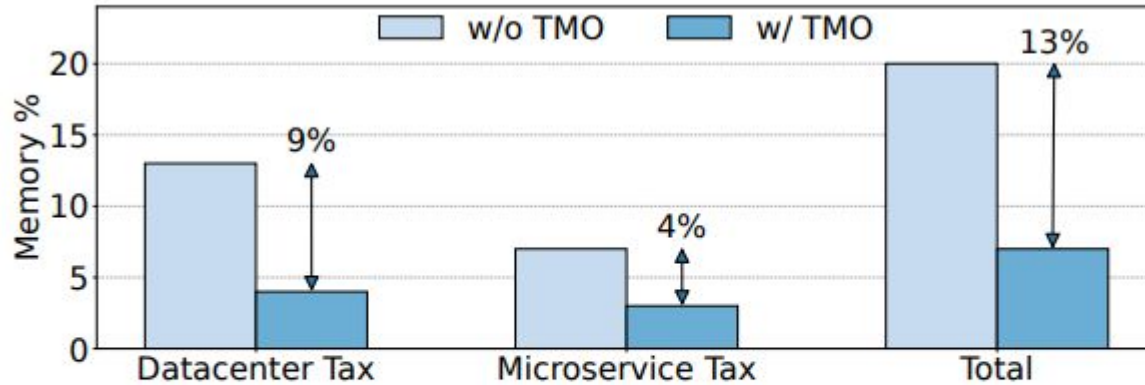


Figure 10: Datacenter and application memory tax savings normalized to a server's total memory.

Limiting Swap

SSD have limited write endurance, so limiting Swap is necessary for certain SSDs.

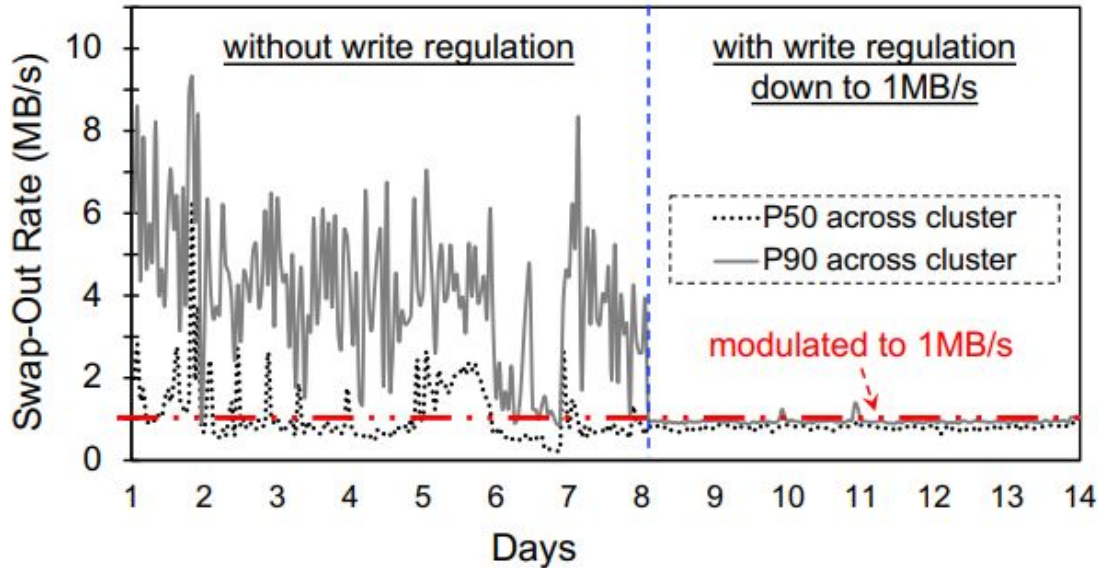


Figure 14: Swap rate with and without write regulation.

TMO Advantages

- PSI has been upstreamed to linux and available with cgroups2
- PSI measures pressure at the container level, so each container can have individual memory reclamation
- Senpai works directly without any kernel modification on a user level and is easily configurable
- Senpai switches between file cache and anonymous memory depending on the re-fault rate of file caches.

TMO in Practice

- PSI is not constant between machines:
 - Hardware differences
 - Different disks
 - Background processes
 - “Noisy neighbours”
 - This makes having a constant threshold hard since some machines inherently have a higher threshold than others, resulting in lower or no reclaim

TMO in Practice

- There is still effort needed to optimize the parameters of Senpai
 - The default parameters are too conservative.
 - PSI metrics differ based on machine and load.
 - Sensitivity of performance on PSI for different application varies vastly.
- Container right sizing and memory sharing
 - Reducing memory used does not equal to savings directly since the allocated memory to container does not change.
 - Applications need to be colocated after reducing memory, which in turn might affect the PSI