# Load Stall Minimization

**Hassan S. A. Arafat, David Bremner, Kenneth B. Kent**
{harafat,bremner,ken}@unb.ca
**Faculty of Computer Science, University of New Brunswick**

**Julian Wang**
zlwang@ca.ibm.com
**IBM JIT PPC CodeGen, IBM Canada**

## Problem Statement

- Modern processors are very fast.
- Modern memory has failed to keep up.
- Can result in the processor waiting for data to load from memory.
- Caches can mitigate the impact sometimes.
- On cache miss, get data from slow main memory.
- Wasted time, lower performance.
- Certain code patterns are more likely to cause those waits.
- Aim to reduce the number of times the processor is forced to wait.
- Many modern languages, e.g., Java, run in an environment called a Virtual Machine (VM).
- Different design choices can change the number of waits.

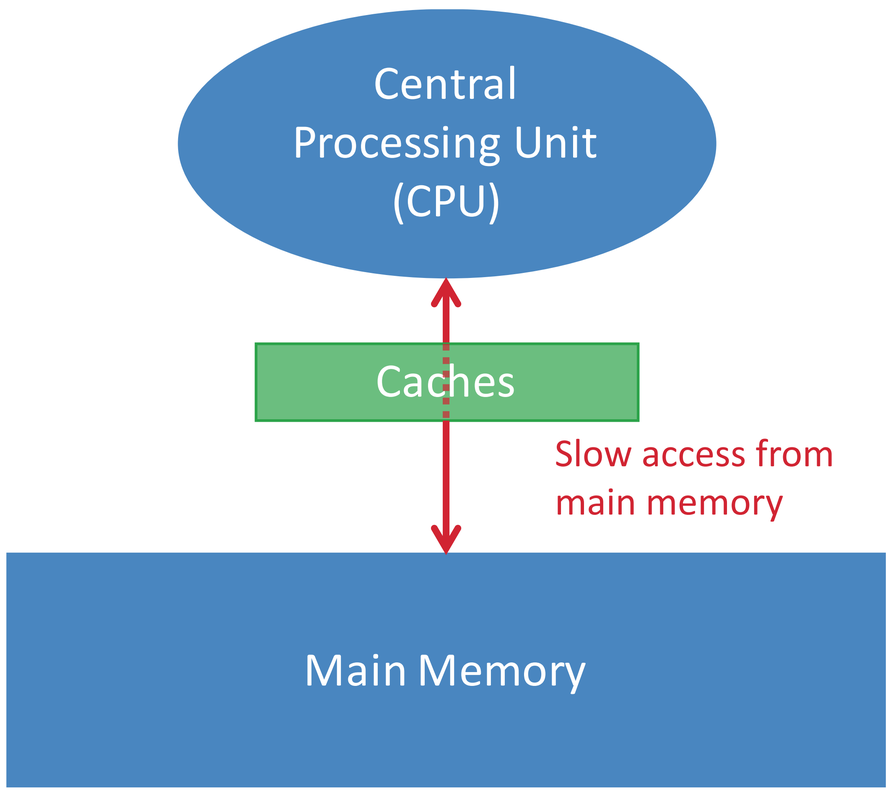


Fig. 1, Memory accesses

## Solution Rationale

- We want to reduce the number of waits by using the cache more judiciously.
- The behaviour of the CPU is dependent on the program structure.
- Being able to predict what the CPU will be trying to access next would be very useful.
- We can modify the VM to provide us with information on the programs being run.

## Proposed solution

- We aim to make a predictor that specifically targets a problematic type of access.
- Our predictor analyzes the input program and makes a prediction.
- Those predictions can be used to better organize objects in memory and better utilize the caches.
- The better organization can help mitigate the number and effects of those waits and thus improve performance.

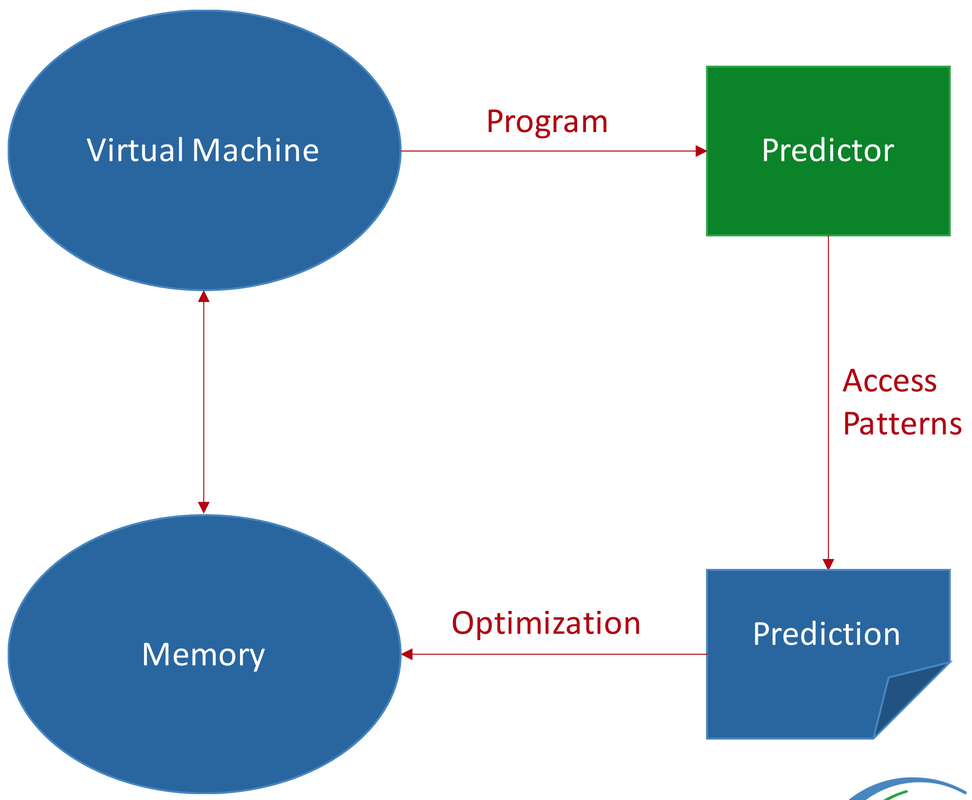


Fig. 2, Proposed solution structure

CAS–Atlantic