Automating the measure of the NAS MG cache behavior on many architectures using CControl and Grid'5000

Swann Perarnau, Pierre Neyron

Moais INRIA Team, CNRS LIG Lab, Grenoble University
France

Grid'5000 Spring School

Our Domain: HPC

High Performance Computing

Applications with huge computational requirements.
Typically run on dedicated systems.

Goal 1: Optimization

Measure an application behavior.
Understand architecture / system influence.
Identify optimization points.
Perform architecture-aware optimizations.

Goal 2: Better Runtimes

Develop portable optimization mechanisms.

Our Focus

Hardware caches

Increasingly critical for performance.
New architectures present complex cache hierarchies.
→ Measure/Understand cache usage of applications.

CControl

A new tool to control the cache available to an application.
Allows fine measurements of cache usage.
→ Used for intra-application cache partitioning.

Using Grid'5000

Testing CControl

Validate cache partitioning.
Use different architectures.

Analyze a classic benchmark

Ensure our results matches literature.
Test CControl applicability.
Understand architecture impact on performance.

Why Grid'5000

Control over the software stack.
Multiple versions of processors.

Outline

Classic Metrics

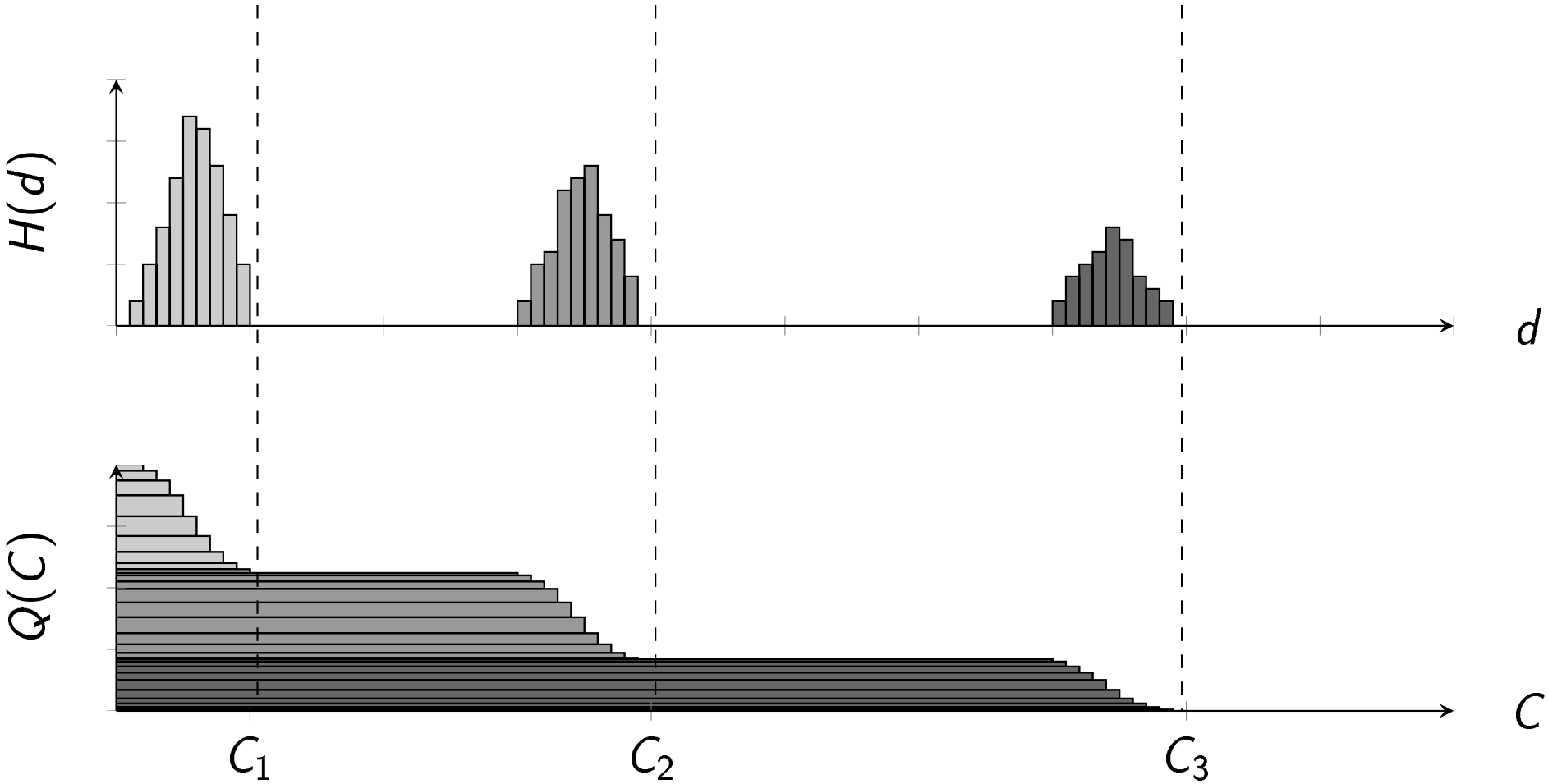
Reuse Distance

For each memory access: number of different memory accesses before the next access to the same location.

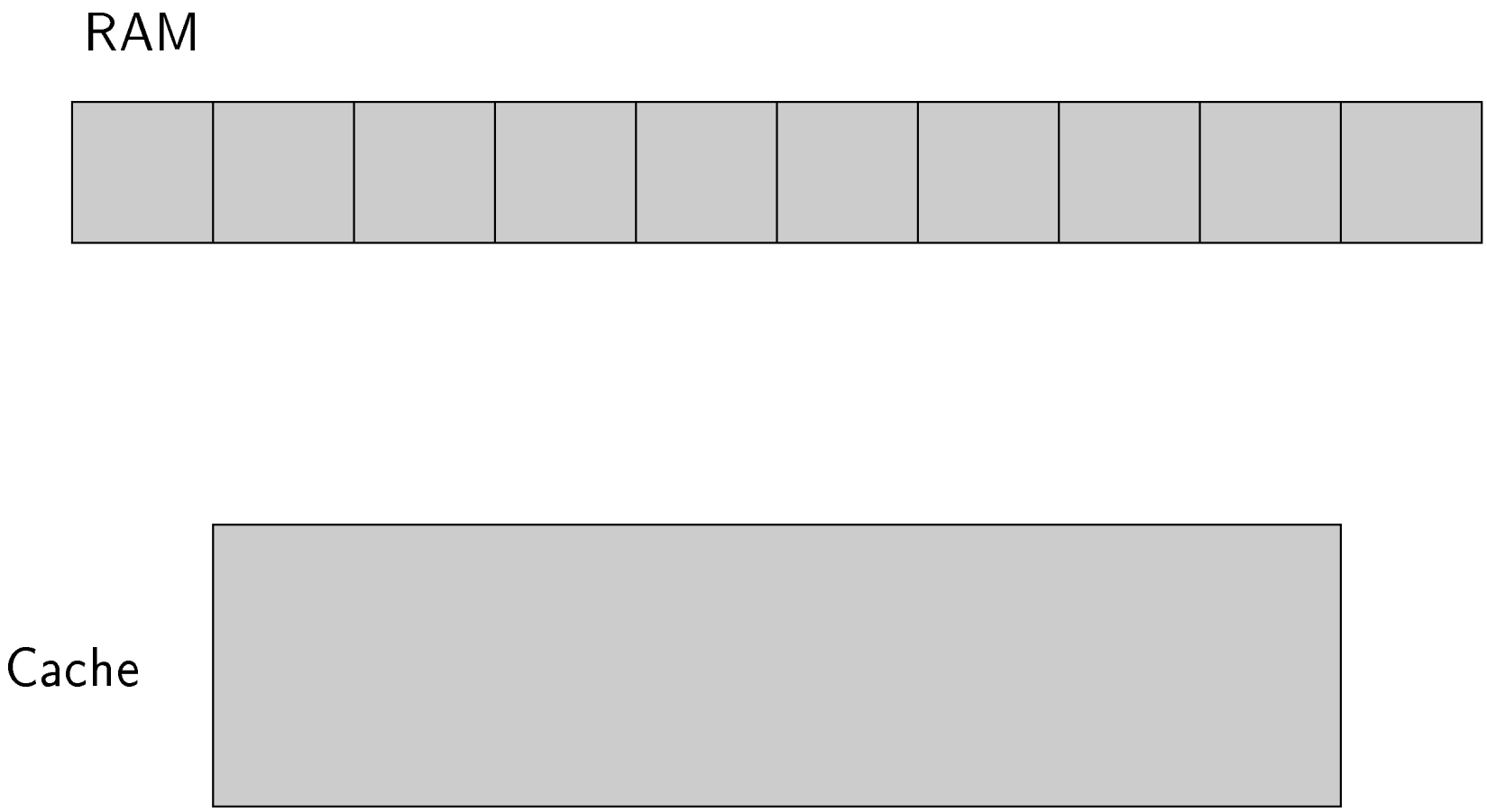
Working Set

Amount of cache required to achieve a given performance.

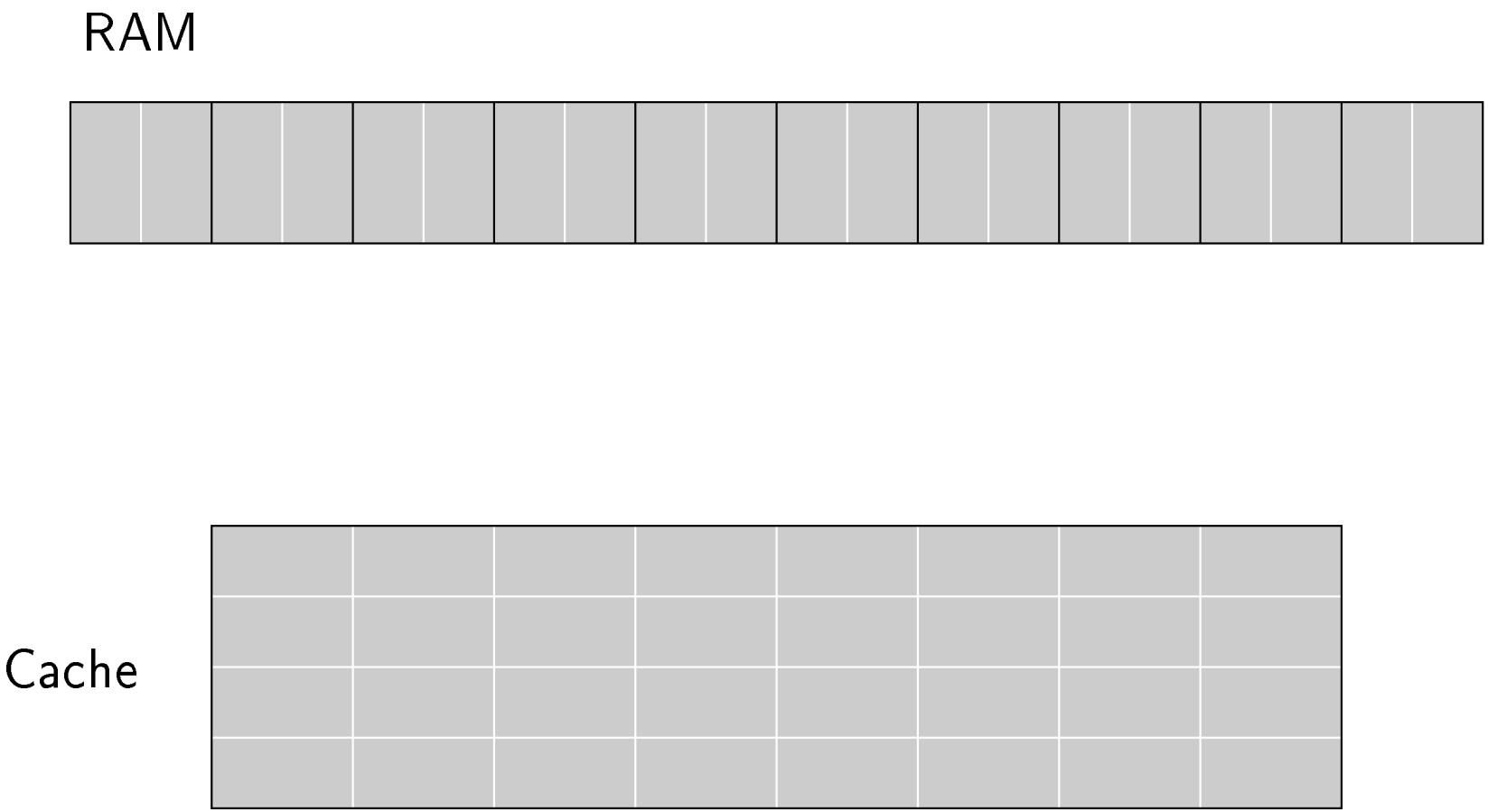
WS vs Reuse Distance



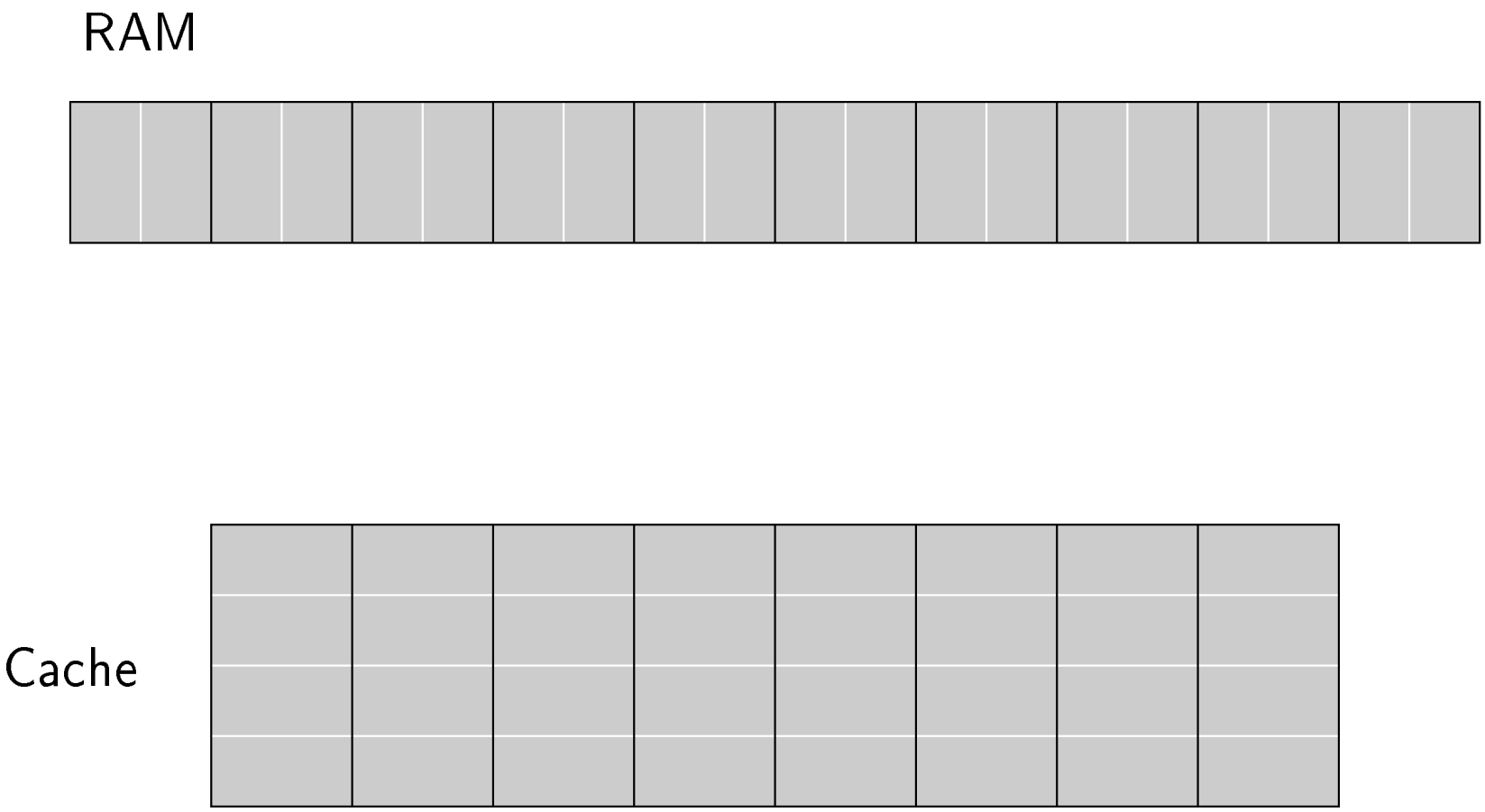
Page Coloring



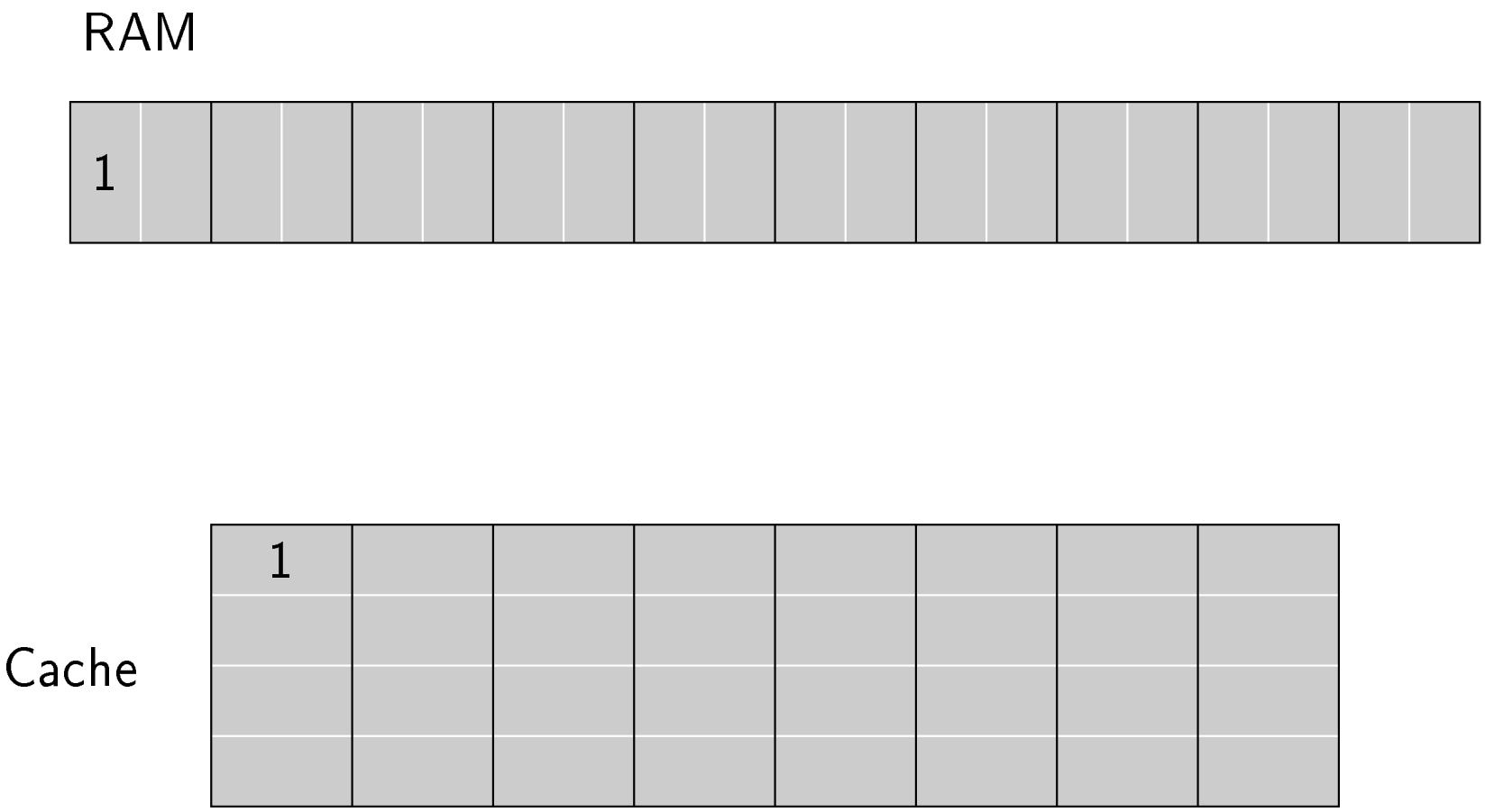
Page Coloring



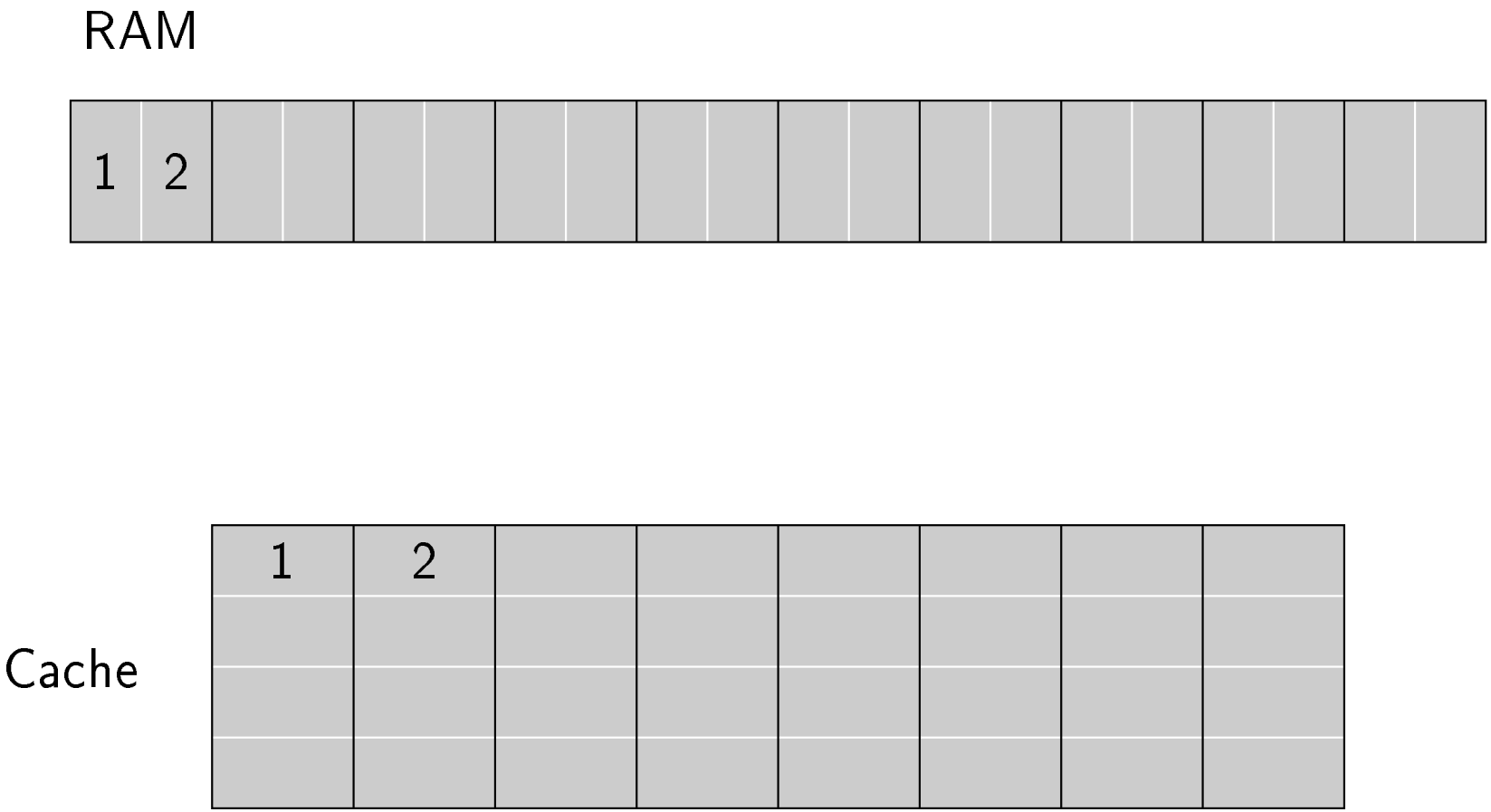
Page Coloring



Page Coloring

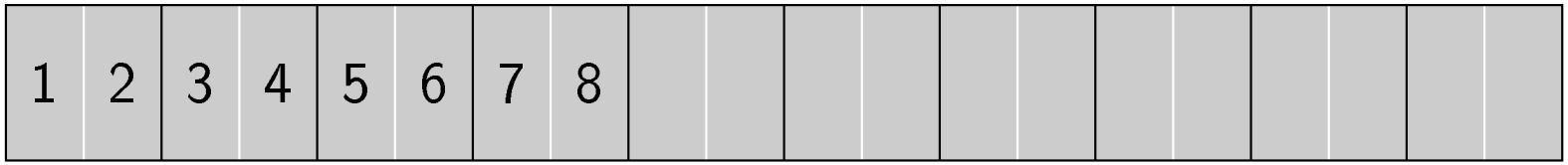


Page Coloring

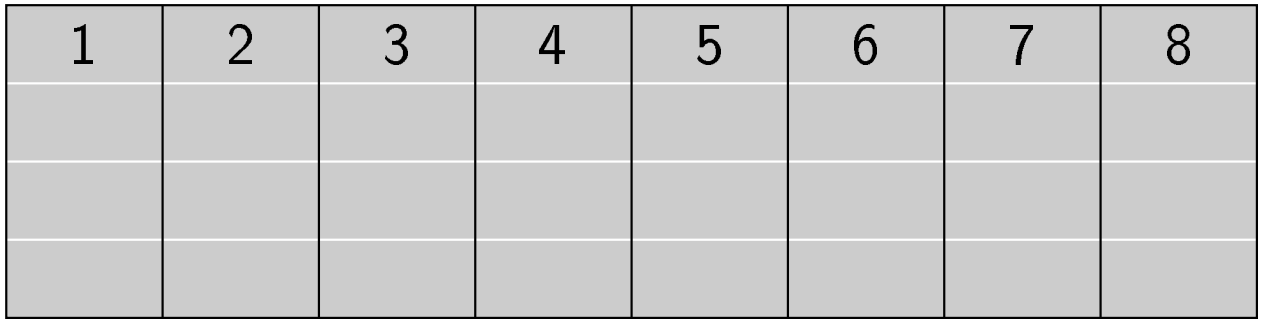


Page Coloring

RAM

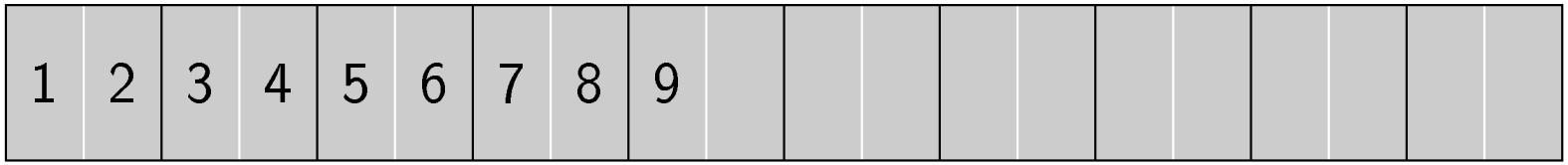


Cache

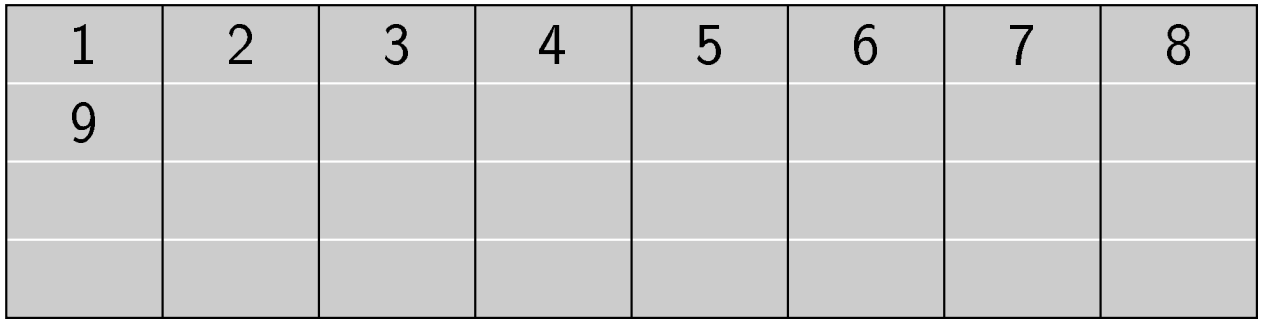


Page Coloring

RAM

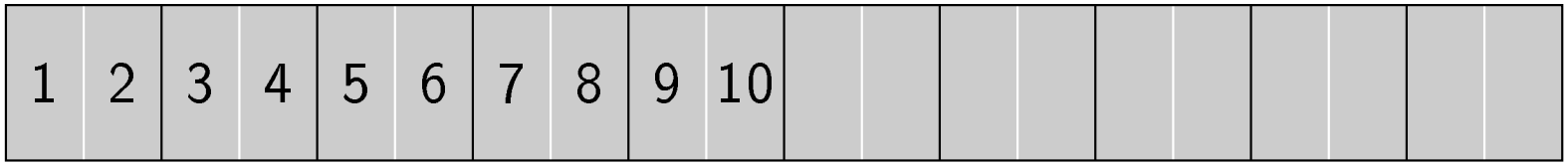


Cache

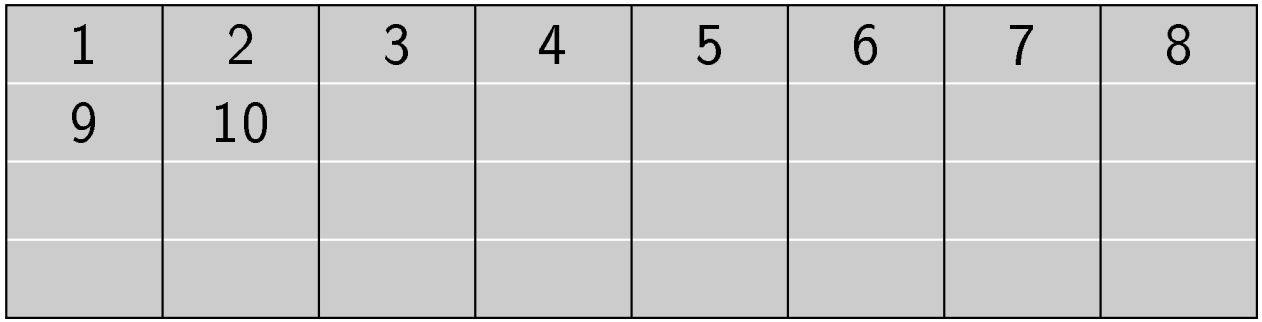


Page Coloring

RAM



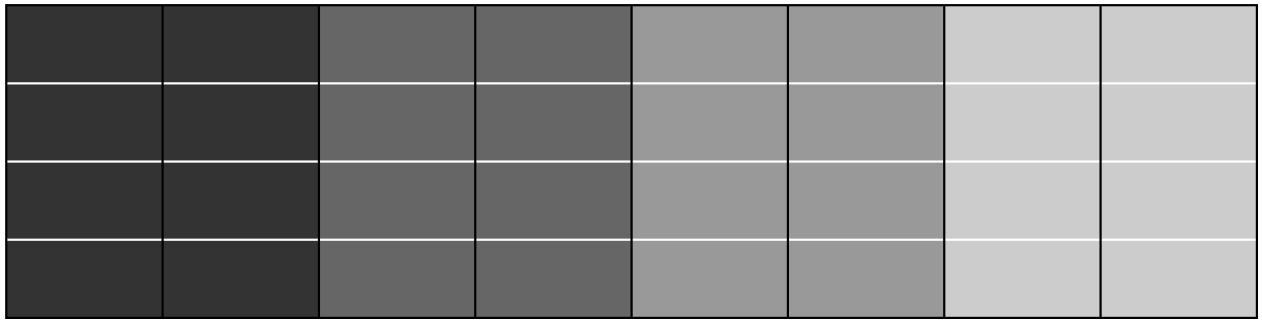
Cache



Page Coloring

RAM

Cache



Outline

Cache Partitioning

General Idea

Limit the number of colors a virtual memory address range can use.

Make it easy

Let user programs decide which colors to use.
Force a program to use specific colors.

User Control

- **Admin:** Load a Linux kernel module.
- **Module:** Allocate physical memory.
- **Module:** Export a virtual device (*control*).
- **Application:** `ioctl` *control*, asking colors.
- **Module:** create new virtual device with the good colors.
- **Application:** `mmap` the colored device.
- **Module:** On each page fault give a physical page of the good color.

Usage

```
#include<ccontrol.h>

void do_stuff(char *t, size_t s);

int main(void) {
    char *t;
    struct ccontrol_zone *z;
    color_set c;

    /* use first 32 colors */
    COLOR_ZERO(&c);
    for(int i = 0; i < 32; i++)
        COLOR_SET(i,&c);

    z = ccontrol_new(); // bookkeeping struct
    ccontrol_create_zone(z,&c,400); // create colored device

    /* alloc our char array in colored memory */
    t = (char *) ccontrol_malloc(z,100*sizeof(char));

    do_stuff(t,100);

    ccontrol_free(z,t); // free char array
    ccontrol_destroy_zone(z); // destroy device
    ccontrol_delete(z); // delete bookkeeping struct
    return 0;
}
```

Outline

Measure the Working Sets of NAS MG

NAS Parallel Benchmarks

Popular benchmark in HPC.
Several applications coming from the NASA.
Benchmark the system using fixed input.
→ OpenMP C version 2.3.

Experiment

Measure Mop/s on different cache sizes.
Use 4 cores, sharing the same cache.
Use various architectures.

Using Grid'5000

Resource API

Discover interesting hardware (CPU model).
Check resource requirements (RAM/CPU).

Controlled Software

Same OS stack for all experiments.
Custom image (Squeeze-based) deployed with kadeploy.

Root privileges

Kernel module install requires root access.
Real-time scheduling policy for increased stability.

Wished Features

Missing info

Architecture fields are not always filled right.
Missing specific fields: cache line size, cache associativity.

Machine selection

```
oarsub -p 'memory_size/(8*(cache_size/(cache_assoc*page_size)) >= 500MB'
```

BIOS access

Hardware prefetch problems.
Need to modify BIOS before experiment.

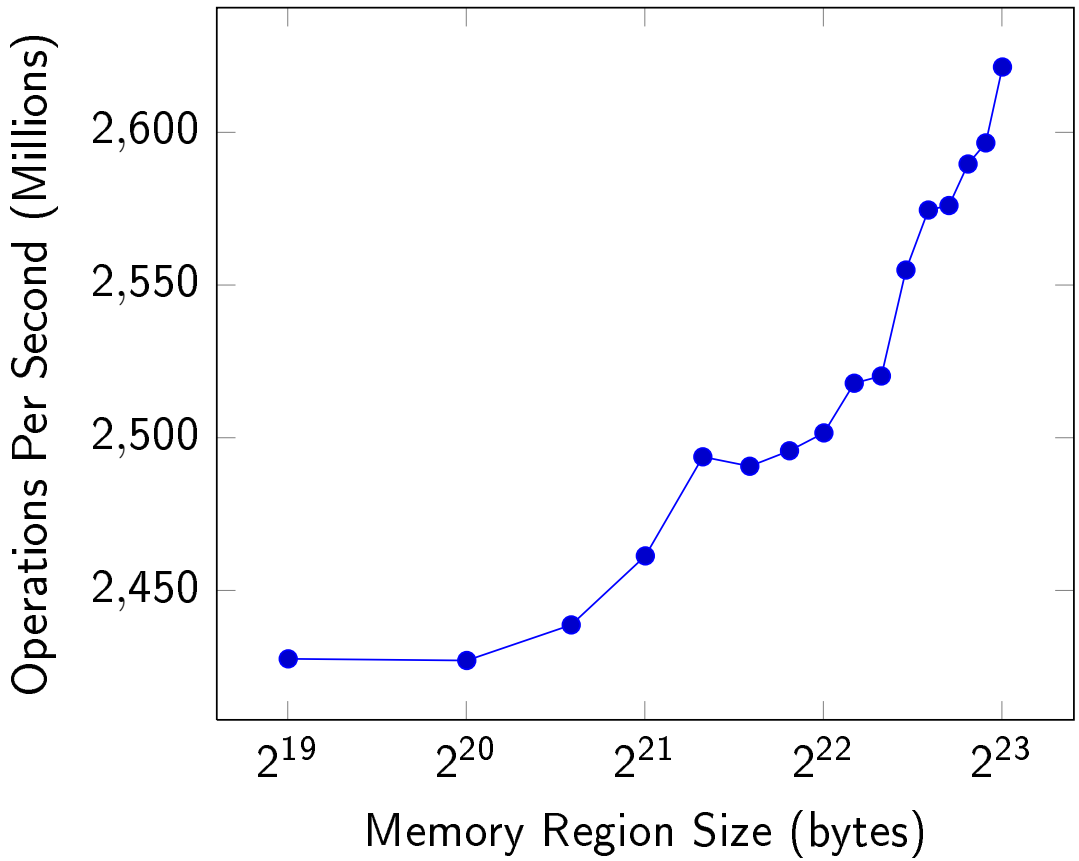


Figure: Operations per second (in Millions) of the NAS MG program under varying cache sizes (in bytes) on an Intel Xeon E5520 with 8 MB L2 cache.

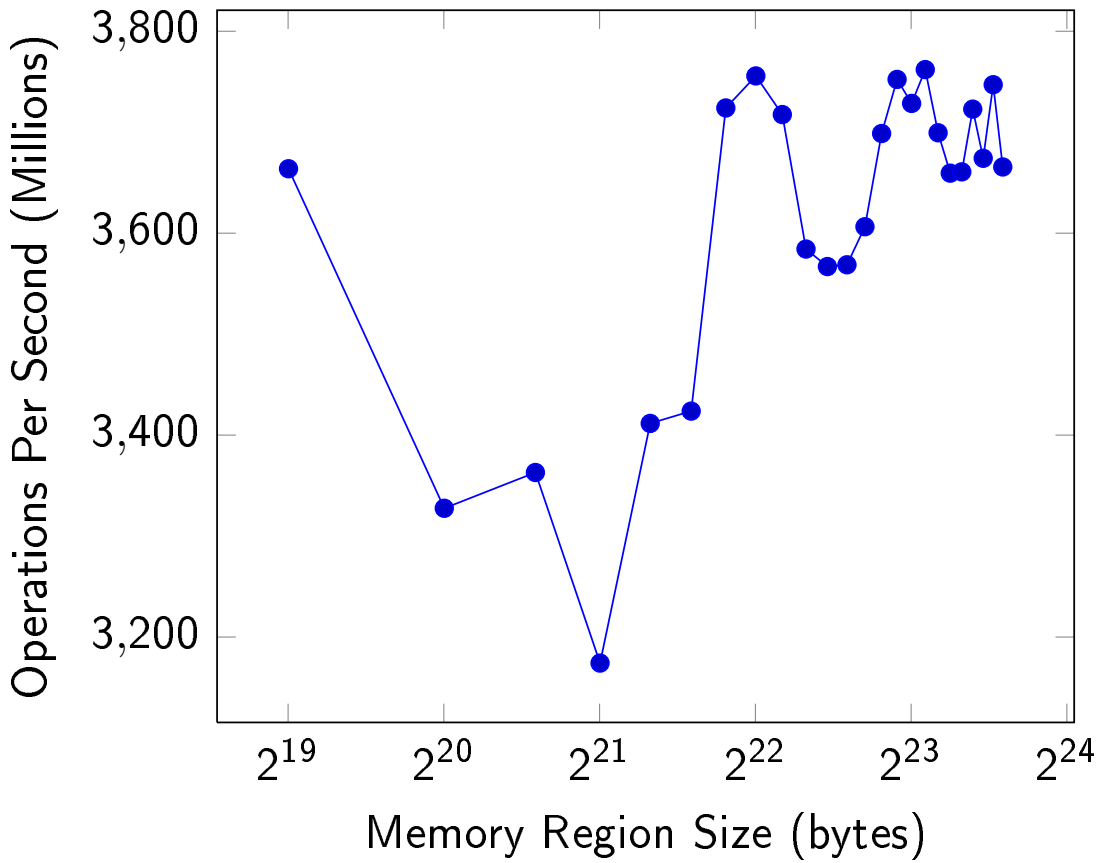


Figure: Operations per second (in Millions) of the NAS MG program under varying cache sizes (in MB). The hardware memory prefetcher is incompatible with our experiment on this platform.

What we have

A set of experiments to understand the memory behavior of the NAS.
Skeleton of automated experiment.

What's missing

Configurable BIOS.
More info inside the API.
More architectures ?