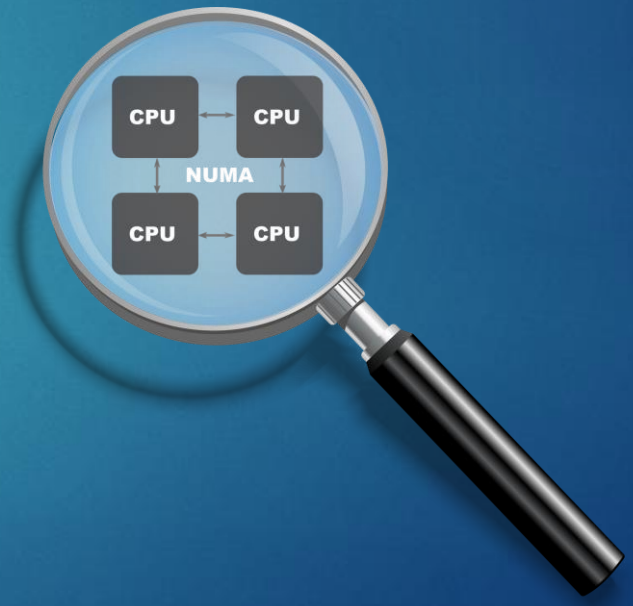
Sébastien Valat
15/02/2018

NUMAPROF

A NUMA MEMORY PROFILING TOOL



PLAN

- Reminder & what we want
- NUMAPROF internals
- GUI and example
- Conclusion

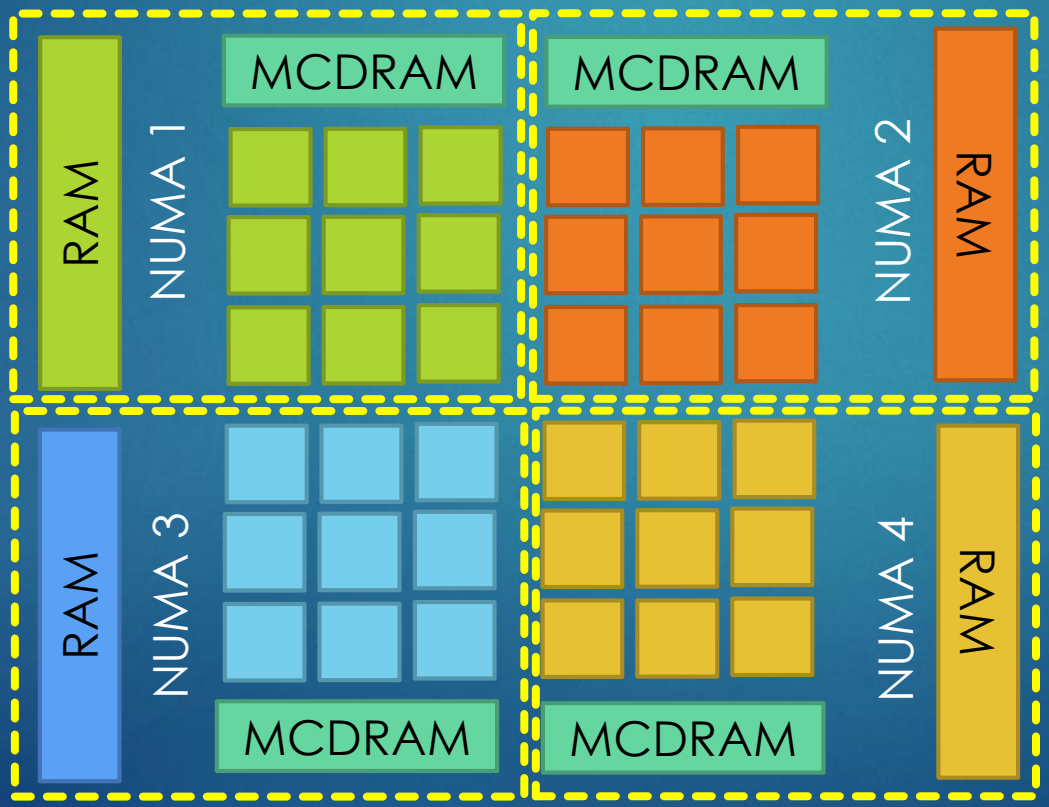
Sébastien Valat

Reminder on NUMA

Today topology

Sébastien Valat
15/02/2018

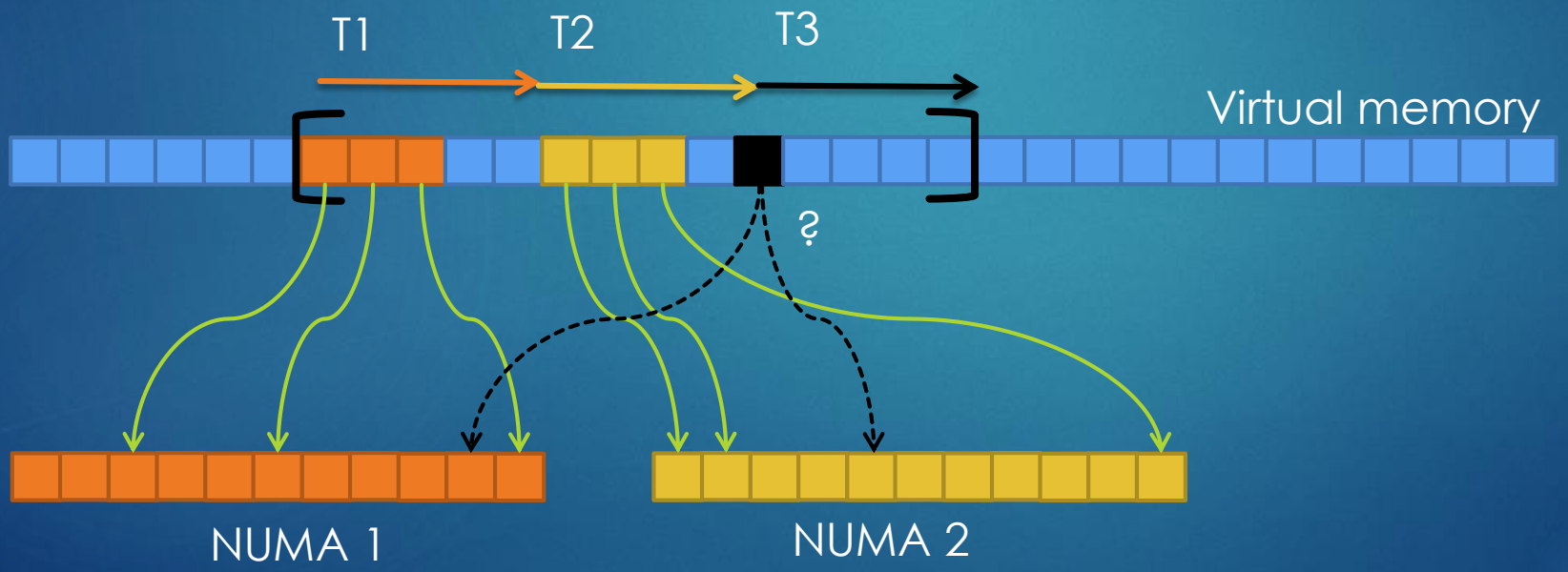
- Example current **Intel Knight Landing**, mode **SNC2** or **SNC4**
- Also add fast memory **MCDRAM** presented as **NUMA** or **LLC cache**



Implicit binding : first touch

Sébastien Valat
15/02/2018

- New allocated segments are **physically empty**
- They are filled on **first touch**
- Page selection **depend** of the **thread position**



Typical OpenMP mistake

Sébastien Valat
15/02/2018

- Make first **init outside of OpenMP** (in thread 1)
- So **each pages** will be first touched **on NUMA 1**

```
#pragma omp parallel for
for (int i = 0 ; i < SIZE ; i++)
        array[i] = 0;
```

- Then access

```
#pragma omp parallel for
for (int i = 0 ; i < SIZE ; i++)
        array[i]++;
```

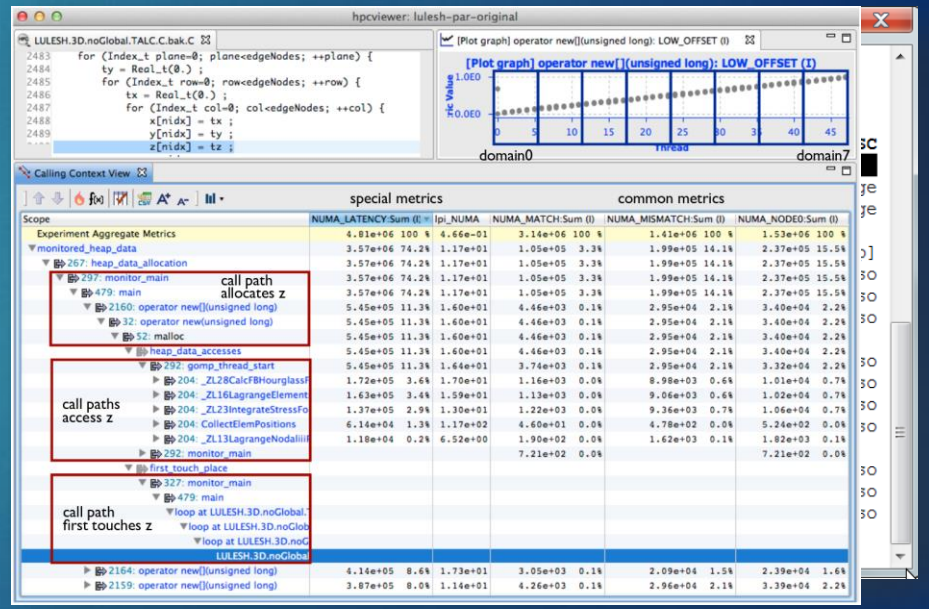
- **Bad performance** due to remote accesses !

Wish list for a profiling tool…

- We want to know if we make **remote accesses**
- Ideally we need to know **where**…
- We can dream, we want to know **which allocation contain issues**
- We want to know **where** the **first touch** has been done
- On KNL we want to check **MCRAM accesses**

Existing tools

- **MemProf** [1], a research paper from Grenoble
  - Hardware feature provided by AMD & kernel module, No GUI
- **SNPERF** [2], again a research project
  - Link utilization on time chart
- **NUMAgrind** [3]
  - Looks nice but not available
- **Numatop**
  - Similar to top, global profiling
- **HPCToolkit** [4]
  - Also have a nice interface, but hw counters & sampling



[1] Renaud Lachaize and Al. MemProf: A Memory Profiler for NUMA Multicore Systems.
[2] U. Prestor and Al., "An application-centric ccNUMA memory profiler,"
[3] Profiling Directed NUMA Optimization on Linux Systems: A Case Study of the Gaussian Computational Chemistry Code
[4] A Tool to Analyze the Performance of Multithreaded Programs on NUMA architectures

Sébastien Valat
15/02/2018

NUMPROF

HOW TO KNOW IF WE ARE RIGHT IN A REAL APPLICATION ?

NUMAPROF

- Take back the idea from **MALT**
  - **Web interface**
  - **Source annotation**
  - **Global metrics**

- Use intel **Pin**
  - Permit to **instrument** all **memory accesses**
  - **Parallel** opposite to valgrind
  - **Difficulty**: we cannot easily use libs inside the tool
  - I would have used hwloc and libnuma.....

On access we need…

- On each access we want to know if it is
  - **Remote** access
  - **Local** access
  - **MCDRAM** access
  - **Page is pinned**
  - **Thread is pinned**
- So, we need to know
  - Where is **the page**
  - Where is **the current thread**
- We can **skip** accesses to **local stack** (overhead 80x -> 40x)

Keep track of page mapping

Sébastien Valat
15/02/2018

- Can **query page location** with

```
int status
long ret = move_pages(0,1,pages,NULL,&status,0);
```

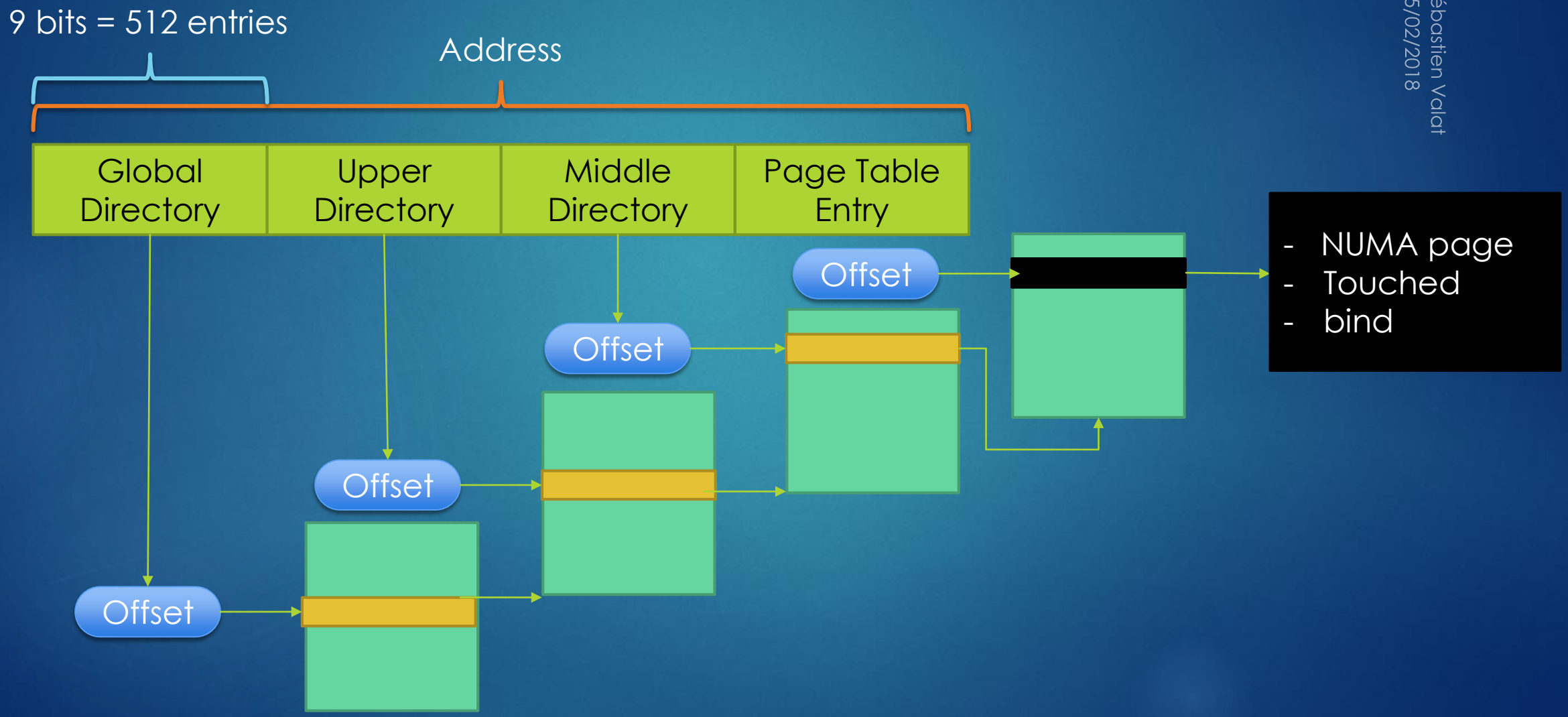
- It cost a **system call**
- **Cannot do** it for **every access**
- Need to build **a cache**

Similar to kernel page table

- I use the same layout than kernel **page table**
  - With **multiple levels** of 512 entries
- For each page **we track**
  - **NUMA location**
  - If has **already** been **touched**
  - If first touch was from the **binded or not binded** thread
- Need to track **mmap/mremap/munmap**
  - To **update** page **touched status**

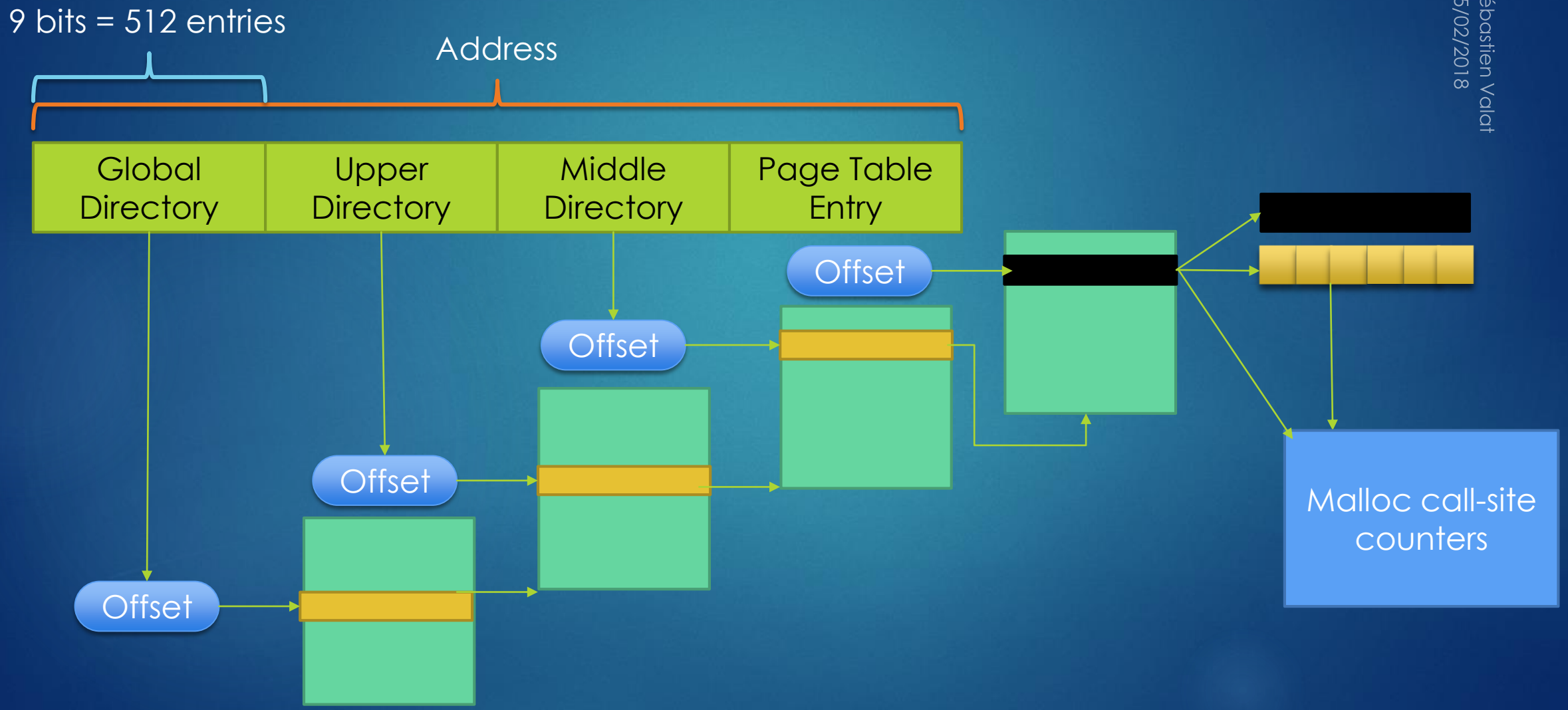
Shadow Page table

Sébastien Valat
15/02/2018



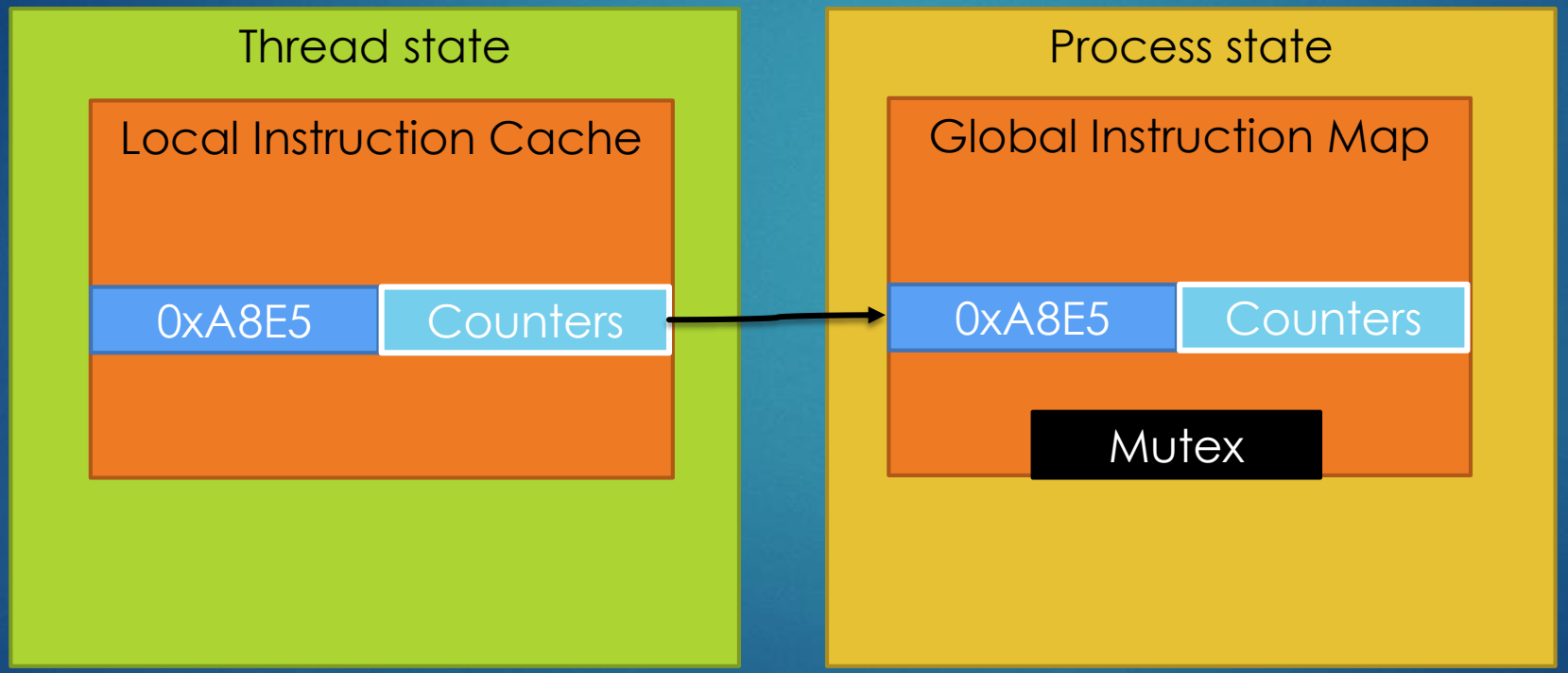
Allocation site

Sébastien Valat
15/02/2018



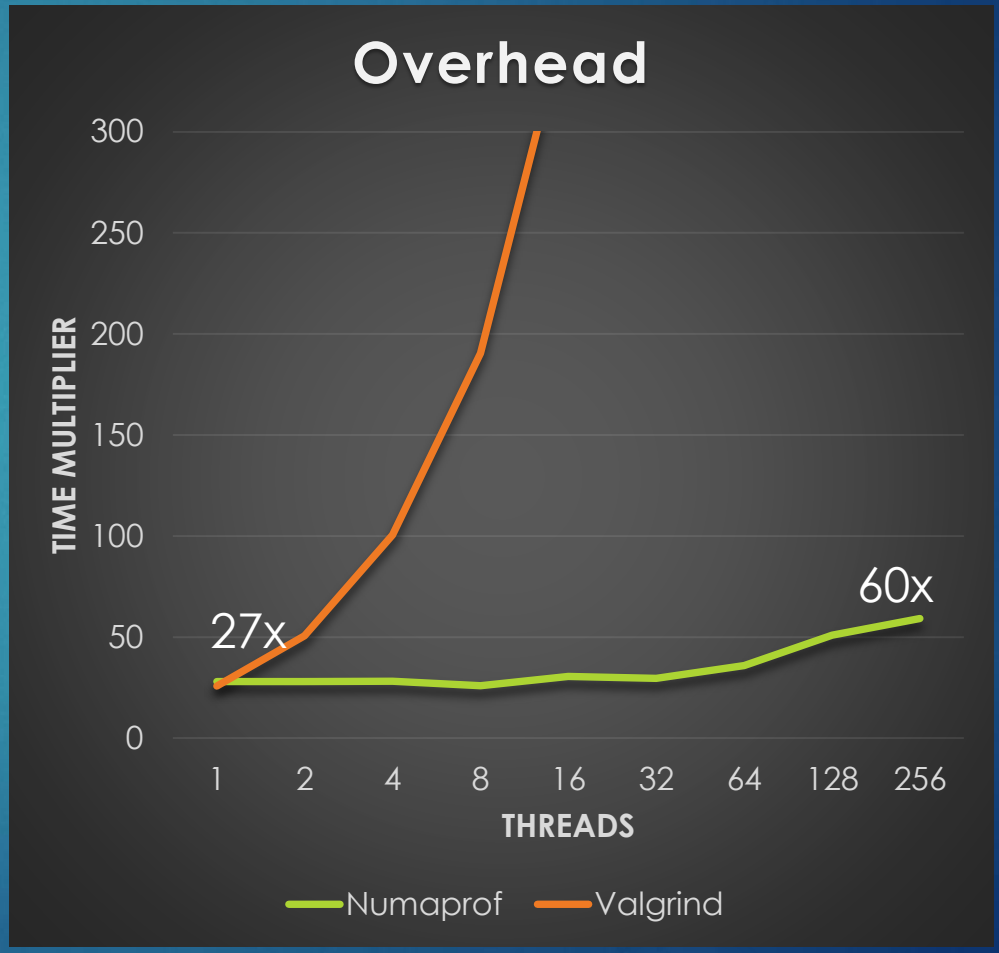
Limit mutexes & atomics

- I use caches to **accumulate locally** and **flush** sometimes



Overhead and scalability

- Of course overhead is large: **~30x**
- But is **scale**
- Example code hydro on **KNL**:



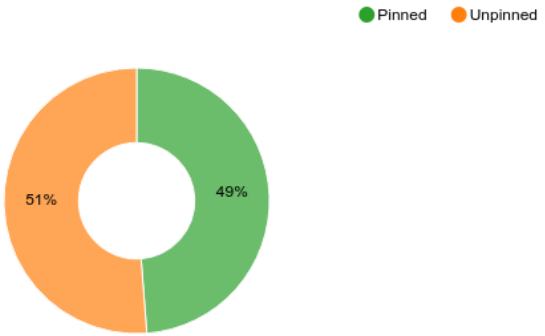
Sébastien Valat
15/02/2018

GUI and example

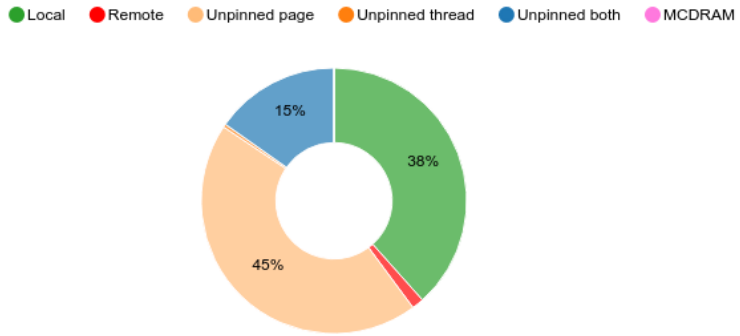
Global summary

Sébastien Valat
15/02/2018

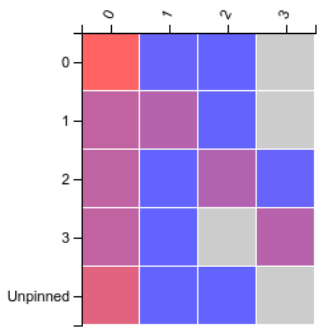
First touch



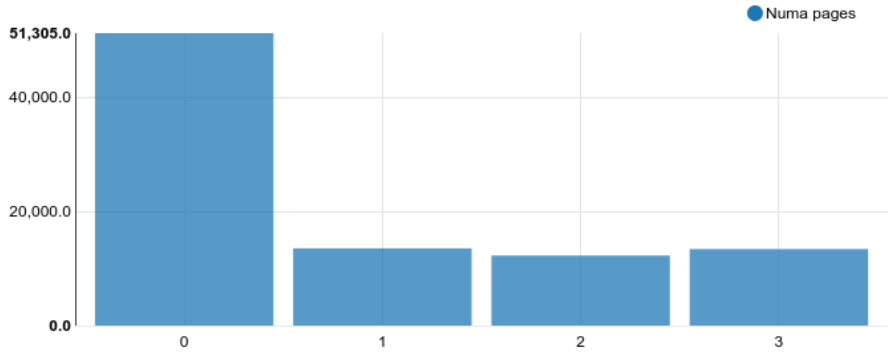
Memory access



Access matrix

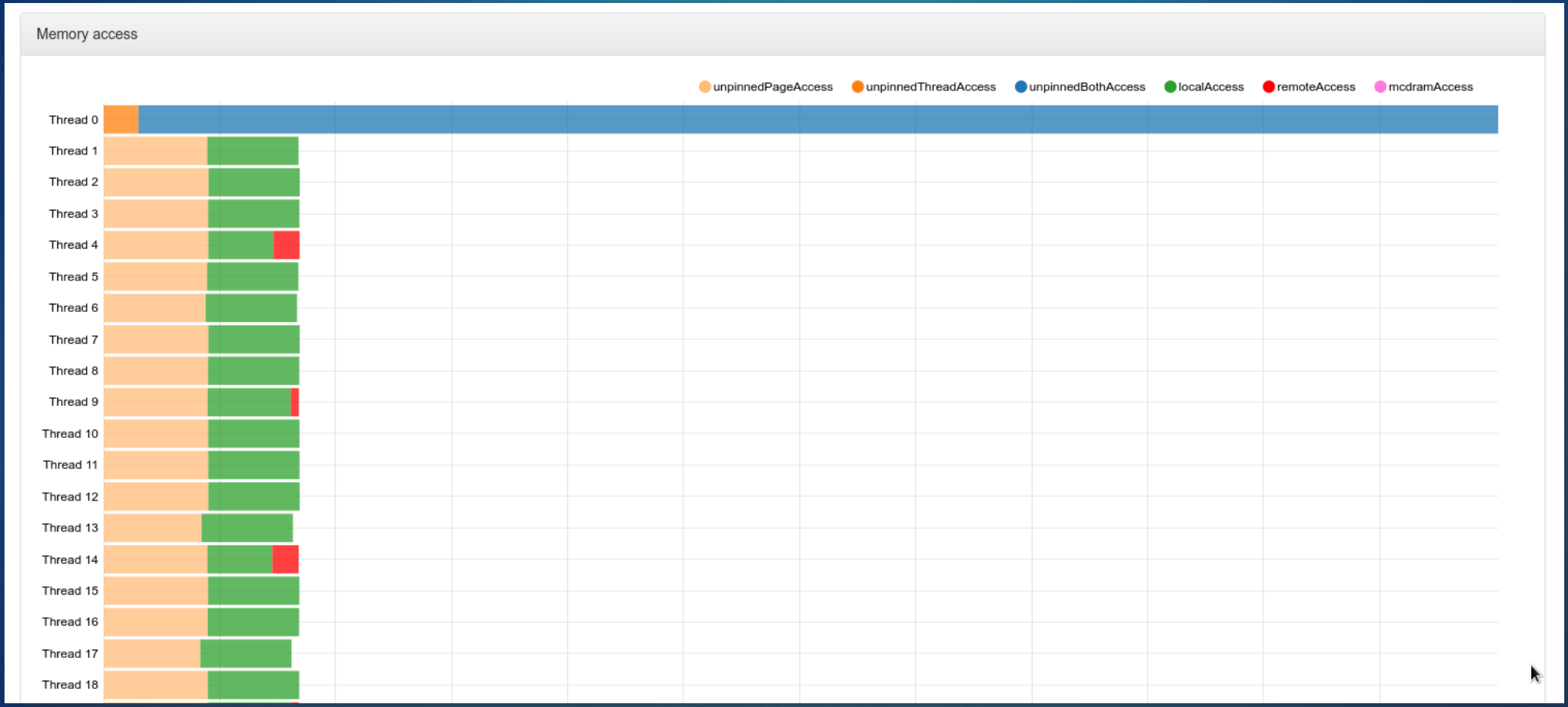


Peak allocated numa pages



Statistics per thread

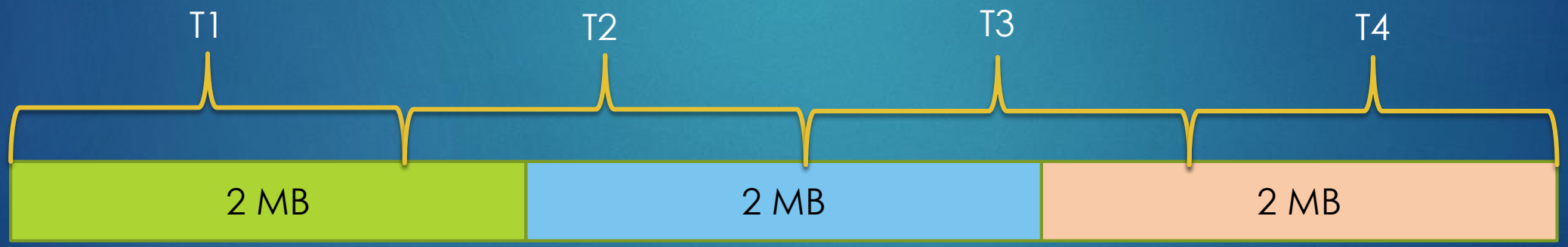
Sébastien Valat
15/02/2018



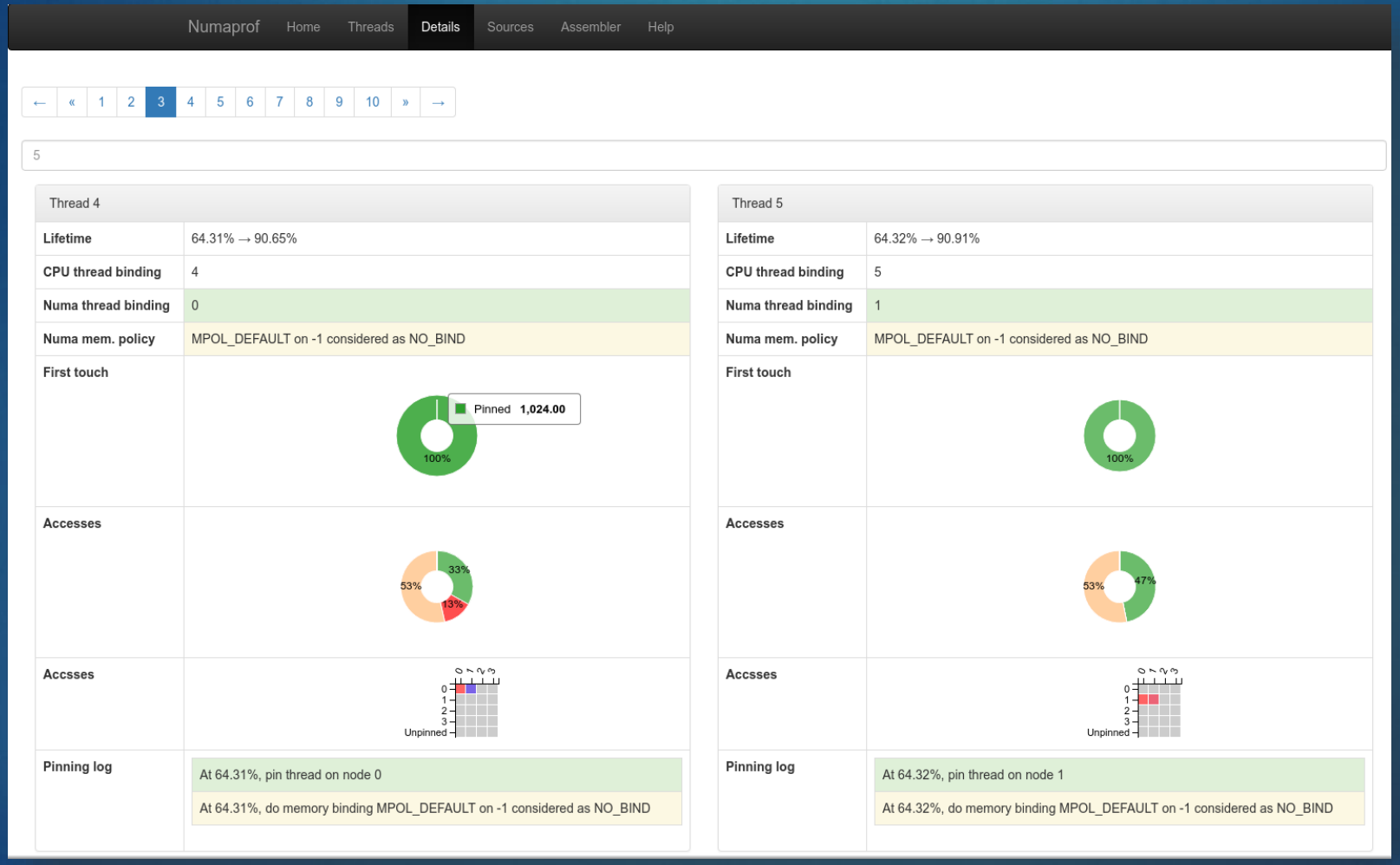
OMP and huge pages

Sébastien Valat
15/02/2018

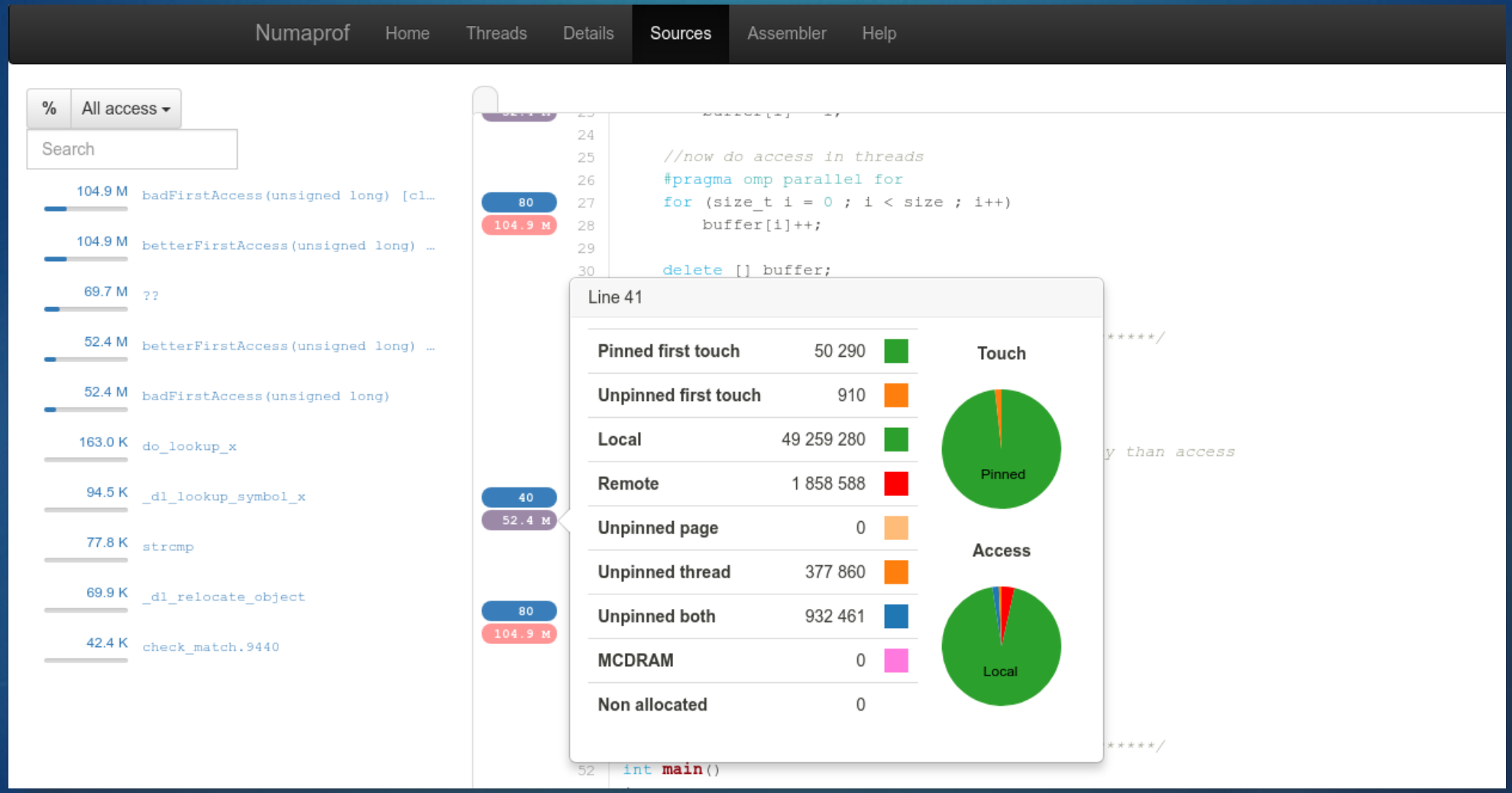
- Huge pages & thread splitting
- Most of the time do not match exactly
- Not a big issue if limited



Details per thread

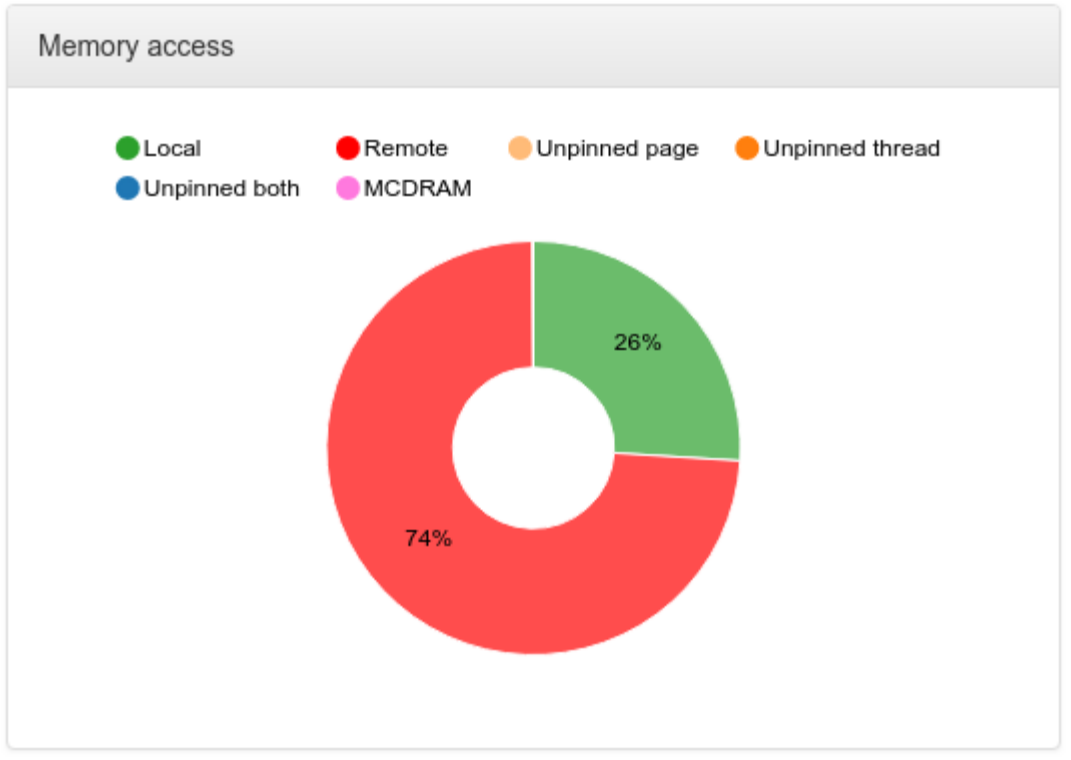


Source & asm annotations

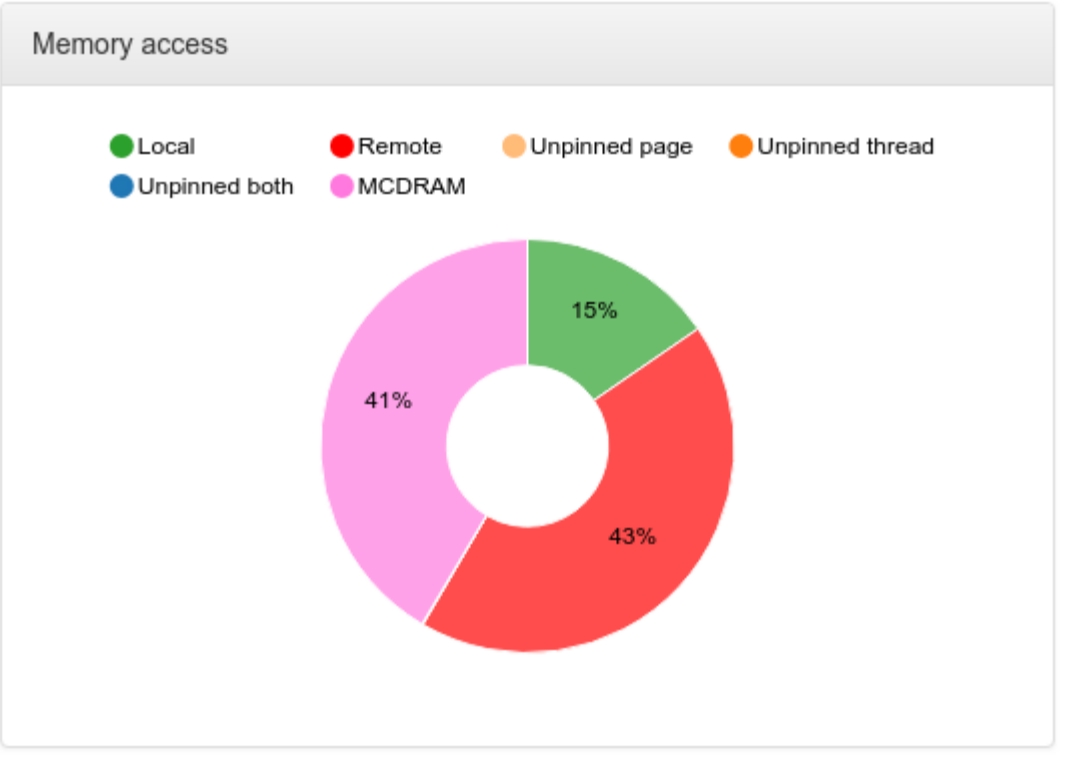


Code Hydro

- KNL Without HBM

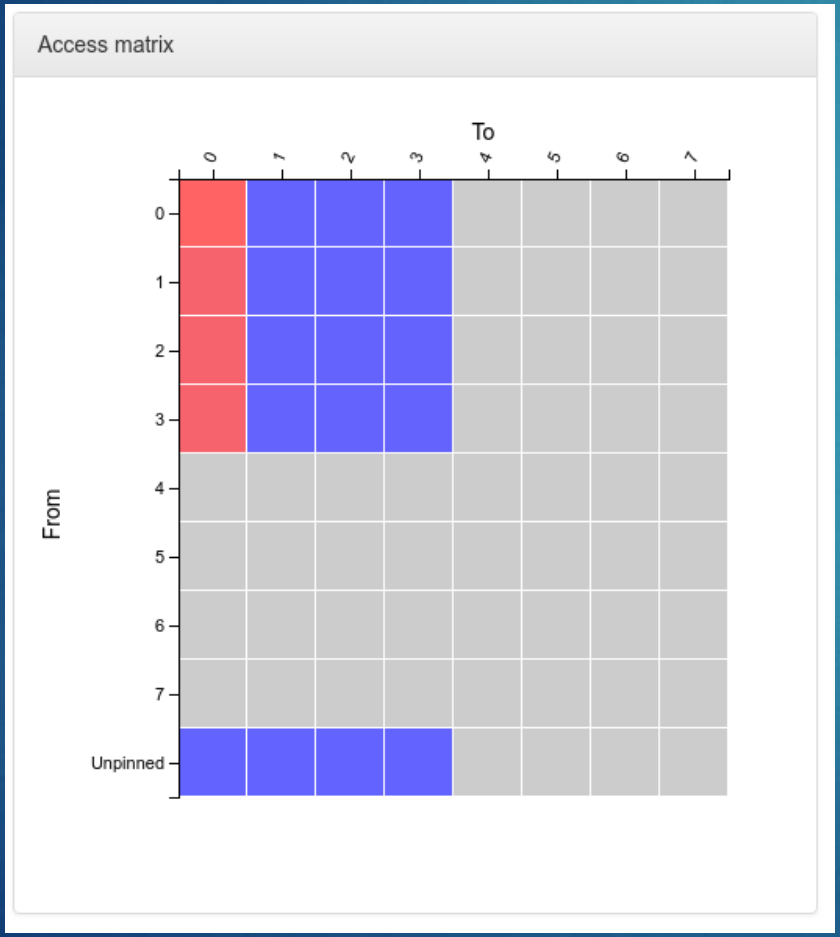


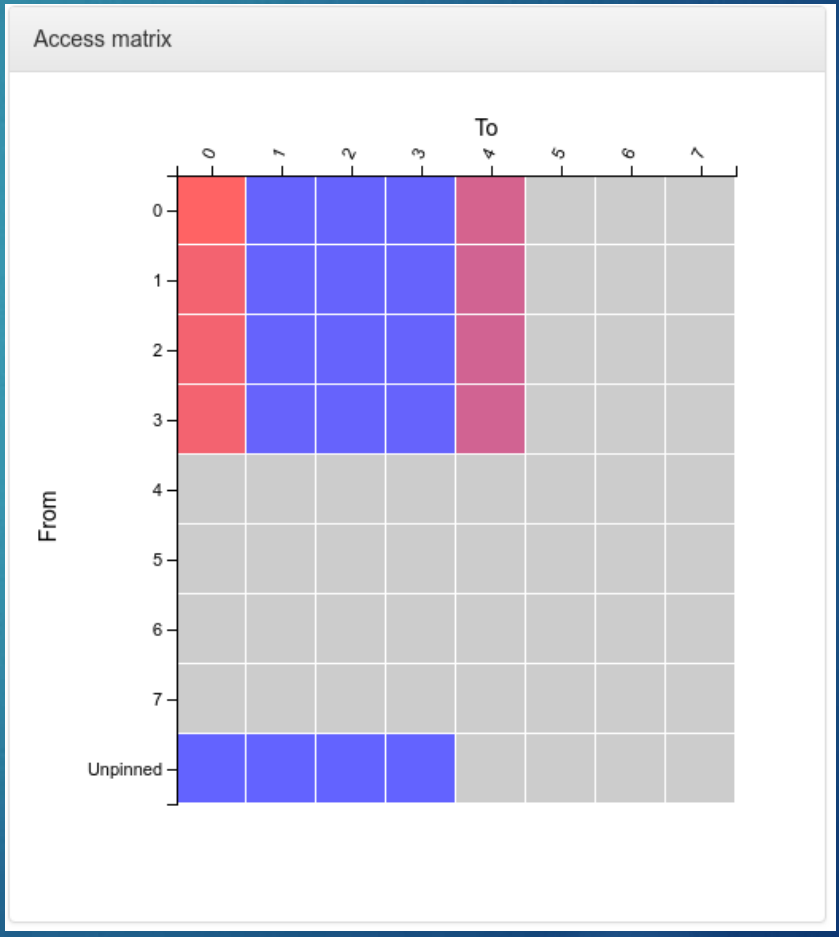
WITH HBM



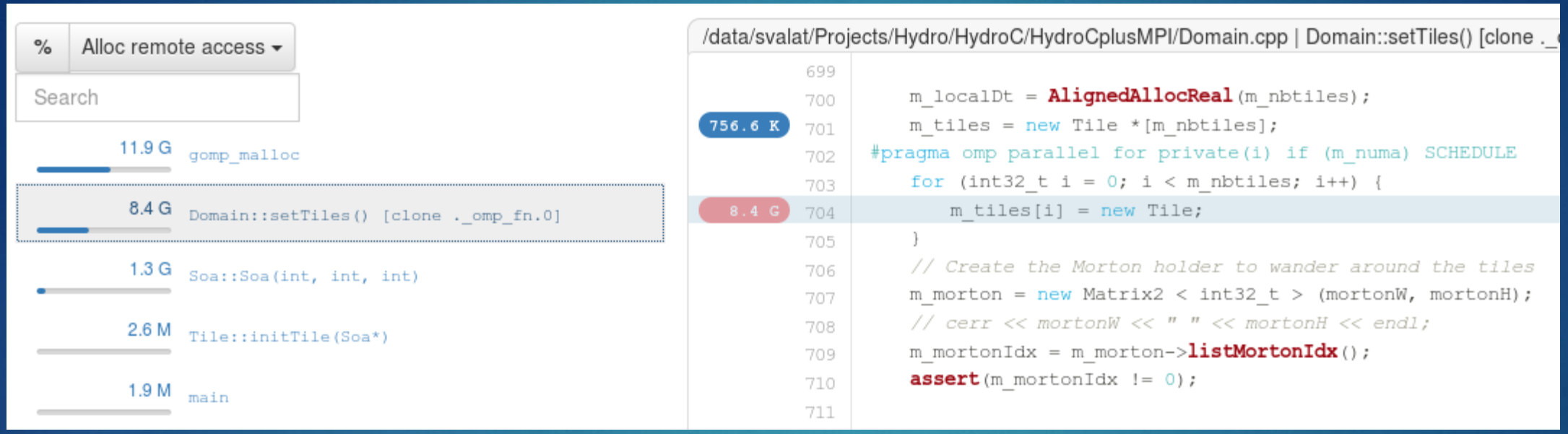
Original Hydro access matrix

Sébastien Valat
15/02/2018



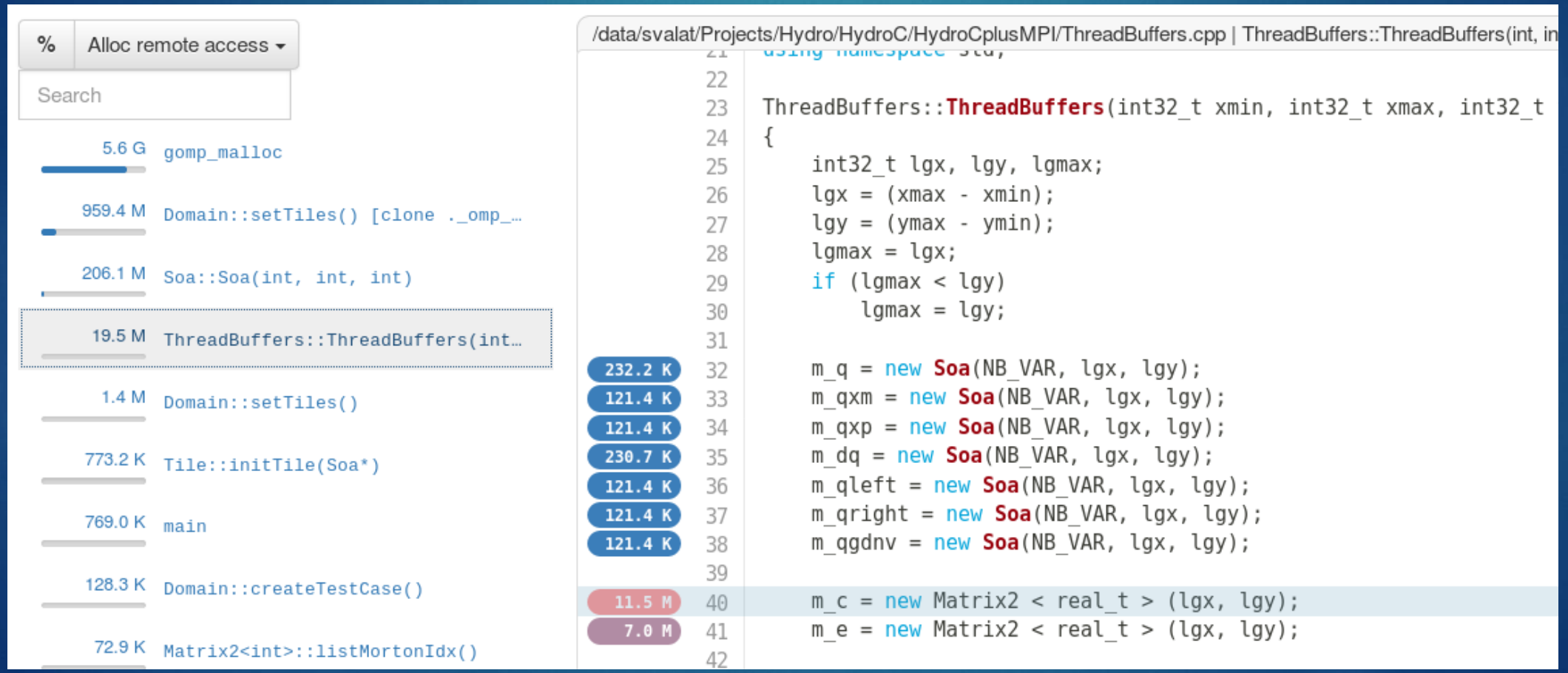


Ordering issue




```
#pragma omp parallel for private(i) if (m_numa) SCHEDULE
        for (int32_t i = 0; i < m_nbtiles; i++) {
                int t = m_mortonIdx[i];
                m_tiles[t] = new Tile;
        }
```

Non parallel allocations



Parallel allocations

- Original

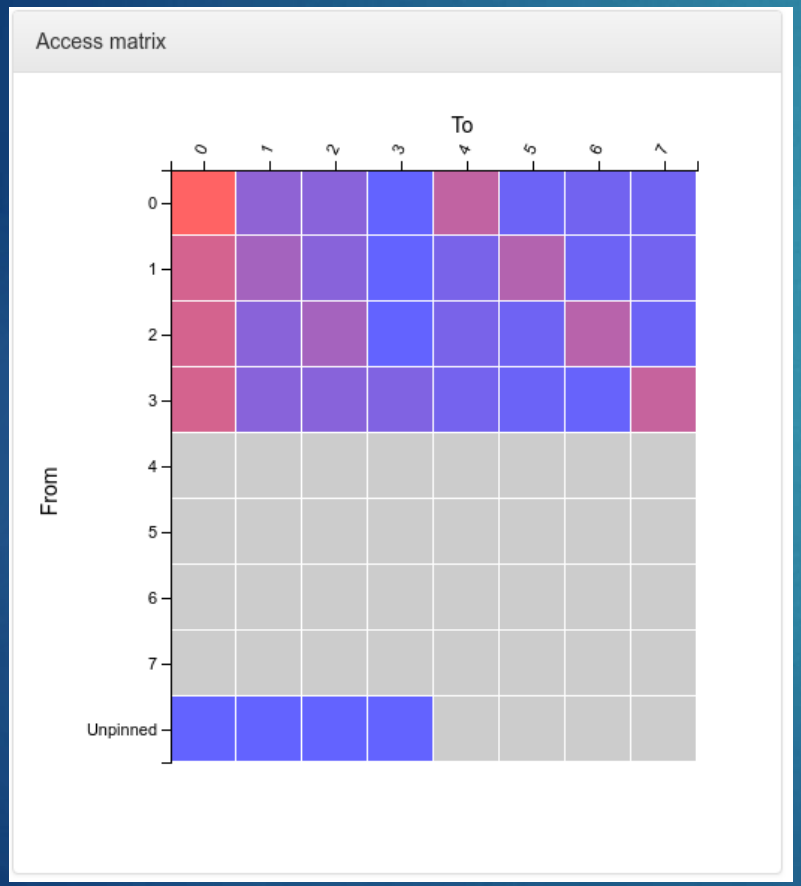
```
for (int32_t i = 0; i < m_numThreads; i++) {
        m_buffers[i] = new ThreadBuffers(…);
        assert(m_buffers[i] != 0);
}
```

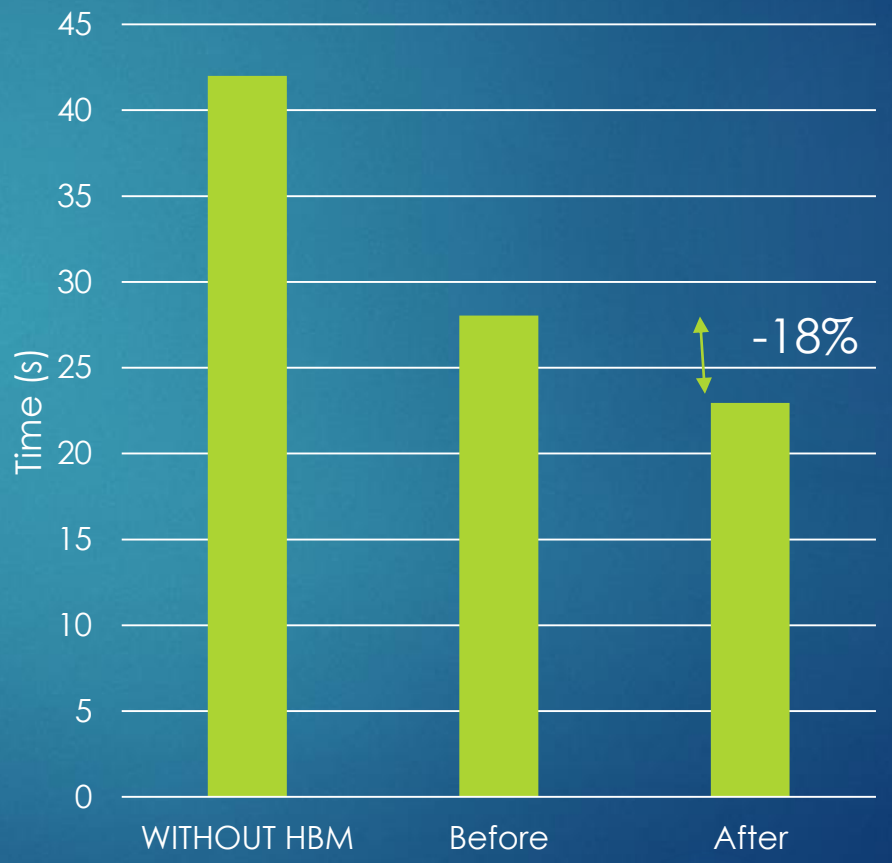
- Modified

```
#pragma omp parallel
{
        int i = omp_get_thread_num();
         #pragma omp critical
        m_buffers[i] = new ThreadBuffers(..);
        assert(m_buffers[i] != 0);
}
```

Speed up obtained on Hydro

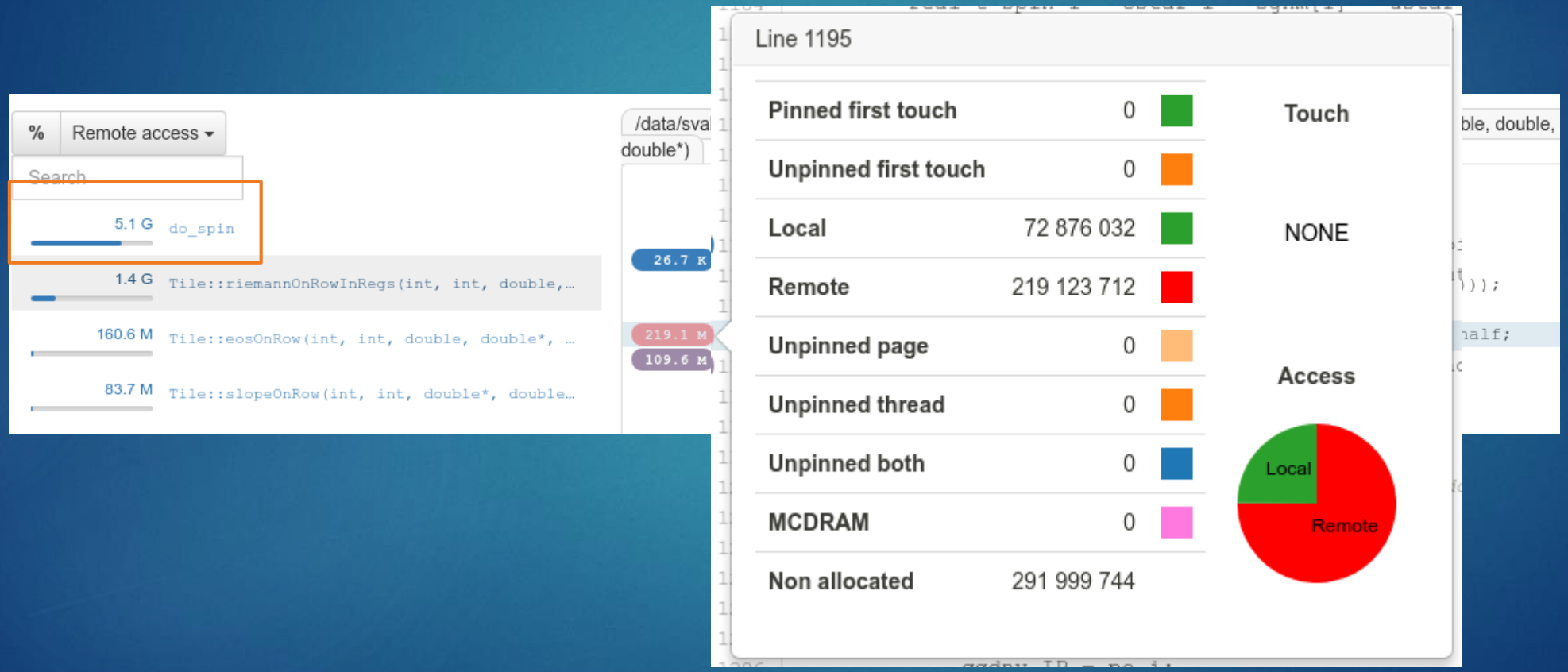
Sébastien Valat
15/02/2018





Remaining consts

Sébastien Valat
15/02/2018



Conclusion

- Tool easy to use coming with a nice GUI
- Useful to check an application.

- Of course, still a lot of works to do
  - Add support of **call stacks**
  - Consider **cache simulation**
  - **Optimizations**
  - **Time charts**

http://memtt.github.io/

- Also wan to to support **DynamorRIO** and **valgrind**.

Sébastien Valat
15/02/2018

BACKUP

```
hwloc-bind --cpubind node:0 ./mycommand
```

```
hwloc-bind –-membind node:0 ./mycommand
```

```
numactl –-membind –m 0 ./mycommand
```

How to control memory placement

FIRST SOLUTION : **BIND THE PROCESS** AT LAUNCH TIME

FINE TO SELECT **MCDRAM** IN **SNC1** MODE

BUT WHAT IF WE HAVE **4 NUMA** NODES ?

NEED TO CONTROL AT **THREAD LEVEL**

```
cpu_set_t cpuset;
CPU_ZERO(&cpuset);
CPU_SET(1,&cpuset);

pthread_set th = pthread_self();
pthread_setaffinity_np(thread, sizeof(cpu_set_t), &cpuset);
```

Thread binding

BIND A **THREAD** TO A **GIVEN CORE**

```
KMP_AFFINITY=scatter ./mycommand
```

```
OMP_PROC_BIND=TRUE ./mycommand
```

Extracted metrics

- First touch
  - **Pinned** first touch
  - **Unpinned** first touch
- Accesses
  - **Local** access
  - **Remote** access
  - **Unpinned thread** access
  - **Unpinned page** access
  - **Unpinned both** access
  - **MCDRAM** access

Performance of this example

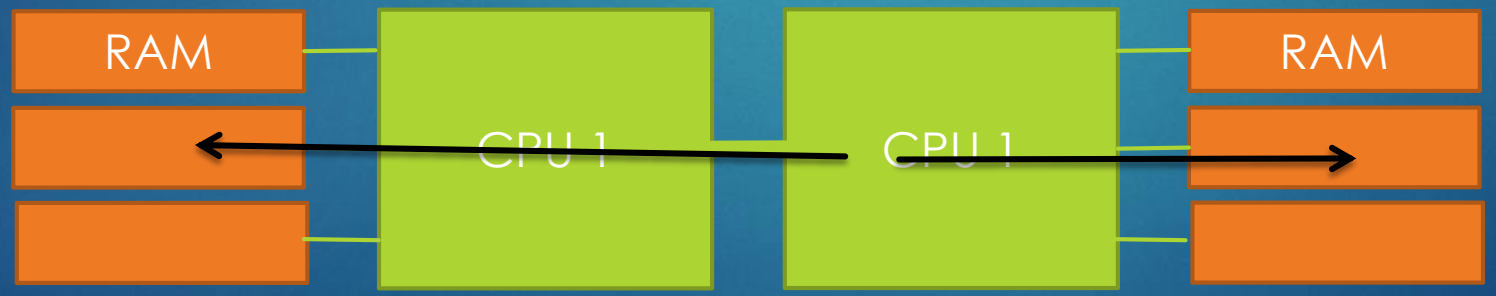
EXAMPLE OF RUN ON 8 THREADS, 2 NUMA NODES

THIS IS TRUE ONLY IF WE BIND THE THREADS

What is NUMA ?

Sébastien Valat
15/02/2018

- Each CPU has its **own memory**
- Access to **remote memory** we need to **go through the owner** CPU



Want to link to allocation site

- I want to provide statistics on **allocation site**
- Need on **each access** to know **where** the bloc **was allocated**
- Add **entries** into the **page table**
- **Split** page into blocs of **8 bytes**
- **Store** a **pointer** for each bloc to point the **segment descriptor**
- **Issue** if allocation are **smaller** then 8 bytes (**Incompatible** with jemalloc)