Design of Parallel and High Performance Computing
HS 2018
Markus Püschel, Torsten Hoefler **Homework 4**
Department of Computer Science
ETH Zurich

Filter Lock

Prove that the filter lock (given below) provides mutual exclusion for n threads. Assume sequential consistency.

```
volatile int level[n] = {0,0, ..., 0};
volatile int victim[n];

void lock(){
  for (int l=1; l<n; l++){
    level[tid] = l;
    victim[l] = tid;
    while ((∃k != tid) (level[k] >= l && victim[l] == tid)) {};
  }
}


void unlock() { level[tid] = 0; }
```

Solution

We will proof that for $0 \leq j \leq n$ there are at most $n - j$ threads at level j. We do that by induction. For $j = 0$, the base case, this is trivially true. For the induction step the induction hypothesis implies that there are at most $n - j + 1$ threads at level $j - 1$. To show that at least one thread cannot progress to level j, we argue by contradiction: assume there are $n - j + 1$ threads at level j. Let A be the last thread at level j to write to $victim[j]$. Because A is last, for any other B at level j:

$$W_B(victim[j]) \rightarrow W_A(victim[j])$$

From the code we see that B writes $level[B]$ before it writes $victim[j]$

$$W_B(level[B] = j) \rightarrow W_B(victim[j]) \rightarrow W_A(victim[j])$$

also from the code we see that A reads $level[B]$ after writing to $victim[j]$

$$W_B(level[B] = j) \rightarrow W_B(victim[j]) \rightarrow W_A(victim[j]) \rightarrow R_A(level[B])$$

Because B is at level j, every time A reads $level[B]$, it observes a value greater than or equal to j, implying that A could not have completed the waiting loop, a contradiction.

Measure Cache Misses

Consider the false sharing benchmark of Assignment n.2. Verify that the increase in time you observe when increasing the number of threads actually depends on the number of cache misses. This can be showed by instrumenting the benchmark with the PAPI library, that allows you to read the performance counters provided by your system.

Hint: Read the `PAPI_L1_DCM` counter inside each thread (i.e., inside the OMP parallel section) and report the maximum among the number of cache misses observed by each thread, as it is already done for the time.

Design of Parallel and High Performance Computing
HS 2018
Markus Püschel, Torsten Hoefler **Homework 4**
Department of Computer Science
ETH Zurich

Solution

See attached code.