CAP-399/2019 - Segunda Lista de Exercícios

Data: 2019-10-04

CONTEÚDO

EXERCÍCIO 1....2
Item (a)....2
Item (b)....3
EXERCÍCIO 2....7
Item (a)....7
Item (b)....9
REFERÊNCIAS....19

INFORMAÇÕES

Os arquivos estão em

```
fish://_______._______@login.sdumont.lncc.br/prj/padinpe/_______._______/
fish://_______._______@login.sdumont.lncc.br/scratch/padinpe/_______._______/
```

Os fontes foram obtidos em

```
fish://_______._______@login.sdumont.lncc.br/prj/padinpe/_______._______2/
```

A filha escolhida foi a `cpu_dev`

Fila	Wall-clock máximo (em horas)	Número mínimo de nós (núcleos+ dispositivos)	Número máximo de nós (núcleos+ dispositivos)	Número máximo de tarefas em execução por usuário	Número máximo de tarefas em fila por usuário	Custo em Unidade de Alocação (UA)
cpu (Nós B710)	96	21 (504)	50 (1200)	4	24	1
cpu_dev 1	0:20	1 (24)	4 (96)	1	1	1

fonte: https://sdumont.lncc.br/support_manual.php?pg=support

EXERCÍCIO 1

Item (a)

COMPILANDO

```
module load papi
module load papi-devel
module load intel_psxe
icc -O0 -o papiex $PAPI_INC papiex.c $PAPI_LIB -lpapi -lm
```

papi1.srm

```
#!/bin/bash
# Script baseado em:
#   "6.3. Jobs paralelos (threads/OpenMP)
#   Forma Geral de um Script"
#   <https://sdumont.lncc.br/support_manual.php?pg=support>
# Uso:
#   $ sbatch papi1.srm          (roda em prj)(anotar o my_job_no)
#   $ squeue --job my_job_no    (verifica jobs)
#   A saída "slurm-my_job_no.out" aparece em "scracth"

#SBATCH --nodes=1               #Numero de Nós  (a)
#SBATCH --ntasks-per-node=1     #Numero de tarefas por Nó  (b)
#SBATCH --ntasks=1              #Numero total de tarefas MPI  (a x b)
#SBATCH -p cpu_dev              #Fila (partition) a ser utilizada
#SBATCH -J pname                #Nome do job
#SBATCH --time=00:02:00         #Tempo limite
#SBATCH --exclusive             #Utilizacao exclusiva dos nós durante o job

echo '== CAP399/2019 - Lista 02 - Exerc 01 - Item (a) =='
echo 'Número total de tarefas (ntasks):' $SLURM_NTASKS
echo 'Listagem de Nós alocados para o Job:' $SLURM_JOB_NODELIST

nodeset -e $SLURM_JOB_NODELIST
cd $SLURM_SUBMIT_DIR

#Configura
module load papi/5.5.1.0
#module load papi-devel
module load intel_psxe/2019

#Configura o executavel
EXEC=/scratch/padinpe/_______._______/papiex

#exibe informações sobre o executável (opcional)
#echo /usr/bin/ldd $EXEC
/usr/bin/ldd $EXEC

#Dispara a execução
#   $SLURM_NTASKS : Same as --ntasks
# echo srun -n $SLURM_NTASKS $EXEC
srun -n $SLURM_NTASKS $EXEC
```

slurm-411278.out

```
== CAP399/2019 - Lista 02 - Exerc 01 - Item (a) ==
Número total de tarefas (ntasks): 1
Listagem de Nós alocados para o Job: sdumont1121
sdumont1121
     linux-vdso.so.1 =>  (0x00007ffda4369000)
     libpapi.so.5 => /opt/bullxde/perftools/papi/5.5.1.0/lib64/libpapi.so.5
(0x00002b19d547b000)
     libm.so.6 => /usr/lib64/libm.so.6 (0x00002b19d5738000)
```

```
     libgcc_s.so.1 => /usr/lib64/libgcc_s.so.1 (0x00002b19d5a3a000)
     libc.so.6 => /usr/lib64/libc.so.6 (0x00002b19d5c50000)
     libdl.so.2 => /usr/lib64/libdl.so.2 (0x00002b19d601d000)
     libpfm.so.4 => /opt/bullxde/perftools/papi/5.5.1.0/lib64/libpfm.so.4
(0x00002b19d6221000)
     /lib64/ld-linux-x86-64.so.2 (0x00002b19d5257000)
Num.Counters=11
PAPI_TOT_CYC,PAPI_TOT_INS,PAPI_FP_INS
Apos primeira leitura dos counters: 17542377, 23752959, 7347883, x=0.001549
Apos segunda leitura dos counters: 69305487, 95000614, 29126567, x=0.001549
All done
```

PAPI_TOT_CYC: is the processor cycle event
PAPI_TOT_INS: Instructions completed
PAPI_FP_INS: Floating point instructions

leitura contadores	PAPI_TOT_CYC processor cycle event	PAPI_TOT_INS Instructions completed	PAPI_FP_INS Floating point instructions
primeira	17.542.377	23.752.959	7.347.883
segunda	69.305.487	95.000.614	29.126.567

Na primeira leitura a quantidade de loops é dividida por quatro “for (i=0; i<TIMES/4; i++)”.
Portanto a segunda leitura deve estar próxima de quatro vezes a primeira leitura, o que se confirma.

Item (b)

```
[_______._______@sdumont13 ~]$ papi_avail
Available PAPI preset and user defined events plus hardware information.
--------------------------------------------------------------------------------
PAPI Version             : 5.5.1.0
Vendor string and code   : GenuineIntel (1)
Model string and code    : Intel(R) Xeon(R) CPU E5-2695 v2 @ 2.40GHz (62)
CPU Revision             : 4.000000
CPUID Info               : Family: 6  Model: 62  Stepping: 4
CPU Max Megahertz        : 3200
CPU Min Megahertz        : 1200
Hdw Threads per core     : 1
Cores per Socket         : 12
Sockets                  : 2
NUMA Nodes               : 2
CPUs per Node            : 12
Total CPUs               : 24
Running in a VM          : no
Number Hardware Counters : 11
Max Multiplex Counters   : 384
--------------------------------------------------------------------------------

================================================================================
  PAPI Preset Events
================================================================================
    Name        Code    Avail Deriv Description (Note)
PAPI_L1_DCM  0x80000000  Yes   No   Level 1 data cache misses
PAPI_L1_ICM  0x80000001  Yes   No   Level 1 instruction cache misses
PAPI_L2_DCM  0x80000002  Yes   Yes  Level 2 data cache misses
PAPI_L2_ICM  0x80000003  Yes   No   Level 2 instruction cache misses
PAPI_L3_DCM  0x80000004  No    No   Level 3 data cache misses
PAPI_L3_ICM  0x80000005  No    No   Level 3 instruction cache misses
PAPI_L1_TCM  0x80000006  Yes   Yes  Level 1 cache misses
PAPI_L2_TCM  0x80000007  Yes   No   Level 2 cache misses
```

```
PAPI_L3_TCM  0x80000008  Yes   No   Level 3 cache misses
PAPI_CA_SNP  0x80000009  No    No   Requests for a snoop
PAPI_CA_SHR  0x8000000a  No    No   Requests for exclusive access to shared
cache line
PAPI_CA_CLN  0x8000000b  No    No   Requests for exclusive access to clean cache
line
PAPI_CA_INV  0x8000000c  No    No   Requests for cache line invalidation
PAPI_CA_ITV  0x8000000d  No    No   Requests for cache line intervention
PAPI_L3_LDM  0x8000000e  No    No   Level 3 load misses
PAPI_L3_STM  0x8000000f  No    No   Level 3 store misses
PAPI_BRU_IDL 0x80000010  No    No   Cycles branch units are idle
PAPI_FXU_IDL 0x80000011  No    No   Cycles integer units are idle
PAPI_FPU_IDL 0x80000012  No    No   Cycles floating point units are idle
PAPI_LSU_IDL 0x80000013  No    No   Cycles load/store units are idle
PAPI_TLB_DM  0x80000014  Yes   Yes  Data translation lookaside buffer misses
PAPI_TLB_IM  0x80000015  Yes   No   Instruction translation lookaside buffer
misses
PAPI_TLB_TL  0x80000016  No    No   Total translation lookaside buffer misses
PAPI_L1_LDM  0x80000017  Yes   No   Level 1 load misses
PAPI_L1_STM  0x80000018  Yes   No   Level 1 store misses
PAPI_L2_LDM  0x80000019  No    No   Level 2 load misses
PAPI_L2_STM  0x8000001a  Yes   No   Level 2 store misses
PAPI_BTAC_M  0x8000001b  No    No   Branch target address cache misses
PAPI_PRF_DM  0x8000001c  No    No   Data prefetch cache misses
PAPI_L3_DCH  0x8000001d  No    No   Level 3 data cache hits
PAPI_TLB_SD  0x8000001e  No    No   Translation lookaside buffer shootdowns
PAPI_CSR_FAL 0x8000001f  No    No   Failed store conditional instructions
PAPI_CSR_SUC 0x80000020  No    No   Successful store conditional instructions
PAPI_CSR_TOT 0x80000021  No    No   Total store conditional instructions
PAPI_MEM_SCY 0x80000022  No    No   Cycles Stalled Waiting for memory accesses
PAPI_MEM_RCY 0x80000023  No    No   Cycles Stalled Waiting for memory Reads
PAPI_MEM_WCY 0x80000024  No    No   Cycles Stalled Waiting for memory writes
PAPI_STL_ICY 0x80000025  Yes   No   Cycles with no instruction issue
PAPI_FUL_ICY 0x80000026  No    No   Cycles with maximum instruction issue
PAPI_STL_CCY 0x80000027  No    No   Cycles with no instructions completed
PAPI_FUL_CCY 0x80000028  No    No   Cycles with maximum instructions completed
PAPI_HW_INT  0x80000029  No    No   Hardware interrupts
PAPI_BR_UCN  0x8000002a  Yes   Yes  Unconditional branch instructions
PAPI_BR_CN   0x8000002b  Yes   No   Conditional branch instructions
PAPI_BR_TKN  0x8000002c  Yes   Yes  Conditional branch instructions taken
PAPI_BR_NTK  0x8000002d  Yes   No   Conditional branch instructions not taken
PAPI_BR_MSP  0x8000002e  Yes   No   Conditional branch instructions mispredicted
PAPI_BR_PRC  0x8000002f  Yes   Yes  Conditional branch instructions correctly
predicted
PAPI_FMA_INS 0x80000030  No    No   FMA instructions completed
PAPI_TOT_IIS 0x80000031  No    No   Instructions issued
PAPI_TOT_INS 0x80000032  Yes   No   Instructions completed
PAPI_INT_INS 0x80000033  No    No   Integer instructions
PAPI_FP_INS  0x80000034  Yes   Yes  Floating point instructions
PAPI_LD_INS  0x80000035  Yes   No   Load instructions
PAPI_SR_INS  0x80000036  Yes   No   Store instructions
PAPI_BR_INS  0x80000037  Yes   No   Branch instructions
PAPI_VEC_INS 0x80000038  No    No   Vector/SIMD instructions (could include
integer)
PAPI_RES_STL 0x80000039  No    No   Cycles stalled on any resource
PAPI_FP_STAL 0x8000003a  No    No   Cycles the FP unit(s) are stalled
PAPI_TOT_CYC 0x8000003b  Yes   No   Total cycles
PAPI_LST_INS 0x8000003c  No    No   Load/store instructions completed
PAPI_SYC_INS 0x8000003d  No    No   Synchronization instructions completed
PAPI_L1_DCH  0x8000003e  No    No   Level 1 data cache hits
PAPI_L2_DCH  0x8000003f  Yes   Yes  Level 2 data cache hits
PAPI_L1_DCA  0x80000040  No    No   Level 1 data cache accesses
PAPI_L2_DCA  0x80000041  Yes   No   Level 2 data cache accesses
PAPI_L3_DCA  0x80000042  Yes   Yes  Level 3 data cache accesses
PAPI_L1_DCR  0x80000043  No    No   Level 1 data cache reads
PAPI_L2_DCR  0x80000044  Yes   No   Level 2 data cache reads
PAPI_L3_DCR  0x80000045  Yes   No   Level 3 data cache reads
PAPI_L1_DCW  0x80000046  No    No   Level 1 data cache writes
PAPI_L2_DCW  0x80000047  Yes   No   Level 2 data cache writes
PAPI_L3_DCW  0x80000048  Yes   No   Level 3 data cache writes
PAPI_L1_ICH  0x80000049  No    No   Level 1 instruction cache hits
PAPI_L2_ICH  0x8000004a  Yes   No   Level 2 instruction cache hits
PAPI_L3_ICH  0x8000004b  No    No   Level 3 instruction cache hits
PAPI_L1_ICA  0x8000004c  No    No   Level 1 instruction cache accesses
```

```
PAPI_L2_ICA  0x8000004d  Yes   No   Level 2 instruction cache accesses
PAPI_L3_ICA  0x8000004e  Yes   No   Level 3 instruction cache accesses
PAPI_L1_ICR  0x8000004f  No    No   Level 1 instruction cache reads
PAPI_L2_ICR  0x80000050  Yes   No   Level 2 instruction cache reads
PAPI_L3_ICR  0x80000051  Yes   No   Level 3 instruction cache reads
PAPI_L1_ICW  0x80000052  No    No   Level 1 instruction cache writes
PAPI_L2_ICW  0x80000053  No    No   Level 2 instruction cache writes
PAPI_L3_ICW  0x80000054  No    No   Level 3 instruction cache writes
PAPI_L1_TCH  0x80000055  No    No   Level 1 total cache hits
PAPI_L2_TCH  0x80000056  No    No   Level 2 total cache hits
PAPI_L3_TCH  0x80000057  No    No   Level 3 total cache hits
PAPI_L1_TCA  0x80000058  No    No   Level 1 total cache accesses
PAPI_L2_TCA  0x80000059  Yes   Yes  Level 2 total cache accesses
PAPI_L3_TCA  0x8000005a  Yes   No   Level 3 total cache accesses
PAPI_L1_TCR  0x8000005b  No    No   Level 1 total cache reads
PAPI_L2_TCR  0x8000005c  Yes   Yes  Level 2 total cache reads
PAPI_L3_TCR  0x8000005d  Yes   Yes  Level 3 total cache reads
PAPI_L1_TCW  0x8000005e  No    No   Level 1 total cache writes
PAPI_L2_TCW  0x8000005f  Yes   No   Level 2 total cache writes
PAPI_L3_TCW  0x80000060  Yes   No   Level 3 total cache writes
PAPI_FML_INS 0x80000061  No    No   Floating point multiply instructions
PAPI_FAD_INS 0x80000062  No    No   Floating point add instructions
PAPI_FDV_INS 0x80000063  Yes   No   Floating point divide instructions
PAPI_FSQ_INS 0x80000064  No    No   Floating point square root instructions
PAPI_FNV_INS 0x80000065  No    No   Floating point inverse instructions
PAPI_FP_OPS  0x80000066  Yes   Yes  Floating point operations
PAPI_SP_OPS  0x80000067  Yes   Yes  Floating point operations; optimized to
count scaled single precision vector operations
PAPI_DP_OPS  0x80000068  Yes   Yes  Floating point operations; optimized to
count scaled double precision vector operations
PAPI_VEC_SP  0x80000069  Yes   Yes  Single precision vector/SIMD instructions
PAPI_VEC_DP  0x8000006a  Yes   Yes  Double precision vector/SIMD instructions
PAPI_REF_CYC 0x8000006b  Yes   No   Reference clock cycles
-----------------------------------------------------------------------------
Of 108 possible events, 50 are available, of which 17 are derived.

avail.c                                        PASSED
```

Eventos escolhidos:
PAPI_L2_TCM 0x80000007 Yes No Level 2 cache misses
PAPI_L3_TCM 0x80000008 Yes No Level 3 cache misses
PAPI_BR_MSP 0x8000002e Yes No Conditional branch instructions mispredicted

Programa papiex2.c

```c
#include <papi.h>
#include <stdio.h>
#include <math.h>

#define NUM_EVENTS 3
#define TIMES 1000000

main()
{
  int i, Events[NUM_EVENTS] = {PAPI_L2_TCM,PAPI_L3_TCM,PAPI_BR_MSP};
  long_long avalues[NUM_EVENTS],bvalues[NUM_EVENTS];
  double x=0.5;

  int nc = PAPI_num_counters();
  printf("Num.Counters=%d\n",nc);
  printf("PAPI_L2_TCM,PAPI_L3_TCM,PAPI_BR_MSP\n");

/* Start counting events */
if (PAPI_start_counters(Events, NUM_EVENTS) != PAPI_OK) exit(-1);

  for (i=0; i<TIMES/4; i++) x=sin(x);

/* Read the counters */
  if (PAPI_read_counters(avalues, NUM_EVENTS) != PAPI_OK) exit(-1);

  for (i=0; i<TIMES; i++) x=sin(x);
```

```
/* Read the counters again */
  if (PAPI_read_counters(bvalues, NUM_EVENTS) != PAPI_OK) exit(-1);

/* Print counters */
  printf("Apos primeira leitura dos counters: %lld, %lld, %lld, x=%lf\n",
                 avalues[0],avalues[1],avalues[2],x);
  printf("Apos segunda leitura dos counters: %lld, %lld, %lld, x=%lf\n",
                 bvalues[0],bvalues[1],bvalues[2],x);

  printf("All done\n");
}
```

Compilando e executando

```
[_______._______@sdumont13 ~]$ module load papi
[_______._______@sdumont13 ~]$ module load papi-devel
[_______._______@sdumont13 ~]$ module load intel_psxe
[_______._______@sdumont13 ~]$ ./c papi2
... copia o executável para scratch ...
[_______._______@sdumont13 ~]$ sbatch papi2.srm
Submitted batch job 411411
```

Arquivo de lote auxiliar “c”:

```
#!/bin/sh
icc -O0 -o $1 $PAPI_INC $1.c $PAPI_LIB -lpapi -lm
```

slurm-411411.out

```
== CAP399/2019 - Lista 02 - Exerc 01 - Item (b) ==
Número total de tarefas (ntasks): 1
sdumont1121
     linux-vdso.so.1 =>  (0x00007ffd611a3000)
     libpapi.so.5 => /opt/bullxde/perftools/papi/5.5.1.0/lib64/libpapi.so.5
(0x00002ada2f189000)
     libm.so.6 => /usr/lib64/libm.so.6 (0x00002ada2f446000)
     libgcc_s.so.1 => /usr/lib64/libgcc_s.so.1 (0x00002ada2f748000)
     libc.so.6 => /usr/lib64/libc.so.6 (0x00002ada2f95e000)
     libdl.so.2 => /usr/lib64/libdl.so.2 (0x00002ada2fd2b000)
     libpfm.so.4 => /opt/bullxde/perftools/papi/5.5.1.0/lib64/libpfm.so.4
(0x00002ada2ff2f000)
     /lib64/ld-linux-x86-64.so.2 (0x00002ada2ef65000)
Num.Counters=11
PAPI_L2_TCM,PAPI_L3_TCM,PAPI_BR_MSP
Apos primeira leitura dos counters: 356, 38, 72, x=0.001549
Apos segunda leitura dos counters: 42, 1, 19, x=0.001549
All done
```

leitura contadores	PAPI_L2_TCM Level 2 cache misses	PAPI_L3_TCM Level 3 cache misses	PAPI_BR_MSP Conditional branch instructions mispredicted
primeira	356	38	72
segunda	42	1	19

Na segunda leitura temos uma quantidade muito menor de L3 misses.

EXERCÍCIO 2

Item (a)

Compilando e executando na máquina local

```
ef@TOPS:~/lista02$ mpicc -o mpi1 mpi1.c -lm
ef@TOPS:~/lista02$ mpirun -n 8 ./mpi1
[0] All done: x=0.000577
[1] All done: x=0.000408
[2] All done: x=0.000333
[3] All done: x=0.000289
[4] All done: x=0.000258
[5] All done: x=0.000236
[6] All done: x=0.000218
[7] All done: x=0.000204
```

/proc/cpuinfo (máquina local)

```
$ cat /proc/cpuinfo
processor       : 7
vendor_id       : GenuineIntel
cpu family      : 6
model           : 42
model name      : Intel(R) Core(TM) i7-2630QM CPU @ 2.00GHz
stepping        : 7
microcode       : 0x2f
cpu MHz         : 803.445
cache size      : 6144 KB
physical id     : 0
siblings        : 8
core id         : 3
cpu cores       : 4
apicid          : 7
initial apicid  : 7
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx rdtscp lm
constant_tsc arch_perfmon pebs bts nopl xtopology nonstop_tsc cpuid aperfmperf
pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 cx16 xtpr pdcm pcid sse4_1
sse4_2 x2apic popcnt tsc_deadline_timer aes xsave avx lahf_lm epb pti ssbd ibrs
ibpb stibp tpr_shadow vnmi flexpriority ept vpid xsaveopt dtherm ida arat pln
pts md_clear flush_l1d
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds
swapgs
bogomips        : 3991.22
clflush size    : 64
cache_alignment : 64
address sizes   : 36 bits physical, 48 bits virtual
```

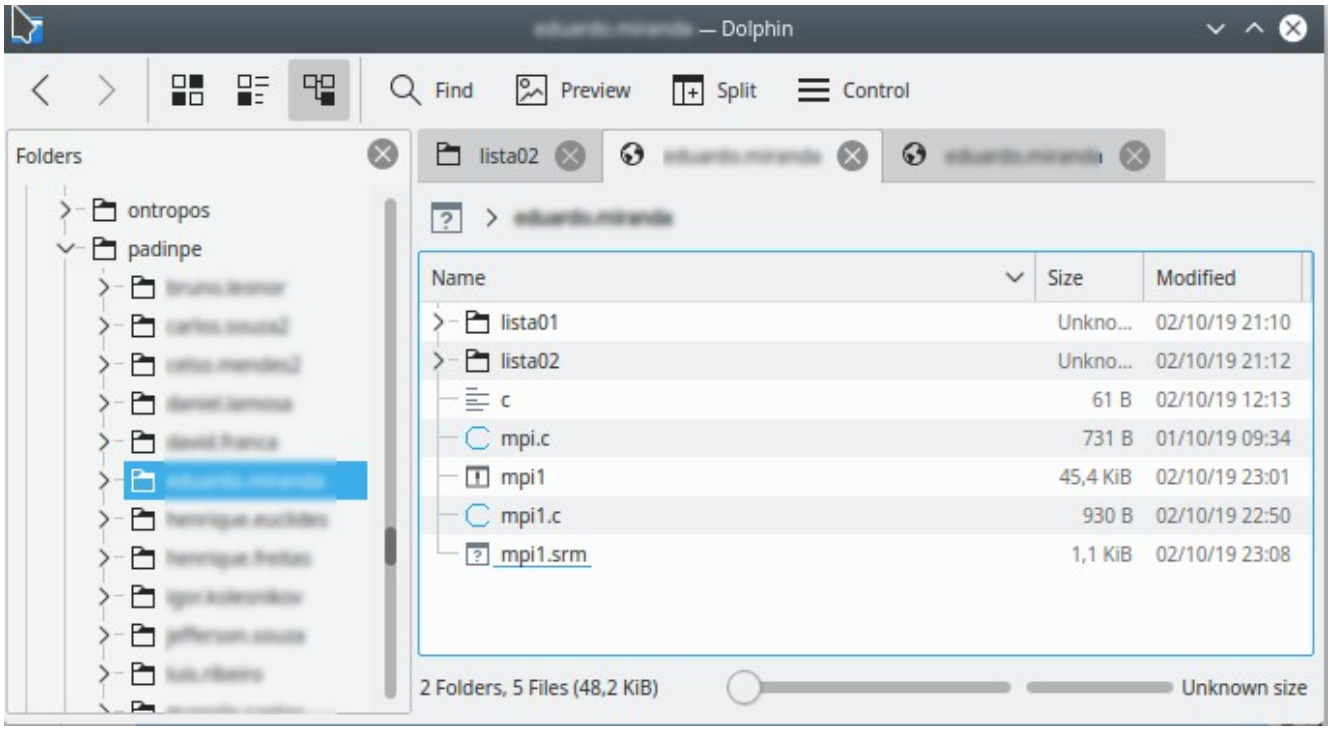
Compilando no nó de login e executando em 8 processadores:

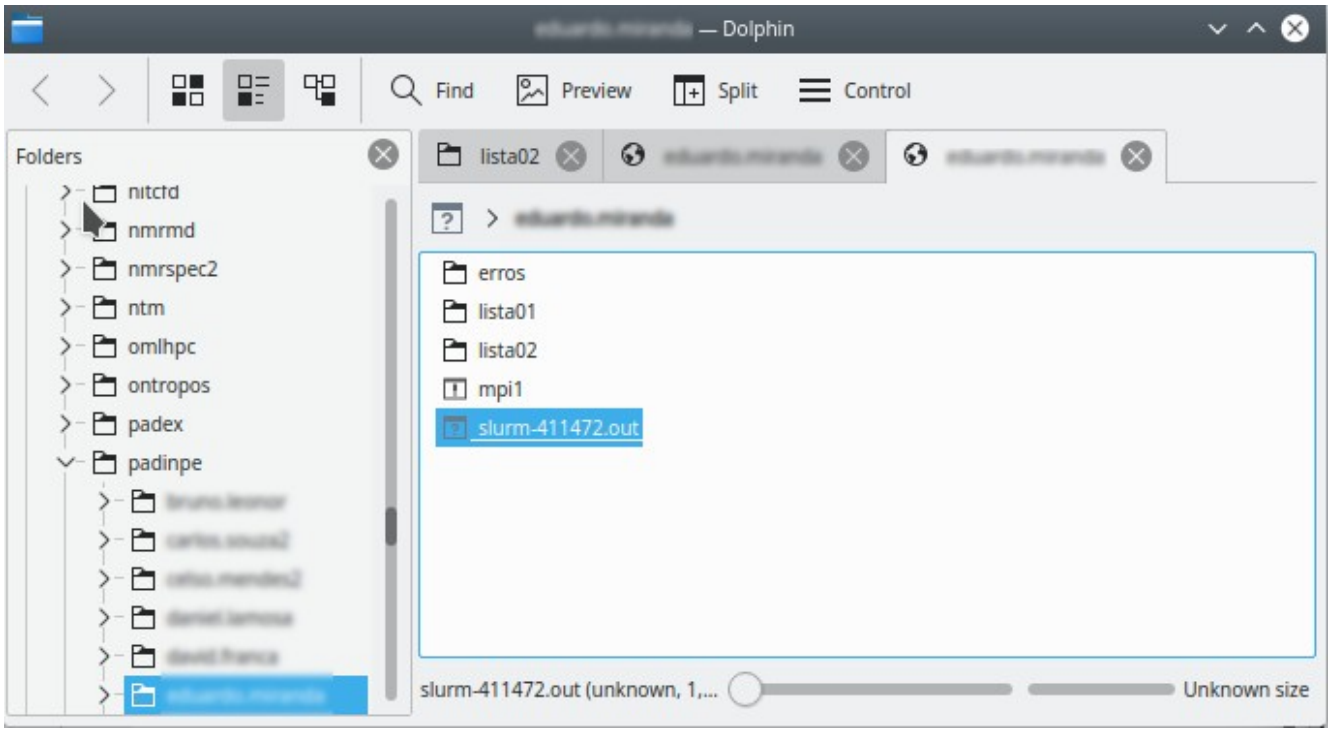
```
[_______._______@sdumont13 ~]$ module load intel_psxe/2019
[_______._______@sdumont13 ~]$ mpiicc -o mpi1 mpi1.c -lm
... copia o executável para scratch ...
[_______._______@sdumont13 ~]$ sbatch mpi1.srm
Submitted batch job 411472
```

slurm-411472.out

```
== CAP399/2019 - Lista 02 - Exerc 02 - Item (a) ==
Número total de tarefas (ntasks): 8
sdumont1069
     linux-vdso.so.1 =>  (0x00007ffc2618e000)
```

```
	libm.so.6 => /usr/lib64/libm.so.6 (0x00002abd61b97000)
	libmpifort.so.12 =>
/opt/intel/parallel_studio_xe_2019/compilers_and_libraries_2019.3.199/linux/mpi/
intel64/lib/libmpifort.so.12 (0x00002abd61e99000)
	libmpi.so.12 =>
/opt/intel/parallel_studio_xe_2019/compilers_and_libraries_2019.3.199/linux/mpi/
intel64/lib/release/libmpi.so.12 (0x00002abd62257000)
	libdl.so.2 => /usr/lib64/libdl.so.2 (0x00002abd633e5000)
	librt.so.1 => /usr/lib64/librt.so.1 (0x00002abd635e9000)
	libpthread.so.0 => /usr/lib64/libpthread.so.0 (0x00002abd637f1000)
	libgcc_s.so.1 => /usr/lib64/libgcc_s.so.1 (0x00002abd63a0d000)
	libc.so.6 => /usr/lib64/libc.so.6 (0x00002abd63c23000)
	/lib64/ld-linux-x86-64.so.2 (0x00002abd61973000)
	libfabric.so.1 =>
/opt/intel/parallel_studio_xe_2019/compilers_and_libraries_2019.3.199/linux/mpi/
intel64/libfabric/lib/libfabric.so.1 (0x00002abd63ff0000)
[0] All done: x=0.000577
[1] All done: x=0.000408
[2] All done: x=0.000333
[3] All done: x=0.000289
[4] All done: x=0.000258
[5] All done: x=0.000236
[6] All done: x=0.000218
[7] All done: x=0.000204
```





Item (b)

USANDO PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS

mpi2a.c

```
/* mpi2a.c
 * Para compilar:
 *      module load papi
 *      module load papi-devel
 *      module load intel_psxe/2019
 *      mpiicc -o mpi2a $PAPI_INC mpi2a.c $PAPI_LIB -lpapi -lm        # Intel
 *      mpicc -o mpi2a $PAPI_INC mpi2a.c $PAPI_LIB -lpapi -lm         # GNU
 * Rodar:
 *      mpirun -n 8 ./mpi2a
 */

#include "mpi.h"
#include <papi.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <math.h>

#define NUM_EVENTS 3
#define SIZE 1000000

int m;
void exit(int), do_flops(double *,int);

int main(int argc, char *argv[])
{
    int Events[NUM_EVENTS] = {PAPI_TOT_CYC,PAPI_TOT_INS,PAPI_FP_INS};
    long_long avalues[NUM_EVENTS], bvalues[NUM_EVENTS];
    int nc = PAPI_num_counters();
    printf("Num.Counters=%d\n",nc);
    printf("PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS\n");

    int myid, numprocs;
    MPI_Init(&argc, &argv);
```

```
    MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD, &myid);
    m = myid + 1 ;

    /* Initializations */
    double x = 0.5;

    /* Warmup */                /* primeira chamada */
    do_flops(&x, 4*SIZE);

    /* Start counting events */
    if (PAPI_start_counters(Events, NUM_EVENTS) != PAPI_OK) exit(-1);

    /* Do flops */              /* segunda chamada <-- instrumentar */
    do_flops(&x, SIZE);

    /* Read the counters */
    if (PAPI_read_counters(avalues, NUM_EVENTS) != PAPI_OK) exit(-1);

    /* Do many more flops */    /* terceira chamada <-- instrumentar */
    do_flops(&x, 4*SIZE);

    /* Read the counters again */
    if (PAPI_read_counters(bvalues, NUM_EVENTS) != PAPI_OK) exit(-1);

    /* Print counters */
    printf("[%d] Apos primeira leitura dos counters: %lld, %lld, %lld, x=%lf\n",
                myid, avalues[0], avalues[1], avalues[2], x);
    printf("[%d] Apos segunda leitura dos counters: %lld, %lld, %lld, x=%lf\n",
                myid, bvalues[0], bvalues[1], bvalues[2], x);

    printf("[%d] All done: x=%lf\n", myid, x);
    MPI_Finalize();
}

void do_flops(double *x, int k)
{
    int i, passes;

    for (passes=0; passes<m; passes++)
        for (i=0; i<k; i++) *x = sin(*x);
}
```

Compilando e executando

```
[_______._______@sdumont14 ~]$ module load papi
[_______._______@sdumont14 ~]$ module load papi-devel
[_______._______@sdumont14 ~]$ module load intel_psxe/2019
[_______._______@sdumont14 ~]$ mpiicc -o mpi2a $PAPI_INC mpi2a.c $PAPI_LIB -
lpapi -lm
... copia o executável para scratch ...
[_______._______@sdumont14 ~]$ sbatch mpi2a.srm
Submitted batch job 412207
```

mpi2a.srm

```
#!/bin/bash
# Uso:
#   $ sbatch mpi2a.srm           (roda em prj)(anotar o my_job_no)
#   $ squeue --job my_job_no     (verifica jobs)
#   A saída "slurm-my_job_no.out" aparece em "scracth"

#SBATCH --nodes=1               #Numero de Nós  (a)
#SBATCH --ntasks-per-node=8     #Numero de tarefas por Nó  (b)
#SBATCH --ntasks=8              #Numero total de tarefas MPI  (a x b)
#SBATCH -p cpu_dev              #Fila (partition) a ser utilizada
#SBATCH -J mpi2a                #Nome do job
#SBATCH --time=00:02:00         #Tempo limite
#SBATCH --exclusive             #Utilizacao exclusiva dos nós durante o job

echo '== CAP399/2019 - Lista 02 - Exerc 02 - Item (b) =='
echo 'Número total de tarefas (ntasks):' $SLURM_NTASKS
```

```
nodeset -e $SLURM_JOB_NODELIST
cd $SLURM_SUBMIT_DIR

#Configura
module load papi/5.5.1.0
module load papi-devel/5.5.1.0
module load intel_psxe/2019

#Configura o executavel
EXEC=/scratch/padinpe/_______._______/mpi2a

#exibe informações sobre o executável (opcional)
#echo /usr/bin/ldd $EXEC
/usr/bin/ldd $EXEC

#Dispara a execução
#   $SLURM_NTASKS : Same as --ntasks
# echo srun -n $SLURM_NTASKS $EXEC
srun -n $SLURM_NTASKS $EXEC
```

slurm-412207.out

```
== CAP399/2019 - Lista 02 - Exerc 02 - Item (b) ==
Número total de tarefas (ntasks): 8
sdumont1411
     linux-vdso.so.1 =>  (0x00007ffe443de000)
     libpapi.so.5 => /opt/bullxde/perftools/papi/5.5.1.0/lib64/libpapi.so.5
(0x00002acc8e600000)
     libm.so.6 => /usr/lib64/libm.so.6 (0x00002acc8e8bd000)
     libmpifort.so.12 =>
/opt/intel/parallel_studio_xe_2019/intelpython3/lib/libmpifort.so.12
(0x00002acc8ebbf000)
     libmpi.so.12 =>
/opt/intel/parallel_studio_xe_2019/intelpython3/lib/libmpi.so.12
(0x00002acc8ef7d000)
     libdl.so.2 => /usr/lib64/libdl.so.2 (0x00002acc9010b000)
     librt.so.1 => /usr/lib64/librt.so.1 (0x00002acc9030f000)
     libpthread.so.0 => /usr/lib64/libpthread.so.0 (0x00002acc90517000)
     libgcc_s.so.1 => /usr/lib64/libgcc_s.so.1 (0x00002acc90733000)
     libc.so.6 => /usr/lib64/libc.so.6 (0x00002acc90949000)
     libpfm.so.4 => /opt/bullxde/perftools/papi/5.5.1.0/lib64/libpfm.so.4
(0x00002acc90d16000)
     /lib64/ld-linux-x86-64.so.2 (0x00002acc8e3dc000)
     libfabric.so.1 =>
/opt/intel/parallel_studio_xe_2019/intelpython3/lib/libfabric/libfabric.so.1
(0x00002acc910f8000)
Num.Counters=11
PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS
Num.Counters=11
PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS
Num.Counters=11
PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS
Num.Counters=11
PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS
Num.Counters=11
PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS
Num.Counters=11
PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS
Num.Counters=11
PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS
Num.Counters=11
PAPI_TOT_CYC, PAPI_TOT_INS, PAPI_FP_INS
[0] Apos primeira leitura dos counters: 56966616, 87002220, 29000209, x=0.000577
[0] Apos segunda leitura dos counters: 227778954, 348000749, 116000719,
x=0.000577
[0] All done: x=0.000577
[1] Apos primeira leitura dos counters: 113927977, 174002250, 58000438,
x=0.000408
[1] Apos segunda leitura dos counters: 455712996, 696000848, 232001591,
x=0.000408
[1] All done: x=0.000408
[2] Apos primeira leitura dos counters: 170842901, 261002276, 87000681,
x=0.000333
```

```
[2] Apos segunda leitura dos counters: 683087678, 1044000949, 348002328,
x=0.000333
[2] All done: x=0.000333
[3] Apos primeira leitura dos counters: 227792494, 348002304, 116000852,
x=0.000289
[3] Apos segunda leitura dos counters: 911188166, 1392001047, 464002949,
x=0.000289
[3] All done: x=0.000289
[4] Apos primeira leitura dos counters: 284726449, 435002331, 145001042,
x=0.000258
[4] Apos segunda leitura dos counters: 1138788123, 1740001147, 580003636,
x=0.000258
[4] All done: x=0.000258
[5] Apos primeira leitura dos counters: 341687288, 522002358, 174001132,
x=0.000236
[5] Apos segunda leitura dos counters: 1366934380, 2088001247, 696004752,
x=0.000236
[5] All done: x=0.000236
[6] Apos primeira leitura dos counters: 398534905, 609002387, 203001358,
x=0.000218
[6] Apos segunda leitura dos counters: 1594282043, 2436001346, 812005313,
x=0.000218
[6] All done: x=0.000218
[7] Apos primeira leitura dos counters: 455551098, 696002415, 232001536,
x=0.000204
[7] Apos segunda leitura dos counters: 1822279002, 2784001445, 928006008,
x=0.000204
[7] All done: x=0.000204
```

8 Processadores — Unidade: Eventos x 1.000.000

Rank	PAPI_TOT_CYC Processor cycle event		PAPI_TOT_INS Instructions completed		PAPI_FP_INS Floating point instructions	
	primeira	segunda	primeira	segunda	primeira	segunda
0	57	228	87	348	29	116
1	114	456	174	696	58	232
2	171	683	261	1.044	87	348
3	228	911	348	1.392	116	464
4	285	1.139	435	1.740	145	580
5	342	1.367	522	2.088	174	696
6	399	1.594	609	2.436	203	812
7	456	1.822	696	2.784	232	928

O número de eventos aumenta para cada rank, pois cada rank MPI executa, na rotina do_flops(), uma quantidade de trabalho proporcional a “rank+1”:

```
    for (passes=0; passes<m; passes++)
        for (i=0; i<k; i++) *x = sin(*x);
```

Como curiosidade:

USANDO PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP

mpi2.c

```
/* mpi2.c
 * Para compilar:
 *      module load papi
 *      module load papi-devel
 *      module load intel_psxe/2019
 *      mpiicc -o mpi2 $PAPI_INC mpi2.c $PAPI_LIB -lpapi -lm        # Intel
 *      mpicc -o mpi2 $PAPI_INC mpi2.c $PAPI_LIB -lpapi -lm         # GNU
 * Rodar:
 *      mpirun -n 8 ./mpi2
 */

#include "mpi.h"
#include <papi.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <math.h>

#define NUM_EVENTS 3
#define SIZE 1000000

int m;
void exit(int), do_flops(double *,int);

int main(int argc, char *argv[])
{
    int Events[NUM_EVENTS] = {PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP};
    long_long avalues[NUM_EVENTS], bvalues[NUM_EVENTS];
    int nc = PAPI_num_counters();
    printf("Num.Counters=%d\n",nc);
    printf("PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP\n");

    int myid, numprocs;
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD, &myid);
    m = myid + 1 ;

    /* Initializations */
    double x = 0.5;

    /* Warmup */                /* primeira chamada */
    do_flops(&x, 4*SIZE);

    /* Start counting events */
    if (PAPI_start_counters(Events, NUM_EVENTS) != PAPI_OK) exit(-1);

    /* Do flops */              /* segunda chamada <-- instrumentar */
    do_flops(&x, SIZE);

    /* Read the counters */
    if (PAPI_read_counters(avalues, NUM_EVENTS) != PAPI_OK) exit(-1);

    /* Do many more flops */    /* terceira chamada <-- instrumentar */
    do_flops(&x, 4*SIZE);

    /* Read the counters again */
    if (PAPI_read_counters(bvalues, NUM_EVENTS) != PAPI_OK) exit(-1);

    /* Print counters */
    printf("[%d] Apos primeira leitura dos counters: %lld, %lld, %lld, x=%lf\n",
                myid, avalues[0], avalues[1], avalues[2], x);
    printf("[%d] Apos segunda leitura dos counters: %lld, %lld, %lld, x=%lf\n",
                myid, bvalues[0], bvalues[1], bvalues[2], x);
```

```
    printf("[%d] All done: x=%lf\n", myid, x);
    MPI_Finalize();
}

void do_flops(double *x, int k)
{
    int i, passes;

    for (passes=0; passes<m; passes++)
        for (i=0; i<k; i++) *x = sin(*x);
}
```

Compilando e executando

```
[________._______@sdumont13 ~]$ module load papi
[________._______@sdumont13 ~]$ module load papi-devel
[________._______@sdumont13 ~]$ module load intel_psxe/2019
[________._______@sdumont13 ~]$ mpicc -o mpi2 $PAPI_INC mpi2.c $PAPI_LIB -lpapi -
lm
... copia o executável para scratch ...
[________._______@sdumont13 ~]$ sbatch mpi2.srm
Submitted batch job 411859
```

mpi2.srm

```
#!/bin/bash
# Uso:
#   $ sbatch mpi2.srm           (roda em prj)(anotar o my_job_no)
#   $ squeue --job my_job_no    (verifica jobs)
#   A saída "slurm-my_job_no.out" aparece em "scracth"

#SBATCH --nodes=1               #Numero de Nós  (a)
#SBATCH --ntasks-per-node=8     #Numero de tarefas por Nó  (b)
#SBATCH --ntasks=8              #Numero total de tarefas MPI  (a x b)
#SBATCH -p cpu_dev              #Fila (partition) a ser utilizada
#SBATCH -J mpi2                 #Nome do job
#SBATCH --time=00:02:00         #Tempo limite
#SBATCH --exclusive             #Utilizacao exclusiva dos nós durante o job

echo '== CAP399/2019 - Lista 02 - Exerc 02 - Item (b) =='
echo 'Número total de tarefas (ntasks):' $SLURM_NTASKS

nodeset -e $SLURM_JOB_NODELIST
cd $SLURM_SUBMIT_DIR

#Configura
module load papi/5.5.1.0
module load papi-devel/5.5.1.0
module load intel_psxe/2019

#Configura o executavel
EXEC=/scratch/padinpe/________._______/mpi2

#exibe informações sobre o executável (opcional)
#echo /usr/bin/ldd $EXEC
/usr/bin/ldd $EXEC

#Dispara a execução
#   $SLURM_NTASKS : Same as --ntasks
# echo srun -n $SLURM_NTASKS $EXEC
srun -n $SLURM_NTASKS $EXEC
```

slurm-411859.out

```
== CAP399/2019 - Lista 02 - Exerc 02 - Item (b) ==
Número total de tarefas (ntasks): 8
sdumont1407
     linux-vdso.so.1 =>  (0x00007ffe171cd000)
     libpapi.so.5 => /opt/bullxde/perftools/papi/5.5.1.0/lib64/libpapi.so.5
(0x00002ad7e58ac000)
     libm.so.6 => /usr/lib64/libm.so.6 (0x00002ad7e5b69000)
```

```
     libmpifort.so.12 =>
/opt/intel/parallel_studio_xe_2019/compilers_and_libraries_2019.3.199/linux/mpi/
intel64/lib/libmpifort.so.12 (0x00002ad7e5e6b000)
     libmpi.so.12 =>
/opt/intel/parallel_studio_xe_2019/compilers_and_libraries_2019.3.199/linux/mpi/
intel64/lib/release/libmpi.so.12 (0x00002ad7e6229000)
     librt.so.1 => /usr/lib64/librt.so.1 (0x00002ad7e73b7000)
     libpthread.so.0 => /usr/lib64/libpthread.so.0 (0x00002ad7e75bf000)
     libdl.so.2 => /usr/lib64/libdl.so.2 (0x00002ad7e77db000)
     libc.so.6 => /usr/lib64/libc.so.6 (0x00002ad7e79df000)
     libpfm.so.4 => /opt/bullxde/perftools/papi/5.5.1.0/lib64/libpfm.so.4
(0x00002ad7e7dac000)
     /lib64/ld-linux-x86-64.so.2 (0x00002ad7e5688000)
     libgcc_s.so.1 => /usr/lib64/libgcc_s.so.1 (0x00002ad7e818e000)
     libfabric.so.1 =>
/opt/intel/parallel_studio_xe_2019/compilers_and_libraries_2019.3.199/linux/mpi/
intel64/libfabric/lib/libfabric.so.1 (0x00002ad7e83a4000)
Num.Counters=11
PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP
Num.Counters=11
PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP
Num.Counters=11
PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP
Num.Counters=11
PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP
Num.Counters=11
PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP
Num.Counters=11
PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP
Num.Counters=11
PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP
Num.Counters=11
PAPI_L2_TCM, PAPI_L3_TCM, PAPI_BR_MSP
[0] Apos primeira leitura dos counters: 206, 38, 72756, x=0.000577
[0] Apos segunda leitura dos counters: 26, 0, 252659, x=0.000577
[0] All done: x=0.000577
[1] Apos primeira leitura dos counters: 203, 3, 101656, x=0.000408
[1] Apos segunda leitura dos counters: 14, 0, 333379, x=0.000408
[1] All done: x=0.000408
[2] Apos primeira leitura dos counters: 220, 3, 123732, x=0.000333
[2] Apos segunda leitura dos counters: 193, 1, 422027, x=0.000333
[2] All done: x=0.000333
[3] Apos primeira leitura dos counters: 212, 3, 159647, x=0.000289
[3] Apos segunda leitura dos counters: 109, 0, 590771, x=0.000289
[3] All done: x=0.000289
[4] Apos primeira leitura dos counters: 246, 3, 169778, x=0.000258
[4] Apos segunda leitura dos counters: 39, 0, 645096, x=0.000258
[4] All done: x=0.000258
[5] Apos primeira leitura dos counters: 220, 3, 197948, x=0.000236
[5] Apos segunda leitura dos counters: 30, 0, 769883, x=0.000236
[5] All done: x=0.000236
[6] Apos primeira leitura dos counters: 214, 4, 228287, x=0.000218
[6] Apos segunda leitura dos counters: 80, 0, 890012, x=0.000218
[6] All done: x=0.000218
[7] Apos primeira leitura dos counters: 220, 3, 256829, x=0.000204
[7] Apos segunda leitura dos counters: 178, 0, 1017503, x=0.000204
[7] All done: x=0.000204
```

Mais curiosidade

Verificando qual é o modelo do processador no nó de execução

```
/* showprocessor.c
 * compilar:
 *      module load intel_psxe/2019
 *      icc -O0 -o showprocessor showprocessor.c
*/
#include <stdio.h>
#include <string.h>

int main () {
```

```
   char command[50];

   strcpy(command, "cat /proc/cpuinfo");
   system(command);

   return(0);
}
```

showprocessor.srm

```
#!/bin/bash
# Uso:
#   $ sbatch showprocessor.srm          (roda em prj)(anotar o my_job_no)
#   $ squeue --job my_job_no    (verifica jobs)
#   A saída "slurm-my_job_no.out" aparece em "scracth"

#SBATCH --nodes=1               #Numero de Nós  (a)
#SBATCH --ntasks-per-node=1     #Numero de tarefas por Nó  (b)
#SBATCH --ntasks=1              #Numero total de tarefas MPI  (a x b)
#SBATCH -p cpu_dev              #Fila (partition) a ser utilizada
#SBATCH -J pname                #Nome do job
#SBATCH --time=00:02:00         #Tempo limite
#SBATCH --exclusive             #Utilizacao exclusiva dos nós durante o job

echo '== CAP399/2019 - showprocessor =='
echo 'Número total de tarefas (ntasks):' $SLURM_NTASKS

nodeset -e $SLURM_JOB_NODELIST
cd $SLURM_SUBMIT_DIR

#Configura
module load intel_psxe/2019

#Configura o executavel
EXEC=/scratch/padinpe/_______._______/showprocessor

#exibe informações sobre o executável (opcional)
#/usr/bin/ldd $EXEC

#Dispara a execução
#   $SLURM_NTASKS : Same as --ntasks
srun -n $SLURM_NTASKS $EXEC
```

Compilando e executando

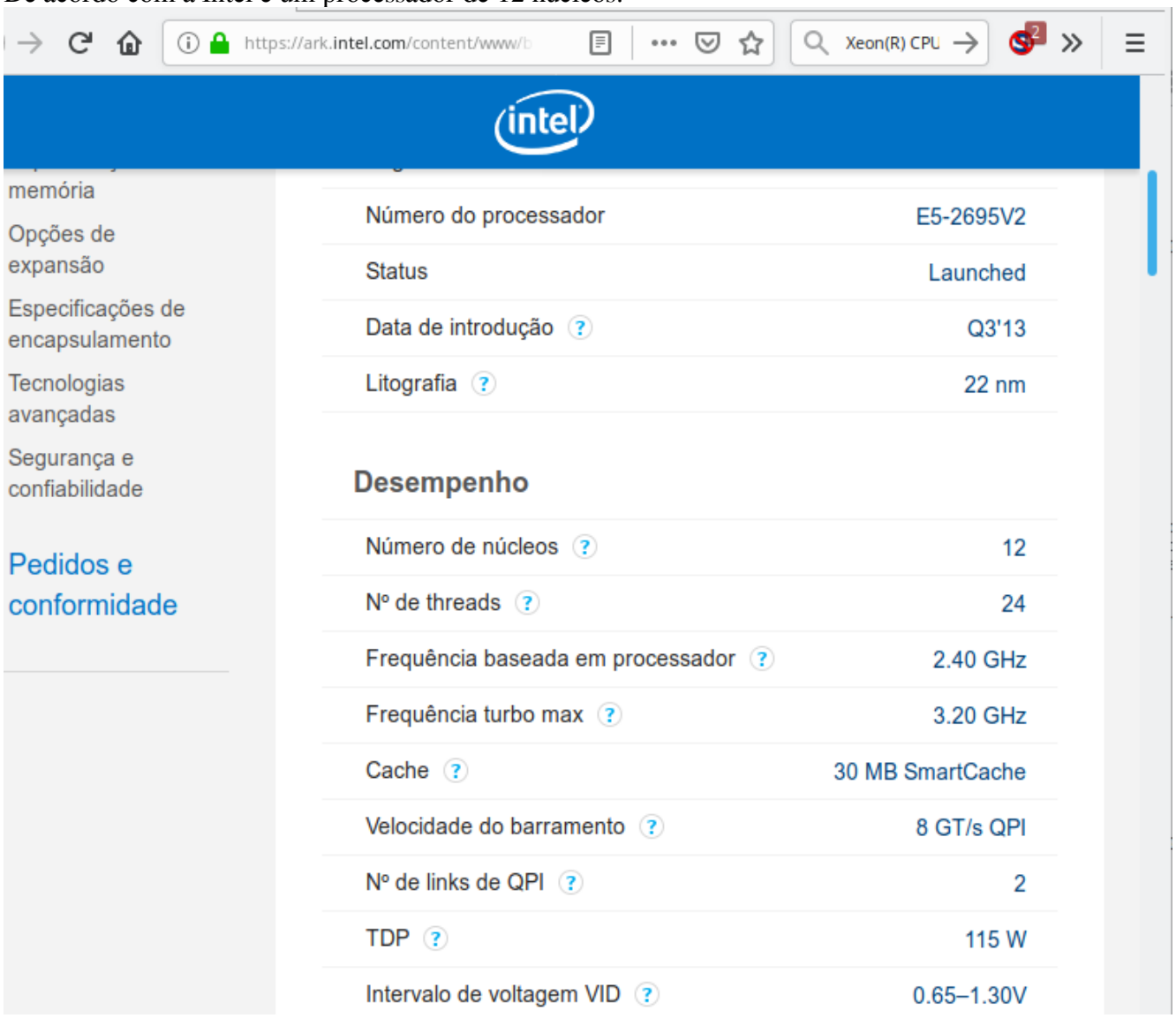
```
[_______._______@sdumont13 ~]$ module load intel_psxe/2019
[_______._______@sdumont13 ~]$ icc -O0 -o showprocessor showprocessor.c
... copia o executável para scratch ...
[_______._______@sdumont13 ~]$ sbatch showprocessor.srm
Submitted batch job 411387
```

slurm-411387.out (mostrando só o primeiro processador)

```
processor  : 0
vendor_id  : GenuineIntel
cpu family : 6
model            : 62
model name : Intel(R) Xeon(R) CPU E5-2695 v2 @ 2.40GHz
stepping   : 4
microcode  : 0x42d
cpu MHz    : 2560.400
cache size : 30720 KB
physical id      : 0
siblings   : 12
core id    : 0
cpu cores  : 12
apicid     : 0
initial apicid: 0
fpu        : yes
```

```
fpu_exception: yes
cpuid level      : 13
wp         : yes
flags            : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb
rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc
aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3
cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic popcnt tsc_deadline_timer aes xsave
avx f16c rdrand lahf_lm epb ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority
ept vpid fsgsbase smep erms xsaveopt dtherm ida arat pln pts spec_ctrl
intel_stibp flush_l1d
bogomips   : 4800.32
clflush size     : 64
cache_alignment  : 64
address sizes    : 46 bits physical, 48 bits virtual
power management:
```

De acordo com a Intel é um processador de 12 núcleos:



É o mesmo modelo de processador do nó de login:

```
processor        : 23
vendor_id        : GenuineIntel
cpu family       : 6
model            : 62
model name       : Intel(R) Xeon(R) CPU E5-2695 v2 @ 2.40GHz
stepping         : 4
microcode        : 0x42d
cpu MHz          : 2890.722
cache size       : 30720 KB
```

```
physical id     : 1
siblings        : 12
core id         : 13
cpu cores       : 12
apicid          : 58
initial apicid  : 58
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb
rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc
aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3
cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic popcnt tsc_deadline_timer aes xsave
avx f16c rdrand lahf_lm epb ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority
ept vpid fsgsbase smep erms xsaveopt dtherm ida arat pln pts spec_ctrl
intel_stibp flush_l1d
bogomips        : 4806.50
clflush size    : 64
cache_alignment : 64
address sizes   : 46 bits physical, 48 bits virtual
power management:
```

REFERÊNCIAS

- http://www.lac.inpe.br/~_____/cap399-2019/
- http://www.lac.inpe.br/~stephan/
- https://sdumont.lncc.br/support_manual.php
- https://slurm.schedmd.com/documentation.html
- https://stackoverflow.com
- https://software.intel.com/en-us/cpp-compiler-developer-guide-and-reference-compiler-reference
- https://icl.cs.utk.edu/projects/papi
- https://www.arc.vt.edu/userguide/papi/
- https://docs.hpc.qmul.ac.uk/using/UsingModules/