Exclusive Mode 파티션에서 Multi-Process 작업 수행 가이드

본 문서는 `olaf_c_core` 파티션의 작업 포화 상태를 해결하고, `normal_cpu`, `long_cpu`와 같은 Exclusive Mode 파티션에서 Multi-Process 작업을 수행할 수 있는 방법을 안내합니다.

1. Exclusive Mode란?

`Exclusive Mode`는 작업 제출 시 **하나의 노드에 하나의 작업만 실행**되도록 설정된 모드입니다.

- **특징:** 단일 작업이 노드의 모든 자원을 독점적으로 사용.
- **장점:** 작업 간 간섭 없이 안정성과 성능 최적화 보장.
- **단점:** 단일 코어 작업의 경우 자원 낭비 가능.

2. Multi-Process 작업의 필요성

- `olaf_c_core` 파티션의 작업 대기 시간이 길어짐에 따라, 독점 모드(Exclusive Mode)가 적용된 파티션을 활용하여 Multi-Process 작업을 수행하면 자원의 효율성을 극대화하고 대기 시간을 단축할 수 있습니다.
- Multi-Process 작업을 통해 Exclusive Mode의 자원 독점을 최대한 활용할 수 있습니다.

3. Multi-Process 작업 수행 스크립트

3-1. Serial Multi job 예시 (MPI, OpenMP x)

SHELL

```
#!/bin/bash
#SBATCH -J Serial_multi_job
#SBATCH -N 1
#SBATCH --ntasks=14
#SBATCH -p normal_cpu

source ~/.bashrc
module load gcc openmpi

export PROG=[path to binary file]

for ps_num in $(seq -w 1 ${SLUMR_NTASKS})
    do
            $PROG [input] > ${SLURM_JOB_ID}_$ps_num.out &
    done

wait
```

※ 1개의 노드에서 14개의 Serial process를 백그라운드로 실행

Required arguments

- `#SBATCH --ntasks` : 프로세스수 지정
- `PROG`: 실행시킬 프로그램의 경로를 입력
  ※ Log_num 지정 없을시 하나의 output, log만 출력하게됨

3-2. MPI multi job 예시 (MPI, OpenMP O)

```shell
#!/bin/bash
#SBATCH -J multi_job
#SBATCH -p normal_cpu
#SBATCH -N 2
#SBATCH --ntasks-per-node=7
#SBATCH --cpus-per-task=10

source ~/.bashrc
module load gcc openmpi

export I_MPI_PMI_LIBRARY=/usr/lib64/libpmi.so
export PROG=[path to binary file]

run_Prog() {
        srun -n 1 --exclusive=user --mem-per-cpu=10gb \
         $PROG $1 > ${SLURM_JOB_ID}_$2.out &
}

for ps_num in $(seq 1 ${SLURM_NTASKS})
    do
        run_Prog [input] $ps_num
    done

wait
```

※ 2개의 노드에서 10개의 thread로 동작하는 7개의 process(--ntasks-per-node)를 노드마다 실행

Required arguments

- `#SBATCH -N` : 노드 수 지정
- `#SBATCH --ntasks-per-node` : 노드별 수행될 프로세스 수 지정
- `#SBATCH --cpus-per-task` : 프로세스당 사용될 Thread 수
- `PROG`: 실행시킬 프로그램의 경로를 입력
- `ps_num` : 프로세스별로 생성되는 output, log를 지정하기 위한 변수
  *※ Log_num 지정 없을시 하나의 output, log만 출력하게됨

4. 권장 사항

1. **스크립트 내용 수정** : 자신의 프로그램에 맞게 스크립트 내용 수정하여 작업 제출
2. **작업 자원 확인** : 작업에 필요한 CPU 코어와 메모리를 정확히 계산하여 요구량을 설정
3. **병렬화 지원 확인:** MPI multi job의 경우 실행할 프로그램이 병렬 처리를 지원하도록 구성되어 있는지 확인 필요