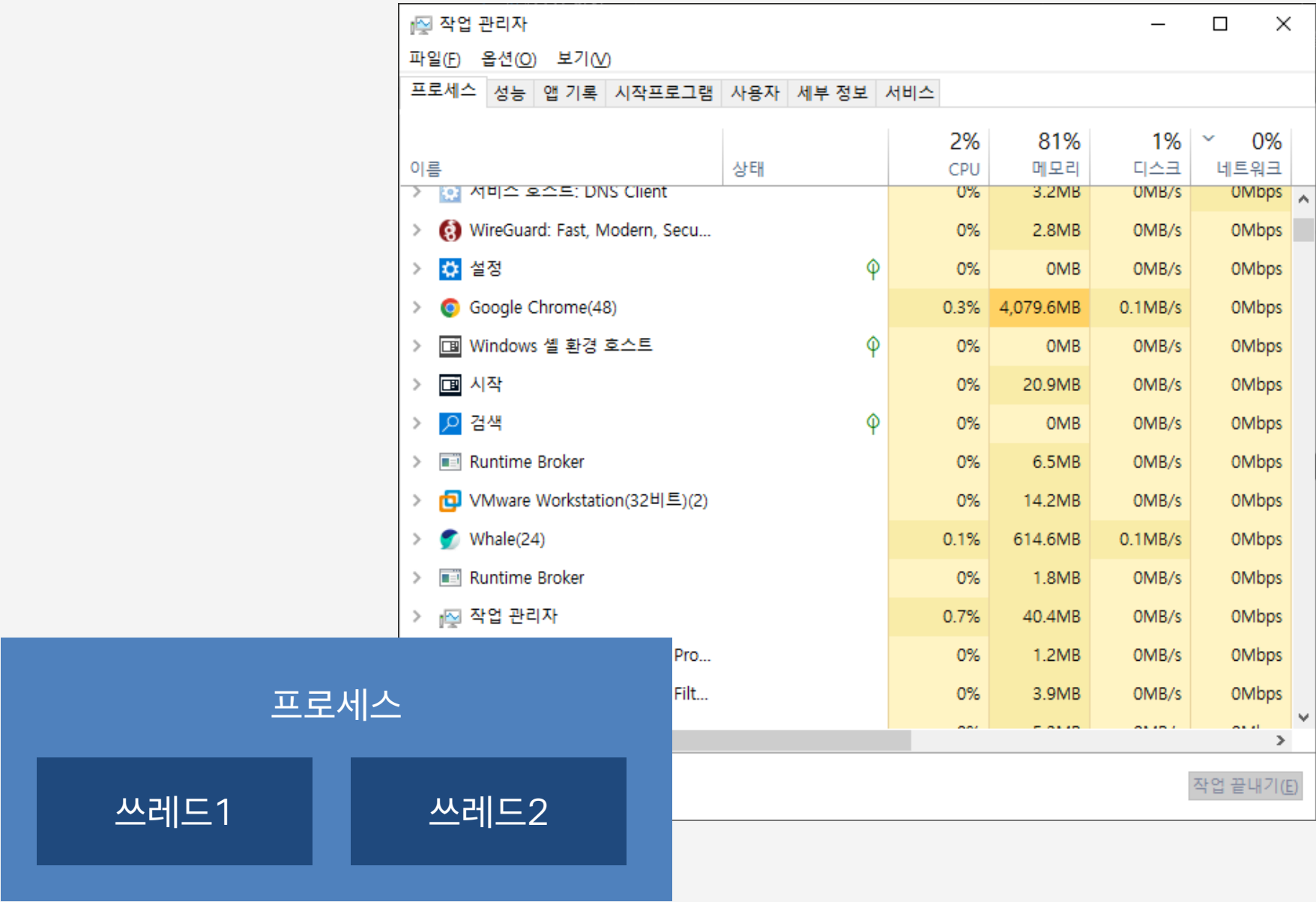

Python 병렬처리

구준한

- 프로세스: 실행 중인 프로그램
- 쓰레드: 프로세스 내 실제 작업을 수행



- 인터프리터 언어로 싱글 쓰레드에서 순차적으로 동작하는 것이 기본

=> 병렬 처리를 위해선 별도의 모듈 필요

```
from multiprocessing import Pool

def f(x):
    return x*x

if __name__ == '__main__':
    p = Pool(4)
    result = p.map(f, [1,2,3,4])
    p.close()
    print(result)
```

```
import os
from multiprocessing import Process

def f(x):
    print(x*x)

if __name__ == '__main__':
    numbers = [1, 2, 3, 4]

    proc1 = Process(target=f, args =(numbers[0],))
    proc1.start()
    proc2 = Process(target=f, args =(numbers[1],))
    proc2.start()
    proc3 = Process(target=f, args =(numbers[2],))
    proc3.start()
    proc4 = Process(target=f, args =(numbers[3],))
    proc4.start()

    proc1.join()
    proc2.join()
    proc3.join()
    proc4.join()
```

- **Pool:** 처리할 일들을 pool에 뿌려 놓고 알아서 병렬 처리
- **Process:** 각 프로세스별로 할당량을 명시적으로 적어서 처리

But 실제 둘의 차이는 조금 더 복잡

- **쓰레드:** 가벼우나, 파이썬의 GIL 정책으로 I/O 처리시 효과적
- **프로세스:** 각 고유한 메모리 영역을 가지기 때문에 처음 프로세스 생성 시 시간이 소요 및 메모리를 조금 더 필요

병렬적인 CPU 작업 가능하여 빠름

```
(base) root@Junhan-Jupyter:/home/cssa/나라장터_물품상세내역_18012205# ls
물품계약상세내역_2018년01월.csv 물품계약상세내역_2019년07월.csv 물품계약상세내역_2021년01월.csv
물품계약상세내역_2018년02월.csv 물품계약상세내역_2019년08월.csv 물품계약상세내역_2021년02월.csv
물품계약상세내역_2018년03월.csv 물품계약상세내역_2019년09월.csv 물품계약상세내역_2021년03월.csv
물품계약상세내역_2018년04월.csv 물품계약상세내역_2019년10월.csv 물품계약상세내역_2021년04월.csv
물품계약상세내역_2018년05월.csv 물품계약상세내역_2019년11월.csv 물품계약상세내역_2021년05월.csv
물품계약상세내역_2018년06월.csv 물품계약상세내역_2019년12월.csv 물품계약상세내역_2021년06월.csv
물품계약상세내역_2018년07월.csv 물품계약상세내역_2020년01월.csv 물품계약상세내역_2021년07월.csv
물품계약상세내역_2018년08월.csv 물품계약상세내역_2020년02월.csv 물품계약상세내역_2021년08월.csv
물품계약상세내역_2018년09월.csv 물품계약상세내역_2020년03월.csv 물품계약상세내역_2021년09월.csv
물품계약상세내역_2018년10월.csv 물품계약상세내역_2020년04월.csv 물품계약상세내역_2021년10월.csv
물품계약상세내역_2018년11월.csv 물품계약상세내역_2020년05월.csv 물품계약상세내역_2021년11월.csv
물품계약상세내역_2018년12월.csv 물품계약상세내역_2020년06월.csv 물품계약상세내역_2021년12월.csv
물품계약상세내역_2019년01월.csv 물품계약상세내역_2020년07월.csv 물품계약상세내역_2022년01월.csv
물품계약상세내역_2019년02월.csv 물품계약상세내역_2020년08월.csv 물품계약상세내역_2022년02월.csv
물품계약상세내역_2019년03월.csv 물품계약상세내역_2020년09월.csv 물품계약상세내역_2022년03월.csv
물품계약상세내역_2019년04월.csv 물품계약상세내역_2020년10월.csv 물품계약상세내역_2022년04월.csv
물품계약상세내역_2019년05월.csv 물품계약상세내역_2020년11월.csv 물품계약상세내역_2022년05월.csv
물품계약상세내역_2019년06월.csv 물품계약상세내역_2020년12월.csv
(base) root@Junhan-Jupyter:/home/cssa/나라장터_물품상세내역_18012205# du -sh ../나라장터_물품상세내역_18012205
5.4G    ../나라장터_물품상세내역_18012205
```



```
import pandas as pd
import os
import glob
import multiprocessing as mp

# core value required when run this script
global num_cores
num_cores = mp.cpu_count()

def convert_csv(excel):
    df = pd.read_excel(excel) # if only the first sheet is needed.
    df.to_csv(f'{excel}.csv')

def main():
    p = mp.Pool(num_cores)

    # assume the path
    excel_files = glob.glob('*.xlsx')

    p.map(convert_csv, excel_files)

    p.close()
    p.join()

    exit()
```

root@Junhan-Jupyter: /home/cssa

File Edit View Options Transfer Script Tools Window Help

Enter host <Alt+R>

root@Junhan-Jupyter: /home/cssa/xlsx_convert root@Junhan-Jupyter: /home/cssa/xlsx_convert root@Junhan-Jupyter: /home/cssa

Junhan-Jupyter (Ubuntu 20.04 64bit / Linux 5.4.0-125-generic) - IP 192.168.1.185/24 Pub Uptime: 1:20:58

CPU [|||||100.0%] CPU - 100.0% nice: 0.0% ctx_sw: 946 MEM - 25.0% active: 10.0G SWAP - 0.0% LOAD 16-core
MEM [||| 25.0%] user: 99.6% irq: 0.0% inter: 4286 total: 39.2G inactive: 2.65G total: 4.00G 1 min: 16.52
SWAP [0.0%] system: 0.4% iowait: 0.0% sw_int: 4739 used: 9.01G buffers: 80.0M used: 4.00G 5 min: 14.53
idle: 0.0% steal: 0.0% free: 29.4G cached: 3.75G free: 4.00G 15 min: 8.24

NETWORK Rx/s Tx/s
apt-kakao 0b 0b
docker0 0b 0b
ens160 2Kb 29Kb
lo 0b 0b
vethadff92 0b 0b

DefaultGateway 36ms

DISK I/O R/s W/s
dm-0 0 0
sda 0 0
sda1 0 0
sda2 0 0
sda3 0 0
sdb 0 0
sdb1 0 0
sr0 0 0

FILE SYS Used Total
/ 37.3G 77.0G
/app (sdb1) 28.2G 78.4G

CONTAINERS 1 (served by Docker 20.10.17)

Name	Status	CPU%	MEM	/MAX	IOR/s	IOW/s	Rx/s	Tx/s	Command
all-notebook	running	1.5	180M	39.2G	0B	0B	0b	0b	tini -g --

TASKS 358 (489 thr), 17 run, 205 slp, 136 oth sorted automatically by CPU consumption

CPUPX	MEM%	VIRT	RES	PID	USER	TIME+	THR	NI	S	R/s	W/s	Command
100.1	1.9	1.04G	745M	17764	root	10:38 1	0	R	0	0	0	python convert.py
100.0	1.7	1.00G	702M	17773	root	10:40 1	0	R	0	0	0	python convert.py
100.0	1.6	985M	659M	17768	root	10:42 1	0	R	0	0	0	python convert.py
100.0	1.6	974M	648M	17775	root	10:42 1	0	R	0	0	0	python convert.py
100.0	0.2	422M	99.5M	17769	root	10:38 1	0	R	0	0	0	python convert.py
100.0	0.2	422M	99.4M	17771	root	10:41 1	0	R	0	0	0	python convert.py
99.7	1.9	1.06G	759M	17770	root	10:40 1	0	R	0	0	0	python convert.py
99.7	1.8	1.04G	737M	17772	root	10:38 1	0	R	0	0	0	python convert.py
99.7	1.8	1.03G	729M	17776	root	10:38 1	0	R	0	0	0	python convert.py
99.7	1.8	1.02G	714M	17767	root	10:41 1	0	R	0	0	0	python convert.py
99.7	1.8	1.01G	710M	17779	root	10:39 1	0	R	0	0	0	python convert.py
99.5	1.8	1.02G	722M	17765	root	10:41 1	0	R	0	0	0	python convert.py
99.4	0.5	517M	195M	17777	root	10:41 1	0	R	0	0	0	python convert.py
99.1	1.9	1.05G	747M	17774	root	10:38 1	0	R	0	0	0	python convert.py
96.5	1.5	943M	618M	17778	root	10:39 1	0	R	0	0	0	python convert.py
95.7	1.8	1.04G	739M	17766	root	10:41 1	0	R	0	0	0	python convert.py
9.7	0.2	906M	82.8M	5826	root	3:07 2	0	R	0	0	0	/usr/bin/python3 /usr/bin/gla
0.7	0.0	231M	7.66M	986	root	0:02 4	0	S	0	0	0	/usr/bin/vmtoolsd
0.3	0.2	2.24G	88.8M	1254	root	0:18 22	0	S	0	0	0	/usr/bin/dockerd -H fd:// --c
0.3	0.1	2.21G	55.7M	1115	root	0:18 22	0	S	0	0	0	/usr/bin/containerd
0.3	0.0	696M	10.6M	1438	root	0:14 13	0	S	0	0	0	/usr/bin/containerd-shim-runc
0.3	0.0	0	0	213	root	0:01 1	0	I	0	0	0	[kworker/2:1-events]
0.0	0.2	583M	96.4M	17747	root	0:01 12	0	S	0	0	0	python convert.py
0.0	0.2	313M	88.6M	1532	cssa	0:07 2	0	S	0	0	0	python3.9 /opt/conda/bin/jupy
0.0	0.1	880M	57.0M	1083	root	0:04 2	0	S	0	0	0	/usr/bin/python3 /usr/bin/gla
0.0	0.1	1.89G	38.1M	1098	root	0:02 26	0	S	0	0	0	/usr/lib/snapd/snapd
0.0	0.1	105M	20.5M	1156	root	0:00 2	0	S	0	0	0	/usr/bin/python3 /usr/share/u
0.0	0.0	74.9M	19.2M	644	root	0:01 1	-1	S	0	41K	0	/lib/systemd/systemd-journald
0.0	0.0	29.1M	18.2M	1091	root	0:00 1	0	S	0	0	0	/usr/bin/python3 /usr/bin/net
0.0	0.0	402M	17.6M	923	root	0:02 7	0	S	0	0	0	/sbin/multipathd -d -s
0.0	0.0	386M	13.7M	1110	root	0:00 5	0	S	0	0	0	/usr/lib/udisks2/udisksd
0.0	0.0	311M	13.2M	1138	root	0:00 3	0	S	0	0	0	/usr/sbin/ModemManager
0.0	0.0	24.0M	11.8M	1057	systemd-r	0:00 1	0	S	0	0	0	/lib/systemd/systemd-resolved
0.0	0.0	164M	11.4M	1	root	0:02 1	0	S	0	0	0	/sbin/init maybe-ubiquity
0.0	0.0	46.4M	10.4M	982	root	0:00 1	0	S	0	0	0	/usr/bin/VGAAuthService
0.0	0.0	18.7M	9.46M	3708	cssa	0:01 1	0	S	0	0	0	/lib/systemd/systemd --user
0.0	0.0	234M	9.18M	1071	root	0:00 3	0	S	0	0	0	/usr/lib/accountsservice/acco
0.0	0.0	231M	8.77M	1092	root	0:00 3	0	S	0	0	0	/usr/lib/policykit-1/polkitd
0.0	0.0	13.6M	8.70M	16293	root	0:00 1	0	S	0	0	0	sshd: cssa [priv]
0.0	0.0	13.6M	8.68M	3694	root	0:00 1	0	S	0	0	0	sshd: cssa [priv]
0.0	0.0	17.0M	7.73M	1103	root	0:00 1	0	S	0	0	0	/lib/systemd/systemd-logind

High CPU user mode

2022-09-08 08:27:58 UTC 2022-09-08 08:17:15 (ongoing) - CPU_USER (Min:80.4 Mean:99.5 Max:99.8): python, python, python

JupyterLab idrac - iDRAC8 - Tempera

주의 요함 | https://192.

ShortKey


모든 환경에서 검색

Administrator@VSPHERE.LOCAL

192.168.1.173

작업

요약 모니터 구성 사용 권한 VM 리소스 풀 데이터스토어 네트워크 업데이트



하이퍼바이저: VMware ESXi, 7.0.3, 19482537
모델: PowerEdge R430
프로세서 유형: Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz
논리 프로세서: 16
NIC: 4
가상 시스템: 6
상태: 연결됨
가동 시간: 66일

CPU 사용 가능: 1 MHz
사용됨: 19.18 GHz
메모리 사용 가능: 31.42 GB
사용됨: 48.49 GB
스토리지 사용 가능: 1.55 TB
사용됨: 263.67 GB

호스트 CPU 사용량

확인 녹색으로 재설정

하드웨어

구성

관련 개체

없음

제조업체	Dell Inc.
모델	PowerEdge R430
CPU	8개 CPU x 2.4 GHz
메모리	48.49 GB/79.91 GB
가상 플래시 리소스	3.56 GB/7.75 GB
네트워킹	hnuce-h02.local
스토리지	2개 데이터스토어

상태

세부 정보

이니시에이터

대기 시간

시작 시간

완료 시간

서버

Thank You
