olmv1-metrics

July 18, 2024

1 OLM v1 metrics

Capture metrics for both `catalogd` and `operator-controller` pods

```
[1]: import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
```

```
[2]: df1 = pd.read_csv('work/olm_v1_metrics-20240717-1606.csv', index_col=0)
```

```
[3]: df1.loc[:6]
```

```
[3]:                                                pod       c0     cpu0  \
     0       catalogd-controller-manager-7f9766cf5d-cx4gh  manager  888559n
     1  operator-controller-controller-manager-9676486…  manager  877422n
     2       catalogd-controller-manager-7f9766cf5d-cx4gh  manager  888559n
     3  operator-controller-controller-manager-9676486…  manager  877422n
     4       catalogd-controller-manager-7f9766cf5d-cx4gh  manager  888559n
     5  operator-controller-controller-manager-9676486…  manager  877422n
     6       catalogd-controller-manager-7f9766cf5d-cx4gh  manager  888559n

          mem0               c1     cpu1     mem1
     0  25576Ki  kube-rbac-proxy  35905n   9252Ki
     1  16572Ki  kube-rbac-proxy       0  10896Ki
     2  25576Ki  kube-rbac-proxy  35905n   9252Ki
     3  16572Ki  kube-rbac-proxy       0  10896Ki
     4  25576Ki  kube-rbac-proxy  35905n   9252Ki
     5  16572Ki  kube-rbac-proxy       0  10896Ki
     6  25576Ki  kube-rbac-proxy  35905n   9252Ki
```

1.1 Operator Controller metrics

```
[4]: df = df1.loc[df1['pod'].str.
      ↪contains('operator-controller'),['cpu0','cpu1','mem0','mem1']]
```

```
[5]: cputotal = np.ceil((df.cpu0.str.rstrip('n').astype(int) + df.cpu1.str.
      ↪rstrip('n').astype('int')) / 1000000).astype(int)
```

[6]:
```
df.insert(4, "cputotal", cputotal, True)
```

[7]:
```
memtotal = np.round((df.mem0.str.rstrip('Ki').astype(int) + df.mem1.str.
↪rstrip('Ki').astype(int)) / 1024).astype(int)
```

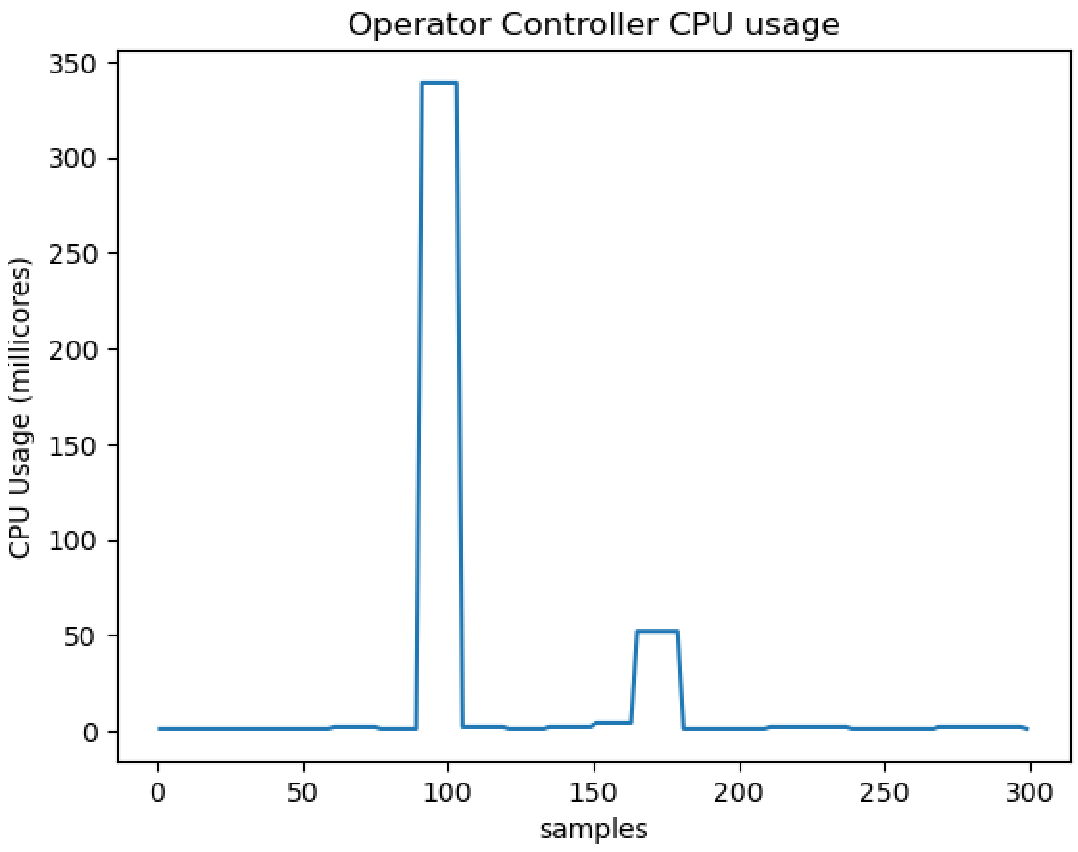
[8]:
```
df.insert(5, "memtotal", memtotal, True)
```

[9]:
```
df
```

[9]:
```
         cpu0   cpu1     mem0     mem1  cputotal  memtotal
1     877422n      0  16572Ki  10896Ki         1        27
3     877422n      0  16572Ki  10896Ki         1        27
5     877422n      0  16572Ki  10896Ki         1        27
7     877422n      0  16572Ki  10896Ki         1        27
9     877422n      0  16572Ki  10896Ki         1        27
..        …      …       …       …        …         …
291  1916366n  3694n  42280Ki   8772Ki         2        50
293  1916366n  3694n  42280Ki   8772Ki         2        50
295  1916366n  3694n  42280Ki   8772Ki         2        50
297  1916366n  3694n  42280Ki   8772Ki         2        50
299   888907n      0  42280Ki   8772Ki         1        50

[150 rows x 6 columns]
```

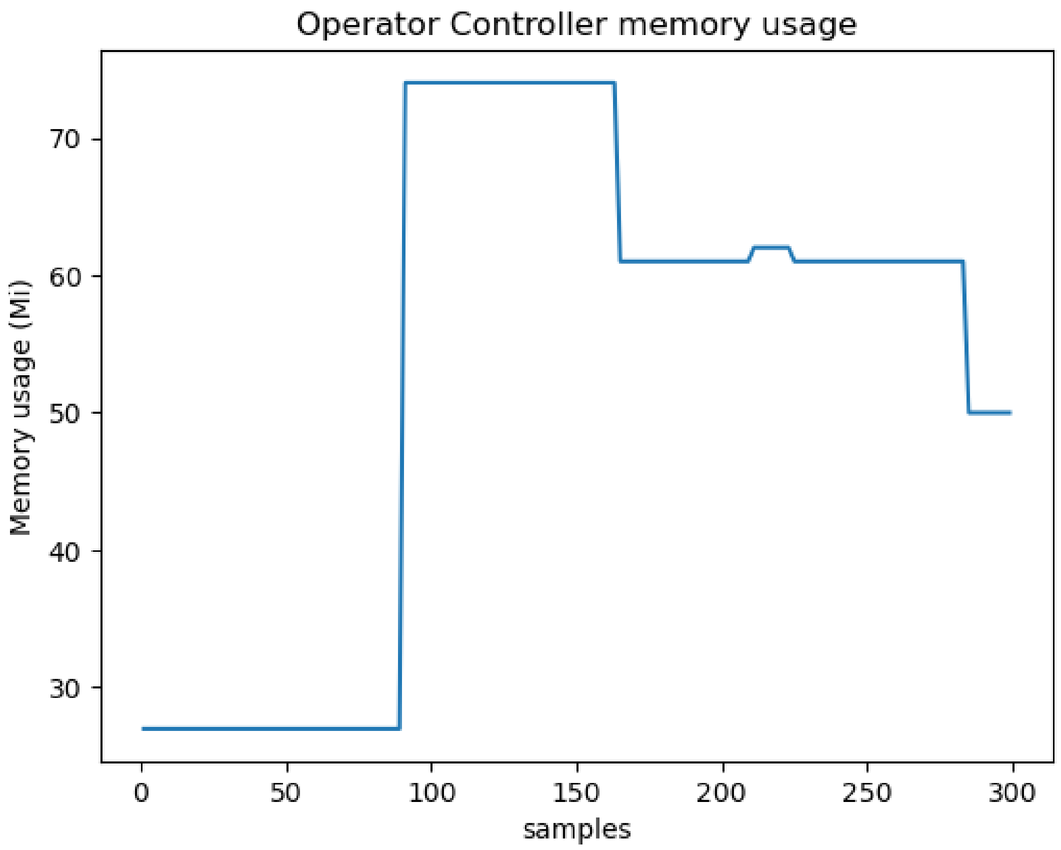
[10]:
```
df.cputotal.plot(title='Operator Controller CPU usage', xlabel='samples',␣
↪ylabel='CPU Usage (millicores)', kind='line')
```

[10]:
```
<Axes: title={'center': 'Operator Controller CPU usage'}, xlabel='samples',
ylabel='CPU Usage (millicores)'>
```



```
[11]: df.memtotal.plot(title='Operator Controller memory usage', kind='line',␣
      ↪xlabel='samples', ylabel='Memory usage (Mi)')
```

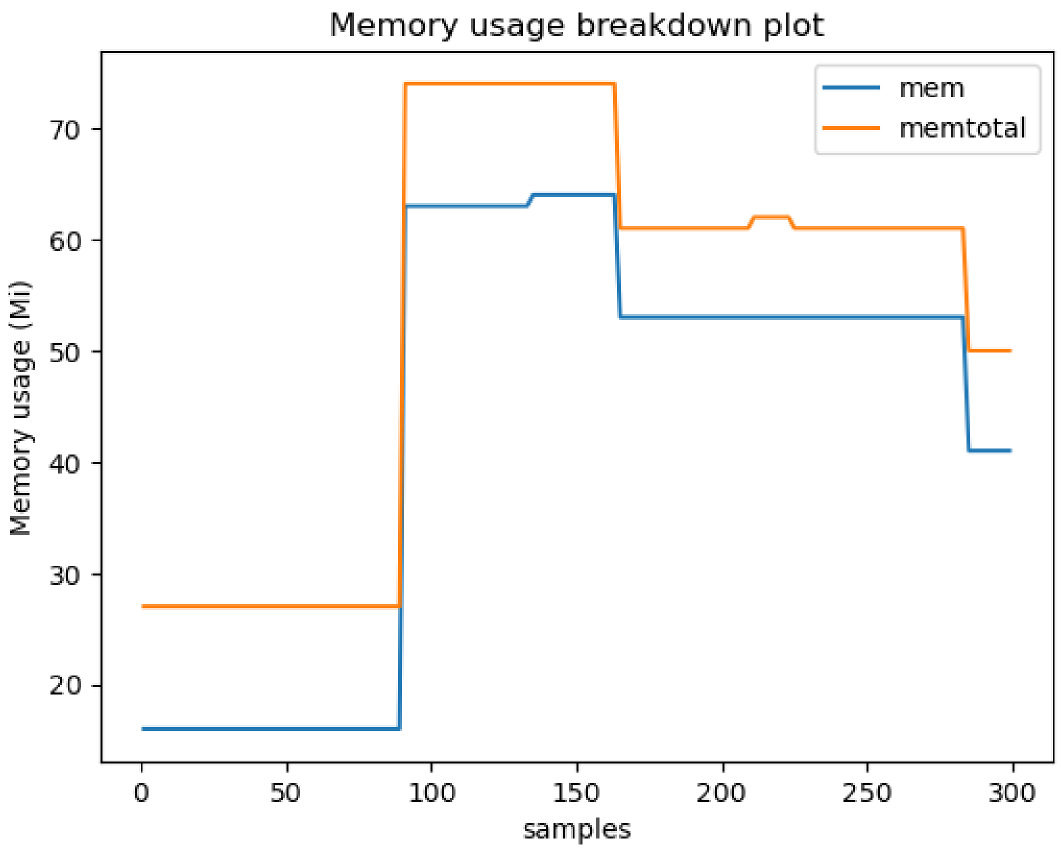
```
[11]: <Axes: title={'center': 'Operator Controller memory usage'}, xlabel='samples',
      ylabel='Memory usage (Mi)'>
```



```
[12]: df.columns

[12]: Index(['cpu0', 'cpu1', 'mem0', 'mem1', 'cputotal', 'memtotal'], dtype='object')

[13]: cpu = np.ceil(df.cpu0.str.rstrip('n').astype(int) / 1000000).astype(int)

[14]: df.insert(6, "cpu", cpu, True)

[15]: mem = np.round(df.mem0.str.rstrip('Ki').astype(int) / 1024).astype(int)

[16]: df.insert(7, "mem", mem, True)

[17]: df[:5]

[17]:       cpu0 cpu1     mem0     mem1  cputotal  memtotal  cpu  mem
      1  877422n    0  16572Ki  10896Ki         1        27    1   16
      3  877422n    0  16572Ki  10896Ki         1        27    1   16
      5  877422n    0  16572Ki  10896Ki         1        27    1   16
      7  877422n    0  16572Ki  10896Ki         1        27    1   16
      9  877422n    0  16572Ki  10896Ki         1        27    1   16
```

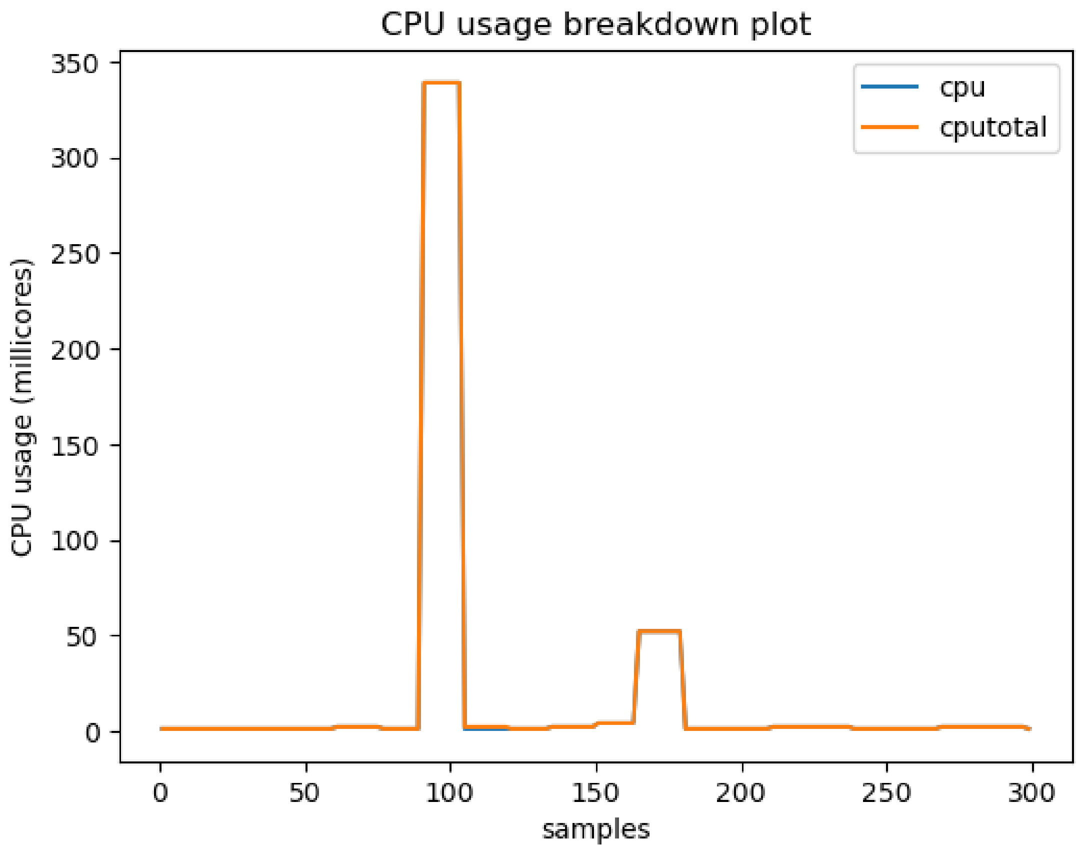
```
[18]: df.plot(y=['mem','memtotal'], title='Memory usage breakdown plot', kind='line',␣
        ↪xlabel='samples', ylabel='Memory usage (Mi)')
```

```
[18]: <Axes: title={'center': 'Memory usage breakdown plot'}, xlabel='samples',
      ylabel='Memory usage (Mi)'>
```



```
[19]: df.plot(y=['cpu','cputotal'], title='CPU usage breakdown plot', kind='line',␣
        ↪xlabel='samples', ylabel='CPU usage (millicores)')
```

```
[19]: <Axes: title={'center': 'CPU usage breakdown plot'}, xlabel='samples',
      ylabel='CPU usage (millicores)'>
```



1.2 Catalogd metrics

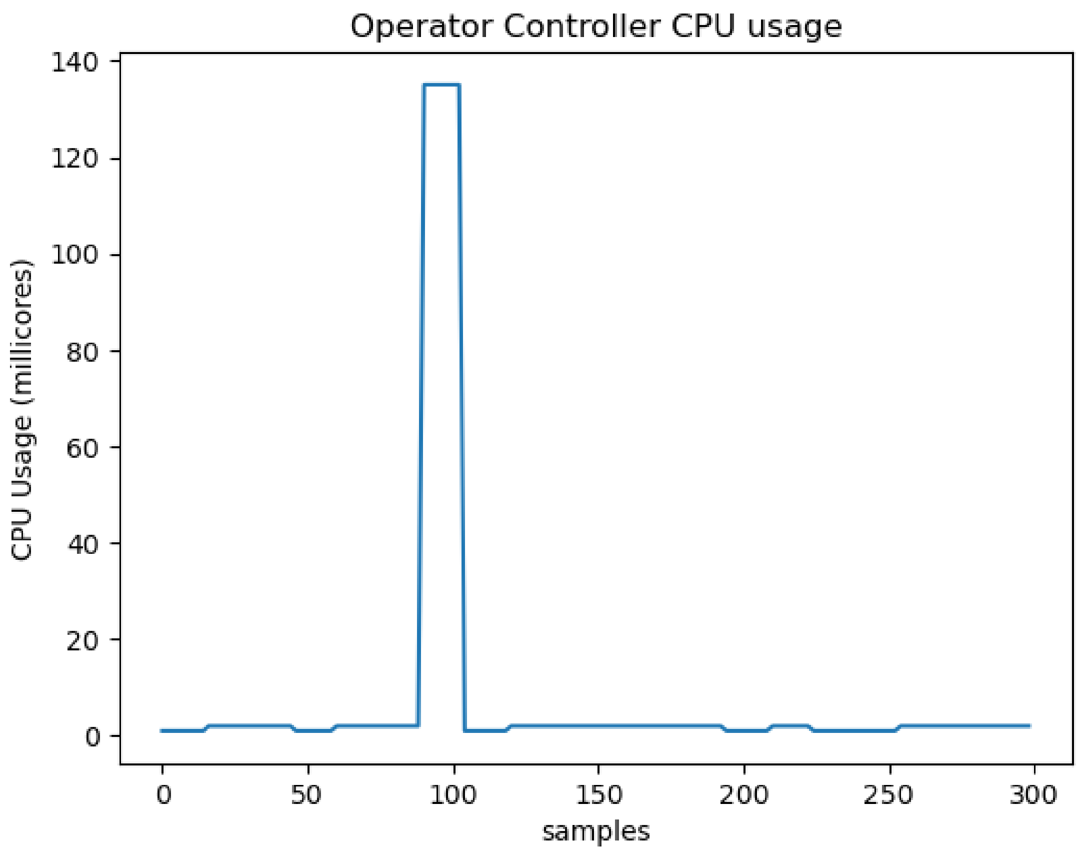
```
[20]: df = None
      df = df1.loc[df1['pod'].str.
       ↪contains('catalogd-controller-manager'),['cpu0','cpu1','mem0','mem1']]
```

```
[21]: df.insert(4, "cputotal", np.ceil((df.cpu0.str.rstrip('n').astype(int) + df.cpu1.
       ↪str.rstrip('n').astype('int')) / 1000000).astype(int), True)
```

```
[22]: df.insert(5, "memtotal", np.round((df.mem0.str.rstrip('Ki').astype(int) + df.
       ↪mem1.str.rstrip('Ki').astype(int)) / 1024).astype(int), True)
```

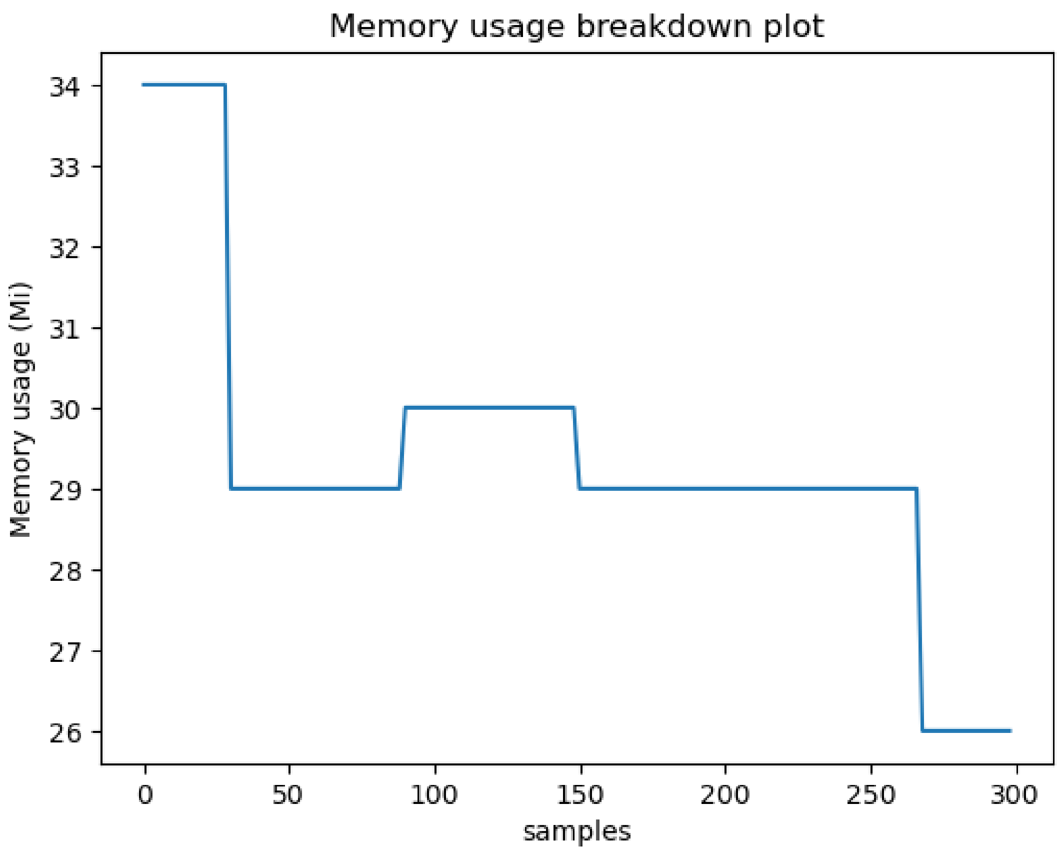
```
[23]: df.cputotal.plot(title='Operator Controller CPU usage', xlabel='samples',␣
       ↪ylabel='CPU Usage (millicores)', kind='line')
```

```
[23]: <Axes: title={'center': 'Operator Controller CPU usage'}, xlabel='samples',
      ylabel='CPU Usage (millicores)'>
```



```
[24]: df.memtotal.plot(title='Memory usage breakdown plot', kind='line',␣
        ↪xlabel='samples', ylabel='Memory usage (Mi)')
```

```
[24]: <Axes: title={'center': 'Memory usage breakdown plot'}, xlabel='samples',
      ylabel='Memory usage (Mi)'>
```



[25]:
```
df
```

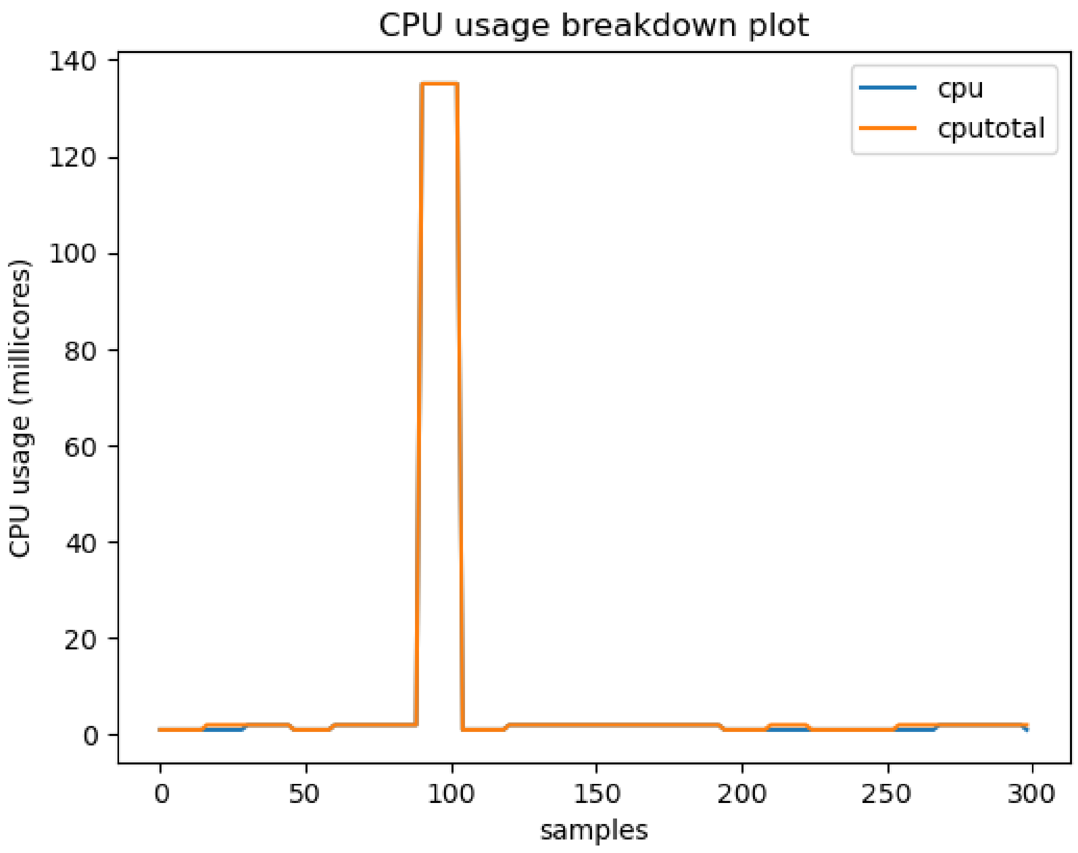
[25]:
```
          cpu0    cpu1     mem0    mem1  cputotal  memtotal
0      888559n  35905n  25576Ki  9252Ki         1        34
2      888559n  35905n  25576Ki  9252Ki         1        34
4      888559n  35905n  25576Ki  9252Ki         1        34
6      888559n  35905n  25576Ki  9252Ki         1        34
8      888559n  35905n  25576Ki  9252Ki         1        34
..         …      …       …     …        …        …
290   1319206n  23445n  17988Ki  8740Ki         2        26
292   1319206n  23445n  17988Ki  8740Ki         2        26
294   1319206n  23445n  17988Ki  8740Ki         2        26
296   1319206n  23445n  17988Ki  8740Ki         2        26
298    979382n  26186n  17988Ki  8740Ki         2        26

[150 rows x 6 columns]
```

[26]:
```
cpu = np.ceil(df.cpu0.str.rstrip('n').astype(int) / 1000000).astype(int)
df.insert(6, "cpu", cpu, True)
```

[27]:
```
mem = np.round(df.mem0.str.rstrip('Ki').astype(int) / 1024).astype(int)
df.insert(7, "mem", mem, True)
```

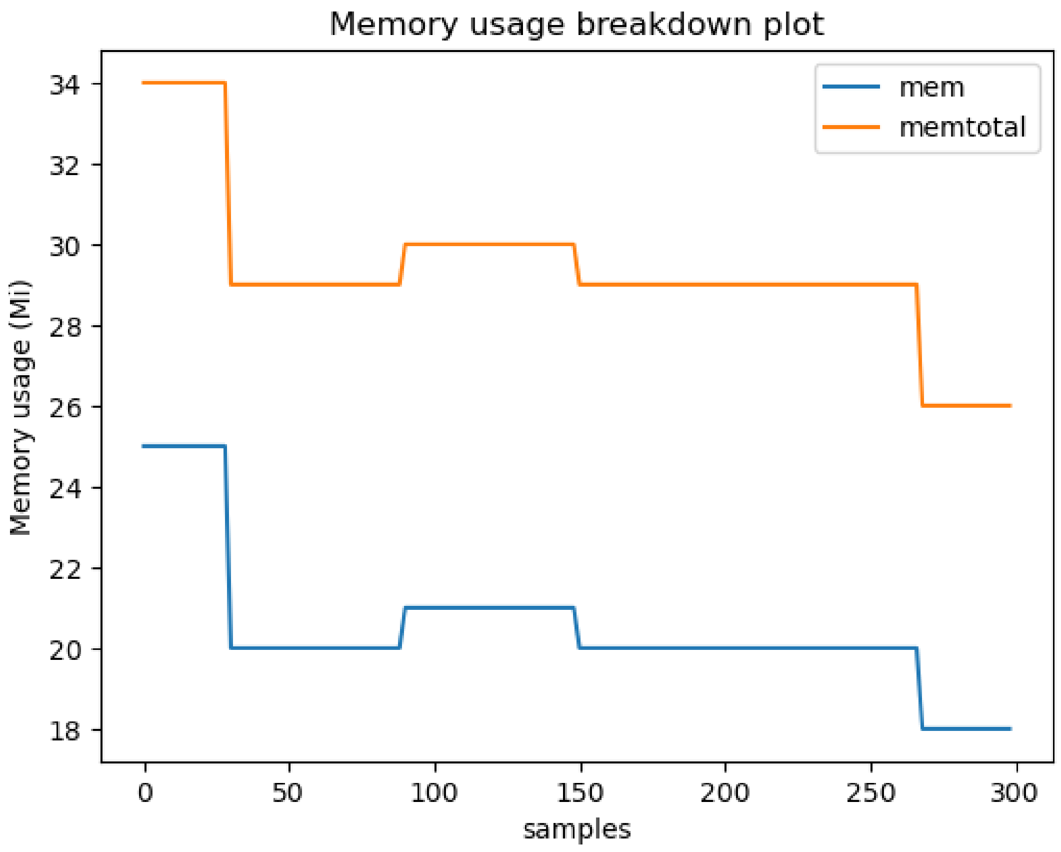
[28]:
```
df.plot(y=['cpu','cputotal'], title='CPU usage breakdown plot', kind='line',␣
 ↪xlabel='samples', ylabel='CPU usage (millicores)')
```

[28]:
```
<Axes: title={'center': 'CPU usage breakdown plot'}, xlabel='samples',
ylabel='CPU usage (millicores)'>
```



[29]:
```
df.plot(y=['mem','memtotal'], title='Memory usage breakdown plot', kind='line',␣
 ↪xlabel='samples', ylabel='Memory usage (Mi)')
```

[29]:
```
<Axes: title={'center': 'Memory usage breakdown plot'}, xlabel='samples',
ylabel='Memory usage (Mi)'>
```


Memory usage breakdown plot
Memory usage (Mi)
samples
mem
memtotal
34
32
30
28
26
24
22
20
18
0
50
100
150
200
250
300

```
[ ]:
```