Hatchet: Identifying performance issues

Abhinav Bhatele
Department of Computer Science, University of Maryland





Hatchet operations (recap)

- filter: sub-select the data by filtering the dataframe
  - callpath query language to specify callpath patterns for filtering
- squash: prune the graph based on a previously applied filter
- drop_index_levels: aggregate data across ranks and/or threads
- add / subtract / multiply / divide
- tree: string representation of graph

Example 1: Generating a flat profile

```
gf = ht.GraphFrame.from_hpctoolkit('kripke')
gf.drop_index_levels()

grouped = gf.dataframe.groupby('name').sum()
sorted_df = grouped.sort_values(by=['time'], ascending=False)
print(sorted_df)
```

Example 1: Generating a flat profile

```
gf = ht.GraphFrame.from_hpctoolkit('kripke')
gf.drop_index_levels()

grouped = gf.dataframe.groupby('name').sum()
sorted_df = grouped.sort_values(by=['time'], ascending=False)
print(sorted_df)
```

Example 1: Generating a flat profile

```
gf = ht.GraphFrame.from_hpctoolkit('kripke')
gf.drop_index_levels()

grouped = gf.dataframe.groupby('name').sum()
sorted_df = grouped.sort_values(by=['time'], asce
print(sorted_df)
```

name	nid	time	time (inc)
<unknown file> [kripke]:0	17234	1.825282e+08	1.825282e+08
Kernel_3d_DGZ::scattering	60	7.669936e+07	7.896253e+07
Kernel_3d_DGZ::LTimes	30	5.010439e+07	5.240528e+07
Kernel_3d_DGZ::LPlusTimes	115	4.947707e+07	5.104498e+07
Kernel_3d_DGZ::sweep	981	5.018862e+06	5.018862e+06
memset.S:99	3773	3.168982e+06	3.168982e+06
memset.S:101	3970	2.120895e+06	2.120895e+06
Grid_Data::particleEdit	1201	1.131266e+06	1.249157e+06
<unknown file> [libpsm2.so.2.1]:0	324763	9.733415e+05	9.733415e+05
memset.S:98	3767	6.197776e+05	6.197776e+05

Example 2: Identifying load imbalance

```
gf1 = ht.GraphFrame.from_caliper('lulesh-512cores')
gf2 = gf1.copy()

gf1.drop_index_levels(function=np.mean)
gf2.drop_index_levels(function=np.max)

gf1.dataframe['imbalance'] = gf2.dataframe['time'].div(gf1.dataframe['time'])

sorted_df = gf1.dataframe.sort_values(by=['imbalance'], ascending=False)
print(sorted_df)
```

Example 2: Identifying load imbalance

```
gf1 = ht.GraphFrame.from_caliper('lulesh-512cores')
gf2 = gf1.copy()

gf1.drop_index_levels(function=np.mean)
gf2.drop_index_levels(function=np.max)

gf1.dataframe['imbalance'] = gf2.dataframe['time'].div(gf1.dataframe['time'])

sorted_df = gf1.dataframe.sort_values(by=['imbalance'], ascending=False)
print(sorted_df)
```

Example 2: Identifying load imbalance

```
gf1 = ht.GraphFrame.from_caliper('lulesh-512cores')
gf2 = gf1.copy()

gf1.drop_index_levels(function=np.mean)
gf2.drop_index_levels(function=np.max)

gf1.dataframe['imbalance'] = gf2.dataframe['time'].div(gf1.dataframe['time'])

sorted_df = gf1.dataframe.sort_values(by=['imbalance'], ascending=False)
print(sorted_df)
```

Example 2: Identifying load imbalance

```
gf1 = ht.GraphFrame.from_caliper('lulesh-512cores')
gf2 = gf1.copy()

gf1.drop_index_levels(function=np.mean)
gf2.drop_index_levels(function=np.max)

gf1.dataframe['imbalance'] = gf2.dataframe['time'].div(gf1.dataframe['time'])

sorted_df = gf1.dataframe.sort_values(by=['imbalance'], ascending=False)
print(sorted_df)
```

node	name	nid	time	time (inc)	imbalance
LagrangeNodal	LagrangeNodal	3.0	2.242594e+06	2.593621e+07	2.494720
main	main	0.0	1.106013e+05	5.357208e+07	2.161845
CalcForceForNodes	CalcForceForNodes	4.0	1.033639e+06	2.369361e+07	2.142526
CalcQForElems	CalcQForElems	16.0	3.351894e+06	6.649351e+06	2.037651
CalcEnergyForElems	CalcEnergyForElems	22.0	1.571996e+06	2.807323e+06	2.013174
CalcPressureForElems	CalcPressureForElems	23.0	1.235327e+06	1.235327e+06	2.005437

Example 3: Comparing two executions

```
gf1 = ht.GraphFrame.from_caliper('lulesh-1core.json')
gf2 = ht.GraphFrame.from_caliper('lulesh-27cores. json')

gf2.drop_index_levels()
gf3 = gf2 - gf1

sorted_df = gf3.dataframe.sort_values(by=['time'], ascending=False)
print(sorted_df)
```

Example 3: Comparing two executions

```
gf1 = ht.GraphFrame.from_caliper('lulesh-1core.json')
gf2 = ht.GraphFrame.from_caliper('lulesh-27cores. json')

gf2.drop_index_levels()
gf3 = gf2 - gf1

sorted_df = gf3.dataframe.sort_values(by=['time'], ascending=False)
print(sorted_df)
```

Example 3: Comparing two executions

```
gf1 = ht.GraphFrame.from_caliper('lulesh-1core.json')
gf2 = ht.GraphFrame.from_caliper('lulesh-27cores. json')

gf2.drop_index_levels()
gf3 = gf2 - gf1

sorted_df = gf3.dataframe.sort_values(by=['time'], ascending=False)
print(sorted_df)
```

node	name	nid	time	time (inc)
TimeIncrement	TimeIncrement	25.0	8.505048e+06	8.505048e+06
CalcQForElems	CalcQForElems	16.0	4.455672e+06	5.189453e+06
CalcHourglassControlForElems	CalcHourglassControlForElems	7.0	3.888798e+06	4.755817e+06
LagrangeNodal	LagrangeNodal	3.0	1.986046e+06	8.828475e+06
CalcForceForNodes	CalcForceForNodes	4.0	1.017857e+06	6.842429e+06

Example 4: Filtering by library

```
gf1 = GraphFrame.from_caliper('lulesh-27cores')
gf1.drop_index_levels()
filtered_gf1 = gf1.filter(lambda x: x['name'].startswith('MPI'))
squashed_gf1 = filtered_gf1.squash()

gf2 = GraphFrame.from_caliper('lulesh-512cores')
gf2.drop_index_levels()
filtered_gf2 = gf2.filter(lambda x: x['name'].startswith('MPI'))
squashed_gf2 = filtered_gf2.squash()

diff_gf = squashed_gf2 - squashed_gf1

sorted_df = diff_gf.dataframe.sort_values(by=['time'], ascending=False)
print(sorted_df)
```

Example 4: Filtering by library

```
gf1 = GraphFrame.from_caliper('lulesh-27cores')
gf1.drop_index_levels()
filtered_gf1 = gf1.filter(lambda x: x['name'].startswith('MPI'))
squashed_gf1 = filtered_gf1.squash()

gf2 = GraphFrame.from_caliper('lulesh-512cores')
gf2.drop_index_levels()
filtered_gf2 = gf2.filter(lambda x: x['name'].startswith('MPI'))
squashed_gf2 = filtered_gf2.squash()

diff_gf = squashed_gf2 - squashed_gf1

sorted_df = diff_gf.dataframe.sort_values(by=['time'], ascending=False)
print(sorted_df)
```

Example 4: Filtering by library

```
gf1 = GraphFrame.from_caliper('lulesh-27cores')
gf1.drop_index_levels()
filtered_gf1 = gf1.filter(lambda x: x['name'].startswith('MPI'))
squashed_gf1 = filtered_gf1.squash()

gf2 = GraphFrame.from_caliper('lulesh-512cores')
gf2.drop_index_levels()
filtered_gf2 = gf2.filter(lambda x: x['name
squashed_gf2 = filtered_gf2.squash()

diff_gf = squashed_gf2 - squashed_gf1

sorted_df = diff_gf.dataframe.sort_values(b
print(sorted_df)
```

node	node	time (inc)	name	nid	time
MPI_Allreduce	MPI_Allreduce	2.072371e+06	MPI_Allreduce	3	2.072371e+06
MPI_Finalize	MPI_Finalize	4.042198e+04	MPI_Finalize	0	4.042198e+04
MPI_Isend	MPI_Isend	1.753768e+04	MPI_Isend	15	1.753768e+04
MPI_Isend	MPI_Isend	7.718737e+03	MPI_Isend	13	7.718737e+03
MPI_Isend	MPI_Isend	7.542969e+03	MPI_Isend	7	7.542969e+03
MPI_Waitall	MPI_Waitall	4.573508e+03	MPI_Waitall	5	4.573508e+03
MPI_Barrier	MPI_Barrier	4.240952e+03	MPI_Barrier	12	4.240952e+03

Example 5: Scaling study

```
datasets = glob.glob('lulesh*.json')
datasets.sort()

dataframes = []
for dataset in datasets:
    gf = ht.GraphFrame.from_caliper(dataset)
    gf.drop_index_levels()

    num_pes = re.match('(.*)-(\d+)(.*)', dataset).group(2)
    gf.dataframe['pes'] = num_pes
    filtered_gf = gf.filter(lambda x: x['time'] > 1e6)
    dataframes.append(filtered_gf.dataframe)

result = pd.concat(dataframes)
pivot_df = result.pivot(index='pes', columns='name', values='time')
pivot_df.loc[:,:].plot.bar(stacked=True, figsize=(10,7))
```

Example 5: Scaling study

```
datasets = glob.glob('lulesh*.json')
datasets.sort()

dataframes = []
for dataset in datasets:
    gf = ht.GraphFrame.from_caliper(dataset)
    gf.drop_index_levels()

    num_pes = re.match('(.*)-(\d+)(.*)', dataset).group(2)
    gf.dataframe['pes'] = num_pes
    filtered_gf = gf.filter(lambda x: x['time'] > 1e6)
    dataframes.append(filtered_gf.dataframe)

result = pd.concat(dataframes)
pivot_df = result.pivot(index='pes', columns='name', values='time')
pivot_df.loc[:,:].plot.bar(stacked=True, figsize=(10,7))
```

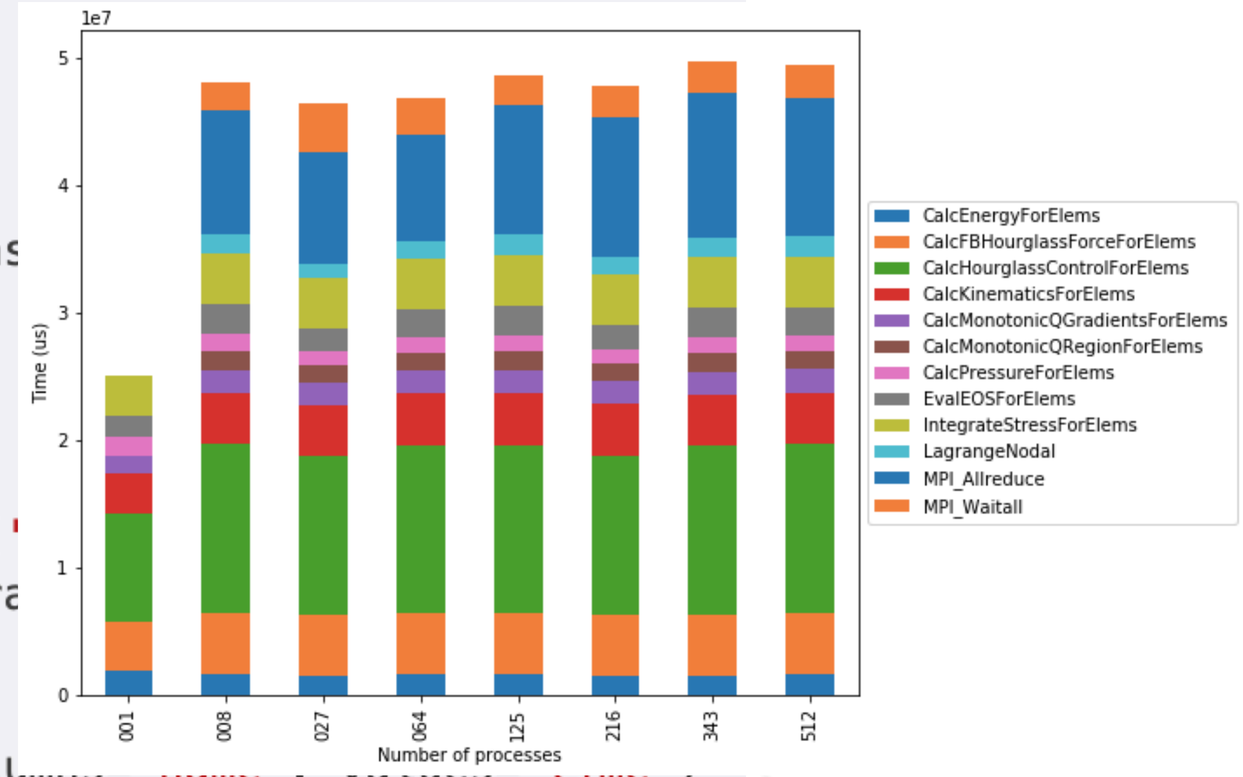
Example 5: Scaling study

```
datasets = glob.glob('lulesh*.json')
datasets.sort()

dataframes = []
for dataset in datasets:
    gf = ht.GraphFrame.from_caliper(datas
    gf.drop_index_levels()

    num_pes = re.match('(.*)-(\d+)(.*)',
    gf.dataframe['pes'] = num_pes
    filtered_gf = gf.filter(lambda x: x['
    dataframes.append(filtered_gf.datafra

result = pd.concat(dataframes)
pivot_df = result.pivot(index='pes', colu
pivot_df.loc[:,:].plot.bar(stacked=True, figsize=(10,7))
```



Abhinav Bhatele

Parallel Software and Systems Group

5218 Brendan Iribe Center (IRB) / College Park, MD 20742

phone: 301.405.4507 / e-mail: bhatele@cs.umd.edu

UNIVERSITY OF MARYLAND