Introduction to Parallel Computing (CMSC498X / CMSC818X)

Abhinav Bhatele, Department of Computer Science





Announcements

- Quiz I has been posted
- Deadline: October 1, 11:59 pm AoE
- Department seminar tomorrow at 11:00 am
 - Zoom link forwarded by e-mail



Abhinav Bhatele (CMSC498X/CMSC818X)

Performance analysis

- Parallel performance of a program might not be what the developer expects
- How do we find performance bottlenecks?
- Two parts to performance analysis: measurement and analysis/visualization
- Simplest tool: timers in the code and printf



Abhinav Bhatele (CMSC498X/CMSC818X)





Using timers

double start, end; double phase1, phase2, phase3;

```
start = MPI_Wtime();
 ... phasel code ...
end = MPI Wtime();
phase1 = end - start;
```

```
start = MPI Wtime();
 ... phase2 ...
end = MPI_Wtime();
phase2 = end - start;
```

```
start = MPI_Wtime();
 ... phase3 ...
end = MPI_Wtime();
phase3 = end - start;
```



Abhinav Bhatele (CMSC498X/CMSC818X)

Using timers

double start, end; double phase1, phase2, phase3;

```
start = MPI_Wtime();
 ... phase1 code ...
end = MPI Wtime();
phase1 = end - start;
```

```
start = MPI_Wtime();
 ... phase2 ...
end = MPI_Wtime();
phase2 = end - start;
```

```
start = MPI_Wtime();
 ... phase3 ...
end = MPI_Wtime();
phase3 = end - start;
```



Phase I took 2.45 s

Phase 2 took 11.79 s

Phase 3 took 4.37 s

Abhinav Bhatele (CMSC498X/CMSC818X)

Performance tools

Tracing tools

• Capture entire execution trace

Profiling tools

• Provide aggregated information

- Typically use statistical sampling
- Many tools can do both



Abhinav Bhatele (CMSC498X/CMSC818X)



Metrics recorded

- Counts of function invocations
- Time spent in code
- Number of bytes sent
- Hardware counters
- To fix performance problems we need to connect metrics to source code



Abhinav Bhatele (CMSC498X/CMSC818X)

Tracing tools

• Record all the events in the program with timestamps

• Events: function calls, MPI events, etc.

Vampir visualization: <u>https://hpc.llnl.gov/software/development-environment-software/vampir-vampir-server</u>



Abhinav Bhatele (CMSC498X/CMSC818X)



Tracing tools

Record all the events in the program with timestamps

• Events: function calls, MPI events, etc.

Vampir visualization: <u>https://hpc.llnl.gov/software/development-environment-software/vampir-vampir-server</u>





Abhinav Bhatele (CMSC498X/CMSC818X)



Tracing tools

- Record all the events in the program with timestamps
- Events: function calls, MPI events, etc.



Vampir visualization: <u>https://hpc.llnl.gov/software/development-environment-software/vampir-vampir-server</u>





Abhinav Bhatele (CMSC498X/CMSC818X)



Examples of tracing tools

- VampirTrace
- Score-P
- TAU
- Projections
- HPCToolkit



Abhinav Bhatele (CMSC498X/CMSC818X)



Profiling tools

- Ignore the specific times at which events occurred
- Provide aggregate information about different parts of the code
- Examples:
 - Gprof, perf
 - mpiP
 - HPCToolkit, caliper
- Python tools: cprofile, pyinstrument, scalene



Abhinav Bhatele (CMSC498X/CMSC818X)

gprof Data	Q Enter filter	text		🔵 🕀 🖻 🕀 🗋
4 bytes per bucket, each sample count	ts as 10.000ms			
Name (location)	Samples	Calls	Time/Call	% Time
▼Summary	2228			100.0%
►calc.c	590			26.48%
▶ copy.c	0			0.0%
▶diag.c	25			1.12%
▶main.c	0			0.0%
▶time.c	653			29.31%
▼tstep.c	958			43.0%
▼tstep	958	10000	957.999us	43.0%
tstep (tstep.c:47)	1			0.04%
tstep (tstep.c:48)	62			2.78%
tstep (tstep.c:49)	46			2.06%
tstep (tstep.c:50)	46			2.06%
tstep (tstep.c:51)	48			2.15%
tstep (tstep.c:58)	101			4.53%
tstep (tstep.c:59)	135			6.06%
tstep (tstep.c:60)	120			5.39%
tstep (tstep.c:61)	126			5.66%
tstep (tstep.c:66)	3			0.13%
tstep (tstep.c:67)	108			4.85%
tstep (tstep.c:68)	63			2.83%
tstep (tstep.c:69)	43			1.93%
tstep (tstep.c:70)	56			2.51%
▶worker.c	2			0.09%

Gprof data in hpctView



Calling contexts, trees, and graphs

- Calling context or call path: Sequence of function invocations leading to the current sample
- Calling context tree (CCT): dynamic prefix tree of all call paths in an execution
- Call graph: merge nodes in a CCT with the same name into a single node but keep caller-callee relationships as arcs



Abhinav Bhatele (CMSC498X/CMSC818X)





LIVE RECORDING







Abhinav Bhatele (CMSC498X/CMSC818X)

LIVE RECORDING





Abhinav Bhatele (CMSC498X/CMSC818X)

LIVE RECORDING





Contextual information

File Line number Function name Callpath Load module Process ID Thread ID





Abhinav Bhatele (CMSC498X/CMSC818X)

Contextual information

File Line number Function name Callpath Load module Process ID Thread ID

Performance Metrics

- Time
- Flops
- Cache misses





LIVE RECORDING

Output of profiling tools

- Flat profile: Listing of all functions with counts and execution times
- Call graph profile
- Calling context tree



Abhinav Bhatele (CMSC498X/CMSC919X)



foo



Hatchet

- Hatchet enables programmatic analysis of parallel profiles
- Leverages pandas which supports multi-dimensional tabular datasets
- Create a structured index to enable indexing pandas dataframes by nodes in a graph
- A set of operators to filter, prune and/or aggregate structured data

https://hatchet.readthedocs.io/en/latest/



Abhinav Bhatele (CMSC498X/CMSC818X)





Abhinav Bhatele (CMSC498X/CMSC818X)



 Pandas is an open-source Python library for data analysis



Abhinav Bhatele (CMSC498X/CMSC818X)



- Pandas is an open-source Python library for data analysis
- Dataframe: two-dimensional tabular data structure
 - Supports many operations borrowed from SQL databases



C .		
CO	lum	INS
	_	_

			node	name	time (inc)	time
		0	{'name': 'main'}	main	200.0	10.0
		1	{'name': 'physics'}	physics	60.0	40.0
a		2	{'name': 'mpi'}	mpi	20.0	5.0
	Rows	3	{'name': 'psm2'}	psm2	15.0	30.0
		4	{'name': 'solvers'}	solvers	100.0	10.0
		5	{'name': 'hypre'}	hypre	65.0	30.0
		6	{'name': 'mpi'}	mpi	35.0	20.0
		7	{'name': 'psm2'}	psm2	25.0	60.0



- Pandas is an open-source Python librar for data analysis
- Dataframe: two-dimensional tabular data structure
 - Supports many operations borrowed from SQL databases



	Index	< // C	Columns									
		node	name	time (inc)	time							
	o	{'name': 'main'}	main	200.0	10.0							
	1	{'name': 'physics'}	physics	60.0	40.0							
	2	{'name': 'mpi'}	mpi	20.0	5.0							
Rows	3	{'name': 'psm2'}	psm2	15.0	30.0							
	4	{'name': 'solvers'}	solvers	100.0	10.0							
	5	{'name': 'hypre'}	hypre	65.0	30.0							
	6	{'name': 'mpi'}	mpi	35.0	20.0							
	7	{'name': 'psm2'}	psm2	25.0	60.0							



- Pandas is an open-source Python librar for data analysis
- Dataframe: two-dimensional tabular da structure
 - Supports many operations borrowed from SQL databases
- MultiIndex enables working with highdimensional data in a 2D data structure



	Inde	x // //C	Columns									
Y	\leq	node	name	time (inc)	time							
	ο	{'name': 'main'}	main	200.0	10.0							
ta	/ 1	{'name': 'physics'}	physics	60.0	40.0							
La	2	{'name': 'mpi'}	mpi	20.0	5.0							
Row	S 3	{'name': 'psm2'}	psm2	15.0	30.0							
	4	{'name': 'solvers'}	solvers	100.0	10.0							
	5	{'name': 'hypre'}	hypre	65.0	30.0							
	6	{'name': 'mpi'}	mpi	35.0	20.0							
e	7	{'name': 'psm2'}	psm2	25.0	60.0							



Central data structure: a GraphFrame

- Consists of a structured index graph object and a pandas dataframe
- Graph stores caller-callee relationships
- Dataframe stores all numerical and categorical data



Abhinav Bhatele (CMSC498X/CMSC818X)

Central data structure: a GraphFrame

- Consists of a structured index graph object and a pandas dataframe
- Graph stores caller-callee relationships
- Dataframe stores all numerical and categorical data



Abhinav Bhatele (CMSC498X/CMSC818X)



Central data structure: a GraphFrame

physics

mpi

psm2

- Consists of a structured index graph object and a pandas dataframe
- Graph stores caller-callee relationships
- Dataframe stores all numerical and categorical data



		name	nid	node	time	time (inc)
hin	node					
	main	main	0	main	40.0	200.0
	physics	physics	1	physics	40.0	60.0
	mpi	mpi	2	mpi	5.0	20.0
	psm2	psm2	3	psm2	15.0	15.0
mpi	solvers	solvers	4	solvers	0.0	100.0
	hypre	hypre	5	hypre	65.0	65.0
psm2	mpi	mpi	6	mpi	10.0	35.0
	psm2	psm2	7	psm2	25.0	25.0



Dataframe operation: filter

filtered_gf = gf.filter(lambda x: x['time'] > 10.0)



Abhinav Bhatele (CMSC498X/CMSC818X)



Dataframe operation: filter

filtered_gf = gf.filter(lambda x: x['time'] > 10.0)

	name	nid	node	time	time (inc)
node					
main	main	0	main	40.0	200.0
physics	physics	1	physics	40.0	60.0
mpi	mpi	2	mpi	5.0	20.0
psm2	psm2	3	psm2	15.0	15.0
solvers	solvers	4	solvers	0.0	100.0
hypre	hypre	5	hypre	65.0	65.0
mpi	mpi	6	mpi	10.0	35.0
psm2	psm2	7	psm2	25.0	25.0



Abhinav Bhatele (CMSC498X/CMSC818X)



Dataframe operation: filter

filtered_gf = gf.filter(lambda x: x['time'] > 10.0)

	name	nid	node	time	time (inc)
node					
main	main	0	main	40.0	200.0
physics	physics	1	physics	40.0	60.0
mpi	mpi	2	mpi	5.0	20.0
psm2	psm2	3	psm2	15.0	15.0
solvers	solvers	4	solvers	0.0	100.0
hypre	hypre	5	hypre	65.0	65.0
mpi	mpi	6	mpi	10.0	35.0
psm2	psm2	7	psm2	25.0	25.0



Abhinav Bhatele (CMSC498X/CMSC818X)

	name	nid	node	time	time (inc)
node					
main	main	0	main	40.0	200.0
physics	physics	1	physics	40.0	60.0
psm2	psm2	3	psm2	15.0	15.0
hypre	hypre	5	hypre	65.0	65.0
psm2	psm2	7	psm2	25.0	25.0



filtered_gf = gf.filter(lambda x: x['time'] > 10.0)

	name	nid	node	time	time (inc)						
node											
main	main	0	main	40.0	200.0						
physics	physics	1	physics	40.0	60.0		name	nid	node	time	time (inc)
mpi	mpi	2	mpi	5.0	20.0	node					
psm2	psm2	3	psm2	15.0	15.0	main	main	0	main	40.0	200.0
solvers	solvers	4	solvers	0.0	100.0	physics	physics	1	physics	40.0	60.0
hypre	hypre	5	hypre	65.0	65.0	psm2	psm2	3	psm2	15.0	15.0
mpi	mpi	6	mpi	10.0	35.0	hypre	hypre	5	hypre	65.0	65.0
psm2	psm2	7	psm2	25.0	25.0	psm2	psm2	7	psm2	25.0	25.0





COMPUTER SCIENCE

Abhinav Bhatele (CMSC498X/CMSC818X)



filtered_gf = gf.filter(lambda x: x['time'] > 10.0)

							name	nid	node	time	time (inc)						
	namo	nid	nodo	timo	time (inc)	node											
nodo	name	ma	node	ume	ume (mc)	main	main	0	main	40.0	200.0						
node				40.0	000.0	physics	physics	1	physics	40.0	60.0		name	nid	node	time	time (inc)
main	main	0	main	40.0	200.0	mpi	mpi	2	mpi	5.0	20.0	node					
physics	physics	1	physics	40.0	60.0	psm2	psm2	3	psm2	15.0	15.0	main	main	0	main	40.0	200.0
mpi	mpi	2	mpi	5.0	20.0	solvers	solvers	4	solvers	0.0	100.0	physics	physics	1	physics	40.0	60.0
psm2	psm2	3	psm2	15.0	15.0	hypre	hypre	5	hypre	65.0	65.0	psm2	psm2	3	psm2	15.0	15.0
solvers	solvers	4	solvers	0.0	100.0	mpic	mpic	6	mpic	10.0	25.0	bypre	hypre	5	hypre	65.0	65.0
hypre	hypre	5	hypre	65.0	65.0	mpi	mpi	0	mpi	10.0	35.0	nypie	nemQ	7	nemO	05.0	05.0
mpi	mpi	6	mpi	10.0	35.0	psm2	psm2	7	psm2	25.0	25.0	psm2	psmz		psmz	25.0	25.0
psm2	psm2	7	psm2	25.0	25.0	psm2	psm2	7	psm2	25.0	25.0						



COMPUTER SCIENCE

Abhinav Bhatele (CMSC498X/CMSC818X)



filtered_gf = gf.filter(lambda x: x['time'] > 10.0) squashed_gf = filtered_gf.squash() node time time (inc) name nid main 40.0 200.0 60.0 physics 40.0 psm2 15.0 15.0 hypre 65.0 65.0 psm2 25.0 25.0 main filter physics physics solvers hypre hypre psm2 psm2 psm2 psm2

	time (inc)	time	node	nid	name							
						node	time (in c)	t ine e	nede	n i d		
	200.0	40.0	main	0	main	main	time (inc)	ume	node	nia	name	
	60.0	40.0	physics	1	physics	physics						node
node	20.0	5.0	mpi	2	mpi	mpi	200.0	40.0	main	0	main	main
main	15.0	15.0	psm2	3	psm2	psm2	60.0	40.0	physics	1	physics	physics
physics	100.0	0.0	solvers	4	solvers	solvers	20.0	5.0	mpi	2	mpi	mpi
psm2	65.0	65.0	hypre	5	hypre	hypre	15.0	15.0	psm2	3	psm2	psm2
hvpre	35.0	10.0	mpi	6	mpi	mpi	100.0	0.0	solvers	4	solvers	solvers
nem2	05.0	05.0	mam0	7	mam0	inpi	65.0	65.0	hypre	5	hypre	hypre
panz	25.0	25.0	psmz	-	psm2	psm2	35.0	10.0	mpi	6	mpi	mpi
	25.0	25.0	psm2	7	psm2	psm2	25.0	25.0	psm2	7	psm2	psm2



COMPUTER SCIENCE

Abhinav Bhatele (CMSC498X/CMSC818X)



filtered_gf = gf.filter(lambda x: x['time'] > 10.0)

							name	nid	node	time	time (inc)						// '	name	nid	node	time	time (inc)						
	name	nid	node	time	time (inc)	node											node											
node						main	main	0	main	40.0	200.0						main	main	0	main	40.0	200.0						
main	main	0	main	40.0	200.0	physics	physics	1	physics	40.0	60.0		name	nid	node	time 1	ⁱ physics	physics	1	physics	40.0	60.0		name	nid	node	time	time (inc)
man		۲ ۲		40.0	200.0	mpi	mpi	2	mpi	5.0	20.0	node					mpi	mpi	2	mpi	5.0	20.0	node					
pnysics	pnysics	1	pnysics	40.0	60.0	psm2	psm2	3	psm2	15.0	15.0	main	main	0	main	40.0	psm2	psm2	3	psm2	15.0	15.0	main	main	0	main	40.0	200.0
mpi	mpi	2	mpi	5.0	20.0	solvers	solvers	4	solvers	0.0	100.0	physics	physics	1	physics	40.0	solvers	solvers	4	solvers	0.0	100.0	physics	physics	1	physics	40.0	60.0
psm2	psm2	3	psm2	15.0	15.0	hypre	hypre	5	hypre	65.0	65.0	psm2	psm2	3	psm2	15.0	hypre	hypre	5	hypre	65.0	65.0	psm2	psm2	3	psm2	15.0	15.0
solvers	solvers	4	solvers	0.0	100.0	mpi	mpi	6	mpi	10.0	35.0	hvpre	hvore	5	hvpre	65.0	mni	mpi	6	mpio	10.0	35.0	hypre	hvore	5	hvpre	65.0	65.0
hypre	hypre	5	hypre	65.0	65.0	IIIpi	трі	-	mpi	10.0	35.0	nomQ	nom	7	nom	25.0	mpi	тр	-	IIIpi	10.0	35.0	nom0	nom0	7	nom	25.0	25.0
mpi	mpi	6	mpi	10.0	35.0	psm2	psm2	-	psm2	25.0	25.0	psiliz	psinz	1	psinz	25.0	psm2	psm2		psm2	25.0	25.0	psinz	psmz	'	psinz	25.0	25.0
psm2	psm2	7	psm2	25.0	25.0	psm2	psm2	7	psm2	25.0	25.0																	
							name	nic	l node	time	time (inc)			Y,				<u></u>	-		<u>-</u>							
	(main				no					200.0	X	ma	in				S	50	ju	22	sh			(main		
			4			physi	cs physics	s 1	physics	40.0	60.0	na ph	ame hid lysics	no	de time olvers	time (inc)	$\neg \chi \chi$			77					/	T		
(physics) (solvers			_ n	npi mp	i 2	2 mpi	5.0	20.0	node					_//\							\prec		*	-	
						psi	m2 psm2	2 3	B psm2	15.0	15.0	main r	mein 0	ma	en 40.0	200.0	\sim							physics		hypre	psm	2
				*		solve	ers solvers	6 4	solvers	0.0	100.0	physics phy	vsics 1	physi	fife 20.0	mpi 60 0												
	mpi		nypre	m	pi	hyp	bre hypre	e 5	b hypre	65.0	65.0	psm2 p	sin2 3	psn	m2 15.0	15.0								psm2				
						n	npi mp	i 6	6 mpi	10.0	35.0	hypre h	vire 5	hyp	ore 65.0	65.0												
	psm2			psn	m2	psi	m2 psm2	2 7	psm2	25.0	25.0	psm2 🏼 🖻	sm2 7	psn	m2 25.0	psm225.2												

Abhinav Bhatele (CMSC498X/CMSC818X)

COMPUTER SCIENCE

time (inc)

200.0

60.0

15.0

65.0

25.0

squashed_gf = filtered_gf.squash()





gf1 = ht.GraphFrame.from_literal(...)
gf2 = ht.GraphFrame.from_literal(...)
gf2 -= gf1



Abhinav Bhatele (CMSC498X/CMSC818X)



gf2 = gf1





gf2 -= gf1





gf1 = ht.GraphFrame.from_literal(...)
gf2 = ht.GraphFrame.from_literal(...)
gf2 -= gf1





Abhinav Bhatele (CMSC498X/CMSC818X)

https://hatchet.readthedocs.io



Visualizing small graphs

FlameGraph

print(gf.tree(color=True))

```
0.000 foo
⊢ 5.000 bar
  ⊣ 5.000 baz
  └─ 10.000 grault
⊢ 0.000 qux
  └─ 5.000 quux
     └─ 10.000 corge
        ⊢ 5.000 bar
         ⊣ 5.000 baz
         └─ 10.000 grault
        └─ 15.000 garply
└─ 0.000 waldo
   ⊢ 5.000 fred
     ⊢ 5.000 plugh
     └─ 5.000 xyzzy
        └─ 5.000 thud
           ⊢ 5.000 baz
           └─ 15.000 garply
   └─ 15.000 garply
```

Abhinav Bhatele (CMSC498X/CMSC818X)





Visualizing small grap





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)



Abhinav Bhatele (CMSC498X/CMSC818X)



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)



Abhinav Bhatele (CMSC498X/CMSC818X)



Example 1: Generating a flat profile

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

grouped = gf.dataframe.groupby('name').s sorted_df = grouped.sort_values(by=['tim print(sorted_df)



sum()		nid		time	time (inc)
me'], asc∈	name				
	<unknown file=""> [kripke]:0</unknown>	17234	1.82528	2e+08	1.825282e+08
	Kernel_3d_DGZ::scattering	60	7.66993	6e+07	7.896253e+07
	Kernel_3d_DGZ::LTimes	30	5.01043	9e+07	5.240528e+07
	Kernel_3d_DGZ::LPlusTimes	115	4.94770	7e+07	5.104498e+07
	Kernel_3d_DGZ::sweep	981	5.01886	2e+06	5.018862e+06
	memset.S:99	3773	3.16898	2e+06	3.168982e+06
	memset.S:101	3970	2.12089	5e+06	2.120895e+06
	Grid_Data::particleEdit	1201	1.13126	6e+06	1.249157e+06
	<unknown file=""> [libpsm2.so.2.1]:0</unknown>	324763	9.73341	5e+05	9.733415e+05
	memset.S:98	3767	6.19777	e+05	6.197776e+05



Example 2: Comparing two executions

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

```
gf2_drop_index_levels()
qf3 = qf2 - qf1
```

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



Abhinav Bhatele (CMSC498X/CMSC818X)



Example 2: 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)



Abhinav Bhatele (CMSC498X/CMSC818X)



Example 2: Comparing two executions

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

```
gf2_drop_index_levels()
f_{gf3} = gf2 - gf1
```

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

node

TimeIncremen

CalcQForElems

CalcHourglassControlForElems

LagrangeNoda

CalcForceForNodes



)	name	nid	time	time (inc)
+	Timelnerement	25.0	9 5050490+06	9 5050490 06
L	Timeincrement	25.0	6.5050468+06	6.5050466+06
5	CalcQForElems	16.0	4.455672e+06	5.189453e+06
5	CalcHourglassControlForElems	7.0	3.888798e+06	4.755817e+06
l	LagrangeNodal	3.0	1.986046e+06	8.828475e+06
S	CalcForceForNodes	4.0	1.017857e+06	6.842429e+06



Example 3: 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 3: 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 3: 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))
```





UNIVERSITY OF MARYLAND

Abhinav Bhatele 5218 Brendan Iribe Center (IRB) / College Park, MD 20742 phone: 301.405.4507 / e-mail: bhatele@cs.umd.edu

