In [43]:
```
import sys
sys.path.append("/scratch/group/csce435-f23/python-3.8.17/lib/python3.8/site-packages"
sys.path.append("/scratch/group/csce435-f23/thicket")
from glob import glob

import matplotlib.pyplot as plt
import pandas as pd

import thicket as th

pd.set_option("display.max_rows", None)
pd.set_option("display.max_columns", None)
```

Weak Scaling

(increase problem size, increase number of processors)

Random

In [44]:
```
tkrand = th.Thicket.from_caliperreader(glob("cali_data_cuda/*-0.cali"))
tkrand.dataframe = tkrand.dataframe.drop(["nid", "spot.channel", "Total time", "Min ti
                                         "Avg GPU time/rank", "Min GPU time/rank", "M
```

In [45]:
```
gbrand = tkrand.groupby("InputSize")
```

```
5  thickets created...
{65536: <thicket.thicket.Thicket object at 0x2ad85bd63d00>, 262144: <thicket.thicket.
Thicket object at 0x2ad85bdccaf0>, 1048576: <thicket.thicket.Thicket object at 0x2ad8
5b1949a0>, 4194304: <thicket.thicket.Thicket object at 0x2ad85bdc5610>, 16777216: <th
icket.thicket.Thicket object at 0x2ad85bcfaa90>}
```

In [46]:
```
ctkrand = th.Thicket.concat_thickets(
    thickets=list(gbrand.values()),
    headers=list(gbrand.keys()),
    axis="columns",
    metadata_key="num_threads"
)
```

In [47]:
```
ctkrand.dataframe
```

Out[47]:

		65536	262144	1048576	4194304	16777216	
		Avg time/rank	Avg time/rank	Avg time/rank	Avg time/rank	Avg time/rank	
node	**num_threads**						
{'name': 'main', 'type': 'function'}	**2**	0.565978	0.980870	2.930263	12.370992	55.649466	
	4	0.348867	0.786451	2.916862	12.323504	55.578328	
	8	0.525454	0.932992	2.919970	12.612572	55.568973	
	16	0.525797	0.974294	2.893572	12.360763	55.646997	
	32	0.499803	0.987799	2.898303	12.338248	55.563891	
{'name': 'comm', 'type': 'function'}	**2**	0.563724	0.973532	2.902043	12.260180	55.205275	c
	4	0.346728	0.779057	2.888579	12.212627	55.137371	c
	8	0.523398	0.925575	2.891529	12.501300	55.127458	c
	16	0.523700	0.967011	2.865452	12.249997	55.202421	c
	32	0.497718	0.980410	2.870130	12.227546	55.122971	c
{'name': 'comm_large', 'type': 'function'}	**2**	0.563649	0.973471	2.901965	12.260099	55.205178	comm_
	4	0.346671	0.778992	2.888491	12.212554	55.137278	comm_
	8	0.523348	0.925521	2.891461	12.501205	55.127347	comm_
	16	0.523649	0.966947	2.865380	12.249921	55.202325	comm_
	32	0.497656	0.980347	2.870051	12.227464	55.122857	comm_
{'name': 'comp', 'type': 'function'}	**2**	0.000240	0.000092	0.000095	0.000151	0.000111	
	4	0.000130	0.000125	0.000153	0.000095	0.000096	
	8	0.000097	0.000092	0.000096	0.000110	0.000105	
	16	0.000095	0.000103	0.000096	0.000095	0.000112	
	32	0.000096	0.000099	0.000105	0.000104	0.000108	
{'name': 'comp_large', 'type': 'function'}	**2**	0.000197	0.000071	0.000075	0.000125	0.000083	comp_
	4	0.000107	0.000104	0.000122	0.000074	0.000075	comp_
	8	0.000075	0.000072	0.000074	0.000088	0.000081	comp_
	16	0.000074	0.000080	0.000074	0.000075	0.000088	comp_
	32	0.000073	0.000078	0.000081	0.000079	0.000084	comp_
{'name': 'correctness_check', 'type': 'function'}	**2**	0.000193	0.000735	0.002952	0.011671	0.046571	correctness_
	4	0.000195	0.000746	0.002936	0.011698	0.046870	correctness_
	8	0.000192	0.000745	0.002929	0.011687	0.046913	correctness_
	16	0.000192	0.000750	0.002933	0.011675	0.046649	correctness_
	32	0.000197	0.000757	0.002929	0.011685	0.046775	correctness_

		65536	262144	1048576	4194304	16777216	
		Avg time/rank	Avg time/rank	Avg time/rank	Avg time/rank	Avg time/rank	
node	**num_threads**						
{'name': 'data_init', 'type': 'function'}	**2**	0.001564	0.006268	0.024757	0.098522	0.396941	da
	4	0.001561	0.006269	0.024782	0.098656	0.393431	da
	8	0.001555	0.006344	0.025008	0.099005	0.393906	da
	16	0.001587	0.006189	0.024712	0.098554	0.397240	da
	32	0.001564	0.006297	0.024731	0.098474	0.393451	da

In [48]:
```
ctkrand.dataframe = ctkrand.dataframe.reset_index().drop(("node"), axis=1)
ctkrand.dataframe = ctkrand.dataframe.rename({("name", ""): "name", ("num_threads", "'
```

```
<ipython-input-48-625f039d34c3>:1: PerformanceWarning: dropping on a non-lexsorted mu
lti-index without a level parameter may impact performance.
  ctkrand.dataframe = ctkrand.dataframe.reset_index().drop(("node"), axis=1)
```

In [49]:
```
ctkrand.dataframe
```

Out[49]:

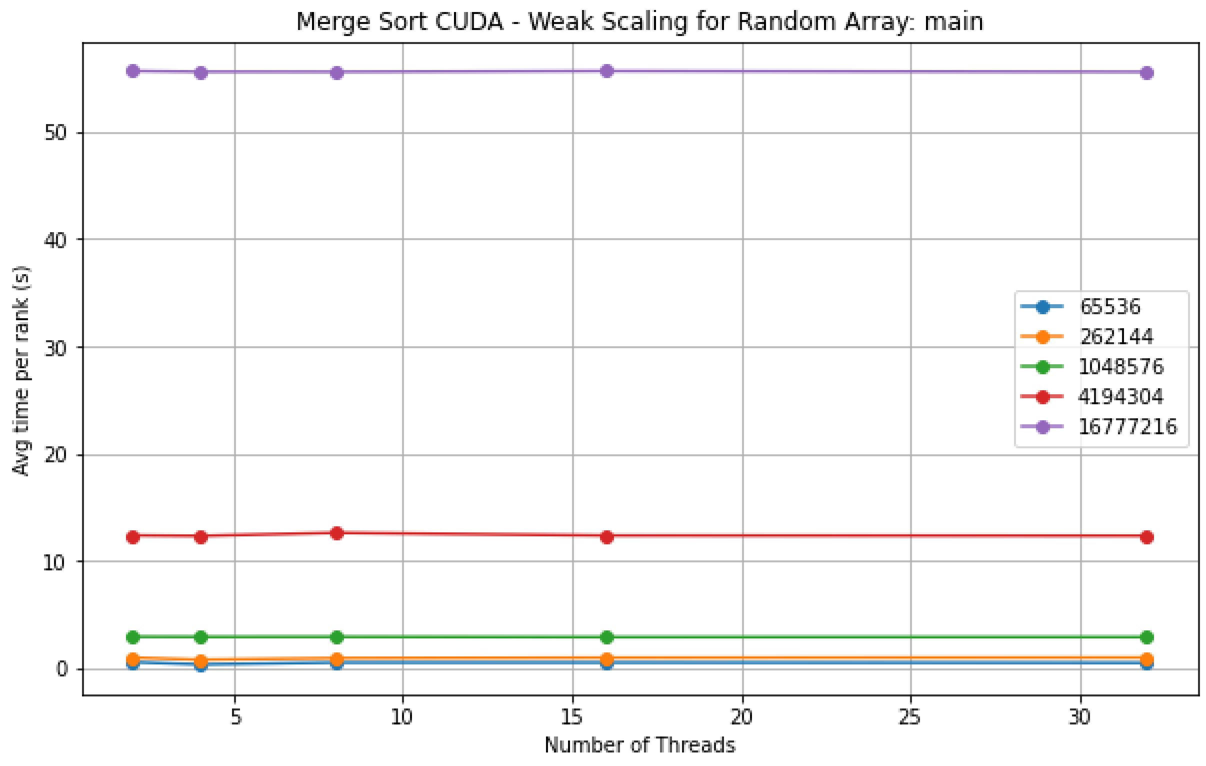
		65536	262144	1048576	4194304	16777216
		Avg time/rank	Avg time/rank	Avg time/rank	Avg time/rank	Avg time/rank
name	**num_threads**					
main	**2**	0.565978	0.980870	2.930263	12.370992	55.649466
	4	0.348867	0.786451	2.916862	12.323504	55.578328
	8	0.525454	0.932992	2.919970	12.612572	55.568973
	16	0.525797	0.974294	2.893572	12.360763	55.646997
	32	0.499803	0.987799	2.898303	12.338248	55.563891
comm	**2**	0.563724	0.973532	2.902043	12.260180	55.205275
	4	0.346728	0.779057	2.888579	12.212627	55.137371
	8	0.523398	0.925575	2.891529	12.501300	55.127458
	16	0.523700	0.967011	2.865452	12.249997	55.202421
	32	0.497718	0.980410	2.870130	12.227546	55.122971
comm_large	**2**	0.563649	0.973471	2.901965	12.260099	55.205178
	4	0.346671	0.778992	2.888491	12.212554	55.137278
	8	0.523348	0.925521	2.891461	12.501205	55.127347
	16	0.523649	0.966947	2.865380	12.249921	55.202325
	32	0.497656	0.980347	2.870051	12.227464	55.122857
comp	**2**	0.000240	0.000092	0.000095	0.000151	0.000111
	4	0.000130	0.000125	0.000153	0.000095	0.000096
	8	0.000097	0.000092	0.000096	0.000110	0.000105
	16	0.000095	0.000103	0.000096	0.000095	0.000112
	32	0.000096	0.000099	0.000105	0.000104	0.000108
comp_large	**2**	0.000197	0.000071	0.000075	0.000125	0.000083
	4	0.000107	0.000104	0.000122	0.000074	0.000075
	8	0.000075	0.000072	0.000074	0.000088	0.000081
	16	0.000074	0.000080	0.000074	0.000075	0.000088
	32	0.000073	0.000078	0.000081	0.000079	0.000084
correctness_check	**2**	0.000193	0.000735	0.002952	0.011671	0.046571
	4	0.000195	0.000746	0.002936	0.011698	0.046870
	8	0.000192	0.000745	0.002929	0.011687	0.046913
	16	0.000192	0.000750	0.002933	0.011675	0.046649
	32	0.000197	0.000757	0.002929	0.011685	0.046775
data_init	**2**	0.001564	0.006268	0.024757	0.098522	0.396941

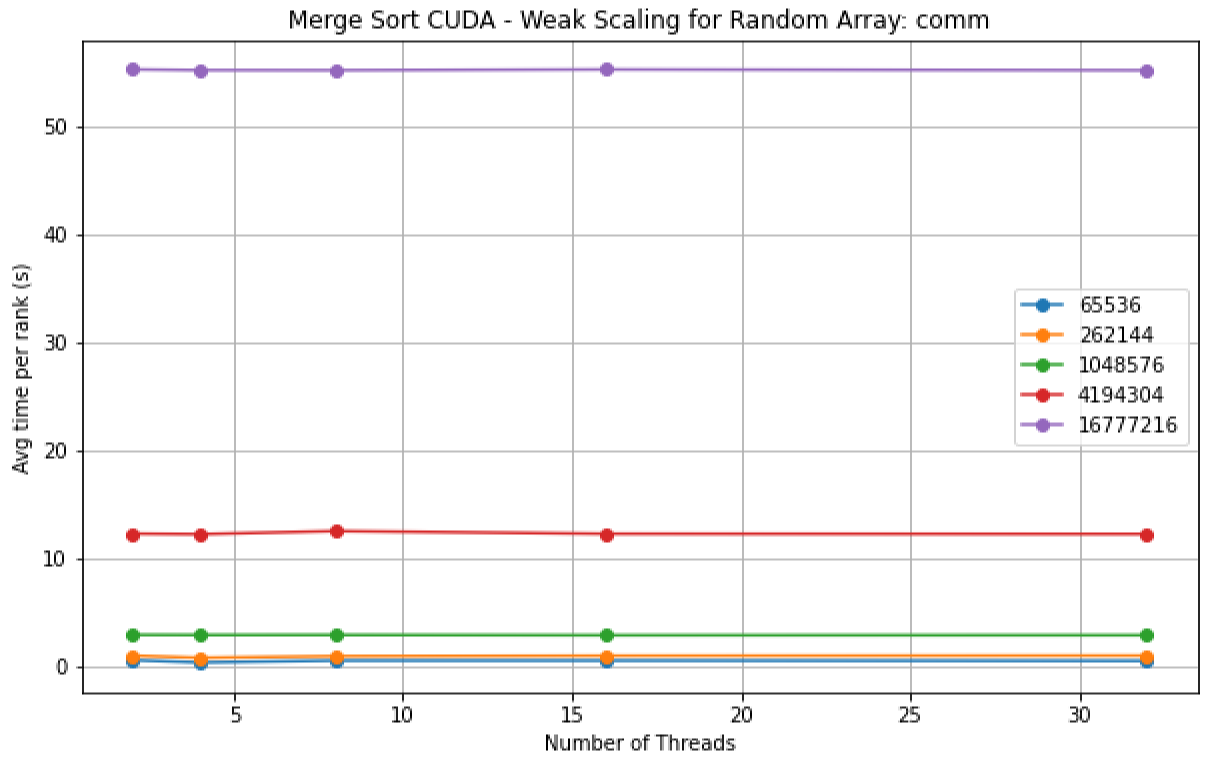
		65536	262144	1048576	4194304	16777216
		Avg time/rank	Avg time/rank	Avg time/rank	Avg time/rank	Avg time/rank
name	**num_threads**					
	4	0.001561	0.006269	0.024782	0.098656	0.393431
	8	0.001555	0.006344	0.025008	0.099005	0.393906
	16	0.001587	0.006189	0.024712	0.098554	0.397240
	32	0.001564	0.006297	0.024731	0.098474	0.393451

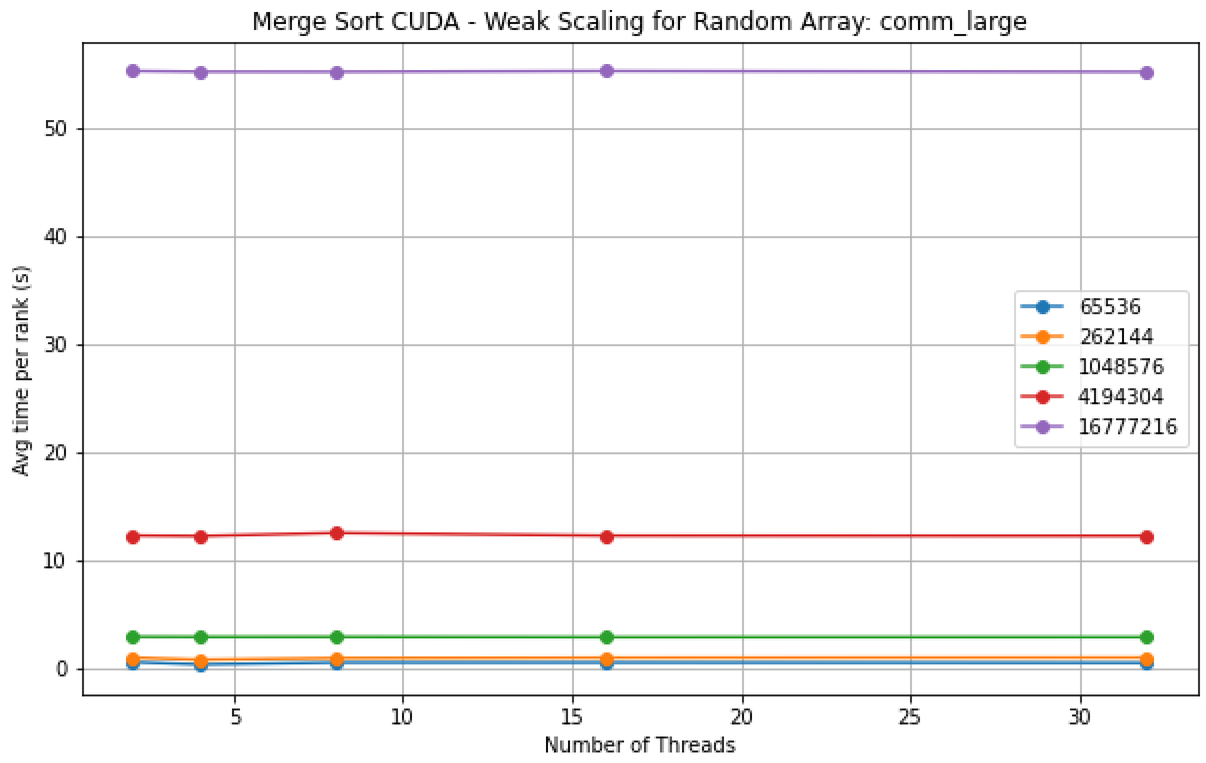
In [50]:
```
main = ctkrand.dataframe.loc["main"]
comm = ctkrand.dataframe.loc["comm"]
comm_large = ctkrand.dataframe.loc["comm_large"]
comp = ctkrand.dataframe.loc["comp"]
comp_large = ctkrand.dataframe.loc["comp_large"]
correctness_check = ctkrand.dataframe.loc["correctness_check"]
data_init = ctkrand.dataframe.loc["data_init"]
```

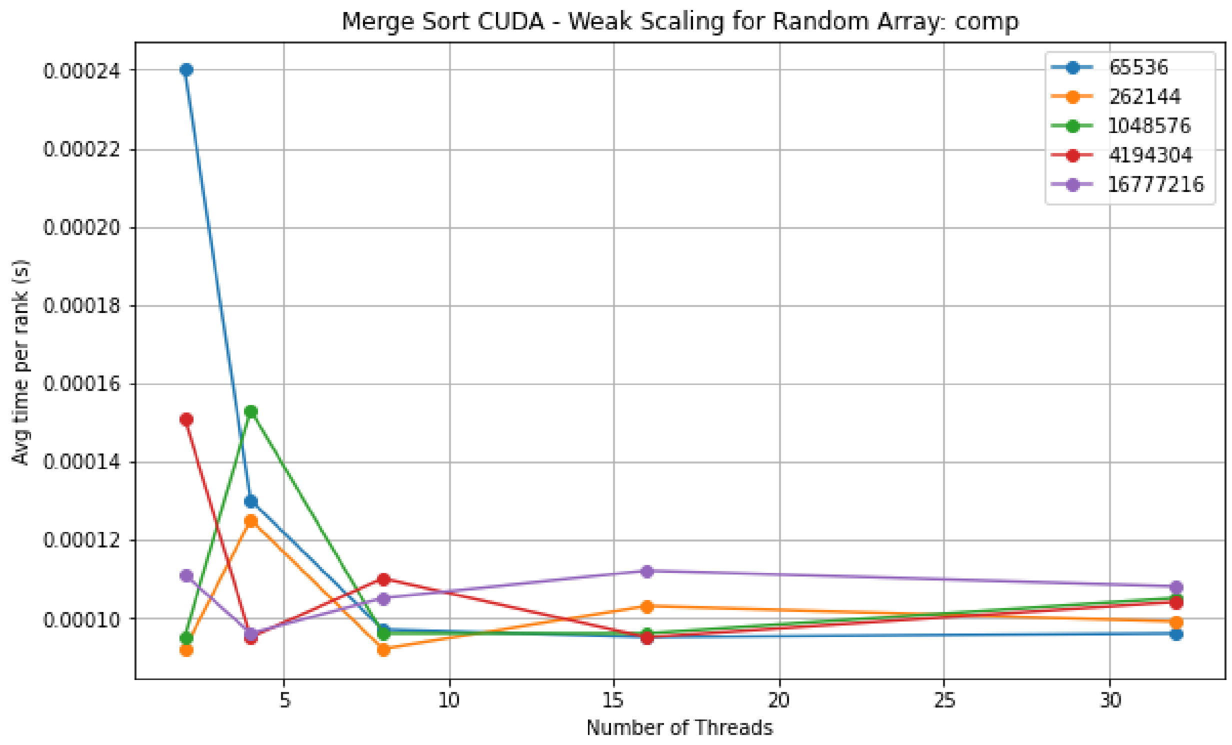
In [51]:
```
regions = [main, comm, comm_large, comp, comp_large, correctness_check, data_init]
names = ["main", "comm", "comm_large", "comp", "comp_large", "correctness_check", "dat
```

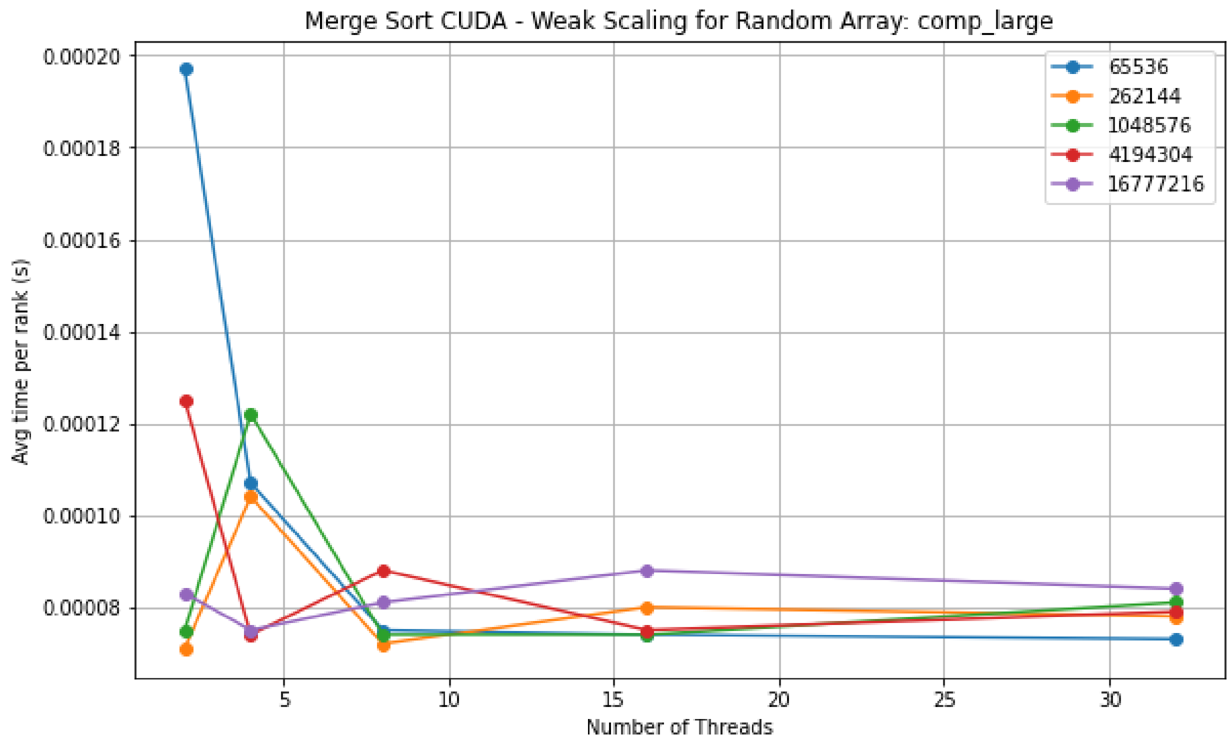
In [52]:
```
for region, name in zip(regions, names):
    plt.figure(figsize=(10, 6))  # Adjust the figure size if needed
    legend_labels = []
    for column in region.columns:
        first_index = column[0]  # Extract the first index
        legend_labels.append(first_index)
        plt.plot(region.index, region.xs(column, axis=1), marker='o', label=column)

    plt.xlabel('Number of Threads')
    plt.ylabel('Avg time per rank (s)')
    plt.title(f'Merge Sort CUDA - Weak Scaling for Random Array: {name}')
    plt.legend(legend_labels)
    plt.grid(True)
    plt.show()
```

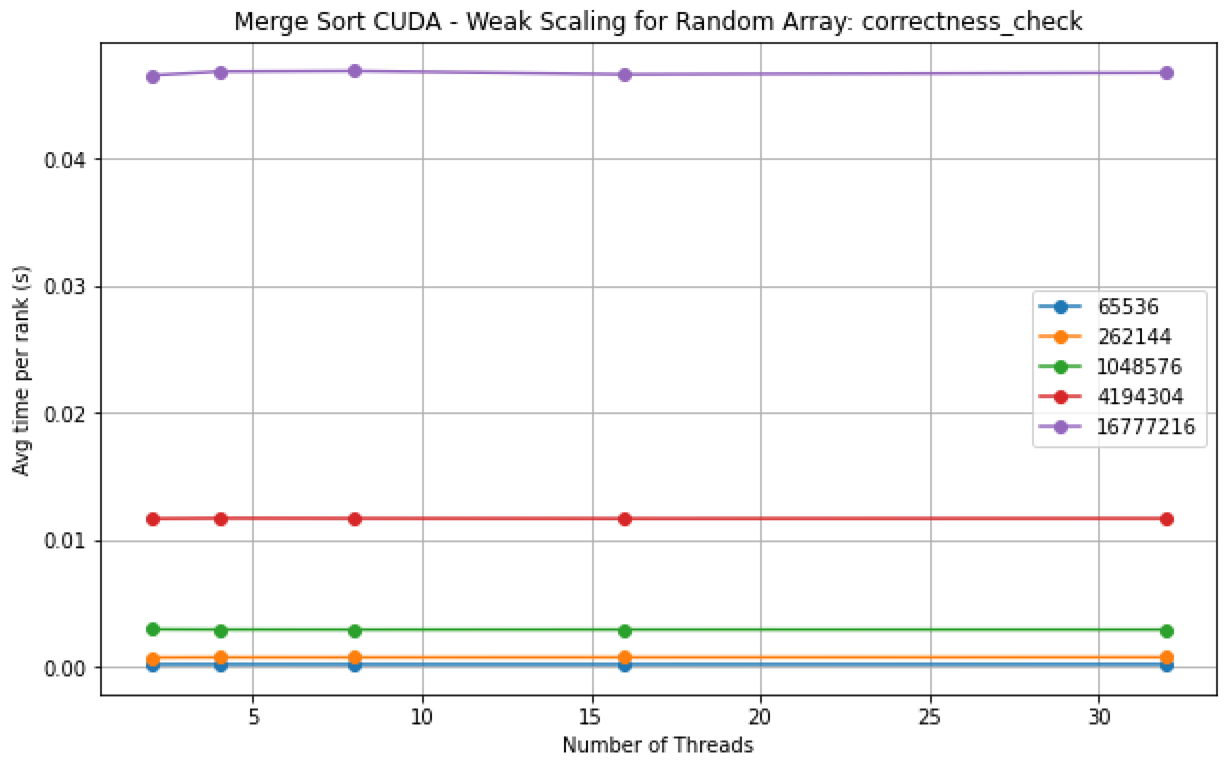

Merge Sort CUDA - Weak Scaling for Random Array: main
Avg time per rank (s)
Number of Threads
0
10
20
30
40
50
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216

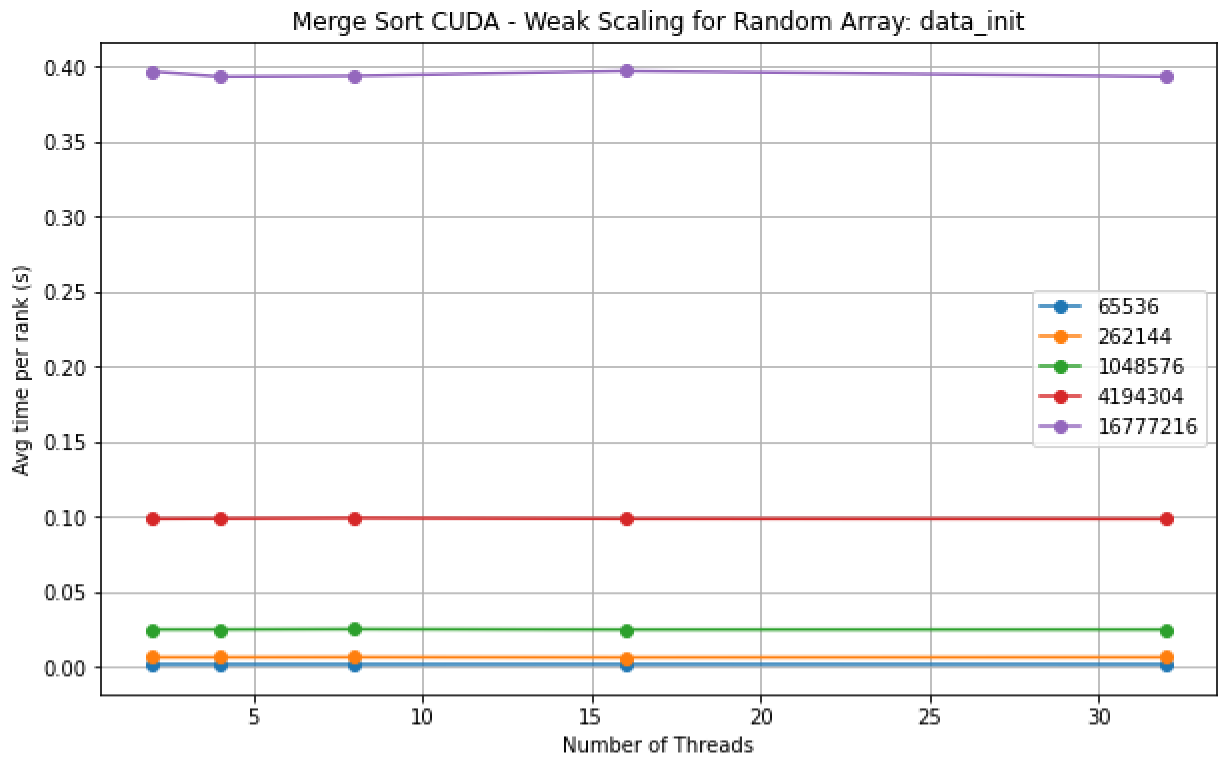

Merge Sort CUDA - Weak Scaling for Random Array: comm
Avg time per rank (s)
Number of Threads
0
10
20
30
40
50
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for Random Array: comm_large
Avg time per rank (s)
Number of Threads
0
10
20
30
40
50
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for Random Array: comp
Avg time per rank (s)
Number of Threads
0.00010
0.00012
0.00014
0.00016
0.00018
0.00020
0.00022
0.00024
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for Random Array: comp_large
Avg time per rank (s)
Number of Threads
0.00020
0.00018
0.00016
0.00014
0.00012
0.00010
0.00008
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for Random Array: correctness_check
Avg time per rank (s)
Number of Threads
0.04
0.03
0.02
0.01
0.00
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216



Sorted

In [53]:
```
tksorted = th.Thicket.from_caliperreader(glob("cali_data_cuda/*-1.cali"))
tksorted.dataframe = tksorted.dataframe.drop(["nid", "spot.channel", "Total time", "Mi
                                             "Avg GPU time/rank", "Min GPU time/rank", "M
gbsorted = tksorted.groupby("InputSize")

ctksorted = th.Thicket.concat_thickets(
    thickets=list(gbsorted.values()),
    headers=list(gbsorted.keys()),
    axis="columns",
    metadata_key="num_threads"
)
```

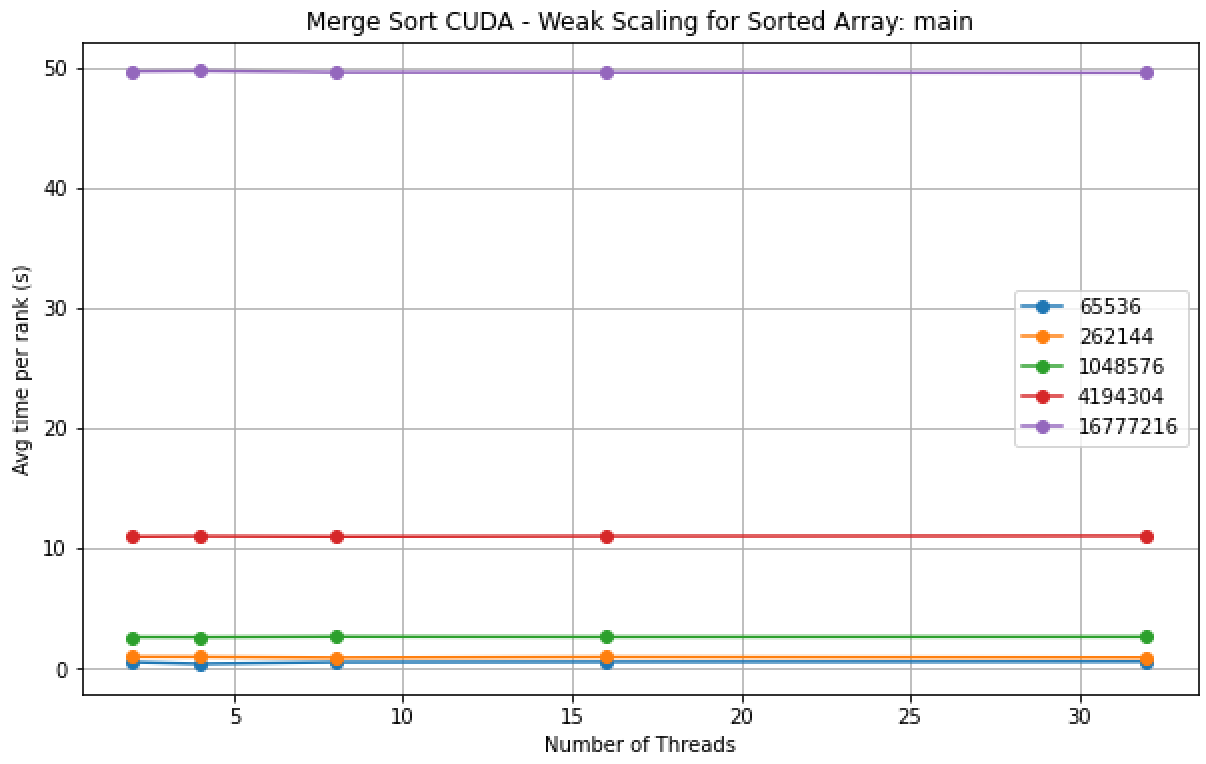
```
5  thickets created...
{65536: <thicket.thicket.Thicket object at 0x2ad85bdad430>, 262144: <thicket.thicket.
Thicket object at 0x2ad85bcbe6a0>, 1048576: <thicket.thicket.Thicket object at 0x2ad8
3659caf0>, 4194304: <thicket.thicket.Thicket object at 0x2ad85bef8c10>, 16777216: <th
icket.thicket.Thicket object at 0x2ad85bdf2a00>}
```

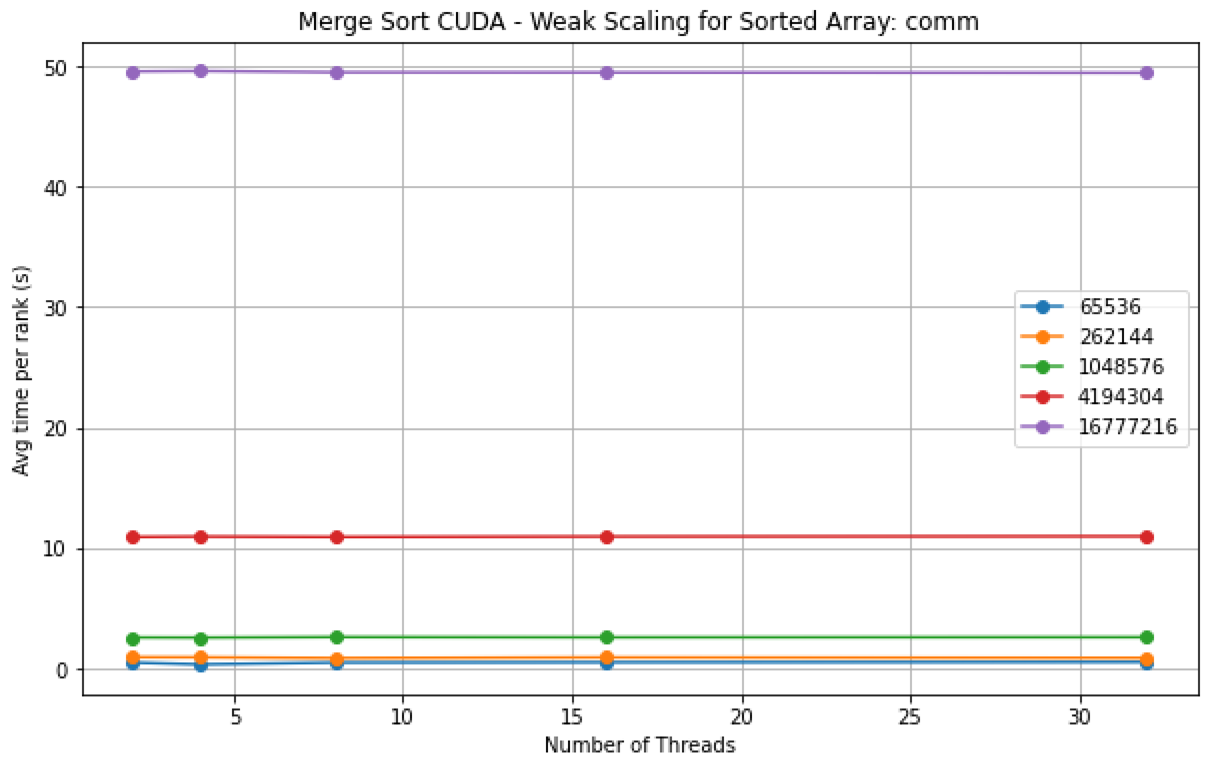
In [54]:
```
ctksorted.dataframe = ctksorted.dataframe.reset_index().drop(("node"), axis=1)
ctksorted.dataframe = ctksorted.dataframe.rename({("name", ""): "name", ("num_threads'

main = ctksorted.dataframe.loc["main"]
comm = ctksorted.dataframe.loc["comm"]
comm_large = ctksorted.dataframe.loc["comm_large"]
comp = ctksorted.dataframe.loc["comp"]
comp_large = ctksorted.dataframe.loc["comp_large"]
correctness_check = ctksorted.dataframe.loc["correctness_check"]
data_init = ctksorted.dataframe.loc["data_init"]
```

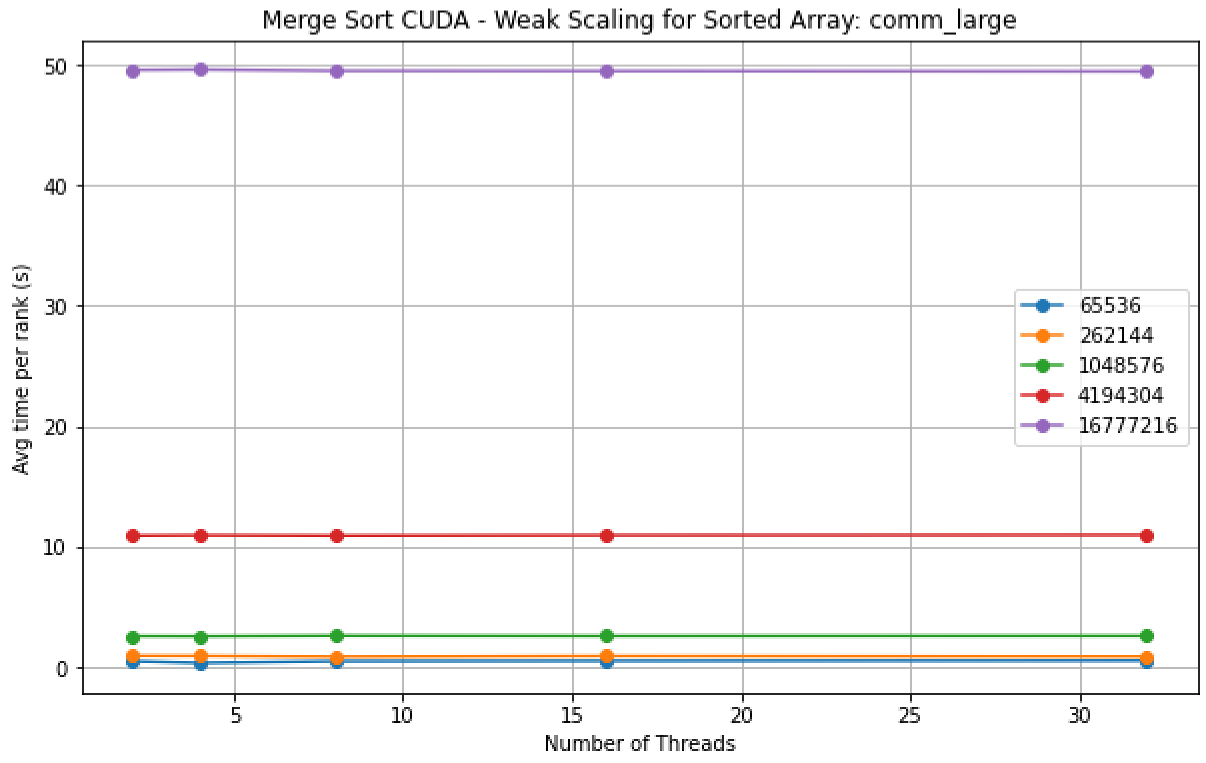
```
<ipython-input-54-36f24b98dd61>:1: PerformanceWarning: dropping on a non-lexsorted mu
lti-index without a level parameter may impact performance.
  ctksorted.dataframe = ctksorted.dataframe.reset_index().drop(("node"), axis=1)
```

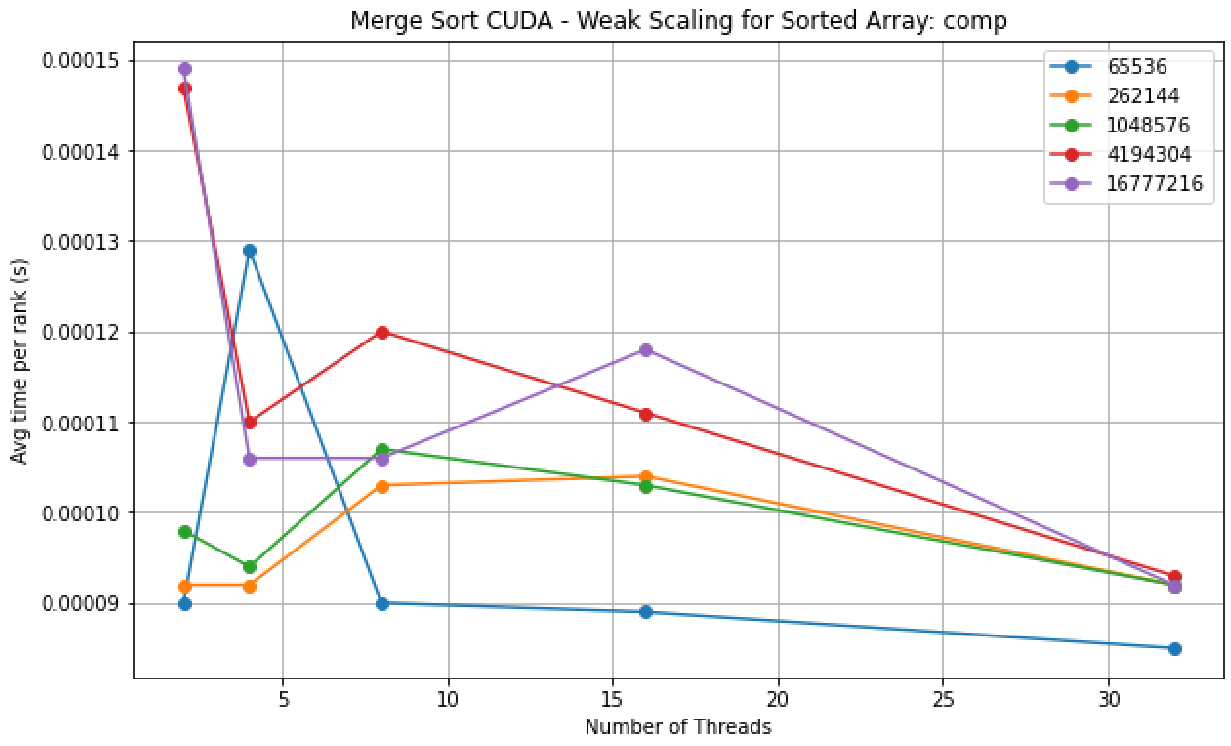
In [55]:
```python
regions = [main, comm, comm_large, comp, comp_large, correctness_check, data_init]
names = ["main", "comm", "comm_large", "comp", "comp_large", "correctness_check", "dat
```

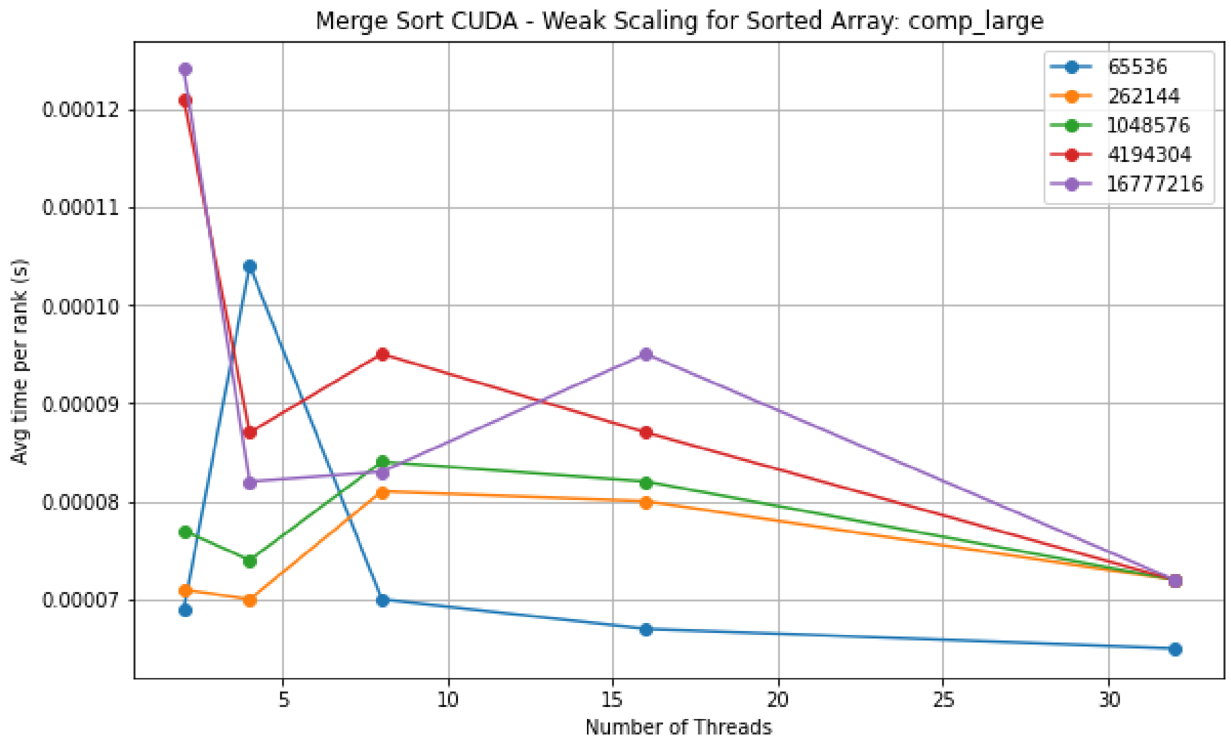
In [56]:
```python
for region, name in zip(regions, names):
    plt.figure(figsize=(10, 6))  # Adjust the figure size if needed
    legend_labels = []
    for column in region.columns:
        first_index = column[0]  # Extract the first index
        legend_labels.append(first_index)
        plt.plot(region.index, region.xs(column, axis=1), marker='o', label=column)

    plt.xlabel('Number of Threads')
    plt.ylabel('Avg time per rank (s)')
    plt.title(f'Merge Sort CUDA - Weak Scaling for Sorted Array: {name}')
    plt.legend(legend_labels)
    plt.grid(True)
    plt.show()
```

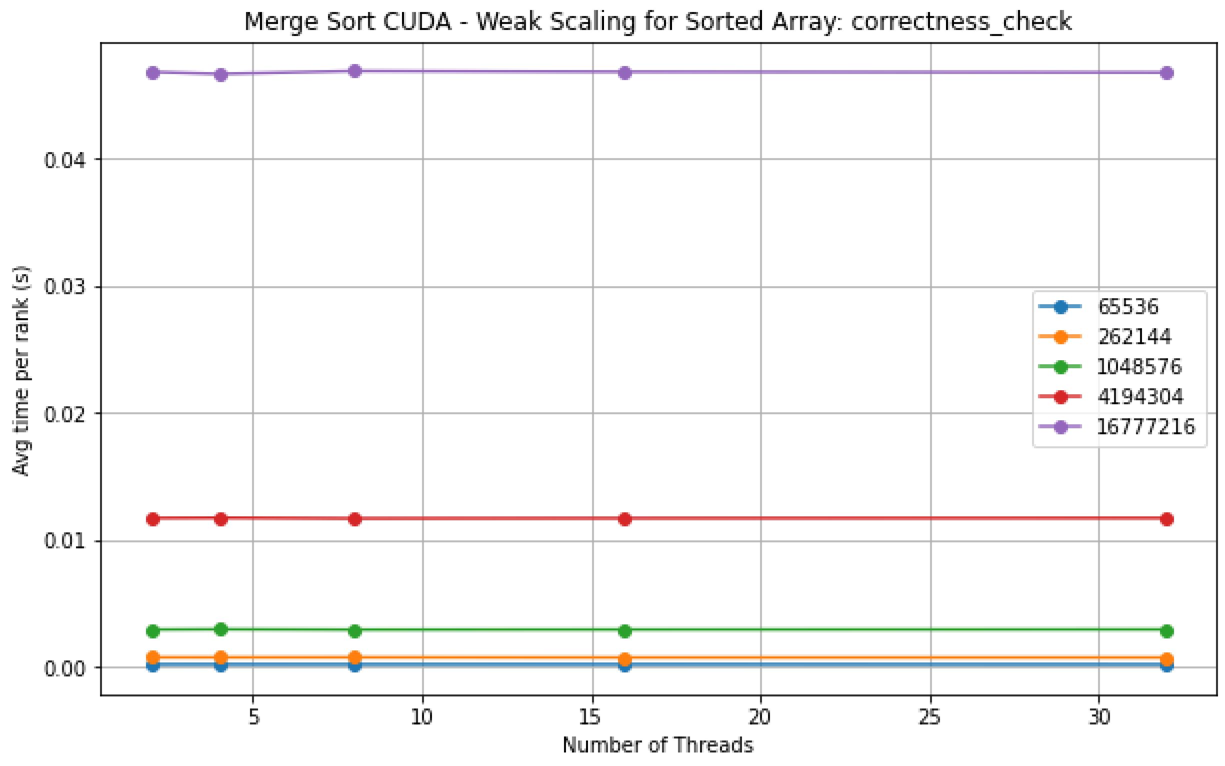


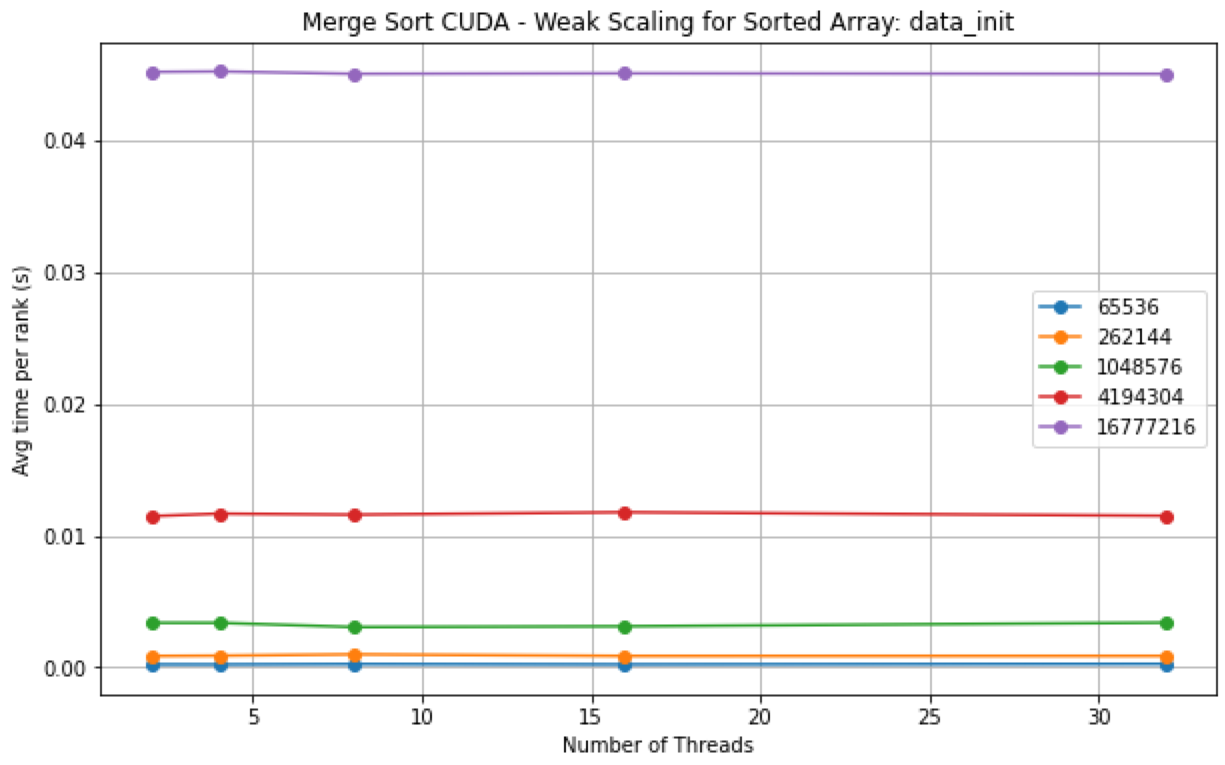

Merge Sort CUDA - Weak Scaling for Sorted Array: comm
Avg time per rank (s)
Number of Threads
65536
262144
1048576
4194304
16777216
0
10
20
30
40
50
5
10
15
20
25
30


Merge Sort CUDA - Weak Scaling for Sorted Array: comm_large
Avg time per rank (s)
Number of Threads
65536
262144
1048576
4194304
16777216
0
10
20
30
40
50
5
10
15
20
25
30


Merge Sort CUDA - Weak Scaling for Sorted Array: comp
Avg time per rank (s)
Number of Threads
0.00015
0.00014
0.00013
0.00012
0.00011
0.00010
0.00009
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for Sorted Array: comp_large
Avg time per rank (s)
Number of Threads
0.00012
0.00011
0.00010
0.00009
0.00008
0.00007
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216





Reverse Sorted

In [57]:
```
tkrev = th.Thicket.from_caliperreader(glob("cali_data_cuda/*-2.cali"))
tkrev.dataframe = tkrev.dataframe.drop(["nid", "spot.channel", "Total time", "Min time
                                         "Avg GPU time/rank", "Min GPU time/rank", "M

gbrev = tkrev.groupby("InputSize")

ctkrev = th.Thicket.concat_thickets(
    thickets=list(gbrev.values()),
    headers=list(gbrev.keys()),
    axis="columns",
    metadata_key="num_threads"
)
```

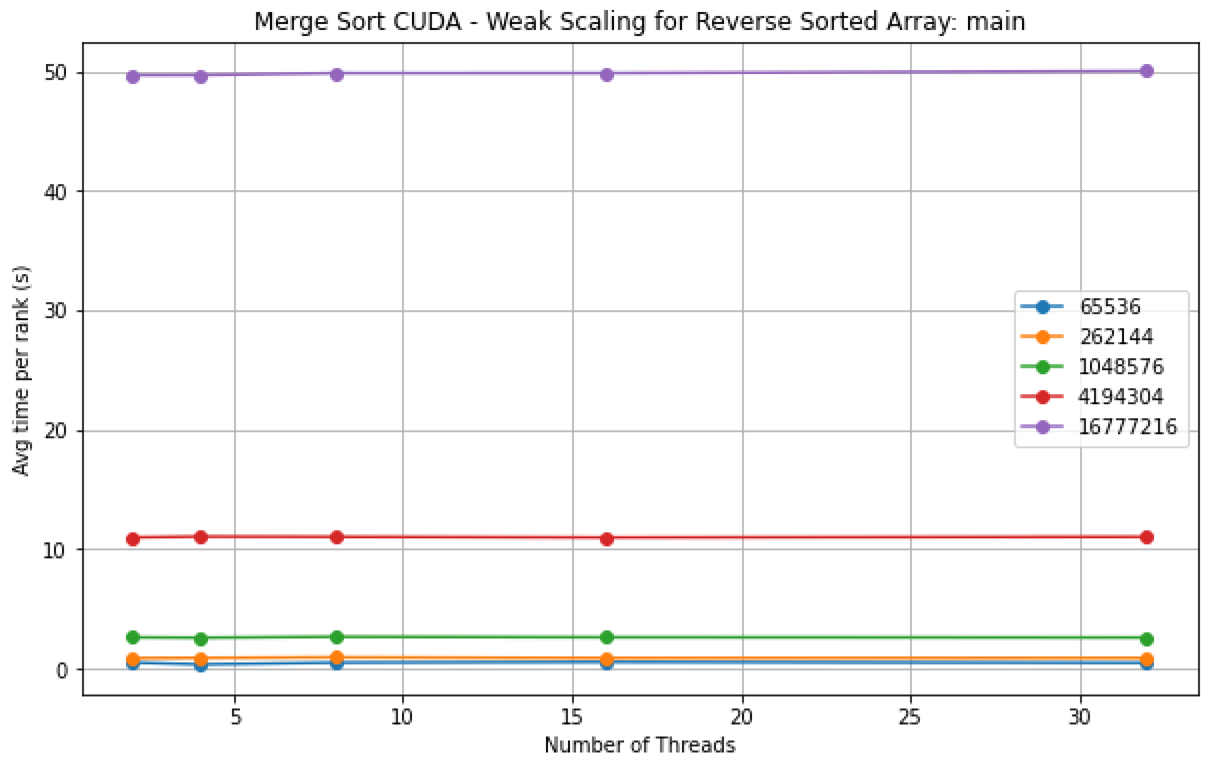
```
5  thickets created...
{65536: <thicket.thicket.Thicket object at 0x2ad85bee52e0>, 262144: <thicket.thicket.
Thicket object at 0x2ad85bd1a940>, 1048576: <thicket.thicket.Thicket object at 0x2ad8
5b1ffd90>, 4194304: <thicket.thicket.Thicket object at 0x2ad85bddb490>, 16777216: <th
icket.thicket.Thicket object at 0x2ad85b194700>}
```

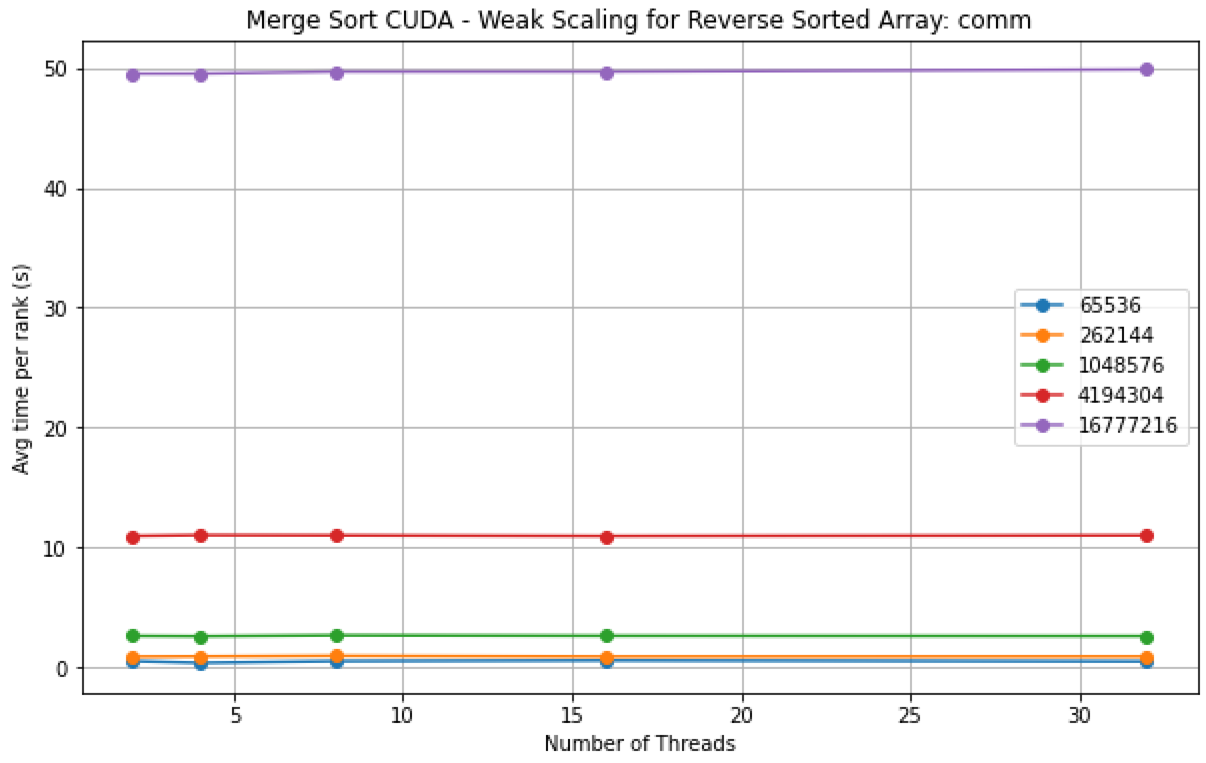
In [58]:
```
ctkrev.dataframe = ctkrev.dataframe.reset_index().drop(("node"), axis=1)
ctkrev.dataframe = ctkrev.dataframe.rename({("name", ""): "name", ("num_threads", ""):

main = ctkrev.dataframe.loc["main"]
comm = ctkrev.dataframe.loc["comm"]
comm_large = ctkrev.dataframe.loc["comm_large"]
comp = ctkrev.dataframe.loc["comp"]
comp_large = ctkrev.dataframe.loc["comp_large"]
correctness_check = ctkrev.dataframe.loc["correctness_check"]
data_init = ctkrev.dataframe.loc["data_init"]
```

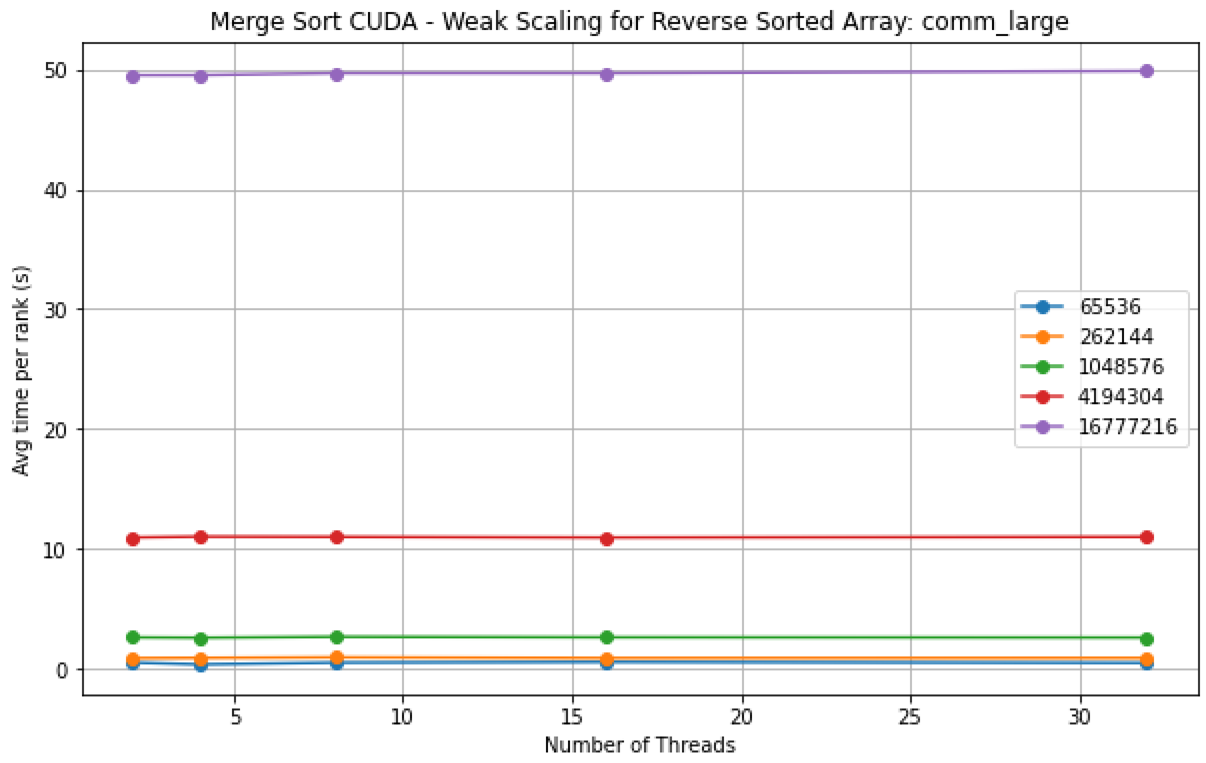
```
<ipython-input-58-ad95fc67489a>:1: PerformanceWarning: dropping on a non-lexsorted mu
lti-index without a level parameter may impact performance.
  ctkrev.dataframe = ctkrev.dataframe.reset_index().drop(("node"), axis=1)
```

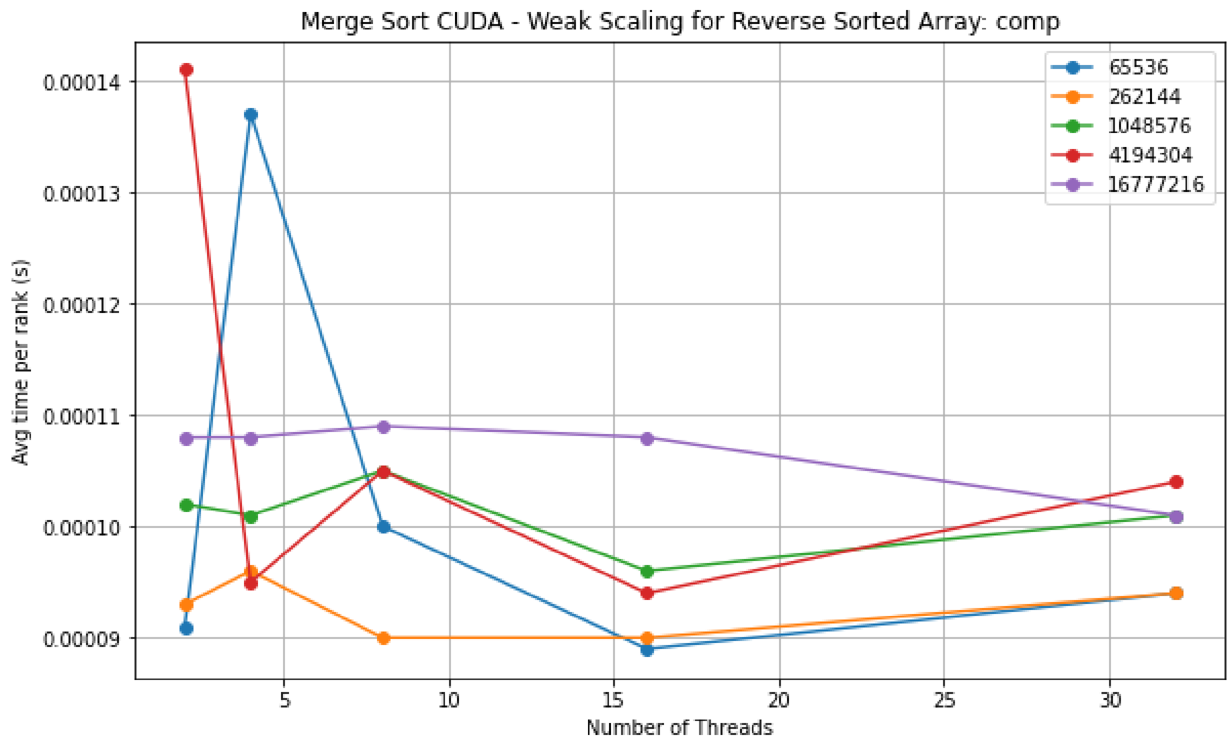
In [59]:
```
regions = [main, comm, comm_large, comp, comp_large, correctness_check, data_init]
names = ["main", "comm", "comm_large", "comp", "comp_large", "correctness_check", "dat
```

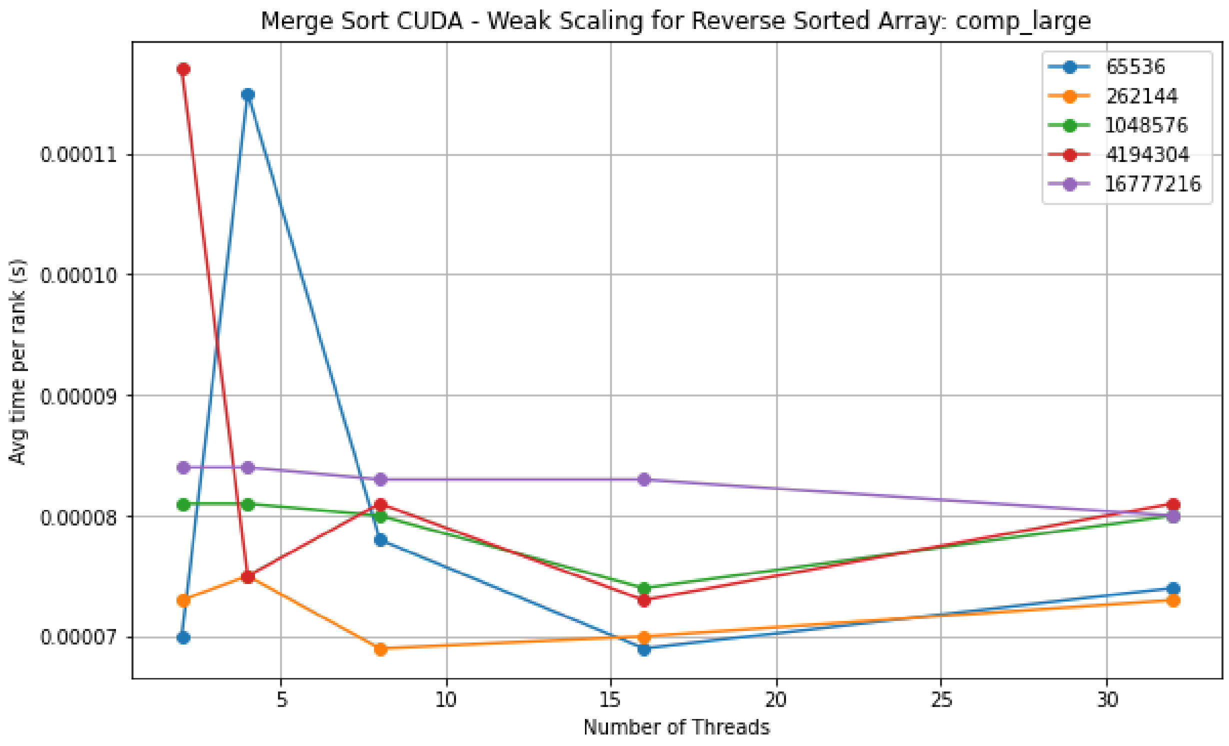
In [60]:
```
for region, name in zip(regions, names):
    plt.figure(figsize=(10, 6))  # Adjust the figure size if needed
    legend_labels = []
    for column in region.columns:
        first_index = column[0]  # Extract the first index
        legend_labels.append(first_index)
        plt.plot(region.index, region.xs(column, axis=1), marker='o', label=column)

    plt.xlabel('Number of Threads')
    plt.ylabel('Avg time per rank (s)')
    plt.title(f'Merge Sort CUDA - Weak Scaling for Reverse Sorted Array: {name}')
    plt.legend(legend_labels)
    plt.grid(True)
    plt.show()
```

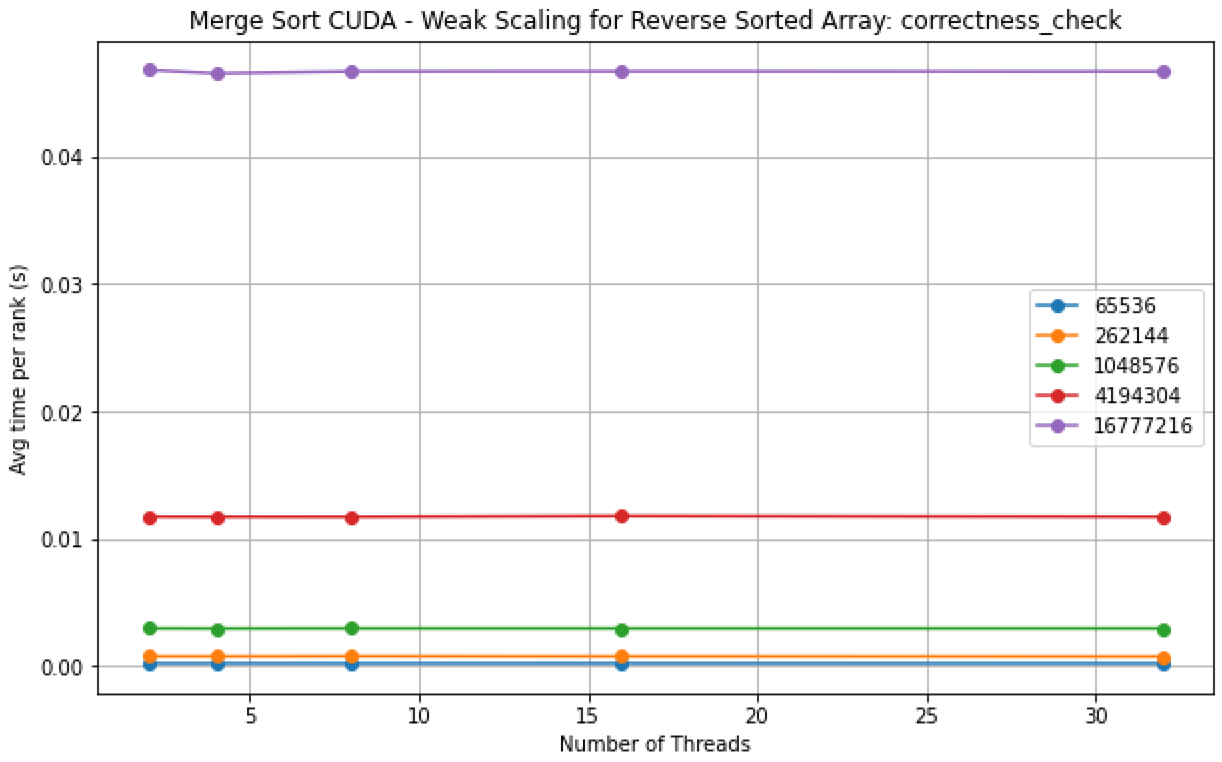

Merge Sort CUDA - Weak Scaling for Reverse Sorted Array: main
Avg time per rank (s)
Number of Threads
65536
262144
1048576
4194304
16777216
0
10
20
30
40
50
5
10
15
20
25
30

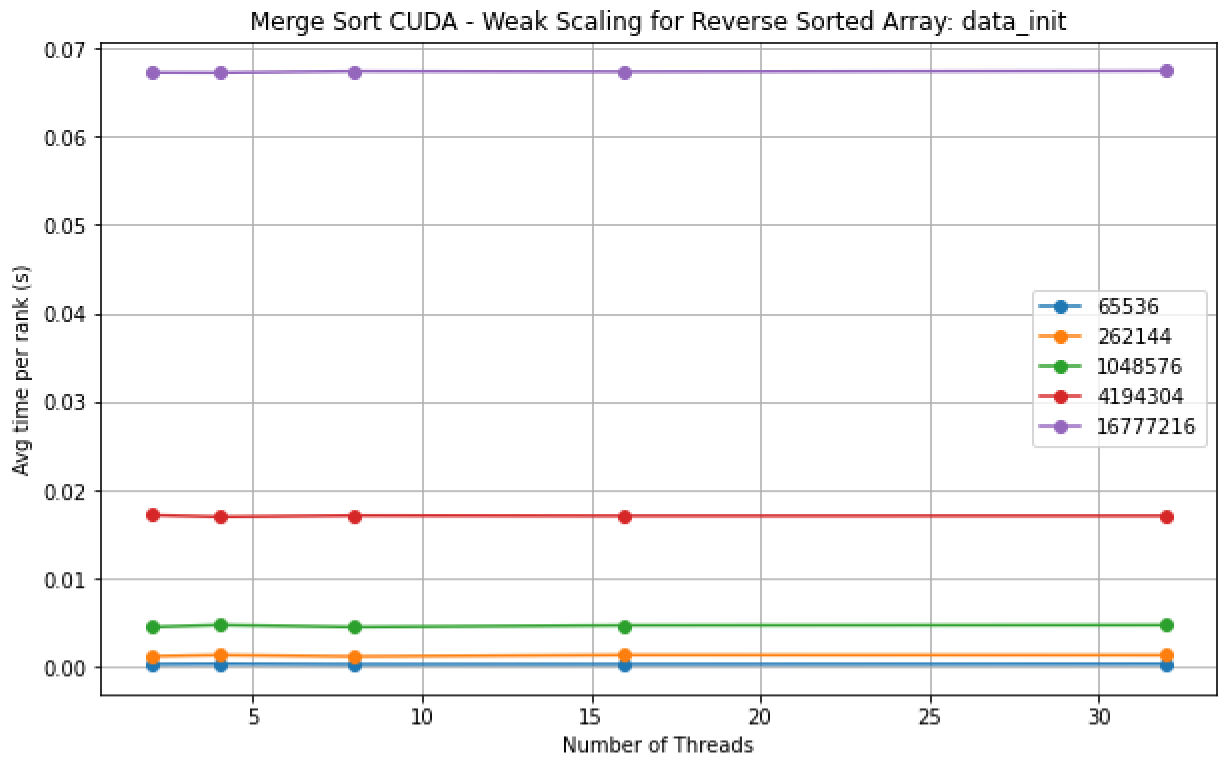

Merge Sort CUDA - Weak Scaling for Reverse Sorted Array: comm
Avg time per rank (s)
Number of Threads
65536
262144
1048576
4194304
16777216
0
10
20
30
40
50
5
10
15
20
25
30


Merge Sort CUDA - Weak Scaling for Reverse Sorted Array: comm_large
Avg time per rank (s)
Number of Threads
0
10
20
30
40
50
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for Reverse Sorted Array: comp
Avg time per rank (s)
Number of Threads
0.00009
0.00010
0.00011
0.00012
0.00013
0.00014
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for Reverse Sorted Array: comp_large
Avg time per rank (s)
0.00011
0.00010
0.00009
0.00008
0.00007
5
10
15
20
25
30
Number of Threads
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for Reverse Sorted Array: correctness_check
Avg time per rank (s)
0.04
0.03
0.02
0.01
0.00
5
10
15
20
25
30
Number of Threads
65536
262144
1048576
4194304
16777216



1% Perturbed

In [61]:
```
tk1 = th.Thicket.from_caliperreader(glob("cali_data_cuda/*-2.cali"))
tk1.dataframe = tk1.dataframe.drop(["nid", "spot.channel", "Total time", "Min time/rar
                                    "Avg GPU time/rank", "Min GPU time/rank", "M

gb1 = tk1.groupby("InputSize")

ctk1 = th.Thicket.concat_thickets(
    thickets=list(gb1.values()),
    headers=list(gb1.keys()),
    axis="columns",
    metadata_key="num_threads"
)
```

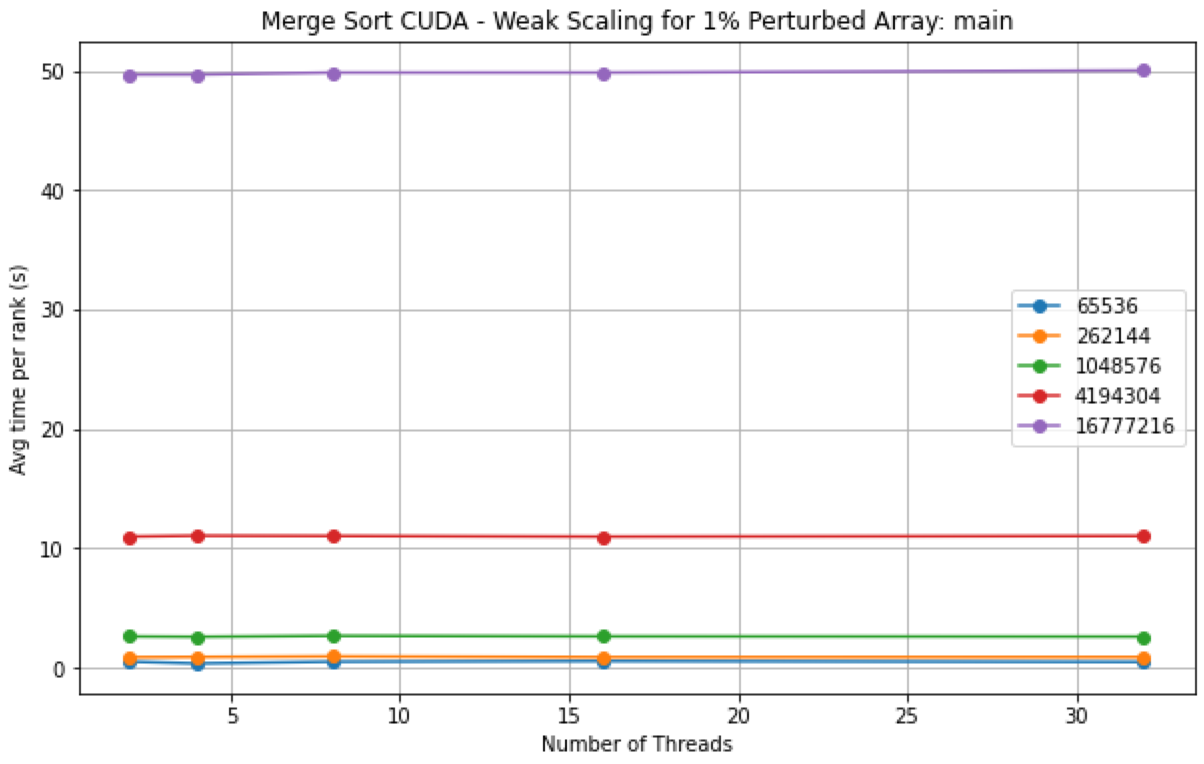
```
5  thickets created...
{65536: <thicket.thicket.Thicket object at 0x2ad85bf18ac0>, 262144: <thicket.thicket.
Thicket object at 0x2ad85ae4c190>, 1048576: <thicket.thicket.Thicket object at 0x2ad8
5bce23a0>, 4194304: <thicket.thicket.Thicket object at 0x2ad85b239850>, 16777216: <th
icket.thicket.Thicket object at 0x2ad85b2394c0>}
```

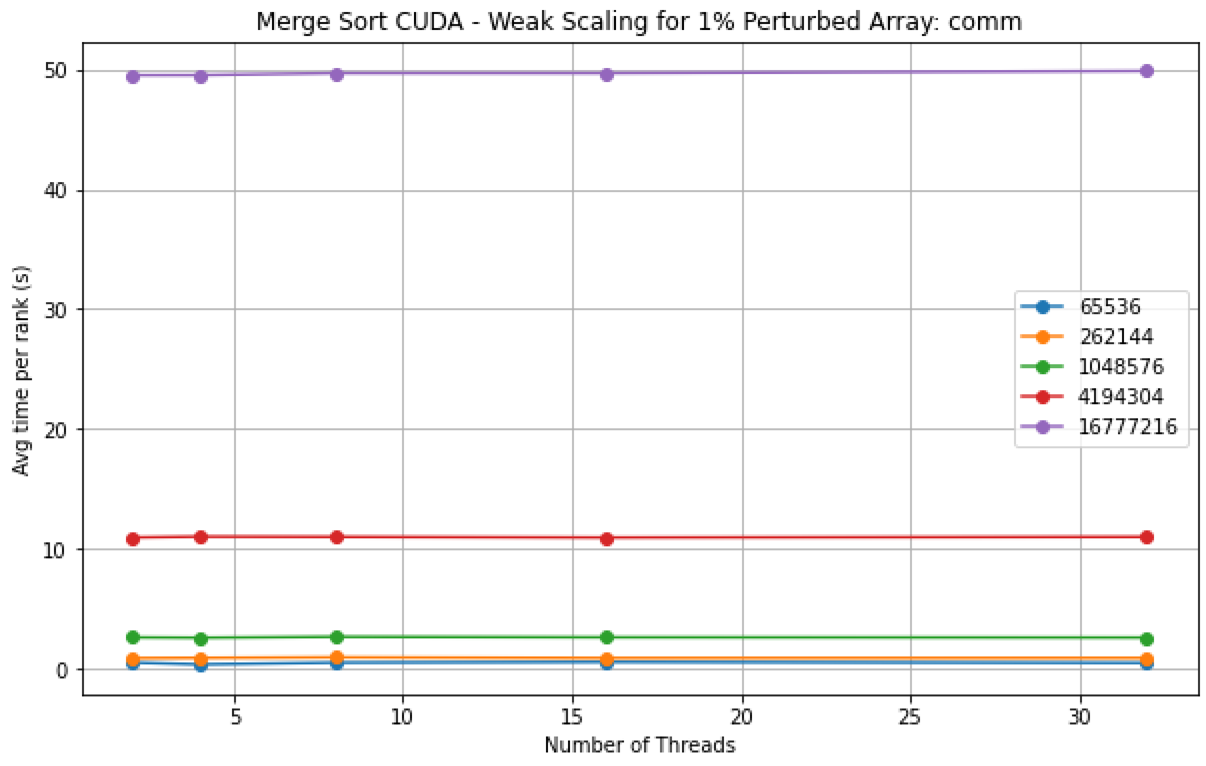
In [62]:
```
ctk1.dataframe = ctk1.dataframe.reset_index().drop(("node"), axis=1)
ctk1.dataframe = ctk1.dataframe.rename({("name", ""): "name", ("num_threads", ""): "nu

main = ctk1.dataframe.loc["main"]
comm = ctk1.dataframe.loc["comm"]
comm_large = ctk1.dataframe.loc["comm_large"]
comp = ctk1.dataframe.loc["comp"]
comp_large = ctk1.dataframe.loc["comp_large"]
correctness_check = ctk1.dataframe.loc["correctness_check"]
data_init = ctk1.dataframe.loc["data_init"]
```

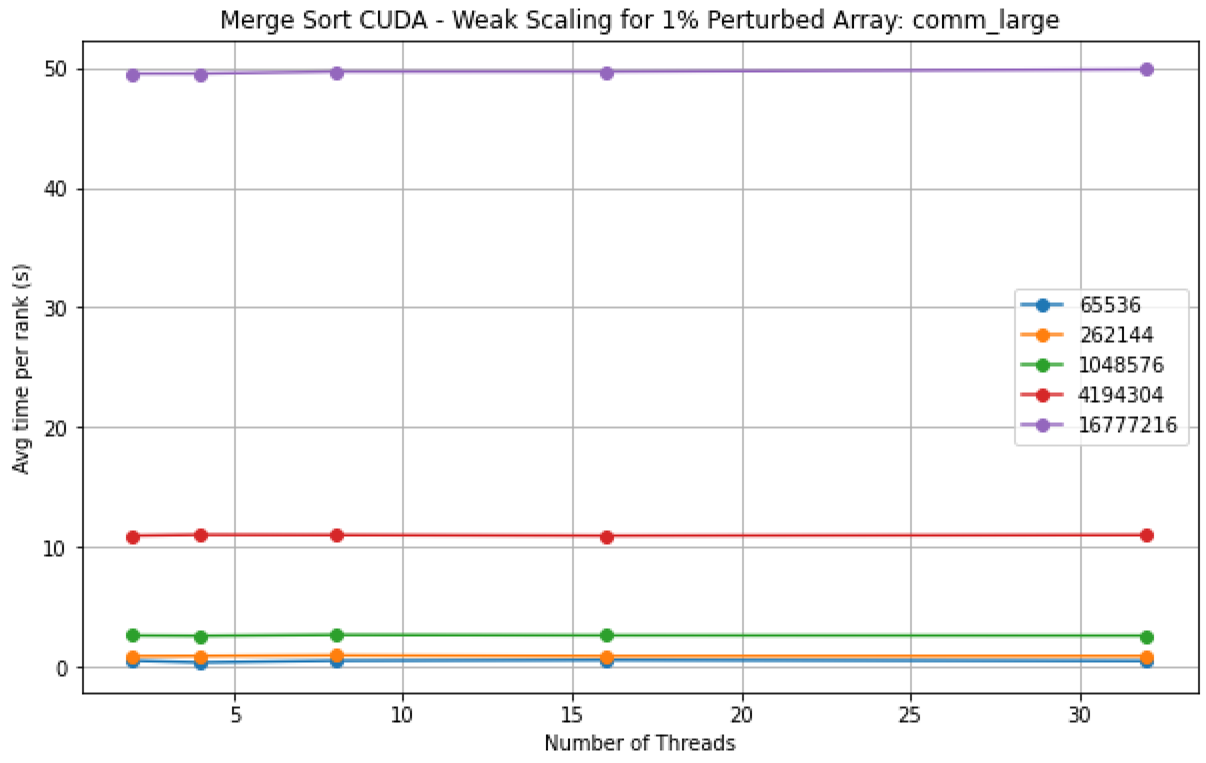
```
<ipython-input-62-b557b8adc448>:1: PerformanceWarning: dropping on a non-lexsorted mu
lti-index without a level parameter may impact performance.
  ctk1.dataframe = ctk1.dataframe.reset_index().drop(("node"), axis=1)
```

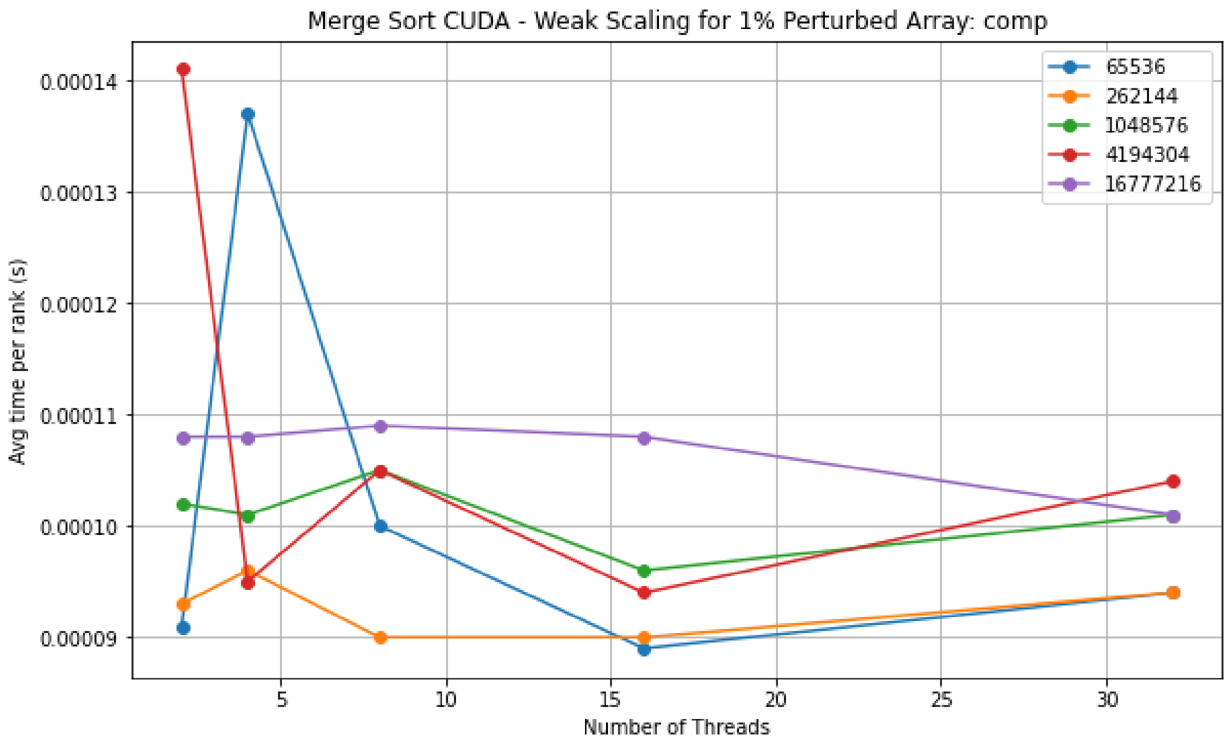
In [63]:
```
regions = [main, comm, comm_large, comp, comp_large, correctness_check, data_init]
names = ["main", "comm", "comm_large", "comp", "comp_large", "correctness_check", "dat
```

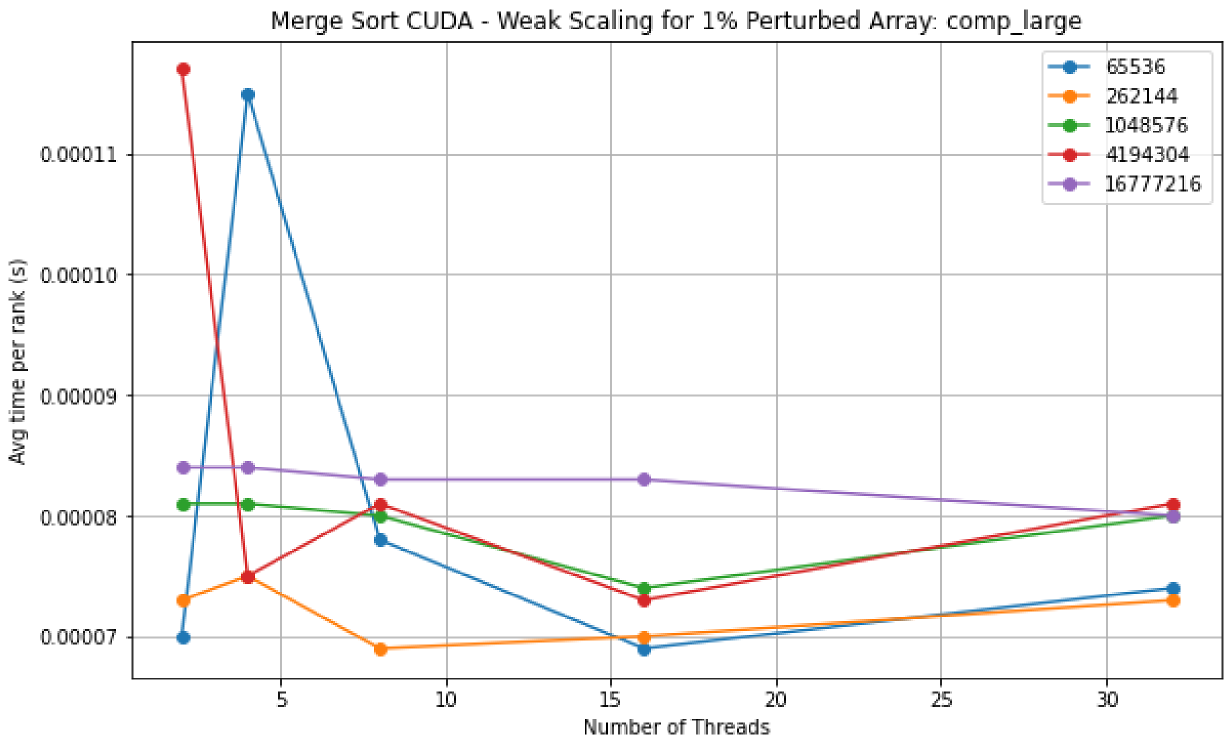
In [64]:
```
for region, name in zip(regions, names):
    plt.figure(figsize=(10, 6))  # Adjust the figure size if needed
    legend_labels = []
    for column in region.columns:
        first_index = column[0]  # Extract the first index
        legend_labels.append(first_index)
        plt.plot(region.index, region.xs(column, axis=1), marker='o', label=column)

    plt.xlabel('Number of Threads')
    plt.ylabel('Avg time per rank (s)')
    plt.title(f'Merge Sort CUDA - Weak Scaling for 1% Perturbed Array: {name}')
    plt.legend(legend_labels)
    plt.grid(True)
    plt.show()
```

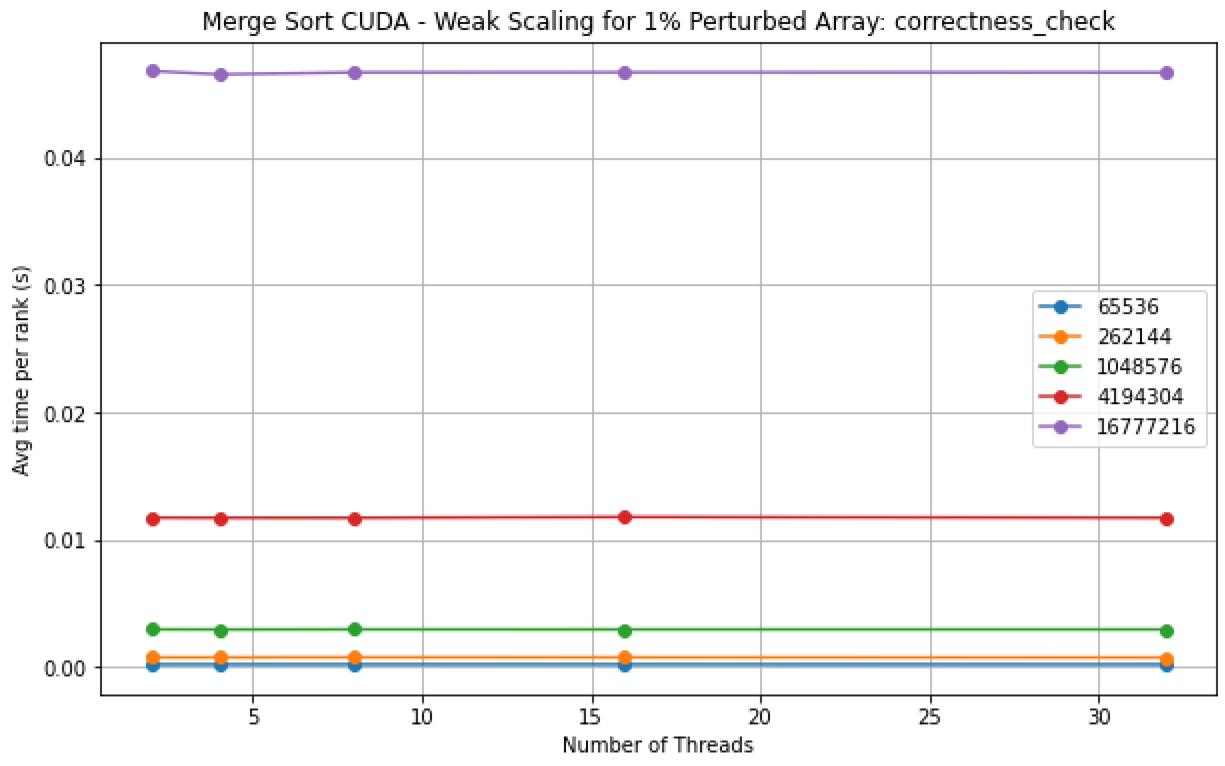


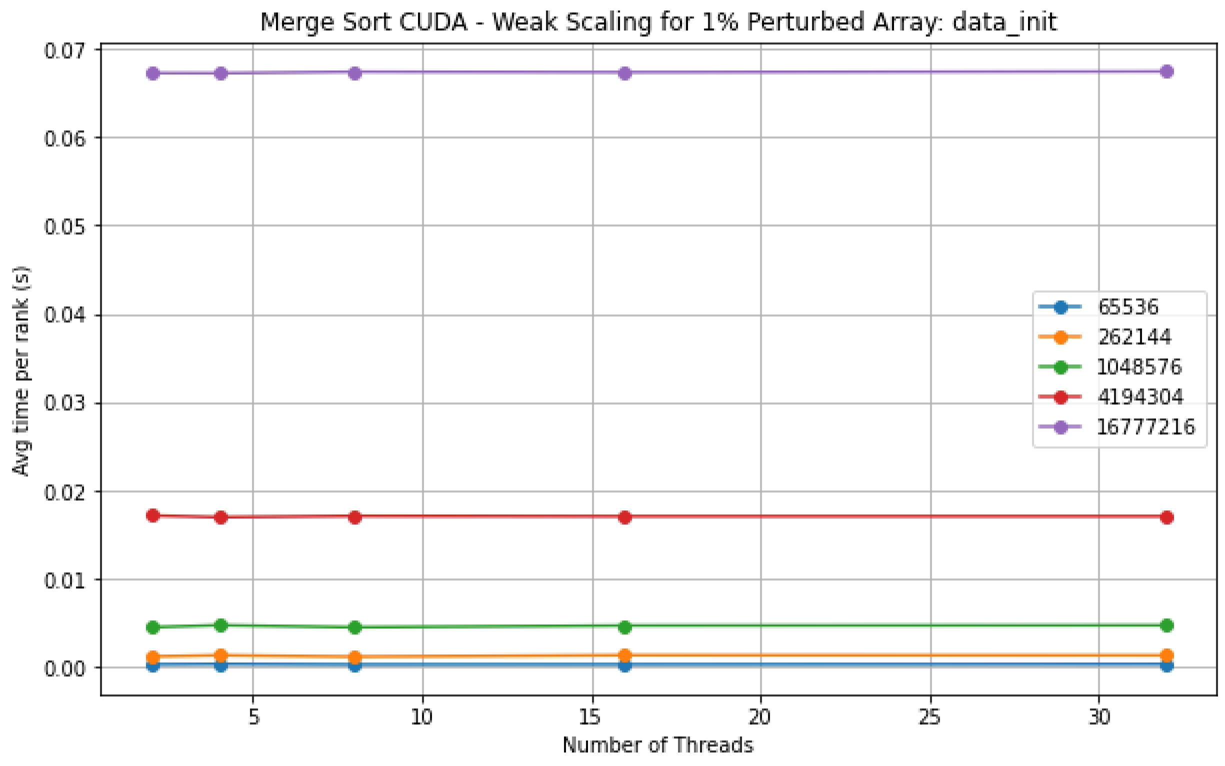

Merge Sort CUDA - Weak Scaling for 1% Perturbed Array: comm
Avg time per rank (s)
Number of Threads
0
10
20
30
40
50
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for 1% Perturbed Array: comm_large
Avg time per rank (s)
Number of Threads
0
10
20
30
40
50
5
10
15
20
25
30
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for 1% Perturbed Array: comp
Avg time per rank (s)
Number of Threads
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for 1% Perturbed Array: comp_large
Avg time per rank (s)
Number of Threads
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for 1% Perturbed Array: correctness_check
Avg time per rank (s)
Number of Threads
65536
262144
1048576
4194304
16777216


Merge Sort CUDA - Weak Scaling for 1% Perturbed Array: data_init
Avg time per rank (s)
Number of Threads
65536
262144
1048576
4194304
16777216

```
In [ ]:
```