

short_sector_state_miners

February 28, 2021

1 Stats for the sectors with less than 400d lifetime, segmented by miner group

1.1 Setting-up

```
[1]: %load_ext autotime  
  
%load_ext autoreload  
  
%autoreload 2
```

time: 20.6 ms (started: 2021-02-26 13:45:01 -03:00)

```
[2]: # External dependences  
import pandas as pd  
import numpy as np  
import plotly.express as px  
  
# Move path to parent folder  
import sys  
sys.path.insert(1, '../')  
  
import plotly  
plotly.offline.init_notebook_mode()
```

time: 753 ms (started: 2021-02-26 13:45:01 -03:00)

```
[3]: from filecoin_metrics.connection import get_connection, get_connection_string  
conn_string = get_connection_string('../config/sentinel-conn-string.txt')  
connection = get_connection(conn_string)
```

time: 4.48 s (started: 2021-02-26 13:45:02 -03:00)

```
[4]: QUERY = """  
/* Get the last state of the sectors */  
with sector_states as (  
    select  
    msi.*,
```

```

        max(msi.height) over (partition by msi.sector_id, msi.miner_id) as
        ↳max_height
        from miner_sector_infos msi
        where msi.activation_epoch > 0
        and (msi.expiration_epoch - msi.activation_epoch) < 1152000 /* Get
        ↳sectors with less than 400d lifetime */
        and msi.expiration_epoch > msi.height /* Get only active sectors */
        order by max_height
    )
    select
    ss.miner_id,
    count(*) as sector_count,
    sum(ss.initial_pledge::numeric) / 1e18 as initial_pledge_in_fil,
    count(*) * 32 as network_power_in_gb,
    date_trunc('WEEK', to_timestamp(height_to_unix(ss.activation_epoch))) as
    ↳activation_date,
    date_trunc('WEEK', to_timestamp(height_to_unix(ss.expiration_epoch))) as
    ↳expiration_date
    from sector_states as ss
    where ss.max_height = ss.height /* get the last state of the info */
    group by activation_date, expiration_date, miner_id
    order by activation_date, expiration_date, miner_id
    """"

df = (pd.read_sql(QUERY, connection)
      .assign(network_power_in_pib=lambda df: df.network_power_in_gb / (1024
      ↳** 2))
      .assign(initial_pledge_in_thousand_fil=lambda df: df.
      ↳initial_pledge_in_fil / 1000))

```

time: 1min 20s (started: 2021-02-26 13:45:06 -03:00)

```

[5]: import json

path = '../config/miners.json'

with open(path, 'r') as fid:
    miners_map = json.load(fid)

miner_id_map = {miner_id: miner
                 for miner, miner_ids
                 in miners_map.items()
                 for miner_id in miner_ids}

df = df.assign(miner=df.miner_id.map(miner_id_map).fillna('None'))

```

time: 12.2 ms (started: 2021-02-26 13:46:26 -03:00)

1.2 Visualizations

```
[6]: print("Basic stats")
print("----")
print(f"Total sectors (#): {df.sector_count.sum()}")
print(f"Raw bytes power (PiB): {df.network_power_in_gb.sum() / (1024 ** 2) :.
↪3g}")
print(f"Initial pledge (FIL): {df.initial_pledge_in_fil.sum()}")
print("----")
```

Basic stats

Total sectors (#): 15358152

Raw bytes power (PiB): 469

Initial pledge (FIL): 3782633.0368385282

time: 10.2 ms (started: 2021-02-26 13:46:26 -03:00)

```
[7]: def resample_and_bar_plot(df, resample_rule, time_column, value_column, title):
    fig_df = df.groupby('miner').resample(resample_rule, on=time_column,
↪label='left').sum()
    fig = px.bar(fig_df.reset_index(),
                  x=time_column,
                  y=value_column,
                  color='miner',
                  title=title)

    return fig
```

```
def resample_and_bar_plot_relative(df, resample_rule, time_column,
↪value_column, title):
    fig_df = df.groupby('miner').resample(resample_rule, on=time_column,
↪label='left').sum()
    y = fig_df.groupby(time_column).sum()
    fig_df /= y
    fig = px.bar(fig_df.reset_index(),
                  x=time_column,
                  y=value_column,
                  color='miner',
                  title=title)

    return fig
```

time: 8.61 ms (started: 2021-02-26 13:46:26 -03:00)

```
[22]: resample_rule = '1m'
time_column = 'expiration_date'
value_column = 'sector_count'
title = 'Upcoming Sector Expiration Count (#)'
```

```
resample_and_bar_plot(df, resample_rule, time_column, value_column, title).  
    ↪ show()
```

time: 189 ms (started: 2021-02-26 13:47:54 -03:00)

```
[23]: resample_rule = '1m'  
time_column = 'activation_date'  
value_column = 'sector_count'  
title = 'Count of Sector Activation Date (#)'  
resample_and_bar_plot(df, resample_rule, time_column, value_column, title).  
    ↪ show()
```

time: 174 ms (started: 2021-02-26 13:47:55 -03:00)

```
[24]: resample_rule = '7d'  
time_column = 'activation_date'  
value_column = 'sector_count'  
title = 'Count of Sector Activation Date (#)'  
  
groups = ['miner',  
          pd.Grouper(key='activation_date', freq=resample_rule),  
          pd.Grouper(key='expiration_date', freq=resample_rule)]  
fig_df = df.groupby(groups).sum().reset_index()  
px.density_heatmap(fig_df,  
                   x='activation_date',  
                   y='expiration_date',  
                   z='sector_count',  
                   animation_frame='miner')
```

time: 185 ms (started: 2021-02-26 13:48:00 -03:00)

```
[25]: resample_rule = '7d'  
time_column = 'activation_date'  
value_column = 'sector_count'  
title = 'Count of Sector Activation Date (#)'  
  
groups = ['miner',  
          pd.Grouper(key='activation_date', freq=resample_rule),  
          pd.Grouper(key='expiration_date', freq=resample_rule)]  
fig_df = df.groupby(groups).sum().reset_index()  
fig = px.density_contour(fig_df,  
                         x='activation_date',  
                         y='expiration_date',  
                         z='sector_count',  
                         histfunc='sum',
```

```
color='miner')  
fig.show()
```

time: 112 ms (started: 2021-02-26 13:48:00 -03:00)

```
[14]: resample_rule = '1m'  
time_column = 'expiration_date'  
value_column = 'sector_count'  
title = 'Upcoming Sector Expiration Count (%)'  
resample_and_bar_plot_relative(df, resample_rule, time_column, value_column,   
→title).show()
```

time: 103 ms (started: 2021-02-26 13:46:27 -03:00)

```
[15]: resample_rule = '1m'  
time_column = 'activation_date'  
value_column = 'sector_count'  
title = 'Count of Sector Activation Date (%)'  
resample_and_bar_plot_relative(df, resample_rule, time_column, value_column,   
→title).show()
```

time: 108 ms (started: 2021-02-26 13:46:27 -03:00)

```
[16]: resample_rule = '1m'  
time_column = 'expiration_date'  
value_column = ['initial_pledge_in_fil']  
title = 'Sum of Initial Pledge (kFIL) across expiration dates'  
resample_and_bar_plot(df, resample_rule, time_column, value_column, title).  
→show()
```

time: 114 ms (started: 2021-02-26 13:46:28 -03:00)

```
[17]: resample_rule = '1m'  
time_column = 'expiration_date'  
value_column = ['network_power_in_pib']  
title = 'Sum of RB Network Power (PiB) across expiration dates'  
resample_and_bar_plot(df, resample_rule, time_column, value_column, title).  
→show()
```

time: 130 ms (started: 2021-02-26 13:46:28 -03:00)

```
[18]: resample_rule = '1m'  
time_column = 'activation_date'  
value_column = ['initial_pledge_in_thousand_fil']  
title = 'Sum of Initial Pledge (kFIL) across activation dates'  
resample_and_bar_plot(df, resample_rule, time_column, value_column, title).  
→show()
```

time: 115 ms (started: 2021-02-26 13:46:28 -03:00)

```
[19]: resample_rule = '1m'
time_column = 'activation_date'
value_column = ['network_power_in_pib']
title = 'Sum of RB Network Power (PiB) across activation dates'
resample_and_bar_plot(df, resample_rule, time_column, value_column, title).
    ↪ show()
```

time: 112 ms (started: 2021-02-26 13:46:28 -03:00)