FlashV2_Analysis

September 8, 2021

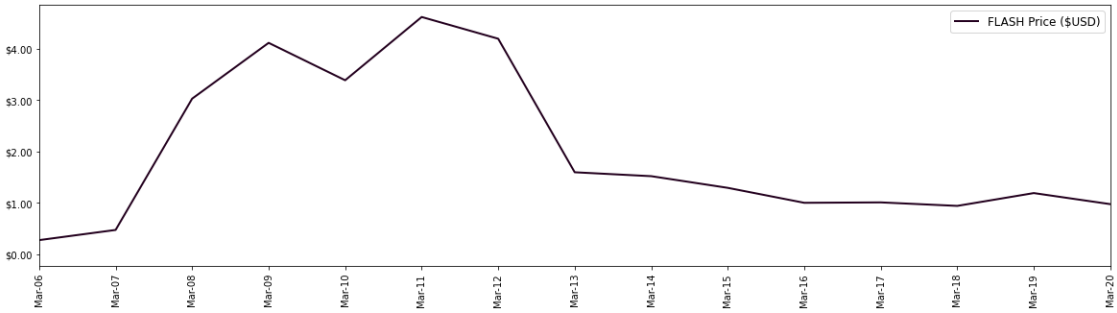
[132]:
```
import sys
sys.path.append('../')
import datetime
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
from matplotlib.dates import DateFormatter
from matplotlib.ticker import FormatStrFormatter, StrMethodFormatter,␣
↪PercentFormatter
import matplotlib as mpl
import numpy as np
import statsmodels.api as sm

from utils import db

conn = db.connect()
pd.set_option('display.max_rows', 100)
# pd.set_option("display.max_rows", None, "display.max_columns", None)

# pd.read_sql("""SELECT * FROM stakes""", conn)
```

1 Flash V2 History

On March 6, 2021, the FLASH/ETH pair pool was created on Uniswap.

[75]:
```
hist = pd.read_sql("""SELECT *
FROM uniswap uni
ORDER BY uni.date;""",conn)

fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

# Plot the Uniswap FLASH/USD price
ax1.plot(hist["date"], hist["price"], color='#23001E', linewidth=2,␣
↪label="FLASH Price ($USD)", zorder=0)
```

```
ax1.zorder = 1
ax1.patch.set_alpha(0)
# ax1.set_ylim([0, 1.5])
ax1.yaxis.set_major_formatter(StrMethodFormatter('${x:,.2f}'))
ax1.xaxis.set_major_locator(mdates.DayLocator(interval=1))
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax1.legend(bbox_to_anchor = (1, 1), prop={'size':12})
ax1.set_xlim([datetime.date(2021,3,6), datetime.date(2021,3,20)])
plt.show()
```

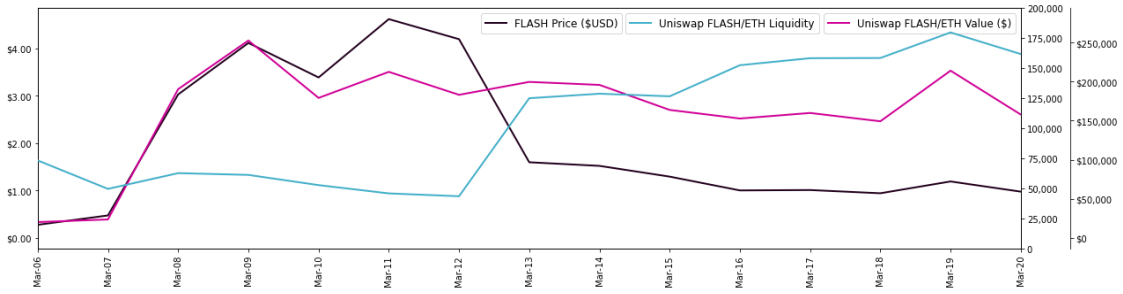


1.1 Initial Price Spike

Due to low liquidity and immediate buying of FLASH, the price quickly piked to over \$4.50 before settling down to around $1.00 as more liquidity was added, stabilizing the price.

[78]:
```
hist = pd.read_sql("""SELECT *
FROM uniswap uni
LEFT JOIN
    (
        SELECT stakedatetime, COUNT(fromaddress) AS stakecount, SUM(amountin)␣
 ↪staketotal
        FROM stakes
        GROUP BY stakedatetime
    ) AS s1
ON s1.stakedatetime=uni.date
LEFT JOIN
    (
        SELECT date AS ethdate, close AS ethclose
        FROM ethpx
    ) AS e1
ON e1.ethdate=uni.date
ORDER BY uni.date;""",conn)

fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
```

```
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

# Plot the Uniswap FLASH/USD price
ax1.plot(hist["date"], hist["price"], color='#23001E', linewidth=2,␣
 ↪label="FLASH Price ($USD)", zorder=0)
ax1.zorder = 1
ax1.patch.set_alpha(0)
# ax1.set_ylim([0, 1.5])
ax1.yaxis.set_major_formatter(StrMethodFormatter('${x:,.2f}'))
ax1.xaxis.set_major_locator(mdates.DayLocator(interval=1))
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax1.legend(bbox_to_anchor = (.6, 1), prop={'size':12})
ax1.set_xlim([datetime.date(2021,3,6), datetime.date(2021,3,20)])

# Plot the Uniswap FLASH/ETH pool historical liquidity value
ax3 = ax1.twinx()
ax3.plot(hist["date"], hist["value"], color='#D10097', linewidth=2,␣
 ↪label="Uniswap FLASH/ETH Value ($)", zorder=10)
ax3.zorder = 2
ax3.patch.set_alpha(0)
ax3.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax3.xaxis.set_major_locator(mdates.DayLocator(interval=1))
ax3.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax3.legend(bbox_to_anchor = (1, 1), prop={'size':11.75})

ax3.spines['right'].set_position(("axes", 1.05))
ax3.spines['right'].set_visible(True)
ax3.yaxis.set_label_position('right')
ax3.yaxis.set_ticks_position('right')

# Plot the Uniswap FLASH/ETH pool historical liquidity quantity
ax4 = ax1.twinx()
ax4.plot(hist["date"], hist["liquidity"], color='#43B0CA', linewidth=2,␣
 ↪label="Uniswap FLASH/ETH Liquidity", zorder=5)
ax4.zorder = 3
ax4.patch.set_alpha(0)
ax4.set_ylim([0, 200000])
ax4.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax4.xaxis.set_major_locator(mdates.DayLocator(interval=1))
ax4.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
# ax4.legend("Liquidity", bbox_to_anchor = (1.05, 0.6))
ax4.legend(bbox_to_anchor = (.8, 1), prop={'size':12})
plt.show()
```



1.2 Flash Staking

On the same day that a lot of liquidity was dumped into the Uniswap FLASH/ETH pool, nearly 80 stakes were executed with over 2 million $FLASH staked.

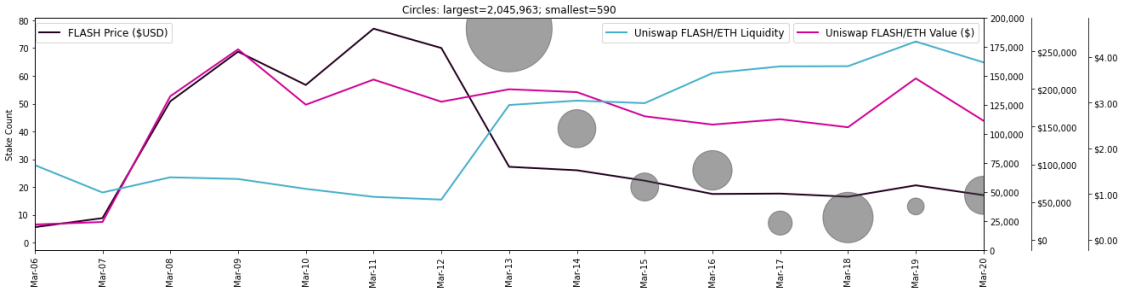
[108]:
```
mar13 = pd.read_sql("""SELECT uni.date, TO_CHAR(uni.liquidity, '999,999,999')␣
 ↪AS "Liquidity", s1.stakecount AS "Stake Count", TO_CHAR(s1.staketotal,␣
 ↪'999,999,999') AS "Total Tokens Staked"
FROM uniswap uni
LEFT JOIN
    (
        SELECT stakedatetime, COUNT(fromaddress) AS stakecount, SUM(amountin)␣
 ↪staketotal
        FROM stakes
        GROUP BY stakedatetime
        ORDER BY stakedatetime DESC
    ) AS s1
ON s1.stakedatetime=uni.date
WHERE uni.date='2021-03-13'::date;""",conn)
mar13 = mar13.transpose()
mar13.columns = mar13.iloc[0]
mar13 = mar13.drop(mar13.iloc[0].index.name)
mar13.columns.name = None
mar13 = mar13.transpose()
display(mar13)
```

	Liquidity	Stake Count	Total Tokens Staked
2021-03-13	125,151	77	2,045,963

[109]:
```
fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)
plt.title("Flash staking activity (circle size: Stake $ Amount)")
```

```
# Plot the stake data
# ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
 ↪color='#cad084', alpha=0.7)
ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
 ↪color='#424242', alpha=0.5, zorder=50) #, label="Number of Stakes")
ax1.zorder = 5
ax1.patch.set_alpha(0)
ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax1.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax1.set_ylabel('Stake Count')
ax1.set_xlim([datetime.date(2021,3,6), datetime.date(2021,3,20)])
totalsfiltered = list(filter(lambda x: pd.isnull(x) != True,␣
 ↪hist["staketotal"]))
plt.title("Circles: largest={:,.0f}; smallest={:,.0f}".
 ↪format(max(totalsfiltered), min(totalsfiltered)))

# Plot the Uniswap FLASH/USD price
ax2 = ax1.twinx()
ax2.plot(hist["date"], hist["price"], color='#23001E', linewidth=2,␣
 ↪label="FLASH Price ($USD)", zorder=0)
ax2.zorder = 1
ax2.patch.set_alpha(0)
# ax2.set_ylim([0, 1.5])
ax2.yaxis.set_major_formatter(StrMethodFormatter('${x:,.2f}'))
ax2.xaxis.set_major_locator(mdates.DayLocator(interval=1))
ax2.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax2.legend(bbox_to_anchor = (.15, 1), prop={'size':12})
ax2.set_xlim([datetime.date(2021,3,6), datetime.date(2021,3,20)])

ax2.spines['right'].set_position(("axes", 1.11))
ax2.spines['right'].set_visible(True)
ax2.yaxis.set_label_position('right')
ax2.yaxis.set_ticks_position('right')

# Plot the Uniswap FLASH/ETH pool historical liquidity value
ax3 = ax1.twinx()
ax3.plot(hist["date"], hist["value"], color='#D10097', linewidth=2,␣
 ↪label="Uniswap FLASH/ETH Value ($)", zorder=10)
ax3.zorder = 2
ax3.patch.set_alpha(0)
ax3.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax3.xaxis.set_major_locator(mdates.DayLocator(interval=1))
ax3.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax3.legend(bbox_to_anchor = (1, 1), prop={'size':11.75})
```

```
ax3.spines['right'].set_position(("axes", 1.05))
ax3.spines['right'].set_visible(True)
ax3.yaxis.set_label_position('right')
ax3.yaxis.set_ticks_position('right')

# Plot the Uniswap FLASH/ETH pool historical liquidity quantity
ax4 = ax1.twinx()
ax4.plot(hist["date"], hist["liquidity"], color='#43B0CA', linewidth=2,␣
 ↪label="Uniswap FLASH/ETH Liquidity", zorder=5)
ax4.zorder = 3
ax4.patch.set_alpha(0)
ax4.set_ylim([0, 200000])
ax4.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax4.xaxis.set_major_locator(mdates.DayLocator(interval=1))
ax4.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
# ax4.legend("Liquidity", bbox_to_anchor = (1.05, 0.6))
ax4.legend(bbox_to_anchor = (.8, 1), prop={'size':12})
plt.show()
```



We can check to see how much overlap exists between accounts that added liquidity to Uniswap and staked on that same day.

[15]:
```
pd.read_sql("""SELECT stakes.fromaddress AS "Account", stakedatetime AS "Date",␣
 ↪TO_CHAR(u2.liquidity, '999,999.00') AS "Liquidity (ETH) Deposited",␣
 ↪TO_CHAR(amountin, '999,999,999') AS "$FLASH staked", totalstakeddays AS␣
 ↪"Days Actually Staked"
FROM stakes
INNER JOIN
    (
        SELECT fromaddress, SUM(value) AS liquidity
        FROM uniswaptx
        WHERE datetime='2021-03-13'::date AND (direction='IN' AND␣
 ↪tokensymbol='WETH')
        GROUP BY fromaddress
    ) AS u2
ON u2.fromaddress=stakes.fromaddress
```

```
WHERE stakedatetime='2021-03-13'::date AND flashversion=2
ORDER BY amountin DESC;""",conn)
```

[15]:

	Account	Date	\
0	0xaaf1eb836c73fe48fe67150bd5eb412deb39de23	2021-03-13	
1	0x109cbb5f91951b29c025294ea2b8678ff1386d0e	2021-03-13	
2	0x588031347beaa0d43978bc8c0094138a67d1a071	2021-03-13	
3	0x588031347beaa0d43978bc8c0094138a67d1a071	2021-03-13	

	Liquidity (ETH) Deposited	$FLASH staked	Days Actually Staked
0	1.45	64,259	30.0
1	.26	5,696	118.0
2	.26	2,504	35.0
3	.26	631	35.0

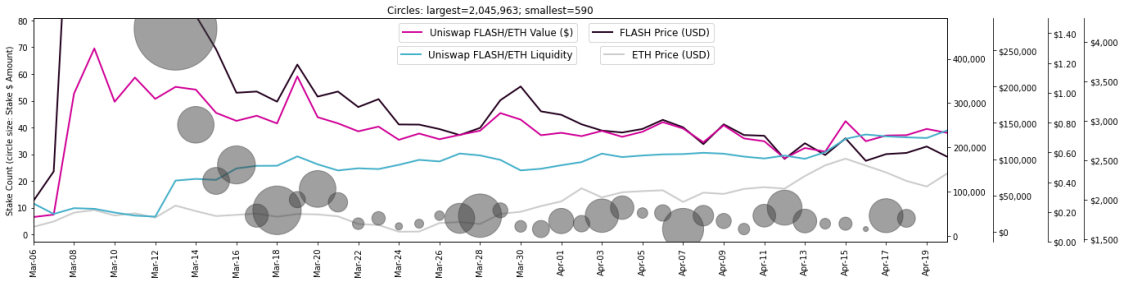
1.3 Staking / Price Stability

After this early spike in staking in early to mid March, staking stabilizes at less than 10 stakes per day while the price of FLASH drops close to \$0.60 by mid-April.

[110]:
```
fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

# Plot the stake data
# ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
 ↪color='#cad084', alpha=0.7)
ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
 ↪color='#424242', alpha=0.5, zorder=50) #, label="Number of Stakes")
ax1.zorder = 5
ax1.patch.set_alpha(0)
ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax1.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax1.set_ylabel('Stake Count (circle size: Stake $ Amount)')
ax1.set_xlim([datetime.date(2021,3,6), datetime.date(2021,4,20)])
totalsfiltered = list(filter(lambda x: pd.isnull(x) != True,␣
 ↪hist["staketotal"]))
plt.title("Circles: largest={:,.0f}; smallest={:,.0f}".
 ↪format(max(totalsfiltered), min(totalsfiltered)))

# Plot the stake data Regression Fit line
x = mdates.date2num(hist["date"])
y = hist["liquidity"]
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
```

```
ax2 = ax1.twinx()
# plt.plot(x, p(x), "--", color='#3C3E27', alpha=0.8)
ax2.plot(x, p(x), "--", color='#424242', alpha=0.99)
ax2.zorder = 6
ax2.patch.set_alpha(0)
ax2.set_ylim([0, 80])
ax2.yaxis.set_visible(False)

# Plot the Uniswap FLASH/ETH pool historical liquidity value
ax3 = ax1.twinx()
ax3.plot(hist["date"], hist["value"], color='#D10097', linewidth=2,␣
↪label="Uniswap FLASH/ETH Value ($)", zorder=10)
ax3.zorder = 2
ax3.patch.set_alpha(0)
ax3.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax3.xaxis.set_major_locator(mdates.DayLocator(interval=2))
ax3.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax3.legend(bbox_to_anchor = (.6, 1), prop={'size':11.75})

ax3.spines['right'].set_position(("axes", 1.05))
ax3.spines['right'].set_visible(True)
ax3.yaxis.set_label_position('right')
ax3.yaxis.set_ticks_position('right')

# Plot the Uniswap FLASH/ETH pool historical liquidity quantity
ax4 = ax1.twinx()
ax4.plot(hist["date"], hist["liquidity"], color='#43B0CA', linewidth=2,␣
↪label="Uniswap FLASH/ETH Liquidity", zorder=5)
ax4.zorder = 3
ax4.patch.set_alpha(0)
ax4.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax4.xaxis.set_major_locator(mdates.DayLocator(interval=2))
ax4.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
# ax4.legend("Liquidity", bbox_to_anchor = (1.05, 0.6))
ax4.legend(bbox_to_anchor = (.6, .9), prop={'size':12})

# Plot the Uniswap FLASH/USD price
ax5 = ax1.twinx()
ax5.plot(hist["date"], hist["price"], color='#23001E', linewidth=2,␣
↪label="FLASH Price (USD)", zorder=0)
ax5.zorder = 1
ax5.patch.set_alpha(0)
ax5.set_ylim([0, 1.5])
ax5.yaxis.set_major_formatter(StrMethodFormatter('${x:,.2f}'))
ax5.xaxis.set_major_locator(mdates.DayLocator(interval=2))
ax5.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax5.legend(bbox_to_anchor = (.75, 1), prop={'size':12})
```

```
ax5.spines['right'].set_position(("axes", 1.11))
ax5.spines['right'].set_visible(True)
ax5.yaxis.set_label_position('right')
ax5.yaxis.set_ticks_position('right')

# Plot the ETH/USD price
ax6 = ax1.twinx()
ax6.plot(hist["date"], hist["ethclose"], color='#CCC', linewidth=2, label="ETH␣
 ↪Price (USD)", zorder=0)
ax6.zorder = 1
ax6.patch.set_alpha(0)
# ax6.set_ylim([0, 1.5])
ax6.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax6.xaxis.set_major_locator(mdates.DayLocator(interval=2))
ax6.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax6.legend(bbox_to_anchor = (.75, .9), prop={'size':12})

ax6.spines['right'].set_position(("axes", 1.15))
ax6.spines['right'].set_visible(True)
ax6.yaxis.set_label_position('right')
ax6.yaxis.set_ticks_position('right')

# Regression Fitting
x = mdates.date2num(hist["date"])
y = hist["staketotal"]
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"b--")

plt.show()
```



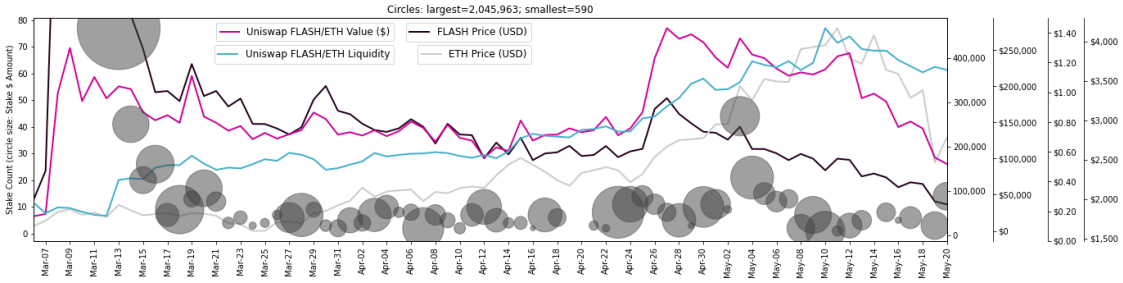
1.4 May Peak

By the end of April however, as the crypto market began its climb toward its May peak, the price of $FLASH rapidly rose and liquidity quickly pooled in Uniswap.

[111]:
```
fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

# Plot the stake data
# ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
 ↪color='#cad084', alpha=0.7)
ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
 ↪color='#424242', alpha=0.5, zorder=50) #, label="Number of Stakes")
ax1.zorder = 5
ax1.patch.set_alpha(0)
ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax1.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax1.set_ylabel('Stake Count (circle size: Stake $ Amount)')
ax1.set_xlim([datetime.date(2021,3,6), datetime.date(2021,5,20)])
totalsfiltered = list(filter(lambda x: pd.isnull(x) != True,␣
 ↪hist["staketotal"]))
plt.title("Circles: largest={:,.0f}; smallest={:,.0f}".
 ↪format(max(totalsfiltered), min(totalsfiltered)))

# Plot the stake data Regression Fit line
x = mdates.date2num(hist["date"])
y = hist["liquidity"]
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
ax2 = ax1.twinx()
# plt.plot(x, p(x), "--", color='#3C3E27', alpha=0.8)
ax2.plot(x, p(x), "--", color='#424242', alpha=0.99)
ax2.zorder = 6
ax2.patch.set_alpha(0)
ax2.set_ylim([0, 80])
ax2.yaxis.set_visible(False)

# Plot the Uniswap FLASH/ETH pool historical liquidity value
ax3 = ax1.twinx()
ax3.plot(hist["date"], hist["value"], color='#D10097', linewidth=2,␣
 ↪label="Uniswap FLASH/ETH Value ($)", zorder=10)
ax3.zorder = 2
ax3.patch.set_alpha(0)
ax3.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax3.xaxis.set_major_locator(mdates.DayLocator(interval=2))
ax3.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax3.legend(bbox_to_anchor = (.4, 1), prop={'size':11.75})

ax3.spines['right'].set_position(("axes", 1.05))
```

```
ax3.spines['right'].set_visible(True)
ax3.yaxis.set_label_position('right')
ax3.yaxis.set_ticks_position('right')

# Plot the Uniswap FLASH/ETH pool historical liquidity quantity
ax4 = ax1.twinx()
ax4.plot(hist["date"], hist["liquidity"], color='#43B0CA', linewidth=2,␣
↪label="Uniswap FLASH/ETH Liquidity", zorder=5)
ax4.zorder = 3
ax4.patch.set_alpha(0)
ax4.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax4.xaxis.set_major_locator(mdates.DayLocator(interval=2))
ax4.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
# ax4.legend("Liquidity", bbox_to_anchor = (1.05, 0.6))
ax4.legend(bbox_to_anchor = (.4, .9), prop={'size':12})

# Plot the Uniswap FLASH/USD price
ax5 = ax1.twinx()
ax5.plot(hist["date"], hist["price"], color='#23001E', linewidth=2,␣
↪label="FLASH Price (USD)", zorder=0)
ax5.zorder = 1
ax5.patch.set_alpha(0)
ax5.set_ylim([0, 1.5])
ax5.yaxis.set_major_formatter(StrMethodFormatter('${x:,.2f}'))
ax5.xaxis.set_major_locator(mdates.DayLocator(interval=2))
ax5.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax5.legend(bbox_to_anchor = (.55, 1), prop={'size':12})

ax5.spines['right'].set_position(("axes", 1.11))
ax5.spines['right'].set_visible(True)
ax5.yaxis.set_label_position('right')
ax5.yaxis.set_ticks_position('right')

# Plot the ETH/USD price
ax6 = ax1.twinx()
ax6.plot(hist["date"], hist["ethclose"], color='#CCC', linewidth=2, label="ETH␣
↪Price (USD)", zorder=0)
ax6.zorder = 1
ax6.patch.set_alpha(0)
# ax6.set_ylim([0, 1.5])
ax6.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax6.xaxis.set_major_locator(mdates.DayLocator(interval=2))
ax6.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax6.legend(bbox_to_anchor = (.55, .9), prop={'size':12})

ax6.spines['right'].set_position(("axes", 1.15))
ax6.spines['right'].set_visible(True)
```

```
ax6.yaxis.set_label_position('right')
ax6.yaxis.set_ticks_position('right')

# Regression Fitting
x = mdates.date2num(hist["date"])
y = hist["staketotal"]
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"b--")

plt.show()
```



1.5 Liquidity Delay

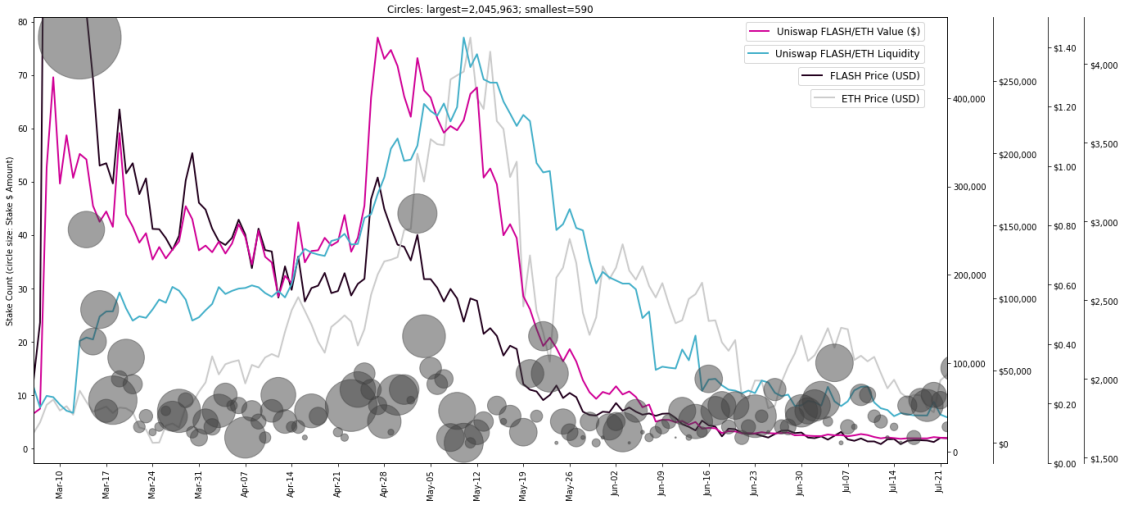
The price of $\backslash FLASH quickly dropped after the late-April spike. However, Uniswap liquidity in the FLASH/ETH p
ETH.

Just like the $\backslash ETH price spike in May, the Uniswap FLASH/ETH liquidity did not last long, with liquidity gradually
$June, when low-levels of liquidity stabilized along with the price of$
FLASH.

[112]:
```
fig, ax1 = plt.subplots(figsize=(20, 10))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

# Plot the stake data
# ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
↪color='#cad084', alpha=0.7)
ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
↪color='#424242', alpha=0.5, zorder=50) #, label="Number of Stakes")
ax1.zorder = 5
ax1.patch.set_alpha(0)
ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax1.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
```

```
ax1.set_ylabel('Stake Count (circle size: Stake $ Amount)')
ax1.set_xlim([datetime.date(2021,3,6), datetime.date(2021,7,22)])
totalsfiltered = list(filter(lambda x: pd.isnull(x) != True,␣
 ↪hist["staketotal"]))
plt.title("Circles: largest={:,.0f}; smallest={:,.0f}".
 ↪format(max(totalsfiltered), min(totalsfiltered)))

# Plot the stake data Regression Fit line
x = mdates.date2num(hist["date"])
y = hist["liquidity"]
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
ax2 = ax1.twinx()
# plt.plot(x, p(x), "--", color='#3C3E27', alpha=0.8)
ax2.plot(x, p(x), "--", color='#424242', alpha=0.99)
ax2.zorder = 6
ax2.patch.set_alpha(0)
ax2.set_ylim([0, 80])
ax2.yaxis.set_visible(False)

# Plot the Uniswap FLASH/ETH pool historical liquidity value
ax3 = ax1.twinx()
ax3.plot(hist["date"], hist["value"], color='#D10097', linewidth=2,␣
 ↪label="Uniswap FLASH/ETH Value ($)", zorder=10)
ax3.zorder = 2
ax3.patch.set_alpha(0)
ax3.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax3.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax3.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax3.legend(bbox_to_anchor = (.98, 1), prop={'size':11.75})

ax3.spines['right'].set_position(("axes", 1.05))
ax3.spines['right'].set_visible(True)
ax3.yaxis.set_label_position('right')
ax3.yaxis.set_ticks_position('right')

# Plot the Uniswap FLASH/ETH pool historical liquidity quantity
ax4 = ax1.twinx()
ax4.plot(hist["date"], hist["liquidity"], color='#43B0CA', linewidth=2,␣
 ↪label="Uniswap FLASH/ETH Liquidity", zorder=5)
ax4.zorder = 3
ax4.patch.set_alpha(0)
ax4.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax4.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax4.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
# ax4.legend("Liquidity", bbox_to_anchor = (1.05, 0.6))
ax4.legend(bbox_to_anchor = (.98, .95), prop={'size':12})
```

```
# Plot the Uniswap FLASH/USD price
ax5 = ax1.twinx()
ax5.plot(hist["date"], hist["price"], color='#23001E', linewidth=2,␣
 ↪label="FLASH Price (USD)", zorder=0)
ax5.zorder = 1
ax5.patch.set_alpha(0)
ax5.set_ylim([0, 1.5])
ax5.yaxis.set_major_formatter(StrMethodFormatter('${x:,.2f}'))
ax5.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax5.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax5.legend(bbox_to_anchor = (.98, .9), prop={'size':12})

ax5.spines['right'].set_position(("axes", 1.11))
ax5.spines['right'].set_visible(True)
ax5.yaxis.set_label_position('right')
ax5.yaxis.set_ticks_position('right')

# Plot the ETH/USD price
ax6 = ax1.twinx()
ax6.plot(hist["date"], hist["ethclose"], color='#CCC', linewidth=2, label="ETH␣
 ↪Price (USD)", zorder=0)
ax6.zorder = 1
ax6.patch.set_alpha(0)
# ax6.set_ylim([0, 1.5])
ax6.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax6.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax6.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax6.legend(bbox_to_anchor = (.98, .85), prop={'size':12})

ax6.spines['right'].set_position(("axes", 1.15))
ax6.spines['right'].set_visible(True)
ax6.yaxis.set_label_position('right')
ax6.yaxis.set_ticks_position('right')

# Regression Fitting
x = mdates.date2num(hist["date"])
y = hist["staketotal"]
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"b--")

plt.show()
```



1.5.1 ETH Correlation?

A natural question when considering the fate of V2 is whether the general crypto market played a role. Unfortunately, this analysis does not adequately examine this influence. Although a rough analysis of correlation does seem to show some related aspects (an R-squared valuation of > 0.7), this is not enough evidence to conclude that the price of \$ETH was the main determiner of liquidity in Uniswap or staking in Flash.

[ ]:
```
model = sm.OLS(hist["liquidity"], hist["ethclose"]).fit()
predictions = model.predict(x)
model.summary()
```

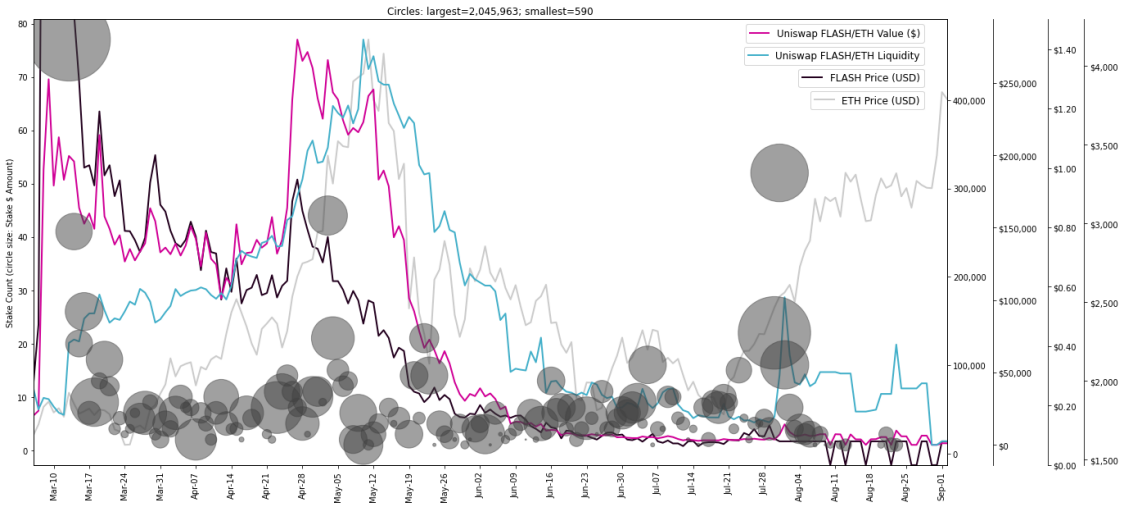
1.6 July 31 V3 Deadline

If we zoom out to consider the entire Flash V2 period up through early September, we can see the July 31 deadline for establishing \$FLASH staking positions for the V3 consideration causing a spike in staking.

[113]:
```
fig, ax1 = plt.subplots(figsize=(20, 10))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

# Plot the stake data
# ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
 ↪color='#cad084', alpha=0.7)
ax1.scatter(hist["date"], hist["stakecount"], s=hist["staketotal"] / 200,␣
 ↪color='#424242', alpha=0.5, zorder=50) #, label="Number of Stakes")
ax1.zorder = 5
ax1.patch.set_alpha(0)
```

```
ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax1.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax1.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax1.set_ylabel('Stake Count (circle size: Stake $ Amount)')
ax1.set_xlim([datetime.date(2021,3,6), datetime.date(2021,9,2)])
totalsfiltered = list(filter(lambda x: pd.isnull(x) != True,␣
 ↪hist["staketotal"]))
plt.title("Circles: largest={:,.0f}; smallest={:,.0f}".
 ↪format(max(totalsfiltered), min(totalsfiltered)))

# Plot the stake data Regression Fit line
x = mdates.date2num(hist["date"])
y = hist["liquidity"]
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
ax2 = ax1.twinx()
# plt.plot(x, p(x), "--", color='#3C3E27', alpha=0.8)
ax2.plot(x, p(x), "--", color='#424242', alpha=0.99)
ax2.zorder = 6
ax2.patch.set_alpha(0)
ax2.set_ylim([0, 80])
ax2.yaxis.set_visible(False)

# Plot the Uniswap FLASH/ETH pool historical liquidity value
ax3 = ax1.twinx()
ax3.plot(hist["date"], hist["value"], color='#D10097', linewidth=2,␣
 ↪label="Uniswap FLASH/ETH Value ($)", zorder=10)
ax3.zorder = 2
ax3.patch.set_alpha(0)
ax3.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax3.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax3.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax3.legend(bbox_to_anchor = (.98, 1), prop={'size':11.75})

ax3.spines['right'].set_position(("axes", 1.05))
ax3.spines['right'].set_visible(True)
ax3.yaxis.set_label_position('right')
ax3.yaxis.set_ticks_position('right')

# Plot the Uniswap FLASH/ETH pool historical liquidity quantity
ax4 = ax1.twinx()
ax4.plot(hist["date"], hist["liquidity"], color='#43B0CA', linewidth=2,␣
 ↪label="Uniswap FLASH/ETH Liquidity", zorder=5)
ax4.zorder = 3
ax4.patch.set_alpha(0)
ax4.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax4.xaxis.set_major_locator(mdates.DayLocator(interval=7))
```

```
ax4.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
# ax4.legend("Liquidity", bbox_to_anchor = (1.05, 0.6))
ax4.legend(bbox_to_anchor = (.98, .95), prop={'size':12})

# Plot the Uniswap FLASH/USD price
ax5 = ax1.twinx()
ax5.plot(hist["date"], hist["price"], color='#23001E', linewidth=2,␣
↪label="FLASH Price (USD)", zorder=0)
ax5.zorder = 1
ax5.patch.set_alpha(0)
ax5.set_ylim([0, 1.5])
ax5.yaxis.set_major_formatter(StrMethodFormatter('${x:,.2f}'))
ax5.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax5.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax5.legend(bbox_to_anchor = (.98, .9), prop={'size':12})

ax5.spines['right'].set_position(("axes", 1.11))
ax5.spines['right'].set_visible(True)
ax5.yaxis.set_label_position('right')
ax5.yaxis.set_ticks_position('right')

# Plot the ETH/USD price
ax6 = ax1.twinx()
ax6.plot(hist["date"], hist["ethclose"], color='#CCC', linewidth=2, label="ETH␣
↪Price (USD)", zorder=0)
ax6.zorder = 1
ax6.patch.set_alpha(0)
# ax6.set_ylim([0, 1.5])
ax6.yaxis.set_major_formatter(StrMethodFormatter('${x:,.0f}'))
ax6.xaxis.set_major_locator(mdates.DayLocator(interval=7))
ax6.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax6.legend(bbox_to_anchor = (.98, .85), prop={'size':12})

ax6.spines['right'].set_position(("axes", 1.15))
ax6.spines['right'].set_visible(True)
ax6.yaxis.set_label_position('right')
ax6.yaxis.set_ticks_position('right')

# Regression Fitting
x = mdates.date2num(hist["date"])
y = hist["staketotal"]
z = np.polyfit(x, y, 1)
p = np.poly1d(z)
plt.plot(x,p(x),"b--")

plt.show()
```



2 Flash V2 User Behavior

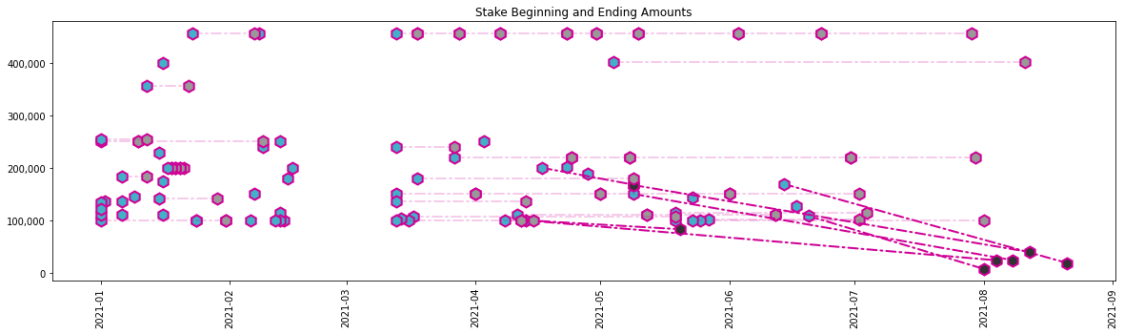
Now let's consider the staking behavior of accounts, grouped by the little amount of information we can use to categorize them.

2.1 Length of Staking & Unstaking Early

By tracking when a stake began, when it ended, and how much was lost due to unstaking early, we can see staking behavior patterns in the most intuitive way possible. We will group these behaviors by amount staked at the start for more visual clarity.

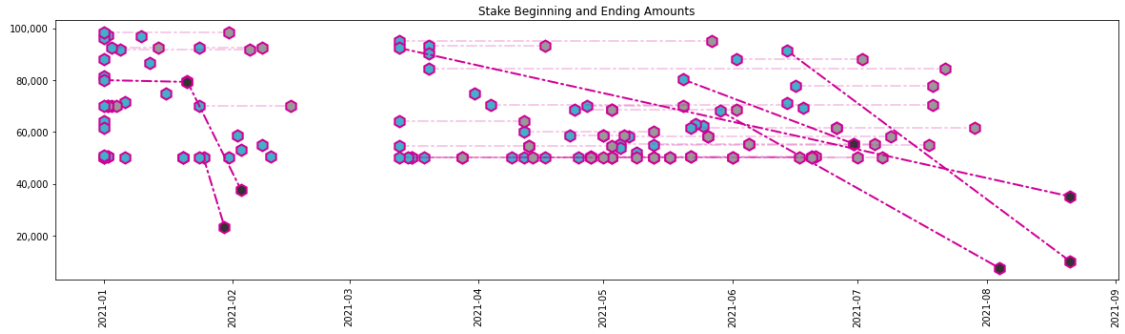
[153]:
```
freq = pd.read_sql("""SELECT amountin, stakedatetime, unstakedatetime,␣
 ↪unstakeearlyburnedamount
FROM stakes
WHERE stakedatetime < '2021-07-01'::date AND amountin >= 100000
ORDER BY stakedatetime DESC;""",conn)

fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

x1 = freq["stakedatetime"]
x2 = freq["unstakedatetime"]
y1 = freq["amountin"]
y2 = freq["unstakeearlyburnedamount"]
for i in range(0, len(x1)):
    if np.isnan(y2[i]):
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]], color='#D10097', linewidth=2,␣
 ↪alpha=0.2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
```

```
        ax1.plot([x2[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
↪markerfacecolor='#999', markeredgewidth=2, markersize=12, markevery=3)
    else:
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]-y2[i]], color='#D10097',␣
↪linewidth=2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]-y2[i]], color='#D10097', linewidth=4,␣
↪marker='h', markerfacecolor='#333', markeredgewidth=2, markersize=12,␣
↪markevery=3)
    ax1.plot([x1[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
↪markerfacecolor='#43B0CA', markeredgewidth=2, markersize=12, markevery=3)

ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
plt.title("Stake Beginning and Ending Amounts")
plt.show()
```



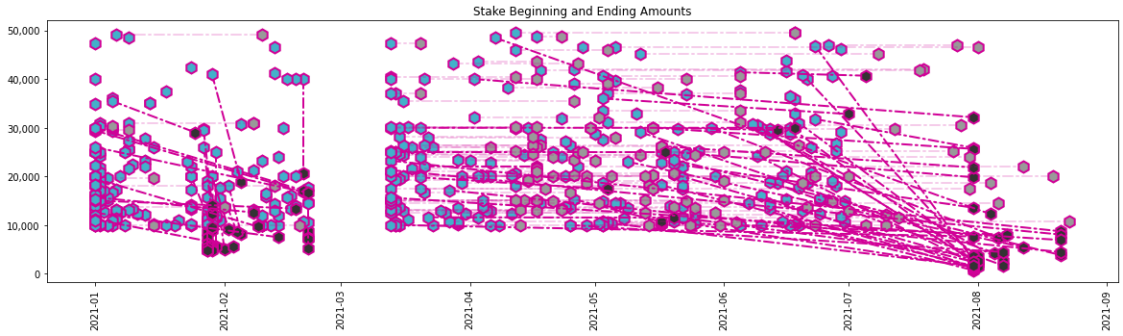
[154]:
```
freq = pd.read_sql("""SELECT amountin, stakedatetime, unstakedatetime,␣
↪unstakeearlyburnedamount
FROM stakes
WHERE stakedatetime < '2021-07-01'::date AND (amountin >= 50000 AND amountin <␣
↪100000)
ORDER BY stakedatetime DESC;""",conn)
# AND flashversion=2

fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

x1 = freq["stakedatetime"]
x2 = freq["unstakedatetime"]
y1 = freq["amountin"]
y2 = freq["unstakeearlyburnedamount"]
for i in range(0, len(x1)):
    if np.isnan(y2[i]):
```

```
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]], color='#D10097', linewidth=2,␣
↪alpha=0.2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
↪markerfacecolor='#999', markeredgewidth=2, markersize=12, markevery=3)
    else:
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]-y2[i]], color='#D10097',␣
↪linewidth=2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]-y2[i]], color='#D10097', linewidth=4,␣
↪marker='h', markerfacecolor='#333', markeredgewidth=2, markersize=12,␣
↪markevery=3)
    ax1.plot([x1[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
↪markerfacecolor='#43B0CA', markeredgewidth=2, markersize=12, markevery=3)

ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
plt.title("Stake Beginning and Ending Amounts")
plt.show()
```



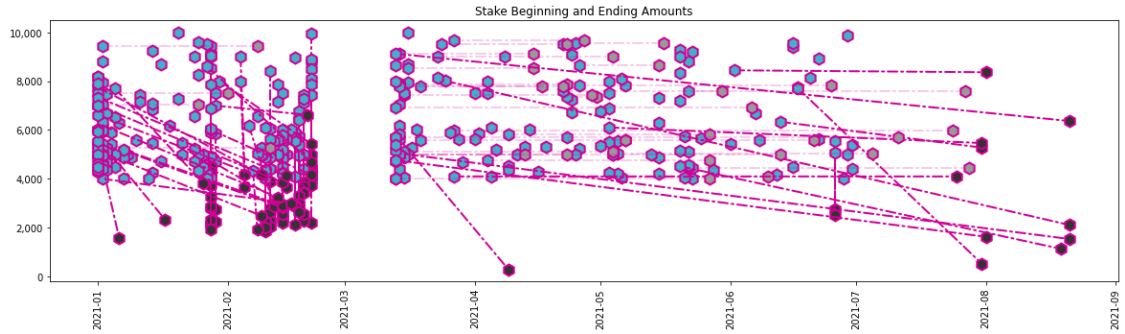
[155]:
```
freq = pd.read_sql("""SELECT amountin, stakedatetime, unstakedatetime,␣
↪unstakeearlyburnedamount
FROM stakes
WHERE stakedatetime < '2021-07-01'::date AND (amountin >= 10000 AND amountin <␣
↪50000)
ORDER BY stakedatetime DESC;""",conn)
# AND flashversion=2

fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

x1 = freq["stakedatetime"]
x2 = freq["unstakedatetime"]
y1 = freq["amountin"]
y2 = freq["unstakeearlyburnedamount"]
```

```
for i in range(0, len(x1)):
    if np.isnan(y2[i]):
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]], color='#D10097', linewidth=2,␣
↪alpha=0.2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
↪markerfacecolor='#999', markeredgewidth=2, markersize=12, markevery=3)
    else:
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]-y2[i]], color='#D10097',␣
↪linewidth=2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]-y2[i]], color='#D10097', linewidth=4,␣
↪marker='h', markerfacecolor='#333', markeredgewidth=2, markersize=12,␣
↪markevery=3)
    ax1.plot([x1[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
↪markerfacecolor='#43B0CA', markeredgewidth=2, markersize=12, markevery=3)

ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
plt.title("Stake Beginning and Ending Amounts")
plt.show()
```



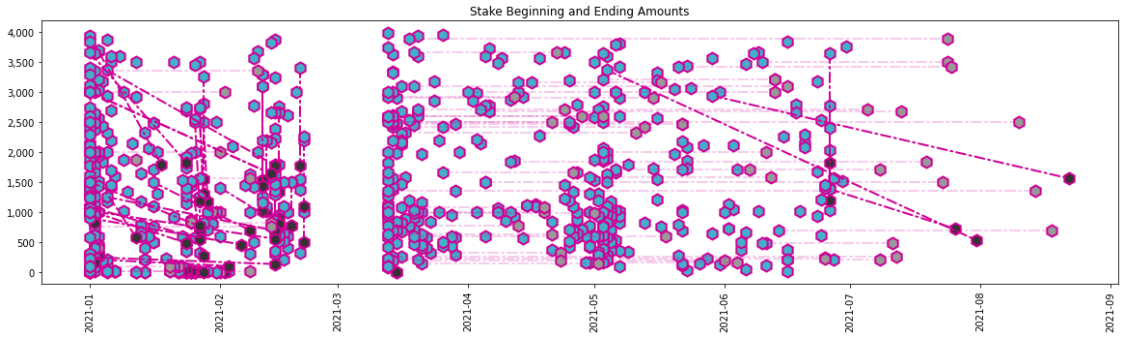
[156]:
```
freq = pd.read_sql("""SELECT amountin, stakedatetime, unstakedatetime,␣
↪unstakeearlyburnedamount
FROM stakes
WHERE stakedatetime < '2021-07-01'::date AND (amountin >= 4000 AND amountin <␣
↪10000)
ORDER BY stakedatetime DESC;""",conn)
# AND flashversion=2

fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

x1 = freq["stakedatetime"]
x2 = freq["unstakedatetime"]
```

```
y1 = freq["amountin"]
y2 = freq["unstakeearlyburnedamount"]
for i in range(0, len(x1)):
    if np.isnan(y2[i]):
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]], color='#D10097', linewidth=2,␣
 ↪alpha=0.2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
 ↪markerfacecolor='#999', markeredgewidth=2, markersize=12, markevery=3)
    else:
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]-y2[i]], color='#D10097',␣
 ↪linewidth=2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]-y2[i]], color='#D10097', linewidth=4,␣
 ↪marker='h', markerfacecolor='#333', markeredgewidth=2, markersize=12,␣
 ↪markevery=3)
    ax1.plot([x1[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
 ↪markerfacecolor='#43B0CA', markeredgewidth=2, markersize=12, markevery=3)

ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
plt.title("Stake Beginning and Ending Amounts")
plt.show()
```



[157]:
```
freq = pd.read_sql("""SELECT amountin, stakedatetime, unstakedatetime,␣
 ↪unstakeearlyburnedamount
FROM stakes
WHERE stakedatetime < '2021-07-01'::date AND (amountin > 0 AND amountin < 4000)
ORDER BY stakedatetime DESC;""",conn)
# AND flashversion=2

fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

x1 = freq["stakedatetime"]
```

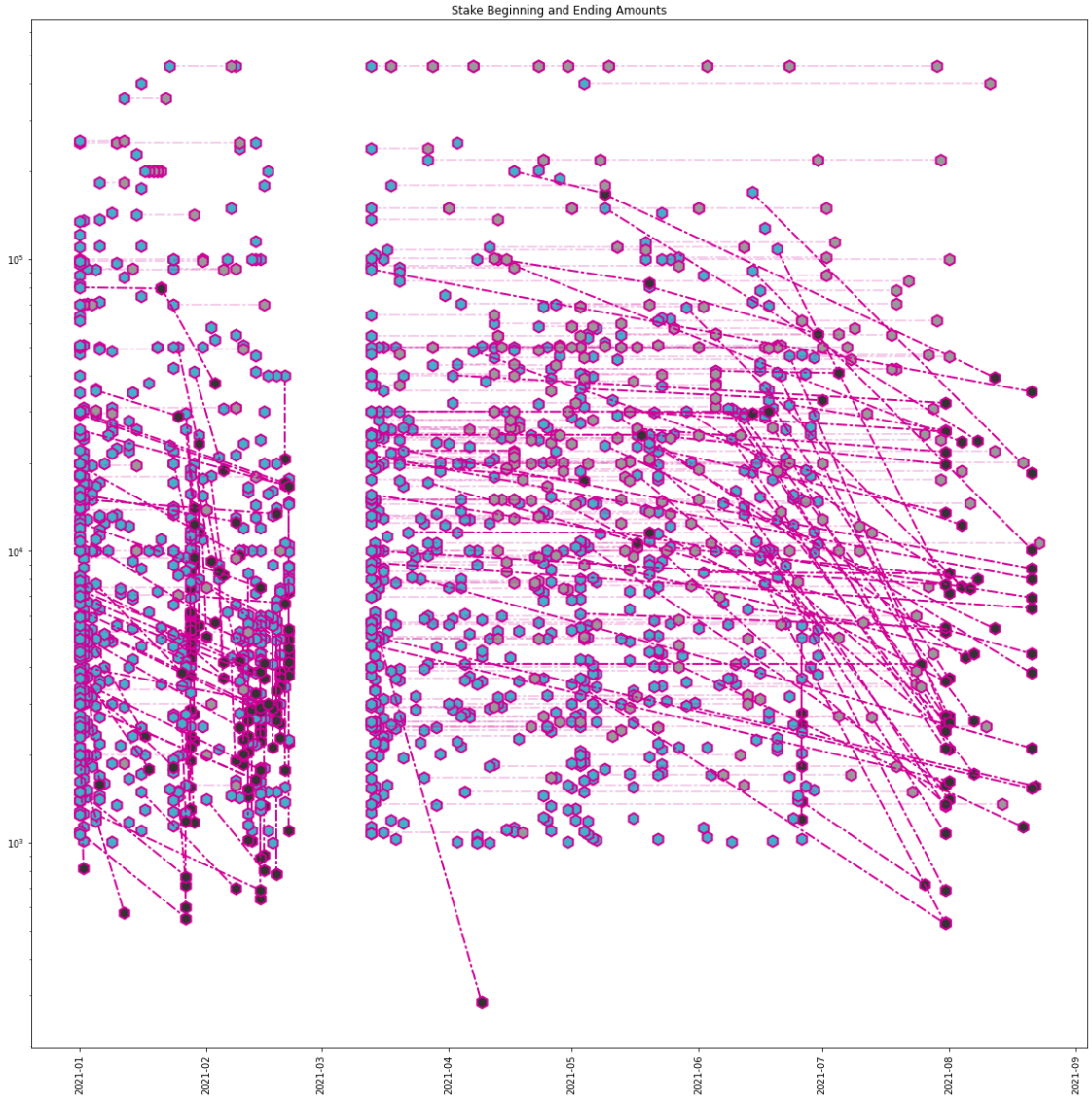
```
x2 = freq["unstakedatetime"]
y1 = freq["amountin"]
y2 = freq["unstakeearlyburnedamount"]
for i in range(0, len(x1)):
    if np.isnan(y2[i]):
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]], color='#D10097', linewidth=2,␣
 ↪alpha=0.2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
 ↪markerfacecolor='#999', markeredgewidth=2, markersize=12, markevery=3)
    else:
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]-y2[i]], color='#D10097',␣
 ↪linewidth=2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]-y2[i]], color='#D10097', linewidth=4,␣
 ↪marker='h', markerfacecolor='#333', markeredgewidth=2, markersize=12,␣
 ↪markevery=3)
    ax1.plot([x1[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
 ↪markerfacecolor='#43B0CA', markeredgewidth=2, markersize=12, markevery=3)

ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
plt.title("Stake Beginning and Ending Amounts")
plt.show()
```



To view most of the data on a single chart, we can use a logarithmic scale. However, this graph is less intuitive since the visual distance between points is not a consistent value.

[158]:
```
freq = pd.read_sql("""SELECT amountin, stakedatetime, unstakedatetime,␣
 ↪unstakeearlyburnedamount
FROM stakes
WHERE stakedatetime < '2021-07-01'::date AND (amountin > 1000 AND amountin <␣
 ↪1000000)
ORDER BY stakedatetime DESC;""",conn)
# AND flashversion=2

fig, ax1 = plt.subplots(figsize=(20, 20))
```

```
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

x1 = freq["stakedatetime"]
x2 = freq["unstakedatetime"]
y1 = freq["amountin"]
y2 = freq["unstakeearlyburnedamount"]
for i in range(0, len(x1)):
    if np.isnan(y2[i]):
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]], color='#D10097', linewidth=2,␣
↪alpha=0.2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
↪markerfacecolor='#999', markeredgewidth=2, markersize=12, markevery=3)
    else:
        ax1.plot([x1[i],x2[i]], [y1[i],y1[i]-y2[i]], color='#D10097',␣
↪linewidth=2, linestyle=(0, (5, 2, 1, 2)), dash_capstyle='round')
        ax1.plot([x2[i]], [y1[i]-y2[i]], color='#D10097', linewidth=4,␣
↪marker='h', markerfacecolor='#333', markeredgewidth=2, markersize=12,␣
↪markevery=3)
    ax1.plot([x1[i]], [y1[i]], color='#D10097', linewidth=4, marker='h',␣
↪markerfacecolor='#43B0CA', markeredgewidth=2, markersize=12, markevery=3)

ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax1.set_yscale('log')
plt.title("Stake Beginning and Ending Amounts")
plt.show()
```



Unstaking early appears to mostly occur at milestones related to Flash version changes. Let's check that guess with a better summary analysis of unstakes, specifically early unstakes.

Note: We will only consider stakes before July 1. Activity after that period is either too recent or tainted by July 31 V3 deadline activity.

[106]:
```
stats = pd.read_sql("""SELECT COUNT(stakedatetime) AS "Total Stakes",
    TO_CHAR((COUNT(unstakedatetime)::float / COUNT(stakedatetime)::float) *␣
↪100, '999,999.0%') AS "Percent Unstaked",
    TO_CHAR((COUNT(unstakeearlyburnedamount)::float / COUNT(stakedatetime)::
↪float) * 100, '999,999.0%') AS "Percent Unstaked Early",
    TO_CHAR(SUM(amountin), '999,999,999') AS "Total Staked",
    TO_CHAR(SUM(unstakeearlyburnedamount), '999,999,999') AS "Total Burned",
```

```
    TO_CHAR((SUM(unstakeearlyburnedamount) / SUM(amountin)) * 100, '999,999.
 ↪0%') AS "Percent Burned"
FROM stakes
WHERE stakedatetime < '2021-07-01'::date;""",conn)
stats_t = stats.transpose()
stats_t.columns = stats_t.iloc[0]
stats_t = stats_t.drop(stats_t.iloc[0].index.name)
# stats_t.columns.name = None
display(stats_t)
```

Total Stakes	1606
Percent Unstaked	35.6%
Percent Unstaked Early	16.1%
Total Staked	34,570,644
Total Burned	2,158,429
Percent Burned	6.2%

So only about a third of stakes are unstaked and about half of those are unstaked early, leading to 6.2% of all staked amounts being burned.

Let's see a summary of when unstaking occurs. First we will look at data from days with many unstakes.

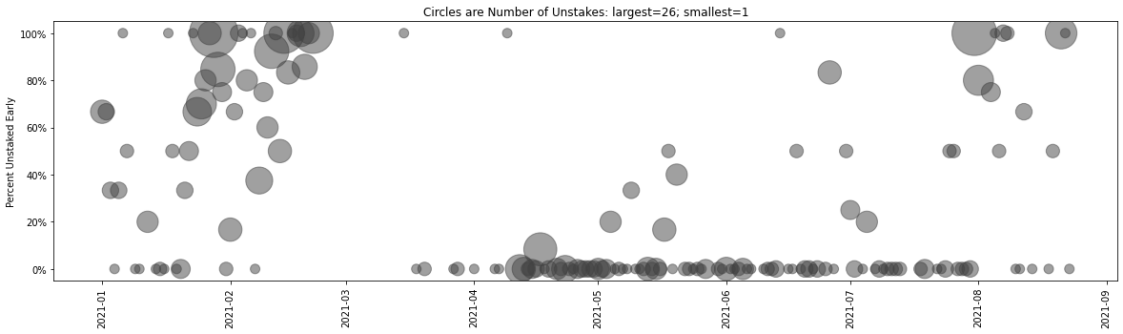
[139]:
```
stats = pd.read_sql("""SELECT unstakedatetime,
    COUNT(unstakedatetime) AS "Total Unstaked",
    TO_CHAR((COUNT(unstakeearlyburnedamount)::float / COUNT(stakedatetime)::
 ↪float) * 100, '999,999.0%') AS "Percent Unstaked Early",
    TO_CHAR(SUM(unstakeearlyburnedamount), '999,999,999') AS "Total Burned",
    TO_CHAR((SUM(unstakeearlyburnedamount) / SUM(amountin)) * 100, '999,999.
 ↪0%') AS "Percent Burned"
FROM stakes
WHERE stakedatetime < '2021-07-01'::date AND unstakedatetime is not null
GROUP BY unstakedatetime
HAVING COUNT(unstakedatetime) > 9
ORDER BY unstakedatetime ASC;""",conn)
stats = stats.transpose()
stats.columns = stats.iloc[0]
stats = stats.drop(stats.iloc[0].index.name)
stats.columns.name = None
stats = stats.transpose()
display(stats)
```

	Total Unstaked	Percent Unstaked Early	Total Burned	Percent Burned
2021-01-25	10	70.0%	6,496	15.0%
2021-01-28	26	100.0%	105,584	52.7%
2021-01-29	13	84.6%	80,325	27.2%
2021-02-11	13	92.3%	28,812	48.8%
2021-02-14	18	100.0%	38,609	47.5%

2021-02-21	18	100.0%	79,872	46.2%
2021-04-17	12	8.3%	5	.0%
2021-07-31	22	100.0%	250,694	58.2%
2021-08-01	10	80.0%	224,986	56.1%
2021-08-21	11	100.0%	370,110	77.8%

The end of January, mid-late February, and around July 31 appear to be the hotspots.

[160]:
```
stats = pd.read_sql("""SELECT unstakedatetime,
    COUNT(unstakedatetime) AS totalunstaked,
    (COUNT(unstakeearlyburnedamount)::float / COUNT(stakedatetime)::float) AS␣
 ↪percunstakedearly,
    SUM(unstakeearlyburnedamount) AS totalburned,
    (SUM(unstakeearlyburnedamount) / SUM(amountin)) AS percburned
FROM stakes
WHERE stakedatetime < '2021-07-01'::date AND unstakedatetime is not null
GROUP BY unstakedatetime
ORDER BY unstakedatetime ASC;""",conn)

fig, ax1 = plt.subplots(figsize=(20, 5))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)
plt.xticks(rotation = 90)

# Plot the unstake data
sc = ax1.scatter(stats["unstakedatetime"], stats["percunstakedearly"] * 100,␣
 ↪s=stats["totalunstaked"] * 100, color='#424242', alpha=0.5, zorder=50,␣
 ↪label="Total Stake Amount")
# ax1.zorder = 5
# ax1.patch.set_alpha(0)
# ax1.yaxis.set_major_formatter(StrMethodFormatter('{x:,.0f}'))
ax1.yaxis.set_major_formatter(PercentFormatter())
# ax1.xaxis.set_major_locator(mdates.DayLocator(interval=7))
# ax1.xaxis.set_major_formatter(mdates.DateFormatter('%b-%d'))
ax1.set_ylabel('Percent Unstaked Early')
# ax1.set_xlim([datetime.date(2021,3,1), datetime.date(2021,9,2)])
# ax1.legend(*sc.legend_elements("sizes", num=2), loc="upper right",␣
 ↪title="Size")
# print(sc.legend_elements())
totalsfiltered = list(filter(lambda x: pd.isnull(x) != True,␣
 ↪stats["totalunstaked"]))
# plt.title("Circles: largest=%.2f, smallest=%.2f" % (max(totalsfiltered),␣
 ↪min(totalsfiltered)))
plt.title("Circles are Number of Unstakes: largest={:,.0f}; smallest={:,.0f}".
 ↪format(max(totalsfiltered), min(totalsfiltered)))
plt.show()
```

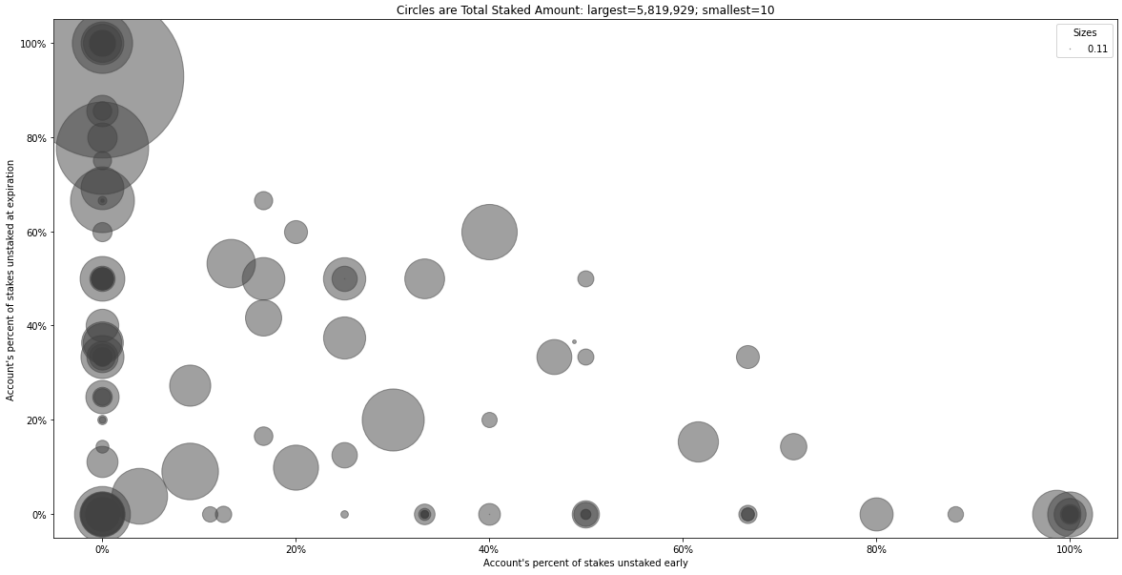


2.1.1 Accounts that unstake early

Let's break out the accounts by total amount staked over Flash's lifetime to understand which ones unstake early.

[147]:
```
demog = pd.read_sql("""SELECT fromaddress,
    to_timestamp(avg(extract(epoch from stakedatetime)))::date AS avgstakedate,
    to_timestamp(avg(extract(epoch from unstakedatetime)))::date AS␣
 ↪avgunstakedate,
    COUNT(stakedatetime) AS stakecount,
    COUNT(unstakedatetime) AS unstakecount,
    COUNT(unstakeearlyburnedamount) as unstakeearlycount,
    SUM(amountin) as staketotal,
    SUM(unstakeearlyburnedamount) as burntotal
FROM stakes
WHERE stakedatetime < '2021-07-15'::date
GROUP BY fromaddress
ORDER BY avgstakedate DESC;""",conn)
# display(best)

fig, ax1 = plt.subplots(figsize=(20, 10))
fig.patch.set_facecolor('white')
fig.patch.set_alpha(1)

x1 = demog["avgstakedate"]
y1 = demog["stakecount"]
y2 = demog["unstakecount"]
y3 = demog["unstakeearlycount"]
s1 = demog["staketotal"]
d1 = demog["avgstakedate"]
d2 = demog["avgunstakedate"]
for i in range(0, len(x1)):
#     print("%s, %5.2f" % (demog["fromaddress"][i], y2[i] / y1[i]))
    # ax1.plot(x1[i], y1[i], color='#D10097', linewidth=0, marker='h',␣
 ↪markerfacecolor='#43B0CA', markeredgewidth=2, markersize=12, markevery=3)
```

```
    # datetime.date.today()
#     color = 1 if d2[i] is None and d1[i] is None else
    scatter = ax1.scatter((y3[i] / y1[i]) * 100, ((y2[i] - y3[i]) / y1[i]) *␣
 ↪100, s=s1[i] / 200, c='#424242', cmap='RdPu', alpha=0.5, zorder=50)
#     ax1.scatter((y3[i] / y1[i]) * 100, ((y2[i] - y3[i]) / y1[i]) * 100,␣
 ↪s=s1[i] / 200, c=(datetime.date.today() - d1[i]).days * 100, cmap='RdPu',␣
 ↪alpha=0.5, zorder=50)
#     ax1.scatter(x1[i], (y2[i] / y1[i]) * 100, s=s1[i] / 200, color='#43B0CA',␣
 ↪alpha=0.5, zorder=50) #, label="Number of Stakes") #blue #43B0CA
#     ax1.scatter(x1[i], (y3[i] / y1[i]) * 100, s=s1[i] / 200, color='#D10097',␣
 ↪alpha=0.5, zorder=50) #, label="Number of Stakes") #pink #D10097 #gray␣
 ↪#424242
    handles, labels = scatter.legend_elements(prop="sizes", alpha=0.6)

ax1.xaxis.set_major_formatter(PercentFormatter())
ax1.yaxis.set_major_formatter(PercentFormatter())
ax1.set_xlabel("Account's percent of stakes unstaked early")
ax1.set_ylabel("Account's percent of stakes unstaked at expiration")
legend2 = ax1.legend(handles, labels, loc="upper right", title="Sizes")
totalsfiltered = list(filter(lambda x: pd.isnull(x) != True,␣
 ↪demog["staketotal"]))
# plt.title("Circles: largest=%.2f, smallest=%.2f" % (max(totalsfiltered),␣
 ↪min(totalsfiltered)))
plt.title("Circles are Total Staked Amount: largest={:,.0f}; smallest={:,.0f}".
 ↪format(max(totalsfiltered), min(totalsfiltered)))
# plt.title("Unstaking Behavior by Total Staking Amount")
plt.show()
```

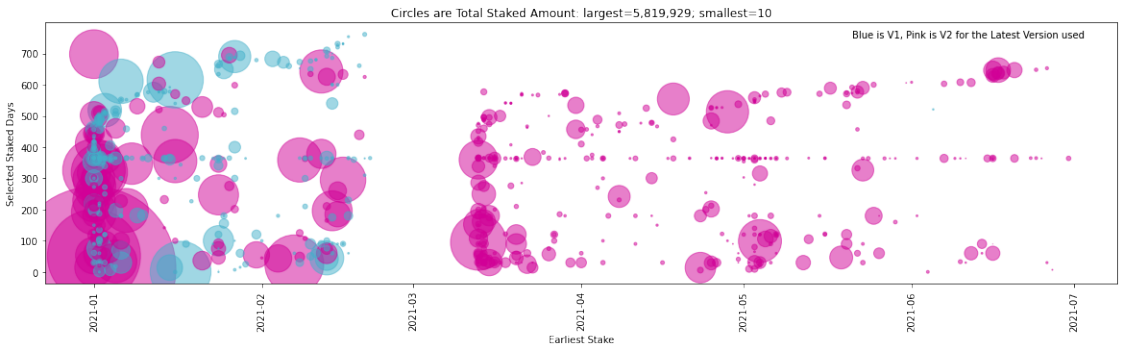


We can see that the accounts that tend to unstake early are not ones that stake a large amount of tokens.

2.2 Old vs. New Accounts

Finally let's consider the behavior differences between older and newer accounts. We will also include V1 data for this analysis.

[151]:
```
hodlers = pd.read_sql("""SELECT fromaddress AS "Account", SUM(amountin) as␣
 ↪"Total Staked", MIN(stakedatetime) AS "Earliest Stake", AVG(totalstakeddays)␣
 ↪AS "Selected Staked Days", MAX(flashversion) AS "Latest Version"
FROM stakes
WHERE stakedatetime < '2021-07-01'::date
GROUP BY fromaddress
ORDER BY "Total Staked" DESC;""",conn)

colors = ["#D10097" if v==2 else "#43B0CA" for v in hodlers["Latest Version"]]
ax = hodlers.plot.scatter(x="Earliest Stake", y="Selected Staked Days",␣
 ↪s=hodlers["Total Staked"] / 200, c=colors, rot=90, alpha=0.5,␣
 ↪figsize=(20,5));
# ax.set_xticklabels(['{:%m-%d}'.format(d) for d in hodlers['earlieststake']])
# ax.set_xticklabels(['{:%d-:%d}'.format(d.month, d.day) for d in␣
 ↪hodlers['earlieststake']])
# ax.xaxis.set_major_formatter(DateFormatter('%m-%d'))
# plt.gca().invert_xaxis()
plt.text(datetime.date(2021,5,21), 750, 'Blue is V1, Pink is V2 for the Latest␣
 ↪Version used')
totalsfiltered = list(filter(lambda x: pd.isnull(x) != True, hodlers["Total␣
 ↪Staked"]))
plt.title("Circles are Total Staked Amount: largest={:,.0f}; smallest={:,.0f}".
 ↪format(max(totalsfiltered), min(totalsfiltered)))
# plt.title("Blue is V1, Pink is V2 for the Latest Version used")
plt.show()
```



Accounts with more total staked and longer days staked appear during the V1 period. This might

indicate a difference between the accounts interacting with V1 vs. V2, or more likely indicate that many V2 accounts also interacted with V1.

However, it does appear that newer accounts are more reluctant to commit to staking more over longer periods.

3 Final Notes

This analysis is far from comprehensive. Additional analysis of user behavior and/or economic trends can reveal more actionable information for future versions of Flash.