Uniform Exploitation

In [61]:

```
#!/usr/bin/env python
# coding: utf-8

import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt

import requests

def pull(user_group, secret_key, arm):
        url = ('http://10.243.255.29:5787/pull_arm/%s/%s/%s' % (user_group,secret_key,str(arm)))

        while True:
                r = requests.get(url)
                if r.ok:
                        output = r.json()['result']
                        return(output)
```

In [97]:

```
arms = [str(x) for x in range(24)]
rewards = [0]*24
tries = [0]*24

result = pd.DataFrame({'arm': arms, 'reward': rewards, 'tries': tries})
```

In [18]:
```
dict = {"result":{"Arm":"4","NetReward":49382,"Pull":2047,"Reward":10},"status":200}
{'Arm': '1', 'NetReward': 49395, 'Pull': 2048, 'Reward': 13}
{'Arm': '11', 'NetReward': 49415, 'Pull': 2049, 'Reward': 20}
{'Arm': '2', 'NetReward': 49451, 'Pull': 2051, 'Reward': 9}
{'Arm': '0', 'NetReward': 53588, 'Pull': 2256, 'Reward': 27}
```

In [98]:
```
result
```

Out[98]:

	arm	reward	tries
0	0	0	0
1	1	0	0
2	2	0	0
3	3	0	0
4	4	0	0
5	5	0	0
6	6	0	0
7	7	0	0
8	8	0	0
9	9	0	0
10	10	0	0
11	11	0	0
12	12	0	0
13	13	0	0
14	14	0	0

15	15	0	0
16	16	0	0
17	17	0	0
18	18	0	0
19	19	0	0
20	20	0	0
21	21	0	0
22	22	0	0
23	23	0	0

In [99]:
```
def uniform_bandit(n):
    for j in range(4):
        for arm in arms:
            output_dict = pull('user24','IyqHJZcK',arm)
            result['reward'].loc[result['arm'] == arm] += output_dict['Reward']
            result['tries'].loc[result['arm'] == arm] += 1

    leftover_n = n - (3*len(arms))
    winning_arm = result['arm'].loc[result['reward'] == result['reward'].max()].values[0]
    for i in range(leftover_n):
        output_dict = pull('user24','IyqHJZcK',winning_arm)
        result['reward'].loc[result['arm'] == winning_arm] += output_dict['Reward']
        result['tries'].loc[result['arm'] == winning_arm] += 1
    return result
```

In [100…
```
uniform_bandit(180)
```

```
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  self._setitem_single_block(indexer, value, name)
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  self._setitem_single_block(indexer, value, name)
```

Out[100...

	arm	reward	tries
0	0	2769	112
1	1	65	4
2	2	29	4
3	3	83	4
4	4	27	4
5	5	57	4
6	6	26	4
7	7	72	4
8	8	101	4
9	9	44	4
10	10	60	4
11	11	85	4
12	12	44	4
13	13	44	4
14	14	78	4
15	15	28	4

16	16	35	4
17	17	84	4
18	18	78	4
19	19	59	4
20	20	82	4
21	21	94	4
22	22	55	4
23	23	79	4

In [101…

```
result.to_csv('reward_pull_2.csv')
```

In [93]:

```
result
```

Out[93]:

	arm	reward	tries
0	0	0	0
1	1	0	0
2	2	0	0
3	3	0	0
4	4	0	0
5	5	0	0
6	6	0	0
7	7	0	0
8	8	0	0

9	9	0	0
10	10	0	0
11	11	0	0
12	12	0	0
13	13	0	0
14	14	0	0
15	15	0	0
16	16	0	0
17	17	0	0
18	18	0	0
19	19	0	0
20	20	0	0
21	21	0	0
22	22	0	0
23	23	0	0

In [94]:

```
arm = '0'
output_dict = pull('user24','IyqHJZcK',arm)
result['reward'].loc[result['arm'] == arm] += output_dict['Reward']
result['tries'].loc[result['arm'] == arm] += 1
```

```
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  self._setitem_single_block(indexer, value, name)
```

In [96]:
```
output_dict
```

Out[96]:
```
{'Arm': '0', 'NetReward': 53588, 'Pull': 2256, 'Reward': 27}
```

In [ ]:
```

```

$\epsilon - Greedy$

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import random

import requests

def pull(user_group, secret_key, arm):
        url = ('http://10.243.255.29:5787/pull_arm/%s/%s/%s' % (user_group,secret_key,str(arm)))

        while True:
                r = requests.get(url)
                if r.ok:
                        output = r.json()['result']
                        return(output)
```

In [2]:
```
arms = [str(x) for x in range(24)]
rewards = [0]*24
tries = [0]*24
mean_reward = [0]*24

result = pd.DataFrame({'arm': arms, 'reward': rewards, 'tries': tries, 'mean_reward': mean_reward})
```

In [3]:
```
def epsilon_greedy(n, p):
    winning_arm = str(random.randint(0,23))
    arms.remove(winning_arm)

    for i in range(n):
        if np.random.uniform() <= p:
            output_dict = pull('user24','IyqHJZcK', random.choice(arms))

        else:
            output_dict = pull('user24','IyqHJZcK', winning_arm)

        result['reward'].loc[result['arm'] == output_dict['Arm']] += output_dict['Reward']
        result['tries'].loc[result['arm'] == output_dict['Arm']] += 1
        result['mean_reward'].loc[result['arm'] == output_dict['Arm']] = result['reward']/result['tries']

        new_winning_arm = result['arm'].loc[result['mean_reward'] == result['mean_reward'].max()].values[0]

        if new_winning_arm in arms:
            arms.remove(new_winning_arm)
            arms.append(winning_arm)

            winning_arm = new_winning_arm

        else:
            continue


    return result
```

In [4]:
```
epsilon_greedy(180, 0.3)
```

```
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
```

```
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  self._setitem_single_block(indexer, value, name)
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  self._setitem_single_block(indexer, value, name)
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  self._setitem_single_block(indexer, value, name)
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  self._setitem_single_block(indexer, value, name)
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  self._setitem_single_block(indexer, value, name)
/Users/salimwid/opt/anaconda3/lib/python3.8/site-packages/pandas/core/indexing.py:1637: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  self._setitem_single_block(indexer, value, name)
```

Out[4]:

	arm	reward	tries	mean_reward
0	0	2277	90	25.300000

1	1	0	0	0.000000
2	2	26	4	6.500000
3	3	35	2	17.500000
4	4	12	2	6.000000
5	5	30	2	15.000000
6	6	6	1	6.000000
7	7	66	4	16.500000
8	8	89	4	22.250000
9	9	18	2	9.000000
10	10	15	1	15.000000
11	11	43	2	21.500000
12	12	48	4	12.000000
13	13	17	2	8.500000
14	14	31	2	15.500000
15	15	45	6	7.500000
16	16	45	7	6.428571
17	17	15	1	15.000000
18	18	48	3	16.000000
19	19	23	1	23.000000
20	20	638	30	21.266667
21	21	84	4	21.000000
22	22	40	3	13.333333
23	23	67	3	22.333333

In [5]:
```
random.randint(0,23)
```

Out[5]: 2

In [7]:
```
result.to_csv('greedy_result.csv')
```

In [ ]:
```

```

UCB1

In [ ]:
```
import pandas as pd,\
  utils.server_pull as usp,\
  numpy as np
```

Intialise desired pulls, API secret

In [ ]:
```
desired_pulls = 2000
team = ''
group_key = ''
```

Create function for getting top linUCB index of all arms

In [ ]:
```
def get_max_ucb_of_arms(input_df, input_global_round):
# Get the mean reward of each arm, and count of runs so far
# TODO: currently not safe for historical runs, would need indexing on inputdf
  df_mean_reward = input_df.pivot_table(
    index='arm',
    values=['arm_reward','arm_pull'],
    aggfunc={'arm_reward': np.mean, 'arm_pull': np.max }
  ).reset_index()
  df_mean_reward['ucb_index'] = df_mean_reward.apply(
    lambda x: x['arm_reward'] + np.sqrt(
      (2 * np.log( input_global_round ) ) / (x['arm_pull'] + 1)
    ),
    axis=1
  )
  return df_mean_reward['ucb_index'].idxmax()
# TODO: there is a non trivial circumstance where this can return multiple rows, rather than a scalar;
#   could be improved with getting random as typebreaker
```

Get historical data

In [ ]:
```
df_historical_pulls = pd.read_csv(
  './data/pulls.csv',
  header='infer',
  index_col=False
)
```

Get current arm pulls so far

In [ ]:
```
existing_arm_pulls = df_historical_pulls['global_pull'].max()
```

safety check if in exploration phase

In [ ]:
```
if existing_arm_pulls >= 23:
  # pre compute UCB for best arm
  target_arm = get_max_ucb_of_arms( df_historical_pulls, existing_arm_pulls )
  for i in range(existing_arm_pulls+1,existing_arm_pulls+desired_pulls+1):
    # pull arm, get output
    arm_output = usp.pull(team,group_key,target_arm)
    arm_pull_count = df_historical_pulls[df_historical_pulls['arm'] == target_arm  ]['arm_pull'].count()
    # append output
    df_historical_pulls = df_historical_pulls.append(
      {'arm_pull': arm_pull_count, 'arm':target_arm,'global_pull':i ,'arm_reward':arm_output['Reward'] },
      ignore_index=True
    )
    # get next arm
    target_arm = get_max_ucb_of_arms( df_historical_pulls,  i )
else:
  raise
```

write out results

In [ ]:

```
df_historical_pulls.to_csv(
  './data/pulls_output.csv',
  index=False
)
```

TODO: get arm pulls for exploration phase

Analysis

In [1]:
```
import pandas as pd,\
  seaborn as sns,\
  matplotlib.pyplot as pypl,\
  numpy as np

#  plotly.express as px,\
```

In [2]:
```
pypl.style.use('bmh')
```

In [3]:
```
df_historical_pulls = pd.read_csv(
  '../dba5101_gp3/data/pulls_output.csv',
  header='infer',
  index_col=False
)
```

In [4]:
```
## add_uniform_algo

df_uniform_pulls_input = pd.read_json(
  '../dba5101_gp3/data/uniform_pulls_1.json',
  orient='records',
  lines=True
)
```

In [5]:
```
## add_greed_algo

df_greedy_pulls_input = pd.read_json(
  '../dba5101_gp3/data/greedy_pull_3.json',
  orient='records',
  lines=True
)
```

In [6]:
```
df_greedy_pulls_input['index_pull'] = df_greedy_pulls_input['Pull'].rank() - 1
df_uniform_pulls_input['index_pull'] = df_uniform_pulls_input['Pull'].rank() - 1
```

In [7]:
```
df_greedy_pulls = df_greedy_pulls_input[df_greedy_pulls_input['index_pull'] >= 1]
```

In [8]:
```
df_uniform_pulls =  df_uniform_pulls_input[df_uniform_pulls_input['index_pull'] >= (24*3 )]
```

In [9]:
```
df_ucb1_exploit = df_historical_pulls[df_historical_pulls['global_pull'] >= 24 ]
```

In [ ]:
```

```

In [10]:
```
df_uniform_pulls['local_pull'] = df_uniform_pulls['Pull'].rank() - 1
```

```
/var/folders/88/st0km2xx06b1nm8trg753nqm0000gn/T/ipykernel_19265/1785843218.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  df_uniform_pulls['local_pull'] = df_uniform_pulls['Pull'].rank() - 1
```

In [ ]:

In [11]:
```
df_greedy_pulls['local_pull'] = df_greedy_pulls['Pull'].rank() - 1
```

```
/var/folders/88/st0km2xx06b1nm8trg753nqm0000gn/T/ipykernel_19265/3050684879.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  df_greedy_pulls['local_pull'] = df_greedy_pulls['Pull'].rank() - 1
```

In [12]:
```
df_greedy_pulls['rolling_sum'] = df_greedy_pulls['Reward'].cumsum()

df_uniform_pulls['rolling_sum'] = df_uniform_pulls['Reward'].cumsum()

df_historical_pulls['rolling_sum'] = df_historical_pulls['arm_reward'].cumsum()
```

```
/var/folders/88/st0km2xx06b1nm8trg753nqm0000gn/T/ipykernel_19265/4265629572.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  df_greedy_pulls['rolling_sum'] = df_greedy_pulls['Reward'].cumsum()
/var/folders/88/st0km2xx06b1nm8trg753nqm0000gn/T/ipykernel_19265/4265629572.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  df_uniform_pulls['rolling_sum'] = df_uniform_pulls['Reward'].cumsum()
```

In [13]:
```
df_ucb1_exploit['rolling_sum'] = df_ucb1_exploit['arm_reward'].cumsum()
```

```
/var/folders/88/st0km2xx06b1nm8trg753nqm0000gn/T/ipykernel_19265/3231549739.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
ing-a-view-versus-a-copy
  df_ucb1_exploit['rolling_sum'] = df_ucb1_exploit['arm_reward'].cumsum()
```

In [14]:

```
df_comparative = df_ucb1_exploit.merge(
  right=df_uniform_pulls,
  left_on ='global_pull',
  right_on = 'local_pull',
  how='left' ,
  suffixes=('_ucb1', '_uniform')
)

df_comparative = df_comparative.merge(
  right=df_greedy_pulls,
  left_on ='global_pull',
  right_on = 'local_pull',
  how='left' ,
  suffixes=('', '_greedy')
)
```

In [15]:

```
df_comparative
```

Out[15]:

	global_pull	arm	arm_pull	arm_reward	next_chosen_arm	rolling_sum_ucb1	Arm	NetReward	Pull	Reward	index_pull	local_pu
0	24.0	3.0	1.0	17.0	NaN	17.0	0	194485	8959	27	96.0	24.
1	25.0	8.0	1.0	24.0	NaN	41.0	1	194501	8960	16	97.0	25.
2	26.0	8.0	2.0	19.0	NaN	60.0	2	194509	8961	8	98.0	26.
3	27.0	0.0	1.0	18.0	NaN	78.0	3	194533	8962	24	99.0	27.
4	28.0	8.0	3.0	20.0	NaN	98.0	4	194541	8963	8	100.0	28.
...	...	...	...	...	...	...	...	...	...	...	...	.
1995	2019.0	0.0	1821.0	22.0	NaN	48414.0	0	242273	10954	22	2091.0	2019.
1996	2020.0	0.0	1822.0	27.0	NaN	48441.0	0	242295	10955	22	2092.0	2020.
1997	2021.0	0.0	1823.0	31.0	NaN	48472.0	0	242315	10956	20	2093.0	2021.
1998	2022.0	0.0	1824.0	20.0	NaN	48492.0	0	242337	10957	22	2094.0	2022.
1999	2023.0	0.0	1825.0	29.0	NaN	48521.0	0	242366	10958	29	2095.0	2023.

2000 rows × 20 columns

In [19]:
```
df_comparative [df_comparative['global_pull'] <= 2000 ] [
  ['global_pull','rolling_sum_ucb1','rolling_sum_uniform','rolling_sum']
].rename(columns={'rolling_sum':'greedy_epsilon','rolling_sum_uniform':'uniform','rolling_sum_ucb1':'ucb1'}) .plot
  kind ='line',
  x = 'global_pull'
)



pypl.title("Comparision of UCB1 +  Uniform MAB + Greedy Episilon - Exploitation Phase Cumulative Reward")

pypl.gcf().set_size_inches(17, 10)

pypl.ylabel('Cumulative Reward')
pypl.xlabel('Pull')

#pypl.show()

pypl.savefig(fname='./model_cumulative_sum.png')

pypl.close('all')
```

In [ ]:

In [ ]:

In [4]:
```
# Add in historical pulls over all time

df_total_pulls = df_historical_pulls.groupby(['arm'])['global_pull'].count().reset_index()

df_total_pulls.rename(columns={"global_pull": "total_pulls"},inplace=True)

df_historical_pulls =  df_historical_pulls.merge(right =df_total_pulls , how ='left', on ='arm' )
```

In [5]:
```
df_arms = pd.DataFrame( {'arm': df_historical_pulls['arm'].unique().tolist()  } )

df_pulls = pd.DataFrame( {'global_pull': df_historical_pulls['global_pull'].unique().tolist()  } )

df_cartesian = df_arms.merge(right=df_pulls, how='cross')

df_cartesian= df_cartesian.merge( right=df_historical_pulls, on=['global_pull','arm'] ,how='left' )

df_cartesian['arm_reward'].fillna(value=0,inplace=True)

df_cartesian['cumulative_reward'] = df_cartesian.groupby(['arm'])['arm_reward'].cumsum(skipna=True)

df_cartesian['arm_pull'].ffill(inplace=True)
```

In [6]:
```
df_cartesian = df_cartesian[df_cartesian['global_pull'] >= 24]
```

In [7]:
```
#df_cartesian['ucb_index'] =
df_cartesian['contemporary_linucb'] = (df_cartesian['cumulative_reward'] / (df_cartesian['arm_pull'] + 1) )  + np.
  (2 * np.log( df_cartesian['global_pull']  ) ) / (df_cartesian['arm_pull'] + 1)
)

df_cartesian['ucb_mean'] = (df_cartesian['cumulative_reward'] / (df_cartesian['arm_pull'] + 1) )

df_cartesian['penalty'] = np.sqrt(
  (2 * np.log( df_cartesian['global_pull']  ) ) / (df_cartesian['arm_pull'] + 1)
)
```

In [8]:
```
sns.lineplot(
  x='global_pull',
  y='contemporary_linucb',
  hue='arm',
  data=df_cartesian[
    (df_cartesian['global_pull'] <= 220)
    & (df_cartesian['global_pull'] >= 24)
    & (df_cartesian['arm'].isin([0,3,8,23]) )
  ]
)

pypl.title("UCB Over Time")

pypl.gcf().set_size_inches(17, 10)

pypl.ylabel('UCB Index')
pypl.xlabel('Pull')

#pypl.show()

pypl.savefig(fname='./ucb_exploitation_line.png')

pypl.close('all')
```

In [9]:
```
df_cartesian[
  df_cartesian['global_pull'] == 2020
].sort_values(
  by=['contemporary_linucb'],
  ascending=False
)[['arm','arm_pull','cumulative_reward','contemporary_linucb','ucb_mean','penalty']].head(n=5)
```

Out[9]:

	arm	arm_pull	cumulative_reward	contemporary_linucb	ucb_mean	penalty
2020	0.0	1822.0	44651.0	24.584520	24.493143	0.091377
16188	7.0	0.0	19.0	22.901500	19.000000	3.901500
42500	20.0	0.0	19.0	22.901500	19.000000	3.901500
20236	9.0	0.0	19.0	22.901500	19.000000	3.901500
36428	17.0	1.0	40.0	22.758777	20.000000	2.758777

In [10]:

```
sns.lineplot(
  x='global_pull',
  y='contemporary_linucb',
  hue='arm',
#  color='b',
  data=df_cartesian[
    (df_cartesian['global_pull'] >= 24)
    & (df_cartesian['arm'].isin([0,7,17]) )
  ]
)

pypl.title("UCB Over Time - Candidates For Next Arm, Due To Uncertainity")

pypl.gcf().set_size_inches(17, 10)

pypl.ylabel('UCB Index')
pypl.xlabel('Pull')

#pypl.show()
pypl.savefig(fname='./ucb_future_line.png')

pypl.close('all')
```

In [11]:

```
f, ax = pypl.subplots(figsize=(17, 10))

sns.lineplot(
  x='global_pull',
  y='ucb_mean',
  hue='arm',
#  color='b',
  data=df_cartesian[
    (df_cartesian['global_pull'] >= 24)
    & (df_cartesian['arm'].isin([0,7,17]) )
  ]
)

sns.lineplot(
  x='global_pull',
  y='penalty',
  hue='arm',
  data=df_cartesian[
    (df_cartesian['global_pull'] >= 24)
    & (df_cartesian['arm'].isin([0,7,17]) )
  ]
)


pypl.title("Arm Mean & Penalty Term Over Time")

pypl.ylabel('UCB Index Component Quantity')
pypl.xlabel('Pull')


#pypl.show()
pypl.savefig(fname='./ucb_penalty_line.png')

pypl.close('all')
```

In [12]:

```
#Create combo chart
fig, ax1 = pypl.subplots(figsize=(17,10))


#bar plot creation
ax1.set_title('UCB Index & Reward - Exploitation Phase', fontsize=16)
ax1.set_xlabel('Pull')
ax1.set_ylabel('Reward')
ax1 = sns.scatterplot(x='global_pull', y='arm_reward', hue='arm', data = df_historical_pulls[
(df_historical_pulls['global_pull'] <= 250)
& (df_historical_pulls['global_pull'] >= 24)
#& (df_historical_pulls['arm'].isin([0,3,8,23]) )
])
ax1.tick_params(axis='y')
#specify we want to share the same x-axis
ax2 = ax1.twinx()

#line plot creation
ax2.set_ylabel('UCB Index')
ax2 = sns.lineplot(
  x='global_pull',
  y='contemporary_linucb',
  hue='arm',
  data=df_cartesian[
    (df_cartesian['global_pull'] <= 250)
    & (df_cartesian['global_pull'] >= 24)
#    & (df_cartesian['arm'].isin([0,3,8,23]) )
  ]
)


#show plot
#pypl.show()
pypl.savefig(fname='./ucb_index_reward_line.png')

pypl.close('all')
```

In [48]:
```
#sns.scatterplot( x='global_pull', y='arm_reward',   hue='arm',
#  data = df_historical_pulls[   (df_historical_pulls['global_pull'] <= 250)
#    & (df_historical_pulls['global_pull'] >= 24) ])

#pypl.gcf().set_size_inches(17, 10)

#pypl.show()

#pypl.close('all')
```

In [47]:
```
#sns.displot(  data=df_historical_pulls[ df_historical_pulls['global_pull'] >= 24] ,
#  x="arm_reward", hue="arm", kind="kde")

#pypl.gcf().set_size_inches(17, 10)

#pypl.show()

#pypl.close('all')
```

In [13]:

```
sns.violinplot(
  cut=0,
  inner='quartiles',
  data=df_historical_pulls[ (df_historical_pulls['global_pull'] >= 24) & (df_historical_pulls[ 'total_pulls'] > 2
  x='arm',
  y='arm_reward',
)

pypl.gcf().set_size_inches(17, 10)
pypl.ylim([0,45])

pypl.title("Arm Reward Violinplot, Exploitation Phase - With Quartiles")
pypl.ylabel('Reward')
pypl.xlabel('Arm')

#pypl.show()
pypl.savefig(fname='./reward_violin.png')

pypl.close('all')
```