In [15]:
```
import plotly.io as pio
pio.renderers.default = "notebook"
```

In [16]:
```
from dwave.system import DWaveSampler, EmbeddingComposite
from scheduls import *
import neal
import plotly.express as px
from tabu import TabuSampler
from dwave.gbsolv import QBSolv
from anneal_solver import solveloq_dwave
```

Job-Shop-Scheduling mit Annealer

In [17]:
```
#Flag damit nicht unbeabsichtigt der quantumcomputer verwendet wird
use_quantum=True
```

Erstelle den Ablaufplan

In [18]:
```
#maximale Zeit
s = Schedule(time_max=10)
# maximal 3 maschinen
# bitte nur benötigte anzahl der maschinen angeben, wir haben keinen preprocessing step um unnötige maschinen h
s.build_machines(2)

#Jobs
# 1:(2,3) bedeutet auf der maschine 1 wird die arbeit 2 zeiteinheiten am stück durchgeführt und 3 mal wiederho
# es sollte wenn möglich nur die dauer auf 1 begrenzt sein, da wir keine zeit hatten die anderen optionen gut 
# frühst möglicher anfangszeitpunkte können zwar gesetzt werden, dies bricht zurzeit noch einige feature und m
s.create_job([0: (1, 1), 1: (1, 1)], parallel_operations=1)
s.create_job([1: (1, 2), 0: (1, 1)], parallel_operations=1)
s.create_job([0: (1, 1), 1: (1, 1)], parallel_operations=1)


print(s)
```

```
--------------------------
Arbeitsplan
Job 0 Operationen:
Machine(nr=1, name='m0', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 1
Machine(nr=1, name='m1', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 1

Job 1 Operationen:
Machine(nr=1, name='m1', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 2
Machine(nr=0, name='m0', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 1

Job 2 Operationen:
Machine(nr=0, name='m0', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 1
Machine(nr=1, name='m1', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 1

--------------------------
```

Qubostraf Terme und Berechnung des QUBO

In [19]:
```
qubo=Qubo(s)

#start term
qubo.penalty_terms[1]=1

# nur eine machine/job pro job/machine gleichzeitig
qubo.penalty_terms[2]=1
qubo.penalty_terms[3]=1

# 4 hat keine bedeutung, war ursprünglich für die deadlines vorgesehen
# da wir diese aber ohne strafterme lösen, ist dieses feld leer

# just in time
qubo.penalty_terms[5]=6


qubo.calculate_qubo()
print(f"Qubits: {len(qubo.h)*"}

# qubo berechnung falls mehrere Maschinen oder Jobs gleichzeitig verwendet werden sollten - (dies muss im sched
# qubo.calculate_qubo_virtuell()
```

```
Qubits: 60
```

Visualisierung des QUBOs

In [20]:
```
qubo.plot_qubo()
```

In [21]:
```
qubo.plot_connections_qubo_states()
```

Out[21]:

Strafterme 1-3

In [22]:
```
qubo.plot_qubo_terms()
```

Out[22]:

Simulation mit QBSolve

In [23]:
```
response = QBSolv().sample_qubo(qubo.Q, num_repeats=1000)
sol3 = response.samples()[0]
*temp, _ = qubo.interpret_solution_dict(sol3)
print(temp)
qubo.plot_solution()
```

```
[(8, 17, 27, 38, 39, 49, 56), {state(j:0,m:0,t:8), state(j:0,m:1,t:7), state(j:1,m:0,t:7), state(j:1,m:1,t:8),
state(j:2,m:1,t:9), state(j:2,m:0,t:9), state(j:2,m:1,t:6)}, -7.08, {i: -9, 2: 0, 3: 0, 4: 0, 5: 3.1920000000000
00041, 9}
```

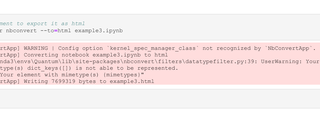
Simulation mit Tabusampler von Dwave

In [24]:
```
samples = TabuSampler().sample_qubo(qubo.Q, num_reads=1000)
sol2 = samples.samples()[0]
*temp, _ = qubo.interpret_solution_dict(sol2)
print(temp)
qubo.plot_solution()
```

Simulation mit Simulated Annealer

In [ ]:
```
sampler = neal.SimulatedAnnealingSampler()
samplerset = sampler.sample_qubo(qubo.Q, num_reads=100)
sol1=samplerset.samples()[0]
*temp,_=qubo.interpret_solution_dict(sol1)
print(temp)
qubo.plot_solution()
```

```
[(3, 7, 12, 18, 19, 24, 26), {state(j:0,m:0,t:3), state(j:0,m:1,t:2), state(j:1,m:0,t:2), state(j:1,m:1,t:3), s
tate(j:1,m:1,t:4), state(j:2,m:0,t:4), state(j:2,m:1,t:3)}, -5.1599999999999, {i: -9, 2: 0, 3: 0, 4: 0, 5: 3.
840000000000071, 9}
```



Quantum Computer von Dwave

In [ ]:
```
if use_quantum:
    answer = solveloq_dwave(qubo,samples=10000)
    *temp2, _ = qubo.interpret_solution_dict(
        [x[ y for x, y in answer.samples()[0].items()])
    print(temp2)
qubo.plot_solution()
```

```
[(3, 7, 22, 14, 16, 18, 29), {state(j:0,m:0,t:3), state(j:0,m:1,t:2), state(j:2,m:0,t:2), state(j:1,m:0,t:4), s
tate(j:1,m:1,t:1), state(j:1,m:1,t:3), state(j:2,m:1,t:4)}, -5.16, {i: -9, 2: 0, 3: 0, 4: 0, 5: 3.841, 9}
```


Reverse Annealing mit Dwave

In [ ]:
```
if use_quantum:
    samplerq = DWaveSampler(solver=dict(qpu=True))
    samplerq = EmbeddingComposite(samplerq)
    init_qubit=answer.samples()[0]
    reverse_schedule = [[0.0, 1.0], [10.0, 0.0], [20, 1.0]]
    reverse_anneal_params = dict(anneal_schedule=reverse_schedule,
                                 initial_state=init_qubit,
                                 reinitialize_state=True)
    reverse_responses=samplerq.sample_qubo(
        {x: y for x, y in qubo.Q.items()} if y != 0), num_reads=1000, **reverse_anneal_params)
    *temp3, _ = qubo.interpret_solution_dict(
        [x[ y for x, y in answer.samples()[0].items()])
    print(temp3)
qubo.plot_solution()
```

```
[(3, 7, 22, 14, 16, 18, 29), {state(j:0,m:0,t:3), state(j:0,m:1,t:2), state(j:2,m:0,t:2), state(j:1,m:0,t:4), s
tate(j:1,m:1,t:1), state(j:1,m:1,t:3), state(j:2,m:1,t:4)}, -5.16, {i: -9, 2: 0, 3: 0, 4: 0, 5: 3.841, 9}
```


In [ ]:
```
# uncomment to export it as html
!jupyter nbconvert --to=html example3.ipynb
```

```
[NbConvertApp] WARNING | Config option `kernel_spec_manager_class` not recognized by `NbConvertApp`.
[NbConvertApp] Converting notebook example3.ipynb to html
D:\Anaconda3\envs\Quantum\lib\site-packages\nbconvert\filters\datatypefilter.py:39: UserWarning: Your element w
ith mimetype(s) dict_keys([]) is not able to be represented.
  warn("Your element with mimetype(s) {mimetypes} "
[NbConvertApp] Writing 7699319 bytes to example3.html
```

In [ ]:
```

```