

```
In [1]: import plotly.io as pio  
pio.renderers.default = "notebook"
```

```
In [2]: from dwave.system import DWaveSampler, EmbeddingComposite  
from schedule import *  
import neal  
import plotly.express as px  
from tabu import TabuSampler  
from dwave_qbsolv import QBSolv  
from anneal_solver import solvelog_dwave
```



Job-Shop-Scheduling mit Annealer

```
In [3]: #Flag damit nicht unbeabsichtigt der quantumcomputer verwendet wird  
use_quantum=True
```

Erstelle den Ablaufplan

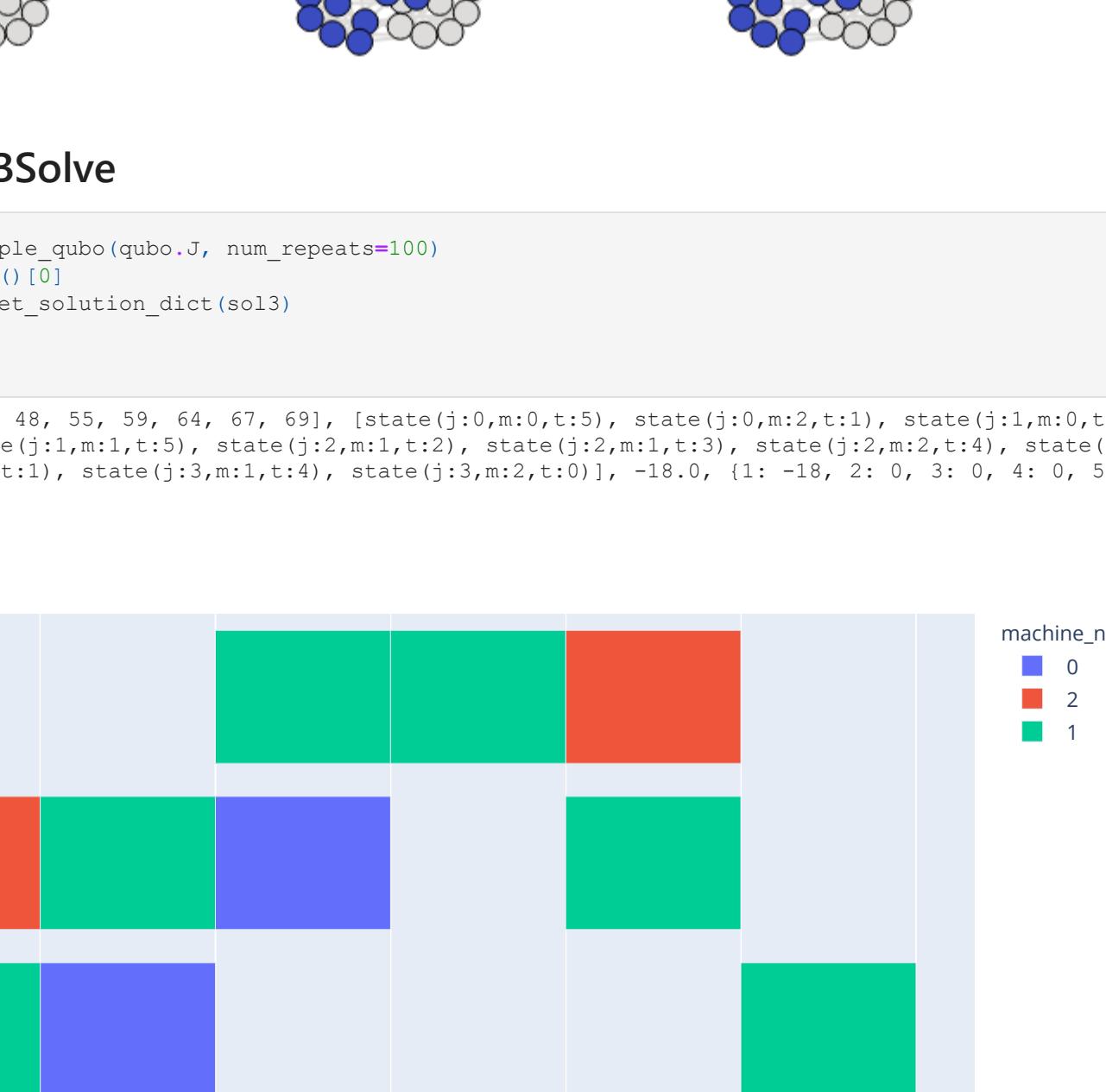
```
In [4]: #maximale Zeit  
s = Schedule(time_max=6)  
# bitte nur benötigte anzahl der maschinen angeben, wir haben keinen preprocessing step um unnötige maschinen zu verhindern  
s.build_machines(3)  
  
#Jobs  
s.create_job(({0: (1, 1), 2: (1, 1)}))  
s.create_job(({0: (1, 1), 1: (1, 2)}))  
s.create_job(({1: (1, 2), 2: (1, 1)}))  
s.create_job(({0: (1, 1), 1: (1, 2), 2: (1, 1)}))  
  
print(s)  
-----  
Arbeitsplan  
Job 0 Operationen:  
Machine(nr=0, name='m0', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 1  
Machine(nr=2, name='m2', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 1  
  
Job 1 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: 2  
  
Job 2 Operationen:  
Machine(nr=1, name='m1', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 2  
Machine(nr=2, name='m2', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 1  
  
Job 3 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: 2  
Machine(nr=2, name='m2', parallel_operations=1, start_time=0) Dauer: 1 Anzahl: 1
```

Qubostrafe Terme und Berechnung des QUBO

```
In [5]: qubo=Qubo(s)  
  
#start term  
qubo.penalty_terms[1]=1  
  
# nur eine maschine/job pro job/maschine 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]=0  
  
qubo.calculate_qubo()  
print(f"qubits: {len(qubo.h)}")  
  
# qubo berechnung falls mehrere Maschinen oder Jobs gleichzeitig verwendet werden sollten - (dies muss im sche  
# qubo.calculate_qubo_virtuell()  
  
qubits: 75
```

Visualisierung des QUBOs

```
In [6]: qubo.plot_qubo()
```



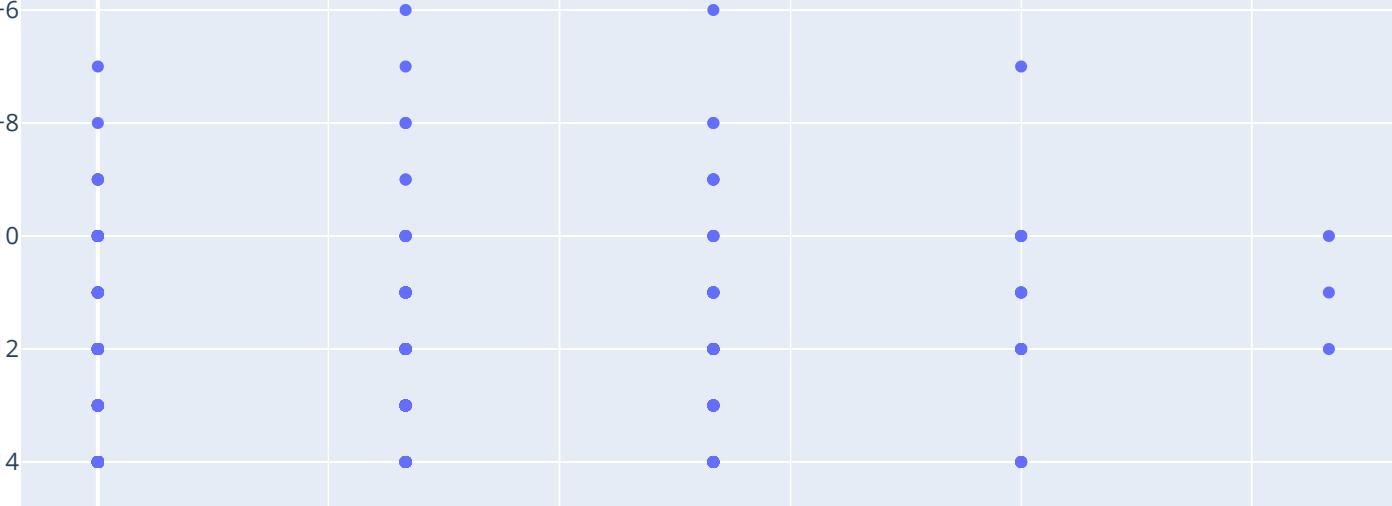
```
In [7]: qubo.plot_connections_qubo_states(250)
```



Strafterme 1-3

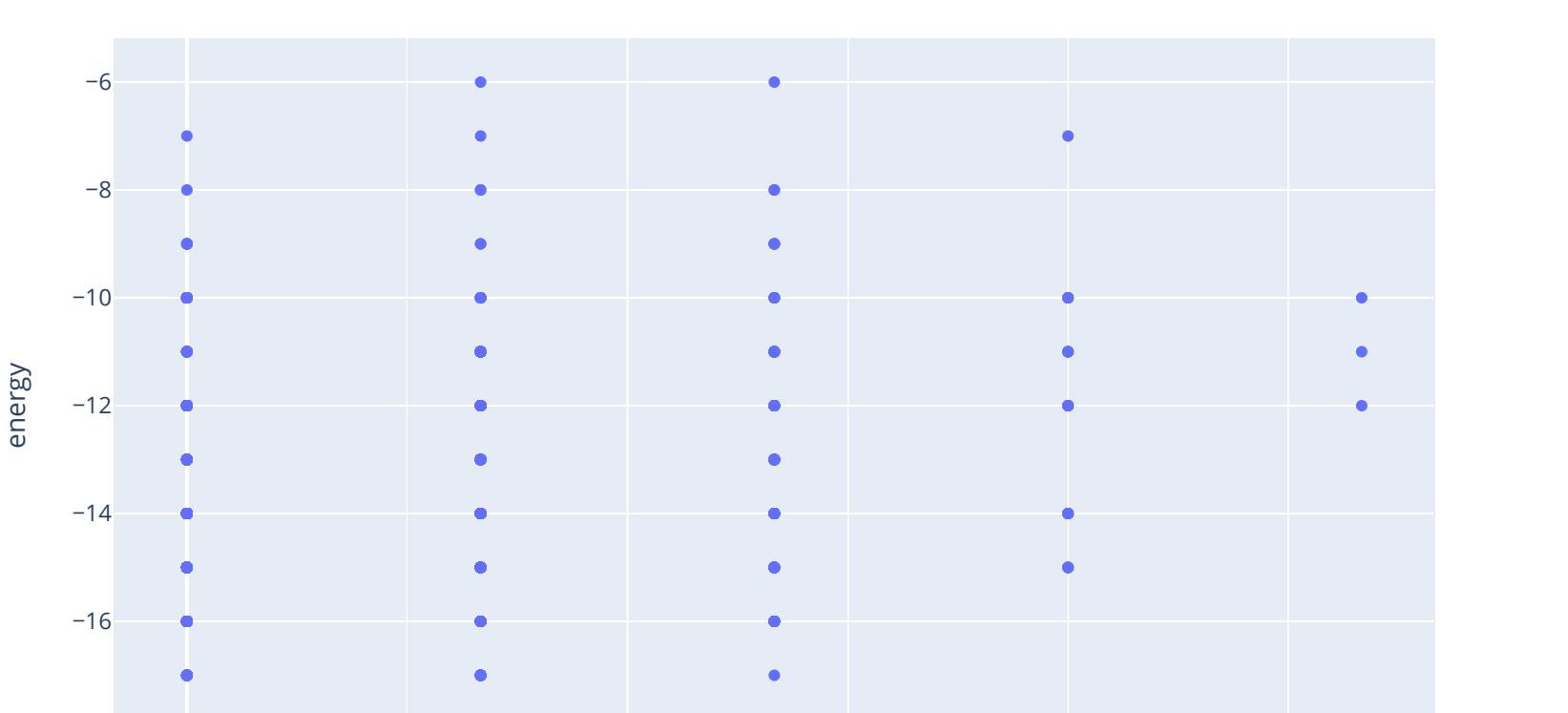
```
In [8]: qubo.plot_qubo_terms(250)
```

```
Out[8]:
```



```
In [9]: response = QBSolv().sample_qubo(qubo.J, num_repeats=100)  
sol3 = response.samples() [0]  
*temp, _ = qubo.interpret_solution_dict(sol3)  
print(temp)  
qubo.plot_solution()
```

```
[{5, 14, 20, 25, 30, 47, 48, 55, 59, 64, 67, 69}, {state(j:0,m:0,t:5), state(j:0,m:2,t:1), state(j:1,m:0,t:1),  
Machine(nr=0, name='m0', parallel_operations=1, start_time=0), state(j:1,m:1,t:0), state(j:2,m:1,t:2), state(j:2,m:1,t:3), state(j:2,m:2,t:4), state(j:3,m:0,t:2), state(j:3,m:1,t:1), state(j:2,m:1,t:3), state(j:3,m:1,t:4), state(j:3,m:2,t:0)}, -18.0, {1: -18, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0}, 18]
```



```
In [10]: if use_quantum:  
    answer = solvelog_dwave(qubo,1000)  
    *temp, _ = qubo.interpret_solution_dict(  
        {x: y for x, y in answer.samples() [0].items()})  
    print(temp)  
    qubo.plot_solution()
```

```
[{2, 13, 57, 24, 52, 71, 25, 27, 64, 48, 49, 68}, {state(j:0,m:0,t:5), state(j:0,m:2,t:1), state(j:1,m:0,t:0),  
Machine(nr=1, name='m1', parallel_operations=1, start_time=0), state(j:1,m:1,t:5), state(j:2,m:2,t:1), state(j:2,m:1,t:2), state(j:3,m:1,t:1), state(j:2,m:1,t:3), state(j:2,m:1,t:4), state(j:3,m:1,t:5)}, -18.0, {1: -18, 2: 0, 3: 0, 4: 0, 5: 0, 6: 0, 7: 0}, 18]
```



```
In [11]: df=answer.to_pandas_dataframe()
```

```
In [12]: px.histogram(df,"chain_break_fraction")
```



```
In [13]: px.scatter(df, y="energy", x="chain_break_fraction")
```



```
In [14]: # uncomment to export it as html  
#!jupyter nbconvert --to=html example2.ipynb
```

```
In [ ]:
```