Solution Description

Problem Statement:

Buildability Constraint

- There are 'p' number constraints.
- Each constraint rule is an Boolean expression is a 'm' length long Boolean expression.

$$\phi_k(b_{i1}, \dots, b_{im}) = 1 , \qquad \text{for } i = 1, \dots, n \ \text{ and } \ k = 1, \dots, p$$

▼ 'm' is the number of features that are tested for each vehicle

▼ A binary variable b is used to symbolise if a car has a feature or not.

- There are 'n' vehicles

$$\bigwedge_{i=1}^{n} \bigwedge_{k=1}^{p} \phi_k(b_{i1}, \dots, b_{im})$$

Testing Constraint

- Each test requires a vehicle with a particular constraint rule to be satisfied(A boolean expression). for a test 'l' :

$$\psi_l(b_{i1}, \dots, b_{im}) = 1 \text{ for } l = 1, \dots, q.$$

- All cars need to fulfil the **buildability constraints**.
- It's enough to have at least one car that fulfils the testing requirements.

Problem Solution Approach

SAT Problem:

- Is there any configuration of features for a number of n cars so that they satisfy the buildability constraints (1) and that all testing requirements (4) are met by at least one car.

$$\left(\bigwedge_{k=1}^{p} \bigwedge_{i=1}^{n} \phi_k(b_{i1}, \dots, b_{im}) \right) \wedge \left(\bigwedge_{l=1}^{q} \bigvee_{i=1}^{n} \psi_l(b_{i1}, \dots, b_{im}) \right) = 1$$

- There may be test requirements that need to be met by more than one vehicle in a real-world case, say by d vehicles.

Max - SAT:

- Find the constellation of features for n test vehicles, each complying with the configuration rules, so weighted number q of the fulfilled testing requests is maximized.

$$\underset{b_{ij}}{\text{maximize}}\, f(b_{11}, \dots, b_{nm}) = \sum_{l=1}^{q} w_l \cdot \bigvee_{i=1}^{n} \psi_l(b_{i1}, \dots, b_{im})$$

subject to the configuration constraints (1), with $wl > 0$ being the weight of the l-th test for l = 1,…, q. The weights may vary, making specific tests have a higher priority than others.

- Solve to get the feautures set that has maximum benefit

Scheduling Tests(Job_Shop Problem):

- After getting the set of features that needs to be present from solving the Partial Weighted Max-SAT , we need to schedule these tests
- Each test has a time frame in which it needs to be performed: t_l^{start}, t_l^{end} , for all l in q
- Scheduling is done on day-to-day basis

Quantum Solution Approaches:

- Using PySAT solver to simplify the problem to make it reach a scalable level(Tseitin Algorithm)
- QAOA(Solved using Pennylane Feedback Approach) for Max SAT
- Using Annealling Model to solve the scheduling problem.

Target :

The main metric considered for a new approach evaluation is the innovative character of the solution and how efficient it scales (i.e. a solution with lower hardware requirements).

Symbol Table:

Symbol	Meaning
n	Number of test vehicles
m	Number of different available features
v_i	Vehicle i
a_i	Feature i
b_{ij}	Vehicle v_i has feature a_j - boolean / binary variable
$\phi_k(b_{i1}, \ldots, b_{im})$	configuration constrain k for vehicle i as a Boolean expression of features
p	number of configuration constraints
$\psi_l(b_{i1}, \ldots, b_{im})$	test constrain l for vehicle i as a Boolean expression of features
q	number of test constraints

Sample Solution:

- **Feature Set:**

Car feature	Meaning
a_1	small engine
a_2	big engine
a_3	strong sub-woofer
a_4	seat heating
a_5	comfort seating

- **Test Constraints:**

$$\psi_{breaks}(b_{i1}, \ldots, b_{im}) = b_{i1}$$

$$\psi_{breaks2}(b_{i1}, \ldots, b_{im}) = b_{i2}$$

$$\psi_{Noise\_level}(b_{i1}, \ldots, b_{im}) = b_{i2} \land b_{i3}$$

$$\psi_{Bass\&Seat\_interaction}(b_{i1}, \ldots, b_{im}) = b_{i3} \land b_{i4}$$

$$\psi_{Power\_consumption}(b_{i1}, \ldots, b_{im}) = b_{i2} \land b_{i3} \land b_{i5}$$

- **Classical Solution** : Feauture a_2 and a_3
- **Quantum Solution:**

```
 python test.py
c Number of variables occuring in the formula: 4 max variable = 5 -> remapping
-0.000000 iter=6 maxdiff=0.000001
c first lower bound: 0
o 0
c 1 branches 3 propagates
c total generalized unit propagation = 4, success = 100.00%
s OPTIMUM FOUND
c Optimal Solution = 0
v -2 1 4 -3 5
[-1, 1, 1, -1]
```

- ***There are 3 cars which satisfy this as thier buildibility constraint.***

- **Schedulling Solution:**

```
Leap IDE /workspace/job-shop-scheduling $ /usr/local/bin/python /workspace/job-shop-scheduling/
Jobs and their machine-specific tasks:
vehicle1 : [('Test1', 2), ('Test2', 1)]
vehicle2 : [('Test1', 1)]
vehicle3 : [('Test2', 2)]

Jobs and the start times of each task:
vehicle1 : [0, 2]
vehicle2 : [2]
vehicle3 : [0]
Leap IDE /workspace/job-shop-scheduling $
```