The goal is to come up with a optimization approach that allows to run one full off-line approach for one Job and then customize the optimal solution found for another Job.

If we are able to measure the cpu/memory utilization of an optimal spark configuration for a particular job,

What is the request rate for a job? Number of tenants?

Ideal throughput = Cpu / Nr of tenants

Then the expected cpu usage of another job for that spark configuration

Expected cpu usage = Ideal Throughput \* Nr of tenants of another job

Workload complexity = Actual cpu usage / expected cpu usage

RU = number of tenants \* Ideal Throughput \* Workload complexity

Horizontal scaling

F1(nr of tenants-Job1) = (x1,x2,x3) => RU = x1.a1 + x2.a2 + x3.a3

Transition costs: V x in Dom(F1): Minimize F1(x) – F1(x-1) and F1(x+1) – F1(x)

Calucate F2(nr of tenants-Job2) = RU = y1.a1 + y2.a2 + y3.a3

Find optimal Y = (y1,y2,y3) for A = (a1,a2,a3) so that that Y.A = min {Z = (z1,z2,z3) | Z.A > RU-job2(tenants)}

Fit a non-linear function to