Task allocation

Aim

Balance the work load of workers during task allocation

Formulation

Let W be the set of all workers

Let T be the set of all tasks

The load of tasks assigned to

a worker $\mathrm{n} \in W$ is

$$\sum_{m \in T} c_m \, x_{n,m}$$

where $x_{n,m}$ = 1 if task m is assigned to worker m and 0 otherwise.

c_m is the job load of task m.

Formulation (cont)

The task load of a worker $\mathrm{n} \in W$

after allocation is:

$$l_n + \sum_{m \in T} c_m \, x_{n,m}$$

Where l_n is the current work load of worker n.

Task allocation

Formulation (cont)

We formulate the objective function to minimise the highest work load among all workers.

$$\min(\max(\ l_n + \sum_{m \in T} c_m\, x_{n,m}),\ \forall m \in T), \forall n \in W$$

For convenience, we introduce an auxiliary variable z and formulate the problem as:

Min z,

s.t. z >= $l_n + \sum_{m \in T} c_m\, x_{n,m} \quad \forall m \in T, \forall n \in W$ (Guarantee that workload of all worker smaller than z)

$$\sum_{n \in W} x_{n,m} = 1 \quad \forall m \in T$$ (Guarantee all tasks assigned to 1 worker)

Code

Task_allocation.py *pyomo solve task_allocation.py --solver=glpk*

```
from pyomo.environ import *

N_worker = 3
N_task = 10

worker_cur_load = {1:1, 2:2, 3:12}

task_load = {1:1, 2:2, 3:3, 4:4, 5:1, 6:2, 7:4, 8:5, 9:2, 10:2}

model = ConcreteModel()

model.workers = range(1,N_worker+1)
model.tasks = range(1,N_task+1)

model.z = Var( within=PositiveIntegers )

model.x = Var( model.workers, model.tasks, within=Binary )

model.obj = Objective(expr=model.z, sense=minimize)

model.aux_z = ConstraintList()

for n in model.workers:
    model.aux_z.add(model.z >= worker_cur_load[n] + \
    sum(model.x[n,m]*task_load[m] for m in model.tasks))

model.single_x = ConstraintList()

for m in model.tasks:
    model.single_x.add( sum( model.x[n,m] for n in model.workers ) == 1.0 )
```

Predefined workload for each worker

Predefined load for each task

objective

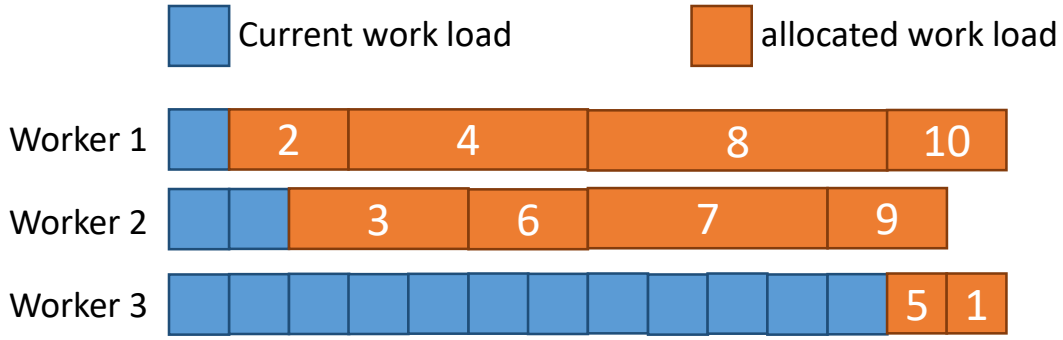
constraints

Example

```
worker_cur_load = {1:1, 2:2, 3:12}

task_load = {1:1, 2:2, 3:3, 4:4, 5:1, 6:2, 7:4, 8:5, 9:2, 10:2}
```

Output:




```
errorcode: 0
retval: instance: <pyomo.core.base.PyomoModel.ConcreteModel object at 0x00000277B840A990>
local:
    time_initial_import: 2.156989574432373
    usermodel: <module 'task_allocation' from 'C:\\Users\\16808949_admin\\Documents\\KaplanMeier\\task_allocation.py'>
options: <pyutilib.misc.config.ConfigBlock object at 0x00000277B83E5728>
results: {'Problem': [{'Name': 'unknown', 'Lower bound': 14.0, 'Upper bound': 14.0, 'Number of objectives': 1, 'Number of const
raints': 14, 'Number of variables': 32, 'Number of nonzeros': 64, 'Sense': 'minimize'}], 'Solver': [{'Status': 'ok', 'Terminati
on condition': 'optimal', 'Statistics': {'Branch and bound': {'Number of bounded subproblems': '11', 'Number of created subprob
lems': '11'}}, 'Error rc': 0, 'Time': 0.0609984397888183 6}], 'Solution': [OrderedDict([('number of solutions', 1), ('number of
solutions displayed', 1)])], {'Gap': 0.0, 'Status': 'optimal', 'Message': None, 'Problem': {}, 'Objective': {'obj': {'Value': 1
4.0}}, 'Variable': {'z': {'Value': 14.0}, 'x[1,1]': {'Value': 0.0}, 'x[1,2]': {'Value': 1.0}, 'x[1,3]': {'Value': 0.0}, 'x[1,
4]': {'Value': 1.0}, 'x[1,5]': {'Value': 0.0}, 'x[1,6]': {'Value': 0.0}, 'x[1,7]': {'Value': 0.0}, 'x[1,8]': {'Value': 1.0}, 'x
[1,9]': {'Value': 0.0}, 'x[1,10]': {'Value': 1.0}, 'x[2,1]': {'Value': 0.0}, 'x[2,2]': {'Value': 0.0}, 'x[2,3]': {'Value': 1.
0}, 'x[2,4]': {'Value': 0.0}, 'x[2,5]': {'Value': 0.0}, 'x[2,6]': {'Value': 1.0}, 'x[2,7]': {'Value': 1.0}, 'x[2,8]': {'Value':
0.0}, 'x[2,9]': {'Value': 1.0}, 'x[2,10]': {'Value': 0.0}, 'x[3,1]': {'Value': 1.0}, 'x[3,2]': {'Value': 0.0}, 'x[3,3]': {'Valu
e': 0.0}, 'x[3,4]': {'Value': 0.0}, 'x[3,5]': {'Value': 1.0}, 'x[3,6]': {'Value': 0.0}, 'x[3,7]': {'Value': 0.0}, 'x[3,8]': {'V
alue': 0.0}, 'x[3,9]': {'Value': 0.0}, 'x[3,10]': {'Value': 0.0}}, 'Constraint': {}}]}
```

Task allocation (with task types and competencies)

Formulation (cont)

$s_{n,a}$ is the competency of worker n on task type a

If work n is competent at task type a, $s_{n,a}$ = 1,

else $s_{n,a}$ = 0.

To ensure that all tasks assigned to 1 worker with the right competency, we add another constraint:

$$\sum_{n \in W} s_{n,a} x_{n,m} = 1 \qquad \forall m \in T$$

Code (with task types and competencies)

Task_allocation_r1.py *pyomo solve task_allocation_r1.py --solver=glpk*

```
from pyomo.environ import *

N_worker = 3
N_task = 10

worker_cur_load = {1:1, 2:2, 3:12}

task_load = {1:1, 2:2, 3:3, 4:4, 5:1, 6:2, 7:4, 8:5, 9:2, 10:2}

task_type = {1:1, 2:2, 3:3, 4:2, 5:2, 6:3, 7:1, 8:2, 9:3, 10:1}

#competence worker,task_type
competence = {(1, 1): 1, (1, 2): 1, (1, 3): 0,
              (2, 1): 1, (2, 2): 0, (2, 3): 1,
              (3, 1): 0, (3, 2): 1, (3, 3): 1}

model = ConcreteModel()

model.workers = range(1,N_worker+1)
model.tasks = range(1,N_task+1)

model.z = Var( within=PositiveIntegers )

model.x = Var( model.workers, model.tasks, within=Binary )

model.obj = Objective(expr=model.z, sense=minimize)

model.aux_z = ConstraintList()

for n in model.workers:
    model.aux_z.add(model.z >= worker_cur_load[n] + \
    sum(model.x[n,m]*task_load[m] for m in model.tasks))

model.single_x = ConstraintList()

for m in model.tasks:
    model.single_x.add( sum( model.x[n,m] for n in model.workers ) == 1.0 )

model.competence_req = ConstraintList()

for m in model.tasks:
    model.competence_req.add( sum( model.x[n,m]*competence[n,task_type[m]] for n in model.workers ) == 1.0 )
```

Task type

Define competency of worker

New constraints to ensure that all tasks assigned to worker with competency

Example (with task types and competencies)

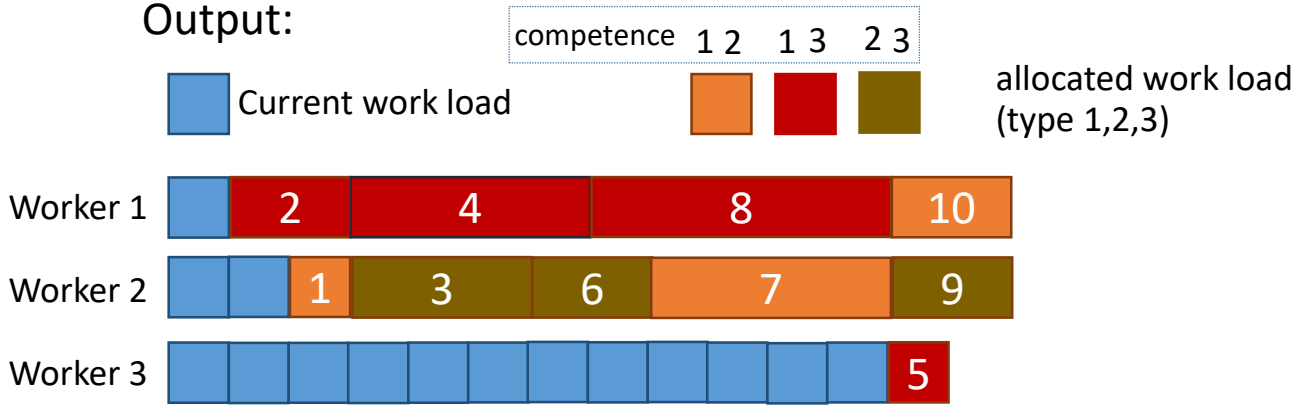
```
worker_cur_load = {1:1, 2:2, 3:12}

task_load = {1:1, 2:2, 3:3, 4:4, 5:1, 6:2, 7:4, 8:5, 9:2, 10:2}

task_type = {1:1, 2:2, 3:3, 4:2, 5:2, 6:3, 7:1, 8:2, 9:3, 10:1}

#competence worker,task_type
competence = {(1, 1): 1, (1, 2): 1, (1, 3): 0,
              (2, 1): 1, (2, 2): 0, (2, 3): 1,
              (3, 1): 0, (3, 2): 1, (3, 3): 1}
```

Output:




```
errorcode: 0
retval: instance: <pyomo.core.base.PyomoModel.ConcreteModel object at 0x000001D1E01CCBD0>
local:
    time_initial_import: 2.351915121078491
    usermodel: <module 'task_allocation_r1' from 'C:\\Users\\16808949_admin\\Documents\\KaplanMeier\\task_allocation_r1.py'>
options: <pyutilib.misc.config.ConfigBlock object at 0x000001D1E0198728>
results: {'Problem': [{'Name': 'unknown', 'Lower bound': 14.0, 'Upper bound': 14.0, 'Number of objectives': 1, 'Number of const
raints': 24, 'Number of variables': 32, 'Number of nonzeros': 84, 'Sense': 'minimize'}], 'Solver': [{'Status': 'ok', 'Terminati
on condition': 'optimal', 'Statistics': {'Branch and bound': {'Number of bounded subproblems': '11', 'Number of created subprob
lems': '11'}}, 'Error rc': 0, 'Time': 0.1385359764099121}], 'Solution': [OrderedDict([('number of solutions', 1), ('number of s
olutions displayed', 1)])], {'Gap': 0.0, 'Status': 'optimal', 'Message': None, 'Problem': {}, 'Objective': {'obj': {'Value': 14.
0}}, 'Variable': {'z': {'Value': 14.0}, 'x[1,1]': {'Value': 0.0}, 'x[1,2]': {'Value': 1.0}, 'x[1,3]': {'Value': 0.0}, 'x[1,4]':
{'Value': 1.0}, 'x[1,5]': {'Value': 0.0}, 'x[1,6]': {'Value': 0.0}, 'x[1,7]': {'Value': 0.0}, 'x[1,8]': {'Value': 1.0}, 'x[1,
9]': {'Value': 0.0}, 'x[1,10]': {'Value': 1.0}, 'x[2,1]': {'Value': 1.0}, 'x[2,2]': {'Value': 0.0}, 'x[2,3]': {'Value': 1.0},
'x[2,4]': {'Value': 0.0}, 'x[2,5]': {'Value': 0.0}, 'x[2,6]': {'Value': 1.0}, 'x[2,7]': {'Value': 1.0}, 'x[2,8]': {'Value': 0.
0}, 'x[2,9]': {'Value': 1.0}, 'x[2,10]': {'Value': 0.0}, 'x[3,1]': {'Value': 0.0}, 'x[3,2]': {'Value': 0.0}, 'x[3,3]': {'Valu
e': 0.0}, 'x[3,4]': {'Value': 0.0}, 'x[3,5]': {'Value': 1.0}, 'x[3,6]': {'Value': 0.0}, 'x[3,7]': {'Value': 0.0}, 'x[3,8]': {'V
alue': 0.0}, 'x[3,9]': {'Value': 0.0}, 'x[3,10]': {'Value': 0.0}}, 'Constraint': {}}]}
```