4.2 Job Shop Scheduling

In [4]:

```
!pip install -q pyomo
!apt-get install -y -qq glpk-utils
!apt-get install -y -qq coinor-cbc
```

```
/bin/sh: apt-get: command not found
/bin/sh: apt-get: command not found
```

In [ ]:

```
from pyomo.environ import *
from pyomo.gdp import *

#solver = SolverFactory('glpk')
solver = SolverFactory('cbc', executable='/usr/bin/cbc')
#solver = SolverFactory('gurobi', executable='/usr/local/bin/gurobi.sh')
```

Background

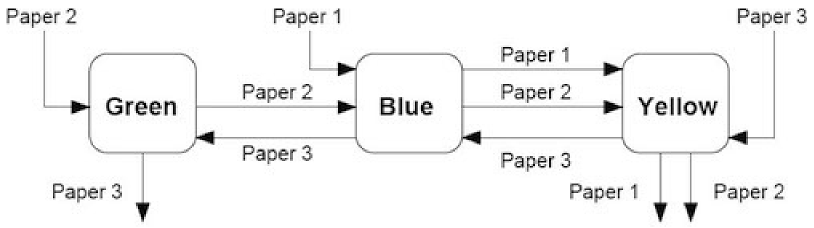
A job shop consists of a set of distinct machines that process jobs. Each job is a series of tasks that require use of particular machines for known durations, and which must be completed in specified order. The job shop scheduling problem is to schedule the jobs on the machines to minimize the time necessary to process all jobs (i.e, the makespan) or some other metric of productivity. Job shop scheduling is one of the classic problems in Operations Research.

Data consists of two tables. The first table is decomposition of the jobs into a series of tasks. Each task lists a job name, name of the required machine, and task duration. The second table list task pairs where the first task must be completed before the second task can be started. This formulation is quite general, but can also specify situations with no feasible solutions.

Job Shop Example

The following example of a job shop is from from Christelle Gueret, Christian Prins, Marc Sevaux, "Applications of Optimization with Xpress-MP," Dash Optimization, 2000.

In this example, there are three printed paper products that must pass through color printing presses in a particular order. The given data consists of a flowsheet showing the order in which each job passes through the color presses



and a table of data showing, in minutes, the amount of time each job requires on each machine.

Machine	Color	Paper 1	Paper 2	Paper 3
1	Blue	45	20	12
2	Green	-	10	17
3	Yellow	10	34	28

What is the minimum amount of time (i.e, what is the makespan) for this set of jobs?

Task Decomposition

The first step in the analysis is to decompose the process into a series of tasks. Each task is a (job,machine) pair. Some tasks cannot start until a prerequisite task is completed.

Task (Job,Machine)	Duration	Prerequisite Task
(Paper 1, Blue)	45	-
(Paper 1, Yellow)	10	(Paper 1,Blue)
(Paper 2, Blue)	20	(Paper 2, Green)
(Paper 2, Green)	10	-
(Paper 2, Yellow)	34	(Paper 2, Blue)
(Paper 3, Blue)	12	(Paper 3, Yellow)
(Paper 3, Green)	17	(Paper 3, Blue)
(Paper 3, Yellow)	28	-

We convert this to a JSON style representation where tasks are denoted by (Job,Machine) tuples in Python. The task data is stored in a Python dictionary indexed by (Job,Machine) tuples. The task data conists of a dictionary with duration ('dur') and (Job,Machine) pair for any prerequisite task.

In [1]:

```
TASKS = {
    ('Paper_1','Blue')   : {'dur': 45, 'prec': None},
    ('Paper_1','Yellow') : {'dur': 10, 'prec': ('Paper_1','Blue')},
    ('Paper_2','Blue')   : {'dur': 20, 'prec': ('Paper_2','Green')},
    ('Paper_2','Green')  : {'dur': 10, 'prec': None},
    ('Paper_2','Yellow') : {'dur': 34, 'prec': ('Paper_2','Blue')},
    ('Paper_3','Blue')   : {'dur': 12, 'prec': ('Paper_3','Yellow')},
    ('Paper_3','Green')  : {'dur': 17, 'prec': ('Paper_3','Blue')},
    ('Paper_3','Yellow') : {'dur': 28, 'prec': None},
}
```

Model Formulation

Each task is represented as an ordered pair (j, m) where j is a job, and m is a machine.

Parameter	Description
$\text{dur}_{j,m}$	Duration of task (j, m)
$\text{prec}_{j,m}$	A task $(k, n) = \text{Prec}_{j,m}$ that must be completed before task (j, m)

Decision Variables	Description
makespan	Completion of all jobs
$\text{start}_{j,m}$	Start time for task (j, m)
$y_{j,k,m}$	boolean variable for tasks (i, m) and (j, m) on machine m where $j < k$

Upper and lower bounds on the start and completion of task (j, m)

$$\begin{aligned} \text{start}_{j,m} & \geq 0 \\ \text{start}_{j,m} + \text{Dur}_{j,m} & \leq \text{makespan} \end{aligned}$$

Satisfying prerequisite tasks

$$\text{start}_{k,n} + \text{Dur}_{k,n} \leq \text{start}_{j,m} \quad \text{for } (k, n) = \text{Prec}_{j,m}$$

Disjunctive Constraints

If M is big enough, then satisfying

$$\begin{aligned} \text{start}_{j,m} + \text{Dur}_{j,m} & \leq \text{start}_{k,m} + M(1 - y_{j,k,m}) \\ \text{start}_{k,m} + \text{Dur}_{k,m} & \leq \text{start}_{j,m} + M y_{j,k,m} \end{aligned}$$

avoids conflicts for use of the same machine.

Pyomo Implementation

The job shop scheduling problem is implemented below in Pyomo. The implementation consists of of a function JobShop(TASKS) that accepts a dictionary of tanks and returns a pandas dataframe containing an optimal schedule of tasks. An optional argument to JobShop allows one to specify a solver.

In [16]:

```
def JobShop(TASKS):

    model = ConcreteModel()

    model.TASKS     = Set(initialize=TASKS.keys(), dimen=2)
    model.JOBS      = Set(initialize=set([j for (j,m) in TASKS.keys()]))
    model.MACHINES = Set(initialize=set([m for (j,m) in TASKS.keys()]))
    model.TASKORDER = Set(initialize = model.TASKS * model.TASKS, dimen=4,
        filter = lambda model,j,m,k,n: (k,n) == TASKS[(j,m)]['prec'])
    model.DISJUNCTIONS = Set(initialize=model.JOBS * model.JOBS * model.MACHINES, dimen
=3,
        filter = lambda model,j,k,m: j < k and (j,m) in model.TASKS and (k,m) in model.T
ASKS)

    t_max = sum([TASKS[(j,m)]['dur'] for (j,m) in TASKS.keys()])

    model.makespan = Var(bounds=(0, t_max))
    model.start = Var(model.TASKS, bounds=(0, t_max))

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

    model.fini = Constraint(model.TASKS, rule=lambda model,j,m:
        model.start[j,m] + TASKS[(j,m)]['dur'] <= model.makespan)

    model.prec = Constraint(model.TASKORDER, rule=lambda model,j,m,k,n:
        model.start[k,n] + TASKS[(k,n)]['dur'] <= model.start[j,m])

    model.disj = Disjunction(model.DISJUNCTIONS, rule=lambda model,j,k,m:
        [model.start[j,m] + TASKS[(j,m)]['dur'] <= model.start[k,m],
         model.start[k,m] + TASKS[(k,m)]['dur'] <= model.start[j,m]])

    TransformationFactory('gdp.chull').apply_to(model)
    solver.solve(model)

    results = [{'Job': j,
                'Machine': m,
                'Start': model.start[j, m](),
                'Duration': TASKS[(j, m)]['dur'],
                'Finish': model.start[j, m]() + TASKS[(j, m)]['dur']}
               for j,m in model.TASKS]
    return results

results = JobShop(TASKS)
results
```

Out[16]:

```
[{'Duration': 1.5,
  'Finish': 25.5,
  'Job': 'C1',
  'Machine': 'Packaging',
  'Start': 24.0},
 {'Duration': 1.5,
  'Finish': 24.0,
  'Job': 'A2',
  'Machine': 'Packaging',
  'Start': 22.5},
 {'Duration': 4,
  'Finish': 18.5,
  'Job': 'A1',
  'Machine': 'Separator',
  'Start': 14.5},
 {'Duration': 5,
  'Finish': 5.0,
  'Job': 'C1',
  'Machine': 'Separator',
  'Start': 0.0},
 {'Duration': 1,
  'Finish': 28.0,
  'Job': 'B2',
  'Machine': 'Packaging',
  'Start': 27.0},
 {'Duration': 1.5,
  'Finish': 19.0,
  'Job': 'C2',
  'Machine': 'Packaging',
  'Start': 17.5},
 {'Duration': 3,
  'Finish': 17.5,
  'Job': 'C2',
  'Machine': 'Reactor',
  'Start': 14.5},
 {'Duration': 1, 'Finish': 1.0, 'Job': 'A1', 'Machine': 'Mixer', 'Start': 0.0},
 {'Duration': 1, 'Finish': 2.0, 'Job': 'A2', 'Machine': 'Mixer', 'Start': 1.0},
 {'Duration': 5,
  'Finish': 11.0,
  'Job': 'A2',
  'Machine': 'Reactor',
  'Start': 6.0},
 {'Duration': 1,
  'Finish': 21.5,
  'Job': 'B1',
  'Machine': 'Packaging',
  'Start': 20.5},
 {'Duration': 3,
  'Finish': 20.5,
  'Job': 'C1',
  'Machine': 'Reactor',
  'Start': 17.5},
 {'Duration': 5,
  'Finish': 14.5,
  'Job': 'C2',
  'Machine': 'Separator',
  'Start': 9.5},
 {'Duration': 4.5,
  'Finish': 27.0,
  'Job': 'B2',
  'Machine': 'Separator',
  'Start': 22.5},
 {'Duration': 1.5,
  'Finish': 20.5,
  'Job': 'A1',
  'Machine': 'Packaging',
  'Start': 19.0},
 {'Duration': 5,
  'Finish': 6.0,
  'Job': 'A1',
  'Machine': 'Reactor',
  'Start': 1.0},
 {'Duration': 4,
  'Finish': 22.5,
  'Job': 'A2',
  'Machine': 'Separator',
  'Start': 18.5},
 {'Duration': 4.5,
  'Finish': 9.5,
  'Job': 'B1',
  'Machine': 'Separator',
  'Start': 5.0}]
```

Printing Schedules

In [17]:

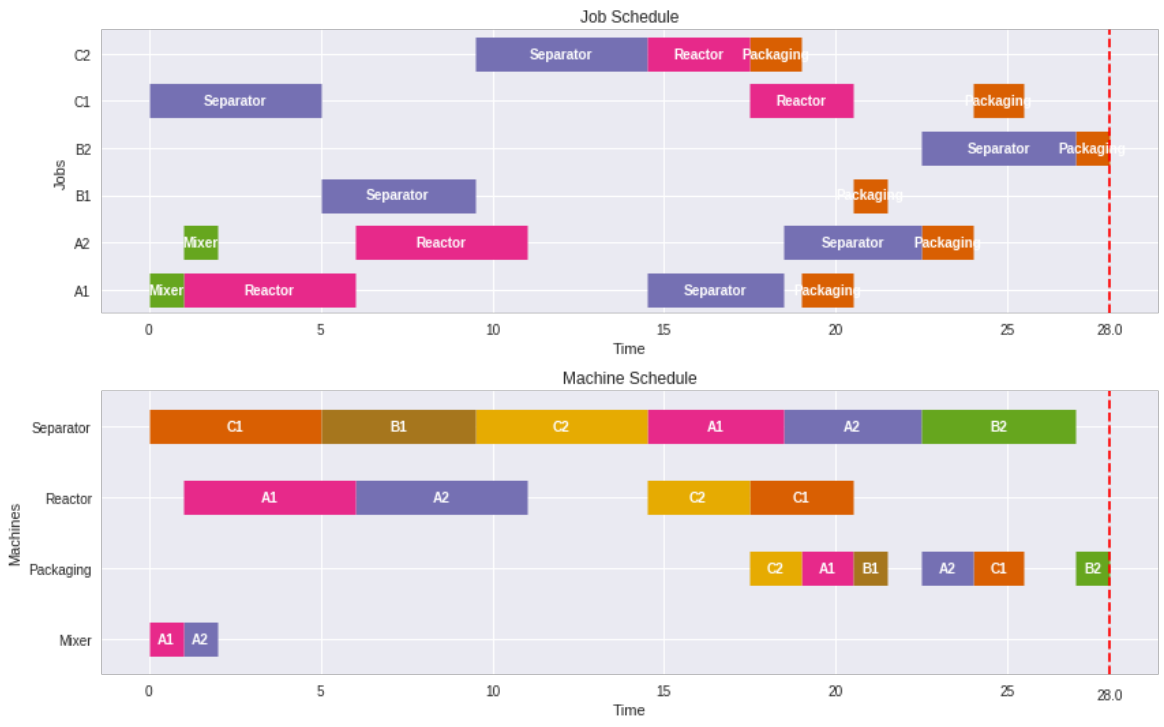
```
import pandas as pd

schedule = pd.DataFrame(results)

print('\nSchedule by Job')
print(schedule.sort_values(by=['Job','Start']).set_index(['Job', 'Machine']))

print('\nSchedule by Machine')
print(schedule.sort_values(by=['Machine','Start']).set_index(['Machine', 'Job']))
```

```
Schedule by Job
               Duration  Finish  Start
Job Machine
A1  Mixer           1.0     1.0    0.0
    Reactor         5.0     6.0    1.0
    Separator       4.0    18.5   14.5
    Packaging       1.5    20.5   19.0
A2  Mixer           1.0     2.0    1.0
    Reactor         5.0    11.0    6.0
    Separator       4.0    22.5   18.5
    Packaging       1.5    24.0   22.5
B1  Separator       4.5     9.5    5.0
    Packaging       1.0    21.5   20.5
B2  Separator       4.5    27.0   22.5
    Packaging       1.0    28.0   27.0
C1  Separator       5.0     5.0    0.0
    Reactor         3.0    20.5   17.5
    Packaging       1.5    25.5   24.0
C2  Separator       5.0    14.5    9.5
    Reactor         3.0    17.5   14.5
    Packaging       1.5    19.0   17.5

Schedule by Machine
               Duration  Finish  Start
Machine   Job
Mixer     A1        1.0     1.0    0.0
          A2        1.0     2.0    1.0
Packaging C2        1.5    19.0   17.5
          A1        1.5    20.5   19.0
          B1        1.0    21.5   20.5
          A2        1.5    24.0   22.5
          C1        1.5    25.5   24.0
          B2        1.0    28.0   27.0
Reactor   A1        5.0     6.0    1.0
          A2        5.0    11.0    6.0
          C2        3.0    17.5   14.5
          C1        3.0    20.5   17.5
Separator C1        5.0     5.0    0.0
          B1        4.5     9.5    5.0
          C2        5.0    14.5    9.5
          A1        4.0    18.5   14.5
          A2        4.0    22.5   18.5
          B2        4.5    27.0   22.5
```

Visualizing Results with Gantt Charts

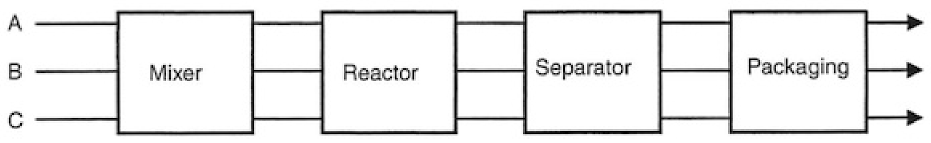
In [18]:

```
%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib as mpl
import pandas as pd

def Visualize(results):
    schedule = pd.DataFrame(results)
    JOBS = list(schedule['Job'].unique())
    MACHINES = list(schedule['Machine'].unique())
    makespan = schedule['Finish'].max()

    schedule.sort_values(by=['Job','Start'])
    schedule.set_index(['Job', 'Machine'], inplace=True)

    plt.figure(figsize=(12, 5 + (len(JOBS)+len(MACHINES))/4))
    plt.subplot(2,1,1)

    jdx = 0
    for j in sorted(JOBS):
        jdx += 1
        mdx = 0
        for m in MACHINES:
            mdx += 1
            c = mpl.cm.Dark2.colors[mdx%7]
            if (j,m) in schedule.index:
                plt.plot([schedule.loc[(j,m),'Start'],schedule.loc[(j,m),'Finish']],
                     [jdx,jdx],color = c,alpha=1.0,lw=25,solid_capstyle='butt')
                plt.text((schedule.loc[(j,m),'Start'] +
schedule.loc[(j,m),'Finish'])/2.0,jdx,
                    m, color='white', weight='bold',
                    horizontalalignment='center', verticalalignment='center')

    plt.ylim(0.5,jdx+0.5)
    plt.title('Job Schedule')
    plt.gca().set_yticks(range(1,1+len(JOBS)))
    plt.gca().set_yticklabels(sorted(JOBS))
    plt.plot([makespan,makespan],plt.ylim(),'r--')
    plt.text(makespan,plt.ylim()[0]-0.2,str(round(makespan,2)),
            horizontalalignment='center', verticalalignment='top')
    plt.xlabel('Time')
    plt.ylabel('Jobs')

    plt.subplot(2,1,2)
    mdx = 0
    for m in sorted(MACHINES):
        mdx += 1
        jdx = 0
        for j in JOBS:
            jdx += 1
            c = mpl.cm.Dark2.colors[jdx%7]
            if (j,m) in schedule.index:
                plt.plot([schedule.loc[(j,m),'Start'],schedule.loc[(j,m),'Finish']],
                     [mdx,mdx],color = c,alpha=1.0,lw=25,solid_capstyle='butt')
                plt.text((schedule.loc[(j,m),'Start'] +
schedule.loc[(j,m),'Finish'])/2.0,mdx,
                    j, color='white', weight='bold',
                    horizontalalignment='center', verticalalignment='center')

    plt.ylim(0.5,mdx+0.5)
    plt.title('Machine Schedule')
    plt.gca().set_yticks(range(1,1+len(MACHINES)))
    plt.gca().set_yticklabels(sorted(MACHINES))
    plt.plot([makespan,makespan],plt.ylim(),'r--')
    plt.text(makespan,plt.ylim()[0]-0.2,str(round(makespan,2)),
            horizontalalignment='center', verticalalignment='top')
    plt.xlabel('Time')
    plt.ylabel('Machines')

    plt.tight_layout()

Visualize(results)
```



Application to Scheduling of Batch Processes

We will now turn our attention to the application of the job shop scheduling problem to the short term scheduling of batch processes. We illustrate these techniques using Example II from Dunn (2013).



Process	Mixer	Reactor	Separator	Packaging
A	1.0	5.0	4.0	1.5
B	-	-	4.5	1.0
C	-	3.0	5.0	1.5

Single Product Strategies

Before going further, we create a function to streamline the generation of the TASKS dictionary.

In [2]:

```
def Recipe(jobs,machines,durations):
    TASKS = {}
    for j in jobs:
        prec = (None,None)
        for m,d in zip(machines,durations):
            task = (j,m)
            if prec == (None,None):
                TASKS.update({(j,m): {'dur': d, 'prec': None}})
            else:
                TASKS.update({(j,m): {'dur': d, 'prec': prec}})
            prec = task
    return TASKS

RecipeA = Recipe('A',['Mixer','Reactor','Separator','Packaging'],[1,5,4,1.5])
RecipeB = Recipe('B',['Separator','Packaging'],[4.5,1])
RecipeC = Recipe('C',['Separator','Reactor','Packaging'],[5,3,1.5])

Visualize(JobShop(RecipeA))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-184d66ea5167> in <module>
     16 RecipeC = Recipe('C',['Separator','Reactor','Packaging'],[5,3,1.5])
     17
---> 18 Visualize(JobShop(RecipeA))

NameError: name 'Visualize' is not defined
```

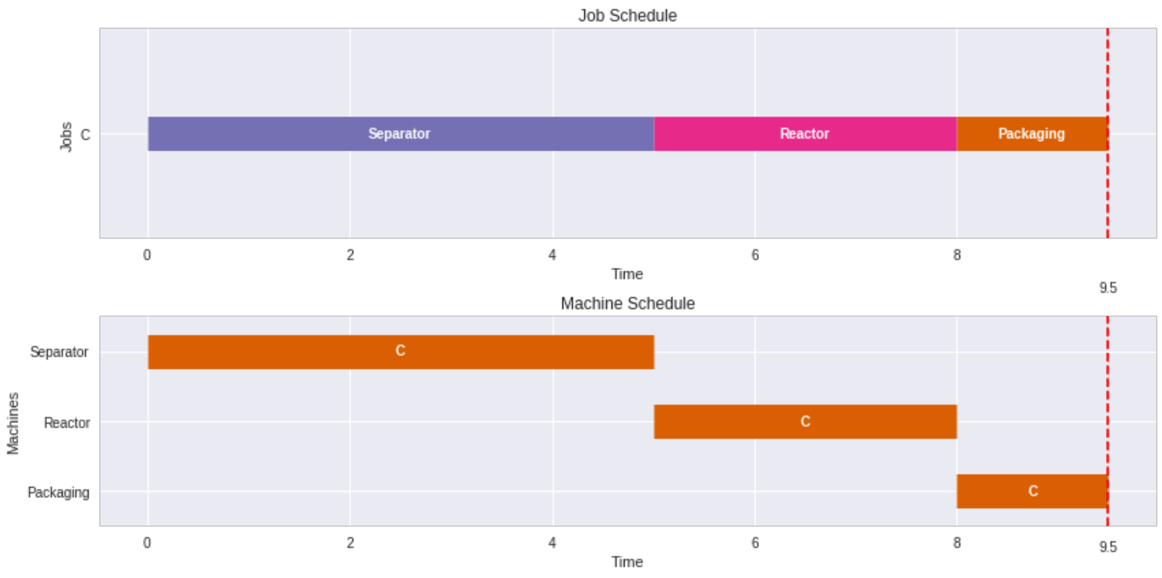
In [3]:

```
Visualize(JobShop(RecipeB))
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-0b43ff55198c> in <module>
----> 1 Visualize(JobShop(RecipeB))

NameError: name 'Visualize' is not defined
```

In [21]:

```
Visualize(JobShop(RecipeC))
```



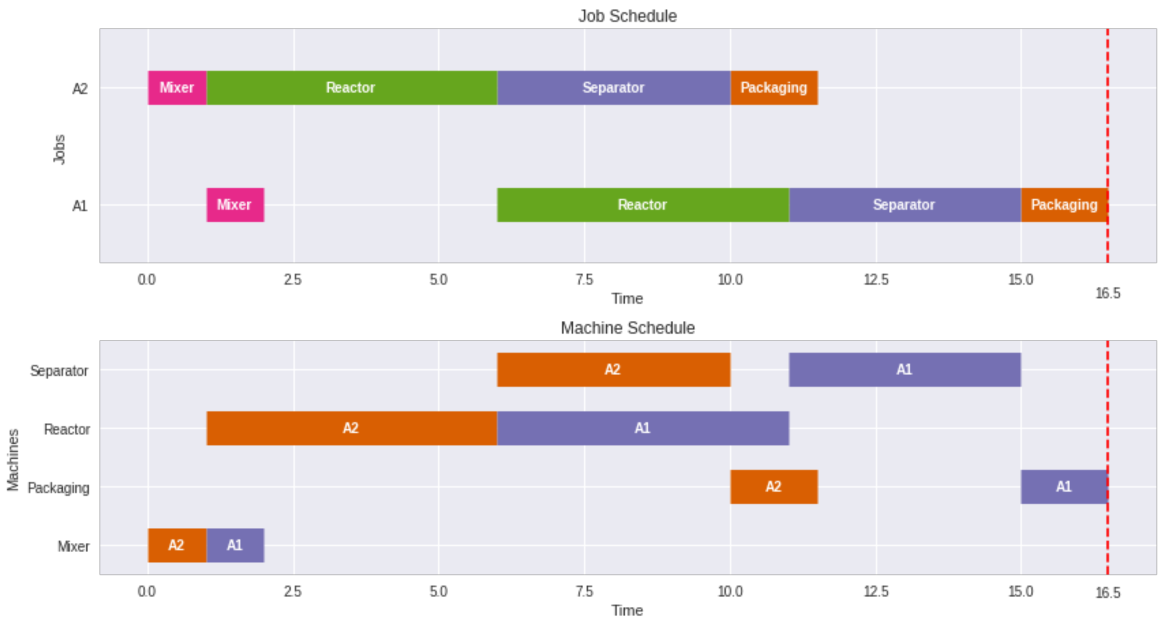
Overlapping Tasks

Let's now consider an optimal scheduling problem where we are wish to make two batches of Product A.

In [22]:

```
TASKS = Recipe(['A1','A2'],['Mixer','Reactor','Separator','Packaging'],[1,5,4,1.5])
results = JobShop(TASKS)
Visualize(results)
print("Makespan =", max([task['Finish'] for task in results]))
```

```
Makespan = 16.5
```



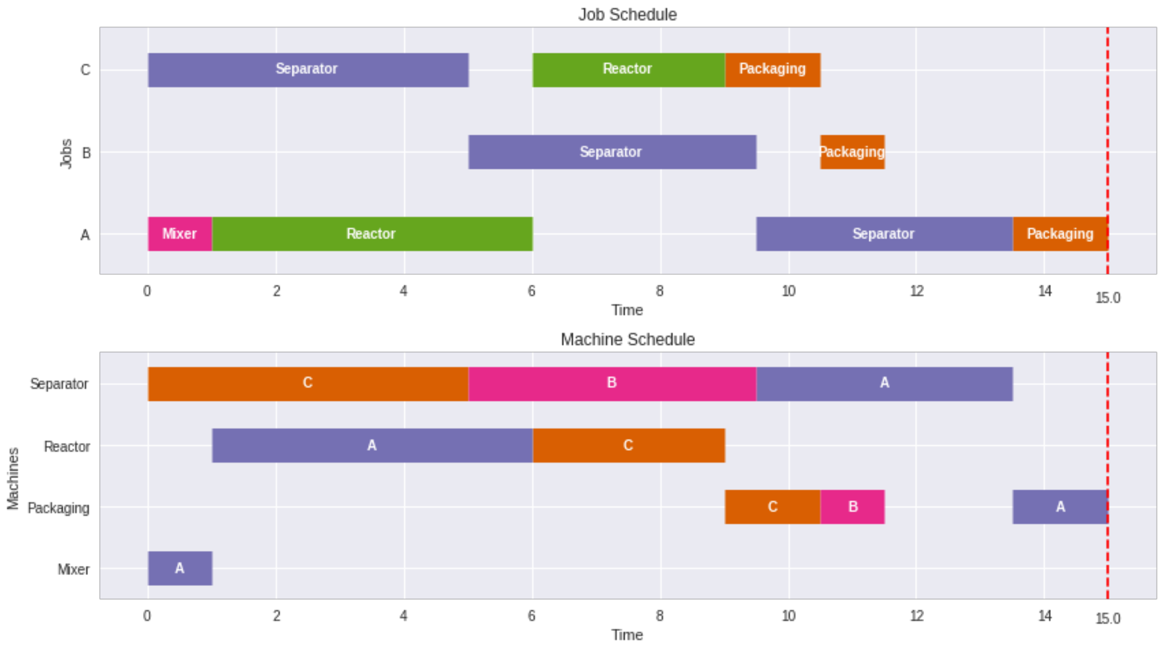
Earlier we found it tood 11.5 hours to produce one batch of product A. As we see here, we can produce a second batch with only 5.0 additional hours because some of the tasks overlap. The overlapping of tasks is the key to gaining efficiency in batch processing facilities.

Let's next consider production of a single batch each of products A, B, and C.

In [23]:

```
TASKS = RecipeA
TASKS.update(RecipeB)
TASKS.update(RecipeC)

results = JobShop(TASKS)
Visualize(results)
print("Makespan =", max([task['Finish'] for task in results]))
```

```
Makespan = 15.0
```



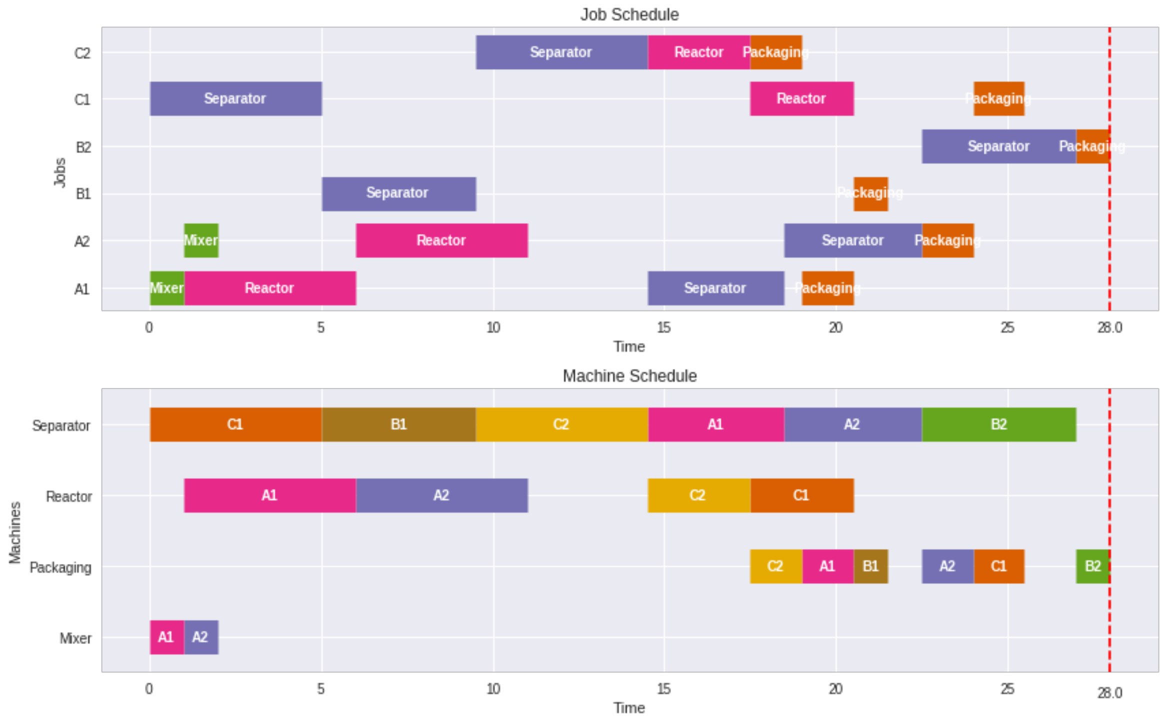
The individual production of A, B, and C required 11.5, 5.5, and 9.5 hours, respectively, for a total of 25.5 hours. As we see here, by scheduling the production simultaneously, we can get all three batches done in just 15 hours.

As we see below, each additional set of three products takes an additionl 13 hours. So there is considerable efficiency gained by scheduling over longer intervals whenever possible.

In [24]:

```
TASKS = Recipe(['A1','A2'],['Mixer','Reactor','Separator','Packaging'],[1,5,4,1.5])
TASKS.update(Recipe(['B1','B2'],['Separator','Packaging'],[4.5,1]))
TASKS.update(Recipe(['C1','C2'],['Separator','Reactor','Packaging'],[5,3,1.5]))

results = JobShop(TASKS)
Visualize(results)
print("Makespan =", max([task['Finish'] for task in results]))
```

```
Makespan = 28.0
```



Unit Cleanout

A common feature in batch unit operations is a requirement that equipment be cleaned prior to reuse.

In most cases the time needed for clean out would be equipment and product specific. Bur for the purposes of illustration, we implement this policy with a single non-negative parameter $t_{clean} \geq 0$ which, if specified, requires a period no less than t_{clean} between the finish of one task and the start of another on every piece of equipment.

This is implemented by modifying the usual disjunctive constraints to avoid machine conflicts, i.e.,

$$\begin{aligned} \text{start}_{j,m} + \text{Dur}_{j,m} & \leq \text{start}_{k,m} + M(1 - y_{j,k,m}) \\ \text{start}_{k,m} + \text{Dur}_{k,m} & \leq \text{start}_{j,m} + M y_{j,k,m} \end{aligned}$$

to read

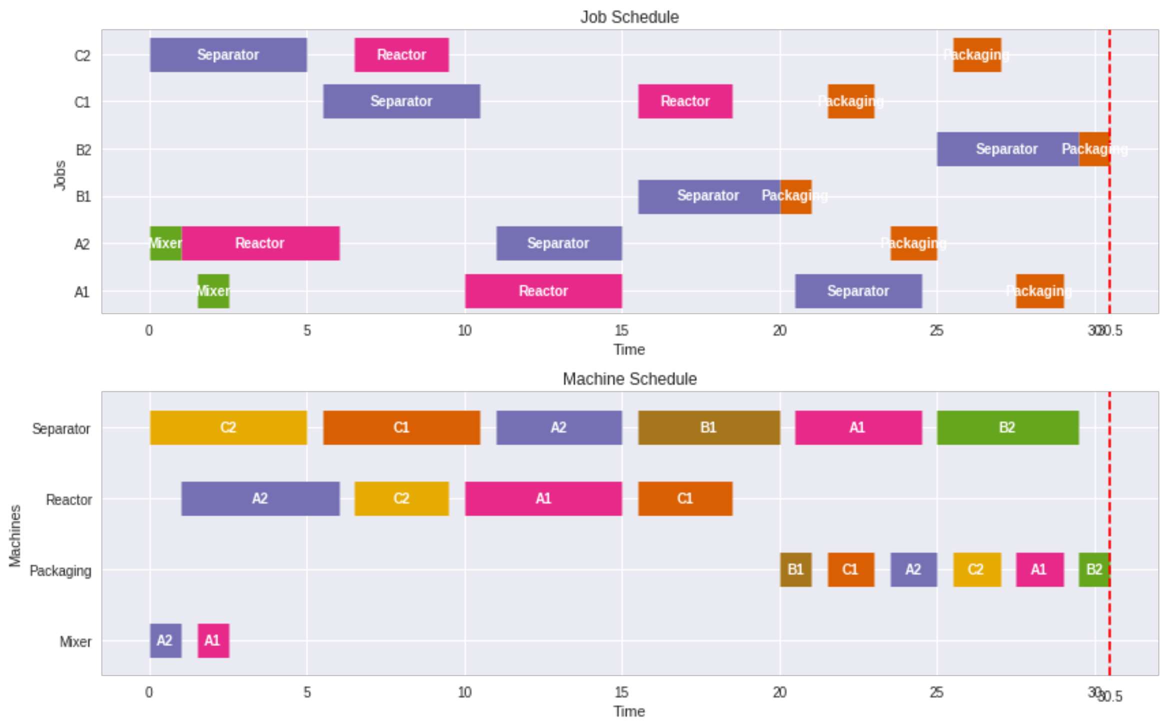
$$\begin{aligned} \text{start}_{j,m} + \text{Dur}_{j,m} + t_{clean} & \leq \text{start}_{k,m} + M(1 - y_{j,k,m}) \\ \text{start}_{k,m} + \text{Dur}_{k,m} + t_{clean} & \leq \text{start}_{j,m} + M y_{j,k,m} \end{aligned}$$

for sufficiently large M.

```
In [27]:
```

```
def JobShop(TASKS, tclean=0):

    model = ConcreteModel()

    model.TASKS     = Set(initialize=TASKS.keys(), dimen=2)
    model.JOBS      = Set(initialize=set([j for (j,m) in TASKS.keys()]))
    model.MACHINES = Set(initialize=set([m for (j,m) in TASKS.keys()]))
    model.TASKORDER = Set(initialize = model.TASKS * model.TASKS, dimen=4,
        filter = lambda model,j,m,k,n: (k,n) == TASKS[(j,m)]['prec'])
    model.DISJUNCTIONS = Set(initialize=model.JOBS * model.JOBS * model.MACHINES, dimen
=3,
        filter = lambda model,j,k,m: j < k and (j,m) in model.TASKS and (k,m) in model.T
ASKS)

    t_max = sum([TASKS[(j,m)]['dur'] for (j,m) in TASKS.keys()])

    model.makespan = Var(bounds=(0, t_max))
    model.start = Var(model.TASKS, bounds=(0, t_max))

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

    model.fini = Constraint(model.TASKS, rule=lambda model,j,m:
        model.start[j,m] + TASKS[(j,m)]['dur'] <= model.makespan)

    model.prec = Constraint(model.TASKORDER, rule=lambda model,j,m,k,n:
        model.start[k,n] + TASKS[(k,n)]['dur'] <= model.start[j,m])

    model.disj = Disjunction(model.DISJUNCTIONS, rule=lambda model,j,k,m:
        [model.start[j,m] + TASKS[(j,m)]['dur'] + tclean <= model.start[k,m],
         model.start[k,m] + TASKS[(k,m)]['dur'] + tclean <= model.start[j,m]])

    TransformationFactory('gdp.chull').apply_to(model)
    solver.solve(model)

    results = [{'Job': j,
                'Machine': m,
                'Start': model.start[j, m](),
                'Duration': TASKS[(j, m)]['dur'],
                'Finish': model.start[j, m]() + TASKS[(j, m)]['dur']}
               for j,m in model.TASKS]
    return results

results = JobShop(TASKS, tclean=0.5)
Visualize(results)
print("Makespan =", max([task['Finish'] for task in results]))
```

```
Makespan = 30.5
```



Zero Wait Policy

One of the issues in the use of job shop scheduling for batch processing are situations where there it isn't possible to store intermediate materials. If there is no way to store intermediates, either in the processing equipment or in external vessels, then a **zero-wait** policy may be appropriate.

A zero-wait policy requires subsequent processing machines to be available immediately upon completion of any task. To implement this policy, the usual precident sequencing constraint of a job shop scheduling problem, i.e.,

$$\text{start}_{k,n} + \text{Dur}_{k,n} \leq \text{start}_{j,m} \quad \text{for } (k,n) = \text{Prec}_{j,m}$$

is changed to

$$\text{start}_{k,n} + \text{Dur}_{k,n} = \text{start}_{j,m} \quad \text{for } (k,n) = \text{Prec}_{j,m} \text{ and ZW is True}$$

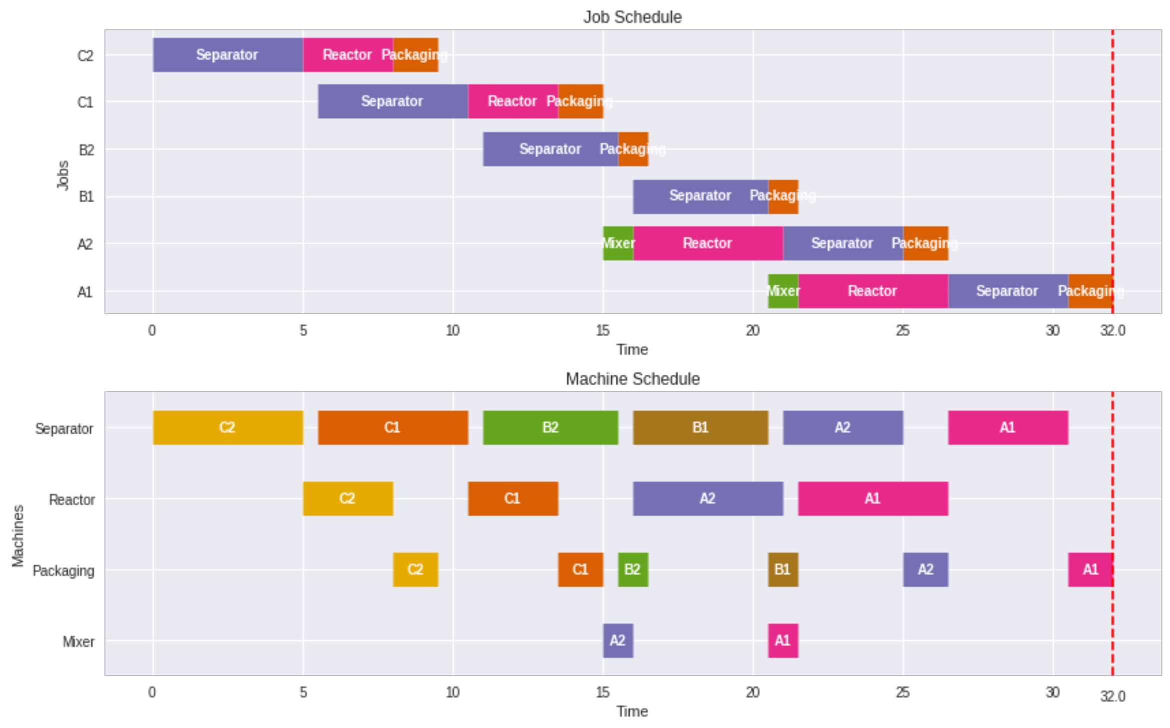
if the zero-wait policy is in effect.

While this could be implemented on an equipment or product specific basis, here we add an optional ZW flag to the JobShop function that, by default, is set to False.

In [29]:

```
def JobShop(TASKS, tclean=0, ZW=False):

    model = ConcreteModel()

    model.TASKS     = Set(initialize=TASKS.keys(), dimen=2)
    model.JOBS      = Set(initialize=set([j for (j,m) in TASKS.keys()]))
    model.MACHINES = Set(initialize=set([m for (j,m) in TASKS.keys()]))
    model.TASKORDER = Set(initialize = model.TASKS * model.TASKS, dimen=4,
        filter = lambda model,j,m,k,n: (k,n) == TASKS[(j,m)]['prec'])
    model.DISJUNCTIONS = Set(initialize=model.JOBS * model.JOBS * model.MACHINES, dimen
=3,
        filter = lambda model,j,k,m: j < k and (j,m) in model.TASKS and (k,m) in model.T
ASKS)

    t_max = sum([TASKS[(j,m)]['dur'] for (j,m) in TASKS.keys()])

    model.makespan = Var(bounds=(0, t_max))
    model.start = Var(model.TASKS, bounds=(0, t_max))

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

    model.fini = Constraint(model.TASKS, rule=lambda model,j,m:
        model.start[j,m] + TASKS[(j,m)]['dur'] <= model.makespan)

    if ZW:
        model.prec = Constraint(model.TASKORDER, rule=lambda model,j,m,k,n:
            model.start[k,n] + TASKS[(k,n)]['dur'] == model.start[j,m])
    else:
        model.prec = Constraint(model.TASKORDER, rule=lambda model,j,m,k,n:
            model.start[k,n] + TASKS[(k,n)]['dur'] <= model.start[j,m])

    model.disj = Disjunction(model.DISJUNCTIONS, rule=lambda model,j,k,m:
        [model.start[j,m] + TASKS[(j,m)]['dur'] + tclean <= model.start[k,m],
         model.start[k,m] + TASKS[(k,m)]['dur'] + tclean <= model.start[j,m]])

    TransformationFactory('gdp.chull').apply_to(model)
    solver.solve(model)

    results = [{'Job': j,
                'Machine': m,
                'Start': model.start[j, m](),
                'Duration': TASKS[(j, m)]['dur'],
                'Finish': model.start[j, m]() + TASKS[(j, m)]['dur']}
               for j,m in model.TASKS]
    return results

results = JobShop(TASKS, tclean=0.5, ZW=True)
Visualize(results)
print("Makespan =", max([task['Finish'] for task in results]))
```

```
Makespan = 32.0
```



Benchmark Problems LA19

The file `jobshop1.txt` is a well known collection of 82 benchmark problems is a well-known collection of job shop scheduling problems in the OR-Library maintained by J. E. Beasley. The data format for each example consists of a single line for each job. The data on each line is a sequence of (machine number, time) pairs showing the order in which machines process each job.

LA19 is a benchmark problem for job shop scheduling introduced by Lawrence in 1984, and a solution presented by Cook and Applegate in 1991. The following cell may take many minutes to hours to run, depending on the choice of solver and hardware.

In [30]:

```
data = """
2  44  3   5  5  58  4  97  0   9  7  84  8  77  9  96  1  58  6  89
4  15  7  31  1  87  8  57  0  77  3  85  2  81  5  39  9  73  6  21
9  82  6  22  4  10  3  70  1  49  0  40  8  34  2  48  7  80  5  71
1  91  2  17  7  62  5  75  8  47  4  11  3   7  6  72  9  35  0  55
6  71  1  90  3  75  0  64  2  94  8  15  4  12  7  67  9  20  5  50
7  70  5  93  8  77  2  29  4  58  6  93  3  68  1  57  9   7  0  52
6  87  1  63  4  26  5   6  2  82  3  27  7  56  8  48  9  36  0  95
0  36  5  15  8  41  9  78  3  76  6  84  4  30  7  76  2  36  1   8
5  88  2  81  3  13  6  82  4  54  7  13  8  29  9  40  1  78  0  75
9  88  4  54  6  64  7  32  0  52  2   6  8  54  5  82  3   6  1  26
"""

TASKS = {}
prec = ''

lines = data.splitlines()
job= 0
for line in lines[1:]:
    j = "J{0:1d}".format(job)
    nums = line.split()
    prec = ''
    for m,dur in zip(nums[::2],nums[1::2]):
        task = (j,'M{0:s}'.format(m))
        if prec:
            TASKS[task] = {'dur':int(dur), 'prec':prec}
        else:
            TASKS[task] = {'dur':int(dur), 'prec':None}
        prec = task
    job += 1

Visualize(JobShop(TASKS))
```

```
  Signal handler called from  /usr/lib/python3.6/subprocess.py _try_wait 1404
  Waiting...
  Signaled process 319 with signal 2
ERROR: Solver (cbc) returned non-zero return code (-1)
ERROR: Solver log: Welcome to the CBC MILP Solver Version: 2.9.9 Build Date:
    Aug 21 2017

    command line - /usr/bin/cbc -printingOptions all -import
    /content/tmp_4xbvphs.pyomo.lp -stat=1 -solve -solu
    /content/tmp_4xbvphs.pyomo.soln (default strategy 1) Option for
    printingOptions changed from normal to all Presolve 2890 (-1351) rows,
    1451 (-1351) columns and 8480 (-1801) elements Statistics for presolved
    model Original problem has 900 integers (900 of which binary) Presolved
    problem has 450 integers (450 of which binary) ==== 1450 zero objective 2
    different 1450 variables have objective of 0 1 variables have objective of
    1 ==== absolute objective values 2 different 1450 variables have objective
    of 0 1 variables have objective of 1 ==== for integers 450 zero objective
    1 different 450 variables have objective of 0 ==== for integers absolute
    objective values 1 different 450 variables have objective of 0 ===== end
    objective counts


    Problem has 2890 rows, 1451 columns (1 with objective) and 8480 elements
    Column breakdown: 0 of type 0.0->inf, 1001 of type 0.0->up, 0 of type
    lo->inf, 0 of type lo->up, 0 of type free, 0 of type fixed, 0 of type
    -inf->0.0, 0 of type -inf->up, 450 of type 0.0->1.0 Row breakdown: 0 of
    type E 0.0, 0 of type E 1.0, 0 of type E -1.0, 0 of type E other, 0 of
    type G 0.0, 0 of type G 1.0, 0 of type G other, 1350 of type L 0.0, 0 of
    type L 1.0, 1540 of type L other, 0 of type Range 0.0->1.0, 0 of type
    Range other, 0 of type Free Continuous objective value is 617 - 0.02
    seconds Cgl0003I 0 fixed, 0 tightened bounds, 900 strengthened rows, 0
    substitutions Cgl0003I 0 fixed, 0 tightened bounds, 48 strengthened rows,
    0 substitutions Cgl0004I processed model has 2890 rows, 1451 columns (450
    integer (450 of which binary)) and 8480 elements Cbc0038I Initial state -
    432 integers unsatisfied sum - 31.8732 Cbc0038I Pass   1: suminf.
    0.08567 (1) obj. 4186 iterations 655 Cbc0038I Pass   2: suminf.    0.00000
    (0) obj. 4186 iterations 2 Cbc0038I Solution found of 4186 Cbc0038I
    Relaxing continuous gives 4186 Cbc0038I Before mini branch and bound, 18
    integers at bound fixed and 478 continuous Cbc0038I Mini branch and bound
    did not improve solution (0.18 seconds) Cbc0038I Round again with cutoff
    of 3829.1 Cbc0038I Pass   3: suminf.    0.14188 (5) obj. 3829.1 iterations
    4 Cbc0038I Pass   4: suminf.    0.00000 (0) obj. 3829.1 iterations 15
    Cbc0038I Solution found of 3829.1 Cbc0038I Relaxing continuous gives 3762
    Cbc0038I Before mini branch and bound, 18 integers at bound fixed and 474
    continuous Cbc0038I Mini branch and bound did not improve solution (0.25
    seconds) Cbc0038I Round again with cutoff of 3133 Cbc0038I Pass   5:
    suminf.    0.27367 (11) obj. 3133 iterations 8 Cbc0038I Pass   6: suminf.
    0.00000 (0) obj. 3133 iterations 32 Cbc0038I Solution found of 3133
    Cbc0038I Relaxing continuous gives 3091 Cbc0038I Before mini branch and
    bound, 18 integers at bound fixed and 468 continuous Cbc0038I Mini branch
    and bound did not improve solution (0.33 seconds) Cbc0038I Round again
    with cutoff of 2348.8 Cbc0038I Pass   7: suminf.    0.62110 (27) obj.
    2348.8 iterations 21 Cbc0038I Pass   8: suminf.    0.02174 (4) obj. 2348.8
    iterations 74 Cbc0038I Pass   9: suminf.    0.00724 (1) obj. 2348.8
    iterations 20 Cbc0038I Pass  10: suminf.    0.00000 (0) obj. 2348.8
    iterations 5 Cbc0038I Solution found of 2348.8 Cbc0038I Relaxing
    continuous gives 2344 Cbc0038I Before mini branch and bound, 18 integers
    at bound fixed and 454 continuous Cbc0038I Mini branch and bound did not
    improve solution (0.42 seconds) Cbc0038I Round again with cutoff of 1825.9
    Cbc0038I Pass  11: suminf.    1.07503 (43) obj. 1825.9 iterations 24
    Cbc0038I Pass  12: suminf.    0.42644 (28) obj. 1825.9 iterations 38
    Cbc0038I Pass  13: suminf.    0.04707 (11) obj. 1825.9 iterations 75
    Cbc0038I Pass  14: suminf.    0.07513 (7) obj. 1825.9 iterations 70
    Cbc0038I Pass  15: suminf.    0.02039 (5) obj. 1825.9 iterations 34
    Cbc0038I Pass  16: suminf.    0.02302 (1) obj. 1825.9 iterations 18
    Cbc0038I Pass  17: suminf.    0.00000 (0) obj. 1825.9 iterations 2
    Cbc0038I Solution found of 1825.9 Cbc0038I Relaxing continuous gives 1809
    Cbc0038I Before mini branch and bound, 18 integers at bound fixed and 437
    continuous Cbc0038I Mini branch and bound did not improve solution (0.53
    seconds) Cbc0038I Freeing continuous variables gives a solution of 1809
    Cbc0038I Round again with cutoff of 1332.2 Cbc0038I Pass  18: suminf.
    1.96379 (71) obj. 1332.2 iterations 52 Cbc0038I Pass  19: suminf.
    0.58600 (47) obj. 1332.2 iterations 59 Cbc0038I Pass  20: suminf.
    0.13103 (23) obj. 1332.2 iterations 104 Cbc0038I Pass  21: suminf.
    0.15270 (16) obj. 1332.2 iterations 78 Cbc0038I Pass  22: suminf.
    0.07938 (13) obj. 1332.2 iterations 65 Cbc0038I Pass  23: suminf.
    0.13261 (12) obj. 1332.2 iterations 66 Cbc0038I Pass  24: suminf.
    0.06156 (9) obj. 1332.2 iterations 42 Cbc0038I Pass  25: suminf.
    0.14238 (11) obj. 1332.2 iterations 38 Cbc0038I Pass  26: suminf.
    5.81772 (145) obj. 1332.2 iterations 323 Cbc0038I Pass  27: suminf.
    3.38257 (103) obj. 1332.2 iterations 83 Cbc0038I Pass  28: suminf.
    1.40182 (57) obj. 1332.2 iterations 103 Cbc0038I Pass  29: suminf.
    0.32342 (30) obj. 1332.2 iterations 73 Cbc0038I Pass  30: suminf.
    0.12258 (12) obj. 1332.2 iterations 126 Cbc0038I Pass  31: suminf.
    0.04616 (7) obj. 1332.2 iterations 66 Cbc0038I Pass  32: suminf.
    0.04705 (5) obj. 1332.2 iterations 35 Cbc0038I Pass  33: suminf.
    0.03656 (5) obj. 1332.2 iterations 22 Cbc0038I Pass  34: suminf.
    0.01994 (3) obj. 1332.2 iterations 24 Cbc0038I Pass  35: suminf.
    0.01056 (4) obj. 1332.2 iterations 21 Cbc0038I Pass  36: suminf.
    5.73919 (154) obj. 1332.2 iterations 361 Cbc0038I Pass  37: suminf.
    3.51088 (113) obj. 1332.2 iterations 85 Cbc0038I Pass  38: suminf.
    2.54396 (97) obj. 1332.2 iterations 51 Cbc0038I Pass  39: suminf.
    1.68684 (66) obj. 1332.2 iterations 71 Cbc0038I Pass  40: suminf.
    0.40749 (40) obj. 1332.2 iterations 99 Cbc0038I Pass  41: suminf.
    0.15058 (23) obj. 1332.2 iterations 62 Cbc0038I Pass  42: suminf.
    0.13852 (9) obj. 1332.2 iterations 117 Cbc0038I Pass  43: suminf.
    0.04514 (7) obj. 1332.2 iterations 37 Cbc0038I Pass  44: suminf.
    0.04947 (5) obj. 1332.2 iterations 31 Cbc0038I Pass  45: suminf.
    0.03443 (5) obj. 1332.2 iterations 15 Cbc0038I Pass  46: suminf.
    4.60897 (136) obj. 1332.2 iterations 352 Cbc0038I Pass  47: suminf.
    2.93035 (111) obj. 1332.2 iterations 54 Cbc0038I No solution found this
    major pass Cbc0038I Before mini branch and bound, 5 integers at bound
```

```
fixed and 183 continuous Cbc0038I Full problem 2890 rows 1451 columns,
reduced to 2697 rows 1263 columns - 15 fixed gives 2627, 1227 - still too
large Cbc0038I Mini branch and bound did not improve solution (0.97
seconds) Cbc0038I After 0.97 seconds - Feasibility pump exiting with
objective of 1809 - took 0.84 seconds Cbc0012I Integer solution of 1809
found by feasibility pump after 0 iterations and 0 nodes (0.97 seconds)
Cbc0031I 739 added rows had average density of 5.0460081 Cbc0013I At root
node, 739 cuts changed objective from 617 to 682.11199 in 21 passes
Cbc0014I Cut generator 0 (Probing) - 16701 row cuts average 5.1 elements,
0 column cuts (190 active)  in 0.184 seconds - new frequency is 1 Cbc0014I
Cut generator 1 (Gomory) - 2657 row cuts average 7.1 elements, 0 column
cuts (0 active)  in 0.149 seconds - new frequency is 1 Cbc0014I Cut
generator 2 (Knapsack) - 0 row cuts average 0.0 elements, 0 column cuts (0
active)  in 0.023 seconds - new frequency is -100 Cbc0014I Cut generator 3
(Clique) - 0 row cuts average 0.0 elements, 0 column cuts (0 active)  in
0.004 seconds - new frequency is -100 Cbc0014I Cut generator 4
(MixedIntegerRounding2) - 2000 row cuts average 2.2 elements, 0 column
cuts (0 active)  in 0.050 seconds - new frequency is 1 Cbc0014I Cut
generator 5 (FlowCover) - 139 row cuts average 2.0 elements, 0 column cuts
(0 active)  in 0.043 seconds - new frequency is -100 Cbc0014I Cut
generator 6 (TwoMirCuts) - 2174 row cuts average 4.1 elements, 0 column
cuts (0 active)  in 0.142 seconds - new frequency is 1 Cbc0010I After 0
nodes, 1 on tree, 1809 best solution, best possible 682.11199 (7.51
seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to 2431
rows 1145 columns - 4 fixed gives 2419, 1137 - still too large Cbc0038I
Full problem 2890 rows 1451 columns, reduced to 2356 rows 1095 columns - 2
fixed gives 2350, 1091 - still too large Cbc0010I After 100 nodes, 59 on
tree, 1809 best solution, best possible 682.11199 (47.98 seconds) Cbc0038I
Full problem 2890 rows 1451 columns, reduced to 2224 rows 1007 columns - 1
fixed gives 2221, 1005 - still too large Cbc0010I After 200 nodes, 112 on
tree, 1809 best solution, best possible 682.11199 (54.88 seconds) Cbc0012I
Integer solution of 1095 found by rounding after 93314 iterations and 294
nodes (57.95 seconds) Cbc0038I Full problem 2890 rows 1451 columns,
reduced to 1554 rows 569 columns - 1 fixed gives 1551, 567 - still too
large Cbc0010I After 300 nodes, 154 on tree, 1095 best solution, best
possible 682.11199 (58.97 seconds) Cbc0016I Integer solution of 1061 found
by strong branching after 95344 iterations and 302 nodes (59.11 seconds)
Cbc0038I Full problem 2890 rows 1451 columns, reduced to 1804 rows 735
columns - 1 fixed gives 1801, 733 - still too large Cbc0010I After 400
nodes, 201 on tree, 1061 best solution, best possible 682.11199 (74.64
seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to 1695
rows 662 columns - 1 fixed gives 1692, 660 - still too large Cbc0010I
After 500 nodes, 252 on tree, 1061 best solution, best possible 682.11199
(80.64 seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to
1671 rows 646 columns - 1 fixed gives 1668, 644 - still too large Cbc0010I
After 600 nodes, 297 on tree, 1061 best solution, best possible 682.11199
(87.16 seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to
1662 rows 640 columns - 1 fixed gives 1659, 638 - still too large Cbc0010I
After 700 nodes, 348 on tree, 1061 best solution, best possible 682.11199
(91.58 seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to
1677 rows 650 columns - 1 fixed gives 1674, 648 - still too large Cbc0010I
After 800 nodes, 391 on tree, 1061 best solution, best possible 682.11199
(96.49 seconds) Cbc0010I After 900 nodes, 443 on tree, 1061 best solution,
best possible 682.11199 (100.81 seconds) Cbc0010I After 1000 nodes, 487 on
tree, 1061 best solution, best possible 682.11199 (104.82 seconds)
Cbc0004I Integer solution of 956 found after 205226 iterations and 1051
nodes (107.19 seconds) Cbc0010I After 1100 nodes, 449 on tree, 956 best
solution, best possible 682.11199 (111.57 seconds) Cbc0038I Full problem
2890 rows 1451 columns, reduced to 1618 rows 602 columns - 1 fixed gives
1615, 600 - still too large Cbc0038I Full problem 2890 rows 1451 columns,
reduced to 1613 rows 599 columns - too large Cbc0010I After 1200 nodes,
493 on tree, 956 best solution, best possible 682.11199 (117.94 seconds)
Cbc0010I After 1300 nodes, 537 on tree, 956 best solution, best possible
682.11199 (124.43 seconds) Cbc0010I After 1400 nodes, 584 on tree, 956
best solution, best possible 682.11199 (131.25 seconds) Cbc0010I After
1500 nodes, 633 on tree, 956 best solution, best possible 682.11199
(137.03 seconds) Cbc0010I After 1600 nodes, 679 on tree, 956 best
solution, best possible 682.11199 (142.13 seconds) Cbc0010I After 1700
```

```
nodes, 728 on tree, 956 best solution, best possible 682.11199 (148.08
seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to 1614
rows 599 columns - 1 fixed gives 1611, 597 - still too large Cbc0038I Full
problem 2890 rows 1451 columns, reduced to 1607 rows 595 columns - too
large Cbc0010I After 1800 nodes, 773 on tree, 956 best solution, best
possible 682.11199 (152.37 seconds) Cbc0010I After 1900 nodes, 816 on
tree, 956 best solution, best possible 682.11199 (156.69 seconds) Cbc0010I
After 2000 nodes, 860 on tree, 956 best solution, best possible 682.11199
(161.81 seconds) Cbc0010I After 2100 nodes, 906 on tree, 956 best
solution, best possible 682.11199 (166.74 seconds) Cbc0010I After 2200
nodes, 950 on tree, 956 best solution, best possible 682.11199 (172.31
seconds) Cbc0004I Integer solution of 897 found after 408219 iterations
and 2218 nodes (172.75 seconds) Cbc0010I After 2300 nodes, 587 on tree,
897 best solution, best possible 682.11199 (192.91 seconds) Cbc0038I Full
problem 2890 rows 1451 columns, reduced to 1821 rows 746 columns - 1 fixed
gives 1818, 744 - still too large Cbc0038I Full problem 2890 rows 1451
columns, reduced to 1810 rows 740 columns - too large Cbc0010I After 2400
nodes, 638 on tree, 897 best solution, best possible 682.11199 (206.71
seconds) Cbc0010I After 2500 nodes, 678 on tree, 897 best solution, best
possible 682.11199 (216.18 seconds) Cbc0010I After 2600 nodes, 718 on
tree, 897 best solution, best possible 682.11199 (222.99 seconds) Cbc0010I
After 2700 nodes, 764 on tree, 897 best solution, best possible 682.11199
(230.27 seconds) Cbc0010I After 2800 nodes, 811 on tree, 897 best
solution, best possible 682.11199 (238.46 seconds) Cbc0010I After 2900
nodes, 834 on tree, 897 best solution, best possible 682.11199 (244.36
seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to 1857
rows 770 columns - 4 fixed gives 1845, 762 - still too large Cbc0038I Full
problem 2890 rows 1451 columns, reduced to 1837 rows 758 columns - too
large Cbc0010I After 3000 nodes, 879 on tree, 897 best solution, best
possible 682.11199 (252.88 seconds) Cbc0010I After 3100 nodes, 914 on
tree, 897 best solution, best possible 682.11199 (260.71 seconds) Cbc0010I
After 3200 nodes, 960 on tree, 897 best solution, best possible 682.11199
(270.26 seconds) Cbc0010I After 3300 nodes, 1003 on tree, 897 best
solution, best possible 682.11199 (280.57 seconds) Cbc0010I After 3400
nodes, 1052 on tree, 897 best solution, best possible 682.11199 (289.26
seconds) Cbc0010I After 3500 nodes, 1096 on tree, 897 best solution, best
possible 682.11199 (296.03 seconds) Cbc0038I Full problem 2890 rows 1451
columns, reduced to 1725 rows 682 columns - 2 fixed gives 1719, 678 -
still too large Cbc0038I Full problem 2890 rows 1451 columns, reduced to
1711 rows 674 columns - too large Cbc0010I After 3600 nodes, 1120 on tree,
897 best solution, best possible 682.11199 (299.83 seconds) Cbc0010I After
3700 nodes, 1151 on tree, 897 best solution, best possible 682.11199
(304.81 seconds) Cbc0010I After 3800 nodes, 1189 on tree, 897 best
solution, best possible 682.11199 (312.51 seconds) Cbc0010I After 3900
nodes, 1231 on tree, 897 best solution, best possible 682.11199 (319.32
seconds) Cbc0010I After 4000 nodes, 1273 on tree, 897 best solution, best
possible 682.11199 (325.60 seconds) Cbc0010I After 4100 nodes, 1309 on
tree, 897 best solution, best possible 682.11199 (330.28 seconds) Cbc0038I
Full problem 2890 rows 1451 columns, reduced to 1656 rows 636 columns - 1
fixed gives 1653, 634 - still too large Cbc0038I Full problem 2890 rows
1451 columns, reduced to 1645 rows 630 columns - too large Cbc0010I After
4200 nodes, 1328 on tree, 897 best solution, best possible 682.11199
(333.68 seconds) Cbc0010I After 4300 nodes, 1368 on tree, 897 best
solution, best possible 682.11199 (338.49 seconds) Cbc0010I After 4400
nodes, 1401 on tree, 897 best solution, best possible 682.11199 (343.07
seconds) Cbc0010I After 4500 nodes, 1437 on tree, 897 best solution, best
possible 682.11199 (347.56 seconds) Cbc0010I After 4600 nodes, 1472 on
tree, 897 best solution, best possible 682.11199 (353.37 seconds) Cbc0010I
After 4700 nodes, 1514 on tree, 897 best solution, best possible 682.11199
(362.49 seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to
1753 rows 701 columns - 1 fixed gives 1750, 699 - still too large Cbc0038I
Full problem 2890 rows 1451 columns, reduced to 1744 rows 696 columns -
too large Cbc0010I After 4800 nodes, 1553 on tree, 897 best solution, best
possible 682.11199 (368.94 seconds) Cbc0010I After 4900 nodes, 1603 on
tree, 897 best solution, best possible 682.11199 (376.35 seconds) Cbc0010I
After 5000 nodes, 1648 on tree, 897 best solution, best possible 682.11199
(384.04 seconds) Cbc0010I After 5100 nodes, 1628 on tree, 897 best
solution, best possible 682.11199 (385.64 seconds) Cbc0010I After 5200
nodes, 1613 on tree, 897 best solution, best possible 682.11199 (387.53
nodes, 1513 on tree, 897 best solution, best possible 682.11199 (387.55
seconds) Cbc0004I Integer solution of 889 found after 1133511 iterations
and 5214 nodes (387.76 seconds) Cbc0010I After 5300 nodes, 1549 on tree,
889 best solution, best possible 682.11199 (395.63 seconds) Cbc0038I Full
problem 2890 rows 1451 columns, reduced to 1803 rows 734 columns - 3 fixed
gives 1794, 728 - still too large Cbc0038I Full problem 2890 rows 1451
columns, reduced to 1786 rows 724 columns - too large Cbc0010I After 5400
nodes, 1597 on tree, 889 best solution, best possible 682.11199 (402.52
seconds) Cbc0010I After 5500 nodes, 1642 on tree, 889 best solution, best
possible 682.11199 (408.40 seconds) Cbc0010I After 5600 nodes, 1688 on
tree, 889 best solution, best possible 682.11199 (415.23 seconds) Cbc0010I
After 5700 nodes, 1734 on tree, 889 best solution, best possible 682.11199
(421.17 seconds) Cbc0010I After 5800 nodes, 1765 on tree, 889 best
solution, best possible 682.11199 (426.11 seconds) Cbc0010I After 5900
nodes, 1806 on tree, 889 best solution, best possible 682.11199 (430.59
seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to 1829
rows 752 columns - 3 fixed gives 1820, 746 - still too large Cbc0010I
After 6000 nodes, 1849 on tree, 889 best solution, best possible 682.11199
(438.81 seconds) Cbc0010I After 6100 nodes, 1895 on tree, 889 best
solution, best possible 682.11199 (447.88 seconds) Cbc0010I After 6200
nodes, 1935 on tree, 889 best solution, best possible 682.11199 (453.79
seconds) Cbc0010I After 6300 nodes, 1980 on tree, 889 best solution, best
possible 682.11199 (460.38 seconds) Cbc0010I After 6400 nodes, 2026 on
tree, 889 best solution, best possible 682.11199 (467.50 seconds) Cbc0010I
After 6500 nodes, 2077 on tree, 889 best solution, best possible 682.11199
(474.42 seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to
1774 rows 715 columns - 4 fixed gives 1762, 707 - still too large Cbc0038I
Full problem 2890 rows 1451 columns, reduced to 1756 rows 704 columns -
too large Cbc0010I After 6600 nodes, 2127 on tree, 889 best solution, best
possible 682.11199 (480.82 seconds) Cbc0010I After 6700 nodes, 2170 on
tree, 889 best solution, best possible 682.11199 (488.06 seconds) Cbc0010I
After 6800 nodes, 2214 on tree, 889 best solution, best possible 682.11199
(495.06 seconds) Cbc0010I After 6900 nodes, 2252 on tree, 889 best
solution, best possible 682.11199 (500.87 seconds) Cbc0010I After 7000
nodes, 2293 on tree, 889 best solution, best possible 682.11199 (506.50
seconds) Cbc0010I After 7100 nodes, 2334 on tree, 889 best solution, best
possible 682.11199 (512.29 seconds) Cbc0038I Full problem 2890 rows 1451
columns, reduced to 1783 rows 721 columns - 1 fixed gives 1780, 719 -
still too large Cbc0038I Full problem 2890 rows 1451 columns, reduced to
1774 rows 716 columns - too large Cbc0010I After 7200 nodes, 2378 on tree,
889 best solution, best possible 682.11199 (518.48 seconds) Cbc0010I After
7300 nodes, 2416 on tree, 889 best solution, best possible 682.11199
(523.77 seconds) Cbc0010I After 7400 nodes, 2460 on tree, 889 best
solution, best possible 682.11199 (530.39 seconds) Cbc0010I After 7500
nodes, 2501 on tree, 889 best solution, best possible 682.11199 (537.20
seconds) Cbc0010I After 7600 nodes, 2540 on tree, 889 best solution, best
possible 682.11199 (542.48 seconds) Cbc0010I After 7700 nodes, 2581 on
tree, 889 best solution, best possible 682.11199 (547.17 seconds) Cbc0010I
After 7800 nodes, 2615 on tree, 889 best solution, best possible 682.11199
(551.03 seconds) Cbc0010I After 7900 nodes, 2642 on tree, 889 best
solution, best possible 682.11199 (555.95 seconds) Cbc0010I After 8000
nodes, 2677 on tree, 889 best solution, best possible 682.11199 (560.56
seconds) Cbc0010I After 8100 nodes, 2705 on tree, 889 best solution, best
possible 682.11199 (565.32 seconds) Cbc0010I After 8200 nodes, 2747 on
tree, 889 best solution, best possible 682.11199 (573.65 seconds) Cbc0010I
After 8300 nodes, 2780 on tree, 889 best solution, best possible 682.11199
(578.17 seconds) Cbc0010I After 8400 nodes, 2806 on tree, 889 best
solution, best possible 682.11199 (582.03 seconds) Cbc0010I After 8500
nodes, 2832 on tree, 889 best solution, best possible 682.11199 (585.60
seconds) Cbc0010I After 8600 nodes, 2865 on tree, 889 best solution, best
possible 682.11199 (590.66 seconds) Cbc0010I After 8700 nodes, 2905 on
tree, 889 best solution, best possible 682.11199 (597.21 seconds) Cbc0010I
After 8800 nodes, 2953 on tree, 889 best solution, best possible 682.11199
(605.35 seconds) Cbc0010I After 8900 nodes, 2996 on tree, 889 best
solution, best possible 682.11199 (613.53 seconds) Cbc0010I After 9000
nodes, 3029 on tree, 889 best solution, best possible 682.11199 (621.04
seconds) Cbc0010I After 9100 nodes, 3075 on tree, 889 best solution, best
possible 682.11199 (627.92 seconds) Cbc0010I After 9200 nodes, 3066 on
tree, 889 best solution, best possible 682.11199 (631.29 seconds) Cbc0010I
```

```
After 9300 nodes, 3077 on tree, 889 best solution, best possible 682.11199
(634.53 seconds) Cbc0010I After 9400 nodes, 3066 on tree, 889 best
solution, best possible 682.11199 (637.27 seconds) Cbc0010I After 9500
nodes, 3060 on tree, 889 best solution, best possible 682.11199 (640.13
seconds) Cbc0010I After 9600 nodes, 3074 on tree, 889 best solution, best
possible 682.11199 (643.69 seconds) Cbc0010I After 9700 nodes, 3061 on
tree, 889 best solution, best possible 682.11199 (646.84 seconds) Cbc0010I
After 9800 nodes, 3076 on tree, 889 best solution, best possible 682.11199
(649.77 seconds) Cbc0010I After 9900 nodes, 3071 on tree, 889 best
solution, best possible 682.11199 (652.28 seconds) Cbc0010I After 10000
nodes, 3072 on tree, 889 best solution, best possible 682.11199 (654.44
seconds) Cbc0010I After 10100 nodes, 3074 on tree, 889 best solution, best
possible 682.11199 (657.05 seconds) Cbc0010I After 10200 nodes, 3071 on
tree, 889 best solution, best possible 682.11199 (660.70 seconds) Cbc0010I
After 10300 nodes, 3071 on tree, 889 best solution, best possible
682.11199 (663.65 seconds) Cbc0010I After 10400 nodes, 3076 on tree, 889
best solution, best possible 682.11199 (666.72 seconds) Cbc0010I After
10500 nodes, 3068 on tree, 889 best solution, best possible 682.11199
(669.80 seconds) Cbc0010I After 10600 nodes, 3066 on tree, 889 best
solution, best possible 682.11199 (672.84 seconds) Cbc0010I After 10700
nodes, 3066 on tree, 889 best solution, best possible 682.11199 (676.16
seconds) Cbc0010I After 10800 nodes, 3076 on tree, 889 best solution, best
possible 682.11199 (680.44 seconds) Cbc0010I After 10900 nodes, 3067 on
tree, 889 best solution, best possible 682.11199 (683.40 seconds) Cbc0010I
After 11000 nodes, 3073 on tree, 889 best solution, best possible
682.11199 (686.38 seconds) Cbc0010I After 11100 nodes, 3123 on tree, 889
best solution, best possible 686.18691 (751.86 seconds) Cbc0010I After
11200 nodes, 3167 on tree, 889 best solution, best possible 700.41015
(791.68 seconds) Cbc0010I After 11300 nodes, 3217 on tree, 889 best
solution, best possible 711.83532 (818.62 seconds) Cbc0010I After 11400
nodes, 3261 on tree, 889 best solution, best possible 715.25415 (847.23
seconds) Cbc0010I After 11500 nodes, 3310 on tree, 889 best solution, best
possible 718.8264 (866.97 seconds) Cbc0010I After 11600 nodes, 3355 on
tree, 889 best solution, best possible 722.46862 (882.54 seconds) Cbc0010I
After 11700 nodes, 3405 on tree, 889 best solution, best possible 723.3876
(898.20 seconds) Cbc0010I After 11800 nodes, 3455 on tree, 889 best
solution, best possible 724.10366 (912.29 seconds) Cbc0010I After 11900
nodes, 3505 on tree, 889 best solution, best possible 725.79412 (928.09
seconds) Cbc0010I After 12000 nodes, 3555 on tree, 889 best solution, best
possible 726.70413 (941.71 seconds) Cbc0010I After 12100 nodes, 3602 on
tree, 889 best solution, best possible 728.00778 (960.26 seconds) Cbc0010I
After 12200 nodes, 3652 on tree, 889 best solution, best possible
728.54962 (978.17 seconds) Cbc0010I After 12300 nodes, 3700 on tree, 889
best solution, best possible 728.72002 (988.83 seconds) Cbc0010I After
12400 nodes, 3750 on tree, 889 best solution, best possible 728.99992
(1006.50 seconds) Cbc0010I After 12500 nodes, 3800 on tree, 889 best
solution, best possible 729.0175 (1025.90 seconds) Cbc0010I After 12600
nodes, 3849 on tree, 889 best solution, best possible 729.05658 (1041.18
seconds) Cbc0010I After 12700 nodes, 3898 on tree, 889 best solution, best
possible 729.11291 (1054.89 seconds) Cbc0010I After 12800 nodes, 3945 on
tree, 889 best solution, best possible 729.23024 (1069.36 seconds)
Cbc0010I After 12900 nodes, 3994 on tree, 889 best solution, best possible
729.26201 (1083.11 seconds) Cbc0010I After 13000 nodes, 4042 on tree, 889
best solution, best possible 729.45193 (1096.44 seconds) Cbc0010I After
13100 nodes, 4069 on tree, 889 best solution, best possible 729.45193
(1100.43 seconds) Cbc0010I After 13200 nodes, 4060 on tree, 889 best
solution, best possible 729.45193 (1103.07 seconds) Cbc0010I After 13300
nodes, 4060 on tree, 889 best solution, best possible 729.45193 (1105.58
seconds) Cbc0010I After 13400 nodes, 4058 on tree, 889 best solution, best
possible 729.45193 (1108.24 seconds) Cbc0010I After 13500 nodes, 4061 on
tree, 889 best solution, best possible 729.45193 (1111.28 seconds)
Cbc0010I After 13600 nodes, 4055 on tree, 889 best solution, best possible
729.45193 (1114.01 seconds) Cbc0010I After 13700 nodes, 4060 on tree, 889
best solution, best possible 729.45193 (1117.31 seconds) Cbc0010I After
13800 nodes, 4053 on tree, 889 best solution, best possible 729.45193
(1120.99 seconds) Cbc0010I After 13900 nodes, 4050 on tree, 889 best
solution, best possible 729.45193 (1124.83 seconds) Cbc0010I After 14000
nodes, 4056 on tree, 889 best solution, best possible 729.45193 (1127.83
seconds) Cbc0010I After 14100 nodes, 4105 on tree, 889 best solution, best
```

```
seconds) Cbc0010I After 14100 nodes, 4105 on tree, 889 best solution, best
possible 729.67153 (1140.12 seconds) Cbc0010I After 14200 nodes, 4153 on
tree, 889 best solution, best possible 729.75885 (1153.37 seconds)
Cbc0010I After 14300 nodes, 4201 on tree, 889 best solution, best possible
729.90514 (1165.94 seconds) Cbc0038I Full problem 2890 rows 1451 columns,
reduced to 1952 rows 833 columns - 1 fixed gives 1949, 831 - still too
large Cbc0010I After 14400 nodes, 4251 on tree, 889 best solution, best
possible 730.02826 (1177.60 seconds) Cbc0010I After 14500 nodes, 4299 on
tree, 889 best solution, best possible 730.32972 (1191.06 seconds)
Cbc0010I After 14600 nodes, 4347 on tree, 889 best solution, best possible
730.52212 (1202.16 seconds) Cbc0010I After 14700 nodes, 4395 on tree, 889
best solution, best possible 730.52215 (1211.07 seconds) Cbc0010I After
14800 nodes, 4444 on tree, 889 best solution, best possible 730.75517
(1224.02 seconds) Cbc0010I After 14900 nodes, 4492 on tree, 889 best
solution, best possible 730.76288 (1234.17 seconds) Cbc0010I After 15000
nodes, 4540 on tree, 889 best solution, best possible 730.76452 (1243.83
seconds) Cbc0010I After 15100 nodes, 4579 on tree, 889 best solution, best
possible 730.76491 (1253.21 seconds) Cbc0010I After 15200 nodes, 4627 on
tree, 889 best solution, best possible 730.88663 (1264.42 seconds)
Cbc0010I After 15300 nodes, 4675 on tree, 889 best solution, best possible
731.12728 (1278.40 seconds) Cbc0010I After 15400 nodes, 4725 on tree, 889
best solution, best possible 731.70567 (1293.29 seconds) Cbc0010I After
15500 nodes, 4774 on tree, 889 best solution, best possible 731.88981
(1304.89 seconds) Cbc0010I After 15600 nodes, 4823 on tree, 889 best
solution, best possible 732.09451 (1317.70 seconds) Cbc0010I After 15700
nodes, 4872 on tree, 889 best solution, best possible 732.26423 (1328.91
seconds) Cbc0010I After 15800 nodes, 4922 on tree, 889 best solution, best
possible 732.29782 (1337.26 seconds) Cbc0010I After 15900 nodes, 4972 on
tree, 889 best solution, best possible 732.43945 (1349.34 seconds)
Cbc0010I After 16000 nodes, 5020 on tree, 889 best solution, best possible
732.66332 (1361.22 seconds) Cbc0010I After 16100 nodes, 5061 on tree, 889
best solution, best possible 732.70552 (1369.41 seconds) Cbc0010I After
16200 nodes, 5105 on tree, 889 best solution, best possible 732.77095
(1379.22 seconds) Cbc0010I After 16300 nodes, 5154 on tree, 889 best
solution, best possible 732.84891 (1392.29 seconds) Cbc0010I After 16400
nodes, 5202 on tree, 889 best solution, best possible 732.99927 (1405.42
seconds) Cbc0010I After 16500 nodes, 5252 on tree, 889 best solution, best
possible 733 (1421.74 seconds) Cbc0010I After 16600 nodes, 5300 on tree,
889 best solution, best possible 733 (1437.09 seconds) Cbc0010I After
16700 nodes, 5348 on tree, 889 best solution, best possible 733.07999
(1453.39 seconds) Cbc0010I After 16800 nodes, 5392 on tree, 889 best
solution, best possible 733.10336 (1465.95 seconds) Cbc0010I After 16900
nodes, 5441 on tree, 889 best solution, best possible 733.10336 (1477.46
seconds) Cbc0010I After 17000 nodes, 5489 on tree, 889 best solution, best
possible 733.10338 (1488.87 seconds) Cbc0010I After 17100 nodes, 5505 on
tree, 889 best solution, best possible 733.10338 (1492.93 seconds)
Cbc0010I After 17200 nodes, 5497 on tree, 889 best solution, best possible
733.10338 (1497.12 seconds) Cbc0010I After 17300 nodes, 5523 on tree, 889
best solution, best possible 733.10338 (1500.93 seconds) Cbc0010I After
17400 nodes, 5507 on tree, 889 best solution, best possible 733.10338
(1503.52 seconds) Cbc0010I After 17500 nodes, 5504 on tree, 889 best
solution, best possible 733.10338 (1506.32 seconds) Cbc0016I Integer
solution of 882 found by strong branching after 4993588 iterations and
17553 nodes (1507.69 seconds) Cbc0010I After 17600 nodes, 5277 on tree,
882 best solution, best possible 733.10338 (1512.99 seconds) Cbc0010I
After 17700 nodes, 5323 on tree, 882 best solution, best possible
733.10338 (1523.10 seconds) Cbc0010I After 17800 nodes, 5370 on tree, 882
best solution, best possible 733.10338 (1533.36 seconds) Cbc0010I After
17900 nodes, 5419 on tree, 882 best solution, best possible 733.10338
(1544.10 seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to
1899 rows 798 columns - 2 fixed gives 1893, 794 - still too large Cbc0010I
After 18000 nodes, 5467 on tree, 882 best solution, best possible
733.10338 (1554.88 seconds) Cbc0010I After 18100 nodes, 5516 on tree, 882
best solution, best possible 733.3401 (1568.29 seconds) Cbc0010I After
18200 nodes, 5566 on tree, 882 best solution, best possible 733.46123
(1581.49 seconds) Cbc0010I After 18300 nodes, 5614 on tree, 882 best
solution, best possible 733.57827 (1593.84 seconds) Cbc0010I After 18400
nodes, 5661 on tree, 882 best solution, best possible 733.80484 (1607.33
seconds) Cbc0010I After 18500 nodes, 5710 on tree, 882 best solution, best
```

```
possible 734.08726 (1620.18 seconds) Cbc0010I After 18600 nodes, 5757 on
tree, 882 best solution, best possible 734.20593 (1632.05 seconds)
Cbc0010I After 18700 nodes, 5806 on tree, 882 best solution, best possible
734.44881 (1643.71 seconds) Cbc0010I After 18800 nodes, 5855 on tree, 882
best solution, best possible 734.62801 (1656.89 seconds) Cbc0010I After
18900 nodes, 5904 on tree, 882 best solution, best possible 734.71231
(1671.00 seconds) Cbc0010I After 19000 nodes, 5950 on tree, 882 best
solution, best possible 734.82349 (1683.76 seconds) Cbc0010I After 19100
nodes, 5999 on tree, 882 best solution, best possible 735.07555 (1695.65
seconds) Cbc0010I After 19200 nodes, 6045 on tree, 882 best solution, best
possible 735.29913 (1708.17 seconds) Cbc0010I After 19300 nodes, 6092 on
tree, 882 best solution, best possible 735.60607 (1720.55 seconds)
Cbc0010I After 19400 nodes, 6141 on tree, 882 best solution, best possible
735.86852 (1733.62 seconds) Cbc0010I After 19500 nodes, 6190 on tree, 882
best solution, best possible 736.07229 (1749.52 seconds) Cbc0010I After
19600 nodes, 6238 on tree, 882 best solution, best possible 736.2792
(1762.00 seconds) Cbc0010I After 19700 nodes, 6288 on tree, 882 best
solution, best possible 736.41781 (1774.57 seconds) Cbc0010I After 19800
nodes, 6338 on tree, 882 best solution, best possible 736.69957 (1785.89
seconds) Cbc0010I After 19900 nodes, 6386 on tree, 882 best solution, best
possible 736.90498 (1798.08 seconds) Cbc0010I After 20000 nodes, 6435 on
tree, 882 best solution, best possible 737.12255 (1811.84 seconds)
Cbc0010I After 20100 nodes, 6483 on tree, 882 best solution, best possible
737.46756 (1824.72 seconds) Cbc0010I After 20200 nodes, 6531 on tree, 882
best solution, best possible 737.75776 (1836.25 seconds) Cbc0010I After
20300 nodes, 6579 on tree, 882 best solution, best possible 737.93748
(1846.52 seconds) Cbc0010I After 20400 nodes, 6627 on tree, 882 best
solution, best possible 738.08643 (1860.42 seconds) Cbc0010I After 20500
nodes, 6677 on tree, 882 best solution, best possible 738.21701 (1873.98
seconds) Cbc0010I After 20600 nodes, 6723 on tree, 882 best solution, best
possible 738.53513 (1888.31 seconds) Cbc0010I After 20700 nodes, 6769 on
tree, 882 best solution, best possible 738.73596 (1900.05 seconds)
Cbc0010I After 20800 nodes, 6818 on tree, 882 best solution, best possible
738.96624 (1912.10 seconds) Cbc0010I After 20900 nodes, 6867 on tree, 882
best solution, best possible 739 (1924.70 seconds) Cbc0010I After 21000
nodes, 6913 on tree, 882 best solution, best possible 739.115 (1939.04
seconds) Cbc0010I After 21100 nodes, 6917 on tree, 882 best solution, best
possible 739.115 (1942.67 seconds) Cbc0010I After 21200 nodes, 6927 on
tree, 882 best solution, best possible 739.115 (1947.10 seconds) Cbc0010I
After 21300 nodes, 6924 on tree, 882 best solution, best possible 739.115
(1949.91 seconds) Cbc0010I After 21400 nodes, 6920 on tree, 882 best
solution, best possible 739.115 (1952.50 seconds) Cbc0010I After 21500
nodes, 6928 on tree, 882 best solution, best possible 739.115 (1954.79
seconds) Cbc0010I After 21600 nodes, 6924 on tree, 882 best solution, best
possible 739.115 (1956.62 seconds) Cbc0010I After 21700 nodes, 6921 on
tree, 882 best solution, best possible 739.115 (1960.12 seconds) Cbc0010I
After 21800 nodes, 6923 on tree, 882 best solution, best possible 739.115
(1964.12 seconds) Cbc0010I After 21900 nodes, 6922 on tree, 882 best
solution, best possible 739.115 (1967.49 seconds) Cbc0010I After 22000
nodes, 6922 on tree, 882 best solution, best possible 739.115 (1970.57
seconds) Cbc0010I After 22100 nodes, 6972 on tree, 882 best solution, best
possible 739.24296 (1986.87 seconds) Cbc0010I After 22200 nodes, 7017 on
tree, 882 best solution, best possible 739.37994 (2003.74 seconds)
Cbc0010I After 22300 nodes, 7064 on tree, 882 best solution, best possible
739.60326 (2017.23 seconds) Cbc0010I After 22400 nodes, 7111 on tree, 882
best solution, best possible 740 (2035.32 seconds) Cbc0010I After 22500
nodes, 7161 on tree, 882 best solution, best possible 740.24006 (2050.10
seconds) Cbc0010I After 22600 nodes, 7210 on tree, 882 best solution, best
possible 740.46446 (2061.60 seconds) Cbc0010I After 22700 nodes, 7258 on
tree, 882 best solution, best possible 740.76773 (2075.83 seconds)
Cbc0010I After 22800 nodes, 7307 on tree, 882 best solution, best possible
740.99991 (2088.74 seconds) Cbc0010I After 22900 nodes, 7356 on tree, 882
best solution, best possible 741.19873 (2104.03 seconds) Cbc0010I After
23000 nodes, 7404 on tree, 882 best solution, best possible 741.49656
(2117.32 seconds) Cbc0010I After 23100 nodes, 7452 on tree, 882 best
solution, best possible 741.73392 (2133.11 seconds) Cbc0010I After 23200
nodes, 7501 on tree, 882 best solution, best possible 741.99774 (2147.38
seconds) Cbc0010I After 23300 nodes, 7551 on tree, 882 best solution, best
```

```
possible 742.06794 (2161.51 seconds) Cbc0010I After 23400 nodes, 7601 on
tree, 882 best solution, best possible 742.32271 (2175.24 seconds)
Cbc0010I After 23500 nodes, 7650 on tree, 882 best solution, best possible
742.48402 (2189.04 seconds) Cbc0010I After 23600 nodes, 7699 on tree, 882
best solution, best possible 742.77206 (2202.06 seconds) Cbc0010I After
23700 nodes, 7746 on tree, 882 best solution, best possible 742.88849
(2215.79 seconds) Cbc0010I After 23800 nodes, 7793 on tree, 882 best
solution, best possible 743.08754 (2229.42 seconds) Cbc0010I After 23900
nodes, 7842 on tree, 882 best solution, best possible 743.32798 (2242.52
seconds) Cbc0010I After 24000 nodes, 7889 on tree, 882 best solution, best
possible 743.58079 (2254.53 seconds) Cbc0010I After 24100 nodes, 7938 on
tree, 882 best solution, best possible 743.84176 (2267.73 seconds)
Cbc0010I After 24200 nodes, 7986 on tree, 882 best solution, best possible
744.02519 (2281.98 seconds) Cbc0010I After 24300 nodes, 8034 on tree, 882
best solution, best possible 744.22381 (2296.40 seconds) Cbc0010I After
24400 nodes, 8081 on tree, 882 best solution, best possible 744.56534
(2309.17 seconds) Cbc0010I After 24500 nodes, 8129 on tree, 882 best
solution, best possible 744.81551 (2323.49 seconds) Cbc0010I After 24600
nodes, 8173 on tree, 882 best solution, best possible 744.93327 (2334.52
seconds) Cbc0010I After 24700 nodes, 8220 on tree, 882 best solution, best
possible 744.99929 (2346.90 seconds) Cbc0010I After 24800 nodes, 8268 on
tree, 882 best solution, best possible 744.99974 (2359.83 seconds)
Cbc0010I After 24900 nodes, 8317 on tree, 882 best solution, best possible
744.99979 (2375.28 seconds) Cbc0010I After 25000 nodes, 8367 on tree, 882
best solution, best possible 744.9998 (2385.34 seconds) Cbc0010I After
25100 nodes, 8379 on tree, 882 best solution, best possible 744.9998
(2390.18 seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to
1799 rows 732 columns - 2 fixed gives 1793, 728 - still too large Cbc0010I
After 25200 nodes, 8381 on tree, 882 best solution, best possible 744.9998
(2394.11 seconds) Cbc0010I After 25300 nodes, 8375 on tree, 882 best
solution, best possible 744.9998 (2398.91 seconds) Cbc0010I After 25400
nodes, 8370 on tree, 882 best solution, best possible 744.9998 (2404.02
seconds) Cbc0010I After 25500 nodes, 8375 on tree, 882 best solution, best
possible 744.9998 (2408.97 seconds) Cbc0010I After 25600 nodes, 8373 on
tree, 882 best solution, best possible 744.9998 (2413.47 seconds) Cbc0010I
After 25700 nodes, 8384 on tree, 882 best solution, best possible 744.9998
(2417.12 seconds) Cbc0010I After 25800 nodes, 8384 on tree, 882 best
solution, best possible 744.9998 (2420.25 seconds) Cbc0010I After 25900
nodes, 8379 on tree, 882 best solution, best possible 744.9998 (2423.10
seconds) Cbc0010I After 26000 nodes, 8378 on tree, 882 best solution, best
possible 744.9998 (2426.36 seconds) Cbc0010I After 26100 nodes, 8428 on
tree, 882 best solution, best possible 744.99983 (2439.28 seconds)
Cbc0010I After 26200 nodes, 8476 on tree, 882 best solution, best possible
745 (2452.64 seconds) Cbc0010I After 26300 nodes, 8524 on tree, 882 best
solution, best possible 745.00488 (2465.77 seconds) Cbc0010I After 26400
nodes, 8571 on tree, 882 best solution, best possible 745.03137 (2477.56
seconds) Cbc0010I After 26500 nodes, 8619 on tree, 882 best solution, best
possible 745.21208 (2489.63 seconds) Cbc0010I After 26600 nodes, 8664 on
tree, 882 best solution, best possible 745.33924 (2502.24 seconds)
Cbc0010I After 26700 nodes, 8706 on tree, 882 best solution, best possible
745.41945 (2512.23 seconds) Cbc0010I After 26800 nodes, 8753 on tree, 882
best solution, best possible 745.64595 (2525.28 seconds) Cbc0010I After
26900 nodes, 8801 on tree, 882 best solution, best possible 745.82833
(2537.43 seconds) Cbc0010I After 27000 nodes, 8849 on tree, 882 best
solution, best possible 746.0656 (2551.34 seconds) Cbc0010I After 27100
nodes, 8897 on tree, 882 best solution, best possible 746.23246 (2564.95
seconds) Cbc0010I After 27200 nodes, 8944 on tree, 882 best solution, best
possible 746.37669 (2577.08 seconds) Cbc0010I After 27300 nodes, 8990 on
tree, 882 best solution, best possible 746.46401 (2598.91 seconds)
Cbc0010I After 27400 nodes, 9034 on tree, 882 best solution, best possible
746.48485 (2609.84 seconds) Cbc0010I After 27500 nodes, 9083 on tree, 882
best solution, best possible 746.59214 (2621.37 seconds) Cbc0010I After
27600 nodes, 9129 on tree, 882 best solution, best possible 746.77434
(2634.25 seconds) Cbc0010I After 27700 nodes, 9174 on tree, 882 best
solution, best possible 746.97052 (2646.12 seconds) Cbc0010I After 27800
nodes, 9221 on tree, 882 best solution, best possible 747 (2660.41
seconds) Cbc0010I After 27900 nodes, 9268 on tree, 882 best solution, best
possible 747.12041 (2673.07 seconds) Cbc0010I After 28000 nodes, 9315 on
tree, 882 best solution, best possible 747.24282 (2685.60 seconds)
```

```
Cbc0010I After 28100 nodes, 9363 on tree, 882 best solution, best possible
747.3398 (2697.46 seconds) Cbc0010I After 28200 nodes, 9411 on tree, 882
best solution, best possible 747.43881 (2709.62 seconds) Cbc0010I After
28300 nodes, 9455 on tree, 882 best solution, best possible 747.63329
(2722.04 seconds) Cbc0010I After 28400 nodes, 9502 on tree, 882 best
solution, best possible 747.8581 (2735.72 seconds) Cbc0010I After 28500
nodes, 9549 on tree, 882 best solution, best possible 747.99928 (2748.03
seconds) Cbc0010I After 28600 nodes, 9596 on tree, 882 best solution, best
possible 748.04545 (2759.53 seconds) Cbc0010I After 28700 nodes, 9642 on
tree, 882 best solution, best possible 748.2553 (2773.36 seconds) Cbc0038I
Full problem 2890 rows 1451 columns, reduced to 2041 rows 883 columns - 1
fixed gives 2038, 881 - still too large Cbc0010I After 28800 nodes, 9690
on tree, 882 best solution, best possible 748.41111 (2786.39 seconds)
Cbc0010I After 28900 nodes, 9737 on tree, 882 best solution, best possible
748.57325 (2800.52 seconds) Cbc0010I After 29000 nodes, 9784 on tree, 882
best solution, best possible 748.67239 (2811.61 seconds) Cbc0010I After
29100 nodes, 9790 on tree, 882 best solution, best possible 748.67239
(2816.67 seconds) Cbc0010I After 29200 nodes, 9789 on tree, 882 best
solution, best possible 748.67239 (2820.03 seconds) Cbc0010I After 29300
nodes, 9800 on tree, 882 best solution, best possible 748.67239 (2824.65
seconds) Cbc0010I After 29400 nodes, 9794 on tree, 882 best solution, best
possible 748.67239 (2828.17 seconds) Cbc0010I After 29500 nodes, 9792 on
tree, 882 best solution, best possible 748.67239 (2832.67 seconds)
Cbc0010I After 29600 nodes, 9788 on tree, 882 best solution, best possible
748.67239 (2835.45 seconds) Cbc0010I After 29700 nodes, 9785 on tree, 882
best solution, best possible 748.67239 (2839.65 seconds) Cbc0010I After
29800 nodes, 9773 on tree, 882 best solution, best possible 748.67239
(2842.17 seconds) Cbc0010I After 29900 nodes, 9761 on tree, 882 best
solution, best possible 748.67239 (2844.48 seconds) Cbc0010I After 30000
nodes, 9747 on tree, 882 best solution, best possible 748.67239 (2846.39
seconds) Cbc0010I After 30100 nodes, 9794 on tree, 882 best solution, best
possible 748.77244 (2858.89 seconds) Cbc0010I After 30200 nodes, 9844 on
tree, 882 best solution, best possible 748.89377 (2871.43 seconds)
Cbc0010I After 30300 nodes, 9891 on tree, 882 best solution, best possible
748.99979 (2884.38 seconds) Cbc0010I After 30400 nodes, 9939 on tree, 882
best solution, best possible 749 (2906.00 seconds) Cbc0010I After 30500
nodes, 9981 on tree, 882 best solution, best possible 749.02192 (2926.05
seconds) Cbc0010I After 30600 nodes, 10029 on tree, 882 best solution,
best possible 749.15506 (2939.29 seconds) Cbc0010I After 30700 nodes,
10080 on tree, 882 best solution, best possible 749.16161 (2951.48
seconds) Cbc0010I After 30800 nodes, 10124 on tree, 882 best solution,
best possible 749.27708 (2964.89 seconds) Cbc0010I After 30900 nodes,
10167 on tree, 882 best solution, best possible 749.37811 (2978.89
seconds) Cbc0010I After 31000 nodes, 10214 on tree, 882 best solution,
best possible 749.53188 (2993.11 seconds) Cbc0010I After 31100 nodes,
10260 on tree, 882 best solution, best possible 749.53188 (3004.64
seconds) Cbc0010I After 31200 nodes, 10309 on tree, 882 best solution,
best possible 749.53188 (3014.48 seconds) Cbc0010I After 31300 nodes,
10357 on tree, 882 best solution, best possible 749.53188 (3024.49
seconds) Cbc0010I After 31400 nodes, 10403 on tree, 882 best solution,
best possible 749.53188 (3035.61 seconds) Cbc0010I After 31500 nodes,
10449 on tree, 882 best solution, best possible 749.53188 (3045.83
seconds) Cbc0010I After 31600 nodes, 10495 on tree, 882 best solution,
best possible 749.53188 (3056.29 seconds) Cbc0010I After 31700 nodes,
10544 on tree, 882 best solution, best possible 749.53188 (3067.94
seconds) Cbc0010I After 31800 nodes, 10590 on tree, 882 best solution,
best possible 749.53188 (3077.88 seconds) Cbc0010I After 31900 nodes,
10638 on tree, 882 best solution, best possible 749.53188 (3088.42
seconds) Cbc0010I After 32000 nodes, 10682 on tree, 882 best solution,
best possible 749.53188 (3099.48 seconds) Cbc0010I After 32100 nodes,
10675 on tree, 882 best solution, best possible 749.53188 (3101.15
seconds) Cbc0012I Integer solution of 880 found by rounding after 11235570
iterations and 32108 nodes (3101.28 seconds) Cbc0010I After 32200 nodes,
10641 on tree, 880 best solution, best possible 749.53188 (3109.67
seconds) Cbc0010I After 32300 nodes, 10684 on tree, 880 best solution,
best possible 749.53188 (3120.28 seconds) Cbc0038I Full problem 2890 rows
1451 columns, reduced to 1947 rows 820 columns - 2 fixed gives 1941, 816 -
still too large Cbc0010I After 32400 nodes, 10731 on tree, 880 best
```

```
solution, best possible 749.53188 (3129.47 seconds) Cbc0010I After 32500
nodes, 10775 on tree, 880 best solution, best possible 749.53188 (3140.58
seconds) Cbc0010I After 32600 nodes, 10822 on tree, 880 best solution,
best possible 749.53188 (3150.13 seconds) Cbc0010I After 32700 nodes,
10866 on tree, 880 best solution, best possible 749.53188 (3160.99
seconds) Cbc0010I After 32800 nodes, 10911 on tree, 880 best solution,
best possible 749.53188 (3170.81 seconds) Cbc0010I After 32900 nodes,
10957 on tree, 880 best solution, best possible 749.53188 (3180.96
seconds) Cbc0010I After 33000 nodes, 11005 on tree, 880 best solution,
best possible 749.53188 (3191.91 seconds) Cbc0010I After 33100 nodes,
11028 on tree, 880 best solution, best possible 749.53377 (3196.50
seconds) Cbc0010I After 33200 nodes, 11027 on tree, 880 best solution,
best possible 749.53377 (3200.86 seconds) Cbc0010I After 33300 nodes,
11041 on tree, 880 best solution, best possible 749.53377 (3203.84
seconds) Cbc0010I After 33400 nodes, 11023 on tree, 880 best solution,
best possible 749.53377 (3206.66 seconds) Cbc0010I After 33500 nodes,
11024 on tree, 880 best solution, best possible 749.53377 (3210.79
seconds) Cbc0010I After 33600 nodes, 11023 on tree, 880 best solution,
best possible 749.53377 (3215.39 seconds) Cbc0010I After 33700 nodes,
11030 on tree, 880 best solution, best possible 749.53377 (3218.72
seconds) Cbc0010I After 33800 nodes, 11022 on tree, 880 best solution,
best possible 749.53377 (3221.36 seconds) Cbc0010I After 33900 nodes,
11023 on tree, 880 best solution, best possible 749.53377 (3224.80
seconds) Cbc0010I After 34000 nodes, 11029 on tree, 880 best solution,
best possible 749.53377 (3227.75 seconds) Cbc0010I After 34100 nodes,
11075 on tree, 880 best solution, best possible 749.53377 (3239.05
seconds) Cbc0010I After 34200 nodes, 11120 on tree, 880 best solution,
best possible 749.53377 (3248.81 seconds) Cbc0010I After 34300 nodes,
11163 on tree, 880 best solution, best possible 749.53377 (3260.16
seconds) Cbc0010I After 34400 nodes, 11204 on tree, 880 best solution,
best possible 749.53377 (3269.86 seconds) Cbc0010I After 34500 nodes,
11254 on tree, 880 best solution, best possible 749.53377 (3280.74
seconds) Cbc0010I After 34600 nodes, 11297 on tree, 880 best solution,
best possible 749.53377 (3291.49 seconds) Cbc0010I After 34700 nodes,
11340 on tree, 880 best solution, best possible 749.53377 (3301.86
seconds) Cbc0010I After 34800 nodes, 11381 on tree, 880 best solution,
best possible 749.53377 (3311.84 seconds) Cbc0010I After 34900 nodes,
11427 on tree, 880 best solution, best possible 749.53377 (3322.69
seconds) Cbc0010I After 35000 nodes, 11467 on tree, 880 best solution,
best possible 749.53377 (3330.93 seconds) Cbc0010I After 35100 nodes,
11515 on tree, 880 best solution, best possible 749.53377 (3342.01
seconds) Cbc0010I After 35200 nodes, 11559 on tree, 880 best solution,
best possible 749.53377 (3352.56 seconds) Cbc0010I After 35300 nodes,
11605 on tree, 880 best solution, best possible 749.53377 (3363.13
seconds) Cbc0010I After 35400 nodes, 11646 on tree, 880 best solution,
best possible 749.53377 (3372.22 seconds) Cbc0010I After 35500 nodes,
11689 on tree, 880 best solution, best possible 749.53377 (3381.46
seconds) Cbc0010I After 35600 nodes, 11732 on tree, 880 best solution,
best possible 749.53377 (3391.87 seconds) Cbc0010I After 35700 nodes,
11776 on tree, 880 best solution, best possible 749.53377 (3402.49
seconds) Cbc0010I After 35800 nodes, 11822 on tree, 880 best solution,
best possible 749.53377 (3413.52 seconds) Cbc0010I After 35900 nodes,
11867 on tree, 880 best solution, best possible 749.53377 (3422.52
seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to 1886
rows 789 columns - 2 fixed gives 1880, 785 - still too large Cbc0010I
After 36000 nodes, 11911 on tree, 880 best solution, best possible
749.53377 (3432.69 seconds) Cbc0010I After 36100 nodes, 11899 on tree, 880
best solution, best possible 749.53377 (3434.26 seconds) Cbc0010I After
36200 nodes, 11887 on tree, 880 best solution, best possible 749.53377
(3435.69 seconds) Cbc0010I After 36300 nodes, 11879 on tree, 880 best
solution, best possible 749.53377 (3437.34 seconds) Cbc0010I After 36400
nodes, 11868 on tree, 880 best solution, best possible 749.53377 (3438.99
seconds) Cbc0010I After 36500 nodes, 11858 on tree, 880 best solution,
best possible 749.53377 (3441.02 seconds) Cbc0010I After 36600 nodes,
11851 on tree, 880 best solution, best possible 749.53377 (3442.67
seconds) Cbc0010I After 36700 nodes, 11853 on tree, 880 best solution,
best possible 749.53377 (3444.69 seconds) Cbc0010I After 36800 nodes,
11839 on tree, 880 best solution, best possible 749.53377 (3446.81
seconds) Cbc0010I After 36900 nodes, 11827 on tree, 880 best solution,
seconds) Cbc0010I After 36900 nodes, 11827 on tree, 880 best solution,
best possible 749.53377 (3449.15 seconds) Cbc0010I After 37000 nodes,
11819 on tree, 880 best solution, best possible 749.53377 (3451.03
seconds) Cbc0010I After 37100 nodes, 11838 on tree, 880 best solution,
best possible 749.53377 (3454.74 seconds) Cbc0010I After 37200 nodes,
11837 on tree, 880 best solution, best possible 749.53377 (3457.53
seconds) Cbc0010I After 37300 nodes, 11835 on tree, 880 best solution,
best possible 749.53377 (3461.33 seconds) Cbc0010I After 37400 nodes,
11833 on tree, 880 best solution, best possible 749.53377 (3465.33
seconds) Cbc0010I After 37500 nodes, 11838 on tree, 880 best solution,
best possible 749.53377 (3468.12 seconds) Cbc0010I After 37600 nodes,
11825 on tree, 880 best solution, best possible 749.53377 (3470.97
seconds) Cbc0010I After 37700 nodes, 11831 on tree, 880 best solution,
best possible 749.53377 (3474.46 seconds) Cbc0010I After 37800 nodes,
11834 on tree, 880 best solution, best possible 749.53377 (3478.49
seconds) Cbc0010I After 37900 nodes, 11825 on tree, 880 best solution,
best possible 749.53377 (3481.65 seconds) Cbc0010I After 38000 nodes,
11831 on tree, 880 best solution, best possible 749.53377 (3485.21
seconds) Cbc0010I After 38100 nodes, 11872 on tree, 880 best solution,
best possible 749.53377 (3495.96 seconds) Cbc0010I After 38200 nodes,
11918 on tree, 880 best solution, best possible 749.53377 (3506.72
seconds) Cbc0010I After 38300 nodes, 11961 on tree, 880 best solution,
best possible 749.53377 (3515.28 seconds) Cbc0010I After 38400 nodes,
12009 on tree, 880 best solution, best possible 749.53377 (3525.32
seconds) Cbc0010I After 38500 nodes, 12054 on tree, 880 best solution,
best possible 749.53377 (3536.11 seconds) Cbc0010I After 38600 nodes,
12093 on tree, 880 best solution, best possible 749.53377 (3545.49
seconds) Cbc0010I After 38700 nodes, 12137 on tree, 880 best solution,
best possible 749.53377 (3555.04 seconds) Cbc0010I After 38800 nodes,
12183 on tree, 880 best solution, best possible 749.53377 (3564.79
seconds) Cbc0010I After 38900 nodes, 12228 on tree, 880 best solution,
best possible 749.53377 (3574.20 seconds) Cbc0010I After 39000 nodes,
12269 on tree, 880 best solution, best possible 749.53377 (3584.21
seconds) Cbc0010I After 39100 nodes, 12315 on tree, 880 best solution,
best possible 749.53377 (3593.44 seconds) Cbc0010I After 39200 nodes,
12360 on tree, 880 best solution, best possible 749.53377 (3603.12
seconds) Cbc0010I After 39300 nodes, 12407 on tree, 880 best solution,
best possible 749.53377 (3612.69 seconds) Cbc0010I After 39400 nodes,
12450 on tree, 880 best solution, best possible 749.53377 (3623.39
seconds) Cbc0010I After 39500 nodes, 12490 on tree, 880 best solution,
best possible 749.53377 (3634.07 seconds) Cbc0038I Full problem 2890 rows
1451 columns, reduced to 1905 rows 802 columns - 1 fixed gives 1902, 800 -
still too large Cbc0010I After 39600 nodes, 12533 on tree, 880 best
solution, best possible 749.53377 (3643.89 seconds) Cbc0010I After 39700
nodes, 12575 on tree, 880 best solution, best possible 749.53377 (3655.23
seconds) Cbc0010I After 39800 nodes, 12623 on tree, 880 best solution,
best possible 749.53377 (3664.25 seconds) Cbc0010I After 39900 nodes,
12670 on tree, 880 best solution, best possible 749.53377 (3673.87
seconds) Cbc0010I After 40000 nodes, 12715 on tree, 880 best solution,
best possible 749.53377 (3681.33 seconds) Cbc0016I Integer solution of 878
found by strong branching after 13530063 iterations and 40069 nodes
(3683.62 seconds) Cbc0010I After 40100 nodes, 12651 on tree, 878 best
solution, best possible 749.53377 (3688.28 seconds) Cbc0010I After 40200
nodes, 12694 on tree, 878 best solution, best possible 749.53377 (3702.36
seconds) Cbc0010I After 40300 nodes, 12734 on tree, 878 best solution,
best possible 749.53377 (3713.43 seconds) Cbc0010I After 40400 nodes,
12774 on tree, 878 best solution, best possible 749.53377 (3724.82
seconds) Cbc0010I After 40500 nodes, 12817 on tree, 878 best solution,
best possible 749.53377 (3736.37 seconds) Cbc0010I After 40600 nodes,
12859 on tree, 878 best solution, best possible 749.53377 (3747.06
seconds) Cbc0010I After 40700 nodes, 12907 on tree, 878 best solution,
best possible 749.53377 (3756.49 seconds) Cbc0010I After 40800 nodes,
12952 on tree, 878 best solution, best possible 749.53377 (3767.08
seconds) Cbc0010I After 40900 nodes, 13001 on tree, 878 best solution,
best possible 749.53377 (3776.79 seconds) Cbc0010I After 41000 nodes,
13046 on tree, 878 best solution, best possible 749.53377 (3787.46
seconds) Cbc0010I After 41100 nodes, 13051 on tree, 878 best solution,
best possible 749.54718 (3794.05 seconds) Cbc0010I After 41200 nodes,
13050 on tree, 878 best solution, best possible 749.54718 (3800.56
seconds) Cbc0010I After 41300 nodes, 13047 on tree, 878 best solution,
best possible 749.54718 (3805.09 seconds) Cbc0010I After 41400 nodes,
13051 on tree, 878 best solution, best possible 749.54718 (3811.56
seconds) Cbc0010I After 41500 nodes, 13051 on tree, 878 best solution,
best possible 749.54718 (3817.33 seconds) Cbc0010I After 41600 nodes,
13054 on tree, 878 best solution, best possible 749.54718 (3822.68
seconds) Cbc0010I After 41700 nodes, 13052 on tree, 878 best solution,
best possible 749.54718 (3828.24 seconds) Cbc0010I After 41800 nodes,
13051 on tree, 878 best solution, best possible 749.54718 (3834.69
seconds) Cbc0010I After 41900 nodes, 13044 on tree, 878 best solution,
best possible 749.54718 (3838.99 seconds) Cbc0010I After 42000 nodes,
13029 on tree, 878 best solution, best possible 749.54718 (3841.82
seconds) Cbc0010I After 42100 nodes, 13078 on tree, 878 best solution,
best possible 749.54718 (3852.81 seconds) Cbc0010I After 42200 nodes,
13126 on tree, 878 best solution, best possible 749.54718 (3865.74
seconds) Cbc0010I After 42300 nodes, 13175 on tree, 878 best solution,
best possible 749.54718 (3874.16 seconds) Cbc0010I After 42400 nodes,
13219 on tree, 878 best solution, best possible 749.54718 (3885.07
seconds) Cbc0010I After 42500 nodes, 13265 on tree, 878 best solution,
best possible 749.54718 (3897.82 seconds) Cbc0010I After 42600 nodes,
13310 on tree, 878 best solution, best possible 749.54718 (3909.05
seconds) Cbc0010I After 42700 nodes, 13352 on tree, 878 best solution,
best possible 749.54718 (3920.17 seconds) Cbc0010I After 42800 nodes,
13395 on tree, 878 best solution, best possible 749.54718 (3931.87
seconds) Cbc0010I After 42900 nodes, 13438 on tree, 878 best solution,
best possible 749.54718 (3942.51 seconds) Cbc0010I After 43000 nodes,
13482 on tree, 878 best solution, best possible 749.54718 (3952.90
seconds) Cbc0010I After 43100 nodes, 13523 on tree, 878 best solution,
best possible 749.55451 (3964.29 seconds) Cbc0038I Full problem 2890 rows
1451 columns, reduced to 1819 rows 745 columns - 1 fixed gives 1816, 743 -
still too large Cbc0010I After 43200 nodes, 13569 on tree, 878 best
solution, best possible 749.55451 (3975.69 seconds) Cbc0010I After 43300
nodes, 13618 on tree, 878 best solution, best possible 749.55451 (3986.95
seconds) Cbc0010I After 43400 nodes, 13656 on tree, 878 best solution,
best possible 749.55451 (3997.08 seconds) Cbc0010I After 43500 nodes,
13705 on tree, 878 best solution, best possible 749.55451 (4007.01
seconds) Cbc0010I After 43600 nodes, 13753 on tree, 878 best solution,
best possible 749.55451 (4017.93 seconds) Cbc0010I After 43700 nodes,
13792 on tree, 878 best solution, best possible 749.55451 (4026.44
seconds) Cbc0010I After 43800 nodes, 13832 on tree, 878 best solution,
best possible 749.55451 (4033.71 seconds) Cbc0010I After 43900 nodes,
13870 on tree, 878 best solution, best possible 749.55451 (4040.20
seconds) Cbc0010I After 44000 nodes, 13909 on tree, 878 best solution,
best possible 749.55451 (4051.46 seconds) Cbc0010I After 44100 nodes,
13896 on tree, 878 best solution, best possible 749.55451 (4054.20
seconds) Cbc0010I After 44200 nodes, 13881 on tree, 878 best solution,
best possible 749.55451 (4057.33 seconds) Cbc0010I After 44300 nodes,
13881 on tree, 878 best solution, best possible 749.55451 (4059.53
seconds) Cbc0010I After 44400 nodes, 13871 on tree, 878 best solution,
best possible 749.55451 (4062.41 seconds) Cbc0010I After 44500 nodes,
13859 on tree, 878 best solution, best possible 749.55451 (4065.31
seconds) Cbc0010I After 44600 nodes, 13855 on tree, 878 best solution,
best possible 749.55451 (4067.76 seconds) Cbc0010I After 44700 nodes,
13853 on tree, 878 best solution, best possible 749.55451 (4070.76
seconds) Cbc0010I After 44800 nodes, 13853 on tree, 878 best solution,
best possible 749.55451 (4073.51 seconds) Cbc0010I After 44900 nodes,
13844 on tree, 878 best solution, best possible 749.55451 (4075.75
seconds) Cbc0010I After 45000 nodes, 13831 on tree, 878 best solution,
best possible 749.55451 (4079.06 seconds) Cbc0010I After 45100 nodes,
13841 on tree, 878 best solution, best possible 749.55451 (4085.08
seconds) Cbc0010I After 45200 nodes, 13841 on tree, 878 best solution,
best possible 749.55451 (4090.21 seconds) Cbc0010I After 45300 nodes,
13842 on tree, 878 best solution, best possible 749.55451 (4094.54
seconds) Cbc0010I After 45400 nodes, 13840 on tree, 878 best solution,
best possible 749.55451 (4098.55 seconds) Cbc0010I After 45500 nodes,
13846 on tree, 878 best solution, best possible 749.55451 (4103.05
seconds) Cbc0010I After 45600 nodes, 13848 on tree, 878 best solution,
best possible 749.55451 (4107.13 seconds) Cbc0010I After 45700 nodes,
13845 on tree, 878 best solution, best possible 749.55451 (4111.33
13845 on tree, 878 best solution, best possible 749.55451 (4111.33
seconds) Cbc0010I After 45800 nodes, 13847 on tree, 878 best solution,
best possible 749.55451 (4115.79 seconds) Cbc0010I After 45900 nodes,
13845 on tree, 878 best solution, best possible 749.55451 (4118.87
seconds) Cbc0010I After 46000 nodes, 13844 on tree, 878 best solution,
best possible 749.55451 (4123.06 seconds) Cbc0010I After 46100 nodes,
13886 on tree, 878 best solution, best possible 749.55451 (4134.15
seconds) Cbc0010I After 46200 nodes, 13933 on tree, 878 best solution,
best possible 749.55451 (4145.90 seconds) Cbc0010I After 46300 nodes,
13974 on tree, 878 best solution, best possible 749.55451 (4155.66
seconds) Cbc0010I After 46400 nodes, 14020 on tree, 878 best solution,
best possible 749.55451 (4168.16 seconds) Cbc0010I After 46500 nodes,
14063 on tree, 878 best solution, best possible 749.55451 (4177.30
seconds) Cbc0010I After 46600 nodes, 14109 on tree, 878 best solution,
best possible 749.55451 (4188.01 seconds) Cbc0010I After 46700 nodes,
14153 on tree, 878 best solution, best possible 749.55451 (4197.81
seconds) Cbc0038I Full problem 2890 rows 1451 columns, reduced to 1744
rows 695 columns - 1 fixed gives 1741, 693 - still too large Cbc0010I
After 46800 nodes, 14196 on tree, 878 best solution, best possible
749.55451 (4207.68 seconds) Cbc0010I After 46900 nodes, 14241 on tree, 878
best solution, best possible 749.55451 (4218.06 seconds) Cbc0010I After
47000 nodes, 14285 on tree, 878 best solution, best possible 749.55451
(4228.36 seconds) Cbc0010I After 47100 nodes, 14326 on tree, 878 best
solution, best possible 749.55451 (4237.31 seconds) Cbc0010I After 47200
nodes, 14372 on tree, 878 best solution, best possible 749.55451 (4247.80
seconds) Cbc0010I After 47300 nodes, 14414 on tree, 878 best solution,
best possible 749.55451 (4257.45 seconds) Cbc0010I After 47400 nodes,
14464 on tree, 878 best solution, best possible 749.55451 (4268.40
seconds) Cbc0010I After 47500 nodes, 14505 on tree, 878 best solution,
best possible 749.55451 (4277.66 seconds) Cbc0010I After 47600 nodes,
14553 on tree, 878 best solution, best possible 749.55451 (4288.76
seconds) Cbc0010I After 47700 nodes, 14598 on tree, 878 best solution,
best possible 749.55451 (4298.49 seconds) Cbc0010I After 47800 nodes,
14637 on tree, 878 best solution, best possible 749.55451 (4308.45
seconds) Cbc0010I After 47900 nodes, 14682 on tree, 878 best solution,
best possible 749.55451 (4318.82 seconds) Cbc0010I After 48000 nodes,
14720 on tree, 878 best solution, best possible 749.55451 (4329.10
seconds) Cbc0010I After 48100 nodes, 14717 on tree, 878 best solution,
best possible 749.55451 (4332.06 seconds) Cbc0010I After 48200 nodes,
14708 on tree, 878 best solution, best possible 749.55451 (4334.65
seconds) Cbc0010I After 48300 nodes, 14703 on tree, 878 best solution,
best possible 749.55451 (4337.34 seconds) Cbc0010I After 48400 nodes,
14703 on tree, 878 best solution, best possible 749.55451 (4339.13
seconds) Cbc0004I Integer solution of 877 found after 16037696 iterations
and 48455 nodes (4340.62 seconds) Cbc0010I After 48500 nodes, 14702 on
tree, 877 best solution, best possible 749.55451 (4343.84 seconds)
Cbc0010I After 48600 nodes, 14745 on tree, 877 best solution, best
possible 749.55451 (4351.93 seconds) Cbc0010I After 48700 nodes, 14788 on
tree, 877 best solution, best possible 749.55451 (4360.56 seconds)
Cbc0010I After 48800 nodes, 14832 on tree, 877 best solution, best
possible 749.55451 (4368.84 seconds) Cbc0010I After 48900 nodes, 14876 on
tree, 877 best solution, best possible 749.55451 (4379.21 seconds)
Cbc0010I After 49000 nodes, 14919 on tree, 877 best solution, best
possible 749.55451 (4388.57 seconds) Cbc0010I After 49100 nodes, 14927 on
tree, 877 best solution, best possible 749.58541 (4394.67 seconds)
Cbc0010I After 49200 nodes, 14929 on tree, 877 best solution, best
possible 749.58541 (4400.31 seconds) Cbc0010I After 49300 nodes, 14929 on
tree, 877 best solution, best possible 749.58541 (4405.30 seconds)
Cbc0010I After 49400 nodes, 14921 on tree, 877 best solution, best
possible 749.58541 (4410.88 seconds) Cbc0010I After 49500 nodes, 14915 on
tree, 877 best solution, best possible 749.58541 (4415.32 seconds)
Cbc0010I After 49600 nodes, 14915 on tree, 877 best solution, best
possible 749.58541 (4418.04 seconds) Cbc0010I After 49700 nodes, 14906 on
tree, 877 best solution, best possible 749.58541 (4420.82 seconds)
Cbc0010I After 49800 nodes, 14915 on tree, 877 best solution, best
possible 749.58541 (4423.36 seconds) Cbc0010I After 49900 nodes, 14915 on
tree, 877 best solution, best possible 749.58541 (4425.86 seconds)
Cbc0010I After 50000 nodes, 14895 on tree, 877 best solution, best
possible 749.58541 (4428.47 seconds) Cbc0010I After 50100 nodes, 14944 on
```

```
tree, 877 best solution, best possible 749.58541 (4442.85 seconds)
Cbc0010I After 50200 nodes, 14989 on tree, 877 best solution, best
possible 749.58541 (4453.77 seconds) Cbc0010I After 50300 nodes, 15030 on
tree, 877 best solution, best possible 749.58541 (4465.61 seconds)
Cbc0010I After 50400 nodes, 15076 on tree, 877 best solution, best
possible 749.58541 (4476.21 seconds) Cbc0010I After 50500 nodes, 15122 on
tree, 877 best solution, best possible 749.58541 (4487.39 seconds)
Cbc0010I After 50600 nodes, 15166 on tree, 877 best solution, best
possible 749.58541 (4497.96 seconds) Cbc0010I After 50700 nodes, 15212 on
tree, 877 best solution, best possible 749.58541 (4508.91 seconds)
Cbc0010I After 50800 nodes, 15259 on tree, 877 best solution, best
possible 749.58541 (4520.39 seconds) Cbc0010I After 50900 nodes, 15306 on
tree, 877 best solution, best possible 749.58541 (4532.47 seconds)
Cbc0010I After 51000 nodes, 15350 on tree, 877 best solution, best
possible 749.58541 (4543.58 seconds) Cbc0010I After 51100 nodes, 15393 on
tree, 877 best solution, best possible 749.58541 (4552.54 seconds)
Cbc0010I After 51200 nodes, 15440 on tree, 877 best solution, best
possible 749.58541 (4561.98 seconds) Cbc0010I After 51300 nodes, 15489 on
tree, 877 best solution, best possible 749.58541 (4572.32 seconds)
Cbc0010I After 51400 nodes, 15525 on tree, 877 best solution, best
possible 749.58541 (4582.14 seconds) Cbc0010I After 51500 nodes, 15574 on
tree, 877 best solution, best possible 749.58541 (4593.31 seconds)
Cbc0010I After 51600 nodes, 15620 on tree, 877 best solution, best
possible 749.58541 (4604.43 seconds) Cbc0010I After 51700 nodes, 15669 on
tree, 877 best solution, best possible 749.58541 (4616.53 seconds)
Cbc0010I After 51800 nodes, 15713 on tree, 877 best solution, best
possible 749.58541 (4627.70 seconds) Cbc0010I After 51900 nodes, 15759 on
tree, 877 best solution, best possible 749.58541 (4637.97 seconds)
Cbc0010I After 52000 nodes, 15798 on tree, 877 best solution, best
possible 749.58541 (4649.50 seconds) Cbc0010I After 52100 nodes, 15791 on
tree, 877 best solution, best possible 749.58541 (4652.18 seconds)
Cbc0010I After 52200 nodes, 15789 on tree, 877 best solution, best
possible 749.58541 (4655.06 seconds) Cbc0010I After 52300 nodes, 15784 on
tree, 877 best solution, best possible 749.58541 (4657.66 seconds)
Cbc0010I After 52400 nodes, 15768 on tree, 877 best solution, best
possible 749.58541 (4660.50 seconds) Cbc0010I After 52500 nodes, 15767 on
tree, 877 best solution, best possible 749.58541 (4663.73 seconds)
Cbc0010I After 52600 nodes, 15760 on tree, 877 best solution, best
possible 749.58541 (4666.21 seconds) Cbc0010I After 52700 nodes, 15761 on
tree, 877 best solution, best possible 749.58541 (4669.31 seconds)
Cbc0010I After 52800 nodes, 15750 on tree, 877 best solution, best
possible 749.58541 (4672.09 seconds) Cbc0010I After 52900 nodes, 15747 on
tree, 877 best solution, best possible 749.58541 (4674.47 seconds)
Cbc0010I After 53000 nodes, 15742 on tree, 877 best solution, best
possible 749.58541 (4677.10 seconds) Cbc0010I After 53100 nodes, 15758 on
tree, 877 best solution, best possible 749.58541 (4681.99 seconds)
Cbc0010I After 53200 nodes, 15759 on tree, 877 best solution, best
possible 749.58541 (4685.49 seconds) Cbc0010I After 53300 nodes, 15755 on
tree, 877 best solution, best possible 749.58541 (4689.72 seconds)
Cbc0010I After 53400 nodes, 15763 on tree, 877 best solution, best
possible 749.58541 (4693.69 seconds) Cbc0010I After 53500 nodes, 15754 on
tree, 877 best solution, best possible 749.58541 (4697.89 seconds)
Cbc0010I After 53600 nodes, 15759 on tree, 877 best solution, best
possible 749.58541 (4702.55 seconds) Cbc0010I After 53700 nodes, 15754 on
tree, 877 best solution, best possible 749.58541 (4707.12 seconds)
Cbc0010I After 53800 nodes, 15760 on tree, 877 best solution, best
possible 749.58541 (4711.59 seconds) Cbc0010I After 53900 nodes, 15757 on
tree, 877 best solution, best possible 749.58541 (4716.08 seconds)
Cbc0010I After 54000 nodes, 15751 on tree, 877 best solution, best
possible 749.58541 (4720.77 seconds) Cbc0010I After 54100 nodes, 15794 on
tree, 877 best solution, best possible 749.58541 (4730.99 seconds)
Cbc0010I After 54200 nodes, 15842 on tree, 877 best solution, best
possible 749.58541 (4741.95 seconds) Cbc0010I After 54300 nodes, 15889 on
tree, 877 best solution, best possible 749.58541 (4749.30 seconds)
Cbc0010I After 54400 nodes, 15934 on tree, 877 best solution, best
possible 749.58541 (4758.42 seconds) Cbc0010I After 54500 nodes, 15978 on
tree, 877 best solution, best possible 749.58541 (4768.70 seconds)
Cbc0010I After 54600 nodes, 16017 on tree, 877 best solution, best
```

```
possible 749.58541 (4779.28 seconds) Cbc0010I After 54700 nodes, 16059 on
tree, 877 best solution, best possible 749.58541 (4789.44 seconds)
Cbc0010I After 54800 nodes, 16102 on tree, 877 best solution, best
possible 749.58541 (4801.07 seconds) Cbc0010I After 54900 nodes, 16148 on
tree, 877 best solution, best possible 749.58541 (4812.60 seconds)
Cbc0010I After 55000 nodes, 16194 on tree, 877 best solution, best
possible 749.58541 (4821.03 seconds) Cbc0010I After 55100 nodes, 16237 on
tree, 877 best solution, best possible 749.58541 (4831.55 seconds)
Cbc0010I After 55200 nodes, 16280 on tree, 877 best solution, best
possible 749.58541 (4842.14 seconds) Cbc0010I After 55300 nodes, 16326 on
tree, 877 best solution, best possible 749.58541 (4851.44 seconds)
Cbc0010I After 55400 nodes, 16370 on tree, 877 best solution, best
possible 749.58541 (4860.75 seconds) Cbc0010I After 55500 nodes, 16414 on
tree, 877 best solution, best possible 749.58541 (4870.45 seconds)
Cbc0010I After 55600 nodes, 16458 on tree, 877 best solution, best
possible 749.58541 (4882.06 seconds) Cbc0010I After 55700 nodes, 16506 on
tree, 877 best solution, best possible 749.58541 (4893.20 seconds)
Cbc0010I After 55800 nodes, 16548 on tree, 877 best solution, best
possible 749.58541 (4905.23 seconds) Cbc0010I After 55900 nodes, 16592 on
tree, 877 best solution, best possible 749.58541 (4914.58 seconds)
Cbc0010I After 56000 nodes, 16635 on tree, 877 best solution, best
possible 749.58541 (4924.57 seconds) Cbc0004I Integer solution of 875
found after 18383734 iterations and 56041 nodes (4925.94 seconds) Cbc0010I
After 56100 nodes, 16500 on tree, 875 best solution, best possible
749.58541 (4931.64 seconds) Cbc0010I After 56200 nodes, 16544 on tree, 875
best solution, best possible 749.58541 (4940.87 seconds) Cbc0010I After
56300 nodes, 16580 on tree, 875 best solution, best possible 749.58541
(4947.95 seconds) Cbc0010I After 56400 nodes, 16617 on tree, 875 best
solution, best possible 749.58541 (4954.85 seconds) Cbc0010I After 56500
nodes, 16660 on tree, 875 best solution, best possible 749.58541 (4964.29
seconds) Cbc0010I After 56600 nodes, 16703 on tree, 875 best solution,
best possible 749.58541 (4974.87 seconds) Cbc0010I After 56700 nodes,
16748 on tree, 875 best solution, best possible 749.58541 (4985.45
seconds) Cbc0010I After 56800 nodes, 16794 on tree, 875 best solution,
best possible 749.58541 (4994.24 seconds) Cbc0010I After 56900 nodes,
16835 on tree, 875 best solution, best possible 749.58541 (5003.61
seconds) Cbc0010I After 57000 nodes, 16879 on tree, 875 best solution,
best possible 749.58541 (5012.28 seconds) Cbc0010I After 57100 nodes,
16888 on tree, 875 best solution, best possible 749.58891 (5018.32
seconds) Cbc0010I After 57200 nodes, 16892 on tree, 875 best solution,
best possible 749.58891 (5024.35 seconds) Cbc0010I After 57300 nodes,
16884 on tree, 875 best solution, best possible 749.58891 (5030.04
seconds) Cbc0010I After 57400 nodes, 16881 on tree, 875 best solution,
best possible 749.58891 (5035.48 seconds) Cbc0010I After 57500 nodes,
16883 on tree, 875 best solution, best possible 749.58891 (5041.15
seconds) Cbc0010I After 57600 nodes, 16888 on tree, 875 best solution,
best possible 749.58891 (5045.87 seconds) Cbc0010I After 57700 nodes,
16883 on tree, 875 best solution, best possible 749.58891 (5050.62
seconds) Cbc0010I After 57800 nodes, 16880 on tree, 875 best solution,
best possible 749.58891 (5056.48 seconds) Cbc0010I After 57900 nodes,
16865 on tree, 875 best solution, best possible 749.58891 (5059.06
seconds) Cbc0010I After 58000 nodes, 16858 on tree, 875 best solution,
best possible 749.58891 (5061.61 seconds) Cbc0010I After 58100 nodes,
16904 on tree, 875 best solution, best possible 749.58891 (5075.12
seconds) Cbc0010I After 58200 nodes, 16950 on tree, 875 best solution,
best possible 749.58891 (5087.17 seconds) Cbc0010I After 58300 nodes,
16996 on tree, 875 best solution, best possible 749.58891 (5100.72
seconds) Cbc0010I After 58400 nodes, 17043 on tree, 875 best solution,
best possible 749.58891 (5112.85 seconds) Cbc0010I After 58500 nodes,
17090 on tree, 875 best solution, best possible 749.58891 (5123.67
seconds) Cbc0010I After 58600 nodes, 17140 on tree, 875 best solution,
best possible 749.58891 (5135.08 seconds) Cbc0010I After 58700 nodes,
17184 on tree, 875 best solution, best possible 749.58891 (5146.74
seconds) Cbc0010I After 58800 nodes, 17232 on tree, 875 best solution,
best possible 749.58891 (5160.10 seconds) Cbc0010I After 58900 nodes,
17279 on tree, 875 best solution, best possible 749.58891 (5171.58
seconds) Cbc0010I After 59000 nodes, 17323 on tree, 875 best solution,
best possible 749.58891 (5181.45 seconds) Cbc0010I After 59100 nodes,
17366 on tree, 875 best solution, best possible 749.58891 (5191.83
17366 on tree, 875 best solution, best possible 749.58891 (5191.65
seconds) Cbc0010I After 59200 nodes, 17413 on tree, 875 best solution,
best possible 749.58891 (5203.30 seconds) Cbc0010I After 59300 nodes,
17449 on tree, 875 best solution, best possible 749.58891 (5212.93
seconds) Cbc0010I After 59400 nodes, 17492 on tree, 875 best solution,
best possible 749.58891 (5224.36 seconds) Cbc0010I After 59500 nodes,
17540 on tree, 875 best solution, best possible 749.58891 (5235.82
seconds) Cbc0010I After 59600 nodes, 17584 on tree, 875 best solution,
best possible 749.58891 (5246.50 seconds) Cbc0010I After 59700 nodes,
17624 on tree, 875 best solution, best possible 749.58891 (5254.69
seconds) Cbc0010I After 59800 nodes, 17667 on tree, 875 best solution,
best possible 749.58891 (5266.18 seconds) Cbc0010I After 59900 nodes,
17714 on tree, 875 best solution, best possible 749.58891 (5277.38
seconds) Cbc0010I After 60000 nodes, 17756 on tree, 875 best solution,
best possible 749.58891 (5290.00 seconds) Cbc0010I After 60100 nodes,
17754 on tree, 875 best solution, best possible 749.58891 (5293.24
seconds) Cbc0010I After 60200 nodes, 17747 on tree, 875 best solution,
best possible 749.58891 (5296.48 seconds) Cbc0010I After 60300 nodes,
17734 on tree, 875 best solution, best possible 749.58891 (5299.60
seconds) Cbc0010I After 60400 nodes, 17727 on tree, 875 best solution,
best possible 749.58891 (5302.38 seconds) Cbc0010I After 60500 nodes,
17718 on tree, 875 best solution, best possible 749.58891 (5305.46
seconds) Cbc0010I After 60600 nodes, 17714 on tree, 875 best solution,
best possible 749.58891 (5308.11 seconds) Cbc0010I After 60700 nodes,
17710 on tree, 875 best solution, best possible 749.58891 (5311.21
seconds) Cbc0010I After 60800 nodes, 17703 on tree, 875 best solution,
best possible 749.58891 (5314.39 seconds) Cbc0010I After 60900 nodes,
17705 on tree, 875 best solution, best possible 749.58891 (5317.53
seconds) Cbc0010I After 61000 nodes, 17709 on tree, 875 best solution,
best possible 749.58891 (5320.11 seconds) Cbc0010I After 61100 nodes,
17717 on tree, 875 best solution, best possible 749.58891 (5325.80
seconds) Cbc0010I After 61200 nodes, 17722 on tree, 875 best solution,
best possible 749.58891 (5330.05 seconds) Cbc0010I After 61300 nodes,
17724 on tree, 875 best solution, best possible 749.58891 (5334.63
seconds) Cbc0010I After 61400 nodes, 17722 on tree, 875 best solution,
best possible 749.58891 (5338.17 seconds) Cbc0010I After 61500 nodes,
17720 on tree, 875 best solution, best possible 749.58891 (5342.04
seconds) Cbc0010I After 61600 nodes, 17719 on tree, 875 best solution,
best possible 749.58891 (5345.65 seconds) Cbc0010I After 61700 nodes,
17731 on tree, 875 best solution, best possible 749.58891 (5349.44
seconds) Cbc0010I After 61800 nodes, 17722 on tree, 875 best solution,
best possible 749.58891 (5352.85 seconds) Cbc0010I After 61900 nodes,
17727 on tree, 875 best solution, best possible 749.58891 (5356.62
seconds) Cbc0010I After 62000 nodes, 17721 on tree, 875 best solution,
best possible 749.58891 (5360.21 seconds) Cbc0010I After 62100 nodes,
17762 on tree, 875 best solution, best possible 749.62686 (5372.94
seconds) Cbc0010I After 62200 nodes, 17805 on tree, 875 best solution,
best possible 749.62686 (5382.87 seconds) Cbc0010I After 62300 nodes,
17844 on tree, 875 best solution, best possible 749.62686 (5392.61
seconds) Cbc0010I After 62400 nodes, 17889 on tree, 875 best solution,
best possible 749.62686 (5402.40 seconds) Cbc0010I After 62500 nodes,
17933 on tree, 875 best solution, best possible 749.62686 (5412.83
seconds) Cbc0010I After 62600 nodes, 17979 on tree, 875 best solution,
best possible 749.62686 (5424.02 seconds) Cbc0010I After 62700 nodes,
18021 on tree, 875 best solution, best possible 749.62686 (5435.61
seconds) Cbc0010I After 62800 nodes, 18062 on tree, 875 best solution,
best possible 749.62686 (5445.69 seconds) Cbc0010I After 62900 nodes,
18102 on tree, 875 best solution, best possible 749.62686 (5455.78
seconds) Cbc0010I After 63000 nodes, 18144 on tree, 875 best solution,
best possible 749.62686 (5465.91 seconds) Cbc0010I After 63100 nodes,
18188 on tree, 875 best solution, best possible 749.62686 (5478.82
seconds) Cbc0010I After 63200 nodes, 18233 on tree, 875 best solution,
best possible 749.62686 (5489.61 seconds) Cbc0010I After 63300 nodes,
18272 on tree, 875 best solution, best possible 749.62686 (5498.73
seconds) Cbc0010I After 63400 nodes, 18311 on tree, 875 best solution,
best possible 749.62686 (5508.05 seconds) Cbc0010I After 63500 nodes,
18353 on tree, 875 best solution, best possible 749.62686 (5518.96
seconds) Cbc0010I After 63600 nodes, 18398 on tree, 875 best solution,
best possible 749.62686 (5527.29 seconds) Cbc0010I After 63700 nodes,
18444 on tree, 875 best solution, best possible 749.62686 (5537.30
seconds) Cbc0010I After 63800 nodes, 18483 on tree, 875 best solution,
best possible 749.62686 (5548.20 seconds) Cbc0010I After 63900 nodes,
18526 on tree, 875 best solution, best possible 749.62686 (5559.13
seconds) Cbc0010I After 64000 nodes, 18569 on tree, 875 best solution,
best possible 749.62686 (5570.21 seconds) Cbc0010I After 64100 nodes,
18568 on tree, 875 best solution, best possible 749.62686 (5572.49
seconds) Cbc0010I After 64200 nodes, 18564 on tree, 875 best solution,
best possible 749.62686 (5574.57 seconds) Cbc0010I After 64300 nodes,
18558 on tree, 875 best solution, best possible 749.62686 (5577.24
seconds) Cbc0012I Integer solution of 872 found by rounding after 20954446
iterations and 64393 nodes (5579.76 seconds) Cbc0010I After 64400 nodes,
18424 on tree, 872 best solution, best possible 749.62686 (5580.94
seconds) Cbc0010I After 64500 nodes, 18460 on tree, 872 best solution,
best possible 749.62686 (5588.32 seconds) Cbc0010I After 64600 nodes,
18492 on tree, 872 best solution, best possible 749.62686 (5594.51
seconds) Cbc0010I After 64700 nodes, 18533 on tree, 872 best solution,
best possible 749.62686 (5603.63 seconds) Cbc0038I Full problem 2890 rows
1451 columns, reduced to 1762 rows 707 columns - 1 fixed gives 1759, 705 -
still too large Cbc0010I After 64800 nodes, 18572 on tree, 872 best
solution, best possible 749.62686 (5612.80 seconds) Cbc0010I After 64900
nodes, 18614 on tree, 872 best solution, best possible 749.62686 (5621.86
seconds) Cbc0010I After 65000 nodes, 18648 on tree, 872 best solution,
best possible 749.62686 (5631.52 seconds) Cbc0010I After 65100 nodes,
18664 on tree, 872 best solution, best possible 749.62686 (5639.07
seconds) Cbc0010I After 65200 nodes, 18657 on tree, 872 best solution,
best possible 749.62686 (5643.54 seconds) Cbc0010I After 65300 nodes,
18666 on tree, 872 best solution, best possible 749.62686 (5648.88
seconds) Cbc0010I After 65400 nodes, 18663 on tree, 872 best solution,
best possible 749.62686 (5653.82 seconds) Cbc0010I After 65500 nodes,
18658 on tree, 872 best solution, best possible 749.62686 (5658.41
seconds) Cbc0010I After 65600 nodes, 18664 on tree, 872 best solution,
best possible 749.62686 (5662.65 seconds) Cbc0010I After 65700 nodes,
18657 on tree, 872 best solution, best possible 749.62686 (5667.07
seconds) Cbc0010I After 65800 nodes, 18662 on tree, 872 best solution,
best possible 749.62686 (5671.48 seconds) Cbc0010I After 65900 nodes,
18662 on tree, 872 best solution, best possible 749.62686 (5675.86
seconds) Cbc0010I After 66000 nodes, 18662 on tree, 872 best solution,
best possible 749.62686 (5678.86 seconds) Cbc0010I After 66100 nodes,
18708 on tree, 872 best solution, best possible 749.62686 (5692.09
seconds) Cbc0010I After 66200 nodes, 18754 on tree, 872 best solution,
best possible 749.62686 (5701.04 seconds) Cbc0010I After 66300 nodes,
18800 on tree, 872 best solution, best possible 749.62686 (5712.02
seconds) Cbc0010I After 66400 nodes, 18846 on tree, 872 best solution,
best possible 749.62686 (5725.19 seconds) Cbc0010I After 66500 nodes,
18894 on tree, 872 best solution, best possible 749.62686 (5736.42
seconds) Cbc0010I After 66600 nodes, 18939 on tree, 872 best solution,
best possible 749.62686 (5747.79 seconds) Cbc0010I After 66700 nodes,
18984 on tree, 872 best solution, best possible 749.62686 (5760.36
seconds) Cbc0010I After 66800 nodes, 19031 on tree, 872 best solution,
best possible 749.62686 (5771.37 seconds) Cbc0010I After 66900 nodes,
19079 on tree, 872 best solution, best possible 749.62686 (5782.29
seconds) Cbc0010I After 67000 nodes, 19125 on tree, 872 best solution,
best possible 749.62686 (5793.36 seconds) Cbc0010I After 67100 nodes,
19171 on tree, 872 best solution, best possible 749.62686 (5805.79
seconds) Cbc0010I After 67200 nodes, 19215 on tree, 872 best solution,
best possible 749.62686 (5817.07 seconds) Cbc0010I After 67300 nodes,
19264 on tree, 872 best solution, best possible 749.62686 (5829.20
seconds) Cbc0010I After 67400 nodes, 19306 on tree, 872 best solution,
best possible 749.62686 (5841.42 seconds) Cbc0010I After 67500 nodes,
19352 on tree, 872 best solution, best possible 749.62686 (5853.67
seconds) Cbc0010I After 67600 nodes, 19397 on tree, 872 best solution,
best possible 749.62686 (5864.97 seconds) Cbc0010I After 67700 nodes,
19441 on tree, 872 best solution, best possible 749.62686 (5875.73
seconds) Cbc0010I After 67800 nodes, 19484 on tree, 872 best solution,
best possible 749.62686 (5883.88 seconds) Cbc0010I After 67900 nodes,
19531 on tree, 872 best solution, best possible 749.62686 (5892.81
seconds) Cbc0010I After 68000 nodes, 19573 on tree, 872 best solution,
best possible 749.62686 (5905.03 seconds) Cbc0010I After 68100 nodes,
best possible 749.62686 (5905.03 seconds) Cbc0010I After 68100 nodes,
19561 on tree, 872 best solution, best possible 749.62686 (5907.70
seconds) Cbc0010I After 68200 nodes, 19558 on tree, 872 best solution,
best possible 749.62686 (5910.96 seconds) Cbc0010I After 68300 nodes,
19556 on tree, 872 best solution, best possible 749.62686 (5914.06
seconds) Cbc0010I After 68400 nodes, 19551 on tree, 872 best solution,
best possible 749.62686 (5916.66 seconds) Cbc0010I After 68500 nodes,
19549 on tree, 872 best solution, best possible 749.62686 (5919.99
seconds) Cbc0010I After 68600 nodes, 19543 on tree, 872 best solution,
best possible 749.62686 (5922.82 seconds) Cbc0010I After 68700 nodes,
19540 on tree, 872 best solution, best possible 749.62686 (5925.51
seconds) Cbc0010I After 68800 nodes, 19527 on tree, 872 best solution,
best possible 749.62686 (5928.41 seconds) Cbc0010I After 68900 nodes,
19527 on tree, 872 best solution, best possible 749.62686 (5931.62
seconds) Cbc0010I After 69000 nodes, 19523 on tree, 872 best solution,
best possible 749.62686 (5934.00 seconds) Cbc0010I After 69100 nodes,
19532 on tree, 872 best solution, best possible 749.62893 (5942.49
seconds) Cbc0010I After 69200 nodes, 19532 on tree, 872 best solution,
best possible 749.62893 (5947.31 seconds) Cbc0010I After 69300 nodes,
19532 on tree, 872 best solution, best possible 749.62893 (5952.67
seconds) Cbc0010I After 69400 nodes, 19531 on tree, 872 best solution,
best possible 749.62893 (5958.06 seconds) Cbc0010I After 69500 nodes,
19533 on tree, 872 best solution, best possible 749.62893 (5962.65
seconds) Cbc0010I After 69600 nodes, 19534 on tree, 872 best solution,
best possible 749.62893 (5968.12 seconds) Cbc0010I After 69700 nodes,
19532 on tree, 872 best solution, best possible 749.62893 (5972.97
seconds) Cbc0010I After 69800 nodes, 19530 on tree, 872 best solution,
best possible 749.62893 (5978.25 seconds) Cbc0010I After 69900 nodes,
19533 on tree, 872 best solution, best possible 749.62893 (5984.85
seconds) Cbc0010I After 70000 nodes, 19534 on tree, 872 best solution,
best possible 749.62893 (5990.54 seconds) Cbc0010I After 70100 nodes,
19574 on tree, 872 best solution, best possible 749.62893 (6001.42
seconds) Cbc0010I After 70200 nodes, 19622 on tree, 872 best solution,
best possible 749.62893 (6012.35 seconds) Cbc0010I After 70300 nodes,
19666 on tree, 872 best solution, best possible 749.62893 (6024.42
seconds) Cbc0010I After 70400 nodes, 19707 on tree, 872 best solution,
best possible 749.62893 (6034.55 seconds) Cbc0010I After 70500 nodes,
19753 on tree, 872 best solution, best possible 749.62893 (6044.82
seconds) Cbc0010I After 70600 nodes, 19799 on tree, 872 best solution,
best possible 749.62893 (6058.24 seconds) Cbc0010I After 70700 nodes,
19848 on tree, 872 best solution, best possible 749.62893 (6071.73
seconds) Cbc0010I After 70800 nodes, 19890 on tree, 872 best solution,
best possible 749.62893 (6081.86 seconds) Cbc0010I After 70900 nodes,
19934 on tree, 872 best solution, best possible 749.62893 (6091.38
seconds) Cbc0010I After 71000 nodes, 19974 on tree, 872 best solution,
best possible 749.62893 (6101.67 seconds) Cbc0010I After 71100 nodes,
20016 on tree, 872 best solution, best possible 749.62893 (6113.36
seconds) Cbc0010I After 71200 nodes, 20059 on tree, 872 best solution,
best possible 749.62893 (6122.59 seconds) Cbc0010I After 71300 nodes,
20100 on tree, 872 best solution, best possible 749.62893 (6132.14
seconds) Cbc0010I After 71400 nodes, 20147 on tree, 872 best solution,
best possible 749.62893 (6141.35 seconds) Cbc0010I After 71500 nodes,
20180 on tree, 872 best solution, best possible 749.62893 (6149.11
seconds) Cbc0010I After 71600 nodes, 20224 on tree, 872 best solution,
best possible 749.62893 (6158.50 seconds) Cbc0010I After 71700 nodes,
20267 on tree, 872 best solution, best possible 749.62893 (6169.64
seconds) Cbc0010I After 71800 nodes, 20313 on tree, 872 best solution,
best possible 749.62893 (6180.27 seconds) Cbc0010I After 71900 nodes,
20356 on tree, 872 best solution, best possible 749.62893 (6190.17
seconds) Cbc0010I After 72000 nodes, 20395 on tree, 872 best solution,
best possible 749.62893 (6200.47 seconds) Cbc0010I After 72100 nodes,
20386 on tree, 872 best solution, best possible 749.62893 (6203.26
seconds) Cbc0010I After 72200 nodes, 20378 on tree, 872 best solution,
best possible 749.62893 (6206.41 seconds) Cbc0010I After 72300 nodes,
20370 on tree, 872 best solution, best possible 749.62893 (6209.30
seconds) Cbc0010I After 72400 nodes, 20374 on tree, 872 best solution,
best possible 749.62893 (6212.00 seconds) Cbc0010I After 72500 nodes,
20363 on tree, 872 best solution, best possible 749.62893 (6214.67
seconds) Cbc0010I After 72600 nodes, 20353 on tree, 872 best solution,
best possible 749.62893 (6217.75 seconds) Cbc0010I After 72700 nodes,
20344 on tree, 872 best solution, best possible 749.62893 (6220.77
seconds) Cbc0010I After 72800 nodes, 20347 on tree, 872 best solution,
best possible 749.62893 (6223.93 seconds) Cbc0010I After 72900 nodes,
20347 on tree, 872 best solution, best possible 749.62893 (6227.14
seconds) Cbc0010I After 73000 nodes, 20341 on tree, 872 best solution,
best possible 749.62893 (6229.92 seconds) Cbc0010I After 73100 nodes,
20359 on tree, 872 best solution, best possible 749.62893 (6234.67
seconds) Cbc0010I After 73200 nodes, 20359 on tree, 872 best solution,
best possible 749.62893 (6238.17 seconds) Cbc0010I After 73300 nodes,
20352 on tree, 872 best solution, best possible 749.62893 (6242.46
seconds) Cbc0010I After 73400 nodes, 20352 on tree, 872 best solution,
best possible 749.62893 (6246.78 seconds) Cbc0010I After 73500 nodes,
20349 on tree, 872 best solution, best possible 749.62893 (6250.88
seconds) Cbc0010I After 73600 nodes, 20356 on tree, 872 best solution,
best possible 749.62893 (6254.08 seconds) Cbc0010I After 73700 nodes,
20358 on tree, 872 best solution, best possible 749.62893 (6257.64
seconds) Cbc0010I After 73800 nodes, 20357 on tree, 872 best solution,
best possible 749.62893 (6261.48 seconds) Cbc0010I After 73900 nodes,
20363 on tree, 872 best solution, best possible 749.62893 (6265.59
seconds) Cbc0010I After 74000 nodes, 20364 on tree, 872 best solution,
best possible 749.62893 (6268.55 seconds) Cbc0010I After 74100 nodes,
20407 on tree, 872 best solution, best possible 749.62893 (6280.98
seconds) Cbc0010I After 74200 nodes, 20447 on tree, 872 best solution,
best possible 749.62893 (6290.71 seconds) Cbc0010I After 74300 nodes,
20489 on tree, 872 best solution, best possible 749.62893 (6300.27
seconds) Cbc0010I After 74400 nodes, 20534 on tree, 872 best solution,
best possible 749.62893 (6311.80 seconds) Cbc0010I After 74500 nodes,
20574 on tree, 872 best solution, best possible 749.62893 (6322.35
seconds) Cbc0010I After 74600 nodes, 20614 on tree, 872 best solution,
best possible 749.62893 (6332.54 seconds) Cbc0010I After 74700 nodes,
20656 on tree, 872 best solution, best possible 749.62893 (6342.63
seconds) Cbc0010I After 74800 nodes, 20702 on tree, 872 best solution,
best possible 749.62893 (6353.84 seconds) Cbc0010I After 74900 nodes,
20746 on tree, 872 best solution, best possible 749.62893 (6363.31
seconds) Cbc0010I After 75000 nodes, 20781 on tree, 872 best solution,
best possible 749.62893 (6371.71 seconds) Cbc0010I After 75100 nodes,
20827 on tree, 872 best solution, best possible 749.62893 (6384.42
seconds) Cbc0010I After 75200 nodes, 20867 on tree, 872 best solution,
best possible 749.62893 (6394.62 seconds) Cbc0010I After 75300 nodes,
20906 on tree, 872 best solution, best possible 749.62893 (6404.39
seconds) Cbc0010I After 75400 nodes, 20949 on tree, 872 best solution,
best possible 749.62893 (6415.95 seconds) Cbc0010I After 75500 nodes,
20991 on tree, 872 best solution, best possible 749.62893 (6425.46
seconds) Cbc0010I After 75600 nodes, 21037 on tree, 872 best solution,
best possible 749.62893 (6437.76 seconds) Cbc0010I After 75700 nodes,
21077 on tree, 872 best solution, best possible 749.62893 (6449.43
seconds) Cbc0010I After 75800 nodes, 21123 on tree, 872 best solution,
best possible 749.62893 (6458.35 seconds) Cbc0010I After 75900 nodes,
21164 on tree, 872 best solution, best possible 749.62893 (6468.15
seconds) Cbc0010I After 76000 nodes, 21207 on tree, 872 best solution,
best possible 749.62893 (6475.11 seconds) Cbc0010I After 76100 nodes,
21208 on tree, 872 best solution, best possible 749.62893 (6479.17
seconds) Cbc0010I After 76200 nodes, 21215 on tree, 872 best solution,
best possible 749.62893 (6483.07 seconds) Cbc0010I After 76300 nodes,
21209 on tree, 872 best solution, best possible 749.62893 (6486.12
seconds) Cbc0010I After 76400 nodes, 21210 on tree, 872 best solution,
best possible 749.62893 (6488.66 seconds) Cbc0010I After 76500 nodes,
21203 on tree, 872 best solution, best possible 749.62893 (6491.68
seconds) Cbc0010I After 76600 nodes, 21204 on tree, 872 best solution,
best possible 749.62893 (6494.87 seconds) Cbc0010I After 76700 nodes,
21192 on tree, 872 best solution, best possible 749.62893 (6497.59
seconds) Cbc0010I After 76800 nodes, 21192 on tree, 872 best solution,
best possible 749.62893 (6500.64 seconds) Cbc0010I After 76900 nodes,
21191 on tree, 872 best solution, best possible 749.62893 (6502.82
seconds) Cbc0010I After 77000 nodes, 21185 on tree, 872 best solution,
best possible 749.62893 (6505.93 seconds) Cbc0010I After 77100 nodes,
21192 on tree, 872 best solution, best possible 749.6553 (6512.37 seconds)
```

```
Cbc0010I After 77200 nodes, 21191 on tree, 872 best solution, best
possible 749.6553 (6517.76 seconds) Cbc0010I After 77300 nodes, 21191 on
tree, 872 best solution, best possible 749.6553 (6523.31 seconds) Cbc0010I
After 77400 nodes, 21196 on tree, 872 best solution, best possible
749.6553 (6527.61 seconds) Cbc0010I After 77500 nodes, 21199 on tree, 872
best solution, best possible 749.6553 (6530.69 seconds) Cbc0010I After
77600 nodes, 21196 on tree, 872 best solution, best possible 749.6553
(6533.53 seconds) Cbc0010I After 77700 nodes, 21193 on tree, 872 best
solution, best possible 749.6553 (6536.99 seconds) Cbc0010I After 77800
nodes, 21199 on tree, 872 best solution, best possible 749.6553 (6542.33
seconds) Cbc0010I After 77900 nodes, 21197 on tree, 872 best solution,
best possible 749.6553 (6546.42 seconds) Cbc0010I After 78000 nodes, 21191
on tree, 872 best solution, best possible 749.6553 (6550.92 seconds)
Cbc0010I After 78100 nodes, 21228 on tree, 872 best solution, best
possible 749.6553 (6560.61 seconds) Cbc0010I After 78200 nodes, 21271 on
tree, 872 best solution, best possible 749.6553 (6571.76 seconds) Cbc0010I
After 78300 nodes, 21314 on tree, 872 best solution, best possible
749.6553 (6582.09 seconds) Cbc0010I After 78400 nodes, 21356 on tree, 872
best solution, best possible 749.6553 (6592.72 seconds) Cbc0010I After
78500 nodes, 21397 on tree, 872 best solution, best possible 749.6553
(6601.92 seconds) Cbc0010I After 78600 nodes, 21439 on tree, 872 best
solution, best possible 749.6553 (6611.17 seconds) Cbc0010I After 78700
nodes, 21481 on tree, 872 best solution, best possible 749.6553 (6619.45
seconds) Cbc0010I After 78800 nodes, 21525 on tree, 872 best solution,
best possible 749.6553 (6627.80 seconds) Cbc0010I After 78900 nodes, 21567
on tree, 872 best solution, best possible 749.6553 (6638.92 seconds)
Cbc0010I After 79000 nodes, 21605 on tree, 872 best solution, best
possible 749.6553 (6648.49 seconds) Cbc0010I After 79100 nodes, 21646 on
tree, 872 best solution, best possible 749.6553 (6657.51 seconds) Cbc0010I
After 79200 nodes, 21689 on tree, 872 best solution, best possible
749.6553 (6666.44 seconds) Cbc0010I After 79300 nodes, 21734 on tree, 872
best solution, best possible 749.6553 (6677.15 seconds) Cbc0010I After
79400 nodes, 21776 on tree, 872 best solution, best possible 749.6553
(6687.32 seconds) Cbc0010I After 79500 nodes, 21817 on tree, 872 best
solution, best possible 749.6553 (6697.31 seconds) Cbc0010I After 79600
nodes, 21858 on tree, 872 best solution, best possible 749.6553 (6707.95
seconds) Cbc0010I After 79700 nodes, 21902 on tree, 872 best solution,
best possible 749.6553 (6718.54 seconds) Cbc0010I After 79800 nodes, 21946
on tree, 872 best solution, best possible 749.6553 (6729.18 seconds)
Cbc0010I After 79900 nodes, 21992 on tree, 872 best solution, best
possible 749.6553 (6738.41 seconds) Cbc0010I After 80000 nodes, 22038 on
tree, 872 best solution, best possible 749.6553 (6748.60 seconds) Cbc0010I
After 80100 nodes, 22040 on tree, 872 best solution, best possible
749.6553 (6751.44 seconds) Cbc0010I After 80200 nodes, 22032 on tree, 872
best solution, best possible 749.6553 (6754.74 seconds) Cbc0010I After
80300 nodes, 22025 on tree, 872 best solution, best possible 749.6553
(6757.98 seconds) Cbc0010I After 80400 nodes, 22026 on tree, 872 best
solution, best possible 749.6553 (6761.72 seconds) Cbc0010I After 80500
nodes, 22018 on tree, 872 best solution, best possible 749.6553 (6764.30
seconds) Cbc0010I After 80600 nodes, 22009 on tree, 872 best solution,
best possible 749.6553 (6767.49 seconds) Cbc0010I After 80700 nodes, 22002
on tree, 872 best solution, best possible 749.6553 (6770.73 seconds)
Cbc0010I After 80800 nodes, 22007 on tree, 872 best solution, best
possible 749.6553 (6773.82 seconds) Cbc0010I After 80900 nodes, 21998 on
tree, 872 best solution, best possible 749.6553 (6777.48 seconds) Cbc0010I
After 81000 nodes, 22005 on tree, 872 best solution, best possible
749.6553 (6780.85 seconds) Cbc0010I After 81100 nodes, 22022 on tree, 872
best solution, best possible 749.6553 (6786.00 seconds) Cbc0010I After
81200 nodes, 22018 on tree, 872 best solution, best possible 749.6553
(6789.73 seconds) Cbc0010I After 81300 nodes, 22020 on tree, 872 best
solution, best possible 749.6553 (6794.00 seconds) Cbc0010I After 81400
nodes, 22028 on tree, 872 best solution, best possible 749.6553 (6798.75
seconds) Cbc0010I After 81500 nodes, 22022 on tree, 872 best solution,
best possible 749.6553 (6803.30 seconds) Cbc0010I After 81600 nodes, 22023
on tree, 872 best solution, best possible 749.6553 (6807.11 seconds)
Cbc0010I After 81700 nodes, 22025 on tree, 872 best solution, best
possible 749.6553 (6809.88 seconds) Cbc0010I After 81800 nodes, 22026 on
tree, 872 best solution, best possible 749.6553 (6813.29 seconds) Cbc0010I
After 81900 nodes, 22019 on tree, 872 best solution, best possible
```

```
    749.6553 (6817.96 seconds) Cbc0010I After 82000 nodes, 22021 on tree, 872
    best solution, best possible 749.6553 (6821.38 seconds) Cbc0010I After
    82100 nodes, 22062 on tree, 872 best solution, best possible 749.6553
    (6833.30 seconds) Cbc0010I After 82200 nodes, 22103 on tree, 872 best
    solution, best possible 749.6553 (6843.10 seconds) Cbc0010I After 82300
    nodes, 22149 on tree, 872 best solution, best possible 749.6553 (6852.36
    seconds) Cbc0010I After 82400 nodes, 22193 on tree, 872 best solution,
    best possible 749.6553 (6863.33 seconds) Cbc0010I After 82500 nodes, 22238
    on tree, 872 best solution, best possible 749.6553 (6873.51 seconds)
    Cbc0010I After 82600 nodes, 22281 on tree, 872 best solution, best
    possible 749.6553 (6882.52 seconds) Cbc0010I After 82700 nodes, 22322 on
    tree, 872 best solution, best possible 749.6553 (6893.36 seconds) Cbc0010I
    After 82800 nodes, 22358 on tree, 872 best solution, best possible
    749.6553 (6903.55 seconds) Cbc0010I After 82900 nodes, 22405 on tree, 872
    best solution, best possible 749.6553 (6913.78 seconds) Cbc0010I After
    83000 nodes, 22451 on tree, 872 best solution, best possible 749.6553
    (6924.79 seconds) Cbc0010I After 83100 nodes, 22490 on tree, 872 best
    solution, best possible 749.6553 (6935.14 seconds) Cbc0010I After 83200
    nodes, 22530 on tree, 872 best solution, best possible 749.6553 (6945.03
    seconds) Cbc0010I After 83300 nodes, 22574 on tree, 872 best solution,
    best possible 749.6553 (6954.64 seconds) Cbc0010I After 83400 nodes, 22614
    on tree, 872 best solution, best possible 749.6553 (6963.72 seconds)
    Cbc0010I After 83500 nodes, 22656 on tree, 872 best solution, best
    possible 749.6553 (6972.70 seconds) Cbc0010I After 83600 nodes, 22701 on
    tree, 872 best solution, best possible 749.6553 (6983.43 seconds) Cbc0010I
    After 83700 nodes, 22747 on tree, 872 best solution, best possible
    749.6553 (6992.03 seconds) Cbc0010I After 83800 nodes, 22787 on tree, 872
    best solution, best possible 749.6553 (7002.54 seconds) Cbc0010I After
    83900 nodes, 22834 on tree, 872 best solution, best possible 749.6553
    (7014.11 seconds) Cbc0010I After 84000 nodes, 22879 on tree, 872 best
    solution, best possible 749.6553 (7024.53 seconds) Cbc0010I After 84100
    nodes, 22880 on tree, 872 best solution, best possible 749.6553 (7027.31
    seconds) Cbc0010I After 84200 nodes, 22875 on tree, 872 best solution,
    best possible 749.6553 (7030.38 seconds) Cbc0010I After 84300 nodes, 22869
    on tree, 872 best solution, best possible 749.6553 (7032.94 seconds)
    Cbc0010I After 84400 nodes, 22867 on tree, 872 best solution, best
    possible 749.6553 (7036.96 seconds)
```

```
---------------------------------------------------------------------------
ApplicationError                          Traceback (most recent call last)
<ipython-input-30-023d2fee3b0f> in <module>()
     30     job += 1
     31
---> 32 Visualize(JobShop(TASKS))

<ipython-input-29-8a3da2f5248d> in JobShop(TASKS, tclean, ZW)
     33
     34     TransformationFactory('gdp.chull').apply_to(model)
---> 35     solver.solve(model)
     36
     37     results = [{'Job': j,

/usr/local/lib/python3.6/dist-packages/pyomo/opt/base/solvers.py in solve(self, *args,
**kwds)
    624                     logger.error("Solver log:\n" + str(_status.log))
    625                 raise pyutilib.common.ApplicationError(
--> 626                     "Solver (%s) did not exit normally" % self.name)
    627             solve_completion_time = time.time()
    628             if self._report_timing:

ApplicationError: Solver (cbc) did not exit normally
```

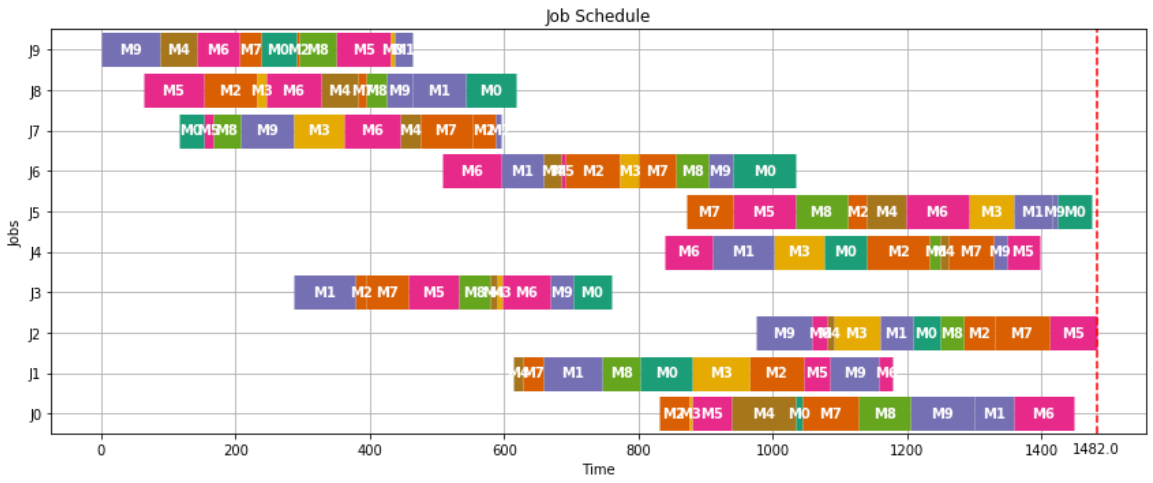
Recalculate Benchmark Problem with a Zero-Wait Policy

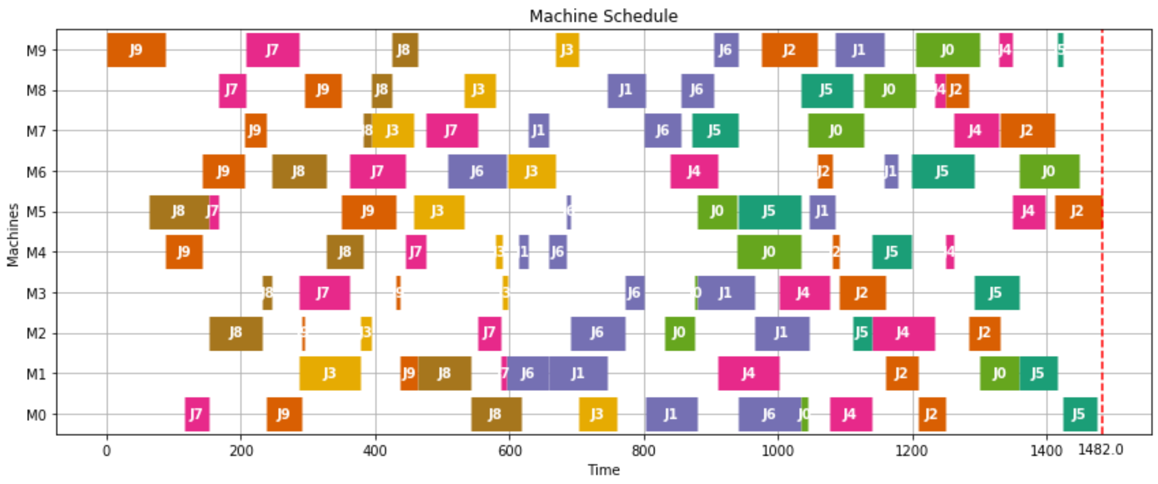
The following calculation is quite intensive and will take several minutes to finish with the `gurobi` solver.

In [ ]:

```
Visualize(JobShop(TASKS, ZW=True))
```

```
Academic license - for non-commercial use only
Optimize a model with 4241 rows, 2802 columns and 10281 nonzeros
Variable types: 1902 continuous, 900 integer (900 binary)
Coefficient statistics:
  Matrix range     [1e+00, 5e+03]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 5e+03]
  RHS range        [1e+00, 1e+02]
Presolve removed 1531 rows and 1441 columns
Presolve time: 0.02s
Presolved: 2710 rows, 1361 columns, 8120 nonzeros
Variable types: 911 continuous, 450 integer (450 binary)

Root relaxation: objective 6.200000e+02, 485 iterations, 0.01 seconds

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0  620.00000    0   64          -  620.00000      -     -    0s
     0     0  640.56756    0   87          -  640.56756      -     -    0s
     0     0  650.03766    0   94          -  650.03766      -     -    0s
     0     0  650.03766    0   88          -  650.03766      -     -    0s
     0     0  669.53768    0  108          -  669.53768      -     -    0s
     0     0  670.11736    0   91          -  670.11736      -     -    0s
     0     0  670.11736    0   92          -  670.11736      -     -    0s
     0     0  671.34649    0   82          -  671.34649      -     -    0s
     0     0  671.34649    0  110          -  671.34649      -     -    0s
     0     0  671.55468    0  101          -  671.55468      -     -    0s
     0     0  671.60312    0   97          -  671.60312      -     -    0s
     0     0  671.60312    0   98          -  671.60312      -     -    0s
     0     0  671.60312    0  109          -  671.60312      -     -    0s
     0     0  671.60312    0   91          -  671.60312      -     -    0s
     0     0  671.60312    0  102          -  671.60312      -     -    0s
     0     0  671.60312    0   81          -  671.60312      -     -    0s
     0     0  671.60312    0   99          -  671.60312      -     -    0s
     0     0  671.60312    0   99          -  671.60312      -     -    0s
     0     2  671.60312    0   91          -  671.60312      -     -    1s
*  980   641             243    2284.0000000  707.00000  69.0%  38.0    1s
  1092   696 1185.00000  151   64 2284.00000  707.00000  69.0%  37.5    5s
  1117   713  917.00000   76   97 2284.00000  714.59422  68.7%  36.7   10s
  1162   743  904.00000   16  105 2284.00000  837.00000  63.4%  53.5   15s
* 1271   758              29    2258.0000000  837.00000  62.9%  59.3   17s
* 2014   729              80    2174.0000000  837.00000  61.5%  47.7   18s
* 2017   696              80    2155.0000000  837.00000  61.2%  47.7   18s
* 2254   704              77    2131.0000000  837.00000  60.7%  48.2   18s
* 2992   897              78    2099.0000000  837.00000  60.1%  45.0   19s
  3691  1280 1229.00000   24  112 2099.00000  837.00000  60.1%  42.9   20s
* 5173  2022              49    1952.0000000  837.00000  57.1%  41.8   21s
* 6455  2683              56    1945.0000000  915.00000  53.0%  41.1   22s
  8670  4050 1248.00000   22  104 1945.00000  951.00000  51.1%  40.6   25s
* 9764  4687              65    1940.0000000  966.00000  50.2%  42.0   26s
* 9770  4668              69    1931.0000000  966.00000  50.0%  42.0   26s
*10409  5092              53    1922.0000000  966.00000  49.7%  41.5   27s
*10596  5096              68    1897.0000000  966.00000  49.1%  41.4   27s
*12115  6015              84    1874.0000000  966.00000  48.5%  41.5   29s
 13356  6794 1048.00000   19   99 1874.00000  985.00000  47.4%  40.8   30s
 13381  6807 1212.00000   22  102 1874.00000  985.00000  47.4%  41.0   35s
*13394  6679              43    1829.0000000  985.00000  46.1%  41.0   36s
*13433  6564              46    1799.0000000  985.00000  45.2%  41.2   36s
*13484  6572              28    1798.0000000  985.00000  45.2%  41.5   37s
*13518  6098              34    1724.0000000  985.00000  42.9%  41.7   38s
*13573  5468              42    1643.0000000 1002.00000  39.0%  42.0   38s
 13651  5470 1389.00000   29   46 1643.00000 1007.00000  38.7%  42.9   40s
 16123  6399 1332.00000   26   61 1643.00000 1031.00000  37.2%  45.5   46s
 17506  6866 1580.00000   28   78 1643.00000 1046.00000  36.3%  47.1   50s
 17506  6866 1580.00000   28   78 1643.00000 1046.00000  36.3%  47.1   50s
 21928  8374 infeasible   36      1643.00000 1084.00000  34.0%  50.4   55s
*23405  8655              44    1641.0000000 1094.12976  33.3%  51.8   57s
 25236  9220 1368.00000   23   79 1641.00000 1106.00000  32.6%  53.0   60s
*28702  9838              27    1621.0000000 1134.00000  30.0%  54.4   64s
 28752  9768 1435.00000   30   38 1621.00000 1134.00000  30.0%  54.8   65s
 29759 10079 1311.00000   61   99 1621.00000 1134.00000  30.0%  55.0   81s
 29764 10082 1564.00000   50  143 1621.00000 1134.00000  30.0%  55.0   85s
 29773 10088 1605.00000   50  157 1621.00000 1134.00000  30.0%  54.9   90s
 29781 10094 1189.23008   22  170 1621.00000 1134.00000  30.0%  54.9   95s
 29789 10099 1489.00000   48  179 1621.00000 1134.00000  30.0%  54.9  100s
 29799 10106 1526.00000   38  204 1621.00000 1134.00000  30.0%  54.9  105s
 29804 10109 1219.00000   27  140 1621.00000 1134.00000  30.0%  54.9  110s
 29812 10114 1588.00000   42  131 1621.00000 1134.00000  30.0%  54.9  115s
 29819 10119 1493.00000   43  169 1621.00000 1134.00000  30.0%  54.9  120s
 29825 10123 1265.00000   39  184 1621.00000 1134.00000  30.0%  54.8  125s
 29831 10127 1255.00000   30  181 1621.00000 1134.00000  30.0%  54.8  130s
 29839 10137 1134.00000   26  144 1621.00000 1134.00000  30.0%  57.0  135s
 29946 10170 1134.00000   34  124 1621.00000 1134.00000  30.0%  57.4  142s
 30048 10196 1134.00000   39  113 1621.00000 1134.00000  30.0%  57.6  145s
 30266 10206 1163.00000   52   74 1621.00000 1134.00000  30.0%  57.9  155s
 34550 10713 1134.00000   47   87 1621.00000 1134.00000  30.0%  55.7  160s
 38097 10754 1219.00000   38  107 1621.00000 1134.00000  30.0%  54.6  165s
*41661  9683              69    1497.0000000 1134.00000  24.2%  53.9  168s
 42973  9420 1251.00000   45  126 1497.00000 1134.00000  24.2%  53.9  170s
 46829  8371 infeasible   44      1497.00000 1163.00000  22.3%  54.1  175s
 50556  6656 infeasible   52      1497.00000 1237.00000  17.4%  54.8  180s
*52349  4939              40    1491.0000000 1280.00000  14.2%  55.1  182s
 53561  3617 infeasible   43      1491.00000 1325.00000  11.1%  55.6  185s
*53924  2654              32    1482.0000000 1343.00000  9.38%  55.8  185s

Cutting planes:
  Gomory: 26
  Cover: 3
  Implied bound: 144
  Projected implied bound: 4
  Clique: 1
  MIR: 12
  StrongCG: 1
  Flow cover: 111
  Inf proof: 111

Explored 56061 nodes (3129028 simplex iterations) in 188.12 seconds
Thread count was 8 (of 8 available processors)

Solution count 10: 1482 1491 1497 ... 1829

Optimal solution found (tolerance 1.00e-04)
Best objective 1.482000000000e+03, best bound 1.482000000000e+03, gap 0.0000%
Freed default Gurobi environment
```


Job Schedule
Jobs
Time
1482.0


Machine Schedule
Machines
Time
1482.0

```
In [ ]:
```