2.1 Production Models with Linear Constraints

This notebook demonstrates the use of linear programming to maximize profit for a simple model of a multiproduct production facility. The notebook uses Pyomo to represent the model with the glpk solver to calculate solutions.

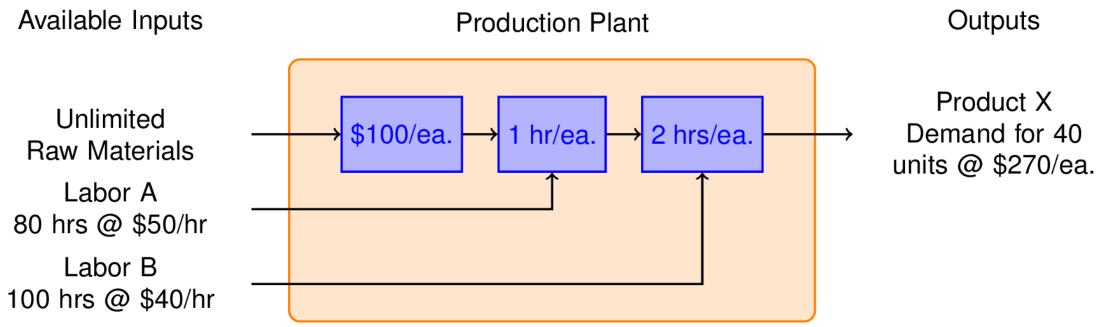
In [ ]:

```
%%capture
!pip install -q pyomo
!apt-get install -y -qq coinor-cbc
```

Example: Production Plan for a Single Product Plant

Suppose you are thinking about starting up a business to produce Product X. You have determined there is a market for X of up to 40 units per week at a price of USD 270 each. The production of each unit requires USD 100 of raw materials, 1 hour of type A labor, and 2 hours of type B labor. You have an unlimited amount of raw material available to you, but only 80 hours per week of labor A at a cost of USD 50/hour, and 100 hours per week of labor B at a cost of USD 40 per hour. Ignoring all other expenses, what is the maximum weekly profit?

To get started on this problem, we sketch a flow diagram illustrating the flow of raw materials and labor through the production plant.



The essential decision we need to make is how many units or Product X to produce each week. That's our *decision variable* which we denote as x. The weekly revenues are then

$$\text{Revenue} = \$270x$$

The costs include the value of the raw materials and each form of labor. If we produce x units a week, then the total cost is

$$\text{Cost} = \underbrace{\$100x}_{\text{Raw Material}} + \underbrace{\$50x}_{\text{Labor A}} + \underbrace{2 \times \$40x}_{\text{Labor B}} = \$230x$$

We see immediately that the gross profit is just

$$\begin{aligned} \text{Profit} &= \text{Revenue} - \text{Cost} \\ &= \$270x - \$230x \\ &= \$40x \end{aligned}$$

which means there is a profit earned on each unit of X produced, so let's produce as many as possible.

There are three constraints that limit how many units can be produced. There is market demand for no more than 40 units per week. Producing $x = 40$ units per week will require 40 hours per week of Labor A, and 80 hours per week of Labor B. Checking those constraints we see that we have enough labor of each type, so the maximum profit will be

$$\max \text{Profit} = \quad \$$$

What we conclude is that market demand is the 'most constraining constraint.' Once we've made that deduction, the rest is a straightforward problem that can be solved by inspection.

Pyomo Model

While this problem can be solved by inspection, here we show a Pyomo model that generates a solution to the problem.

In [2]:

```
from pyomo.environ import *
model = ConcreteModel()

# declare decision variables
model.x = Var(domain=NonNegativeReals)

# declare objective
model.profit = Objective(
    expr = 40*model.x,
    sense = maximize)

# declare constraints
model.demand = Constraint(expr = model.x <= 40)
model.laborA = Constraint(expr = model.x <= 80)
model.laborB = Constraint(expr = 2*model.x <= 100)

# solve
SolverFactory('cbc').solve(model).write()
```

```
# ==========================================================
# = Solver Results                                         =
# ==========================================================
# ----------------------------------------------------------
#   Problem Information
# ----------------------------------------------------------
Problem:
- Name: unknown
  Lower bound: -1600.0
  Upper bound: -1600.0
  Number of objectives: 1
  Number of constraints: 4
  Number of variables: 2
  Number of nonzeros: 4
  Sense: minimize
# ----------------------------------------------------------
#   Solver Information
# ----------------------------------------------------------
Solver:
- Status: ok
  User time: -1.0
  Termination condition: optimal
  Error rc: 0
  Time: 0.01619696617126465
# ----------------------------------------------------------
#   Solution Information
# ----------------------------------------------------------
Solution:
- number of solutions: 0
  number of solutions displayed: 0
```

The results of the solution step show the solver has converged to an optimal solution. Next we display the particular components of the model of interest to us.

In [3]:

```
print("Profit = ", model.profit(), " per week")
print("X = ", model.x(), " units per week")
```

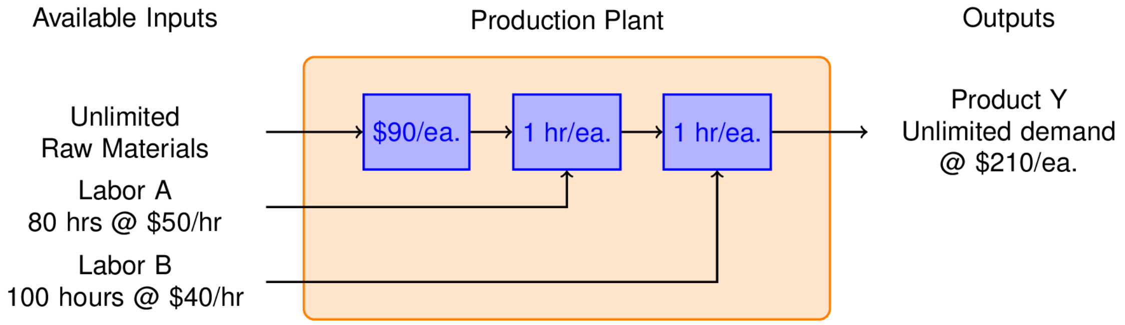
```
Profit =  1600.0  per week
X =  40.0  units per week
```

Exercises

1. Suppose the demand could be increased to 50 units per month. What would be the increased profits? What if the demand increased to 60 units per month? How much would you be willing to pay for your marketing department for the increased demand?
2. Increase the cost of LaborB. At what point is it no longer financially viable to run the plant?

Production Plan: Product Y

Your marketing department has developed plans for a new product called Y. The product sells at a price of USD 210/each, and they expect that you can sell all that you can make. It's also cheaper to make, requiring only USD 90 in raw materials, 1 hour of Labor type A at USD 50 per hour, and 1 hour of Labor B at USD 40 per hour. What is the potential weekly profit?



In [4]:

```
from pyomo.environ import *
model = ConcreteModel()

# declare decision variables
model.y = Var(domain=NonNegativeReals)

# declare objective
model.profit = Objective(
    expr = 30*model.y,
    sense = maximize)

# declare constraints
model.laborA = Constraint(expr = model.y <= 80)
model.laborB = Constraint(expr = model.y <= 100)

# solve
SolverFactory('cbc').solve(model).write()
```

```
# ==========================================================
# = Solver Results                                         =
# ==========================================================
# ----------------------------------------------------------
#   Problem Information
# ----------------------------------------------------------
Problem:
- Name: unknown
  Lower bound: -2400.0
  Upper bound: -2400.0
  Number of objectives: 1
  Number of constraints: 3
  Number of variables: 2
  Number of nonzeros: 3
  Sense: minimize
# ----------------------------------------------------------
#   Solver Information
# ----------------------------------------------------------
Solver:
- Status: ok
  User time: -1.0
  Termination condition: optimal
  Error rc: 0
  Time: 0.022749662399291992
# ----------------------------------------------------------
#   Solution Information
# ----------------------------------------------------------
Solution:
- number of solutions: 0
  number of solutions displayed: 0
```

In [5]:

```
print("Profit = ", model.profit())
print("Units of Y = ", model.y())
```

```
Profit =  2400.0
Units of Y =  80.0
```

Compared to product X, we can manufacture and sell up 80 units per week for a total profit of $2,400. This is very welcome news.

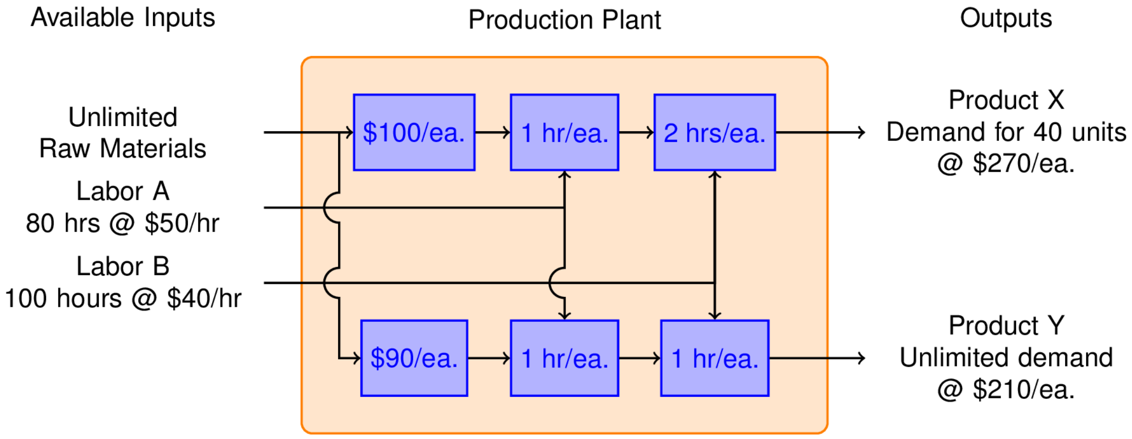
Exercises

1. What is the limiting resource? That is, which of the two types of labor limits the capacity of your plant to produce more units of Y?
2. What rate would you be willing to pay for the additional labor necessary to increase the production of Y?

Production Plan: Mixed Product Strategy

So far we have learned that we can make $1,600 per week by manufacturing product X, and $2,400 per week manufacturing product Y. Is it possible to do even better?

To answer this question, we consider the possibilty of manufacturing both products in the same plant. The marketing department assures us that product Y will not affect the sales of product X. So the same constraints hold as before, but now we have two decision variables, x and y.



In [6]:

```
from pyomo.environ import *
model = ConcreteModel()

# declare decision variables
model.x = Var(domain=NonNegativeReals)
model.y = Var(domain=NonNegativeReals)

# declare objective
model.profit = Objective(
    expr = 40*model.x + 30*model.y,
    sense = maximize)

# declare constraints
model.demand = Constraint(expr = model.x <= 40)
model.laborA = Constraint(expr = model.x + model.y <= 80)
model.laborB = Constraint(expr = 2*model.x + model.y <= 100)

# solve
SolverFactory('cbc').solve(model).write()
```

```
# ==========================================================
# = Solver Results                                         =
# ==========================================================
# ----------------------------------------------------------
#   Problem Information
# ----------------------------------------------------------
Problem:
- Name: unknown
  Lower bound: -2600.0
  Upper bound: -2600.0
  Number of objectives: 1
  Number of constraints: 4
  Number of variables: 3
  Number of nonzeros: 6
  Sense: minimize
# ----------------------------------------------------------
#   Solver Information
# ----------------------------------------------------------
Solver:
- Status: ok
  User time: -1.0
  Termination condition: optimal
  Error rc: 0
  Time: 0.023347616195678711
# ----------------------------------------------------------
#   Solution Information
# ----------------------------------------------------------
Solution:
- number of solutions: 0
  number of solutions displayed: 0
```

In [7]:

```
# display solution
print("Profit = ", model.profit())
print("Units of X = ", model.x())
print("Units of Y = ", model.y())
```

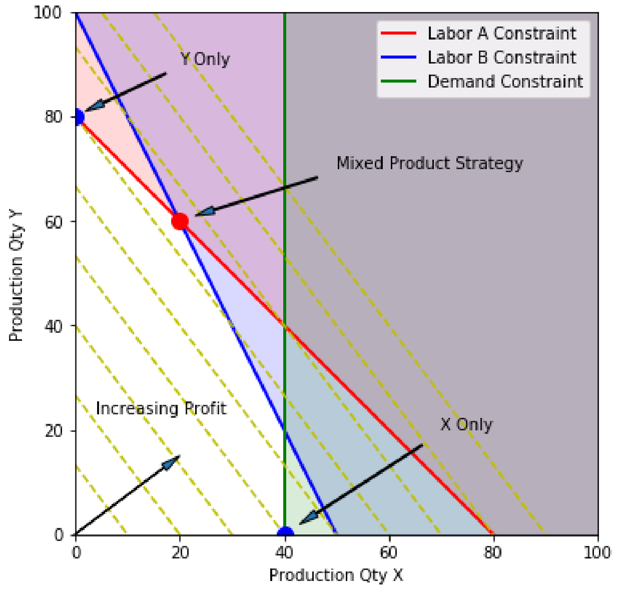
```
Profit =  2600.0
Units of X =  20.0
Units of Y =  60.0
```

The mixed product strategy earns more profit than either of the single product srategies. Does this surprise you? Before going further, try to explain why it is possible for a mixed product strategy to earn more profit than either of the possible single product strategies.

What are the active constraints?

In [8]:

```
%matplotlib inline
from pylab import *

figure(figsize=(6,6))
subplot(111, aspect='equal')
axis([0,100,0,100])
xlabel('Production Qty X')
ylabel('Production Qty Y')

# Labor A constraint
x = array([0,80])
y = 80 - x
plot(x,y,'r',lw=2)
fill_between([0,80,100],[80,0,0],[100,100,100],color='r',alpha=0.15)

# Labor B constraint
x = array([0,50])
y = 100 - 2*x
plot(x,y,'b',lw=2)
fill_between([0,50,100],[100,0,0],[100,100,100],color='b',alpha=0.15)

# Demand constraint
plot([40,40],[0,100],'g',lw=2)
fill_between([40,100],[0,0],[100,100],color='g',alpha=0.15)

legend(['Labor A Constraint','Labor B Constraint','Demand Constraint'])

# Contours of constant profit
x = array([0,100])
for p in linspace(0,3600,10):
    y = (p - 40*x)/30
    plot(x,y,'y--')

# Optimum
plot(20,60,'r.',ms=20)
annotate('Mixed Product Strategy', xy=(20,60), xytext=(50,70),
         arrowprops=dict(shrink=.1,width=1,headwidth=5))

plot(0,80,'b.',ms=20)
annotate('Y Only', xy=(0,80), xytext=(20,90),
         arrowprops=dict(shrink=0.1,width=1,headwidth=5))

plot(40,0,'b.',ms=20)
annotate('X Only', xy=(40,0), xytext=(70,20),
         arrowprops=dict(shrink=0.1,width=1,headwidth=5))

text(4,23,'Increasing Profit')
annotate('', xy=(20,15), xytext=(0,0),
         arrowprops=dict(width=0.5,headwidth=5))

savefig('LPprob01.png',bbox_inches='tight')
```



What is the incremental value of labor?

In [9]:

```
from pyomo.environ import *
model = ConcreteModel()

# for access to dual solution for constraints
model.dual = Suffix(direction=Suffix.IMPORT)

# declare decision variables
model.x = Var(domain=NonNegativeReals)
model.y = Var(domain=NonNegativeReals)

# declare objective
model.profit = Objective(
    expr = 40*model.x + 30*model.y,
    sense = maximize)

# declare constraints
model.demand = Constraint(expr = model.x <= 40)
model.laborA = Constraint(expr = model.x + model.y <= 80)
model.laborB = Constraint(expr = 2*model.x + model.y <= 100)

# solve
SolverFactory('cbc').solve(model).write()
```

```
# ==========================================================
# = Solver Results                                         =
# ==========================================================
# ----------------------------------------------------------
#   Problem Information
# ----------------------------------------------------------
Problem:
- Name: unknown
  Lower bound: -2600.0
  Upper bound: -2600.0
  Number of objectives: 1
  Number of constraints: 4
  Number of variables: 3
  Number of nonzeros: 6
  Sense: minimize
# ----------------------------------------------------------
#   Solver Information
# ----------------------------------------------------------
Solver:
- Status: ok
  User time: -1.0
  Termination condition: optimal
  Error rc: 0
  Time: 0.023309707641601562
# ----------------------------------------------------------
#   Solution Information
# ----------------------------------------------------------
Solution:
- number of solutions: 0
  number of solutions displayed: 0
```

Analysis of the constraints.

In [10]:

```
str = "   {0:7.2f} {1:7.2f} {2:7.2f} {3:7.2f}"

print("Constraint  value  lslack  uslack    dual")
for c in [model.demand,model.laborA,model.laborB]:
    print(c, str.format(c(), c.lslack(), c.uslack(), model.dual[c]))
```

```
Constraint  value  lslack  uslack    dual
demand       20.00     inf   20.00    0.00
laborA       80.00     inf    0.00  -20.00
laborB      100.00     inf    0.00  -10.00
```

Theory of Constraints

- For n decisions you should expect to find n 'active' constraints.
- Each inactive constraint has an associated 'slack.' The associated resources have no incremental value.
- Each active constraint has an associated 'shadow price'. This is additional value of additional resources.

In [ ]:

```

```