LAF+ARG

November 8, 2016

1 Minimize the cost of shipment, given that Brazil can not manufacture.

In [11]:
```
using JuMP
```

2 2016

In [12]:
```
ORIG = ["LAF", "MDP", "PAB-LAF", "PAB-MDP"];
DEST = ["ILN", "BLG", "JPN", "CHN", "MEX", "PER", "SINK"];

supply = [16 16 15 15];
BR=15
demand = [8.86 0.74 8.26 8.86 2.96 4.42 0]



#@assert sum(supply) == sum(demand);
bl = 700
ba=150
cost = [
120         1250      2580      2300      450      720   0;
720         1330      1550      1450      450      220   0;
120+bl   1250+bl   2580+bl   2300+bl   450+bl   720+bl   0;
720+ba   1330+ba   1550+ba   1450+ba   450+ba   220+ba   0
]

m = Model();

@variable(m, Trans[i=1:length(ORIG), j=1:length(DEST)] >= 0);


@objective(m, Min, sum{cost[i,j] * Trans[i,j], i=1:length(ORIG), j=1:length(DEST)});

@constraint(m, xyconstr[i=1:1:length(ORIG)], sum{Trans[i,j], j=1:length(DEST)} == supply[i]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=1:length(ORIG)} >= demand[j]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=3:(length(ORIG)-1)} <= BR);


println("Solving original problem...")
status = solve(m);
```

```
        if status == :Optimal
            @printf("Optimal!\n");
            @printf("Objective value: %.2f\n", getobjectivevalue(m));
            @printf("Transpotation:\n");
            for j = 1:length(DEST)
                @printf("\t%s", DEST[j]);
            end
            @printf("\n");
            for i = 1:length(ORIG)
                @printf("%s", ORIG[i]);
                for j = 1:length(DEST)
                    @printf("\t%.2f", getvalue(Trans[i,j]));
                end
                @printf("\n");
            end
        else
            @printf("No solution\n");
        end
```

```
Solving original problem...
Optimal!
Objective value: 30773.60
Transpotation:
        ILN        BLG        JPN        CHN        MEX        PER        SINK
LAF        8.86        0.74        0.00        0.00        2.96        0.00        3.44
MDP        0.00        0.00        8.26        7.74        0.00        0.00        0.00
PAB-LAF        0.00        0.00        0.00        0.00        0.00        0.00        15.00
PAB-MDP        0.00        0.00        0.00        1.12        0.00        4.42        9.46
```

3 2017

In [13]:
```
ORIG = ["LAF", "MDP", "PAB-LAF", "PAB-MDP"];
DEST = ["ILN", "BLG", "JPN", "CHN", "MEX", "PER", "SINK"];

supply = [16 18 18 18];
BR=18
demand = [8.86 0.74 8.26 8.86 2.96 4.42 0]



#@assert sum(supply) == sum(demand);
bl = 700
ba=150
cost = [
120         1250      2580      2300      450      720   0;
720         1330      1550      1450      450      220   0;
120+bl   1250+bl   2580+bl   2300+bl   450+bl   720+bl   0;
720+ba   1330+ba   1550+ba   1450+ba   450+ba   220+ba   0
]

m = Model();

@variable(m, Trans[i=1:length(ORIG), j=1:length(DEST)] >= 0);
```

```
@objective(m, Min, sum{cost[i,j] * Trans[i,j], i=1:length(ORIG), j=1:length(DEST)});

@constraint(m, xyconstr[i=1:1:length(ORIG)], sum{Trans[i,j], j=1:length(DEST)} == supply[i]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=1:length(ORIG)} >= demand[j]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=3:(length(ORIG)-1)} <= BR);


println("Solving original problem...")
status = solve(m);

if status == :Optimal
    @printf("Optimal!\n");
    @printf("Objective value: %.2f\n", getobjectivevalue(m));
    @printf("Transpotation:\n");
    for j = 1:length(DEST)
        @printf("\t%s", DEST[j]);
    end
    @printf("\n");
    for i = 1:length(ORIG)
        @printf("%s", ORIG[i]);
        for j = 1:length(DEST)
            @printf("\t%.2f", getvalue(Trans[i,j]));
        end
        @printf("\n");
    end
else
    @printf("No solution\n");
end
```

```
Solving original problem...
Optimal!
Objective value: 30473.60
Transpotation:
        ILN         BLG         JPN         CHN         MEX         PER         SINK
LAF         8.86        0.74        0.00        0.00        2.96        0.00        3.44
MDP         0.00        0.00        8.26        8.86        0.00        0.88        0.00
PAB-LAF         0.00        0.00        0.00        0.00        0.00        0.00        18.00
PAB-MDP         0.00        0.00        0.00        0.00        0.00        3.54        14.46
```

4 2018

In [14]:
```
ORIG = ["LAF", "MDP", "PAB-LAF", "PAB-MDP"];
DEST = ["ILN", "BLG", "JPN", "CHN", "MEX", "PER", "SINK"];

supply = [21 18 12 12];
BR=12
demand = [11.88 0.99 11.09 11.88 3.96 5.94 0]
```

```
#@assert sum(supply) == sum(demand);
bl = 700
ba=150
cost = [
120         1250      2580      2300      450      720   0;
720         1330      1550      1450      450      220   0;
120+bl   1250+bl   2580+bl   2300+bl   450+bl   720+bl   0;
720+ba   1330+ba   1550+ba   1450+ba   450+ba   220+ba   0
]


m = Model();

@variable(m, Trans[i=1:length(ORIG), j=1:length(DEST)] >= 0);


@objective(m, Min, sum{cost[i,j] * Trans[i,j], i=1:length(ORIG), j=1:length(DEST)});

@constraint(m, xyconstr[i=1:1:length(ORIG)], sum{Trans[i,j], j=1:length(DEST)} == supply[i]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=1:length(ORIG)} >= demand[j]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=3:(length(ORIG)-1)} <= BR);


println("Solving original problem...")
status = solve(m);

if status == :Optimal
    @printf("Optimal!\n");
    @printf("Objective value: %.2f\n", getobjectivevalue(m));
    @printf("Transpotation:\n");
    for j = 1:length(DEST)
        @printf("\t%s", DEST[j]);
    end
    @printf("\n");
    for i = 1:length(ORIG)
        @printf("%s", ORIG[i]);
        for j = 1:length(DEST)
            @printf("\t%.2f", getvalue(Trans[i,j]));
        end
        @printf("\n");
    end
else
    @printf("No solution\n");
end
```

```
Solving original problem...
Optimal!
Objective value: 41803.90
Transpotation:
        ILN         BLG         JPN         CHN         MEX         PER         SINK
LAF         11.88         0.99         0.00         0.00         3.96         0.00         4.17
MDP         0.00         0.00         11.09         6.91         0.00         0.00         0.00
```

```
PAB-LAF        0.00        0.00        0.00        0.00        0.00        0.00        12.00
PAB-MDP        0.00        0.00        0.00        4.97        0.00        5.94        1.09
```

5 2019

In [15]:
```
ORIG = ["LAF", "MDP", "PAB-LAF", "PAB-MDP"];
DEST = ["ILN", "BLG", "JPN", "CHN", "MEX", "PER", "SINK"];

supply = [22 20 20 20];
BR=20
demand = [12.56 1.04 11.64 12.47 4.16 6.24 0]



#@assert sum(supply) == sum(demand);
bl = 700
ba=150
cost = [
120         1250      2580      2300      450      720   0;
720         1330      1550      1450      450      220   0;
120+bl   1250+bl   2580+bl   2300+bl   450+bl   720+bl   0;
720+ba   1330+ba   1550+ba   1450+ba   450+ba   220+ba   0
]


m = Model();

@variable(m, Trans[i=1:length(ORIG), j=1:length(DEST)] >= 0);


@objective(m, Min, sum{cost[i,j] * Trans[i,j], i=1:length(ORIG), j=1:length(DEST)});

@constraint(m, xyconstr[i=1:1:length(ORIG)], sum{Trans[i,j], j=1:length(DEST)} == supply[i]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=1:length(ORIG)} >= demand[j]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=3:(length(ORIG)-1)} <= BR);


println("Solving original problem...")
status = solve(m);

if status == :Optimal
    @printf("Optimal!\n");
    @printf("Objective value: %.2f\n", getobjectivevalue(m));
    @printf("Transpotation:\n");
    for j = 1:length(DEST)
        @printf("\t%s", DEST[j]);
    end
    @printf("\n");
    for i = 1:length(ORIG)
        @printf("%s", ORIG[i]);
        for j = 1:length(DEST)
```

```
                @printf("\t%.2f", getvalue(Trans[i,j]));
            end
            @printf("\n");
        end
    else
        @printf("No solution\n");
    end
```

```
Solving original problem...
Optimal!
Objective value: 43728.00
Transpotation:
        ILN        BLG        JPN        CHN        MEX        PER        SINK
LAF       12.56       1.04       0.00       0.00       4.16       0.00       4.24
MDP       0.00       0.00      11.64       8.36       0.00       0.00       0.00
PAB-LAF       0.00       0.00       0.00       0.00       0.00       0.00      20.00
PAB-MDP       0.00       0.00       0.00       4.11       0.00       6.24       9.65
```

6 2020

In [16]:
```
ORIG = ["LAF", "MDP", "PAB-LAF", "PAB-MDP"];
DEST = ["ILN", "BLG", "JPN", "CHN", "MEX", "PER", "SINK"];

supply = [28 24 21 21];
BR=21
demand = [14.02 1.17 13.09 14.02 4.67 7.01 0]



#@assert sum(supply) == sum(demand);
bl = 700
ba=150
cost = [
120         1250      2580      2300      450      720   0;
720         1330      1550      1450      450      220   0;
120+bl   1250+bl   2580+bl   2300+bl   450+bl   720+bl   0;
720+ba   1330+ba   1550+ba   1450+ba   450+ba   220+ba   0
]

m = Model();

@variable(m, Trans[i=1:length(ORIG), j=1:length(DEST)] >= 0);


@objective(m, Min, sum{cost[i,j] * Trans[i,j], i=1:length(ORIG), j=1:length(DEST)});

@constraint(m, xyconstr[i=1:1:length(ORIG)], sum{Trans[i,j], j=1:length(DEST)} == supply[i]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=1:length(ORIG)} >= demand[j]);

@constraint(m, xyconstr[j = 1:length(DEST)], sum{Trans[i,j], i=3:(length(ORIG)-1)} <= BR);


println("Solving original problem...")
```

```
status = solve(m);

if status == :Optimal
    @printf("Optimal!\n");
    @printf("Objective value: %.2f\n", getobjectivevalue(m));
    @printf("Transpotation:\n");
    for j = 1:length(DEST)
        @printf("\t%s", DEST[j]);
    end
    @printf("\n");
    for i = 1:length(ORIG)
        @printf("%s", ORIG[i]);
        for j = 1:length(DEST)
            @printf("\t%.2f", getvalue(Trans[i,j]));
        end
        @printf("\n");
    end
else
    @printf("No solution\n");
end
```

```
Solving original problem...
Optimal!
Objective value: 48925.10
Transpotation:
        ILN     BLG     JPN     CHN     MEX     PER     SINK
LAF     14.02   1.17    0.00    0.00    4.67    0.00    8.14
MDP     0.00    0.00    13.09   10.91   0.00    0.00    0.00
PAB-LAF 0.00    0.00    0.00    0.00    0.00    0.00    21.00
PAB-MDP 0.00    0.00    0.00    3.11    0.00    7.01    10.88
```

In [ ]: