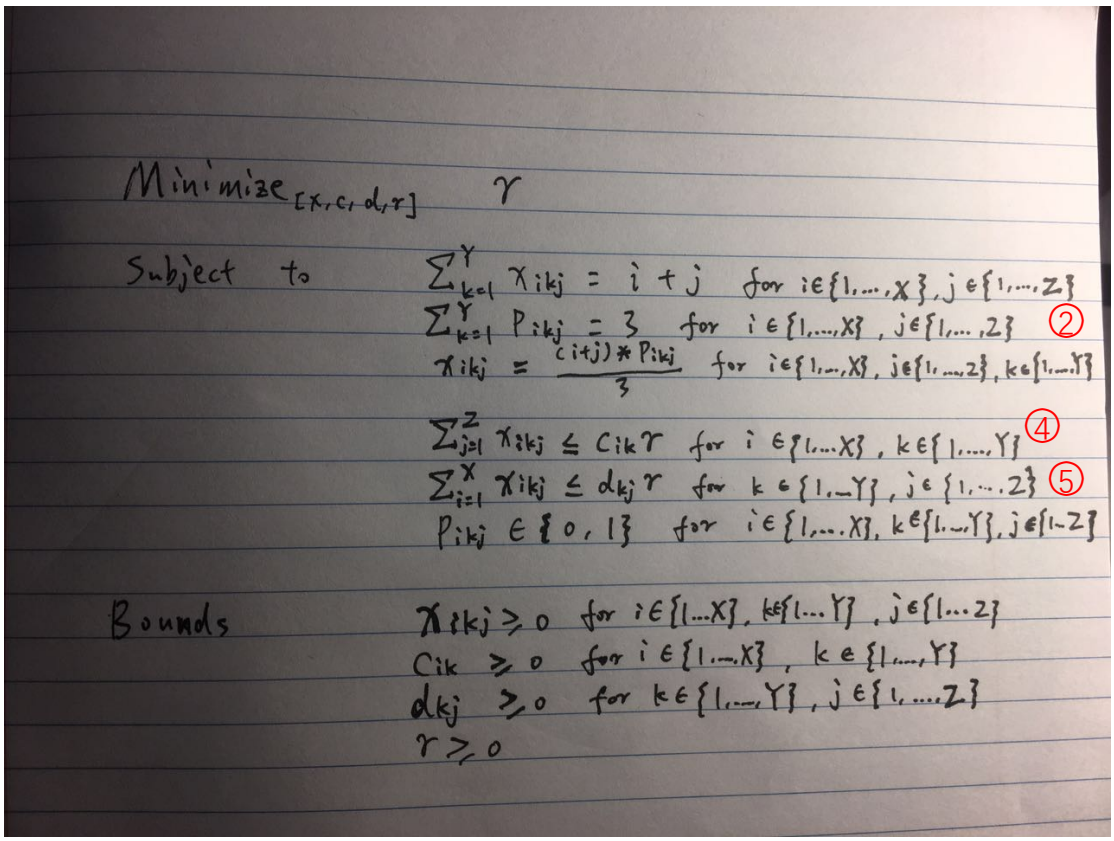
COSC364

Assignment 2

Jingwei Li

39923031

Individual submission permitted on May 4th



The objective is to balance the load, so we introduce r to "linearise" this particular objective function(max{flow / capacity,⋯}), so that we end up with a linear program.

For ①, it corresponds the amount of flow from source i to destination j. $h_{ij} = i+j$.

For ②, it corresponds the selected paths from source i to destination j constrains to 3, in other words, there are only 3 transit nodes are using.

For ③, it corresponds each path gets an equal share of the demand volume.

For ④ and ⑤, these are additional constraints which are the transformation of the objective function. Each flow limits to the capacity of the link.

For ⑥, 1 means this is a selected path, otherwise 0.

And for the demand volume, capacity of links and utilization, they are all non-negative real numbers.

Y	3	4	5	6	7
Time (s)	0.00127	0.0722	0.2924	0.1767	0.1045
Non-zero links	42	56	70	84	94
Highest capacity	26	21	17	13	11

We can reduce the capacities of links by adding more transit nodes, also, it needs more links to support, and then it makes many idle links.

According to the data we collect, 7 transit nodes is the best option. When it is more than 7, it takes longer time.

And the CPLEX execution time increases as the number of transit nodes increases, but it is decreasing when it is closing to 7.

Appendix

Source code

```
import sys


def printObjFunc():
    print("Minimize")
    print("r")


def print1(x, y, z):
    for i in range(1, x+1):
        for j in range(1, z+1):
            n = i + j
            for k in range(1, y+1):
                if(k == y):
                    print(" + x{}{}{} = {}".format(i,k,j,n))
                elif(k == 1):
                    print("x{}{}{}".format(i,k,j), end='')
                else:
                    print(" + x{}{}{}".format(i,k,j), end='')


def print2(x, y, z):
```

```
    for i in range(1, x+1):
        for j in range(1, z+1):
            for k in range(1, y+1):
                if(k == y):
                    print(" + p{}{}{} = 3".format(i,k,j))
                elif(k == 1):
                    print("p{}{}{}".format(i,k,j), end='')
                else:
                    print(" + p{}{}{}".format(i,k,j), end='')


def print3(x, y, z):
    for i in range(1, x+1):
        for k in range(1, y+1):
            for j in range(1, z+1):
                n = i + j
                print("3 x{}{}{} - {} p{}{}{} = 0".format(i,k,j,n,i,k,j))



def print4(x, y, z):
    for i in range(1, x+1):
        for k in range(1, y+1):
            for j in range(1, z+1):
```

```
                if(j == z):
                    print(" + x{}{}{} - 30 r <= 0".format(i,k,j))
                elif(j == 1):
                    print("x{}{}{}".format(i,k,j), end='')
                else:
                    print(" + x{}{}{}".format(i,k,j), end='')


def print5(x, y, z):
    for k in range(1, y+1):
        for j in range(1, z+1):
            for i in range(1, x+1):
                if(i == x):
                    print(" + x{}{}{} - 30 r <= 0".format(i,k,j))
                elif(i == 1):
                    print("x{}{}{}".format(i,k,j), end='')
                else:
                    print(" + x{}{}{}".format(i,k,j), end='')


def print6(x, y, z):
    for i in range(1, x+1):
        for k in range(1, y+1):
```

```
            for j in range(1, z+1):
                if(j == z):
                    print(" + x{}{}{} - c{}{} <= 0".format(i,k,j,i,k))
                elif(j == 1):
                    print("x{}{}{}".format(i,k,j), end='')
                else:
                    print(" + x{}{}{}".format(i,k,j), end='')


def print7(x, y, z):
    for k in range(1, y+1):
        for j in range(1, z+1):
            for i in range(1, x+1):
                if(i == x):
                    print(" + x{}{}{} - d{}{} <= 0".format(i,k,j,k,j))
                elif(i == 1):
                    print("x{}{}{}".format(i,k,j), end='')
                else:
                    print(" + x{}{}{}".format(i,k,j), end='')


def printConst(x, y, z):
```

```
    print("Subject to")
    print1(x, y, z)
    print2(x, y, z)
    print3(x, y, z)
    print4(x, y, z)
    print5(x, y, z)
    print6(x, y, z)
    print7(x, y, z)


def printx(x, y, z):
    for i in range(1, x+1):
        for k in range(1, y+1):
            for j in range(1, z+1):
                print("0 <= x{}{}{}".format(i,k,j))


def printc(x, y):
    for i in range(1, x+1):
        for k in range(1, y+1):
            print("0 <= c{}{}".format(i,k))


def printd(y, z):
    for k in range(1, y+1):
```

```
            for j in range(1, z+1):
                print("0 <= d{}{}".format(k,j))


def printBounds(x, y, z):
    print("Bounds")
    printx(x, y, z)
    printc(x, y)
    printd(y, z)
    print("0 <= r")



def printBin(x, y, z):
    print("Binaries")
    for i in range(1, x+1):
        for k in range(1, y+1):
            for j in range(1, z+1):
                print("p{}{}{}".format(i,k,j))




def main():
```

```
    if(len(sys.argv) == 4):
        x = int(sys.argv[1])
        y = int(sys.argv[2])
        z = int(sys.argv[3])
    else:
        print("Please enter 3 integers for X Y Z")
        return 0

    printObjFunc()
    printConst(x, y, z)
    printBounds(x, y, z)
    printBin(x, y, z)
    print("End")

if __name__ == "__main__":
    main()
```

Generated LP file for x=3, y=2, z=4

```
Minimize
r
Subject to
x111 + x121 = 2
x112 + x122 = 3
x113 + x123 = 4
x114 + x124 = 5
x211 + x221 = 3
x212 + x222 = 4
x213 + x223 = 5
x214 + x224 = 6
x311 + x321 = 4
x312 + x322 = 5
x313 + x323 = 6
x314 + x324 = 7
p111 + p121 = 3
p112 + p122 = 3
p113 + p123 = 3
p114 + p124 = 3
p211 + p221 = 3
p212 + p222 = 3
p213 + p223 = 3
p214 + p224 = 3
p311 + p321 = 3
p312 + p322 = 3
p313 + p323 = 3
p314 + p324 = 3
3 x111 - 2 p111 = 0
3 x112 - 3 p112 = 0
3 x113 - 4 p113 = 0
3 x114 - 5 p114 = 0
3 x121 - 2 p121 = 0
3 x122 - 3 p122 = 0
3 x123 - 4 p123 = 0
3 x124 - 5 p124 = 0
3 x211 - 3 p211 = 0
3 x212 - 4 p212 = 0
3 x213 - 5 p213 = 0
3 x214 - 6 p214 = 0
3 x221 - 3 p221 = 0
3 x222 - 4 p222 = 0
3 x223 - 5 p223 = 0
```

```
3 x224 - 6 p224 = 0
3 x311 - 4 p311 = 0
3 x312 - 5 p312 = 0
3 x313 - 6 p313 = 0
3 x314 - 7 p314 = 0
3 x321 - 4 p321 = 0
3 x322 - 5 p322 = 0
3 x323 - 6 p323 = 0
3 x324 - 7 p324 = 0
x111 + x112 + x113 + x114 - 30 r <= 0
x121 + x122 + x123 + x124 - 30 r <= 0
x211 + x212 + x213 + x214 - 30 r <= 0
x221 + x222 + x223 + x224 - 30 r <= 0
x311 + x312 + x313 + x314 - 30 r <= 0
x321 + x322 + x323 + x324 - 30 r <= 0
x111 + x211 + x311 - 30 r <= 0
x112 + x212 + x312 - 30 r <= 0
x113 + x213 + x313 - 30 r <= 0
x114 + x214 + x314 - 30 r <= 0
x121 + x221 + x321 - 30 r <= 0
x122 + x222 + x322 - 30 r <= 0
x123 + x223 + x323 - 30 r <= 0
x124 + x224 + x324 - 30 r <= 0
x111 + x112 + x113 + x114 - c11 <= 0
x121 + x122 + x123 + x124 - c12 <= 0
x211 + x212 + x213 + x214 - c21 <= 0
x221 + x222 + x223 + x224 - c22 <= 0
x311 + x312 + x313 + x314 - c31 <= 0
x321 + x322 + x323 + x324 - c32 <= 0
x111 + x211 + x311 - d11 <= 0
x112 + x212 + x312 - d12 <= 0
x113 + x213 + x313 - d13 <= 0
x114 + x214 + x314 - d14 <= 0
x121 + x221 + x321 - d21 <= 0
x122 + x222 + x322 - d22 <= 0
x123 + x223 + x323 - d23 <= 0
x124 + x224 + x324 - d24 <= 0
Bounds
0 <= x111
0 <= x112
0 <= x113
0 <= x114
0 <= x121
0 <= x122
```

0 <= x123
0 <= x124
0 <= x211
0 <= x212
0 <= x213
0 <= x214
0 <= x221
0 <= x222
0 <= x223
0 <= x224
0 <= x311
0 <= x312
0 <= x313
0 <= x314
0 <= x321
0 <= x322
0 <= x323
0 <= x324
0 <= c11
0 <= c12
0 <= c21
0 <= c22
0 <= c31
0 <= c32
0 <= d11
0 <= d12
0 <= d13
0 <= d14
0 <= d21
0 <= d22
0 <= d23
0 <= d24
0 <= r
Binaries
p111
p112
p113
p114
p121
p122
p123
p124
p211
p212

p213
p214
p221
p222
p223
p224
p311
p312
p313
p314
p321
p322
p323
p324
End