学号：　201618013229011
姓名：　李坚松

Assignment 4

Problem-2

Solution

We mark the landing time airplane I by x_i, and the smallest gap by z, then we can get the following formulation:

$$\begin{aligned}&\max\ z\\&s.t.\ s_i <= \mathrm{x}_i <= t_i, i=1,2,...,n\\&x_{i+1}-x_i >= z, i=1,2,...,n-1\\&x_i >= 0, z >= 0, i=1,2,...,n\end{aligned}$$

We take given data for example, the time window of 3 airplanes are [10:00-11:00],[11:20-11:40],[12:00-12:20], using GLPK we input the following source code,

```
/* Variables */
var x1 >= 0;
var x2 >= 0;
var x3 >= 0;
var z0 >= 0;

/* Object function */
maximize z: z0;

/* Constrains */
s.t. con1: x1 >= 10;
s.t. con2: x1    <= 11;
s.t. con3: x2 - x1 >= z0;
s.t. con4: x2 <= 11.66;
s.t. con5: x2 >= 11.33;
s.t. con6: x3 - x2 >= z0;
s.t. con7: x3 <= 12.33;
s.t. con8: x3 >= 12;

end;
```

By the following command, “glpsol -m p2.mod -o p2.txt”, we can get the following result:

```
Problem:    p2
```

Rows: 9
Columns: 4
Non-zeros: 13
Status: OPTIMAL
Objective: z = 1 (MAXimum)

No.	Row name	St	Activity	Lower bound	Upper bound	Marginal
1	z	B	1			
2	con1	NL	10	10		< eps
3	con2	B	10		11	
4	con3	B	0.33	-0		
5	con4	B	11.33		11.66	
6	con5	NL	11.33	11.33		-1
7	con6	NL	0	-0		-1
8	con7	NU	12.33		12.33	1
9	con8	B	12.33	12		

No.	Column name	St	Activity	Lower bound	Upper bound	Marginal
1	x1	B	10	0		
2	x2	B	11.33	0		
3	x3	B	12.33	0		
4	z0	B	1	0		

Karush-Kuhn-Tucker optimality conditions:

KKT.PE: max.abs.err = 0.00e+00 on row 0
max.rel.err = 0.00e+00 on row 0
High quality

KKT.PB: max.abs.err = 0.00e+00 on row 0
max.rel.err = 0.00e+00 on row 0
High quality

KKT.DE: max.abs.err = 0.00e+00 on column 0
max.rel.err = 0.00e+00 on column 0
High quality

KKT.DB: max.abs.err = 0.00e+00 on row 0
max.rel.err = 0.00e+00 on row 0
High quality

End of output

So, clearly we can know that the maximum gap is 1 hour when x1=10,x2=11.33,x3=12.33, i.e. the three airplanes land at 10:00, 11:20, 12:20 respectively.

Problem-4

Solution

We mark the maximum distance between two successive gas station by z, and the distance or gas station i is x_i, then we can get the following formulation:

$$
\begin{aligned}
&\min\ z \\
&s.t.\ d_i - r <= x_i <= d_i + r, i = 1, 2, \ldots, n \\
&d_{i+1} - d_i <= z, i = 1, 2, \ldots, n-1 \\
&x_i >= 0, z >= 0, i = 1, 2, \ldots, n
\end{aligned}
$$

Problem -7

Referring to the book <**Introduction to Algorithms**> 3rd edition, we get the following Simplex Algorithm:

```
SIMPLEX(A, b, c)
(N, B, A, b, c, v) = INITIALIZE-SIMPLEX(A, b, c)
let Δ be a new vector of length n
while some index j ∈ N has c_j > 0
    choose an index e ∈ N for which c_e > 0
    for each index i ∈ B
        if a_ie > 0
            Δ_i = b_i / a_ie
        else Δ_i = ∞
    choose an index l ∈ B that minimizes Δ_i
    if Δ_l == ∞
        return "unbounded"
    else (N, B, A, b, c, v) = PIVOT(N, B, A, b, c, v, l, e)
for i = 1 to n
    if i ∈ B
        x̄_i = b_i
    else x̄_i = 0
return (x̄_1, x̄_2, ..., x̄_n)
```

The pivot operation is shown below:

PIVOT(N, B, A, b, c, v, l, e)

// Compute the coefficients of the equation for new basic variable x_e.
let $\widehat{A}$ be a new $m \times n$ matrix
$\widehat{b}_e = b_l / a_{le}$
for each $j \in N - \{e\}$
    $\widehat{a}_{ej} = a_{lj} / a_{le}$
$\widehat{a}_{el} = 1 / a_{le}$
// Compute the coefficients of the remaining constraints.
for each $i \in B - \{l\}$
    $\widehat{b}_i = b_i - a_{ie}\widehat{b}_e$
    for each $j \in N - \{e\}$
        $\widehat{a}_{ij} = a_{ij} - a_{ie}\widehat{a}_{ej}$
    $\widehat{a}_{il} = -a_{ie}\widehat{a}_{el}$
// Compute the objective function.
$\widehat{v} = v + c_e\widehat{b}_e$
for each $j \in N - \{e\}$
    $\widehat{c}_j = c_j - c_e\widehat{a}_{ej}$
$\widehat{c}_l = -c_e\widehat{a}_{el}$
// Compute new sets of basic and nonbasic variables.
$\widehat{N} = N - \{e\} \cup \{l\}$
$\widehat{B} = B - \{l\} \cup \{e\}$
return $(\widehat{N}, \widehat{B}, \widehat{A}, \widehat{b}, \widehat{c}, \widehat{v})$

The initialize operation is shown below:

INITIALIZE-SIMPLEX(A, b, c)

let k be the index of the minimum b_i
if $b_k \geq 0$ // is the initial basic solution feasible?
    return $(\{1, 2, \ldots, n\}, \{n+1, n+2, \ldots, n+m\}, A, b, c, 0)$
form L_{aux} by adding $-x_0$ to the left-hand side of each constraint
    and setting the objective function to $-x_0$
let (N, B, A, b, c, v) be the resulting slack form for L_{aux}
$l = n + k$
// L_{aux} has $n + 1$ nonbasic variables and m basic variables.
$(N, B, A, b, c, v) =$ PIVOT$(N, B, A, b, c, v, l, 0)$
// The basic solution is now feasible for L_{aux}.
iterate the **while** loop of lines 3–12 of SIMPLEX until an optimal solution
    to L_{aux} is found
if the optimal solution to L_{aux} sets $\bar{x}_0$ to 0
    if $\bar{x}_0$ is basic
        perform one (degenerate) pivot to make it nonbasic
    from the final slack form of L_{aux}, remove x_0 from the constraints and
        restore the original objective function of L, but replace each basic
        variable in this objective function by the right-hand side of its
        associated constraint
    return the modified final slack form
else return "infeasible"

Input

We define the input format like this:
N M
min / max
objective function
constraint equations
where N is the number of variables, M is the number of constraint equations, min/max means that whether we want to get the minimum or maximum of the objective function, and the objective function is the coefficient of the variables, e.g. we have the following input:
3 3
max
3 1 2
1 1 3 <= 30
2 2 5 <= 24
4 1 2 <= 36
It means that we want to solve the following linear programming problem:

$$\max\ 3x_1 + x_2 + 2x_3$$
$$s.t.$$
$$x_1 + x_2 + 3x_3 \le 30$$
$$2x_1 + 2x_2 + 5x_3 \le 24$$
$$4x_1 + x_2 + 2x_3 \le 36$$
$$x_1, x_2, x_3 \ge 0$$

Based on the pesudo code and the input format, we can implement simplex algorithm in cpp, see the source file simplex.cpp

Output

If we take the given example as input, we will get the following output:

```
js-lee@jslee-virtual-machine:~/workdir/simplex$ g++ -o simplex simplex.cpp
js-lee@jslee-virtual-machine:~/workdir/simplex$ ./simplex input.txt
Time used 0.000487 seconds
```

```
Final Table:
0 0 0.5 1 -0.5 0 | 18
0 1 2.66667 0 0.666667 -0.333333 | 4
1 0 -0.166667 0 -0.166667 0.333333 | 8
0 0 0.166667 0 0.166667 0.666667 | 28

z* = 28
x* = (8, 4, 0)
```

Obviously, when x1=8, x2=4,x3=0, max objective function value is 28, it takes about 0.000487s to solve this linear programming problem.

Comparison

Run the input example by GLPK, input the source code:

```
/* Variables */
var x1 >= 0;
var x2 >= 0;
var x3 >= 0;

/* Object function */
maximize z: 3*x1 + x2 + 2*x3;

/* Constrains */
s.t. con1: x1 + x2 + 3*x3 <= 30;
s.t. con2: 2*x1 + 2*x2 + 5*x3    <= 24;
s.t. con3: 4*x1 + 1*x2 + 2*x3    <= 36;

end;
```

Then we will get the following results:

```
js-lee@jslee-virtual-machine:~/workdir/simplex$ glpsol -m input.mod -o out-glpk.txt
GLPSOL: GLPK LP/MIP Solver, v4.60
Parameter(s) specified in the command line:
 -m input.mod -o out-glpk.txt
Reading model section from input.mod...
15 lines were read
Generating z...
Generating con1...
Generating con2...
Generating con3...
Model has been successfully generated
GLPK Simplex Optimizer, v4.60
4 rows, 3 columns, 12 non-zeros
Preprocessing...
3 rows, 3 columns, 9 non-zeros
Scaling...
 A: min|aij| =  1.000e+00  max|aij| =  5.000e+00  ratio =  5.000e+00
Problem data seem to be well scaled
Constructing initial basis...
Size of triangular part is 3
*     0: obj =  -0.000000000e+00 inf =   0.000e+00 (3)
*     3: obj =   2.800000000e+01 inf =   0.000e+00 (0)
OPTIMAL LP SOLUTION FOUND
Time used:   0.0 secs
Memory used: 0.1 Mb (102294 bytes)
Writing basic solution to 'out-glpk.txt'...
```

```
Problem:    input
Rows:       4
Columns:    3
Non-zeros:  12
Status:     OPTIMAL
Objective:  z = 28 (MAXimum)

   No.   Row name   St   Activity     Lower bound   Upper bound    Marginal
------ ------------ -- ------------- ------------- ------------- -------------
     1 z            B             28
     2 con1         B             12                          30
     3 con2         NU            24                          24      0.166667
     4 con3         NU            36                          36      0.666667

   No. Column name  St   Activity     Lower bound   Upper bound    Marginal
------ ------------ -- ------------- ------------- ------------- -------------
     1 x1           B              8             0
     2 x2           B              4             0
     3 x3           NL             0             0                   -0.166667

Karush-Kuhn-Tucker optimality conditions:

KKT.PE: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

KKT.PB: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

KKT.DE: max.abs.err = 0.00e+00 on column 0
        max.rel.err = 0.00e+00 on column 0
        High quality

KKT.DB: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

End of output
```

The output of GLPK is the same with implement, but it is faster than my own implement