Problem Solving 문제해결기법

2015 Spring Semester

Jinkyu Lee

Dept. of Computer Science and Engineering,
Sungkyunkwan University (SKKU), Republic of Korea

Contents

- Homework 6
- Chapter 12. Grid

Homework 6

- Announcement: 2015/5/26
- Due date: 2015/6/5 23:59
- 100 points
- 1 coding problem and 2 non-coding problems
- Last homework!

Elevator Optimization2

- Read the elevator optimization problem in Pages 253—256. We change the problem such that people **can walk up but cannot walk down.**

Elevator Optimization2

- Sample Input

5 (number of people <= MAX_RIDER=50)
3 (number of stops)
3 (the floor on which Person 1's home is located <= NFLOORS=110)
16 (the floor on which Person 2's home is located <= NFLOORS=110)
2 (the floor on which Person 3's home is located <= NFLOORS=110)
10 (the floor on which Person 4's home is located <= NFLOORS=110)
15 (the floor on which Person 5's home is located <= NFLOORS=110)

- Sample Output

3 (the floor on which Person 1 gets off)
16 (the floor on which Person 2 gets off)
3 (the floor on which Person 3 gets off)
10 (the floor on which Person 4 gets off)
16 (the floor on which Person 5 gets off)

Sequence Rearrangement

- For given two binary strings, we want to make them the same by cutting and pasting a string. For example, 0010011 can be cut as 00, 1, 00 and 11. Then, we can paste the 4 pieces, yielding 1110000.

- Develop a way to find the minimum number of cutting to make two strings the same in an efficient manner. (Assume that there is a way to make the two strings the same.)

Sequence Rearrangement

- Explain your algorithm.
- Explain your algorithm with 111100101 and 111001011.

Land Lease

- You are going to lease a piece of land. For each unit of land, you should pay X amount of fee, and you will earn some money from the land.
- Develop an algorithm that finds a rectangle maximizing your profit.
- Example: $X=100$,

110	130	80	170	160
150	120	170	160	90
110	190	180	120	30
10	80	110	130	140
120	130	30	120	90
60	120	80	100	50

Land Lease

- *Then, your maximum profit is 490, when the rectangle (1,1) – (3,4) (colored as grey) is chosen. In this case, the amount of profit is*
- *110+130+80+170+150+120+170+160+110+190+180+120 – 100*12= 490*

110	130	80	170	160
150	120	170	160	90
110	190	180	120	30
10	80	110	130	140
120	130	30	120	90
60	120	80	100	50

Land Lease

- Explain your algorithm.
- Explain your algorithm with the following example with X=150.

210	130	80	170	160
150	120	170	160	90
210	190	180	120	30
10	80	110	130	140
120	130	230	120	90
60	120	80	100	250

Contents

- Homework 6
- **Chapter 12 – Grids**

Contents

- **Program Design Example**
  - **Problem Description**
  - **Plate Packing**
  - **Plate Weight**

From now on, slides are adapted from Prof. Chang Wook Ahn’ s slides.

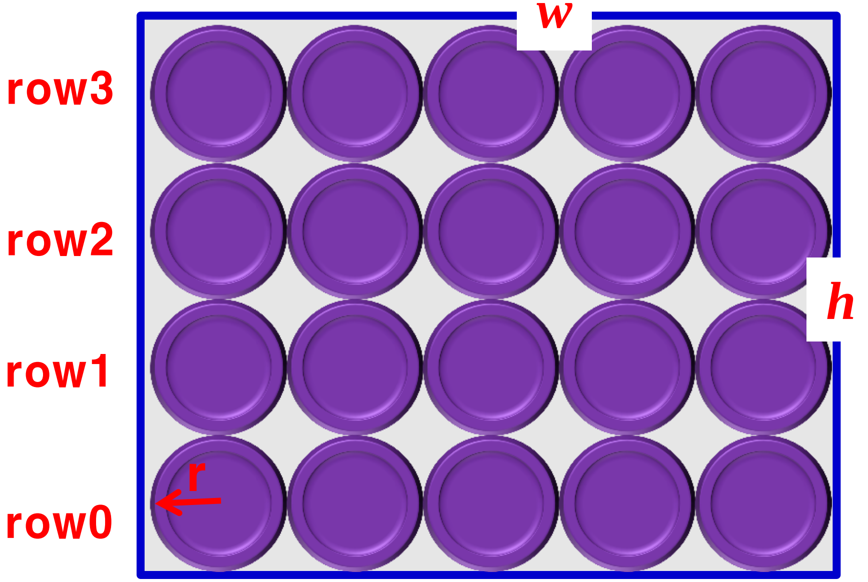
1. Problem Description

Description of Plate Packing Problem

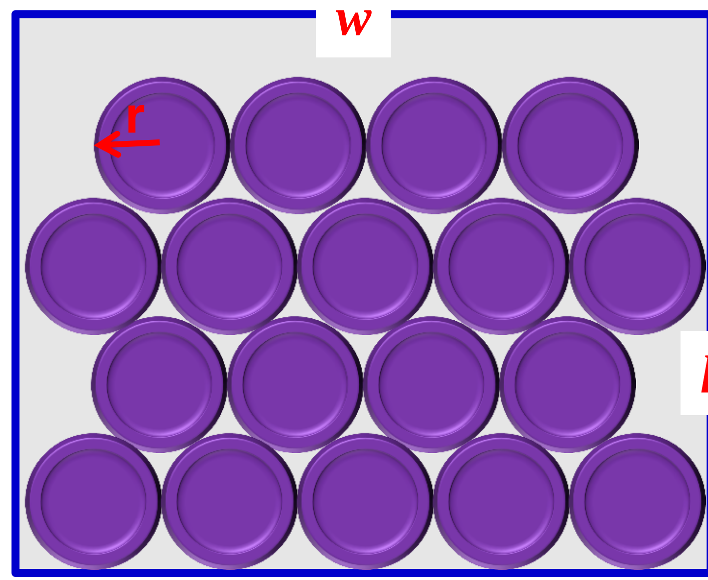
- A manufacturer seeks to enter the competitive campus dining hall market.
- Dining halls only buy plates in a single standard size.
- The company seeks an edge in the market through its unique packing method
- The company tries to pack the plates as many as possible, not to be broken
  The packing box size: horizontal *w* and vertical *h*, The palate radius: *r*
- Question 1) Which packing method should be chosen?
- Question 2) How many plates can be packed by the method?
- Question 3) How many plates can be placing on the top of any given plate?

2. Plate Packing (1)

- **Question 1) & 2) Which packing method? How many plates packed?**
  - **Consider two methods!: 1) Rectangular lattices, 2) Hexagonal lattices**



Method 1: Rectangular



Method 2: Hexagonal

L = num. of layers

P=num. of plates per layer

T = total num. of plates

$$L = \lfloor h/2r \rfloor$$

$$P = \lfloor w/2r \rfloor$$

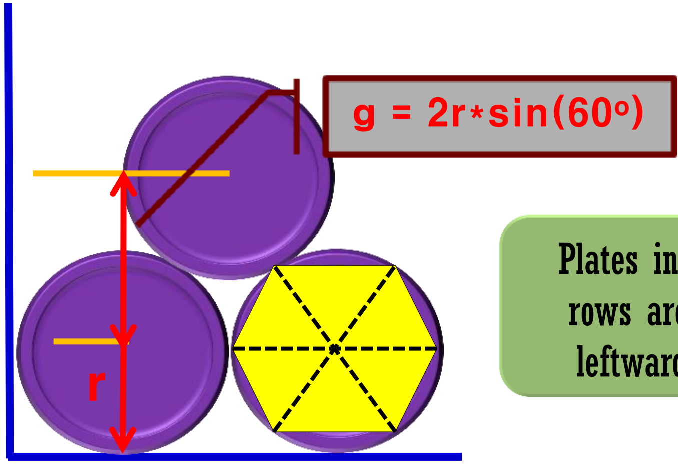
$$T = \lfloor h/2r \rfloor \times \lfloor h/2r \rfloor$$

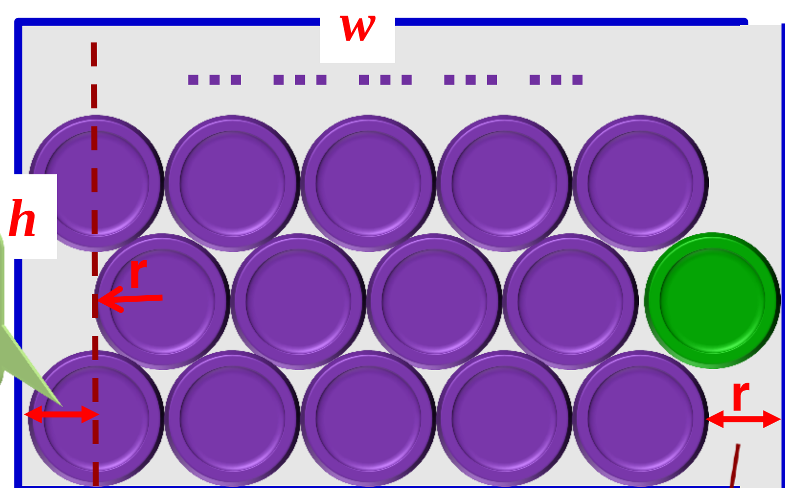
L = ?

P = ?

T = ?

2. Plate Packing (2)





Unless the remaining space in the even rows is greater than r, #plates in the odd rows is less than… by 1!

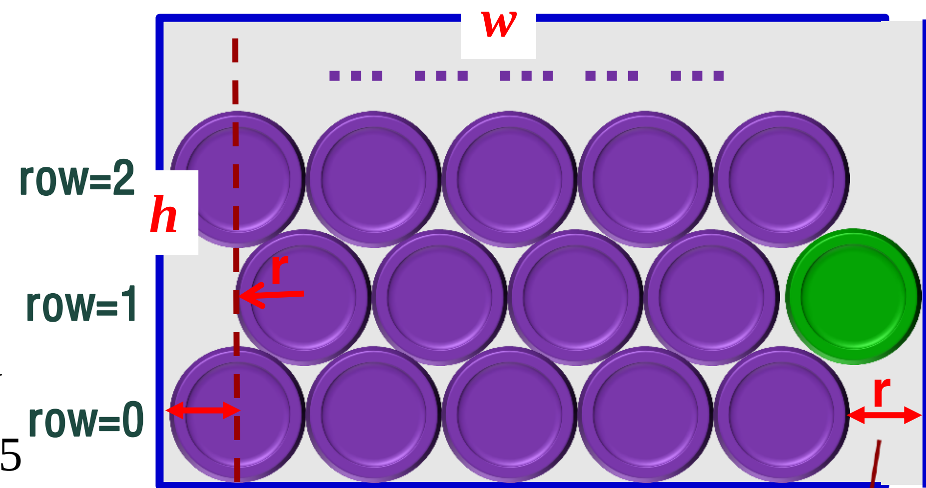
$$L = (1 + \lfloor (h-2r)/g \rfloor) = \left(1 + \left\lfloor \frac{h-2r}{2r\cdot(\sqrt{3}/2)} \right\rfloor\right)$$

```
int dense_layers(double w, double h, double r)
{
        double gap;

        if ((2*r) > h) return(0);

        gap = 2.0 * r * (sqrt(3)/2.0);
        return( 1 + floor((h-2.0*r)/gap) );
}
```

$$P = \begin{cases} \lfloor w/2r \rfloor, & \text{for even-numbered row} \\ \lfloor w/2r \rfloor, & \text{for odd-numbered row} \\ & \quad \& \, (w/2r) - \lfloor w/2r \rfloor \geq 0.5 \\ \lfloor w/2r \rfloor - 1, & \text{for odd-numbered row} \\ & \quad \& \, (w/2r) - \lfloor w/2r \rfloor < 0.5 \end{cases}$$

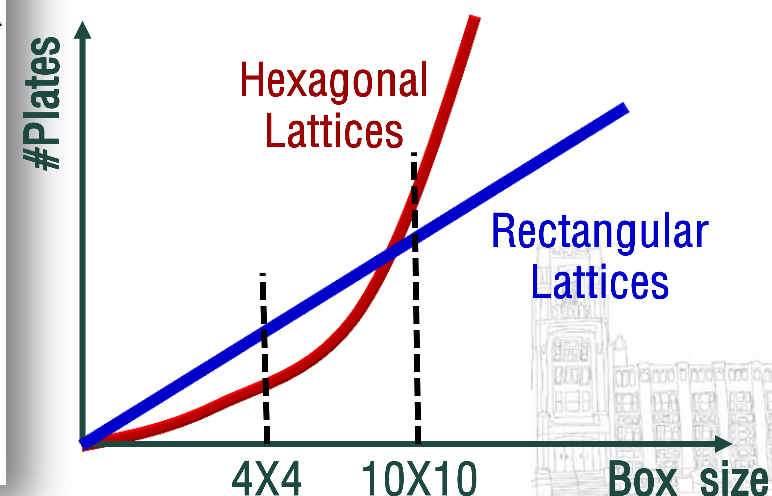
2. Plate Packing (3)

$$P=\begin{cases}\lfloor w/2r\rfloor, & \text{for even-numbered row}\\[2ex] \lfloor w/2r\rfloor, & \text{for odd-numbered row}\\ & \quad \&\,(w/2r)-\lfloor w/2r\rfloor\geq 0.5\\ \lfloor w/2r\rfloor-1, & \text{for odd-numbered row}\\ & \quad \&\,(w/2r)-\lfloor w/2r\rfloor< 0.5\end{cases}$$



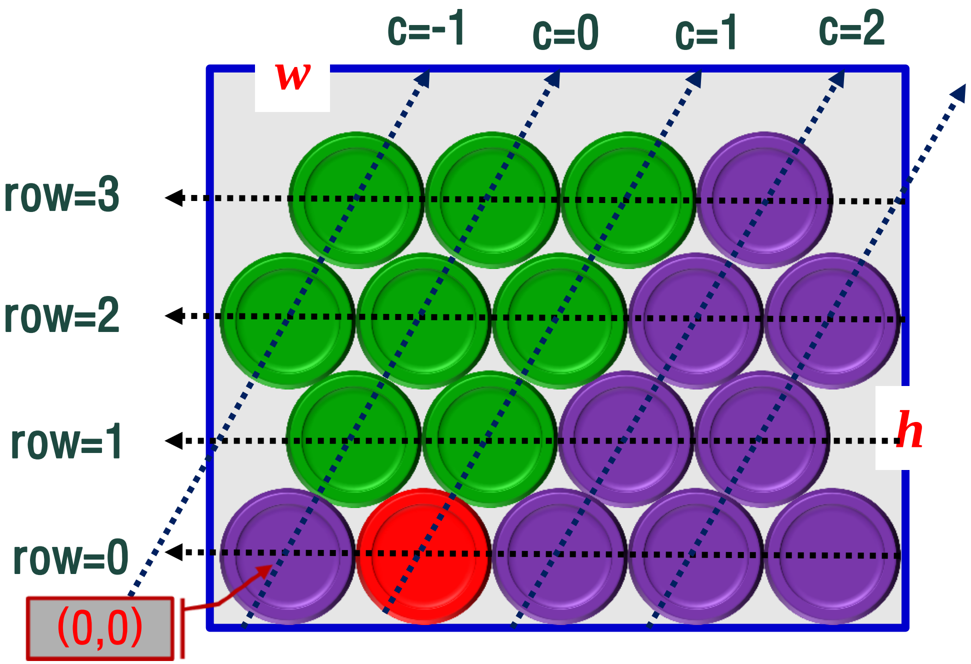
Unless the remaining space in the even ...ws is greater than r, #plates in the odd rows is less than… by 1!

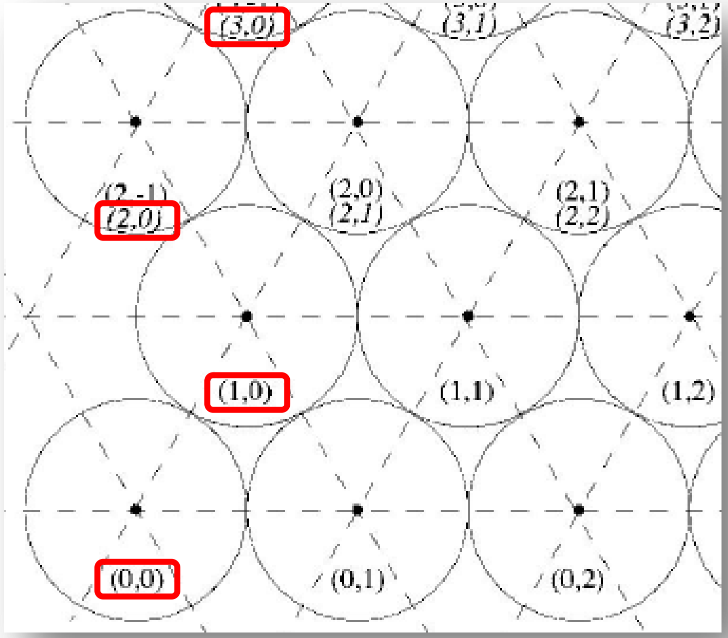
```
int plates_per_row(int row, double w, double r)
{
        int plates_per_full_row;

        plates_per_full_row = floor(w/(2*r));

        if ((row % 2) == 0) return(plates_per_full_row);

        if (((w/(2*r))-plates_per_full_row) >= 0.5)
                return(plates_per_full_row);
        else
                return(plates_per_full_row - 1);
}
```



3. Plate Weight (1)

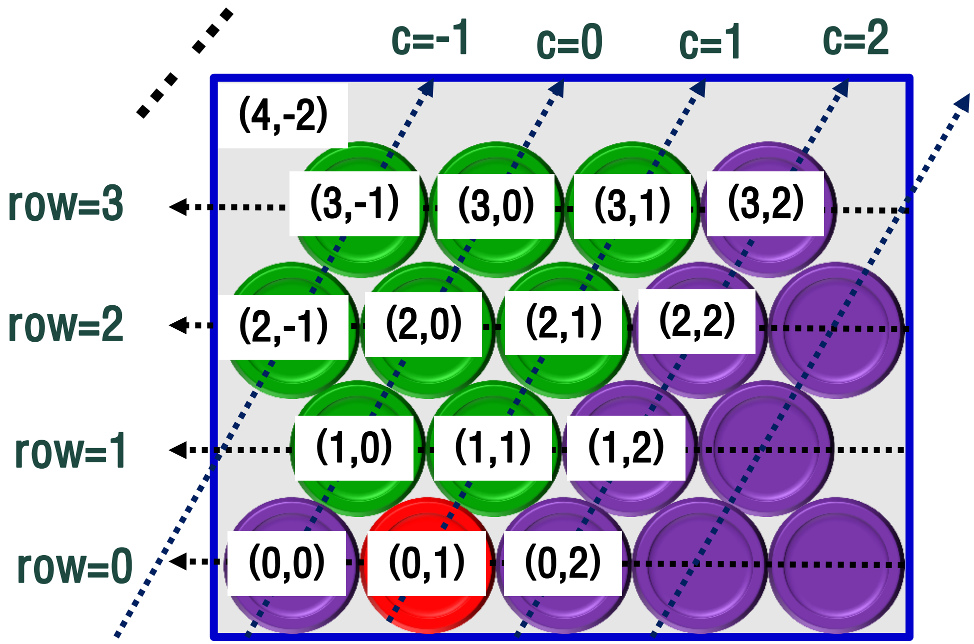
- **Question 3) How many plates can be placing on the top of any given plate?**
  - **Consider the hexagonal method! (Very trivial for the rectangular method)**





- The red plate only receives monotonic loads from the green plates.
- In the hexagonal axis, two plates piling up the plate at (k,c) are at (k+1,c-1) & (k+1,c).
- In the (k+i)the row, #plates that give the loads to the plate at (k,c) is i+1.
- But, such #plates is dependent on the boundary of box.
- To this end, the hexagonal coordinate is transformed into the array coordinate.

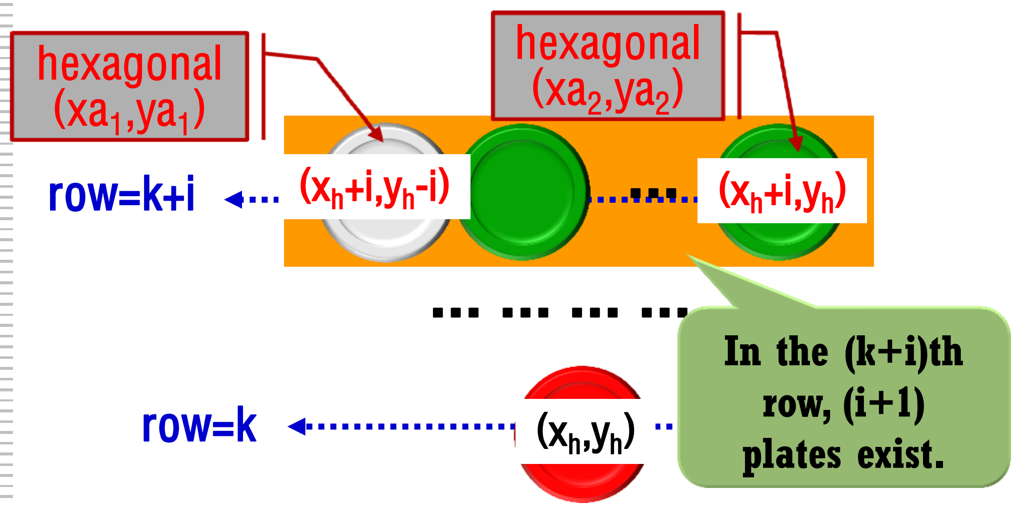
3. Plate Weight (2)



- In the hexagonal one, a pattern on the increment of **negative numbers** exists.
- The following rule can be obtained.

$$x_a = x_h$$
$$y_a = y_h + (x_h - \lceil x_h / 2 \rceil)$$

(2,-1)→(2,0), (2,0)→(2,1), ⋯
(3,-1)→(3,0), (3,0)→(3,1), ⋯
(4,-2)→(4,0), (4,-1)→(4,1), ⋯



if $(ya_1<0)$ $ya_1 = 0$;
If $(ya_2>\text{row\_length})$ $ya_2 = \text{row\_length}$;

- In the **(k+i)th row**, #plates giving the loads the plate at (x_h,y_h) becomes ya_2-ya_1+1.
- Thus, the **total load** is computed by all the plates w.r.t. all the rows.

3. Plate Weight (3)

```
int plates_on_top(int xh, int yh, double w, double l, double r)
{
        int number_on_top = 0;          /* total plates on top */
        int layers;                     /* number of rows in grid */
        int rowlength;                  /* number of plates in row */
        int row;                        /* counter */
        int xla,yla,xra,yra;            /* array coordinates */

        layers = dense_layers(w,l,r);

        for (row=xh+1; row<layers; row++) {
            rowlength = plates_per_row(row,w,r) - 1;

            hex_to_array(row,yh-row,&xla,&yla);
            if (yla < 0) yla = 0;                 /* left boundary */

            hex_to_array(row,yh,&xra,&yra);
            if (yra > rowlength) yra = rowlength;   /* right boundary */

            number_on_top += yra-yla+1;
        }

        return(number_on_top);
}
```

```
hex_to_array(int xh, int yh, int *xa, int *ya)
{
        *xa = xh;
        *ya = yh + xh - ceil(xh/2.0);
}
```

Problems in Chapter 12

- Ant on a chessboard
- The monocycle
- Star
- Bee Maja
- Robbery
- (2/3/4)-D Sqr/Rects/Cubes/Boxes?
- Dermuba triangle
- Airlines