National Taiwan University of Science and Technology
Department of Electrical Engineering

Software Development for Electronic Design Automation, Spring 2024

Programming Assignment #3
Layout Decomposition for Triple Patterning Lithography

(due May 31, 2024 (Friday) on-line)

1. Problem Description

Litho-etch (LE) triple patterning lithography (TPL) has become one of major techniques to enhance lithography resolution during semiconductor manufacturing for sub-20nm technology nodes. For realizing TPL, layout decomposition is a must step, in which each layout pattern is assigned to one of the three masks such that patterns on each mask conforms to the mask spacing rule. A layout decomposition example is illustrated in Figure 1. For a contact/via on the coordinate (x, y), the neighboring contacts/vias within the region $(x \pm 2, y \pm 2)$ are conflicting with it (within the mask spacing rule). For the layout shown in Figure 1(a), a conflict graph can be constructed (see Figure 1(b)), where each layout pattern is represented by a vertex, and two vertices is connected with an edge if they are within the conflicting region of each other. Having the conflict graph, the layout decomposition problem can be transformed into a 3-coloring problem; that is, we find a 3-coloring solution on the conflict graph such that any pair of connected vertices can be in different colors as many as possible. Figures 1(c) and 1(d) respectively show the 3-coloring solution of the conflict graph and the corresponding mask assignment solution for the input layout, where 1 coloring conflict exists while it is the optimal solution.

In this problem, you are asked to generate a TPL layout decomposition result for an input layout such that the number of coloring conflicts is minimized. **Note that stitch insertion is not considered to simplify the problem.**

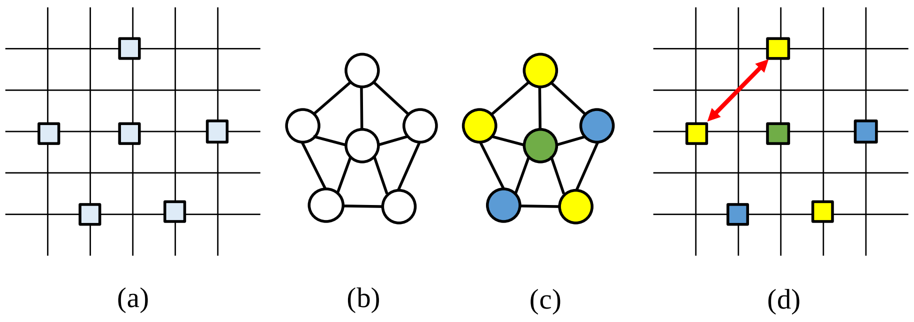

Figure 1: Layout decomposition in TPL. (a) An input contact/via layout. (b) The corresponding conflict graph. (c) The coloring result of the conflict graph. (d) The decomposition result.

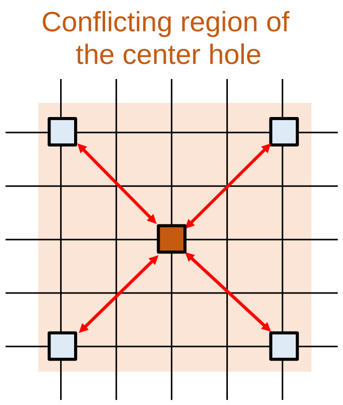

Figure 2: The conflicting region of the center contact/via.

2. Input

The input file format is illustrated, with comments in italics (these will not be in actual input files). The 1st line gives the dimension of the input layout, which indicates the number of grids in the x- and y-directions, and the 2nd line gives the number of contact/via patterns in the input layout. The following lines specify the x- and y-coordinates of each contact/via pattern. Patterns are sorted according to their IDs in the range [0, #patterns - 1]. The input file format is as follows:

> **layout # #** // *dimension of the input layout (numbers of grids in the x- and y-direction)*
> **num holes #** *// number of contact/via patterns*
> **hole_id x y** *// (x, y): x- and y-coordinates*
> **...**
> *//repeat for the appropriate number of patterns*

3. Output

The 1st line of an output file gives the number of coloring conflicts in the decomposition result you generate. The following lines give the mask number each layout pattern is assigned to. The three masks are distinguished by numbers “1”, “2”, and “3” respectively, and patterns are required to be sorted according to their IDs. The output format is shown as follows:

> **<num conflicts #>** // *number of conflicts in your decomposition result*
> **pattern_id <mask #>**
> **...**
> *// repeat for the appropriate number of patterns*

Sample input/output files:

The sample input file of the layout in Figure 1(a) and the sample output file of the mask assignment result in Figure 1(d) are shown below.

Sample Input	Sample Output
layout 5 5	**1**
num holes 6	**0 1**
0 0 2	**1 2**
1 1 0	**2 1**
2 3 0	**3 2**
3 4 2	**4 3**
4 2 2	**5 1**
5 2 4	

4. Language/Platform

(a) Language: C or C++.

(b) Platform: Unix/Linux.

5. Runtime Limitation

Your program must be terminated within 10 minutes. The dimension of the layout, will not be greater than 50x50, and the number of vias will not be greater than 300.

6. Command-line Parameter

In order to test your program, you are asked to add the following command-line parameters to your program (e.g., ./tpl 5_5_6.in 5_5_6.out):

[executable file name] [input file name] [output file name]

7. Submission

You need to submit the following materials in a compressed [student id]-p3.tgz file (e.g., b11007000-p3.tgz) at the course website by the deadline: (1) source codes, (2) Makefile, and (3) a text readme file (readme.txt) stating how to build and conduct your program. Please carefully read the following instructions:

- The compressed file [student id]-p3.tgz file contains only a single folder named [student id]-p3 (e.g., b11007000-p3). Use only lowercase letters for the compressed file and folder names.
- Only a compressed file in the *.tgz format will be accepted.
- Do not submit files or folders other than those specified above.
- Please ensure that your work can be successfully executed in the Linux environment.

**If the above requirements are not met, penalties will be imposed

8. Grading Policy

This programming assignment will be graded based on (1) the correctness of the output file format, (2) the correctness of the number of conflicts corresponding to the reported decomposition result, (3) solution quality (the number of coloring conflicts in the generated decomposition result), (4) runtime limitation, and (5) readme.txt. Please check these items before your submission.

9. Online Resources

Sample input files and the tutorial for CPLEX installation and usage can be found at the course website.