2003 IC/CAD Contest

Problem 7: Optimization on cell clustering

Source: Cadence Taiwan, Inc.

March 19, 2003, revised on March 24, 2003

1. Introduction

The cell clustering is a divide-and-conquer process that identifies repeated cell patterns in a physical layout database and groups them into new hierarchical cells. Figure 1 is an example that illustrates the process. At first, eight instances, A1~A4, and B1~B4, form a two-level design hierarchy and the TOP is the parent of them. Given that A1~A4 are of instance type A and B1~B4 of instance type B, the cell clustering can create a new instance type C and restructure a new design hierarchy as shown in Figure 1(b) where C1~C4 are of instance type C and the instance type C consists of A1 and B1. **In this contest problem, we call the instance type C an inserted dummy cell.** The advantages of cell clustering are two-folded. First, the cell clustering is in general toward a divide-and-conquer methodology. Introducing new hierarchical cells may reduce the amount of work for processing the parent cell. Second, clustering cells in a design will lead to a smaller design database if the repeated patterns are carefully selected.

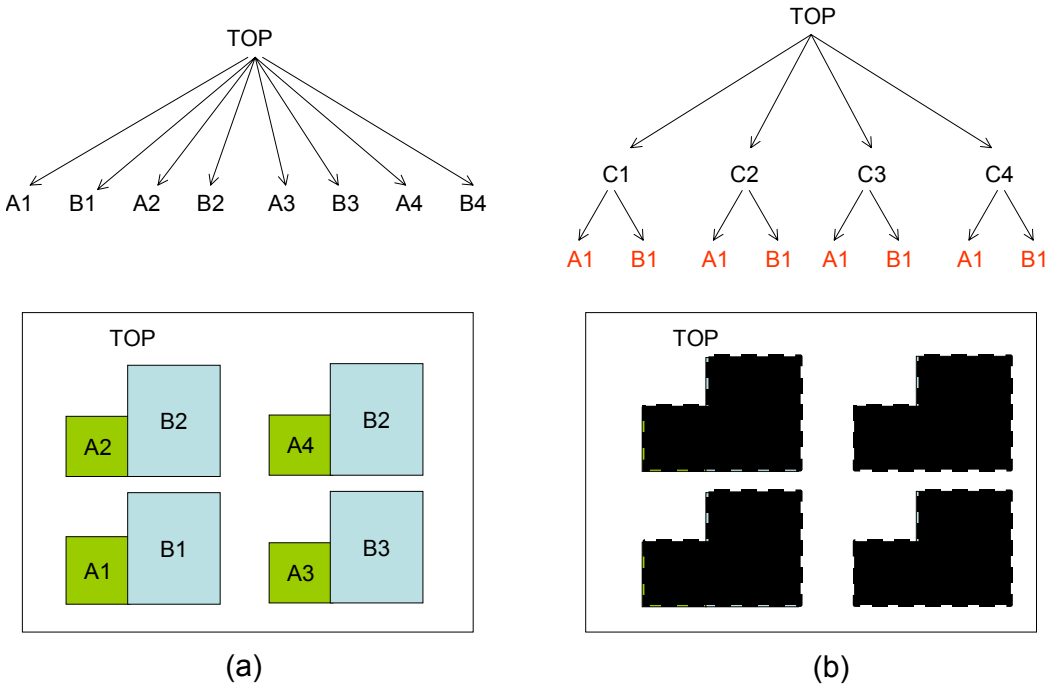

Figure 1 A cell clustering example

The objective of the problem is to properly cluster repeated patterns into new hierarchical cells so that the database size can be reduced at a minimum. To simplify the problem, the input is a two-level hierarchy with base cells. Your program should restructure the design hierarchy with hierarchical cells and produce output files to describe the resulting design hierarchy. The following sections will describe the input hierarchy and the format of output files in details.

2. Input

The input is a two-level hierarchy with base cells. Each base cell is a rectangular instance bounding boxes in the Cartesian coordinate space. The format of input file is: ([] means optional statement.)

INSTANCE COST:
<*the_storage_cost_of_instance*>
CELL COSTS:
[<*instance_type_name*> <*the_storage_cost_of_base_cell*>
]
[TOP#1/<*ITN*>#<*ON*> <*lower-left x*>,<*lower-left y*> <*upper-right x*>,<*upper-right y*>
]

where <*ITN*> instance type name and <*ON*> is the occurrence number. For example, the input file of Figure 1(a) can be described as:

INSTANCE COST:
5
CELL COSTS:
A 20
B 30
TOP 2
TOP#1/A#1 0,0 10,10
TOP#1/B#1 10,0 25,18
TOP#1/A#2 0,20 10,30
TOP#1/B#2 10,20 25,38
TOP#1/A#3 35,0 45,10
TOP#1/B#3 45,0 60,18
TOP#1/A#4 35,20 45,30
TOP#1/B#4 45,20 60,38

3. Output

Given an input file, your program should produce the output file in the following format:

INSTANCE COST:
<the_storage_cost_of_instance>
CELL COSTS:
[*<instance_type_name>* *<the_storage_cost_of_base_cell>*
]
[TOP#1/*<ITN>*#*<ON>* *<lower-left x>*,*<lower-left y>* *<upper-right x>*,*<upper-right y>*
]

For example, the outfile of Figure 1(b) can be described as:

INSTANCE COST:
5
CELL COSTS:
A 20
B 30
TOP 2
C 1
TOP#1/C#1/A#1 0,0 10,10
TOP#1/C#1/B#1 10,0 25,18
TOP#1/C#2/A#1 0,20 10,30
TOP#1/C#2/B#1 10,20 25,38
TOP#1/C#3/A#1 35, 0 45,10
TOP#1/C#3/B#1 45,20 60,18
TOP#1/C#4/A#1 35,20 45,30
TOP#1/C#4/B#1 45,20 60,38

4. Language/Platform

1. Language: C and C++
2. Platform: Sun OS/Solaris or PC Windows

5. Evaluation

The evaluation will based on:

- CPU time
- Storage cost of the output file
  - The total storage cost is equal to:
    the_storage_cost_of_instance×(the number of instances)+

    $\sum_i$(the storage cost of instance type T_i)

    Note, the cell cost for each inserted dummy cell is always 1.

  For instance, the storage cost of Figure 1(a) is equal to 5×8+(20+30+2)=92 and that of Figure 1(b) equal to 5×6+(20+30+2+1)=83.

6. Questions

Please report any question regarding to this problem to cad@cs.nthu.edu.tw with the email subject "CAD Contest: Problem x." Your question(s) will be answered in two weeks, and the Q&A's will be posted at the contest Web site.