CAD for VLSI Design

PA4 Report

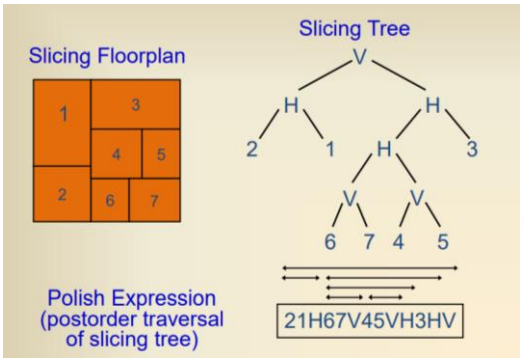
Student: 黃柏燁
Student ID: 109521018

I. Introduction

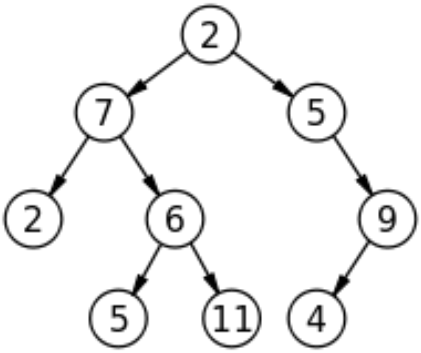
本次作業要用 C/C++設計 Simulate-Annealing based Slicing Floorplan Algorithm 去解決 Rectangle Packing Problem。所有 Module 皆為 Soft Module，須將長寬 ratio 皆控制在 0.5~2 之間，Floorplan 的結果不能有 Module 互相 Overlapping，使用設計的 Floorplan Algorithm 去找出最小的面積。

II. Data Structure

在處理 Floorplan 的問題時，需要將 Floorplan 轉化成 Slicing Tree，如圖(1)所示。

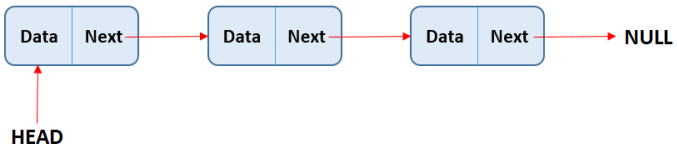


圖(1)



圖(2)

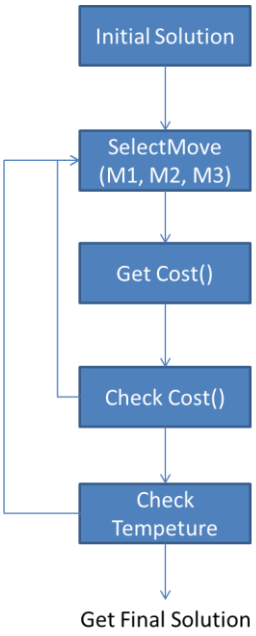
所以此次作業我採用 Binary Tree 來存取 Slicing Tree，如圖(2)所示。
再來是由於題目的所有 Module 皆為 Soft Module，故需要考慮不同長寬的 Module，所以我又再採用 Linked-list，如圖(3)所示，用來存取不同的長寬 ratio 的 Module。



圖(3)

III. Algorithms

我在這次作業實現的是由 D. F. Wong 以及 C. L. Liu 提出的 SA-based Floorplan Algorithm，圖(4)為 Flow Chart。



圖(4)

IV. Makefile

```
# Compiler
CXX = g++

# Path
SRC_PATH = .
BUILD_PATH = build

# Executable
EXE = go

# Source
SOURCES = $(SRC_PATH)/PA4_109521018.cpp  $(SRC_PATH)/linkedlist.cpp $(SRC_PATH)/binarytree.cpp
OBJECTS = $(BUILD_PATH)/PA4_109521018.o $(BUILD_PATH)/linkedlist.o $(BUILD_PATH)/binarytree.o

# Compiler flags
CXXFLAG = -O3 -Wall
INCLUDE = -I$(SRC_PATH)

# Make-command list
.PHONY: all run clean

# Target: Dependencies
#       Shell-command
all: $(BUILD_PATH) $(EXE)


run: $(EXE)
        ./go $(INPUT) $(OUTPUT)

clean:
        @echo "Removing objects"
        @rm -rf $(BUILD_PATH)
        @echo "Removing executable file"
        @rm -rf $(EXE)

$(EXE): $(OBJECTS)
        @echo "Generating executable file: $^ -> $@"
        @$(CXX) $^ -o $@

$(BUILD_PATH)/%.o: $(SRC_PATH)/%.cpp
        @echo "Compiling: $< -> $@"
        @$(CXX) -std=c++11 $(CXXFLAGS) $(INCLUDE) -c $< -o $@

$(BUILD_PATH):

        @echo "Creating object directory"
        @mkdir -p $@
```

V. Hardest part of the assignment

此次作業我認為最困難的就是實現一個 Binary Tree 了，我原本認為 Binary Tree 是 Linked-list 多一個 pointer 而已，但是在實作上，光是要長成一個 Tree 就很困難了，因為多了 Left Child 以及 Right Child，所以讓我在思考怎麼移動時真的下足了苦功，雖然最後寫出來的程式輸出的結果並未非常理想，但是難完成這份作業我也覺得我在撰寫程式的部份同時成長了不少。