University of Toronto, Faculty of Applied Science and Engineering
Department of Electrical and Computer Engineering

ECE 1387 - CAD for Digital Circuit Synthesis and Layout

Assignment #4 - Technology Mapping via Dynamic Programming

April 2010 J. Anderson
(Orig. version: J. Rose)

Assignment Date: April 13, 2010
Due Date: May 5, 2010, 11:59 PM (by email to instructor)
Late Penalty: **-1 per day late, with total marks available = 10**

You are to write an implementation of the dynamic programming-based standard cell technology mapping algorithm described in class. This is a simplified version of the general case.

Assume that the input subject network is a tree that has already been decomposed into 2-input NAND gates and inverters. The input library of gates will also be given as an already decomposed network of 2-input NANDs and inverters. The output of your program should be a network of gates with **minimum** total cost. You **will not** need to implement graphics for this assignment.

Your program should use the following input format, which defines the subject network (the circuit being technology mapped) and the library elements (the gates in the standard cell library). The first line of the file describes the record type. There are two kinds of records: 1) a record defining the subject network, and, 2) records defining the gates. There is only one subject network record. The record description line indicates the type of record, and the inputs and output. All fields are integers.

RecordType [GateNum] [Cost] In1 In2 ... InN -1 Out

RecordType is 1 for the subject network and 2 for a library (gate) element. GateNum and Cost are only present for library elements. GateNum is a positive number that represents the number of the gate. Each gate has a unique number, and these numbers are used in the output netlist produced by your program, as described below. Cost is area cost of the gate, in the case of library record. The In#'s give the numbers of the nets that are the inputs to either the subject network or the gate, and Out is the output net number of the subject network or gate. The list of record descriptors is terminated by a RecordType of -1.

You should assume that the subject and gate networks are trees (i.e., all nodes are single-fanout). Following each record descriptor is a set of lines that define the tree of either the subject network or gate network, with each line having the following form:

NodeNum NodeType NetNum1 NetNum2 [NetNum3]

NodeNum is a unique number of a node. Note that NodeNum is unique **only** within the network, and is **not** unique between the subject network and gate networks. NodeType is either 1 (for an inverter) or 2 (for a 2-input NAND gate). For the inverter, NetNum1 is the number of the net attached to the input, and NetNum2 is the number of the net attached to the output, and NetNum3 is not present. For the 2-input NAND, NetNum1 and NetNum2 are the numbers of the nets attached to the inputs, and NetNum3 is the number of the net attached to the output. The list of nodes is terminated with a NodeNum of -1.

Here is an example:

File Contents	*Meaning*
1 1 2 3 4 -1 9	**Subject Network (SN), inputs 1,2,3,4, output 9**
1 2 1 2 5	**SN - nand2 - inputs 1,2 output 5**
2 2 3 4 6	**SN - nand2 - inputs 3,4 output 6**
3 1 5 7	**SN - inverter - input 5, output 7**
4 1 6 8	**SN - inverter - input 6, output 8**
5 2 7 8 9	**SN - nand2 - inputs 7,8, output 9**
-1	**End of subject network**
2 1 5 1 -1 2	**Gate Network (GN), (inverter) Gate #1, cost=5, input 1, output 2**
1 1 1 2	**GN – inverter - input 1, output 2**
-1	**End of gate network**
2 2 10 1 2 -1 3	**New GN, (nand2) Gate #2, cost=10, inputs 1,2, output 3**
1 2 1 2 3	**GN - nand2 - inputs 1, 2, output 3**
-1	**End of gate network**
2 3 12 1 2 -1 4	**New GN, (and2) Gate #3, cost=12, inputs 1,2, output 4**
1 2 1 2 3	**GN - nand2 - inputs 1,2, output 3**
2 1 3 4	**GN - inverter - input 3, output 4**
-1	**End of gate network**
2 4 12 1 2 -1 5	**New GN, (or2) Gate #4, cost=12, inputs 1,2, output 5**
1 1 1 3	**GN - inverter - input 1, output 3**
2 1 2 4	**GN - inverter - input 2, output 4**
3 2 3 4 5	**GN - nand2 - inputs 3,4, output 5**
-1	**End of gate network**
-1	**End of record descriptors (end of file)**

Your program should produce text output that shows the technology mapping solution in a similar manner to the network descriptor files:

NodeNum GateNum NetNum1 Netnum2 .... NetNumN

Where **NodeNum** is a unique Gate identifier and **GateNum** is one of the gate numbers given in the gate record descriptor above (e.g., or2 is GateNum 4) and NetNum1 ... NetNum(N-1) are the numbers of the nets attached to the gate inputs, and NetNumN is the output net number. For example, one possible mapping of the network above is:

1 3 1 2 5	**NodeNum 1 of GateNum #3 (and2), inputs 1,2, output 5**
2 3 3 4 6	**NodeNum 2 of GateNum #3 (and2), inputs 3,4, output 6**
3 2 5 6 7	**NodeNum 3 of GateNum #2 (nand2), inputs 5,6, output 7**

You should test your program on the test files provided on the course web page.

What to hand in:

1. The location of the executable and source code, which should be on one of the EECG networks (ECF is fine too). Provide instructions on how to run the program. Please check that the file and directory permissions allow me to execute your program.

2. Include a short description (< 2 pages) of the flow of your program and its main routines, assuming that I already have a basic knowledge of the dynamic programming-based technology mapping algorithm. Explain how you implemented tree matching. Indicate any enhancements to the basic algorithm that you have made.

3. You should also hand in the solution to each of the test files, **in the format described above** (with the exception of cct4 owing to its size). Provide a table showing the cost of the minimum cost mapping for each test file. Specifically, show two costs for each test file: 1) *without* consideration of input pin permutations on the gates; 2) *with* consideration of input pin permutations.