L7b: Rule-based optimization
Reading material


Main Idea

• The circuit is optimized by step-wise refinement
• Local transformations based on a set of replacement rules are applied at each step
• The replacement rules are generated with the help of NPN (Negation-Permutation-Negation) classes of Boolean functions
• The objective is to minimize the number of gates in the circuit
NPN classes

- Each NPN class consists of all functions which differ by:
  - Negation of some input variables \( x_1, \ldots, x_n \) and/or,
  - Permutation of some input variables \( x_1, \ldots, x_n \) and/or,
  - Negation of the function output.
Replacement rules

- Functions belonging to the same NPN class have a minimum AND/INVERTER graph of the same size.
- Rules on how to transform a non-minimum graph to a minimum need to be defined.
- Rules for removing redundancies also need to be defined.
Rules are derived from axioms and properties of Boolean algebra

A3: \( \forall a,b,c \in B, \ a \cdot (b+c) = a \cdot b + a \cdot c, \)
\( \ a + b \cdot c = (a+b) \cdot (a+c) \)

A4: \( \forall a \in B, \ a \cdot 1 = a, \ a + 0 = a \)

A5: \( \forall a \in B, \ a \cdot a' = 0, \ a + a' = 1 \)

P1: \( \forall a \in B, \ (a')' = a \)

P2: \( \forall a,b,c \in B, \ a \cdot (b \cdot c) = (a \cdot b) \cdot c, \ (a + b) + c = a + (b + c) \)

P3: \( \forall a \in B, \ a \cdot 0 = 0, \ a + 1 = 1 \)

P5: \( \forall a,b \in B, \ a \cdot (a+b) = a, \ a + a \cdot b = a \)

P6: \( \forall a \in B, \ a \cdot a = a, \ a + a = a \)
Completeness of rules

- By using axioms Boolean algebra, we can derive any Boolean expression representing a given Boolean function.
- Therefore, by applying them as rules, we can obtain any circuit for any circuit implementing the same function.
- Consequently, we can find a minimal circuit starting from any circuit if we search long enough.
Simulated Annealing Algorithm

- The order of applying rules can be controlled by a simulated annealing algorithm.
- A \textit{cost} function assigns a cost to each state $s$ of the search space $S$.
- The set of \textit{neighbours} is the set of states which can be obtained from $s \in S$ by a single move.
- Two functions are assigned to states:
  - Selection probability
  - Acceptance probability
Simulated Annealing, cont

- Let $S$ be a set of all possible combinational acyclic Boolean circuits implementing a Boolean function $f$
- At each step, the annealing algorithm considers some neighbor $s' \in S$ of the current state $s \in S$, and probabilistically decides between moving to state $s'$ of staying in state $s'$
- The probabilities are selected so that the system ultimately tends to move to state the lower cost
Neighbours

• In our case, the neighbours of a state s are the states which can be obtained from s by applying one of the rules listed on p. 7

• There are also a rules which is related specifically to circuits:
  – Two isomorphic vertices can be merged into one
  – A vertex with a multiple fan-out can be divided into two vertices
Pseudo-code of annealing algorithm

```
algorithm ANNEALING(c_1, \varepsilon);
    best_c = c_1;
    best_cost = \varepsilon(c_1);
    (t_b, t_e, \Delta t, N_{of\_moves}) = INITIALIZE(c_1, \varepsilon);
    t = t_b;
    while t > t_e do
        for N_{of\_moves} do
            c_2 = SELECT(c_1);
            if ACCEPT(\varepsilon(c_1), \varepsilon(c_2), t) then
                c_1 = c_2;
                cost = \varepsilon(c_2);
                if cost < best_cost then
                    best_c = c_1;
                    best_cost = cost;
            end
        end
        t = t - \Delta t;
    end
```
The annealing schedule

- As the simulation proceeds, the temperature $t$ is gradually reduced
  - Initially, $t$ is set to a high value
  - Then, it is decreased at each step according to some annealing schedule
  - Finally, it becomes 0

- In the way, the system is expected to initially explore a broad region of the search space containing good solutions

- Then, it narrows and at $t = 0$ moves downhill
Convergence to optimum

- If the annealing process is scheduled correctly, it converges to optimum solution asymptotically.
- In general, annealing can smoothly trade-off complex, multi-dimensional objective functions:
  - Area, delay, power, etc.