Example:
\[ t_1 = a + bc; \]
\[ t_2 = d + e; \]
\[ t_3 = ab + d; \]
\[ t_4 = t_1 t_2 + fg; \]
\[ t_5 = t_4 h + t_2 t_3; \]
\[ F = t_5'; \]

This shows an unoptimized set of logic equations consisting of 17 literals.
Optimized equations

Using technology independent optimization, these equations are optimized using only 13 literals:

\[ t_1 = d + e; \]
\[ t_2 = b + h; \]
\[ t_3 = at_2 + c; \]
\[ t_4 = t_1t_3 + fgh; \]
\[ F = t_4' \]

Implement this network using a set of gates which form a library. Each gate has a cost (i.e. its area, delay, etc.)
Technology mapping

Two approaches:
- Rule-Based [LSS]
- Algoritmic [DAGON, MISII]
  - Represent each function of a network using a set of *base functions*. This representation is called the *subject graph*.
    - Typically the base is 2-input NANDs and inverters [MISII].
    - The set should be *functionally complete*.
  - Each gate of the library is likewise represented using the base set. This generates *pattern graphs*
    - Represent each gate in *all possible* ways

Subject graph

Subject graph of 2-input NANDs and invertors
Algorithmic approach

- A **cover** is a collection of pattern graphs such that
  - every node of the subject graph is contained in one (or more) pattern graphs
  - each input required by a pattern graph is actually an output of some other graph (i.e. the inputs of one gate must exist as outputs of other gates.)
- For minimum area, the cost of the cover is the sum of the areas of the gates in the cover.
- Technology mapping problem: *Find a minimum cost covering of the subject graph by choosing from the collection of pattern graphs for all the gates in the library.*

Subject graph

\[
\begin{align*}
t_1 &= d + e; \\
t_2 &= b + h; \\
t_3 &= at_2 + c; \\
t_4 &= t_1t_3 + fgh; \\
F &= t_4; \\
\end{align*}
\]
Pattern graphs for the IWLS library

- inv(1)
- nand(2)
- nor(2)
- aoi(1)
- xor(2)
- nand(2)
- nor(2)
- aoi(2)
- xor(2)

Subject graph covering

\[
\begin{align*}
  t_1 &= d + e; \\
  t_2 &= b + h; \\
  t_3 &= at_2 + c; \\
  t_4 &= t_1t_3 + fgh; \\
  F &= t_4'; \\
\end{align*}
\]

Total cost = 23
Better covering

\[ t_1 = d + e; \]
\[ t_2 = b + h; \]
\[ t_3 = a \cdot t_2 + c; \]
\[ t_4 = t_1 \cdot t_3 + f \cdot g \cdot h; \]
\[ F = t_4'; \]

Total area = 18

Alternate covering

\[ t_1 = d + e; \]
\[ t_2 = b + h; \]
\[ t_3 = a \cdot t_2 + c; \]
\[ t_4 = t_1 \cdot t_3 + f \cdot g \cdot h; \]
\[ F = t_4'; \]

Total area = 15
Tech. mapping using DAG covering

Input:
  – Technology independent, optimized logic network
  – Description of the gates in the library with their cost

Output:
  – Netlist of gates (from library) which minimizes total cost

General Approach:
  – Construct a subject DAG for the network
  – Represent each gate in the target library by pattern DAG’s
  – Find an optimal-cost covering of subject DAG using the collection of pattern DAG’s

Complexity of DAC covering

- Complexity of DAG covering:
  – NP-hard
  – Remains NP-hard even when the nodes have output degree ≤ 2
  – If subject DAG and pattern DAG’s are trees, an efficient algorithm exists