Multi-Level Optimization: Overview of Basic Transforms
(taken from Giovanni De Micheli, "Synthesis and Optimization of Digital Circuits" (McGraw Hill), ch. 8.2)
Basic Representation #1: Logic Network

Fig. 8.3(a)

Basic Representation #2: Logic Network Graph

Fig. 8.3(b)
Comparison: Logic Network vs. Logic Network Graph

Multi-Level Optimization Example: Initial Design
Transform #1: ELIMINATE (collapse)

Eliminate a node: “collapse” network structure

Fig. 8.3(a)

before

Transform #1: ELIMINATE (collapse)

Eliminate a node: “collapse” network structure

Fig. 8.3(a)

after
Transform #2: DECOMPOSE

Break 1 larger node into several smaller nodes

\[ v = a'd + bd + c'd + ae' \]

Before decompose(v)

\[ w \]

\[ r = p + a' \]

\[ s = r + b' \]

\[ x \]

\[ t = ac + ad + bc + bd + e \]

\[ y \]

\[ q = a + b \]

\[ u = q'c + qc' + qc \]

\[ z \]

Fig. 8.3(a) before

Transform #2: DECOMPOSE

Break 1 larger node into several smaller nodes

\[ j = a'b + b + e' \]

After decompose(v)

\[ w \]

\[ r = p + a' \]

\[ s = r + b' \]

\[ x \]

\[ t = ac + ad + bc + bd + e \]

\[ y \]

\[ q = a + b \]

\[ u = q + c \]

\[ z \]

Fig. 8.3(a) after
Transform #3: EXTRACTION

Create/extract “common subexpression” for 2 or more nodes

Before:

\[ v = a'd + bd + c'd + ae' \]

\[ p = ce + de \]

\[ t = ac + ad + bc + bd + e \]

\[ q = a + b \]

\[ u = q'c + qc' + qc \]

After:

\[ v = a'd + bd + c'd + ae' \]

\[ p = ke \]

\[ k = c + d \]

\[ t = ka + kb + e \]

\[ q = a + b \]

\[ u = q'c + qc' + qc \]

Fig. 8.3(a)
Perform optimization (usually Boolean) within a single node

Transform #4: SIMPLIFICATION

Fig. 8.3(a) before

Transform #4: SIMPLIFICATION

Fig. 8.3(a) after
Transform #5: SUBSTITUTION

Find an existing “common subexpression” for 1 or more nodes

before

after