Retiming

Lan-Da Van (范倫達), Ph. D.
Department of Computer Science
National Chiao Tung University
Taiwan, R.O.C.
Fall, 2010

ldvan@cs.nctu.edu.tw
http://www.cs.nctu.tw/~ldvan/
Outlines

- Introduction
- Definitions and Properties
- Solving Systems of Inequalities
- Retiming Techniques
- Conclusion
- Supplement
Introduction (1/2)

- Retiming is a transformation technique used to change the locations of delay elements in a circuit without affecting the input/output characteristics of the circuit.
- Critical path is defined to be the path with the longest computation time among all paths that contain zero delay.
- The lower bound on the clock period of the circuit can be achieved by retiming.
**Application of Retiming**

- Reduce the clock period.
- Reduce the number of registers.
- Reduce the power consumption and logic synthesis.

\[ w(n) = ay(n-1) + by(n-2) \]
\[ y(n) = w(n-1) + x(n) \]
\[ = ay(n-2) + by(n-3) + x(n) \]

3 u.t. -> 2 u.t.

\[ w1(n) = ay(n-1) \]
\[ w2(n) = by(n-2) \]
\[ y(n) = w1(n-1) + w2(n-1) + x(n) \]
\[ = ay(n-2) + by(n-3) + x(n) \]
Outlines

- Introduction
- *Definitions and Properties*
- Solving Systems of Inequalities
- Retiming Techniques
- Conclusion
- Supplement
Definitions

\( V, U : \text{node} \)
\( r(V) : \text{retiming value} \)
\( w(e) : \text{weight of the edge } e \text{ in } G \)
\( w_r(e) : \text{weight of the edge } e \text{ in } G_r \)

\[ w_r(e) = w(e) + r(V) - r(U) \]

ex: \( r(1)=0, r(2)=1, r(3)=0, r(4)=0 \)

\( w_r(3 \rightarrow 2) = w(3 \rightarrow 2) + r(2) - r(3) \)
\[ = 0 + 1 - 0 = 1 \]

\( w_r(2 \rightarrow 1) = w(2 \rightarrow 1) + r(1) - r(2) \)
\[ = 1 + 0 - 1 = 0 \]
Properties of Retiming

4.2.1 The weight of the retimed path \( p = V_0 \rightarrow V_1 \rightarrow \ldots \rightarrow V_k \) is given by \( w_r(p) = w(p) + r(V_k) - r(V_0) \).

4.2.2 Retiming does not change the number of delays in a cycle.

4.2.3 Retiming does not alter the iteration bound in a DFG.

4.2.4 Adding the constant value \( j \) to the retiming value of each node does not change the mapping from \( G \) to \( G_r \).
Property 4.2.1

The weight of the retimed path $p = V_0 \xrightarrow{e_0} V_1 \xrightarrow{e_1} \ldots \xrightarrow{e_{k-1}} V_k$ is given by $w_r(p) = w(p) + r(V_k) - r(V_0)$.

Proof

$$w_r(p) = \sum_{i=0}^{k-1} w_r(e_i)$$

$$= \sum_{i=0}^{k-1} (w(e_i) + r(V_{i+1}) - r(V_i))$$

$$= \sum_{i=0}^{k-1} w(e_i) + (\sum_{i=0}^{k-1} r(V_{i+1}) - \sum_{i=0}^{k-1} r(V_i))$$

$$= w(p) + r(V_k) - r(V_0)$$
Outlines

- Introduction
- Definitions and Properties
- *Solving Systems of Inequalities*
- Retiming Techniques
- Conclusion
- Supplement
Solving Systems of Inequalities

- **Step 1: Draw a constraint graph**
  1. Draw the node $i$ for each of the $N$ variables $r_i, i=1,2...,N$.
  2. Draw the node $N+1$.
  3. For each inequality $r_i - r_j \leq k$, draw the edge $j \rightarrow i$ from node $j$ to $i$ with length $k$.
  4. For each node $i, i=1,2...,N$, draw the edge $N+1 \rightarrow i$ from the node $N+1$ to $i$ with length 0.

- **Step 2: Solve using a shortest path algorithm**
  1. The system of inequalities has a solution if and only if the constraint graph contains no negative cycles.
  2. If a solution exists, one solution is where $r_i$ is the minimum-length path from the node $N+1$ to the node $i$. 
Example 4.3.1 with M=5, N=4 (1/2)

Given Inequalities:

\[
\begin{align*}
  r_1 - r_2 &\leq 0 \\
  r_3 - r_1 &\leq 5 \\
  r_4 - r_1 &\leq 4 \\
  r_4 - r_3 &\leq -1 \\
  r_3 - r_2 &\leq 2 
\end{align*}
\]

Step 1:

\[
\begin{array}{cccc}
  V=1 & 0 & 0 & 0 \\
  V=2 & 0 & 0 & 0 \\
  V=3 & 0 & 0 & 0 \\
  V=4 & 0 & -1 & -1 \\
  V=5 & 0 & 0 & 0 \\
\end{array}
\]

Step 2: Bellman-Ford algorithm

<table>
<thead>
<tr>
<th>( r^{(k)}(V) )</th>
<th>( r^{(1)} )</th>
<th>( r^{(2)} )</th>
<th>( r^{(3)} )</th>
<th>( r^{(4)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( V=1 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( V=2 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( V=3 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( V=4 )</td>
<td>0</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>( V=5 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Solution: \( r_1 = 0, \ r_2 = 0, \ r_3 = 0, \ r_4 = -1 \)

Bellman-Ford algorithm

\[
R^{(6)} = \begin{bmatrix}
\infty & \infty & 5 & 4 & \infty \\
0 & \infty & 2 & 1 & \infty \\
\infty & \infty & \infty & -1 & \infty \\
\infty & \infty & \infty & \infty & \infty \\
0 & 0 & 0 & -1 & \infty 
\end{bmatrix}
\]

Floyd-Warshall algorithm

Lan-Da Van
Example 4.3.1 with M=5, N=4 (2/2)

Alternative Computation for Floyd-Warshall algorithm

\[
R^{(1)} = \begin{bmatrix}
0 & \infty & 5 & 4 & \infty \\
0 & 0 & 2 & \infty & \infty \\
\infty & \infty & 0 & -1 & \infty \\
\infty & \infty & \infty & 0 & \infty \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
R^{(2)} = \begin{bmatrix}
0 & \infty & 5 & 4 & \infty \\
0 & 0 & 2 & 1 & \infty \\
\infty & \infty & 0 & -1 & \infty \\
\infty & \infty & \infty & 0 & \infty \\
0 & 0 & 0 & -1 & 0
\end{bmatrix}
\]

\[
R^{(3)} = \begin{bmatrix}
0 & \infty & 5 & 4 & \infty \\
0 & 0 & 2 & 1 & \infty \\
\infty & \infty & 0 & -1 & \infty \\
\infty & \infty & \infty & 0 & \infty \\
0 & 0 & 0 & -1 & 0
\end{bmatrix}
\]

\[
R^{(4)} = \begin{bmatrix}
0 & \infty & 5 & 4 & \infty \\
0 & 0 & 2 & 1 & \infty \\
\infty & \infty & 0 & -1 & \infty \\
\infty & \infty & \infty & 0 & \infty \\
0 & 0 & 0 & -1 & 0
\end{bmatrix}
\]

\[
R^{(5)} = \begin{bmatrix}
0 & \infty & 5 & 4 & \infty \\
0 & 0 & 2 & 1 & \infty \\
\infty & \infty & 0 & -1 & \infty \\
\infty & \infty & \infty & 0 & \infty \\
0 & 0 & 0 & -1 & 0
\end{bmatrix}
\]

\[
R^{(6)} = \begin{bmatrix}
\infty & \infty & 5 & 4 & \infty \\
0 & \infty & 2 & 1 & \infty \\
\infty & \infty & 0 & -1 & \infty \\
\infty & \infty & \infty & 0 & \infty \\
0 & 0 & 0 & -1 & \infty
\end{bmatrix}
\]

\[
r_{i,j}(m+1) = \min\{r_{i,k}(m) + r_{k,j}(1)\}
\]
Outlines

- Introduction
- Definitions and Properties
- Solving Systems of Inequalities
- Retiming Techniques
  - Cutset retiming and pipelining
  - Systolic transformation
  - Retiming for clock period minimization
- Conclusion
- Supplement
Cutset Retiming (1/2)

Definition: A cutset is a set of edges that can be removed from the graph to create 2 disconnected subgraphs.

Procedures:
- Step 1: Construct 2 disjointed sub-graphs $G_1$ and $G_2$
- Step 2: Add $k$ delays to each edge from $G_1$ to $G_2$ and remove $k$ delays from each edge from $G_2$ to $G_1$.

Feasible solution of cutset retiming
- $w_{r}(G_1 \rightarrow G_2) = w(G_1 \rightarrow G_2) + r(G_2) - r(G_1) = w(G_1 \rightarrow G_2) + k$
- $w_{r}(G_2 \rightarrow G_1) = w(G_2 \rightarrow G_1) + r(G_1) - r(G_2) = w(G_2 \rightarrow G_1) - k$
- $\Rightarrow -\min\{w(G_1 \rightarrow G_2)\} \leq k \leq \min\{w(G_2 \rightarrow G_1)\}$
Cutset Retiming (2/2)

- **Upper case**
  - Cutset = \{2\rightarrow 1, 3\rightarrow 2, 1\rightarrow 4\} and feasible solution: \(0 \leq k \leq 1\)

- **Lower case**
  - Cutset = \{2\rightarrow 1, 3\rightarrow 2, 4\rightarrow 2\} and feasible solution: \(0 \leq k \leq 1\)
Pipelining is a special case of cutset retiming where there are no edges in the cutset from $G_2$ to $G_1$, i.e., pipelining applies to graphs without loops.
Cutset Retiming and Slow-Down

Procedure:
- Replace each delay with N delays to create an N-slow version of the DFG.
- Perform cutset retiming on the N-slow DFG.

In (a), CP=$101 \oplus 2 \otimes$  
$T_{cp}=101 \times 1 + 2 \times 2 = 105$

In (c), CP=$2 \oplus 2 \otimes$  
$T_{cp}=2 \times 1 + 2 \times 2 = 6$
Systolic Transformation

Retiming Relationship

- Cutset Retiming
- Pipelining
- Systolic Transformation

VLSI Digital Signal Processing Systems
Retiming for Clock Period Minimization (1/2)

Definition:
- $\Phi(G) = \max\{t(p) : w(p)=0\}$ (min. feasible clk period)
- $W(U,V) = \min\{w(p) : U \rightarrow V\}$
- $D(U,V) = \max\{t(p) : U \rightarrow V \text{ and } w(p) = W(U,V)\}$

Algorithm for computing $W(U,V)$ and $D(U,V)$:
1. Let $M=\max_{n}$, where $\max_{n}$ is the maximum computation time of the nodes in $G$ and $n$ is the number of nodes in $G$.
2. Form a new graph $G'$ which is the same as $G$ except the edge weights are replaced by $w'(e)=Mw(e)-t(U)$ for all edges.
3. Solve the all-pairs shortest path problem on $G'$. Let $S'_{UV}$ be the shortest path from $U$ to $V$.
4. If $U \neq V$, then $W(U,V)=\text{ceiling}\left(\frac{S'_{UV}}{M}\right)$ and $D(U,V)=MW(U,V)-S'_{UV}+t(V)$. If $U=V$, then $W(U,V)=0$ and $D(U,V)=t(U)$. 
W(U,V) and D(U,V) are used to determine if there is a retiming solution that can achieve a desired clock period.

Given a desired clock period c, if:

1. feasibility constraint:
   \[ r(U) - r(V) \leq w(e) \text{ for every edge } e : U \rightarrow V \]

2. critical path constraint:
   \[ r(U) - r(V) \leq W(U,V) - 1 \text{ for all vertices } U,V \text{ in } G \text{ such that } D(U,V) > c. \]

\[ \Rightarrow \] There is a feasible retiming solution \( r \) such that \( \Phi(Gr) \leq c. \)
Example with $c=3$ (1/3)

Step 1: $t_{\text{max}}=2$, $n=4$, $M=t_{\text{max}}n=8$

Step 2: As $G'$ shows.

Step 3:

\[
\begin{array}{l|cccc}
S'_{UV} & 1 & 2 & 3 & 4 \\
1 & 12 & 5 & 7 & 15 \\
2 & 7 & 12 & 14 & 22 \\
3 & 5 & -2 & 12 & 20 \\
4 & 5 & -2 & 12 & 20 \\
\end{array}
\]

Step 4:

\[
\begin{array}{l|cccc}
W(U,V) & 1 & 2 & 3 & 4 \\
1 & 0 & 1 & 1 & 2 \\
2 & 1 & 0 & 2 & 3 \\
3 & 1 & 0 & 0 & 3 \\
4 & 1 & 0 & 2 & 0 \\
\end{array} \quad \begin{array}{l|cccc}
D(U,V) & 1 & 2 & 3 & 4 \\
1 & 1 & 4 & 3 & 3 \\
2 & 2 & 1 & 4 & 4 \\
3 & 4 & 3 & 2 & 6 \\
4 & 4 & 3 & 6 & 2 \\
\end{array}
\]
Example with c=3 (2/3)

Step 5: Constraints

- Feasibility constraint:
  \[ r(1) - r(3) \leq 1 \]
  \[ r(1) - r(4) \leq 2 \]
  \[ r(2) - r(1) \leq 1 \]
  \[ r(3) - r(2) \leq 0 \]
  \[ r(4) - r(2) \leq 0 \]

- Critical path constraint:
  \[ r(1) - r(2) \leq 0 \]
  \[ r(2) - r(3) \leq 1 \]
  \[ r(2) - r(4) \leq 2 \]
  \[ r(3) - r(1) \leq 0 \]
  \[ r(3) - r(4) \leq 2 \]
  \[ r(4) - r(1) \leq 0 \]
  \[ r(4) - r(3) \leq 1 \]

Step 6: Constraint Graph
Example with c=3 (3/3)

Step 7: Using either **Bellman-Ford algorithm** or **Floyd-Warshall algorithm**, the retiming values for each node are calculated as

- \( r(1) = r(2) = r(3) = r(4) = 0 \)

Step 8: Retimed DFG can be obtained as

\[
\begin{array}{c}
\text{(1)} \\
\text{1} \\
\text{D} \\
\text{2} \\
\text{3} \\
\text{D} \\
\text{D} \\
\text{4} \\
\text{2D} \\
\text{2} \\
\end{array}
\]
Example with $c=2$ (1/2)

**Feasibility constraint:**
- $r(1) - r(3) \leq 1$
- $r(1) - r(4) \leq 2$
- $r(2) - r(1) \leq 1$
- $r(3) - r(2) \leq 0$
- $r(4) - r(2) \leq 0$

**Critical path constraint:**
- $r(1) - r(4) \leq 1$
- $r(2) - r(3) \leq 1$
- $r(2) - r(4) \leq 2$
- $r(3) - r(1) \leq 0$
- $r(3) - r(2) \leq -1$
- $r(3) - r(4) \leq 2$
- $r(4) - r(1) \leq 0$
- $r(4) - r(2) \leq -1$
- $r(4) - r(3) \leq 1$
Example with c=2 (2/2)

Using either Bellman-Ford algorithm or Floyd-Warshall algorithm, the retiming values for each node are calculated as

\[ r(1) = -1, \quad r(2) = 0 \]
\[ r(3) = -1, \quad r(4) = -1 \]
\[ r(1) = 0, \quad r(2) = 1 \]
\[ r(3) = 0, \quad r(4) = 0 \]

Retimed DFG can be obtained as

![Retimed DFG Diagram](image)
Conclusion

- Reduction of clock period, number of registers, and power consumption by retiming for synchronous systems
- Shortest path algorithms can be used to obtain a retiming solution if one exists.
- Retiming can also be used as a preprocessing step for folding and for computation of round-off noise as discussed in Chap 6 and 11, respectively.
Supplement Part
The Shortest Path Algorithm
Shortest Path Algorithms

- Length : sum of weights
- Negative cycle : when sum of lengths is negative
  ex : cycle 2→4→1→2
- $S_{UV}$ : the shortest path from the node U to the node V.
- The Bellman-Ford algorithm
- The Floyd-Warshall algorithm
The Bellman-Ford algorithm

• Single-point shortest path algorithm

\begin{align*}
&r^{(1)}(U)=0 \\
&\text{For } k=1 \text{ to } n \quad \text{If } k \neq U \\
&\quad r^{(1)}(k)=w(U \xrightarrow{e} k) \\
&\text{For } k=1 \text{ to } n-2 \\
&\quad \text{For } V=1 \text{ to } n \\
&\quad \quad r^{(k+1)}(V)=r^{(k)}(V) \\
&\quad \quad \text{For } W=1 \text{ to } n \\
&\quad \quad \quad \text{If } r^{(k+1)}(V) > r^{(k)}(W)+w(W \xrightarrow{e} V) \\
&\quad \quad \quad \quad r^{(k+1)}(V)=r^{(k)}(W)+w(W \xrightarrow{e} V) \\
&\text{For } V=1 \text{ to } n \\
&\quad \text{For } W=1 \text{ to } n \\
&\quad \quad \text{If } r^{(n-1)}(V) > r^{(n-1)}(W)+w(W \xrightarrow{e} V) \\
&\quad \quad \quad \text{return } \text{FALSE and exit} \\
&\text{return } \text{TRUE and exit}
\end{align*}
Example

U=2 (node 2)

<table>
<thead>
<tr>
<th>( r^{(k)}(V) )</th>
<th>k=1</th>
<th>k=2</th>
<th>k=3</th>
</tr>
</thead>
<tbody>
<tr>
<td>V=1</td>
<td>( \infty )</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>V=2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>V=3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>V=4</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

\( r^{(3)}(V) \) is the shortest path from node 2 to node V for V=1,2,3,4
The Floyd-Warshall algorithm

• All-points shortest path algorithm

For $V=1$ to $n$
  For $U=1$ to $n$
    $r^{(1)}(U,V)=w(U \rightarrow V)$

For $k=1$ to $n$
  For $V=1$ to $n$
    For $U=1$ to $n$
      $r^{(k+1)}(U,V)=r^{(k)}(U,V)$
      If $r^{(k+1)}(U,V)> r^{(k)}(U,k)+r^{(k)}(k,V)$
        $r^{(k+1)}(U,V)= r^{(k)}(U,k)+r^{(k)}(k,V)$

For $k=1$ to $n$
  For $U=1$ to $n$
    If $r^{(k)}(U,U)<0$
      return FALSE and exit

return TRUE and exit
### Example

Let's consider a simple graph with nodes 1, 2, 3, and 4. The connection between the nodes is represented by the network graph shown in the image.

The distance between nodes can be represented by the following matrices:

- **$R^{(1)}$**
  - $\begin{bmatrix} \infty & -3 & \infty & \infty \\ \infty & \infty & 1 & 2 \\ \infty & \infty & \infty & 2 \\ 1 & \infty & \infty & \infty \end{bmatrix}$

- **$R^{(2)}$**
  - $\begin{bmatrix} \infty & -3 & \infty & \infty \\ \infty & \infty & 1 & 2 \\ \infty & \infty & \infty & 2 \\ 1 & -2 & \infty & \infty \end{bmatrix}$

- **$R^{(3)}$**
  - $\begin{bmatrix} \infty & -3 & -2 & -1 \\ \infty & \infty & 1 & 2 \\ \infty & \infty & \infty & 2 \\ 1 & -2 & -1 & 0 \end{bmatrix}$

- **$R^{(4)}$**
  - $\begin{bmatrix} \infty & -3 & -2 & -1 \\ \infty & \infty & 1 & 2 \\ \infty & \infty & \infty & 2 \\ 1 & \infty & -1 & 0 \end{bmatrix}$

- **$R^{(5)}$**
  - $\begin{bmatrix} 0 & -3 & -2 & -1 \\ 3 & 0 & 1 & 2 \\ 3 & 0 & 1 & 2 \\ 1 & -2 & -1 & 0 \end{bmatrix}$

The notation $R^{(5)}(U,V)$ represents the shortest path from node $U$ to node $V$. For example, $R^{(5)}(U,V)$ indicates the shortest path from node $U$ to node $V$. The graph shows the connections between nodes with the distances corresponding to the matrices above.