Statistical Bellman-Ford Algorithm
With An Application to Retiming

Mongkol Ekpanyapong, Thaisiri Watewai* and Sung Kyu Lim

School of Electrical and Computer Engineering
Georgia Institute of Technology

Department of Industrial Engineering and Operations Research
University of California, Berkeley*
Outline

- Introduction and Motivation
- Related Work
- Methodology
- Experimental Results
- Conclusions
Deep Submicron Design

- Make transistor on the same die having different delay
Deep Submicron Design Issues

- Harder to get exact desired transistor size even on the same die

- Transistor size has directed impact on transistor delay

- Timing calculation becomes more complex

- Not only transistor size, wire size can also suffer from this variations
Static vs. Statistical Timing Analysis

- Static Timing Analysis

- Statistical Timing Analysis (Assume Gaussian distribution)
Outline

- Introduction and Motivation
- Related Work
- Methodology
- Experimental Results
- Conclusions
Related Work

- Chang, H. and Sapatnekar, S., *Statistical timing analysis considering spatial correlations using a single pert-like traversal*, ICCAD03


- Chen, R. and Zhou, H., *Clock schedule verification under process variations*, ICCAD04

- Cong, J. and Lim, S. K., *Retiming-based timing analysis with an application to mincut-based global placement*, TCAD04
Outline

- Introduction and Motivation
- Related Work
- Methodology
- Experimental Results
- Conclusions
What Make SSTA Difficult?

- Correlation

- Maximum/Minimum function’s approximation
Correlation

- Reconvergence Path Correlation

- Spatial Correlation
Principal Component Analysis

- Delay estimation with variation

\[
d = d_0 + \sum_{i \in \Gamma_g} \left[ \frac{\partial d}{\partial L_g^i} \right] \Delta L_g^i + \sum_{i \in \Gamma_g} \left[ \frac{\partial d}{\partial W_g^i} \right] \Delta W_g^i \\
+ \sum_{i \in \Gamma_{int}} \left[ \frac{\partial d}{\partial T_{int}^i} \right] \Delta T_{int}^i
\]

- Principal component analysis classifies each coefficient into orthogonal terms

\[
a_0 + \sum_{i=1}^{n} a_i \Delta x_i + a_{n+1} \Delta R_a
\]

\[
b_0 + \sum_{i=1}^{n} b_i \Delta x_i + b_{n+1} \Delta R_b
\]
Maximum/Minimum Function Approximation

\[
\phi(x) \equiv \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}} \quad (1)
\]

\[
\Phi(y) \equiv \int_{-\infty}^{y} \phi(x) \, dx \quad (2)
\]

\[
\theta \equiv \left( \sigma^2_A + \sigma^2_B - 2\rho \sigma_A \sigma_B \right)^{1/2} \quad (3)
\]

\[
T_A = \int_{-\infty}^{\infty} \frac{1}{\sigma_A} \phi \left( \frac{x - a_0}{\sigma_A} \right) \Phi \left( \frac{\left( \frac{x - b_0}{\sigma_B} \right) - \rho \left( \frac{x - a_0}{\sigma_A} \right)}{\sqrt{1 - \rho^2}} \right) \, dx \quad (4)
\]

\[
E[\max(A, B)] = a_0 T_A + b_0 (1 - T_A) + \theta \phi \left( \frac{a_0 - b_0}{\theta} \right) \quad (5)
\]

\[
\text{var}[\max(A, B)] = \sigma^2_A T_A + a_0^2 (1 - T_A) + \sigma^2_B (1 - T_A) + (a_0 + b_0) \theta \phi \left( \frac{a_0 - b_0}{\theta} \right) - \left[ E[\max(A, B)] \right]^2 \quad (6)
\]
Distribution Approximation

- Assume Gaussian distribution (can be represented by using mean and standard variation)

- Addition/Subtraction function results in Gaussian distribution as a solution

- Maximum/Minimum function approximation results in a new kind of distribution
Distribution Approximation

- Assume Gaussian distribution (can be represented by using mean and standard variation)

- Addition/Subtraction function can be computed using convolution

- Maximum/Minimum function approximation results in a new kind of distribution
Retiming Algorithm

- Retiming Algorithm
  - Can handle sequential circuits

- Bellman-Ford Algorithm
  - Can be modified to handle Statistical Timing Analysis
  - Faster comparing with other approaches
Longest Path Bellman-Ford Algorithm

Bellman-Ford Algorithm
input: directed graph \((G, w, s)\)
output: longest path lengths from \(s\)

1. for (each \(v \in V\))
2. \(a[v] \leftarrow -\infty\);
3. \(a[s] \leftarrow 0\);
4. \(iter \leftarrow 1\);
5. while (STOP = FALSE and \(iter < |V|\))
6. \(iter \leftarrow iter + 1\);
7. STOP = TRUE;
8. for (each edge \((u, v) \in E\))
9. if \((a[v] < a[u] + w(u, v))\)
10. \(a[v] \leftarrow a[u] + w(u, v)\);
11. STOP = FALSE;
12. for (each \((u, v) \in E\))
13. if \((a[v] < a[u] + w(u, v))\)
14. return (FALSE);
15. return (TRUE);
Longest Path Bellman-Ford Algorithm

Bellman-Ford Algorithm
input: directed graph \((G, w, s)\)
output: longest path lengths from \(s\)

1. for (each \(v \in V\))
2. \(a[v] \leftarrow -\infty\);
3. \(a[s] \leftarrow 0\);
4. \(\text{iter} \leftarrow 1\);
5. while (\(\text{STOP} = \text{FALSE} \text{ and } \text{iter} < |V|\))
6. \(\text{iter} \leftarrow \text{iter} + 1\);
7. \(\text{STOP} = \text{TRUE};\)
8. for (each edge \((u, v) \in E\))
9. if \((a[v] < a[u] + w(u, v))\)
10. \(a[v] \leftarrow a[u] + w(u, v);\)
11. \(\text{STOP} = \text{FALSE};\)
12. for (each \((u, v) \in E\))
13. if \((a[v] < a[u] + w(u, v))\)
14. return (\(\text{FALSE}\));
15. return (\(\text{TRUE}\));
Bellman-Ford Algorithm
input: directed graph \( (G, w, s) \)
output: longest path lengths from \( s \)

1. for (each \( v \in V \))
2. \( a[v] \leftarrow -\infty \);
3. \( a[s] \leftarrow 0 \);
4. \( iter \leftarrow 1 \);
5. while (STOP = FALSE and \( iter < |V| \))
6. \( iter \leftarrow iter + 1 \);
7. STOP = TRUE;
8. for (each edge \((u, v) \in E\))
9. if \( (a[v] < a[u] + w(u, v)) \)
10. \( a[v] \leftarrow a[u] + w(u, v) \);
11. STOP = FALSE;
12. for (each \((u, v) \in E\))
13. if \( (a[v] < a[u] + w(u, v)) \)
14. return (FALSE);
15. return (TRUE);
Problems of Bellman-Ford Update

- The approximation error can cause infinite update

- If error bound technique is used, some paths can be ignored
Problems of Bellman-Ford Update

- The approximation error can cause infinite update

- If error bound technique is used, some paths can be ignored
Problems of Bellman-Ford Update

- The approximation error can cause infinite update

\[ m = 1 \quad m = 2.2 \]

\[ m = 1.1 \]

\[ m = 0.1 \quad m = 2.1 \]

- If error bound technique is used, some paths can be ignored

\[ m = 0.1 \quad m = 1.1 \]

\[ m = 2.2 \]

\[ m = 0.1 \quad m = 2.1 \]

Error bound = 0.2
**K-SBF Algorithm**

<table>
<thead>
<tr>
<th>K-bounded Statistical Bellman-Ford Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>input: directed graph $G$ with node/edge delay pdf</td>
</tr>
<tr>
<td>output: longest path length distribution from source</td>
</tr>
<tr>
<td>1. DFS($s$) to find all backward edges;</td>
</tr>
<tr>
<td>2. for (each $v \in V$)</td>
</tr>
<tr>
<td>3. if (backward edge connected to $v$)</td>
</tr>
<tr>
<td>4. $BN \leftarrow v$;</td>
</tr>
<tr>
<td>5. $K = 0$;</td>
</tr>
<tr>
<td>6. for (each $v \in BN$)</td>
</tr>
<tr>
<td>7. perform DFS($v$);</td>
</tr>
<tr>
<td>8. $k =</td>
</tr>
<tr>
<td>9. if ($K &lt; k$)</td>
</tr>
<tr>
<td>10. $K = k$;</td>
</tr>
<tr>
<td>11. for (each $v \in V$)</td>
</tr>
<tr>
<td>12. $a[v] \leftarrow -\infty$;</td>
</tr>
<tr>
<td>13. $a[s] \leftarrow 0$;</td>
</tr>
<tr>
<td>14. for ($itr = 1$ to $k + 1$)</td>
</tr>
<tr>
<td>15. for (each $v \in V$)</td>
</tr>
<tr>
<td>16. $a[v] \leftarrow \max_{u \in FI(v)} (a[u] + d(v) + d(u, v))$;</td>
</tr>
<tr>
<td>17. $P(\text{cycle}) \leftarrow \text{check_pos_cycle();}$</td>
</tr>
<tr>
<td>18. if ($P(\text{cycle}) \leq Pa$)</td>
</tr>
<tr>
<td>19. return (FALSE);</td>
</tr>
<tr>
<td>20. return (TRUE);</td>
</tr>
</tbody>
</table>

---

**Backward node**
Retiming Delay Computation

- Arrival time computation

\[ \phi \geq a[t] = \max_{i=1,...,K} \{\psi_i - \kappa_i \phi\} \]

- Maximum cycle computation

\[ \zeta_j = \xi_j - \sigma_j \phi \leq 0 \quad j = 1,...,C. \]

- Retiming delay computation

\[ \phi = \max \left[ \max_{i=1,...,K} \left\{ \frac{\psi_i}{\kappa_i + 1} \right\}, \max_{j=1,...,C} \left\{ \frac{\xi_j}{\sigma_j} \right\} \right] \]
Algorithm for Maximum Cycle Distribution/Positive Cycle Detection

- Based on modified algorithm by Chen et al, ICCAD04
Algorithm for Maximum Cycle Distribution/Positive Cycle Detection

- Removing backward edge and compute cycle distribution from new src to new sink of the new graph
Positive Cycle Detection

\[ \text{mean (µ)} + 3 \cdot \text{std. variation (σ)} \leq 0 \]
Bound on Retiming Delay

- Definition
  \( \phi_l : \) the value of clock period when gate and interconnect are replaced by best case delay value
  \( \phi_m : \) the value of clock period when gate and interconnect are replaced by mean case delay value
  \( \phi_u : \) the value of clock period when gate and interconnect are replaced by worst case delay value

- \( \phi_l \leq \phi \leq \phi_u \)

- \( \phi_m \leq E[\phi] \)
Outline

- Introduction and Motivation
- Related Work
- Methodology
- Experimental Results
- Conclusions
Sensitivity Parameters for SSTA

<table>
<thead>
<tr>
<th></th>
<th>L</th>
<th>W</th>
<th>Tox</th>
<th>W</th>
<th>H</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.099</td>
<td>-0.099</td>
<td>-0.095</td>
<td>0.019</td>
<td>-0.133</td>
<td>-0.12</td>
</tr>
</tbody>
</table>
## Benchmark Characteristics

<table>
<thead>
<tr>
<th>ckt</th>
<th>gate</th>
<th>PI</th>
<th>PO</th>
<th>FF</th>
<th>K + 1</th>
<th>b-node</th>
</tr>
</thead>
<tbody>
<tr>
<td>s5378</td>
<td>2828</td>
<td>36</td>
<td>49</td>
<td>163</td>
<td>76</td>
<td>95</td>
</tr>
<tr>
<td>s9234</td>
<td>5597</td>
<td>36</td>
<td>39</td>
<td>211</td>
<td>239</td>
<td>354</td>
</tr>
<tr>
<td>s13207</td>
<td>8027</td>
<td>31</td>
<td>121</td>
<td>669</td>
<td>510</td>
<td>637</td>
</tr>
<tr>
<td>s15850</td>
<td>9786</td>
<td>14</td>
<td>87</td>
<td>597</td>
<td>495</td>
<td>699</td>
</tr>
<tr>
<td>s38417</td>
<td>22397</td>
<td>28</td>
<td>106</td>
<td>1636</td>
<td>1444</td>
<td>1660</td>
</tr>
<tr>
<td>s38584</td>
<td>19407</td>
<td>12</td>
<td>278</td>
<td>1452</td>
<td>1860</td>
<td>2054</td>
</tr>
<tr>
<td>b140</td>
<td>5401</td>
<td>32</td>
<td>299</td>
<td>245</td>
<td>451</td>
<td>616</td>
</tr>
<tr>
<td>b150</td>
<td>7092</td>
<td>37</td>
<td>519</td>
<td>449</td>
<td>988</td>
<td>1408</td>
</tr>
<tr>
<td>b200</td>
<td>11979</td>
<td>32</td>
<td>512</td>
<td>490</td>
<td>1486</td>
<td>2197</td>
</tr>
<tr>
<td>b210</td>
<td>12156</td>
<td>32</td>
<td>512</td>
<td>490</td>
<td>1511</td>
<td>2209</td>
</tr>
<tr>
<td>b220</td>
<td>17351</td>
<td>32</td>
<td>725</td>
<td>703</td>
<td>1870</td>
<td>2770</td>
</tr>
</tbody>
</table>
Distribution Comparison on S5378

Monte = Monte-Carlo Simulation
eSBF = SBF with Error Bound
## Distribution Comparison

<table>
<thead>
<tr>
<th>ckt</th>
<th>Monte-Carlo</th>
<th></th>
<th>eSBF</th>
<th></th>
<th>kSBF</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mean</td>
<td>std.dev.</td>
<td>mean</td>
<td>std.dev.</td>
<td>mean</td>
<td>std.dev.</td>
</tr>
<tr>
<td>s5378</td>
<td>179.65</td>
<td>6.27</td>
<td>163.29</td>
<td>5.49</td>
<td>179.47</td>
<td>6.51</td>
</tr>
<tr>
<td>s9234</td>
<td>229.24</td>
<td>16.18</td>
<td>164.58</td>
<td>7.14</td>
<td>225.53</td>
<td>10.51</td>
</tr>
<tr>
<td>s13207</td>
<td>304.93</td>
<td>13.27</td>
<td>279.47</td>
<td>8.72</td>
<td>305.76</td>
<td>13.05</td>
</tr>
<tr>
<td>s15850</td>
<td>363.16</td>
<td>15.83</td>
<td>317.32</td>
<td>7.57</td>
<td>363.37</td>
<td>15.72</td>
</tr>
<tr>
<td>s38417</td>
<td>187.11</td>
<td>4.53</td>
<td>184.95</td>
<td>5.28</td>
<td>189.00</td>
<td>8.41</td>
</tr>
<tr>
<td>s38584</td>
<td>437.02</td>
<td>19.41</td>
<td>399.62</td>
<td>10.45</td>
<td>436.75</td>
<td>19.33</td>
</tr>
<tr>
<td>b14o</td>
<td>161.19</td>
<td>10.70</td>
<td>117.42</td>
<td>5.37</td>
<td>160.43</td>
<td>5.31</td>
</tr>
<tr>
<td>b15o</td>
<td>247.00</td>
<td>10.00</td>
<td>212.72</td>
<td>18.45</td>
<td>247.88</td>
<td>10.23</td>
</tr>
<tr>
<td>b20o</td>
<td>259.68</td>
<td>9.03</td>
<td>229.16</td>
<td>6.12</td>
<td>267.13</td>
<td>9.33</td>
</tr>
<tr>
<td>b21o</td>
<td>267.57</td>
<td>6.18</td>
<td>240.01</td>
<td>4.08</td>
<td>267.98</td>
<td>9.24</td>
</tr>
<tr>
<td>b22o</td>
<td>286.63</td>
<td>18.92</td>
<td>300.49</td>
<td>25.17</td>
<td>320.62</td>
<td>15.26</td>
</tr>
<tr>
<td>runtime</td>
<td>21 days</td>
<td></td>
<td>2.7 hours</td>
<td></td>
<td>3.7 hours</td>
<td></td>
</tr>
</tbody>
</table>
Worst Case Delay Comparison

<table>
<thead>
<tr>
<th>ckt</th>
<th>Normal dly deter</th>
<th>Normal dly stat</th>
<th>Retiming dly deter</th>
<th>Retiming dly stat</th>
</tr>
</thead>
<tbody>
<tr>
<td>s5378</td>
<td>66.96</td>
<td>57.25</td>
<td>58</td>
<td>49.19</td>
</tr>
<tr>
<td>s9234</td>
<td>110.2</td>
<td>94.98</td>
<td>65</td>
<td>53.62</td>
</tr>
<tr>
<td>s13207</td>
<td>117.26</td>
<td>105.47</td>
<td>101</td>
<td>82.34</td>
</tr>
<tr>
<td>s15850</td>
<td>147.05</td>
<td>124.62</td>
<td>112</td>
<td>86.47</td>
</tr>
<tr>
<td>s38417</td>
<td>83.38</td>
<td>72.49</td>
<td>54</td>
<td>53.83</td>
</tr>
<tr>
<td>s38584</td>
<td>116.95</td>
<td>103.95</td>
<td>102</td>
<td>83.51</td>
</tr>
<tr>
<td>b14o</td>
<td>90.53</td>
<td>79.27</td>
<td>61</td>
<td>56.6</td>
</tr>
<tr>
<td>b15o</td>
<td>99.11</td>
<td>88.28</td>
<td>73</td>
<td>58.69</td>
</tr>
<tr>
<td>b20o</td>
<td>136.12</td>
<td>114.78</td>
<td>83</td>
<td>70.75</td>
</tr>
<tr>
<td>b21o</td>
<td>163.54</td>
<td>139.15</td>
<td>90</td>
<td>77.91</td>
</tr>
<tr>
<td>b22o</td>
<td>145.45</td>
<td>121.96</td>
<td>85</td>
<td>72.05</td>
</tr>
<tr>
<td>Avg.</td>
<td>1</td>
<td>0.87</td>
<td>0.7</td>
<td>0.6</td>
</tr>
</tbody>
</table>
# Algorithm Summary

<table>
<thead>
<tr>
<th></th>
<th>STA</th>
<th>SSTA</th>
<th>RTA</th>
<th>SRTA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>goal</strong></td>
<td>computation of deterministic timing slack values in combinational circuits</td>
<td>computation of statistical timing slack distribution in combinational circuits</td>
<td>computation of deterministic timing slack values after retiming in sequential circuits</td>
<td>computation of statistical timing slack distribution after retiming in sequential circuits</td>
</tr>
<tr>
<td><strong>delay values</strong></td>
<td>deterministic</td>
<td>statistical distribution</td>
<td>deterministic</td>
<td>statistical distribution</td>
</tr>
<tr>
<td><strong>retiming</strong></td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td><strong>circuit graph</strong></td>
<td>directed acyclic graph, FF removed</td>
<td>directed acyclic graph, FF removed</td>
<td>cyclic retiming graph, FF becomes edge weight</td>
<td>cyclic retiming graph, FF becomes edge weight</td>
</tr>
<tr>
<td><strong>basic algorithm</strong></td>
<td>topological sort</td>
<td>topological sort, statistical min/max and arithmetic operation</td>
<td>Bellman-Ford</td>
<td>Bellman-Ford, statistical min/max and arithmetic operation</td>
</tr>
<tr>
<td><strong>approach</strong></td>
<td>visit nodes in forward (backward) topological order to compute arrival (require) time.</td>
<td>visit nodes in forward (backward) topological order to compute and propagate statistical arrival (require) time distribution.</td>
<td>compute longest path for cyclic graph with negative edge weights to compute timing slack after retiming.</td>
<td>compute “statistical longest path distribution” and “slack distribution after retiming” with statistical Bellman-Ford algorithm</td>
</tr>
<tr>
<td><strong>complexity</strong></td>
<td>$O(n)$</td>
<td>$O(n)$</td>
<td>$O(n^2)$, $O(k \cdot n)$ in practice</td>
<td>$O(n^2)$, $O(k \cdot n)$ in practice</td>
</tr>
<tr>
<td><strong>advantage</strong></td>
<td>simple and fastest</td>
<td>handle statistical analysis</td>
<td>model FFs and predict delay after retiming</td>
<td>perform statistical retiming and report delay distribution after retiming</td>
</tr>
<tr>
<td><strong>disadvantage</strong></td>
<td>can’t handle retiming nor statistical variations</td>
<td>slow and no retiming consideration</td>
<td>slow and can’t handle statistical variations</td>
<td>slow</td>
</tr>
</tbody>
</table>


Outline

- Introduction and Motivation
- Related Work
- Methodology
- Experimental Results
- Conclusions
Conclusions

- In deep submicron design, it is harder to control process parameters.

- Process variation will make timing analysis more difficult.

- Here, we propose Statistical Bellman-Ford algorithm that can be used to compute longest path with cycle.

- We show that our result is accurate comparing with Monte-Carlo simulation.
QUESTIONS?