Design Optimization by Fine-grained Interleaving of Local Netlist Transformations in Lagrangian Relaxation

Apostolos Stefanidis, Dimitrios Mangiras, Giorgos Dimitrakopoulos
Democritus University of Thrace, Greece

Chrystostomos Nicopoulos
University of Cyprus, Cyprus

David Chinnery
Mentor, a Siemens Business, USA

June 30, 2020
• Design optimization
• Timing-Power optimization using Langrangian relaxation (LR)
• Embedding multiple heuristics inside the same Multi Mode Multi Corner LR optimization loop
  • The criteria that each heuristic should satisfy to be compatible with LR-based optimization
  • Order of applying each heuristic
• Experimental results based on benchmarks of the TAU2019 contest
• Conclusions
Design optimization

• Gate-level netlist changes to optimize:
  • Timing – fix early/late violations
  • Reduce leakage/dynamic power, area, wire length ...

• Can be applied in any physical design step
  • Additional considerations (e.g. SI noise) + need for accuracy increase through the flow

• Examples:

Initial circuit

Sizing

Buffer insertion

Relocating
Design optimization using Lagrangian relaxation

- Relaxes timing constraints into a simplified objective function
  - Lagrangian multipliers (LMs) weigh the constraints to try and ensure that they are met
- Already successfully applied for
  - Combinational gate sizing
  - Clock tree sizing
  - Timing driven incremental placement
- **Our proposal**: embed multiple optimization heuristics in the same Lagrangian relaxation optimization loop
Problem formulation

\[
\min \sum_{c \in \text{cells}} P(c) + A(c) - \sum_{j \in \text{POs}} slk_j^L - \sum_{j \in \text{POs}} slk_j^E
\]

\textbf{s. t.: } \begin{align*}
slk_j^L & \leq 0 \text{ and } slk_j^E \leq 0, \quad \forall j \in \text{POs} \\
slk_j^L & \leq r_j^L - a_j^L \text{ and } slk_j^E \leq a_j^E - r_j^E, \quad \forall j \in \text{POs} \\
a_i^L + d_{i \rightarrow j}^L & \leq a_j^L \text{ and } a_j^E \leq a_i^E + d_{i \rightarrow j}^E, \quad \forall i \rightarrow j \in \text{arcs}
\end{align*}

\textbf{Target:} minimize sum of leakage power, area, and total negative slack (TNS)

\begin{itemize}
  \item \(P(c)\): leakage of cell \(c\)
  \item \(A(c)\): area of cell \(c\)
  \item \(L\): late timing information
  \item \(E\): early timing information
  \item \(slk_j\): negative slack of pin \(j\)
  \item \(r_j\): required time of pin \(j\)
  \item \(a_j\): arrival time of pin \(j\)
  \item \(d_{i \rightarrow j}\): delay of timing arc \(i \rightarrow j\)
  \item \(\text{arcs}\): timing arcs of the design
  \item \(\text{cells}\): cells of the design
  \item \(\text{POs}\): Primary outputs or timing endpoints of the design
\end{itemize}
Lagrangian relaxation formulation (1)

\[
\begin{align*}
\min & \sum_{c \in \text{cells}} P(c) + A(c) - \sum_{j \in \text{POs}} slk_j^L - \sum_{j \in \text{POs}} slk_j^E \\
\text{s. t.:} & \quad slk_j^L \leq 0 \text{ and } slk_j^E \leq 0, \quad \forall j \in \text{POs} \\
& \quad slk_j^L \leq r_j^L - a_j^L \text{ and } slk_j^E \leq a_j^E - r_j^E, \quad \forall j \in \text{POs} \\
& \quad a_i^L + d_{i \rightarrow j}^L \leq a_j^L \text{ and } a_j^E \leq a_i^E + d_{i \rightarrow j}^E, \quad \forall i \rightarrow j \in \text{arcs} \\
\end{align*}
\]

\[\lambda_{j_0}^L, \lambda_{j_1}^L: \text{Late LMs for slack constraints on endpoints}\]
\[\lambda_{i \rightarrow j}^L: \text{Late LMs for early delay constraints on arcs}\]

\[
\begin{align*}
\min & \sum_{c \in \text{cells}} P(c) + A(c) - \sum_{j \in \text{POs}} slk_j^L - \sum_{j \in \text{POs}} slk_j^E + \\
& \quad \sum_{j \in \text{POs}} (\lambda_{j_0}^L slk_j^L + \lambda_{j_0}^E slk_j^E) + \\
& \quad \sum_{j \in \text{POs}} (\lambda_{j_1}^L (slk_j^L - r_j^L + a_j^L) + \lambda_{j_1}^E (slk_j^E - a_j^E + r_j^E)) + \\
& \quad \sum_{i \rightarrow j \in \text{arcs}} (\lambda_{i \rightarrow j}^L (a_i^L + d_{i \rightarrow j}^L - a_j^L) + \lambda_{i \rightarrow j}^E (a_j^E - a_i^E - d_{i \rightarrow j}^E))
\end{align*}
\]

\[\lambda_{j_0}^E, \lambda_{j_1}^E: \text{Early LMs for slack constraints on endpoints}\]
\[\lambda_{i \rightarrow j}^E: \text{Early LMs for early delay constraints on arcs}\]
Lagrangian relaxation formulation (2)

- \( \lambda \) values represent the criticality of each constraint
- Karush-Kuhn-Tucker (KKT) optimality conditions
  \[
  \sum_{i \in in} \lambda^L_{i \rightarrow j} = \sum_{k \in out} \lambda^L_{j \rightarrow k}, \quad \sum_{i \in in} \lambda^E_{i \rightarrow j} = \sum_{k \in out} \lambda^E_{j \rightarrow k}
  \]
- By applying the KKT conditions and simplifying:
  \[
  \min \sum_{c \in cells} P(c) + A(c) + \sum_{i \rightarrow j \in arcs} \lambda^L_{i \rightarrow j} d^L_{i \rightarrow j} - \lambda^E_{i \rightarrow j} d^E_{i \rightarrow j}
  \]

\[
\begin{align*}
\lambda^L_{1 \rightarrow 3} &= \lambda^L_{3 \rightarrow 4} + \lambda^L_{2 \rightarrow 4} \\
\lambda^L_{1 \rightarrow 3} &= \lambda^L_{3 \rightarrow 4} + \lambda^L_{2 \rightarrow 4} \\
\lambda^E_{40} &= \lambda^E_{3 \rightarrow 4} + \lambda^E_{2 \rightarrow 4} \\
\lambda^E_{1 \rightarrow 3} &= \lambda^E_{3 \rightarrow 4}
\end{align*}
\]
Timing arc \( i \to j \) affects cost function by:

\[
\lambda_{i\to j}^L d_{i\to j}^L - \lambda_{i\to j}^E d_{i\to j}^E
\]

- High late LM \( \Rightarrow \) delay should decrease \( \Rightarrow \) late critical arc
- High early LM \( \Rightarrow \) delay should increase \( \Rightarrow \) early critical arc

Update method:

- \[
\lambda_{i\to j}^L = \lambda_{i\to j}^L \frac{(a_i^L + d_{i\to j}^L)}{a_j^L}, \quad \lambda_{j0}^L = \lambda_{j0}^L \frac{a_j^L}{r_j^L},
\]

- \[
\lambda_{i\to j}^E = \lambda_{i\to j}^E \frac{a_j^E}{(a_i^E + d_{i\to j}^E)}, \quad \lambda_{j0}^E = \lambda_{j0}^E \frac{r_j^E}{a_j^E},
\]

- LM values are propagated backwards proportionally to respect KKT optimality conditions
• Recalculate only timing information around the cell’s local arcs
  • Calculating the cost function for every timing arc of the design is avoided to save runtime
  • $LC(v) = P(v) + A(v) + \sum_{i \rightarrow j \in local\_arcs} \lambda^L_{i \rightarrow j} d^L_{i \rightarrow j} - \lambda^E_{i \rightarrow j} d^E_{i \rightarrow j}$
How LR optimization loop works: Gate sizing example

- Make decisions based on local information ⇒ timing is updated only on local arcs
- Have discrete choices ⇒ different size / Vt options
- Evaluate each choice using LM values ⇒ pick the choice with the lowest local cost
Incorporating design transformations in the LR loop

• Any transformation satisfying certain criteria can be applied inside the Lagrangian relaxation

• The method has to:
  • Make decisions based on local information
  • Have discrete choices
  • Evaluate each choice using LM values and the same local cost function
  • Apply small changes each iteration ⇒ allows LR to adapt to the change
In this work we apply five transformations inside the LR-based optimization loop:

- Cell sizing
- Pin swapping
- Buffering for early violations
- Buffering for late violations
- Clock skew assignment

All make local decisions based on LM values.
Cell sizing

• Try every option for cell to resize
• Keep the option with the lowest local cost
  • Options that cause load/slew/slack violations are rejected
• Applied on gates and flip flops that are
  • Early or late timing critical
  • Power/area critical
Handling of early/late timing conflicts

- Refers to cells with **conflicting early/late timing violations**
- LR based resizing will try to **balance the slacks** based on LM values ⇒ **slow convergence**
- Solution: only include **late LMs** in the local cost function of these cells
  - Sizing focuses on late violations
  - Early violations will be solved by other methods (e.g. buffering)

---

**Initial circuit**

- $\text{slk}^E=-80\text{ps}$
- $\text{slk}^L=-40\text{ps}$

**Conflict handling by late slack focus**

- $\text{slk}^E=-120\text{ps}$
- $\text{slk}^L=0\text{ps}$

---

A. Stefanidis / Democritus University of Thrace, Greece
Buffer insertion for fixing late timing violations

• Used for driving large net loads
• Applied on the outputs of cells with high input to output capacitance ratio
• Try every buffer type and keep the lowest local cost option (including adding no buffer as an option)
Buffer insertion for fixing hold timing violations

• Increase the delay on early-timing violating paths
• Where to add delay
  • On the most critical path through all early violating endpoints
  • One the pin on the most critical path with the highest late-early LM difference
• How much delay is added?
  • Add that much delay that does not degrade Late negative slack
Pin swapping

• Reconnect nets of logically equivalent pins to improve timing

• For each gate that has equivalent pins:
  • Find the most critical input net
  • Try to assign it to each other equivalent pin
  • Keep the option with the lowest local LR cost
Useful clock skew assignment

• Changes the clock arrival time on registers
• The LMs of the clock pin guide delay addition/removal
  \[ \lambda_{clk}^L = \lambda_Q^L + \lambda_D^E \]
  \[ \lambda_{clk}^E = \lambda_Q^E + \lambda_D^L \]
• Delay added if \( \lambda_{clk}^E > \lambda_{clk}^L \) else delay is removed
Overall optimization loop

- Starts with initializations
- Every iteration performs LR based transformations
- Timing information derived from the most critical late/early timing corners
- Final recovery steps improve QoR and remove hold violations
Multi-corner approach

- Optimization happens across many different corners
- Timing closure required across all corners
- Leakage only measured on the typical corner

- At the start of each iteration we identify the most critical corner for early and late mode ⇒ corner with the worst slack
- Each corner has different delays, setup/hold times ⇒ most critical corner can change during optimization
- All transformations and LM updates use timing information from the most critical corners
- Only the most critical corners gets updated in the local timing updates
The order of applying each heuristic facilitates the propagation of timing info using only local timing updates

1. Heuristics affecting endpoints – clock skew, register sizing
2. Heuristics traversing the design in topological order – datapath sizing, pin swapping
3. Heuristics performed on intermediate levels – early and late buffering

In this way timing information is carried from the start points to the end points using only local timing updates
Experimental setup

• Applied on the benchmarks of the TAU 2019 Multi-Mode Multi-Corner (MMMC) Design Optimization contest
  • Six benchmarks provided
    • Sizes from 600 to about 800,000 cells
    • SPEF files provided for the three smallest designs
  • Optimization across five timing corners
• Compared against TAU contest winner’s executable
Experimental results – best period

• The best period achieved by each method
  • Timing closure across all corners

• Our approach achieves 14% lower clock period (i.e. faster)
  • Also saves 15% leakage and 5% area

<table>
<thead>
<tr>
<th>Design</th>
<th>Period (ps)</th>
<th>Leakage (uW)</th>
<th>Area (um$^2$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Ours</td>
<td>Winner</td>
<td>Ours</td>
</tr>
<tr>
<td>s1196</td>
<td>1040</td>
<td>918</td>
<td>12.1</td>
</tr>
<tr>
<td>systemcdes</td>
<td>1777</td>
<td>1788</td>
<td>87</td>
</tr>
<tr>
<td>usb_funct</td>
<td>2166</td>
<td>2306</td>
<td>419</td>
</tr>
<tr>
<td>vga_lcd</td>
<td>1871</td>
<td>2826</td>
<td>3215</td>
</tr>
<tr>
<td>leon2_iccad</td>
<td>4532</td>
<td>4677</td>
<td>25170</td>
</tr>
<tr>
<td>leon3mp_iccad</td>
<td>3878</td>
<td>5246</td>
<td>20045</td>
</tr>
</tbody>
</table>
Experimental Results – Common clock period

- Leakage/Area comparison on the clock period where both algorithms close timing
  - Our approach saves 16% more leakage and 6% more area
- Better QoR for higher runtime on larger benchmarks

<table>
<thead>
<tr>
<th>Design</th>
<th>Period (ps)</th>
<th>Leakage (μW)</th>
<th>Area (μm²)</th>
<th>Runtime (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Ours</td>
<td>Winner</td>
<td>Ours</td>
</tr>
<tr>
<td>s1196</td>
<td>1040</td>
<td>12</td>
<td>13</td>
<td>550</td>
</tr>
<tr>
<td>systemcdes</td>
<td>1788</td>
<td>85</td>
<td>96</td>
<td>3603</td>
</tr>
<tr>
<td>usb_funct</td>
<td>2306</td>
<td>397</td>
<td>402</td>
<td>18002</td>
</tr>
<tr>
<td>vga_lcd</td>
<td>2826</td>
<td>3075</td>
<td>3106</td>
<td>145752</td>
</tr>
<tr>
<td>leon2_iccad</td>
<td>4677</td>
<td>24996</td>
<td>30354</td>
<td>1234180</td>
</tr>
<tr>
<td>leon3mp_iccad</td>
<td>5246</td>
<td>19632</td>
<td>23816</td>
<td>962764</td>
</tr>
</tbody>
</table>
The contribution of each heuristic

- Most impactful methods:
  - Cell sizing and clock skew assignment
  - But these are also the most runtime expensive
Conclusions

- Presented the simultaneous application of multiple heuristics inside the same LR optimization loop
- Additional specific novel parts
  - The overall formulation and the criteria needed for each heuristic to be embedded in the same optimization loop
  - Novel approach on handling early/late timing conflicts
  - Optimizes both combinational cells and sequential cells (registers), and optimizes the clock arrival time (useful skew)
- Future work:
  - Runtime improvements
  - Better explore the order of applying each heuristic