Problem C:
Incremental Timing-driven Placement

http://cad-contest.el.cycu.edu.tw/problem_C/default.html

Myung-Chul Kim, Jin Hu
IBM Corporation

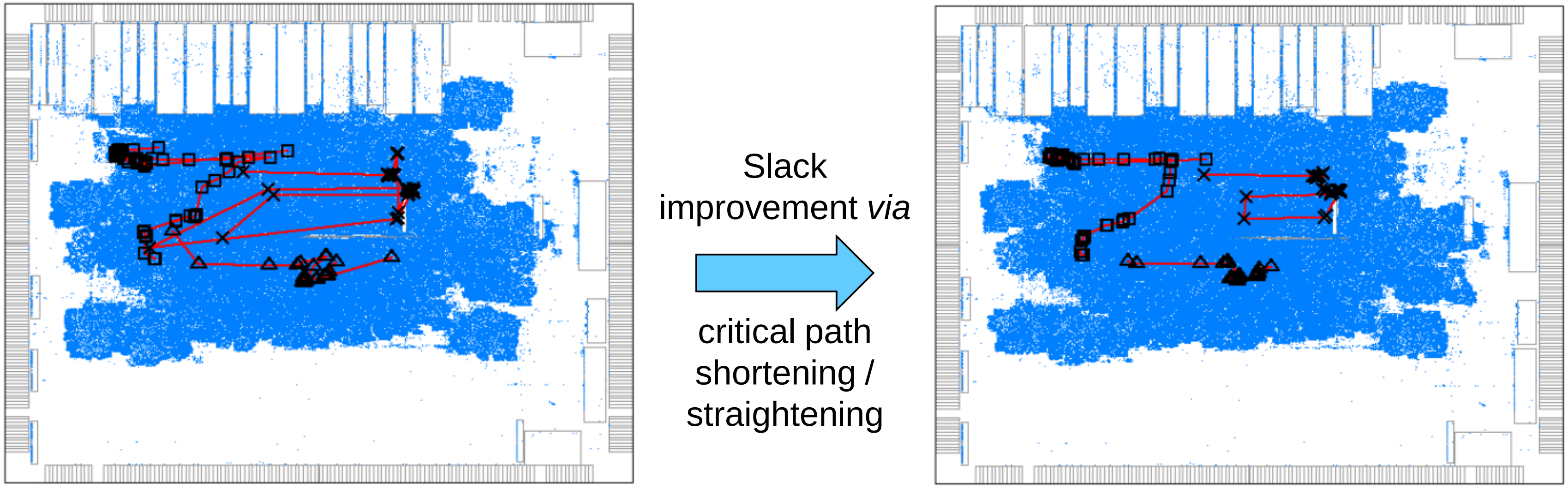
Sponsored by:

Outline

- **Motivation**
- **Contest Constraints**
- **New Feature at ICCAD 2015 Contest – LCB**
- **Contest Evaluation Methodology**

Timing-driven Placers in Physical Synthesis

- **Timing-driven placement (TDP) plays a key role in *timing closure***
  - Traditional placement methods often overlook the requirements of specific nets, and therefore cannot effortlessly close timing
  - In order to satisfy timing requirements, the slacks of timing-critical nets must be improved (e.g., by reducing the delay)
  - TDP incorporates timing information within, and perform placement operations based on timer feedback.



Challenges in TDP

- **Placer stability**
  - Globally-oriented placement methods can significantly change the placement landscape
  - Timing information often becomes inaccurate or obsolete, especially at early design stages
- **Timing-improvement convergence**
  - Improving slacks of a subset of paths ≠ TNS/WNS improv.
  - “Fixing” one failing path can cause several other paths to fail.
- **Preserving original placement optimizations**
  - Timing optimization can conflict with traditional placement

Motivation for the ICCAD-2015 Contest in TDP

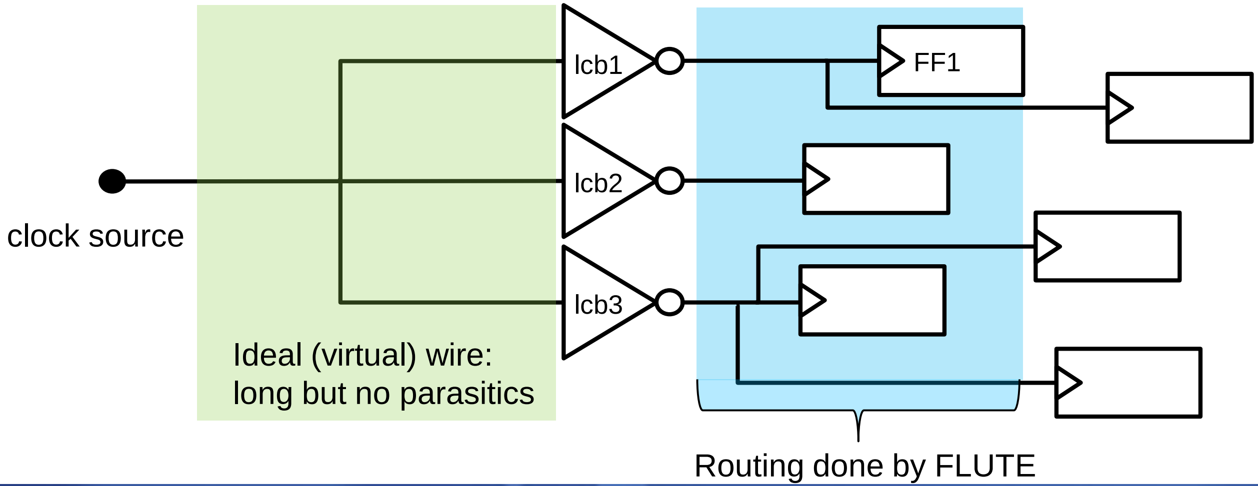
- **Provide a timing-integrated placement framework**
  - Encourage academic research and development in TDP
  - The use of an academic timer UI-Timer2.0 that works on industry standard liberty format w/ CPPR – one of the top performers @ TAU 2015 timer contest
- **Release large realistic industrial ICCAD-2015 contest benchmarks**
  - Reversed-engineered from past contests' bookshelf format files by individual academic researchers using their published methods
  - Mapped to 45nm technology
  - Includes millions of gates and numerous non-rectangular shaped macros equipped with timing and hierarchy info
  - Provides a common timing evaluation framework in the academic environment
  - Provide comprehensive technology information for future PD research (info for gate sizing / CNS / buffering / routing..)

ICCAD 2015 Placement Contest Constraints

- **Hard constraints: disqualify solutions when violated**
  - **A. Maximum cell displacements (Incrementality)**
    - Significant disruptions are discouraged to maintain the quality of the original solution provided by upstream optimization
    - At different stages in the flow, different degrees of freedom is required, modeled by two different cell displacement limits per benchmark
  - **B. Legality** site alignment without cell overlaps
  - **C. Maximum runtime of 12 hours**
  - **D. Clock-LCB-FF connection validity**
- **Soft constraint: penalized when violated**
  - **A. Minimal degradation in density profiles**
    - Placers are discouraged to create overly-packed placements, to further accommodate downstream transforms in physical synthesis

New feature at ICCAD 2015 - Local Clock Buffer (LCB)

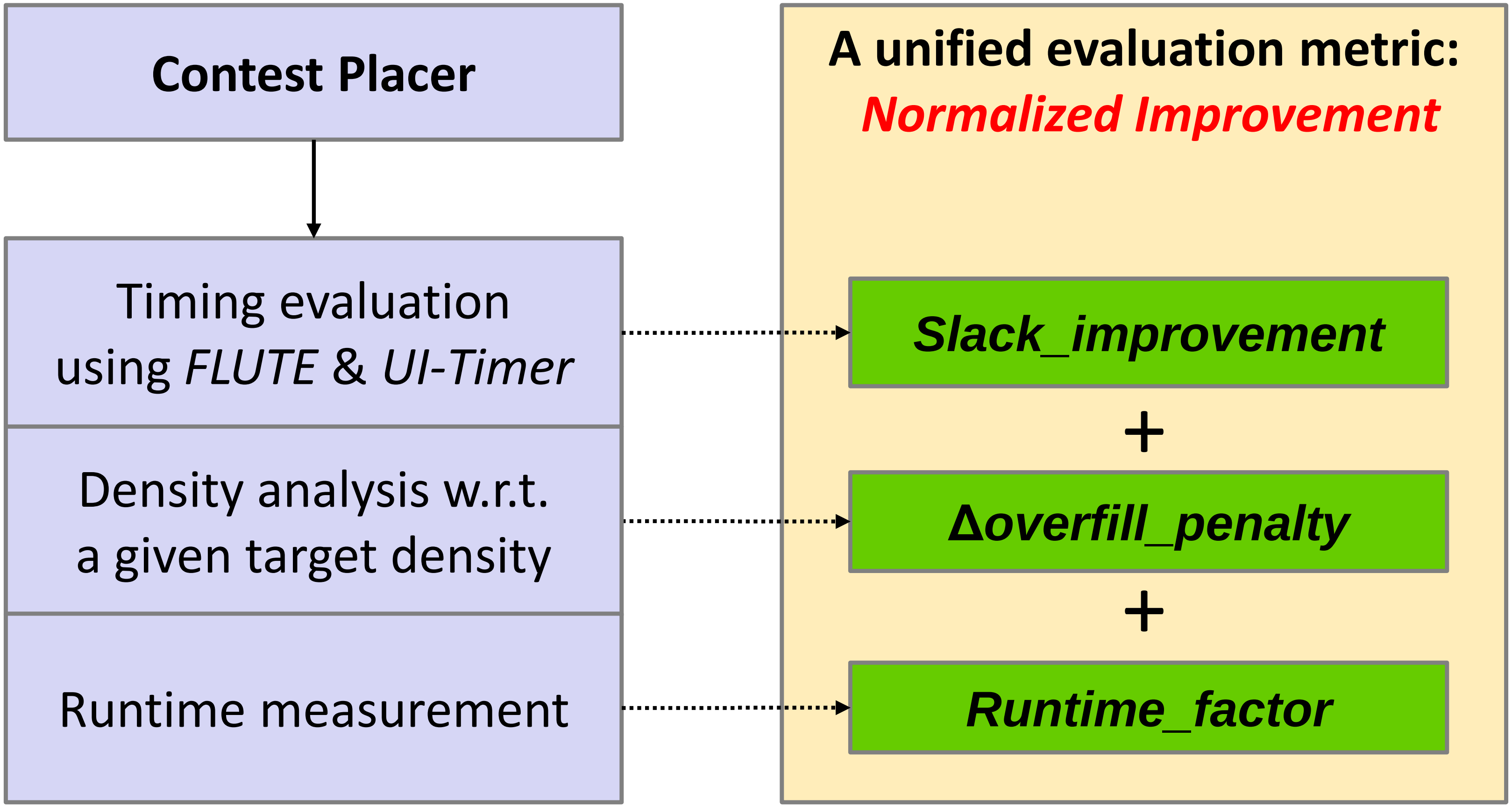
- **Lessons learned from the previous contest..**
  - Clock routing by FLUTE is very sensitive to placement change, so thus RC and timing
  - Even with smaller res/cap, very long clock routes can potentially dominate overall performance due to lack of buffering
- **Local Clock Buffers (LCBs) introduced for clock network**
  - Each FF's clock is now driven by a LCB, implemented by the largest inverter available
  - Ideal wires are assumed between LCBs and clock source, and other relatively short signal routing is still done by FLUTE
  - This effectively assumes ideal clock routing (i.e,. zero skew) that follows TDP stage
  - Contestants are allowed to move LCBs and also change FF-to-lcb associations (e.g., from FF1->lcb1 to FF1->lcb2)



Clock-to-LCB-to-FF connection validity

- **The contestants are allowed to change FF-to-LCB association by providing your own .ops file, along with final .def (placement) file.**
  - See ICCAD 2015 contest file formats for details
- **Nevertheless, the following properties must be maintained**
  - The clock signal must be fed to every FF's clock pin via LCBs.
  - Each FFs's clock pins must driven by a single LCB.
  - Each LCB's # of fanouts must be less than MAXIMUM_LCB_FANOUTS from ICCAD15.parm

Contest Evaluation Flow and Metric



** Please see the ICCAD 2014 paper for technology-dependent parameters and details of parasitic extraction and timing evaluation*

Evaluation Metric: *Slack_improvement*

- **With respect to the initial timing results of the original placement,**

$\boldsymbol{Slack\_improvement}$ (a unit of percentage) =

$$\{\mathbf{w}_{\mathbf{TNS}} \times \left(\boldsymbol{w}_{\boldsymbol{late}} \times \boldsymbol{TNS\_improv.}^{\boldsymbol{late}} + \boldsymbol{w}_{\boldsymbol{early}} \times \boldsymbol{TNS\_improv.}^{\boldsymbol{early}}\right)$$

$$+\boldsymbol{w}_{\boldsymbol{WNS}} \times (\boldsymbol{w}_{\boldsymbol{late}} \times \boldsymbol{WNS\_improv.}^{\boldsymbol{late}} + \boldsymbol{w}_{\boldsymbol{early}} \times \boldsymbol{WNS\_improv.}^{\boldsymbol{early}})\}$$

where all TNS or WNS improvements are relative percentages

- **We set** $\{\boldsymbol{w}_{\mathbf{TNS}}, \mathbf{w}_{\mathbf{WNS}}, \mathbf{w}_{\mathbf{late}}, \mathbf{w}_{\mathbf{early}}\} = \{\mathbf{2.0}, \mathbf{1.0}, \mathbf{5.0}, \mathbf{1.0}\}$
  - Emphasis on TNS > WNS, and late slack >> early slack improvements
  - With given weights, maximum achievable slack improvement = 1800 (%)

NOTE: The weights maybe subject to change during the final evaluation

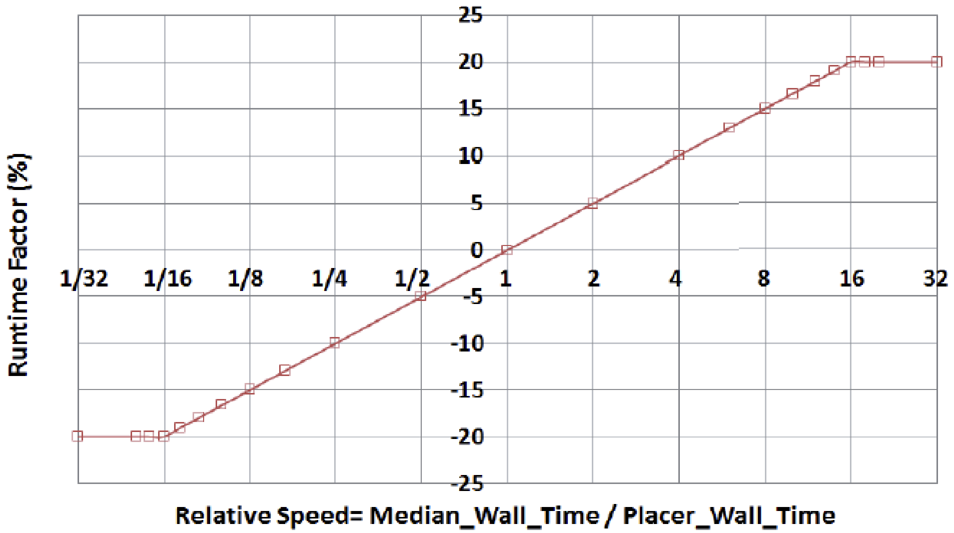
Evaluation Metric: Δ*overfill_penalty*

- ***Overfill_penalty* is defined as a weighted sum of** $\gamma\_over\_utilization$ **that tracks the average overflow of top** Γ**% densities bins.**
  - $Overfill\ penality = \frac{K_\gamma \times \gamma\_over\_utilization}{\Sigma K_\gamma}$ where $\gamma = \{2,5,10,20\}$
  - Higher weights on peak utilization: $K_2 = 10, K_5 = 4, K_{10} = 2, K_{20} = 1$
  - The best case : overfill_penalty = 0.0

- **We track Δoverfill_penalty between before and after each run**
  - $\boldsymbol{Quality\ Score = max(\ Slack\_improv. \times (\ 1 - \Delta\ overfill\_penalty), 0)}$
  - **Any degradation (overfill_penalty increase) diminishes quality score!**

Evaluation Metric: *Runtime_factor*

- **Allow parallel / multi-threaded implementations up to 8**
- **For each test case**
  - $Relative\_speed = \frac{Median\,Wall\,Time}{Placer\,Wall\,Time}$
  - $Runtime\_factor = 0.05\ \times \log_2(Relative\_speed)$



i.e., 5% bonus to the quality score for 2X speed-up (capped at ±20%)

A Unified Evaluation Metric: Normalized Improv.

Normalized Improvement

= *Slack_improv.*

× *(1 - Δoverfill_penalty)*

Placement quality

× *(1 + Runtime_factor)*

Placer runtime

- **First round: Top 5 teams based on quality of placements**
  - Zero score is given to solutions that violate hard constraints or degrade
- **Second round: Top 3 teams based on normalized improv.**
  - The median runtimes were calculated for top 5 teams