Decoupling Capacitor Insertion Minimizing IR-Drop Violations and Routing DRVs

Daijoon Hyun[*], **Younggwang Jung**[†], Insu Cho[†], and Youngsoo Shin[†]

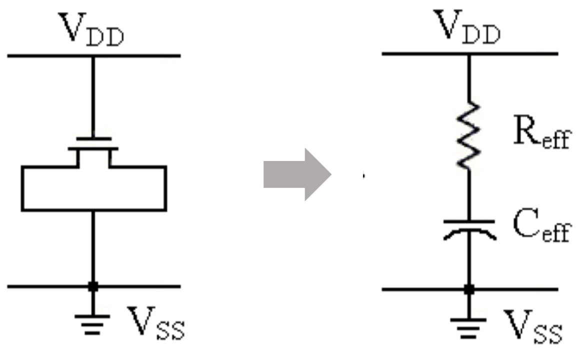
[*]Department of EE, Cheongju University, Korea
[†]School of EE, KAIST, Korea

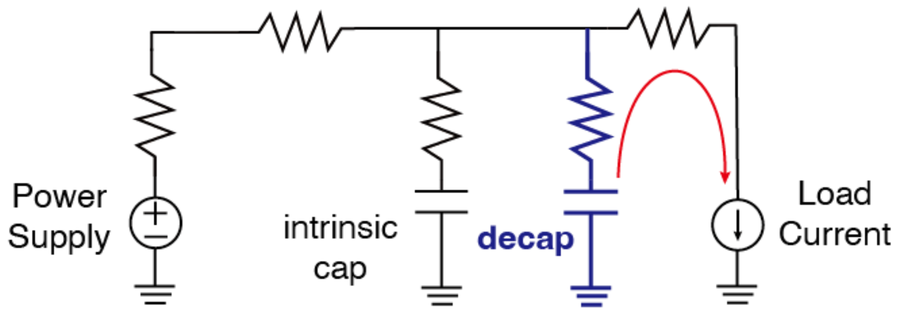
Outline

- Introduction
- Motivation
- Proposed method
  - DRV penalty prediction
  - Decoupling capacitance insertion
- Experimental results
- Summary

Introduction

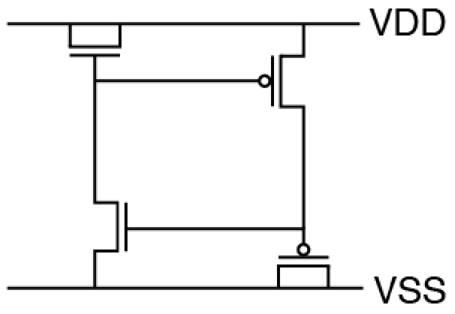
- Typical decoupling capacitor (decap)
  - Reduce dynamic IR-drop by supplying current to local switching cells
  - Drawbacks
    - Large gate leakage, accounting for 10% of total power [TVLSI'09]
    - Risk of oxide breakdown [ISQED'06]

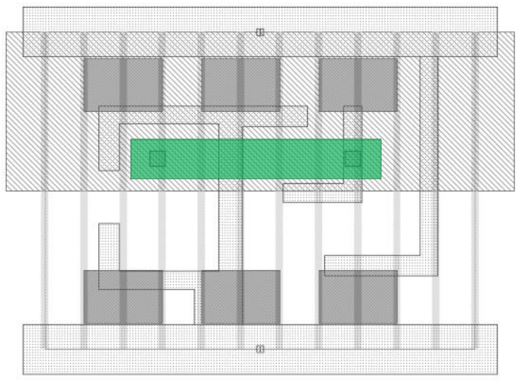




Introduction

- Cross-coupled decap [ISQED'06, TVLSI'08]
  - Gate voltage is decreased by 10%
    - 41% smaller gate leakage
    - Improved ESD reliability
  - Internal metal patterns in M2 are exposed as routing blockage





🟩 : M2 metal

Introduction

- Several works for decap budgeting
  - Adjoint sensitivity-based budgeting [ISPD'02, ISQED'05]
  - Budgeting using charge-based model [DAC'06, DATE'09]
  - Fast budgeting by random work approach [ASPDAC'07]
  - Hierarchical budgeting using cross-entropy method [DAC'10, TCAD'11]

- **There is no work for considering complex decap cell**

Introduction

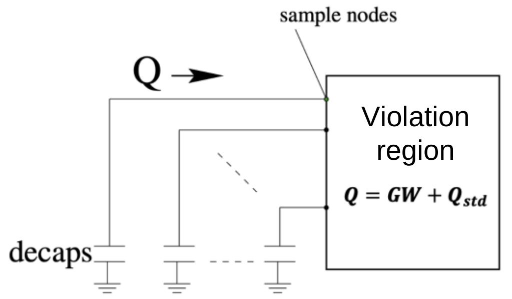
- Budgeting using charge-based model [DAC'06]
  1. Compute capacitance for each sample node to supply demanded charge (Q=CV) for satisfying IR-drop constraint using LP
  2. Place decaps uniformly in violation region

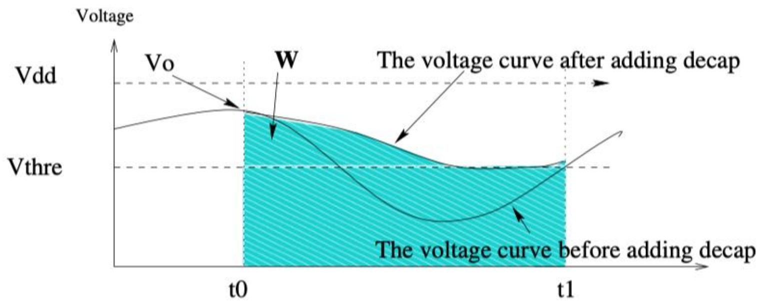
$$Minimize \sum_{i \in SP} C_i$$

$$\boldsymbol{s.t.}\ \boldsymbol{MC} \geq \boldsymbol{GW} + \boldsymbol{Q_{std}}$$

$$W_i \geq (V_{o,i} - V_{thre}) * \frac{t_1 - t_0}{2}, i \in SP$$

$$\boldsymbol{M} = \begin{bmatrix} V_{o,1} - V_1 \\ V_{o,2} - V_2 \\ \dots \\ V_{o,m} - V_m \end{bmatrix}, \boldsymbol{C} = [C_1, C_2, \dots, C_m]^T,\ Q_{std,i} = \int_{t0}^{t1} I_i$$



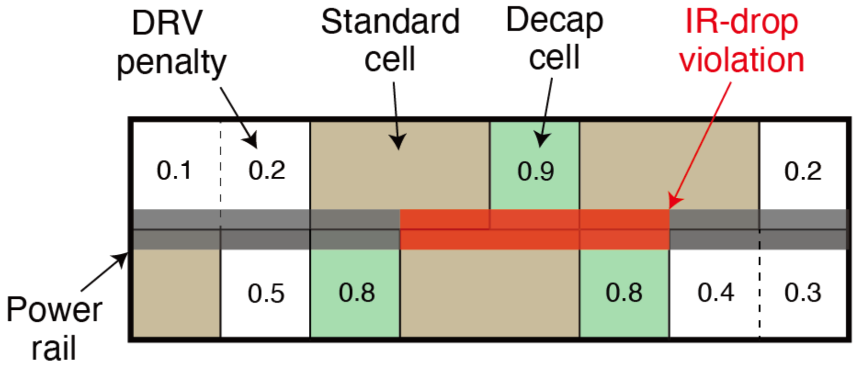


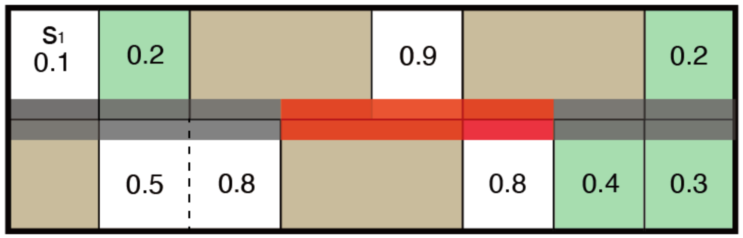
Zhao, Min, et al. "A fast on-chip decoupling capacitance budgeting algorithm using macromodeling and linear programming." DAC, 2006.

Motivation

- In conventional methods
  - As a result, decap cells are placed close to violated nodes
  - Routing DRV can be severly increased (up to 40%)

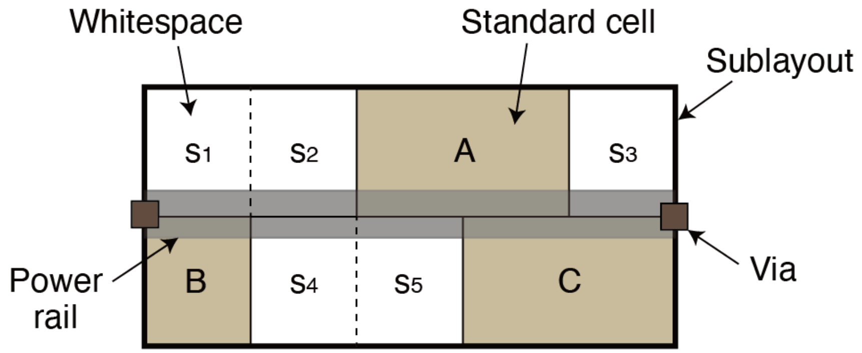
- Decap insertion considering routing DRV
  - Reduce the increase in routing DRV to 12%

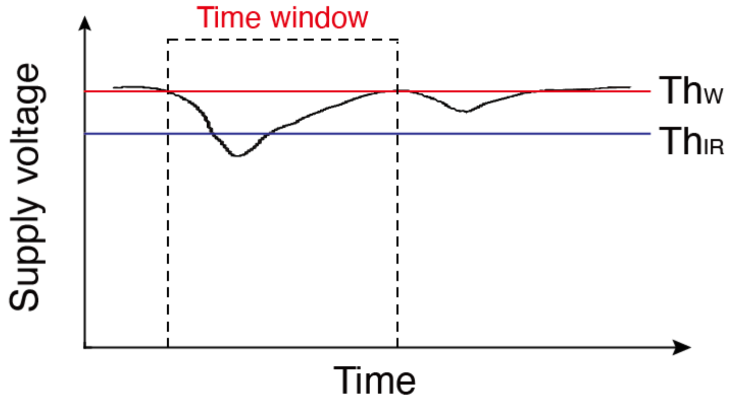




Overall Flow

- Given a circuit layout after cell placement
  1. Perform IR drop analysis & identify dynamic IR-drop violations
  2. Extract sublayouts and time windows
  3. DRV penalty prediction for every possible decap insertion
  4. Decap insertion considering DRV penalty
     - MIQCP formulation
     - Heuristic method





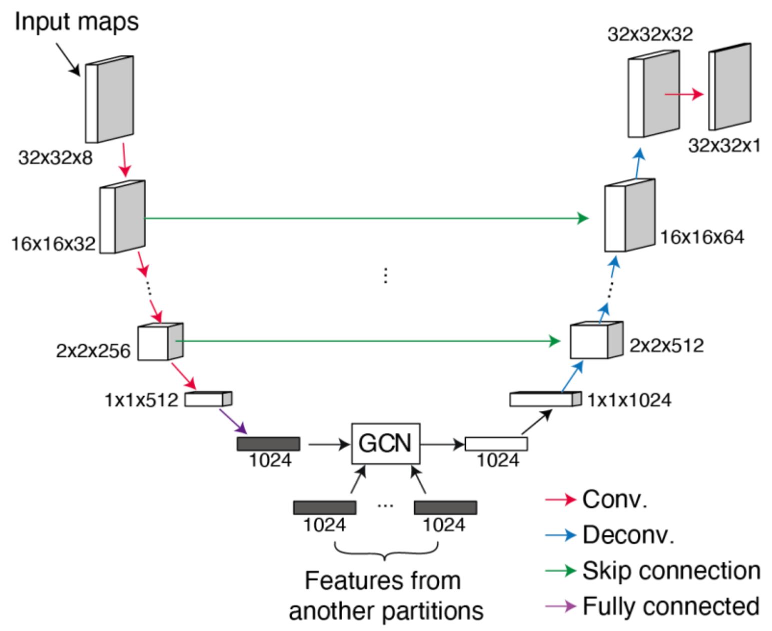
DRV Penalty Prediction

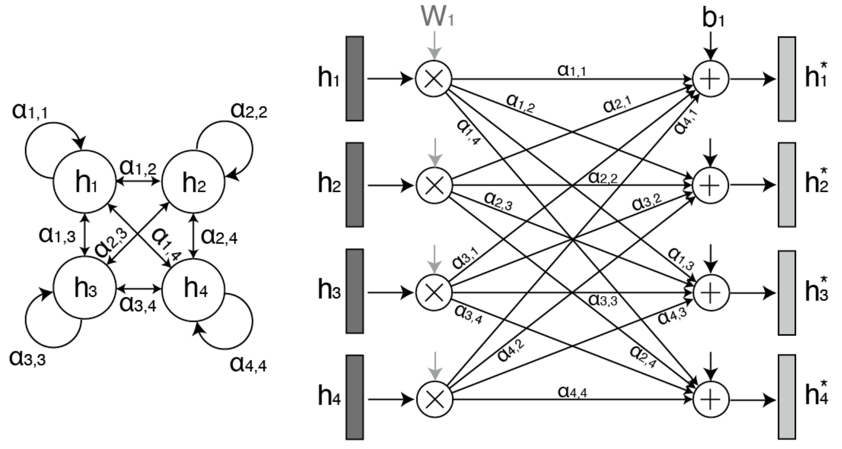
- DRV penalty
  - Total sum of DRV probabilities changed from a decap insertion
  - Require two times of DRV probability predictions

- DRV probability prediction
  - Input features are prepared for each partition of layout
  - Input feature
    - Cell density map
    - Pin density map
    - Decap metal density map
    - Metal density maps after first iteration of detailed routing
  - Output
    - DRV probability map

DRV Penalty Prediction

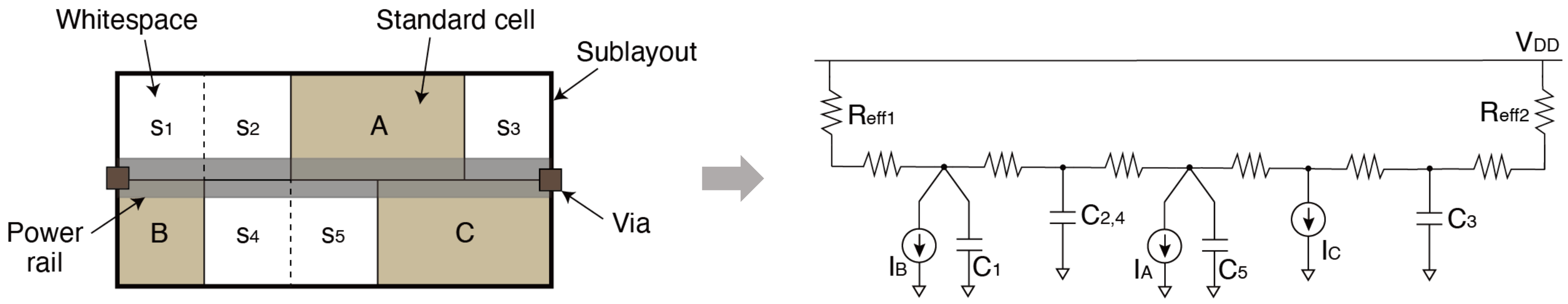
- ML model: U-Net + graph convolutional network (GCN)
  - One U-Net for a layout partition
    - 5 convolutional and 5 deconvolutional layers
    - Skip connection
  - Multiple U-Nets intersect in GCN with the weight of distance between partitions





Decap Insertion

- Circuit modeling
  - Sublayout is modeled by a RC network with current sources
    - Standard cell → time-varying current source
    - Candidate site → capacitor
    - Power rail → resistor
    - PDN from via to power PAD → effective resistances



Decap Insertion

- MIQCP formulation
  - Goal is to minimize IR-drop violation and DRV penalty
  - Constraints
    - Cell overlapping is not allowed
    - In KCL, differential term is reduced to linear form using backward Euler

Objective: Minimize $\Delta V_{max} + \delta P_t$

Subject to:

$$
\begin{aligned}
& x_{i,j} = 0, && \forall (i, j) \in O_c \\
& x_{i,j} + x_{i',j'} \leq 1, && \forall (i, j, i', j') \in O_d \\
& c_i = \sum_{j \in D} c_j^d x_{i,j}, && \forall i \in N_d \\
& \mathbf{G} \cdot \mathbf{V(t)} + \mathbf{C} \cdot \frac{d\mathbf{V(t)}}{dt} = \mathbf{I(t)}, && \forall t \in T \\
& P_t = \sum_{i \in N_d} \sum_{j \in D} p_{i,j}\, x_{i,j}, && \\
& \Delta V_{max} \geq (1 - \alpha) V_{DD} - v_i(t), && \forall i \in N,\ \forall t \in T \\
& \Delta V_{max} \geq 0. &&
\end{aligned}
$$

Decap Insertion

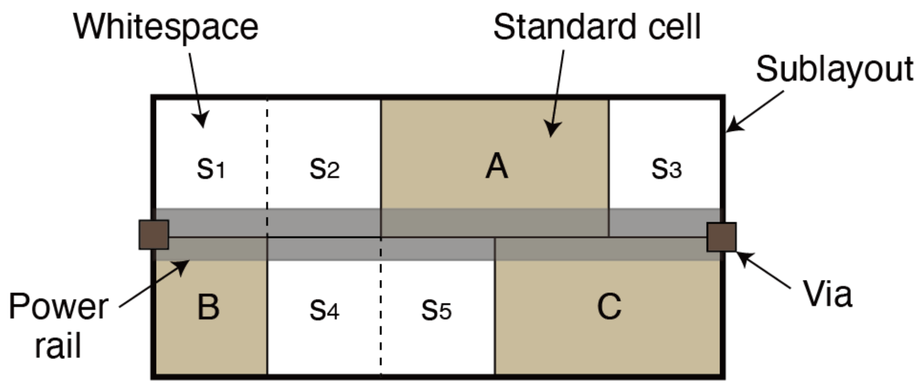
- Heuristic algorithm
    1. Derive voltage equations in circuit model
        - Use modified nodal analysis & backward Euler method
    2. Find total capacitance C_T resolving violation
        - Iteratively add unit capacitance to each node with smallest IR-drop
    3. Decap insertion with <u>constrained minimum cost MIS* search</u>

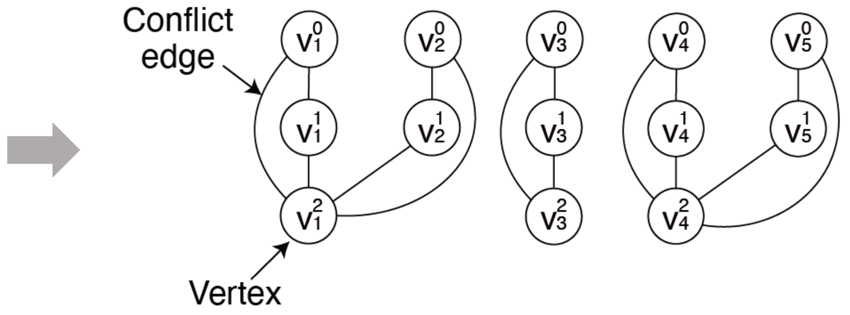
*) MIS: maximum independent set

    4. Iteratively reduce the size of decap with largest sensitivity
        - Sensitivity $S_{i,j} = \frac{\Delta p_{i,j}}{\Delta c_j \, \Delta \overline{V}_i^{max}}$

Decap Insertion

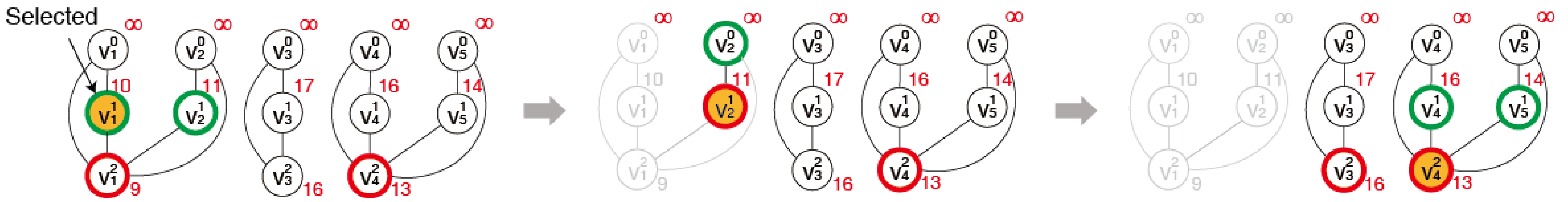
- Constrained minimum cost MIS search
  - Graph modeling
    - Vertex v_i^j: placing decap of size j on site i
    - Edge $\left(v_i^j, v_{i'}^{j'}\right)$ exists when two vertices cannot be existing together
    - Cost of vertex v_i^j: $cost_{i,j} = p_{i,j} - \beta c_j \Delta \bar{V}_i^{max}$





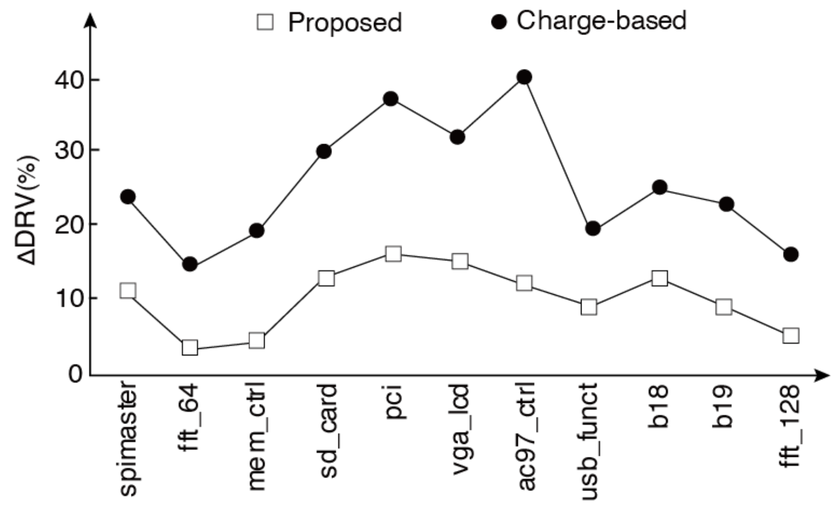
Decap Insertion

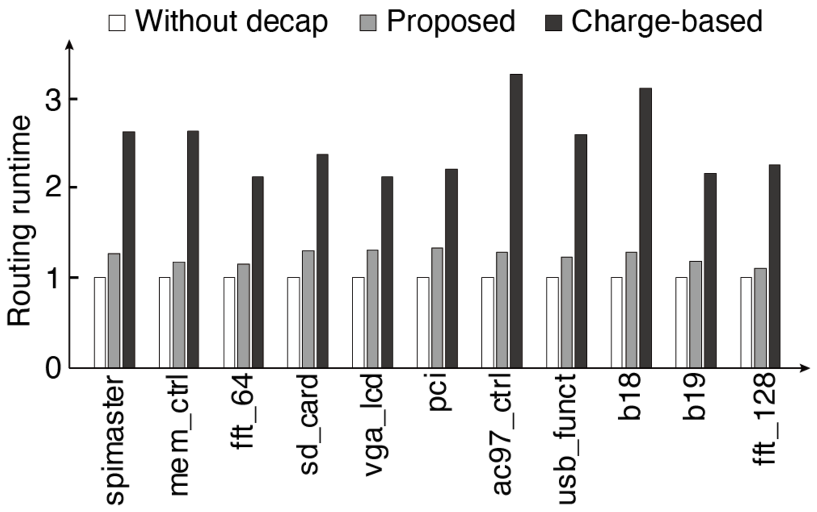
- Algorithm
  1. Select a vertex (V_{min}) with the smallest cost and the next one except its neighbors (red)
  2. Select two vertices with smallest cost on neighbors of V_{min} (green)
  3. Compare the cost sums of (1) and (2), and select the smaller one in the vertex set with smaller cost
  4. Remove the vertex and its neighbors
  5. Repeat (1)-(4) until total capacitance is greater than C_T
  6. If the capacitance is not over C_T, increase β and repeat all process
     - $cost_{i,j} = p_{i,j} - \beta c_j \Delta \bar{V}_i^{max}$



Experimental Results

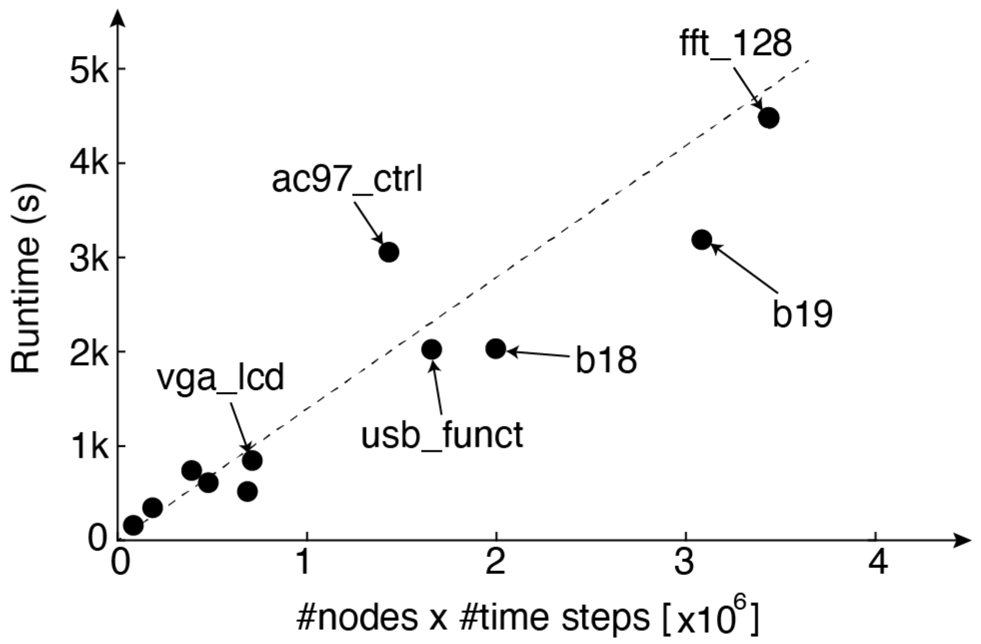
- Assessment of routing DRV
  - Achieves 16% less routing DRVs and 48% smaller routing runtime
    - Compared to charge-based method, not considering routing DRVs





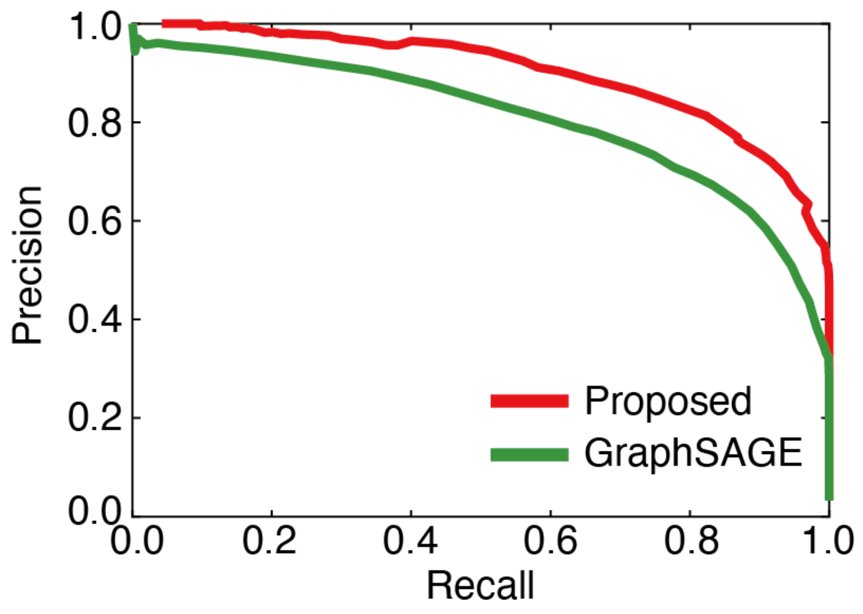
Experimental Results

- Analysis of runtime
  - Proportional to product of total number of nodes and total number of time steps
  - Large runtime is spent for fft_128 due to the largest number of sublayout with violations



Experimental Results

- Assessment of ML model
  - F1-score of 82% and AUC of 0.86
  - 7.3% and 0.09 higher than GNN-based prediction (GraphSAGE)



X. Chen, Z. Di, W. Wu, Q. Wu, J. Shi and Q. Feng, "Detailed Routing Short Violation Prediction Using Graph-Based Deep Learning Model," in TCAS II: Express Briefs, Feb. 2022.

Summary

- Post-placement decap insertion considering routing DRVs is addressed

- Proposed method
  - ML model (U-Net + GCN), which predicts DRV penalty from a decap insertion
  - MIQCP formulation and heuristic algorithm for decap insertion
  - It achieves 16% less DRVs while satisfying dynamic IR-drop