60

WHERE **INNOVATION** BEGINS

DESIGN AUTOMATION CONFERENCE

JULY 9–13, 2023

MOSCONE WEST CENTER

SAN FRANCISCO, CA, USA

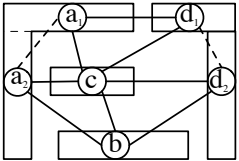
Layout Decomposition via Boolean Satisfiability

Hongduo Liu[1], Peiyu Liao[1], Mengchuan Zou[2], Bowen Pang[2], Xijun Li[2], Mingxuan Yuan[2], Tsung-Yi Ho[1], Bei Yu[1]

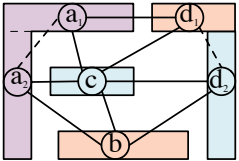
[1]Chinese University of Hong Kong
[2]Huawei Noah's Ark Lab

Layout Decomposition

- Decompose one layout onto multiple masks for better manufacturability
- Avoid assigning two features within a given distance to the same mask
- Layout decomposition can be formulated as graph coloring



(a)



(b)

Literature Review

- Exact Algorithm: Integer Linear Programming[1]
- Approximation Algorithm:
  - Semidefinte Programming[2]
  - Linear Programming[3]
  - Heuristic methods[4]

---

[1] W. Li *et al.*, "Openmpl: An open-source layout decomposer",, vol. 40, no. 11, pp. 2331–2344, 2020.

[2] B. Yu *et al.*, "Layout decomposition for triple patterning lithography",, vol. 34, no. 3, pp. 433–446, 2015.

[3] Y. Lin *et al.*, "Triple/quadruple patterning layout decomposition via linear programming and iterative rounding",, vol. 16, no. 2, p. 023 507, 2017.

[4] S.-Y. Fang *et al.*, "A novel layout decomposition algorithm for triple patterning lithography",, 2012, pp. 1185–1190.

Motivation

Drawbacks of previous methods:

- ILP can get optimal solutions but too slow.
- Approximation algorithms show great scalability, but the solution quality is poor.

Two important observations:

- Boolean nature of decision variables in ILP formulation $\Rightarrow$ Boolean satisfiability
- Conflict optimization and stitch minimization are two problems nested with each other $\Rightarrow$ Bilevel Reformulation

Satisfiable Problem

- A propositional logic formula is said to be in Conjunctive Normal Form (CNF) if it is a conjunction ("and") of disjunctions ("ors") of literals.
- A literal is either a boolean variable x or its negation $\neg x$.
- For example, $(p \vee q) \wedge (\neg q \vee \neg q)$ is a CNF, where $p, q, \neg p \neg q$ are all literals. The disjunctions $(p \vee q)$ and $(\neg q \vee \neg q)$ are also called clauses.
- The satisfiability (SAT) problem is to find a satisfying assignment to the boolean variables such that the CNF formula yields true.

ILP Formulation

$$\min \sum_{r_i \in p_m, r_j \in p_n, c_{ij} \in CE} C_{mn} + \alpha \sum_{s_{ij} \in SE} s_{ij}, \tag{1a}$$

$$\text{s.t.} \quad x_{i1} + x_{i2} \leq 1, \quad \forall i \in V, \tag{1b}$$

$$x_{i1} + x_{i2} + x_{j1} + x_{j2} + C_{mn} \geq 1, \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, \tag{1c}$$

$$x_{i1} - x_{i2} + x_{j1} - x_{j2} - C_{mn} \leq 1, \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, \tag{1d}$$

$$-x_{i1} + x_{i2} - x_{j1} + x_{j2} - C_{mn} \leq 1, \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, \tag{1e}$$

$$x_{i1} + x_{i2} + x_{j1} + x_{j2} - C_{mn} \leq 3, \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, \tag{1f}$$

$$x_{i1} - x_{j1} + s_{ij} \geq 0, \quad \forall e_{ij} \in SE, \tag{1g}$$

$$x_{i1} - x_{j1} - s_{ij} \leq 0, \quad \forall e_{ij} \in SE, \tag{1h}$$

$$x_{i2} - x_{j2} + s_{ij} \geq 0, \quad \forall e_{ij} \in SE, \tag{1i}$$

$$x_{i2} - x_{j2} - s_{ij} \leq 0, \quad \forall e_{ij} \in SE. \tag{1j}$$

Bilevel Optimization

A bilevel optimization problem reads

$$\min_{x \in X, y} \; F(x, y) \tag{2a}$$

$$\text{s.t.} \; G(x, y) \geq 0, \tag{2b}$$

$$y \in S(x), \tag{2c}$$

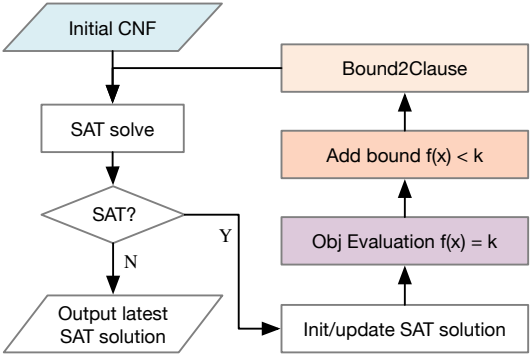
where $S(x)$ is the set of optimal solutions of the x-parameterized problem

$$\min_{y \in Y} \; f(x, y) \tag{3a}$$

$$\text{s.t.} \; g(x, y) \geq 0. \tag{3b}$$

Problem 2 is called the upper-level (or the leader's) problem, and problem 3 is called the lower-level (or the follower's) problem, which is parameterized by x.

Overall Flow



Construction of Initial CNF

Constraint $x_1 + x_2 + \ldots + x_k \geq 1$ is equal to a CNF clause $(x_1 \vee x_2 \vee \ldots x_k)$.
$x_{i1} + x_{i2} \leq 1$ can be transformed into a CNF clause through the following steps:

- Let the $\leq$ be $\geq$ by multiplying -1 on both sides of the inequality. We have $-x_{i1} - x_{j1} \geq -1$.
- Replace x_{i1}, x_{j1} by $-(1 - \overline{x_{i1}}), -(1 - \overline{x_{j1}})$ respectively. We can get $-(1 - \overline{x_{i1}}) - (1 - \overline{x_{j1}}) \geq -1$. Here $\bar{x}$ is the negation of x, and it is easy to see $\bar{\bar{x}} = x$.
- Reorganize the terms we have $\overline{x_{i1}} + \overline{x_{j1}} \geq 1$, which can be represented by a CNF clause $(\overline{x_{i1}} \vee \overline{x_{j1}})$.

Construction of Initial CNF

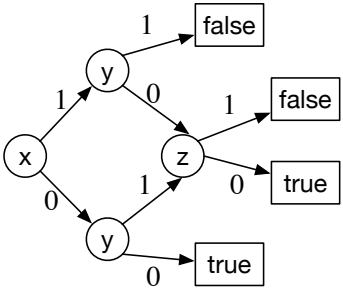
Then, the original ILP formulation can be rewritten as

$$
\begin{aligned}
\min \quad & \sum_{r_i \in p_m, r_j \in p_n, c_{ij} \in CE} C_{mn} + \alpha \sum_{s_{ij} \in SE} s_{ij}, && \text{(4a)}\\
\text{s.t.} \quad & (\overline{x_{i1}} \vee \overline{x_{i2}}), \quad \forall i \in V, && \text{(4b)}\\
& \wedge (x_{i1} \vee x_{i2} \vee x_{j1} \vee x_{j2} \vee C_{mn}), \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, && \text{(4c)}\\
& \wedge (\overline{x_{i1}} \vee x_{i2} \vee \overline{x_{j1}} \vee x_{j2} \vee C_{mn}), \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, && \text{(4d)}\\
& \wedge (x_{i1} \vee \overline{x_{i2}} \vee x_{j1} \vee \overline{x_{j2}} \vee C_{mn}), \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, && \text{(4e)}\\
& \wedge (\overline{x_{i1}} \vee \overline{x_{i2}} \vee \overline{x_{j1}} \vee \overline{x_{j2}} \vee C_{mn}), \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, && \text{(4f)}\\
& \wedge (x_{i1} \vee \overline{x_{j1}} \vee s_{ij}), \quad \forall e_{ij} \in SE, && \text{(4g)}\\
& \wedge (\overline{x_{i1}} \vee x_{j1} \vee s_{ij}), \quad \forall e_{ij} \in SE, && \text{(4h)}\\
& \wedge (x_{i2} \vee \overline{x_{j2}} \vee s_{ij}), \quad \forall e_{ij} \in SE, && \text{(4i)}\\
& \wedge (\overline{x_{i2}} \vee x_{j2} \vee s_{ij}), \quad \forall e_{ij} \in SE. && \text{(4j)}
\end{aligned}
$$

Objective Bound to Clause

Consider constraint $5x + 2y + 4z \leq 5$.

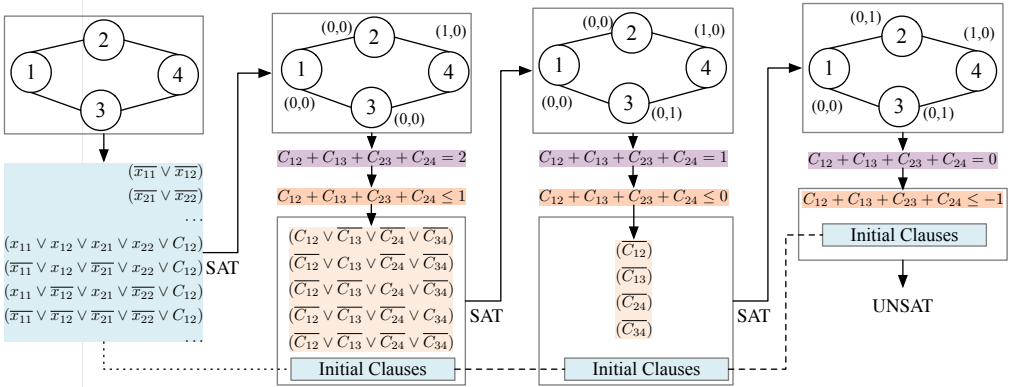
- Construct the Binary Decision Diagram.
- Extract all path to `false`.
- $x \xrightarrow{1} y \xrightarrow{1}$ `false` derives a clause $\neg x \vee \neg y$.
- $x \xrightarrow{0} y \xrightarrow{1} z \xrightarrow{1}$ `false` derives a clause $x \vee \neg y \vee \neg z$.



More complex methods can be found in MiniSat+[5].

[5] N. Eén and N. Sörensson, "Translating pseudo-boolean constraints into sat", *Journal on Satisfiability, Boolean Modeling and Computation*, vol. 2, no. 1-4, pp. 1–26, 2006.

A Toy Example



Bilevel Reformulation

The layout decomposition problem can also be formulated as a bilevel optimization problem. The upper-level optimization problem is given by

$$\begin{aligned} \min_{C,s} \quad & \sum_{r_i \in p_m, r_j \in p_n, c_{ij} \in CE} C_{mn} + \alpha \sum_{s_{ij} \in SE} s_{ij}, \\ \text{s.t.} \quad & \text{constraint (1b)} - \text{constraint (1f)}, \\ & s \in S(C), \end{aligned}$$

where $S(C)$ is the set of optimal solutions of the C-parameterized problem

$$\begin{aligned} \min_{s} \quad & \sum_{s_{ij} \in SE} s_{ij}, \\ \text{s.t.} \quad & \text{constraint (1b)} - \text{constraint (1j)}. \end{aligned}$$

How to solve the bilevel optimization problem?

- Single level reduction: the reduced single-level problem is shown exactly as the original ILP formulation.
- Nested optimization: solves the lower-level optimization problem corresponding to every upper-level member.

Our solution:

- Get the assignments of upper-level variables by solving the upper-level problem ignoring the lower-level variables.
- Solve the lower-level problem

Approximation Algorithm

Get the assigments of upper-level variables by solving

$$\min_{C} \sum_{r_i \in p_m, r_j \in p_n, c_{ij} \in CE} C_{mn},$$
$$\text{s.t. constraint (1b) – constraint (1f).}$$

In the layout decomposition problem, it can be seen as only considering conflict minimization. Once all conflict variables C_{mn} are fixed, we can perform stitch cost minimization.

Experimental Setup

- The experiments are performed on a Linux machine with eight 3.6GHz Intel Xeon CPUs and 16G DRAM.
- We rely on the open-source framework OpenMPL[6] for benchmark parsing, stitch insertion, graph simplification, etc.
- Our SAT-based optimization framework is mainly based on MiniSat+[7], a solver capable of translating pseudo-boolean constraints to CNF clauses and SAT solving.

[6] W. Li *et al.*, "Openmpl: An open-source layout decomposer",, vol. 40, no. 11, pp. 2331–2344, 2020.

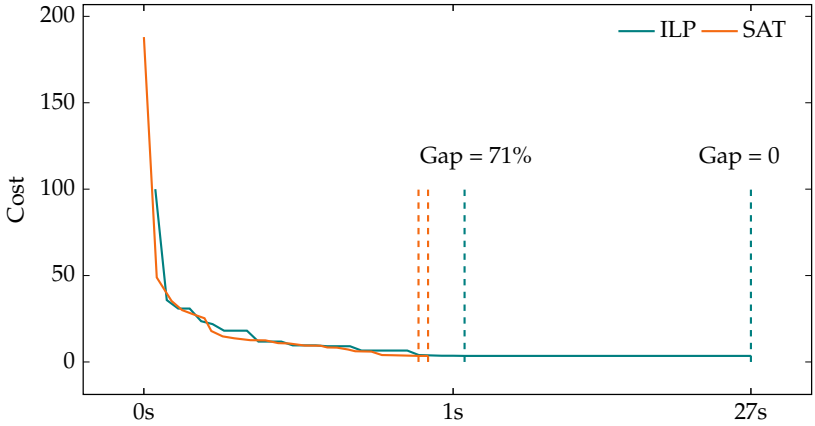
[7] N. Eén and N. Sörensson, "Translating pseudo-boolean constraints into sat", *Journal on Satisfiability, Boolean Modeling and Computation*, vol. 2, no. 1-4, pp. 1–26, 2006.

Evaluation of Our Exact Algorithm

Table: Results on ISCAS benchmarks. “RT" indicates runtime.

Circuit	ILP [Li+20]		SDP [Yu+15]		EC [Jia+17]		Ours	
	Cost	RT (s)	Cost	RT (s)	Cost	RT (s)	Cost	RT (s)
C432	0.4	0.087	0.4	0.027	0.4	0.021	0.4	0.029
C499	0.0	0.081	0.0	0.028	0.0	0.025	0.0	0.030
C880	0.7	0.083	0.8	0.032	0.7	0.026	0.7	0.034
C1355	0.3	0.062	0.3	0.039	0.3	0.036	0.3	0.044
C1908	0.1	0.063	0.1	0.054	0.1	0.051	0.1	0.056
C2670	0.6	0.109	0.6	0.084	0.6	0.079	0.6	0.090
C3540	1.8	0.153	1.8	0.112	1.8	0.100	1.8	0.123
C5315	0.9	0.217	0.9	0.147	0.9	0.130	0.9	0.156
C6288	21.4	2.999	27.3	0.434	21.4	0.300	21.4	0.606
C7552	2.3	0.402	2.3	0.235	3.1	0.208	2.3	0.255
S1488	0.2	0.082	0.2	0.051	0.2	0.043	0.2	0.057
S38417	24.4	2.352	31.6	1.445	24.4	0.771	24.4	2.072
S35932	48.0	6.451	66.0	4.248	48.7	2.034	48.0	6.069
S38584	47.6	6.533	58.5	4.195	47.7	2.216	47.6	5.915
S15850	43.7	5.854	56.3	3.821	44.0	2.075	43.7	5.415
Avg. Ratio	**1.00**	1.79	1.11	0.85	1.02	0.67	**1.00**	1.00

Analysis of Runtime Improvement



A case study on convergence of ILP and SAT-based decomposers.

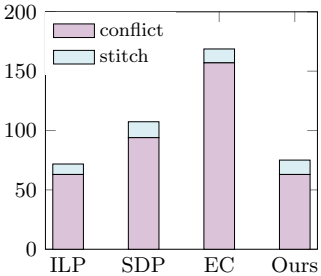
Evaluation on Large Benchmarks

Table: Layout decomposition results on ISPD19 benchmarks. “RT" indicates runtime.

Circuit	ILP [Li+20]		SDP [Yu+15]		EC [Jia+17]		Ours	
	Cost	RT (s)	Cost	RT (s)	Cost	RT (s)	Cost	RT (s)
test1_100	242.9	56.24	297.7	2.61	390.5	9.51	242.9	5.73
test5_101	452.0	78.32	549.8	5.60	629.8	16.73	452.0	10.65
test6_102	153.4	188.56	191.7	35.58	344.1	59.21	153.4	69.79
test8_100	6005.9	82.13	6206.2	32.27	6245.6	34.39	6005.9	37.55
test9_100	9223.3	128.91	9532.4	52.72	9664.0	56.08	9223.3	60.50
test10_100	10 449.5	244.93	10 910.1	85.52	11 130.6	128.96	10 449.5	103.32
Avg. Ratio	**1.00**	4.43	1.13	0.67	1.40	1.19	**1.00**	1.00
test1_101*	71.8	2370.45	107.4	19.65	168.7	71.51	75.1	6.87
test2_100*	5236.7	12 941.22	7259.4	187.31	9893.7	1404.07	5391.3	124.58
test2_102*	213.4	7810.46	526.7	304.76	593.9	2722.24	211.8	149.37
Avg. Ratio	**0.98**	167.07	1.75	2.13	2.30	13.30	1.00	1.00

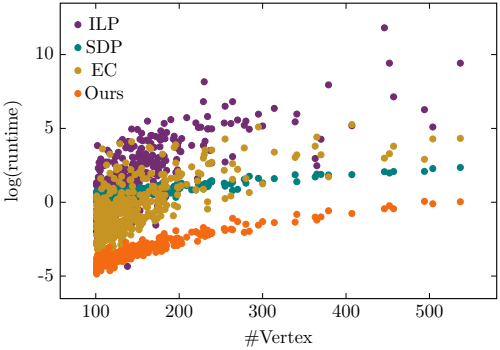
* Our approximation algorithm is enabled. For ILP, we set the timelimit to 3600s.

Evaluation of Our Approximation Algorithm



Cost breakdown of various algorithms on `test1_101`.

Evaluation of Our Approximation Algorithm



As the graphs get larger, our approximation algorithm remains effective, while the runtime of other methods can grow drastically.

THANK YOU!

HUAWEI