Layout Decomposition via Boolean Satisfiability

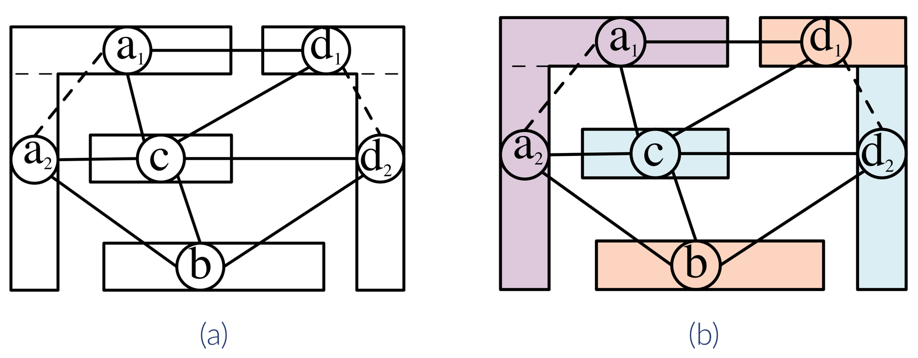
Hongduo Liu[1], Peiyu Liao[1], Mengchuan Zou[2], Bowen Pang[2], Xijun Li[2], Mingxuan Yuan[2], Tsung-Yi Ho[1], Bei Yu[1]

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

Introduction

- Multiple Patterning is used to enhance mask printability.
- Layout decomposition assigns layouts onto different masks.
- A new SAT-based exact algorithm for layout decomposition.
- A new approximation method based on bilevel reformulation.

Background



Layout decomposition can be formulated as graph coloring.

Literature Review

- Approximation Algorithms: Semidefinite Programming [5], Linear Programming [4], Heuristic methods [1].
- Exact Algorithm: Integer Linear Programming [3].

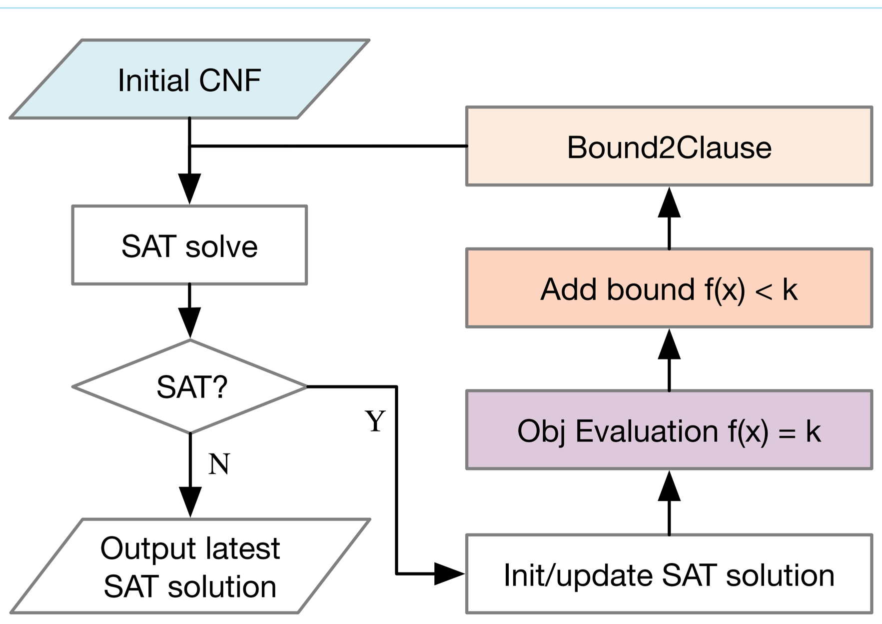
$$\min \sum_{r_i\in p_m, r_j\in p_n, c_{ij}\in CE} C_{mn} + \alpha \sum_{s_{ij}\in SE} s_{ij}, \tag{1a}$$

$$\begin{aligned}
\text{s.t. } & x_{i1} + x_{i2} \leq 1, \quad \forall i \in V, && \text{(1b)}\\
& x_{i1} + x_{i2} + x_{j1} + x_{j2} + C_{mn} \geq 1, \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, && \text{(1c)}\\
& x_{i1} - x_{i2} + x_{j1} - x_{j2} - C_{mn} \leq 1, \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, && \text{(1d)}\\
& -x_{i1} + x_{i2} - x_{j1} + x_{j2} - C_{mn} \leq 1, \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, && \text{(1e)}\\
& x_{i1} + x_{i2} + x_{j1} + x_{j2} - C_{mn} \leq 3, \quad \forall c_{ij} \in CE, r_i \in p_m, r_j \in p_n, && \text{(1f)}\\
& x_{i1} - x_{j1} + s_{ij} \geq 0, \quad \forall e_{ij} \in SE, && \text{(1g)}\\
& x_{i1} - x_{j1} - s_{ij} \leq 0, \quad \forall e_{ij} \in SE, && \text{(1h)}\\
& x_{i2} - x_{j2} + s_{ij} \geq 0, \quad \forall e_{ij} \in SE, && \text{(1i)}\\
& x_{i2} - x_{j2} - s_{ij} \leq 0, \quad \forall e_{ij} \in SE, && \text{(1j)}\\
& \text{All decision variables are binary.} && \text{(1k)}
\end{aligned}$$

Motivation

- Boolean nature of decision variables in ILP formulation ⇒ **Boolean satisfiability** ⇒ Faster convergence.
- Conflict optimization and stitch minimization are two problems nested with each other ⇒ **Bilevel Reformulation** ⇒ Tighter Approximation.

Overall Flow



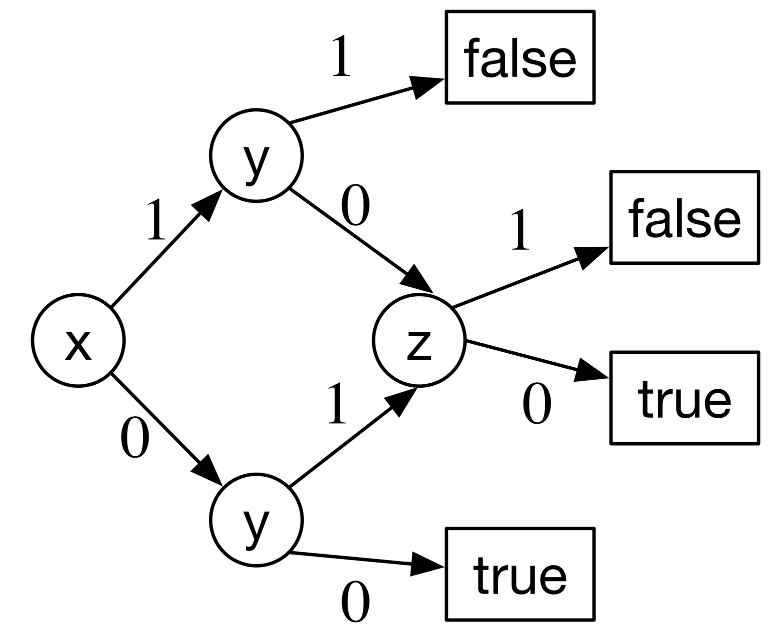
Construction of Initial CNF

The 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.
- Reorganize the terms we have $\overline{x_{i1}} + \overline{x_{j1}} \geq 1$, which can be represented by a CNF clause $(\overline{x_{i1}} \vee \overline{x_{j1}})$.

Objective Bound to Clause

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



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

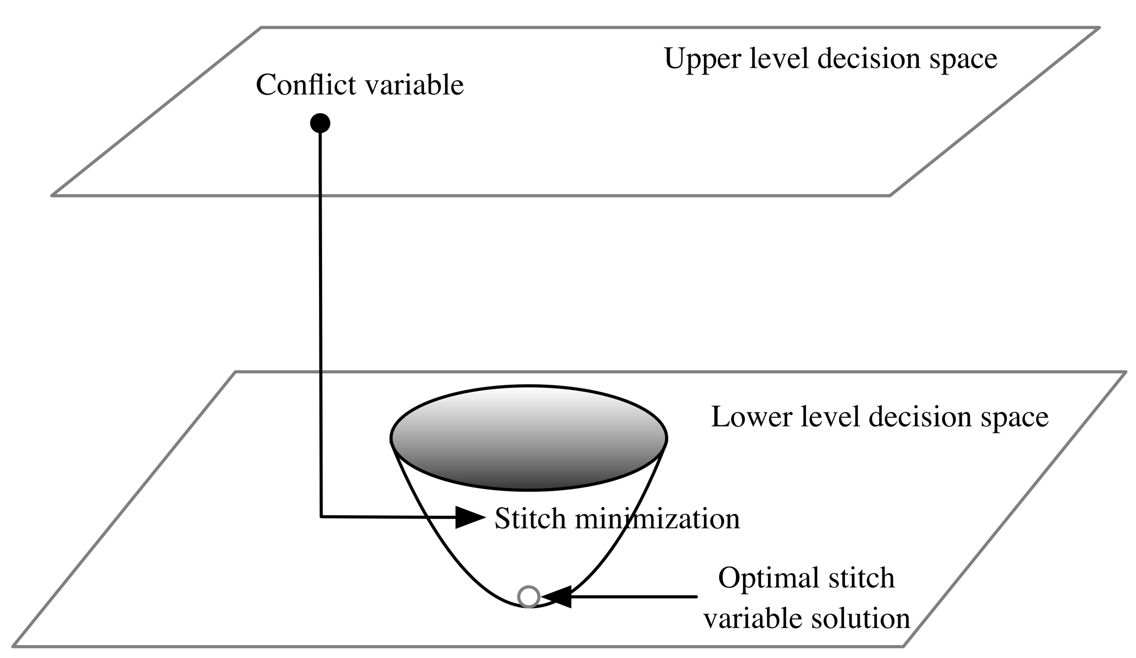
Bilevel Reformulation

The upper-level optimization problem is given by

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

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

$$\min_{s} \sum_{s_{ij}\in SE} s_{ij},$$
$$\text{s.t. constraint (1b) – constraint (1j)}.$$



Approximation Algorithm

- Conflict Minimization without considering stitch cost.
- Stitch Minimization with fixed conflict variables.

Evaluation of Our Exact Algorithm

Table 1. Results on ISCAS benchmarks. "RT" indicates runtime.

Circuit	ILP [3]		SDP [5]		EC [2]		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

Our SAT-based layout decomposer obtains optimal solutions as ILP but converges faster.

Evaluation on Large Benchmarks

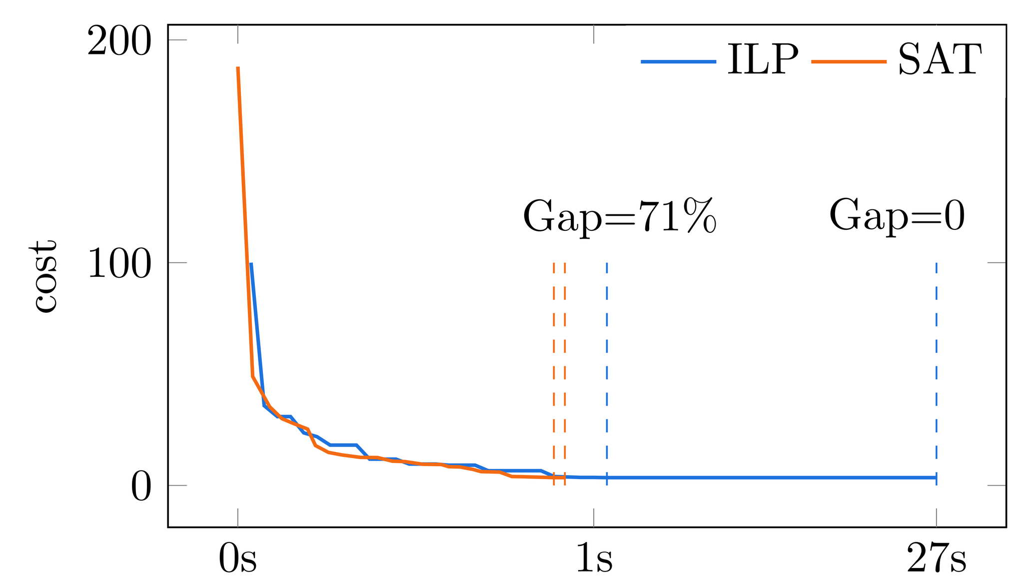
Table 2. Layout decomposition results on ISPD19 benchmarks. "RT" indicates runtime. For ILP, we set the timelimit to 3600s.

Circuit	ILP [3]		SDP [5]		EC [2]		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	**10449.5**	244.93	10910.1	**85.52**	11130.6	128.96	**10449.5**	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**	12941.22	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.

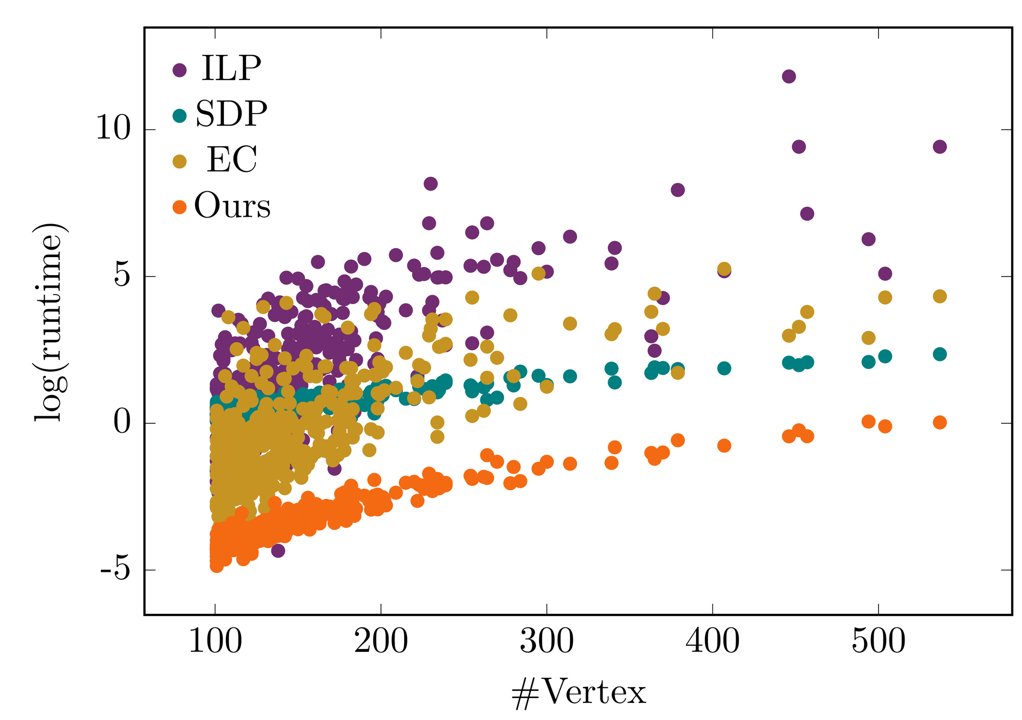
Runtime Improvement of SAT-based Decomposer

- The scale of SAT problems remains controllable as original ILP constraints are all cadinality constraints.
- Optimality is easier to prove.



The first dashed line indicates when an optimal solution is found, and the second indicates when the optimality is proven.

Approximation Algorithm Scalability



Conclusion

- The SAT-based exact algorithm can achieve faster convergence than ILP without losing optimality.
- Layout decomposition can be seen as a bilevel optimization problem. The proposed approximation is tighter and shows better scalability.

References

[1] Shao-Yun Fang, Yao-Wen Chang, and Wei-Yu Chen. A novel layout decomposition algorithm for triple patterning lithography. pages 1185–1190, 2012.

[2] Iris Hui-Ru Jiang and Hua-Yu Chang. Multiple patterning layout decomposition considering complex coloring rules and density balancing. 36(12):2080–2092, 2017.

[3] Wei Li, Yuzhe Ma, Qi Sun, Lu Zhang, Yibo Lin, Iris Hui-Ru Jiang, Bei Yu, and David Z Pan. Openmpl: An open-source layout decomposer. 40(11):2331–2344, 2020.

[4] Yibo Lin, Xiaoqing Xu, Bei Yu, Ross Baldick, and David Z Pan. Triple/quadruple patterning layout decomposition via linear programming and iterative rounding. 16(2):023507, 2017.

[5] Bei Yu, Kun Yuan, Duo Ding, and David Z Pan. Layout decomposition for triple patterning lithography. 34(3):433–446, 2015.

hdliu21@cse.cuhk.edu.hk