ROAD: Routability Analysis and Diagnosis Framework Based on SAT Techniques

Dongwon Park
Electrical and Computer Engineering,
UC San Diego
La Jolla, California
dwp003@ucsd.edu

Ilgweon Kang
Cadence Design Systems, Inc.
San Jose, California
ikang@cadence.com

Yeseong Kim
Computer Science and Engineering,
UC San Diego
La Jolla, California
yek048@ucsd.edu

Sicun Gao
Computer Science and Engineering,
UC San Diego
La Jolla, California
sicung@ucsd.edu

Bill Lin
Electrical and Computer Engineering,
UC San Diego
La Jolla, California
billin@ucsd.edu

Chung-Kuan Cheng
Computer Science and Engineering,
UC San Diego
La Jolla, California
ckcheng@ucsd.edu

ABSTRACT

Routability diagnosis has increasingly become the bottleneck in detailed routing for sub-10nm technology due to the limited tracks, high density, and complex design rules. The conventional ways to examine the routability of detailed routing are ILP- and SAT-based techniques. However, once we identify the routability, the diagnosis remains an open problem for physical designers. In this paper, we propose a novel framework, called *ROAD*, which diagnoses explicit reasons for routing failures. The proposed *ROAD* framework utilizes a diagnosis-friendly SAT formulation to represent design's layout and diagnoses the routability with SAT solving techniques. Based on the diagnosis, *ROAD* provides human-interpretable explanations for conflicted routing conditions. To show the practical value of our framework, we also generate comprehensive test-sets that enable exhaustive exploration of layouts based on Rent's rule. We demonstrate that *ROAD* successfully examines conflict causes for diverse pin layouts. Throughout extensive diagnosis, we also present several key findings for design failure. *ROAD* performs routability diagnosis within 2 minutes on average for 90 grids test-sets, while diagnosing the exact causes of routing failures in terms of congestion and conditional design rules.

KEYWORDS

Physical Design, Detailed Routing, Routability, Diagnosis, SAT, MUS, Longest Path, BCP, PIG

ACM Reference Format:

Dongwon Park, Ilgweon Kang, Yeseong Kim, Sicun Gao, Bill Lin, and Chung-Kuan Cheng. 2019. ROAD: Routability Analysis and Diagnosis Framework Based on SAT Techniques. In *2019 International Symposium on Physical Design (ISPD '19)*, April 14–17, 2019, San Francisco, CA, USA. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3299902.3309752>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ISPD '19, April 14–17, 2019, San Francisco, CA, USA

© 2019 Association for Computing Machinery.

ACM ISBN 978-1-4503-6253-5/19/04...\$15.00

<https://doi.org/10.1145/3299902.3309752>

1 INTRODUCTION

As manufacturing technology nodes are continuously evolved into sub-10nm, the detailed routing has been more challenging [1]. One of the major difficulties in the detailed routing is exposed by the resolution limitations coming from the diffraction limit of optical lithography which has 193nm (i.e., 193i) wavelength [11, 12, 23]. Although recent multi-patterning techniques [4, 18, 19] enable to deliver 10nm and sub-10nm technology nodes to foundries, they induce more complex conditional design rules for manufacturability which are new hurdles in the detailed routing procedure [2]. Routability becomes a critical bottleneck in detailed routing due to less number of routing tracks, higher pin density, and smaller pin geometry [23, 25]. Moreover, many layout revisions are urgently required during engineering change order (ECO) procedure since the mismatch between a global-route congestion map and detailed-route violations (i.e., design-rule violations) is growing [15].

For the detailed routability problem, several ILP-based approaches are proposed to achieve design rule-correct routability analysis [10, 12, 14]. SAT (Boolean satisfiability) techniques are widely utilized for efficient routability estimation [3, 9, 22], but they only focus on routing for standard cell designs, rather than layout design. Recently, a fast routability analysis technique based on a SAT formulation for the detailed routing is proposed [16]. However, even if we achieve the routability analysis successfully, the diagnosis for the routing failure remains an open question in the detailed routing.

In this paper, we propose a novel framework which diagnoses explicit reasons for the detailed routing failure while efficiently estimating the routability of layout architectures. To identify conflict geometry and design rules, *ROAD* converts design constraints of pin layouts to SAT clauses. It then diagnoses design conflicts based on several SAT techniques, such as Minimal Unsatisfiable Subset (MUS), Boolean Constraint Propagation (BCP) and Partial Implication Graph (PIG). Using the MUS technique [6, 27] identifying a minimum unsatisfiable subset of conflicting clauses, we first extract partial layouts that include the causes of conflicts in the unroutable case. BCP [17] is a simplification procedure for clauses using unit clauses. We iteratively perform BCP procedures until either there are no more remained unit clauses or any conflict is found. *ROAD* then represents variable nodes and clause edges generated by the

BCP procedure as a directed acyclic graph (DAG). This subprocedure is conducted by the PIG technique [17], and we exploit the PIG to produce diagnosis results of the conflict layouts.

The main contributions of this paper are listed as follows:

- We propose a fast and intuitive routability diagnosis framework, called *ROAD*, by utilizing the routability analysis technique based on SAT formation [16].
- *ROAD* provides comprehensive information of unroutable layouts, e.g., conflict geometries and design rules. This information is useful to layout designers for fast trouble-shooting and to design-rule managers for fine-grained decisions of design rule priority.
- We demonstrate that our diagnosis procedures are suitable to identify the causes of unroutable pin layouts.
- *ROAD* utilizes an automated technique [16] which generates pin layout testsets based on *Rent's Rule* [26]. It enables comprehensive exploration of conflicting cases which layout designers encounter in practice.

Our *ROAD* framework allows designers to diagnose the causes of routing failure cases and provides useful insights with enhancement of understanding for the unroutability. In this paper, we also present several key findings. We find that layout conflicts happen due to two main factors: (i) pin-shape pattern and (ii) routing resource shortages (i.e., routing congestion). The pin-shape pattern is the main concern of pin accessibility problems. When the resource is insufficient, technology limitation may hinder layout routability, e.g., due to insufficient tracks and metal layers about pin layout. We find the exact geometry and design rules related to the conflict, and thus it can be a guideline how to resolve the conflict.

2 ROAD : ROUTABILITY ANALYSIS AND DIAGNOSIS

In this section, we describe the detailed procedures of *ROAD*: (i) Overview of *ROAD*, (ii) SAT formulation, (iii) Diagnosis Procedure, (iv) Diagnosis Result Report and (v) Automated Pin Layout Generation.

2.1 Overview of ROAD

Figure 1 shows the overview of the *ROAD* framework. The *ROAD* framework first performs the routability analysis based on the SAT formulation presented in [16]. If it is unroutable, we diagnose causes of the unroutability. As the first step of the diagnosis procedure, we extract MUS (Minimal Unsatisfiable Subset), i.e., the subset of clauses including the conflict region and design rules which require a careful attention during the diagnosis. In the next initial propagation phase, we construct geometric information of switch-boxes by simplifying the extracted MUS. The key diagnosis steps are the iterative decision and propagative procedures. We decide appropriate via positions (Decision) and execute BCP (Boolean Constraint Propagation) with partial assignments (Propagation). The BCP procedure generates PIGs (Partial Implication Graphs), i.e., a DAG including variable nodes and clause edges. By combining multiple PIGs, we obtain a designer-friendly interpretation of the conflict geometry and design rules.

2.2 SAT Formulation

SAT formulation is derived on top of the multi-commodity network flow theory and conditional design rules [8, 16]. Among several

Figure 1: An overview of *ROAD* framework.

Table 1: Notations for SAT formulation.

Term	Description
$G(V, E)$	Three-dimensional (3-D) routing graph
V	Set of vertices in the routing graph G
V_i	Set of vertices in i^{th} metal layer of the routing graph G
v	A vertex at the coordinate (x_v, y_v, z_v)
v_d	0-1 indicator if d -directional adjacent vertex about v
$a(v)$	Set of adjacent vertices of v
E	Set of edges in the routing graph G
$e_{v,u}$	An edge between v and u , $u \in a(v)$
N	Set of multi-pin nets in the given routing box
n	n^{th} multi-pin net
s^n	A source of n
T^n	Set of sinks in n
t_m^n	m^{th} sink of n
f_m^n	m^{th} commodity flow of n heading to t_m^n
$e_{v,u}^n$	0-1 indicator if $e_{v,u}$ is used for n
$f_m^n(v, u)$	0-1 indicator if $e_{v,u}$ is used for commodity t_m^n
$m_{v,u}$	0-1 indicator if there is a metal segment on $e_{v,u}$
$g_{d,v}$	0-1 indicator if v forms d -side EOL of a metal segment

Figure 2: Constraint relations of the proposed SAT formulation.

kinds of SAT encoding methods, our formulation is a kind of the “muldirect” method, which is a directly encoding method of constraint satisfaction problem. It allows multiple assignments for the multi-commodity network to represent the detailed routing problem [20, 24].

The formulation consists of three subgroups, (i) *flow formulation*, (ii) *design rule formulation* and (iii) *layout structure map*. The notations and relations between constraints and variables are shown in Table 1 and Figure 2. The formulated subgroups create the foundation for the SAT framework. The SAT framework identifies if there are violations for the formulation, deciding the routability of the given layout.

Most of our formulation is composed by the Exactly-One (EO) constraint which is related to the flow formulation, or the At-Most-One (AMO) constraints which are related to design rules and exclusiveness. ROAD adopts the “pair-wise” and “commander” encoding methodology for EO and AMO constraints [21].

2.2.1 Flow Formulation (F). We use a multi-commodity flow foundation to represent routing flows for a given layout. We achieve more efficient formulation by reducing CNF (conjunctive normal form) representation, i.e., regarding the flows as an undirected graph (1). This simplification still holds the correct representation since we only need to consider the source-sink connectivity.

$$f_m^n(v, u) = f_m^n(u, v), \text{ No Direction} \quad (1)$$

Commodity Flow Conservation (CFC). Logical Expressions (3, 4) represent the constraint of CFC, describing the commodity flow constraint on each vertex at a per-net and per-commodity granularity. F_{CFC} (2) is either F_{CFC_1} or F_{CFC_2} depending on the vertex type.

$$F_{CFC}(v, n, m) = \begin{cases} F_{CFC_1}(v, n, m), & \text{if } v \neq s^n, t_m^n \\ F_{CFC_2}(v, n, m), & \text{if } v = s^n, t_m^n \end{cases} \quad (2)$$

F_{CFC_1} (3) represents the flow feasibility, i.e., either no flow passes or the flow of commodity m is connected between a pair of edges $(e_{v,p}$ and $e_{v,q})$, when a vertex v is neither a source nor a sink.

$$F_{CFC_1}(v, n, m) = \bigwedge_{u \in a(v)} \neg f_m^n(v, u) \vee \bigvee_{p, q \in a(v), p \neq q} (f_m^n(v, p) \wedge f_m^n(v, q) \wedge \bigwedge_{u \in a(v), u \neq p, u \neq q} \neg f_m^n(v, u)), \forall v \in V, \forall n \in N, \forall t_m^n \in T^n \quad (3)$$

F_{CFC_2} (4) represents the flow feasibility of the commodity m on a vertex v using an EO constraint [21] of commodity flow indicators $f_m^n(v, u)$, when v is either a source or a sink.

$$F_{CFC_2}(v, n, m) = \text{EO} \left(\{f_m^n(v, p) \mid p \in a(v)\} \right), \forall v \in V, \forall n \in N, \forall t_m^n \in T^n \quad (4)$$

Exclusiveness Use of Vertex (EUV). Logical Expressions (6, 7) represent the constraint of EUV, ensuring that there are no intersecting nets on any vertex. F_{EUV} (5) is either F_{EUV_1} or F_{EUV_2} depending on the vertex type as well.

$$F_{EUV}(v) = \begin{cases} F_{EUV_1}(v), & \text{if } v \neq s^n, t_m^n \\ F_{EUV_2}(v), & \text{if } v = s^n, t_m^n \end{cases} \quad (5)$$

F_{EUV_1} (6) represents the exclusiveness feasibility on each vertex v using an AMO net constraint [21], when the vertex v is neither a source nor a sink.

$$F_{EUV_1}(v) = \text{AMO} \left(\left\{ \bigvee_{p \in a(v)} e_{v,p}^n \mid n \in N \right\} \right), \forall v \in V \quad (6)$$

F_{EUV_2} (7) represents the exclusiveness feasibility on each vertex v using an EO constraint of edge indicators $e_{v,u}^n$, when v is either a source or sink.

$$F_{EUV_2}(v) = \text{EO} \left(\{e_{v,p}^n \mid p \in a(v), n \in N\} \right), \forall v \in V \quad (7)$$

Figure 3: An example to determine GV.

Edge Assignment (EA). Logical Expression (8) represents the constraint of Edge Assignment (EA), ensuring the multi-commodity flow of net n . If the flow indicator $f_m^n(v, u)$ is 1, then the edge $e_{v,u}$ is 1.

$$F_{EA}(e_{v,u}, n, m) = e_{v,u}^n \vee \neg f_m^n(v, u), \forall e_{v,u} \in E, \forall n \in N, \forall t_m^n \in T^n \quad (8)$$

Metal Segment (MS). Logical Expression (9) represents the constraint of Metal Segment (MS) using an EO constraint, i.e., either exclusiveness use of edge for certain net or no edge uses. The expression is a part of the AMO commander encoding [21] whose commander is the metal segment.

$$F_{MS}(e_{v,u}) = \text{EO} \left(\{ \neg m_{v,u} \} \cup \{ e_{v,u}^n \mid n \in N \} \right), \forall e_{v,u} \in E \quad (9)$$

The flow feasibility F is represented by a conjunction of each sub-formulation as Expression (10).

$$F = \bigwedge_{v \in V} \left(F_{EUV}(v) \wedge \bigwedge_{n \in N} \bigwedge_{t_m^n \in T^n} F_{CFC}(v, n, m) \right) \wedge \bigwedge_{e_{v,u} \in E} \left(F_{MS}(e_{v,u}) \wedge \bigwedge_{n \in N} \bigwedge_{t_m^n \in T^n} F_{EA}(e_{v,u}, n, m) \right) \quad (10)$$

2.2.2 Design Rule Formulation (D). The detailed routing procedure, several critical conditional-design rules come from the gap between the technology node and the diffraction limitation of optical lithography source. The gap perpetually increases because the technology node is continuously shrinking while the resolution limitation is fixed. Our proposed SAT formulation adopts representative conditional design rules such as *Minimum Area Rule*, *End-of-Line Spacing Rule*, and *Via Rule* from [10, 14, 16].

Geometric Variable (GV). Logical Expressions (11-13) represent the feasibility of Geometric Variable (GV) determined by the EOL of a metal segment as shown in Figure 3. Each geometric-directional feasibility is determined by a linear representation, $g_{d,v}$, converted from a logical AND of metal segments (i.e., $D_{GV_L}(v)$ is converted from $g_{L,v} = \neg m_{v_L,v} \wedge m_{v,v_R}$) [12, 16]. The geometry feasibility D_{GV} (13) is the conjunction of each directional GV representation $D_{GV_L}(v)$, $D_{GV_R}(v)$, $D_{GV_F}(v)$, $D_{GV_B}(v)$ for all vertices. The front- and back-directional GV are derived by changing L and R to F and B , respectively.

$$D_{GV_L}(v) = (g_{L,v} \wedge \neg m_{v_L,v} \wedge m_{v,v_R}) \vee (\neg g_{L,v} \wedge \neg m_{v,v_R}) \vee (\neg g_{L,v} \wedge m_{v_L,v}), \forall v \in V_2 \quad (11)$$

$$D_{GV_R}(v) = (g_{R,v} \wedge m_{v_L,v} \wedge \neg m_{v,v_R}) \vee (\neg g_{R,v} \wedge \neg m_{v_L,v}) \vee (\neg g_{R,v} \wedge m_{v,v_R}), \forall v \in V_2 \quad (12)$$

$$D_{GV} = \bigwedge_{v \in V_2} \left(D_{GV_L}(v) \wedge D_{GV_R}(v) \right) \wedge \bigwedge_{v \in V_1, V_3} \left(D_{GV_F}(v) \wedge D_{GV_B}(v) \right) \quad (13)$$

Figure 4: An example of MAR.

Figure 5: An example of EOL.

Based on GV, we formulate conditional design rules as follows.

Minimum Area Rule (MAR). Logical Expressions (14,15) represent MAR using a simple AMO constraint for GVs. Figure 4 depicts an example of MAR. Each disjoint metal segment should be no smaller than the resolution limitation. In our formulation, a metal segment must cover at-least three vertices. The MAR feasibility formulation for M_1 and M_3 ($\mathbf{D}_{MAR_{FB}}(v)$) are derived by changing L and R to F and B , respectively. The MAR feasibility $\mathbf{D}_{MAR}$ (15) is the conjunction of $\mathbf{D}_{MAR_{LR}}$ and $\mathbf{D}_{MAR_{FB}}$.

$$\mathbf{D}_{MAR_{LR}}(v) = \text{AMO}(g_{L,v}, g_{R,v}, g_{L,v_R}, g_{R,v_R}), \forall v \in V_2 \quad (14)$$

$$\mathbf{D}_{MAR} = \bigwedge_{v \in V_2} \mathbf{D}_{MAR_{LR}}(v) \wedge \bigwedge_{v \in V_1, V_3} \mathbf{D}_{MAR_{FB}}(v) \quad (15)$$

End-of-Line(EOL) Space Rule. Logical Expressions (16,17) represent EOL Spacing Rule related to the distance between any of two opposite EOLs (i.e., “tips”) using simple AMO constraints for GVs. The minimum distance between each of the tips depends on the design rule, we assume that it has to be larger than two Manhattan distance (i.e., L^1 Norm) as depicted in Figure 5. The other cases, i.e., $\mathbf{D}_{EOL_L}$, $\mathbf{D}_{EOL_F}$, and $\mathbf{D}_{EOL_B}$ are derived in similar ways to $\mathbf{D}_{EOL_R}$.

$$\mathbf{D}_{EOL_R}(v) = \text{AMO}(g_{R,v}, g_{L,v_{FR}}) \wedge \text{AMO}(g_{R,v}, g_{L,v_{BR}}) \wedge \text{AMO}(g_{R,v}, g_{L,v_R}, g_{L,v_{RR}}), \forall v \in V_2 \quad (16)$$

$$\mathbf{D}_{EOL} = \bigwedge_{v \in V_2} (\mathbf{D}_{EOL_L}(v) \wedge \mathbf{D}_{EOL_R}(v)) \wedge \bigwedge_{v \in V_1, V_3} (\mathbf{D}_{EOL_F}(v) \wedge \mathbf{D}_{EOL_B}(v)) \quad (17)$$

Via Rule (VR). Logical Expression (18) represents VR related to restriction rules of inter-layer via (i.e., via) locations. In our formulation, the distance between two vias should be larger than $\sqrt{2}$ Euclidean distance (i.e., L^2 Norm) as shown in Figure 6(a) (i.e., via-to-via spacing rule). The via placement on the top of another via is not allowed as shown in Figure 6(b) (i.e., stacked-via regulation). The expression of VR feasibility can be represented by the conjunction of simple AMO constraints for GVs as well.

$$\mathbf{D}_{VR}(v) = \text{AMO}(m_{v,v_U}, m_{v_R,v_{UR}}, m_{v_B,v_{UB}}, m_{v_{BR},v_{UBR}}) \wedge \text{AMO}(m_{v_D,v}, m_{v,v_U}), \forall v \in V \quad (18)$$

The design rule correctness $\mathbf{D}$ is represented by the conjunction of each design rule subset (19).

Figure 6: VR rule. (a) via-to-via spacing rule. (b) stacked-via regulation.

$$\mathbf{D} = \mathbf{D}_{GV} \wedge \mathbf{D}_{MAR} \wedge \mathbf{D}_{EOL} \wedge \bigwedge_{v \in V} \mathbf{D}_{VR}(v) \quad (19)$$

2.2.3 Layout Structure Map (L). Layout structure map is composed by unit clauses determined by the geometry of the switchbox, e.g., the information of prohibited power rail track in M_2 , the pin allocation in M_1 and the boundary condition of the switchbox.

2.2.4 Design Rule-Correct Routability Analysis. The design rule-correct routability $\mathbf{R}$ can be derived by combining all sub formation $\mathbf{F}$, $\mathbf{D}$ and $\mathbf{L}$ as shown in Expression (20).

$$\mathbf{R} = \mathbf{F} \wedge \mathbf{D} \wedge \mathbf{L} \quad (20)$$

Through the SAT analysis, we evaluate if the expression is true, i.e., all the given design rules are correct. When it calls “unroutable”, we proceed our diagnosis procedure discussed in the next section.

2.3 Diagnosis Procedure

The diagnosis procedure consists of four phases, (i) *MUS Extraction*, (ii) *Initial Propagation*, (iii) *Decision*, and (iv) *Propagation phase*. The key diagnosis steps are the iterative Decision and Propagation until we face conflicting clauses as shown in Figure 1. The diagnosis using clauses is similar to one of the longest-path-search problem for a DAG [5, 13]. In this paper, the DAG is defined as follows:

Definition 2.1. (Diagnosis DAG) The DAG $H(U, D)$ is defined as

- (1) D is a set of clauses in an MUS.
- (2) $U = \{U_b, U_s, U_p\} \subset V$ is a set of variables involving the conflict. U_b is a set of input layout variables ($U_b \subset I$), U_s is the variables selected by the decision procedure of an iteration and U_p is the variables determined in an iteration of BCP.
- (3) $H(U, D) = \text{PIG}(U_b \cup U_s \cup U_p, D)$, where $\text{PIG}(U, D)$ is generated in each propagation step with BCP.

2.3.1 MUS Extraction Phase. For the unroutable pin layout, we first extract the MUS which is the minimal unsatisfiable set if all its proper subsets are satisfiable. It significantly contributes the diagnosis time reduction by replacing the diagnosis target with the very small number of clauses. ROAD uses one of the state-of-the-art MUS Extractor *MUSer2* [7].

2.3.2 Initial Propagation Phase. Based on the MUS, ROAD executes initial BCP using the clauses of $\mathbf{L}$. It restricts the design-conflict region of the unroutable layout. Figure 7 shows an example (9_13_100)¹ of restricted conflict region. The estimated conflict region is the target range of the next iterative decision and propagation phases.

¹The naming convention for a switchbox is $\langle 1 \rangle _ \langle 2 \rangle _ \langle 3 \rangle$, where $\langle 1 \rangle _ \langle 2 \rangle _ \langle 3 \rangle$ are the number of vertical tracks, horizontal tracks and pin density, respectively.

Figure 7: An example of unroutable layout (9_13_100).

Figure 8: Path length. (a) 2 @ (3,8,1). (b) 1 @ (3,9,1), (c) 4 @ (3,10,1).

2.3.3 Decision with Longest-path Search (DLS) Phase. In DLS phase, ROAD selects a certain via position for the next propagation phase. A technical challenge for the explanation of practical design failure is that there are too many ways to explain the design-rule conflicts. We address this issue by adopting a longest-path search algorithm. The underlying intuition is that the longest routing path usually represents the causes of the routability failure in a comprehensive manner.

Figure 8 shows that the 9_13_100 layout has three possible via positions ($m_{(3,8,1),(3,8,2)}$, $m_{(3,9,1),(3,9,2)}$ and $m_{(3,10,1),(3,10,2)}$) in the first iteration. The designer would consider that the short paths depicted in Figure 8(a), (b) are obvious to avoid, while the longest path in Figure 8(c) provides the intrinsic problem caused by the design rules. The DLS phase thus selects a via which initiates the maximum path length of $H(U, D)$.

2.3.4 Propagation with True Assignment (PTA) Phase. We then execute the propagation by assigning 1 for the selected via position. As illustrated in Figure 9, there are three kinds of variables assigned with 1 for a certain via position. For the vertex v , the variables between Pin_j^2 to M_1 are set as True (step (1)). The via elements between v and vertex v_U are also set as True (step (2)). Finally, the M_2 elements connected with v_U are set as True (step(3)). We also choose the direction of variables on the M_2 layer so that its PIG can have the longest path as well.

Figure 10 shows an example of the PTA results for via#1 of 9_13_100 layout with its PIG. Three via positions ($m_{(4,9,1),(4,9,2)}$, $m_{(4,10,1),(4,10,2)}$ and $m_{(3,10,2),(3,10,3)}$) are blocked by the VR rule. The direction of the variables at the M_2 layer is determined by “left” to find the longest path.

² Pin_j is a supernode which covers vertices of the pin on M_1 .

Figure 9: PTA phase for M_1 vertex v .

Figure 10: Via#1 of 9_13_100. (a) PTA result. (b) Some part of PIG.

Figure 11: PFA phase for v . (a) Via-to-via spacing. (b) Stacked-via. (c) Vias in the same pin. (d) In-layer variables with direction against PTA.

2.3.5 Propagation with False Assignment (PFA) Phase. The PFA phase executes the propagation with 0 value for the selected via position. The false assignment happens to cover four reasons: via-to-via spacing void, stacked-via void, void of vias in the same pin and void of the in-layer variables with direction against forcing. Figure 11 describes each reason.

Figure 12: Via#1 of 9_13_100. (a) PFA result. (b) Some part of PIG for in-layer element with direction against PTA. (c) Some part of PIG for via-to-via spacing void.

By the via-to-via spacing PFA case as depicted in Figure 11(a), vias within $\sqrt{2} L^2$ Norm from the v are blocked. The stacked-via PFA case (Figure 11(b)) blocks via positions which are on the top of another via. In Figure 11(c), only one via is allowable in a pin. In-layer variables, which direction is against PTA, are blocked from Figure 11(d) rule.

Figure 12 illustrates an example with the via at the first iteration (i.e., via#1 = $m_{(3,10,1)}(3,10,2)$). The elements between $(3,10,2)$ and $(4,10,2)$ are blocked by PFA phase as depicted in Figure 12(a). Figure 12(b)-(c) are some parts of PIG related to this PFA case. The detailed blocking procedure of M_2 layer's elements is found in Figure 12(b). With PTA phase result, PFA blocks two via positions in $\text{pin}11$ and so the only $m_{(4,8,1)}(4,8,2)$ is selected as also shown in Figure 12(c). The selected via is accepted as the output of the DLS phase in the following iteration.

2.4 Diagnosis Result Report

We perform the iterative procedures until it has a conflict. ROAD provides precise information for conflicts in terms of geometry and conflicted conditional design rules. Figure 14 shows that, e.g.,

Figure 13: Pin layout configuration.

Figure 14: Diagnosis result of 9_13_100.

the conflict of the 9_13_100 layout occurs at the PFA phase when the path length is 4, i.e., the fourth iteration. The cause of the unroutability is the conflict between VR and CFC rules due to $f_0^7(\text{pin}0)(7,10,1)$ variable. From VR rule, all possible via locations in $\text{pin}0$ vertices are blocked in the PFA phase (i.e., $f_0^7(\text{pin}0)(7,10,1)$ is set as False). However, since at-least one vertex must have a via for $\text{pin}0$ connection due to CFC rule (i.e., $f_0^7(\text{pin}0)(7,10,1)$ is set as True), we can identify the variable values are conflicted each other.

2.5 Automated Pin Layouts Generation

ROAD has a feature of an automated test-sets generation which creates practical pin layouts based on *Rent's Rule* [26]. For a test-set generation, we specify the inputs as the number of vertical tracks (V-tracks) and horizontal tracks (H-tracks), and pin density for the switchbox. The number of Grids is defined by the number of intersection between V-tracks and H-tracks. As depicted in Figure 13, we assume 7-track cell library, which means five tracks for each routing row are eligible for detailed routing, and total four metal layers are in the switchbox. M_1 is only for pin placement and each pin cover 3-5 vertices. We don't use any detailed-placement refinements (e.g., cell padding) in the pin layout generation since this refinement affects quantification results.

3 EXPERIMENTS

In this section, we describe the experimental results of ROAD, i.e., the analysis and diagnosis. ROAD provides the routability analysis through the routing assessment using SAT formulation and the causes of unroutable layouts through the novel diagnosis procedure. The diagnosis procedure is built on the longest path search-based decision and BCP-based propagation technique. Our ROAD is implemented and validated on 2.6GHz Intel Xeon E5-2640 Linux workstation with 128GB memory.

Table 2: Unroutable layout examples. #N=#Nets, #P=#Pins.

SwitchBox	Spec.		SAT Formulation		MUS #Clauses	Conflict Cause
	#N	#P	#Variable	#Clauses		
11_7_80	6	12	16,776	378,551	215	Simple-CP
9_19_100	13	29	101,943	5,368,125	234	
9_13_100	10	22	54,884	2,280,223	272	Propagated-CP
12_13_70	10	23	79,471	3,478,371	446	
11_7_90	7	14	21,014	580,925	514	Routing Congestion
7_7_100	4	9	7,972	110,389	1,194	
7_13_100	7	15	24,236	595,898	4,060	
15_7_90	9	20	45,782	1,725,676	6,579	
19_13_70	15	33	171,092	11,287,222	19,375

Figure 15: 9_13_100 : Simple-CP pattern (3-3-n-3-3).

3.1 Unroutable layout classification

Based on our diagnosis procedure, we show a couple of findings regarding the causes of the conflicts. Table 2 summarizes unroutable layouts with their conflict causes. The causes are classified into the three categories: *Simple*-, *Propagated-Conflict Pin-shape (CP) Pattern* and *Routing Congestion*. In our SAT formulation, the number of clauses is determined by the number of V-tracks, H-tracks, nets, and pins. On the other hand, the number of MUS is highly related to the conflict causes rather than the specification of the layout.

3.1.1 Simple-CP Pattern. We classify an unroutable layout into the Simple-CP type if it has an intrinsic CP pattern, while it is obviously infeasible without any further investigation in terms of pin accessibility. The conflict of the Simple-CP type is only related to the region inside the pattern. For example, the layout in Figure 15 is a Simple-CP type since it has a 3-3-n-3-3 CP pattern, where $n > 3$.

3.1.2 Propagated-CP pattern. We classify an unroutable layout into the Propagated-CP type if it has no intrinsic CPs but a CP can be found after multiple-sequential propagations. For example, as shown in Figure 16(a), the 12_13_70 layout has no intrinsic CPs to explicitly identify from the pin-shape. However, we can find a simple “3-3-3” CP pattern after the propagation of the fourth via as depicted in Figure 16(b) since the via at (5,7,1) is blocked by the EOL rule.

3.1.3 Routing Congestion. The unroutable layout that has no violations in locating vias on the pins in M_1 is the Routing Congestion type. The cause of this type is related to the lack of routing resources. For example, Figure 17(b) shows the 15_7_90 layout has no violations in locating vias between M_1 and M_2 layers. This means that there must be a conflict in either M_2 / M_3 layers or vias between M_2 and M_3 layers. Net 5 of the 15_7_90 is one of the in-switchbox connections (i.e., Pin 1 and 4 in net 5). Pin 1 and 4 should be connected through the inside routing tracks. However, all tracks of

Figure 16: An example of propagated-CP pattern. (a) Given layout. (b) Propagated-CP pattern(3-3-3). **Figure 17: An example of Routing Congestion. (a) Given layout. (b) No violation in locating vias between M_1 and M_2 . (c) Lack of routing tracks.**

which x coordinates are between 7 and 9 on M_2 layer are either already occupied by other connections or blocked by design rules (e.g., VR and EOL rule) after several propagations as shown in Figure 17(c). In this case, there are no ways to connect pin 1 to pin 4 since the connection between two pins needs at least one track in any x coordinates.

3.2 Experimental Results

Table 3 presents our diagnosis experimental statistics with comparisons for each conflict type. Each row in Table 3 reports the average results. The result shows that ROAD can give the detailed diagnosis results with an efficient runtime cost. For example, the total diagnosis time, including the MUS Extraction and the iterative diagnosis procedure (Decision & Propagation), is finished with 21.1 seconds on average for CP pattern cases. Even the routing congestion cases, the run-time for the MUS Extraction and the Decision & Propagation is 72.9 and 33.4 seconds on average, respectively. Within 2 minutes (strictly, 106.3 seconds), the routability analysis and diagnosis are completed with exact conflict reasoning.

As mentioned in Section 2.3, this diagnosis efficiency is achieved by adopting MUS clauses for the diagnosis target. As shown in Table 3, the amount of MUS for the CP pattern and routing congestion is 0.010 and 0.086% of original representation, respectively.

Table 3: Experiment statistics of routability diagnosis (94 pin layouts @ 90 grids). In the table, #N=#Nets, #P=#Pins.

Conflict Type	#N (avg.)	#P (avg.)	#Variable (avg.)	#Clauses (avg.)			Diagnosis Time [s] (avg.)		
				Original	MUS	Ratio [%]	MUS Extraction	Decision & Propagation	Total
CP pattern (80 cases)	10.2	22.4	68,954.6	3,068,017.3	291.5	0.010	17.0	4.1	21.1
Routing Congestion (14 cases)	10.0	22.5	68,803.3	3,049,687.6	2,627.8	0.086	72.9	33.4	106.3

Figure 18: Unroutable cause classification of layout with 90 grids, 80% density and (a) one row, (b) two rows, (c) three rows of pins.

From the statistics, the number of MUS clauses and the diagnosis time for routing congestion type is about nine and five times those of CP pattern type, respectively. It demonstrates that routing congestion type requires more iterations of Decision & Propagation to verify the conflicting clauses even though the specification of the layout is similar between two types as mentioned in Section 3.1.

The composition of unroutable causes fully vary depending on the row number of pins (e.g., the number of routing tracks). Figure 18 summarizes the findings for 90 grids. As shown in Figure 18(a), 69% of pin layouts with one row is classified into the Routing Congestion type, while there is no Propagated-CP type. The Routing Congestion type's portion is significantly reduced by 8% for cases with two pin rows as shown in Figure 18(b). Unlike the Routing Congestion, the portion of the Propagated-CP and Simple-CP type increase to 80% and 12%, respectively. Figure 18(c) shows that, for the pin layouts with three rows, no routing congestion type occurs and all cases are CP pattern type (80% simple-CP, 20% propagated-CP).

4 CONCLUSION

We have described our ROAD, a new analysis and diagnosis framework that examines routing feasibility for the detailed pin layout architecture. Our diagnosis procedure performs iterative procedures of decision and propagation to identify the longest path of the conflicted pin layouts. It provides useful insight into the causes of the conflict design rule, unlike existing ILP-based routability analysis methodologies. Throughout a comprehensive diagnosis, we also identify three categories of conflict causes, Simple-CP, Propagated-CP pattern and Routing Congestion. We demonstrate the diagnosis results with 6,000 pin layouts, and 165 selected unroutable pin layouts. Our ROAD suggests strong theoretical background towards diverse pin pattern exploration for unroutability. Our future plan includes applying machine learning techniques in the detailed routing.

REFERENCES

- [1] 2015. ITRS Report 2015. <http://www.itrs2.net/itrs-reports.html>
- [2] 2018. Design Rule Complexity Rising. <https://semiengineering.com/design-rule-complexity-rising/>.
- [3] Charles J Alpert and Gustavo E Tellez. 2010. The importance of routing congestion analysis. *DAC Knowledge Center Online Article* (2010).
- [4] Yasmine A Badr, Ko-wei Ma, and Puneet Gupta. 2014. Layout pattern-driven design rule evaluation. *Journal of Micro/Nanolithography, MEMS, and MOEMS* 13, 4 (2014), 043018.
- [5] Oliver Bastert and Christian Matuszewski. 2001. Layered drawings of digraphs. In *Drawing graphs*. Springer, 87–120.
- [6] Anton Belov and EPCL Basic Training Camp. 2012. Minimal Unsatisfiability: Theory, Algorithms and Applications. (2012).
- [7] Anton Belov and Joao Marques-Silva. 2012. MUSer2: An efficient MUS extractor. *Journal on Satisfiability, Boolean Modeling and Computation* 8 (2012), 123–128.
- [8] RC Carden, Jianmin Li, and Chung-Kuan Cheng. 1996. A global router with a theoretical bound on the optimal solution. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 15, 2 (1996), 208–216.
- [9] Jian Chen, Jun Wang, ChengYu Zhu, Wei Xu, Shuai Li, Eason Lin, Odie Ou, Ya-Chieh Lai, and Shengrui Qu. 2018. Pin routability and pin access analysis on standard cells for layout optimization. In *Design-Process-Technology Co-optimization for Manufacturability XII*, Vol. 10588. International Society for Optics and Photonics, 105880D.
- [10] Pascal Cremer, Stefan Hougardy, Jan Schneider, and Jannik Silvanus. 2017. Automatic Cell Layout in the 7nm Era. In *International Symposium on Physical Design*. ACM, 99–106.
- [11] Yixiao Ding, Chris Chu, and Wai-Kei Mak. 2017. Pin Accessibility-Driven Detailed Placement Refinement. In *International Symposium on Physical Design*. ACM, 133–140.
- [12] Kwangsoo Han, Andrew B Kahng, and Hyein Lee. 2015. Evaluation of BEOL design rule impacts using an optimal ILP-based detailed router. In *the 52nd Annual Design Automation Conference*. ACM, 68.
- [13] T.C. Hu and M.T. Shing. 2002. *Combinatorial Algorithms 2nd Ed*. Dover Publications.
- [14] Xiaotao Jia, Yici Cai, Qiang Zhou, Gang Chen, Zhuoyuan Li, and Zuwei Li. 2014. MCFRoute: a detailed router based on multi-commodity flow method. In *IEEE/ACM International Conference on Computer-Aided Design*. IEEE Press, 397–404.
- [15] Andrew B Kahng. 2018. Machine Learning Applications in Physical Design: Recent Results and Directions. In *International Symposium on Physical Design*. ACM, 68–73.
- [16] Ilgweon Kang, Dongwon Park, Changho Han, and Chung-Kuan Cheng. 2018. Fast and precise routability analysis with conditional design rules. In *Proceedings of the 20th System Level Interconnect Prediction Workshop*. ACM, 4.
- [17] Daniel Kroening and Ofer Strichman. 2016. *Decision Procedures: An Algorithmic Point of View*. Springer.
- [18] Kevin Lucas, Chris Cork, Bei Yu, Gerard Luk-Pat, Ben Painter, and David Z Pan. 2012. Implications of triple patterning for 14nm node design and patterning. In *Design for Manufacturability through Design-Process Integration VI*, Vol. 8327. International Society for Optics and Photonics, 832703.
- [19] Yuanqiang Ma, Jason Sweis, Hidekazu Yoshida, Yan Wang, Jongwook Kye, and Harry J Levinson. 2012. Self-aligned double patterning (SADP) compliant design flow. In *Design for Manufacturability through Design-Process Integration VI*, Vol. 8327. International Society for Optics and Photonics, 832706.
- [20] G-J Nam, Fadi Aloul, Kareem A Sakallah, and Rob A Rutenbar. 2004. A comparative study of two Boolean formulations of FPGA detailed routing constraints. *IEEE Trans. Comput.* 53, 6 (2004), 688–696.
- [21] Van-Hau Nguyen. 2014. SAT Encodings of Finite CSPs. (2014).
- [22] Nikolai Ryzhenko and Steven Burns. 2012. Standard cell routing via boolean satisfiability. In *the 49th Annual Design Automation Conference*. ACM, 603–612.
- [23] Jaewoo Seo, Jinwook Jung, Sangmin Kim, and Youngsoo Shin. 2017. Pin accessibility-driven cell layout redesign and placement optimization. In *the 54th Annual Design Automation Conference 2017*. ACM, 54.
- [24] Miroslav N Velev. 2007. Exploiting hierarchy and structure to efficiently solve graph coloring as SAT. In *Proceedings of the 2007 IEEE/ACM international conference on Computer-aided design*. IEEE Press, 135–142.
- [25] Xiaoqing Xu, Bei Yu, Jih-Rong Gao, Che-Lun Hsu, and David Z Pan. 2016. PARR: Pin-access planning and regular routing for self-aligned double patterning. *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 21, 3 (2016), 42.
- [26] Payman Zarkesh-Ha, Jeffrey A Davis, William Loh, and James D Meindl. 2000. Prediction of interconnect fan-out distribution using Rent's rule. In *System-level interconnect prediction*. ACM, 107–112.
- [27] Jianmin Zhang, Tiejun Li, and Sikun Li. 2015. Application and analysis of unsatisfiable cores on circuits synthesis. In *Advanced Computational Intelligence (ICACI), 2015 Seventh International Conference on*. IEEE, 407–410.