An Empirical Measure for Characterizing 3-SAT

Hachemi Bennaceur¹ and Chu Min Li²

Abstract. The literal encoding of the satisfiability problem (SAT) as a binary constraint satisfaction problem (CSP) allows us to make a technical comparison between the propositional logic and the constraint network fields. Equivalencies or analogies between concepts and techniques used in the two fields are then shown. For instance it is shown that arc consistency is equivalent to unit resolution [7, 2]. This result is generalized here by relating the local consistency concepts with inference rules in the propositional logic. We relate the pure literal rule with an interesting property of constraint networks called “row convexity”, and we propose an empirical characterization of satisfiable and unsatisfiable 3-SAT instances.

1 Introduction

This work is based on the literal encoding of the satisfiability problem (SAT) as a binary constraint satisfaction problem (CSP) [1]. The literal encoding of the satisfiability problem (SAT) as a binary constraint satisfaction problem (CSP) allows us to make a technical comparison between the propositional logic and the constraint network fields [7]. Equivalencies or analogies between concepts and techniques used in the two fields are then shown. For instance it is shown that arc consistency is equivalent to unit resolution [7, 2]. We generalize this result by relating the path consistency concept with unit and binary resolution rules (binary resolution will be defined later), and we relate the pure literal rule with an interesting property of constraint networks called “row convexity”. The constraint networks verifying this property can be solved without backtracking [6]. From the link between the pure literal rule and the row convexity property we propose an empirical characterization of satisfiable and unsatisfiable 3-SAT instances.

The organization of this paper is as follows. Section 2 reviews basic definitions on a satisfiability problem and the constraint satisfaction frameworks. Section 3 presents the literal encoding and the connection between the path consistency concept and inference rules. Sections 4 recalls the row convexity property, and presents its relation with the pure literal rule and the measure which we propose for 3-SAT instances, and section 5 deals with the computational experiments. Section 6 concludes.

2 Background

A constraint network or a Constraint Satisfaction Problem (CSP) involves the assignment of values to variables which are subject to a set of constraints. Formally, a binary CSP is defined by a quadruplet (X,D,C,R) where:

- X is a set of n variables \( \{X_1,X_2,\ldots,X_n\} \);
- D is a set of n domains \( \{D_1,D_2,\ldots,D_n\} \) where each \( D_i \) is a set of \( d_i \) possible values for \( X_i \);
- C is a set of m constraints where each constraint \( C_{ij} \) between the variables \( X_i \) and \( X_j \), \( i \neq j \), is defined by its relation \( R_{ij} \);
- R is a set of m relations, where \( R_{ij} \) is a subset of the Cartesian product \( D_i \times D_j \).

A solution of the CSP is a total assignment satisfying each of the constraints.

A binary relation between variables \( X_i \) and \( X_j \) may be represented as a (0-1)-matrix with \( |D_i| \) rows and \( |D_j| \) columns by imposing an ordering on the domains of the variables. A zero at the intersection of row \( r \) and column \( c \) means that the pair consisting of the \( r^{th} \) element of \( D_i \) and the \( c^{th} \) element of \( D_j \) is not allowed; an one at this intersection means that the pair is allowed.

Definition 2.1 [6] A (0-1)-matrix is row convex if in each row there is no two 1 separated by 0. A relation is row convex if its associated matrix is row convex.

We recall now the definitions of arc consistency and path consistency.

Definition 2.2 A domain \( D_i \) of \( D \) is arc consistent (2-consistent) iff, for each \( v_r \in D_i \), and for each \( X_j \in X \) such that \( C_{ij} \in C \), there exists \( v_s \in D_j \) such that \( R_{ij}(v_r,v_s) \) holds. A CSP is arc consistent iff for each \( D_i \in D \), \( D_i \) is arc consistent and \( D_i \) is not empty.

Definition 2.3 A pair of variables \( X_i,X_j \) is path consistent (3-consistent) iff for each pair \( (v_r,v_s) \) such that \( R_{ij}(v_r,v_s) \) holds, and for each \( X_k \in X \) such that \( C_{ik} \in C \) and \( C_{kj} \in C \), there exists \( v_t \in D_k \) such that \( R_{ik}(v_r,v_t) \) and \( R_{kj}(v_s,v_t) \) hold. A CSP is path consistent iff each pair \( X_i,X_j \) is path consistent.

A SAT instance is a conjunction of distinct clauses. A clause is a disjunction of literals. A literal is a propositional variable \( x \) or its negation \( \overline{x} \). If \( I \) denotes a positive or a negative literal, \( \overline{I} \) is its complement. A SAT instance is satisfiable (has a model), if there exists an assignment of its literals to true or false which makes all its clauses true. If every clause of the SAT instance exactly has \( k \) literals, the SAT problem is called \( k \)-SAT. If exactly one literal \( I \) occurs negated in one clause and unnegated in another, the resolution rule consists in generating a resolvent on \( I \), the clause containing all literals occurring in either parent, except the literals \( I \) and \( \overline{I} \). Obviously, a SAT instance can be seen as a particular CSP with the domain of each variable is the Boolean set \( \{True, False\} \).
3 Local consistencies and inference rules

We recall the translation of a SAT instance to a binary CSP and we express the path consistency concept in the propositional logic framework. Namely, we show that achieving path consistency is equivalent to performing unit and binary resolution rules.

3.1 From SAT to binary CSP

The SAT problem may be translated into a binary CSP as follows: let \( S \) be a SAT problem with \( n \) clauses and \( n \) literals, we associate to each clause \( c_i \) of \( S \) a variable \( X_i \), the domain \( D_c \) of \( X_i \) is the set of literals of \( c_i \).

For instance, if \( c_1 \) is the clause \( \bar{I}_1 \lor \bar{I}_2 \lor \bar{I}_3 \lor \bar{I}_4 \), then the domain \( D_{c_1} \) of \( X_1 \) is \( \{ \bar{I_1}, \bar{I}_2, \bar{I}_3, \bar{I}_4 \} \).

A constraint between two variables \( X_i, X_j \) associated to two clauses \( c_i \) and \( c_j \) is created if the clause \( c_i \) contains a literal \( l \) and the clause \( c_j \) contains its complement \( \bar{l} \).

For instance, if \( c_1 \) and \( c_2 \) are respectively the clauses \( \bar{I}_1 \lor \bar{I}_2 \lor \bar{I}_3 \) and \( \bar{I}_1 \lor \bar{I}_3 \lor I_4 \), a constraint is created between \( X_1 \) and \( X_2 \) since \( c_1 \) contains \( \bar{l}_1 \) and \( c_2 \) contains \( \bar{l}_1 \).

For each constraint between two variables \( X_i, X_j \), a relation \( R_{ij} \) is defined by the Cartesian product \( D_i \times D_j \) minus the tuples \( (l, \bar{l}) \). For instance, below is the relation \( R_{ij} \) of the above constraint.

\[
R_{ij} = \{(l_1, l_3), (l_1, l_4), (\bar{l}_2, \bar{l}_1), (\bar{l}_2, l_3), (\bar{l}_2, l_4), (\bar{l}_3, l_3), (\bar{l}_3, \bar{l}_4)\}
\]

We sometimes also explicitly associate a relation \( R_{ij} \) defined by the Cartesian product \( D_i \times D_j \) to two clauses \( c_i \) and \( c_j \) not sharing any complementary literal.

In the rest of the paper, \( S \) denotes a SAT problem and \( P(S) \) its associated CSP. \( S \) is satisfiable if and only if \( P(S) \) has a solution [1].

Property 3.1 [1] If each clause of \( S \) contains at least two literals (resp. three literals), then \( P(S) \) is arc consistent (resp. path-consistent).

3.2 Path consistency and Binary Resolution

Definition 3.1 We call Unit (resp. Binary) Resolution the resolution rule applied to two clauses of which at least one is unary (resp. binary).

Let \( U(R)(S) \) denotes the Unit Resolution closure of \( S \). It is shown in [7] that:

Theorem 1 [7] Achieving arc consistency on \( P(S) \) is equivalent to computing \( U(R)(S) \).

The following result, generalizing theorem 1, relates path consistency to unit and binary resolution.

Let \( B(R)(S) \) denote the Binary and Unit Resolution closure of \( S \).

Theorem 2 Achieving path consistency on \( P(S) \) is equivalent to computing \( B(R)(S) \).

Proof

We show by induction that a pair \( (l_1, l_2) \) is path-inconsistent iff \( \bar{l}_1 \lor \bar{l}_2 \) or \( \bar{l}_1 \lor l_2 \) is a clause of \( S \), or one of these clauses is produced by the Binary or Unit Resolution during the constraint propagation.

Achieving path consistency on a CSP can be done in two phases: the checking phase and the constraint propagation phase. A pair of values can be detected inconsistent in one of these phases.

(i) A pair \( (l_1, l_2) \) is detected inconsistent during the checking phase iff \( \bar{l}_1 \lor \bar{l}_2 \lor l_4 \) or \( \bar{l}_1 \lor \bar{l}_2 \lor l_5 \) is a clause of \( S \), since \( (l_1, l_2) \) has no support in these latter clauses.

(ii) A pair \( (l_1, l_2) \) of \( R_{ij} \) is detected inconsistent during the constraint propagation phase iff there exists a variable \( X_k \) of \( P(S) \) such that \( (l_1, l_2) \) has no support in \( D_k \).

Assume that initially \( D_k \) is \( \{l_5, l_4, l_3\} \) (note that before the filtering process \( D_k \) may contain more than three elements, in this case we develop the same reasoning). Then, the clause \( c_k \) associated to \( D_k \) is

\[
l_5 \lor l_4 \lor l_5
\]

Since the pair \( (l_1, l_2) \) of \( R_{ij} \) is path-inconsistent, then one pair in \( \{(l_1, l_3), (l_2, l_3)\} \), one pair in \( \{(l_1, l_4), (l_2, l_4)\} \), and one pair in \( \{(l_1, l_5), (l_2, l_5)\} \) are already detected path-inconsistent.

By induction, one of clauses 2 and 3

\[
\bar{l}_1 \lor \bar{l}_3
\]

and one of clauses 4 and 5

\[
\bar{l}_2 \lor \bar{l}_3
\]

and one of clauses 6 and 7

\[
\bar{l}_1 \lor \bar{l}_5
\]

\[
\bar{l}_2 \lor \bar{l}_5
\]

are generated by unit or binary resolutions during the constraint propagation phase. Assume that clauses 3, 4 and 7 are the clauses generated. The following binary resolutions

\[
\text{clause (1) \land clause (3)} \implies \bar{l}_2 \lor \bar{l}_4 \lor l_5
\]

\[
\text{clause (1) \land clause (4)} \implies \bar{l}_1 \lor \bar{l}_2 \lor l_5
\]

\[
\text{clause (3) \land clause (7)} \implies \bar{l}_1 \lor l_2
\]

lead to the binary clause \( \bar{l}_1 \lor \bar{l}_2 \). All other possibilities lead to one of the clauses \( \bar{l}_1 \lor \bar{l}_2, \bar{l}_1, \bar{l}_2 \).

The above theorem allows us to achieve path consistency on a SAT problem without effectively transforming the SAT problem to a CSP.

The computing of the \( B(R)(S) \) leads to a path consistent SAT problem (see example 4.1).

4 Characterization of satisfiable and unsatisfiable 3-SAT instances

We first relate the row convexity property of CSPs with the pure literal rule and then we present a measure characterizing satisfiable and unsatisfiable 3-SAT instances.
4.1 Row convexity

Van Beek and Dechter have identified a tractable class of CSPs for which a solution can be found using a path consistency algorithm [6]. The result is based on the row convexity property.

**Definition 4.1** [6] Given an ordering of the variables $X_1, ..., X_n$, a binary CSP is directionally row convex if each of the (0,1)-matrices $R_{ij}$, where variable $X_i$ occurs before variable $X_j$ in the ordering, is row convex.

**Theorem 3** [6] Let $P$ be a path consistent binary CSP. If there exists an ordering of the variables $X_1, ..., X_n$ and of the domains $D_1, ..., D_n$ of $P$ such that $P$ is directionally row convex, then a solution can be found without backtracking.

**Corollary 4.1** The satisfiability problems containing only binary clauses (2-SAT) can be solved using a path consistency algorithm.

**Proof**

Let $S$ be a 2-SAT problem. For any ordering of its clauses, $P(S)$ is row convex, and achieving a path consistency on $P(S)$ leads to a row convex and path consistent CSP for which a solution can be found without backtracking, or to an empty relation of $P(S)$ and then we conclude that $P(S)$ has no solution.

**Example 4.1** Let $S$ be the 2-SAT problem defined by the following set of clauses

$$
\bar{t}_1 \lor \bar{t}_2 \\
\bar{t}_3 \lor t_2 \\
\bar{t}_3 \lor t_1 \\
\bar{t}_4 \lor \bar{t}_3 \\
\bar{t}_4 \lor \bar{t}_4
$$

(10)  (11)  (12)  (13)  (14)  (15)

We solve this 2-SAT instance in two corresponding ways by achieving path consistency on $P(S)$ or by computing BUR($S$) (See theorem 2).

The corresponding CSP $P(S)$ contains 6 variables $X_1, X_2, X_3, X_4, X_5, X_6$. The domains of these variables are $D_1 = \{\bar{t}_1, t_2\}$, $D_2 = \{\bar{t}_3, t_1\}$, $D_3 = \{\bar{t}_4, \bar{t}_4\}$, $D_4 = \{t_2, \bar{t}_1\}$, $D_5 = \{\bar{t}_3, \bar{t}_4\}$, $D_6 = \{t_4, \bar{t}_3\}$. The relation between variables $X_2$ and $X_5$ is $R_{25} = \{\bar{t}_3 \lor \bar{t}_4, \bar{t}_3 \lor t_2 \lor \bar{t}_4, t_2 \lor \bar{t}_3 \lor \bar{t}_4\}$

The checking phase on $P(S)$:

(1) $(\bar{t}_3, \bar{t}_4)$ of $R_{25}$ has no support in $D_4$;
(2) $(\bar{t}_3, \bar{t}_4)$ of $R_{25}$ has no support in $D_5$.

Then, the pairs $(\bar{t}_3, \bar{t}_4)$ and $(\bar{t}_3, \bar{t}_4)$ can be removed from $R_{25}$.

Thus, the value $\bar{t}_3$ can be removed from $D_5$.

(18) $D_2 = \{t_2\}$ and $D_3 = \{t_1\}$.

**4.2 Row convexity and the pure literal rule**

The following result relates the pure convexity property with the pure literal rule using the literal encoding of a 3-SAT instance.

**Lemma 4.1** Let $S$ be a 3-SAT instance and $P(S)$ its associated CSP. If $P(S)$ is directionally row convex then there exists in $S$ a pure literal.

**Proof**

Assume that $S$ does not contain any pure literal. Let $c_1$ be a clause, for any literal ordering of $c_1$, there exists a clause $c_2$ containing the complementary literal of the literal placed in the middle of $c_1$. Thus, the matrix encoding the relation between the clauses $c_2$ and $c_1$ is not row convex, hence, $P(S)$ is not directionally row convex.

**Theorem 4** Let $m$ be the number of clauses of $S$. $P(S)$ is directionally row convex if there exists a partition $S_1, ..., S_k : (1 \leq k \leq m)$ of the set of clauses $S$ such that $S$ contains a pure literal which occurs only in the clauses of $S_i$, and each set of clauses $S_i : (1 \leq i \leq k)$ contains a pure literal.

**Proof**

⇒ By construction, we determine an ordering of the clauses of $S$ such that $P(S)$ will be directionally row convex as follows: all the clauses of $S_i$ must be placed in the clause of the ordering and the pure literal of $S$ must be placed at the middle of each clause of $S_i$. We repeat the same ordering for $S_i, S_{i+1}, ..., S_k$; the clauses of $S_i$ will be placed before the clauses of $S_i$ in the ordering and the pure literal of $S - S_i$ will be placed in the middle of the clauses of $S_i$, and so on for others sets of clauses $S_k, ..., S_1$. Thus, the ordering clauses $S_k, S_{k-1}, ..., S_1$ leads to a directionally row convex CSP $P(S)$.  

---

203
4.3 Measure characterizing satisfiable and unsatisfiable 3-SAT instances

When the CSP is not row convex, we define the minimal row non-convexity $NC$ of a 3-SAT instance to be the smallest number of row non-convex (0,1)-matrices representing the constraints for any variable and value ordering. We use a simple heuristic shown in figure 1 to estimate $NC$ for a 3-SAT instance. This heuristic is based on the proof of Theorem 4.

Procedure MinimalRowNonConvexity($S$)
Begin
$NC:=0$;
repeat
  let $neg(x)$ and $pos(x)$ denote the number of positive and negative occurrences of variable $x$ in the current $S$;
  select $x$ such that $neg(x) \ast pos(x)$ is the smallest;
  break tie by choosing $x$ such that $neg(x) + pos(x)$ is the largest;
  remove all clauses containing $x$ or $\overline{x}$ from $S$, for every removed clause $c$, record its removing time $f(c)$;
  Order the literals of the removed clauses such that $x$ or $\overline{x}$ is in the middle position;
  $NC := NC + neg(x) \ast pos(x)$;
until $S$ becomes empty;
put all removed clauses in order of $f(c)$;
return $NC$;
End
Figure 1.

The MinimalRowNonConvexity procedure essentially computes an ordering of clauses in the initial $S$ and of literals in each clause. If $P(S)$ is directionally row convex, the procedure returns 0 and is similar to the FindOrder procedure in [6], otherwise it computes the number of row non-convex (0,1)-matrices under the ordering. To see this, it is sufficient to note that only the (0,1)-matrices between clauses removed at the same stage of the algorithm can have zeros separating two ones, because of the complementary literal in the middle position (if any).

Intuitively, since the MinimalRowNonConvexity procedure first treats all clauses containing $x$ or $\overline{x}$ such that $neg(x) \ast pos(x)$ is the smallest, $NC$ increases slowly. The removing of these clauses reduces the number of occurrences of remaining variables, eventually introducing some pure literals. For these reasons, we believe that the MinimalRowNonConvexity procedure computes an ordering of clauses and literals such that the number $NC$ of row non-convex constraints is small. The counting of the number of occurrences of variables can be done in $O(n + m)$ time, where $n$ and $m$ are respectively the number of clauses and variables in $S$. So the complexity of the MinimalRowNonConvexity procedure is $O(n(n + m))$.

5 Experimentation

We use the MinimalRowNonConvexity procedure to estimate $NC$ for a large sample of random 3-SAT instances. We vary $n$ from 300 to 450 incrementing by 50 and $m$ so that $m/n = 4, 4.1, 4.2, 4.25, 4.3, 4.4, 4.5$ respectively. Empirically, when the ratio $m/n$ is near 4.25, $S$ is unsatisfiable with the probability 0.5 and is the most difficult to solve [5, 3].

Table 1 shows the number of instances included in our study at each point $(m/n, n)$.

Table 1. The number of instances studied at each point $(m/n, n)$

<table>
<thead>
<tr>
<th></th>
<th>300</th>
<th>350</th>
<th>400</th>
<th>450</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>5000</td>
<td>3000</td>
<td>3000</td>
<td>3000</td>
</tr>
<tr>
<td>4.1</td>
<td>1000</td>
<td>1000</td>
<td>600</td>
<td>400</td>
</tr>
<tr>
<td>4.2</td>
<td>1000</td>
<td>1000</td>
<td>600</td>
<td>400</td>
</tr>
<tr>
<td>4.25</td>
<td>1000</td>
<td>1000</td>
<td>600</td>
<td>400</td>
</tr>
<tr>
<td>4.3</td>
<td>1000</td>
<td>1000</td>
<td>600</td>
<td>400</td>
</tr>
<tr>
<td>4.4</td>
<td>1000</td>
<td>1000</td>
<td>600</td>
<td>400</td>
</tr>
</tbody>
</table>

In order to compare the minimal row non-convexity $NC$ of satisfiable and unsatisfiable instances, we solve these instances using Satz [4] and divide the instances at each point into satisfiable class and unsatisfiable class. Tables 2 and 3 show the average value of $NC$ (in boldface) for each class and the number of instances in the class (between parentheses).

Table 2. Average estimated minimal row non-convexity $NC$ of satisfiable instances at each point $(m/n, n)$

<table>
<thead>
<tr>
<th></th>
<th>300</th>
<th>350</th>
<th>400</th>
<th>450</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1005 (2983)</td>
<td>1169 (2998)</td>
<td>1335 (2999)</td>
<td>1580 (2999)</td>
</tr>
<tr>
<td>4.1</td>
<td>1074 (944)</td>
<td>1251 (972)</td>
<td>1425 (582)</td>
<td>1606 (392)</td>
</tr>
<tr>
<td>4.2</td>
<td>1140 (715)</td>
<td>1327 (746)</td>
<td>1515 (452)</td>
<td>1706 (307)</td>
</tr>
<tr>
<td>4.25</td>
<td>1172 (551)</td>
<td>1367 (547)</td>
<td>1561 (319)</td>
<td>1754 (224)</td>
</tr>
<tr>
<td>4.3</td>
<td>1208 (370)</td>
<td>1406 (353)</td>
<td>1604 (197)</td>
<td>1802 (128)</td>
</tr>
<tr>
<td>4.4</td>
<td>1272 (128)</td>
<td>1486 (85)</td>
<td>1686 (46)</td>
<td>1905 (27)</td>
</tr>
<tr>
<td>4.5</td>
<td>1353 (32)</td>
<td>1564 (11)</td>
<td>1770 (5)</td>
<td>2002 (4)</td>
</tr>
</tbody>
</table>

Table 3. Average estimated minimal row non-convexity $NC$ of unsatisfiable instances at each point $(m/n, n)$

<table>
<thead>
<tr>
<th></th>
<th>300</th>
<th>350</th>
<th>400</th>
<th>450</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1040 (17)</td>
<td>1244 (22)</td>
<td>1375 (11)</td>
<td>1596 (1)</td>
</tr>
<tr>
<td>4.1</td>
<td>1108 (56)</td>
<td>1274 (28)</td>
<td>1462 (18)</td>
<td>1640 (8)</td>
</tr>
<tr>
<td>4.2</td>
<td>1163 (285)</td>
<td>1351 (254)</td>
<td>1544 (148)</td>
<td>1726 (93)</td>
</tr>
<tr>
<td>4.25</td>
<td>1197 (449)</td>
<td>1392 (453)</td>
<td>1585 (281)</td>
<td>1778 (176)</td>
</tr>
<tr>
<td>4.3</td>
<td>1229 (630)</td>
<td>1429 (647)</td>
<td>1630 (403)</td>
<td>1835 (272)</td>
</tr>
<tr>
<td>4.4</td>
<td>1302 (872)</td>
<td>1512 (915)</td>
<td>1728 (554)</td>
<td>1942 (373)</td>
</tr>
<tr>
<td>4.5</td>
<td>1378 (968)</td>
<td>1603 (989)</td>
<td>1830 (595)</td>
<td>2059 (396)</td>
</tr>
</tbody>
</table>

In the sequel, we directly denote by $NC$ the average estimated minimal row non-convexity of a class of instances. We have empirically observed in tables 2 and 3 that the ratio $NC/n$ tends to be a constant for a given value $m/n$ and a class, except for the points where few instances are solved (e.g. unsatisfiable instances for $m/n = 4$, satisfiable instances for $m/n = 4.5$). This is particular true for satisfiable instances when $m/n \leq 4.25$ and for unsatisfiable instances when $m/n > 4.25$. The difference of $NC/n$
between satisfiable and unsatisfiable classes tends to become larger when \( m/n < 4.2 \) and \( m/n > 4.3 \), which can be used to explain the threshold phenomenon. Tables 4 and 5 show the ratio \( NC/n \) in boldface (and the number of instances averaged between parentheses) at each point \( (m/n, n) \) for satisfiable and unsatisfiable instances.

### Table 4. Ratio \( NC/n \) for satisfiable instances at each point \( (m/n, n) \)

<table>
<thead>
<tr>
<th>4</th>
<th>4.35 (2558)</th>
<th>3.34 (1298)</th>
<th>3.54 (3399)</th>
<th>3.34 (2999)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>3.58 (944)</td>
<td>3.57 (972)</td>
<td>3.56 (582)</td>
<td>3.57 (392)</td>
</tr>
<tr>
<td>4.2</td>
<td>3.80 (715)</td>
<td>3.79 (746)</td>
<td>3.79 (452)</td>
<td>3.79 (307)</td>
</tr>
<tr>
<td>4.25</td>
<td>3.90 (551)</td>
<td>3.91 (547)</td>
<td>3.90 (319)</td>
<td>3.90 (224)</td>
</tr>
<tr>
<td>4.3</td>
<td>4.02 (370)</td>
<td>4.02 (353)</td>
<td>4.01 (197)</td>
<td>4.00 (128)</td>
</tr>
<tr>
<td>4.4</td>
<td>4.24 (128)</td>
<td>4.25 (85)</td>
<td>4.22 (46)</td>
<td>4.23 (27)</td>
</tr>
<tr>
<td>4.5</td>
<td>4.51 (32)</td>
<td>4.49 (11)</td>
<td>4.43 (5)</td>
<td>4.45 (4)</td>
</tr>
</tbody>
</table>

To verify the phenomenon, we solve a large sample of instances at other points from \( n = 220 \) to \( n = 380 \) increasing by step 20 (excluding \( n = 300 \)). Table 6 and 7 show the ratio \( NC/n \) at these points. Note that there are always few unsatisfiable instances when \( m/n = 4 \) and few satisfiable instances when \( m/n = 4.5 \).

### Table 5. Ratio \( NC/n \) for unsatisfiable instances at each point \( (m/n, n) \)

<table>
<thead>
<tr>
<th>4</th>
<th>3.47 (117)</th>
<th>3.58 (2)</th>
<th>3.43 (1)</th>
<th>3.55 (1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>3.69 (56)</td>
<td>3.64 (28)</td>
<td>3.65 (18)</td>
<td>3.64 (8)</td>
</tr>
<tr>
<td>4.2</td>
<td>3.88 (285)</td>
<td>3.86 (254)</td>
<td>3.86 (148)</td>
<td>3.84 (93)</td>
</tr>
<tr>
<td>4.25</td>
<td>3.99 (449)</td>
<td>3.98 (453)</td>
<td>3.96 (281)</td>
<td>3.95 (176)</td>
</tr>
<tr>
<td>4.3</td>
<td>4.10 (630)</td>
<td>4.08 (647)</td>
<td>4.08 (403)</td>
<td>4.08 (272)</td>
</tr>
<tr>
<td>4.4</td>
<td>4.34 (872)</td>
<td>4.32 (915)</td>
<td>4.32 (554)</td>
<td>4.32 (373)</td>
</tr>
<tr>
<td>4.5</td>
<td>4.59 (968)</td>
<td>4.58 (989)</td>
<td>4.58 (595)</td>
<td>4.58 (396)</td>
</tr>
</tbody>
</table>

It appears that the ratio \( NC/n \) could be used to estimate the satisfiability of a given random 3-SAT instance. For example, we know that an instance near the threshold \( (m/n = 4.25) \) is satisfiable with an empirical probability 0.5. However if \( NC/n < 3.90 \), it should be satisfiable with a more important probability. One might also think that a reason for the unsatisfiability of an instance is the large number of its row non-convex (0-1)-matrices.

### Table 6. Ratio \( NC/n \) for satisfiable instances at each point \( (m/n, n) \)

<table>
<thead>
<tr>
<th>4</th>
<th>3.36</th>
<th>3.36</th>
<th>3.35</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
<th>3.34</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2</td>
<td>3.81</td>
<td>3.81</td>
<td>3.80</td>
<td>3.80</td>
<td>3.80</td>
<td>3.79</td>
<td>3.79</td>
<td>3.79</td>
<td>3.79</td>
<td>3.79</td>
<td>3.79</td>
<td>3.79</td>
<td>3.79</td>
<td>3.79</td>
<td>3.79</td>
<td>3.79</td>
</tr>
<tr>
<td>4.3</td>
<td>4.03</td>
<td>4.02</td>
<td>4.02</td>
<td>4.02</td>
<td>4.02</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
<td>4.00</td>
</tr>
<tr>
<td>4.5</td>
<td>4.51</td>
<td>4.47</td>
<td>4.43</td>
<td>4.43</td>
<td>4.43</td>
<td>4.44</td>
<td>4.44</td>
<td>4.44</td>
<td>4.44</td>
<td>4.44</td>
<td>4.44</td>
<td>4.44</td>
<td>4.44</td>
<td>4.44</td>
<td>4.44</td>
<td>4.44</td>
</tr>
</tbody>
</table>

### Table 7. Ratio \( NC/n \) for unsatisfiable instances at each point \( (m/n, n) \). At points (400, 4) and (450, 4), none of the 1000 solved instances is unsatisfiable.

<table>
<thead>
<tr>
<th>4</th>
<th>3.47</th>
<th>3.47</th>
<th>3.51</th>
<th>3.46</th>
<th>3.45</th>
<th>3.53</th>
<th>-</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>3.70</td>
<td>3.69</td>
<td>3.69</td>
<td>3.65</td>
<td>3.66</td>
<td>3.66</td>
<td>3.69</td>
<td>3.61</td>
</tr>
<tr>
<td>4.2</td>
<td>3.89</td>
<td>3.89</td>
<td>3.89</td>
<td>3.88</td>
<td>3.86</td>
<td>3.86</td>
<td>3.86</td>
<td>3.85</td>
</tr>
<tr>
<td>4.25</td>
<td>4.00</td>
<td>4.00</td>
<td>3.99</td>
<td>3.99</td>
<td>3.99</td>
<td>3.97</td>
<td>3.97</td>
<td>3.96</td>
</tr>
<tr>
<td>4.3</td>
<td>4.12</td>
<td>4.11</td>
<td>4.10</td>
<td>4.09</td>
<td>4.09</td>
<td>4.07</td>
<td>4.08</td>
<td>4.07</td>
</tr>
<tr>
<td>4.4</td>
<td>4.35</td>
<td>4.35</td>
<td>4.34</td>
<td>4.34</td>
<td>4.33</td>
<td>4.31</td>
<td>4.32</td>
<td>4.31</td>
</tr>
<tr>
<td>4.5</td>
<td>4.61</td>
<td>4.60</td>
<td>4.60</td>
<td>4.59</td>
<td>4.59</td>
<td>4.57</td>
<td>4.57</td>
<td>4.57</td>
</tr>
</tbody>
</table>

proposed an empirical measure based on this property for identifying satisfiable and unsatisfiable 3-SAT instances. We have empirically observed that the measure tends to be a constant for a given value \( m/n \). This result might be used to partly explain why an instance is unsatisfiable.

### REFERENCES


### Conclusion

We have found that it is possible to associate to each local consistency level in the constraint network equivalent inference rules in the propositional logic. We have expressed the path consistency technique in propositional logic and relating an interesting property of CSPs called Row Convexity with the pure literal rule. We have then