Probability in the Analysis of CNF Satisfiability Algorithms and Properties

John Franco

Computer Science, University of Cincinnati
Why is it Worthwhile?

The Questions:
- Why are some problems so difficult?
- Are there algorithms that will make them easier?

Why Probability?
- Results and process tend to draw out intuition
  - Identify properties that may be exploited by a fast algorithm and properties that may prevent exploitation.
  - Identify reasons for the hardness of various problems - why lots of instances are hard.
- Can explain the good or bad behavior of an algorithm
- Afford comparison of incomparable classes of formulas
- De-randomization may yield fast algorithms
**Terms**

A **variable** looks like this: $v_9$, takes a value from $\{0, 1\}$

A **positive literal**: $v_9$, a **negative literal**: $\neg v_9$

A **clause** looks like this: $(\neg v_1 \lor \neg v_2 \lor v_5 \lor v_9)$

An **instance** of SAT looks like this:

$$(v_1 \lor \neg v_2 \lor v_7) \land (\neg v_2 \lor v_6) \land (\neg v_2 \lor \neg v_4 \lor \neg v_5) \land (v_{10})\ldots$$

Clause **width**: $\#$ literals; $k$-**SAT**: fixed width $k$

An assignment of values satisfying $\mathcal{F}$ is a **model** for $\mathcal{F}$

An important **splitting** operation in solvers:

$$(\neg v_1 \lor \neg v_2 \lor v_i) \land (\neg v_i \lor v_3) \land (\neg v_2 \lor v_4)$$
Some problems

- Must assume an input distribution, often not reflecting reality (but sometimes we do not need to)
- Analysis can be difficult or impossible - algorithmic steps may significantly change distribution (known tools are limited)
- Can yield misleading results
A Misleading Result

Example: A probabilistic model for random formulas

Given: set \( L = \{v_1, \neg v_1, ..., v_n, \neg v_n\} \) of literals and \( 0 < p < 1 \)

Construct clause \( c \): \( l \in L \) independently in \( c \) with probability \( p \)

Construct formula: \( m \) independently constructed clauses

Justification: All formulas are equally likely if \( p = 1/3 \).

Applied to splitting (DPLL):

\[
(\neg v_1 \lor \neg v_2 \lor v_i) \land (\neg v_i \lor v_3) \land (\neg v_2 \lor v_4)
\]
A Misleading Result

Analysis sketch:

Given $m$ clauses, the average number of clauses removed when a value is assigned is $pm$

Let $T_i$ be the average number of clauses remaining on the $i^{th}$ iteration

Then $T_0 = m$ and $T_i = (1 - p)T_{i-1}$

For what $i$ does $T_i = 1$? When $1 = m(1 - p)^i$ or

$$\lg(m) = -i \lg(1 - p)$$

$$i = \frac{\lg(m)}{\lg(1 - p)}$$

So, size of search space is $2^i \approx 2^{\lg(m)/p} = m^c$ if $p = 1/3$

**Conclusion:** DPLL is fantastic, on the average!
A Misleading Result

Problems with the analysis:

- The input model is funky!
  The probability that a random assignment satisfies a random formula is
  \[(1 - (1 - p)^{2n})^m \approx e^{-me^{-2pn}}\]
  which tends to 1 if \(\ln(m)/pn = o(1)\) and means the average width of a clause can be at least \(\ln(m)\).
  With \(p = 1/3\), this holds if \(n > \ln(m)\).

- If average clause width is constant \((p = c/n)\)
  then the average search space size is
  \[2^{-\lg(m)/\lg(1-p)} \approx 2^{\lg(m)/(c/n)} = 2^{n \lg(m)/c} = m^{n/c}\]

Exponential complexity!
Probabilistic Toolbox

Linearity of expectation:

\[ E\left\{ \sum_i X_i \right\} = \sum_i E\{X_i\}, \quad X_i \text{ real valued r.v.s} \]

First Moment Method: show \( Pr(P) \to 0 \) as \( n \to \infty \).

Let \( X \) be a count of some entity
Suppose some property \( P \) holds if and only if \( X \geq 1 \).
Then
\[ Pr(X > 1) \leq E\{X\} \]
So, if \( E\{X\} \to 0 \) then \( Pr(P) \to 0 \), as \( n \to \infty \).
First Moment Method

An Example:

Assume $\mathcal{F}$ is a random $k$-SAT formula, $m$ clauses, $n$ variables

Let $P$ be the property that $\mathcal{F}$ has a model

Let $X$ be the number of models for $\mathcal{F}$

Let $X_i = \begin{cases} 1 & \text{if the } i^{th} \text{ assignment is a model for } \mathcal{F} \\ 0 & \text{otherwise} \end{cases}$

$$Pr(X_i = 1) = (1 - 2^{-k})^m$$

$$E\{X\} = \sum_{i=1}^{2^n} Pr(X_i = 1) = 2^n (1 - 2^{-k})^m$$

$$Pr(\mathcal{F} \text{ has a model}) \rightarrow 0 \text{ if } \frac{m}{n} > \frac{1}{\lg(1 - 2^{-k})} \approx 2^k$$

For $k = 3$ $\mathcal{F}$ has no model w.h.p. if $\frac{m}{n} > 5.19$
Probabilistic Toolbox

Flow Analysis:

Consider straight-line (non-backtracking) variants of DPLL

Let $F$ be a CNF Boolean expression
Set $M = \emptyset$
Repeat the following:
  Choose an unassigned literal $l$ in $F$
  If $l$ is a positive literal, set $M = M \cup \{l\}$
  Set $F = \{c \setminus \{\neg l\} : c \in F, l \notin c\}$
  If $F$ is satisfied then return $M$
  If some clause in $F$ is falsified return "give up"

When does this not give up with probability tending to 1?
Answer depends on the way literals are chosen
Flow Analysis

\[ C_k(j) \xrightarrow{z_k(j)} \]
\[ C_{k-1}(j) \xrightarrow{z_{k-1}(j)} w_{k-1}(j) \]
\[ \cdots \]
\[ C_2(j) \xrightarrow{z_2(j)} w_2(j) \]
\[ C_1(j) \xrightarrow{z_1(j)} w_1(j) \]
\[ w_0(j) \]

\[ E\{m_4(j)\} \]
\[ E\{m_3(j)\} \]
\[ E\{m_2(j)\} \]
Flow Analysis

Example:

Unit clause heuristic: When there is a clause with one unassigned variable remaining, set the value of such a variable so as to satisfy its clause.

Intuitively: If the clause flow $w_1(j) < 1$ then any accumulation of unit clauses can be prevented and no clauses will ever be eliminated.

Analysis:

Write difference equations describing flows:

$$m_i(j + 1) = m_i(j) + w_i(j) - w_{i-1}(j) - z_i(j), \quad \forall 0 \leq i \leq k, 1 < j < n$$

Take expectations and rearrange:

$$E\{m_i(j + 1) - m_i(j)\} = E\{w_i(j)\} - E\{w_{i-1}(j)\} - E\{z_i(j)\}$$
Flow Analysis

Compute the expectations:

\[ E\{z_i(j)\} = E\{E\{z_i(j)|m_i(j)\}\} = E\left\{ \frac{i \cdot m_i(j)}{2(n-j)} \right\} = \frac{i \cdot E\{m_i(j)\}}{2(n-j)} \]

\[ E\{w_i(j)\} = E\{E\{\ldots\}\} = E\left\{ \frac{(i+1)m_{i+1}(j)}{2(n-j)} \right\} = \frac{(i+1)E\{m_{i+1}(j)\}}{2(n-j)} \]

Substitute into difference equations:

\[ E\{m_i(j + 1) - m_i(j)\} = \frac{(i+1)E\{m_{i+1}(j)\}}{2(n-j)} - \frac{i \cdot E\{m_i(j)\}}{n-j}, \quad i < k \]

\[ E\{m_k(j + 1) - m_k(j)\} = -\frac{k \cdot E\{m_k(j)\}}{n-j} \]

Switch to differential equations (use \(x\) for \(j/n\)):

\[ \frac{d\bar{m}_i(x)}{dx} = \frac{(i+1)\bar{m}_{i+1}(x)}{2n(1-x)} - \frac{i \cdot \bar{m}_i(x)}{n-j}; \quad \bar{m}_k(0) = m/n, \quad \bar{m}_i(0) = 0 \quad \text{for} \quad i < k \]

Solve:  

Translation:

\[ \bar{m}_i(x) = \frac{1}{2^{k-i}} \left( \frac{m}{n} \right) \binom{k}{i} (1-x)^i x^{k-i} \quad E\{m_i(j)\} = \frac{m}{2^{k-i}} \binom{k}{i} \left( \frac{1-j}{n} \right)^i \left( \frac{j}{n} \right)^{k-i} \]
Flow Analysis

The important flow:

\[ E\{w_1(j)\} = \frac{E\{m_2(j)\}}{(n-j)} = \frac{1}{2k-2} \binom{k}{2} \left(1 - \frac{j}{n}\right)^2 \left(\frac{j}{n}\right)^{k-2} m \]

Find location of maximum (set derivative = 0):

\[ j^* = \left(\frac{k-2}{k-1}\right) n \]

So,

\[ E\{w_1(j^*)\} < 1 \quad \text{when} \quad \frac{m}{n} < \frac{2^{k-1}}{k} \left(\frac{k-1}{k-2}\right)^{k-1} \]

for \( k = 3 \) this is \( \frac{m}{n} < \frac{8}{3} \)

Conclusion: unit clause heuristic succeeds with probability bounded by a constant when \( \frac{m}{n} < \frac{8}{3} \).

Flow Analysis

What makes this work?

The clausal distribution is the same, up to parameters $m$ and $n$, at each level (algorithm is myopic - no information is revealed)

There is pretty much never a sudden spurious flow

$$Pr(||C_i(j + 1) - |C_i(j)|| > n^{0.2}) = o(n^{-3})$$

The average flow change is pretty smooth

$$E{|C_i(j + 1)| - |C_i(j)|} = f_i(j/n, |C_0(j)|/n, ..., |C_k(j)|/n) + o(1)$$

$f_i$ is continuous and

$$|f_i(u_1, ..., u_{k+2}) - f_i(v_1, ..., v_{k+2})| \leq L \sum_{1 \leq i \leq j} |u_i - v_i|$$
## Flow Analysis

<table>
<thead>
<tr>
<th>literal selection</th>
<th>$k$-SAT</th>
<th>3-SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Choose from unit clause, otherwise randomly</td>
<td>$2^k/k$</td>
<td>2.66</td>
</tr>
<tr>
<td>Choose var with maximum difference between occurrences of positive and negative lits. Set value to maximize satisfied clauses</td>
<td>$c \cdot 2^k/k$</td>
<td>3.003</td>
</tr>
<tr>
<td>Choose randomly from a clause with fewest non-falsified literals</td>
<td>$1.125 \cdot 2^k/k$</td>
<td>3.09</td>
</tr>
<tr>
<td>Best possible myopic algorithm</td>
<td>$c \cdot 2^k/k$</td>
<td>3.26</td>
</tr>
</tbody>
</table>

### literal selection, non-myopic algorithms

| Greedy algorithm: maximize number of clauses satisfied, eliminate unit clauses when they exist | $c \cdot 2^k/k?$ | 3.52          |
| Maximize the expected number of models of the reduced instance                    | $c \cdot 2^k/k?$ | > 3.6?        |
Flow Analysis

Pr(success)

Typical probability curve of these algorithms

Is $\frac{2^k}{k}$ the crossover to unsatisfiability or is it $2^k$?
**Probabilistic Toolbox**

**Second Moment Method:** show $Pr(P) \rightarrow 1$ as $n \rightarrow \infty$

Let $P$ be some property of a formula

A **witness** for $P$: a structure whose presence in $\mathcal{F}$ implies $P$

Let $W = \{w : w$ is a witness for $P$ in $\mathcal{F}\}$

Let $X_i = \begin{cases} 1 & \text{if structure } s_i \in W \\ 0 & \text{otherwise} \end{cases}$

Let $X = \sum_i X_i$ and $E\{X_i\} = q$, then $E\{X\} \triangleq \mu = q|W|$

Suppose $\text{var}(X) \triangleq \sigma^2 = o(\mu^2)$. Then, since

$$Pr(X = 0) \leq \frac{\sigma^2}{\mu^2}$$

we get

$$Pr(P) \rightarrow 1 \text{ as } n \rightarrow \infty$$
Second Moment Method

To show $\sigma^2 = o(\mu^2)$:

Start with one witness $w$ chosen arbitrarily
Let $A_w$ be all witnesses sharing at least one clause with $w$
Let $D_w$ be all witnesses sharing no clause with $w$. Then

$$\sigma^2 = \mu(1 - q) + \mu \left( \sum_{z \in A_w} (Pr(z|w) - q) + \sum_{z \in D_w} (Pr(z|w) - q) \right)$$

Need $|A_w| \ll |D_w|$ or “little” overlap among witnesses since

$Pr(z|w) = Pr(z) = q$ if $z \in D_w$ so $\sum_{z \in D_w} (Pr(z|w) - q) = 0$.

If $\mu \to \infty$ and $\sum_{z \in A_w} Pr(z|W) \to o(\mu)$ as $n \to \infty$ then

$Pr(P) \to 1$ as $n \to \infty$
Where is the Crossover?

Cannot apply the second moment method directly to $k$-SAT - variance of the number of models is too high
the reason: the asymmetry of $k$-SAT

Let $z$ and $w$ be assignments agreeing in $\alpha n$ variables

$$Pr(z \text{ satisfies } F \mid w \text{ satisfies } F) = \left(1 - \frac{1 - \alpha^k}{2^k - 1}\right)^m$$

Variance gets too big, maximum occurs at $\alpha \neq 1/2$

$$\sum_{z \in A_w} Pr(z \mid w) = \sum_{0 < \alpha \leq 1} \binom{n}{\alpha n} \left(1 - \frac{1 - \alpha^k}{2^k - 1}\right)^m$$
Where is the Crossover?

Analyze a different problem: Not All Equal \(k\)-SAT

A NAE model: one for which every clause has at least one true and at least one false literal

\[
Pr(\exists \text{ a model for } \mathcal{F}) > Pr(\exists \text{ a NAE-model for } \mathcal{F})
\]

Let \(X = \text{number of models, } X_{\text{NAE}} = \text{number of NAE models}\)

\[
Pr(X_{\text{NAE}} = 0) > Pr(X = 0)
\]

\[
Pr(z \text{ NAE-satisfies } \mathcal{F} \mid w \text{ NAE-satisfies } \mathcal{F}) = \left(1 - \frac{1 - \alpha^k - (1 - \alpha)^k}{2^{k-1} - 1}\right)^m
\]

Variance is low, maximum term occurs at \(\alpha = 1/2\).

\[
\sum_{0 < \alpha \leq 1} \binom{n}{\alpha n} \left(1 - \frac{1 - \alpha^k - (1 - \alpha)^k}{2^{k-1} - 1}\right)^m \approx \frac{2^n(1 - 2^{1-k})^m}{\sqrt{n}} \approx o(\mu)
\]

\[
Pr(X_{\text{NAE}} = 0) < \frac{1}{\mu_{\text{NAE}}} \approx \frac{1}{2^n(1 - 2^{k-1})^m} \rightarrow 0 \text{ if } \frac{m}{n} > \frac{1}{\lg(1 - 2^{k-1})} \approx 2^{k-1}
\]
What About Easy Classes?

Examples:
  Horn, Hidden Horn, 2-SAT, Extended Horn, q-Horn, CC-balanced, SLUR, Matched, Linear Autarkies

Horn:
  every clause has at most one positive literal
  solved in linear time by unit clause algorithm

Probabilistic analysis of polytime solvable classes can reveal:
  • What critically distinguishes an easy class from more difficult classes
  • Whether one class is much larger than another incomparable class in a probabilistic sense
### Polynomial Time Solvable Classes

**Example: q-Horn**

<table>
<thead>
<tr>
<th>Horn clauses</th>
<th>Zero2-SA T</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1 0 1 0 0</td>
<td>0 0 0 0 0 0</td>
</tr>
<tr>
<td>1 -1 -1 0 1</td>
<td>0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 0 -1 -1 0</td>
<td>0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 1 -1 0 0</td>
<td>0 0 0 0 0 0</td>
</tr>
<tr>
<td>0 -1 -1 0 0</td>
<td>0 0 1 0 1 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2-SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1 0 0 -1 0</td>
</tr>
<tr>
<td>0 0 0 -1 0</td>
</tr>
<tr>
<td>0 0 -1 -1 0</td>
</tr>
</tbody>
</table>

If the **satisfiability index** of a given formula is no greater than 1, then the formula is q-Horn.

The class of formulas with a satisfiability index greater than $1 + n^{-\epsilon}$, for any $\epsilon$, is NP-complete.
Polynomial Time Solvable Classes

Vulnerability of q-Horn to particular cycles

\[ \ldots (u_2 \lor \neg u_1 \lor \ldots)^{u_1} (u_1 \lor \neg v_0 \lor \neg u_{3p} \lor \ldots)^{u_{3p}} (u_{3p} \lor \neg u_{3p-1} \lor \ldots)^{u_{3p}} \ldots \]

\[ \ldots (\neg u_{p-1} \lor u_p \lor \ldots)^{u_p} (\neg u_p \lor \neg v_0 \lor u_{p+1} \lor \ldots)^{u_{p+1}} (\neg u_{p+1} \lor u_{p+2} \lor \ldots)^{u_{p+1}} \ldots \]

<table>
<thead>
<tr>
<th>class</th>
<th>as ( n \to \infty )</th>
<th>( k )-SAT</th>
<th>3-SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horn</td>
<td>0</td>
<td>( m &gt; \epsilon )</td>
<td>( m &gt; \epsilon )</td>
</tr>
<tr>
<td>Hidden Horn</td>
<td>0</td>
<td>( m/n &gt; 1/(k - \lg(k + 1)) )</td>
<td>( m/n &gt; 1 )</td>
</tr>
<tr>
<td>q-Horn</td>
<td>0</td>
<td>( m/n &gt; 4/(k^2 - k) )</td>
<td>( m/n &gt; 0.66 )</td>
</tr>
<tr>
<td>SLUR</td>
<td>0</td>
<td>( m/n &gt; 4/(k^2 - k) )</td>
<td>( m/n &gt; 0.66 )</td>
</tr>
<tr>
<td>Matched</td>
<td>1</td>
<td>( m/n &lt; c_k, c_k \to 1 )</td>
<td>( m/n &lt; 0.64 )</td>
</tr>
<tr>
<td>No Cycles</td>
<td>1</td>
<td>( m/n &lt; 1.36/(k^2 - k) )</td>
<td>( m/n &lt; 0.226 )</td>
</tr>
</tbody>
</table>
Algorithms for Unsatisfiability

Resolution performs badly on random unsatisfiable formulas

\[ Pr(\text{random } k\text{-SAT formula is unsatisfiable}) \rightarrow 1 \quad \text{if} \quad \frac{m}{n} > 2^k \]

\[ Pr(\text{resolution does well}) \rightarrow 1 \quad \text{only if} \quad \frac{m}{n} > \left( \frac{n}{\lg(n)} \right)^{k-2} \]

Are there better alternatives?
Must avoid getting stuck on “sparse” nature of formulas

One possibility: Hitting Set
Focus attention on clauses which are all positive or negative
Let \( n^+ = \min \) number of 1 valued variables to satisfy positives
Let \( n^- = \min \) number of 0 valued variables to satisfy negatives
If \( n^+ + n^- > n \) then some variable must be set to 1 AND 0
That is impossible, conclude the formula is unsatisfiable
**Hitting Set**

**Construct graphs** $G^+, G^-$

Vertices are labeled as pairs of variables

Edge $\langle a, b \rangle \iff$ some clause contains all variables labeling $a, b$

$$(v_1 \lor v_2 \lor v_3 \lor v_4) \land (v_2 \lor v_3 \lor v_4 \lor v_5) \land (v_1 \lor v_3 \lor v_4 \lor v_5)$$
Hitting Set

Construct graphs $G^+, \ G^-$

Vertices are labeled as pairs of variables

Edge $\langle a, b \rangle \iff$ some clause contains all variables labeling $a, b$

$$(v_1 \lor v_2 \lor v_3 \lor v_4) \land (v_2 \lor v_3 \lor v_4 \lor v_5) \land (v_1 \lor v_3 \lor v_4 \lor v_5)$$

For $k = 4$, if a model exists, one graph has $|IS| > \frac{n^2}{8}$
Hitting Set

Construct matrices $M^+, M^-$

Columns and rows are indexed on vertices

$$M_{i,j} = \begin{cases} \frac{1-p}{p} & \text{if there is an edge between vertices } i \text{ and } j \\ 1 & \text{otherwise} \end{cases}$$

where $p$ represents a probability that can be adjusted

Exploit relationship between maximum eigenvalue and maximum IS

$$\alpha(G^+) < \lambda_1(M^+) \quad \text{and} \quad \alpha(G^-) < \lambda_1(M^-)$$

For purposes of proving a bound

If $p = \frac{\ln^7(n')}{n'}$ then $\max_i |\lambda_i(M)| = \frac{2n'}{\ln^{3.5}(n')} (1 + o(1))$ w.h.p.

This leads to $Pr(HS \text{ does well}) \rightarrow 1$ if $\frac{m}{n} > n^{(k-2)/2}$

Recall $Pr(resolution \text{ does well}) \rightarrow 1$ only if $\frac{m}{n} > n^{k-2}$. 
A De-randomized Algorithm for MAXSAT

MAX $k$-SAT

Given: A CNF formula $\mathcal{F}$ with $k$ literals per clause
Find: An assignment to the variables of $\mathcal{F}$ that satisfies a maximum number of its clauses.

Suppose $\mathcal{F}$ has variables $v_1, v_2, \ldots, v_n$. Define indicators

$$ A^i_{t_1, \ldots, t_j} = \begin{cases} 1 & \text{if clause } i \text{ satisfied given } v_1 = t_1, \ldots, v_j = t_j \\ 0 & \text{otherwise} \end{cases} $$

What is the probability that $A^i_{t_1, \ldots, t_j} = 1$ for random $t_{j+1}, \ldots, t_n$?

For example: $Pr((\neg v_1 \lor v_3 \lor \neg v_5) = 1 \mid t_1 = 1, t_2 = 0) = Pr(A^i_{1,0}) = 3/4$
Let $N$ be the number of satisfied clauses in $\mathcal{F}$. Then

$$E\{N\} = \sum_{i=1}^{n} Pr(A^i) = (1 - 2^{-k})m$$

and

$$Pr(A^i_{t_1,\ldots,t_{j-1}}) = (Pr(A^i_{t_1,\ldots,t_{j-1},1}) + Pr(A^i_{t_1,\ldots,t_{j-1},0}))/2$$

$$\leq \max \{Pr(A^i_{t_1,\ldots,t_j})\}$$

so keep choosing a value $t_j$, for $j = 1, 2, \ldots$ that maximizes $Pr(A^i_{t_1,\ldots,t_j})$ to get

$$E\{N\} = (1 - 2^{-k})m = \sum_{i=1}^{n} Pr(A^i) \leq \sum_{i=1}^{n} \max \{Pr(A^i_{t_1})\} \ldots$$

$$\leq \sum_{i=1}^{n} \max \{Pr(A^i_{t_1,\ldots,t_n})\} = \# \text{ clauses satisfied given } t_1, \ldots, t_n$$
A De-randomized Algorithm for MAXSAT

MAX $k$-SAT approximation algorithm uses this:

```c
bool chooseValue (Variable *var) {
    double sum_pos = 0.0, sum_neg = 0.0, prob = 0.5;
    for (int sz=1 ; sz <= k ; sz++) {
        sum_pos += var->no_clauses_as_pos_lit[sz]*prob;
        sum_neg += var->no_clauses_as_neg_lit[sz]*prob;
        prob *= 0.5;
    }
    return (sum_pos >= sum_neg) ? true : false;
}
```

and will always find an assignment that satisfies at least
$(1 - 2^{-k})m$ clauses - for any input formula!