A Polynomial Time Algorithm for 3SAT

ROBERT QUIGLEY

It is shown that any two clauses in an instance of 3SAT sharing the same terminal which is positive in one clause and negated in the other can imply a new clause composed of the remaining terms from both clauses. Clauses can also imply other clauses as long as all the terms in the implying clauses exist in the implied clause. It is shown an instance of 3SAT is unsatisfiable if and only if it can derive contradicting 1-terminal clauses in exponential time. It is further shown that these contradicting clauses can be implied with the aforementioned techniques without processing clauses of length 4 or greater, reducing the computation to polynomial time. Therefore there is a polynomial time algorithm that will produce contradicting 1-terminal clauses if and only if the instance of 3SAT is unsatifiable. Since such an algorithm exists and 3SAT is NP-Complete, we can conclude P = NP.

CCS Concepts: • **Theory of computation → Complexity classes**.

Additional Key Words and Phrases: NP-Complete, SAT, 3SAT, Satisfiability, P vs NP, Polynomial Time, Non-deterministic Polynomial Time, Complexity Theory, Computational Complexity, Complexity Classes, Cook-Levin Theorem, Karp's 21 NP-Complete Problems

ACM Reference Format:
Robert Quigley. 2024. A Polynomial Time Algorithm for 3SAT. 1, 1 (January 2024), 27 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

This section introduces the 3SAT problem, the implications of solving it in polynomial time, and the structure of the paper.

As seen in [1], The boolean satisfiability problem is given as a set of terminals, $x_1, x_2, ..., x_n$, each of which can be assigned a value of True or False, combined by logical AND operators, logical OR operators, and negations. The problem is to determine whether or not there exists an assignment for each terminal that allows the instance to evaluate to True. As seen in [2], the satisfiability problem with at most 3 literals per clause is NP-Complete. This is the same as the boolean satisfiability problem, but it is presented such that a clause contains exactly three terminals combined with logical OR operators and possbily negations, and each clause is combined with logical AND operators. The idea of NP-completeness shows that if one NP-complete problem can be solved in polynomial time, then all problems in the class NP can be solved in polynomial time. In other words, P = NP.

The paper is structured as follows:

(1) Introduction
(2) Standard Definitions
    - Terms relating to the 3SAT problem
(3) Algorithm-Specific Definitions
    - Terms relating to specific aspects of 3SAT regularly referenced in this paper

Author's address: Robert Quigley, robertquigley21@gmail.com.

Manuscript submitted to ACM

(4) Reformatting and Processing
   - Defines how clauses and instances of the 3SAT problem will appear within the paper

(5) Lemmas
   - A list of lemmas and their proofs pertaining to the algorithm

(6) Algorithm
   - A step-by-step description of the algorithm to solve 3SAT in polynomial time

(7) Time Complexity Analysis
   - A step-by-step analysis of the time complexity of the algorithm

(8) Proof of Correctness
   - A proof showing the algorithm works for every instance of 3SAT

(9) Conclusion

(10) References

2 STANDARD DEFINITIONS

Definition 2.1. Terminals: symbols used in the 3SAT problem that can be assigned a value of either 0 or 1, True or False, or any other binary assignment. They usually take the form x_i where i is a natural number.

Definition 2.2. Terms: terminals in either the positive or negated form that appear in a clause. If a term is positive, then the value of the terminal will be the same as the value of the term. If a term is negated, then the value of the terminal will be the opposite of the value of the term.

Definition 2.3. Clauses: a set of terms combined by logical OR operators. Clauses usually take the form $(x_i \vee \neg x_j \vee x_k)$ where $x_i \neq x_j \neq x_k$

Definition 2.4. (3SAT) Instance: a set of any number of clauses combined by logical AND operators. Terminals may not repeat within a clause [1] , but they are free to repeat between clauses. Instances usually take the form:
$(x_i \vee \neg x_j \vee x_k) \wedge (x_l \vee x_m \vee x_n)$

Definition 2.5. Assignment: A list of values in which each value represents either True or False such that each item in the list corresponds to a terminal and all terminals are assigned a value.

Definition 2.6. Satisfying Assignment: An assignment, A, is said to satisfy the instance, if applying A will make the instance evaluate to True.

Definition 2.7. Satisfiable: an instance is satisfiable iff there exists a satisfying assignment.

Definition 2.8. Unsatisfiable: an instance is unsatisfiable iff there does not exist a satisfying assignment.

3 ALGORITHM-SPECIFIC DEFINITIONS

Definition 3.1. Blocking an Assignment: An assignment, A, is said to be blocked by a clause, C if, given an instance containg C, there is no way that A allows C to evaluate to True, and thus there is no way A allows the instance to evaluate to True.

[1] See Lemma 5.3

Definition 3.2. Implication: A clause, C, is said to imply another clause, D, if all assignments blocked by D are also blocked by C.

Definition 3.3. Given Clauses: clauses which were given in the original instance.

Definition 3.4. Implied Clauses: clauses which are implied by the clauses in the original instance.

Definition 3.5. k-terminal (k-t) clause: a clause is described as a k-terminal or a k-t clause if there are k terminals in the clause.

Definition 3.6. Reduction: A special type of implication in which the implied clause is shorter than the implying clause(s).

Definition 3.7. Expansion: A special type of implication in which the implied clause is longer than the implying clause(s) and all terms in the implying clause exist in the implied clause.

Definition 3.8. Contradicting Clauses: A set of clauses is said to be contradicting if they block every assignment.

4 REFORMATTING AND PROCESSING

Since there are a lot of constant characteristics about an instance of 3SAT, we can remove most of them to allow ourselves to focus only on what changes from instance to instance. A list of unchanging characteristics follows:

- the symbol x
- logical AND operators
- logical OR operators

The only difference between instances, therefore, is the subscript of the terminal.

Additionally, the following items will be changed to improve compatibility with the Python programming language wherein an instance is expressed as a list of lists and each inner list represents a clause:

- parentheses will become square brackets
- negation symbols will become minus signs
- an instance may be surrounded with square brackets to show it is a list of lists

For example, the instance:
$(\neg x_a \vee x_b \vee x_c) \wedge (x_a \vee x_d \vee x_e)$
will be written as:
$[[-a, b, c], [a, d, e]]$

Instances of 3SAT will further be processed by removing any clauses that do not block any assignments. Since this algorithm relies on implications of new clauses, if we ever come across a clause that blocks no assignments, then by definition it will not be able to imply any additional clauses that block at least one assignment.

As such, we will ignore any clauses given or derived that are described in Lemma 5.3

5 LEMMAS

Lemma 5.1. *A clause can block an assignment*

Proof. Recall the definition of a clause blocking an assignment: An assignment is blocked if it does not allow the clause to evaluate to True.

Since terms in a clause are combined by logical OR operators, a clause cannot evalute to True iff all terms evalute to False.

A term evaluates to False if it's either (1) negated and the terminal's value is True or (2) positive and the terminal's value is False

Given a clause, we know there are some number of unique terminals and we can find assignments where all terms evalute to False.

Since all the terms evaluate to False and are combined by logical OR operators, the clause will evalutae to False.

Since the instance is composed of clauses combined by logical AND operators and one clause evalutes to False, then the entire instance evalutes to False.

Since the instance evaluates to False, the assignment cannot satisfy the instance. □

Lemma 5.2. *For a given instance with n terminals, there are 2^n possible assignments.*

Proof. An assignment for this instance consists of n values, each with two possible values, True or False.

Therefore, there are 2^n possible assignments. □

Lemma 5.3. *If a clause contains the same terminal in its negated and positive form, it will not block any assignments.*

Proof. Consider a clause containing the same terminal in both the positive and negative form.

We know that terminal must either be True or False. Consider both cases:

That terminal's value is True: The positive form of the terminal will be True and the clause will evaluate to True.

That terminal's value is False: The negated form of the terminal will be True and the clause will evaluate to True.

Since the clause evaluates to True in every case, there will never be an assignment with which it is impossible to make this clause evalute to False. □

Lemma 5.4. *Each clause of length k blocks 2^{n-k} assignments.*

Proof. Consider the generic k-terminal clause, C, in an instance with n terminals.

As seen in Lemma 5.1, this blocks all assignments where all of the terms evaluate to False.

Since the values for k terminals are set, there are $n - k$ terminals left whose values could be True or False.

Since an assignment exists for every possible way to assign values to each terminal, we know an assignment exists for every possible way to assign a value for these $n - k$ terminals.

There are two possible ways to assign values to each of these $n - k$ terminals so there are 2^{n-k} unique assignments blocked by C. □

Lemma 5.5. *For any clause, C, if we select a terminal, t, that's not in C then half of the assignments blocked by C will assign True to t and the other half will assign False to t.*

Proof. We have a clause, C, of fixed yet arbitrary length:

$[a, b, c, ...]$

Now we select a terminal, t, that's not in C, say we call this terminal t.

Want to show half of the assignments blocked by C assign True to t and the other half assign False to t.

We know there will be no overlap between these assignments because a single assignment cannot assign both the values True and False to the same terminal.

Now we just have to show that exactly half of the assignments are blocked by assigning either True or False to t.

Let's say there are k terms in C, then we know it blocks assignments where all k terms evaluate to False.

By Lemma 5.4, this clause blocks 2^{n-k} assignments.

If we fix the value of t, then there are only $n-k-1$ terminals whose values could be 0 or 1. Since we have two choices per terminal and there are $n-k-1$ terminals, then there are 2^{n-k-1} assignments blocked by C where the value of t is fixed.

Divide to get the ratio of the number of assignments blocked by adding t to the number of assignments blocked by C without t:

$2^{n-k-1}/2^{n-k}$

$= 2^{n-k-1-(n-k)}$

$= 2^{-1}$

$= 1/2$

This shows that half of the assignments blocked by C assign a fixed value to t.

Since there are two possible values for t and each block mutually exclusive halves of the assignments blocked by C, the lemma holds.

For any clause, C, if we select a terminal, t, that's not in C then half of the assignments blocked by C will assign True to t and the other half will assign False to C. □

Lemma 5.6. *Given a clause, C, and another clause, D, such that all of the terms in C also exist in D, then all of the assignments blocked by D are also blocked by C.*

Proof. Given a clause, C, of a fixed yet arbitrary length:

$[a, b, c, ...]$

And another clause, D, containing all the terms in C with possibly additional terms:

$[a, b, c, ..., d, e, f, ...]$

Want to show all the assignments blocked by D are also blocked by C.

We know that C blocks all assignments that cause all the terms to evaluate to False.

In other words, C blocks all assignments where

$a = b = c = ... = False$

Similarly, D blocks all assignments where

$a = b = c = ... = d = e = f = ... = False$

Clearly all assignments consistent with the terminal assignments from D are also consistent with the terminal assignments from C.

Therefore, every assignment blocked by D is also blocked by C. □

Lemma 5.7 (Reduction). *Given the following conditions:*

- *A is a clause of length k*
- *B is a clause of length k*
- *A and B share $k-1$ identical terms*
- *A and B share one terminal that is negated in one clause and positive in the other*
- *C is a clause of length $k-1$ composed of the shared terms from A and B*

Then A and B imply C.

Proof. Given clauses consistent with the description:

$A := [a, b, c, ...i]$

$B := [a, b, c, ..., -i]$

where $a, b, c, ...$ is shared between the clauses and i is a terminal not in $a, b, c, ...$

Want to show this implies the described clause.

Recall by Lemma 5.3 that the same terminal cannot appear both negated and positive within the same clause and still block an assignment, so i cannot exist in $a, b, c...$

Consider the clause

$C := [a, b, c, ...]$

We know by Lemma 5.5 that if we select a terminal that's not in C, say t, then half of the assignments blocked by C assign True to t and the other half of the assignments blocked by C assign False to t.

Let this terminal t that's not in C be the terminal i that's in A and B.

We know that A blocks all assignments blocked by C where i is assigned the value of False.

We know that B blocks all assignments blocked by C where i is assigned the value of True.

Since A and B both block mutually exclusive halves of the assignments blocked by C, we can say that A and B imply C. □

Lemma 5.8 (Expansion). *Given a clause, C, and a terminal, t, that's not in C, two new clauses can be implied consisting of all the terms of C appended to either the positive form of t or the negated form of t.*

Proof. Given a clause, C, and a terminal, t, that's not in C:

$C := [a, b, c, ...]$

We can compose two new clauses:

$D := [a, b, c, ..., t]$

$E := [a, b, c, ..., -t]$

Since D and E both contain all the terminals from C, by Lemma 5.6 then D and E are implied by C. □

Lemma 5.9 (General Lemma 5.7). *If two clauses share the same terminal, t, such that t is positive in one clause and negated in the other, then these clauses imply a new clause which is composed of all the terms in both clauses that do not have t.*

Proof. Consider two clauses,

$C := [a, b, c, ..., t]$

$D := [d, e, f, ..., -t]$

Where within a clause, the same terminal does not repeat, but between clauses the same terminal may repeat.

Want to show we can imply a clause consistent with the lemma description:

$E := [a, b, c, ..., d, e, f, ...]$

Let's define some additional clauses:

$E' := [a, b, c, ..., d, e, f, ..., t]$

$E'' := [a, b, c, ..., d, e, f, ..., -t]$

By Lemma 5.6, we know that C implies E', ie, all assignments blocked by E' are also blocked by C.

By Lemma 5.6, we know that D implies E''.

By Lemma 5.7, since E' and E'' share all the same terms except for t, which is positive in one clause and negated in the other, we can create a new clause composed of all the shared terms in E' and E''.

Such a clause is already defined as E.

Now there are a couple extra cases to consider:

- There is some overlap between a, b, c, ... and d, e, f, ...
- There is the same terminal that's positive in a, b, c, ... and negated in d, e, f, ...

First, if the same term exists in a, b, c, ... and d, e, f, ..., then we can just remove one of the duplicates since one term being true implies an identical term being True.

Secondly, if the same terminal exists, but is of the opposite form in a, b, c, ... and d, e, f, ... then by Lemma 5.3, this clause will always be True and thus blocks no assignments. In this case, the lemma is vacuously true, but we disregard the clause as it is of no value.

□

Lemma 5.10. *Given two clauses of lengths k and m that share a terminal, t, which is positive in one clause and negated in the other, you will be able to directly imply clauses of length $max(k, m) - 1$ to $(k + m - 2)$ where the function $max(a, b)$ represents the parameter with the greatest value.*

Proof. The smallest clause that can be implied by clauses of length k and m using Lemma 5.9 occur when all but one of the terms in one clause exist in the other.

As such, the unique terms will come from the clause that's longer.

Removing t, you are left with 1 less than the maximum of k and m.

The largest clause can be implied if there are no terms shared between the two clauses. In this case you subtract 1 from the length of each clause to account for t and since no duplicates will be removed, the resulting clause's length is 2 less than the sum of the lengths of the clauses. □

Lemma 5.11. *Given four clauses of length $k - 1$, A, B, C, and D, and one clause of length k, E, such that*

- *A and B imply E by Lemma 5.9*
- *C and E imply D by Lemma 5.9*

Then A, B, and C, imply D by processing only clauses with a maximum length of $k - 1$.

Proof. Consider the following figure:

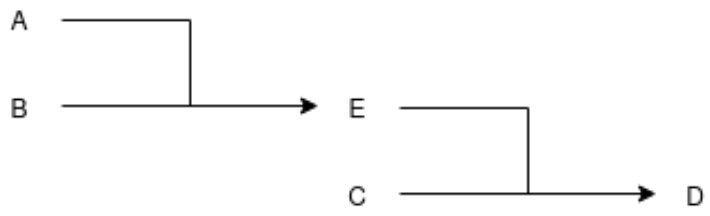

Fig. 1. A graph illustrating the derivations described by the lemma

Define the clauses in the following manner:

A := [a, b, β, i]

B := [c, d, δ -i]

C := [-a, e, f, ϕ]

Then the following are derived by Lemma 5.9:

$E = [a, b, \beta, c, d, \delta]$ (By A and B)

$D = [b, \beta, c, d, \delta, e, f, \phi]$ (By C and D)

$F = [b, \beta, i, e, f, \phi]$ (By A and C)

Where β, δ, and ϕ are generic sets of terms in the instance.

Note that since C and E imply D, then C and E must share the same terminal such that it is positive in one clause and negated in the other. Since all of the terms in E came from A or B (not including i), then such a term in C must have the same term of the opposite form in A or B (again it cannot be i since i is not in E). And since A and B are fixed yet arbitrary sets, it does not matter which clause we pick as long as it is fixed for the rest of the proof. Let's pick a the terminal, a, from clause A. Now C contains $-a$.

Recall from Lemma 5.3 that if a clause blocks any assignments, it cannot contain the same term in both forms, so if β, δ, or ϕ contain the same terminal in both forms then D blocks no assignments and the resulting clause may be disregarded.

We know C is of length k and D is of length $k - 1$ or k.

In the following equations, let the presence of a term represent a count of one, and the presence of a set of terms represent the number of terms in that set. If multiple sets of terms are shown in parentheses, let this represent the number of terms found in both sets.

Consider the case where D has length k then we can define k in terms of D:

$k = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

length of $F = b + i + e + f + \beta + \phi - (\beta\phi)$

Want to show length of F is less than k

$b + i + e + f + \beta + \phi - (\beta\phi) < b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

$\rightarrow i + \beta + \phi - (\beta\phi) < c + d + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

$\rightarrow i < c + d + \delta - (\beta\delta) - (\delta\phi) + (\beta\delta\phi)$

As seen by using a Venn Diagram or other set intuition, $\delta - (\beta\delta) - (\delta\phi) + (\beta\delta\phi)$, represents the number of terminals in δ not in β and not in ϕ.

The lowest case for the right hand side of the inequality is where this is 0, ie, all of the terms in δ are in β or ϕ. In this case, the inequality becomes:

$\rightarrow i < c + d$

Which is true as long as c and d exist in B.

Want to show c and d always exists in B:

Suppose not, then we can rewrite B without c or d:

$B := [\delta, -i]$

Notice that δ is simply a fixed, yet arbitrary set of terms following the rules of a valid clause.

As long as δ contains at least two terms, we can simply use any two terms from δ as c and d.

If δ does not contain two terms, then the largest possible length of B is 2.

By Lemma 5.10, the largest clause this can imply is of length $(|A| + |B| - 2)$ where $|A|$ is the length of A and $|B|$ is the length of B.

This means the largest E implied by A and B has the same length as A.

Because the lengths are equal, such a case would not allow A to have length $k - 1$ and E to have length k.

Such a case would be inconsistent with the conditions of this lemma and would not apply.

Therefore, at least two terms have to exist in $\{c, d\}$.

Since c and d always exist in B, you can derive D by processing clauses of length at most $k - 1$.

Consider the case where the length of D is $k - 1$:

This means k is one greater than the length of D, so k is now:

$k = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi) + 1$

Similarly as before, the inequality will become:

$\rightarrow i < c + d + 1$

Which is again true as long as c and d are in B and i is in A.

As seen in the first half of this proof, c and d have to be in B.

i has to be in A by the conditions of this lemma.

Since the length of F is less than k in all cases, you can derive D by processing only clauses with a maximum length of $k - 1$. □

Lemma 5.12. *Given the following:*

- *A is a clause of length less than k*
- *B is a clause of length k*
- *C is a clause of length less than k*
- *D is a clause of length k or k − 1*
- *A expands to imply B by Lemma 5.8*
- *B and C imply D by Lemma 5.9*

Then A and C can imply D by processing clauses of at most length k − 1.

Proof. Consider the following figure

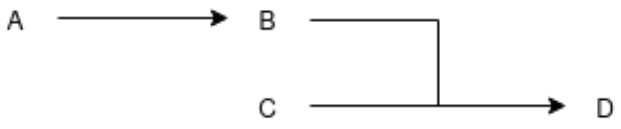

Fig. 2. An illustration of the derivations described in the lemma

Let the clauses be defined with the following generic definitions:

$A := [a, b, \beta]$

$A' := [a, b, \beta, c]$

$B := [a, b, \beta, c, d, \delta]$

$C_1 :=$ [-a, e, f, ϕ]

$C_2 :=$ [-c, e, f, ϕ]

$D_1 :=$ [b, β, c, d, δ, e, f, ϕ]

$D_2 :=$ [a, b, β, d, δ, e, f, ϕ]

E := [b, β, e, f, ϕ]

F := [a, b, β, e, f, ϕ]

Where β, δ, and ϕ are fixed, yet arbitrary sets of terms abiding by the rules for a valid clause (see Lemma 5.3) and consistent with the conditions of this lemma.

Notice that C has to contain a term from B in the opposite form by lemma 5.9.

Notice that B is made up of terms from A and terms not in A by lemma 5.6.

The term in C which is opposite from the term in B can therefore be opposite (1) from a term in A (in this case, use C_1 and D_1) or (2) a term not in A (in this case use C_2 and D_2).

(1) Consider the opposite form term in C is in A, ie, use C_1 and D_1:

Notice A and C_1 share an opposite term, then they can derive a clause, E, by Lemma 5.9. And E can imply a clause D_1 by Lemma 5.8:

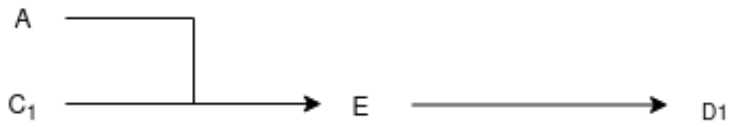

Fig. 3. An illustration of the derivations by clauses A, C_1, and E

Want to show you only have to process clauses whose length is less than k to derive D_1.

Want to show length of E is less than k.

Consider two cases: (1a) D is of length k and (1b) D is of length $k-1$

Consider (1a) where D is of length k:

In the following equations, let the presence of a term represent a count of one, and the presence of a set of terms represent the number of terms in that set. If multiple sets of terms are shown in parentheses, let this represent the number of terms found in both sets.

Since D is of length k, we can define k as follows:

$k = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

Length of $E = b + e + f + \beta + \phi - (\beta\phi)$

Want to show the length of E is less than k:

$b + e + f + \beta + \phi - (\beta\phi) < b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

$\rightarrow 0 < c + d + \delta - (\beta\delta) - (\delta\phi) + (\beta\delta\phi)$

Using a Venn Diagram or by other set intuition, $\delta - (\beta\delta) - (\delta\phi) + (\beta\delta\phi)$ represents the number of terms in δ not in β and not in ϕ. The lowest this value can be is 0 if all terms in δ are in β or ϕ.

The inequality becomes:

$\rightarrow 0 < c + d$

Which is true as long as c or d exist in B.

Want to show c or d always exists in B.

Suppose not, then we can rewrite B as follows:

$B := [a, b, \beta]$

Note that the maximum length of δ is 0 because if it were 1 or more, then we can just extract c or d from δ.

In this case, the length of B is the same as the length of A.

Recall the conditions of the lemma defined the length of B as k and the length of A as $k-1$.

Therefore it is impossible for the lemma conditions defining the lengths of A and B to be true and c or d must always exist in B.

Since c or d must exist in B, the inequality is always true, and the length of E is indeed less than k.

Consider (1b) D is of length $k-1$:

Then the length of k is redefined as:

$k = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi) + 1$

Similarly as before, the inequality becomes:

$\rightarrow 0 < c + d + 1$

Which is always true so the length of E is indeed less than k.

Notice that, by lemma 5.8, you only have to process clauses of length at most $k-1$ to derive a clause of length k. In context, you can derive D via expansion while only processing clauses with a maximum length of $k-1$.

(2) Consider the case using C_2 and D_2:

We can construct a clause, A', such that A expands to A' by Lemma 5.8 and there is a term in A' whose opposite term is in C_2.

By Lemma 5.9, A' and C_2 imply a clause, F.

And by Lemma 5.6, all the terms in F are in D_2 so F implies D_2.

This is seen in the following figure:

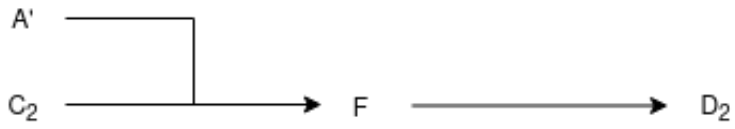

Fig. 4. An illustration of the derivations using A', C_2, and F

Want to show (2a) A' is shorter than k and (2b) F is shorter than k.

(2a) Want to show the length of A' is less than k:

Consider two cases, (2ai) D_2 is of length k and (2aii) D_2 is of length k - 1

Consider (2ai) D_2 is of length k, then we can define k as follows:

$k = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

length of $A' = a + b + \beta + c$

Want to show the length of A' is less than k:

$\rightarrow a + b + \beta + c < b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

$\rightarrow a < d + e + f + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

By Venn Diagram or other set intuition, $\delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$ represents the number of unique terms in δ and ϕ. The smallest value for this is when $\delta = \phi$, so this can be replaced with ϕ.

$\rightarrow a < d + e + f + \phi$

Which is clearly true as long as there are at least two terms in d, e, f, δ or ϕ.

Consider the cases where 0 terms exist in d, e, f, δ, ϕ

Then

$D_2 := [a, b, \beta]$

Recall

$A := [a, b, \beta]$

It is seen A and D_2 are the same length, but the conditions of the lemma state A is of length less than k and D is of length k.

This is a contradiction so at least one term has to exist in d, e, f, δ, ϕ

Consider the cases where 1 term exists in d, e, f, δ, ϕ

Then

$D_2 := [a, b, \beta, x]$

Where x is a single term from $\{d, e, f, \delta, \phi\}$

Recall $A = [a, b, \beta]$

Since x is exactly one term, D_2 is larger than A.

The clause A can expand to D_2 by Lemma 5.6. In this case, we can expand A to D_2 directly and we do not necessarily need A',

Since D_2 is of length k, it can be expanded to by processing only clauses with a maximum length of $k - 1$.

Therefore, either the inequality holds, or if it doesn't we can derive D_2 directly without processing clauses longer than length $k - 1$.

Consider (2aii) D_2 is of length $k - 1$:

k is now defined as:

$k = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi) + 1$

Similarly as before, the inequality becomes:

$\rightarrow a < d + e + f + \phi + 1$

Where $\phi = \delta$

Which is true as long as there is at least one term in d, e, f, δ or ϕ.

As seen before, if there are fewer than two terms in $\{d, e, f, \delta, \phi\}$ then we can derive D_2 without processing clauses larger than length $k - 1$.

Therefore if D_2 is of length $k - 1$, then either the length of A' is shorter than k or D_2 can be derived directly from A whose length is less than k.

(2b) want to show F is shorter than k.

Consider two cases, (2bi) D_2 is of length k and (2bii) D_2 is of length $k - 1$.

Consider (2bi) where D_2 is of length k:

By D_2, k is defined as

$k = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

length of $F = a + b + e + f + \beta + \phi - (\beta\phi)$

Want to show the length of F is less than k:

$a + b + e + f + \beta + \phi - (\beta\phi) < b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

$\rightarrow a < c + d + \delta - (\beta\delta) - (\delta\phi) + (\beta\delta\phi)$

Using a Venn Diagram or by other set intuition, $\delta - (\beta\delta) - (\delta\phi) + (\beta\delta\phi)$ represents the terms in δ that are not in β or ϕ.

This is lowest when all the terms in δ are in β or ϕ, this part of the inequality then becomes 0.

Then the inequality becomes:

$\rightarrow a < c + d$

Which is clearly true if c and d exist.

Consider the case where c and d don't exist.

Then B can be redefined:

$B := [a, b, \beta, x]$

where x is a single term. Note that if x is more than one term, then we could use the two terms in x as c and d.

Since x is a single term, the length of B is the same as the length of A' which was already shown to be shorter than k or irrelevant as D_2 could be derived directly from A without processing clauses of length greater than $k-1$.

Consider (2bii) where D_2 is of length $k-1$.

Then k is defined as:

$k = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi) + 1$

Want to show length of F is less than k:

$a + b + e + f + \beta + \phi - (\beta\phi) < b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi) + 1$

Similarly as before, this becomes:

$\rightarrow a < c + d + 1$

Which is always true since we already proved c and d have to exist or if they don't we can derive D_2 without processing clauses greater than length $k-1$.

Since the cases where we cannot directly derive D_2 by processing clauses with a maximum length of $k-1$ imply the inequality holds, then F is smaller than k and D_2 has a maximum length of k, and by Lemma 3.6 the largest a clause has to be in order to imply another clause of length k is $k-1$. Meaning you only have to process clauses of length $k-1$ in order to derive a clause of length k.

Since A, C_1, C_2, E, and F are all shorter than k or D can be directly derived from clauses shorter than k and D can be derived by expanding E or F, then D can be derived without the need to process clauses of length greater than $k-1$. □

Lemma 5.13. *Given an instance of 3SAT, you can expand all of the given clauses to the point where you are considering clauses of length n.*

Proof. Given an instance of 3SAT, we know all clauses are of length 3.

If we want to consider a generic n-terminal clause, B, that's implied by a given clause, A, then by Lemma 5.6 we know it's implied iff B contains all of the terms in A. □

Lemma 5.14. *If you expand given 3-t clauses as described in Lemma 5.13, you will derive 2^n unique clauses of length n iff the instance is unsatisfiable.*

Proof. Want to show an unsatifiable instance $\implies$ 2^n unique n-terminal clauses can be derived from the given 3-t clauses:

By lemma 5.4, a clause of length n blocks 1 assignment.

Recall an instance is unsatisfiable iff all 2^n assignments are blocked.

If a 3-terminal clause blocks an assignment, then it also implies the corresponding n-terminal assignment because there is one possible n-terminal clause for any assignment.

Since all assignments are blocked, there exists a 3-terminal clause for each assignment such that the terminals in the clause are assigned to make the clause evalute to false.

Similarly, for each assignment, there exists an n-terminal clause such that the terminals in the clause are assigned to make the clause evaluate to false.

This will create a unique n-terminal clause for each assignment since each clause can block only one assignment and overlap would imply the same terminal having two values in the same assignment.

Notice that for each of these n-terminal clauses, they must contain three terms from at least one given clause. If they didn't, then the assignment blocked by that n-terminal clause would not be blocked and the instance would be satisfiable.

Since each n-terminal clause blocks one assignment, blocking all assignments requires 2^n n-terminal clauses.

Want to show 2^n unique n-terminal clauses are derived by the given 3-t clauses $\implies$ then the instance is unsatifiable.

By lemma 5.4, a clause of length n blocks 1 assignment.

Therefore if there are 2^n unique n-terminal clauses, then all 2^n assignments will be blocked.

Note that there will be no overlap because each n-terminal clause sets the value for each terminal and overlap would imply the same terminal having two values by the same assignment which is impossible. □

Lemma 5.15. *The n-terminal clauses described in Lemma 5.14 can be reduced to derive any pair of contradicting 1-terminal clauses.*

Proof. Given 2^n n-terminal clauses, want to show you can imply any pair of contradicting 1-terminal clauses by lemma 5.7.

Pick a terminal that will not exist in the final 1-terminal clauses.

Half of the existing n-terminal clauses have that terminal assigned the value of False and the other half have that terminal assigned the value of True.

Pick one clause that blocks an assignment where the terminal is True.

Then there exists an assignment for each possible value for the remaining n-1 terminals.

Therefore, there must exist another clause that shares all of the same terms, but where that one terminal is assigned the value of False.

Using these two terms, we can create a new clause by lemma 5.7.

Now all of the clauses of length n - 1 do not contain that terminal.

Repeat this process while never selecting the same terminal twice until you are left with two contradicting 1-terminal clauses. □

Lemma 5.16. *Contradicting 1-terminal clauses can be expanded to imply every possible clause.*

Proof. Let the following clauses be a pair of contradicting 1-t clauses:

[a]

[-a]

By lemma 5.6, we can expand to any clause that contains a or $-a$.

Let the following be a generic 3-terminal clause that does not contain a or $-a$:

[b, c, d, ...]

By Lemma 5.6, we know the 1-terminal clauses imply the following 3-t clauses:

[a, b]

[-a, c, d, ...]

By Lemma 5.9, these clauses imply:

[b, c, d, ...]

Therefore we can imply any clause containing a, $-a$, or neither, which encompasses every possible clause. □

Lemma 5.17. *Given the following:*

- *A, B, C, and D, are clauses shorter than k*
- *E and F are clauses of length k*
- *G is a clause of length k or k - 1*
- *A and B imply E*
- *C and D imply F*
- *E and F imply G*

Then G can be implied by processing clauses with a maximum length of k - 1

Proof. Consider the following figure:

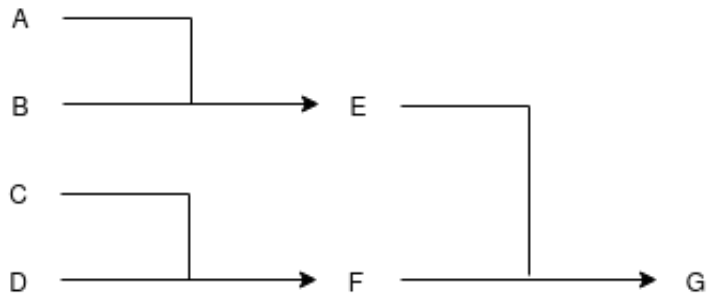

Fig. 5. An illustration of the derivations described in the lemma

Let the clauses be defined as follows:

$A := [a, b, \beta, i]$

$B := [c, d, \delta, -i]$

$C := [-a, f, \phi, j]$

$D := [g, h, \gamma, -j]$

Then the following are implied by Lemma 5.9:

$E = [a, b, \beta, c, d, \delta]$

$F = [-a, f, \phi, g, h, \gamma]$

$G = [b, \beta, c, d, \delta, f, \phi, g, h, \gamma]$

Where β, δ, ϕ, and γ are fixed yet arbitrary sets of terms consistent with the rules for a valid clause as by Lemma 5.3.

Notice that E and F have to share a term of the opposite form in order to imply G by Lemma 5.9. All of the terms in E and F came from A, B, C, and D. Since A, B, C, and D are all arbitrary clauses, selecting which term to negate does not have an effect on the outcome as long as one form of the term exists in E and the other form exists in F. Here the opposite term is shared between A and D, but any term that appears positive in A or B and negated in C or D will yield the same results.

Want to show G can be derived by processing clauses with a maximum length of k - 1.

Define additional implications:

$H = [b, \beta, i, f, \phi, j]$ (By clauses A and C)

$I = [c, d, \delta, b, \beta, f, \phi, j]$ (By clauses B and H)

$J = [g, h, \gamma, c, d, \delta, b, \beta, f, \phi]$ (By clauses D and I)

Notice J is equivalent to G.

Want to show (1) H is shorter than k and (2) I is shorter than k

(1) Want to show H is shorter than k

Two cases to consider (1a) G is of length k and (1b) G is of length $k-1$

(1a) G is of length k

In the following equations, let the presence of a term represent a count of one, and the presence of a set of terms represent the number of terms in that set. If multiple sets of terms are shown in parentheses, let this represent the number of terms found in both sets.

Since G is of length k we can define k as follows:

$k = b+c+d+f+g+h+\beta+\delta+\phi+\gamma-(\beta\delta)-(\beta\phi)-(\beta\gamma)-(\delta\phi)-(\delta\gamma)-(\phi\gamma)+(\beta\delta\phi)+(\beta\delta\gamma)+(\beta\phi\gamma)+(\delta\phi\gamma)-(\beta\delta\phi\gamma)$

Length of $H = b + i + f + j + \beta + \phi - (\beta\phi)$

Want to show the length of H is less than k:

$b + i + f + j + \beta + \phi - (\beta\phi) < b + c + d + f + g + h + \beta + \delta + \phi + \gamma - (\beta\delta) - (\beta\phi) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$

$\rightarrow i + j < c + d + g + h + \delta + \gamma - (\beta\delta) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$

Using a Venn Diagram or by other set intuition, the generic sets of terms on the R.H.S. of the inequality represent the following:

- terms just in δ
- terms just in γ
- terms in both δ and γ

The lowest possible value for this is 0 when the three aforementioned sets have no terms.

The inequality becomes:

$b + i + f + j < b + c + d + f + g + h$

$\rightarrow i + j < c + d + g + h$

Which requires at least three unique terms in c, d, g, and h.

Want to show at least three unique terms have to exist in c, d, g, and h.

Suppose not. Then at most two unique terms exist in c, d, g, and h.

Want to show the length of either E or F is the same as A, B, C, or D.

Consider all cases where there are fewer than three terms in c, d, g, h

Case 1, 0 terms exist in c, d, g, h

Recall the value for B

$B := [c, d, \delta, -i]$

But since 0 terms exist in c, d, g, h, this becomes

$B := [-i]$

Note that no terms may exist in δ because any terms in δ could be extracted to count as c or d.

Then a new derivation of E occurs:

$E := [a, b, \beta]$

Which is identical to A except for a missing i terminal.

Therefore the length of E is less than the length of A.

This is a contradiction since the length of E is given as k and the length of A is given as less than k.

Therefore this case is impossible.

Case 2, 1 term exists in c, d, g, h

It was shown that if 0 terms exist in c, d, there's a contradiction, so at least one term has to exist in c, d.

These terms are fixed, yet arbitrary so let's pick c to be the one term that exists.

B is now assigned:

$B := [c, -i]$

Again, no additional terms may exist in δ since they could be used as the terminal d.

E is again recalculated as:

$E := [a, b, \beta, c]$

By Lemma 5.10, the maximum length of E is (length of A + length of B - 2).

Since exactly one term exists in c and exactly one term exists in $-i$, the length of B is 2 and therefore the length of E is the same as the length of A.

However it was given that the length of E is k and the length of A is less than k.

This is a contradiction so this case cannot exist.

Case 3, 2 terms exist in c, d, g, h

As seen in Case 2, at least two terms must exist in c, d. This means no terms may exist in g, h.

The clause D can be rewritten:

$D := [-j]$

Note that γ cannot contain any terms because any terms in γ could be extracted and used as g or h

Since $-j$ represents one terminal, the length of D is 1.

Now the maximum length of F is (length of C + length of D - 2). Which means the maximum length of F is one less than the length of C.

However it was given that the length of F is k and the length of C is less than k.

This is a contradiction so this case cannot exist.

Since at least three terms must exist in c, d, g, h, the inequality holds and H is shorter than k.

(1b) G is of length $k - 1$

If G is of length $k - 1$, then k is defined as:

$k = b+c+d+f+g+h+\beta+\delta+\phi+\gamma-(\beta\delta)-(\beta\phi)-(\beta\gamma)-(\delta\phi)-(\delta\gamma)-(\phi\gamma)+(\beta\delta\phi)+(\beta\delta\gamma)+(\beta\phi\gamma)+(\delta\phi\gamma)-(\beta\delta\phi\gamma)+1$

Similarly to before, the inequality becomes:

$\rightarrow i + j < c + d + g + h + 1$

Which is true as long as at least two terms exist in c, d, g, h.

We already showed at least three terms must exist in c, d, g, h so the inequality is always true.

Therefore H is always shorter than k.

(2) Want to show I is shorter than k

Two cases to consider (2a) G is of length k and (2b) G is of length $k - 1$

(2a) G is of length k

Since G is of length k we can define k as follows:

$k = b+c+d+f+g+h+\beta+\delta+\phi+\gamma-(\beta\delta)-(\beta\phi)-(\beta\gamma)-(\delta\phi)-(\delta\gamma)-(\phi\gamma)+(\beta\delta\phi)+(\beta\delta\gamma)+(\beta\phi\gamma)+(\delta\phi\gamma)-(\beta\delta\phi\gamma)$

Length of $I = c + d + b + f + j + \delta + \beta + \phi - (\delta\beta) - (\delta\phi) - (\beta\phi) + (\delta\beta\phi)$

Want to show the length of I is less than k:

$c+d+b+f+j+\delta+\beta+\phi-(\delta\beta)-(\delta\phi)-(\beta\phi)+(\delta\beta\phi) < b+c+d+f+g+h+\beta+\delta+\phi+\gamma-(\beta\delta)-(\beta\phi)-(\beta\gamma)-(\delta\phi)-(\delta\gamma)-(\phi\gamma)+(\beta\delta\phi)+(\beta\delta\gamma)+(\beta\phi\gamma)+(\delta\phi\gamma)-(\beta\delta\phi\gamma)$

By Venn Diagram or other set intuition, the generic sets of terms on the L.H.S. represent the number of unique terms in δ, β, and ϕ. Similarly, the generic sets of terms on the R.H.S. represent the number of unique terms in δ, β, ϕ, and γ.

Subtacting the shared sets of terms from both sides, on the R.H.S. we are left with the terms in γ that are not present in $\delta, \beta, or \phi$. The lowest value for this is 0 when all terms in γ exist in δ, β, and ϕ.

The inequality therefore becomes:

$c+d+b+f+j < b+c+d+f+g+h$

$\rightarrow j < g+h$

Which is true as long as both g and h exist.

Want to show g and h exist.

Suppose not, then g and h don't exist.

Recall the value for D:

$D := [g, h, \gamma, -j]$

Since g and h don't exist, this becomes

$D := [-j]$

Note that no terms may exist in γ because any terms in γ could be extracted and counted as g or h.

Now we recalculate E:

$E := [-a, f, \phi]$

Which is smaller than C because j has to exist by the lemma's conditions.

However it was given that the length of E is k and the length of C is less than k.

This is a contradiction, therefore g and h must exist.

Therefore the inequality is always true and the length of I is less than k.

(2b) G is of length $k-1$

Similarly as before, we define k in terms of the length of G:

$k = b+c+d+f+g+h+\beta+\delta+\phi+\gamma-(\beta\delta)-(\beta\phi)-(\beta\gamma)-(\delta\phi)-(\delta\gamma)-(\phi\gamma)+(\beta\delta\phi)+(\beta\delta\gamma)+(\beta\phi\gamma)+(\delta\phi\gamma)-(\beta\delta\phi\gamma)+1$

Similarly to before, the inequality becomes

$\rightarrow j < g+h+1$

Which is true as long as g and h exist.

We already showed g and h exist so the inequality always holds true.

Therefore, the length of I is at most $k-1$.

Since we can derive J (which is equivalent to G) by processing A, B, C, D, H, and I, all of which whose lengths are shorter than k, we can derive G by only processing clauses whose lengths are less than k. □

Lemma 5.18. *Given the following:*

- *a clause, A, of length less than k*
- *a clause, B, of length less than k*
- *a clause, C, of length less than k*
- *a clause, D, of length k*
- *a clause E, of length k*
- *a clause F, of length $k-1$*

- *A and B imply D by Lemma 5.9*
- *C expands to E by Lemma 5.8*
- *D and E imply F by Lemma 5.9*

Then F can be implied by only processing clauses with a maximum length of $k - 1$.

Proof. Consider the following figure:

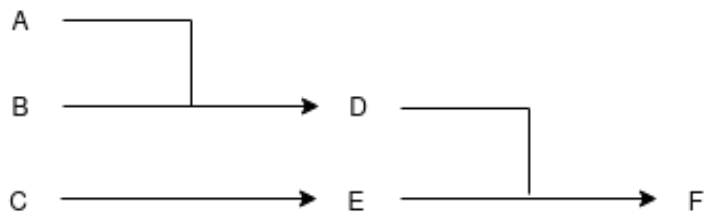

Fig. 6. An illustration of the derivations described by the lemma

Let the following clauses be defined:
$A := [a, b, \beta, i]$
$B := [c, d, \delta, -i]$
$C_1 := [-a, f, \phi]$
$C_2 := [e, f, \phi]$
Then the following clauses are implied:
$D = [a, b, \beta, c, d, \delta]$
$E_1 = [-a, f, \phi, g, h, \gamma]$
$E_2 = [e, f, \phi, -a, h, \gamma]$
$F_1 = [b, \beta, c, d, \delta, f, \phi, g, h, \gamma]$
$F_2 = [b, \beta, c, d, \delta, e, f, \phi, h, \gamma]$
Where β, δ, ϕ and γ are generic sets of terms such that A, B, and C block at least one clause by Lemma 5.3.

Notice E and D must share a term of the opposite form and there are two possible cases for what this term is: (1) it exists in C or (2) it exists in the additional terms in E, ie, it does not exist in C. Either way the opposite term must exist in A or B since it must exist in D and D is composed of terms from A or B. Since A and B are both generic clauses, it does not matter which clause the term exists in as long as it is fixed.

Consider (1) the opposite form term exists in C (use E_1 and F_1)
Want to show F_1 can be derived by processing clauses with a maximum length of k - 1.
Let the following clauses be defined:
$G = [b, \beta, i, f, \phi]$ (By clause A and C_1 with Lemma 5.9)
$H = [b, \beta, f, \phi, c, d, \delta]$ (By clause G and B with Lemma 5.9)
Since all of the terms in H exist in F_1, Lemma 5.8 can be used to derive F_1 from H.
Want to show (1a) G is shorter than k and (1b) H is shorter than k
(1a) G is shorter than k.

In the following equations, let the presence of a term represent a count of one, and the presence of a set of terms represent the number of terms in that set. If multiple sets of terms are shown in parentheses, let this represent the number of terms found in both sets.

Recall F is of length k - 1 so k can be defined as follows:

$k = b+c+d+f+g+h+\beta+\delta+\phi+\gamma-(\beta\delta)-(\beta\phi)-(\beta\gamma)-(\delta\phi)-(\delta\gamma)-(\phi\gamma)+(\beta\delta\phi)+(\beta\delta\gamma)+(\beta\phi\gamma)+(\delta\phi\gamma)-(\beta\delta\phi\gamma)+1$

Length of $G = b + i + f + \beta + \phi - (\beta\phi)$

Want to show length of G is less than k:

$b + i + f + \beta + \phi - (\beta\phi) < b + c + d + f + g + h + \beta + \delta + \phi + \gamma - (\beta\delta) - (\beta\phi) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma) + 1$

$\rightarrow b+i+f < b+c+d+f+g+h+\delta+\gamma-(\beta\delta)-(\beta\gamma)-(\delta\phi)-(\delta\gamma)-(\phi\gamma)+(\beta\delta\phi)+(\beta\delta\gamma)+(\beta\phi\gamma)+(\delta\phi\gamma)-(\beta\delta\phi\gamma)+1$

Using a Venn Diagram or by other set intuition, the generic sets on the L.H.S. of the inequality represent the unique terms in β and ϕ. Similarly, the generic sets of the R.H.S. represent the unique terms in β, δ, ϕ, and γ.

Subtracting the sets of generic terms on the L.H.S. from both sides yields the R.H.S. sets of generic terms as follows:

- the number of terms just in δ
- the number of terms just in γ
- the number of terms just in δ and γ

The lowest case for this is 0 if there are no terms in the aforementioned sets.

In such a case, the inequality becomes:

$b + i + f < b + c + d + f + g + h + 1$

$\rightarrow i < c + d + g + h + 1$

Which is true as long as at least one term exists in c, d, g, h

WTS at least one term has to exist in c, d, g, h.

Suppose not, then no terms from c, d, g, h exist.

Then some clauses are redefined:

$B := [\delta, -i]$

$E_1 := [-a, f, \phi, \gamma]$

Recall that the number of terms in just δ, γ, and $\delta \cap \gamma$ is 0, so any terms in these sets have to exist in β or ϕ.

Now in E_1 all of the terms in γ have to exist in ϕ or β. E_1 can be rewritten as:

$E_1 := [-a, f, \phi, \gamma']$

Where γ' is a subset of β

Similarly, D can be rewritten as

$D := [a, b, \beta, \delta']$

Where δ' is a subset of ϕ

Combining these with Lemma 5.9, we get a new clause

$I = [f, b, \phi, \beta, \gamma', \delta']$

But since γ' and δ' are subsets of β and ϕ respectively, these clauses only contain duplicates, so I may be rewritten as:

$I = [f, b, \phi, \beta]$

The length of I is clearly at most the length of F since all the terms in I exist in F. Therefore, the greatest length of I is $k - 1$.

Since γ' and δ' are subsets of β and ϕ, their lengths are at most the lengths of the latter sets.

The longest D is such that δ' is of the same length as ϕ, which is the length of I.

Recall the length of D is given as k.

D cannot have a length of k while also having a maximum length of $k-1$.

This is a contradiction, so at least one term has to exist in c, d, g, h so the length of G is indeed less than k.

(1b) Want to show H is shorter than k

Recall F is of length $k-1$, so we can define k as follows:

$k = b+c+d+f+g+h+\beta+\delta+\phi+\gamma-(\beta\delta)-(\beta\phi)-(\beta\gamma)-(\delta\phi)-(\delta\gamma)-(\phi\gamma)+(\beta\delta\phi)+(\beta\delta\gamma)+(\beta\phi\gamma)+(\delta\phi\gamma)-(\beta\delta\phi\gamma)+1$

Length of $H = b + f + c + d + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$

Want to show the length of H is less than k.

$b + f + c + d + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi) < b + c + d + f + g + h + \beta + \delta + \phi + \gamma - (\beta\delta) - (\beta\phi) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma) + 1$

$\rightarrow b + f + c + d < b + c + d + f + g + h + \gamma - (\beta\gamma) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma) + 1$

Using a Venn Diagram or by other set intuition, the sets left on the R.H.S. of the inequality represent the number of terms existing only in γ. This is lowest when all the terms in γ exist in another set, which would result in the generic sets on the R.H.S. summing to 0.

The inequality therefore becomes:

$\rightarrow b + f + c + d < b + c + d + f + g + h + \gamma - (\beta\gamma) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma) + 1$

$\rightarrow 0 < g + h + 1$

Which is always true.

Therefore H is shorter than k.

Since G and H are shorter than k, we can derive F by only processing clauses with a maximum length of $k-1$.

Consider (2) the opposite form term does not exist in C.

We can create two new clauses:

$J := [b, \beta, i, e, f, \phi, h, \gamma]$ (By clauses E_2 and A)

$K := [c, d, \delta, b, \beta, e, f, \phi, h, \gamma]$ (By clauses J and B)

Since all of the terms in K exist in F_2, Lemma 5.8 can be used to derive F_2 from K.

Want to show (2a) J is shorter than k and (2b) K is shorter than k

(2a) Want to show the length of J is less than k

We can define k in terms of F_2:

$k = b+c+d+e+f+h+\beta+\delta+\phi+\gamma-(\beta\delta)-(\beta\phi)-(\beta\gamma)-(\delta\phi)-(\delta\gamma)-(\phi\gamma)+(\beta\delta\phi)+(\beta\delta\gamma)+(\beta\phi\gamma)+(\delta\phi\gamma)-(\beta\delta\phi\gamma)+1$

Length of $J = b + i + e + f + H + \beta + \phi + \gamma - (\beta\phi) - (\beta\gamma) - (\phi\gamma) + (\beta\phi\gamma)$

Want to show the length of J is less than k

$b + i + e + f + H + \beta + \phi + \gamma - (\beta\phi) - (\beta\gamma) - (\phi\gamma) + (\beta\phi\gamma) < b + c + d + e + f + h + \beta + \delta + \phi + \gamma - (\beta\delta) - (\beta\phi) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma) + 1$

$\rightarrow i < c + d\delta - (\beta\delta) - (\delta\phi) - (\delta\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma) + 1$

Using a Venn Diagram or by other set intuition, the generic sets of terms on the R.H.S. represent the number of terms in δ that do not exist in any other set.

The lowest possible value for this is 0 and the inequality becomes:

$i < c + d + 1$

Which is true as long as one term exists in c, d.

Want to show at least one term exists in c, d.

Suppose not, then we can rewrite B as follows:

$B := [-i]$

Note that δ cannot have any terms because any terms from δ could be extracted to be treated as c or d.

Then D is recalculated:

$D = [a, b, \beta]$

We know D is of length k and A is of length k - 1.

Here it is clearly seen the length of A is one more than the length of D which is impossible since the length of A is k - 1 while the length of D is k.

This is a contradiction, therefore at least one term must exist in c, d.

Therefore the inequality holds and J is shorter than k.

(2b) K is shorter than k

Recall

$K := [c, d, \delta, b, \beta, e, f, \phi, h, \gamma]$ (By clauses J and B)

$F_2 = [b, \beta, c, d, \delta, e, f, \phi, h, \gamma]$

These two clauses are the same so K has a length of $k - 1$.

Since J and K are shorter than k and J and K can be used to derive F_2, we can derive F_2 by processing clauses with a maximum length of $k - 1$.

Since F_1 and F_2 are all possible cases of F and they can be derived without processing clauses longer than length $k - 1$, F can be derived by processing only clauses with a maximum length of $k - 1$. □

Lemma 5.19. *Given the following:*

- *a clause A, of length less than k*
- *a clause B, of length less than k*
- *a clause C, of length k*
- *a clause D, of length k*
- *a clause E, of length k - 1*
- *A expands to C by lemma 5.8*
- *B expands to D by lemma 5.8*
- *C and D imply E by lemma 5.7*

Then E can be derived by processing clauses whose length is at most k - 1.

Proof. Consider the following figure:

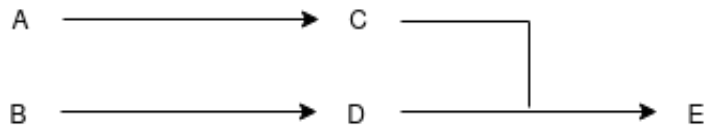

Fig. 7. An illustration of the derivations described in this lemma

Let the clauses be defined as follows:

$A := [a, b, \beta]$

$B := [c, d, \delta]$

Then the following clauses are derived:

$C = [a, b, \beta, e, f, \phi]$

$D = [c, d, \delta, g, h, \gamma]$

$E = [a, b, \beta, e, f, \phi, c, d, \delta, g, h, \gamma]$

Where β, δ, ϕ, and γ are fixed yet arbitrary set of terms in which the clauses block at least one assignment by Lemma 5.3.

In order for C and D to imply E by Lemma 5.7, they have to be identical except for one term which is positive in one clause and negated in the other.

There are 3 cases: (1) the opposite form term does not exist in A or B (2) the opposite form term exists in A or B (3) the opposite form term exists in A and B

Consider case (1) the opposite form term does not exist in A or B.

Then we fix one clause, say C, and redefine D in accordance with the requirements of Lemma 5.7. The clauses become:

$C := [a, b, \beta, e, f, \phi]$

$D := [a, b, \beta, -e, f, \phi]$

$E = [a, b, \beta, f, \phi]$

Notice that all of the terms in A exist in E. Therefore, we can expand A, one term at a time by lemma 5.8, to derive E. Since a single term is added in each step, and E is of length k - 1, the largest clause that needs to be processed is of length k - 2.

Consider case (2) the opposite form term exists in either A or B.

Again since C and D are both generic clauses derived in the same way from other generic clauses, we can fix either. Let's fix C and redefine D. The clauses become:

$C := [a, b, \beta, e, f, \phi]$

$D := [-a, b, \beta, e, f, \phi]$

$E = [b, \beta, e, f, \phi]$

We know $-a$ does not exist in B and B expands to D.

The terms in B are therefore a subset of the terms in D not counting $-a$.

More clearly $\{c, d, \delta\} \subset \{b, \beta, e, f, \phi\}$

Notice the set on the right is just all of the terms in E.

Therefore all of the terms in B exist in E and we can expand B to E by Lemma 5.8.

The greatest possible length of B is k - 1 and we know only one term gets appended in each step by Lemma 5.8 so we only have to process clauses with a maximum length of k - 1 to derive E.

Consider case (3) the opposite form term exists in A and B.

In this case, we redefine the clauses as follows:

$A := [a, b, \beta]$

$B := [-a, d, \delta]$

$C = [a, b, \beta, e, f, \phi]$

$D = [-a, d, \delta, g, h, \gamma]$

$E = [b, \beta, e, f, \phi, d, \delta, g, h, \gamma]$

In this case, by Lemma 5.7, A and B can imply a new clause:

$F = [b, \beta, d, \delta]$

whose terms are clearly a subset of the terms in E. As such, F can be used to derive E by Lemma 5.8. Since Lemma 5.8 only adds a single term with each step and E is of length k - 1 then the maximum length of F is k - 1 (since it's a subset of E) and you only have to process clauses with a maximum length of k - 1 to derive E.

Therefore if you have clauses, A, B, C, D, and E, as described, then E can be derived by processing clauses with a maximum length of k - 1. □

6 ALGORITHM

(1) For each clause in the instance, C, of length 3 or less:
  (a) For each clause in the instance, D, of length 3 or less:
    (i) Get all clauses implied by C and D according to Lemma 5.9 and add them to the instance
  (b) For each clause in the instance, E, of length 1:
    (i) For each clause in the instance, F, of length 1:
      (A) if E and F contain the same terminal in which it is positive in one clause and negated in the other, the clauses are contradicting and the instance is unsatisfiable, end
  (c) Expand clauses to get all possible clauses of length 3 and add them to the instance
(2) Repeat (1) until no new clauses are added
(3) If it reaches here, the instance is satisfiable, end

7 TIME COMPLEXITY ANALYSIS

In this section I will analyze the time complexity of the algorithm in section 4

(1) At most $\binom{n}{3} * 8 + \binom{n}{2} * 4 + \binom{n}{1} * 2$ clauses to iterate which is on the order of

$O(n^3)$

(1.a) At most $\binom{n}{3} * 8 + \binom{n}{2} * 4 + \binom{n}{1} * 2$ clauses to iterate which is on the order of

$O(n^3)$

(1.a.i) For each terminal in C, check if it's opposite form is in D

$O(3^2)$ which is just constant time which is on the order of

$O(1)$

(1.b) At most $\binom{n}{1} * 2$ clauses of length 1 exist, which is on the order of

$O(n)$

(1.b.i) At most $\binom{n}{1} * 2$ clauses of length 1 exist, which is on the order of

$O(n)$

(1.b.i.A) Constant time to check if two 1-terminal clauses contain the same terminal in the opposite form

$O(1)$

(1.c) For a 2-terminal clause, there are $2 * n$ possible 3-terminal clauses that could be expanded to. For a 1-terminal clause, there are $4 * n^2$ possible 3-terminal clauses that could be expanded to. This is upper bounded by the latter case.

$O(n^2)$

(2) Worst case, we add one new clause each time so we have to do (1) $\binom{n}{3} * 8 + \binom{n}{2} * 8 + \binom{n}{1} * 2$ times which is on the order of

$O(n^3)$

(3) Constant time to check and return satisfiable

$O(1)$

The time complexity breaks down:

$(2) * ((1) * ((1.a) * ((1.a.i)) + (1.b) * ((1.b.i) * (1.b.i.A)) + (1.c))) + (3)$

$= O(n^3) * (O(n^3) * (O(n^3) * (O(1)) + O(n) * (O(n) * O(1)) + O(n^2))) + O(1)$

$= O(n^3) * (O(n^3) * (O(n^3) * (O(1)) + O(n) * O(n) + O(n^2))) + O(1)$

$= O(n^3) * (O(n^3) * (O(n^3) * (O(1)) + O(n^2) + O(n^2))) + O(1)$

$= O(n^3) * (O(n^3) * (O(n^3) + O(n^2) + O(n^2))) + O(1)$

$= O(n^3) * (O(n^3) * (O(n^3) + O(n^2))) + O(1)$

$= O(n^3) * (O(n^3) * O(n^3)) + O(1)$

$= O(n^3) * O(n^6) + O(1)$

$= O(n^9) + O(1)$

$= O(n^9)$

8 PROOF OF CORRECTNESS

Want to show an instance is unsatifiable iff we can derive contradicting 1-terminal clauses by the algorithm.

WTS Contradicting 1-terminal clauses can be derived $\implies$ the instance is unsatifiable

Contradicting 1-terminal clauses take the form:

$[a]$

$[-a]$

Where a is a terminal in the problem.

In all possible assignments, a can have the value of True or False.

If a is True, the clause $[-a]$ will be False and the assignment does not satisfy the instance. If a is False, the clause $[a]$ will be False and the assignment does not satisfy the instance.

Since all possible assignments do not allow both clauses to be true, the instance is unsatifiable.

Want to show an unsatisfiable instance $\implies$ the algorithm will derive contradicting 1-terminal clauses

By Lemma 5.14, the given 3-terminal clauses can be expanded to every possible n-terminal clause.

By Lemma 5.15 these n-terminal clauses can be reduced by Lemma 5.7 to derive contradicting 1-terminal clauses.

So we know if we can derive these n-terminal clauses, we can derive contradicting 1-terminal clauses.

The way in which we derive these 1-terminal clauses is we use Lemma 5.7 to reduce the current set of k-terminal clauses to a set of $(k-1)$-terminal clauses. We repeat for all k from n to 2 inclusive.

Notice this passes through every possible k from length 2 to n.

Recall that deriving all of the n-terminal clauses was done by using Lemma 5.8.

These n-terminal clauses were then used to derive clauses of length $(n-1)$ by Lemma 5.7.

By Lemma 5.18, such a case allows us to derive the clauses of length $(n-1)$ without ever having to process a clause of length n.

Now we have all of the clauses of length $(n-1)$ that we would have derived if we processed clauses of length n.

Note how each of these clauses are either derived from given clauses by Lemma 5.8 or by Lemma 5.9 (all Lemma 5.7 derivations are a subset of all Lemma 5.9 derivations).

We now want to use these $(n-1)$-terminal clauses to derive clauses of length $(n-2)$, but we want to do it without processing clauses whose length is larger than $(n-2)$.

Since it takes two $(n-1)$-terminal clauses to derive a $(n-2)$-terminal clause by Lemma 5.7, the possible clauses could be of the form:

(1) both clauses were derived by Lemma 5.8 (expansion)
(2) both clauses were derived by Lemma 5.9 (reduction/implication)
(3) each clause was derived in a different manner

Note that this list is exhaustive because these are the only manner of implications used in the Lemmas that allowed us to derive these clauses.

We can use the following lemmas to handle each case:

(1) Lemma 5.19
(2) Lemma 5.17
(3) Lemma 5.18

As such we can derive the clauses of length $(n-2)$ without processing a clause whose length is greater than $(n-2)$.

In a more general sense, for every w from 1 to $n-4$, we know we have clauses of length $(n-w)$ that we want to use to derive clauses of length $(n-w-1)$ and since we derived the clauses of length $(n-w)$ only using Lemma 5.9 (which is a more general form of 5.7) and Lemma 5.8, we know all clauses of length $(n-w)$ have to be derived one of those two ways. We use two $(n-w)$-terminal clauses to derive a clause of length $(n-w-1)$ so the list of three ways these clauses could be derived is exhaustive. Since we have Lemmas 5.17, 5.18, and 5.19, all of these cases allow us to derive the clause of length $(n-w-1)$ without processing a clause longer than $(n-w-1)$. As such, we can derive all of the clauses we need down to clauses of length 4. Again all clauses of length 4 or greater were implied either by Lemma 5.8 (expansion) or Lemma 5.9 (reduction/implication).

At this point, we have 4-terminal clauses and we want to derive 3-terminal clauses. We need two clauses and we can either include a 3-terminal clause or we cannot.

If we don't include a 3-terminal clause, then the clauses used in the implication were themselves derived and Lemmas 5.17-5.19 can be used to imply the 3-terminal clause without processing clauses of length 3 or higher.

If we do use a 3-terminal clause, then the other clause has to be a 4-terminal clause because we are using 4-terminal clause(s) to derive a 3-terminal clause, but Lemma's 5.17-5.19 only cover cases where the implying clauses were themselves derived. Since we are given 3-terminal clauses, there are some clauses which are not derived. Now we want to show any 3-terminal clause implied by a 4-terminal clause can be derived by processing clauses of length no more than 3. The 4-terminal clause can either be derived by Lemma 5.9 or expanded to by Lemma 5.8. In the former case, we can use Lemma 5.11 and in the latter case we can use Lemma 5.12. In both cases, we can derive all the necessary 3-terminal clauses while processing clauses with a maximum length of 3.

Now we have all the 3-terminal clauses that would have been derived while the n-terminal clauses were being reduced to contradicting 1-terminal clauses.

We can now reduce the 3-terminal clauses to 1-terminal clauses.

Since we have all the necessary 3-terminal clauses without having to process a 4-terminal clause, this clearly shows we do not have to process any clauses of length 4 or greater to derive contradicting 1-terminal clauses.

Even though the algorithm only explicitly uses Lemma 5.9 and Lemma 5.8, this will cover cases where reduction is needed because Lemma 5.9 is a more general case of Lemma 5.7. Notice, too, that Lemmas 5.11, 5.12, 5.17, 5.18, and 5.19 rely on Lemmas 5.8 and 5.9 so we do not have to explicitly capture the cases where the former lemmas would apply.

Therefore, this coincides with the described algorithm.

9 CONCLUSION

In this paper, we present an algorithm to solve 3SAT in polynomial time.

This algorithm relies on Lemmas 5.8 and 5.9. The former of which says a clause can expand to imply another clause by adding on any term that's not in the original clause. The latter of which says two clauses sharing one terminal which is positive in one clause and negated in the other can imply a new clause composed of the rest of the terms in either clause. Using these strategies, we can process an instance of 3SAT while only considering clauses of length 3 or less are guaranteed to derive a pair of contradicting 1-terminal clauses if and only if the instance is unsatisfiable.

According to [2], 3SAT is NP-complete, every problem in NP can be represented as an instance of 3SAT and such a conversion can happen in polynomial time.

Since we have an algorithm to solve 3SAT in polynomial time, then all problems in NP can be solved in polynomial time.

Thus, P = NP.

REFERENCES

[1] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, page 151–158, New York, NY, USA, 1971. Association for Computing Machinery.

[2] Richard M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972.