A Polynomial Time Algorithm for 3SAT

Robert Quigley

Sections

01 Introduction

Introduce the problem and the approach

02 Algorithm Goals

Discuss what aspects of an algorithm would be valuable

03 The Algorithm

Explore the means to achieve the goals of the algorithm

04 Algorithm Proof

Prove the correctness of the algorithm assuming accurate lemmas

05 Lemma Proofs

Prove the correctness of the lemmas required by the algorithm

01

Introduction

An introduction to the problem
and the approach

The Boolean Satisfiability Problem

Given a group of variables (called terminals), x_1, x_2, x_3, ..., x_n,

all combined by logical AND operators, logical OR operators, with potential negations and repetitions, does there exist an assignment, True or False, for each terminal such that the instance evaluates to True?

If yes, call the instance satisfiable with a satisfying assignment

If no, call the instance unsatisfiable and no satisfying assignment exists

An instance may look like this:

$$x_i \vee (\neg x_j \vee x_k) \wedge (x_l \vee x_m) \vee x_n \ldots$$

This problem was shown to be NP-Complete by the Cook-Levin Theorem

The 3SAT Problem

A representation of the boolean satisfiability problem in which:

- Groups of exactly three terms are combined by logical OR operators, called clauses
- Each clause is combined by logical AND operators
- The same term may appear across clauses, but not within any given clause

An instance of 3SAT usually takes the form

$$(x_i \vee \neg x_j \vee x_k) \wedge (x_l \vee x_m \vee x_n) \ldots$$

It was shown in Karp's 21 NP-Complete Problems that 3SAT is also NP-Complete

Reformatting

The following changes are made for ease of use:

- **The symbol, x, will be removed**
- **The subscript will become the entire term**
- **Clauses will be represented as square brackets**
- **Logical AND operators will be removed**
- **Logical OR operators will be removed**
- **Negations will be written as minus signs**

For example,

$$(x_i \vee \neg x_j \vee x_k) \wedge (x_l \vee x_m \vee x_n)$$

Will be presented as

[i, -j, k], [l, m, n]

02

Algorithm Goals

Discuss what aspects of an algorithm would be valuable

Lemma 5.1

A clause can block an assignment

Since clauses are combined by logical AND operators, all clauses must be True for the instance to evaluate to True

If a single clause cannot evaluate to true given an assignment, then that assignment cannot satisfy the entire instance

Such an assignment occurs when all terms in a clause evaluate to False

For Example,

The clause [-1, 2, 3] blocks all assignments where

x_1 = True and x_2 = False and x_3 = False

Instance:
n: 3
clauses:
(2)
[-1, 2, 3],
[1, 2, 3],

Blocked	x_1	x_2	x_3
b: 1	0	0	0
b: 0	0	0	1
b: 0	0	1	0
b: 0	0	1	1
b: 1	1	0	0
b: 0	1	0	1
b: 0	1	1	0
b: 0	1	1	1

Figure 1. An example instance and assignment table

Note 1

If all assignments are blocked this implies an instance is unsatisfiable

Recall an instance is satisfiable iff there exists a satisfying assignment

A blocked assignment implies it cannot satisfy the instance

Therefore if all assignments are blocked, no assignments can satisfy the instance

Instance:
n: 3
clauses:
(8)
[-3, -2, -1],
[-3, -2, 1],
[-3, -1, 2],
[-3, 1, 2],
[-2, -1, 3],
[-2, 1, 3],
[-1, 2, 3],
[1, 2, 3],

Blocked	x_1	x_2	x_3
b: 1	0	0	0
b: 1	0	0	1
b: 1	0	1	0
b: 1	0	1	1
b: 1	1	0	0
b: 1	1	0	1
b: 1	1	1	0
b: 1	1	1	1

Figure 2. An unsatisfiable instance with all assignments blocked

Note 2

Contradicting 1-terminal clauses block all assignments

<u>def</u>: Two clauses are considered contradicting 1-terminal clauses if (1) each has a single term, (2) each have the same terminal, and (3) the terminal is positive in one clause and negated in the other

In other words, contradicting 1-terminal clauses imply the instance is unsatisfiable

Instance:
n: 3
clauses:
(2)
[-1],
[1],

Blocked	x_1	x_2	x_3
b: 1	0	0	0
b: 1	0	0	1
b: 1	0	1	0
b: 1	0	1	1
b: 1	1	0	0
b: 1	1	0	1
b: 1	1	1	0
b: 1	1	1	1

Figure 3. All assignments are blocked if two contradicting 1-terminal clauses exist

What We Know and Want to Show

- **<u>We Know</u>**
- An instance is unsatisfiable if all assignments are blocked
- All assignments are blocked if contradicting 1-terminal clauses exist
- Therefore an instance is unsatisfiable if contradicting 1-terminal clauses exist
- **<u>We Want</u>**
- A polytime way of determining whether or not all assignments are blocked
- A guaranteed way of deriving contradicting 1-terminal clauses iff the instance is unsatisfiable

- So we'll do just that

03

The Algorithm

Explore the means to achieve the goals of the algorithm

A Quick Definition

A clause or group of clauses, A and B, are said to imply another clause, C, if all of the assignments blocked by C are blocked by A and B

For example,

The clauses

A := [-3, 1, 2]

B := [1, 2, 3]

block all assignments that are blocked by

C := [1, 2]

Instance:
n: 3
clauses:
(2)
[-3, 1, 2],
[1, 2, 3],

Blocked	x_1	x_2	x_3
b: 1	0	0	0
b: 1	0	0	1
b: 0	0	1	0
b: 0	0	1	1
b: 0	1	0	0
b: 0	1	0	1
b: 0	1	1	0
b: 0	1	1	1

Instance:
n: 3
clauses:
(1)
[1, 2],

Blocked	x_1	x_2	x_3
b: 1	0	0	0
b: 1	0	0	1
b: 0	0	1	0
b: 0	0	1	1
b: 0	1	0	0
b: 0	1	0	1
b: 0	1	1	0
b: 0	1	1	1

Figure 4. The assignment tables for the described clauses

Algorithm Goal

Want to use the given clauses to imply contradicting 1-terminal clauses without processing clauses of length 4 or greater

Lemma 5.7 [Reduction]

Given the following:

- A clause, A
- A clause, B
- A clause, C
- A and B share all the same terms, except for one
- The unshared term is the same terminal which is positive in one clause and negated in the other
- C is composed of the remaining terms in either clause that are not that one terminal in either form

Then A and B imply C

The general structure is as follows:

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

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

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

Instance:
n: 3
clauses:
(3)
[-3, 1, 2],
[1, 2],
[1, 2, 3],

Blocked	x_1	x_2	x_3
b: 2	0	0	0
b: 2	0	0	1
b: 0	0	1	0
b: 0	0	1	1
b: 0	1	0	0
b: 0	1	0	1
b: 0	1	1	0
b: 0	1	1	1

Figure 5. The assignment table for the described clauses

Lemma 5.8 [Expansion]

Given the following

- A clause, C
- A terminal, t, that's not in C
- A clause, D, containing all of the terms in C appended to t
- A clause, E, containing all of the terms in C appended to -t

Then C implies D and E

The general structure is as follows:

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

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

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

Instance:
n: 3
clauses:
(3)
[-3, 1, 2],
[1, 2],
[1, 2, 3],

Blocked	x_1	x_2	x_3
b: 2	0	0	0
b: 2	0	0	1
b: 0	0	1	0
b: 0	0	1	1
b: 0	1	0	0
b: 0	1	0	1
b: 0	1	1	0
b: 0	1	1	1

Figure 6. The assignment table for the described clauses

Lemma 5.9 [General Lemma 5.7]

Given the following

- Two clauses sharing the same terminal which is positive in one clause and negated in the other
- A clause containing all of the terms from each clause except for the two opposite form terms

Then the first two clauses imply the third clause

The general structure is as follows:

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

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

$$\rightarrow C := [a, b, \ldots, c, d, \ldots]$$

Lemma 5.9 Example

Instance:
n: 5
clauses:
(3)
[-3, 4, 5],
[1, 2, 3],
[1, 2, 4, 5],

Blocked	x_1	x_2	x_3	x_4	x_5
b: 2	0	0	0	0	0
b: 1	0	0	0	0	1
b: 1	0	0	0	1	0
b: 1	0	0	0	1	1
b: 2	0	0	1	0	0
b: 0	0	0	1	0	1
b: 0	0	0	1	1	0
b: 0	0	0	1	1	1
b: 0	0	1	0	0	0
b: 0	0	1	0	0	1
b: 0	0	1	0	1	0
b: 0	0	1	0	1	1
b: 1	0	1	1	0	0

Blocked	x_1	x_2	x_3	x_4	x_5
b: 0	0	1	1	0	1
b: 0	0	1	1	1	0
b: 0	0	1	1	1	1
b: 0	1	0	0	0	0
b: 0	1	0	0	0	1
b: 0	1	0	0	1	0
b: 0	1	0	0	1	1
b: 1	1	0	1	0	0
b: 0	1	0	1	0	1
b: 0	1	0	1	1	0
b: 0	1	0	1	1	1
b: 0	1	1	0	0	0
b: 0	1	1	0	0	1
b: 0	1	1	0	1	0

Blocked	x_1	x_2	x_3	x_4	x_5
b: 0	1	1	0	1	1
b: 1	1	1	1	0	0
b: 0	1	1	1	0	1
b: 0	1	1	1	1	0
b: 0	1	1	1	1	1

Figure 7. The assignment tables demonstrating an example of Lemma 5.9

The Idea

- We can use these rules from Lemma 5.8 and Lemma 5.9 to keep implying new clauses until we either (1) run out of new clauses or (2) derive contradicting 1-terminal clauses
- (1) in this case, the instance is satisfiable
- (2) in this case, the instance is unsatisfiable

The Algorithm

$O(n^3)$	(1) For each clause in the instance, C, of length 3 or less:
$O(n^3)$	(a) For each clause in the instance, D, of length 3 or less:
$O(3^2)$	(i) Get all clauses implied by C and D according to Lemma 5.9 and add them to the instance
$O(n^3)$	(ii) Check if new clause is in the instance, update a flag to indicate a new addition
$O(n^2)$	(b) Use Lemma 5.8 to expand C to all possible clauses of length 3
$O(n^3)$	(2) For each clause in the instance, E, of length 1:
$O(n^3)$	(a) For each clause in the instance, F, of length 1:
$O(1)$	(i) 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
$O(n^3)$	(3) Repeat (1) - (2) until no new clauses are added
$O(1)$	(4) If it reaches here, the instance is satisfiable, end

Time Complexity: (3) * (1) * (1.a) * (1.a.ii) = $O(n^3) * O(n^3) * O(n^3) * O(n^3) = O(n^{12})$

04

Algorithm Proof

Prove the correctness of the algorithm assuming accurate lemmas

Overview

Want to show

an instance of 3SAT is unsatisfiable $\Leftrightarrow$ contradicting 1-terminal clauses can be implied by the algorithm

1. Contradicting 1-terminal clauses → the instance is unsatisfiable

2. The instance is unsatisfiable → contradicting 1-terminal clauses can be derived

Contra. 1-t clauses → Unsatisfiability

A 1-terminal clause blocks all assignments that make it evaluate to false

There are only two possible values for one terminal

Either one of the contradicting 1-terminal clauses block all assignments with either of the two possible values for the terminal

Therefore all assignments are blocked and no satisfying assignment can exist and the instance is unsatisfiable

Instance:
n: 3
clauses:
(2)
[-1],
[1],

Blocked	x_1	x_2	x_3
b: 1	0	0	0
b: 1	0	0	1
b: 1	0	1	0
b: 1	0	1	1
b: 1	1	0	0
b: 1	1	0	1
b: 1	1	1	0
b: 1	1	1	1

Figure 8. All assignments are blocked if two contradicting 1-terminal clauses exist

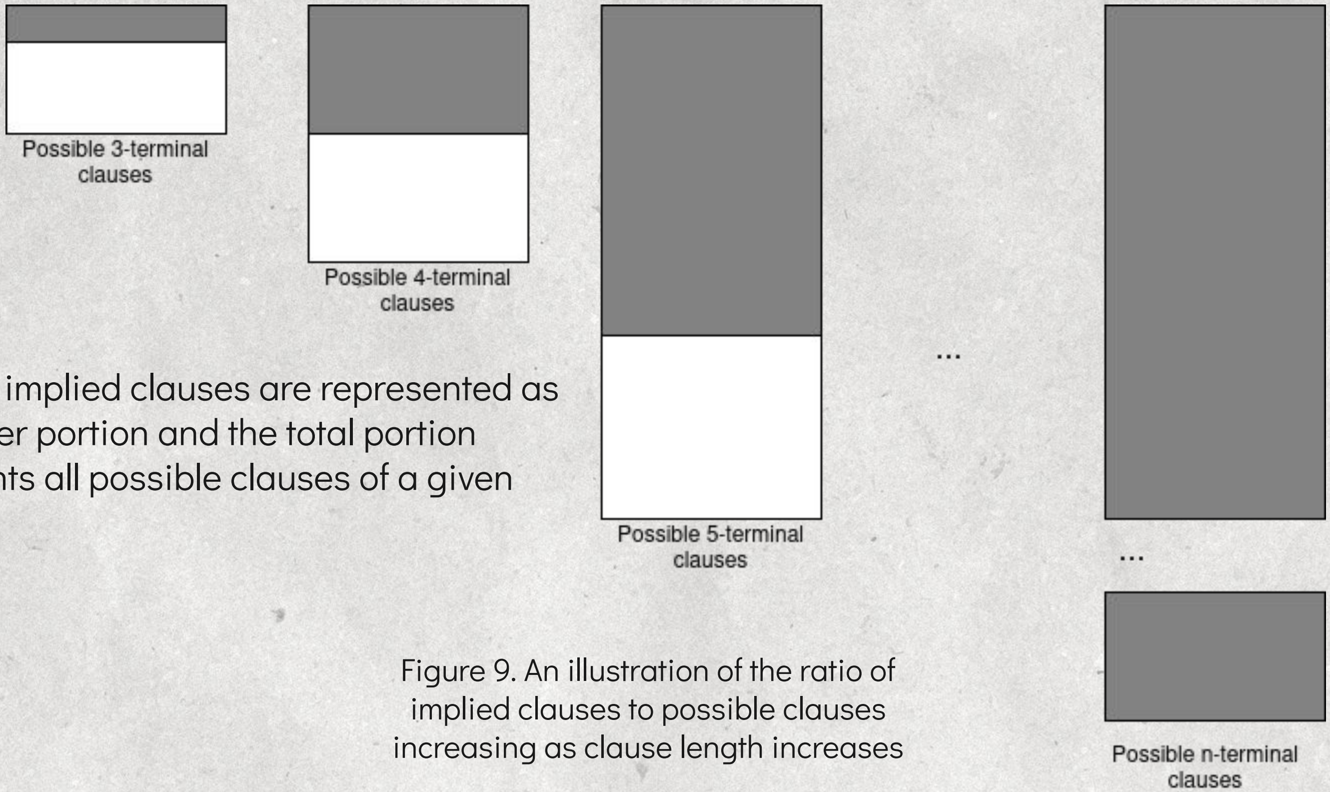
Unsatisfiability → Contra. 1-t Clauses

Oi jeez, buckle in for a ride

Idea:

- By Lemma 5.8 we can expand the given 3-terminal clauses to 2^n unique n-terminal clauses
- By Lemma 5.7 we can reduce these n-terminal clauses to contradicting 1-terminal clauses
- Want to show we never have to process clauses of length 4 or greater to imply these contradicting 1-terminal clauses
- Since the time complexity is bounded by the number of unique 3-terminal clauses, these contradicting 1-terminal clauses will be implied in polynomial time

Expanding Given Clauses



Given or implied clauses are represented as the darker portion and the total portion represents all possible clauses of a given length

Figure 9. An illustration of the ratio of implied clauses to possible clauses increasing as clause length increases

Reducing n-Terminal Clauses

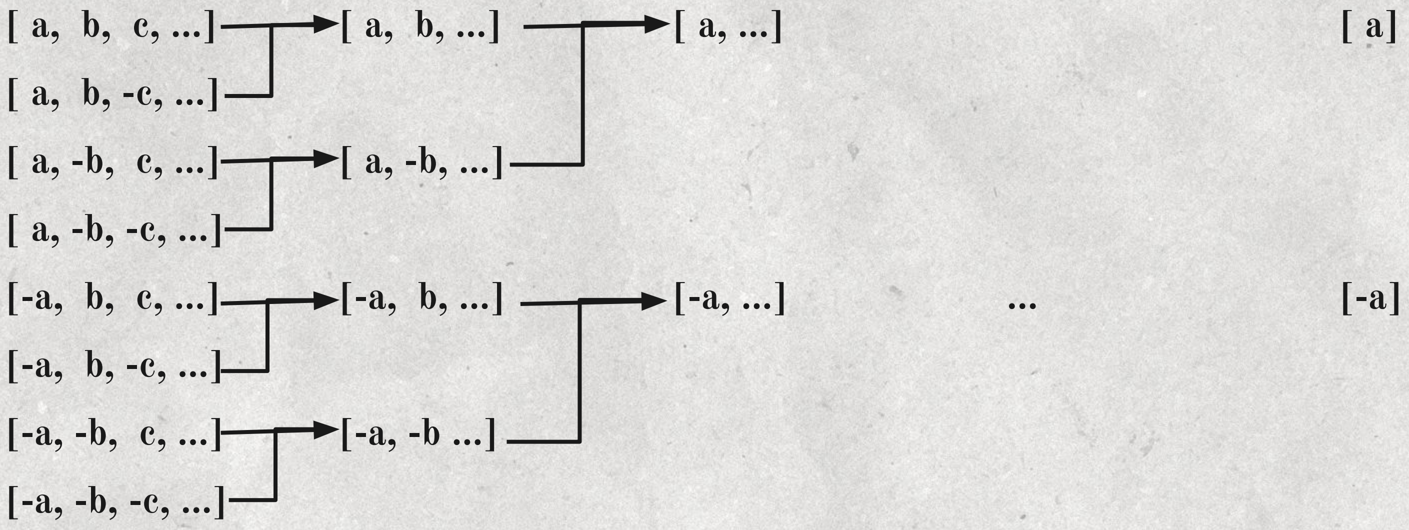


Figure 10. An illustration of reducing n-terminal clauses to contradicting 1-terminal clauses

Reducing n-Terminal Clauses

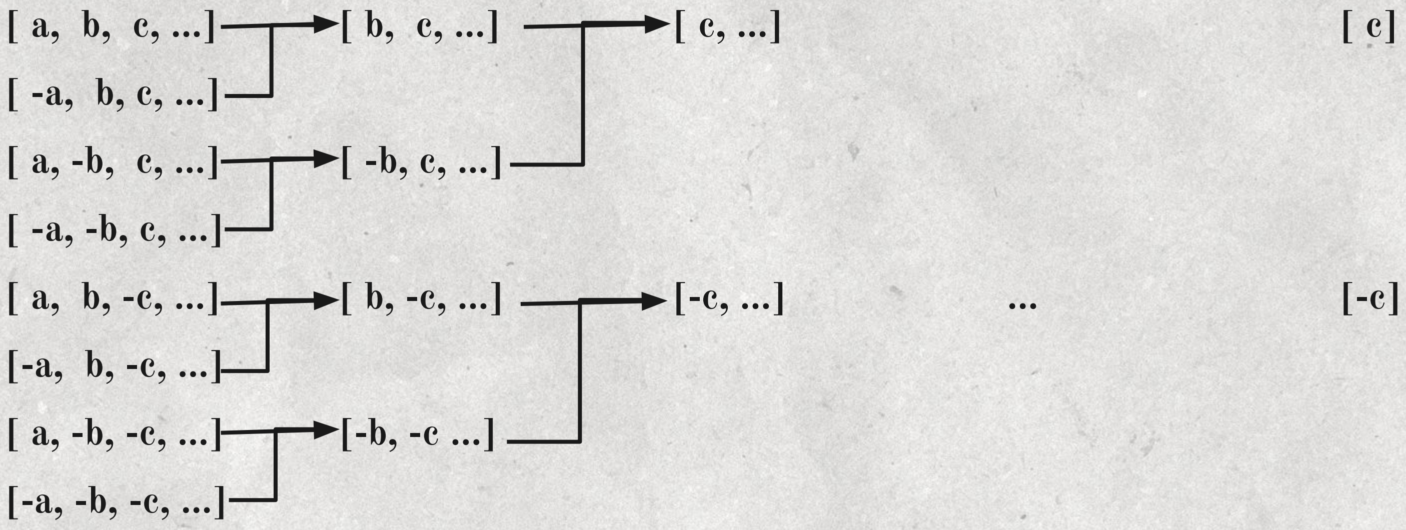


Figure 11. An alternative to figure 10
resulting in different 1-terminal clauses

Reducing n-Terminal Clauses

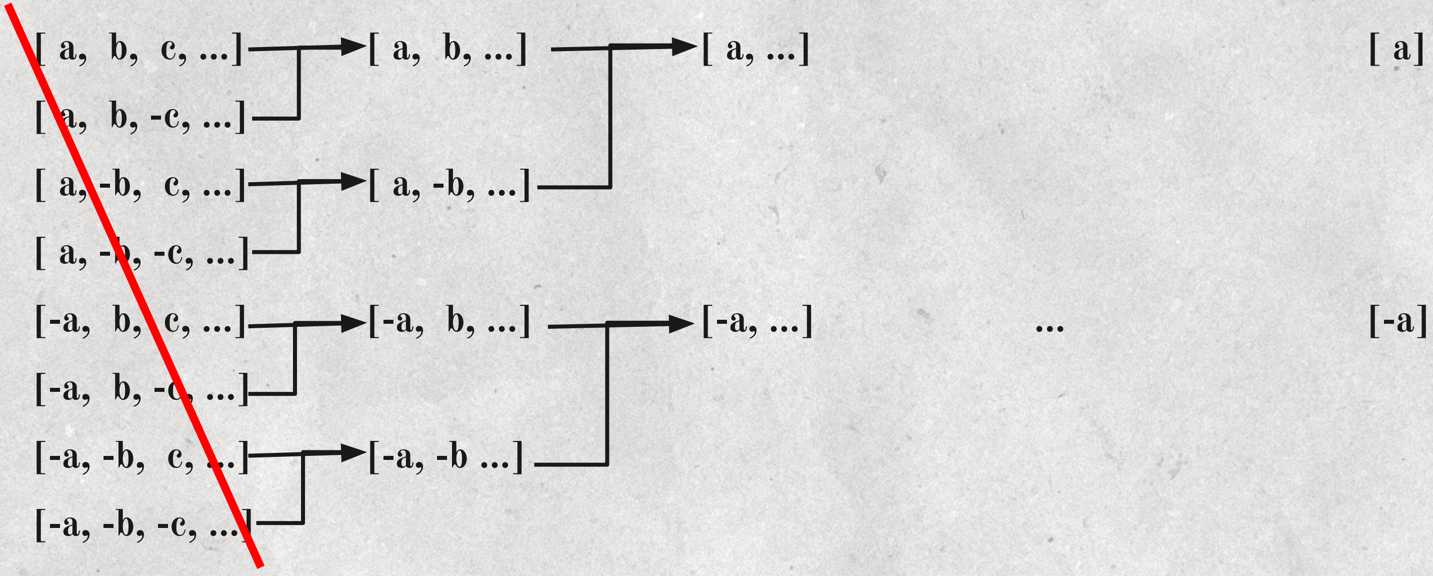


Figure 12. An illustration showing 1-terminal clauses can be derived from the (n-1)-terminal clauses

Reducing n-Terminal Clauses

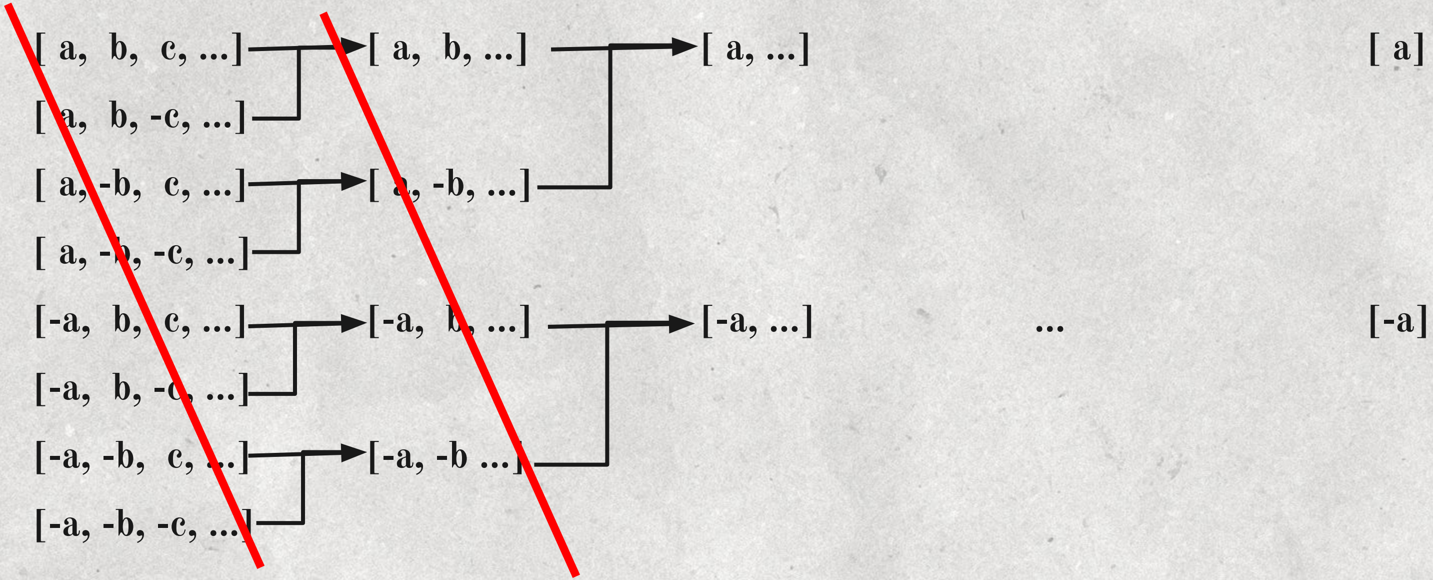


Figure 13. An illustration showing 1-terminal clauses can be derived from the (n-2)-terminal clauses

Goal: Reducing 3-Terminal Clauses

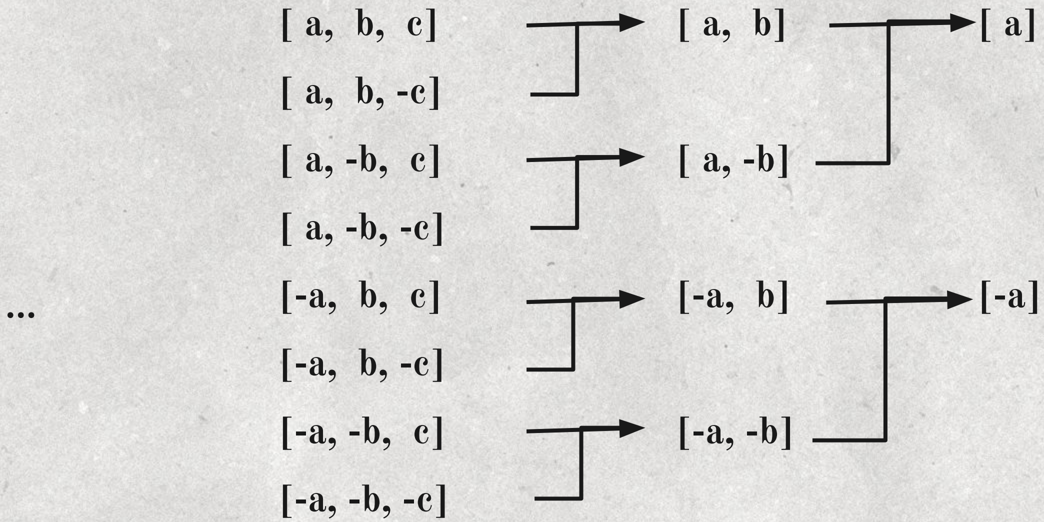


Figure 14. An illustration showing the reduction of 3-terminal clauses to 1-terminal clauses

Specific Claim

The n-terminal clauses do not have to be explicitly processed to derive the (n-1)-terminal clauses

Consider the following n-terminal clauses

C := [a, b, c, ...]
D := [a, -b, c, ...]

Which reduce to the following (n-1)-terminal clause:

E := [a, c, ...]

Where C is expanded to from the (n-1)-terminal clause

A := [b, c, ...]

Then want to show we can derive E without processing D

Small Proof for Specific Claim

Recall the clauses:

C := [a, b, c, ...] of length n
D := [a, -b, c, ...] of length n
E := [a, c, ...] of length n - 1
A := [b, c, ...] of length n - 1

Then want to show we can derive E without processing C

By Lemma 5.9, the clauses A and D can directly imply E

However, D is still of length n and the claim states E can be derived without processing n-terminal clauses

Use Lemma 5.19

We know all clauses of length n were derived by Lemma 5.8 [Expansion] so we have something like this

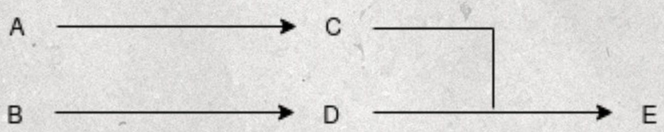


Figure 15. An illustration of the described clauses

Where A, B, C, D, and E are clauses with the following restrictions:

- A is of length n - 1
- B is of length n - 1
- C is of length n
- D is of length n
- E is of length n - 1 (or n for future reference)

Assume this is true for now, the proof is in the last section

Implication

Thanks to Lemma 5.19, we don't have to process any clauses of length n to derive all the clauses of length n-1 we need

In fact, we can generalize this and take it a step further

Claim

Given any two implied clauses of length k, say A and B, that imply a clause of length k or k - 1, say C, then we can directly derive C by processing clauses with a maximum length of k - 1 where $4 \leqslant k \leqslant n$ (with this restriction, k is guaranteed to be an implied clause)

This banks on the idea that any implied clauses can only exist because they were derived by existing/implied clauses and we can use those clauses to derive any clauses that would be implied

Supporting the Claim

After applying Lemma 5.19, we have all the needed clauses of length n - 1 and the next step is to further reduce these to clauses of length n - 2 using Lemma 5.7

Using Lemma 5.7 requires two clauses of length k and outputs one clause of length k - 1

Since all clauses of length n - 1 are implied, we know the n-1 clauses will fall into these cases:

- Both clauses were derived by Lemma 5.8 [Expansion]
- Both clauses were derived by Lemma 5.9 [Reduction/Implication]
- Each clause was derived in a different manner

Let's explore how we can handle each case

Both Clauses Derived by Lemma 5.8

This was already seen in Lemma 5.19

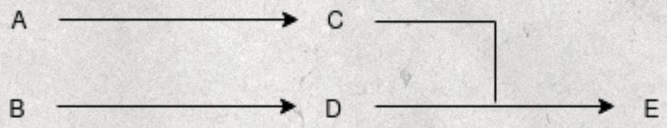


Figure 16. An illustration of the described clauses

Where the clauses are as follows

- A is of length k - 1
- B is of length k - 1
- C is of length k
- D is of length k
- E is of length k or k - 1

Then we can derive E by processing clauses with a maximum length of k - 1

Both Clauses Derived by Lemma 5.9

Use Lemma 5.17

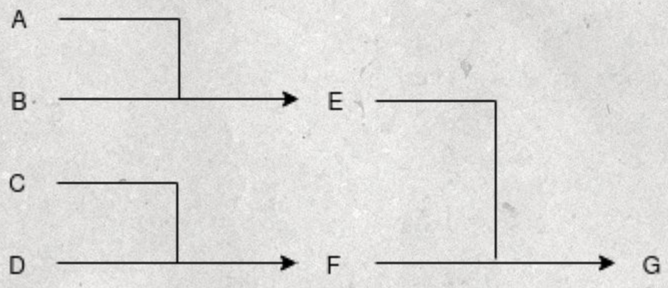


Figure 17. An illustration of the described clauses

Where the clauses are as follows

- A, B, C, D are of length k - 1
- E, F are of length k
- G is of length k or k - 1

Then we can derive G by processing clauses with a maximum length of k - 1

One Clause by 5.8 One Clause by 5.9

Use Lemma 5.18

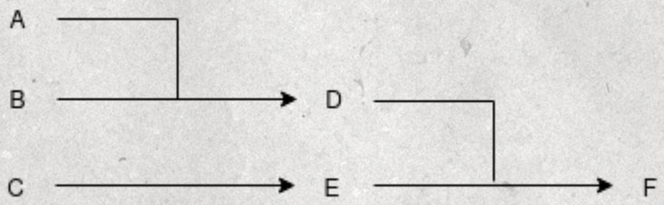


Figure 18. An illustration of the described clauses

Where the clauses are as follows:

- A, B, C are of length k - 1
- D, E are of length k
- F is of length k or k - 1

Then we can derive F by processing clauses with a maximum length of k - 1

Applications

Now we can use Lemmas 5.17, 5.18, and 5.19 to derive all the necessary clauses of length k by processing clauses with a maximum length of k - 1, assuming the input clauses are of length 4 or greater

As such we will be left with all the clauses of length 4 - that we need to imply contradicting 1-terminal clauses - by processing clauses of length 3 or less

Problem: We have all of the 4-terminal clauses we need and we want to imply all the 3-terminal clauses we need, but the aforementioned lemmas require the input clauses to be derived

Immediate Goals

We have all the 4-terminal clauses we need

We want to derive all the 3-terminal clauses we'll need using Lemma 5.7

3-terminal clauses can be derived in the following manner:

- Both input clauses are 4-terminal clauses
- One clause is of length 4 and the other clause is of length 3
- Both input clauses are of length 3 or less

We can disregard the final point because we want to show we can generate all 3-terminal clauses while processing only clauses of length 3 or less so the claim is vacuously true

Two 4-Terminal Clauses

In this case, both clauses are derived so we can use Lemma 5.17, Lemma 5.18, or Lemma 5.19

One 3-t and One 4-t Clause

We can no longer rely on the fact that both input clauses were constructed using other clauses

However, we do know the 4-terminal clause was derived using other clauses

Similarly as before, the 4-terminal clause can be derived one of two ways:

- Using Lemma 5.9 [Reduction/Implication] (which is a superset of 5.7)
- Using Lemma 5.8 [Expansion]

Using Lemma 5.9

In this case, the path of implications is as follows, use Lemma 5.11

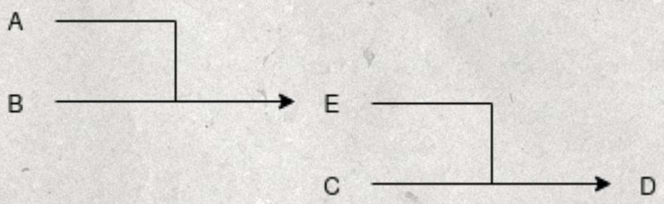


Figure 19. An illustration of the described clauses

Where the following is true:

- E is the 4-terminal clause
- C is the 3-terminal clause
- A and B are other 3-terminal clauses implying E
- D is a clause of length 3 or 4

Then D can be derived by processing clauses with a maximum length of 3

Using Lemma 5.8

In this case, the path of implications is as follows, use Lemma 5.12:

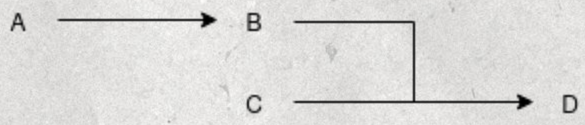


Figure 20. An illustration of the described clauses

Where the following is true:

- B is the 4-terminal clause
- C is the 3-terminal clause
- A is another 3-terminal clause expanding to B
- D is a clause of length 3 or 4

Then D can be derived by processing clauses with a maximum length of 3

Results

Now we can derive all the 3-terminal clauses - that we need to derive contradicting 1-terminal clauses - without processing a clause of length 4 or greater

We continue making all possible implications and because the instance is unsatisfiable (recall we're proving unsatisfiable → contradicting 1-terminal clauses will be derived) we will eventually derive contradicting 1-terminal clauses

Note that we only need the strategies in Lemma 5.8 and Lemma 5.9 because the set of implications in 5.9 is a superset of the implications in 5.7 and all of the intermediate lemmas (5.11, 5.12, 5.17, 5.18, and 5.19) rely only on Lemma 5.8 and Lemma 5.9

Concluding the IFF

Recall we want to show
an instance of 3SAT is unsatisfiable $\Leftrightarrow$ contradicting 1-terminal clauses can be implied by the algorithm

1. Contradicting 1-terminal clauses are derived $\rightarrow$ the instance is unsatisfiable
   - Shown true because contradicting 1-terminal clauses block all assignments

2. The instance is unsatisfiable $\rightarrow$ Contradicting 1-terminal clauses can be derived
   - Shown true because if an instance is unsatisfiable then all of the required 3-terminal clauses - that are needed to imply contradicting 1-terminal clauses - can be derived without processing a clause of length 4 or greater

05

Lemma Proofs

Prove the correctness of the lemmas required by the algorithm

Notation - Generic Sets of Clauses

In the following slides, greek letters are used to represent generic sets of terms

For example,

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

Where β represents any generic set of terms that allow for a valid clause according to Lemma 5.3 (doesn't contain the same terminal in both positive and negated forms)

However the same term or the same terminal in either form may appear between generic sets. We may have a clause like

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

Where a term in β may appear negated in δ. In this case, we usually just discard the clause as it bring no valuable information, but we still treat this case as a possibility in intermediate clauses.

Notation - Counting Clause Lengths

In the following slides, we use the presence of a term to indicate a length of 1, a greek letter to indicate the size of that set, and multiple greek letters to indicate intersection between sets

For example, the length of the following clause:

$$\mathrm{D} = [b, \beta, c, d, \delta, e, f, \phi]$$

Will be described as:

$$\text{Length of D} = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi)$$

Lemma 5.1

A clause can block an assignment

An assignment is considered <u>blocked</u> if it cannot possibly satisfy the instance

Since all clauses are combined by logical AND operators, all clauses must individually be true for the instance to evaluate to true

If there exists an assignment that forces a clause to evaluate to false, then that assignment cannot possibly satisfy the instance

Such an assignment exists if it makes every term in the instance false

For example, consider the clause

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

This blocks all assignments where

$$a = b = c = \text{False}$$

Lemma 5.2

For a given instance with n terminals, there are 2^n possible assignments

An assignment for this instance assigns values to n terminals

Each terminal has 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

Consider the clause

$$\mathbf{A} := [a, b, \ldots, -a]$$

Since any one assignment assigns exactly one value to each and every terminal, we know it assigns a value to *a*

There are two possible values for *a:*

$$a = \text{True}$$
$$a = \text{False}$$

In the first case, the positive form of the terminal will be true and the clause will be true
In the second case, the negated form of the terminal will be true and the clause will be true

Therefore all possible assignments make this clause true so this clause will block no assignments

Lemma 5.4

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

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

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

$$C := [a, b, c, \ldots]$$

And we have a terminal, *t*, that's not in *C*

For example, we have the clause

$$C := [1, 2]$$

And the terminal, x_3

Instance:

n: 3

clauses:

(1)

[1, 2],

Blocked	x_1	x_2	x_3
b: 1	0	0	0
b: 1	0	0	1
b: 0	0	1	0
b: 0	0	1	1
b: 0	1	0	0
b: 0	1	0	1
b: 0	1	1	0
b: 0	1	1	1

Figure 21. The assignment table of an example clause C and terminal t as described

Prove 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

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

$$C := [a, b, c, \ldots]$$

And we have a terminal, *t*, that's not in *C*

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*

Prove Lemma 5.5

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

Prove Lemma 5.5

We know 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

For example, say we have the clauses

$$C := [1, 2]$$
$$D := [1, 2, 3]$$

Then all of the assignments blocked by D are already blocked by C

Intuitively, you can think of it as a shorter clause has fixed values for some terminals and covers all cases for the remaining terminals, so when you add more terms to the clause, the blocked assignments get more specific and fewer assignments are blocked

Instance:
n: 3
clauses:
(2)
[1, 2],
[1, 2, 3],

Blocked	x_1	x_2	x_3
b: 2	0	0	0
b: 1	0	0	1
b: 0	0	1	0
b: 0	0	1	1
b: 0	1	0	0
b: 0	1	0	1
b: 0	1	1	0
b: 0	1	1	1

Figure 22. An example assignment table for the described clauses

Prove Lemma 5.6

Given the clauses, C and D as described:

$$C := [a, b, c, \ldots]$$
$$D := [a, b, c, \ldots, d, e, f, \ldots]$$

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

In other words, C blocks all assignments where

$$a = b = c = \ldots = \mathit{False}$$

Similarly, D blocks all assignments where

$$a = b = c = \ldots = d = e = f = \ldots = \mathit{False}$$

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:

- A clause, A
- A clause, B
- A clause, C
- A and B share all the same terms, except for one
- The unshared term is the same terminal which is positive in one clause and negated in the other
- C is composed of the remaining terms in either clause that are not that one terminal in either form

Then A and B imply C

The general structure is as follows:

A := [a, b, ..., i]
B := [a, b, ..., -i]
C := [a, b, ...]

Instance:
n: 3
clauses:
(3)
[-3, 1, 2],
[1, 2],
[1, 2, 3],

Blocked	x_1	x_2	x_3
b: 2	0	0	0
b: 2	0	0	1
b: 0	0	1	0
b: 0	0	1	1
b: 0	1	0	0
b: 0	1	0	1
b: 0	1	1	0
b: 0	1	1	1

Figure 23. An example assignment table for the described clauses

Prove Lemma 5.7 [Reduction]

Recall the described clauses:

A := [a, b, ..., i]
B := [a, b, ..., -i]
C := [a, b, ...]

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 the following

- A clause, C
- A terminal, t, that's not in C
- A clause, D, containing all of the terms in C appended to t
- A clause, E, containing all of the terms in C appended to -t

Then C implies D and E

The general structure is as follows:

C := [a, b, ...]
D := [a, b, ..., t]
E := [a, b, ..., -t]

Instance:
n: 3
clauses:
(3)
[-3, 1, 2],
[1, 2],
[1, 2, 3],

Blocked	x_1	x_2	x_3
b: 2	0	0	0
b: 2	0	0	1
b: 0	0	1	0
b: 0	0	1	1
b: 0	1	0	0
b: 0	1	0	1
b: 0	1	1	0
b: 0	1	1	1

Figure 24. An example assignment table for the described clauses

Prove Lemma 5.8 [Expansion]

Recall the described clauses:

C := [a, b, ...]
D := [a, b, ..., t]
E := [a, b, ..., -t]

Since *t* is not in C then by Lemma 5.6 all assignments blocked by D and E are blocked by C

In other words, we can say C implies D and E

Lemma 5.9 [General Lemma 5.7]

Given the following

- Two clauses sharing the same terminal which is positive in one clause and negated in the other, say A and B
- A clause containing all of the terms from each clause except for the two opposite form terms, say C

Then A and B imply C

The general structure is as follows:

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

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

$$\rightarrow C := [a, b, \ldots, c, d, \ldots]$$

Prove Lemma 5.9

Consider two clauses,

$$C := [a, b, c, ..., t]$$
$$D := [d, e, f, ..., -t]$$

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:

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

Prove Lemma 5.9

Recall the clauses

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

By Lemma 5.6, we know that C implies E'

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

Prove Lemma 5.9

Recall the clauses

$$C := [a, b, c, \ldots, t]$$
$$D := [d, e, f, \ldots, \neg t]$$
$$E := [a, b, c, \ldots, d, e, f, \ldots]$$

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.

Second, 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

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 the following clauses:

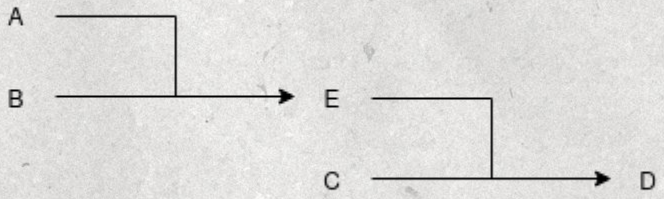


Figure 25. An illustration of the described clauses

Where the following is true:

- A, B, and C are clauses of length less than k
- E is a clause of length k
- D is a clause of length k or k - 1
- A and B imply E by Lemma 5.9
- E and C imply D by Lemma 5.9

Then D can be derived by processing only clauses of length k - 1 or less

Prove Lemma 5.11

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 E or by F and B)

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

Note that C must contain a negated term from E and all of the terms in E come from A or B (not counting i or -i) and A and B are logically equivalent so we pick one and fix it and say C contains a negated term from clause A.

Idea: We use an intermediate clause, F, to derive D and we want to show F is shorter than k

Prove Lemma 5.11

We have two cases to consider:

- D is of length k
- D is of length k - 1

Prove 5.11 (D is of length k)

Recall the clause D

$$D = [b, \beta, c, d, \delta, e, f, \phi]$$

In this case, we can define k as follows:

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

And we can define the length of F as follows:

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

Want to show length of F is less than k

Prove 5.11 (D is of length k)

Want to show length of F $<$ k

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

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

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

Note that i has to exist based on the lemma requirements

The inequality is true as long as i exists and at least two terms exist on the R.H.S. so WTS two terms must exist on the R.H.S.

Prove 5.11 (D is of length k)

Want to show

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

Using a Venn Diagram or by other set intuition, the sets of generic terms on the R.H.S. represent the number of terms in δ that exist in no other set. The lowest value for this is 0 so the inequality becomes:

$$i < c + d$$

Want to show c and d have to exist

Prove 5.11 (D is of length k)

WTS c and d have to exist. Suppose not, then at maximum either c or d exist and we can redefine some clauses:

$$A := [a, b, \beta, i]$$
$$B := [\delta, \neg i]$$
$$D := [a, b, \beta, \delta]$$

Notice the maximum size of δ is 1 because if two terms existed in that set, we could extract them and use them as c and d, but we know both c and d do not exist.

Since δ is of size 1, and i is just a single term, we can clearly see that the length of D is the same as the length of A. This is a contradiction because the lemma states A is of length less than k and D is a clause of length k.

Therefore both c and d have to exist → the inequality holds → F is shorter than k and we can derive D by processing clauses with a maximum length of k - 1

Prove 5.11 (D is of length k - 1)

Now we consider the case where D is of length k - 1

We define k in terms of D and want to show F is shorter than k

$$k = b + c + d + e + f + \beta + \delta + \phi - (\beta\delta) - (\beta\phi) - (\delta\phi) + (\beta\delta\phi) + 1$$
$$\text{length of } F = b + i + e + f + \beta + \phi - (\beta\phi)$$

WTS length of F is less than k

Similarly to before, this becomes

$$i < c + d + 1$$

Which is true as long as c or d exist

Want to show c or d have to exist

Prove 5.11 (D is of length k - 1)

Want to show c or d have to exist. Suppose not, then c and d do not exist and we can redefine some of the clauses:

$$A := [a, b, \boldsymbol{\beta}, i]$$
$$\boldsymbol{B} := [\boldsymbol{-i}]$$
$$D := [a, b, \boldsymbol{\beta}]$$

Notice that no terms may exist in δ because if it contains at least one term then that term can be extracted and treated as c or d but we know neither c nor d not exist

Since i is exactly one term, we can see the length of D is one less than the length of A. This is a contradiction because the length of A is given as less than k (so it has a maximum value of k - 1) and the length of D is given as k - 1 so it cannot be less than the length of A.

Therefore c or d must exist, the inequality holds, F is shorter than k and we can derive D by processing clauses with a maximum length of k - 1

Lemma 5.11 Conclusion

Given the aforementioned clauses, we showed we can derive D by processing clauses with a maximum length k - 1 by using a new clase, F, which is derived using A and C by Lemma 5.9

This is done without loss of generality because we know C must contain a term of the opposite form of a term in E and all terms in E come from A or B so we pick an arbitrary yet fixed clause, A, to have the term of the opposite form

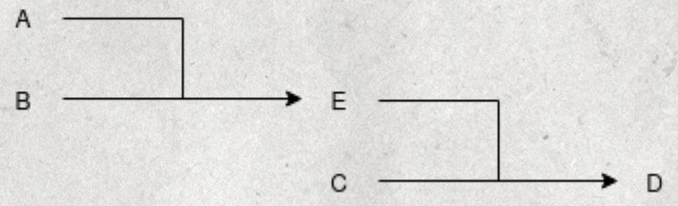


Figure 27. An illustration of the described clauses

Lemma 5.12

Consider the following clauses

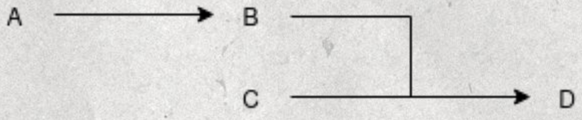


Figure 28. An illustration of the described clauses

- A and C are clauses of length less than k
- B is a clause of length k
- D is a clause of length k or k - 1

Then D can be derived by processing clauses with a maximum length of k - 1

Prove Lemma 5.12

Let's define the following clauses:

$$A := [a, b, \beta]$$
$$B := [a, b, \beta, c, d, \delta]$$
$$C_1 := [\text{-a, e, f}, \phi]$$
$$C_2 := [\text{-c, e, f}, \phi]$$
$$D_1 := [\text{b}, \beta, \text{c, d}, \delta, \text{e, f}, \phi]$$
$$D_2 := [\text{a, b}, \beta, \text{d}, \delta, \text{e, f}, \phi]$$
$$\text{E} := [\text{b}, \beta, \text{e, f}, \phi]$$

Where all of the following are true:

- A expands to B by Lemma 5.8
- B and C_1 imply D_1 by Lemma 5.9
- B and C_2 imply D_2 by Lemma 5.9
- C_1 or C_2 are used based on different possible cases described in the next slide

Prove Lemma 5.12

Given the following clauses

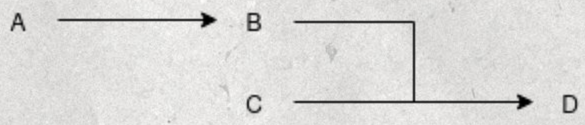


Figure 29. An illustration of the described clauses

Notice B and C have to share a term of the opposite form in order to imply D

Two cases for this term:

- The term exists in A
- The term does not exist in A

Prove Lemma 5.12

Consider case 1, C and A share a term of the opposite form

We can define the clauses as follows:

Given

$$A := [a, b, \beta]$$
$$B := [a, b, \beta, c, d, \delta]$$
$$C_1 := [-a, \mathrm{e}, \mathrm{f}, \phi]$$

Implied

$D_1 := [\mathrm{b}, \beta, \mathrm{c}, \mathrm{d}, \delta, \mathrm{e}, \mathrm{f}, \phi]$ (By A and B or by expanding E)

$E := [\mathrm{b}, \beta, \mathrm{e}, \mathrm{f}, \phi]$ (By A and C_1 using Lemma 5.9)

Since C_1 and A share a term of the opposite form, we can imply a new clause, E, as shown

Notice all of the terms in E exist in D_1 so we can use Lemma 5.8 to expand E to D_1

We derived D_1 using only clauses A, C_1, E, and any clauses between E and D_1

Prove Lemma 5.12

Want to show all clauses used to derive D_1 are shorter than k

As seen in Lemma 5.8, expanding a clause only outputs clauses of length 1 greater than the input, so if we have E and we know it's shorter than k, we will never have to process a clause of length k or greater to derive D

Note A and C_1 are given to be shorter than k in the lemma conditions

Now we just want to show E is shorter than k

We have two cases:

- D is of length k
- D is of length k - 1

Prove Lemma 5.12

Consider case (1a) D is of length k, then we can define k in terms of D and we can define the length of E as follows:

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

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

Want to show length of E < k

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

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

Which is true as long as one term exists on the R.H.S.

Prove Lemma 5.12

Want to show at least one term must exist in

$$c + d + \delta - (\beta\delta) - (\delta\phi) + (\beta\delta\phi)$$

Suppose not, then none of those terms exist and we can redefine some clauses:

Recall the clauses used to be defined as

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

But with the new requirements, the clause B gets redefined:

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

Notice A is exactly B. This is a contradiction because the length of A was given to be less than k and the length of B was given to be k

Therefore at least one term must exist on the R.H.S. of the inequality and the inequality holds

E is shorter than k and we can derive D without processing a clause of length k or greater

Prove Lemma 5.12 Checklist

Want to show we can derive D by processing clauses with a maximum length of k - 1:

- ❏ **E contains an opposite form term in A**
  - ✓ D is of length k
  - ❏ D is of length k - 1
- ❏ **E does not contain an opposite form term in A**

Prove Lemma 5.12

Consider (1b) where D is of length k - 1

Similarly as before we define k in terms of D and want to show E is shorter than k:

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

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

The only difference now is k is one greater so the inequality becomes

$$\rightarrow 0 < c + d + \delta - (\beta\delta) - (\delta\phi) + (\beta\delta\phi) + 1$$

Which is clearly always true since the lowest value for the sets of generic terms is 0

Therefore E is shorter than k when D is of length k - 1

Prove Lemma 5.12 Checklist

Want to show we can derive D by processing clauses with a maximum length of k - 1:

- ✓ **E contains an opposite form term in A**
  - ✓ D is of length k
  - ✓ D is of length k - 1
- ❏ **E does not contain an opposite form term in A**

Prove Lemma 5.12

Consider the case where E does not contain an opposite form term from A

In this case we can define the following clauses:

$$A := [a, b, \beta]$$
$$B := [a, b, \beta, c, d, \delta]$$
$$C_2 := [\text{-c, e, f}, \phi]$$
$$D_2 := [\text{a, b}, \beta, \text{d}, \delta, \text{e, f}, \phi]$$

Where all of the following are true:

- A expands to B by Lemma 5.8
- B and C_2 imply D_2 by Lemma 5.9

Notice all terms in A exist in D_2 so we can directly imply D_2 using A and since A was given to be shorter than k, we can derive D_2 by processing only clauses of length k - 1 or less

Prove Lemma 5.12 Checklist

Want to show we can derive D by processing clauses with a maximum length of k - 1:

- ✓ **E contains an opposite form term in A**
  - ✓ D is of length k
  - ✓ D is of length k - 1
- ✓ **E does not contain an opposite form term in A**

Therefore in all cases we can derive D by processing only clauses with a maximum length of 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.

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 if all of the terms in *A* exist in *B*

For example,

A := [a, b, c]
B := [a, b, c, ...]

Since all the terms in A exist in B, then any and every term in the instance can exist in the rest of B and all assignments blocked by B will be blocked by 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

Want to show an unsatisfiable instance ⇒ 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 by the given 3-terminal clauses

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

Since all 2^n assignments are blocked and each assignment is blocked by a unique n-terminal clause, then 2^n n-terminal clauses can be implied by the given 3-terminal clauses

Prove Lemma 5.14

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

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

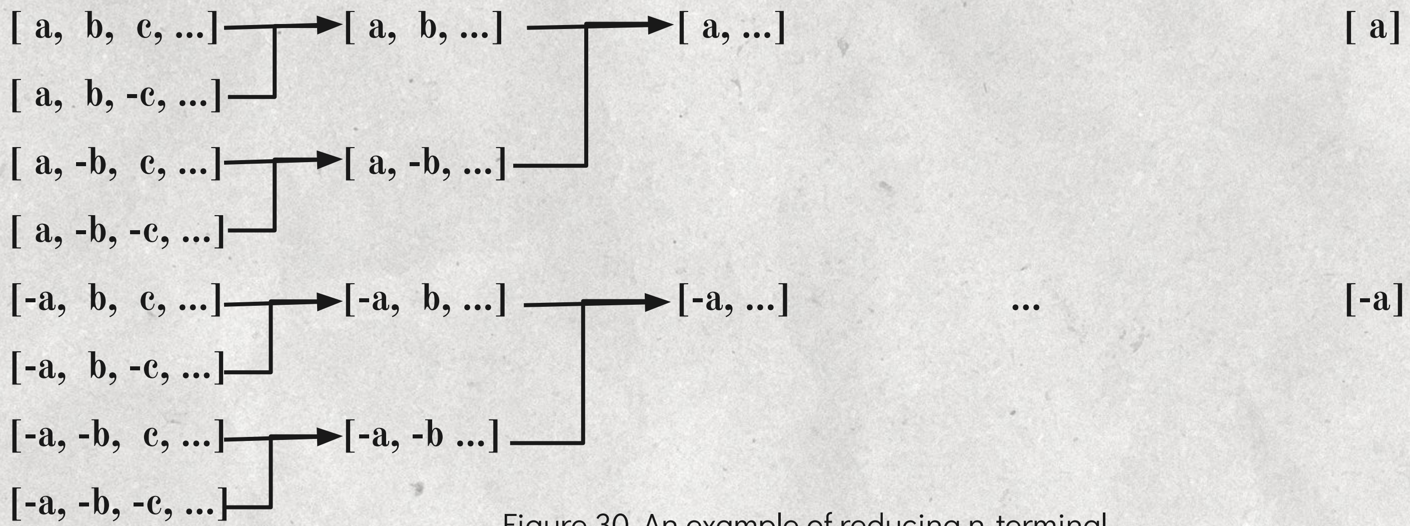


Figure 30. An example of reducing n-terminal clauses to 1-terminal clauses

Prove Lemma 5.15

Algorithm:

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

Notice 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

Let the following clauses be defined:

A := [a]
B := [-a]
C := [b, c, d, ...]
D := [a, b]
E := [-a, c, d, ...]

By lemma 5.8, we can expand to any clause that contains *a* or -*a*

Now want to show we can imply C which is a clause that that does not contain *a* or -*a*

By Lemma 5.8, we know A implies D, B implies E, and Lemma 5.9 can be used to show D and E imply C

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

Lemma 5.17

Given the following clauses

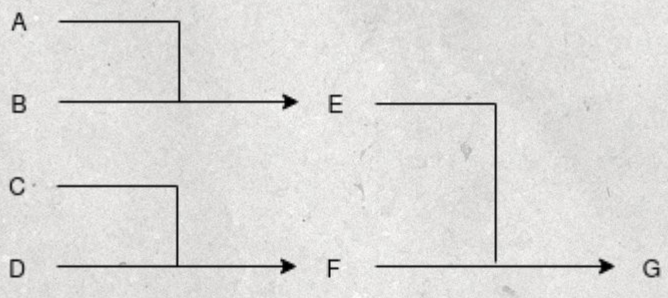


Figure 31. An illustration of the described clauses

Where the clauses are as follows

- **A, B, C, D are of length less than k**
- **E, F are of length k**
- **G is of length k or k - 1**
- **A and B imply E by Lemma 5.9**
- **C and D imply F by Lemma 5.9**
- **E and F imply G by Lemma 5.9**

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

Prove Lemma 5.17

Let the clauses be defined as follows:

$$A := [a, b, \beta, i]$$
$$B := [c, d, \delta, -i]$$
$$C := [-a, f, \phi, j]$$
$$D := [g, \hbar, \gamma, -j]$$

Then the following are implied by Lemma 5.9:

$$E = [a, b, \beta, c, d, \delta]$$
$$F = [-a, f, \phi, g, \hbar, \gamma]$$
$$G = [b, \beta, c, d, \delta, f, \phi, g, \hbar, \gamma]$$

Note than all terms in E come from A and B and all terms in F come from C and D

So the opposite form term in E and F have to exist in (A or B) and (C or D)

Since these are generic clauses, we can say the positive form exists in A and the negated term exists in D

Prove Lemma 5.17

Let's imply some additional clauses based on what we know about the opposite form terms
Recall the given clauses:

$$A := [a, b, \beta, i]$$
$$B := [c, d, \delta, -i]$$
$$C := [-a, f, \phi, j]$$
$$D := [g, \hbar, \gamma, -j]$$
$$E := [a, b, \beta, c, d, \delta]$$
$$F := [-a, f, \phi, g, \hbar, \gamma]$$
$$G := [b, \beta, c, d, \delta, f, \phi, g, \hbar, \gamma]$$

Then we define some intermediate clauses:

$H = [b, \beta, i, f, \phi, j]$ (By clauses A and C using Lemma 5.9)
$I = [c, d, \delta, b, \beta, f, \phi, j]$ (By clauses B and H using Lemma 5.9)
$J = [g, \hbar, \gamma, c, d, \delta, b, \beta, f, \phi]$ (By clauses D and I using Lemma 5.9)

Notice J is exactly G and we only used clauses A, B, C, D, H, and I to derive J so want to show all these clauses are shorter than k

Prove Lemma 5.17

Want to show A, B, C, D, H, and I are shorter than k

Recall A, B, C, and D were given as shorter than k so just want to show H and I are shorter than k

For each of these claims, we have two cases

- G is of length k
- G is of length k - 1

Prove Lemma 5.17 Checklist

- ❑ **H is shorter than k**
  - ❑ G is of length k
  - ❑ G is of length k - 1
- ❑ **I is shorter than k**
  - ❑ G is of length k
  - ❑ G is of length k - 1

Prove Lemma 5.17

Want to show H is shorter than k when G is of length k

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

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

And the length of H is

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

Want to show the length of H is less than k

$$b + i + f + j + \beta + \phi - (\beta\phi)$$

$$<$$

$$b + c + d + f + g + \hbar + \beta + \delta + \phi + \gamma - (\beta\delta) - (\beta\phi) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$$

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

Prove Lemma 5.17

Want to show the inequality holds:

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

Which is true if at least three terms exist on the R.H.S.

Consider the following cases:

- None of the terms on the R.H.S. exist
- Exactly One of the terms on the R.H.S. exists
- Exactly Two of the terms on the R.H.S. exist

Prove Lemma 5.17

Consider case 1, where no terms exist in

$$c + d + g + h + \delta + \gamma - (\beta\delta) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$$

Recall some clauses

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

But since no terms exist in {c, d}, 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:

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

Notice the length of E is one less than the length of A. This is a contradiction because the length of E was given as k while the length of A was given as less than k. This case is impossible.

Prove Lemma 5.17

Consider case 2, where exactly one terms exists in

$$c + d + g + h + \delta + \gamma - (\beta\delta) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$$

Recall some clauses

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

It was already shown to be impossible if no terms exist in {c, d} so the single existing term must be in {c, d}.

Since c and d are generic terms, let's pick c to be the term that exists. We redefine B:

$$B := [\mathrm{c}, -i]$$

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:

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

Prove Lemma 5.17

Now we have the clauses

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

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

Notice the length of E is the same as the length of A. This is a contradiction because the length of E is given as k while the length of A is given as less than k. Therefore this case is impossible and at least two terms must exist in the R.H.S. of the inequality

Prove Lemma 5.17

Consider case 3 where exactly two terms exist in

$$c + d + g + \hbar + \delta + \gamma - (\beta\delta) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$$

As seen in case 2, at least two terms must exist in {c, d} so none of the remaining terms may exist.

Recall the clauses

$$C := [-a, f, \phi, j]$$
$$D := [g, \hbar, \gamma, -j]$$
$$F := [-a, f, \phi, g, \hbar, \gamma]$$

Which get redefined based on the new requirements:

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

It can be seen that the length of F is less than the length of C. This is a contradiction because the length of C is given as less than k and and the length of F is given as k.

Prove Lemma 5.17

Since the cases where zero, one, and two terms exist on the R.H.S. are impossible, we know at least three terms must exist on the R.H.S.

In such a case, the inequality is always true

Therefore H is shorter than k when the length of G is k

Prove Lemma 5.17 Checklist

- ❑ **H is shorter than k**
  - ✓ G is of length k
    - ✓ No terms exist on the R.H.S. of the inequality
    - ✓ Exactly one term exists on the R.H.S. of the inequality
    - ✓ Exactly two terms exist on the R.H.S. of the inequality
  - ❑ G is of length k - 1
- ❑ **I is shorter than k**
  - ❑ G is of length k
  - ❑ G is of length k - 1

Prove Lemma 5.17

Want to show H is shorter than k when G is of length k - 1

In this case we define k and the length of H similarly to before:

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

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

Similarly to before, the inequality will become:

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

Which is true as long as at least two terms exist on the R.H.S.

It was already shown at least three terms must exist and that proof did not rely on the length of G so the inequality is true here as well

The length of H is less than k when the length of G is k - 1

Prove Lemma 5.17 Checklist

- ✓ **H is shorter than k**
  - ✓ G is of length k
    - ✓ No terms exist on the R.H.S. of the inequality
    - ✓ Exactly one term exists on the R.H.S. of the inequality
    - ✓ Exactly two terms exist on the R.H.S. of the inequality
  - ✓ G is of length k - 1
- ❏ **I is shorter than k**
  - ❏ G is of length k
  - ❏ G is of length k - 1

Prove Lemma 5.17

Want to show I is shorter than k when G is of length k

We define k in terms of G and define the length of I as follows:

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

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

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

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

$$<$$

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

Prove Lemma 5.17

The inequality becomes

$$j < g + \hbar + \gamma - (\beta\gamma) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$$

Which is true as long as at least two terms exist on the R.H.S.

Prove Lemma 5.17

Want to show at least two terms exist in

$$g + \hbar + \gamma - (\beta\gamma) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$$

Suppose not, then a maximum of one term exists in the set
Recall some clauses

$$C := [-a, f, \phi, j]$$
$$D := [g, \hbar, \gamma, -j]$$
$$F := [-a, f, \phi, g, \hbar, \gamma]$$

Which get redefined

$$D := [\mathrm{x}, -j]$$
$$F := [-a, f, \phi, \mathrm{x}]$$

Where x is at most a single term.

It is seen the length of F is at most the length of C. This is a contradiction because the length of F was given as k and the length of C was given as less than k.

Therefore at least two terms must exist on the R.H.S. and the inequality is true

Prove Lemma 5.17 Checklist

- ✓ **H is shorter than k**
  - ✓ G is of length k
    - ✓ No terms exist on the R.H.S. of the inequality
    - ✓ Exactly one term exists on the R.H.S. of the inequality
    - ✓ Exactly two terms exist on the R.H.S. of the inequality
  - ✓ G is of length k - 1
- ❏ **I is shorter than k**
  - ✓ G is of length k
  - ❏ G is of length k - 1

Prove Lemma 5.17

Want to show I is shorter than k when G is of length k - 1

We define k in terms of G and define the length of I as follows:

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

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

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

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

$$<$$

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

Prove Lemma 5.17

The inequality becomes

$$j < g + h + \gamma - (\beta\gamma) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma) + 1$$

Which is true as long as at least one terms exists on the R.H.S.

It was already shown that at least two terms must exist on the R.H.S. and the proof did not rely on the length of G so the inequality is still true

Therefore the length of I is less than k when G is of length k - 1

Prove Lemma 5.17 Checklist

- ✓ **H is shorter than k**
  - ✓ G is of length k
    - ✓ No terms exist on the R.H.S. of the inequality
    - ✓ Exactly one term exists on the R.H.S. of the inequality
    - ✓ Exactly two terms exist on the R.H.S. of the inequality
  - ✓ G is of length k - 1
- ✓ **I is shorter than k**
  - ✓ G is of length k
  - ✓ G is of length k - 1

Since G can be derived by processing only A, B, C, D, H, and I and all of those clauses are shorter than k, we can derive D by processing clauses with a maximum length of k - 1

Lemma 5.18

Given the following clauses

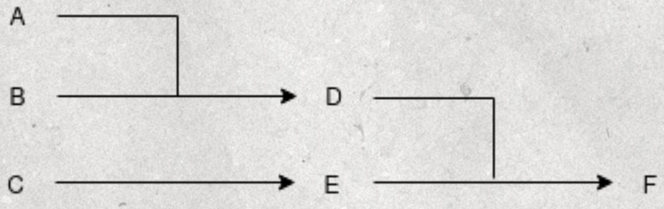


Figure 32. An illustration of the described clauses

Where the following are true

- A, B, C are of length less than k
- D, E are of length k
- F is of length k or k - 1

Then F can be derived by processing clauses with a maximum length of k - 1

Prove Lemma 5.18

Let's define some clauses

$$A := [a, b, \beta, i]$$
$$B := [c, d, \delta, -i]$$
$$C_1 := [-a, f, \phi]$$
$$C_2 := [e, f, \phi]$$
$$D := [a, b, \beta, c, d, \delta]$$
$$E_1 := [-a, f, \phi, g, \hbar, \gamma]$$
$$E_2 := [e, f, \phi, -a, \hbar, \gamma]$$
$$F_1 := [b, \beta, c, d, \delta, f, \phi, g, \hbar, \gamma]$$
$$F_2 := [b, \beta, c, d, \delta, e, f, \phi, \hbar, \gamma]$$

Where the following is true:

- A and B imply D by Lemma 5.9
- C_1 expands to E_1 by Lemma 5.8
- C_2 expands to E_2 by Lemma 5.8
- D and E_1 imply F_1 by Lemma 5.9
- D and E_2 imply F_2 by Lemma 5.9
- Where C has two possible cases: C_1 or C_2 as described in the following slides

Prove Lemma 5.18

Recall the implications

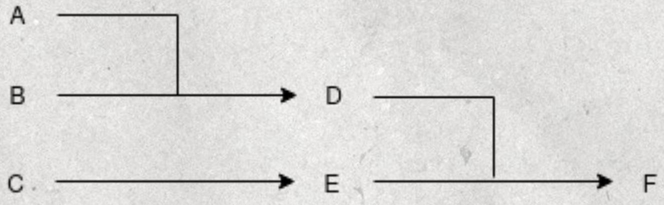


Figure 33. An illustration of the described clauses

Since E and D imply F then E and D must share a term of the opposite form
Since A and B imply D then all terms in D come from A or B

So we have two cases for this opposite form term in E:

- **It exists in C (use C_1)**
- **It does not exist in C (use C_2)**

And for each of these cases we have to consider the two cases of F:

- **F is of length k**
- **F is of length k - 1**

Lemma 5.18 Checklist

Want to show we can derive F by processing clauses with a maximum length of k - 1

- ❏ **The opposite form term in E exists in C (use C_1)**
  - ❏ F is of length k
  - ❏ F is of length k - 1
- ❏ **The opposite form term in E does not exist in C (use C_2)**
  - ❏ F is of length k
  - ❏ F is of length k - 1

Prove Lemma 5.18

Consider case 1 using C_1 and F_1

Then we define some additional clauses:

$$G = [b, \beta, i, f, \phi] \text{ (By clause A and } C_1 \text{ with Lemma 5.9)}$$
$$H = [b, \beta, f, \phi, c, d, \delta] \text{ (By clause G and B with Lemma 5.9)}$$

Recall F_1

$$F_1 := [b, \beta, c, d, \delta, f, \phi, g, \hbar, \gamma]$$

Since all of the terms in H exist in F_1, we can expand H using Lemma 5.8 to derive F_1

We are able to derive F_1 using clauses A, C_1, B, G, and H

Want to show all of these are shorter than k

Prove Lemma 5.18

Want to show A, C_1, B, G, and H are shorter than k

It was given that A, B, and C are all shorter than k so just want to show G and H are shorter than k

Let's update the checklist

Lemma 5.18 Checklist

Want to show we can derive F by processing clauses with a maximum length of k - 1

- ❑ **The opposite form term in E exists in C (use C_1)**
  - ❑ G is shorter than k
    - ❑ F is of length k
    - ❑ F is of length k - 1
  - ❑ H is shorter than k
    - ❑ F is of length k
    - ❑ F is of length k - 1
- ❑ **The opposite form term in E does not exist in C (use C_2)**
  - ❑ F is of length k
  - ❑ F is of length k - 1

Prove Lemma 5.18

Want to show G is shorter than k when using F_1 and F_1 is of length k

Let's define k in terms of F_1 and the length of G as follows:

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

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

Want to show length of G $<$ k

$$b + i + f + \beta + \phi - (\beta\phi)$$

$$<$$

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

Prove Lemma 5.18

The inequality becomes

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

Which is true as long as at least two terms exist on the R.H.S.

Want to show at least two terms exist on the R.H.S.

Prove Lemma 5.18

Want to show at least two terms exist in

$$c + d + g + \hbar + \delta + \gamma - (\beta\delta) - (\beta\gamma) - (\delta\phi) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\phi) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$$

Suppose not, then at most one term exists on the R.H.S.
Recall the clauses

$$A := [a, b, \beta, i]$$
$$D := [a, b, \beta, c, d, \delta]$$

We can redefine some clauses since at most one of the aforementioned clauses exist

$$D := [a, b, \beta, \mathrm{x}]$$

Where x is at most one term

Notice the length of D is the same as the length of A. This is a contradiction because it was given that the length of A is less than k and the length of D is k.

Therefore at least two terms must exist on the R.H.S., the inequality is true, and G is shorter than k when using F_1 and the length of F_1 is k

Lemma 5.18 Checklist

Want to show we can derive F by processing clauses with a maximum length of k - 1

- ❑ **The opposite form term in E exists in C (use C_1)**
  - ❑ G is shorter than k
    - ✓ F is of length k
    - ❑ F is of length k - 1
  - ❑ H is shorter than k
    - ❑ F is of length k
    - ❑ F is of length k - 1
- ❑ **The opposite form term in E does not exist in C (use C_2)**
  - ❑ F is of length k
  - ❑ F is of length k - 1

Prove Lemma 5.18

Want to show G is shorter than k when using F_1 and the length of F_1 is k - 1

Let's define k in terms of F_1 and the length of G as follows:

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

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

Want to show length of G $<$ k

$$b + i + f + \beta + \phi - (\beta\phi)$$

$$<$$

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

Prove Lemma 5.18

The inequality becomes

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

Which is true as long as at least one term exists on the R.H.S.

We already showed at least two terms must exist on the R.H.S. and the proof did not rely on the length of F_1 so the inequality is still always true

Therefore the length of G is less than k when using F_1 and F_1 is of length k - 1

Lemma 5.18 Checklist

Want to show we can derive F by processing clauses with a maximum length of k - 1

- ❑ **The opposite form term in E exists in C (use C_1)**
  - ✓ G is shorter than k
    - ✓ F is of length k
    - ✓ F is of length k - 1
  - ❑ H is shorter than k
    - ❑ F is of length k
    - ❑ F is of length k - 1
- ❑ **The opposite form term in E does not exist in C (use C_2)**
  - ❑ F is of length k
  - ❑ F is of length k - 1

Prove Lemma 5.18

Want to show H is shorter than k when using F_1 and F_1 is of length k

Let's define k in terms of F_1 and the length of H as follows:

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

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

Want to show length of H $<$ k

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

$$<$$

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

Prove Lemma 5.18

The inequality becomes

$$\rightarrow 0 < g + \hbar + \gamma - (\beta\gamma) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$$

Which is true as long as at least one term exists on the R.H.S.

Want to show at least one term exists on the R.H.S.

Prove Lemma 5.18

Want to show at least one term exists in

$$g + h + \gamma - (\beta\gamma) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma)$$

Suppose not, then none of the aforementioned terms exist.
Recall the clauses

$$C_1 := [-a, f, \phi]$$
$$E_1 := [-a, f, \phi, g, h, \gamma]$$

Since we know some terms don't exist, we can redefine some clauses:

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

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

Notice E_1 and C_1 are exactly the same. This is a contradiction because the length of E was given as k and the length of C was given as less than k.

Therefore at least one term exists on the R.H.S. and the inequality is true.

Therefore H is shorter than k when using F_1 and F_1 is of length k

Lemma 5.18 Checklist

Want to show we can derive F by processing clauses with a maximum length of k - 1

- ❑ **The opposite form term in E exists in C (use C_1)**
  - ✓ G is shorter than k
    - ✓ F is of length k
    - ✓ F is of length k - 1
  - ❑ H is shorter than k
    - ✓ F is of length k
    - ❑ F is of length k - 1
- ❑ **The opposite form term in E does not exist in C (use C_2)**
  - ❑ F is of length k
  - ❑ F is of length k - 1

Prove Lemma 5.18

Want to show H is shorter than k when using F_1 and F_1 is of length k - 1

Let's define k in terms of F_1 and the length of H as follows:

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

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

Want to show length of H $<$ k

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

$$<$$

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

Prove Lemma 5.18

The inequality becomes

$$\rightarrow 0 < g + h + \gamma - (\beta\gamma) - (\delta\gamma) - (\phi\gamma) + (\beta\delta\gamma) + (\beta\phi\gamma) + (\delta\phi\gamma) - (\beta\delta\phi\gamma) + 1$$

Which is always true

Therefore H is shorter than k when using F_1 and F_1 is of length k - 1

Lemma 5.18 Checklist

Want to show we can derive F by processing clauses with a maximum length of k - 1

- ✓ **The opposite form term in E exists in C (use C_1)**
  - ✓ G is shorter than k
    - ✓ F is of length k
    - ✓ F is of length k - 1
  - ✓ H is shorter than k
    - ✓ F is of length k
    - ✓ F is of length k - 1
- ❑ **The opposite form term in E does not exist in C (use C_2)**
  - ❑ F is of length k
  - ❑ F is of length k - 1

Prove Lemma 5.18

Consider the case where the opposite form term from E does not exist in C (use C_2)

Recall the implication graph:

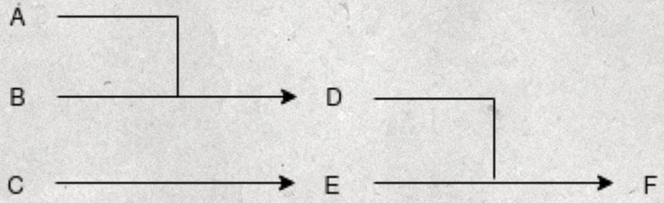


Figure 34. An illustration of the described clauses

Prove Lemma 5.18

Recall the clauses

$$A := [a, b, \beta, i]$$
$$B := [c, d, \delta, -i]$$
$$C_2 := [e, f, \phi]$$
$$D := [a, b, \beta, c, d, \delta]$$
$$E_2 := [e, f, \phi, -a, \hbar, \gamma]$$
$$F_2 := [b, \beta, c, d, \delta, e, f, \phi, \hbar, \gamma]$$

Here it is seen that all of the terms in C_2 exist in F_2 so we can simply expand C_2 and derive F_2 using Lemma 5.8

Since it is given that C_2 is of length less than k, we can derive F_2 by processing clauses with a maximum length of less than k

Lemma 5.18 Checklist

Want to show we can derive F by processing clauses with a maximum length of k - 1

- ✓ **The opposite form term in E exists in C (use C_1)**
  - ✓ G is shorter than k
    - ✓ F is of length k
    - ✓ F is of length k - 1
  - ✓ H is shorter than k
    - ✓ F is of length k
    - ✓ F is of length k - 1
- ✓ **The opposite form term in E does not exist in C (use C_2)**
  - ✓ F is of length k
  - ✓ F is of length k - 1

Lemma 5.19

Consider the clauses and implications

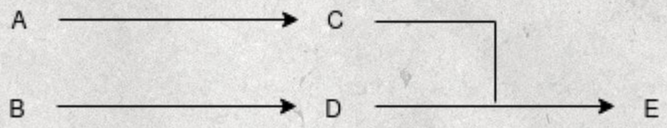


Figure 35. An illustration of the described clauses

Where the following is true

- A and B are clauses of length less than k
- C and D are clauses of length k
- E is of length k or 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 5.9

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

Prove Lemma 5.19

Let's define some clauses

$$A := [a, b, \beta]$$
$$B := [c, d, \delta]$$
$$C := [a, b, \beta, e, f, \phi]$$
$$D := [c, d, \delta, g, \hbar, \gamma]$$
$$E = [a, b, \beta, e, f, \phi, c, d, \delta, g, \hbar, \gamma]$$

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 possible cases:

- the opposite form term does not exist in A or B
- the opposite form term exists in A or B
- the opposite form term exists in A and B

Lemma 5.19 Checklist

Want to derive E by processing clauses with a maximum length of k - 1

The opposite form term refers to a term of the opposite form shared between C and D

- ❑ the opposite form term does not exist in A or B
- ❑ the opposite form term exists in A or B
- ❑ the opposite form term exists in A and B

Prove Lemma 5.19

Consider the case where the opposite form term does not exist in A or B

Now we have the clauses:

$$A := [a, b, \beta]$$
$$B := [c, d, \delta]$$
$$C := [a, b, \beta, e, f, \phi]$$
$$D = [c, d, \delta, \text{-}e, \hbar, \gamma]$$
$$E = [a, b, \beta, f, \phi, c, d, \delta, \hbar, \gamma]$$

Notice all of the terms in A exist in E so we can expand A to derive E by Lemma 5.8

Since the length of A is given as less than k, we can derive E by processing clauses with a maximum length of k - 1

Lemma 5.19 Checklist

Want to derive E by processing clauses with a maximum length of k - 1

The opposite form term refers to a term of the opposite form shared between C and D

- ✓ the opposite form term does not exist in A or B
- ❏ the opposite form term exists in A or B
- ❏ the opposite form term exists in A and B

Prove Lemma 5.19

Consider the case where the opposite form term exists in either A or B

Since C and D are treated the same, let's pick A and D to share the opposite form term

Now we have the clauses:

$$A := [a, b, \beta]$$
$$B := [c, d, \delta]$$
$$C := [a, b, \beta, e, f, \phi]$$
$$D = [c, d, \delta, -a, \hbar, \gamma]$$
$$E = [b, \beta, e, f, \phi, c, d, \delta, \hbar, \gamma]$$

Notice all of the terms in B exist in E so we can expand B by Lemma 5.8 and derive E

Since the length of B is given as less than k, we can derive E by processing clauses with a maximum length of k - 1

Lemma 5.19 Checklist

Want to derive E by processing clauses with a maximum length of k - 1

The opposite form term refers to a term of the opposite form shared between C and D

- ✓ the opposite form term does not exist in A or B
- ✓ the opposite form term exists in A or B
- ❑ the opposite form term exists in A and B

Prove Lemma 5.19

Consider the case the opposite form term exists in A and B
Then we have the clauses

$$A := [a, b, \beta]$$
$$B := [-a, d, \delta]$$
$$C := [a, b, \beta, e, f, \phi]$$
$$D = [-a, d, \delta, g, h, \gamma]$$
$$E = [b, \beta, e, f, \phi, d, \delta, g, h, \gamma]$$

Now we can imply a new clause

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

And since all of the terms in F exist in E, we can expand F to derive E using Lemma 5.8

Now want to show F is shorter than k

Two cases to consider: E is of length k and E is of length k - 1

Lemma 5.19 Checklist

Want to derive E by processing clauses with a maximum length of k - 1

The opposite form term refers to a term of the opposite form shared between C and D

- ✓ **the opposite form term does not exist in A or B**
- ✓ **the opposite form term exists in A or B**
- ❏ **the opposite form term exists in A and B**
  - ❏ E is of length k
  - ❏ E is of length k - 1

Prove Lemma 5.19

Want to show F is shorter than k when E is of length k

We can define k in terms of E and the length of F as follows:

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

$$\text{Length of } F = b + d + \beta + \delta - (\beta\delta)$$

Want to show the length of F $<$ k

$$b + d + \beta + \delta - (\beta\delta)$$

$$<$$

$$b + e + f + d + g + h + \beta + \phi + \delta + \gamma - (\beta\phi) - (\beta\delta) - (\beta\gamma) - (\phi\delta) - (\phi\gamma) - (\delta\gamma) + (\beta\phi\delta) + (\beta\phi\gamma) + (\phi\delta\gamma) - (\beta\phi\delta\gamma)$$

Prove Lemma 5.19

The inequality becomes

$$0 < e + f + g + \hbar + \phi + \gamma - (\beta\phi) - (\beta\gamma) - (\phi\delta) - (\phi\gamma) - (\delta\gamma) + (\beta\phi\delta) + (\beta\phi\gamma) + (\phi\delta\gamma) - (\beta\phi\delta\gamma)$$

Which is true as long as at least one term exists on the R.H.S.

Want to show at least one term exists on the R.H.S.

Prove Lemma 5.19

Want to show at least one term exists in:

$$e + f + g + h + \phi + \gamma - (\beta\phi) - (\beta\gamma) - (\phi\delta) - (\phi\gamma) - (\delta\gamma) + (\beta\phi\delta) + (\beta\phi\gamma) + (\phi\delta\gamma) - (\beta\phi\delta\gamma)$$

Suppose not, then none of the aforementioned terms exist

Recall we have the clauses

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

Since those terms do not exist, we can redefine a clause:

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

Notice A and C are exactly the same. This is a contradiction because the length of A is given as less than k and the length of C is given as k.

Therefore at least one term must exist on the R.H.S. and the inequality is true

Therefore F is shorter than k when E is of length k

Lemma 5.19 Checklist

Want to derive E by processing clauses with a maximum length of k - 1

The opposite form term refers to a term of the opposite form shared between C and D

- ✓ **the opposite form term does not exist in A or B**
- ✓ **the opposite form term exists in A or B**
- ❑ **the opposite form term exists in A and B**
  - ✓ E is of length k
  - ❑ E is of length k - 1

Prove Lemma 5.19

Consider the case when E is of length k - 1
Want to show F is shorter than k when E is of length k - 1

We can define k in terms of E and the length of F as follows:

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

$$\text{Length of } F = b + d + \beta + \delta - (\beta\delta)$$

Want to show the length of F $<$ k

$$b + d + \beta + \delta - (\beta\delta)$$

$$<$$

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

Prove Lemma 5.19

The inequality becomes

$$0 < e + f + g + \hbar + \phi + \gamma - (\beta\phi) - (\beta\gamma) - (\phi\delta) - (\phi\gamma) - (\delta\gamma) + (\beta\phi\delta) + (\beta\phi\gamma) + (\phi\delta\gamma) - (\beta\phi\delta\gamma) + 1$$

Which is always true

Therefore F is shorter than k when E is of length k - 1

Lemma 5.19 Checklist

Want to derive E by processing clauses with a maximum length of k - 1

The opposite form term refers to a term of the opposite form shared between C and D

- ✓ **the opposite form term does not exist in A or B**
- ✓ **the opposite form term exists in A or B**
- ✓ **the opposite form term exists in A and B**
  - ✓ E is of length k
  - ✓ E is of length k - 1

Since we can derive E by processing clauses with a maximum length of k - 1 for all possible cases of the lemma, then the lemma holds

Conclusion

Since 3SAT is NP-Complete according to Karp's List of 21 NP-Complete problems and there exists an algorithm to solve 3SAT in polynomial time,

$$P = NP$$

Special Thanks

Douglas Coates, High School Accelerated STEM program coordinator

for instilling in me an insatiable desire to always keep learning

Dr. Eddie Cheng, Discrete Mathematics professor at Oakland University

for teaching me proof based mathematics and reasoning

Dr. Serge Kruk, Design and Analysis of Algorithms professor at Oakland University

for teaching me intuition about problem solving

Dr. Matthew Toeniskoetter, Theory of Computation professor at Oakland University

for teaching the best class I'll ever take in such a perfect way

References

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.

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