Formalizing the Exponential Blowup in the Transformations between CNF and DNF

Leon Raffael Schulz
Ludwig-Maximilians-Universität München, Germany

Martin Desharnais-Schäfer
Ludwig-Maximilians-Universität München, Germany

Jan Johannsen
Ludwig-Maximilians-Universität München, Germany

Abstract

A well-known result about propositional logic is that transforming a formula into disjunctive or conjunctive normal form can lead to an exponential blowup of the formula size. We formalize this result in the form of two theorems in Isabelle/HOL and discuss the challenges we encountered.

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases Propositional logic, disjunctive normal form, conjunctive normal form, formula size, Isabelle/HOL

Supplementary Material *Other (Isabelle/HOL Formaliziation)*: `https://github.com/leonrschulz/Exp-Blowup/tree/b6f96622d398d3cd766f54529b36f9b115f5c43e/CNF_DNF_Exp_Blowup` [11]

Funding *Martin Desharnais-Schäfer*: This research was co-funded by the European Union (ERC, Nekoka, 101083038). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

Acknowledgements This paper is based on Schulz's bachelor thesis [10] supervised by Johannsen with the help of Desharnais-Schäfer; the thesis formalizes the theorems as found in Johannsen's lecture notes of SAT solving [3]. We thank Jasmin Blanchette for suggesting textual improvements.

1 Introduction

The metatheory of propositional logic has been studied for many decades and has been the subject of extensive formalization effort for many years. However, to the best of our knowledge, very few quantitative results have been formalized. One well-known quantitative result is that transforming a formula into disjunctive or conjunctive normal form can lead to an exponential blowup of the formula size.

In this paper, we first briefly cover the necessary mathematical background (Section 2) and then report on our Isabelle/HOL formalization of this exponential blowup. We start by presenting a few preliminary definitions and lemmas (Section 3).

We then prove the exponential blowup when converting a formula from conjunctive to disjunctive normal form (Section 4). Our formalization splits the paper proof into a low-level Proposition 16 and a high-level Theorem 17. The low-level proposition exhibits the exponential blowup for concrete formulas with specific shapes and properties. The high-level theorem corresponds to the expected mathematical result (e.g., with alternating existential and universal quantifiers) and proves that formulas expected by the low-level proposition can always be obtained.

We analogously prove the exponential blowup when converting a formula from disjunctive to conjunctive normal form (Section 5). Our formalization splits the paper proof into a low-level Proposition 23 and a high-level Theorem 24. The low-level proposition exhibits the

exponential blowup for concrete formulas with specific shapes and properties; its proof uses Proposition 16. The high-level theorem corresponds to the expected mathematical result and proves that formulas expected by the low-level proposition can always be obtained.

Finally, we discuss some of the challenges we encountered when proving statements about the size of formulas (Section 6).

Our work is part of the IsaFoL (Isabelle Formalization of Logic) effort [1]. Our formalization consists of approximately 1100 nonempty lines and is available online [11]; we will soon submit it to the *Archive of Formal Proofs*. It was developed with Isabelle2025 [7, 8] and with extensive usage of Sledgehammer [2, 9] and the structured Isar language [12]. In this paper, we only provide some key lemmas and proof sketches; we refer interested readers to the formalization for more details.

2 Mathematical Background

A *formula* of propositional logic is defined inductively as either falsehood ($\bot$), an atom (or *variable*), the negation ($\neg$) of a formula, or the disjunction ($\vee$), conjunction ($\wedge$), or implication ($\rightarrow$) of two formulas. A *literal* is either an atom x, a negated atom $\neg x$, falsehood $\bot$, or negated falsehood $\neg\bot$. The literals x and $\neg x$ as well as $\bot$ and $\neg\bot$ are *complements* of each other.

A *valuation* is a function from atoms to Boolean values. The *evaluation* of a formula w.r.t. a valuation is the Boolean value obtained after replacing all atoms with the specific truth values assigned by the valuation and evaluating according to the usual Boolean evaluation rules.

We say that a valuation α *satisfies* a formula ϕ, written $\alpha \models \phi$, if the formula evaluates to true w.r.t. α. A formula ϕ is *satisfiable* if there exists a valuation that satisfies it (i.e., $\exists\alpha.\ \alpha \models \phi$) and *tautological* if it is satisfied by every valuation (i.e., $\forall\alpha.\ \alpha \models \phi$). Two formulas ϕ and ψ are *equivalent* if they evaluate to the same truth value for every valuation (i.e., $\forall\alpha.\ \alpha \models \phi \longleftrightarrow \alpha \models \psi$).

A formula ϕ is in *negation normal form* if ϕ contains no implications and if negations only occur immediately in front of atoms or falsehood (i.e., ϕ is built from literals by conjunction and disjunction only).

For a formula ϕ in negation normal form, we define the dual formula $\bar{\phi}$ in negation normal form by replacing every literal by its complement and interchanging the symbols $\wedge$ and $\vee$. For every formula ϕ in negation normal form, $\bar{\phi}$ is equivalent to $\neg\phi$.

▶ **Definition 1.** We define the following classes of formulas:

- A *term* is a conjunction $a_1 \wedge \cdots \wedge a_k$ of literals.
- A *clause* is a disjunction $a_1 \vee \cdots \vee a_k$ of literals.
- A formula in *disjunctive normal form* (DNF) is a disjunction $T_1 \vee \cdots \vee T_m$ of terms.
- A formula in *conjunctive normal form* (CNF) is a conjunction $C_1 \wedge \cdots \wedge C_m$ of clauses.

Note that every formula in DNF or CNF is also in negation normal form. For a formula ϕ in CNF, the dual formula $\bar{\phi}$ is in DNF, and vice versa.

It is well known that every formula of propositional logic can be transformed into an equivalent formula in DNF and into an equivalent formula in CNF. However, the cost of these transformations is an exponential increase in the formula size in the worst case (see, e.g., Kleine Büning and Lettmann's textbook [4]).

▶ **Theorem 2.** *For all sufficiently large $n \in \mathbb{N}$, there exist formulas F_n in CNF with $|F_n| = \mathcal{O}(n)$ such that every equivalent formula G_n in DNF is of size $|G_n| \geq n \cdot 2^n$.*

Proof sketch. The formulas F_n are defined as $F_n = \bigwedge_{i=1}^{n}(x_{i,0} \vee x_{i,1}) \wedge (\neg x_{i,0} \vee \neg x_{i,1})$ (i.e., the conjunction of exclusive disjunctions).

If $G_n = \bigvee_{j \leq m} T_j$ is a formula in CNF equivalent to F_n, then the following two claims hold, which together prove the statement.

Claim 1. Each term T_j contains exactly one of the literals $x_{i,0}$ and $x_{i,1}$, for every $i \leq n$; otherwise, there exists a valuation that satisfies this term, and thus G_n, but does not satisfy F_n. It follows that every term T_j is of size at least n.

Claim 2. For every vector $\epsilon = (\epsilon_1, \ldots, \epsilon_n) \in \{0,1\}^n$, there is one of the terms T_j in G_n, which contains the literals $x_{1,\epsilon_1}, \ldots, x_{n,\epsilon_n}$; otherwise, we can find a valuation that satisfies F_n, but does not satisfy G_n. It follows that G_n has at least $m \geq 2^n$ many terms. ◀

▶ **Theorem 3.** *For all sufficiently large $n \in \mathbb{N}$, there exist formulas F'_n in DNF with $|F'_n| = \mathcal{O}(n)$ such that every equivalent formula G_n in CNF is of size $|G_n| \geq n \cdot 2^n$.*

Proof sketch. Let $F'_n = \bar{F}_n$ for the formula F_n from the proof of Theorem 2. If there was a formula G_n in CNF equivalent to F'_n with $|G_n| < n \cdot 2^n$, then $\bar{G}_n$ would be a formula in DNF equivalent to F_n with $|\bar{G}_n| = |G_n| < n \cdot 2^n$, contradicting Theorem 2. ◀

3 Formalization Preliminaries

For our Isabelle/HOL formalization, we reused the theories of proposition formulas from Michaelis and Nipkow's formalization of proof systems for classical propositional logic [5, 6]. Formulas have type *'a formula*, where *'a* abstracts over the type of atoms. Basic operations on formulas and related lemmas are expressed for any atom type. However, our main theorems instantiate the atom type with variables $x_{i,b}$, where i is a natural number and b is a Boolean value.

A formula is *conjunctive* if it is either a literal or its first subformula is a literal and its second a conjunctive formula. Analogously, a formula is *disjunctive* if it is either a literal or its first subformula is a literal and its second a disjunctive formula.

We now define some operations to build the conjunction and disjunction of a list of formulas.

▶ **Definition 4.** The functions BigAnd' :: *'a formula list* ⇒ *'a formula* and BigOr' :: *'a formula list* ⇒ *'a formula* build the conjunction and disjunction, respectively, of all formulas in a list:[1]

$$\mathsf{BigAnd}'\ [] = \neg\bot \qquad \mathsf{BigOr}'\ [] = \bot$$
$$\mathsf{BigAnd}'\ [\phi] = \phi \qquad \mathsf{BigOr}'\ [\phi] = \phi$$
$$\mathsf{BigAnd}'\ (\phi \# \phi s) = \phi \wedge \mathsf{BigAnd}'\ \phi s \qquad \mathsf{BigOr}'\ (\phi \# \phi s) = \phi \vee \mathsf{BigOr}'\ \phi s$$

In the rest of this paper, we write $\bigwedge_{i=0}^{|\phi s|} \phi s_{[i]}$ for $\mathsf{BigAnd}'\ \phi s$ and $\bigvee_{i=0}^{|\phi s|} \phi s_{[i]}$ for $\mathsf{BigOr}'\ \phi s$.[2]

▶ Remark. Our formalization always uses nonempty lists, and the base cases of Definition 4 could be be left unspecified. However, providing these base cases allows us to prove many lemmas for arbitrary lists without requiring a nonemptiness assumption that would need to be repeatedly discharged.

[1] The infix operation $x \# xs$ adds an element x at the front of a list xs.
[2] The operation $xs_{[i]}$ refers to the i^{th} element of a list xs. Indices are zero-based from left to right.

We will later need to generate lists of formulas for BigAnd′ and BigOr′ and define two functions to do so.

▶ **Definition 5.** The functions unconj :: *′a formula ⇒ ′a formula list* and undisj :: *′a formula ⇒ ′a formula list* extract the list of subformulas directly under a given formula's conjunctions and disjunctions, respectively:[3]

$$\mathsf{unconj}\ (\phi \wedge \psi) = \mathsf{unconj}\ \phi\ @\ \mathsf{unconj}\ \psi \qquad \mathsf{undisj}\ (\phi \vee \psi) = \mathsf{undisj}\ \phi\ @\ \mathsf{undisj}\ \psi$$

$$\mathsf{unconj}\ \phi = [\phi] \qquad \mathsf{undisj}\ \phi = [\phi]$$

We will also need to construct formulas of a very specific family and define a function to do so.

▶ **Definition 6.** The function Fn :: *nat ⇒ ′a formula* creates a formula containing the conjunction of the exclusive disjunctions of each variable pair $x_{i,\mathsf{False}}$ and $x_{i,\mathsf{True}}$ with $i \in \{1, \ldots, n\}$:

$$\mathsf{Fn}\ 0 = \neg\bot$$

$$\mathsf{Fn}\ (n+1) = ((x_{i,\mathsf{False}} \vee x_{i,\mathsf{True}}) \wedge (\neg x_{i,\mathsf{False}} \vee \neg x_{i,\mathsf{True}})) \wedge \mathsf{Fn}\ n$$

In the rest of this paper, we write F_n for $\mathsf{Fn}\ n$.

Definition 6 is designed such that a formula F_n is in CNF and can only be satisfied by valuations that assign different truth values to the variables $x_{i,\mathsf{False}}$ and $x_{i,\mathsf{True}}$.

▶ **Lemma 7.** *Let n be a natural number. The formula F_n is in CNF.*

▶ **Lemma 8.** *Let n be a natural number. Let α be a valuation. The valuation α satisfies F_n if and only if $(\alpha\ x_{i,\mathsf{False}}) \neq (\alpha\ x_{i,\mathsf{True}})$ for every $1 \leq i \leq n$.*

Finally, we provide two different definitions for the size of a formula.

▶ **Definition 9.** The functions size :: *′a formula ⇒ nat* and sizef :: *′a formula ⇒ nat* provide two metrics of formula size: size counts the number of logical symbols, and sizef counts the number of logical symbols that are not negations.

$$\mathsf{size}\ x = 1 \qquad \mathsf{sizef}\ x = 1$$

$$\mathsf{size}\ \bot = 1 \qquad \mathsf{sizef}\ \bot = 1$$

$$\mathsf{size}\ \neg\phi = \mathsf{size}\ \phi + 1 \qquad \mathsf{sizef}\ \neg\phi = \mathsf{sizef}\ \phi$$

$$\mathsf{size}\ \phi \vee \psi = \mathsf{size}\ \phi + \mathsf{size}\ \psi + 1 \qquad \mathsf{sizef}\ \phi \vee \psi = \mathsf{sizef}\ \phi + \mathsf{sizef}\ \psi + 1$$

$$\mathsf{size}\ \phi \wedge \psi = \mathsf{size}\ \phi + \mathsf{size}\ \psi + 1 \qquad \mathsf{sizef}\ \phi \wedge \psi = \mathsf{sizef}\ \phi + \mathsf{sizef}\ \psi + 1$$

$$\mathsf{size}\ \phi \to \psi = \mathsf{size}\ \phi + \mathsf{size}\ \psi + 1 \qquad \mathsf{sizef}\ \phi \to \psi = \mathsf{sizef}\ \phi + \mathsf{sizef}\ \psi + 1$$

In the rest of this paper, we write $|\phi|$ for $\mathsf{size}\ \phi$.

The function $|\cdot|$ corresponds to the intuitive notion of formula size while the function sizef is an artefact of the formalization: Intuitively, for formulas in negation normal form, sizef represents the number of logical symbols if one considers literals as basic unit instead of atoms and falsehood.

Because sizef ignores negations, it always evaluates to a number smaller than or equal to the number of $|\cdot|$.

[3] The infix operation $xs\ @\ ys$ appends a list xs at the front of a list ys.

▶ **Lemma 10.** *Let ϕ be a formula.* sizef $\phi \leq |\phi|$.

We will need to know the size of formulas built using the functions seen so far.

▶ **Lemma 11.** *Let ϕ be a formula in CNF and ψ be a formula in DNF. The following equalities hold:*

$$\text{sizef}\left(\bigwedge_{i=0}^{|\text{unconj }\phi|}(\text{unconj }\phi)_{[i]}\right) = \text{sizef }\phi \qquad \left|\bigwedge_{i=0}^{|\text{unconj }\phi|}(\text{unconj }\phi)_{[i]}\right| = |\phi|$$

$$\text{sizef}\left(\bigvee_{i=0}^{|\text{undisj }\psi|}(\text{undisj }\psi)_{[i]}\right) = \text{sizef }\psi \qquad \left|\bigvee_{i=0}^{|\text{undisj }\psi|}(\text{undisj }\psi)_{[i]}\right| = |\psi|$$

▶ **Lemma 12.** *Let n be a natural number.* sizef $F_n = 8 \cdot n + 1$ *and* $|F_n| = 10 \cdot n + 2$.

Most of the statements about the size of formulas are expressed in terms of sizef but, fortunately, Lemma 10 will allow us to express the statements of Theorems 17 and 24 entirely in terms of the more intuitive $|\cdot|$.

4 CNF to DNF

Our formalization splits the paper proof of Theorem 2 into a low-level proposition, which assumes the existence of formulas with specific shapes and properties, and a theorem, which proves that such formulas can always be obtained. We start by proving a few basic properties of conjunctive formulas w.r.t. valuations.

We first prove a monotonicity property of satisfiability w.r.t. valuations.

▶ **Lemma 13.** *Let ϕ be a conjunctive formula. Let α and β be valuations. If both valuations map every variable occurring in ϕ to the same Boolean value and if α satisfies ϕ, then β also satisfies ϕ.*

Next, we prove some sufficient conditions for a valuation to not satisfy a conjunctive formula.

▶ **Lemma 14.** *Let ϕ be a conjunctive formula. Let α be a valuation. If*

- *there exists a variable that occurs positively in ϕ and is mapped to false by α or*
- *there exists a variable that occurs negatively in ϕ and is mapped to true by α,*

then α does not satisfy ϕ.

Next, we prove a lower bound of the size of $\bigvee_{i=0}^{|\phi s|}\phi s_{[i]}$.

▶ **Lemma 15.** *Let $n > 0$ be a natural number. Let ϕs be a list of formulas. If every $\phi \in$* set ϕs *has* sizef $\phi \geq m$ *and if $|\phi s| \geq 2^n$, then* sizef $(\bigvee_{i=0}^{|\phi s|}\phi s_{[i]}) \geq m \cdot 2^n$.[4]

We can now prove that some specific formulas equivalent to F_n (Definition 6) have a size exponential in n.

▶ **Proposition 16.** *Let $n > 0$ be a natural number. Let Ts be a list of formulas. Let $\psi = \bigvee_{i=0}^{|Ts|} Ts_{[i]}$. If all formulas in Ts are conjunctive and satisfiable, and if F_n is equivalent to ψ, then* sizef $\psi \geq n \cdot 2^n$.

Proof sketch. We proceed in five steps:

[4] The function set :: *'a list* ⇒ *'a set* converts a list into a set.

1. First, we show that every conjunctive formula $T \in \mathsf{set}\, Ts$ contains for all $i \in \{1, \ldots, n\}$ either the variable $x_{i,\mathsf{False}}$ or $x_{i,\mathsf{True}}$ positively. We prove this step by contradiction, where we must derive false for two cases: Either the variables $x_{i,\mathsf{False}}$ and $x_{i,\mathsf{True}}$ are both positively contained in T or they are both not positively contained. In both cases, we proceed by defining a valuation that satisfies ψ but not F_n. For instance, in the second case, we find a valuation that satisfies T and change the assignment of the variables $x_{i,\mathsf{False}}$ and $x_{i,\mathsf{True}}$ to false. We then show using Lemmas 8, 13, and 14 that this modified valuation satisfies T (and thus ψ) but does not satisfy F_n. This means that F_n is not equivalent to ψ, which contradicts the proposition's assumption.
2. From step 1, we derive that every $T \in \mathsf{set}\, Ts$ has $\mathsf{sizef}\, T \geq n$.
3. Furthermore, we prove that for every Boolean list $\epsilon = [\epsilon_1, \ldots, \epsilon_n]$ there exists a $T \in \mathsf{set}\, Ts$ that contains the corresponding variables $x_{1,\epsilon_1}, \ldots, x_{n,\epsilon_n}$ positively. If this is not the case for a specific list ϵ, we show using step 1 that every T must contain one of the variables $x_{1,\neg\epsilon_1}, \ldots, x_{n,\neg\epsilon_n}$ positively. We then define a valuation that maps all variables x_{i,ϵ_i} to true and $x_{i,\neg\epsilon_i}$ to false for $i \in \{1, \ldots, n\}$. We show using Lemma 14 that this valuation satisfies F_n but not ψ. This means that F_n is not equivalent to ψ, which contradicts the proposition's assumption.
4. We then prove that the number of Boolean lists of length n is $2^n \leq |\mathsf{set}\, Ts| \leq |Ts|$ using steps 1 and 3.
5. Finally, we show that $\mathsf{sizef}\, \psi \geq n \cdot 2^n$ by combining the results from steps 2 and 4 to discharge the assumptions of Lemma 15. ◀

Proposition 16 assumes the existence of two equivalent formulas F_n (in CNF) and $\bigvee_{i=0}^{|Ts|} Ts_{[i]}$ (in DNF) and proves a lower bound on the size of the later. We now prove that such equivalent formulas do exist and that the exponential blowup holds for all of them.

▶ **Theorem 17** (Exponential Blowup from CNF to DNF). *Let n be a natural number. There exists a formula ϕ in CNF with size $|\phi| = 10 \cdot n + 2$ such that every equivalent formula ψ in DNF has size $|\psi| \geq n \cdot 2^n$.*

Proof sketch. We start by a case analysis on n. The interesting case is when $n > 0$. We define $\phi = F_n$ (Definition 6); it is easy to show that F_n is in CNF (using Lemma 7) and that $|\phi| = 10 \cdot n + 2$ (using Lemma 12). To prove the last property, we assume that there is an equivalent formula ψ in DNF. We first obtain a list Ts (by filtering $\mathsf{undisj}\, \psi$) of all satisfiable terms in ψ such that all formulas in Ts are conjunctive and satisfiable, $\mathsf{sizef}\, \psi \geq \mathsf{sizef}\, (\bigvee_{i=0}^{|Ts|} Ts_{[i]})$ (using Lemma 11), and $\bigvee_{i=0}^{|Ts|} Ts_{[i]}$ is equivalent to ψ. From this last equivalence, we show that $\bigvee_{i=0}^{|Ts|} Ts_{[i]}$ is equivalent to F_n. This allows us to prove that $\mathsf{sizef}\, (\bigvee_{i=0}^{|Ts|} Ts_{[i]}) \geq n \cdot 2^n$ using Proposition 16. We can prove that $\mathsf{sizef}\, \psi \geq n \cdot 2^n$ by transitivity and, finally, that $|\psi| \geq n \cdot 2^n$ using Lemma 10. ◀

We claim that Theorem 17 is a formalization of the paper proof of Theorem 2.

5 DNF to CNF

Our formalization splits the paper proof of Theorem 3 into a low-level proposition, which assumes the existence of formulas with specific shapes and properties, and a theorem, which proves that such formulas can always be obtained. We start by defining a function to produce the dual of a formula and proving a few basic properties.

▶ **Definition 18.** The function $\mathsf{dual} :: 'a\ formula \Rightarrow 'a\ formula$ converts a formula to its dual form by replacing each literal with its complement: an atom x becomes $\neg x$, falsehood

$\bot$ becomes $\neg\bot$, a negation $\neg\phi$ becomes ϕ, a disjunction $\phi \vee \psi$ becomes $\mathsf{dual}\,\phi \wedge \mathsf{dual}\,\psi$, a conjunction $\phi \wedge \psi$ becomes $\mathsf{dual}\,\phi \vee \mathsf{dual}\,\psi$, and an implication $\phi \to \psi$ becomes $\phi \wedge \mathsf{dual}\,\psi$.

▶ Remark. In contrast to the definition of dual formulas in Section 2, we specify the dual function for arbitrary formulas to avoid the recurring assumption that formulas are in negation normal form.

The semantics of a dual formula is the exact opposite of that of the original formula.

▶ **Lemma 19.** *Let ϕ be a formula. The formula* $\mathsf{dual}\,\phi$ *is equivalent to $\neg\phi$.*

Because the sizef function ignores negations, the dual function preserves the size of the original formula.

▶ **Lemma 20.** *Let ϕ be a formula.* $\mathsf{sizef}\,(\mathsf{dual}\,\phi) = \mathsf{sizef}\,\phi$.

Next, we prove some sufficient condition for a dual formula to be satisfiable.

▶ **Lemma 21.** *Let ϕ be a formula. If ϕ is disjunctive and not tautological, then* $\mathsf{dual}\,\phi$ *is satisfiable.*

Next, we prove that the dual of some specific formulas in DNF are in CNF.

▶ **Lemma 22.** *Let Cs be a list of formulas. If all formulas in Cs are disjunctive and not tautological, then there exists a list Ts such that all formulas in Ts are conjunctive and satisfiable, and* $\mathsf{dual}\,(\bigwedge_{i=0}^{|Cs|} Cs_{[i]}) = \bigvee_{i=0}^{|Ts|} Ts_{[i]}$.

Proof sketch. We prove this by structural induction over the list Cs using Lemma 21. The idea is to map each element of Cs to its dual form. ◀

We can now prove that some specific formulas equivalent to the dual of F_n (Definition 6) have a size exponential in n.

▶ **Proposition 23.** *Let $n > 0$ be a natural number. Let Cs be a list of formulas. Let $\psi = \bigwedge_{i=0}^{|Cs|} Cs_{[i]}$. If all formulas in Cs are disjunctive and not tautological, and* $\mathsf{dual}\,F_n$ *is equivalent to ψ, then* $\mathsf{sizef}\,\psi \geq n \cdot 2^n$.

Proof sketch. We prove this by contradiction using Proposition 16. We assume $\mathsf{sizef}\,\psi < n \cdot 2^n$ and from that derive false in five steps:

1. First, we know from Lemma 20 that $\mathsf{sizef}\,(\mathsf{dual}\,\psi) = \mathsf{sizef}\,\psi$, thus $\mathsf{sizef}\,(\mathsf{dual}\,\psi) < n \cdot 2^n$.
2. Moreover, we show that $\mathsf{dual}\,\psi$ is equivalent to F_n using, among others, Lemma 19.
3. Also, we prove using Lemma 22 that there exists a list Ts such that every formula contained in Ts is both conjunctive and satisfiable and such that $\mathsf{dual}\,\psi$ is syntactically equal to $\bigvee_{i=0}^{|Ts|} Ts_{[i]}$.
4. From steps 2 and 3, we can discharge the assumptions of Proposition 16 for F_n and $\mathsf{dual}\,\psi$ and thus show that $\mathsf{sizef}\,(\mathsf{dual}\,\psi) \geq n \cdot 2^n$.
5. The result of step 1 contradicts the result of step 4, completing the proof. ◀

Proposition 23 assumes the existence of two equivalent formulas $\mathsf{dual}\,F_n$ (in DNF) and $\bigwedge_{i=0}^{|Cs|} Cs_{[i]}$ (in CNF) and proves a lower bound on the size of the later. We now prove that such equivalent formulas do exist and that the exponential blowup holds for all of them.

▶ **Theorem 24** (Exponential Blowup from DNF to CNF)**.** *Let n be a natural number. There exists a formula ϕ in DNF with size $|\phi| = 10 \cdot n + 1$ such that every equivalent formula ψ in CNF has size $|\psi| \geq n \cdot 2^n$.*

Proof sketch. The proof is analogous to the proof of Theorem 17. ◀

We claim that Theorem 24 is a formalization of the paper proof of Theorem 3.

6 Discussion

Whereas most ideas of the paper proof can be directly used for the formalization, some intuitive mathematical proof steps required significant formalization effort. This was most pronounced in the proof steps 2 and 4 of Proposition 16, where we derived statements about the size of a formula from its properties. To prove step 4, we use a general lemma (already included in Isabelle/HOL) about the cardinality of finite sets.

▶ **Lemma 25.** *Let X and Y be sets. Let f be a unary function. If Y is finite, and if $\{f\ x \mid x \in X\} \subseteq Y$, and if f is injective on X, then the cardinality of X is less than or equal to the cardinality of Y.*

We use Lemma 25 to prove that the number of Boolean lists of length n is less than or equal to the cardinality of $\mathsf{set}\ Ts$. We instantiate X with the set of all Boolean lists of length n, Y with $\mathsf{set}\ Ts$, and f with a function that takes a Boolean list and returns some conjunctive formula $T \in \mathsf{set}\, Ts$ for which the predicate of step 3 holds. Finding an appropriate function for f and discharging Lemma 25's assumptions was surprisingly challenging: The first assumption was the most intricate and its proof uses both steps 1 and 3, the second assumption follows from step 3, and the third assumption easily holds.

To prove step 2, we introduced a new lemma, inspired by Lemma 25, that uses a binary function that is right total w.r.t. the target set.

▶ **Lemma 26.** *Let X be a set of elements of type $'a$ and Y be a set of elements of type $'c$. Let $f :: 'a \Rightarrow 'b \Rightarrow 'c$ be a binary function. If X and Y are finite sets, and if for all $x \in X$ there exists an element y such that $f\ x\ y \in Y$, and if the function f is injective on all those pairs of x and y, then the cardinality of X is less than or equal to the cardinality of Y.*

We use Lemma 26 to prove that every $T \in \mathsf{set}\, Ts$ contains at least n atoms and thus that $\mathsf{sizef}\ T \geq n$. To achieve this, we instantiate X with $\{1, \ldots, n\}$, Y with the set of atoms of T, and f with the constructor for variables. Finding the correct statement for Lemma 26 was challenging but, once it was found, it was easy to discharge the assumptions using step 1.

Moreover, we chose to modify the functions BigAnd and BigOr from Michaelis and Nipkow [5] and to define our own versions in Definition 4, where the base cases are handled differently. Our functions avoid appending $\neg\bot$ and $\bot$ to the produced formula when the provided list is nonempty. This allowed us to prove the size equalities in Lemma 11 instead of having the left-hand side of the lemmas be slightly smaller than the right-hand side. Using the original functions would have been possible, but it would have resulted in Theorems 17 and 24 stating that the size $|\psi| + 2 \geq n \cdot 2^n$ instead of $|\psi| \geq n \cdot 2^n$. To keep the statements close to their mathematical counterparts, we therefore chose to introduce the functions BigAnd′ and BigOr′.

Finally we introduced the function sizef to be able to prove the equality of Lemma 20. Without sizef, we could only prove the inequality $|\mathsf{dual}\ \phi| \leq 2 \cdot |\phi|$ because dual adds and removes negations depending on the formula. It would be possible to restate and prove the statement of Proposition 16 in terms of $|\cdot|$. However, if we tried to restate and prove the statement of Proposition 23 in terms of $|\cdot|$, we would prove $|\mathsf{dual}\ \psi| < 2 \cdot n \cdot 2^n$ in step 1 and $|\mathsf{dual}\ \psi| \geq n \cdot 2^n$ in step 4, and we would not be able to derive a contradiction in step 5.

7 Conclusion

We formalized in Isabelle/HOL the well-known result about propositional logic that transforming a formula into DNF or CNF can lead to an exponential blowup of the formula size.

The proof is split into a low-level proposition and a theorem for conversion to DNF and analogously for conversion to CNF. We also reported of some challenges we encountered when proving statements about the size of formulas.

Our formalization may serve as a basis and template for future formalizations of quantitative results in propositional logic.

References

1 Jasmin Christian Blanchette. Formalizing the metatheory of logical calculi and automatic provers in Isabelle/HOL (invited talk). In Assia Mahboubi and Magnus O. Myreen, editors, *Proceedings of the 8th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP 2019)*, pages 1–13. ACM, 2019. doi:10.1145/3293880.3294087.

2 Jasmin Christian Blanchette, Sascha Böhme, and Lawrence C. Paulson. Extending Sledgehammer with SMT solvers. *Journal of Automated Reasoning*, 51(1):109–128, 2013. doi:10.1007/S10817-013-9278-5.

3 Jan Johannsen. SAT Solving: Skript zur Vorlesung im WS 2023/24, 2023. URL: https://www.ifi.lmu.de/institut/personen/jjohannsen/jj_unpublished/skript.pdf.

4 Hans Kleine Büning and Theodor Lettmann. *Aussagenlogik: Deduktion und Algorithmen*. B. G. Teubner, Stuttgart, 1994. doi:10.1007/978-3-322-84809-3.

5 Julius Michaelis and Tobias Nipkow. Propositional proof systems. *Archive of Formal Proofs*, June 2017. https://isa-afp.org/entries/Propositional_Proof_Systems.html, Formal proof development.

6 Julius Michaelis and Tobias Nipkow. Formalized proof systems for propositional logic. In Andreas Abel, Fredrik Nordvall Forsberg, and Ambrus Kaposi, editors, *23rd International Conference on Types for Proofs and Programs (TYPES 2017)*, volume 104 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5:1–5:16, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.TYPES.2017.5, doi:10.4230/LIPIcs.TYPES.2017.5.

7 Tobias Nipkow and Gerwin Klein. *Concrete Semantics: With Isabelle/HOL*. Springer, 2014. doi:10.1007/978-3-319-10542-0.

8 Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science (LNCS)*. Springer, 2002. doi:10.1007/3-540-45949-9.

9 Lawrence C. Paulson and Jasmin Christian Blanchette. Three years of experience with Sledgehammer, a practical link between automatic and interactive theorem provers. In Geoff Sutcliffe, Stephan Schulz, and Eugenia Ternovska, editors, *The 8th International Workshop on the Implementation of Logics (IWIL-2010)*, volume 2 of *EPiC*, pages 1–11. EasyChair, 2012. doi:10.29007/36dt.

10 Leon Raffael Schulz. Formalisierung einer unteren Schranke an die Formelgröße in der Aussagenlogik mit Isabelle, 2026. doi:10.5282/ubm/epub.135623.

11 Leon Raffael Schulz, Martin Desharnais-Schäfer, and Jan Johannsen. Exponential blowup in the transformations between CNF and DNF, 2026. Formal proof development. URL: https://github.com/leonrschulz/Exp-Blowup/tree/b6f96622d398d3cd766f54529b36f9b115f5c43e/CNF_DNF_Exp_Blowup.

12 Makarius Wenzel. Isabelle/Isar—a generic framework for human-readable proof documents. In Roman Matuszewski and Anna Zalewska, editors, *From Insight to Proof: Festschrift in Honour of Andrzej Trybulec*, volume 10(23) of *Studies in Logic, Grammar, and Rhetoric*. University of Białystok, 2007.