Formalizing Frankl's Conjecture: FC-families

Filip Marić, Miodrag Živković, Bojan Vučković

Faculty of Mathematics, University of Belgrade*

Abstract. The Frankl's conjecture, formulated in 1979. and still open, states that in every family of sets closed for unions there is an element contained in at least half of the sets. FC-families are families for which it is proved that every union-closed family containing them satisfies the Frankl's condition (e.g., in every union-closed family that contains a one-element set a, the element a is contained in at least half of the sets, so families of the form a are the simplest FC-families). FC-families play an important role in attacking the Frankl's conjecture, since they enable significant search space pruning. We present a formalization of the computer assisted approach for proving that a family is an FC-family. Proof-by-computation paradigm is used and the proof assistant Isabelle/HOL is used both to check mathematical content, and to perform (verified) combinatorial searches on which the proofs rely. FC-families known in the literature are confirmed, and a new FC-family is discovered.

1 Introduction

Formalized mathematics and interactive theorem provers (sometimes referred to as proof assistants) have made great progress in recent years. Many classical mathematical theorems have been formally proved and proof assistants have been intensively used in hardware and software verification. The most successful proof assistants now days are Coq, Isabelle/HOL, HOL Light, etc.

Several of the most important results in formal theorem proving are for the problems that require proofs with much computational content. These proofs are usually highly complex (and therefore often require justifications by formal means) since they combine classical mathematical statements with complex computing machinery (usually computer implementation of combinatorial algorithms). The corresponding paradigm is sometimes referred to as *proof-by-evaluation* or *proof-by-computation*. Probably, the most famous examples of this approach are the proofs of the Four-Color Theorem and the Kepler's conjecture.

Georges Gonthier has formalized a proof of the Four-Color Theorem[1] in Coq [6]. The Four Colour Theorem is famous for being the first long-standing mathematical problem, analyzed by many famous mathematicians, finally resolved by

* The first author was partially supported by the Serbian Ministry of Education and Science grant 174021 and by the SNF grant SCOPES IZ73Z0127979/1, the second author by the Serbian Ministry of Education and Science grant 174021 and the third author by the Serbian Ministry of Education and Science grant 044006 (III).

[1] In 1852. Francis Guthrie conjectured that every map can be colored with at most 4 colors such that no two adjacent regions share the same color.

a computer program (Appel and Haken [2]). This proof broke new ground because it involved using IBM 370 assembly language computer programs to carry out a gigantic case analysis, which could not be performed by hand. The proof attracted criticism: computer programming is known to be error-prone, and difficult to relate precisely to the formal statement of a mathematical theorem. Several attempts to simplify the proofs were made (e.g., Robertson et al. [13]), number of cases was reduced and programs were written in C instead of assembly language. However, all doubts were removed only when Gonthier employed proof assistant Coq reducing the whole proof to several basic logical principles.

Another example of a similar kind is the proof of Kepler's conjecture[2]. As described by Nipkow et al. [9]: "In 1998. Thomas Hales announced the first (by now) accepted proof of Kepler's conjecture. It involves 3 distinct large computations. After 4 years of refereeing by a team of 12 referees, the referees declared that they were 99% certain of the correctness of the proof. Dissatisfied with this, Hales started the informal open-to-all collaborative *flyspeck* project to formalize the whole proof with a theorem proof."

In this work, we apply the proof-by-evaluation paradigm to a problem of verifying FC-families — a special case of the Frankl's conjecture. Frankl's conjecture, an elementary and fundamental statement formulated by Péter Frankl in 1979., states that for every family of sets closed under unions, there is an element contained in at least half of the sets (or, dually, in every family of sets closed under intersections, there is an element contained in at most half of the sets). Up to the best of our knowledge, the problem is still open. The conjecture has been proved for many special cases. In particular, it is known to be true for: (i) families of at most 36 sets[3] [4]; (ii) families of sets such that their union has at most 11 elements [3].

FC-families are families for which it is proved that all union closed families containing them satisfy the Frankl's condition (if the Frankl's conjecture would be proved, then every family would be an FC-family). For example, it can easily be shown that if a family contains a one-element set, then it satisfies the Frankl's condition. Similar results holds for any two-element set, etc. FC-families are important building block for attempting to prove the Frankl's conjecture since they justify pruning large portions of the search space.

Related work. The Frankl's conjecture has also been formulated and studied as a question in lattice theory [12, 1].

FC-families have been introduced by Poonen [11] and further studied by Gao and Yu [5], Vaughan [14–16], Morris [8], Marković [7], Bošnjak and Marković [3], and Živković and Vučković [17].

[2] In 1611 Kepler asserted that the so called cannonball packing is a densest arrangement of 3-dimensional balls of the same size.

[3] Unpublished report by Roberts from 1992 claimis a similar result for families of at most 40 sets.

The basic technique used (the Frankl's condition characterization based on weight functions and shares) is introduced by Poonen [11] and later successfully used by Bošnjak and Marković [7, 3], and Živković and Vučković [17].

First attempts in using computer-assisted computational approach on solving special cases of the Frankl's conjecture are described by Živković and Vučković [17]. Computations are performed by (unverified) Java programs. However, in order to increase the level of trust, Java programs generate certificates that can be checked by independent tools.

The present paper represent a formalized reformulation of the results of Živković and Vučković [17]. All mathematical content is rigorously formalized within Isabelle/HOL and proofs are mechanically checked. JAVA programs are reimplemented in a functional language of Isabelle/HOL and their correctness is formally verified. A clear separation of mathematical and computational content is done and parts of the proofs that rely on computations are clearly isolated. Since the whole formalization is performed and verified within a proof assistant, there is no need for explicit certificates for statements proved by computation.

Our main contribution are rigorous, machine-verifiable proofs[4] that all FC-families previously described in the literature are indeed FC-families. Unlike most pen-and-paper proofs, our proofs follow a uniform approach, supported by an underlying combinatorial search procedure. The second contribution is a new type of FC-families: four three-element sets all contained in a seven-element set.

Background logic and notation. Logic and the notation given in this paper will follow Isabelle/HOL. Isabelle/HOL [10] is a development of Higher Order Logic (HOL), and it conforms largely to everyday mathematical notation. The basic types include truth values (*bool*), natural numbers (*nat*) and integers (*int*). Functions can be defined by recursion (either primitive or general). Sets over type α, type $\alpha\ set$, follow the usual mathematical conventions[5]. Sets of sets (i.e., object of the type $\alpha\ set\ set$) are called families. Set of all subset for a set A is denoted by $\mathsf{pow}\ A$, and its number of elements is denoted by $|A|$. Lists over type α, type $\alpha\ list$, come with the empty list $[\,]$, the infix prepend constructor #, the infix @ that appends two lists, and the conversion function *set* from lists to sets. N-th element of a list l is denoted by $l_{[n]}$. List $[0, 1, \ldots, n-1]$ is denoted by $[0..<n]$. The function sort sorts a list, listsum calculates its sum, and remdups removes duplicate elements. List with no repeated elements are called distinct. Standard higher order functions map, filter, foldl are also supported (for details see [10]).

All definitions and statements given in this paper are formalized within Isabelle/HOL. However, in order to make the text accessible to a more general audience not familiar with Isabelle/HOL, many minor details are omitted and some imprecisions are introduced (for example, we used standard symbolics used in related work, although it is clear that some symbols are ambigous). Statements

[4] Corresponding Isabelle/HOL proof documents are available from `http://argo.matf.bg.ac.rs`

[5] In a strict type setting, sets containing elements of mixed types are not allowed.

are grouped into propositions, lemmas, and theorems. Propositions usually express simple, technical results and are printed here without proofs. All sets and families are considered to be finite and this assumptions (present in Isabelle/HOL formalization) will not be explicitly stated in the rest of the paper.

Outline. The rest of the paper is organized as follows. In Section 2 we give mathematical background on union-closed families, the Frankl's conjecture and prove main theoretical results. In Section 3 we formulate the combinatorial search algorithm, prove its correctness and give its efficient implementation. In Section 4 we introduce uniform families and techniques used for avoiding symmetries when analyzing them. In Section 5 we verify several kinds of uniform FC-families. Finally, in Section 6 we draw conclusions and give directions for further work.

2 Frankl's Families

2.1 Union Closed Families

First we give basic definitions of union-closed families, closure under unions, and operations used to incrementally obtain closed families.

Definition 1. *Let F and F_c be families.*

F *is* union closed, *denoted by* $\mathsf{uc}\ F$, *iff* $\forall A \in F.\ \forall B \in F.\ A \cup B \in F$. F *is* union closed for F_c, *denoted by* $\mathsf{uc}_{F_c}\ F$, *iff* $\mathsf{uc}\ F \wedge (\forall A \in F.\ \forall B \in F_c.\ A \cup B \in F)$.

Closure of F, *denoted by* $\langle F \rangle$, *is the minimal family of sets (in sense of inclusion) that contains F and is union closed.* Closure of F for F_c, *denoted by* $\langle F \rangle_{F_c}$, *is the minimal family of sets (in sense of inclusion) that contains F and is union closed for F_c.*

Insert and close operation *of set A to family F, denoted by* $\mathsf{ic}\ A\ F$, *is the family* $F \cup\ \{A\}\ \cup\ \{A \cup B.\ B \in F\}$. Insert and close operation for F_c *of set A to family F, denoted by* $\mathsf{ic}_{F_c}\ A\ F$, *is the family* $F \cup\ \{A\}\ \cup\ \{A \cup B.\ B \in F\}\ \cup\ \{A \cup B.\ B \in F_c\}$.

Proposition 1.
1. $\langle F \rangle = \{\bigcup F'.\ F' \in \mathsf{pow}\ F - \{\emptyset\}\}$
2. $\langle F \cup \{A\} \rangle = \mathsf{ic}\ A\ \langle F \rangle, \quad \langle F \cup \{A\} \rangle_I = \mathsf{ic}_I\ A\ \langle F \rangle$
3. *If* $F \subseteq \mathsf{pow}\ \bigcup A$ *and* $\mathsf{uc}_A\ F$ *then* $\mathsf{uc}_{\langle A \rangle}\ F$.

2.2 The Frankl's Condition

The next definition formalizes the Frankl's condition and the notion of FC-family.

Definition 2. *Family of sets F satisfies the* Frankl's condition *and we say that it is a* Frankl's family, *denoted by* $\mathsf{frankl}\ F$, *if it contains an element that occurs in at least half sets in the family, i.e.,* $\mathsf{frankl}\ F\ \equiv\ \exists a.\ a \in \bigcup F\ \wedge\ 2 \cdot \#_a F \geq |F|$, *where* $\#_a F$ *denotes* $|\{A \in F.\ a \in A\}|$

Family of sets F_c is FC-family *if it is proved that every union closed family such that $F \supseteq F_c$ is Frankl's.*

2.3 Family Isomorphisms

The domain of the family does not play any important role for many properties related to the Frankl's condition — many properties are invariant for domain changes using injective functions (that establish a kind of isomorphisms between two families). Therefore, in many cases it suffices to consider only families over canonical domains — initial ranges $\{0, 1, \ldots, n-1\}$ of natural numbers.

Proposition 2. *Let F be a family of sets and f a function injective on $\bigcup F$. Let F' be the image of F under f (then f is a bijection between $\bigcup F$ and $\bigcup F'$).*

1. *If $a \in \bigcup F$, then $\#_a F = \#_{f(a)} F'$.*
2. *$|F| = |F'|$*
3. *If $A \in F$ and $A' \in F'$ is the image of A under f, then $|A| = |A'|$.*
4. *F is union closed if and only if F' is.*
5. *F is Frankl's if and only if F' is.*
6. *If F' is an FC-family, then so is F.*

2.4 FC Characterization by Weight Functions and Shares

We describe the central technique for proving that a family is FC-family, relying on characterizations of the Frankl's condition using weights and shares.

Definition 3. *A function $w : X \to \mathbb{N}$ is a* weight function on $A \subseteq X$, *denoted by* $\mathsf{wf}_A\ w$*, iff $\exists a \in A.\ w(a) > 0$.* Weight of a set A wrt. weight function w, *denoted by $w(A)$, is the value $\sum_{a \in A} w(a)$.* Weight of a family F wrt. weight function w, *denoted by $w(F)$, is the value $\sum_{A \in F} w(A)$.*

Lemma 1. $\mathsf{frankl}\ F \iff \exists w.\ \mathsf{wf}_{(\bigcup F)}\ w\ \wedge\ 2 \cdot w(F)\ \geq\ w(\bigcup F) \cdot |F|$

Proof. Assume $\mathsf{frankl}\ F$ and let a be the element satisfying the Frankl's condition. Let w be the weight function assigning 1 to a and 0 to all other elements. Since $w(F) = \#_a F$ and $w(\bigcup F) = 1$, the statements holds.

Conversely, suppose that $\neg\mathsf{frankl}\ F$. Then, for every $a \in \bigcup F$, $2 \cdot \#_a F < |F|$. Hence, $2 \cdot w(F) = \sum_{a \in \bigcup F} w(a) \cdot 2 \cdot \#_a F < |F| \cdot \sum_{a \in \bigcup F} w(a) = |F| \cdot w(\bigcup F)$.

A concept that will enable a slightly more operative formulation of the previous characterization is the concept of *share*[6].

Definition 4. *Let w be a weight function.* Share of a set A wrt. w and a set X, *denoted by $\bar{w}_X(A)$, is the value $2 \cdot w(A) - w(X)$.* Share of a family F wrt. w and a set X, *denoted by $\bar{w}_X(F)$, is the value $\sum_{A \in F} \bar{w}_X(A)$.*

[6] Note that in order to accommodate for computer implementation only integer weights are allowed, and to avoid rational numbers share of a set A is defined as $2 \cdot w(A) - w(X)$, instead of $w(A) - w(X)/2$ that is used in the literature.

Example 1. Let w be a function such that $w(a_0) = 1, w(a_1) = 2$, and $w(a) = 0$ for all other elements. w is clearly a weight function. Then, $w(\{a_0, a_1, a_2\}) = 3$ and $w(\{\{a_0, a_1\}, \{a_1, a_2\}, \{a_1\}\}) = 7$. Also, $\bar{w}_{\{a_0,a_1,a_2\}}(\{a_1, a_2\}) = 2 \cdot w(\{a_1, a_2\}) - w(\{a_0, a_1, a_2\}) = 4 - 3 = 1$, and $\bar{w}_{\{a_0,a_1,a_2\}}(\{\{a_0, a_1\}, \{a_1, a_2\}, \{a_1\}\}) = (2 \cdot 3 - 3) + (2 \cdot 2 - 3) + (2 \cdot 2 - 3) = 5$.

Proposition 3. $\bar{w}_X(F) = 2 \cdot w(F) - w(X) \cdot |F|$

Lemma 2. $\mathsf{frankl}\ F \iff \exists w.\ \mathsf{wf}_{(\bigcup F)}\ w\ \wedge\ \bar{w}_{(\bigcup F)}(F) \geq 0$

Proof. Follows directly from Proposition 3 and Lemma 1.

Hypercubes. Sets of a family can be grouped into so called hypercubes.

Definition 5. *An* S-hypercube *with a base* K*, denoted by* hc^S_K*, is the family* $\{A.\ K \subseteq A \wedge A \subseteq K \cup S\}$*. Alternatively, a hypercube can be characterized by* $\mathsf{hc}^S_K = \{K \cup A.\ A \in \mathsf{pow}\ S\}$*.*

Example 2. Let $S \equiv \{s_0, s_1\}$, and $K \equiv \{k_0, k_1\}$. If $K' \subseteq K$, then all S-hypercubes with a base K' are:

$$\begin{aligned}
\mathsf{hc}^S_{\{\}} &= \{\{\}, \{s_0\}, \{s_1\}, \{s_0, s_1\}\} \\
\mathsf{hc}^S_{\{k_0\}} &= \{\{k_0\}, \{k_0, s_0\}, \{k_0, s_1\}, \{k_0, s_0, s_1\}\} \\
\mathsf{hc}^S_{\{k_1\}} &= \{\{k_1\}, \{k_1, s_0\}, \{k_1, s_1\}, \{k_1, s_0, s_1\}\} \\
\mathsf{hc}^S_{\{k_0,k_1\}} &= \{\{k_0, k_1\}, \{k_0, k_1, s_0\}, \{k_0, k_1, s_1\}, \{k_0, k_1, s_0, s_1\}\}
\end{aligned}$$

Previous example indicates that (disjoint) S-hypercubes can span the whole $\mathsf{pow}\ (K \cup S)$. Indeed, this is generally the case.

Proposition 4. *(i)* $\mathsf{pow}\ (K \cup S) = \bigcup_{K' \subseteq K} \mathsf{hc}^S_{K'}$*. (ii) If* K_1 *and* K_2 *are different and disjoint with* S*, then* $\mathsf{hc}^S_{K_1}$ *and* $\mathsf{hc}^S_{K_2}$ *are disjoint.*

Families of sets can be separated into (disjoint) parts belonging to different hypercubes (formed as $\mathsf{hc}^S_K \cap F$).

Definition 6. *A* hyper-share of a family F wrt. weight function w, the hypercube hc^S_K and the set X*, denoted by* $\bar{w}^S_{KX}(F)$*, is the value* $\sum_{A \in \mathsf{hc}^S_K \cap F} \bar{w}_X(A)$*.*

Example 3. Let S and K be as in the Example 2, let $X \equiv K \cup S$, let $F \equiv \{\{s_0\}, \{s_1\}, \{k_0, s_0\}, \{k_0, k_1, s_0, s_1\}\}$, and $w(a) = 1$ for all $a \in X$. Then, $\bar{w}^S_{\{\}X}(F) = \bar{w}_X(\{s_0\}) + \bar{w}_X(\{s_1\}) = -4$, $\bar{w}^S_{\{k_0\}X}(F) = \bar{w}_X(\{k_0, s_0\}) = 0$, $\bar{w}^S_{\{k_1\}X}(F) = 0$, and $\bar{w}^S_{\{k_0,k_1\}X}(F) = \bar{w}_X(\{k_0, k_1, s_0, s_1\}) = 4$.

Share of a family can be expressed in terms of sum of hyper-shares.

Proposition 5. *If* $K \cup S = \bigcup F$ *and* $K \cap S = \emptyset$*, then* $\bar{w}_{(\bigcup F)}(F) = \sum_{K' \subseteq K} \bar{w}^S_{K'(\bigcup F)}(F)$*.*

Lemma 3. *Let* w *be a weight function on* $\bigcup F$*. If* $K \cup S = \bigcup F$*,* $K \cap S = \emptyset$*, and* $\forall K' \subseteq K.\ \bar{w}^S_{K'(\bigcup F)}(F) \geq 0$*, then* $\mathsf{frankl}\ F$*.*

Proof. Immediate consequence of Proposition 5 and Lemma 2.

Definition 7. Projection of a family F onto a hypercube hc_K^S, *denoted by* $\mathsf{hc}_K^S \lfloor F \rfloor$, *is the set* $\{A - K.\ A \in \mathsf{hc}_K^S \cap F\}$.

Example 4. Let K, S and F be as in Example 3. Then $\mathsf{hc}_{\{\}}^S \lfloor F \rfloor = \{\{s_0\}, \{s_1\}\}$, $\mathsf{hc}_{\{k_0\}}^S \lfloor F \rfloor = \{\{s_0\}\}$, $\mathsf{hc}_{\{k_1\}}^S \lfloor F \rfloor = \{\}$, and $\mathsf{hc}_{\{k_0,k_1\}}^S \lfloor F \rfloor = \{\{s_0, s_1\}\}$.

Proposition 6.

1. *If* $K \cap S = \emptyset$ *and* $K' \subseteq K$, *then* $\mathsf{hc}_{K'}^S \lfloor F \rfloor \subseteq \mathsf{pow}\ S$
2. *If* $\mathsf{uc}\ F$, *then* $\mathsf{uc}\ (\mathsf{hc}_K^S \lfloor F \rfloor)$.
3. *If* $\mathsf{uc}\ F$, $F_c \subseteq F$, $S = \bigcup F_c$, $K \cap S = \emptyset$, *then* $\mathsf{uc}_{F_c}\ (\mathsf{hc}_K^S \lfloor F \rfloor)$.
4. *If* $\forall x \in K.\ w(x) = 0$, *then* $\bar{w}_{KX}^S(F) = \bar{w}_X(\mathsf{hc}_K^S \lfloor F \rfloor)$.

Union closed extensions. The next definition introduces an important notion for checking FC-families.

Definition 8. Union closed extensions *of a family* F_c *are families that are created from elements of* F_c *and are union closed for* F_c. *Family of all union closed extensions is denoted by* $\mathsf{uce}\ F_c$, *and* $\mathsf{uce}\ F_c \equiv \{F'.\ F' \subseteq \mathsf{pow}\ \bigcup F_c \wedge \mathsf{uc}_{F_c}\ F'\}$.

Lemma 4. *Let F be a non-empty union closed family, and let F_c be a subfamily (i.e., $F_c \subseteq F$). Let S denote $\bigcup F_c$, and let K denote $\bigcup F - \bigcup F_c$. Let w be a weight function on $\bigcup F$, that is zero for all elements of K. If shares of all union closed extension of F_c are nonnegative, then F is Frankl's, i.e., if* $\forall F' \in \mathsf{uce}\ F_c.\ \bar{w}_{(\bigcup F_c)}(F') \geq 0$, *then* $\mathsf{frankl}\ F$.

Proof. Since, $K \cup S = \bigcup F$ and $K \cap S = \emptyset$, by Lemma 3, it suffices to show that $\forall K' \subseteq K.\ \bar{w}_{K'(\bigcup F)}^S(F) \geq 0$. Fix K' and assume that $K' \subseteq K$. Since w is zero on K, by Proposition 6, it holds that $\bar{w}_{K'(\bigcup F)}^S(F) = \bar{w}_{(\bigcup F)}(\mathsf{hc}_{K'}^S \lfloor F \rfloor)$. On the other hand, since $\mathsf{uc}\ F$, $F_c \subseteq F$, and $K \cap S = \emptyset$, by Proposition 6 it holds that $\mathsf{uc}_{F_c}\ (\mathsf{hc}_{K'}^S \lfloor F \rfloor)$. Moreover, $\mathsf{hc}_{K'}^S \lfloor F \rfloor \subseteq \mathsf{pow}\ S$, so $\mathsf{hc}_{K'}^S \lfloor F \rfloor \in \mathsf{uce}\ F_c$. Then, $\bar{w}_{(\bigcup F_c)}(\mathsf{hc}_{K'}^S \lfloor F \rfloor) \geq 0$ holds from the assumption. However, since w is zero on K, it holds that $w(\bigcup F_c) = w(\bigcup F)$ and $\bar{w}_{(\bigcup F)}(\mathsf{hc}_{K'}^S \lfloor F \rfloor) = \bar{w}_{(\bigcup F_c)}(\mathsf{hc}_{K'}^S \lfloor F \rfloor) \geq 0$

Theorem 1. *A family F_c is an FC-family if there is a weight function w such that shares (wrt. w and $\bigcup F_c$) of all union closed extension of F_c are nonnegative.*

Proof. Consider a union-closed family $F \supseteq F_c$. Let w be the weight function such that $\forall F' \in \mathsf{uce}\ F_c.\ \bar{w}_{(\bigcup F_c)}(F') \geq 0$. Let w' be a function equal to w on $\bigcup F_c$ and 0 on other elements. Since $\forall F' \in \mathsf{uce}\ F_c.\ \bar{w'}_{(\bigcup F_c)}(F') = \bar{w}_{(\bigcup F_c)}(F')$, Lemma 4 applies to F and F is Frankl's.

3 Combinatorial search

Theorem 1 inspires a procedure for verifying FC families. It should take a weight function on $\bigcup F_c$ and check that all union closed extensions of F_c have non-negative shares. We will now define a procedure *SomeShareNegative*, denoted by $\mathsf{ssn}\ F_c\ w$, such that if $\mathsf{ssn}\ F_c\ w = \bot$, then for all $F' \in \mathsf{uce}\ F_c$ it holds that $\bar{w}_{(\bigcup F_c)}(F') \geq 0$. The heart of this procedure will be a recursive function $\mathsf{ssn}^{F_c,w,X}\ L\ F_t$ that preforms a systematic traversal of all union closed extensions of F_c, but with pruning that speeds up the search. If a union closed extension of F_c has a negative share, it must contain one or more sets with a negative share. Therefore, a list L of all different subsets of $\bigcup F_c$ with negative shares is formed and each candidate family is determined by elements of L that it includes. A recursive procedure creates all candidate families by processing elements of L sequentially, either skipping them (in one recursive branch) or including them into the current candidate family F_t (in the other recursive branch), maintaining the invariant that the current candidate family F_t is always union closed. If the current element of L has been already included in F_t (by earlier closure operations required to maintain the invariant) the search can be pruned. If the sum of (negative) shares of the remaining elements of L is less then the (non-negative) share of the current F_t, then F_t cannot be extended to a family with a negative share (even in the extreme case when all the remaining elements of L are included) so, again, the search can be pruned.

Definition 9. *The function* $\mathsf{ssn}^{F_c,w,X}\ L\ F_t$ *is defined by a primitive recursion (over the structure of the list* L*):*

$$
\begin{aligned}
\mathsf{ssn}^{F_c,w,X}\ [\,]\ F_t &\equiv \bar{w}_X(F_t) < 0 \\
\mathsf{ssn}^{F_c,w,X}\ (h \mathbin{\#} t)\ F_t &\equiv \text{if } \bar{w}_X(F_t) + \sum_{A \in h \mathbin{\#} t} \bar{w}_X(A) \geq 0 \text{ then } \bot \\
&\quad\ \text{else if } \mathsf{ssn}^{F_c,w,X}\ t\ F_t \text{ then } \top \\
&\quad\ \text{else if } h \in F_t \text{ then } \bot \\
&\quad\ \text{else } \mathsf{ssn}^{F_c,w,X}\ t\ (\mathsf{ic}_{F_c}\ h\ F_t)
\end{aligned}
$$

Let L *be a distinct list such that its set is* $\{A.\ A \in \mathsf{pow}\ \bigcup F_c \wedge \bar{w}_X(A) < 0\}$*.*

$$\mathsf{ssn}\ F_c\ w \equiv \mathsf{ssn}^{\langle F_c\rangle,w,(\bigcup F_c)}\ L\ \emptyset$$

Next we prove the soundnes of the $\mathsf{ssn}\ F_c\ w$ function.

Lemma 5. *If (i)* $\mathsf{ssn}^{F_c,w,X}\ L\ F_t = \bot$*, (ii) for all elements* A *in* L *it holds that* $\bar{w}_X(A) < 0$*, (iii) for all* $A \in F' - F_t$*, if* $\bar{w}_X(A) < 0$*, then* A *is in* L*, (iv)* $F' \supseteq F_t$*, and (v)* $\mathsf{uc}_{F_c}\ F'$*, then* $\bar{w}_X(F') \geq 0$*.*

Proof. The proof is by induction. First, note that

$$\bar{w}_X(F') = \sum_{A \in F'} \bar{w}_X(A) = \sum_{A \in F_t} \bar{w}_X(A) + \sum_{A \in F' - F_t} \bar{w}_X(A). \tag{1}$$

Consider the base case of $L = []$. Since $\mathsf{ssn}^{F_c,w,X}\ []\ F_t = \bot$, it holds that $\sum_{A \in F_t} \bar{w}_X(A) = \bar{w}_X(F_t) \geq 0$ and first term in (1) is nonnegative. If there were some $A \in F' - F_t$ such that $\bar{w}_X(A) < 0$, then, from the assumptions it would be in L, which is impossible since L is empty. Therefore, the second term in (1) is also nonnegative which completes the proof.

Consider the inductive step, and assume that $L \equiv h \# t$.

First consider the case when $\bar{w}_X(F_t) + \sum_{A \in h \# t} \bar{w}_X(A) \geq 0$. Let P denote the set $\{A.\ A \in F' - F_t \wedge \bar{w}_X(A) \geq 0\}$, and let N denote the set $\{A.\ A \in F' - F_t \wedge \bar{w}_X(A) < 0\}$. Since, by assumptions, all elements of N are in $L \equiv h \# t$, and since, by assumptions, all shares of $h \# t - N$ are negative, it holds that

$$\sum_{A \in h \# t} \bar{w}_X(A) = \sum_{A \in N} \bar{w}_X(A) + \sum_{A \in h \# t - N} \bar{w}_X(A) \leq \sum_{A \in N} \bar{w}_X(A). \tag{2}$$

It holds that $\sum_{A \in F' - F_t} \bar{w}_X(A) = \sum_{A \in P} \bar{w}_X(A) + \sum_{A \in N} \bar{w}_X(A)$. Therefore, since all shares of P are nonnegative, from (1) and (2) and the assumption of the current case it holds that

$$\bar{w}_X(F') \geq \sum_{A \in F_t} \bar{w}_X(A) + \sum_{A \in N} \bar{w}_X(A) \geq \bar{w}_X(F_t) + \sum_{A \in h \# t} \bar{w}_X(A) \geq 0.$$

Next, consider the case when $\bar{w}_X(F_t) + \sum_{A \in h \# t} \bar{w}_X(A) < 0$. Since, by assumptions, $\mathsf{ssn}^{F_c,w,X}\ (h \# t)\ F_t = \bot$, by the definition of ssn it must hold that $\mathsf{ssn}^{F_c,w,X}\ t\ F_t = \bot$.

Consider the case when $h \in F_t$ or $h \notin F'$. Then $h \notin F' - F_t$. The conclusion follows by induction hypothesis for the recursive call $\mathsf{ssn}^{F_c,w,X}\ t\ F_t$, since all assumptions are satisfied. Indeed, all elements of $F' - F_t$ with negative shares must be in t, since $h \notin F' - F_t$, and other assumptions are trivially satisfied.

Finally, consider the case when $h \notin F_t$ and $h \in F'$. The conclusion follows by induction hypothesis for the recursive call $\mathsf{ssn}^{F_c,w,X}\ t\ (\mathsf{ic}_{F_c}\ h\ F_t)$, since all assumptions are satisfied for this call. Indeed, in this case $\mathsf{ssn}^{F_c,w,X}\ (h \# t)\ F_t = \mathsf{ssn}^{F_c,w,X}\ t\ (\mathsf{ic}_{F_c}\ h\ F_t)$ and the left hand side is $\bot$ from the current assumptions. All elements of $F' - \mathsf{ic}_{F_c}\ h\ F_t$ with negative shares must be in t. Indeed, this holds since $F_t \subseteq \mathsf{ic}_{F_c}\ h\ F_t$, and $h \in \mathsf{ic}_{F_c}\ h\ F_t$, and since all elements of $F' - F_t$ with negative shares are in $h \# t$. It holds that $\mathsf{ic}_{F_c}\ h\ F_t \subseteq F'$ since $F_t \subseteq F'$, $h \in F'$ and $\mathsf{uc}_{F_c}\ F'$. Other assumptions trivially hold.

Theorem 2. *If* $\mathsf{ssn}\ F_c\ w = \bot$ *and* $F' \in \mathsf{uce}\ F_c$ *then* $\bar{w}_{(\bigcup F_c)}(F') \geq 0$.

Proof. Fix F' from $\mathsf{uce}\ F_c$. Then $F' \subseteq \mathsf{pow}\ \bigcup F_c$ and $\mathsf{uc}_{F_c}\ F'$. Let L be a distinct list such that its set is $\{A.\ A \in \mathsf{pow}\ \bigcup F_c \wedge \bar{w}_X(A) < 0\}$. From $\mathsf{ssn}\ F_c\ w = \bot$ and the definition of ssn it holds that $\mathsf{ssn}^{\langle F_c \rangle, w, (\bigcup F_c)}\ L\ \emptyset = \bot$. All assumptions of Lemma 5 apply. Indeed, for all A in L, $\bar{w}_{(\bigcup F_c)}(A) < 0$. For all A in $F' - \emptyset$, if $\bar{w}_{(\bigcup F_c)}(A) < 0$, then, since $F' \subseteq \mathsf{pow}\ \bigcup F_c$, A is in L. $\emptyset \subseteq F'$. Since $\mathsf{uc}_{F_c}\ F'$, by Proposition 1, it holds that $\mathsf{uc}_{\langle F_c \rangle}\ F'$. Therefore, $\bar{w}_{(\bigcup F_c)}(F') \geq 0$ holds.

Apart from being sound, the procedure can also be shown to be complete. Namely, it could be shown that if $\mathsf{ssn}\ F_c\ w = \top$, then there is an $F' \in \mathsf{uce}\ F_c$ such that $\bar{w}_{(\bigcup F_c)}(F') < 0$. This comes from the invariant that the current family F_t in the search is always in $\mathsf{uce}\ F_c$, which is maintained by taking the closure $\mathsf{ic}_{F_c}\ h\ F_t$ whenever an element h is added. Since this aspect of the procedure is not relevant for the rest of the proofs, it will not be formally stated nor proved.

3.1 Efficient implementation

In order to obtain executability and increase efficiency, a series of refinements of $\mathsf{ssn}\ F\ w$ is done. Each refined version introduces a new implementation feature that makes it more efficient than the previous one, but still equivalent with it.

First, a function cannot operate on families of sets. Without loss of generality, it suffices only to consider families of sets of natural numbers. Sets of natural numbers are represented by natural number codes. A set A is represented by the code $\tilde{A} = \sum_{k \in A} 2^k$. Families of sets of natural numbers F are represented by (distinct) lists of natural number codes $\tilde{F}$. This representation will be referred to as *list-of-nats* representation (e.g., $F = \{\{0,1\},\{1,2\},\{0,1,2\}\}$ is represented by the list-of-nats $\tilde{F} = [3,6,7]$). Basic set operations have their corresponding list-of-nat counterparts.

- The union of two sets $\cup$ corresponds to bitwise disjunction (denoted by $\sqcup$). It holds that if $C = A\ \cup\ B$, then $\tilde{C} = \tilde{A}\ \sqcup\ \tilde{B}$.
- Adding a set A to a family of sets F (i.e., $A \cup F$) corresponds to the operation (also denoted by $\sqcup$) that prepends $\tilde{A}$ to $\tilde{F}$, but only if it is not already present, i.e., by: if $\tilde{A} \in \tilde{F}$ then $\tilde{F}$ else $\tilde{A}\ \#\ \tilde{F}$. It holds that if $F' = A \cup F$, then $\tilde{F'} = \tilde{A} \sqcup \tilde{F}$.
- Union of two families (i.e., $F' \cup F$), also denoted by $\sqcup$, is performed by iteratively adding sets from one family to another, i.e., as $\mathsf{foldl}\ (\lambda\ \tilde{A}\ \tilde{F}.\ \tilde{A} \sqcup \tilde{F})\ \tilde{F}\ \tilde{F'}$. It holds that if $F'' = F \cup F'$, then $\tilde{F''} = \tilde{F} \sqcup \tilde{F'}$.
- Adding a set A to all members of a family of sets F (i.e., $\{A \cup B.\ B \in F\}$), denoted by $[\tilde{A}\ \sqcup\ \tilde{B}.\ \tilde{B} \in \tilde{F}]$, is performed by $\mathsf{map}\ (\lambda\ \tilde{B}.\ \tilde{A} \sqcup \tilde{B})\ \tilde{F}$. It holds that if $F' = \{A \cup B.\ B \in F\}$, then $\tilde{F'} = [\tilde{A}\ \sqcup\ \tilde{B}.\ \tilde{B} \in \tilde{F}]$.
- Insert and close for F (i.e., $\mathsf{ic}_{F_c}\ a\ F$), denoted by $\tilde{\mathsf{ic}}$, is computed as $([\tilde{A}]\ @\ [\tilde{A} \sqcup \tilde{B}.\ \tilde{B} \in \tilde{F}]\ @\ [\tilde{A}\ \sqcup\ \tilde{B}.\ \tilde{B} \in \tilde{F_c}])\ \sqcup\ \tilde{F}$. It holds that if $F' = \mathsf{ic}_{F_c}\ a\ F$, then $\tilde{F'} = \tilde{\mathsf{ic}}_{\tilde{F_c}}\ \tilde{a}\ \tilde{F}$.

Important optimization to the basic $\mathsf{ssn}\ F_c\ w$ procedure is to avoid repeated computations of family shares (both for the elements of the list L and the current family F_t). So, instead of accepting a list of families of sets L, and the current family of sets F_t, the function is modified to accept a list of ordered pairs where first component is a list-of-nats representation of corresponding element of L, and the second component is its share (wrt. w and X), and to accept an ordered pair $(\tilde{F_t}, s_t)$ where $\tilde{F_t}$ is the list-of-nats representation of F_t, and s_t is its family share (wrt. w and X). The summation of shares of elements in L is also unnecessarily repeated. It can be avoided if the sum (s_l) is passed trough the function.

$$
\begin{aligned}
\mathsf{ssn}^{\tilde{F_c},w,X}\ ([\,],0)\ (\tilde{F_t},s_t) &\equiv s_t < 0 \\
\mathsf{ssn}^{\tilde{F_c},w,X}\ ((\tilde{h},s_h)\ \#\ t,\ s_l)\ (\tilde{F_t},\ s_t) &\equiv \text{if } s_t + s_l \geq 0 \text{ then } \bot \\
&\quad \text{else if } \mathsf{ssn}^{\tilde{F_c},w,X}\ (t,\ s_l - s_h)\ (\tilde{F_t},\ s_t) \text{ then } \top \\
&\quad \text{else if } \tilde{h} \in \tilde{F_t} \text{ then } \bot \\
&\quad \text{else let } \tilde{F_t}' = \tilde{\mathsf{ic}}_{\tilde{F_c}}\ \tilde{h}\ \tilde{F_t};\ s_t' = \bar{w}_X(\tilde{F_t}') \text{ in} \\
&\qquad \mathsf{ssn}^{\tilde{F_c},w,X}\ (t, ls - s_h)\ (\tilde{F_t}', s_t')
\end{aligned}
$$

Another source of inefficiency is the calculation of $\bar{w}_X(\tilde{F_t}')$. If performed directly based on the definition of family share for $\tilde{F_t}'$, the sum would contain shares of all elements from $\tilde{F_t}$ and of all elements that are added to $\tilde{F_t}$ when adding $\tilde{h}$ and closing for $\tilde{F}$. However, it is already known that the sum of shares for elements of $\tilde{F_t}$ is s_t and the implementation could benefit from this fact. Also, calculating shares of sets that are added to $\tilde{F_t}$ can be made faster. Namely, it happens that set share of a same set is calculated over and over again in different parts of the search space. So, it is much better to precompute shares of all sets from $\mathsf{pow}\ X$ and store them in a lookup table that will be consulted each time a set share is needed. Note that in this case there is no more need to pass the function w itself, nor the domain X, but only the lookup table, denoted by s_w.

$$
\begin{aligned}
\mathsf{ssn}^{\tilde{F_c},s_w}\ ([\,],0)\ (\tilde{F_t},s_t) &\equiv s_t < 0 \\
\mathsf{ssn}^{\tilde{F_c},s_w}\ ((\tilde{h},s_h)\ \#\ t,\ s_l)\ (\tilde{F_t},\ s_t) &\equiv \text{if } s_t + s_l \geq 0 \text{ then } \bot \\
&\quad \text{else if } \mathsf{ssn}^{\tilde{F_c},s_w}\ (t,\ s_l - s_h)\ (\tilde{F_t},\ s_t) \text{ then } \top \\
&\quad \text{else if } \tilde{h} \in \tilde{F_t} \text{ then } \bot \\
&\quad \text{else } \mathsf{ssn}^{\tilde{F_c},s_w}\ (t, s_l - s_h)\ (\tilde{\mathsf{ic}}^{s_w}_{\tilde{F_c}}\ \tilde{h}\ (\tilde{F_t}, s_t)) \\
\tilde{\mathsf{ic}}^{s_w}_{\tilde{F_c}}\ \tilde{h}\ (\tilde{F_t},s_t) &\equiv \text{let } \ add\ = [\tilde{h}]\ @\ [\tilde{h}\ \sqcup\ \tilde{A}.\ \tilde{A} \in \tilde{F_t}]\ @\ [\tilde{h}\ \sqcup\ \tilde{A}.\ \tilde{A} \in \tilde{F_c}]; \\
&\qquad\quad add\ = \mathsf{filter}\ (\lambda \tilde{A}.\ \tilde{A} \notin \tilde{F})\ (\mathsf{remdups}\ add) \text{ in} \\
&\quad (add\ @\ \tilde{F},\ \ s + \mathsf{listsum}\ (\mathsf{map}\ s_w\ add))
\end{aligned}
$$

It is shown that this implementation is (in some sense) equivalent to the starting, abstract one. This proof is technically involved, but conceptually uninteresting so we omit it in the text.

4 Uniform *nkm*-families

Most FC-families that are considered in this paper are *uniform*, i.e., consist of sets having the same number of elements.

Definition 10. *A family of sets F is a* uniform *nkm*-family *if it contains m different sets, each containing k elements and their union has at most n elements. Uniform nkm-family is* natural *if its union is contained in $\{0, 1, \ldots, n-1\}$.*

Within the Isabelle/HOL implementation, natural nkm-families will be represented by *nkm-lists* — (lexicografically) sorted, distinct lists of length m containing sorted, distinct lists of length k with all elements contained in $\{0, 1, \ldots, n-1\}$. To simplify presentation, we will identify natural nkm-families with their corresponding nkm-lists. Assuming that the Isabelle/HOL function $\mathsf{comb}\ l\ k$ generates all sorted k-element sublists of a sorted list l, all nkm-lists for given n, k and m can be generated by $\mathsf{fams}^{nkm} \equiv \mathsf{comb}\ (\mathsf{comb}\ [0..< n]\ k)\ m$.

Symmetries. Often one uniform nkm-family can be obtained from the other by permuting its elements (e.g., $\{\{a_0, a_1, a_2\}, \{a_1, a_3, a_4\}, \{a_2, a_3, a_4\}\}$ can be obtained from $\{\{a_0, a_1, a_2\}, \{a_0, a_1, a_3\}, \{a_2, a_3, a_4\}\}$ by the permutation $(a_0, a_1, a_2, a_3, a_4) \mapsto (a_3, a_4, a_1, a_2, a_0)$). Applying permutations on sets and families can be implemented in Isabelle/HOL by the functions $\mathsf{perm\_set}\ A\ p \equiv \mathsf{sort}\ (\mathsf{map}\ (\lambda x.\ p_{[x]})\ A)$ and $\mathsf{perm\_fam}\ F\ p \equiv \mathsf{sort}\ (\mathsf{map}\ \mathsf{perm\_set}\ F)$. Permutations establish bijections between natural uniform families:

Proposition 7. *If p is a permutation of $[0, 1, \ldots, n-1]$ and F is a natural uniform family, then $\mathsf{perm\_fam}\ F\ p$ is also natural uniform family and there is a bijection between F and $\mathsf{perm\_fam}\ F\ p$.*

Since, by Proposition 2, FC-families are preserved under bijections (isomorphisms), to check if all elements of a given list of nkm-families $\mathcal{F}$ are FC-families, many elements need not be considered. Indeed, it suffices to consider only a list (denoted by $\mathsf{nef}^P\ \mathcal{F}$) of its non-equivalent representatives (under a given list of permutations P). Computation of such representatives can start from the given list $\mathcal{F}$, choose its arbitrary member for a representative, remove it and all its permuted variants from the lists, and repeat this sieving process until the list becomes empty. Isabelle/HOL implementation of this procedure can be given by:

$$
\begin{aligned}
\mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r \equiv\ & \text{case } \mathcal{F} \text{ of } [\,] \Rightarrow \mathcal{F}_r \\
& \mid\ F \# \_ \Rightarrow \text{let } \mathcal{F}_F^P = \mathsf{remdups}\ (\mathsf{map}\ (\lambda\ p.\ \mathsf{perm\_fam}\ F\ p)\ P) \text{ in} \\
& \qquad \mathsf{nef\_aux}^P\ (\mathsf{filter}\ (\lambda\ F.\ F \notin \mathcal{F}_F^P)\ \mathcal{F})\ (F \# \mathcal{F}_r) \\
\mathsf{nef}^P\ \mathcal{F} \equiv\ & \mathsf{nef\_aux}^P\ \mathcal{F}\ [\,]
\end{aligned}
$$

The following lemma proves the correctness of this implementation.

Lemma 6. *If P is a list of permutations of $[0, 1, \ldots, n-1]$ and if $\mathcal{F}$ is a list of natural nkm-families, then for each element $F \in \mathcal{F}$ there is an $F' \in \mathsf{nef}^P\ \mathcal{F}$ such there is a bijection between F and F'.*

Proof. First, note that the function $\mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r$ is monotone, i.e., $\mathcal{F}_r \subseteq \mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r$.

By induction, we show that if the assumptions hold for $\mathcal{F}$ and P, then for each element $F \in \mathcal{F}$ there is an element $F' \in \mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r$ such there is a bijection between F and F'.

In the base case, when $\mathcal{F}$ is empty, the statement trivially holds.

Assume that $\mathcal{F} \equiv F \# \mathcal{F}'$. Let $\mathcal{F}_F^P$ denote all different families obtained by permuting F by all elements of P (i.e., $\mathcal{F}_F^P \equiv \mathsf{remdups}\ (\mathsf{map}\ (\lambda\, p.\ \mathsf{perm\_fam}\ F\ p)\ P))$ and let $\mathcal{F}^-$ denote what remains of $\mathcal{F}$ when those are removed (i.e., $\mathcal{F}^- \equiv \mathsf{filter}\ (\lambda\ F.\ F \notin \mathcal{F}_F^P)\ \mathcal{F}$. It holds that $\mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r = \mathsf{nef\_aux}^P\ \mathcal{F}^-\ (F \# \mathcal{F}_r)$.

Let F' be an arbitrary element from $\mathcal{F}$. Since $\mathcal{F} = F \# \mathcal{F}'$, either $F' = F$ or $F' \in \mathcal{F}'$.

Assume that $F' = F$. By monotonicity it holds that $F \in \mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r$, so F is an element from $\mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r$ such that there is a bijection (identity function) between F' and it.

Assume that $F' \in \mathcal{F}'$.

Consider the case when $F' \in \mathcal{F}_F^P$. Then there is $p \in P$ such that $F' = \mathsf{perm\_fam}\ F\ p$. Since $F' \in \mathcal{F}$ is natural and $p \in P$ is a permutation of $[0, 1, \ldots, n-1]$, by Proposition 7, there is a bijection between F and F'. Since, by monotonicity, it holds that $F \in \mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r$, F is an element in $\mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r$ such that there is a bijection between F' and it.

Consider the case when $F' \notin \mathcal{F}_F^P$. Then $F' \in \mathcal{F}^-$. By inductive hypothesis for the call $\mathsf{nef\_aux}^P\ \mathcal{F}^-\ (F \# \mathcal{F}_r)$, there is an element F'' in $F \# \mathcal{F}_r$ such that there is a bijection between F' and it. By monotonicity, $F'' \in F \# \mathcal{F}_r \subseteq \mathsf{nef\_aux}^P\ \mathcal{F}^-\ (F \# \mathcal{F}_r) = \mathsf{nef\_aux}^P\ \mathcal{F}\ \mathcal{F}_r$, so the statement holds.

Finally, the following lemma shows that only non-equivalent representatives need to be considered when checking FC-families.

Lemma 7. *Let* $\mathcal{F} \subseteq \mathsf{fams}^{nkm}$ *and* $P \subseteq \mathsf{perm}\ [0, 1, \ldots, n-1]$. *If all families represented by elements of* $\mathsf{nef}^P\ \mathcal{F}$ *are FC-families, then all families represented by elements of* fams^{nkm} *are FC-families.*

Proof. Let $F \in \mathsf{fams}^{nkm}$. By Lemma 6 there is an $F' \in \mathsf{nef}^P\ \mathcal{F}$ and a bijection between F and F'. So, F' is an FC-family, and by Proposition 2, so is F.

5 FC-families verified

Having established all the necessary mathematics, in this Section we prove that certain uniform families are FC-families (mainly by performing verified calculations). First, we calculate non-equivalent representatives for fams^{533}, fams^{634}, and fams^{734}.

Lemma 8. *The first column of Table 1 contains (respectively) all elements of:*

$\mathsf{nef}^{\mathsf{perm}\ [0..<5]}\ \mathsf{fams}^{533}$,

$\mathsf{nef}^{\mathsf{perm}\ [0..<6]}\ (\mathsf{filter}\ (\lambda F.\ \neg\mathsf{check}_{533}\ \mathsf{F})\ \mathsf{fams}^{634})$,

$\mathsf{nef}^{\mathsf{perm}\ [0..<7]}\ (\mathsf{filter}\ (\lambda F.\ \neg\mathsf{check}_{533}\ F \wedge \neg\mathsf{check}_{634}\ F)\ \mathsf{fams}^{734})$,

where $\mathsf{perm}\ l$ *is the function that generates all permutations of a list* l*,* check_{533} *is a function that checks if any 3 of the 4 given 3-element sets are have their union contained in a 5-element set, and* check_{634} *is a function that checks if the union of 4 given 3-element sets is contained in a 6-element set.*[7]

[7] Formal definition of these functions is not given here and is available in the Isabelle/HOL proof documents, along with correctness arguments.

F_c	w
$[[0,1]]$	$0 \mapsto 1, 1 \mapsto 1$
$[[0,1,2],[0,1,3],[2,3,4]]$	$0 \mapsto 2, 1 \mapsto 2, 2 \mapsto 2, 3 \mapsto 2, 4 \mapsto 1$
$[[0,1,2],[0,1,3],[0,2,4]]$	$0 \mapsto 6, 1 \mapsto 5, 2 \mapsto 5, 3 \mapsto 3, 4 \mapsto 3$
$[[0,1,2],[0,1,3],[0,2,3]]$	$0 \mapsto 1, 1 \mapsto 1, 2 \mapsto 1, 3 \mapsto 1$
$[[0,1,2],[0,1,3],[0,1,4]]$	$0 \mapsto 3, 1 \mapsto 3, 2 \mapsto 2, 3 \mapsto 2, 4 \mapsto 2$
$[[0,1,2],[0,3,4],[1,3,5],[2,4,5]]$	$0 \mapsto 1, 1 \mapsto 1, 2 \mapsto 1, 3 \mapsto 1, 4 \mapsto 1, 5 \mapsto 1$
$[[0,1,2],[0,1,3],[2,4,5],[3,4,5]]$	$0 \mapsto 1, 1 \mapsto 1, 2 \mapsto 1, 3 \mapsto 1, 4 \mapsto 1, 5 \mapsto 1$
$[[0,1,2],[0,3,4],[1,3,5],[2,4,6]]$	$0 \mapsto 2, 1 \mapsto 2, 2 \mapsto 2, 3 \mapsto 2, 4 \mapsto 2, 5 \mapsto 1, 6 \mapsto 1$
$[[0,1,2],[0,3,4],[0,5,6],[1,3,5]]$	$0 \mapsto 2, 1 \mapsto 1, 2 \mapsto 1, 3 \mapsto 1, 4 \mapsto 1, 5 \mapsto 1, 6 \mapsto 1$
$[[0,1,2],[0,1,3],[2,4,5],[4,5,6]]$	$0 \mapsto 3, 1 \mapsto 3, 2 \mapsto 4, 3 \mapsto 2, 4 \mapsto 3, 5 \mapsto 3, 6 \mapsto 2$
$[[0,1,2],[0,1,3],[2,4,5],[3,4,6]]$	$0 \mapsto 3, 1 \mapsto 3, 2 \mapsto 3, 3 \mapsto 3, 4 \mapsto 2, 5 \mapsto 1, 6 \mapsto 1$
$[[0,1,2],[0,1,3],[0,4,5],[4,5,6]]$	$0 \mapsto 6, 1 \mapsto 4, 2 \mapsto 3, 3 \mapsto 3, 4 \mapsto 4, 5 \mapsto 4, 6 \mapsto 2$
$[[0,1,2],[0,1,3],[0,4,5],[2,4,6]]$	$0 \mapsto 3, 1 \mapsto 2, 2 \mapsto 3, 3 \mapsto 1, 4 \mapsto 3, 5 \mapsto 2, 6 \mapsto 2$
$[[0,1,2],[0,1,3],[0,4,5],[1,4,6]]$	$0 \mapsto 2, 1 \mapsto 2, 2 \mapsto 1, 3 \mapsto 1, 4 \mapsto 1, 5 \mapsto 1, 6 \mapsto 1$
$[[0,1,2],[0,1,3],[0,4,5],[0,4,6]]$	$0 \mapsto 2, 1 \mapsto 1, 2 \mapsto 1, 3 \mapsto 1, 4 \mapsto 1, 5 \mapsto 1, 6 \mapsto 1$

Table 1. Families and weights

Proof. By calculations performed by a computer.

Next, we show that all these representatives have non-negative shares.

Lemma 9. *For all F_c and w given in Table 1, it holds that* $\mathsf{ssn}\ \tilde{F_c}\ w = \bot$.

Proof. By calculations performed by a computer.

Finally, the main result can be easily proved.

Theorem 3. *The following are FC-families:*

1. *all families containing one 1-element set (i.e., $\{\{a\}\}$);*
2. *all families containing one 2-element set (i.e., $\{\{a,b\}\}$, for $a \neq b$);*
3. *all families containing 3 3-element sets whose union is contained in a 5-element set (i.e., uniform* 533*-families);*
4. *all families containing 4 3-element sets whose union is contained in a 6-element set (i.e., uniform* 634*-families);*
5. *all families containing 4 3-element sets whose union is contained in a 7-element set (i.e., uniform* 734*-families).*

Proof. The case 1 trivially holds (since for each family member A that does not contain a, there is a member $A \cup \{a\}$ that contains a).

Other proofs are based on the techniques described in this paper. By Proposition 2 it suffices to consider only families F such that $\bigcup F \subseteq \{0, 1, \ldots, n-1\}$. All families corresponding to rows in Table 1 are FC-families. Indeed, for each F_c and w given in a table row, by Lemma 9 it holds that $\mathsf{ssn}\ F_c\ w$. Therefore, by Lemma 2 for all $F' \in \mathsf{uce}\ F_c$ it holds that $\bar{w}_{(\bigcup F_c)}(F') \geq 0$. Then, F_c is FC-family by Theorem 1.

In the case 2 this completes the proof.

In the case 3 the statement holds by Lemma 7, since, by Lemma 8 four rows given in Table 1 correspond to four non-equivalent families.

To show the case 4, let F_c be any family containing 4 3-element sets whose union is contained in $\{0, 1, \ldots, 5\}$ and let F be a union-closed family such that $F \supseteq F_c$. If $\mathsf{check}_{533}\ F_c$ holds (i.e., if union of any 3 members of F_c is contained in a 5-element set), then F is Frankl's by case 3. If $\neg\mathsf{check}_{533}\ F_c$ holds, then F_c is in $\mathsf{filter}\ (\lambda F. \neg\mathsf{check}_{533}\ F)\ \mathsf{fams}^{634}$. The statement then holds by Lemma 7, since, by Lemma 8 two rows given in Table 1 correspond to two non-equivalent families of $\mathsf{filter}\ (\lambda F. \neg\mathsf{check}_{533}\ F)\ \mathsf{fams}^{634}$.

The case 5 is proved similarly, using the proofs for both the case 3 and the case 4.

6 Conclusions and further work

In this paper, we have formalized (within Isabelle/HOL) a computer-assisted approach of Živković and Vučković for verifying FC-families. Well-known FC-families are confirmed and a new uniform FC-family is discovered.

The Isabelle/HOL formalization has around 260KB of data organized into around 6500 lines of Isabelle/Isar proof text. Ratio between the size of the formalization and the size of the corresponding pen and paper proof (DeBruijn index) is estimated at around 5.5. Total time required to do the formalization is very roughly estimated at around 200 man/hours (25 full working days spread over a period of around 8 months).

Total proof checking time of Isabelle/HOL takes around 28 minutes on a notebook PC with 2.1GHz Intel/Pentium CPU and 4GB RAM. The major fraction of this time (around 23 minutes) is spent in the combinatorial search. Checking Lemma 9 consumes most of this time, and its last 8 cases (related to the uniform-734 families) alone take 22.8 minutes. This is quite long compared to the original JAVA programs (that perform the whole combinatorial search in around 1 minute), but still bearable. The big difference is due to the use of machine-integers supporting atomic bitwise-or in JAVA and the use of big-integers that do not support atomic bitwise-or in Isabelle/ML. The search time could be reduced if machine-integers were also used in Isabelle/ML. In a simple approach, the code generator could be instructed to replace mathematical integers in the formalization by machine-integers in the code, but that would make a gap between the formalization and the generated code and would require trusting that no overflows occur. A better approach would require formalizing machine-integers and their properties and using them within the formalization itself.

Compared to the prior pen-and-paper work, the computer assisted approach significantly reduces the complexity of mathematical arguments behind the proof and employs computing-machinery in doing its best — quickly enumerating and checking a large search space. This enables formulation of a general framework for checking various FC-families, without the need of employing human intellectual resources in analyzing specificities of separate families. Compared to the work of Živković and Vučković, apart from achieving the highest level of trust possible, the significant contribution of the formalization is the clear separation of mathematical background and combinatorial search algorithms, not present in earlier work. Also, separation of abstract properties of search algorithms and technical details of their implementation significantly simplifies reasoning about their correctness and brings them much closer to classic mathematical audience, not inclined towards computer science.

This work represents a significant part in formally proving the Frankl's conjecture for families F such that $|\bigcup F| \leq 11$, and $|\bigcup F| \leq 12$ (already informally done by Živković and Vučković [17]) which in the focus of our current and future work. We also plan to investigate other FC-families (not necessarily uniform).

References

1. Tetsuya Abe. Strong Semimodular Lattices and Frankl's Conjecture. *Algebra Universalis*, 44:379–382, 2000.
2. Kenneth I. Appel and Wolfgang Haken. *Every Planar Map is Four Colorable.* American Mathematical Society, 1989.
3. Ivica Bošnjak and Petar Marković. The 11-element Case of Frankl's Conjecture. *Electronic Journal of Combinatorics*, 15(1), 2008.
4. Giovanni Lo Faro. Union-closed Sets Conjecture: Improved Bounds. *J. Combin. Math. Combin. Comput.*, 16:97–102, 1994.
5. Weidong Gao and Hongquan Yu. Note on the Union-Closed Sets Conjecture. *Ars Combinatorica*, 49, 1998.
6. Georges Gonthier. Formal Proof – the Four-Color Theorem. *Notices of AMS*, 55(11), 2008.
7. Petar Marković. An attempt at Frankl's Conjecture. *Publications de l'Institut Mathmatique*, 81(95):29–43, 2007.
8. Robert Morris. FC-families and Improved Bounds for Frankls Conjecture. *European Journal of Combinatorics*, 27(2):269 – 282, 2006.
9. Tobias Nipkow, Gertrud Bauer, and Paula Schultz. Flyspeck I: Tame Graphs. In Ulrich Furbach and Natarajan Shankar, editors, *IJCAR*, volume 4130 of *LNCS*, pages 21–35. Springer, 2006.
10. Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.
11. Bjorn Poonen. Union-closed Families. *Journal of Combinatorial Theory, Series A*, 59(2):253 – 268, 1992.
12. Jürgen Reinhold. Frankl's Conjecture is True for Lower Semimodular Lattices. *Graphs and Combinatorics*, 16:115–116, 2000.
13. N. Robertson, D. P. Sanders, P. D. Seymour, and R. Thomas. The Four Colour Theorem. *Journal of Combinatorial Theory, Series B*, 1997.
14. Theresa P. Vaughan. Families Implying the Frankl Conjecture. *European Journal of Combinatorics*, 23(7):851 – 860, 2002.
15. Theresa P. Vaughan. A Note on the Union-closed Sets Conjecture. *J. Combin. Math. Combin. Comput.*, 45:95–108, 2003.
16. Theresa P. Vaughan. Three-sets in a Union-closed Family. *J. Combin. Math. Combin. Comput.*, 49:95–108, 2004.
17. Miodrag Živković and Bojan Vučković. The 12-element Case of Frankls Conjecture. *submitted*, 2012.