Research Article
The Formalization of Discrete Fourier Transform in HOL

Zhiping Shi, 1 Yupeng Zhang, 1 Yong Guan, 1 Liming Li, 1 and Jie Zhang 2

1 Beijing Key Laboratory of Electronic System Reliability Technology, Capital Normal University, Beijing 100048, China
2 College of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029, China

Correspondence should be addressed to Zhiping Shi; shizp@cnu.edu.cn

Received 6 March 2015; Revised 7 June 2015; Accepted 25 June 2015

Academic Editor: Jean-François Monin

Copyright © 2015 Zhiping Shi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Traditionally, Discrete Fourier Transform (DFT) is performed with numerical or symbolic computation, which cannot guarantee 100% accurate analysis which may be necessary for safety-critical applications. Machine theorem proving is one of the formal methods that perform accurate analysis with completeness to some extent. This paper proposes the formalization of DFT in a higher-order logic theorem prover named HOL. We propose the formal definition of DFT and verify the fundamental properties of DFT. Two case studies are presented to illustrate usefulness and correctness of the formalized DFT, including formal verifications of Fast Fourier Transform (FFT) and cosine frequency shift.

1. Introduction

Fourier Transform (FT), through which signal is transformed from time domain to frequency one, is a fundamental and core method for signal processing. DFT is the discrete version of FT on both time domain and frequency domain which is adapted to the actual environment of computers and digital signal processing (DSP) chips. DFT has been widely used in many fields such as spectral analysis, data compressing, digital communication, signal processing, and the solution of partial differential equations [1, 2]. Therefore tremendous algorithms and packages have been implemented to perform DFT. The algorithms and packages are based on numerical or symbolic computation. The numerical computation intuitively produces approximate solutions because floating-point numbers are used to represent reals in computers. Computer algebra systems (CAS) such as MATLAB, Maple, and Mathematica offer a large collection of powerful algebraic packages and can efficiently perform symbolic computation precisely without round-off. But CAS still can output incorrect results because they have no mechanism to check correctness of results mathematically. Errors can arise in mismatching the side conditions of functions, the determination of equality of two expressions, or the definedness of expressions with respect to symbolic parameters [3, 4]. Moreover both numerical and symbolic computations are implemented in unverified huge computer programs, which are bug prone. Thus the analysis of DFT based on the above mentioned techniques cannot be relied upon for safety-critical applications of which even tiny errors can cause loss of human lives.

Formal methods have succeeded in precise analysis and verification of hardware and software systems in the past decades. Theorem proving, one of the formal methods, models the systems in logic and then reasons and verifies the model’s properties using axioms, theorems, and inference rules in the logic. The proof is performed in a theorem prover [5, 6]. The precise analysis and verification of systems employing DFT need the formal implementation of DFT in a theorem prover. Formal methods have been successfully applied in the precise analyses and verifications of hardware and software systems in the past decades. As a formal method, theorem proving can be used to model the systems based on logic and then to reason and verify the model’s properties using axioms, theorems, and inference rules in the logic. Generally, the proving tasks are performed in a theorem prover [5, 6]. For example, one needs to implement DFT formally in a theorem prover to perform the precise analyses and verifications based on DFT. In recent years, some
formalization and verification [7–10] have been done on Fast Fourier Transform (FFT), which is the most popular and efficient DFT method. All these works focused on the specific formal implementations of FFT at different levels in diverse algorithms. Bjesse [7] verified FFT hardware at the netlist level. Capretta [8] formalized FFT at the power list level in the Coq theorem prover, while Gamboa [9] performed similar work in ACL2. Akbarpour and Tahar [10] formally verified FFT at different abstraction levels including the real specification level, the floating and fixed-point description level, and the RT and netlist gate levels, and they also analyzed the round-off accumulation errors. Although FFT may be the most widely used implementation of DFT, it cannot take place of DFT completely. DFT has its own theoretical advantages and is much simpler than FFT in both theory and algorithm. Therefore, compared with FFT, DFT has been all along a dominant method in theoretical analysis, especially when theorem proving is used to analyze the functions and properties of a system model. The applications of DFT in theorem proving are very important in the early stages of engineering design. Generally, DFT can be adopted to perform real-time computation on local data in the process of sampling, while FFT computation can only be done when all the data is obtained. Besides, DFT has more advantages over FFT in the choice of sampling rate, the memory consumption of data, and so forth. To the best of our knowledge, no work has been done previously to formalize DFT in any theorem prover. In the present paper, DFT is for the first time formalized in the HOL theorem prover. To some extent, the present work is to develop a theoretical tool rather than to just provide an application. Systematic and detailed formalization of DFT is presented here to support flexible formal analyses of various systems.

The HOL system is a well-developed theorem prover from the University of Cambridge. It is equipped with the formalized mathematic theories needed for DFT such as real analysis [11], complex field [12], and matrix theory [13, 14]. The rest of the paper is organized as follows. After the formal definition of DFT in Section 2, the fundamental properties of DFT are formalized and verified in Section 3. And then in Section 4, two case studies are presented to illustrate the usefulness of the proposed formalization of DFT, including formal analysis of FFT and the cosine frequency shift. Section 5 concludes the paper.

2. Formal Definition of DFT

For a real or complex sequence \( x(n) \) with length of \( N \) in time domain, DFT transforms it to a new complex sequence \( X(k) \) with the same length in frequency domain. The mathematical definition of DFT can be written as follows:

\[
X(k) = \sum_{n=0}^{N-1} x(n) e^{-j(2\pi/N)nk}, \quad 0 \leq k \leq N - 1, \quad (1)
\]

where \( e^{-j(2\pi/N)nk} \), \( 0 \leq n, k \leq N - 1 \), is the frequency spectrum. For simplification, it is substituted by \( W_N^{nk} \) in the rest of the paper. The definition can be formalized in HOL as follows.

**Definition 1.** DFT\(_{def}\)

\[ \forall f \in \text{complex}. \quad \text{DFT} f N k = \text{csum}(0, N)(\lambda n. f n * e^{i(-2 * \pi N n k)}), \]

where \( \text{csum}(0, N) \) accumulates the complex sequence \( f \) from number 0 item for \( N \) items. The function \( \text{exp}(\cdot) \) is the base-\( e \) exponential function in which \( i \) stands for the imaginary unit and \( \pi \) stands for \( \pi \). The above mentioned symbols keep the same meanings in the rest of this paper.

When \( X = \text{DFT}(x) \), the inverse DFT (IDFT) can be expressed as follows:

\[
x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j(2\pi/N)nk}, \quad 0 \leq n \leq N - 1. \quad (2)
\]

The IDFT can be formalized as follows.

**Definition 2.** IDFT\(_{def}\)

\[ \forall g \in \text{complex}, n. \text{IDFT g N n} = \text{csum}(0, N)(\lambda k. g k * e^{i(2*\pi N n k)})/N, \]

where \( g \) stands for the DFT sequence of \( f \) with length of \( N \).

3. Formal Verification of the Properties

In this section, the classical properties of DFT are formalized and verified based on Definition 1 in HOL. The formal verification of the properties can check whether or not the definition is correct and reasonable, and moreover the verified properties turn into theorems in HOL and they can be utilized directly to facilitate formal analysis of DFT based systems.

3.1. Some Useful Lemmas. To conveniently read and understand formalizations of properties of DFT, some useful lemmas verified in HOL are presented in this subsection. Consider

\[
\forall r, f(r) = g(r) \implies \sum_{r=m+n}^{m+n} f(r) = \sum_{r=m+n}^{m+n} g(r). \quad (3)
\]

The formalization is as follows.

**Lemma 3.** Equality of two complex functions’ cumulative sum

\[
\forall (f : \text{num-} \to \text{complex}), (g : \text{num-} \to \text{complex}), (m : \text{num}), (n : \text{num}). (\forall (r : \text{num}), m \leq r \land r < m + n \implies (f r = g r)) \implies (\text{csum}(m, n) f = \text{csum}(m, n) g),
\]

where functions \( f \) and \( g \) are functions from the natural numbers to complex numbers for convenience to DFT.

Intuitively, \( W_N^{nk} \) is an \( N \)-periodic complex sequence. It holds that

\[
W_N^{nk} = W_N^{nk}. \quad (4)
\]

The formalization is as follows.
Lemma 4. Periodicity of $W_{nk}$ is

$$\forall k m n \ N \ N < 0 \land 0 \leq k \land k < N \Rightarrow (\exp(i \ast (-2 \ast \pi/N \ast n \ast (k + m \ast N))) = \exp(i \ast (-2 \ast \pi/N \ast n \ast k))).$$

For a complex number $z$, $IM z$ returns the imaginary component of $z$. One has

$$- (IM z) = IM (-z). \quad (5)$$

Lemma 5. Inverse operation of imaginary parts of complex numbers

$$\forall z : complex. - IM z = IM (-z).$$

Suppose $g$ is a complex function; it holds that

$$g(0) = 0 \land g(x + y) = g(x) + g(y) \Rightarrow g \left( \sum_{k=n}^{m-1} f(k) \right) = \sum_{k=n}^{m-1} g(f(k)). \quad (6)$$

The formalization is as follows.

Lemma 6. Compound operation and accumulating operation

$$\forall f g n m.(\forall x y.(g(0) = 0 \land (g(x + y) = g(x) + g(y))) \Rightarrow (g(\text{sum}(n, m) f) = \text{sum}(n, m)(\lambda k. g(f(k)))).$$

For the conjugate operator $*$ and accumulating function, there is a similar property as follows:

$$\left[ \sum_{k=n}^{m-1} f(k) \right]^* = \sum_{k=n}^{m-1} [f(k)]^*, \quad (7)$$

where $*$ stands for the conjugate operator of complex numbers. The formalization is as follows.

Lemma 7. Conjugate operation and accumulating operation

$$\forall f N . \text{conj}(\text{sum}(0, N)(\lambda n. f(n))) = \text{sum}(0, N)(\lambda n. \text{conj}(f(n))),$$

where the function conj returns the conjugate of a complex number.

Accumulation of a series of complex numbers can be performed on the real and imaginary parts, respectively. Consider

$$\sum z = \sum RE (z) + \sum IM (z), \quad (8)$$

where the functions $RE$ and $IM$ return the real and imaginary components of a complex number, respectively. The formalization is as follows.

Lemma 8. Accumulating operation of complex numbers

$$\forall N : num(z w : num \rightarrow real). \text{sum}(0, N)(\lambda n. (z n, w n)) = (\text{sum}(0, N)(\lambda n. (z n)), \text{sum}(0, N)(\lambda n. (w n))).$$

3.2. Implicit Periodicity. DFT is derived from the Discrete Fourier Series. Considering $W_{nk}^N$ is an $N$-periodic sequence, suppose $x(\cdot)$ is a series with $N$-periodic extension. Then for $X = \text{DFT}(x)$, $X$ is periodic. One has

$$X(k + mN) = X(k), \quad 0 \leq k \leq N - 1, \ m \in Z. \quad (9)$$

The formal description is as follows.

Theorem 9. $\text{DFT\_PERIODIC}$

$$\forall k m n : num. ((N) : real <> (0) \land (0 \leq k \land k < N)) \Rightarrow (\text{DFT}(f : num -> real)(N : num)(k + m \ast N) = \text{DFT}(f : num \rightarrow real)(N : num)k).$$

3.3. Linearity. For two $N$-length sequences $x_1(n)$ and $x_2(n)$, $X_1(k)$ and $X_2(k)$ are their DFT sequences. The linearity is as follows:

$$\text{DFT}(ax_1(n) + bx_2(n)) = aX_1(k) + bX_2(k). \quad (10)$$

We verified this property as the following theorem.

Theorem 10. $\text{DFT\_LINEAR}$

$$\forall f_1 f_2 f_3 a b k N.(\forall n.0 \leq n \land n < N \Rightarrow (f_3 n = a \ast f_1 n + b \ast f_2 n)) \Rightarrow (\text{DFT}(f_3 N k = a \ast \text{DFT} f_1 N k + b \ast \text{DFT} f_2 N k).$$

The linearity of DFT shows that it can be applied in discrete linear system. The proof is based on Definition 1 and Lemma 3.

3.4. Symmetry. The time domain and frequency domain of DFT are conjugated dually and written as

$$X^* (-k)_N = \text{DFT}(X^*(n)). \quad (11)$$

The formal description is as follows.

Theorem 11. $\text{DFT\_SYMM}$

$$\forall N : num. ((N) : real <> (0) \land (0 \leq k \land k < N)) \Rightarrow (\text{conj}(\text{sum}(0, N)(\lambda n. f(n))) = (\text{sum}(0, N)(\lambda n. \text{conj}(f(n))))).$$

During verifying the theorem, we need to introduce a subgoal:

$$\forall n : num. \text{conj}(\exp(i \ast (k \ast 2 \ast \pi \ast \text{inv}(N) \ast n))) = \exp(i \ast (k \ast -2 \ast \pi \ast \text{inv}(N) \ast n)) \quad (12)$$

in order to simplify the goal. Then apply some general operation rules in real script and Lemma 3 to make this theorem proved.

3.5. Conjugated Symmetry. The sequence after DFT is of conjugated symmetry. Consider

$$X^* (k) = X[(−k)]_N. \quad (13)$$

The formal description is as follows.
4. Mathematical Problems in Engineering

**Theorem 12.** DFT, CONJ, SYMM

\[ \forall k \in \mathbb{N}, \text{num}(f, \text{num} \rightarrow \text{real}).\text{conj}(\text{csu}(0, N)(\lambda n \in \text{num}. (f(n, 0) \ast \exp(i \ast (-2 \ast pi/N \ast n \ast k)))) = \text{csu}(0, N)(\lambda n \in \text{num}. (f(n, 0) \ast \exp(i \ast (-2 \ast pi/N \ast n \ast k)))) \]

**3.6. Frequency Shift.** The phase shift of the sequence \( x \) in the time domain corresponds to the circular shift of the sequence \( X \) in the frequency domain. The frequency shift contains the left shift and the right shift. They can be used in formal verification of cosine frequency shift. One has

\[
\text{DFT} \left[ x(n) e^{-j(2\pi/N)n} \right] = X \left( (k + l)_N \right), \tag{14}
\]

\[
\text{DFT} \left[ x(n) e^{j(2\pi/N)n} \right] = X \left( (k - l)_N \right). \tag{15}
\]

The left shift and the right shift are formalized into Theorems 13 and 14, respectively.

**Theorem 13.** DFT, FREQUENCY, LSHIFT

\[ \forall k \in \mathbb{N}. f, N \notin 0 \land 0 \leq k \land k < N \Rightarrow (\text{csu}(0, N)(\lambda n \ast \exp(i \ast (-2 \ast pi/N \ast n \ast l)) \ast \exp(i \ast (-2 \ast pi/N \ast n \ast k))) = \text{csu}(0, N)(\lambda n \ast f \ast n \ast \exp(i \ast (-2 \ast pi/N \ast n \ast (k - l)))) \]

**Theorem 14.** DFT, FREQUENCY, RSHIFT

\[ \forall k \in \mathbb{N}. f, N \notin 0 \land 0 \leq k \land k < N \Rightarrow (\text{csu}(0, N)(\lambda n \ast \exp(i \ast (2 \ast pi/N \ast n \ast l)) \ast \exp(i \ast (-2 \ast pi/N \ast n \ast k))) = \text{csu}(0, N)(\lambda n \ast f \ast n \ast \exp(l \ast (-2 \ast pi/N \ast n \ast (k - l)))) \]

**3.7. Time Shift.** Because of the duality property between the time domain and the frequency one, the circular shift of \( x \) in the time domain corresponds to the phase shift of \( X \) in the frequency domain. Consider

\[
\text{DFT} \left[ x(n+l)_N \right] = X(k)W_N^{kl}. \tag{16}
\]

The formal description is as follows.

**Theorem 15.** DFT, TIME, SHIFT

\[ \forall f \in \mathbb{N}, k, N \notin 0 \land 0 \leq k \land k < N \Rightarrow (\text{csu}(0, N)(\lambda n \ast f(n+l) \ast \exp(i \ast (2 \ast pi/N \ast n \ast l \ast k))) = \text{DFT} f(n,k) \]

**3.8. Convolution.** The convolution can be obtained as the product’s IDFT of two sequences’ DFT. The property facilitates a remarkable simplification. For two \( N \)-length sequences \( x_1(n) \) and \( x_2(n) \), their circular convolution \( y(n) \) is defined as follows:

\[
y(n) = \sum_{m=0}^{N-1} x_1(m) x_2((n-m))_N, \quad 0 \leq n \leq N - 1. \tag{17}
\]

The formal definition is as follows.

**Definition 16.** DFT, CONVOLUTION

\[ \forall f, g \in \mathbb{N}, n, f \ast g \in \text{sum}(0, N)(\lambda m \ast f(m) \ast g(n-m)). \]

Let \( Y \) be DFT of \( y \); the circular convolution theorem holds as follows:

\[
y(k) = \text{DFT} \left[ y(n) \right] = X_1(k) X_2(k), \quad 0 \leq k \leq N - 1. \tag{18}
\]

The formal theorem is as follows.

**Theorem 17.** DFT, CONVOLUTION, TIME

\[ \forall f, g \in \mathbb{N}, k, n, f \ast g \in \text{sum}(0, N)(\lambda m \ast f(m) \ast g(n-m)). \]

\[ \text{DFT} f \ast g = \text{DFT} f \ast \text{DFT} g \]

According to the theorem, the convolution can be calculated from the IDFT of \( Y \). Benefiting from Fast FT algorithms, the computing complexity can be reduced greatly.

4. Applications

To illustrate the correctness and usefulness of our work, we formally verify FFT and cosine frequency shift as examples in this section. DFT always has been implemented by employing FFT, which is one of the most important algorithms. The formal verification of FFT is the basis to verify applications which employed FFT. Frequency shift is a significant property which should be taken into account at DFT applications. The two examples are chosen for their representativeness.

4.1. FFT. Without question, DFT is extremely useful in many fields but it often takes too long time to compute DFT directly by the definition, especially for long date sequences. Therefore, many FFT algorithms had been developed to perform DFT quickly. Directly computing DFT of an \( N \)-length sequence takes \( O(N^2) \) arithmetical operations; by comparison, FFT algorithms take \( O(N \ast \log N) \) arithmetical operations to produce the same result. According to Formula (1) of DFT, the rotating operator \( W_N^{nk} \), which is a periodic complex exponential sequence, enables the complexity reduction.

For the periodic complex-power sequence \( W_N^{nk} \), it holds conjugated symmetry and reducibility as follows:

\[
W_N^{-nk} = (W_N^{nk})^*. \tag{19}
\]

The formalization is as follows.

**Lemma 18.** WN, CONJ, SYMM

\[ \forall k, m, N \in \mathbb{N}, \text{sum}(0, N)(\lambda n \ast \exp(i \ast (2 \ast pi/N \ast n \ast k))) = \text{conj}(\text{exp}(i \ast (-2 \ast pi/N \ast n \ast k))), \quad 0 \leq k \leq N - 1. \tag{20} \]
Mathematical Problems in Engineering 5

Algorithm 1: The formal definition of FFT.

Suppose that $X_1(k) = \text{DFT}(x_1(r)), X_3(k) = \text{DFT}(x_3(r)), 0 \leq k \leq N/2 - 1$; FFT can be described as follows:

$X(k) = X_1(k) + W_N^k X_2(k), 0 \leq k \leq \frac{N}{2} - 1,$

$X \left( k + \frac{N}{2} \right) = X_1(k) - W_N^k X_2(k), 0 \leq k \leq \frac{N}{2} - 1.$

FFT is consisting of the inverse processing and the butterfly shaped computation. And according to the above expressions, the FFT of $x(n)$ is the concatenation of Formula (23). The formal description of FFT is shown in Algorithm 1.

The FFT can be formally verified as an implementation of DFT. Firstly, use DEF def and WN_REDUCE_HALF to rewrite the goal. We also need to introduce a subgoal “$\exp(i \times (-2 \times pi/N \times (N/2))) = -1$” to simplify the current goal. Then we can use some tactics in hol4 and some appropriate operation rules in real script and complex script to extend and transform the goal as well as WN_PERIODIC, WN_CONJ_SYM, WN_REDUCE, and other properties mentioned above. The procedure of validation is shown in Algorithm 2.

4.2. Cosine Frequency Shift. Frequency shift is an important property of DFT, and it can derive many other similar properties like cosine frequency shift. This subsection presents formal verification of cosine frequency shift. A typical cosine frequency shift can be described as follows:

$$
\text{DFT} \left[ x(n) \cos \left( \frac{2\pi m}{N} n \right) \right] = \frac{1}{2} \left[ X(k - m) + X(k + m) \right].
$$

To verify the formula above using the frequency shift property of DFT, $\cos(n)$ should be transformed to exponential formula with Euler’s formula as follows.
Algorithm 2: The validation of FFT.

```plaintext
REWRITE_TAC [DFT_def, WN_REDUCE_HALF] THEN REPEAT STRIP_TAC;

// the initial goal was decomposed into 2 subgoals
[ASM_REWRITE_TAC [],]
// the 1st subgoal proved!
POP_ASSUM MP_TAC THEN
Q_SUBGOAL THEN "exp ((-2 * pi/N * (N/2))) = -1" ASSUME_TAC;
// the 1st auxiliary subgoal was added!
THENL
[REWRITE_TAC [EXP_IMAGINARY] THEN
Q_SUBGOAL THEN "-(2) * (pi:real)/N * (N/2):real = -pi" ASSUME_TAC;

// the 2nd auxiliary subgoal was added!
THENL
[REWRITE_TAC [REAL_MUL_ASSOC] THEN REWRITE_TAC [GSYM REAL_NEG_LMUL] THEN
ONCE_REWRITE_TAC [REAL_MUL_SYM] THEN
REWRITE_TAC [GSYM REAL_MUL_ASSOC] THEN
Q_SUBGOAL THEN "inv (N:real) * N:real = 1:real" ASSUME_TAC;

// the 3rd auxiliary subgoal was added!
THENL
[REWRITE_TAC [REAL_INV_1OVER] THEN SRW_TAC [],
REWRITE_TAC [GSYM REAL_1] THEN MATCH_MP_TAC REAL_MUL_LINV THEN REWRITE_TAC [REAL_0] THEN
METIS_TAC [],
ASM_REWRITE_TAC [] THEN
REWRITE_TAC [REAL_MUL_ASSOC] THEN Q_SUBGOAL THEN "inv 2:real <> 0:real" ASSUME_TAC;
// the 4th auxiliary subgoal was added!
THENL
[Q_SUBGOAL THEN "inv 2:real <> 0:real" ASSUME_TAC;

// all the auxiliary subgoals was proved!
ONCE_ASM_REWRITE_TAC [] THEN
ONCE_REWRITE_TAC [COMPLEX_MUL_COMM] THEN
REWRITE_TAC [GSYM COMPLEX_NEG_LMUL] THEN
REWRITE_TAC [COMPLEX_MUL_LID] THEN
REWRITE_TAC [GSYM COMPLEX_NEG_RMUL] THEN
METIS_TAC [GSYM complex_sub]
// the second subgoal proved!
]```

Algorithm 2: The validation of FFT.
Algorithm 3: The formalization of EULER_COS.

val EULER_COS = store_thm("EULER_COS",
"∀n:real.(cos(n,0)=(1)/(2) ∗ (exp(i ∗ n)+exp(i ∗ (-n))))",
GEN_TAC THEN REWRITE_TAC[EXP_IMAGINARY] THEN
REWRITE_TAC[complex_add, RE, IM, FST, SND] THEN
REWRITE_TAC[COS_NEG, SIN_NEG] THEN
SRW_TAC[REAL_DOUBLE] THEN
REWRITE_TAC[complex_scalar_lmul, RE, IM, FST, SND] THEN
Q.SUBGOAL THEN
METIS_TAC[]

Lemma 21. EULER_COS:

\[ \cos(n) = \frac{1}{2} \left( e^{in} + e^{-in} \right). \]  
(25)

Its formalization is shown in Algorithm 3.

Based on the definition of DFT and the lemma EULER_COS, Formula (25) can be formally described and verified as in Algorithm 4.

Algorithm 4: The formalization of cosine frequency shift.

Algorithm 4: The formalization of cosine frequency shift.

val DFT_MOD_COS = store_thm("DFT_MOD_COS",
"∀(k N m:num).(f:num->real).
(DFT f n (k - m) = csum(0,N) (∑(n:num). (f n) ∗ (cos(2 ∗ pi/N ∗ n ∗ (m))) \) ∗
(DFT f n (k + m) = csum(0,N) (∑(n:num). (f n) ∗ (cos(2 ∗ pi/N ∗ n ∗ (m)))))
⇒
(\(\tfrac{csum(0,N) (∑(n:num). (f n) ∗ (cos(2 ∗ pi/N ∗ n ∗ (m))) \)}{2} \) ∗
(DFT f n (k-m)) + \(\tfrac{1}{2} \) ∗ (DFT f n (k + m)))"
REPEAT_STRIP_TAC THEN ASM_REWRITE_TAC[] THEN REWRITE_TAC[EULER_COS] THEN
REWRITE_TAC[COMPLEX_ADD_SCALAR_LMUL, COMPLEX_ADD_RDISTRIB, CSUM_ADD] THEN
Q.SUBGOAL THEN
"∀(a b:real)c:complexd:complex.a ∗ (b ∗ c) ∗ d = b ∗ (a ∗ c ∗ d)" ASSUME_TAC
//the 1st auxiliary subgoal was added!
THEN
[REPEAT_STRIP_TAC THEN REWRITE_TAC[COMPLEX_LMUL_SCALAR_LMUL] THEN
ONCE_REWRITE_TAC[COMPLEX_SCALAR_MUL_COMM] THEN
GEN_REWRITE_TAC LAND_CONV [GSYM COMPLEX_SCALAR_MUL_COMM] THEN
GEN_REWRITE_TAC LAND_CONV [COMPLEX_SCALAR_LMUL] THEN
ONCE_REWRITE_TAC[COMPLEX_SCALAR_LMUL_COMM] THEN
SRW_TAC[GSYM COMPLEX_SCALAR_RMUL]
//the 1st auxiliary subgoal proved!,
ASM_REWRITE_TAC[] THEN REWRITE_TAC[CSUM_RMUL] THEN
REWRITE_TAC[GSYM COMPLEX_ADD_SCALAR_LMUL] THEN
AP_TERM_TAC THEN REWRITE_TAC[real_div, REAL_NEG_RMUL] THEN
REWRITE_TAC[GSYM real_div] THEN
BINOP_TAC
//The initial goal was decomposed into 2 subgoals
THEN
[MATCH_MP_TAC DFT_FREQUENCY_RSHIFT THEN
METIS_TAC[]
//The 1st subgoal proved,
MATCH_MP_TAC DFT_FREQUENCY_LSHIFT THEN
METIS_TAC[]
The 2nd subgoal proved]
]

5. Conclusion

In the description of cosine frequency shift as shown in Algorithm 4, DFT\[x(n) \cos(2 \pi m/N \times n)\] is directly expanded by the definition of DFT to facilitate the proving process.

DFT is widely used in DSP and linear time invariant (LTI) systems which could be in safety-critical fields where formal methods are expected to be employed to assure accurate and
complete verification. This paper presented the formalization of DFT in HOL, including formal definition of DFT, formal verification of properties of DFT, and two case studies demonstrating usefulness and correctness of our work. Our work can be equipped as a library of HOL to facilitate formal verification of systems employing DFT.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors thank Professor Shengzhen Jin for his helpful suggestions. This work was supported by the International Cooperation Program on Science and Technology (2010DFB10930 and 2011DFG13000), the National Natural Science Foundation of China (61170304, 61104035, 61373034, 61303014, and 61472468), the Project of Construction of Innovative Teams and Teacher Career Development for Universities and Colleges Under Beijing Municipality, and the Scientific Research Base Development Program of the Beijing Municipal Commission of Education (TJSHG201310028014).

References

Submit your manuscripts at http://www.hindawi.com