FORMALIZED MATHEMATICS
Vol. 16, No. 3, Pages 231–245, 2008

Model Checking. Part II

Kazuhisa Ishida
Shinshu University
Nagano, Japan

Summary. This article provides the definition of linear temporal logic (LTL) and its properties relevant to model checking based on [9]. Mizar formalization of LTL language and satisfiability is based on [2, 3].

MML identifier: `MODELC_2`, version: `7.9.01 4.101.1015`

The articles [8], [11], [6], [5], [7], [1], [4], [12], and [10] provide the notation and terminology for this paper.

Let x be a set. The functor $\mathrm{CastNat}\, x$ yielding a natural number is defined by:

(Def. 1) $\mathrm{CastNat}\, x = \begin{cases} x, & \text{if } x \text{ is a natural number,} \\ 0, & \text{otherwise.} \end{cases}$

Let W_1 be a set. A sequence of W_1 is a function from $\mathbb{N}$ into W_1.

For simplicity, we adopt the following rules: k, n denote natural numbers, a denotes a set, D, S denote non empty sets, and p, q denote finite sequences of elements of $\mathbb{N}$.

Let us consider n. The functor $\mathrm{atom.}\, n$ yielding a finite sequence of elements of $\mathbb{N}$ is defined as follows:

(Def. 2) $\mathrm{atom.}\, n = \langle 6 + n \rangle$.

Let us consider p. The functor $\neg p$ yielding a finite sequence of elements of $\mathbb{N}$ is defined by:

(Def. 3) $\neg p = \langle 0 \rangle ^\frown p$.

Let us consider q. The functor $p \wedge q$ yields a finite sequence of elements of $\mathbb{N}$ and is defined by:

(Def. 4) $p \wedge q = \langle 1 \rangle ^\frown p ^\frown q$.

The functor $p \vee q$ yielding a finite sequence of elements of $\mathbb{N}$ is defined by:

ISSN 1426–2630(p), 1898-9934(e)

(Def. 5) $p \vee q = \langle 2 \rangle \frown p \frown q$.

Let us consider p. The functor $\mathcal{X}\, p$ yielding a finite sequence of elements of $\mathbb{N}$ is defined as follows:

(Def. 6) $\mathcal{X}\, p = \langle 3 \rangle \frown p$.

Let us consider q. The functor $p\,\mathcal{U}\,q$ yielding a finite sequence of elements of $\mathbb{N}$ is defined by:

(Def. 7) $p\,\mathcal{U}\,q = \langle 4 \rangle \frown p \frown q$.

The functor $p\,\mathcal{R}\,q$ yields a finite sequence of elements of $\mathbb{N}$ and is defined as follows:

(Def. 8) $p\,\mathcal{R}\,q = \langle 5 \rangle \frown p \frown q$.

The non empty set $\mathrm{WFF}_{\mathrm{LTL}}$ is defined by the conditions (Def. 9).

(Def. 9) For every a such that $a \in \mathrm{WFF}_{\mathrm{LTL}}$ holds a is a finite sequence of elements of $\mathbb{N}$ and for every n holds $\mathrm{atom.}\, n \in \mathrm{WFF}_{\mathrm{LTL}}$ and for every p such that $p \in \mathrm{WFF}_{\mathrm{LTL}}$ holds $\neg p \in \mathrm{WFF}_{\mathrm{LTL}}$ and for all p, q such that $p, q \in \mathrm{WFF}_{\mathrm{LTL}}$ holds $p \wedge q \in \mathrm{WFF}_{\mathrm{LTL}}$ and for all p, q such that $p, q \in \mathrm{WFF}_{\mathrm{LTL}}$ holds $p \vee q \in \mathrm{WFF}_{\mathrm{LTL}}$ and for every p such that $p \in \mathrm{WFF}_{\mathrm{LTL}}$ holds $\mathcal{X}\, p \in \mathrm{WFF}_{\mathrm{LTL}}$ and for all p, q such that $p, q \in \mathrm{WFF}_{\mathrm{LTL}}$ holds $p\,\mathcal{U}\,q \in \mathrm{WFF}_{\mathrm{LTL}}$ and for all p, q such that $p, q \in \mathrm{WFF}_{\mathrm{LTL}}$ holds $p\,\mathcal{R}\,q \in \mathrm{WFF}_{\mathrm{LTL}}$ and for every D such that for every a such that $a \in D$ holds a is a finite sequence of elements of $\mathbb{N}$ and for every n holds $\mathrm{atom.}\, n \in D$ and for every p such that $p \in D$ holds $\neg p \in D$ and for all p, q such that $p, q \in D$ holds $p \wedge q \in D$ and for all p, q such that $p, q \in D$ holds $p \vee q \in D$ and for every p such that $p \in D$ holds $\mathcal{X}\, p \in D$ and for all p, q such that $p, q \in D$ holds $p\,\mathcal{U}\,q \in D$ and for all p, q such that $p, q \in D$ holds $p\,\mathcal{R}\,q \in D$ holds $\mathrm{WFF}_{\mathrm{LTL}} \subseteq D$.

Let I_1 be a finite sequence of elements of $\mathbb{N}$. We say that I_1 is LTL-formula-like if and only if:

(Def. 10) I_1 is an element of $\mathrm{WFF}_{\mathrm{LTL}}$.

Let us observe that there exists a finite sequence of elements of $\mathbb{N}$ which is LTL-formula-like.

An LTL-formula is a LTL-formula-like finite sequence of elements of $\mathbb{N}$.

Next we state the proposition

(1) a is an LTL-formula iff $a \in \mathrm{WFF}_{\mathrm{LTL}}$.

In the sequel F, F_1, G, H, H_1, H_2 denote LTL-formulae.

Let us consider n. Observe that $\mathrm{atom.}\, n$ is LTL-formula-like.

Let us consider H. Note that $\neg H$ is LTL-formula-like and $\mathcal{X}\, H$ is LTL-formula-like. Let us consider G. One can check the following observations:

* $H \wedge G$ is LTL-formula-like,
* $H \vee G$ is LTL-formula-like,
* $H\,\mathcal{U}\,G$ is LTL-formula-like, and

∗ $H \mathcal{R} G$ is LTL-formula-like.

Let us consider H. We say that H is atomic if and only if:

(Def. 11) There exists n such that $H = \mathrm{atom}.\,n$.

We say that H is negative if and only if:

(Def. 12) There exists H_1 such that $H = \neg H_1$.

We say that H is conjunctive if and only if:

(Def. 13) There exist F, G such that $H = F \wedge G$.

We say that H is disjunctive if and only if:

(Def. 14) There exist F, G such that $H = F \vee G$.

We say that H has *next* operator if and only if:

(Def. 15) There exists H_1 such that $H = \mathcal{X}\, H_1$.

We say that H has *until* operator if and only if:

(Def. 16) There exist F, G such that $H = F \,\mathcal{U}\, G$.

We say that H has *release* operator if and only if:

(Def. 17) There exist F, G such that $H = F \,\mathcal{R}\, G$.

Next we state two propositions:

(2) H is either atomic, or negative, or conjunctive, or disjunctive, or has *next* operator, or *until* operator, or *release* operator.

(3) $1 \leq \mathrm{len}\, H$.

Let us consider H. Let us assume that H is either negative or has *next* operator. The functor $\mathrm{Arg}(H)$ yields an LTL-formula and is defined by:

(Def. 18)(i) $\neg \mathrm{Arg}(H) = H$ if H is negative,

(ii) $\mathcal{X}\, \mathrm{Arg}(H) = H$, otherwise.

Let us consider H. Let us assume that H is either conjunctive or disjunctive or has *until* operator or *release* operator. The functor $\mathrm{LeftArg}(H)$ yielding an LTL-formula is defined as follows:

(Def. 19)(i) There exists H_1 such that $\mathrm{LeftArg}(H) \wedge H_1 = H$ if H is conjunctive,

(ii) there exists H_1 such that $\mathrm{LeftArg}(H) \vee H_1 = H$ if H is disjunctive,

(iii) there exists H_1 such that $\mathrm{LeftArg}(H) \mathcal{U} H_1 = H$ if H has *until* operator,

(iv) there exists H_1 such that $\mathrm{LeftArg}(H) \,\mathcal{R}\, H_1 = H$, otherwise.

The functor $\mathrm{RightArg}(H)$ yields an LTL-formula and is defined by:

(Def. 20)(i) There exists H_1 such that $H_1 \wedge \mathrm{RightArg}(H) = H$ if H is conjunctive,

(ii) there exists H_1 such that $H_1 \vee \mathrm{RightArg}(H) = H$ if H is disjunctive,

(iii) there exists H_1 such that $H_1 \,\mathcal{U}\, \mathrm{RightArg}(H) = H$ if H has *until* operator,

(iv) there exists H_1 such that $H_1 \,\mathcal{R}\, \mathrm{RightArg}(H) = H$, otherwise.

The following propositions are true:

(4) If H is negative, then $H = \neg \mathrm{Arg}(H)$.

(5) If H has *next* operator, then $H = \mathcal{X}\,\mathrm{Arg}(H)$.

(6) If H is conjunctive, then $H = \mathrm{LeftArg}(H) \wedge \mathrm{RightArg}(H)$.

(7) If H is disjunctive, then $H = \mathrm{LeftArg}(H) \vee \mathrm{RightArg}(H)$.

(8) If H has *until* operator, then $H = \mathrm{LeftArg}(H)\,\mathcal{U}\,\mathrm{RightArg}(H)$.

(9) If H has *release* operator, then $H = \mathrm{LeftArg}(H)\,\mathcal{R}\,\mathrm{RightArg}(H)$.

(10) If H is either negative or has *next* operator, then $\mathrm{len}\,H = 1 + \mathrm{len}\,\mathrm{Arg}(H)$ and $\mathrm{len}\,\mathrm{Arg}(H) < \mathrm{len}\,H$.

(11) Suppose H is either conjunctive or disjunctive or has *until* operator or *release* operator. Then $\mathrm{len}\,H = 1 + \mathrm{len}\,\mathrm{LeftArg}(H) + \mathrm{len}\,\mathrm{RightArg}(H)$ and $\mathrm{len}\,\mathrm{LeftArg}(H) < \mathrm{len}\,H$ and $\mathrm{len}\,\mathrm{RightArg}(H) < \mathrm{len}\,H$.

Let us consider H, F. We say that H is an immediate constituent of F if and only if:

(Def. 21) $F = \neg H$ or $F = \mathcal{X}\,H$ or there exists H_1 such that $F = H \wedge H_1$ or $F = H_1 \wedge H$ or $F = H \vee H_1$ or $F = H_1 \vee H$ or $F = H\,\mathcal{U}\,H_1$ or $F = H_1\,\mathcal{U}\,H$ or $F = H\,\mathcal{R}\,H_1$ or $F = H_1\,\mathcal{R}\,H$.

We now state a number of propositions:

(12) For all F, G holds $(\neg F)(1) = 0$ and $(F \wedge G)(1) = 1$ and $(F \vee G)(1) = 2$ and $(\mathcal{X}\,F)(1) = 3$ and $(F\,\mathcal{U}\,G)(1) = 4$ and $(F\,\mathcal{R}\,G)(1) = 5$.

(13) H is an immediate constituent of $\neg F$ iff $H = F$.

(14) H is an immediate constituent of $\mathcal{X}\,F$ iff $H = F$.

(15) H is an immediate constituent of $F \wedge G$ iff $H = F$ or $H = G$.

(16) H is an immediate constituent of $F \vee G$ iff $H = F$ or $H = G$.

(17) H is an immediate constituent of $F\,\mathcal{U}\,G$ iff $H = F$ or $H = G$.

(18) H is an immediate constituent of $F\,\mathcal{R}\,G$ iff $H = F$ or $H = G$.

(19) If F is atomic, then H is not an immediate constituent of F.

(20) If F is negative, then H is an immediate constituent of F iff $H = \mathrm{Arg}(F)$.

(21) If F has *next* operator, then H is an immediate constituent of F iff $H = \mathrm{Arg}(F)$.

(22) If F is conjunctive, then H is an immediate constituent of F iff $H = \mathrm{LeftArg}(F)$ or $H = \mathrm{RightArg}(F)$.

(23) If F is disjunctive, then H is an immediate constituent of F iff $H = \mathrm{LeftArg}(F)$ or $H = \mathrm{RightArg}(F)$.

(24) If F has *until* operator, then H is an immediate constituent of F iff $H = \mathrm{LeftArg}(F)$ or $H = \mathrm{RightArg}(F)$.

(25) If F has *release* operator, then H is an immediate constituent of F iff $H = \mathrm{LeftArg}(F)$ or $H = \mathrm{RightArg}(F)$.

(26) Suppose H is an immediate constituent of F. Then F is either negative, or conjunctive, or disjunctive, or has *next* operator, or *until* operator, or

release operator.

In the sequel L denotes a finite sequence.

Let us consider H, F. We say that H is a subformula of F if and only if the condition (Def. 22) is satisfied.

(Def. 22) There exist n, L such that

(i) $1 \leq n$,

(ii) $\operatorname{len} L = n$,

(iii) $L(1) = H$,

(iv) $L(n) = F$, and

(v) for every k such that $1 \leq k < n$ there exist H_1, F_1 such that $L(k) = H_1$ and $L(k+1) = F_1$ and H_1 is an immediate constituent of F_1.

We now state the proposition

(27) H is a subformula of H.

Let us consider H, F. We say that H is a proper subformula of F if and only if:

(Def. 23) H is a subformula of F and $H \neq F$.

One can prove the following propositions:

(28) If H is an immediate constituent of F, then $\operatorname{len} H < \operatorname{len} F$.

(29) If H is an immediate constituent of F, then H is a proper subformula of F.

(30) If G is either negative or has *next* operator, then $\operatorname{Arg}(G)$ is a subformula of G.

(31) Suppose G is either conjunctive or disjunctive or has *until* operator or *release* operator. Then $\operatorname{LeftArg}(G)$ is a subformula of G and $\operatorname{RightArg}(G)$ is a subformula of G.

(32) If H is a proper subformula of F, then $\operatorname{len} H < \operatorname{len} F$.

(33) If H is a proper subformula of F, then there exists G which is an immediate constituent of F.

(34) If F is a proper subformula of G and G is a proper subformula of H, then F is a proper subformula of H.

(35) If F is a subformula of G and G is a subformula of H, then F is a subformula of H.

(36) If G is a subformula of H and H is a subformula of G, then $G = H$.

(37) If G is either negative or has *next* operator and F is a proper subformula of G, then F is a subformula of $\operatorname{Arg}(G)$.

(38) Suppose that

(i) G is either conjunctive or disjunctive or has *until* operator or *release* operator, and

(ii) F is a proper subformula of G.

Then F is a subformula of $\mathrm{LeftArg}(G)$ or a subformula of $\mathrm{RightArg}(G)$.

(39) If F is a proper subformula of $\neg H$, then F is a subformula of H.

(40) If F is a proper subformula of $\mathcal{X}\, H$, then F is a subformula of H.

(41) If F is a proper subformula of $G \wedge H$, then F is a subformula of G or a subformula of H.

(42) If F is a proper subformula of $G \vee H$, then F is a subformula of G or a subformula of H.

(43) If F is a proper subformula of $G\,\mathcal{U}\,H$, then F is a subformula of G or a subformula of H.

(44) If F is a proper subformula of $G\,\mathcal{R}\,H$, then F is a subformula of G or a subformula of H.

Let us consider H. The functor $\mathrm{Subformulae}\, H$ yields a set and is defined by:

(Def. 24) $a \in \mathrm{Subformulae}\, H$ iff there exists F such that $F = a$ and F is a subformula of H.

One can prove the following proposition

(45) $G \in \mathrm{Subformulae}\, H$ iff G is a subformula of H.

Let us consider H. Observe that $\mathrm{Subformulae}\, H$ is non empty.

Next we state two propositions:

(46) If F is a subformula of H, then $\mathrm{Subformulae}\, F \subseteq \mathrm{Subformulae}\, H$.

(47) If a is a subset of $\mathrm{Subformulae}\, H$, then a is a subset of $\mathrm{WFF}_{\mathrm{LTL}}$.

In this article we present several logical schemes. The scheme *LTLInd* concerns a unary predicate $\mathcal{P}$, and states that:

For every H holds $\mathcal{P}[H]$

provided the following conditions are satisfied:

- For every H such that H is atomic holds $\mathcal{P}[H]$,
- For every H such that H is either negative or has *next* operator and $\mathcal{P}[\mathrm{Arg}(H)]$ holds $\mathcal{P}[H]$, and
- Let given H. Suppose H is either conjunctive or disjunctive or has *until* operator or *release* operator and $\mathcal{P}[\mathrm{LeftArg}(H)]$ and $\mathcal{P}[\mathrm{RightArg}(H)]$. Then $\mathcal{P}[H]$.

The scheme *LTLCompInd* concerns a unary predicate $\mathcal{P}$, and states that:

For every H holds $\mathcal{P}[H]$

provided the following condition is met:

- For every H such that for every F such that F is a proper subformula of H holds $\mathcal{P}[F]$ holds $\mathcal{P}[H]$.

Let x be a set. The functor $\mathrm{Cast}_{\mathrm{LTL}}\, x$ yielding an LTL-formula is defined by:

(Def. 25) $$\mathrm{Cast}_{\mathrm{LTL}}\, x = \begin{cases} x, & \text{if } x \in \mathrm{WFF}_{\mathrm{LTL}}, \\ \mathrm{atom}.\,0, & \text{otherwise.} \end{cases}$$

We introduce LTL-model structures which are systems

⟨ assignations, basic assignations, a conjunction, a disjunction, a negation, a next-operation, an until-operation, a release-operation ⟩,

where the assignations constitute a non empty set, the basic assignations constitute a non empty subset of the assignations, the conjunction is a binary operation on the assignations, the disjunction is a binary operation on the assignations, the negation is a unary operation on the assignations, the next-operation is a unary operation on the assignations, the until-operation is a binary operation on the assignations, and the release-operation is a binary operation on the assignations.

Let V be an LTL-model structure. An assignation of V is an element of the assignations of V.

The subset $\mathrm{atomic}_{\mathrm{LTL}}$ of $\mathrm{WFF}_{\mathrm{LTL}}$ is defined by:

(Def. 26) $\mathrm{atomic}_{\mathrm{LTL}} = \{x; x$ ranges over LTL-formulae: x is atomic$\}$.

Let V be an LTL-model structure, let K_1 be a function from $\mathrm{atomic}_{\mathrm{LTL}}$ into the basic assignations of V, and let f be a function from $\mathrm{WFF}_{\mathrm{LTL}}$ into the assignations of V. We say that f is an evaluation for K_1 if and only if the condition (Def. 27) is satisfied.

(Def. 27) Let H be an LTL-formula. Then

(i) if H is atomic, then $f(H) = K_1(H)$,

(ii) if H is negative, then $f(H) =$ (the negation of V)$(f(\mathrm{Arg}(H)))$,

(iii) if H is conjunctive, then $f(H) =$ (the conjunction of V)$(f(\mathrm{LeftArg}(H)), f(\mathrm{RightArg}(H)))$,

(iv) if H is disjunctive, then $f(H) =$ (the disjunction of V)$(f(\mathrm{LeftArg}(H)), f(\mathrm{RightArg}(H)))$,

(v) if H has *next* operator, then $f(H) =$ (the next-operation of V)$(f(\mathrm{Arg}(H)))$,

(vi) if H has *until* operator, then $f(H) =$ (the until-operation of V)$(f(\mathrm{LeftArg}(H)), f(\mathrm{RightArg}(H)))$, and

(vii) if H has *release* operator, then $f(H) =$ (the release-operation of V)$(f(\mathrm{LeftArg}(H)), f(\mathrm{RightArg}(H)))$.

Let V be an LTL-model structure, let K_1 be a function from $\mathrm{atomic}_{\mathrm{LTL}}$ into the basic assignations of V, let f be a function from $\mathrm{WFF}_{\mathrm{LTL}}$ into the assignations of V, and let n be a natural number. We say that f is a n-pre-evaluation for K_1 if and only if the condition (Def. 28) is satisfied.

(Def. 28) Let H be an LTL-formula such that $\mathrm{len}\, H \leq n$. Then

(i) if H is atomic, then $f(H) = K_1(H)$,

(ii) if H is negative, then $f(H) =$ (the negation of V)$(f(\mathrm{Arg}(H)))$,

(iii) if H is conjunctive, then $f(H) =$ (the conjunction of V)$(f(\mathrm{LeftArg}(H)), f(\mathrm{RightArg}(H)))$,

(iv) if H is disjunctive, then $f(H) =$ (the disjunction of V)$(f(\mathrm{LeftArg}(H)), f(\mathrm{RightArg}(H)))$,

(v) if H has *next* operator, then $f(H) =$ (the next-operation of V)$(f(\mathrm{Arg}(H)))$,

(vi) if H has *until* operator, then $f(H) =$ (the until-operation of V)$(f(\mathrm{LeftArg}(H)), f(\mathrm{RightArg}(H)))$, and

(vii) if H has *release* operator, then $f(H) =$ (the release-operation of V)$(f(\mathrm{LeftArg}(H)), f(\mathrm{RightArg}(H)))$.

Let V be an LTL-model structure, let K_1 be a function from $\mathrm{atomic}_{\mathrm{LTL}}$ into the basic assignations of V, let f, h be functions from $\mathrm{WFF}_{\mathrm{LTL}}$ into the assignations of V, let n be a natural number, and let H be an LTL-formula. The functor $\mathrm{GraftEval}(V, K_1, f, h, n, H)$ yields a set and is defined by:

(Def. 29) $\mathrm{GraftEval}(V, K_1, f, h, n, H)$

$$= \begin{cases} f(H), \text{ if } \mathrm{len}\, H > n+1, \\ K_1(H), \text{ if } \mathrm{len}\, H = n+1 \text{ and } H \text{ is atomic}, \\ (\text{the negation of } V)(h(\mathrm{Arg}(H))), \text{ if } \mathrm{len}\, H = n+1 \text{ and } H \text{ is negative}, \\ (\text{the conjunction of } V)(h(\mathrm{LeftArg}(H)), h(\mathrm{RightArg}(H))), \\ \quad \text{if } \mathrm{len}\, H = n+1 \text{ and } H \text{ is conjunctive}, \\ (\text{the disjunction of } V)(h(\mathrm{LeftArg}(H)), h(\mathrm{RightArg}(H))), \\ \quad \text{if } \mathrm{len}\, H = n+1 \text{ and } H \text{ is disjunctive}, \\ (\text{the next-operation of } V)(h(\mathrm{Arg}(H))), \\ \quad \text{if } \mathrm{len}\, H = n+1 \text{ and } H \text{ has } \textit{next} \text{ operator}, \\ (\text{the until-operation of } V)(h(\mathrm{LeftArg}(H)), h(\mathrm{RightArg}(H))), \\ \quad \text{if } \mathrm{len}\, H = n+1 \text{ and } H \text{ has } \textit{until} \text{ operator}, \\ (\text{the release-operation of } V)(h(\mathrm{LeftArg}(H)), h(\mathrm{RightArg}(H))), \\ \quad \text{if } \mathrm{len}\, H = n+1 \text{ and } H \text{ has } \textit{release} \text{ operator}, \\ h(H), \text{ if } \mathrm{len}\, H < n+1, \\ \emptyset, \text{ otherwise.} \end{cases}$$

We adopt the following convention: V denotes an LTL-model structure, K_1 denotes a function from $\mathrm{atomic}_{\mathrm{LTL}}$ into the basic assignations of V, and f, f_1, f_2 denote functions from $\mathrm{WFF}_{\mathrm{LTL}}$ into the assignations of V.

Let V be an LTL-model structure, let K_1 be a function from $\mathrm{atomic}_{\mathrm{LTL}}$ into the basic assignations of V, and let n be a natural number. The functor $\mathrm{EvalSet}(V, K_1, n)$ yields a non empty set and is defined by:

(Def. 30) $\mathrm{EvalSet}(V, K_1, n) = \{h; h$ ranges over functions from $\mathrm{WFF}_{\mathrm{LTL}}$ into the assignations of V: h is a n-pre-evaluation for $K_1\}$.

Let V be an LTL-model structure, let v_0 be an element of the assignations of V, and let x be a set. The functor $\mathrm{CastEval}(V, x, v_0)$ yielding a function from $\mathrm{WFF}_{\mathrm{LTL}}$ into the assignations of V is defined by:

(Def. 31) $\mathrm{CastEval}(V, x, v_0) = \begin{cases} x, \text{ if } x \in (\text{the assignations of } V)^{\mathrm{WFF}_{\mathrm{LTL}}}, \\ \mathrm{WFF}_{\mathrm{LTL}} \longmapsto v_0, \text{ otherwise.} \end{cases}$

Let V be an LTL-model structure and let K_1 be a function from atomic$_{\mathrm{LTL}}$ into the basic assignations of V. The functor EvalFamily(V, K_1) yielding a non empty set is defined by the condition (Def. 32).

(Def. 32) Let p be a set. Then $p \in$ EvalFamily(V, K_1) if and only if the following conditions are satisfied:

(i) $p \in 2^{(\text{the assignations of } V)^{\mathrm{WFF}_{\mathrm{LTL}}}}$, and

(ii) there exists a natural number n such that $p = \mathrm{EvalSet}(V, K_1, n)$.

We now state two propositions:

(48) There exists f which is an evaluation for K_1.

(49) If f_1 is an evaluation for K_1 and f_2 is an evaluation for K_1, then $f_1 = f_2$.

Let V be an LTL-model structure, let K_1 be a function from atomic$_{\mathrm{LTL}}$ into the basic assignations of V, and let H be an LTL-formula. The functor Evaluate(H, K_1) yields an assignation of V and is defined by:

(Def. 33) There exists a function f from WFF$_{\mathrm{LTL}}$ into the assignations of V such that f is an evaluation for K_1 and Evaluate$(H, K_1) = f(H)$.

Let V be an LTL-model structure and let f be an assignation of V. The functor $\neg f$ yielding an assignation of V is defined by:

(Def. 34) $\neg f =$ (the negation of V)(f).

Let V be an LTL-model structure and let f, g be assignations of V. The functor $f \wedge g$ yields an assignation of V and is defined by:

(Def. 35) $f \wedge g =$ (the conjunction of V)(f, g).

The functor $f \vee g$ yields an assignation of V and is defined as follows:

(Def. 36) $f \vee g =$ (the disjunction of V)(f, g).

Let V be an LTL-model structure and let f be an assignation of V. The functor $\mathcal{X} f$ yielding an assignation of V is defined by:

(Def. 37) $\mathcal{X} f =$ (the next-operation of V)(f).

Let V be an LTL-model structure and let f, g be assignations of V. The functor $f \,\mathcal{U}\, g$ yielding an assignation of V is defined by:

(Def. 38) $f \,\mathcal{U}\, g =$ (the until-operation of V)(f, g).

The functor $f \,\mathcal{R}\, g$ yields an assignation of V and is defined as follows:

(Def. 39) $f \,\mathcal{R}\, g =$ (the release-operation of V)(f, g).

One can prove the following propositions:

(50) $\mathrm{Evaluate}(\neg H, K_1) = \neg\, \mathrm{Evaluate}(H, K_1)$.

(51) $\mathrm{Evaluate}(H_1 \wedge H_2, K_1) = \mathrm{Evaluate}(H_1, K_1) \wedge \mathrm{Evaluate}(H_2, K_1)$.

(52) $\mathrm{Evaluate}(H_1 \vee H_2, K_1) = \mathrm{Evaluate}(H_1, K_1) \vee \mathrm{Evaluate}(H_2, K_1)$.

(53) $\mathrm{Evaluate}(\mathcal{X}\, H, K_1) = \mathcal{X}\, \mathrm{Evaluate}(H, K_1)$.

(54) $\mathrm{Evaluate}(H_1 \,\mathcal{U}\, H_2, K_1) = \mathrm{Evaluate}(H_1, K_1) \,\mathcal{U}\, \mathrm{Evaluate}(H_2, K_1)$.

(55) $\mathrm{Evaluate}(H_1 \,\mathcal{R}\, H_2, K_1) = \mathrm{Evaluate}(H_1, K_1) \,\mathcal{R}\, \mathrm{Evaluate}(H_2, K_1)$.

Let S be a non empty set. The infinite sequences of S yielding a non empty set is defined by:

(Def. 40) The infinite sequences of $S = S^{\mathbb{N}}$.

Let S be a non empty set and let t be a sequence of S. The functor $\mathrm{CastSeq}\, t$ yields an element of the infinite sequences of S and is defined by:

(Def. 41) $\mathrm{CastSeq}\, t = t$.

Let S be a non empty set and let t be a set. Let us assume that t is an element of the infinite sequences of S. The functor $\mathrm{CastSeq}(t, S)$ yielding a sequence of S is defined by:

(Def. 42) $\mathrm{CastSeq}(t, S) = t$.

Let S be a non empty set, let t be a sequence of S, and let k be a natural number. The functor $\mathrm{Shift}(t, k)$ yielding a sequence of S is defined as follows:

(Def. 43) For every natural number n holds $(\mathrm{Shift}(t, k))(n) = t(n + k)$.

Let S be a non empty set, let t be a set, and let k be a natural number. The functor $\mathrm{Shift}(t, k, S)$ yielding an element of the infinite sequences of S is defined as follows:

(Def. 44) $\mathrm{Shift}(t, k, S) = \mathrm{CastSeq}\, \mathrm{Shift}(\mathrm{CastSeq}(t, S), k)$.

Let S be a non empty set, let t be an element of the infinite sequences of S, and let k be a natural number. The functor $\mathrm{Shift}(t, k)$ yielding an element of the infinite sequences of S is defined as follows:

(Def. 45) $\mathrm{Shift}(t, k) = \mathrm{Shift}(t, k, S)$.

Let S be a non empty set and let f be a set. The functor $\mathrm{Not}_0(f, S)$ yields an element of ModelSP (the infinite sequences of S) and is defined by the condition (Def. 46).

(Def. 46) Let t be a set. Suppose $t \in$ the infinite sequences of S. Then $\neg\,\mathrm{Castboolean}(\mathrm{Fid}(f, \text{the infinite sequences of } S))(t) = \mathit{true}$ if and only if $(\mathrm{Fid}(\mathrm{Not}_0(f, S), \text{the infinite sequences of } S))(t) = \mathit{true}$.

Let S be a non empty set. The functor $\mathrm{Not}\, S$ yielding a unary operation on ModelSP (the infinite sequences of S) is defined by:

(Def. 47) For every set f such that $f \in$ ModelSP (the infinite sequences of S) holds $(\mathrm{Not}\, S)(f) = \mathrm{Not}_0(f, S)$.

Let S be a non empty set, let f be a function from the infinite sequences of S into *Boolean*, and let t be a set. The functor $\text{Next-univ}(t, f)$ yields an element of *Boolean* and is defined as follows:

(Def. 48) $\text{Next-univ}(t, f) = \begin{cases} \mathit{true}, & \text{if } t \text{ is an element of the infinite sequences} \\ & \quad \text{of } S \text{ and } f(\mathrm{Shift}(t, 1, S)) = \mathit{true}, \\ \mathit{false}, & \text{otherwise.} \end{cases}$

Let S be a non empty set and let f be a set. The functor $\mathrm{Next}_0(f, S)$ yielding an element of ModelSP (the infinite sequences of S) is defined by the condition

(Def. 49).

(Def. 49) Let t be a set. Suppose $t \in$ the infinite sequences of S. Then $\text{Next-univ}(t, \text{Fid}(f, \text{the infinite sequences of } S)) = \mathit{true}$ if and only if $(\text{Fid}(\text{Next}_0(f, S), \text{the infinite sequences of } S))(t) = \mathit{true}$.

Let S be a non empty set. The functor $\text{Next}\, S$ yields a unary operation on ModelSP (the infinite sequences of S) and is defined as follows:

(Def. 50) For every set f such that $f \in$ ModelSP (the infinite sequences of S) holds $(\text{Next}\, S)(f) = \text{Next}_0(f, S)$.

Let S be a non empty set and let f, g be sets. The functor $\text{And}_0(f, g, S)$ yields an element of ModelSP (the infinite sequences of S) and is defined by the condition (Def. 51).

(Def. 51) Let t be a set. Suppose $t \in$ the infinite sequences of S. Then $\text{Castboolean}(\text{Fid}(f, \text{the infinite sequences of } S))(t) \wedge \text{Castboolean}(\text{Fid}(g, \text{the infinite sequences of } S))(t) = \mathit{true}$ if and only if $(\text{Fid}(\text{And}_0(f, g, S), \text{the infinite sequences of } S))(t) = \mathit{true}$.

Let S be a non empty set. The functor $\text{And}\, S$ yielding a binary operation on ModelSP (the infinite sequences of S) is defined by the condition (Def. 52).

(Def. 52) Let f, g be sets. Suppose $f \in$ ModelSP (the infinite sequences of S) and $g \in$ ModelSP (the infinite sequences of S). Then $(\text{And}\, S)(f, g) = \text{And}_0(f, g, S)$.

Let S be a non empty set, let f, g be functions from the infinite sequences of S into *Boolean*, and let t be a set. The functor $\text{Until-univ}(t, f, g, S)$ yields an element of *Boolean* and is defined as follows:

(Def. 53) $$\text{Until-univ}(t, f, g, S) = \begin{cases} \mathit{true}, & \text{if } t \text{ is an element of the infinite sequences of } S \text{ and there exists a natural number } m \text{ such that for every natural number } j \text{ such that } j < m \text{ holds } f(\text{Shift}(t, j, S)) = \mathit{true} \text{ and } g(\text{Shift}(t, m, S)) = \mathit{true}, \\ \mathit{false}, & \text{otherwise.} \end{cases}$$

Let S be a non empty set and let f, g be sets. The functor $\text{Until}_0(f, g, S)$ yields an element of ModelSP (the infinite sequences of S) and is defined by the condition (Def. 54).

(Def. 54) Let t be a set. Suppose $t \in$ the infinite sequences of S. Then $\text{Until-univ}(t, \text{Fid}(f, \text{the infinite sequences of } S), \text{Fid}(g, \text{the infinite sequences of } S), S) = \mathit{true}$ if and only if $(\text{Fid}(\text{Until}_0(f, g, S), \text{the infinite sequences of } S))(t) = \mathit{true}$.

Let S be a non empty set. The functor $\text{Until}\, S$ yielding a binary operation on ModelSP (the infinite sequences of S) is defined by the condition (Def. 55).

(Def. 55) Let f, g be sets. Suppose $f \in$ ModelSP (the infinite sequences of S) and $g \in$ ModelSP (the infinite sequences of S). Then $(\text{Until}\, S)(f, g) =$

$\mathrm{Until}_0(f, g, S)$.

Let S be a non empty set. The functor $\vee_S$ yields a binary operation on ModelSP (the infinite sequences of S) and is defined by the condition (Def. 56).

(Def. 56) Let f, g be sets. Suppose $f \in$ ModelSP (the infinite sequences of S) and $g \in$ ModelSP (the infinite sequences of S). Then $\vee_S(f, g) = (\mathrm{Not}\, S)((\mathrm{And}\, S)((\mathrm{Not}\, S)(f), (\mathrm{Not}\, S)(g)))$.

The functor Release S yields a binary operation on ModelSP (the infinite sequences of S) and is defined by the condition (Def. 57).

(Def. 57) Let f, g be sets. Suppose $f \in$ ModelSP (the infinite sequences of S) and $g \in$ ModelSP (the infinite sequences of S). Then $(\mathrm{Release}\, S)(f, g) = (\mathrm{Not}\, S)((\mathrm{Until}\, S)((\mathrm{Not}\, S)(f), (\mathrm{Not}\, S)(g)))$.

Let S be a non empty set and let B_1 be a non empty subset of ModelSP (the infinite sequences of S). The functor $\mathrm{Model}_{\mathrm{LTL}}(S, B_1)$ yields an LTL-model structure and is defined as follows:

(Def. 58) $\mathrm{Model}_{\mathrm{LTL}}(S, B_1) = \langle$ModelSP (the infinite sequences of S), $B_1, \mathrm{And}\, S, \vee_S, \mathrm{Not}\, S, \mathrm{Next}\, S, \mathrm{Until}\, S, \mathrm{Release}\, S\rangle$.

In the sequel B_1 denotes a non empty subset of ModelSP (the infinite sequences of S), t denotes an element of the infinite sequences of S, and f, g denote assignations of $\mathrm{Model}_{\mathrm{LTL}}(S, B_1)$.

Let S be a non empty set, let B_1 be a non empty subset of ModelSP (the infinite sequences of S), let t be an element of the infinite sequences of S, and let f be an assignation of $\mathrm{Model}_{\mathrm{LTL}}(S, B_1)$. The predicate $t \models f$ is defined by:

(Def. 59) $(\mathrm{Fid}(f, \text{the infinite sequences of } S))(t) = \mathit{true}$.

Let S be a non empty set, let B_1 be a non empty subset of ModelSP (the infinite sequences of S), let t be an element of the infinite sequences of S, and let f be an assignation of $\mathrm{Model}_{\mathrm{LTL}}(S, B_1)$. We introduce $t \not\models f$ as an antonym of $t \models f$.

The following propositions are true:

(56) $f \vee g = \neg(\neg f \wedge \neg g)$ and $f \mathcal{R}\, g = \neg(\neg f\, \mathcal{U} \neg g)$.

(57) $t \models \neg f$ iff $t \not\models f$.

(58) $t \models f \wedge g$ iff $t \models f$ and $t \models g$.

(59) $t \models \mathcal{X} f$ iff $\mathrm{Shift}(t, 1) \models f$.

(60) $t \models f\, \mathcal{U}\, g$ if and only if there exists a natural number m such that for every natural number j such that $j < m$ holds $\mathrm{Shift}(t, j) \models f$ and $\mathrm{Shift}(t, m) \models g$.

(61) $t \models f \vee g$ iff $t \models f$ or $t \models g$.

(62) $t \models f\, \mathcal{R}\, g$ if and only if for every natural number m such that for every natural number j such that $j < m$ holds $\mathrm{Shift}(t, j) \models \neg f$ holds $\mathrm{Shift}(t, m) \models g$.

The non empty set AtomicFamily is defined as follows:

(Def. 60) $\text{AtomicFamily} = 2^{\text{atomic}_{\text{LTL}}}$.

Let a, t be sets. The functor $\text{AtomicFunc}(a, t)$ yielding an element of *Boolean* is defined as follows:

(Def. 61) $\text{AtomicFunc}(a, t) = \begin{cases} \mathit{true}, & \text{if } t \in \text{the infinite sequences of AtomicFamily} \\ & \text{and } a \in (\text{CastSeq}(t, \text{AtomicFamily}))(0), \\ \mathit{false}, & \text{otherwise.} \end{cases}$

Let a be a set. The functor $\text{AtomicAsgn}\, a$ yields an element of ModelSP (the infinite sequences of AtomicFamily) and is defined by:

(Def. 62) For every set t such that $t \in$ the infinite sequences of AtomicFamily holds $(\text{Fid}(\text{AtomicAsgn}\, a, \text{the infinite sequences of AtomicFamily}))(t) = \text{AtomicFunc}(a, t)$.

The non empty subset AtomicBasicAsgn of ModelSP (the infinite sequences of AtomicFamily) is defined by:

(Def. 63) $\text{AtomicBasicAsgn} = \{x \in \text{ModelSP}\,(\text{the infinite sequences of AtomicFamily})\colon \bigvee_{a\,:\,\text{set}}\ x = \text{AtomicAsgn}\, a\}$.

The function AtomicKai from $\text{atomic}_{\text{LTL}}$ into the basic assignations of $\text{Model}_{\text{LTL}}(\text{AtomicFamily}, \text{AtomicBasicAsgn})$ is defined as follows:

(Def. 64) For every set a such that $a \in \text{atomic}_{\text{LTL}}$ holds $(\text{AtomicKai})(a) = \text{AtomicAsgn}\, a$.

Let r be an element of the infinite sequences of AtomicFamily and let H be an LTL-formula. The predicate $r \models H$ is defined by:

(Def. 65) $r \models \text{Evaluate}(H, \text{AtomicKai})$.

Let r be an element of the infinite sequences of AtomicFamily and let H be an LTL-formula. We introduce $r \not\models H$ as an antonym of $r \models H$.

Let r be an element of the infinite sequences of AtomicFamily and let W be a subset of WFF_{LTL}. The predicate $r \models W$ is defined by:

(Def. 66) For every LTL-formula H such that $H \in W$ holds $r \models H$.

Let r be an element of the infinite sequences of AtomicFamily and let W be a subset of WFF_{LTL}. We introduce $r \not\models W$ as an antonym of $r \models W$.

Let W be a subset of WFF_{LTL}. The functor $\mathcal{X}\, W$ yielding a subset of WFF_{LTL} is defined as follows:

(Def. 67) $\mathcal{X}\, W = \{x; x \text{ ranges over LTL-formulae}\colon \bigvee_{u\,:\,\text{LTL-formula}} (u \in W \ \wedge\ x = \mathcal{X}\, u)\}$.

In the sequel r denotes an element of the infinite sequences of AtomicFamily.

We now state a number of propositions:

(63) If H is atomic, then $r \models H$ iff $H \in (\text{CastSeq}(r, \text{AtomicFamily}))(0)$.

(64) $r \models \neg H$ iff $r \not\models H$.

(65) $r \models H_1 \wedge H_2$ iff $r \models H_1$ and $r \models H_2$.

(66) $r \models H_1 \vee H_2$ iff $r \models H_1$ or $r \models H_2$.

(67) $r \models \mathcal{X}\, H$ iff $\mathrm{Shift}(r, 1) \models H$.

(68) $r \models H_1 \,\mathcal{U}\, H_2$ if and only if there exists a natural number m such that for every natural number j such that $j < m$ holds $\mathrm{Shift}(r, j) \models H_1$ and $\mathrm{Shift}(r, m) \models H_2$.

(69) $r \models H_1 \,\mathcal{R}\, H_2$ if and only if for every natural number m such that for every natural number j such that $j < m$ holds $\mathrm{Shift}(r, j) \models \neg H_1$ holds $\mathrm{Shift}(r, m) \models H_2$.

(70) $r \models \neg(H_1 \vee H_2)$ iff $r \models \neg H_1 \wedge \neg H_2$.

(71) $r \models \neg(H_1 \wedge H_2)$ iff $r \models \neg H_1 \vee \neg H_2$.

(72) $r \models H_1 \,\mathcal{R}\, H_2$ iff $r \models \neg(\neg H_1 \,\mathcal{U}\, \neg H_2)$.

(73) $r \not\models \neg H$ iff $r \models H$.

(74) $r \models \mathcal{X}\, \neg H$ iff $r \models \neg\, \mathcal{X}\, H$.

(75) $r \models H_1 \,\mathcal{U}\, H_2$ iff $r \models H_2 \vee H_1 \wedge \mathcal{X}(H_1 \,\mathcal{U}\, H_2)$.

(76) $r \models H_1 \,\mathcal{R}\, H_2$ iff $r \models H_1 \wedge H_2 \vee H_2 \wedge \mathcal{X}(H_1 \,\mathcal{R}\, H_2)$.

In the sequel W is a subset of $\mathrm{WFF}_{\mathrm{LTL}}$.

One can prove the following propositions:

(77) $r \models \mathcal{X}\, W$ iff $\mathrm{Shift}(r, 1) \models W$.

(78)(i) If H is atomic, then H is not negative and H is not conjunctive and H is not disjunctive and H does not have *next* operator and H does not have *until* operator and H does not have *release* operator,

(ii) if H is negative, then H is not atomic and H is not conjunctive and H is not disjunctive and H does not have *next* operator and H does not have *until* operator and H does not have *release* operator,

(iii) if H is conjunctive, then H is not atomic and H is not negative and H is not disjunctive and H does not have *next* operator and H does not have *until* operator and H does not have *release* operator,

(iv) if H is disjunctive, then H is not atomic and H is not negative and H is not conjunctive and H does not have *next* operator and H does not have *until* operator and H does not have *release* operator,

(v) if H has *next* operator, then H is not atomic and H is not negative and H is not conjunctive and H is not disjunctive and H does not have *until* operator and H does not have *release* operator,

(vi) if H has *until* operator, then H is not atomic and H is not negative and H is not conjunctive and H is not disjunctive and H does not have *next* operator and H does not have *release* operator, and

(vii) if H has *release* operator, then H is not atomic and H is not negative and H is not conjunctive and H is not disjunctive and H does not have *next* operator and H does not have *until* operator.

(79) For every element t of the infinite sequences of S holds $\mathrm{Shift}(t, 0) = t$.

(80) For every element s_1 of the infinite sequences of S holds $\mathrm{Shift}(\mathrm{Shift}(s_1, k), n) = \mathrm{Shift}(s_1, n + k)$.

(81) For every sequence s_1 of S holds $\mathrm{CastSeq}(\mathrm{CastSeq}\, s_1, S) = s_1$.

(82) For every element s_1 of the infinite sequences of S holds $\mathrm{CastSeq}\,\mathrm{CastSeq}(s_1, S) = s_1$.

(83) If H, $\neg H \in W$, then $r \not\models W$.

References

[1] Grzegorz Bancerek. The fundamental properties of natural numbers. *Formalized Mathematics*, 1(**1**):41–46, 1990.
[2] Grzegorz Bancerek. A model of ZF set theory language. *Formalized Mathematics*, 1(**1**):131–145, 1990.
[3] Grzegorz Bancerek. Models and satisfiability. *Formalized Mathematics*, 1(**1**):191–199, 1990.
[4] Grzegorz Bancerek and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Formalized Mathematics*, 1(**1**):107–114, 1990.
[5] Czesław Byliński. Binary operations. *Formalized Mathematics*, 1(**1**):175–180, 1990.
[6] Czesław Byliński. Functions and their basic properties. *Formalized Mathematics*, 1(**1**):55–65, 1990.
[7] Czesław Byliński. Functions from a set to a set. *Formalized Mathematics*, 1(**1**):153–164, 1990.
[8] Czesław Byliński. Some basic properties of sets. *Formalized Mathematics*, 1(**1**):47–53, 1990.
[9] E. M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 2000.
[10] Kazuhisa Ishida. Model checking. Part I. *Formalized Mathematics*, 14(**4**):171–186, 2006.
[11] Zinaida Trybulec. Properties of subsets. *Formalized Mathematics*, 1(**1**):67–71, 1990.
[12] Edmund Woronowicz. Many–argument relations. *Formalized Mathematics*, 1(**4**):733–737, 1990.

Received April 21, 2008