Semantics of MML Query

Grzegorz Bancerek
Białystok Technical University
Poland

Summary. In the paper the semantics of MML Query queries is given. The formalization is done according to [4].

MML identifier: MMLQUERY, version: 7.12.02 4.181.1147

The notation and terminology used here have been introduced in the following papers: [1], [5], [11], [8], [10], [6], [2], [3], [15], [13], [14], [9], [12], and [7].

1. Elementary Queries

Let $X$ be a set. A list of $X$ is a subset of $X$. An operation of $X$ is a binary relation on $X$.

Let $x$, $y$, $R$ be sets. The predicate $x, y \in R$ is defined by:

(Def. 1) $\langle x, y \rangle \in R$.

Let $x$, $y$, $R$ be sets. We introduce $x, y \notin R$ as an antonym of $x, y \in R$.

For simplicity, we use the following convention: $X$, $Y$, $z$, $s$ denote sets, $L$, $L_1$, $L_2$, $A$ denote lists of $X$, $x$ denotes an element of $X$, $O$, $O_2$, $O_3$ denote operations of $X$, and $m$ denotes a natural number.

The following proposition is true

(1) For all binary relations $R_1$, $R_2$ holds $R_1 \subseteq R_2$ iff for every $z$ holds $R_1^o z \subseteq R_2^o z$.

Let us consider $X$, $O$, $x$. We introduce $x \circ O$ as a synonym of $O^o x$.

Let us consider $X$, $O$, $x$. Then $x \circ O$ is a list of $X$.

One can prove the following proposition

---

1This work has been supported by the Polish Ministry of Science and Higher Education project “Managing a Large Repository of Computer-verified Mathematical Knowledge” (N N519 385136).
(2) \( x, y \in O \) iff \( y \in x \cdot O \).

Let us consider \( X, O, L \). We introduce \( L\cdot O \) as a synonym of \( O^\circ L \).

Let us consider \( X, O, L \). Then \( L\cdot O \) is a list of \( X \) and it can be characterized by the condition:

(Def. 2) \( L\cdot O = \bigcup \{ x \cdot O : x \in L \} \).

The functor \( L\&O \) yielding a list of \( X \) is defined as follows:

(Def. 3) \( L\&O = \bigcap \{ x \cdot O : x \in L \} \).

The functor \( L\ where \ O \) yielding a list of \( X \) is defined as follows:

(Def. 4) \( L\ where \ O = \{ x : \exists y (x, y \in O \land x \in L) \} \).

Let \( O_2 \) be an operation of \( X \). The functor \( L\ where \ O = O_2 \) yielding a list of \( X \) is defined as follows:

(Def. 5) \( L\ where \ O = O_2 = \{ x : \overline{x \cdot O} \subseteq \overline{x \cdot O_2} \land x \in L \} \).

The functor \( L\ where \ O \leq O_2 \) yielding a list of \( X \) is defined by:

(Def. 6) \( L\ where \ O \leq O_2 = \{ x : \overline{x \cdot O} \subseteq \overline{x \cdot O_2} \land x \in L \} \).

The functor \( L\ where \ O \geq O_2 \) yields a list of \( X \) and is defined by:

(Def. 7) \( L\ where \ O \geq O_2 = \{ x : \overline{x \cdot O} \subseteq \overline{x \cdot O_2} \land x \in L \} \).

The functor \( L\ where \ O \lt O_2 \) yielding a list of \( X \) is defined as follows:

(Def. 8) \( L\ where \ O \lt O_2 = \{ x : \overline{x \cdot O} \in \overline{x \cdot O_2} \land x \in L \} \).

The functor \( L\ where \ O \gt O_2 \) yields a list of \( X \) and is defined by:

(Def. 9) \( L\ where \ O \gt O_2 = \{ x : \overline{x \cdot O} \in \overline{x \cdot O_2} \land x \in L \} \).

Let us consider \( X, L, O, n \). The functor \( L\ where \ O = n \) yielding a list of \( X \) is defined as follows:

(Def. 10) \( L\ where \ O = n = \{ x : \overline{x \cdot O} = n \land x \in L \} \).

The functor \( L\ where \ O \leq n \) yielding a list of \( X \) is defined by:

(Def. 11) \( L\ where \ O \leq n = \{ x : \overline{x \cdot O} \subseteq n \land x \in L \} \).

The functor \( L\ where \ O \geq n \) yielding a list of \( X \) is defined as follows:

(Def. 12) \( L\ where \ O \geq n = \{ x : n \subseteq \overline{x \cdot O} \land x \in L \} \).

The functor \( L\ where \ O \lt n \) yields a list of \( X \) and is defined as follows:

(Def. 13) \( L\ where \ O \lt n = \{ x : \overline{x \cdot O} \in n \land x \in L \} \).

The functor \( L\ where \ O \gt n \) yields a list of \( X \) and is defined by:

(Def. 14) \( L\ where \ O \gt n = \{ x : n \subseteq \overline{x \cdot O} \land x \in L \} \).

One can prove the following propositions:

(3) \( x \in L\ where \ O \) iff \( x \in L \) and \( x \neq \emptyset \).
(4) \( L\ where \ O \subseteq L \).
(5) If \( L \subseteq \text{dom} \cdot O \), then \( L\ where \ O = L \).
(6) If \( n \neq 0 \) and \( L_1 \subseteq L_2 \), then \( L_1\ where \ O \geq n \subseteq L_2\ where \ O \).
(7) \( L\ where \ O \geq 1 = L\ where \ O \).
We introduce $A$ as a synonym of $A$. Let us consider $A$. Then $A$ is a list of $X$. Then $A$ is a list of $X$.

We now state several propositions:

1. If $L_1 \subseteq L_2$, then $L_1 \text{where } O > n \subseteq L_2 \text{where } O$.
2. If $n \neq 0$ and $L_1 \subseteq L_2$, then $L_1 \text{where } O = n \subseteq L_2 \text{where } O$.
3. $L \text{where } O \geq n + 1 = L \text{where } O > n$.
4. $L \text{where } O \leq n = L \text{where } O < n + 1$.
5. If $n \leq m$ and $L_1 \subseteq L_2$ and $O_1 \subseteq O_2$, then $L_1 \text{where } O_1 \geq m \subseteq L_2 \text{where } O_2 \geq n$.
6. If $n \leq m$ and $L_1 \subseteq L_2$ and $O_1 \subseteq O_2$, then $L_1 \text{where } O_1 > m \subseteq L_2 \text{where } O_2 > n$.
7. If $n \leq m$ and $L_1 \subseteq L_2$ and $O_1 \subseteq O_2$, then $L_1 \text{where } O_2 \leq n \subseteq L_2 \text{where } O_1 \leq m$.
8. If $n \leq m$ and $L_1 \subseteq L_2$ and $O_1 \subseteq O_2$, then $L_1 \text{where } O_2 < n \subseteq L_2 \text{where } O_1 < m$.
9. If $O_1 \subseteq O_2$ and $L_1 \subseteq L_2$ and $O \subseteq O_3$, then $L_1 \text{where } O \geq O_2 \subseteq L_2 \text{where } O_3 \geq O_1$.
10. If $O_1 \subseteq O_2$ and $L_1 \subseteq L_2$ and $O \subseteq O_3$, then $L_1 \text{where } O > O_2 \subseteq L_2 \text{where } O_3 > O_1$.
11. If $O_1 \subseteq O_2$ and $L_1 \subseteq L_2$ and $O \subseteq O_3$, then $L_1 \text{where } O_3 \leq O_1 \subseteq L_2 \text{where } O \leq O_2$.
12. If $O_1 \subseteq O_2$ and $L_1 \subseteq L_2$ and $O \subseteq O_3$, then $L_1 \text{where } O_3 < O_1 \subseteq L_2 \text{where } O < O_2$.
13. $L \text{where } O > O_1 \subseteq L \text{where } O$.
14. If $O_1 \subseteq O_2$ and $L_1 \subseteq L_2$, then $L_1 \text{where } O_1 \subseteq L_2 \text{where } O_2$.
15. $a \in L|O$ iff there exists $b$ such that $a \in b \ O$ and $b \in L$.

Let us consider $X$, $A$, $B$. We introduce $A$ and $B$ as a synonym of $A \cap B$. We introduce $A \ or \ B$ as a synonym of $A \cup B$. We introduce $A \ but\ not \ B$ as a synonym of $A \ \setminus \ B$.

Let us consider $X$, $A$, $B$. Then $A$ and $B$ is a list of $X$. Then $A$ or $B$ is a list of $X$. Then $A$ but not $B$ is a list of $X$.
2. Operations

One can prove the following two propositions:

(29) For all operations $O_1, O_2$ of $X$ such that for every $x$ holds $x \cdot O_1 = x \cdot O_2$ holds $O_1 = O_2$.

(30) For all operations $O_1, O_2$ of $X$ such that for every $L$ holds $L \cdot O_1 = L \cdot O_2$ holds $O_1 = O_2$.

The functor $\text{not} \ O$ yielding an operation of $X$ is defined as follows:

(Def. 15) For every $L$ holds $L \cdot \text{not} \ O = \bigcup \{(x \cdot O = \emptyset, \{x\}, \emptyset) : x \in L\}$.

Let us consider $X$ and let $O_1, O_2$ be operations of $X$. We introduce $O_1 \ \text{and} \ O_2$ as a synonym of $O_1 \cap O_2$. We introduce $O_1 \ \text{or} \ O_2$ as a synonym of $O_1 \cup O_2$. We introduce $O_1 \ \text{butnot} \ O_2$ as a synonym of $O_1 \ \backslash \ O_2$. We introduce $O_1 \ | \ O_2$ as a synonym of $O_1 \cdot O_2$.

Let us consider $X$ and let $O_1, O_2$ be operations of $X$. Then $O_1 \ \text{and} \ O_2$ is an operation of $X$ and it can be characterized by the condition:

(Def. 16) For every $L$ holds $L \cdot (O_1 \ \text{and} \ O_2) = \bigcup \{(x \cdot O_1) \ \text{and} \ (x \cdot O_2) : x \in L\}$.

Then $O_1 \ \text{or} \ O_2$ is an operation of $X$ and it can be characterized by the condition:

(Def. 17) For every $L$ holds $L \cdot (O_1 \ \text{or} \ O_2) = \bigcup \{(x \cdot O_1) \ \text{or} \ (x \cdot O_2) : x \in L\}$.

Then $O_1 \ \text{butnot} \ O_2$ is an operation of $X$ and it can be characterized by the condition:

(Def. 18) For every $L$ holds $L \cdot (O_1 \ \text{butnot} \ O_2) = \bigcup \{(x \cdot O_1) \ \text{butnot} \ (x \cdot O_2) : x \in L\}$.

Then $O_1 \ \text{and} \ O_2$ is an operation of $X$ and it can be characterized by the condition:

(Def. 19) For every $L$ holds $L \cdot (O_1 \ | \ O_2) = L \cdot O_1 \ | \ O_2$.

The functor $O_1 \ \text{and} \ O_2$ yielding an operation of $X$ is defined as follows:

(Def. 20) For every $L$ holds $L \cdot (O_1 \ \text{and} \ O_2) = \bigcup \{(x \cdot O_1) \ \text{and} \ O_2 : x \in L\}$.

We now state a number of propositions:

(31) $x \cdot (O_1 \ \text{and} \ O_2) = (x \cdot O_1) \ \text{and} \ (x \cdot O_2)$.

(32) $x \cdot (O_1 \ \text{or} \ O_2) = (x \cdot O_1) \ \text{or} \ (x \cdot O_2)$.

(33) $x \cdot (O_1 \ \text{butnot} \ O_2) = (x \cdot O_1) \ \text{butnot} \ (x \cdot O_2)$.

(34) $x \cdot (O_1 \ | \ O_2) = (x \cdot O_1) \ | \ O_2$.

(35) $x \cdot (O_1 \ & \ O_2) = (x \cdot O_1) \ & \ O_2$.

(36) $z, s \in \text{not} \ O$ iff $z = s$ and $z \in X$ and $z \notin \text{dom} \ O$.

(37) $\text{not} \ O = \text{id}_{X \ \backslash \ \text{dom} \ O}$.

(38) $\text{dom} \text{not} \ O = \text{dom} \ O$.

(39) $L \ \text{where not not} \ O = L \ \text{where} \ O$.

(40) $L \ \text{where} \ O = 0 = L \ \text{where not} \ O$.

(41) $\text{not not} \ O = \text{not} \ O$.

(42) $\text{not} O_1 \ \text{or} \ \text{not} O_2 \subseteq \text{not} (O_1 \ \text{and} \ O_2)$.
(43) \( \text{not}(O_1 \text{ or } O_2) = \text{not } O_1 \text{ and } \text{not } O_2. \)

(44) If \( \text{dom} O_1 = X \) and \( \text{dom} O_2 = X \), then \( (O_1 \text{ or } O_2) \& O = (O_1 \& O) \text{ and } (O_2 \& O). \)

Let us consider \( X, O \). We say that \( O \) is filtering if and only if:

(Def. 21) \( O \subseteq \text{id}_X. \)

Next we state the proposition

(45) \( O \) is filtering iff \( O = \text{id}_{\text{dom } O}. \)

Let us consider \( X, O \). Note that \( \text{not } O \) is filtering.

Let us consider \( X \). Note that there exists an operation of \( X \) which is filtering.

In the sequel \( F_1, F_2 \) denote filtering operations of \( X \).

Let us consider \( X, F, O \). One can check the following observations:

* \( F \text{ and } O \) is filtering,
* \( O \text{ and } F \) is filtering, and
* \( F \text{ but not } O \) is filtering.

Let us consider \( X, F_1, F_2 \). One can verify that \( F_1 \text{ or } F_2 \) is filtering.

(46) If \( z \in x F \), then \( z = x. \)

(47) \( L|F = L \text{ where } F. \)

(48) \( \text{not not } F = F. \)

(49) \( \text{not}(F_1 \text{ and } F_2) = \text{not } F_1 \text{ or } \text{not } F_2. \)

(50) \( \text{dom}(O \text{ or } \text{not } O) = X. \)

(51) \( F \text{ or } \text{not } F = \text{id}_X. \)

(52) \( O \text{ and } \text{not } O = \emptyset. \)

(53) \( (O_1 \text{ or } O_2) \text{ and } \text{not } O_1 \subseteq O_2. \)

3. Rough Queries

Let \( A \) be a finite sequence and let \( a \) be a set. The functor \( \#\text{occurrences}(a, A) \) yielding a natural number is defined as follows:

(Def. 22) \( \#\text{occurrences}(a, A) = \{i : i \in \text{dom } A \land a \in A(i)\}. \)

We now state two propositions:

(54) For every finite sequence \( A \) and for every set \( a \) holds \( \#\text{occurrences}(a, A) \leq \text{len } A. \)

(55) For every finite sequence \( A \) and for every set \( a \) holds \( A \neq \emptyset \) and \( \#\text{occurrences}(a, A) = \text{len } A \) iff \( a \in \bigcap \text{rng } A. \)

The functor \( \text{max#} A \) yielding a natural number is defined as follows:

(Def. 23) For every set \( a \) holds \( \#\text{occurrences}(a, A) \leq \text{max# } A \) and for every \( n \) such that for every set \( a \) holds \( \#\text{occurrences}(a, A) \leq n \) holds \( \text{max# } A \leq n. \)
(56) For every finite sequence $A$ holds $\text{max}\#A \leq \text{len} A$.
(57) For every finite sequence $A$ and for every set $a$ such that $\#\text{occurrences}(a, A) = \text{len} A$ holds $\text{max}\#A = \text{len} A$.

Let us consider $X$, let $A$ be a finite sequence of elements of $2^X$, and let $n$ be a natural number. The functor $\text{rough}_n(A)$ yields a list of $X$ and is defined as follows:

(Def. 24) $\text{rough}_n(A) = \{ x : n \leq \#\text{occurrences}(x, A) \}$ if $X \neq \emptyset$.

Let $m$ be a natural number. The functor $\text{rough}_{n-m}(A)$ yields a list of $X$ and is defined by:

(Def. 25) $\text{rough}_{n-m}(A) = \{ x : n \leq \#\text{occurrences}(x, A) \land \#\text{occurrences}(x, A) \leq m \}$ if $X \neq \emptyset$.

Let us consider $X$ and let $A$ be a finite sequence of elements of $2^X$. The functor $\text{rough}(A)$ yielding a list of $X$ is defined by:

(Def. 26) $\text{rough}(A) = \text{rough}\text{max}\#(A)$.

Next we state several propositions:

(58) For every finite sequence $A$ of elements of $2^X$ holds $\text{rough}_{n-\text{len}}A(A) = \text{rough}_n(A)$.
(59) For every finite sequence $A$ of elements of $2^X$ such that $n \leq m$ holds $\text{rough}_m(A) \subseteq \text{rough}_n(A)$.
(60) Let $A$ be a finite sequence of elements of $2^X$ and $n_1, n_2, m_1, m_2$ be natural numbers. If $n_1 \leq m_1$ and $n_2 \leq m_2$, then $\text{rough}_{m_1-m_2}(A) \subseteq \text{rough}_{n_1-n_2}(A)$.
(61) For every finite sequence $A$ of elements of $2^X$ holds $\text{rough}_{n-m}(A) \subseteq \text{rough}_n(A)$.
(62) For every finite sequence $A$ of elements of $2^X$ such that $A \neq \emptyset$ holds $\text{rough}\text{len}A(A) = \bigcap \text{rng}A$.
(63) For every finite sequence $A$ of elements of $2^X$ holds $\text{rough}_1(A) = \bigcup A$.
(64) For all lists $L_1, L_2$ of $X$ holds $\text{rough}_2(\langle L_1, L_2 \rangle) = L_1$ and $L_2$.
(65) For all lists $L_1, L_2$ of $X$ holds $\text{rough}_1(\langle L_1, L_2 \rangle) = L_1$ or $L_2$.

4. Constructor Database

We introduce constructor databases which are extensions of 1-sorted structures and are systems

$\langle \text{a carrier, constructors, a ref-operation} \rangle$,

where the carrier is a set, the constructors constitute a list of the carrier, and the ref-operation is a relation between the carrier and the constructors.

Let $X$ be a 1-sorted structure. A list of $X$ is a list of the carrier of $X$. An operation of $X$ is an operation of the carrier of $X$. 
Let us consider $X$, let $S$ be a subset of $X$, and let $R$ be a relation between $X$ and $S$. The functor $\oplus R$ yields a binary relation on $X$ and is defined by:

(Def. 27) $\oplus R = R$.

Let $X$ be a constructor database and let $a$ be an element of $X$. The functor $a \mathit{ref}$ yielding a list of $X$ is defined as follows:

(Def. 28) $a \mathit{ref} = a \oplus \mathit{the \, ref-operation \, of \, X}$.

The functor $a \mathit{ocur}$ yields a list of $X$ and is defined as follows:

(Def. 29) $a \mathit{ocur} = a \mathit{ref} \setminus \mathit{the \, ref-operation \, of \, X}$.

The following proposition is true

(66) For every constructor database $X$ and for all elements $x, y$ of $X$ holds $x \in y \mathit{ref}$ iff $y \in x \mathit{ocur}$.

Let $X$ be a constructor database. We say that $X$ is ref-finite if and only if:

(Def. 30) For every element $x$ of $X$ holds $x \mathit{ref}$ is finite.

One can verify that every constructor database which is finite is also ref-finite.

Let us note that there exists a constructor database which is finite and non empty.

Let $X$ be a ref-finite constructor database and let $x$ be an element of $X$. Observe that $x \mathit{ref}$ is finite.

Let $X$ be a constructor database and let $A$ be a finite sequence of elements of the constructors of $X$. The functor $\mathit{atleast}(A)$ yielding a list of $X$ is defined by:

(Def. 31) $\mathit{atleast}(A) = \{ x \in X : \mathit{rng} A \subseteq x \mathit{ref} \}$ if the carrier of $X \neq \emptyset$.

The functor $\mathit{atmost}(A)$ yielding a list of $X$ is defined as follows:

(Def. 32) $\mathit{atmost}(A) = \{ x \in X : x \mathit{ref} \subseteq \mathit{rng} A \}$ if the carrier of $X \neq \emptyset$.

The functor $\mathit{exactly}(A)$ yields a list of $X$ and is defined by:

(Def. 33) $\mathit{exactly}(A) = \{ x \in X : x \mathit{ref} = \mathit{rng} A \}$ if the carrier of $X \neq \emptyset$.

Let $n$ be a natural number. The functor $\mathit{atleast \, minus \, n}(A)$ yields a list of $X$ and is defined by:

(Def. 34) $\mathit{atleast \, minus \, n}(A) = \{ x \in X : \mathit{rng} A \setminus x \mathit{ref} \leq n \}$ if the carrier of $X \neq \emptyset$.

Let $X$ be a ref-finite constructor database, let $A$ be a finite sequence of elements of the constructors of $X$, and let $n$ be a natural number. The functor $\mathit{atmost \, plus \, n}(A)$ yields a list of $X$ and is defined by:

(Def. 35) $\mathit{atmost \, plus \, n}(A) = \{ x \in X : x \mathit{ref} \setminus \mathit{rng} A \leq n \}$ if the carrier of $X \neq \emptyset$.

Let $m$ be a natural number. The functor $\mathit{exactly \, plus \, minus \, n \, m}(A)$ yielding a list of $X$ is defined by:
Exactly plus $n$ minus $m(A) = \{x \in X : \text{ref}\{\text{rng}\{A\} \leq n \land \text{rng}\{A\} \setminus \text{ref}\{x\} \leq m\}\}$ if the carrier of $X \neq \emptyset$.

In the sequel $X$ denotes a constructor database, $x$ denotes an element of $X$, $B$ denotes a finite sequence of elements of the constructors of $Y$, and $y$ denotes an element of $Y$.

The following propositions are true:

(67) atleast minus $0(A) = \text{atleast}(A)$.

(68) atmost plus $0(B) = \text{atmost}(B)$.

(69) exactly plus $0$ minus $0(B) = \text{exactly}(B)$.

(70) If $n \leq m$, then atleast minus $n(A) \subseteq \text{atleast minus } m(A)$.

(71) If $n \leq m$, then atmost plus $n(B) \subseteq \text{atmost plus } m(B)$.

(72) For all natural numbers $n_1, n_2, m_1, m_2$ such that $n_1 \leq m_1$ and $n_2 \leq m_2$ holds exactly plus $n_1$ minus $n_2(B) \subseteq \text{exactly plus } m_1$ minus $m_2(B)$.

(73) atleast(A) $\subseteq \text{atleast minus } n(A)$.

(74) atmost(B) $\subseteq \text{atmost plus } n(B)$.

(75) exactly(B) $\subseteq \text{exactly plus } n$ minus $m(B)$.

(76) exactly(A) = $\text{atleast}(A)$ and atmost(A).

(77) exactly plus $n$ minus $m(B) = \text{atleast minus } m(B) \land \text{atmost plus } n(B)$.

(78) If $A \neq \emptyset$, then atleast(A) = $\bigcap \{x \text{ occur : } x \in \text{rng}\{A\}\}$.

(79) For all elements $c_1, c_2$ of $X$ such that $A = \langle c_1, c_2 \rangle$ holds atleast(A) = $c_1 \text{ occur and } c_2 \text{ occur}$.

References

Bo Zhang, Hiroshi Yamazaki, and Yatsuka Nakamura. Set sequences and monotone class. 

Received December 18, 2011