Marking Solvable Variables

Murat Sinan Aygün

August 21, 2021

Marking Solvable Variables

Murat Sinan Aygün[1]

email sinan_aygun@yahoo.com

Abstract. The places of solvable variables are marked by a special function symbol. After they are substituted, a transformation is applied to make the result independent of it for generality.

Keywords: Logic · Lambda calculus · Recursive specifications · The most general substitution · Non-determinism

1 Introduction

Automated theorem proving [1, 2] began after the discovery of unification. The key point is the inference mechanism. A logical statement L_2 can be deduced from another logical statement L_1, in other words, L_2 is a consequence of L_1, if every truth value which makes L_1 correct also makes L_2 correct.

For example, $(Q \wedge R)$ can be deduced from $(P_1 \wedge Q)$ and $(P_1 \Rightarrow R)$. For the first statement to be true, P_1 and Q should be true. R should also be true since P_1 is true. Because Q and R are true, the second statement should be true. That means, the second statement can be deduced from the first or is a consequence of the first.

When logical statements contain variables, unification is used in the inference mechanism. For example, if the atom statements Q, R, P_1 and P_2 contain variables, unification is also involved. $(Q \wedge R)$ can be deduced from $(P_1 \wedge Q)$ and $(P_2 \Rightarrow R)$ only when $\omega(P_1) = \omega(P_2)$.

Logic programming languages such as prolog [3] use the resolution principle as an inference engine.

$$(P(x) \vee Q(x)) \wedge (\neg P(a) \vee R(y)) \Rightarrow Q(a) \vee R(y) \tag{1}$$

On the other hand, based on λ-term, which acts like a function, λ calculus [4] is another computation mechanism and plays an important role in proof systems. For example, λ-terms embedded in proof system enhance computations [5].

Similarly, recursively defined equivalence relation $\approx$, which is used to check whether two terms are equivalent to each other, enhances computations. However, its trivial definition which works fine for terms that do not contain variables leads to non-determinism for terms that contain variables.

Definition 1. *The recursive relation $\approx$ is defined as follows*

1. $a \approx b$
2. $b \approx a$
3. $a \approx a$
4. $b \approx b$
5. *if* $f(x) \approx f(y)$ $x \approx y$
6. *if* $g(x_1,x_2) \approx g(x_3,x_4)$ $x_1 \approx x_3$ *and* $x_2 \approx x_4$

Example 1. The rules of Definition 1 are applied to (2)

$$g(g(a,b),f(b)) \approx g(g(b,a),f(a)) \tag{2}$$

- g(g(a,b),f(b)) $\approx$ g(g(b,a),f(a)) (using rule 6)
- g(a,b) $\approx$ g(b,a) and f(b) $\approx$ f(a) (using rules 5 and 6)
- a $\approx$ b and b $\approx$ a and b $\approx$ a

Example 2. Some terms may be variables as in (3)

$$g(x,f(b)) \approx \ g(f(y),f(a)) \tag{3}$$

- g(x,f(b)) $\approx$ g(f(y),f(a)) (using rule 6)
- x $\approx$ f(y) and f(b) $\approx$ f(a) (using rule 5)
- x $\approx$ f(y) and b $\approx$ a (using rule 2)
- x $\approx$ f(y) (if x is substituted by f(x_1), then by using rule 5)
- $x_1 \approx$ y (if x_1 is substituted by f(x_2) and y is substituted by f(y_1), then by using rule 5)
- $x_2 \approx y_1$

$x_2 \approx y_1$ in Example 2 leads to an infinite branch. The aim here is to find a substitution ω for s and t terms such that ω(s) $\approx$ ω(t) holds. Although the rules in Definition 1 can be used for this purpose, they lead to infinite branches as in Example 2. The question here is how the relation $\approx$ can be changed to produce a general substitution by eliminating unnecessary ones.

2 The Redefinition Of The Relation $\approx$

2.1 Preliminaries

In the following, the prior knowledge this paper needs is given. In particular, terms, λ-terms, logical terms, substitution, signature are defined.

Definition 2. *τ is used for base types and other types are built from τ by using the symbol $\rightarrow$ such as $\tau \rightarrow \tau$, $\tau \rightarrow \tau \rightarrow \tau$, ...,$\tau \rightarrow \tau \rightarrow$....$\rightarrow \tau$. The capital letters A_1, A_2, A_3,.. are used to represent an unknown type in type expressions and can be τ, $\tau \rightarrow \tau$, $\tau \rightarrow \tau \rightarrow \tau$, ..., $\tau \rightarrow \tau \rightarrow$....$\rightarrow \tau$. The symbol o is used as the type of logical arguments. τ and o are used as base types, the first is used for terms, the second for logical proposition, in other words, true and false statements. On the other hand, A_1, A_2,...are variables in type expressions to denote something whose value is unknown and will be set during the computations. They can take any value from the infinite set $\{\tau, \tau \rightarrow \tau, \tau \rightarrow \tau \rightarrow \tau, ..., \tau \rightarrow \tau \rightarrow ....\rightarrow \tau,..\}$.*

In order to prevent any ambiguity, the type expression $k_1 \rightarrow k_2 \rightarrow k_3$ can be interpreted as $k_1 \rightarrow (k_2 \rightarrow k_3)$.

Definition 3. *Terms are defined as follows. The constants a and b are of the type τ. The function f is of the type $\tau \rightarrow \tau$ and g is of the type $\tau \rightarrow \tau \rightarrow \tau$. Logical terms are defined as follows. The propositions p, q, r are of the type $\tau \rightarrow o$ and called atomic logical terms. The logical operators conjunction, disjunction and implication, which are of the type $o \rightarrow o \rightarrow o$, are respectively represented by the symbols $\wedge$, $\vee$ and $\Rightarrow$. The operators $\wedge$, $\vee$ and $\Rightarrow$ are used in infix notation. p or q is represented as $p \vee q$, p and q as $p \wedge q$ and p implies q as $p \Rightarrow q$.*

Definition 4. *If m is a term of the type $k_1 \rightarrow k_2$ and n is a term of the type k_1, then m n is a term of the type k_2.*

Example 3. Assume that the variables x and y are of the type τ. g(x,b), f(f(y)), f(f(b)) are valid terms of the type τ whereas p(g(x,y)), q(f(f(b))), p(g(x,y)) $\wedge$ q(f(y)), p(g(x,y)) $\vee$ q(f(y)), p(g(x,y)) $\Rightarrow$ q(f(y)) are valid logical terms of the type o.

In the following, the symbols x, x_1, x_2, x_3, x_4, y, y_1, v, v_1 represent variables, t, t_1, t_2, t_3 terms, k_1, k_2 types and ω, ω_1, ω_2 substitutions.

Definition 5. *The function symbol ν is of the type $A \rightarrow \tau$ and is used to mark the places of solvable variables.*

Definition 6. *Substituting a variable x_1 by a term t_1 is represented by $x_1 \rightarrow t_1$. Substitution is a mapping from variables to terms and can be denoted by $\{x_1 \rightarrow t_1,\ x_2 \rightarrow t_2,\ x_3 \rightarrow t_3,...\}$. Substitution is called renaming if t_1, t_2, t_3,.. are all variables. Applying substitution ω to an expression e, meaning replacing the variables of e with the terms in the range of ω if they are in the domain of ω, can be denoted by $\omega(e)$. $\omega_1(\omega_2)$ is the resulting substitution after applying ω_1 to each term in the range of ω_2.*

Example 4.

$$\{x_1 \rightarrow f(a), y_1 \rightarrow g(x_2, x_3)\}(p(x_1) \vee q(y_1)) = p(f(a)) \vee q(g(x_2, x_3)) \quad (4)$$

Example 5. Assume that r_1 and r_2 are unary relations.

$$\{x_1 \rightarrow f(a), y_1 \rightarrow g(b, x_3)\}(r_1(x_1)\ and\ r_2(y_1)) = r_1(f(a))\ and\ r_2(g(b, x_3)) \quad (5)$$

Definition 7. *Substitution φ is more general than substitution ω if there is a substitution ϕ such that $\omega = \phi(\varphi)$.*

Example 6.

$$\{x_2 \rightarrow g(x_3, x_4)\}\{x_1 \rightarrow f(x_2)\} = \{x_1 \rightarrow f(g(x_3, x_4))\} \quad (6)$$

$\{x_1 \rightarrow f(x_2)\}$ is more general than $\{x_1 \rightarrow f(f(x_3))\}$.

Definition 8. *A term t is linear if each variable of t is different.*

Definition 9. *Abstraction and application operations are defined as follows. If s is a term of k_2 and x is of k_1, $\lambda x.s$ is a term of $k_1 \rightarrow k_2$ and used to denote functions. On the other hand, application operation, which is used to evaluate functions, is denoted by*

$$(\lambda x.t_1)t_2 = \{x \rightarrow t_2\}t_1 \tag{7}$$

If $(\lambda x.t_1)$ is a term of $k_1 \rightarrow k_2$ and t_2 is a term of k_1, $\{x \rightarrow t_2\}$ t_1 is a term of k_2.

$(\lambda x_1.\lambda x_2.t_1)$ t_2 t_3 can be interpreted as $(((\lambda x_1.\lambda x_2.t_1)$ $t_2)$ $t_3)$.

Example 7. Assume that x, y are variables of the type τ. λx.f(x), λx.λy.g(x,y) are lambda terms of the types $\tau \rightarrow \tau$, $\tau \rightarrow \tau \rightarrow \tau$ respectively.

Definition 10. *Signature, denoted by Σ, is a set of constants and function symbols.*

Variables can only be substituted by the elements of Σ.

Example 8. By Definition 3 and 5,

$$\Sigma = \{a, b, f, g, p, q, r, \wedge, \vee, \Rightarrow, \nu\} \tag{8}$$

The term g(x,y) can be g(x,b) if {y → b} is applied or g(d,y) if {x → d} is applied, but g(d,y) is not a valid term since {x → d} is not a valid substitution (d is not an element of Σ).

2.2 The Redefinition Of The Relation ≈ And The Transformations

For the desired computations, the relation structure will be extended.

Definition 11. *A goal statement is either a substitution or a relation, which is called an atomic goal statement. If A_1, A_2, A_3 are goal statements,*

- *A_1 and A_2*
- *if A_1 else A_2*
- *if A_1 else A_2 else A_3*
- *$\lambda x.A_1$*

are also goal statements. If B_1 is a relation and B_2 is a goal statement,

- *B_1*
- *if B_1 B_2*

are relation statements.

Definition 12. *The goal statements are used as follows.*

$$if\ A_1\ else\ A_2\ else\ A_3 \tag{9}$$

$$if\ A_1\ else\ A_2 \tag{10}$$

$$A_1\ and\ A_2 \tag{11}$$

$$\lambda x.A_1 \tag{12}$$

The expression 9 is used to denote the following. If A_1 is satisfied, do not consider A_2 and A_3. If A_1 fails, consider A_2. If A_2 is satisfied, do not consider A_3. If A_2 fails, consider A_3. Similarly, for 10, consider A_2 only if A_1 fails, if A_1 is satisfied, do not consider A_2. On the other hand, for 11, consider both A_1 and A_2. For the expression 12, consider $\{x \to c_1\}A_1$ where c_1 is a new constant of the type τ.

The first is called **ELSE 3** rule, the second **ELSE 2**, the third **AND** and the fourth **ABSTRACTION**, shortly **ABS**.

Example 9. Given the relation $\asymp$

1. a $\asymp$ b
2. b $\asymp$ a
3. a $\asymp$ a
4. b $\asymp$ b
5. if ν(x) $\asymp$ ν(y) {y $\to$x}
6. if x $\asymp$ y λc_1.((x c_1) $\asymp$ (y c_1))
7. if f(x) $\asymp$ f(y) x $\asymp$ y
8. if g(x_1,x_2) $\asymp$ g(x_3,x_4) x_1 $\asymp$ x_3 and x_2 $\asymp$ x_4

Consider the expression 13

$$\lambda c_1.g(\nu(c_1), b) \asymp \lambda c_1.g(\nu(c_1), a) \tag{13}$$

where both λc_1.g(ν(c_1),b), λc_1.g(ν(c_1),a) are of the type $\tau \to \tau$.

- λc_1.g(ν(c_1),b) $\asymp$ λc_1.g(ν(c_1),a) where Σ = {a, b, f, g, p, q, r, $\wedge$, $\vee$, $\Rightarrow$, ν} (using rule 6 and **ABS**)
- g(ν(c_1),b) $\asymp$ g(ν(c_1),a) where Σ = {a, b, f, g, p, q, r, $\wedge$, $\vee$, $\Rightarrow$, ν, c_1} (using rule 8)
- ν(c_1) $\asymp$ ν(c_1) and b $\asymp$ a (using rule 2)
- ν(c_1) $\asymp$ ν(c_1) and true (using rule 5)
- true and true

Similarly,

- λc_1.g(ν(c_1),b) $\asymp$ x where Σ = {a, b, f, g, p, q, r, $\wedge$, $\vee$, $\Rightarrow$, ν} (using rule 6 and **ABS**)
- g(ν(c_1),b) $\asymp$ (x c_1) where Σ = {a, b, f, g, p, q, r, $\wedge$, $\vee$, $\Rightarrow$, ν, c_1} (if x is substituted by λc_1.g(x_1,x_2) and using rule 8)

- $\nu(c_1) \asymp x_1$ and $b \asymp x_2$ (if x_1 is substituted by $\nu(x_3)$ and x_2 is substituted by b)
- $\nu(c_1) \asymp \nu(x_3)$ and $b \asymp b$ (using rule 5 and 4)
- $\{x_3 \rightarrow c_1\}$and true

The following is given because the rules of $\approx$ are applied to terms in well defined form.

Definition 13. *A well defined form is either the constants a, b or $\nu(x)$, which are called atomic well defined forms. If t_1 and t_2 are well defined forms, then $f(t_1)$, $p(t_1)$, $q(t_1)$, $r(t_1)$, $g(t_1,t_2)$, $(t_1 \wedge t_2)$, $(t_1 \vee t_2)$ and $(t_1 \Rightarrow t_2)$ are also called well defined forms.*

Example 10. The terms $g(\nu(x),f(b))$, $g(f(\nu(y)),f(a))$ are well defined forms whereas the terms $g(\nu(f(b)),f(b))$, $g(f(\nu(y)),x)$ are not well defined forms.

The function ν is treated differently from constants or other functions. The following definition is given to denote a unary relation that holds for constants or functions other than ν.

Definition 14. *The one-place relation μ holds for the terms a, b, $f(t_1)$, $g(t_1,t_2)$ whereas it does not hold for $\nu(x)$.*

Example 11. μ f(a), μ $f(\nu(x))$, μ a, μ $g(a,\nu(x_2))$ return true whereas μ $\nu(x_2)$ returns false.

Example 12. The rules in $\approx$ (see Supplementary Material) are applied to the well defined forms in 14 as follows.

$$g(\nu(x), f(b)) \approx g(f(\nu(y)), f(a)) \tag{14}$$

- $g(\nu(x),f(b)) \approx g(f(\nu(y)),f(a))$ where $\Sigma = \{a, b, f, g, p, q, r, \wedge, \vee, \Rightarrow, \nu\}$ (using rule 9)
- $\nu(x) \approx f(\nu(y))$ and $f(b) \approx f(a)$ (using rule 8)
- $\nu(x) \approx f(\nu(y))$ and $b \approx a$ (using rule 2)
- $\nu(x) \approx f(\nu(y))$ and true (using rule 6)
- μ $f(\nu(y))$ and $\{x \rightarrow f(\nu(y))\}$ and true
- true and $\{x \rightarrow f(\nu(y))\}$ and true

$\nu(x) \approx f(\nu(y))$ in Example 12 leads to one solution only. The type of x is unknown before the substitution. After the substitution, its type becomes τ. When the rules of $\approx$ are applied, the variable x in $\nu(x)$ should not be substituted before in order to make things clear. This necessitates the specific forms. Given $s \approx t$, s and t should be linear and the variables of s should be different from those of t.

Additionally, when $s \approx t$ succeeds, the condition that any term should be a well defined form is not guaranteed. The condition is satisfied after a transformation step, which is called normalization.

Definition 15. *Let x_1, x_2,..., x_n be variables of a term t_1. Also assume that y_1, y_2,..., y_n are new variables and $\omega = \{x_1 \rightarrow y_1,\ x_2 \rightarrow y_2,\ ...,x_n \rightarrow y_n\}$. Let t_2 be the resulting term after removing all ν functions not marking a variable from t_1. $\omega(t_2)$ is called the normalization of the term t_1.*

Example 13. The expression 15 is the normalization of the expression 16 where $\omega = \{\mathrm{x}_1 \rightarrow \mathrm{y}_1,\ \mathrm{x}_2 \rightarrow \mathrm{y}_2\}$

$$p(g(f(\nu(y_1)), g(\nu(y_2), \nu(y_1)))) \tag{15}$$

$$p(g(\nu(f(\nu(x_1))), g(\nu(x_2), \nu(x_1)))) \tag{16}$$

If A_1, A_2, A_3, and A_4 are linear atomic terms and the variables of A_1 and A_2 are different from those of A_3 and A_4, the following logical transformations can be specified:

$$A_1 \wedge A_2 \text{ and } A_3 \Rightarrow A_4 \text{ leads to } A_1 \approx A_3 \text{ and } A_2 \wedge A_4 \tag{17}$$

$$A_1 \vee A_2 \text{ and } A_3 \Rightarrow A_4 \text{ leads to } A_1 \approx A_3 \text{ and } A_2 \vee A_4 \tag{18}$$

Normalization is applied to the logical terms $\mathrm{A}_2 \wedge \mathrm{A}_4$ and $\mathrm{A}_2 \vee \mathrm{A}_4$

The normalization of a term is carried out in three steps. The first step is called abstraction.

Definition 16. *Let x_1, x_2,..., x_n be variables of a term t_1. $\lambda c_1.\lambda c_2...\lambda c_n.t_2$ is called the abstraction form of t_1, if the expression 19 holds*

$$t_1 = (\lambda c_1.\lambda c_2...\lambda c_n.t_2)\ x_1 x_2 ... x_n \tag{19}$$

In the definition above, ordering in the list $\lambda\mathrm{c}_1.\lambda\mathrm{c}_2....\lambda\mathrm{c}_n$ is not important.

Example 14. The expression 20 is the abstraction form of the expression 21 where x_1 and x_2 are respectively represented by y_1 and y_2 in the abstraction form.

$$\lambda y_1.\lambda y_2.p(g(\nu(f(\nu(y_1))), g(\nu(y_2), b))) \wedge q(\nu(f(\nu(y_1)))) \tag{20}$$

$$p(g(\nu(f(\nu(x_1))), g(\nu(x_2), b))) \wedge q(\nu(f(\nu(x_1)))) \tag{21}$$

The second step is called elimination.

Definition 17. *Let t_1 be the abstraction form of a term such that each variable in t_1 is bound by an abstraction. t_2 is called the elimination form of t_1 if t_2 is the resulting term after all ν functions in t_1 are eliminated.*

Example 15. The expression 22 is the elimination form of the expression 23.

$$\lambda y_1.\lambda y_2.p(g(f(y_1), g(y_2, b))) \wedge q(f(y_1)) \tag{22}$$

$$\lambda y_1.\lambda y_2.p(g(\nu(f(\nu(y_1))), g(\nu(y_2), b))) \wedge q(\nu(f(\nu(y_1)))) \tag{23}$$

The third step is called marking.

Definition 18. *Let* $\lambda c_1.\lambda c_2...\lambda c_n.t_1$ *be the elimination form of a term such that each variable in* t_1 *is bound by an abstraction and* t_1 *does not contain* ν. t_2 *is called the marking form of* $\lambda c_1.\lambda c_1...\lambda c_n.t_1$ *if the expression 24 holds*

$$t_2 = (\lambda c_1.\lambda c_2...\lambda c_n.t_1)\ \nu(y_1)\nu(y_2)....\nu(y_n). \tag{24}$$

Example 16. The expression 25 is the marking form of the expression 26.

$$p(g(f(\nu(v_1)), g(\nu(v_2), b))) \wedge q(f(\nu(v_1))) \tag{25}$$

$$\lambda y_1.\lambda y_2.p(g(f(y_1), g(y_2, b))) \wedge q(f(y_1)) \tag{26}$$

Abstraction form is computed by $\rightarrow_A$, $\rightarrow_B$, $\rightarrow_C$ transformations. Elimination form is computed by $\rightarrow_D$ transformation. Marking form is computed by $\rightarrow_E$ transformation. (see Supplementary Material).

3 Verification

Definition 19. *The equivalence relation* $\equiv$ *is defined for the type* τ *as follows. 27, 28, 29, 30, 31 hold.*

$$a \equiv a \tag{27}$$

$$b \equiv b \tag{28}$$

$$a \equiv b \tag{29}$$

$$b \equiv a \tag{30}$$

$$x \equiv x \tag{31}$$

If $t_1 \equiv t_2$ *holds for* t_1, t_2 *which are of the type* τ, *32 holds.*

$$f(t_1) \equiv f(t_2) \tag{32}$$

If $t_1 \equiv t_3$ *and* $t_2 \equiv t_4$ *hold for* t_1, t_2, t_3, t_4 *which are the type* τ, *33 holds.*

$$g(t_1, t_2) \equiv g(t_3, t_4) \tag{33}$$

The relation can also be defined for the type o. *If* $t_1 \equiv t_2$ *holds for the terms* t_1, t_2 *which are of the type* τ, *then 34, 35, 36 hold.*

$$p(t_1) \equiv p(t_2) \tag{34}$$

$$q(t_1) \equiv q(t_2) \tag{35}$$

$$r(t_1) \equiv r(t_2) \tag{36}$$

If $t_1 \equiv t_3$ *and* $t_2 \equiv t_4$ *hold for the logical terms* t_1, t_2, t_3, t_4, *then 37 and 38 hold.*

$$(t_1 \wedge t_2) \equiv (t_3 \wedge t_4) \tag{37}$$

$$(t_1 \vee t_2) \equiv (t_3 \vee t_4) \tag{38}$$

Example 17. 39, 40, 41, 42, 43, 44 hold.

$$g(f(x), f(b)) \equiv g(f(x), f(a)) \tag{39}$$

$$g(f(x), y) \equiv g(f(x), y) \tag{40}$$

$$g(g(f(x), y), a) \equiv g(g(f(x), y), b) \tag{41}$$

$$p(f(a)) \equiv p(f(b)) \tag{42}$$

$$q(g(f(y), a)) \equiv q(g(f(y), b)) \tag{43}$$

$$(p(g(x, f(a))) \wedge q(g(y, f(x)))) \equiv (p(g(x, f(b))) \wedge q(g(y, f(x)))) \tag{44}$$

Definition 20. *Let x_1, x_2,..., x_n be variables of a term t not containing ν. Let y_1, y_2,..., y_n be new variables and assume that the equation 45 holds.*

$$\omega = \{x_1 \to \nu(y_1), x_2 \to \nu(y_2), ..., x_n \to \nu(y_n)\} \tag{45}$$

Then, $\omega(t)$ is called the well defined form of the term t.

Example 18. 47 is the well defined form of 48 where the equation 46 holds.

$$\omega = \{x_1 \to \nu(x_2), y_1 \to \nu(y_2)\} \tag{46}$$

$$(p(g(\nu(x_2), f(b))) \vee q(g(\nu(y_2), f(\nu(x_2))))) \tag{47}$$

$$(p(g(x_1, f(b))) \vee q(g(y_1, f(x_1)))) \tag{48}$$

Lemma 1. *Let t_1, t_2 be linear terms not containing ν, $\wedge$, $\vee$ and $\Rightarrow$. Assume that the variables of t_1 are different from the variables of t_2 and s_1, s_2 are the well defined forms of t_1, t_2 respectively. Similarly assume that the variables of s_1 are different from the variables of s_2. Then the statement 49 holds*

$$\phi(t_1) \equiv \phi(t_2) \; if \; and \; only \; if \; s_1 \approx s_2 \tag{49}$$

Lemma 2. *Let t_1, t_2 be linear terms not containing ν, $\wedge$, $\vee$, $\Rightarrow$ and x_1, x_2,..., x_n be variables of t_1, t_2. Assume that the variables of t_1 are different from the variables of t_2. Given ω in 50, let $s_1 = \omega(t_1)$ and $s_2 = \omega(t_2)$ hold.*

$$\omega = \{x_1 \to \nu(y_1), x_2 \to \nu(y_2), ..., x_n \to \nu(y_n)\} \tag{50}$$

If $s_1 \approx s_2$ holds and ω_1 is the resulting substitution and a substitution ω_2 is formed using 51 and 52 where $1 \leq j \leq n$, $1 \leq i \leq n$, u is a term other than a variable, then ω_2 is the most general substitution that satisfies 53

$$x_j \to x_i \in \omega_2 \; if \; and \; only \; if \; y_j \to y_i \in \omega_1 \tag{51}$$

$$x_j \to u \in \omega_2 \; if \; and \; only \; if \; y_j \to \omega(u) \in \omega_1 \tag{52}$$

$$\omega_2(t_1) \equiv \omega_2(t_2) \tag{53}$$

Lemma 3. *Given a term t_1 such that all the variables of t_1 are marked by the function ν and there is no bound variable in t_1, $\rightarrow_A\ t_1\ x$ holds once and terminates. The resulting substitution is $\{x \rightarrow t_2\}$ where t_2 is the abstraction form of t_1.*

Lemma 4. *Given a term t_1 such that each variable of t_1 is bound by an abstraction, $\rightarrow_D\ t_1\ x$ holds once and terminates. The resulting substitution is $\{x \rightarrow t_2\}$ where t_2 is the elimination form of t_1.*

Lemma 5. *Given a term t_1 such that each variable of t_1 is bound by an abstraction and t_1 does not contain ν,$\rightarrow_E\ t_1\ x$ holds once and terminates. The resulting substitution is $\{x \rightarrow t_2\}$ where t_2 is the marking form of t_1.*

Definition 21. *Given a term t_1 such that all the variables of t_1 are marked by the function ν and there is no bound variable in t_1, the three transformations $\rightarrow_A\ t_1\ x_1$ and $\rightarrow_D\ x_1\ x_2$ and $\rightarrow_E\ x_2\ x_3$ is briefly denoted by $\rightarrow_N\ t_1\ x_3$.*

Logical transformations are applied to the special logical forms.

Definition 22. *Let P_1, P_2, Q, R be linear and atomic logical terms not containing ν. Assume that the variables of P_1 and Q are different from those of P_2 and R. ω is a most general substitution. The expressions 54 and 55 are used to show the same transformation. The notation in 55 is preferred since it is more compact form.*

$$(P_1\ \wedge\ Q)\ and\ (P_2 \Rightarrow R)\ implies\ (\omega(P_1) \equiv \omega(P_2))\ and\ \omega(R) \wedge \omega(Q) \tag{54}$$

$$(P_1 \wedge Q) \models_{P_2 \Rightarrow R} (\omega(R) \wedge \omega(Q)) \tag{55}$$

The same formulation can be done by using well defined forms. Let ${P_1}^w$, ${P_2}^w$, Q^w, R^w be well defined forms of linear and atomic logical terms not containing ν. Assume that the variables of ${P_1}^w$ and Q^w are different from those of ${P_2}^w$ and R^w. The expression 57 is the compact form of 56.

$$(P_1^w \wedge Q^w)\ and\ (P_2^w \Rightarrow R^w)\ implies\ (P_1^w \approx P_2^w)\ and\ (\rightarrow_N\ (R^w \wedge Q^w)(R_1^w \wedge Q_1^w)) \tag{56}$$

$$(P_1^w \wedge Q^w) \models_{P_2^w \Rightarrow R^w} (R_1^w \wedge Q_1^w) \tag{57}$$

Similarly, 59 and 61 are the compact forms of 58 and 60 respectively.

$$(P_1 \vee Q)\ and\ (P_2 \Rightarrow R)\ implies\ (\omega(P_1) \equiv \omega(P_2))\ and\ \omega(R) \vee \omega(Q) \tag{58}$$

$$(P_1 \vee Q) \models_{P_2 \Rightarrow R} (\omega(R) \vee \omega(Q)) \tag{59}$$

$$(P_1^w \vee Q^w)\ and\ (P_2^w \Rightarrow R^w)\ implies\ (P_1^w \approx P_2^w)\ and\ (\rightarrow_N\ (R^w \vee Q^w)(R_1^w \vee Q_1^w)) \tag{60}$$

$$(P_1^w \vee Q^w) \models_{P_2^w \Rightarrow R^w} (R_1^w \vee Q_1^w) \tag{61}$$

The logical transformations have the following property.

Theorem 1. *Let P_1 be a logical term not containing ν. Also assume that there is a logical term P_2 such that $P_1 \equiv P_2$ holds and Q_1 is the well defined form of P_2. Then, the statement 62 holds where R^w is the well defined form of R.*

$$P_1 \models_R P_3 \; if \; and \; only \; if \; Q_1 \models_{R^w} Q_2 \tag{62}$$

Additionally, if $P_1 \models_R P_3$ and $Q_1 \models_{R^w} Q_2$ succeed, then there is a logical term P_4 such that $P_3 \equiv P_4$ holds and Q_2 is the well defined form of P_4.

Lemma 6. *The logical transformations $\models_R$ produce linear atomic logical terms.*

The expression 63 leads to the expression 64.

$$p(f(\nu(x))) \wedge q(f(\nu(x))) \; and \; p(\nu(z)) \Rightarrow r(\nu(z)) \tag{63}$$

$$p(f(\nu(x))) \approx p(\nu(z)) \; and \; \rightarrow_N (q(f(\nu(x))) \wedge r(\nu(z))) \, U \tag{64}$$

In the following examples, the proofs of the statements in 64 are given

Example 19.

$$p(f(\nu(x))) \approx p(\nu(z)) \tag{65}$$

$$f(\nu(x)) \approx \nu(z) \tag{66}$$

$$\mu f(\nu(x)) \; and \; \{z \rightarrow f(\nu(x))\} \tag{67}$$

$$true \; and \; \{z \rightarrow f(\nu(x))\} \tag{68}$$

Example 20.

$$\rightarrow_B (q(f(\nu(x))) \wedge r(\nu(f(\nu(x))))) \; c \; (U_1 c) \tag{69}$$

$$\rightarrow_B q(f(\nu(x))) \; c \, U_2 \; and \; \rightarrow_B r(\nu(f(\nu(x)))) \; c \, U_3 \tag{70}$$

$$\{U_1 \rightarrow \lambda c.(U_2 \wedge U_3)\} \tag{71}$$

$$\rightarrow_B f(\nu(x)) \; c \, U_4 \; and \; \rightarrow_B r(\nu(f(\nu(x)))) \; c \, U_3 \tag{72}$$

$$\{U_1 \rightarrow \lambda c.(U_2 \wedge U_3), U_2 \rightarrow q(U_4)\} \tag{73}$$

$$\rightarrow_B \nu(x) \; c \, U_5 \; and \; \rightarrow_B r(\nu(f(\nu(x)))) \; c \, U_3 \tag{74}$$

$$\{U_1 \rightarrow \lambda c.(U_2 \wedge U_3), U_2 \rightarrow q(U_4), U_4 \rightarrow f(U_5)\} \tag{75}$$

$$\rightarrow_B \nu(x) \; c \, \nu(U_6) \; and \; \rightarrow_B r(\nu(f(\nu(x)))) \; c \, U_3 \tag{76}$$

$$\{U_1 \rightarrow \lambda c.(U_2 \wedge U_3), U_2 \rightarrow q(U_4), U_4 \rightarrow f(U_5), U_5 \rightarrow \nu(U_6)\} \tag{77}$$

$$\rightarrow_B r(\nu(f(\nu(x)))) \; c \, U_3 \tag{78}$$

$$\{U_1 \rightarrow \lambda c.(U_2 \wedge U_3), U_2 \rightarrow q(U_4), U_4 \rightarrow f(U_5), U_5 \rightarrow \nu(U_6), U_6 \rightarrow (xc)\} \tag{79}$$

$$\rightarrow_B r(\nu(f(\nu(x)))) \; c \, U_3 \tag{80}$$

$$\{U_1 \rightarrow \lambda c.(q(f(\nu(x \; c))) \wedge U_3)\} \tag{81}$$

$$\rightarrow_B \nu(f(\nu(x))) \; c \, U_7 \tag{82}$$

$$\{U_1 \rightarrow \lambda c.(q(f(\nu(x \; c))) \wedge U_3), U_3 \rightarrow r(U_7)\} \tag{83}$$

$$\rightarrow_B \nu(f(\nu(x)))\, c\, \nu(U_8) \tag{84}$$

$$\{U_1 \rightarrow \lambda c.(q(f(\nu(x\, c))) \wedge U_3), U_3 \rightarrow r(U_7), U_7 \rightarrow \nu(U_8)\} \tag{85}$$

$$\rightarrow_B f(\nu(x))\, c\, U_8 \tag{86}$$

$$\{U_1 \rightarrow \lambda c.(q(f(\nu(x\, c))) \wedge U_3), U_3 \rightarrow r(U_7), U_7 \rightarrow \nu(U_8)\} \tag{87}$$

$$\rightarrow_B \nu(x)\, c\, U_9 \tag{88}$$

$$\{U_1 \rightarrow \lambda c.(q(f(\nu(x\, c))) \wedge U_3), U_3 \rightarrow r(U_7), U_7 \rightarrow \nu(U_8), U_8 \rightarrow f(U_9)\} \tag{89}$$

$$\rightarrow_B \nu(x)\, c\, \nu(U_{10}) \tag{90}$$

$$\{U_1 \rightarrow \lambda c.(q(f(\nu(xc))) \wedge U_3), U_3 \rightarrow r(U_7), U_7 \rightarrow \nu(U_8), U_8 \rightarrow f(U_9), U_9 \rightarrow \nu(U_{10})\} \tag{91}$$

$$true \tag{92}$$

$$\{U_1 \rightarrow \lambda c.(q(f(\nu(x\, c))) \wedge r(\nu(f(\nu(x\, c)))))\} \tag{93}$$

Example 21.

$$\rightarrow_C (q(f(\nu(x\, c))) \wedge r(\nu(f(\nu(x\, c)))))\, c \tag{94}$$

$$\rightarrow_C q(f(\nu(x\, c)))\, c \tag{95}$$

$$\rightarrow_C f(\nu(x\, c))\, c \tag{96}$$

$$\rightarrow_C \nu(x\, c)\, c \tag{97}$$

$$true \tag{98}$$

$$\{x \rightarrow \lambda c.c\} \tag{99}$$

Example 22.

$$\rightarrow_A (q(f(\nu(x))) \wedge r(\nu(f(\nu(x)))))\, Y \tag{100}$$

$$\rightarrow_B (q(f(\nu(x))) \wedge r(\nu(f(\nu(x)))))\, c\, (Y_1 c)\ and\ \rightarrow_C (Y_1 c)\, c\ and \tag{101}$$

$$\rightarrow_A (Y_1 c)(Y c) \tag{102}$$

$$true\ and\ \rightarrow_C (q(f(\nu(x\, c))) \wedge r(\nu(f(\nu(x\, c)))))\, c\ and \tag{103}$$

$$\rightarrow_A (q(f(\nu(x\, c))) \wedge r(\nu(f(\nu(x\, c)))))\, (Y c) \tag{104}$$

$$\{Y_1 \rightarrow \lambda c.(q(f(\nu(x\, c))) \wedge r(\nu(f(\nu(x\, c)))))\} \tag{105}$$

$$true\ and\ true\ and\ \rightarrow_A (q(f(\nu(c))) \wedge r(\nu(f(\nu(c)))))\, (Y c) \tag{106}$$

$$\{x \rightarrow \lambda c.c\} \tag{107}$$

$$true\ and\ true\ and\ \rightarrow_A (q(f(\nu(c))) \wedge r(\nu(f(\nu(c)))))\, Y_2 \tag{108}$$

$$\{Y \rightarrow \lambda c.Y_2\} \tag{109}$$

$$true\ and\ true\ and\ true \tag{110}$$

$$\{Y \rightarrow \lambda c.Y_2, Y_2 \rightarrow (q(f(\nu(c))) \wedge r(\nu(f(\nu(c)))))\} \tag{111}$$

$$true\ and\ true\ and\ true \tag{112}$$

$$\{Y \rightarrow \lambda c.(q(f(\nu(c))) \wedge r(\nu(f(\nu(c)))))\} \tag{113}$$

Example 23.

$$\to_D \lambda c.(q(f(\nu(c))) \wedge r(\nu(f(\nu(c))))) \, U \tag{114}$$

$$\to_D (q(f(\nu(c))) \wedge r(\nu(f(\nu(c))))) \, (U \; c) \tag{115}$$

$$\to_D q(f(\nu(c))) \, U_1 \; and \; \to_D r(\nu(f(\nu(c)))) \, U_2 \tag{116}$$

$$\{U \to \lambda c.(U_1 \wedge U_2)\} \tag{117}$$

$$\to_D f(\nu(c)) \, U_3 \; and \; \to_D r(\nu(f(\nu(c)))) \, U_2 \tag{118}$$

$$\{U \to \lambda c.(U_1 \wedge U_2), U_1 \to q(U_3)\} \tag{119}$$

$$\to_D \nu(c) \, U_4 \; and \; \to_D r(\nu(f(\nu(c)))) \, U_2 \tag{120}$$

$$\{U \to \lambda c.(U_1 \wedge U_2), U_1 \to q(U_3), U_3 \to f(U_4)\} \tag{121}$$

$$true \, and \; \to_D r(\nu(f(\nu(c)))) \, U_2 \tag{122}$$

$$\{U \to \lambda c.(U_1 \wedge U_2), U_1 \to q(U_3), U_3 \to f(U_4), U_4 \to c\} \tag{123}$$

$$true \, and \; \to_D r(\nu(f(\nu(c)))) \, U_2 \tag{124}$$

$$\{U \to \lambda c.(q(f(c)) \wedge U_2)\} \tag{125}$$

$$true \, and \; \to_D \nu(f(\nu(c))) \, U_5 \tag{126}$$

$$\{U \to \lambda c.(q(f(c)) \wedge U_2), U_2 \to r(U_5)\} \tag{127}$$

$$true \, and \; \to_D f(\nu(c)) \, U_5 \tag{128}$$

$$\{U \to \lambda c.(q(f(c)) \wedge U_2), U_2 \to r(U_5)\} \tag{129}$$

$$true \, and \; \to_D \nu(c) \, U_6 \tag{130}$$

$$\{U \to \lambda c.(q(f(c)) \wedge U_2), U_2 \to r(U_5), U_5 \to f(U_6)\} \tag{131}$$

$$true \, and \, true \tag{132}$$

$$\{U \to \lambda c.(q(f(c)) \wedge U_2), U_2 \to r(U_5), U_5 \to f(U_6), U_6 \to c\} \tag{133}$$

$$true \, and \, true \tag{134}$$

$$\{U \to \lambda c.(q(f(c)) \wedge r(f(c)))\} \tag{135}$$

Example 24.

$$\to_E \lambda c.(q(f(c)) \wedge r(f(c))) \, U \tag{136}$$

$$\to_E (q(f(\nu(y))) \wedge r(f(\nu(y)))) \, U \tag{137}$$

$$\to_E (q(f(\nu(y))) \wedge r(f(\nu(y)))) \, (U_1 \wedge U_2) \tag{138}$$

$$\{U \to (U_1 \wedge U_2)\} \tag{139}$$

$$true \tag{140}$$

$$\{U \to (U_1 \wedge U_2), U_1 \to q(f(\nu(y))), U_2 \to r(f(\nu(y)))\} \tag{141}$$

$$true \tag{142}$$

$$\{U \to (q(f(\nu(y))) \wedge r(f(\nu(y))))\} \tag{143}$$

4 Conclusion

The specification of the relation $\approx$ is trivial for terms not containing variables. When terms contain variables, its trivial formulation leads to non-determinism. This paper addresses this issue. It redefines the relatiın by using the function ν, which is used to mark the places of variables, to solve this problem. So terms that are used are defined in a special format, which is called well defined forms. But this specific form also brings the two restrictions. First, logical atomic terms should be linear. Second, after substitution, well defined form is not preserved. As a result, this necessitates an extra transformation step, which is called normalization. After normalization, terms are guaranteed to be in a good shape, they are again in well defined form.

Further research in this approach can be done to eliminate the first restriction. One way, variables are marked not in the original term but in its copy form. To mark the variables of an original term using its copy eliminates the linearity restriction. The other way is to change the definition of linearity to include more terms.

5 Supplementary Material

1. a $\approx$ b
2. b $\approx$ a
3. a $\approx$ a
4. b $\approx$ b
5. if ν(x) $\approx$ ν(y) {y $\to$ x}
6. if ν(x) $\approx$ y (μ y) and {x $\to$ y}
7. if y $\approx$ ν(x) (μ y) and {x $\to$ y}
8. if f(x) $\approx$ f(y) x $\approx$ y
9. if g(x_1,x_2) $\approx$ g(x_3,x_4) $x_1 \approx x_3$ and $x_2 \approx x_4$
10. if p(x) $\approx$ p(y) x $\approx$ y
11. if q(x) $\approx$ q(y) x $\approx$ y
12. if r(x) $\approx$ r(y) x $\approx$ y

1. if $\to_A$ x y
   if λc_1.($\to_B$ x c_1 (y_1 c_1) and $\to_C$ (y_1 c_1) c_1 and $\to_A$ (y_1 c_1) (y c_1))
   else {y $\to$ x}

1. $\to_B$ a v a
2. $\to_B$ b v b
3. if $\to_B$ ν(x_1) v ν(x_2) if {x_2 $\to$ (x_1 v)} else $\to_B$ x_1 v x_2 else {x_2 $\to$ x_1}
4. if $\to_B$ f(x) v f(y) $\to_B$ x v y
5. if$\to_B$ g(x_1,x_2) v g(x_3,x_4) $\to_B$ x_1 v x_3 and $\to_B$ x_2 v x_4
6. if $\to_B$ p(x) v p(y) $\to_B$ x v y
7. if $\to_B$ q(x) v q(y) $\to_B$ x v y
8. if $\to_B$ r(x) v r(y) $\to_B$ x v y
9. if $\to_B$ ($x_1 \wedge x_2$) v ($x_3 \wedge x_4$) $\to_B$ x_1 v x_3 and $\to_B$ x_2 v x_4

10. if $\rightarrow_B$ ($x_1 \vee x_2$) v ($x_3 \vee x_4$) $\rightarrow_B$ x_1 v x_3 and $\rightarrow_B$ x_2 v x_4

1. if $\rightarrow_C$ $\nu(x_1)$ v (if $\omega(x_1)$ = v else $\rightarrow_C$ x_1 v)[1]
2. if $\rightarrow_C$ f(x) v $\rightarrow_C$ x v
3. if $\rightarrow_C$ g(x_1,x_2) v (if $\rightarrow_C$ x_1 v else $\rightarrow_C$ x_2 v)
4. if $\rightarrow_C$ p(x) v $\rightarrow_C$ x v
5. if $\rightarrow_C$ q(x) v $\rightarrow_C$ x v
6. if $\rightarrow_C$ r(x) v $\rightarrow_C$ x v
7. if $\rightarrow_C$ ($x_1 \wedge x_2$) v (if $\rightarrow_C$ x_1 v else $\rightarrow_C$ x_2 v)
8. if $\rightarrow_C$ ($x_1 \vee x_2$) v (if $\rightarrow_C$ x_1 v else $\rightarrow_C$ x_2 v)

1. $\rightarrow_D$ a a
2. $\rightarrow_D$ b b
3. if $\rightarrow_D$ ν(x) y if $\rightarrow_D$ x y else {y $\rightarrow$ x}
4. if $\rightarrow_D$ f(x) f(y) $\rightarrow_D$ x y
5. if $\rightarrow_D$ g(x_1,x_2) g(x_3,x_4) $\rightarrow_D$ x_1 x_3 and $\rightarrow_D$ x_2 x_4
6. if $\rightarrow_D$ p(x) p(y) $\rightarrow_D$ x y
7. if $\rightarrow_D$ q(x) q(y) $\rightarrow_D$ x y
8. if $\rightarrow_D$ r(x) r(y) $\rightarrow_D$ x y
9. if $\rightarrow_D$ ($x_1 \wedge x_2$) ($x_3 \wedge x_4$) $\rightarrow_D$ x_1 x_3 and $\rightarrow_D$ x_2 x_4
10. if $\rightarrow_D$ ($x_1 \vee x_2$) ($x_3 \vee x_4$) $\rightarrow_D$ x_1 x_3 and $\rightarrow_D$ x_2 x_4
11. if $\rightarrow_D$ x_1 x_2 λc_1.($\rightarrow_D$ (x_1 c_1) (x_2 c_1))

1. if $\rightarrow_E$ p(x_1) p(x_2) {$x_2 \rightarrow x_1$}
2. if $\rightarrow_E$ q(x_1) q(x_2) {$x_2 \rightarrow x_1$}
3. if $\rightarrow_E$ r(x_1) r(x_2) {$x_2 \rightarrow x_1$}
4. if $\rightarrow_E$ ($x_1 \wedge x_2$) ($x_3 \wedge x_4$) {$x_3 \rightarrow x_1$, $x_4 \rightarrow x_2$}
5. if $\rightarrow_E$ ($x_1 \vee x_2$) ($x_3 \vee x_4$) {$x_3 \rightarrow x_1$, $x_4 \rightarrow x_2$}
6. if $\rightarrow_E$ x_1 x_2 $\rightarrow_E$ (x_1 $\nu(y_1)$) x_2

References

1. Chang, C., Char, R.-Lee, T.: Symbolic Logic and Mechanical Theorem Proving. Academic Press, New York, San Francisco, London (1973)
2. Robinson, A., Voronkov, A.: Handbook of Automated Reasoning. 1st edn. The Mit Press, Cambridge, Massachusetts (2001)
3. Clocksin, W., Mellish, C.: Programming in Prolog. Springer-Verlag, Berlin, New York (2003)
4. Stanford Encyclopedia of Philosophy, https://plato.stanford.edu/entries/lambda-calculus/
5. LFM'99 Proceedings, https://www.site.uottawa.ca/ afelty/LFM99/index.html/

[1] $\omega = \{x_2 \rightarrow \lambda c_1.\lambda c_2 ... \lambda c_n.v\}$ and x_2 is a variable of a term replacing the variable x_1.