

OIJ-2506 Program Verification

Notations

Antti Valmari & Antero Kangas
Department of Software Systems

August 30, 2010

Description of Notations

In this section there are collected the mathematical and pseudocode notations used in the course, except the most common ones.

Mathematics general

$\lfloor x \rfloor$ The greatest integer that is at most equal to x . E.g. $\lfloor 4 \rfloor = \lfloor 4, 2 \rfloor = \lfloor 4, 9 \rfloor = 4$, and $\lfloor -4 \rfloor = -4$, and $\lfloor -4, 2 \rfloor = \lfloor -4, 9 \rfloor = -5$. Useful formulas: $\lfloor x \rfloor \leq x$ and $0 \leq x - \lfloor x \rfloor < 1$.

$\lceil x \rceil$ The smallest integer that is at least equal to x . E.g. $\lceil 4 \rceil = 4$ ja $\lceil 4, 2 \rceil = \lceil 4, 9 \rceil = 5$, and $\lceil -4 \rceil = \lceil -4, 2 \rceil = \lceil -4, 9 \rceil = -4$. Useful formulas: $\lceil x \rceil \geq x$, and $0 \leq \lceil x \rceil - x < 1$ ja $\lceil -x \rceil = -\lfloor x \rfloor$.

$\star_{i \in I}$ If \star is some binary operator that meets the suitable conditions, then $\star_{i \in I} f(i)$ means that all elements of set I are went through, $f(i)$ is evaluated for every element, and the results are composed by operator \star . E.g. $\star_{i \in \{2,3,4,5\}} i^2 = 2^2 + 3^2 + 4^2 + 5^2 = \sum_{i=2}^5 i^2$. It is used especially with the set theoretical operators \cup and \cap .

\rightarrow^* $u \rightarrow^* v$ means that it is possible to go from u to v by zero or more steps. E.g. for graphs the step is the moving from the starting vertice of an edge to its ending vertice.

\mathbb{N} See natural numbers.

$\mathbb{Z}, \mathbb{Z}^+, \mathbb{Z}^-$ See integers.

$\mathbb{Q}, \mathbb{Q}^+, \mathbb{Q}^-$ See rational numbers.

$\mathbb{R}, \mathbb{R}^+, \mathbb{R}^-$ See real numbers.

Direction of associativity See direction of binding

Direction of binding It is also called as direction of associativity (liitännäisyyden suunta) The direction of binding of operator \star defines how $x \star y \star z$ is interpreted. If \star binds to left, then $x \star y \star z = (x \star y) \star z$. If \star binds to right then $x \star y \star z = x \star (y \star z)$.

In mathematics comparative operators do not bind in either direction: the formula $x \leq y = z$ does not mean $(x \leq y) = z$ nor $x \leq (y = z)$ but $(x \leq y) \wedge (y = z)$. The interpretation $(x \leq y) = z$ would not usually be reasonable in mathematics, since $(x \leq y)$ produces a truth value, and comparing a truth value to a number etc. is not usually defined. In a programming language the situation is usually different: there exist comparative operators that bind to left and/or to right.

See also level of binding. See directions of binding for logical etc. operators under the heading “direction of binding”.

Integers The numbers $\dots, -3, -2, -1, 0, 1, 2, 3, \dots$. The set of integers is denoted as \mathbb{Z} . The set of positive integers is $\mathbb{Z}^+ = \{n \in \mathbb{Z} | n > 0\}$ ja The set of negative integers is $\mathbb{Z}^- = \{n \in \mathbb{Z} | n < 0\}$.

Level of binding It is also called precedence. If operator \star binds stronger, that is, is on higher level of binding than operator \circ , then $x \star y \circ z$ is interpreted as $(x \star y) \circ z$, and $x \circ y \star z$ is interpreted as $x \circ (y \star z)$.

Human beings have – when analysing formulas – a tendency to take in account hints given by content and typesetting. E.g. $\forall x : P(x) \vee \forall x : Q(x)$ is easily interpreted as $(\forall x : P(x)) \vee (\forall x : Q(x))$, even, by the binding rules of logic in this text, it means $\forall x : (P(x) \vee \forall x : Q(x))$. For this reason it is useful to use extra parenthesis, and when reading formulas it pays off to take into account the possibility that the author may accidentally have left out obligate parenthesis. Extra parenthesis are useful also since binding rules used in literature are varying.

See also direction of binding. See levels of binding for logical etc. operators under “level of binding”

Natural numbers The numbers $0, 1, 2, 3, \dots$. The set of natural numbers is denoted as \mathbb{N} . Warning: some authors mean by natural numbers the numbers $1, 2, 3, \dots$

Precedence See the level of binding

Rational numbers The set of rational numbers is $\mathbb{Q} = \{\frac{m}{n} | m \in \mathbb{Z} \wedge n \in \mathbb{Z} - \{0\}\}$. The set of positive rational numbers is $\mathbb{Q}^+ = \{x \in \mathbb{Q} | x > 0\}$, and the set of negative rational numbers is $\mathbb{Q}^- = \{x \in \mathbb{Q} | x < 0\}$.

Real numbers The set of real numbers is denoted as \mathbb{R} . The set of positive real numbers is $\mathbb{R}^+ = \{x \in \mathbb{R} | x > 0\}$, and the set of negative real numbers is $\mathbb{R}^- = \{x \in \mathbb{R} | x < 0\}$.

Used set of numbers In this text, unless the used set of numbers is especially notified, the set of integers is used.

Logic

\wedge “And”. $\varphi \wedge \xi$ means that both φ and ξ hold.

\vee “Or”. $\varphi \vee \xi$ means that φ holds, or ξ holds, or both hold.

\neg “Not” (negation). $\neg\varphi$ means that φ does not hold.

\rightarrow Implication as a logical operator. $\varphi \rightarrow \xi$ means that if φ holds then also ξ holds. In other words, $\varphi \rightarrow \xi$ produces False if and only if φ produces True and ξ produces False. C.f. \Rightarrow .

- \leftrightarrow Equivalence as a logical operator. $\varphi \leftrightarrow \xi$ is same as $\varphi \rightarrow \xi$ and $\xi \rightarrow \varphi$ together. In other words, $\varphi \leftrightarrow \xi$ produces True if and only if either both φ and ξ produce True, or both φ and ξ produce False. C.f. \leftrightarrow .
- \Rightarrow **1.** Implication as a comparison of two logical claims. It is usually used to express a step of deduction. $\varphi \Rightarrow \xi$ holds if and only if in every possible situation in the context in question and in which φ holds, also ξ holds. (Different situations can be made by varying the values of free variables inside φ and ξ .) In other words, $\varphi \Rightarrow \xi$ holds if and only if $\varphi \rightarrow \xi$ always produces True regardless the values of free variables inside φ and ξ , unless a combination of values impossible in the context is not chosen. E.g. $x > 1 \Rightarrow x + y > 1$ is generally not a valid deduction but it is valid, if it is known from context that $y \geq 0$.
- \Rightarrow is similar to \rightarrow , but \Rightarrow exists between formulas and \rightarrow exists inside a formula. Therefore it is allowed to write $(\varphi \rightarrow \xi) \rightarrow \zeta$, but it is not allowed to write $(\varphi \Rightarrow \xi) \Rightarrow \zeta$, because there \Rightarrow come into a formula. It is allowed to write $\varphi \Rightarrow \xi \Rightarrow \zeta$. It means the same as $\varphi \Rightarrow \xi$ and $\xi \Rightarrow \zeta$ together, but $\varphi \rightarrow \xi \rightarrow \zeta$ means $(\varphi \rightarrow \xi) \rightarrow \zeta$. (Warning: some authors mean by $\varphi \rightarrow \xi \rightarrow \zeta$ actually $\varphi \rightarrow (\xi \rightarrow \zeta)$.)
- 2.** it is often used also in the same meaning as \rightarrow .
- \Leftrightarrow Equivalence as comparison of two claims. $\varphi \Leftrightarrow \xi$ holds if and only if φ holds exactly in the same situations, allowed by the context, than ξ . C.f. \Rightarrow and \leftrightarrow .
- \forall Universal quantifier (kaikki-kvanttori in Finnish). $\forall x : \varphi(x)$ means that for every element x the condition $\varphi(x)$ holds.
- $\forall x; \varphi(x) : \xi(x)$ means that for every element x for which the condition $\varphi(x)$ holds, also the condition ξ holds. It means the same as $\forall x : \varphi(x) \rightarrow \xi(x)$. (Warning: this style of using punctuation marks with quantifier “ \forall ” is not very common, even it helps grouping formulas to become easier to understand.)
- $\forall x \in A : \varphi(x)$ means that for all element x of set A the condition φ holds. It means the same as $\forall x : x \in A \rightarrow \varphi(x)$.
- \exists Existence quantifier (on olemassa -kvanttori in Finnish). $\exists x : \varphi(x)$ means that for at least one element x the condition $\varphi(x)$ holds.
- $\exists x; \varphi(x) : \xi(x)$ means that there exists at least one element x for which the condition φ holds and for which also the condition ξ holds. It means the same as $\exists x : \varphi(x) \wedge \xi(x)$. (Warning: this style of using punctuation marks with quantifier “ \exists ” is not very common, even it helps grouping formulas to become easier to understand.)
- $\exists x \in A : \varphi(x)$ means that for at least one element x of set A the condition φ holds. It means the same as $\exists x : x \in A \wedge \varphi(x)$.

Bound variable A variable x is bound when it is quantified. If a variable is not bound, it is free. E.g. in formula $y > 0 \wedge (\forall y : x + y < 0) \vee y = 0$ the centermost occurrences of y are bound and the outermost are free. The situation can be interpreted also so that a quantifier introduces a new variable that has same name than some other variable.

Direction of bounding In this text $\wedge, \forall, \rightarrow$ ja \leftrightarrow bound to left, and \Rightarrow ja \Leftrightarrow behave like comparative operators (see \Rightarrow). Warning: the direction of bounding of \rightarrow is varying in literature.. See also Mathematics general \triangleright direction of bounding.

False The logical truth value “untrue”. If a formula does not hold then its truth value is False. Often (especially in programming languages) the numerical value 0 is used instead of False.

Free variable See bound variable.

Level of bounding Levels of bounding for logical operators are varying in literature. In this text they are from the strongest to the weakest: \neg , \wedge , \vee , \rightarrow , \leftrightarrow , quantifiers. The operators used for comparing formulas \Rightarrow and \Leftrightarrow bind weaker than the previously listed and behave like comparative operators. The operators that produce truth values from non-logical values, like $=$, \neq , $<$, and \subseteq bind more strongly than logical operators. See also Mathematics general \triangleright level of bounding.

Quantifier See \forall and \exists . Sometimes also other similarly used symbols are called as quantifiers.

True The logical truth value “true”. If a formula holds, its truth value is True. Often (especially in programming languages) the numerical value 1 is used instead of True.

Truth value The value produced by a logical formula. Truth values are False ja True.

Undefined operations in formulas A predicate is well-defined if and only if the predicates of it that include undefined operations become eliminated by using the following calculation rules: $\text{False} \wedge (\text{what ever}) \Leftrightarrow (\text{what ever}) \wedge \text{False} \Leftrightarrow \text{False}$, $\text{True} \vee (\text{what ever}) \Leftrightarrow (\text{what ever}) \vee \text{True} \Leftrightarrow \text{True}$, $\text{False} \rightarrow (\text{what ever}) \Leftrightarrow \text{True}$, and $(\text{what ever}) \rightarrow \text{True} \Leftrightarrow \text{True}$. Therefore e.g. for real numbers $\forall x : \frac{1}{x} \neq 0 \vee \frac{1}{x} = 0$ is not well-defined, but $\forall x : \frac{1}{x} \neq 0 \vee x = 0$ is well-defined and produces True. Predicates must be written so that they are well-defined.

Set theory

\emptyset Empty set, that is, the set that does not have any elements. It holds that $A = \emptyset \Leftrightarrow \forall x : x \notin A$.

$\{\dots\}$ $\{a_1, a_2, \dots, a_n\}$ is the set, which elements are a_1, a_2, \dots, a_n . It holds that $x \in \{a_1, a_2, \dots, a_n\} \Leftrightarrow x = a_1 \vee x = a_2 \vee \dots \vee x = a_n$. E.g. $\text{DaysOfWeek} = \{\text{Monday}, \text{Tuesday}, \dots, \text{Sunday}\}$.

$\{\}$ is a consistent but seldom used alternative notation for set \emptyset .

$\{x \mid \varphi(x)\}$ is the set of the elements x , for which the condition φ hold. E.g. $\{x \mid \exists y \in \mathbb{Z} : x = 2 \cdot y\}$ is the set of even integers. It holds that $x \in \{x \mid \varphi(x)\} \Leftrightarrow \varphi(x)$.

$\{x \in A \mid \varphi(x)\}$ is the set the elements x of set A for which the condition φ hold. It holds that $\{x \in A \mid \varphi(x)\} = \{x \mid x \in A \wedge \varphi(x)\} = \{x \mid \varphi(x)\} \cap A$.

$\{f(x) \mid \varphi(x)\}$ is the set produced by going through all the elements x for which the condition φ hold; computing $f(x)$ for each of them; and by combining the results to set. E.g. $\{2 \cdot x \mid x \in \mathbb{Z}\}$ is the set of even integers. Warning: since this notation does not separately tell the variable which values are went through, it is unclear when the formula has several variable symbols.

$\langle \dots \rangle$ See \times .

\in Being a member of set. E.g. $\text{Tampere} \in \text{Finnish_cities}$.

\notin Being not a member of set. E.g. $\text{Melbourne} \notin \text{Finnish_cities}$. Can be defined by the formula $x \notin A \Leftrightarrow \neg(x \in A)$.

ε Empty sequence, the only element of set A^0 . See \times .

- = Equality of sets is defined as $A = B \Leftrightarrow \forall x : x \in A \leftrightarrow x \in B$.
- \subseteq Subset. E.g. $Finnish_cities \subseteq Cities$. It holds that $A \subseteq B \Leftrightarrow \forall x : x \in A \rightarrow x \in B$, and $A \subseteq B \Leftrightarrow A \subset B \vee A = B$.
- \subset Proper subset. It holds that $A \subset B \Leftrightarrow A \subseteq B \wedge A \neq B$. Warning: some authors use \subset to mean same as \subseteq means in this text.
- \cup Union. E.g. $\{1, 2, 3\} \cup \{2, 4\} = \{1, 2, 3, 4\}$. It holds that $A \cup B = \{x | x \in A \vee x \in B\}$.
- \cap Intersection. E.g. $\{1, 2, 3\} \cap \{2, 4\} = \{2\}$. It holds that $A \cap B = \{x | x \in A \wedge x \in B\}$.
- Set difference. E.g. $\{1, 2, 3\} - \{2, 4\} = \{1, 3\}$. It holds that $A - B = \{x | x \in A \wedge x \notin B\}$. In literature also the symbol \setminus is used.
- \times Cartesian product. E.g. $\{1, 2, 3\} \times \{2, 4\} = \{(1, 2), (1, 4), (2, 2), (2, 4), (3, 2), (3, 4)\}$. It holds that $A \times B = \{(x, y) | x \in A \wedge y \in B\}$.
- Also connecting more than two sets by operator \times is possible: $A_1 \times A_2 \times \dots \times A_n = \{(a_1, a_2, \dots, a_n) | \forall i; 1 \leq i \leq n : a_i \in A_i\}$.
- Instead of the ordinary parenthesis sometimes \langle and \rangle are used for the reason that the elements of a Cartesian product would be better distinguished and would not be confused to the other parenthesis of a formula. In the theory of languages the parenthesis and commas between elements are often totally left out.
- When $n = 1$ it is often considered that $A_1 \times \dots \times A_n = A_1$. Technically it is $\{(a) | a \in A_1\}$, but (a) and a are often treated as equals.
- When $n = 0$ then the Cartesian product has exactly one element. It would be consistent to denote, and sometimes it is denoted, $()$ or $\langle \rangle$. This notation is not useful when parenthesis are totally left out. Then the symbol ϵ is used.
- A^n A raised to power n . $A^n = A \times A \times \dots \times A$, where A appears in the right side n times. In other words it is all the n -element tuples that can be produced from the elements of A . Especially A^0 is, depending on the way of notation, $\{()\}$, $\{\langle \rangle\}$, or $\{\epsilon\}$.
- A^* The set of finite sequences that can be produced from set A . E.g. $\{+, -\}^* = \{\epsilon, +, -, ++, +-, -+, --, +++ , ++-, \dots\}$. It holds that $A^* = A^+ \cup \{\epsilon\} = A^0 \cup A^1 \cup A^2 \cup \dots$.
- A^+ The set of finite non-empty sequences that can be produced from set A . It holds that $A^+ = A^* - \{\epsilon\} = A^1 \cup A^2 \cup A^3 \cup \dots$.
- $|A|$ The amount of elements of finite set A . E.g. $|\{+, -\}| = 2$, $|\emptyset| = 0$, and $|\{1, 2, 3, 2, 1\}| = 3$.
- 2^A The set of all subsets of A , that is, the power set of A . E.g. $2^{\{+, -\}} = \{\emptyset, \{+\}, \{-\}, \{+, -\}\}$. It holds that $2^A = \{X | X \subseteq A\}$.
- $\mathcal{P}(A)$ Same as 2^A .

Direction of binding In this text \cup , \cap , $-$, and \times bind to left. \in , \notin , \subseteq , and \subset are comparative operators and behave like that. See also Mathematics general \triangleright direction of binding.

Level of binding In this text \times binds the most strongly, then \cap , then \cup and $-$ equally strongly, and finally \in , \notin , \subseteq and \subset . In literature the levels of binding can vary. See also Mathematics general \triangleright level of binding.

Pseudocode (program code)

The pseudocode used for presenting programs is adapted from the notations of Algol–Pascal–Ada languages. In conditions and expressions mathematical notations are preferred. Handy extras have been picked from C/C++, like +=.

Use of Algol-style pseudocode is common in literature. Many C/C++ notations are poor for teaching purposes since they are complicated, and the assignment- and comparison of equality operators conflict with the comparison operator of equality in mathematics: in mathematics equality is = but in C/C++ = means assignment and the comparison operator of equality is ==. Also, the difference of pseudocode helps to distinguish general things from C/C++ dependents.

If not otherwise mentioned or it is not clear from the context, all variables are integer type.

- Selector of a field of record (or object). E.g. *person.name*. Works as in C++:ssa.
- ↑ Referring to a record by pointer. *ref*↑ is same as **ref* in C++, and *ref*↑.*field* works as *ref*->*field* in C++.
- := (Ordinary) assignment operator. *v* := *e* works as *v* = *e* in C++.
- += *v* += *e* works otherwise similar as *v* := *v* + *e*, but the address of *v* is computed only once. The difference has seldom any consequences. It has consequences e.g. when *v* is of form $A[f(\dots)]$, where evaluation of function *f* causes side effects (like printing) As += in C++.
- = As +=, but for subtraction. As -= in C++.
- .= As +=, but for multiplication. As *= in C++.
- and** Otherwise same as \wedge , but the right side is not evaluated if the left produced **False**. As && in C++.
- for** The loop structure **for** *i* := *low* **to** *high* **do** ... **endfor** works similar as `for(i = low; i <= high; ++i){...}` in C++. When using them there is still the difference that in **for** loop it is not allowed to even try to assign to variable *i*. This way we can be sure that the loop is repeated at most $\max(0, high - low + 1)$ round.
The word **to** can be substituted by word **downto**, and then the value of the loop variable is decreased by one in every round.
- if** The condition structure **if** *condition* **then** ... **else** ... **endif** works as `if(condition){...}else{...}` in C++. The **else**-part can be left out.
- mod** Remainder operator. If $b \neq 0$, then $a \bmod b$ is the number *x* for which $b \cdot \lfloor \frac{a}{b} \rfloor + x = a$. For non-negative values of *a* and *b* it behaves as % in C++. For negative values there usually is a difference, since for integers C++ rounds the result of *a*/*b* (probably) to the zero, but $\lfloor \frac{a}{b} \rfloor$ rounds to the smaller number. To avoid this confusion the meaning of **mod** should always be checked if it is wanted use when $a < 0$ or $b < 0$.
- Nil** The value of a pointer when it is not pointing to anywhere.

or Otherwise same as \vee , but the right side is not evaluated if the left side produced True. As as `||` in C++.

repeat The loop structure **repeat** ... **until** *condition* works as `do{...}while(!condition);` in C++, in other words, “...” is evaluated at least once. Notice that the loop is exited when *condition* holds; therefore the C++ correspondence has “!” before the condition.

while The loop structure **while** *condition* **do** ... **endwhile** works as `while(condition){...}` in C++.

Greek letters

Table 1: Greek letters

	in English	suomeksi		in English	suomeksi
$A \alpha$	alpha	alfa	$N \nu$	nu	nyy
$B \beta$	beta	beeta	$\Xi \xi$	xi	ksii
$\Gamma \gamma$	gamma	gamma	$O o$	omicron	omikron
$\Delta \delta$	delta	delta	$\Pi \pi \varpi$	pi	pii
$E \epsilon \varepsilon$	epsilon	epsilon	$P \rho \rho$	rho	rhoo
$Z \zeta$	zeta	zeeta	$\Sigma \sigma \varsigma$	sigma	sigma
$H \eta$	eta	eeta	$T \tau$	tau	tau
$\Theta \theta \vartheta$	theta	theeta	$Y \upsilon$	upsilon	ypsilon
$I \iota$	iota	ioota	$\Phi \phi \varphi$	phi	phi
$K \kappa \varkappa$	kappa	kappa	$X \chi$	chi	khii
$\Lambda \lambda$	lambda	lambda	$\Psi \psi$	psi	psii
$M \mu$	mu	myy	$\Omega \omega$	omega	oomega