Scientific Communication CITS7200

Computer Science & Software Engineering

Lecture 3

The Inclusion of Symbolic Terms

Writing in a mathematical language requires special points of consideration that are not normally necessary when writing simple prose. Just as a novelist breaks up a page with dialogue, and there are rules about the placement of quotations marks and flow of such dialogue, a scientific writer dealing with mathematical terms needs to follow certain rules.

These rules for mixing a symbolic language with English are primarily designed to help the reader parse your writing in such a way that your ideas appear forceful, clear, and well-posed. This lecture contains a number of simple rules that you should always try to obey, concerning the use of symbolic terms within English prose. These rules are well-accepted by the scientific community and are likely to be imposed upon you by your supervisor, anonymous referees of your papers, or the editor of a journal. They serve collectively to reduce the cognitive load on the reader.

We begin with some very simple rules concerning the use of symbolic terms in sentences. Then we will consider how to mix symbols with English words, and finally how to write more generally about symbols.

1 Rules for symbolic terms

It is very difficult to write in computer science, or indeed any mathematical science, without introducing special symbols. These symbols could be math-

ematical terms, formulas, variable names, or language specific signs, to name but a few. The reader translates every symbol into a word, and attempts to attach a meaning to each symbol, or set of symbols. Thus the way in which you place these symbols within the text can affect this meaning, and even introduce ambiguities if care is not taken. The following rules are all absolutely standard in mathematical writing.

1. *Do not start a sentence with a symbol.*

   † $n(t)$ is a small noise component.
   A small noise component is denoted by $n(t)$.

   † `src_pid` is a parameter passed by reference.
   The parameter `src_pid` is passed by reference.

   The ambiguity this practice can produce becomes apparent with the following examples:

   † There are three possible values for this variable: -1, 0, and 1.
   1 of these values corresponds to the imaginary component
   and so can be ignored.

   † The data structure A is very powerful and can represent
   many types of images. A can represent an image of a Coke can.
   A can is a difficult image object to interpret because of the
   specularity of the metal.

2. *Always use words in preference to symbols such as* $\Rightarrow, \exists, \forall, \ni$, *etc.; unless you are writing specifically about formal logic.*

   † Since $\exists$ a nonzero solution to this recurrence relation,
   then `num = 0` $\Rightarrow$ `false := true`.

3. *When you develop an idea that follows from many formulas, do not simply list the formulas one after another (known as* homework style*). Each formula is read as a phrase in English, and the phrases should be tied together with words that help parse the global structure and aid in understanding.*

4. *Avoid subscripts and superscripts whenever possible, especially if you are tempted to use them recursively.*

   † Let $X = \{x_1, ..., x_n\}$ and consider a subset of X, say $S = \{x_{i_1}, ..., x_{i_m}\}$.

   Let X be a finite set and consider a subset S of X. Let x be a member of S.

   Do not write X_i in one place, and X_n in another, unless you have a good reason to do so. Another common example in poor symbol usage is to call a vector (x_1, x_2, x_3) in one place, and then (x, y, z) in another.

5. *Display important formulas on a line by themselves.*

   Give reference numbers to all the most important formulas, even if they are not referenced in the text. But do not overuse reference numbers in the text. Reading is a linear activity, and references are just `goto` statements that send the reader elsewhere. This can be very frustrating, without being very illuminating.

6. *Small numbers (integers less than 10) should be spelled out when used as adjectives, but not when used as names (that is, when talking about numbers as numbers).*

   † The nodes at the input layer should contain one one.

   Each node at the input layer should have a maximum of one 1 and a minimum of one 0 for its entries. A typical input node would hold 010.

7. *Do not use superfluous mathematical symbols.*

   This algorithm has $t = \log_2 n$ loops in the worst case.

   The "$t =$ " is unnecessary unless you are using this sentence to define t.

8. *Standard mathematical functions such as* sin, cos, arctan, max, gcd *etc. are set in roman type. Multiple-letter variable names and computer code are set in courier type.*

   In the following algorithm we use the data types `face` and `number`. The fast Fourier transform processes the `face` before the values of $\sin(\mu x)$ and $\sin(\nu y)$ are determined.

Typographic conventions are often useful. Traditionally, ϵ and δ denote small quantities, i, j, k, m and n are integers (although i and j also represent imaginary units), and π and e are fundamental constants. Lower case Greek letters are used for scalars, lower case roman letters are used for column vectors, and upper case Greek and roman letters are used for matrices (also for sets). LaTeX will handle many of these conventions automatically for you.

Mathematical symbols should, if possible, be the same font size as the other characters in the sentence. The expression $(n(n+1)+1)/2$ is easier to read than $\frac{n(n+1)+1}{2}$, even though the latter uses fewer characters. However, you must be careful to avoid potentially ambiguous expressions such as $a/b+c$.

If you define a formal term, for example, the Linda `out` operation, it is best to avoid using that word as if it were a word in English (for example, by using it in a verbal form). However, if you must use such a phrase, then use mixed fonts within the word.

> The data is then `out`ed and passed to the next operation.

2 Mixing symbols with English

When symbols and words appear together in the same phrase, there are special rules concerning punctuation and placement that help in parsing the phrase correctly.

9. *Symbols in distinct formulas must be separated by words.*

   † Let $\mathcal{P}$ be a set of n points M_i, $i = 1, ..., n$ of ∂X.

   > Let $\mathcal{P}$ be a set of n points M_i, where $i = 1, ..., n$
   > and each M_i is a member of ∂X.

   This can be particularly problematic when the formulas themselves contain normal punctuation marks, such as commas.

   † $D^j(n) =< f(a = 2^{-j}, b = 2^{-j}n), \phi(x) >, -J < j < 0.$

10. *Many readers will skim over formulas when they first read your manuscript. Therefore, sentences should flow smoothly when all but the simplest sets of symbols are replaced by "blah", as in the following example.*

If a multi-way merge is used, the running time can be reduced to

$$T = blah + blah + blah \tag{1}$$

seconds, where *blah* is the size of the input buffer, *blah* is the number of runs written to a temporary file, and *blah* is the number of records that can be held in main memory.

11. *Do not use unnecessary punctuation when you mix text with symbols.*

    † If C and P are subsets of A, let:
    $\mathcal{A}(\mathcal{C}, \mathcal{P}) = ...$

The colon is wrong. Correct usage of colons, and other punctuation marks, was discussed in Lecture 2.

12. *Do not use mathematical jargon when it is easier to understand the concept in words. Introduce symbols only when they are needed for precision or clarity.*

    † Let $Sk_{int}(X) = \overline{B_{max_{int}}}$ where
    $B_{max_{int}} = \{x, \exists r_x > 0, B(x, r_x) \subset X$ and if $B'(x', r') \subset X$
    with $B(x, r_x) \subset B'(x', r')$ then $x' = x, r' = r_x\}$

Rather, give the reader some idea of what you are talking about:

> We define the interior skeleton of an object X to be the closure of the set of the centres of all maximal balls that are included in X.

3 Writing about symbols

13. *The statement just preceding a theorem, algorithm, etc., should be a complete sentence or should end with a colon.*

    † We now have the following
    THEOREM. $S(x)$ is complete.

    We can now prove the following result.
    THEOREM. The state space function $S(x)$ defined in (1.5) is complete.

Even better would be to replace the first sentence by a more suggestive motivation, tying the theorem to the previous discussion.

Using the sequential execution of the language and the recursive nature of the functions defined in (1.3) and (1.4) we can now prove the following result.
THEOREM. The state space function $S(x)$ defined in (1.5) is complete.

14. *The statement of the theorem should usually be self-contained, not depending on the assumptions of the previous text.*

    This is illustrated with the example above.

15. *Do not omit the word "that" when it helps the reader parse the sentence.*

    † Assume p is a program.

    Assume that p is a program.

    Conversely, never say "we have that" or "because of the fact that". Both introduce unnecessary padding.

16. *Parallelism.*

    Vary the sentence structure and choice of words to avoid monotony.

    † The binary image obtained above is skeletonized. The skeletonized image is shown in Fig. 5. The edge detection is carried out using the procedure below.

    But use parallelism when parallel concepts are being discussed.

    † Whilst we imagine that Euclidean geometry best describes the world, in fact vision and graphics are better defined in terms of perspective geometry.

    Whilst we imagine that Euclidean geometry best describes the world, perspective geometry is more appropriate for vision and graphics.

    Consider the more subtle example below:

    We can see that $\mathcal{P} < \mathcal{Q}$ for $x < y$.

    The first "<" translates to "is less than" whilst the second reads "less than". The symbol is read differently when the same reading would reinforce the concept. A better expression would be:

    We can see that $\mathcal{P} < \mathcal{Q}$ when $x < y$.

17. *Try to state things twice in complementary ways, especially when giving a definition.*

    This reinforces the reader's understanding.

    > Next we present patterns cyclically and whenever an error is encountered, that is
    >
    > $$h_\mu = \mathbf{w} \cdot \xi_\mu \leq 0 \qquad (2)$$
    >
    > for some pattern μ, a learning step is taken.

    All variables must be defined, at least informally, when they are first introduced.

18. *Motivate the reader for what follows.*

    > We now prove that our algorithm terminates after a finite number of steps.

    Keep the reader uppermost in mind — What does the reader know so far? What does the reader expect next, and why?

19. *Capitalise names like Theorem 1, Lemma 2, Algorithm 3, Method 4.*

20. *Use a spelling checker!*

    There is no excuse for poor spelling these days. Use of a spelling checker should be systematic on all text files. However, even spelling checkers won't find semantic errors, and the following words are commonly misspelt or misused by computer scientists:

complement	not	compliment
principle	not	principal
stationary	not	stationery
implement	not	impliment
occurrence	not	occurence
auxiliary	not	auxillary
feasible	not	feasable
referring	not	refering
editing	not	editting
category	not	catagory
consistent	not	consistant
analogous	not	analagous
dependent	not	dependant
preceding	not	preceeding
discrete	not	discreet
affect (verb)	not	effect (noun)
in practice	not	in practise
descendant	not	descendent
maxima (plural)	not	maximum (singular)
schemata (plural)	not	schema (singular)
phenomenon (singular)	not	phenomena (plural)
its (belonging)	not	it's (it is)

The words "affect" and "effect" are often confused since they both have a noun and a verb form, although the above usage is the most common. "Affect", as a noun, means "a mental disposition" or a "desire or passion", as in *the affects and passions of the heart*. It is unusual that "affect" would be used as a noun in scientific writing. "Effect", as a verb, means "to bring about" or "to accomplish", as in *to effect the marriage of these two terms*.

The words "nonnegative" and "nonzero" are no longer hyphenated in the literature.

21. *Resist the temptation to use long strings of nouns as adjectives.*

    † Let us now study the complexity of the communication network null-hypothesis accumulation algorithms.