Formal Aspects of Computing (2002) 14: 99–107

Formal Aspects of Computing

EWD1300: The Notational Conventions I Adopted, and Why

Edsger W. Dijkstra

At a given moment, the concept of polite mathematics emerged, the underlying idea of which is that, even if you have only 60 readers, it pays to spend an hour if by doing so you can save your average reader a minute. By inventing an idealized 'average reader', we could translate most of the lofty, human goal of politeness into more or less formal criteria we could apply to our texts. This note is devoted to the resulting notational and stylistic conventions that were adopted as the years went by.

We don't want to baffle or puzzle our reader, in particular it should be clear what has to be done to check our argument and it should be possible to do so without pencil and paper. This dictates small, explicit steps. On the other hand it is well known that brevity is the leading characteristic of mathematical elegance, and some fear that this ideal excludes the small, explicit steps, but one of the joys of my professional life has been the discovery that this fear is unfounded, for brevity can be achieved without committing the sin of omission.

I should point out that my ideal of crisp clarity is not universally shared. Some consider the puzzles that are created by their omissions as spicy challenges, without which their texts would be boring; others shun clarity lest their work is considered trivial.

* * *

As time went by, we accepted as challenge to avoid pulling rabbits out of the magician's hat. There is a mathematical style in which proofs are presented as strings of unmotivated tricks that miraculously do the job, but we found greater intellectual satisfaction in showing how each next step in the argument, if not actually forced, is at least something sweetly reasonable to try. Another reason for avoiding rabbits as much as possible was that we did not want to teach proofs, we wanted to teach proof design. Eventually, expelling rabbits became another joy of my professional life.

I should point out that also my ideal of proofs without rabbits is not universally shared. Some authors believe that, in order to keep the reader awake, one has to tickle him with surprises. Others believe that simulated sparks of divine inspiration are essential for earning the respect of their readership.

* * *

As a final influence I must mention our desire to let the symbols do the work – more precisely: as much of the work as profitably possible –. The intuitive mathematician feels that he understands what he is talking about and uses formulae primarily to summarize situations and relations in to him familiar universes. When he seems to derive one formula from another, the transformations he allows are those that seem to be true in the universe he has in mind. The formalist, however, prefers to manipulate his formulae, temporarily ignoring all interpretations they might admit, the rules for the permissible symbol manipulations being formulated in terms of those symbols: the formalist calculates with uninterpreted formulae. While the average intuitive mathematician is perfectly happy with semantically ambiguous formulae 'because he knows what is meant', the formalist obviously insists on an unambiguous formalism.

Saying that the preferences of the formalist are not universally shared, would be the understatement of the year: intuitive mathematicians can hate more formally inclined colleagues with the same visceral hatred as Republicans can hate Democratic presidents with.

* * *

Infix notation and expressions

Without much hesitation I have decided to stick to the usual infix operators. They have the disadvantage of requiring parentheses to distinguish $(p + q)/r$ from $p + (q/r)$, which prefix notation would have rendered as $/ + pq$ and $+p/qr$ respectively.

Remark. One of the names of prefix or Polish notation is 'parenthesis-free' notation, but that is a little bit misleading, for though no longer needed, the parentheses are still permissible: we could have written $/(+pq)r$ and $+p(/qr)$ respectively. (End of Remark.)

A little hesitance was induced when my late colleague Wouter Peremans remarked in passing that infix notation was of course 'a notational relic'. It made me realize why I like it so much for associative operators: it allows us to write $p + q + r$ without being forced to choose between $(p + q) + r$ and $p + (q + r)$: in prefix notation, the choice between $+ + pqr$ and $+p + qr$ would have been unavoidable.

The main reason to stick to the infix notation for the usual operators was, of course, that we all are so terribly used to it, but the associativity does play an honest role: I did introduce – with great satisfaction! – infix operators for what I used to denote by max (x, y) and gcd (x, y), and find that I don't introduce anymore infix operators that are not associative.

Priority rules, i.e., rules of relative binding power, are the standard way of reducing the number of parenthesis pairs that would otherwise be needed, but we should remember that we are not obliged to reduce them to the absolute minimum. We learned to appreciate the following customs.

- Only use priority rules that are frequently appealed to, and hence are familiar. I would not hesitate to write $p + q \cdot r$, but would avoid $p/q \cdot r$ and $p \Rightarrow q \Rightarrow r$.
- Do not introduce priority rules that destroy symmetry. I remember how much more pleasant the predicate calculus became to work with after we had decided to give con- and disjunction the same binding power and thus to consider $p \wedge q \vee r$ an ill-formed formula.
- When introducing new infix operators, parenthesize so as to postpone decisions about binding powers until experience justifies a preference.
- When you have the freedom, choose the larger symbol for the operator with the lower binding power.
- Surround the operators with the lower binding power with more space than those with a higher binding power. E.g.,

  $$p \wedge q \quad \Rightarrow \quad r \qquad \equiv \qquad p \quad \Rightarrow \quad (q \Rightarrow r)$$

  is safely readable without knowing that $\wedge \Rightarrow \equiv$ is the order of decreasing binding power. This spacing is a vital visual aid to parsing and the fact that nowadays it is violated in many publications means that these publications do a disservice to the propagation of formal techniques. (The visual aid is vital because the mathematician does not do string manipulation but manipulation of parsed formulae: when substituting equals for equals, identifying subexpressions is essential.)
- For unary operators, give them the highest binding power and stick to prefix operators.
- For the relation/predicate calculus we settled on the following order of decreasing binding power

$\neg$	$\smile$	(negation, relational converse)
;		(relational composition)
$\wedge$	$\vee$	(conjunction, disjunction)
$\Rightarrow$	$\Leftarrow$	(implication, consequence)
$\equiv$		(equivalence)

As time went on, I learned to appreciate expressions built with operators as a way of avoiding functional notation. We can write the difference $a - b$ as $\mathrm{dif}.(a, b)$ or $\mathrm{exc}.(b, a)$, where min and exc are functions of an

ordered pair. (I believe that the notion of a 'function of 2 arguments' is now obsolete.) But we can also write the 'curried' versions $(\min\,.\,a)\,.\,b$ and $(\mathrm{sub}\,.\,b)\,.\,a$ in which, for instance, sub is a higher-order function such that $\mathrm{sub}\,.\,b$ decreases its argument by b (so that $\mathrm{sub}\,.\,(-1)$ would be the successor function). Functions dif, exc, min and sub are different functions and as soon as the operation of functional composition enters the game, the distinctions are essential. But as long as we don't do functional composition, the choice between the four options is irrelevant and writing the expression $a - b$ is a lovely way of avoiding being overspecific. (Not making superfluous distinctions should always be encouraged; it is bad enough that we don't have a canonical representation for the unordered pair.)

* * *

Invisible operators

A. N. Whitehead made many wise remarks it is a pleasure to agree with, but I cannot share his judgement when he applauds the introduction of the invisible multiplication sign. The multiplication being so common, he praises the mathematical community for the efficiency of its convention, but he ignores the price.

The one price is confusion: look at the different semantics of the juxtapositions in

$$3\tfrac{1}{2} \quad 3y \quad 32.$$

Is it a wonder that little children (many of whom have a most systematic mind) get confused by the mathematics they are taught? One would like the little kids to spend their time on learning the techniques of effective thought, but regrettably they spend a lot of it on familiarizing themselves with odd, irregular notational conventions whose wide adoption is their only virtue.

The much heavier price is that the invisible multiplication all but rules out the introduction of multiple-letter identifiers, thus creating the need of symbols of other alphabets (which – and Whitehead should have known that! – significantly contributed to the pain of typing mathematics). And when the multiple-letter identifiers could no longer be avoided, they were admitted at the price of ambiguity, as

$$\sin(s + i + n)$$

illustrates.

Remark. I also think that Whitehead made another error of judgement. Brevity is much more effectively obtained by macroscopic measures such as avoiding duplication, case analysis and superfluous nomenclature, than by such microscopic measures as making the multiplication sign invisible. (End of Remark.)

I now avoid invisible infix operators almost entirely. I do remember a few texts dealing with theorems about strings in which concatenation was denoted by juxtaposition. I could afford that because at the beginning I had declared the types of my one-letter variables: x, y, z were of type character, X, Y, Z were of type (possibly empty) character string: in xX, x would stand for the leading character of the string xX.

The most far-reaching consequence is the introduction of a visible infix operator for function application. I have given it the highest binding power and therefore I have chosen the smallest symbol I could think of: the infix dot '.'. So, instead of the traditional $f(x)$ I now write $f\,.\,x$. I have started doing this very tentatively in 1984; I have stuck to the convention and I see more and more people adopting it. One of its attractions is the way in which it caters for λ-abstraction: in 1973 I wrote 'wp (S,R)' for the weakest precondition such that statement S establishes postcondition R, but wp.S.R is much more attractive, wp.S now being the predicate transformer associated with statement S. Note that we have chosen the function application dot to be left-associative: wp.S.R is short for (wp.S).R.

Remark. The convention that function application is left-associative has been adopted almost universally. Note that the parentheses then necessary in

$$f\,.\,(g\,.\,x)$$

can be eliminated by writing the above as

$$f \circ g\,.\,x$$

and postulating that functional composition has a greater binding power than functional application. Please note that

$$f \circ g \circ h \,.\, x$$

is semantically unambiguous, functional composition being associative. (End of Remark.)

* * *

Quantification

I had three major desires with respect to quantification, (i) unambiguous identification of the dummy, (ii) clear delineation of the scope, and (iii) range as a uniform ingredient.

ad (i) The need for unambiguous dummy identification becomes clear when we consider the list of sets, which starts with the set of the first 10 squares:

$$\{i^2 | i < 10\}, \quad \{i^3 | i < 10\}, \quad \{i^n | i < 10\}, \quad \{i^n | i < n\}$$

where $\{i^{i+1}, i^{i+2}, i^{i+3}, \ldots\}$ would be a possible interpretation of the last set.

ad (ii) It is already an ambiguity with the traditional summation; compare

$$\sum_{n=1}^{3} n^2, \quad 1 + \sum_{n=1}^{3} n^2, \quad \sum_{n=1}^{3} n^2 + 1, \quad \sum_{n=1}^{3} 1 + n^2,$$

possible interpretations being, in order, 14, 15, 15 or 17, 17 or $3 + n^2$. But even logicians write

$$\forall x \; P(x) \wedge Q$$

without making clear whether they mean

$$(\forall x \; P(x)) \wedge Q \quad \text{or} \quad \forall x \; (P(x) \wedge Q)$$

which differ if the range for x is empty.

ad (iii) Although the properties of the underlying (symmetric and associative) operators may differ – addition and multiplication are not idempotent and maximum and minimum may lack a neutral element – we would like a uniform format. There is, for instance, no reason why summation should have a dummy ranging over consecutive integers.

The format is based on our format for set notation. The set that we formerly described as

$$\{i^2 | i < 10\}$$

we now denote by

$$\langle i : i < 10 : i^2 \rangle$$

where 'i' is the dummy – if there are more dummies, we separate them by commas –, '$i < 10$' is the range and 'i^2' is the term. The semicolons around the range are only separators, the 'angle brackets' delineate the scope of the dummies. The explicit enumeration of the dummies between $\langle$ and the first : acts like ALGOL 60's declaration of the local variables of an inner block. The convention enables us to distinguish between

$$\langle i : \quad i < n : \quad i^n \rangle, \quad \langle n : \quad i < n : \quad i^n \rangle \quad \text{and} \quad \langle i, n : \quad i < n : \quad i^n \rangle;$$

the last one is the set of all powers for which the exponent exceeds the base, the first two are subsets thereof.

In ALGOL 60, the declaration of a local variable includes the type of that variable, and I see no logical objection against adopting that convention: we could have written

$$\langle i \in \mathbb{N} : \quad i < 10 : \quad i^2 \rangle,$$

but I have never done so: I have always found it preferable to supply such information in the surrounding text.

Remark. It took many years before I introduced the angle brackets $\langle \; \rangle$; all that time I used the standard parentheses ( ). The transition to angle brackets for scope delineation was a great improvement. (End of Remark.)

When a quantified operator is applied to the set

$\langle i: \quad r.i: \quad t.i\rangle,$

we write it just after the opening angle bracket:

$$\left\langle \sum i: \quad r.i: \quad t.i \right\rangle, \quad \left\langle \prod i: \quad r.i: \quad t.i \right\rangle, \quad \langle \forall i: \quad r.i: \quad t.i\rangle, \quad \langle \exists i: \quad r.i: \quad t.i\rangle,$$

$$\langle \uparrow i: \quad r.i: \quad t.i\rangle \quad \text{and} \quad \langle \downarrow i: \quad r.i: \quad t.i\rangle.$$

For the first two, the type of the term $t.i$ must admit addition and multiplication, for the next two, universal and existential quantification respectively, the type of the term $t.i$ had better be boolean, while for the supremum and the infimum the surrounding text had better specify the partial order with respect to which these extremes have been defined.

From a logical point of view, universal and existential quantification don't need a variable range parameter since it can be absorbed in the term:

$$\langle \forall i: \quad r.i: \quad t.i\rangle \quad \equiv \quad \langle \forall i: \quad true: \quad r.i. \Rightarrow t.i\rangle$$
$$\langle \exists i: \quad r.i: \quad t.i\rangle \quad \equiv \quad \langle \exists i: \quad true: \quad r.i \wedge t.i\rangle.$$

But the range parameter allows more elegant formulae, such as the defining property of the supremum

$$\langle \uparrow i: \quad r.i: \quad k.i\rangle \sqsubseteq z \quad \equiv \quad \langle \forall i: \quad r.i: \quad k.i \sqsubseteq z\rangle.$$

The explicit range parameter is also welcome from a practical point of view. During many manipulations, the range can remain constant, as is for instance illustrated by de Morgan's Law

$$\neg\langle \forall i: \quad r.i: \quad t.i\rangle \equiv \langle \exists i: \quad r.i: \quad \neg t.i\rangle$$

and we shall show below the notational exploitation of such range constancy.

* * *

The proof format

When, for transitive $\rightarrow$, we conclude $A \rightarrow C$ from $A \rightarrow B$ and $B \rightarrow C$, we wish for brevity's sake to write down the intermediate result B only once. We therefore present the calculation in the following format

$$\begin{array}{ll} & A \\ \rightarrow & \quad \{\text{hint why } A \rightarrow B\} \\ & B \\ \rightarrow & \quad \{\text{hint why } B \rightarrow C\} \\ & C \end{array}$$

We shall return to the role of the hints later; we now only report that it is absolutely essential that at least one line is available for each hint, so as to discourage economizing on them.

Another, purely editorial way of reducing duplication is declaring something, once and for all, to hold for all intermediate results of a calculation rather than repeating it each time. We do this for the type specifications of variables and the range of dummies. This is a special case of 'embedding the calculation in a context'. To illustrate some of these matters, let me show a few ways of proving the following simple

Theorem 0. For associative $\cdot$, we have for all x, y, z

$$x \cdot y = y \quad \wedge \quad y \cdot z = z \quad \Rightarrow \quad x \cdot z = z.$$

Proof 0. This proof uses Leibniz's Principle in the form

$$p = q \wedge f.p \quad \Rightarrow \quad f.q,$$

and has the structure of a weakening chain of boolean expressions.

We prove Theorem 0 by observing for any x, y, z and associative $\cdot$

$$
\begin{array}{ll}
 & x \cdot y = y \quad \wedge \quad y \cdot z = z \\
\equiv & \quad \{\text{predicate calculus}\} \\
 & y \cdot z = z \quad \wedge \quad x \cdot y = y \quad \wedge \quad y \cdot z = z \\
\Rightarrow & \quad \{\text{Leibniz}\} \\
 & y \cdot z = z \quad \wedge \quad (x \cdot y) \cdot z = z \\
\equiv & \quad \{\cdot \text{ associative}\} \\
 & y \cdot z = z \quad \wedge \quad x \cdot (y \cdot z) = z \\
\Rightarrow & \quad \{\text{Leibniz}\} \\
 & x \cdot z = z \qquad \text{(End of Proof 0.)}
\end{array}
$$

Here the calculation is embedded in the context of $\cdot$ being associative, a proof shape that is suggested by the formulation of Theorem 0.

Proof 1. This proof uses Leibniz's Principle in the form

$$p = q \quad \Rightarrow \quad f.p = f.q.$$

We prove Theorem 0 by observing for associative $\cdot$ and x, y, z satisfying the antecedent

$$
\begin{array}{ll}
 & x \cdot z \\
= & \quad \{y \cdot z = z \text{ from antecedent}\} \\
 & x \cdot (y \cdot z) \\
= & \quad \{\cdot \text{ associative}\} \\
 & (x \cdot y) \cdot z \\
= & \quad \{x \cdot y = y \text{ from antecedent}\} \\
 & y \cdot z \\
= & \quad \{y \cdot z = z \text{ from antecedent}\} \\
 & z \qquad \text{(End of Proof 1.)}
\end{array}
$$

In Proof 1, we have been very – one could argue: too! – explicit in our hints: we could have started with

$$
\begin{array}{ll}
 & x \cdot z \\
= & \quad \{\text{antecedent}\} \\
 & x \cdot (y \cdot z)
\end{array}
$$

because the observed equality is obviously justified by $y \cdot z = z$, which fortunately is one of the conjuncts of the antecedent. But personally I am in favour of making the hints as clear and helpful as possible. And in the style of Proof 1 we can afford it, for we have embedded our calculation in a more elaborate context – viz. that of the antecedent – from which the hint selects a conjunct only when it is needed. This is more concise than the style of Proof 0, in which conjuncts – e.g. $y \cdot z = z$ – are dragged along as long as they are needed. In the next example we'll show another way of how 'context' can be used to avoid repetition of the same subexpression.

We recall the defining property of the supremum $\uparrow$ with respect to the complete partial order $\sqsubseteq$:

$$\langle \uparrow i : \quad r.i : \quad k.i \rangle \sqsubseteq z \quad \equiv \quad \langle \forall i : \quad r.i : \quad k.i \sqsubseteq z \rangle \quad \text{for all } z$$

Part of the Theorem of Knaster–Tarski states that with h defined by

$$h = \langle \uparrow y : \quad y \sqsubseteq f.y : \quad y \rangle$$

we have

$$h \sqsubseteq f.h$$

if f is monotonic with respect to $\sqsubseteq$, i.e.,

$$p \sqsubseteq q \quad \Rightarrow \quad f.p \sqsubseteq f.q \quad \text{for all } p, q.$$

(In another language: 'f-application is $\sqsubseteq$-preserving'.)

Proof. In this proof $y \sqsubseteq f.y$ is understood to be the range of the dummy y.

To begin with we conclude from the definitions of $\uparrow$ and h that h's defining property is that

$$h \sqsubseteq z \quad \equiv \quad \langle \forall y :: y \sqsubseteq z \rangle \quad \text{for all } z. \tag{0}$$

And now we observe

$$\begin{array}{ll} & h \sqsubseteq f.h \\ \equiv & \quad \{(0) \text{ with } z := f.h\} \\ & \langle \forall y :: y \sqsubseteq f.h \rangle \\ \Leftarrow & \quad \{\sqsubseteq \text{ transitive; } \forall \text{ monotonic}\} \\ & \langle \forall y :: y \sqsubseteq f.y \quad \wedge \quad f.y \sqsubseteq f.h \rangle \\ \equiv & \quad \{\forall \text{ distributes over } \wedge\} \\ & \langle \forall y :: y \sqsubseteq f.y \rangle \wedge \langle \forall y :: f.y \sqsubseteq f.h \rangle \\ \equiv & \quad \{\text{range is } y \sqsubseteq f.y, \text{ predicate calculus}\} \\ & \langle \forall y :: f.y \sqsubseteq f.h \rangle \\ \Leftarrow & \quad \{f \text{ monotonic; } \forall \text{ monotonic}\} \\ & \langle \forall y :: y \sqsubseteq h \rangle \\ \equiv & \quad \{(0) \text{ with } \quad z := h\} \\ & h \sqsubseteq h \\ \equiv & \quad \{\sqsubseteq \text{ reflexive}\} \\ & \text{true} \qquad \text{(End of Proof.)} \end{array}$$

Remark. As an aside I would like to point out that the second step, which strengthens

$$y \sqsubseteq f.h$$

to

$$y \sqsubseteq f.y \quad \wedge \quad f.y \sqsubseteq f.h$$

is absolutely standard. In the original relation the operands differ in two respects: y versus h and the number of f-applications, in the strengthened version each of the differences has been localized in one of the conjuncts, so that they can be dealt with separately. We could have tried the separation the other way round

$$y \sqsubseteq h \quad \wedge \quad h \sqsubseteq f.h$$

but as that choice reintroduces the original proof obligation it is not helpful. (End of Remark.)

* * *

The role of the hints

With suitable formalization, the hints would fully define the manipulations, and we could omit all the intermediate results, as they would follow. Clearly we have not done so; we consider the intermediate results an essential ingredient of the proof since it is the 'intermediate' result that indicates whether we are done or, if not, suggests what to do next.

Our hints were intended to be helpful, and as time went by, we made up our mind how we wanted to help the reader. The most important property of the hint became that it restrict and delineate the search space where the justification should be found. So, if the hint is

{predicate calculus},

then this means that the rules of the predicate calculus suffice. When anything beyond that is needed, it is something, a rule or a property, from the context in which the calculation is embedded; in that case the hint should identify that context element uniquely. For this purpose we commonly number formulae (like '(0)' in the last example).

Remark. For the formula number's place the left margin is to be preferred over the right margin, because then it is easier to maintain a small distance between the number and the formula it numbers. (A similar remark applies to page numbers in tables of contents.) (End of Remark.)

In the hints, in our last example, that referred to (0), we gave the instantiations – '$z := f\,.\,h$' and '$z := h$' respectively –, though in a simple example like this it is perhaps a little bit overdone. As soon as the pattern matching might become a problem, however, the instantiation is really helpful and we therefore don't omit it: our texts should not pose puzzles to the reader.

To save space we sometimes combine two manipulations within a single step. We do not hesitate to do so in the case of independent commuting operations on disjoint subexpressions of the intermediate result. We never combine nested substitutions, for that would be asking for trouble. For such microscopic space savings, the general advice is 'In dubio abstine.'; the space savings that can be obtained by proper proof structure – such as effective disentanglement and avoided case analyses – are greater and intellectually more satisfying.

* * *

Overloading and an unsolved dilemma

Derivation distributes over addition, i.e.,

$$\frac{d}{dt}(x+y) \quad = \quad \frac{dx}{dt} + \frac{dy}{dt}.$$

Usually we don't consider this an example of overloading, but if $x + y$ stands for a speed and t is time, than the $+$ at the right-hand side adds accelerations. Yet we don't feel obliged to introduce two different $+$ signs and to write something like

$$\frac{d}{dt}(x +_s y) \quad = \quad \frac{dx}{dt} +_a \frac{dy}{dt},$$

primarily because the absence of a notational distinction between the two additions did not introduce any ambiguity.

Let now x and y be two functions which, when applied to t yield two addable values. Then it is customary to define the function that is denoted by $x + y$ by

$$(x+y)\,.\,t \quad = \quad x\,.\,t + y\,.\,t \quad \text{for all } t.$$

Note that, formally, '$.\,t$' now distributes (to the left) over $+$. (In our former example the distribution of $\frac{d}{dt}$ was 'to the right', so we are still on reasonably familiar ground.)

But now the question arises what meaning to ascribe to $x = y$. The traditional interpretation is that $x = y$ is a boolean expression stating that x and y are the same function, i.e.,

$$x = y \quad \equiv \quad \langle \forall t :: x\,.\,t = y\,.\,t \rangle.$$

(Here we have used $\equiv$ for equality between boolean operands.) But in analogy to our previous example $x = y$ could also be interpreted as a boolean function given by

$$(x = y)\,.\,t \quad \equiv \quad x\,.\,t = y\,.\,t \quad \text{for all } t, \tag{1}$$

i.e., the boolean function that yields *true* wherever x and y coincide, and *false* everywhere else.

In many of my writings I have chosen interpretation (1). In addition I have chosen to use a pair of square brackets to denote for a boolean function universal quantification over its domain:

$$[b] \equiv \langle \forall t :: b\,.\,t \rangle. \tag{2}$$

The great advantage of the square brackets is that one does not need to introduce a dummy argument. And now one proceeds

$$\begin{array}{ll} & (x \text{ and } y \text{ are the same function}) \\ \equiv & \{\text{definition of functional equality}\} \\ & \langle \forall t :: x\,.\,t = y\,.\,t \rangle \\ \equiv & \{(1)\} \\ & \langle \forall t :: (x = y)\,.\,t \rangle \\ \equiv & \{(2) \text{ with } b := x = y\} \\ & [x = y]; \end{array}$$

the annoying moral of the story is that to express what we know and appreciate as plain and honest-to-God equality may require an extra pair of square brackets. All this is not very satisfactory, but I don't know enough of type theory to try to improve the scene.

Nuenen, 25 July 2000

prof. dr Edsger W. Dgkstra
Department of Computer Sciences
The University of Texas at Austin
Austin, TX 78712-1188
USA