The Joy of Formula Manipulation

dedicated to prof. dr. Edsger W. Dijkstra
on the occasion of his 70th birthday
and/or his retirement

In mathematical circles, there is the overall opinion that formulae, in their capacity of syntactic units, are dead things, and that formula manipulation tears the heart out of mathematics. In these circles, formulae largely live by virtue of what they stand for, of what they mean, of how they feel and appeal to our intuition – our? intuition? . And their meanings then tell which formulae to consider next. Poor Leibniz, poor Lagrange, poor Boole, poor Hilbert, and all others who shifted their attention towards uninterpreted formulae manipulation: they were all wrong, weren't they? Oh, and poor we, Edsger W. Dijkstra and all those programmers who converted themselves to formulae manipulators, because their profession demanded it.
(This cultural gap in doing mathematics was once expressed quite aptly by Dijkstra when he remarked: "Ik hou van wiskunde, maar spaar me de mathemataten." ["I love

mathematics, but it's the mathematicians I cannot stand."])

Well, our profession of programming demanded a conscious and active engagement in formula manipulation and therefore we entered that field of endeavour; and we learned —even far beyond our expectations— how mighty and powerful, how prosperous and effective, and how indicative of designing this change in attitude turned out to be, not only for the benefit of programming, but for vast parts of mathematics as well. And moreover, we learned to enjoy the activity.

In this note we try to convey the effectiveness and joy of formula manipulation through a small number of simple examples from both mathematics and programming.

0. Introducing Floor and Ceiling

In a highly respectable book on Concrete Mathematics [GKP94], we can find an entire chapter devoted to the standard mathematical functions $\lfloor\cdot\rfloor$ (floor) and $\lceil\cdot\rceil$ (ceiling). The chapter begins with a three-page introduction of the functions, stating and proving their most prominent properties, and then it proceeds with an impressive amount of applications (of over thirty pages). The three-page introduction concludes with the following passage, in which x is real and n is integer:

"
$$\begin{array}{llll} x<n & \Leftrightarrow \lfloor x\rfloor < n, & (a) & \\ n<x & \Leftrightarrow n<\lceil x\rceil, & (b) & \\ x\leq n & \Leftrightarrow \lceil x\rceil \leq n, & (c) & (3.7)\\ n\leq x & \Leftrightarrow n\leq \lfloor x\rfloor. & (d) & \end{array}$$

These rules are easily proved. [...].

It would be nice if the four rules in (3.7) were as easy to remember as they are to prove. [...]."

The passage feels as if the authors, though fully aware of the importance of the rules, only let them in as an after-thought, since, apparently, they are so difficult to remember and, therefore, so difficult to use. The suggestion is, however, unfortunate since rules (3.7.d) and (3.7.c) can serve as a beautiful starting point to uncover all properties of floor and ceiling, as we shall demonstrate next.

* * *

Functions $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ are each of type $\mathbb{R}\to\mathbb{Z}$ (real to integer), and by definition they satisfy

(0a) $\quad n\le\lfloor x\rfloor \;\equiv\; n\le x$

(0b) $\quad \lceil x\rceil\le n \;\equiv\; x\le n$,

$$\text{for } n\in\mathbb{Z}\,,\; x\in\mathbb{R}\;.$$

These are the aforementioned rules (3.7.d) and (3.7.c) respectively. The other two rules in (3.7) emerge by just negating both sides of the above equivalences:

(1a) $\quad \lfloor x\rfloor < n \;\equiv\; x<n$

(1b) $\quad n<\lceil x\rceil \;\equiv\; n<x$.

Of (0a) and (0b), we only need firmly remember one, since the other is the dual (with $\le$ and $\ge$ interchanged).

<u>Remark</u> Rules (0) are examples of so-called Galois-connections: for partial orders $\le$ and $\sqsubseteq$, functions f and g of the appropriate types are "Galois-con-

nected" whenever

$$f.x \leq y \quad \equiv \quad x \sqsubseteq g.y \qquad\qquad (\forall x, y) \ .$$

Galois-connections abound in mathematics, and it was R.C. Backhouse who strongly advocated their importance.
End of Remark.

In what follows, we mainly focus on $\lfloor . \rfloor$. In order to reassure the reader that (0a) indeed captures the traditional floor, we show that it implies the traditional definition

"$\lfloor x \rfloor$ is the greatest integer that is at most x",

which -rendered formally- is

(i) $\langle \forall n : n \leq x : n \leq \lfloor x \rfloor \rangle$ and

(ii) $\lfloor x \rfloor \leq x$:

(i) is just one half of (0a), viz. the implication from right to left;

(ii) follows from (0a) instantiated with $n := \lfloor x \rfloor$.

The reader may check that, the other way around, (0a) follows from (i) and (ii).

In spite of the equivalence between (0a) and the pair (i, ii), the greater simplicity of (0a) should not be ignored lightly. Thanks to its succinct and vigorous shape, it offers strong guidance in the arrangement and design of calculations. We shall demonstrate this through a number of examples.

* *
*

(2) $\lfloor\cdot\rfloor$ is a contracting closure, i.e.

(2a) $\lfloor x\rfloor \le x$ % $\lfloor\cdot\rfloor$ is contracting,

(2b) $\lfloor x\rfloor \le \lfloor y\rfloor$ % $\lfloor\cdot\rfloor$ is monotonic,
$\Leftarrow \quad x \le y$

(2c) $\lfloor\lfloor x\rfloor\rfloor = \lfloor x\rfloor$ % $\lfloor\cdot\rfloor$ is idempotent.

Re (2a) Follows from (0a) with $n := \lfloor x\rfloor$.

Re (2b) $\lfloor x\rfloor \le \lfloor y\rfloor$
$\equiv$ { (0a) with $n, x := \lfloor x\rfloor, y$ }
$\lfloor x\rfloor \le y$
$\Leftarrow$ { (2a) and transitivity of $\le$ }
$x \le y$.

Re (2c) • $\lfloor\lfloor x\rfloor\rfloor \le \lfloor x\rfloor$, since $\lfloor\cdot\rfloor$ is contracting;

- $\lfloor x \rfloor \le \lfloor \lfloor x \rfloor \rfloor$, from (0a) with $n, x := \lfloor x \rfloor, \lfloor x \rfloor$.

<u>End</u> of Re's .

<u>Remark</u> For someone who happens to know that property

$$f.x \le y \;\equiv\; f.x \le f.y \qquad (\forall x, y)$$

captures that f is a contracting closure, the proof of (2) is just a walk-over:

$$\lfloor x \rfloor \le y \;\equiv\; \lfloor x \rfloor \le \lfloor y \rfloor$$

is (0a) with $n, x := \lfloor x \rfloor, y$
<u>End</u> of Remark.

* * *

(3) $\quad n = \lfloor x \rfloor \;\equiv\; n \le x \;\wedge\; x < n+1$,

shown as follows:

$$\begin{array}{ll} & n \le \lfloor x \rfloor \\ \equiv & \quad \{(0a)\} \\ & n \le x \end{array} \qquad\qquad \begin{array}{ll} & \lfloor x \rfloor \le n \\ \equiv & \quad \{\text{integers}\} \\ & \lfloor x \rfloor < n+1 \\ \equiv & \quad \{(1a) \text{ with } n := n+1\} \\ & x < n+1 \; , \end{array}$$

and now conjoin the two established equivalences.

* * *

(4) $\lfloor x+n \rfloor = \lfloor x \rfloor + n$.

We prove this by an appeal to the following "rule of indirect equality":

$$c = d \equiv \langle \forall m :: m \leq c \equiv m \leq d \rangle .$$

With $c, d := \lfloor x+n \rfloor, \lfloor x \rfloor + n$ we thus observe that for any integer m

$m \leq \lfloor x+n \rfloor$

$\equiv$ { (0a) }

$m \leq x+n$

$\equiv$ { algebra }

$m-n \leq x$

$\equiv$ { (0a) }

$m-n \leq \lfloor x \rfloor$

$\equiv$ { algebra }

$m \leq \lfloor x \rfloor + n$.

<u>Remarks</u>

a. The appeal to the rule of indirect equality is not a rabbit pulled out of the magic hat: the rule belongs to the standard repertoire of the calculating mathematician

b. The other rule of indirect equality reads

$$c = d \equiv \langle \forall m :: c \leq m \equiv d \leq m \rangle ,$$

but we used the above one because the inequalities in it so nicely match

the inequalities in (0a).
c. The above proof is of the type "there is hardly anything else you can do". This is largely caused by the compelling shape of (0a) which leaves us with with hardly any manipulative freedom.
<u>End</u> of Remarks.

* * *

(5) $\lfloor x \rfloor + \lfloor y \rfloor \leq \lfloor x+y \rfloor$.

This can be shown in a variety of competing ways.

a. By an appeal to the following "rule of indirect inequality":

$$c \leq d \;\equiv\; \langle \forall m :: m \leq c \Rightarrow m \leq d \rangle \; .$$

With $c, d := \lfloor x \rfloor + \lfloor y \rfloor , \lfloor x+y \rfloor$ we thus observe that for any integer m

$$\begin{array}{ll} & m \leq \lfloor x+y \rfloor \\ \equiv & \quad \{ (0a) \} \\ & m \leq x+y \\ \Leftarrow & \quad \{ \lfloor \cdot \rfloor \text{ is contracting} \} \\ & m \leq \lfloor x \rfloor + \lfloor y \rfloor \; . \end{array}$$

b. Or more directly, as follows:

$\lfloor x \rfloor + \lfloor y \rfloor \leq \lfloor x+y \rfloor$
$\equiv$ { (0a) }
$\lfloor x \rfloor + \lfloor y \rfloor \leq x+y$
$\Leftarrow$ { algebra }
$\lfloor x \rfloor \leq x \quad \wedge \quad \lfloor y \rfloor \leq y$
$\equiv$ { $\lfloor \cdot \rfloor$ is contracting }
true .

c. Etcetera .

* * *

(6) For $0 \leq x$, $\lfloor \sqrt{\lfloor x \rfloor} \rfloor = \lfloor \sqrt{x} \rfloor$

The proof is by indirect equality. For any integer m, $0 \leq m$, and for any x , $0 \leq x$, we observe

$m \leq \lfloor \sqrt{\lfloor x \rfloor} \rfloor$
$\equiv$ { (0a) }
$m \leq \sqrt{\lfloor x \rfloor}$
$\equiv$ { algebra, using $0 \leq m$ }
$m^2 \leq \lfloor x \rfloor$
$\equiv$ { (0a) }
$m^2 \leq x$
$\equiv$ { algebra, using $0 \leq m$ and $0 \leq x$ }
$m \leq \sqrt{x}$
$\equiv$ { (0a) }
$m \leq \lfloor \sqrt{x} \rfloor$

<u>Remark</u> By restricting ourselves to the use of (0a), the three appeals to it in the above calculation are pretty predictable and unavoidable, since (6) itself contains three distinct references to function $\lfloor\cdot\rfloor$.
<u>End</u> of Remark.

* * *

(7) This is a question. For $\alpha, \beta \in \mathbb{R}$, how many integers are contained in the two-sided open interval $(\alpha..\beta)$?

The answer is $\langle \# n :: \alpha < n \wedge n < \beta \rangle$, and let us now manipulate the term of this quantified expression:

$$\begin{aligned}
& \alpha < n \ \wedge\ n < \beta \\
\equiv \quad & \{ (1a) \text{ with } x := \alpha \} \\
& \lfloor\alpha\rfloor < n \ \wedge\ n < \beta \\
\equiv \quad & \{ (1b) \text{ with } x := \beta \} \\
& \lfloor\alpha\rfloor < n \ \wedge\ n < \lceil\beta\rceil \\
\equiv \quad & \{ \text{integers} \} \\
& \lfloor\alpha\rfloor + 1 \le n \ \wedge\ n < \lceil\beta\rceil ,
\end{aligned}$$

and since both $\lfloor\alpha\rfloor$ and $\lceil\beta\rceil$ are integer, the answer is

$$(\lceil\beta\rceil - \lfloor\alpha\rfloor - 1) \ \underline{\max}\ 0 \quad .$$

* * *

And herewith we conclude our introduction to $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$, which was based on Galois-connections (0) only. Most of the examples were taken from [GKP94], which contains a wealth of additional material, fit for the kind of calculational games we have been playing here.

1. <u>DO and The Invariance Theorem</u>

In a highly respectable book on Predicate Calculus and Program Semantics [DS90], we can find an entire chapter devoted to the semantics of the repetitive construct. The heart of the chapter contains a proof of the "Main Repetition Theorem", better known as The Invariance Theorem for the repetition. The proof extends, not including the preparatory work, over nearly four pages. And then the text proceeds with the following passage:

" The Main Repetition Theorem involves well-founded sets because it deals with wp.DO, which captures guaranteed termination of the repetition. Since wlp.DO is not concerned with guaranteed termination, we may expect wlp.DO to be simpler to deal with than wp.DO ."

The passage feels as if the authors, not quite happy with the relative length of their proof, will next address the Invariance Theorem in the wlp-semantics. Quod non, and hence this little section.

* * *

We consider program DO given by

$$DO = \underline{do}\ B \rightarrow S\ \underline{od}\ .$$

From our operational understanding of a repetition, we know how to unfold it, and in doing so once, we find that

```
DO = if ¬B → skip
     [] B → S ; DO
     fi .
```

By Leibniz and the usual wlp-semantics for the if-statement and the composition, we conclude

$$[\ wlp.DO.R \\ \equiv (B \vee R) \wedge (\neg B \vee wlp.S.(wlp.DO.R))\ ] .$$

Hence, wlp.DO.R solves equation

$$X:\ [\ X \equiv (B \vee R) \wedge (\neg B \vee wlp.S.X)\ ] .$$

Because wlp.S is monotonic, the right hand side of this equation is monotonic (in X) as well, and therefore the equation has extreme solutions.
By definition, wlp.DO.R is its weakest solution. By Knaster-Tarski, wlp.DO.R therefore enjoys the following extremity property:

$$(\forall X :: [X \Rightarrow (B \vee R) \wedge (\neg B \vee wlp.S.X)] \Rightarrow [X \Rightarrow wlp.DO.R] \;) . \qquad (*)$$

Now, let us manipulate the term's antecedent with the purpose of disentangling it:

$$\begin{array}{ll} & [X \Rightarrow (B \vee R) \wedge (\neg B \vee wlp.S.X)] \\ \equiv & \{ (X \Rightarrow) \text{ and } [\cdot] \text{ over } \wedge \} \\ & [X \Rightarrow B \vee R] \wedge [X \Rightarrow \neg B \vee wlp.S.X] \\ \equiv & \{ \text{shunting} \} \\ & [X \wedge \neg B \Rightarrow R] \wedge [X \wedge B \Rightarrow wlp.S.X] . \end{array}$$

And this latter expression is quite reminiscent of the intimate relationship between wlp's and Hoare-triples:

$$\{P\}\, S\, \{Q\} \;\equiv\; [P \Rightarrow wlp.S.Q] \; .$$

By the above calculation, extremity

property (*) rendered in Hoare-triple format reads:

$$\langle \forall X :: [X \wedge \neg B \Rightarrow R] \wedge \{X \wedge B\} S \{X\} \\ \Rightarrow \\ \{X\} DO \{R\} \quad \rangle ,$$

or, in the traditional format of an inference rule,

$$\frac{[X \wedge \neg B \Rightarrow R] \quad , \quad \{X \wedge B\} S \{X\}}{\{X\} DO \{R\}} ,$$

C.A.R. Hoare's famous Theorem of Invariance.

* * *

A few final remarks are in order.

- By definition, wp.DO.R is the strongest solution of the equation of which wlp.DO.R is the weakest solution. The demonstrandum in the Invariance Theorem is $\{X\} DO \{R\}$, or rather

$$[X \Rightarrow w(l)p.DO.R] ,$$

and from this we see that there is no use for the extremity property of wp.DO.R, which is a strongest solution. And indeed, in proving the Invariance

Theorem, [DS90] only uses that wp.DO.R solves the equation. In our wlp-context we only used the extremity property, and this diffence might be food for further thought.

- One may wonder by what kind of traditional mathematical intuition it could become apparent that the extremity property of wlp.DO.R equivales the Theorem of Invariance (in spite of the fact that the formal calculation is so remarkably simple).

- The simplicity with which the Invariance Theorem emerges from the definition of wlp.DO casts doubts on the adequacy of the concept wp.

And indeed, also in everyday practice, the programmer almost always chooses his variant function to start with, and only then will he be bothered by invariances, thus fully separating the concerns of progress and partial correctness.

2. Reconsidering the Binary Search

It was in the mid 1970s, when Edsger W. Dijkstra — at the time heavily engaged in the formal derivation of programs — designed the "ultimate" Binary Search program. For given integer X and ascending integer array $A[0..N]$, satisfying

(0) $A[0] \leq X \wedge X < A[N]$,

he designed a program to establish

(1) $A[i] \leq X \wedge X < A[i+1]$

(with indices within the array bounds), without an appeal to the ascendingness of A. This latter property only entered the picture in concluding, from (1) alone, the absence of value X in array A. Twenty-five years later, he revisited the design with the intention to be even more explicit about the design considerations [EWD1293].

Dijkstra's program is "ultimate" in the sense that the Binary Search itself is absolutely independent of the ascendingness of the array, a fact that up to this very day flabbergasts or even confuses most audiences, because they were taught otherwise.

* * *

There is however a little more to be said about the algorithm, and this becomes apparent when we abstract a little from the specifics of (0) and (1). Let Z be some binary relation on integers, such that, for $c < d$, condition

(2) $\quad c \; Z \; d$

is satisfied. Our goal is to explore circumstances that allow us to conclude the existence of some i, $c \le i \wedge i+1 \le d$, satisfying

(3) $\quad i \; Z \; (i+1)$.

This is a generalization of Dijksta's original problem indeed: with Z defined as

$$p \; Z \; q \quad \equiv \quad A[p] \le X \; \wedge \; X < A[q] \; ,$$

(0) $\equiv \quad 0 \; Z \; N$, and
(1) $\equiv \quad i \; Z \; (i+1)$.

Without much ado, we now give the program that establishes (3) on the premise of (2), the technicalities being largely the same as in Dijkstra's original design (and besides: easily checked by the reader).

```
{$c<d$}{$c \; Z \; d$}
 $i, j := c, d$
;{inv: $P0 \wedge P1$}{vf: $j-i$}
 do $j \neq i+1 \rightarrow$
        {$i+1<j$} $h := (i+j)$ div $2$
      ; {$i<h<j$}{$i \; Z \; j$, from $P1$}
        if $h \; Z \; j \rightarrow i := h$
        [] $i \; Z \; h \rightarrow j := h$
        fi
 od
 {$P1 \wedge j = i+1$, hence $i \; Z \; (i+1)$}
```

where

$P0$: $c \leq i \wedge i<j \wedge j \leq d$

$P1$: $i \; Z \; j$.

The only difference with Dijkstra's program is that we now have to be concerned about the absence of abortion in the if-clause (which is for free in Dijkstra's special case). There is no danger of abortion if in the initial state of the if-statement at least one of the guards is true, i.e. if

$$i\ Z\ h \ \vee\ h\ Z\ j\ .$$

Since we know nothing about Z, this had better be a consequence of pre-condition $i\ Z\ j$ of the if-statement. That is, we had better insist that Z satisfy

$$i\ Z\ h\ \vee\ h\ Z\ j\ \Leftarrow\ i\ Z\ j\ ,$$

which equivales - counterpositive and De Morgan -

$$i\ (\neg Z)\ h\ \wedge\ h\ (\neg Z)\ j\ \Rightarrow\ i\ (\neg Z)\ j\ ,$$

a condition which is fulfilled for transitive $\neg Z$.

And herewith we identified a whole class of situations in which the "Binary Search" is applicable. The uncovering of this would never have been possible without Dijkstra's "ultimate" program, and without his work on making programming into a scientific discipline.

3. References

[DS90]

Edsger W. Dijkstra and Carel S. Scholten ;
Predicate Calculus and Program Semantics ;
Springer Verlag, New York, 1990 .

[EWD 1293]

Edsger W. Dijkstra ;
Constructing the Binary Search once more ;
Technical Note, Austin TX, November 1999 .

[GKP94]

Ronald L. Graham, Donald E. Knuth,
and Oren Patashnik ;
Concrete Mathematics (2nd ed.) ;
Addison-Wesley Publishing Company, Inc,
1994 .

May 2000

W.H.J. Feijen,
Department of Mathematics and Computing Science,
Technical University of Eindhoven,
5600 MB Eindhoven,
The Netherlands