Founded 1989
Editor–in–chief:
Johan T. Jeuring

The Squiggolist

The Squiggolist, Volume 1, Number 3, February 1990

LONDON — on presentation. "The general thesis which I advance can only be understood when one reads my work · ·tively, which is only ar ·on of it; not that · ' point of vie· ·nplica- r· · my b· ·t so · readers, self, I th· tendency in avoid calculatio· jects which I tr· which, I recognized, i· surmountable difficulty those who would want proceed generally with the material that I treated."

E. Galois, Preface, 1832. (Translation J. Gray) In: J. Fauvel and J. Gray. The History of Mathematics: A Reader. MacMillan Education Ltd., 1987.

LONDON — Operators still not properly distributed. The telecommunication companies keep complaining about the distribution of operators. Though the situation seems to be less urgent than at the time of appearance of the previous issue of this newspaper, some companies have announced measures. These measures might be successful, as shown by a company from Groningen, which, together with its relations, has tackled many of the problems caused by the malfunction of distributivity with success.

Utrecht — Transformation system sold. Utrecht University has sold the transformation system built by the Computing Science department to Volcam bv for fl.50,000. Both parties were very happy with the deal: Utrecht University because the system had been built by a student who paid for rather ·n got paid for building ·stem, and Volcam bv according to a : 'Program · is the future, ·ant to miss

Homomorphisms, Factorisation and Promotion

Nico Verwer

The Largest Ascending Substr·
—An exercise in nub theory

Johan Jeuring

Shall we calculate-II

Richard Bird

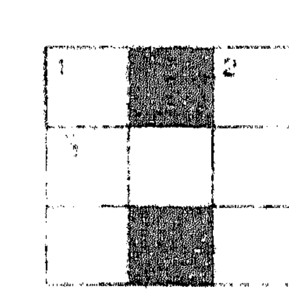
·suspect· Pub· ·n inv· affairs Utrecht · years, S. ·· tax-papers o· leagues, shar· Thus he al· income. F· this ext· tax-p·

On Induced Congruences

Roland Backhouse and Grant Malcolm

OXFORD — Tabloid news. First there were monoids, then there were binoids. Now we know about greedoids too. Richoid Boid (of Oxfoid University) has emploid several students on the matroid problem. "I enjoid the work, although devoid of success, and this annoid me; I convoid my feelings to Richoid, and took up a job in the city (Lloids)" one student said. Paranoid, Boid supploid unalloid praise of the Matroid. "Years of work have been destroid, and we are left with a void, which having toid with the ideas I was keen to avoid. I reloid too much on others, and this deloid results." Boid is undergoing Froidian analysis.

'The complete article is avail- · on celluloid, in the form ·head projector sloids ·son Wesley]



1. $\oplus/ \cdot (\oplus/ \cdot f)*$
2. $h \cdot g** \cdot \mathsf{inits}$
3. Suppose $T = +\!\!+/ \cdot f* \cdot \mathsf{tails}$
   Rewrite $+\!\!+/ \cdot f* \cdot \mathsf{segs}$

Squiggoling in Context

Grant Malcolm

address: Centre for Mathematics and Computer Science
Dept. of Algorithmics & Architecture
P.O. Box 4079
1009 AB Amsterdam
The Netherlands
(jt@cwi.nl)

This is the third issue of the Squiggolist. The Squiggolist is a forum for people who work with the Bird–Meertens formalism. It is meant for the quick distribution of short papers, summaries of results, or current points of interest. You cannot subscribe to the Squiggolist: either you receive it, or you don't

Since until now the size of each issue is twice the size of its preceding issue, I expect to send around thick books soon. I will produce the next issue of the Squiggolist in May or June, so please send your contributions to me before the end of May.

Submit your contributions (camera–ready copy or a LaTeX–file) in A4 format. They will be reduced to A5 ($\times$ 0.71), so use pointsize 12. There are no restrictions on the fonts used in the camera–ready copy: it may be LaTeX, handwritten or typewritten, as long as it is black on white, readable and large enough to be turned into A5. I will be the editor, and contributions should be sent to me:

Johan Jeuring
CWI, dept AA
P.O. Box 4079
1009 AB Amsterdam
The Netherlands
email: jt@cwi.nl

On Induced Congruences

Roland Backhouse
Grant Malcolm

Groningen University

In this note we present a proof of a theorem to be found in Ehrig and Mahr [3]. The theorem states that a relation constructed from a given function is a congruence relation iff that function is a homomorphism; we go on to generalise this result to the relational homomorphisms treated by the first author in [1]. Specifically, we prove that a congruence relation can be constructed from a relational homomorphism. Our construction generalises that of Ehrig and Mahr. The significance of this note lies both in the economy of our calculations and in the novel use we make of weakest prespecifications.

We have been unable to extend the theorem to an equivalence: we offer the remaining half of the equivalence as a challenge to the reader.

We consider a type $\mathcal{T}$ defined according to the paradigm $\mathcal{T} = \mu(\tau : \mathsf{F})$, where F is a *relator* (such types are special cases of "Hagino types", details of which may be found in Hagino [4]). For the purposes of this note, the important properties of a relator F are the following.

- if α is a type, then $\alpha\mathsf{F}$ is a type;
- if $f \in \alpha \leftarrow \beta$ is a function, then there is a function $f\mathsf{F} \in \alpha\mathsf{F} \leftarrow \beta\mathsf{F}$;
- if $R \in \alpha \sim \beta$ is a relation, then there is a relation $R\mathsf{F} \in \alpha\mathsf{F} \sim \beta\mathsf{F}$;
- $R\mathsf{F} \circ S\mathsf{F} = (R \circ S)\mathsf{F}$, for R and S either functions or relations (in fact, we shall adopt the position that functions are special cases of relations, with the property that $x\langle f\rangle y \equiv x = f.y$: hence composition of relations and of functions is the same thing);
- $\mathrm{I}\mathsf{F} = \mathrm{I}$, where I denotes the identity relation of the appropriate type;
- if $R \supseteq S$, then $R\mathsf{F} \supseteq S\mathsf{F}$; and
- $(R\mathsf{F})\cup = (R\cup)\mathsf{F}$, where $R\cup$ denotes the reverse of relation R; i.e., for all x and y of the appropriate types, $x\langle R\cup\rangle y \equiv y\langle R\rangle x$.

The type $\mathcal{T}$ can be viewed as the least fixed point of the relator F, whose constructor is the total function $\tau \in \mathcal{T} \leftarrow \mathcal{T}\mathsf{F}$. The type enjoys the following unique extension property: given $R \in \beta \sim \beta\mathsf{F}$, there is a unique relation $(\![R]\!) \in \beta \sim \mathcal{T}$ which satisfies

(1) $$(\![R]\!) \circ \tau = R \circ (\![R]\!)\mathsf{F}.$$

Moreover, if R is a function, so too is $(\![R]\!)$.

As already remarked, we consider functions to be special cases of relations; their additional properties are captured in the following definition.

Definition 1 (total functions) That f is a function is expressed by

(functionality) $\qquad \mathrm{I} \supseteq f \circ f^{\cup}$

and that it is total by

(totality) $\qquad f^{\cup} \circ f \supseteq \mathrm{I}.$

Moreover, f is injective iff $f^{\cup}$ is functional, and f is surjective iff $f^{\cup}$ is total. □

Definition 2 For $R \in \alpha \sim \beta$ and $S \in \gamma \sim \delta$, the relation $R \longleftarrow S \in (\alpha \leftarrow \gamma) \sim (\beta \leftarrow \delta)$ is defined by: for all f and g,

$$f\langle R \longleftarrow S\rangle g \quad \equiv \quad R \circ g \supseteq f \circ S.$$

□

This overloading of the $\leftarrow$ operator as a constructor of both types and relations has been used severally by Wadler, de Bruin and Backhouse [6,2,1] to investigate properties of polymorphic functions. Such overloading encourages us to confuse types and relations yet further and write $e \in R$ if $e\langle R\rangle e$.

The relational calculus allows us to formulate the following elegant definition of congruence relations.

Definition 3 (congruence) Relation $R \in \mathcal{T} \sim \mathcal{T}$ is a congruence relation if it is an equivalence relation and respects the structure of $\mathcal{T}$; that is, if R is reflexive: $R \supseteq \mathrm{I}$, transitive: $R \supseteq R \circ R$, symmetric: $R = R^{\cup}$, and furthermore $\tau \in R \longleftarrow R\mathsf{F}$.
□

Elementary properties of equivalence relations will be assumed, namely: R is reflexive iff $R^{\cup}$ is reflexive, and R is transitive iff $R^{\cup}$ is transitive.

We now prove the theorem on congruence relations from Ehrig and Mahr ([3], p. 77).

Theorem 4 (induced congruences) For total functions $f \in B \leftarrow \mathcal{T}$, $f^{\cup} \circ f$ is a congruence relation if and only if f is a homomorphism.
Proof: by mutual implication.
($\Leftarrow$): It is straightforward to show that $f^{\cup} \circ f$ is an equivalence relation, for all functions f; we prove only that a homomorphism $f = (\![g]\!)$ respects the structure of $\mathcal{T}$; i.e., $\tau \in (f^{\cup} \circ f) \longleftarrow (f^{\cup} \circ f)\mathsf{F}$. By definition 2, this means we have to show that

$$f^{\cup} \circ f \circ \tau \supseteq \tau \circ (f^{\cup} \circ f)\mathsf{F}.$$

We calculate as follows:

$$
\begin{array}{ll}
& f_\cup \circ f \circ \tau \\
\supseteq & \quad \{ \text{ functionality of } \tau \ \} \\
& \tau \circ \tau_\cup \circ f_\cup \circ f \circ \tau \\
= & \quad \{ \text{ reverse } \} \\
& \tau \circ (f \circ \tau)_\cup \circ f \circ \tau \\
= & \quad \{ \text{ (1), twice } \} \\
& \tau \circ (g \circ f_\mathsf{F})_\cup \circ g \circ f_\mathsf{F} \\
= & \quad \{ \text{ reverse } \} \\
& \tau \circ f_{\mathsf{F}\cup} \circ g_\cup \circ g \circ f_\mathsf{F} \\
\supseteq & \quad \{ \text{ totality of } g\text{; relators } \} \\
& \tau \circ (f_\cup \circ f)_\mathsf{F}
\end{array}
$$

($\Rightarrow$): Suppose now that $f_\cup \circ f$ is a congruence relation; i.e.

(2) $$f_\cup \circ f \circ \tau \ \supseteq \ \tau \circ (f_\cup \circ f)_\mathsf{F}.$$

We have to find $g \in \beta \leftarrow \beta_\mathsf{F}$ such that $(\![g]\!) = f$. From type considerations alone we are led to the following choice: $g \triangleq f \circ \tau \circ (f_\cup)_\mathsf{F}$ and we must show

(3) $$f \circ \tau \ = \ g \circ f_\mathsf{F}$$

whence by the unique extension property, $(\![g]\!) = f$, and thus f is a homomorphism. We prove (3) by mutual inclusion:

$$
\begin{array}{ll}
& g \circ f_\mathsf{F} \\
= & \quad \{ \text{ defn. } g \ \} \\
& f \circ \tau \circ (f_\cup)_\mathsf{F} \circ f_\mathsf{F} \\
= & \quad \{ \text{ relators } \} \\
& f \circ \tau \circ (f_\cup \circ f)_\mathsf{F} \\
\supseteq & \quad \{ \text{ totality of } f\text{; monotonicity; identity } \} \\
& f \circ \tau
\end{array}
$$

So far we have not used that $f_\cup \circ f$ is a congruence; we do need that assumption to prove the other inclusion:

$$
\begin{array}{ll}
& g \circ f_\mathsf{F} \\
= & \quad \{ \text{ defn. } g\text{, relators } \} \\
& f \circ \tau \circ (f_\cup \circ f)_\mathsf{F} \\
\subseteq & \quad \{ \text{ (2) } \} \\
& f \circ f_\cup \circ f \circ \tau \\
\subseteq & \quad \{ \text{ functionality of } f \ \} \\
& f \circ \tau
\end{array}
$$

Note that, in general, g need not be a total function, since it makes use of $f_\cup$. However, functionality of g follows straightforwardly from the property that $f_\cup \circ f$ is a congruence; a sufficient condition for g to be total is that f be surjective.
□

In the above, a congruence relation was constructed from a functional homomorphism; we now turn to the question of whether it is possible to generalise this to the construction of a congruence relation from a *relational* homomorphism. We formulate the generalised construction with the aid of the following definition.

Definition 5 For a relation $R \in \alpha \sim \beta$, the relation $R\dagger \in \beta \sim \beta$ is defined by the following property: for all S,

$$R\dagger \supseteq S \quad \equiv \quad R \supseteq R \circ S.$$

□

The relation $R\dagger$ is the "weakest prespecification" $R \backslash R$ of Hoare and He Jifeng (see [5]); its equational presentation lends itself well to the sort of calculational style of proof in which we are interested.

Theorem 6 $f^{\cup} \circ f = f\dagger$.
Proof: we first note that for all R, $f^{\cup} \circ f \supseteq R \quad \equiv \quad f \supseteq f \circ R$:

$$\begin{array}{ll} & f \supseteq f \circ R \\ \Rightarrow & \qquad \{ \text{ monotonicity } \} \\ & f^{\cup} \circ f \supseteq f^{\cup} \circ f \circ R \\ \Rightarrow & \qquad \{ \text{ totality of } f \text{ } \} \\ & f^{\cup} \circ f \supseteq R \\ \Rightarrow & \qquad \{ \text{ monotonicity } \} \\ & f \circ f^{\cup} \circ f \supseteq f \circ R \\ \Rightarrow & \qquad \{ \text{ functionality of } f \text{ } \} \\ & f \supseteq f \circ R \end{array}$$

Hence, by definition 5, $f^{\cup} \circ f = f\dagger$.
□

Property 7 $R\dagger$ is reflexive.
Proof: $R \supseteq R \circ \mathrm{I}$, hence by definition 5, $R\dagger \supseteq \mathrm{I}$.
□

Property 8 $R \supseteq R \circ R\dagger$.
Proof: $R\dagger \supseteq R\dagger$, hence by definition 5, $R \supseteq R \circ R\dagger$.
□

Property 9 $R\dagger$ is transitive.
Proof:

$$\begin{array}{ll} & R\dagger \supseteq R\dagger \circ R\dagger \\ \equiv & \qquad \{ \text{ defn. 5 } \} \\ & R \supseteq R \circ R\dagger \circ R\dagger \\ \Leftarrow & \qquad \{ \text{ property 8, twice } \} \\ & R \supseteq R \\ \equiv & \\ & \mathsf{true} \end{array}$$

□

Corollary 10 $(R\dagger) \cap (R\dagger)^{\cup}$ is an equivalence relation.
Proof: intersection preserves reflexivity and transitivity.
□

Since $R\dagger$ is not in general symmetric, we have had to take the intersection of $R\dagger$ with its own reverse to obtain symmetry. Note however that if R is a total function, then $(R\dagger) \cap (R\dagger)^{\cup} \;=\; R\dagger$, so taking the intersection is simply a generalisation of the previous construction. We have, then, constructed an equivalence relation, but is it also a congruence relation? The following lemmata allow us to give a positive answer.

Lemma 11 $\tau \in R \longleftarrow R\mathsf{F} \;\;\equiv\;\; \tau \in R^{\cup} \longleftarrow (R^{\cup})\mathsf{F}$.
Proof:

$$
\begin{array}{ll}
 & \tau \in R \longleftarrow R\mathsf{F} \\
\equiv & \qquad \{ \text{ defn. 2 } \} \\
 & R \circ \tau \;\supseteq\; \tau \circ R\mathsf{F} \\
\equiv & \qquad \{ \text{ reverse; relators } \} \\
 & \tau^{\cup} \circ R^{\cup} \;\supseteq\; (R^{\cup})\mathsf{F} \circ \tau^{\cup} \\
\Leftarrow & \qquad \{ \text{ monotonicity; functionality and totality of } \tau \text{ } \} \\
 & R^{\cup} \circ \tau \;\supseteq\; \tau \circ (R^{\cup})\mathsf{F} \\
\equiv & \qquad \{ \text{ defn. 2 } \} \\
 & \tau \in R^{\cup} \longleftarrow (R^{\cup})\mathsf{F}
\end{array}
$$

This shows $\tau \in R\mathsf{F} \longleftarrow R \;\;\Leftarrow\;\; \tau \in (R^{\cup})\mathsf{F} \longleftarrow R^{\cup}$; since R was arbitrary, we may replace it by $R^{\cup}$ and so obtain the desired equivalence.
□

Lemma 12 If R is a homomorphism, then $\tau \in R\dagger \longleftarrow (R\dagger)\mathsf{F}$.
Proof: Let R be the homomorphism $(\![S]\!)$.

$$
\begin{array}{ll}
 & \tau \in R\dagger \longleftarrow (R\dagger)\mathsf{F} \\
\equiv & \qquad \{ \text{ defn. 5 } \} \\
 & R\dagger \circ \tau \;\supseteq\; \tau \circ (R\dagger)\mathsf{F} \\
\Leftarrow & \qquad \{ \text{ monotonicity; totality of } \tau \text{ } \} \\
 & R\dagger \;\supseteq\; \tau \circ (R\dagger)\mathsf{F} \circ \tau^{\cup} \\
\equiv & \qquad \{ \text{ defn. 5 } \} \\
 & R \;\supseteq\; R \circ \tau \circ (R\dagger)\mathsf{F} \circ \tau^{\cup} \\
\equiv & \qquad \{ \text{ (1) } \} \\
 & R \;\supseteq\; S \circ R\mathsf{F} \circ (R\dagger)\mathsf{F} \circ \tau^{\cup} \\
\equiv & \qquad \{ \text{ relators } \} \\
 & R \;\supseteq\; S \circ (R \circ R\dagger)\mathsf{F} \circ \tau^{\cup} \\
\Leftarrow & \qquad \{ \text{ property 8 } \} \\
 & R \;\supseteq\; S \circ R\mathsf{F} \circ \tau^{\cup} \\
\Leftarrow & \qquad \{ \text{ monotonicity; functionality of } \tau \text{ } \} \\
 & R \circ \tau \;\supseteq\; S \circ R\mathsf{F} \\
\equiv & \qquad \{ \text{ (1) } \} \\
 & \mathsf{true}
\end{array}
$$

□

Lemma 13 If $\tau \in R \longleftarrow R_F$ and $\tau \in S \longleftarrow S_F$, then $\tau \in (R \cap S) \longleftarrow (R \cap S)_F$.
Proof: Assume the antecedents; i.e.,

$$R \circ \tau \supseteq \tau \circ R_F \tag{4}$$

$$S \circ \tau \supseteq \tau \circ S_F \tag{5}$$

then we calculate:

$$\begin{array}{ll} & (R \cap S) \circ \tau \\ = & \qquad \{ \text{ set theory, } \tau \text{ is a function } \} \\ & (R \circ \tau) \cap (S \circ \tau) \\ \supseteq & \qquad \{ \text{ (4) and (5); monotonicity } \} \\ & (\tau \circ R_F) \cap (\tau \circ S_F) \\ \supseteq & \qquad \{ \text{ set theory } \} \\ & \tau \circ (R_F \cap S_F) \\ \supseteq & \qquad \{ \text{ monotonicity of relators } \} \\ & \tau \circ (R \cap S)_F \end{array}$$

□

Corollary 14 If R is a relational homomorphism, then $(R\dagger) \cap (R\dagger)^\cup$ is a congruence relation.
□

The open question that we leave to the reader is whether every congruence relation on $\mathcal{T}$ can be expressed in the form $(R\dagger) \cap (R\dagger)^\cup$, where R is a relational homomorphism.

Acknowledgement: Peter de Bruin pointed out to us that the component g of the homomorphism constructed in the proof of theorem 4 is not necessarily total.

References

[1] R.C. Backhouse. Naturality of homomorphisms. Lecture notes, International Summer School on Constructive Algorithmics, vol. 3, 1989.

[2] P.J. de Bruin. Naturalness of polymorphism. 1989. Department of Mathematics and Computing Science, University of Groningen.

[3] H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 1. EATCS Monographs on Theoretical Computer Science*, Springer-Verlag Berlin, 1985.

[4] T. Hagino. A typed lambda calculus with categorical type constructors. In D.H. Pitt, A. Poigne, and D.E. Rydeheard, editors, *Category Theory and Computer Science*, pages 140–57, Springer-Verlag Lecture Notes in Computer Science 283, 1988.

[5] C.A.R. Hoare and Jifeng He. The weakest prespecification. *Fundamenta Informaticae*, 9:51–84, 217–252, 1986.

[6] P. Wadler. Theorems for free! March 1989. Draft report, Dept. Comp. Science, University of Glasgow.

Shall we calculate - II?

Richard S. Bird
Programming Research Group, Oxford University

In the first issue of The Squiggolist I posed the following exercise. Define

(0) $$Eq\,f = \sqcap_{\#}/ \cdot all\; q_f\triangleleft \cdot Inv(\cup/)$$

where $q_f\, x$ is the condition that the set x is nonempty and every value of x maps to the same value under f. Prove that

(1) $$Eq\,f \cdot p\triangleleft = (\neq \{\ \})\triangleleft \cdot p\triangleleft * \cdot Eq\,f$$

In this note I shall try and solve the exercise. In fact a somewhat more general result is proved. Consider the following generalisation of (0):

(2) $$\Pi p = \sqcap_{\#}/ \cdot all\; p\triangleleft \cdot Inv(\cup/)$$

Applied to a nonempty set x, Πp returns some coarsest partition of x all of whose components satisfy the set predicate p. Note that the binary operator $\sqcap_{\#}$ returns the smaller of its two arguments under an ordering that respects size ($\#$). Without specifying this ordering any further, the reduction $\sqcap_{\#}/$ is guaranteed only to produce some coarsest partition. For example, with $p\,x = +/\,x \leqslant 6$, both $\{\{1,2,3\},\{4\}\}$ and $\{\{1,4\},\{2,3\}\}$ are coarsest partitions of $\{1,2,3,4\}$.

There is, however, a simple condition on p that ensures a unique coarsest partition, independent of details of the ordering. Say p is *overlap-closed* if

$$p\,(x \cup y) = p\,x \wedge p\,y$$

for all x and y with $x \cap y \neq \{\ \}$. One example of an overlap-closed predicate is q_f, as the reader can easily check. Another example is given below.

We shall prove the following generalisation of (1): if q is overlap-closed, then

(3) $$\Pi q \cdot p\triangleleft = (\neq \{\,\})\triangleleft \cdot p\triangleleft * \cdot \Pi q$$

For the proof we require the following two definitions:

$$\begin{aligned} \mathit{same}\; p &= \mathit{all}\; p \vee \mathit{all}\,(\neg p) \\ \rho\, x &= (x = \{\,\} \to \{\,\},\ \{x\}) \end{aligned}$$

In outline the proof of (3) is:

$$\begin{aligned} & \Pi q \cdot p\triangleleft \\ = & \quad \{\text{claim (6); see below}\} \\ & \mathit{all}\; p\triangleleft \cdot \cup/ \cdot \Pi q * \cdot \Pi(\mathit{same}\; p) \\ = & \quad \{\text{claim (9); see below}\} \\ & \mathit{all}\; p\triangleleft \cdot \cup/ \cdot \Pi(\mathit{same}\; p) * \cdot \Pi q \\ = & \quad \{\text{filter promotion}\} \\ & \cup/ \cdot (\mathit{all}\; p\triangleleft \cdot \Pi(\mathit{same}\; p)) * \cdot \Pi q \\ = & \quad \{\text{claim (5); see below}\} \\ & \cup/ \cdot (\rho \cdot p\triangleleft) * \cdot \Pi q \\ = & \quad \{\text{map distributivity}\} \\ & \cup/ \cdot \rho * \cdot p\triangleleft * \cdot \Pi q \\ = & \quad \{\text{definition of } (\neq \{\,\})\triangleleft\} \\ & (\neq \{\,\})\triangleleft \cdot p\triangleleft * \cdot \Pi q \end{aligned}$$

It remains to prove the claims. Claims (6) and (5) depend on

(4) $$\Pi(\mathit{same}\; p)\, x = (\neq \{\,\}) \triangleleft \{p \triangleleft x, \neg p \triangleleft x\}$$

We shall not prove (4). The proof of (5), viz

(5) $$\mathit{all}\; p\triangleleft \cdot \Pi(\mathit{same}\; p) = \rho \cdot p\triangleleft$$

is:

$$
\begin{array}{ll}
 & all\ p \triangleleft \Pi(same\ p)\ x \\
= & \quad \{(4)\} \\
 & all\ p \triangleleft (\neq \{\ \}) \triangleleft \{p \triangleleft x, \neg p \triangleleft x\} \\
= & \quad \{\text{filters commute}\} \\
 & (\neq \{\ \}) \triangleleft all\ p \triangleleft \{p \triangleleft x, \neg p \triangleleft x\} \\
= & \quad \{\text{definition of } all\ p\} \\
 & (\neq \{\ \}) \triangleleft \{p \triangleleft x\} \\
= & \quad \{\text{definition of } \rho\} \\
 & \rho\,(p \triangleleft x)
\end{array}
$$

The proof of claim (6), viz

(6) $\qquad \Pi q \cdot p\triangleleft = all\ p\triangleleft \cdot \cup/\ \cdot \Pi q* \cdot \Pi(same\ p)$

is as follows:

$$
\begin{array}{ll}
 & \Pi q\,(p \triangleleft x) \\
= & \quad \{\text{claim; see below}\} \\
 & \cup/\ \{all\ p \triangleleft \Pi q\,(p \triangleleft x), all\ p \triangleleft \Pi q\,(\neg p \triangleleft x)\} \\
= & \quad \{\text{definition of } *\} \\
 & \cup/\ (all\ p\triangleleft \cdot \Pi q) * \{p \triangleleft x, \neg p \triangleleft x\} \\
= & \quad \{\text{filter promotion}\} \\
 & all\ p \triangleleft \cup/\ \Pi q * \{p \triangleleft x, \neg p \triangleleft x\} \\
= & \quad \{\Pi q\{\ \} = \{\ \}\} \\
 & all\ p \triangleleft \cup/\ \Pi q * (\neq \{\ \}) \triangleleft \{p \triangleleft x, \neg p \triangleleft x\} \\
= & \quad \{(4)\} \\
 & all\ p \triangleleft \cup/\ \Pi q * \Pi(same\ p)\ x
\end{array}
$$

The claim used in the first step follows from:

$$\begin{array}{rcl} all\, p \triangleleft \Pi q\,(p \triangleleft x) & = & p \triangleleft x \\ all\, p \triangleleft \Pi q(\neg p \triangleleft x) & = & \{\,\} \end{array}$$

This leaves claim (9). So far we have not exploited the assumption that q is overlap-closed. Now we do so. If q is overlap-closed, then for any x there is a unique coarsest partition of x into components satisfying q. In fact more is true: if q is overlap-closed, then the set of partitions of x into components, each of which satisfy q, is equal to the set of refinements of any coarsest partition. In particular, if there are two coarsest partitions, then each is a refinement of the other, and so they are equal. Expressed equationally, we have:

(7) $$all\, q\triangleleft \cdot Inv(\cup/\,) = \times_{\cup}/ \cdot Inv(\cup/\,)* \cdot \Pi q$$

whenever q is overlap-closed. We shall not prove (7). Note that $\times_{\cup}$ is not an idempotent operator, so the reduction $\times_{\cup}/$ is not defined over (arbitrary) sets. However, in the given context we can interpret $\cup$ as *disjoint* set union, since the reduction $\times_{\cup}/$ is applied to a set of pairwise disjoint sets (of sets).

We shall prove that if p and q are both overlap-closed, then

(8) $$\cup/ \cdot \Pi p* \cdot \Pi q = \cup/ \cdot \Pi q* \cdot \Pi p$$

In (8) the operator $\cup$ can again be interpreted as disjoint set union since the reduction $\cup/$ is over pairwise disjoint sets. Claim (9), viz

(9) $$\cup/ \cdot \Pi q* \cdot \Pi(same\, p) = \cup/ \cdot \Pi(same\, p)* \cdot \Pi q$$

follows from (8) since the predicate *same p* is overlap-closed.

We prove (8) by showing that the left-hand side is equal to $\Pi(p \wedge q)$, from which the result follows by commutativity of $\wedge$.

$$\begin{array}{ll} & \Pi(p \wedge q) \\ = & \quad \{(2)\} \\ & \sqcap_{\#}/ \cdot all\,(p \wedge q)\triangleleft \cdot Inv(\cup/\,) \\ = & \quad \{\text{property of } all\text{; see below}\} \\ & \sqcap_{\#}/ \cdot all\, p\triangleleft \cdot all\, q\triangleleft \cdot Inv(\cup/\,) \end{array}$$

Squiggoling in Context

Grant Malcolm
Groningen University

Distributivity properties lie at the very heart of Squiggol; in particular they underlie the promotion theorems for the various data structures most commonly used in programming (lists, bags, rose trees, term algebras...). While such data structures have been quite succesfully incorporated into Squiggol, there remains the problem that certain properties, such as distributivity, are enjoyed by only a subset of a given structure; for example, heap-sorted trees or lists of a fixed length. In such cases promotion cannot be applied, and it often seems that the only recourse is to proof by induction (anathema to the ardent Squiggolist!). In this note we examine such a case: a distributivity property holds on the range of a given function, and we seek some means of including this contextual information in our calculations in order that we may use promotion and avoid explicit recourse to induction. To this end we "move up" into the realm of relations, exploiting the properties of homomorphic relations as investigated by Backhouse [1].

The problem we address is stated in terms of non-empty snoc lists: we begin with a brief revision of the type structure.

Definition 1 (non-empty snoc lists) For each type α, the type of non-empty snoc lists over α (denoted by $\alpha+$) has two constructors: the singleton constructor $\tau \in \alpha+ \leftarrow \alpha$, and concatenation $\rightarrowtail\!\!+ \in \alpha+ \leftarrow \alpha+\times\alpha$; and for every $f \in \beta \leftarrow \alpha$ and $\oplus \in \beta \leftarrow \beta\times\alpha$, there is a unique function $(\![f,\oplus]\!) \in \beta \leftarrow \alpha+$ such that:

$$(1) \qquad (\![f,\oplus]\!)\circ\tau \;=\; f$$

$$(2) \qquad (\![f,\oplus]\!)\circ\rightarrowtail\!\!+ \;=\; \oplus\circ(\![f,\oplus]\!)\times\mathrm{I}.$$

(For functions $g \in \gamma \leftarrow \alpha$ and $h \in \delta \leftarrow \beta$, the function $g\times h \in \gamma\times\delta \leftarrow \alpha\times\beta$ takes the pair $\langle x,y\rangle$ to the pair $\langle g.x, h.y\rangle$. In the above equations and henceforth we give $\times$ higher priority than $\circ$.)
□

We call such functions $(\![f,\oplus]\!)$ "homomorphisms". From the uniqueness property of homomorphisms we can prove that $(\![\tau,\rightarrowtail\!\!+]\!)$ is the identity function in $\alpha+ \leftarrow \alpha+$, a fact we shall use later on. Two common examples are maps and reductions:

Definition 2 (map) For $f \in \beta \leftarrow \alpha$, define $f+ \;\triangleq\; (\![\tau\circ f, \rightarrowtail\!\!+\circ\mathrm{I}\times f]\!) \in \beta+ \leftarrow \alpha+$.
□

Definition 3 (reduction) For $\oplus \in \alpha \leftarrow \alpha\times\alpha$, define $\oplus/ \;\triangleq\; (\![\mathrm{I},\oplus]\!) \in \alpha \leftarrow \alpha+$.
□

The promotion theorem for the type is derived in the standard way (see Malcolm [5]):

Theorem 4 (promotion) For $g \in \gamma \leftarrow \beta$, $f \in \beta \leftarrow \alpha$, $\oplus \in \beta \leftarrow \beta \times \alpha$ and $\otimes \in \gamma \leftarrow \gamma \times \alpha$,

$$g \circ (\!|f, \oplus|\!) = (\!|g \circ f, \otimes|\!) \quad \Leftarrow \quad g \circ \oplus = \otimes \circ g \times \mathrm{I}.$$

□

One can now use promotion to verify the following properties of maps and reductions.

Property 5 $\oplus/ \circ f+ = (\!|f, \oplus \circ \mathrm{I} \times f|\!)$.
□

Property 6 $f+ \circ g+ = (f \circ g)+$.
□

The statement of the problem we shall consider is due to Zwiggelaar [6], and comes from his investigation of the "aggregated segment sums" of Backhouse (see [2]). In solving the longest ascending sequence problem, the following subgoal occurs:

(3) $$\Uparrow/ \circ (\!|\diamond 1, \otimes|\!)+ \circ \mathsf{tls} \;=\; (\!|\diamond 1, \oslash|\!) \;\in\; \mathbb{Z} \times \mathbb{N} \leftarrow \mathbb{Z}+.$$

where, for $m, m' \in \mathbb{Z}$, $l, l' \in \mathbb{N}$, $x \in \mathbb{Z}{+}{+}$ and $y \in \mathbb{Z} \times \mathbb{N}$:

$$\begin{array}{rcl} \langle m, l\rangle \Uparrow \langle m', l'\rangle &=& \mathsf{if}\ l \geq l'\ \mathsf{then}\ \langle m, l\rangle\ \mathsf{else}\ \langle m', l'\rangle\ \mathsf{fi} \\ (\diamond 1).m &=& \langle m, 1\rangle \\ \langle m, l\rangle \otimes m' &=& \langle m', \mathsf{if}\ m \leq m'\ \mathsf{then}\ l+1\ \mathsf{else}\ 1\ \mathsf{fi}\rangle \\ \mathsf{tls} &=& (\!|\tau \circ \tau, \odot|\!) \\ x \odot m &=& ((\succ\!\!+ m)+.x) \succ\!\!+ \tau.m \\ y \oslash m &=& (y \otimes m) \Uparrow \langle m, 1\rangle \end{array}$$

The difficulty in proving equation (3) is that we require that $\otimes$ distributes backwards through $\Uparrow$ (in that case the equality follows by Horner's rule, cf. Backhouse [2]). However, the distributivity property does not hold in general: the crucial observation made by Zwiggelaar is that it does hold when the first components of the pairs are equal, i.e., for all m, l, l' and a:

(4) $$(\langle m, l\rangle \Uparrow \langle m, l'\rangle) \otimes a \;=\; (\langle m, l\rangle \otimes a) \Uparrow (\langle m, l'\rangle \otimes a).$$

Now it is the case that in the lists of pairs in the range of the function $(\!|\diamond 1, \otimes|\!)+ \circ \mathsf{tls}$ all the first components are equal, and Zwiggelaar's inductive proof of (3) makes use of this property: in the remainder of this note we construct a calculational proof in which we can make use of the fact that we are working in the context of the range of the above function. In order to be able to use contextual information of this kind we introduce the notion of guards: these have been studied by Hesselink in the context of command algebras and program transformation (see [3,4]).

$$
\begin{array}{ll}
= & \{(7)\} \\
& \sqcap_{\#}/ \cdot \mathit{all}\, p\triangleleft \cdot \mathsf{X}_{\cup}/ \cdot \mathit{Inv}(\cup/)* \cdot \Pi q \\
= & \{\text{filter cross promotion; see below}\} \\
& \sqcap_{\#}/ \cdot \mathsf{X}_{\cup}/ \cdot (\mathit{all}\, p\triangleleft \cdot \mathit{Inv}(\cup/))* \cdot \Pi q \\
= & \{\text{reduce cross promtion; see below}\} \\
& \cup/ \cdot (\sqcap_{\#}/ \cdot \mathit{all}\, p\triangleleft \cdot \mathit{Inv}(\cup/))* \cdot \Pi q \\
= & \{(2)\} \\
& \cup/ \cdot \Pi p* \cdot \Pi q
\end{array}
$$

The property of *all* used in the calculation is the combination of

$$
\begin{array}{rcl}
\mathit{all}\,(p \wedge q) & = & \mathit{all}\, p \wedge \mathit{all}\, q \\
(P \wedge Q)\triangleleft & = & P\triangleleft \cdot Q\triangleleft
\end{array}
$$

The filter cross promotion law is

$$\mathit{all}\, p\triangleleft \cdot \mathsf{X}_{\cup}/ = \mathsf{X}_{\cup}/ \cdot (\mathit{all}\, p\triangleleft)*$$

The reduce cross promotion law is: if $\otimes$ distributes over $\oplus$, then

$$\oplus/ \cdot \mathsf{X}_{\otimes}/ = \otimes/ \cdot \oplus/*$$

Use of this law is justified in the given context, since for pairwise disjoint sets x, y, z we have

$$(x \sqcap_{\#} y) \cup z = (x \cup z) \sqcap_{\#} (y \cup z)$$

In other words, $\cup$ distributes through $\sqcap_{\#}$

This completes the justification of the claims and the proof of the exercise. For what it achieves the calculation is surprisingly complicated. Moreover, there are one or two omitted proofs. Can it be simplified?

Definition 7 (guards) For predicate $p \in \mathsf{Bool} \leftarrow \alpha$, define the *guard* $p? \in \alpha \sim \alpha$ by:

$$a\langle p?\rangle b \quad \equiv \quad a = b \ \wedge \ p.b.$$

Thus $p?$ is the restriction of the identity on α to those elements satisfying p.
□

Property 8 (idempotence) $p? \circ p? = p?$.
□

Property 9 (precondition) If f is a (partial) function, then $p? \circ f = f \circ (p \circ f)?$. (Note that we treat functions f also as relations, where $a\langle f\rangle b \equiv a = f.b$.)
□

By introducing guards and working in the domain of relations we do not compromise our ability to calculate, as evidenced by the following result proven by Backhouse in [1]:

Theorem 10 (generalised promotion) The promotion theorem (thm. 4) also holds for relations; i.e., when $g \in \gamma \sim \beta$, $f \in \beta \sim \alpha$, $\oplus \in \beta \sim \beta \times \alpha$ and $\otimes \in \gamma \sim \gamma \times \alpha$ are all relations.
□

We can now construct some lemmas on conditional distributivity.

Definition 11 (invariance) We say that predicate p is $\oplus$-*invariant* if for all x and y, $p.(x \oplus y) \Leftarrow p.x \wedge p.y$. This implication is equivalent to the equation:

$$p? \circ \oplus \circ p? \times p? \ = \ \oplus \circ p? \times p?.$$

□

The reader can easily check that equality of first components is $\Uparrow$-invariant: for any m, construct the predicate $(=m \circ \mathsf{fst})$ and we have for all x and y:

$$\mathsf{fst}.(x \Uparrow y) = m \quad \Leftarrow \quad \mathsf{fst}.x = m \ \wedge \ \mathsf{fst}.y = m. \tag{5}$$

Property 12 If p is $\oplus$-invariant, then $\oplus\!/ \circ (p?)\!+ = p? \circ (\!|p?,\ \oplus \circ p? \times p?|\!)$.
Proof:

$$\begin{array}{cl}
 & \oplus\!/ \circ (p?)\!+ = p? \circ (\!|p?,\ \oplus \circ p? \times p?|\!) \\
\equiv & \qquad \{ \text{ property 5 } \} \\
 & (\!|p?,\ \oplus \circ \mathrm{I} \times p?|\!) = p? \circ (\!|p?,\ \oplus \circ p? \times p?|\!) \\
\Leftarrow & \qquad \{ \text{ promotion; } p? \circ p? = p? \ \} \\
 & p? \circ \oplus \circ p? \times p? = \oplus \circ \mathrm{I} \times p? \circ p? \times \mathrm{I} \\
\equiv & \qquad \{ \text{ defn. 11 } \} \\
 & p \text{ is } \oplus\text{-invariant}
\end{array}$$

□

Property 13 (conditional distributivity) Suppose p is $\oplus$-invariant and $f \in \alpha \leftarrow \alpha$ distributes over $\oplus$ on condition p:

$$f \circ \oplus \circ p? \times p? \;=\; \oplus \circ f \times f \circ p? \times p?$$

(or, equivalently, $f.(x \oplus y) = f.x \oplus f.y \;\Leftarrow\; p.x \wedge p.y$); then

$$f \circ \oplus/ \circ (p?)+ \;=\; \oplus/ \circ f+ \circ (p?)+.$$

Proof:

$$\begin{array}{cl}
 & f \circ \oplus/ \circ (p?)+ = \oplus/ \circ f+ \circ (p?)+ \\
\equiv & \qquad \{ \text{ property 12; properties 6, 5 } \} \\
 & f \circ p? \circ (\!| p?, \oplus \circ p? \times p? |\!) = (\!| f \circ p?, \oplus \circ \mathrm{I} \times (f \circ p?) |\!) \\
\Leftarrow & \qquad \{ \text{ promotion; } f \circ p? \circ p? = f \circ p? \ \} \\
 & f \circ p? \circ \oplus \circ p? \times p? = \oplus \circ \mathrm{I} \times (f \circ p?) \circ (f \circ p?) \times \mathrm{I} \\
\equiv & \qquad \{ \ p \text{ is } \oplus\text{-invariant } \} \\
 & f \circ \oplus \circ p? \times p? = \oplus \circ f \times f \circ p? \times p?
\end{array}$$

□

With reference to the problem in hand, (4) gives the following conditional distribution

$$(\otimes a) \circ \Uparrow \circ (=m \circ \mathsf{fst})? \times (=m \circ \mathsf{fst})? \;=\; \Uparrow \circ (\otimes a) \times (\otimes a) \circ (=m \circ \mathsf{fst})? \times (=m \circ \mathsf{fst})?$$

and we already have the invariance property (5), so property 13 gives:

(6) $$(\otimes a) \circ \Uparrow/ \circ (=m \circ \mathsf{fst})?+ \;=\; \Uparrow/ \circ (\otimes a)+ \circ (=m \circ \mathsf{fst})?+$$

We return now to the proof of (3): in fact, we shall use promotion to prove:

(7) $$\Uparrow/ \circ (\!| \diamond 1, \otimes |\!)+ \circ \mathsf{tls} \circ (\!| \tau, \rightarrowtail |\!) = (\!| \diamond 1, \oslash |\!).$$

This is equivalent to (3) since $(\!| \tau, \rightarrowtail |\!)$ is the identity homomorphism. Promotion then yields the following subgoals:

(8) $$\Uparrow/ \circ (\!| \diamond 1, \otimes |\!)+ \circ \mathsf{tls} \circ \tau \;=\; \diamond 1$$

(9) $$\Uparrow/ \circ (\!| \diamond 1, \otimes |\!)+ \circ \mathsf{tls} \circ \rightarrowtail \;=\; \oslash \circ (\Uparrow/ \circ (\!| \diamond 1, \otimes |\!)+ \circ \mathsf{tls}) \times \mathrm{I}$$

The former is straightforward calculation using (1); we concentrate on the latter. We shall use the following lemmas, the proofs of which are straightforward and omitted. We define $\mathsf{lst} \triangleq \mathsf{snd}/$.

Lemma 14 For all $x \in \alpha+$, $\mathsf{tls} \circ (=x)? \;=\; ((=\mathsf{lst}.x \circ \mathsf{lst})?)+ \circ \mathsf{tls} \circ (=x)?$.
□

Lemma 15 $\mathsf{fst} \circ (\!| \diamond 1, \otimes |\!) = \mathsf{lst}$.
□

Finally, the proof of (9) is given below: we use guards to introduce contextual information, and then use the lemmas above to push this information leftwards through the expression until we can apply conditional distributivity. The guards which are so propagated always hold in their particular context, playing the role of comments in a program text: since the guards always hold, they can simply be omitted when no longer required.

$$
\begin{array}{ll}
 & (\Uparrow\!/ \circ (\!|\diamond 1, \otimes|\!)\!+ \circ \mathsf{tls}).(x \mathrel{>\!\!\!+} a) \\
= & \qquad \{ \text{ evaluation, using (2) and } \triangleq \oslash \ \} \\
 & (\Uparrow\!/ \circ ((\!|\diamond 1, \otimes|\!) \circ (\mathrel{>\!\!\!+} a))\!+ \circ \mathsf{tls})\, .x \;\Uparrow\; \langle a, 1\rangle \\
= & \qquad \{ \text{ (2); introduce context: } x = x \ \} \\
 & (\Uparrow\!/ \circ (\otimes a)\!+ \circ (\!|\diamond 1, \otimes|\!)\!+ \circ \mathsf{tls} \circ (=x)?)\, .x \;\Uparrow\; \langle a, 1\rangle \\
= & \qquad \{ \text{ lemma 14; property 6 } \} \\
 & (\Uparrow\!/ \circ (\otimes a)\!+ \circ ((\!|\diamond 1, \otimes|\!) \circ (=\mathsf{lst}.x \circ \mathsf{lst})?)\, + \circ \mathsf{tls})\, .x \;\Uparrow\; \langle a, 1\rangle \\
= & \qquad \{ \text{ lemma 15 } \} \\
 & (\Uparrow\!/ \circ (\otimes a)\!+ \circ ((\!|\diamond 1, \otimes|\!) \circ (=\mathsf{lst}.x \circ \mathsf{fst} \circ (\!|\diamond 1, \otimes|\!))?)\, + \circ \mathsf{tls})\, .x \;\Uparrow\; \langle a, 1\rangle \\
= & \qquad \{ \text{ properties 9 and 6 } \} \\
 & (\Uparrow\!/ \circ (\otimes a)\!+ \circ (=\mathsf{lst}.x \circ \mathsf{fst})?\!+ \circ (\!|\diamond 1, \otimes|\!)\!+ \circ \mathsf{tls})\, .x \;\Uparrow\; \langle a, 1\rangle \\
= & \qquad \{ \text{ (6) } \} \\
 & ((\otimes a) \circ \Uparrow\!/ \circ (\!|\diamond 1, \otimes|\!)\!+ \circ \mathsf{tls})\, .x \;\Uparrow\; \langle a, 1\rangle \\
= & \qquad \{ \triangleq \oslash \ \} \\
 & (\Uparrow\!/ \circ (\!|\diamond 1, \otimes|\!)\!+ \circ \mathsf{tls})\, .x \oslash a
\end{array}
$$

The above proof suggests that guards can be used effectively in calculational proofs, even though we did cheat in the sense that introducing the identity homomorphism into the equation (7) meant that we were effectively using induction under the name of promotion. Backhouse's demonstration in [1] that the equational approach to homomorphisms which we had developed in [5] could be generalised to relations provided the setting for the use of guards.

References

[1] R.C. Backhouse. Naturality of homomorphisms. Lecture notes, International Summer School on Constructive Algorithmics, vol. 3, 1989.

[2] R.C. Backhouse. Aggregated segment sums. Dept. Comp. Sci., Groningen University, 1989.

[3] W.H. Hesselink. *An algebraic calculus of commands.* Technical Report CS 8808, Dept. Comp. Sci., Groningen University, 1988.

[4] W.H. Hesselink. *Command algebras, recursion and program transformation.* Technical Report CS 8812, Dept. Comp. Sci., Groningen University, 1988.

[5] G. Malcolm. Homomorphisms and promotability. In J.L.A. van de Snepscheut, editor, *Conference on the Mathematics of Program Construction*, pages 335–347, Springer-Verlag LNCS 375, 1989.

[6] F. Zwiggelaar. A new Horner's rule. Afstudeerverslag, Dept. Comp. Sci., Groningen University, 1989.

The Largest Ascending Substree
— An exercise in nub theory —

Johan Jeuring*
CWI, Amsterdam

1 Introduction

Given a binary labelled tree, a substree is a special subtree of that tree. Substrees can be viewed as the equivalent of subsequences on binary labelled trees. In this paper we derive an algorithm for finding the largest ascending substree of a tree. The main motivation for undertaking this exercise is to apply nub theory, a theory for deriving algorithms, introduced by de Moor and Bird in [dMB89]. Nub theory helps to solve problems which require the tupling of extra information. Problems which can be solved using nub theory are the longest upsequence problem and the "Mark Thatcher" problem. Both of these problems are defined on lists; we solve a problem on binary labelled trees. For this purpose, we have to generalize part of nub theory slightly. We suppose the reader is familiar with the theory of binary labelled trees described in [Jeu89].

2 Substrees and substree promotion

A substree, defined on binary labelled trees, is the equivalent of a subsequence defined on lists. Substrees are computed by means of a function subs, which is defined by

$$\begin{array}{lcl} \mathsf{subs}\ \langle a \rangle & = & [\langle a \rangle] \\ \mathsf{subs}\ l \swarrow b \searrow r & = & [\langle b \rangle] +\!\!+ (\mathsf{subs}\ l) +\!\!+ (\mathsf{subs}\ r) +\!\!+ ((\swarrow b \searrow)* ((\mathsf{subs}\ l) \mathsf{X} (\mathsf{subs}\ r)))\ . \end{array}$$

Hence subs is a tree homomorphism $\swarrow\!\!\times\!\!\searrow / \cdot f*$, where $f\ a = [\langle a \rangle]$, and the ternary operator $\swarrow\!\!\times\!\!\searrow$ is defined by

$$l \swarrow\!\!\times\!\!\searrow_b r = [\langle b \rangle] +\!\!+ l +\!\!+ r +\!\!+ ((\swarrow b \searrow)* (l \mathsf{X} r))\ .$$

Because the definition of X we use deviates from the previous definitions, and because X plays an important role in the subsequent sections, we give its exact definition.

$$x \mathsf{X} y = +\!\!+ /\ \nabla y * x$$

$$a \nabla x = (a||)*\ x$$

$$a||b = (a, b)\ .$$

*This research has been supported by the Dutch organisation for scientific research NWO,under project-nr. NF 62.518.

We derive a promotion theorem for subs, that is, we give conditions under which a homomorphism composed with the function subs equals a tree homomorphism. We have, from [Jeu89],

$$
\begin{array}{ll}
 & \oplus/ \cdot g* \cdot \mathsf{subs} \\
= & \qquad \text{promotion theorem for trees} \\
 & \odot/ \cdot h* \, ,
\end{array}
$$

provided

$$\oplus/\ g*\ x \,{\swarrow\!\!\times\!\!\searrow}_b\, y = (\oplus/\ g*\ x) \odot_b (\oplus/\ g*\ y) \, ,$$

for some operator $\odot$. In order to determine $\odot$, we derive

$$
\begin{array}{ll}
 & \oplus/\ g*\ x \,{\swarrow\!\!\times\!\!\searrow}_b\, y \\
= & \qquad \text{definition of } {\swarrow\!\!\times\!\!\searrow} \\
 & \oplus/\ g*\ [\langle b \rangle] +\!\!\!+\, x +\!\!\!+\, y +\!\!\!+\, ((\swarrow b \searrow)*\,(x \mathsf{X} y)) \\
= & \qquad \text{property of homomorphism} \\
 & (g\ \langle b \rangle) \oplus (\oplus/\ g*\ x) \oplus (\oplus/\ g*\ y) \oplus (\oplus/\ g*\ (\swarrow b \searrow)*\,(x \mathsf{X} y)) \, .
\end{array}
$$

The first three arguments of $\oplus$ need not be developed any further. For the last argument we have (see also [Jeu89])

$$\oplus/\ g*\ (\swarrow b \searrow)*\,(x \mathsf{X} y) = (\oplus/\ g*\ x) \otimes_b (\oplus/\ g*\ y) \, ,$$

provided g is a tree homomorphism $\otimes/ \cdot j*$ such that $\otimes$ distributes through $\oplus$. We have proved the following theorem.

Theorem 1 (subs–promotion) *Let h be a homomorphism $\oplus/ \cdot g*$ defined on lists, and g a tree homomorphism $\otimes/ \cdot j*$ such that $\otimes$ distributes through $\oplus$. Then*

$$h \cdot \mathsf{subs} = \odot/ \cdot (g \cdot \langle \cdot \rangle)* \, ,$$

where

$$x \odot_b y = (g\ \langle b \rangle) \oplus x \oplus y \oplus (x \otimes_b y) \, .$$

3 The largest ascending subtreecut problem

The specification of the largest ascending subtreecut problem reads

$$\uparrow_{\#}/ \cdot \mathit{ascending} \triangleleft \cdot \mathsf{subs} \, ,$$

where $\#$ measures the size of a tree. The operator $\uparrow_{\#}$ is underspecified. It will have to be refined later. If we rewrite $\uparrow_{\#}/ \cdot \mathit{ascending} \triangleleft$ into the form of a homomorphism $\uparrow_{\#}/ \cdot f*$, we have that f satisfies the first condition of the subs–promotion theorem, but it fails to satisfy the second condition, i.e., f is a tree homomorphism $\otimes/ \cdot j*$, but the operator $\otimes$ we obtain does not distribute through $\uparrow_{\#}$. It follows that the subs–promotion theorem is not applicable.

4 Nub theory

A special function, called the nub function, has been introduced by de Moor and Bird in [dMB89] to overcome difficulties encountered when distributivity conditions are not satisfied (see for example the previous section). Informally, given a list x, a total pre-order $\olessthan$, and a total linear order $\oslash$, $\mathsf{nub}(\olessthan, \oslash)$ enumerates the elements of x in increasing $\olessthan$-order, choosing one representative from every $\olessthan$-equivalent class using the order $\oslash$. We will choose as representative the minimum element under $\oslash$, but any selector function suffices.

The nub function is a homomorphism defined by

$$\mathsf{nub}(\olessthan, \oslash) = \Wedge / \cdot \tau * \,,$$

where $1_{+\!\!+}$ is the identity element of $\Wedge$, and

$$\begin{array}{lll} ([a] +\!\!+ x) \Wedge ([b] +\!\!+ y) = & [a] +\!\!+ (x \Wedge ([b] +\!\!+ y)) & \text{if } a \olessthan b \\ & [a \downarrow_{\oslash} b] +\!\!+ (x \Wedge y) & \text{if } a \circledequal b \\ & [b] +\!\!+ (([a] +\!\!+ x) \Wedge y) & \text{if } b \olessthan a \,, \end{array}$$

where $\circledequal$ is the equivalence relation generated by the order $\olessthan$.

In [dMB89] a number of properties of nub (an abbreviation for $\mathsf{nub}(\olessthan, \oslash)$) is listed. The proofs of these properties are left as exercises, but apparently they are carried out for the class/representative characterisation of nub. Using the homomorphic characterisation of nub, the proofs become more elegant. Every proof consists of two parts: the (high level) calculational part using promotion, and the (element level) part where the promotability conditions are verified.

We give proofs of the filter-nub rule, a modified version of the map-nub rule (both from [dMB89]), and the cross-nub rule. In each proof we leave the verification of the promotability conditions to the reader.

Given a pre-order $\olessthan$ and a linear order $\oslash$, the order $\olessthan \,\&\, \oslash$ is defined by

$$a(\olessthan \,\&\, \oslash)b = a \olessthan b \vee (a \circledequal b \wedge b \oslash a) \,.$$

We have the following property (called the no-loss lemma) for nub

(1) $$\uparrow_{\olessthan \& \oslash} / = \gg / \cdot \mathsf{nub}(\olessthan, \oslash) \,.$$

The function $p\triangleleft$ may be promoted over nub provided a condition is satisfied. This is expressed by the following law.

Lemma 2 (filter-nub rule) *If p satisfies*

$$p\, y \wedge x \circledequal y \wedge x \oslash y \Rightarrow p\, x \,,$$

then

(2) $$p\triangleleft \cdot \mathsf{nub}(\olessthan, \oslash) = \mathsf{nub}(\olessthan, \oslash) \cdot p\triangleleft \,.$$

Proof by promotion. We have that $p\triangleleft$ is $(\Wedge, \Wedge)$–promotable on singletons, i.e.,

$$\begin{array}{rcl} p\triangleleft\, 1_{+\!\!+} & = & 1_{+\!\!+} \\ p\triangleleft\, ([x] \Wedge [y]) & = & (p\triangleleft\, [x]) \Wedge (p\triangleleft\, [y]) \,, \end{array}$$

if p satisfies the above condition. Hence

$$
\begin{array}{cl}
 & p\triangleleft \cdot \mathsf{nub} \\
= & \qquad \text{definition of } \mathsf{nub} \\
 & p\triangleleft \cdot \wedge\!\!\wedge/ \cdot \tau* \\
= & \qquad \text{promotion theorem} \\
 & \wedge\!\!\wedge/ \cdot (p\triangleleft)* \cdot \tau* \\
= & \qquad \text{map distributivity} \\
 & \wedge\!\!\wedge/ \cdot (p\triangleleft \cdot \tau)* \\
= & \qquad \text{equality below} \\
 & \wedge\!\!\wedge/ \cdot \tau* \cdot p\triangleleft \\
= & \qquad \text{definition of } \mathsf{nub} \\
 & \mathsf{nub} \cdot p\triangleleft
\end{array}
$$

The equality applied in the above derivation reads

$$\tau* \cdot p\triangleleft = (p\triangleleft \cdot \tau)* \ ,$$

for all predicates p. The proof of this equality (using the unique extension property) is easy and omitted.
(End of Proof)

We need a slightly more general version than the one given in [dMB89] of the map-nub rule.

Lemma 3 (map-nub rule) *Given a function f, let $\olessthan$ and $\olessthan'$ be pre-orders, and $\ogreaterthan$ and $\ogreaterthan'$ linear orders satisfying for all x and y*

$$
\begin{array}{lcl}
x \olessthan' y & \Rightarrow & x \olessthan_f y \\
x \ogreaterthan' y & \Rightarrow & x \ogreaterthan_f y \ ,
\end{array}
$$

then

(3) $$f* \cdot \mathsf{nub}(\olessthan, \ogreaterthan) = \mathsf{nub}(\olessthan', \ogreaterthan') \cdot f* \ .$$

Proof by promotion. If for some relations $\ogreaterthan$ and $\ogreaterthan'$, f satisfies the implications listed above, then $f*$ is $(\wedge\!\!\wedge', \wedge\!\!\wedge)$–promotable on singletons, where $\wedge\!\!\wedge'$ is the operator of the reduction part of $\mathsf{nub}(\olessthan', \ogreaterthan')$. The function $\mathsf{nub}(\olessthan', \ogreaterthan')$ is abbreviated to nub'. We have

$$
\begin{array}{cl}
 & \mathsf{nub}(\olessthan, \ogreaterthan) \cdot f* \\
= & \qquad \text{definition of } \mathsf{nub} \\
 & \wedge\!\!\wedge/ \cdot \tau* \cdot f* \\
= & \qquad \text{map distributivity, one-point rule, map distributivity} \\
 & \wedge\!\!\wedge/ \cdot f** \cdot \tau* \\
= & \qquad \text{promotion theorem} \\
 & f* \cdot \wedge\!\!\wedge'/ \cdot \tau* \\
= & \qquad \text{definition of } \mathsf{nub}' \\
 & f* \cdot \mathsf{nub}(\olessthan', \ogreaterthan')
\end{array}
$$

(End of Proof)

Finally, we have a rule not mentioned in [dMB89], the cross-nub rule.

Lemma 4 (cross-nub rule) *Let $\olessthan$ and $\olessthan'$ be pre-orders, and $\ogreaterthan$ and $\ogreaterthan'$ linear orders such that for all x, y and a*

$$
\begin{array}{lcl}
x \olessthan y & \Rightarrow & ((a \| x) \olessthan' (a \| y)) \wedge ((x \| a) \olessthan' (y \| a)) \\
x \ogreaterthan y & \Rightarrow & ((a \| x) \ogreaterthan' (a \| y)) \wedge ((x \| a) \ogreaterthan' (y \| a)) \; .
\end{array}
$$

Then

(4) $$\mathsf{nub}(\olessthan', \ogreaterthan') \cdot \mathsf{X} \;= \Xi \cdot (\mathsf{nub}(\olessthan, \ogreaterthan) \| \mathsf{nub}(\olessthan, \ogreaterthan)) \; ,$$

where Ξ is defined by

$$x \,\Xi\, y = \Lambda\!\!\Lambda' / \, (\nabla y) * \, x$$

Proof by promotion. We have

$$
\begin{array}{ll}
 & \mathsf{nub}(\olessthan', \ogreaterthan') \; x \, \mathsf{X} \, y \\
= & \quad \text{definition of nub and cross} \\
 & \Lambda\!\!\Lambda' / \; \tau * \, \mathbin{+\!\!\!+} / \; (\nabla y) * \, x \\
= & \quad \text{reduction promotion, map distributivity} \\
 & \Lambda\!\!\Lambda' / \; (\Lambda\!\!\Lambda' / \cdot \tau * \cdot (\nabla y)) * \, x \; .
\end{array}
$$

The function $\Lambda\!\!\Lambda' / \cdot \tau * \cdot (\nabla y)$ is developed as follows.

$$
\begin{array}{ll}
 & \Lambda\!\!\Lambda' / \; \tau * \, (a \nabla y) \\
= & \quad \text{definition of } \nabla \\
 & \Lambda\!\!\Lambda' / \; \tau * \, (a \|) * \, y \\
= & \quad \text{map distributivity, one-point rule, map distributivity} \\
 & \Lambda\!\!\Lambda' / \; (a \|) ** \, \tau * \, y \\
= & \quad \text{promotion theorem (see proviso below)} \\
 & (a \|) * \, \Lambda\!\!\Lambda / \; \tau * \, y \\
= & \quad \text{definition of nub} \\
 & (a \|) * \, \mathsf{nub}(\olessthan, \ogreaterthan) \; y \\
= & \quad \text{definition of } \nabla \\
 & a \nabla \mathsf{nub}(\olessthan, \ogreaterthan) \; y \; ,
\end{array}
$$

provided $(a \|) *$ is $(\Lambda\!\!\Lambda, \Lambda\!\!\Lambda')$–promotable on singletons. This follows from the conditions of the theorem. We substitute the derived equality in the main derivation, and continue the main derivation as follows

$$
\begin{array}{ll}
 & \Lambda\!\!\Lambda' / \; (\Lambda\!\!\Lambda' / \cdot \tau * \cdot (\nabla y)) * \, x \\
= & \quad \text{derived equality} \\
 & \Lambda\!\!\Lambda' / \; (\nabla \mathsf{nub}(\olessthan, \ogreaterthan) \; y) * \, x \\
= & \quad \text{introduction of } \Xi \\
 & \Lambda\!\!\Lambda' / \; ((\Xi \mathsf{nub}(\olessthan, \ogreaterthan) \; y) \cdot \tau) * \, x \; ,
\end{array}
$$

provided Ξ satisfies $[a] \,\Xi\, z = a \nabla z$. In order to make progress, we define Ξ like X by further requiring $(x \mathbin{+\!\!+} y) \,\Xi\, z = (x \,\Xi\, z) \otimes (y \,\Xi\, z)$. At this point, this equation is irrelevant, and so is $\otimes$. But below we shall exploit this freedom. Proceeding with the derivation, we obtain

$$
\begin{array}{ll}
 & /\!\!\!\wedge\!\setminus'/\ ((\Xi\mathsf{nub}(\olessthan, \ogreaterthan)\ y) \cdot \tau)*\ x \\
= & \qquad \text{map distributivity} \\
 & /\!\!\!\wedge\!\setminus'/\ (\Xi\mathsf{nub}(\olessthan, \ogreaterthan)\ y)*\ \tau*\ x \\
= & \qquad \text{promotion theorem (see proviso below)} \\
 & (\Xi\mathsf{nub}(\olessthan, \ogreaterthan)\ y)\ /\!\!\!\wedge\!\setminus/\ \tau*\ x \\
= & \qquad \text{definition of } \mathsf{nub} \\
 & (\mathsf{nub}(\olessthan, \ogreaterthan)\ x) \,\Xi\, (\mathsf{nub}(\olessthan, \ogreaterthan)\ y)\ ,
\end{array}
$$

provided (Ξz) is $(/\!\!\!\wedge\!\setminus, /\!\!\!\wedge\!\setminus')$–promotable on singletons. This condition is satisfied if the implications given in the theorem hold, and if $\otimes$ is chosen to be equal to $/\!\!\!\wedge\!\setminus'$.
(End of Proof)

5 Substree promotion revisited

Nub theory is used to derive a new substree promotion theorem. A predicate p on trees is *buc–closed* (bottom-up components-closed) with derivative p' if and only if for all x, y, and b,

$$p\ (x \swarrow b \searrow y) \equiv (p\ x) \wedge (p\ y) \wedge (p'\ b\ (x, y))\ .$$

We have

Theorem 5 (subs-nub-promotion theorem) *Given a total pre-order $\olessthan$ and a total linear order $\ogreaterthan$, define the order $\olessthan$ & $\ogreaterthan$ by*

$$a(\olessthan \,\&\, \ogreaterthan)b = a \olessthan b \vee (a \circledequal b \wedge a \ogreaterthan b)\ ,$$

where $\circledequal$ is the equivalence relation induced by $\olessthan$. Suppose there exist a total pre-order $\olessthan'$ and a total linear order $\ogreaterthan'$ such that for all x, y, a and b,

$$
\begin{array}{lcl}
x \olessthan y & \Rightarrow & ((a\|x) \olessthan' (a\|y)) \wedge ((x\|a) \olessthan' (y\|a)) \\
x \olessthan' y & \Rightarrow & x \olessthan_{\swarrow b \searrow} y \\
x \ogreaterthan y & \Rightarrow & ((a\|x) \ogreaterthan' (a\|y)) \wedge ((x\|a) \ogreaterthan' (y\|a)) \\
x \ogreaterthan' y & \Rightarrow & x \ogreaterthan_{\swarrow b \searrow} y\ .
\end{array}
$$

Furthermore, suppose the predicate p is buc–closed with derivative p' satisfying

$$p'\ y \wedge x \circledequal' y \wedge x \ogreaterthan' y \Rightarrow p'\ x\ .$$

Then there exists a tree homomorphism $\ominus/ \cdot j*$ such that

$$\uparrow_{\olessthan \& \ogreaterthan}/ \cdot p\triangleleft \cdot \mathsf{subs} = \gg/ \cdot \ominus/ \cdot j*\ .$$

Proof Using equality (1), we have

$$\begin{array}{ll} & \uparrow_{\otimes \& \ominus}/ \cdot p\triangleleft \cdot \mathsf{subs} \\ = & \qquad (1) \\ & \gg/ \cdot \mathsf{nub}(\otimes, \ominus) \cdot p\triangleleft \cdot \mathsf{subs}\,. \end{array}$$

So it suffices to prove that

$$\mathsf{nub}(\otimes, \ominus) \cdot p\triangleleft \cdot \mathsf{subs} = \ominus/ \cdot j* \,.$$

Similar to the derivation of the subs–promotion theorem, we apply the promotion theorem for trees. This theorem is applicable provided

$$\mathsf{nub}\; p\triangleleft\, (\swarrow b \searrow)*\, (\mathsf{subs}\; l) \mathsf{X} (\mathsf{subs}\; r) = (\mathsf{nub}\; p\triangleleft\, \mathsf{subs}\; l) \,\Xi\, (\mathsf{nub}\; p\triangleleft\, \mathsf{subs}\; r)\,,$$

for some operator Ξ. The definition of Ξ is derived as follows.

$$\begin{array}{ll} & \mathsf{nub}\; p\triangleleft\, (\swarrow b \searrow)*\, (\mathsf{subs}\; l) \mathsf{X} (\mathsf{subs}\; r) \\ = & \qquad p \text{ is } \mathit{buc}\text{–closed with derivative } p' \\ & \mathsf{nub}\, (\swarrow b \searrow)*\, (p'\; b)\triangleleft\, (p\triangleleft\, \mathsf{subs}\; l) \mathsf{X} (p\triangleleft\, \mathsf{subs}\; r) \\ = & \qquad \text{map-nub rule (3)} \\ & (\swarrow b \searrow)*\, \mathsf{nub}'\, (p'\; b)\triangleleft\, (p\triangleleft\, \mathsf{subs}\; l) \mathsf{X} (p\triangleleft\, \mathsf{subs}\; r) \\ = & \qquad \text{filter-nub rule (2)} \\ & (\swarrow b \searrow)*\, (p'\; b)\triangleleft\, \mathsf{nub}'\, (p\triangleleft\, \mathsf{subs}\; l) \mathsf{X} (p\triangleleft\, \mathsf{subs}\; r) \\ = & \qquad \text{cross-nub rule (4)} \\ & (\swarrow b \searrow)*\, (p'\; b)\triangleleft\, (\mathsf{nub}\; p\triangleleft\, \mathsf{subs}\; l) \,\Xi\, (\mathsf{nub}\; p\triangleleft\, \mathsf{subs}\; r)\,. \end{array}$$

The rules are applicable by virtue of the conditions of the theorem. The function Ξ is defined as in the cross-nub rule. For completeness' sake we give the definitions of j and $\ominus$.

$$\begin{array}{rll} j\; a = & [\langle a \rangle] & \text{if } p\; \langle a \rangle \\ & 1_{\#} & \text{otherwise} \end{array}$$

$$l \ominus_b r = (j\; b) \mathbin{\wedge\!\!\!\wedge} l \mathbin{\wedge\!\!\!\wedge} r \mathbin{\wedge\!\!\!\wedge} ((\swarrow b \searrow)*\, (p'\; b)\triangleleft\, l \,\Xi\, r)$$

(End of Proof)

6 An application

We apply the subs-nub-promotion theorem to find an efficient algorithm for the largest ascending substree problem specified in Section 3. For that purpose, we have to refine the specification of the problem. Define the orders $\otimes$ and $\otimes \,\&\, \ominus$ by

$$\begin{array}{rcl} a \otimes b & = & a <_{\#} b \\ a(\otimes \,\&\, \ominus)b & = & a \otimes b \vee (a \ominus b \wedge b \ominus a)\,, \end{array}$$

where $\ominus$ is some total order, which has yet to be defined. If we define $\otimes'$ by

$$(a, b) \otimes' (c, d) = (a, b) <_{+\cdot(\#\|\#)} (c, d)\,,$$

it follows that

$$\begin{array}{lcl} x \olessthan y & \Rightarrow & ((a\|x) \olessthan' (a\|y)) \wedge ((x\|a) \olessthan' (y\|a)) \\ x \olessthan' y & \Rightarrow & x \olessthan_{\swarrow b \searrow} y \end{array}$$

We have to find linear orders $\ogreaterthan$ and $\ogreaterthan'$ such that

$$\begin{array}{lcl} x \ogreaterthan y & \Rightarrow & ((a\|x) \ogreaterthan' (a\|y)) \wedge ((x\|a) \ogreaterthan' (y\|a)) \\ x \ogreaterthan' y & \Rightarrow & x \ogreaterthan_{\swarrow b \searrow} y \end{array}$$

This is the most difficult part of the derivation: we have to invent definitions of $\ogreaterthan$ and $\ogreaterthan'$. The following definitions of $\ogreaterthan$ and $\ogreaterthan'$ satisfy the conditions. The proof of this claim is left to the reader.

$$x \ogreaterthan y = x <_{\mathsf{top}} y \vee (x =_{\mathsf{top}} y \wedge x \prec_{\mathsf{children}} y)$$

$$x \ogreaterthan' y = x \prec_{\mathsf{listify}} y$$

$$\mathsf{listify}\ (a, b) = [a, b]$$

$$\begin{array}{lcl} \mathsf{children}\ \langle a \rangle & = & 1_{+\!\!+} \\ \mathsf{children}\ l \swarrow b \searrow r & = & [l, r] \end{array}$$

$$\begin{array}{ll} x \prec y = & x <_{\uparrow/\cdot top*} y \vee \\ & x =_{\uparrow/\cdot top*} y \wedge x <_{L\cdot top*} y \vee \\ & x =_{\uparrow/\cdot top*} y \wedge x =_{L\cdot top*} y \wedge x \prec_{+\!\!+/\cdot \mathsf{children}*} y\ . \end{array}$$

where $<_L$ is the lexicographical ordering defined on lists, and *top* computes the element in the root of a tree. Finally, we have to show that the predicate *ascending* defined by

$$\begin{array}{lcl} ascending\ \langle a \rangle & = & \mathsf{True} \\ ascending\ l \swarrow b \searrow r & = & (ascending\ l) \wedge (ascending\ r) \wedge (b \geq top\ l) \wedge (b \geq top\ r)\ . \end{array}$$

satisfies the condition of the theorem. From the definition of *ascending* it is immediately clear that *ascending* is buc–closed predicate with derivative p' defined by

$$p'\ b\ (l, r) = b \geq top\ l \wedge b \geq top\ r\ .$$

Since from $x \ogreaterthan' y$ it follows that $\uparrow/\ top*\ x \leq \uparrow/\ top*\ y$, and from $p'\ b\ \ y$ it follows that $\uparrow/\ top*\ y \leq b$, we have $\uparrow/\ top*\ x \leq b$, and hence $p'\ b\ x$, so the condition is satisfied.

All the conditions of the **subs-nub-promotion** theorem are satisfied, and we obtain the algorithm

$$\gg/ \cdot \ominus/ \cdot j*\ ,$$

where $j\ a = [\langle a \rangle]$, and the operator $\ominus$ is defined by

$$l \ominus_b r = [\langle b \rangle] \mathbin{\wedge\!\!\wedge} l \mathbin{\wedge\!\!\wedge} r \mathbin{\wedge\!\!\wedge} (\swarrow b \searrow)*\ (p'\ b) \triangleleft (l \,\Xi\, r))\ .$$

Given a tree of size n, this algorithm requires time $O(n^2)$ to compute the largest ascending substree.

7 Conclusions

Nub theory seems to be applicable to many problems where tupling is involved. We have illustrated nub theory by deriving an algorithm for finding the largest ascending substree of a tree. Another problem on trees which can be solved using nub theory is finding the largest treecut the sum of which is at most a given constant C. A derivation of an algorithm for this problem is presented in [Jeu89]. The results derived here and in [dMB89] provide a way to derive this algorithm much simpler. This problem is also interesting because the preorder used is not $<_{\#}$, as in all the examples we have seen until now, but instead $<_{+/}$.

The proofs of the relevant rules of nub theory can be formulated elegantly using promotion. However, more elegant proofs may be obtained if the 'de Bruin-Reynolds-Wadler' theorem is applied. Because of the polymorphic nature of nub, Lemma 2 and Lemma 3 seem to follow easily from this theorem. I do not yet see how to prove the cross-nub rule with the dBRW theorem.

References

[dMB89] O. de Moor and R.S. Bird. Lecture notes on nub theory. Lecture Notes International Summer School on Constructive Algorithmics, Hollum-Ameland, The Netherlands, 1989.

[Jeu89] J. Jeuring. Deriving algorithms on binary labelled trees. In P.M.G. Apers, D. Bosman, and J. van Leeuwen, editors, *Proceedings SION Computing Science in the Netherlands*, pages 229–249, 1989.

Homomorphisms, Factorisation and Promotion

Nico Verwer *
RUU

January 23, 1990

1 Why this note?

Reading Grant Malcolm's paper 'Factoring Homomorphisms', I did not like the notation for type functors very much. I would like to have a more concise notation, to show the essence of the factorisation theorem more clearly. The notation used here is based on the one Lambert Meertens used at the 'wednesday afternoon sessions'. It uses categorical concepts, but little knowledge of category theory is required to read this note.

Using this notation, I found that proving the factorisation theorem is really a simple diagram-chasing exercise. I also found that I had to generalise the notion of reductions (which I also define for snoc-lists) and factorability. It appears that factorability can be seen as a property of homomorphisms, not of type-functors.

Together with Johan Jeuring, I found that the new notation can also be used to express the promotion theorem more clearly. We also made a link with the simple promotion law which is used in squigol.

A remark on the presentation: Some people like to draw diagrams instead of writing down formulas, whereas others maintain that pictures give a false sense of understanding and can not give an exact description. Personally I like diagrams as an illustration of formulas, and I think they can help in understanding formulas. Therefore I shall use them throughout this note, but I shall not rely on them for proofs. The diagrams were typeset with Francis Borceux's macro package [1], which saved me a lot of time and work.

*verwer@cs.ruu.nl

2 Hagino type-functors

We write $A + B$ for the disjoint union of A and B. This is a functor, and we have a corresponding action on functions; for $f : A \to A'$, $g : B \to B'$ we have

$$f + g : A + B \to A' + B'.$$

We also have a 'case-construction'

$$[f, g] : A + B \to C$$

for $f : A \to C, g : B \to C$.

Other functors we often use are the cartesian product $\times$ and constant functors. The functor which constantly yields A when applied to any type, can (like every functor) be applied to functions, and then yields i_A, the identity on A.

A data type definition consists of the *name* of a *type functor*, and *constructors* for the new type, with the types of their *components*. For example, list-formation is indicated by $*$, and the type A^* of lists over A has constructors $\square$ and $\succ\!\!+$, which have component types $\mathtt{1}$ (the one-point set) and $A \times A^*$. This is generalized in the following definition.

Definition 1. A *Hagino type-functor* $\dagger$ is determined by

- Its corresponding *components-functor*, indicated by $\boxed{\dagger}$ ($\dagger$ with a box around it). The type of the components needed to construct a $A^\dagger$-value is $A^\dagger \boxed{\dagger} A$.

- Its *constructors*, given by a polymorphic function

$$\epsilon : \Lambda A \,.\, A^\dagger \boxed{\dagger} A \to A^\dagger.$$

Also, the type $A^\dagger$ is *initial*, i.e. for every type B with a function

$$\phi : B \boxed{\dagger} A \to B$$

there is a unique function

$$\phi^\natural : A^\dagger \to B$$

which is a *homomorphism* (respects the structure):

$$\phi^\natural \circ \epsilon_A = \phi \circ (\phi^\natural \boxed{\dagger} \mathrm{i}_A)$$

□

This is shown in the following commuting diagram.

$$\begin{tikzcd}
A^{\dagger} \boxed{\dagger} A \arrow[r, "\epsilon_A"] \arrow[d, "\phi^{\natural} \boxed{\dagger} \mathrm{i}_A"'] & A^{\dagger} \arrow[d, dotted, "\phi^{\natural}"] \\
B \boxed{\dagger} A \arrow[r, "\phi"'] & B
\end{tikzcd}$$

The dotted arrow indicates that it is the *unique* function which makes the diagram commute.

The constructor ϵ is an isomorphism, with inverse $(\epsilon \boxed{\dagger} \mathrm{i})^{\natural}$ (see for instance [4]). Because of this isomorphism, $A^{\dagger}$ can be seen as a fixed point of $(\boxed{\dagger} A)$:

$$A^{\dagger} \boxed{\dagger} A \;\cong\; A^{\dagger}.$$

For this reason, some people write $A^{\dagger}$ as $(\mu X \,.\, X \boxed{\dagger} A)$. The function

$$(\epsilon_A \boxed{\dagger} \mathrm{i}_A)^{\natural} : A^{\dagger} \rightarrow A^{\dagger} \boxed{\dagger} A$$

splits a $A^{\dagger}$-term in its components. It is a 'pattern-matching' function, which is often used implicitly in functional programming languages to do case-analysis on the construction of a term. This provides a recursive definition of $\phi^{\natural}$ (which is easily established from the commuting diagram):

$$\phi^{\natural} = \phi \circ (\phi^{\natural} \boxed{\dagger} \mathrm{i}_A) \circ (\epsilon_A \boxed{\dagger} \mathrm{i}_A)^{\natural}.$$

Malcolm [2, 3] would write $(\!|\phi|\!)$ instead of $\phi^{\natural}$, and $(\otimes_0, \ldots, \otimes_n)$ for a components-functor $\boxed{\dagger} = \lambda X \lambda A \,.\, (X \otimes_0 A + \ldots + X \otimes_n A)$. He also sometimes writes $(F_1, \ldots, F_n)$ for $(\boxed{\dagger} A)$. Although in many cases the components-type is indeed a disjoint union of types, we think that it is not necessary to indicate this, and we rather have one components-functor.

Example 2. An example of a type-functor is the cons-list constructor $*$, with

$$B \boxed{*} A = \mathbf{1} + (A \times B) \;\;,\;\; g \boxed{*} f = \mathrm{i}_{\mathbf{1}} + (f \times g)$$

$$\epsilon_A = [\square, \succ\!\!+] : A^{*} \boxed{*} A \rightarrow A^{*}.$$

In this case, the commuting diagram says that for all types A, B and functions $[c, \oplus] : \mathbf{1} + (A \times B) \rightarrow B$:

$$[c, \oplus]^{\natural} \circ [\square, \succ\!\!+] = [c, \oplus] \circ (\mathrm{i}_{\mathbf{1}} + (\mathrm{i}_A \times [c, \oplus]^{\natural}))$$

or, equivalently:

$$[c, \oplus]^{\natural} \circ \square = c \;\;,\;\; [c, \oplus]^{\natural} \circ \succ\!\!+ = \oplus \circ (\mathrm{i}_A \times [c, \oplus]^{\natural}).$$

$$
\begin{tikzcd}
A^\dagger \boxed{\dagger} A \arrow[rr, "\epsilon_A"] \arrow[d, "(\epsilon_B \circ (\mathrm{i}_{B^\dagger} \boxed{\dagger} f))^\natural \boxed{\dagger} \mathrm{i}_A"'] & & A^\dagger \arrow[d, dotted, "(\epsilon_B \circ (\mathrm{i}_{B^\dagger} \boxed{\dagger} f))^\natural = f^\dagger"] \\
B^\dagger \boxed{\dagger} A \arrow[r, "\mathrm{i}_{B^\dagger} \boxed{\dagger} f"'] & B^\dagger \boxed{\dagger} B \arrow[r, "\epsilon_B"'] & B^\dagger
\end{tikzcd}
$$

$\square$

This definition is exactly the same as the one in Malcolm's paper [2], who also proves that maps indeed preserve identity and composition.

Proposition 5.

$$f^\dagger \circ \epsilon_A = \epsilon_B \circ (f^\dagger \boxed{\dagger} f).$$

Proof. This corresponds exactly to the commuting diagram. Using the fact that (bi)functors preserve composition,

$$(h \boxed{\dagger} k) \circ (p \boxed{\dagger} q) = (h \circ p) \boxed{\dagger} (k \circ q)$$

the identity laws and the definition of $f^\dagger$, we obtain

$$(\mathrm{i}_{B^\dagger} \boxed{\dagger} f) \;\circ\; ((\epsilon_B \circ (\mathrm{i}_{B^\dagger} \boxed{\dagger} f))^\natural \boxed{\dagger} \mathrm{i}_A) = (\epsilon_B \circ (\mathrm{i}_{B^\dagger} \boxed{\dagger} f))^\natural \boxed{\dagger} f = f^\dagger \boxed{\dagger} f.$$

$\square$

In the case of lists, this proposition amounts to

$$f^\dagger \circ \square = \square \quad , \quad f^\dagger \circ {>\!\!+} = {>\!\!+} \circ (f \times f^\dagger)$$

which is just the usual recursive definition of map on lists.

4 Reductions

On cons-lists, we define reductions $\oplus \nleftarrow_e : A^* \rightarrow A$ as

$$(\oplus \nleftarrow_e)\square = e \quad , \quad (\oplus \nleftarrow_e)(a >\!\!+ x) = a \oplus ((\oplus \nleftarrow_e)x)$$

for $\oplus : A \times A \rightarrow A$ and $e : A$. This is slighty different from the usual definition, where $\oplus : A \times B \rightarrow B$ (see the note below). A reduction is primarily a function on the structure, not on the elements of a $A^\dagger$-term. (One might argue that the function $+ \nleftarrow_0 : N^* \rightarrow N$ does affect the integer elements in the list, but this is really a consequence of equations that hold in the integer domain.)

We can define reductions as homomorphisms, just as we did for maps.

Definition 6. A $A^\dagger$-reduction is defined for functions

$$\phi : A \boxed{\dagger} A \rightarrow A$$

as

$$\phi^{\natural} : A^{\dagger} \to A.$$

This is illustrated by the diagram below.

$$\begin{tikzcd}
A^{\dagger} \boxed{\dagger} A \arrow[r, "\epsilon_A"] \arrow[d, "(\phi^{\natural}) \boxed{\dagger} \mathrm{i}_A"'] & A^{\dagger} \arrow[d, dotted, "\phi^{\natural}"] \\
A \boxed{\dagger} A \arrow[r, "\phi"'] & A
\end{tikzcd}$$

□

Reductions are usually written as $\oplus \nleftarrow_e$ for $\phi = [e, \oplus]$ (for cons-lists), or another notation considered appropriate.

Example 7. On non-empty join-lists, reductions are defined for functions

$$[f, \oplus] : B + (B \times B) \to B.$$

In the special case that $f = \mathrm{i}_B$, we write $\oplus/$ for the reduction $[\mathrm{i}_B, \oplus]^{\natural}$. We shall use this notation for other data types to emphasize the fact that some homomorphism is a reduction. The above diagram gives the recursive definition of $\oplus/$. □

Note that in the above definition we do not require $\dagger$ to be *factorable*, like Malcolm [2] does. Thus reductions over cons- or snoc-lists can be defined in the usual way. For instance, the reduction which is normally written as $\oplus \nleftarrow_e$ is exactly the same as $[e, \oplus]^{\natural}$ In this case, the commuting diagram from the definition above amounts to the definition of $\oplus \nleftarrow_e$ given earlier.

In the literature on constructive functional programming, reductions on cons-lists are often defined for operators $\oplus : A \times B \to B$. Although this is more general, we chose not to do so, because then reductions would be exactly the same as homomorphisms. We feel that reductions like they are defined here can be very useful, because on the elements of a structure $A^{\dagger}$ they act as a function from A to A. In Malcolm's paper, reductions act as the identity function on elements. We had to mention this property explicitly in the join-list example above. Reductions in the sense of Malcolm [2] are also reductions according to our definition.

5 Factorisation

It is well known that homomorphisms on lists can be factored into a map followed by a reduction. In his paper, Malcolm [2] shows that homomorphisms on *factorable* type-functors can be factored this way. His definition of factorable requires $\boxed{\dagger}$ to have a special form, namely

$$X \boxed{\dagger} A = A + X^F$$

where A does not occur in X^F. Also, functions from $A \boxed{\dagger} B$ to B must be of the form

$$[f, g] : A + B^F \rightarrow B.$$

This means, for instance, that join-lists are factorable, but cons-lists are not (reductions are not even defined for cons lists).

In the previous section we defined reductions for *all* type functors. Still it is not possible to factor every homomorphism we can think of into a map followed by a reduce. We define the factorability of homomorphisms as a property of the homomorphisms themselves, not of the type functor on which they are defined.

Definition 8. A homomorphism $\phi^{\natural} : A^{\dagger} \rightarrow B$ is *factorable* if the function $\phi : B \boxed{\dagger} A \rightarrow B$ can be written as

$$\phi = \oplus \circ (\mathrm{i}_B \boxed{\dagger} f)$$

where

$$\oplus : B \boxed{\dagger} B \rightarrow B \quad , \quad f : A \rightarrow B.$$

□

Proposition 9. Homomorphisms on a type functor which is factorable in the sense of Malcolm [2] are factorable in the sense of the previous definition.

Proof. Consider a homomorphism $\phi^{\natural} : A^{\dagger} \rightarrow B$, where $\dagger$ is factorable in Malcolm's sense, i.e. $X \boxed{\dagger} A = A + X^F$. Then by his definition, $\phi = [f, g] : A + B^F \rightarrow B$. Now because

$$\begin{aligned} [f, g] &= [\mathrm{i}_B, g] \circ (f + \mathrm{i}_{B^F}) \\ &= [\mathrm{i}_B, g] \circ (\mathrm{i}_B \boxed{\dagger} f) \end{aligned}$$

ϕ is factorable according to our definition, as shown in the diagram.

$$\begin{tikzcd} B \boxed{\dagger} A = A + B^F \arrow[rr, "{[f,g]}"] \arrow[dr, "{\mathrm{i}_B \boxed{\dagger} f = f + \mathrm{i}_{B^F}}"'] & & B \\ & B \boxed{\dagger} B = B + B^F \arrow[ur, "{[\mathrm{i}_B, g]}"'] & \end{tikzcd}$$

□

The inverse of $[\square, >\!\!\!+]$ is $([\square, >\!\!\!+] \boxed{\ast} \mathrm{i}_A)^\natural$ which splits up a list in its head and tail:

$$([\square, >\!\!\!+] \boxed{\ast} \mathrm{i}_A)^\natural \circ \square = \mathrm{i}_1 \quad , \quad ([\square, >\!\!\!+] \boxed{\ast} \mathrm{i}_A)^\natural \circ >\!\!\!+ = \mathrm{i}_A \times \mathrm{i}_{A^{\ast}}.$$

$\square$

Example 3. Another type-functor is the non-empty join-list constructor $\ast$, with

$$B \boxed{\ast} A = A + (B \times B) \quad , \quad g \boxed{\ast} f = f + (g \times g)$$

$$\epsilon_A = [[\cdot], +\!\!\!+].$$

The reader is encouraged to draw the corresponding diagram, and investigate its meaning. (We have not required $+\!\!\!+$ to be associative, so we really have specified binary trees.) $\square$

3 Maps

A map is the part of a type-functor that works on functions. In general we have, for a type-functor $\dagger$ and a function $f : A \to B$, a mapped function:

$$f^\dagger : A^\dagger \to B^\dagger.$$

Functors preserve identity and composition:

$$\mathrm{i}_A{}^\dagger = \mathrm{i}_{A^\dagger} \quad , \quad (g \circ f)^\dagger = g^\dagger \circ f^\dagger.$$

The idea is that a mapped function only works on the A-elements of a $A^\dagger$-term, leaving the structure unchanged.

We can define maps as homomorphisms. In order to do so, we try to find a function

$$\phi : B^\dagger \boxed{\dagger} A \to B^\dagger$$

such that

$$\phi^\natural = f^\dagger.$$

We can do this by first applying f to the A-elements of the $(B^\dagger \boxed{\dagger} A)$-term, giving a term in $B^\dagger \boxed{\dagger} B$. Then we embed this in a $B^\dagger$-structure by applying the constructors ϵ_B.

Definition 4. The map corresponding to a type-functor $\dagger$ is defined for all functions $f : A \to B$ as

$$f^\dagger = (\epsilon_B \circ (\mathrm{i}_{B^\dagger} \boxed{\dagger} f))^\natural.$$

This is illustrated in the following commuting diagram.

We can now formulate the factorisation theorem, which says that factorable homomorphisms can be factored into a map followed by a reduction:

Proposition 10. If $\phi^\natural$ is factorable and $\phi = \oplus \circ (\mathrm{i}_B \boxed{\dagger} f)$, where $\oplus : B \boxed{\dagger} B \rightarrow B$ and $f : A \rightarrow B$, then

$$\phi^\natural = (\oplus/) \circ f^\dagger.$$

Proof. We first prove:

$$\begin{array}{lcl}
\oplus/ \circ f^\dagger \circ \epsilon_A & = & \text{(map diagram)} \\
\oplus/ \circ \epsilon_B \circ (\mathrm{i}_{B^\dagger} \boxed{\dagger} f) \circ (f^\dagger \boxed{\dagger} \mathrm{i}_A) & = & \text{(reduction diagram)} \\
\oplus \circ (\oplus/ \boxed{\dagger} \mathrm{i}_B) \circ (\mathrm{i}_{B^\dagger} \boxed{\dagger} f) \circ (f^\dagger \boxed{\dagger} \mathrm{i}_A) & = & \text{(functors preserve composition,} \\
 & & \text{identity laws)} \\
\oplus \circ (\mathrm{i}_B \boxed{\dagger} f) \circ (\oplus/ \boxed{\dagger} \mathrm{i}_A) \circ (f^\dagger \boxed{\dagger} \mathrm{i}_A) & = & \text{(functors preserve composition,} \\
 & & \text{definition of } \phi) \\
\phi \circ ((\oplus/ \circ f^\dagger) \boxed{\dagger} \mathrm{i}_A). & &
\end{array}$$

By definition, we also know that $\phi^\natural = (\oplus \circ (\mathrm{i}_B \boxed{\dagger} f))^\natural$ is the *unique* function which satisfies

$$\phi^\natural \circ \epsilon_A = \phi \circ (\phi^\natural \boxed{\dagger} \mathrm{i}_A).$$

Since $\oplus/ \circ f^\dagger$ has the same property, we conclude that

$$\phi^\natural = \oplus/ \circ f^\dagger.$$

$$\begin{tikzcd}
A^\dagger \boxed{\dagger} A \arrow[rr, "\epsilon_A"] \arrow[d, "f^\dagger \boxed{\dagger} \mathrm{i}_A"'] \arrow[drr, phantom, "\text{map}"] & & A^\dagger \arrow[d, "f^\dagger"] \arrow[dd, bend left=60, "\phi^\natural"] \\
B^\dagger \boxed{\dagger} A \arrow[r, "\mathrm{i}_{B^\dagger} \boxed{\dagger} f"] \arrow[d, "\oplus/ \boxed{\dagger} \mathrm{i}_A"'] & B^\dagger \boxed{\dagger} B \arrow[r, "\epsilon_B"] \arrow[d, "\oplus/ \boxed{\dagger} \mathrm{i}_B"'] \arrow[dr, phantom, "\text{reduction}"] & B^\dagger \arrow[d, "\oplus/"] \\
B \boxed{\dagger} A \arrow[r, "\mathrm{i}_B \boxed{\dagger} f"'] & B \boxed{\dagger} B \arrow[r, "\oplus"'] & B
\end{tikzcd}$$

□

6 Promotion

A very important theorem is the promotion theorem given by Malcolm in [3]. In our notation it reads:

Proposition 11. Let $\phi : B \boxed{\dagger} A \to B$, $\psi : C \boxed{\dagger} A \to C$ and $f : B \to C$. If

$$f \circ \phi = \psi \circ (f \boxed{\dagger} \mathrm{i}_A)$$

(f is $\phi \to \psi$-promotable), then

$$\psi^\natural = f \circ \phi^\natural.$$

Proof.

$$\begin{array}{lcl} (f \circ \phi^\natural) \circ \epsilon_A & = & \text{(def. of } \phi^\natural) \\ f \circ \phi \circ (\phi^\natural \boxed{\dagger} \mathrm{i}_A) & = & \text{(promotability-assumption)} \\ \psi \circ (f \boxed{\dagger} \mathrm{i}_A) \circ (\phi^\natural \boxed{\dagger} \mathrm{i}_A) & = & \text{(functors preserve composition)} \\ \psi \circ ((f \circ \phi^\natural) \boxed{\dagger} \mathrm{i}_A) & & \end{array}$$

Since $\psi^\natural$ is the unique function with the property

$$\psi^\natural \circ \epsilon_A = \psi \circ (\psi^\natural \boxed{\dagger} \mathrm{i}_A)$$

we conclude that

$$\psi^\natural = f \circ \phi^\natural.$$

$$\begin{tikzcd}
A^\dagger \boxed{\dagger} A \arrow[r, "\epsilon_A"] \arrow[d, "\phi^\natural \boxed{\dagger} \mathrm{i}_A"'] & A^\dagger \arrow[d, "\phi^\natural"'] \arrow[dd, bend left=60, "\psi^\natural"] \\
B \boxed{\dagger} A \arrow[r, "\phi"] \arrow[d, "f \boxed{\dagger} \mathrm{i}_A"'] & B \arrow[d, "f"'] \\
C \boxed{\dagger} A \arrow[r, "\psi"'] & C
\end{tikzcd}$$

□

A special case arises when $\phi = \oplus : A \boxed{\dagger} A \to A$, (then $\phi^\natural$ is a reduction), and $\psi^\natural$ is factorable as

$$\psi = \otimes \circ (\mathrm{i}_C \boxed{\dagger} f).$$

Then the promotion theorem becomes:

Proposition 12. If

$$f \circ \oplus = \otimes \circ (f \boxed{\dagger} f)$$

(f is $\oplus \to \otimes$-promotable), then

$$\otimes / \circ f^\dagger = f \circ \oplus /.$$

Proof. Because of the simplifying assumptions, $\phi^\natural$ may be written as $\oplus/$, and by the factorisation theorem $\psi^\natural = \otimes/\circ f^\dagger$. Substituting this in the general promotion theorem then gives the special one. This is illustrated in the figure below.

$$\begin{tikzcd}
A^\dagger \boxed{\dagger} A \arrow[rr, "\epsilon_A"] \arrow[d, "\oplus/ \boxed{\dagger} \mathrm{i}_A"'] & & A^\dagger \arrow[d, "\oplus/"'] \arrow[dd, bend left, "\otimes/\circ f^\dagger"] \\
A \boxed{\dagger} A \arrow[rr, "\oplus"] \arrow[d, "f \boxed{\dagger} \mathrm{i}_A"'] \arrow[dr, "f \boxed{\dagger} f"'] & & A \arrow[d, "f"'] \\
B \boxed{\dagger} A \arrow[r, "\mathrm{i}_B \boxed{\dagger} f"'] & B \boxed{\dagger} B \arrow[r, "\otimes"'] & B
\end{tikzcd}$$

□

Example 13. In the case of non-empty join-lists, the last proposition is the well-known law for list-promotion. If we substitute $[\mathrm{i}_A, \oplus]$ for $\oplus$, and $[\mathrm{i}_B, \otimes]$ for $\otimes$, the promotability-condition becomes

$$f \circ \oplus = \otimes \circ (f \times f)$$

and we then have

$$f \circ \oplus / = \otimes/ \circ f^*$$

where $\oplus/$, $\otimes/$ are defined as in the earlier example. □

References

[1] Francis Borceux, *User's guide for the diagram macro's*, UCL, Louvain-la-Neuve, Belgium. (this macro package can be obtained via FTP from `praxis.cs.ruu.nl`, 131.211.80.6.)

[2] Grant Malcolm, *Factoring Homomorphisms.*

[3] Grant Malcolm, *Homomorphisms and Promotability.*

[4] G.C. Wraith, *A note on categorical data types*, in *Category theory and computer science 1989*, LNCS 389.