

## The linear algebra of UTP

**Bernhard Möller**

### **Angaben zur Veröffentlichung / Publication details:**

Möller, Bernhard. 2005. "The linear algebra of UTP." Augsburg: Institut für Informatik, Universität Augsburg.

### **Nutzungsbedingungen / Terms of use:**

**licgercopyright**

Dieses Dokument wird unter folgenden Bedingungen zur Verfügung gestellt: / This document is made available under the following conditions:

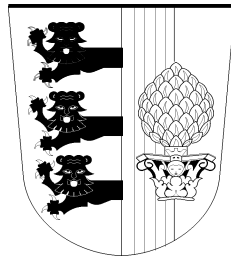
**Deutsches Urheberrecht**

Weitere Informationen finden Sie unter: / For more information see:

<https://www.uni-augsburg.de/de/organisation/bibliothek/publizieren-zitieren-archivieren/publizieren/>



UNIVERSITÄT AUGSBURG

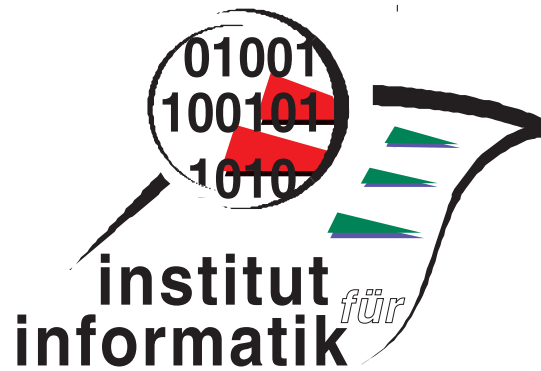


## The Linear Algebra of UTP

Bernhard Möller

Report 2005-14

September 2005



INSTITUT FÜR INFORMATIK  
D-86135 AUGSBURG

Copyright © Bernhard Möller  
Institut für Informatik  
Universität Augsburg  
D-86135 Augsburg, Germany  
<http://www.Informatik.Uni-Augsburg.DE>  
— all rights reserved —

# The Linear Algebra of UTP

Bernhard Möller

Institut für Informatik, Universität Augsburg, D-86135 Augsburg, Germany  
moeller@informatik.uni-augsburg.de

**Abstract.** We show that the well-known algebra of matrices over semirings can be used to reason conveniently about predicates and designs as used in the Unifying Theories of Programming of Hoare and He.

## 1 A Matrix View of UTP

The Unifying Theories of Programming (UTP) developed in [1] model the termination behaviour of programs using two special variables  $ok$  and  $ok'$  that express whether a program has been started and has terminated, respectively. Programs are identified with predicates relating the initial values  $v$  of variables with their final values  $v'$ ; moreover,  $ok$  and  $ok'$  may occur freely in predicates.

However, the set of all such predicates is too general for a number of reasons not to be discussed here. Therefore, Hoare and He introduce a special class of predicates, called *designs*, of the form

$$P \vdash Q \stackrel{\text{def}}{\Leftrightarrow} ok \wedge P \Rightarrow ok' \wedge Q ,$$

where  $ok$  and  $ok'$  are not allowed to occur in  $P$  or  $Q$ . The informal meaning is: if the design has been started and satisfies the precondition satisfying  $P$  it will eventually terminate and satisfy the postcondition  $Q$ .

The aim of the present paper is to present a calculational more workable form of the theory of predicates and designs that does no longer mention the “unobservable” variables  $ok$  and  $ok'$ ; in fact it is even completely variable-free and hence, in particular, does not need to work with substitutions.

Because of the special role of  $ok$  and  $ok'$  we can first look at the possible values of these two variables and for each combination obtain a residual predicate depending only on the proper program variables. To emphasize this view we use the notation  $R(ok, ok')$ . The basic idea of our calculus is to record the residual predicates defined by  $R$  in a  $2 \times 2$ -matrix. The rows are indexed by the values of  $ok$  and the columns by those of  $ok'$ ; the entries are predicates in which  $ok$  and  $ok'$  do not occur.

This may seem as a complication at first. But let us look at sequential composition of predicates, defined as

$$R ; S \stackrel{\text{def}}{\Leftrightarrow} \exists ok_0, v_0 : R[ok_0, v_0/ok', v'] \wedge S[ok_0, v_0/ok, v] ,$$

where  $v$  stands for the list of all proper program variables. In the matrix view this works out to

$$(R; S)(ok, ok') \stackrel{\text{def}}{\Leftrightarrow} \exists ok_0 : \exists v_0 : R(ok, ok_0)[v_0/v'] \wedge S(ok_0, ok')[v_0/v] \\ \Leftrightarrow \exists ok_0 : R(ok, ok_0); S(ok_0, ok') .$$

Now, as in graph algorithms such as Warshall's, we can view  $\exists ok_0$  as summation over all possible values of  $ok_0$  and  $\wedge$  as multiplication. With this interpretation the above formula gives just the entries for the product of the matrices  $R$  and  $S$ , i.e.,  $R; S = R \cdot S$ .

The advantage of this view is that composition can now be treated in a completely component-free manner and existential quantification and substitution disappear. Moreover, the pseudo-variables  $ok$  and  $ok'$  need no longer be mentioned explicitly at all.

Moreover, also the other Boolean operations are supported by the matrix algebra: negation, conjunction and disjunction all are defined componentwise. With the usual definition

$$R \Rightarrow S \stackrel{\text{def}}{\Leftrightarrow} R \vee S = S$$

we see that also implication works componentwise.

Let us now see what designs look like in this view. The first and second matrix rows contain the entries for  $ok = false$  and  $ok = true$ , respectively, and the analogous order is used for the rows. Then

$$P \vdash Q = \begin{pmatrix} true & true \\ \overline{P} & \overline{P} \vee Q \end{pmatrix} . \quad (1)$$

If for some reason we want to talk about  $ok$  and  $ok'$  explicitly, we can represent them as

$$ok = \begin{pmatrix} false & false \\ true & true \end{pmatrix}, \quad ok' = \begin{pmatrix} false & true \\ false & true \end{pmatrix},$$

while a predicate  $P$  not depending on  $ok$  and  $ok'$  corresponds to the constant matrix

$$\begin{pmatrix} P & P \\ P & P \end{pmatrix} .$$

As a first calculation, let us derive the above representation of designs:

$$\begin{aligned} & ok \wedge P \Rightarrow ok' \wedge Q \\ = & \overline{ok \vee \overline{P}} \vee (ok' \wedge Q) \\ = & \begin{pmatrix} false & false \\ true & true \end{pmatrix} \vee \begin{pmatrix} P & P \\ P & P \end{pmatrix} \vee \left( \begin{pmatrix} false & true \\ false & true \end{pmatrix} \wedge \begin{pmatrix} Q & Q \\ Q & Q \end{pmatrix} \right) \\ = & \begin{pmatrix} true & true \\ false & false \end{pmatrix} \vee \begin{pmatrix} \overline{P} & \overline{P} \\ \overline{P} & \overline{P} \end{pmatrix} \vee \left( \begin{pmatrix} false & true \\ false & true \end{pmatrix} \wedge \begin{pmatrix} Q & Q \\ Q & Q \end{pmatrix} \right) \\ = & \begin{pmatrix} true & true \\ \overline{P} & \overline{P} \end{pmatrix} \vee \begin{pmatrix} false & Q \\ false & Q \end{pmatrix} \\ = & \begin{pmatrix} true & true \\ \overline{P} & \overline{P} \vee Q \end{pmatrix} \end{aligned}$$

We defer further calculations till we obtain a more compact notation in the next section.

## 2 Abstracting to Semirings

Again, as in certain graph algorithms, it is useful to base the treatment not on the concrete model of matrices over predicates but on matrices over semirings. Semirings provide the basic operations of choice and sequential composition under the notations  $+$  and  $\cdot$  as well as a basic set of algebraic laws for these.

A *semiring* is a structure  $(S, +, 0, \cdot, 1)$  such that

- $(S, +, 0)$  is a commutative monoid,
- $(S, \cdot, 1)$  is a monoid,
- operation  $\cdot$  distributes over  $+$  in both arguments
- and  $0$  is a left and right annihilator, i.e.,  $0 \cdot x = 0 = x \cdot 0$ .

A semiring is *idempotent* if  $+$  is idempotent, i.e.,  $x + x = x$ . In this case the relation  $a \leq b \stackrel{\text{def}}{\iff} a + b = b$  is a partial order, called the *natural order* on  $S$ . It has  $0$  as its least element. Moreover,  $+$  and  $\cdot$  are isotone w.r.t.  $\leq$  and  $x + y$  is the least upper bound or join of  $x$  and  $y$  w.r.t.  $\leq$ .

An idempotent semiring is *Boolean* if it also has a greatest lower-bound or meet operation  $\sqcap$ , such that  $+$  and  $\sqcap$  distribute over each other, and an operation  $\bar{\phantom{x}}$  that satisfies de Morgan's laws as well as  $x \sqcap \bar{x} = 0$  and  $x + \bar{x} = \top$  where  $\top = \bar{0}$  is the greatest element. In other words, a Boolean semiring is a Boolean algebra with a sequential composition operation. To save parentheses we use the convention that  $\sqcap$  binds tighter than  $+$  but equally tight as  $\cdot$  does.

In the previous section we have already used the Boolean semiring of predicates with  $;$  as composition. Another important semiring is  $\text{REL}(M)$ , the algebra of binary relations under union and composition over a set  $M$ , of which the predicates form a special instance.

Many other examples exist but will not be used here except for the matrix semiring. Let  $(S, +, 0, \cdot, 1)$  be a semiring and  $M$  be a finite set. Then the set  $S^{M \times M}$  of functions from  $M \times M$  to  $S$  can be viewed as the set of  $|M| \times |M|$  matrices with indices in  $M$  and elements in  $S$ . Consider the structure  $\text{MAT}(M, S) = (S^{M \times M}, +, \mathbf{0}, \cdot, \mathbf{1})$  where  $+$  and  $\cdot$  are the usual operations of matrix addition and multiplication, and  $\mathbf{0}$  and  $\mathbf{1}$  are the zero and unit matrices. Then  $\text{MAT}(M, S)$  again forms a semiring, the *matrix semiring* over  $M$  and  $S$ .  $\text{MAT}(M, S)$  is idempotent if  $S$  is. In this case, the natural order is the componentwise order. If  $S$  is a Boolean semiring, so is  $\text{MAT}(M, S)$ , with componentwise meet.

Taking  $S$  to be the two-element Boolean semiring of truth values yields the usual Boolean matrix representation of  $\text{REL}(M)$  as  $\text{MAT}(M, S)$  in terms of adjacency matrices.

For abstractly representing predicates that depend on two Boolean variables  $ok$  and  $ok'$  we use elements from a Boolean semiring  $S$  as matrix entries and the elements  $0$  and  $\top$  as indices, standing for *false* and *true*, respectively. The

element 1 represents the predicate  $\text{skip} \stackrel{\text{def}}{\Leftrightarrow} v = v'$ . We will use the identifiers *false*, *skip* and *true* instead of 0, 1 and  $\top$  when appropriate.

Finally we note that

$$\top \cdot \top = \top. \quad (2)$$

The direction ( $\leq$ ) is trivial, since  $\top$  is the greatest element. The converse direction follows by neutrality and isotonicity:

$$\top = \top \cdot 1 \leq \top \cdot \top.$$

### 3 The Algebra of Designs

Generalizing the definition in (1) we set for elements  $a, b \in S$  of a Boolean semiring  $S$

$$a \vdash b = \begin{pmatrix} \top & \top \\ \bar{a} & \bar{a} + b \end{pmatrix}. \quad (3)$$

We want to calculate the behaviour of designs under  $+$  and  $\cdot$ .

First,

$$\begin{aligned} & (a \vdash b) + (c \vdash d) \\ &= \begin{pmatrix} \top & \top \\ \bar{a} & \bar{a} + b \end{pmatrix} + \begin{pmatrix} \top & \top \\ \bar{c} & \bar{c} + d \end{pmatrix} \\ &= \begin{pmatrix} \top & \top \\ \bar{a} + \bar{c} & \bar{a} + b + \bar{c} + d \end{pmatrix} \\ &= \begin{pmatrix} \top & \top \\ (a \sqcap c) & (a \sqcap c) + b + d \end{pmatrix} \\ &= (a \sqcap c) \vdash (b + d). \end{aligned}$$

In particular, within the set of designs  $\top \vdash 0 = \overline{ok}$  is a neutral element w.r.t.  $+$ . Moreover, we obtain

$$(a \vdash b) \leq (c \vdash d) \Leftrightarrow (a \vdash b) + (c \vdash d) = (c \vdash d) \Leftrightarrow c \leq a \wedge b \sqcap c \leq d$$

and

$$(a \vdash b) = (c \vdash d) \Leftrightarrow a = c \wedge \bar{a} + b = \bar{c} + d \Leftrightarrow a = c \wedge a \sqcap b = c \sqcap d. \quad (4)$$

For composition we obtain, using (2),

$$\begin{aligned} & (a \vdash b); (c \vdash d) \\ &= \begin{pmatrix} \top & \top \\ \bar{a} & \bar{a} + b \end{pmatrix} \cdot \begin{pmatrix} \top & \top \\ \bar{c} & \bar{c} + d \end{pmatrix} \\ &= \begin{pmatrix} \top \cdot \top + \top \cdot \bar{c} & \top \cdot \top + \top \cdot (\bar{c} + d) \\ \bar{a} \cdot \top + (\bar{a} + b) \cdot \bar{c} & \bar{a} \cdot \top + (\bar{a} + b) \cdot (\bar{c} + d) \end{pmatrix} \\ &= \begin{pmatrix} \top & \top \\ \bar{a} \cdot \top + \bar{a} \cdot \bar{c} + b \cdot \bar{c} & \bar{a} \cdot \top + \bar{a} \cdot (\bar{c} + d) + b \cdot \bar{c} + b \cdot d \end{pmatrix} \\ &= \begin{pmatrix} \top & \top \\ \bar{a} \cdot \top + b \cdot \bar{c} & \bar{a} \cdot \top + b \cdot \bar{c} + b \cdot d \end{pmatrix} \end{aligned}$$

$$\begin{aligned}
&= \left( \begin{array}{c} \top \\ \overline{\overline{a \cdot \top \sqcap b \cdot \overline{c}}} \quad \overline{\overline{a \cdot \top \sqcap b \cdot \overline{c}} + b \cdot d} \end{array} \right) \\
&= \overline{\overline{a \cdot \top \sqcap b \cdot \overline{c}}} \vdash (b \cdot d).
\end{aligned}$$

In particular, within the set of designs both

$$\overline{ok} = \top \vdash 0 \text{ and } true$$

are left zeros and

$$\mathbb{I} \stackrel{\text{def}}{=} \top \vdash 1$$

is a left-neutral element w.r.t. composition.

Let us finally look at the conditional. For elements  $a, p, b \in S$  of a Boolean semiring  $S$  we define

$$a \triangleleft p \triangleright b \stackrel{\text{def}}{=} p \sqcap a + \overline{p} \sqcap b.$$

UTP restricts  $p$  to be a condition, i.e., a predicate that does not depend on output variables. We will discuss in a later section how this class of predicates can be characterized abstractly. The law we are going to derive for conditional choice between designs does not depend on  $p$  being a condition. We will use the following easy consequence of the definition:

$$\overline{(a \triangleleft p \triangleright b)} = \overline{a} \triangleleft p \triangleright \overline{b}. \quad (5)$$

To ease notation we write just  $p$  instead of the constant matrix  $\begin{pmatrix} p & p \\ p & p \end{pmatrix}$ .

Then we calculate

$$\begin{aligned}
&(a \vdash b) \triangleleft p \triangleright c \vdash d \\
&= \begin{pmatrix} p & p \\ p & p \end{pmatrix} \sqcap \begin{pmatrix} \top & \top \\ \overline{a} & \overline{a+b} \end{pmatrix} + \begin{pmatrix} \overline{p} & \overline{p} \\ \overline{p} & \overline{p} \end{pmatrix} \sqcap \begin{pmatrix} \top & \top \\ \overline{c} & \overline{c+d} \end{pmatrix} \\
&= \begin{pmatrix} p & p \\ p \sqcap \overline{a} & p \sqcap (\overline{a+b}) \end{pmatrix} + \begin{pmatrix} \overline{p} & \overline{p} \\ \overline{p} \sqcap \overline{c} & \overline{p} \sqcap (\overline{c+d}) \end{pmatrix} \\
&= \begin{pmatrix} \top & \top \\ \overline{a \triangleleft p \triangleright \overline{c}} & \overline{a \triangleleft p \triangleright \overline{c}} + b \triangleleft p \triangleright d \end{pmatrix} \\
&= \begin{pmatrix} \top & \top \\ \overline{a \triangleleft p \triangleright \overline{c}} & \overline{a \triangleleft p \triangleright \overline{c}} + b \triangleleft p \triangleright d \end{pmatrix} \\
&= (a \triangleleft p \triangleright c) \vdash (b \triangleleft p \triangleright d)
\end{aligned}$$

## 4 Healthiness Conditions

In [1] the set of all predicates is classified according to certain *healthiness conditions*. In matrix terminology, designs are characterized by two properties:

(H1) The first row must be constantly  $\top$ .

(H2) Both rows must be increasing w.r.t.  $\leq$ . Clearly every design in the sense of (3) satisfies (H1) and (H2). Conversely, if  $a \leq b$  then

$$\begin{pmatrix} \top & \top \\ a & b \end{pmatrix} = \begin{pmatrix} \top & \top \\ a & a+b \end{pmatrix} = \overline{a} \vdash b.$$

It is straightforward to see that Matrix  $A$  satisfies (H1) iff

$$A = A + \overline{ok} = A + \begin{pmatrix} \top & \top \\ \top & \top + 0 \end{pmatrix} .$$

This type of characterization by a fixpoint property is particularly useful if the underlying Boolean semiring (and hence the matrix semiring over it) is even a complete lattice, since Tarski's fixpoint theorem then implies that the set of all (H1) predicates forms a complete sublattice.

Next we show how the fixpoint characterization of (H2) given in Example 4.1.21(1) of [1] can be derived in a systematic way in our matrix calculus. First we observe that

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \text{ satisfies (H2)} \Leftrightarrow a + b = b \wedge c + d = d \Leftrightarrow \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a & a + b \\ c & c + d \end{pmatrix} .$$

So if we manage to generate the latter matrix from the original one by an isotone function defined in terms of the algebra we are done.

In linear algebra this type of transformation is known as a *shearing* and can be described by the multiplication

$$\begin{pmatrix} a & a + b \\ c & c + d \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} .$$

The shearing matrix can be decomposed as follows:

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \top & \top \\ 0 & \top \end{pmatrix} \sqcap \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = (\top \vdash \top) \sqcap 1 .$$

Therefore we have the following result.

**Lemma 4.1** *A satisfies (H2) iff  $A = A \cdot B$  where*

$$B = (\text{true} \vdash \text{true}) \sqcap \text{skip} .$$

This is indeed a fixpoint characterization with isotone generating function, and so the set of all (H2)-matrices forms a complete lattice (provided the underlying semiring  $S$  is complete).

The further healthiness conditions (H3) and (H4) serve to characterize the designs for which  $\top \vdash 0$  and  $\top \vdash 1$  are also a right zero and a right-neutral element w.r.t.  $\cdot$ , respectively. They are directly given as algebraic conditions:

(H3)  $A ; \mathbb{I} = A$ .

(H4)  $A ; \text{true} = \text{true}$ .

We only work these out for the case where  $A$  is a design. Here it is easier to work directly with the matrices than going through the composition formula for designs.

First,

$$(a \vdash b) ; \mathbb{I} = \begin{pmatrix} \top & \top \\ \bar{a} & \bar{a} + b \end{pmatrix} \cdot \begin{pmatrix} \top & \top \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} \top & \top \\ \bar{a} \cdot \top & \bar{a} \cdot \top + \bar{a} + b \end{pmatrix} ,$$

so that

$$a \vdash b \text{ satisfies (H3)} \Leftrightarrow \bar{a} \cdot \top = \bar{a} \Leftrightarrow \bar{a} \cdot \top \leq \bar{a} .$$

This means that  $\bar{a}$  has to be a *right ideal* (in UTP also known as a *condition*). In the semiring REL of relations this is equivalent to  $a$  itself being a right ideal, since by Schröder's law

$$\bar{a} \cdot \top \leq \bar{a} \Leftrightarrow a \cdot \top \leq a \Leftrightarrow a \cdot \top \leq a .$$

In general semirings this need not be the case.

Second,

$$(a \vdash b); true = \begin{pmatrix} \top & \top \\ \bar{a} & \bar{a} + b \end{pmatrix} \cdot \begin{pmatrix} \top & \top \\ \top & \top \end{pmatrix} = \begin{pmatrix} \top & \top \\ (\bar{a} + b) \cdot \top & (\bar{a} + b) \cdot \top \end{pmatrix} ,$$

so that

$$a \vdash b \text{ satisfies (H4)} \Leftrightarrow (\bar{a} + b) \cdot \top = \top .$$

## 5 Conclusion and Outlook

It seems that the matrix calculus is a convenient vehicle for reasoning about general predicates as well as designs. It remains to be seen whether a similar approach can be followed when further observation variables are added.

**Acknowledgements:** I am grateful to W. Guttmann and P. Höfner for helpful discussions and remarks.

## References

1. C.A.R. Hoare, J. He: Unifying theories of programming. Prentice Hall 1998