Pair Algebras and Galois Connections

Roland Backhouse
Department of Mathematics and Computing Science,
Eindhoven University of Technology,
P.O. Box 513,
5600 MB Eindhoven,
The Netherlands.

June 11, 1998

Abstract

Most studies of Galois connections begin with a *function* and ask the question: when is there a second function that is connected to the first? In possibly the very first application of Galois connections directly related to the modern digital computer, Hartmanis and Stearns posed a subtly different question, namely: when does a *relation* define two functions that are Galois connected? Such a relation they called a "pair algebra". In this paper we derive a general, necessary and sufficient condition for a relation between complete posets to define a Galois connection between the posets. We also give several examples of pair algebras illustrating why this seemingly forgotten notion is relevant to the science of computing.

1 Introduction

Over a period of many years it has become clear that Galois connections play an important role in the science of computing. The name "Galois connection" is derived, of course, from Évariste Galois' analysis in 1832 of necessary and sufficient conditions for a polynomial equation to be solvable by radicals. The *Galois correspondence* is a correspondence between field extensions and groups (see, for example, [16]). The general notion of a Galois *connection* was introduced by Oystein Ore in 1944 [15].

Since Ore's introduction of the general notion, Galois connections have been used in many contexts, although often without specific reference to the notion. Examples include Lambek's analysis of sentence structure [12], Conway's "factors" [2] in the context of regular algebra, Hoare and He's "weakest prespecifications" [10] and Feijen's discussion of the properties of maxima [5]. Since the eighties, however, the notion has become part of the everyday vocabulary of many computing scientists and its use is becoming more explicit.

In the field of abstract interpretation of computer programs Cousot and Cousot [3] have done a great deal towards making the notion widely known. Other examples are [14], [13], [4], [11], and [17].

The very first time that the notion of a Galois connection was applied to a problem directly related to the modern digital computer is possibly Hartmanis and Stearns' analysis of the state assignment problem first published in 1964 [7]. At the time of their original work, Hartmanis and Stearns were unaware of the notion of a Galois connection. They introduced the notion of a "pair algebra" and then, without explicitly referencing the notion of a Galois connection, showed that every pair algebra defines a Galois connection. In addition, they discovered for themselves many of the abstract properties of Galois connections. Some time later, however, they had become aware of the notion — a cryptic bibliographical note at the end of the chapter on "pair algebras" in their book [8] states: "For related mathematical concepts, see the discussion of Galois connections between partially ordered sets in [3]", the reference "[3]" being to Birkhoff's text on Lattice Theory [1].

In spite of the fact that many of the properties of "pair algebras" presented by Hartmanis and Stearns were rediscoveries of known properties of Galois connections, the notion of "pair algebra" did involve a novel element which, to this day, seems to have been ignored in the literature on Galois connections. The purpose of this paper is to explicate that novel element, and point out several examples.

A Galois connection relates two functions. Typically, therefore, studies of Galois connections begin with a function and ask the question: when is there a second function that is connected to the first? The novelty in Hartmanis and Stearns' work is that they did not begin with a function, but with a relation. The question they asked is: when does a relation define two functions that are Galois connected? Such a relation they called a "pair algebra".

Hartmanis and Stearns limited their analysis to finite, complete posets. Moreover, their analysis was less general than is strictly possible. In this paper we derive a general, necessary and sufficient condition for a relation between complete posets to define a Galois connection between the posets. Several elementary and more advanced examples conclude the paper.

2 Existence Theorem

This section includes the main technical results of the paper. We begin with a precise statement of the question we want to explore. The viewpoint is different from Hartmanis and Stearns' since we assume that the importance of being able to identify a Galois connection is already well understood. The main theorem is theorem 9 in which we effectively derive the general definition of a "pair algebra". Subsequently this theorem is specialised to the problem of deriving a necessary and sufficient condition for a function to be an "adjoint" function in a Galois connection. (The latter condition is, of course, well-known. Our purpose is just to show how this theorem is a simple consequence of the pair algebra theorem.)

Another application is the (equally well-known) theorem that a poset is complete if and only if it is cocomplete.

For completeness we include all necessary definitions. Galois connections are defined in section 2.1; infima and related notions are defined in section 2.3, and their duals in section 2.5.

2.1 The Question

A Galois connection involves two posets[1] $(\mathcal{A},\sqsubseteq)$ and $(\mathcal{B},\preceq)$ and two functions, $F\in\mathcal{A}\leftarrow\mathcal{B}$ and $G\in\mathcal{B}\leftarrow\mathcal{A}$. These four components together form a *Galois connection* iff for all $x\in\mathcal{B}$ and $y\in\mathcal{A}$

(1) $\qquad F.x\sqsubseteq y\equiv x\preceq G.y\quad .$

We refer to F as the *lower adjoint* and to G as the *upper adjoint*.

A Galois connection is thus a connection between two functions between posets. Typical accounts of the properties of Galois connections (for example [6]) focus on the properties of these *functions*. For example, given a function F, one may ask the question whether or not F is a lower adjoint in a Galois connection. The question posed by Hartmanis and Stearns was, however, rather different.

To motivate Hartmanis and Stearns' question, note that the statement $F.x\sqsubseteq y$ defines a *relation* between $\mathcal{B}$ and $\mathcal{A}$. So too does the statement $x\preceq G.y$. The existence of a Galois connection states that these two relations are equal. A natural question is therefore: under which conditions does an arbitrary (binary) relation between two posets define a Galois connection between the sets?

Exploring the question in more detail leads to two separate questions. The first is: suppose R is a relation between posets $\mathcal{B}$ and $\mathcal{A}$ (i.e. $R\subseteq\mathcal{B}\times\mathcal{A}$). What is a necessary and sufficient condition that there exist a function F such that

$$(x,y)\in R\ \equiv\ F.x\sqsubseteq y\qquad ?$$

The second is the dual of the first: given relation R, what is a necessary and sufficient condition that there exist a function G such that

$$(x,y)\in R\ \equiv\ x\preceq G.y\qquad ?$$

The conjunction of these two conditions is a necessary and sufficient condition for a given relation R to define a Galois connection. Such a relation is called a *pair algebra*.

[1]The pair $(\mathcal{A},\sqsubseteq)$ is a *poset* if $\sqsubseteq$ is a binary relation on $\mathcal{A}$ that is reflexive (i.e. $x\sqsubseteq x$ for all x), transitive (i.e. $x\sqsubseteq y\wedge y\sqsubseteq z\Rightarrow x\sqsubseteq z$ for all x, y and z) and antisymmetric (i.e. $x\sqsubseteq y\wedge y\sqsubseteq x\Rightarrow x=y$ for all x and y .

2.2 Least Elements

We begin with the first of the questions raised above. That is, supposing R is a relation between posets $\mathcal{B}$ and $\mathcal{A}$ (i.e. $R\subseteq\mathcal{B}\times\mathcal{A}$), what is a necessary and sufficient condition that there exist a function $F\in\mathcal{A}\leftarrow\mathcal{B}$ such that

$$(x,y)\in R \;\equiv\; F.x\sqsubseteq y \quad ?$$

In order to simplify our analysis let us first make an abstraction step. We are required to find —*for each* x— a value $F.x$ such that $(x,y)\in R \equiv F.x\sqsubseteq y$. Suppose we fix x and hide the dependence on x. Then the problem becomes one of determining for a given predicate p necessary and sufficient conditions guaranteeing that

$$(2) \qquad p.y \equiv a\sqsubseteq y$$

for some a. If we can solve this simplified problem then we have also solved our original problem by defining $p.y$ to be $(x,y)\in R$ and $F.x$ to be a.

It is easy to identify a necessary condition for (2) to hold. It must be the case that a is the *least* y satisfying p. That is, $p.a$ must hold (since $a\sqsubseteq a$) and for all y such that $p.y$ holds it must also be the case that $a\sqsubseteq y$. The question thus becomes: when is there a least element satisfying a given predicate p, and when does this least element, a say, have the property that, for all y, $p.y\Leftarrow a\sqsubseteq y$?

The least element satisfying a given property (if it exists) is characterised by two properties. First, it itself satisfies the property and, second, it is the "infimum" of all values satisfying the property. Let us introduce the definition of infimum in its full generality.

2.3 Infima and Their Preservation

We have occasion to use the notation $\langle x{::}\,E\rangle$ to denote the function that maps x to the value denoted by expression E. Here the domain of the function is understood from the context. We also write $\langle x{:}\,p{:}\,E\rangle$ where p is some Boolean valued expression if we want to restrict the domain of a function to arguments satisfying a property p. For example, $\langle x{:}\; a\sqsubseteq x{:}\; x\rangle$ denotes the identity function on that subset of some implicitly understood poset $\mathcal{A}$ consisting of all elements that are at least the element a. Function application is denoted by an infix dot in the case that an identifier is used to denote the function, and by juxtaposition if a symbol is used.

Suppose $(\mathcal{A},\sqsubseteq)$ and $(\mathcal{B},\preceq)$ are posets and $f\in\mathcal{A}\leftarrow\mathcal{B}$ is a monotonic function. Then an *infimum* of f is a solution of the equation:

$$x \;::\;\; \forall\langle a{::}\; a\sqsubseteq x \equiv \forall\langle b{::}\; a\sqsubseteq f.b\rangle\rangle \quad .$$

The poset $\mathcal{A}$ is *complete* if every monotonic function with range $\mathcal{A}$ has an infimum[2].

[2] A poset that is complete can be shown to be a lattice and is thus the term *complete lattice* is often used. Completeness of a poset should not be confused with the notion in the computing science literature of a *cpo* —short for *complete partial order*— where *co*completeness is required for *directed* sets.

Assuming this is the case, we denote the function mapping monotonic functions to their infima by $\sqcap$. That is, for all $a{\in}\mathcal{A}$ and monotonic $f\in\mathcal{A}{\leftarrow}\mathcal{B}$ for some $\mathcal{B}$,

(3) $\quad a\sqsubseteq\sqcap f \;\equiv\; \forall\langle b{::}\; a\sqsubseteq f.b\rangle$.

As an example consider $\sqcap\langle b{:}\,c\sqsubseteq b{:}\,b\rangle$ where c is some given constant. This is equal to c since, for all a,

$$a\sqsubseteq c \;\equiv\; \forall\langle b{:}\,c\sqsubseteq b{:}\,a\sqsubseteq b\rangle \quad .$$

Suppose that $\mathcal{B}$ is complete. Let $\sqcap$ denote the infimum operator for $\mathcal{B}$. Then $f\in\mathcal{A}{\leftarrow}\mathcal{B}$ is *inf-preserving* if and only if for all monotonic functions g with range $\mathcal{B}$

(4) $\quad \forall\langle a{::}\; a\sqsubseteq f.(\sqcap g) \;\equiv\; \forall\langle x{::}\; a\sqsubseteq f.(g.x)\rangle\rangle$.

The definition of inf-preserving does not require that $\mathcal{A}$ be complete. If that is the case, and we abuse notation by using $\sqcap$ to denote the infimum operator for both $\mathcal{A}$ and $\mathcal{B}$, then $f\in\mathcal{A}{\leftarrow}\mathcal{B}$ is inf-preserving if for all functions g with range $\mathcal{B}$

(5) $\quad f.(\sqcap g) \;=\; \sqcap(f{\circ}g)$.

(Here $f{\circ}g$ is, of course, the composition of f after g.) Although more complicated, we choose to use (4) rather than (5) because its form is closer to the statement of our original problem.

By turning the orderings around we obtain, in the usual way, the dual notions of *supremum*, *cocomplete* and *sup-preserving*. We will return to these dual notions later.

A predicate is a function with range Bool, the two-element set with elements true and false. Ordering Bool by implication ($\Rightarrow$) the infimum of a (monotonic) predicate p is the universal quantification $\forall p$ (that is $\forall\langle x{::}\,p.x\rangle$). Also, instantiating the definition of inf-preserving and simplifying using $\mathsf{true}{\Rightarrow}q \,\equiv\, q$ and $\mathsf{false}{\Rightarrow}q \,\equiv\, \mathsf{true}$, one easily obtains that, for predicate p,

(6) $\quad p$ is inf-preserving $\;\equiv\; \forall\langle g{::}\; p.(\sqcap g)\equiv\forall\langle x{::}\,p.(g.x)\rangle\rangle$.

It is easy to establish that an inf-preserving predicate is monotonic. For suppose $g_0\sqsubseteq g_1$. Then the function g with domain the poset $\{0{,}1\}$ ordered by $0\leq 1$ has infimum g_0. Thus, by the fact that p is inf-preserving, $p.g_0\equiv p.(\sqcap g)\equiv p.g_0\wedge p.g_1$. That is, if p is inf-preserving then

$$g_0\sqsubseteq g_1 \;\Rightarrow\; (p.g_0\Rightarrow p.g_1) \quad .$$

2.4 Basic Theorems

We are now ready to derive the definition of a pair algebra. Two elementary lemmas form the basis.

Lemma 7 Function f is inf-preserving if and only if for all a the predicate $\langle x{::}\; a\sqsubseteq f.x\rangle$ is inf-preserving.

Proof

$$\begin{array}{cl}
 & f \text{ is inf-preserving} \\
\equiv & \{ \quad \text{definition of inf-preserving} \quad \} \\
 & \forall\langle g{::}\ \forall\langle a{::}\ a \sqsubseteq f.(\sqcap g) \equiv \forall\langle x{::}\ a \sqsubseteq f.(g.x)\rangle\rangle\rangle \\
\equiv & \{ \quad \text{calculus} \quad \} \\
 & \forall\langle a{::}\ \forall\langle g{::}\ a \sqsubseteq f.(\sqcap g) \equiv \forall\langle x{::}\ a \sqsubseteq f.(g.x)\rangle\rangle\rangle \\
\equiv & \{ \quad (6) \quad \} \\
 & \forall\langle a{::}\ \langle x{::}\ a \sqsubseteq f.x\rangle \text{ is inf-preserving}\rangle \quad .
\end{array}$$

□

Noting that the identity function is inf-preserving, an immediate corollary of lemma 7 is that, for each a, the predicate $\langle x{::}\ a \sqsubseteq x\rangle$ is inf-preserving. A stronger statement is the following (well-known) lemma:

Lemma 8 Suppose p is a predicate on a complete poset. Then

$$p \text{ is inf-preserving} \quad \equiv \quad \exists\langle a{::}\ p \equiv \langle x{::}\, a \sqsubseteq x\rangle\rangle \quad .$$

Furthermore, if p is inf-preserving

$$\sqcap\langle x{:}\, p.x{:}\, x\rangle$$

is the unique solution of the equation

$$a{::} \quad p \equiv \langle x{::}\, a \sqsubseteq x\rangle \quad .$$

Proof The proof of the first part is by mutual implication. The follows-from is obtained by instantiating f in lemma 7 to the identity function, as already observed.

For the implication, assume that p is inf-preserving. Let a be $\sqcap\langle x{:}\,p.x{:}\,x\rangle$. We have to show that a satisfies $\forall\langle x{::}\ p.x \equiv a \sqsubseteq x\rangle$. Specifically, it is immediate from the definition of infimum that $p.x \Rightarrow a \sqsubseteq x$. For the other implication, we observe first that

$$\begin{array}{cl}
 & p.a \\
\equiv & \{ \quad \text{definition of } a \quad \} \\
 & p.\sqcap\langle x{:}\, p.x{:}\, x\rangle \\
\equiv & \{ \quad p \text{ is inf-preserving} \quad \} \\
 & \forall\langle x{:}\, p.x{:}\, p.x\rangle \\
\equiv & \{ \quad \text{calculus} \quad \} \\
 & \mathsf{true} \quad .
\end{array}$$

Thus,

$$
\begin{array}{ll}
 & a \sqsubseteq x \\
\Rightarrow & \{ \quad p \text{ is inf-preserving and thus monotonic} \quad \} \\
 & p.a \Rightarrow p.x \\
\equiv & \{ \quad \text{above} \quad \} \\
 & p.x \quad .
\end{array}
$$

The second part of the lemma is equally straightforward. Specifically,

$$
\begin{array}{ll}
 & p \;\equiv\; \langle x{::}\, a \sqsubseteq x \rangle \\
\Rightarrow & \{ \quad \text{predicate calculus, reflexivity of } \sqsubseteq \quad \} \\
 & \forall\langle x{:}\ p.x{:}\ a \sqsubseteq x \rangle \wedge p.a \\
\Rightarrow & \{ \quad \text{transitivity and reflexivity of } \sqsubseteq \quad \} \\
 & \forall\langle y{::}\ y \sqsubseteq a \equiv \forall\langle x{:} p.x{:} y \sqsubseteq x \rangle\rangle \\
\equiv & \{ \quad \text{definition of infimum} \quad \} \\
 & a = \sqcap\langle x{:} p.x{:} x \rangle \quad .
\end{array}
$$

□

Applying lemma 8 as we indicated we would be doing we have resolved "half" of the original pair algebra question:

Theorem 9 Suppose $\mathcal{B}$ is a set and $(\mathcal{A},\sqsubseteq)$ is a complete poset. Suppose $R \subseteq \mathcal{B} \times \mathcal{A}$ is a relation between the two sets. Then the following two statements are equivalent.

- There is a unique function F such that, for all $x \in \mathcal{B}$ and all $y \in \mathcal{A}$,

$$(x,y) \in R \;\equiv\; F.x \sqsubseteq y \quad .$$

- For all x, the predicate $\langle y{::}\ (x,y) \in R \rangle$ is inf-preserving.

Proof Define F by

(10) $\quad F.x \;=\; \sqcap\langle y{:} (x,y) \in R{:} y \rangle$

Then,

$$
\begin{array}{ll}
 & \forall\langle x,y{::}\ (x,y) \in R \;\equiv\; F.x \sqsubseteq y \rangle \\
\equiv & \{ \quad \text{lemma 8 with } p \text{ defined by } p.y \;\equiv\; (x,y) \in R \quad \} \\
 & \forall\langle x{::} \langle y{::}\ (x,y) \in R \rangle \text{ is inf-preserving} \rangle \quad .
\end{array}
$$

□

A simple example of theorem 9 is provided by the membership relation. For all sets S and all x (in a given universe of which S is a subset) we have:

$$x{\in}S \;\equiv\; \{x\}{\subseteq}S \quad .$$

This statement has the form

$$(x,y)\in R \;\equiv\; F.x \subseteq y$$

where the relation R is the membership relation. We thus deduce that the predicate $x{\in}$ is inf-preserving. That is, for all bags of sets $\mathcal{S}$,

$$x \in \cap\mathcal{S} \;\equiv\; \forall\langle S{:}\, S{\in}\mathcal{S}{:}\, x{\in}S\rangle \quad .$$

Another application of theorem 9 is the fundamental theorem on the existence of an adjoint to a given function:

Theorem 11 (Fundamental Theorem) Suppose that $\mathcal{B}$ is a poset and $\mathcal{A}$ is a complete poset. Then a monotonic function $G \in \mathcal{B} {\leftarrow} \mathcal{A}$ is an upper adjoint in a Galois connection iff G is inf-preserving.

Proof Define the relation R by $(x,y) \in R \equiv x \preceq G.y$ and function F as in (10). Then,

$$\begin{array}{ll}
 & G \text{ is inf-preserving} \\
\equiv & \quad \{ \quad \text{lemma 7} \quad \} \\
 & \forall\langle x{::}\; \langle y{::}\; x \preceq G.y\rangle \text{ is inf-preserving}\rangle \\
\equiv & \quad \{ \quad \text{theorem 9 with } R \text{ and } F \text{ defined as above} \quad \} \\
 & \forall\langle x,y{::}\; F.x \sqsubseteq y \;\equiv\; x \preceq G.y\rangle \quad .
\end{array}$$

□

2.5 Suprema and Cocompleteness

Every notion in poset theory can be dualised by turning the ordering relations around. The notion dual to infimum is "supremum" and the notion dual to completeness is "cocompleteness". Formally, suppose $(\mathcal{A}, \sqsubseteq)$ and $(\mathcal{B}, \preceq)$ are posets and $f \in \mathcal{A} {\leftarrow} \mathcal{B}$ is a monotonic function. Then a *supremum* of f is a solution of the equation:

$$x \;::\; \forall\langle a{::}\; x \sqsubseteq a \equiv \forall\langle b{::}\; f.b \sqsubseteq a\rangle\rangle \quad .$$

The poset $\mathcal{A}$ is *cocomplete* if every monotonic function with range $\mathcal{A}$ has a supremum. Assuming this is the case, we denote the function mapping monotonic functions to their supremum by $\sqcup$. That is, for all $a{\in}\mathcal{A}$ and monotonic $f \in \mathcal{A} {\leftarrow} \mathcal{B}$ for some $\mathcal{B}$,

(12) $\quad \sqcup f \sqsubseteq a \;\equiv\; \forall\langle b{::}\; f.b \sqsubseteq a\rangle \quad .$

The following theorem is of fundamental importance. It is of particular interest here because it is an immediate corollary of theorem 9.

Theorem 13 A partially ordered set is complete if and only if it is cocomplete.

Proof Suppose $\mathcal{A}$ is a complete poset. Let $\sqcap$ denote the infimum operator for $\mathcal{A}$. We use theorem 9 to show that $\mathcal{A}$ is cocomplete.

Consider the relation R between the set of functions with range $\mathcal{A}$ and $\mathcal{A}$ defined by

$$(f,a)\in R \equiv \forall\langle b{::}\ f.b \sqsubseteq a\rangle \quad .$$

Then, applying theorem 9, there is a function $\sqcup$ satisfying (12) if and only if the predicate $\langle a{::}\ (f,a)\in R\rangle$ is inf-preserving. We verify that this is indeed the case as follows:

$$\begin{array}{ll}
 & \langle a{::}\ (f,a)\in R\rangle \quad \text{is inf-preserving} \\
\equiv & \qquad \{ \qquad \text{definition of inf-preserving: } (6) \\
 & \qquad\qquad\quad \text{with } p.x \equiv \forall\langle b{::}\ f.b \sqsubseteq x\rangle \qquad \} \\
 & \forall\langle g{::}\ \forall\langle b{::}\ f.b \sqsubseteq \sqcap g\rangle \equiv \forall\langle x{::}\ \forall\langle b{::}\ f.b \sqsubseteq g.x\rangle\rangle\rangle \\
\equiv & \qquad \{ \qquad \text{definition of infimum: } (3) \quad \} \\
 & \forall\langle g{::}\ \forall\langle b{::}\forall\langle x{::}\ f.b \sqsubseteq g.x\rangle\rangle \equiv \forall\langle x{::}\forall\langle b{::}\ f.b \sqsubseteq g.x\rangle\rangle\rangle \\
\equiv & \qquad \{ \qquad \text{predicate caculus} \quad \} \\
 & \mathsf{true} \quad .
\end{array}$$

□

3 Pair Algebras

By continuing the dualisation process begun in section 2.5 we obtain a dual to theorem 9 and a solution to our original problem.

Recall that the predicate $\langle y{::}\ (x,y)\in R\rangle$ is inf-preserving if and only if

(14) $\quad \forall\langle x,f{::}\ (x,\sqcap.f)\in R \ \equiv\ \forall\langle z{::}\,(x,f.z)\in R\rangle\rangle \quad .$

Dually, the predicate $\langle x{::}\ (x,y)\in R\rangle$ is sup-preserving if and only if

(15) $\quad \forall\langle y,f{::}\ (\sqcup.f,y)\in R \ \equiv\ \forall\langle z{::}\,(f.z,y)\in R\rangle\rangle \quad .$

A relation R that satisfies both (14) and (15) is called a *pair algebra*. If R is a pair algebra then, by theorem 9 and its dual, the functions F and G defined by

$$F.x = \sqcap\langle y{:}\,(x,y)\in R{:}\,y\rangle$$

and

$$G.y = \sqcup\langle x{:}\,(x,y)\in R{:}\,x\rangle$$

form a Galois connection between the two posets. This theorem we call the *pair algebra theorem.* In this section we discuss several instances of the pair algebra theorem.

3.1 Concept Lattices

A common way to define a pair algebra is to take a function or relation and extend it to a relation between sets in such a way that the infimum and supremum preserving properties are automatically satisfied. An example is the following.

Consider a relation R on two posets $\mathcal{A}$ and $\mathcal{B}$. (Thus $R \subseteq \mathcal{A} \times \mathcal{B}$.) Define the relation $\overline{R}$ on subsets X and Y of $\mathcal{A}$ and $\mathcal{B}$, respectively, by

$$(X,Y) \in \overline{R} \;\equiv\; X \times Y \subseteq R \quad .$$

Clearly, the predicates $\langle X :: X \times Y \subseteq R \rangle$ and $\langle Y :: X \times Y \subseteq R \rangle$ are both sup-preserving. Thus, by the pair algebra theorem (taking care with the direction of the orderings) there are functions F_R and G_R such that for all subsets X and Y of $\mathcal{A}$ and $\mathcal{B}$, respectively,

$$F_R.X \supseteq Y \;\equiv\; X \times Y \subseteq R \;\equiv\; X \subseteq G_R.Y \quad .$$

This Galois connection is the basis of so-called *concept lattices* [4].

3.2 The Galois Correspondence

Galois' original correspondence between groups and fields has as basis a simple relation that is extended to be a pair algebra. This is explained below using the standard terminology of Galois theory [16].

Let K be a field and let F be an extension of K. Define the binary relation fixes between automorphisms, σ, of F and elements, f, of F by

$$\sigma \,\mathsf{fixes}\, f \;\equiv\; \sigma.f = f \quad .$$

Now we extend fixes in two steps. (We use the same name for the extensions, resolving ambiguity by the type of the variables in the definition.) First, we extend it to a relation between automorphisms, σ, and intermediate fields, E. (E is an intermediate field if $K \subseteq E \subseteq F$.) Specifically, we define

$$\sigma \,\mathsf{fixes}\, E \;\equiv\; \forall\langle e : e \in E : \sigma \,\mathsf{fixes}\, e \rangle \quad .$$

Second, we extend it to groups of automorphisms, G, by the definition

$$G \,\mathsf{fixes}\, E \;\equiv\; \forall\langle \sigma : \sigma \in G : \sigma \,\mathsf{fixes}\, E \rangle \quad .$$

(Note that, ignoring the group and field structures, the latter extension is exactly the same as the one used in the construction of concept lattices in section 3.1.)

Now the relation fixes between groups, G, (ordered by the subgroup relation) of automorphisms of F that fix K and intermediate fields, E, (ordered by the subfield relation) is a pair algebra. The Galois connection obtained from the pair algebra fixes is the classical "Galois correspondence" between automorphism groups and field extensions.

3.3 Hoare Triples

Perhaps the most prominent example of a pair algebra in the computing science literature is the notion of a Hoare triple [9].

Suppose S is a program statement and p and q are predicates on the state space of S. Then, one writes $\{p\}S\{q\}$ if after successful execution of statement S beginning in a state satisfying the predicate p the resulting state will satisfy predicate q. In such a case one says that statement S is *conditionally correct* with respect to precondition p and postcondition q. ("Conditional" refers to the fact that satisfying predicate q is conditional on the termination of statement S.)

In this way each program statement defines a relation on state predicates. This relation is such that, for all bags of predicates P,

$$\{\exists\langle p{:}\, p{\in}P{:}\, p\rangle\}S\{q\} \;\;\equiv\;\; \forall\langle p{:}\; p{\in}P{:}\; \{p\}S\{q\}\rangle$$

and also, for all bags of predicates Q,

$$\{p\}S\{\forall\langle q{:}\, q{\in}Q{:}\, q\rangle\} \;\;\equiv\;\; \forall\langle q{:}\; q{\in}Q{:}\; \{p\}S\{q\}\rangle \quad .$$

Thus, by the pair algebra theorem, for each statement S and predicate q there is a predicate $\mathsf{wlp}(S,q)$ satisfying

$$\{p\}S\{q\} \;\equiv\; p \Rightarrow \mathsf{wlp}(S,q) \quad .$$

There is also a predicate $\mathsf{slp}(S,\,p)$ satisfying

$$\{p\}S\{q\} \;\equiv\; \mathsf{slp}(S,\,p) \Rightarrow q \quad .$$

Combining the last two equations we thus have the Galois connection: for all predicates p and q,

$$\mathsf{slp}(S,\,p) \Rightarrow q \equiv p \Rightarrow \mathsf{wlp}(S,\,q) \quad .$$

The abbreviation "wlp" stands for "weakest liberal precondition" and "slp" for "strongest liberal postcondition".

3.4 Partition Pairs

Hartmanis and Stearns [8, p. 71] gave several examples of pair algebras, all of which were relations between partitions of finite sets. We describe one such example.

A *machine* is described by three items: its state set, its input alphabet and its state transition function. The transition function maps a state and an input symbol to a state. Given a machine, we will denote its transition function by δ. That is, for given state s and given input symbol a, $\delta(s,a)$ is the next state.

Suppose that π and τ are partitions of the state set of a machine M. Let $[s]_\pi$ denote the equivalence class of states to which state s belongs under partition π. Then the pair (π,τ) is a *partition pair* if it is the case that

(16) $\quad [s]_\pi = [t]_\pi \;\Rightarrow\; [\delta(s,a)]_\tau = [\delta(t,a)]_\tau$

for all states s and t, and all inputs a.

The observation made by Hartmanis and Stearns was that the partition pair relation is a pair algebra. Because the lattice of partitions (ordered by refinement) of the state set of a finite state machine is itself finite this can be checked by verifying the following properties: First, the pair $(\pi, \top\!\!\top)$ is a partition pair, where $\top\!\!\top$ denotes the top element of the lattice. (The top element of the lattice of partitions lumps all states together in one state. Thus substituting $\top\!\!\top$ for τ in (16) makes the right side of the implication vacuously true.) Second, the pair $(\perp\!\!\perp, \tau)$ is a partition pair, where $\perp\!\!\perp$ denotes the bottom element of the lattice. (This follows because, for all states s and t, $[s]_{\perp\!\!\perp} = [t]_{\perp\!\!\perp}$ equivales $s=t$.) Third, if (π_1, τ) and (π_2, τ) are both partition pairs then so is $(\pi_1 \sqcup \pi_2, \tau)$ ($\sqcup$ being the supremum operator on partitions). Fourth, if (π, τ_1) and (π, τ_2) are both partition pairs then so is $(\pi, \tau_1 \sqcap \tau_2)$ ($\sqcap$ being the infimum operator on partitions).

Applying the pair algebra theorem it thus follows that there is a function next such that

$$(\pi, \tau) \text{ is a partition pair} \equiv \mathsf{next}.\pi \sqsubseteq \tau$$

where $\sqsubseteq$ denotes the refinement ordering on partitions. Specifically, using $\sqcap$ to denote the infimum operation on partitions,

$$\mathsf{next}.\pi = \sqcap\langle\tau{:}\ (\pi, \tau) \text{ is a partition pair:}\ \tau\rangle \ .$$

Note that $\mathsf{next}.\pi$ is the *least* partition τ such that (π, τ) is a partition pair. Thus, $\mathsf{next}.\pi$ specifies the *largest* amount of information that is known about the next state given only the π-class of the current state. Dually, there is a function prev such that

$$(\pi, \tau) \text{ is a partition pair} \equiv \pi \sqsubseteq \mathsf{prev}.\tau \ .$$

Specifically,

$$\mathsf{prev}.\tau = \sqcup\langle\pi{:}\ (\pi, \tau) \text{ is a partition pair:}\ \pi\rangle \ .$$

Given a partition τ, $\mathsf{prev}.\tau$ specifies the least amount of information needed about the current state in order to determine the τ-class of the next state.

Combining the two equivalences above, we have the Galois connection between next and prev:

$$\mathsf{next}.\pi \sqsubseteq \tau \ \equiv \ \pi \sqsubseteq \mathsf{prev}.\tau \ .$$

Acknowledgement

Thanks go to Lex Bijlsma and Henk Doornbos for their careful and constructive reading of a draft of this report.

References

[1] Garrett Birkhoff. *Lattice Theory.* American Mathematical Society, Providence, Rhode Island, revised edition, 1948.

[2] J.H. Conway. *Regular Algebra and Finite Machines.* Chapman and Hall, London, 1971.

[3] Patrick Cousot and Radhia Cousot. Abstract interpretation: A unifed lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth Annual ACM Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, January 1977.

[4] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order.* Cambridge Mathematical Textbooks. Cambridge University Press, first edition, 1990.

[5] W.H.J. Feijen and Lex Bijlsma. Exercises in formula manipulation. In E.W. Dijkstra, editor, *Formal Development of Programs and Proofs*, pages 139–158. Addison-Wesley Publ. Co., 1990.

[6] G. Gierz, K. H. Hofmann, K. Keimel, J.D. Lawson, M. Mislove, and D. S. Scott. *A Compendium of Continuous Lattices.* Springer-Verlag, 1980.

[7] J. Hartmanis and R.E. Stearns. Pair algebras and their application to automata theory. *Information and Control*, 7(4):485–507, 1964.

[8] J. Hartmanis and R.E. Stearns. *Algebraic Structure Theory of Sequential Machines.* Prentice-Hall, 1966.

[9] C.A.R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, 1969.

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

[11] C.A.R. Hoare and He Jifeng. *Unifying Theories of Programming.* Prentice-Hall International, 1998.

[12] J. Lambek. The mathematics of sentence structure. *The American Mathematical Monthly*, 65:154–170, 1958.

[13] J. Lambek. Some Galois connections in elementary number theory. *J. of Number Theory*, 47(3):371–377, June 1994.

[14] A. Melton, D. A. Schmidt, and G. E. Strecker. Galois connections and computer science applications. In David Pitt, Samson Abramsky, Axel Poigné, and David Rydeheard, editors, *Category Theory and Computer Programming*, number 240 in Lecture Notes in Computer Science, pages 299–312. Springer-Verlag, 1986.

[15] Oystein Ore. Galois connexions. *Transactions of the American Mathematical Society*, 55:493–513, 1944.

[16] Ian Stewart. *Galois Theory*. Chapman and Hall, 2nd edition, 1989.

[17] B. von Karger. *Temporal Algebra*. Habilitationsschrift, University of Kiel, 1998.