Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 207 (2008) 203–217
www.elsevier.com/locate/entcs

Pre-Galois Connection on Coalgebras for Generic Component Refinement

Sun Meng[1]

CWI, Kruislaan 413, Amsterdam, The Netherlands

Abstract

The technique of Galois connections has been applied successfully in many areas of computer science. By employing coalgebras as models for software components, we claim that different forms of behavior model and types of state transitions for components are instances of a single form of coalgebra in a Kleisli category. Based on the Kleisli category, the results on forward/backward morphisms and refinement of components in **Set** are still satisfied in this more generic framework. We propose a notion of pre-Galois connection in the context of coalgebras for refinement of state-based software components which takes into consideration not only the refinement ordering but also the dynamics of the components, and we study its properties in the Kleisli category. This notion is a powerful tool for relating a component to its refinement and for relating a component to its abstraction. Thus it provides a basis for reasoning about state-based software designs and reverse engineering.

Keywords: Component, Refinement, Coalgebra, Pre-Galois Connection

1 Introduction

In the past decade or so, the notion of component based software development [17] emerged as a promising paradigm to deal with the ever increasing complexity in software design, evolution and reuse. A component is a modular, deployable, and replaceable part of a system that encapsulates implementation and exposes a public interface. In component based software development, a component must be specified and then implemented before it can be analyzed and used. The development step from specification to implementation is called refinement.

Based on a coalgebraic model for component based systems [2,3,5] in which components are modelled as coalgebras and can be aggregated through a number of combinators to build hierarchical models of complex systems, [13] introduces the notion of refinement for generic components, including a soundness result. Later [12] investigates *architectural refinement* and proved a completeness result. By

[1] Email: M.Sun@cwi.nl

doi:10.1016/j.entcs.2008.03.094

using the pointfree binary relation calculus, [4] addresses the refinement for partial components in the coalgebra context.

Galois connection is a concept introduced by Oystein Ore in [15], and has been widely used in many areas of computer science to ensure the correctness of implementation with respect to the specification [6,8,11]. In this paper, we approach the notion of *pre-Galois connection* between coalgebras as a formal relationship between one component and its refinement. As in our previous work [13,12], we consider components as coalgebras. The difference between previous work and this paper is that the coalgebra model for components in this paper goes one step higher on the "generic" ladder, because it unifies the behavior model and transition types into one functor over the Kleisli category instead of using one monad and one functor for representing them respectively. By using the coalgebra framework we can take refinement (or, abstraction on the other direction) as transformation from one coalgebraic structure to another.

The concept of pre-Galois connection we study in this paper is similar to Galois connection in a coalgebraic context, based on the notion of ordering on a functor proposed in [10]. The difference between this and a Galois connection is that it is a notion for two dimensions: not only the refinement ordering but also the dynamics of the components are considered here. Furthermore, the refinement order considered in this paper is not a *partial order*, but a *preorder*. Therefore, for some properties that are satisfied by the classical Galois connection, the corresponding parts in pre-Galois connection are not exactly the same. Especially, the *equality* for many results in Galois connection is replaced by the *two-way similarity* relationship in the context of pre-Galois connection. We also show how pre-Galois connections can be used in reasoning about refinement of components. A pre-Galois connection is a pair of arrows on the carrier sets U and V of two $\mathscr{K}(\mathsf{T})$-coalgebras in the Kleisli category. By using the familiar "lifting" technique [9,10], it can be transformed into a Galois connection between $\mathscr{K}(\mathsf{T})U$ and $\mathscr{K}(\mathsf{T})V$, which are equipped with the refinement preorder. We explain in what formal sense a pre-Galois connection establishes that one component refines another, and show how pre-Galois connections help to prove existence of simulations between components.

The structure of this paper goes as follows: The underlying coalgebraic model for components is briefly discussed in Section 2. Section 3 introduces forward and backward morphisms as refinement "witnesses". Section 4 contains an introduction on refinement and simulation of components. Section 5 gives the definition of pre-Galois connection and provides a family of properties that a pre-Galois connection should satisfy. Section 6 discusses the possibilities of applying pre-Galois connections in refinement of state-based components. Concluding remarks, together with some prospects for future work, are presented in Section 7.

2 Components as Coalgebras

In [2,3] software components have been characterized as dynamical systems with a public interface and a private, encapsulated state. In this paper, we adopt the

coalgebraic model for software components which follows closely the "*components as coalgebras*" approach proposed in [2,3]. This approach provides an observational semantics for software components and an assembly calculus. For more details the reader is referred to [1,2]. However, we make one step further by using the Kleisli category instead of **Set** as the base category and thus unifying the behavior model and transition type aspects into one functor over the Kleisli category.

In this approach, components are specified in a generic way, where "generic" means that the underlying behavior model is taken as a specification parameter, and abstracted to a monad B. Some useful possibilities are as follows:

- *Identity*, $\mathsf{B} = \mathsf{Id}$, which retrieves the simple total and deterministic behavior.
- *Partiality*, $\mathsf{B} = \mathsf{Id} + \mathbf{1}$, i.e., the maybe monad, capturing the partial behavior which describes the possibility of deadlock or failure.
- *Non-determinism*, $\mathsf{B} = \mathscr{P}$, modeling the non-deterministic branching behavior.
- *Probabilistic non-determinism*, $\mathsf{B} = \mathscr{D}$ which models the probabilistic branching behavior. Here for a set X, $\mathscr{D}X = \{\xi : X \to [0,1] \mid \sum_{x \in X} \xi(x) \leq 1\}$, and ξ is called a probability subdistribution over X.
- *General "metric" non-determinism*, supported on a notion of a *bag* monad based on commutative monoids $(M, \oplus, \otimes)$ where $\otimes$ distributes over $\oplus$. This monads captures situations in which, among the possible future evolutions of the component, some are more likely (or cheaper, more secure, etc.) than others.

Moreover, the type of state transitions of the component is described by a functor T. For example, if we take I and O be sets acting as component input and output interfaces, then T can be defined as the **Set** endofunctor $\mathsf{T} = (\mathsf{Id} \times O)^I$. More possibilities of the functor T can be found in [1]. Therefore, a state-based component can be modeled as a pointed coalgebra $(u \in U, \alpha : U \to \mathsf{B}\mathsf{T}U)$ in **Set** with B a monad, T a functor, and a distributive law $\mathsf{T}\mathsf{B} \Rightarrow \mathsf{B}\mathsf{T}$ implicit, where the point u is taken as the "initial" or "seed" state. The distributive law describes the way how B's effect is distributed over the transition type represented by T. In this paper, we only consider T as a polynomial functor.

For each monad B on **Set**, the Kleisli category for B, denoted by $\mathscr{K}(\mathsf{B})$, can be constructed as follows:

- Objects in $\mathscr{K}(\mathsf{B})$ are the same as in **Set**. They are just sets.
- An arrow $U \to V$ in $\mathscr{K}(\mathsf{B})$ is a function $U \to \mathsf{B}V$ in **Set**.
- Composition of arrows in $\mathscr{K}(\mathsf{B})$ is defined using multiplication $\mu_U : \mathsf{B}\mathsf{B}U \to \mathsf{B}U$.
- Identity arrow $\mathsf{id} : U \to U$ in $\mathscr{K}(\mathsf{B})$ is the unit $\eta_U : U \to \mathsf{B}U$ in **Set**.

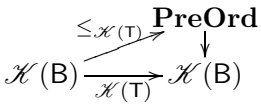
This category $\mathscr{K}(\mathsf{B})$ will be our base category in the following sections. Note that when we write an arrow $U \to V$ in $\mathscr{K}(\mathsf{B})$, the effects of monad B is implicit because it is a function $U \to \mathsf{B}V$ in **Set**. As an example, we consider the category $\mathscr{K}(\mathscr{P})$ for the powerset monad $\mathscr{P}$. It is in fact isomorphic to the category **Rel** of sets and relations. That is, an arrow $U \to V$ in $\mathscr{K}(\mathscr{P})$ is exactly a relation between U and V via the correspondence: given a function $f : U \to \mathscr{P}V$ in **Set**

we can obtain a relation $R_f = \{(u, v) \mid v \in f(u)\}$. Furthermore, the composition of arrows and identity arrow in $\mathscr{K}(\mathscr{P})$ are the relational composition and diagonal relation respectively.

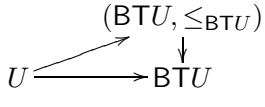
In [7], it has been shown that via the distributive law, the functor T can be lifted to a functor $\mathscr{K}(\mathsf{T})$ on the Kleisli category $\mathscr{K}(\mathsf{B})$, and the base category can be moved from **Set** to $\mathscr{K}(\mathsf{B})$. Thus, considering the component model in [1,2,13], a component is just a pointed coalgebra $(u \in U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ in the Kleisli category $\mathscr{K}(\mathsf{B})$. In the following sections we may omit the seed state for simplifying notations.

Note that the notion of behavior models captured by the monad B can naturally come with notions of order on $\mathsf{B}U$.

Definition 2.1 For a Kleisli category $\mathscr{K}(\mathsf{B})$ and any functor T, an order $\leq_{\mathscr{K}(\mathsf{T})}$ on $\mathscr{K}(\mathsf{T})$ is defined as a collection of preorders $\leq_{\mathsf{BT}U} \subseteq \mathsf{BT}U \times \mathsf{BT}U$, for each set U, such that the following diagram commutes:



concretely



and for any $f : U \to V$, $\mathscr{K}(\mathsf{T})f$ preserves the order, i.e.,

$$u_1 \leq_U u_2 \;\Rightarrow\; \mathscr{K}(\mathsf{T})f(u_1) \leq_{\mathsf{BT}V} \mathscr{K}(\mathsf{T})f(u_2)$$

We can consider some possible examples of $\leq_{\mathscr{K}(\mathsf{T})}$:

- The first example is as follows:

$$x \subseteq_{\mathsf{Id}} y \;\text{ iff }\; x = y$$
$$x \subseteq_{\mathscr{P}} y \;\text{ iff }\; \forall_{e \in x} \exists_{e' \in y} .e \subseteq_{\mathsf{Id}} e'$$

  The order $\subseteq_{\mathscr{P}}$ captures the classical notion of nondeterministic reduction.

- The order $\subseteq_{\mathscr{P}}$ can be turned into more specific cases. For example, the failure forcing variant $\subseteq^E_{\mathscr{P}}$, where E stands for emptyset, guarantees that the first component fails no more than the second one. It is defined by replacing the clause for $\subseteq_{\mathscr{P}}$ by

$$x \subseteq^E_{\mathscr{P}} y \;\text{ iff }\; (x = \emptyset \Rightarrow y = \emptyset) \;\wedge\; \forall_{e \in x} \exists_{e' \in y} .e \subseteq_{\mathsf{Id}} e'$$

- Consider the partiality monad $\mathsf{B} = \mathsf{Id} + \mathbf{1}$. The set $\mathsf{B}U$ carry the familiar "flat" order:

$$x \subseteq_{\mathsf{B}} y \;\text{ iff }\; x \neq * \Rightarrow x = y \;\wedge x = * \Rightarrow y = *$$

In the following sections, sometimes we may drop the subscripts in $\leq_{\mathsf{B}U}$, e.g., $\leq$ instead of $\leq_{\mathsf{B}U}$, when it is clear from context for notation economy. Moreover, we denote the lifting of the order $\leq$ to the arrows as:

$$f \leq g \;\text{ iff }\; \forall x \;.f(x) \leq g(x) \tag{1}$$

3 Forward and Backward Morphisms

The dynamics of a component is captured by the functor $\mathscr{K}(\mathsf{T})$. Thus, a possible (and intuitive) way of considering component p as a refinement of another component q is to consider that p-transitions are simply preserved in q. For example, for non-deterministic components this means set inclusion. To make precise such a "definition", the transitions are generalized as follows: Recall that a component morphism from p to q is a seed preserving function $h : U_p \to U_q$ such that

$$\mathscr{K}(\mathsf{T})h \cdot \alpha_p = \alpha_q \cdot h \tag{2}$$

In fact, just as transition systems can be coded back as coalgebras, any T-coalgebra also specifies a T-shaped transition structure over the carrier U:

$$u \to u' \;\equiv\; u' \in_{\mathsf{T}} \alpha(u)$$

where $\in_{\mathsf{T}}$ can be defined by induction on the structure of T:

$$\begin{aligned}
x \in_{\mathsf{Id}} y &\;\text{ iff }\; x = y \\
x \in_{K} y &\;\text{ iff }\; \mathsf{false} \\
x \in_{\mathsf{T}_1 \times \mathsf{T}_2} y &\;\text{ iff }\; x \in_{\mathsf{T}_1} \pi_1 y \;\vee\; x \in_{\mathsf{T}_2} \pi_2 y \\
x \in_{\mathsf{T}_1 + \mathsf{T}_2} y &\;\text{ iff }\; \begin{cases} y = \iota_1 y' & \Rightarrow\; x \in_{\mathsf{T}_1} y' \\ y = \iota_2 y' & \Rightarrow\; x \in_{\mathsf{T}_2} y' \end{cases} \\
x \in_{\mathsf{T}^K} y &\;\text{ iff }\; \exists_{k \in K}\, .x \in_{\mathsf{T}} yk \\
x \in_{\mathscr{P}(\mathsf{T})} y &\;\text{ iff }\; \exists_{y' \in y}.\, x \in_{\mathsf{T}} y'
\end{aligned}$$

In terms of transitions, equation (2) can be translated into the following two requirements by a straightforward generalization of an argument in [16]:

$$\begin{aligned}
u \to u' &\;\Rightarrow\; h(u) \to h(u') \\
h(u) \to v' &\;\Rightarrow\; \exists u' \in U\, .u \to u' \wedge v' = h(u')
\end{aligned}$$

which jointly states that not only p dynamics as represented by the induced transition relation, is preserved by h, but also the q dynamics is reflected back over the same h. Note that in the study of refinement, both preservation and reflection of behavior are useful, because we need the abstract model for both validating desired properties and analyzing for the presence of undesirable properties.

The classic tool for relating two coalgebras is the homomorphism which can be defined by (2). However, from such homomorphisms we can only derive bisimulations [16]. Thus, in order to build a witness for refinement relations, we separate the preservation and reflection aspects in homomorphism and get the following definition:

Definition 3.1 For a Kleisli category $\mathscr{K}(\mathsf{B})$ and two coalgebras $p = (U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $q = (V, \beta : V \to \mathscr{K}(\mathsf{T})V)$. A forward morphism $h : p \to q$ with

respect to an order $\leq$ on $\mathscr{K}(\mathsf{T})$ is an arrow $h : U \to V$ such that

$$\mathscr{K}(\mathsf{T})h \cdot \alpha \;\leq\; \beta \cdot h$$

Dually, h is called a backward morphism if the following conditions are satisfied:

$$\beta \cdot h \;\leq\; \mathscr{K}(\mathsf{T})h \cdot \alpha$$

Although the base category has been changed from **Set** to Kleisli category, some results similar to those in **Set** can still be easily proved. For example,

Lemma 3.2 *For a Kleisli category $\mathscr{K}(\mathsf{B})$ and a* **Set** *functor T, pointed $\mathscr{K}(\mathsf{T})$ coalgebras and forward (backward, respectively) morphisms form a category.*

Proof. In both cases, identities are the identities on the carrier and composition is inherited from $\mathscr{K}(\mathsf{B})$. What remains to be shown is that the composition of forward (backward respectively) morphisms yields a forward (backward respectively) morphism. So, for $\mathscr{K}(\mathsf{T})$ coalgebras $p = (U, \alpha)$, $q = (V, \beta)$ and $r = (W, \gamma)$, let $h : p \to q$ and $k : q \to r$ be two forward (backward respectively) morphisms, then

(*forward* case)	(*backward* case)
$\mathscr{K}(\mathsf{T})(k \cdot h) \cdot \alpha$	$\gamma \cdot (k \cdot h)$
$= \{\mathscr{K}(\mathsf{T})$ functor$\}$	$= \{\cdot$ associate$\}$
$\mathscr{K}(\mathsf{T})k \cdot (\mathscr{K}(\mathsf{T})h \cdot \alpha)$	$(\gamma \cdot k) \cdot h$
$\leq \{h$ forward and (2.1)$\}$	$\leq \{k$ backward$\}$
$\mathscr{K}(\mathsf{T})k \cdot (\beta \cdot h)$	$(\mathscr{K}(\mathsf{T})k \cdot \beta) \cdot h$
$= \{\cdot$ associate$\}$	$= \{\cdot$ associate$\}$
$(\mathscr{K}(\mathsf{T})k \cdot \beta) \cdot h$	$\mathscr{K}(\mathsf{T})k \cdot (\beta \cdot h)$
$\leq \{k$ forward$\}$	$\leq \{h$ backward and (2.1)$\}$
$(\gamma \cdot k) \cdot h$	$\mathscr{K}(\mathsf{T})k \cdot \mathscr{K}(\mathsf{T})h \cdot \alpha$
$= \{\cdot$ associate$\}$	$= \{\mathscr{K}(\mathsf{T})$ functor$\}$
$\gamma \cdot (k \cdot h)$	$\mathscr{K}(\mathsf{T})(k \cdot h) \cdot \alpha$

□

4 Refinement and Simulation of Components

The existence of a forward (backward) morphism connecting two components p and q witnesses a refinement situation whose symmetric closure coincides, as expected, with bisimulation. In the sequel we will restrict ourselves to forward refinement [2] and define *behavior refinement* as the existence of a forward morphism up to bisimulation $\sim$. Formally,

Definition 4.1 Component p is a behavior refinement of q, written $p \sqsubseteq_B q$, if there exist components r and s such that $p \sim r$, $q \sim s$ and a (seed preserving) forward morphism from r to s.

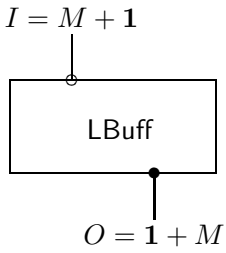
[2] A similar study can be made about backward refinement.

The exact meaning of a refinement assertion $p \sqsubseteq_B q$ depends, of course, on the concrete refinement preorder $\leq$ adopted. Behavior refinement has a number of pleasant properties. For example,

$$p \sqsubseteq_B p$$
$$p \sqsubseteq_B q \wedge q \sqsubseteq_B r \Rightarrow p \sqsubseteq_B r$$

Example 4.2 As an example of behaviour refinement, we consider the lossy buffer LBuff component, which is a buffered channel that occasionally loses messages, as represented below:

$$\begin{cases} \mathsf{put}: & M \longrightarrow \mathbf{1} \\ \mathsf{pick}: & \mathbf{1} \longrightarrow M \end{cases}$$



The put and pick operations are regarded as 'buttons' or 'ports', whose signatures are grouped together in the diagram (M stands for a message parameter type, $\mathbf{1}$ for the nullary datatype and $+$ for 'datatype sum'). One might capture LBuff dynamics by a function $a_{\mathsf{LBuff}} : U \times I \to \mathscr{P}(U \times O)$ where U denotes the space state. This describes how LBuff reacts to input stimuli, produces output data (if any) and changes state. It can also be written in a curried form as a coalgebra of signature $U \longrightarrow \mathscr{K}(\mathsf{T})U$ in the Kleisli category $\mathscr{K}(\mathscr{P})$ where functor T captures the transition 'shape':

$$\mathsf{T} = (\mathsf{Id} \times O)^I$$

As a refinement we consider a deterministic buffered channel Buff specified as a coalgebra $M^* \longrightarrow (M^* \times (\mathbf{1} + M))^{M+\mathbf{1}}$ with nil as the initial state, and dynamics given by standard operations on lists. Other possible behavior refinements of LBuff would arise by choosing different strategies for delivering elements from the buffer. Here are some possibilities, each of them is witnessed by a forward morphism:

- the queuing strategy, leading to the specification Buff;
- the stack strategy (LIFO deliver);
- the priority strategy (in which elements carry some probability information);
- the lift strategy (a linear order on the elements is served in alternating increasing and decreasing order).

In the priority strategy, for example, elements are labelled with a 'show-up' probability, introducing an elementary form of probabilistic nondeterminism. As detailed in [3], the corresponding behavior monad is generated by a monoid $\langle [0,1], \mathsf{min}, \times \rangle$ with the additional requirement that for each $m \in M$, $\sum(\mathscr{P}\pi_2)m = 1$. 'Probabilis-

tic' components can be embedded into the space of 'plain nondeterministic' ones where behaviour refinement, wrt the corresponding refinement order, is discussed.

A forward morphism is a "behavior preserving" mapping, but lying inside it is a more fundamental concept: to relate two coalgebras, one must show that all the transitions in one coalgebra are "mimicked" by the other. Such an intuition is formalized by the notion of *simulation*. In [13,12] we have proved that behavior refinement, as characterized above, can be established by a *simulation* relation $R \subseteq U_p \times U_q$ on the state spaces of the 'concrete' (p) and the 'abstract' (q) components. The generic definition of simulation is due to Jacobs and Hughes in [10]:

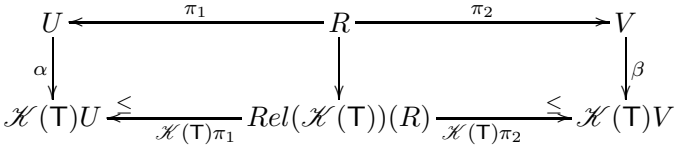
Definition 4.3 For a given Kleisli category $\mathscr{K}(\mathsf{B})$, a functor T and a refinement preorder $\leq$, a lax relation lifting is an operation $Rel_{\leq}(\mathscr{K}(\mathsf{T}))$ mapping relation R to $\leq \cdot Rel(\mathscr{K}(\mathsf{T}))(R) \cdot \leq$, where $Rel(\mathscr{K}(\mathsf{T}))(R)$ is the lifting of R to $\mathscr{K}(\mathsf{T})$ defined, as usual, as the $\mathscr{K}(\mathsf{T})$-image of inclusion $\langle r_1, r_2\rangle : R \to U \times V$:

$$\langle \mathscr{K}(\mathsf{T})r_1, \mathscr{K}(\mathsf{T})r_2\rangle : \mathscr{K}(\mathsf{T})R \to \mathscr{K}(\mathsf{T})U \times \mathscr{K}(\mathsf{T})V$$

Given $\mathscr{K}(\mathsf{T})$-coalgebras (U, α) and (V, β), a simulation is a $Rel_{\leq}(\mathscr{K}(\mathsf{T}))$-coalgebra over α and β, i.e., a relation R such that, for all $u \in U, v \in V$,

$$(u, v) \in R \;\Rightarrow\; (\alpha\, u, \beta\, v) \in Rel_{\leq}(\mathscr{K}(\mathsf{T}))(R)$$

The following diagram in $\mathscr{K}(\mathsf{B})$ for $\mathscr{K}(\mathsf{T})$ coalgebras, captures simulations:



The union of all simulations is still a simulation, which is called similarity and denoted by $\lesssim$. The two-way similarity is defined as $\eqsim = \lesssim \cap \lesssim^{op}$, i.e., for two states u and v,

$$u \eqsim v \text{ iff } u \lesssim v \wedge v \lesssim u$$

Note that bisimilarity implies two-way similarity, but the converse is not always satisfied. The condition under which two-way similarity implies bisimilarity can be found in [10]. For two arrows $f, g : U \to V$, we write $f \eqsim g$ iff for all $u \in U$, $f(u) \eqsim g(u)$.

The soundness and completeness results of simulation for behavior refinement in the category **Set** has been proved in [13,12], and also holds in the Kleisli category. It is given in the following lemma:

Lemma 4.4 *For two coalgebras p and q,*

- *To prove $p \sqsubseteq_B q$ it is sufficient to exhibit a simulation R relating p and q.*
- *If $p \sqsubseteq_B q$ and h is the witness forward morphism, then $\sim \cdot Graph(h) \cdot \sim$ is a simulation between p and q.*

It is easy to extract the simulation embedded within a forward morphism: for $h : U \to V$, we define $(u, v) \in R$ iff $h(u) \leq_{\mathsf{B}V} v$. Note that $h(u) \leq_{\mathsf{B}V} v$ is stated rather than $h(u) = v$ as in **Set**, to take into account the order upon V. It is straightforward to prove that R is a simulation.

Simulations prove to be a useful foundation for reasoning about refinement of components, and we can synthesize a simulation from scratch for two finite-state components. In next section, we consider another powerful tool: *pre-Galois connection.*

5 Pre-Galois Connection

In the previous section we have seen how components are related by forward/backward morphisms and simulations to witness the refinement relationship. However, in practice, because forward/backward morphisms are functions, they only give incomplete insight about how one determines the states of the abstract component that correspond to a set of states of the concrete component. To solve these problems, we make a generalization of forward/backward morphism and define pre-Galois connection on coalgebras as follows:

Definition 5.1 For a Kleisli category $\mathscr{K}(\mathsf{B})$ and functor T, let $\leq$ be an order on B, a pre-Galois connection between two $\mathscr{K}(\mathsf{T})$-coalgebras (U, α) and (V, β) is a pair of arrows $f : U \to V$ and $g : V \to U$, such that for all $u \in U$ and $v \in V$,

$$\alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})g \cdot \beta(v) \text{ iff } \mathscr{K}(\mathsf{T})f \cdot \alpha(u) \leq_{\mathscr{K}(\mathsf{T})V} \beta(v)$$

We say that f is the lower adjoint and g is the upper adjoint of the pre-Galois connection.

The following lemmas show the basic results that pre-Galois connected functions can be composed to form new pre-Galois connections, and identity arrow can always be used to build pre-Galois connection.

Lemma 5.2 *If (f, g) is a pre-Galois connection between two coalgebras $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$, and (h, k) is a pre-Galois connection between two coalgebras $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$ and $(W, \gamma : W \to \mathscr{K}(\mathsf{T})W)$, then $(h \cdot f, g \cdot k)$ is a pre-Galois connection between $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $(W, \gamma : W \to \mathscr{K}(\mathsf{T})W)$.*

Proof. We have, for all $u \in U$ and $w \in W$,

$$\begin{aligned}
&\alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})(g \cdot k) \cdot \gamma(w) \\
\equiv\ &\{\mathscr{K}(\mathsf{T})\text{is a functor}\} \\
&\alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})g \cdot \mathscr{K}(\mathsf{T})k \cdot \gamma(w) \\
\equiv\ &\{(f,g)\text{is a pre-Galois connection}\} \\
&\mathscr{K}(\mathsf{T})f \cdot \alpha(u) \leq_{\mathscr{K}(\mathsf{T})V} \mathscr{K}(\mathsf{T})k \cdot \gamma(w) \\
\equiv\ &\{(h,k)\text{is a pre-Galois connection}\} \\
&\mathscr{K}(\mathsf{T})h \cdot \mathscr{K}(\mathsf{T})f \cdot \alpha(u) \leq_{\mathscr{K}(\mathsf{T})W} \gamma(w) \\
\equiv\ &\{\mathscr{K}(\mathsf{T})\text{is a functor}\} \\
&\mathscr{K}(\mathsf{T})(h \cdot f) \cdot \alpha(u) \leq_{\mathscr{K}(\mathsf{T})W} \gamma(w)
\end{aligned}$$

□

Lemma 5.3 $(\mathsf{id}, \mathsf{id})$ *where* id *denotes the identity function on* U *is a pre-Galois connection between a coalgebra* $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ *and itself.*

Proof. Obvious. □

If we introduce an order $\preccurlyeq_U$ on U for $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ as follows:

$$u \preccurlyeq_U u' \text{ iff } \alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \alpha(u') \tag{3}$$

i.e., we assume that $\preccurlyeq$ reflects the transition structure $\to$. In other words, the functor $\mathscr{K}(\mathsf{T})$ is order-preserving, and then the cancellation results similar to those in Galois connections can be inferred.

Lemma 5.4 *If* (f, g) *is a pre-Galois connection between two coalgebras* $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ *and* $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$*, then we have*

$$f \cdot g \preccurlyeq_V \mathsf{id}_V \text{ and } \mathsf{id}_U \preccurlyeq_U g \cdot f$$

Proof. It suffices to show that $f \cdot g \preccurlyeq_V \mathsf{id}_V$. The proof for $\mathsf{id}_U \preccurlyeq_U g \cdot f$ is similar.

$$\begin{aligned}
&f \cdot g \preccurlyeq_V \mathsf{id}_V \\
\equiv\ &\{\text{the definition of order} \preccurlyeq \text{in}(3)\} \\
&\mathscr{K}(\mathsf{T})f \cdot \mathscr{K}(\mathsf{T})g \cdot \beta(v) \leq_{\mathscr{K}(\mathsf{T})V} \beta(v) \\
\equiv\ &\{(f,g)\text{is a pre-Galois connection}\} \\
&\mathscr{K}(\mathsf{T})g \cdot \beta(v) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})g \cdot \beta(v)
\end{aligned}$$

□

Additionally, we can get the following result:

Lemma 5.5 *If* (f, g) *is a pre-Galois connection between two coalgebras* $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ *and* $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$*, then* f *and* g *are both monotonic with respect to* $\preccurlyeq_U$ *and* $\preccurlyeq_V$*.*

Proof. Suppose (f, g) is a pre-Galois connection. We only need to show that f is monotonic:

$$\begin{aligned}
& f(u) \preccurlyeq_V f(u') \\
\equiv & \{\text{the definition of order } \preccurlyeq \text{ in}(3)\} \\
& \beta(f(u)) \leq_{\mathscr{K}(\mathsf{T})V} \beta(f(u')) \\
\Leftarrow & \{(2.1)\} \\
& \alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \alpha(u') \\
\equiv & \{\text{the definition of order } \preccurlyeq \text{ in}(3)\} \\
& u \preccurlyeq_U u'
\end{aligned}$$

□

Furthermore, we have the result that a pre-Galois connection is also a Galois connection w.r.t. the order infered by (3).

Theorem 5.6 *If (f, g) is a pre-Galois connection between two coalgebras $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$, for the orders on U and V defined by (3), (f, g) is a Galois connection.*

Proof. The result is immmediate from Lemma 5.4 and 5.5. □

Proposition 5.7 *If (f, g) is a pre-Galois connection between two coalgebras $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$, then $f \cdot g \cdot f \overline{\sim} f$ and $g \cdot f \cdot g \overline{\sim} g$.*

Proof. We only need to prove $g \cdot f \cdot g \overline{\sim} g$. The proof for $f \cdot g \cdot f \overline{\sim} f$ is similar. From Lemma 5.4, we know that $f \cdot g \preccurlyeq_V \mathsf{id}_V$. Thus for any $v \in V$, we have

$$\beta(f \cdot g(v)) \leq_{\mathscr{K}(\mathsf{T})V} \beta(v)$$

Since the order is preserved by $\mathscr{K}g$, we get $\alpha(g \cdot f \cdot g(v)) \leq_{\mathscr{K}(\mathsf{T})U} \alpha(g(v))$. Let relation $R \subseteq U \times U$ be defined as $R = \{(g \cdot f \cdot g(v), g(v)) \mid v \in V\}$, then from Definition 4.3, we know that R is a simulation. Similarly we can prove that $g(v) \lesssim g \cdot f \cdot g(v)$. Therefore, $g \cdot f \cdot g \overline{\sim} g$. □

The adjoints in a pre-Galois connection uniquely determine each other when the order $\leq$ is a partial order and $\mathscr{K}(\mathsf{T})$ is a faithful functor.

Proposition 5.8 *If the order $\leq$ is a partial order, and (f, g) and (f, h) are pre-Galois connections between two coalgebras $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$ where $\mathscr{K}(\mathsf{T})$ is faithful, then $g = h$ (similarly for the dual case).*

Proof. Since (f, g) and (f, h) are pre-Galois connections, we have that for any $u \in U$ and $v \in V$,

$$\begin{aligned}
& \alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})g \cdot \beta(v) \\
\equiv & \ \mathscr{K}(\mathsf{T})f \cdot \alpha(u) \leq_{\mathscr{K}(\mathsf{T})V} \beta(v) \\
\equiv & \ \alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})h \cdot \beta(v)
\end{aligned}$$

Thus $\mathscr{K}(\mathsf{T})g \cdot \beta(v) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})h \cdot \beta(v)$ and $\mathscr{K}(\mathsf{T})h \cdot \beta(v) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})g \cdot \beta(v)$. Since $\leq$ is a partial order, we have $\mathscr{K}(\mathsf{T})g = \mathscr{K}(\mathsf{T})h$. Therefore, $g = h$. □

¿From the proof of Proposition 5.8, we can easily derive the following result:

Corollary 5.9 *If $\leq$ is a preorder, and (f, g) and (f, h) are pre-Galois connections between two coalgebras $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$ where $\mathscr{K}(\mathsf{T})$ is faithful, then $g \eqsim h$ (similarly for the dual case).*

Similar like the results for Galois connection between partially ordered sets, we have the following proposition for the adjoints in a pre-Galois connection.

Proposition 5.10 *If the order $\leq$ is a partial order, and (f, g) is a pre-Galois connection between two coalgebras $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$ where $\mathscr{K}(\mathsf{T})$ is faithful, then*

- *f is monic* iff *g is epic* iff *$g \cdot f = \mathsf{id}_U$;*
- *g is monic* iff *f is epic* iff *$f \cdot g = \mathsf{id}_V$.*

Proof. We only need to prove the first item. Suppose (f, h) is another pre-Galois connection between $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$. First, we assume that $g \cdot f = \mathsf{id}_U$. Therefore, if $f \cdot g = f \cdot h$, then $g \cdot f \cdot g = g \cdot f \cdot h$, i.e., $g = h$. On the other hand, if $h \cdot g = k \cdot g$, then $h \cdot g \cdot f = k \cdot g \cdot f$, i.e., $h = k$.

Suppose f is monic, i.e., for any arrows g, h, $f \cdot g = f \cdot h \Rightarrow g = h$. Since (f, g) is a pre-Galois connection, from Lemma 5.4, for any $u \in U$ and $v \in V$, we have $\alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \alpha(g \cdot f(u))$ and $\beta(f \cdot g(v)) \leq_{\mathscr{K}(\mathsf{T})V} \beta(v)$. Thus we can get

$$\alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})g \cdot \mathscr{K}(\mathsf{T})f \cdot \alpha(u) \text{ and } \mathscr{K}(\mathsf{T})f \cdot \mathscr{K}(\mathsf{T})g \cdot \beta(v) \leq_{\mathscr{K}(\mathsf{T})V} \beta(v)$$

In other words,

$$\mathsf{id}_{\mathscr{K}(\mathsf{T})U} \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})g \cdot \mathscr{K}(\mathsf{T})f \text{ and } \mathscr{K}(\mathsf{T})f \cdot \mathscr{K}(\mathsf{T})g \leq_{\mathscr{K}(\mathsf{T})V} \mathsf{id}_{\mathscr{K}(\mathsf{T})V}$$

Since the order is preserved by $\mathscr{K}(\mathsf{T})f$, we have

$$\begin{aligned} & \mathscr{K}(\mathsf{T})f \\ = & \mathscr{K}(\mathsf{T})f \cdot \mathsf{id}_{\mathscr{K}(\mathsf{T})U} \\ \leq_{\mathscr{K}(\mathsf{T})V} & \mathscr{K}(\mathsf{T})f \cdot \mathscr{K}(\mathsf{T})g \cdot \mathscr{K}(\mathsf{T})f \\ \leq_{\mathscr{K}(\mathsf{T})V} & \mathsf{id}_{\mathscr{K}(\mathsf{T})V} \cdot \mathscr{K}(\mathsf{T})f \\ = & \mathscr{K}(\mathsf{T})f \end{aligned}$$

Therefore, we have $\mathscr{K}(\mathsf{T})f \cdot \mathscr{K}(\mathsf{T})g \cdot \mathscr{K}(\mathsf{T})f = \mathscr{K}\mathsf{T}f$. Because $\mathscr{K}(\mathsf{T})$ is faithful, $f \cdot (g \cdot f) = f = f \cdot \mathsf{id}_U$. Thus $g \cdot f = \mathsf{id}_U$.

Similarly, if g is epic, we can also derive the result that $g \cdot f = \mathsf{id}_U$. □

Corollary 5.11 *If $\leq$ is a preorder, and (f, g) is a pre-Galois connection between two coalgebras $(U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $(V, \beta : V \to \mathscr{K}(\mathsf{T})V)$ where $\mathscr{K}(\mathsf{T})$ is faithful, then*

- f *is monic* $\Rightarrow g \cdot f \eqsim \mathsf{id}_U$;
- g *is monic* $\Rightarrow f \cdot g \eqsim \mathsf{id}_V$;
- f *is epic* $\Rightarrow f \cdot g \eqsim \mathsf{id}_V$;
- g *is epic* $\Rightarrow g \cdot f \eqsim \mathsf{id}_U$.

Proof. Similar to the proof for Proposition 5.10. □

Suppose that we have two coalgebras $p = (U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $q = (V, \beta : V \to \mathscr{K}(\mathsf{T})V)$, the obvious way of using a pre-Galois connection to relate them is by constructing the order $\leq_{\mathscr{K}(\mathsf{T})U}$ and $\leq_{\mathscr{K}(\mathsf{T})V}$, and formulating f and g. If we use p and q for the concrete and abstract components respectively, then we say that $f : U \to V$ is the abstraction map and $g : V \to U$ is the concretization map. Note that the arrow does map a state not to a single state, but to a B effect on states. So for every concrete state u, $f(u)$ is its abstract counterpart, and dually, for every abstract state v, $g(v)$ denotes the concrete states that v represents. In other words, $g(v)$ defines the potential refinements of v. In the following section, we show how pre-Galois connections are used in reasoning about refinement of components.

6 Linking Pre-Galois Connection with Refinement

Since simulation is both sound and complete for proving refinement between components, we hope to construct a simulation relationship between components p and q as coalgebras $p = (U, \alpha : U \to \mathscr{K}(\mathsf{T})U)$ and $q = (V, \beta : V \to \mathscr{K}(\mathsf{T})V)$. Given a pre-Galois connection $(f : U \to V, g : V \to U)$, we can extract the relation $R_{(f,g)} \subseteq U \times V$ as follows:

$$R_{(f,g)} = \{(u, v) \mid \mathscr{K}(\mathsf{T})f \cdot \alpha(u) \leq_{\mathscr{K}(\mathsf{T})V} \beta(v)\}$$

or equivalently

$$R_{(f,g)} = \{(u, v) \mid \alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} \mathscr{K}(\mathsf{T})g \cdot \beta(v)\}$$

The intuition behind the relation $R_{(f,g)}$ is that if $(u, v) \in R_{(f,g)}$, then u is a possible refinement of v. Now we are left with the task of proving that $R_{(f,g)}$ can be used as one candidate for proving a simulation between p and q.

Theorem 6.1 *The relation $R_{(f,g)}$ is a simulation.*

Proof. Suppose $(u, v) \in R_{(f,g)}$, let $x = \alpha(u)$ and $y = \mathscr{K}(\mathsf{T})f \cdot \alpha(u)$, then we have $\alpha(u) \leq_{\mathscr{K}(\mathsf{T})U} x$ and $y \leq_{\mathscr{K}(\mathsf{T})V} \beta(v)$. Since $y = \mathscr{K}(\mathsf{T})f \cdot x$, we have

$$\mathscr{K}(\mathsf{T})f \cdot x \leq_{Rel(\mathscr{K}(\mathsf{T}))(R_{(f,g)})} y$$

Therefore, $(x, y) \in Rel(\mathscr{K}(\mathsf{T}))(R_{(f,g)})$. So

$$(\alpha(u), \beta(v)) \in \leq_{\mathscr{K}(\mathsf{T})U} \cdot Rel(\mathscr{K}(\mathsf{T}))(R_{(f,g)}) \cdot \leq_{\mathscr{K}(\mathsf{T})V}$$

and thus $R_{(f,g)}$ is a simulation. □

Corollary 6.2 *If the preorder $\leq$ be equality $=$, then $R_{(f,g)}$ is a bisimulation.*

Proof. Similar to the proof of Theorem 6.1. □

An elegant approach to prove a simulation with a pre-Galois connection is to use a forward (backward) morphism as an intermediary. For given components $p = (U, \alpha)$ and $q = (V, \beta)$, if $h : U \to V$ is proved to be a forward morphism between them, then p is a behavior refinement of q, witnessed by h. We can lift h into a pre-Galois connection (f, g) as follows:

$$f(u) = h(u) \text{ and } g(v) = \overline{U}$$

where for every $u \in_{\mathsf{B}} \overline{U}$, $\beta(h(u)) \leq_{\mathscr{K}(\mathsf{T})V} \beta(v)$.

By this lifting technique, once an arrow h is defined and proved to be a forward morphism, then it is lifted automatically into a pre-Galois connection. And we can easily recover the forward morphism h from a pre-Galois connection (f, g), by defining $h = f$.

7 Conclusion

In this paper the notion of pre-Galois connection between coalgebras in the Kleisli category has been introduced. We rebuild the coalgebraic model for state-based components in [2,13] in the Kleisli category, which goes one step higher on the "generic" ladder, because it unifies the behavior model and transition types into one functor over the Kleisli category instead of using one monad and one functor for representing them as in [2,13] respectively. The refinement theory for generic state-based components in [13,14] is re-examined for coalgebras in the Kleisli category. We defined the notion of pre-Galois connection based on the refinement preorder over the functors, and proved a series of properties for pre-Galois connection. This concept provides a powerful tool in witnessing refinement relation between state-based component.

In terms of future work, what we would like to do in the next step is to apply pre-Galois connection to some refinement examples. Another challange to this work is to go deeper into the concept itself. For example, one issue we intend to study is to establish the theorems that predict the existence of the adjoints in a pre-Galois connection, another interesting problem is the completeness of pre-Galois connections for refinement, i.e., if we can build a pre-Galois connection for every refinement relation.

Acknowledgement

The work reported in this paper is supported by the National Natural Science Foundation of China under Grant No. 60473056, and a grant from the GLANCE funding program of the Dutch National Organization for Scientific Research (NWO),

through project CooPer (600.643.000.05N12). It is a pleasure to acknowledge Luis S. Barbosa for motivating discussions. The author is also indebted to Prof. Jan Rutten, Clemens Kupke and Helle H. Hansen for helpful suggestions.

References

[1] Luís Soares Barbosa. *Components as Coalgebras.* PhD thesis, Universidade do Minho, Braga, Portugal, 2001.

[2] Luís Soares Barbosa. Towards a Calculus of State-based Software Components. *Journal of Universal Computer Science*, 9(8):891–909, August 2003.

[3] Luís Soares Barbosa and Josè Nuno Fonseca de Oliveira. State-based components made generic. In H. Peter Gumm, editor, *Elect. Notes in Theor. Comp. Sci. (CMCS'03 - Workshop on Coalgebraic Methods in Computer Science)*, volume 82.1, Warsaw, April 2003.

[4] Luís Soares Barbosa and Josè Nuno Fonseca de Oliveira. Transposing Partial Coalgebras: An exercise on coalgebraic refinement. *Theoretical Computer Science*, 365(1-2):2–22, 2006.

[5] Luís Soares Barbosa, Sun Meng, Bernhard K. Aichernig, and Nuno Rodrigues. On the semantics of componentware: a coalgebraic perspective. In Jifeng He and Zhiming Liu, editors, *Mathematical Frameworks for Component Software.- Models for Analysis and Synthesis*, chapter 3. World Scientific, 2006.

[6] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs. In *Proceedings of 4th ACM Symp. on Principles of Programming Languages*, pages 238–252. ACM Press, 1977.

[7] Ichiro Hasuo. Generic Forward and Backward Simulations. In C. Baier and H. Hermanns, editors, *CONCUR 2006*, volume 4137 of *LNCS*, pages 406–420. Springer, 2006.

[8] Charles Antony Richard Hoare and Jifeng He. *Unifying Theories of Programming.* Prentice Hall International, 1998.

[9] Bart Jacobs. Exercises in coalgebraic specification. In R. Backhouse, R. Crole, and J. Gibbons, editors, *Algebraic and Coalgebraic Methods in the Mathematics of Program Construction*, volume 2297 of *LNCS*, pages 237–280. Springer, 2002.

[10] Bart Jacobs and Jesse Hughes. Simulations in coalgebra. In H. Peter Gumm, editor, *Elect. Notes in Theor. Comp. Sci. (CMCS'03 - Workshop on Coalgebraic Methods in Computer Science)*, volume 82, pages 245–263, Warsaw, April 2003.

[11] A. Melton, D. A. Schmidt, and G. E. Strecker. Galois connections and computer science applications. In *Category Theory and Computer Programming*, volume 240 of *LNCS*, pages 299–312. Springer-Verlag, 1985.

[12] Sun Meng, Luís S. Barbosa, and Zhang Naixiao. On Refinement of Software Architectures. In *Proceedings of ICTAC'05*, volume 3722 of *LNCS*, pages 482–497. Springer, 2005.

[13] Sun Meng and Luís Soares Barbosa. On Refinement of Generic State-based Software Components. In C. Rattray, S. Maharaj, and C. Shankland, editors, *Algebraic Methodology And Software Technology, 10th International Conference, AMAST'04, Proceedings*, volume 3116 of *LNCS*, pages 506–520. Springer, 2004.

[14] Sun Meng and Luís Soares Barbosa. Components as Coalgebras: the Refinement Dimension. *Theoretical Computer Science*, 351(2):276–294, 2006.

[15] Oystein Ore. Galois connexions. *Trans. Amer. Math. Soc.*, 55:493–513, 1944.

[16] Jan Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249:3–80, 2000.

[17] Clemens Szyperski, Dominik Gruntz, and Stephan Murer. *Component Software – Beyond Object-Oriented Programming, Second Edition.* Publishing House of Electronics Industry, 2003.