



Centrum voor Wiskunde en Informatica

Automata, power series, and coinduction: taking input derivatives seriously (extended abstract)

J.J.M.M. Rutten

Software Engineering (SEN)

**SEN-R9901 January 31, 1999**

Report SEN-R9901  
ISSN 1386-369X

CWI  
P.O. Box 94079  
1090 GB Amsterdam  
The Netherlands

CWI is the National Research Institute for Mathematics and Computer Science. CWI is part of the Stichting Mathematisch Centrum (SMC), the Dutch foundation for promotion of mathematics and computer science and their applications.

SMC is sponsored by the Netherlands Organization for Scientific Research (NWO). CWI is a member of ERCIM, the European Research Consortium for Informatics and Mathematics.

Copyright © Stichting Mathematisch Centrum  
P.O. Box 94079, 1090 GB Amsterdam (NL)  
Kruislaan 413, 1098 SJ Amsterdam (NL)  
Telephone +31 20 592 9333  
Telefax +31 20 592 4199

# Automata, Power Series, and Coinduction: taking input derivatives seriously (extended abstract)

J.J.M.M. Rutten

CWI

*P.O. Box 94079, 1090 GB Amsterdam, The Netherlands\**

## ABSTRACT

Formal power series, which are functions from the set of words over an alphabet  $A$  to a semiring  $k$ , are viewed coalgebraically. In summary, this amounts to supplying the set of all power series with a deterministic automaton structure, which has the universal property of being final. Finality then forms the basis for both definitions and proofs by coinduction, the coalgebraic counterpart of induction. Coinductive definitions of operators on power series take the shape of what we have called behavioural differential equations, after Brzozowski's notion of input derivative, and include many classical differential equations for analytic functions. The use of behavioural differential equations leads, amongst others, to easy definitions of and proofs about both existing and new operators on power series, as well as to the construction of finite (syntactic) nondeterministic automata, implementing them.

*1991 Mathematics Subject Classification:* 68Q10, 68Q55

*1991 Computing Reviews Classification System:* D.3, F.1, F.3

*Keywords & Phrases:* Coalgebra, automaton, coinduction, formal power series, differential equation, input derivative

---

\*Email: [janr@cw.nl](mailto:janr@cw.nl), URL: [www.cwi.nl/~janr](http://www.cwi.nl/~janr).

# Contents

1	Introduction	3
2	Preliminaries	4
3	The automaton of formal power series	5
4	Behavioural differential equations	5
5	Proofs by coinduction	7
6	Shuffle inverse	7
7	Rational series	8
8	Nondeterministic automata	8
9	Recognizability	9
10	An example in the max-plus semiring: task resource systems	11
11	Discussion	12



As another example of our approach, differential equations will be used to solve the well-known problem of task-resource systems (in Section 10).

And so we see the following general scheme emerging: (a) (rational) behaviour is *specified* by differential equations, which often can be solved in a canonical way, giving rise to (b) (finite) nondeterministic automata that (efficiently) *implement* the specified behaviour. We see (a) and (b) as the two main contributions of our work.

*Related work:* The perspective of the present paper is essentially coalgebraic, and generalizes [Rut98], which deals with languages and regular expressions. Our way of solving differential equations is essentially based on the *processes as terms* methodology, used in [RT94]. The notion of input derivative of formal power series, generalizes Brzozowski's original definition for regular expressions [Brz64, Con71]. Its relation with *function derivatives*  $f'$  of functions  $f$  on  $\mathbb{R}$  will be explained by invoking an example from [PE98], where a coinductive treatment of analytic functions in terms of their Taylor expansions is given. Our present theory generalizes the settings of [Rut98]:  $k = \mathbb{B}$  and  $A$  is arbitrary, and [PE98]:  $k = \mathbb{R}$  and  $A = \{X\}$ , since we are dealing with formal power series in many non-commutative variables ( $A$  is arbitrary) over any semiring ( $k$  is arbitrary). Moreover, we are dealing with higher-order, behavioural differential equations (such as the one for  $\parallel$  above). Although it is well known that rational series can be finitely represented (see [BR88], which has been our main reference on formal power series), also the syntactic construction of  $k$ -nondeterministic automata from their defining differential equations is to the best of our knowledge new.

*Acknowledgements:* I am grateful to Maurice Nivat, who offered me the opportunity to present a preliminary version of this paper at the University of Paris VII.

## 2 Preliminaries

We briefly recall the definitions of semiring and formal power series, and give a coalgebraic presentation of the notion of deterministic automaton.

*Semirings:* A semiring is something like a ring without subtraction. More formally, a semiring  $k = \langle k, +, \times, 0, 1 \rangle$  consists of a set  $k$  together with two binary operations  $+$  and  $\times$  (sum and product) and two constants  $0$  and  $1$ , such that  $(k, +, 0)$  is a commutative monoid with  $0$  as identity;  $(k, \times, 1)$  is a monoid with  $1$  as identity; product is distributive with respect to sum; and  $0x = x0 = 0$ , all  $x \in k$  (writing  $xy$  for  $x \times y$ ). The following semirings will occur in examples in the paper: the Boolean semiring  $\mathbb{B} = (\{0, 1\}, \vee, \wedge, 0, 1)$ , the reals  $\mathbb{R} = (\mathbb{R}, +, \times, 0, 1)$ , and the max-plus semiring  $\mathbb{R}_{\max} = ([-\infty, \infty), \max, +, -\infty, 0)$ . Note that both  $\mathbb{B}$  and  $\mathbb{R}_{\max}$  are *idempotent* semirings in that they satisfy  $x + x = x$ .

*Words:* Let  $A$  be a possibly infinite set and let  $A^*$  be the set of all finite words over  $A$ . Prefixing a word  $w$  in  $A^*$  with a letter  $a$  in  $A$  is denoted by  $aw$ . Concatenation of words  $w$  and  $w'$  is denoted by  $ww'$ . Let  $\varepsilon$  denote the empty word.

*Formal power series:* A *formal power series* is a function  $\sigma : A^* \rightarrow k$ . The set of all series is denoted by  $k\langle\langle A \rangle\rangle$ . A series  $\sigma$  assigns to each finite word  $w \in A^*$  a *coefficient*  $\sigma(w)$  in  $k$ , which may be interpreted as the *multiplicity* with which the word  $w$  occurs in  $\sigma$ . These multiplicities may have different interpretations, depending on the semiring  $k$ . If  $k = \mathbb{B}$  then  $\sigma(w)$  is either 1 or 0, indicating whether or not  $w$  belongs to  $\sigma$ , which in this case simply is a set of words. In other cases, the elements of  $A$  are best viewed as (formal non-commutative) *variables*. A basic but important example is  $A = \{X\}$  and  $k = \mathbb{R}$ , when one gets the usual power series. As usual,  $k$  and  $A$  can be considered as subsets of  $k\langle\langle A \rangle\rangle$ , by taking  $x \in k$  as the function  $x : A^* \rightarrow k$  with  $x(\varepsilon) = x$ , and 0 everywhere else; similarly,  $a \in A$  is identified with  $a : A^* \rightarrow k$ , defined by  $a(a) = 1$ , and 0 otherwise.

*Deterministic automata:* Let  $A$  be a possibly infinite set and let  $k$  be a semiring. A *deterministic automaton* (or Moore machine) with inputs in  $A$  and outputs in  $k$  is a pair  $S = (S, \langle o_S, t_S \rangle)$  consisting of a set  $S$  of *states*, and a pair of functions: an *output function*  $o_S : S \rightarrow k$ , and a *transition function*  $t_S : S \rightarrow S^A$ . Here  $S^A$  is the set of all functions from  $A$  to  $S$ . The transition function  $t_S$  assigns to a state  $s$  a function  $t_S(s) : A \rightarrow S$ , which specifies the state  $t_S(s)(a)$

that is reached after an input symbol  $a$  has been consumed. We shall sometimes write  $s \xrightarrow{x}$  for  $o_S(s) = x$  and  $s \xrightarrow{a} s'$  for  $t_S(s)(a) = s'$ . Also we shall simply write  $o$  and  $t$  whenever the automaton  $S$  is clear from the context. A *homomorphism* between automata  $S = (S, \langle o, t \rangle)$  and  $S' = (S', \langle o', t' \rangle)$  is any function  $f : S \rightarrow S'$  such that for all  $s$  in  $S$ ,  $o(s) = o'(f(s))$  and, for all  $a$  in  $A$ ,  $f(t(s)(a)) = t'(f(s))(a)$ . A subset  $i : S' \subseteq S$  of an automaton  $S$  is a *subautomaton* if  $i$  is a homomorphism. For a state  $s$  in  $S$ ,  $\langle s \rangle$  denotes the subautomaton *generated* by  $s$ . Homomorphisms map subautomata to subautomata. A relation  $R \subseteq S \times S'$  is a *bisimulation* between two automata  $S$  and  $S'$  if, for all  $s$  in  $S$ ,  $s'$  in  $S'$ , and  $a$  in  $A$ : if  $s R s'$  then  $o(s) = o'(s')$  and  $t(s)(a) R t'(s')(a)$ . If there exists a bisimulation (between  $S$  and itself) containing  $s, s' \in S$ , then we write  $s \sim s'$  ( $s$  and  $s'$  are *bisimilar*). Bisimilarity itself is a bisimulation relation and an equivalence relation.

### 3 The automaton of formal power series

The set  $k\langle\langle A \rangle\rangle$  of formal power series is turned into a deterministic automaton with inputs in  $A$  and outputs in  $k$ , having the universal property of being *final* and satisfying a principle of *coinduction*.

For an input  $a$  in  $A$ , the *input derivative*  $\sigma_a$  (or  $\partial\sigma/\partial a$  or  $a^{-1}\sigma$ ) of a series  $\sigma : A^* \rightarrow k$  is defined by  $\sigma_a(w) = \sigma(aw)$ , for  $w \in A^*$ . The *constant part* (or output) of a series  $\sigma$  is defined by  $\sigma(\varepsilon)$ . These notions determine an automaton structure  $k\langle\langle A \rangle\rangle = (k\langle\langle A \rangle\rangle, \langle o_k, t_k \rangle)$ , defined, for  $\sigma \in k\langle\langle A \rangle\rangle$  and  $a \in A$ , by  $o_k(\sigma) = \sigma(\varepsilon)$  and  $t_k(\sigma)(a) = \sigma_a$ .

**Theorem 3.1** *The automaton  $k\langle\langle A \rangle\rangle$  satisfies the principle of (1) coinduction: for all series  $\sigma$  and  $\tau$  in  $k\langle\langle A \rangle\rangle$ , if  $\sigma \sim \tau$  then  $\sigma = \tau$ . Moreover,  $k\langle\langle A \rangle\rangle$  is (2) final: for any automaton  $S$  there exists a unique homomorphism  $l : S \rightarrow k\langle\langle A \rangle\rangle$ , satisfying: for  $s, s' \in S$ ,  $s \sim s'$  iff  $l(s) = l(s')$ .*

The series  $l(s)$  is called the *behaviour* of the state  $s$  of the automaton  $S$ . It assigns to any word  $w \in A^*$  the output  $o(s')$  of the state  $s'$  that is reached from  $s$  after reading  $w$ . We say that  $s$  *represents* the series  $l(s)$ , and also that  $l(s)$  is the series *accepted* by the state  $s$ . Part (1) is easily proved by induction on the length of words  $w \in A^*$ , and implies (2). The proof also follows from general coalgebraic reasoning (see, e.g., [Rut96]), and does not depend on the semiring structure of  $k$ .

Coinduction serves as a *proof* principle: in order to show  $\sigma = \tau$ , it is sufficient to establish the existence of a bisimulation relation  $R$  with  $\sigma R \tau$ . The proof principle will be illustrated in some detail in Section 5. Finality will be used as a *coinductive definition* principle (for instance, in Section 4).

The relation between derivatives  $f'$  of functions  $f$  on  $\mathbb{R}$ , and input derivatives  $\sigma_a$  of formal power series  $\sigma$  is explained by the following example on analytic functions, taken from [PE98]. Let  $k = \mathbb{R}$  and  $A = \{X\}$ . Thus  $\mathbb{R}\langle\langle X \rangle\rangle = \mathbb{R}\langle\langle X \rangle\rangle^* \cong \mathbb{R}^\omega$ , where  $\omega = \{0, 1, \dots\}$  is the set of natural numbers. In other words, formal power series are now infinite sequences, also called streams, of real numbers. Consider the set  $\mathcal{A}$  of functions that are analytic in 0: the  $n$ -th derivative  $f^{(n)}(0)$  exists, for all  $n \geq 0$ . Following [PE98],  $\mathcal{A}$  can be turned into an automaton by defining  $o_{\mathcal{A}} : \mathcal{A} \rightarrow \mathbb{R}$  and  $t_{\mathcal{A}} : \mathcal{A} \rightarrow \mathcal{A}$  (identifying  $\mathcal{A}\langle\langle X \rangle\rangle \cong \mathcal{A}$ ) by  $o_{\mathcal{A}}(f) = f(0)$  and  $t_{\mathcal{A}}(f) = f'$ . Because  $\mathbb{R}\langle\langle X \rangle\rangle$  is a final automaton, there exists a unique homomorphism  $l : \mathcal{A} \rightarrow \mathbb{R}\langle\langle X \rangle\rangle$ , which maps a function  $f$  to the series of its *Taylor* coefficients:  $l(f) = (f(0), f'(0), f''(0), \dots)$ . Because  $l$  is a homomorphism,  $l(f)_X = l(f')$ . In words, the input derivative of the Taylor series of  $f$  is equal to the Taylor series of the derivative of  $f$ .

### 4 Behavioural differential equations

A number of operators on formal power series will be defined by *coinduction*. Similar to the way one can define, for instance, a function of exponentiation  $exp : \mathbb{R} \rightarrow \mathbb{R}$  by specifying a differential equation and initial value:  $exp' = exp$  and  $exp(0) = 1$ , coinductive definitions of (elements of and) operators on  $k\langle\langle A \rangle\rangle$  amount to the specification of *behavioural differential equations*. It will be a

consequence of the finality of the automaton  $k\langle\langle A \rangle\rangle$  that the systems of differential equations we shall use, have unique solutions.

Plunging into the matter, the aim of this section is to prove the following theorem.

**Theorem 4.1** *There are unique functions  $+$ ,  $\times$ ,  $(-)^*$ ,  $\parallel$ , and  $(-)^{-1}$ , called sum, product, star, shuffle product, and inverse, satisfying the following (higher-order) behavioural differential equations: For all  $\sigma, \tau \in k\langle\langle A \rangle\rangle$  and  $a \in A$ ,*

differential equation	initial value
$(\sigma + \tau)_a = \sigma_a + \tau_a$	$(\sigma + \tau)(\varepsilon) = \sigma(\varepsilon) + \tau(\varepsilon)$
$(\sigma\tau)_a = \sigma_a \tau + \sigma(\varepsilon)\tau_a$	$(\sigma\tau)(\varepsilon) = \sigma(\varepsilon)\tau(\varepsilon)$
$(\sigma^*)_a = \sigma_a \sigma^*$	$(\sigma^*)(\varepsilon) = 1$
$(\sigma \parallel \tau)_a = (\sigma_a \parallel \tau) + (\sigma \parallel \tau_a)$	$(\sigma \parallel \tau)(\varepsilon) = \sigma(\varepsilon)\tau(\varepsilon)$
$(\sigma^{-1})_a = -\sigma(\varepsilon)^{-1}\sigma_a \sigma^*$	$(\sigma^{-1})(\varepsilon) = \sigma(\varepsilon)^{-1}$

Note that we have written  $\sigma\tau$  for  $\sigma \times \tau$ , and that we use the same symbols for sum and product on  $k\langle\langle A \rangle\rangle$  as for sum and product on  $k$ , respectively. Also note that in the expression  $\sigma(\varepsilon)\tau_a = \sigma(\varepsilon) \times \tau_a$  above, we are interpreting  $\sigma(\varepsilon)$ , which is an element of  $k$ , as an element of  $k\langle\langle A \rangle\rangle$ , following the convention described in Section 2. Observe that in the definition of the initial values, the operators of the semiring structure of  $k$  are used. Finally note that the latter equation assumes  $k$  to be a *ring* rather than a semiring, since it uses subtraction. It is moreover *partial* since it only applies to such  $\sigma$  for which  $\sigma(\varepsilon)$  is invertible in  $k$ . Whenever we shall write  $-\sigma(\varepsilon)^{-1}$ , both conditions will silently be assumed to apply. If either of these conditions does not hold then we put  $(\sigma^{-1})_a = 0$  and  $(\sigma^{-1})(\varepsilon) = 0$  (simply in order to keep all functions total).

**Proof of Theorem 4.1:** Let the set  $\mathcal{E}$  of *expressions* be given by the following syntax:

$$E ::= \underline{\sigma} \mid E + F \mid EF \mid E^* \mid (E \parallel F) \mid E^{-1}$$

where we write  $EF$  rather than  $E \times F$ , and where for every series  $\sigma$  in  $k\langle\langle A \rangle\rangle$  a *symbol*  $\underline{\sigma}$  is included in  $\mathcal{E}$ . The set  $\mathcal{E}$  is next supplied with an automaton structure  $(\mathcal{E}, \langle o_{\mathcal{E}}, t_{\mathcal{E}} \rangle)$ . The functions  $o_{\mathcal{E}} : \mathcal{E} \rightarrow k$  and  $t_{\mathcal{E}} : \mathcal{E} \rightarrow \mathcal{E}^A$  are defined by induction on the structure of expressions, using the automaton structure of  $k\langle\langle A \rangle\rangle$  for the symbols  $\underline{\sigma}$ , and following the structure of the differential equations in Theorem 4.1 for the operators:

$$\begin{aligned} o_{\mathcal{E}}(\underline{\sigma}) &= \sigma(\varepsilon), & o_{\mathcal{E}}(E + F) &= o_{\mathcal{E}}(E) + o_{\mathcal{E}}(F), & o_{\mathcal{E}}(EF) &= o_{\mathcal{E}}(E \parallel F) = o_{\mathcal{E}}(E)o_{\mathcal{E}}(F), \\ o_{\mathcal{E}}(E^*) &= 1, & o_{\mathcal{E}}(E^{-1}) &= o_{\mathcal{E}}(E)^{-1} \end{aligned}$$

(The latter expression should be interpreted as 0 if the inverse in  $k$  does not exist.) Writing  $E_a$  for  $t_{\mathcal{E}}(E)(a)$ , the function  $t_{\mathcal{E}} : \mathcal{E} \rightarrow \mathcal{E}^A$  is given by the following clauses:

$$\begin{aligned} (\underline{\sigma})_a &= \underline{\sigma}_a, & (E + F)_a &= E_a + F_a, & (EF)_a &= E_a F + o_{\mathcal{E}}(E)F_a, & (E^*)_a &= E_a E^* \\ (E \parallel F)_a &= (E_a \parallel F) + (E \parallel F_a), & (E^{-1})_a &= -o_{\mathcal{E}}(E)^{-1}E_a E^* \end{aligned}$$

(Read  $\underline{\sigma}$  for the last expression whenever  $-o_{\mathcal{E}}(E)^{-1}$  is undefined.) Because  $\mathcal{E}$  now has been turned into an automaton  $(\mathcal{E}, \langle o_{\mathcal{E}}, t_{\mathcal{E}} \rangle)$ , and because  $k\langle\langle A \rangle\rangle$  is a final automaton, there exists, by Theorem 3.1, a unique homomorphism  $l : \mathcal{E} \rightarrow k\langle\langle A \rangle\rangle$ , which assigns to each expression  $E$  the formal power series  $l(E)$  it represents. It can be used to define the operators on  $k\langle\langle A \rangle\rangle$  that we are looking for:

$$\sigma + \tau = l(\underline{\sigma} + \underline{\tau}), \quad \sigma\tau = l(\underline{\sigma}\underline{\tau}), \quad \sigma^* = l(\underline{\sigma}^*), \quad \sigma \parallel \tau = l(\underline{\sigma} \parallel \underline{\tau}), \quad \sigma^{-1} = l(\underline{\sigma}^{-1})$$

(Note that the symbols for the operators on  $k\langle\langle A \rangle\rangle$  are the same as the syntactic operators. The type will always be clear from the context.) One can show that  $l(\underline{\sigma}) = \sigma$  and that  $l$  is compositional, e.g.,  $l(EF) = l(E)l(F)$ , using the principle of coinduction (Theorem 3.1) and the fact that bisimilarity on  $\mathcal{E}$  is a congruence relation (e.g., if  $E \sim E'$  and  $F \sim F'$  then  $EF \sim E'F'$ ). For instance, the first statement follows by coinduction from the fact that  $\{l(\underline{\sigma}), \sigma \mid \sigma \in k\langle\langle A \rangle\rangle\}$  is a bisimulation relation on  $k\langle\langle A \rangle\rangle$ . One can now readily prove that the operators we have

defined are solutions of their defining differential equations, using the fact that  $l$  is a compositional homomorphism. Uniqueness of these solutions follows from the uniqueness of  $l$ .  $\square$

Either by coinduction or, alternatively, using the uniqueness part of Theorem 4.1, one can prove that the above coinductive definitions of the operators on  $k\langle\langle A \rangle\rangle$  coincide with the usual ones. For instance,  $(\sigma^*)(w) = \sum_{n \geq 0} \sigma^n(w)$ , if  $\sigma(\varepsilon) = 0$ .

Given the correspondence between derivative and input derivative, mentioned at the end of Section 3, one can easily show that the Taylor series of the *product* of analytic functions equals the *shuffle* product of their corresponding Taylor series:  $l(fg) = l(f) \parallel l(g)$ , where  $(fg)(x) = f(x)g(x)$ , as usual. (A proof (by coinduction) is easy, using the fact that  $(fg)' = f'g + fg'$ , which is of the same shape as the defining differential equation for  $\parallel$ .) This fact will be used in the definition of the shuffle inverse in Section 6. The correspondence between derivative and input derivative also shows that many classical differential equations, such as the example of *exp* mentioned at the beginning of this section, have a unique corresponding behavioural differential equation. For *exp*, this is the equation  $(e)_X = e$  (with initial value  $e(\varepsilon) = 1$ ), which determines a unique series  $e = (1, 1, 1, \dots)$  in  $\mathbb{R}\langle\langle X \rangle\rangle$ , that is, the Taylor series of *exp*.

## 5 Proofs by coinduction

The use of coinduction is illustrated by proving some of the following familiar laws:

(1) $1 + \sigma\sigma^* = \sigma^*$ , if $\sigma(\varepsilon) = 0$	(6) $\sigma = \sigma(\varepsilon) + \sum_{a \in A} a\sigma_a$
(2) $\sigma = \tau\sigma + \rho \Rightarrow \sigma = \tau^*\rho$ , if $\tau(\varepsilon) = 0$	(7) $\sigma \parallel (\tau + \rho) = (\sigma \parallel \tau) + (\sigma \parallel \rho)$
(3) $(\sigma + \tau)^* = \sigma^*(\tau\sigma^*)^*$ , if $\tau(\varepsilon) = 0$	(8) $\sigma \parallel (\tau \parallel \rho) = (\sigma \parallel \tau) \parallel \rho$
(4) $(\sigma + \tau)^* = (\sigma^*\tau)^*\sigma^*$ , if $\tau(\varepsilon) = 0$	(9) $\sigma\sigma^{-1} = 1$
(5) $\sigma^* = (1 + \sigma)^*$	(10) $\sigma^{-1}\sigma = 1$

Coinductive proofs of equalities such as  $\sigma_0 = \tau_0$  always proceed in the same way, by defining, in stages, a bisimulation relation  $R$  containing  $\langle\sigma_0, \tau_0\rangle$ . The first pair to be included in  $R$  is  $\langle\sigma_0, \tau_0\rangle$ . Next the following step is repeated until it does not yield any new pairs: the  $a$ -derivatives of the pairs  $\langle\sigma, \tau\rangle$  already in  $R$  are computed—for the operators, these are precisely given by the defining differential equations—and the resulting pairs  $\langle\sigma_a, \tau_a\rangle$  are added to  $R$ . When adding a pair  $\langle\sigma, \tau\rangle$  to  $R$ , at any stage of its construction, we should check that the constant parts are equal:  $\sigma(\varepsilon) = \tau(\varepsilon)$ . If this does not hold, the procedure aborts, and we conclude  $\sigma \neq \tau$ . Otherwise, the relation  $R$  that is thus obtained is by construction a bisimulation, and  $\sigma_0 = \tau_0$  follows by coinduction. For instance, (1) follows by coinduction from the fact that

$$R_1 = \{\langle 1 + \sigma\sigma^*, \sigma^* \rangle \mid \sigma \in k\langle\langle A \rangle\rangle, \sigma(\varepsilon) = 0\} \cup \{\langle \sigma, \sigma \rangle \mid \sigma \in k\langle\langle A \rangle\rangle\}$$

is a bisimulation relation on  $k\langle\langle A \rangle\rangle$ : if  $\sigma(\varepsilon) = 0$  then  $(1 + \sigma\sigma^*)(\varepsilon) = 1 = \sigma^*(\varepsilon)$ ; moreover,  $\langle(1 + \sigma\sigma^*)_a, (\sigma^*)_a\rangle = \langle 0 + \sigma_a\sigma^* + 0\sigma_a\sigma^*, \sigma_a\sigma^* \rangle = \langle \sigma_a\sigma^*, \sigma_a\sigma^* \rangle$ , which is in  $R_1$  again. Similarly,

$$R_2 = \{\langle \alpha\sigma + \beta, \alpha\tau^*\rho + \beta \rangle \mid \alpha, \beta \in k\langle\langle A \rangle\rangle\}$$

is a bisimulation relation, for  $\sigma, \tau, \rho$  with  $\sigma = \tau\sigma + \rho$  and  $\tau(\varepsilon) = 0$ , implying (2) by coinduction. All the other laws are proved similarly. To mention one last example, let  $R_8$  be the smallest relation on  $k\langle\langle A \rangle\rangle$  such that  $\langle\sigma \parallel (\tau \parallel \rho), (\sigma \parallel \tau) \parallel \rho\rangle \in R_8$ , for all  $\sigma, \tau, \rho \in k\langle\langle A \rangle\rangle$ , and such that  $\langle\sigma_1, \tau_1\rangle, \langle\sigma_2, \tau_2\rangle \in R_8$  implies  $\langle\sigma_1 + \sigma_2, \tau_1 + \tau_2\rangle \in R_8$ . It is straightforward to prove that  $R_8$  is a bisimulation (using (7)), whence (8) by coinduction. Note that for none of the proofs above, additional structure had to be introduced. Notably, there is no need of turning  $k\langle\langle A \rangle\rangle$  into a topological semiring, which is what is usually done (see, for instance, [BR88, Lm 4.1, p.5]).

## 6 Shuffle inverse

The correspondence between the product of two functions on the reals and the *shuffle* product of their corresponding Taylor series (mentioned at the end of Section 4), suggests the following

definition of an operator that acts as a quotient with respect to the shuffle product. Recalling the familiar quotient law for derivatives:  $(f^{-1})' = -f'(1/f^2) = -f'(ff)^{-1}$ , consider the following behavioural differential equation:  $(\sigma_{-1})_a = -\sigma_a \parallel (\sigma \parallel \sigma)_{-1}$  with initial value  $(\sigma_{-1})(\varepsilon) = \sigma(\varepsilon)^{-1}$ . Note that we write  $\sigma_{-1}$  rather than  $\sigma^{-1}$ , since the latter notation is used, in Section 4, for the inverse with respect to multiplication. Further note that  $k$  is assumed to be a ring and that the above equation only applies to such  $\sigma$  for which  $\sigma(\varepsilon)$  is invertible in  $k$ . The above equation has a unique solution, which can be proved along the same lines as Theorem 4.1. Assuming that  $k$  is a ring, the following equalities hold for all  $\sigma \in k\langle\langle A \rangle\rangle$  for which  $\sigma(\varepsilon)$  is invertible in  $k$ , showing that the shuffle inverse behaves as intended:  $\sigma \parallel \sigma_{-1} = 1$  and  $(\sigma_{-1})_{-1} = \sigma$ . This can be readily proved by coinduction. It is not immediately obvious how this operator could be defined without coinduction. For now, we are satisfied with the fact that it has been possible to define it at all. Its use for the theory of power series is to be studied further (see also Section 11).

## 7 Rational series

We recall the notion of rational series, and illustrate the need of nondeterministic automata with multiplicities in  $k$  in order to obtain finite representations for them.

Let the set  $\mathcal{R}$  of *regular expressions* be given by the following syntax:

$$E ::= x \in k \mid a \in A \mid E + F \mid EF \mid E^*$$

Note that, for convenience, we write  $x$  and  $a$  rather than  $\underline{x}$  and  $\underline{a}$  and that, under the embedding of  $k$  and  $A$  in  $k\langle\langle A \rangle\rangle$ ,  $\mathcal{R}$  is a subset of the set of expressions  $\mathcal{E}$ , introduced in (the proof of) Theorem 4.1. A series  $\sigma$  is called *rational* if there exists a regular expression  $E$  with  $\sigma = l(E)$ , where  $l : \mathcal{E} \rightarrow k\langle\langle A \rangle\rangle$  is the unique homomorphism of Theorem 4.1. Because  $l$  is compositional, a series is rational iff it is contained in the smallest subset of  $k\langle\langle A \rangle\rangle$  that contains  $k$  and  $A$  (viewed as subsets of  $k\langle\langle A \rangle\rangle$ ) and that is closed under the operators of sum, product, and star.

In order to see whether a series  $\sigma$  is rational or not, it is sufficient to look at the subautomaton  $\langle\sigma\rangle$  of  $k\langle\langle A \rangle\rangle$  that it generates. This subautomaton is generally infinite: for instance,  $(xa)^*$ , with  $x$  in  $k$  and  $a$  in  $A$ , gives rise to the following transitions in  $k\langle\langle A \rangle\rangle$  (note that when  $x$  is interpreted as an element of  $k\langle\langle A \rangle\rangle$ —cf. Section 2—we have  $(x)_a = 0$  and  $x(\varepsilon) = x$ ):

$$(xa)^* \xrightarrow{a} x(xa)^* \xrightarrow{a} x^2(xa)^* \xrightarrow{a} \dots$$

whence  $\langle(xa)^*\rangle = \{x^n(xa)^* \mid n \geq 0\}$ . At the same time, this example is typical in the sense that the generated subautomaton of a rational series is characterized by the property that it is *finitely generated*. We shall not prove this in the present paper, but in Section 9, we shall see another, truly finitary characterization of rational power series (from which this property easily follows). There it will be shown that a rational series is recognized by a finite *nondeterministic* automaton with multiplicities in the semiring  $k$ . For instance, the above transition sequence can be captured by one single transition of the form  $(xa)^* \xrightarrow{a|x} (xa)^*$ , which will allow the construction of a finite automaton for  $(xa)^*$ . The point will not be so much the existence of such finite representations for rational series (which is classical, see [BR88]), but rather the (syntactic) way in which they are constructed.

## 8 Nondeterministic automata

In order to give a truly finite representation of rational series, this section introduces (a coalgebraic formulation of) nondeterministic automata and gives a coinductive definition of their behaviour. In Section 9, *finite* nondeterministic automata for rational series will be constructed.

A *k-nondeterministic automaton* (*nd-automaton* for short, also called *k-transducer*) with inputs in  $A$  and outputs in  $k$  is a pair  $S = (S, \langle o, t \rangle)$  consisting of a set  $S$  of *states*, and a pair of functions:

an *output function*  $o : S \rightarrow k$ , and a *nondeterministic transition function*  $t : S \rightarrow k(S)^A$ . Here  $k(S)^A$  is the set of all functions from  $A$  to  $k(S)$ , which at its turn is defined by

$$k(S) = \{ \phi : S \rightarrow k \mid \text{supp}(\phi) \text{ is finite} \}$$

where  $\text{supp}(\phi) = \{s \in S \mid \phi(s) \neq 0\}$  is the *support* of  $\phi$ . The observation function  $o$  assigns to each state  $s$  in  $S$  a multiplicity  $o(s)$  in  $k$ . The transition function  $t$  assigns to a state  $s$  in  $S$  a function  $t(s) : A \rightarrow k(S)$ , which specifies for any  $a$  in  $A$  a function  $t(s)(a) \in k(S)$ . Such a function can be viewed as a kind of nondeterministic or distributed state, and specifies for any state  $s'$  in  $S$  a multiplicity  $t(s)(a)(s')$  in  $k$  with which the  $a$ -transition from  $s$  to  $s'$  occurs. We shall sometimes write  $s \xrightarrow{a|x} s'$  for  $t(s)(a)(s') = x$  and  $s \xrightarrow{x}$  for  $o(s) = x$ .

The *behaviour* of a state in a nd-automaton, which is again a formal power series, is defined coinductively. To this end, we shall first associate with every nondeterministic automaton  $\langle o, t \rangle : S \rightarrow k \times k(S)^A$  a corresponding *deterministic* automaton. The set of states of the new automaton is given by the set  $k(S)$  (of distributed states) mentioned above. Next the set  $k(S)$  is turned into a deterministic automaton  $(k(S), \langle \hat{o}, \hat{t} \rangle)$ , by defining an observation function  $\hat{o} : k(S) \rightarrow k$  and a deterministic transition function  $\hat{t} : k(S) \rightarrow k(S)^A$ , as follows:

$$\hat{o}(\phi) = \sum_{s \in S} o(s)\phi(s), \quad \hat{t}(\phi)(a)(s) = \sum_{s' \in S} \phi(s') (t(s')(a)(s))$$

Note that both these sums exist because  $\phi$  in  $k(S)$  has finite support. The behaviour of a nd-automaton  $S$  can now be defined in terms of the, by Theorem 3.1, unique homomorphism  $\lambda : k(S) \rightarrow k\langle\langle A \rangle\rangle$ , which assigns to each configuration  $\phi$  in  $k(S)$  the formal power series  $\lambda(\phi)$  it represents. Because of the existence of the obvious inclusion  $\{\cdot\} : S \rightarrow k(S)$ , with  $\{\cdot\}(s') = 1$  if  $s = s'$ , and  $= 0$  otherwise, we have obtained a function  $\lambda \circ \{\cdot\} : S \rightarrow k\langle\langle A \rangle\rangle$ , which is the coinductive definition of the behaviour of  $(S, \langle o, t \rangle)$  that we were after.

The term ‘multiplicity’ used above may have many different interpretations, depending on the semiring  $k$ . If  $k = \mathbb{B}$  then  $\mathbb{B}$ -nondeterministic automata are precisely the classical nondeterministic automata, with  $o : S \rightarrow \mathbb{B}$  specifying which states are terminal (accepting), and where for a state  $s \in S$  and input letter  $a \in A$ ,  $t(s)(a) \in \mathbb{B}(S) \cong \mathcal{P}_f(S)$  gives the (finite) set of possible next states. The construction of a deterministic automaton above amounts in this case exactly to the familiar power set construction.

The following characterization of the behaviour of nd-automata, in terms of transition sequences, can be readily proved by coinduction: For all  $s$  in a nd-automaton  $S$  and all words  $a_0 \cdots a_{m-1} \in A^*$ ,

$$\lambda(\{\cdot\})(a_0 \cdots a_{m-1}) = \sum \{x_0 x_1 \cdots x_{m-1} \mid s = u_0 \xrightarrow{a_0|x_0} u_1 \xrightarrow{a_1|x_1} \cdots \xrightarrow{a_{m-1}|x_{m-1}} u_m \xrightarrow{x}\}$$

This characterization can be sometimes convenient when computing small concrete examples. *Proofs* about nd-automata will all be based on the coinductive definition in terms of  $\lambda \circ \{\cdot\}$ .

## 9 Recognizability

A formal power series  $\sigma \in k\langle\langle A \rangle\rangle$  is *recognizable* if there exists a *finite* nd-automaton  $S$  and a state  $s \in S$  such that  $\lambda(\{\cdot\}) = \sigma$  (with  $\lambda$  as defined in Section 8). The pair  $(S, s)$  is then called a *finite representation* of  $\sigma$ . In this section, we construct a finite representation for any *rational* series  $l(E)$ , with  $E$  a regular expression, thus giving a new proof of the well-known fact that any rational series is recognizable (cf. [BR88]). The representation is *syntactic* in the sense that its state space consists of (regular) expressions.

To this end, the entire set  $\mathcal{R}$  of regular expressions is turned into an (infinite) nondeterministic automaton  $(\mathcal{R}, \langle o_{\mathcal{R}}, t_{\mathcal{R}} \rangle)$ , such that the behaviour of  $E$  is precisely given by  $l(E)$ . As we shall see, the subautomaton  $\mathcal{R}_E \subseteq \mathcal{R}$  generated by  $E$  is finite, giving a finite representation  $(\mathcal{R}_E, E)$  of  $l(E)$ .

The observation function  $o_{\mathcal{R}} : \mathcal{R} \rightarrow k$  is defined by  $o_{\mathcal{R}}(E) = o_{\mathcal{E}}(E)$ , where  $o_{\mathcal{E}}$  is the observation function for expressions, defined in Section 4. The nondeterministic transition function  $t_{\mathcal{R}} : \mathcal{R} \rightarrow k(\mathcal{R})^A$  is defined by induction on the structure of regular expressions, following the shape of the behavioural differential equations of Theorem 4.1:

$$\begin{aligned} t_{\mathcal{R}}(x)(a)(G) &= 0 & t_{\mathcal{R}}(b)(a)(G) &= \begin{cases} 1 & \text{if } b = a \text{ and } G = 1 \\ 0 & \text{otherwise} \end{cases} \\ t_{\mathcal{R}}(E + F)(a)(G) &= t_{\mathcal{R}}(E)(a)(G) + t_{\mathcal{R}}(F)(a)(G) \\ t_{\mathcal{R}}(EF)(a)(G) &= \begin{cases} t_{\mathcal{R}}(E)(a)(E') + o_{\mathcal{R}}(E) t_{\mathcal{R}}(F)(a)(G) & \text{if } G = E'F \\ o_{\mathcal{R}}(E) t_{\mathcal{R}}(F)(a)(G) & \text{otherwise} \end{cases} \\ t_{\mathcal{R}}(E^*)(a)(G) &= \begin{cases} t_{\mathcal{R}}(E)(a)(E') & \text{if } G = E'E^* \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

(Using induction on  $\mathcal{R}$ , one easily proves that for all  $F \in \mathcal{R}$  and  $a \in A$ , the function  $t_{\mathcal{R}}(F)(a)$  is in  $k(\mathcal{R})$ , that is, has finite support.) Applying now the definitions from Section 8, we obtain the following diagram, in which we have also included the homomorphism  $l : \mathcal{E} \rightarrow k\langle\langle A \rangle\rangle$ :

$$\mathcal{R} \xrightarrow{\{\cdot\}} k(\mathcal{R}) \xrightarrow{\lambda} k\langle\langle A \rangle\rangle \xleftarrow{l} \mathcal{E}$$

We can prove by coinduction that  $\lambda(\{E\}) = l(E)$ , for any regular expression  $E \in \mathcal{R}$ , since

$$\{\langle \lambda(\phi), \sum_{E \in \text{supp}(\phi)} \phi(E) l(E) \rangle \mid \phi \in k(\mathcal{R})\}$$

is a bisimulation on  $k\langle\langle A \rangle\rangle$ . Because for a regular expression  $E$  in  $\mathcal{R}$ , the subautomaton  $\mathcal{R}_E$  of  $\mathcal{R}$  generated by  $E$ , is finite (which follows from the definition of the transition function  $t_{\mathcal{R}}$  by induction on the structure of expressions), we have proved the following theorem.

**Theorem 9.1** *For a regular expression  $E$  in  $\mathcal{R}$ ,  $(\mathcal{R}_E, E)$  is a finite representation of  $l(E)$  (hence any rational series is recognizable).  $\square$*

**Example 9.2** Here are three examples of finite representations:

$$\begin{array}{ccc} 7a \xrightarrow{a|7} 1 \xrightarrow{1} & a|7 \begin{array}{c} \curvearrowright \\ (7a)^* \end{array} \xrightarrow{1} & \begin{array}{ccc} (2 + 3X + 7X^2) \xrightarrow{2} & & \\ X|7 \downarrow & \searrow^{X|3} & \\ X & \xrightarrow{X|1} & 1 \xrightarrow{1} \end{array} \end{array}$$

Finite representations for the shuffle product and the inverse (with respect to multiplication) of rational expressions can be obtained in a similar fashion, by extending the above approach. More precisely, one should first extend the definition of  $\mathcal{R}$  as to include expressions of the form

$$E ::= x \in k \mid a \in A \mid E + F \mid EF \mid E^* \mid E \parallel F \mid E^{-1}$$

and next extend the definition of  $o_{\mathcal{R}}$  and  $t_{\mathcal{R}}$ , again by following the defining differential equations from Theorem 4.1. E.g., assuming  $k$  to be a ring, we define  $o_{\mathcal{R}}(E^{-1}) = o_{\mathcal{E}}(E^{-1})$  and

$$t_{\mathcal{R}}(E^{-1})(a)(G) = \begin{cases} -o_{\mathcal{R}}(E)^{-1} t_{\mathcal{R}}(E)(a)(E') & \text{if } G = E'E^{-1} \text{ and } o_{\mathcal{R}}(E) \text{ is invertible in } k \\ 0 & \text{otherwise} \end{cases}$$

Since the proof of Theorem 9.1 can be easily extended, we have thus obtained for a rational series  $\sigma = l(E)$  (with  $E$  a regular expression) a finite representation  $(\mathcal{R}_{E^{-1}}, E^{-1})$  of  $\sigma^{-1}$ . (It is not difficult to see that this implies that the latter series is rational again.)

**Example 9.3** Applying the definitions above to  $E = 2 + 3X + 7X^2$ , the two state automaton depicted in the introduction is obtained, with  $s_1 = E^{-1}$  and  $s_2 = XE^{-1}$  (in the picture, we have simplified labels  $X|x$  to  $x$ , for  $x \in \mathbb{R}$ ). This automaton is a finite representation of  $E^{-1}$  (strictly speaking of  $l(E^{-1})$ ), and can be used to generate its coefficients as follows. Since  $l(E^{-1}) = \lambda(\{E^{-1}\})$ , we can use the formula at the end of Section 8, to compute, for instance, the coefficient in  $E^{-1}$  of  $X^2$ , by considering all transition sequences of length 2 starting and ending in  $s_1 = E^{-1}$  (note that paths ending in  $s_2 = XE^{-1}$  do not contribute since  $o_{\mathcal{R}}(XE^{-1}) = 0$ ). There are two such sequences:

$$E^{-1} \xrightarrow{-3/2} E^{-1} \xrightarrow{-3/2} E^{-1} \xrightarrow{1/2} \quad \text{and} \quad E^{-1} \xrightarrow{-7/2} XE^{-1} \xrightarrow{1} E^{-1} \xrightarrow{1/2}$$

yielding  $(-3/2)(-3/2)1/2 + (-7/2)1(1/2) = -5/8$  for the coefficient of  $X^2$  in  $l(E^{-1})$ .  $\square$

## 10 An example in the max-plus semiring: task resource systems

It is shown how the timed behaviour of task-resource systems can be *specified* coinductively, that is, by means of a behavioural differential equation, and *implemented* by a finite nd-automaton, derived from this differential equation. The relevance of this example is not so much the obtained automaton (which is not new—see, e.g., [GM98]), but rather the way in which it illustrates our methodology of *deriving* finite representations from behavioural specifications.

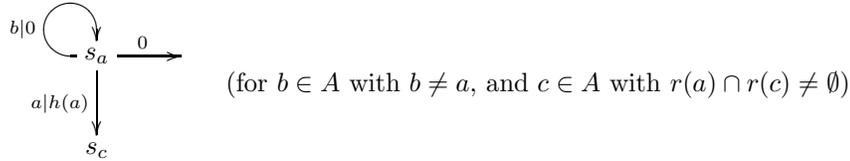
A *(timed) task-resource system* is a four tuple  $(A, R, r, h)$  consisting of a finite set  $A$  of *tasks*, a finite set  $R$  of *resources*, a function  $r : A \rightarrow \mathcal{P}(R)$  assigning to each task the finite non-empty set of resources it uses, and a function  $h : A \rightarrow \mathbb{R}^+$  specifying for each task its execution time. A word  $w = a_1 \cdots a_n \in A^*$  is interpreted as a *schedule*, that is, a sequence of tasks. The question to be addressed is how long it takes to perform all the tasks in a schedule, while adhering to the following rules: all resources are available at time 0; the execution of a task  $a_i$  in  $w$  begins as soon as all resources  $r(a_i)$  it requires, possibly used by earlier tasks, are available; and a task  $a_i$  uses the resources in  $r(a_i)$  for  $h(a_i)$  time units, during which these resources are *not* available for other tasks. To mention two extreme cases, the execution time of a word  $a_1 \cdots a_n$  will be equal to the *maximum* of the durations  $h(a_i)$  in case no tasks have any resources in common. If, in contrast, any two subsequent tasks do have a common resource, the execution time will be the *sum* of the durations  $h(a_i)$ .

The latter examples explain why we shall be working in the so-called max-plus semiring  $\mathbb{R}_{\max} = ([-\infty, \infty), \max, +, -\infty, 0)$ . Taking the finite set  $A$  of tasks as our input alphabet, we are thus looking for a formal definition of a power series  $y \in \mathbb{R}_{\max}\langle\langle A \rangle\rangle$ , assigning to each schedule  $w$  its execution time  $y(w)$  according to the above informal specification. In order to define  $y$ , first a power series  $\tilde{a}$  is associated with every task  $a \in A$ , which for a schedule  $w \in A^*$  will give the execution time  $\tilde{a}(w)$  of  $w$  counted *from the first time that the task  $a$  in  $w$  is executed*. Then  $y$  can be expressed as the maximum of all  $\tilde{a}$ . The following system of behavioural differential equations defines all these series formally:

differential equation	initial value
$(\tilde{a})_b = \tilde{a}$ (for $b \neq a$ )	$(\tilde{a})(\varepsilon) = 0$
$(\tilde{a})_a = \max \{h(a) + \tilde{c} \mid r(a) \cap r(c) \neq \emptyset\}$	
$y = \max \{\tilde{a} \mid a \in A\}$	

Here we silently adopt the convention that the maximum of an empty set is (the constant series)  $-\infty$ , which is the 0 of the semiring  $\mathbb{R}_{\max}$ . The existence of unique solutions for these equations is proved by a straightforward variation on the proof of Theorem 4.1. Following the procedure of Section 9, a finite nd-automaton for the  $\tilde{a}$  can be derived from the defining differential equations.

It consists of a set of states  $\{s_a | a \in A\}$ , all with output 0, and with transitions



This automaton represents  $y$ , that is,  $y = \max\{\lambda(\{s_a\}) \mid a \in A\}$ , which can be easily proved by—you guessed right—coinduction!!

## 11 Discussion

We briefly mention some of the work that remains to be done: (1) General formats for behavioural differential equations, ensuring the existence of a unique solution, should be determined. (Ideally, one should also be able to tell from the format of the equation whether its solution is rational and, hence, *finitely* representable by a nd-automaton.) The definition of such formats will be related to that of transition system specifications, for instance as described in [GV92] and used in [RT94]. See also [PE98]. (2) The effectiveness of the coinduction proof principle is to be further investigated, as well as the minimization of finite representations, to which it is closely related. In [Rut98], we have shown, for the case of  $k = \mathbb{B}$ , that coinduction is an effective proof method for equality of regular expressions. The proof is based on the observation that a language  $\sigma$  is regular iff the subautomaton  $\langle \sigma \rangle \subseteq \mathbb{B}\langle\langle A \rangle\rangle$  is finite (which amounts to a coalgebraic reformulation of Kleene’s theorem). In the present setting, this easily generalizes to finite semirings, but more should be said about infinite semirings as well. (3) The example of the shuffle inverse should be further investigated. For instance, in control theory, one looks for a controller  $C$  such that its parallel composition with a plant  $P$  satisfies a given specification  $S$ , that is,  $P \parallel C \subseteq S$ . It might be helpful to rewrite this into  $C \subseteq P^{-1} \parallel S$ . (4) Many more examples of specifications by behavioural differential equations and the corresponding implementations are needed, in order to get a better understanding of their usefulness. Notably, other tropical and idempotent semirings, as described in [Gun98], should be investigated from the present perspective. Another class of examples is to do with probabilistic nd-transition systems and Markov chains. (5) Finally, applying the universal coalgebraic definition of bisimulation directly to nd-automata (and not only to deterministic automata as we have done here) will yield notions of equivalence that have an interest in their own right. For instance, taking  $k = \mathbb{R}$  gives a notion of bisimulation that is (under conditions) probabilistic bisimulation [LS91], treated coalgebraically in [dVR97]; see [Fuj98] for another example.

## References

- [BR88] J. Berstel and C. Reutenauer. *Rational series and their languages*, volume 12 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1988.
- [Brz64] J.A. Brzozowski. Derivatives of regular expressions. *Journal of the ACM*, 11(4):481–494, 1964.
- [Con71] J.H. Conway. *Regular algebra and finite machines*. Chapman and Hall, 1971.
- [dVR97] E.P. de Vink and J.J.M.M. Rutten. Bisimulation for probabilistic transition systems: a coalgebraic approach. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela, editors, *Proceedings of ICALP’97*, volume 1256 of *Lecture Notes in Computer Science*, pages 460–470, 1997. To appear in *Theoretical Computer Science*.
- [Fuj98] Y. Fujiwara. Bisimulation as performance equivalence criterion:  $(\max, +)$  case. Technical report, 1998. Preprint.

- [GM98] S. Gaubert and J. Mairesse. Task resource models and  $(\max,+)$  automata. In *[Gun98]*, pages 133–144, 1998.
- [Gun98] J. Gunawardena. *Idempotency*. Publications of the Newton Institute. Cambridge University Press, 1998.
- [GV92] J.F. Groote and F. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, October 1992.
- [LS91] K.G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94:1–28, 1991.
- [PE98] Dusko Pavlović and Martín Escardó. Calculus in coinductive form. In *Proceedings of the 13th Annual IEEE Symposium on Logic in Computer Science*, pages 408–417. IEEE Computer Society Press, 1998.
- [RT94] J.J.M.M. Rutten and D. Turi. Initial algebra and final coalgebra semantics for concurrency. In J.W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Proceedings of the REX School/Symposium ‘A decade of concurrency’*, volume 803 of *Lecture Notes in Computer Science*, pages 530–582. Springer-Verlag, 1994. FTP-available at `ftp.cwi.nl` as `pub/CWIreports/AP/CS-R9409.ps.Z`.
- [Rut96] J.J.M.M. Rutten. Universal coalgebra: a theory of systems. Report CS-R9652, CWI, 1996. FTP-available at `ftp.cwi.nl` as `pub/CWIreports/AP/CS-R9652.ps.Z`. To appear in *Theoretical Computer Science*.
- [Rut98] J.J.M.M. Rutten. Automata and coinduction (an exercise in coalgebra). Report SEN-R9803, CWI, 1998. FTP-available at `ftp.cwi.nl` as `pub/CWIreports/SEN/SEN-R9803.ps.Z`. Also in the proceedings of CONCUR '98, LNCS 1466, 1998, pp. 194-218.