Stochastic Analysis of Periodic Timed Data Flow Diagrams with Markovian Transition Times

Jürgen Symanzik

TR #96–24

December 1996

Keywords: Statistical Software Engineering, Formalized Data Flow Diagrams, Embedded Markov Process, Embedded Markov Chain, Aggregation Principle, Periodicity.

Department of Computer Science

226 Atanasoff Hall

Iowa State University

Ames, Iowa 50011–1040, USA

1 STOCHASTIC ANALYSIS OF PERIODIC TIMED DATA FLOW DIAGRAMS WITH MARKOVIAN TRANSITION TIMES

Abstract

Timed (or Stochastic) Data Flow Diagrams (TDFD's or SDFD's) introduced in [SB96b] are an extension of the Formalized Data Flow Diagrams, defined in [LWBL96]. This extension allows us to assess the quantitative behavior (e. g., performance, throughput, average load of a bubble, etc.) as well as the qualitative behavior (e. g., deadlock, reachability, termination, finiteness, liveness, etc.), eventually depending on different types of transition times, for the system modeled through the TDFD. In this paper, we consider Markovian transition times for the consumption of in–flow items and for the production of items on the out–flow. Moreover, we require the TDFD to be periodic and irreducible and it must have a finite reachability set. For these models, we have been able to apply an aggregation principle of [Sch84], extended for periodic Markov chains by [Woo93], to efficiently determine stationary probabilities, expected waiting times, and limiting process probabilities.

1.1 Introduction

In [SB96b] we introduced Timed (or Stochastic) Data Flow Diagrams (TDFD's or SDFD's) as an extension of Formalized Data Flow Diagrams (FDFD's), defined in [LWBL96]. In SDFD's, time is modeled through the definition of a stochastic time behavior for the consumption of in–flow items as well as a stochastic time behavior for the production of items on the out–flow. We followed the general approach of Stochastic Petri Nets given in [MBB+85] when defining SDFD's.

In this paper we consider one particular subclass of TDFD's, i. e., those that are periodic, irreducible, and have Markovian transition times. We call these SDFD's periodic Markovian Timed Data Flow Diagrams (periodic M–TDFD's). We will demonstrate how stationary probabilities, expected waiting times, and limiting process probabilities can be derived for M–TDFD's. The periodicity of the Markov chain, embedded in the Markov process (which is embedded in the given periodic M–TDFD), plays an important role for a computationally efficient analysis of interesting questions. This analysis is based on the aggregation principle of [Sch84]. Similarly, [Woo93] (Chapter 2) used this aggregation principle and the periodicity of N–stage stochastic service systems (it appears, as pointed out in [Woo93], that [Pat64] first noted this periodicity) to efficiently derive stationary probabilities and limiting process probabilities for 3– and 4–stage Markovian production lines.

Some work has been done to exploit the periodic functioning of Timed Petri Nets. In [Hil90] for example, results have been obtained for the performance evaluation of multi–stage production systems where this periodic functioning often occurs. [Yua86] defines process periods for Petri Nets and uses those to describe the system behavior. However, to our best knowledge, there exists no prior approach to aggregate the state space of periodic Timed Petri Nets or similar computational models in a manner suggested by the aggregation principle of [Sch84] and the extension for periodic Markov chains by [Woo93].

The typical two–step firing behavior of FDFD's has been applied to Timed Petri Nets as well. [RP84] allocates both an enabling time and a firing time to each transition. After a transition is enabled, it has to wait for a time (called the "enabling time") before it absorbs all tokens from its input bag. The tokens remain absorbed for the "firing time" after which the transition places tokens in the appropriate output bag. The idea of alternating enabling and firing time points is also built into the work of [HT91].

The work presented in this paper can not only be applied to periodic M–TDFD's, but it can be directly used for the previously described Petri Nets with alternating enabling and firing time points. However, since our background is in Software Engineering, in particular in "Structured Analysis" (SA) (e. g., [DeM78], [WM85a]) where traditional Data Flow Diagrams (DFD's) are probably the most widely

used specification technique in industry today ([BB93]), we wanted to present our results in this context. Even though little work has been done on TDFD's to date, we want to encourage the reader to consider TDFD's as a helpful model for complex time dependend systems. It does not matter for the stochastic analysis whether the Markov process and Markov chain under consideration result from a TDFD or a Timed Petri Net. Thus, many known methods and results for the stochastic analysis of Timed Petri Nets can be directly applied to TDFD's. Moreover, several advantages of TDFD's over Timed Petri Nets make them the more natural selection for software engineers as it has been pointed out in [SB96b].

The work presented in this paper relates to the new interdisciplinary field "Statistical Software Engineering", introduced in [Nat96]. We use statistical techniques that allow us to reduce computations when we analyze in the Specifications phase of the spiral software development process model ([Nat96], p. 63) whether quantitative requirements of the software system are fulfilled.

In Section 1.2, we will summarize basic definitions required within this paper. Section 1.3 deals with the characterization of periodic FDFD's. In Section 1.4, we demonstrate how to apply the aggregation principle of [Sch84] to periodic and irreducible M–TDFD's with finite reachability set. We conclude this paper with an overview of future work in Section 1.5.

1.2 Definitions

1.2.1 Stochastic Data Flow Diagrams

Data Flow Diagrams have been formalized at multiple places within the technical literature, e. g., in [DeM78], [WM85a], [WM85b], [Har87], [TP89], [You89], [Har92], and [Har96]. Within this paper, we make use of the definitions of Formalized Data Flow Diagrams (FDFD's) developed by Coleman, Wahls, Baker, and Leavens in [Col91], [CB94], [WBL93], and [LWBL96]. In particular for our examples, we use the notation from [LWBL96]. This cited paper also contains a more detailed explanation of the underlying operational semantics of FDFD's and an extended example.

In addition, we need the following defintions from [SB96b]:

Definition (1.2.1.1): A *firing sequence* (*computation sequence*) of an FDFD is a possibly infinite sequence $(b_i, a_i, j_i) \in B \times \{C, P\} \times I\!N, i \geq 0$, such that, if transition (b_i, a_i, j_i) is fired in state (bm, r, fs), then

$$(fs', r') = \begin{cases} (Consume(b_i))_{j_i}(fs, r), & \text{if } a_i = C \\ (Produce(b_i))_{j_i}(fs, r), & \text{if } a_i = P \end{cases}$$

$$bm'(b_i) = \begin{cases} working, & \text{if } a_i = C \\ idle, & \text{if } a_i = P \end{cases}$$

$$bm'(b) = bm(b) \quad \forall\, b \in B - \{b_i\}$$

and

$$(bm, r, fs) \rightarrow (bm', r', fs').$$

We introduce the notation $(bm, r, fs)[(b, a, j)]$ to indicate that transition (b, a, j) is fireable in state (bm, r, fs) and $(bm, r, fs)[(b, a, j)](bm', r', fs')$ to indicate that state (bm', r', fs') is reached upon the firing of transition (b, a, j) in state (bm, r, fs).

By induction, we extend this notation for firing sequences:

$$(bm_0, r_0, fs_0)[(b_1, a_1, j_1), \ldots, (b_{n-1}, a_{n-1}, j_{n-1}), (b_n, a_n, j_n)]$$

is used to indicate that transition (b_n, a_n, j_n) is fireable in state $(bm_{n-1}, r_{n-1}, fs_{n-1})$, given that

$$(bm_0, r_0, fs_0)[(b_1, a_1, j_1), \ldots, (b_{n-1}, a_{n-1}, j_{n-1})](bm_{n-1}, r_{n-1}, fs_{n-1})$$

holds. By analogy, we use

$$(bm_0, r_0, fs_0)[(b_1, a_1, j_1), \ldots, (b_n, a_n, j_n)](bm_n, r_n, fs_n)$$

to indicate that state (bm_n, r_n, fs_n) is reached upon the firing of the sequence $(b_1, a_1, j_1), \ldots, (b_n, a_n, j_n)$.

■

Definition (1.2.1.2): The *set of firing sequences* (*set of computation sequences, language*) of an FDFD, denoted by $FS(FDFD, \gamma_{initial})$, is the set containing all firing sequences that are possible for this FDFD, given $\gamma_{initial} = (bm_{initial}, r_{initial}, fs_{initial})$, i. e.,

$$FS(FDFD, \gamma_{initial}) = \{s \mid s \in (B \times \{C, P\} \times I\!N)^* \wedge \gamma_{initial}[s]\}.$$

By analogy, we define

$$FS_i(FDFD, \gamma_{initial}) = \{s \mid s \in (B \times \{C, P\} \times I\!N)^i \wedge \gamma_{initial}[s]\}, i \geq 0,$$

the set of firing sequences of length i when starting in $\gamma_{initial}$. ■

Definition (1.2.1.3): The *Reachability Set* of an FDFD, denoted by $RS(FDFD, \gamma_{initial})$, is the set of states $\gamma = (bm, r, fs)$ that are reachable from $\gamma_{initial} = (bm_{initial}, r_{initial}, fs_{initial})$, i. e.,

$$RS(FDFD, \gamma_{initial}) = \{\gamma \mid \gamma \in , \ \wedge\, \exists\, s \in FS(FDFD, \gamma_{initial}) \ : \ \gamma_{initial}[s]\gamma\}.$$

By analogy, we define

$$RS_i(FDFD, \gamma_{initial}) = \{\gamma \mid \gamma \in , \ \wedge \exists\, s \in FS_i(FDFD, \gamma_{initial}) \ : \ \gamma_{initial}[s]\gamma\}, i \geq 0,$$

the set of states that are reachable in i steps when starting in $\gamma_{initial}$. ■

Definition (1.2.1.4): Let $EN(\gamma) \subseteq B \times \{C, P\} \times I\!N$ be the set of transitions that are enabled in state $\gamma = (bm, r, fs)$, i. e.,

$$EN(\gamma) = \{s \mid s \in (B \times \{C, P\} \times I\!N) \wedge \gamma[s]\} = FS_1(FDFD, \gamma).$$ ■

Now, we specialize our definitions from [SB96b] with respect to Markovian transition times.

Definition (1.2.1.5): A *timed firing sequence* (TFS) of an FDFD with initial state $\gamma_{initial}$ is a pair $tfs = (s, \tau)$, where $s \in FS(FDFD, \gamma_{initial})$ and τ is a non–decreasing sequence (of the same length) of real non–negative values representing the instants of firing (called *epochs*) of each transition, such that consecutive transitions (b_i, a_i, j_i) and $(b_{i+1}, a_{i+1}, j_{i+1})$ correspond to ordered epochs $\tau_i \leq \tau_{i+1}$. The time intervals $[\tau_i, \tau_{i+1})$ between consecutive epochs represent the periods in which the FDFD remains in state γ_i (assuming $\tau_0 = 0$). A *history* of the FDFD up to the kth epoch τ_k is denoted by $Z(k)$. ■

Definition (1.2.1.6): A *Markovian Timed Data Flow Diagram* (M–TDFD) is a SDFD with associated FDFD and initial state $\gamma_{initial}$ where the selection of the transition that fires is based on the Race Policy with marginal distributions (that do not depend on state $\underline{\gamma}$ and past history $\underline{Z}$)

$$\phi_i(x) = \phi_i(x \mid \underline{\gamma}, \underline{Z}) = Exp(x; \lambda_i), i = 1, \ldots, \mid EN(\underline{\gamma}) \mid \ \forall \underline{\gamma} \ \forall \underline{Z}$$

and with an initial probability distribution on the Reachability Set $RS(FDFD, \gamma_{initial})$. $Exp(x; \lambda_i)$ represents the Exponential distribution with probability density function $f(x) = \lambda_i \exp(-\lambda_i x) I_{(0,\infty)}(x)$ and cumulative distribution function $F(x) = (1 - \exp(-\lambda_i x)) I_{(0,\infty)}(x)$ for $\lambda_i > 0$. ■

Because of the memoryless property of the Exponential distribution, we do not have to distinguish among the possible cases introduced in [SB96b] how to deal with the past history $\underline{Z}$. We have the same behavior for Resampling, Work Age Memory, and Enabling Age Memory.

This definition introduces the *embedded Markov process* of the M–TDFD with a one–on–one mapping between the discrete state space and the reachability set $RS(FDFD, \gamma_{initial})$. We refer to this state

space together with the rules for state changes (implied by the mappings *Enabled*, *Consume*, and *Produce* of the M–TDFD) as the *embedded Markov chain* of the M–TDFD. For the initial probability distribution on $RS(FDFD, \gamma_{initial})$, we assume that $Pr(\text{system is in state } \gamma_{initial} \text{ at time } \tau_0 = 0) = 1$.

1.2.2 Periodic Markov Chains

In this section we will summarize definitions and theorems on periodic Markov chains given in [IM76], Chapters 2 and 3. It is assumed that the reader is familiar with the basic notations for Markov chains.

Definition (1.2.2.1): A subset, C, of the state space, S is called *closed* if $p_{ik} = 0$ for all $i \in C$ and $k \notin C$. If a closed set consists of a single state, then that state is called an *absorbing state*. ■

Definition (1.2.2.2): A Markov chain is called *irreducible* if there exists no nonempty closed set other than S itself. If S has a proper closed subset, it is called *reducible*. ■

Definition (1.2.2.3): Two states, i and j, are said to *intercommunicate* if for some $n \geq 0$, $p_{ij}^{(n)} > 0$ and for some $m \geq 0$, $p_{ji}^{(m)} > 0$. ■

Theorem (1.2.2.4): A Markov chain is irreducible if and only if all pairs of states intercommunicate.

Definition (1.2.2.5): State j has *period* d if the following two conditions hold:

(i) $p_{jj}^{(n)} = 0$ unless $n = md$ for some positive integer m and

(ii) d is the largest integer with property (i).

State j is called *aperiodic* when $d = 1$. ■

Theorem (1.2.2.6): State j has period d if and only if d is the greatest common divisor of all those n's for which $p_{jj}^{(n)} > 0$ (that is, $d = G.C.D.\{n \mid p_{jj}^{(n)} > 0\}$).

Lemma (1.2.2.7): The state space of a periodic irreducible Markov chain of period d can be partitioned into d disjoint classes $D_0, D_1, \ldots, D_{d-1}$ such that from D_j the chain goes, in the next step,

to D_{j+1} for $j = 0, 1, \ldots, d-2$. From D_{d-1} the chain returns in the next step to D_0. ■

Finally, we indicate the following Theorem, introduced as Proposition 6–28 and proved in [KSK76].

Theorem (1.2.2.8): The period of a recurrent chain for the state i is a constant independent of the state i.

1.2.3 Periodic Formalized Data Flow Diagrams

Similiar to Subsection 1.2.2, we now define related terms for an FDFD with initial state $\gamma_{initial}$. It should be obvious that there exists a one–on–one mapping between the reachability set $RS(FDFD, \gamma_{initial})$ of an FDFD and the discrete state space of a Markov chain. We refer to this state space together with the rules for state changes (implied by the mappings *Enabled*, *Consume*, and *Produce* of the FDFD) as the *embedded Markov chain* of the FDFD.

Definition (1.2.3.1): A subset C, of the reachability set, $RS(FDFD, \gamma_{initial})$, is called *closed* if for all $\gamma_i \in C$ and $\gamma_k \notin C$ there exists no transition $s \in (B \times \{C, P\} \times I\!N)$ such that $\gamma_i[s]\gamma_k$. If a closed set consists of a single state, then that state is called a *deadlock* state. ■

Definition (1.2.3.2): An FDFD with initial state $\gamma_{initial}$ is called *irreducible* if there exists no nonempty closed set other than $RS(FDFD, \gamma_{initial})$ itself. If $RS(FDFD, \gamma_{initial})$ has a proper closed subset, it is called *reducible*. ■

Definition (1.2.3.3): Two states, γ_i and $\gamma_j \in RS(FDFD, \gamma_{initial})$, are said to *intercommunicate* if for some firing sequences s and t, $\gamma_i[s]\gamma_j$ and $\gamma_j[t]\gamma_i$. ■

Theorem (1.2.3.4): An FDFD with initial state $\gamma_{initial}$ is irreducible if and only if all pairs of states intercommunicate.

Proof: Follows directly from the embedded Markov chain. ■

Definition (1.2.3.5): A state $\gamma_i \in RS(FDFD, \gamma_{initial})$ has *period d* if the following two conditions hold:

(i) $\gamma_i[s]\gamma_i$ does not hold unless $s \in FS_n(FDFD, \gamma_i)$ where $n = md$ for some positive integer m and

(ii) d is the largest integer with property (i). ■

Note that the period d of a state γ_i is related to the times at which the FDFD might return to state γ_i. It does not mean that the FDFD with current state γ_i can return to this state upon the firing of exactly d transitions nor does it mean that the $FDFD$ will ever return to this state. Also, we do not have to define an aperiodic state γ_i, where $d = 1$. This case can never happen as we show in the next theorem.

Definition (1.2.3.6): An FDFD with initial state $\gamma_{initial}$ is *periodic with period d* if all of its states $\gamma \in RS(FDFD, \gamma_{initial})$ have period d. ■

Theorem (1.2.3.7): An FDFD with initial state $\gamma_{initial}$ is either aperiodic or it is periodic with period $d \geq 2$, where d is even.

Proof: It is impossible that the state does not change upon the firing of a transition. Even if nothing is consumed and nothing is produced upon the firing of a transition, at least one bubble changes its *BubbleMode*. Every bubble has to move from *idle* to *working* (*working* to *idle*) before it can return to *idle* (*working*). Therefore, $d \geq 2$. Through the execution of every transition, the *BubbleMode* of exactly one bubble is altered. Thus, after an odd number of transitions, at least one bubble has a *BubbleMode* different from its *BubbleMode* in the starting state. Thus, d is even. However, if the FDFD contains a deadlock state, if it can never return to some previously reached state, or if some states have different periods, then it is aperiodic. ■

Corollary (1.2.3.8): An irreducible FDFD with initial state $\gamma_{initial}$ and finite reachability set is periodic with period $d \geq 2$, where d is even.

Proof: Since the FDFD is irreducible, it has no deadlock state and all pairs of states intercommunicate. It is possible to return to any state in $RS(FDFD, \gamma_{initial})$. This means, the embedded Markov chain is recurrent since the state space (the reachability set of the FDFD) also is finite. According to Theorem (1.2.2.8), all states have the same period d. But then, as we have seen in the previous Theorem, $d \geq 2$ and d is even. ■

1.3 Characterization of Periodic FDFD's

For the results in Section 1.4 we will assume that a given M–TDFD with initial state $\gamma_{initial}$ is periodic, irreducible, and has a finite reachability set. However, what does a M–TDFD with these

features look like, what are the necessary criteria it has to fulfill? To answer these questions, we will first demonstrate the unpredictable behavior of FDFD's in Subsection 1.3.1 and indicate how the period d can be determined for FDFD's with finite reachability set in Subsection 1.3.2.

1.3.1 Unpredictable Behavior of FDFD's

As we have seen in [SB96a], FDFD's are computationally equivalent to Turing Machines. This implies that all interesting decidability problems such as reachability, termination, deadlock and liveness properties, and finiteness, that are undecidable for Turing Machines are also undecidable for FDFD's. Unfortunately, the questions whether an FDFD is periodic, irreducible, and has a finite reachability set are directly related to these decidability problems.

In the next example, we will see that the question whether an FDFD is periodic, irreducible, and has a finite reachability set does not only depend on the structure and the mappings *Enabled*, *Consume*, and *Produce*, but it depends on the initial state $\gamma_{initial}$ as well. Moreover, an FDFD with initial state $\gamma_{initial_1}$ might be periodic, irreducible, and might have a finite reachability set, but the same FDFD with initial state $\gamma_{initial_2}$ may not have any of these features.

Example (1.3.1.1): This example shows an FDFD whose features highly depend on its initial state $\gamma_{initial}$.

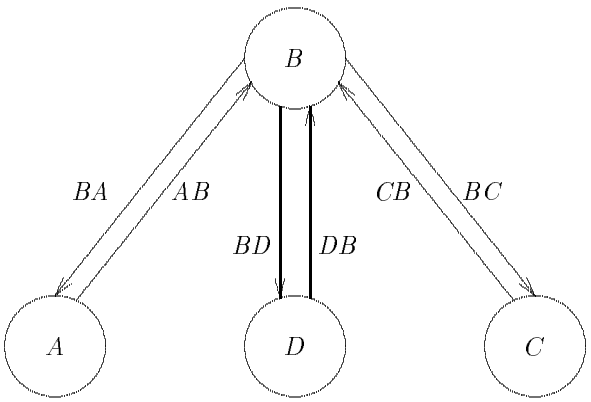

Figure 1.1: Example of an FDFD Highly Depending on $\gamma_{initial}$.

The mappings *Enabled*, *Consume*, and *Produce* for the FDFD shown in Figure 1.1 are defined as:

$$Enabled(A) = \lambda fs\ .\ (\neg IsEmpty(BA) \wedge Head(fs(BA)) = d)$$

$$Enabled(B) = \lambda fs\ .\ (\neg IsEmpty(AB) \wedge Head(fs(AB)) = a)$$
$$\vee\ (\neg IsEmpty(CB) \wedge Head(fs(CB)) = c)$$

$$\vee\ (\neg IsEmpty(AB) \wedge Head(fs(AB)) = a$$
$$\wedge \neg IsEmpty(CB) \wedge Head(fs(CB)) = c)$$
$$\vee\ (\neg IsEmpty(AB) \wedge Head(fs(AB)) = a$$
$$\wedge \neg IsEmpty(DB) \wedge Head(fs(DB)) = f)$$

$Enabled(C) = \lambda fs\ .\ (\neg IsEmpty(BC) \wedge Head(fs(BC)) = b)$

$Enabled(D) = \lambda fs\ .\ (\neg IsEmpty(BD) \wedge Head(fs(BD)) = e)$

Transition

$Consume(A) = \lambda(fs, r)\ .$

{**if** $(\neg IsEmpty(BA) \wedge Head(fs(BA)) = d)$
then $In(BA, A)(fs, r)$ (A, C, 1)
fi
}

$Consume(B) = \lambda(fs, r)\ .$

{**if** $(\neg IsEmpty(AB) \wedge Head(fs(AB)) = a)$
then $In(AB, B)(fs, r)$ (B, C, 1)
fi,
if $(\neg IsEmpty(CB) \wedge Head(fs(CB)) = c)$
then $In(CB, B)(fs, r)$ (B, C, 2)
fi,
if $(\neg IsEmpty(AB) \wedge Head(fs(AB)) = a$
$\wedge \neg IsEmpty(CB) \wedge Head(fs(CB)) = c)$
then $In(AB, B)(In(CB, B)(fs, r))$ (B, C, 3)
fi,
if $(\neg IsEmpty(AB) \wedge Head(fs(AB)) = a$
$\wedge \neg IsEmpty(DB) \wedge Head(fs(DB)) = f)$
then $In(AB, B)(In(DB, B)(fs, r))$ (B, C, 4)
fi
}

$Consume(C) = \lambda(fs, r)\ .$

{**if** $(\neg IsEmpty(BC) \wedge Head(fs(BC)) = b)$
then $In(BC, C)(fs, r)$ (C, C, 1)
fi

$\}$

$Consume(D) = \lambda(fs, r)$.

$\{$**if** $(\neg IsEmpty(BD) \wedge Head(fs(BD)) = e)$

then $In(BD, D)(fs, r)$ (D, C, 1)

fi

$\}$

$Produce(A) = \lambda(fs, r)$.

$\{$**if** $r(A)(BA) = d$

then $Out(a, AB, A)(fs, r)$ (A, P, 1)

fi

$\}$

$Produce(B) = \lambda(fs, r)$.

$\{$**if** $r(B)(AB) = a$

then $Out(b, BC, B)(fs, r)$ (B, P, 1)

fi,

if $r(B)(CB) = c$

then $Out(d, BA, B)(fs, r)$ (B, P, 2)

fi,

if $r(B)(AB) = a \wedge r(B)(CB) = c$

then $Out(e, BD, B)(fs, r)$ (B, P, 3)

fi,

if $r(B)(AB) = a \wedge r(B)(DB) = f$

then $Out(b, BC, B)(fs, r)$ (B, P, 4)

fi

$\}$

$Produce(C) = \lambda(fs, r)$.

$\{$**if** $r(C)(BC) = b$

then $Out(c, CB, C)(fs, r)$ (C, P, 1)

fi

$\}$

$Produce(D) = \lambda(fs, r)$.

$\{$**if** $r(D)(BD) = e$

then $Out(f, DB, D)(fs, r)$ (D, P, 1)

fi

$\}$

We consider three initial states

$$\gamma_{initial_1} = ((idle, idle, idle, idle), (\perp, \perp, \perp, \perp, \perp, \perp), ((a), (), (), (), (), ())),$$

$$\gamma_{initial_2} = ((idle, idle, idle, idle), (\perp, \perp, \perp, \perp, \perp, \perp), ((aa), (), (), (), (), ())),$$

and

$$\gamma_{initial_3} = ((idle, idle, idle, idle), (\perp, \perp, \perp, \perp, \perp, \perp), ((aaa), (), (), (), (), ())).$$

Note that the only difference between these initial states is the number of times the $OBJECT$ "a" initially appears on flow AB. This notation indicates

$$(bm_{initial}, r_{inital}, fs_{inital}) =$$
$$((bm(A), bm(B), bm(C), bm(D)),$$
$$(r(B)(AB), r(C)(BC), r(B)(CB), r(A)(BA), r(D)(BD), r(B)(DB)),$$
$$(fs(AB), fs(BC), fs(CB), fs(BA), fs(BD), fs(DB))).$$

For $\gamma_{initial_1}$, the reachability set $RS(FDFD, \gamma_{initial_1})$ consists only of the following eight states $\gamma_1, \ldots, \gamma_8$:

- $\gamma_1 = ((idle, idle, idle, idle), (\perp, \perp, \perp, \perp, \perp, \perp), ((a), (), (), (), (), ())) = \gamma_{initial_1}$
- $\gamma_2 = ((idle, working, idle, idle), (a, \perp, \perp, \perp, \perp, \perp), ((), (), (), (), (), ()))$
- $\gamma_3 = ((idle, idle, idle, idle), (\perp, \perp, \perp, \perp, \perp, \perp), ((), (b), (), (), (), ()))$
- $\gamma_4 = ((idle, idle, working, idle), (\perp, b, \perp, \perp, \perp, \perp), ((), (), (), (), (), ()))$
- $\gamma_5 = ((idle, idle, idle, idle), (\perp, \perp, \perp, \perp, \perp, \perp), ((), (), (c), (), (), ()))$
- $\gamma_6 = ((idle, working, idle, idle), (\perp, \perp, c, \perp, \perp, \perp), ((), (), (), (), (), ()))$
- $\gamma_7 = ((idle, idle, idle, idle), (\perp, \perp, \perp, \perp, \perp, \perp), ((), (), (), (d), (), ()))$
- $\gamma_8 = ((idle, idle, working, idle), (\perp, \perp, \perp, d, \perp, \perp), ((), (), (), (), (), ()))$

At any time during the execution of the FDFD, we have $\mid EN(\gamma_i) \mid = 1, i = 1, \ldots, 8$, i. e., the FDFD behaves deterministically. Therefore, upon the execution of the firing sequence

$$s_1 = ((B,C,1),(B,P,1),(C,C,1),(C,P,1),(B,C,2),(B,P,2),(A,C,1),(A,P,1))$$

the FDFD returns to $\gamma_{initial_1}$ when started in $\gamma_{initial_1}$, i. e., $\gamma_{initial_1}[s_1]\gamma_{initial_1}$. Trivially, the FDFD with $\gamma_{initial_1}$ is irreducible, periodic with period 8, and has a finite reachability set of size 8.

Now we consider the FDFD with $\gamma_{initial_2}$. Since only an *OBJECT* on a flow has been replicated but no *OBJECT* has been removed in comparision with $\gamma_{initial_1}$, it should be obvious that $\gamma_{initial_2}[s_1]\gamma_{initial_2}$ holds. But, is the FDFD with $\gamma_{initial_2}$ still irreducible and periodic? Consider

$$s_2 = ((B,C,1),(B,P,1),(C,C,1),(C,P,1),(B,C,3),(B,P,3),(D,C,1),(D,P,1)).$$

We have $\gamma_{initial_2}[s_2]\gamma_{dead}$, where $\gamma_{dead} = ((idle, idle, idle, idle),(\bot,\bot,\bot,\bot,\bot,\bot),((),(),(),(),(),(f)))$ denotes a deadlock state. Hence, the FDFD with $\gamma_{initial_2}$ is not irreducible. γ_{dead} has no period d and thus the FDFD with $\gamma_{initial_2}$ is not periodic.

Finally, we consider the FDFD with $\gamma_{initial_3}$. Again, one *OBJECT* on a flow has been replicated twice and no *OBJECT* has been removed in comparision with $\gamma_{initial_1}$. We consider s_2 first. We have $\gamma_{initial_3}[s_2]\gamma_{not\_dead}$, where $\gamma_{not\_dead} = ((idle, idle, idle, idle),(\bot,\bot,\bot,\bot,\bot,\bot),((a),(),(),(),(),(f)))$ represents a state that is not a deadlock state. Instead, we have $\gamma_{not\_dead}[(B,C,4),(B,P,4)]\gamma_3$, where γ_3 is identical to the state of the same name in $RS(FDFD, \gamma_{initial_1})$. Once γ_3 has been reached, only those states can be reached that are also reachable for the FDFD with $\gamma_{initial_1}$. But none of these states intercommunicates with $\gamma_{initial_3}$ for example, thus the FDFD with $\gamma_{initial_3}$ can not be irreducible. We leave it to the reader to determine whether the FDFD with $\gamma_{initial_3}$ is periodic and which deadlock states can possibly be reached. This exmple illustrates how such small changes in the initial state result in quite different behavior. We leave it to the reader to imagine how inpredictably a more complex FDFD might behave. ■

Based on the previous example, it becomes obvious that our initial assumption that the period d (if the FDFD is periodic at all) might depend on the number of bubbles and flows does not hold at all. We found several examples where the period d is proportional to $(2 \cdot \#bubbles)$, allowing each of the bubbles to alter between *idle* and *working* for the same number of times. However, we can not generalize from these examples to the general behavior of FDFD's. Especially since the period does not only depend on the structure but on the initial state as well. This is reasonable when recalling the fact that FDFD's have the same computational power as Turing Machines. Anything can happen.

1.3.2 Determination of d

As we already pointed out earlier, FDFD's are computationally equivalent to Turing Machines ([SB96a]). Therefore, we can not generally answer the questions whether an FDFD is periodic, irreducible, and has a finite reachability set. However, we can indicate an algorithm that determines whether the reachability set is finite or terminates after a fixed number of steps if the reachability set has not been fully exploited by then. If the reachability set is finite, we can determine whether the FDFD is irreducible, and if so, what period d it has.

First we want to introduce a definition from graph theory (e. g., [AHU74], p. 189):

Definition (1.3.2.1): Let $G = (V, E)$ be a directed graph. We can partition V into equivalence classes, $V_i, 1 \leq i \leq r$, such that vertices v and w are equivalent if and only if there is a path from v to w and a path from w to v. Let $E_i, 1 \leq i \leq r$, be the set of edges connecting the pairs of vertices in V_i. The graphs $G_i = (V_i, E_i)$ are called the *strongly connected components* of G. Even though every vertex of G is in some V_i, G may have edges not in any E_i. A graph is said to be *strongly connected* if it has only one strongly connected component. ■

Obviously, the question whether an FDFD with initial state $\gamma_{initial}$ is irreducible is equivalent to the question whether its reachability graph is strongly connected.

We can summarize the procedure that eventually returns the period d in four steps:

Step 1 : Determine the reachability set $RS(FDFD, \gamma_{initial})$

This can be done using the following breadth–first algorithm:

$i = 0$; $New_i = \{\gamma_{initial}\}$; $Reached_i = New_i$; $Examine_i = New_i$
while ($i \leq$ MAXSTEPS and $Examine_i \neq \{\}$)
{
$New_{i+1} = \bigcup_{\gamma \in Examine_i} RS_1(FDFD, \gamma)$ % all states reachable in one more step
$Reached_{i+1} = Reached_i \cup New_{i+1}$ % all states reached so far
$Examine_{i+1} = New_{i+1} - Reached_i$ % all states not yet examined
$i = i + 1$
}
if $Examine_i = \{\}$ exit "Reachability set is finite."
else exit "No solution found."

We stop if we find no solution.

Step 2 : Determine whether the reachability graph is strongly connected

Let $G = (V, E)$ with $V = RS(FDFD, \gamma_{initial})$ and E the set of edges implied by the mappings *Consume* and *Produce* be the (directed) reachability graph of the related FDFD with initial state $\gamma_{initial}$. Let n = number of vertices = $\mid RS(FDFD, \gamma_{initial}) \mid$ and e = number of edges. We can apply an algorithm that finds the strongly connected components of G. For example, Algorithm 5.4 in [AHU74], p. 193, performs this task in $O(MAX(n, e))$ time. We stop if G is not strongly connected, i. e., if the FDFD with $\gamma_{initial}$ is not irreducible.

Step 3 : Determine the shortest return path

For every $\gamma_i \in RS(FDFD, \gamma_{initial}), i = 1, \ldots, n$, we determine the shortest path from γ_i to γ_i with length $d_i > 0$. This can be done applying Dijkstra's Algorithm (e. g., Algorithm 5.6 in [AHU74], p. 207, or Section 6.4 in [PS82]) to every γ_i in time $O(n^2)$. We could also use the Floyd–Warshall Algorithm (e. g., Section 6.5 in [PS82]) that finds the shortest paths between all pairs of nodes in $O(n^3)$ time.

Step 4 : Determine the period d

According to Corollary (1.2.3.8) the FDFD with $\gamma_{initial}$ is periodic with period $d \geq 2$, where d is even. We can determine d as $G.C.D.\{d_i \mid i = 1, \ldots, n\}$.

Obviously, this 4–step approach is not the most efficient one. For example, we do not really need Dijkstra's Algorithm or the Floyd–Warshall Algorithm to determine the shortest path since the cost for each step is the same, no matter which edge is selected. Moreover, it should be possible to combine Steps 1 to 3 into a more efficient algorithm. However, this goes beyond the scope of this paper.

1.4 Analysis of Periodic M–TDFD's

1.4.1 The Aggregation Principle

The idea presented in the following extract from [Sch84] is commonly referred to as the *aggregation principle*:

> "Let S be a finite or countably infinite set and $X = \{X_n; n \geq 0\}$ a homogeneous irreducible recurrent Markov chain on S with transition matrix $P = (p_{ij})$. Let S' be a nonempty subset of S, and denote by $n_1, n_2, \ldots$ the successive random times at which X is visiting S'. Then

$X' = \{X_{n_1}, X_{n_2}, \ldots\}$ is a homogeneous irreducible recurrent Markov chain on S' ([KSK76], p. 164[1]). Its transition matrix, P', is given by

$$(1.1) \quad p'_{ij} = p_{ij} + \sum_{k \in S - S'} p_{ik} r_{kj}, \quad i, j \in S',$$

where $r_{kj}, k \in S - S', j \in S'$, is the probability that X will first hit S' at state j given start in k. Likewise, p'_{ij} is the probability for X to first reenter S' at j given start in i. If X is ergodic, so is X' (but not vice versa). X' will be said to arise from X by watching X on S', only. The equations (1.2) $x = xP$ and (1.3) $x' = x'P'$, x and x' denoting row vectors, possess strictly positive solutions p, p' which are unique up to multiplicative constants, and for which (1.4) $p'_i = cp_i, i \in S'$, c a constant ([KSK76], p. 164[1]). The p, p' shall be assumed to denote probability vectors in the ergodic case."

1.4.2 Application to Periodic M–TDFD's

As pointed out in [Woo93], in the case of a periodic Markov chain, if S' is taken as a periodic subset of S, then the constant c in [Sch84] (1.4) is simply the period d of the Markov chain.

We can summarize the process to analyze a periodic and irreducible M–TDFD of period d with initial state $\gamma_{initial}$ and finite reachability set $RS(FDFD, \gamma_{initial})$ in the following algorithm:

(i) Determine the d equivalence classes of states of the FDFD associated with M–TDFD (the periodic subsets) $S_1, \ldots, S_d$. It is

$$S_j = \bigcup_{i=0}^{ceil(|RS(FDFD,\gamma_{initial})|/d)-1} RS_{id+j}(FDFD, \gamma_{initial}), j = 1, \ldots, d,$$

and

$$S = \bigcup_{j=1}^{d} S_j = RS(FDFD, \gamma_{initial}).$$

(ii) Determine P, with columns and rows representing states $\gamma_1, \ldots, \gamma_n$, $n = \mid RS(FDFD, \gamma_{initial}) \mid$, ordered according to the periodic subsets, starting with $S_d, S_1, \ldots, S_{d-1}$. It is

$$p_{ij} = \frac{\lambda_k}{\sum_{l=1}^{m} \lambda_l}, i, j = 1, \ldots, n,$$

such that $\gamma_i[(b_k, a_k, j_k)]\gamma_j$ and $\gamma_i[(b_l, a_l, j_l)], l = 1, \ldots, m$, where $m = \mid EN(\gamma_i) \mid$ and $\lambda_k, \lambda_1, \ldots, \lambda_m$ are the rates of the Exponential distributions related to the transitions $(b_k, a_k, j_k), (b_1, a_1, j_1), \ldots,$ (b_m, a_m, j_m), respectively.

[1] The reference [KSK76], p. 164, refers to Exercise 5 on page 164. In addition, the definition of P^E on page 133 and Lemma 6–6 on page 134 of the same reference are required to understand the reasoning in [Sch84].

(iii) Let S' be any of the periodic subsets $S_1, \ldots, S_d$ such that $\mid S' \mid \leq \mid S_j \mid \quad \forall j = 1, \ldots, d$. Let $\Delta S = S - S'$. Derive P' according to [Sch84] (1.1). r_{kj}, related to states $\gamma_k \in \Delta S, \gamma_j \in S'$, can be determined via the products of $p_{ij}s$ related to a firing sequence from state γ_k to state γ_j that does not reach any other state $\delta \in S'$, summed up over all possible firing sequences of this type. Note that all firing sequences to be considered are those with $d-1$ or less steps. Now, solve $x' = x'P'$ where solutions p' are strictly positive and can be normalized such that $p'\mathbf{1} = 1$, where $\mathbf{1}$ represents a vector of all 1's.

(iv) From [Sch84] (1.4) and by using P, we get the stationary probabilities p:

$$\begin{aligned} p_i &= \frac{1}{d} p'_i \quad \forall i : \gamma_i \in S' \\ p_i &= \frac{1}{d} \sum_{j\,:\gamma_j \in S'} p'_j r_{ji} \quad \forall i : \gamma_i \in \Delta S \end{aligned}$$

(v) The expected waiting times Ψ_i in the states $\gamma_i, i = 1, \ldots, n$, can be computed as

$$\Psi_i = \frac{1}{\sum_{l=1}^{m} \lambda_l}, i = 1, \ldots, n,$$

where $m = \mid EN(\gamma_i) \mid$ and $\lambda_1, \ldots, \lambda_m$ are the rates of the related Exponential distributions (as in (ii) above).

(vi) Based on the specialisation in [Woo93] for the general case of ergodic stationary semi Markov processes with countable state space ([AD88]), we can compute the limiting process probabilities $\Pi_i = \lim_{t \to \infty} P_i(t)$ of the states $\gamma_i, i = 1, \ldots, n$, i. e., the probability that the system is in the state γ_i for $t \to \infty$, as

$$\Pi_i = \frac{p_i \Psi_i}{\sum_{j=1}^{n} p_j \Psi_j}, i = 1, \ldots, n.$$

1.4.3 An Example

This example of an M–TDFD represents a Producer/Consumer problem with bounded buffer of size 2 (Figure 1.2). This means, the Producer can only produce two more items than the Consumer has consumed.

The mappings *Enabled*, *Consume*, and *Produce* are defined as:

$Enabled(P) = \lambda fs\ .\ (\neg IsEmpty(Done) \wedge Head(fs(Done)) = Yes)$

$Enabled(C) = \lambda fs\ .\ (\neg IsEmpty(f) \wedge Head(fs(f)) = 1)$

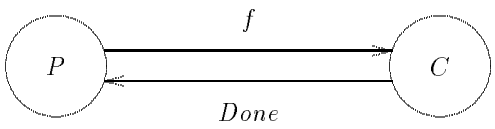

Figure 1.2: Example of a Periodic and Irreducible M–TDFD.

	Transition	Rate
$Consume(P) = \lambda(fs, r)$.		
{**if** $(\neg IsEmpty(Done) \wedge Head(fs(Done)) = Yes)$		
then $In(Done, P)(fs, r)$	(P, C, 1)	λ_1
fi		
}		
$Consume(C) = \lambda(fs, r)$.		
{**if** $(\neg IsEmpty(f) \wedge Head(fs(f)) = 1)$		
then $In(f, C)(fs, r)$	(C, C, 1)	λ_2
fi		
}		
$Produce(P) = \lambda(fs, r)$.		
{**if** $r(C)(Done) = Yes$		
then $Out(1, f, P)(fs, r)$	(P, P, 1)	λ_3
fi		
}		
$Produce(C) = \lambda(fs, r)$.		
{**if** $r(C)(f) = 1$		
then $Out(Yes, Done, C)(fs, r)$	(C, P, 1)	λ_4
fi		
}		

Initially, we have

$$\gamma_1 = \gamma_{initial} = (bm_{initial}, r_{inital}, fs_{inital}) = ((idle, idle), (\perp, \perp), ((), (Yes\, Yes))).$$

This notation indicates

$$((bm(P), bm(C)), (r(P)(Done), r(C)(f)), (fs(f), fs(Done))).$$

The reachability set $RS(FDFD, \gamma_{initial})$ consists of the following eight states:

- $\gamma_1 = ((idle,\ idle),\ (\perp,\ \perp),\ (()),\ (Yes\ \ Yes)))$
- $\gamma_4 = ((working,\ idle),\ (Yes,\ \perp),\ (()),\ (Yes)))$
- $\gamma_6 = ((idle,\ idle),\ (\perp,\ \perp),\ ((1)),\ (Yes)))$
- $\gamma_7 = ((working,\ idle),\ (\text{Yes},\ \perp),\ ((1)),\ ()))$
- $\gamma_8 = ((idle,\ working),\ (\perp,\ 1),\ (()),\ (Yes)))$
- $\gamma_2 = ((idle,\ idle),\ (\perp,\ \perp),\ ((1\quad 1)),\ ()))$
- $\gamma_3 = ((working,\ working),\ (Yes,\ 1),\ (()),\ ()))$
- $\gamma_5 = ((idle,\ working),\ (\perp,\ 1),\ ((1)),\ ()))$

According to Step 1 in Subsection 1.3.2 the reachability set has been gained in the following way:

i	New	Reached	Examine
0	$\{\gamma_1\}$	$\{\gamma_1\}$	$\{\gamma_1\}$
1	$\{\gamma_4\}$	$\{\gamma_1, \gamma_4\}$	$\{\gamma_4\}$
2	$\{\gamma_6\}$	$\{\gamma_1, \gamma_4, \gamma_6\}$	$\{\gamma_6\}$
3	$\{\gamma_7, \gamma_8\}$	$\{\gamma_1, \gamma_4, \gamma_6, \gamma_7, \gamma_8\}$	$\{\gamma_7, \gamma_8\}$
4	$\{\gamma_2, \gamma_3, \gamma_1\}$	$\{\gamma_1, \gamma_4, \gamma_6, \gamma_7, \gamma_8, \gamma_2, \gamma_3\}$	$\{\gamma_2, \gamma_3\}$
5	$\{\gamma_5, \gamma_4\}$	$\{\gamma_1, \gamma_4, \gamma_6, \gamma_7, \gamma_8, \gamma_2, \gamma_3, \gamma_5\}$	$\{\gamma_5\}$
6	$\{\gamma_6\}$	$\{\gamma_1, \gamma_4, \gamma_6, \gamma_7, \gamma_8, \gamma_2, \gamma_3, \gamma_5\}$	$\{\}$

Following Steps 2 to 4 in Subsection 1.3.2 reveals that the FDFD with $\gamma_{initial}$ is irreducible and periodic with period $d = 4$. The same can be seen in the reachability graph (Figure 1.3).

We follow the algorithm from Subsection 1.4.2 to further analyze this M–TDFD:

(i) The $d = 4$ equivalence classes of states are:

- $S_1 = \{\gamma_4, \gamma_5\}$
- $S_2 = \{\gamma_6\}$
- $S_3 = \{\gamma_7, \gamma_8\}$

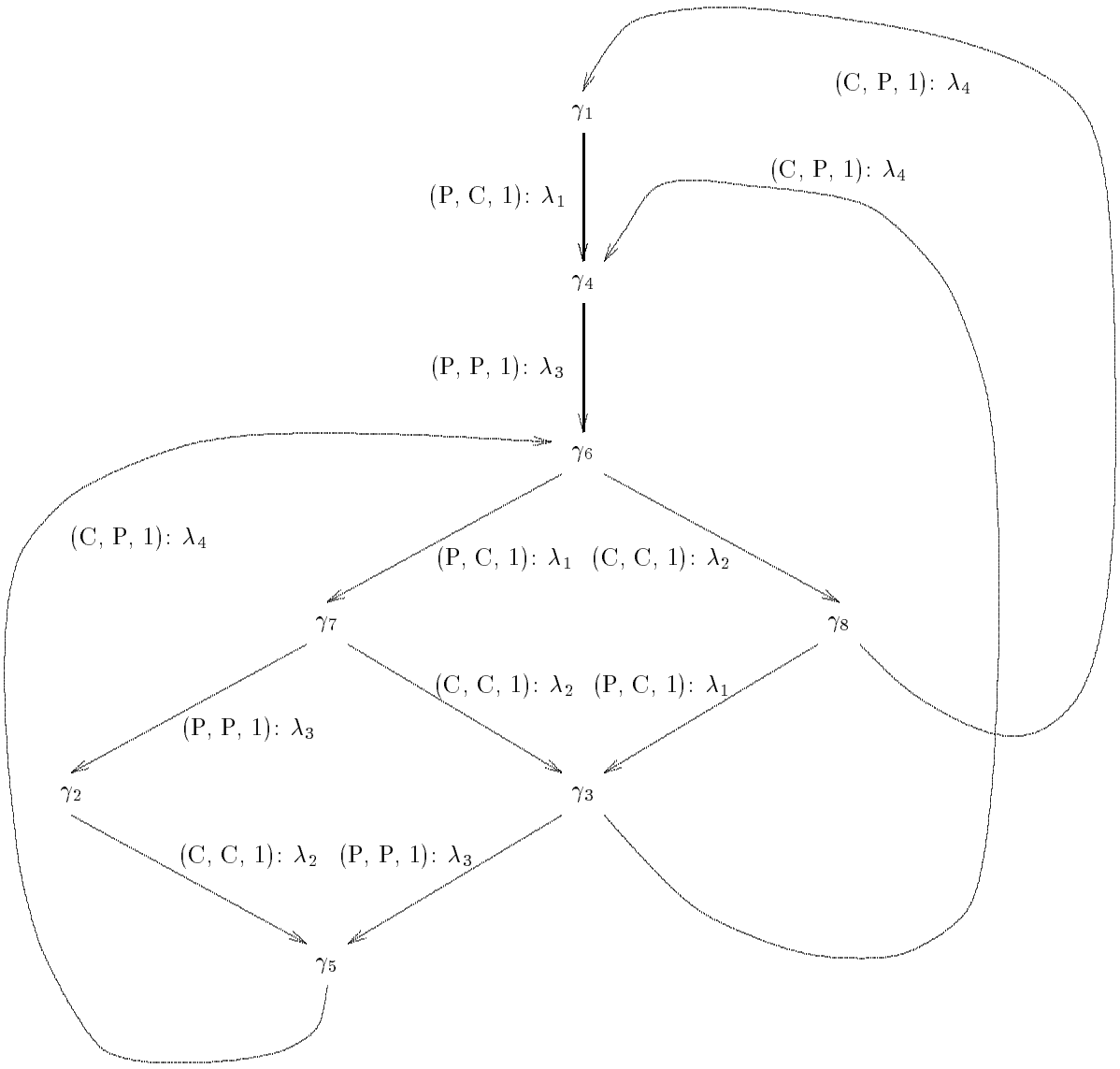

Figure 1.3: Reachability Graph of a Periodic and Irreducible M–TDFD.

- $S_4 = \{\gamma_1, \gamma_2, \gamma_3\}$
- $S = \{\gamma_1, \ldots, \gamma_8\}$

(ii) P can be determined using the reachability graph. It is arranged such that γ_1 appears in the first row/column and γ_8 appears in the last row/column:

$$P = \left(\begin{array}{ccc|cc|c|cc} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{\lambda_4}{\lambda_3+\lambda_4} & \frac{\lambda_3}{\lambda_3+\lambda_4} & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & \frac{\lambda_1}{\lambda_1+\lambda_2} & \frac{\lambda_2}{\lambda_1+\lambda_2} \\ \hline 0 & \frac{\lambda_3}{\lambda_2+\lambda_3} & \frac{\lambda_2}{\lambda_2+\lambda_3} & 0 & 0 & 0 & 0 & 0 \\ \frac{\lambda_4}{\lambda_1+\lambda_4} & 0 & \frac{\lambda_1}{\lambda_1+\lambda_4} & 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

(iii) We select $S' = S_2 = \{\gamma_6\}$. Then $\Delta S = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_7, \gamma_8\}$. We have

$$\begin{aligned} P' &= (p'_{66}) \\ &= \left(p_{66} + \sum_{k \in \Delta S} p_{6k} r_{k6}\right) \\ &= (0 + p_{67} r_{76} + p_{68} r_{86}) \\ &= \left(\frac{\lambda_1}{\lambda_1+\lambda_2}\left(\frac{\lambda_3}{\lambda_2+\lambda_3}\cdot 1 \cdot 1 + \frac{\lambda_2}{\lambda_2+\lambda_3}\cdot\frac{\lambda_4}{\lambda_3+\lambda_4}\cdot 1 + \frac{\lambda_2}{\lambda_2+\lambda_3}\cdot\frac{\lambda_3}{\lambda_3+\lambda_4}\cdot 1\right)\right. \\ &\quad \left. + \frac{\lambda_2}{\lambda_1+\lambda_2}\left(\frac{\lambda_4}{\lambda_1+\lambda_4}\cdot 1 \cdot 1 + \frac{\lambda_1}{\lambda_1+\lambda_4}\cdot\frac{\lambda_4}{\lambda_3+\lambda_4}\cdot 1 + \frac{\lambda_1}{\lambda_1+\lambda_4}\cdot\frac{\lambda_3}{\lambda_3+\lambda_4}\cdot 1\right)\right) \\ &= (1) \end{aligned}$$

and $p' = (1)$.

(iv) The stationary probabilities p are:

$$\begin{aligned} p_6 &= \frac{1}{4} \\ p_1 &= \frac{1}{4}\frac{\lambda_2}{\lambda_1+\lambda_2}\frac{\lambda_4}{\lambda_1+\lambda_4} \\ p_2 &= \frac{1}{4}\frac{\lambda_1}{\lambda_1+\lambda_2}\frac{\lambda_3}{\lambda_2+\lambda_3} \\ p_3 &= \frac{1}{4}\left(\frac{\lambda_1}{\lambda_1+\lambda_2}\frac{\lambda_2}{\lambda_2+\lambda_3} + \frac{\lambda_2}{\lambda_1+\lambda_2}\frac{\lambda_1}{\lambda_1+\lambda_4}\right) \\ p_4 &= \frac{1}{4}\left(\frac{\lambda_1}{\lambda_1+\lambda_2}\frac{\lambda_2}{\lambda_2+\lambda_3}\frac{\lambda_4}{\lambda_3+\lambda_4} + \frac{\lambda_2}{\lambda_1+\lambda_2}\frac{\lambda_1}{\lambda_1+\lambda_4}\frac{\lambda_4}{\lambda_3+\lambda_4} + \frac{\lambda_2}{\lambda_1+\lambda_2}\frac{\lambda_4}{\lambda_1+\lambda_4}\cdot 1\right) \\ p_5 &= \frac{1}{4}\left(\frac{\lambda_1}{\lambda_1+\lambda_2}\frac{\lambda_3}{\lambda_2+\lambda_3}\cdot 1 + \frac{\lambda_1}{\lambda_1+\lambda_2}\frac{\lambda_2}{\lambda_2+\lambda_3}\frac{\lambda_3}{\lambda_3+\lambda_4} + \frac{\lambda_2}{\lambda_1+\lambda_2}\frac{\lambda_1}{\lambda_1+\lambda_4}\frac{\lambda_3}{\lambda_3+\lambda_4}\right) \\ p_7 &= \frac{1}{4}\frac{\lambda_1}{\lambda_1+\lambda_2} \\ p_8 &= \frac{1}{4}\frac{\lambda_2}{\lambda_1+\lambda_2} \end{aligned}$$

(v) The vector of expected waiting times Ψ is:

$$\Psi = \left(\frac{1}{\lambda_1}, \frac{1}{\lambda_2}, \frac{1}{\lambda_3 + \lambda_4}, \frac{1}{\lambda_3}, \frac{1}{\lambda_4}, \frac{1}{\lambda_1 + \lambda_2}, \frac{1}{\lambda_2 + \lambda_3}, \frac{1}{\lambda_1 + \lambda_4} \right)$$

(vi) The limiting process probabilities are calculated as:

$$\Pi_i = \frac{p_i \Psi_i}{\sum_{j=1}^{8} p_j \Psi_j}, i = 1, \ldots, 8$$

(vii) If we assume that we have identical rates $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1$, we get

$$\begin{aligned} p &= \frac{1}{4} \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 1, \frac{1}{2}, \frac{1}{2} \right), \\ \Psi &= \frac{1}{2} (2, 2, 1, 2, 2, 1, 1, 1), \text{ and} \\ \Pi &= \frac{1}{11} (1, 1, 1, 2, 2, 2, 1, 1). \end{aligned}$$

1.5 Future Directions

In this paper we have demonstrated how the aggregation principle from [Sch84] can be used to analyze periodic and irreducible M–TDFD's with finite reachability sets. Especially for large models, this approach is very helpful to efficiently determine stationary probabilities, expected waiting times, and limiting process probabilities. So far, we have used this approach only to analyze periodic and irreducible M–TDFD's with finite reachability sets. Future work can be directed into two directions:

- Analysis of M–TDFD's with infinite reachability sets: Many real systems behave like queueing systems or queueing networks with finite or infinite queue lengths and tend to be periodic and irreducible. M–TDFD's representing such systems might be candidates to be analyzed in a manner similiar to the one described in this paper.

- Analysis of TDFD's with arbitrary transition times: The main idea in [Sch84] was the application of the aggregation principle to queueing systems and networks with arbitrary service and inter–arrival times, approximated through mixtures of Erlang distributions. We might be capable to do a computationally efficient analysis of TDFD's where transition times are modeled as mixtures of Erlang distributions, using the aggregation principle in its original context.

Finally, there exist several types of real systems that are good candidates to be correctly modeled and analyzed through (periodic and irreducible) M–TDFD's, while currently still being modeled and

analyzed through Timed Petri Nets. Examples for these systems are communication protocols (e. g., [MAT+77], [MB83], [Wal83]) and complex computer systems (e. g., [Zub80]). Another type of system that might work quite well are general Producer/Consumer systems or networks of these, e. g., multi–stage production systems (e. g., [Hil90]).

However, many systems modeled through Data Flow Diagrams, e. g., the case study of an elevator system in [You89], the cruise control system, the bottle–filling system, the pocket–sized logic analyser, and the defect inspection system, all in [WM85b], probably would have non–Markovian transition times. For models like these, an approximation of the real time behavior through mixtures of Erlang distributions might be possible and an analysis based on the aggregation principle should be preferable to results gained from simulation runs based on the TDFD.

Acknowledgements

Symanzik's research was partially supported by a German "DAAD–Doktorandenstipendium aus Mitteln des zweiten Hochschulsonderprogramms". The author wishes to thank Herbert T. David and Albert L. Baker for many valuable suggestions and references, including referral to the Ph.D. dissertation of Hoon–Shik Woo.

Bibliography

[AD88] E. Abdurachman and H.T. David. Cesaro Limits of Marked Point Processes on the Line. *Communication in Statistics — Stochastic Models*, 4(1):77–98, 1988.

[AHU74] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *The Design and Analysis of Computer Algorithms.* Addison–Wesley, Reading, Massachusetts, 1974.

[BB93] J.P. Bansler and K. Bødker. A Reappraisal of Structured Analysis: Design in an Organizational Context. *ACM Transactions on Information Systems*, 11(2):165–193, 1993.

[CB94] D.L. Coleman and A.L. Baker. Synthesizing Structured Analysis and Object–Oriented Specifications. Technical Report 94-04, Iowa State University, Department of Computer Science, 226 Atanasoff Hall, Ames, Iowa 50011, March 1994. Available by anonymous ftp from ftp.cs.iastate.edu or by e-mail from almanac@cs.iastate.edu.

[Col91] D.L. Coleman. *Formalized Structured Analysis Specifications.* PhD Thesis, Iowa State University, Ames, Iowa, 50011, 1991.

[DeM78] T. DeMarco. *Structured Analysis and System Specification.* Yourdon, Inc., New York, New York, 1978.

[Har87] D. Harel. Statecharts: A Visual Formalism for Complex Systems. *Science of Computer Programming*, 8(3):231–274, 1987.

[Har92] D. Harel. Biting the Silver Bullet. *Computer*, 21(1):8–20, January 1992.

[Har96] D. Harel. Executable Object Modeling with Statecharts. In *Proceedings of the 18th International Conference on Software Engineering*, pages 246–257. IEEE Computer Society Press, January 1996.

[Hil90] H.P. Hillion. Timed Petri Nets and Application to Multi–Stage Production Systems. In G. Rozenberg, editor, *Lecture Notes in Computer Science Vol. 424: Advances in Petri Nets 1989*, pages 281–305, Springer–Verlag, Berlin, Heidelberg, 1990.

[HT91] W. Henderson and P.G. Taylor. Embedded Processes in Stochastic Petri Nets. *IEEE Transactions on Software Engineering*, 17(2):108–116, 1991.

[IM76] D.L. Isaacson and R.W. Madsen. *Markov Chains, Theory and Applications.* Wiley, New York, London, Sydney, Toronto, 1976.

[KSK76] J.G. Kemeny, J.L. Snell, and A.W. Knapp. *Denumerable Markov Chains (Second Edition).* Springer–Verlag, New York, 1976.

[LWBL96] G.T. Leavens, T. Wahls, A.L. Baker, and K. Lyle. An Operational Semantics of Firing Rules for Structured Analysis Style Data Flow Diagrams. Technical Report 93–28d, Iowa State University, Department of Computer Science, 226 Atanasoff Hall, Ames, Iowa 50011, December 1993, revised, July 1996. Available by anonymous ftp from ftp.cs.iastate.edu or by e–mail from almanac@cs.iastate.edu.

[MAT+77] M. Mori, T. Araki, K. Taniguchi, N. Tokura, and T. Kasami. Some Decision Problems for Time Petri Nets and Applications to the Verification of Communication Protocols. *Transactions of the Institute of Electronics and Communication Engineers of Japan, Section E (English)*, 60(10):598–599, 1977.

[MB83] M. Menasche and B. Berthomieu. Time Petri Nets for Analyzing and Verifying Time Dependent Communication Protocols. In H. Rudin and C.H. West, editors, *Protocol Specification, Testing, and Verification, III*, pages 161–172. Elsevier (North–Holland), 1983.

[MBB+85] M.A. Marsan, G. Balbo, A. Bobbio, G. Chiola, G. Conte, and A. Cumani. On Petri Nets with Stochastic Timing. In *International Workshop on Timed Petri Nets, Torino, Italy, July 1985*, pages 80–87, 1985.

[Nat96] National Academy of Sciences. *Statistical Software Engineering.* National Academy Press, Washington, D.C., 1996.

[Pat64] R.L. Patterson. Markov Processes Occurring in the Theory of Traffic Flow through an N–Stage Stochastic Service System. *Journal of Industrial Engineering*, 15:188–193, 1964.

[PS82] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice–Hall, Inc., Englewood Cliffs, New Jersey, 1982.

[RP84] R.R. Razouk and C.V. Phelps. Performance Analysis using Timed Petri Nets. In R.M. Keller, editor, *Proceedings of the 1984 International Conference on Parallel Processing*, pages 126–128, IEEE Computer Society Press, Silver Spring, Maryland, 1984.

[SB96a] J. Symanzik and A.L. Baker. Formalized Data Flow Diagrams and Their Relation to Other Computational Models. Technical Report 96–20, Iowa State University, Department of Computer Science, 226 Atanasoff Hall, Ames, Iowa 50011, December 1996. Available by anonymous ftp from ftp.cs.iastate.edu or by e–mail from almanac@cs.iastate.edu.

[SB96b] J. Symanzik and A.L. Baker. Timed Data Flow Diagrams. Technical Report 96–23, Iowa State University, Department of Computer Science, 226 Atanasoff Hall, Ames, Iowa 50011, December 1996. Available by anonymous ftp from ftp.cs.iastate.edu or by e–mail from almanac@cs.iastate.edu.

[Sch84] R. Schassberger. An Aggregation Principle for Computing Invariant Probability Vectors in Semi–Markovian Models. In G. Iazeolla, P.J. Courtois, and A. Hordijk, editors, *Mathematical Computer Performance and Reliability*, pages 259–273, Elsevier (North–Holland), Amsterdam, 1984.

[TP89] T.H. Tse and L. Pong. Towards a Formal Foundation for DeMarco Data Flow Diagrams. *The Computer Journal*, 32(1):1–12, February 1989.

[Wal83] B. Walter. Timed Petri–Nets for Modelling and Analyzing Protocols with Real–Time Characteristics. In H. Rudin and C.H. West, editors, *Protocol Specification, Testing, and Verification, III*, pages 161–172. Elsevier (North–Holland), 1983.

[WBL93] T. Wahls, A.L. Baker, and G.T. Leavens. An Executable Semantics for a Formalized Data Flow Diagram Specification Language. Technical Report 93–27, Iowa State University, Department of Computer Science, 226 Atanasoff Hall, Ames, Iowa 50011, November 1993. Available by anonymous ftp from ftp.cs.iastate.edu or by e–mail from almanac@cs.iastate.edu.

[WM85a] P.T. Ward and S.J. Mellor. *Structured Development for Real–Time Systems*, Volume 1: Introduction and Tools. Yourdon, Inc., New York, New York, 1985.

[WM85b] P.T. Ward and S.J. Mellor. *Structured Development for Real–Time Systems*, Volume 2: Essential Modeling Techniques. Yourdon, Inc., New York, New York, 1985.

[Woo93] H.-S. Woo. *On Deterministic and Markovian Production Systems.* PhD Thesis, Iowa State University, Ames, Iowa, 50011, 1993.

[You89] E. Yourdon. *Modern Structured Analysis.* Yourdon Press Computing Series. Prentice–Hall, Englewood Cliffs, New Jersey, 1989.

[Yua86] C.Y. Yuan. Process Periods and System Reconstruction. In G. Rozenberg, editor, *Lecture Notes in Computer Science Vol. 222: Advances in Petri Nets 1985*, pages 122–141, Springer–Verlag, Berlin, Heidelberg, 1986.

[Zub80] W.M. Zuberek. Timed Petri Nets and Preliminary Performance Evaluation. In *IEEE Proceedings of the 7th Annual Symposium on Computer Architecture, May 1980, La Baule, France*, pages 88–96, 1980.