

Bisimulation and Monotonicity for Fault Tolerance

April 16, 2019

Abstract

The basic concept in fault tolerant system modelling is that anticipated faults being outside of our control may or may not occur. According to the standard process algebras semantics anticipated faults must occur when the system is not able to follow the standard behaviour. The result is that one fault may compensate for the effects of another fault. A typical example is the loose and the creation of messages in a communication channel. This make bisimulation not monotonic.

1 Introduction

Process algebras are a standard model for specifying concurrent systems. In order to specify a process and to prove its correctness, it is useful to decide which properties of the model are relevant and which can be ignored. Following [?], the semantics of processes is given in terms of labelled transition systems, which allow to describe their behaviour in details, including particulars of their internal computations. It is common to define equivalences over labelled transition systems to define properties which consider a particular subset of the details of the specification. We can indentify a process with its equivalent class, according to the defined equivalence relation.

Since for a fault tolerant system it is not its internal structure which is of interest but its effects on the environment and its reactions to stimuli from the environment, *observational equivalence* can be applied to check the correctness of a fault tolerant system design.

A fault tolerant system is generally specified as a set of fault tolerant processes which may employ various techniques to detect, confine and recover from erroneous states. The behaviour of a system can be divided into *normal* (or *correct*) *behaviour*, the behaviour of the system when no fault occurs, and *failing behaviour*, the behaviour of the system in presence of faults. The failing behaviour may be different for different kind of faults and we refer to the set of failing behaviours as *failure mode*. Important in fault tolerant system design is the *fault hypothesis* which gives the constraint on how faults are supposed to occur in the system. Given a set of anticipated faults, under a particular failure mode, a system is designed to tolerate the occurrence of faults as stated by the fault hypothesis if and only if the occurrence of such faults in the system does not inhibit the system's ability to correctly satisfy its specification.

Since for a fault tolerant system it is not its internal structure which is of interest but its effects on the environment and its reactions to stimuli from the environment, *weak bisimulation equivalence* (named also *observational equivalence*) can be applied to check the correctness of a fault tolerant system design. Observational equivalence, first introduced in [?], ignore properties which cannot be observed in the finite interval of time and is based in fact on the idea that the behaviour of the system is determined by the way it interacts with the environment: two systems are observational equivalent whenever no observation can distinguish them.

As stated in [], the problem of using bisimulation equivalence for fault tolerance is that proving fault tolerance towards a given set of faults does not imply fault tolerance towards a subset of those faults. A typical example is that of compensating faults as the loss and the creation of messages in a communication channel. According to the standard process algebras semantics anticipated faults must occur when the system is not able to follow the standard behaviour. The result is that one fault may compensate for the effects of another fault.

This paper presents a modelling approach of fault tolerance which makes bisimulation monotonic. The basic idea is that of modelling anticipated faults as events which may or may not occur. We reduce the proof of fault-tolerance to correctness with respect to the specification in the presence of all faults. We prove that this corresponds to prove correctness without taking faults into account as well as to prove correctness under any given subset of faults.

2 Background

Given a set of observable actions A , we define $\mathcal{A} = A \cup \{\tau\}$, where τ is internal and represents the outcome of a joint activity (interaction) between two processes. The set A is partitioned between actions a and their complements \bar{a} . We assume $\bar{\bar{a}} = a$. Let $L \subseteq A$, $\alpha \in \mathcal{A}$ and $f : A \rightarrow A$ where $f(\bar{a}) = \bar{f(a)}$. The language of finite processes, ranged over by P , is defined by the following grammar:

$$P ::= 0 \mid \alpha \cdot P \mid P + P' \mid P \parallel P' \mid P \setminus L \mid P[f]$$

Informally, 0 is the process which is incapable of any action. $\alpha \cdot P$ is the process that executes action α and then behaves like P . $P + P'$ is the nondeterministic choice between the process P and the process P' . $P \parallel P'$ is the parallel composition of process P and process P' where P and P' can proceed independently but also synchronise on complementary actions, performing the action τ . $P \setminus L$ is the restriction of P to a process like P apart from actions in L and their complements that cannot be executed. Finally, $P[f]$ behaves like P apart from all actions a renamed into $f(a)$.

The syntax permits a two layered design of process terms. The first level is related to *sequential regular terms*, the second one to general terms of sub-processes supporting communication and action renaming or restriction. *Sequential terms* are generated by \cdot and $+$ operators. The specification of *general terms* requires the \parallel operator.

Table 1 shows the structural operational semantics of the language in terms of labelled transition systems [?] which describe the behaviour of a process in terms of states and transition between states.

Definition 2.1 *A labelled transition system is a 4-tuple $\mathcal{Q} = (Q, q_0, A \cup \{\tau\}, \rightarrow)$, where: Q is a finite set of states; q_0 is the initial state; A is a finite set of observable actions and τ is the internal action; $\rightarrow \subseteq Q \times A \cup \{\tau\} \times Q$ is the transition relation.*

We write $q \xrightarrow{a} q'$ to denote the move from the state q to the state q' by executing the action a . In particular, $q \xrightarrow{\tau} q'$ denotes an internal move from q to q' .

Equivalence relations over labelled transition systems relate states according to performable actions [?]. Weak bisimulation equivalence (bisimulation equivalence in the following) abstract unobservable moves during observation. We shall use the transition relation defined as follows:

$$\forall a \in A, \xRightarrow{a} \stackrel{\text{def}}{=} (\xrightarrow{\tau})^* \xrightarrow{a} (\xrightarrow{\tau})^*$$

Operator	Operational rules
$a \cdot P$	$\frac{}{a \cdot P \xrightarrow{\alpha} P}$
$P + Q$	$\frac{P \xrightarrow{\alpha} P' \quad Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} P' \quad P + Q \xrightarrow{\alpha} Q'}$
$P \parallel Q$	$\frac{P \xrightarrow{a} P' \quad Q \xrightarrow{a} Q' \quad P \xrightarrow{a} P', Q \xrightarrow{\bar{a}} Q'}{P \parallel Q \xrightarrow{a} P' \parallel Q' \quad P \parallel Q \xrightarrow{\tau} P' \parallel Q'}$
$P \setminus L$	$\frac{P \xrightarrow{a} P'}{P \setminus L \xrightarrow{a} P' \setminus L} \quad a, \bar{a} \notin L$
$P[f]$	$\frac{P \xrightarrow{a} P'}{P[f] \xrightarrow{f(a)} P' \setminus L}$

Table 1: Operational semantics

where \star means zero or any number of times.

Bisimulation is then defined upon the \xRightarrow{a} .

Definition 2.2 (bisimulation) *Bisimulation is the maximal fixed point of the functional \mathcal{U} on the set of binary relations B on P , $(P_1, P_2) \in \mathcal{U}(B)$ iff:*

$$\text{whenever } P_1 \xrightarrow{a} P'_1 \text{ then } \exists P'_2, P_2 \xRightarrow{a} P'_2 \text{ and } (P'_1, P'_2) \in B \quad (1)$$

$$\text{whenever } P_2 \xrightarrow{a} P'_2 \text{ then } \exists P'_1, P_1 \xRightarrow{a} P'_1 \text{ and } (P'_1, P'_2) \in B \quad (2)$$

We write $P_1 \approx P_2$.

3 Modelling

Faults are formalized by explicit observable actions. Let Φ includes an action for every kind of fault. The alphabet of observable actions of processes is $A \cup \Phi$. The execution of an action φ in the process corresponds to the occurrence of a fault in the modelled system. The effect of the occurrence of a fault in the system is that of transforming the normal behaviour of the system in the failing one.

The failure mode and possible recover is added to the process. **Assumption:** we assume at most one fault at a time, so no other fault can occur in this phase. Moreover we assume unrelated faults. **IMPORTANT!!!**

Given a process P , we use the following notation:

- FP denotes the process P after the introduction of the faults and the corresponding failure mode
- $\forall \mathcal{I} \subseteq \Phi$, $H_{\mathcal{I}}$ is a process modelling the occurrence of faults belonging to \mathcal{I}
- $\forall \mathcal{I} \subseteq \Phi$, $FP_{\mathcal{I}}$ is the process FP under the assumption that only faults belonging to \mathcal{I} may occur.

Definition 3.1 (well-formed) *Given a set Φ of faults, a process $FP = (Q, q_0, A \cup \Phi \cup \{\tau\}, \rightarrow)$ is well-formed iff:*

$\forall q \in Q$ such that $\exists q' \in Q, \exists \varphi \in \Phi$ such that $q \xrightarrow{\varphi} q'$, P satisfies the following conditions

- 1) $\exists q'' \in Q$ such that $q \xrightarrow{\tau} q''$
- 2) $\forall a \in A, \nexists q'' \in Q$ such that $q \xrightarrow{a} q''$

Informally, FP is well-formed if every state q allowing a transition labelled by a fault action, does not contains transitions labelled by actions belonging to A .

Given a faulty free process P , we insert faults into sequential terms and we follows the following lines, see figure 1:

$\forall q \in Q$, we insert a state q' and set *rightarrow* as follows:
there is a transition labelled τ from q to q' ;

for every transition $q \xrightarrow{a} q_1$, we add the transition $q' \xrightarrow{a} q_1$;

for every fault φ in the state q , we add the transition $q \xrightarrow{\varphi} \bar{q}$, where \bar{q} is the state reached after the occurrence of the fault.

We add the states and the transitions for specifying the failure mode (\bar{q} may be a state in Q or a new state)

Figure 1: Adding faults

We specify a fault hypothesis as a process allowing certain faults, see figure 2:

$H_{\mathcal{I}} = (\{q_0\}, q_0, \mathcal{I}, \rightarrow)$, where \rightarrow is defined as follows: $\forall \varphi \in \mathcal{I}, q_0 \xrightarrow{\varphi} q_0$

Figure 2: Fault hypothesis $\mathcal{I} = \{f_1, f_2\}$

Definition 3.2 Given P , FP and Φ , FP is monotonic for P if

- $P \approx FP_{\Phi}$ implies
- $\forall \mathcal{I} \subseteq \Phi, P \approx FP_{\mathcal{I}}$
- where
- $FP_{\mathcal{I}} = (FP \parallel H_{\mathcal{I}}) \setminus \Phi$

We specify processes under a certain assumption about faults $FP_{\mathcal{I}}$ by parallel composition and restriction operator.

Definition 3.3 (**= relation**) Given P_1 and P_2 , $P_1 \models P_2$ iff
for every state in P_2 there is a state in P_1 having the same observable transitions up to τ .

Lemma 3.1 $FP \setminus \Phi \approx P$.

Proof. We distinguish two cases.

P is a sequential term. The proof follows directly by the transition relation of the two transition systems. Classe equivalenza stati (q, q') .

P is a general term. Assume $P = P_1 \parallel P_2$.

We have $FP = FP_1 \parallel FP_2$. By the point above, $FP_1 \approx P_1$ And $FP_2 \approx P_2$. The proof follows by compositionality of bisimulation relation. Similarly for restriction and relabelling.

NOTA: P is a transition system.

Classi di equivalenza degli stati. Per ogni stato di P abbiamo:

- 1) nello stato q era ammesso un guasto: ci sono due stati in FP equivalenti a q e

sono lo stato q stesso lo stato q' raggiungibile con τ .

2) nello stato q non erano ammessi guasti, c'lo stato q in FP equivalente a q in P .

$P = P_1 + P_2$ come sopra per P_1 e P_2 singolarmente

$P = P_1 \parallel P_2$. Applicando la regola del \parallel della semantica operativa e per induzione sulla lunghezza della computatione.

Passo 1

$$\begin{array}{c}
\frac{P_1 \xrightarrow{a} P'_1 \quad P_2 \xrightarrow{a} P'_2 \quad P_1 \xrightarrow{a} P'_1, P_2 \xrightarrow{\bar{a}} P'_2}{P_1 \parallel P_2 \xrightarrow{a} P'_1 \parallel P_2 \quad P_1 \parallel P_2 \xrightarrow{a} P_1 \parallel P'_2 \quad P_1 \parallel P_2 \xrightarrow{\tau} P'_1 \parallel P'_2} \\
\frac{P_1 \xrightarrow{a} P'_1}{P_1 \parallel P_2 \xrightarrow{a} P'_1 \parallel P_2} \\
\frac{P_1 \parallel P_2 \xrightarrow{\tau} P'_1 \parallel P_2}{FP_1 \xrightarrow{\tau} FP'_1}
\end{array}$$

Passo n vero. Dimostriamo passo $n+1$

Theorem 3.1 *Given P , FP and Φ . If FP is well-formed and $P \approx FP \setminus \Phi$ then FP is monotonic for P .*

Proof. *The proof is given by showing that, given $\mathcal{I} \subseteq \Phi$, for every state in $FP_{\mathcal{I}}$ there is a state in FP having the same observable transitions up to τ transitions, we write $FP \models FP_{\mathcal{I}}$.*

4 Pezzi

Various notions of systems equivalence based on the reactions of systems to stimuli from the outside world have been defined. Generally, these notions allow abstraction from unwanted details in models of the systems.

Assumptions

- system S as a parallel composition of synchronising processes
- every kind f of fault modelled by a special action f .
- failure mode \subseteq observable actions
- methodology proposed in CJ for studying the behaviour of fault tolerant systems under a fault assumptions using constraint processes

Definition 4.1 (well-formed) *Given a process $P = (Q, q_0, A \cup \{\tau\}, \rightarrow)$, and Φ , $FP = (S, s_0, A \cup \Phi \cup \{\tau\}, \rightarrow_s)$ is well-formed if every state s is the non deterministic choice between internal action seguita dall'azione fault seguita da possibili azioni osservabili del processo and internal action followed by the standard behaviour of the system.*

$\forall q \in Q$, we insert a state q' and set \rightarrow as follows:
there is a transition labelled τ from q to q' ;

for every transition $q \xrightarrow{a} q_1$, we add the transition $q' \xrightarrow{a} q_1$;

for every fault φ in the state q , we add the transition $q \xrightarrow{\varphi} \bar{q}$, where \bar{q} is the state reached after the occurrence of the fault.

We add the states and the transitions for specifying the failure mode (\bar{q} may be a state in Q or a new state)