A Model for Probabilistic Reasoning on Assume/Guarantee Contracts

Benoît Delahaye, Benoit Caillaud

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Model for Probabilistic Reasoning on Assume/Guarantee Contracts

Benoît Delahaye, Université de Rennes 1 / IRISA
— Benoît Caillaud, INRIA / IRISA

N° 6719

Novembre 2008

Thèmes COM et SYM

ISSN 0249-6399 ISRN INRIA/RR--6719--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
RENNES - BRETAGNE ATLANTIQUE

A Model for Probabilistic Reasoning on Assume/Guarantee Contracts

Benoît Delahaye, Université de Rennes 1 / IRISA

, Benoît Caillaud, INRIA / IRISA

Thèmes COM et SYM — Systèmes communicants et Systèmes symboliques
Équipe-Projet S4

Rapport de recherche n° 6719 — Novembre 2008 — 15 pages

Abstract: In this paper, we present a probabilistic adaptation of an Assume/Guarantee contract formalism. For the sake of generality, we assume that the extended state machines used in the contracts and implementations define sets of runs on a given set of variables, that compose by intersection over the common variables. In order to enable probabilistic reasoning, we consider that the contracts dictate how certain input variables will behave, being either non-deterministic, or probabilistic; the introduction of probabilistic variables leading us to tune the notions of implementation, refinement and composition. As shown in the report, this probabilistic adaptation of the Assume/Guarantee contract theory preserves compositionality and therefore allows modular reliability analysis, either with a top-down or a bottom-up approach.

Key-words: Assume/Guarantee Reasoning, Contracts, Probabilistic reasoning, Reliability analysis.

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu, 35042 Rennes Cedex
Téléphone : +33 2 99 84 71 00 — Télécopie : +33 2 99 84 71 71

Un modèle de raisonnement probabiliste basé sur les contrats Assume/Guarantee

Résumé : Ce document présente une adaptation probabiliste d'un formalisme de contrats *Assume/Guarantee*. On supposera, dans le but d'être le plus général possible, que les machines tats tendues utilisées pour les contrats et implémentations définissent des ensembles d'histoires sur un ensemble de variables donn, et qu'elles se composent par intersection sur les variables communes. Pour permettre un raisonnement probabiliste, on considère que les contrats précisent le comportement des variables d'entrées, non-déterministes ou probabilistes. Le fait de considérer des entrées probabilistes nécessite une adaptation des notions d'implémentation, composition et raffinement. Ce rapport montre que cette adaptation probabiliste de la théorie des contrats *Assume/Guarantee* préserve la compositionalité, et permet de ce fait une analyse de fiabilité modulaire, que ce soit par une approche ascendante ou descendante.

Mots-clés : Raisonnement Assume/Guarantee, Contrats, Raisonnement probabiliste, Analyse de fiabilité.

1 Introduction

Several industrial sectors involving complex embedded systems have recently experienced deep changes in their organization, aerospace and automotive being the most prominent examples. In the past, they were organized around vertically integrated companies, supporting in-house design activities. These sectors have now evolved into more specialized, horizontally structured companies: equipment suppliers and OEMs. OEMs perform system design and integration by importing/reusing entire subsystems provided by equipment suppliers. As a consequence, part of the design load has been moved from OEMs to suppliers. An inconvenient of this change is the increased occurrence of late error discovery, system level design errors uncovered at integration time. This is particularly true for system reliability, for state of the art reliability analysis techniques are not modular [HR94, SA96].

A corrective action, taken in the last decade is that the OEMs now focus on the part of the system design at the core of their business, and as far as possible, rely on industry-wide standard platforms. This has an impact on design methods and modeling formalisms: Virtual prototyping, design space exploration are required early in the design cycle. Component based design has emerged as the most promising technique to address the challenges resulting from this new organization of the industry.

However, little has been done regarding the capture of reliability requirements, their formalization in behavioural models and the verification techniques capable of analyzing in a modular way the reliability aspects of a system, at an early stage of design. The paper contributes to solve these issues: The semantics foundations presented in this paper consists in a mathematical formalism designed to support a component based design methodology and to offer modular and scalable reliability analysis techniques. At its basis, the mathematical formalism is a language theoretic abstraction of systems behaviour. This basic formalism can be instantiated to cover several aspects, including functional, timeliness, hybrid and reliability [BCP07]. This report presents the reliability aspect.

The central concept of the formalism is the notion of contract, built on top of a basic behavioural formalism. Contracts allow to distinguish hypotheses on a component from hypotheses made on its environment. Contracts are central to component based design methodologies.

This paper focuses on developing a compositional theory of probabilistic contracts, capable of capturing reliability aspects of components and systems. The key contributions are the definition of probabilistic satisfaction, composition and refinement relations that ensure that they will be compositional.

The paper is organized as follows: In the first section we present the Assume/Guarantee formalism upon which we built our probabilistic theory. In the second section we formally define the probabilistic Assume/Guarantee theory we developed and we prove that it is compositional. In a third section we compare our work to classical related formalisms like pCTL and pCTL* [ASBSV95, HJ89], or developed more recently, such as Dynamic Fault Trees [BCS07] and Arcade [BCH$^+$08].

2 Classical Assume/Guarantee Reasoning

The model we have built is based upon the notion of components and Assume/Guarantee reasoning. In this section, we will define the background upon which the report is based. In a first subsection we will give the definitions we used for contracts and implementations. Then we will present the basic operations already existing on contracts and the main theorems we want to preserve in our probabilistic adaptation.

2.1 Contracts and Implementations

In order to define contracts and implementations, we need to consider the abstract notion of "assertion".

Definition 1 *An assertion $E = S :: \sigma$ possesses a set of ports and variables (its signature, σ) through which it interacts with its environment. S is identified with the set of runs it defines or accepts, each run assigning a history for each variable and port. If necessary, the inverse projection of $E = S :: \sigma$ on $\sigma' \supseteq \sigma$ will be denoted by $E \uparrow^{\sigma'}$.*

We assume that there exists a complementation operator for an assertion E, relative to its signature σ. It is denoted by $\neg E$. Assertions compose by intersection over the common sets of ports and variables (assuming the appropriate inverse projections have been performed to equalize the involved signatures). We will denote products either by $E_1 \times E_2$ or $E_1 \cap E_2$ equivalently.

$$E_1 \times E_2 = E_1 \cap E_2 = S_1 \uparrow^{\sigma} \cap S_2 \uparrow^{\sigma} :: \sigma, \text{ with } \sigma = \sigma 1 \sup \sigma 2$$

With these notations and definitions, we will be able to define implementations and contracts.

Definition 2 *An implementation is an assertion, i.e. a set of runs with a given signature.*

We will use the symbol $M = S_M :: \sigma_M$ to refer to implementations. They are ordered by inclusion over the runs they contain (one more time assuming that the appropriate inverse projections have been performed). We will say that an implementation M refines an implementation M' with respect to the signature σ, written $M \preceq^{\sigma} M'$, if and only if $S_M \subseteq^{\sigma} S_{M'}$, i.e. $S_M \uparrow^{\sigma} \subseteq S_{M'} \uparrow^{\sigma}$.

Composition preserves implementation refinement.

Definition 3 *A contract $C :: \sigma$ is a pair of assertions $(A :: \sigma_A, G :: \sigma_G)$ with $\sigma \supseteq \sigma_A \cup \sigma_G$.*

- *A is the* **assumption***;*
- *G is the* **guarantee***, i.e. the promised behavior, under the hypothesis that A holds.*

Note that for a contract $C :: \sigma = (A :: \sigma_A, G :: \sigma_G)$, we can consider the equivalent contract $C' :: \sigma = (A \uparrow^{\sigma} :: \sigma, G \uparrow^{\sigma} :: \sigma)$. Whenever convenient, we will thus suppose that both assertions of a contract C have the same signature

and we will denote respectively the assumption, the guarantee and the common signature of C by A_C, G_C and σ_C.

The following definition of satisfaction will precise the interpretation we make of a contract.

Definition 4 *An implementation M satisfies a contract $C :: \sigma_C = (A_C, G_C)$ (written $M \models C$) if and only if*

$$M \cap A_C \subseteq^{\sigma_C} G_C$$

Satisfaction can be checked using equivalent formulas:

$$M \models C \iff M \subseteq G_C \cup \neg A_C \iff M \cap (A_C \cap \neg G_C) = \emptyset$$

From these equivalent definitions, we can show that there exists a unique maximal implementation M_C satisfying a contract C:

$$M_C = (G_C \cup \neg A_C) :: \sigma_C$$

This maximal implementation is to be interpreted as the implication $A \Rightarrow C$. We can prove that an implementation M satisfies contract $C = (A_C, G_C)$ if and only if it satisfies the equivalent contract (A_C, M_C), and if and only if $M \preceq^{\sigma_C} M_C$. We will say that a contract $C :: \sigma = (A :: \sigma_A, G :: \sigma_G)$ is in **canonical form** when $G = M_C$, or equivalently when $\neg A \subseteq G$ or $G \subseteq \neg A$, and $\sigma = \sigma_A = \sigma_G$. As the canonical form of a contract is unique and the satisfaction of the contract equivalent to the satisfaction of its canonical form, we will consider only contracts in canonical form in the rest of this document.

2.2 Operations on Contracts

The notion of composition of contracts formalizes how contracts attached to different components of a system should be combined in order to represent one single component. If $C_1 :: \sigma_1 = (A_1, G_1)$ and $C_2 :: \sigma_2 = (A_2, G_2)$ are two contracts defined as in the previous section, their composition should respect some rules. First, their promises should be composed, as we want to guarantee that both G_1 and G_2 must be respected. Remember that composition is the intersection of the two assertions, after computing the appropriate inverse projections in order to equalize the sets of variables and ports.

Regarding the assumptions, we also want to assume that both A_1 and A_2 are respected, but we must consider the case when the second contract guarantees that part of the assumption of the first one is respected and vice-versa. A run satisfying the assumption of the composition should consequently either satisfy both A_1 and A_2 or be made unacceptable by the composition of the guarantees. Thus the following definition:

Definition 5 *Let $C_1 :: \sigma_1 = (A_1, G_1)$ and $C_2 :: \sigma 2 = (A_2, G_2)$ be contracts, we define $C_1 \parallel C_2$ to be the contract $C :: \sigma = (A, G)$ such that:*

- $\sigma = \sigma_1 \cup \sigma_2$;
- $A = (A_1 \uparrow^\sigma \cap A_2 \uparrow^\sigma) \cup \neg(G_1 \uparrow^\sigma \cap G_2 \uparrow^\sigma)$;
- $G = G_1 \uparrow^\sigma \cap G_2 \uparrow^\sigma$.

Remark that the so defined contract is in canonical form.

With the above definition, we can prove that the composition preserves the implementation relation.

Lemma 1 *If* $M_1 \models C_1$ *and* $M_2 \models C_2$ *then* $M_1 \times M_2 \models C_1 \parallel C_2$.

Proof :
As the two contracts are supposed in canonical form, we have $M_i \subseteq^{\sigma_i} G_i$. *Thus* $M_1 \times M_2 \subseteq^{\sigma_1 \cup \sigma_2} G_1 \cap G_2$, *and* $M_1 \times M_2 \models C_1 \cap C_2$.

□

Next, we will need to build a refinement relation. Intuitively, this relation must be compatible with the composition operation and the implementation relation. We will thus say that a contract C refines another contract C' if it assumes less and guarantees more:

Definition 6 *The contract* $C :: \sigma = (A, G)$ *refines the contract* $C' :: \sigma' = (A', G')$, *written* $C \preceq C'$, *if and only if* $\sigma \subseteq \sigma'$, $A \supseteq^{\sigma'} A'$ *and* $G \subseteq^{\sigma'} G'$.

We can now prove the following properties (the proof is quite straightforward and left to the reader):

Lemma 2 *If* M *is an implementation,* C_1, C_2, C_3 *and* C_4 *four contracts,*

1. *If* $M \models C_1$ *and* $C_1 \preceq C_2$, *then* $M \models C_2$.
2. *If* $C_1 \preceq C_2$ *and* $C_3 \preceq C_4$ *then* $C_1 \parallel C_3 \preceq C_2 \parallel C_4$.

3 Extension to Probabilistic Approach

In this section we will adapt the definitions presented above in order to be able to express probabilistic properties, like reliability, while preserving compositionality.

What we want to express is the affirmation "This particular implementation M satisfies the given contract C with level α", meaning that given the information of the contract C, we can prove that the (probabilistic) measure of the runs of M that do not satisfy the contract C (i.e. that are within the assumptions but outside of the guarantees) is not above $1 - \alpha$. More precisely, we still want to consider non-probabilistic assertions, but we want to be able to express that the environment may induce randomness in our assertions. We must therefore precise which of the variables/ports associated to the contract are controlled (internal variables for instance) or uncontrolled (controlled by the environment). We can then choose a subset of the uncontrolled ports to be subject to probability distributions. There will then remain a subset of the uncontrolled ports that we will consider non-deterministic. As a consequence, the signature of each assertion will be divided into two disjoint sets of ports, controlled or uncontrolled $\sigma = \mathbf{u} \uplus \mathbf{c}$ (note that for a contract, there will only be one such signature).

3.1 Probabilistic Contracts, Implementations and Satisfaction

Definition 7 *A probabilistic contract is a tuple* $\mathcal{C} = (C, \mathbf{p}, \mathbb{P})$ *with*

- $C = (\mathbf{u}, \mathbf{c}, A, G)$ *a non-probabilistic contract;*
- $\mathbf{p} \subseteq \mathbf{u}$ *a set of uncontrolled ports;*
- $\mathbb{P}$ *a probability distribution over the set of all histories of* $\mathbf{p}$.

Note that the probability distribution is attached to the contract itself and not to the implementation. We can therefore give our contract to a supplier saying "Knowing that the histories of the ports of $\mathbf{p}$ will follow this distribution, can you build an implementation ensuring that 90% of the runs will satisfy the contract?". Let's now formally define this probabilistic satisfaction relation:

Definition 8 *An implementation* M *satisfies probabilistic contract* $\mathcal{C} = (C, \mathbf{p}, \mathbb{P})$ *with level* β *(written* $M \models_\beta \mathcal{C}$*) iff* $\mathbf{u}_M \subseteq \mathbf{u}_C, \mathbf{c}_M = \mathbf{c}_C$ *and*

$$\mathcal{M}(M \subseteq G_\mathcal{C}) \geq \beta$$

N.B.: We still consider that C *is in canonical form.*

The predicate $M \subseteq G_C$ is a reference to the set of all histories of the ports in $\mathbf{p}$ that ensure the induced behaviors of the implementation M (i.e. the inverse projection on the set of runs of M) is included in the guaranteed behaviors, whatever other non-deterministic choices have been made. Formally: If ω is one possible history of the ports in $\mathbf{p}$, we call Ω the set of all such histories,

$$\mathcal{M}(M \subseteq G_\mathcal{C}) = \mathbb{P}(\{\omega \in \Omega \mid \{w\} \cap M \subseteq^{\sigma_\mathcal{C}} G_\mathcal{C}\})$$

This means that we measure the set of histories of $\mathbf{p}$ that ensure that the runs of M will be included in the guaranteed behavior, whatever non-deterministic choices are. Finally, $M \models_\beta \mathcal{C}$ means that the probability that $M \models C$ w.r.t. the distribution of histories on the probabilistic ports is higher than β, whatever non-deterministic choices are.

3.2 Probabilistic Composition

The probabilistic point of view makes it more complicated to compose contracts. As the distributions on the probabilistic ports are linked to the contracts, what we absolutely do not want is to compose two contracts whose probabilistic ports overlap. Moreover we also want to avoid the case where a probabilistic port for the first contract is controlled by the second one (i.e. the first considers an input port as probabilistic, but this port is an output of the second). In order to keep a quite simple definition, and because we think this is not too major a restriction, we will only define the composition of two contracts when they have compatible sets of controlled/uncontrolled ports (i.e. $\mathbf{c}_1 \cap \mathbf{c}_2 = \emptyset$). Thus the following definition:

Definition 9 *If* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *are 2 probabilistic contracts, their parallel composition* $\mathcal{C}_1 \parallel \mathcal{C}_2$ *is defined if and only if*

1. $C_1 \parallel C_2$ *is defined (i.e.* $\mathbf{c}_1 \cap \mathbf{c}_2 = \emptyset$*);*
2. $\mathbf{p}_1$ *and* $\mathbf{p}_2$ *are* **disjoint** *sets of* **uncontrolled** *ports in* $C_1 \parallel C_2$ *(i.e. ports that are neither controlled by* C_1 *or* C_2*).*

Then we have

$$\mathcal{C}_1 \parallel \mathcal{C}_2 = (C, \mathbf{p}, \mathbb{P}) \text{ with } \begin{cases} C = C_1 \parallel C_2 \\ \mathbf{p} = \mathbf{p}_1 \uplus \mathbf{p}_2 \\ \mathbb{P} = \mathbb{P}_1 \times \mathbb{P}_2 \end{cases}$$

The above definition makes it impossible to compose two contracts whose probabilistic and controlled ports overlap. This could be seen as a major restriction but there is a way to make such contracts compatible. Consider two probabilistic contracts $\mathcal{C}_1$ and $\mathcal{C}_2$, and suppose that the port x is controlled by $\mathcal{C}_1$, but considered as probabilistic by $\mathcal{C}_2$. If we want to compose $\mathcal{C}_1$ and $\mathcal{C}_2$, we have to make the port x non-probabilistic in $\mathcal{C}_2$. Thus we consider the contracts $\mathcal{C}_2' = (C_2, \mathbf{p}_2 \setminus \{x\}, \mathbb{P}_2')$ and $\mathcal{C}_x = (C_x, \{x_p\}, \mathbb{P}_2^x)$. $\mathbb{P}_2'$ is the restriction of $\mathbb{P}_2$ without x, $\mathbb{P}_2^x$ is the probability distribution considering only x, and C_x is a non-probabilistic contract we will call a *wrapper*, with three uncontrolled ports (x_p, x_c and s) and one controlled port x. This wrapper selects with a non-deterministic port $s \in \{p, c\}$ the value that will be given to x between the probabilistic one and the one given by $\mathcal{C}_1$. Composing $\mathcal{C}_2'$ with $\mathcal{C}_x$ thus enables us to compose it with $\mathcal{C}_1$ (renaming x to x_c). This is illustrated in Fig.1, where thick triangles denote probabilistic ports. The wrong version is on the top and the correct wrapped one is on the bottom.

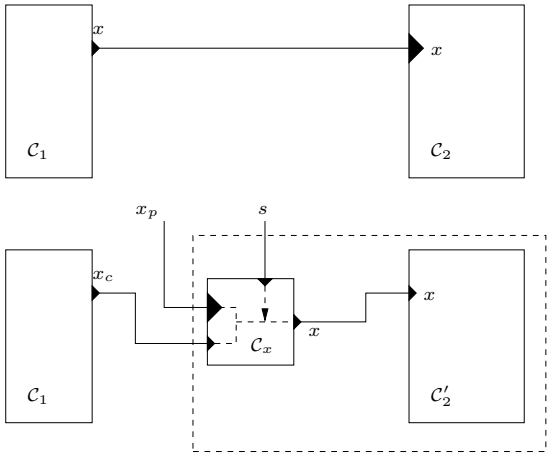


Figure 1: Illustrating the wrapper mechanism

We now prove that the composition is compatible with the satisfaction relation. The proof of this theorem relies on the fact that the probabilistic ports of contracts $\mathcal{C}_1$ and $\mathcal{C}_2$ are disjoint.

Theorem 1 *If* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *are 2 probabilistic contracts that can be composed,* M_1 *and* M_2 *2 implementations such that* $M_i \models_{\beta_i} \mathcal{C}_i$ *for* $i = 1, 2$ *then*

$$M_1 \times M_2 \models_{\beta_1 \cdot \beta_2} \mathcal{C}_1 \parallel \mathcal{C}_2.$$

Proof :

The intuition behind this proof is to show that separate histories on the ports $\mathbf{p}_1$ *and* $\mathbf{p}_2$*, each ensuring that its particular implementation behaves correctly, also ensure that the composition of the implementations will behave correctly. In short, the product of two correct histories is correct w.r.t. the composition of the contracts and implementations. This will be true because every "correct" history satisfies the corresponding contract* whatever non-deterministic choices are*.*

Let's consider histories w_1 *and* w_2*, respectively on the sets* $\mathbf{p}_1$ *and* $\mathbf{p}_2$*, such that* $M_i \cap \{w_i\} \subseteq^{\sigma_i} G_i$*.*

As we said before, composition is by intersection over the common ports and variables. We therefore begin with the inverse projection over the set of variables we want to consider, and then intersect the runs.

It is clear that $\forall w$*,* $\{w\} \cap M_1 \cap M_2 \subseteq^{\sigma} \{w\} \cap M_1$*, with* $\sigma = \sigma_1 \cup \sigma_2$ *(assuming the inverse projections are correctly done on both sides). Moreover,* $\{w_1 \times w_2\} \cap^{\sigma} M_i \subseteq^{\sigma} \{w_j\} \cap^{\sigma} M_i$*, whatever* i *or* j*. As a consequence:*

$$\begin{aligned} \{w_1 \times w_2\} \cap M_1 \cap M_2 &\subseteq^{\sigma} \{w_1 \times w_2\} \cap M_1 \\ &\subseteq^{\sigma} \{w_1\} \cap M_1 \subseteq^{\sigma} G_1 \end{aligned}$$

$$\begin{aligned} \{w_1 \times w_2\} \cap M_1 \cap M_2 &\subseteq^{\sigma} \{w_1 \times w_2\} \cap M_2 \\ &\subseteq^{\sigma} \{w_2\} \cap M_2 \subseteq^{\sigma} G_2 \end{aligned}$$

$$\Rightarrow \{w_1 \times w_2\} \cap M_1 \cap M_2 \subseteq^{\sigma} G_1 \cap G_2$$

As a consequence, $\{w_i\} \cap M_i \subseteq^{\sigma_i} G_i$ *implies that*

$$\{w_1 \times w_2\} \cap M_1 \cap M_2 \subseteq^{\sigma_1 \cup \sigma_2} G_1 \cap G_2$$

And finally

$$\mathbb{P}(\{w \mid \{w\} \cap M_1 \cap M_2 \subseteq^{\sigma_1 \cup \sigma_2} G_1 \cap G_2\}) \geq \beta_1 \cdot \beta_2$$

Thus $\mathbb{P}(M_1 \cap M_2 \subseteq \mathcal{C}_1 \parallel \mathcal{C}_2) \geq \beta_1 \cdot \beta_2$*.*

□

Note that we cannot find a better bound that $\beta_1 \cdot \beta_2$, because if the contracts are independent ($\sigma_1 \cap \sigma_2 = \emptyset$), we clearly have

$$\mathcal{M}((M_1 \times M_2) \subseteq^{\sigma_1 \cup \sigma_2} (G_{C_1} \cap G_{C_2})) = \mathcal{M}(M_1 \subseteq^{\sigma_1} G_{C_1}) \cdot \mathcal{M}(M_2 \subseteq^{\sigma_2} G_{C_2})$$

3.3 Probabilistic Refinement

As we want the probabilistic refinement relation to be compatible with composition and satisfaction (and with the non-probabilistic relation), there is not much liberty in the way we can define it. Let's say that a contract C_1 refines a contract C_2 (written $\mathcal{C}_1 \preceq \mathcal{C}_2$). In order to be compatible with the composition, the probabilistic ports of $\mathcal{C}_1$ must be a subset of those of $\mathcal{C}_2$, and the distribution on these ports must be the same for $\mathcal{C}_1$ and $\mathcal{C}_2$. Moreover we want this relation to be compatible with implementation, which means that if an implementation satisfies $\mathcal{C}_1$ with level α, it must satisfy $\mathcal{C}_2$ with a level β that can be computed

from α. The idea for this is to measure the inclusion of the guarantees of $\mathcal{C}_1$ in the guarantees of $\mathcal{C}_2$, and to use this measure in order to compute β.

Definition 10 *If $\mathcal{C}_1 = (C_1, \mathbf{p}_1, \mathbb{P}_1)$ and $\mathcal{C}_2 = (C_2, \mathbf{p}_2, \mathbb{P}_2)$ are two probabilistic contracts, we say that $\mathcal{C}_1$ refines $\mathcal{C}_2$ with level γ ($\mathcal{C}_1 \preceq_\gamma \mathcal{C}_2$) if and only if*

1. *$\sigma_1 \subseteq \sigma_2$;*
2. *$\mathbf{p}_1 \subseteq \mathbf{p}_2$ and $\mathbb{P}_1$ is the marginal of $\mathbb{P}_2$ over $\mathbf{p}_1$;*
3. *$\mathbb{P}_2(\{\omega\} \subseteq^{\sigma_2} G_2 | \{\omega\} \subseteq^{\sigma_2} G_1) \geq \gamma$, whatever non-deterministic choices are.*

γ is a measure of the inclusion of G_1 in G_2, $\gamma = 1$ meaning that $G_1 \subseteq G_2$ almost all the time.

As the definition for the refinement relation was built to be compatible with implementation, it is quite logical to prove the following theorem, which says that if an implementation satisfies a probabilistic contract with level β, and if this contracts refines a second one with level γ, then the implementation satisfies the second contract with level $\beta \cdot \gamma$. This should enable us to use simpler contracts in order to prove satisfaction.

Theorem 2 *If $\mathcal{C}_1 = (C_1, \mathbf{p}_1, \mathbb{P}_1)$ and $\mathcal{C}_2 = (C_2, \mathbf{p}_2, \mathbb{P}_2)$ are 2 probabilistic contracts and M an implementation, then*

$$M \models_\beta \mathcal{C}_1 \text{ and } \mathcal{C}_1 \preceq_\gamma \mathcal{C}_2 \Rightarrow M \models_{\beta \cdot \gamma} \mathcal{C}_2.$$

Proof :
Because of the definition of the probabilistic refinement, this result is quite clear:

$$\mathbb{P}_2(M \models \mathcal{C}_2) = \mathbb{P}_2(\{w \mid \{w\} \cap M \subseteq^{\sigma_2} G_2\})$$

And

$$\mathbb{P}_2(\{w \mid \{w\} \cap M \subseteq^{\sigma_2} G_2\}) \geq \\ \mathbb{P}_2(\{w \mid \{w\} \cap M \subseteq^{\sigma_2} G_1\}) \cdot \mathbb{P}_2(\{w\} \subseteq^{\sigma_2} G_2 \setminus \{w\} \subseteq^{\sigma_2} G_1)$$

And as $\mathbb{P}_1$ is the marginal of $\mathbb{P}_2$ over $\mathbf{p}_1$,

$$\mathbb{P}_2(M \models \mathcal{C}_2) \geq \beta \cdot \gamma$$

□

Once again, we cannot find a finer bound because if $G_2 \subseteq^{\sigma_2} G_1$, we have $\mathbb{P}_2(M \models \mathcal{C}_2) = \beta \cdot \gamma$.

3.4 Problems with finer satisfaction relations

The satisfaction relation we chose above is adapted to reliability analysis. It measures the runs of the implementations that have the right behaviour *whatever non-deterministic choices are*. Consequently one could wonder whether it would be of interest to try finer satisfaction relations, for example existential or even finer, checking every state the system goes through.

These finer relations have been studied and ruled out of our work because they cannot be compositional for the following reasons:

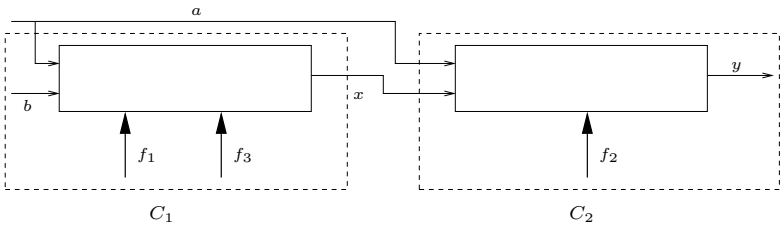


Figure 2: Global System

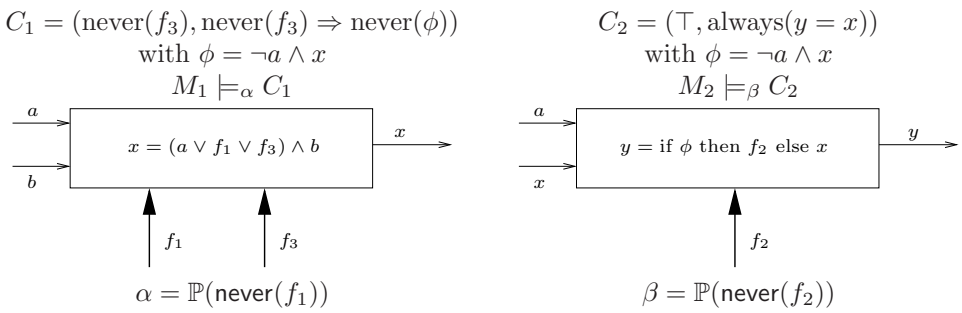
- An existential relation would allow us to say that "in 90% of the runs, there exist a way in which the environment can force our implementation to stay within the bounds of the guarantees". This would be contrary to the principle of re-use, where we want to be sure that whatever the user asks of the component, it behaves safely.
- An even finer satisfaction relation that would check every state the system goes through would be quite convenient in order to express disponibility properties. This kind of satisfaction relation would ponderate each infinite history of the probabilistic ports with the "amount" of visited states that ensure the guarantees. This would mean, for example, that for a particular history on the probabilistic ports, the induced behavior will stay within the guarantees with a probability at least α, whatever non-deterministic choices are. But knowing this probability is not enough to ensure compositionality, because if we compose two contracts, the induced behaviour for a fixed history of the probabilistic ports must satisfy both contracts *at the same time*.

3.5 Example

Let's assume we want to build a system with 2 input ports a and b, 1 output port y. We want this system to avoid a state where y is true and a is false. We know there are possibly different sources of failure f_1, f_2 and f_3 but we suppose for the moment that f_3 will never happen. We decide to split the system into 2 subsystems (Fig. 2). We will ask a first supplier to build the first subsystem as a component satisfying a contract C_1, and to a second supplier, we give a contract C_2.

The first supplier will then provide us with an implementation M_1 satisfying C_1 with a level α (Fig. 3(a)), and the second will give us an implementation M_2 satisfying C_2 with a level β (Fig. 3(b)). Once we have these components, we know that their composition will satisfy the contract $C = C_1 \parallel C_2 = (\text{never}(f_3), \text{never}(f_3) \Rightarrow \text{never}(\neg a \wedge y))$, with a level $\alpha \cdot \beta$ (Fig. 4).

Now consider the case when we discover that f_3 may in fact happen. The contract C is not realistic anymore, as it supposes that f_3 never happens. In consequence, we want to know how our components will satisfy a contract $C' = (\top, \text{never}(\neg a \wedge y))$. Instead of trying to find a new decomposition into different subcontracts, we just have to compute the level of refinement γ such that $C \preceq_\gamma C'$. We will then know that $M_1 \parallel M_2 \models_{\alpha\beta\gamma} C'$. This probability γ may be written as follows:



(a) M_1 (b) M_2

Figure 3: Subcomponents

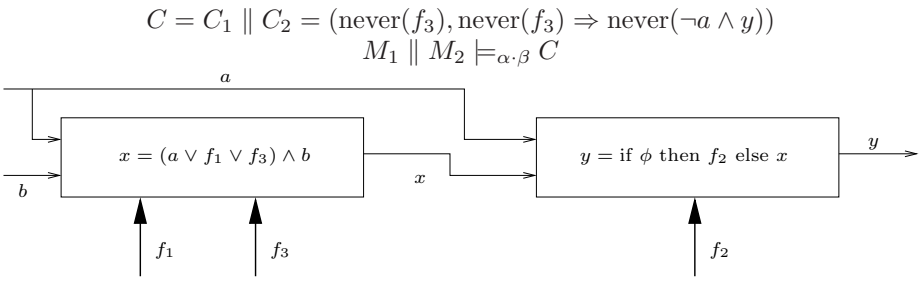


Figure 4: Composition of the implementations

$$\gamma = \mathbb{P}(\text{never}(\neg a \wedge y) | \text{never}(f_3) \Rightarrow \text{never}(\neg a \wedge y))$$

4 Related work

The problem of reliability analysis is widely present in the literature. Several attempts have been presented in the domain of probabilistic model checking in order to express probabilistic properties and check whether a particular system satisfies them. pCTL and pCTL*, for example, can be used to specify properties such as reliability and performance [HJ89, ASBSV95]. There even exist extensions of pCTL and pCTL* in which the probabilistic behaviour coexist with non-determinism [BdA95]. However, in these formalisms, the probabilistic point of view is inherent to the system checked. Consequently, compositionality is not an issue for them. Our formalism, on the contrary, considers probabilities as an assumption on the environment. In this way, we only consider open systems for which probabilities and non-determinism comes from the environment. In this way, compositionality can be proved and used in order to obtain a modular analysis.

On the other hand, compositional reliability analysis tools and formalisms have already been developed in the literature, such as Arcade [BCH+08] or Dynamic Fault Trees [BCS07] for example. These formalisms present compositional reliability analysis as it is actually done in the industry, that is to say without any behavioural interpretation. Our approach is different. We want to be able to reason on the behaviours of components, and not only on their failure probability. Of course our formalism captures such classical analysis, but it allows much more because the satisfaction relation is strongly linked to the behaviour of the implementations. Moreover, as our formalism allows assumptions on the environment, it can capture situations where two separate implementations do not satisfy their respective contracts, but their composition satisfies the composition of the contracts because of the assumptions on the environment, which would not be possible with a classical reliability analysis.

Finally, the probabilistic refinement relation we have built does not have an equivalent in the classical reliability analysis. It allows to compute the probabilistic satisfaction of a contract while only considering information on the probabilistic satisfaction of another contract and on the relations between these contracts.

5 Conclusion and further work

In this paper, we have presented a compositional theory of probabilistic contracts, capable of capturing reliability aspects of components and systems. This theory enables a behavioural interpretation of reliability, which was not the case in the existing compositional formalisms.

There are several natural directions to continue this work.

First, what we present here is a very general theory with few direct applications. Computing the satisfaction and refinement probabilities efficiently would require to narrow the field of applications. In practice, assertions and machines will be deterministic open transition systems and never sets of run. We are

actually developing a more practical approach where contracts are Markov Decision Processes and implementations open transition systems. In this approach, computing the satisfaction and refinement probabilities relies on the existence of pure optimal strategies in mean-payoff Markov Decision Processes [Gim07].

Finally, the same kind of probabilistic point of view could be adapted to contracts residuation [Rac08], which would give a practical way to build (canonical?) implementations from the residuation of the guarantees of a contract by its assumptions.

References

[ASBSV95] A. Aziz, V. Singhal, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, *It usually works: The temporal logic of stochastic systems*, 7th International Conference On Computer Aided Verification (Liege, Belgium) (P. Wolper, ed.), Lecture Notes in Computer Science, vol. 939, Springer Verlag, July 1995, pp. 155–165.

[BCH+08] Hichem Boudali, Pepijn Crouzen, Boudewijn R. Haverkort, Matthias Kuntz, and Marille Stoelinga, *Arcade - a formal, extensible, model-based dependability evaluation framework*, vol. 0, IEEE Computer Society, 2008, pp. 243–248.

[BCP07] A. Benveniste, B. Caillaud, and R. Passerone, *A generic model of contracts for embedded systems*, Research report 6214, INRIA Rennes, jun 2007.

[BCS07] Hichem Boudali, Pepijn Crouzen, and Mariëlle Stoelinga, *A compositional semantics for dynamic fault trees in terms of interactive markov chains*, ATVA (Kedar S. Namjoshi, Tomohiro Yoneda, Teruo Higashino, and Yoshio Okamura, eds.), Lecture Notes in Computer Science, vol. 4762, Springer, 2007, pp. 441–456.

[BdA95] Bianco and de Alfaro, *Model checking of probabilistic and nondeterministic systems*, FSTTCS: Foundations of Software Technology and Theoretical Computer Science **15** (1995).

[Gim07] Hugo Gimbert, *Pure stationary optimal strategies in markov decision processes*, STACS (Wolfgang Thomas and Pascal Weil, eds.), Lecture Notes in Computer Science, vol. 4393, Springer, 2007, pp. 200–211.

[HJ89] H. Hansson and B. Jonsson, *A framework for reasoning about time and reliability*, Proc. IEEE Real-Time Systems Symp., 1989, pp. 102–111.

[HR94] A. Høyland and M. Rausand, *System reliability theory: Models and statistical methods*, J. Wiley & Sons, New York, 1994.

[Rac08] Jean-Baptiste Raclet, *Residual for component specifications*, Electr. Notes Theor. Comput. Sci. **215** (2008), 93–110.

[SA96] R.M. Sinnamon and J.D. Andrews, *Fault tree analysis and binary decision diagrams*, Jan 1996, pp. 215–222.

Contents

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399