

# Finite Complete Suites for CSP Refinement Testing

Ana Cavalcanti<sup>1</sup>, Wen-ling Huang<sup>2</sup>, Jan Peleska<sup>2</sup>, and Adenilso Simao<sup>3</sup>

<sup>1</sup> University of York, United Kingdom

`ana.cavalcanti@york.ac.uk`

<sup>2</sup> University of Bremen, Germany

`{peleska,huang}@uni-bremen.de`

<sup>3</sup> University of São Paulo, Brazil

`adenilso@icmc.usp.br`

**Abstract.** In this paper, new contributions to testing Communicating Sequential Processes (CSP) are presented, with focus on the generation of complete, finite test suites. A test suite is complete if it can uncover every conformance violation of the system under test with respect to a reference model. Both reference models and implementation behaviours are represented as CSP processes. As conformance relation, we consider trace equivalence and trace refinement, as well as failures equivalence and failures refinement. Complete black-box test suites here rely on the fact that the SUT's true behaviour is represented by a member of a fault-domain, that is, a collection of CSP processes that may or may not conform to the reference model. We define fault domains by bounding the number of excessive states occurring in a fault domain member's representation as a normalised transition graph, when comparing it to the number of states present in the graph of the reference model. This notion of fault domains is quite close to the way they are defined for finite state machines, and these fault domains guarantee the existence of *finite* complete test suites.

**Keywords:** Model-based testing, CSP, Trace Refinement, Failures Refinement, Complete Test Suites

## 1 Introduction

**Motivation** Model-based testing (MBT) is an active research field that is currently evaluated and integrated into industrial verification processes by many companies. This holds particularly for the embedded and cyber-physical systems domain. While MBT is applied in different flavours, we consider the most effective variant to be the one where test cases and concrete test data, as well as checkers for the expected results (*test oracles*), are automatically generated from a reference model: it guarantees the maximal return of investment for the time and effort invested into creating the test model. The test suites generated in this

way, however, usually have different test strength, depending on the generation algorithms applied.

For the safety-critical domain, test suites with guaranteed fault coverage are of particular interest. For black-box testing, guarantees can be given only if certain hypotheses are satisfied. These hypotheses are usually specified by a *fault domain*: a set of models that may or may not conform to the SUT. The so-called *complete* test strategies guarantee to uncover every conformance violation of the SUT with respect to a reference model, provided that the true SUT behaviour is captured by a member of the fault domain.

Generation methods for complete test suites have been developed for various modelling formalisms. In this paper, we use *Communicating Sequential Processes (CSP)* [4, 8]; this is a mature process-algebraic approach that has been shown to be well-suited for the description of reactive control systems in many publications over almost five decades. Industrial success has also been reported.

**Contributions** This paper complements work published by two of the authors in [1]. There, fault domains are specified as collections of processes refining a “most general” fault domain member. With this concept of fault domains, complete test suites may be finite or infinite. While this gives important insight into the theory of complete test suites, we are particularly interested in finite suites when it comes to their practical application.

Therefore, we present a complementary approach to the definition of CSP fault domains in this paper. To this end, we observe that every finite-state CSP process can be semantically represented as a finite normalised transition graph, whose edges are labelled by the events the process engages in, and whose nodes are labelled by minimal acceptances or, alternatively, maximal refusals [7]. The maximal refusals express the degree of nondeterminism that is present in a given CSP process state that is in one-one-correspondence to a node of the normalised transition graph. Inspired by the way that fault-domains are specified for finite state machines (FSMs), we define them here as the set of CSP processes whose normalised transition graphs do not exceed the size of the reference model’s graph by more than a give constant.

Our main contributions in this paper are as follows.

1. It is proven that for fault domains of the described type, complete test suite generation methods can be given for verifying (1) trace equivalence, (2) trace refinement, (3) failures equivalence, and (4) failures refinement.
2. We prove that finite complete test suites can be generated in all four cases, when using the fault domains based on the size of the members’ normalised transition graphs.
3. We present test suite generation techniques for each of the four conformance relations by translating algorithms originally elaborated for the FSM domain into the CSP world. This translation preserves the completeness properties that have previously been established for the FSM domain by other authors.

The possibility to translate complete test suites between different formalisms (here FSMs and CSP processes) has been investigated before by two of the authors [5].

**Overview** @todo

## 2 Preliminaries

In this section, we present the background material relevant to our work.

### 2.1 Complete Test Suites

We use the term *signature* to denote a collection of comparable models represented in an arbitrary formalism. In this article, signatures represent sets of finite state machines over fixed input and output alphabets, or CSP processes with finite state, represented by their normalised transition graphs (see Section 2.2).

Given a signature  $Sig$  of models, a *fault model*  $\mathcal{F} = (M, \leq, Dom)$  specifies a *reference model*  $M \in Sig$ , a *conformance relation*  $\leq \subseteq Sig \times Sig$  between models, and a *fault domain*  $Dom \subseteq Sig$ . This terminology follows [6], where fault models have been originally introduced in the context of testing of finite state machines. Note that fault domains may contain both models conforming to the reference model and models violating the conformance relation. Note further that the reference model  $M$  is not necessarily a member of the fault domain, although a model of the SUT behaviour is. For example,  $M$  could be nondeterministic, while only deterministic implementation behaviours might be considered in the fault domain.

Let  $TC(Sig)$  denote the set of all *test cases* applicable to elements of  $Sig$ . This abstract notion of test case requires only the existence of a relation  $\underline{\text{pass}} \subseteq Sig \times TC(Sig)$ . For  $(M, U) \in \underline{\text{pass}}$ , the infix notation  $M \underline{\text{pass}} U$  is used, and interpreted as ‘*model  $M$  passes the test case  $U$* ’. If  $(M, U) \notin \underline{\text{pass}}$  holds, this is abbreviated by  $M \underline{\text{fail}} U$ . Our specific notion of test cases for CSP models is elaborated in Section ??.

A *test suite*  $TS \subseteq TC(Sig)$  denotes a set of test cases. A model  $M$  *passes the test suite*  $TS$ , also written as  $M \underline{\text{pass}} TS$ , if, and only if,  $M \underline{\text{pass}} U$  for all  $U \in TS$ . A test suite  $TS$  is called *complete* for fault model  $\mathcal{F} = (M, \leq, Dom)$ , if, and only if, the following properties hold.

1. If a member  $M'$  of the fault domain conforms to the reference model  $M$ , it passes the test suite, that is,

$$\forall M' \in Dom : M' \leq M \Rightarrow M' \underline{\text{pass}} TS$$

This property is usually called *soundness* of the test suite.

FiXme Warning: alcc:  
I miss a paragraph  
that discusses what is  
difficult. So far, it  
sounds like just  
application of known  
results, which is not  
true.

2. If a member of the fault domain passes the test suite, it conforms to the reference model, that is,

$$\forall M' \in Dom : M' \text{ pass TS} \Rightarrow M' \leq M$$

This property is usually called *exhaustiveness*.

A test suite TS is *finite* if it contains finitely many test cases and every test case  $U \in \text{TS}$  is finite in the sense that it terminates after a finite number of steps. It is trivial to see that, if TS is complete for  $\mathcal{F} = (M, \leq, Dom)$  and  $Dom' \subseteq Dom$ , then TS is also complete for  $\mathcal{F}' = (M, \leq, Dom')$ .

## 2.2 CSP and Refinement

### Communicating Sequential Processes @todo

**Normalised Transition Graphs for CSP Processes** As shown in [7], any finite-state CSP process  $P$  can be represented by a *normalised transition graph*

$$G(P) = (N, \underline{n}, \Sigma, t : N \times \Sigma \rightarrow N, r : N \rightarrow \mathbb{P}(\Sigma)),$$

with nodes  $N$ , initial node  $\underline{n} \in N$ , and process alphabet  $\Sigma$ . The partial *transition function*  $t$  maps a node  $n$  and an event  $e \in \Sigma$  to its successor node  $t(n, e)$ , if, and only if,  $(n, e)$  is in the domain of  $t$ , that is, there is a transition from  $n$  with label  $e$ . Normalisation of  $G(P)$  is reflected by the fact that  $t$  is a function.

A finite sequence of events  $s \in \Sigma^*$  is a *trace* of  $P$ , if there is a path through  $G(P)$  starting at  $\underline{n}$  whose edge labels coincide with  $s$ . The set of traces of  $P$  is denoted by  $\text{traces}(P)$ . If  $s \in \text{traces}(P)$ , then the process corresponding to  $P$  after having executed  $s$  is denoted by  $P/s$ . Since  $G(P)$  is normalised, there is a unique node reached by applying the events from  $s$  one by one, starting in  $\underline{n}$ . Therefore,  $G(P)/s$  is also well defined.

By  $[n]^0$  we denote the *fan-out* of  $n$ : the set of events occurring as labels in any outgoing transitions.

$$[n]^0 = \{e \in \Sigma \mid (n, e) \in \text{dom } t\}$$

We also use this notation for CSP processes:  $[P]^0$  is the set of events  $P$  may engage into, in other words, the initials of  $P$  after the empty trace of events, that is,  $\text{initials}(P/\langle \rangle)$  as defined in [8].

The total function  $r$  maps each node  $n$  to its *refusals*  $r(n) = \text{Ref}(n)$ . Each element of  $r(n)$  is a set of events that the CSP process  $P$  might refuse to engage into, when in a process state corresponding to  $n$ . The number of refusal sets in  $\text{Ref}(P/s)$  specifies the degree of nondeterminism that is present in process state  $P/s$ : the more refusal sets contained in  $\text{Ref}(P/s)$ , the more nondeterministic is the behaviour in state  $P/s$ . If  $P/s$  is deterministic, its refusals coincide with the set of subsets of  $\Sigma - [P/s]^0$ , including the empty set.

FiXme Warning: alcc:  
I can make this small  
contribution.

For finite CSP processes, since the refusals of each process state are subset-closed [4, 8],  $\text{Ref}(P/s)$  can be re-constructed by knowing the set of *maximal refusals*  $\text{maxRef}(P/s) \subseteq \text{Ref}(P/s)$ . More formally, the maximal refusals  $\text{maxRef}(P/s)$  are defined as

$$\text{maxRef}(P/s) = \{R \in \text{Ref}(P/s) \mid \forall R' \in \text{Ref}(P/s) - \{R\} : R \not\subseteq R'\}$$

Conversely, with the maximal refusals at hand, we can reconstruct the refusals by subset-closure:

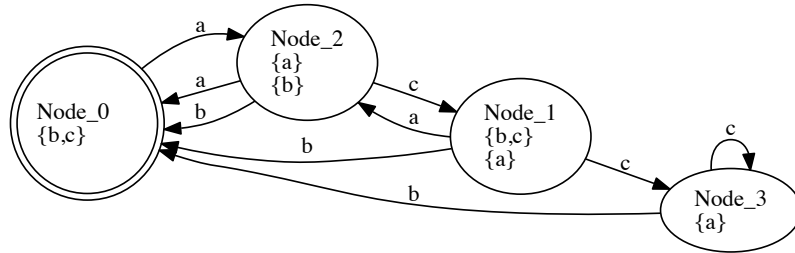
$$\text{Ref}(P/s) = \{R' \in \mathbb{P}(\Sigma) \mid \exists R \in \text{maxRef}(P/s) : R' \subseteq R\}.$$

To see that this approach works only for finite CSP processes, consider the example where  $\Sigma$  is infinite. In this case,  $\text{maxRef}(STOP/\langle \rangle)$  is empty, and so we cannot use this set to calculate the refusals of  $STOP$ , that is,  $\text{Ref}(STOP/\langle \rangle)$  as defined above. As with refusals, we also use the transition graph-oriented notation  $\text{maxRef}(n) \subseteq r(n)$  to denote the maximal refusals associated with graph state  $n$ : if  $n$  is the state reached in the transition graph by following the edge labels in trace  $s$ , then  $\text{maxRef}(n) = \text{maxRef}(P/s)$ .

Well-formed normalised transition graphs must not refuse an event of the fan-out of a state in *every* refusal applicable in this state; more formally,

$$\forall n \in N, e \in \Sigma : (n, e) \in \text{dom } t \Rightarrow \exists R \in \text{maxRef}(n) : e \notin R \quad (1)$$

By construction, normalised transition graphs reflect the *failures semantics* of finite-state CSP processes: the traces  $s$  of a process are the sequences of transition associated with paths through the graph, starting at  $\underline{n}$ . The pairs  $(s, R)$  with  $s \in \text{traces}(P)$  and  $R \in r(G(P)/s)$  represent the failures  $\text{failures}(P)$  of  $P$ .



**Fig. 1.** Normalised transition graph of CSP process  $P$  from Example ??.

*Example 1.* ex:a Consider the CSP process  $P$  defined below, and that the set of events  $\Sigma$  is  $\{a, b, c\}$ .

$$\begin{aligned} P &= a \rightarrow (Q \sqcap R) \\ Q &= a \rightarrow P \sqcap c \rightarrow P \\ R &= b \rightarrow P \sqcap c \rightarrow R \end{aligned}$$

Its transition graph  $G(P)$  is shown in Fig. 1. Process state  $P/\langle \rangle$  is represented there as Node\_0, with  $\{b, c\}$  as the only maximal refusal, since  $a$  can never be refused, and no other events are accepted. Having engaged into  $a$ , the transition emanating from Node\_0 leads to Node\_2 representing the process state  $P/a = Q \sqcap R$ . The internal choice operator induces several refusal sets derived from  $Q$  and  $R$ . Since these processes accept their initial events in external choice,  $Q \sqcap R$  induces two maximal refusal sets  $\{b\}$  and  $\{a\}$ . Note that event  $c$  can never be refused, since it is not a member of any maximal refusal.

Having engaged into  $c$ , the next process state is represented by Node\_1. Due to normalisation, there is only a single transition satisfying  $t(\text{Node\_2}, c) = \text{Node\_1}$ . This transition, however, can have been caused by either  $Q$  or  $R$  engaging into  $c$ , so Node\_1 corresponds to process state  $Q/c \sqcap R/c = P \sqcap R$ . This is reflected by the two maximal refusals labelling Node\_1.

Similar considerations explain the other nodes and transitions in Fig. 1.

Note that the node names are generated by the FDR tool (see next paragraph). The node numbering is generated by FDR during the normalisation procedure. Therefore, the node numbers do not reflect the distance from the initial node Node\_0.

**Tool Considerations** The FDR tool [3] supports model checking and semantic analyses of CSP processes. It provides an API [2] that can be used to construct normalised transition graphs for CSP processes.

The FDR graph nodes are labelled by *minimal acceptances* instead of maximal refusals as described above. Since such a minimal acceptance set is the complement of a maximal refusal, the function  $r$  mapping states to their refusals can be implemented by creating the complements of all minimal acceptances and then building all subsets of these complements. For practical applications, the subset closure is never constructed in an explicit way; instead, sets are checked with respect to containment in a maximal refusal.

### 3 Finite Complete Test Suites for CSP Trace Refinement

### 4 Finite Complete Test Suites for CSP Failures Refinement

### 5 Applications

#### 5.1 Testing for Trace Equivalence

#### 5.2 Testing for Trace Refinement

#### 5.3 Testing for Failures Equivalence

#### 5.4 Testing for Failures Refinement

### 6 Related Work

### 7 Conclusion

### References

1. Cavalcanti, A., da Silva Simão, A.: Fault-based testing for refinement in CSP. In: Yevtushenko, N., Cavalli, A.R., Yenigün, H. (eds.) Testing Software and Systems - 29th IFIP WG 6.1 International Conference, ICTSS 2017, St. Petersburg, Russia, October 9-11, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10533, pp. 21–37. Springer (2017), [https://doi.org/10.1007/978-3-319-67549-7\\_2](https://doi.org/10.1007/978-3-319-67549-7_2)
2. Gibson-Robinson, T., Armstrong, P., Boulgakov, A., Roscoe, A.: Failures Divergences Refinement (FDR) Version 3 (2013), <https://www.cs.ox.ac.uk/projects/fdr/>
3. Gibson-Robinson, T., Armstrong, P., Boulgakov, A., Roscoe, A.: FDR3 — A Modern Refinement Checker for CSP. In: brahm, E., Havelund, K. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. Lecture Notes in Computer Science, vol. 8413, pp. 187–201 (2014)
4. Hoare, C.A.R.: Communicating Sequential Processes. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1985)
5. Huang, W.L., Peleska, J.: Complete model-based equivalence class testing for non-deterministic systems. Formal Aspects of Computing 29(2), 335–364 (2017), <http://dx.doi.org/10.1007/s00165-016-0402-2>
6. Petrenko, A., Yevtushenko, N., Bochmann, G.v.: Fault models for testing in context. In: Gotzhein, R., Bredereke, J. (eds.) Formal Description Techniques IX – Theory, application and tools, pp. 163–177. Chapman&Hall (1996)
7. Roscoe, A.W. (ed.): A Classical Mind: Essays in Honour of C. A. R. Hoare. Prentice Hall International (UK) Ltd., Hertfordshire, UK, UK (1994)
8. Roscoe, A.W.: Understanding Concurrent Systems. Springer, London, Dordrecht Heidelberg, New York (2010)