Towards stochastic inference driven SOA testing

Bayesian Networks for Services Architecture Testing (BN4SAT)

Ariele P. Maesano (Ariele.Maesano@lip6.fr)
Fabio De Rosa (fabio.de-rosa@simple-eng.com)

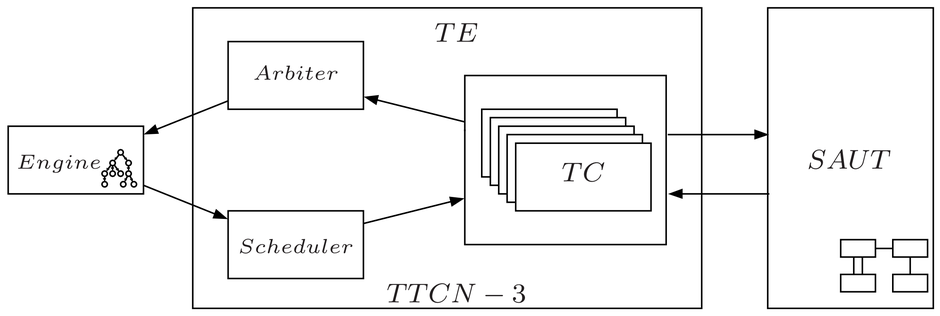
Service Oriented Architecture

SOA is a design and implementation style that allows multiple organizations to put into operation distributed architectures of loosely coupled systems in order to achieve flexible business automation. In the SOA approach, systems implementations are black boxes, hidden and private to each participant organization. The only available verification methods are black-box testing of individual participants and grey-box testing of interactions among participants. SOA testing is a means for failure discovering and troubleshooting of services architectures.

Problem

The same SOA key issues (reduced control, reduced observability, reduced trust) that make black-box/grey-box testing the only practicable verification method make also it a heavy and complex task. From the standpoint of SOA testing, a Services Architecture Under Test (SAUT) is a collection of SUTs (Systems under test) connected by channels conveying communicative actions (message sending, remote procedure call, rpc reply). Some or all of these channels are observable. The only information available to the Tester is the match / mismatch between actual communicative actions and expected ones.

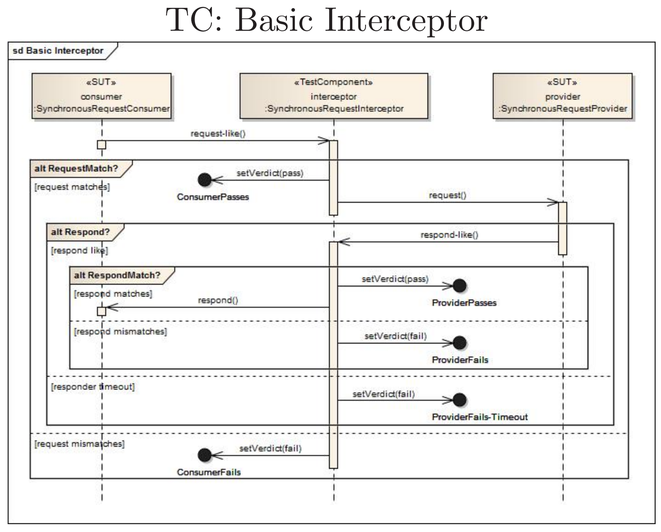
Testing Solution



BN4SAT Engine cycle: (i) the Engine invokes the Scheduler with the indication of the test case to run; (ii) the Scheduler prepares and launches the test on the TCs; (iii) the TCs run the test and return the local test verdicts to the Arbiter; (iv) the Arbiter sets global verdicts and assigns them probability distributions such as {(pass,P),(fail,1-P)} that return as evidences to the Engine; (v) the Engine puts the evidences in the appropriate variables, triggering the BN inference process; (vi) the result of the inference process is the choice of the next test case to run.

The Arbiter, the Scheduler and the Test Components run on a TTCN-3-compliant test automation platform.

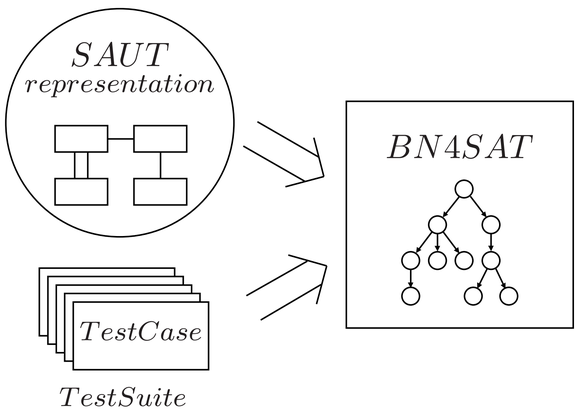
TC: Basic Interceptor



Bayesian network inference

A Bayesian network is a probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG). It can be used to find out updated knowledge of the state of a subset of variables when other variables (the evidence variables) are observed. This process of computing the posterior distribution of variables given evidence is called probabilistic inference. For example, a Bayesian network could represent the probabilistic relationships between diseases and symptoms: given symptoms, the network can be used to compute the probabilities of the presence of various diseases.

Building the BN

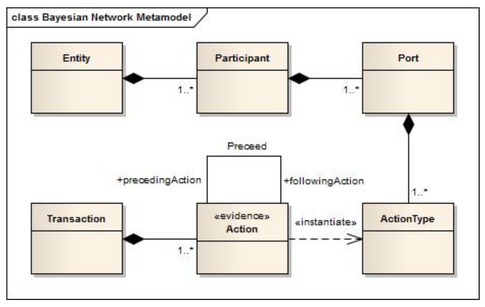


SAUT + Test Suite

SAUT representation =
UDDI V3 Data Structure +
collection of WSDL's

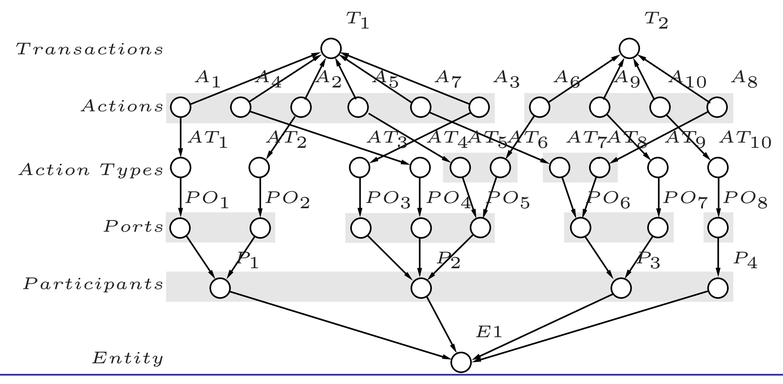
Test Suite representation =
collection of
(UML Interaction (XMI) +
collection of SOAP messages)

BN4SAT variables metamodel



Entity: business organization {notFaulty,faulty}
Participant: architecture participant {notFaulty,faulty}
Port: participant required interface {notFaulty,faulty}
ActionType: type of communicative action {notFaulty,faulty}
***Action* (evidence): test case action {pass,fail}**
Transaction: test case {pass,fail}

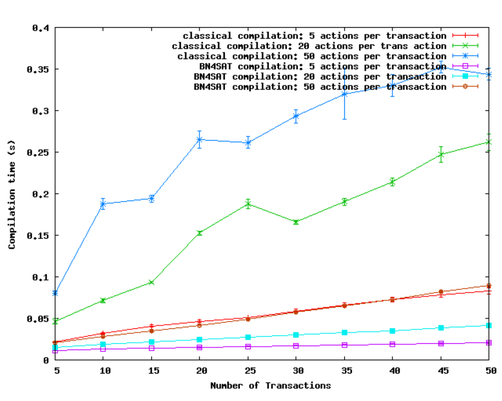
BN4SAT example

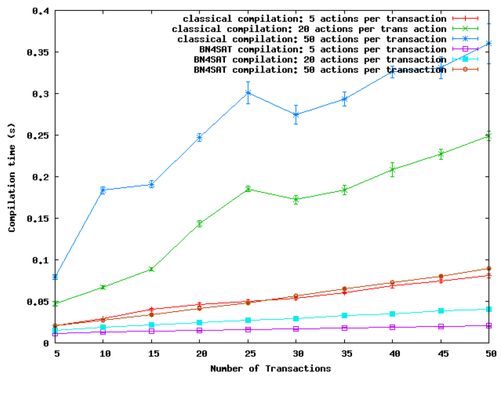


Problem

A BN for SOA testing can exhibit a very large size, and can be very dense, therefore with large conditional probability tables. The classical BN inference methods, such as lazy propagation, variable elimination, Shafer-Shenoy are inadequate in terms of response time and ability to process large amounts of data.

Results





Solution: inference by compilation + model driven encoding

Inference by compilation is based on the idea that each Bayesian network can be interpreted as a multi-linear function(MLF) and therefore implemented by an arithmetic circuit (AC). MLFs distinguish between two kinds of proposition variables: (i) evidence indicators - variables that can be observed (the Action variables in the BN4SAT model); (ii) network parameters - variables that cannot be observed (their probability distribution is calculated through the conditional probability tables of the Bayesian network). There exist various techniques able to transform BNs into ACs. Darwiche and colleagues report a method for the encoding of the BN in Conjunctive Normal Form (CNF), followed by the compilation of the CNF into the deterministic disjunction negation normal form (d-DNNF), that can be easily transformed into an AC. Traditional CNF generation methods are multi-step: in the first step a coarse version of the CNF is built and then is optimized to a more concise version in a second step. Our method utilizes the BN4SAT model in order to infer the topology and content of the "virtual" BN but instead generates an optimized CNF directly from the SAUT and the Test Suite representation.