Diagnosis of Discrete-Event Systems Benchmarks

Alban Grastien **Anbulagan**
KRR/NICTA and ANU **LC/NICTA and ANU**
Canberra – Australia
{**alban.grastien , anbulagan**}**@nicta.com.au**

1 Diagnosis of Discrete Event Systems

Diagnosis is to find out whether a system has a good or a bad behaviour, based on *observations* of this behaviour. The difficulty comes from the fact that part of the behaviour is not directly observed and cannot be completly reconstructed from the observations.

We consider discrete-event systems [Cassandras and Lafortune, 1999], i.e. systems whose states can be represented by the assignment of a finite set of variables in finite domains. The system is a set of interconnected components. The model of each component is denoted $\langle V_i, D_i, E_i, R_i, I_i\rangle$ where V_i is a set of variables, $\forall v \in V_i,\ D_i(v)$ is the finite domain of the variable v, E_i is a set of events, R_i is a set of transition rules and I_i is an assignment of the variables V_i.

A state of the component is an assignment of the variables. A rule is a tuple $\langle pre, \Sigma, \mathit{eff}\rangle$. A rule is enabled in a state s if $s \models pre$. The triggerring of the rule generates the events $\Sigma \subseteq E_i$ and leads to the effects eff (modification of the assignment of some variables).

A binary synchronisation relation S is defined on the system: $(e, e') \in S$ means that the occurrence of the event e on one component leads to the immediate occurrence of the event e' on another component. This means that if a rule $\langle pre_i, \Sigma_i, \mathit{eff}_i\rangle$ is triggerred on a component such that $\exists e, e',\ e \in \Sigma_i \wedge (e, e') \in S$, then a rule $\langle pre_j, \Sigma_j, \mathit{eff}_j\rangle$ must be triggerred at the same moment on another component such that $e' \in \Sigma_j$.

A trajectory is a sequence of states and sets of events such that the ith set of events is enabled in the ith state and leads to the $(i+1)$th state. The trajectories are partitionned into set of different fault levels.

Observations are provided by the system. It is modelled as a Boolean function on the trajectories that is *true* if the trajectory is consistent with the observations.

The goal of the diagnosis is to determine what is the smallest fault level such that there exists a trajectory consistent with the observations and the fault level.

2 Translation into a SAT problem

We construct a formula such that satisfiable valuations of this formula correspond to sequences $(s_0, \ldots, s_n)$ of states and sequences $(E_0, \ldots, E_{n-1})$ of events consistent with the observations.

We first translate the component model $\langle V_i, D_i, E_i, R_i, I_i\rangle$ into a CNF denoted CNF_i. The propositional variables, with superscript t corresponding to time step t, are the following:

- $(v = \nu)^t$ for all $v \in V_i$, $\nu \in D_i(v)$, and $t \in \{0, \ldots, n\}$,
- e^t for all $e \in E_i$ and $t \in \{0, \ldots, n-1\}$, and
- r^t for all $r \in R_i$ and $t \in \{0, \ldots, n-1\}$.

At each time point $t \in \{0, \ldots, n\}$, a variable $v \in V_i$ has exactly one valuation (rules 1 and 2). A rule $r = \langle pre, \Sigma, \mathit{eff}\rangle$ is triggerred at time t only if the precondition is satisfied (rule 3). The effect $(v = \nu) \in \mathit{eff}$ of the triggering rule $r = \langle pre, \Sigma, \mathit{eff}\rangle$ at time t apply at time $t+1$ (rule 4). The valuation of state variable v is changed to ν only if a rule is triggered with effect $v = \nu$ (rule 5). When a rule $r = \langle pre, \Sigma, \mathit{eff}\rangle$ is triggerred at time t, the event $e \in \Sigma$ occurs (rule 6). An event e occurs at time t only if a rule that contains this event is triggered (rule 7). A unique transition can be triggered at time t on the component (rule 8). The initial state is set by rule 9. The conjunction of all these rules is CNF_i.

$$\neg(v = \nu)^t \vee \neg(v = \nu')^t \qquad \nu \neq \nu' \tag{1}$$

$$\bigvee_{\nu \in D_i(v)} (v = \nu)^t. \tag{2}$$

$$r^t \rightarrow pre^t. \tag{3}$$

$$r^t \rightarrow (v = \nu)^{t+1}. \tag{4}$$

$$(v = \nu)^t \vee \neg(v = \nu)^{t+1} \vee r_1^t \vee \cdots \vee r_k^t \tag{5}$$

where $r_1, \ldots, r_k$ are all the rules such that $(v = \nu) \in \mathit{eff}(r_j)$.

$$r^t \rightarrow e^t. \tag{6}$$

$$\neg e^t \vee r_1^t \vee \cdots \vee r_k^t \tag{7}$$

where $r_1, \ldots, r_k$ are all the rules that contain event e.

$$\neg r_1^t \vee r_2^t \qquad r_1 \neq r_2.^1 \tag{8}$$

$$\bigwedge_{v \in V} (v = I(v))^0 \tag{9}$$

[1] Actually, a more compact representation is used based on the event variables.

The synchronisation S is translated into a CNF denoted CNF_S. If $(e, e') \in S$, then e occurs at time step t iff e' occurs at time step t (rule 10).

$$e^t \leftrightarrow e'^t. \tag{10}$$

The translation of the observation into CNF_O is given in subsection 4.3, and the translation of the normal behaviour into CNF_N is given in subsection 4.2. The diagnosis query is then:

$$\Delta = CNF_1 \wedge \cdots \wedge CNF_p \wedge CNF_S \wedge CNF_O \wedge CNF_N$$

If the CNF Δ is satisfiable, then the diagnosis of the system is normal.

3 The system we diagnose

The system contains 20 components with the same behaviour in a 5×4 grid such that each component is connected with its 4 neighbours. The system is show on Figure 1.

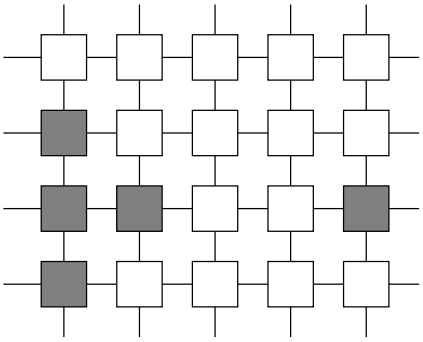

Figure 1: Topology of the system

For instance, the component in position $(0, 2)$ is connected to components in positions $(0, 1)$, $(0, 3)$, $(1, 2)$ and $(4, 2)$. All these components are represented in grey on Figure 1. A flat and simplified representation (i.e with fewer events) of the behaviour of each component is given Figure 2. There is only one state variable and its initial value is O. When a failure occurs on a component, its state is changed to F and the message *reboot!* is sent by the component to its neighbours that receive the message *reboot?*, leading to state W, FF or R depending on their current state.

Each component has 6 boolean state variables, 17 rule variables and 15 event variables per time step.

4 Parameters for the benchmark

We used several criteria to build the CNF files.

4.1 Difficulty of the scenario

The diagnosis problem is built from the generation of a random scenario (what did happen on the system). In this system, the scenario is much difficult to reconstruct if most of the components come back to state O as soon as possible. Based on this remark, we have built two different kinds of scenarios: medium (represented by parameter 5), and hard (10).

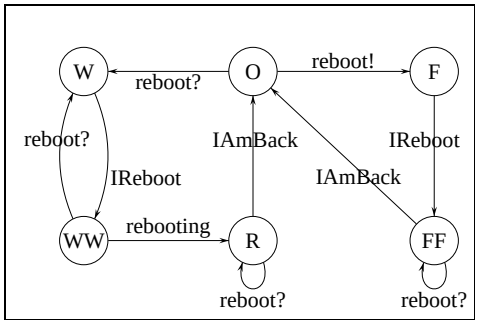

Figure 2: The behaviour of one component.

4.2 Number of faults

A trajectory is faulty if it contains a certain number of occurrences of the event *reboot!* on any component. CNF_{N_k} is satisfiable iff k faulty events occurred during the trajectory. The CNF is implemented as proposed in [Bailleux and Boufkhad, 2003].

We have built scenarios with different numbers of faulty events from 11 to 19 and used the formula CNF_{N_k} for each scenario which leads to a satisfiable CNF Δ. We have also built Δ with $CNF_{N_{k-1}}$ which should be unsatisfiable. However, since the scenario are randomly generated, it may happen that a scenario with $k-1$ faults can be found consistent with the observations; in this case, the CNF Δ is satisfiable.

4.3 Observability

The observations are extracted from the random scenario and CNF_O is built from these observations. They represent what was actually seen of the system behaviour. The observations can be either accurate or imprecise depending on the system. We considered three cases:

Dated observations OBS is a set of dated observable event occurrences. The observable event e occurred at time step t iff $\langle e, t\rangle \in OBS$. The representation of the observations consists of setting the e^t to *true* if this observable event e occurred at date t, and to *false* if this observable event e did not.

$$\bigwedge_{\langle e,t\rangle \in OBS} e^t \wedge \bigwedge_{\langle e,t\rangle \notin OBS} \neg\, e^t \tag{11}$$

Total order OBS is a set of observable event occurrences with a total order relation $\prec$. If $e \prec e'$, then e occurred before e'. The set of observations is translated in a set of dated observations where we consider that the ith observation occurred at date $i \times 2$, and then translated into a CNF as mentioned for dated observations.

Partial order The relation $\prec$ is partial. We denote $d(j)$ the date of occurrence of the jth observable event. A propositional variable $o^{d(j)}$ is created that indicates that the observable event associated with o occurred at date $d(j)$ and a variable $\hat{o}^{d(j)}$ that indicates that the observable event associated with o occurred *before* or at the date $d(j)$.

The formula Φ_{OBS} is the conjunction of the following clauses. An observable event occurred before $d(j)$ if it occurred before $d(j-1)$ or at $d(j)$: $\hat{o}^{d(j)} \leftrightarrow \hat{o}^{d(j-1)} \vee o^{d(j)}$. An observation is emitted only once: $\hat{o}^{d(j-1)} \rightarrow \neg o^{d(j)}$. All the observations were emitted: $\hat{o}^{d(p)}, \forall o \in OBS$. The partial ordering must be defined: $o_2^{d(j)} \rightarrow \hat{o}_1^{d(j-1)}, \forall o_1, o_2 : o_1 \prec o_2$. Finally, an observable event occurs if and only if an observation associated with this event was received: $e^{d(j)} \leftrightarrow o_{f1}^{d(j)} \vee \cdots \vee o_{fk}^{d(j)}$ where $o_{f1}, \ldots, o_{fk}$ represent the observations emitted by the event e.

4.4 Examples

Here are examples of files.

`total-5-19-u.cnf` corresponds to the unsatisfiable problem from a medium-difficulty scenario with 19 faults and totally ordered observations.

`dated-10-11-s.cnf` corresponds to the satisfiable problem from a hard scenario with 11 faults and dated observations.

`partial-10-15-s.cnf` corresponds to the satisfiable problem from a hard scenario with 15 faults and partially ordered observations.

References

[Bailleux and Boufkhad, 2003] O. Bailleux and Y. Boufkhad. Efficient CNF encoding of boolean cardinality constraints. In *Ninth International Conference on Principles and Practice of Constraint Programming (CP'03)*, pages 108–122. Springer-Verlag, 2003.

[Cassandras and Lafortune, 1999] C. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Kluwer Academic Publishers, 1999.