ECAI 2008
M. Ghallab et al. (Eds.)
IOS Press, 2008

doi:10.3233/978-1-58603-891-5-787

Incremental Diagnosis of DES by Satisfiability[1]

Alban Grastien and Anbulagan
NICTA and Australian National University

Abstract. We propose a SAT-based algorithm for incremental diagnosis of discrete-event systems. The monotonicity is ensured by a *prediction window* that uses the future observations to lead the current diagnosis. Experiments stress the impact of parameters tuning on the correctness and the efficiency of the approach.

1 Diagnosis by SAT

Diagnosis is the AI problem of determining whether a system is running correctly during a time window, and of identifying any failure otherwise. Consider a system which is completely modeled by a DES (basically a finite state machine) denoted Mod. This system is running and generates observations. The goal of the diagnosis is to determine from the model and the observations whether faulty events occurred on the system. The problem can be reduced to finding particular paths on the DES consistent with the observations [4]. Since failures are rare events, we can consider paths that minimize the number of faults.

In [2], we proposed to solve the DES diagnosis problem with satisfiability (SAT) algorithms. SAT is the problem of finding an assignment of the variables of a given Boolean formula in such a way as to make the formula evaluate to *true*. Given an upper bound on the number of transitions in the paths that are considered, a diagnosis problem – finding a particular path – can be encoded as a SAT problem. The SAT-based algorithm then simply uses SAT solver to look for a path with increasing number of faults until a diagnosis is found.

2 Incremental Diagnosis by SAT

Incremental diagnosis (ID) consists in computing the diagnosis for a temporal window, and then updating this diagnosis to consider a larger temporal window. The incremental diagnosis can serve for two purposes. First, it is used when the observations for the latter temporal window are not immediately available: a diagnosis for the first temporal window is computed, and then must be updated as the other observations are provided. This is typically the case for on-line diagnosis, where the system is monitored while it is running.

Second, an incremental approach can be used to simplify a non-ID problem. Given a diagnosis task on a large temporal window, the window is *sliced* into small windows to obtain simpler diagnosis problems. In both cases, the complexity of the ID must be independent of the previous diagnoses. This paper considers the second approach where all the observations are available. The on-line problem contains additional issues mostly independant from ID.

[1] This research was supported by NICTA in the framework of the SuperCom project. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

Consider that the observations on a window are denoted *Obs*, and denote $\oplus$ the concatenation of two windows. Note that the concatenation may be non-trivial in case of uncertain observations [3]. Given the diagnosis of Obs_1, given the observations Obs_2, the *incremental diagnosis* is the computation of the diagnosis of $Obs_1 \oplus Obs_2$. The complexity of the incremental diagnosis of $Obs_1 \oplus Obs_2$ given the diagnosis of Obs_1 must not depend on the size of Obs_1. It is wise to perform the diagnosis of Obs_1 in such a way as to ease the ID of $Obs_1 \oplus Obs_2$. In this case, the complexity of the diagnosis of Obs_1 must not depend on the size of Obs_2.

Rather than diagnosing the whole period (t_0, t_n), we do an ID and diagnose n windows of size λ ($t_{i+1} = t_i + \lambda$). The n diagnoses must be consistent with each other: the path computed for (t_1, t_2) must be a continuation of the path computed for (t_0, t_1). However, when the diagnosis of (t_0, t_1) is computed, we cannot be sure that the extracted path will be consistent with the next observations. In diagnosis, there is usually a delay between the occurrence of an event and the reception of observations proving this occurrence. However, this delay is generally bounded: it is unlikely an observation will explain what happened several days or weeks ago. Thus, we ensure that the path of (t_0, t_1) is consistent not only with observations (t_0, t_1) but also with the observations $(t_1, t_1 + \mu)$. This way, the diagnosis for this window should be globally consistent. The period of time $(t_1, t_1 + \mu)$ is called *prediction window* of the *diagnosis window* (t_0, t_1). Note that the diagnosis is approximate as the best global path may be lost if it includes the early occurrence of many faults.

Algorithm 1 Incremental Diagnosis(Mod,$\mathcal{I}$,Obs,Que,λ,μ)

```
1:  S(0) := I(0);                          // I represents the initial states
2:  for i := 0 ; i < n ; i ++ do // Diagnoses the window (t_i, t_{i+1})
3:    while no solution found for (t_i, t_i + λ) do
4:      for (k := 0 ; k < K and no solution found ; k ++) do
5:        F := Mod(t_i, t_i + λ + μ) ∪ Obs(t_i, t_i + λ + μ) ∪
          Que_k(t_i, t_i + λ) ∪ S(i);
6:        if SAT(F) is satisfiable then
7:          extract_path(SAT(F));
8:          S(i + 1) := extract_state(SAT(F));
9:      if no solution found for (t_i, t_i + λ) then      // path reset
10:       S(i) = ∅
```

We propose Algorithm 1 for the ID of (t_0, t_n). Let K be the maximum number of faults that can occur during λ time steps. For each window $(t_i, t_i + \lambda)$, the SAT solver tries to find a path starting from state $S(i)$, consistent with the observations $(t_i, t_i + \lambda + \mu)$ by increasing the number of faults (lines 4–8). $\mathcal{F}$ is the CNF that models the set of contraints on the path we are looking for. When the path is found, the function `extract_path` extracts the path computed during $(t_i, t_i + \lambda)$. The function `extract_state` computes $S(i + 1)$

in order to force the next path to be a prolongation of the current path.

If no path is found starting from $S(i)$ for $(t_i, t_i + \lambda)$, the path for the previous window is not consistent with the new observations. For complexity reasons, backtracking is not allowed. The algorithm simply tries to find a new path that does not start from the previous path (line 10). We call this a *path reset*. When a path reset is performed, the path of (t_0, t_n) is not globally consistent. However for most systems, it can be expected that the misinterpretation of the observations will be only localised on a small time frame.

3 Empirical Validation

The experiments are conducted on an Intel Pentium 4 PC running at 3 GHz CPU, under Linux using MINISAT v2.0 [1]. For this study, we use the system presented in [2]. The maximum number of faults K is set to $1 + \lambda/2$ in the experiment.

n	100	200	299	300	400	500	599	600	699	700	799	800	899	900	999	1000
t	116	19	>2d	>2d	268	127	>2d	153	832	669	185	574	151	370	>2d	3204

Table 1. Runtime in seconds of MINISAT solver on nID satisfiable problem instances with n observations

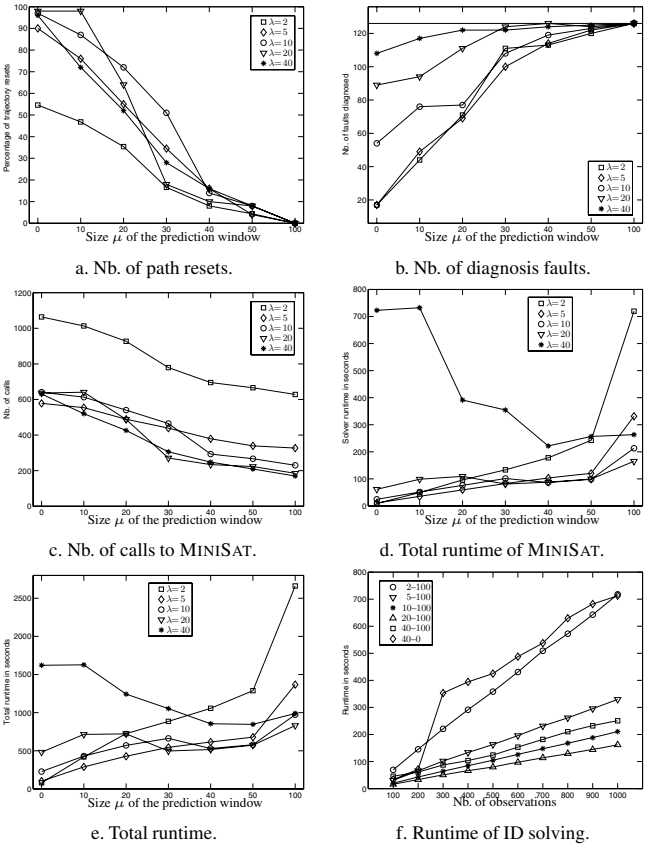
Table 1 shows the runtime required by MINISAT to find a scenario consistent with the n observations and containing $k(n) \simeq n/8$ faults. In this Table, t >2d means that the instance cannot be solved in 2 days. Note that this computation is not a diagnosis in the sense that it should first be proved that there is no path with k' faults where $k' < k(n)$, which is usually more expensive as these problems are unsatisfiable. Note that the runtime does not increase linearly but in a chaotic way, such as the difference between $n = 999$ and $n = 1000$.

We now run Algorithm 1 on the scenario of 1 000 observations by varying the parameter λ in the range of $\{2, 5, 10, 20, 40\}$ and the parameter μ in the range of $\{0, 10, 20, 30, 40, 50, 100\}$.

Quality of the diagnosis Figure 1a presents the percentage of path resets, and Figure 1b gives the number of faults computed for each pair of parameters. These measure the quality of the diagnosis. An accurate diagnosis should have no reset and the smallest number $d(0, 1000)$ of faults consistent with the observations (this value is unknown but less than 128). As expected, the number of resets decreases when the size of the prediction window increases. In this example, a value $\mu = 100$ is sufficient to avoid any reset. The Figure 1b also shows that a large diagnosis window partially avoids the bad-quality results of small prediction windows though it generates a big number of path reset. This is simply because enlarging the size of the diagnosis windows makes the incremental diagnosis more and more look like non incremental diagnosis.

Runtime Figure 1c gives the number of calls to MINISAT, Figure 1d presents the total runtime of MINISAT, and Figure 1e presents the total runtime including the preprocessing time. All the computations are done in less than one hour, which is better than the incomplete computations of Table 1.

The runtime generally increases when μ increases. Thus, a trade-off might be required here between quality and efficiency. Note that the tendency is inverted when λ is large because the number of path restart decreases; for large diagnosis windows, large prediction windows increase quality *and* efficiency. Finally, note that the smallest runtime is not achieve with smallest diagnosis windows but with medium-large diagnosis windows.



a. Nb. of path resets. b. Nb. of diagnosis faults.

c. Nb. of calls to MINISAT. d. Total runtime of MINISAT.

e. Total runtime. f. Runtime of ID solving.

Figure 1. Results of our incremental algorithm on nID problems.

Incremental runtime Figure 1f shows the evolution of the SAT runtime during the incremental diagnosis for some pairs $\langle \lambda, \mu \rangle$ (other pairs lead to similar results). The experiments clearly show a linear runtime for most pairs of parameters. Note however that small prediction windows potentially generates picks of computation.

These results validate our approach. The incremental algorithm of DES can be performed using SAT algorithms, and the runtime is lower than in a non-incremental approach. The results stress the importance of the parameters λ and μ both for efficiency and for diagnosis correctness. These parameters should be tested off-line before running the diagnosis to address the quality of diagnosis required and the resources available.

REFERENCES

[1] N. Eén and N. Sörensson, 'An extensible SAT-solver', in *Sixth International Conference on Theory and Applications of Satisfiability Testing (SAT-03)*, (2003).

[2] A. Grastien, Anbulagan, J. Rintanen, and E. Kelareva, 'Diagnosis of discrete-event systems using satisfiability algorithms', in *Proc. of 19th AAAI*, pp. 305–310, (2007).

[3] A. Grastien, M.-O. Cordier, and Ch. Largouët, 'Incremental diagnosis of discrete-event systems', in *Sixteenth International Workshop on Principles of Diagnosis (DX-05)*, pp. 119–124, (2005).

[4] G. Lamperti and M. Zanella, *Diagnosis of Active Systems*, Kluwer Academic Publishers, 2003.