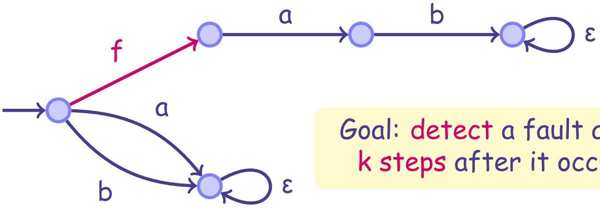
A Note on Fault Diagnosis Algorithms

Franck Cassez

National ICT Australia & CNRS
Sydney, Australia

December 18th, 2009
CDC'09, Shanghai, China

Fault Diagnosis for Discrete Event Systems



Goal: detect a fault at most k steps after it occurred

Given:

- A finite automaton A over $\Sigma^{\varepsilon,f} = \Sigma \cup \{\varepsilon, f\}$
- f is the fault action, Σ is the set of observable events

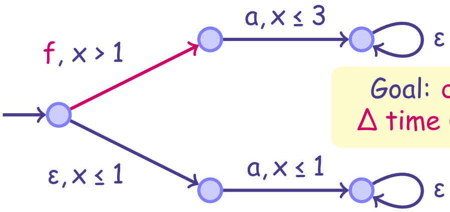
Define:

- **Faulty$_{\geq k}$**(A): k-faulty runs that contain f followed by $\geq$ k actions
- **NonFaulty**(A): Non faulty runs that contain no f

Purpose of fault diagnosis: given k, and observable events Σ

- never raise an alarm on non-faulty runs
- always raise an alarm on k-faulty runs

Fault Diagnosis for Dense-Time Systems



Goal: detect a fault at most Δ time units after it occurred

Given:

- A timed automaton with continuous variables A over $\Sigma^{\varepsilon,f} = \Sigma \cup \{\varepsilon, f\}$
- f is the fault action, Σ is the set of observable events

Define:

- **Faulty$_{\geq\Delta}$**(A): Δ-faulty runs that contain f followed by $\geq \Delta$ time units
- **NonFaulty**(A): Non faulty runs (contain no f)

Purpose of fault diagnosis: given Δ, and observable events Σ

- never raise an alarm on non-faulty runs
- always raise an alarm on Δ-faulty runs

Diagnosability Problem

trace(ρ) $=$ trace of the run ρ (a word in $(\Sigma \cup \{\varepsilon, f\})^*$)
$\pi_{/\Sigma}(\text{trace}(\rho))$ $=$ projection of the trace on observable events

Definition (k-diagnoser)

A mapping $D : \Sigma^* \longrightarrow \{0,1\}$ is a k-diagnoser for A if:

- for each run $\rho \in \textbf{NonFaulty}(A)$, $D(\pi_{/\Sigma}(\text{trace}(\rho))) = 0$;
- for each run $\rho \in \textbf{Faulty}_{\geq k}(A)$, $D(\pi_{/\Sigma}(\text{trace}(\rho))) = 1$.

k-Diagnosability Problem

Given A and $k \in \mathbb{N}$, is there a k-diagnoser for A?

Diagnosability Problem

Given A, is there a $k \in \mathbb{N}$ s.t. A is k-diagnosable ?

Dense-time version defined using timed words, and timed languages

Algorithms for Checking Diagnosability

Necessary and Sufficient Condition for Diagnosability

$$A \text{ is not diagnosable} \iff \forall k \in \mathbb{N}^*, A \text{ is not } k\text{-diagnosable}$$

Results for discrete event and dense-time systems

Diagnosability reduces to checking Büchi emptiness

Diagnosability reduces to bounded diagnosability (reachability)

Complexity

	Δ-Diagnosability	Diagnosability	
	Reachability Algorithm	Büchi Emptiness	Reachability
DES	PTIME $O(\|A\|^4)$	PTIME $O(\|A\|^2)$	PTIME $O(\|A\|^4)$
TA	PSPACE-C.	PSPACE-C. $O(\|A\|^2)$	PSPACE-C. $O(\|A\|^4)$

Consequences & Applications

- Easy proofs of existing results
  [Sampath et al., 95, Jiang et al., 2001, Yoo et al., 2002]
- Shows that Büchi based algorithms are better
- Use of standard model-checking tools for the diagnosability problem
  - on-the-fly algorithms: SPIN, NuSMV
  - efficient tools for timed systems: UPPAAL
- Expressive languages for specifying systems

Selected References

[Jiang et al., 2001] Shengbing Jiang, Zhongdong Huang, Vigyan Chandra, and Ratnesh Kumar.
A polynomial algorithm for testing diagnosability of discrete event systems.
IEEE Transactions on Automatic Control, 46(8), August 2001.

[Sampath et al., 95] Meera Sampath, Raja Sengupta, Stephane Lafortune, Kasim Sinnamohideen, and Demosthenis C. Teneketzis.
Diagnosability of discrete event systems.
IEEE Transactions on Automatic Control, 40(9), September 1995.

[Yoo et al., 2002] Yoo, T.-S., Lafortune, S.
Polynomial-Time Verification of Diagnosability of Partially-Observed Discrete-Event Systems,
IEEE Transactions on Automatic Control, 47(9), September 2002, 1491–1495.