Fault-Tolerant Supervisory Control

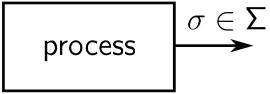
Thomas Moor
Friedrich-Alexander Universität Erlangen-Nürnberg

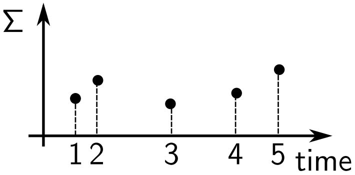
IFAC DCDS 2015
Cancun, Mexico

1. Supervisory Control
   - Discrete-Event Systems
   - Closed-Loop Configuration
   - Controller Synthesis
2. Fault-Tolerant Control
   - Naive Approach
   - Active Fault-Tolerant Control
   - Post-Fault Recovery
   - Fault-Hiding Approach

A discrete-event system is a model of a process
... with a particular focus the ocurrence of events

- finite set Σ of symbols $\sigma \in \Sigma$ (alphabet)
- only event ordering is regarded relevant (logic time)
- within finite time a finite sequence $s \in \Sigma^*$ is generated
- set $L \subseteq \Sigma^*$ of sequences that can be generated





A discrete-event system is a model of a process
… with a particular focus the ocurrence of events

- finite set Σ of symbols $\sigma \in \Sigma$ (alphabet)
- only event ordering is regarded relevant (logic time)
- within finite time a finite sequence $s \in \Sigma^*$ is generated
- set $L \subseteq \Sigma^*$ of sequences that can be generated
- write $\mathrm{pre}\, L$ to emphasise that $L = \mathrm{pre}\, L$ (local behaviour)

prefix operator $\mathrm{pre}\, L := \{s | \exists\, t : st \in L\}$

A closed language $\mathrm{pre}\, L \subseteq \Sigma^*$ is a discrete-event system.

Literature: Ramadge and Wonham 1989

A natural domain of interpretation for models with liveness properties are ω-languages, i.e., sets of inf. strings $w \in \Sigma^\omega$.

In the absence of deadlocks, use

$$\lim \operatorname{pre} L := \{ w \in \Sigma^\omega \mid \operatorname{pre} w \subseteq \operatorname{pre} L \}$$

to model the process w.r.t. infinite time.

If, in addition, there are no livelocks, choose L s.t. $L = M \cap \operatorname{pre} L$ and consider

$$\lim L := \{ w \in \Sigma^\omega \mid \|(\operatorname{pre} w \cap L)\| = \infty \},$$

to model the process w.r.t. infinite time.

Properties

- safety – bad things never happen

  with $\operatorname{pre} E \subseteq \Sigma^*$, require

  $$\operatorname{pre} L \subseteq \operatorname{pre} E$$

- liveness – good things do happen

  free of deadlocks

  $$(\forall\, s \in \operatorname{pre} L)(\exists\, \sigma \in \Sigma)[\, s\sigma \in \operatorname{pre} L\,]$$

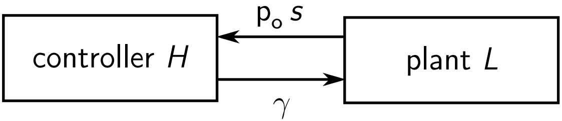
  free of livelocks wrt $M \subseteq \Sigma^*$

  $$(\forall\, s \in \operatorname{pre} L)(\exists\, t \in \Sigma^*)[\, st \in M \cap \operatorname{pre} L\,]$$

For systems with liveness properties:

A language $L \subseteq \Sigma^*$ is a discrete-event system.

With the common partitioning $\Sigma = \Sigma_c \dot\cup \Sigma_{uc} = \Sigma_o \dot\cup \Sigma_{uo}$ regarding controllable and observable events, consider a plant $L \subseteq \Sigma^*$.



- At any time, the controller is provided $p_o\, s \in \Sigma_o^*$ where $s \in \Sigma^*$ is the sequence generated so far;
- in return, the controller applies a control pattern γ of enabled events, where $\Sigma_{uc} \subseteq \gamma$;
- liveness properties of the plant shall be retained.

natural projection $p_o\, \Sigma^* \to \Sigma_o^*$ to remove any symbols not from Σ_o.

Literature: Ramadge and Wonham 1987, Lin and Wonham 1988

Def. A controller $H \subseteq \Sigma^*$ is *admissible w.r.t. the plant* $L \subseteq \Sigma^*$, if

[H0] $H = \mathrm{pre}\, H$

[H1] $H\Sigma_{\mathrm{uc}} \subseteq H$,

[H2] $H = \mathrm{p_o^{-1}}\, \mathrm{p_o}\, H$,

[H3] $(\mathrm{pre}\, L) \cap (\mathrm{pre}\, H)$ does not deadlock, and

[H4] $(\mathrm{pre}\, L) \cap (\mathrm{pre}\, H) = \mathrm{pre}\,(L \cap H)\,.$

Then, $K := L \cap H$ represents the cosed-loop behaviour. $\square$

Thm. Consider tha case $\Sigma_c \subseteq \Sigma_o$. For a plant $L \subseteq \Sigma^*$ and an admissible controller $H \subseteq \Sigma^*$ let $K = L \cap H$. Then

[K0] K is relatively prefix-closed w.r.t. L,

[K1] K is controllable w.r.t. L,

[K2] K prefix-normal w.r.t. L, and

[K3] K does not deadlock.

Vice versa, if K satisfies [K0]-[K3], then there exists an admissible controller H such that $K = L \cap H$. $\square$

Literature: Lin and Wonham 1988, Kumar et al 1992, Moor et al 2012

Control Problem. Given (L, E) with plant $L \subseteq \Sigma^*$ and a specification $E \subseteq \Sigma^*$ construct an admissible controller $H \subseteq \Sigma^*$ such that

$$K := L \cap H \subseteq E.$$

Solution. All properties properties are retained under arbitraty union. Thus

$$K^{\uparrow} = \sup\{K \subseteq L \cap E \mid K \text{ satisfies [K0]–[K3]}\}$$

itself satisfies [K0]–[K3] and is used to extract a minimal restrictive controller.

Note. E can be substituted by a closed language without affecting solutions – it is effectivly a safety specification.

Control Problem. Given (L, E) with plant $L \subseteq \Sigma^*$ and a specification $E \subseteq \Sigma^*$ construct an admissible controller $H \subseteq \Sigma^*$ such that

$$K := L \cap H \subseteq E.$$

Solution. All properties properties are retained under arbitraty union. Thus

$$K^{\uparrow} = \sup\{K \subseteq L \cap$$

itself satisfies [K0]–[K3] and is u
controller.

Note. E can be substituted by
solutions – it is effectivly a safet

Interpretation by corresponding $\omega-$ *languages*

- in general, E can not be substituted by a closed language
- if E is (rel.) closed, same solution procedures as with $*$-languages (Ramadge 1989, Kumar et al 1993, Moor et at 2012)
- if E is not (rel.) closed it imposes liveness properties — completely different story
- solution procedure for $\Sigma_o = \Sigma$ by Thistle and Wonham 1994
- solution procedure for $\Sigma_o \neq \Sigma$ and closed L by Thistle and Lamouchi 2009

1. Supervisory Control

    Discrete-Event Systems

    Closed-Loop Configuration

    Controller Synthesis

2. Fault-Tolerant Control

    Naive Approach

    Active Fault-Tolerant Control

    Post-Fault Recovery

    Fault-Hiding Approach

Fault-Tolerant Control

- a fault is a sudden change of behaviour
- passive approach: have a single controller that can handle pre-fault and post-fault behaviour (robust control)
- active approach: detect the fault and switch to another controller (adaptive control)
- core challenge: switching of plant and controller dynamics

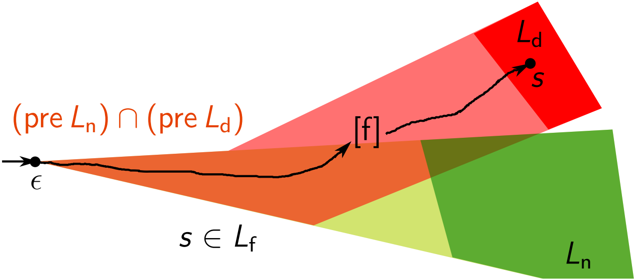
... for continuous systems. However, for discrete-event systems ...

Sudden change of behaviour and switching in the control scheme are the very nature of discrete-event systems. Hence, fault-tolerant control can be synthesised by the same methods as nominal control [??]

Naive approach to fault-tolerant control:

- nominal plant $L_\mathrm{n} \subseteq \Sigma_\mathrm{n}$
- fault event $\mathrm{f} \not\in \Sigma_\mathrm{n}$, uncontrollable and unobservable
- degraded post-fault behaviour $L_\mathrm{d} \subseteq \Sigma_\mathrm{f}^*$ with $\Sigma_\mathrm{f} = \Sigma_\mathrm{n} \dot\cup \{\mathrm{f}\}$ and

$$(\mathrm{pre}\, L_\mathrm{d}) \cap \Sigma_\mathrm{n}^* \subseteq \mathrm{pre}\, L_\mathrm{n} \quad \text{and} \quad L_\mathrm{d} \cap \Sigma_\mathrm{n}^* = \emptyset$$



Naive approach to fault-tolerant control:

- nominal plant $L_\mathsf{n} \subseteq \Sigma_\mathsf{n}$
- fault event $\mathsf{f} \notin \Sigma_\mathsf{n}$, uncontrollable and unobservable
- degraded post-fault behaviour $L_\mathsf{d} \subseteq \Sigma_\mathsf{f}^*$ with $\Sigma_\mathsf{f} = \Sigma_\mathsf{n} \dot{\cup} \{\mathsf{f}\}$ and

$$(\mathrm{pre}\, L_\mathsf{d}) \cap \Sigma_\mathsf{n}^* \subseteq \mathrm{pre}\, L_\mathsf{n} \quad \text{and} \quad L_\mathsf{d} \cap \Sigma_\mathsf{n}^* = \emptyset$$

- *fault-accommodating model* $L_\mathsf{f} = L_\mathsf{n} \cup L_\mathsf{d}$ where

$$\mathrm{pre}\, L_\mathsf{f} = (\,\mathrm{pre}\, L_\mathsf{n}\,) \cup (\,\Sigma_\mathsf{n}^* \mathsf{f} \Sigma_\mathsf{f}^* \cap \mathrm{pre}\, L_\mathsf{d})$$
$$L_\mathsf{f} = L_\mathsf{n} \cup (\,\Sigma_\mathsf{n}^* \mathsf{f} \Sigma_\mathsf{f}^* \cap L_\mathsf{d}\,)$$

- likewise, the specification $E_\mathsf{f} = E_\mathsf{n} \cup E_\mathsf{d}$ to accommodate for degraded post-fault performance E_d

Naive approach to fault-tolerant control (cnt.)

- invoke synthesis procedure for $(L_\mathsf{f}, E_\mathsf{f})$ to obtain a minimal restrictive admissible fault-tolerant controller H_f.
- note: diagnosibility required only relative to specifications
- option: re-interpretation H_f as active fault-tolerant control

switch on first escape from $H_\mathbf{n}$:

$$T := \{s \in \Sigma_\mathbf{f}^* \,|\, \exists\, \sigma : s\sigma \in H_\mathsf{f} \not\leftrightarrow s\sigma \in \mathsf{p}_\mathbf{f}^{-1}\, H_\mathbf{n}\}$$

$$H_\mathbf{d} := \{s\sigma \in H_\mathsf{f} \,|\, (\mathrm{pre}\, s) \cap T \neq \emptyset\} \cup \{\epsilon\}$$

$$H_\mathsf{f} = ((\mathsf{p}_\mathbf{f}^{-1}\, H_\mathbf{n}) \cap (\Sigma_\mathbf{f}^* - T\Sigma_\mathsf{f}\Sigma_\mathbf{f}^*)) \cup H_\mathbf{d}$$

Naive approach to fault-tolerant control (cnt.)

- invoke synthesis procedure for $(L_\mathsf{f}, E_\mathsf{f})$ to obtain a minimal restrictive admissible fault-tolerant controller H_f.
- note: diagnosibility required only relative to specifications
- option: re-interpretation H_f as active fault-tolerant control
- note: in general, $H_\mathsf{f} \cap \Sigma_\mathsf{n}^*$ is not admissible w.r.t. L_n

> this requires two more closed-loop properties for the synthesis of H_f:
> [K4] $K \cap \Sigma_\mathsf{n}^*$ does not deadlock
> [K5] $\mathrm{pre}\,(K \cap \Sigma_\mathsf{n}^*) = (\mathrm{pre}\,K) \cap \Sigma_\mathsf{n}^*$

Literature: Wittmann et al 2012

Naive approach to fault-tolerant control (cnt.)

- invoke synthesis procedure for $(L_\mathrm{f}, E_\mathrm{f})$ to obtain a minimal restrictive admissible fault-tolerant controller H_f.
- note: diagnosibility required only relative to specifications
- option: re-interpretation H_f as active fault-tolerant control
- note: in general, $H_\mathrm{f} \cap \Sigma_\mathrm{n}^*$ is not admissible w.r.t. L_n
- option: compute a minimal restrictive nominal controller H_n that solves $(L_\mathrm{n}, E_\mathrm{n})$ and test whether $L_\mathrm{f} \cap H_\mathrm{f} \cap \Sigma_\mathrm{n}^* = L_\mathrm{n} \cap H_\mathrm{n}$

we have $L_\mathrm{f} \cap H_\mathrm{f} \cap \Sigma_\mathrm{n}^* \subseteq E_\mathrm{f} \cap \Sigma_\mathrm{n}^* = E_\mathrm{n}$ for free, and, if $H_\mathrm{f} \cap \Sigma_\mathrm{n}^*$ is admissible w.r.t. L_n,

$L_\mathrm{f} \cap H_\mathrm{f} \cap \Sigma_\mathrm{n}^* \subseteq L_\mathrm{n} \cap H_\mathrm{n}$

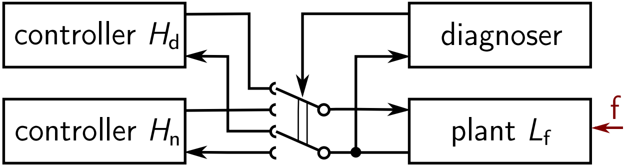
Naive approach to fault-tolerant control (cnt.)

- invoke synthesis procedure for $(L_\mathrm{f}, E_\mathrm{f})$ to obtain a minimal restrictive admissible fault-tolerant controller H_f.
- note: diagnosibility required only relative to specifications
- option: re-interpretation H_f as active fault-tolerant control
- note: in general, $H_\mathrm{f} \cap \Sigma_\mathrm{n}^*$ is not admissible w.r.t. L_n
- option: compute a minimal restrictive nominal controller H_n that solves $(L_\mathrm{n}, E_\mathrm{n})$ and test whether $L_\mathrm{f} \cap H_\mathrm{f} \cap \Sigma_\mathrm{n}^* = L_\mathrm{n} \cap H_\mathrm{n}$
- option: explicit diagnosis by controllable event $\mathrm{F} \in \Sigma_\mathrm{n}$ with plant $\mathrm{p}_\mathrm{F}^{-1}\, L_\mathrm{f}$ and specification $(\mathrm{p}_\mathrm{F}^{-1}\, E_\mathrm{f}) \cap \mathrm{pre}\,(\Sigma_\mathrm{n}^* \mathrm{f} \Sigma_\mathrm{f}^* \mathrm{F} \Sigma_\mathrm{f}^*)$

need to interpret F as forcible event

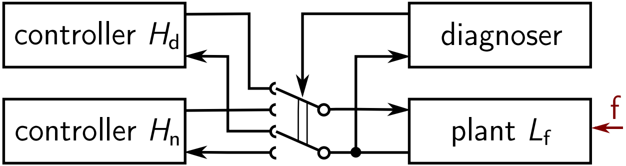
Literature: Wittmann et al 2012

Active fault-tolerant control



- require the fault to be diagnosible, denote $T \subseteq L_d$ the strings corresponding to f-certain diagnoser states
- require/test that the post-fault behaviour satisfies a safety specification (safe diagnosibility)
- design H_d to take over H_n when the plant first enters T
- note: nominal pre-fault behaviour is guaranteed
- option: synthesise H_d online once the fault has been detected

Active fault-tolerant control



- require the fault to be diagnosible, denote $T \subseteq L_d$ the strings corresponding to f-certain diagnoser states
- require/test that the post-fault bel specification (safe diagnosibility)
- design H_d to take over H_n when th
- note: nominal pre-fault behaviour i
- option: synthesise H_d online once t

Diagnosis of DES (Sampath et al 1995)

diagnoser: observer automaton with dedicated state labels

f-certain state: state in which the fault must have occured.

diagnosibility: require the plant to evolve to an f-certain state after a bounded number of transitions.

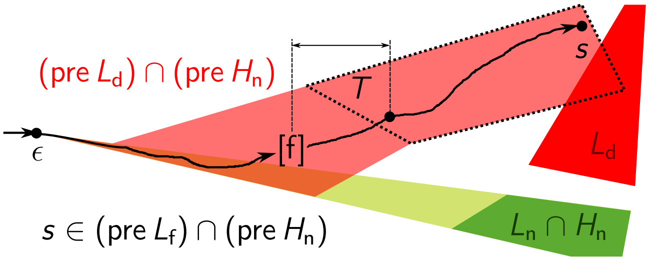
Literature: Paoli et al 2005, 2008, 2011

- diagnosibility ensures that every string after f evolves into T

$$T = \{ s \in (\mathrm{pre}\, L_\mathrm{d}) \cap (\mathrm{pre}\, H_\mathrm{n}) \,|\, (\mathrm{p}_\mathrm{o}^{-1}\, \mathrm{p}_\mathrm{o}\, s) \cap (\mathrm{pre}\, L_\mathrm{f}) \subseteq \Sigma_\mathrm{n}^* \mathrm{f} \Sigma_\mathrm{f}^* \}$$

- safe diagnosibility ensures that

$$[(\mathrm{pre}\, L_\mathrm{d}) \cap (\mathrm{pre}\, H_\mathrm{n})] - T\Sigma_\mathrm{f}^* \subseteq E_\mathrm{d}$$



Literature: Paoli et al 2005, 2008, 2011

- diagnosibility ensures that every string after f evolves into T

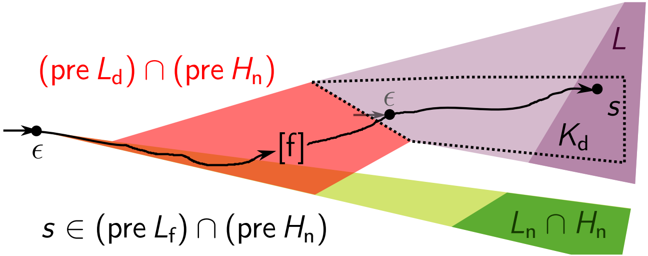
$$T = \{s \in (\mathrm{pre}\, L_\mathrm{d}) \cap (\mathrm{pre}\, H_\mathrm{n}) \,|\, (\mathrm{p}_\mathrm{o}^{-1}\, \mathrm{p}_\mathrm{o}\, s) \cap (\mathrm{pre}\, L_\mathrm{f}) \subseteq \Sigma_\mathrm{n}^* \mathrm{f} \Sigma_\mathrm{f}^* \}$$

- safe diagnosibility ensures that

$$[(\mathrm{pre}\, L_\mathrm{d}) \cap (\mathrm{pre}\, H_\mathrm{n})] - T\Sigma_\mathrm{f}^* \subseteq E_\mathrm{d}$$

- synthesise H_d for the post-fault detection plant

$$L = \{s | \exists\, t : ts \in L_\mathrm{d} | (\mathrm{pre}\, s) \cap T = s \in H_\mathrm{n}\}$$



- diagnosibility ensures that every string after f evolves into T

$$T = \{s \in (\mathrm{pre}\, L_\mathrm{d}) \cap (\mathrm{pre}\, H_\mathrm{n}) \,|\, (\mathrm{p}_\mathrm{o}^{-1}\, \mathrm{p}_\mathrm{o}\, s) \cap (\mathrm{pre}\, L_\mathrm{f}) \subseteq \Sigma_\mathrm{n}^* \mathrm{f} \Sigma_\mathrm{f}^* \}$$

- safe diagnosibility ensures that

$$[(\mathrm{pre}\, L_\mathrm{d}) \cap (\mathrm{pre}\, H_\mathrm{n})] - T\Sigma_\mathrm{f}^* \subseteq E_\mathrm{d}$$

- synthesise H_d for the post-fault detection plant

$$L = \{s | \exists\, t : ts \in L_\mathrm{d} | (\mathrm{pre}\, s) \cap T = s \in H_\mathrm{n}\}$$

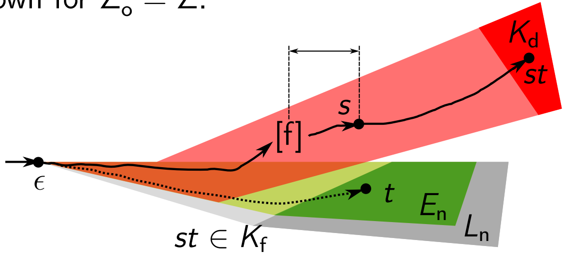
- re-interpret within naive approach:
  - synthesise H_f with $\mathrm{pre}\, \{s\sigma \in T | s \notin T\} \subseteq E_\mathrm{d}$
  - test for $L_\mathrm{f} \cap H_\mathrm{f} \cap \Sigma_\mathrm{n}^* = L_\mathrm{n} \cap H_\mathrm{n}$
  - extract H_d from H_f
  - mimique re-initialisation

Post-Fault Recovery: nominal safety specification

- add a formal closed-loop requirement to [K0]-[K3] for the synthesis of H_f

[K6] $\qquad E_\mathrm{n} \Leftarrow K_\mathrm{f}/(\Sigma_\mathrm{n}^*\mathrm{f})$

- although [K6] is not retained under union, synthesis procedures are known for $\Sigma_\mathrm{o} = \Sigma$.

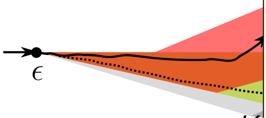


Post-Fault Recovery: nominal safety specification

- add a formal closed-loop re
  synthesis of H_f

  [K6] $\qquad E_n \Leftarrow K_f/(\Sigma_n^* f)$

- although [K6] is not retaine
  are known for $\Sigma_o = \Sigma$.



$st \in K_f$

> Language Convergence (Kumar et al 1993)
>
> K is said to *finitely converge* to E if there exists a uniform bound k such that every $s \in K$ can be decomposed
>
> $$s = vw,\ w \in E, \text{ and } |v| \le k.$$
>
> This is written $E \Leftarrow K$.
>
> Without the uniform bound, the condition becomes equivalent to
>
> $$K \subseteq \Sigma^* E,$$
>
> which quite weak.
>
> Alternative approach: refer to the respective ω-languages and require
>
> $$\lim K \subseteq \lim(\Sigma^* E)$$
>
> for not-uniform bounded convergence.

Post-Fault Recovery: liveness

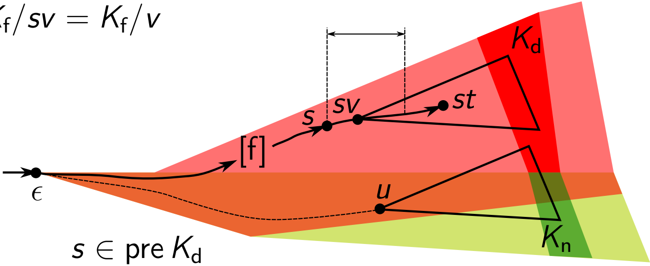
- a closed loop $K_f = K_n \cup K_d$ is *fault toleant* if

  [K7] there exists a uniform bound k such that for every s, t, $|t| \geq k$ with

  $$s \in (\text{pre } K_f) - (\text{pre } K_n) \text{ and } st \in \text{pre } K_f$$

  there exists $u \in \text{pre } K_n$, $v \in \text{pre } t$, $|v| \leq k$ with

  $$K_f/sv = K_f/v$$



$s \in \text{pre } K_d$

Post-Fault Recovery: liveness

- a closed loop $K_\mathsf{f} = K_\mathsf{n} \cup K_\mathsf{d}$ is *fault toleant* if

  [K7] there exists a uniform bound k such that for every s, t, $|t| \geq k$ with

  $$s \in (\text{pre}\, K_\mathsf{f}) - (\text{pre}\, K_\mathsf{n}) \text{ and } st \in \text{pre}\, K_\mathsf{f}$$

  there exists $u \in \text{pre}\, K_\mathsf{n}$, $v \in \text{pre}\, t$, $|v| \leq k$ with

  $$K_\mathsf{f}/sv = K_\mathsf{f}/v$$

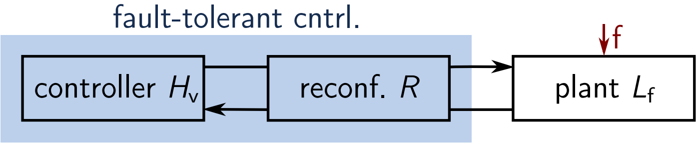
- synthesis problem: given $L_\mathsf{f} = L_\mathsf{n} \cup L_\mathsf{d}$ and E_f, compute an admissible controller H_f such that the closed loop satisfies [K7].

- the property is not retained under union; synthesis procedure exists for $\Sigma_\mathsf{o} = \Sigma$

Literature: Wen et al 2008, 2014

Fault Hiding

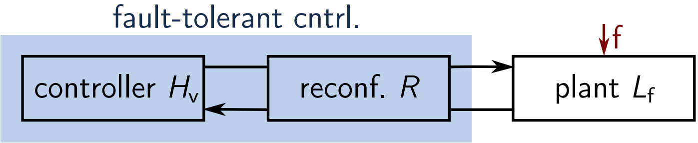
Given $L_\mathrm{f} = L_\mathrm{n} \cup L_\mathrm{d}$, $E_\mathrm{f} = E_\mathrm{n} \cup E_\mathrm{d}$, and a solution H_n to $(L_\mathrm{n}, E_\mathrm{n})$



- disconnect nominal controller, i.e., $H_\mathrm{v} = h(H_\mathrm{n}) \subseteq \Sigma_\mathrm{v}^*$ with $\Sigma_\mathrm{v} \cap \Sigma_\mathrm{f} = \emptyset$, h bijective and applied per event.
- synthesise reconfiguration dynamics $R \subseteq (\Sigma_\mathrm{v} \cup \Sigma_\mathrm{o})^*$ to re-connect
- do so by interpreting $H_\mathrm{v} \parallel L_\mathrm{f}$ as plant and use std. procedures on adapted language inclusion specification

Fault Hiding

Given $L_\mathrm{f} = L_\mathrm{n} \cup L_\mathrm{d}$, $E_\mathrm{f} = E_\mathrm{n} \cup E_\mathrm{d}$, and a solution H_n to $(L_\mathrm{n}, E_\mathrm{n})$



- when using a minimal restrictive solution $H_\mathrm{v}^\uparrow$ for the design, and if the closed loop K satisfies in addition to [K0]-[K3]

  [K8] $$(\forall\, s \in \mathrm{pre}\, K\,)[\ ((\mathrm{p_v}\, s) \ \cap\ h(\Sigma_\mathrm{uc})\ \cap\ (\mathrm{pre}\, h(L_\mathrm{n})) \neq \emptyset$$
  $$\Rightarrow\ \ s(\Sigma - h(\Sigma_\mathrm{c}))^* h(\Sigma_\mathrm{uc})\ \cap\ (\mathrm{pre}\, K) \neq \emptyset\ ]$$

  then R is admissible to any nominal controller that solves $(L_\mathrm{n}, E_\mathrm{n})$.

- [K8] is retained under union, synthesis procedures are known.

- note: nominal controller does not need to be known

Literature: Wittmann et al 2013

Summary

Fault-tolerant supervisory control is addressed by the recent literature in various ways, including passive and active approaches, post-fault recovery and fault-hiding.

Conclusions

- switching is addressed by the common modelling framework — any method for fault-tolerant supervisory control should be interpretable within this framework
- additional features of individual approaches amount to additional closed-loop properties — and novel synthesis problems
- insisting in uniform bounds for diagnosibility and language convergence may be too strict for particular applications — discussion in terms of ω-languages may turn out beneficial

Literature — Supervisory Control

Kumar, R., Garg, V., and Marcus, S. (1993). Language stability and stabilizability of discrete event dynamical systems. SIAM J. Control and Optimization, 31, 1294–1320.

Lin, F. and Wonham, W.M. (1988). On observability of discrete-event systems. Information Sciences, 44, 173–198.

Moor, T., Baier, C., Yoo, T.S., Lin, F., and Lafortune, S. (2012). On the computation of supremal sublanguages relevant to supervisory control. WODES, 175–180.

Ramadge, P.J. and Wonham, W.M. (1989). The control of discrete event systems. Proceedings of the IEEE, 77, 81–98.

Ramadge, P.J. (1989). Some tractable supervisory control problems for discrete-event systems modeled by büchi automata. IEEE TAC, 34, 10–19.

Sampath, M., Sengupeta, R., , Lafortune, S., and Sinnamohideen, K. (1995). Diagnosability of discrete-event systems. IEEE TAC, 40:9, 1555–1575.

Thistle, J.G. and Lamouchi, H.M. (2009). Effective control synthesis for partially observed discrete-event systems. SIAM J. Control and Optimization, 48, 1858–1887.

Thistle, J.G. and Wonham, W.M. (1994). Supervision of infinite behavior of discrete event systems. SIAM J. Control and Optimization, 32, 1098–1113.

Willner, Y. and Heymann, M. (1995). Language convergence in controlled discrete-event systems. IEEE TAC, 40(4), 616–627.

[**more references are given in the paper**]

Literature — Fault-Tolerant Control

Paoli, A. and Lafortune, S. (2005). Safe diagnosability for fault-tolerant supervision of discrete-event systems. Automatica, 41(8), 1335–1347.

Paoli, A., Sartini, M., and Lafortune, S. (2008). A fault tolerant architecture for supervisory control of discrete event systems. Proceedings of the 17th IFAC world congress, 6542–6547.

Paoli, A., Sartini, M., and Lafortune, S. (2011) . Active fault tolerant control of discrete event systems using online diagnostics. Automatica, 47(4), 639–649.

Sülek, A.N. and Schmidt, K.W. (2014). Computation of supervisors for fault-recovery and repair for discrete event systems. WODES, 428–438.

Wen, Q., Kumar, R., and Huang, J. (2014). Framework for optimal fault-tolerant control synthesis: maximize prefault while minimize post-fault behaviorsfor discrete event systems. IEEE Trans. Syst. Man Cybern. Syst., 44, 1056–1066.

Wen, Q., Kumar, R., Huang, J., and Liu, H. (2008). A framework for fault-tolerant control for discrete event systems. IEEE TAC, 53, 1839–1849.

Wittmann, T., Richter, J., and Moor, T. (2012). Fault-tolerant control of discrete event systems based on fault-accommodating models. SAFEPROCESS, 854–859.

Wittmann, T., Richter, J., and Moor, T. (2013). Fault-hiding control reconfiguration for a class of discrete-event systems. IFAC DCDS.

[**more references are given in the paper**]