Petri Net model-based distributed diagnosis

George Jiroveanu and René Boel
EESA-SYSTeMS research Group
Ghent University

goal

- On-line Model based fault detection and identification based on observation of occurrence of some events (alarms, messages,…)
- Computationally efficient algorithms for large Discrete Event Systems

  modelled as interacting Petri net components
- Using communication between local agents of individual components

goal

Intended applications:

Power system backup protection

Incident detection in road traffic network

Extensions?

Methodology should allow easy extension to

- time(d) Petri net models
- probabilistic DES models

outline

- Petri net models of interacting components
- Distributed diagnosis with communication between local agents
- Methodology: Backward search

  to generate minimal explanations
- Distributed diagnosis algorithm
- Conclusions and extensions

Petri net modelling

- Petri net encodes the constraints on the evolution of discrete event systems
- This statement is equivalent to:

  Petri net with given initial state defines

  language = set of allowable traces of events

- “state” encoded by “marking of places”

Petri net definition

- structure: a Petri net $\mathbb{N} = (P, T, \mathrm{Pre}, \mathrm{Post})$

where P = set of places

T = set of transitions

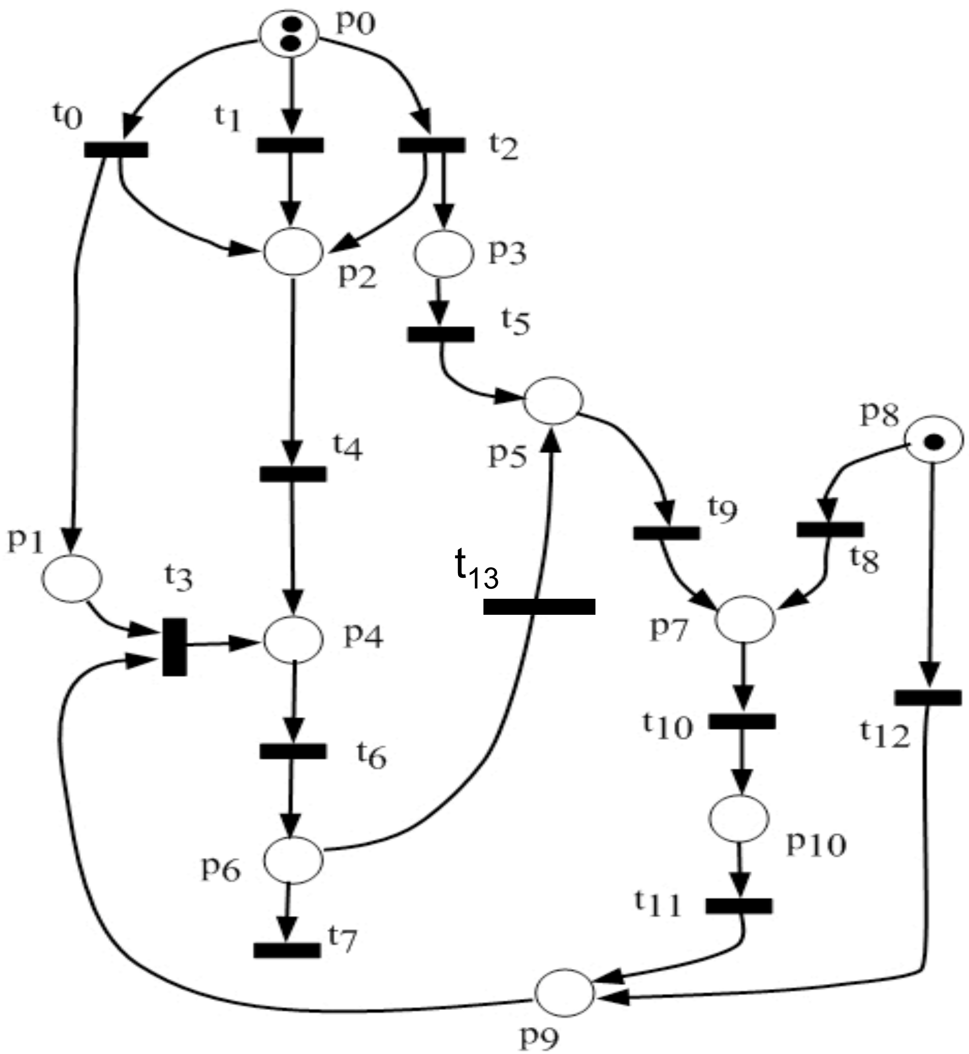
Pre = mapping from $P\ x\ T$ into {0, 1}

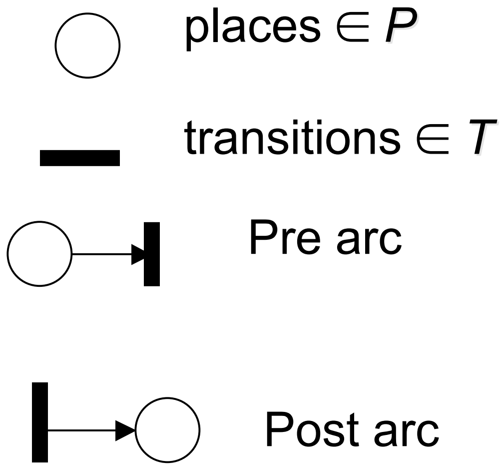
= incidence function defining arcs from place to transition

Post = mapping from $T\ x\ P$ into {0, 1}

= incidence function defining arcs from transition to place

Petri net: example

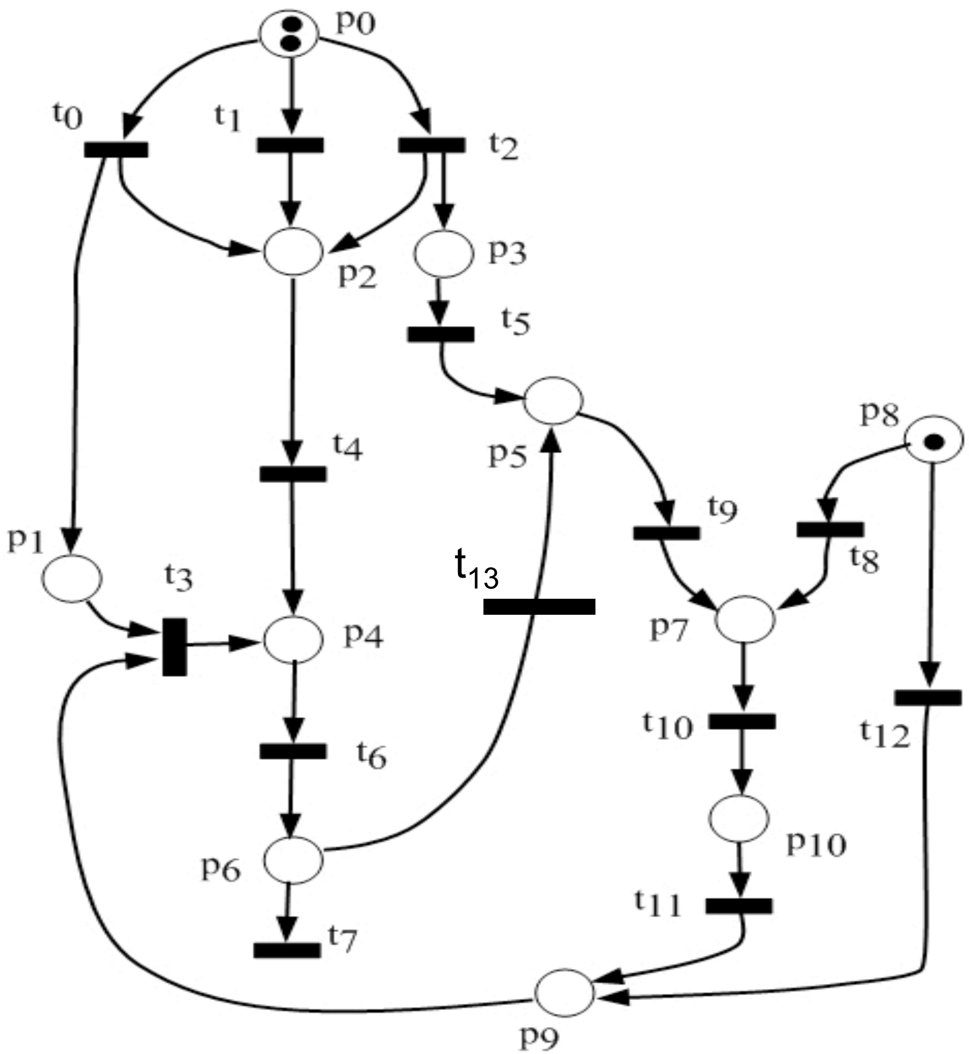


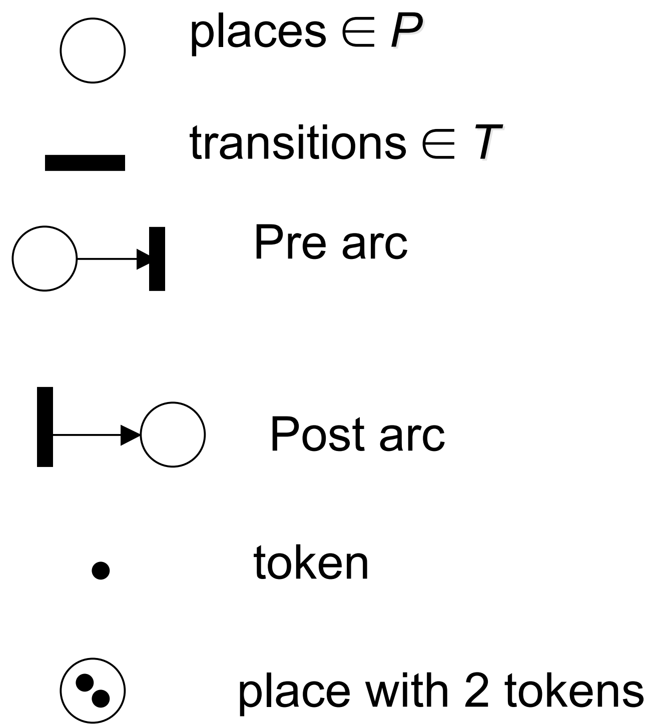


Petri net: marking (state)

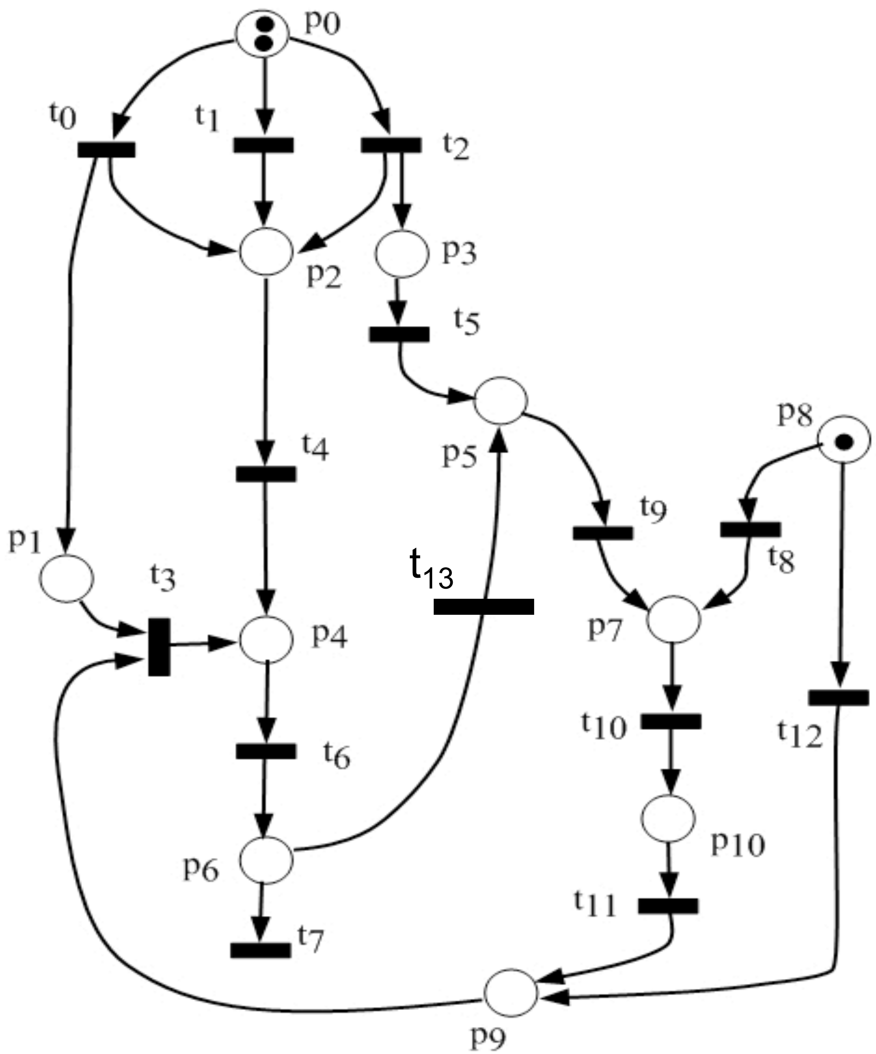
- Marking m of Petri net $\mathbb{N}$: $P \rightarrow N$ is integer-valude # P-vector assigning to each place p a (natural) number m(p) of tokens

Petri net: example





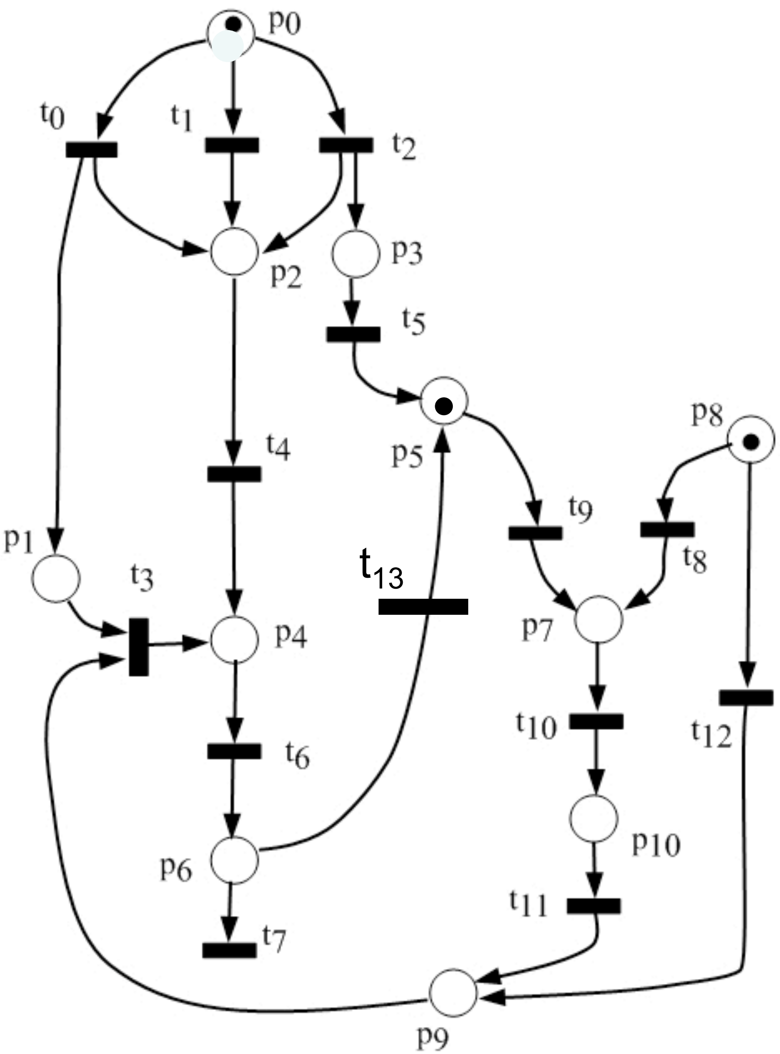
Petri net: dynamic behaviour



enabling rule:
transition t is enabled
if $\forall p$: $Pre(p,t) \leq m(p)$
if all input places of t contain at least 1 token

in example transitions
t_0, t_1, t_2, t_8, t_{12}
are enabled

Petri net: dynamic behaviour



if t is enabled then t can fire

firing rule:
when t fires new marking
$m' = m - Pre(.,t) + Post(t,.)$
is produced

in example when enabled transition t_2 fires the new marking m' puts 1 token in p_0, p_3, p_8

Petri net: example of evolution

execute t_3

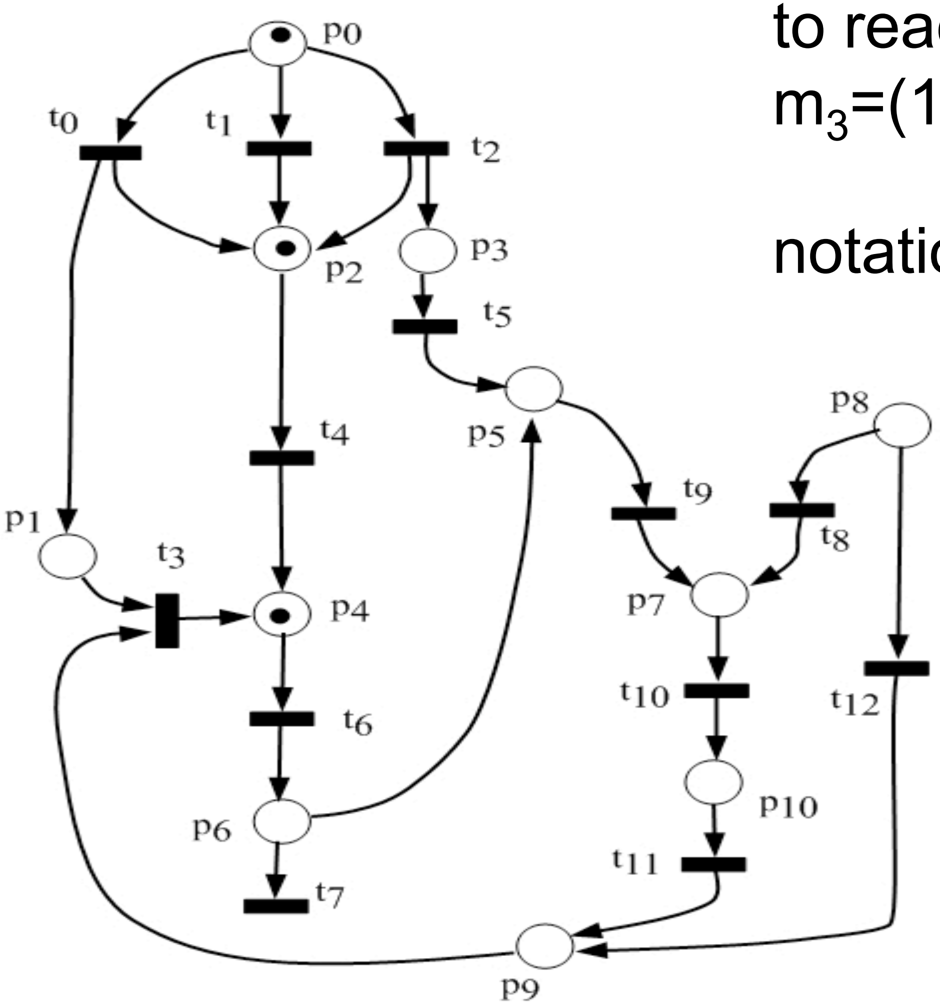
to reach new marking

$m_3=(1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0)$

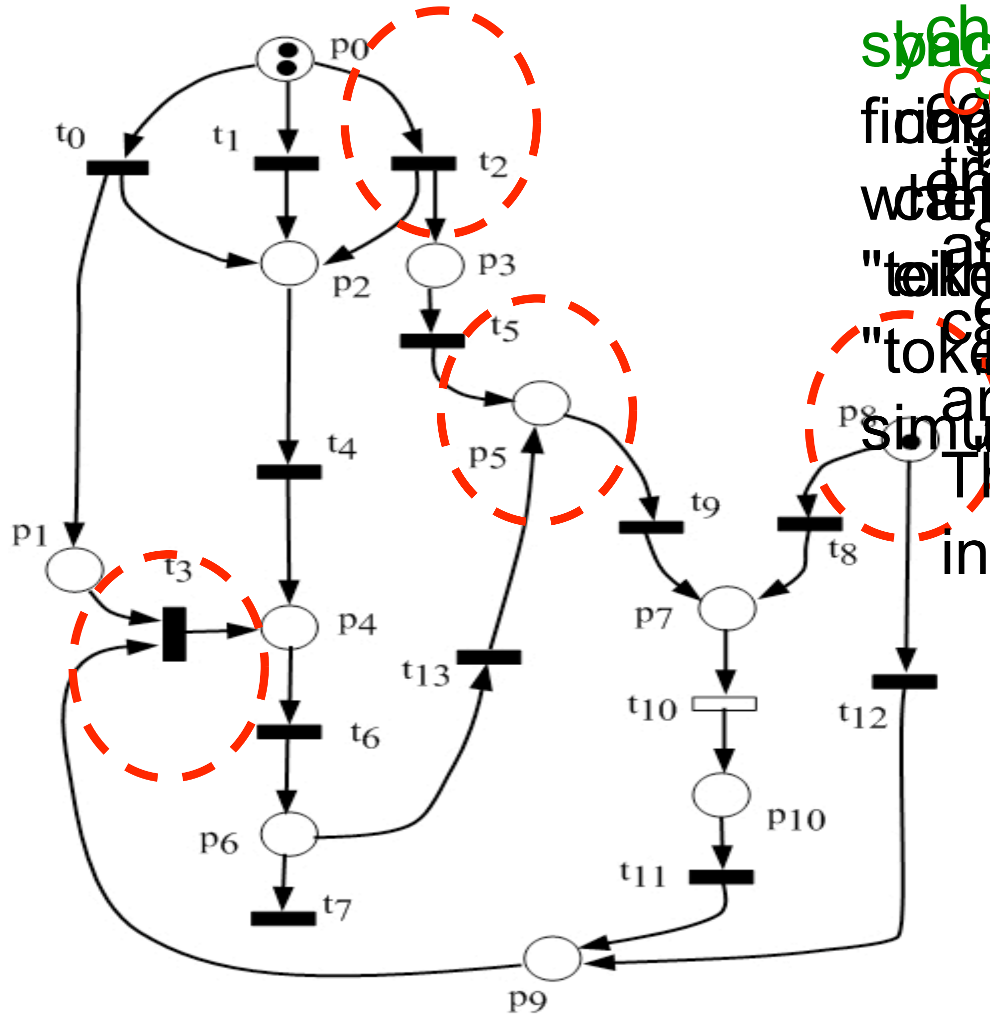
notation: $m_0 \xrightarrow{t_0} m_1 \xrightarrow{t_{12}} m_2 \xrightarrow{t_3} m_3$

initial marking m_0

can generate many traces of events and corresponding sequences of markings



Petri net: modelling power



Concurrency:

Their executions are independent

Petri net: allowable traces, languages and reachable sets

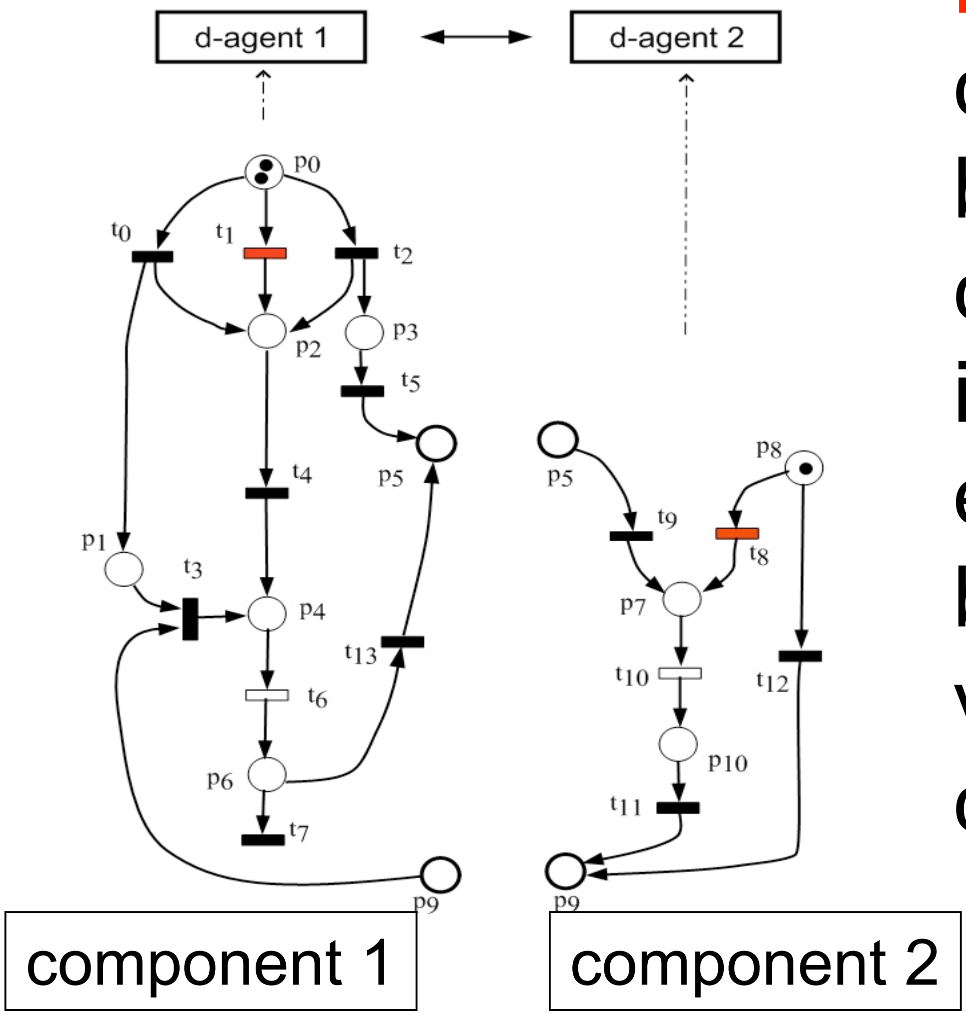
- a trace $\tau = m_0 \xrightarrow{t_1} m_1 \xrightarrow{t_2} m_2 \xrightarrow{t_3} m_3 \xrightarrow{t_4} \ldots\ldots \xrightarrow{t_{N-1}} m_N$ is allowable if the firing condition of the successive events is always satisfied
- The language $\mathcal{L}_{\mathbb{N}}(m_0)$ generated by the Petri net $\mathbb{N}$ with initial marking m_0 is the set of all allowable traces τ

Petri net: allowable traces, languages and reachable sets

- The language $\mathcal{L}_{\mathbb{N}}(m_0)$ generated by the Petri net $\mathbb{N}$ with intial marking m_0 is the set of all allowable traces (depending on the context markings are included in language, or only sequence of events is considered)

- The reachable set $\Re_{\mathbb{N}}(m_0)$ is the set of all markings included $\mathcal{L}_{\mathbb{N}}(m_0)$

Petri net: compositional modelling



Large plants
can be represented
by several Petri net
components,
interacting with
each other
by exchanging tokens
via
common places

Petri net: compositional modelling

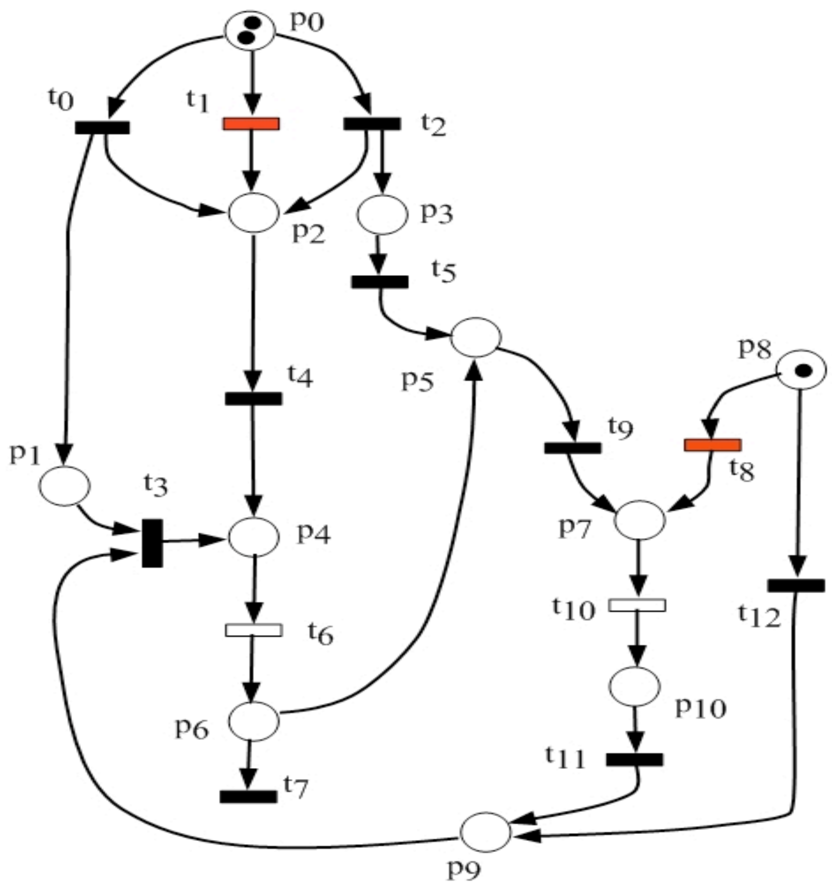
- assumptions
  - set T of transitions partitioned
  - set P of places consists of
    - "local places" in each component i
    - for component i: "input places $P_{IN,i,j}$ that have input transitions (Pre) in component j and output transitions (Post) in component i
    - for component i: "output places $P_{OUT,i,j}$ that have output transitions (Pre) in component j and input transitions (Post) in component i

Petri net: compositional modelling

To simplify presentation assume overall Petri net bounded, i.e. all reachable markings have bounded number of tokens in each place

Difficulty: this assumption depends on the global structure of the Petri ent, cannot be verified locally!

Petri nets: observable events



Some traces of the events in a plant are undesirable, usually starting with an unobservable "fault event"

Only some of the events executed in plant modelled by Petri net can usually be observed

Fault transitions in Petri net example are unobservable transitions t_1 and t_8

Transitions in Petri net partitioned in unobservable and observable transitions (in this example only execution of t_6 and of t_{10} can be observed)

Petri net: projection operator

- Observed sequences of events are obtained form an allowable sequence τ
- by the projection operator $\prod$:
  - $\prod(\varepsilon) = \varepsilon$ where ε is the empty string
  - $\prod(t) = t$ if $t \in T_o$ *where* T_o is the set of observable transitions
  - $\prod(t) = \varepsilon$ if $t \in T_o$ *where* T_{uo} is the set of unobservable transitions
  - $\prod(s.t) = \prod(s). \prod(t)$ for $s \in \mathcal{L}_{\mathbb{N}}(m_0)$ and $t \in T$

Outline

- Petri net model
- model-based diagnosis:
  - problem formulation in centralised case
  - problem formulation in distributed case
  - minimal explanations

Model based diagnosis

Given Petri net $\mathbb{N}$ with known initial marking m_0, after observing a sequence

$\mathcal{OBS} = t_1^o.t_2^o....t_K^o \in \mathcal{T}_o^*$ of observable events

determine:

the set $\mathcal{E}(\mathcal{OBS})$ of all explanations

$\mathcal{E}(\mathcal{OBS}) = \{$allowable traces $\tau \in \mathcal{L}_{\mathbb{N}}(m_0)$
such that the projection $\prod(\tau) = \mathcal{OBS}\}$

Model based diagnosis

- all traces τ in $\mathcal{E}(\mathcal{OBS})$ explain $\mathcal{OBS}$ in the sense that τ
  – contains the observed events in the proper ordering,
  – and τ contains no other observable events
- Denote by $\mathcal{M}(\mathcal{OBS})$ the set of all markings reachable in Petri net $(\mathbb{N}, m_0)$ by executing a trace τ in $\mathcal{E}(\mathcal{OBS})$

Model based diagnosis

- Denote by $\mathcal{D}(\mathcal{OBS})$ the set of all faults t_f in any trace τ in $\mathcal{E}(\mathcal{OBS})$
- when $\mathcal{D}(\mathcal{OBS})$ is empty the plant is in the normal state,
- when all traces in $\mathcal{D}(\mathcal{OBS})$ contain at least one fault event t_f then the plant is faulty
- when some traces in $\mathcal{D}(\mathcal{OBS})$, but not all of them, contain a fault event t_f then the plant state is uncertain

Model based diagnosis

- assume plant model $\mathbb{N}$ known
- assume initial marking m_0 known
- remember assumption: set of reachable markings of the Petri net is bounded

then it is in principle possible to diagnose the plant via forward reachability analysis:

Model based diagnosis via forward reachability analysis

Calculate (enumerate) all allowable traces $\tau \in \mathcal{L}_{\mathbb{N}}(m_0)$ and their projection $\prod(\tau)$

Preserve all allowable traces s.t. $\prod(\tau) = \mathcal{OBS}$

This evaluates the set $\mathcal{E}(\mathcal{OBS})$ and allows diagnosis

Sampath, Lafortune et al. (1995) developed an efficient algorithm to implement this approach in an on-line fashion (constructing an observer automaton taking observations as inputs)

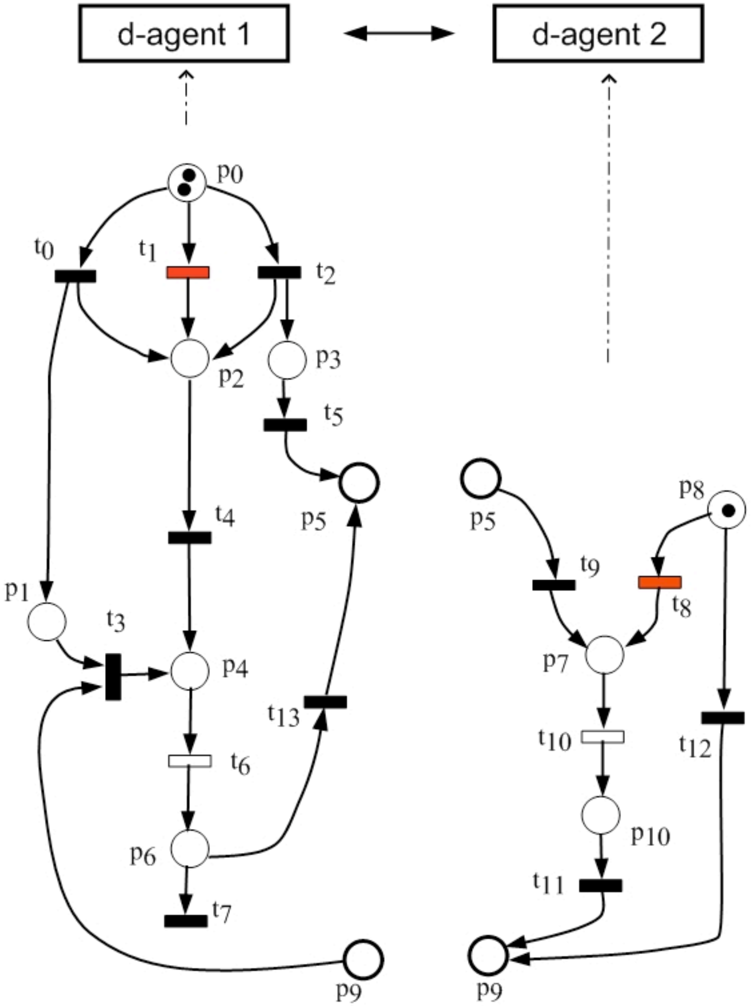
Model based diagnosis via observer automaton

- Recursive (on-line) analysis avoids enumeration of traces that are incompatible with past observation:
- each new observation adds further constraints to set of explanations
- BUT: state space of observer automaton grows in the worst case exponentially in size of Petri net $\mathbb{N}$

Distributed fault diagnosis

- For large plant models observer automaton becomes very large
- For large plants it is often difficult to guarantee that all plant updates (modifications) are always known by central agent
- Sensors communication to central node may be unreliable (lossy link or communication delay)

Distributed fault diagnosis



Each component has one local d-agent

Model plant via interacting Petri net components

•knowing local model only

•knowing what places it has in common with which neighbouring component

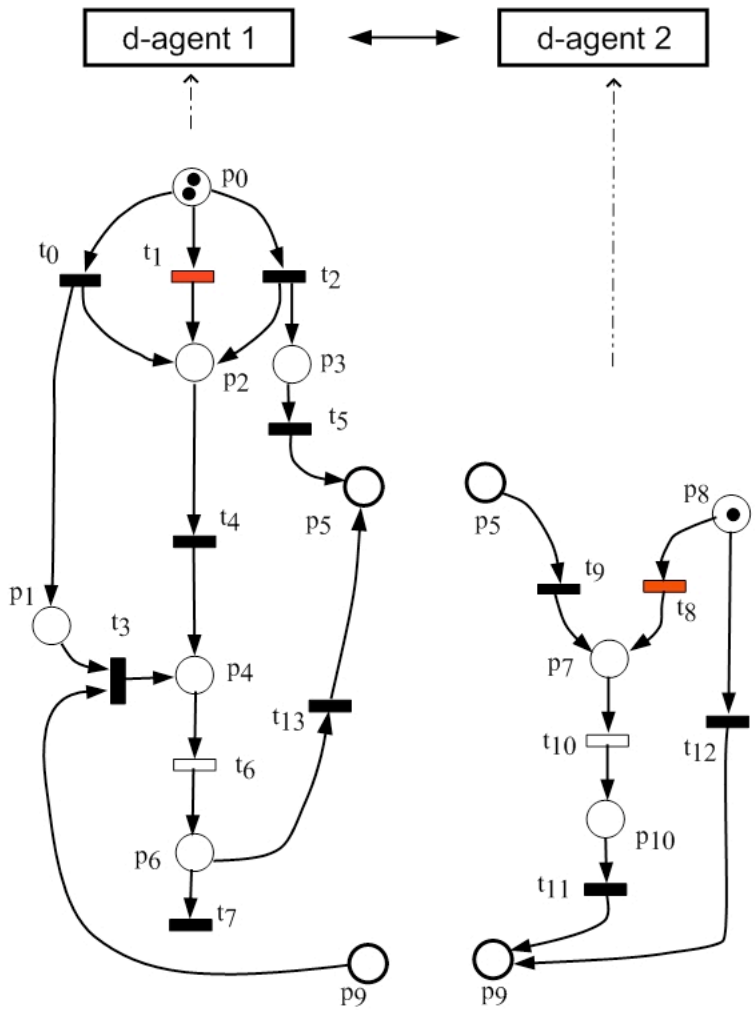
•receiving local sensor output

•exchanging message with neighbouring d-agents

Distributed fault diagnosis

- Forward reachability method not applicable
  - agents do not know the complete initial condition of the local component since they do not know when tokens can arrive from neighbouring agent
  - apart from computational problem of exponential growth of observer automaton

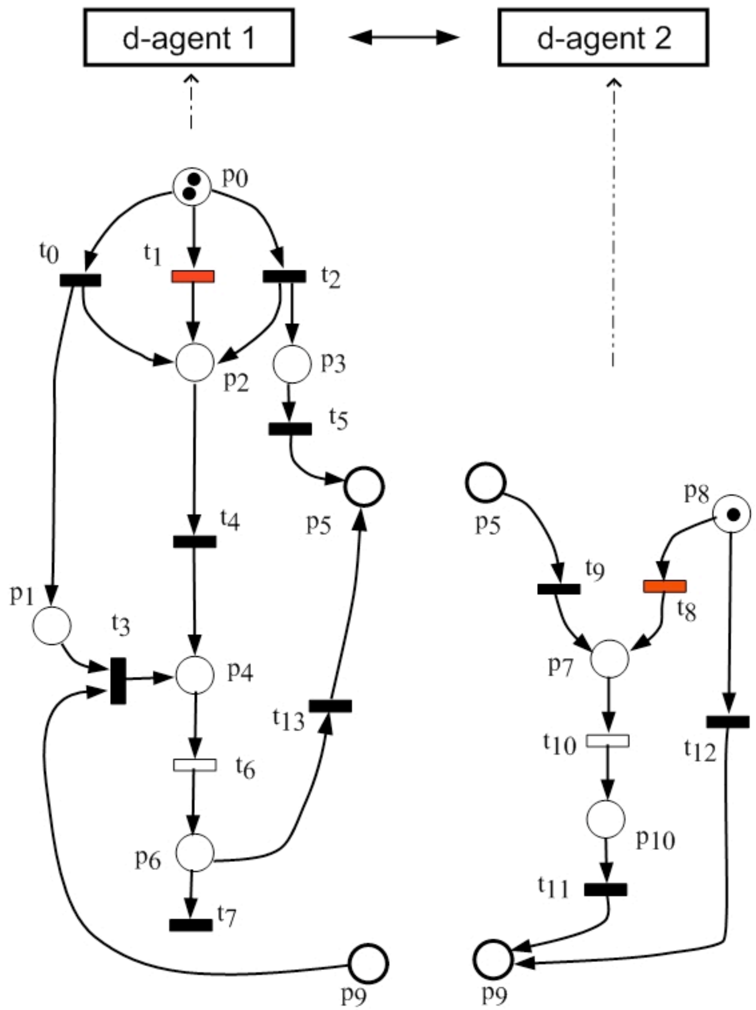
Distributed fault diagnosis



Each local d-agent

- knows local model and local initial condition
- knows places in $P_{IN,i,j}$ where it can unpredictably receive tokens from neighbouring component j

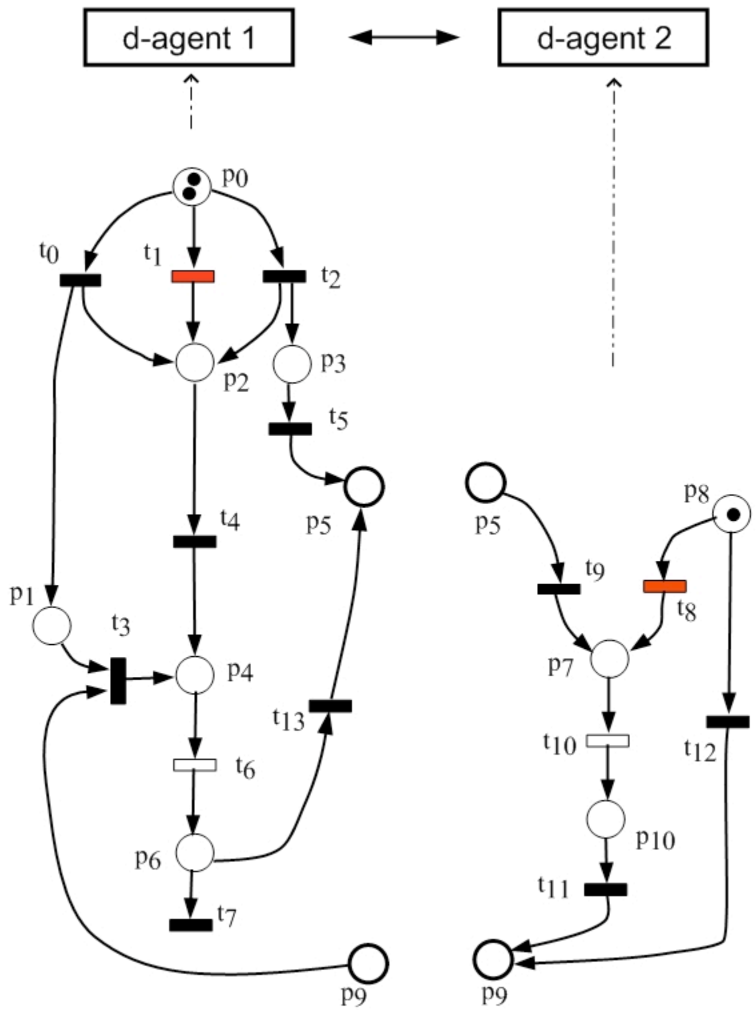
Distributed fault diagnosis



Each local d-agent receives local observations (projection $\mathcal{OBS}_i$ of $\mathcal{OBS}$ onto locally observable tranistions in $T_i \cap T_o$)

Local agent tries to calculate set of local explanations (projections of allowable traces onto local transitions) for $\mathcal{OBS}_i$

Distributed fault diagnosis



Each local d-agent i CANNOT enumerate all local explanations, without a lot of unnecessary calculations since it would have to assume that each input place in $P_{IN,i,j}$ contains a number of tokens specified by the a priori upper bound (which requires global calculation anyway)

Distributed fault diagnosis

- Most previous research on FDI

(Benveniste, Fabre, et al.; Genc and Lafortune; Lamperti et al.; Su and Wonham;...)

- resolves this problem by assuming that tokens entering from a neighbouring component are in some way observable

fault diagnosis

- Here: relax diagnosis goal!
- Enumerate only the minimal traces containing sequences of events that ***must*** have happened for $\mathcal{OBS}$ (or in distributed case $\mathcal{OBS}_i$) to be allowable
- i.e. do not expand traces with transitions that do not lead to satisfaction of constraints necessary for occurrence of observbale event and only use minimal number of tokens

Centralised fault diagnosis

- Set $\mathcal{E}_{Min}(\mathcal{OBS})$ of minimal explanations allows us
  - to decide if a fault happened for sure
  - but if fault t_{f1} is not included in any trace in $\mathcal{E}_{Min}(\mathcal{OBS})$ we do not know whether this is because plant is free of fault t_{f1} or whether fault t_{f1} has occurred but has not yet had any observable effect

Fault detectability

- Fault detection with minimal explanations is reasonable if we assume (as for most FDI papers)
  - that each fault leads to an observable effect after a bounded number of events (no unobservable cycles between fault and observable event), and each choice after a fault event leads to an observation
  - faults cannot be anticipated, i.e. input places of faults are choice places that also lead to a "good" trace

Construction of minimal explanations

- assume $\mathcal{OBS} = \{t_1^o\}$, 1st observation at time $q(t_1^o)$
- necessary constraint for execution of event t_1^o is marking by at least 1 token of each place p_{in}^k in $Pre(t_1^o)$
- this in turn requires that for each place p_{in}^k at least one of the input transitions (determined by $Post(., p_{in}^k)$ of p_{in}^k has fired prior to $q(t_1^o)$

Construction of minimal explanations centralised agent

- recursive enumeration alternatingly using Pre and Post ends when backward path reaches a place that contains enough tokens according to the initial marking
- if there are unobservable cycles then it is possible that on a first visit to an intially marked place there are not enough tokens, but that repeating cycle several times may provide enough tokens

Construction of minimal explanations

- BUT: if no unobservable cycles with choice places exist, then number of tokens in cycle cannot unobservably change
- this ensures validity of stopping criterion: stop when enough tokens are available; remove trace form set of possible explanations if place is visited for 2nd time and still not enough tokens are found
- Each minimal explanation consists of a trace that CANNOT contain any observable event

Construction of minimal explanations

- remark: assumption no unobservable cycles with choice places, ensures that no problems due to tokens moving unobservably from one cycles containing several initially marked places

Construction of set of minimal explanations

- How large is set $\mathcal{E}_{Min}(\mathcal{OBS})$ of minimal explanations?
- At each backward choice place each possible input transition must be explored: size exponential in #backward choice places

Construction of set of minimal explanations

- if Petri net does not have unobservable cycles then one can prove that result is independent of order in which different explanations are explored
- Hence: result can be proven correct and complete (all obtained traces are minimal explanations and all minimal explanations are found)

Recursive calculation of set of minimal explanations

- After 1st observation $\mathcal{OBS} = \{t_1^o\}$ calculate for each trace $\tau_{exp}^n \in \mathcal{E}_{Min}(\mathcal{OBS})$
  the marking $m_1(\tau_{exp}^n)$ reached by executing τ_{exp}^n starting from m_0
- For each τ_{exp}^n use marking $m_1(\tau_{exp}^n)$ as the new initial marking for calculating the set of minimal explanations when the 2nd observed event t_2^o is detected

Recursive calculation of set of minimal explanations

- if starting from $m_1(\tau_{exp}{}^n)$ the trace $\tau_{exp,2}{}^m$ is a possible minimal explanation of $t_2{}^0$
- the concatenatation $\tau_{exp}{}^n.\ \tau_{exp,2}{}^m$ is a minimal explanation of observation $\mathcal{OBS} = \{t_1{}^o,\ t_2{}^0\}$

Recursive calculation of set of minimal explanations

- the set of all the minimal explanations for

  $\{t_1^o, t_2^0\}$ is $\mathcal{E}_{Min}(\{t_1^o, t_2^0\})$
- each trace $\tau_{exp}^n \in \mathcal{E}_{Min}(\{t_1^o, t_2^0\})$ leads to a new marking $m_2(\tau_{exp}^n)$ to be used as initial marking

  when the 3rd observed event is detected

Construction of minimal explanations distributed case

- Problem in distributed case:
- local calculation: when to stop if backward search path leads to input place in $P_{IN,i,j}$?
- whenever backward search for possible minimal explanation τ_{exp}^{n} leads to input place $p_{in,k} \in P_{IN,i,j}$, stop backward search and add information to τ_{exp}^{n} that component j must have some local explanations that put at least one (additional) token in $p_{in,k}$

Construction of minimal explanations distributed case

- When must token arrive in $p_{in,k}$?
- Prior to occurrence time of t_n^0 that leads to generation of trace ending in $p_{in,k}$
- Implies assumption: all components have perfectly synchronised clocks, and can time stamp all observed event accurately

Global clock!

Some particularities of our model

- Unlike many other distributed anayses (Fabre, Jard, Su,...)
- we assume global clock available
- but each agent only knows local model and interactions with neighbouring models
- justification:
  - GPS timer sufficiently accurate for applications,
  - but many reconfigurations make it difficult for each agent to know global model

Goals of distributed fault detection

- From time to time local agnets should exchange enough information
- so that local diagnosis result in component i detects all the local faults that global diagnoser would detect at same time
- i.e. after communication between agents

  local diagnosis = projection of global diagnosis

Questions to be answered

- when to communicate?
- what is minimal information to be exchanged between agents?
- what assumptions must be made on model and on protocol to make this possible?

Outline

- Petri net model
- model-based diagnosis:
  - problem formulation in centralised case
  - problem formulation in distributed case
  - minimal explanations
- distributed diagnosis algorithm

Distributed diagnosis

- arbitrarily select time q > 0
- for each agent i
  - sense $\mathcal{OBS}_i(q)$ such that all observations in $\mathcal{OBS}_i(q)$ occur prior to q, are local to component i, and all observable events in i prior to q are included
  - enumerate locally in agent i the set of local minimal explanations $\mathcal{E}_{Min}(\mathcal{OBS}_i(q))$
- at time q "stop the clock" and exchange information between all agents until stopping criterion

Distributed diagnosis

"stop the clock" means assumption of

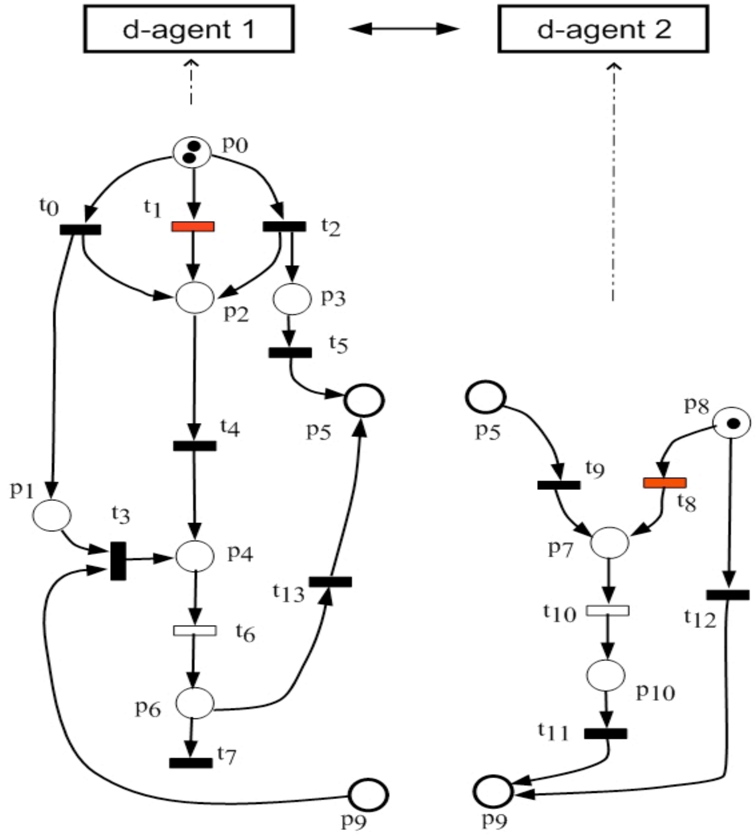
- no communication delays
- infinitely fast processor

Moreover we assume that the global clock is perfect, i.e. all events over all components can be perfectly ordered after exchanging information between agents

choice of q

- since there are no common events between agents, q must be treated as an arbittrary time
- it can be triggered by any event in any component, and will then be accepted by all other agents
- or it may be set by an outside supervisor

local calculation of $\mathcal{E}_{Min}(\mathcal{OBS}_i(q))$



Assume agent 1 knows:

model $\mathcal{N}_1$

initial marking of 2 tokens in p_0

observes $\mathcal{OBS}_1 = \{t_6 \text{ at } q(t_6)\}$

minimal explanations:

$\tau_{exp}^{1,1} = t_0 t_3 t_6$ with constraint token reaches p_9 from $\mathcal{N}_2$ before $q(t_6)$

$\tau_{exp}^{2,1} = t_0 t_4 t_6$

$\tau_{exp}^{3,1} = t_1 t_4 t_6$

$\tau_{exp}^{4,1} = t_2 t_4 t_6$

Assume agent 2 knows:

model $\mathcal{N}_2$

initial marking in p_8

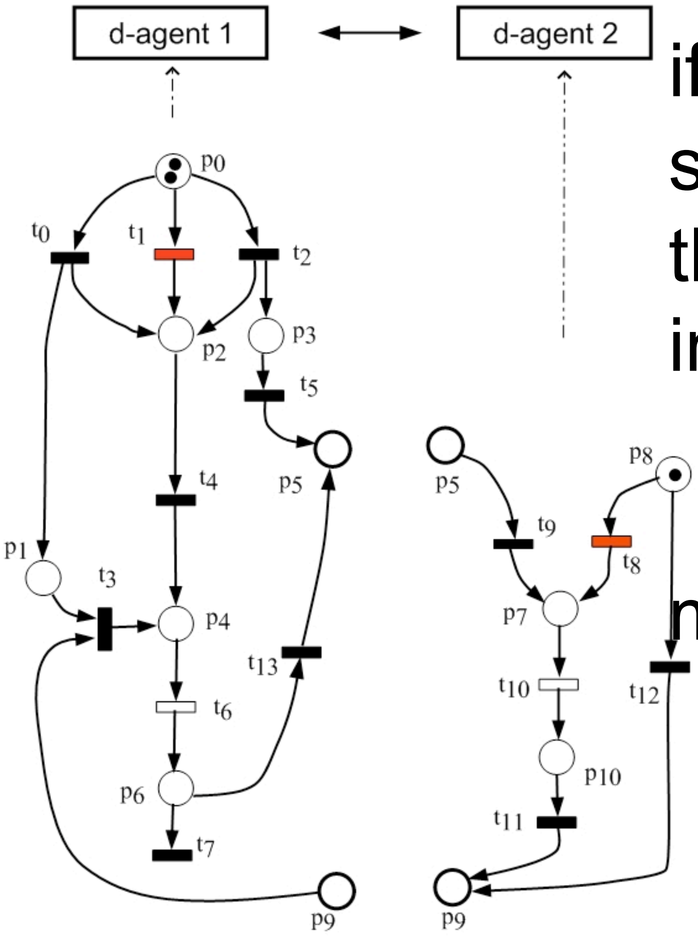
observes $\mathcal{OBS}_2 = \{t_{10} \text{ at } q(t_{10})\}$

minimal explanations:

$\tau_{exp}^{1,2} = t_9 t_{10}$ with constraint token reaches p_5 from $\mathcal{N}_1$ before $q(t_{10})$

$\tau_{exp}^{2,2} = t_8 t_{10}$

reconciling minimal explanations at q when "clock is stopped"

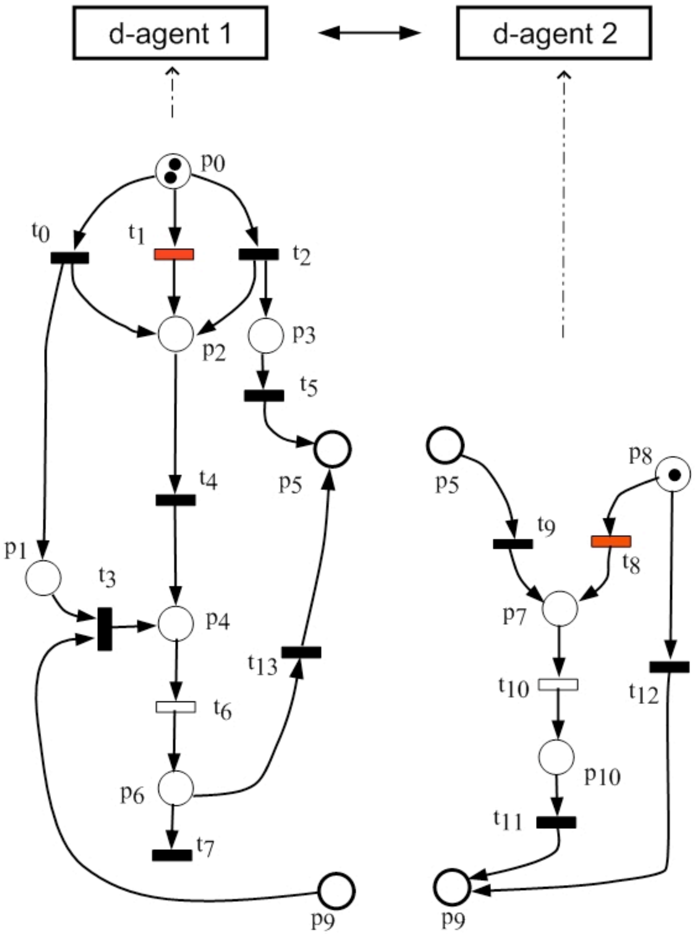


if $q(t_{10}) < q$ and agent 2 has not sensed any other event before q, then agent 2 can initiate information exchange with agent 1:

minimal explanation $\tau_{exp}^{1,2} = t_9.t_{10}$ asks agent 1 which of its minimal explanations can put token in p_5 before $q(t_{10})$

agent 1 calculates that $\tau_{exp}^{4,1}$ can do this whatever the value of $q(t_6)$

reconciling minimal explanations at q when "clock is stopped"

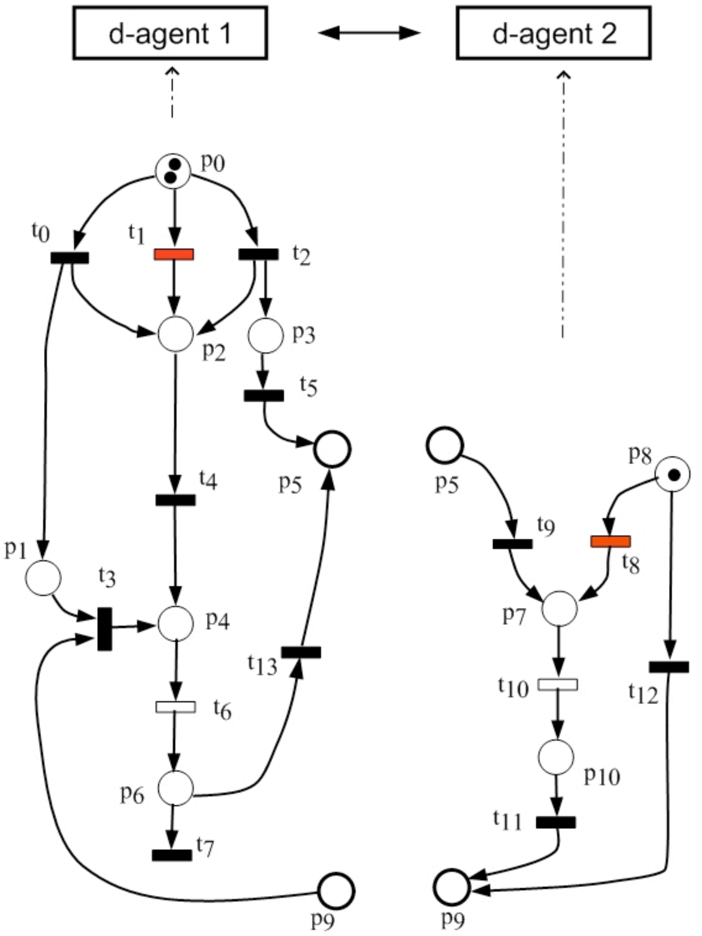


Hence $\tau_{exp}^{1,2} = t_9.t_{10}$ and $\tau_{exp}^{4,1} = t_2t_4t_6$ together form a global minimal explanation without any fault event

$\tau_{exp}^{2,2}$ can be combined with $\tau_{exp}^{2,1}$ forming a global explanation containing fault t_8

$\tau_{exp}^{2,2}$ can be combined with $\tau_{exp}^{3,1}$ forming a global explanation containing both faults t_1 and t_8

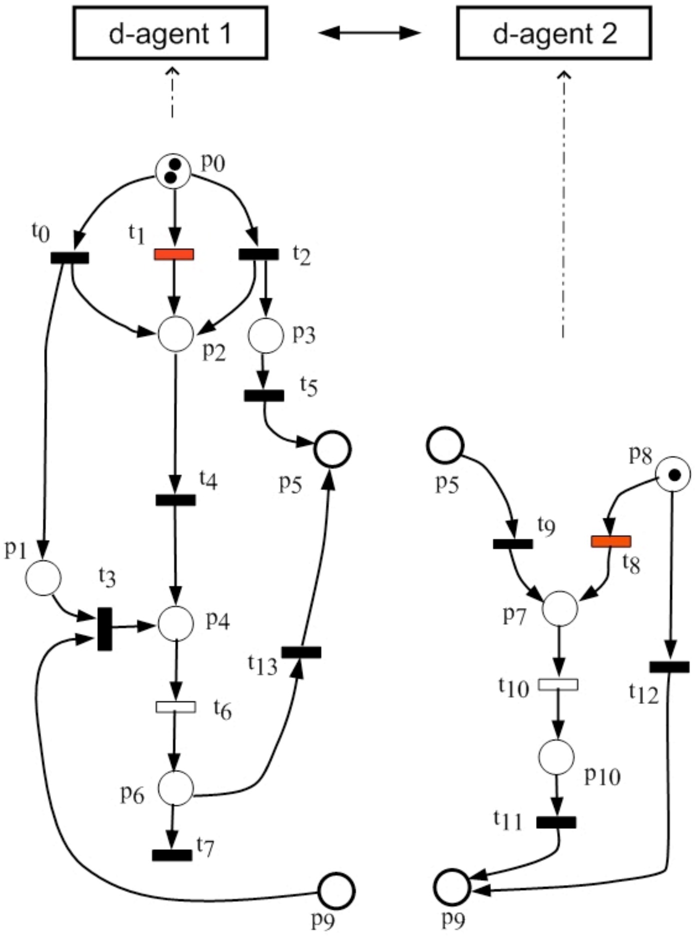
reconciling minimal explanations at q when "clock is stopped"



Moreover agent 1 will respond to the question of agent 2 that it can always put a token into place p_5 via the unobservable trace t_2t_5

This requires agent 1 to do a backward calculation finding a minimal explanation for a token in p_5

reconciling minimal explanations at q when "clock is stopped"



In order to form the complete set of global minimal explanations agent 1 must also ask agent 2 if it can put a token into p_9 before time $q(t_6)$ in order to allow $\tau_{exp}{}^{1,1}$
Agent 2 will respond that this can via the unobservable trace t_{12}, (excluding $\tau_{exp}{}^{1,2}$ but not excluding $\tau_{exp}{}^{2,2}$), and that moreover it can also be combined with $\tau_{exp}{}^{1,2}$ provided $q(t_{10}) < q(t_6)$ ($< q$)

reconciling minimal explanations at q when "clock is stopped"

- In previous example question-and-answer session comes to conclusion as soon as all pairs of local minimal epxlanations have been validated or rejected
- This depends on assumption that no token can pass unobserved from input place to output place (e.g. observable transition t_{10} on path from p_5 to p_9)

reconciling minimal explanations at q when "clock is stopped"

- assumption of at least one observable transition on any path from input to output place in any component can be dropped at price of longer calculations:

  need to use upper bounds on number of tokens in output places to guarantee that iterations eventually stop

reconciling minimal explanations at q when "clock is stopped"

- More than 2 components:
- communication protocol must ensure "fairness" i.e. each agent can only exchange a bounded number of messages with one of its neighbours before starting a round of communication exchange with each of its other neighbours.
- This guarantees that algorithm eventually decides for each possible combination of local explanations whether they are global minimal explanations

reconciling minimal explanations at q when "clock is stopped"

- this "fairness" requirement is analogous to requirements in asynchronous solution of lsets of algebraic equations

  (like asynchronous Gauss-Seidel)

Recursive distributed fault detection

- after completion of communication round at time q
- each local agent can calculate the marking reached after executing the local projection of each of the accepted global minimal explanations
- these marking can be used a initial marking for the next run of the distirbuted fault detection algorithm

Outline

- Petri net model
- model-based diagnosis:
  - problem formulation in centralised case
  - problem formulation in distributed case
  - minimal explanations
- distributed diagnosis algorithm
- conclusion

conclusion

- distributed fault diagnosis can be implemented using local agents
  - knowing only local model and local initial marking
  - sensing local event occurrences (incl. times)
  - calculating recursively local minimal epxlanations

  up to some time q when communication between all agents is initiated by supervisory protocol

  this allows each local agent to decide which local minimal epxlanations are globally valid

extensions

- include event timing information in model (e.g. using time Petri net models for some components) in order to further reduce set of minimal explanations

extensions

- model based fault detection can only detect faults that are explicitly modelled, but often the number of potential faults is so large that this is computationally infeasible
- probabilistic models where most likely models are checked first become attractive

extensions

- both timed models and probabilistic models require combination of forward analysis (using unfoldings and configurations) and backward analysis