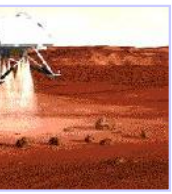
- Fault Aware Systems Through Model-based Programming
- Diagnosis as Detective Work
- Model-based Diagnosis

Brian C. Williams
16.412J/6.834J
March 8th, 2004

Mars Polar Lander Failure



Leading Diagnosis:

- Legs deployed during descent.
- Noise spike on leg sensors latched by software monitors.
- Laser altimeter registers 50ft.
- Begins polling leg monitors to determine touch down.
- Latched noise spike read as touchdown.
- Engine shutdown at ~50ft.

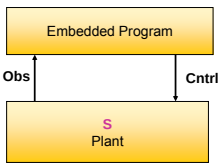
Aware Systems:
bedded languages
on and coordinate
fly from models

Programmers are overwhelmed by the bookkeeping of reasoning about unlikely hidden states

Like Storyboards, Model-based Programs Specify The Evolution of Abstract States

Embedded programs evolve actions by interacting with plant sensors and actuators:

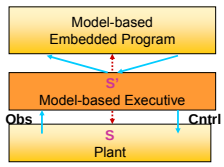
- Read sensors
- Set actuators



Programmer maps between state and sensors/actuators.

Model-based programs evolve abstract states through direct interaction:

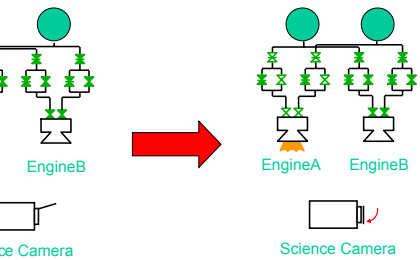
- Read abstract state
- Write abstract state



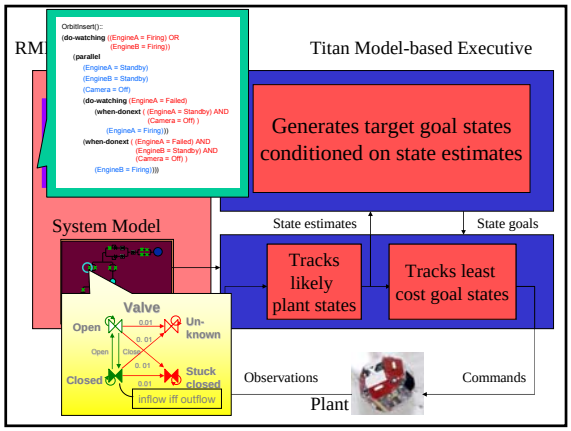
Model-based executive maps between state and sensors/actuators.

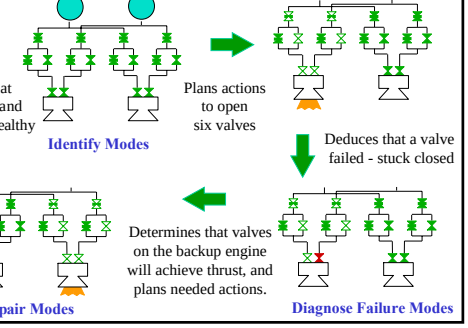
Descent Example

nera off and engine on



RM… Titan Model-based Executive





state trajectories:

- fires one of two engines
- sets both engines to 'standby'
- prior to firing engine, camera must be turned off to avoid plume contamination
- in case of primary engine failure, fire backup engine instead

Plant Model describes behavior of each component:

- Nominal and Off nominal
- qualitative constraints
- likelihoods and costs

```
(do-watching ((EngineA = Thrusting) OR
              (EngineB = Thrusting))
   (parallel
      (EngineA = Standby)
      (EngineB = Standby)
      (Camera = Off)
      (do-watching (EngineA = Failed)
         (when-donext ( (EngineA = Standby) AND
                        (Camera = Off) )
            (EngineA = Thrusting)))
      (when-donext ( (EngineA = Failed) AND
                     (EngineB = Standby) AND
                     (Camera = Off) )
         (EngineB = Thrusting))))
```

Plant Model

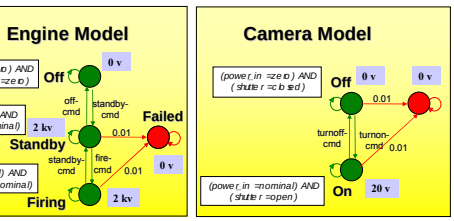
MERS

nt modes…

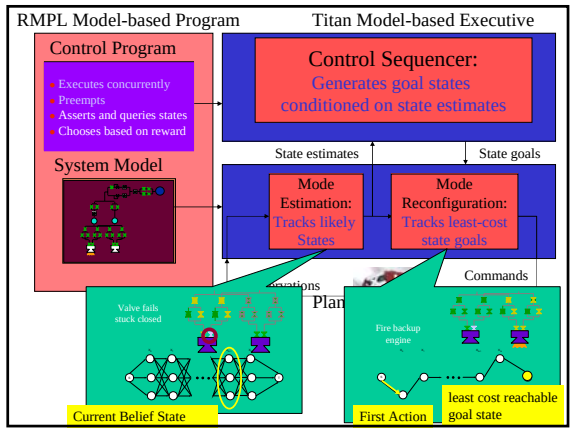
escribed by finite domain constraints on variables…

deterministic and probabilistic transitions

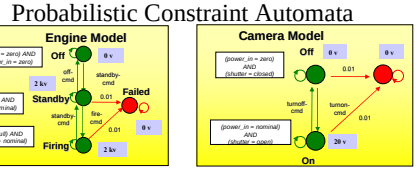
cost/reward



omponent … operating concurrently



Modeling Complex Behaviors through Probabilistic Constraint Automata



x, discrete behaviors

eled through concurrency, hierarchy and timed transitions.

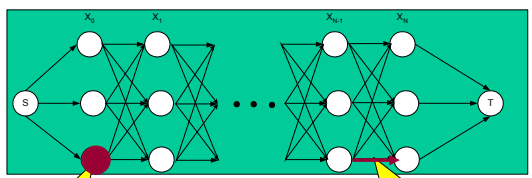
es and uncertainty

eled by probabilistic transitions

interactions

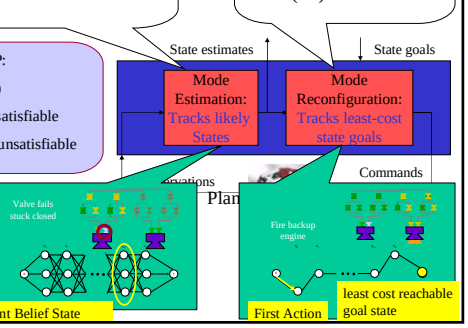
eled by discrete and continuous constraints

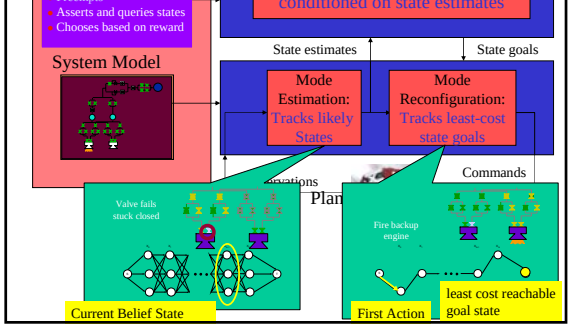
The Plant's Behavior



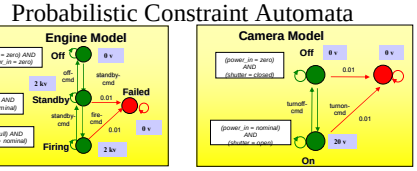
- Assigns a value to each variable (e.g.,3,000 vars).
- Consistent with all state constraints (e.g., 12,000).

- A set of concurrent transitions, one per automata (e.g., 80).
- Previous & Next states consistent with source & target of transitions



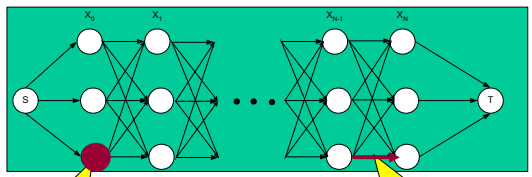


Modeling Complex Behaviors through Probabilistic Constraint Automata



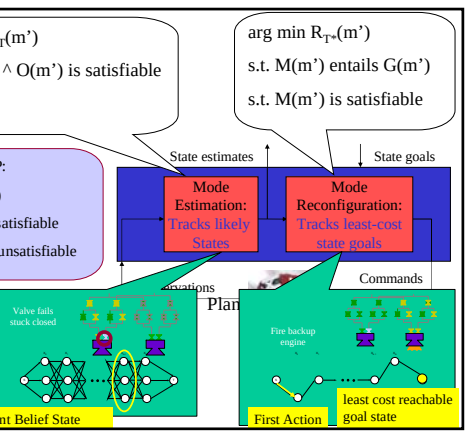
- x, discrete behaviors
- eled through concurrency, hierarchy and timed transitions.
- es and uncertainty
- eled by probabilistic transitions
- interactions
- eled by discrete and continuous constraints

The Plant's Behavior



- Assigns a value to each variable (e.g.,3,000 vars).
- Consistent with all state constraints (e.g., 12,000).

- A set of concurrent transitions, one per automata (e.g., 80).
- Previous & Next states consistent with source & target of transitions



Outline

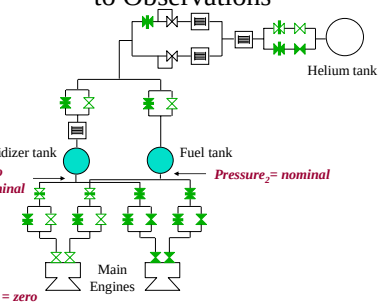
- Fault Aware Systems Through Model-based Programming
- Diagnosis as Detective Work
- Model-based Diagnosis

When you have eliminated the impossible, whatever remains, however improbable, must be the truth.

- Sherlock Holmes. The Sign of the Four.

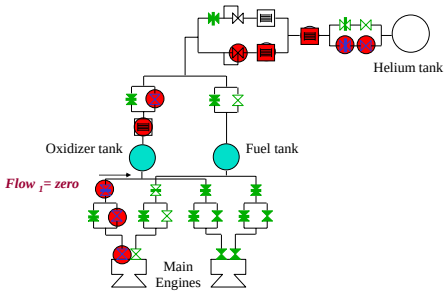
1. Test Hypothesis
2. If Inconsistent, learn reason for inconsistency (a Conflict).
3. Use conflicts to leap over similarly infeasible options to next best hypothesis.

Compare Most Likely Hypothesis to Observations



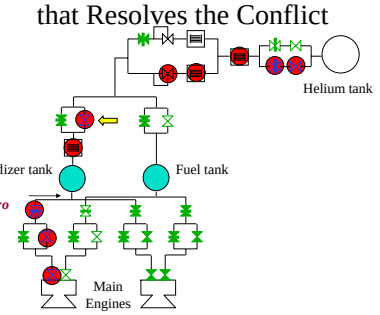
likely that all components are okay.

Isolate Conflicting Information



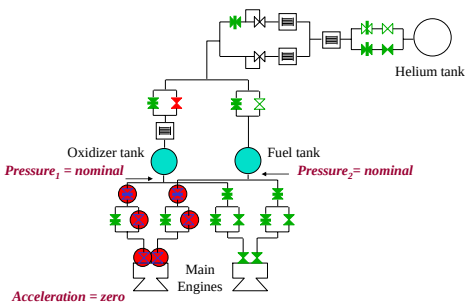
The red component modes *conflict* with the model and observations.

to the Next Most Likely Hypothesis that Resolves the Conflict

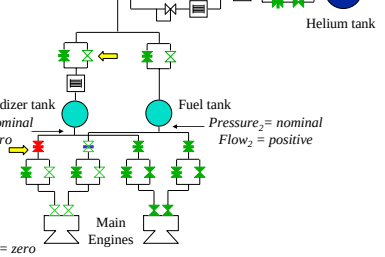


hypothesis must remove the conflict

New Hypothesis Exposes Additional Conflicts



Another conflict, try removing both

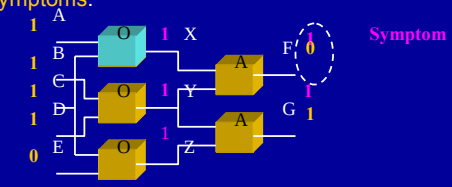


ntation: Conflict-directed A* search.

- Fault Aware Systems Through Model-based Programming
- Diagnosis as Detective Work
- Model-based Diagnosis

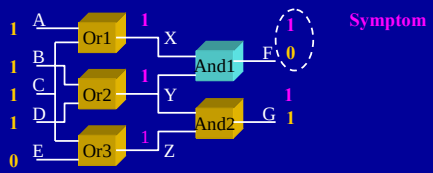
el-based Diagnosis

n a system with symptomatic behavior and a el of the system, find diagnoses that account ymptoms.



Model-based Diagnosis

Given a system with symptomatic behavior and a model of the system, find diagnoses that account for symptoms.

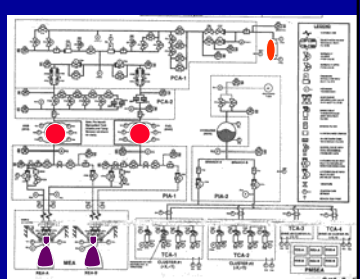


gnosis as othesis Testing

enerate candidates, given symptoms.
st if candidates account for all symptoms.

d Properties:
t of diagnoses should be complete.
t of diagnoses should consider all ailable information.

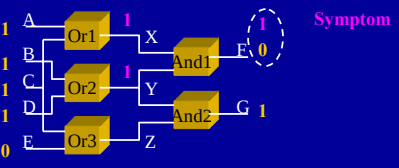
Issue 2: Failures are Often Novel:



Mars Observer: Explosion due to oxidizer/fuel leakage?

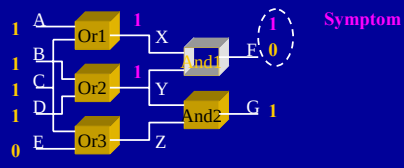
tency-based Diagnosis: Given symptoms, diagnoses that are consistent with symptoms.

nding Constraints: Make no presumptions ut faulty component behavior.



Consistency-based Diagnosis: Given symptoms, find diagnoses that are consistent with symptoms.

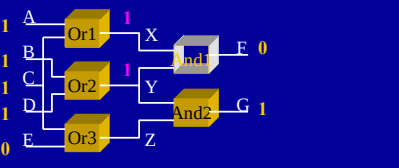
Suspending Constraints: Make no presumptions about faulty component behavior.



e 2: How Should Diagnoses ount for Novel Failures?

tency-based Diagnosis: Given symptoms, diagnoses that are consistent with symptoms.

nding Constraints: Make no presumptions ut faulty component behavior.



Issue 3: Multiple Faults Occur

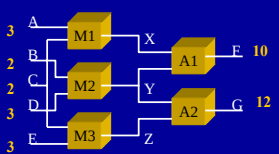
APOLLO 13

- three shorts, tank-line and pressure jacket burst, panel flies off.

➔ Divide & Conquer
- ➔ Diagnose each symptom.
- ➔ Summarize (conflicts)
- ➔ Combine

gnosis identifies sistent modes

i) = In1(i)+In2(i)

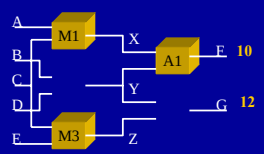


Candidate = {A1=G, A2=G, M1=G, M2=G, M3=G}

late: Assignment to all component modes.

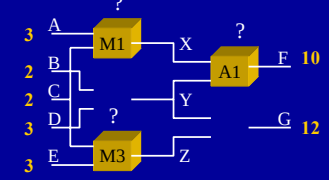
Diagnosis identifies All sets of consistent modes

Adder(i):
- G(i):
  Out(i) = In1(i)+In2(i)
- U(i):



Diagnosis = {A1=G, A2=U, M1=G, M2=U, M3=G}

- Diagnosis D: Candidate consistent with model Phi and observables OBS.
  - As more constraints are relaxed, candidates are more easily satisfied.
  - ➔Typically an exponential number of candidates.



el Diagnosis = {A2=U, M2=U}

llest" sets of modes that remove all symptoms

candidate that is a subset of a kernel diagnosis
iagnosis.

- DPLL Sat algorithm
- Unit propagation (incomplete)

•Finite Domain Constraints
- Backtrack Search w Forward Checking, …
- AC-3/Waltz constraint propagation (incomplete)

• Algebraic Constraints
- Sussman/Steele Constraint Propagation:
  - Propagate newly assigned values through equations mentioning variables.
  - To propagate, use assigned values of constraint to deduce unknown value(s) of constraint.

ng Models In Propositional Logic

= In1(i) AND In2(i)

¬(i=G) ∨ ¬(In1(i)=0) ∨ Out(i)=0
¬(i=G) ∨ ¬(In2(i)=0) ∨ Out(i)=0
¬(i=G) ∨ ¬(In1(i)=1) ∨ ¬(In2(i)=1) ∨ Out(i)=1

= In1(i) OR In2(i)

¬(i=G) ∨ ¬(In1(i)=1) ∨ Out(i)=1
¬(i=G) ∨ ¬(In2(i)=1) ∨ Out(i)=1
¬(i=G) ∨ ¬(In1(i)=0) ∨ ¬(In2(i)=0) ∨ Out(i)=0

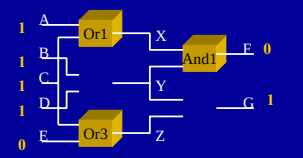
X=1 ∨ X=0
¬X=1∨ ¬X=0

Summary: Consistency-based Diagnosis

- Component Model + Structure:

And(i):
- G(i):
  Out(i) = In1(i) AND In2(i)
- U(i):

ALL components have "unknown Mode" U, Whose assignment is never mentioned in C



Diagnosis = {A1=G, A2=U O1=G, O2=U, O3=G}

- Obs: Assignment to O
- Candidate C_i: Assignment of modes to X
- Diagnosis D_i: A candidate such that $D_i \wedge Obs \wedge C(X,Y)$ is satisfiable.

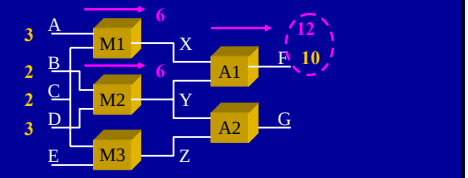
ine

-based Diagnosis
flicts and Kernel Diagnoses
erating Kernels from Conflicts
ling Consistent Modes
mating Likely Modes
flict-directed A*

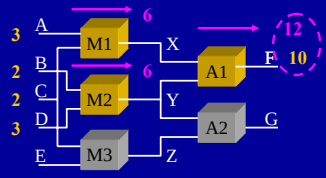
Diagnosis by Divide and Conquer

Given model Phi and observations OBS
- 1. Find all symptoms
- 2. Diagnose each symptom separately
  (each generates a conflict → candidates)
- 3. Merge diagnoses
  (set covering → kernel diagnoses)

General Diagnostic Engine
[de Kleer & Williams, 87]



om:
bserved 10, but should be 12 if A1, M1 & M2 are okay.



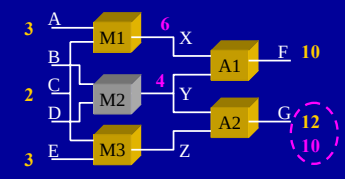
Symptom:
F is observed 10, but should be 12 if A1, M1 & M2 are okay.

Conflict: A1=G & M1=G & M2=G is inconsistent

A1=U or M1=U or M2=U removes conflict.

i.e., at least one is broken

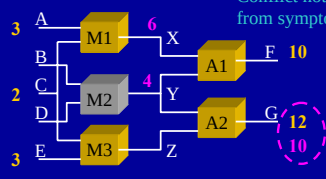
Another Symptom



m:
bserved 12, but should be 10 ...

... and its Conflict

Conflict not just upstream from symptom

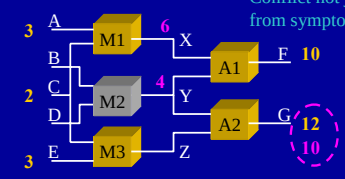


Symptom:
G is observed 12, but should be 10

Conflict: A1=G & M2=G & M1=G & M3=G is inconsistent

nd its Conflict

Conflict not just upstream from symptom

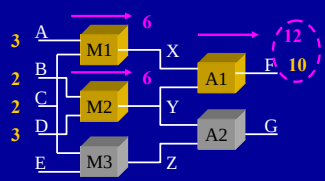


m:
bserved 12, but should be 10

: A1=G & M2=G & M1=G & M3=G is inconsistent

A1=U or A2=U or M1=U or M3=U removes conflict

Summary: Conflicts



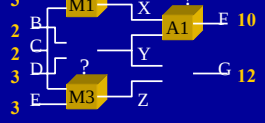
Conflict:
A set of component modes M that are inconsistent with the model and observations.

Properties:

- Every superset of a conflict is a conflict
- Only need conflicts that are minimal under subset
- Logically, not M is an implicate of Model & Obs

el Diagnosis

A2=U & M2=U}



Diagnosis: A set of component modes M all of whose ons are diagnoses.

removes all symptoms

entails Model & Obs *(implicant)*

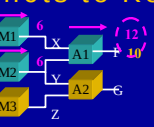
Diagnosis: A minimal partial diagnosis K

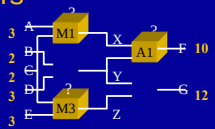
is a prime implicant of model & obs

Model-based Diagnosis

- Conflicts and Kernel Diagnoses
- Generating Kernels from Conflicts
- Finding Consistent Modes
- Estimating Likely Modes
- Conflict-directed A*

noses Found by Mapping licts to Kernels





A set of component modes M that are stent with the model and observations.

M is an implicate of Model & Obs

iagnosis: A minimal set of component modes K that all symptoms.

a prime implicant of Model & Obs

cts map to Kernels by minimal set covering

(see "Characterizing Diagnosis," de Kleer, Reiter, Mackworth)

Generate Kernels From Conflicts

{A1=G, M1=U, M2=U} conflict 1.

{A1=U, A2=U, M1=U, M3=U} conflict 2

A1=U or M1=U or M2=U removes conflict 1.

A1=U or A2=U or M1=U or M3=U removes conflict 2

Kernel Diagnoses =

"Smallest" sets of modes that remove all conflicts

erate Kernels From Conflicts

G, M1=U, M2=U} conflict 1.

U, A2=U, M1=U, M3=U} conflict 2

or M1=U or M2=U removes conflict 1.

or A2=U or M1=U or M3=U removes conflict 2

Diagnoses = {A1=U}

t" sets of modes that remove all conflicts

Generate Kernels From Conflicts

{A1=G, M1=U, M2=U} conflict 1.

{A1=U, A2=U, M1=U, M3=U} conflict 2

A1=U or M1=U or M2=U removes conflict 1.

A1=U or A2=U or M1=U or M3=U removes conflict 2

Kernel Diagnoses = {M1=U}
{A1=U}

"Smallest" sets of modes that remove all conflicts

U, A2=U, M1=U, M3=U} conflict 2

or M1=U or M2=U removes conflict 1.

or A2=U or M1=U or M3=U removes conflict 2

Diagnoses = {A2=U, M2=U}
{M1=U}
{A1=U}

st" sets of modes that remove all conflicts

{A1=U, A2=U, M1=U, M3=U} conflict 2

A1=U or M1=U or M2=U removes conflict 1.

A1=U or A2=U or M1=U or M3=U removes conflict 2

Kernel Diagnoses = {M2=U, M3=U}
{A2=U, M2=U}
{M1=U}
{A1=U}

"Smallest" sets of modes that remove all conflicts

le Fault Diagnoses are the rsection of All Conflicts

G, M1=U, M2=U} conflict 1.

U, A2=U, M1=U, M3=U} conflict 2

or M1=U or M2=U removes conflict 1.

or A2=U or M1=U or M3=U removes conflict 2

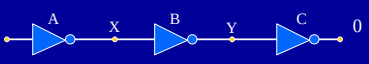
Fault Diagnoses = {A1=U, M1=U}

Outline

Model-based Diagnosis

- Conflicts and Kernel Diagnoses
- Generating Kernels from Conflicts
- Finding Consistent Modes
- Estimating Likely Modes
- Conflict-directed A*

gnosis With Only the nown



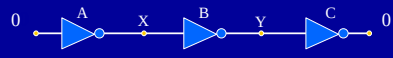
Inverter(i):

- G(i): Out(i) = not(In(i))
- U(i):

• Isolates surprises
• Doesn't explain

ote: Nominal and Unknown Modes

Diagnosis With Only the Known



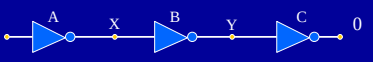
Inverter(i):

- G(i): Out(i) = not(In(i))
- S1(i): Out(i) = 1
- S0(i): Out(i) = 0

• No surprises
• Explains

Exhaustive Fault Modes

[de Kleer & Williams, IJCAI 89]

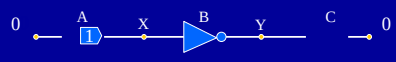


Inverter(i):

- G(i): Out(i) = not(In(i))
- S1(i): Out(i) = 1
- S0(i): Out(i) = 0
- U(i):

- Isolates surprises
- Explains

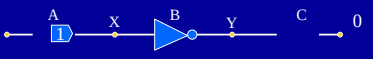
Nominal, Fault and Unknown Modes



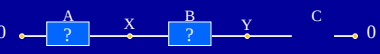
Diagnosis: [S1(A),G(B),U(C)]

mple Diagnoses

Sherlock
[de Kleer & Williams, 89]

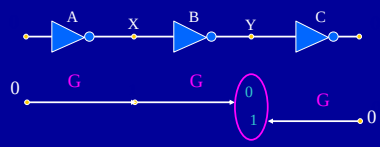


sis: [S1(A),G(B),U(C)]



Diagnosis: [U(C)]

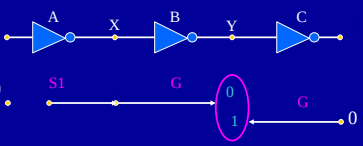
1. Find Symptoms & Conflicts



Conflict:

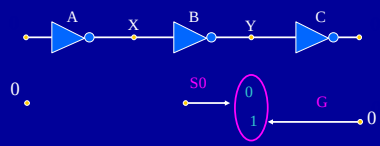
not [G(A), G(B) and G(C)]

e Symptoms & Conflicts

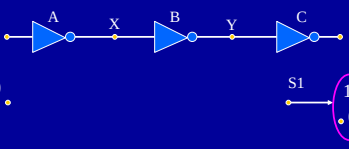


Not [S1(A), G(B), and G(C)]

More Symptoms & Conflicts



not [S0(B) and G(C)]



not S1(C)

- < S1(C) >
- < S0(B), G(C) >
- < S1(A), G(B), G(C) >
- < G(A), G(B), G(C) >

onstituent Diagnoses
n Conflicts

) >
),S0(C) or U(C)
), G(C) >
),S1(B),U(B),S1(C),S0(C) or U(C)
), G(B), G(C) >
),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C) or U(C)
, G(B), G(C) >
A),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C) or U(C)

3. Generate Kernel Diagnoses

- [G(C),S0(C),U(C)]
- [G(B),S1(B),U(B),S1(C),S0(C),U(C)]
- [G(A),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]
- [S1(A),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]

- [U(C)]

enerating Kernel gnoses

,S0(C),U(C)]
,S1(B),U(B),S1(C),S0(C),U(C)]
,S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]
),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]

)]
C)]

3. Generating Kernel Diagnoses

- [G(C),S0(C),U(C)]
- [G(B),S1(B),U(B),S1(C),S0(C),U(C)]
- [G(A),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]
- [S1(A),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]

- [U(C)]
- [S0(C)]
- [U(B),G(C)]

,S0(C),U(C)]
,S1(B),U(B),S1(C),S0(C),U(C)]
,S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]
),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]

- [S1(B),G(C)]

C)]
),G(C)]

- [G(C),S0(C),U(C)]
- [G(B),S1(B),U(B),S1(C),S0(C),U(C)]
- [G(A),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]
- [S1(A),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]

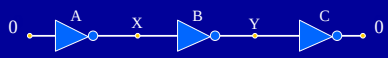
- [U(C)]
- [S0(C)]
- [U(B),G(C]
- [S1(B),G(C)]
- [U(A),G(B),G(C)]

enerate Kernel gnoses

,S0(C),U(C)]
,S1(B),U(B),S1(C),S0(C),U(C)]
,S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]
),S0(A),U(A),S1(B),S0(B),U(B),S1(C),S0(C),U(C)]

C)]
),G(C]

- [S1(B),G(C)]
- [U(A),G(B),G(C)]
- [S0(A),G(B),G(C)]



Diagnoses: (42 of 64 candidates)

Fully Explained Failures
- [G(A),G(B),S0(C)]
- [G(A),S1(B),S0(C)]
- [S0(A),G(B),G(C)]

. . .

Partial Explained
- [G(A),U(B),S0(C)]
- [U(A),S1(B),G(C)]
- [S0(A),U(B),G(C)]

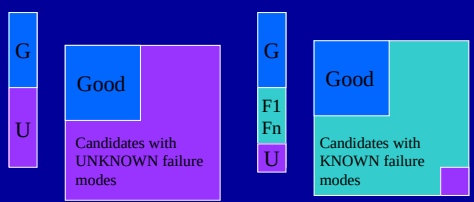
. . .

Fault Isolated, But Unexplained
- [G(A),G(B),U(C)]
- [G(A),U(B),G(C)]
- [U(A),G(B),G(C)]

ine

- -based Diagnosis
- flicts and Kernel Diagnoses
- erating Kernels from Conflicts
- ling Consistent Modes
- mating Likely Modes
- flict-directed A*

Due to the unknown mode, there tends to be an exponential number of diagnoses.



But these diagnoses represent a small fraction of the probability density space.

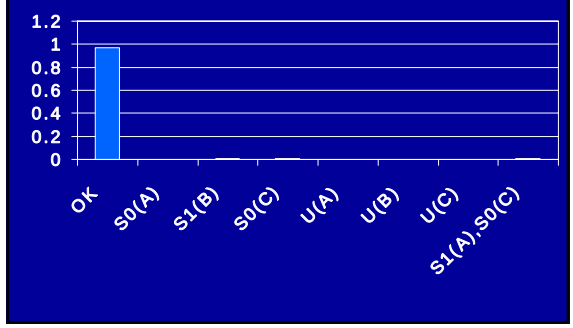
➡ Most of the density space may be represented by enumerating the few most likely diagnoses

$p(c) = \prod_{m \in c} p(m)$

Assume Failure Independence

A	B	C
.99	.99	.99
008	.008	.001
001	.001	.008
001	.001	.001

p([G(A),G(B),G(C)]) = .97
p([S1(A),G(B),G(C)]) = .008
p([S1(A),G(B),S0(C)]) = .00006
p([S1(A),S1(B),S0(C)]) = .0000005



terior Probability, after ervation x = v

$$p(c \mid x = v) = \frac{p(x = v \mid c)p(c)}{p(x = v)}$$

Bayes' Rule

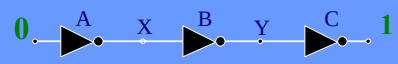
c) estimated using Model:

Normalization Term

vious obs, c and Phi entails x = v
p(x = v | c) = 1

vious obs, c and Phi entails x <> v
p(x = v | c) = 0

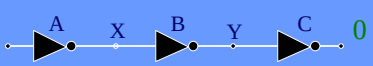
consistent with all values for x
p(x = v | c) is based on priors
.g., uniform prior = 1/m for m possible values of x



Observe out = 1:

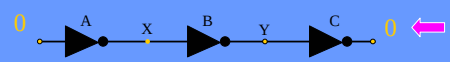
- C = [G(A),G(B),G(C)]
- Prior: P(C) = .97
- P(out = 1 | C) = ?
- = 1
- P(C | out = 0) = ?
- = .97/p(x=v)

$$p(c \mid x = v) = \frac{p(x = v \mid c)p(c)}{p(x = v)}$$



rve out = 0:
[G(A),G(B),G(C)]
C) = .97
ut = 0 | C) = ?
= 0
C | out = 0) = ?
= 0 x .97/p(x=v) = 0

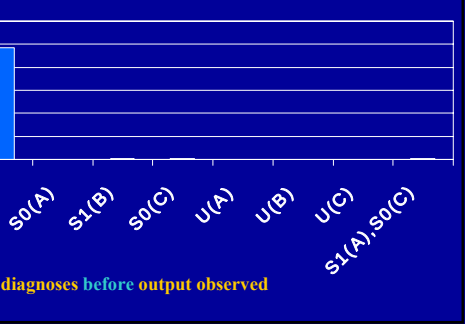
$$p(c \mid x = v) = \frac{p(x = v \mid c)p(c)}{p(x = v)}$$

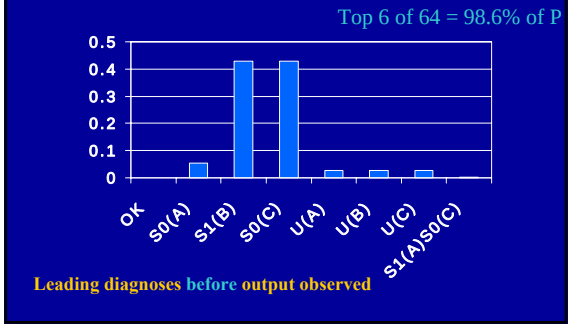


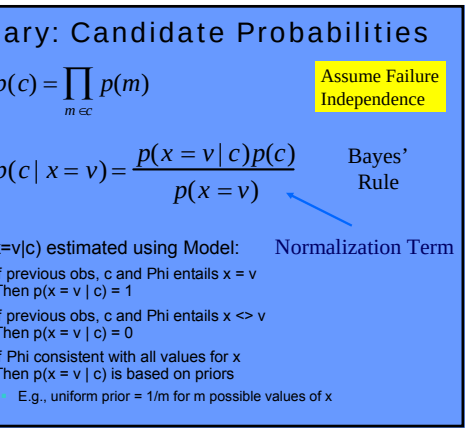
Example: Tracking Single Faults
- which are eliminated?
- which predict observations?
- Which are agnostic?

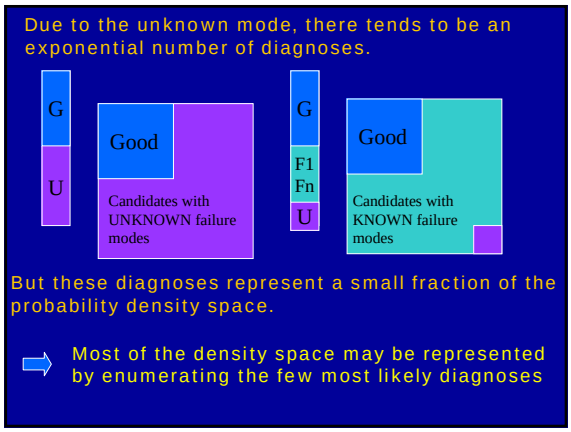
Priors for Single Fault Diagnoses:

	A	B	C
p(S1)	~~.008~~	.008	~~.001~~
p(S0)	.001	~~.001~~	.008
p(U)	.001	.001	.001


S0(A)
S1(B)
S0(C)
U(A)
U(B)
U(C)
S1(A),S0(C)
diagnoses before output observed


Top 6 of 64 = 98.6% of P
0.5
0.4
0.3
0.2
0.1
0
OK
S0(A)
S1(B)
S0(C)
U(A)
U(B)
U(C)
S1(A)S0(C)
Leading diagnoses before output observed


ary: Candidate Probabilities
$p(c) = \prod_{m \in c} p(m)$
Assume Failure Independence
$p(c \mid x = v) = \frac{p(x = v \mid c)p(c)}{p(x = v)}$
Bayes' Rule
Normalization Term
=v|c) estimated using Model:
f previous obs, c and Phi entails x = v
Then p(x = v | c) = 1
f previous obs, c and Phi entails x <> v
Then p(x = v | c) = 0
f Phi consistent with all values for x
Then p(x = v | c) is based on priors
E.g., uniform prior = 1/m for m possible values of x


Due to the unknown mode, there tends to be an exponential number of diagnoses.
G
U
Good
Candidates with UNKNOWN failure modes
G
F1
Fn
U
Good
Candidates with KNOWN failure modes
But these diagnoses represent a small fraction of the probability density space.
Most of the density space may be represented by enumerating the few most likely diagnoses