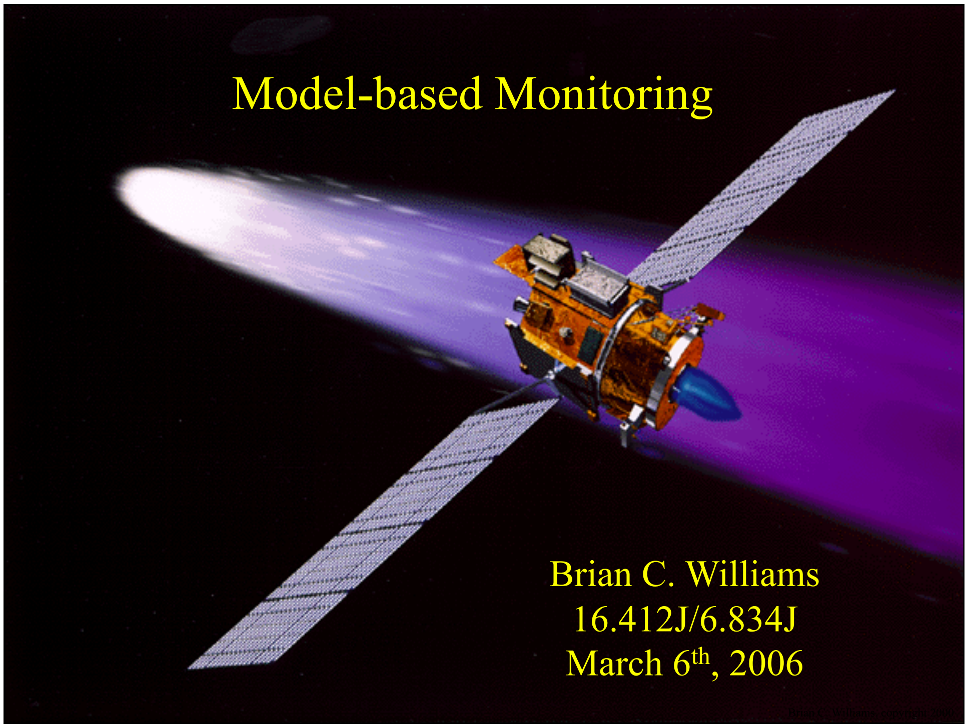
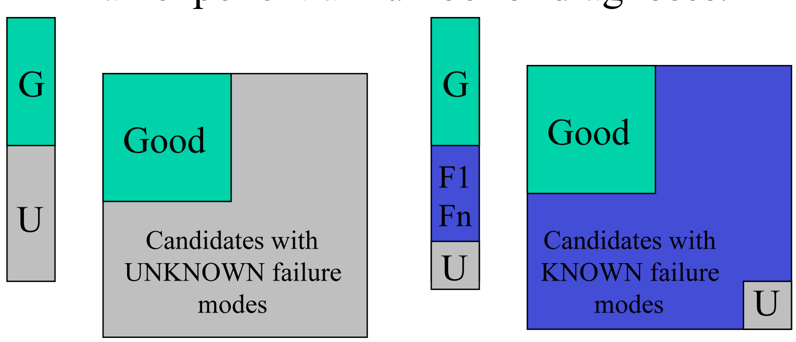


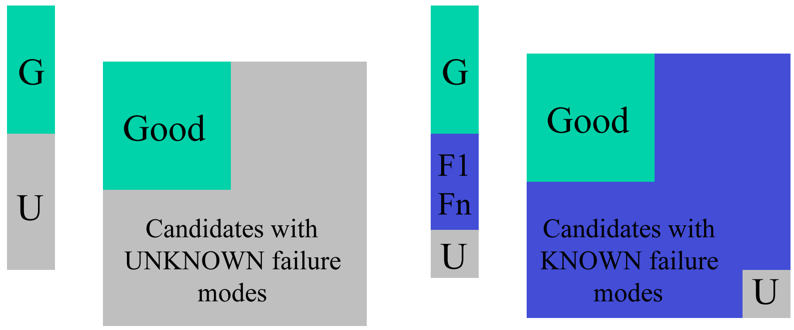
Image credit: NASA.

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



Fault Models don't help.

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



Fault Models don't help.

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

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

Sequential Model-based Diagnosis

Input:

- Set of component mode variables M, with finite domains.
- Set of observables X, with finite domains.
- Device model Φ over M and X, in propositional logic.

- Prior distribution $P(M_i)$ of mode assignments for each component i.
- Observation sequence $X_{1,n} = x_{1,n}$ provided dynamically.

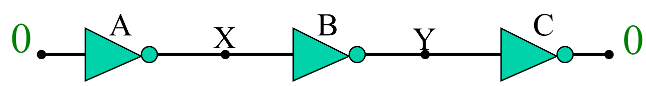
Output:

- P(M) — Prior Probability of Failure
- $P(M \mid X_{1,n} = x_{1,n})$ — Posterior Given Observation
  updated after each observation is received.

Assume:

- Independence of component mode prior distribution.
- Conditional independence of observations given candidate (Naïve Bayes).
- Uniform distribution of observables, given candidate.

Mode Estimation Example



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

Sherlock
[de Kleer & Williams, IJCAI 89]

Candidate (Prior) Initial Probabilities

$$P(M) = P(M_1, M_2, \ldots, M_n)$$
$$= P(m_n \mid m_{n-1} \ldots m_1) \cdots P(m_2 \mid m_1) P(m_1)$$

$$P(M) = \prod_{M_i \in M} P(M_i)$$

Assume Independence Of Initial Mode

	A	B	C
P(G)	.99	.99	.99
P(S1)	.008	.008	.001
P(S0)	.001	.001	.008
P(U)	.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

Posterior Probability, after Observations $X_{1,n} = x_{1,n}$

$$P(M \mid x_1) = \frac{P(x_1 \mid M)P(M)}{P(x_1)}$$

Bayes' Rule

$$= \in P(x_1 \mid M)P(M)$$

For n > 1:

$$P(M \mid x_{1,n}) = \frac{P(x_n \mid M, x_{1,n-1})P(M \mid x_{1,n-1})}{P(x_n \mid x_{1,n-1})}$$

Observations are conditionally independent

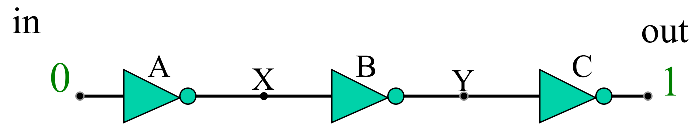
$$= \in P(x_n \mid M)P(M \mid x_{1,n-1})$$

Estimating the Observation Probability $P(x_i \mid M)$

Assumption: All consistent observations for X_i are equally likely

$P(x_i \mid M)$ is estimated using model, Φ, according to:

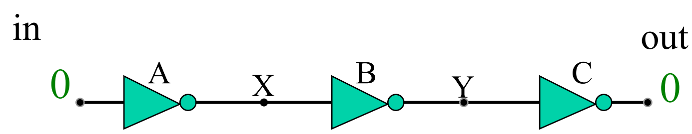
- **If** previous observations $X_{1,i-1} = x_{1,i-1}$, M and Φ entails $X_i = x_i$
  Then $P(x_i \mid M) = 1$
- **If** previous observations $X_{1,i-1} = x_{1,i-1}$, M and Φ entails $X_i \neq v_i$
  Then $P(x_i \mid c) = 0$
- **Otherwise**, Assume all consistent assignments to X_i are equally likely observations:
  let $D_{ci} \equiv \{x_c \in D_{Xi} \mid c, \Phi \text{ is consistent with } X_i = x_c\}$
  Then $P(x_i \mid M) = 1/|D_{ci}|$



$PM(|x)(P(x|)M)pM=x\in$ $_{1,n,n}$ 2

Observe out = 1:

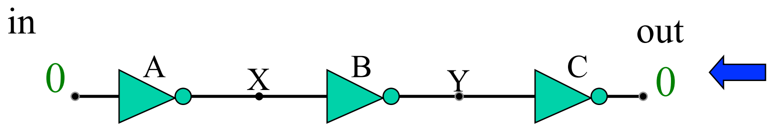
- m = <G(A),G(B),G(C)>
- Prior: P(m) = .97
- P(out = 1 | m) = ?
- = 1
- P(m | out = 0) = ?
- = 1 x .97 x α



$PM(|x)(P(x|)MP)M=x\in$ $_{1,n,n}$ 2

Observe out = 0:

- m = <G(A),G(B),G(C)>
- P(m) = .97
- P(out = 0 | m) = ?
- = 0
- P(m | out = 0) = ?
- = 0 x .97 x α = 0

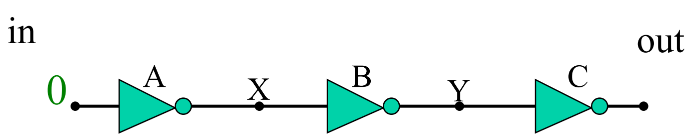


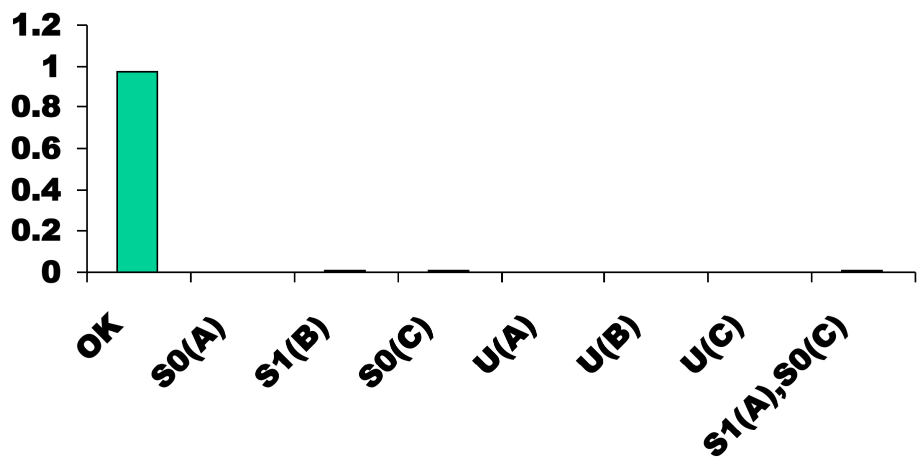
Example: Tracking Single Faults

- Which are eliminated?
- Which are predict observations?
- Which are agnostic?

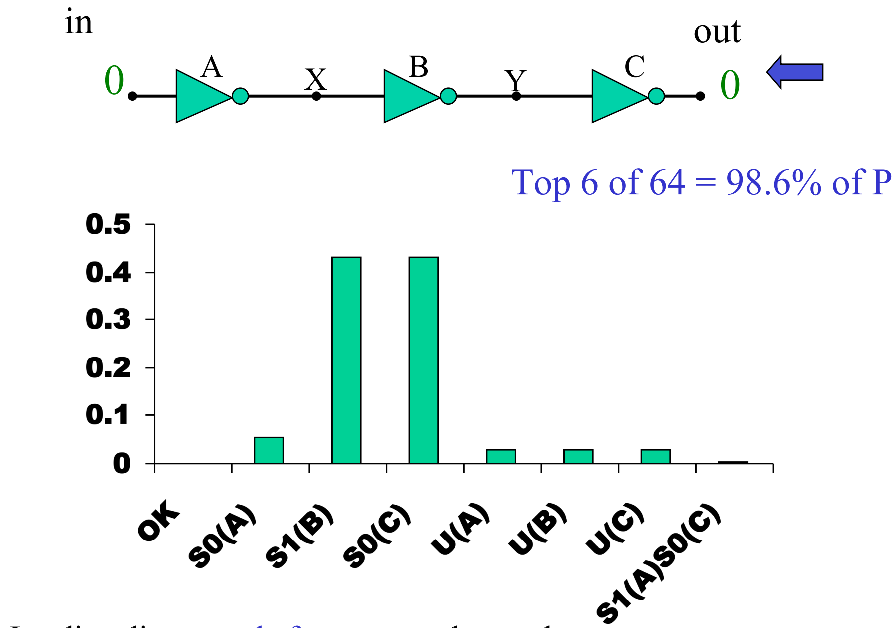
Priors for Single Fault Diagnoses:

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



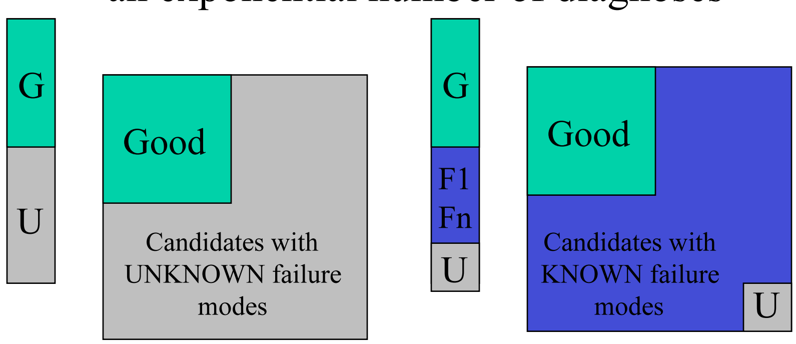


Leading diagnoses before output observed



Leading diagnoses before output observed

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

Optimal CSP

OCSP= <Y, *g*, CSP>

- Decision variables Y with domain D_Y
- Utility function g(Y): $D_Y \rightarrow \Re$
- CSP is over variables <X,Y>

Find Leading $\arg\max_{Y \in D_y} g(Y)$

s.t. $\exists\, X \in D_X$ s.t. C(X,Y) is True

➔ Encode C in propositional state logic

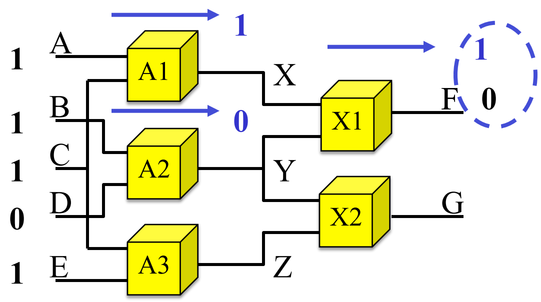
➔ g() is a multi-attribute utility function that is preferentially independent.

Outline

- Self-Repairing Agents
- Formulating Diagnosis
- Diagnosis from Conflicts
- Single Fault Diagnosis
- Extracting Conflicts

11/02/09 18

Extracting Conflicts using Unit Propagation

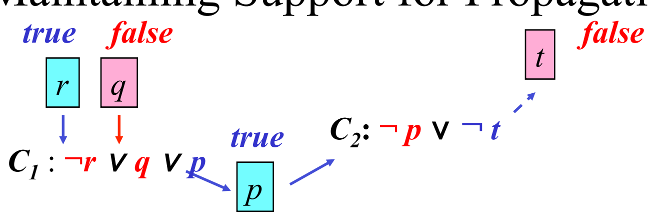


Symptom:
F is observed 0, but should be **1** if A1, A2 and X1 are okay.

Conflict: {A1=G, A2=G, X1=G} is inconsistent.

→ At least A1=U, A2=U or X1=U

Find Symptom Using Unit Propagation while Maintaining Support for Propagation



procedure *propagate*(***C***)

Input: ***C*** is a clause

if all literals in ***C*** are false except **l**, and **l** is unassigned

then assign true to **l** and

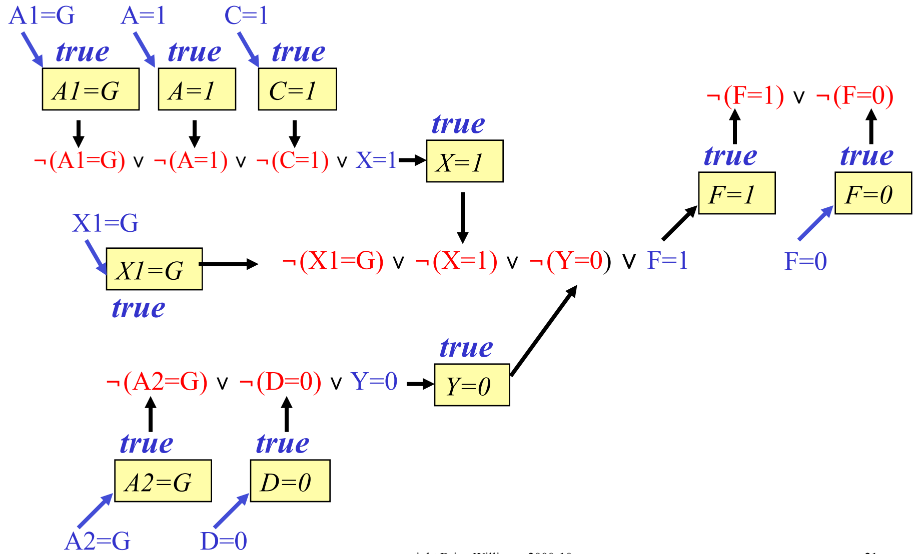
record **C** as a support for **l** and

for each clause **C'** mentioning "not **l**",

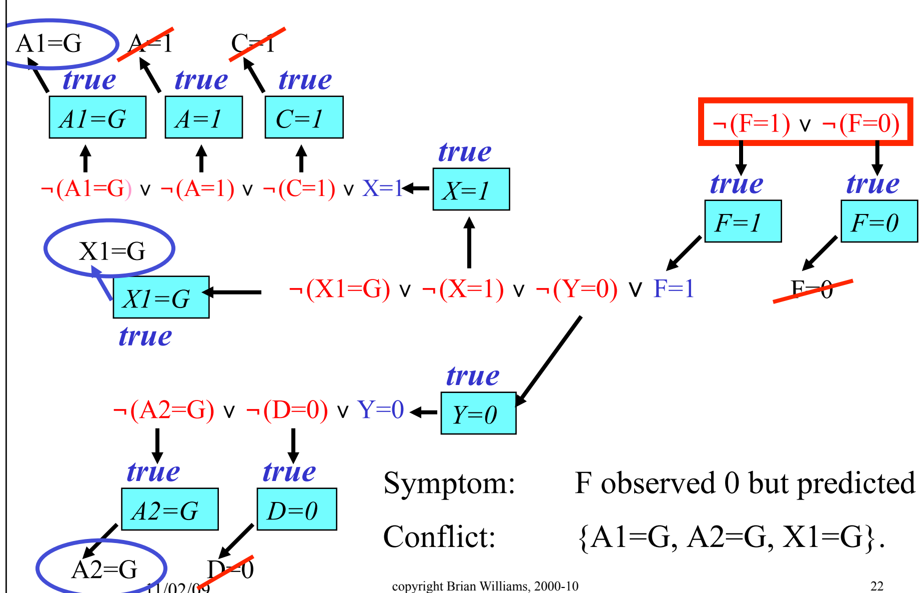
propagate(**C'**)

end *propagate*

Find Symptom Using Unit Propagation while Maintaining Support for Propagation



Extract Conflict by Tracing Support



Symptom: F observed 0 but predicted 1.

Conflict: {A1=G, A2=G, X1=G}.

Extract Conflict by Tracing Support

procedure Conflict(*C*)
Input: **an inconsistent clause C.**
Output: A conflict of C.

for each literal I in C
union Support-Conflict(l, support(l))
end *Conflict*

procedure Support-Conflict(l, S)
Input: **l is a literal and S is the support clause of l.**
Output: A set of mode assignments supporting l.

If unit-clause?(C)
If mode-assignment?(literal(C))
Then {literal(C)}
Else {}
Else for each literal I1 in C, other than l
Union Support-Conflict(l1, support(l1))
end *Support-Conflict*

procedure *Test_Candidate(c, M, obs)*

Input: Candidate c, Model M, Observation Obs.
Output: Consistent or a conflict.

Assert candidate assignment c;
Propagate obs through model M using unit propagation;
If propagate results in an inconsistent clause
Return Conflict(c);
Else
Search for satisfying solution using DPLL;
If inconsistent
Return c as a conflict;
Else
Return consistent;
End Test_Candidate