Outline

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

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

Obs: Assignment to O
Candidate Ci: Assignment of modes to X
Diagnosis Di: A candidate such that
Di ∧ Obs ∧ C(X,Y) is satisfiable.

As more constraints are relaxed, candidates are more easily satisfied.
Typically an exponential number of candidates.

Diagnosis identifies All sets of consistent modes

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

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.
Representing Diagnoses Compactly: Kernel Diagnoses

Kernel Diagnosis = \{A_2=U, M_2=U\}

“Smallest” sets of modes that remove all symptoms

Every candidate that is a subset of a kernel diagnosis is a diagnosis.

Diagnosis by Divide and Conquer

Given model \(\Phi\) and observations \(OBS\)

1. Find all symptoms
2. Diagnose each symptom separately (each generates a conflict \(\rightarrow\) candidates)
3. Merge diagnoses (set covering \(\rightarrow\) kernel diagnoses)

General Diagnostic Engine
[de Kleer & Williams, 87]
Conflicts Explain How To Remove Symptoms

Symptom:
F is observed 0, but should be 1 if O1, O2 and A1 are okay.

Conflict: {A1=G, O1=G, O2=G} is inconsistent
→ At least A1=U, O1=U, or O2=U

Conflicts Explain How To Remove Symptoms

Symptom:
F is observed 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
Find Another Symptom

Symptom:
G is observed 12, but should be 10 ...

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

... and its Conflict

Symptom:
G is observed 12, but should be 10

Conflict: A1=G & M2=G & M1=G & M3=G is inconsistent
... and its Conflict

Symptom:
G is observed 12, but should be 10

Conflict: A1=G & M2=G & M1=G & M3=G is inconsistent
A1=U or A2=U or M1=U or M3=U removes conflict

Conflict not just upstream from symptom

Recap: Conflicts

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

Kernel Diagnosis

\[ \{A_2 = U & M_2 = U\} \]

Partial Diagnosis: A set of component modes \( M \) all of whose extensions are diagnoses.

- \( M \) removes all symptoms
- \( M \) entails Model & Obs \( (\text{implicant}) \)

Kernel Diagnosis: A minimal partial diagnosis \( K \)

- \( M \) is a prime implicant of model & obs

Outline

- Conflicts and Kernel Diagnoses
- Generating Kernels from Conflicts
- Finding Consistent Modes
- Estimating Likely Modes
- Conflict-directed A*
Conflict: A set of component modes $M$ that are **inconsistent** with the model and observations.

- not $M$ is an implicate of Model & Obs

Kernel Diagnosis: A minimal set of component modes $K$ that eliminate all symptoms.

- $M$ is a prime implicant of Model & Obs

Conflicts map to Kernels by minimal set covering

---

**Generate Kernels From Conflicts**

- $\{A_1=G, M_1=U, M_2=U\}$ conflict 1.
- $\{A_1=U, A_2=U, M_1=U, M_3=U\}$ conflict 2
- $A_1=U$ or $M_1=U$ or $M_2=U$ removes conflict 1.
- $A_1=U$ or $A_2=U$ or $M_1=U$ or $M_3=U$ removes conflict 2

Kernel Diagnoses =

“Smallest” sets of modes that remove all conflicts
Generate Kernels From Conflicts

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 = {A1=U}

“Smallest” sets of modes that remove all conflicts
Generate Kernels From Conflicts

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

“Smallest” sets of modes that remove all conflicts
Outline

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

Diagnosis With Only the Unknown

Inverter(i):
- $G(i): \text{Out}(i) = \text{not}(\text{In}(i))$
- $U(i):$
  - Isolates surprises
  - Doesn’t explain

Nominal and Unknown Modes
Diagnosis With Only the Known

Inverter(i):
- \( G(i) \): \( \text{Out}(i) = \neg \text{In}(i) \)
- \( S1(i) \): \( \text{Out}(i) = 1 \)
- \( S0(i) \): \( \text{Out}(i) = 0 \)

Exhaustive Fault Modes
- No surprises
- Explains

Solution: Diagnosis as Estimating Behavior Modes

Inverter(i):
- \( G(i) \): \( \text{Out}(i) = \neg \text{In}(i) \)
- \( S1(i) \): \( \text{Out}(i) = 1 \)
- \( S0(i) \): \( \text{Out}(i) = 0 \)
- \( U(i) \): \( \text{Isolates surprises} \)
- Explains

Nominal, Fault and Unknown Modes
Diagnosis: \([S1(A), G(B), U(C)]\)

Kernel Diagnosis: \([U(C)]\)
1. Find Symptoms & Conflicts

Conflict:
not $G(A)$, $G(B)$ and $G(C)$

More Symptoms & Conflicts

Not $S1(A)$, $G(B)$, and $G(C)$
More Symptoms & Conflicts

not S0(B) and G(C)

not S1(C)
All Conflicts

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

2. Constituent Diagnoses from Conflicts

- < S1(C) >
  => G(C), S0(C) or U(C)
- < S0(B), G(C) >
  => G(B), S1(B), U(B), S1(C), S0(C) or U(C)
- < S1(A), G(B), G(C) >
  => G(A), S0(A), U(A), S1(B), S0(B), U(B), S1(C), S0(C) or U(C)
- < G(A), G(B), G(C) >
  => S1(A), S0(A), U(A), S1(B), S0(B), U(B), S1(C), S0(C) or U(C)
3. Generate Kernel Diagnoses

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

\[\text{downarrow}\]

- \([U(C)]\)

---

3. Generating Kernel Diagnoses

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

\[\text{downarrow}\]

- \([U(C)]\)
- \([S0(C)]\)
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)]
3. Generating Kernel Diagnoses

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

3. Generate Kernel Diagnoses

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

Outline

- Conflicts and Kernel Diagnoses
- Generating Kernels from Conflicts
- Finding Consistent Modes
- Estimating Likely Modes
- Conflict-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.

**Candidate Initial (prior) Probabilities**

\[ p(c) = \prod_{m \in c} p(m) \]

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>p(G)</td>
<td>.99</td>
<td>.99</td>
<td>.99</td>
</tr>
<tr>
<td>p(S1)</td>
<td>.008</td>
<td>.008</td>
<td>.001</td>
</tr>
<tr>
<td>p(S0)</td>
<td>.001</td>
<td>.001</td>
<td>.008</td>
</tr>
<tr>
<td>p(U)</td>
<td>.001</td>
<td>.001</td>
<td>.001</td>
</tr>
</tbody>
</table>

Assume Failure Independence

\[ p([G(A),G(B),G(C)]) = .97 \]
\[ p([S1(A),G(B),G(C)]) = .008 \]
\[ p([S1(A),G(B),S0(C)]) = .00006 \]
\[ p([S1(A),S1(B),S0(C)]) = .0000005 \]
Posterior Probability, after Observation $x = v$

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

Bayes’ Rule

$P(x=v\mid c)$ estimated using Model:

- If previous obs, $c$ and Phi entails $x = v$
  Then $p(x = v \mid c) = 1$

- If previous obs, $c$ and Phi entails $x <> v$
  Then $p(x = v \mid c) = 0$

- If Phi consistent with all values for $x$
  Then $p(x = v \mid c)$ is based on priors
    - E.g., uniform prior $= 1/m$ for $m$ possible values of $x$
Observe out = 1:
- $C = [G(A), G(B), G(C)]$
- $P(C) = .97$
- $P(out = 1 \mid C) = ?$
  - $= 1$
- $P(C \mid out = 0 ) = ?$
  - $= .97/p(x=v)$

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

Observe out = 0:
- $C = [G(A), G(B), G(C)]$
- $P(C) = .97$
- $P(out = 0 \mid C) = ?$
  - $= 0$
- $P(C \mid out = 0 ) = ?$
  - $= 0 \times .97/p(x=v) = 0$
How do the single faults change?
- which are eliminated?
- which predict observations?
- Which are agnostic?

Priors for Single Fault Diagnoses:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>p(S1)</td>
<td>.008</td>
<td>.008</td>
<td>.001</td>
</tr>
<tr>
<td>p(S0)</td>
<td>.001</td>
<td>.001</td>
<td>.008</td>
</tr>
<tr>
<td>p(U)</td>
<td>.001</td>
<td>.001</td>
<td>.001</td>
</tr>
</tbody>
</table>
Summary: 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

\( P(x=v\mid c) \) estimated using Model:

- If previous obs, c and Phi entails \( x = v \)
  Then \( p(x = v \mid c) = 1 \)
- If previous obs, c and Phi entails \( x <> v \)
  Then \( p(x = v \mid c) = 0 \)
- If Phi consistent with all values for x
  Then \( p(x = v \mid c) \) is based on priors
  - E.g., uniform prior = \( 1/m \) for \( m \) possible values of \( x \)

Top 6 of 64 = 98.6% of P
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.
Exploring the Improbable

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

- Sherlock Holmes.
The Sign of the Four.

Compare Most Likely Hypothesis to Observations

Helium tank

Oxidizer tank

Fuel tank

Flow\textsubscript{1} = zero
Pressure\textsubscript{1} = nominal
Pressure\textsubscript{2} = nominal

Acceleration = zero

It is most likely that all components are okay.
Increasing Cost

Feasible

Infeasible

A*

Conflict-directed A*

Increasing Cost

Infeasible

Feasible
Conflict-directed A*

Increasing Cost

Conflicting 1

Infeasible

Conflicting 1

Feasible

Conflict-directed A*

Increasing Cost

Conflicting 1

Infeasible

Conflicting 1

Feasible
Increasing Cost

Conflict-directed A*

Conflict 1
Infeasible
Conflict 2
Feasible

Conflict 3

Conflict-Directed A*: Generating The Best Kernel

A1=U, M1=U, M2=U

Insight:
• Kernels found by minimal set covering
• Minimal set covering is an instance of breadth first search.
• To find the best kernel, expand tree in best first order.

A1=U, A2=U, M1=U, M3=U

A1=U, M1=U, M3=U
Conflict-Directed A*: Generating The Best Kernel

Insight:
- Kernels found by minimal set covering
- Minimal set covering is an instance of breadth first search.
- To find the best kernel, expand tree in best first order.

Summary:
Model-based Diagnosis

- A failure is a discrepancy between the model and observations of an artifact.
- Diagnosis is symptom directed
- Symptoms identify conflicting components as initial candidates.
- Test novel failures by suspending constraints and testing consistency.
- Newly discovered conflicts further prune candidates.
Research in Model-based Diagnosis

Methods exist for:

- Focusing on likely diagnoses
- Active probing
- Diagnosing dynamic systems and monitoring behavior
- Repairing and compensating for failures
- Diagnosing hybrid discrete/continuous systems
- Performing distributed diagnosis

Appendix 1

Extracting Conflicts Through Unit Propagation
procedure propagate($C$)  // $C$ 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

\[\neg (O_1 = G) \lor \neg (A = 1) \lor X = 1 \Rightarrow X = 1\]

\[\neg (A_1 = G) \lor \neg (X = 1) \lor \neg (Y = 1) \lor \neg (F = 1) \lor \neg (F = 0)\]

\[\neg (O_2 = G) \lor \neg (B = 1) \lor Y = 1 \Rightarrow Y = 1\]

\[\neg (F = 1) \lor \neg (F = 0)\]

\[\neg (F = 1) \lor \neg (F = 0)\]
Find Symptom Using Unit Propagation

O1=G
\[ \neg (O1=G) \lor \neg (A=1) \lor X=1 \Rightarrow \]
\[ \neg (A1=G) \lor \neg (X=1) \lor \neg (Y=1) \lor F=1 \]

A=1
\[ \neg (O1=G) \lor \neg (A=1) \lor X=1 \Rightarrow \]
\[ \neg (A1=G) \lor \neg (X=1) \lor \neg (Y=1) \lor F=1 \]

A1=G
\[ \neg (O2=G) \lor \neg (B=1) \lor Y=1 \Rightarrow \]
\[ \neg (O2=G) \lor \neg (B=1) \lor Y=1 \]

B=1
\[ \neg (O1=G) \lor \neg (A=1) \lor X=1 \Rightarrow \]
\[ \neg (A1=G) \lor \neg (X=1) \lor \neg (Y=1) \lor F=1 \]

O2=G
\[ \neg (O2=G) \lor \neg (B=1) \lor Y=1 \Rightarrow \]
\[ \neg (O2=G) \lor \neg (B=1) \lor Y=1 \]
**Extract Conflict by Tracing Support**

**procedure** `Conflict(C)`  // C is an inconsistent clause
  **for each** literal I in C
    **union** `Support-Conflict(l, support(l) )`
  **end** `Conflict`

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

---

**Candidate Test with Conflict Extraction**

**procedure** `Test_Candidate(c,M,obs)`
1. Assert candidate assignment c
2. Propagate obs through model M using unit propagation.
3. If inconsistent clause return Conflict(c)
4. Else search for satisfying solution using DPLL
   • If inconsistent **return c** as a conflict.
   • Else **return “consistent”**
Appendix 2

Single-Fault Diagnosis using Conflicts

Single Fault Diagnosis w Conflicts: Generate Candidates From Symptom

Single_Fault_w_Conflicts(M, X, Obs)
\ Model M, Mode variables X, Observation Obs

1. Assume all components okay,
   All_Good = \{ x=G | x \in X\}

2. Conflict \leftarrow Test_Candidate(All_Good, M, Obs)

3. If Conflict = “consistent” return All_Good

4. Generate single fault candidates
   Cands \leftarrow \{\{x=U\} \cup Z=G | x=G \in Conflict, Z=X-\{x\}\}

5. Test_Candidates(Cands, M, Obs)
Symptom: F is observed 0, but should be 1
Conflict: \{O1=G, O3=G, A1=G, A2=G\} is inconsistent
Candidates: \{O1=U, O3=U, A1=U, A2=G\}

Single Fault Diagnosis w Conflicts: Test Candidates Collecting Conflicts

Single_Fault_Test_Candidates(C, M, Obs)
\| Candidates C, Model M, Observation Obs
Solutions \leftarrow \{\}, Conflicts \leftarrow \{
For each c in C

If c is a superset of some conflict in Conflicts
Then inconsistent candidate, ignore.
Else Conflict = Test_Candidate(c, M, Obs)
If Conflict = “consistent”
Then add c to Solutions
Else add Conflict to Conflicts
return Solutions
Test Candidates Collecting Conflicts

Candidates: \({\{O1=U\ldots\}, \{O3=U\ldots\}, \{A1=U\ldots\}, \{A3=U\ldots\}}\)

Solutions: \(\emptyset\)

- First candidate \(\{O1=U, \ldots\}\)

- Suspend \(O1\)'s constraints

Test Candidates Collecting Conflicts

Candidates: \({\{O1=U\ldots\}, \{O3=U\ldots\}, \{A1=U\ldots\}, \{A3=U\ldots\}}\)

Solutions: \(\emptyset\)

- First candidate \(\{O1=U, \ldots\}\)
- Suspend \(O1\)'s constraints
Test Candidates Collecting Conflicts

Candidates: \{ \{ O_1 = U \ldots \} , \{ O_3 = U \ldots \} , \{ A_1 = U \ldots \} , \{ A_3 = U \ldots \} \}

Solutions: \{ \}

- First candidate \{ O_1 = U, \ldots \}
- Suspend \( O_1 \)'s constraints
- Test consistency

Test Candidates Collecting Conflicts

Candidates: \{ \{ O_3 = U \ldots \} , \{ A_1 = U \ldots \} , \{ A_3 = U \ldots \} \}

Solutions: \{ \{ O_1 = U \ldots \} \}

- First candidate \{ O_1 = U, \ldots \}
- Suspend \( O_1 \)'s constraints
- Test consistency \( \rightarrow \) Consistent: Add to solutions
Test Candidates Collecting Conflicts

Candidates: \{\{O3=U\ldots\}, \{A1=U\ldots\}, \{A2=U\ldots\}\}
Solutions: \{\{O1=U\ldots\}\}

- Second candidate \{O3=U, \ldots\}
- Suspend O3's constraints
- Test consistency

Test Candidates Collecting Conflicts

Candidates: \{\{O3=U\ldots\}, \{A1=U\ldots\}, \{A2=U\ldots\}\}
Solutions: \{\{O1=U\ldots\}\}

- Second candidate \{O3=U, \ldots\}
- Suspend O3’s constraints
- Test consistency → Inconsistent
Test Candidates Collecting Conflicts

Candidates: \{\{O3=U\ldots\}, \{A1=U\ldots\}, \{A2=U\ldots\}\}
Solutions: \{\{O1=U\ldots\}\}
Conflicts: \{\{O1=G, O2=G, A1=G\}\}

- Second candidate \{O3=U, \ldots\}
- Suspend O3’s constraints
- Test → Inconsistent

• Extract Conflict: \{O1=G, O2=G, A1=G\}
• Use to prune candidates

Test Candidates Collecting Conflicts

Candidates: \{\{A1=U\ldots\}, \{A2=U\ldots\}\}
Solutions: \{\{O1=U\ldots\}\}
Conflicts: \{\{O1=G, O2=G, A1=G\}\}

- Third candidate \{A1=U, \ldots\}
- Subsumed by conflict? → No, since A1 = U, not A1=G
- Suspend A1’s constraints
- Test → Consistent
**Test Candidates Collecting Conflicts**

Candidates: \{A2=U\…\}\}

Solutions: \{O1=U\…\}, \{A1=U\…\}\}

Conflicts: \{O1=G, O2=G, A1=G\}\}

- Fourth candidate \{A2=U, \…\}
- Subsumed by conflict? → Yes, since O1=G, O2=G and A1=G
- Eliminate candidate

**Consistent**

---

**Test Candidates Collecting Conflicts**

Candidates: \{}

Solutions: \{O1=U\…\}, \{A1=U\…\}\}

Conflicts: \{O1=G, O2=G, A1=G\}\}

- Return Solutions → O1 or A1 broken
Single Fault Diagnoses are the Intersection of All Conflicts

\{A1=G, M1=U, M2=U\} \quad \text{conflict 1.}
\{A1=U, A2=U, M1=U, M3=U\} \quad \text{conflict 2}

A1=U or M1=U or M2=U \quad \text{removes conflict 1.}
A1=U or A2=U or M1=U or M3=U \quad \text{removes conflict 2}

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