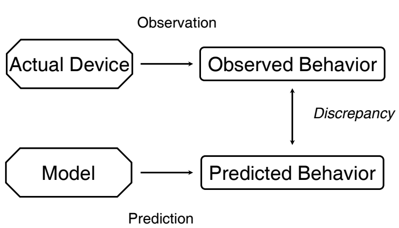
Model Based Reasoning

6.871 - Lecture 16

Outline

- Basics of the task
- The nature of models
- What we know how to do
- What we don't know how to do (so well)

Interaction of Prediction and Observation



Components of the Task

- Given
  - Observations of a device behavior (inputs, outputs)
  - a description of internal structure
  - a description of component behavior
- Determine
  - which components could have failed so as to product the observed misbehavior
  - the simplest set of component failures which can explain the misbehavior
- Buzzwords
  - Reasoning from design models
  - Reasoning from first principles
  - Deep reasoning

Why Model Based Diagnosis

- Familiar task that people do well

Compared to *heuristic classification*
- Don't need new rule set needed for each device
- Device independent
- "Free" given a design description

Compared to *traditional diagnostics*
- Diagnosis ≠ verification or manufacturing testing
- Symptom directed
- Can cover a wider range of faults

When not to use it

- Some things are too difficult to infer from the models
  - intermittent or flaky behavior
- The device and range of faults is small enough to permit exhaustive simulation
- The device and range of faults is small enough to generate an exhaustive fault dictionary

Basic Theses

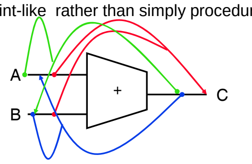
- Hypothesis generation, test and discrimination are fundamental problems of diagnosis
- Different amounts and types of knowledge can be brought to bear at each phase
- The set of possibilities explored spans a wide range of potential systems within this common PSP
- More complex devices require better abstractions.

Useful Characteristics of Structure Representations

- Hierarchical
  - Possibly multiple: behavioral, physical
  - Possibly not strict: components with multiple functional roles
- Object-oriented, isomorphic to the device
  - Procedural objects
  - Interconnected in same topology
- Unified: Both runnable and examinable

Behavior Representation

- Expressions capturing relationships between values at terminals
  - Multi-directional
  - Constraint-like rather than simply procedural



- To compute C: Evaluate A + B
- To compute A: Evaluate C - B
- To compute B: Evaluate C - A

Three Fundamental Problems

- Hypothesis Generation
  - Given a symptom, which components could have produced it?
  - (Which are most likely to have produced it)
- Hypothesis Testing
  - Which components could have failed to account for all observations?
- Hypothesis Discrimination
  - What additional information should we acquire to distinguish among the remaining candidates?

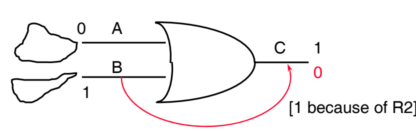
Generation

- Generator provides plausible hypotheses
  - Complete
  - Non-redundant
  - *Informed*

Generation

But: Not every input influences the specified output

- G4: Use behavior model to determine relevant inputs
  - Have simulation keep dependency records
  - Trace back through these to determine candidates



R1: IF A=1 then C=1
R2: IF B=1 then C=1
R3: IF A=0 and B=0 then C= 0

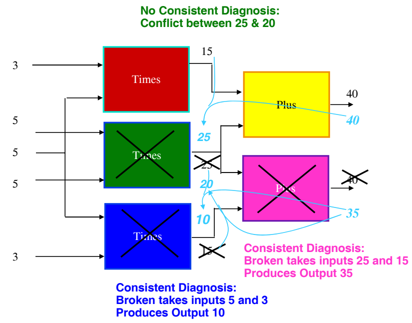
Generation

Generators should be:

- Complete
- Non-redundant
- *Informed*

- G1: Exhaustive enumeration of components
- G2: Find all components connected to the discrepancy
- G3: Find all components *upstream of* the discrepancy
- G4: Use behavior model to determine relevant inputs

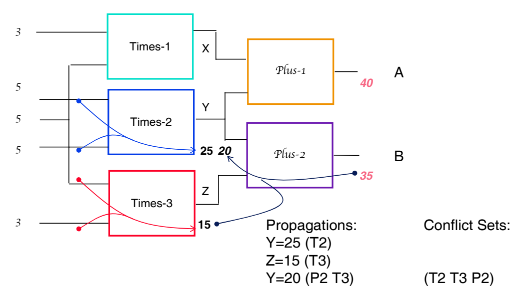
Model Based Troubleshooting Constraint Suspension



Using Behavior Information: GDE

- GDE = General Diagnostic Engine
- Propagate not just values, but underlying assumptions as well
  - Assumptions are the proposition that a component is working according to design

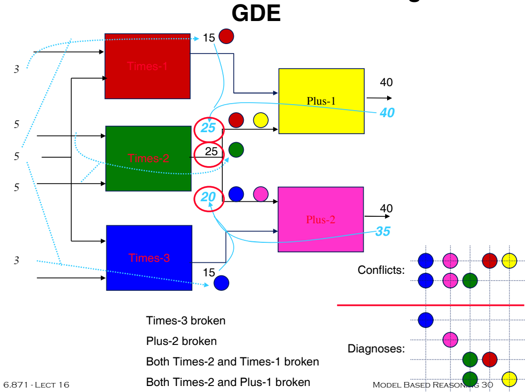
Model Based Troubleshooting GDE



Using Behavior Information: GDE Assumption Propagation and Set Covering

- GDE = General Diagnostic Engine
- Propagate not just values, but underlying assumptions as well
  - Assumptions are the proposition that a component is working according to design
- Construct conflict sets
  - Sets of assumptions, not all of which can be true at once
    eg: (T2 T3 P2)
    (T1 T3 P1 P2)
- "Explain" each conflict set
  - By a set covering
    eg: (P2) (T3 P2)
  - By a *minimal* set covering: eg: (T3)

Model Based Troubleshooting GDE



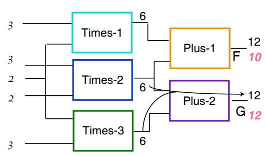
Good News/Bad News

- The good news
  - Generates all the logically possible candidates
  - Including multiple point of failure
- The bad news
  - Set covering is well known to be exponential
- The (slightly less) bad news
  - The number of components at any level of detail is relatively small

Corroboration Proves Nothing

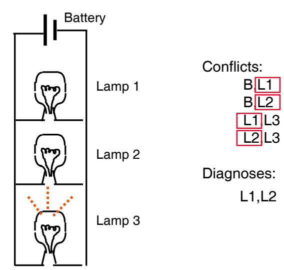
- The basic intuition
  - Involved in discrepancy means suspect
  - Therefore: Involved in corroboration means exonerated
- This is wrong
  - Involved in corroboration only means that you didn't tickle this problem yet.
    - with these inputs
    - with the specific observations you chose to make so far

Corroboration Example and Counter Example



Fault Models

- Good News: what we've seen so far doesn't need them
- Bad News: what we've seen so far can't use them



Fault Models

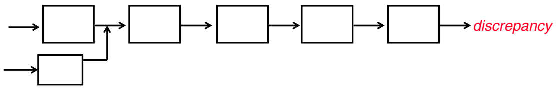
- Extend the notion of fault model to include multiple behavioral modes:
  - Designed behavior (i.e., the *correct* behavior)
  - Known faulty behaviors
  - Residual behavior (i.e. everything *besides* designed and known faults)
  - Their probabilities
- Start with models of correct behavior
- When conflicts exist, substitute a fault model for some member of the conflict set
- Drive the choice of substitution by failure probabilities
  - best diagnosis is most likely set of behavior modes for the various candidates capable of removing all discrepancies
  - i.e., best first search for conflict free set of behavior modes

Three Fundamental Problems

- Hypothesis Generation
  - Given a symptom, which components could have produced it?
- Hypothesis Testing
  - Which components could have failed to account for all observations?
- Hypothesis Discrimination
  - What additional information should we acquire to distinguish among the remaining candidates?

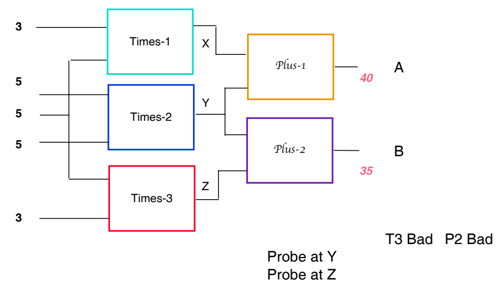
Probing and Testing

- Purely structural
  - Follow discrepancies upstream (guided probe)
  - Split candidate space topologically



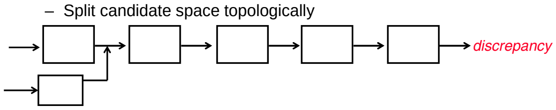
- Add behavioral information:
  - Split topologically: G&T on the sub-problem
  - Predict consequences of candidate malfunction; probe where it is most informative.

Informative Probes



Probing and Testing

- Purely structural
  - Follow discrepancies upstream (guided probe)
  - Split candidate space topologically



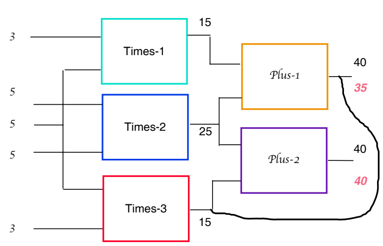
- Add behavioral information:
  - Split topologically: G&T on the sub-problem
  - Predict consequences of candidate malfunction; probe where it is most informative.
- Add failure probabilities
  - Cost-benefit calculation using maximum entropy methods

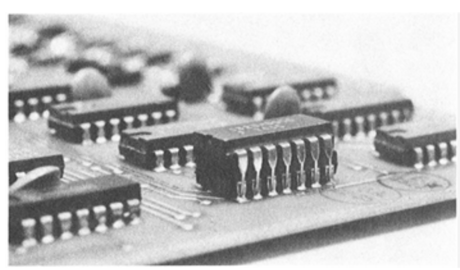
Assumption: Computation is cheap compared to probing (think of chips)

Difficulties

- Model based reasoning is only as good as the model
- Tension between completeness of description and tractability of reasoning.
- Scaling: size alone isn't the issue (but it is an issue)
- Complex behavior is an issue
  - VCR, ALU, Pentium, PowerPC, Disk Controller
  - This requires new vocabulary, new abstractions
  - Temporally coarse descriptions are often important
    - Memory and state are hard to model
    - Temporally coarse representations can hide the state usefully

The Model Isn't How It Is





The Model Isn't How It Is

- Because it shouldn't be that way
  - bridge faults, assembly error
- Because of unexpected pathways of interaction
  - eg heat, radiation
- In practice, by our choices
  - deciding not to represent each individual wire segment
- In principle: it's impossible

Complexity vs Completeness

- Any simplifying assumption risks incompleteness
- Make too few assumptions and
  - diagnosis becomes indiscriminate
  - drown in complexity, ambiguity

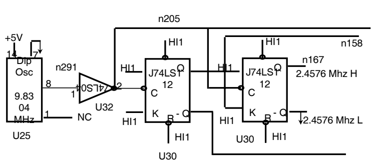
Model Selection and Formulation Is a Key Problem

- There are no assumption-free representations
  - perhaps we can use more than one
- Completeness and complexity conflict
  - we'll need to choose judiciously
- Basic question: *whence the model?*
  How do we know how to think about the device?

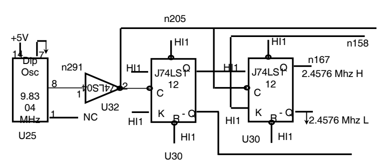
Another Problem: Complex Behavior

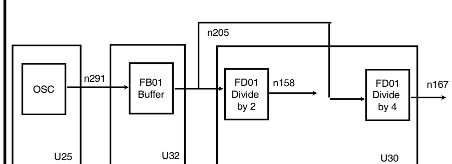
- An engineer plugs in a broken circuit board, makes a half dozen simple probes with an oscilloscope, and after ten minutes ends up swapping a chip, which fixes the problem.
- A model-based troubleshooting program spends a day simulating the expected behavior of the same misbehaving board, and requests that a logic analyzer be used to capture a certain subset of the signals. After some hours of computation, it concludes that any of the 40 chips or 400 wires on the board could be responsible for the misbehavior.
- Why?

The Two Different Approaches to MBT



The Two Different Approaches to MBT





If n167 is "flat" then U25, U32 and U30 form a conflict. But Oscillators tend to fail more frequently, so U25 is more likely to be broken. A probe of n291 is advised.

More (detail) is Worse

- The naïve approach suggests a detailed, step by step simulation of the device as the first phase of the diagnosis.
- For a reasonable circuit with internal states, all interesting behavior exists over the time span of many thousands to millions of clock cycles.
- The naïve approach fails to capture the right functional abstractions
  - Devices: Central controller
  - Behavior: Frequency
    - Changing
    - Stable

The Problems to be Faced

- Models are incomplete.
- Observations are costly.
- Observations are incomplete and imprecise.
- Prediction is costly.
- Prediction is incomplete.

How to Address these Problems

- Choose the representation of primitive elements and connections so as to sacrifice completeness for efficiency.
  - Treat physically separate components with indistinguishable failure modes as one component.
  - Treat devices whose failure requires the same repair as one device.
  - Don't represent very unlikely failure modes
- Describe signals in a way which is easy to observe.
- Represent the likelihood of failure modes.
- Use temporally abstract description of signals.
- Use multiple levels of behavioral abstraction.

Principles of Modeling

- Components in the *physical representation* should correspond to the possible repairs.
- Components in the *functional representation* should facilitate behavioral abstraction.

Principles of Modeling

- Components' behavioral representation should employ features that are easy to observe.
- A temporally coarse description is better than no description.
- A sequential circuit should be encapsulated into a single component whose behavior can be described in a temporally coarse manner.
- Represent a failure mode if it has a high likelihood.
- Represent a failure mode if the misbehavior is drastically simpler than the normal behavior

Conclusions

- General purpose paradigm (with variations)
- Largely domain independent
- Successfully employed in practice
- Major research issues are in modeling, not reasoning methods
  - complex behavior
  - model selection
  - model formulation