Z [zed] defines state schemas with variables and predicates

- Rich mathematical toolkit to express sets and conditions
- Shown here
  - Powerset P
  - (Partial) function
  - Subset condition
- Originally developed in Oxford by J.-R. Abrial, 1980
- Large body of textbook literature
- The Z user group has an annual conference for 20 years

An operation is defined like a state

- Purchase
  - $s \in \text{BoxOffice}$
  - $s? : \text{Seat}$
  - $c? : \text{Customer}$
  - $s? \in \text{seating} \setminus \text{dom sold}$
  - $\text{sold}' = \text{sold} \setminus \{(s? \leftrightarrow c?)\}$

- Available
  - $\exists \in \text{BoxOffice}$
  - $s? : \text{Seat}$
  - $r! : \text{Response}$
  - $s? \in \text{seating} \setminus \text{dom sold}$
  - $r! = \text{available}$

BoxOffice is changing
- $s$ & $c$ are inputs
- sold is the after image of sold
- What isn’t mentioned doesn’t change

BoxOffice does not change
- $r$ is an output
Verification requires manual proof

- We frequently want to prove properties for the systems we model
  - E.g. we could prove that a seat is never purchased to more than one customer
- In Z this requires manual derivation of the property from the specification
  - The specification language is very rich
  - The derivation rules are basically all we know about sets and their relationships
- We would like to automate the verification…

Agenda

- Structured state specification
- Transition-system based verification
- The challenge of unification

Transition systems contain only uninterpreted atomic states

Here is our box office again

\[ S = (S, T, P, L) \]
\[ S : \text{Set of states} \]
\[ T \subseteq S \times S : \text{Transitions} \]
\[ P : \text{Atomic propositions} \]
\[ L : S \rightarrow 2^P : \text{Labels} \]

Think Moore automata, but simpler
We want to talk about temporal properties

- Temporal logic (Pnueli 1977 and others)
  - Temporal operators
    - X: Next time
    - G: Globally
    - F: Finally
    - U: Until
  - Path quantifiers
    - A: All paths
    - E: Exists a paths

Model Checkers simply walk through the transition system

- Clarke, Emerson, Sistla, 1986
  - Automatic Verification of Finite State Concurrent Systems Using Temporal Logic Specifications
  - ACM TOPLAS, Vol. 8, No. 2
  - What did you think of the paper?

Model Checkers simply walk through the transition system

- Clarke et al., 1987
  - Advanced Solution
  - Question: Can we verify interesting properties automatically?
  - Contribution: Yes, we can! By exploring the state space and see what formulas hold. Efficient algorithm to verify CTL formulas is given.
  - Extension to handle fairness
  - Verification: Alternating Bit Protocol

Software systems have infinite state spaces

- String s = "";
  - for (; ; )
    - s += (char) System.in.read();

  - Mitigations
    - Increase possible state space size
    - Symbolic model checking
    - Transform to finite system
    - Predicate abstraction
    - Just don’t explore indefinitely
    - Bounded model checking
  - This all does not solve the underlying problem
Agenda

- Structured state specification
- Transition-system based verification
- The challenge of unification

**Z and model checking come from different ends of the spectrum**

---

Let’s talk about the Timed Communicating Object-Z (TCOZ) paper

- Mahony, Dong, 1998
  - Blending Object-Z and Timed CSP: An introduction to TCOZ
  - ICSE-20
  - Could you make sense of the paper?

**TCOZ offers some leverage**

- Mahony, Dong, 1998
  - Blending Object-Z and Timed CSP: An introduction to TCOZ
  - ICSE-20
  - Enhanced method
    - Problem: We can model algorithms, and we can model concurrency, but not both
    - Contribution: Introduce CSP channels into Object-Z with reasonable semantics, augmented with timing
    - Validation: Two common modeling examples

- T. Hoare’s CSP (1985)
  - Classic concurrent system calculus
  - Pretty close to CCS

- Object-Z
  - OO version of Z proposed by G. Smith

- TCOZ can model abstract communication between “types” (object schemas)
  - Can prove object correctness individually
  - Verify global temporal properties automatically
More work needs to be done for convenient specification and verification

From Infinity to 42 and Back
- Structured state specification
- Transition-system based verification
- The challenge of unification

Coming Up

- Research Method: Case Studies
  - Possible papers
    - Futurebus+ verification with model checking
    - TCOS II specification with Statecharts
- Technology Maturation
  - Topic suggestion
    - Specification of reactive systems