Heterogeneous Models of Computation: An Abstract Algebra Approach

EE249 Lecture
Taken from
Roberto Passerone PhD Thesis
Objectives

- Provide the foundation to represent different semantic domains for the Metropolis metamodel
- Study the problem of heterogeneous interaction
- Formalize concepts such as abstraction and refinement
An Example of Interaction

- Combine a synchronous model with a dataflow model
- Synchronous model
  - Total order of event
- Data flow model
  - Partial order of events
- Discrete Time model
  - Metric order of events
An Example of Heterogeneous Interaction

- The interaction is derived from a common refinement of the heterogeneous models.
- The resulting interaction depends on the particular refinements employed.
- Our objective is to derive the consequences of the interaction at the higher levels of abstraction.
**Data Flow Model**

- Assume signals take values from a set $V$
- Each signal is a sequence from $V$ (an element of $V^*$)
- Let $A$ be the set of signals
- One behavior is a function
  - $f : A \rightarrow V^*$
- A data-flow agent is a set of those behaviors
Synchronous Model

- Signals are again sequences from \( V \) (elements of \( V^* \))
  ... But are synchronized

- One element of the sequence is \( g : A \rightarrow V \)

- One behavior is a sequence of those functions
  \[ <g_i> \in (A \rightarrow V)^* \]

- A synchronous agent is a set of those sequences
Discrete Time Model

- Assume time is represented by the positive integers $\mathbb{N}$
- Then define a behavior
  - $h: \mathbb{N} \rightarrow (A \rightarrow V)$
- A discrete time agent is a set of those functions
Discrete to Synchronous Abstraction

Synchronous

Discrete

* * * * *

a b e

n p r

g j l

o n p p r s

a b c e e

g g j j l m

* * * * *
Discrete to Data Flow Abstraction
Interaction Propagation

Synchronous

Data flow

Discrete

1. Refinement
2. Composition
3. Projection
4. Abstraction
Objectives

◆ Provide a semantic foundations for integrating different models of computation
  ∗ Independent of the design language
  ∗ Not just specific to the Metropolis meta-model

◆ Maximize flexibility for using different levels of abstraction
  ∗ For different parts of the design
  ∗ At different stages of the design process
  ∗ For different kinds of analysis

◆ Support many forms of abstraction
  ∗ Model of computation (model of time, synchronization, etc.)
  ∗ Scoping
  ∗ Structure (hierarchy)
Overview

Meta Model

Data Flow

Pre-Post

Process Networks

Discrete Time

Non-metric Time

Continuous Time

Agent Algebras

Conservative Approximations

Domain of agents with operations: projection, renaming and composition
Scope

◆ Concentrate on
  ◇ Natural semantic domains (sets of agents)
  ◇ Relations and functions over semantic domains
  ◇ Relationships between semantic domains and their relations and functions

◆ Defer worrying about specific abstract syntaxes and semantic functions
  ◇ Convenient for manual, formal reasoning
  ◇ De-emphasizing executable and finitely-representable models (for now)
Agents and Behaviors

- For each model of computation we always distinguish between
  - the domain of individual behaviors
  - the domain of agents

- For different models of computation individual behaviors can be very different mathematical objects
  - We always call these objects traces
  - The nature of the elements of the carrier is irrelevant!

- An agent is primarily a set $P$ of traces
  - We call them trace structures
  - Also includes the signature: $T = (\gamma, P)$
Trace and Trace Structure Algebras

Model of individual behaviors

Set of traces $C$
- Trace algebra
  - Projection
  - Renaming
  - Concatenation

Set of trace structures $A$
- Trace structure algebra
  - Composition
  - Scoping
  - Instantiation

A trace structure contains a set of traces

Model of agents (semantic domain)
Essential Elements

- Must be able to name elements of the model
  - Variables, actions, signals, states
  - We do not distinguish among them and refer to them collectively as a set of signals $W$

- Each agent has an alphabet and a signature
  - Alphabet: $A \subseteq W$
  - Signature: $\gamma = A, \gamma = (I, O)$, etc.

- The operations on traces and trace structures must satisfy certain axioms
  - The axioms formalize the intuitive meaning of the operations
  - They also provide hypothesis used in proving theorems
  - Trade-off between generality and structure
Metric Time Traces

\[
\gamma = (V_R, V_N, M_I, M_O)
\]
\[
x = (\gamma, \delta, f)
\]
\[
f(v) = [0, \delta] \mapsto R
\]
\[
f(n) = [0, \delta] \mapsto N
\]
\[
f(a) = [0, \delta] \mapsto \{0, 1\}
\]

- **Model time as a metric space**
  - Can talk about the difference in time between points in the behavior in quantitative terms
  - Able to specify timing constraints in quantitative terms

- **Able to represent continuous as well as discrete behavior**

- **Projection and renaming easily defined on the functions**
Metric Time Model: Traces

◆ A trace \( x \) models one execution of a hybrid system:

◆ Signature \( \gamma = (\) 
  
  \( V_R \): real valued var's,
  
  \( V_N \): integer valued var's,
  
  \( M_I \): input actions,
  
  \( M_O \): output actions)

◆ The alphabet \( A \) of \( x \) is the union of the components of \( \gamma \)

◆ \( \delta \) is a non-negative real number
  
  ◦ Length (in time) of \( x \)
  
  ◦ Can be infinity

◆ \( f \) gives values as a function of time:
  
  \( f: V_R \rightarrow [0, \delta] \rightarrow \mathbb{R} \),
  
  \( f: V_N \rightarrow [0, \delta] \rightarrow \mathbb{N} \),
  
  \( f: M_I \rightarrow [0, \delta] \rightarrow \{0, 1\} \),
  
  \( f: M_O \rightarrow [0, \delta] \rightarrow \{0, 1\} \).
Metric Time Model: Operations on Traces

- Let $x' = \text{proj}(B)(x)$
  - represents scoping
  - $B$ is a subset of $A$
  - $\gamma'$ and $f'$ are restricted to variables and actions in $B$
  - $\delta' = \delta$

- Let $x' = \text{rename}(r)(x)$
  - represents instantiation
  - $r$ is a one-to-one function with domain $A$
  - variables and actions in $\gamma'$ and $f'$ are renamed by $r$
  - $\delta' = \delta$

- Let $x'' = x \cdot x'$
  (concatenation)
  - represents sequential composition
  - $\gamma' = \gamma$, $d$ is finite, and end of $x$ matches beginning of $x'$
  - $\gamma'' = \gamma$
  - $d'' = d + d'$
  - $f''(v, t)$ is equal to $f(v, t)$ for $t \leq d$
    $f'(v, t - d)$ for $t \geq d$
A trace structure $T = (\gamma, P)$ models a process or an agent of a hybrid system

- $P$ is a set of traces with signature $\gamma$

Traits:

- $T$ refines $T'$ if $P \subseteq P'$
- Natural model for physical components (such as those described with differential equations, possibly with discrete control variables)
- Too detailed for many other aspects of embedded systems
- Not a finite representation
  - Finite representations, synthesis and verifications algorithms are clearly important, but not a focus of this class
- Trace structures constructed the same way for any trace algebra
**Metric Time Model: Operations on Trace Structures**

- Let $T' = \text{proj}(B)(T)$
  - $B$ is a subset of $A$
  - $\gamma'$ is restricted to variables and actions in $B$
  - $P' = \text{proj}(B)(P)$

- Let $T' = \text{rename}(r)(T)$
  - $r$ is a one-to-one function with domain $A$
  - variables and actions in $\gamma'$ are renamed by $r$
  - $P' = \text{rename}(r)(P)$

- Let $T'' = T \parallel T'$ (par. comp.)
  - $\gamma''$ combines $\gamma$ and $\gamma'$
  - $P''$ maximal set s.t.
    - $P = \text{proj}(A)(P'')$
    - $P' = \text{proj}(A')(P'')$

- Let $x'' = x \cdot x'$ (seq. comp.)
  - $\gamma' = \gamma$
  - $P'' = P \cdot P'$ (roughly)
Non-metric Time Traces

\[ \gamma = (V_R, V_N, M_I, M_O) \]
\[ x = (\gamma, L) \]
\[ m(t) = V_R \rightarrow R \]
\[ V_N \rightarrow N \]
\[ M \rightarrow \{0, 1\} \]

- Model time as a non-metric space
  - Can only talk about precedence in time (including dense time)
- Based on Totally Ordered Multi-Sets
  - Totally ordered vertex set \( V \)
  - Labeling function \( \mu \) from the vertex set \( V \) to a set of actions \( \Sigma \)
  - We do not distinguish isomorphic vertex sets
Pre-Post Traces

\( \gamma = (M_I, M_O) \)
\( x = (\gamma, s_i, s_f) \)

- Model only pre- and post-conditions (not intermediate states)
- Suitable for studying the semantics of programming languages
- Trace theory version of Hoare triples
Relationships between Semantic Domains

◆ Each semantic domain has a refinement order
  ◆ Based on trace containment
  ◆ $T_1 \subseteq T_2$ means $T_1$ is a refinement of $T_2$
  ◆ Guiding intuition: $T_1 \subseteq T_2$ means $T_1$ can be substituted for $T_2$

◆ Abstraction mapping
  ◆ If a function $H$ between semantic domains is monotonic, detailed implies abstract: If $T_1 \subseteq T_2$ then $H(T_1) \subseteq H(T_2)$
  ◆ Analogy for real numbers $r$ and $s$: If $r \leq s$ then $\lceil r \rceil \leq \lceil s \rceil$

◆ Conservative approximations
  ◆ A pair of functions $\Psi = (\Psi_l, \Psi_u)$ is a conservative approximation if $\Psi_u(T_1) \subseteq \Psi_l(T_2)$ implies $T_1 \subseteq T_2$
  ◆ Analogy: $\lceil r \rceil \leq \lceil s \rceil$ implies $r \leq s$
  ◆ Abstract implies detailed
Trace and Trace Structure Algebras

Trace algebra \( C \)

Trace structure algebra \( A \)

Upper Bound

Lower Bound

"Abstract" Domain

\( \Psi_u \)

\( \Psi_l \)

\( \Psi_{\text{inv}} \)

"Detailed" Domain
Deriving Conservative Approximations

Homomorphism: mapping that commutes with the operations of projection, renaming and concatenation on traces
Homomorphism

- From metric to non-metric
  - Must define a notion of event in the metric model
  - Must define how to construct the corresponding vertex set

- From non-metric to pre-post
  - Simply remove the intermediate steps and keep only the end-points
Metric to Non-Metric Traces

Event: point in time where the function changes value

Homomorphism discards non-event points

The information about metric time is effectively lost
**From Metric to Non-metric Time**

- \( f \) is stable at \( t_0 \) if there is \( \varepsilon > 0 \) such that \( f \) is constant on \([t_0 - \varepsilon, t_0]\)
- \( f \) has an event at \( t_0 \) if it is not stable
- Vertex Set \( V = \{ t_0 \mid f \) has an event at \( t_0 \} \)
Building the Upper Bound

◆ Let P be a set of traces, and consider the natural extension to sets $h(P)$ of h

◆ Clearly $P \subseteq h^{-1}(h(P))$
  ♦ Because h is many-to-one
  ♦ This indeed is an upper bound!
  ♦ Equality holds if h is one-to-one

◆ Hence define
  ♦ $\Psi_u(T) = (\gamma, h(P))$
Building the Upper Bound
Building the Lower Bound

- We want \( P \supseteq h^{-1}( \text{lb of } P ) \)
- If \( x \) is not in \( P \), then \( h(x) \) should not be in the lower bound of \( P \)
- Hence define
  \[ \Psi_1(T) = h(P) - h(B_c(A) - P) \]
- There is a tighter lower bound
Building the Lower Bound

\[ h(P) - h(B_c(A) - P) \]

\[ h^{-1}(h(P) - h(B_c(A) - P)) \]

\[ h(B_c(A) - P) \]

\[ B_c(A) - P \]
Conservative Approximations: Inverses

- Apply $\Psi_u$
- Apply $\Psi_l$
- Consider $T$ such that

$$\Psi_u(T) = \Psi_l(T) = T'$$
Conservative Approximations: Inverses

- Apply $\Psi_u$
- Apply $\Psi_l$
- Consider $T$ such that $\Psi_u(T) = \Psi_l(T) = T'$
- Then $\Psi_{inv}(T') = T$
Conservative Approximations: Inverses

- Apply $\Psi_u$
- Apply $\Psi_l$
- Consider $T$ such that
  \[
  \Psi_u(T) = \Psi_l(T) = T'
  \]
- Then $\Psi_{\text{inv}}(T') = T$
- Can be used to embed high-level model in low level
Combining MoCs

Want to compose $T_1$ and $T_2$ from different trace structure algebras

- Construct a third, more detailed trace algebra, with homomorphisms to the other two
- Construct a third trace structure algebra
- Construct cons. approximations and their inverses
- Map $T_1$ and $T_2$ to $T_1'$ and $T_2'$ in the third trace structure algebra
- Compose $T_1'$ and $T_2'$
Conclusions

- Semantic foundations for the Metropolis meta-model
- All models of computation of importance “reside” in a unified framework
  - They may be better understood and optimized
- Trace Algebra used as the underlying mathematical machinery
  - Showed how to formalize a semantic domain for several models of computation
- Conservative approximations and their inverses used to relate different models