

CmpE 593 Multiagent Systems

Pınar Yolum
pyolum@cmpe.boun.edu.tr

Department of
Computer Engineering
Boğaziçi University

Chapter 4 Agent Communication

Based largely on
Service-Oriented Computing: Semantics, Processes, Agents
– Munindar P. Singh and Michael N. Huhns, Wiley, 2004

Interaction and Communication

- Interactions occur when agents exist and act in close proximity:
 - resource contention, e.g., bumping into each other
- Communications are the interactions that preserve autonomy of all participants, but maybe realized through physical actions that don't:
 - Communications are inevitable when composing services.
- Communications can be realized in several ways, e.g.,
 - through shared memory
 - because of shared conventions
 - by messaging passing

3

Rationalistic Tradition

- Orientation
 - Describe the situation in terms of objects and their properties
 - Derive rules that apply to situations
 - Apply the rule to the current situation
- Literal meaning (not context-dependent)
- Hard to use in many settings
 - Example of water in the fridge (Winograd and Flores)
 - “John has never failed a student in Linguistics 265” (Winograd and Flores)

4

Speech Act Theory

- Speech act theory, developed for natural language, views communication as action
- It considers three aspects of a message:
 - *Locution*, or how it is phrased, e.g., "It is hot here" or "Turn on the air conditioner"
 - *Illocution*, or how it is meant by the sender or understood by the receiver, e.g., a request to turn on the air conditioner or an assertion about the temperature
 - *Perlocution*, or how it influences the recipient, e.g., turns on the air conditioner, opens the window, ignores the speaker

Illocution is the core aspect

5

Speech Act Theory

- Assertives: Describe the state of the world
- Directives: Attempt (in varying degrees) to make the other person do something
- Commissives: Commit the speaker (in varying degrees) to a course of actions
- Expressives: Express a psychological state.
- Declaratives: Make the content of the act match reality

6

Speech Act Theory Applied

- Classifications of illocutions motivate message types, but are typically designed for natural language
 - rely on NL syntax, e.g., they conflate directives and prohibitives
- Most research in speech act theory is about determining the agents' beliefs and intentions, e.g., how locutions map to illocutions
- For Web Services,
 - determining the message type is trivial, because it is explicitly encoded
 - determining the agents' beliefs and intentions is impossible, because the internal details of the agents are not known

7

Syntax, Semantics, Pragmatics

For message passing

- *Syntax*: requires a common language to represent information and queries, or languages that are intertranslatable
- *Semantics*: requires a structured vocabulary and a shared framework of knowledge-a shared ontology
- *Pragmatics*:
 - knowing whom to communicate with and how to find them
 - knowing how to initiate and maintain an exchange
 - knowing the effect of the communication on the recipient

8

ACL Semantics

What is the semantics of queries, requests, promises?

- *Mentalist*: each agent has a knowledge base that its messages refer to. An agent promises something if it intended to make that promise
- *Public*: semantics depends on laws, protocols, and observable behavior

Evaluation: For open systems, public semantics is appropriate, because a semantics without compliance doesn't make sense

9

Informing

How can one agent tell another agent something?

- Send the information in a message (message passing)
- Write the information in a location where the other agent is likely to look (shared memory)
- Show or demonstrate to the other agent (teaching)
- Insert or program the information directly into the other agent (master --> slave; controller --> controllee; "brain surgery")

10

Querying

How can one agent get information from another agent?

- Ask the other agent a question (message passing)
- Read a location where the other agent is likely to write something (shared memory)
- Observe the other agent (learning)
- Access the information directly from the other agent ("brain surgery")

11

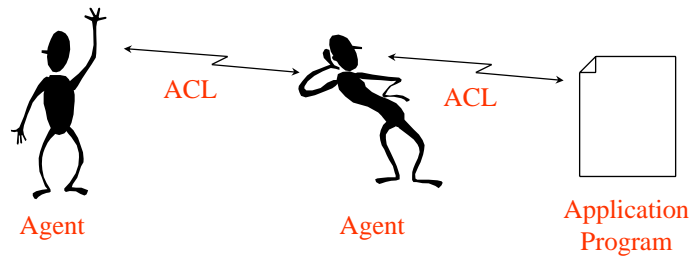
A Classification of Message Classifications

- Structure-based (syntactic)
 - distinguish messages based on grammatical forms in natural language
- Meaning-based (semantic)
 - distinguish messages based on a notion of intrinsic meaning
prohibitive is different from *directive*, despite syntactic similarity
- Use-based (pragmatic)
 - distinguish messages based on their roles in specific classes of protocols
assertion is different from *acknowledgment*

12

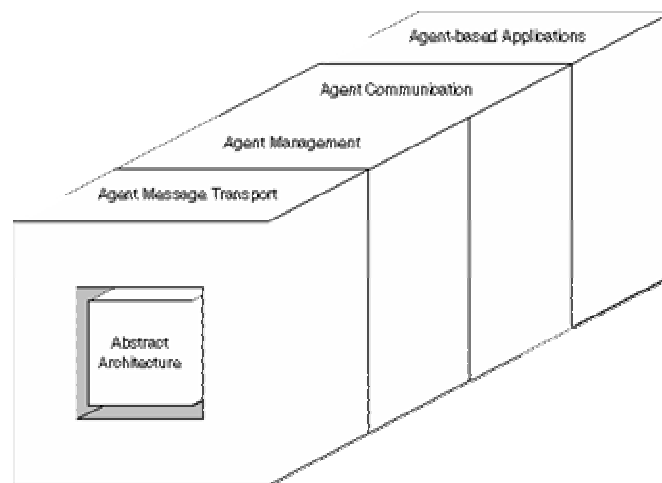
Agent Communication Languages (ACL)

- KQML: Knowledge Query and Manipulation Language
- FIPA ACL



13

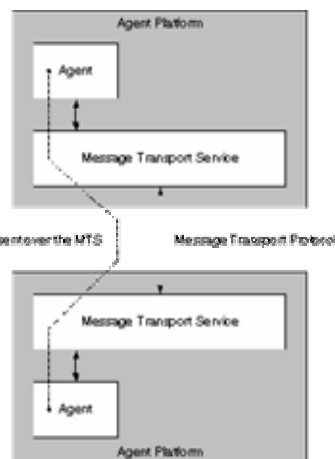
Structure of Specifications



14

Agent Message Transport

- Agent Message Transport (AMT) defines a message as an **envelope** plus a **body**. Together they handle
 - Guidelines for various transport protocols (e.g., IIOP, HTTP, WAP)
 - Message envelope representation (e.g., XML for HTTP, bit-efficient for WAP).
 - FIPA ACL representations (e.g., string encoding, XML encoding, bit-efficient encoding).



15

Ontology

- A specification of a conceptualization or a set of knowledge terms for a particular domain, including
 - The vocabulary
 - The semantic interconnections
 - Some simple rules of inference and logic
- Some representation languages for ontologies:
 - Uniform Modeling Language (UML)
 - Resource Description Framework Language Schema (RDFS)
 - Web Ontology Language (OWL)
- Some ontology editors: Protégé, Webonto, OilEd

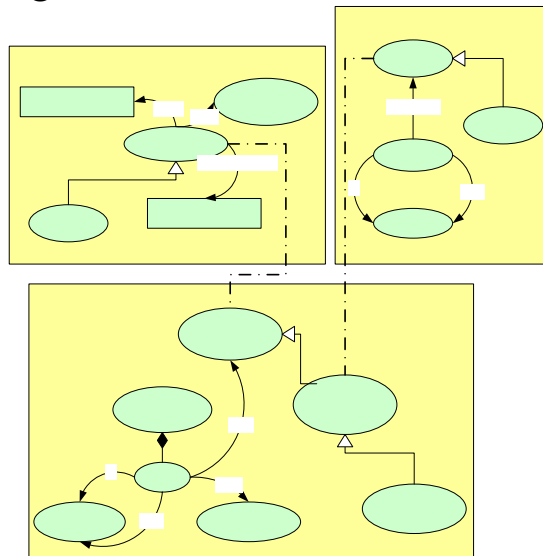
16

Common Ontologies

- A shared representation is essential to successful communication and coordination
 - For humans: physical, biological, and social world
 - For computational agents: common ontology (terms used in communication)
- Representative efforts are
 - Cyc (and Opencyc)
 - WordNet (Princeton)
 - Several *upper-level* ontologies

17

Ontologies and Articulation Axioms



18

Knowledge Representation

- Interoperability levels
 - Syntactic: parse
 - Semantic: understand
- Expressive power
- Procedural versus declarative
 - Declarative pros: enables standardization, optimization, improved productivity
 - Declarative cons: nontrivial to achieve and causes short-term loss of performance
 - Trade-offs shifted by Web to favor declarative modeling

19

Frames versus Descriptions

- Frame-based approaches are intuitive but rely on names of classes and properties to indicate meaning
 - Description logics provide a computationally rigorous means to represent meaning; difficult for people
- Managing this trade-off is a major challenge for KR

20

Mappings among Ontologies

- Term-to-term (one-to-one), e.g.,
 $\text{hookupWire}_{O_1} \equiv \text{wire}_{O_2}$
- Many-to-one, e.g.,
 $\text{solidWire}_{O_1}(x, \text{size}, \text{color}) \wedge \text{strandedWire}_{O_1}(x, \text{size}, \text{color})$
 $\equiv \text{wire}_{O_2}(x, \text{size}, \text{color}, (\text{Stranded}|\text{Solid}))$
- Many-to-many, e.g.,
 $\text{solidBlueWire}_{O_1}(x, \text{size}) \wedge$
 $\text{solidRedWire}_{O_1}(x, \text{size}) \wedge$
 $\text{strandedBlueWire}_{O_1}(x, \text{size}) \wedge$
 $\text{strandedRedWire}_{O_1}(x, \text{size})$
 \equiv
 $\text{solidWire}_{O_2}(x, \text{size}, (\text{Red}|\text{Blue})) \wedge$
 $\text{strandedWire}_{O_2}(x, \text{size}, (\text{Red}|\text{Blue}))$

21

Relations

- Hierarchies in knowledge representation
 - Inheritance (isA) relation
 - Part-whole (isPartOf) relation
- Binary relation R between S and T relates zero or more members in S to zero or more members in T
- Partial order between objects
 - Antisymmetry: If $x \prec y$ and $y \prec x$, then $x=y$
 - Transitivity: If $x \prec y$ and $y \prec z$, then $x \prec z$

22

Hierarchies

- Partially-ordered binary relations
- Taxonomy:
 - *isA* relation denotes subclasses
 - Ex: A human is a mammal
 - Antisymmetric and transitive
- Meronymy:
 - *isPartOf* relation denotes one object is a part of another object
 - Ex: A wheel is part of a car
 - Asymmetric and irreflexive

23

Modeling

- A universe of discourse (set of entities)
- Concepts that identify the entities
- Relationships among entities
 - Cardinality Constraints
 - Temporal Constraints
 - Rule Constraints
- Functions that map entities to other entities

24

Exercise: Which Conceptualization Has More Expressive Power?

- `awg22SolidBlueWire(ID5)`
- `blueWire(ID5, AWG22, Solid)`
- `solidWire(ID5, AWG22, Blue)`
- `wire(ID5, AWG22, Solid, Blue)`
- `wire(ID5)^size(ID5, AWG22)^type(ID5, solid)^color(ID5, Blue)`

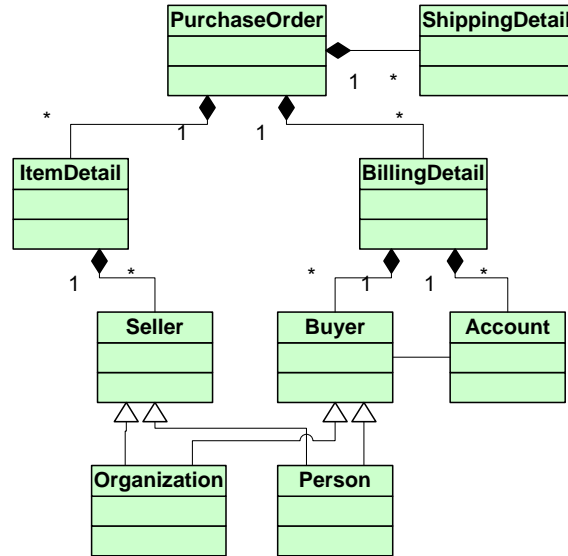
25

Conceptualization

- Guidelines
 - Concepts must have instances
 - Inference of properties based on membership
 - Nonredundancy: Subconcepts must have one different property
- Modularity
 - Don't rewrite predicates when adding properties
 - Ex: `wire(ID5, AWG22, Solid, Blue)`
- Extensibility
 - Model values as objects
 - Ex: `permanent (Blue) ^color(ID5, Blue)`

26

Unified Modeling Language (UML) for Ontologies

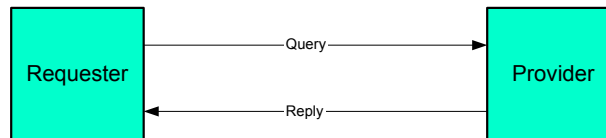


27

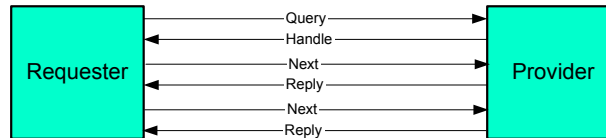
Comparison of Modeling Languages

Feature	RDF	UML	SHOE	OEM	OIL	OWL
<i>Object ID</i>	+	+	+	+	+	+
<i>Subclass</i>	+	+	+	+	+	+
<i>Not Subclass</i>	-	-	-	-	+	+
<i>Basic Typing</i>	+	i	i	+	+	+
<i>Multiple inheritance</i>	+	+	+	+	+	+
<i>Reification</i>	+	+	-	-	-	+
<i>Partitions</i>	-	-	-	-	+	+
<i>Instance Attributes</i>	+	+	+	+	+	+
<i>Class Attributes</i>	-	-	-	-	+	+
<i>Constraints: Cardinality</i>	-	+	-	-	+	+
<i>Constraints: Ordering</i>	-	i	-	-	-	-
<i>Relationships: Binary</i>	+	+	+	+	+	+
<i>Relationships: N-ary</i>	-	+	i	-	-	-

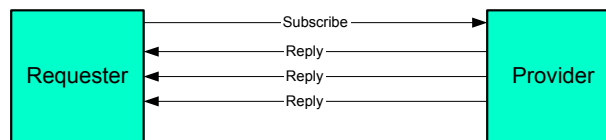
Interaction Patterns



Synchronous: a blocking query waits for an expected reply



Provider maintains state; replies sent individually when requested



Asynchronous: a nonblocking subscribe; replies sent as available

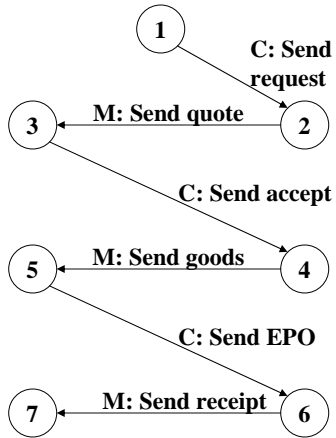
29

Commitment Protocols

- Protocols enable open systems to be constructed
- Interaction protocols expressed in terms of
 - Participants' commitments
 - Actions for performing operations on commitments (to create and manipulate them)
 - Constraints on the above, e.g., captured in temporal logic
- Examples: escrow, payment, RosettaNet (107 request-response PIPs)

30

FSM Representation of NetBill Protocol



- The merchant may start the protocol by sending a quote.
- The customer may send an accept prior to offer.
- The merchant may send the goods prior to accept.

These variations are not allowed in the FSM representation.

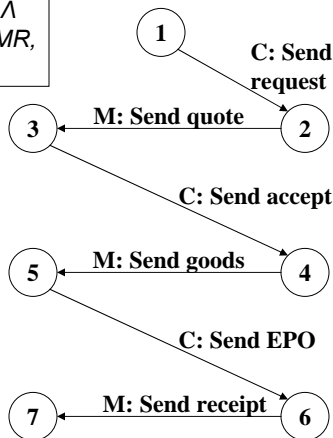
31

Definitions for Message Content

$promiseGoods(i, m): CC(MR, CT, accept(i, m), goods(i)) \wedge promiseReceipt(i, m): CC(MR, CT, pay(m), receipt(i))$

$promiseReceipt(i, m): CC(MR, CT, pay(m), receipt(i))$

$receipt(i):$ the merchant has delivered the receipt.



$request(i):$ the customer has requested a quote.

$accept(i, m): CC(CT, MR, goods(i), pay(m))$

$pay(m):$ the customer has paid the agreed amount.

32

Event Calculus

Formalism to reason about events.

- Based on first-order logic
- Many sorted (Events, Time, Fluents)
- $Happens(a, t)$: event a happens at time t .
- $HoldsAt(f, t)$: fluent f holds at time t .
- $Initiates(a, f, t)$: fluent f holds after event a at time t .
- $Terminates(a, f, t)$: fluent f does not hold after event a at time t .

33

Message Content

Initiates Clauses

- $Initiates(sendRequest(i), request(i), t)$
- $Initiates(sendQuote(i, m), promiseGoods(i, m), t)$
- $Initiates(sendQuote(i, m), promiseReceipt(i, m), t)$
- $Initiates(sendAccept(i, m), accept(i, m), t)$
- $Initiates(sendGoods(i, m), goods(i, m), t)$
- $Initiates(sendGoods(i, m), promiseReceipt(i, m), t)$
- $Initiates(sendEPO(i, m), pay(m), t) \leftarrow HoldsAt(goods(i), t)$
- $Initiates(sendReceipt(i, m), receipt(i), t) \leftarrow HoldsAt(pay(m), t)$

Terminates Clauses

- $Terminates(sendQuote(i, m), request(i), t) \leftarrow HoldsAt(request(i), t)$

34

Commitments

- A commitment is an obligation from one party to another to bring about a condition.
- A unilateral commitment
 - $C(x, y, p)$: x commits to y to bring about p .
 - $C(\text{merchant}, \text{customer}, \text{receipt})$
- A conditional commitment
 - $CC(x, y, p, q)$ is a conditional commitment: x commits to y to bring about q if p is brought out first.
 - $CC(\text{merchant}, \text{customer}, \text{pay}, \text{receipt})$

35

Commitment Operations

1. $Create(e, x, c)$: Establishes the commitment c .
(I will pay 5YTL to Ali)
 $\{Happens(e, t) \wedge Initiates(e, C(x,y,p), t)\}$
2. $Discharge(e, x, c)$: Resolves the commitment c .
(I paid 5YTL to Ali)
 $\{Happens(e, t) \wedge Initiates(e, p, t)\}$
3. $Cancel(e, x, c)$: Cancels the commitment c .
(I cancel my commitment to pay 5YTL to Ali)
 $\{Happens(e, t) \wedge Terminates(e, C(x,y,p), t)\}$

36

Commitment Operations (2)

4. *Release*(e, x, c) : Releases the debtor from the commitment c .

Release($e, x, C(x,y,p)$):

$\{Happens(e, t) \wedge Terminates(e, C(x,y,p), t)\}$

5. *Assign*(e, y, z, c) : Assigns a new creditor, z , to an existing commitment c .

Assign($e, y, z, C(x,y,p)$): $\{Happens(e, t) \wedge$

$Terminates(e, C(x,y,p), t) \wedge Initiates(e, C(x,z,p), t)\}$

6. *Delegate*(e, x, z, c) : Delegates a new debtor, z , to an existing commitment c .

Delegate($e, x, z, C(x,y,p)$): $\{Happens(e, t) \wedge$

$Terminates(e, C(x,y,p), t) \wedge Initiates(e, C(z,y,p), t)\}$

37

Reasoning Rules

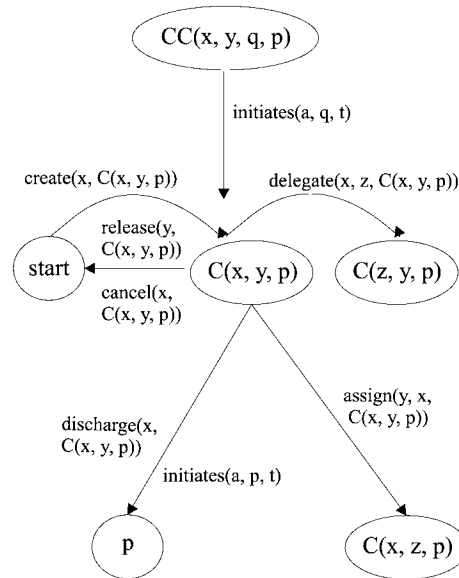
1. **$C(x,y,p)$ ceases to exist when the proposition p becomes true.**

2. **$CC(x,y,p,q)$ ceases to exist when the proposition p becomes true, but $C(x,y,q)$ is created.**

- $CC(\text{merchant}, \text{customer}, \text{paid}, \text{receipt})$
- Customer makes "paid" true
- $C(\text{merchant}, \text{customer}, \text{receipt})$

38

Commitment Manipulations



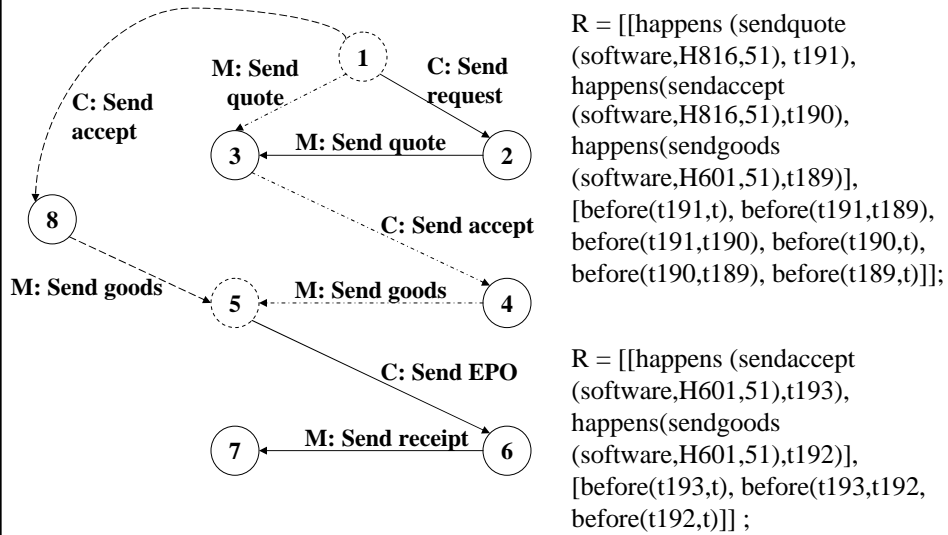
39

Commitment Protocol

- *A protocol specification*
 - contains a set of *actions* and the commitments and propositions they initiate.
 - does not specify any final states.
 - does not explicitly state the transitions; transitions follow from operations and reasoning rules on commitments.
- *A protocol run*
 - specifies the paths between states
 - lists which actions happen and their ordering
 - is complete if all unilateral commitments are resolved at the end.

40

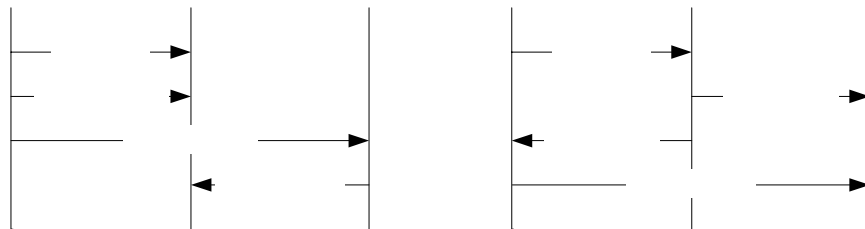
Sample Protocol Runs



41

Message Patterns for Commitment Operations

Ensure that information about commitment operations flows to the right parties, to enable local decisions



42

Compliance with Protocols

In an open environment, agents are contributed by different vendors and serve different interests

- How can an application check if the agents *comply* with specified protocols?
 - Coordination aspects: traditional techniques
 - Commitment aspects: representations of the agents' commitments in temporal logic
- Commitment protocols are specified in terms of
 - Main roles and sphere of commitment
 - Roles essential for coordination
 - Domain-specific propositions and actions

43

Verifying Compliance

- Specification
 - models based on *potential causality*
 - commitments based on branching-time TL
- Run-time Verification
 - respects design autonomy
 - uses TL model-checking
 - local verification based on observed messages

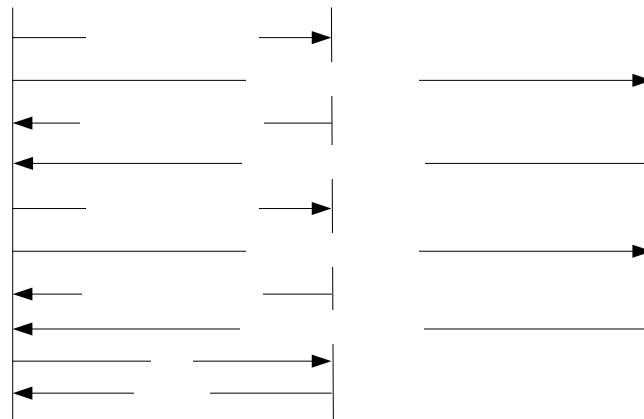
44

Run-Time Compliance Checking

- An agent can keep track of
 - its pending commitments
 - commitments made by others that are not satisfied
- It uses this local model to see if a commitment has been violated
- An agent who benefits from a commitment can always determine if it was violated

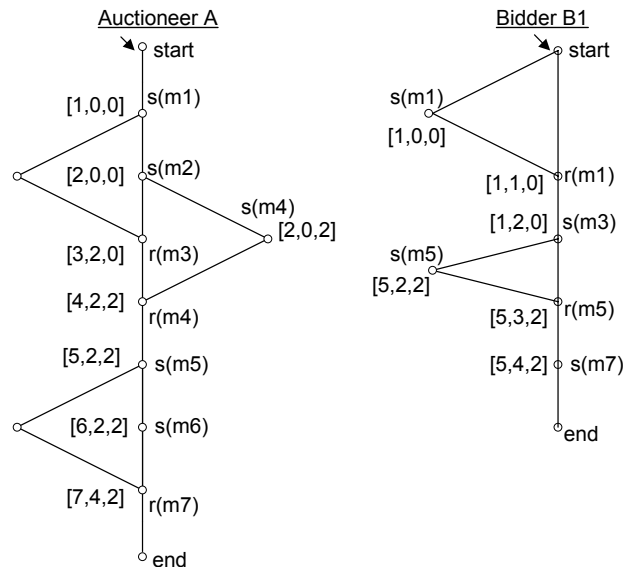
45

Fish-Market Sample Execution



46

Fish-Market Local Observations



47

Fish-Market Compliance

- Auctioneer can verify if the bidders comply
- An individual bidder cannot verify if the auctioneer complies
- If bidders pool their observations, then they can verify if the auctioneer complies
- Asymmetry indicates need for third party

48