Agent Oriented Programming

ASSIGNMENT
due at class on October 14

- Write max 2 pages describing the functionality of the modules in your architecture for a simple example from the topic you chose for your project. (Please briefly describe the example as well).
- Be prepared to present this example in the beginning of our October 14 class

Lecture Outline

- Agent Oriented Programming
- Generic AOP Interpreter
- Formal Semantics of an AOP language
- Agent-0

References

- "Agent-Oriented Programming" – Yoav Shoam (next document posted on the website)

Agent Oriented Programming

- Computational framework proposed by Yoav Shoham
- Agents have *mental states*
  - Beliefs, Decisions, Capabilities, Obligations
- AOP – *framework* + formal mechanisms for assigning mental states to agents
- **Operators** and a precisely defined **syntax** for beliefs, knowledge etc.

Agent Oriented Programming

- Fixing the mental state of agents
- Place **constraints** on the mental states
- Perform computations in an agent-oriented fashion
- Agent computations are based on **speech-act theory**
  - Computations: *inform, request, offer, reject, assist etc.*

OOP vs AOP

	OOP	AOP
Basic Unit	Object	Agent
Parameters defining state of basic unit	Unconstrained	Beliefs, commitments, capabilities, choices…
Process of computation	Message passing and response methods	Message passing and response methods
Types of messages	Unconstrained	Inform, request, offer, promise, decline…
Constraints on messages	None	Honesty, consistency…

AOP Framework

- Three main components
  - Formal language with clear syntax and semantics for describing mental state
  - Interpreted language in which to define and program agents
  - An “agentifier” to convert neutral devices into programmable agents
- Paper really only addresses the first two components

Mental States and Properties

- **Decisions or Choices**
  - Determine agent actions
  - Logically constrained by Beliefs and other Decisions
- **Beliefs**
  - State of the world
  - Mental state of other agents
  - Capabilities of self and other agents
- **Capability**

Mental States and Properties

- Obligation or Commitment rather than choice or decision
- Decision = Obligation to oneself

- Summary – 3 mental states
  - Obligations
  - Beliefs
  - Capabilities

Mental State Language

- Language similar to logic languages (LISP, ProLog, etc…)
  - In fact implementation in LISP

Formal Language Semantics

- A language for belief, obligation and capability
- Time
- Action
- Belief
- Obligation
- Decision
- Capability
- Internal Consistency
- Good Faith

Time

- Basic to all mental categories
- Simple point-based temporal language
  - holding(robot, cup)^{t}
  - **Robot is holding the cup at time t**

Action

- Actions take place at different points
- Have certain effects
- Occurrence of an action is represented as a fact
  - `raise-arm(robot)`^{t}

Belief

- Modal Operator B to represent belief
- An agent believes things *at certain times* and *about certain times*
  - $B^t_a\varphi$
  - a – agent
  - t – time term
  - φ - recursively defined sentence
- Example
  - $B^3_a B^{10}_b like(a,b)^7$
  - At time 3, agent a believes that at time 10 agent b will believe that at time 7 a liked b

Obligation

- Operator OBL represents obligation
  - $OBL^{t}_{a,b}\varphi$
  - At time t agent a is obligated to agent b about φ

Decision

- Central to the notion of agents
- Generally is considered a ***primitive notion***
- Decision is defined in terms of Obligation
- Decision is defined as an Obligation to Self
  - $DEC^{t}_{a} =_{def} OBL^{t}_{a,a}\varphi$

Capability

- An agent can only do what it is capable of
- Highly debatable – should capability be a mental term ?
  - $CAN^{t}_{a}\varphi$
  - At time t agent a is capable of φ
- Constraints on capabilities
- Example
  - $CAN^{5}_{robot}open(door)^{8}$
  - At time 5: Robot has the capability of opening the door at time 8 – but not at time 6

Capability

- ABLE – Immediate version of CAN
  - $ABLE_a\varphi =_{def} CAN^{time(\varphi)}{}_a\varphi$
  - time(φ) is the outermost time occurring in φ
- Example
  - $ABLE_{robot}open(door)^5 = CAN^5{}_{robot}open(door)^5$
  - Robot is able to open the door at 5 – irrespective of what the current time is

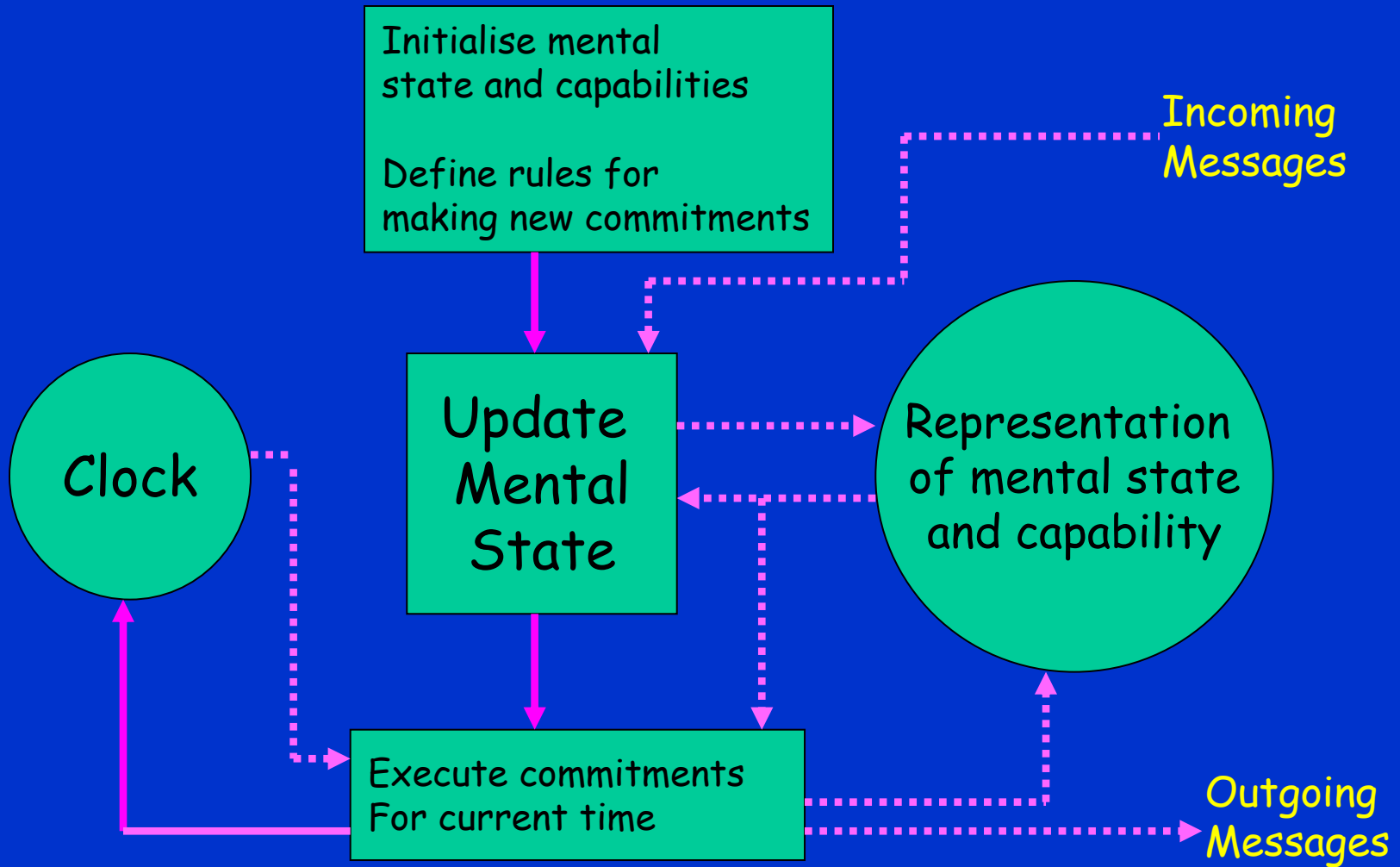
Generic Agent Interpreter

- Agent Programs
  - Control evolution of agent's mental state
- Can be done by logic programming
- Multi-modal logic programming is not easy ☺
- A standard operationally defined programming language
- Support for data structures to represent belief, capabilities etc. and operations on them

Generic Agent Interpreter

- Each agent iterates the following steps at regular intervals
  - Read current messages and update mental state
  - Execute commitments for the current time

Generic Agent Interpreter



Generic Agent Interpreter

- Message passing infrastructure
- Clock
  - Initiate iterations of the two-step process
  - Length of these intervals – "timegrain" – should be settable
  - Variable "now" - current time
  - Synchronisation issues – when multiple machines are involved

AGENT-0

- Simple AOP language and its interpreter
- Proof of concept developed to support the framework
  - Very simplistic
  - Fact statements are atomic objective statements and can not contain conjunction or disjunction

AGENT-0 Language

- Actions
  - May be Private or Communicative
  - May be Conditional or Unconditional

  - Private Actions
    - May or may not involve I/O
  - Communicative Actions
    - Always involve I/O
    - Are uniform and common across all agents

Actions

- Private action
  - $(DO \quad t \quad p-action)$
- Inform action
  - $(INFORM \quad t \quad a \quad fact)$
- Request action
  - $(REQUEST \quad t \quad a \quad action)$
- Unrequest action
  - $(UNREQUEST \quad t \quad a \quad action)$
- Refrain action $(REFRAIN \quad action)$

WHERE

t – time point
a – agent name
p-action – private action
fact – fact statement
action – any action statement

Conditional Actions

- Conditional actions are triggered by one condition – a mental condition
  - Mental condition: a logical combination of mental patterns
  - Mental pattern is one of two items:
    - $(B \quad fact)$ or $((OBL \quad a)action)$
  - Syntax for Conditional Action
    - $(IF \quad mntlcond \quad action)$
  - Example
    - $(IF(B(t'(employee \quad smith \quad acme)))(INFORM \quad t \quad a(t'(employee \quad smith \quad acme))))$
    - if at time t you believe that at time t' smith is an employee of acme, then at time t inform agent a of that fact

Mental Conditions

- May contain logical connectors
  - AND, OR, & NOT
- Example
  - $(REQUEST \quad t \quad b \quad (IF \quad (B, fact) \quad (INFORM \quad t+1 \quad a \quad fact)))$
    $(REQUEST \quad t \quad b \quad (IF \quad (B \quad (NOT \quad fact)) \quad (INFORM \quad t+1 \quad a \quad (NOT \quad fact))))$
    $(REQUEST \quad t \quad b \quad (IF \quad (NOT \quad (BW \quad fact)) \quad (INFORM \quad t+1 \quad a \quad (NOT \quad (BW \quad fact)))))$
    - BW is "believe whether" defined as $(t \quad (BW \quad a \quad p)) \equiv (t \quad (B \quad a \quad p)) \vee (t \quad (B \quad a \quad (NOT \quad p)))$
  - This is example of a query about another agents belief about a particular fact

Variables

- Language syntax has support for variables denoted by a *?* Prefix
  - (*IF* (*NOT* ((*OBL* ? *x*) (*REFRAIN* *dance*))) *dance*)
  - Variables are scoped locally until the first NOT operator (or the entire statement in the absence of such a connector)
    - Can use universally quantified variables which are prefixed with *?!* and are always scoped throughout the entire formula

Commitments (Obligations)

- Commitments rules are dictated in agent design
  - Commitments are made when both mental conditions and message conditions are met
  - Message Conditions
    - Logical combination of Message Patterns
    - *(From Type Content)*
    - *Example*
      - *(AND (a REQUEST (DO t walk)) (NOT (?x REQUEST (DO t chew-gum))))*

Commitment Rules

- Syntax
  - *(COMMIT msgcond mntlcond (agent action)*)*
    - * denotes zero or more times
  - Example
    - *(COMMIT (?a REQUEST ?action) (B (now (myfriend ?a))) (?a ?action))*

Example Implementation

- Scenario (Air Travel Booking)
  - March
    - P to C: Please inform me what flights you have from San Francisco to New York on April 18.
    - C to P: Flight #354 departs at 08:30, flight #293 departs at 10:00, flight #441 departs at noon . . . .
    - P to C: Please book me on #354.
    - C to P: That is sold out.
    - P to C: Please book me on #293.
    - C to P: That is confirmed; your reservation number is 112358.
    - P to C: Please book me also on #441.
    - C to P: That conflicts with #293: I am not allowed to double book a passenger.
    - P to C: Please get permission to do so.
    - C to S: I request permission for the following double booking: ...
    - S to C: Permission denied.
    - C to P: Sorry, I cannot get approval.
  - April 18, at the airport
    - P to C: My name is P; I have a reservation for flight 293.
    - C to P: Here is your boarding pass.

AGENT-0 Implementation

- Use of macro definitions
  - *(issue_bp pass flightnum time) → (IF (AND (B ((- time h) (present pass))) (B (time (flight ?from ?to flightnum)))) (DO time-h (physical_issue_bp pass flightnum time)))*
    - where :
      - *h* = one hour
      - *physical_issue_bp* = private action to physically issue boarding pass

AGENT-0 Implementation (Cont)

- *(query_which t asker askee q) → (REQUEST t askee (IF (B q) (INFORM (+ t i) asker q)))*
  - query_which requests only a positive answer; if q contains a universally-quantified variable then query_which requests to be informed of all instances of the answer to the query q
- *(query_whether t asker askee q) → (REQUEST t askee (IF (B q) (INFORM (+ t I) asker q))) (REQUEST t askee (IF (B (NOT q)) (INFORM (+ t i) asker (NOT q))))*
  - query_whether expects either a confirmation or a disconfirmation of a fact

AGENT-0 Implementation (Cont)

- Capabilities Assignments
  - *((issue_bp ?a ?flight ?time) true)*
    - If it is time to issue a boarding pass then possess the ability to do so
  - *((DO ?time (update_remaining_seats ?timel ?flight_number ?additional_seats)) (B (?time (remaining_seats ?timel ?flight_number ?current_seats))))*
    - *update_remaining_seat* = is a private action which changes the belief regarding remaining_seats
    - Update the current number of seats for a particular flight

AGENT-0 Implementation (Cont)

- Commitment Rules
  - *(COMMIT (?pass REQUEST (IF (B,?p) (INFORM ?t ?pass ?p))) true ?pass (IF (B,?p) (INFORM ?t ?pass ?p)))*
    - Respond to requests for information about flight availability
  - *(COMMIT (?cust REQUEST (issue_bp ?pass ?flight ?time)) (AND (B (?time (remaining_seats ?flight ?n))) (?n>O) (NOT ((CMT ?anyone) (issue_bp ?pass ?anyflight ?time)))) (myself (DO (+ now 1) (update_remaining_seats ?time ?flight -I))) (?cust (issue_bp ?pass ?flight ?time)))*
    - Respond to requests to issue a boarding pass for a flight

AGENT-0 Implementation (Cont)

- That gives the complete implementation for the airline agent from the previous example
  - Customer implementation not detailed
  - Supervisor implementation not detailed

AGENT-0 Implementation

agent	action
smith	(query_which imarch/l:O0 smith airline (18april/?!time (flight sf ny,?!num)))
airline	(INFORM imarch/2:00 smith (18april/8:30 (flight sf ny #354)))
airline	(INFORM imarch/2:00 smith (18april/lO:O0 (flight sf ny #293)))
airline	(INFORM imarch/2:00 smith (18april/ ...
smith	(REQUEST imarch/3:00 airline (issue_bp smith #354 18april/8:30))
smith	(query_whether imarch/4:00 smith airline ((CMT smith) (issue_bp smith #354 18april/8:30)))
airline	(INFORM imarch/5:00 smith (NOT ((CMT smith) (issue_bp smith #354 18april/8:30))))
smith	(REQUEST imarch/6:00 airline (issue_bp smith #293 18april/lO:O0))
smith	(query_whether imarch/7:O0 smith airline ((CMT smith) (issue_bp smith #293 18april/lO:O0)))
airline	(INFORM imarch/8:00 smith ((CMT smith) (issue_bp smith #293 18april/lO:O0)))
…	
smith	(INFORM 18april/9:00 airline (present smith))
airline	(DO 18april/9:00 (issue_bp smith #293 18april/lO:O0))

New Work

- Many follow-on works that expand on the AOP proposed by Shoham
  - Agent-Oriented Programming: A Practical Evaluation by David Parks (1997)

- JADE (Java Agent Development Framework)
  - Utilizes many of the notions put forth by Shoham