Slide 1

**North American Summer School in Logic Language and Information, June 2003**

**Algorithmic Verification for Epistemic Logic**

**Ron van der Meyden**

**University of New South Wales/National ICT Australia**

Slide 2

**Lecture 4**

**Examples and MCK System Demonstration**



Slide 3

*n* agents

$ACT_i$ a set of actions for each agent $i = 1 \dots n$

*joint actions*: $ACT = ACT_1 \times \dots \times ACT_n$.

Slide 4

A *finite interpreted environment for n agents* is a tuple $E$ of the form $\langle S_e, I_e, \tau, O, \pi_e, \alpha \rangle$ where the components are as follows:

1. $S_e$ is a finite set of *states of the environment*.
2. $I_e$ is a subset of $S_e$, the *initial states* of the environment.
3. $\tau$ is a function mapping joint actions $\mathbf{a} \in ACT$ to state transition relations $\tau(\mathbf{a}) \subseteq S_e \times S_e$.
4. $O = \langle O_1, \dots, O_n \rangle$ is a tuple of *observation functions* $O_i : S_e \rightarrow Obs$
5. $\pi_e : S_e \times Prop \rightarrow \{0, 1\}$ is an *interpretation*,
6. $\alpha \subseteq S_e$ is a Büchiacceptance condition

Slide 5

A *protocol* for agent $i$ is a function $P_i : S_e^+ \longrightarrow \mathcal{P}(ACT_i)$.

A *joint protocol* $\mathbf{P}$ is a tuple $\langle P_1, \ldots, P_n \rangle$, where each $P_i$ is a protocol for agent $i$.

A *run* of a joint protocol $\mathbf{P}$ in an environment $E$ is an *infinite* sequence $\varepsilon = s_0 s_1 \ldots$ of states of $E$ such that

1. $s_0 \in I_e$,
2. for all $k \geq 0$, there exists a joint action $\mathbf{a} = \langle a_1, \ldots, a_n \rangle$ such that $(s_k, s_{k+1}) \in \tau(\mathbf{a})$ and $a_i \in P_i(r[0 \ldots k])$
3. some $s \in \alpha$ occurs infinitely often.

Slide 6

### Local state defined wrt a view

Let $\varepsilon$ be a run of $\mathbf{P}$ in $E$. A *view* associates a local state with each agent at each point of time, determining a mapping $\varepsilon^v : \mathbf{N} \rightarrow L^n \times S_e$

In all cases $\varepsilon_e^v(m) = \varepsilon(m)$

Examples:

1. The *observational* view: $\varepsilon_i^{\texttt{obs}}(m) = O_i(\varepsilon(m)))$
2. The *clock* view: $\varepsilon_i^{\texttt{obs}}(m) = (m, O_i(\varepsilon(m)))$
3. The *synchronous perfect recall* view: $\varepsilon_i^{\texttt{spr}}(m) = O_i(\varepsilon(0)) \ldots O_i(\varepsilon(m))$



Slide 7

### System Generated by an Environment wrt a View

Let $v$ be a view of an environment $E$. Define $I^v(\mathbf{P}, E) = (\mathcal{R}^v(\mathbf{P}, E), \pi)$ to be the interpreted system with

1. $\mathcal{R}^v(\mathbf{P}, E)$ the set of $\varepsilon^v$ such that $\varepsilon$ is a run of $\mathbf{P}$ in $E$.
2. $\pi(r(m), p) = \pi_e(r_e(m), p)$ for all $r \in \mathcal{R}^v(\mathbf{P}, E)$, $p \in$ *Prop*

Slide 8

| Operator | Description |
|---|---|
| $\mathbf{AX}\ f$ | $f$ in all next states. |
| $\mathbf{EX}\ f$ | $f$ in at least one next state. |
| $\mathbf{A}[f\ \mathbf{U}\ g]$ | on all paths, $f$ until $g$. |
| $\mathbf{E}[f\ \mathbf{U}\ g]$ | on at least one path, $f$ until $g$. |
| $\mathbf{AF}\ f$ | On all paths, in some future state, $f$. |
| $\mathbf{EF}\ f$ | On at least one path, in some future state, $f$. |
| $\mathbf{AG}\ f$ | On all paths, in all future states, $f$. |
| $\mathbf{EG}\ f$ | On at least one path, in all future states, $f$. |

**Slide 9**

| Operator | Description |
|---|---|
| $\mathbf{F}\ f$ | eventually $f$. |
| $\mathbf{G}\ f$ | always $f$. |
| $f\ \mathbf{U}\ g$ | $f$ until $g$. |
| $\mathbf{X}\ f$ | $f$ in the next state. |
| $\mathbf{X}\ \texttt{int}\ f$ | $f$ in `int` steps. |

**Slide 10**

| Operator | Description |
|---|---|
| $\mathbf{Knows}\ i\ f$ | agent $i$ knows $f$ |
| $\mathbf{CK}\ f$ | $f$ is common knowledge to all agents |
| $\mathbf{CK}\ \{i_1,\ldots,i_n\}\ f$ | $f$ is common knowledge to $i_1,\ldots i_k$ |

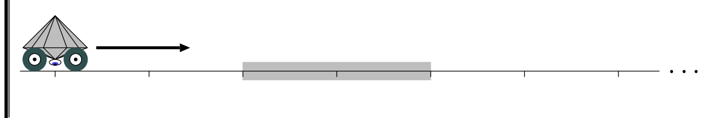


**Slide 11**

| Language | Observational | Clock | Sync. Perfect Recall |
|---|---|---|---|
| leading $X^n$ | **spec_obs** | **spec_clock** | **spec_pr** |
| CTL | **spec_obs** | | |
| LTL | **spec_obs_ltl** | | |

**Slide 12**

Brafman, Latombe, Moses, Shoham: Applications of a logic of knowledge to motion planning under uncertainty. JACM 1997



- Sensor ∈ [position-1, position+1]
- Robot moves under control of the environment, *at most* one step per unit time.

Slide 13

A knowledge-based program:

```
wait until Know(position in Goal);
halt.
```

Implementations when `Goal` = $\{2,3,4\}$ and agent's view = `Sensor`:

```
I1: wait until Sensor = 3;
    halt.

I2: wait until Sensor in {3,4,5};
   halt.
```

Slide 14

**Dining Cryptographers**

David Chaum, J. Cryptology 1988:

Three cryptographers are sitting down to dinner at their favorite three-star restaurant. Their waiter informs them that arragements have been made with the maitre d'hotel for the bill to be paid anonymously. One of the cryptographers might be paying for the dinner, or it might have been the NSA (US National Security Agency). The three cryptographers respect each other's right to make an anonymous payment, but they wonder if the NSA is paying. They resolve their uncertainty fairly by carrying out the following protocol:



Slide 15

Assumption: at most one cryptographer is paying.

1. Each cryptographer flips an umbiased coin behind his menu, between him and the cryptographer to his right, so that only the two of them can see the outcome
2. Each cryptographer then states aloud whether the two coins that he can see - the one he flipped and the one his left-hand neighbour flipped – fell on the same side or different sides

2e. If one of the cryptographers is the payer, he states the opposite of what he sees.

3. An odd number of differences uttered at the table indicates that NSA is paying, an even number of differences indicates that a cryptographer is paying.

Slide 16

If a cryptographer is paying neither of the other two learns anything from the utterances about which cryptographer it is.