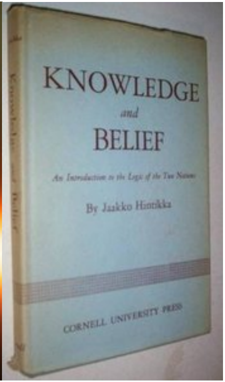
Protocol Design via Epistemic Analysis

Yoram Moses
Technion

Dedication



This talk is dedicated to the memory of Jaakko Hintikka

1929 – 2015

Themes

Workshop: Synthesizing Epistemic Protocols

(DEL)

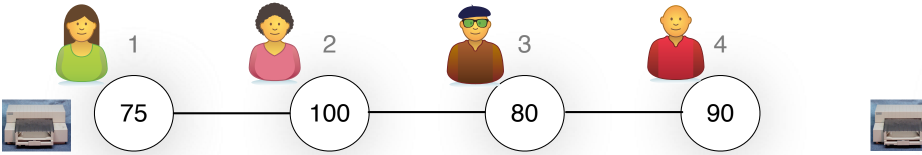
This talk: Using Epistemics for Protocol Design

(in the runs & systems model)

Plan of the talk

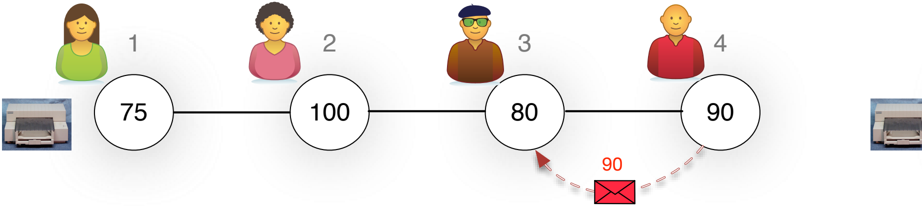
- Motivating knowledge in distributed computing by example
- Modeling knowledge in distributed computing
- Relating Knowledge and Action: Knowledge of Preconditions (**K*o*P**)
- Using **K*o*P** to derive an efficient protocol

Example: Computing the Maximum (CtM)



- Each node i has an initial value v_i
- Agent 1 must print the maximal value
- After receiving "$v_2 = 100$," can 1 act?
- No — she may lack the necessary knowledge

Collecting Values



- Collecting all values is not necessary
- Collecting all values is not sufficient
- Alice may not know how many they are

What is CTM about if not collecting values?

Knowing that $Max = c$ is a necessary condition necessary and sufficient for printing c.

This knowledge depends on various parameters:

- The agents' protocol
- The possible initial values
- The network topology
- Timing guarantees re: communication, synchrony, activation
- Reliability, . . .

Needing to know the maximum is an instance of a general principle:

The **Knowledge of Preconditions** Principle (**K*o*P**)

If φ must be true when i performs α

Then $K_i\varphi$ must be true when i performs α

This is a fundamental theorem of multi-agent systems $\Longrightarrow$ Standard specifications are epistemic

A Theory of Knowledge in Distributed Systems

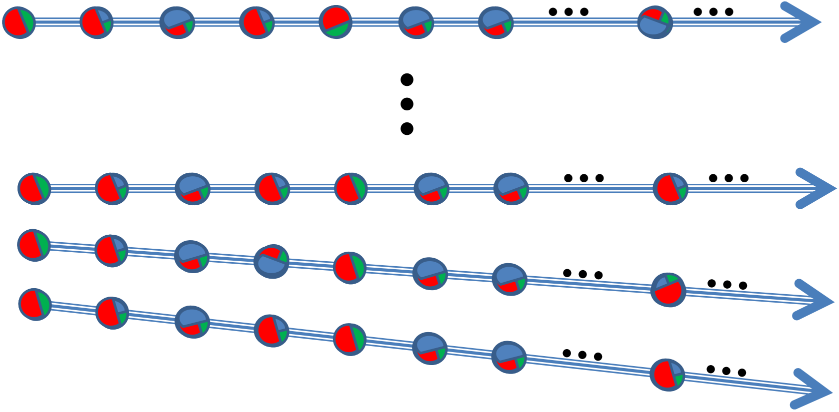
A three decades old theory of knowledge is based on

- Halpern and M. [1984]
- Parikh and Ramanujam [1985]
- Chandy and Misra [1986]
- Fagin *et al.* [1995], Reasoning about Knowledge

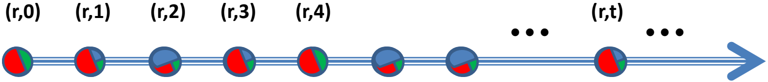
The Runs & Systems Model [Fagin *et al.* 1995]

- A global state is a "snapshot" of the whole system at an instant. $\mathcal{G}$ is the application-dependent set of global states.
- A run is a sequence $r : \mathbb{N} \to \mathcal{G}$ of global states.
- A system is a set R of runs.

Modeling: A System



Runs and points



The role of "possible worlds" is played by points $(r, t) \in R \times \mathbb{N} \triangleq \mathrm{Pts}(R)$.

A Logic of Knowledge

Starting from a set Φ of primitive propositions, define $\mathcal{L}_n^K = \mathcal{L}_n^K(\Phi)$ by

$$\varphi \quad := \quad p \in \Phi \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_1\varphi \mid \cdots \mid K_n\varphi$$

Given an interpretation $\pi : \Phi \times \mathrm{Pts}(R) \to \{\mathrm{True}, \mathrm{False}\}$

$(R, r, t) \models p$, for $p \in \Phi$, iff $\pi(p, r, t) = \mathrm{True}$.

$(R, r, t) \models \neg\varphi$ iff $(R, r, t) \not\models \varphi$

$(R, r, t) \models \varphi \wedge \psi$ iff both $(R, r, t) \models \varphi$ and $(R, r, t) \models \psi$.

Defining Knowledge

Assumption:

Each global state $r(t)$ determines a *local state* $r_i(t)$ for every agent i.

$$(\boldsymbol{R}, \boldsymbol{r}, \boldsymbol{t}) \vDash \boldsymbol{K_i}\psi \quad \text{iff} \quad (R, r', t') \vDash \psi \text{ for all points } (r', t') \text{ of } R$$
$$\text{such that } r_i(t) = r'_i(t').$$

Comments:

An agent's information is identified with its local state.

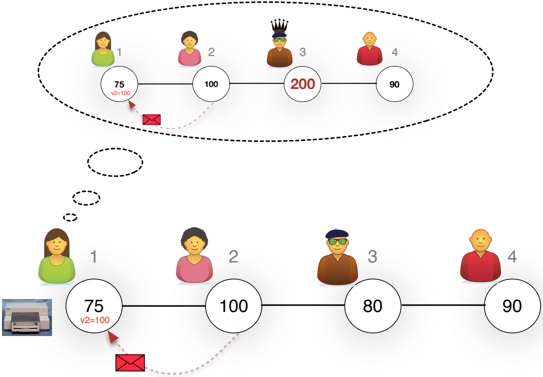
$K_i\psi$ holds if ψ is guaranteed to hold in R given i's local state.

The knowledge operator K_i is an ***S5*** modal operator.

Knowing the Maximum

$(r', 1) \;=$

$(r, 1) \;=$



- $(R, r, 1) \models K_1(Max \geq 100)$, but
- $(R, r, 1) \models \neg K_1(Max = 100)$, because
  $(R, r', 1) \models Max \neq 100$ and $r_1(1) = r'_1(1)$

Protocol + Context = System

In applications, systems have the form $R = R(P, \gamma)$:

$$R(P, \gamma) \quad = \quad \{r | r \text{ is a run of protocol } P \text{ in context } \gamma\}$$

where

- $P = (P_1, \ldots, P_n)$ is a protocol for the agents
- The context $\gamma = (\mathcal{G}_0, P_e, \tau, \Psi)$ describes the model:
  - $\mathcal{G}_0$ is a set if initial states;
  - P_e is a protocol for the environment;
  - τ is a transition function;
  - Ψ determines reliability and fairness conditions.

Necessary Conditions for Actions (aka "preconditions")

$Max = c$ is a necessary condition for $\texttt{print}_1(c)$ in $\textsc{CtM}$.

Definition

ψ is a necessary condition for $\texttt{does}_i(\alpha)$ in R if

$$(R, r, t) \models \texttt{does}_i(\alpha) \Rightarrow \psi \quad \text{for all } (r, t) \in \mathrm{Pts}(R).$$

Specifications impose necessary conditions:

- Dispensing cash at an ATM requires "customer has credit"
- Entering the critical section in Mutual Exclusion requires "critical section is empty"
- Deciding 1 in Consensus requires "no correct process will ever decide 0 in this run"

Knowledge of Preconditions

Definition

α is a ***conscious action*** for i in R if

$(R, r, t) \models \mathtt{does}_i(\alpha)$ & $r'_i(t') = r_i(t)$ implies $(R, r', t') \models \mathtt{does}_i(\alpha)$

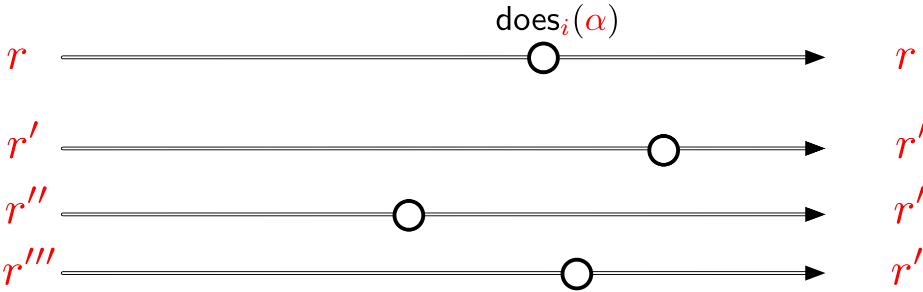
Theorem (KoP)

Suppose that α is a conscious action for i in the system R.

If φ is a necessary condition for $\mathtt{does}_i(\alpha)$ in R, then

$K_i\varphi$ is a necessary condition for $\mathtt{does}_i(\alpha)$ in R.

Proof of **KoP**



α is a conscious action for i

φ is a necessary condition for $\texttt{does}_i(\alpha)$

KoP Converts Specs into Epistemic Specs

Specifications induce epistemic conditions:

- CTM: $K_1(Max = c)$
- Dispensing cash at the ATM: K_{atm}(customer has credit)
- Mutual Exclusion: K_i(critical section is empty)
- Consensus: K_i(no correct process will ever decide 0)
- Betting on Camelot in the Royal Ascot requires K_i(Camelot will win).

An Application: Distributed Consensus

We consider

- a complete communication graph with n nodes
- each starts with a binary initial value $v_i \in \{0, 1\}$
- a discrete global clock
- synchronous round-based message passing
- up to $t < n$ crash failures

The scheduler chooses

- the initial values — a vector in $\{0, 1\}^*$
- the failure pattern — who crashes, when, and in what form.

We call this choice an adversary $\beta = (\vec{v}, fp)$.

Consensus

A consensus protocol must guarantee:

Decision: Every correct process decides on some value

Validity: If $v_i = c$ for all i then nobody decides $1 - c$, for $c \in \{0, 1\}$

Agreement: All correct processes decide on the same value

A process is correct in a run if it does not crash.

$t+1$ round Lower Bound

Theorem (Dolev-Strong '82, Fischer-Lynch '82)

Every consensus protocol in this model requires at least $t+1$ *rounds to decide in its worst-case run.*

Consensus Protocols I

We focus on full-information protocols (fip's): Each non-crashed process broadcasts its state in every round.

Protocol P_1 (for undecided process i):

if $\texttt{time} = t+1$ & $K_i \exists 0$ **then** $\mathrm{decide_i}(0)$

elseif $\texttt{time} = t+1$ & $\neg K_i \exists 0$ **then** $\mathrm{decide_i}(1)$

Optimal: All decisions at time $t+1$

Consensus Protocols II

A better protocol:

Protocol P_2 (for undecided process i):

if $K_i \exists 0$ **then** $\mathrm{decide_i}(0)$

elseif $\mathtt{time} = t+1$ & $\neg K_i \exists 0$ **then** $\mathrm{decide_i}(1)$

Optimal: All decisions by time $t+1$

Consensus Protocols III

Even better (Dolev, Reischuck, Strong '83) "early stopping":

Protocol P_3 (for undecided process i):

if $K_i \exists 0$ **then** $\mathsf{decide_i}(0)$

elseif `sender_set_repeats` for i **then** $\mathsf{decide_i}(1)$

Optimal: All decisions by time $f + 1 \leq t + 1$

for $f = \#$ actual failures

On Being Better

P_3 strictly improves on P_2, which strictly improves on P_1;

Can every consensus protocol be strictly improved upon?

A Knowledge-based Analysis: Deciding on 0

- By **Validity**, $\exists 0$ is a necessary condition for $\mathrm{decide_i}(0)$.
- By the **K*o*P**, $K_i \exists 0$ is a necessary condition for $\mathrm{decide_i}(0)$.

Both P_2 and P_3 decide on 0 using the rule:

$$\textbf{if} \quad K_i \exists 0 \quad \textbf{then} \quad \mathrm{decide_i}(0)$$

No consensus protocol can decide on 0 any faster than that!

A Knowledge-based Analysis: Deciding on 1

Suppose the rule for deciding 0 is $K_j \exists 0 \Leftrightarrow \mathsf{decide_j}(0)$.

When can $\mathsf{decide_i}(1)$ be performed?

- By **Agreement**, "no currently active process has decided 0" is a necessary condition for $\mathsf{decide_i}(1)$; so
- ψ = "$K_j \exists 0$ holds for no active process" is a necessary condition for $\mathsf{decide_i}(1)$;
- By the **K*o*P**, $K_i \psi$ is a necessary condition for $\mathsf{decide_i}(1)$.

An Unbeatable Protocol [Castanèda, Gonczarowski & M. '14]

Protocol OPT_0 (for undecided process i):

if	$K_i \exists 0$	**then** $\mathrm{decide_i}(0)$
elseif	$K_i\big(\texttt{nobody\_knows(0)}\big)$	**then** $\mathrm{decide_i}(1)$

Theorem (CGM)

- *OPT_0 strictly dominates P_3, in some cases by $O(t)$ rounds;*
- *OPT_0 is the first unbeatable consensus protocol;*
- *OPT_0 can be implemented very efficiently.*

Conclusions

- Knowledge is inherent in distributed and multi-agent protocols
- **K*o*P** relates knowledge and action in a new way
- **K*o*P** applies in all models of distributed and multi-agent systems
- Knowledge-based analysis and **K*o*P** facilitate structured design of efficient protocols
- Diverse applications including VLSI, Biology, real-time coordination and more