Concept of Knowledge in Distributed Systems

Swarup Mohalik
LaBRI, Université de Bordeaux I

Plan

- Motivating examples
- The framework
- Formulation of the examples
- Main result about common knowledge
- Variants of common knowledge
- Other issues

Distributed Systems

N agents (or *processes*).

Communication subsystem (Message passing, shared memory, ..)

Processes may or may not start with some publicly known information.

Muddy Children Problem

n children playing in the mud.
k of them have mud on their forehead.
Of course, nobody can see her (his) own forehead (no mirrors).
Father comes and tells "At least one of you has a muddy forehead".
Then he asks "Can you prove that you have mud on your forehead ?" again and again.

Assuming the children are intelligent and also truthful, what happens?
Can anyone prove that he has mud on his forehead?
How many rounds he takes to do so?

Solution

Claim 1 *For the first $k-1$ times, all the children will say* no*, but after the k'th question, all the k dirty children will simultaneously say* yes*.*

Prove by induction on k. When $k = 1$, the dirty child knows immediately after father's announcement, that he is the one who is dirty. So he answers *yes* to the first question.

$k = 2$: Suppose A and B are the dirty children. Initially, A knows that he may or may not be dirty, but of the others, only B is dirty. A assumes he is not dirty. Then, B should have answered *yes* after the first question. But, B also sees A and hence says *no* to the first question. Hence, A knows that he is also dirty. So at the second question both A and B answer *yes*.

The general case is similar.

Muddy children - continued

When $k \geq 1$, every child knows already the announcement of the Father (Ann : there is at least one with muddy forehead). Hence Ann is redundant.

Muddy children - continued

When $k \geq 1$, every child knows already the announcement of the Father (Ann : there is at least one with muddy forehead). Hence Ann is redundant.

This conclusion is FALSE !!!

Without the announcement, muddy children can never prove that their foreheads are muddy. How does one prove?

Coordinated Attack Problem

Two divisions of an army are camped on two hilltops overlooking a common valley, where there is an enemy army. If both divisions attack the enemy simultaneouly then they will win. If only one division attacks then it will be defeated. The divisions do not initially have any plan to attack. General of one of the division wants to coordinate a simultaneous attack the next day (say at 0800 hrs). Neither general will start the attack unless he is sure that the other one will also start attacking at the identical time.

The generals can only communicate through messengers who take an hour to reach the other hilltop. They may, however, get lost (in the dark, by the enemy etc.) On this night, everything went smoothly (the messenger delivers the messages sent through him every hour).

How long it will take for the generals to coordinate the attack?

Solution

Despite the fact that everything goes smoothly, no agreement can be reached among the generals and hence no general can decide to attack.

Intuition : No finite number of messages suffice to achieve coordination.

Prove by induction on the number of messages that if there is an attack by a general, simultaneity is not guaranteed.

Simultaneity $\equiv$ Common knowledge (to be formalized shortly).

Logic of Knowledge

Propositional modal logic of n-agents.

Modal operators $K_1, K_2, \ldots, K_n$.

$p \mid \neg\phi \mid \phi \vee \psi \mid K_i\phi$

$K_i\phi$ is read *agent i knows ϕ*.

$K_1\phi \wedge K_1K_2\phi \wedge \neg K_2K_1K_2\phi$

Knowledge of group of agents

Everyone in a group G knows p.

(More complicated) Everyone in G knows that everyone in G knows p.

(Common knowledge) In a group G, everyone knows p; everyone knows that everyone knows that p, everyone knows that everyone knows that everyone knows that p. ....

New modalities

- $E_G(p)$ - *everyone's knowledge*.
- $E_G^{k+1}(p)$ is an abbreviation of $E_G E_G^k(p)$
- $C_G(p)$ - *common knowledge*.

Kripke Structures

$M = (S, \pi, \mathcal{K}_1, \ldots, \mathcal{K}_n)$, where

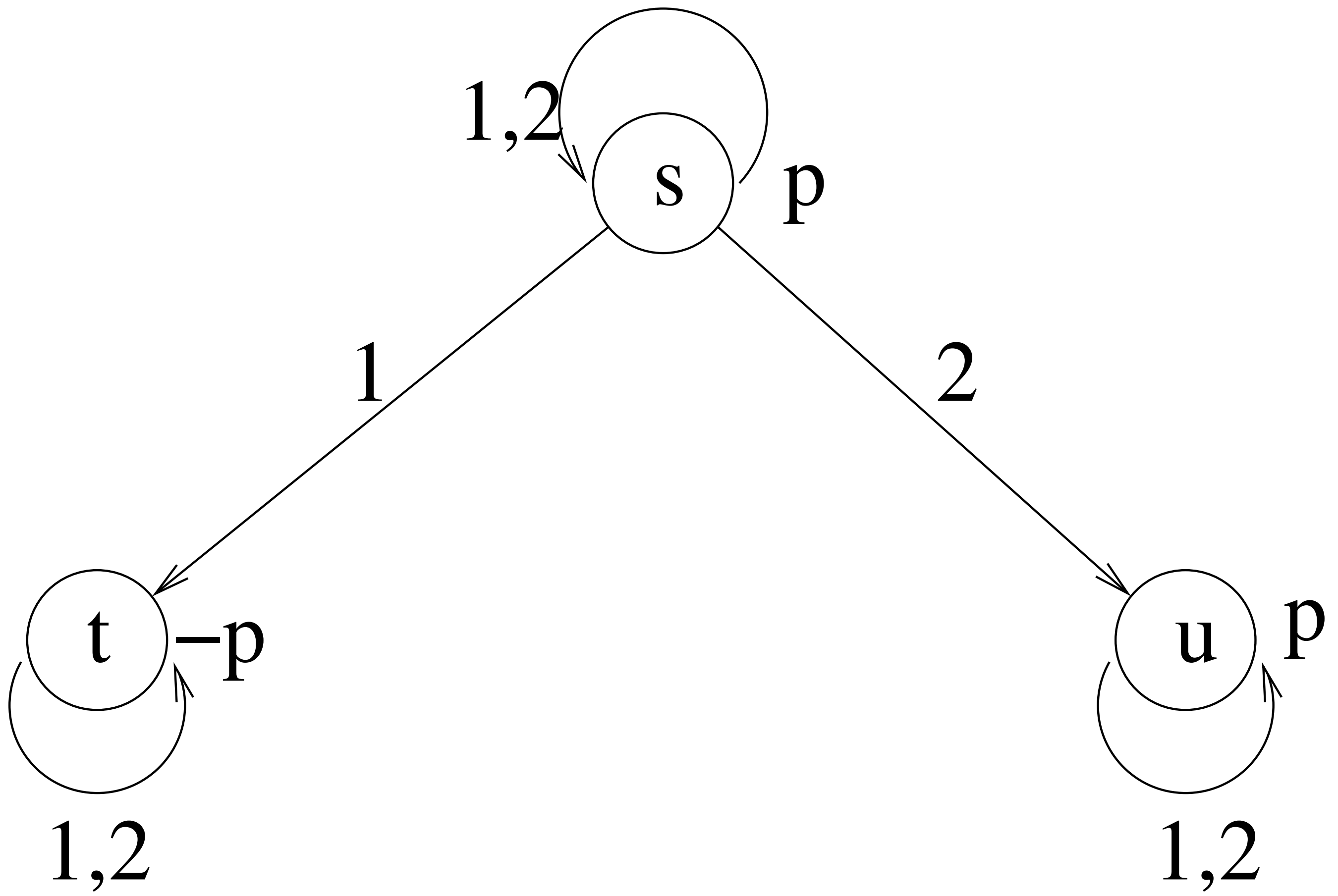
- S is a set of states (or *possible worlds*),
- $\pi : S \to \mathcal{P}(Prop)$,
- $\mathcal{K}_i$ is an equivalence relation on S (the possibility relation of agent i).

Interpretation of Knowledge modalities

- $(M, s) \models K_i\phi$ if $(M, t) \models \phi$ for all t such that $(s, t) \in \mathcal{K}_i$.
- $(M, s) \models E_G\phi$ if $(M, s) \models K_i\phi$ for all $i \in G$.
- $(M, s) \models C_G\phi$ if $(M, s) \models E_G^k\phi$ for all $k = 1, 2, ...$

  A formula is valid if it holds in all models and at all states.

Example



$(M, s) \models p \wedge \neg K_1 p \wedge K_2 p \wedge \neg K_2 \neg K_1 p$

Useful properties

- $E_G\phi \iff \bigwedge_{i\in G} K_i\phi$
- $C_G\phi \iff E_G(\phi \wedge C_G\phi)$
- From $\phi \implies E_G(\phi \wedge \psi)$ infer $\phi \implies C_G\psi$ (Induction rule)

Modeling Distributed Systems

n Agents with local states $S_1, S_2, \ldots, S_n$; One environment with local state S_e.

A global state is a tuple $(s_e, s_1, s_2, \ldots, s_n)$. Let ST be the set of all global states.

A run of the system is a sequence of global states.

A point is a pair (r, m) : a run r and a time m. At a point (r, m), the system is in the global state $r(m)$.

If $r(m) = (s_e, s_1, s_2, \ldots, s_n)$, then $r_i(m) = s_i$ is the local state of agent i at the point (r, m).

The system is defined as a set of runs R.

Modelling - continued

A Knowledge Interpretation is $I = (R, \pi, v)$ where
$\pi : ST \to \mathcal{P}(Prop)$, $Prop$ is the set of primitive propositions about the system under study, $v(p_i) : (r, m) \to \Sigma$ such that it is a function of the local history of p_i.

The points of R are the points of I and $\pi(r, m) = \pi(r(m))$.

The Kripke structure associated with I:
$M_I = (S, \pi, \mathcal{K}_i)$ where

- S is the set of points in I,
- $(r, m)\mathcal{K}_i(r', m')$ if $v(p_i)(r, m) = v(p_i)(r', m')$

Given $I = (R, \pi, v)$, ϕ is *valid* in R if for all points (r, t), $(I, r, t) \models \phi$.

Facts about common knowledge

Fact 2 *Let I be a knowledge interpretation for R and let (r, t) be a point of R. Then the following are equivalent for any nonempty subset G of processors:*

- $(I, r, t) \models C_G\phi$
- $(I, r, t) \models K_i(\phi \wedge C_G\phi)$ for all processors $p_i \in G$
- $(I, r, t) \models K_i(\phi \wedge C_G\phi)$ for some processor $p_i \in G$.

Coordinated attack - formulation

Let the generals A and B follow protocols P_A and P_B respectively. Then the distributed system is modelled as the set of runs consistent with the joint protocol (P_A, P_B).

At the beginning, there is no plan to attack.

Proposition 3 *Any correct protocol for the coordinated attack problem satisfies the following: whenever the generals attack, it is common knowledge that they are attacking.*

Common Knowledge can not be Attained

Consider a system where communication is not guaranteed.

Theorem 4 *Let R be a system in which communication is not guaranteed. Let I be a knowledge interpretation for R and let $|G| \geq 2$. Let r be a run of R and let r^- be another run with the same initial configuration and the same clock readings as r such that no messages are received in r^- up to time t. Then, for all formulas ϕ, $(I, r, t) \models C_G\phi$ iff $(I, r, t) \models C_G\phi$.*

In a system where communication is not guaranteed, no fact can become common knowledge through the exchange of messages.

Corollary 5 Any correct protocol for the coordinated attack problem guarantees that neither general ever attacks.

Common knowledge and communication

Common knowledge is not attainable in systems where

- communication is not guaranteed,
- communication is guaranteed but there is no bound on the message delivery times,
- communication is guaranteed, messages are guaranteed to be delivered *within* a fixed bound.

In systems with temporal imprecision, common knowledge can not be attained.

Paradox of Common Knowledge

Common knowledge can not be attained in practical distributed systems and hence agreements can not be achieved, actions can not be coordinated.

On the other hand, practical distributed systems do reach agreement and perform coordinated actions regularly.

Where is the contradiction?

Common knowledge corresponds to absolutely simultaneous coordination. For many applications, weaker forms of coordination and hence weaker variants of common knowledge often suffice. These variants are, of course, attainable.

In many other situations, the model may be abstracted safely so that in the abstracted model common knowledge can be attained.

Approximations of Common Knowledge-I

ϵ-common knowledge for synchronous broadcast channels.

$(I, r, t) \models E_G^\epsilon \phi$ if there is an interval $INT = [t', t' + 1]$ such that $t \in INT$ and for all p_i there is a $t_i \in INT$ such that $(I, r, t_i) \models K_i \phi$.

Processes come to know about ϕ not simultaneously but within an ϵ interval.

Consider 2 processes with fixed bound of time for messages. Common knowledge is not attainable for this case whereas ϵ-common knowledge is.

Approximations of Common Knowledge-II

Eventual common knowledge for asynchronous channels with no bound on message delivery time.

$(I, r, t) \models E^{\diamond}_{G}\phi$ if for all p_i there is a $t_i \geq t$ such that $(I, r, t_i) \models K_i\phi$.

Byzantine agreement - whenever a correct processor decides on a given value, every other correct processor is guaranteed to *eventually* decide on the same value.

In error-free asynchronous broadcast channels, a processor knows that *sent(m)* is eventual common knowledge as soon as it receives a message m.

Because of asynchrony, ϵ-common knowledge of *sent(m)* is never attained.

Knowledge-based Programs

Transition systems with edges labeled by *knowledge tests*.

```
Sender::
i = 0; read y;
while true {
   if ¬K_S K_R(x@i)
      send < i, y >; receive z;
    else
      i = i++; read y; receive z;
}
```

```
Receiver::
j = 0;
while true {
    if ¬K_R(x_j)
      send [¬K_R(x_j)]; receive z';
    else if K_R(x_j = 0)
            write 0; j = j++; receive z';
         else
            write 1; j = j++; receive z';
}
```

Many protocols for the sequence transmission problem are implementations of the knowledge-based protocol above.

Conclusion

Variants of common knowledge for different applications

Verification of KBPrograms.

Synthesis for knowledge formulas.

Synthesis of knowledge-based programs.

Refinement of knowledge-based programs to produce implementable protocols.