Logics for Multi-agent Systems

Sophie Pinchinat
http://people.irisa.fr/Sophie.Pinchinat/

7-11 december 2015

Motivations

Multi-agent systems are everywhere

- Agent-based computing addresses the challenges in managing distributed computing systems and networks through monitoring, communication, consensus-based decision-making and coordinated actuation.
- Multi-agent systems have demonstrated the capability to use intelligence, knowledge representation and reasoning, and other social metaphors like 'trust', 'game' and 'institution'.
- Transformative impact in many application domains, planning, logistics, manufacturing, e-commerce, robotics, decision support, transportation, entertainment, emergency relief & disaster management, and data mining & analytics.

As one of the largest and still growing research fields of Computer Science, agent-based computing today remains a unique enabler of inter-, multi- and trans-disciplinary research.

Developing logics for MAS

A challenging field where one should design

Syntax and Semantics
+
Model-checking and Models Synthesis
(effective methods)

In classical books on logic

Syntax then Semantics

In formal methods, it is rather

Semantics then Syntax

Part I: Models for Time, Knowledge, and both

Content

Content

What do we mean by "time"?

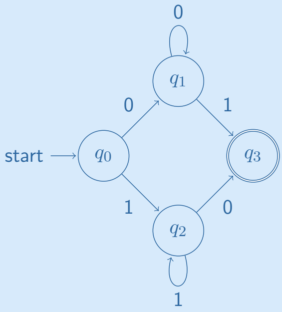
- We talk here about logical time, instants
- Nothing quantified like 3 seconds
- The state-transition models are very well adapted

State-transition models

- States = configurations
- Transitions from states to states, triggered by an event/action.

You know many of those.

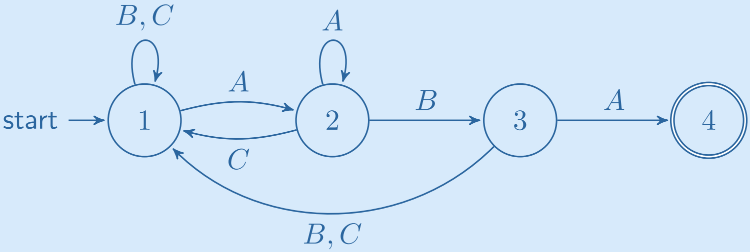
Example (Finite automata, pushdown automata, Turing machines)



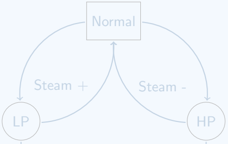
Physical systems

Example (A digicode whose code is ABA)

[BBF$^+$99, Sec. 1.1]

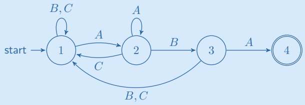


Example (Nuclear plant + its environment)

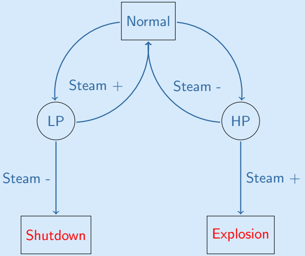


Physical systems

Example (A digicode whose code is ABA)



Example (Nuclear plant + its environment)



Distributed systems/Multi-player games

Example (Rock/Paper/Scissors)

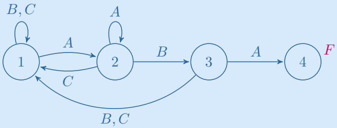
Exercise

How would you model Rock/Paper/Scissors?

Read Section 2 of [AHK02], very easy to access on-line.

Transition systems

Example



$Prop = \{F\}$ and $Ev = \{A, B, C\}$
$S = \{1, 2, 3, 4\}$
$\ell(s) = \emptyset$ for $s \in \{1, 2, 3\}$
$\ell(4) = \{F\}$

Fix a set $Prop$ of *atomic propositions* and a set Ev of *events*.

Definition

A *transition system* over $Prop$ and Ev is a structure $\mathcal{S} = \langle S, \delta, \ell \rangle$ where

- S is a set of state, and sometimes $S_0 \subseteq S$ a set if initial states
- $\delta \subseteq S \times Ev \times S$ is the transition relation
- $\ell : S \to 2^{Prop}$

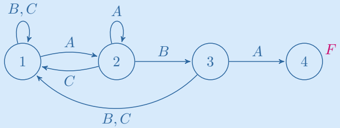
Write $s \xrightarrow{e} s'$ for $(s, e, s') \in \delta$, as in pictures.

Paths, traces of transition systems

Definition (Paths, Executions, Traces)

A finite *path* is a finite sequence $\rho = s_1 \xrightarrow{e_2} s_2 \ldots \xrightarrow{e_n} s_n$, and a *execution* is as a path where only states are recorded $s_1 s_2 s_3 \ldots$
We can also define *infinite* paths, with typical element π, and *initial* (finite and infinite) paths that start from a some of the distinguished states
The *trace* of a path $\pi = s_1 \xrightarrow{e_2} s_2 \ldots$ is the sequence $\ell(s_1) e_2 \ell(s_2) \ldots$

Example



$1A2A2B3A4$ is an initial path
$1.2.2.3.4$ is an execution
$\emptyset A \emptyset A \emptyset B \emptyset A \{F\}$ is a trace.

Combining state-transition models

- Product of finite automata for language intersection
- Parallel composition for distributed systems

Definition

Given two transition systems $\mathcal{S}_1 = \langle S_1, \delta_1, \ell_1 \rangle$ and $\mathcal{S}_2 = \langle S_2, \delta_2, \ell_2 \rangle$, over $Prop_1$ and Ev_1, and $Prop_2$ and Ev_2, respectively.
Assume $Prop_1 \cap Prop_2 = \emptyset$.
Let a partial function $f : Ev_1 \times Ev_2 \rightarrow Ev$ be called a *synchronization table* [Arn92]. We let $\mathcal{S}_1 \mid_f \mathcal{S}_2 = \langle S_1 \times S_2, \delta, \ell \rangle$ be the transition system over $Prop_1 \cup Prop_2$ and Ev such that:

- $(s_1, s_2) \stackrel{f(e_1,e_2)}{\rightarrow} (s_1', s_2')$ whenever $f(e_1, e_2)$ is defined, and $s_1 \stackrel{e_1}{\rightarrow} s_1'$ and $s_2 \stackrel{e_1}{\rightarrow} s_2'$.
- $\ell((s_1, s_2)) = \ell_1(s_1) \cup \ell_2(s_2)$.

Classic parallel compositions

- Truly parallel composition

$$\begin{array}{ccc} s_1 & s_2 & s_3 \\ \downarrow e_1 & \downarrow e_2 & \downarrow e_3 \\ s_1' & s_2' & s_3' \end{array}$$

Ev_1	Ev_2	Ev_3	$Ev_1 \times Ev_2 \times Ev_3$
e_1	e_2	e_3	(e_1, e_2, e_3)

Exercise

Do you think Rock/Paper/Scissors is of this kind?

- Interleaving parallel composition

$$\begin{array}{ccc} s_1 & s_2 & s_3 \\ \downarrow e_1 & \circlearrowright & \circlearrowright \\ s_1' & - & - \end{array}$$

Ev_1	Ev_2	Ev_3	$Ev_1 \cup Ev_2 \cup Ev_3$
e_1	$-$	$-$	e_1
$-$	e_2	$-$	e_2
$-$	$-$	e_3	e_3

Example of parallel composition

- Regarding truly and interleaving parallel composition of previous slide

Exercise

Define them formally.

- Synchronization over shared actions, of the kind:

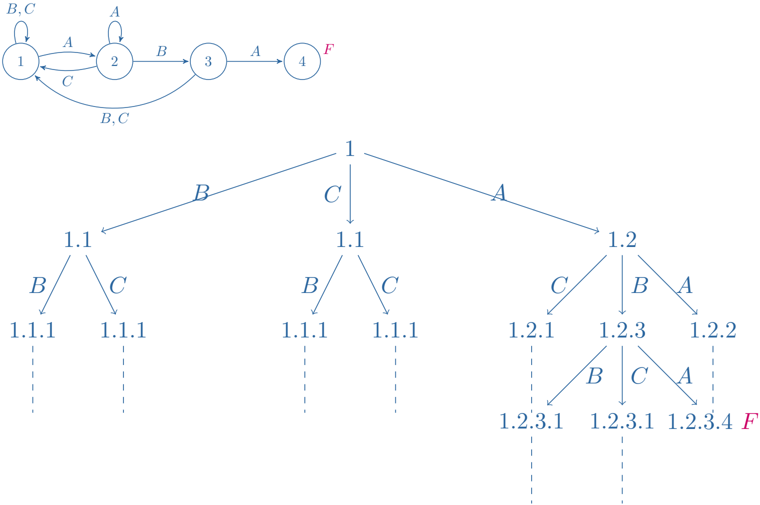
$$\begin{array}{ccc} s_1 & s_2 & s_3 \\ \downarrow e & \downarrow e & \downarrow e \\ s_1' & s_2' & s_3' \end{array} \qquad \text{but also} \qquad \begin{array}{ccc} s_1 & s_2 & s_3 \overset{e}{\not\rightarrow} \\ \downarrow e & \downarrow e & \circlearrowleft \\ s_1' & s_2' & - \end{array} \qquad \ldots$$

Exercise

Define it formally.

- Take a look at [BBF+99, Part I, Sec. 1] for examples of use of parallel composition.

Unfolding transition systems



Computation trees (1/2)

for the "branching-time" behavior of transition systems

- They are infinite objects.
- They serve as models for temporal logics, as well as an inputs to infinite-tree automata [Tho90].

Definition (Trees)

Given (finite) subset $D \subseteq \mathbb{N}$, a *(D)-tree* τ is a set of words $\tau \subseteq D^+$ such that:

1. the root is ϵ
2. if $x \cdot d \in \tau$, then $x \in \tau$, and
3. if $x \in \tau$ then there exists $d \in D$ such that $x \cdot d \in \tau$.

The nodes can be labelled over some set Σ according to $\ell : \tau \to \Sigma$, and we denote by t a labelled tree, that is some pair (τ, ℓ).

Computation Trees (2/2)

Definition (Computation tree over $Prop$ and Ev)

They basically are trees whose nodes are labelled over 2^{Prop} and whose edges are labelled over Ev.

Definition (Computation tree of a transition system)

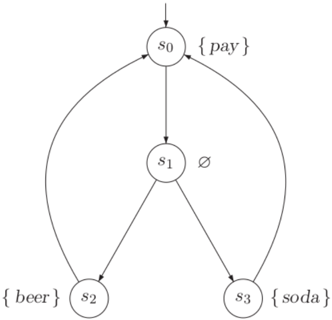
Let $\mathcal{S} = \langle S, \delta, \ell \rangle$ be a transition system over $Prop$ and Ev, and let $s_0 \in S$ be a distinguished state.
The *computation tree of* $\mathcal{S}$ from state s_0 is the infinite labelled tree $t_{\mathcal{S},s_0}$ whose nodes are the initial executions of $\mathcal{S}$ etc. and whose labelling is $\ell(s_0 s_1 \dots s_n) = \ell(s_n)$ and $s_0 s_1 \dots s_n \xrightarrow{e} s_0 s_1 \dots s_n s_{n+1}$ whenever $s_n \xrightarrow{e} s_{n+1}$ in $\mathcal{S}$.

Remark

- The computation tree $t_{\mathcal{S},s_0}$ itself is a transition system
- The branches of $t_{\mathcal{S},s_0}$ are the paths of $\mathcal{S}$

Behaviors: Two Beverage Vending Machines

[BK08]



Exercise

- Draw the computation trees of each machine, and check they are intrinsically different.
- Check that they share the same set of traces

An abstraction of behaviors: Bisimulation

Introduced by [Par81, Mil83]

Definition (Bisimulation)

Let $\mathcal{S}_1 = \langle S_1, \delta_1, \ell_1 \rangle$ and $\mathcal{S}_2 = \langle S_2, \delta_2, \ell_2 \rangle$ and Ev.
Distinguish two state $s_1^0 \in S_1$ and $s_2^0 \in S_2$.
We say that $\langle \mathcal{S}_1, s_1^0 \rangle$ and $\langle \mathcal{S}_2, s_2^0 \rangle$ are *bisimilar* if there exists binary relation $\mathcal{B} \subseteq S_1 \times S_2$ s.t.

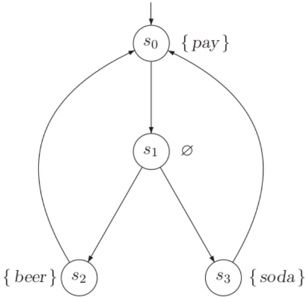
- (initial) $(s_1^0, s_2^0) \in \mathcal{B}$, and
- for every $(s_1, s_2) \in \mathcal{B}$
  - (atoms) $\ell_1(s_1) = \ell_2(s_2)$
  - (forth) for any $s_1 \xrightarrow{e} s_1'$ there exists $s_2 \xrightarrow{e} s_2'$ s.t. $(s_1', s_2') \in \mathcal{B}$
  - (back) for any $s_2 \xrightarrow{e} s_2'$ there exists $s_1 \xrightarrow{e} s_1'$ s.t. $(s_1', s_2') \in \mathcal{B}$

$\mathcal{B}$ is a *bisimulation* between $\langle \mathcal{S}_1, s_1^0 \rangle$ and $\langle \mathcal{S}_2, s_2^0 \rangle$.

Remark

Bisimilarity is an equivalence relation.

Bisimilar transition systems



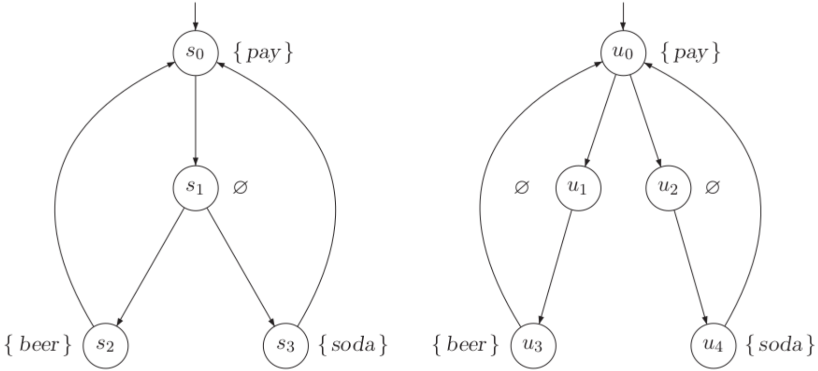
Exercise

Check that the two machines are bisimilar.

Exercise

Prove that whenever two transition systems are bisimilar, they share the same set of traces.

Non-bisimilar transition systems



Exercise

- Argue that the two machines are not bisimilar.
- Check that the machines share the same set of traces.

Transition systems and their computation trees

A transition system and its computation tree are bisimilar:

Theorem

Let $\mathcal{S} = \langle S, \delta, \ell \rangle$ be a transition system and let $s^0 \in S$.
Then $\langle t_{\mathcal{S},s_0}, s^0 \rangle$ and $\langle \mathcal{S}, s^0 \rangle$ are bisimilar.

Proof.

Write $Execs_{\mathcal{S}}$ for the set of executions of $\mathcal{S}$.
Define $\mathcal{B} \subseteq Execs_{\mathcal{S}} \times S$ by: for every execution $s^0 \ldots s$ of $\mathcal{S}$,

$$(s^0 \ldots s, s) \in \mathcal{B}$$

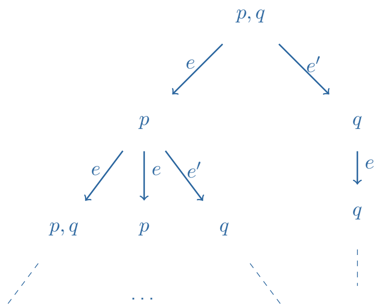
and check that $\mathcal{B}$ is a bisimulation. □

Exercise

Finish the proof.

Computation trees will be our main objects

$Prop = \{p, q \ldots\}$ and $Ev = \{e, e'\}$



Remark

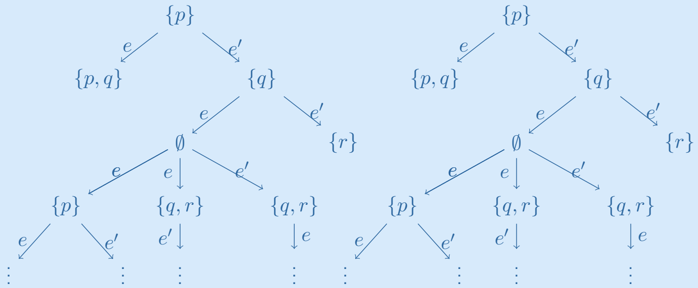
Computation trees are infinite Kripke structures.

(Finite) forests of computation trees

Definition

A *forest* over $Prop$ and Ev is a finite set $\mathcal{F} = \{t^j\}_{j \in J}$ of computation trees over $Prop$ and Ev

Example



Content

What do we mean by "knowledge"?

- [vDvdHK07]

  > *We regard information as something that is relative to a subject who has a certain perspective on the world, called an agent, and the kind of information we have in mind is meaningful as a whole, not just loose bits and pieces.*
  > *This makes us call it knowledge (and, to a lesser extent, belief). This conception of information is due to the fields known as epistemic and doxastic logic.*

- The clearest source is the book of Hintikka (12 January 1929 – 12 August 2015) *Knowledge and Belief: An Introduction to the Logic of the Two Notions* [Hin62].
- A reference is the book of Fagin, Halpern, Moses, and Vardi *Reasoning About Knowledge* [FMHV03].

Representing knowledge

- Knowledge of agents is modelled by using Kripke models, called *epistemic models* in this context.
- Two notions are of main importance: that of *state* and that of *indistinguishability*.
- We will denote by Ag a finite set of agents, with typical elements $a, b, \ldots$

An introductory example: the GLO-scenario [vDvdHK07, p. 16]

An agent, say b, lives in Groningen and builds a theory about the weather conditions in both Groningen and Liverpool. In Groningen (resp. Liverpool) it is either sunny, denoted by the atom g (resp. atom l), or not $\neg g$ (resp. atom $\neg l$).

A priori, there are 4 possible situations.

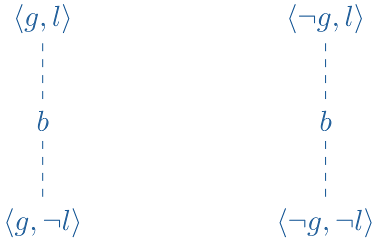
$\langle g, l \rangle$ $\langle \neg g, l \rangle$

$\langle g, \neg l \rangle$ $\langle \neg g, \neg l \rangle$

An introductory example: the GLO-scenario [vDvdHK07, p. 16]

An agent, say b, lives in Groningen and builds a theory about the weather conditions in both Groningen and Liverpool. In Groningen (resp. Liverpool) it is either sunny, denoted by the atom g (resp. atom l), or not $\neg g$ (resp. atom $\neg l$).

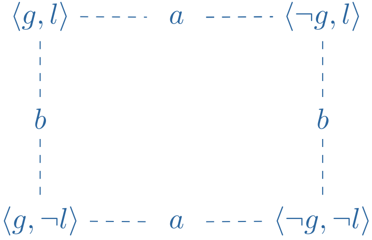
Since agent b cannot observe property l, it cannot e.g. distinguish between $\langle g, l \rangle$ and $\langle g, \neg l \rangle$



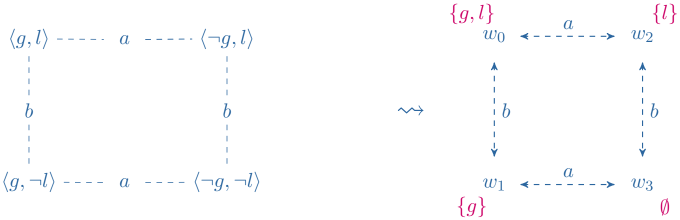
An introductory example: the GLO-scenario [vDvdHK07, p. 16]

An agent, say b, lives in Groningen and builds a theory about the weather conditions in both Groningen and Liverpool. In Groningen (resp. Liverpool) it is either sunny, denoted by the atom g (resp. atom l), or not $\neg g$ (resp. atom $\neg l$).

A second agent, say a, is situated in Liverpool and knows about the weather there.



Epistemic models

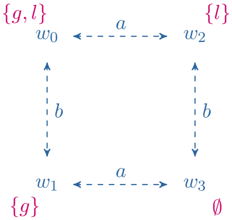


Definition (Epistemic model)

An *epistemic model* over atomic propositions $Prop$ and agent set Ag is a structure $\mathcal{M} = \langle W, \mathcal{R}, \ell\rangle$ where

- W is a set of *possible words*
- $\mathcal{R} = \{\mathcal{R}_a\}_{a \in Ag}$ where $\mathcal{R}_a \subseteq W \times W$
- $\ell : W \rightarrow 2^{Prop}$ is a labelling

Epistemic models



Definition (Epistemic model)

An *epistemic model* over atomic propositions $Prop$ and agent set Ag is a structure $\mathcal{M} = \langle W, \mathcal{R}, \ell \rangle$ where

- W is a set of *possible words*
- $\mathcal{R} = \{\mathcal{R}_a\}_{a \in Ag}$ where $\mathcal{R}_a \subseteq W \times W$
- $\ell : W \to 2^{Prop}$ is a labelling

Epistemic models

Definition (Epistemic model)

An *epistemic model* over atomic propositions $Prop$ and agent set Ag is a structure $\mathcal{M} = \langle W, \mathcal{R}, \ell \rangle$ where

- W is a set of *possible words*
- $\mathcal{R} = \{\mathcal{R}_a\}_{a \in Ag}$ where $\mathcal{R}_a \subseteq W \times W$
- $\ell : W \to 2^{Prop}$ is a labelling

Remark

As in the GLO-scenario, relations $\mathcal{R}_a$ are very often equivalence relations, but still we keep it general (and it is so for the "more plausible" relations in *doxastic logic*).

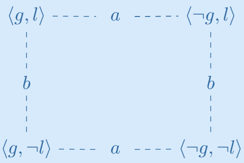
We write $w \overset{a}{\hookrightarrow} w'$ whenever $(w, w') \in \mathcal{R}_a$.

Where is knowledge?

- Indistinguishability gives rise to a natural notion of knowledge: in an actual world w, agent a knows a fact/property (here just think of Boolean combinations of atomic propositions) if in any possible world w' such that $w \overset{a}{\leadsto} w'$, the fact holds.
- This will be the semantics of the *knowledge modality* in Epistemic Logic **K**.

Knowledge for the GLO-scenario

Example



In world w_0:

- Agent b knows that g but does not know that l.
- Agent b neither knows that l, nor that $\neg l$.
- Agent b considers both l and $\neg l$ possible.
- Agent b knows that agent a does not know that g.
- Agent a knows that agent b knows that agent a does not know that g.

More on Knowledge

- *Common knowledge* arises from the indistinguishability relation induced by

  the transitive closure of $\bigcup_{a \in Ag} \overset{a}{\curvearrowright}$

  $$w \rightarrowtail_{CK} w' \text{ whenever } w = w_0 \overset{a_1}{\curvearrowright} w_1 \overset{a_2}{\curvearrowright} \ldots \overset{a_k}{\curvearrowright} w_k = w'$$

  A fact is common knowledge if agent a knows that agent b knows that agent a knows that ... at any depth.

- *Distributed knowledge* arises from the indistinguishability relation

  $$\bigcap_{a \in Ag} \overset{a}{\curvearrowright}$$

- The models have some defects: agents are *omniscient* since they then know all validities.

We point to [FMHV03] for full details.

Bisimulation over epistemic models

Exercise (Bisimulation between epistemic models)

Write down a clean definition.

The Muddy Children Puzzle

A group of children has been playing outside and are called back into the house by their father. The children gather round him.
As one may imagine, some of them have become dirty from the play and in particular: they may have mud on their forehead.
Children can only see whether other children are muddy, and not if there is any mud on their own forehead. All this is commonly known, and the children are, obviously, perfect logicians.
Father now says: "At least one of you has mud on his or her forehead." And then: "Will those who know whether they are muddy step forward."
If nobody steps forward, father keeps repeating the request.

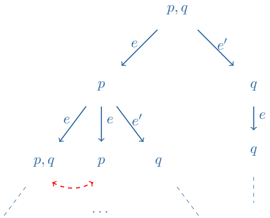
What happens?

Content

Enriching Computation Trees to attain Knowledge

- Computation trees give us evolution of a system over time.
- A node denotes a particular *history*: the finite branch leading to that node.
- Agents with their own perception abilities might consider the actual history indistinguishable from some others. Histories play the role of possible worlds, yielding an infinite epistemic model.

Computation tree can be enriched with "transverse" indistinguishability relations between nodes.



How do we get these transverse relations?

Two main approaches

- From time to knowledge: Interpreted Systems [HF89, FMHV03]
  - They are natural models for distributed systems (each sub-system is an agent)
  - Global states are vectors of the local states of each sub-system
  - Indistinguishably results from the local view of each sub-system (it is a *grounded semantics*).
- From knowledge to time: Dynamic Epistemic Logic models [vDvdHK07]
  - Indistinguishably/knowledge is inherent to the initial epistemic configuration
  - Indistinguishably/knowledge evolves according to the occurrences of legitimate epistemic events/actions.

Both approaches meet when considering abstract models:

the *Epistemic Temporal Logic frames*.

Interpreted systems over $Prop$ and Ev [HF89]

Definition

- $Ag = \{1, \ldots, n\}$ (n components of a distributed system)
- Each agent i is associated a transition system $\mathcal{S}_i = \langle S_i, \delta_i, \ell_i \rangle$ over $Prop$ and Ev: local states of agent i range over S_i.
- Global states are those of $\mathcal{S}_1 \mid \mathcal{S}_2 \mid \ldots \mid \mathcal{S}_n$ (for some notion of parallel composition $\mid$), that is of the form $\langle s_1, s_2, \ldots, s_n \rangle \in S_1 \times S_2 \times \ldots \times S_n$.
- For each agent $i \in Ag$ we let $\stackrel{i}{\hookrightarrow}$ be: $\langle s_1, s_2, \ldots, s_n \rangle \stackrel{i}{\hookrightarrow} \langle s'_1, s'_2, \ldots, s'_n \rangle$ iff $s_i = s'_i$

Extending relations $\stackrel{i}{\hookrightarrow}$ onto histories, depends on agents' abilities:

1. how they perceive time evolution, and
2. what they remember from the past.

A zoology of knowledge semantics [HV86]

- Interpreted systems are essentially identified with their computation trees.
- Classic semantics for knowledge are classified according:
  - synchronous vs. asynchronous (time evolution perception)
  - imperfect recall such as memoryless, bounded memory, etc., vs. perfect recall (memory capabilities)
- Epistemic Temporal Logic (ETL), where one can mix temporal and knowledge modalities, is then studied (axiomatization, computational perspective, etc.)

We defer the definition of the different knowledge semantics to the more abstract setting of ETL frames.

Epistemic Temporal Logic frames

- They abstract from the internal structure of global states hence from the means to obtain the global transitions.
- They have binary relations $\overset{a}{\hookrightarrow}$ between nodes, for each $a \in Ag$.
- They serve as models for Epistemic Temporal Logics.

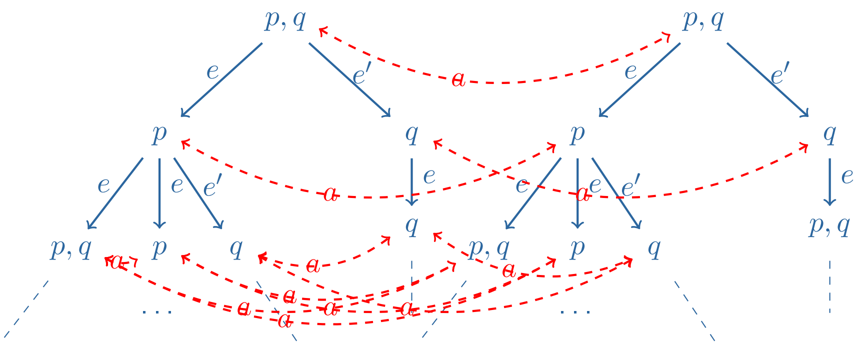
Definition

An *ETL frame over propositions* $Prop$, *events* Ev *and (finite) set of agents* Ag is a structure $\langle \mathcal{F}, \{\overset{a}{\hookrightarrow}\}_{a\in Ag}\rangle$, where $\mathcal{F}$ is a forest over $Prop$ and Ev, and $\overset{a}{\hookrightarrow}$ are binary relations between nodes of the forest $\mathcal{F}$.

Definition

Given an ETL frame $\langle \mathcal{F}, \{\overset{a}{\hookrightarrow}\}_{a\in Ag}\rangle$ and a node $x \in \mathcal{F}$, we write $t(x)$ for the unique computation tree of $\mathcal{F}$ node x belongs to.

ETL frames (are forests)



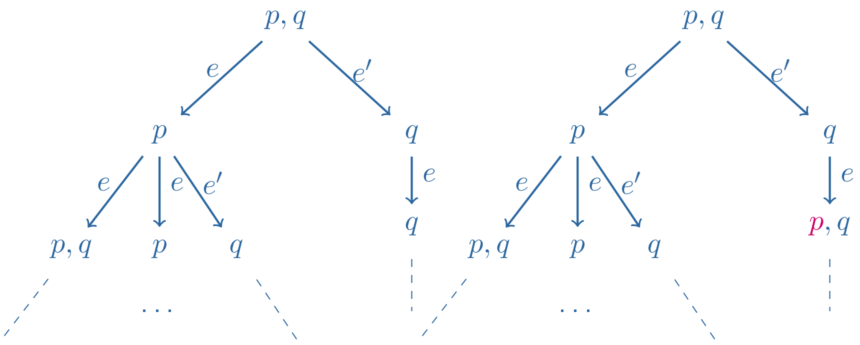
a sees p, e and e', hears the ticks and has perfect recall
(arrows are missing)

Various Knowledge semantics

Definition

- Synchronicity: $x \leadsto x'$ implies $|x| = |x'|$
  where $|x|$ is the height of node x in his tree
  (we count from 0 from the root).

- Perfect recall/not forgetting: $x \not\leadsto x'$ implies $y \not\leadsto y'$
  for each child y (resp. y') of x (resp. x')



Exercise

Draw the tranverse relations if agent a is memoryless.

Exercise

Think of other cases.

PDEL models

PDEL means *Propositional Dynamic Epitemic Logic*

- We start with epistemic models $\mathcal{M} = \langle W, \mathcal{R}, \ell \rangle$.
- We consider *event models* $\mathcal{E}$ that are epistemic strutures where elements of E are concrete events with *preconditions*,and epistemic relations express what agents do not distinguish.

Definition (non ontic version)

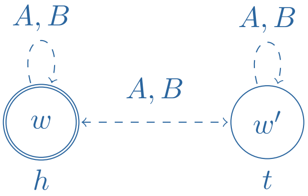
An *event model* is a tuple $\mathcal{E} = (\mathsf{E}, \{\mathsf{R}_a\}_{a \in Ag}, \mathsf{pre})$ where:

- E is a non-empty finite set of possible *events*,
- $\mathsf{R}_a \subseteq \mathsf{E} \times \mathsf{E}$ is an *accessibility relation* on E for agent a,
- $\mathsf{pre} : \mathsf{E} \to 2^{Prop}$ is a *precondition function* and

- When $\mathcal{E}$ occurs, $\mathcal{M} \otimes \mathcal{E}$ is the new epistemic situation.

An example

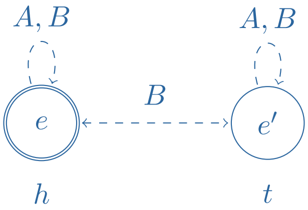
An initial epistemic situation $\mathcal{M}$



Alice and Bob: where Alice places a coin in a cup, shakes the cup and puts it upside down on a table. Assume that Alice and Bob are interested in knowing whether the upside of the coin is heads or tails. In the initial situation we described, neither Alice nor Bob knows it. Atomic propositions h and t mean respectively heads and tails. The doubly circled world, w, is the actual world, the coin is actually on heads.

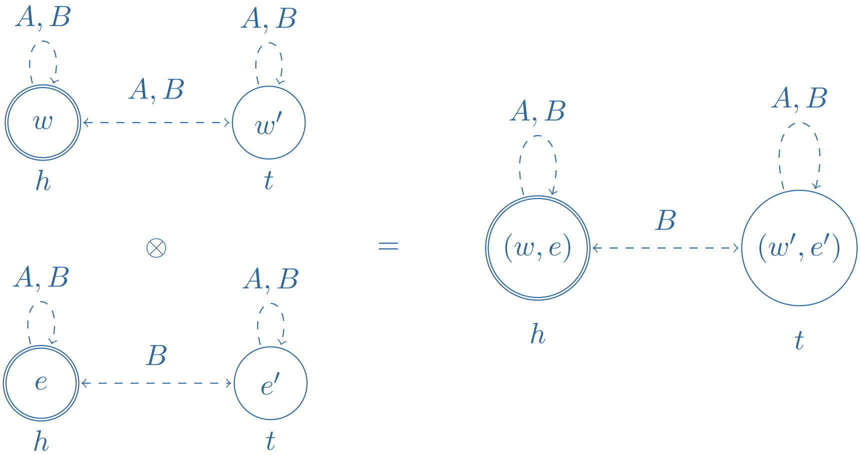
An event model $\mathcal{E}$

Imagine that Alice looks under the cup. Bob sees her doing so, but does not manage to see if it is heads or tails.



Event e is Alice seeing heads, so pre$(e) = h$, and e' is Alice seeing tails, so pre$(e') = t$.
Alice knows what she observes, so that she distinguishes between the events, while Bob does whether e or e' occurs.

Updating $\mathcal{M} \otimes \mathcal{E}$



Update product

Definition

Given an epistemic model $\mathcal{M} = \langle W, \mathcal{R}, \ell \rangle$ and an event model $\mathcal{E} = (\mathsf{E}, \{\mathsf{R}_a\}_{a \in Ag}, \mathsf{pre})$, the *update product* of $\mathcal{M}$ and $\mathcal{E}$ is the epistemic model $\mathcal{M} \otimes \mathcal{E} = (W^{\otimes}, \mathcal{R}^{\otimes}, \ell^{\otimes})$ where:

$$W^{\otimes} = \{(w, e) \in W \times \mathsf{E} \mid \mathcal{M}, w \models \mathsf{pre}(e)\},$$
$$\mathcal{R}^{a\otimes}(w, e) = \{(w', e') \in W^{\otimes} \mid w' \in \mathcal{R}^a(w) \text{ and } e' \in \mathsf{R}_i(e)\},$$
$$\ell^{\otimes}((w, e)) = \ell((w, e))$$

Remark

We will simply write we instead of (w, e)

Where is time?

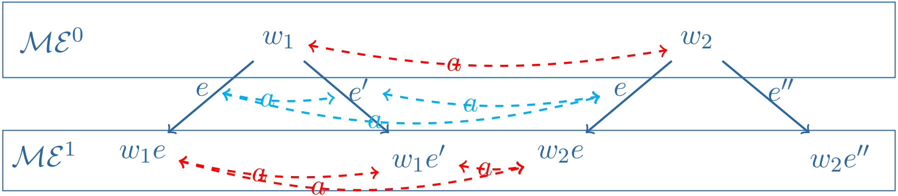
- [vBL04] show how an iterative application of event models to an initial epistemic model yields a structure that can be seen as an ETL frame, and conversely how ETL frames that verify certain properties can be seen as “DEL-generated”.
- Also [AMP14] reconsider this setting and apply the correspondance to show the decidability of a general epistemic protocol problem.

PDEL-generated models

For an epistemic model $\mathcal{M} = (W, \{\mathcal{R}^a\}_{a\in Ag}, \ell)$ and an event model $\mathcal{E} = (\mathsf{E}, \{\mathsf{R}_a\}_{a\in Ag}, \mathsf{pre})$.

Definition

We define the family of epistemic models $\{\mathcal{ME}^n\}_{n\geq 0}$ by letting $\mathcal{ME}^0 = \mathcal{M}$ and $\mathcal{ME}^{n+1} = \mathcal{ME}^n \otimes \mathcal{E}$.



PDEL-generated models

Let $\mathcal{M}$ be an epitemic model and $\mathcal{E}$ be an event model.
For each n, let $\mathcal{M}\mathcal{E}^n = (W_n, \{\mathcal{R}^a{}_n\}_{a \in Ag}, \ell_n)$.

Definition (very tedious ... so not that rigorous here)

The *PDEL-generated model* from $\mathcal{M}$ and $\mathcal{E}$ is the ETL frame $\mathcal{M}\mathcal{E}^* = \langle \{t^w\}_{w \in W}, \{\overset{a}{\hookrightarrow}\}_{a \in Ag}, \ell \rangle$, where:

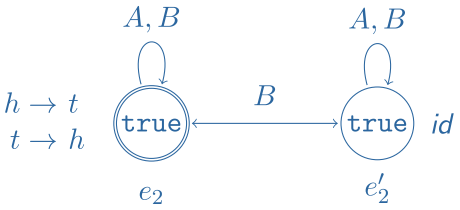
- $\bigcup_{w \in W} t^w = \bigcup_{n \geq 0} W_n$,
- Nodes of $\mathcal{M}\mathcal{E}^*$ are of the form $x = we_1 \dots e_n$
- $x \overset{a}{\hookrightarrow} x'$ if there is some n such that $x, x' \in \mathcal{M}_n$ and $x \mathcal{R}^a{}_n x'$

Theorem

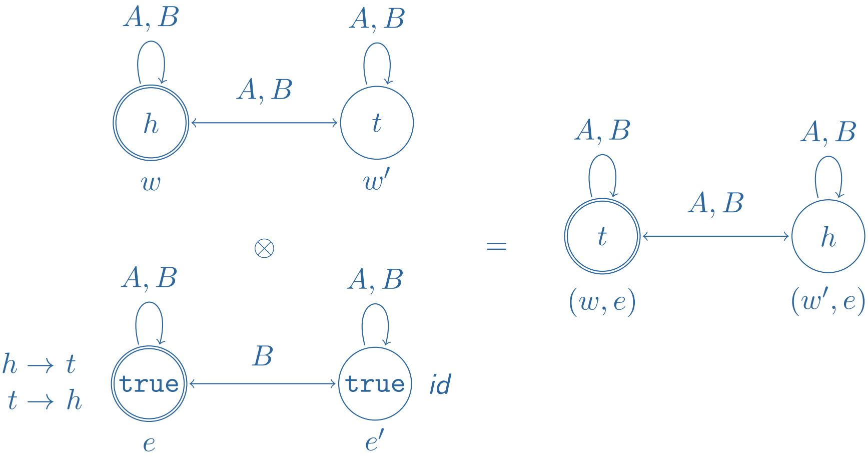
PDEL-generated models are ETL frames where the transverse relations are synchronous perfect recall.

Propositional DEL with ontic events

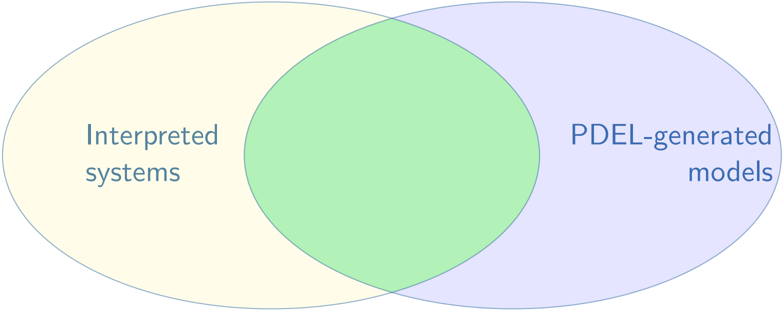
Now Alice can flip the coin without watching it, and Bob sees her manipulating the cup but does not know whether she flipped the coin of not.
We add postconditions to events like $h \to t$.



Applying the ontic event



Time + Knowledge = Knowledge + Time?



Part II: Logics for Time, Knowledge, and both

Content

Modal logic

- Kripke structures are the models the logic talks about
- Formulas are extension of classical propositional logic with new operator $\Box$ (*necessity*) and its dual $\Diamond$ (*possibly*)

Definition (Modal logic)

$\mathcal{M}, w \models \Box\varphi$	iff	$\mathcal{M}, w' \models \varphi$	for all w' **neighbor of** w
$\mathcal{M}, w \models \Diamond\varphi$	iff	$\mathcal{M}, w' \models \varphi$	for some w' **neighbor of** w'

The actual accessibility relation **neighbor of** can capture various dimensions of the reality, and therefore gives rise to different kinds of modal logics: knowledge (epistemic logic), beliefs (doxastic logic), obligations (deontic logic), actions(dynamic logic), time (temporal logic), etc. [BdRV01]

In particular, various aspects of agents (and agent systems) can be naturally captured within this generic framework.

Multi-modal logic

- Kripke structures are the models the logic talks about
- Formulas are extension of classical propositional logic with new operator $\Box_i$ (*necessity*) and its dual $\Diamond_i$ (*possibly*)

Definition (Multi-modal logic)

$\mathcal{M}, w \models \Box_i\varphi$ iff $\mathcal{M}, w' \models \varphi$ for all w' i-**neighbor of** w
$\mathcal{M}, w \models \Diamond_i\varphi$ iff $\mathcal{M}, w' \models \varphi$ for some w' i-**neighbor of** w'

The actual accessibility relation **neighbor of** can capture various dimensions of the reality, and therefore gives rise to different kinds of modal logics: knowledge (epistemic logic), beliefs (doxastic logic), obligations (deontic logic), actions(dynamic logic), time (temporal logic), etc. [BdRV01]

In particular, various aspects of agents (and agent systems) can be naturally captured within this generic framework.

Modal logic

- Kripke structures are the models the logic talks about
- Formulas are extension of classical propositional logic with new operator $\Box$ (*necessity*) and its dual $\Diamond$ (*possibly*)

Definition (logic)

$\mathcal{M}, w \models \Box\varphi$	iff	$\mathcal{M}, w' \models \varphi$	for all w' **neighbor of** w
$\mathcal{M}, w \models \Diamond\varphi$	iff	$\mathcal{M}, w' \models \varphi$	for some w' **neighbor of** w'

The actual accessibility relation **neighbor of** can capture various dimensions of the reality, and therefore gives rise to different kinds of modal logics: knowledge (epistemic logic), beliefs (doxastic logic), obligations (deontic logic), actions(dynamic logic), time (temporal logic), etc. [BdRV01]

In particular, various aspects of agents (and agent systems) can be naturally captured within this generic framework.

Classic decision problems in logic

Definition

Given a logic $\mathcal{L}$, the *model-checking problem for $\mathcal{L}$* is, given an input model $\mathcal{N}$ of $\mathcal{L}$ and an input formula $\varphi \in \mathcal{L}$, to answer the question

$$\text{“}\mathcal{N} \models \varphi\text{?”}$$

Definition

Given a logic $\mathcal{L}$, the *satisfiability problem for $\mathcal{L}$* is, given an input formula $\varphi \in \mathcal{L}$, to answer the question

“Does there exist a model $\mathcal{N}$ such that $\mathcal{N} \models \varphi$?”

Remark

The validity problem can be rephrased as a satifiability problem for the negated formula.

Content

Temporal logics

- Classic temporal logics:
  - (Propositional) Linear Time Logic LTL [Pnu77]
  - Computation Tree Logic CTL [CE81, EH82]
  - CTL* [EH86]
  - (Propositional) μ-calculus L_μ [Koz83]
- They split into two families:
  - Linear-time temporal logics (e.g. LTL, linear-time μ-calculus)
  - Branching-time temporal logics (e.g. CTL, CTL*, L_μ)

  Read the Handbook Chapter by [Eme90].
- Differences are:
  - The former refers to a fixed path/branch in the transition system/computation tree
  - The latter refers to a state/node in the transition system/computation tree

Temporal logics

- Classic temporal logics:
  - (Propositional) Linear Time Logic LTL [Pnu77]
  - Computation Tree Logic CTL [CE81, EH82]
  - CTL^* [EH86]
  - (Propositional) μ-calculus L_μ [Koz83]
- They split into two families:
  - Linear-time temporal logics (e.g. LTL, linear-time μ-calculus)
  - Branching-time temporal logics (e.g. CTL, CTL^*, L_μ)

  Read the Handbook Chapter by [Eme90].
- Differences are:
  - The former refers to a fixed path/branch in the transition system/computation tree
  - The latter refers to a state/node in the transition system/computation tree

Linear-time logic: LTL

Definition

- Syntax

$$\psi, \psi_1, \psi_2\, (\in \mathsf{LTL}) ::= p \mid \psi_1 \wedge \psi_2 \mid \neg\psi \mid \mathbf{X}\,\psi \mid \psi_1\,\mathbf{U}\,\psi_2$$

  where $p \in Prop$

- Semantics: let t be a computation tree and π be a path in t.

$$\begin{array}{lll} t,\pi \models p & \text{iff} & p \in \ell(x) \\ t,\pi \models \psi_1 \wedge \psi_2 & \text{iff} & t,\pi \models \psi_1 \text{ and } t,\pi \models \psi_2 \\ t,\pi \models \neg\psi_1 & \text{iff} & t,\pi \not\models \psi_1 \\ t,\pi \models \mathbf{X}\,\psi & \text{iff} & t,\pi[1\ldots] \models \psi \\ t,\pi \models \psi_1\,\mathbf{U}\,\psi_2 & \text{iff} & \begin{array}{l} \exists j \in \mathbb{N}, t,\pi[j\ldots] \models \psi_2 \text{ and} \\ \forall 0 \le i < j, t,\pi[i\ldots] \models \psi_1 \end{array} \end{array}$$

Use $\mathbf{F}\varphi$ for true $\mathbf{U}\,\varphi$ and $\mathbf{G}\varphi$ for $\neg\mathbf{F}\neg\varphi$.

LTL expressiveness

We refer to [EH86, Eme90] for a study of LTL expressiveness.

Theorem ([Kam68])

LTL $\equiv$ FOLLO (the firt order logic of linear orders).

LTL can also be interpreted over computation trees:

Definition (LTL interpretation over computation trees)

Let $\varphi \in$ LTL, $t \models \varphi$ if formula φ holds for every path in t.

Example

- Writing $\overset{\infty}{\mathbf{F}}\varphi$ for $\mathbf{GF}\varphi$, the formula $\overset{\infty}{\mathbf{F}}p$ expresses that p occurs infintely often.
- The property that all even times $(0, 2, 4, 6, \ldots)$, p is true is not expressible in LTL.

Branching-time logic: CTL

Definition

- Syntax ($p \in Prop$)

$$\varphi, \varphi_1, \varphi_2 (\in \text{CTL}) ::= p \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \mathbf{AX}\,\varphi \mid \mathbf{A}\,\varphi_1\,\mathbf{U}\,\varphi_2 \mid \mathbf{E}\,\varphi_1\,\mathbf{U}\,\varphi_2$$

- Semantics: let t be a computation tree over $Prop$ and Ev, and let $x \in t$ be a node.

$t, x \models p$	iff	$p \in \ell(x)$
$t, x \models \varphi_1 \wedge \varphi_2$	iff	$t, x \models \varphi_1$ and $t, x \models \varphi_2$
$t, x \models \neg\varphi$	iff	$t, x \not\models \varphi$
$t, x \models \mathbf{AX}\,\varphi$	iff	$t, x' \models \varphi$, for all x' s.t. $x \to x'$
$t, x \models \mathbf{A}\,\varphi_1\,\mathbf{U}\,\varphi_2$	iff	$t_{\mid x}, \pi \models \varphi_1\,\mathbf{U}\,\varphi_2$ for all path π of t that starts in x
$t, x \models \mathbf{E}\,\varphi_1\,\mathbf{U}\,\varphi_2$	iff	$t_{\mid x}, \pi \models \varphi_1\,\mathbf{U}\,\varphi_2$ for some path π of t that starts in x

More on CTL

- Notations:

$$\begin{aligned} \mathbf{EX}\,\varphi &:= \neg\mathbf{AX}\,\neg\varphi, \\ \mathbf{EF}\,\varphi &:= \mathbf{E}\,\mathtt{true}\,\mathbf{U}\,\varphi, \\ \mathbf{AG}\,\varphi &:= \neg\mathbf{EF}\,\neg\varphi, \\ \mathbf{AF}\,\varphi &:= \mathbf{A}\,\mathtt{true}\,\mathbf{U}\,\varphi, \\ \mathbf{EG}\,\varphi &:= \neg\mathbf{AF}\,\neg\varphi \end{aligned}$$

Example

- The CTL formula $\mathbf{AF}\,\mathbf{AG}\,p$ is not expressible in LTL.
- The LTL formula $\overset{\infty}{\mathbf{F}}p$ is not expressible in CTL.

CTL* merges CTL and LTL

- Syntax

State formulas: $\varphi ::= p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \mathbf{A}\,\psi$

Path formulas: $\psi ::= \varphi \mid \neg\psi \mid \psi_1 \wedge \psi_2 \mid \mathbf{X}\,\psi \mid \psi\,\mathbf{U}\,\psi,$

- Semantics

$t, x \models p$	iff	$p \in \ell(x)$
$t, x \models \varphi_1 \wedge \varphi_2$	iff	$t, x \models \varphi_1$ and $t, x \models \varphi_2$
$t, x \models \neg\varphi_1$	iff	$t, x \not\models \varphi_1$
$t, x \models \mathbf{A}\,\psi$	iff	$t, \pi \models \psi$

$t, \pi \models \varphi$	iff	$t, \pi[0] \models \varphi$
$t, \pi \models \psi_1 \wedge \psi_2$	iff	$t, \pi \models \psi_1$ and $t, \pi \models \psi_2$
$t, \pi \models \neg\psi_1$	iff	$t, \pi \not\models \psi_1$
$t, \pi \models \mathbf{X}\,\psi$	iff	$t, \pi[1\ldots] \models \psi$
$t, \pi \models \psi_1\,\mathbf{U}\,\psi_2$	iff	$\exists j \in \mathbb{N}, t, \pi[j\ldots] \models \psi_2$ and $\forall 0 \leq i < j, t, \pi[i\ldots] \models \psi_1$

Linear-time vs. Branching-time: expressiveness

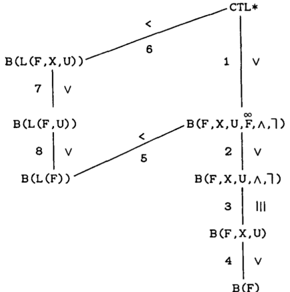
Theorem ([EH86])

LTL and CTL are expressively incomparable.

As intended, CTL* subsumes LTL and CTL, as they are fragments.

Theorem ([EH86])

CTL* is strictly more expressive than LTL and CTL, by considering for example formula $(\mathbf{AF}\,\mathbf{G}p) \vee (\mathbf{AG}\,\mathbf{EF}\,q)$.



Temporal logics and bisimulation

Fix some logic $\mathcal{L}$ interpreted over transtion systems.

Definition

$\mathcal{L}$ is *bisimulation invariant* if any two bisimlar pointed transtion systems satisfy the same formulas of $\mathcal{L}$.

Write $\mathcal{S}_1, s_1^0 \equiv_{\mathcal{L}} \mathcal{S}_2, s_2^0$ if $\mathcal{S}_1, s_1^0 \models \varphi$ iff $\mathcal{S}_2, s_2^0 \models \varphi$, $\forall \varphi \in \mathcal{L}$.

Theorem

For every $\mathcal{L} \in \{\mathsf{LTL}, \mathsf{CTL}, \mathsf{CTL}^*\}$, and every pair of pointed transition systems $\mathcal{S}_1, s_1^0$ and $\mathcal{S}_2, s_2^0$,

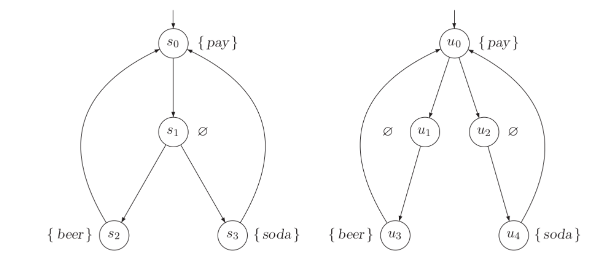
$$\mathcal{S}_1, s_1^0 \text{ and } \mathcal{S}_2, s_2^0 \text{ are bisimilar implies } \mathcal{S}_1, s_1^0 \equiv_{\mathcal{L}} \mathcal{S}_2, s_2^0$$

Remark

If moreover $\mathcal{S}_1, s_1^0$ and $\mathcal{S}_2, s_2^0$ are finitely branching, then the converse also holds for $\mathcal{L} \in \{\mathsf{CTL}, \mathsf{CTL}^*\}$. See for example [Sti98].

Temporal logics and bisimulation

Exercise



Find a CTL formula which distinguishes the two models.

Exercise

- Find out which theorem holds for $\equiv_{\text{LTL}}$.
- Justify why the two models above have the same LTL theory.

Decision problems

Theorem

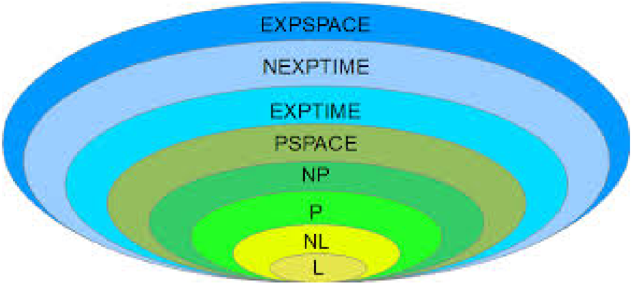
- [SC85] Satisfiability of LTL is PSPACE-complete.
- [FL79] Satisfiability of CTL is EXPTIME-complete.
- [VS85] Satisfiability of CTL* is 2EXPTIME-complete.

Regarding model-checking issues, we recommend [Sch02].

Theorem

- The Model-Checking problem for LTL is PSPACE-complete.
- The Model-Checking problem for CTL is PTIME-complete..
- The Model-Checking problem for CTL* is PSPACE-complete.

Recall on complexity classes



Content

Epistemic logic

Definition

- Syntax

$$\varphi, \varphi_1, \varphi_2 (\in \mathbf{K}) ::= p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid K_a \varphi$$

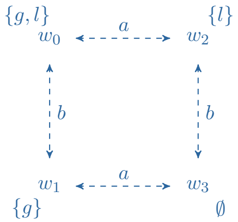
  where $p \in Prop$ and $a \in Ag$.

- Semantics: let $\mathcal{M} = \langle W, \mathcal{R}, \ell \rangle$ be an epistemic model over $Prop$ and Ag, and let $w \in W$.

$$\begin{array}{lll}
\mathcal{M}, w \models p & \text{iff} & p \in \ell(w) \\
\mathcal{M}, w \models \varphi_1 \wedge \varphi_2 & \text{iff} & \mathcal{M}, w \models \varphi_1 \text{ and } \mathcal{M}, w \models \varphi_2 \\
\mathcal{M}, w \models \neg\varphi & \text{iff} & \mathcal{M}, w \not\models \varphi \\
\mathcal{M}, w \models K_a\varphi & \text{iff} & \mathcal{M}, w' \models \varphi, \text{ for every } w \overset{a}{\hookrightarrow} w'
\end{array}$$

The GLO-scenario

$\mathcal{M} = \langle W, \mathcal{R}, \ell \rangle$



Example

- $\mathcal{M}, w_0 \models K_b g \wedge \neg K_b l \wedge \neg K_b \neg l$, that is agent b knows that it is sunny in Groningen, but does not know whether it is sunny in Liverpool or not. This is also the case in every other world.
- $\mathcal{M}, w_0 \models K_b \neg K_b l$. Agent b knows about his ignorance.
- $\mathcal{M}, w_0 \models \neg K_a g \wedge \neg K_a \neg g \wedge K_a(K_b g \vee K_b \neg g)$

More on epistemic logic

Remark

The logical omniscience "problem": agents know all validities (see for example [vDvdHK07, p. 23]

Theorem ([BdRV01])

Logic **K** is bisimulation invariant.

Remark

On finite image epitemic models, $\equiv_{\mathbf{K}}$ coincides with bisimilarity, with the same proof as for temporal logic.

On the GLO-scenario, formalize the following claims:

1. In state w_0, agent b considers it possible that it is sunny in Groningen, and also that it is sunny in Liverpool, and also that it is not sunny in Liverpool.
2. In w_2, agent b knows it is not sunny in Groningen, although he does not know it is sunny in Liverpool.
3. In state w_0, agent b knows both that he knows that it is sunny in Groningen and that he does not know that it is sunny in Liverpool.
4. It is true that agent b knows whether it is sunny in Groningen, but he does not know whether it is sunny in Liverpool.
5. In any world, any agent knows that any fact or its negation holds.
6. It is not a validity that an agent always knows either a fact, or that he knows its negation.

Decision problems in epistemic logic

Theorem

Satisfiability of **K** is PSPACE-complete.

See [BdRV01], [vDvdHK07, Chap. 2]

Theorem

The Model-Checking problem for **K** is in PTIME-complete.

See [Sch02, Sec. 3.2.1] which is the only proof I know.

Content

Content

Epistemic Temporal Logic

Definition

- Syntax
  - State formulas

$$\varphi, \varphi_1, \varphi_2\,(\in \mathsf{ETL}) ::= p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \mathbf{A}\,\psi \mid K_a\varphi$$

  where $p \in Prop$ and $a \in Ag$.

  - Path formulas (as for CTL*)

$$\psi ::= \varphi \mid \neg\psi \mid \psi_1 \wedge \psi_2 \mid \mathbf{X}\,\psi \mid \psi\,\mathbf{U}\,\psi,$$

- Semantics

Exercise

Your job! Combine the semantics of CTL* and **K**.

Practicing

Example

Formula $\mathbf{AG}\,(p \;\Rightarrow\; K_a p)$ being true at the root of some computation tree t of the forest means that:

for every node x of t where p holds, all the nodes *in the forest* that are $\overset{a}{\hookrightarrow}$-related to x are also labelled p.

Exercise

Draw a picture for formula $\mathbf{AG}\,(p \;\Rightarrow\; K_a p)$.

Results for ETL

[HVDMV04] considers

- CKL_m for LTL + all knowledge operators including common knowledge and fragment KL_m of ETL

	$CKL_m,\ m \geq 2$	$KL_m,\ m \geq 2$	KL_1
$\mathcal{C}_m^{pr}$, $\mathcal{C}_m^{pr,sync}$, $\mathcal{C}_m^{pr,uis}$, $\mathcal{C}_m^{pr,sync,uis}$	Π_1^1	nonelementary time $ex(ad(\varphi)+1, c\lvert\varphi\rvert)$	double-exponential time
$\mathcal{C}_m^{nl}$, $\mathcal{C}_m^{nl,pr}$, $\mathcal{C}_m^{nl,pr,sync}$, $\mathcal{C}_m^{nl,sync}$	Π_1^1	nonelementary space $ex(ad(\varphi), c\lvert\varphi\rvert)$	EXPSPACE
$\mathcal{C}_m^{nl,pr,uis}$	Π_1^1	Π_1^1	EXPSPACE
$\mathcal{C}_m^{nl,uis}$	co-r.e.	co-r.e.	EXPSPACE
$\mathcal{C}_m^{nl,sync,uis}$, $\mathcal{C}_m^{nl,pr,sync,uis}$	EXPSPACE	EXPSPACE	EXPSPACE
$\mathcal{C}_m$, $\mathcal{C}_m^{sync}$, $\mathcal{C}_m^{sync,uis}$, $\mathcal{C}_m^{uis}$	EXPTIME	PSPACE	PSPACE

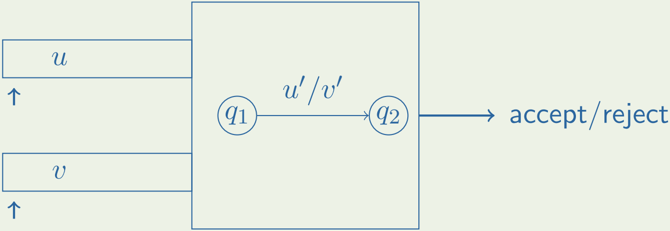
  where $\mathcal{C}_m$ is the class of Interpreted Systems with $|Ag| = m$,

  and superscript means nl, pr, $sync$, and uis to *no learning*, *perfect recall*, *synchronous*, *unique initial state* (one tree in the forest) respectively.
- Many scattered results on model-checking: [vdMS99],[VDHW02],[MP13] . . .

Transducer

Definition

A *transducer* over an alphabet $\Sigma = \{\alpha, \beta, \dots\}$ is a (non-deterministic) finite state machine with two tapes ...



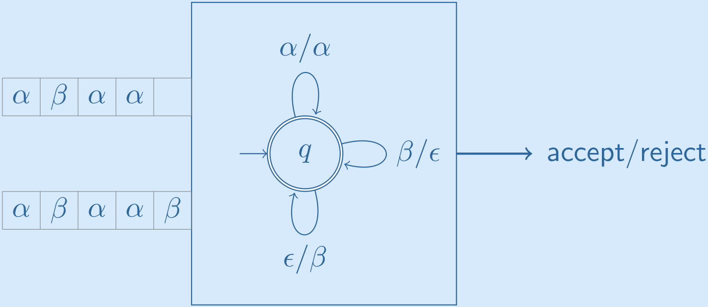
$\mathcal{L}(T) = \{(u, v) \mid T$ has an accepting run with input pair $(u, v)\}$

Exercise

Take a look at [Ber79, Sec. 6] and also [Eil74] to complete the definition of a transducer.

An example of transducer

Example



$\{(\alpha^k, \alpha^k) \mid k \in \mathbb{N}\} \subseteq \mathcal{L}(T)$, $(\beta\alpha\beta\alpha, \beta\alpha\alpha\beta) \in \mathcal{L}(T)$
$(\alpha^2, \alpha) \notin \mathcal{L}(T)$

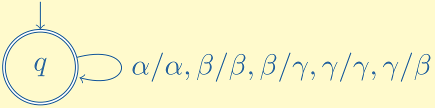
Exercise

Show that $\mathcal{L}(T) = \{(u, v) \mid |u|_\alpha = |v|_\alpha\}$

Exercises on transducers

Exercise

What does this one recognizes?
(i.e. what is $\mathcal{L}(T)$?)



Exercise

Can you find a transducer for the language below?

$$\{(u,v) \mid |u|_\alpha = |v|_\alpha \text{ and } |u| = |v|\}$$

Exercise

Would you say that $\mathcal{L}(T)$ from previous slide is synchronous? perfect recall? Justify.

Some decision problems on rational relations

Theorem

Given two rational relations $\mathcal{R}, \mathcal{R}' \subseteq \Sigma^* \times \Sigma^*$.

- Decidable problems
  - $\mathcal{R} = \emptyset$?
  - Is $\mathcal{R}$ finite?
- Undecidable problems
  - $\mathcal{R} \cap \mathcal{R}' = \emptyset$?
  - $\mathcal{R} \subseteq \mathcal{R}'$?
  - $\mathcal{R} = \mathcal{R}'$?
  - $\mathcal{R} = \Sigma^* \times \Sigma^*$?
  - Is $\Sigma^* \times \Sigma^* \setminus \mathcal{R}$ finite?

We recommend to read [Ber79, Chap. 8].

Closure properties of rational relations

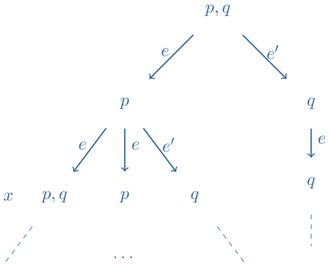
Theorem

- The class of rational relations is closed under union but not closed under intersection.
- The class of rational relations is closed under composition.
- Equivalence of rational relations is undecidable, but equivalence of deterministic rational relations is decidable.

See for example [CDG+97, Ber79]

Transducers for Knowledge semantics

In an ETL frame $\langle \mathcal{F}, \{\overset{a}{\hookrightarrow}\}_{a \in Ag} \rangle$, we associate to a node/history x is a sequence/word over alphabet $2^{Prop}.(Ev \times 2^{Prop})^*$, called *the word of* x and written $\mathsf{w}(x)$.



$$\mathsf{w}(x) = \{p, q\}e\{p\}e\{p, q\}$$

From this abstract information, we derive a binary relation between nodes according to

$$x \overset{a}{\hookrightarrow} y \text{ if } \mathsf{w}(x) \overset{a}{\hookrightarrow} \mathsf{w}(y)$$

Rational ETL frames

Definition

An ETL frame $\langle \{t^j\}_{j\in J}, \{\overset{a}{\hookrightarrow}\}_{a\in Ag}\rangle$ is *rational* if each $\overset{a}{\hookrightarrow}$ is a rational relation.

Exercise

What do you think of the following decision problems, where $\mathcal{R}$ rational relation?

- Is $\mathcal{R}$ perfect recall?
- Is $\mathcal{R}$ synchronous?
- Is $\mathcal{R}$ perfect recall?

Exercise

From theorem above, try to understand why distributed knowledge might cause severe problems from a computational point of view.

Content

Dynamic epistemc logics

Introduced by Baltag, Moss and Solecki, Slawomirin in "*The logic of public announcements, common knowledge, and private suspicions* [BMS98]".

- It contains a modility for epitemic event models.
- It contains more that we will not discuss.
- The logic of *Public Announcement* is a fragment.
- It is not more expressive than **K**

Definition of DEL

Definition

- Syntax

$$\varphi, \varphi_1, \varphi_2 \, (\in \text{DEL}) ::= p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid K_a\varphi \mid [\mathcal{E}, e]\varphi$$

  where $p \in Prop$, $a \in Ag$ and $\mathcal{E}, e$ is a (finite) epistemic event.

- Semantics: let $\mathcal{M} = \langle W, \mathcal{R}, \ell \rangle$ be an epitemic model over $Prop$ and Ag.

$$\begin{array}{lll}
\mathcal{M}, w \models p & \text{iff} & \ell(\text{w}) \\
\mathcal{M}, w \models \varphi_1 \wedge \varphi_2 & \text{iff} & \mathcal{M}, w \models \varphi_1 \text{ and } \mathcal{M}, w \models \varphi_2 \\
\mathcal{M}, w \models \neg\varphi & \text{iff} & \mathcal{M}, w \not\models \varphi \\
\mathcal{M}, w \models K_a\varphi & \text{iff} & \mathcal{M}, w \models \varphi, \text{ for every } w \overset{a}{\hookrightarrow} w' \\
\mathcal{M}, w \models [\mathcal{E}, e]\varphi & \text{iff} & \mathcal{M} \otimes \mathcal{E}, we \models \varphi
\end{array}$$

Public Announcement Logic PAL [Pla07]

Definition

- Syntax

$$\varphi, \psi, \varphi_1, \varphi_2\,(\in \mathsf{PAL}) ::= p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid [\psi!]\varphi$$

  where $p \in Prop$.

- Semantics: let $\mathcal{M} = \langle W, \mathcal{R}, \ell\rangle$ be an epitemic model over $Prop$ and Ag.

$$\begin{array}{lll}
\mathcal{M}, w \models p & \text{iff} & \\
\mathcal{M}, w \models \varphi_1 \wedge \varphi_2 & \text{iff} & \mathcal{M}, w \models \varphi_1 \text{ and } \mathcal{M}, w \models \varphi_2 \\
\mathcal{M}, w \models \neg\varphi & \text{iff} & \mathcal{M}, w \not\models \varphi \\
\mathcal{M}, w \models [\psi!]\varphi & \text{iff} & \mathcal{M}, w \models \psi \text{ and } \mathcal{M}_\psi, w \models \varphi
\end{array}$$

  where $\mathcal{M}_\psi$ is the restriction of $\mathcal{M}$ to the set of worlds that satisfy ψ

PAL $\hookrightarrow$ DEL

What do you think?

Lemma

$$\mathcal{M}, w \models [\psi!]\varphi \text{ iff } \mathcal{M}, w \models [\mathcal{E}_\psi, e_\psi]\varphi$$

where $\mathcal{E}_\psi$ is

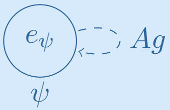
PAL $\hookrightarrow$ DEL

What do you think?

Lemma

$$\mathcal{M}, w \models [\psi!]\varphi \text{ iff } \mathcal{M}, w \models [\mathcal{E}_\psi, e_\psi]\varphi$$

where $\mathcal{E}_\psi$ is



Muddy children

Exercise

Try to formalize the muddy children puzzle in PAL or DEL.

Results on DEL and PAL

- Expressiveness

Theorem ([Pla07],[Lut06])

PAL is not more expressive than **K** (but it is exponentially more succinct).
DEL is not more expressive than **K** (but it is exponentially more succinct).

- Satisfiability

Theorem ([Lut06],[AS13])

The satisfiability in single-agent PAL is NPTIME-complete.
The satisfiability problem for PAL is PSPACE-complete.
The satisfiability problem for DEL is in $\mathrm{NEXPTIME}$-complete.

Model-checking DEL and PAL

Theorem ([KvB04])

The model-checking problem for PAL is in PTIME.

Theorem ([AS13])

The model-checking for DEL is PSPACE-complete.

R. Alur, T.A. Henzinger, and O. Kupferman, *Alternating-time temporal logic*, Journal of the ACM **49** (2002), no. 5, 672–713.

Guillaume Aucher, Bastien Maubert, and Sophie Pinchinat, *Automata techniques for epistemic protocol synthesis*, Proceedings 2nd International Workshop on Strategic Reasoning, SR 2014, Grenoble, France, April 5-6, 2014 (Fabio Mogavero, Aniello Murano, and Moshe Y. Vardi, eds.), EPTCS, vol. 146, 2014, pp. 97–103.

A. Arnold, *Systèmes de transitions finis et sémantique de processus communicants*, Masson, 1992, Available as Finite transition systems: semantics of communicating systems. Prentice Hall, 1994.

Guillaume Aucher and François Schwarzentruber, *On the complexity of dynamic epistemic logic*, TARK-Theoretical Aspects of Rationality and Knowledge-2013, 2013.

- B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and Ph. Schnoebelen, *Systems and software verification. model-checking techniques and toolmodel checking*, Springer-Verlag, 1999.
- Patrick Blackburn, Maarten de Rijke, and Yde Venema, *Modal logic, volume 53 of*, Cambridge Tracts in Theoretical Computer Science (2001), 29.
- Jean Berstel, *Transductions and context-free languages*, Teubner Studienbucher, 1979.
- C. Baier and J.P. Katoen, *Principles of model checking*, vol. 26202649, MIT Press, 2008.
- Alexandru Baltag, Lawrence S Moss, and Slawomir Solecki, *The logic of public announcements, common knowledge, and private suspicions*, Proceedings of the 7th conference on Theoretical aspects of rationality and knowledge, Morgan Kaufmann Publishers Inc., 1998, pp. 43–56.

- H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi, *Tree automata techniques and applications*, 1997, October, 2002.
- E. M. Clarke and E. A. Emerson, *Design and synthesis of synchronization skeletons using branching time temporal logic*, May 1981, pp. 52–71.
- E. A. Emerson and J. Y. Halpern, *Decision procedures and expressiveness in the temporal logic of branching time*, May 1982, pp. 169–180.
- ______, *"Sometimes" and "Not Never" revisited: On branching versus linear time temporal logic*, no. 1, 151–178.
- Samuel Eilenberg, *Automata, languages, and machines, volume A*, Academic Press, New York, 1974.

E. A. Emerson, *Temporal and modal logic*, Handbook of Theoretical Computer Science, vol. B (J. van Leeuwen, ed.), 1990, pp. 995–1072.

M. J. Fischer and R. E. Ladner, *Propositional dynamic logic of regular programs*, 194–211.

Ronald Fagin, Yoram Moses, Joseph Y Halpern, and Moshe Y Vardi, *Reasoning about knowledge*, MIT press, 2003.

Joseph Y Halpern and Ronald Fagin, *Modelling knowledge and action in distributed systems*, Distributed computing **3** (1989), no. 4, 159–177.

Jaakko Hintikka, *Knowledge and belief: an introduction to the logic of the two notions*, vol. 181, Cornell University Press Ithaca, 1962.

Joseph Y Halpern and Moshe Y Vardi, *The complexity of reasoning about knowledge and time*, Proceedings of the eighteenth annual ACM symposium on Theory of computing, ACM, 1986, pp. 304–315.

Joseph Y Halpern, Ron Van Der Meyden, and Moshe Y Vardi, *Complete axiomatizations for reasoning about knowledge and time*, SIAM Journal on Computing **33** (2004), no. 3, 674–703.

J. A. W. Kamp, *Tense logic and the theory of linear order*, Ph.D. thesis, UCLA, Los Angeles, 1968.

D. Kozen, *Results on the propositional μ-calculus*, no. 3, 333–354.

Barteld Kooi and Johan van Benthem, *Reduction axioms for epistemic actions*, AiML-2004: Advances in Modal Logic, Department of Computer Science, University of Manchester, Technical report series, UMCS-04-9-1 (2004), 197–211.

Carsten Lutz, *Complexity and succinctness of public announcement logic*, Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, ACM, 2006, pp. 137–143.

R. Milner, *Calculi for synchrony and asynchrony*, no. 3, 267–310.

Bastien Maubert and Sophie Pinchinat, *Jumping automata for uniform strategies.*, FSTTCS, vol. 13, 2013, pp. 287–298.

D. Park, *Concurrency and automata on infinite sequences*, March 1981, pp. 167–183.

Jan Plaza, *Logics of public communications*, Synthese **158** (2007), no. 2, 165–179.

A. Pnueli, *The temporal logic of programs*, November 1977, pp. 46–57.

A. P. Sistla and E. M. Clarke, *The complexity of propositional linear temporal logics*, no. 3, 733–749.

- Philippe Schnoebelen, *The complexity of temporal logic model checking.*, Advances in modal logic **4** (2002), no. 393-436, 35.
- Colin Stirling, *The joys of bisimulation*, Mathematical Foundations of Computer Science 1998, Springer, 1998, pp. 142–151.
- Wolfgang Thomas, *Automata on infinite objects*, Handbook of theoretical computer science **2** (1990).
- Johan van Benthem and Fenrong Liu, *Diversity of logical agents in games*, Philosophia Scientiae **8** (2004), no. 2, 163–178.
- Wiebe Van Der Hoek and Michael Wooldridge, *Model checking knowledge and time*, Model Checking Software, Springer, 2002, pp. 95–111.
- Ron van der Meyden and Nikolay V Shilov, *Model checking knowledge and time in systems with perfect recall*,

Foundations of Software Technology and Theoretical Computer Science, Springer, 1999, pp. 432–445.

Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Pieter Kooi, *Dynamic epistemic logic*, vol. 337, Springer Science & Business Media, 2007.

Moshe Y Vardi and Larry Stockmeyer, *Improved upper and lower bounds for modal logics of programs*, Proceedings of the seventeenth annual ACM symposium on Theory of computing, ACM, 1985, pp. 240–251.