INFOMLSAI Cheat Sheet for the Exam

Temporal Logics (TL)

Definition (State Transition System)

Definition (State Transition System)

An state transition system is a triple

$$\langle St, \longrightarrow, \mathcal{V} \rangle$$

where:

- St is a non-empty set of states,
- $\longrightarrow \subseteq St \times St$ is a transition relation
- $\mathcal{V} : \mathcal{PV} \rightarrow 2^{St}$ is a valuation function that assigns to each proposition a set of states where it is true ($\mathcal{PV}$ is the set of propositions).

Paths (in a transition system)

Definition (Paths in a transition system)

A path λ in $\langle St, \longrightarrow, \mathcal{V} \rangle$ is a sequence of states from St, $q_0 q_1 q_2 \ldots$, such that for each q_i and q_{i+1}, $q_i \longrightarrow q_{i+1}$.

A path must be full, i.e. either infinite, or ending in a state with no outgoing transition.

In this course, unless stated otherwise, each state has an outgoing transition and the paths are infinite in the coursework, we introduce a finite path version of one of the logics.

Temporal operators

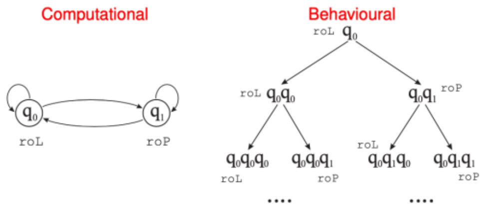
$X\varphi$	φ is true in the next moment in time
$G\varphi$	φ is true in all future moments
$F\varphi$	φ is true in some future moment
$\varphi U \psi$	φ is true until the moment when ψ becomes true

Properties

- Safety property – usually with G
- Liveliness property – usually with F
- Fairness property – usually with GF

Linear Temporal Logics (LTL)

Definition (Models of LTL)



- In LTL, models are defined as behavioral structures!
- ...But input to the verification problem is defined by the computational structure.

Definition (Models of LTL)

A **model of LTL** is a sequence of time moments (states). We call such models **paths**, and denote them by λ.

Evaluation of atomic propositions at particular time moments is also needed.

Notation:

- $\lambda[i]$: ith time moment (starting from 0)
- $\lambda[i \dots j]$: all time moments between i and j
- $\lambda[i \dots \infty]$: all timepoints from i on

Definition (Semantics of LTL)

Definition (Semantics of LTL)

$\lambda \models p$	iff p is true at moment $\lambda[0]$ (that is, $\lambda[0] \in \mathcal{V}(p)$);
$\lambda \models X\varphi$	iff $\lambda[1..\infty] \models \varphi$;
$\lambda \models F\varphi$	iff $\lambda[i..\infty] \models \varphi$ for some $i \geq 0$;
$\lambda \models G\varphi$	iff $\lambda[i..\infty] \models \varphi$ for all $i \geq 0$;
$\lambda \models \varphi \, U \, \psi$	iff $\lambda[i..\infty] \models \psi$ for some $i \geq 0$, and $\lambda[j..\infty] \models \varphi$ for all $0 \leq j < i$.

$\lambda \models \neg\varphi$	iff not $\lambda \models \varphi$;
$\lambda \models \varphi \wedge \psi$	iff $\lambda \models \varphi$ and $\lambda \models \psi$.

Note that:

$$G\varphi \equiv \neg F \neg \varphi$$

$$F\varphi \equiv \neg G \neg \varphi$$

$$F\varphi \equiv \top \, U \, \varphi$$

Definition (Semantics of LTL in Transition Systems)

Truth in transition systems is verified as follows

Definition (Semantics of LTL in Transition Systems)

$M, q \models \varphi$ iff $\lambda \models \varphi$ for every path λ in M starting from q.

In Model (or transition system) and state q,
LTL formula φ is true if and only if
for every path starting in q this path satisfies φ.

Definition (Model checking problem for LTL)

Definition (Model checking problem for LTL)

Given a finite state transition system M, a state q in M, and an LTL formula φ, check whether $M, q \models \varphi$.

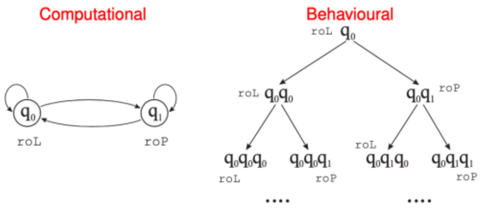
Remember that we need to check whether $\lambda \models \varphi$ for every path λ in M starting from q.

Complexity

- LTL model checking is usually done using Büchi automata (automata over infinite strings)
  - Given M, q and ϕ, two automata are constructed:
  - A¬ϕ that accepts all paths satisfying ¬ϕ
  - A*M*,q that accepts all paths in M starting from q
- the non-emptiness check is done by constructing a product automaton of A¬ϕ and AM,q and checking non-emptyness of its language (the paths it accepts)
- the non-emptyness check can be done in linear time in the size of the product automaton
- AM,q is of size linear in |M|, but unfortunately A¬ϕ is exponential in |ϕ|
- so the whole procedure is polynomial in |M| but exponential in |ϕ|
- the problem itself is PSPACE-complete, so it is very unlikely one can do much better

Computational Tree Logic(CTL/CTL*)

Definition (Models of CTL)



- **In CTL/CTL*, models are defined as computational structures!**
- Reasoning about all possible computations of a system
- Path quantifiers: A (for all paths), E (there is a path)
- Temporal operators: X (next), F (sometime), G (always), U (until)
- “Vanilla” CTL: every temporal operator must be immediately preceded by exactly one path quantifier
- CTL*: no syntactic restrictions; two kinds of formulas: state formulas vs. path formulas
- model checking CTL is much easier
- Models: transition systems; include: states (time points, situations), transitions (changes)
- Paths: full sequences of states that can be produced by following transitions in the transition system

Definition (Semantics of CTL*: state formulae)

Definition (Semantics of CTL*: state formulae)

$M, q \models E\gamma$ iff there is a path λ, starting from q, such that $M, \lambda \models \gamma$;
$M, q \models A\gamma$ iff for all paths λ, starting from q, we have $M, \lambda \models \gamma$.

Definition (Semantics of CTL*: path formulae)

$M, \lambda \models \varphi$ iff $M, \lambda[0] \models \varphi$, for a state formula φ;
$M, \lambda \models X\varphi$ iff $M, \lambda[1...\infty] \models \varphi$;
$M, \lambda \models F\varphi$ iff $M, \lambda[i..\infty] \models \varphi$ for some $i \geq 0$;
$M, \lambda \models G\varphi$ iff $M, \lambda[i..\infty] \models \varphi$ for all $i \geq 0$;
$M, \lambda \models \varphi\, U\, \psi$ iff $M, \lambda[i...\infty] \models \psi$ for some $i \geq 0$, and $M, \lambda[j...\infty] \models \varphi$ for all $0 \leq j < i$.

Alternative Semantics for “Vanilla” CTL

The semantics of **“vanilla” CTL** can be given entirely with respect to states in the model:

$M, q \models EX\varphi$ iff there is a path λ starting from q, such that $M, \lambda[1] \models \varphi$;

$M, q \models EF\varphi$ iff there is a path λ starting from q, such that $M, \lambda[i] \models \varphi$ for some $i \geq 0$;

$M, q \models EG\varphi$ iff there is a path λ starting from q, such that $M, \lambda[i] \models \varphi$ for all $i \geq 0$;

$M, q \models E\varphi\, U\psi$ iff there is a path λ starting from q, such that $M, \lambda[i] \models \psi$ for some $i \geq 0$, and $M, \lambda[j] \models \varphi$ for all $0 \leq j < i$.

...and analogously for AX, AF, AG, AU.

Fixpoint Equivalences in CTL

Theorem (Fixpoint characterization of branching-time operators)

*The following formulae are **valid** in CTL:*

- $EF\varphi \quad \leftrightarrow \quad \varphi \vee EX\, EF\varphi$
- $EG\varphi \quad \leftrightarrow \quad \varphi \wedge EX\, EA\varphi$
- $E\varphi_1\, U\, \varphi_2 \quad \leftrightarrow \quad \varphi_2 \vee (\varphi_1 \wedge EX\, E\varphi_1\, U\, \varphi_2)$.

- $AF\varphi \quad \leftrightarrow \quad \varphi \vee AX\, AF\varphi$
- $AG\varphi \quad \leftrightarrow \quad \varphi \wedge AX\, AG\varphi$
- $A\varphi_1\, U\, \varphi_2 \quad \leftrightarrow \quad \varphi_2 \vee (\varphi_1 \wedge AX\, A\varphi_1\, U\, \varphi_2)$.

Model checking CTL

Global model checking

We want to implement the function of two arguments M and ϕ

$$mcheck(M, \varphi) = \{q \in St \mid M, q \models \varphi\}$$

set of states q where
ϕ is true with q in the model M

function MCHECK(M, φ_0)
  for $\varphi' \in Sub(\varphi_0)$ **do**
    case $\varphi' = p$
      $[\varphi']_M \leftarrow \mathcal{V}(p)$
    case $\varphi' = \neg\psi$
      $[\varphi']_M \leftarrow St \setminus [\psi]_M$
    case $\varphi' = \psi_1 \wedge \psi_2$
      $[\varphi']_M \leftarrow [\psi_1]_M \cap [\psi_2]_M$
    case $\varphi' = \psi_1 \vee \psi_2$
      $[\varphi']_M \leftarrow [\psi_1]_M \cup [\psi_2]_M$

case $\varphi' = EX\psi$
  $[\varphi']_M \leftarrow pre_{\exists}([\psi]_M)$

case $\varphi' = EG\psi$
  $Q_1 \leftarrow St; \quad Q_2 \leftarrow [\psi]_M$
  while $Q_1 \not\subseteq Q_2$ **do**
    $Q_1 \leftarrow Q_2; \quad Q_2 \leftarrow pre_{\exists}(Q_1) \cap Q_1$
  $[\varphi']_M \leftarrow Q_1$

case $\varphi' = E\psi_1\, U\, \psi_2$
  $Q_1 \leftarrow \emptyset; \quad Q_2 \leftarrow [\psi_2]_M$
  while $Q_2 \not\subseteq Q_1$ **do**
    $Q_1 \leftarrow Q_1 \cup Q_2; \quad Q_2 \leftarrow pre_{\exists}(Q_1) \cap [\psi_1]_M$
  $[\varphi']_M \leftarrow Q_1$

case $\varphi' = AX\psi$
  $[\varphi']_M \leftarrow pre_{\forall}([\psi]_M)$

Only difference with E is that
for A, we use the universal pre-image

For example, for AX's case as we want to
ensure that we only label states
where all the successors are ψ states

case $\varphi' = AG\psi$
  $Q_1 \leftarrow St; \quad Q_2 \leftarrow [\psi]_M$
  while $Q_1 \not\subseteq Q_2$ **do**
    $Q_1 \leftarrow Q_2; \quad Q_2 \leftarrow pre_{\forall}(Q_1) \cap Q_1$
  $[\varphi']_M \leftarrow Q_1$

case $\varphi' = A\psi_1\, U\, \psi_2$
  $Q_1 \leftarrow \emptyset; \quad Q_2 \leftarrow [\psi_2]_M$
  while $Q_2 \not\subseteq Q_1$ **do**
    $Q_1 \leftarrow Q_1 \cup Q_2; \quad Q_2 \leftarrow pre_{\forall}(Q_1) \cap [\psi_1]_M$
  $[\varphi']_M \leftarrow Q_1$

Theorem (Complexity of CTL Model Checking)

Model checking of CTL is P-complete, and can be done in time $O(m \cdot l)$ where m is the number of transitions in the model and the l is the number of subformulae in the formula.

Epistemic Logic (EL)

- Epistemic Logic (EL) is a logic for reasoning about knowledge
- its models are Kripke models
- they look very much like state transition systems, but instead of transition relation they have an indistinguishability relation (for every agent in the system)

Definition (Kripke model)

Definition (Kripke model)

Let $\mathcal{PV}$ be a set of atomic propositions $(p, q, r, \ldots)$ and Agt a finite set of agents.
A **Kripke model** $\mathcal{M} = \langle St, \{\sim_i (i \in Agt)\}, \mathcal{V}\rangle$ consists of

- a non-empty set of states St
- an indistinguishability relation $\sim_i$ for each agent i (which is reflexive, transitive and symmetric)
- a valuation of propositions $\mathcal{V} : \mathcal{PV} \to 2^{St}$

Definition (Semantic Clauses)

M,q |= K_iϕ iff ϕ holds in all worlds that for the agent i are indistinguishable from q

Definition (Semantic Clauses)

- $\mathcal{M}, q \models p$ iff $q \in \mathcal{V}(p)$;
- $\mathcal{M}, q \models \neg\varphi$ iff not $\mathcal{M}, q \models \varphi$;
- $\mathcal{M}, q \models \varphi \wedge \psi$ iff $\mathcal{M}, q \models \varphi$ and $\mathcal{M}, q \models \psi$;
- $\mathcal{M}, q \models K_i\varphi$ iff, for every $q' \in St$ such that $q \sim_i q'$, we have $\mathcal{M}, q' \models \varphi$.

Extra information on slides 19, 20, 21 in EL slides

- Properties of knowledge (Ki)
- Logical Omniscience
- Applications

Knowledge in groups

A group of agents A can "know" that ϕ in several different epistemic modes:

- E_Aϕ: everybody in A knows that ϕ (or: A have mutual knowledge that ϕ)
- C_Aϕ: it is a common knowledge among A that ϕ. Also check transitive closure.
- D_Aϕ: A have distributed knowledge that ϕ

Group Epistemic Logic (EL)

Definition of the syntax of ELCD and intuitions are described in group EL slides 3, 4, 5.

Group Knowledge: Semantics

or truth definitions

In state q, everybody in A knows ϕ if and only if for every state q prime which is indistinguishable from q by a group indistinguishability relation holds ϕ

- $\mathcal{M}, q \models E_A\varphi$ iff $\mathcal{M}, q' \models \varphi$ for every q' such that $q \sim_A^E q'$, where $\sim_A^E = \bigcup_{i \in A} \sim_i$

$$E_A\varphi = \bigwedge_{i \in A} K_i\varphi$$

- $\mathcal{M}, q \models C_A\varphi$ iff $\mathcal{M}, q' \models \varphi$ for every q' such that $q \sim_A^C q'$, where $\sim_A^C$ is the transitive closure of $\sim_A^E$

$$C_A\varphi = E_A\varphi \wedge E_AE_A\varphi \wedge E_AE_AE_A\varphi \wedge \ldots$$

- $\mathcal{M}, q \models D_A\varphi$ iff $\mathcal{M}, q' \models \varphi$ for every q' such that $q \sim_A^D q'$, where $\sim_A^D = \bigcap_{i \in A} \sim_i$

Computational Tree Logic with Knowledge (CTLK)

Local and Global States are described in CTLK slides 6, 7.

Definition (Interpreted System)

Definition (System)

A **system** is a set of runs.

Definition (Interpreted system)

An **interpreted system** $\mathcal{I}$ is a set of runs $\mathcal{R}$ plus valuation of propositions: $\mathcal{V} : PV \to 2^{St}$.

Semantics of CTLK in Interpreted Systems

Knowledge is interpreted as before:

- $\mathcal{I}, \langle r, m \rangle \models K_i\varphi$ iff $\mathcal{I}, \langle r', m' \rangle \models \varphi$ for every $\langle r', m' \rangle$ such that $\langle r, m \rangle \sim_i \langle r', m' \rangle$.
- similarly for E_A, C_A, D_A

Interpretation of temporal operators (AX, AU, AG are similar):

$\mathcal{I}, \langle r, m \rangle \models EX\varphi$ iff there is r' such that $r'[0...m] = r[0...m]$ and $\mathcal{I}, \langle r', m+1 \rangle \models \varphi$;

$\mathcal{I}, \langle r, m \rangle \models E\varphi U\psi$ iff there is r' such that $r'[0...m] = r[0...m]$ and $\mathcal{I}, \langle r', m' \rangle \models \psi$ for some $m' \geq m$ and $\mathcal{I}, \langle r', m'' \rangle \models \varphi$ for all m'' such that $m \leq m'' < m'$.

$\mathcal{I}, \langle r, m' \rangle \models EG\varphi$ iff there is r' such that $r'[0...m] = r[0...m]$ and $\mathcal{I}, \langle r', m' \rangle \models \varphi$ for all $m' \geq m$

Model Checking CTLK

- Model checking CTLK is essentially the same as model checking CTL, but with more complex states. Find the function in model checking CTLK slides 13-16
- It is more efficient to model-check dual versions of (group) knowledge modalities using only the existential pre-image. Find the function in model checking CTLK slides 24-27

Complexity

Theorem

Model checking of CTLK can be done in linear time with respect to the size of the Kripke model and the length of the formula and is P-complete.

Coalition Logic (CL)

Definition (Concurrent Game Structure)

Definition (Concurrent Game Structure)

A concurrent game structure is a tuple $M = \langle \mathbb{A}\mathrm{gt}, St, \mathcal{V}, Act, d, o \rangle$, where:

- $\mathbb{A}\mathrm{gt}$: a finite set of all agents
- St: a set of states
- $\mathcal{V}$: a valuation of propositions
- Act: a finite set of (atomic) actions
- $d : \mathbb{A}\mathrm{gt} \times St \to 2^{Act}$ defines actions available to an agent in a state
- o: a deterministic transition function that assigns outcome states $q' = o(q, \alpha_1, \ldots, \alpha_k)$ to states and tuples of actions actions are simultaneous: that's why the games are called concurrent

Definition (Semantics of CL)

Definition (Semantics of CL)

- $\mathcal{M}, q \models p$ iff $q \in \mathcal{V}(p)$;
- $\mathcal{M}, q \models \neg\varphi$ iff not $\mathcal{M}, q \models \varphi$;
- $\mathcal{M}, q \models \varphi \wedge \psi$ iff $\mathcal{M}, q \models \varphi$ and $\mathcal{M}, q \models \psi$;
- $\mathcal{M}, q \models [A]\varphi$ iff there is a collective action $\alpha_A \in d_A(q)$ such that, for every response $\alpha_{\mathbb{A}\mathrm{gt}\setminus A} \in d_{\mathbb{A}\mathrm{gt}\setminus A}$, we have that $\mathcal{M}, o(q, \alpha_A, \alpha_{\mathbb{A}\mathrm{gt}\setminus A}) \models \varphi$

Coalition Logic Extra Information

- Knowledge and epistemic information can be found in CL extra slide 7,8
- Responsibility and blameworthiness can be found in CL extra slide 10,11

Complexity

- satisfiability for CL is in PSPACE (Pauly 2001)
- satisfiability for CL + K + D and interaction axioms: also PSPACE

Alternating-time Temporal Logic (ATL/ATL*)

$\langle\langle A \rangle\rangle \Phi$: coalition A has a collective strategy to enforce temporal property Φ

Two main syntactic variants:

- "Vanilla" ATL: every temporal operator preceded by exactly one cooperation modality

  ATL extends Coalition Logic CL by the following **syntactic** translation:

  - $[A]\varphi \equiv \langle\langle A \rangle\rangle X \varphi$

- ATL*: no syntactic restrictions

Definition (Concurrent Game Structure)

Definition (Concurrent Game Structure)

A concurrent game structure is a tuple $M = \langle \mathbb{A}\mathrm{gt}, St, \mathcal{V}, Act, d, o \rangle$, where:

- $\mathbb{A}\mathrm{gt}$: a finite set of all agents
- St: a set of states
- $\mathcal{V}$: a valuation of propositions
- Act: a finite set of (atomic) actions
- $d : \mathbb{A}\mathrm{gt} \times St \to 2^{Act}$ defines actions available to an agent in a state
- o: a deterministic transition function that assigns outcome states $q' = o(q, \alpha_1, \ldots, \alpha_k)$ to states and tuples of actions actions are simultaneous: that's why the games are called concurrent

Definition (Strategy)

Definition (Strategy)

A **strategy** is a conditional plan.

memoryless strategy: a function $s_a : St \to Act$

perfect recall strategy: a function $s_a : St^+ \to Act$

A collective strategy is simply a tuple of individual strategies.

Definition (Outcome of a strategy)

Function $out(q, s_A)$ returns the set of all paths that may result from agents A executing strategy s_A from state q onward.

$\models_R$: perfect recall is assumed, strategies are of type $f : St^+ \to Act$

$\models_r$: only memoryless strategies are allowed, i.e., $f : St \to Act$

Definition (Semantics of ATL*)

Definition (Semantics of ATL*: state formulae)

$M, q \models p$	iff q is in $\mathcal{V}(p)$;
$M, q \models \neg\varphi$	iff $M, q \not\models \varphi$;
$M, q \models \varphi_1 \wedge \varphi_2$	iff $M, q \models \varphi_1$ and $M, q \models \varphi_2$;
$M, q \models \langle\langle A\rangle\rangle\Phi$	iff there is a collective strategy s_A such that, for every path $\lambda \in out(q, s_A)$, we have $M, \lambda \models \Phi$;

Definition (Semantics of ATL*: path formulae)

$M, \lambda \models \varphi$	iff $M, \lambda[0] \models \varphi$, for a state formula φ;
$M, \lambda \models X\gamma$	iff $M, \lambda[1..\infty] \models \gamma$;
$M, \lambda \models F\gamma$	iff $M, \lambda[i..\infty] \models \gamma$ for some $i \geq 0$;
$M, \lambda \models G\gamma$	iff $M, \lambda[i..\infty] \models \gamma$ for all $i \geq 0$;
$M, \lambda \models \gamma_1 \, U \, \gamma_2$	iff $M, \lambda[i..\infty] \models \gamma_2$ for some $i \geq 0$, and $M, \lambda[j..\infty] \models \gamma_1$ forall $0 \leq j < i$;

Definition (Semantics of ATL)

The semantics of “vanilla” ATL can be given entirely in terms of states:

$M, q \models p$	iff p is in $\mathcal{V}(q)$;
$M, q \models \neg\varphi$	iff $M, q \not\models \varphi$;
$M, q \models \varphi_1 \wedge \varphi_2$	iff $M, q \models \varphi_1$ and $M, q \models \varphi_2$;
$M, q \models \langle\langle A\rangle\rangle X\varphi$	iff there is a collective strategy s_A such that, for every path $\lambda \in out(q, s_A)$, we have $M, \lambda[1] \models \varphi$;
$M, q \models \langle\langle A\rangle\rangle F\varphi$	iff there is s_A such that, for every $\lambda \in out(q, s_A)$, we have $M, \lambda[i] \models \varphi$ for some $i \geq 0$;
$M, q \models \langle\langle A\rangle\rangle G\varphi$	iff there is s_A such that, for every $\lambda \in out(q, s_A)$, we have $M, \lambda[i] \models \varphi$ for all $i \geq 0$;
$M, q \models \langle\langle A\rangle\rangle \varphi_1 \, U \, \varphi_2$	iff there is s_A such that, for every $\lambda \in out(q, s_A)$, we have $M, \lambda[i] \models \varphi_2$ for some $i \geq 0$ and $M, \lambda[j] \models \varphi_1$ for all $0 \leq j < i$.

Syntactic Embeddings (Translations) can be found on ATL slides 13,14.

Memory

Let us discern between two definitions of the satisfaction relation:

$\models_R$: perfect recall is assumed, strategies are of type $f : St^+ \rightarrow Act$

$\models_r$: only memoryless strategies are allowed, i.e., $f : St \rightarrow Act$

- **Vanilla ATL’s ability is not influenced by memory**

Theorem

For any M, q and φ, we have:

$$M, q \models_r \varphi \quad \Leftrightarrow \quad M, q \models_R \varphi.$$

- **ATL* - memory matters**

Theorem

There is a model M, a state q in M, and a formula φ, such that

$$M, q \models_r \varphi \quad \not\Leftrightarrow \quad M, q \models_R \varphi$$

Fixpoint Properties

Theorem

*The following formulae are **valid** in ATL:*

- $\langle\langle A\rangle\rangle G\varphi \quad \leftrightarrow \quad \varphi \wedge \langle\langle A\rangle\rangle X \langle\langle A\rangle\rangle G\varphi$
- $\langle\langle A\rangle\rangle \varphi_1 \, U \varphi_2 \quad \leftrightarrow \quad \varphi_2 \vee \varphi_1 \wedge \langle\langle A\rangle\rangle X \langle\langle A\rangle\rangle \varphi_1 \, U \varphi_2.$

Corollary

Strategy for A that achieves an objective specified in "vanilla" ATL can be **synthesized incrementally** (no backtracking is necessary).

Model Checking ATL

function MCHECK(M, φ_0)

for $\varphi' \in Sub(\varphi_0)$ **do**

case $\varphi' = p$

$[\varphi']_M \leftarrow \mathcal{V}(p)$

case $\varphi' = \neg\psi$

$[\varphi']_M \leftarrow St \setminus [\psi]_M$

case $\varphi' = \psi_1 \wedge \psi_2$

$[\varphi']_M \leftarrow [\psi_1]_M \cap [\psi_2]_M$

case $\varphi' = \psi_1 \vee \psi_2$

$[\varphi']_M \leftarrow [\psi_1]_M \cup [\psi_2]_M$

case $\varphi' = \langle\langle A\rangle\rangle X\psi$

$[\varphi']_M \leftarrow pre(A, [\psi]_M)$

case $\varphi' = \langle\langle A\rangle\rangle G\psi$

$Q_1 \leftarrow St; \quad Q_2 \leftarrow [\psi]_M$

while $Q_1 \not\subseteq Q_2$ **do**

$Q_1 \leftarrow Q_1 \cap Q_2; \quad Q_2 \leftarrow pre(A, Q_1) \cap [\psi]_M$

$[\varphi']_M \leftarrow Q_1$

case $\varphi' = \langle\langle A\rangle\rangle \psi_1 \, U \psi_2$

$Q_1 \leftarrow \emptyset; \quad Q_2 \leftarrow [\psi_2]_M$

while $Q_2 \not\subseteq Q_1$ **do**

$Q_1 \leftarrow Q_1 \cup Q_2; \quad Q_2 \leftarrow pre(A, Q_1) \cap [\psi_1]_M$

$[\varphi']_M \leftarrow Q_1$

Complexity

Theorem (Alur, Kupferman & Henzinger 1998/2002)

Model checking ATL is P-complete, and can be done in time $O(ml)$ where $m =$ #transitions in the model and $l =$ #symbols in the formula.

ATLir

ATLir incorporates ATL and the notion of knowledge.

Definition (Uniform strategy)

Definition (Uniform strategy)

Strategy s_a is **uniform** iff it specifies the same choices for indistinguishable situations:

- (no recall:) if $q \sim_a q'$ then $s_a(q) = s_a(q')$
  If memoryless, same action for both states
- (perfect recall:) if $h \approx_a h'$ then $s_a(h) = s_a(h')$
  where $h \approx_a h'$ iff $h[i] \sim_a h'[i]$ for every i If two outcome histories (h, h') are the same, same action should be chosen for both states

It is indistinguishable, when for both histories (h,h'), each point i in h corresponds to each point i in h'.

A collective strategy is uniform iff it consists only of uniform individual strategies

Uniform if every agent is executing a uniform strategy

Strategic ability (levels)

Our cases for $\langle\langle A\rangle\rangle\gamma$ under imperfect information:

1. there is σ (not necessarily uniform) such that, for every execution of σ, γ holds
2. there is a uniform σ such that, for every execution of σ, γ holds
3. A know that there is a uniform σ such that, for every execution of σ, γ holds
4. there is a uniform σ such that A know that, for every execution of σ, γ holds

From now on, we restrict our discussion to uniform memoryless strategies (unless explicitly stated otherwise)

Definition (Semantics of ATLir)

Definition (Semantics of ATL_{ir})

$M, q \models \langle\langle A\rangle\rangle_{ir} X\varphi$	iff there is a collective uniform strategy s_A such that, for every path $\lambda \in \bigcup_{q' \sim^E_A q} out(q', s_A)$, we have $M, \lambda[1] \models \varphi$
$M, q \models \langle\langle A\rangle\rangle_{ir} G\varphi$	iff there is a collective uniform strategy s_A such that, for every path $\lambda \in \bigcup_{q' \sim^E_A q} out(q', s_A)$, we have $M, \lambda[i] \models \varphi$ for all $i \geq 0$
$M, q \models \langle\langle A\rangle\rangle_{ir} \varphi_1 U \varphi_2$	iff there is a collective uniform strategy s_A such that, for every path $\lambda \in \bigcup_{q' \sim^E_A q} out(q', s_A)$, we have $M, \lambda[i] \models \varphi_2$ for some $i \geq 0$, and $M, \lambda[j] \models \varphi_1$ forall $0 \leq j < i$;

- <<A>>F phi is an abbreviation of <<A>>true U phi.
- <<A>>!phi U phi

Fixpoint Equivalences

Interesting: ⟨⟨A⟩⟩ir are not fixpoint operators any more! Read more on ATLir intro slides 19,20.

Complexity

Theorem (Schobbens 2004; Jamroga & Dix 2006)

Model checking ATL_{ir} is Δ_2-complete in the number of transitions in the model and the length of the formula.

Model Checking ATLir

Definition (Semantics of ATL_{ir})

$M, q \models_{ir} \langle\langle A\rangle\rangle X\varphi$	iff there is a collective uniform strategy s_A such that, for every path $\lambda \in \bigcup_{q' \sim^E_A q} out(q', s_A)$, we have $M, \lambda[1] \models_{ir} \varphi$
$M, q \models_{ir} \langle\langle A\rangle\rangle G\varphi$	iff there is a collective uniform strategy s_A such that, for every path $\lambda \in \bigcup_{q' \sim^E_A q} out(q', s_A)$, we have $M, \lambda[i] \models_{ir} \varphi$ for all $i \geq 0$
$M, q \models_{ir} \langle\langle A\rangle\rangle \varphi_1 U \varphi_2$	iff there is a collective uniform strategy s_A such that, for every path $\lambda \in \bigcup_{q' \sim^E_A q} out(q', s_A)$, we have $M, \lambda[i] \models_{ir} \varphi_2$ for some $i \geq 0$, and $M, \lambda[j] \models_{ir} \varphi_1$ forall $0 \leq j < i$;

function $mcheckatlir(M, q, \langle\langle A \rangle\rangle \gamma)$**.**
Model checking simple formulae of $\mathbf{ATL}_{\mathrm{ir}}$.
Returns $\top$ if $M, q \models_{\mathrm{ir}} \varphi$ and $\perp$ otherwise.

- Guess a uniform strategy s_A;
- Remove from M all the transitions that are *not* going to be executed according to s_A;
- Model-check $\mathbf{CTL}$ formula $\mathsf{A}\gamma$ in the resulting model, and return the outcome.

Complexity in Model Checking ATLir

- $\Delta_2^P = P^{NP}$ complexity: we need an NP oracle for guessing uniform strategies,the rest of the computation is polynomial
- ATL$_{ir}$ model checking is Δ_2^P-complete, so there is no polynomial algorithm (unless $P = \Delta_2^P$ which is even more unlikely than P = NP).

Complexity

Just open the slides on complexity. We both know you need to.

- P (polynomial time): problems solvable in polynomial time by a deterministic Turing machine
- NP (polynomial nondeterministic time): problems solvable in polynomial time by a non-deterministic Turing machine
- co-NP: problems whose complement (swap yes/no answer) is in NP
- PSPACE (polynomial space): problems solvable by a Turing machine with a polynomially bounded tape (tape can be reused!)
- EXPTIME (exponential time): problems solvable in exponential time by a deterministic Turing machine

	m, l
CTL	P-complete
LTL	PSPACE-complete
CTL*	PSPACE-complete
ATL	P-complete
ATL*	PSPACE-complete

For strategies with perfect recall:

	m, l
ATL	P-complete
ATL*	2EXPTIME-complete