Alternating-Time Temporal Logics

Fabienne Eigner

Seminar
"Games in Verification and Synthesis"
June 19, 2008

Outline

Outline

Motivation
Railroad Crossing Example

Motivation

Railroad Crossing Example

- Use abstractions to model (real world) situations and applications, like the 

example.
- Use logic to formulate a specification.
- Check whether the model satisfies the specification.
- Examples for such logics: LTL, CTL, CTL*

Motivation

Railroad Crossing Example

Examples for properties the 

model might fulfill:

- "Sooner or later the train will drive past the gate."
  in CTL: $\forall\Diamond \textit{in\_gate}$
- "It is possible that the train will never enter the gate."
  in CTL: $\exists\Box \textit{out\_of\_gate}$
- **BUT:** What about open system where multiple participants interact, like the train and some gate-controller?
- Questions like "Can the controller prevent the train from entering the gate?" cannot be formulated in LTL or CTL.
- Solution: **ATL** (Alternating-Time Temporal Logic) can be used to state such properties.

Outline

Concurrent Game Structures

- Goal: Model compositions of open systems with multiple participants
- Open system: system components and environment interact
- Our computational model: **Concurrent Game Structures**

Concurrent Game Structures
Definition

A concurrent game structure $S = (k, Q, \Pi, \pi, d, \delta)$ is defined as:

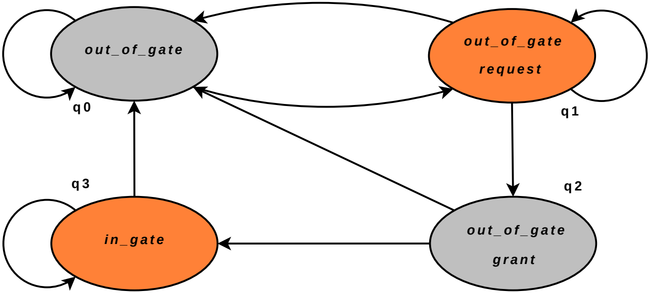
- $k \in \mathbb{N}$: number of players (named from 1 to k)
- Q: finite set of states
- Π: finite set of propositions
- $\pi : Q \rightarrow 2^{\Pi}$: labeling function
- $d : \{1, \ldots, k\} \times Q \rightarrow \mathbb{N}^{+}$: $d_a(q)$ = number of possible moves of player a at state q
- δ: transition function from one state to the next, w.r.t. the moves of each player

Concurrent Game Structures

Example

Define the 

situation as a (turn-based) game between the train and the controller:

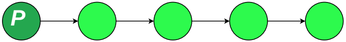


Outline

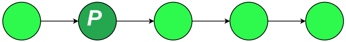
Linear-Time Temporal Logic LTL

An Intuition

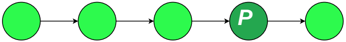
P “boolean formula”



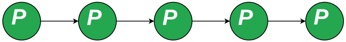
$\bigcirc P$ “next P”



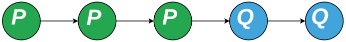
$\Diamond P$ “eventually P”



$\Box P := \neg\Diamond\neg P$ “always P”

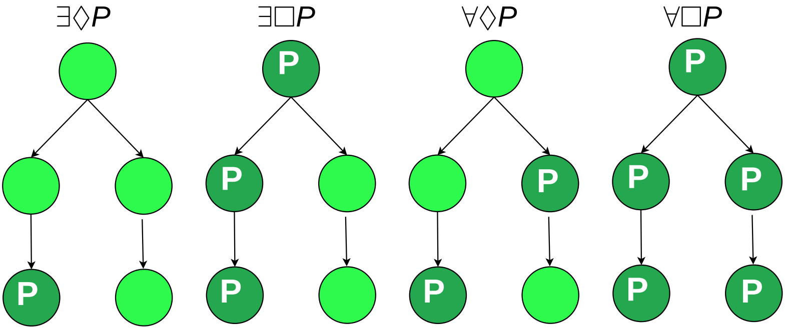


$P\,\mathcal{U}\,Q$ “P until Q”



Computational Tree Logic CTL

An Intuition



Alternating-Time Temporal Logic

Syntax

- ATL defined w.r.t. set $\Sigma = \{1, \ldots, k\}$ of players and finite set Π of propositions
- An ATL formula must be one of the following:
  - **(S1)**: p, where $p \in \Pi$
  - **(S2)**: $\neg\phi$ or $\phi \vee \psi$,
    where ϕ, ψ ATL formulas
  - **(S3)**: $\langle\langle A \rangle\rangle \bigcirc \phi$, or $\langle\langle A \rangle\rangle \Box \phi$, or $\langle\langle A \rangle\rangle \Diamond \phi$, or $\langle\langle A \rangle\rangle \phi \, \mathcal{U} \psi$,
    where $A \subseteq \Sigma$ is a set of players, and ϕ, ψ are ATL formulas

Alternating-Time Temporal Logic
Semantics

State q satifies ATL formula ϕ in structure S ($S, q \models \phi$), iff:

- $q \models p$ for propositions $p \in \Pi$, iff $p \in \pi(q)$
- $q \models \neg\phi$, iff $q \not\models \phi$
- $q \models \phi \vee \psi$, iff $q \models \phi$ or $q \models \psi$
- $q \models \langle\langle A \rangle\rangle \bigcirc \phi$, iff there exists a strategy for each player in A, s.t. for all computations λ according to these strategies from q: $\lambda[1] \models \phi$
- $q \models \langle\langle A \rangle\rangle \Box \phi$, iff for λ defined as above, and all positions $i \geq 1$: $\lambda[i] \models \phi$
- $q \models \langle\langle A \rangle\rangle \phi \, \mathcal{U} \, \psi$, iff for λ defined as above, there is a position $i \geq 1$: $\lambda[j < i] \models \phi$ and $\lambda[i] \models \psi$

ATL
A Few Remarks

- The definition of the other boolean operators and the $\Diamond$ operator follows immediately
- Dual of path quantifier $\langle\langle A\rangle\rangle$: $[[A]]$ defined as follows:

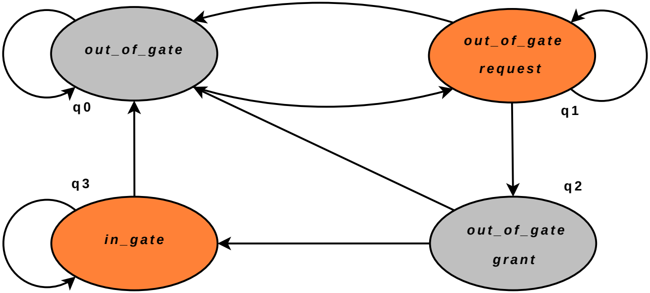
$$[[A]]\phi := \neg\langle\langle A\rangle\rangle\neg\phi$$

- Intuitive interpretation of the duals:
  - $\langle\langle A\rangle\rangle\phi$: Players in A can cooperate to make ϕ true
  - $[[A]]\phi$: Players in A cannot cooperate to make ϕ false
- There are other variants of ATL: e.g., ATL*, Fair ATL

ATL
Example Formulas

Remember the 

example:



ATL
Example Formulas

1. Whenever the train is outside the gate and has no grant to enter, then the controller can prevent it from entering:
$\langle\langle\rangle\rangle\Box((out\_of\_gate \wedge \neg grant) \rightarrow \langle\langle ctr\rangle\rangle\Box out\_of\_gate)$
2. Whenever the train is outside the gate, the controller cannot force it to enter:
$\langle\langle\rangle\rangle\Box(out\_of\_gate \rightarrow [[ctr]]\Box out\_of\_gate)$
3. Whenever the train is outside the gate, the train and the controller can cooperate so that the train will enter the gate:
$\langle\langle\rangle\rangle\Box(out\_of\_gate \rightarrow \langle\langle train, ctr\rangle\rangle\Diamond in\_gate)$

ATL
Difference ATL - CTL

- Only way of modelling the first two statements in CTL:
- $\forall\Box(\mathit{out\_of\_gate} \rightarrow \exists\Box\mathit{out\_of\_gate})$
- From this formula we cannot deduce whether the train, or the controller, or the overall system keep the train from entering the gate
- $\Rightarrow$ ATL is more expressive than CTL

Outline

The Model-Checking Problem for ATL

The model-checking problem for ATL is defined as follows:

- For a given game structure $S = (k, Q, \Pi, \pi, d, \delta)$ and an ATL formula ϕ compute the set $[\phi]_S$ of all states in S that satisfy ϕ
- We define a symbolic algorithm to solve the MP as follows (with implicit S), using the functions:
- $Reg(p),\ p \in \Pi$ returns all states where p holds
- $q \in Pre(A, \rho)$, iff in state q, the players in A can enforce the next state to be in ρ, by cooperating

The Model-Checking Problem for ATL

Algorithm

$MC(\phi)$

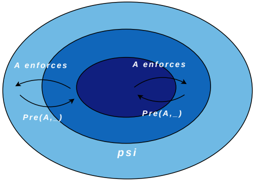
- **foreach** subformula ϕ' of ϕ (in ascending order) **do**
  - **case** $\phi' = p$: **return** $Reg(\phi')$
  - **case** $\phi' = \neg\psi$: **return** $Q \setminus MC(\psi)$
  - **case** $\phi' = \psi_1 \vee \psi_2$: **return** $MC(\psi_1) \cup MC(\psi_2)$
  - **case** $\phi' = \langle\langle A\rangle\rangle \bigcirc \psi$: **return** $Pre(A, MC(\psi))$

The Model-Checking Problem for ATL

Algorithm Cont'd

- **case** $\phi' = \langle\langle A\rangle\rangle\Box\psi$:
  - $\rho := Q; \tau := MC(\psi)$
  - **while** $\rho \not\subseteq \tau$ **do** $\rho := \tau; \tau := Pre(A, \rho) \cap MC(\psi)$;
  - **return** ρ

Intuitively: **do**

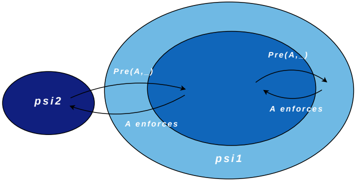


until the greatest fixpoint is reached

The Model-Checking Problem for ATL
Algorithm Cont'd

- **case** $\phi' = \langle\langle A\rangle\rangle\psi_1\ \mathcal{U}\ \psi_2$:
  - $\rho := \emptyset; \tau := MC(\psi_2)$
  - **while** $\tau \not\subseteq \rho$ **do** $\rho := \rho \cup \tau; \tau := Pre(A, \rho) \cap MC(\psi_1)$;
  - **return** ρ

Intuitively: **do**



until the smallest fixpoint is reached

The Model-Checking Problem for ATL

Complexity

The complexity of the algorithm $MC(\phi)$ lies in $\mathcal{O}(m \cdot l)$ where

- l denotes the length of the formula ϕ
  One can see that the outer loop of the algorithm is evoked $\mathcal{O}(l)$ times
- m describes the number of transitions in the game structure S
  Each case statement can be executed in $\mathcal{O}(m)$ time

Conclusion

- In order to model interactions between multiple participants we can use concurrent games
- ATL: powerful logic to state properties of concurrent games
- ATL more expressive than CTL (captures the capabilities of individual user)
- We have seen a symbolic algorithm for ATL-model-checking that runs in $\mathcal{O}(m \cdot l)$ time

Literature I

Rajeev Alur, Thomas Henzinger, and Orna Kupferman.
Alternating-Time Temporal Logic.
http://www.eecs.berkeley.edu/~tah/Publications/alternating-time_temporal_logic.html

Thank you!