Bernd Finkbeiner Date: October 23, 2012

Automata, Games, and Verification: Lecture 1

---

0 Course Organization

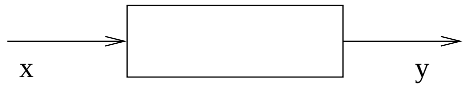
- Vertiefungsvorlesung (6 CP)
- www.react.uni-saarland.de/teaching/automata-games-verification-12/
- Bernd Finkbeiner: E1.3/506, office hours Wednesdays 3-4
- Felix Klein, Markus Rabe, Hazem Torfah
- Tutorial: Thursdays 2-4pm, Room 010, building E1.7.
  Exception: first tutorial on Wednesday Oct 31, 12-2pm, Room U12 building E1.1
- Exams: End-of-term exam: February 14th, 2013, HS 001 & 002, E1.3 1pm-3:30pm
  End-of-semester exam: April 4th, 2013, HS 001 & 002, E1.3
- Your final grade will depend 100% on the final exam.
- Every week, assignments will be given out and solutions presented the following week. Credit points will be given to students who present correct solutions to problems during the tutorial.
- Each problem will be assigned in advance to a group of two members (single person groups are allowed). If your group is assigned a problem in a particular week, please stop by for a 15 minute meeting to discuss your solution (time slots are indicated on the problem sheet) and present your solution at the tutorial.
- The solutions will not be graded. Problems will be distributed fairly (on a rotating scheme), and participation (max 1 unexcused no-show) is required to write the final exam.
- Challenge problems: not assigned to any group; take you out of the rotation once
- Literature:
  Erich Grädel et al: Automata, Logics, and Infinite Games (available online)
  Khoussainov/Nerode: Automata Theory and its Applications
  Lecture notes (online after lecture)
  Summary slides (online after lecture)

1 Motivation and Overview

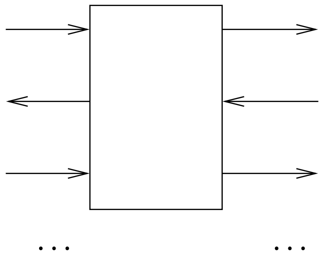
1.1 Reactive Systems

We distinguish

- Transformational programs



- Reactive systems



  - nonterminating behavior
  - interaction (program vs. environment)

Examples of reactive systems include controllers in embedded systems, operating systems, communication protocols, and online applications. The semantics of such systems is usually given as a graph called *transition system*, where the nodes are states and the edges transitions. The nodes are labeled with sets of *atomic propositions*, i.e., basic facts that are true in a particular state.

Definition 1 *A* transition system $(AP, S, s_0, \rightarrow, L)$ *consists of*

- *AP: atomic propositions*
- *S: finite set of states*

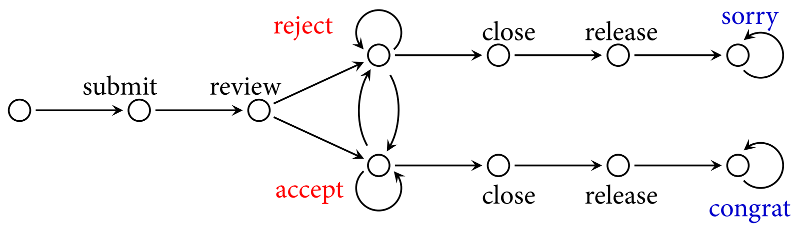


Figure 1: Transition system for "BusyChair".

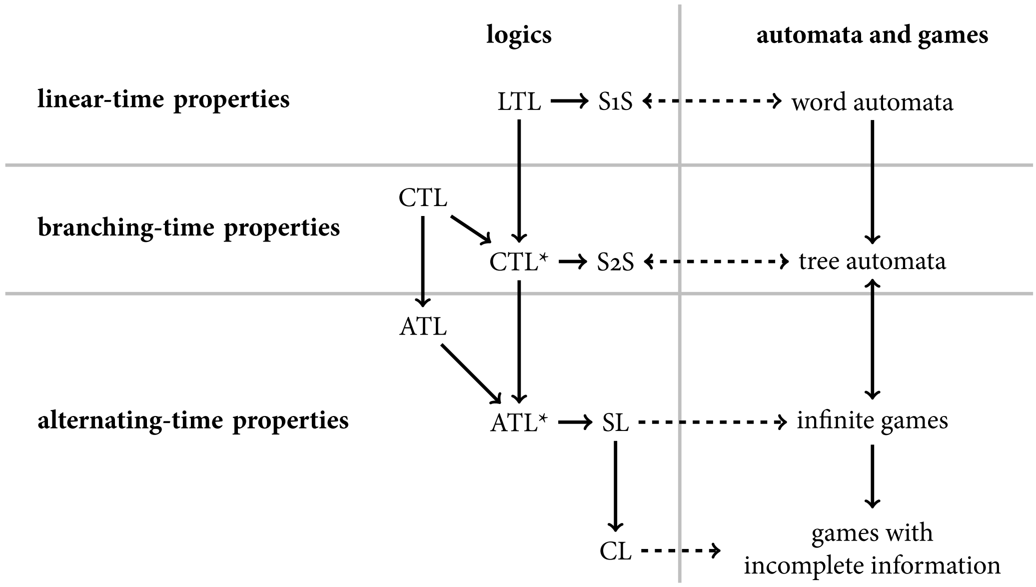


Figure 2: Logics, automata, and games for reactive systems.

- $s_0 \in S$: *initial state*
- $\rightarrow \subseteq S \times S$: *transition relation*
- $L : S \rightarrow 2^{AP}$: *labeling function*

Example: Figure 1 shows the transition system of "BusyChair," a simple online application for the management of submissions to academic conferences. After an author submits a paper, the programme committee discusses whether to accept or reject the submission and finally closes the discussion and sends a message (congrats/sorry) to the authors. ✚

1.2 Properties and Logic

We specify the correctness of a reactive system using collections of *properties* that can be checked individually. We will in particular focus on properties expressed in temporal logic. Temporal logics and properties can be classified according to the *linear/branching-time spectrum*.

Example: Consider again the "BusyChair" application.

- "there is never both a sorry and congrat on the *same* computation path" is a *linear-time* property
- "there is both a path with sorry and a path with congrat" is a *branching-time* property
- "authors cannot *enforce* congrat" is an *alternating-time* (or: *game*) property

✚

We will consider various temporal logics along the *linear/branching-time spectrum*.

- Linear-time temporal logic (LTL) describes sets of infinite sequences. A system is correct if all the label sequences of the Kripke strucure are contained in this set.
- Computation-tree logic (CTL/CTL*) describes sets of infinite trees. A system is correct if the unrolling of the Kripke structure into a tree is an element of this set.
- Alternating-time temporal logic (ATL/ATL*) describes objectives for coalitions of agents. A system is correct if the coalition has a strategy to accomplish the objective.
- Strategy logic (SL) relates multiple, existentially and universally quantified strategies. A system is correct if the specified relation ship is true on the game arena defined by the system.
- Coordination logic (CL) extends CL with strategies under incomplete information, such as the information visible at the interface of a component.

In addition to the temporal logics, we will study a few other logics, in particular the *monadic second-order logics*.

- Monadic second-order logic with one successor (S1S) is the logical representation of infinite words. Its expressiveness exceeds that of LTL.
- Monadic second-order logic with two successors (S2S) is the logical representation of (binary) trees. Its expressiveness exceeds that of CTL*.

Figure 2 gives an overview over the logics and their relative expressiveness.

1.3 Automata and Games

We will see that all these logics correspond to various types of automata and games.

- Automata over infinite words (ω-automata) recognize subsets of Σ^ω, the set of infinite sequences over a given alphabet Σ.
- Automata over infinite (binary) trees recognize subsets of $\{0,1\}^* \to \Sigma$, the set of infinite binary trees labeled with letters from Σ.
- Infinite games over finite graphs are two-player games where the plays are infinite paths through a game arena given as a finite graph. Strategies in such games are mappings from sequences of states (histories) to decisions.
- Games over incomplete information limit the informedness of the players. Strategies in such games are mappings from sequences of observations to decisions.

Figure 2 relates the automata and games to the logics.

1.4 Linear-time properties

A *linear-time property* is a subset of $(2^{AP})^\omega$.

- The *set of natural numbers* $\{0, 1, 2, 3, \ldots\}$ is denoted by ω.
- An *alphabet* Σ is a finite set of symbols.
- An *infinite word (or sequence, string)* is a function from natural numbers to an alphabet: $\alpha : \omega \to \Sigma$

  An infinite word is composed of its letters, so that in particular $\alpha = \alpha(0)\alpha(1)\alpha(2)\ldots$
- Σ^ω: set of *infinite words* over alphabet Σ.
- An *ω-language* L is a subset of Σ^ω.
- Σ^*: set of *finite words* over alphabet Σ.
- Σ^+: set of *finite non-empty words* over alphabet Σ.
- AP: set of *atomic propositions*
- 2^{AP}: subsets of AP. For $p \in AP, s \in 2^{AP}$ we write $s \models p$ iff $p \in s$.

Linear-time temporal logic

Linear-time temporal logic (LTL) is a modal logic over infinite sequences [Pnueli 1977].

Syntax

- *Propositional logic*
  - $a \in AP$ — atomic proposition
  - $\neg\varphi$ and $\varphi \land \psi$ — negation and conjunction
- *Temporal operators*
  - $\mathrm{X}\varphi$ — next state fulfills φ
  - $\varphi \,\mathrm{U}\, \psi$ — φ holds Until a ψ-state is reached
- *Derived operators*
  - $\mathrm{F}\,\varphi \equiv \mathit{true}\,\mathrm{U}\,\varphi$ — "some time in the future"
  - $\mathrm{G}\,\varphi \equiv \neg\,\mathrm{F}\,\neg\,\varphi$ — "from now on forever"

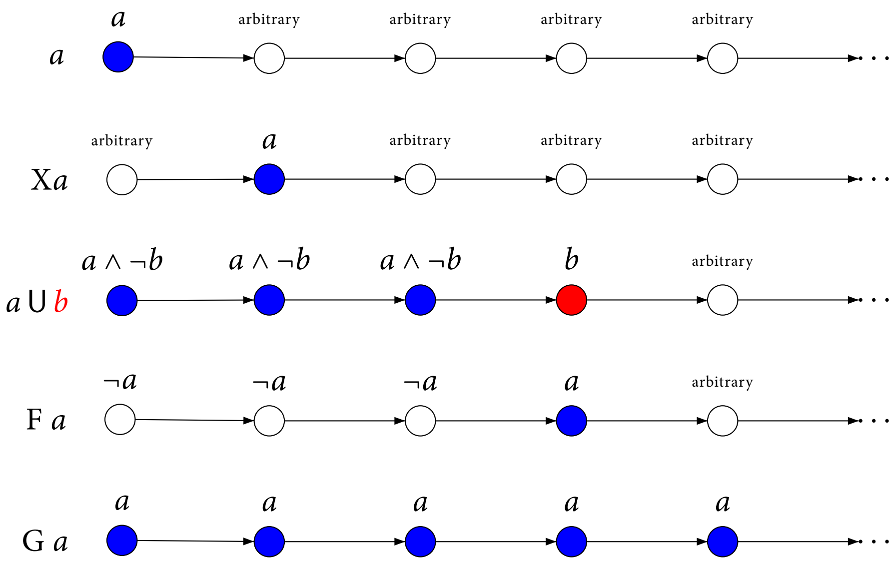


Figure 3: Semantics of LTL

Semantics

An LTL formula φ over AP defines the *linear-time property*

$$\mathcal{L}(\varphi) = \left\{\sigma \in \left(2^{AP}\right)^{\omega} \mid \sigma \vDash \varphi\right\},$$

where $\vDash$ is the smallest relation satisfying:

$$\begin{array}{lllll}
\sigma & \vDash & a & \text{iff} & a \in \sigma(0) \quad (\text{i.e., } \sigma(0) \vDash a) \\
\sigma & \vDash & \varphi_1 \wedge \varphi_2 & \text{iff} & \sigma \vDash \varphi_1 \text{ and } \sigma \vDash \varphi_2 \\
\sigma & \vDash & \neg\varphi & \text{iff} & \sigma \nvDash \varphi \\
\sigma & \vDash & \mathrm{X}\varphi & \text{iff} & \sigma[1..] = \sigma(1)\sigma(2)\sigma(3)\ldots \vDash \varphi \\
\sigma & \vDash & \varphi_1 \,\mathrm{U}\, \varphi_2 & \text{iff} & \exists j \geq 0.\ \sigma[j..] \vDash \varphi_2 \ \text{ and } \ \sigma[i..] \vDash \varphi_1 \text{ for all } 0 \leq i < j
\end{array}$$

The semantics is illustrated in Figure 3.

Example:

- $\mathrm{G}(\neg\, \text{reject})$
- $\neg(\mathrm{F}\ \text{sorry}\ \wedge\ \mathrm{F}\ \text{congrat})$
- $\mathrm{G}(\text{close}\ \rightarrow \mathrm{X}\ \text{release})$

The satisfaction of an LTL formula over a transition system is defined as follows:

- A *path* from a state $s \in S$ is an infinite sequence $s_0 s_1 s_2 \ldots \in S^{\omega}$ such that $s_0 = s$ and $s_i \rightarrow s_{i+1}$ for all $i \geq 0$.

- The *trace* of a path $s_0 s_1 s_2 \ldots$ is an infinite sequence $a_o a_1 a_2 \in (2^{AP})^\omega$ such that $a_i = L(s_i)$ for all $i \geq 0$.
- $\mathcal{T} \vDash \varphi$: transition system $\mathcal{T}$ *satisfies* LTL formula φ iff $\mathit{Traces}(\mathcal{T}, s_0) \subseteq \mathcal{L}(\varphi)$,
  where $\mathit{Traces}(\mathcal{T}, s)$: set of traces of paths from s in transition system $\mathcal{T}$.

1.5 Branching-time properties

Computation tree logic (CTL)

CTL is a modal logic over infinite trees [Clarke & Emerson 1981].

Syntax

- *CTL state formulas.*
  - $a \in AP$ — atomic proposition
  - $\neg\,\Phi$ and $\Phi\ \wedge\ \Psi$ — negation and conjunction
  - $\mathsf{E}\,\varphi$ — there *exists* a path fulfilling φ
  - $\mathsf{A}\,\varphi$ — *all* paths fulfill φ
- *CTL path formulas.*
  - $\mathsf{X}\,\Phi$ — the next state fulfills Φ
  - $\Phi\,\mathsf{U}\,\Psi$ — Φ holds until a Ψ-state is reached

Note that X and U alternate with A and E

Semantics

- *CTL state formulas.* Semantics defined by a relation $\vDash$ such that $s \vDash \Phi$ if and only if formula Φ holds in state s.

$$
\begin{array}{ll}
s \vDash a & \text{iff } \ a \in L(s) \\
s \vDash \neg\,\Phi & \text{iff } \ \neg\,(s \vDash \Phi) \\
s \vDash \Phi\ \wedge\ \Psi & \text{iff } \ (s \vDash \Phi)\ \wedge\ (s \vDash \Psi) \\
s \vDash \mathsf{E}\,\varphi & \text{iff } \ \pi \vDash \varphi \text{ for } \textit{some} \text{ path } \pi \text{ that starts in } s \\
s \vDash \mathsf{A}\,\varphi & \text{iff } \ \pi \vDash \varphi \text{ for } \textit{all} \text{ paths } \pi \text{ that start in } s
\end{array}
$$

- *CTL path formulas.* Semantics defined by a relation $\vDash$ such that $\pi \vDash \varphi$ if and only if path π satisfies φ.

$$
\begin{array}{ll}
\pi \vDash \mathrm{X}\Phi & \text{iff } \pi[1] \vDash \Phi \\
\pi \vDash \Phi\,\mathsf{U}\,\Psi & \text{iff } (\exists\, j \geq 0.\ \pi[j] \vDash \Psi\ \wedge\ (\forall\, 0 \leq k < j.\ \pi[k] \vDash \Phi))
\end{array}
$$

where $\pi[i]$ denotes the state s_i in the path π

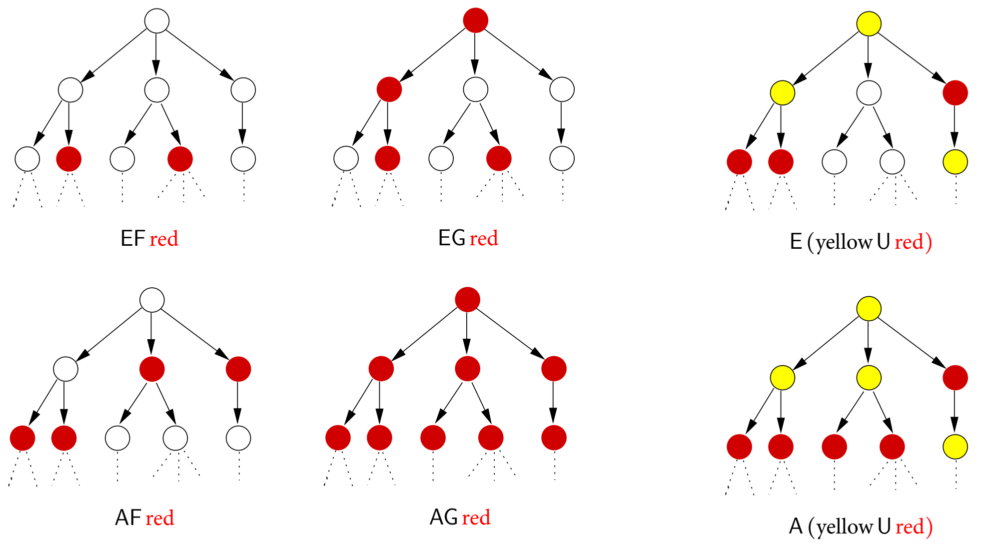


Figure 4: Examples of CTL properties

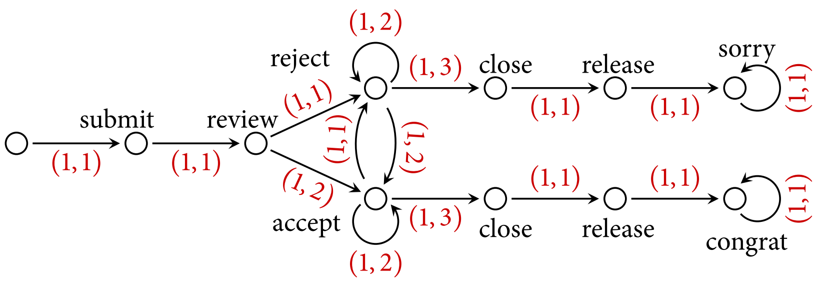
The satisfaction of a CTL formula over a transition system is defined as follows: $\mathcal{T} \models \Phi$ iff $s_0 \models \Phi$.

Example:

- EF sorry $\wedge$ EF congrat
- Figure 4 shows a few more examples of CTL formulas and their semantics.

1.6 Game properties

Concurrent game structures



Definition 2 *A* **concurrent game structure** $(k, AP, S, s_0, d, \delta, L)$ *consists of*

- $k \in \mathbb{N}$: *number of players*
- AP: *atomic propositions*
- S: *finite set of states,* $s_0 \in S$: *initial state*
- $d : \{1, \dots, k\} \times S \to \mathbb{N}$: *number of moves available to player*

- $\delta : S \times \{1, \ldots, d(1)\} \times \ldots \times \{1, \ldots, d(k)\} \to S$: *transition function*
- $L : S \to 2^{AP}$: *labeling function*

Alternating temporal logic (ATL)

Syntax

- *ATL state formulas*:
  - $a \in AP$ — atomic proposition
  - $\neg\,\Phi$ and $\Phi\ \wedge\ \Psi$ — negation and conjunction
  - $\langle\langle A \rangle\rangle \varphi$ — agents in A have strategy to enforce φ
- *ATL path formulas* as for CTL.

$A \subseteq \{1, \ldots, k\}$ is a set of players.

Semantics.

- A *strategy* for player a is a function $f_a : S^+ \to \mathbb{N}$
  such that $f_a(\sigma \cdot q) \leq d_a(q)$.
- Given a set $F_A = \{f_a \mid a \in A\}$ of strategies for a set of players A,
  the *outcomes* $Outcomes(F_A, s)$ of F_A from state s are the paths $s_0 s_1 s_2 \ldots$ such that $s_0 = s$ and for all $i \geq 0$ there is a vector $(j_1, \ldots, j_k) \in \mathbb{N}^k$ such that
  - $j_a = f_a(s_0 \ldots s_i)$ for all players $a \in A$, and
  - $\delta(s_i, j_1, \ldots, j_k) = s_{i+1}$
- $s \models \langle\langle A \rangle\rangle \varphi$ iff there exists a set of strategies F_A for the players in A,
  such that $\pi \models \varphi$ for all $\pi \in Outcomes(F_A, s)$.

Example:

- $\neg\langle\langle \{1\} \rangle\rangle \mathrm{F}$ congrat
- $\langle\langle \{2\} \rangle\rangle \mathrm{F}$ congrat $\wedge$ $\langle\langle \{2\} \rangle\rangle \mathrm{F}$ sorry

Comment: ATL subsumes CTL: $\mathsf{A}\,\Phi \equiv \langle\langle \varnothing \rangle\rangle \Phi$, $\mathsf{E}\,\Phi \equiv \langle\langle \{1, \ldots, k\} \rangle\rangle \Phi$

1.7 Verification and Synthesis

- *Verification.* “Does a given system satisfy a given property?”
- *Realizabilty/Synthesis.* “Does there exist a system that satisfies a given property?”