ALTERNATING-TIME TEMPORAL EPISTEMIC LOGIC

Mike Wooldridge & Wiebe van der Hoek

University of Liverpool

Overview

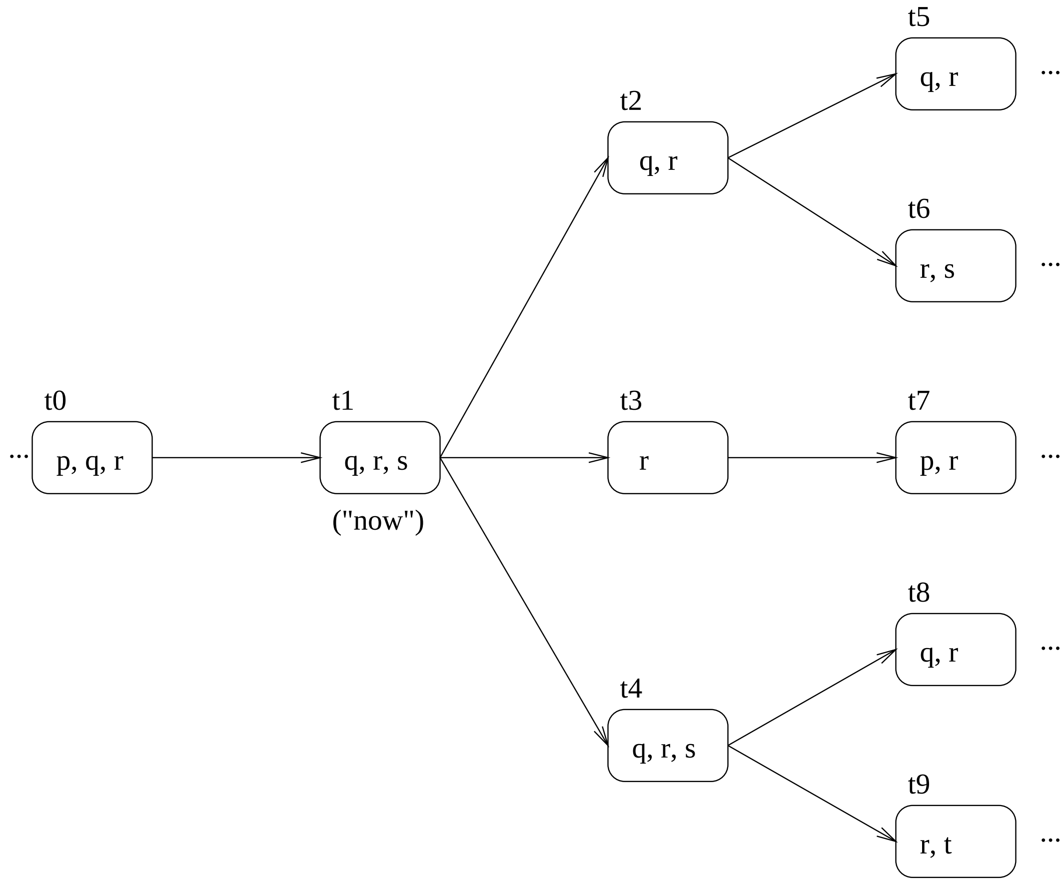
Aims of this presentation:

- review the motivation & history of branching temporal logic in CS;
- introduce the variant of branching temporal logic known as ATL;
- show how ATL can naturally be extended by *knowledge* modalities in ATEL;
- survey steps towards model checking for ATEL;
- illustrate these ideas with a case study.

Branching Temporal Logic

- Natural to view the possible computations of a system as a *tree* linear in the past, branching into the future.
- Branching corresponds to different ways in which non-determinism can be resolved.

A Branching time model



Computation Tree Logic: CTL

- The most successful branching temporal logic is *CTL*.
- Extends propositional logic with
  - *path quantifiers* A, E
  - *tense modalities* $\bigcirc$, $\Diamond$, $\Box$, $\mathcal{U}$
- Possible combinations of these are restricted as follows:

A$\bigcirc\varphi$	"on all paths, φ is true next
A$\Diamond\varphi$	"on all paths, φ is eventually true
A$\Box\varphi$	"on all paths, φ is always true
A$\varphi\mathcal{U}\psi$	"on all paths, φ is true until ψ
E$\bigcirc\varphi$	"on some path, φ is true next
E$\Diamond\varphi$	"on some path, φ is eventually true
E$\Box\varphi$	"on some path, φ is always true
E$\varphi\mathcal{U}\psi$	"on some path, φ is true until ψ

Models for CTL

- Models for CTL are *Kripke structures*:

$$\langle S, R, \pi \rangle$$

  where

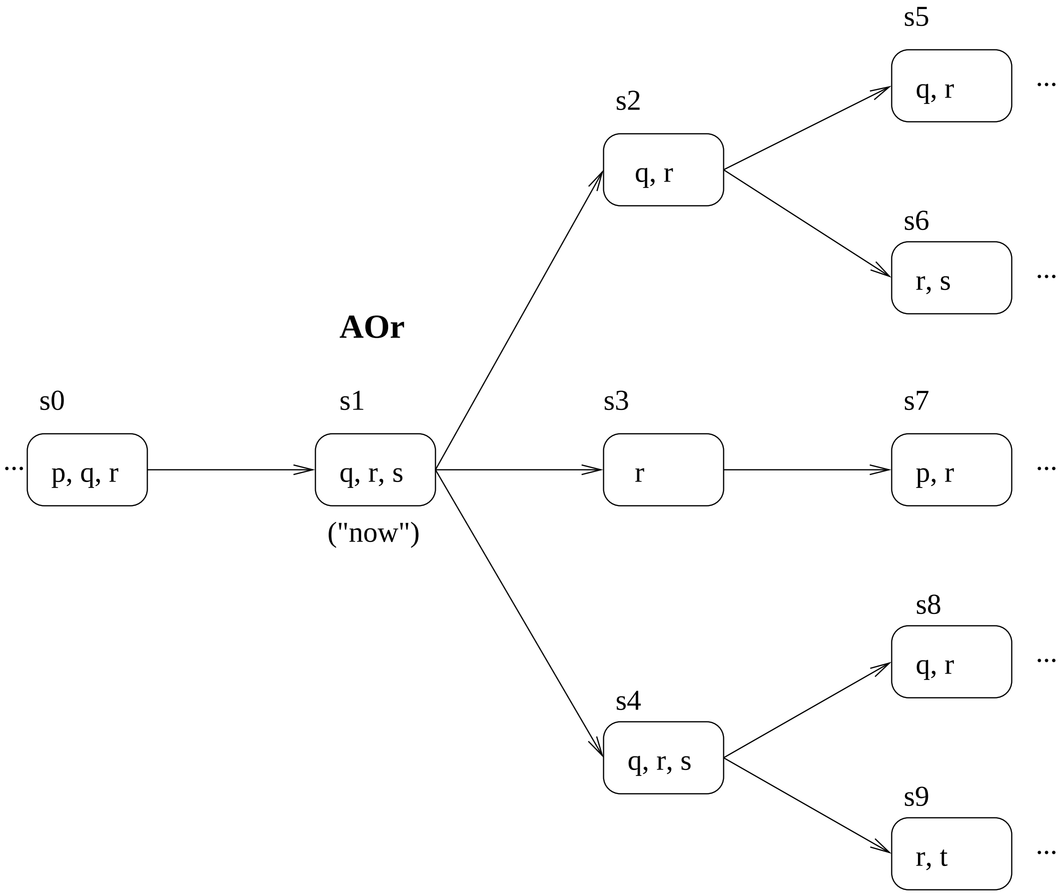
  – S is the set of possible system states

  – $R \subseteq S \times S$ is a total binary *next state* relation on S

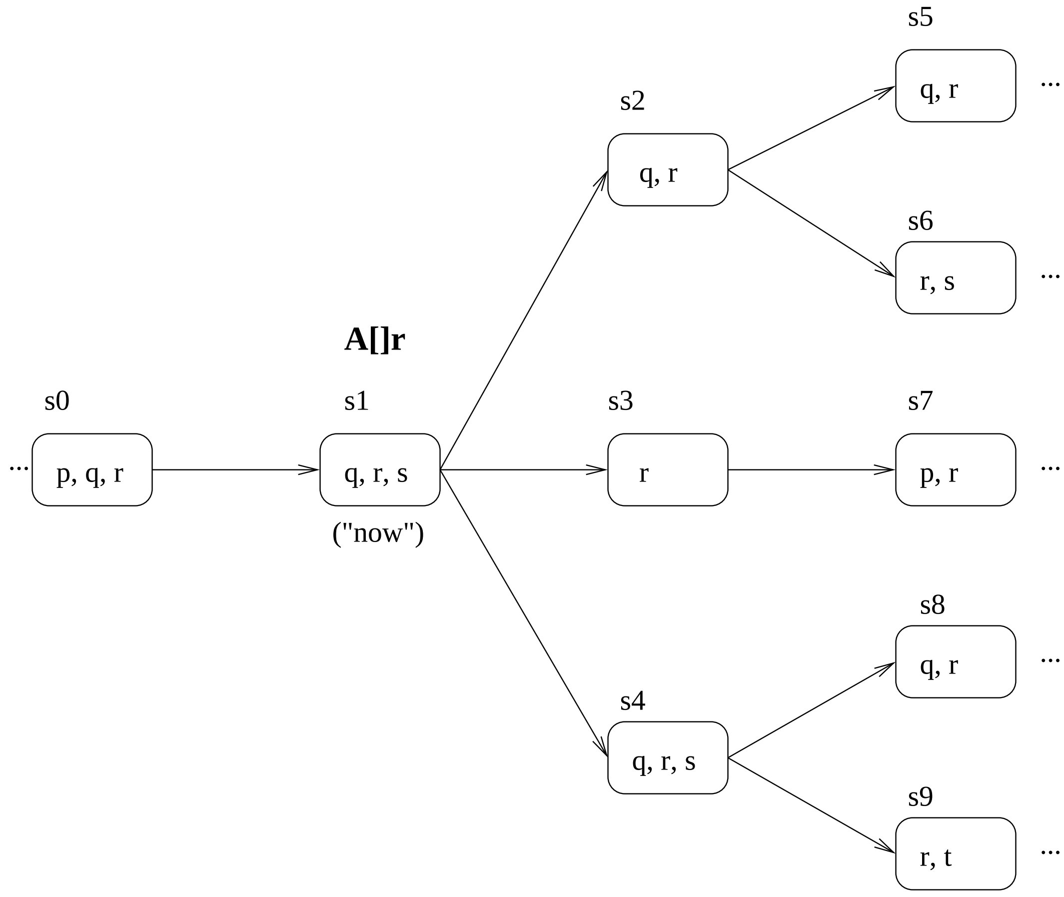
  – $\pi : S \to 2^{\Pi}$ says which propositions are true in each state.

- The branches are obtained by *unwinding* this relation, giving *paths* through the structure.

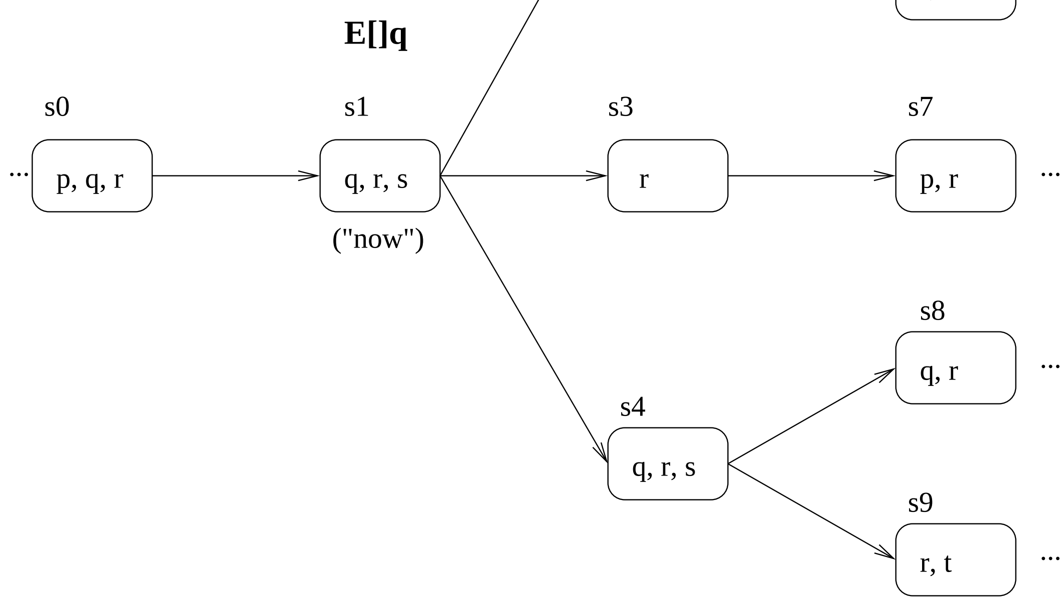
Example 1



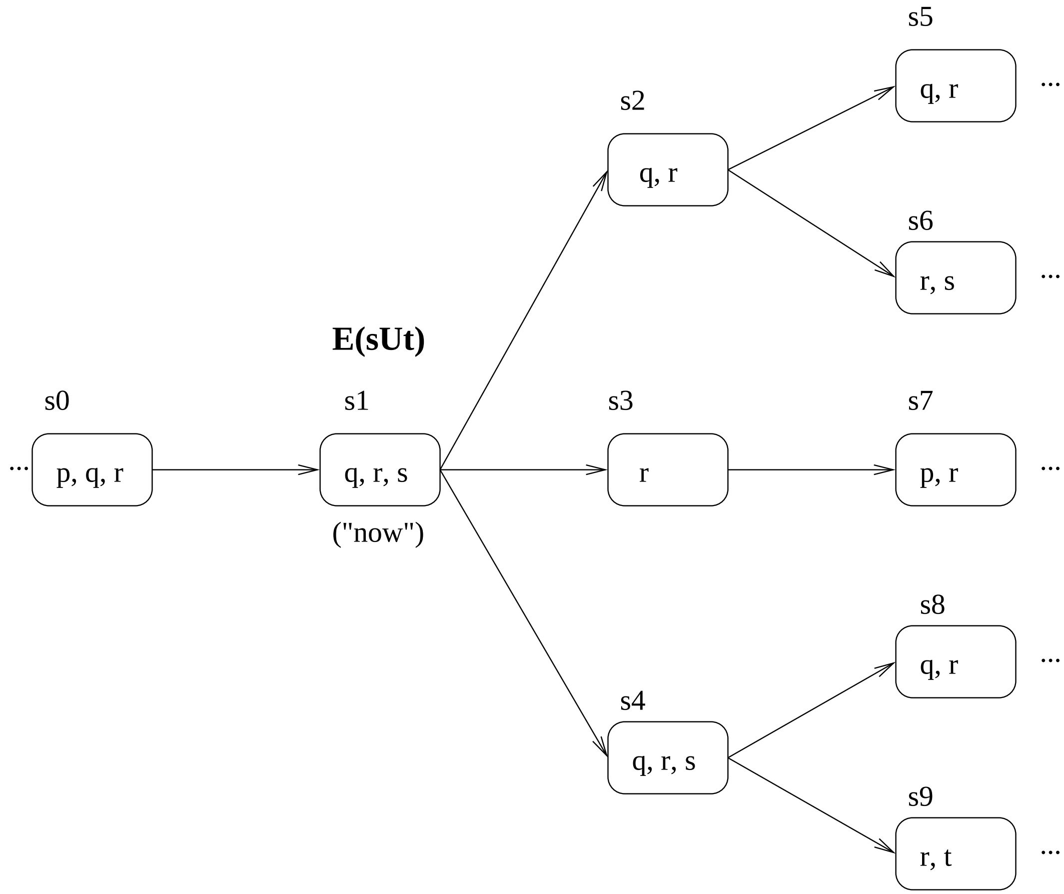
Example 2



Example 3



Example 4



Computational Properties of CTL

- Proof problem for CTL:

  Given CTL formulae φ, ψ can we prove ψ from φ?

  Time complexity: EXPTIME-complete.

  (So directly proving properties of systems using CTL looks to be v hard.)

- Model checking problem for CTL:

  Given model $M = \langle S, R, \pi \rangle$, state $s_0 \in S$, and formula φ, is φ is true at state s_0 in M?

  Time complexity: $O(|M|.|\varphi|)$.

  (So model checking properties of systems using CTL is (comparatively) easy... many practical model checkers for CTL available: SMV the best known.

Alternating-time Temporal Logic

- In 1997, Alur, Henzinger & Kupferman proposed a natural variation of CTL known as *Alternating-time Temporal Logic* (ATL).
- Branching used to model evolution of a system controlled by a set of *agents*, which can affect the future by making *choices*.
- The particular future that will emerge depends on *combination* of choices that agents make.

Cooperation Modalities

- Path quantifiers A, E are replaced by *cooperation modalities*:

$$\langle\langle G \rangle\rangle \varphi$$

  means

  "group G can cooperate to ensure that φ"

  or

  "G have a collective strategy to force φ"

- Note that:

$$\begin{array}{ll} \langle\langle \emptyset \rangle\rangle & \text{is same as A} \\ \langle\langle \Sigma \rangle\rangle & \text{is same as E} \end{array}$$

Example ATL Formulae

- $\langle\langle mjw \rangle\rangle \Diamond \mathit{bored\text{-}audience}$

  mjw has a strategy for ensuring that the audience is eventually bored

- $\neg\langle\langle mjw \rangle\rangle \Box \mathit{excited}$

  mjw has no strategy for ensuring that the audience is always excited

- $\langle\langle gwb, tb \rangle\rangle \Diamond \mathit{peace}$

  gwb and tb have a strategy for ensuring that, eventually, there is peace (!)

Semantics: Alternating Transition Systems

Semantics of ATL given in terms of ATSs:

$$\langle \Pi, \Sigma, Q, \pi, \delta \rangle,$$

where:

- Π is a finite, non-empty set of *atomic propositions*;
- $\Sigma = \{a_1, \ldots, a_n\}$ is a finite, non-empty set of *agents*;
- Q is a finite, non-empty set of *states*;
- $\pi : Q \to 2^{\Pi}$ gives the set of primitive propositions satisfied in each state;
- $\delta : Q \times \Sigma \to 2^{2^Q}$ is the *system transition function*: $\delta(q, a)$ is the set of choices available to agent a when the system is in state q.

ATL Complexity Properties

- Proof problem for ATL:

  Time complexity: EXPTIME-hard (probably much worse).
- ATL model checking:

  Time complexity: PTIME-complete.

  ATL model checking implemented in MOCHA system

Alternating-time Temporal Epistemic Logic

- ATL is a powerful language for expressing properties of multiagent systems.
- ATEL extends it by *knowledge modalities*, of the kind pioneered by Halpern et al:

$K_i\varphi$	means	agent i knows φ
$C_\Gamma\varphi$		φ is common knowledge in Γ
$E_\Gamma\varphi$		everyone in Γ knows φ
$D_\Gamma\varphi$		there is distributed knowledge of φ in Γ

Applications of ATEL: Bit transmission

- Consider a system containing a sender S, a receiver R, and an environment env through which messages are sent.
- Under certain fairness conditions we can express the fact that the environment cannot prevent the sender from sending a message until it is received.

$$[\![env]\!] send_m \, \mathcal{U} \, K_R m \tag{1}$$

Applications of ATEL: Cooperative Problem Solving

- Group Γ can guarantee that their implicit knowledge eventually becomes explicly known by everyone:

$$D_\Gamma \varphi \rightarrow \langle\langle \Gamma \rangle\rangle \Diamond E_\Gamma \varphi \tag{2}$$

Applications of ATEL: Secure Communication

- Agent i can send private information to j, without revealing the message to another agent h:

$$K_a\varphi \wedge \neg K_j\varphi \wedge \neg K_h\varphi \wedge \langle\langle i, j\rangle\rangle(K_a\varphi \wedge K_j\varphi \wedge \neg K_h\varphi) \tag{3}$$

Applications of ATEL: Rights to Secure Communication

- Suppose we have three agents, of which agent 1 knows whether p, i.e., $K_1 p \vee K_1 \neg p$, and this is common knowledge.
- 1 can tell the truth only to 2, or to 2 and 3 separately or he can announce p in public:

$$\langle\langle 1 \rangle\rangle \bigcirc (K_2 p \wedge \neg K_3 p) \wedge \langle\langle 1 \rangle\rangle \bigcirc (K_2 p \wedge K_3 p \wedge \neg C_{\{2,3\}} p) \wedge \langle\langle 1 \rangle\rangle \bigcirc (C_{\{2,3\}} p)$$

Applications of ATEL: Knowledge Games

- *Epistemic updates* are interpreted in a simple card game, where the aim of the player is to find out a particular deal d of cards.

$$d \rightarrow \langle\langle i \rangle\rangle \Diamond (K_i d \wedge \bigwedge_{i \neq j} \neg K_j d) \tag{4}$$

Applications of ATEL: Knowledge Preconditions

- If Bob knows that the combination of the safe is s, then he is able to open it (o), as long as the combination remains unchanged.

$$K_b(c = s) \rightarrow \langle\langle b \rangle\rangle (\langle\langle b \rangle\rangle \bigcirc o) \,\mathcal{U}\, (c \neq s) \tag{5}$$

Alternating Epistemic Transition Systems

Semantics in terms of *alternating epistemic transition system* (AETS) is a tuple

$$\langle \Pi, \Sigma, Q, \sim_1, \ldots, \sim_n, \pi, \delta \rangle,$$

where:

- Π is a set of *atomic propositions*;
- $\Sigma = \{a_1, \ldots, a_n\}$ is a set of *agents*;
- Q is a set of *states*;
- $\sim_a \subseteq Q \times Q$ is an *epistemic accessibility relation* for each agent $a \in \Sigma$
- $\pi : Q \to 2^{\Pi}$ is an interpretation
- $\delta : Q \times \Sigma \to 2^{2^Q}$ is the system transition function.

ATEL Complexity Properties

- Proof problem for ATEL:
  Time complexity: EXPTIME-hard (probably much worse).
- ATEL model checking:
  Time complexity: as ATL (PTIME complete)
  No model checker implemented yet.

An Interpeted Systems Model of Knowledge

- We reduce ATEL model checking to ATL model checking...
  but to do this, we need to obtain the $\sim_a$ relations!
- Given state $q \in Q$ and agent $a \in \Sigma$, write $state_a(q)$ to denote *local* state of agent a when the system is in state q.
- Then obtain the knowledge accessibility relation as follows:

$$q \sim_a q' \quad \text{iff } state_a(q) = state_a(q'). \tag{6}$$

Model Checking Epistemic Properties with MOCHA

- Suppose we want to check whether, when the system is in some state q, agent a knows φ.

  This amounts to showing that

$$\forall q' \in Q \text{ s.t. } state_a(q) = state_a(q') \text{ we have } S, q' \models \varphi. \tag{7}$$

- We can represent such properties directly as formulae of ATL, which can be automatically checked using MOCHA...

- We want to check whether

$$S, q \models K_a \varphi$$

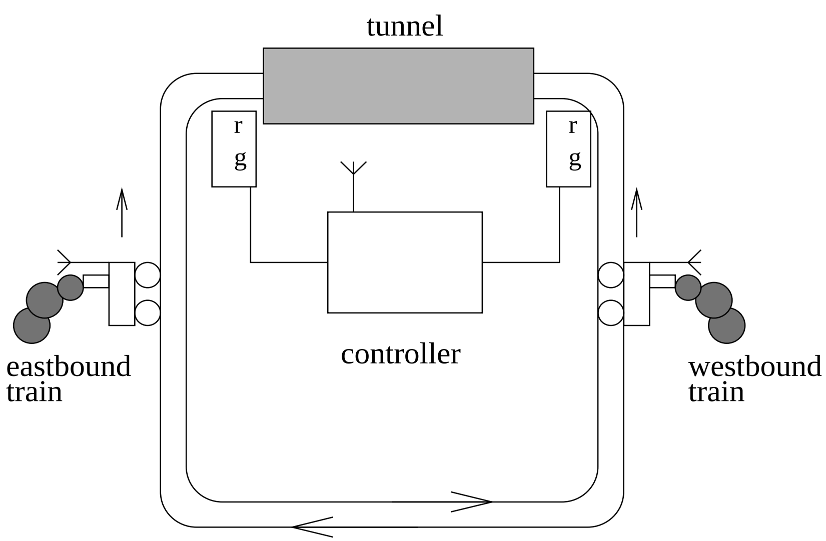
- Let $state_a(q) = s$.
- Then we merely need to check:

$$\langle\langle\rangle\rangle \Box ((state_a = s) \rightarrow \varphi) \tag{8}$$

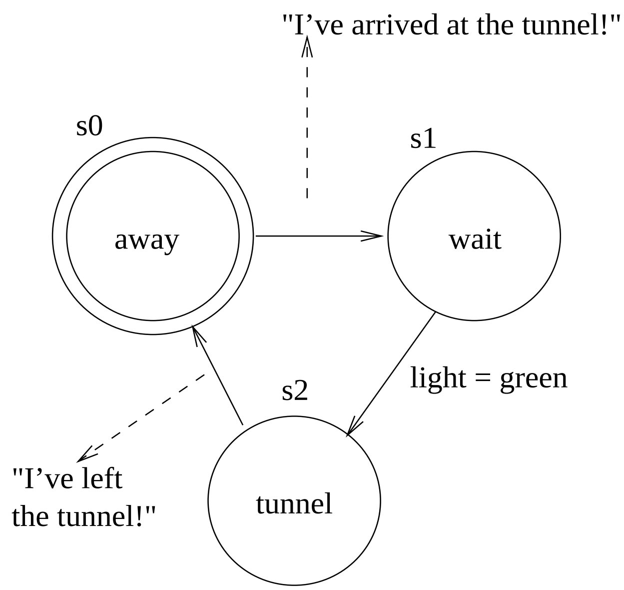
- In MOCHA notation:

```
<< >>G((stateA = s) -> phi)
```

A Case Study: The Train Controller



(a) Overall structure of the train controller sytem



(b) Train states, transitions, and signals

A Simple Example

- "When one train is in the tunnel, it knows the other train is not in the tunnel":

$$(state_a = tunnel) \rightarrow K_a(state_b \neq tunnel) \quad (a \neq b \in \{E, W\})$$

- Translating into MOCHA, this schema gives the following:

```
<<>> G ((stateE=tunnel) => (~(stateW=tunnel)))
<<>> G ((stateW=tunnel) => (~(stateE=tunnel)))
```

which were successfully model checked.

Absence of Knowledge

- When a train is away from the tunnel, it does not know whether or not the other train is in the tunnel.

$$
\begin{aligned}
&\langle\langle\rangle\rangle \Box (state_a \neq tunnel) \rightarrow \\
&\quad [(\neg K_a(state_b = tunnel)) \wedge (\neg K_a(state_b \neq tunnel))] \\
&\qquad\qquad\qquad\qquad (a \neq b \in \{E, W\})
\end{aligned}
$$

- For the westbound train, we do this by checking the following formulae, both of which fail.

```
<<>> G (~(stateE=tunnel)) => (stateW=tunnel)
<<>> G (~(stateE=tunnel)) => ~(stateW=tunnel)
```

Bringing about Knowledge

- Saying that Γ can bring about knowledge of φ in agent a is the same as saying:
  - agent a's state s_1 carries information φ and Γ can ensure that a enters s_1; or
  - agent a's state s_2 carries information φ and Γ can ensure that a enters s_2; or...
  - agent a's state s_n carries information φ and Γ can ensure that a enters s_n; or
- This allows us to rewrite

$$\langle\langle\Gamma\rangle\rangle\Diamond K_a\varphi$$

as:

$$\bigvee_{1\leq i\leq n}(\langle\langle\rangle\rangle\Box((state_a = s_i)\rightarrow\varphi)\wedge\langle\langle\Gamma\rangle\rangle\Diamond state_a = s_i)$$

Conclusions

- Branching time: a natural semantics for multiagent systems.
- CTL: a powerful language for representing properties of branching structures... but no notion of *agency* or *cooperation*.
- ATL: a powerful generalisation of CTL for cooperation & agency... but no notion of *knowledge*.
- ATEL: a powerful language for expressing properties of multiagent systems.