A4M33MAS - Multiagent Systems

Agents and their behaviour modeling by means of formal logic

Michal Pechoucek & Michal Jakob

Department of Computer Science
Czech Technical University in Prague

Selected graphics taken from Valentin Goranko and Wojtek Jamroga: Modal Logics for Multi-Agent Systems, 8th European Summer School in Logic Language and Information

Multi-agent systems & Logic

Multi-agent systems & Logic

- Multi-agent systems
  - Complex decentralized systems whose behaviour is given by interaction among autonomous, rational entities. We study MAS so that we understand behaviour of such systems and can design such software systems.

Multi-agent systems & Logic

- Multi-agent systems
  - Complex decentralized systems whose behaviour is given by interaction among autonomous, rational entities. We study MAS so that we understand behaviour of such systems and can design such software systems.

- Logic
  - Provides a paradigm for modeling and reasoning about the complex world in a precise and exact manner
  - Provides methodology for specification and verification of complex programs

Multi-agent systems & Logic

- Multi-agent systems
  - Complex decentralized systems whose behaviour is given by interaction among autonomous, rational entities. We study MAS so that we understand behaviour of such systems and can design such software systems.

- Logic
  - Provides a paradigm for modeling and reasoning about the complex world in a precise and exact manner
  - Provides methodology for specification and verification of complex programs

- Can be used for practical things (also in MAS):
  - automatic verification of multi-agent systems
  - and/or executable specifications of multi-agent systems

Best logic for MAS?

Modal logic

Modal logic is an extension of classical logic by new connectives $\Box$ and $\Diamond$: necessity and possibility.

- $\Box\varphi$ means that φ is necessarily true
- $\Diamond\varphi$ means that φ is possibly true

Independently of the precise definition, the following holds:

$$\Diamond\varphi \leftrightarrow \neg\Box\neg\varphi$$

Modal logic

01

Definition 1.1 (Modal Logic with n modalities)

The language of modal logic with n modal operators $\Box_1, \ldots, \Box_n$ is the smallest set containing:

- atomic propositions $p, q, r, \ldots$;
- for formulae φ, it also contains $\neg\varphi, \Box_1\varphi, \ldots, \Box_n\varphi$;
- for formulae φ, ψ, it also contains $\varphi \wedge \psi$.

We treat $\vee, \rightarrow, \leftrightarrow, \Diamond$ as macros (defined as usual).

Modal logic

Definition 1.1 (Modal Logic with n modalities)

The language of modal logic with n modal operators $\Box_1, \ldots, \Box_n$ is the smallest set containing:

- atomic propositions $p, q, r, \ldots$;
- for formulae φ, it also contains $\neg\varphi, \Box_1\varphi, \ldots, \Box_n\varphi$;
- for formulae φ, ψ, it also contains $\varphi \wedge \psi$.

We treat $\vee, \rightarrow, \leftrightarrow, \Diamond$ as macros (defined as usual).

Note that the modal operators can be nested:

$$(\Box_1\Box_2\Diamond_1 p) \;\vee\; \Box_3\neg p$$

Modal logic

More precisely, necessity/possibility is interpreted as follows:

- p is necessary $\Leftrightarrow$ p is true in all possible scenarios
- p is possible $\Leftrightarrow$ p is true in at least one possible scenario

$\rightsquigarrow$ possible worlds semantics

Modal logic

Definition 1.2 (Kripke Structure)

A Kripke structure is a tuple $\langle \mathcal{W}, \mathcal{R} \rangle$, where $\mathcal{W}$ is a set of possible worlds, and $\mathcal{R}$ is a binary relation on worlds, called accessibility relation.

Definition 1.3 (Kripke model)

A possible worlds model $\mathcal{M} = \langle \mathcal{S}, \pi \rangle$ consists of a Kripke structure $\mathcal{S}$, and a valuation of propositions $\pi : \mathcal{W} \rightarrow \mathcal{P}(\{p, q, r, \ldots\})$.

Modal logic

Remarks:

- $\mathcal{R}$ indicates which worlds are relevant for each other; $w_1 \mathcal{R} w_2$ can be read as "world w_2 is relevant for (reachable from) world w_1"
- $\mathcal{R}$ can be any binary relation from $\mathcal{W} \times \mathcal{W}$; we do not require any specific properties (yet).

Modal logic

01

Definition 1.4 (Semantics of modal logic)

The truth of formulae is relative to a Kripke model $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \pi \rangle$, and a world $w \in \mathcal{W}$. It can be defined through the following clauses:

- $\mathcal{M}, w \models p$ iff $p \in \pi(w)$;
- $\mathcal{M}, w \models \neg\varphi$ iff not $\mathcal{M}, w \models \varphi$;
- $\mathcal{M}, w \models \varphi \wedge \psi$ iff $\mathcal{M}, w \models \varphi$ and $\mathcal{M}, w \models \psi$;
- $\mathcal{M}, w \models \Box\varphi$ iff, for every $w' \in \mathcal{W}$ such that $w\mathcal{R}w'$, we have $\mathcal{M}, w' \models \varphi$.

Modal logic

Modal logic

$$\text{run} \rightarrow \Diamond \text{stop}$$

Modal logic

$$\text{run} \longrightarrow \Diamond\text{stop}$$

$$\text{stop} \longrightarrow \Box\text{stop}$$

Modal logic

$$\text{run} \longrightarrow \Diamond\text{stop}$$
$$\text{stop} \longrightarrow \Box\text{stop}$$
$$\text{run} \longrightarrow \Diamond\Box\text{stop}$$

Modal logic

Modal logic

- Note:
  - most modal logics can be translated to classical logic

    . . . *but the result looks horribly ugly,*

    . . . *and in most cases it is much harder to automatize anything*

Axiom in Modal logic

Definition 1.5 (System K)

System **K** is an extension of the propositional calculus by the axiom

Distribution axiom

$$\mathbf{K} \quad (\Box\varphi \wedge \Box(\varphi \to \psi)) \to \Box\psi$$

and the inference rule

Generalization axiom $$\frac{\varphi}{\Box\varphi}.$$

Axiom in Modal logic

Theorem 1.6 (Soundness/completeness of system K)

*System **K** is sound and complete with respect to the class of all Kripke models.*

Axiom in Modal logic

Definition 1.7 (Extending K with axioms D, T, 4, 5)

System **K** is often extended by (a subset of) the following axioms (called as below for historical reasons):

- T: $\Box\varphi \rightarrow \varphi$
- D: $\Box\varphi \rightarrow \Diamond\varphi$
- 4: $\Box\varphi \rightarrow \Box\Box\varphi$
- B: $\varphi \rightarrow \Box\Diamond\varphi$
- 5: $\Diamond\varphi \rightarrow \Box\Diamond\varphi$

Proofs

T: because $\models \varphi \Rightarrow \Box\varphi$ and due reflexivity $\forall w : (w, w) \in R$ ◎

$$\text{T: } \Box\varphi \rightarrow \varphi$$

Proofs

T: because $\models \varphi \Rightarrow \Box\varphi$ and due reflexivity $\forall w : (w, w) \in R$ ◎

D: $(\mathcal{M}_1 \models_w \varphi \, \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} \varphi)$ and due to seriality $(\mathcal{M}_1 \models_w (\exists w' : (w, w') \in R))$ we can say that $\mathcal{M}_1 \models_w \exists w'' : (w, w'') \in R : \mathcal{M}_1 \models_{w'} \varphi)$ ◎

D: $\Box\varphi \rightarrow \Diamond\varphi$

Proofs

T: because $\models \varphi \Rightarrow \Box\varphi$ and due reflexivity $\forall w : (w, w) \in R$ ⊚

D: $(\mathcal{M}_1 \models_w \varphi\ \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} \varphi)$ and due to seriality $(\mathcal{M}_1 \models_w (\exists w' : (w, w') \in R))$ we can say that $\mathcal{M}_1 \models_w \exists w'' : (w, w'') \in R : \mathcal{M}_1 \models_{w'} \varphi)$ ⊚

4: provided that there is transitive relation on R we may say that $(\mathcal{M}_1 \models_w \varphi\ \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} \varphi) \Rightarrow (\mathcal{M}_1 \models_w \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} (\forall w'' : (w', w'') \in R : \mathcal{M}_1 \models_{w''} \varphi))$ ⊚

$$4:\ \Box\varphi \rightarrow \Box\Box\varphi$$

Proofs

T: because $\models \varphi \Rightarrow \Box\varphi$ and due reflexivity $\forall w : (w, w) \in R$ ◎

D: $(\mathcal{M}_1 \models_w \varphi \; \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} \varphi)$ and due to seriality $(\mathcal{M}_1 \models_w (\exists w' : (w, w') \in R))$ we can say that $\mathcal{M}_1 \models_w \exists w'' : (w, w'') \in R : \mathcal{M}_1 \models_{w'} \varphi)$ ◎

4: provided that there is transitive relation on R we may say that $(\mathcal{M}_1 \models_w \varphi \; \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} \varphi) \Rightarrow (\mathcal{M}_1 \models_w \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} (\forall w'' : (w', w'') \in R : \mathcal{M}_1 \models_{w''} \varphi))$ ◎

B: provided that there is symetric relation on R we say that $\mathcal{M}_1 \models_w \varphi \Rightarrow \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} \exists w'' : (w', w'') \in R : \mathcal{M}_1 \models_{w''} \varphi$ if $(\forall w, w', (w, w') \in R \Rightarrow (w', w) \in R)$ then $w = w''$ and $\mathcal{M}_1 \models_w \varphi$ ◎

$$\text{B: } \varphi \rightarrow \Box\Diamond\varphi$$

Proofs

T: because $\models \varphi \Rightarrow \Box\varphi$ and due reflexivity $\forall w : (w, w) \in R$ ◎

D: $(\mathcal{M}_1 \models_w \varphi\ \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} \varphi)$ and due to seriality $(\mathcal{M}_1 \models_w (\exists w' : (w, w') \in R))$ we can say that $\mathcal{M}_1 \models_w \exists w'' : (w, w'') \in R : \mathcal{M}_1 \models_{w'} \varphi)$ ◎

4: provided that there is transitive relation on R we may say that $(\mathcal{M}_1 \models_w \varphi\ \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} \varphi) \Rightarrow (\mathcal{M}_1 \models_w \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} (\forall w'' : (w', w'') \in R : \mathcal{M}_1 \models_{w''} \varphi))$ ◎

B: provided that there is symetric relation on R we say that $\mathcal{M}_1 \models_w \varphi \Rightarrow \forall w' : (w, w') \in R : \mathcal{M}_1 \models_{w'} \exists w'' : (w', w'') \in R : \mathcal{M}_1 \models_{w''} \varphi$ if $(\forall w, w', (w, w') \in R \Rightarrow (w', w) \in R)$ then $w = w''$ and $\mathcal{M}_1 \models_w \varphi$ ◎

5: $(\mathcal{M}_1 \models_w \exists w' : (w, w') \in R \models_{w'} \varphi) \Rightarrow (\mathcal{M}_1 \models_w \forall w'' : (w, w'') \in R : \mathcal{M}_1 \models_{w''} \exists w'(w'', w') \in R : \mathcal{M}_1 \models_{w'} \varphi)$ due to euclidean property if $(w, w') \in R \wedge (w, w'') \in R$ then $(w', w'') \in R$ ◎

5: $\Diamond\varphi \rightarrow \Box\Diamond\varphi$

Axiom in Modal logic

- T: $\Box\varphi \rightarrow \varphi$ due to reflexivity
- D: $\Box\varphi \rightarrow \Diamond\varphi$ due to seriality
- 4: $\Box\varphi \rightarrow \Box\Box\varphi$ due to transitivity
- B: $\varphi \rightarrow \Box\Diamond\varphi$ due to symetricity
- 5: $\Diamond\varphi \rightarrow \Box\Diamond\varphi$ due to euclidean property

Model of Belief

Model of Belief

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:

Model of Belief

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:
  - to satisfy the K axioms

Model of Belief

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:
  - to satisfy the K axioms
  - *an agent knows what it does know*: positive introspection axiom (4 axiom).

Model of Belief

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:
  - to satisfy the K axioms
  - *an agent knows what it does know*: positive introspection axiom (4 axiom).
  - *an agent knows what it does not know*: positive introspection axiom (5 axiom).

Model of Belief

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:
  - to satisfy the K axioms
  - *an agent knows what it does know*: positive introspection axiom (4 axiom).
  - *an agent knows what it does not know*: positive introspection axiom (5 axiom).
  - *it beliefs are not contradictory*: if it knows something it means it does not allow the negation of its being true (D axiom).

Model of Belief

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:
  - to satisfy the K axioms
  - *an agent knows what it does know*: positive introspection axiom (4 axiom).
  - *an agent knows what it does not know*: positive introspection axiom (5 axiom).
  - *it beliefs are not contradictory*: if it knows something it means it does not allow the negation of its being true (D axiom).

Model of Belief

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:
  - to satisfy the K axioms
  - *an agent knows what it does know*: positive introspection axiom (4 axiom).
  - *an agent knows what it does not know*: positive introspection axiom (5 axiom).
  - *it beliefs are not contradictory*: if it knows something it means it does not allow the negation of its being true (D axiom).

- Belief is surely a KD45 system -- modal logic system where the B relation is serial, transitive and euclidean.

Model of Belief

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:
  - to satisfy the K axioms
  - *an agent knows what it does know*: positive introspection axiom (4 axiom).
  - *an agent knows what it does not know*: positive introspection axiom (5 axiom).
  - *it beliefs are not contradictory*: if it knows something it means it does not allow the negation of its being true (D axiom).

- Belief is surely a KD45 system -- modal logic system where the B relation is serial, transitive and euclidean.

Model of Belief

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:
  - to satisfy the K axioms
  - *an agent knows what it does know*: positive introspection axiom (4 axiom).
  - *an agent knows what it does not know*: positive introspection axiom (5 axiom).
  - *it beliefs are not contradictory*: if it knows something it means it does not allow the negation of its being true (D axiom).

- Belief is surely a KD45 system -- modal logic system where the B relation is serial, transitive and euclidean.

- Knowledge is more difficult – it needs to be also true – this why the knowledge accessibility relation needs to be also reflexive.

Model of Belief

01

- Once we are implementing an intelligent agent what do we want the program to implement e.g. its beliefs:
  - to satisfy the K axioms
  - *an agent knows what it does know*: positive introspection axiom (4 axiom).
  - *an agent knows what it does not know*: positive introspection axiom (5 axiom).
  - *it beliefs are not contradictory*: if it knows something it means it does not allow the negation of its being true (D axiom).

- Belief is surely a KD45 system -- modal logic system where the B relation is serial, transitive and euclidean.

- Knowledge is more difficult – it needs to be also true – this why the knowledge accessibility relation needs to be also reflexive.
- Therefore knowledge is a KTD45 system.

Model of Belief

- φ can be true in $\mathcal{M}$ and q $(\mathcal{M}, q \models \varphi)$
- φ can be valid in $\mathcal{M}$ $(\mathcal{M}, q \models \varphi$ for all $q)$
- φ can be valid $(\mathcal{M}, q \models \varphi$ for all $\mathcal{M}, q)$
- φ can be satisfiable $(\mathcal{M}, q \models \varphi$ for some $\mathcal{M}, q)$
- φ can be a theorem (it can be derived from the axioms via inference rules)

Model of Belief

- model checking (local): "given $\mathcal{M}$, q, and φ, is φ true in $\mathcal{M}, q$?"
- model checking (global): "given $\mathcal{M}$ and φ, what is the set of states in which φ is true?"

- Model checking is a technique for automatically verifying correctness properties of finite-state systems. Given a model of a system, exhaustively and automatically check whether this model meets a given specification (such as the absence of deadlocks and similar critical states that can cause the system to crash).

Model of Belief

- model checking (local): “given $\mathcal{M}$, q, and φ, is φ true in $\mathcal{M}, q$?”
- model checking (global): “given $\mathcal{M}$ and φ, what is the set of states in which φ is true?”
- satisfiability: “given φ, is φ true in at least one model and state?”
- validity: “given φ, is φ true in all models and their states?”
- theorem proving: “given φ, is it possible to prove (derive) φ?”

Model of Belief

Modal logic is a generic framework.

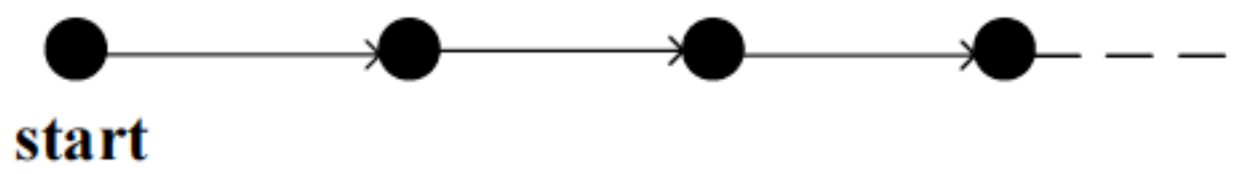
Various modal logics:

- knowledge $\rightsquigarrow$ epistemic logic,
- beliefs $\rightsquigarrow$ doxastic logic,
- obligations $\rightsquigarrow$ deontic logic,
- actions $\rightsquigarrow$ dynamic logic,
- time $\rightsquigarrow$ temporal logic,
- ability $\rightsquigarrow$ strategic logic,
- and combinations of the above

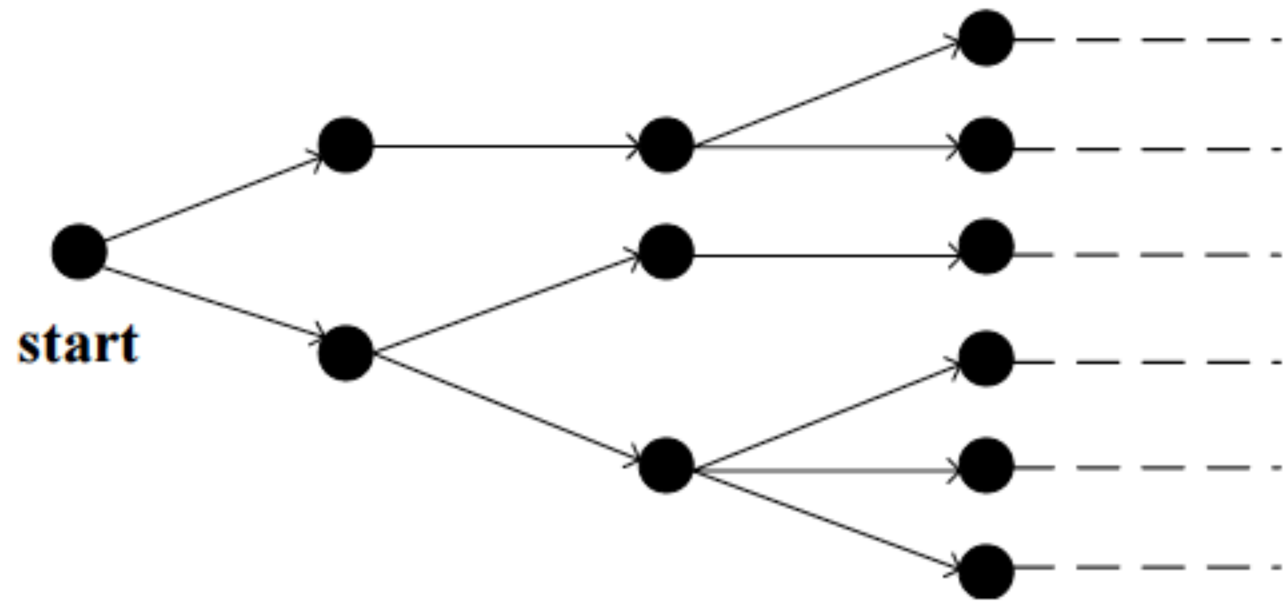
Model of Time

- Modeling time as an instance of modal logic where the accessibility relation represents the relationship between the past, current and future time moments.

- Time:
  - linear



  - branching



Typical Temporal Operators

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

Typical Temporal Operators

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

$$\mathcal{G}((\neg\text{passport} \lor \neg\text{ticket}) \;\rightarrow\; \mathcal{X}\neg\text{board\_flight})$$

$$\text{send}(\text{msg}, \text{rcvr}) \;\rightarrow\; \mathcal{F}\text{receive}(\text{msg}, \text{rcvr})$$

Safety Property

- *something bad will not happen*
- *something good will always hold*

Safety Property

– *something bad will not happen*
– *something good will always hold*

- Typical examples:

$$\mathcal{G} \neg \text{bankrupt}$$

Safety Property

– *something bad will not happen*
– *something good will always hold*

- Typical examples

$$\mathcal{G}\neg\text{bankrupt}$$
$$\mathcal{G}(\text{fuelOK} \vee \mathcal{X}\text{fuelOK})$$
and so on ...

Safety Property

– *something bad will not happen*

– *something good will always hold*

- Typical examples

$$\mathcal{G}\neg\text{bankrupt}$$

$$\mathcal{G}(\text{fuelOK} \vee \mathcal{X}\text{fuelOK})$$

and so on ...

Usually: $\mathcal{G}\neg \ldots.$

Liveness Property

– *something good will happen*

Liveness Property

– *something good will happen*

- Typical examples

$$\mathcal{F}\text{rich}$$

Liveness Property

– *something good will happen*

- Typical examples

$$\mathcal{F}\mathsf{rich}$$

$$\mathsf{rocketLondon} \rightarrow \mathcal{F}\mathsf{rocketParis}$$

and so on ...

Liveness Property

– *something good will happen*

- Typical examples

$\mathcal{F}\mathsf{rich}$

$\mathsf{rocketLondon} \longrightarrow \mathcal{F}\mathsf{rocketParis}$

and so on . . .

Usually: $\mathcal{F}$....

Fairness Property

- *Useful when scheduling processes, responding to messages, etc.*
- *Good for specifying interaction properties of the environment*

- Typical examples:

$$\mathcal{G}(\mathsf{rocketLondon} \rightarrow \mathcal{F}\mathsf{rocketParis})$$

- Strong Fairness:
  if something is attempted/requested, then it will be successful

- Typical examples:

$$\mathcal{G}(\mathsf{attempt} \rightarrow \mathcal{F}\mathsf{success})$$

$$\mathcal{GF}\mathsf{attempt} \rightarrow \mathcal{GF}\mathsf{success}$$

Linear Temporal Logic - LTL

- Reasoning about a particular computation of a system where time is linear - just one possible future path is included.

Definition 3.4 (Models of LTL)

A model of LTL is a sequence of time moments. 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
- $\lambda[i \ldots j]$: all time moments between i and j
- $\lambda[i \ldots \infty]$: all timepoints from i on

Linear Temporal Logic - LTL

Definition 3.5 (Semantics of LTL)

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

Linear Temporal Logic - LTL

Definition 3.5 (Semantics of LTL)

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

Note that:

$$\mathcal{G}\varphi \equiv \neg\mathcal{F}\neg\varphi$$

$$\mathcal{F}\varphi \equiv \neg\mathcal{G}\neg\varphi$$

$$\mathcal{F}\varphi \equiv \top\,\mathcal{U}\varphi$$

Computational Tree Logic - CTL

- Reasoning about possible computations of a system. Time is branching -- we want all alternative paths included.

- Path quantifiers: **A** (for all paths), **E** (there is a path);
- Temporal operators: $\mathcal{X}$ (nexttime), $\mathcal{F}$ (sometime), $\mathcal{G}$ (always) and $\mathcal{U}$ (until);

Computational Tree Logic - CTL

- Reasoning about possible computations of a system. Time is branching -- we want all alternative paths included.

  - Path quantifiers: **A** (for all paths), **E** (there is a path);
  - Temporal operators: $\mathcal{X}$ (nexttime), $\mathcal{F}$ (sometime), $\mathcal{G}$ (always) and $\mathcal{U}$ (until);

- Vanilla CTL: every temporal operator must be immediately preceded by exactly one path quantier
- CTL*: no syntactic restrictions
- Reasoning in Vanilla CTL can be automatized.

Computational Tree Logic - CTL

Definition 3.8 (Semantics of CTL*: state formulae)

$M, q \models \mathbf{E}\varphi$ iff there is a path λ, starting from q, such that $M, \lambda \models \varphi$;

$M, q \models \mathbf{A}\varphi$ iff for all paths λ, starting from q, we have $M, \lambda \models \varphi$.

Computational Tree Logic - CTL

Definition 3.8 (Semantics of CTL*: state formulae)

$M, q \models \mathbf{E}\varphi$ iff there is a path λ, starting from q, such that $M, \lambda \models \varphi$;

$M, q \models \mathbf{A}\varphi$ iff for all paths λ, starting from q, we have $M, \lambda \models \varphi$.

Definition 3.9 (Semantics of CTL*: path formulae)

Exactly like for LTL!

Computational Tree Logic - CTL

Definition 3.8 (Semantics of CTL*: state formulae)

$M, q \models \mathbf{E}\varphi$ iff there is a path λ, starting from q, such that $M, \lambda \models \varphi$;

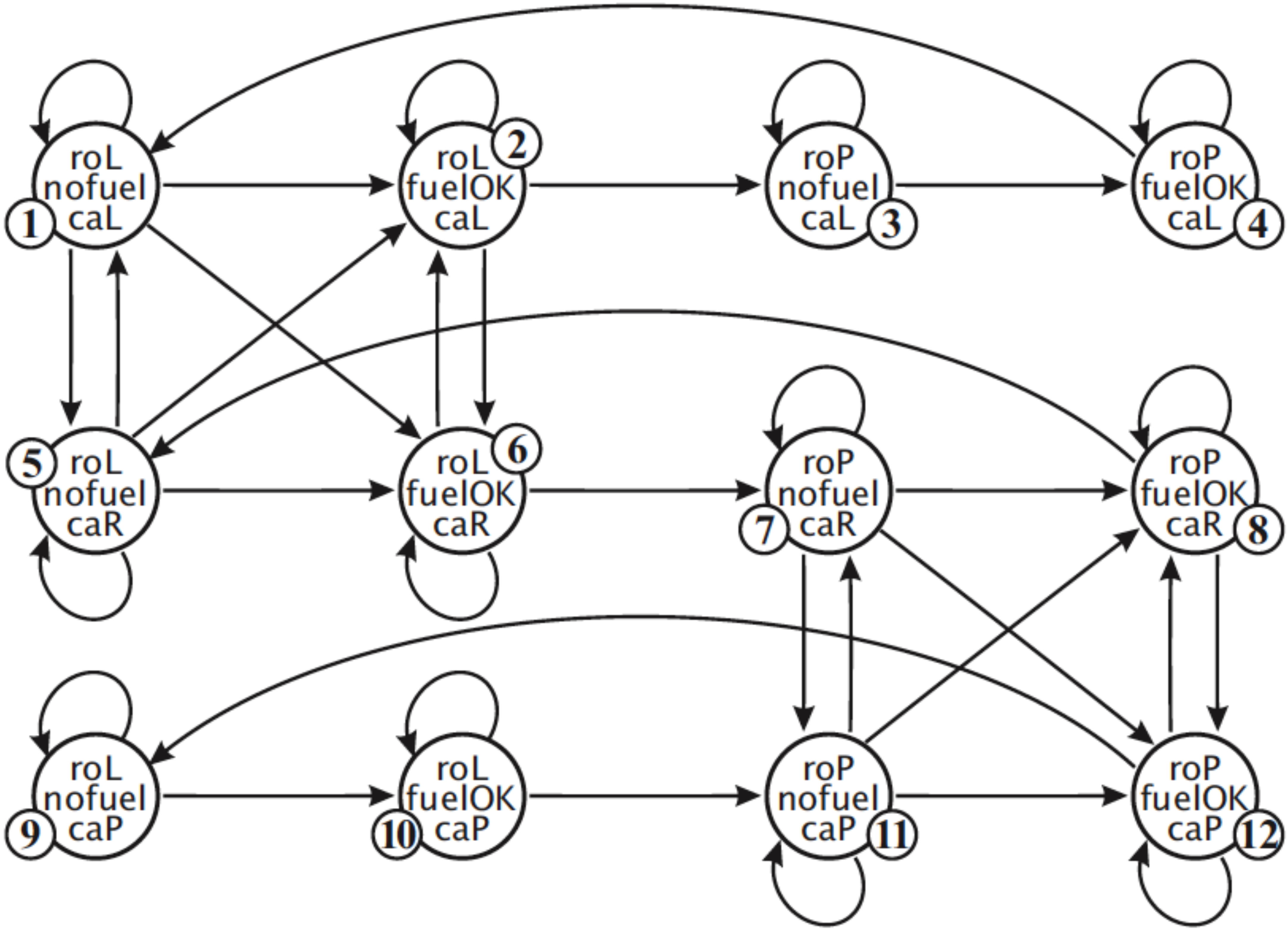
$M, q \models \mathbf{A}\varphi$ iff for all paths λ, starting from q, we have $M, \lambda \models \varphi$.

Definition 3.9 (Semantics of CTL*: path formulae)

$M, \lambda \models \mathcal{X}\varphi$ iff $M, \lambda[1...\infty] \models \varphi$;

$M, \lambda \models \varphi\,\mathcal{U}\,\psi$ iff $M, \lambda[i...\infty] \models \psi$ for some $i \geq 0$, and $M, \lambda[j...\infty] \models \varphi$ for all $0 \leq j \leq i$.

Computational Tree Logic - CTL



Dynamic Logic

Dynamic Logic

01

1^{st} idea: Consider actions or programs α. Each such α defines a transition (accessibility relation) from worlds into worlds.

Dynamic Logic

01

1^{st} idea: Consider actions or programs α. Each such α defines a transition (accessibility relation) from worlds into worlds.

2^{nd} idea: We need statements about the outcome of actions:

- $[\alpha]\varphi$: "after every execution of α, φ holds,
- $\langle\alpha\rangle\varphi$: "after some executions of α, φ holds.

Dynamic Logic

1^{st} idea: Consider actions or programs α. Each such α defines a transition (accessibility relation) from worlds into worlds.

2^{nd} idea: We need statements about the outcome of actions:

- $[\alpha]\varphi$: "after every execution of α, φ holds,
- $\langle\alpha\rangle\varphi$: "after some executions of α, φ holds.

As usual, $\langle\alpha\rangle\varphi \equiv \neg[\alpha]\neg\varphi$.

Dynamic Logic

3rd idea: Programs/actions can be combined (sequentially, nondeterministically, iteratively), e.g.:

$$[\alpha;\beta]\varphi$$

would mean "after every execution of α and then β, formula φ holds".

Dynamic Logic

Definition 3.1 (Labelled Transition System)

A labelled transition system is a pair

$$\langle St, \{\xrightarrow{\alpha} : \alpha \in \mathbf{L}\}\rangle$$

where St is a non-empty set of states and $\mathbf{L}$ is a non-empty set of labels and for each $\alpha \in \mathbf{L}$: $\xrightarrow{\alpha} \subseteq St \times St$.

Dynamic Logic

Definition 3.1 (Labelled Transition System)

A labelled transition system is a pair

$$\langle St, \{\xrightarrow{\alpha} : \ \alpha \in \mathbf{L}\}\rangle$$

where St is a non-empty set of states and $\mathbf{L}$ is a non-empty set of labels and for each $\alpha \in \mathbf{L}$: $\xrightarrow{\alpha} \subseteq St \times St$.

Definition 3.2 (Dynamic Logic: Models)

A model of propositional dynamic logic is given by a labelled transition systems and an evaluation of propositions.

Dynamic Logic

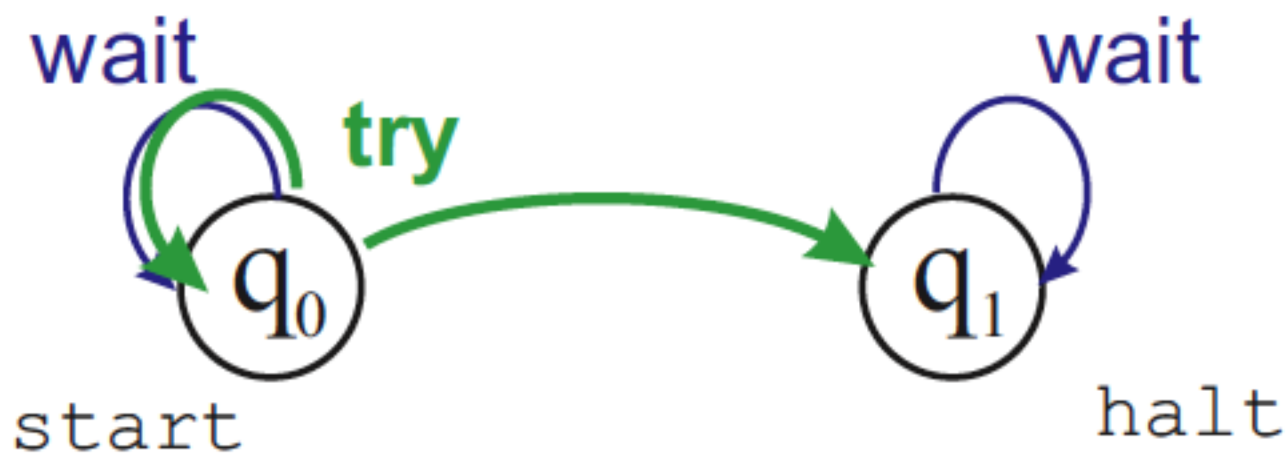
Definition 3.2 (Dynamic Logic: Models)

A model of propositional dynamic logic is given by a labelled transition systems and an evaluation of propositions.

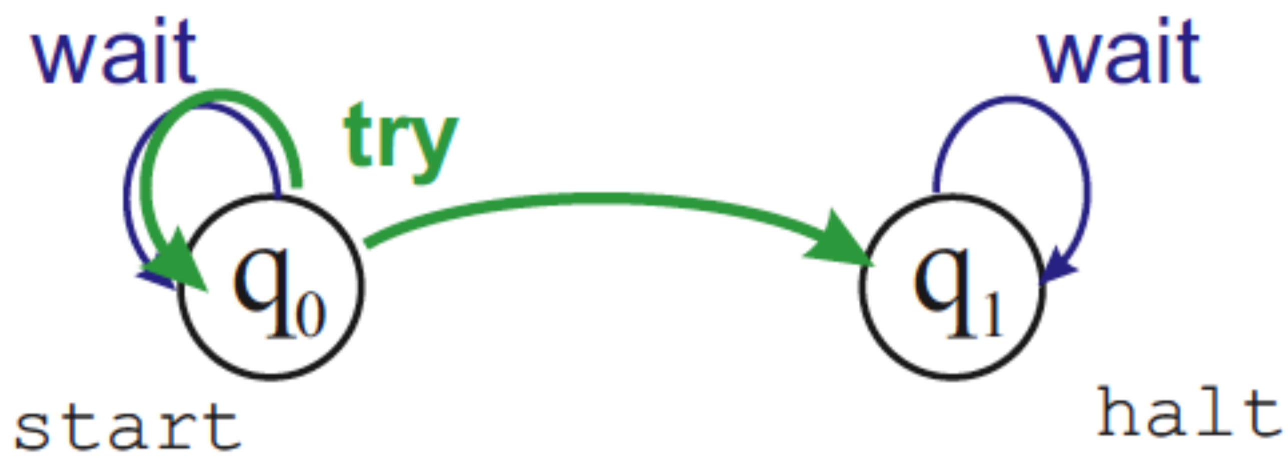
Definition 3.3 (Semantics of DL)

$\mathcal{M}, s \models [\alpha]\varphi$ iff for every t such that $s \xrightarrow{\alpha} t$, we have $\mathcal{M}, t \models \varphi$.

Dynamic Logic

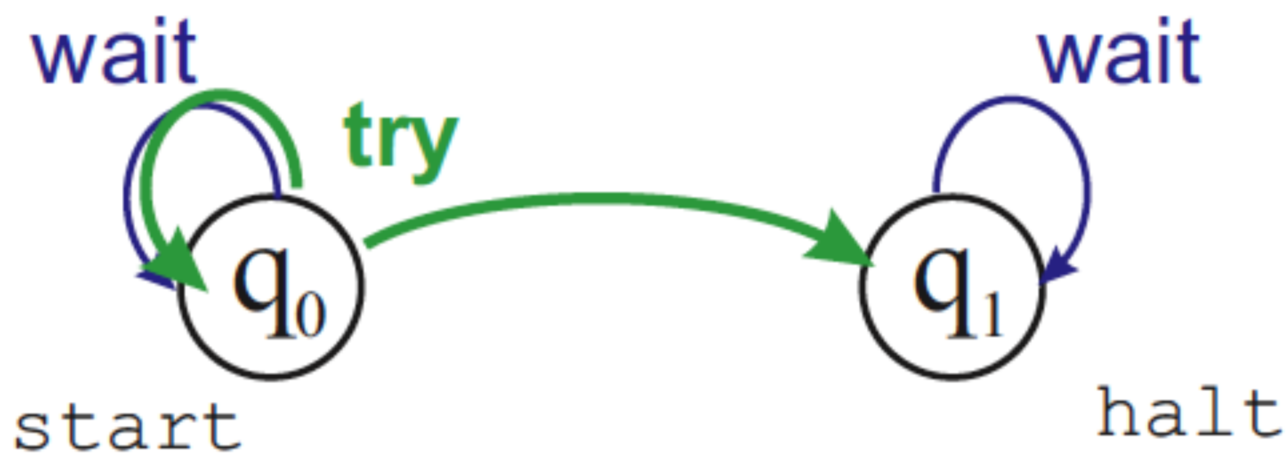


Dynamic Logic



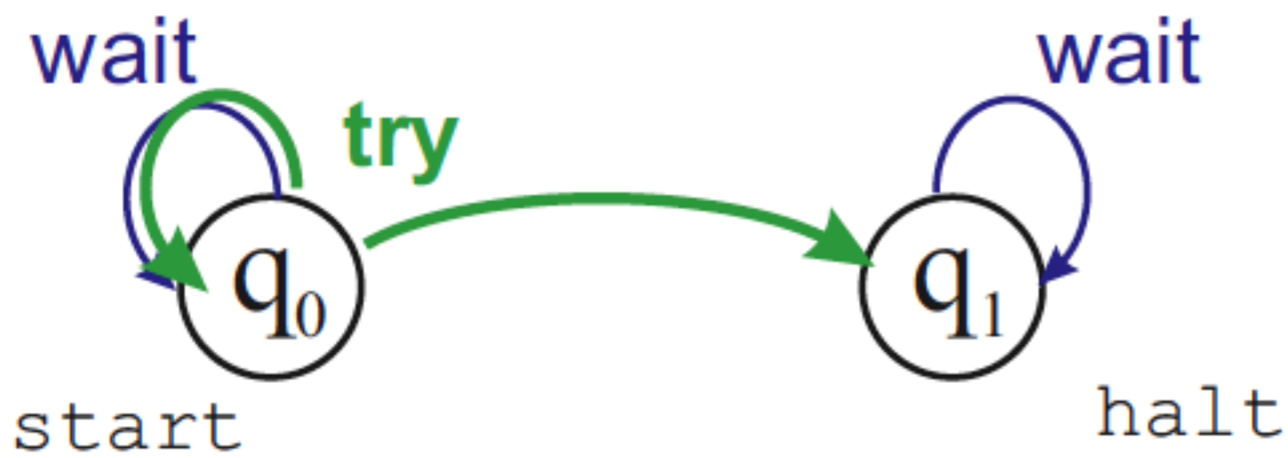
$$\text{start} \rightarrow \langle \mathit{try} \rangle \text{halt}$$

Dynamic Logic



$$\text{start} \longrightarrow \langle \mathit{try} \rangle \text{halt}$$
$$\text{start} \longrightarrow \neg[\mathit{try}]\text{halt}$$

Dynamic Logic



$$\mathsf{start} \longrightarrow \langle \mathit{try} \rangle \mathsf{halt}$$

$$\mathsf{start} \longrightarrow \neg[\mathit{try}]\mathsf{halt}$$

$$\mathsf{start} \longrightarrow \langle \mathit{try} \rangle [\mathit{wait}]\mathsf{halt}$$

Concluding Remarks

- Practical Importance of Temporal and Dynamic Logics:
  - Automatic verication in principle possible (model checking).
  - Can be used for automated planning.
  - Executable specications can be used for programming.

- Note:

  When we combine time and actions with knowledge (beliefs, desires, intentions, obligations...), we finally obtain a fairly realistic model of MAS.