ATL with Dependent Strategies

Jessica Newman
School of Electronics and Computer Science
University of Southampton

April 14, 2025

Overview

1. **Motivation**
2. **ATLDS**
3. **Effectivity**
4. **Axiomatisation**
5. **Model Checking**
6. **Conclusion**

Motivation

Modelling Multi-Agent Systems

Dynamic systems of multiple autonomous agents are common.

We want to model multi-agent systems and verify properties we can guarantee in them.

Specifically, what any group of agents can guarantee.

Modelling Multi-Agent Systems

$$\varphi ::= p | \neg\varphi | (\varphi \wedge \varphi) | \langle\langle C \rangle\rangle \mathrm{X}\varphi | \langle\langle C \rangle\rangle \mathrm{G}\varphi | \langle\langle C \rangle\rangle \varphi \mathrm{U} \varphi$$

Alternating-Time Temporal Logic (ATL) models temporal properties in multi-agent systems.

Extends CTL with quantifiers indexed by sets of agents.

Modelling Multi-Agent Systems

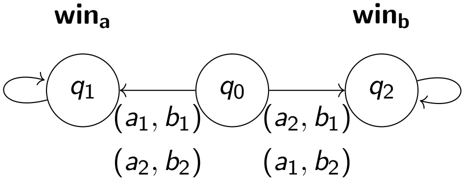
$$\varphi ::= p | \neg\varphi | (\varphi \wedge \varphi) | \langle\langle C \rangle\rangle \mathrm{X}\varphi | \langle\langle C \rangle\rangle \mathrm{G}\varphi | \langle\langle C \rangle\rangle \varphi \mathrm{U} \varphi$$

$\langle\langle C \rangle\rangle \mathrm{X}\ win$
Coalition C has a strategy to win at the next step.

$\langle\langle \{a, b\} \rangle\rangle \mathrm{G}(\ req_i \to \forall \mathrm{F}\ grant_i)$ [1]
Coalition $\{a, b\}$ has a strategy s.t. whenever request i is made, it is always granted at some point in the future.

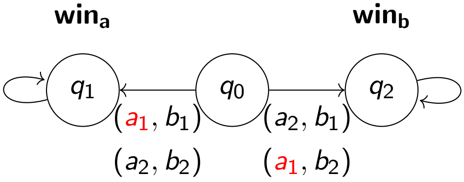
[1] $\forall$ shorthand for $\langle\langle \emptyset \rangle\rangle$

Modelling Multi-Agent Systems



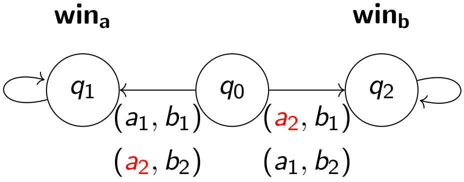
- a has moves $\{a_1, a_2\}$, b has moves $\{b_1, b_2\}$
- Strategies chosen concurrently:
- a chooses $\alpha \in \{a_1, a_2\}$ without knowing what b chooses.
- $q_0 \models \neg\langle\langle\{a\}\rangle\rangle \mathrm{X}\ win_a$

Modelling Multi-Agent Systems



- a has moves $\{a_1, a_2\}$, b has moves $\{b_1, b_2\}$
- Strategies chosen concurrently:
- a chooses $\alpha \in \{a_1, a_2\}$ without knowing what b chooses.
- $q_0 \models \neg \langle\langle \{a\} \rangle\rangle \mathrm{X}\ win_a$

Modelling Multi-Agent Systems



- a has moves $\{a_1, a_2\}$, b has moves $\{b_1, b_2\}$
- Strategies chosen concurrently:
- a chooses $\alpha \in \{a_1, a_2\}$ without knowing what b chooses.
- $q_0 \models \neg\langle\langle\{a\}\rangle\rangle \mathrm{X}\ win_a$

Modelling Multi-Agent Systems

Concurrent games do not entirely describe strategic ability.

Some game theoretic properties require order: a moves after b.

e.g. Stackelberg games.

Want to allow agents to condition their strategy on other agents:

- Express things that are inexpressible in ATL.
- But maintain nice properties of ATL.

ATLDS

Alternating-Time Temporal Logic with Dependent Strategies

ATLDS extends ATL by augmenting each $\langle\langle C \rangle\rangle$ with a permutation P of Ag.

$$\varphi ::= p | \neg\varphi | (\varphi \land \varphi) | \langle\langle C \rangle\rangle_P \mathrm{X}\varphi | \langle\langle C \rangle\rangle_P \varphi \mathrm{U} \varphi$$

$\langle\langle C \rangle\rangle_P \varphi$
'the agents in C, when selecting moves in the order of P, have a collective strategy to enforce φ'.

Alternating-Time Temporal Logic with Dependent Strategies

ATLDS extends ATL by augmenting each $\langle\langle C \rangle\rangle$ with a permutation P of Ag.

$$\varphi ::= p|\neg\varphi|(\varphi \wedge \varphi)|\langle\langle C \rangle\rangle_P \mathrm{X}\varphi|\langle\langle C \rangle\rangle_P \varphi \mathrm{U} \varphi$$

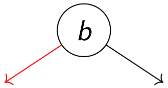
$\langle\langle \{a, b\} \rangle\rangle_{(b,c,a)} \mathrm{X}\, Goal$
'agent b has a strategy,
such that for all strategies of c,
agent a has a strategy
that guarantees *Goal* is achieved at
the next step'

Alternating-Time Temporal Logic with Dependent Strategies

ATLDS extends ATL by augmenting each $\langle\langle C \rangle\rangle$ with a permutation P of Ag.

$$\varphi ::= p | \neg\varphi | (\varphi \wedge \varphi) | \langle\langle C \rangle\rangle_P \mathrm{X}\varphi | \langle\langle C \rangle\rangle_P \varphi \mathrm{U} \varphi$$

$\langle\langle \{a, b\} \rangle\rangle_{(b,c,a)} \mathrm{X}\, Goal$
'agent b has a strategy,
such that for all strategies of c,
agent a has a strategy
that guarantees $Goal$ is achieved at
the next step'

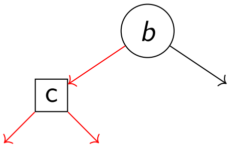


Alternating-Time Temporal Logic with Dependent Strategies

ATLDS extends ATL by augmenting each $\langle\langle C \rangle\rangle$ with a permutation P of Ag.

$$\varphi ::= p | \neg\varphi | (\varphi \wedge \varphi) | \langle\langle C \rangle\rangle_P \mathrm{X}\varphi | \langle\langle C \rangle\rangle_P \varphi \mathrm{U} \varphi$$

$\langle\langle \{a, b\} \rangle\rangle_{(b,c,a)} \mathrm{X}\, Goal$
'agent b has a strategy,
such that for all strategies of c,
agent a has a strategy
that guarantees $Goal$ is achieved at
the next step'

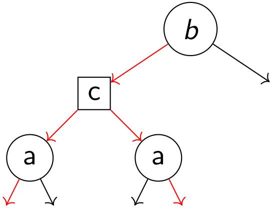


Alternating-Time Temporal Logic with Dependent Strategies

ATLDS extends ATL by augmenting each $\langle\langle C \rangle\rangle$ with a permutation P of Ag.

$$\varphi ::= p|\neg\varphi|(\varphi \wedge \varphi)|\langle\langle C \rangle\rangle_P \mathrm{X}\varphi|\langle\langle C \rangle\rangle_P \varphi \mathrm{U} \varphi$$

$\langle\langle\{a, b\}\rangle\rangle_{(b,c,a)}\mathrm{X}\,Goal$
'agent b has a strategy,
such that for all strategies of c,
agent a has a strategy
that guarantees $Goal$ is achieved at
the next step'

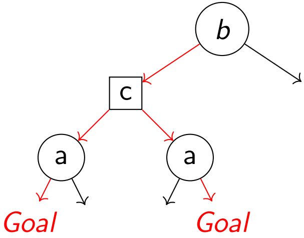


Alternating-Time Temporal Logic with Dependent Strategies

ATLDS extends ATL by augmenting each $\langle\langle C \rangle\rangle$ with a permutation P of Ag.

$$\varphi ::= p | \neg\varphi | (\varphi \wedge \varphi) | \langle\langle C \rangle\rangle_P \mathrm{X}\varphi | \langle\langle C \rangle\rangle_P \varphi \mathrm{U} \varphi$$

$\langle\langle \{a, b\} \rangle\rangle_{(b,c,a)} \mathrm{X} Goal$
'agent b has a strategy,
such that for all strategies of c,
agent a has a strategy
that guarantees *Goal* is achieved at the next step'



Alternating-Time Temporal Logic with Dependent Strategies

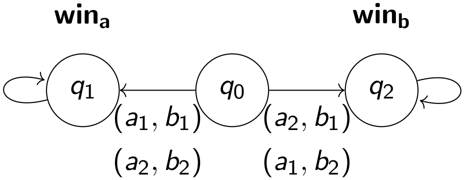
Now a strategy is a function from moves of previous agents.

Under permutation (b, c, a):

A strategy for b is just fixing an action.

A strategy for a is a function from actions of b and c to actions of a.

Matching Pennies



$q_0 \models \neg\langle\langle\{a\}\rangle\rangle X win_a$

$q_0 \models \langle\langle\{a\}\rangle\rangle_{(b,a)} X win_a$

$b_1 \mapsto a_1$
$b_2 \mapsto a_2$

Responding to Strategies

Similarities:
The fixpoint characterisation of temporal formulae still hold, e.g.

$$\langle\!\langle C \rangle\!\rangle_P \mathsf{G}\varphi \leftrightarrow \varphi \wedge \langle\!\langle C \rangle\!\rangle_P \mathsf{X} \langle\!\langle C \rangle\!\rangle_P \mathsf{G}\varphi$$

Differences:
The fact of whether a coalition C can force φ under P is determined:

$$\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\varphi \leftrightarrow \neg \langle\!\langle \overline{C} \rangle\!\rangle_P \mathsf{X}\neg\varphi$$

ATLDS is more expressive for $|Ag| > 2$.

Characterisation

- So we know ATLDS behaves a lot like ATL, but how exactly can we characterise the differences?
- It's hard to construct games with required properties...
- Idea: go from abstract description $\rightarrow$ game

Effectivity

Effectivity Functions

$$E(C, P) = \{X_1, X_2, \dots, X_k\}$$

$$X_i \subseteq Q$$

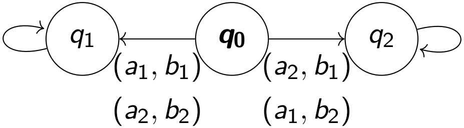
Map from C, P to sets of states C can enforce under P.

$\{q_1, q_2\} \in E(\{a\}, (a, b, c))$
means a can guarantee either state q_0 or q_1 under permutation (a, b, c).

We can construct an effectivity function E_q^π to represent the transition function at a state.

Effectivity Functions

Can represent transition function at a state as a game:



$\downarrow$

	$\mathbf{a_1}$	$\mathbf{a_2}$
$\mathbf{b_1}$	q_1	q_2
$\mathbf{b_2}$	q_2	q_1

π-Effectivity

For example, take $C = \{a\}$ and $P = (b, a)$ in the following game:

	$\mathbf{b_1}$	$\mathbf{b_2}$	$\mathbf{b_3}$
$\mathbf{a_1}$	x	z	z
$\mathbf{a_2}$	y	x	x

A strategy for a is a choice of a_1 or a_2 for *every* move from b.

i.e. set of a's strategies is set of functions from moves of b to moves of a.

π-Effectivity

For example, take $C = \{a\}$ and $P = (b, a)$ in the following game:

	b₁	**b₂**	**b₃**
a_1	x	z	z
a_2	y	x	x

Let us take the joint strategy where a chooses a_2 and c chooses the following responses:

$$b_1 \mapsto a_2$$

$$b_2 \mapsto a_1$$

$$b_3 \mapsto a_1$$

No matter the choice of b, the outcome is x. So $\{y, z\} \in E_q^\pi(\{a\}, (b, a))$

Neighbourhood Models

So we can have an *effectivity function* at each state instead of a game:

$$\mathcal{S} = (Ag, Q, (E_q^\pi)_{q \in Q})$$

$q \models \langle\langle C \rangle\rangle_P \mathrm{X}\varphi$ iff $[\![\varphi]\!] \in E_q^\pi(C, P)$

We get a neighbourhood model by calculating E_q^π at each state...

Much easier to construct models, but how can we get a game back from an effectivity function?

α-Effectivity

Already known for ATL -
'Truly playable':

1. (outcome monotonic) $X \in E(C)$ implies $Y \in E(C)$ for all $Y \supseteq X$
2. (superadditivity) $X \in E(C)$ and $Y \in E(S)$ implies $X \cap Y \in E(C \cup S)$ for disjoint C, S
3. (N-maximality) $X \notin E(\emptyset) \implies \overline{X} \in E(N)$
4. (liveness) $\emptyset \notin E(C)$
5. (safety) $E(C) \neq \emptyset$
6. (regularity) $X \in E(C) \implies \overline{X} \notin E(\overline{C})$
7. (crown condition) $X \in E(N)$ implies there is some $x \in X$ such that $\{x\} \in E(N)$

Back and Forth between Games and Effectivity

Existing construction that goes from truly playable effectivity function to a game G.

Theorem (Pauly's Representation Theorem [Goranko et al., 2013])

An effectivity function E is truly playable iff $E = E_G^\alpha$ for some normal-form game G

π-Effectivity

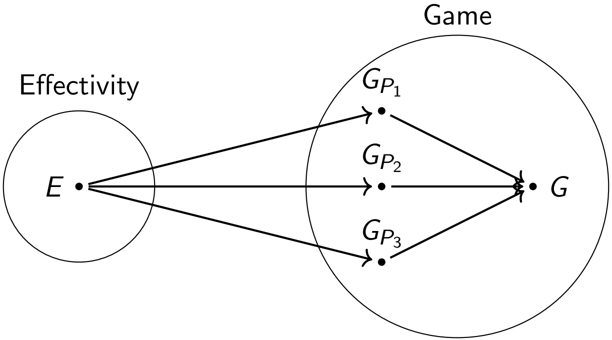
Truly playable + maximal + order monotonic:

1. (outcome monotonic) $X \in E(C, P)$ implies $Y \in E(C, P)$ for all $Y \supseteq X$
2. (superadditivity) $X \in E(C, P)$ and $Y \in E(S, P)$ implies $X \cap Y \in E(C \cup S, P)$ for disjoint C, S
3. **(maximality)** $\boldsymbol{X \notin E(C, P) \implies \overline{X} \in E(\overline{C}, P)}$
4. (liveness) $\emptyset \notin E(C, P)$
5. (safety) $E(C, P) \neq \emptyset$
6. **(order monotonicity)** $\boldsymbol{X \in E(C, P)}$ **implies** $\boldsymbol{X \in E(C, P')}$ **for** $\boldsymbol{P \leq_C P'}$
7. (regularity) $X \in E(C, P) \implies \overline{X} \notin E(\overline{C}, P)$
8. (crown condition) $X \in E(N, P)$ implies there is some $x \in X$ such that $\{x\} \in E(N, P)$

Back and Forth between Games and Effectivity

Theorem

An ordered effectivity function E is order-monotonic, maximal, and truly playable iff $E = E_G^\pi$ for some normal-form game G



Axiomatisation

Effectivity Axioms into ATLDS Axioms

- (outcome monotonic) $X \in E(C, P)$ implies $Y \in E(C, P)$ for all $Y \supseteq X$
- (superadditivity) $X \in E(C, P)$ and $Y \in E(S, P)$ implies $X \cap Y \in E(C \cup S, P)$ for disjoint C, S
- (maximality) $X \notin E(C, P) \implies \overline{X} \in E(\overline{C}, P)$
- (liveness) $\emptyset \notin E(C, P)$
- (safety) $E(C, P) \neq \emptyset$
- (order monotonicity) $X \in E(C, P)$ implies $X \in E(C, P')$ for $P \leq_C P'$

Effectivity Axioms into ATLDS Axioms

- (X-Monotonicity) $\dfrac{\varphi \implies \psi}{\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\varphi \implies \langle\!\langle C \rangle\!\rangle_P \mathsf{X}\psi}$
- (S) $\langle\!\langle C_1 \rangle\!\rangle_P \mathsf{X}\varphi \wedge \langle\!\langle C_2 \rangle\!\rangle_P \mathsf{X}\psi \implies \langle\!\langle C_1 \cup C_2 \rangle\!\rangle_P \mathsf{X}(\varphi \wedge \psi)$ (for disjoint C_1, C_2)
- (M) $\neg\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\varphi \implies \langle\!\langle \overline{C} \rangle\!\rangle_P \mathsf{X}\neg\varphi$
- ($\bot$) $\neg\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\bot$
- ($\top$) $\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\top$
- (Mon) $\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\varphi \implies \langle\!\langle C \rangle\!\rangle_{P'} \mathsf{X}\varphi$ (for $P \leq_C P'$)

Effectivity Axioms into ATLDS Axioms

- (X-Monotonicity) $\dfrac{\varphi \implies \psi}{\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\varphi \implies \langle\!\langle C \rangle\!\rangle_P \mathsf{X}\psi}$
- (S) $\langle\!\langle C_1 \rangle\!\rangle_P \mathsf{X}\varphi \wedge \langle\!\langle C_2 \rangle\!\rangle_P \mathsf{X}\psi \implies \langle\!\langle C_1 \cup C_2 \rangle\!\rangle_P \mathsf{X}(\varphi \wedge \psi)$ (for disjoint C_1, C_2)
- (M) $\neg\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\varphi \implies \langle\!\langle \overline{C} \rangle\!\rangle_P \mathsf{X}\neg\varphi$
- ($\bot$) $\neg\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\bot$
- ($\top$) $\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\top$
- (Mon) $\langle\!\langle C \rangle\!\rangle_P \mathsf{X}\varphi \implies \langle\!\langle C \rangle\!\rangle_{P'} \mathsf{X}\varphi$ (for $P \leq_C P'$)

+

- (FP) $\langle\!\langle C \rangle\!\rangle_P \psi \mathsf{U} \varphi \iff \varphi \vee (\psi \wedge \langle\!\langle C \rangle\!\rangle_P \mathsf{X} \langle\!\langle C \rangle\!\rangle_P \psi \mathsf{U} \varphi)$
- (LFP) $\langle\!\langle \emptyset \rangle\!\rangle \mathsf{G}((\varphi \vee (\psi \wedge \langle\!\langle C \rangle\!\rangle_P \mathsf{X}\chi)) \implies \chi) \implies \langle\!\langle \emptyset \rangle\!\rangle \mathsf{G}(\langle\!\langle C \rangle\!\rangle_P \psi \mathsf{U} \varphi \implies \chi)$
- (G-Necessitation) $\dfrac{\varphi}{\langle\!\langle C \rangle\!\rangle_P \mathsf{G}\varphi}$

Soundness and Completeness

The ATLDS axioms do not guarantee the property:

(crown condition) $X \in E(N, P)$ implies there is some $x \in X$ such that $\{x\} \in E(N, P)$

But on finite models this is guaranteed from the other axioms.

Finite Model Property

More expressive fragment of Strategy Logic enjoys the finitely-branching tree model property [Mogavero et al., 2016].

So via a filtration-style process:

Proposition

ATLDS has the finite model property

Soundness and Completeness

Theorem

The axiomatic system for ATLDS is sound and weakly complete for order-monotonic, maximal, and (truly) playable effectivity models.

And from the π-effectivity representation theorem, we get:

Corollary

The axiomatic system for ATLDS is sound and weakly complete for CGMs.

Model Checking

Model Checking

Can adapt algorithm for ATL for PTIME model checking...

...But only when transition functions are listed explicitly ($|Q| \times |A|^{|Ag|}$)

We can look at *implicit* CGMs, where transition function is encoded polynomially.

Model Checking

When restricted to implicit CGMs:

The model checking problem for ATLDS is PSPACE-complete.

The model checking problem for ATLDS with a fixed no. of agents is in $\mathrm{NP} \cap \mathrm{coNP}$.

The model checking problem for ATLDS restricted to formulae with k quantifier alternations is in Δ^P_{k+1}.

Conclusion

Conclusion

- We can add in dependency/order required for certain game-theoretic concepts to ATL.
- Still behaves like ATL in appropriate ways.
- We incur a small cost for model checking in certain scenarios.

Future Directions

- More efficient construction from neighbourhood models to games.
- Branching/Independence Friendly Quantifiers
- Where else can effectivity take us?

References

Belardinelli, F., Jamroga, W., Kurpiewski, D., Malvone, V., and Murano, A. (2019).
Strategy logic with simple goals: Tractable reasoning about strategies.
In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 88–94. International Joint Conferences on Artificial Intelligence Organization.

Goranko, V., Jamroga, W., and Turrini, P. (2013).
Strategic games and truly playable effectivity functions.
Autonomous Agents and Multi-Agent Systems, 26(2):288–314.

Mogavero, F., Murano, A., Perelli, G., and Vardi, M. (2016).
Reasoning about strategies: on the satisfiability problem.
Logical Methods in Computer Science, Vol. 13(1:9)2017, pp. 1–37.

The End