

A comparison of automata and tableau methods for modal satisfiability*

Franz Baader
Theoretical Computer Science
RWTH Aachen

- Advantages and disadvantages of both methods
- Intuitive comparison for K with global axioms
- Formal connection between the automata approach and the inverse tableau method for K

Methods for modal satisfiability

- Semantic tableaux and related methods.
- Translation into classical first-order logic and application of general theorem provers.
- Translation into quantified Boolean formulae and application of QBF solvers.
- Reduction to the emptiness problem for certain tree automata.
- Others: show finite model property; mosaic method; ...

Methods for modal satisfiability

- Semantic tableaux and related methods.
- Translation into classical first-order logic and application of general theorem provers.
- Translation into quantified Boolean formulae and application of QBF solvers.
- Reduction to the emptiness problem for certain tree automata.
- Others: show finite model property; mosaic method; ...

Methods for modal satisfiability

- Semantic tableaux and related methods.
- Translation into classical first-order logic and application of general theorem provers.
- Translation into quantified Boolean formulae and application of QBF solvers.
- Reduction to the emptiness problem for certain tree automata.
- Others: show finite model property; mosaic method; ...

Advantages and disadvantages

for the case of
K with global axioms

tableau-based

- + optimized implementations
- + behave well in practice
- hard to obtain exact worst-case complexity upper-bound:
 - "natural" tableau algorithm is NExpTime
 - problem is ExpTime-complete

tree automata-based

- implementations?
- best-case exponential
- + easy to obtain exact worst-case complexity upper-bound:
 - + "natural" approach yields ExpTime algorithm

Advantages and disadvantages

for the case of
K with global axioms

tableau-based

- + optimized implementations
- + behave well in practice
- hard to obtain exact worst-case complexity upper-bound:
 - "natural" tableau algorithm is NExpTime
 - problem is ExpTime-complete

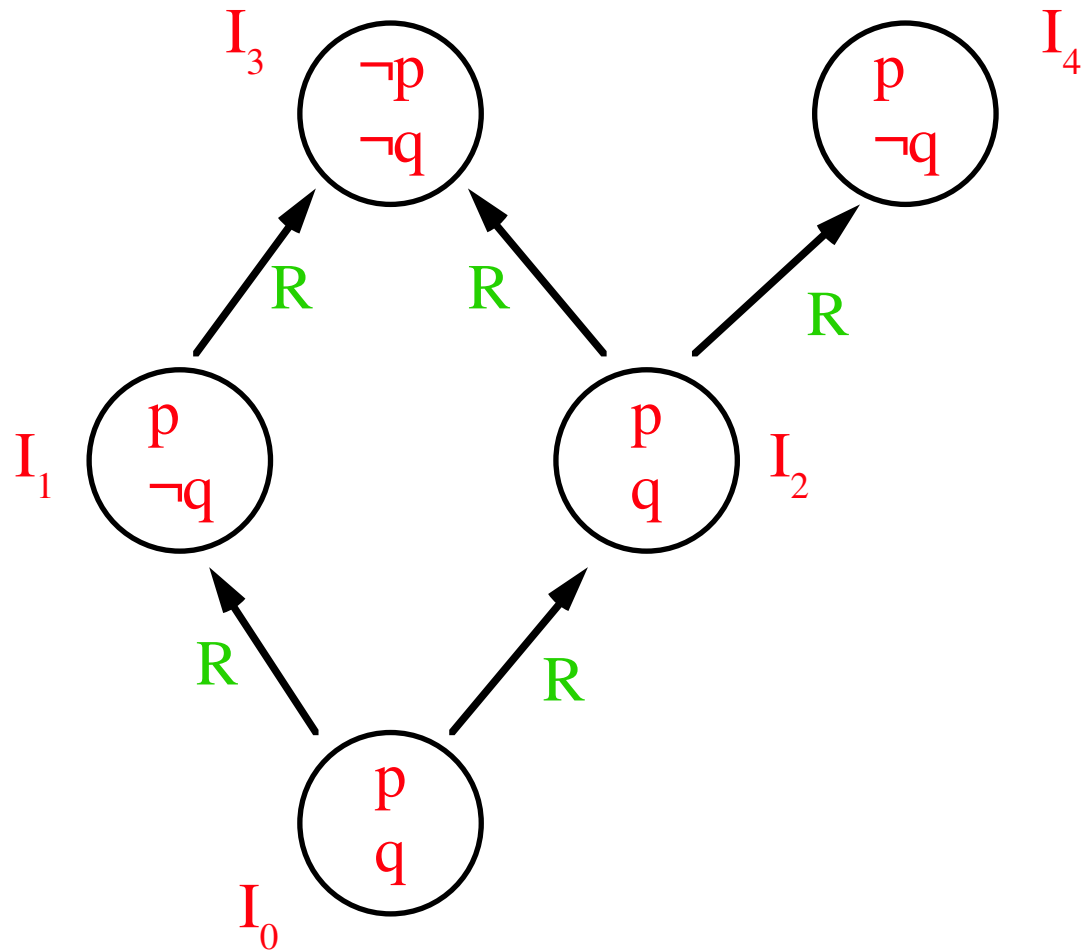
tree automata-based

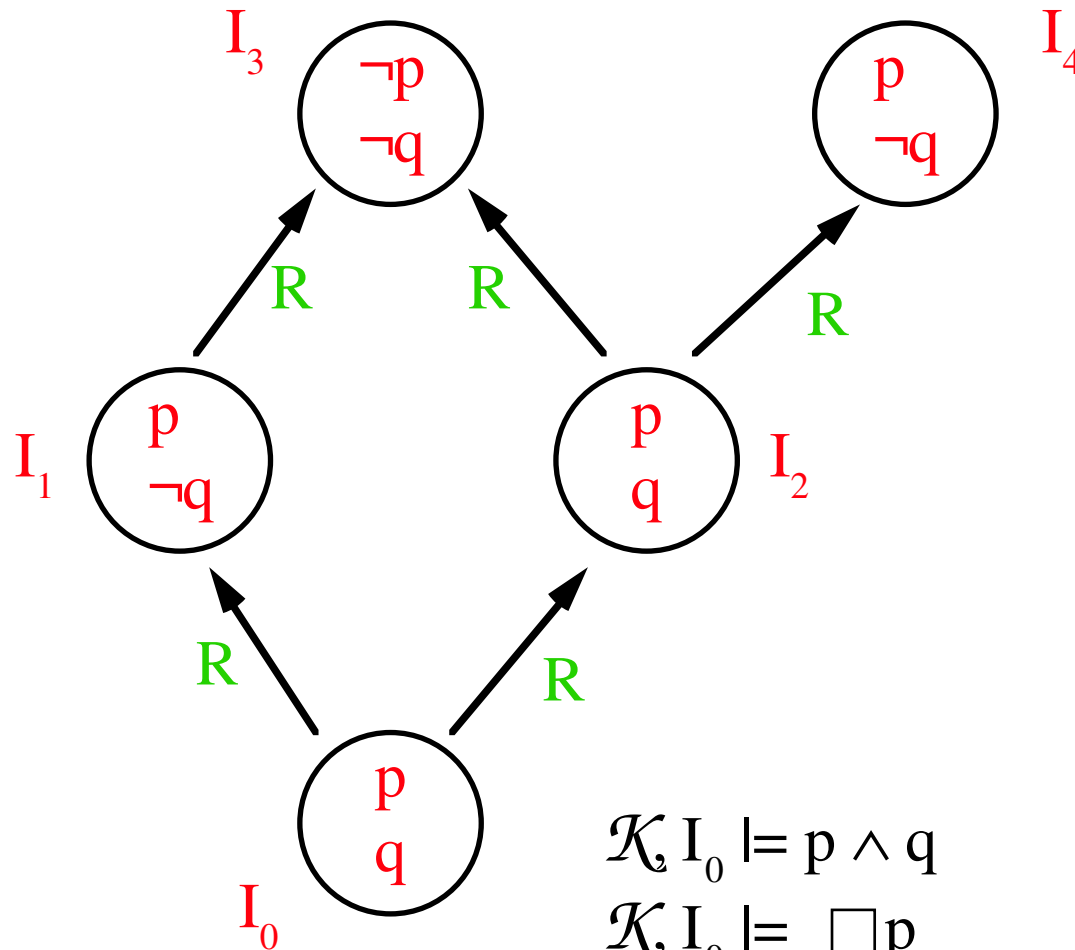
- implementations?
- best-case exponential
- + easy to obtain exact worst-case complexity upper-bound:
 - + "natural" approach yields ExpTime algorithm

Approach that combines the advantages of both?

The basic modal logic K

- Extends propositional logic by a pair of unary modal operators box \Box and diamond \Diamond .
- Semantics is defined via Kripke structures, i.e., sets of propositional interpretations linked by an accessibility relation.
 - box: $\Box G$ means that G holds in all accessible worlds.
 - diamond: $\Diamond G$ means that G holds in some accessible worlds.





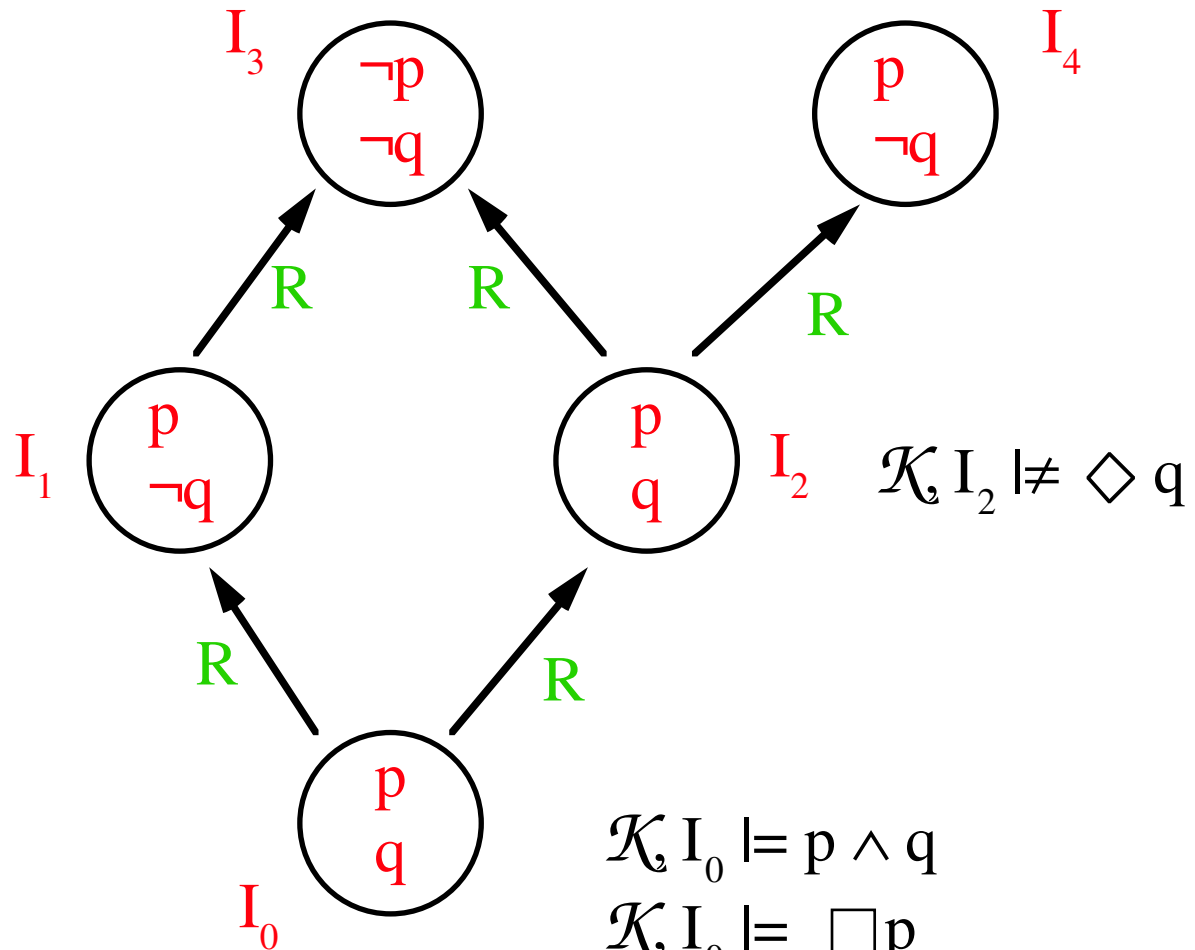
$$\mathcal{K} I_0 \models p \wedge q$$

$$\mathcal{K} I_0 \models \Box p$$

$$\mathcal{K} I_0 \models \Diamond q \text{ and } \mathcal{K} I_0 \models \Diamond \neg q$$

$$\mathcal{K} I_0 \not\models \Box q$$

$$\mathcal{K} I_0 \models \Diamond \Box (\neg p \wedge \neg q)$$



$$\mathcal{K} I_0 \models p \wedge q$$

$$\mathcal{K} I_0 \models \Box p$$

$$\mathcal{K} I_0 \models \Diamond q \text{ and } \mathcal{K} I_0 \models \Diamond \neg q$$

$$\mathcal{K} I_0 \not\models \Box q$$

$$\mathcal{K} I_0 \models \Diamond \Box (\neg p \wedge \neg q)$$

The basic modal logic K

- Extends propositional logic by a pair of unary modal operators box \Box and diamond \Diamond .
- Semantics is defined via Kripke structures, i.e., sets of propositional interpretations linked by an accessibility relation.
 - ➔ box: $\Box G$ means that G holds in all accessible worlds.
 - ➔ diamond: $\Diamond G$ means that G holds in some accessible worlds.

The basic modal logic K

- Extends propositional logic by a pair of unary modal operators box \Box and diamond \Diamond .
- Semantics is defined via Kripke structures, i.e., sets of propositional interpretations linked by an accessibility relation.
 - ➔ box: $\Box G$ means that G holds in all accessible worlds.
 - ➔ diamond: $\Diamond G$ means that G holds in some accessible worlds.

The modal formula G
is satisfiable

iff

G is satisfiable in some world
of some Kripke structure

The basic modal logic K

- Extends propositional logic by a pair of unary modal operators box \Box and diamond \Diamond .
- Semantics is defined via Kripke structures, i.e., sets of propositional interpretations linked by an accessibility relation.
 - ➔ box: $\Box G$ means that G holds in all accessible worlds.
 - ➔ diamond: $\Diamond G$ means that G holds in some accessible worlds.

The modal formula G
is satisfiable

iff

G is satisfiable in some world
of some Kripke structure

finite tree structures suffice

The basic modal logic K

- Extends propositional logic by a pair of unary modal operators box \Box and diamond \Diamond .
- Semantics is defined via Kripke structures, i.e., sets of propositional interpretations linked by an accessibility relation.
 - ➔ box: $\Box G$ means that G holds in all accessible worlds.
 - ➔ diamond: $\Diamond G$ means that G holds in some accessible worlds.

The modal formula G
is satisfiable

iff

G is satisfiable in some world
of some Kripke structure

finite tree structures suffice

- Satisfiability in K is PSpace-complete.

Tableau approach

for K without global axioms

- Tries to generate a finite tree structure satisfying G (where G is without loss of generality in NNF).
- Generates an initial world labeled with G, and
- then applies tableau rules:
 - propositional rules expand the label of the given world; rule for disjunction is nondeterministic.
 - diamond rule generates new accessible worlds
 - box rule extends the label of accessible worlds
 - clash rule detects obvious contradictions (p and $\neg p$ for propositional variable p)

Tableau approach

Example:

Satisfiability of $\diamond p \wedge \Box(\neg p \vee q)$



$\diamond p \wedge \Box(\neg p \vee q)$

Tableau approach

Example:

Satisfiability of $\diamond p \wedge \Box(\neg p \vee q)$



$\diamond p \wedge \Box(\neg p \vee q)$

$\diamond p, \Box(\neg p \vee q)$

Tableau approach

Example:

Satisfiability of $\diamond p \wedge \Box(\neg p \vee q)$

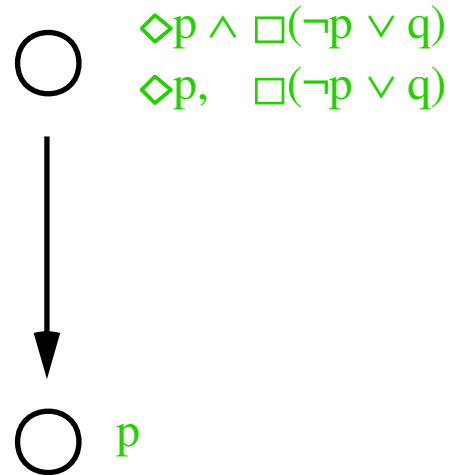


Tableau approach

Example:

Satisfiability of $\diamond p \wedge \Box(\neg p \vee q)$

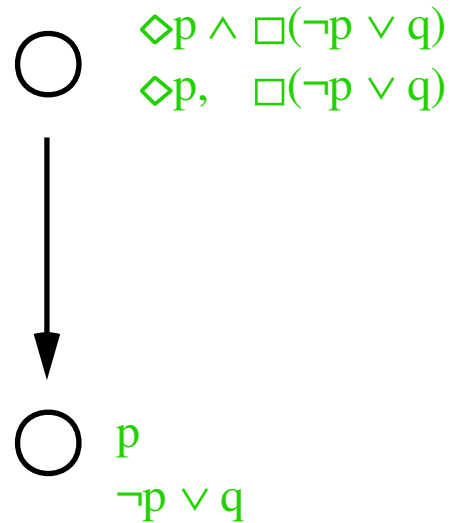


Tableau approach

Example:

Satisfiability of $\diamond p \wedge \Box(\neg p \vee q)$

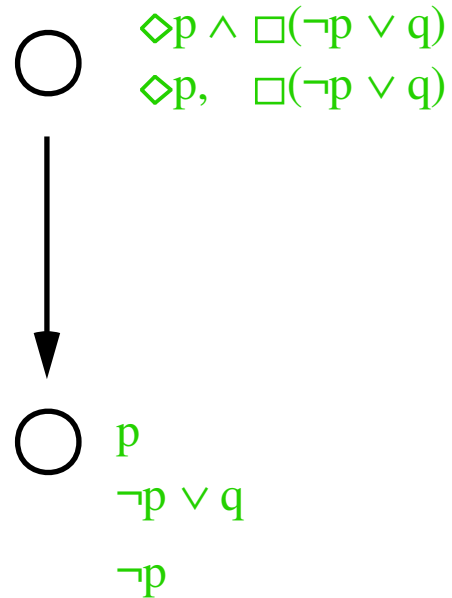
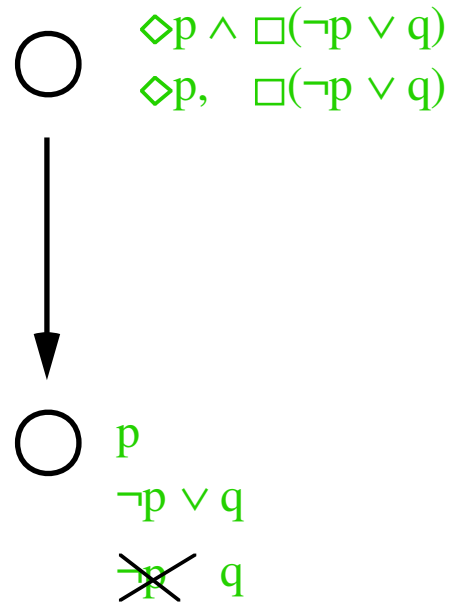


Tableau approach

Example:

Satisfiability of $\diamond p \wedge \Box(\neg p \vee q)$



Satisfiability in K w.r.t. global axioms

ExpTime-complete

[Spaan 93]

G is satisfiable w.r.t.
the global axiom H

iff

G is satisfiable in a Kripke structure
in which all worlds satisfy H

Satisfiability in K w.r.t. global axioms

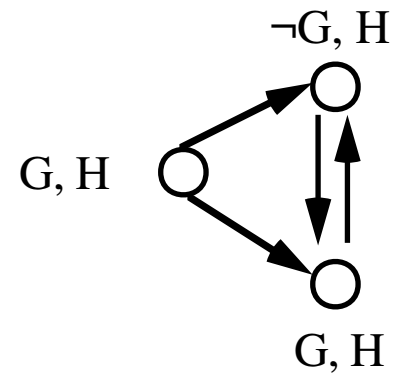
ExpTime-complete

[Spaan 93]

G is satisfiable w.r.t.
the global axiom H

iff

G is satisfiable in a Kripke structure
in which all worlds satisfy H



Satisfiability in K w.r.t. global axioms

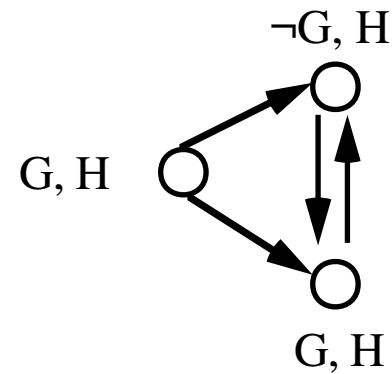
ExpTime-complete

[Spaan 93]

G is satisfiable w.r.t.
the global axiom H

iff

G is satisfiable in a Kripke structure
in which all worlds satisfy H



finite structures suffice

Satisfiability in K w.r.t. global axioms

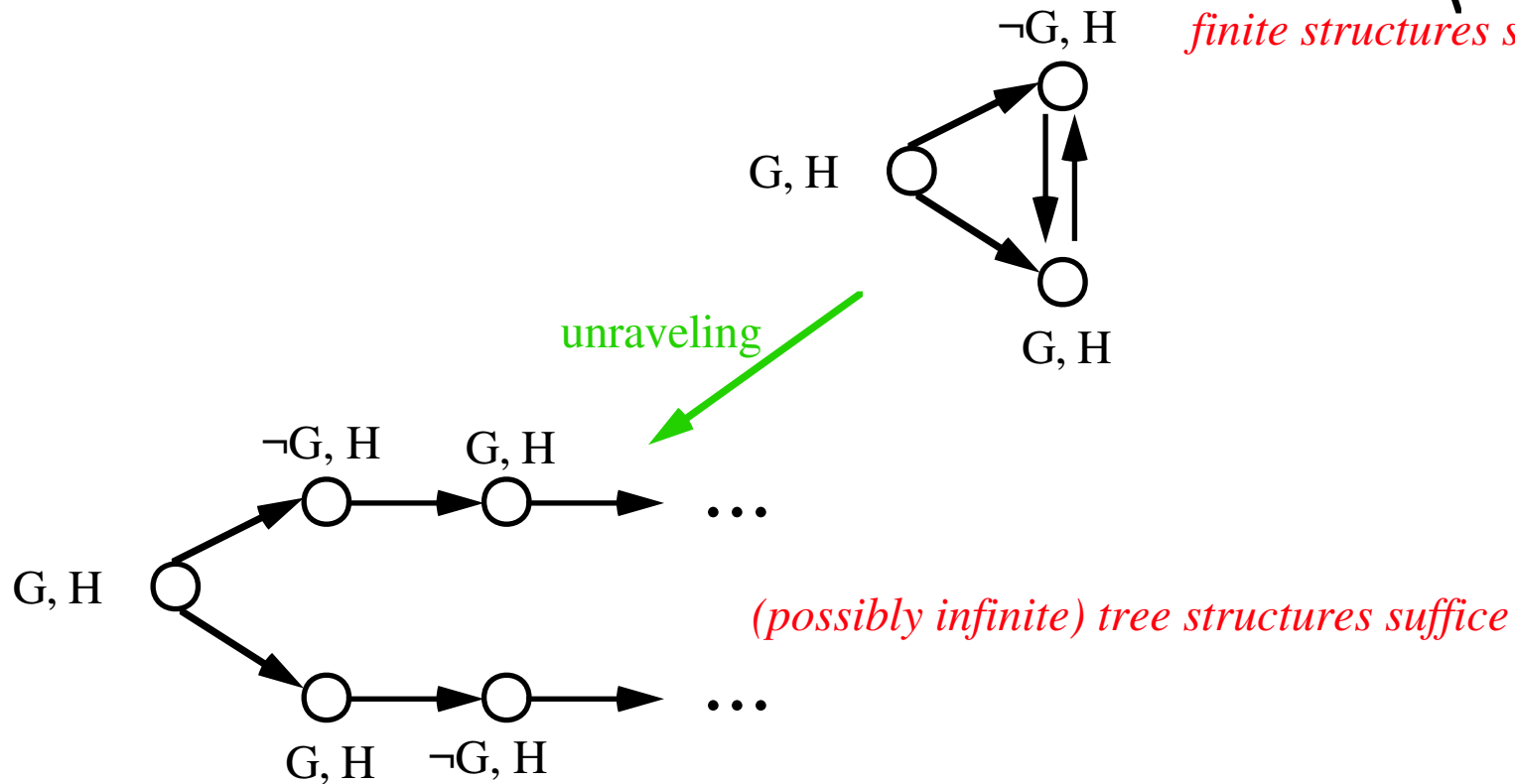
ExpTime-complete

[Spaan 93]

G is satisfiable w.r.t.
the global axiom H

iff

G is satisfiable in a Kripke structure
in which all worlds satisfy H



Satisfiability in K w.r.t. global axioms

ExpTime-complete
[Spaan 93]

G is satisfiable w.r.t.
the global axiom H

iff

G is satisfiable in a Kripke structure
in which all worlds satisfy H

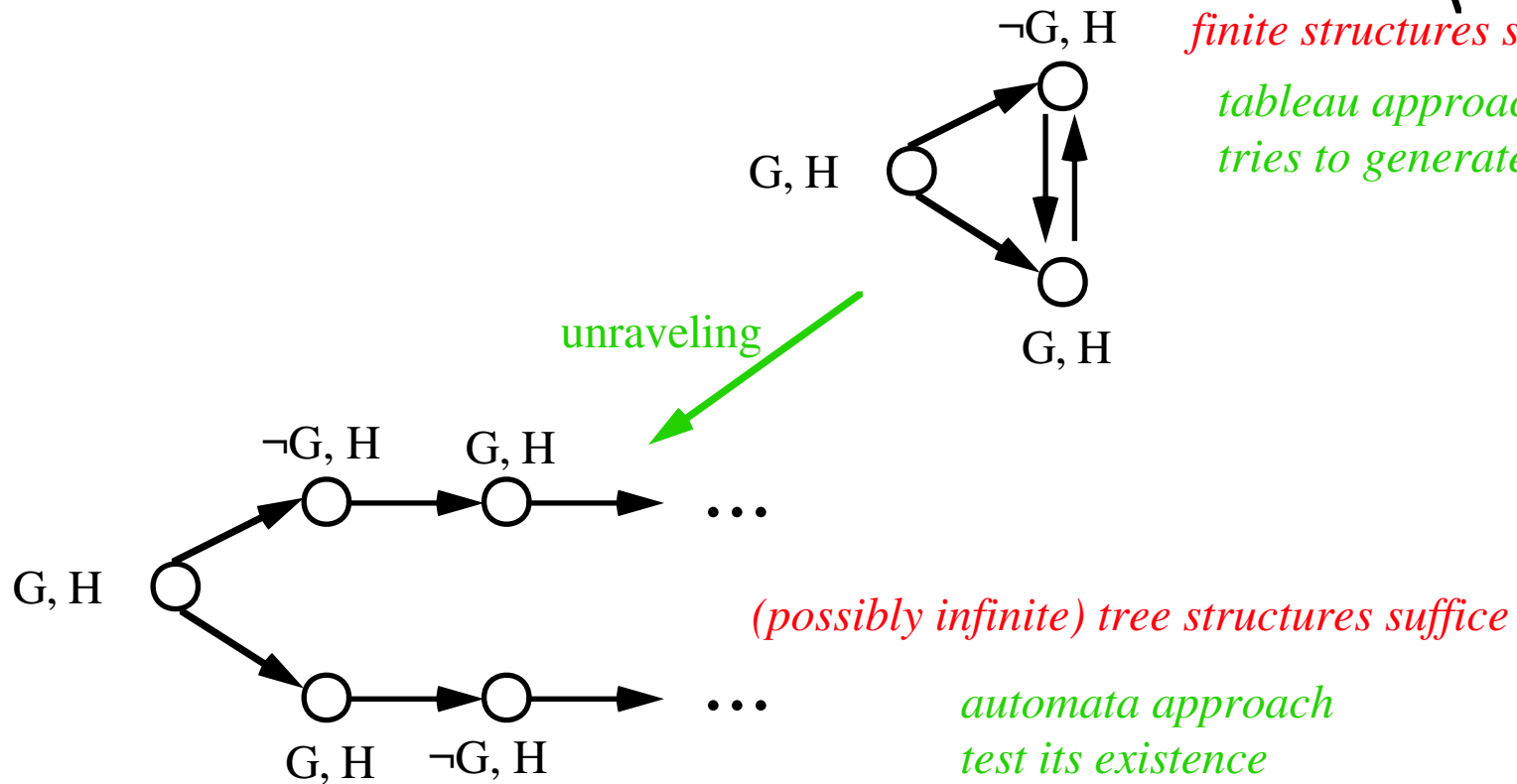


Tableau-approach

for K with global axioms

- Tries to generate a finite structure satisfying G w.r.t. H .
- The axiom H is added to every world generated by the algorithm.
- Blocking required to ensure termination (cyclic structures).

$$G = A \wedge \Box B$$

$$H = \Diamond A$$

Tableau-approach

for K with global axioms

- Tries to generate a finite structure satisfying G w.r.t. H .
- The axiom H is added to every world generated by the algorithm.
- Blocking required to ensure termination (cyclic structures).

$$G = A \wedge \Box B$$

G, H



$$H = \Diamond A$$

Tableau-approach

for K with global axioms

- Tries to generate a finite structure satisfying G w.r.t. H .
- The axiom H is added to every world generated by the algorithm.
- Blocking required to ensure termination (cyclic structures).

$$G = A \wedge \Box B$$

$$H = \Diamond A$$

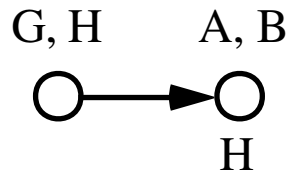


Tableau-approach

for K with global axioms

- Tries to generate a finite structure satisfying G w.r.t. H .
- The axiom H is added to every world generated by the algorithm.
- Blocking required to ensure termination (cyclic structures).

$$G = A \wedge \Box B$$

$$H = \Diamond A$$

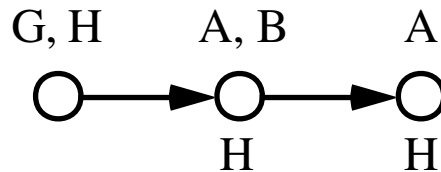


Tableau-approach

for K with global axioms

- Tries to generate a finite structure satisfying G w.r.t. H .
- The axiom H is added to every world generated by the algorithm.
- Blocking required to ensure termination (cyclic structures).

$$G = A \wedge \Box B$$

$$H = \Diamond A$$

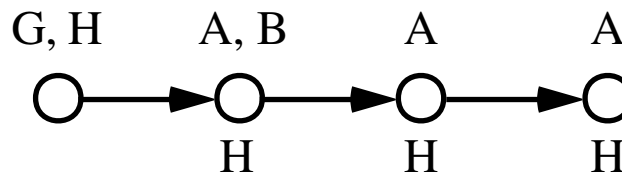


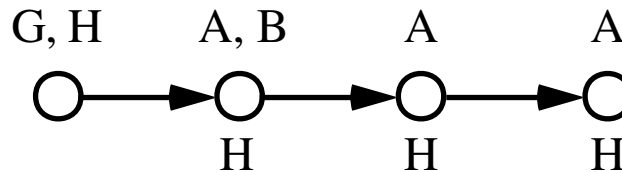
Tableau-approach

for K with global axioms

- Tries to generate a finite structure satisfying G w.r.t. H .
- The axiom H is added to every world generated by the algorithm.
- Blocking required to ensure termination (cyclic structures).

$$G = A \wedge \Box B$$

$$H = \Diamond A$$



- Length of paths: may become exponential before blocking occurs.
- Nondeterminism: treatment of disjunction.

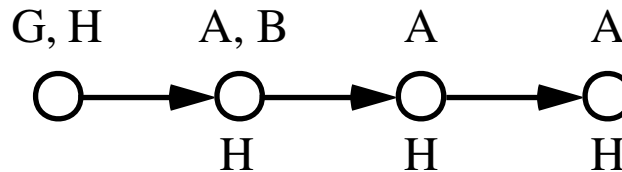
Tableau-approach

for K with global axioms

- Tries to generate a finite structure satisfying G w.r.t. H .
- The axiom H is added to every world generated by the algorithm.
- Blocking required to ensure termination (cyclic structures).

$$G = A \wedge \Box B$$

$$H = \Diamond A$$



- Length of paths: may become exponential before blocking occurs.
- Nondeterminism: treatment of disjunction.

*NExpTime
complexity*

Automata approach

reduction to the emptiness problem for automata on infinite trees

- Tests for the existence of an (infinite) tree structure satisfying G w.r.t. H .
- States of the automaton: propositionally expanded sets of subformulae of G and H that contain H .
- Initial state: contains G .

Automata approach

reduction to the emptiness problem for automata on infinite trees

- Tests for the existence of an (infinite) tree structure satisfying G w.r.t. H .
- States of the automaton: propositionally expanded sets of subformulae of G and H that contain H .
- Initial state: contains G .
- Transitions: look for the existence of appropriate sons if they are required by diamond formulae (otherwise: "dummy" sons).
No transition from states containing a clash.
- Looping tree automaton: accepts if there is an infinite run.

Automata approach

reduction to the emptiness problem for automata on infinite trees

- Tests for the existence of an (infinite) tree structure satisfying G w.r.t. H .
- States of the automaton: propositionally expanded sets of subformulae of G and H that contain H .
- Initial state: contains G .
- Transitions: look for the existence of appropriate sons if they are required by diamond formulae (otherwise: "dummy" sons).
No transition from states containing a clash.
- Looping tree automaton: accepts if there is an infinite run.
- Run looks like an infinite tableau (no blocking required).
- Automaton is nondeterministic due to presence of disjunction.

Automata approach

Example:

$$H = \diamond p \quad \text{and} \quad G = \square(\neg p \vee q)$$

Automata approach

Example:

$H = \diamond p$ and $G = \square(\neg p \vee q)$

successful run on



Automata approach

Example:

$$H = \diamond p \quad \text{and} \quad G = \square(\neg p \vee q)$$

successful run on



$$\diamond p, \square(\neg p \vee q), \neg p \vee q, \neg p$$

Automata approach

Example:

$$H = \diamond p \quad \text{and} \quad G = \square(\neg p \vee q)$$

successful run on



$\diamond p, \square(\neg p \vee q), \neg p \vee q, \neg p$



$\diamond p, p, \neg p \vee q, q$



⋮

Automata approach

Example:

$H = \diamond p$ and $G = \square(\neg p \vee q)$

successful run on

○ $\diamond p, \square(\neg p \vee q), \neg p \vee q, \neg p$



○ $\diamond p, p, \neg p \vee q, q$



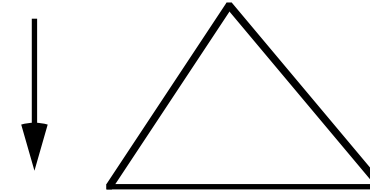
○ $\diamond p, p, q$



⋮

Emptiness test

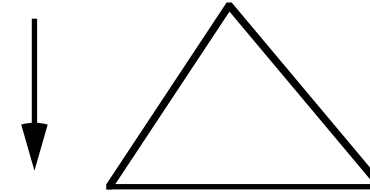
naive top-down approach



- Tries to **construct** an infinite tree and an **infinite run** on this tree.
- Starts with an **initial state** at the root, and then generates son nodes labeled according to the **transition function**.
- Looks for **state repetition** on paths to ensure **termination**.
- Very **similar to tableau**-approach with blocking.
- **Complexity: NP** in size of automaton if the **automaton is nondeterministic**.

Emptiness test

naive top-down approach

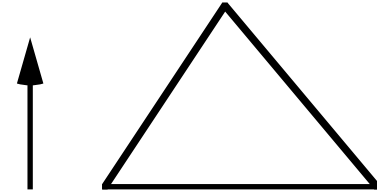


- Tries to **construct** an infinite tree and an **infinite run** on this tree.
- Starts with an **initial state** at the root, and then generates son nodes labeled according to the **transition function**.
- Looks for **state repetition** on paths to ensure **termination**.
- Very **similar to tableau**-approach with blocking.
- **Complexity: NP** in size of automaton if the **automaton is nondeterministic**.

*Since the constructed automaton is exponential in the size of the formula, this still leaves us with a **NExpTime** procedure.*

Emptiness test

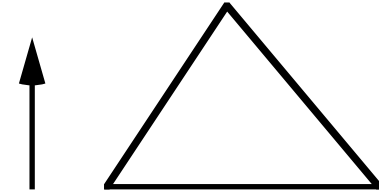
improved bottom-up approach
(dynamic programming)



- Computes **inactive states**, i.e., states that cannot occur on an infinite run of the automaton:
 - ➔ Starts with **obviously inactive states**, i.e., states that do not have successors states w.r.t. the transition function.
 - ➔ **Propagates inactiveness** along the transition function.

Emptiness test

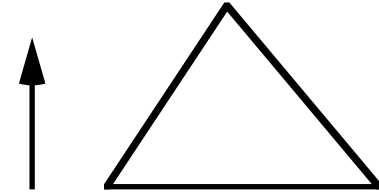
improved bottom-up approach
(dynamic programming)



- Computes **inactive states**, i.e., states that cannot occur on an infinite run of the automaton:
 - ➔ Starts with **obviously inactive states**, i.e., states that do not have successors states w.r.t. the transition function.
 - ➔ **Propagates inactiveness** along the transition function.
- Naive implementation already **polynomial**.
- Using appropriate data structures, the set of inactive states can be computed in **linear time**.

Emptiness test

improved bottom-up approach
(dynamic programming)



- Computes **inactive states**, i.e., states that cannot occur on an infinite run of the automaton:
 - ➔ Starts with **obviously inactive states**, i.e., states that do not have successors states w.r.t. the transition function.
 - ➔ **Propagates inactiveness** along the transition function.
- Naive implementation already **polynomial**.
- Using appropriate data structures, the set of inactive states can be computed in **linear time**.

*Since the constructed automaton is exponential in the size of the formula, this provides us with an **ExpTime** procedure.*

Emptiness test

bottom-up approach in more detail
(naive implementation)

Obviously inactive states:

$$Q_0 := \{q \mid q \text{ is state s.t. there is not transition } (q, \cdot) \rightarrow (\dots)\}$$

Propagation of inactiveness:

$$Q \succ Q \cup \{q\} \text{ iff all transitions } (q, \cdot) \rightarrow (q_1, \dots, q_n) \text{ are such that some } q_i \in Q$$

Set of inactive states:

$$Q_0^{\succ} := \bigcup \{Q \mid Q_0 \succ^* Q\} \quad (\text{propagation closure})$$

Emptiness test:

$$L(\mathcal{A}) = \emptyset \text{ iff all initial states belong to } Q_0^{\succ}$$

Emptiness test

bottom-up approach in more detail
(naive implementation)

Obviously inactive states:

*states containing $p, \neg p$
in constructed automaton*

$$Q_0 := \{q \mid q \text{ is state s.t. there is not transition } (q, \cdot) \rightarrow (\dots)\}$$

Propagation of inactiveness:

$$Q \succcurlyeq Q \cup \{q\} \text{ iff all transitions } (q, \cdot) \rightarrow (q_1, \dots, q_n) \text{ are such that some } q_i \in Q$$

Set of inactive states:

$$Q_0^{\succcurlyeq} := \bigcup \{Q \mid Q_0 \succcurlyeq^* Q\} \quad (\text{propagation closure})$$

Emptiness test:

$$L(\mathcal{A}) = \emptyset \text{ iff all initial states belong to } Q_0^{\succcurlyeq}$$

Emptiness test

bottom-up approach in more detail
(naive implementation)

Obviously inactive states:

*states containing $p, \neg p$
in constructed automaton*

$$Q_0 := \{q \mid q \text{ is state s.t. there is not transition } (q, \cdot) \rightarrow (\dots)\}$$

Propagation of inactiveness:

$$Q \succcurlyeq Q \cup \{q\} \text{ iff all transitions } (q, \cdot) \rightarrow (q_1, \dots, q_n) \text{ are such that some } q_i \in Q$$

Set of inactive states:

$$Q_0^{\succcurlyeq} := \bigcup \{Q \mid Q_0 \succcurlyeq^* Q\} \quad (\text{propagation closure})$$

*states containing the
the formula G in
constructed automaton*

Emptiness test:

$$L(\mathcal{A}) = \emptyset \text{ iff all initial states belong to } Q_0^{\succcurlyeq}$$

The inverse calculus

for modal K [Voronkov 01]
generates unsatisfiable formulae

Rules

$\frac{\Gamma, A}{\Gamma, A \wedge B}$	$\frac{\Gamma, B}{\Gamma, A \wedge B}$	$\frac{\Gamma, A \mid \Delta, B}{\Gamma, \Delta, A \vee B}$
$\frac{\Gamma, A}{\Box \Gamma, \Diamond A}$	$\frac{\Gamma}{\Box \Gamma, \Diamond A}$	

Axioms

$\{p, \neg p\}$
for propositional
variables p

The inverse calculus

for modal K [Voronkov 01]
generates unsatisfiable formulae

Rules

$\frac{\Gamma, A}{\Gamma, A \wedge B}$	$\frac{\Gamma, B}{\Gamma, A \wedge B}$	$\frac{\Gamma, A \mid \Delta, B}{\Gamma, \Delta, A \vee B}$
$\frac{\Gamma, A}{\Box \Gamma, \Diamond A}$	$\frac{\Gamma}{\Box \Gamma, \Diamond A}$	

Axioms

$\{p, \neg p\}$
for propositional
variables p

To test for satisfiability of the formula G ,
restrict rules and axioms to subformulae of G .

The inverse calculus

for modal K [Voronkov 01]
generates unsatisfiable formulae

Rules

$\frac{\Gamma, A}{\Gamma, A \wedge B}$	$\frac{\Gamma, B}{\Gamma, A \wedge B}$	$\frac{\Gamma, A \mid \Delta, B}{\Gamma, \Delta, A \vee B}$
$\frac{\Gamma, A}{\Box \Gamma, \Diamond A}$	$\frac{\Gamma}{\Box \Gamma, \Diamond A}$	

Axioms

$\{p, \neg p\}$
for propositional
variables p

To test for satisfiability of the formula G ,
restrict rules and axioms to subformulae of G .

Satisfiability test:

$\mathcal{S}_0 := \{\Gamma \mid \Gamma \text{ is axiom}\}$ and $\mathcal{S}_0^{\vdash} := \cup\{\mathcal{S} \mid \mathcal{S}_0 \vdash^* \mathcal{S}\}$ (inference closure)

G unsatisfiable iff $\{G\} \in \mathcal{S}_0^{\vdash}$

The inverse calculus

for modal K [Voronkov 01] with global axioms

Rules

$\frac{\Gamma, A}{\Gamma, A \wedge B}$	$\frac{\Gamma, B}{\Gamma, A \wedge B}$	$\frac{\Gamma, A \mid \Delta, B}{\Gamma, \Delta, A \vee B}$
$\frac{\Gamma, A}{\Box \Gamma, \Diamond A}$	$\frac{\Gamma}{\Box \Gamma, \Diamond A}$	$\frac{\Gamma, H}{\Gamma}$

Axioms

{p, ¬p}
for propositional variables p

To test for satisfiability of the formula G w.r.t. H,
restrict rules and axioms to subformulae of G and H.

Satisfiability test:

$\mathcal{S}_0 := \{\Gamma \mid \Gamma \text{ is axiom}\}$ and $\mathcal{S}_0^{\perp} := \cup\{\mathcal{S} \mid \mathcal{S}_0 \vdash^* \mathcal{S}\}$ (inference closure)

G unsatisfiable w.r.t. H iff $\{G\} \in \mathcal{S}_0^{\perp}$ or $\emptyset \in \mathcal{S}_0^{\perp}$

Connecting the two approaches

main technical result

$|\Gamma| := \{q \mid q \text{ is state containing } \Gamma\}$ and $|\mathcal{S}| := \bigcup\{|\Gamma| \mid \Gamma \text{ in } \mathcal{S}\}$

Theorem

The automata approach and the inverse method can simulate each other:

- If $Q_0 \succ^* Q$, then there exists a set of sequents \mathcal{S} such that $\mathcal{S}_0 \vdash^* \mathcal{S}$ and $Q \subseteq |\mathcal{S}|$.
- If $\mathcal{S}_0 \vdash^* \mathcal{S}$, then there exists a set of states Q such that $Q_0 \succ^* Q$ and $|\mathcal{S}| \subseteq Q$.

Consequences

of the theorem

- The propagation closure and the inference closure "agree":

$$Q_0^{\succ} = \llbracket \mathcal{S}_0^{\perp} \rrbracket$$

- ➔ The inverse calculus yields an "on-the-fly" implementation of the emptiness test for the constructed automaton.
- ➔ One sequent represents several states (states containing this sequent).

Consequences

of the theorem

- The propagation closure and the inference closure "agree":

$$Q_0^{\succ} = \llbracket \mathcal{S}_0^{\perp} \rrbracket$$

- ➔ The inverse calculus yields an "on-the-fly" implementation of the emptiness test for the constructed automaton.
 - ➔ One sequent represents several states (states containing this sequent).
- The inverse calculus yields an ExpTime decision procedure for satisfiability w.r.t. global axioms in K.
 - ➔ This "on-the-fly" implementation of the emptiness test yields an a procedure that is optimal w.r.t. worst-case complexity.

Further results

concerning optimizations of the procedure

Voronkov introduces **optimizations** of the inverse calculus for K without global axioms:

- ➔ **redundant sequents** (corresponding to unreachable states)
- ➔ **redundant inferences** avoided by imposing an ordering-restriction on the application of the diamond inference rules.

Completeness of both optimizations can be shown **within the automata framework** (which yields **simpler proofs**).

Future work

- Can the inverse method be used to obtain a PSPACE-algorithm for satisfiability in K w/o global axioms?
- Can Voronkov's optimizations be adapted to the calculus dealing with global axioms?
- Can additional optimizations considered by Voronkov (eg. prefix subsumption) also be justified using the automata approach?
- Can the results be transferred to other modal/description logics?

Future work

- Can the inverse method be used to obtain a PSPACE-algorithm for satisfiability in K w/o global axioms?
- Can Voronkov's optimizations be adapted to the calculus dealing with global axioms?
- Can additional optimizations considered by Voronkov (eg. prefix subsumption) also be justified using the automata approach?
- Can the results be transferred to other modal/description logics?
- What can we say about alternating automata?
How can we get a "direct" algorithm that tests emptiness for alternating (looping) tree automata?