Modified tableaux for some kinds of multi-modal logics

Emilio Gómez-Caminero, Ángel Nepomuceno
Grupo de Lógica, Lenguaje e Información
Universidad de Sevilla

Lisbon, 2012

Labelled Tableaux

The basic case

- It is well known that we can use a labelled tableaux method like a decision procedure in modal logic.

Labelled Tableaux

The basic case

- It is well known that we can use a labelled tableaux method like a decision procedure in modal logic.
- We need a system of labels of the form 1, 1.1, 1.2, 1.2.1, and so on.

Labelled Tableaux

The basic case

- It is well known that we can use a labelled tableaux method like a decision procedure in modal logic.
- We need a system of labels of the form 1, 1.1, 1.2, 1.2.1, and so on.
- The recursive definition is:
    - a. 1 is a label.
    - b. If σ is a label, then $\sigma.n$ is a label (for $n \geqslant 1$)

Labelled Tableaux

The basic case

- It is well known that we can use a labelled tableaux method like a decision procedure in modal logic.
- We need a system of labels of the form 1, 1.1, 1.2, 1.2.1, and so on.
- The recursive definition is:
  - a. 1 is a label.
  - b. If σ is a label, then $\sigma.n$ is a label (for $n \geqslant 1$)
- Intuitively, each label represents a possible world, such that $\sigma.n$ is reachable from σ.

The basic case: rules

- The rules for the propositional operators are the usual, whit a label which remains invariable. E.g.:

$$\frac{\sigma :: \alpha \wedge \beta}{\begin{array}{c}\sigma :: \alpha \\ \sigma :: \beta\end{array}}$$

The basic case: rules

- The rules for the propositional operators are the usual, whit a label which remains invariable. E.g.:

$$\frac{\sigma :: \alpha \wedge \beta}{\begin{array}{c}\sigma :: \alpha \\ \sigma :: \beta\end{array}}$$

- The rule for the operator $\Diamond$ is the only one which creates a new label:

$$\mathbf{R}\Diamond : \frac{\sigma :: \Diamond\alpha}{\sigma.n :: \alpha}$$

(Where n is the first positive integer such that $\sigma.n$ is new in the branch.)

The problem of infinity

- The preceding rule, in addition whit the rules for the operator $\Box$ may give rise to an infinite branch.

The problem of infinity

- The preceding rule, in addition whit the rules for the operator $\Box$ may give rise to an infinite branch.
- But we can avoid it using the following restriction:

> *Except if* $\tau :: \alpha$ *appears in the branch and* σ *is reachable since* τ*. In that case, the rule is considered as applied and the formula is marked.*

The operator $\Box$

- The rule for the operator $\Box$ depends on one of the the properties of the accessibility relation:

The operator $\Box$

- The rule for the operator $\Box$ depends on one of the the properties of the accessibility relation:
- If the accessibility relation is reflexive, the rule is:

$$\textbf{R}\Box\ \textbf{refl.} : \frac{\sigma :: \Box\alpha}{\sigma :: \alpha}$$

The operator $\Box$

- The rule for the operator $\Box$ depends on one of the the properties of the accessibility relation:
- If the accessibility relation is reflexive, the rule is:

$$\mathbf{R\Box\ refl.}: \frac{\sigma :: \Box\alpha}{\sigma :: \alpha}$$

- If the accessibility relation is serial, the rule is:

$$\mathbf{R\Box\ ser.}: \frac{\sigma :: \Box\alpha}{\sigma :: \Diamond\alpha}$$

Inheritance rules

- The specific character of the operator $\Box$ is captured by the so called *inheritance rules*:

Inheritance rules

- The specific character of the operator $\Box$ is captured by the so called *inheritance rules*:
- Basic case (the accessibility relation has no more properties than reflexivity or seriality):

$$\textbf{IRT} : \frac{\sigma :: \Box\alpha}{\sigma.n :: \alpha}$$

(For any label $\sigma.n$ which appears in the branch.)

Symmetry and transitivity

We can introduce additional properties replacing the preceding rule whit the followings:

- Euclidianity (S4, KD4):

$$\textbf{IRS4} : \frac{\sigma :: \Box\alpha}{\sigma.n :: \Box\alpha}$$

Symmetry and transitivity

We can introduce additional properties replacing the preceding rule whit the followings:

- Euclidianity (S4, KD4):

$$\textbf{IRS4}: \frac{\sigma :: \Box\alpha}{\sigma.n :: \Box\alpha}$$

- Symmetry (S5, KD45):

$$\textbf{IRS5}: \frac{\sigma :: \Box\alpha}{\sigma.n :: \Box\alpha}$$

Properties

- With the adequate combination of these rules we can create tableau methods for all basic systems of alethic modal logic.
- We can prove that these methods are correct and complete.
- In many cases, we can extend this kind of labelled tableau to systems of multi-modal logic.

Multi-agent systems

- The easier extension of modal logic is the case in which we have a certain number of modal operators of the same kind.

Multi-agent systems

- The easier extension of modal logic is the case in which we have a certain number of modal operators of the same kind.
- This is the case of multi-agent modal logics. The best known are epistemic and doxastic logic.

Multi-agent systems

- The easier extension of modal logic is the case in which we have a certain number of modal operators of the same kind.
- This is the case of multi-agent modal logics. The best known are epistemic and doxastic logic.
- Given a set $\mathcal{A}$ of agents, we have an operator $\Box_{a_i}$ and its dual $\Diamond_{a_i}$ for each agent $a_i \in \mathcal{A}$.

Multi-agent systems

- The easier extension of modal logic is the case in which we have a certain number of modal operators of the same kind.
- This is the case of multi-agent modal logics. The best known are epistemic and doxastic logic.
- Given a set $\mathcal{A}$ of agents, we have an operator $\Box_{a_i}$ and its dual $\Diamond_{a_i}$ for each agent $a_i \in \mathcal{A}$.
- Usually we write K_{a_i} when we are talking about epistemic logic and B_{a_i} when we are dealing with doxastic logic.

Multi-agent systems, labels

To adapt the tableau method to multi-agent modal logic we only have to adapt the form of the labels:

labels

Given a set $\mathcal{A}$ of agents:

- a 1 is a label.
- b If σ is a label, then $\sigma.a_i n$ is a label too (for $n \geq 1$ and $a_i \in \mathcal{A}$).

Rules

The rules are the same that in the general case, but adapted to the new labels:

- $\Diamond$-Rule (**R**$\Diamond$):

$$\frac{\sigma :: \Diamond_{a_i}\alpha}{\sigma.a_i n :: \alpha}$$

- $\Box$-Rule (**R**$\Box$):

Knowledge

$$\frac{\sigma :: \Box_{a_i}\alpha}{\sigma :: \alpha}$$

Belief

$$\frac{\sigma :: \Box_{a_i}\alpha}{\sigma :: \Diamond_{a_i}\alpha}$$

Inheritance rules

T_m/KD_m: $\dfrac{\sigma :: \Box_{a_i}\alpha}{\sigma.a_i n :: \alpha}$

$S4_m/KD4_m$: $\dfrac{\sigma :: \Box_{a_i}\alpha}{\sigma.a_i n :: \Box\alpha}$

$S5_m/KD5_m$: $\dfrac{\sigma :: \Box_{a_i}\alpha}{\dfrac{}{\sigma.a_i n :: \Box\alpha}}$

Combining modalities

- We may combine different kinds of modalities; for example, epistemic and doxastic modalities, epistemic and deontic modalities, and son on.

Combining modalities

- We may combine different kinds of modalities; for example, epistemic and doxastic modalities, epistemic and deontic modalities, and son on.
- In order to do it, we have to consider different kinds of accessibility relations and represent them using the labels.

Combining modalities

- We may combine different kinds of modalities; for example, epistemic and doxastic modalities, epistemic and deontic modalities, and son on.
- In order to do it, we have to consider different kinds of accessibility relations and represent them using the labels.
- For example, if we want to combine epistemic and doxastic operators, rules are:

Knowledge

$$\frac{\sigma :: \widehat{K}_{a_i}\alpha}{\sigma.Ea_in :: \alpha}$$

(where n is new in the branch)

Belief

$$\frac{\sigma :: \widehat{B}_{a_i}\alpha}{\sigma Da_in :: \Diamond_{a_i}\alpha}$$

(where n is new in the branch)

$\sigma.Ea_in$ represents an epistemic alternative to σ. σDa_in represents a doxastic alternative to σ.

Combining modalities. Inheritance rules.

- We have to consider the relations between different kinds of modality.

Combining modalities. Inheritance rules.

- We have to consider the relations between different kinds of modality.
- Depending on these relations, we have to change the inheritance rules. For example, if we accept that $K_{a_i}\varphi \rightarrow B_{a_i}\varphi$, we have to modify the inheritance rule for K (we give the example for S4):

  $$\frac{\sigma :: K_{a_i}\alpha}{\sigma.Xa_in :: \alpha}$$

  (Where X may be E or D)

Group knowledge

- We can extend our multi-agent epistemic/doxastic logic (¿and deontic,...?) with group knowledge (group belief) operators.

Group knowledge

- We can extend our multi-agent epistemic/doxastic logic (¿and deontic,...?) with group knowledge (group belief) operators.
- $E\varphi$ means *everybody knows (believes) that* φ. It can be defined as $\Box_{a_1}\varphi \wedge \Box_{a_2}\varphi \wedge \cdots$ (for any $a_i \in \mathcal{A}$).

Group knowledge

- We can extend our multi-agent epistemic/doxastic logic (¿and deontic,...?) with group knowledge (group belief) operators.
- $E\varphi$ means *everybody knows (believes) that* φ. It can be defined as $\Box_{a_1}\varphi \wedge \Box_{a_2}\varphi \wedge \cdots$ (for any $a_i \in \mathcal{A}$).
- Its dual, $\widehat{E}\varphi$, means *it is possible for somebody that* φ. It can be defined as $\Diamond_{a_1}\varphi \vee \Diamond_{a_2}\varphi \vee \cdots$ (for any $a_i \in \mathcal{A}$).

Group knowledge

- We can extend our multi-agent epistemic/doxastic logic (¿and deontic,...?) with group knowledge (group belief) operators.
- $E\varphi$ means *everybody knows (believes) that* φ. It can be defined as $\Box_{a_1}\varphi \wedge \Box_{a_2}\varphi \wedge \cdots$ (for any $a_i \in \mathcal{A}$).
- Its dual, $\widehat{E}\varphi$, means *it is possible for somebody that* φ. It can be defined as $\Diamond_{a_1}\varphi \vee \Diamond_{a_2}\varphi \vee \cdots$ (for any $a_i \in \mathcal{A}$).
- $C\varphi$ means *it is common knowledge (belief) that* φ. It can be intuitively understood as the infinite conjunction $E\varphi \wedge EE\varphi \wedge EEE\varphi \wedge \cdots$

Group knowledge

- We can extend our multi-agent epistemic/doxastic logic (¿and deontic,...?) with group knowledge (group belief) operators.
- $E\varphi$ means *everybody knows (believes) that* φ. It can be defined as $\Box_{a_1}\varphi \wedge \Box_{a_2}\varphi \wedge \cdots$ (for any $a_i \in \mathcal{A}$).
- Its dual, $\widehat{E}\varphi$, means *it is possible for somebody that* φ. It can be defined as $\Diamond_{a_1}\varphi \vee \Diamond_{a_2}\varphi \vee \cdots$ (for any $a_i \in \mathcal{A}$).
- $C\varphi$ means *it is common knowledge (belief) that* φ. It can be intuitively understood as the infinite conjunction $E\varphi \wedge EE\varphi \wedge EEE\varphi \wedge \cdots$
- Its dual, $\widehat{C}\varphi$ (*it is compatible with common knowledge (belief) that* φ) can be intuitively understood as the infinite disjunction $\widehat{E}\varphi \vee \widehat{E}\widehat{E}\varphi \vee \widehat{E}\widehat{E}\widehat{E}\varphi \vee \cdots$

Rules for E and $\widehat{E}$

E and $\widehat{E}$ may be treated as quantifiers:

E Rule

$$\frac{\sigma :: E\alpha}{\sigma :: \Box_{a_i}\alpha}$$

(For every agent a_i that appears in the branch.)

$\widehat{E}$ Rule

$$\frac{\sigma :: \widehat{E}\alpha}{\sigma :: \widehat{E}_{a_i}\alpha}$$

(Where agent a_i is new in the branch.)

Rules for C

C is treated in the same way that the operator $\Box$:

Knowledge

$$\frac{\sigma :: C\alpha}{\sigma :: \alpha}$$

Belief

$$\frac{\sigma :: C\alpha}{\sigma :: \widehat{C}\alpha}$$

Inheritance rules for C

IRC:

IRT: $$\frac{\sigma :: C\alpha}{\sigma.a_i n :: C\alpha}$$
(For any label $\sigma.a_i n$ that appears in the branch)

IRS5: $$\frac{\sigma :: C\alpha}{\overline{\sigma.a_i n :: C\alpha}}$$
(For any label $\sigma, \sigma.a_i n$ that appears in the branch)

The case of S4 is like T, but we have to consider the cases where we have applied the restriction of the rule R$\Diamond$.

DB-ableaux

- Before talking about the operator $\widehat{C}$, we have to speak about DB-tableaux.

DB-ableaux

- Before talking about the operator $\widehat{C}$, we have to speak about DB-tableaux.
- We call DB-tableaux to a modified tableau method proposed independently by Díaz Estévez and Boolos to deal with formulas of the form $\forall x \exists y \varphi(x)$.

DB-ableaux

- Before talking about the operator $\widehat{C}$, we have to speak about DB-tableaux.
- We call DB-tableaux to a modified tableau method proposed independently by Díaz Estévez and Boolos to deal with formulas of the form $\forall x \exists y \varphi(x)$.
- In this method, the rule

$$\frac{\exists x \varphi}{\varphi\left(k_{n+1}/x\right)}$$

(where k_n is the last parameter that appears in the branch) is replaced with:

$$\frac{\exists x \varphi}{\varphi\left(k_1/x\right) \mid \cdots \mid \varphi\left(k_n/x\right) \mid \varphi\left(k_{n+1}/x\right)}$$

DB-ableaux for $\widehat{C}$

- We will use an infinitary version of the previous rule to deal with the operator $\widehat{C}$.

R$\widehat{C}$

$$\frac{\sigma :: \widehat{C}\alpha}{\sigma :: \widehat{E}\alpha \mid \sigma :: \widehat{E}\widehat{E}\alpha \mid \cdots}$$

(Until we find an open branch.)

DB-ableaux for $\widehat{C}$

- We will use an infinitary version of the previous rule to deal with the operator $\widehat{C}$.

R$\widehat{C}$

$$\frac{\sigma :: \widehat{C}\alpha}{\sigma :: \widehat{E}\alpha \mid \sigma :: \widehat{E}\widehat{E}\alpha \mid \cdots}$$

(Until we find an open branch.)

- If the formula has a model, we will find it in a finite number of steps; if the formula has not a model, the tableau becomes infinite.

Temporal logic

- We can interpret $\Box$ and $\Diamond$ as temporal operators:
  - $\Box\varphi$ means “always (now and in the future) φ”.
  - $\Diamond\varphi$ means “sometime (now or in the future) φ”.

Temporal logic

- We can interpret $\Box$ and $\Diamond$ as temporal operators:
  - $\Box\varphi$ means "always (now and in the future) φ".
  - $\Diamond\varphi$ means "sometime (now or in the future) φ".
- It is usual to introduce two more operators: $\bigcirc$ y U (we are not going to deal with branching-time operators):
  - $\bigcirc\varphi$ means " at the next moment, φ".
  - $\varphi U\psi$ means "φ until ψ".

Tableaux for temporal logic

We have to introduce the following changes:

Labels: Each label $t \in \mathbb{N}$ represents a moment of the time (we are dealing with discrete time).

Tableaux for temporal logic

We have to introduce the following changes:

Labels: Each label $t \in \mathbb{N}$ represents a moment of the time (we are dealing with discrete time).

Operator $\Box$: Rules for $\Box$ are similar to the previous cases:

R$\Box$

$$\frac{t :: \Box\alpha}{t :: \alpha}$$

RI$\Box$

$$\frac{t :: \Box\alpha}{t' :: \Box\alpha}$$

(For any label $t' > t$ that appears in the branch)

Tableaux for temporal logic II

Operator $\bigcirc$: The rules for $\bigcirc$ and its negation are:

R$\bigcirc$

$$\frac{t :: \bigcirc\alpha}{t+1 :: \alpha}$$

R$\neg\Box$

$$\frac{t :: \neg \bigcirc \alpha}{t+1 :: \neg\alpha}$$

Recursive rules

- For dealing with the operators $\Diamond$ and U (and it negation) we need *recursive rules.*

Recursive rules

- For dealing with the operators $\Diamond$ and U (and it negation) we need *recursive rules.*
- Recursive rules are of the form

$$\frac{A}{SC \parallel RC}$$

Recursive rules

- For dealing with the operators $\Diamond$ and U (and it negation) we need *recursive rules.*
- Recursive rules are of the form

$$\frac{A}{SC \parallel RC}$$

- SC is the *stop condition*. RC is the *recursive clause*.

Recursive rules

- For dealing with the operators $\Diamond$ and U (and it negation) we need *recursive rules.*
- Recursive rules are of the form

$$\frac{A}{SC \parallel RC}$$

- SC is the *stop condition*. RC is the *recursive clause*.
- If SC gives rise to an open branch, we have finished the application of the rule.

Recursive rules

- For dealing with the operators $\Diamond$ and U (and it negation) we need *recursive rules.*
- Recursive rules are of the form

$$\frac{A}{SC \parallel RC}$$

- SC is the *stop condition*. RC is the *recursive clause*.
- If SC gives rise to an open branch, we have finished the application of the rule.
- If SC gives rise to a closed branch, we have to apply RC, which makes us to apply the rule again at the next moment of the time.

Recursive rules

- For dealing with the operators $\Diamond$ and U (and it negation) we need *recursive rules.*
- Recursive rules are of the form

$$\frac{A}{SC \parallel RC}$$

- SC is the *stop condition*. RC is the *recursive clause*.
- If SC gives rise to an open branch, we have finished the application of the rule.
- If SC gives rise to a closed branch, we have to apply RC, which makes us to apply the rule again at the next moment of the time.
- One more time, if the formula has a finite model, we find it in a finite number of steps; if the formula has not a model, the tableau becomes infinite.

Rules for $\lozenge$ and U

- R$\lozenge$:

$$\frac{t :: \lozenge\alpha}{t :: \alpha \parallel t :: \bigcirc\lozenge\alpha}$$

- RU:

$$\frac{\beta U \alpha}{\begin{array}{c} t :: \beta \end{array} \left\| \begin{array}{c} t :: \alpha \\ t :: \bigcirc\alpha U \beta \end{array}\right.}$$

- R$\neg U$:

$$\frac{t :: \neg(\alpha U \beta)}{\begin{array}{c} t :: \neg\alpha \\ t :: \neg\beta \end{array} \left\| \begin{array}{c} t :: \alpha \\ t :: \neg \\ t :: \bigcirc\neg(\alpha U \beta) \end{array}\right.}$$

Conclusions

- We can present labelled tableau methods for modal logic. The so called *inheritance rules* expresses the properties of the accessibility relation. We can prove that these methods are correct and complete.

Conclusions

- We can present labelled tableau methods for modal logic. The so called *inheritance rules* expresses the properties of the accessibility relation. We can prove that these methods are correct and complete.
- We can extend these methods to multi-modal logics using more complicated labels and rules.

Conclusions

- We can present labelled tableau methods for modal logic. The so called *inheritance rules* expresses the properties of the accessibility relation. We can prove that these methods are correct and complete.
- We can extend these methods to multi-modal logics using more complicated labels and rules.
- For some infinitary operators we have to use DB-Tableaux.

Conclusions

- We can present labelled tableau methods for modal logic. The so called *inheritance rules* expresses the properties of the accessibility relation. We can prove that these methods are correct and complete.
- We can extend these methods to multi-modal logics using more complicated labels and rules.
- For some infinitary operators we have to use DB-Tableaux.
- For temporal operators we use recursive rules.

Conclusions

- We can present labelled tableau methods for modal logic. The so called *inheritance rules* expresses the properties of the accessibility relation. We can prove that these methods are correct and complete.
- We can extend these methods to multi-modal logics using more complicated labels and rules.
- For some infinitary operators we have to use DB-Tableaux.
- For temporal operators we use recursive rules.
- In the last two cases, the tableau may become infinite.

End

Thank you

Muito obrigado