Available online at [www.sciencedirect.com](http://www.sciencedirect.com/)



[Electronic Notes in Theoretical Computer Science 278 (2011) 17–30](http://dx.doi.org/10.1016/j.entcs.2011.10.003)

[www.elsevier.com/locate/entcs](http://www.elsevier.com/locate/entcs)

Tableau Method and NEXPTIME-Completeness of DEL-Sequents

Guillaume Aucher[1](#_bookmark0)*,*[2](#_bookmark0)

*IRISA*

*INRIA - University of Rennes 1 Rennes, France*

Bastien Maubert[3](#_bookmark0) Franc¸ois Schwarzentruber[4](#_bookmark0)

*IRISA*

*ENS Cachan Rennes, France*

**Abstract**

Dynamic Epistemic Logic (DEL) deals with the representation of situations in a multi-agent and dynamic setting. It can express in a uniform way statements about:

1. what is true about an initial situation
2. what is true about an event occurring in this situation
3. what is true about the resulting situation after the event has occurred.

After proving that what we can infer about (ii) given (i) and (iii) and what we can infer about (i) given

(ii) and (iii) are both reducible to what we can infer about (iii) given (i) and (ii), we provide a tableau method deciding whether such an inference is valid. We implement it in LOTRECscheme and show that this decision problem is NEXPTIME-complete. This contributes to the proof theory and the study of the computational complexity of DEL which have rather been neglected so far.

*Keywords:* Dynamic epistemic logic, tableau method, computational complexity

# Introduction

Dynamic Epistemic Logic (DEL) deals with the logical study in a multi-agent setting of knowledge and belief change, and more generally of information change [[11](#_bookmark24)]. To account for these logical dynamics, the core idea of DEL is to split the task of

1 We thank Sophie Pinchinat for helpful discussions and three reviewers for comments.

2 Email: [guillaume.aucher@irisa.fr](mailto:guillaume.aucher@irisa.fr)

3 Email: [bastien.maubert@irisa.fr](mailto:bastien.maubert@irisa.fr)

4 Email: [francois.schwarzentruber@bretagne.ens-cachan.fr](mailto:francois.schwarzentruber@bretagne.ens-cachan.fr)

1571-0661 © 2011 Elsevier B.V. Open access under [CC BY-NC-ND license.](http://creativecommons.org/licenses/by-nc-nd/3.0/)

[doi:10.1016/j.entcs.2011.10.003](http://dx.doi.org/10.1016/j.entcs.2011.10.003)

representing the agents’ beliefs into three parts: first, one represents their beliefs about an initial situation; second, one represents their beliefs about an event taking place in this situation; third, one represents the way the agents update their beliefs about the situation after (or during) the occurrence of the event. Consequently, one can express uniformly within the logical framework of DEL statements about:

1. what is true about an initial situation,
2. what is true about an event occurring in this situation,
3. what is true about the resulting situation after the event has occurred.

From a logical point of view, this trichotomy begs the following three questions. Given (i) and (ii), what can we infer about (iii)? Given (i) and (iii), what can we infer about (ii)? Given (ii) and (iii), what can we infer about (i)? Providing formal tools that can be used to answer these questions is certainly of interest for human or artificial agents. Indeed, they could not only use them to plan their actions to achieve a given epistemic goal (the first and second questions actually correspond respectively to the problems of *deductive* and *abductive* planning in the situation calculus), but they could also use them to explain and determine a posteriori the causes that lead to a given situation. Nevertheless, to be applicable, these formal tools should lead to implementable decision procedures. To this aim, we provide a tableau method giving an answer to the first question. This is sufficient since we prove that the two other questions are in fact both reducible formally to the first one.

The paper is organized as follows. In Section [2](#_bookmark1), we define our three DEL- sequents corresponding to our three questions above, and we show that these DEL- sequents are interdefinable. In Section [3](#_bookmark3), we provide two terminating, sound and complete tableau methods. This leads us to define in Section [4](#_bookmark11) an algorithm in NEXPTIME, which we prove to be optimal by reducing a tiling problem known to be NEXPTIME-complete to our decision problem. A link to an implementation of our tableau method in LOTRECscheme is provided in Section [5](#_bookmark12). Finally, we conclude in Section [6](#_bookmark13) by a discussion of related works.

# Dynamic Epistemic Logic: DEL-sequents

* 1. *Representation of the initial situation: L-model*

In the rest of this paper, Φ is a countable set of propositional letters (possibly infinite) called *atomic facts* which describe static situations, and *Agt* is a finite set of agents. A *L-model* is a tuple *M* = (*W, R, V* ) where:

* + - *W* is a non-empty set of possible worlds,
    - *R* : *Agt →* 2*W ×W* is a function assigning to each agent *j ∈ Agt* an accessibility relation on *W* ,
    - *V* :Φ *→* 2*W* is a function assigning to each propositional letter of Φ a subset of *W* . The function *V* is called a valuation.

We write *w ∈ M* for *w ∈ W* , and (*M, w*) is called a pointed *L*-model (*w* of- ten represents the actual world). If *w, v ∈ W* , we write *wRjv* for *R*(*j*)(*w, v*) and *Rj*(*w*) = *{v ∈ W | wRjv}*. Intuitively, *wRjv* means that in world *w* agent *j* con- siders that world *v* might correspond to the actual world. Then, we define the following epistemic language *L* that can be used to describe and state properties of *L*-models:

*L* : *φ* ::= *p | ¬φ | φ ∧ φ | Bjφ*

where *p* ranges over Φ and *j* over *Agt*. We define *φ ∨ ψ* =def *¬*(*¬φ ∧ ¬ψ*) and

*⟨Bj⟩φ* =def *¬Bj¬φ*. The symbol *T* is an abbreviation for *p ∨ ¬p* for a chosen *p ∈* Φ. Let *M* be a *L*-model, *w ∈ M* and *φ ∈ L*. *M,w |*= *φ* is defined inductively as follows:

*M,w |*= *p* iff *w ∈ V* (*p*) *M,w |*= *φ ∧ ψ* iff *M,w |*= *φ* and *M,w |*= *ψ*

*M,w |*= *¬φ* iff not *M,w |*= *φ M,w |*= *Bjφ* iff for all *v ∈ Rj*(*w*), *M,v |*= *φ*

The formula *Bjφ* reads as “agent *j* believes *φ*”. Its truth conditions are defined in such a way that agent *j* believes *φ* holds in a possible world when *φ* holds in all the worlds agent *j* considers possible.

* 1. *Representation of the event: Lj-model*

The propositional letters *pj*

*ψ*

describing events are called *atomic events* and range

over Φ*j* = *{pj* *ψ* ranges over *L}*. The reading of *pj* is “an event of precondition *ψ*

*ψ*

*ψ*

is occurring”. A *Lj-model* is a tuple *Mj* = (*Wj, Rj,V j*) where:

* + - *Wj* is a non-empty set of possible events,
    - *Rj* : *Agt →* 2*W ′×W ′* is a function assigning to each agent *j ∈ Agt* an accessibility relation on *Wj*,
    - *V j* : Φ*j →* 2*W ′* is a function assigning to each propositionnal letter of Φ*j* a

subset of *Wj* such that for all *wj ∈ Wj*, there is at most one *pj*

*ψ*

such that

*wj ∈ V* (*pj* ) (Exclusivity).

*ψ*

We write *wj ∈ Mj* for *wj ∈ Wj*, and (*Mj, wj*) is called a *pointed Lj-model* (*wj* often represents the actual event). If *wj, vj ∈ Wj*, we write *wjRj vj* for *Rj*(*j*)(*wj, vj*) and *Rj* (*wj*) = *{vj ∈ Wj | wjRj vj}*. Intuitively, *vj ∈ Rj*(*wj*) means that while the

*j*

*j j*

possible event represented by *wj* is occurring, agent *j* considers possible that the

possible event represented by *vj* is actually occurring. Our definition of a *Lj*-model is equivalent to the definition of an action signature in the logical framework of [[3](#_bookmark16)]. [5](#_bookmark2) Just as we defined a language *L* for *L*-models, we also define a language *Lj* for *Lj*-models:

*Lj* : *φj* ::= *pj | ¬φj | φj ∧ φj | Bjφj*

*ψ*

5 If Σ = (*W ′, R′,* (*w′ ,..., w′* )) is an action signature and *φ*1*,..., φn ∈ L*, then the *L′-model associated*

1 *n*

*to* (Σ*, φ*1*,..., φn*) is the tuple *M′* = (*W ′, R′,V ′*) where *V ′*(*p′* )= *{w′}* if *ψ* = *φi*, *V ′*(*p′* )= *{w′ ,..., w′ }*

if *ψ* = *T*, and *V ′*(*p′* )= *∅* otherwise.

*ψ*

*ψ i ψ* 1 *n*

where *pj* ranges over Φ*j* = *{pj* *ψ ∈ L}* and *j* over *Agt*. In fact, *Lj* was already

*ψ*

*ψ*

introduced in [[5](#_bookmark18)]. In the sequel, formulas of *Lj* are always indexed by the quotation mark *j*, unlike formulas of *L*. The truth conditions of the language *Lj* are identical to the ones of the language *L*. Let *Mj* be a *Lj*-model, *wj ∈ Mj* and *φj ∈ Lj*. *Mj, wj |*= *φj* is defined inductively as follows:

*Mj, wj |*= *pj* iff *wj ∈ V j*(*pj* )

*ψ*

*ψ*

*Mj, wj |*= *¬φj* iff not *Mj, wj |*= *φj*

*Mj, wj |*= *φj ∧ ψj* iff *Mj, wj |*= *φj* and *Mj, wj |*= *ψj*

*Mj, wj |*= *Bjφj* iff for all *vj ∈ Rj*(*wj*), *Mj, vj |*= *φj*

* 1. *Update of the initial situation by the event: product update*

A *Lj*-model induces the definition of a precondition function. The precondition Pre(*wj*) of a possible event *wj* corresponds to the property that should be true at a world *w* of a *L*-model so that the possible event *wj* can ‘physically’ occur in this world *w*. The *precondition function Pre* : *Wj → L induced by the Lj-model*

*Mj* = (*Wj, Rj,V j*) is defined as follows: Pre(*wj*) = *ψ* if there is *pj*

*ψ*

such that

*Mj, wj |*= *pj* ; Pre(*wj*)= *T* otherwise.

*ψ*

We then redefine equivalently in our setting the BMS product update of [[4](#_bookmark17)] as follows. Let (*M, w*) = (*W, R, V, w*) be a pointed *L*-model and let (*Mj, wj*) = (*Wj, Rj,V j, wj*) be a pointed *Lj*-model such that *M,w |*= Pre(*wj*). The *prod- uct update of* (*M, w*) *and* (*Mj, wj*) is the pointed *L*-model (*M⊗ Mj,* (*w, wj*)) = (*W⊗, R⊗,V ⊗,* (*w, wj*)) defined as follows:

* + - *W⊗* = *{*(*v, vj*) *∈ W × Wj | M,v |*= Pre(*vj*)*}*,
    - *R⊗*(*v, vj*)= *{*(*u, uj*) *∈ W⊗ | u ∈ Rj*(*v*) and *uj ∈ Rj* (*vj*)*}*,

*j j*

* + - *V ⊗*(*p*)= *{*(*v, vj*) *∈ W⊗ | M,v |*= *p}*.

This product update yields a new *L*-model (*M, w*) *⊗* (*Mj, wj*) representing how the new situation which was previously represented by (*M, w*) is perceived by the agents after the occurrence of the event represented by (*Mj, wj*).

* 1. *Deﬁnitions of our DEL-sequents*

Let *φ, φjj ∈ L* and *φj ∈ Lj*. We define the logical consequence relations *φ, φj * *φjj*, *φ, φjj* 2 *φj* and *φj, φjj* 3 *φ* as follows. The second and third relations can be used for epistemic planning and goal regression respectively.

*φ, φj * *φjj* iff for all pointed *L*-model (*M, w*), and *Lj*-model (*Mj, wj*) such that

*M,w |*= Pre(*wj*), *M,w |*= *φ* and *Mj, wj |*= *φj*, it holds that (*M, w*) *⊗* (*Mj, wj*) *|*= *φjj*

*φ, φjj* 2 *φj* iff for all pointed *L*-models (*M, w*), and (*Mjj, wjj*) such that *M,w |*= *φ*

and *Mjj, wjj |*= *φjj*,

if (*Mj, wj*) is a pointed *Lj*-model such that *M,w |*= Pre(*wj*) and (*M, w*) *⊗* (*Mj, wj*) is bisimilar to (*Mjj, wjj*), then *Mj, wj |*= *φj*

*φj, φjj* 3 *φ* iff for all pointed *Lj*-model (*Mj, wj*), and *L*-model (*Mjj, wjj*) such that

*Mj, wj |*= *φj* and *Mjj, wjj |*= *φjj*,

if (*M, w*) is a pointed *L*-model such that *M,w |*= Pre(*wj*) and (*M, w*) *⊗* (*Mj, wj*) is bisimilar to (*Mjj, wjj*), then *M,w |*= *φ*

In fact, as the following proposition shows, our three DEL-sequent are interdefinable. Therefore, in the rest of this paper, we will focus only on providing a tableau method for the DEL-sequent *φ, φj * *φjj*. Tableau methods and complexity results for the other DEL-sequents can easily be adapted from the ones provided for this DEL- sequent.

**Proposition 2.1** *For all φ, φjj ∈L and φj ∈ Lj,*

*φ, φjj* 2 *φj iff φ, ¬φj * *¬φjj φj, φjj* 3 *φ iff ¬φ, φj * *¬φjj*

# Tableau method

We consider three formulae, *φ ∈ L*, *φj ∈ Lj* and *φjj ∈ L*, and we want to adress the problem of deciding whether *φ, φj |*= *φjj* holds. To do so we equivalently decide whether there exist a pointed *L*-model (*M, w*) and a pointed *Lj*-model (*Mj, wj*) such that *M,w |*= Pre(*wj*), *M,w |*= *φ*, *Mj, wj |*= *φj* and *M⊗ Mj,* (*w, wj*) *|*= *¬φjj*. We call this dual problem the *satisﬁability problem*.

* 1. *Tableau method description*

The formulas that appear in our tableau method and that we call *tableau formulas*

are of the following kind:

* + - (*l φ*): *l* is a label *lw* (resp. *lw′* ) that represents a world of the model *M* (resp. *Mj*) being constructed, and *φ* is a formula of *L* (resp. *Lj*) that should be true at *M,w* (resp. *Mj, wj*).
    - (*lw lw′ φjj*): *lw* represents a world *w* of *M*, *lw′* a world *wj* of *Mj*, and *φjj* is a formula of *L* that should be true at *M⊗ Mj,* (*w, wj*). Moreover, (*lw lw′* 0) means that (*w, wj*) is not in *M⊗ Mj*.
    - (*R l lj*) (resp. (*Rj l lj*)): *R* (resp. *Rj*) is some *Rj* (resp. *Rj* ), *l* and *lj* represent two worlds *w* and *u* (resp. *wj* and *uj*) such that *wRju* (resp. *wjRjuj*).

*j*

* + - *⊥*: Denotes an inconsistency.

A *tableau rule* is represented by a *numerator N* above a line and a finite list of

*denominators D*1*,..., Dk* below this line, separated by vertical bars:

*N D*1 *| ... | Dk*

The numerator and the denominators are finite sets of tableau formulas.

A *tableau for* a triple (*φ, φj, φjj*) of formulas is a finite tree with a set of tableau formulas at each node, and whose root is:

Γ0 = *{*(*lw φ*)*,* (*lw′ φj*)*,* (*lw lw′ φjj*)*}*

A rule with numerator *N* is *applicable* to a node carrying a set Γ if Γ contains an instance of *N* . If no rule is applicable, Γ is said to be *saturated*. We call a node *n* an *end node* if the set of formulas Γ it carries is saturated, or if *⊥∈* Γ. The tableau is extended the following way:

1. Choose a leaf node *n* carrying Γ where *n* is not an end node, and choose a rule

*ρ* applicable to *n*.

1. (a) If *ρ* has only one denominator, add the appropriate instanciation to Γ.

(b) If *ρ* has *k* denominators with *k >* 1, create *k* successor nodes for *n*, where each successor *i* carries the union of Γ with an appropriate instanciation of denominator *Ði*.

A branch in a tableau is a path from the root to an end node. A branch is *closed* if its end node contains *⊥*, otherwise it is *open*. A tableau is *closed* if all its branches are closed, otherwise it is *open*. A triple (*φ, φj, φjj*) is said to be *consistent* if no tableau for (*φ, φj, φjj*) is closed, and a triple (*φ, φj, φjj*) is a *theorem*, which we write *φ, φj * *φjj*, if there is a closed tableau for (*φ, φj, чφjj*).

* 1. *Tableau rules*

Common rules for *M*, *Mj* and *Mjj* (*l* is either *lw*, *lw′* or *lw lw′* ):

(*l φ Λ ψ*)

(*l ч*(*φ Λ ψ*))

*Λ*

*чΛ* (*l ччφ*) *ч* (*l p*)(*l чp*) *⊥*

(*l φ*) (*l ψ*)

(*l чφ*) *|* (*l чψ*)

(*l φ*)

*⊥*

where *p ∈* Φ

Specific rules for *M* and *Mj* (*l* is either *lw* or *lw′* ):

(*l ⟨Bj⟩φ*)

(*l Bjφ*)(*R l lj*)

(*lw′ pj* )(*lw′ pj* )

(*R l lj*)(*lj φ*)

*⟨Bj⟩*

(*lj φ*) *Bj*

*φ ψ*

*⊥* Excl

Specific rules for *Mjj*:

(*lw lw′ ⟨Bj⟩φ*)

*j*

*⟨Bj⟩⊗*

where *φ /*= *ψ*

(*lw lw′ Bjφ*)(*R lw lu*)(*Rj lw′ lu′* )

*Bj⊗*

(*R lw lu*)(*R lw′ lu′* )(*lu lu′ φ*)

(*lu lu′ φ*)*|*(*lu lu′* 0)

(*lw lw′ p*)

*→*

1

(*lw p*)

(*lw lw′* 0)(*lw′ pj* )

*ψ*

(*lw lw′ чp*)

2

*→*

(*lw чp*)

(*lw lw′ φ*)(*lw′ pj* )

*ψ*

(*lw*

*чψ*) Pre1

(*lw*

*ψ*) Pre2

where *φ /*=0

**Remark 3.1** Another sound and complete tableau method can be obtained by replacing the rule Pre1 above by the following rule:

(*lw lw′* 0) Pre*j*

(*lw′ pj* ) (*lw чψ*1) *| ... |* (*lw′ pj* ) (*lw чψn*) 1

*ψ*1

*ψn*

where *pj*

*ψ*

1

*,..., pj*

*n*

*ψ*

is the set of propositional letters appearing in *φj* at the root of

the tableau. This second tableau method is more modular, in the sense that if we

remove Rule (Excl), then the resulting tableau method is still sound and complete with respect to the semantics where we do not impose the (Exclusivity) condition on *Lj*-models. Note also that the *L*-model and *Lj*-model obtained from an open branch with this tableau method do not need to be adapted to fulfill the satisfiability problem, as in the proof of Proposition [3.3](#_bookmark6) with the first tableau method.

* 1. *Tableau method soundness and completeness*

**Proposition 3.2 (Tableau method soundness)** *For all φ, φjj ∈ L, for all φj ∈ Lj, φ, φj * *φjj implies φ, φj * *φjj*

**Proof.** Instead of proving that *φ, φj ▶ φjj* implies *φ, φj |*= *φjj*, we equivalently prove that *φ, φj $ φjj* implies *φ, φj b φjj*. Suppose there exist a pointed *L*-model (*M, w*), a *Lj*-model (*Mj, wj*) such that *M,w |*= *φ*, *Mj, wj |*= *φj*, *M,w |*= Pre(*wj*) and *M⊗ Mj,* (*w, wj*) *$ φjj*. We must prove that every tableau for (*φ, φj, чφjj*) has an open branch (the proof of termination is postponed to Section [4](#_bookmark11)).

We say that a set Σ of tableau formulae is *interpretable* if there exist a *L*-model *M*, a *Lj*-model *Mj*, *f* : LABEL *→ W* and *fj* : LABEL’ *→ Wj* (where LABEL and LABEL’ are the sets of labels for worlds appearing in Σ) such that (*M, Mj, f,fj*) makes all the tableau formulae in Σ true for the following semantics *|*=*T* :

(*M, Mj, f,fj*) *|*=*T* (*lw φ*) iff *M,f* (*lw*) *|*= *φ* (*M, Mj, f,fj*) *|*=*T* (*lw′ φj*) iff *Mj,fj*(*lw′* ) *|*= *φj* (*M, Mj, f,fj*) *|*=*T* (*R lw lu*) iff *f* (*lw*)*Rf* (*lu*) (*M, Mj, f,fj*) *|*=*T* (*Rj lw′ lu′* ) iff *fj*(*lw′* )*Rjfj*(*lu′* )

(*M, Mj, f,fj*) *|*=*T* (*lw lw′* 0) iff *M,f* (*lw*) *$* Pre(*fj*(*lw′* ))

(*M, Mj, f,fj*) *|*=*T* (*lw lw′ φjj*) iff *M,f* (*lw*) *|*= Pre(*fj*(*lw′* )) and

*M⊗ Mj,* (*f* (*lw*)*,fj*(*lw′* )) *|*= *φjj*

(*M, Mj, f,fj*) *|*=*T ⊥* iff false

Notice that since *φ, φj $ φjj*, the set Γ0 = *{*(*lw φ*)(*lw′ φj*)(*lw lw′ чφjj*)*}* is in- terpretable. Furthermore, if a set of formulas is interpretable, it does not contain

*⊥*. So if we prove that when the numerator of a rule is interpretable, one of the denominators also is, then we have that every tableau for (*φ, φj, чφjj*) has an open branch. We only prove it for the specific rules of *Mjj*, the proof for the other rules being standard. In the following, when *f* is a function, we let *f* (*x '→ a*) be the function that maps *x* to *a* and *y* to *f* (*y*) if *y /*= *x*.

Rule *⟨Bj⟩⊗*: If *M, Mj, f,fj |*=*T* (*lw lw′ ⟨Bj⟩φ*) then *M⊗ Mj,* (*f* (*lw*)*,fj*(*lw′* )) *|*=

*⟨Bj⟩φ*. So there exists (*u, uj*) *∈ Wjj* such that (*f* (*lw*)*,fj*(*lw′* ))*Rjj*(*u, uj*) and *M⊗ Mj,* (*u, uj*) *|*= *φ*. Since (*f* (*lw*)*,fj*(*lw′* ))*Rjj*(*u, uj*) we have that *f* (*lw*)*Ru*, *fj*(*lw′* )*Rjuj* and *M,u |*= Pre(*uj*). So by letting *g* := *f* (*lu '→ u*) and *gj* := *fj*(*lu′ '→ uj*) we have that *M, Mj, g, gj |*=*T {*(*R lw lu*)(*Rj lw′ lu′* )(*lu lu′ φ*)*}*.

Rule *Bj⊗*: If *M, Mj, f,fj |*=*T {*(*lw lw′ Bjφ*)(*R lw lu*)(*Rj lw′ lu′* )*}* then

*M,f* (*lw*) *|*= Pre(*f* (*lw′* )), *M ⊗ Mj,* (*f* (*lw*)*,fj*(*lw′* )) *|*= *Bjφ*, *f* (*lw*)*Rf* (*lu*) and

*fj*(*lw′* )*Rjfj*(*lu′* ). So, either *M,f* (*lu*) *$* Pre(*lu′* ) or *M,f* (*lu*) *|*= Pre(*lu′* ). In the first case, *M, Mj, f,fj |*=*T* (*lu lu′* 0). In the second case, (*f* (*lu*)*,fj*(*lu′* )) is a world of *Mjj*, and (*f* (*lw*)*,fj*(*lw′* ))*Rjj*(*f* (*lu*)*,fj*(*lu′* )). Therefore we have *M⊗ Mj,* (*f* (*lu*)*,fj*(*lu′* )) *|*= *φ*, hence *M, Mj, f,fj |*=*T* (*lu lu′ φ*)

Rules *→*1 and *→*2: If *M, Mj, f,fj |*=*T {lw lw′ p}* then *M,f* (*lw*) *|*= Pre(*f* (*lw′* ) and *M⊗ Mj,* (*f* (*lw*)*,fj*(*lw′* )) *|*= *p*. Since *V jj*(*f* (*lw*)*,fj*(*lw′* )) = *V* (*f* (*lw*)), we have that *M,f* (*l**w*) *|*= *p*, hence *M, Mj, f,fj |*=*T {*(*lw p*)*}*. Rule *→*2 is proved similarly.

Rules Pre1 and Pre2: If *M, Mj, f,fj |*=*T {*(*lw lw′ φ*)(*lw′ pj* )*}* for some *φ /*= 0, then *M,f* (*lw*) *|*= Pre(*fj*(*lw′* )), and *fj*(*lw′* ) *∈ V j*(*pj* ). So *M,f* (*lw*) *|*= Pre(*pj* ), and

*ψ*

*ψ ψ*

*M, Mj, f,fj |*=*T* (*lw ψ*). As for Rule Pre1, if *M, Mj, f,fj |*=*T {*(*lw lw′* 0)(*lw′ pj* )*}*,

*ψ*

then, by definition of *|*=*T* , *M,f* (*lw*) *|*= *ч*Pre(*f* (*lw′* ) and *Mj,f* (*lw′* ) *|*= *pj* . There- fore, by the (Exclusivity) condition, Pre(*f* (*lw′* )) = *ψ*, and so *M,f* (*lw*) *|*= *чψ*, i.e. (*lw чψ*). *2*

*ψ*

**Proposition 3.3 (Tableau method completeness)** *For all φ, φjj ∈ L, for all*

*φj ∈ Lj, φ, φj * *φjj implies φ, φj * *φjj.*

**Proof.** We prove that *φ, φj b φjj* implies *φ, φj $ φjj*. Suppose there is a tableau for (*φ, φj, чφjj*) that has an open branch, we prove that there exist a pointed *L*-model (*M, w*) and a pointed *Lj*-model (*Mj, wj*) such that *M,w |*= Pre(*wj*), *M,w |*= *φ*, *Mj, wj |*= *φj* and *M⊗ Mj,* (*w, wj*) *|*= *чφjj*.

Let Γ*f* be the set of tableau formulas carried by the end node of the open branch.

We define *M* and *Mj* as follows. Each of them is built in two steps.

* Let *M*0 = (*W*0*, R*0*, V*0) with *W*0 = *{w |* (*lw ψ*) *∈* Γ*f }*, *V*0(*p*) = *{w |* (*lw p*) *∈* Γ*f }* and *R*0 = *{*(*w, u*) *|* (*R lw lu*) *∈* Γ*f }*. We then define the pointed *L*-model (*M, w*) as the bisimulation contraction of (*M*0*, w*0). [6](#_bookmark8)
* Let *Mj* = (*Wj, Rj,V j*) with *Wj* = *{wj* (*lw′ ψ*) *∈* Γ*f }*, *V* (*pj* ) =

*ψ*

*{wj |* (*lw′ pj* ) *∈* Γ*f }*, and *Rj* = *{*(*wj, uj*) *|* (*Rj lw′ lu′* ) *∈* Γ*f }*. Moreover, for all

*ψ*

*wj ∈ Mj* such that there is no (*lw′ pj* ) *∈* Γ*f* , we set *wj ∈ V j*(*pj*

) if *Sw′ /*= *∅*,

*ψ δSw′*

where *Sw′* and *δS**w′* are defined as follows. Let *Sw′* = *{w ∈ M* (*lw*0 *lw′ φ*) *∈* Γ*f*

for some *φ /*= 0 and *w*0 *∈ w}*. Then *δSw′* =

*w∈Sw′*

*δ*(*M, w*) where *δ*(*M, w*) is a

characteristic formula of (*M, w*) in *M*. Note that by soundness of Rule (Excl),

*Mj* satisfies the exclusivity condition.

Finally, we define *Mjj* as *M⊗ Mj* (we will prove later that *M,w |*= Pre(*wj*)). Lemmas [3.4](#_bookmark7) and [3.5](#_bookmark9) below establish the completeness of our tableau method. *2*

**Lemma 3.4** *If* (*lw*0 *φ*) *∈* Γ*f then M,w |*= *φ, and if* (*lw′ φj*) *∈* Γ*f then Mj, wj |*= *φj.*

**Proof.** We only prove it for *M*, it is similar for *Mj*. We first prove it for *M*0. The result is then transfered for *M* because *M*0 and *M* are bisimilar. The proof is done

6 Formally, *M* = (*W, R, V* ) where *W* = *{{v | v* is bisimilar to *w*0*}* *w*0 *∈ W*0*}*, *R* = *{*(*w, v*) *∈ W × W* there is *w*0 *∈ w* and *v*0 *∈ v* such that *v*0 *∈ R*(*w*0)*}* and *V* (*p*) = *{w ∈ W* there is *w*0 *∈ w* such that *w*0 *∈ V*0(*p*)*}*. We write *w* for the set of worlds of *W*0 which are bisimilar to *w*0.

by induction on *φ*.

*p, чp***:** by definition of *V* . As for the case *φ Λ ψ*, by saturation of rule *Λ*, Γ*f* also contains (*lw*0 *φ*) and (*lw*0 *ψ*). By induction hypothesis we have that *M*0*, w*0 *|*= *φ* and *M*0*, w*0 *|*= *ψ*, so *M*0*, w*0 *|*= *φ Λ ψ*. The cases *ч*(*φ Λ ψ*) and *ччφ* are proved similarly.

*⟨Bj⟩φ***:** By saturation of rule *⟨Bj⟩* there exists *lu*0 such that (*R lw*0 *lu*0 ) *∈* Γ*f* and (*lu*0 *φ*) *∈* Γ*f* . By induction hypothesis *M, u*0 *|*= *φ*, and *w*0*Ru*0 holds by construc- tion of *M*0, so *M*0*, w*0 *|*= *⟨Bj⟩φ*.

*Bjφ***:** Take some *u*0 in *W*0 such that *w*0*Ru*0 holds, we prove that *M*0*, u*0 *|*= *φ* and conclude that *M*0*, w*0 *|*= *Bjφ*. Since *w*0*Ru*0 holds we know by construction of *M*0 that (*R lw*0 *lu*0 ) is in Γ*f* . So by saturation of rule *Bj*, (*lu*0 *φ*) also belongs to Γ*f* , and by induction hypothesis *M*0*, u*0 *|*= *φ*.

**Lemma 3.5** *If* (*lw*0

*Mj,* (*w, wj*) *|*= *φjj.*

*2*

*lw′ φjj*) *∈* Γ*f with φjj /*= 0*, then M,w |*= *Pre*(*wj*) *and M⊗*

**Proof.** We first prove the following Fact:

**Fact 3.6** *If* (*lw lw′ φ*) *∈* Γ*f with φ /*= 0*, then M,w |*= *Pre*(*wj*)*.*

0

Assume towards a contradiction that *M,w $* Pre(*wj*). There are then two cases: either there is (*lw′ pj* ) *∈* Γ*f* or there is no (*lw′ pj* ) *∈* Γ*f* . In the first case, *M*0*, w*0 *$ ψ*

*ψ ψ*

because *ψ* = Pre(*wj*) by the (Exclusivity) condition, and so *M,w $ ψ*. However, by

the rule Pre2, (*lw*0 *ψ*) *∈* Γ*f* . Then, by Lemma [3.4](#_bookmark7), *M,w |*= *ψ*. This is impossible. In the second case, because (*lw*0 *lw′ φ*) *∈* Γ*f* for some *φ /*= 0, we have that *M,w |*= *δSw′* .

Besides, *Mj, wj |*= *pj*

*δSw′*

by definition of *V j*. Therefore, *M,w |*= Pre(*wj*). This is

also impossible.

We can now prove Lemma [3.5](#_bookmark9). We prove it by induction on *φ*.

*p, чp***:** By Rule *→*1, (*lw*0 *p*) *∈* Γ*f* , and so *M,w |*= *p* by Lemma [3.4](#_bookmark7). Moreover, by Fact [3.6](#_bookmark10), *M,w |*= Pre(*wj*). Therefore, *M⊗ Mj,* (*w, wj*) *|*= *p* by definition of the product update. The proof for *чp* is similar to the case of *p*. The proof of the other boolean cases *φ Λ ψ*, *ч*(*φ Λ ψ*) and *ччφ* is obtained by applying straightforwardly the Induction Hypothesis.

*⟨Bj⟩φ***:** If (*lw*0 *lw′ ⟨Bj⟩φ*) *∈* Γ*f* , then by saturation of Rule *⟨Bj⟩⊗*, (*R lw*0 *lu*),

(*Rj lw′ lu′* ) and (*lu*

0

*lu′ φ*) belong to Γ*f* . Now, by application of Fact [3.6](#_bookmark10), *M,w |*=

Pre(*wj*). Besides, by definition of *M* and *Mj*, *u ∈ R*(*w*) and *uj ∈ Rj*(*wj*). Hence,

(*u, uj*) *∈ R*(*w, wj*). Moreover, (*lu*

0

*lu′ φ*) *∈* Γ*f* and *φ /*= 0, so by application of

the induction hypothesis, *M,u |*= Pre(*uj*) and *M⊗ Mj,* (*u, uj*) *|*= *φ*. Therefore,

*M,w |*= Pre(*wj*) and *M⊗ Mj |*= *⟨Bj⟩φ*.

*Bjφ***:** If (*lw*0

*lw′ Bjφ*) *∈* Γ*f* , then by application of Fact [3.6](#_bookmark10), *M,w |*= Pre(*wj*). Let

(*u, uj*) *∈ R*(*w, wj*). Then *u ∈ R*(*w*) and *uj ∈ Rj*(*wj*) by definition of the product

update. Then there is *u*0 *∈ u* such that *u*0 *∈ R*(*w*0) by definition of *M*. Therefore, by definition of *M* and *Mj*, (*R lw*0 *lu*0 ) *∈* Γ*f* and (*Rj lw′ lu′* ) *∈* Γ*f* . Then, by saturation of Rule *Bj⊗*, either (i) (*lu*0 *lu′* 0) *∈* Γ*f* or (ii) (*lu*0 *lu′ φ*) *∈* Γ*f* .

1. In the first case, assume that there is (*lu′ pj* ) *∈* Γ*f* . Then, by saturation of Rule Pre1, (*lu*0 *чψ*) *∈* Γ*f* . Therefore, *M,u |*= *чψ* by Lemma [3.4](#_bookmark7). This is impossible because *Mj, uj |*= *pj* , and so *M,u |*= *ψ* should also hold because (*u, uj*) *∈M ⊗ Mj*. Therefore, there is no (*lu′ pj* ) *∈* Γ*f* .

*ψ*

*ψ*

*ψ*

* 1. If there is *u∗ ∈ u* such that (*lu∗ lu′ φ*) for some *φ /*= 0, then by Induction

0 0

Hypothesis, *M,u |*= Pre(*uj*) and *M⊗ Mj,* (*u, uj*) *|*= *ψ*.

* 1. If there is no *u∗*

0

*∈ u* such that (*l ∗*

0

*u*

*lu′ φ*) for some *φ /*= 0, then by

definition of *Su′* , *u ∈/ Su′* . Therefore, *M,u $ δSu′* , because the formula

*δ**S ′* characterizes exactly the worlds of *Su′* . Hence, *M,u $* Pre(*uj*) by

*u*

definition of *V j*, because *Mj, uj |*= *pj*

*δSu′*

. However, (*u, uj*) *∈ R*(*w, wj*),and so

*M,u |*= Pre(*uj*). There is a contradiction, so this case is impossible.

1. In the second case, by Induction Hypothesis, *M,u |*= Pre(*uj*) and *M ⊗ Mj,* (*u, uj*) *|*= *φ*

So, in any case, *M⊗ M,* (*u, uj*) *|*= *φ*. Therefore, *M⊗ Mj,* (*w, wj*) *|*= *Bjφ*.

*2*

# Complexity of the satisfiability problem

**Proposition 4.1** *The satisﬁability problem is in NEXPTIME.*

**Proof.** The tableau rules presented in Section [3.2](#_bookmark4) give rise to a non-deterministic algorithm running in exponential time. We say that a label *lu* is of depth *k* if there is a sequence *w* = *u*1*,..., uk* = *u* such that (*R lwi lwi*+1 ) for all *i < k*.

Let *pj*

*ψ*

1

*,..., pj*

*n*

*ψ*

be the set of atomic propositions appearing in *φj*. Let *δ*(*.*) be

the function that gives the modal depth of a given formula. The algorithm starts

with the following set of tableau formulas Γ0 = *{*(*lw φ*)*,* (*lw′ φj*)*,* (*lw lw′ φjj*)*}*. Let

*N* = *max{δ*(*φ*)*, δ*(*φj*)*, δ*(*φjj*)+ *maxk∈{*1*,...,n}δ*(*ψk*)*}*.

The algorithm runs as follows. For *i* =0 to *N* , we execute:

1. Γ*j* := the saturation of Γ*i* by rules *Λ, чΛ, ч, ⊥, Excl, →*1*, →*2*,* Pre1*,* Pre2;

*i*

1. If *⊥∈* Γ*j*, we stop the current execution;

*i*

1. Γ*i*+1 := the set of tableau formulas obtained by applying *⟨Bj⟩, Bj, ⟨Bj⟩⊗, Bj⊗*

on Γ*j*.

*i*

Step 1 is non-deterministic and corresponds to a Boolean saturation of labels of depth *i*. It non-deterministically runs in linear size of Γ*i*. Step 2 consists in checking if rule *⊥* has been executed. In this case, the current execution halts. Step 3 produces tableau formulas where labels are of depth *i* + 1.

Note that the maximal depth of formulas *ψjj* in tableau formulas of the form (*lu lu′ ψjj*) in Γ*i* is strictly decreasing with *i* (see rule *⟨Bj⟩⊗* and *Bj⊗*). So when *i > δ*(*φjj*), there is no more tableau formula of the form (*lu lu′ ψjj*) in Γ*i* with *ψjj /*= 0. So when *i > δ*(*φjj*), the rules Pre2, *⟨Bj⟩⊗* and *Bj⊗* will no more be applied.

Likewise, the maximal depth of formulas *ψ* (resp. *ψj*) in tableau formulas of

the form (*lu ψ*) (resp. (*lu′ ψj*) in Γ*i* is strictly decreasing with *i*. Moreover the depth of the formulas *ψ* appearing in a tableau formula of the form (*lu ψ*) is less

than *max{δ*(*φ*)*, maxk∈{*1*,...,n}δ*(*ψk*)*}*, and the depth of the formulas *ψj* appearing in a tableau formula of the form (*lu′ ψj*) is less than *δ*(*φj*).

At the end, Γ*N*+1 = *∅* and the algorithm has applied rules until saturation, that

is, the set of tableau formulas *N* Γ*i* is saturated. Now let us have a look at the

*i*=0

time required to execute the algorithm. Let *x* be the size of the input, that is the sum of the sizes of *φ, φj, φjj* and Pre(*pk*). Step 1 saturates the worlds *u*, *uj* and (*u, uj*) appearing in the tableau formulas in Γ*i*. For each of those worlds, the saturation is linear in *x*. Step 3 creates new tableau formulas for each *⟨Bj⟩*-formula appearing in Γ*j*. So for each world in Γ*i* it produces at most 2*x* new worlds. If we note *yi* the maximal number of worlds in Γ*i*, we have that *yi*+1 = 2*xyi*. So *yi* = (2*x*)*i*. The numbers of created worlds is bounded by (2*x*)*x*+1 and this construction takes an exponential amount of time. *2*

*i*

To prove NEXPTIME-hardness of the satisfiability problem, we will reduce a NEXPTIME-complete tiling problem to it [[6](#_bookmark19)]. Let *k* be a natural number. A tile type *t* is a 4-tuple of colors *t* = (*left*(*t*)*, right*(*t*)*, up*(*t*)*, down*(*t*))*.* The tiling problem we consider is defined as follows.

* Input: a finite set *T* of tile types, a *t*0 *∈ T* and a natural number *k* written in its binary form.
* Output: yes iff we can tile a *k × k* grid with the tile types of *T* and *t*0 being placed onto (0*,* 0).

In other worlds, the problem is to decide whether there exists a function *τ* from

*{*1*,... k}*2 to *T* satisfying the following constraints:

* 1. *τ* (0*,* 0) = *t*0;
  2. *up*(*τ* (*x, y*)) = *down*(*τ* (*x, y* + 1)) for all *x ∈ {*1*,..., k}*, *y ∈ {*1*,.* *,k −* 1*}*;
  3. *right*(*τ* (*x, y*)) = *left*(*τ* (*x* + 1*, y*)) for all *x ∈ {*1*,...,k −* 1*}*, *y ∈ {*1*,.* *, k}*.

**Proposition 4.2** *The satisﬁability problem is NEXPTIME-hard.*

**Proof.** Without loss of generality we may assume that *k* = 2*n*. Let us consider an instance (*T, t*0*, k*) of the tiling problem. We now define three formulas *φ, φj, φjj* that are computable in polynomial time in *|T |* and *n* such that it is possible to tile a *k × k* grid with the tile types of *T* and *t*0 being placed onto (0*,* 0) iff (*φ, φj, φjj*) is satisfiable.

There is a modal formula *χ* of length *O*(*n*2) which is satisfied in a frame iff the model contains as a submodel a binary tree of depth 2*n*, for instance:

*χ* = *l<*2*n Bjl* (*⟨Bj⟩pl Λ ⟨Bj⟩чpl*) *Λ* *i<l*((*pi → Bjpi*) *Λ* (*чpi → Bjчpi*)) *.*

The 22*n* leaves of the tree are labeled by 2*n*-tuples containing either *pi* or *чpi* for *i <* 2*n*. The 22*n* leaves correspond to the 22*n* tile locations (*x, y*) in the following sense: the values of the propositions *pi*, where *i < n*, correspond to the binary representation of the abscissa *x* and the values of the propositions *pi*, where *n ≤ i <* 2*n*, correspond to the binary representation of the ordinate *y*. For instance, for *n* =4 the location where *x* = 4 and *y* = 3 is represented by the following valuation:

*чp*0*, p*1*, чp*2*, чp*3 *чp*4*, чp*5*, p*6*, p*7

` ˛¸ x ` ˛¸ x

4 3

The idea of encoding the existence of a *k × k* tiling is as follows:

* *φ* encodes a tiling *τ*1 with such a binary tree such that *τ*1(0*,* 0) = *t*0;
* *φj* also encodes a tiling *τ*2 wich such a binary tree;
* *φjj* encodes that *τ*1 = *τ*2 = *τ* , and constraints (ii) and (iii) of the tiling *τ* .

## Defining *φ*

We define the following formula: *path* = *⟨Bj⟩*2*n*+*|T |TΛ i<*2*n*+*|T | Bji⟨Bj⟩T*. The formula *path* says that there is a path whose length is greater that 2*n* + *|T |* but no shorter path in the model.

In order to define *φ*, each tiling type *t* is used as a proposition in the language, and means : ‘for the current location (*x, y*), we have *τ*1(*x, y*)= *t*’.

We define *φ* by:

*φ* = *χ Λ Bj*2*n* *path Λ* *t∈T t Λ* *t∈T* (*t →* *u∈T,u/*=*t чu*) *Λ* (( *i<*2*n чpi*) *→ t*0)

## Defining *φj*

For all *i*, we define *lj*

= *⟨Bj⟩i*+1*Bj⊥*. Let *χj* =

*i* 2*n−i−*1 *j*

*i<*2*n Bj*

*⟨Bj⟩Bj*

*li Λ ⟨Bj⟩Bj*2*n*

*i*

*−i−*1 *j j*

*чli*

. The formula *χ*

has the same

aim as *χ* and enables to enforce the existence of a binary tree where leaves correspond to the locations (*x, y*) of the tiling *τ*2. Formulas *lj* for *i <* 2*n* represent

*i*

the binary representation of (*x, y*).

Let *t*1*,..., t|T |* be an enumeration of elements of *T* . In order to define *φj*, for

= *l*

each tiling type *ti* we use the formula *tj*

*i*

*j*

*i*+2*n*

in the language whose intuitive

meaning is ‘for the current location (*x, y*), we have *τ*2(*x, y*)= *ti*.

We define *φj* by

*k*

*φj* = *χj Λ goodProduct Λ Bj*2*n*(

*tj Λ*

*i∈{*1*,...,|T |}*

(*tj →*

*чtj* )).

*i∈{*1*,...,|T |} k∈{*1*,...,|T |},k/*=*i*

*i*

*i*

where *goodProduct* = *i≤*2*n*+2*n*+*|T |*+1 *Bjipj* ensures that all worlds (*w, wj*) ap-

*T*

pear in the product model.

**Defining** *φjj*

The formula *φjj* will consider all the leaves (*w, wj*) of the product model where *w* is a leaf of the model *M* and *wj* is a leaf of the model *Mj* in order to encode the fact that *τ*1 = *τ*2 and the constraints (ii) and (iii).

We define *φjj* by:

*φjj* = *Bj* 2*n*[ (*α Λ β →* *j∈{*1*,...,|T |}*

(*tj ↔ tj* ))*Λ*

(*α Λ β*1 *→* *j∈{*1*,...,|T |}*(*tj →* *k∈{*1*,...,|T |}|down*(*t* )=*up*(*t* ) *tj* ))*Λ*

*j*

*k*

*j*

*k*

(*α*1 *Λ β →* *j∈{*1*,...,|T |}*(*tj →* *k∈{*1*,...,|T |}|left*(*t* )=*right*(*t* ) *tj* ))]

*k*

*j*

*k*

where:

* + *α* =

*i<n*

(*pi ↔ lj*) means ‘the abscissa *x* of the tile location of *w* is equal to

the absissa *xj* of the tile location of *wj*’;

*i*

* + *β* =

*n≤i<*2*n*

(*pi ↔ lj*) means ‘the ordinate *y* of the tile location of *w* is equal

to the ordinate *yj* of the tile location of *wj*’;

*i*

* + *α*1 = (*pj ↔ lj* ) *Λ чpi Λ lj Λ*

*i<n*

*j<i*

*j*

*i*

*i<j<n*

(*pj Λ чlj* ) means ’‘the abscissa

*j*

*x* of the tile location of *w* and the absissa *xj* of the tile location of *wj*’ are such that *xj* = *x* + 1;

* + *β*1 =

*n≤i<*2*n*

(*pj ↔ lj* ) *Λ чpi Λ lj Λ*

(*pj Λ чlj* ) means ’‘the

ordinate *y* of the tile location of *w* and the ordinate *yj* of the tile location of *wj*’ are such that *yj* = *y* + 1.

*n≤j<i*

*j*

*i*

*i<j<*2*n*

*j*

We leave the reader prove that we can tile a *k × k* grid with the tile types of *T*

and *t*0 being placed onto (0*,* 0) iff (*φ, φj, φjj*) is satisfiable.

*2*

# Implementation

The tableau method described in Remark [3.1](#_bookmark5) of Section [3.2](#_bookmark4) is implemented in LoTRECScheme (a variant of LoTREC [[8](#_bookmark21)] written in Scheme). Contrary to LoTREC, the system of LoTRECScheme allows the name of a node to be a couple (*w, wj*) and this functionnality is suitable for our tableau rules. You can find the implementation at the following web page:

<http://www.irisa.fr/prive/fschwarz/publications/m4m2011/>.

# Concluding remarks and related work

This paper contributes to the proof theory and the study of the computational complexity of DEL, which has been rather neglected so far. Indeed, most work in this field has often been inspired or applied to logico-philosophical puzzles such as for example the muddy children riddle, Fitch paradox, or Moorean sentences. Up to our knowledge, the only known results of computational complexity are the PSPACE-completeness of the satisfiability problem for public announcement logic

[[10](#_bookmark23)] and the polynomial time upper bound of the model-checking problem for public announcement logic. As for proof theory, a sound and complete sequent calculus for DEL has been developped in [[2](#_bookmark14)], yet in an algebraic setting. Because of this dif- ferent setting, the comparison cannot be systematic, but, unlike our DEL-sequents, their sequents *m*1*,..., q*1*,..., A*1*,..., mk,..., ql,..., An ▶ δ* are arbitrarily long and consist of different types of formulas which can contain propositions *m*1*,..., mk*, events *q*1*,..., ql* and agents *A*1*,..., An*, and which resolve into a single proposition *or* event *δ*. Some tableau methods have been proposed for DEL, but only for public announcement logic [[1](#_bookmark15),[7](#_bookmark20)] and hybrid public anouncement logic [[9](#_bookmark22)]. A terminat- ing tableau method has also been proposed for the full BMS framework in [[9](#_bookmark22)] by

encoding the reduction axioms as tableau rules. However, none of these tableau methods can somehow address the three questions raised in the introduction, be- cause the BMS language of [[3](#_bookmark16)] does not allow for partial and incomplete descriptions of events: an event model or a formula announced publicly specifies *completely* how all the agents perceive the occurrence of the corresponding event.

# References

1. Balbiani, P., H. van Ditmarsch, A. Herzig and T. de Lima, *Tableaux for public announcement logic*, Journal of Logic and Computation **20** (2010), pp. 55–76.
2. Baltag, A., B. Coecke and M. Sadrzadeh, *Algebra and sequent calculus for epistemic actions*, in:

*Proceedings of Workshop on Logic and Communication in Multi-Agent Systems (LCMAS’04)*, 2004,

pp. 60–78.

1. Baltag, A. and L. Moss, *Logic for epistemic programs*, Synthese **139** (2004), pp. 165–224.
2. Baltag, A., L. Moss and S. Solecki, *The logic of common knowledge, public announcement, and private suspicions*, in: I. Gilboa, editor, *Proceedings of the 7th conference on theoretical aspects of rationality and knowledge (TARK98)*, 1998, pp. 43–56.
3. Baltag, A., L. Moss and S. Solecki, *The logic of public announcements, common knowledge and private suspicions*, Technical report, Indiana University (1999).
4. Boas, P., *The convenience of tilings*, in: *Complexity, Logic, and Recursion Theory* (1997), pp. 331–363.
5. de Boer, M., *KE tableaux for public anouncement logic*, in: *Proceedings of Formal Approaches to Multi- Agent Systems Workshop (FAMAS 07)*, Durham UK, 2007.
6. Gasquet, O., A. Herzig, D. Longin and M. Saade, *LoTREC: Logical Tableaux Research Engineering* *Companion*, in: B. Beckert, editor, *International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX 2005), Koblenz, Germany, 14/09/05-17/09/05* (2005),

pp. 318–322.

URL <http://www.irit.fr/recherches/LILAC/Lotrec>

1. Hansen, J. U., *Terminating tableaux for dynamic epistemic logic*, Electronic Notes in Theoretical Computer Science **262** (2010), pp. 141–156.
2. Lutz, C., *Complexity and succinctness of public announcement logic*, in: H. Nakashima, M. P. Wellman,

G. Weiss and P. Stone, editors, *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006* (2006), pp. 137–143.

1. van Ditmarsch, H., W. van der Hoek and B. Kooi, “Dynamic Epistemic Logic,” Synthese library **337**, Springer, 2007.