DE GRUYTER OPEN

STUDIES IN LOGIC, GRAMMAR AND RHETORIC 48 (61) 2016

DOI: 10.1515/slgr-2016-0060

Tomasz Jarmużek
Nicolaus Copernicus University in Toruń

DEFINING COGNITIVE LOGICS BY NON-CLASSICAL TABLEAU RULES[1]

Abstract. In the paper we propose a new approach to formalization of cognitive logics. By cognitive logics we understand supraclassical, but non-trivial consequence operations, defined in a propositional language. We extend some paradigm of tableau methods, in which classical consequence Cn is defined, to stronger logics — monotonic, as well as non-monotonic ones — by specific use of non-classical tableau rules. So far, in that context tableaus have been treated as a way of formalizing other approaches to supraclassical logics, but we use them autonomically to generate various consequence operations. It requires a description of the hierarchy of non-classical tableau rules that result in different supraclassical consequence operations, so we give it.

Keywords: cognitive logic, commonsense reasoning, consequence operation, consequence relation, non-classical tableau rules, non-monotonic logics, tableau methods.

1. Aim and overview

The paper is devoted to an interesting and quite technical approach to defining supraclassical logics (logics that are stronger than Classical Propositional Logic[2].). The logics are usually understood as formal counterparts of everyday, commonsense reasoning, especially those among supraclassical ones that are non-monotonic. That is why we call them *cognitive logics*.

The intricacies of terminology that come out of that shall be explained later.

Briefly, in the article we consider the following issues. First, we take into account reasoning as a subject of logic, with special pressure on commonsense reasoning as cognitive reasoning. Next, we present an approach to logics by consequence operations: supraclassical, monotonic, as well as non-monotonic. Finally, we describe a transition between the definition of classical consequence operation by tableau method and supraclassical tableau consequence operations – cognitive logics. The two former issues constitute

ISBN 978-83-7431-512-8 ISSN 0860-150X

a proper framework for developing formal tools that capture cognitive reasoning, while the latter issue is the most original contribution of the article, since we present a hierarchy of tableau rules that enables defining of more and more general cognitive logics in the framework of tableau methods.

2. Reasoning as a subject of logic

Reasoning or argument — since we use these terms interchangeably — can be treated as a set of sentences where one of them, called *conclusion*, is distinguished, while the rest are called *premises*.[3] Symbolically, arguments are usually presented as below:

$$\begin{array}{c} \mathrm{P}_1 \\ \vdots \\ \mathrm{P}_n \\ \hline \mathrm{C}, \end{array}$$

where $n \geq 0$.

2.1. Deductive arguments

Argument $\{\mathrm{P}_1, \ldots, \mathrm{P}_n, \mathrm{C}\}$ is *valid* if and only if (in short: iff) in all situations in which all premises P_1, ..., P_n have got a designated logical value, conclusion C has also got a designated logical value.[4] Sometimes, instead of the word *valid*, we use also the word *correct*. Respectively, if an argument is not valid, we call it *invalid* or *incorrect*.

In the case of CPL a designated logical value is truth (sometimes written as 1, while falsity is written as 0), so classically, argument $\{\mathrm{P}_1, \ldots, \mathrm{P}_n, \mathrm{C}\}$ is valid iff in all situations in which all premises $\mathrm{P}_1, \ldots, \mathrm{P}_n$ are true, also conclusion C is true. When a given argument is valid in a classical sense, we usually say that the conclusion *logically follows from* the premises.

Valid arguments are called *deductive*. Deductive arguments have got very important properties that in some sense make them extraordinary. If given argument $\{\mathrm{P}_1, \ldots, \mathrm{P}_n, \mathrm{C}\}$ is valid, then also valid is an argument in which to the premises of the former argument was added one or more new premises, i.e. $\{\mathrm{P}_1, \ldots, \mathrm{P}_n, \mathrm{P}_{n+1}, \ldots, \mathrm{P}_m, \mathrm{C}\}$ is also valid. It is a result of the fact that by adding new premises P_{n+1}, ..., P_m we do not increase the set of situations that could make premises in the initial set $\{\mathrm{P}_1, \ldots, \mathrm{P}_n\}$ true, while conclusion C false. In other words, in any situation in which premises in the new set $\{\mathrm{P}_1, \ldots, \mathrm{P}_n, \mathrm{P}_{n+1}, \ldots, \mathrm{P}_m\}$ are true, premises in the old set $\{\mathrm{P}_1, \ldots, \mathrm{P}_n\}$ are also true. Hence, in any situation, if premises

$\{P_1, \ldots, P_n, P_{n+1}, \ldots, P_m\}$ are true, then conclusion C is true, since we assumed that argument $\{P_1, \ldots, P_n, C\}$ is valid. Additional premises usually even decrease the set of situations that make premises true. The property we have just discussed is called *monotony*. Hence any deductive argument is *monotonic*.

2.2. Non-deductive arguments as commonsense reasoning

In everyday life very often we do not use deductive arguments. Also, often we make conclusions in non-monotonic ways. To be honest, out of formal contexts (mathematical, logical etc) we very rarely dispose of sufficiently rich sets of premises to make on its base any interesting conclusions in a deductive way. Our arguments most often give the best conclusions we are able to assert on that ground. However, since the conclusions do not follow classically from premises, so additional pieces of information that complete the initial base of knowledge may advise us to reject some of the conclusions as less sure than it was used to be earlier. To better understand the dynamics of changing states of knowledge we propose the following example.

Example 1

We have some set of information:

(A) One of our friends has got a garden in town X.

(B) According to the friend the garden very often is flooded.

(C) One day we hear that in town X there was some cloudburst.

On the ground of (A), (B) and (C) we naturally (but non-classically, of course) conclude that:

(D) The friend's garden is flooded.

After some short time we learn that:

(E) Before the flood took place the gardens in town X had become surrounded by an embankment.

Well, now on the base of (A), (B), (C), enriched by (E) it seems not reliable to conclude (D). Even better is to conclude negation of (D).

However, what happens if we additionally learn that (F) in town X most of the embankments did not resist the pressure of water? Probably, we again will change our opinion, and assert (D).

In the given example we see a situation where the number of premises increases and the status of conclusion varies with the dynamics of the knowledge. The reasoning is non-deductive and non-monotonic. In general there

are a lot of ways for formal representing of such reasoning. Most of them include and accept also deductive arguments. Hence any of the tools starts from the classical approach, CPL, extending the mechanism of accepting conclusions with new principles.

Because arguments of that kind are non-deductive, so making conclusions like in the example can be called a 'guessing game'; however, the conclusions are not contingent: we have some reasons that make us accept rather these conclusions than others. In spite of the 'non-classicality' of the ways we make these conclusions, no matter how many new premises we get, there are still preserved all conclusions that had been made classically, in the deductive way.

The main task for formal research on such arguments is just a reconstruction of formal reasons that encourage us to make uncertain conclusions. However, we do not mean singular commonsense arguments, but inherent logical systems that include them as — in some non-classical sense — correct.

Since we do not want to analyze singular commonsense arguments, but integral logical systems that respect all classically correct arguments, so to name these logics we would like to use the term *cognitive logics*.[5] Hence, by *cognitive logics* we understand logics that are defined on the same language as CPL, preserve all classically correct arguments, and simultaneously may respect some classically invalid arguments.

3. Consequence operations: classical and supraclassical

We will define logics on the classical, propositional language. The alphabet consists of propositional letters $\mathsf{SI} = \{p_1, q_1, r_1, p_2, q_2, r_2, \ldots\}$, logical constants $\mathsf{Con} = \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$, and brackets:), (. Usually, instead of small letters with indexes, we will write bare letters: p, q, r, s, without subscripts, etc.

The set of formulas For is the smallest set of expressions that contains SI and is closed under Con in the following way: if $A, B \in \mathsf{For}$, than $\neg A \in \mathsf{For}$ and $(A * B) \in \mathsf{For}$, where $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$.

In the paper we approach logics in a way that is widely established in the literature on supraclassical logics. We use a consequence operation/relation notion to capture the particular logics we examine, which is typical for the Polish School of Logic[6] — to emphasize: we consequently identify a logic with consequence operation/relation. The notion was introduced by Alfred Tarski (Tarski, 1936) with a few restrictions, like for example monotony.

However, in the context of supraclassical logics, usually a more liberal notion is applied (Makinson, 2005). So we can start from some very general notion of consequence operation, partially reminding of basic concepts from (Wójcicki, 1988) that we complicate later.

Definition 2 (Consequence operation/relation)

A consequence operation is a function $C\colon P(\mathsf{For}) \longrightarrow P(\mathsf{For})$. *A consequence relation* is a relation $R \subseteq P(\mathsf{For}) \times \mathsf{For}$.

When we examine more than one consequence operation, we use C with subscripts. Instead of R we will write $\vdash$, respectively also with subscripts.

It is obvious that notions of consequence operation and consequence relation are interdefinable.

Corollary 3

Any consequence operation C is definable in terms of a consequence relation $\vdash$ (and vice versa) as follows: $A \in C(X)$ iff $X \vdash A$, for all $X \cup \{A\} \subseteq \mathsf{For}$.

Now, we present a handful of definitions and conclusions that are useful for the further considerations. We say that consequence operation C_1 *is stronger than* consequence operation C_2 iff for all $X \subseteq \mathsf{For}$: $C_2(X) \subseteq C_1(X)$.

Corollary 4

The relation of *being stronger than* defined on any set of consequence operations is a partial order, i.e. the relation is reflexive, transitive, and antisymmetric, so for any consequence relations C_1, C_2, C_3:

- C_1 is stronger than C_1
- If C_1 is stronger than C_2 and C_2 is stronger than C_3, then C_1 is stronger than C_3
- If C_1 is stronger than C_2 and C_2 is stronger than C_1, then $C_1 = C_2$.

Since the relation of *being stronger than* is a partial order, so we will write $C_1 \geq C_2$ or $C_2 \leq C_1$, when C_1 is stronger than C_2.[7]

We say that consequence operation C is *monotonic* (in short (M)) iff for all $X, Y \subseteq \mathsf{For}$: if $X \subseteq Y$, then $C(X) \subseteq C(Y)$. Consequence operation C is *non-monotonic* iff C is not monotonic. The notion of being monotonic/non-monotonic usually is crucial for any examination of commonsense reasoning.

Clearly, all the given notions can be rewritten in terms of consequence relation.

3.1. Classical consequence relation

CPL may be determined in many ways; one of the fundamental ways is the semantic approach. A valuation of propositional letters is a function: $v\colon \mathsf{Sl} \longrightarrow \{0,1\}$. Each valuation v can be extended to Boolean valuation of formulas $V\colon \mathsf{For} \longrightarrow \{0,1\}$, that preserves Boolean meaning of logical connectives in Con.

In the standard way we define notions of *being satisfied* and *semantic consequence operation*: formula A *classically follows from* set of formulas X (in short: $X \models A$) iff for all valuations V, if for all $B \in X$, $V(B) = 1$, then $V(A) = 1$. Classical consequence relation $\models$ is defined in set $P(\mathsf{For}) \times \mathsf{For}$, so it combines sets of premises with conclusions, this way determining the set of all valid arguments.

As we already know relation $\models$ can be expressed by consequence operation $Cn\colon P(\mathsf{For}) \longrightarrow P(\mathsf{For})$[8], called *classical consequence operation*, given as follows: $A \in Cn(X) \Leftrightarrow X \models A$. Surely, Cn is monotonic, so: for all $X, Y \subseteq \mathsf{For}$, if $X \subseteq Y$, then $Cn(X) \subseteq Cn(Y)$. Classical operation Cn is also *reflexive*, which means that $X \subseteq Cn(X)$, for all sets of formulas X. Moreover, it is *idempotent*, so $Cn(Cn(X)) \subseteq Cn(X)$, for all sets of formulas X. The three properties make Cn *closure operation*, and any operation that satisfies them is a *closure operation*.

Now, we add two important definitions more. Let C be a consequence operation. Operation C is *supraclassical* iff $Cn \leq C$. Operation C is *trivial* iff $C(X) = \mathsf{For}$, for all $X \subseteq \mathsf{For}$. Of course, there is only one trivial logic, namely C_{TRIV}.

The weakest supraclassical logic is Cn, while the strongest supraclassical one is trivial logic C_{TRIV}. Due to the conceptualization we are able to precisely define what we mean by cognitive logic. A logic/consequence operation C is *cognitive* iff $Cn \leq C \leq C_{\mathsf{TRIV}}$, but $Cn \neq C \neq C_{\mathsf{TRIV}}$. This is a formal and precise conceptualization of the definition of cognitive logic that we introduced informally in the previous section.

There are infinitely many logics between Cn and C_{TRIV}. None of the logics is trivial, since they are not identical to C_{TRIV}, however each of them preserves valid arguments — since they are stronger then classical logic Cn — and add some new argument or arguments that is classically invalid, since $Cn \neq C$, for any cognitive logic C. Some of the logics may be monotonic, as well as the rest are non-monotonic. The latter ones are intensively explored in literature (Antoniou, 1997), (Makinson, 2005).

There are a few mechanisms that allow making CPL stronger, a real cognitive logic. We will come back to them in the last section. Now, we are preparing for application of some mechanism to CPL.

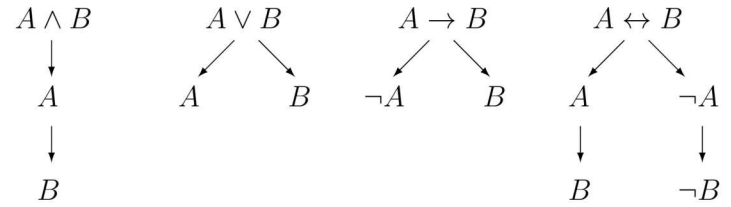
3.2. Tableau approach to CPL

Tableau method relies on the validity of given argument $\{P_1, \ldots, P_n, C\}$ being checked in the following way. We take the premises with negated conclusion $\{P_1, \ldots, P_n, \neg C\}$ and we apply some rules — called *tableau rules* — to decompose those formulas. During decomposing there can appear alternative possibilities — called *branches*. If on any branch there appear two formulas of the form: A, $\neg A$, the argument is valid, otherwise after we have used all applicable tableau rules, it is not valid.

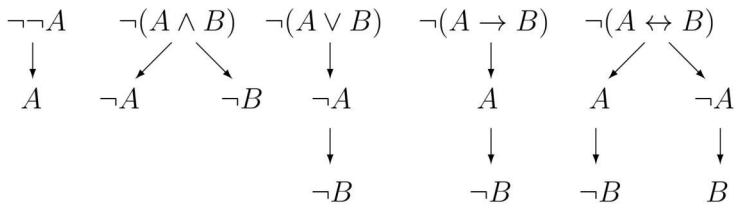
Tableau method is so analytic — we decompose formulas to get simpler ones — and indirect — we try to show that assumption on true premises and false conclusion is false itself.

3.2.1. Standard approach to tableau rules

Below we have some standard — as graphs — presentation of positive tableau rules for CPL:



And all negative (for formulas with an external negation) tableau rules are here:



3.2.2. Formal approach to tableau methods

In works (Jarmużek, 2013), (Jarmużek, 2013b), (Jarmużek & Tkaczyk, 2015), (Jarmużek & Tkaczyk, 2015a) we proposed a formalization of tableau methods for CPL, as well as for other propositional logics. Here we outline the main ideas of formalization of tableau rules, since the rules play a crucial role in any tableau proof. One distinguishing feature of our approach is that our tableau rules have an internal mechanism for blocking applications to closed and to maximal branches. Another point is that in our formalization we have a clear hierarchy of ontological levels of set theory beings: tableau rules and branches are sets of sets, while tableaus are sets

of branches, so sets of sets of sets. Among others, due to this our approach is precise.

Because in our approach we do not apply tableau rules to some kind of inconsistent sets of formulas, we introduce a definition of tableau inconsistency. Set $X \subseteq \mathsf{For}$ is *tableau inconsistent* (in short: *t-inconsistent*) iff $\exists_{A \in \mathsf{For}}\ A, \neg A \in X$. Set X is *tableau consistent* (in short: *t-consistent*) iff it is not t-inconsistent.

By *tableau rule* in general we understand a set of binary or ternary sequences of subsets of $P(\mathsf{For})$. The first element of any sequence we call *initial set* or *input set*, while the rest of the elements are called *output sets*. The very important thing in the presented approach to rules is that inputs are always proper subsets of outputs, so we can not use rules trivially. Furthermore, we will put pressure on that property.

Tableau rules can be written also as fractions, where in the numerator is an input set, in the denominator output set/sets. Now, we write the already mentioned rules for CPL, assuming that inputs sets are proper subsets of appropriate output sets:

$R_{\wedge}$: $\{\langle X \cup \{(A \wedge B)\}, X \cup \{(A \wedge B),\ A,\ B\}\rangle : X \cup \{(A \wedge B)\}$ is t-consistent$\}$

$R_{\vee}$: $\{\langle X \cup \{(A \vee B)\}, X \cup \{(A \vee B),\ A\},\ X \cup \{(A \vee B),\ B\}\rangle :$ $X \cup \{(A \vee B)\}$ is t-consistent$\}$

$R_{\rightarrow}$: $\{\langle X \cup \{(A \rightarrow B)\}, X \cup \{X \cup \{(A \rightarrow B),\ \neg A\},$ $X \cup \{(A \rightarrow B),\ B\}\rangle : X \cup \{(A \rightarrow B)\}$ is t-consistent$\}$

$R_{\leftrightarrow}$: $\{\langle X \cup \{(A \leftrightarrow B)\}, X \cup \{(A \leftrightarrow B),\ A,\ B\},$ $X \cup \{(A \leftrightarrow B),\ \neg A,\ \neg B\}\rangle : X \cup \{(A \rightarrow B)\}$ is t-consistent$\}$

$R_{\neg\neg}$: $\{\langle X \cup \{\neg\neg A\}, X \cup \{\neg\neg A,\ A\}\rangle : X \cup \{\neg\neg A\}$ is t-consistent$\}$

$R_{\neg\wedge}$: $\{\langle X \cup \{\neg(A \wedge B)\}, X \cup \{\neg(A \wedge B), \neg A\},\ X \cup \{\neg(A \wedge B), \neg B\}\rangle :$ $X \cup \{\neg(A \wedge B)\}$ is t-consistent$\}$

$R_{\neg\vee}$: $\{\langle X \cup \{\neg(A \vee B)\}, X \cup \{\neg(A \vee B), \neg A, \neg B\}\rangle : X \cup \{\neg(A \vee B)\}$ is t-consistent$\}$

$R_{\neg\rightarrow}$: $\{\langle X \cup \{\neg(A \rightarrow B)\}, X \cup \{\neg(A \rightarrow B),\ A,\ \neg B\}\rangle : X \cup \{\neg(A \rightarrow B)\}$ is t-consistent$\}$

$R_{\neg\leftrightarrow}$: $\{\langle X \cup \{\neg(A \leftrightarrow B)\}, X \cup \{\neg(A \leftrightarrow B),\ \neg A,\ B\},$ $X \cup \{\neg(A \leftrightarrow B),\ A,\ \neg B\}\rangle : X \cup \{\neg(A \leftrightarrow B)\}$ is t-consistent$\}$

The set of all tableau rules for CPL we denote as $\mathbf{R}_{\mathsf{CPL}}$. To make the notation more lucid, we can use fractions, assuming that input sets are t-consistent.

$$R_{\wedge}: \quad \frac{X \cup \{(A \wedge B)\}}{X \cup \{(A \wedge B),\, A,\, B\}}$$

$$R_{\vee}: \quad \frac{X \cup \{(A \vee B)\}}{X \cup \{(A \vee B),\, A\},\; X \cup \{(A \vee B),\, B\}}$$

$$R_{\rightarrow}: \quad \frac{X \cup \{(A \rightarrow B)\}}{X \cup \{(A \rightarrow B),\, \neg A\},\; X \cup \{(A \rightarrow B),\, B\}}$$

$$R_{\leftrightarrow}: \quad \frac{X \cup \{(A \leftrightarrow B)\}}{X \cup \{(A \leftrightarrow B),\, A,\, B\},\; X \cup \{(A \leftrightarrow B),\, \neg A,\, \neg B\}}$$

$$R_{\neg\neg}: \quad \frac{X \cup \{\neg\neg A\}}{X \cup \{\neg\neg A,\, A\}}$$

$$R_{\neg\wedge}: \quad \frac{X \cup \{\neg(A \wedge B)\}}{X \cup \{\neg(A \wedge B), \neg A\},\; X \cup \{\neg(A \wedge B), \neg B\}}$$

$$R_{\neg\vee}: \quad \frac{X \cup \{\neg(A \vee B)\}}{X \cup \{\neg(A \vee B), \neg A, \neg B\}}$$

$$R_{\neg\rightarrow}: \quad \frac{X \cup \{\neg(A \rightarrow B)\}}{X \cup \{\neg(A \rightarrow B),\, A,\, \neg B\}}$$

$$R_{\neg\leftrightarrow}: \quad \frac{X \cup \{\neg(A \leftrightarrow B)\}}{X \cup \{\neg(A \leftrightarrow B),\, \neg A,\, B\},\; X \cup \{\neg(A \leftrightarrow B),\, A,\, \neg B\}}$$

Having a formal notion of tableau rule, in (Jarmużek, 2013), (Jarmużek, 2013a) by turns we define formally more complex tableau notions: a branch, an open/closed/complete branch, a tableau, an open/closed tableau, a complete tableau, and tableau consequence operation. All those notions depend on $\mathbf{R}_{\mathsf{CPL}}$.

For example, informally we can introduce some intuitions. A branch is a chain of sets $X_1 \subset \ldots \subset X_n$ generated by applications of rules in $\mathbf{R}_{\mathsf{CPL}}$. A branch is *maximal* iff no rule in $\mathbf{R}_{\mathsf{CPL}}$ can be applied to its last member.

A branch is *closed* iff its last set includes a t-inconsistent set. A tableau is ordered triple $\langle X, A, \Phi\rangle$, where $X \cup \{A\} \subseteq \mathsf{For}$, Φ is a set of branches starting from set $X \cup \{\neg A\}$ and satisfying some additional conditions.

Finally, formula A is *a tableau consequence* of set of formulas X (symbolically, $A \in C_T(X)$ or $X \rhd A$) iff there exists a closed tableau $\langle X, A, \Phi\rangle$, i.e. a tableau with a set of branches Φ that arose by application of the rules in $\mathbf{R}_{\mathsf{CPL}}$, all are closed and no branch can be added to Φ. The defined tableau consequence operation C_T is obviously identical to classical consequence:

Fact 5

$C_T(X) = Cn(X)$, for all $X \subseteq \mathsf{For}$.

Hence, by application of rules in $\mathbf{R}_{\mathsf{CPL}}$ we can check the validity of all arguments that are valid from the classical point of view. Additionally, operation C_T can be named as *tableau classical consequence operation*.

An interesting question is what happens when we introduce some additional, non-classical tableau rules to $\mathbf{R}_{\mathsf{CPL}}$?

4. From Tableau Classical Consequence Operation to cognitive logics

To obtain more accepted conclusions than CPL allows, we could define a consequence operation in such a way that we would have a stronger operation. In particular it may not respect monotony (M) and then we come into the realm of non-monotonic arguments. However, to preserve all classical arguments, new operations should be supraclassical.

A transition to supraclassicality or rejection of (M) may be done in a few separate ways. Each of them requires some additional constraints; however, the starting points are as follows. In book (Makinson, 2005) there are extensively described three ways we could:

1. add hidden premises that sometimes work in the background
2. add rules of deduction
3. distinguish some valuations in the set of all classical valuations.

The second way is about syntactic reinforcement of a deductive approach to CPL by non-classical rules of deduction. It is somehow similar to what we will propose, strengthening the tableau approach by non-classical tableau rules. The first two ways are syntactical — like our approach — they strengthen operation Cn by adding new rules of deduction or new premises.

Here we have some example of the first way.

Example 6

Let K be a non-empty set of formulas and let at least some of them be contingent, but none of them be counter-tautology. For all $\{A\} \cup X \subseteq \mathsf{For}$ we determine consequence relation: $X \models_K A$ iff $X \cup K \models A$. Relation $\models_K$ is supraclassical and if K is not closed under substitution, $\models_K$ is not trivial.

Relation $\models_K$ is monotonic. To remove that property we must apply an additional mechanism. We can, for example, define a new relation: $X \vdash_{K'} A$ iff $X \cup K' \models A$, for any $K' \subseteq K$, that is maximally consis-

tent with X. The defined consequence relation is supraclassical and non-monotonic. Moreover: $\models \subset \vdash_{K'} \subset \models_K$, for all K that is not a set of tautologies as well as counter-tautologies.[9]

Our approach to a modification of Cn by amplification of C_T is patterned on the three ways described by David Makinson (Makinson, 2005). We would like to add more tableau rules (like background premises or more deductive rules are added). This way turns out to be very new, since so far tableau methods have been not autonomic in this context, but they have served to formalize other supraclassical systems like Autoepistemic Logic or Reiter's Default Theories (Olivetti, 1999). It may seem strange but no research on adding non-classical tableu rules has been carried.

However, adding new tools (also tableau rules) we must be very careful. A new operation we get may be easily trivialised. This follows from some well known facts about CPL.

We know that Cn is a closure operation. Moreover, classical operation Cn is closed under substitution. A substitution is a function $\mathsf{s}\colon \mathsf{For} \longrightarrow \mathsf{For}$ that for all formulas A, B and connective $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ satisfies conditions:

$\mathsf{s}(\neg A) = \neg \mathsf{s}(A)$

$\mathsf{s}(A * B) = \mathsf{s}(A) * \mathsf{s}(B)$.

Being closed under substitution means that if $A \in Cn(X)$, then $\mathsf{s}(A) \in Cn(\mathsf{s}(X))$, for any formula A, set of formulas X and substitution s. As a part of logical folklore we have a theorem:

Theorem 7

There is no supraclassical closure operation C which is closed under substitution and $Cn \neq C \neq C_{\mathsf{TRIV}}$.

Here, we focus on another way, through tableau consequence operation by adding some non-classical tableau rules. Nonetheless, for technical reasons we will not consider rules with more than one output. Since we have rule for disjunction $R_\vee$, so by a combination or combinations of it with a new rule with disjunctive output we can have the same effect as we employed more complicated tableau rules.

A rule R is *structural* iff if $\langle X_1, X_2\rangle \in R$, then $\langle \mathsf{s}(X_1), \mathsf{s}(X_2)\rangle \in R$, for all X_1, $X_2 \subseteq \mathsf{For}$ and substitution s. However, by adding to $\mathbf{R}_{\mathsf{CPL}}$ non-classical and structural as well — closed under substitution — rules we could trivialize the consequence operation we would get. So, we cannot generally

add to $\mathbf{R}_{\mathsf{CPL}}$ a new rule R that fulfills the structurality condition. Especially, we cannot do it, if $X_1 \not\models A$, where $A \in X_2$, otherwise — according to theorem 7 — we would get the trivial logic C_{TRIV}. The new rules must be so limited to non-structural ones.

4.1. Non-classical tableau rules

We propose some types of non-classical rules that are not structural. Clearly, we assume that all their input sets are t-consistent. The proposed rules probably exhaust all possibilities. On the left column we present the fraction notation of rules, while on the right one the set theory notation.

Let $A \in \mathsf{For}$. We assume that formula A is neither a tautology nor a counter-tautology. Thus we can really have some new, non-classical conclusions, but simultaneously not all formulas can be proved. Surely, A does not belong to input sets, since our rules generally are non-jejune. Formula A can be introduced to a proof by the following types of rules.

(a) $\quad R_A : \quad \dfrac{X}{X \cup \{A\}} \qquad R_A = \{\langle X, X \cup \{A\}\rangle : X \subseteq \mathsf{For}\}$

A rule of (a)-type always introduces formula A to any open branch.

(b) $\quad R'_A : \quad \dfrac{X}{X \cup \{A\}} \qquad R'_A = \{\langle X, X \cup \{A\}\rangle\},$

for some fixed $X \subseteq \mathsf{For}$ such that $A \notin Cn(X)$. Due to this assumption formula A is really a new member of the proof that is independent of all former assumptions. Any rule of (b)-type always introduces formula A to such an open branch that the set of all formulas on the branch is X. Any (b)-rule is obviously a singleton.

(c) $\quad R_{Y,A} : \quad \dfrac{X \cup Y}{X \cup Y \cup \{A\}} \qquad R_{Y,A} = \{\langle X \cup Y, X \cup Y \cup \{A\}\rangle : X \subseteq \mathsf{For}\},$

for some fixed and nonempty $Y \subseteq \mathsf{For}$. A rule of (c)-type always introduces formula A to any such open branch that the set of all formulas on the branch includes set of formulas Y.

(d) $\quad R'_{Y,A} : \quad \dfrac{X \cup Y}{X \cup Y \cup \{A\}} \qquad R'_{Y,A} = \{\langle X \cup Y, X \cup Y \cup \{A\}\rangle\},$

for some fixed $X, Y \subseteq \mathsf{For}$ such that Y is nonempty and $A \notin Cn(X \cup Y)$. Similarly, due to this assumption formula A is really a new member of the proof that is independent of all former assumptions. Any rule of (d)-type always introduces formula A to such an open branch that the set of all

formulas on the branch is $X \cup Y$. A rule of (d)-type is like (c)-type, but also Y is fixed, so it is a singelton.

It is obvious that all rules of type (b), (c), (d) are reducible to some rules of type (a) as their proper subsets. Especially, any rule (b) is a singleton and a subset of some (a) rule and (c) rule. Any (d) rule is a singleton also, and is a subset of some (c) and (a) rule, that consists of such members $\langle X, X \cup \{A\}\rangle$ that $Y \subseteq X$. Finally, (b) and (d) rules are in fact identical — just singletons. So, for simplicity we will write only about (a), (b) and (c) rules.

We would like to analyze one more type of non-classical tableau rule. Their special feature is that inputs are always compared with some additional set Y that consists of formulas that sometimes may block adding A to a proof.

Let $A \in \mathsf{For}$. Here we also assume that formula A is neither a tautology nor a counter-tautology. Again due to this assumption we can really have some new, non-classical conclusions. Surely, A does not belong to input sets, since our rules generally are non-jejune. Formula A can be introduced to a proof by the following types of rules.

(e1) $R_{A,Z}: \quad \dfrac{X}{X \cup \{A\}}$, where $X \cap Z = \emptyset$

$$R_{A,Z} = \{\langle X, X \cup \{A\}\rangle : X \subseteq \mathsf{For} \text{ and } X \cap Z = \emptyset\},$$

for some fixed $Z \subseteq \mathsf{For}$. A rule of (e1)-type always introduces formula A to any open branch, if the branch does not consist of any formula of Z.

(e2) $R'_{A,Z}: \quad \dfrac{X}{X \cup \{A\}}$, where $X \cap Z = \emptyset$

$$R'_A = \{\langle X, X \cup \{A\}\rangle\}, \text{ where } X \cap Z = \emptyset,$$

for some fixed $X, Y \subseteq \mathsf{For}$, such that $A \notin Cn(X)$. Due to this assumption formula A is really a new member of the proof that is independent of all former assumptions. Any rule of (e2)-type always introduces formula A to such an open branch that the set of all formulas on the branch is X and X has got no formula in Z. Any (e2)-rule is obviously a singleton.

(e3) $R_{Y,A,Z}: \dfrac{X \cup Y}{X \cup Y \cup \{A\}}$, where $(X \cup Y) \cap Z = \emptyset$

$$R_{Y,A,Z} = \{\langle X \cup Y, X \cup Y \cup \{A\}\rangle : X \subseteq \mathsf{For} \text{ and } (X \cup Y) \cap Z = \emptyset\},$$

for some fixed $Y, Z \subseteq \mathsf{For}$, where Y is nonempty. A rule of (e3)-type always introduces formula A to any such open branch that the set of all formulas

on the branch includes set of formulas Y and the branch has no common formula with Z.

(e4) $R'_{Y,A}: \quad \dfrac{X \cup Y}{X \cup Y \cup \{A\}}$, where $(X \cup Y) \cap Z = \emptyset$

$$R'_{Y,A} = \{\langle X \cup Y, X \cup Y \cup \{A\}\rangle\}, \text{ where } (X \cup Y) \cap Z = \emptyset$$

for some fixed $X, Y, Z \subseteq \mathsf{For}$, such that Y is nonempty and $A \notin Cn(X \cup Y)$. Similarly, due to this assumption formula A is really a new member of the proof that is independent of all former assumptions. A rule of (e4)-type is like (e3)-type, but also X is fixed. Any (e4)-rule is obviously a singleton.

Analogically like in the former cases (a)–(d), any rule of type (e2), (e3), (e4) is also reducible to some rules of type (e1) as some proper subset. Especially, any rule (e2) is a singleton and a subset of some (e1) rule and (e3) rule. Any (e4) rule is a singleton also and is a subset of some (e3) and (e1) rule that consists of such members $\langle X, X \cup \{A\}\rangle$ that $Y \subseteq X$ and $X \cap Z = \emptyset$. Finally, (e2) and (e4) rules are in fact also identical — just singletons. So, for simplicity we write only about (e1), (e2) and (e3) rules. Moreover, rules (b) are identical to (e2). So, for simplicity we write only about (e1), (e3) rules. Afterward rules of (a)-type can be reduced to (e1) rules, and (c)-type to (e3) rules, when $Z = \emptyset$.

In the further part the set of all defined rules (e1), (e3), (a), (b), (c), will be denoted by $\mathbf{R}_{\mathsf{NCPL}}$. Through the reductions, we have a conclusion.

Corollary 8

For any non-classical tableau rule R in $\mathbf{R}_{\mathsf{NCPL}}$ there exists such a (e1)-type rule R' that $R \subseteq R'$.

Because rules (e1) and (e3) — as a special case of (e1) — are of great importance, we probably would like to know better how they work. We repeat the schema of (e1).

(e1) $R_{A,Z} = \{\langle X, X \cup \{A\} : X \subseteq \mathsf{For} \text{ and } X \cap Z = \emptyset\}$,
for certain $A \in \mathsf{For}$, $Z \subseteq \mathsf{For}$.

If set Z is non-empty, then it 'filters' all sets $X \subseteq \mathsf{For}$ so that to those sets that share formulas with X can not be added formula A. In particular in Z there may be some contradictory formulas to A (some or all), but then A can not be added to any X that consists of some of those formulas.

The mechanism is a generalization of which is present in the definition of non-monotonic consequence operations by non-contradiction condition

of premises with additional assumptions (example 6). Formulas in Z do not have to be condradictory to formula A, we can just find them undesirable in X, if we want to add A. That is why the mechanism of 'filtering' *modulo* Z is more general than mechanism of adding formula A to X, if A is not contradictory to X. In general, set Z in the case of rules (e1) and (e3) we call a *set of filters*.

4.2. Hierarchy of non-classical tableau rules

Let Z be a set of filters in rule (e1) or (e3). If Z is a non-empty set, we call the rule *non-redundant*. When rule (e1) (or (e3)) is non-redundant, then its set of filters really filters premises. Now we can present a hierarchy of non-classical tableau rules in one place.

Tableau 1

Hierarchy of non-classical tableau rules

Tableau rules of (e1)-type are divided into:	
(e1) non-redundant,	(e1) redundant, they are identical to (a)-rules,
their subsets are non-redundant (e3) rules,	their subsets are redundant (e3) rules, they are identical to (c)-type rules,
all rules (b), (d), (e2), (e4) are identical; they are singletons, proper subsets of all other rules.	

4.3. Cognitive consequence operations

Having a set of non-classical rules $\mathbf{R}_{\mathsf{NCPL}}$, we can define cognitive logics. Let $\mathbf{R}'_{\mathsf{NCPL}}$ be a non-empty subset of $\mathbf{R}_{\mathsf{NCPL}}$. Now, we define a new consequence operation.

Definition 9 (Tableau cognitive consequence operation)

Let A be a formula and X be a set of formulas. $A \in C_{T\mathbf{R}'_{\mathsf{NCPL}}}(X)$ iff there exists a closed tableau $\langle A, X, \Phi\rangle$, for some set of branches Φ made by applications of rules in $\mathbf{R}'_{\mathsf{NCPL}} \cup \mathbf{R}_{\mathsf{CPL}}$.

In definition 9 we assume the use of classical tableau rules as well as some non-classical ones. However, we require only one closed tableau, otherwise an order of application could determine closing or non-closing tableaus

(like in Reiter's approach an order of rules can determine the existence of extensions (Antoniou, 1997)).

Surely, since we can still use classical tableau rules, so the defined operations $C_{T\mathbf{R}'_{\mathsf{NCPL}}}$ are supraclassical, and even $Cn < C_{T\mathbf{R}'_{\mathsf{NCPL}}}$, for most sets $\mathbf{R}'_{\mathsf{NCPL}}$ they are cognitive logics.

Under what conditions are they cognitive? Let us focus at the moment on (a) and (c) rules. Surely, formula A that is new in an output must not be a counter-tautology or a tautology, which we have assumed. Moreover, A should not follow classically from all inputs of a given rule. In the case of rules of (a)-type it happens, since their inputs change. The problem may appear in applications of (c) rules, since the fixed set Y could be contradictory and then we would always add A trivially. So, in the case of (c) rules fixed Y must not be contradictory. If the constant set of inputs Y is not contradictory, we call a rule (c) *non-contradictory*. Thus we have a corollary:

Corollary 10

If $\mathbf{R}'_{\mathsf{NCPL}}$ consists of only rules of types:
— type (a),
— type (c) which are non-contradictory,
— type (a) and type (c),
then operation $C_{T\mathbf{R}'_{\mathsf{NCPL}}}$ defined by $\mathbf{R}'_{\mathsf{NCPL}} \cup \mathbf{R}_{\mathsf{CPL}}$ is monotonic and cognitive.

We analyze one example.

Example 11

We take rule $R^{\neg p} = \{\langle X, X \cup \{\neg p\}\rangle : X \subseteq \mathsf{For}\}$ — it is an (a)-type rule — and add it to classical rules $\mathbf{R}_{\mathsf{CPL}}$. By this we obtain set of rules $\mathbf{R}^{\neg p}_{\mathsf{CPL}}$.

We take set of premises $\{p \vee q\}$. We see that $q \notin Cn(\{p \vee q\})$, because there exists such valuation V that $V(q) = 0$ and $V(p) = 1$. So the premise may be true, while the conclusion false.

However, if we define a new consequence operation by set of rules $\mathbf{R}^{\neg p}_{\mathsf{CPL}}$ — let us denote it by $C_{T\mathbf{R}^{\neg p}_{\mathsf{CPL}}}$ — then $q \in C_{T\mathbf{R}^{\neg p}_{\mathsf{CPL}}}(\{p \vee q\})$. It happens because applying rules of $\mathbf{R}^{\neg p}_{\mathsf{CPL}}$ to $\{p \vee q, \neg q\}$ in all branches we obtain t-inconsistency, since to all branches the new rule $R^{\neg p}$ enables us to introduce formula $\neg p$.

A consequence operation defined by rule $R^{\neg p} = \{\langle X, X \cup \{\neg p\}\rangle : X \subseteq \mathsf{For}\}$ and classical rules is monotonic. Taking supersets of $\{p \vee q\}$ does not revoke conclusion q, because rule $R^{\neg p} = \{\langle X, X \cup \{\neg p\}\rangle : X \subseteq \mathsf{For}\}$ enables us to add formula $\neg p$ to any t-consistent set of formulas. In cases of t-inconsistent

sets of premises we have the classical tableau rules that are sufficient to prove any formula. Surely, the consequence operation is also cognitive.

$R^{\neg p}$ is a rule of type (a). We can make it weaker, defining a suitable rule of (e1)-type. Thus we give another example.

Example 12

We take the rule:

$$R_p^{\neg p} = \{\langle X, X \cup \{\neg p\}\rangle : X \subseteq \mathsf{For} \text{ and } X \cap \{p\} = \emptyset\}.$$

By adding $R_p^{\neg p} = \{\langle X, X \cup \{\neg p\}\rangle : X \subseteq \mathsf{For} \text{ and } X \cap \{p\} = \emptyset\}$ to the classical tableau rules, we get $\mathbf{R}_{\mathsf{CPL}}^{\neg p,\{p\}}$. This way we obtain consequence operation $C_{T\mathbf{R}_{\mathsf{CPL}}^{\neg p,\{p\}}}$. Operation $C_{T\mathbf{R}_{\mathsf{CPL}}^{\neg p,\{p\}}}$ is cognitive and non-monotonic.

Returning to the former example, now we have $q \in C_{T\mathbf{R}_{\mathsf{CPL}}^{\neg p,\{p\}}}(\{p \vee q\})$, but $q \notin C_{T\mathbf{R}_{\mathsf{CPL}}^{\neg p,\{p\}}}(\{p \vee q, p\})$, since we can not apply $R_p^{\neg p}$ to $\{p \vee q, p\}$.

Hence for some $X, Y \subseteq \mathsf{For}$, $X \subseteq Y$, but $C_{T\mathbf{R}_{\mathsf{CPL}}^{\neg p,\{p\}}}(X) \not\subseteq C_{T\mathbf{R}_{\mathsf{CPL}}^{\neg p,\{p\}}}(Y)$.

Example 12 clearly shows that (e1)-type rules (and some subsets of them) may destroy the property (M). Considering now (e1)-type and (e3)-type rules we put the same questions as before: under what conditions are they cognitive?

Again, formula A that is new in an output must not be a counter-tautology or a tautology, which we have assumed. Moreover, A should not follow classically from all inputs of a given rule. In the case of rules of (e1)-type this happens, since their inputs change. The problem may appear in applications of (e3) rules, since the fixed set Y could be contradictory and then we would always add A trivially. So, in the case of (e3) rules fixed Y must not be contradictory. If the constant set of inputs Y is not contradictory, we call a rule (e3) *non-contradictory*. Here, however, a new problem appears. A set of filters might block all applications, when it was equal to For. So, each set of filters have to be a proper subset of For. It must not block at least one formula that is not a counter-tautology in an input set. By classical tableau rules counter-tautologies allow one to conclude any conclusion, so we do not need such rules. If filters in the rules of kinds (e1) or (e2) do not block at least one formula that is not a non-counter-tautology, we call them *non-trivial*.

The same is about (b) rules (and (d), (e2), (e4) rules — they are identical, as we know they are singletons). To apply them interestingly they should have non-contradictory inputs. Then we call them *non-contradictory*, too.

At the same time, we would like to have non-monotonic operations. To block some applications and in that way become non-monotonic, the (e1) and (e3) rules must be non-redundant. Thus we have a corollary:

Corollary 13

If $\mathbf{R}'_{\mathsf{NCPL}}$ consists of at least one rule of type:
- (e1)-type that is non-redundant and non-trivial,
- (e3)-type that is non-redundant, non-trivial and non-contradictory,
- (b)-type that is non-contradictory,

then operation $C_{T\mathbf{R}'_{\mathsf{NCPL}}}$ defined by $\mathbf{R}'_{\mathsf{NCPL}} \cup \mathbf{R}_{\mathsf{CPL}}$ is cognitive and non-monotonic.

Now, we can sum up the relationships between non-classical tableau rules and the cognitive consequence operations they determine. The rules of type (e1) and their proper subsets determine cognitive and non-monotonic consequence operations, if they are non-redundant and non-trivial; however, singelton subsets of (e1) must be non-contradictory. But if they are redundant, they and their (c)-types subsets that are non-contradictory determine cognitive and monotonic consequence operations.

Our hypothesis is that all supraclassical consequence operations (monotonic as well as non-monotonic) can be defined by some sets of non-classical tableau rules $\mathbf{R}'_{\mathsf{NCPL}}$ along with classical rules $\mathbf{R}_{\mathsf{CPL}}$. Hence, also all cognitive logics can be defined by the tableau methods presented here. But the hypothesis naturally requires some further examination.

5. Philosophical conclusions

Which of the proposed supraclassical, cognitive consequence operations is right? A probable answer is: none of them. In the case of commonsense reasoning we can only tend to one of the many options. On our choice decides an intuition, similarly as in the case of people who made conclusions incorrectly from the classical point of view, but the most reliably they could.

Where does the intuition come from? Is it a result of our everyday life experiences, our species' knowledge, our dower, or all of these things combined? — the question is about how to transcend the formal ways of representing cognitive reasoning.

NOTES

[1] The research presented in the following article was financed by National Science Centre, Poland, number of grant: 2015/19/B/HS1/02478.

[2] Instead of Classical Propositional Logic we will write in short: CPL.

[3] We obviously mean a language form of argumentation, so the form which is examined in logical research. In fact, any real argumentation can be carried in various environments: artificial, biological, psychological etc. However, in logic we deal with an intersubjective representation of arguments, which are linguistic counterparts of real arguments.

[4] Surely, by using the term *situation* we do not decide that in a truth theory we must have a ontological, epistemic or non-epistemic standpoint etc. In logic when we interpret a formal language, we use valuations/models, generally formal semantics/structures. The relationship between formal structures and the world we describe in a language is a philosophical issue.

[5] Honestly speaking, probably the first who used the term cognitive logic in the mentioned context was Jacek Malinowski. During the process of preparing of cognitive science studies at Nicolaus Copernicus University in Toruń in 2008, he proposed lectures about systems of various so called non-monotonic logics that he called *Cognitive Logic*. Since the first course started on the 1st of October 2009, so far the lectures have still been a part of the main core of that education.

[6] To make it clear, this kind of approach to the research on supraclassical, in particular, so called non-monotonic logics, was described as *deeply rooted in the Polish tradition, especially when we take into account a consequence operation, one of the most important notion that was invented in this tradition* by influential logician, also in a non-monotonic area, David Makinson. He wrote this in the preface to a Polish edition of his book *Bridges from Classical to Nonmonotonic Logic*, one of the most fluent modern synthesis of results in non-monotonic logics' domain (Makinson, 2008, p. IX).

[7] Obviously, when $C_1 \geq C_2$ or $C_2 \leq C_1$ we can also say that C_2 *is weaker than* C_1.

[8] Hence, we may use Cn and CPL interchangeably.

[9] It is what David Makinson in his mentioned book names *pivotal assumptions operation* and in a non-monotonic form *default assumptions operation*, see for details (Makinson, 2005, chapter 2).

REFERENCES

Antoniou Grigoris, *Nonmonotonic Reasoning*, MIT, 1997.

Jarmużek Tomasz, Tkaczyk Marcin, *A method of defining paraconsistent tableaus*, pp. 295–307, in *New Directions in Paraconsistent Logic*, J. Y. Beziau, M. Chakraborty and S. Dutta (eds.), vol. 152 of series *Springer Proceedings in Mathematics and Statistics*, Springer India, 2015.

Jarmużek Tomasz, *Formalizacja metod tablicowych dla logik zdań i logik nazw* (*Formalization of tableau methods for propositional logics and for logics of names*), Wydawnictwo UMK, Toruń, 2013.

Jarmużek Tomasz, Tkaczyk Marcin, *Modal paraconsistent tableau systems of logic*, WSEAS Transactions on Mathematics, vol. 14 (2015), pp. 248–255.

Jarmużek Tomasz, *Tableau Metatheorem for Modal Logics*, pp. 105–128, in *Recent Trends in Philosphical Logic, Trends in Logic*, (eds) Roberto Ciuni, Heinrich Wansing, Caroline Willkomennen, Springer Verlag, 2013.

Makinson David, *Bridges from Classical to Nonmonotonic Logic*, King's College 2005.

Makinson David, Przedmowa do polskiego wydania (Preface to Polish edition), p. IX, in *Od logiki klasycznej do niemonotonicznej*, D. Makinson, translated by Tomasz Jarmużek, Wydawnictwo UMK, Toruń 2008.

Olivetti Nicola, *Tableaux for Nonmonotonic Logics*, pp. 469–529, in D'Agostino M., Gabbay D., Haehnle R., Posegga J. (eds), *Handbook of Tableau Methods*, Kluwer, 1999.

Tarski Alfred, *On the concept of logical consequence*, pp. 409–420 in Alfred Tarski, *Logic, Semantics, Metamathematics Papers from 1923 to 1938*, translated by J.H. Woodger, Oxford: Clarendon Press 1956.

Wójcicki Ryszard, *Theory of Logical Calculi Basic Theory of Consequence Operations*, Springer 1988.