Wojciech Krysztofiak

Logiki drzew derywacyjnych

Filozofia Nauki 15/2, 33-73

2007

Filozofia Nauki
Rok XV, 2007, Nr 2(58)

Wojciech Krysztofiak

Logiki drzew derywacyjnych

Celem artykułu jest konstrukcja nowej klasy logik, które zostaną nazwane logikami drzew derywacyjnych (LDD-logiki).[1] Wyznaczają one kryteria poprawnego derywowania oraz transformowania struktur zwanych drzewami derywacyjnymi. LDD-logiki charakteryzują się pewną osobliwością syntaktyczną, którą można określić jako trójpoziomowość składniowa. Otóż, język konstruowanej logiki składa się z wyrażeń trzech poziomów syntaktycznych: (i) wyrażeń leksykalnych, (ii) wyrażeń derywacyjnych oraz (iii) wyrażeń transformacyjnych (wszystkie te kategorie traktowane są jako niefunktorowe kategorie syntaktyczne). Przedmiotem dowodzenia są wyrażenia transformacyjne, choć dowodami są odpowiednio zbudowane ciągi wyrażeń derywacyjnych lub leksykalnych. W „standardowo" rozumianej logice zarówno dowodzone obiekty językowe, jak i obiekty językowe, z których konstruowane są dowody, należą do tej samej kategorii składniowej (są formułami). W LDD-logikach taka sytuacja nie zachodzi. Ponadto, na gruncie LDD-logiki drzewa nie są traktowane jako struktury teoriomnogościowe, lecz jako wyrażenia językowe.

Inna motywacja dla konstrukcji LDD-logik posiada charakter ontologiczny. Otóż, za pomocą tych logik można opisać mechanizm syntetyzowania (prefabrykowania) rozmaitych typów przedmiotów w sensie Meinonga. Takie przedmioty mogą zostać potraktowane nie jako zbiory cech (własności), ale właśnie jako drzewa, których węzły są interpretowane jako cechy. Projektowana przez autora niniejszego artykułu ontologia derywacyjna przedmiotu pozwoli, na przykład, na wyróżnianie przedmiotów jednowymiarowych czy też wielowymiarowych semantycznie. Ponadto będzie można w jej świetle mówić o takich parametrach, jak głębokość prefabrykacji przedmiotu, jego wielo-światowość czy w końcu jego wielo-aspektowość. Dodatko-

[1] Autorowi nie są znane prace innych autorów opisujące tego rodzaju logiki.

wo teoria taka pozwoli na wyjaśnienie mechanizmów przekształcania obiektów fikcyjnych w inne obiekty tej kategorii.[2]

Struktura artykułu jest następująca: W części pierwszej opisana zostanie w języku nieformalnym inferencyjna struktura drzewa derywacyjnego na podstawie analizy schematu czynnościowego generującego operacje konstruowania drzew przez podmiot poznający. Druga część jest poświęcona konstrukcji języka formalnego, służącego do opisu drzew derywacyjnych. W części trzeciej jest przedstawiony LDD-rachunek logiczny. Czwarta część ma na celu opis podstawowych własności metalogicznych LDD-logiki. W szczególności zaprezentowane zostaną twierdzenia odnoszące się do relacji LDD-dowodliwości. Piąta część stanowić będzie ekspozycję teoriomnogościowego modelu semantycznego LDD-logiki. W zakończeniu zostaną określone rozmaite sposoby rozszerzania LDD-logiki, w taki sposób, że będzie można mówić o nieskończonej klasie LDD-logik.

1. INFERENCYJNA STRUKTURA DRZEWA DERYWACYJNEGO

Drzewa derywacyjne jako narzędzie modelowania rozmaitych empirycznych dziedzin badawczych zostały upowszechnione w głównej mierze przez przedstawicieli lingwistyki generatywnej. Oczywiście, wykorzystywane są one również w rozmaitych teoriach informatycznych czy też cybernetycznych (na przykład w teorii automatów). Za pomocą drzew można reprezentować równania matematyczne czy sekwencje ruchów figurami w dowolnej grze w szachy, ale również struktury organizacyjne instytucji. Dowody konstruowane przez logików także mogą być reprezentowane za pomocą drzew. Zadomowienie się w nauce drzew derywacyjnych jako narzędzi modelowania rozmaitych dziedzin badania wskazuje na ważny fakt. Otóż, użytkownik języka, po przejściu właściwego treningu operacyjnego, jest w stanie nauczyć się posługiwania się drzewami derywacyjnymi bez znajomości teorii tego typu struktur. Student językoznawstwa humanistycznego nie musi znać matematycznej teorii gramatyk generatywnych, aby móc konstruować sprawnie drzewa derywacyjne reprezentujące tak zwane struktury głębokie wypowiedzi języka potocznego. I tak samo inteligentny petent nie musi mieć ukończonych studiów z zakresu teorii algorytmów, aby zrozumieć schemat organizacyjny dowolnego urzędu miejskiego przedstawiony za pomocą drzewa.[3] W antropologii kognitywnej wskazuje się na to,

[2] W pracy: [Krysztofiak 2006a] została skonstruowana teoria na bazie pewnej LDD-logiki opisująca mechanizmy generowania i przekształcania architektonik systemów filozoficznych. W pracy pokazano aplikację tej teorii do systemów filozofii przedsokratejskiej. Zob. dyskusję na temat aplikacji skonstruowanej teorii: [Pawliszcze 2006], [Krysztofiak 2006b].

[3] Drzewa derywacyjne są strukturami zapisu rozmaitych danych (informacji). Można nieco bardziej filozoficznie powiedzieć, iż drzewa są uniwersalnymi sposobami modelowania (reprezentowania) zbiorów danych, czyli kwantów informacji. Na temat drzew zob. [Harel 2001, s. 55-61], [Bolter 1990, s. 135-144]. Bolter w języku filozoficznym próbuje pokazać, że drzewa stanowią sposoby reprezentowania rozmaitych przestrzeni logicznych, służących programistom komputerowym

że drzewa (struktury taksonomiczne) stanowią sposoby porządkowania przez umysły rozmaitych dziedzin semantycznych.[4]

Przedstawione fakty można próbować wyjaśnić poprzez odwołanie się do istnienia kompetencji logicznej w generowaniu drzew derywacyjnych,[5] przy czym taki rodzaj kompetencji aktywowałby się we wczesnej fazie rozwoju umysłu ludzkiego.[6] Tak, jak nie trzeba znać klasycznego rachunku zdań, żeby przeprowadzać poprawne na gruncie tego rachunku rozumowania zdaniowe, tak też nie trzeba znać algebraicznej teorii drzew, aby móc poprawnie je konstruować i przekształcać.

Jeśli przedstawiona hipoteza o istnieniu kompetencji logicznej w generowaniu drzew jest zasadna, to powinna istnieć logika drzew derywacyjnych, stanowiąca system kompetencyjny, regulujący mechanizmy ich generowania oraz przekształcania. Logika taka, podobnie jak gramatyka uniwersalna Chomsky'ego, funkcjonowa-

---

do rozwiązywania ich problemów.

[4] Zob. [Buchowski 1993]. Drzewa jako struktury taksonomiczne są pojmowane jako formy pojęciowe, narzucane przez uczestników danej kultury na percypowaną rzeczywistość.

[5] Pojęcie kompetencji logicznej stanowi rezultat ekstrapolacji pojęcia kompetencji lingwistycznej na dziedzinę rozmaitych czynności logicznego przetwarzania wypowiedzi językowych (czyli czynności wnioskowania). Macnamara określa kompetencję logiczną jako „źródło informacji dla intuicji logicznej". Intuicję zaś charakteryzuje jako zdolność do oceny „logicznej roli składników zdań albo poprawności wnioskowań" [zob. Macnamara 1993, s. 58]. Autor ten nie wymienia zdolności do generowania i rozumienia struktur drzewiastych jako składnika kompetencji logicznej. W niniejszej pracy przez kompetencję logiczną rozumiem ogół zdolności umysłu ludzkiego, manifestujących się w rozmaitych czynnościach przetwarzania rozmaitych struktur (językowych, czynnościowych czy modelowych). Logicznie myślimy nie tylko wtedy, kiedy przeprowadzamy wnioskowania, ale również kiedy przekształcamy pewne struktury modelowe na inne struktury modelowe, np. kiedy połowiąc kwadrat przekątną prefabrykujemy dwa trójkąty.

[6] Piaget twierdzi, że zdolności logiczne (rozumiane jako dyspozycja do manipulowania na podstawowych formach zdaniowych: implikacji, negacji, koniunkcji i alternatywie) aktywują się poprzez doświadczenie dopiero w późnej fazie rozwoju umysłu, w okresie preadolescencji [zob. Piaget, Inhelder, s.126-142]. Z tego punktu widzenia zdolność do generowania drzew derywacyjnych nie stanowiłaby umiejętności logicznej; należałaby ona raczej do kategorii „myślenia konkretnego". Ta kategoria obejmuje operacje na konkretach, a nie na formach artykułowalnych językowo. Epistemologowie genetyczni zaliczają do tej kategorii takie operacje, na przykład, jak: szeregowanie, włączanie do klas, klasyfikowanie, rozmieszczanie czy w końcu liczenie [zob. Piaget, Inhelder, s. 94-105]. Jednakże owe dyspozycje „myślenia konkretnego" stanowią podstawę rozwojową do nabycia przez podmiot kompetencji logicznej: „Operacje konkretne stanowią zatem przejście między działaniem i bardziej ogólnymi strukturami logicznymi […]"[zob. Piaget, Inhelder, s.97]. To stanowisko jest skrytykowane przez Macnamarę: „[…] nie istnieją dane empiryczne mówiące, iż dzieci na pewnym etapie rozwoju nie potrafią zrozumieć logicznej mocy najbardziej podstawowych funktorów zdaniowych, takich jak *i*" [Macnamara 1993, s. 177]. Nie jest istotne, czy zdolność do generowania drzew zostanie określona jako logiczna czy też protologiczna. Posługując się językiem epistemologów genetycznych, można określić zdolność do generowania drzew jako rozwijającą się dyspozycję: od fazy umiejętności rozmieszczania przedmiotów w przestrzeni projekcyjnej zregionalizowanej według dwóch osi: lewo — prawo; góra — dół, do fazy umiejętności abstrahowania form operacyjnych (drzewiastych) od rozmaitych układów czynności rozmieszczania.

łaby na poziomie komputacyjnym umysłu, jako „nieuświadomiona wiedza”.[7] Oczywiście, nie znaczy to wcale, że taka zdolność do generacji i transformacji struktur drzewiastych stanowi fundamentalny rozwojowo element kompetencji logicznej. Być może ufundowanie się w umyśle takiej zdolności wymaga akwizycji jeszcze bardziej „pierwotnych” struktur operacyjnych.

Jakie więc czynności inferencyjne wykonuje użytkownik języka w trakcie generacji drzewa derywacyjnego? Pierwszą czynnością jest wyróżnienie jakiegoś elementu początkowego drzewa derywacyjnego. Taki element określany jest jako wierzchołek drzewa. Następnie według określonych reguł z wierzchołka są derywowane jakieś elementy (niekoniecznie nowe). Z kolei z tych elementów, także w oparciu o pewien zestaw reguł, derywowane są elementy (również niekoniecznie nowe). Proces tego rodzaju derywacji jest prowadzony aż do momentu, w którym z elementów wyprowadzony jest jakiś specjalny element należący do kategorii elementów nieposiadających mocy generatywnej (zwykle elementy takie są określane jako elementy typu *stop*). Uogólniając, na każdy proces generacji drzewa składają się trzy typy czynności inferencyjnych: (i) czynność wyboru wierzchołka wyznaczająca uniwersum derywacji wierzchołkowych, (ii) czynności derywacji z elementów posiadających moc generatywną elementów również posiadających moc generatywną, (iii) czynności derywacji z elementów o jakiejś mocy generatywnej innych elementów bez mocy generatywnej. Tego rodzaju struktura czynnościowa operuje na następujących obiektach: wierzchołku i węzłach oraz gałęziach; przy czym niektóre węzły nie posiadają mocy generatywnej.

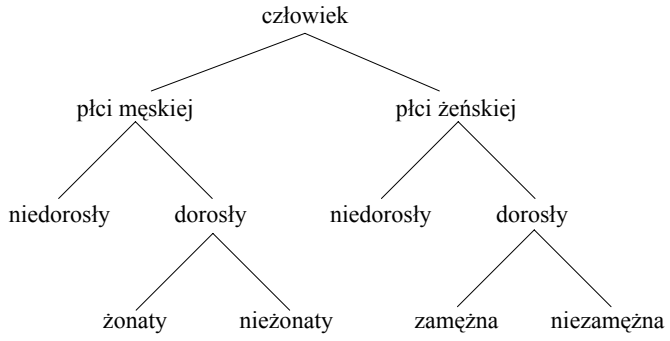
Przekształcanie drzew na inne drzewa jest zawsze przeprowadzane zgodnie z jakimiś regułami, zwykle formułowanymi w „wizualizacyjnej manierze językowej”.[8] Regułom tym odpowiadają rozmaite operacje algebraiczne. Znajomość tych reguł nie jest koniecznym warunkiem sprawnego posługiwania się drzewami derywacyjnymi

[7] Chomsky zwraca uwagę na to, że „[…] dziecko dysponuje wrodzoną i na tyle bogatą […] teorią potencjalnych opisów strukturalnych […], iż jest ono zdolne określić, które opisy strukturalne mogą być stosowne […]” dla emitowanej lub odbieranej wypowiedzi. Zob. [Chomsky 1982, s. 54]. LDD-logika ma być właśnie taką teorią generowania „drzewiastych” opisów strukturalnych (tj. niekoniecznie stanowiących struktury głębokie wypowiedzi językowych). Rzecz jasna nie zakłada się tego, że dziecko jest w stanie rozpisać nawet proste składniowo zdanie w formie drzewa derywacyjnego. Chomsky ma na myśli to, że opis strukturalny zdania, dokonywany przez dziecko, daje się ująć za pomocą modelu drzewa. To, że dziecko nie jest w stanie „narysować” drzewa zdaniowego, znaczyłoby tylko tyle, że nie jest ono w stanie zaimplementować swojego opisu strukturalnego (drzewiastego) w materiale graficznym. Należy więc odróżniać kompetencję do generowania i transformowania drzew derywacyjnych od dyspozycji do ich implementacji w dowolnym tworzywie materialnym.

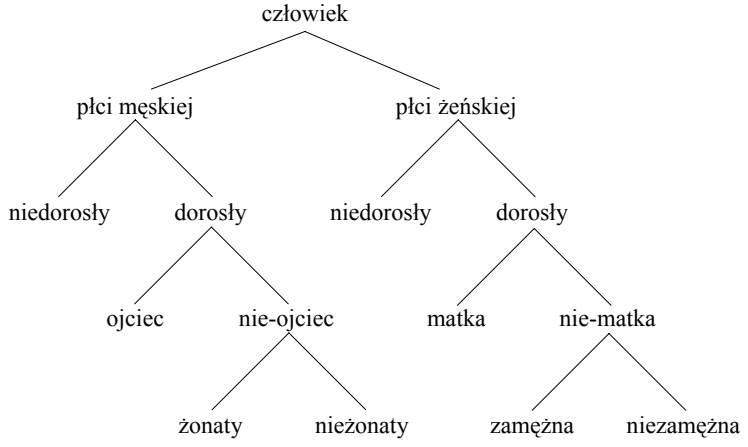
[8] Na przykład, takimi regułami mogą być: (i) zamień gałęzie występujące na takich a takich pozycjach; (ii) obetnij gałąź występującą na takiej a takiej pozycji; (iii) skróć gałąź występującą na takiej a takiej pozycji w taki a taki sposób. Często tego typu instrukcjami kierują się studenci, kiedy uczą się rozmaitych sposobów rekonstruowania drzew derywacyjnych reprezentujących struktury głębokie wypowiedzi języka potocznego.

w różnego rodzaju praktykach językowych. Każdy kompetentny użytkownik języka bez trudu oszacuje, że między następującymi drzewami derywacyjnymi (reprezentującymi struktury klasyfikacyjne) zachodzi relacja transformowalności. Oto te drzewa:

(1)



(2)



Drzewo derywacyjne (2) powstaje z drzewa (1) w wyniku zastosowania następujących instrukcji: (i) wydłuż w drzewie (1) gałęzie prowadzące: od **dorosły** do **nieżonaty** oraz od **dorosły** do **niezamężna**, poprzez wprowadzenie ogniwa pośredniego w obu gałęziach: **nie-ojciec** oraz **nie-matka**; (ii) wykreśl w drzewie (1) gałęzie prowadzące: od **dorosły** do **żonaty** oraz od **dorosły** do **zamężna**; (iii) po wykonaniu czynności (i) dodaj do tak otrzymanej struktury gałęzie prowadzące: od **nie-ojciec** do **żonaty** oraz od **nie-matka** do **zamężna**; (iv) dodaj w drzewie (1) — po wykonaniu czynności (i) — gałęzie prowadzące: od **dorosły** (pod **płci męskiej**) do **ojciec** oraz od **dorosły** (pod **płci żeńskiej**) do **matka**. Użytkownik języka, kierując się wyszczególnionymi instrukcjami, jest w stanie przekształcić drzewo (1) w drzewo (2), a także ocenić to, czy dowolne drzewo może być uzyskane wedle powyższych instrukcji

z drzewa (1). Przedstawione drzewa reprezentują pewne struktury teoriomnogościowe; węzły reprezentują zbiory, gałęzie zaś — relacje inkluzji zachodzące pomiędzy poszczególnymi zbiorami. I otóż znajomość teorii mnogości nie jest w tym wypadku koniecznym warunkiem przekształcenia drzewa (1) na drzewo (2).

Podsumowując: w trakcie generacji drzewa użytkownik języka dokonuje czynności inferencji, określonych na elementach słownikowych drzewa, czyli węzłach i wierzchołku (budując poszczególne jego gałęzie), w trakcie zaś transformowania drzewa — dokonuje czynności inferencji, określonych na gałęziach. Te wnioski sugerują, że składnia języka, w którym użytkownik generuje drzewa derywacyjne, posiada co najmniej trzy poziomy syntaktyczne: (i) poziom leksykalny, na który składają się wyrażenia oznaczające węzły oraz wierzchołek w drzewie; (ii) poziom derywacyjny, na który składają się wyrażenia oznaczające gałęzie drzewa; (iii) poziom architektoniczny, na który składają się wyrażenia oznaczające drzewa.

2. JĘZYK LOGIKI DRZEW DERYWACYJNYCH

Na alfabet języka LDD składają się następującego typu wyrażenia: (1) zmienne oznaczające węzły (zmienne leksykalne); (2) stała ***stop*** oznaczająca „końcówki" drzewa, a więc te pozycje, które w danym drzewie nie posiadają mocy generatywnej; (3) funktor derywacji tworzący ze zmiennych oznaczających węzły lub stałej ***stop*** wyrażenia oznaczające gałęzie drzewa; (4) funktor transformacji tworzący z wyrażeń oznaczających gałęzie drzewa wyrażenia oznaczające sposoby przekształcania gałęzi drzew na inne gałęzie.

2.1. Zmienne leksykalne i stała *stop*

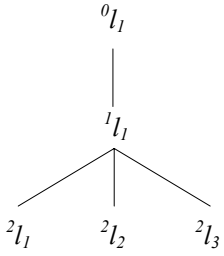
Każdemu węzłowi w drzewie derywacyjnym można przyporządkować poziom derywacyjny oznaczający odległość danego węzła od wierzchołka drzewa. I tak na przykład w drzewie (1) (powyżej) węzeł **niedorosły** jest dalej oddalony od wierzchołka **człowiek** niż węzeł **płci męskiej**. Oba węzły: **niedorosły** oraz **dorosły** występują na tym samym poziomie derywacyjnym, gdyż ich odległość od wierzchołka drzewa jest taka sama. Dlatego też zmienne leksykalne oznaczające węzły są zmiennymi indeksowanymi; lewy górny indeks będzie właśnie oznaczał poziom derywacyjny węzłów, reprezentowanych przez zmienną.

W każdym drzewie węzły znajdujące się na tym samym poziomie derywacyjnym różnią się pozycją swojego występowania w drzewie. Ponadto może zdarzyć się tak, że w danym drzewie dwa różne węzły są zajmowane przez ten sam element. Taka sytuacja ma miejsce w drzewie (2) powyżej. Element **dorosły** występuje dwa razy w tym drzewie. Do formalnego opisu takich sytuacji nie wystarczy więc użycie jednej zmiennej leksykalnej dla danego poziomu; właśnie wielość zmiennych leksykal-

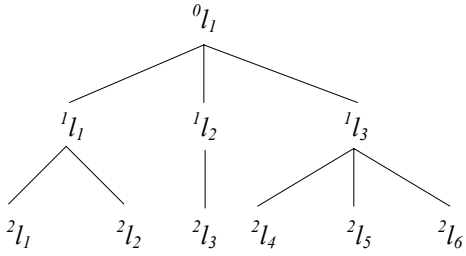
nych ma reprezentować wielość pozycji na danym poziomie derywacyjnym w danym drzewie.

Przyjmijmy więc następujące konwencje terminologiczne: (1) zmienne reprezentujące wierzchołki drzew są kształtu: ${}^0l_1$, ${}^0l_2$, ${}^0l_3$, ${}^0l_4$,…, ${}^0l_i$,… itd.; (2) zmienne reprezentujące węzły n-tego poziomu derywacyjnego są kształtu: ${}^nl_1$, ${}^nl_2$, ${}^nl_3$, ${}^nl_4$,…, ${}^nl_i$,… itd. Na przykład, poprawnie opisanymi fragmentami drzew będą następujące struktury:

(3)

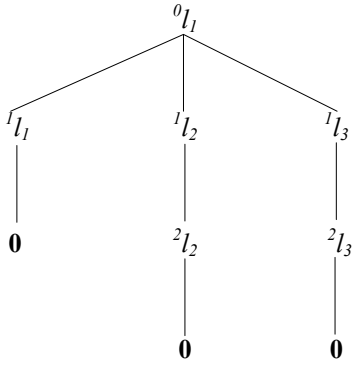


(4)



Stała ***stop*** oznacza te pozycje w dowolnym drzewie, z których nie są już generowane żadne inne węzły. Pozycje te odznaczają się brakiem mocy generatywnej w danym drzewie. Przyjmijmy konwencję, zgodnie z którą „**0**” oznacza właśnie stałą ***stop***. Stała „**0”** nie może więc pojawić się w drzewie w węzłach środkowych (niebędących wierzchołkiem) danego drzewa. Może się jednak pojawiać na różnych poziomach derywacyjnych w danym drzewie. Znaczy to, że odległość elementu **0** od wierzchołka w danym drzewie może się różnić w zależności od poziomu, na którym występuje. Niech następujący przykład drzewa stanowi egzemplifikację zauważonej sytuacji:

(6)



W drzewie (6) element **0** dominowany przez ${}^1l_1$ występuje na drugim poziomie derywacyjnym, podczas gdy w pozostałych dwóch sytuacjach — na trzecim poziomie derywacyjnym.

2.2. Funktor derywacji

Funktor derywacji „**[**…**]**” tworzy ze zmiennej leksykalnej ${}^{i}l_{k}$ oraz skończonej listy (zbioru) różnych zmiennych (tego samego poziomu derywacyjnego) ${}^{n}l_{j},\ldots, {}^{n}l_{m}$ lub ze stałej **0**, takich że $n \geq i$, wyrażenia reprezentujące gałęzie drzew derywacyjnych. Wyrażenia takie będą określane jako wyrażenia derywacyjne, obiekty zaś reprezentowane przez nie — jako derywacje. Następujące napisy stanowią przykłady wyrażeń derywacyjnych: „${}^{0}l_{1}$**[**${}^{n}l_{1}$, …, ${}^{n}l_{i}$**]**”, „${}^{m}l_{i}$**[0]**”. Wyrażenie przed nawiasem będzie określane jako wyrażenie argumentowe (lub w skrócie — jako argument wyrażenia derywacyjnego), lista zaś elementów leksykalnych w nawiasie będzie nazywana terminem „wartość wyrażenia derywacyjnego”. Przy czym w wypadku listy kolejność wyszczególniania jej składników nie ma znaczenia. Definicje: argumentu oraz wartości wyrażenia derywacyjnego, przedstawiają się następująco:

(Df. 1) $(\forall d, \alpha, \beta_1, \beta_n)[d = \alpha\mathbf{[}\beta_1,\ldots, \beta_n\mathbf{]} \rightarrow \Omega(d) = \alpha]$.

(Df. 2) $(\forall d, \alpha, \beta_1, \beta_n)[d = \alpha\mathbf{[}\beta_1,\ldots, \beta_n\mathbf{]} \rightarrow \Omega^{*}(d) = \{\beta_1,\ldots, \beta_n\}]$.

Wyrażenie: „${}^{0}l_{1}$**[**${}^{n}l_{1}$, …, ${}^{n}l_{i}$**]**” może być interpretowane na różne sposoby: (i) proceduralno-algorytmicznie, (ii) teorio-funkcyjnie, (iii) inferencyjnie. Zgodnie z (i), analizowane wyrażenie jest instrukcją algorytmiczną o postaci: wyprowadź derywacyjnie z elementu ${}^{0}l_{1}$ listę elementów ${}^{n}l_{1}$, …, ${}^{n}l_{i}$. Wyraża więc pewną procedurę konstrukcyjną. W świetle teorio-funkcyjnej interpretacji (ii), wyrażenie „${}^{0}l_{1}$**[**${}^{n}l_{1}$, …, ${}^{n}l_{i}$**]**” wyraża to, że funkcja derywacji dla elementu ${}^{0}l_{1}$ przyjmuje wartość będącą zbiorem elementów $\{{}^{n}l_{1}, \ldots, {}^{n}l_{i}\}$. W wypadku inferencyjnego rozumienia wyrażeń derywacyjnych, termin „${}^{0}l_{1}$**[**${}^{n}l_{1}$, …, ${}^{n}l_{i}$**]**” wyraża to, iż z elementu ${}^{0}l_{1}$ jest wyprowadzalny derywacyjnie zbiór elementów $\{{}^{n}l_{1}, \ldots, {}^{n}l_{i}\}$.

Wyrażenia derywacyjne podlegają syntaktycznej operacji superpozycji. Jeśli więc α**[**$\beta_1,\ldots, \beta_i,\ldots, \beta_n$**]**, β_i**[**$\gamma_1,\ldots, \gamma_k$**]** są wyrażeniami derywacyjnymi, to wówczas superpozycja tych wyrażeń o postaci: α**[**$\beta_1,\ldots, \beta_i$**[**$\gamma_1,\ldots, \gamma_k$**]**,…, β_n**]**, jest również wyrażeniem derywacyjnym. Na przykład, „${}^{0}l_{1}$**[**${}^{1}l_{1}$, ${}^{1}l_{2}$**[**${}^{2}l_{1}$**]]**” jest wyrażeniem derywacyjnym, gdyż takimi są: „${}^{0}l_{1}$**[**${}^{1}l_{1}$, ${}^{1}l_{2}$**]**” oraz „${}^{1}l_{2}$**[**${}^{2}l_{1}$**]**”. W interpretacji proceduralno-algorytmicznej, superponowane wyrażenia derywacyjne można interpretować jako koniunkcje instrukcji derywacyjnych. Dlatego też wyrażenie o postaci: α**[**$\beta_1,\ldots, \beta_i$**[**$\gamma_1,\ldots, \gamma_k$**]**,…, β_n**]**, można „czytać” jako koniunkcję następujących instrukcji: (i) wyprowadź derywacyjnie z elementu α listę elementów $\beta_1,\ldots, \beta_i,\ldots, \beta_n$ oraz (ii) wyprowadź z elementu β_i listę elementów $\gamma_1,\ldots, \gamma_k$. Niektóre z superponowanych wyrażeń derywacyjnych reprezentują formy architektoniczne drzew derywacyjnych. Na przykład, drzewo derywacyjne z diagramu (4), powyżej, będzie reprezentowane przez wyrażenie o postaci: „${}^{0}l_{1}$**[**${}^{1}l_{1}$**[**${}^{2}l_{1}$, ${}^{2}l_{2}$**]**, ${}^{1}l_{2}$**[**${}^{2}l_{3}$**]**, ${}^{1}l_{3}$**[**${}^{2}l_{4}$, ${}^{2}l_{5}$, ${}^{2}l_{6}$**]]**”. Wyrażenie to jest koniunkcją czterech instrukcji algorytmicznych, na mocy których drzewo z diagramu (4) jest generowane.

Zdefiniowanie kategorii superponowanych wyrażeń derywacyjnych wymaga posłużenia się operacją podstawiania wyrażeń za wyrażenia. Niech $\alpha(\beta_1,\ldots, \beta_k \mid \eta_1, \ldots\eta_k)$ będzie wyrażeniem powstającym z wyrażenia α w wyniku podstawienia w nim wyrażeń $\beta_1,\ldots, \beta_k$, odpowiednio, wyrażeniami $\eta_1, \ldots\eta_k$. Wówczas definicja kategorii superponowanych wyrażeń derywacyjnych przedstawia się następująco:

(Df. 3) $\alpha \in \boldsymbol{Der}_{sup} \equiv (\exists\beta_1,\ldots, \beta_k)[\beta_1,\ldots, \beta_k \in \boldsymbol{Der} \wedge \beta_1,\ldots, \beta_k \in \Omega^*(\alpha) \wedge \alpha(\beta_1,\ldots, \beta_k \mid \Omega(\beta_1),\ldots, \Omega(\beta_k)) \in \boldsymbol{Der}]$.

Ponieważ superponowane wyrażenia derywacyjne są interpretowane jako „skompresowane zapisy" list wyrażeń derywacyjnych, więc można zdefiniować następującą funkcję dekompresji $\boldsymbol{\Delta}$, przekształcającą superponowane wyrażenie derywacyjne w listę korespondujących niesuperponowanych wyrażeń derywacyjnych (funkcja dekompresji zostanie użyta w zdefiniowaniu funkcji LDD-wartościowania). Zauważmy, że każde wyrażenie derywacyjne składa się z argumentu i wartości, a wartość tego wyrażenia jest listą wyrażeń leksykalnych lub derywacyjnych, taką że kolejność występowania wyrażeń na liście nie wpływa w żaden sposób na właściwości logiczne danego wyrażenia derywacyjnego. Zatem dowolne wyrażenie derywacyjne należące do $\boldsymbol{Der}$ lub $\boldsymbol{Der}_{sup}$ można zapisać w postaci standardowej następująco: $\alpha\boldsymbol{[}\beta_1, \ldots, \beta_n, \beta_{n+1}\boldsymbol{[}\delta_1,\ldots, \delta_i\boldsymbol{]}, \ldots, \beta_{n+k}\boldsymbol{[}\chi_1,\ldots, \chi_{i]}\boldsymbol{]}$, gdzie $\beta_1, \ldots, \beta_n$ są wyrażeniami leksykalnymi (zmiennymi lub stałymi), $\beta_{n+1}\boldsymbol{[}\delta_1,\ldots, \delta_i\boldsymbol{]}, \ldots, \beta_{n+k}\boldsymbol{[}\chi_1,\ldots, \chi_{i]}$ są zaś wyrażeniami derywacyjnymi (superponowanymi lub niesuperponowanymi).

(Df. 4) (i) $\alpha\boldsymbol{[}\beta_1,\ldots, \beta_n\boldsymbol{]} \in \boldsymbol{Der} \rightarrow \boldsymbol{\Delta}(\alpha\boldsymbol{[}\beta_1,\ldots, \beta_n\boldsymbol{]}) = \{\alpha\boldsymbol{[}\beta_1,\ldots, \beta_n\boldsymbol{]}\}$;

(ii) $\alpha\boldsymbol{[}\beta_1, \ldots, \beta_n, \beta_{n+1}\boldsymbol{[}\delta_1,\ldots, \delta_i\boldsymbol{]}, \ldots, \beta_{n+k}\boldsymbol{[}\chi_1,\ldots, \chi_{i]}\boldsymbol{]} \in \boldsymbol{Der}_{sup} \rightarrow \boldsymbol{\Delta}(\alpha\boldsymbol{[}\beta_1, \ldots, \beta_n, \beta_{n+1}\boldsymbol{[}\delta_1,\ldots, \delta_i\boldsymbol{]}, \ldots, \beta_{n+k}\boldsymbol{[}\chi_1,\ldots, \chi_{i]}\boldsymbol{]}) = \{\alpha\boldsymbol{[}\beta_1, \ldots, \beta_n, \beta_{n+1}, \ldots, \beta_{n+k}\boldsymbol{]}\} \cup \boldsymbol{\Delta}(\beta_{n+1}\boldsymbol{[}\delta_1,\ldots, \delta_i\boldsymbol{]}) \cup \ldots \cup \boldsymbol{\Delta}(\beta_{n+k}\boldsymbol{[}\chi_1,\ldots, \chi_i\boldsymbol{]})$.

Operacja dekompresji przekształca niesuperponowane wyrażenie derywacyjne w zbiór jednoelementowy złożony z tego właśnie wyrażenia. Z kolei dekompresja superponowanego wyrażenia derywacyjnego daje w wyniku zbiór niesuperponowanych wyrażeń derywacyjnych superpozycyjnie konstytuujących to pierwsze. Taki zbiór będzie określany jako zakres dekompresji danego wyrażenia derywacyjnego.

Niesuperponowane wyrażenia derywacyjne można różnicować z uwagi na parametr głębokości, który przyporządkowuje wyrażeniu derywacyjnemu o postaci „$^{k}l_1\boldsymbol{[}^{n}l_1, \ldots, {}^{n}l_i\boldsymbol{]}$" różnicę pary dwóch liczb $< k, n >$ taką, że k reprezentuje poziom derywacyjny argumentu wyrażenia derywacyjnego, n zaś — poziom derywacyjny wartości wyrażenia derywacyjnego. O wyrażeniu derywacyjnym „$^{k}l_1\boldsymbol{[}^{n}l_1, \ldots, {}^{n}l_i\boldsymbol{]}$" będzie się mówiło, że opisuje derywację z poziomu k na poziom n. Jeśli różnica $n - k > 1$, to wyrażenie derywacyjne o głębokości wyznaczonej przez parę $<k, n>$ opisuje derywację z przeskokiem derywacyjnym. Jeśli zaś $n - k = 1$, to wówczas wyrażenie derywacyjne opisuje derywację bez przeskoku derywacyjnego. Natomiast w wypadku, gdy $n - k = 0$, wyrażenie derywacyjne opisuje derywację wewnątrzpoziomową.

Oczywiście, niemożliwa jest definicyjnie sytuacja, w której $n - k < 0$. Definicja niesuperponowanych wyrażeń derywacyjnych o wartości parametru głębokości $\boldsymbol{n}$ przedstawia się następująco (niech symbole: „$^{0}\boldsymbol{L}$”, „$^{1}\boldsymbol{L}$”,…, „$^{n}\boldsymbol{L}$” oznaczają zbiory wyrażeń leksykalnych, kolejno, poziomu zerowego, pierwszego oraz n-tego):

(Df. 5) $d \in \boldsymbol{Der^{n}} \equiv \{d \in \boldsymbol{Der} \wedge (\Omega^{*}(d) = \{\text{`}\mathbf{0}\text{'}\} \rightarrow \boldsymbol{n} = \boldsymbol{1}) \wedge (\forall \boldsymbol{i}, \boldsymbol{j})[\Omega(d) \in {}^{i}\boldsymbol{L} \wedge \Omega^{*}(d) \subset {}^{j}\boldsymbol{L} \wedge \Omega^{*}(d) \neq \{\text{`}\mathbf{0}\text{'}\} \rightarrow \boldsymbol{j} - \boldsymbol{i} = \boldsymbol{n}]\}$.

Zgodnie z (Df. 5) wyrażenia derywacyjne, których wartością jest stała **stop** są derywacjami typu $\boldsymbol{Der^{1}}$.

Można również przyporządkowywać superponowanym wyrażeniom derywacyjnym parametr głębokości:

(Df. 6) $\alpha \in \boldsymbol{Der^{n}_{sup}} \equiv [\alpha \in \boldsymbol{Der_{sup}} \wedge (\exists\beta)(\beta \in \boldsymbol{\Delta}(\alpha) \wedge \beta \in \boldsymbol{Der^{n}}) \wedge (\forall\beta)[(\forall \boldsymbol{k})[\beta \in \boldsymbol{\Delta}(\alpha) \wedge \beta \in \boldsymbol{Der^{k}} \rightarrow \boldsymbol{n} \geq \boldsymbol{k}]]]$

Zgodnie z (Df. 6), parametr głębokości przyporządkowuje dowolnemu superponowanemu wyrażeniu derywacyjnemu wartość maksymalną ze zbioru wartości tegoż parametru od wyrażeń derywacyjnych należących do zakresu dekompresji superponowanego wyrażenia derywacyjnego.

Specjalną klasę superponowanych wyrażeń derywacyjnych wyznaczają te, których zakresy dekompresji składają się wyłącznie z wyrażeń derywacyjnych o parametrze głębokości jeden. Nazwijmy je standardowymi, superponowanymi wyrażeniami derywacyjnymi:

(Df. 7) $\alpha \in \boldsymbol{Der^{stand}_{sup}} \equiv [\alpha \in \boldsymbol{Der_{sup}} \wedge (\forall\beta)[\beta \in \boldsymbol{\Delta}(\alpha) \rightarrow \beta \in \boldsymbol{Der^{1}}]]$.

2.3. Funktor transformacji

Funktor transformacji „[…//…]” tworzy z listą wyrażeń derywacyjnych lub wyrażeń leksykalnych zerowego poziomu derywacyjnego $\alpha_1,\ldots, \alpha_n$ oraz z listą wyrażeń derywacyjnych $\beta_1,\ldots, \beta_i$ wyrażenie o postaci: $[\alpha_1,\ldots,\alpha_n // \beta_1,\ldots,\beta_i]$. Tego rodzaju wyrażenia, określane jako wyrażenia transformacyjne, posiadają w konstruowanym języku status formuł. Wyrażenie: $[\alpha_1,\ldots,\alpha_n // \beta_1,\ldots,\beta_i]$ wyraża to, że z listy wyrażeń derywacyjnych lub wyrażeń leksykalnych poziomu początkowego $\alpha_1,\ldots,\alpha_n$ są transformacyjnie wyprowadzalne wyrażenia derywacyjne $\beta_1,\ldots,\beta_i$ (albo w interpretacji przedmiotowej — to, że zbiór derywacji reprezentowanych przez listę wyrażeń derywacyjnych $\alpha_1,\ldots,\alpha_n$ lub początkowych elementów leksykalnych jest transformowalny na zbiór derywacji reprezentowanych przez listę $\beta_1,\ldots,\beta_i$). Można więc wyróżnić dwa szczególne typy formuł transformacyjnych: (i) takich, że argument jest listą wyłącznie wyrażeń derywacyjnych oraz (ii) takich, że argument jest listą wyrażeń leksykalnych zerowego poziomu derywacyjnego. Ta sytuacja mogłaby sugerować, że funktor „//” posiada odmienny indeks kategorialny w obu typach formuł

transformacyjnych. Tak jednak nie jest, gdyż wiąże on listy wyrażeń, a nie poszczególne wyrażenia. Listy wyrażeń zaś oznaczają zbiory odpowiednich obiektów oznaczanych przez wyrażenia składające się na dane listy.

Niech litery: $d_1,\ldots,d_n,\ldots$ itd., będą metazmiennymi reprezentującymi wyrażenia derywacyjne. Wówczas wyrażenia o postaci: $[d_1,\ldots,d_n \,//\, d_i,\ldots,\, d_k]$ będą metanazwami formuł transformacyjnych języka LDD-logiki. Oto przykłady formuł transformacyjnych: „$[{}^0l_1\mathbf{[0]} \,//\, {}^0l_1[{}^1l_1,\, {}^1l_2]$, ${}^0l_1[{}^1l_1,\, {}^1l_2,\, {}^1l_3]]$”, „$[{}^0l_1\mathbf{[0]} \,//\, {}^0l_1[{}^1l_1[{}^2l_1,\, {}^2l_2]$, ${}^1l_2[{}^2l_3]$, ${}^1l_3[{}^2l_4,\, {}^2l_5,\, {}^2l_6]]]$”. Lewa część formuły transformacyjnej będzie określana jako jej argument, prawa zaś część — jako jej wartość.

Formuły transformacyjne można różnicować z uwagi na liczebność ich argumentów oraz liczebność wartości. Najprostszymi syntaktycznie (czyli: elementarnymi) formułami transformacyjnymi są te, które są jednocześnie jednoargumentowe oraz jednowartościowe i których wyrażenia składowe nie są superponowanymi wyrażeniami derywacyjnymi. Żaden z dwóch podanych wyżej przykładów nie należy do kategorii elementarnych formuł transformacyjnych. Przykładem takiej elementarnej formuły jest: „$[{}^0l_1\mathbf{[0]} \,//\, {}^0l_1[{}^1l_1,\, {}^1l_2]]$”. Zarówno argument, jak i jej wartość są jednoelementowymi listami niesuperponowanych wyrażeń derywacyjnych. Wyróżnić można następujących typów formuły transformacyjne: jednoargumentowe, jednowartościowe, wieloargumentowe, wielowartościowe, z superponowanymi derywacjami (w argumencie lub we wartości) i bez superponowanych derywacji. Elementarne formuły transformacyjne charakteryzują się zarówno jednoargumentowością, jak i jednowartościowością; są kształtu: $[\alpha_n \,//\, \beta_i]$.

W odróżnieniu od wyrażeń derywacyjnych, formuły transformacyjne posiadają status dowodzonych obiektów językowych. Wyrażenia derywacyjne nie podlegają procedurze dowodowej w tym znaczeniu, że nie można ich dowodzić. Dowodzeniu podlegają formuły transformacyjne.

3. KONSTRUKCJA LOGIKI DRZEW DERYWACYJNYCH

Reguły inferencji definiujące logikę drzew derywacyjnych są instrukcjami wyznaczającymi zbiór dowodliwych formuł transformacyjnych. Pierwotne reguły inferencji wyznaczają zbiór aksjomatycznych formuł transformacyjnych. Skoro formuły transformacyjne wyrażają to, że pewien zbiór wyrażeń derywacyjnych lub wyrażeń leksykalnych poziomu zerowego jest transformowalny na pewien inny zbiór wyrażeń derywacyjnych, to dowodliwość formuł transformacyjnych będzie oznaczała to, że pewien zbiór wyrażeń derywacyjnych lub wyrażeń leksykalnych poziomu zerowego jest transformowalny na pewien inny zbiór wyrażeń derywacyjnych. Niedowodliwe formuły transformacyjne będą więc błędnie wyrażały to, że pewien zbiór wyrażeń derywacyjnych lub wyrażeń leksykalnych poziomu zerowego jest transformowalny na pewien inny zbiór wyrażeń derywacyjnych.

3.1. Reguły pierwotne

Pierwotne reguły inferencji LDD-logiki można podzielić na dwie grupy: (i) reguły umożliwiające transformowanie formuł derywacyjnych o danym argumencie na formuły derywacyjne, których argument należy do tego samego poziomu leksykalnego, do którego należy argument wyrażenia transformowanego; (ii) reguły pozwalające na przekształcenie formuł derywacyjnych o danym argumencie lub elementów leksykalnych poziomu zerowego na formuły derywacyjne, których argument przynależy do poziomu leksykalnego o jeden większy, niż poziom leksykalny, do którego należy argument wyjściowej formuły derywacyjnej. Pierwsze będą określane mianem pierwotnych reguł horyzontalnych, drugie zaś — mianem pierwotnych reguł wertykalnych. Pierwotne horyzontalne reguły inferencji LDD-logiki są „odpowiedzialne" za mechanizm rozbudowy drzew derywacyjnych w poziomie, pierwotne zaś wertykalne reguły inferencji wyznaczają mechanizm rozbudowy drzew derywacyjnych w pionie. Mówiąc metaforycznie, horyzontalne reguły inferencji są „odpowiedzialne" za przekształcanie pęków gałęzi w drzewach derywacyjnych w bardziej lub mniej złożone pęki gałęzi (a więc za wycinanie lub implementowanie gałęzi w pękach), natomiast wertykalne reguły wyznaczają mechanizmy rozgałęziania się lub atrofii drzew. Lista pierwotnych reguł inferencji LDD-logiki przedstawia się następująco:

Pierwotne horyzontalne reguły inferencji

Reguła multiplikacji: (Mult) ${}^{n}\alpha[{}^{n+1}\beta_1,\ldots,{}^{n+1}\beta_k]$ // ${}^{n}\alpha[{}^{n+1}\beta_1,\ldots,{}^{n+1}\beta_k, {}^{n+1}\beta_{k+1}]$.
Reguła eliminacji: (Elim) ${}^{n}\alpha[{}^{n+1}\beta_1,\ldots,{}^{n+1}\beta_k]$ // ${}^{n}\alpha[{}^{n+1}\beta_1,\ldots,{}^{n+1}\beta_{k-1}]$, dla $k \geq 2$.
Reguła introdukcji: (Intr) ${}^{n}\alpha[\mathbf{0}]$ // ${}^{n}\alpha[{}^{n+1}\beta]$.
Reguła destrukcji: (Dest) ${}^{n}\alpha[{}^{n+1}\beta]$ // ${}^{n}\alpha[\mathbf{0}]$.

Zgodnie z regułą multiplikacji (Mult), jeśli w drzewie jest dany węzeł dający początek jakiemuś pękowi gałęzi, to zawsze można wprowadzić nową gałąź do danego pęku. Zgodnie z regułą eliminacji (Elim), dany pęk gałęzi biorących początek z danego węzła zawsze można zredukować. Reguła introdukcji (Intr) pozwala na zastąpienie dowolnego pustego elementu w danej derywacji na odpowiedni niepusty element leksykalny. I odwrotnie reguła destrukcji (Dest) umożliwia zastąpienie pojedynczej gałęzi biorącej początek z danego węzła na gałąź z pustym elementem leksykalnym.

Pierwotne wertykalne reguły inferencji

Reguła startu: (Start) ${}^{0}\alpha$ // ${}^{0}\alpha[\mathbf{0}]$.
Reguła tranzycji: (Trans) ${}^{n}\alpha[{}^{n+1}\beta]$ // ${}^{n+1}\beta[\mathbf{0}]$.
Reguła unifikacji: (Unif) ${}^{n}\alpha[{}^{n+1}\beta_1,\ldots,{}^{n+1}\beta_j,\ldots,{}^{n+1}\beta_k]$, ${}^{n+1}\beta_j[{}^{n+2}\delta_1,\ldots,{}^{n+2}\delta_i]$ // ${}^{n}\alpha[{}^{n+1}\beta_1,\ldots,{}^{n+1}\beta_j[{}^{n+2}\delta_1,\ldots,{}^{n+2}\delta_i],\ldots,{}^{n+1}\beta_k]$, o ile spełniony jest warunek:

$(\forall\tau)[\tau \in \boldsymbol{\Delta}\ ({}^{n}\alpha[{}^{n+1}\beta_{1,\ldots}, {}^{n+1}\beta_{j,\ldots}, {}^{n+1}\beta_{k}]) \rightarrow \tau \in \boldsymbol{Der}^{I}] \wedge (\forall\tau)[\tau \in \boldsymbol{\Delta}\ ({}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,\ldots}, {}^{n+2}\delta_{i}] \rightarrow \tau \in \boldsymbol{Der}^{I}]$.

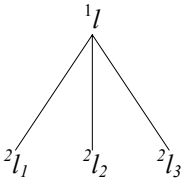
Reguła dezintegracji (Dezint): ${}^{n}\alpha[{}^{n+1}\beta_{1,\ldots}, {}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,\ldots}, {}^{n+2}\delta_{i}], \ldots, {}^{n+1}\beta_{k}]$ // ${}^{n}\alpha[{}^{n+1}\beta_{1,\ldots}, {}^{n+1}\beta_{j,\ldots}, {}^{n+1}\beta_{k}], {}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,\ldots}, {}^{n+2}\delta_{i}]$, o ile spełniony jest warunek:

$(\forall\tau)[\tau \in \boldsymbol{\Delta}\ ({}^{n}\alpha[{}^{n+1}\beta_{1,\ldots}, {}^{n+1}\beta_{j,\ldots}, {}^{n+1}\beta_{k}]) \rightarrow \tau \in \boldsymbol{Der}^{I}] \wedge (\forall\tau)[\tau \in \boldsymbol{\Delta}\ ({}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,\ldots}, {}^{n+2}\delta_{i}] \rightarrow \tau \in \boldsymbol{Der}^{I}]$.

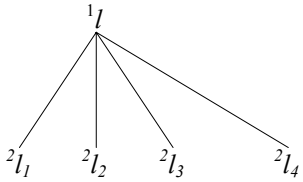
Reguła startu pozwala na wygenerowanie dowolnego wyrażenia derywacyjnego, którego argumentem jest wyrażenie leksykalne poziomu zerowego (oznaczające dowolny, początkowy element leksykalny). Reguła tranzycji umożliwia (wraz z regułą introdukcji) generację gałęzi derywacyjnych o dowolnej długości. Innymi słowy, jeśli dane jest pewne wyrażenie derywacyjne o jednoskładnikowej wartości, to można z niego wyprowadzić dowolne wyrażenie derywacyjne, którego argumentem jest wyrażenie leksykalne stanowiące ów składnik. I wreszcie reguła unifikacji zezwala na wyprowadzanie superponowanych wyrażeń derywacyjnych z odpowiednio dobranych wyrażeń derywacyjnych. Reguła unifikacji może być stosowana tylko wtedy, kiedy parametr głębokości przyporządkowuje argumentowym wyrażeniom derywacyjnym liczbę jeden. Unifikować można tylko wyrażenia derywacyjne, dla których parametr głębokości przyjmuje wartość jeden. Reguła dezintegracji pozwala na wyprowadzenie z superponowanego wyrażenia derywacyjnego niesuperponowanych wyrażeń derywacyjnych, ale tylko w sytuacji, w której parametr głębokości przyporządkowuje niesuperponowanym wyrażeniom derywacyjnym liczbę jeden. Innymi słowy, reguła (Dezint) pozwala z superponowanego wyrażenia derywacyjnego o określonym kształcie (na kształt wskazuje warunek aplikacji tej reguły) wyprowadzić jego zakres dekompresji. Nie da się zdezintegrować superponowanego wyrażenia derywacyjnego, które jest ukonstytuowane z wyrażeń derywacyjnych, którym parametr głębokości przypisuje wartości inne niż jeden.

Mechanizm funkcjonowania wyszczególnionych reguł można zilustrować następującymi diagramami. Niech w diagramach pierwszy rysunek reprezentuje argument reguły, drugi zaś rysunek — jej wartość. Diagram dla reguły (Mult) w pewnej wersji będzie przedstawiał się następująco:

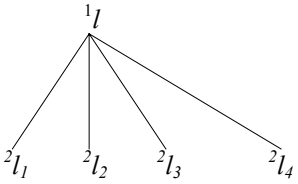
(1)

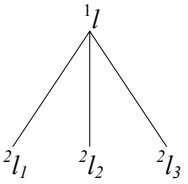


(2)

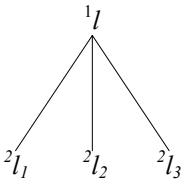


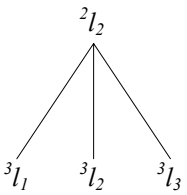
Reguła (Mult) pozwala na przekształcenie pęku gałęzi na rysunku (1) w pęk gałęzi na rysunku (2), w którym występuje jedna dodatkowa gałąź. Reguła (Elim) wyznacza operację odwrotną względem operacji wyznaczanej przez regułę (Mult).

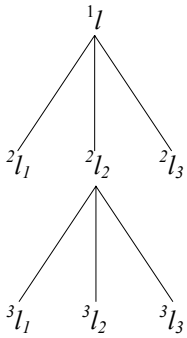
(1) 

(2) 

Na mocy reguły (Elim) w pęku gałęzi (1) wolno obciąć jedną gałąź. W podobny sposób można pokazywać „funkcjonowanie" pozostałych reguł horyzontalnych. Niech następujący diagram ilustruje mechanizm funkcjonowania reguły (Unif).

(1) 

(2) 

(3) 

Reguła (Unif) pozwala z dwóch gałęzi (1) i (2) wygenerować rozbudowaną wertykalnie gałąź (3). Mechanizm odwrotny względem mechanizmu wyznaczonego przez regułę (Unif) wyznacza reguła (Dezint). Z rysunku (3), na mocy reguły (Dezint), można wygenerować rysunki (1) i (2).

3.2. Pojęcie dowodu na gruncie LDD-logiki

Intuicyjnie, LDD-dowodami są zasadniczo ciągi wyrażeń derywacyjnych, które spełniają ustalone definicyjnie warunki. Natomiast tym, co jest dowodzone, są formuły transformacyjne. Wyrażenie o postaci: „LDD$\vdash$ [$\alpha_1,\ldots,\alpha_n$ // $\beta_1,\ldots,\beta_i$]" wyraża to, że formuła transformacyjna [$\alpha_1,\ldots,\alpha_n$ // $\beta_1,\ldots,\beta_i$] jest dowodliwa na gruncie LDD-logiki (czyli że istnieje jej dowód na gruncie tej logiki). Definicja pojęcia LDD-dowodu wymaga konstrukcji pojęć pomocniczych, oznaczających osobliwe klasy formuł transformacyjnych:

(Df. 8) $\tau \in$ (Mult) $\equiv (\exists$ n, $^n\alpha$, $^{n+1}\beta_1,\ldots,^{n+1}\beta_k$, $^{n+1}\beta_{k+1})$ $\tau = (^n\alpha[^{n+1}\beta_1,\ldots,^{n+1}\beta_k]$ // $^n\alpha[^{n+1}\beta_1,\ldots,^{n+1}\beta_k, ^{n+1}\beta_{k+1}])$.

(Df. 9) $\tau \in$ (Elim) $\equiv (\exists$ n, k $\geq$2, $^n\alpha$, $^{n+1}\beta_1,\ldots,^{n+1}\beta_k)$ $\tau = (^n\alpha[^{n+1}\beta_1,\ldots,^{n+1}\beta_k]$ //

$^{n}\alpha[^{n+1}\beta_{1,}\ldots,^{n+1}\beta_{k\,-1}]$).

(Df. 10) $\tau \in$ (Dest) $\equiv (\exists\, n, {}^{n}\alpha, {}^{n+1}\beta)\ \tau = ({}^{n}\alpha[{}^{n+1}\beta] \,//\, {}^{n}\alpha[\mathbf{0}])$.

(Df. 11) $\tau \in$ (Intr) $\equiv (\exists\, n, {}^{n}\alpha, {}^{n+1}\beta)\ \tau = ({}^{n}\alpha[\mathbf{0}] \,//\, {}^{n}\alpha[{}^{n+1}\beta])$.

(Df. 12) $\tau \in$ (Start) $\equiv (\exists\, \alpha)\ \tau = ({}^{0}\alpha \,//\, {}^{0}\alpha[\mathbf{0}])$.

(Df. 13) $\tau \in$ (Trans) $\equiv (\exists\, n, {}^{n}\alpha, {}^{n+1}\beta)\ \tau = ({}^{n}\alpha[{}^{n+1}\beta] \,//\, {}^{n+1}\beta[\mathbf{0}])$.

(Df. 14) $\tau \in$ (Unif) $\equiv (\exists\, n, {}^{n}\alpha, {}^{n+1}\beta_{1,}\ldots, {}^{n+1}\beta_{j,}\ldots, {}^{n+1}\beta_{k}, {}^{n+1}\beta_{j}, {}^{n+2}\delta_{1,}\ldots, {}^{n+2}\delta_{i})$ $[\tau = [{}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots, {}^{n+1}\beta_{j,}\ldots, {}^{n+1}\beta_{k}], {}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots, {}^{n+2}\delta_{i}] \,//\, {}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots, {}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots, {}^{n+2}\delta_{i}],\ldots, {}^{n+1}\beta_{k}]] \wedge (\forall\gamma)[\gamma \in \boldsymbol{\Delta}({}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots, {}^{n+1}\beta_{j,}\ldots, {}^{n+1}\beta_{k}]) \rightarrow \gamma \in \boldsymbol{Der}^{\boldsymbol{I}}] \wedge (\forall\gamma)[\gamma \in \boldsymbol{\Delta}({}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots, {}^{n+2}\delta_{i}]) \rightarrow \gamma \in \boldsymbol{Der}^{\boldsymbol{I}}]]$.

(Df. 15) $\tau \in$ (Dezint) $\equiv (\exists\, n, {}^{n}\alpha, {}^{n+1}\beta_{1,}\ldots, {}^{n+1}\beta_{j,}\ldots, {}^{n+1}\beta_{k}, {}^{n+1}\beta_{j}, {}^{n+2}\delta_{1,}\ldots, {}^{n+2}\delta_{i})$ $[\tau = ({}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots, {}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots, {}^{n+2}\delta_{i}],\ldots, {}^{n+1}\beta_{k}] \,//\, {}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots, {}^{n+1}\beta_{j,}\ldots, {}^{n+1}\beta_{k}], {}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots, {}^{n+2}\delta_{i}]) \wedge (\forall\gamma)[\gamma \in \boldsymbol{\Delta}({}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots, {}^{n+1}\beta_{j,}\ldots, {}^{n+1}\beta_{k}]) \rightarrow \gamma \in \boldsymbol{Der}^{\boldsymbol{I}}] \wedge (\forall\gamma)[\gamma \in \boldsymbol{\Delta}({}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots, {}^{n+2}\delta_{i}]) \rightarrow \gamma \in \boldsymbol{Der}^{\boldsymbol{I}}]]$.

Definicja pojęcia LDD-dowodu przedstawia się następująco:

(Df. 16) LDD$\vdash [\alpha_{1,}\ldots, \alpha_{n} \,//\, \beta_{1,}\ldots,\beta_{i}]$ wtedy i tylko wtedy, gdy istnieje ciąg wyrażeń $\lambda_{1,}\ldots, \lambda_{n,}\ldots, \lambda_{k}$ spełniający następujące warunki:

(1) każde z wyrażeń ciągu $\lambda_{1,}\ldots, \lambda_{n,}\ldots, \lambda_{k}$ należy do kategorii wyrażeń leksykalnych poziomu zerowego lub do kategorii wyrażeń derywacyjnych;

(2) $\lambda_{1} = \alpha_{1} \wedge \ldots \wedge \lambda_{n} = \alpha_{n}$ (czyli początkowe wyrazy ciągu $\lambda_{1,}\ldots, \lambda_{n,}\ldots, \lambda_{k}$ są identyczne z wyrażeniami składającymi się na argument dowodzonej formuły transformacyjnej);

(3) $(\forall\lambda_{j},\, n < j \leq k)(\exists\, \lambda_{i,}\, \lambda_{h},\, i,\, h < j)\ [(\lambda_{i},\, \lambda_{h} \,//\, \lambda_{j}) \in$ (Unif) $\vee\ (\lambda_{i} \,//\, \lambda_{h}, \lambda_{j}) \in$ (Dezint)$] \vee (\forall\lambda_{j},\, n < j \leq k)(\exists\, \lambda_{i,}\, i < j)\ [(\lambda_{i} \,//\, \lambda_{j}) \in$ (Mult) $\cup$ (Elim) $\cup$ (Dest) $\cup$ (Intr) $\cup$ (Start) $\cup$ (Trans)] (czyli każdy wyraz ciągu $\lambda_{1,}\ldots, \lambda_{n,}\ldots, \lambda_{k}$, który nie jest wyrazem początkowym, jest uzyskany z jakiegoś wyrazu wcześniejszego w tym ciągu w wyniku zastosowania jakiejś pierwotnej reguły inferencji);

(4) $(\forall\beta_{j},\, 1 \leq j \leq i)(\exists\, \lambda_{n},\, 1 \leq n \leq k)\ \lambda_{n} = \beta_{j}$ (czyli każde wyrażenie derywacyjne składające się na wartość dowodzonej formuły transformacyjnej jest identyczne z jakimś wyrazem ciągu $\lambda_{1,}\ldots, \lambda_{n,}\ldots, \lambda_{k}$).

Zgodnie z przedstawioną definicją, konstrukcja LDD-dowodu jakiejś formuły transformacyjnej sprowadza się do wypisania wszystkich wyrażeń derywacyjnych składających się na argument dowodzonej formuły i następnie na przekształcaniu tych wyrażeń zgodnie z pierwotnymi regułami inferencji w taki sposób, aby wyprowadzić wszystkie wyrażenia derywacyjne składające się na wartość dowodzonej formuły. Mówiąc metaforycznie, dowody stanowić będą ciągi pęków gałęzi (a także drzew)

pozostających względem siebie w relacjach transformowalności wyznaczonych przez wyszczególnione reguły pierwotne. Każdy LDD-dowód posiada więc swoje odwzorowanie w postaci ciągu rysunków ilustrujących pęki gałęzi lub drzewa.

3.3. Niesprzeczność LDD-logiki

Dowód niesprzeczności reguł konstytuujących LDD-logikę można przeprowadzić metodą interpretacji. Zinterpretujmy więc język LDD-logiki w języku rachunku zdań z jedną zmienną zdaniową, stałą verum, stałą falsum, funktorem implikacji oraz funktorem n-czynnikowej, jednorodnej koniunkcji '$\wedge_n$', zdefiniowanej indukcyjnie w następujący sposób: (i) $\wedge_1 p \equiv p$; (ii) $\wedge_n p \equiv (\wedge_{n-1} p) \wedge p$.

(T.1) LDD $\in$ NSP.

Dowód: Przyjmijmy następujące konwencje dla funkcji przekładu Int: (1) $(\forall i)(\forall k)[k \neq 0 \rightarrow \mathrm{Int}('\,{}^{k}l_i\,') = 'p']$; (2) Int('**0**') = '1' ('1' oznacza stałą verum); (3) Int('**[**…**]**') = '$\rightarrow$'; (4) Int('//') = '$\rightarrow$'; (5) $(\forall i,\ldots,n)$ Int('l_i,…,l_n') = '$\wedge_n$p', (6) $(\forall i)$Int('$\,{}^{0}l_i\,$') = '0' ('0' oznacza stałą falsum), (7) Int(d_i,…,d_n) = Int(d_i) $\wedge$ … $\wedge$ Int(d_n); (8) warunek dla superponowanych wyrażeń derywacyjnych: $\mathrm{Int}({}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots,\ {}^{n+1}\beta_j[{}^{n+2}\delta_{1,}\ldots,\ {}^{n+2}\delta_i]\ldots,\ {}^{n+1}\beta_k]) = \mathrm{Int}({}^{n}\alpha) \rightarrow [\mathrm{Int}({}^{n+1}\beta_1) \wedge \ldots \wedge \mathrm{Int}({}^{n+1}\beta_j[{}^{n+2}\delta_{1,}\ldots,\ {}^{n+2}\delta_i]) \wedge \ldots \wedge \mathrm{Int}({}^{n+1}\beta_k)]$. Po dokonaniu przekładu pierwotne reguły inferencji LDD-logiki przyjmują postać następujących tautologii klasycznego rachunku zdań: (Mult) dla dowolnego ${}^{0}\alpha$: $(0 \rightarrow \wedge_n p) \rightarrow (0 \rightarrow \wedge_{n+1} p)$; (Mult) dla dowolnego ${}^{n}\alpha$ takiego, że $n \neq 0$: $(p \rightarrow \wedge_n p) \rightarrow (p \rightarrow \wedge_{n+1} p)$; (Elim) dla dowolnego ${}^{0}\alpha$: $(0 \rightarrow \wedge_n p) \rightarrow (0 \rightarrow \wedge_{n-1} p)$; (Elim) dla dowolnego ${}^{n}\alpha$ takiego, że $n \neq 0$: $(p \rightarrow \wedge_n p) \rightarrow (p \rightarrow \wedge_{n-1} p)$; (Intr): $(p \rightarrow 1) \rightarrow (p \rightarrow p)$; (Dest): $(p \rightarrow p) \rightarrow (p \rightarrow 1)$; (Start): $0 \rightarrow (0 \rightarrow p)$; (Trans) dla dowolnego ${}^{0}\alpha$: $(0 \rightarrow p) \rightarrow (p \rightarrow 1)$; (Trans) dla dowolnego ${}^{n}\alpha$ takiego, że $n \neq 0$: $(p \rightarrow p) \rightarrow (p \rightarrow 1)$; (Unif) dla dowolnego ${}^{0}\alpha$: $(0 \rightarrow \wedge_k p) \wedge (p \rightarrow \wedge_i p) \rightarrow [p \rightarrow p \wedge \ldots \wedge (p \rightarrow \wedge_i p) \wedge \ldots \wedge p]$; (Unif) dla dowolnego ${}^{n}\alpha$ takiego, że $n \neq 0$: $(p \rightarrow \wedge_k p) \wedge (p \rightarrow \wedge_i p) \rightarrow [p \rightarrow p \wedge \ldots \wedge (p \rightarrow \wedge_i p) \wedge \ldots \wedge p]$; (Dezint) dla dowolnego ${}^{0}\alpha$: $[0 \rightarrow p \wedge \ldots \wedge (p \rightarrow \wedge_i p) \wedge \ldots \wedge p] \rightarrow [(0 \rightarrow \wedge_k p) \wedge (p \rightarrow \wedge_i p)]$; (Dezint) dla dowolnego ${}^{n}\alpha$ takiego, że $n \neq 0$: $[p \rightarrow p \wedge \ldots \wedge (p \rightarrow \wedge_i p) \wedge \ldots \wedge p] \rightarrow [(p \rightarrow \wedge_k p) \wedge (p \rightarrow \wedge_i p)]$ ♦

Przedstawiony dowód pokazuje również to, że LDD-logika posiada swoją interpretację we fragmencie klasycznego rachunku zdań, zbudowanego wyłącznie za pomocą jednej zmiennej zdaniowej, z funktorami koniunkcji, implikacji oraz falsum. Ta sytuacja sugeruje „sensowność" badań nad podlogikami klasycznego rachunku zdań, różniącymi się liczbą różnych zmiennych zdaniowych wymaganych do artykulacji tez logicznych tych podlogik.

3.4. Przykłady dowodów

Poniżej są przedstawione, dla przykładu, dowody reguł, które uczestniczą w mechanizmach generacji (czyli „rysowania") drzew derywacyjnych. Łatwo zauważyć, że niektóre zbiory wyrażeń derywacyjnych stanowiące wartości formuł transformacyjnych będą opisami drzew derywacyjnych.

(1) Reguła n-multiplikacji: ${}^{n}l_i[{}^{n+1}l_1]$ // ${}^{n}l_i[{}^{n+1}l_1,\ldots,\ {}^{n+1}l_k]$. Reguła ta pozwala na „dorysowanie" dowolnej skończonej ilości gałęzi do danego węzła w drzewie, o ile dany węzeł posiada już jakąś gałąź.

${}^{n}l_i[{}^{n+1}l_1]$	założenie
2. ${}^{n}l_i[{}^{n+1}l_1,\ {}^{n+1}l_2]$	Mult: 1
.	.
.	.
.	.
k. ${}^{n}l_i[{}^{n+1}l_1,\ldots,\ {}^{n+1}l_k]$	Mult: k-1

(2) Reguła introdukcji n-multiplikatywnej: ${}^{n}l_i[\mathbf{0}]$ // ${}^{n}l_i[{}^{n+1}l_1,\ldots,\ {}^{n+1}l_k]$. Reguła ta pozwala na „dorysowanie" dowolnej skończonej ilości gałęzi do danego węzła w drzewie, z którego gałąź prowadzi do elementu pustego.

1. ${}^{n}l_i[\mathbf{0}]$	założenie
2. ${}^{n}l_i[{}^{n+1}l_1]$	Intr: 1
3. ${}^{n}l_i[{}^{n+1}l_1,\ {}^{n+1}l_2]$	Mult: 2
.	.
.	.
.	.
k+3. ${}^{n}l_i[{}^{n+1}l_1,\ldots,\ {}^{n+1}l_k]$	Mult: k+2

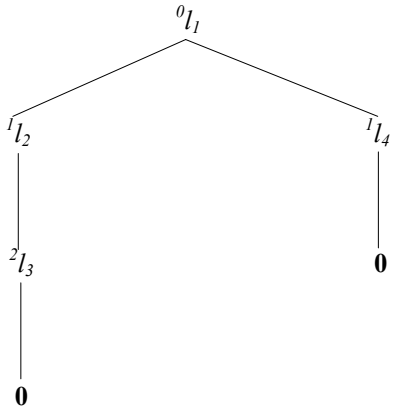
(3) Reguła o postaci: ${}^{0}l_1[{}^{1}l_2,\ {}^{1}l_3]$, ${}^{0}l_4[{}^{1}l_5,\ {}^{1}l_6]$ // ${}^{0}l_1[{}^{1}l_2,\ {}^{1}l_5]$, ${}^{0}l_4[{}^{1}l_3,\ {}^{1}l_6]$.

1. ${}^{0}l_1[{}^{1}l_2,\ {}^{1}l_3]$	założenie
2. ${}^{0}l_4[{}^{1}l_5,\ {}^{1}l_6]$	założenie
3. ${}^{0}l_1[{}^{1}l_2]$	Elim: 1
4. ${}^{0}l_4[{}^{1}l_6]$	Elim: 2
5. ${}^{0}l_1[{}^{1}l_2,\ {}^{1}l_5]$	Mult: 3
6. ${}^{0}l_4[{}^{1}l_3,\ {}^{1}l_6]$	Mult: 4

(4) Reguła o postaci: ${}^{0}l_1$ // ${}^{0}l_1\ [{}^{1}l_2[{}^{2}l_3[\mathbf{0}]]$, ${}^{1}l_4[\mathbf{0}]]$. W tej regule jej wartość stanowi opis nieskomplikowanego drzewa derywacyjnego.

1. ${}^{0}l_1$	założenie
2. ${}^{0}l_1[\mathbf{0}]$	Start: 1
3. ${}^{0}l_1\ [{}^{1}l_2]$	Intr: 2
4. ${}^{0}l_1[{}^{1}l_2,\ {}^{1}l_4]$	Mult: 3
5. ${}^{0}l_1\ [{}^{1}l_4]$	Intr: 2
6. ${}^{1}l_4\ [\mathbf{0}]$	Trans: 5
7. ${}^{0}l_1\ [{}^{1}l_2,\ {}^{1}l_4\ [\mathbf{0}]]$	Unif: 4, 6
8. ${}^{1}l_2\ [\mathbf{0}]$	Trans: 3
9. ${}^{1}l_2\ [{}^{2}l_3]$	Intr: 8
10. ${}^{2}l_3[\mathbf{0}]$	Trans: 9
11. ${}^{1}l_2[{}^{2}l_3[\mathbf{0}]]$	Unif: 9, 10
12. ${}^{0}l_1\ [{}^{1}l_2\ [{}^{2}l_3[\mathbf{0}]]$, ${}^{1}l_4[\mathbf{0}]]$	Unif: 7, 11

Powyższy dowód jest również algorytmem konstrukcji drzewa derywacyjnego o postaci:



Każdy z dowodów można odwzorować w ciąg rysunków, stanowiących odwzorowanie poszczególnych wyrażeń ciągu dowodowego. Zauważmy również, że każde drzewo derywacyjne, poprawnie skonstruowane, daje się „udowodnić" na gruncie swojego wierzchołka.

4. OPERATOR LDD-KONSEKWENCJI I LDD-TEORIE

Zdefiniować można operator LDD-konsekwencji, działający na zbiory wyrażeń derywacyjnych lub wyrażeń leksykalnych poziomu zerowego i zwracający zbiory wyrażeń derywacyjnych. Niech symbole 'D', 'D_1', …, 'D_i' oznaczają zbiory dowolnych wyrażeń derywacyjnych lub dowolnych wyrażeń leksykalnych poziomu zerowego. Definicja operatora C_{LDD} przedstawia się następująco:

(Df. 17) $\quad \alpha \in C_{LDD}(D) \equiv LDD\vdash [D \,//\, \alpha]$.

Wyrażenie derywacyjne α należy do zbioru LDD-konsekwencji zbioru wyrażeń D wtedy i tylko wtedy, gdy formuła transformacyjna: $D \,//\, \alpha$ jest LDD-dowodliwa.

Udowodnić można wiele twierdzeń opisujących właściwości operatora LDD-konsekwencji. Właściwości te są wyznaczone przez właściwości relacji LDD-dowodliwości. Ponadto, można mówić o rozmaitych LDD-teoriach jako najmniejszych zbiorach wyrażeń derywacyjnych zamkniętych ze względu na operację LDD-konsekwencji. Co więcej, wyróżnić można rozmaite typy takich LDD-teorii.

4.1. Właściwości relacji LDD-dowodliwości i operatora LDD-konsekwencji

Następujące twierdzenia opisują podstawowe właściwości LDD-dowodliwości i LDD-konsekwencji:

(T. 2) $(\forall d_{\mathrm{i}},\ldots,d_{\mathrm{n}}, d_{\mathrm{n+1}}, d_{\mathrm{j}})[\mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{j}}) \rightarrow \mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}}, d_{\mathrm{n+1}} /\!/ d_{\mathrm{j}})]$

Dowód: Załóżmy: (1) $\mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{j}})$. Zatem istnieje ciąg dowodowy formuły transformacyjnej $(d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{j}})$. Taki ciąg można przekształcić na ciąg dowodowy formuły transformacyjnej $(d_{\mathrm{i}},\ldots,d_{\mathrm{n}}, d_{\mathrm{n+1}} /\!/ d_{\mathrm{j}})$ poprzez dodanie do listy założeń dowodu wyrażenia derywacyjnego $d_{\mathrm{n+1}}$ ♦

W świetle (T. 2), jeśli pewne wyrażenie derywacyjne jest LDD-wyprowadzalne z pewnego zbioru derywacji, to także jest LDD-wyprowadzalne z każdego zbioru derywacji, w którym ów wyjściowy zbiór jest zawarty. Z (T. 2) wynika to, że operacja LDD-konsekwencji jest monotoniczna.

(T. 3) $(\forall d_i, D_i, D_k)[d_i \in \mathrm{C_{LDD}}(D_i) \rightarrow d_i \in \mathrm{C_{LDD}}(D_i \cup D_k)]$.

(T. 4) $(\forall D_i, D_k)[D_i \subset D_k \rightarrow \mathrm{C_{LDD}}(D_i) \subset \mathrm{C_{LDD}}(D_k)]$.

Zgodnie z innym twierdzeniem (T. 5), dowolne dwa dowody na gruncie LDD o wspólnych przesłankach dadzą się przekształcić na dowód o tych samych przesłankach, w którym są LDD-wyprowadzalne dwie derywacje wyprowadzalne z dowodów wyjściowych.

(T. 5) $(\forall d_{\mathrm{i}},\ldots,d_{\mathrm{n}}, d_{\mathrm{i}}, d_{\mathrm{j}})[\mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{i}}) \wedge \mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{j}}) \rightarrow \mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{i}}, d_{\mathrm{j}})]$.

Dowód: Załóżmy: (1) $\mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{i}})$, (2) $\mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{j}})$. Istnieją więc dwa ciągi dowodowe o wspólnych przesłankach. Dokonując fuzji obu ciągów w taki sposób, że po ostatnim wyrazie ciągu pierwszego następują wszystkie wyrazy ciągu drugiego niebędące przesłankami, otrzymujemy ciąg dowodowy wyrażenia transformacyjnego $(d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{i}}, d_{\mathrm{j}})$ ♦

(T. 5) można określić jako twierdzenie o łączliwości dowodów o tych samych przesłankach. Uogólnieniem (T. 5) jest twierdzenie:

(T. 6) $(\forall d_1,\ldots,d_{\mathrm{n}}, d_{\mathrm{i}},\ldots,d_{\mathrm{j}}, d_{\mathrm{k}},\ldots,d_{\mathrm{h}}, d_{\mathrm{f}},\ldots, d_{\mathrm{z}})[\mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{i}},\ldots, d_{\mathrm{j}}) \wedge \mathrm{LDD}\vdash (d_{\mathrm{k}},\ldots,d_{\mathrm{h}} /\!/ d_{\mathrm{f}},\ldots, d_{\mathrm{z}}) \rightarrow \mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}}, d_{\mathrm{k}},\ldots,d_{\mathrm{h}} /\!/ d_{\mathrm{i}},\ldots, d_{\mathrm{j}}, d_{\mathrm{f}},\ldots, d_{\mathrm{z}})]$.

Dowód: Jeśli $\mathrm{LDD}\vdash (d_{\mathrm{i}},\ldots,d_{\mathrm{n}} /\!/ d_{\mathrm{i}},\ldots, d_{\mathrm{j}})$, to istnieje ciąg dowodowy o postaci: $< d_{\mathrm{i}},\ldots,d_{\mathrm{n}},\ldots,d_{\mathrm{i}},\ldots, d_{\mathrm{j}} >$. Jeśli $\mathrm{LDD}\vdash (d_{\mathrm{k}},\ldots,d_{\mathrm{h}} /\!/ d_{\mathrm{f}},\ldots, d_{\mathrm{z}})$, to istnieje ciąg dowodowy o postaci: $< d_{\mathrm{k}},\ldots,d_{\mathrm{h}},\ldots,d_{\mathrm{f}},\ldots, d_{\mathrm{z}} >$. Dokonując złożenia obu ciągów, otrzymujemy ciąg o postaci: $< d_{\mathrm{i}},\ldots,d_{\mathrm{n}}, d_{\mathrm{k}},\ldots,d_{\mathrm{h}},\ldots,d_{\mathrm{i}},\ldots, d_{\mathrm{j}},\ldots,d_{\mathrm{f}},\ldots, d_{\mathrm{z}} >$. Skonstruowany ciąg spełnia warunki bycia dowodem formuły transformacyjnej: ‘$d_{\mathrm{i}},\ldots,d_{\mathrm{n}}, d_{\mathrm{k}},\ldots,d_{\mathrm{h}} /\!/ d_{\mathrm{i}},\ldots, d_{\mathrm{j}}, d_{\mathrm{f}},\ldots, d_{\mathrm{z}}$’. ♦

(T. 6) można określić mianem prawa składania formuł transformacyjnych stronami. Zgodnie z tym twierdzeniem, LDD-wyprowadzalność derywacji ma charakter kumulatywny. Innymi słowy, im liczniejszy jest zbiór transformowanych wyrażeń derywacyjnych, tym liczniejszy jest zbiór wyrażeń derywacyjnych dowodliwych na gruncie tego pierwszego. Z (T. 6) wynika następujące twierdzenie dla operatora LDD-konsekwencji:

(T. 7) $(\forall D_j,\ D_i,\ D_k,\ D_m)[D_j \subset \mathrm{C}_{\mathrm{LDD}}(D_k) \wedge D_i \subset \mathrm{C}_{\mathrm{LDD}}(D_m) \rightarrow D_j \cup D_i \subset \mathrm{C}_{\mathrm{LDD}}(D_m \cup D_k)]$.

Łatwo wykazać, że każde niesuperponowane wyrażenie derywacyjne, którego parametr głębokości jest równy jeden, jest LDD-wyprowadzalne z samego siebie (najpierw do d stosujemy regułę (Elim), a potem do tak otrzymanego wyrażenia derywacyjnego stosujemy regułę (Mult)).

(T. 8) $(\forall d)[d \in \boldsymbol{Der^1} \rightarrow \mathrm{LDD}\vdash\ (d\ //\ d)]$.

Uogólnieniem (T. 8) jest twierdzenie (T. 9):

(T. 9) $(\forall d_1,\ldots,d_n)[d_1,\ldots,d_n \in \boldsymbol{Der^1} \rightarrow \mathrm{LDD}\vdash\ (d_1,\ldots,d_n\ //\ d_1,\ldots,d_n)]$.

Dowód: Pierwszy warunek dowodu indukcyjnego jest spełniony na mocy (T. 8): $(\forall d)[d \in \boldsymbol{Der^1} \rightarrow \mathrm{LDD}\vdash\ (d_1\ //\ d_1)$. Udowodnić zatem trzeba drugi warunek indukcyjny: $\mathrm{LDD}\vdash (d_1,\ldots,d_n//d_1,\ldots,d_n) \rightarrow \mathrm{LDD}\vdash\ (d_1,\ldots,d_n,d_{n+1}//d_1,\ldots,d_n,\ d_{n+1})$. Załóżmy: (1) $\mathrm{LDD}\vdash\ (d_1,\ldots,d_n\ //\ d_1,\ldots,d_n)$. Na mocy (T. 8) mamy: (2) $\mathrm{LDD}\vdash\ (d_{n+1}\ //\ d_{n+1})$. Stąd na mocy (T. 6) z wierszy (1) i (2) otrzymujemy: $\mathrm{LDD}\vdash (d_1,\ldots,d_n,\ d_{n+1}\ //\ d_1,\ldots,d_n,\ d_{n+1})$ ♦

Z twierdzenia (T. 9) wynika to, że dowolny zbiór wyrażeń derywacyjnych należących do $\boldsymbol{Der^1}$ jest zawarty w zbiorze swoich LDD-konsekwencji.

(T. 10) $(\forall D)[D \subset \boldsymbol{Der^1} \rightarrow D \subset \mathrm{C}_{\mathrm{LDD}}(D)]$.

Analogicznie do powyższych twierdzeń można udowodnić tezy odnoszące się do standardowych superponowanych wyrażeń derywacyjnych:

(T. 11) $(\forall d)[d \in \boldsymbol{Der^{stand}_{sup}} \rightarrow \mathrm{LDD}\vdash\ (d\ //\ d)]$.

Dowód: Jeśli d jest standardowym, superponowanym wyrażeniem derywacyjnym, to stosując regułę dezintegracji (Dezint) do d i sukcesywnie do kolejnych wyrażeń derywacyjnych uzyskanych w wyniku stosowania tej reguły, otrzymujemy zbiór derywacji $\boldsymbol{\Delta}(d)$ stanowiący zakres dekompresji d. Następnie do wyrażeń należących do $\boldsymbol{\Delta}(d)$ (i będących jednocześnie wyrazami ciągu dowodowego) stosujemy regułę unifikacji (Unif). W ten sposób rekonstruujemy wyrażenie derywacyjne d. Obie reguły w tej sytuacji wolno zastosować, gdyż wszystkim wyrażeniom należącym do zakresu dekompresji wyrażenia d parametr głębokości przypisuje wartość jeden ♦

Podobnie dowodzimy następnych twierdzeń:

(T. 12) $(\forall d_1,\ldots,d_n)[d_1,\ldots,d_n \in \boldsymbol{Der^{stand}_{sup}} \rightarrow \mathrm{LDD}\vdash\ (d_1,\ldots,d_n\ //\ d_1,\ldots,d_n)]$.

(T. 13) $(\forall D)[D \subset \boldsymbol{Der}^{\boldsymbol{stand}}_{\boldsymbol{sup}} \rightarrow D \subset \mathrm{C}_{\mathrm{LDD}}(D)]$.

Z (T. 13) i (T. 10) wynika:

(T. 14) $(\forall D)[D \subset \boldsymbol{Der}^{\boldsymbol{1}} \cup \boldsymbol{Der}^{\boldsymbol{stand}}_{\boldsymbol{sup}} \rightarrow D \subset \mathrm{C}_{\mathrm{LDD}}(D)$.

Relacja LDD-dowodliwości odznacza się również tranzytywnością:

(T. 15) $(\forall d_1,\ldots,d_n, d_i,\ldots, d_j, d_f,\ldots, d_z)$ [LDD$\vdash$ $(d_1,\ldots,d_n // d_i,\ldots, d_j) \wedge$ LDD$\vdash$ $(d_i,\ldots,d_j // d_f,\ldots, d_z) \rightarrow$ LDD$\vdash$ $(d_1,\ldots,d_n // d_f,\ldots, d_z)]$.

Dowód: Załóżmy: (1) LDD$\vdash$ $(d_1,\ldots,d_n // d_i,\ldots, d_j)$, (2) LDD$\vdash$ $(d_i,\ldots,d_j // d_f,\ldots, d_z)$. Z (1) wynika to, że istnieje ciąg dowodowy o postaci: $< d_1,\ldots,d_n,\ldots,d_i,\ldots, d_j>$. Z (2) wynika to, że istnieje ciąg dowodowy o postaci: $< d_i,\ldots,d_j,\ldots,d_f,\ldots, d_z >$. Składając oba ciągi, otrzymujemy ciąg: $< d_i,\ldots,d_n,\ldots,d_i,\ldots, d_j,\ldots,d_f,\ldots, d_z >$, który spełnia warunki bycia dowodem transformacji $d_i,\ldots,d_n // d_f,\ldots, d_z$. Zatem LDD$\vdash$ $(d_i,\ldots,d_n // d_f,\ldots, d_z)$ ♦

Zgodnie z następnym twierdzeniem, jeśli z pewnego zbioru wyrażeń derywacyjnych jest wyprowadzalny pewien inny zbiór wyrażeń derywacyjnych, to z pierwszego zbioru jest również wyprowadzalne każde wyrażenie derywacyjne z osobna, należące do drugiego zbioru.

(T. 16) $(\forall D_1, D_2, d)[d \in D_2 \wedge \mathrm{LDD} \vdash (D_1 // D_2) \rightarrow \mathrm{LDD} \vdash (D_1 // d)]$.

Dowód: Załóżmy: (1) $d \in D_2$; (2) LDD $\vdash (D_1 // D_2)$. Z (1) oraz (T. 10): $D \subset \mathrm{C}_{\mathrm{LDD}}(D)$, wynika: (3) $d \in \mathrm{C}_{\mathrm{LDD}}(D_2)$. Z (3) i definicji operatora $\mathrm{C}_{\mathrm{LDD}}$ otrzymujemy: (4) LDD $\vdash (D_2 // d)$. Z kolei z (2) i (4), na mocy twierdzenia (T. 15) o przechodniości relacji LDD-dowodliwości, mamy: (6) LDD $\vdash (D_1 // d)$ ♦

Twierdzenie (T. 16) można uogólnić do postaci, zgodnie z którą jeśli z pewnego zbioru wyrażeń derywacyjnych jest wyprowadzalny pewien zbiór wyrażeń derywacyjnych, to z tego pierwszego jest również wyprowadzalny każdy podzbiór drugiego zbioru wyrażeń derywacyjnych.

(T. 17) $(\forall D_1, D_2, D_3)[D_3 \subset D_2 \wedge \mathrm{LDD} \vdash (D_1 // D_2) \rightarrow \mathrm{LDD} \vdash (D_1 // D_3)]$.

Kolejne twierdzenie stanowi lemat w dowodzie twierdzenia o domkniętości operatora LDD-konsekwencji w zbiorze niesuperponowanych wyrażeń derywacyjnych o parametrze głębokości jeden.

(T. 18) $(\forall D)[D \subset \boldsymbol{Der}^{\boldsymbol{1}} \rightarrow \mathrm{LDD} \vdash (D // \mathrm{C}_{\mathrm{LDD}}(D))]$.

Dowód: Niech: (1) $\mathrm{C}_{\mathrm{LDD}}(D) = \{d_1,\ldots,d_n,\ldots\}$, zatem: (2) $d_1 \in \mathrm{C}_{\mathrm{LDD}}(D) \wedge \ldots \wedge d_n \in \mathrm{C}_{\mathrm{LDD}}(D) \wedge \ldots$ Na mocy definicji $\mathrm{C}_{\mathrm{LDD}}$, z (2) wynika: (3) LDD$\vdash$ $(D // d_1) \wedge \ldots \wedge$ LDD$\vdash$ $(D // d_n) \wedge \ldots$ Z ostatniego wiersza, na mocy (T. 6) mamy: (4) LDD$\vdash$ $(D // d_1,\ldots,d_n, \ldots)$, Z (4) i (1) poprzez regułę ekstensjonalności otrzymujemy: (5) LDD$\vdash$ $(D // \mathrm{C}_{\mathrm{LDD}}(D))$ ♦

(T. 19) $(\forall D)[D \subset \boldsymbol{Der}^{\boldsymbol{1}} \rightarrow \mathrm{C}_{\mathrm{LDD}}(\mathrm{C}_{\mathrm{LDD}}(D)) \subset \mathrm{C}_{\mathrm{LDD}}(D)]$.

Dowód: Załóżmy: (1) $D \subset \boldsymbol{Der}^{1}$ (2) $d \in C_{LDD}(C_{LDD}(D))$. Na mocy definicji C_{LDD} z (1) otrzymujemy: (3) $LDD \vdash (C_{LDD}(D) \,//\, d)$. Z (T. 17) oraz (T. 14) i (1), a także z (3) wynika: (4) $LDD \vdash (D \,//\, d)$. Stosując do (4) definicję C_{LDD}, mamy: (5) $d \in C_{LDD}(D)$. Zatem z (2) i (4) wnioskujemy: (5) $C_{LDD}(C_{LDD}(D)) \subset C_{LDD}(D)$.♦

W podobny sposób, jak (T. 19), dowodzimy:

(T. 20) $\quad (\forall D)[D \subset \boldsymbol{Der}^{stand}_{sup} \rightarrow C_{LDD}(C_{LDD}(D)) \subset C_{LDD}(D)]$.

Z (T. 19) i (T. 20) otrzymujemy twierdzenie, zgodnie z którym operator LDD-konsekwencji charakteryzuje się domkniętością w zbiorze $\boldsymbol{Der}^{1} \cup \boldsymbol{Der}^{stand}_{sup}$.

(T. 21) $\quad (\forall D)[D \subset \boldsymbol{Der}^{1} \cup \boldsymbol{Der}^{stand}_{sup} \rightarrow C_{LDD}(C_{LDD}(D)) \subset C_{LDD}(D)]$.

Okazuje się, że zbiór LDD-konsekwencji zbioru pustego jest zbiorem pustym wyrażeń derywacyjnych.

(T.22) $\quad C_{LDD}(\varnothing) = \varnothing$

Dowód: Załóżmy nie wprost: (1) $d \in C_{LDD}(\varnothing)$. Z (1) i definicji operatora LDD-konsekwencji mamy: (2) $LDD \vdash (\varnothing \,//\, d)$. Z (2) i definicji LDD-wyprowadzalności otrzymujemy: (3) jednowyrazowy ciąg $<d>$ jest dowodem na gruncie LDD. Z kolei: (4) ciąg $<d>$ nie jest dowodem na gruncie LDD (wyrażenie derywacyjne d nie jest uzyskane z wcześniejszych wyrazów ciągu dowodowego na mocy stosowania pierwotnych reguł inferencji). Pomiędzy (3) i (4) zachodzi sprzeczność ♦

Twierdzenia: (T. 4), (T. 13) oraz (T. 21), wyrażają standardowe własności formalne operatora C_{LDD} w zbiorze $\boldsymbol{Der}^{1} \cup \boldsymbol{Der}^{stand}_{sup}$, mianowicie takie, które są posiadane przez każdy operator konsekwencji logicznej. Jednakże w świetle (T. 22), LDD-logika nie może być zdefiniowana jako zbiór konsekwencji C_{LDD} zbioru pustego.[9] LDD-logikę należy po prostu utożsamić ze zbiorem C_{LDD}. Przy takim ujęciu LDD stanowi zbiór par uporządkowanych takich, że pierwszy element takiej pary będzie dowolnym zbiorem wyrażeń derywacyjnych (ewentualnie z elementami leksykalnymi poziomu zerowego), drugi zaś element będzie zbiorem wyrażeń derywacyjnych, którego dowolny podzbiór jest LDD-wyprowadzalny ze zbioru pierwszego.

4.2. LDD-wyrażenia derywacyjne

Nie wszystkie wyrażenia derywacyjne LDD-języka są wyprowadzalne z jakichś innych wyrażeń derywacyjnych. Istnieją więc wyrażenia derywacyjne niedowodliwe na gruncie jakichkolwiek wyrażeń derywacyjnych. Na przykład wyrażenie derywacyjne o postaci „$^{1}l_{1}[^{3}l_{1}, {}^{3}l_{2}]$” nie jest wyprowadzalne z żadnych wyrażeń derywacyjnych. Okazuje się, że tego rodzaju właściwość posiadają wszystkie wyrażenia opi-

[9] Zwrócić jednak trzeba uwagę na to, że właśnie najogólniejsze pojęcie monotonicznej konsekwencji logicznej jest takie, że $Cn(\varnothing) = \varnothing$. Takiego pojęcia używa [Suszko 1998, s. 129].

sujące derywacje wewnątrzpoziomowe lub z przeskokiem derywacyjnym. Tej właściwości nie posiadają wyrażenia opisujące derywacje bez przeskoku derywacyjnego, czyli takie wyrażenia, dla których parametr głębokości przyjmuje wartość równą jeden. Przywołane spostrzeżenie można wyartykułować za pomocą następującego twierdzenia:

(T. 23) $(\forall\, n, k)\,(\forall d)\,[d \in \boldsymbol{Der} \wedge \Omega(d) \in {}^{k}\boldsymbol{L} \wedge \Omega^{*}(d) \subset {}^{n}\boldsymbol{L} \wedge (n = k \vee n - k > 1) \rightarrow \sim (\exists d_i)(d_i \in \boldsymbol{Der} \cup \boldsymbol{Der}_{sup} \wedge \mathrm{LDD}\vdash (d_i \,//\, d)]$.

Dowód: Załóżmy: (1) $d \in \boldsymbol{Der}$, (2) $\Omega(d) \in {}^{k}\boldsymbol{L}$, (3) $\Omega^{*}(d) \subset {}^{n}\boldsymbol{L}$, (4) $n = k \vee n - k > 1$. Załóżmy nie wprost: (5) $(\exists d_i)(d_i \in \boldsymbol{Der} \cup \boldsymbol{Der}_{sup} \wedge \mathrm{LDD}\vdash (d_i \,//\, d)]$. Z (5) wynika: (6) $\mathrm{d_i} \in \boldsymbol{Der}$, (7) $\mathrm{LDD}\vdash (\mathrm{d_i} \,//\, d)$. Z (7) i definicji LDD-dowodliwości mamy: (8) $(\exists d_i)[(d_i \,//\, d) \in (\mathrm{Mult}) \cup (\mathrm{Elim}) \cup (\mathrm{Dest}) \cup (\mathrm{Intr}) \cup (\mathrm{Start}) \cup (\mathrm{Trans}) \cup (\mathrm{Unif})$. Opuszczając w (8) kwantyfikator, dostajemy: (9) $(\mathrm{d_2} \,//\, d) \in (\mathrm{Mult}) \cup (\mathrm{Elim}) \cup (\mathrm{Dest}) \cup (\mathrm{Intr}) \cup (\mathrm{Start}) \cup (\mathrm{Trans}) \cup (\mathrm{Unif})$. Z definicji (Df. 8) — (Df. 15) klas formuł transformacyjnych oraz (9) wynika: (10) $d \in \boldsymbol{Der}^{1}$. Z (2), (3) i (4) wynika zaś, że: (11) $d \notin \boldsymbol{Der}^{1}$. Między (10) i (11) zachodzi sprzeczność ♦

(T. 24) $(\forall\, n, k)(\forall d)\,[d \in \boldsymbol{Der} \wedge \Omega(d) \in {}^{k}\boldsymbol{L} \wedge \Omega^{*}(d) \subset {}^{n}\boldsymbol{L} \wedge (n = k \vee n - k > 1) \rightarrow \sim (\exists d_i)(d_i \in \boldsymbol{Der} \cup \boldsymbol{Der}_{sup} \wedge \mathrm{LDD}\vdash (d \,//\, d_i)]$.

Dowód: Z założenia nie wprost wyprowadzamy: (1) $(d \,//\, d_2) \in (\mathrm{Mult}) \cup (\mathrm{Elim}) \cup (\mathrm{Dest}) \cup (\mathrm{Intr}) \cup (\mathrm{Start}) \cup (\mathrm{Trans}) \cup (\mathrm{Unif})$. Następnie pokazujemy, że z definicji LDD-dowodliwości taka sytuacja nie zachodzi ♦

Twierdzenia (T. 23) i (T. 24) można skomentować metaforycznie w taki oto sposób: niesuperponowane wyrażenia derywacyjne opisujące derywacje o głębokości nierównej jeden są nieaktywne LDD-logicznie. Nie można z nich wyprowadzić żadnych wyrażeń derywacyjnych ani one nie są wyprowadzalne z jakichkolwiek bądź wyrażeń derywacyjnych. Na gruncie LDD-logiki aktywnymi logicznie niesuperponowanymi wyrażeniami derywacyjnymi są tylko te, które przynależą do klasy $\boldsymbol{Der}^{1}$. Tego rodzaju wyrażenia będą określane jako LDD-wyrażenia derywacyjne.

Można również udowodnić twierdzenia dotyczące warunków aktywności logicznej dla superponowanych wyrażeń derywacyjnych:

(T. 25) $(\forall\, n, k)(\forall d)\,[d \in \boldsymbol{Der}_{sup} \wedge (\exists d_j)[d_j \in \boldsymbol{\Delta}(d_j) \wedge \Omega(d_j) \in {}^{k}\boldsymbol{L} \wedge \Omega^{*}(d_j) \subset {}^{n}\boldsymbol{L} \wedge (n = k \vee n - k > 1)] \rightarrow \sim (\exists d_i)(d_i \in \boldsymbol{Der} \cup \boldsymbol{Der}_{sup} \wedge \mathrm{LDD}\vdash (d_i \,//\, d)]$.

Dowód: Z założenia dowodu wynika, że do zakresu dekompresji superponowanego wyrażenia derywacyjnego należy jakieś wyrażenie derywacyjne nienależące do klasy $\boldsymbol{Der}^{1}$. Wyprowadzenie dowolnego superponowanego wyrażenia derywacyjnego z danego wyrażenia derywacyjnego wymaga wyprowadzenia z niego wszystkich wyrażeń należących do zakresu dekompresji danego superponowanego wyrażenia derywacyjnego. A skoro do tego zakresu dekompresji należy jakieś wyrażenie derywacyjne nienależące do klasy $\boldsymbol{Der}^{1}$, to zgodnie z twierdzeniami (T. 23) i (T. 24) jest ono nieaktywne logicznie, a więc niewyprowadzalne z żadnego wyrażenia derywacyjnego. Zatem superponowane wyrażenie derywacyjne, spełniające poprzednik dowodzonego twierdzenia, jest niewyprowadzalne z jakiegokolwiek bądź wyrażenia derywacyjnego ♦

(T. 26) $(\forall\, n, k)(\forall d)\,[d \in \boldsymbol{Der}_{sup} \wedge (\exists d_j)[d_j \in \boldsymbol{\Delta}(\, d_j) \wedge \Omega(d_j) \in {}^{k}\boldsymbol{L} \wedge \Omega^{*}(d_j) \subset {}^{n}\boldsymbol{L} \wedge (n = k \vee n - \mathrm{k} > 1)] \rightarrow \sim (\exists d_i)(d_i \in \boldsymbol{Der} \cup \boldsymbol{Der}_{sup} \wedge \mathrm{LDD}\vdash\ (d \,//\, d_i)]$.

Dowód: Żeby wyprowadzić z superponowanego wyrażenia derywacyjnego jakieś wyrażenie derywacyjne należy posłużyć się regułą (Dezint). Aby zastosować tę regułę, musi być spełniony warunek jej aplikowalności, taki że wszystkim wyrażeniom derywacyjnym należącym do zakresu dekompresji danego superponowanego wyrażenia derywacyjnego parametr głębokości przypisuje wartość jeden. Poprzednik dowodzonego twierdzenia „blokuje" użycie reguły (Dezint). Zatem z takiego niestandardowego superponowanego wyrażenia derywacyjnego nie da się wyprowadzić jakichkolwiek wyrażeń derywacyjnych ♦

LDD-logikę można więc interpretować jako „mechanizm inferencyjny" wyodrębniający (aktywujący) z klasy wszystkich wyrażeń derywacyjnych klasę LDD-wyrażeń derywacyjnych. Można łatwo wykazać, że klasa ${}^{0}\boldsymbol{L}$ elementów leksykalnych stanowi generator klasy wszystkich wyrażeń derywacyjnych $\boldsymbol{Der}^{1} \cup \boldsymbol{Der}^{stand}{}_{sup}$ danego języka z uwagi na relację LDD-dowodliwości.

Niech *L'* oznacza dowolny podzbiór zbioru elementów leksykalnych języka J_{LDD}. Zdefiniujmy funkcję generacji wyrażeń derywacyjnych przez zbiór elementów leksykalnych *L'* w następujący sposób:

(Df. 18) $d \in \boldsymbol{\Gamma}(L') \equiv (\exists\, {}^{0}\alpha)[{}^{0}\alpha \in L' \wedge \mathrm{LDD}\vdash\ ({}^{0}\alpha \,//\, d)$.

Wyrażenie derywacyjne *d* należy do zbioru wyrażeń derywacyjnych generowanych przez zbiór elementów leksykalnych *L'* wtedy i tylko wtedy, gdy istnieje element leksykalny poziomu zerowego, z którego wyrażenie *d* jest LDD-wyprowadzalne. Łatwo zauważyć to, że jeśli *L'* nie ma elementów wspólnych z ${}^{0}\boldsymbol{L}$, to $\boldsymbol{\Gamma}(L')$ jest zbiorem pustym.

(T. 27) $(\forall\, L')[L' \cap {}^{0}\boldsymbol{L} = \varnothing \rightarrow \boldsymbol{\Gamma}(L') = \varnothing]$.

Twierdzenie wyrażające to, że wszystkie (danego, pełnego LDD-języka), niesuperponowane wyrażenia derywacyjne o parametrze głębokości jeden są wyprowadzalne ze zbioru wszystkich elementów leksykalnych poziomu zerowego pełnego języka J_{LDD} (czyli takiego, który zawiera stałe elementy leksykalne każdego poziomu), posiada następującą postać:

(T. 28) $\boldsymbol{L} = {}^{0}\boldsymbol{L} \cup {}^{1}\boldsymbol{L} \cup \ldots \cup {}^{n}\boldsymbol{L} \wedge {}^{0}\boldsymbol{L} \neq \varnothing \wedge \ldots \wedge {}^{n}\boldsymbol{L} \neq \varnothing \rightarrow \boldsymbol{Der}^{1} \subset \boldsymbol{\Gamma}({}^{0}\boldsymbol{L})$.

Dowód: Załóżmy: (1) $\boldsymbol{L} = {}^{0}\boldsymbol{L} \cup {}^{1}\boldsymbol{L} \cup \ldots \cup {}^{n}\boldsymbol{L}$, (2) ${}^{0}\boldsymbol{L} \neq \varnothing \wedge \ldots \wedge {}^{n}\boldsymbol{L} \neq \varnothing$, (3) $d \in \boldsymbol{Der}^{1}$. Zgodnie z (3) otrzymujemy: (4) $d = {}^{n}\alpha\mathbf{[}{}^{n+1}\beta_{1,}\ldots,\ {}^{n+1}\beta_{n}\mathbf{]}$. Na mocy wielokrotnego stosowania reguły (Mult) z ${}^{n}\alpha\mathbf{[}{}^{n+1}\beta_{1}\mathbf{]}$ jest LDD-wyprowadzalne wyrażenie ${}^{n}\alpha\mathbf{[}{}^{n+1}\beta_{1,}\ldots,\ {}^{n+1}\beta_{n}\mathbf{]}$, czyli: (5) LDD $\vdash$ $({}^{n}\alpha\mathbf{[}{}^{n+1}\beta_{1}\mathbf{]}\ //\ d)$. Na mocy reguły (Intr) mamy: (6) LDD $\vdash$ $({}^{n}\alpha\mathbf{[0]}\ //\ {}^{n}\alpha\mathbf{[}{}^{n+1}\beta_{1}\mathbf{]})$. Z (1) i (2) wnioskujemy: (7) ${}^{n-1}\boldsymbol{L} \neq \varnothing$. Z (7) wyprowadzamy: (8) ${}^{n-1}\alpha\mathbf{[}{}^{n}\alpha\mathbf{]} \in J_{LDD}$. Na mocy reguły (Trans) otrzymujemy: (9) LDD $\vdash$ $({}^{n-1}\alpha\mathbf{[}{}^{n}\alpha\mathbf{]}\ //\ ({}^{n}\alpha\mathbf{[0]})$. Indukcyjnie dowodzimy: (10) $(\forall n, k)[k \leq n \wedge n \geq 1 \rightarrow$ LDD $\vdash$ $({}^{n-k}\alpha\mathbf{[}{}^{n-k+1}\alpha\mathbf{]}\ //\ ({}^{n-k+1}\alpha\mathbf{[0]})$. Na mocy twierdzenia o przechodniości relacji wyprowadzalności (T. 15), z (10), (9), (6) i (5) wynika: (11) LDD $\vdash$ $({}^{0}\alpha\mathbf{[}{}^{1}\alpha\mathbf{]}\ //\ d)$. Z kolei na mocy reguły (Start) dostajemy: (12) LDD $\vdash$ $({}^{0}\alpha\ //\ {}^{0}\alpha\mathbf{[0]})$. Z (11) i (12) wynika: (13) LDD $\vdash$ $({}^{0}\alpha\ //\ d)$.

A skoro, na mocy (1) $^{0}\boldsymbol{L} \neq \varnothing$, mamy: (14) $^{0}\alpha \in {}^{0}\boldsymbol{L}$, to stąd i z (13) wyprowadzamy: (14) $(\exists\, ^{0}\alpha)[^{0}\alpha \in {}^{0}\boldsymbol{L} \wedge \mathrm{LDD} \vdash (^{0}\alpha \,//\, d)]$. Z (14) i definicji (Df. 18) wynika: (15) $d \in \boldsymbol{\Gamma}(^{0}\boldsymbol{L})$. ♦

Łatwo jest także udowodnić następujące twierdzenia:

(T. 29) $\boldsymbol{Der}^{stand}{}_{sup} \subset \mathrm{C_{LDD}}(\boldsymbol{Der}^{1})$.

(T. 30) $\boldsymbol{L} = {}^{0}\boldsymbol{L} \cup {}^{1}\boldsymbol{L} \cup \ldots \cup {}^{n}\boldsymbol{L} \wedge {}^{0}\boldsymbol{L} \neq \varnothing \wedge \ldots \wedge {}^{n}\boldsymbol{L} \neq \varnothing \rightarrow \boldsymbol{Der}^{1}{}_{sup} \cup \boldsymbol{Der}^{stand}{}_{sup} = \Gamma(^{0}\boldsymbol{L})$

Zgodnie z (T. 30) zerowy poziom leksykalny generuje w dowolnym, pełnym LDD-języku wszystkie wyrażenia derywacyjne klasy $\boldsymbol{Der}^{1} \cup \boldsymbol{Der}^{stand}{}_{sup}$.

4.3. Typy LDD-teorii

LDD-teoriami są dowolne zbiory wyrażeń derywacyjnych, które są zamknięte z uwagi na relację LDD-wyprowadzalności. Definicja tego pojęcia jest następująca:

(Df. 19) $X \in \mathbf{T}_{\mathrm{LDD}} \equiv (\exists\, D)[X = \mathrm{C_{LDD}}(D) \wedge D \subset X]$.

Łatwo zauważyć, iż LDD-teorie są wyłącznie zbiorami aktywnych inferencyjnie wyrażeń derywacyjnych.

(T. 31) $(\forall\, D)[D \subset \boldsymbol{Der} \wedge \mathrm{C_{LDD}}(D) \not\subset \boldsymbol{Der}^{stand}{}_{sup} \rightarrow \mathrm{C_{LDD}}(D) \notin \mathbf{T}_{\mathrm{LDD}}]$.

Dowód (T. 31) wymaga udowodnienia następującego lematu (uogólnienia T. 8 i T. 11):

(T. 32) $(\forall\, D, d)[\mathrm{LDD} \vdash (D \,//\, d) \rightarrow \mathrm{LDD} \vdash (d \,//\, d)]$

Dowód (T. 32): Załóżmy: (1) $\mathrm{LDD} \vdash (\mathrm{D_1} \,//\, \mathrm{d_i})$. Załóżmy nie wprost: (2) $\sim \mathrm{LDD} \vdash (\mathrm{d_i} \,//\, \mathrm{d_i})$. Z (2) i twierdzenia (T. 8) $(\forall d)[d \in \boldsymbol{Der}^{1} \rightarrow \mathrm{LDD} \vdash (d \,//\, d)]$, otrzymujemy: (3) $\mathrm{d_1} \notin \boldsymbol{Der}^{1}$. Z (1) i definicji LDD-wyprowadzalności wynika: (4) $\mathrm{d_1} \in \boldsymbol{Der}^{1}$. Pomiędzy (4) i (3) zachodzi sprzeczność ♦

Dowód (T. 31): Załóżmy: (1) $\mathrm{d_1} \in \mathrm{C_{LDD}}(D)$, (2) $\mathrm{d_1} \notin \boldsymbol{Der}^{stand}{}_{sup}$, (3) $D \subset \boldsymbol{Der}$. Załóżmy nie wprost: (4) $\mathrm{C_{LDD}}(D) \in \mathbf{T}_{\mathrm{LDD}}$. Z (4) i definicji LDD-teorii wynika: (5) $\mathrm{C_{LDD}}(D) = \mathrm{C_{LDD}}(\mathrm{D_1})$. Z (1) i (5) wyprowadzamy: (6) $\mathrm{d_1} \in \mathrm{C_{LDD}}(\mathrm{D_1})$. Z (6) i definicji LDD-konsekwencji mamy: (7) $\mathrm{LDD} \vdash (\mathrm{D_1} \,//\, \mathrm{d_1})$. Z (3), (2) i (1) wynika: (8) $\mathrm{d_1} \in \boldsymbol{Der} \wedge \Omega(\mathrm{d_1}) \in {}^{k}\boldsymbol{L} \wedge \Omega^{*}(\mathrm{d_1}) \subset {}^{n}\boldsymbol{L} \wedge (n = k \vee k - n > 1)$. Z (8) i twierdzenia (T. 24) wnioskujemy: (9) $\sim (\exists d_i)(d_i \in \boldsymbol{Der} \cup \boldsymbol{Der}_{sup} \wedge \mathrm{LDD} \vdash (d_i \,//\, \mathrm{d_1})]$. Z (8) mamy: (10) $\mathrm{d_1} \in \boldsymbol{Der}$. Z (9), (10) wynika: (11) $\sim \mathrm{LDD} \vdash (\mathrm{d_1} \,//\, \mathrm{d_1})$. Z (7) i (T. 32) wynika: (12) $\mathrm{LDD} \vdash (\mathrm{d_1} \,//\, \mathrm{d_1})$. Między (12) i (11) zachodzi sprzeczność ♦

Ponieważ istnieje pusty zbiór LDD-konsekwencji (gdyż zgodnie z (T. 22): $\mathrm{C_{LDD}}(\varnothing) = \varnothing$), zatem ważne jest następujące twierdzenie:

(T. 33) $\varnothing \in \mathbf{T}_{\mathrm{LDD}}$.

Choć zbiór LDD-konsekwencji dowolnego zbioru nieaktywnych wyrażeń derywacyjnych jest zbiorem pustym, to pomimo tego, że zgodnie z (T. 33) zbiór pusty jest LDD-teorią, zbiory nieaktywnych wyrażeń derywacyjnych nie są LDD-teoriami.

LDD-teorie można typologizować z uwagi na rozmaite kryteria dotyczące rodzaju elementów leksykalnych budujących wyrażenia derywacyjne należące do tych teorii. W celu konstrukcji rozmaitych typów LDD-teorii zdefiniujmy pojęcie bazy leksykalnej dowolnego zbioru wyrażeń derywacyjnych.

(Df. 20) $\alpha \in \boldsymbol{BZL}(\mathrm{X}) \equiv \{\mathrm{X} \subset \boldsymbol{Der} \wedge (\exists d)[d \in \mathrm{X} \wedge (\Omega(d) = \alpha \vee \alpha \in \Omega^*(d))]\}$.

Bazą leksykalną dowolnego zbioru wyrażeń derywacyjnych jest zbiór tych i tylko tych wyrażeń leksykalnych, z których są utworzone wyrażenia derywacyjne należące do danego zbioru. Każdej LDD-teorii można więc przyporządkować pewien typ z uwagi na typ jej bazy leksykalnej.

Wyróżnić można pełne bazy leksykalne. Pełną bazą leksykalną danego zbioru wyrażeń derywacyjnych dowolnego języka J_{LDD}^{n} (gdzie n jest liczbą poziomów derywacyjnych tego języka) jest zbiór takich wyrażeń leksykalnych wraz ze stałą **stop**, że istnieje taka rodzina niepustych jego podzbiorów, której każdy element posiada wspólne wyrażenia leksykalne z każdym kolejnym poziomem derywacyjnym języka J_{LDD}^{n}. Definicja tej kategorii teoretycznej przedstawia się następująco:

(Df. 21) $\mathrm{Z} \in \boldsymbol{Bzl_{peł}} \equiv [\text{`}\mathbf{0}\text{'} \in \mathrm{Z} \wedge (\exists\, \mathrm{X})(\mathrm{Z} = \boldsymbol{BZL}(\mathrm{X}) \wedge \mathrm{Z} \cap {}^{\mathbf{0}}\boldsymbol{L} \neq \varnothing \wedge \ldots \wedge \mathrm{Z} \cap {}^{\mathbf{n}}\boldsymbol{L} \neq \varnothing)]$.

LDD-teorie o pełnych bazach leksykalnych pozwalają na wygenerowanie drzew derywacyjnych kończących się na dowolnym poziomie derywacyjnym języka $J_{LDD}^{\mathbf{n}}$. Pełne bazy leksykalne dla **n**=0 pozwalają na generację wyłącznie rachitycznych drzew derywacyjnych, mianowicie takich, które są pojedynczymi gałęziami z elementem końcowym **stop** i elementem początkowym będącym jakąś stałą leksykalną poziomu zerowego. Jeśli **n**=1, to generowane z takich pełnych baz leksykalnych drzewa mogą również posiadać postać pęków gałęzi posiadających wspólny element początkowy.

Inny typ baz konstytuują pęknięte bazy leksykalne jakiegoś zbioru wyrażeń derywacyjnych. Takie bazy posiadają elementy wspólne z pewnym n-tym poziomem derywacyjnym danego LDD-języka i nie posiadają elementów wspólnych z pewnym niższym od n poziomem derywacyjnym danego LDD-języka. Definicja pękniętych baz przedstawia się następująco:

(Df. 22) $\mathrm{Z} \in \boldsymbol{Bzl_{pęk}} \equiv [\text{`}\mathbf{0}\text{'} \in \mathrm{Z} \wedge (\exists\, \mathrm{X})(\exists\, k)(\mathrm{Z} = \boldsymbol{BZL}(\mathrm{X}) \wedge \mathrm{Z} \cap {}^{k}\boldsymbol{L} \neq \varnothing \wedge \mathrm{Z} \cap {}^{k-1}\boldsymbol{L} = \varnothing)]$.

Na gruncie LDD-teorii z pękniętymi bazami leksykalnymi nie da się wygenerować drzew derywacyjnych kończących się na poziomie derywacyjnym wyższym od poziomu pęknięcia bazy (czyli zgodnie z (Df. 22) wyższym od $k - 1$). Oczywiście, bazy leksykalne niektórych teorii mogą posiadać więcej niż jeden punkt pęknięcia.

Specjalnym rodzajem LDD-teorii są teorie konkretnych drzew derywacyjnych. Jeśli dana jest konstrukcja ustalonego drzewa derywacyjnego, a więc zbioru wyrażeń derywacyjnych zbudowanych wyłącznie za pomocą stałych leksykalnych, to wszystkie wyrażenia derywacyjne, wyprowadzalne z takiego drzewa na gruncie LDD-języka służącego do opisu tego drzewa, konstytuują LDD-teorię danego drzewa. Na gruncie takiej LDD-teorii można wygenerować wszystkie poddrzewa danego drzewa, wszystkie jego fragmenty (gałęzie, drogi, pęki itd.), a także wszystkie LDD-dopuszczalne modyfikacje takiego drzewa. Takie teorie będą więc opisywały algorytmy generacji danego drzewa oraz wszelkie algorytmy przekształcające wyjściowe drzewo w inne drzewa.

Drzewa derywacyjne można zdefiniować jako zbiory wyrażeń derywacyjnych, zbudowanych wyłącznie za pomocą stałych leksykalnych, spełniające następujące warunki: (1) drzewa są skończonymi zbiorami wyrażeń derywacyjnych zbudowanych wyłącznie za pomocą stałych leksykalnych; (2) istnieje dokładnie jedno wyrażenie derywacyjne należące do drzewa, którego argument jest stałą leksykalną poziomu zerowego; (3) każdy argument wyrażenia derywacyjnego należącego do drzewa, niebędący stałą leksykalną poziomu zerowego przynależy do wartości jakiegoś wyrażenia derywacyjnego należącego do drzewa, którego argument jest stałą poziomu leksykalnego mniejszego o jeden od poziomu leksykalnego, do którego przynależy ów pierwszy argument; (4) każda stała leksykalna różna od stałej **stop**, będąca składnikiem wartości jakiegoś wyrażenia derywacyjnego należącego do danego drzewa jest również argumentem jakiegoś innego wyrażenia derywacyjnego należącego do drzewa, którego wartość jest zawarta w zbiorze elementów leksykalnych poziomu o jeden wyższego niż poziom stałej wyjściowej. W celu formalnego zdefiniowania kategorii drzew derywacyjnych wprowadźmy stałą „***Der****" oznaczającą zbiór wyrażeń derywacyjnych zbudowanych wyłącznie za pomocą stałych leksykalnych oraz stałą „***Sk***" oznaczającą zbiory skończone. Definicja przedstawia się następująco:

(Df. 23) $X \in \boldsymbol{Drz} \equiv$ (1) $X \in \boldsymbol{Sk} \wedge (\forall d)[d \in X \rightarrow d \in \boldsymbol{Der*}] \wedge$
(2) $(\exists d)[d \in X \wedge \Omega(d) \in {}^{0}\boldsymbol{L} \wedge (\forall d_i)(d_i \in X \wedge \Omega(d_i) \in {}^{0}\boldsymbol{L} \rightarrow d = d_i)] \wedge$
(3) $(\forall d)(\forall \boldsymbol{k})[d \in X \wedge \boldsymbol{k} \neq \boldsymbol{0} \wedge \Omega(d) \in {}^{k}\boldsymbol{L} \rightarrow (\exists d_i)(d_i \in X \wedge \Omega(d_i) \in {}^{k-1}\boldsymbol{L} \wedge \Omega(d) \in \Omega^*(d_i))] \wedge$
(4) $(\forall d)(\forall \alpha)(\forall \boldsymbol{k})[d \in X \wedge \alpha \in {}^{k}\boldsymbol{L} \wedge \alpha \in \Omega^*(d) \wedge \alpha \neq$ '$\mathbf{0}$' $\rightarrow (\exists d_i)(d_i \in X \wedge \Omega(d_i) = \alpha \wedge \Omega^*(d_i) \subset {}^{k+1}\boldsymbol{L})]$.

Łatwo można wykazać, że dowolna LDD-teoria drzewa derywacyjnego, sformułowana w LDD-języku, w którym nie występują stałe leksykalne różne od stałych służących do konstrukcji danego drzewa, posiada pełną bazę leksykalną:

(T. 34) $(\forall X)[X \in \boldsymbol{Drz} \rightarrow \boldsymbol{BZL}(C_{LDD}(X)) \in \boldsymbol{Bzl_{pel}}]$

(T. 35) $(\forall X)[X \in \boldsymbol{Drz} \rightarrow \boldsymbol{BZL}(X) \in \boldsymbol{Bzl_{pel}}]$

Jeśli dany jest pewien LDD-język, to można w nim wygenerować wiele różnych drzew derywacyjnych. Skoro z każdym drzewem skorelowana jest dokładnie jedna jego LDD-teoria, to wówczas można mówić o zbiorze LDD-teorii skorelowanych z danym LDD-językiem. Taki zbiór można określić mianem przestrzeni derywacyjnej danego LDD-języka. Zdefiniować można rozmaite relacje syntaktyczne zachodzące pomiędzy LDD-teoriami takiej przestrzeni derywacyjnej. Na przykład, można mówić o niewspółmiernych teoriach, czyli takich, które nie posiadają żadnych wspólnych elementów w postaci wyrażeń derywacyjnych zbudowanych wyłącznie za pomocą stałych leksykalnych. Również można skonstruować wiele typów relacji podobieństwa pomiędzy LDD-teoriami danej przestrzeni derywacyjnej.

5. MODEL SEMANTYCZNY LDD-LOGIKI

Charakteryzacja modelu semantycznego LDD-logiki sprowadza się do skonstruowania pewnej struktury teoriomnogościowej zwanej LDD-modelem, a następnie zdefiniowania funkcji wartościowania formuł LDD-języka w tej strukturze i wreszcie podania definicji formuły prawdziwej w LDD-modelu oraz LDD-tautologii (te będą rozumiane jako formuły transformacyjne LDD-języka prawdziwe we wszystkich LDD-modelach). Prawdziwość formuł również jest rozumiana standardowo jako przyjmowanie wartości wyróżnionej dla każdego wartościowania w danym LDD-modelu. Podkreślmy to, że wyrażenia derywacyjne nie posiadają wartości logicznych.

5.1. Konstrukcja LDD-modelu

Niech ***U*** będzie dowolnym uniwersum złożonym z przedmiotów indywiduowych $x_1,\ldots,x_i$. Zdefiniujmy jednoargumentową operację singletonizacji określoną w dziedzinie na elementach zbioru ***U*** w następujący sposób:

(Df. 24) $^{0}x = x$

$^{n}x = \{^{n-1}x\}$.

Zdefiniujmy zbiór wszystkich singletonów generowanych przez uniwersum elementów **U** w następujący sposób:

(Df. 25) $\boldsymbol{Sing}(\boldsymbol{U}) = \{y; (\exists n)\,(\exists x)[x \in \boldsymbol{U} \wedge y = {}^{n}x\}$.

$\boldsymbol{Sing}(\boldsymbol{U})$ jest więc najmniejszym zbiorem zawierającym zbiór ***U*** i zamkniętym na operację singletonizacji. Można więc w tym wypadku mówić o algebrach singletonowych o postaci $<\boldsymbol{Sing}(\boldsymbol{U}), {}^{n}>$, gdzie ***U*** jest zbiorem generatorów takiej algebry.

W następnym kroku konstruujemy zbiór wszystkich podzbiorów zbioru $\boldsymbol{Sing}(\boldsymbol{U})$. Definicja jest następująca:

(Df. 26) $x \in \boldsymbol{Ptg}(\boldsymbol{U}) \equiv x \subset \boldsymbol{Sing}(\boldsymbol{U})$.

Mając skonstruowany zbiór $\boldsymbol{Ptg}(\boldsymbol{U})$, łatwo możemy podać przepis na konstrukcję zbioru derywacji jako zbioru par uporządkowanych zbudowanych w określony sposób z elementów zbioru $\boldsymbol{Sing}(\boldsymbol{U})$ oraz $\boldsymbol{Ptg}(\boldsymbol{U})$. Propozycja jest następująca:

(Df. 27) $\boldsymbol{Dr}(\boldsymbol{U}) = \boldsymbol{Sing}(\boldsymbol{U}) \times \boldsymbol{Ptg}(\boldsymbol{U})$.

Zgodnie z (Df. 27) derywacją generowaną przez dziedzinę $\boldsymbol{U}$ jest dowolna para uporządkowana, której pierwszym składnikiem jest singleton indukowany przez zbiór $\boldsymbol{U}$, drugim zaś elementem jest dowolny podzbiór zbioru singletonów generowanego przez $\boldsymbol{U}$. Na przykład, jeśli $x_1, \ldots x_k$ są elementami $\boldsymbol{U}$, to wówczas pary uporządkowane kształtu: $<x_1, \{{}^{n}x_1,\ldots, {}^{n}x_k\}>$, $<x_1, \varnothing>$, są derywacjami. Oczywiście derywacjami mogą być również takie pary, jak: $< x_1, \{{}^{i}x_1,\ldots, {}^{j}x_k\}>$, gdzie $i \neq j$.

W następnym kroku zdefiniujmy zbiór transformacji. Niech $\boldsymbol{2}^{\boldsymbol{Dr}(\boldsymbol{U})}$ będzie zbiorem wszystkich podzbiorów zbioru derywacji. Niech $\boldsymbol{2}^{\boldsymbol{Dr}(\boldsymbol{U}) \cup \boldsymbol{U}}$ będzie zbiorem wszystkich podzbiorów zbioru utworzonego ze wszystkich derywacji generowanych przez $\boldsymbol{U}$ oraz wszystkich elementów $\boldsymbol{U}$. Zbiór transformacji generowanych przez $\boldsymbol{U}$ jest zbiorem wszystkich par uporządkowanych takich, że pierwszym składnikiem jest jakiś element zbioru $\boldsymbol{2}^{\boldsymbol{Dr}(\boldsymbol{U}) \cup \boldsymbol{U}}$, drugim zaś składnikiem jest jakiś element zbioru $\boldsymbol{2}^{\boldsymbol{Dr}(\boldsymbol{U})}$:

(Df. 28) $\boldsymbol{Tr}(\boldsymbol{U}) = (\boldsymbol{2}^{\boldsymbol{Dr}(\boldsymbol{U}) \cup \boldsymbol{U}}) \times (\boldsymbol{2}^{\boldsymbol{Dr}(\boldsymbol{U})})$.

Na przykład, transformacjami będą pary uporządkowane o postaci: $< \{x_1\}, \{< x_2, \{{}^{n}x_1,\ldots, {}^{n}x_k\}>\}>$, $< \{< x_1, \{{}^{n}x_1,\ldots, {}^{n}x_k\}>, \ldots, < x_2, \{{}^{h}x_1,\ldots, {}^{h}x_k\}>\}$, $\{< x_3, \{{}^{l}x_1,\ldots, {}^{l}x_k\}>, \ldots, < x_2, \{{}^{m}x_1,\ldots, {}^{m}x_k\}>\}>$.

W zbiorze $\boldsymbol{Tr}(\boldsymbol{U})$ można wyróżnić pewien osobliwy zbiór transformacji $\boldsymbol{Tr^*}(\boldsymbol{U})$. Jego konstrukcja wymaga zdefiniowania dodatkowych pojęć. Najpierw wprowadźmy pojęcia pierwszego oraz drugiego składnika pary uporządkowanej:

(Df. 29) $\Omega(x) = \alpha \equiv (\exists\beta)\, x = <\alpha, \beta>$;

$\Omega^*(x) = \beta \equiv (\exists\alpha)\, x = <\alpha, \beta>$.

Zdefiniujmy kategorię derywacji homogenicznych:

(Df. 30) $x \in \boldsymbol{Dr_{hom}}(\boldsymbol{U}) \equiv x \in \boldsymbol{Dr}(\boldsymbol{U}) \wedge [(\exists n)\, (\forall y)[y \in \Omega^*(x) \rightarrow (\exists\, z)(z \in \boldsymbol{U} \wedge y = {}^{n}z)]$

W świetle (Df. 30) derywacje o postaci: $< {}^{k}x, \{{}^{n}x_1,\ldots, {}^{n}x_h\}>$, $< {}^{k}x, \varnothing>$, są derywacjami homogenicznymi, o ile $x, x_1,\ldots, x_h \in \boldsymbol{U}$. Drugi wypadek zachodzi, gdyż $\Omega^*(x) = \varnothing$, dla $< {}^{k}x, \varnothing> \in \boldsymbol{Dr}(\boldsymbol{U})$. A skoro tak jest, to $(\forall y)[y \in \Omega^*(x) \rightarrow (\exists\, z)(z \in \boldsymbol{U} \wedge y = {}^{n}z)$.

Zdefiniujmy funkcję stopnia singletonizacji przyporządkowującą elementom zbiorów: $\boldsymbol{Sing}(\boldsymbol{U})$, $\boldsymbol{Ptg}(\boldsymbol{U})$, $\boldsymbol{Dr}(\boldsymbol{U})$ liczby naturalne w następujący sposób:

(Df. 31) (1) $x \in \boldsymbol{Sing}(\boldsymbol{U}) \rightarrow [\boldsymbol{Stp}(x) = n \equiv (\exists y)(\, y \in \boldsymbol{U} \wedge x = {}^{n}y)]$;

(2) $x \in \boldsymbol{Ptg}(\boldsymbol{U}) \wedge x \neq \varnothing \rightarrow \{\boldsymbol{Stp}(x) = n \equiv [(\exists y)(\, y \in x \wedge y \in \boldsymbol{Sing}(\, \boldsymbol{U}) \wedge \boldsymbol{Stp}(y) = n) \wedge (\forall y)(\, y \in x \rightarrow \sim \boldsymbol{Stp}(y) > n)]\}$;
(3) $\boldsymbol{Stp}(\varnothing) = 0$;
(4) $x \in \boldsymbol{Dr}(\boldsymbol{U}) \rightarrow [\boldsymbol{Stp}(x) = \boldsymbol{Max}[\boldsymbol{Stp}(\Omega(x)), \boldsymbol{Stp}(\Omega^*(x))]$.

Stopień singletonizacji danego singletonu jest liczbą naturalną wskazującą na ilość iteracji wymaganych do wygenerowania danego singletonu z elementu zbioru $\boldsymbol{U}$. Stopień singletonizacji danego zbioru singletonów jest największą liczbą spośród liczb będących stopniami singletonizacji singletonów należących do danego zbioru. Stopień singletonizacji danej derywacji jest identyczny ze stopniem singletonizacji jej argumentu, o ile jest on większy lub równy stopniowi singletonizacji wartości danej derywacji, bądź jest identyczny ze stopniem singletonizacji wartości danej derywacji, o ile jej stopień derywacji jest większy lub równy stopniowi singletonizacji argumentu danej derywacji. Na przykład, $\boldsymbol{Stp}(<^{k}x, \varnothing>) = k$, gdyż $\boldsymbol{Stp}(\varnothing) = 0$.

Kategoria $\boldsymbol{Dr}_{\boldsymbol{hom}}(\boldsymbol{U})$ stanowi podstawę definicyjną dla kategorii derywacji o **n**-tym poziomie głębokości:

(Df. 32) $x \in \boldsymbol{Dr}^{\mathbf{n}}(\boldsymbol{U}) \equiv \{x \in \boldsymbol{Dr}_{\boldsymbol{hom}}(\boldsymbol{U}) \wedge (\forall k)[\boldsymbol{Stp}(\Omega(x)) = k \rightarrow \boldsymbol{Stp}(\Omega^*(x)) = \mathbf{n} + k]\}$.

Poziom głębokości można wyznaczyć jedynie dla derywacji homogenicznych, których stopień singletonizacji argumentu nie jest większy od stopnia singletonizacji wartości. Takie derywacje nazwijmy naturalnymi.

(Df. 33) $x \in \boldsymbol{Dr}_{\boldsymbol{nat}}(\boldsymbol{U}) \equiv (\exists \mathbf{n})\, x \in \boldsymbol{Dr}^{\mathbf{n}}(\boldsymbol{U})$.

Mówiąc metaforycznie, derywacje naturalne są derywacjami „prowadzącymi od danego singletonu do zbioru singletonów tak samo lub bardziej skomplikowanych". Derywacje, których wartością jest zbiór pusty, nazwijmy derywacjami niedokończonymi:

(Df. 34) $x \in \boldsymbol{Dr}_{\boldsymbol{ndk}}(\boldsymbol{U}) \equiv x \in \boldsymbol{Dr}(\boldsymbol{U}) \wedge \Omega^*(x) = \varnothing$.

Definicja zbioru $\boldsymbol{Tr}^*(\boldsymbol{U})$ (wyróżnionych transformacji) przedstawia się następująco:

(Df. 35) $<\alpha, \beta> \in \boldsymbol{Tr}^*(\boldsymbol{U}) \equiv$
(1) $<\alpha, \beta> \in \boldsymbol{Tr}(\boldsymbol{U}) \wedge$
(2) $(\exists x)[x \in \alpha \wedge x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr}_{\boldsymbol{ndk}}(\boldsymbol{U}) \cup \boldsymbol{U}] \wedge$
(3) $(\forall x)[x \in \beta \rightarrow x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr}_{\boldsymbol{ndk}}(\, \boldsymbol{U})] \wedge$
(4) $(\forall y)[y \in \beta \rightarrow (\exists x)(\, x \in \alpha \wedge \sim \boldsymbol{Stp}(x) > \boldsymbol{Stp}(y))] \wedge$
(5) $(\forall y)[y \in \beta \wedge \Omega(y) \in \boldsymbol{U} \rightarrow (\exists x)(x \in \alpha \wedge (\Omega(x) = \Omega(y) \vee x = \Omega(y)))] \wedge$
(6) $(\forall y)[y \in \beta \rightarrow [(\forall x)\, (x \in \alpha \rightarrow \sim \boldsymbol{Stp}(y) > \boldsymbol{Stp}(x)) \rightarrow (\exists z)(z \in \alpha \wedge \Omega(z) = \Omega(y))]]$.

Przykład transformacji należącej do zbioru $\boldsymbol{Tr^*}(\boldsymbol{U})$ jest następujący: o ile x, x_1, x_2, $x_3 \in \boldsymbol{U}$: $<\{<x, \{^1x_1, {}^1x_2\}>, <x_3, \varnothing>\}, \{<^3x, \{^4x_1, {}^4x_2\}>, <^6x, \{^7x\}>\}>$. Sprawdźmy więc, dla przykładu, czy spełnione są warunki (1) — (6). Zauważmy: (1) $<x, \{^1x_1, {}^1x_2\}> \in \boldsymbol{Dr^1}$, (2) $<^3x, \{^4x_1, {}^4x_2\}> \in \boldsymbol{Dr^1}$, (3) $<^6x, \{^7x\}> \in \boldsymbol{Dr^1}$.

Każda z tych par jest derywacją, gdyż ich pierwsze elementy należą do $\boldsymbol{Sing}(\boldsymbol{U})$, drugie zaś elementy — do $\boldsymbol{Ptg}(\boldsymbol{U})$. We wszystkich trzech wypadkach elementy należące do wartości derywacji są zbudowane z singletonów o tym samym stopniu singletonizacji. Zatem wszystkie wymienione derywacje są derywacjami homogenicznymi. We wszystkich trzech przykładach stopień singletonizacji wartości jest wyższy o jeden od stopnia singletonizacji argumentu. Zatem zgodnie z definicją $\boldsymbol{Dr^n}$ analizowane derywacje należą do $\boldsymbol{Dr^1}$. ♦

Ponadto, zgodnie z (Df. 34) mamy: (4) $<x_3, \varnothing> \in \boldsymbol{Dr_{ndk}}(\boldsymbol{U})$. Para $< \{<x, \{^1x_1, {}^1x_2\}>, <x_3, \varnothing>\}, \{<^3x, \{^4x_1, {}^4x_2\}>, <^6x, \{^7x\}>\} >$ jest więc transformacją, gdyż zbiory $\{<x, \{^1x_1, {}^1x_2\}>, <x_3, \varnothing>\}, \{<^3x, \{^4x_1, {}^4x_2\}>, <^6x, \{^7x\}>\}$ są zbiorami derywacji. Spełniony jest więc warunek (1) w definicji $\boldsymbol{Tr^*}(\boldsymbol{U})$. Ponadto, spełnione są warunki (2) i (3) tejże definicji, gdyż wszystkie derywacje konstytuujące badaną transformację należą do $\boldsymbol{Dr^1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U})$.

Zauważmy: $\boldsymbol{Stp}(<x, \{^1x_1, {}^1x_2\}>) = 1$, $\boldsymbol{Stp}(<x_3, \varnothing>) = 0$, $\boldsymbol{Stp}(<^3x, \{^4x_1, {}^4x_2\}>) = 4$, $\boldsymbol{Stp}(<^6x, \{^7x\}>) = 7$. Stopień singletonizacji żadnej z derywacji należącej do argumentu danej transformacji nie jest wyższy od stopnia singletonizacji każdej z derywacji należącej do wartości analizowanej transformacji. ♦

Spełniony jest więc warunek (4) w definicji $\boldsymbol{Tr^*}(\boldsymbol{U})$. Ponieważ żaden z elementów: $\Omega (<^3x, \{^4x_1, {}^4x_2\}>)$, $\Omega(<^6x, \{^7x\}>)$, nie należy do $\boldsymbol{U}$, więc spełniony jest warunek (5). Ponieważ stopień singletonizacji dowolnego elementu ze zbioru $\{<^3x, \{^4x_1, {}^4x_2\}>, <^6x, \{^7x\}>\}$ jest wyższy od stopnia singletonizacji każdego z elementów zbioru $\{<x, \{^1x_1, {}^1x_2\}>, <x_3, \varnothing>\}$, więc w warunku (6) podformuła: $(\forall x)\ (x \in \alpha \rightarrow \sim \boldsymbol{Stp}(y) > \boldsymbol{Stp}(x))$, jest fałszywa, a zatem warunek (6) jest spełniony przez badaną strukturę.

Przykładem transformacji, która nie należy do $\boldsymbol{Tr^*}(\boldsymbol{U})$, jest następująca: $< \{<^6x, \{^7x\}>, <^5x, \varnothing>\}, \{<\ ^0x, \{^1x\}>\}>$. W tym wypadku nie jest spełniony warunek (4), gdyż $\boldsymbol{Stp}(<^6x, \{^7x\}>) = 7$; $\boldsymbol{Stp}(<^5x, \varnothing>) = 5$ i $\boldsymbol{Stp}(<^0x, \{^1x\}>) = 1$, czyli: $\boldsymbol{Stp}(<^6x, \{^7x\}>) > \boldsymbol{Stp}(<^0x, \{^1x\}>)$ i $\boldsymbol{Stp}(<^5x, \varnothing>) > \boldsymbol{Stp}(<^0x, \{^1x\}>)$.

LDD-modele zdefiniujemy jako pary uporządkowane o postaci: $<\boldsymbol{Tr}(\boldsymbol{U}), \boldsymbol{Tr^*}(\boldsymbol{U})>$:

(Df. 36) $\qquad \boldsymbol{M_U} = <\boldsymbol{Tr}(\boldsymbol{U}), \boldsymbol{Tr^*}(\boldsymbol{U})>$

Każdy więc niepusty zbiór elementów generuje LDD-model. Jeśli $\boldsymbol{U}$ jest zbiorem jednoelementowym, to wówczas LDD-model generowany przez $\boldsymbol{U}$ będzie określany jako model liniowy.

5.2. Wartościowanie w LDD-językach

Funkcją wartościowania wyrażeń dowolnego LDD-języka w LDD-modelu generowanym przez uniwersum $\boldsymbol{U}$ jest dowolna funkcja, która spełnia wyznaczone definicyjnie warunki. A więc z każdą parą typu: $<J_{LDD}, \boldsymbol{M_U}>$ może być skorelowanych wiele funkcji wartościowania. Skoro dowolny LDD-język jest trójpoziomowy — wyróżniamy poziom wyrażeń leksykalnych, poziom wyrażeń derywacyjnych oraz poziom wyrażeń transformacyjnych — więc funkcja wartościowania będzie posiadała dla każdego poziomu odmienny mechanizm przyporządkowywania elementów wymienionych poziomów do elementów w strukturze modelowej $\boldsymbol{M_U}$. I tak wyrażeniom leksykalnym (stałym lub zmiennym) są przyporządkowywane elementy zbioru $\boldsymbol{Sing}(\boldsymbol{U})$, wyrażeniom derywacyjnym — elementy zbioru $\boldsymbol{Dr}(\boldsymbol{U})$, formułom zaś transformacyjnym — elementy zbioru $\boldsymbol{Tr}(\boldsymbol{U})$. Zdefiniowanie funkcji $\mathbf{VAL}_{i,\, M(U)}$ wymaga wprowadzenia pewnego technicznego pojęcia, mianowicie funkcji homogenizacji:

(Df. 37) (i) $x \in \boldsymbol{U} \rightarrow \boldsymbol{h}(x) = \{x\}$
(ii) $x \in \boldsymbol{Dr}(\boldsymbol{U}) \rightarrow \boldsymbol{h}(x) = \{x\}$
(iii) $x \subset \boldsymbol{Dr}(\boldsymbol{U}) \rightarrow \boldsymbol{h}(x) = x$.

Funkcja homogenizacji $\boldsymbol{h}$ przekształca element uniwersum $\boldsymbol{U}$ na jego singleton. W ten sam sposób funkcja $\boldsymbol{h}$ działa na derywacje. Natomiast w aplikacji do zbiorów derywacji zachowuje się jak funkcja tożsamościowa.

Niech $\boldsymbol{L}$ będzie zbiorem wszystkich wyrażeń leksykalnych dowolnego LDD-języka, ${}^0\boldsymbol{L}$, …, ${}^k\boldsymbol{L}$ będą podzbiorami $\boldsymbol{L}$ konstytuującymi korespondujące poziomy derywacyjne, $\boldsymbol{L_{const}}$ zaś zbiorem pozalogicznych stałych leksykalnych różnych od stałej **stop**; niech $\boldsymbol{Der}$ będzie zbiorem niesuperponowanych wyrażeń derywacyjnych, $\boldsymbol{Der_{sup}}$ zaś — zbiorem superponowanych wyrażeń derywacyjnych, a $\boldsymbol{Trans}$ — zbiorem wyrażeń transformacyjnych. Wówczas powiemy, że funkcja wartościowania LDD-języka spełnia następujące warunki:

(Df. 38)
(1) $\mathbf{VAL}_{i, M(U)} \subset [(\boldsymbol{L} \times \boldsymbol{Sing}(\boldsymbol{U})) \cup (\boldsymbol{Der} \times \boldsymbol{Dr}(\boldsymbol{U})) \cup (\boldsymbol{Trans} \times \boldsymbol{Tr}(\boldsymbol{U}))]$;
(2) ${}^n\alpha \in \boldsymbol{L} \wedge {}^n\alpha \neq$ '**0**' $\rightarrow (\exists x, x \in \boldsymbol{U})[\mathbf{VAL}_{i, M(U)}({}^n\alpha) = {}^n x]$;
(3) $(\forall i)\, \mathbf{VAL}_{i, M(U)}($'**0**'$) = \varnothing$;
(4) ${}^n\alpha \in \boldsymbol{L_{const}} \rightarrow (\exists x, x \in \boldsymbol{U})(\forall i)[\mathbf{VAL}_{i, M(U)}({}^n\alpha) = {}^n x]$;
(5) $\alpha[$'**0**'$] \in \boldsymbol{Der} \rightarrow \mathbf{VAL}_{i, M(U)}(\alpha[$'**0**'$]) = < \mathbf{VAL}_{i, M(U)}(\alpha), \mathbf{VAL}_{i, M(U)}($'**0**'$)>$;
(6) $\alpha[\beta_1,\ldots,\beta_n] \in \boldsymbol{Der} \wedge \beta_1 \neq$ '**0**' $\wedge \ldots \wedge \beta_n \neq$ '**0**' $\wedge \{\beta_1,\ldots,\beta_n\} \subset \boldsymbol{L} \rightarrow \mathbf{VAL}_{i, M(U)}(\alpha[\beta_1,\ldots,\beta_n]) = < \mathbf{VAL}_{i, M(U)}(\alpha), \{\mathbf{VAL}_{i, M(U}(\beta_1),\ldots, \mathbf{VAL}_{i, M(U)}(\beta_n)\}>$;
(7) $\alpha \in \boldsymbol{Der_{sup}} \rightarrow \mathbf{VAL}_{i, M(U)}(\alpha) = \{x; (\exists \beta)(\beta \in \Delta(\alpha) \wedge \beta \in \boldsymbol{Der} \wedge x = \mathbf{VAL}_{i, M(U)}(\beta)\}$;

(8) $[\alpha_1,\ldots,\alpha_n // \beta_1,\ldots,\beta_i] \in \boldsymbol{Trans} \rightarrow \mathbf{VAL}_{i,\,M(U)}([\alpha_1,\ldots,\alpha_n // \beta_1,\ldots,\beta_k]) = < \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)), \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\beta_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\beta_k))>$.

Pierwszy warunek w (Df. 37) ma charakter ogólny i określa dziedzinę i przeciwdziedzinę funkcji wartościowania. Zgodnie z drugim warunkiem, dowolnemu wyrażeniu leksykalnemu n-tego poziomu derywacyjnego, różnemu od stałej **stop**, funkcja wartościowania przyporządkowuje pewien obiekt należący do uniwersum **sing**($\boldsymbol{U}$), którego stopień singletonizacji wynosi właśnie n. Trzeci warunek wyraża to, że stałej **stop** każde wartościowanie zawsze przyporządkowuje zbiór pusty. Warunek czwarty stwierdza to, że każda funkcja wartościowania dla stałych leksykalnych, różnych od stałej **stop**, zachowuje się sztywno, czyli że zawsze przyporządkowuje danej stałej ten sam obiekt o odpowiednim stopniu singletonizacji (korespondującym z poziomem derywacyjnym stałej). Cztery ostatnie warunki wyznaczają to, w jaki sposób funkcja wartościowania przyporządkowuje wartość (odpowiednio: derywacje oraz transformacje generowane przez $\boldsymbol{U}$) wyrażeniom derywacyjnym oraz transformacyjnym.

Niech następujący przykład stanowi egzemplifikację obliczania wartości logicznej dla wyrażenia transformacyjnego o postaci: ‘$[^0\boldsymbol{l}, {}^3\boldsymbol{l}\mathbf{[0]} // {}^4\boldsymbol{l}[^5\boldsymbol{l}, {}^5\boldsymbol{l}]]$’.

Niech $\boldsymbol{U} = \{x\}$. Zatem: (1) $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^0\boldsymbol{l}$’) = x, (2) $\mathbf{VAL}_{i,\,M(U)}$ (‘**0**’) = ∅, (3) $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^3\boldsymbol{l}$’) = ^{3}x, (4) $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^4\boldsymbol{l}$’) = ^{4}x, (5) $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^5\boldsymbol{l}$’) = ^{5}x, (6) $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^5\boldsymbol{l}$ ’) = ^{5}x. Następnie obliczamy wartości dla wyrażeń derywacyjnych: (7) $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^3\boldsymbol{l}$**[0]**’) = < $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^3\boldsymbol{l}$’), $\mathbf{VAL}_{i,\,M(U)}$(‘**0**’) >, (8) $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^4\boldsymbol{l}[^5\boldsymbol{l}, {}^5\boldsymbol{l}]$’) = < $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^4\boldsymbol{l}$’), {$\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^5\boldsymbol{l}$’), $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^5\boldsymbol{l}$ ’)} >. Obliczamy więc wartość dla wyrażenia transformacyjnego: (9) $\mathbf{VAL}_{i,\,M(U)}$(‘$[^0\boldsymbol{l}, {}^3\boldsymbol{l}\mathbf{[0]} // {}^4\boldsymbol{l}[^5\boldsymbol{l}, {}^5\boldsymbol{l}]]$’) = < $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^0\boldsymbol{l}$’)) ∪ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^3\boldsymbol{l}$**[0]**’)), $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^4\boldsymbol{l}[^5\boldsymbol{l}, {}^5\boldsymbol{l}]$’))>. Stosując (7) i (8), a także definicję funkcji $\boldsymbol{h}$, do (9) otrzymujemy: (10) $\mathbf{VAL}_{i,\,M(U)}$(‘$[^0\boldsymbol{l}, {}^3\boldsymbol{l}\mathbf{[0]} // {}^4\boldsymbol{l}[^5\boldsymbol{l}, {}^5\boldsymbol{l}]]$’) = < {$\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^0\boldsymbol{l}$’)} ∪ {< $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^3\boldsymbol{l}$’), $\mathbf{VAL}_{i,\,M(U)}$(‘ **0** ’) >}, {< $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^4\boldsymbol{l}$’), {$\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^5\boldsymbol{l}$’), $\mathbf{VAL}_{i,\,M(U)}$ (‘ ${}^5\boldsymbol{l}$ ’)} >} >. Stosując do (10) równości: (1), (2), (3), (4), (5), (6) otrzymujemy: (11) $\mathbf{VAL}_{i,\,M(U)}$(‘$[^0\boldsymbol{l}, {}^3\boldsymbol{l}\mathbf{[0]} // {}^4\boldsymbol{l}[^5\boldsymbol{l}, {}^5\boldsymbol{l}]]$’) = < {x, < ^{3}x, ∅ >}, {< ^{4}x, {^{5}x, ^{5}x}>} >. A ponieważ {^{5}x, ^{5}x} = {^{5}x}, więc otrzymujemy: (12) $\mathbf{VAL}_{i,\,M(U)}$(‘$[^0\boldsymbol{l}, {}^3\boldsymbol{l}\mathbf{[0]} // {}^4\boldsymbol{l}[^5\boldsymbol{l}, {}^5\boldsymbol{l}]]$’) = < {x, < ^{3}x, ∅ >}, {< ^{4}x, {^{5}x}>} >.

Łatwo zauważyć, że < {x, < ^{3}x, ∅ >}, {< ^{4}x, {^{5}x}>} > ∈ $\boldsymbol{Tr}(\boldsymbol{U})$, gdyż {x, < ^{3}x, ∅ >} ⊂ $\boldsymbol{Dr}(\boldsymbol{U}) \cup \boldsymbol{U}$ oraz {< ^{4}x, {^{5}x}>} ⊂ $\boldsymbol{Dr}(\boldsymbol{U})$, czyli < {x, < ^{3}x, ∅ >}, {< ^{4}x, {^{5}x}>} > ∈ $(\mathbf{2}^{\boldsymbol{Dr}(U) \cup U}) \times (\mathbf{2}^{\boldsymbol{Dr}(U)})$. Okazuje się również, że < {x, < ^{3}x, ∅ >}, {< ^{4}x, {^{5}x}>} > ∈ $\boldsymbol{Tr}^*(\boldsymbol{U})$.

Pojęcie wartościowania służy do zdefiniowania pojęcia prawdziwości w modelu $\boldsymbol{M_U}$ oraz tautologiczności jako prawdziwości w każdym modelu $\boldsymbol{M_U}$. Definicje wymienionych pojęć przedstawiają się następująco (niech $\boldsymbol{Prawda}(\boldsymbol{M_U})$ będzie zbiorem wszystkich prawdziwych formuł transformacyjnych w danym modelu, ***Taut*** zaś zbiorem wszystkich tautologii LDD-logiki):

(Df. 39) $\alpha \in \boldsymbol{Prawda}(\boldsymbol{M_U}) \equiv (\forall i)\ \mathbf{VAL}_{i,\,M(U)}(\alpha) \in \boldsymbol{Tr}^*(\boldsymbol{U})$;

(Df. 40) $\alpha \in \boldsymbol{Taut} \equiv (\forall\, \boldsymbol{U})\ \alpha \in \boldsymbol{Prawda}(\boldsymbol{M}_{\boldsymbol{U}})$.

Formuła transformacyjna jest prawdziwa w modelu $\boldsymbol{M}_{\boldsymbol{U}}$ wtedy i tylko wtedy, gdy dla każdego wartościowania w tym modelu jej wartością jest transformacja wyróżniona, która jest generowana przez uniwersum $\boldsymbol{U}$. LDD-teorie „wybierają" więc ze zbioru transformacji generowanych przez dane uniwersum elementów pewien podzbiór transformacji o określonych właściwościach formalnych. LDD-logika opisuje więc mechanizm inferencyjny teoriomnogościowego konstruowania par uporządkowanych o pewnych osobliwych właściwościach.

5.3. Pełność LDD-logiki

Można wykazać, że LDD-logika jest pełna w tym znaczeniu, że każda jej tautologia jest dowodliwa (zakładamy, że język LDD-logiki obejmuje zmienne leksykalne każdego poziomu derywacyjnego):

(T. 36) $(\forall\alpha)[\mathrm{LDD}\vdash\ \alpha \equiv \alpha \in \boldsymbol{Taut}]$.

Dowód twierdzenia o pełności LDD-logiki będzie przebiegał dwuetapowo. Najpierw zostanie udowodnione twierdzenie: $(\forall\alpha)[\mathrm{LDD}\vdash\ \alpha \rightarrow \alpha \in \boldsymbol{Taut}]$, a potem: $(\forall\alpha)\ [\alpha \in \boldsymbol{Taut} \rightarrow \mathrm{LDD}\vdash\ \alpha]$. Dowodząc: $(\forall\alpha)[\mathrm{LDD}\vdash\ \alpha \rightarrow \alpha \in \boldsymbol{Taut}]$, należy udowodnić wszystkie warunki definicyjne dla $\boldsymbol{Tr}^*(\boldsymbol{U})$, która to z kolei kategoria jest użyta w definicji LDD-tautologii:

(T. 37) $(\forall\alpha_1,\ldots,\alpha_n)\ (\forall d_1,\ldots,d_k)\ (\forall\ \mathrm{i})[\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/\ d_1,\ldots,d_k]\ \rightarrow \mathbf{VAL}_{\mathrm{i},\boldsymbol{M}(\boldsymbol{U})}\ (\alpha_1,\ldots,\alpha_n /\!/\ d_1,\ldots,d_k) \in \boldsymbol{Tr}(\boldsymbol{U})]$.

Dowód: Załóżmy: (1) $\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/\ d_1,\ldots,d_k]$. Z (1) wynika: (2) $[\alpha_1,\ldots,\alpha_n /\!/\ d_1,\ldots,d_k] \in \boldsymbol{Trans}$. Z (2) i warunku (8) w definicji wartościowania wyprowadzamy: (3) $\mathbf{VAL}_{\mathrm{i},\boldsymbol{M}(\boldsymbol{U})}([\alpha_1,\ldots,\alpha_n /\!/\ d_1,\ldots,d_k]) = \langle \boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(\alpha_n)), \boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(d_k))\rangle$. Z definicji funkcji $\boldsymbol{h}$, (2) i definicji wartościowania dostajemy: (4) $\boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(\alpha_n)) \subset \boldsymbol{Dr}(\boldsymbol{U}) \cup \boldsymbol{U}$, (5) $\boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(d_k)) \subset \boldsymbol{Dr}(\boldsymbol{U})$. Z (3), (4) i (5), na mocy definicji $\boldsymbol{Tr}(\boldsymbol{U})$, wynika: (6) $\mathbf{VAL}_{\mathrm{i},\boldsymbol{M}(\boldsymbol{U})}(\alpha_1,\ldots,\alpha_n /\!/\ d_1,\ldots,d_k) \in \boldsymbol{Tr}(\boldsymbol{U})]\blacklozenge$

(T. 38) $(\forall\alpha_1,\ldots,\alpha_n)\ (\forall d_1,\ldots,d_k)\ (\forall\ \mathrm{i})[\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/\ d_1,\ldots,d_k] \rightarrow (\exists\mathrm{x})[\mathrm{x} \in \boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathrm{i},\,\boldsymbol{M}(\boldsymbol{U})}(\alpha_n)) \wedge \mathrm{x} \in \boldsymbol{Dr}^1(\boldsymbol{U}) \cup \boldsymbol{Dr}_{\boldsymbol{ndk}}(\boldsymbol{U}) \cup \boldsymbol{U}]]$.

Dowód: Załóżmy: (1) $\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/\ d_1,\ldots,d_k]$. Z (1) oraz pierwszego warunku LDD-dowodliwości wynika: (2) $\{\alpha_1,\ldots,\alpha_n\} \subset {}^{0}\boldsymbol{L} \cup \boldsymbol{Der} \cup \boldsymbol{Der}_{\boldsymbol{sup}}$. Z definicji wartościowania wnioskujemy: (3) $\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M}(\boldsymbol{U})}(\alpha_1,\ldots,\alpha_n /\!/\ d_1,\ldots,d_k) = \langle \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M}(\boldsymbol{U})}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M}(\boldsymbol{U})}(\alpha_n)), \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M}(\boldsymbol{U})}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M}(\boldsymbol{U})}(d_i)) \rangle$. Na mocy (T. 24): $(\forall\, n, k)(\forall d)\ [d \in \boldsymbol{Der} \cup \boldsymbol{Der}_{\boldsymbol{sup}} \wedge \Omega(d) \in {}^{k}\boldsymbol{L} \wedge \Omega^*(d) \subset {}^{n}\boldsymbol{L} \wedge (n = k \vee n \text{ — } k > 1) \rightarrow \sim (\exists d_i)(d_i \in \boldsymbol{Der} \cup \boldsymbol{Der}_{\boldsymbol{sup}} \wedge \mathrm{LDD}\vdash\ (d /\!/ d_i)]$, przez transpozycję, wyprowadzamy: (4) $(\forall\, n, k)\ (\forall\ \alpha)[(\exists d_i)(d_i \in \boldsymbol{Der} \cup \boldsymbol{Der}_{\boldsymbol{sup}} \wedge \mathrm{LDD}\vdash (\alpha /\!/ d_i) \rightarrow (\alpha \notin \boldsymbol{Der} \cup \boldsymbol{Der}_{\boldsymbol{sup}} \vee \Omega(\alpha) \notin {}^{k}\boldsymbol{L} \vee \sim\Omega^*(\alpha) \subset {}^{n}\boldsymbol{L} \vee \sim(n = k \vee n \text{ — } k > 1))]$. Z (1) wyni-

ka: (5) $(\exists d_i)(d_i \in \boldsymbol{Der} \cup \boldsymbol{Der_{sup}} \wedge \mathrm{LDD}\vdash\ (\alpha_{h,\, n \geq h \geq 1} /\!/ d_i))$. Z (4) i (5) odrywamy: (6) $\alpha_{h,\, n \geq h \geq 1} \notin \boldsymbol{Der} \cup \boldsymbol{Der_{sup}} \vee \Omega(\alpha_{h\, n \geq h \geq 1}) \notin {}^{k}\boldsymbol{L} \vee {\sim}\Omega^{*}(\alpha_{h,\, n \geq h \geq 1}) \subset {}^{n}\boldsymbol{L} \vee {\sim}(n = k \vee \boldsymbol{n - k} > 1)$.

Załóżmy dodatkowo: (6.1) $\alpha_{h,\, n \geq h \geq 1} \notin \boldsymbol{Der} \cup \boldsymbol{Der_{sup}}$. Zatem z (6.1) oraz (2) wynika: (6.2) $\alpha_{h,\, n \geq h \geq 1} \in {}^{0}\boldsymbol{L}$. Z definicji funkcji wartościowania oraz z (6.2) otrzymujemy: (6.3) $\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_{h,\, n \geq h \geq 1}) \in \boldsymbol{U}$. Skoro $\alpha_{h,\, n \geq h \geq 1}$ jest w zbiorze $\{\alpha_1,\ldots,\alpha_n\}$, to: (6.4) $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_{h,\, n \geq h \geq 1})) \subset \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n))$. Z (6.4), (6.3) i definicji funkcji $\boldsymbol{h}$ wynika: (6.5) $\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_{h,\, n \geq h \geq 1}) \in \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n))$. Z (6.3) oraz (6.5) dostajemy: (6.6) $(\exists x)[x \in \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n)) \wedge x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U}) \cup \boldsymbol{U}]$. Zatem z (6.1) i (6.6) mamy: (7) $\alpha_{h,\, n \geq h \geq 1} \notin \boldsymbol{Der} \cup \boldsymbol{Der_{sup}} \rightarrow (\exists x)[x \in \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n)) \wedge x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U}) \cup \boldsymbol{U}]$.

Załóżmy dodatkowo: (7.1) $\alpha_{h,\, n \geq h \geq 1} \in \boldsymbol{Der} \cup \boldsymbol{Der_{sup}}$. Z (7.1) wynika: (7.2) $\Omega(\alpha_{h,\, n \geq h \geq 1}) \in {}^{k}\boldsymbol{L}$, (gdyż każde wyrażenie derywacyjne posiada swój argument i swoją wartość). Zatem z (7.1), (7.2) i (6) dostajemy (7.3) ${\sim}\Omega^{*}(\alpha_{h,\, n \geq h \geq 1}) \subset {}^{n}\boldsymbol{L} \vee {\sim}(n = k \vee n - k > 1)$. Załóżmy dodatkowo: (7.3.1) ${\sim}(n = k \vee n - k > 1)$. Z (7.3.1), (7.2) oraz definicji parametru głębokości wynika: (7.3.2) $\alpha_{h,\, n \geq h \geq 1} \in \boldsymbol{Der}^{\boldsymbol{I}}$. Z (7.3.2) oraz warunku w definicji wartościowania mamy: (7.3.3) $\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_{h,\, n \geq h \geq 1}) \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U})$. Skoro $\alpha_{h,\, n \geq h \geq 1}$ jest w zbiorze $\{\alpha_1,\ldots,\alpha_n\}$, to: (7.3.4) $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_{h,\, n \geq h \geq 1})) \subset \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n))$. Z (7.3.1) wyprowadzamy: (7.4) ${\sim}(n = k \vee n - k > 1) \rightarrow (\exists x)[x \in \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n)) \wedge x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U}) \cup \boldsymbol{U}]$. Załóżmy dodatkowo: (7.4.1) ${\sim}\Omega^{*}(\alpha_{h,\, n \geq h \geq 1}) \subset {}^{n}\boldsymbol{L}$. Opuszczając kwantyfikator w (5) dostajemy: (7.4.2) $\mathrm{LDD}\vdash\ (\alpha_{h,\, n \geq h \geq 1} /\!/ d_i)$. Zatem $\alpha_{h,\, n \geq h \geq 1}$ jest LDD-aktywne logicznie. Stąd na mocy (7.1) i (7.4.2) wyprowadzamy: (7.4.3) $\alpha_{h,\, n \geq h \geq 1} \in \boldsymbol{Der_{sup}}$. Stosując definicję wartościowania do (7.4.3), dostajemy: (7.4.4) $\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_{h,\, n \geq h \geq 1}) = \{x; (\exists\, \beta)(\beta \in \Delta(\alpha_{h,\, n \geq h \geq 1}) \wedge \beta \in \boldsymbol{Der} \wedge\ x = \mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\beta)\}$. Z (7.4.4) i faktu, że $\alpha_{h,\, n \geq h \geq 1}$ jest w zbiorze $\{\alpha_1,\ldots,\alpha_n\}$ wnioskujemy (gdyż na mocy definicji funkcji $\boldsymbol{h}$ mamy $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_{h,\, n \geq h \geq 1}) = \{x; (\exists\, \beta)(\beta \in \Delta(\alpha_{h,\, n \geq h \geq 1}) \wedge \beta \in \boldsymbol{Der} \wedge\ x = \mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\beta)\})$: (7.4.5) $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_{h,\, n \geq h \geq 1})) \subset \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n))$. Ostatecznie z (7.4.1) oraz (7.4.5) wyprowadzamy: (7.5) ${\sim}\Omega^{*}(\alpha_{h,\, n \geq h \geq 1}) \subset {}^{n}\boldsymbol{L} \rightarrow (\exists x)[x \in \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n)) \wedge x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U}) \cup \boldsymbol{U}]$. Z (7.3), (7.4) oraz (7.5) wynika: (7.6) $(\exists x)[x \in \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n)) \wedge x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U}) \cup \boldsymbol{U}]$. Stąd ostatecznie z (7.1) i (7.6) mamy: (8) $\alpha_{h,\, n \geq h \geq 1} \in \boldsymbol{Der} \cup \boldsymbol{Der_{sup}}\ {}^{n}\boldsymbol{L} \rightarrow (\exists x)[x \in \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n)) \wedge x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U}) \cup \boldsymbol{U}]$. Z (7) i (8) dostajemy: (9) $(\exists x)[x \in \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\, \boldsymbol{M(U)}}(\alpha_n)) \wedge x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U}) \cup \boldsymbol{U}]$ ♦

(T. 39) $\quad (\forall d_1,\ldots,d_k)(\forall\, i)[\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/ d_1,\ldots,d_k] \rightarrow (\forall x)[x \in \boldsymbol{h}(\mathbf{VAL}_{i,\, \boldsymbol{M(U)}}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\, \boldsymbol{M(U)}}(d_k)) \rightarrow x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U})]]$.

Dowód: Załóżmy: (1) $\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/ d_1,\ldots,d_k]$, (2) $x \in \boldsymbol{h}(\mathbf{VAL}_{i,\, \boldsymbol{M(U)}}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\, \boldsymbol{M(U)}}(d_k))$. Z (1) wynika: (3) $\{d_1,\ldots,d_k\} \subset \boldsymbol{Der}^{\boldsymbol{I}} \cup \boldsymbol{Der}^{\boldsymbol{I}}{}_{\boldsymbol{sup}}$ (tak jest, gdyż wszystkie reguły LDD-dowodzenia pozwalają jedynie na wyprowadzanie zbiorów wyrażeń derywacyjnych, dla których parametr głębokości przyjmuje wartość jeden). Skoro funkcja $\boldsymbol{h}$ przekształca derywacje (będące wartościami od funkcji wartościowania wyrażeń derywacyjnych) w zbiory jednoelementowe tychże derywacji, w aplikacji zaś do zbiorów derywacji będących zakresami dekompresji derywacji superponowanych (i jednocześnie wartościami tychże derywacji) jest funkcją tożsamościową, to z (2) i (3) wynika: (4) $x \in \boldsymbol{Dr}^{1}(\boldsymbol{U}) \cup \boldsymbol{Dr_{ndk}}(\boldsymbol{U})$ (tak jest, gdyż każde wyrażenie derywacyjne o parametrze głębokości jeden posiada wartość w postaci derywacji niedokończonej lub derywacji z poziomu głębokości równego jeden)♦

(T. 40) $(\forall\alpha_1,\ldots,\alpha_n)\,(\forall d_1,\ldots,d_k)\,(\forall i)[\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/ d_1,\ldots,d_k] \rightarrow (\forall y)[y \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_k)) \rightarrow (\exists x)(x \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M(U)}}(\alpha_n)) \wedge \sim \boldsymbol{Stp}(x) > \boldsymbol{Stp}(y))]]$.

Dowód: Załóżmy: (1) $\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/ d_1,\ldots,d_k]$, (2) $y \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_k))$. Z (1) i (2) wynika: (3) $y \in \boldsymbol{Dr}^1(\boldsymbol{U})$ (tak jest, gdyż w LDD-logice aktywnymi logicznie wyrażeniami derywacyjnymi są wyrażenia o parametrze głębokości równym jeden, a ich wartościami z uwagi na funkcję $\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}$ są derywacje niedokończone lub derywacje z poziomu głębokości równego jeden). Z (3) i (2) wynika: (4) $(\exists d)\,(\exists d_i,\, k \geq i \geq 1)[y = \mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d) \wedge d \in \boldsymbol{\Delta}(d_i)]$. Z (4) dostajemy: (5) $y = \mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_h)$, (6) $d_h \in \boldsymbol{\Delta}(d_i)$. Załóżmy dodatkowo: (6.1) $d_h = d_i$. Z (1) i (6.1) wnioskujemy: (6.2) $\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/ d_h]$. Ponieważ reguły dowodzenia prowadzą od wyrażeń derywacyjnych danego poziomu derywacyjnego do wyrażeń poziomu derywacyjnego wyższego lub równego poziomowi wyjściowemu, więc otrzymujemy: (6.3) $(\exists x)(x \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M(U)}}(\alpha_n)) \wedge \sim \boldsymbol{Stp}(x) > \boldsymbol{Stp}(y))$ (zauważmy, że poziom derywacyjny wyrażenia derywacyjnego wyznacza stopień singletonizacji jego wartości (będącej derywacją z uwagi na funkcję $\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}$) w taki sposób, że im wyższy poziom derywacyjny tym wyższy jest stopień singletonizacji wartości wyrażenia derywacyjnego). Załóżmy dodatkowo: (7.1) $d_h \neq d_i$. Z (7.1) i (6) dostajemy: (7.2) $d_i \in \boldsymbol{Der}_{\boldsymbol{sup}}$. Z (1) i (7.2) wynika: (7.3) $\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/ d_i]$. Jeśli wyrażenie derywacyjne d_i jest otrzymane z jakichś wyrażeń derywacyjnych ze zbioru $\{\alpha_1,\ldots,\alpha_n\}$, poprzez zastosowanie reguły (Unif) o kształcie: $\tau = ({}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots,\ {}^{n+1}\beta_{j,}\ldots,\ {}^{n+1}\beta_{k}],\ {}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots,\ {}^{n+2}\delta_{i}]\ /\!/\ {}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots,\ {}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots,\ {}^{n+2}\delta_{i}],\ldots,\ {}^{n+1}\beta_{k}])$, to ponieważ stopień singletonizacji wartości dowolnego wyrażenia derywacyjnego należącego do zakresów dekompresji wyrażeń: ${}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots,\ {}^{n+1}\beta_{j,}\ldots,\ {}^{n+1}\beta_{k}]$ oraz ${}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots,\ {}^{n+2}\delta_{i}]$ nie jest wyższy od stopnia singletonizacji wartości dowolnego wyrażenia derywacyjnego należącego do zakresu dekompresji superponowanego wyrażenia o postaci: ${}^{n}\alpha[{}^{n+1}\beta_{1,}\ldots,\ {}^{n+1}\beta_{j}[{}^{n+2}\delta_{1,}\ldots,\ {}^{n+2}\delta_{i}],\ldots,\ {}^{n+1}\beta_{k}]$, zatem z (5), (6) i (7.3) mamy: (7.4) $(\exists x)(x \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M(U)}}(\alpha_n)) \wedge \sim \boldsymbol{Stp}(x) > \boldsymbol{Stp}(y))$. Z (6.3) i (7.4) wynika: (8) $(\exists x)(x \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M(U)}}(\alpha_n)) \wedge \sim \boldsymbol{Stp}(x) > \boldsymbol{Stp}(y))$. ♦

(T. 41) $(\forall\alpha_1,\ldots,\alpha_n)\,(\forall d_1,\ldots,d_k)\,(\forall\ i)[\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/ d_1,\ldots,d_k] \rightarrow (\forall y)[y \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_k)) \wedge \Omega(y) \in \boldsymbol{U} \rightarrow (\exists x)(x \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(\alpha_{\ 1})) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M(U)}}(\ \alpha_n)) \wedge (\Omega(x) = \Omega(y) \vee x = \Omega(y)))]]$.

Dowód: Załóżmy: (1) $\mathrm{LDD}\vdash\ [\alpha_1,\ldots,\alpha_n /\!/ d_1,\ldots,d_k]$, (2) $y \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_k))$, (3) $\Omega(y) \in \boldsymbol{U}$. Z (2) i (3) wynika: (4) $(\exists d)[y = \mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d) \wedge \Omega(d) \in {}^{0}\boldsymbol{L}]$. Z (4) otrzymujemy: (5) $y = \mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_i)$, (6) $\Omega(d_i) \in {}^{0}\boldsymbol{L}$ Z (2), (3), (5) i (6) wynika: (7) $d_i \in \{d_1,\ldots,d_k\} \vee (d_i \in \boldsymbol{\Delta}(d_h) \wedge d_h \in \{d_1,\ldots,d_k\} \wedge d_i \neq d_h)$. Załóżmy dodatkowo: (7.1) $d_i \in \{d_1,\ldots,d_k\}$. Zatem, na mocy (1), d_i jest wyprowadzalne z takiego α_i należącego do $\{\alpha_1,\ldots,\alpha_n\}$, czyli (7.2) $\alpha_i \in \{\alpha_1,\ldots,\alpha_n\}$, że: (7.3) $\Omega(d_i) = \Omega(\alpha_i) \vee \alpha_i \in {}^{0}\boldsymbol{L}$ (tak jest, gdyż: (i) reguły (Mult), (Dest), (Int), (Elim) nie wyprowadzają poza zbiór wyrażeń derywacyjnych o tym samym argumencie; (ii) a skoro na mocy (6) $\Omega(d_i) \in {}^{0}\boldsymbol{L}$, to niemożliwe jest wyprowadzenie d_i z jakiegoś wyrażenia derywacyjnego na mocy reguł: (Trans), (Unif), (Dezint); zatem d_i może być wyprowadzalne na mocy reguły (Start) z jakiegoś elementu leksykalnego $\alpha_i \in {}^{0}\boldsymbol{L}$). Załóżmy dodatkowo: (7.3.1) $\Omega(d_i) = \Omega(\alpha_i)$. Z (7.3.1) wynika: (7.3.2) $\Omega(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(d_i)) = \Omega(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(\alpha_i))$. Z definicji funkcji $\boldsymbol{h}$ oraz (7.2) wynika: (7.3.3) $\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(\alpha_i) \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(\alpha_{\ 1})) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,\boldsymbol{M(U)}}(\ \alpha_n))$. Z (7.3.2) i (7.3.3) otrzymujemy: (7.3.4) $(\exists x)(\ x \in \boldsymbol{h}(\mathbf{VAL}_{i,\,\boldsymbol{M(U)}}(\alpha_{\ 1})) \cup$

… ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Zatem z (7.3.1) i (7.3.4) wnioskujemy: (7.4) Ω(d_i) = Ω(α_i) → (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Załóżmy dodatkowo: (7.4.1) $\alpha_i \in {}^{0}\boldsymbol{L}$. Skoro d_i jest wyprowadzalne z α_i, to na mocy (7.4.1) d_i jest wyprowadzalne z α_i w wyniku zastosowania reguły (Start); zatem dostajemy: (7.4.2) α_i = Ω(d_i). Z (7.4.2) i (7.4.1) wynika: (7.4.3) $\mathbf{VAL}_{i,\,M(U)}(\alpha_i) = \Omega(\mathbf{VAL}_{i,\,M(U)}(d_i))$. Ponadto z definicji funkcji $\boldsymbol{h}$ oraz (7.2) wnioskujemy: (7.4.4) $\mathbf{VAL}_{i,\,M(U)}(\alpha_i) \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$. Z (4), (3) i (7.4.4) wynika: (7.4.5) (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Z (7.4.1) i (7.4.5) otrzymujemy: (7.5) $\alpha_i \in {}^{0}\boldsymbol{L}$ → (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Z (7.3), (7.4) i (7.5) odrywamy: (7.6) (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Zatem z (7.1) i (7.6) wyprowadzamy: (8) $d_i \in \{d_1,\ldots,d_k\}$ → (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Załóżmy dodatkowo: (8.1) $d_i \in \boldsymbol{\Delta}(d_h) \wedge d_h \in \{d_1,\ldots,d_k\} \wedge d_i \neq d_h$. Z (8.1) wynika: (8.2) $d_h \in \boldsymbol{Der}_{sup}$. Skoro d_h jest superponowanym wyrażeniem derywacyjnym na mocy (8.2), to skoro $d_i \in \boldsymbol{\Delta}(d_h)$, to otrzymujemy: (8.3) $d_i \in \{\alpha_1,\ldots,\alpha_n\} \vee (d_i \in \boldsymbol{\Delta}(\alpha_i) \wedge \alpha_i \in \{\alpha_1,\ldots,\alpha_n\})$. Załóżmy dodatkowo: (8.3.1) $d_i \in \{\alpha_1,\ldots,\alpha_n\}$. Z (8.3.1) i (5) wynika: (8.3.2) y ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(y) = Ω(y) ∨ y = Ω(y)). Z (8.3.2) dostajemy: (8.3.3) (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Z (8.3.1) i (8.3.3) wyprowadzamy: (8.4) $d_i \in \{\alpha_1,\ldots,\alpha_n\}$ → (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Załóżmy dodatkowo: (8.4.1) $d_i \in \boldsymbol{\Delta}(\alpha_i) \wedge \alpha_i \in \{\alpha_1,\ldots,\alpha_n\}$. Z (5), (8.4.1) i definicji wartościowania wynika: (8.4.2) y ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_i))$. Z (8.4.1), (8.4.2) wynika: (8.4.3) y ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(y) = Ω(y) ∨ y = Ω(y)). Dołączając kwantyfikator szczegółowy do (8.4.3), dostajemy: (8.4.4) (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Zatem z (8.4.1) i (8.4.4) wnioskujemy: (8.5) $d_i \in \boldsymbol{\Delta}(\alpha_i) \wedge \alpha_i \in \{\alpha_1,\ldots,\alpha_n\}$ → (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Z (8.3), (8.4) i (8.5) wynika: (8.6) (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Z kolei z (8.1) i (8.6) dostajemy: (9) $d_i \in \boldsymbol{\Delta}(d_h) \wedge d_h \in \{d_1,\ldots,d_k\} \wedge d_i \neq d_h$ → (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y))). Z (7), (8) i (9) wnioskujemy: (10) (∃x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1))$ ∪ … ∪ $\boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n))$ ∧ (Ω(x) = Ω(y) ∨ x = Ω(y)))♦

(T. 42) $(\forall\alpha_1,\ldots,\alpha_n)\,(\forall d_1,\ldots,d_k)\,(\forall\, i)\{\mathrm{LDD}\vdash [\alpha_1,\ldots,\alpha_n \,//\, d_1,\ldots,d_k] \rightarrow (\forall y)\,[y \in \boldsymbol{h}(\,\mathbf{VAL}_{i,\,M(U)}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(d_k)) \rightarrow [(\forall x)(x \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\,\alpha_n)) \rightarrow \sim \boldsymbol{Stp}(y) > \boldsymbol{Stp}(x)) \rightarrow (\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))]]\}$.

Dowód: Załóżmy: (1) $\mathrm{LDD}\vdash [\alpha_1,\ldots,\alpha_n \,//\, d_1,\ldots,d_k]$, (2) y ∈ $\boldsymbol{h}(\,\mathbf{VAL}_{i,\,M(U)}(d_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(d_k))$, (3) (∀x)(x ∈ $\boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{\mathbf{i},\,M(U)}(\,\alpha_n))$ → ~ $\boldsymbol{Stp}(y) > \boldsymbol{Stp}(x)$). Z (2) wynika: (3) $[y = \mathbf{VAL}_{i,\,M(U)}(d_h) \wedge d_h \in \{d_1,\ldots,d_k\} \wedge d_h \in \boldsymbol{Der}^{I}] \vee [y = \mathbf{VAL}_{i,\,M(U)}(d_z) \wedge (\exists d_i)(d_z \in \boldsymbol{\Delta}(\,d_i) \wedge d_i \in \{d_1,\ldots,d_k\} \wedge d_i \in \boldsymbol{Der}_{sup})]$ (y jest wartością od funkcji wartościowania jakiegoś niesuperponowanego wyrażenia derywacyjnego ze zbioru $\{d_1,\ldots,d_k\}$ lub y jest wartością od funkcji wartościowania jakiegoś wyrażenia derywacyjnego należącego do zakresu dekompresji jakiegoś superponowanego wyrażenia derywacyjnego ze zbioru $\{d_1,\ldots,d_k\}$). Załóżmy dodatkowo: (3.1) $y = \mathbf{VAL}_{i,\,M(U)}(d_h) \wedge d_h \in \{d_1,\ldots,d_k\} \wedge d_h \in \boldsymbol{Der}^{I}$. Z (1) i (3.1) wynika: (3.2) $\mathrm{LDD}\vdash [\alpha_1,\ldots,\alpha_n \,//\, d_h]$. Ponieważ $d_h \in \boldsymbol{Der}^{I}$, więc: (3.3) d_h jest uzyskany z $\alpha_1,\ldots,\alpha_n$ w wyniku stosowania reguł: (Mult), (Elim), (Dest), (Intr), (Start), (Dezint).

Jeśli d_h jest uzyskany z $\alpha_1,\ldots,\alpha_n$ w wyniku stosowania (Mult), (Elim), (Dest), (Intr), to dostajemy: (3.3.1) $(\exists\alpha_i)[\alpha_i \in \{\alpha_1,\ldots,\alpha_n\} \wedge \Omega(\alpha_i) = \Omega(d_h)]$ (tak jest, gdyż wymienione reguły nie wyprowadzają poza poziom leksykalny argumentów wyrażeń derywacyjnych). Z (3.3.1) mamy: (3.3.2) $\alpha_z \in \{\alpha_1,\ldots,\alpha_n\} \wedge \Omega(\alpha_z) = \Omega(d_h)$. Niech: (3.3.3) $\mathbf{VAL}_{i,\,M(U)}(\alpha_z) = z_1$. Zatem z (3.3.2) i (3.3.3) wynika: (3.3.4) $z_1 \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n))$. Jeśli d_h jest uzyskany z $\alpha_1,\ldots,\alpha_n$ w wyniku stosowania (Mult), (Elim), (Dest), (Intr), to: (3.3.5) $\alpha_z \in \boldsymbol{Der^I}$ (wymienione reguły działają bowiem wyłącznie na niesuperponowanych wyrażeniach derywacyjnych). Z (3.3.5) i (3.3.2) $\Omega(\alpha_z) = \Omega(d_h)$, a także (3.3.3) i (3.1) $y = \mathbf{VAL}_{i,\,M(U)}(d_h)$, otrzymujemy: (3.3.6) $\Omega(z) = \Omega(y)$. Z (3.3.6) i (3.3.4) mamy: (3.3.7) $(\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))$. Z (3.3.1) i (3.3.7) wyprowadzamy: (3.4) $(\exists\alpha_i)[\alpha_i \in \{\alpha_1,\ldots,\alpha_n\} \wedge \Omega(\alpha_i) = \Omega(d_h)] \rightarrow (\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))$.

Jeśli d_h jest uzyskany z $\alpha_1,\ldots,\alpha_n$ w wyniku stosowania (Start), (Dezint), to dostajemy: (3.4.1) $(\exists\alpha_i)[\alpha_i \in \{\alpha_1,\ldots,\alpha_n\} \wedge (\Omega(\alpha_i) \in {}^{n}\boldsymbol{L} \vee \alpha_i \in {}^{0}\boldsymbol{L})]$. Z (3.4.1) opuszczając kwantyfikator, na mocy (Def. 38 i Def. 32) oraz tego, że reguła (Dezint) operuje na superponowanych wyrażeniach derywacyjnych, otrzymujemy: (3.4.2) $[\boldsymbol{Stp}(\mathbf{VAL}_{i,\,M(U)}(\alpha_z)) = \mathbf{k} \wedge \mathbf{k} > \mathbf{n}] \vee \boldsymbol{Stp}(\mathbf{VAL}_{i,\,M(U)}(\alpha_z)) = 0$. Niech: (3.4.3) $\mathbf{VAL}_{i,\,M(U)}(\alpha_z) = z_1$. Jeśli: (3.4.3.1) $\boldsymbol{Stp}(\mathbf{VAL}_{i,\,M(U)}(\alpha_z)) = \mathbf{0}$, to (3.4.3.2) $z_1 \in \boldsymbol{U}$. Wówczas skoro na mocy (3.1) $y = \mathbf{VAL}_{i,\,M(U)}(d_h) \wedge d_h \in \boldsymbol{Der^I}$, to z (3.4.3.2) mamy: (3.4.3.3) $\boldsymbol{Stp}(y) > \boldsymbol{Stp}(z_1)$. Z kolei z (3) oraz tego, że $\alpha_z \in \{\alpha_1,\ldots,\alpha_n\}$, mamy: (3.4.3.4) $\sim\boldsymbol{Stp}(y) > \boldsymbol{Stp}(z_1)$. Między ostatnimi wierszami zachodzi sprzeczność. Zatem otrzymujemy: (3.4.4) $\boldsymbol{Stp}(\mathbf{VAL}_{i,\,M(U)}(\alpha_z)) \neq 0$. Stąd d_h nie jest uzyskany z $\alpha_1,\ldots,\alpha_n$ w wyniku stosowania (Start). Zatem d_h jest uzyskany z $\alpha_1,\ldots,\alpha_n$ w wyniku stosowania (Dezint). Zatem otrzymujemy: (3.4.5) $(\exists\alpha_i)[\alpha_i \in \{\alpha_1,\ldots,\alpha_n\} \wedge d_h \in \boldsymbol{\Delta}(\alpha_i)]$. Jeśli więc d_h należy do zakresu dekompresji jakiegoś $\alpha_i \in \{\alpha_1,\ldots,\alpha_n\}$ (na mocy (3.4.5)), to: (3.4.6) $\mathbf{VAL}_{i,\,M(U)}(d_h) \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n))$. A skoro na mocy (3.1) $y = \mathbf{VAL}_{i,\,M(U)}(d_h)$, to z (3.4.6) wnioskujemy: (3.4.7) $(\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))$. Zatem z (3.4.1) i (3.4.7) wyprowadzamy: (3.5) $(\exists\alpha_i)[\alpha_i \in \{\alpha_1,\ldots,\alpha_n\} \wedge (\Omega(\alpha_i) \in {}^{n}\boldsymbol{L} \vee \alpha_i \in {}^{0}\boldsymbol{L}))] \rightarrow (\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))$.

Skoro jest tak: (3.6) d_h jest uzyskany z $\alpha_1,\ldots,\alpha_n$ w wyniku stosowania reguł: (Mult), (Elim), (Dest), (Intr), (Start), (Dezint), to $(\exists\alpha_i)[\alpha_i \in \{\alpha_1,\ldots,\alpha_n\} \wedge \Omega(\alpha_i) = \Omega(d_h)] \vee (\exists\alpha_i)[\alpha_i \in \{\alpha_1,\ldots,\alpha_n\} \wedge (\Omega(\alpha_i) \in {}^{n}\boldsymbol{L} \vee \alpha_i \in {}^{0}\boldsymbol{L})]$. Z (3.3), (3.6), (3.4) i (3.5) wyprowadzamy: (3.7) $(\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))$. Z (3.1) i (3.7) mamy: (4) $[y = \mathbf{VAL}_{i,\,M(U)}(d_h) \wedge d_h \in \{d_1,\ldots,d_k\} \wedge d_h \in \boldsymbol{Der^I}] \rightarrow (\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))$.

Załóżmy dodatkowo: (4.1) $y = \mathbf{VAL}_{i,\,M(U)}(d_z) \wedge (\exists d_i)(d_z \in \boldsymbol{\Delta}(d_i) \wedge d_i \in \{d_1,\ldots,d_k\} \wedge d_i \in \boldsymbol{Der_{sup}})$. Z (4.1) inferujemy: (4.2) $d_z \in \boldsymbol{\Delta}(d_t)$, (4.3) $d_t \in \{d_1,\ldots,d_k\}$, (4.4) $d_t \in \boldsymbol{Der_{sup}}$. Jeśli na mocy (4.4) d_t jest superponowanym wyrażeniem derywacyjnym, to jest ono, na mocy (4.3) i (1), wyprowadzalne z $\alpha_1,\ldots,\alpha_n$ dzięki stosowaniu reguły (Unif). Zatem otrzymujemy: (4.5) $(\exists\,\alpha_i, \alpha_j)[\alpha_i, \alpha_j \in \{\alpha_1,\ldots,\alpha_n\} \wedge \boldsymbol{\Delta}(d_t) = \boldsymbol{\Delta}(\alpha_i) \cup \boldsymbol{\Delta}(\alpha_j)]$. Z (4.2) i (4.5) wynika: (4.6) $d_z \in \boldsymbol{\Delta}(\alpha_i) \cup \boldsymbol{\Delta}(\alpha_j) \wedge \alpha_i, \alpha_j \in \{\alpha_1,\ldots,\alpha_n\}$. Z (4.6), (4.1) i definicji funkcji wartościowania mamy: (4.7) $y \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n))$. Z (4.7) mamy: (4.8) $(\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))$. Z (4.1) i (4.8) dostajemy: (5) $[y = \mathbf{VAL}_{i,\,M(U)}(d_z) \wedge (\exists d_i)(d_z \in \boldsymbol{\Delta}(d_i) \wedge d_i \in \{d_1,\ldots,d_k\} \wedge d_i \in \boldsymbol{Der_{sup}})] \rightarrow (\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))$. Z (3), (4) i (5) wynika: (6) $(\exists z)(z \in \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_1)) \cup \ldots \cup \boldsymbol{h}(\mathbf{VAL}_{i,\,M(U)}(\alpha_n)) \wedge \Omega(z) = \Omega(y))$ ♦

Z twierdzeń (T. 37)-(T. 42) oraz definicji ***Taut*** dostajemy:

(T. 43) $(\forall\alpha)[\mathrm{LDD}\vdash \alpha \rightarrow \alpha \in \boldsymbol{Taut}]$.

Dowód twierdzenia odwrotnego: $(\forall\alpha)\,[\alpha \in \boldsymbol{Taut} \rightarrow \mathrm{LDD}\vdash \alpha]$, wymaga pokazania tego, że każde wyrażenie transformacyjne, które funkcja wartościowania przekształca w transformację wyróżnioną, a więc należącą do klasy $\boldsymbol{Tr^*(U)}$, jest LDD-dowodliwe:

(T. 44) $(\forall\alpha)\,[\alpha \in \boldsymbol{Taut} \rightarrow \mathrm{LDD}\vdash \alpha]$.

Szkic dowodu: Najpierw wykazać trzeba, że wszystkie LDD-dowodliwe formuły transformacyjne posiadają pewne osobliwe cechy syntaktyczne, których koniunkcja definiuje zbiór LDD-dowodliwych formuł. Następnie zaś wystarczy wykazać, że każda tautologia posiada również te cechy.

Lista tych definicyjnych cech LDD-dowodliwych formuł transformacyjnych prezentuje się następująco: (1) Argument LDD-dowodliwej formuły transformacyjnej jest zbiorem wyrażeń leksykalnych poziomu zerowego lub wyrażeń derywacyjnych, których zakresy dekompresji składają się z niesuperponowanych wyrażeń derywacyjnych, dla których parametr głębokości przyjmuje wartość jeden. (2) Wartość LDD-dowodliwej formuły transformacyjnej jest zbiorem wyrażeń derywacyjnych, których zakresy dekompresji składają się z niesuperponowanych wyrażeń derywacyjnych, dla których parametr głębokości przyjmuje wartość jeden. (3) Zdefiniujmy funkcję przyporządkowującą poziom derywacyjny wyrażeniom leksykalnym lub niesuperponowanym wyrażeniom derywacyjnym w następujący sposób: (i) jeśli wyrażenie leksykalne jest n-tego poziomu leksykalnego, to jest również n-tego poziomu derywacyjnego; (ii) niesuperponowane wyrażenie derywacyjne jest n-tego poziomu derywacyjnego wtedy i tylko wtedy, gdy zmienne wyrażenia leksykalne składające się na jego wartość są n-tego poziomu leksykalnego, lub jeśli wartością wyrażenia derywacyjnego jest stała stop, to jego argument jest n-tego poziomu leksykalnego. Okazuje się, że każda LDD-dowodliwa formuła transformacyjna posiada taką własność, że dla dowolnego wyrażenia derywacyjnego należącego do dowolnego zakresu dekompresji wyrażenia derywacyjnego stanowiącego element wartości LDD-dowodliwej formuły transformacyjnej istnieje wyrażenie leksykalne lub wyrażenie derywacyjne należące do zakresu dekompresji jakiegoś wyrażenia derywacyjnego będącego składnikiem argumentu danej formuły transformacyjnej, którego poziom derywacyjny nie jest większy od poziomu derywacyjnego wyrażenia należącego do zakresu dekompresji wyrażenia derywacyjnego stanowiącego element wartości LDD-dowodliwej formuły transformacyjnej. Innymi słowy, jeśli wszystkie wyrażenia derywacyjne należące do wartości LDD-dowodliwej formuły transformacyjnej podda się dekompresji, to uzyska się pewien zbiór niesuperponowanych wyrażeń derywacyjnych; taki zbiór nazwijmy zdekompresowaną wartością formuły transformacyjnej. Podobną operację można wykonać na wszystkich formułach derywacyjnych należących do argumentu danej formuły transformacyjnej. Jeśli do argumentu formuły transformacyjnej przynależą wyrażenia leksykalne, to sumę zbioru tych wyrażeń oraz zbioru powstałego w wyniku wykonania operacji dekompresji nazwiemy zdekompresowanym argumentem formuły transformacyjnej. Otóż, każda LDD-dowodliwa formuła transformacyjna ma taką własność, że dla każdego elementu należącego do jej zdekompresowanej wartości istnieje element należący do jej zdekompresowanego argumentu taki, że jego poziom derywacyjny nie jest większy od poziomu derywacyjnego elementu należącego do zdekompresowanej wartości. (4) Dla każdego wyrażenia derywacyjnego należącego do zdekompresowanej wartości LDD-dowodliwej formuły transformacyjnej takiego, którego argument jest wyrażeniem leksykalnym poziomu zerowego, jest tak, że to wyrażenie należy do argumentu LDD-do-

wodliwej formuły transformacyjnej lub jest identyczne z jakimś argumentem dowolnego wyrażenia derywacyjnego należącego do zdekompresowanego argumentu LDD-dowodliwej formuły transformacyjnej. (5) Jeśli żadne wyrażenie derywacyjne należące do zdekompresowanej wartości LDD-dowodliwej formuły transformacyjnej nie jest wyższego poziomu derywacyjnego od dowolnego z wyrażeń należących do zdekompresowanego argumentu danej formuły transformacyjnej, to wśród wyrażeń należących do zdekompresowanego argumentu formuły transformacyjnej istnieją takie, których argumenty są identyczne z argumentami wszystkich wyrażeń derywacyjnych należących do zdekompresowanej wartości LDD-dowodliwej formuły transformacyjnej.

Skoro tautologią LDD-logiki jest dowolna formuła transformacyjna, której wartością z uwagi na funkcję wartościowania w dowolnym modelu jest wyróżniona transformacja, to każda tautologia posiada każdą z wymienionych cech LDD-dowodliwej formuły transformacyjnej. Tak jest, gdyż zachodzą następujące korelacje: (1) Jeśli wartość z uwagi na funkcję wartościowania formuły transformacyjnej spełnia drugi warunek z definicji transformacji wyróżnionej, to formuła transformacyjna posiada pierwszą z wyszczególnionych w dowodzie cech. (2) Jeśli wartość z uwagi na funkcję wartościowania formuły transformacyjnej spełnia trzeci warunek z definicji transformacji wyróżnionej, to formuła transformacyjna posiada drugą z wyszczególnionych w dowodzie cech. I podobnie, czwarty warunek w definicji transformacji wyróżnionej jest skorelowany z trzecią cechą LDD-dowodliwych formuł transformacyjnych; piąty zaś warunek — z czwartą cechą i w końcu szósty warunek z piątą cechą z wyżej wyszczególnionych. ♦

LDD-logika posiada więc teoriomnogościowy model semantyczny. Innymi słowy, posługiwanie się drzewami derywacyjnymi przez umysł jest w istocie aktywnością generowania struktur teoriomnogościowych o pewnych osobliwych cechach strukturalnych.

6. ZAKOŃCZENIE

LDD-logikę można rozszerzać na wiele sposobów. Typowym sposobem rozszerzania LDD-logiki jest jej wzmacnianie poprzez wprowadzanie nowych reguł inferencji rozszerzających klasę aktywnych logicznie wyrażeń derywacyjnych czyli, klasę $\boldsymbol{Der^{1}} \cup \boldsymbol{Der^{stand}_{sup}}$. Na przykład, za pomocą dodatkowej reguły introdukcji o postaci: ${}^{n}\alpha\mathbf{[0]}$ // ${}^{n}\alpha\mathbf{[}{}^{n+2}\beta\mathbf{]}$ wraz z regułą multiplikacji można dowodzić formuły transformacyjne, których wartościami są wyrażenia derywacyjne należące do zbioru $\boldsymbol{Der^{2}}$. LDD-logiki mogą więc być o różnych znacznikach derywacyjnych. Znacznikiem derywacyjnym LDD-logiki jest dowolny zbiór liczb naturalnych, którego elementem jest liczba jeden. Znacznik wyraża o danej LDD-logice to, o jakiej głębokości wyrażenia derywacyjne są aktywne logicznie. Dla przykładu, znacznik kształtu: **[0,1]**, wyraża to, że aktywnymi wyrażeniami derywacyjnymi danej LDD-logiki są wyrażenia derywacyjne klas: $\boldsymbol{Der^{0}}$, $\boldsymbol{Der^{1}}$. Przedstawiona w pracy LDD-logika posiada znacznik **[1]**. Można również mówić o nieskończonych LDD-logikach, których znaczniki derywacyjne są nieskończonymi podzbiorami zbioru liczb naturalnych, ale nieidentycznymi ze zbiorem liczb naturalnych. Każda z LDD-logik wyznacza do-

puszczalne sposoby konstruowania drzew derywacyjnych, na przykład — drzewa z przeskokami derywacyjnymi.

BIBLIOGRAFIA

Bolter D. J. (1990), *Człowiek Turinga*, (tłum. T. Goban-Klas), Warszawa, Państwowy Instytut Wydawniczy.

Buchowski M., (red.) (1993), *Amerykańska antropologia kognitywna,* (red. M. Buchowski), Warszawa, Instytut Kultury.

Chomsky N. (1982), *Zagadnienia teorii składni*, (przeł. I. Jakubczak), Wrocław, Ossolineum.

Harel D. (2001), *Rzecz o istocie informatyki. Algorytmika*, (tłum. Z. Weiss, P. Carlson), Warszawa, Wydawnictwo Naukowo-Techniczne.

Krysztofiak W. (2006a), *Gramatyka dyskursu filozoficznego. Kognitywistyczne studium historii filozofii,* Warszawa, Wydawnictwo Naukowe Semper.

Krysztofiak W. (2006b), „Metodologiczne problemy filologicznej historii filozofii. Odpowiedź Jerzemu Pawliszcze", *Filozofia Nauki*, Rok XIV, Nr 3(55), s. 151-159.

Macnamara J. (1993), *Logika i psychologia*, (tłum. M. Zagrodzki), Warszawa, PWN.

Pawliszcze J. (2006), „Wojciecha Krysztofiaka *Gramatyka dyskursu filozoficznego*", *Filozofia Nauki*, Rok XIV, Nr 3(55), s.143-150.

Piaget J., Inhelder B. (1993), *Psychologia dziecka*, (tłum. Z. Zakrzewska), Warszawa, Wydawnictwo Siedmioróg.

Suszko R. (1998), „Formalna teoria wartości logicznych I", [w:] R. Suszko, *Wybór pism*, (red. M. Omyła), Warszawa, Polskie Towarzystwo Semiotyczne.