Filozofia Nauki
Rok XXIII, 2015, Nr 3(91)

Wojciech Krysztofiak

Logiczna składnia liczebnika
Część II: Formatowanie i przetwarzanie liczebnikowych struktur głębokich*

Celem artykułu jest przedstawienie formalnej teorii formatowania oraz przetwarzania reprezentacji umysłowych liczebników — zarówno cyfrowych, jak i słownych. Reprezentacjami umysłowymi liczebników są ich pewnego rodzaju „zapisy" w umyśle i mózgu przeprowadzone zgodnie z określonymi regułami gramatycznymi. Można więc powiedzieć, że istnieje język reprezentacji umysłowych liczebników, który jest przekładalny na rozmaite języki cyfrowe oraz na różne języki liczebników słownych wkomponowane, jako szczególnego rodzaju podstruktury, w języki etniczne.

Zgodnie z paradygmatem kognitywistycznym w umysłach działa mechanizm przekształcania (transformowania) reprezentacji mentalnych w odpowiadające im struktury danego języka etnicznego lub cyfrowego. Struktury te można określić mianem powierzchniowych. Każdy liczebnik ma więc swoją strukturę powierzchniową, która jest wytwarzana (generowana) z określonych liczebnikowych reprezentacji umysłowych. Sposoby wytwarzania struktur powierzchniowych liczebnika z reprezentacji umysłowych różnią się w zależności od języka, do którego należą owe struktury. Otóż formalna teoria liczebników nie opisuje ani struktur powierzchniowych liczebników, ani sposobów ich wytwarzania z reprezentacji umysłowych. Tym zajmują się takie dyscypliny, jak lingwistyka i psycholingwistyka. Teoria logicznej składni liczebnika odpowiada na zgoła odmienne pytania:

* Artykuł stanowi kontynuację pracy (Krysztofiak 2012b). Został napisany w ramach realizacji grantu Narodowego Centrum Nauki nr 2011/01/B/HS1/04029.

(i) dotyczące formatowania reprezentacji liczebników: jakimi podstawowymi jednostkami leksykalnymi oraz jakimi regułami gramatycznymi posługuje się umysł, syntetyzując reprezentacje umysłowe liczebników?

(ii) dotyczące przetwarzania reprezentacji liczebników: według jakich reguł inferencji umysł przetwarza liczebnikowe reprezentacje umysłowe na inne, kodenotacyjne (równoważne zakresowo) reprezentacje umysłowe liczebników w różnych praktykach obliczeniowych? Według jakich mechanizmów rekurencyjnych umysł porównuje reprezentacje liczebników, których powierzchniowym przejawem są stwierdzenia, że dany liczebnik desygnuje większą liczbę lub liczność niż inne liczebniki?

W pierwszej części artykułu przedstawiam język formatowania struktur głębokich liczebników. Podaję przykłady różnych typów struktur głębokich liczebników, a także procesów ich przetwarzania. W drugiej części konstruuję logikę liczebników (LL), czyli system przetwarzania ich struktur głębokich. Zakończenie poświęcone jest — jako zapowiedź trzeciej części studium — wyszczególnieniu podstawowych semantycznych problemów badawczych, które powinny zostać rozwiązane za pomocą przedstawionej teorii.

Podstawowy wniosek artykułu głosi, że wykonując czynności porównywania denotacyjnego liczebników, a także stosując rozmaite algorytmy pisemnego dodawania (czy nawet mnożenia) liczebników cyfrowych, umysł nie stosuje ani *explicite*, ani *implicite* aparatu dedukcyjnego arytmetyki Peana (PA). Wykonywanie przez umysł podstawowych operacji obliczeniowych na liczebnikach jest regulowane przez mechanizmy formatowania i przetwarzania struktur głębokich liczebników. Mechanizmy te są wyznaczone przez reguły gramatyczne i inferencyjne składające się na rachunek logiki liczebników. Opis gramatyki tego systemu wymaga jednak odwołania do PA, ponieważ kategoria funktorów pozycji składniowych, a więc elementów, z których są między innymi syntetyzowane struktury głębokie liczebników, tworzy algebrę izomorficzną ze standardowymi modelami PA. W tym świetle czynności gramatycznego formatowania struktur głębokich liczebników wymagają zastosowania PA, podczas gdy czynności przetwarzania tych struktur zakładają jedynie rachunek logiki liczebników. Innymi słowy, formatowanie liczebników, czyli zapisywanie (kodowanie) oraz odczytywanie (dekodowanie) ich w umyśle, wymaga użycia arytmetyki, natomiast ich przekształcanie w inne liczebniki nie ma charakteru arytmetycznego.

Wskazana własność liczebników przejawia się między innymi w tym, że aby odnieść się przedmiotowo do obiektu liczbowego, umysł musi dokonać równocześnie metaobliczeń znaków cyfrowych występujących w liczebniku cyfrowym użytym w akcie referencji do danego obiektu liczbowego. Na przykład, odnosząc się do liczby *100000000001*, umysł musi policzyć, ile cyfr występuje w liczebniku „100000000001". Uogólniając, posługiwanie się liczebnikami cyfrowymi w celu odniesienia się do obiektów arytmetycznych (liczb, liczności, funkcji określonych na liczbach) wymaga odniesienia się (metareferencji) do samych liczebników jako obiektów arytmetycznych. Natomiast posługiwanie się liczebnikami w obliczeniach

nie wymaga odnoszenia się do nich jako obiektów arytmetycznych, ponieważ poprawność obliczeń jest regulowana wyłącznie rachunkiem logiki liczebników.

1. FORMATOWANIE STRUKTUR GŁĘBOKICH LICZEBNIKÓW

Jako obiekty gramatyczne reprezentacje umysłowe liczebników mają struktury gramatyczne, które będę nazywał strukturami głębokimi liczebników. Kwestie formatowania i przetwarzania reprezentacji umysłowych liczebników sprowadzają się więc do odpowiednio przeformułowanych zagadnień formatowania i przetwarzania struktur głębokich liczebników. Każdy liczebnik w sensie powierzchniowym jest skorelowany z wieloma reprezentacjami umysłowymi, a tym samym z wieloma strukturami głębokimi. To, że liczebniki cyfrowe są przekładalne na liczebniki słowne i *vice versa*, można właśnie wyjaśnić, odwołując się do pojęcia struktur głębokich liczebników. Dwa liczebniki — cyfrowy oraz słowny — są wzajemnie przekładalne z zachowaniem kodenotacyjności wtedy i tylko wtedy, gdy skorelowane są z co najmniej jedną tą samą strukturą głęboką. Proces umysłowy przekładu liczebnika cyfrowego na równoważny zakresowo liczebnik słowny można wówczas modelować jako proces aktywacji (lub syntezy) struktury głębokiej liczebnika cyfrowego, a następnie jako proces jej przekształcenia w odpowiednią reprezentację werbalną danego systemu reprezentacji językowych, również zakodowanych w danym umyśle.

Skoro kompetentny numerycznie umysł jest w stanie porównywać dowolne liczebniki pod względem tego, czy denotują tę samą liczbę (liczność), czy też jeden z nich denotuje liczbę (liczność) większą od liczby (liczności) denotowanej przez inny liczebnik, to należy założyć, że umysł podczas przeprowadzania czynności porównywania liczebników jednocześnie dokonuje rozmaitych operacji przetwarzania struktur głębokich liczebników. Takie operacje muszą być przeprowadzane zgodnie z określonymi regułami. Stąd należy przypuszczać, że istnieje logika struktur głębokich liczebników rozumiana jako odpowiedni system reguł ich przetwarzania, umożliwiający umysłowi dokonywanie denotacyjnych porównań liczebników pod kątem równości lub relacji większości i mniejszości zachodzących między liczbami (licznościami) denotowanymi przez liczebniki. Istnienie takiej logiki liczebników zakłada rekurencyjny sposób formatowania liczebnikowych struktur głębokich w umyśle.

Struktury głębokie liczebników są rozumiane jako „przedmioty umysłowe" pozostające w relacjach asocjacji (rozumianych semantycznie) do reprezentacji liczbowych (liczności, liczb porządkowych oraz wielkości) również zakodowanych w umyśle. Procesy umysłowe przetwarzania struktur głębokich dzięki relacji asocjacji uruchamiają przetwarzanie reprezentacji liczbowych. Dopiero reprezentacje liczbowe oraz relacje zachodzące między nimi znajdują się w stosunku korespondencji semantycznej do modeli semantycznych arytmetyki PA w sensie Tarskiego[1]. Dlatego

[1] W pracy (Krysztofiak 2015a) przedstawiony został model takich relacji zachodzących między reprezentacjami liczebnikowymi, liczbowymi oraz modelami semantycznymi PA.

też ostatecznie „poprawność poznawcza" czynności przetwarzania liczebników jest funkcją poprawności procesów przetwarzania ich struktur głębokich, a ta z kolei jest wyznaczana przez asocjację struktur głębokich z reprezentacjami liczbowymi, które pozostają w relacji korespondencji do rozmaitych fragmentów modeli semantycznych PA.

1.1. Syntaktyka struktur głębokich liczebników

Struktury głębokie liczebników zasadzają się na trzech kategoriach wyrażeń. Są to:

(i) cyfry elementarne (elementarne leksemy cyfrowe),

(ii) funktory pozycji składniowych,

(iii) funktor planu pozycyjnego.

W związku z tym w języku formalnym, który ma służyć do opisu formatu liczebnikowych struktur głębokich, powinny znaleźć się wyrażenia następujących typów:

(i) zmienne oraz stałe przebiegające uniwersum cyfr elementarnych danego systemu zapisu liczebników,

(ii) zmienne oraz stałe przebiegające uniwersum funktorów pozycji składniowych w strukturach głębokich liczebników,

(iii) stała desygnująca funktor planu pozycyjnego struktury głębokiej liczebnika.

Niech symbole o postaci „**0**", „**1**", „**2**", …, „$\mathbf{c_{max}}$" oznaczają kolejne cyfry elementarne danego systemu zapisu liczebników[2]. Cyfra $\mathbf{c_{max}}$ jest najwyższą cyfrą elementarną na gruncie danego systemu zapisu liczebników. $\mathbf{c_{max}}$ może się różnić w zależności od systemu. Symbole o postaci „*c*", „c_1", „c_2" to zmienne przebiegające uniwersum cyfr elementarnych danego systemu zapisu cyfr. Symbole o postaci „$\mathbf{O}_0$", „$\mathbf{O}_1$", „$\mathbf{O}_2$" oznaczają kolejne funktory pozycji składniowych w strukturach głębokich liczebników. $\mathbf{O}_0$ jest funktorem pozycji zerowej, $\mathbf{O}_1$ jest funktorem pozycji jedności, a na przykład $\mathbf{O}_{24}$ jest funktorem pozycji dwudziestekczwórek[3]. Symbole o postaci „O_i", „O_j", „O_k" są zmiennymi przebiegającymi uniwersum funktorów pozycji składniowych w strukturach głębokich liczebników. Wyrażenia odnoszące się do funktorów pozycji syntaktycznych są więc wyrażeniami złożonymi z wyrażenia „**O**" lub „*O*" oraz dowolnego indeksu będącego numerem funktora lub zmienną

[2] Jeśli więc *0*, *1*, *2*, … są kolejnymi liczbami naturalnymi, to można poprawnie twierdzić, że **0** desygnuje *0*, **1** desygnuje *1*, **2** desygnuje *2* itd.

[3] Numery funktorów pozycji składniowych są wskaźnikami sposobów działania tych funktorów. Na przykład, funktor $\mathbf{O}_{12}$ działa semantycznie na swój argument w taki sposób, że argument wskazuje na liczbę tuzinów (dwunastek) w danej strukturze głębokiej jakiegoś liczebnika (np. liczebnika *trzy tuziny*).

przebiegającą zbiór tychże numerów. Symbol „P" oznacza funktor planu pozycyjnego struktury głębokiej liczebnika.

Elementarne stałe cyfrowe dowolnego systemu zapisu liczebników są traktowane jako wyrażenia należące do kategorii wyrażeń nazwowych *n*. Funktory pozycji składniowych uznaje się za funktory jednoargumentowe, których wartościami są wyrażenia kategorii *k*, a argumentami — stałe bądź zmienne cyfrowe lub wyrażenia kategorii *k*. Funktory pozycji charakteryzują się więc hybrydowym indeksem kategorialnym o postaci $k/(k \vee n)$. Zakłada się, że każda elementarna stała cyfrowa danego systemu zapisu liczebników może być argumentem dowolnego funktora pozycji składniowej. Oznacza to, że wszystkie cyfry elementarne są wzajemnie podstawialne z zachowaniem poprawności składniowej w każdej strukturze głębokiej liczebnika[4]. Można sądzić, że systemy zapisu liczebników (w szczególności — liczebników cyfrowych), które spełniają ten warunek, naśladują system arabski. To, że funktory pozycji składniowych mają indeks $k/(k \vee n)$, czyni je podatnymi na operacje iteracji i permutacji (zgodnie z regułą MP gramatyki kategorialnej Ajdukiewicza–Bar-Hillela w wersji rozszerzonej dla funktorów hybrydowych[5]).

Funktor planu pozycyjnego P jest funktorem anadycznym, którego argumentami są wszystkie występujące w danej strukturze głębokiej wyrażenia utworzone za pomocą funktorów pozycji składniowych. Funktor ten tworzy wyrażenia kategorii odmiennej niż kategoria *n*: można ją nazwać kategorią liczebników — *l*.

Język, w którym formatowane są struktury głębokie liczebników, obejmuje więc trzy elementarne kategorie składniowe wyrażeń. Ponieważ struktury głębokie zawierają zmienną liczbę pozycji składniowych i cyfr występujących na tych pozycjach, liczba argumentów funktora P jest zmienna, ale zawsze skończona. Indeks kategorialny funktora P ma postać $l/(k, \ldots, k)$. Oznacza to, że liczebniki, których strukturami głębokimi są wyłącznie wyrażenia utworzone za pomocą funktora planu pozycyjnego, nie są wyrażeniami tej samej kategorii co elementarne leksemy liczebnikowe danego języka. W projektowanym rachunku struktury głębokie liczebników pełnią rolę formuł przetwarzanych przez umysł w procesach obliczeniowych.

Można zarzucić przedstawionemu ujęciu składni struktur głębokich liczebników, że wprowadza funktory pozycji składniowych o hybrydowym indeksie kategorialnym $k/(k \vee n)$. Tę „dziwność" syntaktyczną można ominąć, wprowadzając do języka, w którym formatowane są głębokie struktury liczebnikowe, kategorię funktorów składania funktorów pozycyjnych. Wówczas każdą permutację lub iterację funkto-

---

[4] Wydaje się, że system cyfr rzymskich nie spełnia tego warunku. Np. cyfra elementarna „V" nie może być sensownie podstawiona pod dowolną inną cyfrę w dowolnym liczebniku złożonym.

[5] Reguła MP dostosowana do indeksów kategorialnych odpowiadających funktorom hybrydowym przyjmuje postać dwóch warunków: (i) $\alpha/\alpha\vee\beta$; $\alpha \Rightarrow \alpha$ oraz (ii) $\alpha/\alpha\vee\beta$; $\beta \Rightarrow \alpha$, gdzie α i β są dowolnymi indeksami kategorialnymi, $\alpha/\alpha\vee\beta$ jest funktorowym indeksem hybrydowym, a $\Rightarrow$ to funkcja redukcji rządku indeksów (po lewej stronie) do pewnego ustalonego indeksu (po prawej stronie). Reguła MP w standardowym sformułowaniu przyjmuje postać α/β; $\beta \Rightarrow \alpha$. Na temat własności reguły MP — zob. Humberstone 2005: 283.

rów pozycji składniowych można by traktować jako — utworzony za pomocą operacji składania — złożony funktor pozycji składniowej. Na przykład, konkatenacja funktorów o postaci $\mathbf{O}_{10}\mathbf{O}_{10}\mathbf{O}_{100}$ stanowiłaby wyrażenie, które powstaje ze złożenia f(f($\mathbf{O}_{10}$), f($\mathbf{O}_{10}\mathbf{O}_{100}$)). W tym wypadku operator f byłby swoistym klejem zespalającym iteracje lub permutacje funktorów pozycji składniowych, oddawanym za pomocą indeksu kategorialnego (*k/n*)/(*k/n*), a każdy funktor pozycji składniowej byłby skorelowany z indeksem *k/n*. Taki sposób modelowania formalnego liczebnikowych struktur głębokich komplikowałby jednak ich gramatykę oraz nie odzwierciedlałby potwierdzonego empirycznie faktu nabywania umiejętności mnożenia liczebników w późniejszej fazie rozwoju matematycznego. Operator f, który składałby funktory pozycji składniowej, byłby bowiem operatorem ich mnożenia[6].

Niech **N** stanowi klasę cyfr elementarnych danego systemu zapisu liczebników (rozumianych jako reprezentacje umysłowe odpowiadających im leksemów liczebnikowych), **K** klasę wyrażeń kategorii *k* systemu zapisu struktur głębokich liczebników, **F** klasę wszystkich funktorów pozycji składniowej, a **L** klasę wszystkich struktur głębokich liczebników. Mechanizm wytwarzania wyrażeń kategorii **L** (czyli struktur głębokich liczebników) przebiega zgodnie z następującymi warunkami definicyjnymi:

(Df. **L**) **(i)** $\mathbf{N} \subset \mathbf{K}$

(ii) $\alpha \in \mathbf{K} \wedge \beta \in \mathbf{F} \rightarrow \beta(\alpha) \in \mathbf{K}$

(iii) $\alpha_1 \in \mathbf{K} - \mathbf{N} \wedge \ldots \wedge \alpha_k \in \mathbf{K} - \mathbf{N} \rightarrow \mathrm{P}[\alpha_1, \ldots, \alpha_k] \in \mathbf{L}$

Przyjmijmy następującą konwencję definicyjną:

(Df. 2) $\beta_1 \ldots \beta_k(\alpha)$ jest skrótem $\beta_1(\ldots(\beta_k(\alpha))\ldots)$.

Zgodnie z (Df. 2), zamiast pisać np. $\mathbf{O}_{10}(\mathbf{O}_{10}(\mathbf{O}_{100}(\mathbf{c_5})))$, można użyć skróconej wersji z mniejszą liczbą nawiasów: $\mathbf{O}_{10}\mathbf{O}_{10}\mathbf{O}_{100}(\mathbf{c_5})$.

Działanie mechanizmu wytwarzania struktur głębokich przedstawmy, posiłkując się następującymi przykładami: skoro cyfra **2** należy do klasy **N**, a funktor pozycji $\mathbf{O}_2$ należy do klasy **F**, to zgodnie z (i)-(ii) wyrażenie $\mathbf{O}_2(\mathbf{2})$ należy do klasy **K**. Ponieważ $\mathbf{O}_2(\mathbf{2})$ należy do klasy **K**, a $\mathbf{O}_{10}$ należy do **F**, to $\mathbf{O}_{10}\mathbf{O}_2(\mathbf{2})$ należy do **K**. Zgodnie z warunkiem (iii) P[$\mathbf{O}_{10}\mathbf{O}_2(\mathbf{2})$] należy do **L**. Należy zauważyć, że P[**2**] nie jest poprawnie zbudowaną strukturą liczebnikową, ponieważ **2** nie należy do $\mathbf{K} - \mathbf{N}$ (jako że należy do **N**). Uogólniając, nie istnieją struktury liczebnikowe o postaci P[$c_1, \ldots, c_i$].

---

[6] Akapit powstał pod wpływem dyskusji z recenzentem tego artykułu. Uszczegółowiając te uwagi, należałoby powiedzieć, że np. rozumienie liczebników *dwieście dziesiątek*, o strukturze $\mathbf{O}_{10}\mathbf{O}_{100}(\mathbf{2})$, oraz *dwadzieścia setek*, o strukturze $\mathbf{O}_{100}\mathbf{O}_{10}(\mathbf{2})$, wymagałoby od użytkownika języka umiejętności mnożenia. Innymi słowy, umysł na mocy samego syntetyzowania lub aktywowania struktur głębokich liczebników — gdyby przyjąć propozycję recenzenta — byłby w stanie rozstrzygnąć, czy oba wymienione liczebniki są kodenotacyjne. Tymczasem wiadomo, że aby stwierdzić, iż dwieście dziesiątek to tyle samo co dwadzieścia setek, trzeba biegle mnożyć.

1.2. Przykłady i typy struktur głębokich liczebników

Struktury głębokie zarówno liczebników cyfrowych, jak i liczebników słownych są wytwarzane według tego samego mechanizmu. Na przykład, liczebnikowi *sto tuzinów i dwadzieścia sześć* można przypisać strukturę głęboką o postaci P[$\mathbf{O}_{12}\mathbf{O}_{100}$(**1**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_{1}$(**6**)]. Z kolei liczebnik *jeden i jeden* ma postać P[$\mathbf{O}_{1}$(**1**), $\mathbf{O}_{1}$(**1**)]. Pierwszy z liczebników jest równoważny denotacyjnie zapisowi cyframi arabskimi „1226", któremu można przypisać wiele struktur głębokich w zależności od tego, na jaki liczebnik słowny jest on przekładany podczas czytania. Jeśli liczebnik cyfrowy „1226" jest odczytywany jako *tysiąc dwieście dwadzieścia sześć*, to jego struktura głęboka ma postać P[$\mathbf{O}_{1}$(**6**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_{100}$(**2**), $\mathbf{O}_{1000}$(**1**)]. Jeśli zaś „1226" odczytuje się jako *dwanaście setek i dwadzieścia sześć*, to jego struktura głęboka przyjmuje formę P[$\mathbf{O}_{100}\mathbf{O}_{12}$(**1**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_{1}$(**6**)]. Kolejność występowania argumentów funktora P w strukturach głębokich liczebników nie ma znaczenia w aspekcie denotacyjnym. Istnieją dwa wyróżnione sposoby rozmieszczania tych argumentów — sposób zgodny ze zwrotem obliczeniowym oraz sposób zgodny ze zwrotem wypowiedzeniowym liczebnika. Analiza ta pokazuje, że liczebniki cyfrowe są paradoksalnie wyrażeniami bardziej wieloznacznymi niż ich słowne odpowiedniki, ponieważ tym pierwszym można przypisać wiele struktur głębokich, podczas gdy te drugie są zazwyczaj skorelowane tylko z jedną strukturą głęboką. Umysł wykonuje jednak zalgorytmizowane obliczenia na liczebnikach cyfrowych, a nie słownych, mimo że te drugie są na ogół jednoznaczne, a te pierwsze wieloznaczne (w sensie posiadania wielu struktur głębokich). Wskazana wieloznaczność liczebników cyfrowych ma jednak charakter wyłącznie pozakontekstowy: w danym kontekście użycia są one rozumiane jednoznacznie, ponieważ użytkownik liczebników aktywuje w umyśle dokładnie jedną reprezentację — o określonej strukturze głębokiej — danego liczebnika cyfrowego używanego w akcie obliczeniowym.

Niektóre ze struktur głębokich nie mogą być przekształcane w liczebniki cyfrowe — z uwagi na sposób ich zapisu. Są one wtedy transformowane w niestandardowe liczebniki słowne. Taką własność ma na przykład struktura głęboka liczebnika *jeden i jeden*. Ze strukturą P[$\mathbf{O}_{1}$(**1**), $\mathbf{O}_{1}$(**1**)] nie jest skorelowany transformacyjnie żaden liczebnik cyfrowy w obowiązujących obecnie w Europie systemach zapisu liczebników cyfrowych. Tę samą własność ma liczebnik *tysiąc i jeszcze tysiąc oraz dwadzieścia sześć*, którego struktura głęboka ma postać P[$\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_{1}$(**6**)][7]. Jednym z powodów niemożliwości przekształcenia struktury głębokiej liczebnika słownego w liczebnik cyfrowy w danym systemie zapisu liczebników jest wielokrotne występowanie tego samego funktora pozycji składniowej w danej struk-

[7] Nietrudno wskazać przykłady użycia wymienionych liczebników w praktyce komunikacyjnej. Gdy nauczycielka w przedszkolu zadaje dzieciom pytanie: *Ile to jest — jeden i jeden?*, to nie traktuje liczebnika *jeden i jeden* jako synonimu liczebnika *dwa* (traktuje oba liczebniki jako jedynie kodenotacyjne). Liczebniki typu *tysiąc i jeszcze tysiąc oraz dwadzieścia sześć* bywają używane w sytuacji wydawania reszty w sklepie.

turze głębokiej. W pierwszej ze struktur funktor $\mathbf{O}_1$ występuje dwukrotnie. To samo dotyczy drugiej struktury i funktora $\mathbf{O}_{1000}$.

Oczywiście, gdyby system zapisywania liczebników cyfrowych dopuszczał kolumnowy sposób notacji, struktura głęboka P[$\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_1$(**6**)] mogłaby zostać przekształcona w $\begin{matrix}1026\\1000\end{matrix}_8$. W linearnych systemach zapisu liczebników cyfrowych (w dowolnych układach) nie da się przekształcić struktur z powtórzeniami funktorów pozycji składniowej w liczebniki cyfrowe. Inną przyczyną braku transformowalności struktury głębokiej na linearny liczebnik cyfrowy w danym systemie zapisu liczebników może być występowanie w danej strukturze takich funktorów pozycji składniowej, które w danym systemie zapisu i jego semantyki nie wytwarzają określonych osi obliczeniowych. Na przykład, ze struktury głębokiej o postaci P[$\mathbf{O}_5\mathbf{O}_{20}$(**7**), $\mathbf{O}_{10}$(**3**), $\mathbf{O}_1$(**6**)] umysł nie jest w stanie wytworzyć odpowiadającego jej liczebnika cyfrowego w arabskim systemie zapisu, mimo że w strukturze tej żaden funktor pozycji składniowej się nie powtarza. W układzie dziesiętnym zapisu cyfr arabskich funktory pozycji piątek oraz dwudziestek nie spełniają żadnych funkcji obliczeniowych.

Warto jednak zauważyć, że struktura P[$\mathbf{O}_5\mathbf{O}_{20}$(**7**), $\mathbf{O}_{10}$(**3**), $\mathbf{O}_1$(**6**)] jest przekształcalna w liczebnik słowny o postaci *siedem dwudziestek po pięć i trzydzieści sześć*, używany na przykład podczas rozmowy kasjerów w banku w kontekście: *podaj mi siedem dwudziestek po pięć złotych i trzydzieści sześć złotych*, gdzie *dwudziestki* są rozumiane jako paczki dwudziestomonetowe. Wydaje się więc, że każdą strukturę głęboką można przekształcić w jakiś (czasem nieużyteczny i niestandardowy) liczebnik słowny, podczas gdy niektórych struktur głębokich w danym systemie zapisu liczebników nie sposób przekształcić w liczebniki cyfrowe.

Obok typów struktur głębokich wyróżnionych z uwagi na transformowalność, można wyodrębnić ich typy ze względu na różne własności strukturalne.

Podstawową cechą struktur głębokich liczebnika, która może służyć do ich porządkowania, jest długość pozycyjna. Struktura o postaci P[$\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_1$(**6**)] jest pozycyjnie dłuższa od struktury P[$\mathbf{O}_{1000}$(**2**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_1$(**6**)], mimo że obie struktury są kodenotacyjne. Ta sama liczba lub liczność w zależności od systemu zapisu liczebników może być denotowana przez krótsze lub dłuższe pozycyjnie struktury głębokie. W arabskim systemie zapisu liczebników najkrótszymi pozycyjnie strukturami głębokimi są struktury stanowiące podstawienie schematu struktur o postaci P[$O_i(c_j)$]. Wyjaśniałoby to krótszy czas wykonywania obliczeń w systemie arabskim

[8] Kolumnowym sposobem zapisu niektórych liczebników cyfrowych jest klinowy system sumeryjski (zob. Ifrah 2006: 257-258). Ponadto w algorytmach dodawania, odejmowania czy nawet mnożenia jest stosowany kolumnowy sposób zapisu liczebników cyfrowych. W wypadku dodawania kolumny mogłyby mieć jedną strukturę głęboką z powtórzeniami funktorów pozycji składniowej. Liczebniki cyfrowe zapisywane kolumnowo w algorytmie pisemnego dodawania powstają więc w wyniku fuzji (dodawania) dwóch liczebników. Algorytm pisemnego dodawania dopuszcza *n*-piętrowe kolumny, które również można traktować jako złożone liczebniki cyfrowe (będące rezultatem *n*-członowych fuzji liczebników w zapisie linearnym).

(np. dodawania) na tak zwanych „okrągłych liczbach" (np. „2 + 3", „10 + 30", „200 + 300", itd.) w stosunku do czasu wykonywania obliczeń na pozostałych liczbach (np. „27 + 14", „234 + 67"). W pierwszym wypadku umysł przetwarza struktury głębokie o najmniejszej długości pozycyjnej, podczas gdy w drugim struktury głębokie są w sposób istotny dłuższe pozycyjnie. Warto dodać, że długość pozycyjna struktury głębokiej nie musi odwzorowywać długości leksykalnej odpowiadającego jej liczebnika słownego. Na przykład liczebnik *sześćdziesiąt* jest liczebnikiem leksykalnie złożonym o postaci *sześć-dziesiąt*, a więc leksykalnie dłuższym od liczebników *sześć* oraz *dziesięć*. Natomiast z punktu widzenia struktur głębokich wszystkie trzy wymienione liczebniki mają struktury o tej samej długości pozycyjnej.

Inną własnością syntaktyczną struktur głębokich jest ich stopień głębokości. Pozycje składniowe w strukturach głębokich liczebników mogą być wytwarzane przez permutację rozmaitych funktorów pozycji składniowych. Permutacje te mogą mieć różną długość, która wyznacza właśnie stopnień głębokości danej struktury. Struktury głębokie, których wszystkie pozycje składniowe są utworzone przez jednokrotne zastosowanie dowolnego funktora pozycji składniowej, są najpłytszymi strukturami głębokimi. Na przykład, stopień głębokości struktury P[$\mathbf{O}_5\mathbf{O}_{20}$(**7**), $\mathbf{O}_{10}$(**3**), $\mathbf{O}_1$(**6**)] jest wyższy od stopnia głębokości struktury P[$\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_1$(**6**)]. W pierwszej strukturze pierwsza pozycja jest utworzona przez zastosowanie dwóch funktorów pozycji syntaktycznej, natomiast w strukturze drugiej wszystkie pozycje powstają przez jednokrotne stosowanie funktorów. Wydaje się, że wraz ze wzrostem stopnia głębokości struktur głębokich wzrasta również stopień trudności ich przetwarzania w procesach obliczeniowych, mierzony średnim czasem, którego umysł potrzebuje do wykonania operacji na danej strukturze.

Każdej strukturze głębokiej liczebnika można przypisać metrykę pozycyjną. Jest nią ciąg permutacji funktorów pozycji składniowych odpowiedzialnych za wytworzenie w danej strukturze kolejnych pozycji składniowych. Metryką pozycyjną struktury o postaci P[$\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_1$(**6**)] jest ciąg ⟨$\mathbf{O}_{1000}$, $\mathbf{O}_{1000}$, $\mathbf{O}_{10}$, $\mathbf{O}_1$⟩, a struktura P[$\mathbf{O}_5\mathbf{O}_{20}$(**7**), $\mathbf{O}_{10}$(**3**), $\mathbf{O}_1$(**6**)] ma metrykę ⟨$\mathbf{O}_5\mathbf{O}_{20}$, $\mathbf{O}_{10}$, $\mathbf{O}_1$⟩. Porównywanie struktur głębokich liczebników o tej samej metryce pozycyjnej pod względem tego, która z nich denotuje większą liczbę, nie wymaga przetwarzania porównywanych struktur na równoważne zakresowo struktury o odpowiednim standardowym kształcie. Porównanie struktury P[$\mathbf{O}_{100}$(**1**), $\mathbf{O}_{10}$(**1**), $\mathbf{O}_1$(**7**)] ze strukturą P[$\mathbf{O}_{100}$(**1**), $\mathbf{O}_{10}$(**2**), $\mathbf{O}_1$(**2**)] pod kątem tego, która z nich denotuje większą liczbę naturalną, stanowi o wiele łatwiejsze zadanie niż porównanie w tym samym aspekcie struktur P[$\mathbf{O}_{100}$(**1**), $\mathbf{O}_{10}$(**1**), $\mathbf{O}_1$(**7**)] i P[$\mathbf{O}_{12}$(**6**), $\mathbf{O}_{10}$(**3**), $\mathbf{O}_1$(**7**)]. Struktury drugiej pary mają różne metryki pozycyjne. W związku z tym średni czas rozwiązania zadania *Która z liczb jest większa: sto siedemnaście czy sto dwadzieścia dwa?* jest krótszy niż średni czas rozwiązywania zadania *Która z liczb jest większa: sto siedemnaście czy sześć tuzinów trzydzieści siedem?* Rozwiązując drugie zadanie, umysł musi sprowadzić obie struktury — za pomocą rozmaitych operacji arytmetycznych — do struktur o tej samej metryce.

1.3. Przetwarzanie struktur głębokich liczebników — przykłady

W procesach komunikacji międzyludzkiej użytkownicy języka, posługując się liczebnikami, dokonują różnego rodzaju obliczeń. Elementarnymi operacjami obliczeniowymi są takie, w których muszą rozstrzygnąć, czy dwa liczebniki denotują tę samą liczbę (liczność), czy też jeden z nich denotuje liczbę większą. Powodzenie takich operacji zależy od tego, czy umysł sprawnie przetwarza struktury głębokie liczebników używanych w aktach obliczeniowych.

Skąd umysł wie, że dwa różne liczebniki, *dwadzieścia* oraz *dziesięć i dziesięć*, denotują tę samą liczbę? Nie są to liczebniki synonimiczne. To, że mają różne znaczenie w sensie sposobu użycia, przejawia się w wielu kontekstach wypowiedzeniowych (choć nie zawsze). Na przykład, gdy zamawiamy w pubie piwo, wypowiedź *Proszę dwadzieścia butelek* ma inne znaczenie od wypowiedzi *Proszę dziesięć i dziesięć butelek*. Druga wypowiedź może sprawić, że kelner zada pytanie *Czy podać najpierw dziesięć i potem dziesięć?* Wypowiedź pierwsza nie wywoła u niego takiej reakcji, co różnicuje sposób użycia obu liczebników. Fakt ten można wyjaśnić, twierdząc, że przytoczone wypowiedzi aktywują w umyśle odmienne struktury głębokie liczebników. W pierwszym wypadku taką strukturę opisuje wyrażenie o postaci P[$\mathbf{O}_{10}$(**2**)], a w drugim — P[$\mathbf{O}_{10}$(**1**), $\mathbf{O}_{10}$(**1**)]. Kelner wie jednak, że opłata za liczbę butelek piwa wynosi tyle samo przy obu zamówieniach. Można to wyjaśnić tym, że jego umysł przetwarza strukturę P[$\mathbf{O}_{10}$(**1**), $\mathbf{O}_{10}$(**1**)] na strukturę P[$\mathbf{O}_{10}$(**2**)].

Skąd wiadomo, że liczebniki *dwa tuziny setek*, *dwieście tuzinów*, *dwa tysiące czterysta*, *tysiąc dwieście i tysiąc dwieście* denotują jedną i tę samą liczbę (liczność)? Wymienionym liczebnikom odpowiadają następujące struktury głębokie: P[$\mathbf{O}_{100}\mathbf{O}_{12}$(**2**)], P[$\mathbf{O}_{12}\mathbf{O}_{100}$(**2**)], P[$\mathbf{O}_{100}$(**4**), $\mathbf{O}_{1000}$(**2**)], P[$\mathbf{O}_{100}$(**2**), $\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{100}$(**2**), $\mathbf{O}_{1000}$(**1**)]. Aby wykazać, że struktury te denotują tę samą liczbę, musimy redukcyjnie przetworzyć je na pewną strukturę kanoniczną. W tym wypadku może to być struktura głęboka liczebnika cyfrowego „2400” (odczytanego jako *dwa tysiące czterysta*), czyli P[$\mathbf{O}_{100}$(**4**), $\mathbf{O}_{1000}$(**2**)].

Strategia umysłu w przetwarzaniu struktur głębokich liczebników podczas czynności ich porównywania w wielu wypadkach może podpadać pod następujący model:

> Każdą z dwóch porównywanych struktur głębokich umysł redukuje do struktury o identycznej metryce pozycyjnej. Jeśli w wyniku takiego procesu otrzyma dwie identyczne struktury redukcyjne, to stwierdza, że struktury na wejściu denotują tę samą liczbę (liczność). Jeśli zaś uzyska dwie różne struktury głębokie o tej samej metryce pozycyjnej, to następnie bada, która z obu struktur denotuje większą liczbę (liczność).

Badanie to stanowi wyłącznie analizę syntaktyczną obu redukcyjnych struktur. I tak, dla przykładu, umysł może przetworzyć strukturę P[$\mathbf{O}_{100}\mathbf{O}_{12}$(**2**)] redukcyjnie na strukturę P[$\mathbf{O}_{1200}$(**2**)]; następnie może przetworzyć strukturę P[$\mathbf{O}_{12}\mathbf{O}_{100}$(**2**)] na struk-

turę P[$\mathbf{O}_{1200}$(**2**)]. Stwierdza, że obie struktury są identyczne, i na tej podstawie wyprowadza wniosek, że liczebniki im odpowiadające denotują tę samą liczbę. Podobnie, umysł przetwarza strukturę P[$\mathbf{O}_{100}$(**2**), $\mathbf{O}_{1000}$(**1**), $\mathbf{O}_{100}$(**2**), $\mathbf{O}_{1000}$(**1**)] na strukturę P[$\mathbf{O}_{1200}$(**2**)]. W wyniku tej operacji wnioskuje, że liczebnik *tysiąc dwieście i tysiąc dwieście* denotuje tę samą liczbę, którą denotują liczebniki *dwa tuziny setek*, *dwieście tuzinów*. Kolejnym zadaniem jest redukcja struktury głębokiej P[$\mathbf{O}_{1200}$(**2**)] do struktury kanonicznej P[$\mathbf{O}_{100}$(**4**), $\mathbf{O}_{1000}$(**2**)]. Sukces polega na wydaniu werdyktu, zgodnie z którym liczebnik *dwa tysiące czterysta* denotuje tę samą liczbę co wcześniej analizowane liczebniki.

Podobne przykłady można mnożyć w nieskończoność. Skoro więc umysł przetwarza struktury głębokie liczebników, to musi wykonywać te operacje według pewnych reguł inferencji. System takich reguł można określić mianem logiki liczebników.

2. LOGIKA LICZEBNIKÓW

Teoria logiki liczebników (LL), rozumianej jako kompleksowy mechanizm formatowania i przetwarzania reprezentacji liczebnikowych, zasadza się na logice klasycznej i arytmetyce Peana. LL zawiera trzy moduły. Na pierwszy składa się mechanizm tworzenia funktorów pozycji składniowych, opisywany przez arytmetykę tych funktorów, na drugi — mechanizm tworzenia cyfr elementarnych dowolnego systemu zapisu liczebników, a na trzeci — system reguł wyznaczających poprawne równoważnościowe inferencje określone na liczebnikowych strukturach głębokich.

Skoro zbiór wszystkich funktorów pozycji składniowych jest równoliczny ze zbiorem liczb naturalnych, a na funktorach pozycji można określić takie operacje, których odpowiednikami są operacje następnika, dodawania i mnożenia w zbiorze liczb naturalnych, to algebra funktorów pozycji syntaktycznych jest izomorficzna ze standardową algebrą liczb naturalnych z mnożeniem i dodawaniem. Z kolei moduł cyfrowy nie jest algebraizowalny w arytmetyce Peana, ponieważ w każdym systemie zapisu liczebników przyjmuje się skończoną liczbę cyfr elementarnych. Dopiero nad tymi dwoma modułami nadbudowana jest właściwa logika liczebników (w sensie węższym). W tym świetle arytmetyka Peana stanowi jedynie narzędzie opisu mechanizmów formatowania struktur głębokich liczebników. Umysł, syntetyzując lub aktywując struktury głębokie liczebników, musi stosować liczby naturalne, aby wyznaczać sposoby działania funktorów pozycji składniowych.

Język logiki liczebników powstaje w wyniku rozszerzenia języka, w którym struktury głębokie liczebników są formatowane. Do rozszerzanego słownika zostają wprowadzone następujące wyrażenia:

(i) funktor zdaniotwórczy „$\vdash$”, którego argumentami są struktury głębokie liczebników, wyrażający relację kodenotacyjnej redukcji jednej struktury do drugiej,

(ii) wyrażenie „/” oznaczające rezultat zastąpienia wyrażenia po lewej stronie listą wyrażeń po prawej stronie w kontekście struktury głębokiej liczebnika,

(iii) wyrażenie stałe „∅” oznaczające dowolne puste wyrażenie w strukturze głębokiej liczebnika (wyrażenie to może występować zarówno po prawej, jak i po lewej stronie wyrażenia „/”).

Zatem kategoria **L** w języku logiki liczebników jest rozszerzona o wyrażenia typu: P[…,∅/α, …], P[…, α/∅, …], P[…, α/(β_1, …, β_k), …], gdzie α, β_1, …, $\beta_k \in$ **K**, a ∅ jest wyrażeniem pustym. Oto definicja poprawnie zbudowanej formuły języka LL:

(Df. For) $\alpha \in \mathbf{L} \wedge \beta \in \mathbf{L} \equiv_{df} \alpha \vdash \beta \in \mathbf{For}$

LL jest więc logiką jednej relacji, mianowicie — relacji zachodzącej między strukturami głębokimi liczebników wtedy, gdy jedna z nich redukuje się kodenotacyjnie do drugiej. Pozostałe relacje i operacje, które umysł konstytuuje i wykonuje na strukturach głębokich liczebników, są definiowalne za pomocą relacji redukcji. Zdanie α⊢ β może być odczytywane jako: liczebnikowa struktura głęboka α redukuje się kodenotacyjnie do liczebnikowej struktury głębokiej β. Na przykład, zdanie o postaci „P[$\mathbf{O}_{100}\mathbf{O}_{12}$(**2**)] ⊢ P[$\mathbf{O}_{1200}$(**2**)]” wyraża informację, że struktura głęboka P[$\mathbf{O}_{100}\mathbf{O}_{12}$(**2**)] sprowadza się kodenotacyjnie do struktury głębokiej P[$\mathbf{O}_{1200}$(**2**)]. Jeśli podane zdanie jest prawdziwe, to zgodnie z regułami logiki liczebników struktura P[$\mathbf{O}_{100}\mathbf{O}_{12}$(**2**)] jest przekształcalna w strukturę P[$\mathbf{O}_{1200}$(**2**)].

2.1. Moduł arytmetyczny logiki liczebników

Oto aksjomaty opisujące arytmetyczny mechanizm tworzenia funktorów pozycji składniowych (gdzie *Seq* jest funkcją następnika, której argumentami i wartościami są wyłącznie funktory pozycji):

(AO1) $(\forall O_i)\ \mathbf{O}_0 \neq Seq(O_i)$

(AO2) $(\forall O_i, O_j)\ [Seq(O_i) = Seq(O_j) \rightarrow O_i = O_j]$

(AO3) $\Psi(\mathbf{O}_0) \wedge (\forall O_i)\ [\Psi(O_i) \rightarrow \Psi(Seq(O_i))] \rightarrow (\forall O_i)\ \Psi(O_i)$

Zbiór wszystkich funktorów pozycji składniowych jest liniowo uporządkowany przez operację następnika, którego pierwszym elementem w tym porządku jest funktor $\mathbf{O}_0$. Ponadto w zbiorze tym obowiązuje zasada indukcji arytmetycznej. Ponieważ indeksy funktorów pozycji składniowych zachowują się analogicznie do liczb naturalnych, to na nich również określona jest funkcja następnika, **Seq*, którą można definicyjnie sprowadzić do funkcji *Seq*:

(AO4) $(\forall O_i, i)\ O_{*Seq(i)} = Seq(O_i)$

Aksjomat (AO4) jest bardzo ważny z technicznego punktu widzenia. Narzuca porządek sekwencjalny na sposób indeksowania funktorów pozycji składniowych. Znając indeks funktora, można określić jego pozycję sekwencjalną w liniowo uporządkowanym uniwersum wszystkich funktorów pozycji składniowych.

Na funktorach pozycji składniowych, podobnie jak na liczbach naturalnych, można określić pewne operacje. Służą one do syntezy nowych funktorów pozycji składniowych z funktorów, które umysł w swoim funkcjonowaniu już zinternalizował. Niech „+", „×" oznaczają odpowiednio operacje dodawania funktorów oraz mnożenia funktorów pozycji składniowych. Następne aksjomaty stwierdzają, że wartościami tych operacji są funktory pozycji, których indeksy dolne są numerami w odpowiedni sposób uzyskanymi w wyniku zastosowania operacji dodawania lub mnożenia numerów. Innymi słowy, istnieją operacyjne mechanizmy tworzenia funktorów pozycji składniowych, przy czym operacje dodawania i mnożenia numerów tych funktorów są zdefiniowane indukcyjnie, analogicznie do odpowiadających im operacji arytmetyki liczb naturalnych:

(Df. +) (a) $(\forall j)\ O_{j+0} = O_j$; (b) $(\forall i, j)\ O_{*Seq(i)\ +\ j} = Seq(O_{i+j})$

(Df. ×) (a) $(\forall j)\ O_{j\times 1} = O_j$; (b) $(\forall i, j)\ O_{j\times *Seq(i)} = O_{(j\times i)+i}$

(AO5) $(\forall O_i, O_j) +(O_i, O_j) = O_{i+j}$

(AO6) $(\forall O_i, O_j) \times(O_i, O_j) = O_{i\times j}$

Zgodnie z (AO5) wynikiem dodawania dwóch funktorów o indeksach i oraz j jest funktor o indeksie $i+j$. Podobnie, zgodnie z (AO6) wynikiem pomnożenia dwóch funktorów o indeksach i oraz j jest funktor o indeksie $i\times j$. Indeksy są wyliczane zgodnie z definicjami przyjętymi w arytmetyce Peana.

Inną bardzo ważną operacją tworzenia funktorów pozycji składniowych jest operacja ich potęgowania, która służy do tworzenia kanonicznych funktorów pozycji składniowych.

(Df. ^{n}) (a) $(\forall j)\ O_j^{\,0} = \mathbf{O}_1$; (b) $(\forall j)\ O_j^{\,*Seq(i)} = \times(O_j, O_j^{\,i})$

Relację tworzenia funktorów kanonicznych przez funktor O_i można zdefiniować jako:

(Df. *Kan*) $O_i\ Kan\ O_j \equiv [O_i \neq \mathbf{O}_0 \wedge (\exists n)\ O_j = O_i^{\,n}]$

Funktor O_i wyznacza O_j jako funktor kanoniczny wtedy, gdy funktor O_j jest jakąś potęgą funktora O_i niebędącego funktorem pozycji zer. Każdy funktor o indeksie większym od 0 wyznacza więc dokładnie jedną klasę funktorów kanonicznych. Dowolna klasa funktorów kanonicznych stanowi zestaw funktorów wystarczających do wygenerowania zbioru struktur głębokich liczebników pokrywających z uwagi na funkcję denotacji w sposób jedno-jednoznaczny zbiór wszystkich liczb naturalnych[9].

[9] Wynik ten zostanie udowodniony w trzeciej części całego studium, poświęconej semantyce logiki liczebników.

Można udowodnić twierdzenie, zgodnie z którym jeśli funktor jedności $\mathbf{O}_1$ jest generatorem funktorów kanonicznych, to klasa funktorów kanonicznych generowanych przez $\mathbf{O}_1$ jest jednoelementowa.

(T1) $(\forall O_i)\,[\mathbf{O}_1\ Kan\ O_i \rightarrow O_i = \mathbf{O}_1]$

System zapisu liczebników za pomocą karbów, w którym operuje się jedynie funktorem pozycji jedności $\mathbf{O}_1$, jest więc wystandaryzowanym systemem zapisu liczebników cyfrowych. W takim systemie dowolna sekwencja karbów (kresek lub jedynek) stanowi standardowy liczebnik cyfrowy.

(T2) głosi, że funktor pozycji zerowej $\mathbf{O}_0$ nie należy do żadnej klasy kanonicznych funktorów pozycji:

(T2) $(\forall O_i)\,[\sim O_i\ Kan\ \mathbf{O}_0]$

Znaczy to, że funktor $\mathbf{O}_0$ jest zbędnym funktorem w zapisie liczebników cyfrowych w dowolnym standardowym systemie ich zapisu.

Zgodnie z twierdzeniem (T3) funktor pozycji jedności należy do każdej klasy kanonicznych funktorów pozycji.

(T3) $(\forall O_i)\,[O_i\ Kan\ \mathbf{O}_1]$

Mówiąc metaforycznie, bez osi jedności nie da się przeprowadzać zalgorytmizowanych operacji obliczeniowych.

Łatwo wykazać, że funktory pozycji składniowych zachowują się tak jak liczby naturalne Peana. Wyrażeniom „$\mathbf{O}_0$”, „O_i”, „O_{i+j}”, „$O_{i\times j}$”, „*Seq*”, „+”, „×” można przyporządkować wyrażenia PA o postaci „0”, „i”, „$i+j$”, „$i\times j$”, „S”, „+”, „×” oznaczające kolejno: liczbę *0*, dowolną zmienną liczbową, wynik dodania do liczby i liczby j, wynik pomnożenia liczby i przez liczbę j, operację następnika, operację dodawania oraz operację mnożenia. Wszystkie aksjomaty modułu arytmetycznego LL po przekładzie przekształcają się w aksjomaty PA (lub tautologie logiczne w wypadku (AO5) i (AO6)). Można więc przyjąć, że indeksami funktorów pozycji są liczby naturalne Peana, które wyznaczają sposoby działania tych funktorów. W związku z tym na zbiorze funktorów pozycji składniowych można określić relację większości:

(Df. (>)) $O_i\ (>)\ O_j \equiv i > j$

Relacja (>) nie jest tożsama z relacją >, ponieważ pierwsza jest określona na funktorach, a druga na ich indeksach. Umysł, ucząc się stosowania funktorów pozycji składniowych, koduje informacje na temat uporządkowania tych funktorów ze względu na zachodzącą między nimi relację większości. Na przykład, wiedzę, że *miliony* są większe od *setek tysięcy*, a te od *tysięcy*, zawdzięczamy temu, że uniwersum pozycji składniowych jest w umyśle kodowane właśnie wraz z porządkiem wyznaczonym przez relację (>). Jeśli użytkownik języka nie wie, czy *oktyliony* są większe od *sekstylionów*, znaczy to, że nie utworzył dotychczas wymienionych funktorów i w związku z tym nie może przypisać odpowiednim leksemom powiązanych z nimi struktur głę-

bokich, a to w konsekwencji prowadzi do niemożności ustalenia relacji większości między analizowanymi funktorami.

Następujące twierdzenie ustala związek między relacją (>) a funkcją *Seq*:

(T4) $(\forall O_i)\ Seq(O_i)\ (>)\ O_i$

Oczywiste są również zależności między funktorami pozycji utworzonymi za pomocą działań dodawania oraz mnożenia funktorów a funktorami stanowiącymi argumenty tych działań.

(T5) $(\forall O_i, O_j)\ [O_j \neq \mathbf{O}_0 \rightarrow +(O_i, O_j)\ (>)\ O_i]$

(T6) $(\forall O_i, O_j)\ [O_i \neq \mathbf{O}_0 \rightarrow +(O_i, O_j)\ (>)\ O_j]$

(T7) $(\forall O_i, O_j)\ [O_j\ (>)\ \mathbf{O}_1 \wedge O_i\ (>)\ \mathbf{O}_0 \rightarrow \times(O_i, O_j)\ (>)\ O_i]$

(T8) $(\forall O_i, O_j)\ [O_i\ (>)\ \mathbf{O}_1 \wedge O_j\ (>)\ \mathbf{O}_0 \rightarrow \times(O_i, O_j)\ (>)\ O_j]$

(T9) $(\forall O_i, O_j, n)\ [n > 1 \rightarrow (O_i\ (>)\ O_j \equiv O_i^n\ (>)\ O_j^n)]$

Twierdzenia (T5)-(T9) łącznie głoszą, że funktory pozycji składniowych utworzone za pomocą działań dodawania, mnożenia i potęgowania są zasadniczo większe niż funktory będące argumentami tych działań (poprzedniki tych twierdzeń określają wyjątki od reguły).

Z teoriomodelowego punktu widzenia funktory pozycji składniowych zachowują się jak funkcje działające na pewien skończony zbiór obiektów (cyfr elementarnych), których wartościami są pary zbudowane z danej funkcji-funktora oraz cyfry elementarnej. W najprostszym wystandaryzowanym systemie zapisu liczebników za pomocą karbów funktor pozycji składniowej jedności działa wyłącznie na cyfrę **1**. Ponieważ w innych systemach jest więcej cyfr elementarnych, można powiedzieć, że w zależności od systemu zapisu liczebników funktory pozycji mają różne długości rozumiane jako liczba kardynalna ich dziedzin. W danym systemie długość każdego z nich jest jednak identyczna. W arytmetyce Peana nie można natomiast mówić o czymś takim jak długość liczb naturalnych, która byłaby identyczna dla każdej z nich. Dlatego też, mimo że funktory pozycji zachowują się jak liczby naturalne, przysługują im własności, których liczby naturalne nie mają, choćby z tej racji, że funktory są funkcjami, a standardowo rozumiane liczby naturalne jako obiekty jednostkowe funkcjami nie są. Wydaje się więc, że funktorów pozycji składniowych nie można utożsamić z liczbami naturalnymi.

2.2. Moduł cyfrowy logiki liczebników

Systemy zapisu liczebników mogą się różnić pod względem liczby cyfr elementarnych. Z teoretycznego punktu widzenia nie jest nawet wykluczony system zapisu liczebników operujący na milionie cyfr elementarnych. Zawsze jednak w takim sys-

temie musi występować cyfra maksymalna $\mathbf{c_{max}}$. Najmniejsza liczba cyfr elementarnych wymagana do zapisu liczebników cyfrowych wynosi jeden (na przykład w systemie zapisu liczebników za pomocą karbów). Tą obligatoryjną cyfrą elementarną jest **1**. W systemie takim zachodzi: $\mathbf{1} = \mathbf{c_{max}}$. Niezależnie od tego, ile cyfr elementarnych występuje w danym systemie zapisu liczebników, zbiór ten jest uporządkowany stosunkiem bycia większą cyfrą, symbolizowanym przez znak „[>]". Ponadto, w większości systemów zapisu liczebników przyjmuje się dodatkowy znak cyfrowy dla oznaczenia „braku cyfry" na danej pozycji składniowej; w systemie arabskim tym znakiem jest **0**. Nie jest on jednak obligatoryjną cyfrą elementarną w strukturach głębokich liczebników: każda liczba denotowana przez strukturę głęboką, do której sformatowania użyta jest cyfra **0**, może być również denotowana przez strukturę głęboką, do której sformatowania nie stosuje się cyfry **0**. Cyfra **0** jest wymagana w systemach, w których możliwa jest standaryzacja liczebników cyfrowych, czyli zabieg polegający na tym, że w zbiorze wszystkich struktur głębokich wyróżnia się taki podzbiór, którego każdy element denotuje różną liczbę naturalną i którego obraz funkcji denotacji jest zbiorem wszystkich liczb naturalnych.

Moduł cyfrowy LL funkcjonuje zgodnie z mechanizmem logicznym wyznaczanym przez układ aksjomatów:

(AC1) $(\forall c_i, c_j, c_k)\,[c_i\ [>]\ c_j \wedge c_j\ [>]\ c_k \rightarrow c_i\ [>]\ c_k\ ]$

(AC2) $(\forall c_i) \sim c_i\ [>]\ c_i$

(AC3) $(\forall c_i, c_j)\,[c_i\ [>]\ c_j \rightarrow \sim c_j\ [>]\ c_i\ ]$

Relacja bycia większą cyfrą jest relacją przechodnią, przeciwzwrotną i antysymetryczną. Ponadto w wypadku każdego systemu zapisu liczebników, w którym możliwa jest standaryzacja liczebników cyfrowych, w zbiorze cyfr elementarnych istnieje element największy ze względu na relację [>], którym jest cyfra $\mathbf{c_{max}}$, oraz element najmniejszy, którym jest cyfra **0**.

(AC4) $(\forall c_i)\,[c_i \neq \mathbf{c_{max}} \rightarrow \mathbf{c_{max}}\ [>]\ c_i]$

(AC5) $(\forall c_i)\,[c_i \neq \mathbf{0} \rightarrow c_i\ [>]\ \mathbf{0}]$

Na cyfrach elementarnych określona jest operacja następnika *Sq*:

(AC6) $(\forall c_i)\ \mathbf{0} \neq Sq(c_i)$

(AC7) $\mathbf{1} = Sq(\mathbf{0})$

(AC8) $\mathbf{c_{max}} = Sq(\mathbf{c_{max}})$

(AC9) $(\forall c_i, c_j)\,[c_i \neq \mathbf{c_{max}} \wedge c_j \neq \mathbf{c_{max}} \rightarrow (Sq(c_i) = Sq(c_j) \rightarrow c_i = c_j)]$

(AC10) $\Psi(\mathbf{0}) \wedge (\forall c_i)\,[\Psi(c_i) \rightarrow \Psi(Sq(c_i))] \rightarrow (\forall c_i)\ \Psi(c_i)$

Aksjomat (AC6) głosi, że cyfra elementarna **0** nie jest następnikiem żadnej innej cyfry elementarnej. Ponieważ cyfra **1** jest obowiązkowym składnikiem w każdym systemie zapisu liczebników, (AC7) stanowi jej definicję. Zgodnie z (AC8) następnikiem cyfry maksymalnej jest ona sama, (AC9) stwierdza zaś, że jeśli następniki dwóch cyfr elementarnych różnych od cyfry maksymalnej są identyczne, to cyfry te są również identyczne. (AC9) wyklucza więc wśród cyfr elementarnych duplikowanie się cyfr, czyli sytuację, w której w danym systemie zapisu liczebników występują dwie cyfry elementarne mające dokładnie to samo znaczenie arytmetyczne. (AC10) stanowi zasadę indukcji zastosowaną do skończonego zbioru cyfr elementarnych. Aksjomat ten jest fakultatywny.

Relacja większości jest skorelowana z funkcją następnika (określonego na cyfrach elementarnych) zgodnie z aksjomatem (AC11), głoszącym, że następnik dowolnej cyfry elementarnej, która nie jest cyfrą maksymalną, jest większy od danej cyfry elementarnej:

(AC11) $(\forall c_i)\,[c_i \neq \mathbf{c_{max}} \rightarrow Sq(c_i)\ [>]\ c_i]$

(AC11) opisuje mechanizm uczenia się cyfr elementarnych. Na przykład, dziecko recytujące kolejne cyfry elementarne od **0** do $\mathbf{c_{max}}$ internalizuje w umyśle relację większości.

Ponieważ funktory pozycji składniowych oraz cyfry elementarne stanowią narzędzia tworzenia wyrażeń należących do kategorii **K – N** (z których z kolei funktor planu tworzy struktury głębokie liczebników), należy przyjąć, że oba mechanizmy — tworzenia funktorów pozycji oraz tworzenia cyfr elementarnych — współdziałają w ramach mechanizmu tworzenia struktur należących do kategorii **K – N**. Ten mechanizm wyższego rzędu narzuca na procesy formatowania struktur należących do kategorii **K – N** taką ich własność formalną, że ich uniwersum jest uporządkowane relacją większości. Niech wyrażenie „/>/" oznacza tę relację. Następujące aksjomaty opisują jej własności:

(ACO1) $(\forall c_i, c_j, O_i)\,[O_i \neq \mathbf{O}_0 \rightarrow (O_i(c_k)\ />/\ O_i(c_j) \equiv c_k\ [>]\ c_j)]$

(ACO2) $(\forall O_i, O_j, c_k)\,[O_i \neq \mathbf{O}_0 \wedge c_k \neq \mathbf{0} \rightarrow (O_i(c_k)\ />/\ O_j(c_k) \equiv O_i\ (>)\ O_j)]$

(ACO3) $(\forall c_i, O_j)\,[c_i \neq \mathbf{0} \wedge O_j \neq \mathbf{O}_0 \rightarrow (\forall c_k)\ (O_j(c_i)\ />/\ \mathbf{O}_0(c_k))]$

(ACO4) $(\forall c_i, O_j)\,[c_i \neq \mathbf{0} \wedge O_j \neq \mathbf{O}_0 \rightarrow (\forall O_k)\ (O_j(c_i)\ />/\ O_k(\mathbf{0}))]$

Aksjomaty (ACO1)-(ACO4) można traktować jako definicję przez postulaty relacji />/. Zgodnie z (ACO1) wyrażenie powstające w wyniku „nasycenia" danego funktora różnego od $\mathbf{O}_0$ daną cyfrą jest większe od wyrażenia powstającego w wyniku nasycenia tego samego funktora inną cyfrą wtedy, gdy pierwsza cyfra jest większa od drugiej. Zgodnie z tym aksjomatem *trzy dziesiątki* są większe od *jednej dziesiątki*, ponieważ cyfra **3** jest większa od cyfry **1**. Zgodnie z (ACO2) wyrażenie powstające w wyniku nasycenia danego funktora różnego od $\mathbf{O}_0$ daną cyfrą (ale różną

od **0**) jest większe od wyrażenia powstającego w wyniku nasycenia drugiego funktora tą samą cyfrą wtedy, gdy pierwszy funktor jest większy od drugiego. Na przykład, struktura *trzy setki* jest większa od struktury *trzy dziesiątki*, ponieważ funktor setek jest większy od funktora dziesiątek. Zgodnie z aksjomatami (ACO3) i (ACO4) każda struktura należąca do kategorii **K – N**, która jest utworzona z cyfry elementarnej **0** lub funktora pozycji zerowej, jest mniejsza od dowolnej struktury należącej do kategorii **K – N**, która nie jest utworzona ani z cyfry elementarnej **0**, ani z funktora pozycji zerowej. Na przykład, zgodnie z (ACO3) struktura *trzy jedności* jest większa od struktury *dziewięć zer*. Analogicznie, zgodnie z (ACO4) struktura o postaci *dwie jedności* jest większa od struktury *zero milionów*.

Między cyframi elementarnymi a funktorami pozycji składniowych zachodzi jedno-jednoznaczna relacja odpowiedniości, którą można wyrazić funkcyjnie. Do tego celu potrzebna jest konstrukcja operacji potęgowania funkcji *Seq* i *Sq*.

(Df. Seq^n) (i) $Seq^0(O_i) = O_i$; (ii) $(\forall O_i)\ Seq^k(O_i) = Seq\ Seq^{k-1}(O_i)$

(Df. Sq^n) (i) $Sq^0(c_i) = c_i$; (ii) $(\forall c_i)\ Sq^k(c_i) = Sq\ Sq^{k-1}(c_i)$

Definicja funkcji odpowiedniości * między cyframi elementarnymi a funktorami pozycji składniowych przyjmuje postać:

(Df. *) $c_i* = O_j \equiv_{df} (\exists k)\ [Sq^k(\mathbf{0}) = c_i \wedge Seq^k(\mathbf{O}_0) = O_j]$

(Df. *) ustala jedno-jednoznaczne przyporządkowanie cyfr elementarnych danego systemu zapisu liczebników pierwszym, początkowym funktorom pozycji składniowych. Skoro, jak już wcześniej wykazałem, funktory pozycji składniowych zachowują się jak liczby naturalne, to na mocy (Df. *) cyfry elementarne tworzą skończony ciąg obiektów izomorficzny z określonym, początkowym odcinkiem liczb naturalnych (zaczynającym się od liczby zero, a kończącym się na pewnej liczbie skorelowanej z cyfrą $\mathbf{c_{max}}$). Istnienie funkcji odpowiedniości między cyframi elementarnymi a funktorami pozycji składniowych danego systemu wyklucza sytuację, w której na przykład następnik dowolnej cyfry elementarnej c_i jest skorelowany z funktorem pozycji O_j o własności $O_j = Seq^k(c_i*)$ i $k > 1$. Opisuje to następujące twierdzenie:

(T10) $(\forall c_i)\ [c_i \neq \mathbf{c_{max}} \rightarrow (Sq(c_i))* = Seq(c_i*)]$

W wypadku każdej cyfry niemaksymalnej obrazem jej następnika w uniwersum funktorów pozycji składniowych jest następnik funktora pozycji stanowiącego obraz danej cyfry w tym uniwersum. Zgodnie z (T10) nie można zmodyfikować dowolnego systemu zapisu liczebników w taki sposób, by wprowadzać do niego nową cyfrę elementarną „większą o dwie lub więcej jednostek" od dotychczasowej cyfry maksymalnej. Nie można na przykład wprowadzić do systemu dziesiętnego cyfry elementarnej denotującej tuzin. Każda taka modyfikacja musi respektować definicję

(Df. *). Znaczy to, że następna cyfra elementarna, którą można wprowadzić do systemu dziesiętnego, jest cyfrą odpowiadającą liczebnikowi słownemu *dziesięć*[10].

Na gruncie (Df. *) można skonstruować rozszerzenie operacji mnożenia funktorów pozycji składniowych:

(Df. ×*) $(\forall O_i, c) \times(O_i, c) = \times(O_i, c^*)$

Zgodnie z (Df. ×*) można na przykład wyprowadzić $\times(\mathbf{O}_{10}, \mathbf{5}) = \times(\mathbf{O}_{10}, \mathbf{5}^*)$. A skoro $\mathbf{5}^* = \mathbf{O}_5$, to $\times(\mathbf{O}_{10}, \mathbf{5}) = \times(\mathbf{O}_{10}, \mathbf{O}_5)$[11].

Z przedstawionych aksjomatów można wyprowadzić wiele twierdzeń opisujących różne szczegółowe mechanizmy estymacji cyfrowych, wykorzystywane przez umysł podczas przetwarzania liczebników, np.:

(T11) $(\forall c_i, O_j) [c_i \neq \mathbf{0} \rightarrow Seq(O_j)(c_i) />/ O_j(c_i)]$

(T12) $(\forall O_i, c_k) [O_i \neq \mathbf{O}_0 \rightarrow O_i(Sq(c_i)) />/ O_i(c_k)]$

(T13) $(\forall O_i, c_k) [O_i \neq \mathbf{O}_0 \rightarrow O_i(\mathbf{c_{max}}) />/ O_i(c_k)]$

(T14) $(\forall O_i, O_j, c_k) [O_i \neq \mathbf{O}_0 \wedge c_k \neq \mathbf{0} \rightarrow (+(O_i, O_j)\ (c_k) />/ O_j(c_k))]$

(T15) $(\forall O_i, O_j, c_k) [O_j \neq \mathbf{O}_0 \wedge c_k \neq \mathbf{0} \rightarrow (+(O_i, O_j)\ (c_k) />/ O_i(c_k))]$

(T16) $(\forall O_i, O_j, c_k) [O_i\ (>)\ \mathbf{O}_1 \wedge O_j\ (>)\ \mathbf{O}_0 \wedge c_k \neq \mathbf{0} \rightarrow \times(O_i, O_j)\ (c_k) />/ O_j(c_k)]$

(T17) $(\forall O_i, O_j, c_k) [O_j\ (>)\ \mathbf{O}_1 \wedge O_i\ (>)\ \mathbf{O}_0 \wedge c_k \neq \mathbf{0} \rightarrow \times(O_i, O_j)\ (c_k) />/ O_i(c_k)]$

(T18) $(\forall O_i, c_k, n) [O_i\ (>)\ \mathbf{O}_1 \wedge c_k \neq \mathbf{0} \wedge n > 1 \rightarrow O_i^n(c_k) />/ O_i(c_k)]$

(T19) $(\forall O_i, O_j, c_k, c_n) [O_i^2\ (>) \times(O_i, c_k) \wedge O_i\ Kan\ O_j \rightarrow O_j(c_n) />/ O_i(c_k)]$

(T19) wymaga komentarza. Jeśli funktor O_j jest wyznaczany przez funktor O_i jako funktor kanoniczny i elementarna cyfra c_k spełnia warunek $O_i^2\ (>) \times(O_i, c_k)$, to dla dowolnej cyfry c_n struktura $O_j(c_n)$ jest większa od struktury $O_i(c_k)$. Warunek $O_i^2\ (>) \times(O_i, c_k)$ stwierdza, że c_k należy do kanonicznej długości funktora pozycji składniowej O_i. Zilustrujmy (T19). Niech $\mathbf{O}_{10}$ stanowi funktor wyznaczający pozostałe funktory ka-

---

[10] Można skonstruować odpowiedni znak cyfrowy symbolizujący taką cyfrę, traktowany w trybie czytania jako niezłożony. Warto dodać, że istnienie operatora odpowiedniości wyklucza sytuację, w której między cyframi elementarnymi danego systemu występują „przeskoki". Można sobie wyobrazić taki system zapisu liczebników, w którym cyframi elementarnymi są tzw. cyfry nieparzyste, np. od **0** do cyfry **9.** Wówczas struktura głęboka liczebnika *czternaście* przyjmowałaby postać $P[\mathbf{O}_{10}(\mathbf{1}), \mathbf{O}_1(\mathbf{3}), \mathbf{O}_1(\mathbf{1})]$, która nie jest wystandaryzowanym kształtem w dowolnym systemie zapisu liczebników. Istnienie w danym systemie zapisu liczebników funkcji odpowiedniości gwarantuje standaryzację tego systemu.

[11] Oczywiście, rozszerzenie operacji tworzenia funktorów pozycji składniowych przez ich mnożenie cyframi elementarnymi jest dopuszczalne tylko w takich systemach zapisu liczebników, w których istnieje operacja odpowiedniości między cyframi elementarnymi i funktorami pozycji składniowych.

noniczne (w ten sposób powstaje dziesiątkowy układ zapisu cyfr). Załóżmy, że do systemu wprowadzamy jako $\mathbf{c_{max}}$ cyfrę elementarną odpowiadającą liczebnikowi *dziesięć* (w naszym zapisie $\mathbf{c_{10}}$, co stanowi strukturę głęboką cyfry w zapisie powierzchniowym[12]). Łatwo wykazać, że $\mathbf{c_{10}}$ nie przynależy do długości kanonicznej funktora $\mathbf{O_{10}}$, ponieważ $\times(\mathbf{O}_{10}, \mathbf{c_{10}}) = \mathbf{O}_{10\times10}$. Po obliczeniach $\mathbf{O}_{10\times10} = \mathbf{O}_{100}$. Z drugiej strony $\mathbf{O}_{10}{}^2 = \mathbf{O}_{100}$. Funktor pozycji $\mathbf{O}_{100}$ nie jest większy od funktora pozycji $\mathbf{O}_{100}$.

2.3. Moduł przetwarzania struktur głębokich liczebników

Aby sformułować aksjomatykę logiki liczebników, czyli teorii opisującej mechanizmy przetwarzania struktur głębokich liczebników, należy określić, jakie operacje formalne są wykonywane na składnikach struktur głębokich liczebników tak, aby z danej struktury głębokiej mogła zostać poprawnie wyprowadzona inna struktura głęboka, równoważna zakresowo strukturze danej na wejściu. W proponowanej teorii przyjmuje się, że logiczne przetwarzanie struktur głębokich polega na wykonywaniu odpowiednich operacji podstawiania: symbole, z których utworzone są struktury głębokie za pomocą funktora planu pozycyjnego, zastępowane są innymi symbolami tej samej kategorii składniowej. Skoro zgodnie z regułami formatowania struktur głębokich ich składnikami są wyrażenia (struktury) należące do kategorii **K – N**, to wyrażenia stanowiące zarówno argumenty, jak i wartości operacji podstawiania również należą do kategorii **K – N**. Na logikę liczebników składają się właśnie reguły wyznaczające klasę poprawnych operacji podstawiania określonych na liczebnikowych strukturach głębokich.

Pierwszy aksjomat teorii LL jest schematem aksjomatów, który charakteryzuje relację przetwarzania struktur głębokich liczebników, abstrahując od sposobów konstrukcji tych struktur. Niech φ, φ_1, φ_2, φ_3 stanowią zmienne reprezentujące struktury głębokie liczebników:

(AL1) $\varphi_1 \vdash \varphi_2 \wedge \varphi_2 \vdash \varphi_3 \rightarrow \varphi_1 \vdash \varphi_3$

Zgodnie z (AL1) relacja przetwarzania struktur głębokich jest przechodnia. Przechodniość ta pozwala na skonstruowanie pojęcia dowodu liczebnikowego:

(Df. LD) Ciąg struktur głębokich o postaci $<\varphi_1, \ldots, \varphi_k>$ stanowi dowód liczebnikowy wtedy, gdy spełniony jest warunek:

$$(\forall i, k)\,[1 < k \wedge 1 < i \leq k \rightarrow \varphi_i \vdash \varphi_{i+1}]$$

Skoro zgodnie z (Df. LD) w każdym dowodzie liczebnikowym między następującymi po sobie strukturami głębokimi w dowodzie zachodzi relacja redukcji, to na

[12] Ten nowy znak cyfrowy mógłby mieć np. postać #. Wówczas liczbę 110 można by oznaczać napisami „110” oraz „10#”. Napis „1#3” oznaczałby zaś liczbę naturalną *203*. Struktura głęboka takiego złożonego liczebnika cyfrowego miałaby postać $P[\mathbf{O}_1[\mathbf{c_3}], \mathbf{O}_{10}[\mathbf{c_{10}}], \mathbf{O}_{100}[\mathbf{c_1}]]$. Taka cyfra mogłaby być przekładana na liczebnik słowny (neologizm) *sto dziesięćdziesiąt trzy*.

mocy (AL1) relacja redukcji struktur głębokich zachodzi również między dowolną strukturą głęboką w dowodzie a jakąkolwiek strukturą głęboką wcześniej w nim występującą. Tak więc w każdym dowodzie liczebnikowym o postaci $<\varphi_1, \ldots, \varphi_k>$ zachodzi $\varphi_1 \vdash \varphi_2$; $\varphi_2 \vdash \varphi_3$; …; $\varphi_{k-1} \vdash \varphi_k$, a w związku z tym na mocy (AL1) zachodzi m.in. również $\varphi_1 \vdash \varphi_2$; $\varphi_1 \vdash \varphi_3$; …; $\varphi_1 \vdash \varphi_k$. Dlatego też dowód liczebnikowy o postaci $<\varphi_1, \ldots, \varphi_k>$ można określić jako dowód struktury głębokiej φ_k w strukturze głębokiej φ_1. Dowody liczebnikowe mają decydującą o ich swoistości cechę polegającą na tym, że każdy podciąg dowodu liczebnikowego jest również dowodem liczebnikowym. Na mocy definicji dowodu wszystkie jednowyrazowe ciągi struktur głębokich są dowodami liczebnikowymi. Jeśli więc struktury głębokie liczebników zachowują się analogicznie do formuł w dowolnym rachunku logicznym, to LL stanowi, że każda struktura głęboka dowolnego liczebnika rozumiana jako jednowyrazowy ciąg dowodowy jest tezą logiczną tej logiki.

Można rozszerzyć pojęcie formuły w LL tak, aby każda sekwencja struktur głębokich stanowiła formułę LL. Wówczas nie każda w taki sposób rozumiana formuła LL jest tezą logiczną tej logiki. Niech **TLL** stanowi zbiór wszystkich tez logicznych logiki liczebników:

(Df. **TLL**) $(\varphi_1, \ldots, \varphi_k) \in \mathbf{TLL} \equiv_{df} \varphi_1 \vdash \varphi_2 \wedge \ldots \wedge \varphi_{k-1} \vdash \varphi_k$

Ciąg struktur głębokich liczebników stanowi tezę logiki liczebników wtedy, gdy tworzy on dowód liczebnikowy. Ponieważ relacja redukcji liczebników ma zachowywać ich równoważność zakresową, operację dowodzenia na gruncie LL można potraktować jako operację tworzenia skończonych klas kodenotacyjnych struktur głębokich liczebników.

Kolejne aksjomaty ustalają mechanizm przetwarzania struktur głębokich z uwagi na to, w jaki sposób są one zbudowane. Niech α, β, γ, δ, … będą wyrażeniami oznaczającymi wyrażenia kategorii **K – N**. Niech wyrażenie: „…/…" oznacza wynik operacji podstawiania, przy czym sekwencja wyrażeń występująca po lewej stronie ukośnika jest sekwencją zastępowaną, podczas gdy sekwencja wyrażeń z prawej strony jest sekwencją zastępującą. W skrajnym wypadku podstawianie dotyczy jednego wyrażenia zastępowanego i jednego wyrażenia zastępującego. Drugi z aksjomatów również ma postać schematu aksjomatów:

(AL2) $P[\ldots, \alpha, \ldots, \beta, \ldots] \vdash P[\ldots, \alpha/\beta, \ldots, \beta/\alpha\ldots]$

(AL2) głosi, że dowolną strukturę głęboką można poprawnie przetworzyć na strukturę powstającą ze struktury na wejściu w wyniku przestawienia kolejności występowania poszczególnych wyrażeń składowych należących do kategorii **K – N**. Na mocy (AL2) logicznie poprawne są na przykład inferencje:

(i) $P[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2})] \vdash P[\mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2})]$,

(ii) $[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2})] \vdash P[\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2})]$.

Przykład (i) jest poprawny, ponieważ $P[\mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2})] = P[\mathbf{O}_{10}(\mathbf{2})/\mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2})/\mathbf{O}_{10}(\mathbf{2})]$. Z kolei poprawność (ii) wykazuje się w nieco bardziej złożony sposób, korzystając z (AL1):

(1) $P[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2})] \vdash P[\mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2})]$,

(2) $P[\mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2})] \vdash P[\mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2})/\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{2})/\mathbf{O}_{12}\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2})]$.

Stosując (AL1) do (1) i (2), otrzymujemy (ii). Wyznaczony przez aksjomat (AL2) mechanizm przetwarzania struktur głębokich liczebników ma wyjaśniać empiryczny fakt, że liczebniki cyfrowe (w sensie napisów) mogą być odczytywane zgodnie ze zwrotem obliczeniowym lub zwrotem wypowiedzeniowym danego liczebnika złożonego i że ujmowanie przez umysł liczebników cyfrowych nie przebiega w ściśle sekwencyjnie zdeterminowany sposób. Użytkownicy mogą odczytywać je nie tylko od lewej strony, lecz także od strony prawej, a co więcej, z uwagi na tzw. sakkadowość percepcji wzrokowej dopuszczalne są również inne sposoby odczytywania (np. umysł może odczytywać liczebnik cyfrowy „od środka")[13]. Gdyby struktury głębokie liczebników miały charakter ściśle sekwencyjny, to cecha ta musiałaby się przejawiać empirycznie jako sekwencyjność odczytu odpowiednich napisów. Ponieważ sekwencyjność w odczytywaniu liczebników cyfrowych nie jest zdeterminowana porządkiem występowania cyfr elementarnych w sensie napisów w złożonym liczebniku cyfrowym (zgodnie ze zwrotem wypowiedzeniowym lub zwrotem obliczeniowym), struktury głębokie liczebników powinny odzwierciedlać tę cechę w postaci braku porządku sekwencyjnego wśród składników struktur głębokich liczebników. Aksjomat (AL2) opisuje mechanizm, zgodnie z którym przestawienie szyku składników w strukturze głębokiej liczebnika nie wytwarza różnej denotacyjnie struktury głębokiej. A to z kolei przejawia się empirycznie w tym, że sposób percepcji złożonego liczebnika cyfrowego nie wpływa na jego rozumienie w aspekcie denotacyjnym.

Łatwo pokazać, że aksjomat (AL2) pozwala na udowodnienie twierdzenia, w myśl którego relacja redukcji jest zwrotna:

(T20) $\varphi_1 \vdash \varphi_1$

---

[13] Sakkadowość procesu czytania polega na tym, że odczytując dowolny zrozumiały tekst, podlegamy sekwencyjnie uporządkowanym fiksacjom wzrokowym na jego elementach. Fiksacje te mogą trwać dłużej lub krócej, a w czasie lektury jednej linijki tekstu liczba fiksacji może być zmienna. W wypadku długich napisów możemy zafiksować wzrok na dowolnym składniku liczebnika. Stąd na przykład liczebnik „167542312" można zacząć czytać od strony prawej do lewej zgodnie ze schematem 167542-312. Oznaczałoby to, że główna fiksacja w procesie odczytywania tego liczebnika ogniskuje się na cyfrze-napisie „3". Oczywiście, u innego użytkownika proces czytania może przebiegać od lewej do prawej zgodnie ze schematem 1675-42312. W pierwszym wypadku struktura głęboka analizowanej cyfry będzie miała postać $P[\mathbf{O}_{100}(\mathbf{3}), \mathbf{O}_{10}(\mathbf{1}), \mathbf{O}_{1}(\mathbf{2}), \ldots]$. W drugim wypadku składniki $\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{1})$, $\mathbf{O}_{1}(\mathbf{2})$ nie pojawią się wśród początkowych elementów struktury głębokiej. Na temat sakkadowości procesu czytania — zob. Pöppel 1989: 7-23.

Skoro każda operacja odwrócenia szyku składników w dowolnej strukturze głębokiej jest odwracalna, to po zmianie szyku składników w dowolnej strukturze głębokiej zawsze można powrócić do struktury wyjściowej, ponownie odwracając szyk. Przechodniość relacji redukcji gwarantuje zaś, że uzyskana struktura, która jest tożsama ze strukturą na wejściu, jest wynikiem redukcji tej struktury.

Struktury głębokie liczebników zachowują się ekstensjonalnie w procesach przetwarzania. Oznacza to, że podstawienie za dowolny funktor pozycji składniowej w danej strukturze głębokiej innego (pod względem kształtu składniowego) kodenotacyjnego funktora nie zmienia denotacji danej struktury głębokiej. Innymi słowy, struktury głębokie liczebników również redukują się do struktur kodenotacyjnych z uwagi na logiczną operację ekstensjonalności wyznaczoną przez klasyczny rachunek predykatów.

Niech α, β będą zmiennymi przebiegającymi uniwersum wszystkich dowolnie zsyntetyzowanych funktorów pozycji składniowych:

(AL3) $(\forall \alpha, \beta, c)\ [\alpha = \beta \rightarrow P[\ldots, \alpha(c), \ldots] \vdash P[\ldots, \alpha(c)/\beta(c), \ldots]]$

Na mocy schematu (AL3) relacja ⊢ oraz funktor P tworzą więc konteksty ekstensjonalne. Język, w którym sformatowane są reprezentacje umysłowe liczebników, ma charakter ekstensjonalny. Pod schemat (AL3) podpada na przykład aksjomat: $(\forall O_i, O_j, O_k, c)\ [O_i = O_jO_k \rightarrow P[\ldots, O_i(c), \ldots] \vdash P[\ldots, O_i(c)/O_jO_k(c), \ldots]$. Zgodnie z (AL3) struktury głębokie o postaci $P[\ldots, \mathbf{O}_{100}(c), \ldots]$ redukują się kodenotacyjnie do struktur o postaci $P[\ldots, \mathbf{O}_{10}\mathbf{O}_{10}(c), \ldots]$, ponieważ $\mathbf{O}_{100} = \mathbf{O}_{10}\ \mathbf{O}_{10}$. Zatem tezą LL jest $P[\ldots, \mathbf{O}_{100}(c), \ldots] \vdash P[\ldots, \mathbf{O}_{10}\mathbf{O}_{10}(c), \ldots]$.

Kolejne aksjomaty logiki liczebników są już sformułowane w języku, w którym formatuje się struktury głębokie reprezentacji liczebników (a więc w języku przedmiotowym):

(AL4) $(\forall c)\ P[\ldots, \mathbf{O}_0(c), \ldots] \vdash P[\ldots, \mathbf{O}_0(c)/\varnothing, \ldots]$

(AL5) $(\forall c, c_i)\ P[\ldots, \mathbf{O}_0(c)/\varnothing, \ldots] \vdash P[\ldots, \mathbf{O}_0(c_i), \ldots]$

Aksjomat (AL4) opisuje mechanizm przetwarzania struktur głębokich liczebników: każdą strukturę głęboką, w której występuje dowolne podstawienie stałej cyfrowej za wyrażenie $\mathbf{O}_0(c)$, można poprawnie zredukować do struktury głębokiej, w której nie występuje ani wyrażenie $\mathbf{O}_0(c)$, ani jakiekolwiek jego podstawienie. Inaczej mówiąc, (AL4) pozwala umysłowi na wymazanie w dowolnej strukturze głębokiej wyrażenia utworzonego za pomocą funktora zerowej pozycji. Z kolei aksjomat (AL5) opisuje mechanizm odwrotny do mechanizmu opisywanego przez (AL4). Na mocy (AL5) wolno do każdej struktury głębokiej dopisać dowolne wyrażenie utworzone za pomocą funktora zerowej pozycji składniowej. Oba aksjomaty ustalają zbędność funktora pozycji zerowej w procesach przetwarzania struktur głębokich liczebników. Następujące twierdzenie wyraźnie orzeka tę własność inferencyjną funktora pozycji zerowej:

(T21) $(\forall c_1, c_2, c_k)\ \mathrm{P}[\ldots, \mathbf{O}_0(c_1), \ldots] \vdash \mathrm{P}[\ldots, \mathbf{O}_0(c_1)/(\mathbf{O}_0(c_2), \ldots, \mathbf{O}_0(c_k)), \ldots]$

Kolejne aksjomaty opisują mechanizmy przetwarzania struktur głębokich liczebników, w których występuje cyfra elementarna zero:

(AL6) $(\forall O_i)\ \mathrm{P}[\ldots, O_i(\mathbf{0}), \ldots] \vdash \mathrm{P}[\ldots, O_i(\mathbf{0})/\varnothing, \ldots]$

(AL7) $(\forall O_i)\ \mathrm{P}[\ldots, O_i(\mathbf{0})/\varnothing, \ldots] \vdash \mathrm{P}[\ldots, O_i(\mathbf{0}), \ldots]$

Zgodnie z (AL6) każdą strukturę głęboką, w której występuje dowolne wyrażenie kategorii **K – N** z cyfrą **0** jako argumentem funktora pozycji składniowej, można poprawnie zredukować do struktury głębokiej bez takiego wyrażenia. Aksjomat (AL7) opisuje mechanizm operacji odwrotnej do opisanej w (AL6). Z dotychczas wymienionych aksjomatów wynikają następujące twierdzenia:

(T22) $(\forall O_i, O_j)\ \mathrm{P}[\ldots, O_{i\times j}(\mathbf{0}), \ldots] \vdash \mathrm{P}[\ldots, O_{i\times j}(\mathbf{0})/\varnothing, \ldots]$

(T23) $(\forall O_i, O_j)\ \mathrm{P}[\ldots, O_{i+j}(\mathbf{0}), \ldots] \vdash \mathrm{P}[\ldots, O_{i+j}(\mathbf{0})/\varnothing, \ldots]$

(T24) $(\forall O_i)\ \mathrm{P}[\ldots, O_i^n(\mathbf{0}), \ldots] \vdash \mathrm{P}[\ldots, O_i^n(\mathbf{0})/\varnothing, \ldots]$

(T25) $(\forall O_i, O_j)\ \mathrm{P}[\ldots, O_{i\times j}(\mathbf{0})/\varnothing, \ldots] \vdash \mathrm{P}[\ldots, O_{i\times j}(\mathbf{0}), \ldots]$

(T26) $(\forall O_i, O_j)\ \mathrm{P}[\ldots, O_{i+j}(\mathbf{0})/\varnothing, \ldots] \vdash \mathrm{P}[\ldots, O_{i+j}(\mathbf{0}), \ldots]$

(T27) $(\forall O_i)\ \mathrm{P}[\ldots, O_i^n(\mathbf{0})/\varnothing, \ldots] \vdash \mathrm{P}[\ldots, O_i^n(\mathbf{0}), \ldots]$

Twierdzenia (T22)-(T27) opisują mechanizmy przetwarzania struktur głębokich, zgodnie z którymi eliminacja lub dołączenie wyrażenia kategorii **K — N** z cyfrą **0** jako argumentem funktora pozycji składniowej nie zależy od konstrukcji składniowej funktora pozycji, czyli od tego, czy jest on iteracją innych funktorów, czy też został utworzony z innych funktorów na mocy dowolnych operacji tworzenia funktorów pozycji składniowej.

Zarówno aksjomaty (AL6), (AL7), jak i wyszczególnione twierdzenia wyjaśniają zbędność użycia zera jako liczebnika słownego. Zwykle nie używamy takich liczebników, jak *zero tysięcy sto dwa*, *zero milionów sto tysięcy dwadzieścia jeden*: w standardowych sytuacjach komunikacyjnych posługujemy się kodenotacyjnymi liczebnikami *sto dwa* oraz *sto tysięcy dwadzieścia jeden*. Łatwo zauważyć, że struktura głęboka liczebnika *zero tysięcy sto dwa* jest przetwarzalna na strukturę głęboką liczebnika *sto dwa*. Zachodzi bowiem $\mathrm{P}[\mathbf{O}_{1000}(\mathbf{0}), \mathbf{O}_{100}(\mathbf{1}), \mathbf{O}_1(\mathbf{2})] \vdash \mathrm{P}[\mathbf{O}_{100}(\mathbf{1}), \mathbf{O}_1(\mathbf{2})]$. Co więcej, wypowiedzi w rodzaju: *daj mi dwanaście jabłek i zero gruszek*, mimo że są zrozumiałe, w sytuacjach komunikacyjnych mogą wywoływać konsternację. Poczucie „dziwaczności" złożonych liczebników słownych, w których występuje liczebnik *zero*, może zostać zinterpretowane jako przejaw procesu umysłowego polegającego na aktywowaniu się reprezentacji mentalnych liczebników, których struktury głębokie nie są złożone z wyrażeń o postaci $O_i(\mathbf{0})$. Gdy słyszymy liczebnik *zero tysięcy sto*

dwa, umysł syntetyzuje i aktywuje strukturę P[$\mathbf{O}_{1000}(\mathbf{0})$, $\mathbf{O}_{100}(\mathbf{1})$, $\mathbf{O}_{1}(\mathbf{2})$], która natychmiast jest redukowana do krótszej struktury P[$\mathbf{O}_{100}(\mathbf{1})$, $\mathbf{O}_{1}(\mathbf{2})$].

Kolejne aksjomaty LL dotyczą mechanizmów przetwarzania struktur głębokich — ze względu na ich budowę — z funktorów pozycji utworzonych za pomocą operacji mnożenia funktorów pozycji przez cyfry elementarne (jako elementy języka struktur głębokich):

(AL8) $(\forall O_i, c)\ \mathrm{P}[\ldots, O_i(c), \ldots] \vdash \mathrm{P}[\ldots, O_i(c)/O_{i\times c}(\mathbf{1}), \ldots]$

(AL9) $(\forall O_i, c)\ \mathrm{P}[\ldots, O_{i\times c}(\mathbf{1}), \ldots] \vdash \mathrm{P}[\ldots, O_{i\times c}(\mathbf{1})/O_i(c), \ldots]$

Aksjomaty (AL8) i (AL9) opisują mechanizm przetwarzania w obie strony dowolnych struktur głębokich na kodenotacyjne struktury sformatowane (w aspekcie cyfrowym) wyłącznie za pomocą cyfry elementarnej **1.** Na przykład, strukturę głęboką P[$\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{1}(\mathbf{2})$] umysł może zredukować do struktury P[$\mathbf{O}_{300}(\mathbf{1})$, $\mathbf{O}_{30}(\mathbf{1})$, $\mathbf{O}_{2}(\mathbf{1})$] (zgodnie z inferencją P[$\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{1}(\mathbf{2})$] $\vdash$ P[$\mathbf{O}_{300}(\mathbf{1})$, $\mathbf{O}_{30}(\mathbf{1})$, $\mathbf{O}_{2}(\mathbf{1})$]). Inferencja odwrotna, czyli P[$\mathbf{O}_{300}(\mathbf{1})$, $\mathbf{O}_{30}(\mathbf{1})$, $\mathbf{O}_{2}(\mathbf{1})$] $\vdash$ P[$\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{1}(\mathbf{2})$] wymaga użycia aksjomatów (AL3) i (AL1). Skoro $\mathbf{O}_{300} = \mathbf{O}_{100\times3}$, $\mathbf{O}_{30} = \mathbf{O}_{10\times3}$, $\mathbf{O}_{1\times2} = \mathbf{O}_{2}$, to zgodnie z (AL3) P[$\mathbf{O}_{300}(\mathbf{1})$, $\mathbf{O}_{30}(\mathbf{1})$, $\mathbf{O}_{2}(\mathbf{1})$] $\vdash$ P[$\mathbf{O}_{100\times3}(\mathbf{1})$, $\mathbf{O}_{10\times3}(\mathbf{1})$, $\mathbf{O}_{1\times2}(\mathbf{1})$]. Na mocy (AL9) zachodzi P[$\mathbf{O}_{100\times3}(\mathbf{1})$, $\mathbf{O}_{10\times3}(\mathbf{1})$, $\mathbf{O}_{1\times2}(\mathbf{1})$] $\vdash$ P[$\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{1}(\mathbf{2})$]. Zatem na mocy (AL1) dostajemy P[$\mathbf{O}_{300}(\mathbf{1})$, $\mathbf{O}_{30}(\mathbf{1})$, $\mathbf{O}_{2}(\mathbf{1})$] $\vdash$ P[$\mathbf{O}_{100\times3}(\mathbf{1})$, $\mathbf{O}_{10\times3}(\mathbf{1})$, $\mathbf{O}_{1\times2}(\mathbf{1})$] $\wedge$ P[$\mathbf{O}_{100\times3}(\mathbf{1})$, $\mathbf{O}_{10\times3}(\mathbf{1})$, $\mathbf{O}_{1\times2}(\mathbf{1})$] $\vdash$ P[$\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{1}(\mathbf{2})$] $\rightarrow$ P[$\mathbf{O}_{300}(\mathbf{1})$, $\mathbf{O}_{30}(\mathbf{1})$, $\mathbf{O}_{2}(\mathbf{1})$] $\vdash$ P[$\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{1}(\mathbf{2})$]. Ostatecznie, umysł wytwarza za pomocą *modus ponens* P[$\mathbf{O}_{300}(\mathbf{1})$, $\mathbf{O}_{30}(\mathbf{1})$, $\mathbf{O}_{2}(\mathbf{1})$] $\vdash$ P[$\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{1}(\mathbf{2})$].

Aksjomat (AL8) wyjaśnia fakt, że każdy liczebnik w dowolnym systemie zapisu jest redukowalny do liczebnika w sensie powierzchniowym utworzonego wyłącznie z cyfry *jeden* oraz odpowiednich funktorów pozycji składniowych. Innymi słowy, każdy liczebnik w dowolnym systemie zapisu jest redukowalny do kodenotacyjnego liczebnika w dowolnym systemie binarnym, a w szczególności w wystandaryzowanym systemie zero-jedynkowym.

Kolejne aksjomaty opisują mechanizmy przetwarzania struktur głębokich liczebników pozwalające wytwarzać kodenotacyjne struktury głębokie, które różnią się od wyjściowej struktury długością pozycyjną, metryką lub stopniem głębokości.

(AL10) $(\forall O_i, O_j)\ \mathrm{P}[\ldots, O_{i+j}(c), \ldots] \vdash \mathrm{P}[\ldots, O_{i+j}(c)/(O_i(c), O_j(c)), \ldots]$

(AL11) $(\forall O_i, O_j)\ \mathrm{P}[\ldots, O_i(c), O_j(c), \ldots] \vdash \mathrm{P}[\ldots, (O_i(c), O_j(c))/O_{i+j}(c), \ldots]$

Aksjomaty (AL10) i (AL8) wyjaśniają między innymi fakt, że każdy liczebnik można zapisać w danym systemie zapisu liczebników za pomocą karbów (czyli za pomocą *jedynek* i funktora pozycji jedności). Mechanizm przekształcania dowolnego liczebnika na liczebnik o postaci *jeden i jeden i jeden*… przebiega następująco: najpierw dla każdego O_i w danej strukturze głębokiej φ, φ jest przetwarzane zgodnie z (AL8) na kodenotacyjną strukturę φ*, w której dla dowolnego c zamiast $O_i(c)$ występuje $O_{i\times c}(\mathbf{1})$. W drugiej fazie struktura φ* jest rekurencyjnie przetwarzana na mocy

mechanizmu opisanego w (AL10) do postaci, w której każde $O_{i\times c}(\mathbf{1})$ w φ* zostaje zredukowane do odpowiedniego ciągu o postaci: $\mathbf{O}_1(\mathbf{1})$, …, $\mathbf{O}_1(\mathbf{1})$. Następujący ciąg inferencji stanowi przykład redukcji struktury głębokiej P[$\mathbf{O}_2(\mathbf{3})$, $\mathbf{O}_3(\mathbf{2})$] do struktury głębokiej liczebników-karbów: P[$\mathbf{O}_2(\mathbf{3})$, $\mathbf{O}_3(\mathbf{2})$] ⊢ P[$\mathbf{O}_{2\times3}(\mathbf{1})$, $\mathbf{O}_3(\mathbf{2})$] ⊢ P[$\mathbf{O}_{2\times3}(\mathbf{1})$, $\mathbf{O}_{3\times2}(\mathbf{1})$] ⊢ P[$\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_{3\times2}(\mathbf{1})$] ⊢ P[$\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$]. Ostatnia z wytworzonych struktur jest strukturą głęboką reprezentacji umysłowych dowolnego liczebnika zbudowanego z dwunastu karbów (kresek, nacięć, kamyków, koralików itp.).

Z kolei aksjomaty (A11) i (A9) umożliwiają proces odwrotny, polegający na przetworzeniu dowolnej struktury głębokiej liczebnika-karbu na strukturę wykorzystującą funktory pozycji składniowych różne od $\mathbf{O}_1$ oraz cyfry elementarne różne od **1**. W szczególnym wypadku dowolną strukturę głęboką liczebnika-karbu można zredukować do struktury głębokiej zbudowanej z dokładnie jednego funktora pozycji składniowej według zasady: jeśli w liczebniku występuje *n* karbów, to strukturę głęboką liczebnika-karbu można zredukować do kodenotacyjnej struktury głębokiej o postaci P[$O_n(\mathbf{1})$].

Na podstawie wprowadzonych dotychczas aksjomatów można udowodnić twierdzenie:

(T28) $(\forall O_i, c)\ [c \neq \mathbf{c_{max}} \rightarrow \mathrm{P}[\ldots, O_i(Sq(c)), \ldots] \vdash \mathrm{P}[\ldots, O_i(Sq(c))/(O_i(c), O_i(\mathbf{1})), \ldots]]$

(T28) opisuje mechanizm przetwarzania struktur głębokich liczebników polegający na tym, że dowolne wyrażenie z dowolnym funktorem pozycji składniowej oraz dowolną cyfrą elementarną mniejszą od cyfry maksymalnej można zredukować w dowolnej strukturze głębokiej do sekwencji dwóch wyrażeń: z tym samym funktorem działającym na cyfrę poprzedzającą daną cyfrę w porządku cyfr elementarnych danego systemu oraz z tym samym funktorem działającym na cyfrę elementarną **1**. Na przykład, w systemie zapisu liczebników, w którym cyfra **2** nie jest cyfrą maksymalną, (T28) wyjaśnia poprawność takiego oto ciągu inferencji liczebnikowych: P[$\mathbf{O}_2(\mathbf{3})$] ⊢ P[$\mathbf{O}_2(\mathbf{2})$, $\mathbf{O}_2(\mathbf{1})$] ⊢ P[$\mathbf{O}_2(\mathbf{1})$, $\mathbf{O}_2(\mathbf{1})$, $\mathbf{O}_2(\mathbf{1})$]. Ten mechanizm przetwarzania struktur głębokich pozwala wyjaśnić m.in. brak konsternacji, gdy poprosiwszy kogoś: *daj mi trzydzieści złotych*, otrzymujemy *trzy* banknoty *po dziesięć* złotych. Strukturę głęboką liczebnika *trzy po dziesięć* można zrekonstruować jako strukturę o postaci P[$\mathbf{O}_{10}(\mathbf{1})$, $\mathbf{O}_{10}(\mathbf{1})$, $\mathbf{O}_{10}(\mathbf{1})$]. Struktura ta jest kodenotacyjna ze strukturą głęboką liczebnika *trzydzieści* P[$\mathbf{O}_{10}(\mathbf{3})$].

Kolejne aksjomaty opisują mechanizm przetwarzania struktur głębokich liczebników, w których występują wyrażenia z superpozycjami lub iteracjami funktorów pozycji składniowych.

(AL12) $(\forall O_i, O_j, c)\ \mathrm{P}[\ldots, O_iO_j(c), \ldots] \vdash \mathrm{P}[\ldots, O_iO_j(c)/O_{i\times j}(c), \ldots]$

(AL13) $(\forall O_i, O_j, c)\ \mathrm{P}[\ldots, O_{i\times j}(\mathrm{c}), \ldots] \vdash \mathrm{P}[\ldots, O_{i\times j}(c)/O_iO_j(c), \ldots]$

Zgodnie z (AL12) liczebnik o postaci *trzydzieści setek* można przetworzyć na liczebnik *trzy tysiące*, ponieważ zachodzi $P[\mathbf{O}_{100}\mathbf{O}_{10}(\mathbf{3})] \vdash P[\mathbf{O}_{100\times10}(\mathbf{3})]$, a skoro $\mathbf{O}_{100\times10} = \mathbf{O}_{1000}$, to na mocy (AL3) dostajemy $P[\mathbf{O}_{100}\mathbf{O}_{10}(\mathbf{3})] \vdash P[\mathbf{O}_{1000}(\mathbf{3})]$.

Łatwo wykazać, że relacja redukcji struktur głębokich liczebników $\vdash$ jest relacją symetryczną.

(T29) $\varphi_1 \vdash \varphi_2 \rightarrow \varphi_2 \vdash \varphi_1$

Dowód (T29) przebiega według schematu:

(i) skoro jakiekolwiek przetwarzanie struktur głębokich liczebników polega na zastosowaniu odpowiednio scharakteryzowanej przez aksjomaty operacji podstawiania, to jeśli $\varphi_1 \vdash \varphi_2$, to istnieją takie struktury głębokie $\varphi_i, \ldots, \varphi_k$, które tworzą ciąg inferencji $\varphi_1 \vdash \varphi_i, \ldots, \varphi_{k-1} \vdash \varphi_k, \varphi_k \vdash \varphi_2$, taki że każda inferencja w tym ciągu jest egzemplifikacją typów wyznaczonych przez aksjomaty (AL2)-(AL13);

(ii) wszystkie typy inferencji wyznaczone przez aksjomaty (AL2)-(AL13) spełniają warunek symetryczności, a wobec tego, skoro $\varphi_1 \vdash \varphi_i, \ldots, \varphi_{k-1} \vdash \varphi_k, \varphi_k \vdash \varphi_2$, to zachodzi $\varphi_2 \vdash \varphi_k, \varphi_k \vdash \varphi_{k-1}, \ldots, \varphi_{i} \vdash \varphi_1$;

(iii) zatem na mocy przechodniości relacji $\vdash$ ustalonej przez aksjomat (AL1) wnioskujemy, że $\varphi_2 \vdash \varphi_1$.

Skoro relacja redukcji struktur głębokich liczebników jest zarazem zwrotna, symetryczna i przechodnia, to jest równoważnościowa. Relacja $\vdash$ tworzy więc klasy abstrakcji kodenotacyjnych[14] struktur głębokich liczebników.

LL pozwala na wyjaśnienie wielu faktów dotyczących czynności obliczeniowych umysłu. Na przykład, gdy wiemy, że ekspedient w sklepie ma wydać nam *pięć* złotych reszty, to zdajemy sobie sprawę, że w naszym systemie monetarnym (złotowym) może to zrobić na różne sposoby. Może wydać nam *jedną pięciozłotówkę* lub *dwie dwuzłotówki i jedną złotówkę* czy też *pięć jednozłotówek*. Wiedza, że (i) *pięć złotych* jest kodenotacyjne z (ii) *jedną pięciozłotówką*, (iii) *dwiema dwuzłotówkami i jedną złotówką*, (iv) *pięcioma jednozłotówkami*, jest wynikiem wykonywanych przez nas operacji przetwarzania struktur głębokich liczebników. We frazach wymienionych w (i)-(iv) występują liczebniki słowne o następujących strukturach głębokich: (i) $P[\mathbf{O}_1(\mathbf{5})]$, (ii) $P[\mathbf{O}_5(\mathbf{1})]$, (iii) $P[\mathbf{O}_2(\mathbf{2}), \mathbf{O}_1(\mathbf{1})]$, (iv) $P[\mathbf{O}_1(\mathbf{1}), \mathbf{O}_1(\mathbf{1}), \mathbf{O}_1(\mathbf{1}), \mathbf{O}_1(\mathbf{1}), \mathbf{O}_1(\mathbf{1})]$. Wszystkie wyszczególnione struktury głębokie należą do tej samej klasy abstrakcji relacji $\vdash$.

Aby zilustrować sposób, w jaki LL działa w procesach przetwarzania struktur głębokich liczebników, przedstawimy dowód inferencji $P[\mathbf{O}_1(\mathbf{5})] \vdash P[\mathbf{O}_2(\mathbf{2}), \mathbf{O}_1(\mathbf{1})]$ (odpowiedzialnej za to, że zazwyczaj nie awanturujemy się ze sprzedawcą, gdy wydając resztę pięciu złotych, wręcza nam dwie dwuzłotówki i jedną złotówkę):

(1) $P[\mathbf{O}_1(\mathbf{5})]$ założenie

[14] Zostanie to wykazane w części semantycznej teorii liczebników.

(2)	$P[\mathbf{O}_{1\times5}(\mathbf{1})]$	(1), (AL8)
(3)	$P[\mathbf{O}_{5}(\mathbf{1})]$	(2), (Df. $\times$)
(4)	$P[\mathbf{O}_{2+3}(\mathbf{1})]$	(3), (AL3), $\mathbf{O}_{2+3} = \mathbf{O}_5$
(5)	$P[\mathbf{O}_{2}(\mathbf{1}), \mathbf{O}_{3}(\mathbf{1})]$	(4), (AL10)
(6)	$P[\mathbf{O}_{2}(\mathbf{1}), \mathbf{O}_{2+1}(\mathbf{1})]$	(5), (AL3), $\mathbf{O}_{2+1} = \mathbf{O}_3$
(7)	$P[\mathbf{O}_{2}(\mathbf{1}), \mathbf{O}_{2}(\mathbf{1}), \mathbf{O}_{1}(\mathbf{1})]$	(6), (AL10)
(8)	$P[\mathbf{O}_{1\times2}(\mathbf{1}), \mathbf{O}_{1\times2}(\mathbf{1}), \mathbf{O}_{1}(\mathbf{1})]$	(7), (AL3),$\mathbf{O}_{1\times\mathbf{2}} = \mathbf{O}_2$
(9)	$P[\mathbf{O}_{1}(\mathbf{2}), \mathbf{O}_{1\times2}(\mathbf{1}), \mathbf{O}_{1}(\mathbf{1})]$	(8), (AL9)
(10)	$P[\mathbf{O}_{1}(\mathbf{2}), \mathbf{O}_{1}(\mathbf{2}), \mathbf{O}_{1}(\mathbf{1})]$	(9), (AL9)
(11)	$P[\mathbf{O}_{1+1}(\mathbf{2}), \mathbf{O}_{1}(\mathbf{1})]$	(10), (AL11)
(12)	$P[\mathbf{O}_{2}(\mathbf{2}), \mathbf{O}_{1}(\mathbf{1})]$	(11), (AL3), $\mathbf{O}_{1+1} = \mathbf{O}_2$

Zgodnie z (Df. **TLL**) sekwencja <(1), …, (12)> stanowi więc dowód liczebnikowy. Stąd na mocy aksjomatu (AL1) wyprowadzamy $P[\mathbf{O}_{1}(\mathbf{5})] \vdash P[\mathbf{O}_{2}(\mathbf{2}), \mathbf{O}_{1}(\mathbf{1})]$. W podobny sposób można wykazać, że $P[\mathbf{O}_{1}(\mathbf{5})] \vdash P[\mathbf{O}_{1}(\mathbf{1}), \mathbf{O}_{1}(\mathbf{1}), \mathbf{O}_{1}(\mathbf{1}), \mathbf{O}_{1}(\mathbf{1}), \mathbf{O}_{1}(\mathbf{1})]$ i $P[\mathbf{O}_{1}(\mathbf{5})] \vdash P[\mathbf{O}_{5}(\mathbf{1})]$.

Logika liczebników pozwala również wyjaśniać, dlaczego algorytm pisemnego dodawania liczebników cyfrowych jest poprawny. Wymaga to jednak wprowadzenia do LL aksjomatów opisujących takie działania na strukturach głębokich liczebników, jak dodawanie i mnożenie. Wcześniejsze aksjomaty opisują jedynie działania na wewnętrznych elementach struktur głębokich liczebników. LL pozwala ponadto wyjaśnić fakt trafnych oszacowań liczebników pod względem tego, który z liczebników denotuje większą liczbę lub liczność. To z kolei wymaga jednak wprowadzenia do LL relacji większości między strukturami głębokimi.

2.4. Moduł działań na liczebnikach i ich szacowania

W różnych sytuacjach obliczeniowych umysł przeprowadza rozmaite operacje na liczebnikach. Przedstawię teraz model przeprowadzania operacji dodawania liczebników oraz ich porównywania z uwagi na relację większości. Na przykład, gdy sprawdzamy, czy wystarczy nam pieniędzy na zakup kilku produktów, dodajemy ich ceny wyrażone w liczebnikach. Również pisemny algorytm dodawania jest przykładem operacji dodawania liczebników — w tym wypadku liczebników cyfrowych. Efektywne przeprowadzanie takich operacji nie musi wcale odwoływać się do reguł wywiedzionych z arytmetyki Peana. Dodając liczebniki w różnych sytuacjach obliczeniowych, nie wykorzystujemy rekurencyjnej definicji dodawania z PA. Podobnie, nie korzystamy z aksjomatów Peana, szacując, że kiedy otrzymuje się 200 złotych za

godzinę pracy, to dostaje się więcej niż wówczas, gdy godzina pracy jest wyceniana na 98 złotych.

2.4.1. DODAWANIE

Reprezentacją umysłową dodawania liczebników jest operacja fuzji ich struktur głębokich.

(Df. $\oplus$) $\quad P[O_i\ (c_k),\ \ldots,\ O_j(c_h)] \oplus P[O_n(c_l),\ldots,\ O_m(c_d)] = P[O_i\ (c_k),\ \ldots,\ O_j(c_h),\ O_n(c_l),\ \ldots,\ O_m(c_d)]$

Operacja fuzji struktur głębokich liczebników „zlewa" więc w jedną strukturę wszystkie wyrażenia konstytuujące każdą strukturę z osobna, np. $P[\mathbf{O}_{100}(\mathbf{3}),\ \mathbf{O}_{10}(\mathbf{3}),\ \mathbf{O}_1(\mathbf{2})] \oplus P[\mathbf{O}_2(\mathbf{2}),\ \mathbf{O}_1(\mathbf{1})] = P[\mathbf{O}_{100}(\mathbf{3}),\ \mathbf{O}_{10}(\mathbf{3}),\ \mathbf{O}_1(\mathbf{2}),\ \mathbf{O}_2(\mathbf{2}),\ \mathbf{O}_1(\mathbf{1})]$. Na podstawie operacji $\oplus$ można wyjaśnić mechanizm kognitywny pisemnego dodawania. Zdefiniujmy funkcję odległości między funktorami pozycji składniowych:

(Df. *D*) $\quad D(O_i, O_j) = O_k \equiv_{df} [O_i = Seq^k(O_j) \vee O_j = Seq^k(O_i)]$

Odległością między dwoma funktorami pozycji składniowych O_i oraz O_j jest potęga operacji następnika, na mocy której z funktora O_i umysł syntetyzuje funktor O_j, lub potęga operacji następnika, na mocy której z funktora O_j umysł syntetyzuje funktor O_i. Definicja (Df. *D*) dostarcza narzędzia formalnego w postaci funkcji odległości potrzebnego do sformułowania w teorii LL jednego z twierdzeń wyjaśniających kognitywny mechanizm algorytmu dodawania cyfr w dowolnym układzie cyfrowym. Oto pierwsze z tych twierdzeń:

(T30) $\quad (\forall O_i, O_j, c_1, c_2)\ [O_i\ Kan\ O_j \wedge O_i\ (>)\ +(c_1{*}, c_2{*}) \rightarrow (\exists k)\ (P[\ldots, O_j(c_1), O_j(c_2), \ldots] \vdash P[\ldots, (O_j(c_1), O_j(c_2))/O_j(Sq^k(c_1)), \ldots] \wedge c_2{*} = Seq^k(\mathbf{O}_0))]$

Zgodnie z (T30), jeśli funktor pozycji O_i wyznacza funktor O_j jako kanoniczny i suma funktorów pozycji skorelowanych na mocy funkcji odpowiedniości z danymi cyframi elementarnymi związanymi w strukturze głębokiej przez funktor O_j jest mniejsza od funktora wyznaczającego funktory kanoniczne, to z każdej struktury o postaci $P[\ldots, O_j(c_1), O_j(c_2), \ldots]$ jest wyprowadzana struktura o postaci $P[\ldots, O_j(Sq^k(c_1)), \ldots]$, gdzie potęga następników cyfry elementarnej c_1 spełnia warunek $c_2{*} = Seq^k(\mathbf{O}_0)$. Innymi słowy, struktura o postaci $P[\ldots, O_j(c_1), O_j(c_2), \ldots]$ redukuje się do struktury o postaci $P[\ldots, O_j(c_3), \ldots]$, takiej że cyfra elementarna c_3 jest odpowiednio większa od cyfry c_1.

Podajmy przykład pokazujący, jak (T30) reguluje mechanizm pisemnego dodawania liczebników cyfrowych. Dodanie liczebników „234" oraz „325" przebiega według algorytmu dodania cyfr z tej samej kolumny. Struktury głębokie obu cyfr złożonych mają postać $P[\mathbf{O}_{100}(\mathbf{2}),\ \mathbf{O}_{10}(\mathbf{3}),\ \mathbf{O}_1(\mathbf{4})]$, $P[\mathbf{O}_{100}(\mathbf{3}),\ \mathbf{O}_{10}(\mathbf{2}),\ \mathbf{O}_1(\mathbf{5})]$. Dodając obie cyfry, umysł dokonuje fuzji obu struktur głębokich $P[\mathbf{O}_{100}(\mathbf{2}),\ \mathbf{O}_{10}(\mathbf{3}),\ \mathbf{O}_1(\mathbf{4})] \oplus P[\mathbf{O}_{100}(\mathbf{3}),\ \mathbf{O}_{10}(\mathbf{2}),\ \mathbf{O}_1(\mathbf{5})] = P[\mathbf{O}_{100}(\mathbf{2}),\ \mathbf{O}_{10}(\mathbf{3}),\ \mathbf{O}_1(\mathbf{4}),\ \mathbf{O}_{100}(\mathbf{3}),\ \mathbf{O}_{10}(\mathbf{2}),\ \mathbf{O}_1(\mathbf{5})]$. Na mocy wielokrotnego stosowania inferencji opisanej aksjomatem (AL2) $P[\ldots, \alpha, \ldots, \beta, \ldots]$

$\vdash$ P[…, α/β, …, β/α, …], tworzona jest struktura głęboka P[$\mathbf{O}_{100}(\mathbf{2})$, $\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(\mathbf{4})$, $\mathbf{O}_{1}(\mathbf{5})$]. Wszystkie funktory pozycji syntaktycznych są w niej wyznaczane jako kanoniczne przez funktor $\mathbf{O}_{10}$: (i) $\mathbf{O}_{10}$ *Kan* $\mathbf{O}_{1}$, (ii) $\mathbf{O}_{10}$ *Kan* $\mathbf{O}_{10}$, (iii) $\mathbf{O}_{10}$ *Kan* $\mathbf{O}_{100}$. Co więcej, dla wszystkich cyfr elementarnych w analizowanej strukturze zachodzi warunek $\mathbf{O}_{10}$ (>) +(c_1*, c_2*), czyli (i*) $\mathbf{O}_{10}$ (>) +(**4***, **5***), (ii*) $\mathbf{O}_{10}$ (>) +(**3***, **2***), (iii) $\mathbf{O}_{10}$ (>) +(**2***, **3***).

Dla przykładu sprawdźmy, czy zachodzi (i*). Odpowiednikiem cyfry elementarnej **4** jest $\mathbf{O}_4$, **4*** = $\mathbf{O}_4$. Analogicznie, **5*** = $\mathbf{O}_5$. Stąd +(**4***, **5***) = +($\mathbf{O}_4$, $\mathbf{O}_5$). Zgodnie z (AO5) zachodzi +($\mathbf{O}_4$, $\mathbf{O}_5$) = $\mathbf{O}_9$. Ostatecznie wyprowadzamy $\mathbf{O}_{10}$ (>) $\mathbf{O}_9$. Na podstawie (T30), dla dowolnych dwóch funktorów pozycji składniowych o tym samym indeksie, ze struktury P[…, $O_j(c_1)$, $O_j(c_2)$, …] spełniającej warunki wyznaczone przez O_i *Kan* O_j, O_i (>) +(c_1*, c_2*) można wyprowadzić P[…, $O_j(Sq^k(c_1))$, …], o ile c_2* = $Seq^k(\mathbf{O}_0)$. Ze struktury P[$\mathbf{O}_{100}(\mathbf{2})$, $\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(\mathbf{4})$, $\mathbf{O}_{1}(\mathbf{5})$] jest zatem wyprowadzana struktura o postaci P[$\mathbf{O}_{100}(\mathbf{2})$, $\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(Sq^k(\mathbf{4}))$], o ile **5*** = $Seq^k(\mathbf{O}_0)$. Dla jakiego k zachodzi **5*** = $Seq^k(\mathbf{O}_0)$? Skoro **5*** = $\mathbf{O}_5$, to $\mathbf{O}_5$ = *SeqSeqSeqSeqSeq*($\mathbf{O}_0$). Na mocy definicji potęgi funkcji następnika dla funktorów pozycji składniowych wyprowadzamy $\mathbf{O}_5 = Seq^5(\mathbf{O}_0)$. Stąd k = 5. Jeśli tak, to ze względu na warunek **5*** = $Seq^k(\mathbf{O}_0)$ zachodzi $Sq^k(\mathbf{4}) = Sq^5(\mathbf{4})$. W systemie dziesiętnym $Sq^5(\mathbf{4}) = \mathbf{9}$. Zatem P[$\mathbf{O}_{100}(\mathbf{2})$, $\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(\mathbf{4})$, $\mathbf{O}_{1}(\mathbf{5})$] $\vdash$ P[$\mathbf{O}_{100}(\mathbf{2})$, $\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(\mathbf{9})$]. W analogiczny sposób w wypadku (ii*) otrzymujemy inferencję P[$\mathbf{O}_{100}(\mathbf{2})$, $\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(\mathbf{9})$] $\vdash$ P[$\mathbf{O}_{100}(\mathbf{2})$, $\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{5})$, $\mathbf{O}_{1}(\mathbf{9})$], a dla (iii*) — P[$\mathbf{O}_{100}(\mathbf{2})$, $\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{5})$, $\mathbf{O}_{1}(\mathbf{9})$] $\vdash$ P[$\mathbf{O}_{100}(\mathbf{5})$, $\mathbf{O}_{10}(\mathbf{5})$, $\mathbf{O}_{1}(\mathbf{9})$]. Ostatecznie więc, dodając pisemnie „234” oraz „325”, otrzymuje się w umyśle wynik P[$\mathbf{O}_{100}(\mathbf{2})$, $\mathbf{O}_{10}(\mathbf{3})$, $\mathbf{O}_{1}(\mathbf{4})$] $\oplus$ P[$\mathbf{O}_{100}(\mathbf{3})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(\mathbf{5})$] $\vdash$ P[$\mathbf{O}_{100}(\mathbf{5})$, $\mathbf{O}_{10}(\mathbf{5})$, $\mathbf{O}_{1}(\mathbf{9})$].

Następujący przykład ilustruje proces przetwarzania liczebników podczas pisemnego dodawania w systemie czwórkowym, czyli takim, w którym cyframi elementarnymi są **0**, **1**, **2**, **3**. Dodajmy pisemnie dwa liczebniki cyfrowe zapisane w systemie czwórkowym — „11” i 10”. Funktor pozycji $\mathbf{O}_4$ wyznacza kanoniczne funktory pozycji składniowych; w naszym wypadku będą to zgodnie z (Df. *Kan*) kolejno funktory: $\mathbf{O}_1$, $\mathbf{O}_4$, $\mathbf{O}_{16}$, $\mathbf{O}_{64}$ itd. Struktury głębokie „11” i „10” mają więc postać P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$], P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(\mathbf{0})$]. Zgodnie z aksjomatem (AL2) fuzja obu struktur po przekształceniach przyjmuje formę P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$] $\oplus$ P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(\mathbf{0})$] = P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{0})$]. Ponieważ zachodzi (i) $\mathbf{O}_4$ (>) +(**1***, **1***), (ii) $\mathbf{O}_4$ (>) +(**1***, **0***), operacja pisemnego dodawania obu cyfr jest regulowana przez mechanizm opisany przez (T30): (i) P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{0})$] $\vdash$ P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(Sq^k(\mathbf{1}))$], o ile **0*** = $Seq^k(\mathbf{O}_0)$. Zgodnie z (Df. Seq^n) oraz (Df. *) — k = 0. Zatem $Sq^k(\mathbf{1}) = Sq^0(\mathbf{1})$, czyli $Sq^k(\mathbf{1}) = \mathbf{1}$. Inferencja (i) przyjmuje więc postać P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$, $\mathbf{O}_1(\mathbf{0})$] $\vdash$ P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$]. W podobny sposób wyprowadzić można inferencję (ii): P[$\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_4(\mathbf{1})$, $\mathbf{O}_1(\mathbf{1})$] $\vdash$ P[$\mathbf{O}_4(\mathbf{2})$, $\mathbf{O}_1(\mathbf{1})$]. Zatem wynik dodawania pisemnego w systemie czwórkowym obu cyfr należy zapisać jako 11 + 10 = 21.

(T30) nie opisuje jednak algorytmu przenoszenia jedności, dziesiątek, setek itd. do następnej kolumny w procedurze dodawania pisemnego. Ten mechanizm jest regulowany przez inne twierdzenie:

(T31) $(\forall O_i, O_j, c_1, c_2)\,[O_i\ Kan\ O_j \wedge \sim O_i\ (>)\ +(c_1{*}, c_2{*}) \rightarrow (\exists k)\ (\mathrm{P}[\ldots, O_j(c_1), O_j(c_2), \ldots] \vdash \mathrm{P}[\ldots, (O_j(c_1), O_j(c_2))/(O_{j\times i}(\mathbf{1}), O_j(Sq^k(\mathbf{0}))), \ldots] \wedge Sq^k(\mathbf{0}){*} = D(O_i, +(c_1{*}, c_2{*})))]$

Zilustrujmy mechanizm dodawania pisemnego w systemie dziesiętnym operacją o postaci 27 + 26. Struktury głębokie rozważanych liczebników cyfrowych mają postać $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{7})]$, $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{6})]$. Dodając „26” do „27”, dokonujemy fuzji obu struktur głębokich $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{7})] \oplus \mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{6})] = \mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{7}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{6})]$. Następnie przeprowadzamy inferencję, opierając się na mechanizmie opisanym w (AL2): $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{7}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{6})] \vdash \mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{6}), \mathbf{O}_1(\mathbf{7})]$. Tę ostatnią strukturę umysł przetwarza dalej zgodnie z mechanizmem opisanym w twierdzeniu (T31). Ponieważ zachodzi $\mathbf{O}_{10}\ Kan\ \mathbf{O}_1$ oraz $\sim \mathbf{O}_{10}\ (>)\ +(\mathbf{6}{*}, \mathbf{7}{*})$, umysł przeprowadza inferencję o postaci $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{6}), \mathbf{O}_1(\mathbf{7})] \vdash \mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10\times1}(\mathbf{1}), \mathbf{O}_1(Sq^k(\mathbf{0}))]$, gdzie $Sq^k(\mathbf{0}){*} = D(\mathbf{O}_{10}, +(\mathbf{6}{*}, \mathbf{7}{*}))$. Skoro $D(\mathbf{O}_{10}, +(\mathbf{6}{*}, \mathbf{7}{*})) = D(\mathbf{O}_{10}, \mathbf{O}_{13})$, to $Sq^k(\mathbf{0}){*} = \mathbf{O}_3$, a w konsekwencji $Sq^k(\mathbf{0}) = \mathbf{3}$. Inferencja $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{6}), \mathbf{O}_1(\mathbf{7})] \vdash \mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10\times1}(\mathbf{1}), \mathbf{O}_1(Sq^k(\mathbf{0}))]$ ma więc postać $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{6}), \mathbf{O}_1(\mathbf{7})] \vdash \mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10\times1}(\mathbf{1}), \mathbf{O}_1(\mathbf{3})]$. Na mocy definicji (Df. ×*) oraz (AL3) umysł przeprowadza inferencję $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10\times1}(\mathbf{1}), \mathbf{O}_1(\mathbf{3})] \vdash \mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{1}), \mathbf{O}_1(\mathbf{3})]$. Następnie zgodnie z mechanizmem opisanym w (T30) dokonujemy inferencji $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{1}), \mathbf{O}_1(\mathbf{3})] \vdash \mathrm{P}[\mathbf{O}_{10}(\mathbf{4}), \mathbf{O}_{10}(\mathbf{1}), \mathbf{O}_1(\mathbf{3})]$, $\mathrm{P}[\mathbf{O}_{10}(\mathbf{4}), \mathbf{O}_{10}(\mathbf{1}), \mathbf{O}_1(\mathbf{3})] \vdash \mathrm{P}[\mathbf{O}_{10}(\mathbf{5}), \mathbf{O}_1(\mathbf{3})]$. Ostatecznie więc umysł otrzymuje wynik obliczenia $\mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{7})] \oplus \mathrm{P}[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{6})] \vdash \mathrm{P}[\mathbf{O}_{10}(\mathbf{5}), \mathbf{O}_1(\mathbf{3})]$.

Podobne przykłady można podać dla operacji dodawania liczebników cyfrowych w systemie liczbowym innym niż dziesiętny. Przeprowadźmy więc operację 22 + 22 w systemie trójkowym. W tym wypadku podczas wykonywania pisemnego dodawania obu liczebników cyfrowych dokonujemy fuzji struktur głębokich $\mathrm{P}[\mathbf{O}_3(\mathbf{2}), \mathbf{O}_1(\mathbf{2})] \oplus \mathrm{P}[\mathbf{O}_3(\mathbf{2}), \mathbf{O}_1(\mathbf{2})] = \mathrm{P}[\mathbf{O}_3(\mathbf{2}), \mathbf{O}_3(\mathbf{2}), \mathbf{O}_1(\mathbf{2}), \mathbf{O}_1(\mathbf{2})]$. Tymczasem $\mathbf{O_3}\ Kan\ \mathbf{O}_1$ oraz $\sim \mathbf{O}_3\ (>)\ +(\mathbf{2}{*}, \mathbf{2}{*})$ (skoro $\mathbf{2}{*} = \mathbf{O}_2$, to $+(\mathbf{2}{*}, \mathbf{2}{*}) = \mathbf{O}_4$), a więc na mocy (T31) $(\exists k)\ (\mathrm{P}[\mathbf{O}_3(\mathbf{2}), \mathbf{O}_3(\mathbf{2}), \mathbf{O}_1(\mathbf{2}), \mathbf{O}_1(\mathbf{2})] \vdash \mathrm{P}[\mathbf{O}_3(\mathbf{2}), \mathbf{O}_3(\mathbf{2}), \mathbf{O}_{1\times3}(\mathbf{1}), \mathbf{O}_1(Sq^k(\mathbf{0}))] \wedge Sq^k(\mathbf{0}){*} = D(\mathbf{O}_3, +(\mathbf{2}{*}, \mathbf{2}{*})))$. Skoro $D(\mathbf{O}_3, +(\mathbf{2}{*}, \mathbf{2}{*})) = D(\mathbf{O}_3, \mathbf{O}_4)$, to $D(\mathbf{O}_3, +(\mathbf{2}{*}, \mathbf{2}{*})) = \mathbf{O}_1$. Zatem $Sq^k(\mathbf{0}){*} = \mathbf{O_1}$ i w konsekwencji umysł wytwarza $k = 1$, czyli $\mathbf{O}_1(Sq^k(\mathbf{0})) = \mathbf{O}_1(\mathbf{1})$. Na następnym etapie obliczeniowym na mocy definicji operacji mnożenia funktorów pozycji składniowych (Df. ×), (AO6) oraz (AL3), umysł przeprowadza inferencję $\mathrm{P}[\mathbf{O}_3(\mathbf{2}), \mathbf{O}_3(\mathbf{2}), \mathbf{O}_{1\times3}(\mathbf{1}), \mathbf{O}_1(\mathbf{1})] \vdash \mathrm{P}[\mathbf{O}_3(\mathbf{2}), \mathbf{O}_3(\mathbf{2}), \mathbf{O}_3(\mathbf{1}), \mathbf{O}_1(\mathbf{1})]$. W kolejnej fazie na mocy (T31) dokonywana jest inferencja $\mathrm{P}[\mathbf{O}_3(\mathbf{2}), \mathbf{O}_3(\mathbf{2}), \mathbf{O}_3(\mathbf{1}), \mathbf{O}_1(\mathbf{1})] \vdash \mathrm{P}[\mathbf{O}_3(\mathbf{2}), \mathbf{O}_9(\mathbf{1}), \mathbf{O}_3(\mathbf{0}), \mathbf{O}_1(\mathbf{1})]$. Oto jej mechanizm:

Skoro zachodzi $\mathbf{O_3}$ *Kan* $\mathbf{O}_9$ oraz ~ $\mathbf{O}_3$ (>) +(**2***, **1***), to ($\exists k$) (P[$\mathbf{O}_3$(**2**), $\mathbf{O}_3$(**2**), $\mathbf{O}_3$(**1**), $\mathbf{O}_1$(**1**)] ⊢ P[$\mathbf{O}_3$(**2**), $\mathbf{O}_{3\times3}$(**1**), $\mathbf{O}_3(Sq^k(\mathbf{0}))$, $\mathbf{O}_1$(**1**)] ∧ $Sq^k(\mathbf{0})^* = D(\mathbf{O}_3, +(\mathbf{2}^*, \mathbf{1}^*))$. Skoro zaś $D(\mathbf{O}_3, +(\mathbf{2}^*, \mathbf{1}^*)) = D(\mathbf{O}_3, \mathbf{O}_3)$, to $Sq^k(\mathbf{0})^* = \mathbf{O}_0$, a zatem $k = 0$, a w konsekwencji $\mathbf{O}_3(Sq^k(\mathbf{0})) = \mathbf{O}_3(\mathbf{0})$. Podobnie, na mocy (Df. ×) oraz (AO6) umysł wytwarza $\mathbf{O}_{3\times3}(\mathbf{1}) = \mathbf{O}_9(\mathbf{1})$. Zgodnie z (AL3) przeprowadza inferencję P[$\mathbf{O}_3$(**2**), $\mathbf{O}_{3\times3}$(**1**), $\mathbf{O}_3$(**0**), $\mathbf{O}_1$(**1**)] ⊢ P[$\mathbf{O}_3$(**2**), $\mathbf{O}_9$(**1**), $\mathbf{O}_3$(**0**), $\mathbf{O}_1$(**1**)].

W ostatniej fazie za pomocą mechanizmów opisywanych w (AL6) i (AL2) umysł przeprowadza inferencję P[$\mathbf{O}_3$(**2**), $\mathbf{O}_9$(**1**), $\mathbf{O}_3$(**0**), $\mathbf{O}_1$(**1**)] ⊢ P[$\mathbf{O}_9$(**1**), $\mathbf{O}_3$(**2**), $\mathbf{O}_1$(**1**)]. Ostatecznie więc wyprowadza P[$\mathbf{O}_3$(**2**), $\mathbf{O}_1$(**2**)] ⊕ P[$\mathbf{O}_3$(**2**), $\mathbf{O}_1$(**2**)] ⊢ P[$\mathbf{O}_9$(**1**), $\mathbf{O}_3$(**2**), $\mathbf{O}_1$(**1**)]. W systemie trójkowym zachodzi więc: 22 + 22 = 121.

2.4.2. OSZACOWYWANIE RELACJI WIĘKSZOŚCI MIĘDZY LICZEBNIKAMI

Porównanie dwóch dowolnych liczebników pod względem większości wymaga ich zapisania w danym systemie zapisu złożonych liczebników cyfrowych. Umysł, który porównuje pod względem relacji większości dwa złożone liczebniki cyfrowe, postępuje zwykle według algorytmu:

(i) jeśli pierwszy liczebnik jest dłuższy od drugiego, to pierwszy jest większy od drugiego (i *vice versa*);

(ii) jeśli oba liczebniki mają tę samą długość, to jeśli pierwsza cyfra od lewej strony występująca w pierwszym liczebniku jest większa od pierwszej od lewej w drugim liczebniku, to pierwszy z porównywanych liczebników jest większy od drugiego;

(iii) jeśli początkowe cyfry są identyczne w porównywanych liczebnikach, to umysł porównuje z uwagi na relację większości kolejne cyfry występujące w porównywanych liczebnikach;

(iv) operacja ta trwa tak długo, aż umysł trafi na taką parę kolejnych cyfr w porównywanych liczebnikach, że jedna będzie większa od drugiej;

(v) jeśli umysł nie trafi na taką parę w swoich aktach porównywania, to porównywane liczebniki uzna za identyczne.

Opisana praktyka odzwierciedla wykonywaną przez umysł podczas rozmaitych zadań obliczeniowych operację porównywania struktur głębokich liczebników. Przedstawiona teoria jest w stanie opisać mechanizm tych operacji. Porównując dwie struktury głębokie, umysł musi zredukować je do kodenotacyjnych i wystandaryzowanych struktur głębokich w danym systemie kanonicznym funktorów pozycji składniowych. Są to struktury utworzone wyłącznie z funktorów pozycji wyznaczanych jako kanoniczne przez dany funktor określający system zapisu cyfrowego. Niech formuła o postaci $Stand(O_i, P[\alpha_1, \ldots, \alpha_k])$ oznacza, że funktor pozycji składniowej O_i

wyznacza to, że struktura głęboka $P[\alpha_1, \ldots, \alpha_k]$ jest strukturą standardową względem funktora O_i. Oto definicja relacji *Stand*:

(Df. *Kan*(O_i)) $Stand(O_i, P[\alpha_1, \ldots, \alpha_k]) \equiv_{df}$

(i) $(\forall i)\ [1 \leq i \leq k \rightarrow (\exists O_j, c)\ (\alpha_i = O_j(c) \wedge O_i\ Kan\ O_j \wedge O_i\ (>)\ c^*)] \wedge$

(ii) $(\forall i, j)\ [1 \leq i \leq k \wedge 1 \leq j \leq k \wedge j \neq i \rightarrow (\forall O_n, O_m, c_l, c_h)\ (\alpha_i = O_n(c_l) \wedge \alpha_j = O_m(c_h) \rightarrow O_n \neq O_m)] \wedge$

(iii) $[\alpha_1\ />/\ \alpha_2 \wedge \ldots \wedge \alpha_{k-1}\ />/\ \alpha_k] \wedge$

(iv) $(\forall i)\ [1 \leq i \leq k \rightarrow (\forall O_j)\ \alpha_i \neq O_i(\mathbf{0})]$

Dana struktura głęboka jest strukturą standardową względem danego funktora wtedy, gdy spełnione są cztery warunki:

(i) dana struktura głęboka jest zbudowana wyłącznie za pomocą funktorów kanonicznych (wraz z dowolnymi cyframi elementarnymi) względem danego funktora,

(ii) każdy funktor w danej strukturze głębokiej występuje w niej tylko jeden raz,

(iii) poszczególne składniki w danej strukturze głębokiej tworzą liniowy porządek z uwagi na relację większości zachodzącą między tymi składnikami,

(iv) żaden składnik standardowej struktury głębokiej nie jest kształtu $O_i(\mathbf{0})$, czyli nie jest zbudowany za pomocą cyfry elementarnej **0**.

Oto przykłady standardowych względem funktora $\mathbf{O}_{10}$ struktur głębokich: $P[\mathbf{O}_{1000}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{1}), \mathbf{O}_{10}(\mathbf{5}), \mathbf{O}_1(\mathbf{7})]$, $P[\mathbf{O}_{1000}(\mathbf{2}), \mathbf{O}_1(\mathbf{7})]$. Z kolei struktury o postaci (i) $P[\mathbf{O}_{1000}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{1}), \mathbf{O}_{10}(\mathbf{5}), \mathbf{O}_5(\mathbf{7})]$, (ii) $P[\mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_{10}(\mathbf{2}), \mathbf{O}_1(\mathbf{7})]$, (iii) $P[\mathbf{O}_{100}(\mathbf{2}), \mathbf{O}_{1000}(\mathbf{1}), \mathbf{O}_1(\mathbf{5}), \mathbf{O}_{10}(\mathbf{7})]$ oraz (iv) $P[\mathbf{O}_{1000}(\mathbf{2}), \mathbf{O}_{100}(\mathbf{0}), \mathbf{O}_{10}(\mathbf{0}), \mathbf{O}_1(\mathbf{7})]$ nie są standardowymi strukturami głębokimi względem funktora $\mathbf{O}_{10}$. Pierwsza z nich nie spełnia warunku pierwszego (ponieważ funktor $\mathbf{O}_5$ nie jest wyznaczony jako funktor kanoniczny przez $\mathbf{O}_{10}$), druga — drugiego, trzecia — trzeciego (jako że nie jest zachowany porządek określony relacją />/ między kolejnymi składnikami struktury), a czwarta — czwartego.

Definicja relacji większości dla standardowych struktur głębokich względem funktora pozycji składniowej O_j ma charakter warunkowy i przyjmuje postać:

(Df. >) $(\forall O_j)\ \{Stand(O_i, P[\alpha_1, \ldots, \alpha_k]) \wedge Stand(O_i, P[\beta_1, \ldots, \beta_n]) \rightarrow [P[\alpha_1, \ldots, \alpha_k] > P[\beta_1, \ldots, \beta_n] \equiv_{df} \alpha_1\ />/\ \beta_1 \vee \{[n > 1 \wedge k \geq n \rightarrow (\sim \beta_1\ />/\ \alpha_1 \wedge \alpha_2\ />/\ \beta_2) \vee \ldots \vee (\sim \beta_{n-1}\ />/\ \alpha_{n-1} \wedge \alpha_n\ />/\ \beta_n)] \wedge [k > 1 \wedge n > k \rightarrow (\sim \beta_1\ />/\ \alpha_1 \wedge \alpha_2\ />/\ \beta_2) \vee \ldots \vee (\sim \beta_{k-1}\ />/\ \alpha_{k-1} \wedge \alpha_k\ />/\ \beta_k)]\}\}$

Oto przykład mechanizmu obliczeniowego opisanego w (Df. >):

(1) Porównując złożony liczebnik cyfrowy „10022” z „10200” (oba rozumiane zgodnie z układem dziesiętnym), umysł koreluje oba liczebniki z ich standardowymi strukturami głębokimi P[$\mathbf{O}_{10000}(\mathbf{1})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(\mathbf{2})$], P[$\mathbf{O}_{10000}(\mathbf{1})$, $\mathbf{O}_{100}(\mathbf{2})$].

(2) Ponieważ nie zachodzi $\mathbf{O}_{10000}(\mathbf{1})$ />/ $\mathbf{O}_{10000}(\mathbf{1})$, umysł podejmuje następne czynności obliczeniowe, w których porównuje z uwagi na relację />/ kolejne składniki obu struktur głębokich. Ponieważ obie struktury głębokie są wieloskładnikowe, a pierwsza jest dłuższa od drugiej, umysł wybiera strategię porównywania składników analizowanych struktur odpowiadającą sytuacji opisanej w (Df. >) przez klauzulę: $n > 1 \wedge k \geq n$.

(3) Najpierw umysł formatuje zapis (~ $\mathbf{O}_{10000}(\mathbf{1})$ />/ $\mathbf{O}_{10000}(\mathbf{1})$ $\wedge$ $\mathbf{O}_{10}(\mathbf{2})$ />/ $\mathbf{O}_{100}(\mathbf{2})$), a następnie ocenia go jako fałsz. Ostatecznie uznaje za fałsz sformatowany zapis o postaci $\mathbf{O}_{10000}(\mathbf{1})$ />/ $\mathbf{O}_{10000}(\mathbf{1})$ $\vee$ (~ $\mathbf{O}_{10000}(\mathbf{1})$ />/ $\mathbf{O}_{10000}(\mathbf{1})$ $\wedge$ $\mathbf{O}_{10}(\mathbf{2})$ />/ $\mathbf{O}_{100}(\mathbf{2})$). Na tej podstawie wyprowadza ~ P[$\mathbf{O}_{10000}(\mathbf{1})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(\mathbf{2})$] > P[$\mathbf{O}_{10000}(\mathbf{1})$, $\mathbf{O}_{100}(\mathbf{2})$], czyli wnioskuje, że liczebnik „10022” nie jest większy od „10200”.

Z kolei porównując „10200” z „10022”, umysł przeprowadza obliczenia zgodnie z sytuacją wyznaczoną w (Df. >) przez klauzulę: $k > 1 \wedge n > k$. Mianowicie formatuje zapis $\mathbf{O}_{10000}(\mathbf{1})$ />/ $\mathbf{O}_{10000}(\mathbf{1})$ $\vee$ (~ $\mathbf{O}_{10000}(\mathbf{1})$ />/ $\mathbf{O}_{10000}(\mathbf{1})$ $\wedge$ $\mathbf{O}_{100}(\mathbf{2})$ />/ $\mathbf{O}_{10}(\mathbf{2})$). Następnie ocenia go jako prawdziwy i na tej podstawie wnioskuje, że P[$\mathbf{O}_{10000}(\mathbf{1})$, $\mathbf{O}_{100}(\mathbf{2})$] > P[$\mathbf{O}_{10000}(\mathbf{1})$, $\mathbf{O}_{10}(\mathbf{2})$, $\mathbf{O}_{1}(\mathbf{2})$].

Operacje obliczeniowe umysłu mające na celu oszacowanie, który z dwóch dowolnych liczebników jest większy, nie angażują więc teorii arytmetycznej. Jeśli relacja większości zachodząca między liczebnikami jest izomorficzna z relacją większości zachodzącą między liczbami denotowanymi przez liczebniki, to w celu oszacowania tego, która z dwóch liczb jest większa, umysł nie angażuje teorii arytmetycznej i jej aparatu dedukcyjnego. Umysł nie musi znać aksjomatów arytmetyki Peana, aby stwierdzić, że liczba denotowana przez liczebnik cyfrowy „12342” jest większa od liczby denotowanej przez „12242”. W tego rodzaju zadaniach umysł posługuje się logiką liczebników.

3. UWAGI KOŃCOWE

Dopełnieniem przedstawionej teorii powinna być konstrukcja modelu semantycznego dla LL. Za pomocą liczebników (cyfrowych i słownych) umysł odnosi się do rozmaitych obiektów liczbowych (ilości, liczności, wielkości). Czynność ta (referencja liczebnikowa) jest modelowana jako proces równoczesnej aktywacji (a) struktury głębokiej reprezentacji umysłowej liczebnika oraz (b) określonej reprezentacji semantycznej obiektu liczbowego, do którego umysł odnosi się za pomocą danego liczebnika. Poczucie, że wypowiadając *daj mi sześć jabłek*, odnosimy się do sześciu jabłek, wyjaśniane jest tym, że umysł aktywuje zarówno reprezentację umysłową wypowiedzi *daj mi sześć jabłek*, jak i reprezentację umysłową sześciu jabłek.

Składową tej drugiej reprezentacji jest reprezentacja semantyczna liczby sześć. Liczebnik *sześć* jest więc semantycznie powiązany z obiektem liczbowym sześć. To powiązanie jest wyznaczone przez odpowiednie reguły semantyczne korelujące struktury głębokie reprezentacji umysłowych liczebników z reprezentacjami obiektów liczbowych.

Celem semantyki liczebników jest więc między innymi udzielenie odpowiedzi na następujące pytania: w jaki sposób struktury głębokie liczebników denotują liczby (liczności, wielkości); co jest odpowiednikiem semantycznym cyfr elementarnych występujących w strukturach głębokich liczebników; co jest korelatem semantycznym funktorów pozycji składniowych w tych strukturach; co jest odpowiednikiem semantycznym funktora planu pozycyjnego; w jaki sposób operacja fuzji struktur głębokich liczebników odzwierciedla operację dodawania na liczbach naturalnych.

Część trzecia studium, przedstawiająca model semantyczny logiki liczebników, będzie próbą odpowiedzi na te pytania. Za pomocą struktur głębokich liczebników mają być odwzorowywane reprezentacje liczbowe, do których wytwarzania służą rozmaite operacje dokonywane na umysłowych osiach liczbowych (sumacyjnych, punktowo-miejscowych oraz dokładnych). Formalna teoria tych osi została przedstawiona w (Patro, Krysztofiak 2013), a w wersji rozbudowanej w (Krysztofiak 2015b). Okazuje się, że model semantyczny LL ma strukturę opisywaną przez arytmetykę indeksowanych liczb naturalnych, która stanowi podstawowe narzędzie kompetencji arytmetycznej umysłu wymaganej przy rozwiązywaniu prostych zadań matematycznych z treścią (zob. Krysztofiak 2010, Krysztofiak 2012a).

BIBLIOGRAFIA

Humberstone L. (2005), *Geach's Categorial Grammar*, „Linguistics and Philosophy" 28(3), 281-317.

Ifrah G. (2006), *Historia powszechna cyfr*, t. 1, Warszawa: Wydawnictwo W.A.B.

Krysztofiak W. (2010), *Multi-temporalne struktury obliczeniowe. Indeksowane liczby naturalne w świetle arytmetyki kognitywnej*, „Filozofia Nauki" 18(4) [72], 23-47.

Krysztofiak W. (2012a), *Indexed Natural Numbers in Mind. A Formal Model of the Basic Mature Number Competence*, „Axiomathes" 22(4), 433-456.

Krysztofiak W. (2012b), *Logiczna składnia liczebnika. Studium kognitywistyczne. Część I*, „Filozofia Nauki" 20(1) [77], 59-91.

Krysztofiak W. (2015a), *Representational Structures of Arithmetical Thinking: Part One*, „Axiomathes", DOI 10.1007/s10516-015-9271-1.

Krysztofiak W. (2015b), *Algebraic Models of Mental Number Axes: Part II*, „Axiomathes", DOI 10.1007/s10516-015-9270-2.

Patro K., Krysztofiak W. (2013), *Umysłowe osie liczbowe. Efekt SNARC. Aspekty filozoficzne*, „Filozofia Nauki" 21(3) [83], 45-98.

Pöppel E. (1989), *Granice świadomości. O rzeczywistości i doznawaniu świata*, Warszawa: Państwowy Instytut Wydawniczy.