Uniwersytet Warszawski
Wydział Matematyki, Informatyki i Mechaniki

Miron Szewczyk
Nr albumu: 383504

Tytuł pracy magisterskiej

Praca licencjacka
na kierunku INFORMATYKA

Praca wykonana pod kierunkiem
dr. hab. Filigrana Fifaka
Instytut Blabalii Fetorycznej

Warszawa, Maj 2017

Streszczenie

W pracy przedstawiono prototypową implementację blabalizatora różnicowego bazującą na teorii fetorów σ-ρ profesora Fifaka. Wykorzystanie teorii Fifaka daje wreszcie możliwość efektywnego wykonania blabalizy numerycznej. Fakt ten stanowi przełom technologiczny, którego konsekwencje trudno z góry przewidzieć.

Słowa kluczowe

blabaliza różnicowa, fetory σ-ρ, fooizm, blarbarucja, blaba, fetoryka, baleronik

Dziedzina pracy (kody wg programu Socrates-Erasmus)

Klasyfikacja tematyczna

D. Software
D.127. Blabalgorithms
D.127.6. Numerical blabalysis

Tytuł pracy w języku angielskim

???

Spis treści

Wprowadzenie

VASS jako model obliczeniowy słabszy od maszyny Turinga, osiągalność jest rozstrzygalna. Z-VASS jako jeszcze prostrzy model obliczeniowy. Pytanie o separowalność, znane wyniki.

Następnie mówię co zrobiłem, czyli podaje odpowiedź dla deterministycznych vassów, ciekawe przypadki separacji/braku separacji, przypadek jednowymiarowy.

Co więcej w problemie homespace wyszedł wczoraj trochę podobny podproblem.

Rozdział 1

Podstawowe pojęcia

1.1. Definicje

Głownym tematem pracy są języki rozpoznawane przez struktury nazywane Vector Addition System with States (VASS). O VASSach można intuicyjnie myśleć jak o automacie wyposażonym w stałą ilość liczników, każda tranzycja je modyfikuje lecz nigdy ich wartość nie może spaść poniżej zera. d–wymiarowym VASSem nazywamy krotkę $(Q, T, \Sigma, q_s, v_s, q_f, v_f)$ gdzie

- Q jest skończonym zbiorem stanów,
- $q_s \in Q$ oraz $v_s \in N^d$ stanem i wektorem początkowym,
- $q_f \in Q$ oraz $v_f \in N^d$ stanem i konfiguracją końcową
- $T \in Q \times Z^d \times \Sigma \times Q$ zbiorem tranzycji.

Konfiguracją VASSu nazywamy parę stanu i wektora $q(v)$ gdzie $q \in Q$ oraz $v \in N^d$.

Tranzycje (q, v, a, p) oznaczamy jako $q \xrightarrow[v]{a} p$. Wektor v nazywamy efektem tranzycji, a literą tranzycji, q/p stanem początkowym/końcowym.

Biegiem VASSu długości n z konfiguracji $q(v)$ do konfiguracji $p(u)$ nazywamy ciąg tranzycji i kofniguracji :

$$q_1(v_1) \xrightarrow[a_1]{u_1} q_2(v_2) \ldots \xrightarrow[a_{n-1}]{u_{n_1}} q_n(v_n)$$

gdzie sąsiednie stany początkowe i końcowe się zgadzają,

Słowo $a_1 \ldots a_n$ jest rozpoznawane przez VASS jeśli istnieje poprawny bieg postaci

$$q_s(v_s) \xrightarrow[a_1]{u_1} q_2(v_2) \ldots \xrightarrow[a_n]{u_n} q_f(v_f)$$

Przez język VASSu rozumiemy zbiór wszystkich słów rozpoznawanych przez VASS i oznaczamy $L(A)$ gdzie A to VASS.

Z-VASSy są strukturami bardzo podobnymi do VASSów. Jedyną różnicą jest to, że w biegach Z-VASSu wszystkie kofnigruację mogą przyjmować wartości z $\mathbb{Z}^d$ zamiast tylko z $\mathbb{N}^d$. Słowa rozpoznawane przez $\mathbb{Z}$-VASSy i języki $\mathbb{Z}$-VASSów rozumiemy analogicznie jak w przypadku VASSów. Mechanizm rozróżniający VASSy od Z-VASSów będziemy nazywali słabym zero testem.

Mówimy, że dwa VASSy/Z-VASSy są równoważne jeśli ich języki są sobie równe,

Jeśli mamy dwa rozłączne języki L_1 i L_2 to powiemy, że język L_3 je separuje lewostronnie jeśli $L_1 \subseteq L_3$ oraz $L_3 \cap L_2 = \emptyset$. Piszemy $L_1 |_{L_3} L_2$.

1.2. Deterministyczne VASSy

Wiele istotnych przykładów i wyników tej pracy opiera się na deterministycznych VASSach, które są szczególnym przypadkiem VASSów. W tej pracy, pod pojęciem VASSu deterministycznego rozumiemy VASS w którym dla każdego stanu i litery istnieje co najwyżej jedna tranzycja wychodząca z tego stanu po tej literze. Dzięki determinizmowi możemy traktować VASS jako funkcję z konfiguracji i słowa w konfigurację lub $\perp$. Jeśli VASS A znajdujący się w konfiguracji $p(v)$ po przeczytaniu słowa w znajdzie się w konfiguracji $q(u)$ to piszemy $A(p(v), w) = q(u)$.

VASSy deterministyczne mają istotnie słabszą moc wyrazu niż ogólne VASSy, istnieją języki rozpoznawane tylko przez niedeterministyczne VASSy.

Theorem 1.2.1 *Istnieją VASSy, dla których nie istnieje równoważny deterministyczny VASS.*

Proof 1.2.2 *Weźmy VASS A nad alfabetem $\Sigma = \{a, b\}$ który rozpoznaje język*

$$L = \{a^n b \Sigma^* a^n \mid n \in \mathbb{N}\}$$

. Załózmy, że istnieje deterministyczny VASS $B = (Q, T, \Sigma, q_s, v_s, q_f, v_f)$ który rozpoznaje L. Weźmy $n = |A|$ i słowa

$$w_1 = a^n b a^n,\ w_2 = a^n b a^n b a^n$$

Ponieważ $w_1, w_2 \in L$ to

$$A(q_s(v_s), w_1) = q_f(v_f),\ A(q_s(v_s), w_2) = q_f(v_f)$$

Dodatkowo, ponieważ $w_2 = w_1 b a^n$ to

$$A(q_s(v_s), w_2) = A(q_s(v_s), w_1 b a^n) = A(q_f(v_f), b a^n) = q_f(v_f)$$

Weźmy teraz słowa $w_3 = a^{2n} b a^{2n}$ oraz $w_4 = w_3 b a^n$ $w_3 \in L$ a więc $A(q_s(v_s), w_3) = q_f(v_f)$, natomiast $w_4 \notin L$ Jednak, ponieważ $w_4 = w_3 b a^n$ to $A(q_s(v_s), w_4) = A(q_f(v_f), b a^n) = q_f(v_f)$ czyli $w_4 \in L$ co daje sprzeczność z definicją języka L. Czyli taki B nie może istnieć.

Jedną z zalet rozważania deterministycznych VASSów jest łatwość konstruowania ich dopełnień. Problem konstruowania dopełnień, nierostrzygalny w ogólnym przypadku, w przypadku deterministycznym ma proste rozwiązanie.

Lemma 1.2.3 *Mając dany deterministyczny VASS A można skonstruować VASS B taki, że*

$$w \in L(A) \equiv w \notin L(B)$$

Proof 1.2.4 *Ustalmy $A = (Q, T, \Sigma, q_s, v_s, q_f, v_f)$. Dla każdego $w \in \Sigma^*$ chcemy wykryć czy w nie ma poprawnego biegu w VASSie A. Dzięki determinizmowi A, po przeczytaniu każdej kolejnej litery A może znajdować się w co najwyżej jednej konfiguracji. Nie istnieje poprawny bieg w po A jeśli zachodzi jeden z trzech warunków:*

1. *Po przeczytaniu całego słowa A dojdzie do konfiguracji innej niż $q_f(v_f)$*
2. *Po przeczytaniu pewnego prefiksu słowa w, A dojdzie do stanu z którego nie ma tranzycji wychodzącej po kolejnej literze w*

3. *Po przeczytaniu pewnego prefiksu słowa w, A dojdzie do konfiguracji z której istnieje tranzycja wychodząca po kolejnej literze, lecz skorzystanie z niej spowodowałoby spadnięcie licznika poniżej zera.*

Skonstruujemy VASS B który na początku niedeterministycznie zgaduje, który z trzech warunków sprawia że słowo wejściowe w nie posiada biegu akceptującego w A. Następnie symuluje bieg w po VASSie A do momentu aż niedeterministycznie zgadnie, że ów warunek zaszedł. Sprawdzenie każdego warunku jest kwestią prostego VASSowego gadżetu.

Dowód ten istotnie korzysta z determinizmu VASSu A. Gdyby A był niedeterministyczny, to B musiałby symulować jednocześnie każdy możliwy bieg słowa w po A. Taka konstrukcja jest wykonalna dla automatów, gdzie musimy śledzić tylko wszystkie możliwe stany, lecz jest niemożliwa dla VASSów, ponieważ ilość konfiguracji w których możemy się znaleźć rośnie wykładniczo ze względu na długość słowa. Modelem obliczeniowym który pozwala na symulowanie takiej sytuacji są Upward Closed Transistion Systems TODO!!

Z lematu 1.2.1 wiemy, że istnieje VASSy niedeterminizowalne. Okazuje się, że nie tylko istnieją ale też nie da się rozpoznać, czy dany VASS jest determinizowalny.

Theorem 1.2.5 *Następujący problem jest nierostrzygalny:*

DETERMINIZACJA VASSU

Task: *VASS A*

Problem: *Czy istnieje deterministyczny VASS B, taki że $L(A) = L(B)$*

Proof 1.2.6 *Dowód wykorzystuje techniki podobne jak dowód lematu 1.2.3 o braniu dopełnień. Ustalmy T, maszynę z licznikami i zerotestami, która ma co najwyżej jeden poprawny bieg. Jak dobrze wiadomo, problem stopu jest nierostrzygalny dla maszyn Turinga, które są równoważne maszynom z licznikami i zerotestami. Dowolny bieg maszyny z zerotestami można zakodować jako ciąg krotek (stan maszyny, zawartość liczników) i tranzycji.*

VASSy są zbyt słabym modelem obliczeniowym by móc zweryfikować czy dany zapis jest poprawnym biegiem. Ponieważ problem osiągalności konfiguracji jest rozstrzygalny dla VASSów to, to gdyby były dość silne by weryfikować poprawność zapisu to przy ich pomocy moglibyśmy rozwiązać problem stopu dla maszyn z zerotestami i maszyn Turinga.

Są natomiast wystarczające silne by rozpoznać niepoprawne biegi. To znaczy, dla danej maszyny z zerotestami można łatwo skonstruować VASS który rozpoznaje niepoprawne zapisy biegów tej maszyny. Zapis może być niepoprawny z powodów czysto składniowych, liczniki w sąsiednich konfiguracjach mogą się nie zgadzać lub zerotest może być niepoprawnie wykonany. Każdy z tych powodów może być wykryty przez VASS.

Niech A będzie VASSem, takim że $L(A) = \{$ niepoprawne zapisy biegów maszyny $T\}$. Załóżmy, że potrafimy zdeterminizować A. Niech B będzie deterministycznym VASSem, takim że $L(B) = L(A)$. Ponieważ B jest deterministyczne to zgodnie z TODO potrafimy skonstruować VASS C taki, że $L(C) = \overline{L(B)}$. Czyli C akceptuje dokładnie poprawne biegi maszyny T. Sprawdzając czy C posiada jakikolwiek akceptujący bieg (co można zrobić w Ackermannowym czasie TODO cytowanie) możemy rozwiązać problem stopu dla maszyny T, co jest oczywistą sprzecznością.

Czyli nie może istnieć algorytm który znajduje determinizację VASSów. Warto zauważyć, że to twierdzenie nie implikuje twierdzenia 1.2.1, teoretycznie mogą istnieć VASSy dla których równoważny deterministyczny VASS istnieje ale jest nie obliczalny.

Rozdział 2

Kontekst

Separacja języków jest jednym z klasycznych, szeroko badanych problemów informatyki teoretycznej. W ogólnej formie można go sformułować następująco:

PROBLEM SEPARACJI

Task: Języki L_1, L_2 i klasa języków $\mathbb{B}$.
Problem: Czy istnieje język $L_3 \in \mathbb{B}$ taki, że $L_1 |_{L_3} L_2$.

Problem ten jest rozważany dla różnych klas języków L_1, L_2 i różnych klas $\mathbb{B}$. W tym rozdziale przybliżę kilka interesujących wyników i podam intuicję dlaczego metody zastosowane w nich nie poskutkowały w przypadku separacji języków VASSów przez języki $\mathbb{Z}$-VASSów

2.1. Separacja języków 1-VASSów za pomocą języków regularnych

https://arxiv.org/abs/1701.02808

W pracy R̈egular Separability of One Counter AutomataẄojciech Czerwiński i Sławomir Lasota zbadali między innymi problem separacji języków 1-VASSów. Pokazali, że problem ten jest rozstrzygalny w czasie PSPACE.

Definiton 2.1.1 *Niech $A = TODO$ będzie 1-VASSem. n-aproksymantem A nazwiemy automat A_n który symuluje A dokładnie dla małych wartości licznika i z pewną dokładnością dla dużych.*

- *Zbiór stanów*

2.2. Separacja VASSów z akceptacją przez stan za pomocą języków regularnych

JEst równoważna pustości przecięcia https://arxiv.org/abs/1702.05334

https://arxiv.org/pdf/1701.02808.pdf Tutaj jest o separacji 1-VASSów przez aproksymanty

Rozdział 3

Przypadki separacji

Rozważania na problemem, zacznę od pokazania kilku istotnych przypadków łatwej separacji. Dają one istotną intuicję jak patrzeć na problem oraz na możliwości i ograniczenia VASSów/Z-VASSów.

3.1. Separacja deterministycznych VASSów wykorzystując dopełnienia języków

Niech B będzie dowolnym determininistycznym VASSem. Wtedy istnieje Z-VASS Z, taki że:

$$\forall_A L(A) \cap L(B) = \emptyset \Rightarrow L(A) \mid_{L(Z)} L(B)$$

Takim Z-VASSem jest Z-VASS rozpoznający dopełnienie B, tak jak w lemacie TODO.

Przypadek ten jest ciekaw z kilku względów. Po pierwsze, problem separacji nie zależy w żaden sposób od lewego VASSu. Po drugie, pokazuje że problem jest trywialny dla względnie szerokiej grupy przypadków. By znaleźć przypadek braku separacji, prawy VASS musi istotnie korzystać z niedeterminizmu.

3.2. Niedeterministyczny przypadek

Poprzedni przypadek separacji polegał na determiniźmie prawego VASSu. Co więcej, separujący Z-VASS zależał tylko i wyłącznie od prawego VASSu, był wspólny dla wszystkich możliwych lewych VASSów. Dlatego warto wskazać istotnie niedeterministyczny VASS, który można odseparować lewostronnie od każdego rozłącznego VASSu.

Ustalmy alfabet $\Sigma = \{a, b\}$. Słowa nad tym alfabetem możemy zapisać jako $a^{n_1}ba^{n_2}....ba^{n_k}b$. Niech B będzie VASSem który rozpoznaje słowa w których conajmniej dwa bloki a mają równą długość, czyli $\exists_{i,j} i \neq j \wedge n^i = n^j$. B po przeczytaniu litery b może niedeterministycznie zacząć zliczać długość obecnego bloku inkrementując licznik aż przeczyta następną literę b, niejako zgadując, że to jest jeden z bloków o równej długości. Następnie w identyczny sposób VASS niedeterministycznie zaczyna dekrementować ten sam licznik, zliczając długość kolejnego bloku. Jeśli długości bloków są sobie równe, to licznik po przeczytaniu słowa będzie równy zero.

Języki rozłączne z $L(A)$ składają się z słów zawierających tylko bloki różnej długości.

Język VASSa ma pewne ciekawe właściwości, które pozwalają go odseparować od każdego innego języka 1-VASSa.

Lemma 3.2.1 *Niech A będzie 1-VASSem. Jeśli $L(A) \cap L(B) = \emptyset$ to istnieje liczba $n \in \mathbb{N}$ taka że :*

$$\forall_{w \in L(A)} \#_a(w) < n \#_b(w) < n$$

Innymi słowy, języki rozłączne z $L(B)$ muszą mieć ograniczenie na ilość

Proof 3.2.2 *Ustalmy A i załóżmy, że teza nie zachodzi. Niech n będzie równe $3 * |A|$. Weźmy słowo $w \in L(A)$, takie że $\#_a(w) > n \wedge \#_b(w) > n$. Pokażemy, że możemy tak napompować słowo w by było ciągle akceptowane przez A i zawierało dwa bloki liter a tej samej długości.*

Ustalmy teraz bieg VASSu A po słowie w i przyjrzyjmy się licznikowi tego VASSu podczas biegu. W szczególności przyjrzyjmy się konfiguracjom VASSu po przeczytaniu każdej litery b, nazwijmy je $q_1(v_1), \ldots q_k(v_k)$. Istnieje podciąg tych konfiguracji długości co najmniej XX taki, że wszystkie stany w nim są sobie równe, nazwijmy go $q(u_1), \ldots, q(u_m)$, będziemy na nie mówić konfiguracje wyróżnione. Podzielmy teraz słowo w na trzy podsłowa w_1, w_2, w_3 tak by $w = w_1 w_2 w_3$ oraz by każde z tych podsłów kończyło się na literę b i zawierało co najmniej $\lfloor \frac{m}{3} \rfloor$ liter odpowiadających konfigruacjom wyróżnionym. Czyli w pewnym sensie dzielimy słowo w na trzy "równe"podsłowa.

Reszta dowodu dzieli się na dwa przypadki:

1. *Podczas biegu po słowie w licznik powtórzy swoją wartość w dwóch konfiguracjach wyróżnionych. Niech $w = w_4 w_5 w_6$ gdzie powatarzające się wyróżnione konfiguracje występują na początkach słów w_5 i w_6. W takim wypadku, możemy dowolnie wiele razy napompować podsłowo w_6, czyli słowa $w_4 w_5^* w_6$ też są akceptowane przez VASS A, chociaż zawierają dowolnie dużo bloków liter a o tej samej długości.*

2. *Sytuacja z pierwszego przypadku nie ma miejsca, każda wyróżniona konfiguracja ma inną wartość licznika. Nie możemy zastosować wtedy tak prostego pompowania jak w pierwszym przypadku.*

   Przyjrzyjmy się wartościom licznika w wyróżnionych konfiguracjach, występujących podczas czytania w_2, środkowego podsłowa. Ponieważ wartości licznika w wyróżnionych konfiguracjach się nie powtarzają, muszą one osiągać duże wartości. Dokładniej, musi istnieć taka której licznik wynosi co najmniej XXXXX.

3. *TODO TODOD TODO TODO Załózmy, że znajdziemy dwie takie $q(u_i), q(u_j)$ gdzie $i < j, u_i < u_j$. Jeżeli chcemy pompować podsłowo im odpowiadające*

Rozdział 4

Przypadki braku separacji

Pierwszym pytaniem które należy zadać badając problem separacji, jest pytanie o przypadek braku separacji. W niektórych przypadkach każdą parę rozłącznych języków można odseparować. Przykładowo, jak dowiedli XXXX i XXXX w pracy XXXX, nie istnieje para języków VASSów z warunkiem akceptacji przez pokrywalność, która by była nieseparowalna przez język regularny.

4.1. Przypadek deterministyczny

W XXXX pokazaliśmy, że jeżeli prawy VASS jest deterministyczny to da się go odseparować od każdego innego VASSu. Odwracając sytuację i używając niemal identycznych argumentów jak przy dowodzie XXXX możemy skonstruowaćpierwszy przypadek pary nieseparowalnych VASSów.

Niech A będzie deterministycznym VASSem który nie jest równoważny żadnemu Z-VASSowi. Niech B będzie VASSem rozpoznającym dokładnie dopełnienie A, dzięki XXXXX wiemy jak go skonstruować. Wtedy, nie istnieje Z-VASS Z taki że $A \mid_Z B$. Gdyby istniał, to ponieważ musi zawierać A i mieć puste przecięcie z B, musiałby by być równy dokładnie A. Co jest sprzeczne z założeniami o A. Jako przykładowy A można wziąć język poprawnych nawiasowań.

Ten przypadek

4.2. Przerwa między językami

4.3. Przeciwobraz języka nawiasowań

Wygodnymi narzędziami do badania problemu separacji są języki poprawnych nawiasowań oraz transducery. Narzędzia te pozwalają sprowadzić problem do pojedynczego, kanonicznego przypadku który łatwo rozwiązać a rozwiązanie przenieść na przypadek ogólny. Dodatkowo, dają ciekawy wgląd w problem istnienia Z-VASSu równoważnego danemu VASSowi.

Definiton 4.3.1 *Językiem n-nawiasowań $N_n \subset \{a_1, b_1, \ldots, a_n, b_n\}$ nazwiemy język w którym*

$$\forall_{w \in N_n} \forall_i \forall_{s\, prefiks\, w} \#_{a_i}(s) \geq \#_{b_i}(s)$$
$$\forall_{w \in N_n} \forall_i \#_{a_i}(s) = \#_{b_i}(s)$$

Innymi słowy, jak ustalimy słowo $w \in N_n$ oraz liczbę i to podsłowo w składające się wyłącznie z liter a_i, b_i musi definiować poprawne nawiasowanie.

Definiton 4.3.2 *Transducerem nazwiemy automat, gdzie każda tranzycja jest dodatkowo etykietowana literą/literami wyjściowego alfabetu. Jeśli wejściowy alfabet to Σ a wyjściowy to Γ to transducer T definiuje funkcje :*

$$T : \Sigma^* \to \Gamma^*$$

Lemma 4.3.3 *Jeśli A to VASS/Z-VASS a T to transducer to język $T(L(A))$ jest rozpoznawany przez VASS/Z-VASS*

Lemma 4.3.4 *Języki VASSów są zamknięte na branie przeciwobrazów transducreów. Jeżeli B jest VASSem a T transducerem to istnieje A, taki że*

$$L(B) = T^{-1}(L(A))$$

Lemma 4.3.5 *Jeśli A jest n wymiarowym VASSem to istnieje transducer T i język n-nawiasowań N_n, taki że*

$$L(A) = T(N_n)$$

Lemma 4.3.6 *Dany jest VASS A. Jeśli A jest Z-VASSem to nie istnieje transducer T, taki że $T(L(A)) = N_n$. Ujmując to inaczej, jeśli istnieje T, które przenosi L(A) na N_n to A nie ma równoważnego Z-VASSu.*

Jest to ciekawa własność transducerów i języków nawiasowań. Korzystając z niej możemy zdefiniować istotną rodzinę przypadków braku separacji.

Theorem 4.3.7 *bracket-language Ustalmy n-wymiarowy VASS A który nie jest równoważny żadnemu Z-VASSowi. Jeśli istnieje transducer T, taki że $T(L(A)) = N_n$ to istnieje VASS B którego języka nie da się odseparować od języka A.*

Proof 4.3.8 *Ustalmy n-wymiarowy VASS A i transducer T, $T(L(A)) = N_n$. Weźmy dopełnienie języka N_n, nazwijmy je $\overline{N_n}$, $w \in \overline{N_n} \equiv w \notin N_n$. Łatwo skonstruować taki VASS C, że $L(C) = N_n$. Zgodnie z (TODO odwołanie do lematu), można skonstruować VASS B który rozpoznaje dokładnie przeciw obraz języka $\overline{N_n}$ w T, $L(B) = T^{-1}(\overline{N_n})$. $L(A) \cap L(B) = \emptyset$ ponieważ nic z L(A) nie może być w przeciwobrazie $\overline{N_n}$ względem F skoro obrazem L(A) jest N_n.*

Załóżmy, że istnieje Z-VASS Z, taki żę $L(A) \mid_{L(Z)} L(B)$ i przyjrzyjmy się $F(L(Z))$. Ponieważ $L(A) \subset L(Z)$ to $F(L(A)) = N_n \subset F(Z)$, ale skoro $L(Z) \cap L(B) = \emptyset$ to L(Z) musi być dokładnie równe N_n. Czyli mamy Z-VASS Z i transducer T, takie że $T(L(Z)) = N_n$, czyli potrafimy przedstawić język nawiasowań jako obraz języka Z-VASSu wględem transducera co jest sprzecznością.

Czyli Z-VASS Z nie może istnieć.

Intuicyjnie, twierdzenie **??** mówi, że o ile VASS korzysta choć trochę z właściwości zero testu, to istnieje VASS którego od niego nie odseparujemy. Osobnym problemem pozostaje jak wykryć czy dany język można przekstszatłcić na język nawiasowań.

4.4. Rozpoznawanie czy VASS jest Z-VASSem

Istotnym problemem w kontekście tej pracy jest następujący problem:

ISTNIENIE RÓWNOWAŻNEGO Z-VASSU

Task: VASS A

Problem: Czy istnieje Z-VASS B równoważny VASSowi A, czyli $L(A) = L(B)$

Ustalmy deteterministczny 1-VASS A. Patrząc na tranzycje miedzy stanami znajdźmy wszystkie silne spójne składowe. Cały VASS możemy przedstawić jako skierowany, acykliczny graf silnych spójnych składowych (DAG - directed acyclic graph). Kązdy DAG możemy przedstawić jako skończoną sumę skończonych ścieżek. Czyli VASS A jest równoważny sumie VASSów $A_1, \ldots A_n$ gdzie każdy VASS A_i odpowiada za pojedynczą ścieżkę pokrywającą DAG spójnych składowych. Dzięki determinizmowi A, każdemu słowu w odpowiada co najwyżej jeden A_i taki że $w \in L(A_i)$ Jeśli każdy A_i jest równoważny pewnemu B_i to istnieje B równoważne A. Jeśli któryś A_i nie jest równoważny Z-VASSowi to znajdzimy świadczące o tym przykłady (korzystające z mechanizmów pompowania) które zaświadczą też o tym, że A nie jest równoważne Z-VASSowi. Czyli A jest równoważne Z-VASSowi wtedy i tylko wtedy gdy każdy A_i jest równoważny Z-VASSowi. Przedstawię i udowodnię teraz algorytm sprawdzenia czy jeden A_i jest równoważny pewnemu Z-VASSowi oraz jego konstrukcję.

Theorem 4.4.1 *Niech A będzie deterministycznym 1-VASSem, którego silne spójne składowe tworzą ścieżkę. Istnieje Z-VASS B, równoważny A wtedy i tylko wtedy gdy w żadnej silnej spójnej składowej nie istnieje para cykli o wspólnym stanie, takich że jeden cykl ma efekt ściśle dodatni a drugi ściśle ujemny.*

Intuicyjnie, VASS jest istotnie VASSem (czyli nie istnieje równoważny Z-VASS) jeśli znajdziemy fragment który korzysta z słabego zerotestu.

Proof 4.4.2 *Niech A będzie deterministycznym 1-VASSem, którego silne spójne składowe tworzą ścieżkę i żadna spójna składowa nie zawiera dwóch cykli takich jak w sformułowaniu twierdzenia.*

Czyli A ma następującą postać

$$S_1 \to S_2 \to \cdots \to S_n$$

gdzie $S_1, \ldots, S_n$ są silnie spójnymi składowymi. Dzięki właściwościom A z założeń, w każdej silnie spójnej wszystkie cykle mają niedodatni albo nieujemny efekt. Nazwijmy silnie spójne składowe dodatnimi i ujemnymi.

Weźmy teraz nieujemną silnie spójną składową S_i. Dzięki jej nieujemności, istnieje $k \in \mathbb{N}$ które jest w pewnym sensie ograniczeniem na dekrementację w ramach tej silnej spójnej składowej. Ściślej mówiąc

$$\forall_{q \in Q_{S_i}} \forall_{n \in \mathbb{N}} \forall_w S_i(q(n), w) - n \geq k$$

Czyli możemy zastąpić silnie spójnie składową S_i przez nowy "gadżet" który jest kopią S_i która dodatkowo na początek "konsumuje" zawartość licznika i zapisują ją w stanie, przy czym nie konsumuje więcej niż k. Następnie symuluje działanie S_i operując na wartości zapisanej w stanie zamiast na liczniku, przy czym nie pozwala by wartość licznika zapisana w stanie nie *spadła poniżej zera. Jeśli wartość licznika zapisana w stanie przekroczy wartość k to jest gwarancja, że wewnątrz tej spójnej składowej wartość licznika nie może już spaść poniżej zera. Tak więc można "skonsumować" wartość licznika zapisaną w stanie, zinkrementować "normalny" licznik i dalej operować już tylko na nim nie przejmując się słabymi zerotestami. Podobnie wychodząc z silnie spójnej składowej należy 'skonsumować" wartość licznika zapisaną w stanie i zinkrementować "normalny" licznik.*

Przypadek z niedodatnimi silnymi spójnymi składowymi jest trochę trudniejszy. Dzięki jej niedodatności, istnieje $k \in \mathbb{N}$ które jest w pewnym sensie ograniczeniem na inkrementację w ramach tej silnej spójnej składowej. Ściślej mówiąc

$$\forall_{q \in Q_{S_i}} \forall_{n \in \mathbb{N}} \forall_w S_i(q(n), w) - n \leq k$$

W stanie będziemy trzymali informację "o ile obecna wartość licznika jest większa od najmniejszej wartości licznika wewnątrz obecnej silnej spójnej składowej". Dzięki ograniczeniu na k, wartość ta może być tylko z przedziału
k

. Korzystając z tej informacji oraz z wartości na "normalnym" liczniku możemy przy wyjściu z silnie spójnej składowej sprawdzić, czy najmniejsza wartość na biegu wewnątrz obecnej silnie spójnej składowej była większa od zera. W tym celu, najpierw dekrementujemy wartość na liczniku o wartość zapisaną w stanie, a następnie sprawdzamy czy możemy osiągnąć wartość zero

NIESKOŃCZONE

4.5. Reszta rozdziału, sekcja dla wizualnego rozdzielenia

Sprawdzenie czy zachodzi separacja wymaga sprawdzenia czy lewy VASS jest Z-VASSem.

Pokazuję algorytm i konstrukcję sprawdzenia czy det 1-VASS jest równoważny Z-VASSowi (konstrukcja poprzez patrzenie na DAG silnych spójnych składowych)

Pokazuję, że jeśli nie jest to mogę skonstruować transducer przenoszący go na język nawiasowań a więc i skonstruować nieseparowalny język.

Rozdział 5

Podsumowanie

1. Pokazałem przypadki separacji i braku separacji, dużo zrobione w kontekście deterministycznych.
2. Podałem ogólną (przy założeniu istnienia trnasducera) konstrukcję języka nieseparowalnego.
3. Stawiam hipotezę, że albo taki trnasducer istnieje albo język jest równoważny Z-VASSowi.
4. Celem dalszych badań powinien być ogólny algorytm sprawdzający separację, którego tu mi się nie udało znaleźć.
5. Dodatkowymi celami które wynikły z tej pracy jest problem "Czy dany VASS jest istotnie VASSem", sugerowany kierunek to udowodnie tezy z trnasducerami.

Bibliografia

[Bea65] Juliusz Beaman, *Morbidity of the Jolly function*, Mathematica Absurdica, 117 (1965) 338–9.

[Blar16] Elizjusz Blarbarucki, *O pewnych aspektach pewnych aspektów*, Astrolog Polski, Zeszyt 16, Warszawa 1916.

[Fif00] Filigran Fifak, Gizbert Gryzogrzechotalski, *O blabalii fetorycznej*, Materiały Konferencji Euroblabal 2000.

[Fif01] Filigran Fifak, *O fetorach σ-ρ*, Acta Fetorica, 2001.

[Głomb04] Gryzybór Głombaski, *Parazytonikacja blabiczna fetorów — nowa teoria wszystkiego*, Warszawa 1904.

[Hopp96] Claude Hopper, *On some Π-hedral surfaces in quasi-quasi space*, Omnius University Press, 1996.

[Leuk00] Lechoslav Leukocyt, *Oval mappings ab ovo*, Materiały Białostockiej Konferencji Hodowców Drobiu, 2000.

[Rozk93] Josip A. Rozkosza, *O pewnych własnościach pewnych funkcji*, Północnopomorski Dziennik Matematyczny 63491 (1993).

[Spy59] Mrowclaw Spyrpt, *A matrix is a matrix is a matrix*, Mat. Zburp., 91 (1959) 28–35.

[Sri64] Rajagopalachari Sriniswamiramanathan, *Some expansions on the Flausgloten Theorem on locally congested lutches*, J. Math. Soc., North Bombay, 13 (1964) 72–6.

[Whi25] Alfred N. Whitehead, Bertrand Russell, *Principia Mathematica*, Cambridge University Press, 1925.

[Zen69] Zenon Zenon, *Użyteczne heurystyki w blabalizie*, Młody Technik, nr 11, 1969.