Konstrukcja kompilatorów

Lista zadań nr 2

Na zajęcia 21 października 2020

UWAGA! W trakcie prezentacji rozwiązań należy zdefiniować i wyjaśnić pojęcia, które zostały oznaczone **wytłuszczoną** czcionką.

Zadanie 1. Skonstruuj automat niedeterministyczny rozpoznający język $(aba)^+ + ab^*a + a + b$ a następnie go zdeterminizuj.

Zadanie 2. Napisz specyfikację języka z poprzedniego zadania dla ocamllex. Dodaj **akcje semantyczne** (np. wypisywania odpowiednich tokenów na wyjściu). Skompiluj tę specyfikację używając opcji `-ml` (czyli poleceniem podobnym do `$ ocamlex -ml zad2.mll`) i odszukaj w otrzymanym pliku (w naszym przykładzie będzie to plik `zad2.ml`) strukturę automatu podobną do automatu z poprzedniego zadania. Pokaż jak reprezentowane są stany końcowe automatu. Wytłumacz, jak otrzymany lekser radzi sobie z podziałem na leksemy (jak rozpoznaje początek i koniec leksemu). Pokaż, dlaczego słowo aba jest rozpoznawane jako $(aba)^+$ a nie jako ab^*a. Zademonstruj działanie swojego leksera na kilku przykładach.
Wskazówka: w starszych wersjach programu ocamllex ten automat może być bardziej widoczny niż w nowszych

Zadanie 3. *Język zbalansowanych nawiasów* to język generowany przez gramatykę $S \to (S) \mid SS \mid \varepsilon$. Uzasadnij, że gramatyka $S \to (S)S \mid \varepsilon$ generuje ten sam język i jest **jednoznaczna**.

Zadanie 4. *Język zbalansowanych nawiasów okrągłych i kwadratowych* jest generowany gramatyką

$$S \to (S) \mid [S] \mid SS \mid \varepsilon.$$

Skonstruuj jednoznaczną gramatykę generującą ten język.

Zadanie 5. Rozważmy gramatykę wyrażeń arytmetycznych z dodawaniem, odejmowaniem, mnożeniem i potęgowaniem:

$$E \to id \mid (E) \mid E + E \mid E - E \mid E * E \mid E\,\hat{}\,E.$$

Skonstruuj równoważną jednoznaczną gramatykę, zachowując naturalną łączność i priorytety operatorów (potęgowanie wiąże w prawo i ma wyższy priorytet niż mnożenie).

W zadaniach poniżej $\xrightarrow{*}$ oznacza przechodnie i zwrotne domknięcie relacji $\longrightarrow$, natomiast $\xrightarrow{+}$ oznacza przechodnie domknięcie relacji $\longrightarrow$.

Zadanie 6. Podaj algorytm, który usuwa z gramatyki wszystkie produkcje zawierające zbędne symbole. Symbol X jest zbędny, jeśli w gramatyce nie da się skonstruować wyprowadzenia $S \xrightarrow{*} \alpha X \beta \xrightarrow{*} w$, gdzie S jest symbolem startowym, α i β są słowami mogącymi zawierać zarówno symbole terminalne jak i nieterminalne, a w słowem terminalnym. Zastosuj swój algorytm do gramatyki:

$$\begin{array}{lcl} S & \to & 0 \mid A \\ A & \to & AB \\ B & \to & 1 \\ C & \to & 2 \end{array}$$

Zadanie 7. W poniższej definicji S oznacza symbol startowy gramatyki, Y jest dowolnym symbolem nieterminalnym, u, x, y są słowami terminalnymi, a α, β i γ są słowami mogącymi zawierać zarówno symbole terminalne jak i nieterminalne. Notacja $k : x$ oznacza prefiks długości k słowa x.

Mówimy, że gramatyka bezkontekstowa jest LL(k) jeśli dla każdych dwóch takich **lewostronnych wyprowadzeń** $S \xrightarrow{*} uY\alpha \longrightarrow u\beta\alpha \xrightarrow{*} ux$ oraz $S \xrightarrow{*} uY\alpha \longrightarrow u\gamma\alpha \xrightarrow{*} uy$, że $k : x = k : y$ mamy $\beta = \gamma$ (czyli produkcja $Y \to \beta$ jest jednoznacznie wyznaczona przez prefiks długości k).
Sprawdź, czy następujące gramatyki należą do klasy LL(1):

$$\begin{array}{rcl} S & \to & L \mid a \\ L & \to & Lb \mid b \end{array}$$

$$\begin{array}{rcl} S & \to & Ab \mid cAB \\ A & \to & \epsilon \mid aA \\ B & \to & cA \mid bB \mid d \end{array}$$

Zadanie 8. Powiemy, że gramatyka bez zbędnych symboli jest *lewostronnie rekurencyjna* jeśli istnieje w niej symbol nieterminalny X oraz wyprowadzenie $X \xrightarrow{+} X\alpha$. Uzasadnij, że żadna gramatyka lewostronnie rekurencyjna nie jest LL(k) dla żadnego k.