Dokumentace úlohy DKA: Determinizace konečného automatu v PHP 5 do IPP 2015/2016
Jméno a příjmení: David Kozák
Login: xkozak15

Úvod

Cílem projektu bylo napsat v jazyce PHP5 skript pro determinizaci konečného automatu. Skript má být schopný automat načíst a provést jednu z následujících operací: determinizaci, odstranění epsilon přechodů, či automat pouze vypsat. Kromě základní funkcionality jsem implementoval i rozšíření RUL, STR a WSA.

Postup řešení

Skript nejdříve zpracuje a vyhodnotí argumenty, se kterými byl spuštěn. V rámci této fáze též načte data ze vstupního souboru a otevře výstupní soubor pro zápis. Pokud není vstupní soubor určený argumentem **–input**, tak dojde k načítání dat ze STDIN. Obdobně pokud není specifikován výstupní soubor argumentem **–output**, dojde k vypsání výstupu do STDOUT. Také dojde ke kontrole správnosti dané kombinace argumentů, neboť argument **–d** či **--determinize** pro determinizaci nemůže být kombinován s **–e** či**--no-epsilon-rules** pro odstranění epsilon přechodů. Pokud není žádná operace specifikována, dojde pouze k vypsání konečného automatu. Argument **–i** či **–case-insensitive** spustí skript v režimu, kdy nedochází k rozlišování malých a velkých písmen. Interně je toto zařízeno převedením všech velkých písmen na malé. Řídící pokyny získané zpracováním argumentů jsou uloženy do globální proměnné $arguments.

Pro zpracovávání vstupních dat jsem se implementoval lexikální a syntaktickou analýzu. Pro lexikální analýzu jsem využil konečného automatu. Syntaktická analýza je implementována na základě LL gramatiky. V další fázi dojde k sémantickým kontrolám. Zde se ověří neprázdnost vstupní abecedy a zda se nachází počáteční stav a koncové stavy v množině stavů.

Poté dojde v závislosti na vstupních argumentech k příslušné operaci (determinizace, odstranění epsilon přechodů, převod na dobře specifikovaný konečný automat, či pouze vypsání automatu ve specifikované formě). Nakonec dojde k vypsání výsledku na standardní výstup či do souboru.

Implementační detaily

Lexikální analýza je implementována klasickým způsobem za pomoci konečného automatu, jehož grafická reprezentace se nachází na konci zprávy. Pro syntaktickou analýzu jsem zvolil metodu rekurzivního sestupu. Syntaktická analýza též zároveň shromažďuje informace o stavech, symbolech, pravidlech, počátečním stavu a koncových stavech. Samotný konečný automat je implementován jako třída, která dostane právě výše zmíněnou pětici parametrů v konstruktoru. Ověření sémantické správnosti je implementováno jako jedna z veřejných metod. Algoritmy pro odstranění epsilon přechodů a determinizaci jsou další veřejné metody. Dále jsem v projektu využil ještě další dvě třídy, jednu pro modelování pravidla a druhou pro složené vztahy, které vznikají při algoritmu determinizace.

LL Gramatika využitá při syntaktické analýze

S->(START_TWO)
START_TWO->{STATES},{ALPHABET},{RULES},state,{FINISH}
STATES -> state STATES_N
STATES_N -> epsilon
STATES_N -> ,state STATES_N
ALPHABET -> symbol ALPHABET_N
ALPHABET_N -> epsilon
ALPHABET_N -> ,symbol ALPHABET_N
FINISH -> state FINISH_N
FINISH_N -> epsilon
FINISH_N -> ,state FINISH_N
RULES -> RULE RULES_N
RULES -> epsilon
RULE -> state symbol -> state
RULES_N -> epsilon
RULES_N -> ,RULE RULES_N

Rozšíření

Rozšíření WSA zahrnovalo implementování algoritmu pro vytvoření dobře specifikovaného konečného automatu. Toto rozšíření jsem implementoval jako další metodu instance třídy konečného automatu. Rozšíření STR pro ověření, zda je daný řetězec přijímaný konečným automatem, jsem implementoval podobně. Pro rozšíření RUL jsem přidal speciální stav do konečného automatu lexikální analýzy pro načítání konečného stavu, který je specifikovaný tečkou za jeho identifikátorem. Pro ověření syntaktické správnosti v tomto zkráceném tvaru jsem implementoval vlastní syntaktickou analýzu. Třídě konečného automatu jsem poté přidal statickou tovární metodu, která jako parametr obdrží pole pravidel a konečných stavů a konstruktor zavolá interně po vypreparování všech potřebných informací.

LL Gramatika využitá v rozšíření RUL

START->RULE RULES_N
RULE->state symbol -> state DOT
DOT-> .
DOT-> epsilon
RULES_N->,RULE RULES_N
RULES_N->epsilon

Závěr

Zadání bylo napsat skript pro determinizaci konečného automatu. Své řešení jsem testoval za pomoci referenčních testů i vlastním jednoduchým testováním, dále jsem ho ověřil pokusným odevzdáním, kde jsem dosáhl 80-100%. Věřím tedy, že zadání bylo splněno.

Konečný automat využitý v lexikální analýze