Limbaje formale si automate - Laboratorul 2
Grupele 143 & 144
Versiune 02.03.2020

Bogdan Macovei

Martie 2020

Contents

1 Sarcina de laborator

Sa se implementeze transformarea NFA in DFA, utilizand clasele create in laboratorul anterior. (sau utilizand modul in care ati implementat prima tema)

Programul va avea o structura de forma:

```
class DFA { ... }
class NFA { ... }
...
DFA nfaToDfa (NFA M)
{
    // todo
}
...
int main()
{
    NFA M;
    ...
    f >> M;
    f.close();
    ...
    DFA N = nfaToDfa(M);
    cout << N;
    ...
    return 0;
}
```

Observatie. Functia de tranzitie din DFA are un tip de date suficient de bun pentru a lucra usor? Ganditi-va la starile care pot sa apara, de exemplu q_{023}. Cum tratati aceasta situatie? Incercati, eventual, sa returnati un DFA cu starile redenumite. (sa nu aveti q_{01}, q_{134} etc., ci tot starile q_0, q_1,...).

2 Cerinte pentru Tema 2

Termen de predare: tema se va preda in laboratorul 4 (cu acelasi principiu de notare, 70-30 in favoarea implementarii). Se pot obtine bonusuri pentru implementari eficiente si utilizarea de limbaje functionale, ori pentru utilizarea STL-ului din C++ la capacitate maxima.

Puteti sa va alegeti oricum **maxim doua** teme din lista de mai jos, urmarind si care este punctajul asociat (la suma obtinuta se adauga si cele maxim 3p din evaluarea teoretica / prezentare + bonus de maxim 0.5p din timpul laboratoarelor + bonus de maxim 0.5p din implementare).

- Transformare $NFA \rightarrow DFA$: 2 puncte;
- Transformare $\lambda - NFA \rightarrow DFA$: 2.5 puncte;
- Verificare cuvant acceptat de gramatica regulata: 3.5 puncte;
- Minimizare DFA: 3.5 puncte;
- Transformare $DFA \rightarrow REGEX$: 3.5 puncte;
- Transformare $REGEX \rightarrow DFA$: 3.5 puncte.

3 Programa urmarita la laboratoarele 1-4

3.1 Laboratorul 1

- a acoperit introducerea in teoria limbajelor formale si a automatelor;
- prezentarea limbajelor regulate si focalizarea asupra automatelor finite: DFA, NFA, λ-NFA;
- prezentarea unui mod recomandat de implementare si implementarea propriu-zisa a unui automat finit deterinist in C++11.

3.2 Laboratorul 2

- verificarea primei teme: continut teoretic + implementare cu justificare;
- implementarea individuala a transformarii NFA $\rightarrow$ DFA dupa o prezentare in prealabil a algoritmului la tabla + pseudocod;
- prezentarea altor algoritmi dintre cei propusi pentru implementare in Tema 2.

3.3 Laboratorul 3

- discutie privind Tema 2;
- introducerea familiei de limbaje independente de context;
- implementarea, la tabla + suport in PDF a unui automat push-down determinist (DPDA).

3.4 Laboratorul 4

- verificarea celei de-a doua teme: continut teoretic + implementare cu justificare;
- gramatici in Forma Normala Chomsky si transformarea unei gramatici independente de context in Forma Normala Chomsky;
- prezentarea algoritmului CYK: modul de functionare, implementare la tabla + suport in PDF;
- prezentarea cerintelor pentru Tema 3 (va fi, in principiu, o tema cu un continut teoretic mai ridicat). O parte din nota va putea fi acordata pentru implementarea corecta a algoritmilor CFG $\rightarrow$ CNF si CYK.