Autorzy: Kanstantsin Sasnouski, Tomasz Stefaniak

Automaty skończone - opis projektu

Zaimplementowanie w języku Haskell niedeterministycznego automatu skończonego (NFA), następnie stworzenie procedury przekształcenia go do DFA oraz algorytmu minimalizującego liczbę stanów DFA. Zakładamy, że przestrzenią stanów jest skończony podzbiór liczb naturalnych, natomiast alfabet wejściowy jest z góry określony i skończony.

Szkic projektu rozwiązania

- Zdefiniowanie typów danych dla stanów, alfabetu, funcji przejść oraz reprezentacji automatów
- Implementacja parsera wczytującego reprezentację automatu z pliku tekstowego (lub z konsoli) wraz z walidacją danych wejściowych
- Implementacja funkcji symulujących działanie automatów
- Implementacja procedury determinizacji NFA -> DFA
- Implementacja algorytmu minimalizacji automatu
- Dodanie wyświetlania automatów w formacie tekstowym w kluczowych etapach działania
- Testy

Wstępny podział prac

Kanstantsin:

- Definicje typów stanów, alfabetu, funkcji przejścia
- Implementacja wyświetlania automatów NFA i DFA
- Implementacja funkcji determinizacji

Tomasz:

- Implementacja reprezentacji automatu NFA oraz DFA
- Implementacja wczytywania automatu NFA z pliku
- Implementacja funkcji symulujących działanie automatów

Wspólna praca (ciężkie z góry do podziału):

- Implementacja algorytmu minimalizującego DFA
- Dodanie wyświetlania automatów w istotnych momentach
- Napisanie testów

Wykorzystywane biblioteki

- Prelude (Podstawowe typy i funkcje)

- System.IO (Obsługa wejścia i wyjścia, np. by łatwo wczytywać różne automaty)
- HLint (Zadbanie o czysty kod)