Analiza Algoritmilor: Tema 1
BIN to BCD and back

Responsabili:
Mateescu Sorin, Udrescu Alexandra
Mihai Dumitru

Termen de predare:
28 Noiembrie 2021, ora 23:00

Descriere generală

În cadrul temei va trebui să redactați Mașini Turing utilizând notația prezentată în cel de-al doilea laborator.

Format input

Fiecare fișier de implementare trebuie să aibă următorul format:

- prima linie conține alfabetul intern al mașinii: o serie de caractere, terminată cu un punct "." (caracterul gol, "#" face parte din alfabetul oricărei mașini, deci nu trebuie să se regăsească aici.
  Exemplu: 01abcX.
- urmează apoi una sau mai multe subrutine, separate între ele de o linie goală. Formatul subrutinelor este același ca cel descris în laboratorul 2.

Starea inițială trebuie să se numească "`start`". Poate să existe o singură stare finală, care să se numească "`accept`" (dacă aveți un design care implică mai multe stări finale, adăugați la acestea, pe fiecare simbol, o tranziție către "`accept`").

Task 0: README

Pentru acordarea punctajului este obligatorie documentarea rezolvării. Explicați pe scurt cum ați gândit, motivați utilizarea diferitelor stări și tranziții (mai puțin cele triviale).

Task 1: BIN to BCD

(5p) Task 1.1: implementați o M.T. care shiftează tot conținutul benzii la stânga cu un număr de căsuțe egal cu lungimea cuvântului, iar pe căsuțele unde a fost scris cuvântul inițial se va scrie “X”

(5p) Task 1.2: implementați o M.T. care compară cu 5 un număr reprezentat în binar, șterge conținutul benzii și scrie 0 dacă numărul este mai mic decât 5, 1 altfel

(5p) Task 1.3: implementați o M.T. care adună 3 la un număr reprezentat în binar

(35p) Task 1.4: implementați o M.T. care transformă un string binar din BIN în BCD; la final trebuie să se afle doar “1” și “0” pe bandă

Task 2: BCD to BIN

(5p) Task 2.1: implementați o M.T. care shiftează tot conținutul benzii la dreapta cu un număr de căsuțe egal cu lungimea cuvântului, iar pe căsuțele unde a fost scris cuvântul inițial se va scrie “X”

(10p) Task 2.2: implementați o M.T. care scade 3 dintr-un număr reprezentat în binar

(35p) Task 2.3: implementați o M.T. care transformă un string binar din BCD în BIN; la final trebuie să se afle doar “1” și “0” pe bandă

Indicații:

- Algoritmul este descris aici
- Eliminați toate simbolurile de “0” redundante (din partea stângă a benzii)
- Simbolul “#” are semnificația unui spațiu; la finalul secvenței de instrucțiuni trebuie să se regăsească spații pe bandă doar la stânga și la dreapta rezultatului (adică un șir de forma “010#001” nu este valid!)

Testare

Dependințe: pentru a rula checkerul automat, va trebui să aveți python3, precum și pachetul de python “`antlr4`”. Cel mai probabil îl puteți instala folosind “`python3 -m pip install antlr4`”, sau instalând “`python-antlr4`” din repozitorul distribuției voastre (e.g. “`sudo apt install python-antlr4`”).

Aveți la dispoziție un compilator pentru limbajul de subrutine, precum și un checker automat împreună cu o suită de teste.

Va trebui să redactați rezolvările fiecărui subpunct în fișierul corespunzător din directorul “`solutions/`”; apoi puteți, din directorul “`local-checker/`” invoca scriptul “`tester.py`” pentru a vă testa implementarea (utilizați argumentul “`help`” pentru a explora diversele moduri de rulare).

Pentru a vă ajuta la debugging, vă punem la dispoziție și un script capabil să genereze un format acceptabil pentru [acest simulator grafic de mașină Turing.]

Din directorul “`local-checker/`”, puteți rula “`python3 ParseMain.py <cale/catre/subpunct.tm> tms`” pentru a converti implementarea voastră la formatul acceptat de simulator. Apoi puteți copia codul în cadrul simulatorului și selecta “Compile”. După compilarea cu succes, puteți seta inputul inițial de pe bandă (stânga-sus) și apăsa play pentru a vedea o simulare grafică.

Trimitere

Pentru a încărca rezolvarea temei, folosiți următorul format:

- ☐ NUME_PRENUME_32XCB.zip
  - ☐ solutions/
    - ☐ task_1_1.tm
    - ☐ task_1_2.tm
    - ☐ …
  - ☐ README