Analiza Algoritmilor
Tema 1 - Sudoku 4x4

Termen de predare: **14.Nov.2022, 23:50**

Responsabili temă: Alexandra UDRESCU
Mihai-Valentin DUMITRU

Data publicării: *01.Nov.2022*
Ultima actualizare: *03.Nov.2022*

Pentru prima temă, va trebui să scrieți o Mașină Turing care să verifice dacă un careu de sudoku 4x4 primit ca input este valid sau nu.

1	2	3	4
3	4	1	2
2	3	4	1
4	1	2	3

În contextul temei, un *careu* este o matrice de 4 x 4 *"căsuțe"*; deasemenea, acesta poate fi împărțit în patru submatrici de 2 x 2 căsuțe, numite "regiuni". Ca un careu să fie valid, trebui ca pe fiecare linie, pe fiecare coloană și în fiecare regiune, să se găsească exact numerele de la 1 la 4.

1 Cerință

Va trebui să scrieți tabelul de tranziție al unei Mașini Turing care verifică validitatea unui careu de sudoku 4x4.

Mașina va semnala că un careu e **valid**, tranziționând în starea finală Y. Poziția capului de citire la acest moment e irelevantă.

Mașina va semnala că un careu e **invalid**, tranziționând în starea finală N. Poziția capului de citire la acest moment e irelevantă.

Pentru a facilita elaborarea temei și a oferi posibilitatea obținerii de punctaj parțial, există trei subcerințe:

1. **Verificarea validității pe linii**. Pentru această subcerință e suficient să validați că, pe fiecare linie, se găsesc exact cifrele de la 1 la 4, chiar dacă, per total, careul nu este valid. De exemplu, următorul careu este invalid, dar *valid din punct de vedere al liniilor*:

1	2	3	4
1	2	3	4
2	3	4	1
2	4	1	3

2. **Verificarea validității pe coloane**. Pentru această subcerință e suficient să validați că, pe fiecare coloană, se găsesc exact cifrele de la 1 la 4, chiar dacă, per total, careul nu este valid. De exemplu, următorul careu este invalid, dar *valid din punct de vedere al coloanelor*:

1	2	4	4
2	1	3	3
3	4	2	2
4	3	1	1

3. **Verificarea validității pe regiuni**. Pentru această subcerință e suficient să validați că, în fiecare regiune, se găsesc exact cifrele de la 1 la 4, chiar dacă, per total, careul nu este valid. De exemplu, următorul careu este invalid, dar *valid din punct de vedere al regiunilor*:

1	2	1	4
3	4	2	3
4	2	1	2
1	3	4	3

Pentru punctaj parțial, puteți opta să implementați doar una sau două dintre aceste verificări.

Dacă nu ați implementat toate cele trei criterii de verificarea validității unui sudoku, va trebui ca, în fișierul `README`, prima linie să conțină unul sau două dintre următorii termeni, în funcție de subcerințele implementate. În caz că sunt doi termeni, aceștia trebuia separați printr-un singur spațiu.

- `lines` – pentru validarea pe linii
- `columns` – pentru validarea pe coloane
- `regions` – pentru validarea pe regiuni

Dacă ați implementat integral cerința temei, nu e nevoie să scrieți nimic în README.

Atenție 1. E important ca aceste criterii să fie fix prima linie din README; la fel de important ca, în caz că ați implementat două subcerințe, să fie separate printr-un singur spațiu.
Checkerul automat va interpreta această linie pentru a ști ce teste să ruleze și ce punctaj să acorde.

2 Format input

Inputul constă dintr-un șir ce conține toate cifrele ce constituie careul, plus niște simboluri ajutătoare care să vă ajute cu delimitarea liniilor și a regiunilor.

Formal, alfabetul de input este:

$$\Sigma_{sudoku} = \{1, 2, 3, 4, *, :\}$$

Encodingul unui sudoku constă în encodingul celor patru linii (în ordine de sus în jos), separate printr-un asterisk "*".

Encodingul unei linii constă în alipirea primelor două cifre, apoi un ":", apoi alipirea ultimelor două cifre.

De exemplu, pentru următorul careu:

1	2	4	3
3	4	1	2
2	1	3	4
4	3	2	1

Encodarea este: `*12:43*34:12*21:34*43:21*`.

Pentru următorul careu:

2	2	4	3
1	3	2	4
3	1	4	4
4	4	1	1

Encodarea este: `*22:43*13:24*31:44*44:11*`.

3 Format tabelă de tranziție

Pentru a redacta Mașina Turing există două posibilități:

1. Folosind formatul simulatorului online [†].
2. Scriind funcția de tranziție în formă de tabel într-un spreadsheet.

3.1 Sintaxa de simulator online

Descrierea unei mașini Turing se face într-un fișier text. Extensia recomandată este ".tms".

Descrierea mașinii trebuie să înceapă cu două linii speciale:

1. Prima linie începe cu cuvântul cheie *"init"*, urmat de două puncte, ":", apoi de numele stării inițiale. E.g.: "init: q0"

[†] https://turingmachinesimulator.com/

2. A doua linie este “accept: Y”. Simulatorul online vă permite să alegeți un nume arbitrar pentru starea acceptoare, ba chiar să aveți mai multe. *În contextul temei, este obligatoriu să aveți una singură, numită Y.*

După care urmează tranzițiile, înșirate una după alta. Specificarea fiecărei tranziții este formată din două linii:

1. Prima linie conține starea curentă și simbolul citit pentru care se aplică tranziția respectivă; starea și simbolul sunt separate printr-o virgulă

2. A doua linie conține starea în care mașina va trece, simbolul care trebuie scris pe bandă și direcția de mișcare a capului de citire, reprezentată de unul din simbolurile:
   - <: stânga
   - >: dreapta
   - −: rămâne pe loc

Pentru o descriere completă a sintaxei, urmăriți indicațiile de pe site.

Spre deosebire de site, checkerul temei ar trebui să permită să folosiți în alfabetul de lucru Γ, orice caracter unicode (în afară de virgulă, orice fel de “spațiu”, newline). Dacă încercați ceva mai exotic, ca un emoji din mai multe caractere sau un caracter de control și e respins, **nu vom actualiza checkerul**; schimbați simbolul.

Nu este necesar să definiți explicit toate valorile funcției de tranziție. Este posibil să existe combinații de (stare curentă, simbol citit) care să nu poată apărea în rularea mașinii pe niciun input. În acest caz, puteți să omiteți acea tranziție; acțiunea ei va fi tratată ca: `N, _, -`.

3.2 Formatul de spreadsheet excel

Descrierea unei mașini se face într-un fișier spreadsheet cu extensia `.xlsx`.

Fișierul trebuie să conțină un singur worksheet, ce conține tabelul de tranziție.

Tabelul de tranziții va conține pe prima linie lista cu toate simbolurile care fac parte din alfabetul de lucru, Γ, al mașinii, începând cu cea de-a două căsuță (B1); prima căsuță rămâne liberă. Simbolul blanc poate fi reprezentat:

- fie de simbolul: □ (cod Unicode U+25A1)
- fie de un underscore: _

Checkerul temei ar trebui să permită să folosiți în alfabetul de lucru Γ, orice caracter unicode (în afară de virgulă, orice fel de “spațiu”, newline). Dacă încercați ceva mai exotic, ca un emoji din mai multe caractere sau un caracter de control și e respins, **nu vom actualiza checkerul**; schimbați simbolul.

Pe prima coloană, vor fi listate toate stările din mulțimea de stări Q a mașinii, începând cu cea de-a doua căsuță (A2); prima căsuță rămâne liberă. Prin convenție, prima stare din coloană (cea din căsuță A2) este *starea inițială* q_1 a mașinii. O stare este reprezentată dintr-un nume care poate consta în oricâte caractere.

În fiecare celulă din tabelul rezultat trebuie scris un triplet care corespunde valorii funcției de tranziție δ pentru perechea formată din starea corespunzătoare liniei respective și din simbolul corespunzător coloanei respective. Tripletul constă în numele stării următoare, noul simbol scris pe bandă și direcția în care trebuie deplasat capul de citire:

- $\leftarrow$ (caracterul Unicode U+2190) SAU < SAU L: stânga
- $\rightarrow$ (caracterul Unicode U+2192) SAU > SAU R: dreapta
- − SAU H: rămâne pe loc

Nu este necesar să definiți explicit toate valorile funcției de tranziție. Este posibil să existe combinații de (stare curentă, simbol citit) care să nu poată apărea în rularea mașinii pe niciun input. În acest caz, puteți lăsa celula corespunzătoare din tabel goală; ea va fi tratată ca: `N, □, -`.

4 Checker și teste automate

Pentru a facilita dezvoltarea temei, vă punem la dispoziție un checker automat și o suită de teste.

Checkerul este scris în Python 3, dar nu e nevoie să interacționați cu codul său, ci doar să-l rulați; deci nu e nicio problemă dacă nu sunteți familiari cu Python. Pentru a-l rula aveți nevoie de pachetul `openpyxl`.

```
python3 -m pip install --user openpyxl
```

Checkerul are rolul principal de a citi și interpreta mașina redactată de voi și de a-i simula comportamentul pe un input.

În continuare vă prezentăm toate facilitățile checkerului, care ar putea fi utile în redactarea temei.

4.1 Rulare teste

Pentru a rula toate testele necesare, rulați:

```
./checker.py --tm sudoku.<tms sau xlsx> --run-tests
```

Checkerul va rula toate testele și va afișa, după fiecare, *"PASS"* sau *"FAIL"* după caz. Dacă rularea necesită mai mult de 100000 de tranziții, va fi oprită și va fi afișat *"SLE"* (Step Limit Exceeded). Pentru fiecare test cu rezultatul "PASS" veți primi un punct.

În urma rulării, checkerul creează un director `logs/` în care se vor găsi, pentru fiecare test, câte un fișier cu lista de configurații prin care a trecut mașina pe inputul respectiv, de la configurația inițială la cea finală (sau la configurația cu numărul 100000). Pentru testul cu numărul `i`, veți găsi fișierul `logs/dbglog_i`.

Testele se găsesc în fișierul `tests.py`. Formatul lor este: șirul de input între ghilimele, separat cu un ":" de către un triplet de trei valori boolene; acestea corespund validității careului respectiv pe linii, coloane, respectiv regiuni. Deci un careu valid va avea tripletul asociat: `(True, True, True)`.

4.2 Rulare pe un singur input

Pentru a rula mașina voastră pe un input anume (e.g. *34:21*12:43*21:14*41:32*), rulați:

```
./checker.py --tm sudoku.<tms sau xlsx> --test-input "*34:21*12:43*21:14*41:32*"
```

Dacă rularea necesită mai mult de 100000 de tranziții, va fi oprită și va fi afișat mesajul "Step Limit Exceeded!".

În urma rulării, checkerul creează un fișier `debug.log` în directorul curent, ce conține lista de configurații prin care a trecut mașina pe inputul respectiv, de la configurația inițială la cea finală (sau la configurația cu numărul 100000).

4.3 Rulare pe criterii anume

Dacă vreți să rulați toate testele folosind verificarea pentru doar unul sau două din criteriile de validare, puteți folosi argumentul `-validation-type`.

```
./checker.py --tm sudoku.<tms sau xlsx> --run-tests --validation-type lines regions
```

Criteriile posibile sunt: lines, columns, regions, all Mai multe criterii trebuiesc despărțite printr-un spațiu.

4.4 Conversie de la un format la altul

Dacă vreți sa vedeți reprezentarea mașinii în celălalt format (e.g. ați lucrat cu sintaxa de simulator online și vreți să vedeți forma de table excel), puteți converti mașina astfel:

```
./checker.py --tm sudoku.<tms sau xlsx> --output converted_sudoku.<xlsx sau tms>
```

Checkerul "știe" ce format să citească verificând extensia; similar, știe ce format să scrie.

Atenție 2. Dacă ați lucrat într-un table excel și doriți să o convertiți în formatul `.tms` pentru a vedea live cum funcționează mașina voastră în simulatorul online, aveți grijă că setul de simboluri permis este mai restrâns; dacă folosiți caractere în afara celor ASCII, probabil simulatorul online le va respinge. Checkerul înlocuiește simbolul blanc □ cu underscore _

4.5 Setare număr maxim de pași

Pentru a rula mașina voastră pe un input (sau pe teste) cu o limită de pași custom (e.g. 10000000) puteți folosi argumentul:

```
--max-steps=10000000
```

Scopul acestei opțiuni este strict ca să vă ajute la depanarea comportamentului mașinii. Tema va fi testată cu o limită de 100000.

5 Punctaj

Tema valorează 10 puncte din totalul de 60 de puncte de parcurs. Tema va fi punctată de la 0 la 100, după următoarele criterii.

- verificarea pe linii: **30 de puncte**
- verificarea pe coloane: **30 de puncte**
- verificarea pe regiuni: **30 de puncte**

Există 30 de teste; în funcție de ce e specificat pe prima linie din README, checkerul se va uita doar la criteriile respective (dacă în README nu e specificat nimic, checkerul va verifica validarea tuturor celor trei criterii).

Fiecare test valorează 1 punct per criteriu. Checkerul oferă încă 10 puncte în plus, care vor fi explicate imediat. Astfel, dacă ați făcut verificarea doar pe linii, checkerul va acorda maxim 40 de puncte; dacă ați făcut verificarea pe linii și coloane, checkerul va acorda maxim 70 de puncte etc.

Restul de 10 puncte vor fi acordate în urma unei examinări manuale ce are ca scop evaluarea clarității implementării și a prezentării acesteia în README. Evaluarea va porni de la 10 și se vor scădea puncte pentru criterii ca:

- stări în plus în tabelă, la care nu se poate ajunge
- simboluri în plus în tabelă, care nu apar în input și nici nu sunt scrise
- lipsă README
- explicație insuficientă în README

Atenție 3. Scopul testelor este să ajute cu dezvoltarea cerinței și cu punctarea temei.

Voi trebuie însă să rezolvați *cerința*, indiferent de teste. În cazul în care implementarea voastră nu respectă cerința, punctajul oferit de checker poate fi anulat total sau parțial în urma examinării manuale.

De exemplu:

- dacă descoperiți că un test e greșit, semnalați acest lucru pe forum, nu modificați mașina încât să treacă testul greșit.
- o mașină care acceptă orice input va primi punctaj 0, chiar dacă pe checker vor trece toate testele în care inputul reprezentă un careu valid.

6 Trimitere temă

Tema voastră va consta într-un fișier cu tabela de tranziție, cu unul din următoarele nume, în funcție de formatul folosit:

- `sudoku.tms`
- `sudoku.xlsx`

Alături de un fișier `README` în care să descrieți, high-level și succint, designul mașinii implementate. Fișierul trebuie să se numească `README`, nu `ReadMe`, `readme` etc. Trebuie să nu aibă nicio extensie (“.txt”, “.md” etc.) și să fie un format plaintet (nu pdf, doc, rich-text etc.)

Acestea trebuiesc puse într-o arhivă `.zip` cu numele:

IDLDAP_Grupa_Tema1.zip

Unde IDLAP este ID-ul cu care vă logați pe moodle.

De exemplu:

mihai.dumitru2201_321CB_Tema1.zip

Tema va trebui încărcată pe moodle la adresa: TODO

6.1 Deadline

Tema trebuie trimisă până la data de **14 noiembrie 2022, ora 23:50**.

7 Forum

Pentru orice întrebări sau neclarități legate de conținutul temei, folosiți forumul temei . Evitați punerea de întrebări pe canale private (e.g. Teams, mail).

Nu postați pe forum fișierul cu implementarea voastră, sau fragmente din acesta. Nu postați pe forum fișierul cu lista de configurații obținute pe un input, sau fragmente din acesta.

8 Plagiat

Rezolvarea temei este individuală.

Copierea totală sau parțială a temei de la un alt student va rezulta în aplicare sancțiunilor menționate în regulament[‡] pentru *toți studenții implicați*, fiind irelevant cine de la cine a luat.

Desigur, aveți voie să discutați între voi idei generale de soluții. Dar nu vă uitați pe *implementarea* unui coleg; nu preluați exemple concrete de tranziții, nu cereți ajutor cu depanarea concretă a implementării voastre (i.e. nu prezentați cuiva lista cu configurații pe un anumit input).

În situațiile în care nu sunteți siguri ce constituie un act de plagiat, întrebați pe forumul temei.

[‡] https://ocw.cs.pub.ro/ppcarte/doku.php?id=aa:intro:rules