UNIVERSITATEA ”ALEXANDRU-IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ

LUCRARE DE LICENȚĂ

Verificarea algoritmului DPLL in F*

propusă de

Alexandru Donica

Sesiunea: iunie/iulie, 2023

Coordonator științific

Conf. Dr. Ștefan Ciobâcă

UNIVERSITATEA ”ALEXANDRU-IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ

Verificarea algoritmului DPLL in F*

Alexandru Donica

Sesiunea: iunie/iulie, 2023

Coordonator științific

Conf. Dr. Ștefan Ciobâcă

Avizat,

Îndrumător lucrare de licență,

Conf. Dr. Ștefan Ciobâcă.

Data: ..... Semnătura: .....

Declarație privind originalitatea conținutului lucrării de licență

Subsemnatul **Donica Alexandru** domiciliat în **România, jud. Iași, mun. Iași, strada Costache Negri, nr. 35, bl. A1, ap. 42**, născut la data de **07 aprilie 2000**, identificat prin CNP **5000407226761**, absolvent al Facultății de informatică, **Facultatea de informatică** specializarea **informatică**, promoția 2022, declar pe propria răspundere cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art. 143 al. 4 și 5 referitoare la plagiat, că lucrarea de licență cu titlul **Verificarea algoritmului DPLL in F*** elaborată sub îndrumarea domnului **Conf. Dr. Ștefan Ciobâcă**, pe care urmează să o susțin în fața comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea conținutului ei într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice în vederea facilitării falsificării de către cumpărător a calității de autor al unei lucrări de licență, de diplomă sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Data: ..... Semnătura: .....

Declarație de consimțământ

Prin prezenta declar că sunt de acord ca lucrarea de licență cu titlul **Verificarea algoritmului DPLL in F***, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test, etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de informatică.

De asemenea, sunt de acord ca Facultatea de informatică de la Universitatea ”Alexandru-Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Absolvent **Alexandru Donica**

Data: ..... Semnătura: .....

Cuprins

Motivație

Rolul unui 'SAT solver' este de a rezolva problema satisfiabilității booleene, făcându-se si in ziua de azi cercetări care au scopul de a îmbunătăți algoritmi existenți. Acest 'SAT solver' găsește o soluție pentru o formula data in cazul in care formula este satisfiabilă, in caz contrar formula este nesatisfiabilă.

Corectitudinea oricărui rezultat al unei formula satisfiabile poate fi verificat folosind algoritmi simpli. Însă un 'SAT solver' complex creat pentru procesa formule de dimensiuni din ce in ce mai mari sau pentru a avea o viteza de rezolvare a problemei mai rapida decât alți 'SAT solveri' folosiți pot conține erori de programare ce ar produce rezultate false, cum ar fi, in urma procesării unei formula nesatisfiabile acesta sa enunțe ca este satisfiabilă, sau invers.

De aceea este importanta crearea unor programe care verifica corectitudinea acestor 'SAT solveri' si am creat un 'SAT solver' verificat formal folosind limbajul de programare F* (FStar).

Introducere

detalii despre problema sat
conexiuni cu probleme reale
solvere create pana acum
probleme care apar la aceste solvere
importanta verificarii solverelor
acest solver implementeaza alg DPLL
descriere la alg DPLL
verificare formala
fstar - limbaj de programare si verificare foloseste z3 smt solver
ce presupune un solver verificat
de ce acest solver e verificat si in ce mod de catre fstar

Capitolul 1

Algoritmul DPLL

(Davis-Putnam-Logemann-Loveland)

Capitolul 2

Detalii de implementare

2.1 Dividerea pe module?

rolul fiecarui fisier + cea mai importanta functie de acolo?
link catre fisiere de input?

2.2 despre tipurile de date

2.3 cum functioneaza?

despre asserturi, putin cod efectiv, multe lemme/asserturi
compara cate linii is in .ml fata de .fst

2.4 despre tipurile de date simple

2.5 proofness / soundness / completeness / etc..

2.6 Metrici orientative? / poate in concluzie?

timp sa returneze sat
timp sa dea unsat cam mult
500 secunde sa compileze, verifice si extraga cod pt toate fisierele
datatypes - 8 secunde
datatypeUtils - 33 sec

dpllpropagation - 285 sec

occmatrix - 91 sec

fileparser - 4 sec

dpll - 71 sec

convertorToString - 3 sec

main - 3 sec

Capitolul 3

Pasi necesari pentru a reproduce

3.1 Setup extra / aplicatii externe?

fstar github install file la referinte?

3.2 punerea codului in locul corect? / instructiuni unix bine de stiut

sa mentionez de fisierul makefile

Concluzii

acest solver nu prezinte cele mai eficiente structuri de date si euristici

insa prezinta cum arata specificatii functionale pt algoritmului dpll si implicit reprezinta o baza pt specificatiile oricarei extensii ale sale

solverul este sound, complet, garantat ca se termina? ,verificat formal

schimbarea structurilor de date spre o forma mai eficienta nu ar fi una dificila

Bibliografie

- fstar tutorial,
- fstar github,
- toate linkurile referentiate mai sus?