Gilmorova procedura

Dimitrije Radjenović

3. septembar 2025.

Sažetak

U ovom seminarskom radu predstavljena je Gilmorova procedura, prva implementirana metoda za automatsko dokazivanje teorema u logici prvog reda. Prikazan je teorijski okvir na kome se procedura zasniva, uz sistematski opis algoritma za njenu realizaciju, primer primene i opis C++ implementacije. Poseban akcenat je stavljen na istorijski doprinos procedure, jer je označila početak razvoja praktičnih pristupa automatskom rezonovanju i otvorila put savremenim metodama dokazivanja. Iako nepraktična za širu upotrebu, Gilmorova procedura je zadržala edukativni značaj i važnu ulogu u razumevanju osnova logičkog zaključivanja.

1 Uvod

Automatsko dokazivanje teorema predstavlja jednu od centralnih oblasti matematičke logike i računarskih nauka, sa primenama u formalnoj verifikaciji softvera i inteligentnim sistemima za rezonovanje [1]. U logici prvog reda, problem valjanosti formula je u opštem slučaju neodlučiv [2]-[4]. Medjutim, Erbranova teorema (fr. *Théorème de Herbrand*), formulisana 1930. godine, omogućava poluodlučiv postupak, odnosno algoritam koji za svaku nezadovoljivu formulu može dokazati da je nezadovoljiva [1, 5]. Na osnovu ove teoreme implementirana je prva procedura za ispitivanje valjanosti formula u logici prvog reda, Gilmorova procedura (Paul C. Gilmore), koja se smatra jednim od najranijih automatskih dokazivača teorema [6].

Gilmor je svoju proceduru implementirao na računaru *IBM 704* i testirao je na skupu formula različite složenosti. Jednostavniji primeri su se brzo dokazivali, u nekim slučajevima za manje od jedne sekunde, dok su složeniji izazvali prekoračenje raspoložive memorije, usled eksponencijalnog rasta broja generisanih baznih instanci [6]. Ključni nedostatak procedure je potpuna odvojenost faze generisanja instanci i faze provere zadovoljivosti, što dovodi do neefikasnog korišćenja resusrsa i ograničava njenu praktičnu primenu [5, 6].

Motivacija za proučavanje Gilmoreove procedure proizilazi iz njenog istorijskog značaja kao prve implementirane metode za automatsko dokazivanje teorema u logici prvog reda [6]. Njena jednostavna i jasna struktura, zasnovana na primeni Erbranove teoreme, čini je izuzetno korisnim edukativnim alatom za razumevanje osnova automatskog rezonovanja. Kao preteča metode rezolucije sa unifikacijom, Gilmorova procedura predstavlja važan korak u razvoju automatskog rezonovanja [5].

Ostatak ovog rada organizovan je na sledeći način. U poglavlju 2 navedeni su osnovni pojmovi, definicije i teoreme koji su korišćeni u ostatku rada. U pogla-

vlju 3 dat je detaljan opis algoritma za Gilmorovu proceduru, uz primer njene primene. Poglavlje 4 sadrži informacije o njenoj implementaciji. Na kraju, poglavlje 5 donosi zaključna razmatranja i osvrt na značaj i ograničenja Gilmorove procedure u kontekstu razvoja automatskog rezonovanja.

2 Osnove

U ovom poglavlju navešćemo osnovne pojmove, definicije i teoreme koji su, u ovom radu, korišćeni za prikaz Gilmorove procedure.

2.1 Normalne forme

- **Negaciona normalna forma**

  Definicija: Formula prvog reda je u negacionoj normalnoj formi (NNF) akko je izgradjena od literala (prvog reda) korišćenjem isključivo veznika $\wedge$ i $\vee$ i kvantifikatora ili je logička konstanta ($\top$ ili $\bot$).

- **Prenex normalna forma**

  Definicija: Formula je u prenex normalnoj formi ako je oblika
  $$Q_1 x_1. \ldots Q_n x_n. F,$$
  pri čemu su Q_i kvantifikatori $\forall$ ili $\exists$, a formula F ne sadrži kvantifikatore.

- **Skolemova normalna forma**

  Definicija: Formula je u Skolemovoj normalnoj formi ukoliko je oblika
  $$\forall x_1. \ldots \forall x_n. F$$
  pri čemu formula F ne sadrži kvantifikatore. Formule u Skolemoj normalnoj formi nazivamo i univerzalno kvantifikovane formule. Formulu F nazivamo matricom univerzalno kvantifikovane formule.

  Teorema: Formula dobijena skolemizacijom formule F je ekvizadovoljiva formuli F.

- **Disjunktivna normalna forma**

  Definicija: Iskazna formula je u disjunktivnoj normalnoj formi (DNF) ako je oblika
  $$A_1 \vee A_2 \vee \cdots \vee A_n,$$
  pri čemu je svaka od formula A_i $(1 \leq i \leq n)$ konjunkcija literala.

2.2 Problem zadovoljivosti u logici prvog reda

- **Neodlučivost logike prvog reda**

  Teorema: Problem ispitivanja nezadovoljivosti univerzalno kvantifikovane formule je poluodlučiv.

  Posledica: Problem ispitivanja nezadovoljivosti formule prvog reda u opštem obliku je poluodlučiv.

- **Valjanost**

  Problem valjanosti: Problem valjanosti formule F je dualan problemu nezadovoljivosti formule $\neg F$ (koji je komplementaran problemu zadovoljivosti formule $\neg F$).

Posledica: Problem valjanosti formula prvog reda u opštem obliku je neodlučiv, ali jeste poluodlučiv.

Napomena: U praksi se valjanost obicno dokazuje pobijanjem, tj. dokazivanjem da je negacija date formule nezadovoljiva:
- procedura poluodlučivanja će u konačnom broju koraka pobiti svaku nezadovoljivu formulu.
- u slučaju da je formula zadovoljiva, tada procedura poluodlučivanja moze raditi beskonačno dugo.

2.3 Erbranova teorema

Erbranova teorema daje jedan od načina da se konstruiše postupak poluodlučivanja, odnosno algoritam koji za svaku nezadovoljivu formulu može da dokaže da je nezadovoljiva. Dualno, ovakav postupak omogućava da se za svaku valjanu formulu dokaže da je valjana tako što se dokazuje nezadovoljivost njene negacije.

- **Erbranov univerzum**

  Definicija: Skup svih baznih termova jezika $\mathcal{L}$ nazivamo Erbranov univerzum nad $\mathcal{L}$ i označavamo sa $H(\mathcal{L})$.

  Definicija: Erbranov univerzum formule $H(F)$ je Erbranov univerzum jezika sačinjenog od simbola koji se javljaju u toj formuli.

- **Bazne instance:** Ako u formuli F bez kvantifikatora sve njene promenljive zamenimo elementima Erbranovog univerzuma, dobijamo baznu formulu. Ovako dobijene formule nazivamo baznim instancama formule F.

- **Teorema (Erbran):** Univerzalno kvantifikovana formula je zadovoljiva akko je skup svih baznih instanci njene matrice zadovoljiv. Preciznije, formula $\forall x_1 \dots x_n.\, F.$ je zadovoljiva akko je zadovoljiv skup

$$\{F[x_1 \to t_1, \dots, x_n \to t_n] \mid t_1, \dots, t_n \in H(F)\}.$$

  Teorema: Univerzalno kvantifikovana formula je nezadovoljiva akko postoji konačan nezadovoljiv podskup njenih baznih instanci.

3 Opis metode

Gilmorova procedura se zasniva se na Erbranovoj teoremi, koja tvrdi da je univerzalno kvantifikovana formula nezadovoljiva ako i samo ako postoji konačan skup njenih baznih instanci koji je iskazno nezadovoljiv. Ključni koraci Gilmorove procedure su:

1. **Negacija polazne formule:** Polazna formula F se negira, čime se dobija $\neg F$.
2. **Transformacija u Skolemovu normalnu formu:** Formula $\neg F$ se transformiše u ekvizadovoljivu formulu u Skolemovoj normalnoj formi, kroz sledeće korake:
   - Prenex transformacija: svi kvantifikatori se pomeraju na početak formule.

- Skolemizacija: eliminišu se svi egzistencijalni kvantifikatori.

Dobija se univerzalno kvantifikovana formula:

$$\forall x_1. \ldots \forall x_n. B,$$

gde B ne sadrzi kvantifikatore.

3. **Generisanje Erbranovog univerzuma:** Erbranov univerzum formule F, u oznaci $H(F)$, se gradi korak po korak:
   - Početni korak: ako formula sadrži konstante, tada je $H_0 = \{$sve konstante koje se javljaju u formuli$\}$. Ukoliko formula nema nijednu konstantu, dodaje se jedna nova konstanta, npr. c, kako Erbranov unverzum ne bi bio prazan. Tada je $H_0 = \{c\}$.
   - Svaki sledeći korak: $H_{k+1} = H_k \cup \{f(t_1, \ldots, t_m) \mid f$ je funkcijski simbol, $t_i \in H_k\}$.

   Tako se dobija: $H_0 \subseteq H_1 \subseteq H_2 \subseteq \ldots$.
4. **Generisanje baznih instanci:** Sve promenljive u matrici B univerzalno kvantifikovane formule zamenjuju se istovremeno svim mogućim permutacijama termova iz trenutnog nivoa Erbranovog univerzuma H_k. Dobijene formule su bazne instance.
5. **Konjunkcija i prevodjenje u DNF:** Nakon što su sve bazne instance za trenutni H_k generisane, one se spajaju konjunkcijom u jednu veliku iskaznu formulu. Zatim se ova konjunkcija formula prevodi u disjunktivnu normalnu formu (DNF).
6. **Provera iskazne nezadovoljivosti dobijene DNF:** Ako je dobijena DNF iskazno nezadovoljiva, to znači da svaka njena disjunkcija sadrži par komplementarnih literala. Tada je i univerzalno kvantifikovana formula nezadovoljiva, pa je $\neg F$ nezadovoljiva. Prema tome, formula F je valjana i procedura se u tom slučaju uspešno završava.
7. **Proširenje Erbranovog univerzuma i iterativno ponavljanje:** Ako DNF nije nezadovoljiva, postupak se ponavlja od 3. koraka. Gilmorova procedura je poluodlučiva: ako je formula valjana, dokaz će biti pronadjen u konačnom broju koraka; u suprotnom, procedura može beskonačno trajati (pod uslovom da Erbranov univerzum nije konačan).

3.1 Primer primene Gilmorove procedure:

Dokazati da je formula $(\exists x)(P(x) \Rightarrow (\forall y)P(y))$ *valjana.*

1. **Negacija polazne formule:**

$$\begin{aligned}\neg\big((\exists x)(P(x) \Rightarrow (\forall y)P(y))\big) &\iff (\forall x)\neg(P(x) \Rightarrow (\forall y)P(y)) \\ &\iff (\forall x)\big(P(x) \wedge \neg(\forall y)P(y)\big) \\ &\iff (\forall x)\big(P(x) \wedge (\exists y)\neg P(y)\big)\end{aligned}$$

2. **Transformacija u Skolemovu normalnu formu:**
   - Prenex transformacija: $(\forall x)(\exists y)\big(P(x) \wedge \neg P(y)\big)$
   - Skolemizacija: $(\forall x)\big(P(x) \wedge \neg P(f(x))\big)$

3. **Generisanje Erbranovog univerzuma:** Pošto formula nema konstante, dodajemo jednu proizvoljnu konstantu c.
$$H_0 = \{c\}$$
4. **Generisanje baznih instanci:** Zamenimo x u $(\forall x)(P(x) \wedge \neg P(f(x)))$ sa c:
$$P(c) \wedge \neg P(f(c))$$
5. **Konjunkcija i prevodjenje u DNF:** Formula $P(c) \wedge \neg P(f(c))$ je već u DNF (sa jednom klauzom) pa ne moramo da je transformišemo.
6. **Provera iskazne nezadovoljivosti dobijene DNF:**
   Za $H_0 = \{c\}$, konjunkcija instanci je: $P(c) \wedge \neg P(f(c))$.
   Ova formula nije nezadovoljiva — može biti zadovoljena (npr. ako je $P(c)$ tačno, a $P(f(c))$ netačno). Pošto nismo utvrdili nezadovoljivost, prelazimo na sledeći korak.
7. **Proširenje Erbranovog univerzuma i iterativno ponavljanje:**
   - $H_1 = \{c,\ f(c)\}$
   - Instanciramo formulu $(\forall x(P(x)) \wedge \neg P(f(x)))$ za svaki term u H_1:
     - Za $x = c$: $\quad P(c) \wedge \neg P(f(c))$
     - Za $x = f(c)$: $\quad P(f(c)) \wedge \neg P(f(f(c)))$
   - Prevodjenje u DNF:
     $$P(c) \wedge \neg P(f(c)) \wedge P(f(c)) \wedge \neg P(f(f(c)))$$
     Formula je već u DNF (sa jednom klauzom).
   - Provera zadovoljivosti: Uočavamo da se pojavljuju medjusobno suprotni literali:
     $$P(f(c)) \quad \text{i} \quad \neg P(f(c))$$
     Dakle, konjunkcija sadrži kontradikciju i jeste iskazno nezadovoljiva. Zaključujemo da je i univerzalno kvantifikovana formula nezadovoljiva, pa je i $\neg F$ nezadovoljiva. Prema tome, formula F je valjana i prekidamo sa radom.

4 Implementacija

Gilmorova procedura je implementirana u C++ i podeljena na osam fajlova:: 4 zaglavlja (`.h`) i 4 odgovarajuća izvorna fajla (`.cpp`). Sve deklaracije su izdvojene u zaglavljima, a implementacija u `.cpp` fajlovima.

4.1 Organizacija koda

4.1.1 `first_order_logic.h` i `first_order_logic.cpp`

Ovi fajlovi sadrže deklaracije i implementacije struktura za formule i termove logike prvog reda. Za svaki tip formule (i terma) definisana je posebna struktura, a svi tipovi su objedinjeni korišćenjem `std::variant`. Radi bezbednog upravljanja memorijom i izbegavanja grešaka, koriste se deljeni pokazivači (`std::shared_ptr`). Normalne Forme su realizovane kao vekor klauza, klauze kao vektor literala, a literali kao strukture koje sadrže zapis atoma i indikator pozitivnosti. Ključne definicije tipova i struktura prikazane su u Listingu 1.

```
using Formula=std::variant<True,False,Atom,Not,Binary,Quantifier>;
using FormulaPtr=std::shared_ptr<Formula>;

using Term=std::variant<VariableTerm,FunctionTerm>;
using TermPtr=std::shared_ptr<Term>;
struct VariableTerm{std::string symbol;};
struct FunctionTerm{std::string symbol;std::vector<TermPtr> args;};

//
    ------------------------------------------------------------------------

struct False {};
struct True {};
struct Atom{ std::string name; std::vector<TermPtr> args;};
struct Not {FormulaPtr subformula;};
struct Binary{
  enum Type {And, Or,Imp,Eq} type;
  FormulaPtr left,right;
};
struct Quantifier{
  enum Type { All, Exist} type;
  std::string var;
  FormulaPtr subformula;
 };

struct Literal {
    bool pos;
    std::string name;
};
using Clause = std::vector<Literal>;
using NormalForm = std::vector<Clause>;
```

Listing 1: Ključne definicije tipova i struktura za logiku prvog reda.

Ključne funkcije su:

- `FormulaPtr nnf(FormulaPtr f)` – transformiše formulu u NNF;
- `FormulaPtr prenex(FormulaPtr f)` – transformiše formulu u prenex formu;
- `FormulaPtr skolem(FormulaPtr formula, Signature& s)` – transformiše formulu u Skolemovu normalnu formu;
- `NormalForm dnf(const FormulaPtr& f)` – prevodi formulu (koja je u NNF) u DNF;

- `FormulaPtr substitute(FormulaPtr f, const std::string&v, TermPtr t)` – zamenjuje sva pojavljivanja promenljive v u formuli termom t;
- `FormulaPtr removeUniversal(FormulaPtr f)` – uklanja sve univerzalne kvantifikatore iz formule.

Dodatno su implementirane pomoćne funkcije za upravljanje tipovima, ispis, proveru signatura, zamenu, evaluaciju i pojednostavljivanje formula.

4.1.2 `interpretation.h` i `interpretation.cpp`

Ovi fajlovi definišu okvir za semantiku logike prvog reda. U fajlu `interpretation.h` nalaze se deklaracije struktura koje predstavljaju jezik (`Signature`) i $\mathcal{L}$-strukturu (`LStructure`). Ključne komponente su prikazane u Listingu 2.

```
//sadrzi funkcije,relacije i njihove arnosti
struct Signature{
 std::map<std::string, unsigned> rel;
 std::map<std::string, unsigned> fun;
 std::string getUniqueSymbol(unsigned arity) {
        static unsigned uniqueCounter = 0;
        std::string uniqueSymbol;
        do {
            uniqueSymbol = "f" + std::to_string(++uniqueCounter);
        } while(fun.find(uniqueSymbol) != fun.end());
        fun[uniqueSymbol] = arity;
        return uniqueSymbol;
    }

};

using Domain= std::set<unsigned>;
using Valuation=std::map<std::string,unsigned>;

using Function = std::function<unsigned(const std::vector<unsigned>&)>;
using Relation = std:: function<bool(const std::vector<unsigned>&)>;

struct LStructure
{
 Signature signature;
    Domain domain;
    std::map<std::string, Function> functions;
    std::map<std::string, Relation> relations;
};
```

Listing 2: Strukture za semantiku: jezik i $\mathcal{L}$-struktura.

4.1.3 `herbrand.h` i `herbrand.cpp`

Ovi fajlovi sadrže deklaraciju i implementaciju strukture koja predstavlja Erbranov univerzum za datu formulu. Struktura Herbrand, definisana u `herbrand.h`, obuhvata formulu za koju se univerzum

konstruiše, njenu signaturu, skup imena funkcijskih simbola iz signature (uključujući i konstante, koje se tretiraju kao funkcije arnosti 0), pomoćnu konstantu (koja se koristi ako formula ne sadrži ni jednu konstantu), kao i skup termova koji čine Herbrandov univerzum. Struktura Herbrand i ključne funkcije prikazane su u Listingu 3.

```
struct Herbrand
{
Signature signature;
FormulaPtr formula;
std::set<std::string> functions;
std::string constant;
std::set<TermPtr> universe;
};
void generatePermutationsWithRepetition(const std::set<TermPtr>&
    inputSet, int r, std::vector<TermPtr>& current, std::vector<std::
    vector<TermPtr>>& result);

Herbrand generate(Signature& s,FormulaPtr f);
Herbrand nextLevel(Herbrand h);
```

Listing 3: Struktura Herbrand i ključne funkcije za rad sa Erbranovim univerzumom.

Ključne funkcije su:

- `Herbrand generate (Signature& s, FormulaPtr f)` – inicijalizuje nulti nivo Herbrandovog univerzuma koristeći konstante iz formule ili dodaje pomoćnu konstantu ako nema konstanti;
- `Herbrand nextLevel (Herbrand h)` – proširuje univerzum dodavanjem složenijih termova;
- `generatePermutationsWithRepetition(const std::set<TermPtr> &inputSet, int r, std::vector<TermPtr>& current, std::vector <std::vector<TermPtr>>& result)` – pomoćna funkcija koja generiše sve uredjene nizove termova zadate dužine radi formiranja argumenata za nove funkcijske termove.

4.1.4 `gilmore.hpp` i `gilmore.cpp`

Ovi fajlovi sadrže implementaciju **Gilmorove procedure**, u skladu sa koracima detaljno opisanim u prethodnim poglavljima.

Ključna funkcija je:

- `void gilmore(Signature& s, FormulaPtr f)` – primenjuje Gilmorovu proceduru na datu formulu i signaturu.

Ovde je važno napomenuti da je zbog eksponencijalnog rasta broja instanci, broj iteracija je ograničen radi efikasnosti i izbegavanja beskonačnog izvršavanja.

4.1.5 main.cpp

Ovo je fajl u kome inicijalizujemo test primere i pokrećemo proceduru.

4.2 Uputstvo za prevodjenje i pokretanje

- U terminalu pozicionirajte na željeni direktorijum i kloniratjte repozitorijum:

  ```
  git clone https://github.com/domeGIT/ar-gilmore
  ```

- Preuzmite kompilator (g++) i ažurirajte sistem:

  ```
  sudo apt-get update
  sudo apt-get install g++
  ```

- Kompilirajte program i pokrenite:

  ```
  make
  ./gilmore
  ```

5 Zaključak

Ovaj seminarski rad pruža detaljan prikaz Gilmorove procedure koja se smatra jednim od prvih automatskih dokazivača teorema [1]. Pored teorijskog okvira na kome se zasniva, rad obuhvata detaljan prikaz algoritma procedure sa konkretnim primerom, kao i opis njene C++ implementacije.

Ipak, kao što je i sam Gilmore istakao u svom radu, procedura se suočava sa ozbiljnim praktičnim ograničenjima [6]. Njena efikasnost značajno opada usled eksponencijalnog rasta broja baznih instanci pri svakom proširenju Erbranovog univerzuma, što dovodi do prekoračenja raspoložive memorije [6].

Uprkos ograničenjima, njen istorijski značaj je izuzetan. Gilmorova procedura je pokazala da se na računaru može realizovati logičko zaključivanje, postavljajući temelj za razvoj savremenih dokazivača teorema, a naročito metode rezolucije sa unifikacijom [1]. Danas, iako prevazidjena u praktičnoj primeni, procedura je zadržala važnu edukativnu ulogu i ostala nezaobilazan deo obuke u oblasti automatskog rezonovanja [5].

Literatura

[1] P. Janičić, *Matematička logika u računarstvu.* (4th ed.) (Matematički fakultet, Univerzitet u Beogradu, Beograd, 2008).

[2] D. Hilbert and W. Ackermann, *Principles of mathematical logic.* (Chelsea publishing company, New York, 1928).

[3] A. Church, An unsolvable problem of elementary number theory, *American Journal of Mathematics*, Vol. 58, pp. 345, (1936).

[4] A.M. Turing, On Computable Numbers, with an Application to the Entscheidungsproblem, *Proceedings of the London Mathematical Society*, Vol. 42, pp. 230, (1936).

[5] M. Banković, *Automatsko rezonovanje – beleške sa predavanja Rezonovanje u logici prvog reda.* (Matematički fakultet, Univerzitet u Beogradu, Beograd, 2024/25).

[6] P. C. Gilmore, A proof method for quantification theory, *IBM J. Research Dev.*, Vol. 4, pp. 28, (1960).