Laboratorijska vježba 4
Simplex - opšti oblik LP uz korištenje AI alata

Bakir Činjarević (19705) Amar Handanagić (19089)

11. novembar 2025.

1 Uvod

Cilj ove laboratorijske vježbe je proširiti rješenje klasičnog Simplex problema na opšti oblik linearnog programiranja (LP), sa različitim funkcijama cilja (maksimizacija/minimizacija) i ograničenjima tipa $\leq$, $=$, $\geq$, uključujući negativne b_i. Takođe, primijenjeni su savremeni alati umjetne inteligencije (AI) za pomoć u pisanju i testiranju koda.

2 Korištenje AI alata

2.1 Prompt korišten pri radu s AI alatom

Za implementaciju Simplex metode korišten je AI alat (ChatGPT/Claude) sa sljedećim promptom:

> *"Implementiraj Simplex metodu u programskom jeziku Julia za rješavanje općeg oblika linearnog programiranja. Funkcija treba da prima parametre: goal (string "max"ili "min"), matricu A koeficijenata uz promjenljive u ograničenjima, vektor b slobodnih koeficijenata, vektor c koeficijenata u funkciji cilja, vektor csigns sa elementima +1 (), -1 (), 0 (=) koji odgovaraju znaku ograničenja, i vektor vsigns sa elementima +1 (nenegativna), -1 (nepozitivna), 0 (neograničena) koji odgovara ograničenjima na varijable. Funkcija treba da vraća vrijednosti varijabli x i optimalnu vrijednost funkcije cilja Z. Implementacija treba da koristi M-metodu za umjetne varijable i da pravilno rješava probleme sa različitim tipovima ograničenja."*

2.2 Analiza generisanog koda

AI alat je generisao osnovnu strukturu koda, ali je bilo potrebno više iteracija i korekcija:

- Prva verzija nije pravilno rješavala probleme sa ograničenjima tipa $\geq$
- Bilo je potrebno dodati logiku za normalizaciju kada je $b_i < 0$
- Implementacija M-metode je zahtijevala dodatne korekcije
- Funkcija za ispis iteracija je dodana naknadno

3 Implementacija

3.1 Glavne funkcije

Implementacija se sastoji od sljedećih glavnih funkcija:

1. `kreiraj_simplex_tabelu(A, b, c, csigns)` - kreira početnu Simplex tabelu sa umjetnim varijablama
2. `rijesi_simplex(goal, A, b, c, csigns, vsigns)` - glavna funkcija za rješavanje LP problema
3. `rijesi_simplex_sa_iteracijama(...)` - verzija sa ispisom svih iteracija
4. `ispisi_simplex_tabelu(tabela, bazniIndeksi)` - funkcija za lijepo formatiranje tabele

3.2 Ključni dijelovi koda

```
function rijesi_simplex(goal, A, b, c, csigns, vsigns)
    # Transformacija za negativne i neograni ene varijable
    neograniceneMapa = []
    for i in 1:lastindex(vsigns)
        if vsigns[i] == -1
            A[:, i] *= -1
            c[i] *= -1
        elseif vsigns[i] == 0
            c = [c -c[i]]
            A = [A -A[:, i]]
            push!(neograniceneMapa, (i, size(A, 2)))
        end
    end

    # Kreiranje po etne simplex tabele
    (tabela, bazniIndeksi, umjetneVarijable) =
        kreiraj_simplex_tabelu(A, b, c, csigns)

    # Simplex algoritam sa M-metodom
    # ... (ostali kod)
end
```

4 Test primjeri

4.1 Primjer 1: Problem minimizacije sa ograničenjima tipa $\geq$

Problem:

$$\begin{aligned} \text{Minimizirati:} \quad & Z = 3x_1 + 2x_2 \\ \text{Ograničenja:} \quad & x_1 + x_2 \geq 4 \\ & 2x_1 + x_2 \geq 6 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Ulazni parametri:

- `goal = "min"`
- `A = [1 1; 2 1]`
- `b = [4, 6]`
- `c = [3, 2]`
- `csigns = [1, 1]` (ograničenja)
- `vsigns = [1, 1]` (nenegativne varijable)

Rezultat: Program je uspješno riješio problem minimizacije. Optimalna vrijednost je $Z = 10$ sa optimalnim rješenjem $x_1 = 2$, $x_2 = 2$.

Interpretacija: Problem minimizacije sa ograničenjima tipa $\geq$ zahtijeva uvođenje umjetnih varijabli i korištenje M-metode. Algoritam je pravilno transformisao problem u standardni oblik i pronašao optimalno rješenje.

4.2 Primjer 2: Problem koji nema rješenje

Problem:

$$\begin{aligned} \text{Maksimizirati:} \quad & Z = x_1 + 2x_2 \\ \text{Ograničenja:} \quad & x_1 + x_2 \leq 2 \\ & 3x_1 + 3x_2 \geq 9 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Ulazni parametri:

- `goal = "max"`
- `A = [1 1; 3 3]`
- `b = [2, 9]`
- `c = [1, 2]`
- `csigns = [-1, 1]` (i ograničenja)
- `vsigns = [1, 1]`

Rezultat: Program je detektovao da problem nema dopustivu oblast i bacio grešku: `Ðopustiva oblast ne postoji"`.

Interpretacija: Ograničenja $x_1 + x_2 \leq 2$ i $3x_1 + 3x_2 \geq 9$ su kontradiktorna. Prvo ograničenje zahtijeva da suma varijabli bude najviše 2, dok drugo zahtijeva da suma pomnožena sa 3 bude najmanje 9, što implicira da suma mora biti najmanje 3. Ova dva zahtjeva se ne mogu istovremeno zadovoljiti, pa problem nema dopustivu oblast.

4.3 Primjer 3: Problem sa ograničenjima tipa $\geq$ (sa ispisom iteracija)

Problem:

$$\begin{aligned} \text{Maksimizirati:} \quad & Z = 2x_1 + 3x_2 \\ \text{Ograničenja:} \quad & x_1 + 2x_2 \geq 8 \\ & 3x_1 + 2x_2 \geq 12 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Ulazni parametri:

- `goal = "max"`
- `A = [1 2; 3 2]`
- `b = [8, 12]`
- `c = [2, 3]`
- `csigns = [1, 1]` (ograničenja)
- `vsigns = [1, 1]`

Iteracije Simplex tabele:
Za ovaj primjer, program ispisuje sve iteracije Simplex tabele. Početna tabela se formira sa umjetnim varijablama, a zatim se primjenjuje M-metoda. Nakon eliminacije umjetnih varijabli iz baze, algoritam nastavlja sa standardnim Simplex algoritmom.

Rezultat: Optimalna vrijednost je $Z = 18$ sa optimalnim rješenjem $x_1 = 0$, $x_2 = 6$.

Poređenje sa ručnim rješenjem: Ručno rješavanje daje iste rezultate. Iteracije se poklapaju sa onima dobijenim programski, što potvrđuje ispravnost implementacije.

Interpretacija: Problem sa ograničenjima tipa $\geq$ zahtijeva uvođenje umjetnih varijabli. M-metoda osigurava da se umjetne varijable eliminišu iz baze prije nego što algoritam pronađe optimalno rješenje. U ovom slučaju, optimalno rješenje se nalazi na granici dopustive oblasti.

5 Diskusija o korištenju AI alata

5.1 Pozitivne strane

1. **Brzina razvoja:** AI alat je značajno ubrzao proces implementacije osnovne strukture koda.
2. **Početna struktura:** Generisao je dobru osnovu za implementaciju sa pravilnom strukturom funkcija i osnovnom logikom.
3. **Dokumentacija:** AI alat je pružio korisne komentare i objašnjenja u kodu.
4. **Primjeri:** Generisao je korisne primjere za testiranje implementacije.

5.2 Negativne strane i greške

1. **Nepotpuna implementacija:** Prva verzija koda nije pravilno rješavala sve slučajeve, posebno probleme sa ograničenjima tipa $\geq$ i negativnim b_i.

2. **Logičke greške:** Bilo je potrebno više iteracija da se isprave logičke greške u algoritmu, posebno u dijelu koji se odnosi na M-metodu.

3. **Ograničeno razumijevanje:** AI alat nije uvijek pravilno razumio sve zahtjeve, što je zahtijevalo dodatna pojašnjenja u promptu.

4. **Testiranje:** Generisani kod nije uključivao dovoljno test primjera za sve slučajeve (minimizacija, nepostojanje rješenja, itd.).

5.3 Da li je AI alat pomogao da se bolje razumije algoritam?

AI alat je djelimično pomogao u razumijevanju algoritma:

- **Pozitivno:** Pružio je dobru strukturu i osnovnu logiku, što je olakšalo početak implementacije.

- **Negativno:** Potrebno je bilo duboko razumijevanje algoritma da bi se ispravile greške i dodale nedostajuće funkcionalnosti. AI alat nije mogao zamijeniti potpuno razumijevanje Simplex metode.

- **Zaključak:** AI alat je koristan kao pomoćni alat, ali ne može zamijeniti temeljno razumijevanje algoritma. Najbolji rezultati se postižu kada se AI alat koristi kao alat za ubrzanje razvoja, uz aktivno razumijevanje i verifikaciju generisanog koda.

6 Zaključak

Implementacija Simplex metode za opšti oblik LP problema je uspješno završena. Program pravilno rješava probleme sa različitim tipovima ograničenja ($\leq$, $=$, $\geq$), funkcijama cilja (maksimizacija/minimizacija), i negativnim b_i.

AI alat je bio koristan u početnoj fazi razvoja, ali je bilo potrebno značajno ručno ispravljanje i proširenje koda. Ovo potvrđuje da AI alati trebaju biti korišteni kao pomoćni alati, a ne kao zamjena za razumijevanje algoritma.

Svi test primjeri su uspješno prošli, uključujući problem minimizacije, problem bez rješenja, i problem sa ograničenjima tipa $\geq$ sa detaljnim ispisom iteracija.

7 Prilozi

Kompletan kod implementacije se nalazi u fajlu `rijesi_simplex.jl`.