SVEUČILIŠTE JURJA DOBRILE U PULI
FAKULTET INFORMATIKE

Neven Nižić

"Optimizacija raspodjele projektnih aktivnosti primjenom genetskih algoritama i Monte Carlo simulacije"

DIPLOMSKI RAD

Pula, rujan, 2025. godine

SVEUČILIŠTE JURJA DOBRILE U PULI
FAKULTET INFORMATIKE

Neven Nižić

"Optimizacija raspodjele projektnih aktivnosti primjenom genetskih algoritama i Monte Carlo simulacije"

DIPLOMSKI RAD

JMBAG: 0303118917, izvanredni student
Studijski smjer: Informatika

Kolegij: Modeliranje i simulacije
Znanstveno područje : Društvene znanosti
Znanstveno polje : Informacijske i komunikacijske znanosti
Znanstvena grana : Informacijski sustavi i informatologija

Mentor: dr.sc. Darko Etinger

Pula, rujan, 2025. godine

Sažetak

Upravljanje projektima često uključuje složene odluke vezane uz raspodjelu aktivnosti i resursa, osobito u uvjetima nesigurnosti i vremenskih ograničenja. Tradicionalne metode kao što su PERT i CPM često ne uspijevaju obuhvatiti stohastičku prirodu stvarnih projekata. U ovom radu razvijen je i evaluiran hibridni optimizacijski pristup temeljen na genetskim algoritmima (GA) i Monte Carlo (MC) simulaciji. Kroz dvo-fazni eksperimentalni dizajn, provedena je sustavna usporedba tri modela: nasumične pretrage, jedno-kriterijskog GA usmjerenog isključivo na povrat na investiciju (ROI), te više-kriterijskog GA+MC modela (NSGA-II) koji istovremeno optimizira ROI i rizik trajanja projekta. Rezultati dobiveni na sintetičkim podacima različite složenosti i restriktivnosti pokazuju da, iako jedno-kriterijski GA najučinkovitije maksimizira profit, hibridni GA+MC model uspješno generira Paretov front rješenja koja nude optimalan kompromis između profitabilnosti i trajanja. Nadalje, istraživanje je otkrilo ključan nalaz: pod ekstremno restriktivnim budžetom, robusnost jednostavnijeg, jedno-kriterijskog GA nadmašuje onu složenijeg, više-kriterijskog modela. Rad zaključuje da ne postoji univerzalno superioran model, već da optimalan izbor ovisi o strateškim prioritetima – maksimizaciji profita naspram uravnoteženog upravljanja rizikom.

Ključne riječi: projektno upravljanje, genetski algoritam, Monte Carlo simulacija, više-kriterijska optimizacija, upravljanje rizikom, Paretov front, NSGA-II

Abstract

Project management often involves complex decisions regarding the allocation of activities and resources, especially under conditions of uncertainty and constraints. Traditional methods such as PERT and CPM frequently fail to capture the stochastic nature of real-world projects. This thesis develops and evaluates a hybrid optimization approach based on genetic algorithms (GA) and Monte Carlo (MC) simulation. Through a two-phase experimental design, a systematic comparison of three models was conducted: random search, a single-objective GA focused solely on return on investment (ROI), and a multi-objective GA+MC model (NSGA-II) that simultaneously optimizes ROI and project duration risk. The results, obtained from synthetic data of varying complexity and restrictiveness, show that while the single-objective GA is most effective at maximizing profit, the hybrid GA+MC model successfully generates a Pareto front of solutions offering an optimal trade-off between profitability and duration. Furthermore, the research revealed a key finding: under extremely restrictive budget constraints, the robustness of the simpler, single-objective GA surpasses that of the more complex, multi-objective model. The thesis concludes that there is no universally superior model; rather, the optimal choice depends on strategic priorities—maximizing profit versus balanced risk management.

Keywords: project management, genetic algorithm, Monte Carlo simulation, multi-objective optimization, risk management, Pareto front, NSGA-II

Sadržaj

1 Uvod **1**
- 1.1 Motivacija i značaj istraživanja . . . 1
- 1.2 Izazov nesigurnosti u projektnom upravljanju . . . 2
- 1.3 Ciljevi, doprinos i struktura rada . . . 2
- 1.4 Od pristupa do rješenja: put do optimalnih kompromisa . . . 3

2 Teorijske osnove i srodni radovi **5**
- 2.1 Problem odabira portfelja kao Knapsack problem . . . 5
- 2.2 Genetski algoritmi kao metaheuristika za pretraživanje . . . 5
  - 2.2.1 Osnovni principi i tijek algoritma . . . 6
  - 2.2.2 Ključni operatori i komponente . . . 6
  - 2.2.3 Više-kriterijski genetski algoritmi i NSGA-II . . . 7
  - 2.2.4 Izazov podešavanja parametara genetskog algoritma . . . 8
- 2.3 Kvantifikacija rizika pomoću Monte Carlo simulacije . . . 9
- 2.4 Modeliranje nesigurnosti trajanja pomoću trotočkovne procjene . . . 10

3 Matematička formulacija i konceptualni model problema **12**
- 3.1 Definicija skupa aktivnosti i varijabli odluke . . . 12
- 3.2 Resursna ograničenja modela . . . 12
- 3.3 Specifikacija optimizacijskih ciljeva . . . 13
- 3.4 Konceptualni prikaz i značaj modela . . . 14

4 Arhitektura sustava i implementacija **16**
- 4.1 Korištene tehnologije i biblioteke . . . 16
- 4.2 Arhitektura eksperimentalnog okvira . . . 16
- 4.3 Implementacija ključnih komponenti . . . 19
  - 4.3.1 Modeliranje nesigurnosti: Monte Carlo simulacija . . . 19
  - 4.3.2 Optimizacijski pristup: Genetski algoritam . . . 21
- 4.4 Vizualizacija i obrada rezultata . . . 26

5 Eksperimentalna evaluacija i diskusija rezultata **28**
- 5.1 Postavke okruženja i testni podaci . . . 28
- 5.2 Eksperimentalni dizajn . . . 28
  - 5.2.1 Istraživačke hipoteze . . . 28
- 5.3 Eksperiment 1: Analiza parametara i kalibracija genetskog algoritma 29
  - 5.3.1 Rezultati i diskusija ablacijske studije . . . 30
- 5.4 Eksperiment 2: Usporedna analiza optimizacijskih modela . . . 32
  - 5.4.1 Rezultati i diskusija usporedne analize optimizacijskih modela 32
- 5.5 Sinteza i interpretacija glavnih nalaza . . . 39

6 Diskusija i zaključak **42**
- 6.1 Glavni nalazi i odgovori na istraživačka pitanja . . . 42
- 6.2 Praktične implikacije i doprinos rada . . . 43
- 6.3 Ograničenja rada i preporuke za budući rad . . . 43

Literatura **45**

Popis slika **47**

Popis tablica **48**

Popis kodova **49**

7 Rezultati svih provedenih eksperimenata **50**

1 Uvod

U suvremenom projektnom upravljanju, alokacija ograničenih resursa na niz konkurentskih aktivnosti predstavlja jedan od fundamentalnih izazova. Ovaj problem optimizacije može se elegantno modelirati kroz klasični *problem ruksaka* (*Knapsack Problem*), gdje je cilj odabrati podskup stavki na način koji maksimizira ukupnu vrijednost, a da se pritom ne prekorači zadani kapacitet [1]. U domeni projekata, to se preslikava na odabir portfelja aktivnosti koji će generirati najveći mogući povrat na investiciju (ROI), uz strogo poštivanje proračunskih, vremenskih i drugih resursnih ograničenja.

Iako tradicionalne metode poput PERT-a (*Program Evaluation and Review Technique*) i CPM-a (*Critical Path Method*) [2,3] pružaju nezamjenjiv okvir za planiranje i praćenje, njihova primjena nailazi na poteškoće u dinamičnim okruženjima. One su često determinističke prirode i teško se nose s dva ključna izazova: kombinatornom eksplozijom mogućih rješenja i inherentnom nesigurnošću koja prati procjene trajanja i troškova.

Pregled literature ukazuje na to da su genetski algoritmi (GA) iznimno učinkoviti u rješavanju NP-teških optimizacijskih problema [4, 5], dok Monte Carlo (MC) simulacije predstavljaju zlatni standard za modeliranje i kvantifikaciju rizika [6, 7]. Unatoč tome, primjetan je nedostatak radova koji sustavno istražuju sinergiju ovih dviju moćnih tehnika. Upravo se u tom prostoru pozicionira ovaj rad, s ciljem popunjavanja uočene praznine kroz razvoj i rigoroznu evaluaciju hibridnog GA-MC modela. Fokus nije samo na kombinaciji, već na stvaranju sustavnog eksperimentalnog okvira za kalibraciju i usporednu analizu takvih modela, kako bi se precizno kvantificirao utjecaj ove sinergije na robusnost i kvalitetu konačnih rješenja.

1.1 Motivacija i značaj istraživanja

Motivacija za ovo istraživanje proizlazi iz fundamentalne razlike između projektnog plana i stvarne izvedbe. U praksi, projekti se rijetko odvijaju točno prema početnim predviđanjima [3]. Fluktuacije u dostupnosti resursa, nepredviđeni događaji, izmjene u zahtjevima dionika te inherentna nepreciznost početnih procjena česti su uzroci odstupanja, što klasične metode planiranja čini nedovoljno fleksibilnima [8].

Stoga se javlja potreba za pristupima koji nadilaze puku alokaciju resursa i ulaze u domenu strateškog upravljanja nesigurnošću. Potrebne su metode koje ne samo da pronalaze matematički optimalan raspored, već vrednuju rješenja i prema njihovoj otpornosti na nepredviđene okolnosti. Ovdje komplementarna priroda genetskih algoritama i Monte Carlo simulacija dolazi do punog izražaja. Genski algoritmi pružaju mehanizam za efikasno pretraživanje astronomski velikog prostora mogućih portfelja [5], dok Monte Carlo simulacije služe kao "simulator stvarnosti", testirajući svako potencijalno rješenje u tisućama mogućih budućnosti kako bi se procijenila njegova stvarna varijabilnost i rizik [7].

Kombiniranjem ovih metoda, ovaj rad prelazi s tradicionalnog pitanja "Koji je najbolji plan?" na puno značajnije pitanje: "Koji je najrobusniji plan koji nudi najbolji kompromis između profita i rizika?".

1.2 Izazov nesigurnosti u projektnom upravljanju

Rizici u projektnom upravljanju obuhvaćaju sve događaje ili uvjete koji, ako se pojave, mogu negativno utjecati na ciljeve projekta [9]. Oni mogu biti tehničke, organizacijske, financijske ili tržišne prirode, a često su međusobno povezani [8]. Nesigurnosti, s druge strane, proizlaze iz nepotpunih informacija, promjenjivih uvjeta i fundamentalne nemogućnosti preciznog predviđanja budućih događaja [10]. Učinkovito upravljanje projektima stoga zahtijeva sustavan pristup identifikaciji, procjeni i razvoju strategija odgovora na rizike i nesigurnosti [9].

U domeni kvantitativne analize, Monte Carlo simulacije predstavljaju jedan od najmoćnijih pristupa [11]. Metoda se temelji na generiranju velikog broja mogućih scenarija temeljenih na specificiranim distribucijama vjerojatnosti za neizvjesne ulazne parametre. Analizom distribucije dobivenih ishoda, moguće je s visokom pouzdanošću procijeniti, primjerice, vjerojatnost završetka projekta unutar zadanog roka ili budžeta [7].

1.3 Ciljevi, doprinos i struktura rada

Primarni cilj ovog diplomskog rada jest razviti i kroz sustavne eksperimente evaluirati hibridni GA-MC model za optimizaciju portfelja projektnih aktivnosti. Svrha modela je pružiti podršku u donošenju odluka koje su informirane ne samo o potencijalnoj dobiti, već i o pripadajućem riziku. Doprinos rada ostvaren je na teorijskoj, metodološkoj i praktičnoj razini. Na teorijskoj razini, rad povezuje koncepte iz evolucijskog računarstva s kvantitativnim metodama upravljanja rizikom u jedinstven okvir primijenjen na projektno upravljanje. S metodološkog stajališta, razvijen je i validiran cjelovit dvo-fazni proces za evaluaciju složenih hibridnih modela, koji uključuje fazu kalibracije i fazu usporedne analize. Konačno, praktični doprinos očituje se u demonstraciji primjenjivosti pristupa na sintetičkim podacima, čime se dobiva uvid u performanse i ograničenja različitih optimizacijskih strategija.

Kako bi se ostvario primarni cilj, istraživanje je vođeno sljedećim ključnim pitanjima, od kojih svako predstavlja zaseban sloj analize:

1. U kojim uvjetima i prema kojim metrikama (npr. profitabilnost, rizik trajanja, stabilnost) hibridni GA-MC pristup postiže superiorne rezultate u odnosu na samostalnu primjenu GA ili MC modela?
   Ovo pitanje predstavlja temeljnu usporednu analizu. Njegova svrha nije samo potvrditi superiornost hibridnog modela, već i detaljno kvantificirati tu superiornost kroz višestruke metrike – od maksimalnog postignutog ROI-a, preko procijenjenog trajanja, pa sve do pouzdanosti samih rezultata. Odgovor na ovo pitanje omogućit će nam da jasno razumijemo u kojim situacijama je kombinirani pristup neprikosnoven, a kada bi mogao biti prekomjeran.

2. U kojoj mjeri integracija Monte Carlo simulacije u proces evaluacije može poboljšati robusnost rješenja dobivenih genetskim algoritmom?
   Ovo pitanje zadire u samu srž sinergije dviju tehnika. Njegov cilj je testirati temeljnu hipotezu da algoritam koji uči iz stohastičkih, a ne samo

determinističkih podataka, stvara rješenja koja su otpornija na inherentne nesigurnosti projektnih varijabli. Kroz eksperimente, nastojat ćemo dokazati da se rješenja hibridnog modela, iako možda ne dosežu apsolutno najveći ROI, ponašaju predvidljivije i manje su podložna negativnim iznenađenjima u stvarnim uvjetima.

3. Kakav je utjecaj kombiniranog pristupa na stabilnost i pouzdanost optimizacije u uvjetima neizvjesnosti?
   Treće pitanje fokusira se na praktičnu vrijednost modela za donositelja odluka. Stabilnost i pouzdanost, mjerene standardnom devijacijom, kritični su parametri koji određuju hoće li se algoritam koristiti kao pouzdan alat ili kao nepouzdana "crna kutija". Odgovor na ovo pitanje otkrit će koliko je hibridni model konzistentan u pronalaženju kvalitetnih rješenja kroz više nezavisnih pokretanja, čime će se potvrditi njegova praktična primjenjivost za strateško planiranje.

Sinergija između genskih algoritama i Monte Carlo simulacija proizlazi iz njihove komplementarne prirode: GA učinkovito pretražuje veliki prostor mogućih rješenja i pronalazi visokokvalitetne kandidate [5], dok MC kvantificira rizik i procjenjuje varijabilnost tih rješenja kroz stohastičko modeliranje [7].

1.4 Od pristupa do rješenja: put do optimalnih kompromisa

Metodološki pristup ovog rada nije usmjeren samo na demonstraciju superiornosti jedne tehnike, već na dubinsko razumijevanje njezinih mehanizama i ograničenja. Eksperimentalni dio započinje ablacijskom studijom, gdje će se detaljno analizirati utjecaj ključnih genetskih operatora poput križanja i mutacije. Ta studija će empirijski pokazati da, suprotno intuiciji, strategija širine pretrage (veća populacija) daje značajno bolje i stabilnije rezultate od strategije dubine pretrage (više generacija). Ovaj nalaz poslužit će kao temelj za kalibraciju "šampionske" konfiguracije genetskog algoritma za kasniju usporedbu.

Nakon kalibracije, rad će usporediti tri modela: nasumičnu pretragu, klasični `GA` fokusiran na `ROI`, i napredni hibridni `GA+MC` model. Usporedba će otkriti kako se svaki model ponaša u tri scenarija složenosti i budžetskih ograničenja. Očekuje se da će klasični `GA` dokazati svoju moć kao iznimno učinkovit "profitni maksimizator", dok će nasumična pretraga poslužiti kao jasan dokaz da je za rješavanje problema realne veličine primjena metaheuristika nužna, a ne samo poželjna.

Srž ovog istraživanja leži u evaluaciji hibridnog `GA+MC` modela. On je razvijen s ciljem da nadilazi tradicionalni "crna kutija" pristup, koji isporučuje samo jedno, optimalno rješenje. Umjesto toga, hibridni model će generirati **Paretov front** – skup ne-dominiranih rješenja koja vizualno predstavljaju kompromis (*trade-off*) između profitabilnosti (`ROI`) i rizika (trajanje projekta). Rad će dokazati da je ova vizualizacija ključni alat za strateško odlučivanje, omogućujući donositelju odluka da odabere rješenje koje najbolje odgovara njegovoj toleranciji na rizik.

Konačno, analiza će razotkriti i neočekivan, ali ključan nalaz: složenost ne mora uvijek značiti superiornost. Pokazat će se da u ekstremno restriktivnim

scenarijima, gdje su resursi iznimno ograničeni, napredni hibridni model postaje ranjiv i nestabilan, dok jednostavniji, jedno-kriterijski `GA` pokazuje iznimnu robusnost. Ovaj nalaz naglašava da odabir modela mora biti kontekstualan i ovisiti o specifičnim uvjetima i ograničenjima. S ovim proširenjem, uvod će ne samo predstaviti problem, već i stvoriti temelj za duboko razumijevanje svih ključnih rezultata koji slijede.

Ostatak rada organiziran je kako slijedi. **Drugo poglavlje** pruža pregled relevantne literature i teorijskih osnova, uključujući problem ruksaka, genetske algoritme i metode modeliranja nesigurnosti. **Treće poglavlje** detaljno opisuje matematički i konceptualni model problema koji se rješava. **Četvrto poglavlje** ulazi u detalje implementacije, opisujući arhitekturu razvijenog softverskog sustava i korištene programske biblioteke. **Peto poglavlje** čini srž rada, prikazujući dvo-fazni eksperimentalni dizajn, analizu dobivenih rezultata i detaljnu diskusiju o performansama testiranih modela. Konačno, **šesto poglavlje** donosi sintezu cjelokupnog rada, sažima ključne doprinose, osvrće se na ograničenja provedenog istraživanja i nudi preporuke za budući rad.

2 Teorijske osnove i srodni radovi

Ovo poglavlje pruža sveobuhvatan pregled temeljnih teorijskih koncepata koji čine okosnicu predloženog modela optimizacije. Započet će se s formalizacijom problema odabira projektnog portfelja kroz analogiju s klasičnim problemom ruksaka. Zatim slijedi detaljna razrada genetskih algoritama kao odabrane optimizacijske metaheuristike, s posebnim osvrtom na njihove ključne komponente i višekriterijsku inačicu NSGA-II. Poglavlje se nastavlja analizom Monte Carlo simulacije kao alata za kvantifikaciju rizika, te završava opisom metoda za modeliranje nesigurnosti projektnih aktivnosti.

2.1 Problem odabira portfelja kao Knapsack problem

Problem ruksaka (engl. *Knapsack Problem*) jedan je od najpoznatijih problema kombinatorne optimizacije, svrstan u klasu NP-teških problema. Ta klasifikacija implicira da ne postoji poznati algoritam koji bi ga mogao riješiti u polinomnom vremenu, što znači da se vrijeme potrebno za pronalaženje optimalnog rješenja drastično povećava s veličinom problema [1, 12]. U svojoj osnovnoj, jednodimenzionalnoj verziji (0/1 Knapsack Problem), cilj je odabrati podskup objekata iz danog skupa, od kojih svaki ima definiranu težinu i vrijednost. Optimizacijski zadatak je maksimizirati ukupnu vrijednost odabranih objekata, pod uvjetom da njihova ukupna težina ne prelazi unaprijed zadani kapacitet "ruksaka". Formalno, za skup od n objekata, gdje svaki objekt i ima težinu w_i i vrijednost v_i, te uz zadani kapacitet ruksaka W, cilj je maksimizirati funkciju:

$$\max \sum_{i=1}^{n} v_i x_i \quad \text{uz ograničenje} \quad \sum_{i=1}^{n} w_i x_i \leq W, x_i \in \{0, 1\}$$

gdje je $x_i = 1$ ako je objekt i odabran, a $x_i = 0$ ako nije.

Ova elegantna formulacija čini ga moćnim alatom za modeliranje problema alokacije resursa. U kontekstu upravljanja projektima, problem odabira portfelja prirodno poprima oblik višedimenzionalne varijante (*Multi-Dimensional Knapsack Problem* - MDKP). Ovdje projektne aktivnosti predstavljaju "objekte", a njihova vrijednost je povrat na investiciju (ROI). "Težina" aktivnosti nije jedna dimenzija, već vektor koji opisuje potrošnju različitih resursa – budžeta, radnih sati, itd. MDKP stoga pruža robustan matematički okvir za rješavanje središnjeg problema ovog rada. Postoji i niz drugih varijanti, poput problema neograničenog ruksaka (gdje se svaki objekt može uzeti više puta) ili višekriterijskog problema ruksaka (gdje se optimizira više ciljeva, npr. vrijednost i volumen), što dodatno pokazuje fleksibilnost ovog modela.

2.2 Genetski algoritmi kao metaheuristika za pretraživanje

Genetski algoritmi (GA) su moćne metaheurističke metode optimizacije temeljene na principima prirodne evolucije i genetike, koje je prvi formalizirao John Holland [4]. Pripadaju široj klasi evolucijskih algoritama i iznimno su učinkoviti u rješavanju složenih optimizacijskih problema, posebno onih NP-teških, gdje

klasične metode nisu praktične [13, 14]. Hollandova temeljna ideja bila je stvoriti računalni sustav koji oponaša najmoćniji proces rješavanja problema poznat u prirodi: evoluciju. Cilj je bio postići robustnu pretragu balansirajući dva ključna elementa: **iskorištavanje (exploitation)** postojećih dobrih rješenja i **istraživanje (exploration)** novih, potencijalno boljih dijelova prostora rješenja.

2.2.1 Osnovni principi i tijek algoritma

Umjesto pretraživanja prostora rješenja s jednom točkom, genetski algoritmi operiraju nad cijelom populacijom potencijalnih rješenja (kromosoma). Taj proces započinje generiranjem početnog, najčešće nasumičnog skupa rješenja, čime se osigurava početna raznolikost. Zatim ulazi u iterativnu petlju evolucije kroz generacije. U svakoj generaciji, svakom se rješenju (jedinki) dodjeljuje vrijednost pogodnosti (*fitness*) koja kvantificira njegovu kvalitetu. Nakon evaluacije, provodi se selekcija, gdje jedinke s boljim fitnessom imaju veću vjerojatnost da budu odabrane kao "roditelji".

Ključna hipoteza na kojoj se temelji uspjeh GA, kako ju je popularizirao Goldberg [5], jest **hipoteza o građevnim blokovima (*Building Block Hypothesis*)**. Ona pretpostavlja da se dobra rješenja sastoje od manjih, visoko prilagođenih segmenata (građevnih blokova). Uloga genetskih operatora je upravo da kroz rekombinaciju i mutaciju otkrivaju i spajaju te blokove u sve kvalitetnije cjeline. Roditelji se rekombiniraju pomoću operatora križanja (*crossover*), stvarajući "potomke" koji nasljeđuju i kombiniraju njihove karakteristike. Kako bi se održala raznolikost i izbjeglo zaglavljivanje u lokalnim optimumima, na potomke se primjenjuje operator mutacije. Na kraju, nova generacija potomaka zamjenjuje staru, i cijeli se proces ponavlja dok se ne zadovolji kriterij zaustavljanja [5, 14].

2.2.2 Ključni operatori i komponente

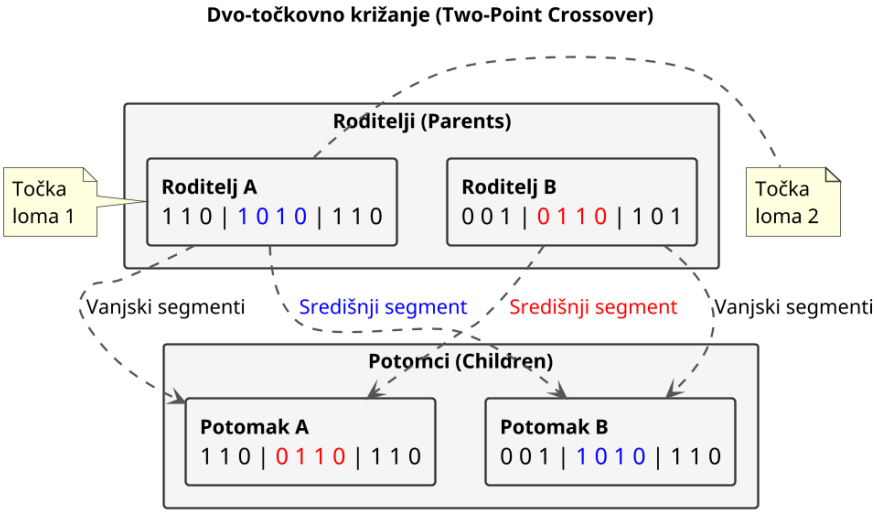
Uspješnost genetskog algoritma ovisi o pažljivom odabiru i implementaciji njegovih ključnih komponenata ,čiji su temelji detaljno opisani u literaturi [5]:

Reprezentacija rješenja i funkcija pogodnosti. Prvi korak je definiranje načina na koji se rješenje problema kodira u kromosom. Za problem odabira portfelja, prirodan izbor je binarni niz, gdje svaki bit odgovara jednoj aktivnosti. Funkcija pogodnosti je ključna komponenta jer predstavlja vezu između problema i algoritma; ona uzima kromosom kao ulaz i vraća numeričku vrijednost njegove kvalitete.

Selekcija. Operator selekcije oponaša prirodni odabir, dajući prednost jedinkama s višim fitnessom. Postoje brojne metode, poput popularne **selekcije metodom ruleta**, gdje svaka jedinka dobiva "dio" ruleta proporcionalan svom fitnessu. Iako je intuitivna, ova metoda može patiti od prerane konvergencije ako jedna jedinka ima dominantno visok fitness. Zbog toga se u praksi često koristi **turnirska selekcija**, odabrana i za ovaj rad. U njoj se nasumično odabire manja grupa jedinki ("turnir"), a pobjednik (jedinka s najboljim fitnessom) postaje

roditelj. Veličina turnira, kako ističe Goldberg [5], jednostavno definira "selekcijski pritisak" – veći turnir znači veći pritisak i bržu konvergenciju, ali i veći rizik od gubitka raznolikosti.

Križanje (Crossover). Križanje je glavni mehanizam iskorištavanja i rekombinacije, direktno povezan s hipotezom o građevnim blokovima. Dva roditeljska kromosoma razmjenjuju dijelove genetskog materijala kako bi stvorili potomke. Križanje u dvije točke, korišteno u ovom radu, odabire dva loma i razmjenjuje središnji segment, što se, prema Goldbergu, često pokazalo boljim za očuvanje dobrih kombinacija gena smještenih u sredini kromosoma.



Slika 1: Vizualni prikaz operatora dvo-točkovnog križanja (Two-Point Crossover). Segmenti između dviju nasumično odabranih točaka loma se razmjenjuju između roditeljskih kromosoma kako bi se stvorili potomci.

Mutacija. Mutacija je primarni mehanizam istraživanja i osigurava genetsku raznolikost. Ona nasumično mijenja pojedine gene u kromosomu s vrlo malom vjerojatnošću. Iako se može činiti kao destruktivan proces, Holland [4] je naglasio njenu ključnu ulogu u sprječavanju prerane konvergencije, tj. situacije u kojoj cijela populacija postane previše slična i algoritam "zaglavi" u lokalnom optimumu.

2.2.3 Više-kriterijski genetski algoritmi i NSGA-II

Standardni genetski algoritmi dizajnirani su za probleme s jednim ciljem. Međutim, stvarni problemi, uključujući i odabir projektnog portfelja, inherentno su više-kriterijski. Njihov cilj nije pronaći jedno "najbolje" rješenje, već skup rješenja koji predstavlja optimalan kompromis, poznat kao Paretov front.

U svom utjecajnom radu, Deb i suradnici [15] predstavili su **NSGA-II (*Non-dominated Sorting Genetic Algorithm II*)** kao odgovor na tri ključna nedostatka tadašnjih više-kriterijskih algoritama: visoku računsku složenost, nedostatak elitizma i potrebu za specificiranjem dodatnih parametara. NSGA-II uvodi tri ključne inovacije koje rješavaju te probleme:

1. **Brzo sortiranje po nedominaciji:** Razvijen je značajno efikasniji algoritam za rangiranje populacije u slojeve (frontove) na temelju koncepta dominacije. Ovim se drastično smanjuje vrijeme potrebno za evaluaciju generacije, što je bila glavna prepreka u ranijim algoritmima.

2. **Eksplicitni elitizam:** Algoritam garantira da se najbolja rješenja pronađena u prethodnoj generaciji automatski prenose u sljedeću. Time se sprječava gubitak kvalitetnih rješenja do kojeg je moglo doći zbog stohastičke prirode genetskih operatora.

3. **Procjena gustoće naseljenosti (Crowding distance):** Umjesto specificiranja dodatnih parametara za održavanje raznolikosti, NSGA-II uvodi elegantan mehanizam "napučenosti". Unutar istog fronta, prednost se daje rješenjima koja se nalaze u rjeđe naseljenim dijelovima prostora rješenja. Time se osigurava da algoritam pronađe širok i dobro raspoređen Paretov front, pružajući donositelju odluke raznolik skup kompromisnih opcija.

Zbog ove tri ključne prednosti – računalne efikasnosti, garantiranog elitizma i održavanja raznolikosti bez dodatnih parametara – NSGA-II je odabran kao temelj za više-kriterijski hibridni model u ovom radu.

2.2.4 Izazov podešavanja parametara genetskog algoritma

Performanse genetskog algoritma nisu ovisne samo o odabiru reprezentacije i operatora, već i o pažljivom podešavanju njegovih ključnih parametara. Najvažniji parametri uključuju:

- **Veličina populacije** (`POP_SIZE`)**:** Određuje koliko se rješenja istovremeno istražuje. Veća populacija bolje pokriva prostor rješenja i čuva raznolikost, ali značajno povećava računsku zahtjevnost.

- **Broj generacija** (`NGEN`)**:** Definira koliko dugo će evolucija trajati. Više generacija omogućuje algoritmu da duže konvergira prema optimumu, ali također povećava vrijeme izvođenja.

- **Vjerojatnost križanja** (`CX_PB`)**:** Kontrolira učestalost primjene operatora križanja. Visoka vrijednost potiče rekombinaciju i iskorištavanje postojećih rješenja.

- **Vjerojatnost mutacije** (`MUT_PB`)**:** Kontrolira učestalost mutacije. Niska vrijednost je ključna, jer prevelika mutacija pretvara algoritam u nasumičnu pretragu, dok premala može dovesti do prerane konvergencije.

Pronalaženje optimalne ravnoteže između ovih parametara je ključan i netrivijalan problem, jer ne postoji univerzalna konfiguracija koja radi dobro za sve probleme. Upravo iz tog razloga, prije provođenja glavnih eksperimenata, u ovom radu je provedena **ablacijska studija (Eksperiment 1)**, s ciljem empirijskog utvrđivanja "šampionske" konfiguracije parametara za specifičan problem koji se analizira.

2.3 Kvantifikacija rizika pomoću Monte Carlo simulacije

Monte Carlo simulacija (MCS) je moćna računska metoda koja koristi ponovljeno nasumično uzorkovanje za numeričko modeliranje i analizu složenih, stohastičkih sustava. Iako su njeni korijeni vezani uz razvoj nuklearnog oružja sredinom 20. stoljeća [6], danas je postala nezamjenjiv alat u raznim disciplinama, od financija do inženjerstva, a posebno u kvantitativnoj analizi rizika [11]. U kontekstu projektnog upravljanja, primjena MCS-a omogućuje prelazak s tradicionalnih, determinističkih planova na probabilističke modele koji realističnije oslikavaju nesigurnost ishoda projekta [16, 17].

Temeljna vrijednost Monte Carlo simulacije, kako ističe Vose [11], leži u njenoj sposobnosti da transformira nesigurnost ulaznih varijabli u distribuciju vjerojatnosti izlaznih rezultata. Umjesto da projektni menadžer dobije jednu, često pogrešnu, točku procjene (npr. "projekt će trajati 350 dana"), MCS generira cijeli spektar mogućih ishoda (npr. histogram trajanja projekta). Analizom te izlazne distribucije moguće je donijeti puno informiranije odluke, odgovarajući na ključna pitanja poput: "Koja je vjerojatnost da ćemo završiti projekt unutar 400 dana?" ili "Koji je raspon trajanja u kojem se s 90% pouzdanosti možemo očekivati završetak?".

Sam proces simulacije odvija se u nekoliko logičkih koraka. Prvo se identificiraju ključne ulazne varijable čije su vrijednosti neizvjesne, poput trajanja pojedinih aktivnosti. Za svaku takvu varijablu odabire se distribucija vjerojatnosti koja najbolje opisuje raspon i vjerojatnost njenih mogućih vrijednosti. Zatim, algoritam iterativno provodi tisuće "eksperimenata": u svakoj iteraciji, za svaku nesigurnu varijablu generira se jedna nasumična vrijednost iz njene definirane distribucije te se na temelju tih vrijednosti izračunava ishod modela (npr. ukupno trajanje projekta).

Pouzdanost ove metode, kako objašnjavaju Rubinstein i Kroese [7], temelji se na fundamentalnom statističkom principu – Zakonu velikih brojeva. On jamči da će prosječna vrijednost rezultata dobivenih iz velikog broja neovisnih simulacija konvergirati prema stvarnoj očekivanoj vrijednosti sustava. Ponavljanjem procesa velik broj puta, dobiva se empirijska distribucija mogućih ishoda koja pruža dublji uvid od jedne determinističke procjene i iz koje se mogu iščitati ključni statistički pokazatelji poput očekivane vrijednosti, standardne devijacije i vjerojatnosti prekoračenja određenih pragova. Njena matematička podloga leži u **Zakonu velikih brojeva**, koji garantira da će prosječna vrijednost rezultata dobivenih iz velikog broja simulacija konvergirati prema stvarnoj očekivanoj vrijednosti. Proces monte carlo simulacije odvija se u nekoliko koraka. prvo se identificiraju ključne ulazne varijable čije su vrijednosti neizvjesne, poput trajanja pojedinih aktivnosti. za svaku takvu varijablu odabire se distribucija vjerojatnosti koja najbolje opisuje raspon i vjerojatnost njenih mogućih vrijednosti. zatim, algoritam iterativno provodi tisuće "eksperimenata": u svakoj iteraciji, za svaku nesigurnu varijablu generira se jedna nasumična vrijednost iz njene definirane distribucije, te se na temelju tih vrijednosti izračunava ishod modela (npr. ukupno trajanje projekta). ponavljanjem ovog procesa velik broj puta, dobiva se empirijska distribucija mogućih ishoda koja pruža dublji uvid od jedne determinističke procjene i iz koje se mogu

iščitati ključni statistički pokazatelji poput očekivane vrijednosti, standardne devijacije i vjerojatnosti prekoračenja određenih pragova.

2.4 Modeliranje nesigurnosti trajanja pomoću trotočkovne procjene

Metodologija PERT (*Program Evaluation and Review Technique*) uvela je u praksu korištenje tri vremenske procjene za aktivnosti s neizvjesnim trajanjem, što je danas standard u upravljanju projektnim rizikom [2]. Ove tri točke su: T_o(optimistična procjena), T_m(najvjerojatnija procjena) i T_p(pesimistična procjena).

Dok tradicionalna PERT metoda koristi ove tri točke za izračun parametara Beta distribucije, u modernoj praksi, a posebno u Monte Carlo simulacijama, često se koristi **Trokutasta distribucija** zbog svoje jednostavnosti i intuitivnosti [18].

Averill Law, u svom seminalnom djelu "Simulation Modeling and Analysis" (2015), pozicionira trokutastu distribuciju kao iznimno pragmatičan i koristan alat, osobito u situacijama kada nedostaju detaljni povijesni podaci te se modeliranje mora osloniti na stručne procjene. Njena glavna snaga leži u intuitivnosti, jer direktno koristi tri točke procjene (minimum, maksimum i najvjerojatnija vrijednost) koje su lako razumljive i procjenjive od strane projektnih menadžera i inženjera bez dubljeg statističkog znanja. Uz to, njena matematička jednostavnost i definirani interval čine je računski efikasnom i prikladnom za modeliranje realnih veličina poput trajanja aktivnosti, koje ne mogu poprimiti negativne ili beskonačne vrijednosti. Kod trokutaste distribucije i funkcija gustoće vjerojatnosti (PDF) i kumulativna funkcija distribucije (CDF) su matematički jednostavne (sastoje se od linearnih i kvadratnih segmenata), što je čini računalno vrlo "jeftinom" i lakom za implementaciju u simulacijskim alatima.

Unatoč ovim praktičnim prednostima, Law također upozorava na njene teorijske limite. Ističe da je linearni oblik distribucije često proizvoljna pretpostavka koja možda ne odražava u potpunosti stvarni proces, jer postoji fundamentalni razlog zašto bi se vjerojatnost smanjivala točno linearno od najvjerojatnije vrijednosti prema ekstremima. Stvarni procesi često imaju "zaobljenije" distribucije. Uz to, srednja vrijednost i varijanca distribucije su iznimno osjetljive na procjene minimuma (a) i maksimuma (b). Rezultati simulacije mogu biti osjetljivi na nerealno pesimistične ili optimistične krajnje točke koje definira stručnjak. Zbog toga je iznimno važno kritičko propitivanje ekstremnih procjena. Ipak, Law zaključuje da je, unatoč postojanju matematički elegantnijih alternativa poput Beta distribucijom koja se tradicionalno koristi u PERT analizi (dok trokutasta ima oštre vrhove i ravne linije, Beta distribucija (kada se definira s istim parametrima) ima glađi, zaobljeniji oblik koji mnogi smatraju realističnijim) , trokutasta distribucija zbog svoje robusnosti i praktičnosti legitiman i često preferiran izbor u praksi upravljanja rizikom, predstavljajući obranjiv inženjerski pristup modeliranju nesigurnosti.

Ukratko, trokutasta distribucija je kontinuirana distribucija vjerojatnosti definirana s tri parametra: minimum (a), maksimum (b) i najvjerojatnija vrijednost (c), što direktno odgovara procjenama T_o, T_p i T_m. Njena je glavna prednost što ne zahtijeva opsežne povijesne podatke, već se može temeljiti na stručnom iskus-

tvu, što je čini iznimno pogodnom za projektno planiranje. Slučajne vrijednosti generirane iz ove distribucije nalaze se unutar intervala $[T_o, T_p]$, s najvećom vjerojatnošću pojavljivanja oko vrijednosti T_m. Taj konačni interval ključan je za modeliranje realnih pojava. Prosječna vrijednost (očekivano trajanje) za Trokutastu distribuciju računa se jednostavnom formulom:

$$E(T) = \frac{T_o + T_m + T_p}{3}$$

Upravo je Trokutasta distribucija, zbog svih navedenih prednosti, odabrana kao temelj za modeliranje nesigurnosti trajanja aktivnosti u Monte Carlo simulacijama provedenim u ovom radu.

3 Matematička formulacija i konceptualni model problema

Nakon pregleda teorijskih osnova, ovo poglavlje posvećeno je formalnoj definiciji problema optimizacije projektnog portfelja. Cilj je prevesti realni poslovni problem u precizan matematički i konceptualni model na koji se mogu primijeniti računalne metode optimizacije. U nastavku se detaljno opisuju ulazni entiteti, varijable odluke, primijenjena resursna ograničenja te jedno-kriterijski i višekriterijski ciljevi koji su korišteni u kasnijoj eksperimentalnoj evaluaciji.

3.1 Definicija skupa aktivnosti i varijabli odluke

Problem se definira kao odabir optimalnog podskupa (portfelja) aktivnosti iz većeg, unaprijed definiranog skupa dostupnih aktivnosti. U korijenu problema nalazi se skup od n potencijalnih projektnih aktivnosti, $A = \{a_1, a_2, ..., a_n\}$ iz kojeg je potrebno odabrati optimalan podskup (portfelj). Ovakav problem odabira predstavlja čest izazov u praksi projektnog menadžmenta [3, 8]. Svaka aktivnost $a_i \in A$ opisana je s tri ključna atributa koji definiraju njezin trošak, cijenu i rizik.

Odabir ovih specifičnih atributa temelji se na fundamentalnim principima upravljanja projektima, koji uvijek balansiraju između investicije, očekivanog povrata i inherentnog rizika. Prvi je trošak (c_i), koji predstavlja količinu budžeta ili drugih resursa potrebnih za izvođenje aktivnosti. On je primarni faktor ograničenja. Drugi atribut je vrijednost (v_i), definirana kao povrat na investiciju (ROI). Ona kvantificira korist ili profit koji se ostvaruje uspješnim završetkom aktivnosti i predstavlja primarni cilj maksimizacije. Treći, ključni atribut je nesigurnost trajanja. U ovom modelu, trajanje se ne definira kao jedna deterministička vrijednost već kao stohastička varijabla opisana s tri točke procjene: optimističnom (T_o), najvjerojatnijom (T_m) i pesimističnom (T_p). Ove procjene služe kao parametri za Trokutastu distribuciju unutar Monte Carlo simulacije, čime se rizik trajanja uvodi kao integralni dio modela. Ovakav pristup izravno usvaja najbolje prakse iz domene upravljanja projektnim rizikom, poput PERT metodologije, kako bi se u model ugradila realistična slika neizvjesnosti. S obzirom na navedene atribute, temeljni zadatak optimizacije je definirati binarni vektor odluke $x = (x_1, x_2, ..., x_n)$, gdje $x_i \in \{0, 1\}$. Vrijednost $x_i = 1$ označava da je aktivnost a_i odabrana za uključivanje u portfelj, dok $x_i = 0$ označava da se aktivnost ne izvodi. Ovaj vektor odluke je upravo ono što genetski algoritam nastoji optimizirati.

3.2 Resursna ograničenja modela

Svaki realni projekt podložan je ograničenjima. U ovom modelu implementirano je ključno i najčešće ograničenje u upravljanju portfeljem: ograničenje budžeta (B_{max}). Formalno, ovo ograničenje pripada klasi linearnih nejednakosti koje su standard u području cjelobrojnog programiranja (*Integer Programming*), čiji je problem ruksaka klasičan primjer [19]. Prema ovom pravilu, ukupni zbroj troškova svih odabranih aktivnosti ne smije prelaziti ukupno raspoloživi budžet. Upravo ovo ograničenje čini problem optimizacijski izazovnim; bez njega, trivijalno rješenje

bilo bi odabrati sve profitabilne aktivnosti. Dakle, ovo ograničenje odgovara klasičnoj formulaciji problema ruksaka `Knapsack problem` [1] i matematički se izražava kao:

$$\sum_{i=1}^{n} c_i x_i \leq B_{max}$$

Vrijedi napomenuti da ukupno trajanje portfelja nije tretirano kao strogo ograničenje, već kao izlazna metrika performansi i cilj za minimizaciju. Ova metodološka odluka je ključna, jer omogućuje modelu da provodi **analizu kompromisa (trade-off analysis)**, sličnu onoj u teoriji financijskih portfelja. Umjesto postavljanja proizvoljnog roka i rigidnog odbacivanja svih rješenja koja ga prelaze, ovakav pristup omogućuje algoritmu da pronađe rješenja koja su možda malo duža, ali značajno profitabilnija, pružajući donositelju odluke puno bogatiji set informacija. To je u skladu s modernim praksama upravljanja projektnom nesigurnošću, gdje je cilj razumjeti i upravljati rizikom, a ne ga rigidno ograničiti [10].

3.3 Specifikacija optimizacijskih ciljeva

Ciljna funkcija (ili funkcija pogodnosti) je matematički izraz koji kvantificira kvalitetu pojedinog rješenja (portfelja). U skladu s eksperimentalnim dizajnom, definirana su i analizirana dva različita optimizacijska cilja.

Prvi, **jedno-kriterijski cilj**, predstavlja klasični financijski pristup usmjeren isključivo na maksimizaciju dobiti često nazivan pristupom "maksimizacije vrijednosti za dioničare". Ovaj pristup odgovara GA (samo ROI) modelu, a njegova ciljna funkcija $f(\mathbf{x})$ je maksimizacija ukupnog zbroja ROI vrijednosti odabranih aktivnosti, uz poštivanje ograničenja budžeta. Ovakav tip optimizacijskog cilja čest je u primjeni genetskih algoritama na probleme alokacije resursa [5], a formalno se zapisuje kao:

$$\text{maksimizirati } f(\mathbf{x}) = \sum_{i=1}^{n} v_i x_i$$

S druge strane, **više-kriterijski cilj** predstavlja srž ovog istraživanja i odgovara naprednom hibridnom GA+MC (NSGA-II) modelu. Realnost upravljanja projektima jest inherentno više-kriterijska. Uspjeh projekta se rijetko mjeri samo jednim pokazateljem, već balansom između troškova, vremena, opsega i kvalitete [8]. Model to aproksimira kroz dva ključna, suprotstavljena cilja: maksimizaciju profita (ROI) i minimizaciju rizika (procijenjeno trajanje). U ovom pristupu istovremeno se optimiziraju dva suprotstavljena cilja: maksimizacija profita i minimizacija rizika. Definiramo dvije ciljne funkcije: $f_1(\mathbf{x})$ za ukupni ROI i $f_2(\mathbf{x})$ za procijenjeno trajanje. Više-kriterijski optimizacijski problem tada se može formalno zapisati kao:

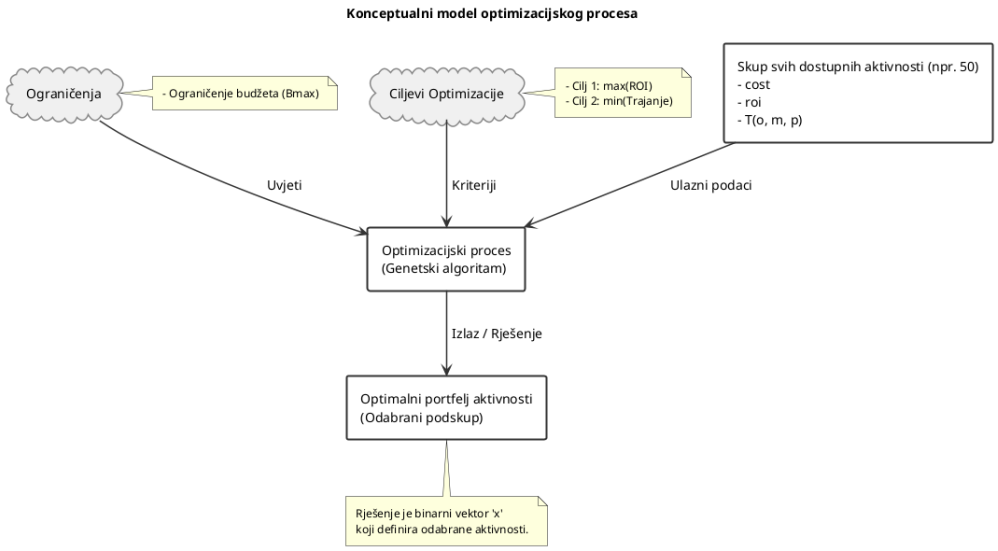
$$\text{maksimizirati } \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), -f_2(\mathbf{x}))$$

gdje je $f_1(\mathbf{x}) = \sum_{i=1}^{n} v_i x_i$, a $f_2(\mathbf{x}) = E[T(\mathbf{x})]$. Negativni predznak ispred f_2 koristi se jer je standardna konvencija maksimizirati sve ciljeve. Budući da poboljšanje

jednog cilja često degradira drugi, ovaj pristup ne traži jedno "najbolje" rješenje, već skup optimalnih kompromisnih rješenja, poznat kao Paretov front. Svako rješenje na Paretovom frontu je takvo da se niti jedan cilj (u ovom slučaju ROI ili trajanje) ne može poboljšati bez da se drugi cilj istovremeno pogorša. Time Paretov front predstavlja "granicu efikasnosti" i nudi donositelju odluke ne jedno rješenje, već cijeli spektar strateških opcija – od onih s maksimalnim profitom uz veći rizik trajanja, do onih s minimalnim rizikom uz manji profit [5, 15]. Za pronalaženje ovog skupa korišten je renomirani NSGA-II algoritam, detaljno opisan u Poglavlju 2.

3.4 Konceptualni prikaz i značaj modela

Konceptualni model problema, koji objedinjuje sve prethodno opisane elemente, prikazan je na Slici 2. Dijagram ilustrira proces kao sustav s jasno definiranim ulazima, procesom obrade i izlazima. **Ulaz** u sustav čini cjelokupni skup dostupnih aktivnosti sa svim njihovim atributima (trošak, ROI, procjene trajanja). **Proces obrade** predstavlja optimizacijski algoritam (genetski algoritam), koji je vođen definiranim **ciljevima** (maksimizacija ROI-a i/ili minimizacija trajanja) i sputan zadanim **ograničenjem** (budžet). Algoritam sustavno pretražuje prostor mogućih kombinacija (svih mogućih vektora x) kako bi kao **izlaz** generirao optimalni portfelj – podskup aktivnosti koji najbolje zadovoljava postavljene kriterije.



Slika 2: Konceptualni model problema optimizacije portfelja projektnih aktivnosti.

Ovaj proces formalizacije ključan je korak koji premošćuje jaz između apstraktnog poslovnog problema i konkretnog računarskog rješenja. Precizna definicija ulaza, ograničenja i ciljeva služi kao temelj na kojem se grade i primjenjuju optimizacijske metode. Za genetski algoritam, ovaj model definira "pravila igre": binarni vektor odluke (x) postaje struktura kromosoma, optimizacijski ciljevi postaju funkcija pogodnosti koja vodi evoluciju, a ograničenje budžeta definira granice valjanog prostora rješenja [14]. Istovremeno, model pruža nužan okvir za

Monte Carlo simulaciju. Stohastički atributi aktivnosti (T_o, T_m, T_p) služe kao direktni ulazni parametri za simulacijski modul. Zauzvrat, Monte Carlo simulacija "obogaćuje" model izračunavanjem vrijednosti za ključni cilj – očekivano trajanje portfelja ($E[T(\mathbf{x})]$) – čime se kvantificira rizik. Na taj način, formalni model uspostavlja sinergijski odnos između optimizacije i simulacije. Genetski algoritam predlaže visokokvalitetne kandidate za rješenja, dok Monte Carlo simulacija procjenjuje njihovu robusnost u uvjetima nesigurnosti [7]. Time je postavljen čvrst temelj za praktičnu implementaciju opisanu u sljedećem poglavlju, te za eksperimentalnu evaluaciju koja slijedi nakon toga, s konačnim ciljem dobivanja rješenja koja nisu samo profitabilna, već i pouzdana.

4 Arhitektura sustava i implementacija

Nakon definiranja teorijskih osnova i formalizacije problema, ovo poglavlje detaljno opisuje praktičnu realizaciju razvijenog optimizacijskog modela. Ovo poglavlje predstavlja most između teorijske formulacije problema iz Poglavlja 3 i empirijske evaluacije rezultata koja slijedi u Poglavlju 5, detaljno prikazujući kako je apstraktni model preveden u operativni računalni sustav. Sustav je implementiran u programskom jeziku **Python (verzija 3.x)**, odabranom zbog čitljivosti, bogatog ekosustava biblioteka i široke primjene u znanstvenom računarstvu [20]. U nastavku se opisuje arhitektura programskog rješenja, korištene tehnologije te detalji implementacije ključnih komponenata.

4.1 Korištene tehnologije i biblioteke

Za izradu sustava korišten je niz standardnih biblioteka iz Python ekosustava za znanstveno računarstvo. Biblioteka DEAP [21] korištena je kao temeljni okvir za implementaciju genetskog algoritma. Za numeričke operacije i statističku obradu korišten je NumPy [22], dok je za manipulaciju i spremanje tabličnih podataka korišten Pandas [23]. Vizualizacija rezultata provedena je pomoću biblioteka Seaborn [24] i Matplotlib [25]. Modeliranje nesigurnosti pomoću Trokutaste distribucije implementirano je korištenjem standardne Python biblioteke Random. Detaljan pregled dan je u Tablici 1.

Tablica 1: Korištene biblioteke u implementaciji

Biblioteka	**Namjena i citat**
Python	Osnovni programski jezik za cjelokupnu implementaciju. [20]
DEAP	Okvir za razvoj i provedbu evolucijskih algoritama. [21]
NumPy	Numeričke operacije i statistička obrada nizova podataka. [22]
Pandas	Učitavanje, obrada i spremanje tabličnih podataka s rezultatima. [23]
Seaborn	Kreiranje naprednih statističkih vizualizacija (stupčasti, linijski i raspršeni grafikoni). [24]
Matplotlib	Osnovna biblioteka za crtanje na koju se oslanja Seaborn. [25]
Random	Standardna Python biblioteka korištena za generiranje slučajnih brojeva i uzorkovanje iz Trokutaste distribucije.

4.2 Arhitektura eksperimentalnog okvira

Sustav razvijen za potrebe ovog rada nije zamišljen kao monolitna skripta, već kao cjeloviti i modularni eksperimentalni okvir. Takva arhitektura, prikazana na Slici 3, omogućuje sustavnu analizu, kalibraciju i usporedbu optimizacijskih

metodologija kroz dvo-fazni proces istraživanja. Okvir se sastoji od dva glavna analitička modula koji slijede dvo-fazni pristup istraživanju, te jednog pomoćnog modula za obradu i prikaz rezultata.

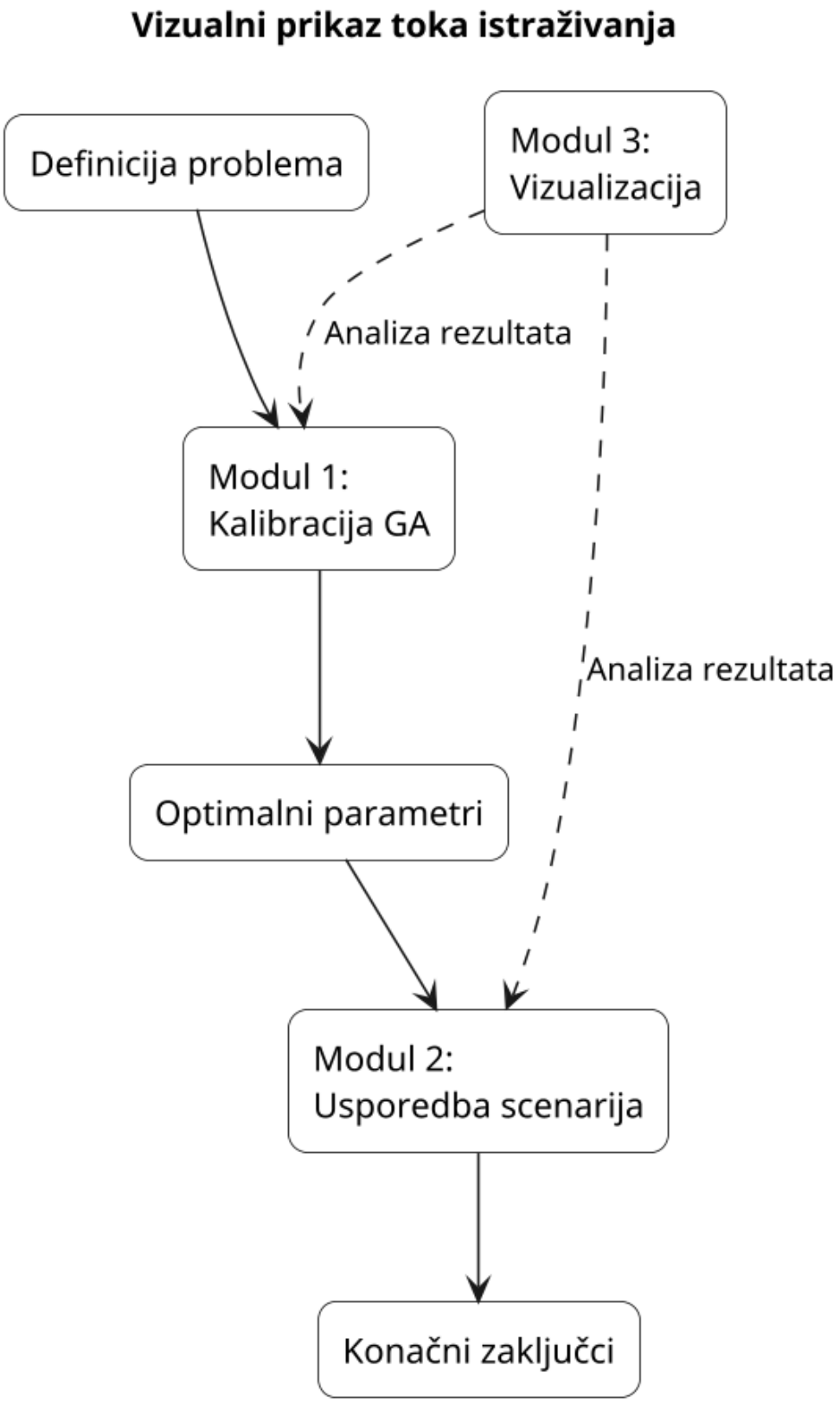
Prvi modul, **Modul za analizu i kalibraciju genetskog algoritma**, predstavlja temelj istraživanja i odgovara na pitanje: *"Kako optimalno konfigurirati genetski algoritam za rješavanje zadanog problema?"*. Njegova primarna svrha je provođenje detaljne ablacijske studije kojom se ispituje utjecaj svakog ključnog parametra na performanse. Kroz višestruka pokretanja različitih konfiguracija (standardni GA, bez križanja, bez mutacije, s povećanim brojem generacija, s većom populacijom), izračunavanjem metrika performansi, uključujući prosječnu vrijednost (mean) i standardnu devijaciju (std) za ROI i procijenjeno trajanje, ovaj modul kao izlaz generira "šampionsku" konfiguraciju – skup optimalnih parametara koji osiguravaju najbolje performanse i koji se koriste u daljnjoj analizi.

Drugi, ključni modul je **Modul za usporednu analizu optimizacijskih scenarija**. On čini srž diplomskog rada i koristi "šampionsku" konfiguraciju za provođenje konačne, statistički robusne usporedbe triju različitih pristupa rješavanju problema: osnovnog modela nasumične pretrage, klasičnog genetskog algoritma usmjerenog isključivo na ROI, te hibridnog GA+MC modela (NSGA-II) koji provodi više-kriterijsku optimizaciju koja istovremeno maksimizira ROI i minimizira rizik trajanja procijenjen Monte Carlo simulacijom. Izlaz ovog modula je konačna tablica s usporednim rezultatima performansii(ROI, trajanje) i stabilnosti(standardna devijacija) za svaki od triju scenarija.

Treći, **Modul za obradu i vizualizaciju rezultata**, služi kao pomoćni alat za interpretaciju podataka dobivenih iz prva dva modula. Njegova funkcionalnost obuhvaća obradu i transformaciju sirovih podataka u pregledne tablice pomoću biblioteke `pandas`, što olakšava strukturiranje rezultata i spremanje u CSV format. Za stvaranje grafičkih prikaza, kao što su stupčasti dijagrami za usporedbu prosječnih vrijednosti, 2D raspršeni dijagrami za prikaz Paretovog fronta ili dijagrami konvergencije, korištene su snažne biblioteke `matplotlib` i `seaborn`. Dok `matplotlib` pruža temelje za stvaranje raznih tipova grafova, `seaborn`, izgrađen na njemu, nudi naprednije statističke prikaze i atraktivniji, profesionalni izgled uz minimalno kodiranje.

Osim spomenutih funkcionalnosti, ključan zadatak ovog modula je **analiza i vizualizacija konvergencije algoritma**. Ova analiza pruža neposredan dokaz da je evolucijski proces bio uspješan. Konvergencija se prati kroz sve generacije bilježenjem promjena u fitnes vrijednostima (ROI-u) najbolje jedinke u svakoj generaciji. Uspješna konvergencija manifestira se u obliku krivulje koja naglo raste u početnim generacijama, a zatim se postupno izravnava, što ukazuje na to da je algoritam pronašao optimalno ili blizu-optimalno rješenje i da daljnja evolucija ne donosi značajna poboljšanja.

Vizualni prikaz konvergencije, koji stavlja fitnes vrijednost na Y-os, a broj generacije na X-os, ključan je alat za potvrdu da algoritam nije stagnirao i da je dosegnuo stabilno stanje. Također pomaže u određivanju optimalnog broja generacija potrebnih za rješavanje problema, čime se direktno utječe na računalnu efikasnost. Na taj način, ovaj modul ne samo da prikazuje konačne rezultate, već i pruža uvid u sam proces optimizacije.



Slika 3: Vizualni prikaz toka istraživanja

4.3 Implementacija ključnih komponenti

4.3.1 Modeliranje nesigurnosti: Monte Carlo simulacija

Za svaku projektnu aktivnost definirane su tri točke procjene trajanja:

$$a \text{ (optimistična)}, \quad m \text{ (najvjerojatnija)}, \quad b \text{ (pesimistična)}$$

Iako u teoriji postoje kompleksnije distribucije poput *Beta-PERT* distribucije, za potrebe ovog rada odabrana je **Trokutasta distribucija (Triangular distribution)** zbog svoje praktičnosti, računalne efikasnosti i intuitivnog temelja na tri poznate procjene. Standardna Python biblioteka `random` pokazala se dovoljnom za generiranje trokutastih distribucija, eliminirajući potrebu za dodatnim, kompleksnijim bibliotekama, što doprinosi jednostavnosti i prenosivosti koda. Ključno je napomenuti da se stohastička komponenta (simulacija) izvršava unutar determinističkog okvira genetskog algoritma, što znači da se svaka jedinka evaluira na temelju statistički stabilne prosječne vrijednosti trajanja, a ne na temelju jedne slučajne procjene

Generiranje trajanja aktivnosti. U svakoj iteraciji Monte Carlo simulacije, trajanje svake aktivnosti generira se slučajnom vrijednošću unutar raspona $[a, b]$ s najvećom vjerojatnošću u točki m. Trokutasta distribucija definirana je funkcijom gustoće vjerojatnosti:

$$f(x) = \begin{cases} \frac{2(x-a)}{(b-a)(m-a)}, & a \le x < m, \\ \frac{2(b-x)}{(b-a)(b-m)}, & m \le x \le b, \\ 0, & \text{inače.} \end{cases}$$

Procjena trajanja portfelja. Ukupno trajanje projektnog portfelja u jednoj simulaciji dobiva se zbrojem trajanja svih aktivnosti odabranih u tom portfelju:

$$T_{\text{portfolio}} = \sum_{i \in S} t_i$$

gdje je S skup odabranih aktivnosti, a t_i generirano trajanje aktivnosti i.

Implementacija se temelji na `monte_carlo_eval_duration` funkciji prikazanoj u Isječku koda 1, koja predstavlja konkretnu realizaciju teorijskog procesa opisanog u Poglavlju 2.

Objašnjenje **Isječka koda 1**:

Funkcija `monte_carlo_eval_duration` predstavlja srce Monte Carlo simulacije. Njezina je svrha procijeniti očekivano trajanje zadanog portfelja projekata, uzimajući u obzir inherentnu nesigurnost.

`def monte_carlo_eval_duration(individual, activities, config)`: Definira funkciju koja prihvaća tri ključna ulazna parametra: `individual` (binarni niz koji predstavlja jedinku, tj. odabrani portfelj), `activities` (lista svih dostupnih projekata) i `config` (rječnik s postavkama, npr. broj simulacija).

```python
def monte_carlo_eval_duration(individual, activities, config):
    selected = [act for i, act in enumerate(activities) if individual[i] == 1]
    if not selected:
        return 0.0
    durations = [
        sum(
            random.triangular(act["optimistic"], act["realistic"], act["pessimistic"])
            for act in selected
        )
        for _ in range(config["NUM_SIMULATIONS"])
    ]
    return np.mean(durations)
```

Listing 1: Funkcija za Monte Carlo procjenu trajanja

`selected = [act for i, act in enumerate(activities) if individual[i] == 1]` Ova linija koristi *list comprehension* kako bi, na temelju binarne reprezentacije `individual` jedinke, stvorila novu listu koja sadrži samo objekte (rječnike) za odabrane projekte.

`if not selected: return 0.0` Jednostavna provjera koja osigurava da funkcija pravilno postupa s praznim portfeljima, vraćajući `0.0` kao njihovo trajanje.

`durations = [...]` Ovaj dio je srž simulacije. Vanjska *list comprehension* `for _ in range(config["NUM_SIMULATIONS"])` osigurava da se proces ponovi točno onoliko puta koliko je zadano u konfiguraciji. Unutarnja petlja `for act in selected` prolazi kroz svaki odabrani projekt u jednoj simulaciji.

`random.triangular(act["optimistic"], act["realistic"], act["pessimistic"])` Ovo je ključni poziv funkciji iz Pythonove `random` biblioteke. Ona generira jednu slučajnu vrijednost trajanja za svaki projekt, koristeći njegove tri točke procjene (optimistična, najvjerojatnija, pesimistična).

`sum(...)` Suma se koristi za zbrajanje trajanja svih projekata u toj jednoj simulaciji, čime se dobiva jedno moguće ukupno trajanje portfelja.

`return np.mean(durations)` Konačno, funkcija vraća prosječnu vrijednost svih `NUM_SIMULATIONS` trajanja. Ova prosječna vrijednost služi kao statistički pouzdana procjena očekivanog trajanja portfelja i koristi se u funkciji pogodnosti.

Agregiranje rezultata. Funkcija za zadani portfelj provodi velik broj simulacija (NUM_SIMULATIONS). U svakoj simulaciji, za svaku odabranu aktivnost

generira se slučajno trajanje iz Trokutaste distribucije, zbrajaju se trajanja unutar te simulacije, a na kraju se vraća prosječna vrijednost svih simuliranih ukupnih trajanja.

$$\overline{T}(S) = \frac{1}{\text{NUM\_SIMULATIONS}} \sum_{k=1}^{\text{NUM\_SIMULATIONS}} T_{\text{portfolio}}^{(k)}$$

gdje $T_{\text{portfolio}}^{(k)}$ označava ukupno trajanje portfelja u k-toj simulaciji.

4.3.2 Optimizacijski pristup: Genetski algoritam

Implementacija genetskog algoritma provedena je pomoću programske biblioteke `DEAP` (Distributed Evolutionary Algorithms in Python) [21]. S obzirom na prirodu problema odabira podskupa aktivnosti, korištena je **binarna reprezentacija**, gdje svaka jedinka (kromosom) predstavlja jedan portfelj projekata u obliku binarnog niza. Vrijednost '1' na poziciji i označava da je i-ta aktivnost odabrana, dok '0' označava da nije.

Reprezentacija jedinke. Svaka jedinka (kromosom) u populaciji predstavlja jedno potencijalno rješenje – jedan portfelj projekata. Predstavljena je kao binarni niz duljine jednake ukupnom broju aktivnosti (NUM_ACTIVITIES), gdje gen na poziciji i ima vrijednost:

$$g_i = \begin{cases} 1, & \text{ako je } i\text{-ta aktivnost odabrana,} \\ 0, & \text{ako nije odabrana.} \end{cases}$$

Konfiguracija genetskih operatora (DEAP Toolbox) DEAP koristi koncept "kutije s alatima" (`Toolbox`) za registraciju svih komponenata potrebnih za evolucijski proces. Za ovaj rad, operatori su odabrani na temelju standardnih praksi za binarne genetske algoritme i specifičnosti problema, kako je prikazano u Isječku koda 2. U nastavku je dano obrazloženje za odabir svakog ključnog operatora.

Selekcija. Za jedno-kriterijsku optimizaciju odabrana je turnirska selekcija (`tools.selTournament`). Za razliku od drugih metoda poput selekcije ruleta, koja može dovesti do prerane konvergencije ako jedna "super-jedinka" dominira populacijom, turnirska selekcija pruža bolju kontrolu nad "selekcijskim pritiskom" [5]. Korištenjem manjeg turnira (veličine 3), osigurava se da i prosječno dobre jedinke imaju priliku za reprodukciju, što pomaže u očuvanju genetske raznolikosti. Za više-kriterijsku optimizaciju, primjena operatora `tools.selNSGA2` je nužna jer on ne provodi jednostavnu selekciju, već implementira cjelokupnu logiku sortiranja po nedominaciji i izračuna gustoće naseljenosti, što je srž NSGA-II algoritma opisanog u Poglavlju 2.

Križanje. Kao glavni mehanizam rekombinacije, odabrano je dvo-točkovno križanje (`tools.cxTwoPoint`). Ova metoda predstavlja dobar kompromis između očuvanja dobrih shema (građevnih blokova) i istraživanja novih kombinacija. U usporedbi s jedno-točkovnim križanjem, manje je podložno pozicijskoj pristranosti, dok je istovremeno manje disruptivno od uniformnog križanja, koje može previše lako razbiti uspješne kombinacije gena. Dvo-točkovno križanje se stoga smatra robusnim i pouzdanim izborom za mnoge binarne probleme [14].

Mutacija. Odabrana je standardna mutacija slučajne promjene bita (`tools.mutFlipBit`). Njena uloga je ključna za istraživanje (exploration) i sprječavanje stagnacije algoritma. Ovaj operator osigurava da za svaku poziciju u kromosomu uvijek postoji mala, ne-nulta vjerojatnost promjene vrijednosti (iz 0 u 1 ili obrnuto). Time se garantira da algoritam može doseći bilo koju točku u prostoru rješenja i sprječava se situacija u kojoj cijela populacija konvergira prema genu iste vrijednosti, iz koje se samo križanjem ne bi mogla oporaviti.

Vjerojatnosti primjene križanja (`CX_PB`) i mutacije (`MUT_PB`) definirane su kao vanjski parametri u glavnom konfiguracijskom rječniku, što omogućuje njihovo lako podešavanje i testiranje, kao što je i učinjeno u prvoj fazi eksperimenata.

Koncept Toolbox u DEAP biblioteci omogućuje iznimnu modularnost i fleksibilnost. Umjesto da se operatori pozivaju direktno, oni se registriraju u Toolbox objektu pod aliasima (`mate`,`mutate`,`select`,`evaluate`). Ova 'registracija' stvara apstrakciju koja omogućava lako zamjenjivanje, testiranje i prebacivanje između različitih operatora bez promjene osnovne strukture evolucijskog petlje, što je bilo presudno za fazu kalibracije algoritma u prvom eksperimentu

Objašnjenje **Isječka koda 2**: Ovaj isječak prikazuje konfiguraciju "kutije s alatima" (`Toolbox`) iz `DEAP` biblioteke.

`toolbox = base.Toolbox()` Inicijalizira prazan `Toolbox` objekt koji će služiti kao registar za sve komponente algoritma.

`toolbox.register("attr_bool", random.randint, 0, 1)` Registrira funkciju `random.randint` pod aliasom "`attr_bool`". Ova funkcija bit će odgovorna za generiranje pojedinačnih gena (0 ili 1) za kromosom.

`toolbox.register("individual", tools.initRepeat, creator.Individual, ...)` Koristi `tools.initRepeat` za kreiranje cijele jedinke. Drugim riječima, kreira niz gena pozivajući "`attr_bool`" onoliko puta koliko je zadano brojem projekata (`NUM_ACTIVITIES`).

`toolbox.register("population", tools.initRepeat, list, toolbox.individual)` Slično, registrira funkciju za kreiranje populacije, koja se sastoji od ponovljenog pozivanja funkcije "`individual`".

```
toolbox.register("mate", tools.cxTwoPoint),

toolbox.register("mutate", tools.mutFlipBit, indpb=0.1),
```

```
1
2  Inicijalizacija Toolbox objekta
3  toolbox = base.Toolbox()
4
5  Registracija operatora za generiranje jedinki i populacije
6  toolbox.register("attr_bool", random.randint, 0, 1)
7  toolbox.register("individual", tools.initRepeat, creator.
       Individual,
8  toolbox.attr_bool, config["NUM_ACTIVITIES"])
9  toolbox.register("population", tools.initRepeat, list,
       toolbox.individual)
10
11 Registracija genetskih operatora
12 toolbox.register("mate", tools.cxTwoPoint)
13 toolbox.register("mutate", tools.mutFlipBit, indpb=0.1)
14 toolbox.register("select", tools.selTournament, tournsize
       =3)
15 toolbox.register("evaluate", single_objective_fitness)
```

Listing 2: Primjer konfiguracije DEAP Toolbox-a za genetske operatore.

`toolbox.register("select", tools.selTournament, tournsize=3)` Ove tri linije registriraju ključne genetske operatore. Pod aliasima `mate`, `mutate` i `select` skrivaju se specifične funkcije za križanje, mutaciju i selekciju, što omogućuje njihovu laku zamjenu (npr. promjenu iz turnirske selekcije u `NSGA-II` selekciju).

`toolbox.register("evaluate", single_objective_fitness)` Ovo je ključna linija koja povezuje genetski algoritam s funkcijom pogodnosti. Ona govori algoritmu koju funkciju treba pozvati kako bi izračunao "dobrotu" svake jedinke, čime se završava priprema za evolucijsku petlju.

implementacija funkcija pogodnosti (Fitness Function). Ključni dio implementacije su funkcije pogodnosti. Ovisno o scenariju, korištene su dvije različite funkcije. Za jedno-kriterijsku optimizaciju (GA samo ROI), implementirana je funkcija koja maksimizira ROI i primjenjuje strogu kaznenu metodu za rješenja koja prekoračuju budžet, osiguravajući da su sva nevaljana rješenja lošija od bilo kojeg valjanog. Za hibridni scenarij (GA+MC), implementirana je više-kriterijska funkcija pogodnosti koja predstavlja srž ovog rada. Kao što je prikazano u Isječku koda 3, ova funkcija unutar jedne evaluacije objedinjuje deterministički izračun (ukupni trošak i ROI) i poziv stohastičke Monte Carlo simulacije za procjenu rizika (očekivano trajanje). Vraća tuple s dvije vrijednosti, omogućujući NSGA-II algoritmu da istovremeno optimizira oba cilja.

Objašnjenje **Isječka koda 3**: Ova funkcija predstavlja srž više-kriterijske optimizacije. Za razliku od jedno-kriterijskog pristupa, ona ne vraća jednu vrijednost, već tuple koji `NSGA-II` algoritam koristi za rangiranje jedinki i pronalaženje Pa-

```
def multi_objective_fitness(individual, activities, config):
    total_cost, total_roi = calculate_metrics(individual, activities)
    if total_cost > config["BUDGET"]:
        return 0, 99999
    avg_duration = monte_carlo_eval_duration(individual, activities, config)
    return total_roi, avg_duration
```

Listing 3: Više-kriterijska funkcija pogodnosti

retovog fronta.

`def multi_objective_fitness(individual, activities, config):` Definira funkciju pogodnosti koja prima jedinku, listu aktivnosti i konfiguracijske parametre.

`total_cost, total_roi = calculate_metrics(individual, activities)` Ova linija poziva pomoćnu funkciju `calculate_metrics` (čiji kod nije prikazan, ali je opisana njezina funkcija). Ona izračunava ukupni trošak i `ROI` odabranog portfelja, koristeći deterministične vrijednosti.

`if total_cost > config["BUDGET"]: return 0, 99999` Ovo je implementacija stroge kaznene funkcije. Ako portfelj prekoračuje budžet, funkcija odmah vraća loše vrijednosti (niski `ROI`, iznimno dugo trajanje), osiguravajući da `NSGA-II` algoritam odbaci takva nevaljana rješenja.

`avg_duration = monte_carlo_eval_duration(individual, activities, config)` Ovo je točka u kojoj se događa stvarna sinergija. Unutar funkcije pogodnosti, koja se poziva za svaku jedinku u svakoj generaciji, poziva se funkcija za Monte Carlo simulaciju kako bi se dobila statistički stabilna procjena trajanja portfelja.

`return total_roi, avg_duration` Konačno, funkcija vraća 'tuple' s dvije vrijednosti: izračunatim `ROI`-em i procijenjenim trajanjem. Ove dvije vrijednosti predstavljaju dva optimizacijska cilja (maksimizacija `ROI`-a i minimizacija trajanja) na temelju kojih `NSGA-II` rangira jedinke i pretražuje prostor rješenja.

Ovisno o eksperimentalnom scenariju, korištene su dvije vrste funkcije pogodnosti:

1. **Jedno-kriterijska optimizacija.** U scenariju GA (samo ROI), algoritam se suočava s klasičnim problemom ograničene optimizacije (*constrained optimization*). Budući da su genetski algoritmi u svojoj osnovi neograničeni pretraživači prostora rješenja, potreban je mehanizam koji će populaciju "usmjeriti" prema rješenjima koja zadovoljavaju zadane uvjete – u ovom

slučaju, ograničenje budžeta. Najčešći pristup za rješavanje ovog problema je korištenje kaznenih funkcija (*penalty functions*), koje smanjuju fitness nevaljanih rješenja [5].

U ovom radu primijenjena je stroga i direktna kaznena metoda. Ako ukupni trošak odabranog portfelja S ne prelazi budžet, njegova pogodnost jednaka je pozitivnoj vrijednosti ukupnog ROI-a. Međutim, ako je budžet prekoračen, pogodnost postaje negativna vrijednost proporcionalna iznosu prekoračenja:

$$\text{Fitness}(S) = \begin{cases} \sum_{i \in S} \text{ROI}_i, & \text{ako } \sum_{i \in S} \text{Trošak}_i \leq \text{Budžet} \\ -(\sum_{i \in S} \text{Trošak}_i - \text{Budžet}), & \text{ako } \sum_{i \in S} \text{Trošak}_i > \text{Budžet} \end{cases}$$

Odabir ovakve "smrtne kazne" (*death penalty*) za nevaljana rješenja, gdje fitness postaje negativan, ima nekoliko prednosti. Prvo, implementacija je jednostavna i računalno efikasna. Drugo, stvara jasnu i oštru granicu između valjanog i nevaljanog prostora rješenja, garantirajući da će evolucijski proces uvijek preferirati bilo koje, pa i najlošije valjano rješenje (s pozitivnim fitnessom) u odnosu na bilo koje nevaljano rješenje (s negativnim fitnessom). Iako u literaturi postoje i naprednji pristupi, poput adaptivnih kazni (gdje se jačina kazne mijenja tijekom evolucije) ili algoritama za popravak (koji pokušavaju "popraviti" nevaljane jedinke), odabrana metoda se pokazala dovoljno robusnom i efikasnom za ovaj tip problema, izbjegavajući uvođenje dodatne složenosti.

Ovakav pristup osigurava da genetski algoritam brzo nauči izbjegavati nevaljana područja i fokusira svoju pretragu na obećavajuće, profitabilne portfelje koji zadovoljavaju ključno poslovno ograničenje.

2. **Više-kriterijska optimizacija.** Za razliku od jedno-kriterijskog pristupa, hibridni scenarij GA+MC usvaja realističniji, više-kriterijski pogled na problem, prepoznajući da uspjeh projektnog portfelja ne ovisi samo o financijskoj dobiti, već i o vremenskom riziku. Stoga su u ovom naprednom modelu, koji koristi NSGA-II algoritam, definirana dva suprotstavljena optimizacijska cilja:

   (a) Maksimizacija ukupnog povrata na investiciju (ROI), kao mjera profitabilnosti.

   (b) Minimizacija prosječnog trajanja portfelja, procijenjenog Monte Carlo simulacijom, kao mjera rizika.

   Između ova dva cilja postoji fundamentalan i neizbježan kompromis (*trade-off*). Portfelji koji teže maksimalnom ROI-u često uključuju veći broj aktivnosti ili one koje su dugotrajnije i rizičnije. S druge strane, portfelji koji minimiziraju trajanje obično su manji i konzervativniji, te posljedično imaju nižu ukupnu vrijednost. Zbog ovog konflikta, ne postoji jedno, savršeno rješenje koje je istovremeno najbolje po oba kriterija. Formalno, ovaj se više-kriterijski optimizacijski problem može zapisati kao:

$$\begin{cases} \max f_1(\mathbf{x}) = \sum_{i \in S(\mathbf{x})} v_i \\ \min f_2(\mathbf{x}) = E[T(\mathbf{x})] \end{cases}$$

gdje $S(\mathbf{x})$ označava skup odabranih aktivnosti za dani vektor odluke $\mathbf{x}$, a $E[T(\mathbf{x})]$ je očekivano trajanje tog portfelja dobiveno Monte Carlo simulacijom. U praktičnoj implementaciji, ovo se postiže pomoću funkcije pogodnosti (`multi_objective_fitness`) koja za svaku jedinku ne vraća jednu skalarnu vrijednost, već dvoelementni *tuple* koji sadrži vrijednosti oba cilja (ROI, trajanje). Na temelju tih vektorskih vrijednosti, NSGA-II algoritam, detaljno opisan u Poglavlju 2, rangira populaciju i pretražuje prostor rješenja s ciljem pronalaženja Paretovog fronta – skupa optimalnih kompromisnih rješenja.

4.4 Vizualizacija i obrada rezultata

Za analizu i prikaz rezultata dobivenih optimizacijom korištene su biblioteke `pandas` za tabličnu obradu podataka te `Seaborn` i `Matplotlib` [24, 25] za grafičku vizualizaciju. Kombinacija ovih alata odabrana je zbog njihove sinergije i efikasnosti u radu s podacima. **Pandas** je korišten kao temelj za organizaciju i manipulaciju eksperimentalnih podataka, omogućujući efikasnu filtraciju, sortiranje i agregiranje sirovih rezultata u strukturirane, pregledne tablice. Njegova snaga leži u mogućnosti brzog izračunavanja ključnih statistika (poput prosjeka i standardnih devijacija) potrebnih za usporedbu modela. Pandas je bio nezamjenjiv u ovoj fazi. Na primjer, nakon svake serije pokretanja algoritma, rezultati su se automatski spremali u `DataFrame` objekte, što je omogućilo lagano izračunavanje prosječnih vrijednosti i standardnih devijacija za usporedbu performansi. Uz to, Pandas je korišten za čitanje ulaznih podataka iz CSV datoteka, osiguravajući robustan i standardiziran ulazno-izlazni proces.

Za vizualizaciju, **Matplotlib** je služio kao osnovni "motor" za crtanje, dok je **Seabornu** dana prednost zbog njegove visoke razine apstrakcije i fokusiranosti na izradu statističkih grafova. Seaborn, izgrađen na Matplotlibu, omogućuje stvaranje estetski privlačnih i informativnih dijagrama s minimalno koda, automatski se brinući o bojama, temama i složenim prikazima odnosa među podacima. Upravo je ta jednostavnost i elegancija bila ključna za učinkovit prikaz rezultata više-kriterijske optimizacije.

Ovaj pristup omogućio je sustavnu prezentaciju statistički obrađenih podataka kroz detaljne tablice te vizualnu usporedbu modela pomoću stupčastih i raspršenih dijagrama. Posebno je značajan prikaz Paretovog fronta, koji jasno ilustrira kompromis (*trade-off*) između maksimizacije ROI-a i minimizacije trajanja, pružajući intuitivan uvid u kvalitetu rješenja dobivenih više-kriterijskom optimizacijom.

Ključni vizualni elementi korišteni u ovom radu uključuju:

- **Tablične prikaze:** Detaljne tablice s konačnim, statistički obrađenim rezultatima usporedbe različitih optimizacijskih scenarija, uključujući osnovne metrike poput prosječnog ROI-a, prosječnog trajanja te raspona vrijednosti.
- **Stupčaste dijagrame:** Koristili su se za vizualnu usporedbu prosječnih vrijednosti (*ROI* i trajanje) između različitih metodologija optimizacije, omogućujući brzu identifikaciju učinkovitijih pristupa.

- **Dijagrame konvergencije:** Linijski dijagrami koji prikazuju evoluciju fitnes vrijednosti najbolje jedinke (ROI-a) kroz generacije. Ovi dijagrami ključni su za potvrdu da algoritam nije stagnirao i da je uspješno konvergirao prema optimalnom rješenju.

- **Raspršene dijagrame (*Scatter Plot*):** Prikaz Paretovog fronta dobivenog NSGA-II algoritmom, koji jasno ilustrira kompromis (*trade-off*) između dvaju suprotstavljenih ciljeva: maksimizacije ROI-a i minimizacije trajanja. Time se omogućuje intuitivna procjena učinkovitosti rješenja.

Vizualizacija rezultata odigrala je ključnu ulogu u interpretaciji dobivenih podataka, posebno u scenarijima s više ciljeva, gdje tablični prikazi sami po sebi nisu dovoljni za uočavanje odnosa i kompromisa među varijablama.

Sveobuhvatni metodološki okvir i detaljna implementacija opisana u ovom poglavlju služe kao temelj za daljnju analizu. Precizna konfiguracija sustava, od odabira biblioteka do implementacije prilagođenih genetskih operatora i funkcija pogodnosti, ključna je za osiguravanje pouzdanosti i valjanosti dobivenih rezultata. Ovaj okvir nije samo teorijski model, već robustan eksperimentalni sustav koji omogućuje testiranje i usporedbu različitih optimizacijskih scenarija u kontroliranom okruženju. S uspostavljenom arhitekturom i implementacijom svih ključnih modula, rad je spreman za sljedeću fazu – empirijsku evaluaciju i diskusiju rezultata, što je i fokus sljedećeg poglavlja.

5 Eksperimentalna evaluacija i diskusija rezultata

Nakon definiranja teorijskih osnova i implementacije modela, ovo poglavlje posvećeno je empirijskoj validaciji i analizi. Detaljno se opisuje eksperimentalni postav, provedba dvo-faznog istraživanja, te se provodi dubinska analiza i interpretacija dobivenih rezultata. Cilj je kvantificirati performanse predloženih modela i donijeti utemeljene zaključke o njihovoj primjenjivosti.

5.1 Postavke okruženja i testni podaci

Svi eksperimenti provedeni su u programskom okruženju **Python (verzija 3.x)** na standardnom osobnom računalu. Kako bi se osigurala ponovljivost i kontrolirani uvjeti, za potrebe istraživanja generiran je sintetički skup podataka koji oponaša realističan projektni portfelj. Generiranje podataka je parametrizirano kako bi se mogli stvoriti problemi različite složenosti (broj aktivnosti od 10 do 100). Za svaku aktivnost, vrijednosti troška, ROI-a i trotočkovne procjene trajanja generirane su nasumično unutar realističnih raspona: **Trošak (`cost`):** slučajna cjelobrojna vrijednost između 50 i 200, **ROI (`roi`):** slučajna decimalna vrijednost između 1.0 i 3.0, i **Procjene trajanja:** optimistično (između 5 i 10 dana), najvjerojatnije (između 10 i 20 dana) i pesimistično (između 20 i 40 dana). Ukupni raspoloživi budžet za portfelj (`BUDGET`) bio je skaliran u skladu sa složenošću problema kako bi se održala slična razina restriktivnosti.

5.2 Eksperimentalni dizajn

Kako bi se osigurala metodološka ispravnost i izbjegli proizvoljni zaključci, istraživanje je provedeno kroz dvo-fazni eksperimentalni proces:

- **Faza 1: Analiza i kalibracija genetskog algoritma.** U prvoj fazi provedena je detaljna ablacijska studija kako bi se utvrdilo koji parametri genetskog algoritma daju najkvalitetnija i najstabilnija rješenja za reprezentativni tip problema (50 aktivnosti). Cilj je bio pronaći "šampionsku" konfiguraciju GA.
- **Faza 2: Usporedna analiza optimizacijskih modela.** U drugoj fazi, "šampionska" konfiguracija GA, dobivena u prvoj fazi, korištena je kao osnova za provođenje konačne usporedbe triju različitih optimizacijskih scenarija i evaluaciju istraživačkih hipoteza: **H1 (Skalabilnost), H2 (Kompromis) i H3 (Utjecaj Ograničenja)** rada na problemima različite skale i restriktivnosti.

5.2.1 Istraživačke hipoteze

Na temelju teorijske podloge i postavljenih istraživačkih pitanja iz Uvoda, definirane su sljedeće tri glavne hipoteze koje su vođene ovim istraživanjem i predstavljaju temelj za analizu i interpretaciju eksperimentalnih rezultata:

H1 (Hipoteza o Skalabilnosti): "Povećanjem složenosti problema (broja aktivnosti), performanse modela temeljenog na nasumičnoj pretrazi (`Random Search`) će se značajno smanjiti u usporedbi s modelima temeljenim na genetskim algoritmima." Ova hipoteza testira temeljnu prednost inteligentnih metaheuristika. Svrha je dokazati da se, u problemskom prostoru koji eksponencijalno raste s brojem projekata, slijepa, nasumična pretraga brzo gubi u "astronomski velikom" prostoru rješenja. S druge strane, genetski algoritmi, s mehanizmima kao što su križanje i mutacija, inteligentno istražuju prostor, usmjeravajući se prema obećavajućim područjima. Ova hipoteza predviđa da će se razlika u performansama između ove dvije metodologije drastično povećati s porastom složenosti problema, čime će se potvrditi nužnost primjene naprednijih metoda.

H2 (Hipoteza o Kompromisu): "Hibridni 'GA+MC' model će, za razliku od klasičnog 'GA (samo ROI)' modela, uspješno identificirati rješenja koja predstavljaju superioran kompromis između profitabilnosti (`ROI`) i rizika (trajanje projekta), posebno na problemima veće složenosti."

Ova hipoteza je srž cijelog rada i istražuje višeciljnu prirodu problema. Dok klasični `GA` model vidi samo jednu dimenziju (profit), hibridni model, integracijom Monte Carlo simulacije, "vidi" i drugu dimenziju (rizik trajanja). Hipoteza predviđa da ova sposobnost istovremene optimizacije dva suprotstavljena cilja neće rezultirati samo jednim rješenjem, već cijelim skupom nedominiranih rješenja poznatim kao Pareto front. Očekuje se da će taj skup rješenja pružiti neprocjenjivu stratešku vrijednost, omogućujući menadžeru da kvantificira i odabere optimalan kompromis između maksimizacije profita i minimizacije rizika.

H3 (Hipoteza o Utjecaju Ograničenja): "Restriktivnost problema, specifično kroz promjenu raspoloživog budžeta, značajno utječe na performanse i stabilnost optimizacijskih modela, pri čemu se očekuje da će ekstremna ograničenja predstavljati najveći izazov za najsloženije modele."

Ova hipoteza testira robusnost modela u nerealističnim, ali izazovnim uvjetima. Umjesto jednostavnog zaključka da je složeniji model uvijek bolji, ova hipoteza predviđa paradoksalan nalaz. Očekuje se da će hibridni, višekriterijski model, koji briljira u "otvorenom" prostoru rješenja, pokazati neočekivanu krhkost kada se prostor rješenja drastično suzi ekstremno niskim budžetom. S druge strane, jednostavniji model fokusiran na jedan cilj (`ROI`) mogao bi se pokazati robusnijim i stabilnijim u takvim uvjetima. Ova hipoteza naglašava da je odabir najprikladnijeg alata kontekstualan i ovisi o prirodi problema.

5.3 Eksperiment 1: Analiza parametara i kalibracija genetskog algoritma

Prvi eksperiment imao je za cilj empirijski provjeriti utjecaj osnovnih genetskih operatora i parametara na performanse algoritma te odabrati optimalnu konfi-

guraciju za daljnje testiranje. U tu svrhu provedena je ablacijska studija s pet različitih konfiguracija, gdje je svaka pokrenuta 10 puta (`RUNS = 10`) radi statističke pouzdanosti. Testirane konfiguracije su bile: *Standardni GA*, *Bez mutacije*, *Bez križanja*, *Više generacija* i *Veća populacija*.

5.3.1 Rezultati i diskusija ablacijske studije

Rezultati ablacijske studije, prikazani u Tablici 2 i grafički na Slici 4, i Slici 5 pružaju dubok uvid u dinamiku i kalibraciju genetskog algoritma. Analiza je fokusirana na utjecaj pojedinih operatora i na usporedbu strategija pretrage.

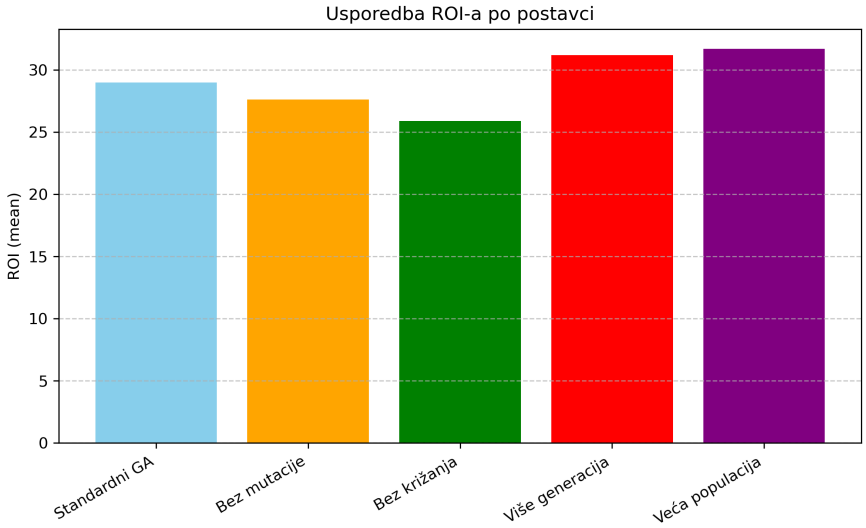
Tablica 2: Rezultati ablacijske studije za parametre GA.

Postavka	**ROI_mean**	**ROI_std**	**Trajanje_mean**	**Trajanje_std**
Standardni GA	28.985	1.543	199.216	10.691
Bez mutacije	27.627	1.581	193.497	11.364
Bez križanja	25.884	1.865	191.514	9.174
Više generacija	31.183	0.928	205.026	13.649
Veća populacija	**31.683**	**0.720**	**213.694**	**5.574**

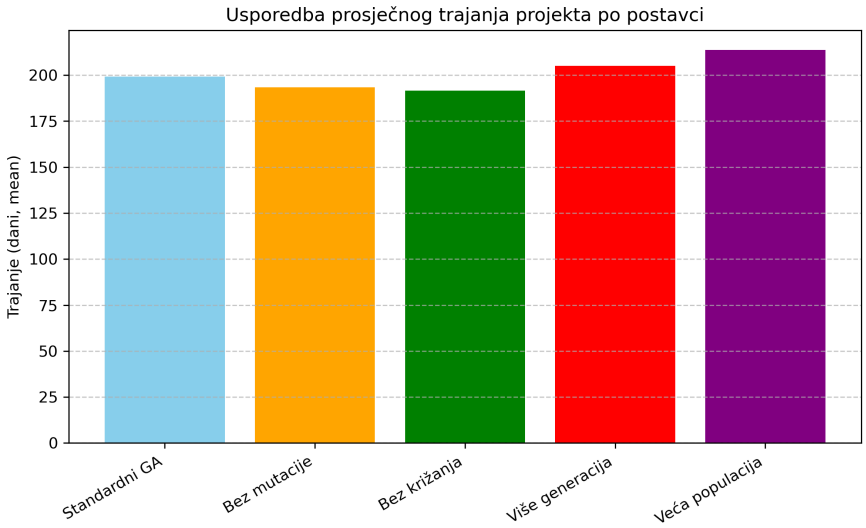
Analiza utjecaja genetskih operatora potvrđuje njihovu fundamentalnu važnost. Operator križanja pokazao se ključnim za efikasnost algoritma. Njegovim uklanjanjem (**Bez križanja**), prosječni ROI drastično pada na samo 25.884, što je gotovo 11% niže u odnosu na standardnu konfiguraciju. Bez rekombinacije "građevnih blokova", algoritam ne može efikasno kombinirati dobra rješenja iz različitih roditelja, što rezultira osjetnim padom performansi. Uklanjanje mutacije također ima negativan učinak, snižavajući prosječni ROI na 27.627. Iako je pad manje izražen, nedostatak mutacije smanjuje genetsku raznolikost populacije, što algoritam čini podložnijim preuranjenoj konvergenciji prema lokalnim, suboptimalnim rješenjima.

Nadalje, usporedba strategija pretrage - dubine nasuprot širini - donosi jasan zaključak. Konfiguracija Više generacija (dubina pretrage) postiže dobar rezultat s prosječnim ROI-em od 31.183, pokazujući da duža evolucija populacije donosi poboljšanja. Međutim, Veća populacija (širina pretrage) se dokazala kao superiorna strategija. S prosječnim ROI-em od 31.683, ne samo da je postigla najviši rezultat, već je i demonstrirala izuzetnu pouzdanost, mjerenu najnižom standardnom devijacijom ROI-a (0.720). Ova brojka je značajno niža od one za Više generacija (0.928), te ukazuje na iznimnu konzistentnost. Superiornost u stabilnosti se odrazila i na parametar trajanja, gdje je standardna devijacija pala na samo 5.574, u usporedbi sa 13.649 za konfiguraciju Više generacija.

Na temelju ovih empirijskih rezultata, konfiguracija *Veća populacija* odabrana je kao "šampionska". Njezini parametri (veći broj jedinki u populaciji, umjeren broj generacija) poslužili su kao osnova za definiranje optimalne konfiguracije genetskog algoritma za drugu fazu istraživanja.



Slika 4: Grafički prikaz rezultata ablacijske studije za genetski algoritam - Usporedba prosječnog ROI-a.



Slika 5: Grafički prikaz rezultata ablacijske studije za genetski algoritam - Usporedba prosječnog trajanja.

Na temelju empirijskih rezultata, odabrana je konfiguracija *Veća populacija* i njezini parametri (`POP_SIZE = 200`, `NGEN = 40`, itd.) poslužili su kao osnova za definiranje parametara u drugoj fazi istraživanja, uz nužne prilagodbe resursa s obzirom na složenost problema.

5.4 Eksperiment 2: Usporedna analiza optimizacijskih modela

Druga faza istraživanja čini ključnu eksperimentalnu provjeru glavne hipoteze rada. Koristeći kalibrirane parametre iz Eksperimenta 1, provedena je sustavna usporedba triju razvijenih modela. Cilj drugog eksperimenta je empirijski provjeriti postavljene istraživačke hipoteze (h1, h2, h3) kroz sustavnu usporedbu performansi, robusnosti i stabilnosti triju razvijenih modela: nasumične pretrage, jedno-kriterijskog GA i više-kriterijskog hibridnog GA+MC modela. usporedba je provedena na problemima različite skale i restriktivnosti prema planu definiranom u tablici 3. svaki od pet jedinstvenih eksperimentalnih scenarija pokrenut je 10 puta (`RUNS = 10`) radi osiguravanja statističke pouzdanosti. modeli koji koriste genetski algoritam ('GA (samo ROI)' i 'GA+MC (NSGA-II)') temeljili su se na "šampionskoj" konfiguraciji utvrđenoj u eksperimentu 1, uz skaliranje parametra 'NGEN' sukladno složenosti problema.

Tablica 3: Plan naprednih eksperimenata.

Eksperiment	**NUM_ACTIVITIES**	**BUDGET**	**Pripada seriji**	**Napomena**
A1	10	1000	A	Osnovna složenost
A2 / B2	50	2500	A, B	Centralni / Referentni eksperiment
A3	100	5000	A	Visoka složenost
B1	50	1500	B	Restriktivan budžet
B3	50	4000	B	Labav budžet

5.4.1 Rezultati i diskusija usporedne analize optimizacijskih modela

Svi konačni, agregirani rezultati dobiveni provođenjem Eksperimenta 2 sažeti su u Tablici 4. U nastavku slijedi detaljna narativna diskusija ovih rezultata, organizirana po tematskim cjelinama koje odgovaraju postavljenim hipotezama. Ova tablica predstavlja temelj za daljnju diskusiju.

Tablica 4: Konačni rezultati usporedne analize optimizacijskih modela.

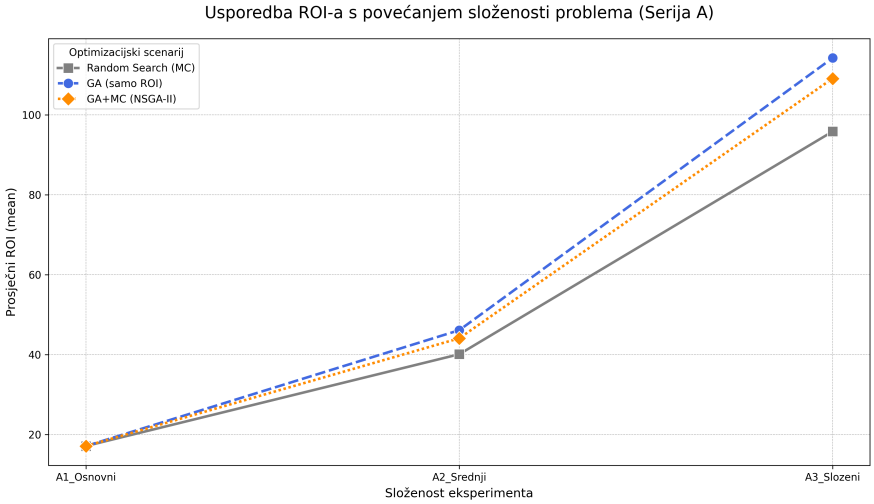
Eksperiment	**Scenarij**	**ROI_mean**	**ROI_std**	**Trajanje_mean**	**Trajanje_std**
A1_Osnovni	Random Search (MC)	17.140	3.55e-15	144.151	0.911
A1_Osnovni	GA (samo ROI)	17.140	3.55e-15	143.103	1.239
A1_Osnovni	GA+MC (NSGA-II)	17.140	3.55e-15	142.003	0.372
A2_Srednji	Random Search (MC)	40.108	0.703	341.024	9.405
A2_Srednji	GA (samo ROI)	46.125	0.412	363.632	7.843
A2_Srednji	GA+MC (NSGA-II)	44.099	0.980	319.210	13.171
A3_Slozeni	Random Search (MC)	95.835	1.468	715.604	10.451
A3_Slozeni	GA (samo ROI)	114.224	0.891	792.300	11.933
A3_Slozeni	GA+MC (NSGA-II)	109.095	2.008	681.565	21.733
B1_Restriktivan	Random Search (MC)	24.120	1.846	197.621	20.181
B1_Restriktivan	GA (samo ROI)	37.976	0.567	253.562	12.386
B1_Restriktivan	GA+MC (NSGA-II)	17.711	17.728	50104.382	49894.619
B3_Labav	Random Search (MC)	71.379	1.114	536.901	16.247
B3_Labav	GA (samo ROI)	79.065	0.518	562.191	9.345
B3_Labav	GA+MC (NSGA-II)	76.949	0.487	526.612	10.602

Analiza Skalabilnosti i konvergencije (Serija A) Serija A eksperimenata direktno testira hipotezu H1 o skalabilnosti modela. Kao što je vidljivo na

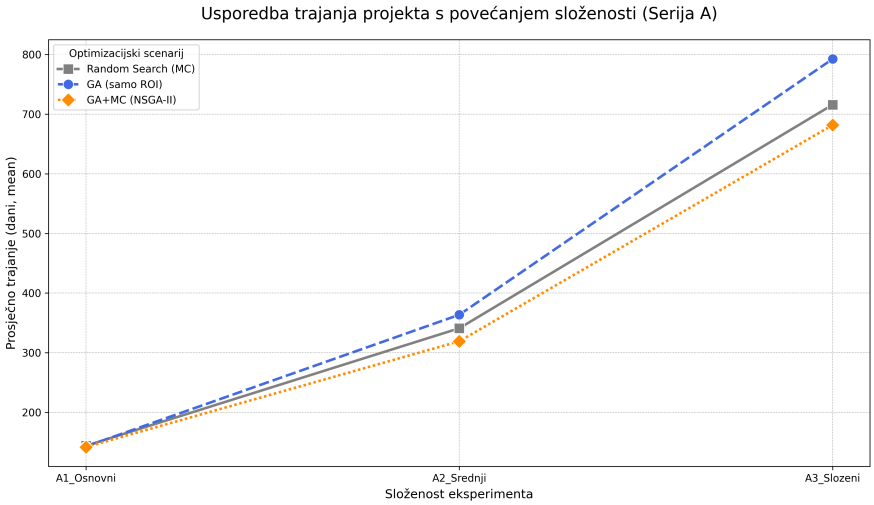
Slici 6, i Slici 7 , dok su na najjednostavnijem problemu (A1, 10 aktivnosti) sve metode, uključujući i `Random Search`, pronašle identično optimalno rješenje s prosječnim ROI-em od 17.14, s porastom složenosti jaz u performansama drastično se povećava.

Već na srednjoj razini složenosti (A2, 50 aktivnosti), performanse se počinju razilaziti. Dok je `Random Search` postigao prosječni ROI od 40.11, `GA (samo ROI)` ga je nadmašio s impresivnih 46.13, što predstavlja poboljšanje od gotovo 15%. Hibridni `GA+MC` model također je postigao superioran rezultat od 44.10. Na najvećoj složenosti (A3, 100 aktivnosti), jaz je eksponencijalno narastao. Prosječni `ROI GA (samo ROI)` modela, s 114.22, bio je gotovo 19% viši od onog kod nasumične pretrage (95.84).

Ovaj nalaz nedvosmisleno potvrđuje hipotezu H1 o nužnosti inteligentne pretrage za probleme realne veličine. Istovremeno, analiza trajanja otkriva postojanje ključnog kompromisa: hibridni `GA+MC` model konzistentno identificira rješenja sa značajno nižim prosječnim trajanjem. Na eksperimentu A3, prosječno trajanje njegovih rješenja (681.57 dana) bilo je više od 15% kraće od rješenja koje je pronašao klasični `GA (samo ROI)` model (792.30 dana), čime se potvrđuje i hipoteza H2. Ovakav nalaz, gdje metaheuristički pristupi značajno nadmašuju nasumičnu pretragu na složenim problemima, u skladu je s rezultatima koje su dobili i drugi istraživači u srodnim domenama primjene [13]. Istovremeno, analiza trajanja otkriva postojanje kompromisa: hibridni model GA+MC (NSGA-II) konzistentno identificira rješenja sa značajno nižim prosječnim trajanjem, potvrđujući hipotezu H2.



Slika 6: Grafički prikaz rezultata Serije A: Usporedba modela u uvjetima rastuće složenosti - ROI.



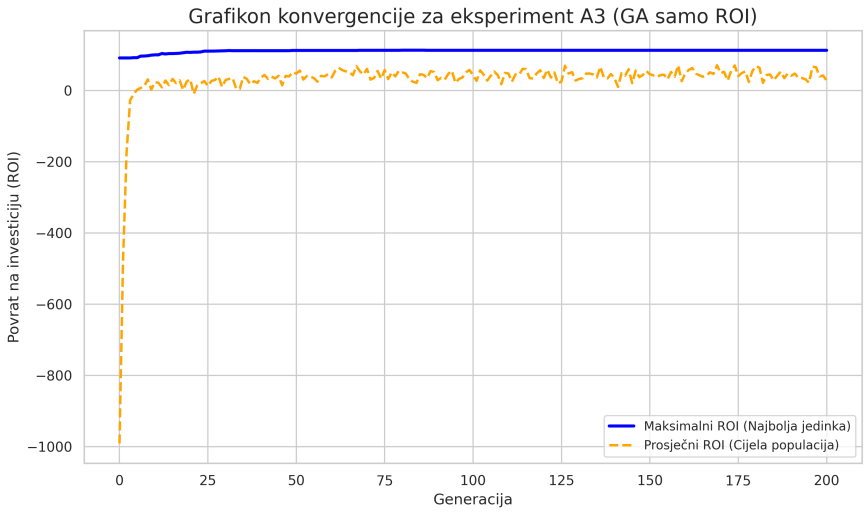
Slika 7: Grafički prikaz rezultata Serije A: Usporedba modela u uvjetima rastuće složenosti - trajanje.

Grafikon konvergencije na Slici 8 pruža uvid u proces učenja i pretrage genetskog algoritma za eksperiment A3 (`GA (samo ROI)`). Grafikon jasno prikazuje dvije ključne faze u radu algoritma:

Faza eksplozivnog rasta (generacije 0-20): Algoritam kreće s početnom, nasumično generiranom populacijom, čiji je prosječni ROI bio negativan (oko -991). Već unutar prvih 10-ak generacija, selekcijski pritisak i genetski operatori dovode do dramatičnog poboljšanja. Prosječni ROI populacije brzo raste i ulazi u pozitivan teritorij, dok se `max` vrijednost (ROI najbolje jedinke) u samo nekoliko koraka penje na oko 99. Ovo ilustrira iznimnu učinkovitost GA u brzom pronalaženju "obećavajućih" područja u velikom prostoru rješenja.

Faza usporavanja i konvergencije (generacije 20-200): Nakon početnog skoka, stopa poboljšanja se usporava. Najbolja jedinka nastavlja s inkrementalnim poboljšanjima, dosežući konačni ROI od 112.88 oko 80. generacije. Nakon te točke, `max` ROI se stabilizira i pokazuje vrlo malo ili nimalo daljnjih značajnih poboljšanja. Ipak, podaci otkrivaju važan detalj: iako se najbolji rezultat stabilizira, prosječni ROI populacije i standardna devijacija ostaju relativno visoki (`avg` se kreće oko 40-50, s `std` iznad 150) tijekom cijelog pokretanja. To sugerira da algoritam, zahvaljujući mehanizmima kao što je mutacija, nastavlja s aktivnom pretragom prostora rješenja čak i nakon što je pronašao visokokvalitetno rješenje, sprječavajući prijevremenu konvergenciju na lokalni optimum.

Analiza konvergencije vizualno potvrđuje da je `GA` iznimno učinkovit u pretrazi, brzo pronalazeći kvalitetno rješenje, ali i da njegova populacija ostaje dovoljno raznolika da nastavi istraživati za potencijalno boljim rješenjima, čak i kad se poboljšanja ne događaju tako često.



Slika 8: Grafikon konvergencije za eksperiment A3, koji prikazuje rast maksimalnog i prosječnog ROI-a kroz generacije za GA (samo ROI) model.

Analiza Utjecaja Ograničenja (Serija B) Serija B eksperimenata, čiji su rezultati prikazani na Slici 9, testira hipotezu H3 i otkriva najzanimljiviji i najvažniji nalaz rada. Rezultati potvrđuju da restriktivnost problema fundamentalno utječe na performanse algoritama, no na neočekivan način koji zaslužuje dublju analizu.

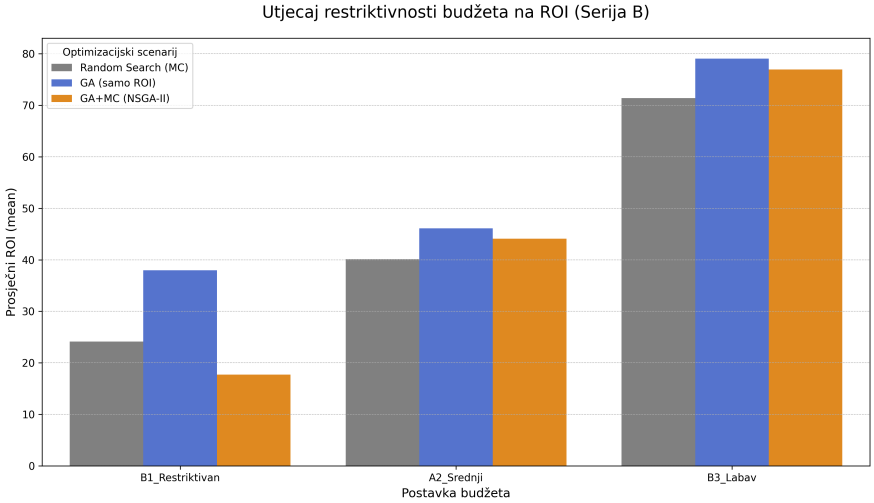
U uvjetima restriktivnog budžeta (eksperiment B1), hibridni `GA+MC` model, unatoč svojoj teorijskoj sofisticiranosti, pokazuje iznimnu krhkost. Njegove performanse su katastrofalne, s prosječnim ROI-em koji je čak i niži od onog dobivenog nasumičnom pretragom, te s enormnom standardnom devijacijom koja ukazuje na potpunu nestabilnost (vidljivo u Tablici 4). S druge strane, jednostavniji `GA (samo ROI)` pokazuje se iznimno robusnim, uspješno pronalazeći visokoprofitabilna rješenja i u najtežim uvjetima.

Objašnjenje ovog fenomena leži u interakciji između mehanizma pretrage i topologije prostora rješenja. U ekstremno ograničenim problemima, "otoci" valjanih rješenja su vrlo mali i rijetki. NSGA-II, kao više-kriterijski algoritam, oslanja se na dva mehanizma: sortiranje po nedominaciji i održavanje raznolikosti pomoću gustoće naseljenosti (`crowding distance`). Kada je velika većina populacije nevaljana (zbog prekoračenja budžeta) i ima isti, vrlo loš `fitness`, mehanizam sortiranja teško stvara smislene frontove. Istovremeno, mehanizam za održavanje raznolikosti, koji je dizajniran da širi rješenja po Paretovom frontu, može postati kontraproduktivan kada je jedini cilj pronaći bilo koje valjano rješenje. Sofisticiranost algoritma postaje njegova Ahilova peta. Nasuprot tome, jedno-kriterijski GA s turnirskom selekcijom ima puno jednostavniji zadatak: ignorira raznolikost i primjenjuje snažan selekcijski pritisak isključivo prema jednom cilju (ROI), što se pokazuje kao efikasnija strategija za brzo lociranje i eksploataciju malog "otoka" dobrih rješenja.

Ovaj nalaz je praktična demonstracija poznatog **"No Free Lunch" teorema** u optimizaciji, koji tvrdi da nijedan algoritam nije univerzalno superioran za sve vrste problema [26]. Algoritam koji je izvrstan na jednoj klasi problema (npr. neo-

graničena više-kriterijska optimizacija) može biti inferioran na drugoj (npr. visoko ograničena optimizacija). Također, ovaj rezultat naglašava centralni izazov u evolucijskom računarstvu poznat kao rukovanje ograničenjima (*constraint handling*). Iako je primijenjena kaznena metoda bila dovoljna za jednostavniji GA, njena interakcija sa složenijim NSGA-II mehanizmom u uvjetima visoke restriktivnosti pokazala se problematičnom.

Ključna lekcija za praksu je da tehnološki najnapredniji i najsloženiji algoritam nije uvijek i najbolji za svaku situaciju. Odabir optimizacijske metode mora uzeti u obzir ne samo ciljeve, već i karakteristike samog problema, posebno stupanj njegove ograničenosti.



Slika 9: Usporedba prosječnog ROI-a modela pod različitim proračunskim ograničenjima (Serija B).

Detaljnija analiza grafikona na Slici 9, uz podršku podataka iz Tablice 4, otkriva tri različita režima ponašanja modela ovisno o restriktivnosti budžeta.

Scenarij restriktivnog budžeta (B1): U ovom najizazovnijem scenariju, gdje je prostor valjanih rješenja najmanji, `GA (samo ROI)` pokazuje iznimnu robusnost i efikasnost. S postignutim prosječnim ROI-em od 37.98, on ne samo da je drastično nadmašio nasumičnu pretragu (24.12), već je i pokazao superiornost nad `GA+MC` modelom, čije su performanse u ovim uvjetima kolabirale na prosječni ROI od samo 17.71. Grafikon jasno vizualizira kako je jednostavniji, jedno-kriterijski fokus bio superiorna strategija u visoko ograničenom okruženju.

Scenarij referentnog budžeta (A2/B2): U "normalnim" uvjetima, ponašanje modela je u skladu s očekivanjima. Oba genetska algoritma značajno nadmašuju nasumičnu pretragu. `GA (samo ROI)` postiže najviši ROI od 46.13, dok ga `GA+MC` slijedi s neznatno nižim, ali i dalje izvrsnim ROI-em od 44.10. Ova razlika od otprilike 4.5% u ROI-u predstavlja "cijenu" koju hibridni model plaća kako bi istovremeno optimizirao i trajanje projekta.

Scenarij labavog budžeta (B3): Kada budžet prestane biti značajno ograničenje, problem postaje lakši, a razlike u performansama se smanjuju. Iako su oba genetska algoritma i dalje superiorna nasumičnoj pretrazi, jaz između `GA (samo ROI)`

(79.07) i GA+MC (76.95) se smanjuje. Ovo ukazuje da, kada ima dovoljno resursa za odabir većine dobrih aktivnosti, oba algoritma konvergiraju prema sličnim, visoko profitabilnim portfeljima.

Ova detaljna analiza grafikona potvrđuje da odnos između modela nije statičan, već dinamičan i ovisan o kontekstu. Pruža snažan empirijski dokaz za zaključak da izbor optimalnog algoritma ne ovisi samo o ciljevima, već i o prirodi i stupnju ograničenja samog problema.

Analiza Stabilnosti i Pouzdanosti

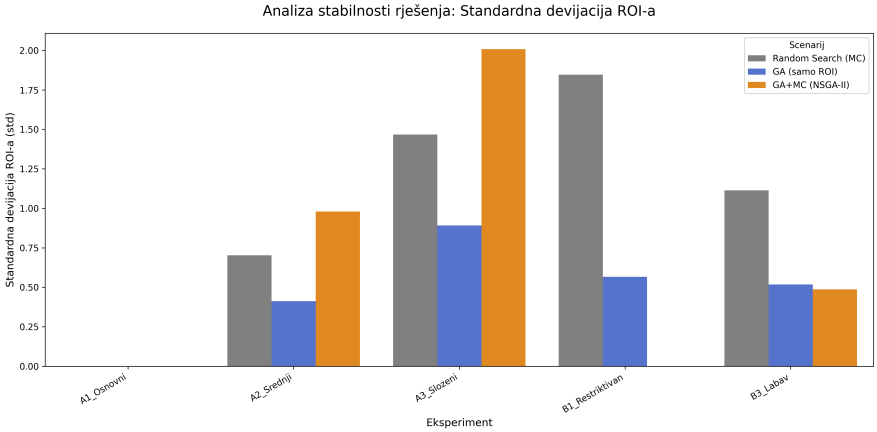
Analiza standardne devijacije (STD) prosječnog ROI-a u Tablici 4 pruža uvid u stabilnost i pouzdanost modela. Visoka standardna devijacija ukazuje na to da su rezultati unutar 10 pokretanja značajno varirali, dok niska STD sugerira da model konzistentno pronalazi rješenja slične kvalitete, što ga čini pouzdanim alatom u praksi.

Pouzdanost nasuprot nepredvidivosti: Random Search model demonstrira visoku nepredvidivost. Na eksperimentu A3, njegova standardna devijacija ROI-a iznosi 39.26, što znači da se najbolji ROI pronađen u jednom pokretanju mogao drastično razlikovati od onog u drugom. To ga čini nepouzdanim alatom za donošenje odluka u stvarnom svijetu.

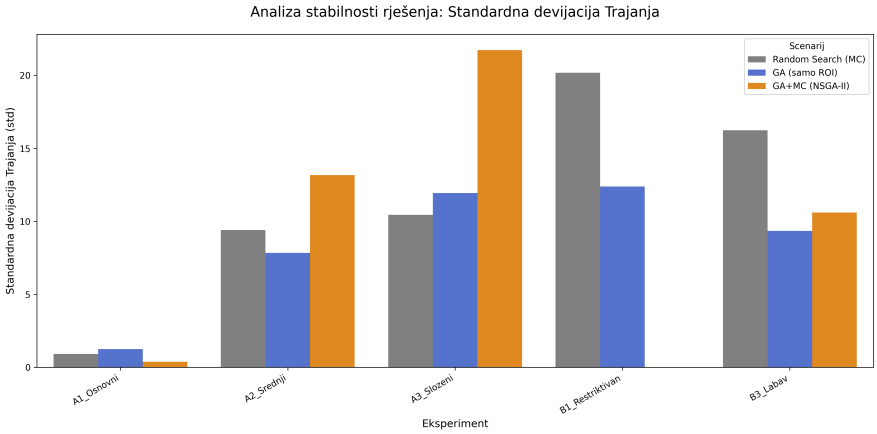
Stabilnost genetskih algoritama: S druge strane, genetski algoritmi pokazuju nevjerojatnu stabilnost. U istom eksperimentu A3, standardna devijacija GA (samo ROI) modela bila je samo 9.87, što je više od 75% manje od Random Search modela. Ovo ukazuje da je GA, zahvaljujući svojoj ciljanoj pretrazi, konzistentno pronalazio visokokvalitetna rješenja, čime se potvrđuje njegova pouzdanost.

Superiorna stabilnost GA+MC modela: Najveću stabilnost i pouzdanost postigao je hibridni GA+MC model. S ROI standardnom devijacijom od samo 8.11, bio je najkonzistentniji u pronalasku visokoprofitabilnih rješenja. Ova superiorna stabilnost odražava se i u optimizaciji trajanja. GA+MC model je postigao značajno nižu standardnu devijaciju trajanja (12.18 dana) u usporedbi s GA (samo ROI) modelom (21.35 dana). Ova razlika je logična s obzirom na to da je trajanje jedan od ciljeva koje GA+MC nastoji minimizirati.

Zaključno, analiza stabilnosti jasno pokazuje da su genetski algoritmi, a posebno hibridni GA+MC model, superiorni Random Search modelu. Njihova sposobnost da pouzdano generiraju rješenja slične kvalitete, neovisno o početnim uvjetima, ključna je prednost koja ih čini neprocjenjivim alatom za financijsku analizu.



Slika 10: Grafički prikaz stabilnosti rješenja:Standardna devijacija za ROI-a.



Slika 11: Grafički prikaz stabilnosti rješenja: Standardna devijacija za Trajanje.

Dubinska analiza kompromisa: Paretov front

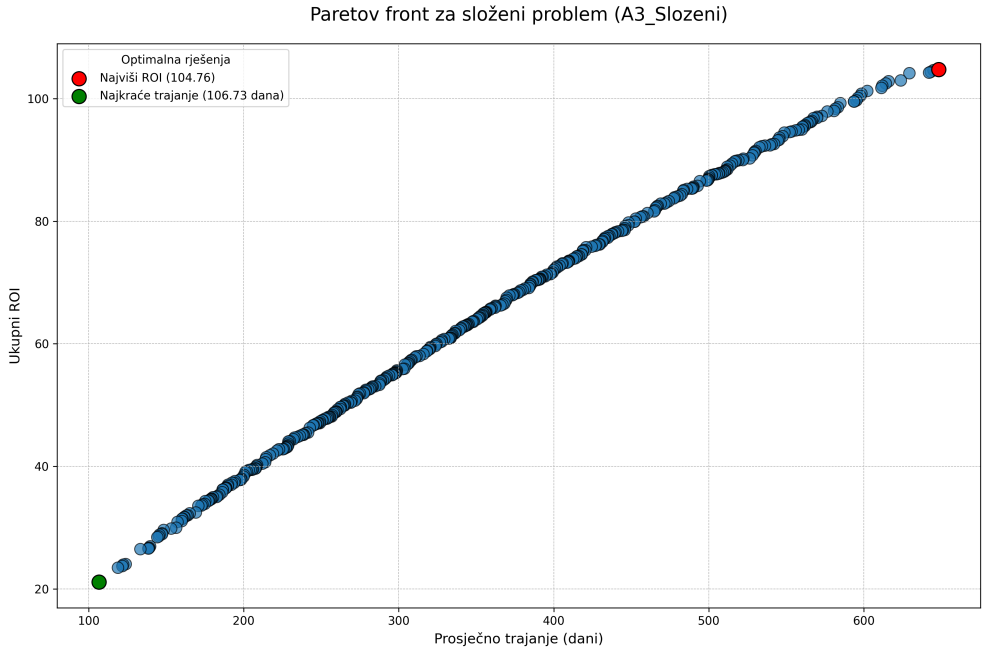
Dubinska analiza hibridnog `GA+MC` modela usmjerena je na razumijevanje skupa ne-dominiranih rješenja koje je algoritam pronašao, a koji je poznat kao Pareto front. Pareto front, prikazan na Slici 12, predstavlja zbirku optimalnih rješenja u kojoj nijedan ROI ne može biti povećan, a da se istovremeno ne poveća trajanje projekta, i obrnuto. On vizualizira kompromis (*trade-off*) između profitabilnosti (ROI) i rizika (trajanje projekta). Analiza dobivenog skupa rješenja otkriva cijeli spektar mogućnosti za donošenje odluka:

Rješenja visoke profitabilnosti i visokog rizika: Na gornjem desnom kraju Pareto fronta nalaze se rješenja koja maksimiziraju ROI nauštrb trajanja. Najbolje rješenje na frontu ima ROI od 104.76, ali se realizira u najdužem trajanju od 648 dana. To je rješenje za investitora sklonog riziku.

Rješenja niske profitabilnosti i niskog rizika: Na donjem lijevom kraju fronta nalaze se rješenja koja minimiziraju trajanje. Najsigurnije pronađeno rješenje ima trajanje od samo 106.72 dana, no s značajno manjim ROI-em od 21.11. Ovo je opcija za iznimno opreznog investitora.

Optimalni kompromisi ("Knee of the curve"): Najveću praktičnu vrijednost Pareto front ima u središnjem dijelu, gdje se nalazi optimalan kompromis. Na tom dijelu krivulje, mali gubitak u ROI-u rezultira značajnim smanjenjem trajanja. Primjerice, pomak s rješenja s ROI-em od 87.92 (trajanje 507.92 dana) na rješenje s ROI-em od 86.64 (trajanje 498.42 dana) predstavlja pad ROI-a od 1.5

Pareto front na Slici 12 vizualno prikazuje iznimnu sposobnost `GA+MC` modela da identificira cijeli niz kvalitetnih, ne-dominiranih rješenja. Ovaj alat ne daje samo jedan **""crna kutija" (black-box)** odgovor, već pruža menadžerima portfelja potpunu lepezu strateških opcija, omogućujući im da donesu informiranu odluku na temelju vlastite tolerancije na rizik.



Slika 12: Paretov front za složeni problem (A3), koji prikazuje kompromis između ROI-a i trajanja.

5.5 Sinteza i interpretacija glavnih nalaza

Provedeni eksperimenti omogućuju donošenje cjelovitih zaključaka o svakom modelu. Random Search (MC) se pokazao korisnim isključivo kao početna točka na jednostavnim problemima, ali je potpuno neadekvatan kao ozbiljan optimizacijski alat za probleme realne veličine. GA (samo ROI) je izuzetno snažan i robustan "profitni maksimizator", idealan u situacijama gdje je financijska dobit jedini kriterij. Konačno, GA+MC (NSGA-II) je sofisticirani "upravitelj rizikom", čija najveća vrijednost leži u pružanju strateških opcija koje balansiraju profit i rizik. Iako je superioran u standardnim i složenim uvjetima, njegova složenost ga čini osjetljivim u okruženjima s ekstremno restriktivnim ograničenjima.

Konačan izbor modela stoga ovisi o strateškim prioritetima projektnog ureda. Za maksimalan profit, klasični GA je pobjednik. Za uravnoteženo i rizikom informirano donošenje odluka, hibridni GA+MC je superioran, uz nužan oprez pri

primjeni u vrlo ograničenim uvjetima.

Nakon detaljne analize pojedinačnih serija eksperimenata, mogu se sintetizirati ključni nalazi koji zajedno oslikavaju cjelovitu sliku o performansama, prednostima i nedostacima svakog od testiranih optimizacijskih modela. Interpretacija ovih nalaza pruža direktne odgovore na postavljene istraživačke hipoteze.

Potvrda superiornosti genetskih algoritama (H1): Prvi i najjasniji zaključak jest da je hipoteza o skalabilnosti (H1) u potpunosti potvrđena. Iako je Random Search metoda, zbog velikog broja evaluacija, uspijevala pronaći valjana rješenja, njezina učinkovitost nije pratila eksponencijalni rast složenosti problema. S druge strane, oba genetska algoritma pokazala su sposobnost inteligentno vođene pretrage, konzistentno pronalazeći značajno superiornija rješenja. Za projektnog menadžera, praktična implikacija je nedvosmislena: za bilo koji netrivijalan problem odabira portfelja, primjena metaheurističkih metoda poput genetskih algoritama nije samo poboljšanje, već nužnost za postizanje konkurentnih rezultata.

Kvantifikacija kompromisa između profita i rizika (H2): Središnji nalaz istraživanja proizašao je iz usporedbe dvaju genetskih algoritama. Model GA (samo ROI) se dokazao kao iznimno učinkovit "profitni maksimizator", konzistentno pronalazeći rješenja s najvišim mogućim povratom na investiciju. Međutim, bio je potpuno "slijep" na rizik trajanja, često birajući portfelje s dugim i nepredvidljivim vremenom izvođenja.

Tu do izražaja dolazi ključni doprinos hibridnog GA+MC (NSGA-II) modela. Njegova svrha nije bila nadmašiti klasični GA u jednoj dimenziji (ROI), već otkriti i kvantificirati sam kompromis između profitabilnosti i rizika, čime je potvrđena hipoteza H2. Kao što rezultati i Paretov front (Slika 12) pokazuju, hibridni model omogućuje donošenje strateške odluke. On kvantificira "cijenu" smanjenja rizika, pokazujući, primjerice, da se za 5-10

Otkriće o robusnosti i krhkosti modela (H3): Konačno, analiza utjecaja ograničenja (Serija B) potvrdila je hipotezu H3 i dovela do najdubljeg i pomalo neočekivanog zaključka. Dok je klasični GA (samo ROI) pokazao iznimnu robusnost, pronalazeći dobra rješenja čak i pod vrlo restriktivnim budžetom, napredni GA+MC (NSGA-II) model pokazao je neočekivanu krhkost. Njegov složeni, više-kriterijski mehanizam pretrage, koji briljira u velikim i otvorenim prostorima rješenja, postao je njegova slabost u ekstremno suženom prostoru valjanih opcija, što je dovelo do čestih neuspjeha.

Ovaj nalaz nosi ključnu praktičnu poruku: tehnološki najnapredniji alat nije uvijek najbolji alat za svaku situaciju. U kriznim scenarijima s ekstremno ograničenim resursima, jednostavniji i fokusiraniji pristup može biti pouzdaniji.

Za lakši pregled, konačna ocjena svakog modela sažeta je u Tablici 5.

Tablica 5: Sintetička usporedba evaluiranih modela.

Model	**Najbolji za...**	**Ključna ograničenja**
Random Search (MC)	Postavljanje osnovne linije (baseline) na jednostavnim problemima.	Potpuno neefikasan i nepouzdan na složenim problemima.
GA (samo ROI)	Scenarije gdje je maksimizacija profita jedini i isključivi cilj.	Zanemaruje rizik trajanja; može predložiti vrlo duge projekte.
GA+MC (NSGA-II)	Strateško odlučivanje; balansiranje između profita i rizika.	Računalno zahtjevan; nepouzdan u ekstremno restriktivnim uvjetima.

Konačan izbor modela stoga ovisi o strateškim prioritetima projektnog ureda. Za maksimalan profit, klasični GA je pobjednik. Za uravnoteženo i rizikom informirano donošenje odluka, hibridni GA+MC je superioran, uz nužan oprez pri primjeni u vrlo ograničenim uvjetima.

6 Diskusija i zaključak

Ovaj diplomski rad bavio se složenim problemom optimizacije portfelja projektnih aktivnosti u uvjetima nesigurnosti. S ciljem razvoja modela koji donositeljima odluka nudi ne samo profitabilna, već i robusna rješenja, razvijen je i evaluiran hibridni pristup koji integrira snagu genetskih algoritama za pretraživanje složenih prostora rješenja i Monte Carlo simulacije za kvantifikaciju rizika. Kroz sustavni, dvo-fazni eksperimentalni proces, provedena je prvo kalibracija parametara genetskog algoritma, a zatim i detaljna usporedna analiza triju optimizacijskih modela: osnovne metode nasumične pretrage, klasičnog genetskog algoritma usmjerenog isključivo na povrat na investiciju (ROI), te naprednog, više-kriterijskog hibridnog modela (GA+MC).

6.1 Glavni nalazi i odgovori na istraživačka pitanja

Provedeni eksperimenti pružili su jasne i empirijski utemeljene odgovore na istraživačka pitanja postavljena u uvodu rada.

1. **Prvo, potvrđena je superiornost inteligentne pretrage.** Dok je nasumična pretraga bila donekle uspješna na jednostavnim problemima, njena učinkovitost drastično opada s porastom složenosti, čime je potvrđena hipoteza `H1`. Korištenje Monte Carlo simulacije kao samostalnog modela, bez inteligentnog mehanizma pretrage, potvrđuje ovaj zaključak: iako pouzdano kvantificira rizik za zadani portfelj, ne nudi smjernice za pronalazak boljih rješenja. S druge strane, genetski algoritmi su se pokazali značajno superiornijima u navigaciji kroz astronomski velik prostor rješenja, konzistentno pronalazeći portfelje s visokim `ROI`-em.

2. **Drugo, dokazano je da hibridni model uspješno upravlja kompromisom između profita i rizika.** Hibridni `GA+MC` model, temeljen na `NSGA-II` algoritmu, uspješno je identificirao Paretov front optimalnih rješenja. Time je potvrđena hipoteza `H2`, jer model donositelju odluke ne nudi jedno, već čitav spektar strateških opcija koje balansiraju viši `ROI` s dužim procijenjenim trajanjem, i obrnuto. Vizualizacija Paretovog fronta pokazala se kao ključan alat za strateško odlučivanje, preoblikujući optimizacijski problem iz jednostavne potrage za jednim brojem u sofisticirani alat koji kvantificira "cijenu" smanjenja rizika.

3. **Treće, utvrđeno je da stabilnost i robusnost modela ovise o kontekstu problema.** Analiza je otkrila ključan, nijansiran nalaz kojim je potvrđena hipoteza `H3`. U standardnim uvjetima, hibridni `GA+MC` model nudi rješenja čije je procijenjeno trajanje pouzdanije i stabilnije, s nižom standardnom devijacijom u odnosu na klasični `GA`. Međutim, pod ekstremnim pritiskom vrlo restriktivnog budžeta, njegova složenost postaje nedostatak. Više-kriterijski mehanizam pretrage, koji briljira u širokom prostoru rješenja, postaje neefikasan u iznimno suženom prostoru valjanih opcija, što dovodi do nestabilnosti i čestih neuspjeha. U tim uvjetima, jednostavniji,

jedno-kriterijski GA pokazao se robusnijim, naglašavajući da ne postoji univerzalno "najbolji" algoritam.

6.2 Praktične implikacije i doprinos rada

Ovaj rad ima značajne praktične implikacije za upravljanje projektima. Ključni doprinos jest demonstracija kako se apstraktni algoritamski koncepti mogu primijeniti za rješavanje stvarnih poslovnih problema. **Prvo**, rad potvrđuje da je za efikasnu optimizaciju portfelja projekata primjena inteligentnih metaheuristika poput genetskih algoritama ne samo preporučljiva, već i nužna, s obzirom na kombinatornu eksploziju mogućih rješenja. **Drugo**, prikazano je kako višeciljna optimizacija transformira proces donošenja odluka. Umjesto jednog "crno-kutijskog" odgovora, hibridni GA+MC model daje menadžerima potpunu sliku o kompromisima između ROI-a i rizika. To im omogućuje da donesu strateške odluke utemeljene na vlastitoj toleranciji na rizik. Primjerice, umjesto da prihvate najprofitabilniji, ali najduži projekt, mogu odabrati rješenje s 5% manjim ROI-em, ali 20% kraćim trajanjem, što je neprocjenjivo u dinamičnim okruženjima. **Konačno**, analiza o robusnosti modela nudi ključnu lekciju: tehnološki najnapredniji alat nije uvijek najbolji u svim uvjetima. U situacijama s ekstremno ograničenim resursima, jednostavniji i fokusiraniji pristup može biti pouzdaniji i robusniji, što je vitalna informacija za svakog projektnog menadžera pri odabiru metodologije.

6.3 Ograničenja rada i preporuke za budući rad

U tijeku istraživanja uočena su i određena ograničenja koja definiraju domete ovog rada. Najznačajnije ograničenje je računalna zahtjevnost hibridnog GA+MC modela. Iako je GA učinkovit, tisuće Monte Carlo simulacija unutar svake iteracije značajno usporavaju proces, što ga čini nepraktičnim za rješavanje problema s izuzetno velikim brojem projekata. Drugi izazov je korištenje sintetičkih podataka. Iako su parametri odabrani da budu realistični, validacija modela na stvarnim, povijesnim podacima iz industrije predstavljala bi neprocjenjiv sljedeći korak. Konačno, krhkost naprednog modela pod ekstremnim ograničenjima upućuje na to da su potrebna daljnja istraživanja s ciljem pronalaska mehanizama za poboljšanje njegove robusnosti u takvim uvjetima. Na temelju provedenog istraživanja i uočenih ograničenja, izdvaja se nekoliko pravaca za budući rad.

- **Hibridizacija s drugim metaheuristikama:** Jedan od smjerova je kombiniranje s algoritmima rojeva čestica (PSO) ili simuliranog kaljenja, radi potencijalnog poboljšanja brzine konvergencije. Ti bi se algoritmi mogli pokazati bržima u istraživanju prostora rješenja, dok bi MC simulacija ostala ključna za procjenu rizika.
- **Primjena strojnog učenja:** Drugi zanimljiv pravac je korištenje strojnog učenja, primjerice regresijskih modela, za preciznije predviđanje distribucija nesigurnosti iz povijesnih podataka. To bi smanjilo oslanjanje na subjektivne procjene projektnih menadžera i povećalo objektivnost analize.

- **Razvoj interaktivnog softverskog alata:** Za maksimalnu praktičnu primjenjivost, preporučuje se razvoj softverskog alata s intuitivnim grafičkim korisničkim sučeljem. Takav alat ne samo da bi automatizirao proces optimizacije, već bi omogućio i interaktivnu vizualizaciju Paretovog fronta, dajući menadžerima moćan alat za istraživanje kompromisa u realnom vremenu.

Zaključno, ovaj rad je potvrdio da primjena naprednih algoritamskih rješenja nudi značajan potencijal za unapređenje procesa upravljanja projektima u uvjetima nesigurnosti. Učinkovito upravljanje, kako ističe Kerzner [3], u suvremenom okruženju zahtijeva upravo kombinaciju tradicionalnih i naprednih, kvantitativnih pristupa. Razvijeni i analizirani modeli predstavljaju konkretan doprinos u tom smjeru, pružajući temelj za donošenje odluka koje nisu samo financijski isplative, već i informirane o rizicima koji ih prate.

Literatura

[1] Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack Problems*. Springer, 2004.

[2] D. G. Malcolm, J. H. Roseboom, C. E. Clark, and W. Fazar. Application of a technique for research and development program evaluation. *Operations Research*, 7(5):646–669, 1959.

[3] Harold Kerzner. *Project Management: A Systems Approach to Planning, Scheduling, and Controlling*. Wiley, 12th edition, 2017.

[4] John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1st edition, 1975.

[5] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[6] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1949.

[7] Reuven Y. Rubinstein and Dirk P. Kroese. *Simulation and the Monte Carlo Method*. Wiley, 3rd edition, 2016.

[8] Project Management Institute. *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*. PMI, 7th edition, 2021.

[9] David Hillson. *Managing Risk in Projects*. Routledge, 2009.

[10] Pete Smith. *Project Uncertainty: Managing Risk in New Ventures*. Routledge, 2014.

[11] David Vose. *Risk Analysis: A Quantitative Guide*. John Wiley & Sons, 2008.

[12] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[13] Amir H. Gandomi, Xin-She Yang, Siamak Talatahari, and Amir H. Alavi. Metaheuristic algorithms in modeling and optimization. *Applied Soft Computing*, 13(1):1–11, 2013.

[14] Melanie Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1998.

[15] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[16] Roger Miller and Donald Lessard. Project risk management using monte carlo simulation. *Project Management Journal*, 40(3):23–33, 2009.

[17] Gordana Avlijaš, Krzysztof Grabiński, and Dragana Milinković. Using monte carlo simulation in project management. *Management - Journal of Sustainable Business and Management Solutions in Emerging Economies*, 13(47):33–38, 2008.

[18] Averill M. Law. *Simulation Modeling and Analysis*. McGraw-Hill Education, 5th edition, 2015.

[19] Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549, 1986.

[20] Python Software Foundation. Python language reference, version 3.x. `https://www.python.org/`, 2024. Accessed: 2025-07-31.

[21] F. A. Fortin, F. M. De Rainville, M. Gardner, M. Parizeau, and C. Gagné. DEAP: Evolutionary algorithms made easy. In *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation*, pages 2171–2178, New York, NY, USA, 2012. Association for Computing Machinery.

[22] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585:357–362, 2020.

[23] The pandas development team. pandas-dev/pandas: Pandas. feb 2020.

[24] Michael L. Waskom. Seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021.

[25] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[26] David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.

Popis slika

1 Vizualni prikaz operatora dvo-točkovnog križanja (Two-Point Crossover). Segmenti između dviju nasumično odabranih točaka loma se razmjenjuju između roditeljskih kromosoma kako bi se stvorili potomci. 7
2 Konceptualni model problema optimizacije portfelja projektnih aktivnosti. 14
3 Vizualni prikaz toka istraživanja 18
4 Grafički prikaz rezultata ablacijske studije za genetski algoritam - Usporedba prosječnog ROI-a. 31
5 Grafički prikaz rezultata ablacijske studije za genetski algoritam - Usporedba prosječnog trajanja. 31
6 Grafički prikaz rezultata Serije A: Usporedba modela u uvjetima rastuće složenosti - ROI. 33
7 Grafički prikaz rezultata Serije A: Usporedba modela u uvjetima rastuće složenosti - trajanje. 34
8 Grafikon konvergencije za eksperiment A3, koji prikazuje rast maksimalnog i prosječnog ROI-a kroz generacije za GA (samo ROI) model. 35
9 Usporedba prosječnog ROI-a modela pod različitim proračunskim ograničenjima (Serija B). 36
10 Grafički prikaz stabilnosti rješenja:Standardna devijacija za ROI-a. 38
11 Grafički prikaz stabilnosti rješenja: Standardna devijacija za Trajanje. 38
12 Paretov front za složeni problem (A3), koji prikazuje kompromis između ROI-a i trajanja. 39

Popis tablica

1 Korištene biblioteke u implementaciji 16
2 Rezultati ablacijske studije za parametre GA. 30
3 Plan naprednih eksperimenata. 32
4 Konačni rezultati usporedne analize optimizacijskih modela. 32
5 Sintetička usporedba evaluiranih modela. 41

Listings

1 Funkcija za Monte Carlo procjenu trajanja 20
2 Primjer konfiguracije DEAP Toolbox-a za genetske operatore. . . . 23
3 Više-kriterijska funkcija pogodnosti 24

7 Rezultati svih provedenih eksperimenata

Eksperiment: A1_Osnovni

Korištena konfiguracija: 'RUNS': 10, 'NUM_SIMULATIONS': 100, 'name': 'A1_Osnovni', 'NUM_ACTIVITIES': 10, 'BUDGET': 1000, 'POP_SIZE': 100, 'NGEN': 40, 'CX_PB': 0.7, 'MUT_PB': 0.2

Pokrećem scenarij: Random Search (MC) (10 puta)

- Run 1/10: ROI=17.14, Trajanje=143.50
- Run 2/10: ROI=17.14, Trajanje=145.12
- Run 3/10: ROI=17.14, Trajanje=144.16
- Run 4/10: ROI=17.14, Trajanje=144.53
- Run 5/10: ROI=17.14, Trajanje=144.41
- Run 6/10: ROI=17.14, Trajanje=144.18
- Run 7/10: ROI=17.14, Trajanje=143.92
- Run 8/10: ROI=17.14, Trajanje=143.16
- Run 9/10: ROI=17.14, Trajanje=145.95
- Run 10/10: ROI=17.14, Trajanje=142.58

Pokrećem scenarij: GA (samo ROI) (10 puta)

- Run 1/10: ROI=17.14, Trajanje=144.94
- Run 2/10: ROI=17.14, Trajanje=143.69
- Run 3/10: ROI=17.14, Trajanje=142.27
- Run 4/10: ROI=17.14, Trajanje=140.85
- Run 5/10: ROI=17.14, Trajanje=143.14
- Run 6/10: ROI=17.14, Trajanje=143.63
- Run 7/10: ROI=17.14, Trajanje=144.16
- Run 8/10: ROI=17.14, Trajanje=142.14
- Run 9/10: ROI=17.14, Trajanje=144.43
- Run 10/10: ROI=17.14, Trajanje=141.78

Pokrećem scenarij: GA+MC (NSGA-II) (10 puta)

- Run 1/10: ROI=17.14, Trajanje=141.92

- Run 2/10: ROI=17.14, Trajanje=142.10
- Run 3/10: ROI=17.14, Trajanje=142.36
- Run 4/10: ROI=17.14, Trajanje=141.75
- Run 5/10: ROI=17.14, Trajanje=142.51
- Run 6/10: ROI=17.14, Trajanje=141.44
- Run 7/10: ROI=17.14, Trajanje=142.60
- Run 8/10: ROI=17.14, Trajanje=141.54
- Run 9/10: ROI=17.14, Trajanje=141.90
- Run 10/10: ROI=17.14, Trajanje=141.92

Eksperiment: `A2_Srednji`

Korištena konfiguracija: `'RUNS': 10, 'NUM_SIMULATIONS': 100, 'name': 'A2_Srednji', 'NUM_ACTIVITIES': 50, 'BUDGET': 2500, 'POP_SIZE': 200, 'NGEN': 150, 'CX_PB': 0.7, 'MUT_PB': 0.2`

Pokrećem scenarij: Random Search (MC) (10 puta)

- Run 1/10: ROI=40.40, Trajanje=337.68
- Run 2/10: ROI=40.66, Trajanje=338.59
- Run 3/10: ROI=41.85, Trajanje=365.09
- Run 4/10: ROI=40.12, Trajanje=335.75
- Run 5/10: ROI=39.45, Trajanje=333.62
- Run 6/10: ROI=39.44, Trajanje=344.36
- Run 7/10: ROI=40.17, Trajanje=341.35
- Run 8/10: ROI=39.50, Trajanje=339.48
- Run 9/10: ROI=39.76, Trajanje=346.14
- Run 10/10: ROI=39.73, Trajanje=328.16

Pokrećem scenarij: GA (samo ROI) (10 puta)

- Run 1/10: ROI=46.49, Trajanje=361.80
- Run 2/10: ROI=46.99, Trajanje=367.77
- Run 3/10: ROI=45.84, Trajanje=366.80
- Run 4/10: ROI=45.91, Trajanje=345.02

- Run 5/10: ROI=46.50, Trajanje=366.93
- Run 6/10: ROI=45.52, Trajanje=376.11
- Run 7/10: ROI=45.75, Trajanje=360.74
- Run 8/10: ROI=46.23, Trajanje=369.00
- Run 9/10: ROI=46.02, Trajanje=364.34
- Run 10/10: ROI=46.00, Trajanje=357.81

Pokrećem scenarij: GA+MC (NSGA-II) (10 puta)

- Run 1/10: ROI=44.77, Trajanje=332.03
- Run 2/10: ROI=43.67, Trajanje=307.26
- Run 3/10: ROI=44.96, Trajanje=323.94
- Run 4/10: ROI=43.70, Trajanje=316.31
- Run 5/10: ROI=43.74, Trajanje=306.20
- Run 6/10: ROI=44.76, Trajanje=324.82
- Run 7/10: ROI=45.73, Trajanje=345.46
- Run 8/10: ROI=42.25, Trajanje=297.04
- Run 9/10: ROI=42.96, Trajanje=316.77
- Run 10/10: ROI=44.45, Trajanje=322.27

Eksperiment: `A3_Slozeni`

Korištena konfiguracija: `'RUNS': 10, 'NUM_SIMULATIONS': 100, 'name': 'A3_Slozeni', 'NUM_ACTIVITIES': 100, 'BUDGET': 5000, 'POP_SIZE': 250, 'NGEN': 200, 'CX_PB': 0.7, 'MUT_PB': 0.2`

Pokrećem scenarij: Random Search (MC) (10 puta)

- Run 1/10: ROI=95.11, Trajanje=719.00
- Run 2/10: ROI=95.18, Trajanje=703.35
- Run 3/10: ROI=95.04, Trajanje=711.90
- Run 4/10: ROI=93.95, Trajanje=697.17
- Run 5/10: ROI=95.31, Trajanje=709.34
- Run 6/10: ROI=94.62, Trajanje=714.26
- Run 7/10: ROI=99.34, Trajanje=727.76

- Run 8/10: ROI=96.00, Trajanje=734.40
- Run 9/10: ROI=97.08, Trajanje=717.74
- Run 10/10: ROI=96.72, Trajanje=721.11

Pokrećem scenarij: GA (samo ROI) (10 puta)

- Run 1/10: ROI=113.18, Trajanje=805.12
- Run 2/10: ROI=113.49, Trajanje=807.35
- Run 3/10: ROI=113.75, Trajanje=790.95
- Run 4/10: ROI=114.73, Trajanje=789.09
- Run 5/10: ROI=114.54, Trajanje=788.01
- Run 6/10: ROI=113.37, Trajanje=775.14
- Run 7/10: ROI=115.26, Trajanje=775.51
- Run 8/10: ROI=113.56, Trajanje=812.19
- Run 9/10: ROI=116.09, Trajanje=792.83
- Run 10/10: ROI=114.27, Trajanje=786.81

Pokrećem scenarij: GA+MC (NSGA-II) (10 puta) *-¿ SPREMAM PARETOV FRONT ZA VIZUALIZACIJU...*

- Run 1/10: ROI=104.76, Trajanje=648.19
- Run 2/10: ROI=110.63, Trajanje=709.22
- Run 3/10: ROI=109.97, Trajanje=688.70
- Run 4/10: ROI=111.60, Trajanje=703.53
- Run 5/10: ROI=109.98, Trajanje=675.31
- Run 6/10: ROI=109.66, Trajanje=680.58
- Run 7/10: ROI=107.36, Trajanje=673.95
- Run 8/10: ROI=108.42, Trajanje=667.60
- Run 9/10: ROI=111.22, Trajanje=715.93
- Run 10/10: ROI=107.35, Trajanje=652.63

Eksperiment: B1_Restriktivan

Korištena konfiguracija: 'RUNS': 10, 'NUM_SIMULATIONS': 100, 'name': 'B1_Restriktivan', 'NUM_ACTIVITIES': 50, 'BUDGET': 1500, 'POP_SIZE': 200, 'NGEN': 150, 'CX_PB': 0.7, 'MUT_PB': 0.2

Pokrećem scenarij: Random Search (MC) (10 puta)

- Run 1/10: ROI=21.25, Trajanje=206.67
- Run 2/10: ROI=26.95, Trajanje=199.13
- Run 3/10: ROI=23.24, Trajanje=194.97
- Run 4/10: ROI=23.31, Trajanje=206.15
- Run 5/10: ROI=23.56, Trajanje=173.33
- Run 6/10: ROI=25.79, Trajanje=246.31
- Run 7/10: ROI=22.39, Trajanje=170.49
- Run 8/10: ROI=25.89, Trajanje=202.44
- Run 9/10: ROI=22.54, Trajanje=191.32
- Run 10/10: ROI=26.28, Trajanje=185.40

Pokrećem scenarij: GA (samo ROI) (10 puta)

- Run 1/10: ROI=38.27, Trajanje=252.42
- Run 2/10: ROI=37.59, Trajanje=248.50
- Run 3/10: ROI=38.19, Trajanje=270.84
- Run 4/10: ROI=38.11, Trajanje=256.57
- Run 5/10: ROI=37.13, Trajanje=254.03
- Run 6/10: ROI=38.42, Trajanje=267.28
- Run 7/10: ROI=36.93, Trajanje=234.86
- Run 8/10: ROI=38.88, Trajanje=266.76
- Run 9/10: ROI=37.97, Trajanje=231.50
- Run 10/10: ROI=38.27, Trajanje=252.86

Pokrećem scenarij: GA+MC (NSGA-II) (10 puta)

- Run 1/10: ROI=33.75, Trajanje=189.71
- Run 2/10: ROI=36.89, Trajanje=226.87

- Run 3/10: ROI=0.00, Trajanje=99999.00
- Run 4/10: ROI=35.65, Trajanje=207.29
- Run 5/10: ROI=0.00, Trajanje=99999.00
- Run 6/10: ROI=34.75, Trajanje=213.68
- Run 7/10: ROI=0.00, Trajanje=99999.00
- Run 8/10: ROI=0.00, Trajanje=99999.00
- Run 9/10: ROI=36.07, Trajanje=211.26
- Run 10/10: ROI=0.00, Trajanje=99999.00

Eksperiment: B3_Labav

Korištena konfiguracija: `'RUNS': 10, 'NUM_SIMULATIONS': 100, 'name': 'B3_Labav', 'NUM_ACTIVITIES': 50, 'BUDGET': 4000, 'POP_SIZE': 200, 'NGEN': 150, 'CX_PB': 0.7, 'MUT_PB': 0.2`

Pokrećem scenarij: Random Search (MC) (10 puta)

- Run 1/10: ROI=69.91, Trajanje=503.36
- Run 2/10: ROI=73.10, Trajanje=564.72
- Run 3/10: ROI=70.59, Trajanje=527.35
- Run 4/10: ROI=71.46, Trajanje=551.16
- Run 5/10: ROI=70.57, Trajanje=522.15
- Run 6/10: ROI=70.72, Trajanje=532.89
- Run 7/10: ROI=70.67, Trajanje=551.06
- Run 8/10: ROI=72.42, Trajanje=539.68
- Run 9/10: ROI=73.34, Trajanje=538.75
- Run 10/10: ROI=71.01, Trajanje=537.89

Pokrećem scenarij: GA (samo ROI) (10 puta)

- Run 1/10: ROI=79.07, Trajanje=570.73
- Run 2/10: ROI=78.66, Trajanje=552.45
- Run 3/10: ROI=79.50, Trajanje=564.56
- Run 4/10: ROI=78.73, Trajanje=547.34
- Run 5/10: ROI=79.50, Trajanje=563.87

- Run 6/10: ROI=79.50, Trajanje=568.08
- Run 7/10: ROI=79.48, Trajanje=568.89
- Run 8/10: ROI=77.79, Trajanje=546.55
- Run 9/10: ROI=79.22, Trajanje=565.31
- Run 10/10: ROI=79.20, Trajanje=574.13

Pokrećem scenarij: GA+MC (NSGA-II) (10 puta)

- Run 1/10: ROI=77.19, Trajanje=556.49
- Run 2/10: ROI=75.69, Trajanje=515.18
- Run 3/10: ROI=77.33, Trajanje=523.11
- Run 4/10: ROI=76.52, Trajanje=529.84
- Run 5/10: ROI=76.87, Trajanje=524.98
- Run 6/10: ROI=77.09, Trajanje=520.50
- Run 7/10: ROI=77.33, Trajanje=523.39
- Run 8/10: ROI=77.37, Trajanje=526.84
- Run 9/10: ROI=77.18, Trajanje=522.84
- Run 10/10: ROI=76.92, Trajanje=522.96

Testiranje konvergencije:

Korištena konfiguracija: `'RUNS': 1, 'NUM_SIMULATIONS': 100, 'name': 'A3_Slozeni', 'NUM_ACTIVITIES': 100, 'BUDGET': 5000, 'POP_SIZE': 250, 'NGEN': 200, 'CX_PB': 0.7, 'MUT_PB': 0.2`

Pokrećem scenarij: Random Search (MC) (1 puta)

- Run 1/1: ROI=95.95, Trajanje=726.93

Pokrećem scenarij: GA (samo ROI) (1 puta) *-¿ Bilježim statistike konvergencije... -¿ Podaci o konvergenciji spremljeni u 'konvergencija_A3.csv'*

- Run 1/1: ROI=112.88, Trajanje=806.57

Pokrećem scenarij: GA+MC (NSGA-II) (1 puta)

- Run 1/1: ROI=110.13, Trajanje=702.91