Optimiranje politike održavanja opreme korištenjem Markovljevih procesa odlučivanja

Štroliga, Luka

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 421

OPTIMIRANJE POLITIKE ODRŽAVANJA OPREME KORIŠTENJEM MARKOVLJEVIH PROCESA ODLUČIVANJA

Luka Štroliga

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 421

OPTIMIRANJE POLITIKE ODRŽAVANJA OPREME KORIŠTENJEM MARKOVLJEVIH PROCESA ODLUČIVANJA

Luka Štroliga

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

DIPLOMSKI ZADATAK br. 421

Pristupnik:	**Luka Štroliga (0023128409)**
Studij:	Računarstvo
Profil:	Programsko inženjerstvo i informacijski sustavi
Mentor:	izv. prof. dr. sc. Mirko Randić

Zadatak: **Optimiranje politike održavanja opreme korištenjem Markovljevih procesa odlučivanja**

Opis zadatka:

Markovljevi procesi odlučivanja (Markov decision processes) jesu modeli slijednog odlučivanja u stohastičkim uvjetima koji su se pokazali efikasni u optimiranju politika održavanja različitih tipova opreme. Stanja procesa predstavljaju operativna stanja opreme, a izazov je pronaći najbolji slijed akcija održavanja koji predstavlja optimalnu politiku održavanja u uvjetima kad imamo potpune ili nepotpune informacije o stanju opreme. Nepotpune informacije rezultat su djelomične vidljivosti ili indirektnih opažanja i pokušaja detektiranja stanja. Akcije održavanja utječu na promjenu stanja pri čemu se pojavljuju trenutni troškovi održavanja, dok su za ostvarivanje kontinuirane dobiti bitna prosječna vremena u kojima je oprema ispravna. Za modeliranje takvih odnosa mogu nam poslužiti polu-Markovljevi procesi odlučivanja i djelomično vidljivi Markovljevi procesi odlučivanja. U sklopu rada potrebno je opisati vrste Markovljevih procesa odlučivanja i standardne metode traženja optimuma. Na temelju poznatih razdioba vremena koja proteknu između ispada, a koja ovise o kvaliteti i tipu popravka te poznatih troškova, kao i dobiti tijekom ispravnog rada, potrebno je definirati model procesa održavanja te uz implementaciju odgovarajuće metode pronaći optimum u smislu poduzimanja odgovarajućih akcija što u konačnici rezultira maksimiziranjem dobiti.

Rok za predaju rada: 28. lipnja 2024.

1. Sadržaj

1. Sadržaj 2
2. Uvod 4
3. Markovljevi procesi odlučivanja 5
   3.1. Markovljevi lanci 5
   3.2. Markovljev proces nagrađivanja 6
   3.3. Markovljevi procesi odlučivanja (MDP) 8
   3.4. Polu-Markovljev procesi odlučivanja 9
4. Opis problema održavanja i modeli 11
   4.1. Jednostavniji model održavanja s jednim tipom akcije 12
      4.1.1. Vrijednosti nagrada u modelu 14
   4.2. Rješavanje problema 15
   4.3. Model održavanja s tri tipa akcija 20
      4.3.1. Vremena boravka u stanjima i vrijednosti nagrada u modelu 22
      4.3.2. Rješavanje problema 23
5. Zaključak 29
6. Literatura 30
7. Sažetak 31
8. Summary 32

2. Uvod

Održavanje opreme, pogotovo telekomunikacijske, jedan je od glavnih aspekata upravljanja sustavima. S krivom ili nepostojećom politikom održavanja opreme sustav postaje sklon kvarovima što za posljedicu može imati značajne financijske gubitke te smanjenje učinkovitosti sustava. U industriji kao što je telekomunikacijska, održavanje opreme je izuzetno važno jer pouzdanost telekomunikacijske opreme utječe na kvalitetu usluge, a time i zadovoljstvo korisnika.

Možemo razlikovati različite pristupe održavanju opreme. Reaktivno održavanje podrazumijeva popravak ili zamjenu opreme nakon što je došlo do kvara, te proaktivno održavanje ili preventivno održavanje s ciljem sprječavanja kvarova prije nego su se dogodili. Reaktivno održavanje je često učinkovita metoda, ali također može dovesti do neplaniranih prekida u radu i visokih troškova. S druge strane proaktivno održavanje može smanjiti učestalost kvarova, ali može dovesti i do skupih i nepotrebnih popravaka na opremi kojoj održavanje možda nije bilo potrebno.

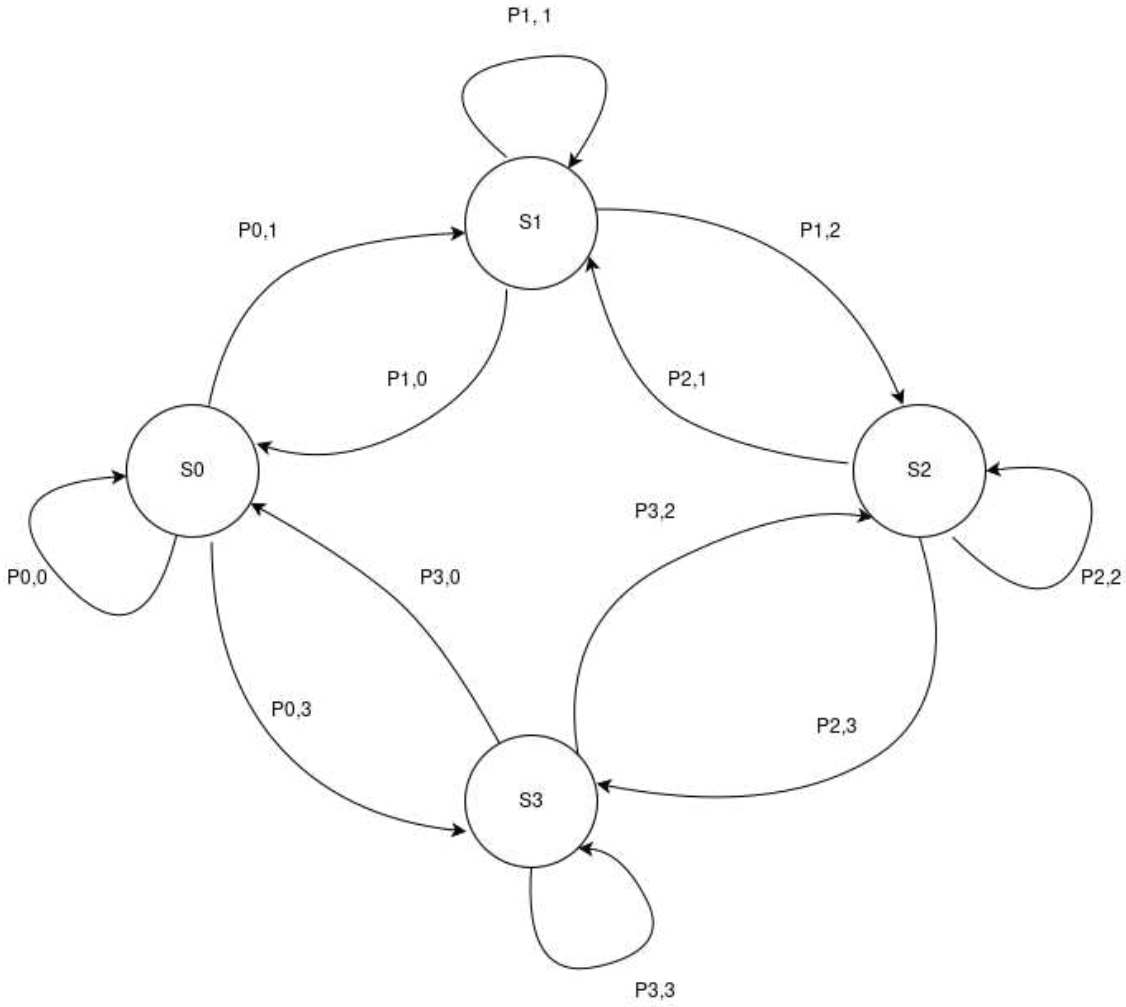
Cilj ovog rada je istražiti i pronaći optimalnu politiku održavanja telekomunikacijske opreme koristeći Markovljeve procese odlučivanja (Markov Decision Process, MDP) i polu-Markovljeve procese odlučivanja (Semi-Markov Decision Process, SMDP). Markovljevi procesi odlučivanja pružaju matematički okvir za modeliranje problema odlučivanja u kojima je ishod neizvjestan, ali ovisi o trenutnom stanju sustava i akcijama koje se poduzimaju. SMDP dodaje mogućnost modeliranja vremena u MDP modelu, što omogućuje preciznije modeliranje i određivanje optimalne politike održavanja opreme. [1]

U prvom poglavlju rada opisana je teorija Markovljevih lanaca i Markovljevih procesa odlučivanja, uključujući njihoviu primjenu u području održavanja opreme. Zatim će se definirati problem održavanja telekomunikacijskih oprema pomoću SMDP modela, s definiranim stanjima, akcijama, vrijednostima prijelaza te nagrada. Implementirat će se algoritmi za pronalaženje optimalne politike održavanja te će se rezultati analizirati kako bi se uvjerili u učinkovitost izračunatih politika održavanja. Očekuje se da će primjena SMDP modela omogućiti pronalazak optimalne politike održavanja s ciljem maksimiziranja raspoloživosti opreme, zadovoljstva korisnika i dobiti.

3. Markovljevi procesi odlučivanja

3.1. Markovljevi lanci

Markovljev lanac, nazvan po ruskom matematičaru Andrey Markov-u, matematički je model koji se koristi za predstavljanje niza stanja sustava. Obuhvaća niz stanja s definiranim prijelazima koji predstavljaju vjerojatnost prelaska iz jednog stanja u drugo. U svakom trenutku sustav može prijeći u neko novo stanje ili ostati u istom stanju. Promjene stanja nazivaju se tranzicije. Važno svojstvo Markovljevog lanca je da vjerojatnost prijelaza u sljedeće stanje ovisi samo o trenutnom stanju, a nikako o prethodnim stanjima.



Slika 1: Markovljev lanac s četiri stanja

Kao što se vidi na Slika 1, Markovljev lanac može se definirati kao uređeni par (S,P):

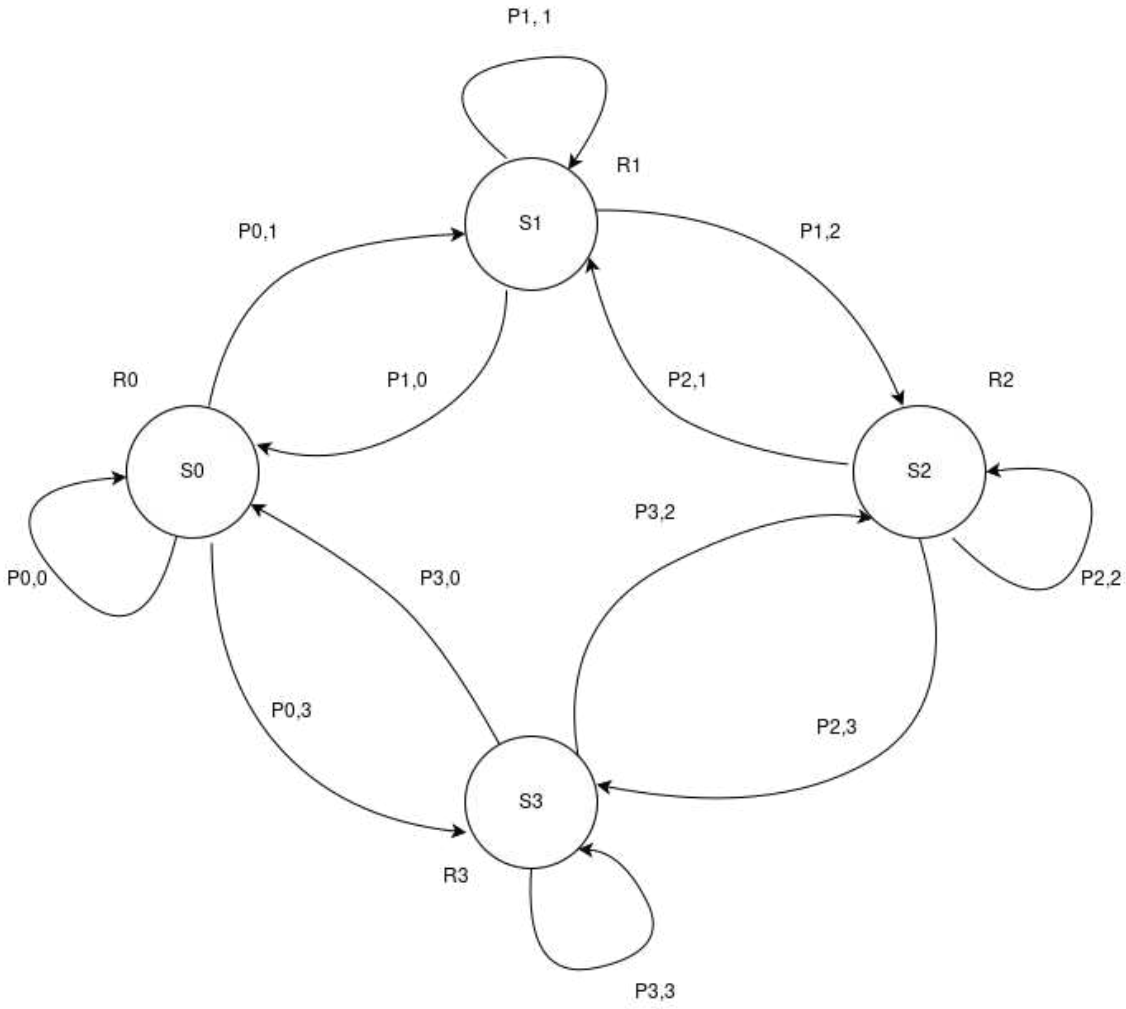
- S - konačan ili prebrojiv skup stanja u kojima se sustav može nalaziti

- P – matrica prijelaznih vjerojatnosti iz jednog stanja u drugo.

Markovljevi lanci često se koriste za modeliranje sustava s diskretnim vremenom napredovanja i stanjima, gdje je prijelaz između stanja opisan vjerojatnostima. Ovi modeli mogu se koristiti u mnogim područjima, uključujući održavanje opreme, gdje se koriste za predviđanje kvarova i optimizaciju politika održavanja.

3.2. Markovljev proces nagrađivanja

Markovljev proces nagrađivanja (Markov Reward Process, MRP) proširuje model Markovljevog lanca dodavanjem pojma nagrade (ili troška) povezanog s prijelazima između stanja. MRP omogućuje modeliranje ne samo prijelaza između stanja već i nagrade (pozitivne i negativne) koje ovise o prijelazima između stanja.



Slika 2: Markovljev proces nagrađivanja

Na Slika 2 vidimo da se MRP može definirati kao uređena trojka(S,P,R):

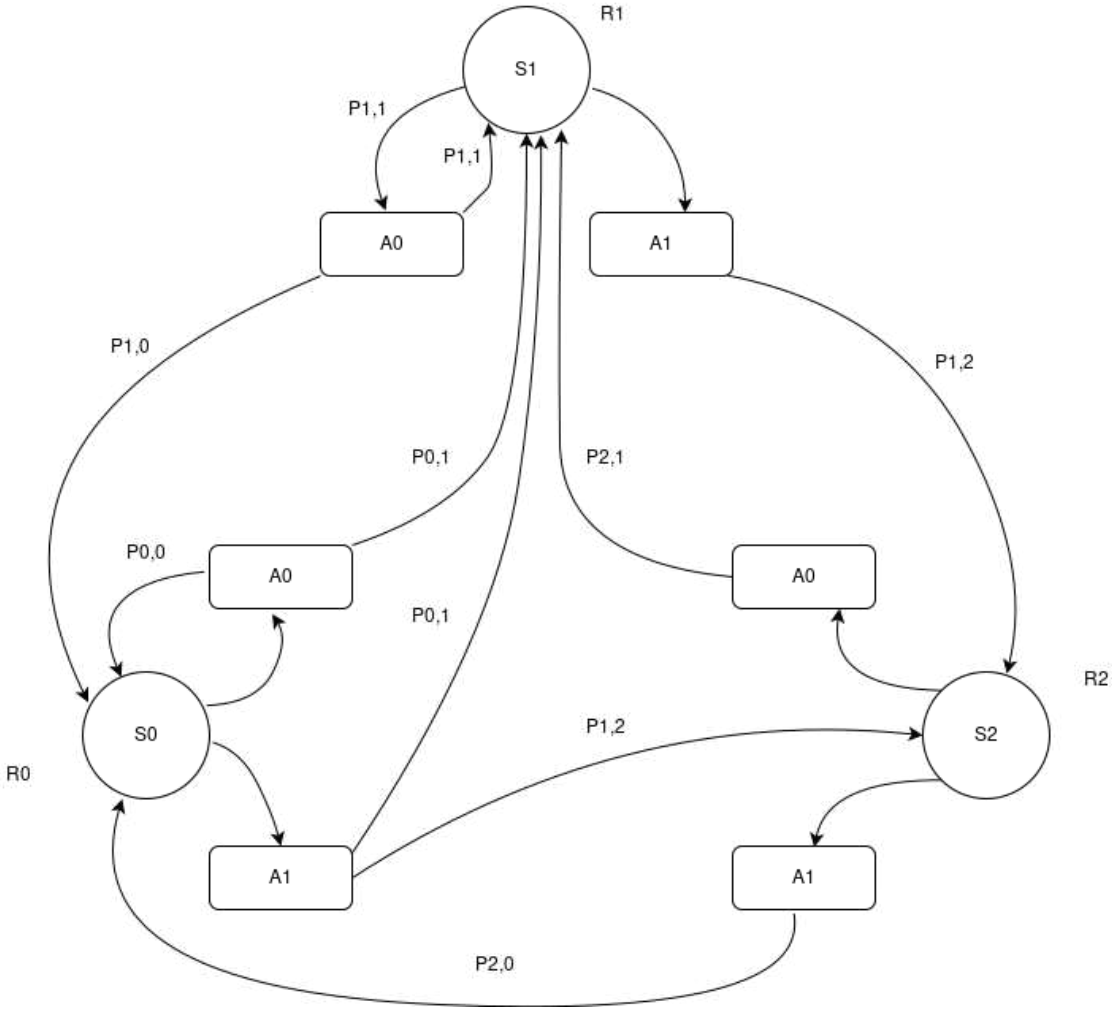
- S - konačan ili prebrojiv skup stanja u kojima se sustav može nalaziti
- P - matrica prijelaznih vjerojatnosti iz jednog stanja u drugo $P(S_i, S_j)$.
- R - funkcija nagrade $R(S_i, S_j)$ koja definira nagradu ili trošak prijelaza iz stanja S_i u stanje S_j.

Kod MRP modela, cilj je izračunati očekivanu kumulativnu nagradu kroz vrijeme, što omogućuje pronalaženje optimalnih odluka koje maksimiziraju ukupnu dobit ili minimiziraju ukupne troškove.

Formulacija MRP-a koristi se u mnogim područjima, uključujući upravljanje inventarom, planiranje proizvodnje, te održavanje opreme, gdje se koristi za optimizaciju strategija održavanja temeljenih na dugoročnim troškovima i dobitima.

3.3. Markovljevi procesi odlučivanja (MDP)

Markovljevi procesi odlučivanja predstavljaju matematički okvir za modeliranje donošenja odluka (poduzimanja akcija) u situacijama gdje su ishodi djelomično pod kontrolom donošenjem odluka, a djelomično slučajni. MDP modeli se koriste za optimizaciju problema u kojima odluke imaju dugoročne posljedice. U MDP-u u svakom koraku promatra se trenutno stanje te se odabire optimalna akcija na temelju zadane politike prijelaza. Odabrana akcija direktno utječe na nagradu (reward) prelaskom u određeno stanje. Cilj je pronalazak optimalne politike odabira akcija koja maksimizira dobit tijekom vremena. [3]



Slika 3: Markovljevi procesi odlučivanja

MDP se definira kao petorka (S,A,P,R,γ):

- S - konačan ili prebrojiv skup stanja u kojima se sustav može nalaziti
- A - konačan ili prebrojiv skup akcija.

- P(Sj|Si, A) - vjerojatnost prijelaza iz stanja Si u stanje Sj s akcijom A.
- R(Sj|Si, A) - nagrada dobivena prijelazom iz stanja Si u stanje Sj s akcijom A.
- γ je faktor umanjenja koji određuje važnost budućih nagrada.

MDP modeli omogućuju optimizaciju politike održavanja opreme uzimajući u obzir vjerojatnost kvarova i dugoročne troškove.

Preciznije, MDP model omogućava optimizaciju strategije održavanja kroz analizu vrijednosti prijelaza iz stanja u stanje te nagrada povezanih s različitim akcijama održavanja. Ovi modeli uzimaju u obzir trenutne troškove održavanja, vjerojatnost kvarova i dugoročne nagrade povezane s održavanjem.

Jedan od ključnih ciljeva primjene MDP-a u održavanju je minimizacija ukupnih troškova održavanja uz maksimiziranje raspoloživosti opreme te dobiti. To se postiže pronalaženjem optimalne politike održavanja koja određuje najbolju akciju održavanja za svako stanje opreme.

3.4. Polu-Markovljev procesi odlučivanja

Polu-Markovljevi procesi odlučivanja predstavljaju proširenje MDP-a koje uključuje vremenske aspekte prijelaza između stanja. U MDP-u prijelazi se javljaju u diskretnim vremenskim koracima, dok se u SMDP-u prijelazi mogu dogoditi u nepravilnim ili kontinuiranim vremenskim intervalima. Vrijeme koje sustav provodi u svakom stanju može biti promjenjivo i slijediti bilo koju raspodjelu. To omogućuje realističnije modeliranje sustava gdje vrijeme prijelaza ima značajan utjecaj na same performanse i troškove. [2]

SMDP se definira kao šestorka (S,A,P,R,γ,T):

- S - konačan ili prebrojiv skup stanja u kojima se sustav može nalaziti
- A - konačan ili prebrojiv skup akcija.
- $P(S_j|S_i, A)$ - vjerojatnost prijelaza iz stanja Si u stanje Sj s akcijom A.
- $R(S_j|S_i, A)$ - nagrada dobivena prijelazom iz stanja Si u stanje Sj s akcijom A.
- γ je faktor umanjenja koji određuje važnost budućih nagrada
- $T(S_j|S_i, A)$ - raspodjela vremena prijelaza iz stanja Si u stanje Sj pod akcijom A.

Korištenjem SMDP-a, moguće je preciznije modelirati dinamiku održavanja opreme, uzimajući u obzir varijabilnost vremena između kvarova i popravaka opreme.

Na kraju ovog poglavlja možemo zaključiti da Markovljevi procesi odlučivanja pružaju snažan matematički alat za modeliranje i optimizaciju politika održavanja. Proširenje Markovljevih procesa odlučivanja na polu-Markovljeve procese odlučivanja omogućuje realističnije modeliranje vremenskih aspekata održavanja. U sljedećim poglavljima, detaljno će se razraditi primjena ovih modela u kontekstu dva specifična slučaja održavanja telekomunikacijske opreme.

4. Opis problema održavanja i modeli

Kako je već u uvodu ukratko navedeno, strategije održavanja mogu se podijeliti u četiri glavne kategorije: reaktivno, preventivno, prediktivno i proaktivno održavanje.

Reaktivno održavanje, podrazumijeva popravak ili zamjenu potrebnih dijelova tek nakon što se kvar dogodio. Iako može biti učinkovit u većini slučaja, reaktivno održavanje može dovesti do neplaniranih i dugih prekida rada i visokih troškova popravaka.

Preventivno održavanje uključuje redovite inspekcije i zamjenu dijelova prema unaprijed definiranom rasporedu, bez obzira na stvarno stanje opreme. Ova strategija smanjuje vjerojatnost kvarova, ali može rezultirati nepotrebnim troškovima zbog zamjene dijelova koji još uvijek funkcioniraju ispravno.

Prediktivno održavanje koristi tehnologiju za praćenje stanja opreme i predviđanje kvarova prije nego što se dogode. Tehnike poput analize vibracija, materijala i analize ulja omogućuju identificiranje potencijalnih problema i poduzimanje mjera prije nego se kvar dogodi.

Proaktivno održavanje kombinira preventivno i prediktivno održavanje s ciljem identificiranja i uklanjanja uzroka problema prije nego se manifestira kao kvar.

Održavanje opreme igra ključnu ulogu u osiguravanju nesmetanog rada i dugovječnosti sustava. U kontekstu telekomunikacijskih linija, pravilno održavanje je od izuzetne važnosti za osiguranje kontinuiteta usluge i zadovoljstva korisnika. Dobro optimizirana politika održavanja maksimizira vrijeme dostupnosti sustava što direktno utječe na zadovoljstvo korisnika koji će vjerojatno nastaviti koristiti i plaćati uslugu. Loše optimizirana politika održavanja će s druge strane dovesti do čestih prekida usluge, a samim time i nezadovoljnijim korisnicima koji će zbog toga možda prestati koristiti uslugu ili će potražiti uslugu kod drugog operatera. U modelima koje opisujemo u nastavku pretpostavlja se mogućnost reaktivnog i proaktivnog održavanja. Na temelju definiranih modela odredit će se optimalne politike (optimalan slijed akcija) održavanja kako bi se minimizirali ukupni troškovi održavanja i maksimizirala raspoloživost opreme.

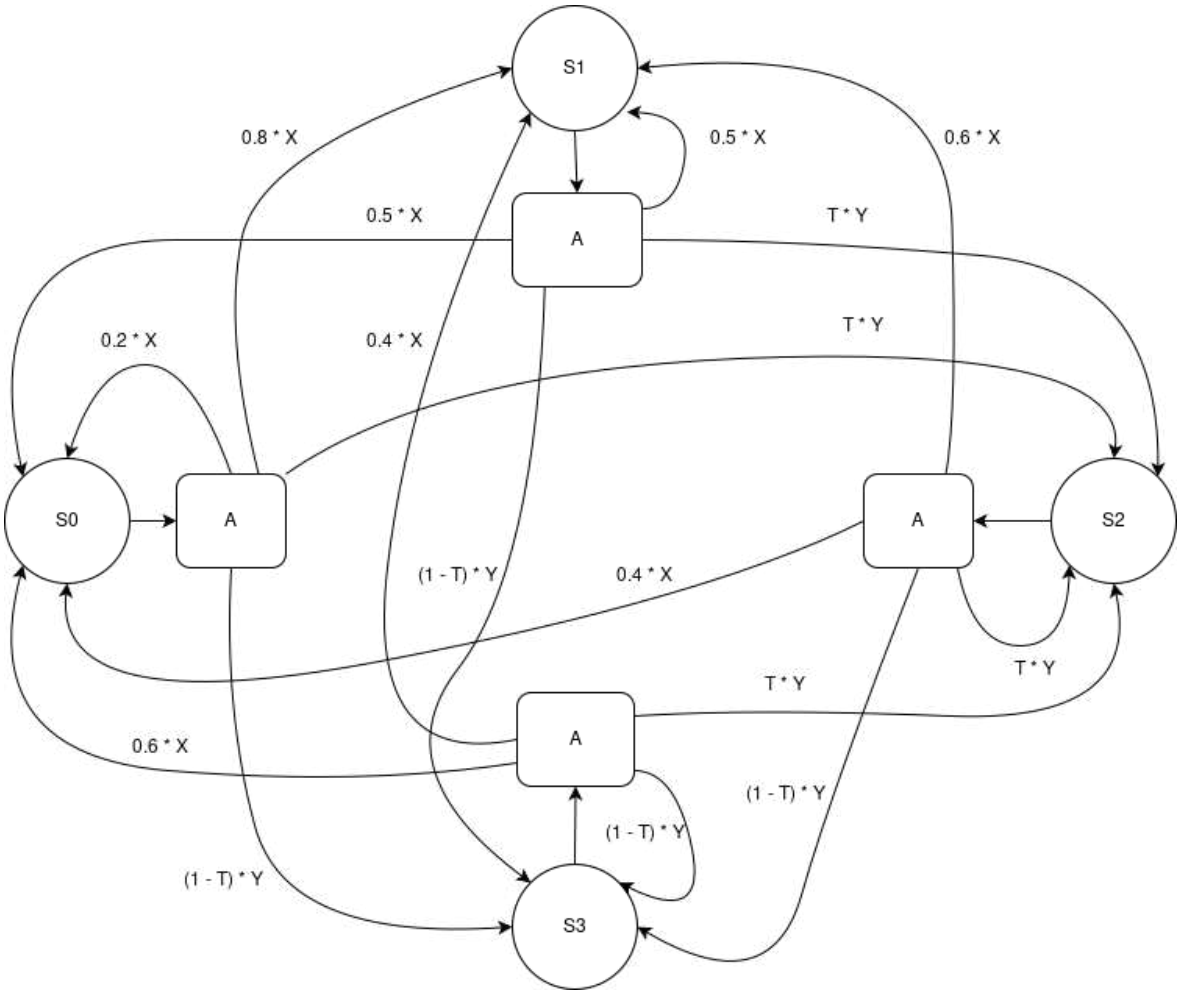
4.1. Jednostavniji model održavanja s jednim tipom akcije

Prvi model koji analiziramo uključuje četiri stanja i jednu akciju „Održavanje“, a koji predstavlja model održavanja telekomunikacijske linije.

Stanja (S):

- S0 – stanje linije u kojem postoji uočljiva degradacija, a linija je prethodno bila popravljena reaktivnom akcijom održavanja tipa „zamjena za novo“
- S1 – stanje linije u kojem postoji uočljiva degradacija, ali je linija prethodno bila popravljena reaktivnom akcijom održavanja tipa „krpanje“
- S2 - stanje linije u kojem postoji uočljiva degradacija, a linija je prethodno bila opravdano popravljena proaktivnom akcijom održavanja kojoj su prethodila točna ispitivanja na liniji te detekcija potencijalne pojave kvara u budućnosti
- S3 - stanje linije u kojem postoji uočljiva degradacija, a linija je prethodno bila neopravdano popravljena proaktivnom akcijom održavanja kojoj su prethodila pogrešna ispitivanja na liniji te detekcija potencijalne pojave kvara u budućnosti

U ovom modelu pomoću SMDP algoritma tražimo idealni omjer reaktivnog i proaktivnog održavanja za zadanu matricu prijelaza, uz točnost detektora koja nam ukazuje na to dali je proaktivno održavanje opreme stvarno potrebno.



Slika 4: Dijagram prijelaza

Dijagram prijelaza modela (Slika 4) prikazuje prijelaze između stanja s vjerojatnostima reaktivnog i proaktivnog održavanja. X označava vjerojatnost reaktivnog održavanja, Y označava omjer proaktivnog održavanja, a T označava točnost detektora za detektiranje budućih pogrešaka u sustavu. Omjer „zamjene za novo“ i „krpanje“ uzeli smo unikatno za svako stanje. Gamma (discount factor) smo uzeli kao 0.85.

4.1.1. Vrijednosti nagrada u modelu

```
up_times = np.array([
    [[365, 244, 237, 163]],
    [[365, 93, 239, 219]],
    [[365, 144, 184, 166]],
    [[365, 250, 246, 219]]
])

gain_per_day = np.array([
    [[5, 5, 5, 5]],
    [[5, 5, 5, 5]],
    [[5, 5, 5, 5]],
    [[5, 5, 5, 5]]
])

immediate_gain = np.array([
    [[-1000, -218, -200, -179]],
    [[-1000, -243, -290, -224]],
    [[-1000, -220, -186, -281]],
    [[-1000, -301, -390, -439]]
])
```

Slika 5: Vrijednosti nagrada u modelu

Vrijednosti nagrada (Slika 5) za prelazak iz stanja u stanje definirane su pomoću sljedećih parametara:

- up_times – vrijeme u danima provedeno u određenom stanju
- gain_per_day – dobit u eurima po danu provedenom u određenom stanju
- immediate_gain – trenutni ili jednokratni trošak prelaska iz stanja u stanje (troškovi održavanja)

Vrijednosti nagrada birane su tako da se što realnije opiše sustav. Dobit po danu je stavljena u svim stanjima kao 5$ na dan. Vrijeme rada sustava i trošak održavanja smo odredili po stanjima i akcijama. Zamjena za novo je uvijek 1000$ te tada je i najduže vrijeme rada sustava od 365 dana. Krpanje košta od 218$ do 301$ te mu je vrijeme rada sustava manje nego kod zamjene za novo te traje od 93 do 250 dana. Proaktivno održavanje košta između 179$ i 439$ te nakon proaktivnog održavanja očekivano vrijeme rada sustava je između 163 i 246 dana.

4.2. Rješavanje problema

Za rješavanje modela implementirat ćemo matematički model SMDP-a u programskom jeziku *Python.* Kroz sljedeće tri funkcije implementira se iterativni proces procjene i poboljšanja politike u SMDP modelu:

- funkcija '*calculate_action_values_model*' izračunava vrijednosti akcija za određeno stanje.
- funkcija '*calculate_value_function_model*' koristi se za izračun funkcije vrijednosti
- funkcija '*policy_improvement*' iterativno poboljšava politiku dok ne dođe do konvergencije.

```
1
2  def calculate_action_values_model(state, old_value_function):
3      num_actions = 1
4      action_values = np.zeros(num_actions)
5
6      for action in range(num_actions):
7          action_values[action] = np.sum(
8              transition_matrices[state, action] * (
9                  gain_per_day[state, action] * up_times[state, action] +
10                 immediate_gain[state, action] +
11                 gamma * old_value_function
12             )
13         )
14
15     return action_values
```

Slika 6: Funkcija calculate_action_values_model

Na Slika 6 je prikazana funkcija '*calculate_action_values_model'*. S obzirom na to da imamo samo jednu akciju '*num_actions'* je stavljen kao 1, te vrijednosti akcija na početku stavljamo kao sve 0. Za izračunavanje vrijednosti akcija, za svaku akciju radimo sumu umnožaka matrice prijelaza ('*transition_matrices'*), dobiti po danu ('*gain_per_day'*), vremena rada ('*up_times'*), trenutne dobiti, tj. troška ('*immediate_gain'*), te vrijednost stare funkcije pomnožene s faktorom gamma (diskontirana vrijednost stare funkcije). Matematički, ovo se može izraziti kao:

$$Q(S,A) = \sum_{S'} P(S'|S,A)(R(S,A) + \gamma * V(S'))$$

Gdje:

- Q(S,A) je vrijednost akcije za stanje S i akciju A
- P(S′|S,A) je prijelazna vjerojatnost iz stanja S u stanje S′ za akciju A
- R(S,A) je neposredna nagrada za izvršenje akcije A u stanju S
- γ je faktor umanjenja
- V(S′) je funkcija vrijednosti za sljedeće stanje s′

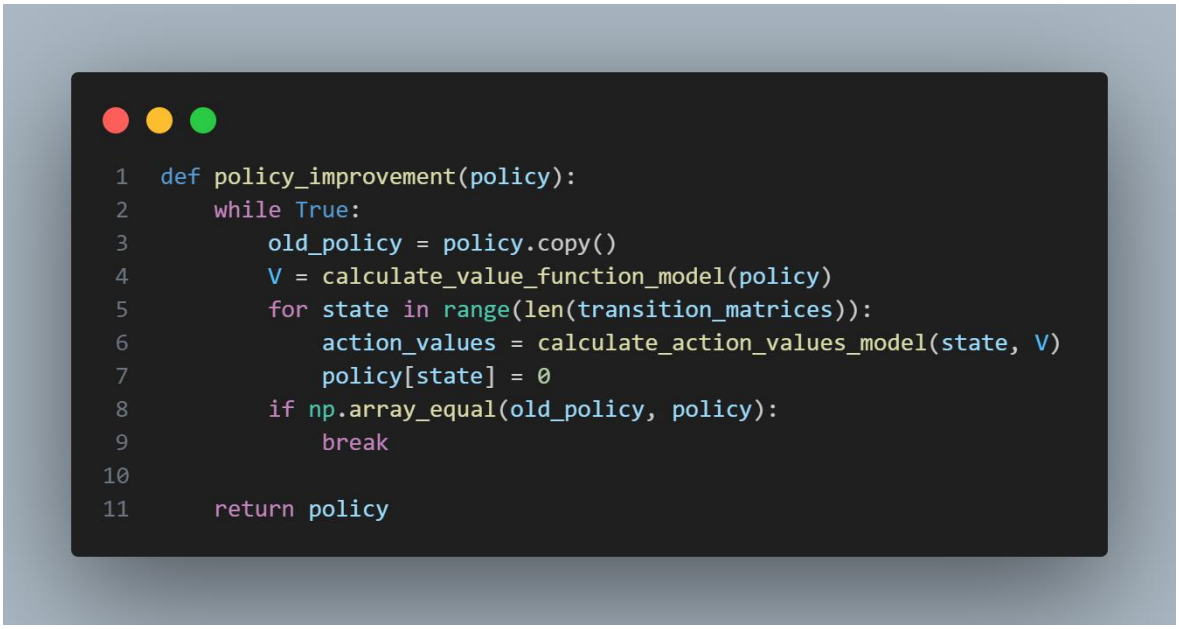
Funkcija vraća izračunate vrijednosti za svaku akciju.

```
def calculate_value_function_model(policy):
    num_states = len(transition_matrices)
    value_function = np.zeros(num_states)

    for iteration in range(max_iterations):
        old_value_function = np.copy(value_function)
        for state in range(num_states):
            action_values = calculate_action_values_model(state, old_value_function)
            value_function[state] = action_values[0]

        if np.max(np.abs(old_value_function - value_function)) < 1e-5:
            break

    return value_function
```

Slika 7: Funkcija calculate_value_function_model

Funkcija *'caluclate_value_function'* (Slika 7) iterativno izračunava vrijednost funkcije za svako stanje kroz određeni broj iteracija ili dok ne dođe do konvergencije. Kriterij

konvergencije je zadovoljen kada je maksimalna apsolutna razlika između vrijednosti stare i nove vrijednosti funkcije manja od 1 * e^{-5}.



```python
def policy_improvement(policy):
    while True:
        old_policy = policy.copy()
        V = calculate_value_function_model(policy)
        for state in range(len(transition_matrices)):
            action_values = calculate_action_values_model(state, V)
            policy[state] = 0
        if np.array_equal(old_policy, policy):
            break

    return policy
```

Slika 8: Funkcija policy_improvement

Funkcija '*policy_improvement*' (Slika 8) u SMDP modelu iterativno poboljšava politiku koristeći funkciju vrijednosti dok vrijednost stare i nove funkcije ne postanu jednake. S obzirom na to da u ovom slučaju postoji samo jedna akcija, funkcija uvijek vraća optimalnu politiku kao [0, 0, 0].

```python
results = []
max_value = float('-inf')
optimal_b = None

for b in range(0, 101):
    b = b / 100
    T = 0.85
    a = 1 - b

    px2 = T * b
    px3 = b - px2

    transition_matrices = np.array([
        [[a * 0.2 ,a * 0.8, px2, px3]],
        [[a * 0.5 ,a * 0.5, px2, px3]],
        [[a * 0.4, a * 0.6, px2, px3]],
        [[a * 0.6, a * 0.4, px2, px3]]
    ])

    initial_policy = np.array([0, 0, 0, 0])
    max_iterations = 50
    gamma = 0.85

    policy = policy_improvement(initial_policy)
    V = calculate_value_function_model(policy)

    total_value = np.sum(V)
    results.append({'b': b, 'total_value': total_value})
    print("b: ", b, ", total Value:", total_value)
    if total_value > max_value:
        optimal_policy = policy
        max_value = total_value
        optimal_b = b

df = pd.DataFrame(results)
df.to_csv('./final_all/action_values.csv', index=False)

print("Optimal value of b:", optimal_b)
```

Slika 9: Glavni program za rješavanje jednostavnijeg modela

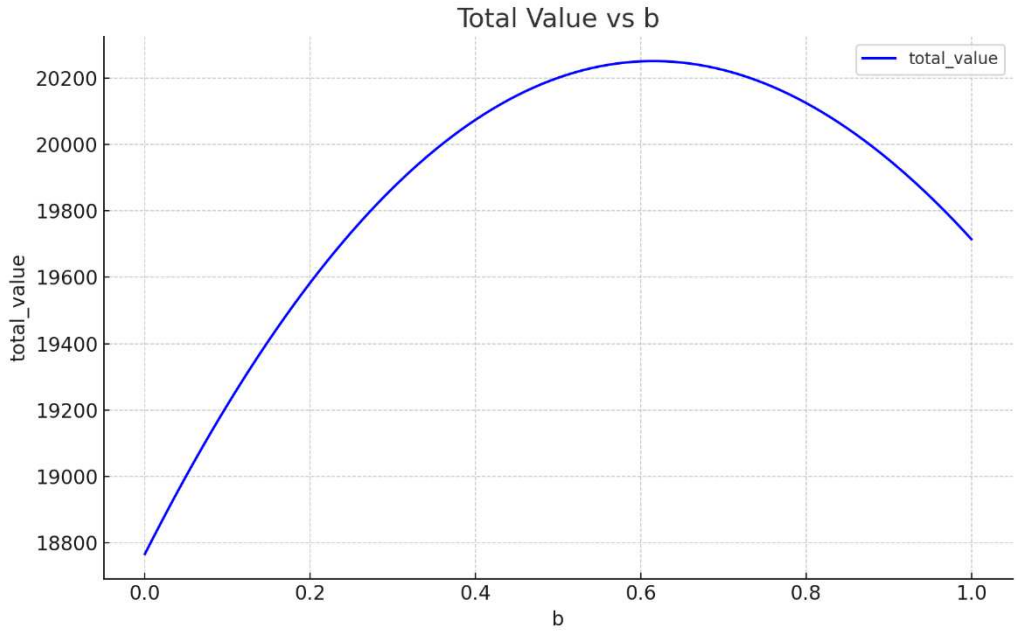
Na Slika 9 prikazan je glavni program rješenja modela 1. U njemu je uzeta realna vrijednost točnosti detektora T kao 0.85, te se iterira po vrijednosti b koja označava omjer vjerojatnosti proaktivnog i reaktivnog održavanja kako bi pronašli optimalni b koji daje maksimum funkcije. Rezultate spremamo u *csv* datoteke kako bi ih mogli dalje koristiti u analizi. Program ispisuje rezultat u sljedećem obliku:

Optimal value of b: 0.62

Max value function with b(0.62) is: 20251.08246546889

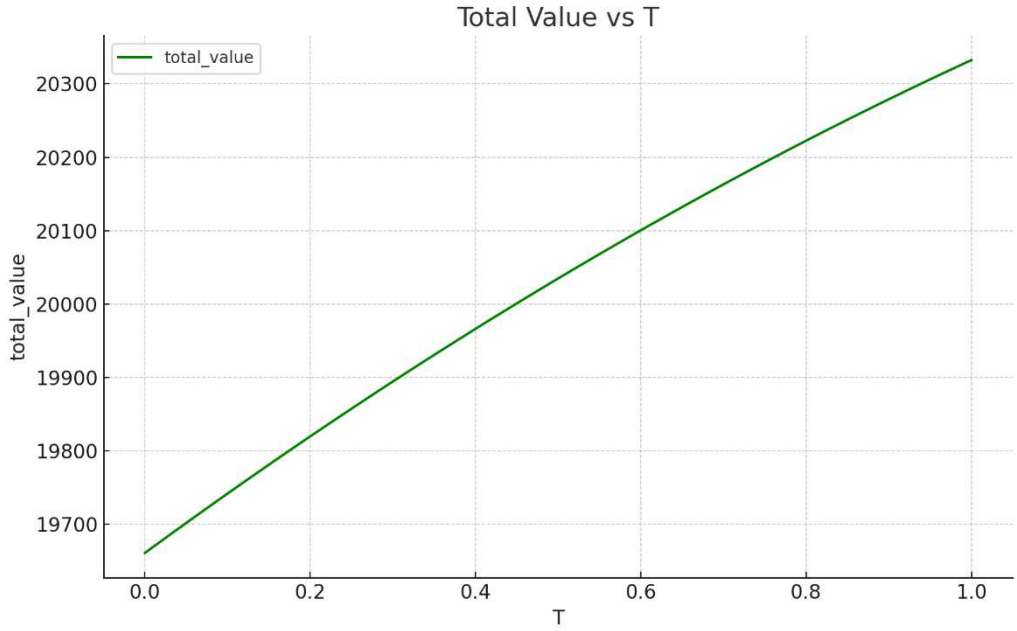
Što nam govori da je maksimalna vrijednost funkcije 20251 za optimalnu vrijednost b 0.62.

S obzirom na to da smo spremali sve vrijednosti funkcije u csv datoteku, od nje će se napraviti graf kako bi bolje vizualizirali samo funkciju.



Slika 10: Dobivena vrijednost kao funkcija b

Iz prikazanog grafa se može primijetiti da kako raste vrijednost parametra b raste i vrijednost funkcije sve dok se ne dođe do vrijednosti parametra b = 0.62 gdje funkcija poprima svoj maksimum, nakon čega vrijednost funkcije počinje padati. Iz priloženog možemo zaključiti da je optimalni omjer proaktivnog i reaktivnog održavanja za zadani model 62%.



Slika 11: Vrijednost funkcije u ovisnosti o točnosti detektora

Na prikazanom grafu (Slika 11) vidi se promjena vrijednosti funkcije s optimalnim parametrom b = 0.62 ovisna o promjeni točnosti detektora. Može se primijetiti da kako točnost detektora rase, tako raste i vrijednost funkcije. Iz toga možemo zaključiti da podaci o nagradama i troškovima za ovaj model dobro odražavaju stvarne uvjete, jer pokazuju slične obrasce kao i u pravom svijetu.

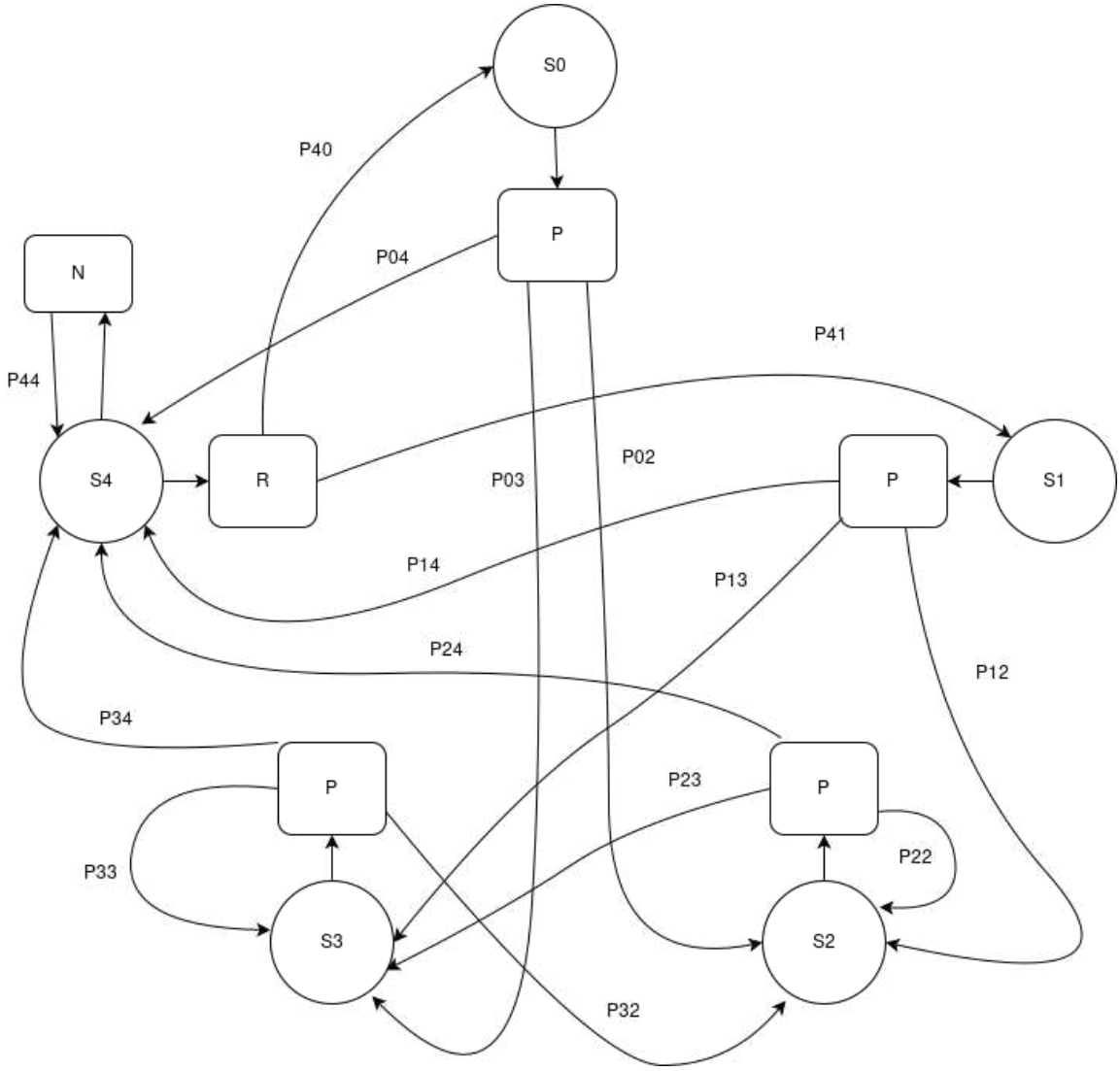
4.3. Model održavanja s tri tipa akcija

Ovaj model uključuje pet stanja te tri akcije „Ne radi ništa“, „Proaktivni popravak u kombinaciji s „ne radi ništa““ i „Reaktivni popravak“:

Stanja (S):

- S0 – linija ispravna, a prethodno je obnovljena „kao nova“
- S1 – linija ispravna, a prethodno je zakrpana
- S2 – linija ispravna, a prethodno je popravljena proaktivno, popravak je bio potreban, tj. detektor je točno detektirao kvar (true positive)
- S3 – linija ispravna, a prethodno je popravljena proaktivno, popravak je bio nepotreban, tj. detektor je krivo detektirao kvar (false positive)
- S4 – linija degradirana, korisnik primjećuje neispravnost, a korištenje je više ili manje nemoguće

Pomoću algoritma za rješavanje SMDP modela tražimo idealni omjer reaktivnog i proaktivnog održavanja za zadanu matricu prijelaza, uz varijabilnu točnost detektora koja nam ukazuje na to da li je proaktivno održavanje opreme stvarno potrebno.



Slika 12: Dijagram prijelaza modela 2

Dijagram prijelaza modela (Slika 12) prikazuje prijelaze između stanja s vrijednostima prijelaza gdje p_{xy} označava vrijednosti prijelaza iz stanja x u stanje y. p_{x2} i p_{x3} označava proaktivno održavanje, p_{44} označava radnju „ne radi ništa“, a p_{40} i p_{41} označava reaktivno održavanje. p_{x4} označava izabranu akciju proaktivno održavanje s kojom se ništa nije popravilo te se sustav doveo u stanje gdje je linija degradirana.

4.3.1. Vremena boravka u stanjima i vrijednosti nagrada u modelu

```python
up_times = np.array([
    # akcija P           , akcija N         , akcija R
    [[0, 0, 180, 231, 5], [0, 0, 0, 0, 0],    [0, 0, 0, 0, 0]],  # Novo 1
    [[0, 0, 250, 205, 5], [0, 0, 0, 0, 0],    [0, 0, 0, 0, 0]],  # Krpano 2
    [[0, 0, 100, 137, 5], [0, 0, 0, 0, 0],    [0, 0, 0, 0, 0]],  # ProaktivT 3
    [[0, 0, 196, 146, 5], [0, 0, 0, 0, 0],    [0, 0, 0, 0, 0]],  # ProaktivF 4
    [[0, 0, 0, 0, 0],    [0, 0, 0, 0, 1],    [346, 187, 0, 0, 0]]  # Degr 5
])


gain_per_day = np.array([
            #akcija P      akcija N       akcija R
    [[0, 0, 5, 5, -5], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]],  # Novo 1
    [[0, 0, 5, 5, -5], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]],  # Krpano 2
    [[0, 0, 5, 5, -5], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]],  # ProaktivT 3
    [[0, 0, 5, 5, -5], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]],   # ProaktivF 4
    [[0, 0, 0, 0, 0], [0, 0, 0, 0, 0], [5, 5, 0, 0, 0]]   # Degr 5
])

immediate_gain = np.array([
    # akcija P           , akcija N          , akcija R
    [[0, 0, -84, -262, 0],  [0, 0, 0, 0, 0],    [0, 0, 0, 0, 0]],  # Novo 1
    [[0, 0, -250, -106, 0],  [0, 0, 0, 0, 0],    [0, 0, 0, 0, 0]],  # Krpano 2
    [[0, 0, -307, -277, 0], [0, 0, 0, 0, 0],   [0, 0, 0, 0, 0]],  # ProaktivT 3
    [[0, 0, -70, -119, 0],   [0, 0, 0, 0, 0],    [0, 0, 0, 0, 0]],  # ProaktivF 4
    [[0, 0, 0, 0, 0],       [0, 0, 0, 0, -100],      [-952, -403, 0, 0, 0]]   # Degr 5
])
```

Slika 13: Vremena boravka u stanjima i vrijednosti nagrada

Vremena boravka u stanjima i vrijednosti nagrada za prelazak iz stanja u stanje definirani su sljedećim parametrima:

- *'up_times'* - vrijeme u danima provedeno u određenom stanju
- *'gain_per_day'* - profit u eurima po danu provedenom u određenom stanju
- *'immediate_gain'* - trenutni trošak prelaska iz stanja u stanje

Vrijednosti nagrada birane su tako da se što realnije opiše sustav. Dobit po danu je stavljena u svim stanjima kao 5$ po danu osim u stanju kada sustav ne radi tada imamo trošak od 5$ po danu. Vrijeme rada sustava i trošak održavanja smo odredili po stanjima i akcijama. Proaktivno održavanje će koštati između 70$ i 307$ te očekujemo da će sustav nakon toga raditi između 100 i 250 dana. Reaktivno održavanje je skuplje te košta 952$ za zamjenu novim, a 403$ za krpanje starog sustava. U slučaju zamjene novim očekivano vrijeme rada sustava je 346 dana, a u slučaju krpanja 187 dana.

4.3.2. Rješavanje problema

S obzirom na to da nam je cilj i kod ovog složenijeg modela dobiti optimalni omjer proaktivnog i reaktivnog održavanja koristeći iste funkcije kao i u prethodnom jednostavnijem modelu varirat ćemo b (vrijednost proaktivnog održavanja) dok ne dobijemo maksimum funkcije. Uz samo variranje vrijednosti proaktivnog održavanja, varirat ćemo i omjer zamjene za novo te popravka linije u slučaju reaktivnog održavanja.

```python
px1_values = [0.25, 0.5, 0.75]
results = []

#trazimo optimalni b
for b in range(0, 101):
    b = b / 100
    a = 1 - b

    T = 0.85 #tocnost detektora

    px5 = a
    px3 = b * T
    px4 = b - px3

    for px1 in px1_values:
        px2 = 1 - px1

        transition_matrices = np.array([
                #akcija P     ,     akcija N   ,    akcija R
            [[0, 0, px3, px4, px5], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]],  # Novo 1
            [[0, 0, px3, px4, px5], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]],  # Krpano 2
            [[0, 0, px3, px4, px5], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]],  # ProaktivT 3
            [[0, 0, px3, px4, px5], [0, 0, 0, 0, 0], [0, 0, 0, 0, 0]],   # ProaktivF 4
            [[0, 0, 0, 0, 0],       [0, 0, 0, 0, 1], [px1, px2, 0, 0, 0]]   # Degr 5
        ])

        initial_policy = np.array([1, 1, 1, 1, 1])
        max_iterations = 30
        gamma = 0.85

        policy = policy_improvement(initial_policy)
        V = calculate_value_function_model(policy)

        total_value = np.sum(V)
        print(f"b: {b:.2f}, px1: {px1:.2f} Total Value: {total_value:.2f}")
        results.append({"b": b, "px0": px1 ,  "total_value": total_value})


        if total_value > max_value:
            optimal_policy = policy
            max_value = total_value
            optimal_b = b

df = pd.DataFrame(results)
df.to_csv('./final_all/action_values2.csv', index=False)
```

Slika 14: Glavni program za rješavanje modela

Na Slika 14 vidimo da glavni program jako sliči onome za prethodni model, ali s malim razlikama. Prvo variramo sve b vrijednosti kao i u gornjem modelu, ali imamo još jednu for petlju koja nam prolazi po omjerima vrijednosti „zamjena za novo“ i „krpanje“ u akciji reaktivnog održavanja. Program nam na kraju ispisuje optimalnu politiku održavanja, optimalnu vrijednost b i maksimalnu vrijednost funkcije za taj b:

Optimal policy: [0 0 0 0 2]

Optimal value of b: 0.69

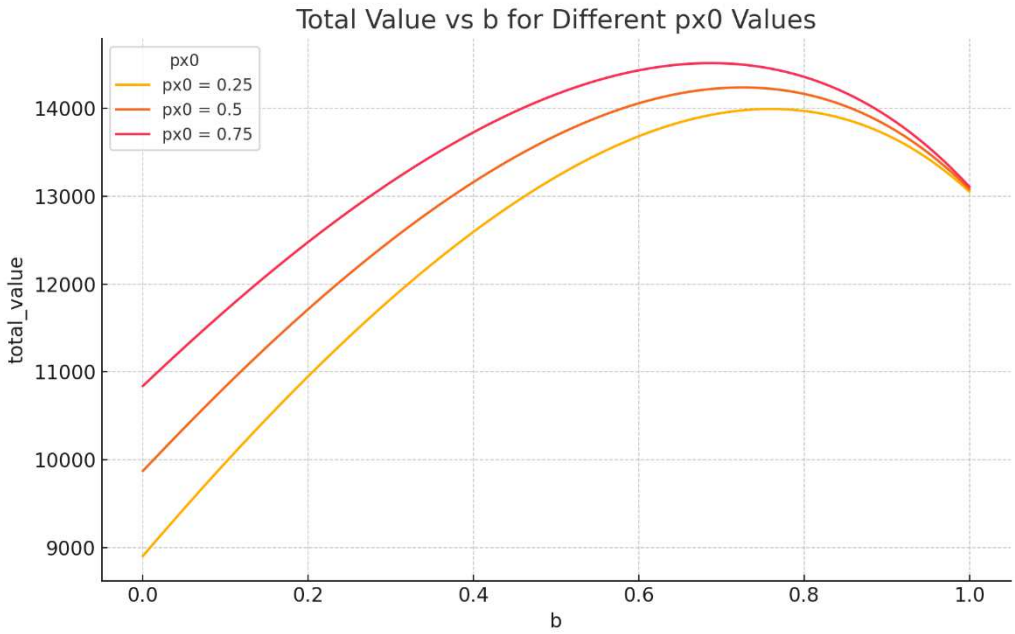
Optimal value of px1: 0.75

Max value function with b(0.69) is: 14514.065939757322

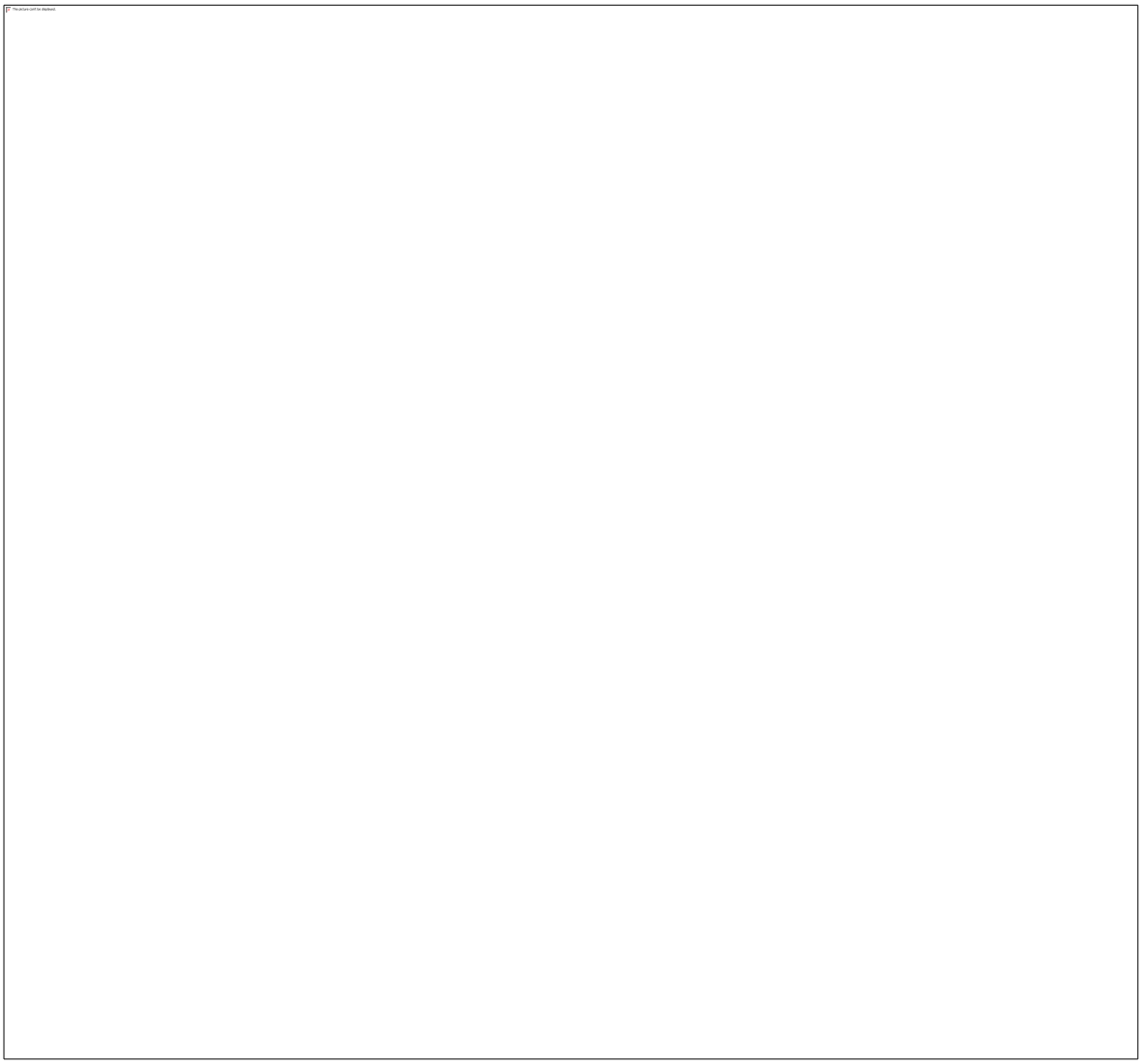
Ispis ovog programa nam govori da je optimalna politika održavanja [0 0 0 0 2], te da je optimalni omjer proaktivnog održavanja 69% sa maksimalnom vrijednosti funkcije od 14514. Osim parametra b tražili smo i optimalni omjer vrijednosti „zamjena za novo“ i „krpanje“ kod akcije proaktivnog održavanja koja iznosi 75%. Sve izračunate vrijednosti funkcije spremamo u csv datoteke za lakšu analizu samog rješenja.

Kao i u prethodnom modelu, spremljene su sve izračunate vrijednosti funkcije za vrijednosti b od 0.00 do 1.00. U ovom modelu osim parametra b spremali smo i omjer „zamjene za novo“ i „krpanje“ kod akcije reaktivnog održavanja, tako da će se i te vrijednosti vizualizirati.

Iz priloženih 2D grafova vidimo da za sva 3 omjera vrijednosti „zamjena za novo“ i „krpanje“ kod akcije reaktivnog održavanja funkcija poprima sličan oblik čime možemo zaključiti da za zadani model taj omjer nije od ključne važnosti iako je razlika maksimuma funkcije između svakih od grafova oko 250. Kao i kod prošlog modela možemo zaključiti kako porastom proaktivnog održavanja vrijednost funkcije raste dok ne dosegne maksimum (u ovom slučaju maksimum funkcije dobivamo kad je parametar b jednak 0.69) te nakon toga vrijednost funkcije ponovo pada.

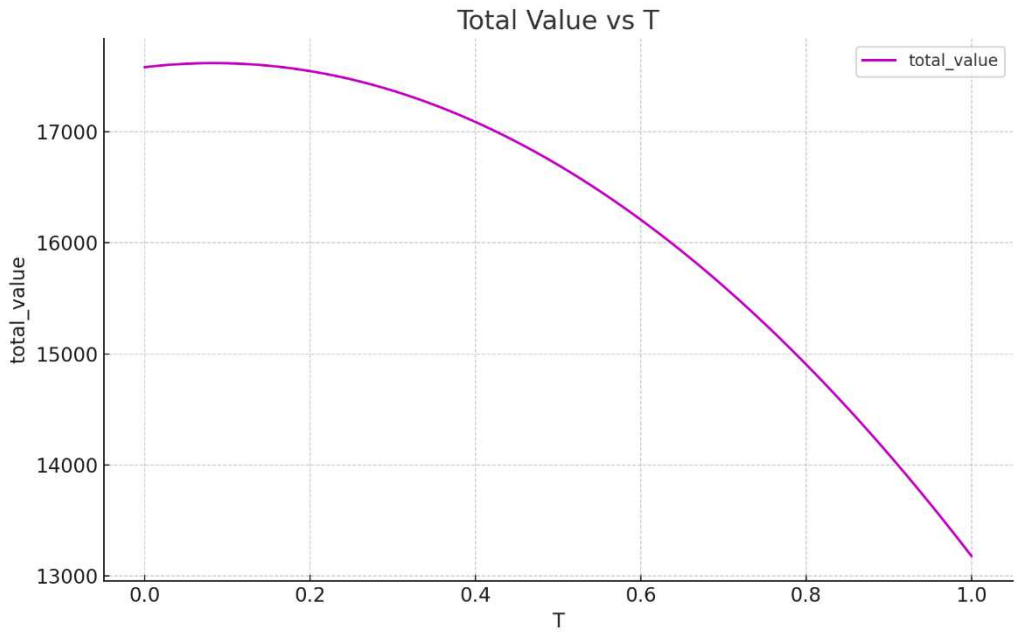


Slika 15: Funkcije vrijednosti za svaku vrijednost p_{x0}



Slika 16: 3D graf funkcije vrijednosti modela

Na ovom 3D grafu prikazani su isti podaci samo su bolje vizualizirani. Također se primijeti da graf funkcije poprima maksimum s vrijednostima [$p_{x1} = 0.75$, $b = 0.69$].



Slika 17: Vrijednost funkcije u ovisnosti o točnosti detektora

Na prikazanom grafu (Slika 17Slika 11) vidi se promjena vrijednosti funkcije s optimalnim parametrima (b = 0.69, p_{x0} = 0.75) ovisna o promjeni točnosti detektora (T). Može se primijetiti da funkcija poprima najveću vrijednost za T = 0.08, te da nakon toga vrijednost funkcije pada. Iz toga možemo zaključiti da podaci o nagradama i troškovima za ovaj model ne prikazuju stvarne uvjete na dobar način, jer ne odražavaju očekivane obrasce iz stvarnog svijeta.

5. Zaključak

Polu-Markovi procesi odlučivanja (SMDP) pokazali su se kao učinkoviti modelu za modeliranje i određivanje optimalne politike održavanja telekomunikacijske opreme. Kroz detaljnu analizu pronašli smo optimalne politike održavanja te optimalni omjer reaktivnog i proaktivnog održavanja za svaki model. Pronađene politike održavanja i omjer reaktivnog i proaktivnog održavanja znatno smanjuju troškove, tj. povećaju dobit samog modela, uz to što povećaju raspoloživost same opreme. Implementacija ovakvih modela u stvarnim uvjetima također mogu dovesti do značajnih financijskih poboljšanja samog sustava. Dodavanjem vremena kao varijable koju podržava SMDP model dobili smo precizniji opis rada sustava. Buduća istraživanja mogu se usmjeriti na poboljšanje modela dodavanjem više varijabli te razvijanjem naprednijeg algoritma koji bi uključio i više scenarija samog sustava. Dugoročno, analiza održavanja uz pomoć SMDP modela može značajno poboljšati učinkovitost i pouzdanost telekomunikacijskih sustava što direktno utječe na povećanje zadovoljstva korisnika i smanjenje operativnih troškova. Naravno, pri tom je potrebno realnije procijeniti kontinuiranu dobit koju operator ostvaruje u vremenu dok oprema radi, kao i troškove različitih vrsta održavanja opreme, ali i direktne i indirektne troškove zbog neispravnosti opreme.

6. Literatura

[1] Puterman, M. L. (1994) Markov Decision Processes: Discrete Stochastic Dynamic Programming

[2] Bertsekas, D. P. (1995) Dynamic Programming and Optimal Control

[3] Bellman, R. E. (1957) Dynamic Programming

7. Sažetak

Dobro izabrana politika održavanja telekomunikacijske opreme ključna je za smanjenje troškova i povećanje dostupnosti sustava, a uz to i direktno utječe na zadovoljstvo korisnika. U ovom radu istražuje se mogućnost pronalaska idealne politike održavanja telekomunikacijskih linija pomoću polu-Markovljevih procesa odlučivanja (SMDP). Markovljev proces odlučivanja omogućava modeliranje sustava uzimajući u obzir trenutna stanja sustava te akcije koje se mogu poduzeti u tim stanjima. SMDP model još dodaje i vremenski aspekt tako da preciznije opisuje proces odlučivanja. Specificirana su dva modela održavanja, a algoritam za rješavanje oba modela implementirali smo u programskom jeziku Python. Nakon provedene analize možemo reći da smo pronašli idealnu politiku održavanja za oba modela te se uz pomoć dobivenih rezultata može značajno smanjiti operativne troškove održavanja uz povećanje raspoloživosti sustava.

8. Summary

A well-chosen telecommunications equipment maintenance policy is essential for reducing costs and increasing system availability, and it also directly affects user satisfaction. This paper investigates the possibility of finding an ideal policy for the maintenance of telecommunication lines using semi-Markov decision processes (SMDP). The Markov decision process enables system modeling by taking into account the current state of the system and the actions that can be taken in those states. The SMDP model also adds a time aspect so that it more precisely describes the decision-making process. Two maintenance models were specified, and the algorithm for solving both models was implemented in the Python programming language. After the analysis, we can say that we have found an ideal maintenance policy for both models, and with the help of the obtained results, operational maintenance costs can be significantly reduced while increasing system availability.