Učinkovito raspoređivanje pacijenata na bolničke preglede

Šobačić, Lea

Undergraduate thesis / Završni rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:180167

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-24**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni prijediplomski studij Računarstvo

UČINKOVITO RASPOREĐIVANJE PACIJENATA NA BOLNIČKE PREGLEDE

Završni rad

Lea Šobačić

Osijek, 2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P: Obrazac za ocjenu završnog rada na sveučilišnom prijediplomskom studiju

Ocjena završnog rada na sveučilišnom prijediplomskom studiju

Ime i prezime pristupnika:	Lea Šobačić
Studij, smjer:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. pristupnika, god.	R4574, 28.07.2020.
JMBAG:	0165086796
Mentor:	prof. dr. sc. Josip Job
Sumentor:	
Sumentor iz tvrtke:	
Naslov završnog rada:	Učinkovito raspoređivanje pacijenata na bolničke preglede
Znanstvena grana završnog rada:	**Informacijski sustavi (zn. polje računarstvo)**
Zadatak završnog rada:	Potrebno je proučiti problematiku listi čekanja na bolničke pretrage. Potrebno je predložiti metodologiju naručivanja i raspoređivanja pacijenata te implementirati testno okruženje putem kojega će se ispitati mogućnosti skraćivanja vremena potrebnog za obavljanje bolničkog pregleda. Tema rezervirana za; Lea Šobačić

Datum prijedloga ocjene završnog rada od strane mentora:	18.09.2024.
Prijedlog ocjene završnog rada od strane mentora:	Izvrstan (5)
Datum potvrde ocjene završnog rada od strane Odbora:	25.09.2024.
Ocjena završnog rada nakon obrane:	Izvrstan (5)
Datum potvrde mentora o predaji konačne verzije završnog rada čime je pristupnik završio sveučilišni prijediplomski studij:	28.09.2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O IZVORNOSTI RADA

Osijek, 28.09.2024.

Ime i prezime Pristupnika:	Lea Šobačić
Studij:	Sveučilišni prijediplomski studij Računarstvo
Mat. br. Pristupnika, godina upisa:	R4574, 28.07.2020.
Turnitin podudaranje [%]:	8

Ovom izjavom izjavljujem da je rad pod nazivom**: Učinkovito raspoređivanje pacijenata na bolničke preglede**

izrađen pod vodstvom mentora prof. dr. sc. Josip Job

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis pristupnika:

SADRŽAJ

1. UVOD **1**
1.1. Zadatak završnog rada 2
2. PREGLED DOSADAŠNJIH ISTRAŽIVANJA **3**
2.1. Prioritizacija pacijenata na kirurškim listama čekanja 3
2.2. Raspoređivanje u zdravstvenom sustavu u kontekstu optimizacije 5
2.3. Distribucija vremena čekanja u javnom zdravstvu 7
2.4. Raspored prijema pacijenata koristeći genetski algoritam 9
2.5. Optimizacija voznog reda 13
3. RAZRADA TEME **15**
3.1. Tehnologije korištene u izradi rada 17
3.1.1. Visual Studio Code uređivač 17
3.1.2. Python 18
3.2. Generiranje sintetičkih podataka 19
3.2.1. Pacijenti 20
3.2.2. Resursi 22
3.3. Implementacija algoritama 23
3.3.1. FCFS 24
3.3.2. PS 27
3.3.3. ILP 29
4. RASPRAVA I ANALIZA REZULTATA **35**
4.1. Analiza početnih skupova podataka 35
4.2. Evaluacija algoritama i metoda 37
4.3. Rasprava 45
5. ZAKLJUČAK **49**
LITERATURA **50**
SAŽETAK **53**
ABSTRACT **54**
ŽIVOTOPIS **55**
PRILOZI **56**

1. UVOD

Zdravstveno stanje stanovništva ključan je pokazatelj gospodarskog razvoja zemlje i bitan preduvjet za daljnji rast društva. Kvaliteta zdravstvene zaštite izravno utječe na kvalitetu života, ekonomski napredak zemlje te podizanje životnog standarda zajednice, a isto se može unaprijediti osiguravanjem odgovarajuće zdravstvene skrbi i jednakih mogućnosti svim građanima.

Ovaj rad bavi se sustavom zdravstva u Republici Hrvatskoj, s posebnim naglaskom na smanjenju vremena čekanja za bolničke preglede, odnosno optimalnom i brzom pružanja usluga pacijentima kako bi se rasteretio sustav. Učinkovito raspoređivanje pacijenata na bolničke preglede predstavlja veliki izazov u suvremenom zdravstvenom sustavu pošto je pod utjecajem mnoštva faktora kao što su: ograničeni resursi, rast broja pacijenata, nedostatak medicinskog osoblja... Unaprjeđenjem efikasnosti raspoređivanja možemo značajno povećati zadovoljstvo pacijenata, poboljšati radnu učinkovitost medicinskog osoblja i smanjiti ukupne troškove zdravstvenih usluga što bi kao krajnji produkt dovelo do poboljšanja kvalitete života.

Problem dugog čekanja za bolničke preglede produkt je ne samo velike potražnje, već i nejednakost u kontekstu financijskih uvjeta. Hrvatska se duži niz godina suočava s iseljavanjem mladih i visokoobrazovanih ljudi kao i sa starenjem populacije što dodatno opterećuje zdravstveni sustav. Također, mnogi se, uoči neefikasnosti javne zdravstvene skrbi, podvrgnu privatnim ustanovama koje, nažalost, nisu u skladu sa svačijim mogućnostima [1].

Raspoređivanje pacijenata uključuje niz odluka koje uzimaju u obzir čimbenike poput hitnosti slučaja, dostupnosti liječnika, raspoloživosti medicinske opreme te specifičnih potreba pacijenata te spremnost da se prilagode situaciji, u smislu fleksibilnosti kako bi čim prije njihova potreba bila zadovoljena. U praksi se koriste različite metode optimizacije i algoritmi za optimizaciju ovog problema kao i njemu srodnih, a svaki od njih ima svoje prednosti i nedostatke.

Cilj ovog rada jest podići svijest o zagušenosti listi čekanja kako u Republici Hrvatskoj tako i u stranim zemljama. Predstaviti će teorijski pregled već postojećih istraživanja te kako su oni pristupili ovom problemu. Nadalje, detaljno će biti opisan rad triju odabranih algoritama te metodologija korištena za njihovu usporedbu i implementaciju. Prikazani rezultati simulacija omogućit će evaluaciju svakog algoritma, dok će rasprava istaknuti njihove prednosti i slabosti. Na kraju, rad će biti zaokružen zaključkom.

1.1. Zadatak završnog rada

Cilj ovog rada je istražiti učinkovitost triju različitih algoritama za raspoređivanje medicinskih termina pacijentima. Analizirat ćemo ih prema broju raspoređenih pacijenata na raniji termin od predviđenog, vremenu čekanja i iskorištenosti dostupnih resursa. Kroz simulacije, ovaj rad nastoji identificirati najprikladniji algoritam za bolničko okruženje.

2. PREGLED DOSADAŠNJIH ISTRAŽIVANJA

Optimizacija raspoređivanja pacijenata na bolničke preglede predstavlja ključni korak u vremenu gdje je potražnja za zdravstvenu skrb znatno superiorna u odnosu na resurse. Stoga, radi poboljšanja učinkovitosti i kvalitete zdravstvenih usluga, različita istraživanja ispituju brojne metode i algoritme kako bi se pronašla najbolja rješenja za smanjenje listi čekanja, povećanje zadovoljstva pacijenata i optimalno korištenje resursa. U nastavku će biti prikazano pet značajnih istraživanja koja se bave ovom tematikom.

2.1. Prioritizacija pacijenata na kirurškim listama čekanja

Kao što je već naglašeno, Hrvatska nije jedina zemlja koja se bori sa zamornim listama čekanja te, prema [2], u Čileu 2021. godine gotovo 250.000 ljudi čekalo je na kirurški zahvat što se dodatno pogoršalo krizom COVID-19. Skupina znanstvenika uočila je mane postojeće strategije određivanja prioriteta pacijenata te su predstavili svoj inovativni sustav odlučivanja.

Određivanje prioriteta pri kirurškim zahvatima jedan je od najvažnijih aspekata. Prvo, ukoliko nije pravovremeno i ispravno evaluirana, može imati ozbiljne posljedice na zdravlje pacijenta. Naravno, nisu svi slučajevi jednako hitni te, sukladno tome, znatno se može promijeniti ishod liječenja. Hitni slučajevi moraju biti tretirani bez odgode kako bi se izbjeglo pogoršanje zdravstvenog stanja, stoga pravilna prioritizacija osigurava da će oni kojima je pomoć najpotrebnija biti zbrinuti. Nadalje, prioritizacija općenito omogućava učinkovitiju uporabu medicinskih resursa. Zdravstvene ustanove nerijetko su ograničene brojem kirurških sala, bolničkih kreveta, opreme i medicinskog osoblja.

Većina sustava za dodjeljivanje prioriteta oslanja se samo na kliničke kriterije, poput glavne bolesti (najvažnija ukoliko pacijent boluje od više), ozbiljnost, morbiditet (učestalost bolesti u populaciji), itd. No, autori ovog članka stavljaju naglasak na dodavanje više kriterija. Predložen je pristup određivanja prioriteta koji uzima u obzir osobni i društveni kontekst, obiteljsku pozadinu, sposobnost pacijenta za napredovanje unatoč kliničkom stanju, koliko je dugo pacijent na listi čekanja, kao i njihovo kliničko stanje te je važnost biopsihosocijalnih čimbenika potkrepljena mnogim istraživanjima.

Razvili su funkcije koje utjelovljavaju mnoge biopsihofizičke varijable te su one prikazane jednadžbama (2-1) i (2-2).

$$s'_p(t) = s'_p(I) + s'_p(I^*, j, t) = \sum_{i \in I} w_i \alpha_{i,p} + \sum_{i^* \in I^*} w_i^* \, \alpha_{i^*,p}(j, t) \tag{2-1}$$

$s'_p(t)$ predstavlja bodovanje pacijenta p u određenom trenutku t, pri čemu se uzimaju u obzir faktori koji su neovisni o vremenu i faktori koji ovise o vremenu čekanja. Prvi dio formule zbraja ocjene koje se temelje na fiksnim karakteristikama pacijenta (npr. ozbiljnost stanja, dob), a drugi one koje se mijenjaju s vremenom, naglašavajući kako se situacija pacijenta pogoršava što duže čeka na liječenje.

$$v_p(t) = \frac{t - f_p}{Jclin_p} \tag{2-2}$$

Formula (2-2) prikazuje relevantni prioritet $v_p(t)$ pacijenta p u trenutku t, gdje je:

- t – trenutno vrijeme,
- f_p – početno vrijeme čekanja pacijenta p,
- $Jclin_p$ – klinička težina pacijenta p (maksimalno vrijeme čekanja koje je liječnik odredio za pacijenta).

Ovu tehniku primjenjivali su tijekom 2018. i 2019. te ih usporedili sa prijašnjom metodom koja se koristila. Evaluacija uspješnosti testirana je izrazom (2-3).

$$a_{swl}(t) = \frac{1}{n} \sum_{p=1}^{n} (t - f_p) \tag{2-3}$$

Gdje je:

- n – broj pacijenata na listi čekanja,
- f_p – datum prijema pacijenta p,

dakle $t - fp$ predstavlja period koji je pacijent p proveo na listi čekanja od trenutka t.

Rezultati su predstavljeni tablicom 2.1.

Tablica 2.1. Efikasnost metode u usporedbi s prijašnjim godinama [2, str. 18].

T*	N	ASWL
31.12.2015	998	419
31.12.2016	1200	470
31.12.2017	1123	496
31.12.2018	1108	281
31.12.2019	1307	282

Sustav je implementiran u bolnici Talca, Čile, 2018. godine, gdje je postignuto značajno smanjenje prosječnog broja dana čekanja na operaciju s 419 na 282 dana unatoč povećanju broja pacijenata. Novi sustav nadmašio je prethodne metode prioritizacije kako kvantitativno, tako i kvalitativno.

2.2. Raspoređivanje u zdravstvenom sustavu u kontekstu optimizacije

Ovaj osvrt ističe problem zakazivanja prijema pacijenata, raspoređivanja medicinskih sestara, zakazivanja operacijskih sala i mnoštvo drugih problema povezanih s raspoređivanjem u zdravstvenom sektoru [3]. Cilj je poboljšati učinkovitost i kvalitetu zdravstvenih usluga izgradnjom sustava koji bi mogao smanjiti vrijeme čekanja pacijenata te olakšati pristup medicinskim uslugama. Iako se rad bavi rješavanjem više različitih problema, mi ćemo se osvrnuti samo na jedan – raspoređivanje pacijenata na prijem.

Zakazivanje pacijenata na prijem predstavlja složen kombinatorni problem. Potrebno je, dakle, efikasno raspodijeliti pacijente u bolničke sobe, na posebne odjele ispunjavajući potrebe pacijenata te na najbolji način iskoristiti dostupne resurse. Kako bi navedeno bilo izvedivo, potrebno je utvrditi tvrda, odnosno meka ograničenja. Tvrda ograničenja ne mogu se kršiti, dok su meka dio funkcije troškova pa ih stoga treba svesti na što manju mjeru.

U radu analizira se nekoliko načina na koji bi se ovaj problem mogao riješiti:

- Verzija 1:
  Pretpostavlja da su datumi prijema i otpusta unaprijed poznati te svaki pacijent treba zauzeti jedan krevet na određeno vrijeme. Nailazi na problem pridržavanja nekih ograničenja. U svakom slučaju, cilj je minimizirati sve mekane uvjete (npr. preferencije

pacijenta oko kapaciteta sobe, specifične potrebe pacijenta) uz poštivanje svih tvrdih uvjeta (dostupnost soba, fiksni datum prijema i otpusta, dva pacijenta ne mogu istovremeno biti u jednom krevetu...).

- Verzija 2:
  Obuhvaća dodatne scenarije u usporedbi sa prvom verzijom. Može uključivati promjenjive datume prijema i otpusta, dinamičko prilagođavanje kapaciteta, te uzimanje u obzir hitnih prijema i nepredviđenih događaja. Također, uključuje svođenje ukupnih troškova na minimum te učinkovito iskoristiti medicinske resurse uzimajući u obzir zadovoljstvo pacijenata.
- Verzija 3:
  Treća verzija problema zakazivanja prijema pacijenata složenija je i realističnija od prethodnih verzija, jer uključuje dodatne čimbenike i ograničenja. Obuhvaća više ciljeva optimizacije, integraciju različitih resursa te višestruke razine prioriteta pacijenata. Uzimajući u obzir više ciljeva i širu sliku, ova verzija uključuje simulaciju planiranja i ponovnog planiranja kako bi se postigao najoptimalniji ishod.

Autori su predstavili brojne metode optimizacije za rješavanje problema zakazivanja pacijenata, a za testiranje korišten je nasumično generirani skup podataka. Problem je sagledan kroz nekoliko metoda: Tabu pretraga – korištena za dodjelu kreveta pacijentima, cjelobrojno linearno programiranje (ILP) – jedan model za raspodjelu pacijenata koji su tek stigli te drugi za planiranje budućih prijema, metaheuristički algoritmi – BBO, AND i LSN...

Zaključak koji je proizašao ovom analizom tvrdi da različite metode imaju različite stupnjeve uspjeha te da ih je najbolje prilagoditi situaciji koja se pokušava riješiti. U tablici 2.2 možemo koje su se točno metode koristile na kojem skupu podataka.

Tablica 2.2. Pregled korištenih metoda za optimizaciju pacijenata [3, str. 456].

Naziv metode	**PASP verzija**	**Kategorija**
Tabu pretraga	verzija 1	Metaheuristika
Jedna hiper-heuristika	verzija 1	Heuristika
Kasno prihvaćanje penjanja po brdu	verzija 1	Metaheuristika
Lokalna pretraga	verzija 1	Metaheuristika
Simulirano kaljenje sa strukturom susjedstva	verzija 2	Metaheuristika
Lokalna pretraga	verzija 3	Metaheuristika
Tabu pretraga	verzija 1	Metaheuristika
Markovljev proces odlučivanja i dinamičko programiranje	verzija 2	Heuristika
Precizna matematika (eng. *Finemath*)	verzija 1	Metaheuristika
Optimizacija temeljena na biogeografiji	verzija 1	Metaheuristika
Nelinearno veliko poplavljivanje	verzija 1	Metaheuristika
Pretraga velikog susjedstva	verzija 2	Metaheuristika
Pristup generaciji stupaca	verzija 1	Heuristika
Mješovito cjelobrojno programiranje	verzija 1	Egzaktno
Generacija stupaca	verzija 2	Heuristika

2.3. Distribucija vremena čekanja u javnom zdravstvu

Još jedno istraživanje bavi se problematikom raspodjele pacijenata, fokusirajući se na distribuciju vremena čekanja u javnom zdravstvenom sustavu, posebno u kontekstu elektivnih operacija – unaprijed planiranih kirurških zahvata koji nisu hitne prirode. Cilj ovog rada je razumjeti kako obrasci prijema u bolnicama generiraju liste čekanja i koji čimbenici utječu na trajanje čekanja u

svrhu izgradnje teorijskog modela koji opisuje optimalnu raspodjelu vremena čekanja za bolnice s ograničenim kapacitetom [4].

Model prikazuje kako različiti faktori, poput kapaciteta bolnice, strukture troškova i postavljanje prioriteta pacijenata, utječu na raspodjelu vremena čekanja. Prema izrazu (2-4), ovaj model omogućava optimizaciju liječenja pacijenata tako da se maksimalizira ukupna korist.

$$U_t = g(k_t) \sum_d \sum_s g(k_{d,s,t}) \tag{2-4}$$

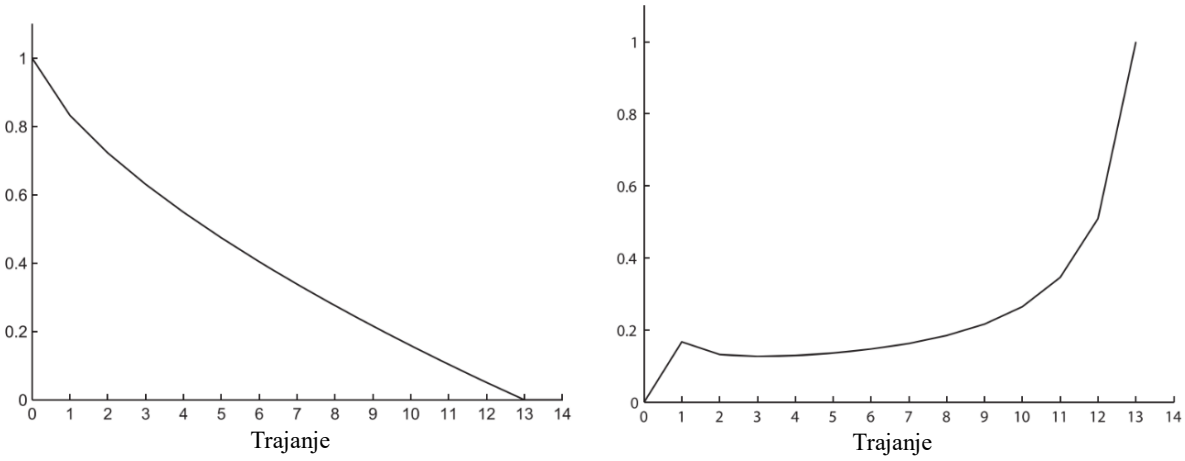
Ovdje se, dakle, sumira korist od liječenja pacijenata različite težine bolesti s i različitih vremenskih duljina čekanja d.

Troškovi pružanja zdravstvene skrbi u bolnici mogu se podijeliti na dvije komponente vidljive izrazom (2-5).

$$C_t = c(k_t; \bar{k}) + \sum_d \sum_s c\,(k_{d,s,t}) \tag{2-5}$$

Prva komponenta jednakosti $c(k_t; \bar{k})$ označava troškove bolnice koji ovise o ukupnom broju pacijenata koji se liječe u vremenskom trenutku k_t i kapacitetu bolnice $\bar{k}$, dok druga komponenta $\sum_d \sum_s c\,(k_{d,s,t})$ opisuje troškove specifične za trajanje čekanja i ozbiljnost bolesti.

Empirijska analiza koristi podatke *Hospital Episode Statistics* (HES) iz razdoblja od 1997. do 2005. godine. Rezultati analize pokazuju značajne razlike u obrascima prijema pacijenata među bolnicama. Na primjer, bolnice s većim kapacitetima $\bar{k}$ imaju tendenciju bržeg liječenja pacijenata, što dovodi do kraćeg vremena čekanja. Funkcija preživljavanja (engl. *survival function*) S(t) i funkcija opasnosti (engl. h*azard function*) h(t) ilustriraju brzinu kojom pacijenti napuštaju liste čekanja.



(a) Funkcija preživljavanja (b) Funkcija opasnosti

Slika 2.1. Funkcije preživljavanja i opasnosti za referentni model [4, str. 6].

Kao što je vidljivo sa slike 2.1., krivulja funkcije preživljavanja pokazuje da se vjerojatnost značajno smanjuje kako vrijeme čekanja raste, dok krivulja funkcije opasnosti pokazuje da se stopa opasnosti povećava nakon početnog perioda čekanja, s naglim porastom nakon određene točke.

Također, kao rezultat istraživanja proizlazi da bolnice koje više prioritiziraju pacijente s kraćim vremenom čekanja imaju strmije funkcije preživljavanja, dok bolnice koje ravnomjernije raspoređuju liječenje pacijenata imaju glađe. Ovi rezultati sugeriraju da bolnice s većim kapacitetima i ravnomjernijom strukturom troškova mogu učinkovitije upravljati listama čekanja i smanjiti vrijeme čekanja za pacijente.

Ove funkcije i grafički prikazi pomažu u razumijevanju utjecaja različitih čimbenika na raspodjelu vremena čekanja i pružaju uvid u učinkovite strategije za optimizaciju zakazivanja prijema pacijenata u bolnicama.

2.4. Raspored prijema pacijenata koristeći genetski algoritam

Prema [5], rad istražuje problem planiranja prijema pacijenata u oftalmološkoj bolnici primjenom genetskog algoritma (GA) s ciljem optimizacije korištenja bolničkih resursa, smanjenja vremena čekanja i povećanja učinkovitosti upravljanja bolnicom. Tradicionalno korišteni pristup "prvi dođe, prvi bude uslužen" (FCFS) zanemaruje različite utjecaje, poput duljine boravka pacijenata, za raspodjelu resursa. U ovom radu se predlaže sveobuhvatniji matematički model i evaluacijski mehanizam za planiranje prijema pacijenata.

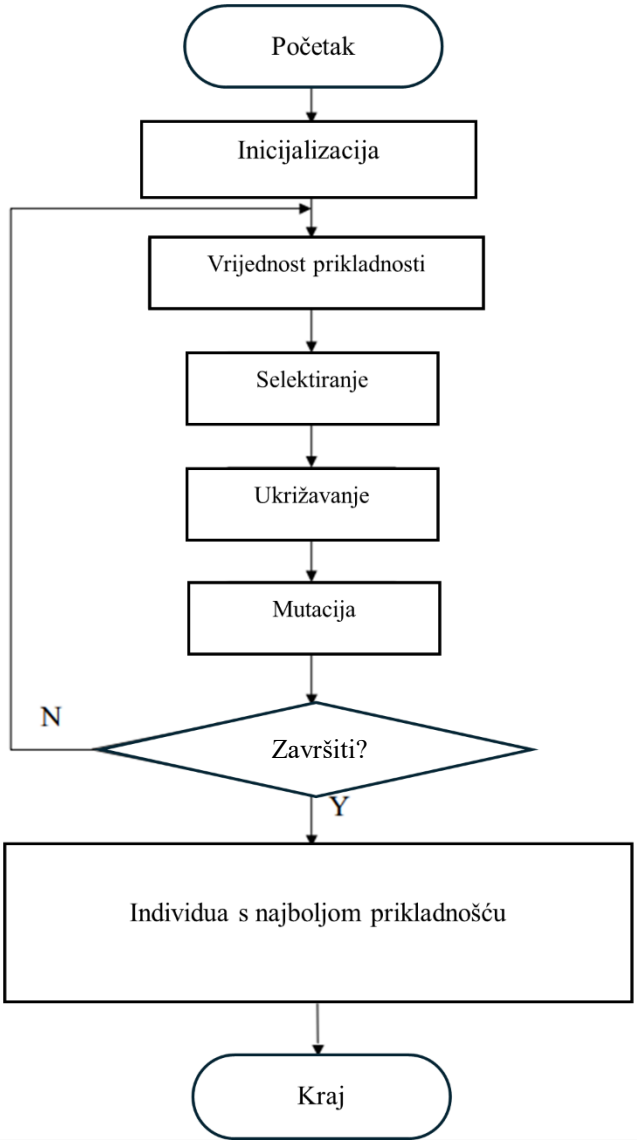
Model zakazivanja pacijenata objedinjuje nekoliko faktora:

- Vrste operacija – operacije katarakte, operacije mrežnice, operacije glaukoma...
- Ograničenja – operacije katarakte se obavljaju samo ponedjeljkom i srijedom, a sve ostale (koje nisu hitne) obavljaju se bilo koji drugi dan...
- Pretpostavke modela – pacijenti sa istom vrstom bolesti se primaju prema redoslijedu dolaska, pacijenti sa istom vrstom bolesti se operiraju prema redoslijedu prijema

Genetski algoritam (GA) je računalni pristup inspiriran prirodnom selekcijom i genetikom koji se koristi za optimizaciju složenih problema. Ključne komponente ovog algoritma su slijedeće:

- Populacija - predstavlja skup mogućih rješenja za problem. Svako rješenje u populaciji naziva se jedinka (kromosom) i sastoji se od gena (varijable)
- Kromosomsko kodiranje – rješenja su predstavljena kao kromosomi, gdje svaki gen označava broj pacijenata određene vrste bolesti koji će biti primljeni sljedećeg dana
- Funkcija prilagodbe – procjenjuje kvalitetu svakog rješenja (jedinke). U ovom radu, funkcija prilagodbe uzima u obzir prosječno vrijeme čekanja za prijem pacijenata, predoperativno vrijeme čekanja, broj pacijenata na listi čekanja i stopu popunjenosti kreveta
- Selekcija – daje prednost jedinkama s boljom prilagodbenom vrijednošću kako bi prenijele svoje gene u sljedeću generaciju
- Križanje – kombinira genetski materijal dviju jedinki kako bi stvorio nove jedinke s potencijalno boljim karakteristikama
- Mutacija – nasumično mijenja gene unutar jedinki kako bi se povećala raznolikost populacije i izbjeglo zapinjanje u lokalnim optimumima
- Iterativni proces – GA ponavlja proces selekcije, križanja i mutacije kroz nekoliko generacija dok se ne pronađe rješenje koje zadovoljava postavljene kriterije (ili dok se ne postigne zadani broj iteracija)

Na slici 2.2. prikazan je dijagram toka algoritma koji nam omogućuje bolje razumijevanje procesa.



Slika 2.2. Dijagram toka genetskog algoritma [5, str. 4].

Eksperiment je proveden u oftalmološkoj bolnici u Pekingu koristeći podatke o broju kreveta i listi čekanja na dan 11. rujna 2008. godine. Uspoređene performanse GA i tradicionalnog FCFS pristupa možemo vidjeti u tablici 2.3.

Tablica 2.3. Usporedba GA i FCFS algoritama [5, str. 7].

Procijenjeni broj otpuštenih pacijenata	**Prosječno vrijeme čekanja za prijem pacijenata**		**Predoperativno prosječno vrijeme čekanja primljenih pacijenata**		**Broj pacijenata na listi čekanja**	**Stopa slobodnih kreveta**
	FCFS	GA	FCFS	GA	FCFS/GA	FCFS/GA
11	6.154	6.308	2.364	2.000	91	100%
12	6.089	6.256	2.333	2.000	90	100%
13	6.022	6.202	2.308	2.000	89	100%
14	5.955	6.102	2.357	2.143	88	100%
15	5.897	6.046	2.333	2.133	87	100%
16	5.837	6.000	2.313	2.125	86	100%
17	5.776	5.929	2.294	2.118	85	100%
18	5.714	5.869	2.333	2.111	84	100%
19	5.651	5.795	2.368	2.158	83	100%
20	5.585	5.720	2.350	2.200	82	100%
21	5.519	5.642	2.333	2.238	81	100%
22	5.450	5.575	2.364	2.273	80	100%
23	5.308	5.504	2.250	2.306	79	100%
24	5.308	5.436	2.375	2.333	78	100%
25	5.247	5.364	2.400	2.360	77	100%

Genetski algoritam pokazao se kao učinkovit alat koji može znatno unaprijediti učinkovitost zakazivanja prijema pacijenata u usporedbi s tradicionalnim FCFS pristupom. GA nudi fleksibilnost u prilagođavanju različitim scenarijima i omogućuje optimizaciju prema različitim ciljevima. Rezultati pokazuju da GA smanjuje duljinu liste čekanja, povećava iskorištenost bolničkih kreveta i skraćuje predoperativno vrijeme čekanja, što je korisno za dostupnost bolničkih resursa i poboljšava upravljanje.

2.5. Optimizacija voznog reda

Naposlijetku, spomenut ćemo rad autorice I. Hartmann Tolić koji se bavi nešto drugačijom tematikom. Naime, raspoređivanje pacijenata na bolničke preglede sličan je problemu optimizacije voznog reda u mreži javnog prijevoza utoliko što oba imaju cilj maksimizirati učinkovitost korištenja resursa, smanjiti vrijeme čekanja korisnika te poboljšati ukupnu kvalitetu usluge. Ovaj rad pruža uvid u metode i algoritme primijenjene za optimizaciju rasporeda vozila u sustavu javnog prijevoza. Posebna pažnja posvećena je testiranju algoritma za raspodjelu putnika i analizi dobivenih rezultata, što može poslužiti kao korisna analogija za optimizaciju rasporeda pacijenata u zdravstvenim ustanovama [6].

Problem, kojim se ovaj rad bavi, je složen i obuhvaća višekriterijske optimizacijske modele koji uzimaju u obzir različite čimbenike poput vremena čekanja putnika, učestalosti vozila i operativnih troškova. Model se temelji na optimizaciji dvaju ciljeva: minimizacija vremena čekanja putnika te maksimizacija popunjenosti vozila. Za rješavanje korišten je MOPSO – algoritam optimizacije rojem čestica, koji je prilagođen za pretraživanje velikog prostora mogućih rješenja na način da generira Pareto optimalan skup rješenja među kojima se mora odabrati jedno optimalno.

U radu predstavljene su dvije već postojeće metode za određivanje preferiranog rješenja iz skupa optimalnih te je autorica prezentirala svoj doprinos novom inovativnom idejom uočivši nedostatke prijašnjih metoda – TOPSIS (engl. *technique for order of preference by similarity to ideal solution*) i GRA (engl. *gray relational analysis*). Predložena metoda zasniva se na normiranju ciljnih funkcija i korištenje euklidske norme kako bi se postigla jednaka težina ciljnih funkcija za određivanje preferiranog rješenja.

Rezultati pokazuju da predložena metoda za određivanje preferirano optimalnog rješenja daje prosječnu iskoristivost vozila od 91%, a statistika drugih metoda je sljedeća: TOPSIS 92% te GRA 85%. Na slici 2.3. možemo vidjeti usporedbu ovih metoda.

	$r_{a \to b}$				
		Polasci	d_o	$f_1(x)$	$f_2(x)$
q=6	Predložena metoda	[7:10,	70	0.095	0.035
		7:18,	490		
		7:29,	140		
		7:37,	490		
		7:45,	70		
		7:51]	490		
	TOPSIS	[7:10,	70	0.080	0.040
		7:19,	490		
		7:29,	70		
		7:36,	490		
		7:45,	70		
		7:51]	490		
	GRA	[7:10,	70	0.145	0.020
		7:17,	490		
		7:26,	490		
		7:36,	70		
		7:43,	490		
		7:51]	490		

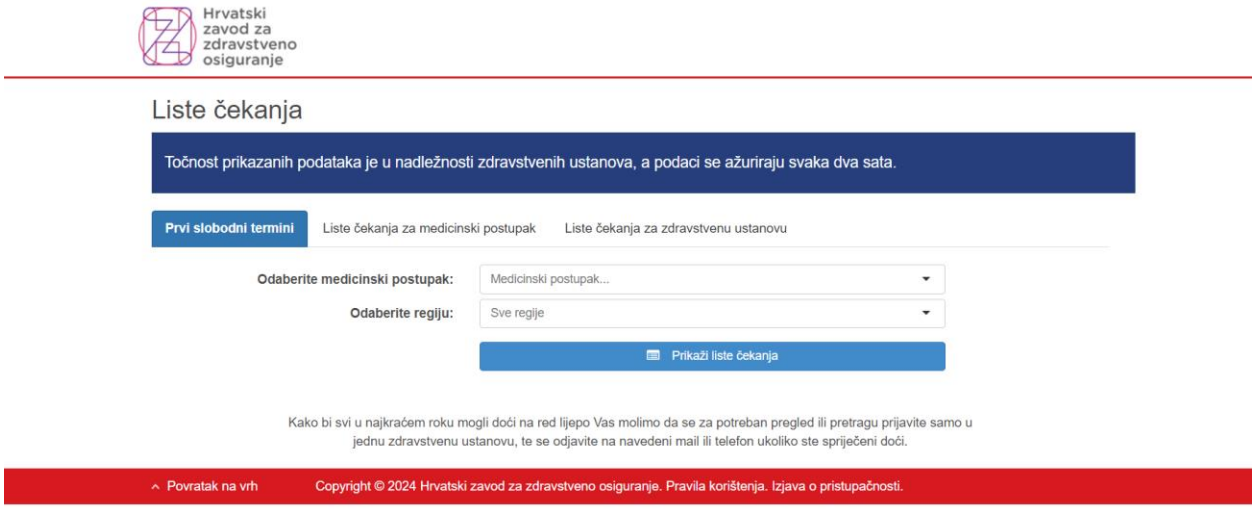
Slika 2.3. Usporedba preferiranih rješenja različitim metodama [6, str. 65].

Slika 2.3. prikazuje tablicu vremena polazaka, maksimalno opterećenje vozila d_o, vrijednosti funkcije iskoristivosti vozila $f_1(x)$ te funkcije čekanja $f_2(x)$ za svaku obrađenu metodu. Rezultat istraživanja pokazao je kako predložena metoda ima najbolje uravnoteženo rješenje za određivanje optimalnog voznog reda tako što nudi odličan balans između smanjenja čekanja putnika te efikasne iskoristivosti vozila.

3. RAZRADA TEME

Učinkovito raspoređivanje pacijenata na bolničke preglede predstavlja jedan od ključnih izazova u modernom zdravstvenom sustavu. U Hrvatskoj, kao i u mnogim drugim zemljama, liste čekanja za razne medicinske preglede dugotrajan su problem te mogu imati ozbiljne posljedice po zdravlje pacijenata. Prema [7], krajem 2023. godine čak 900.000 ljudi čekalo je na potreban pregled. Prema podacima Hrvatskog zavoda za zdravstveno osiguranje (HZZO), prosječno vrijeme čekanja za specijalistički pregled može iznositi nekoliko mjeseci, ovisno o vrsti pregleda i regiji.

Danas, zahvaljujući napretku tehnologije, možemo u svakom trenutku pregledati stanje za željeni medicinski pregled putem internetskih platformi. Na primjer, putem web stranice CEZIH liste čekanja [8], dostupni su termini za razne preglede u stvarnom vremenu. Ova platforma omogućava pacijentima da provjere dostupnost termina i vrijeme čekanja za različite medicinske preglede, što pomaže u boljem planiranju. Sučelje stranice prikazano je slikom 3.1.



Slika 3.1. Liste čekanja u Republici Hrvatskoj.

U usporedbi s drugim zemljama, Hrvatska se suočava s izazovima sličnim onima u drugim europskim zdravstvenim sustavima. Na primjer, u Velikoj Britaniji, unatoč postojanju Nacionalne zdravstvene službe (NHS), liste čekanja za određene procedure također mogu biti vrlo duge. NHS provodi razne strategije za smanjenje vremena čekanja, uključujući optimizaciju rasporeda pregleda i korištenje tehnologije za bolju organizaciju resursa. Također, postigli su značajan napredak u smanjenju dugih čekanja za hitnu i elektivnu skrb. Prema [9], broj pacijenata koji čekaju više od 18 mjeseci za elektivno liječenje smanjen je za više od 90%.

U Skandinavskim zemljama kao što su Švedska, Norveška i Danska, gdje su zdravstveni sustavi često pohvaljeni zbog svoje efikasnosti, koriste se digitalni alati za zakazivanje pregleda i sustavi prioritizacije kako bi pacijenti s najhitnijim potrebama dobili skrb što je prije moguće. Ove metode uključuju centralizirano upravljanje resursima i napredne tehnološke inovacije koje omogućavaju bolju koordinaciju i raspodjelu zdravstvenih resursa [10].

Tablica 3.1. Vrijeme čekanja za kirurške zahvate (u danima) [11, str. 18].

Država	Operacija katarakte	Zamjena kuka	Zamjena koljena	Histerektomija	Operacija prostate	Koronarno premoštenje	Koronarna angioplastika
Australija	84	119	209	61	44	17	
Kanada	66	105	122		40	6	
Čile	97	240	839	57	69	26	
Danska	36	35	44	23	36	10	15
Estonija	187	282	461				
Finska	97	77	99	55	39	15	23
Mađarska	36	43	85		10	22	
Izrael	77	56	85	31	36	5	
Italija	24	50	42	33	36	9	11
Novi Zeland	82	81	89	80	66	62	38
Norveška	132	123	152	118	105	62	43
Poljska	246	179	253				27
Portugal	119	126	204	77	81	5	
Španjolska	74	118	147	55	75	37	35
Švedska	51	75	90	32	45	7	
Velika Britanija	65	92	98	54	35	55	39
Oecd prosjek	**92**	**113**	**189**	**56**	**51**	**24**	**29**

Tablica 3.1. prikazuje medijan vremena čekanja za odabrane elektivne kirurške zahvate (u danima) za različite zemlje članice OECD-a u 2018. godini [11]. Iz podataka možemo izvući nekoliko ključnih zaključaka – zemlje poput Italije, Danske i Mađarske (koje u prosjeku imaju najmanja čekanja za navedene operacije) uspjele su implementirati učinkovite strategije koje smanjuju vrijeme čekanja, dok zemlje poput Estonije, Poljske i Čilea imaju duža vremena čekanja, što može ukazivati na potrebu za poboljšanjem upravljanja resursima i organizacijom zdravstvene skrbi.

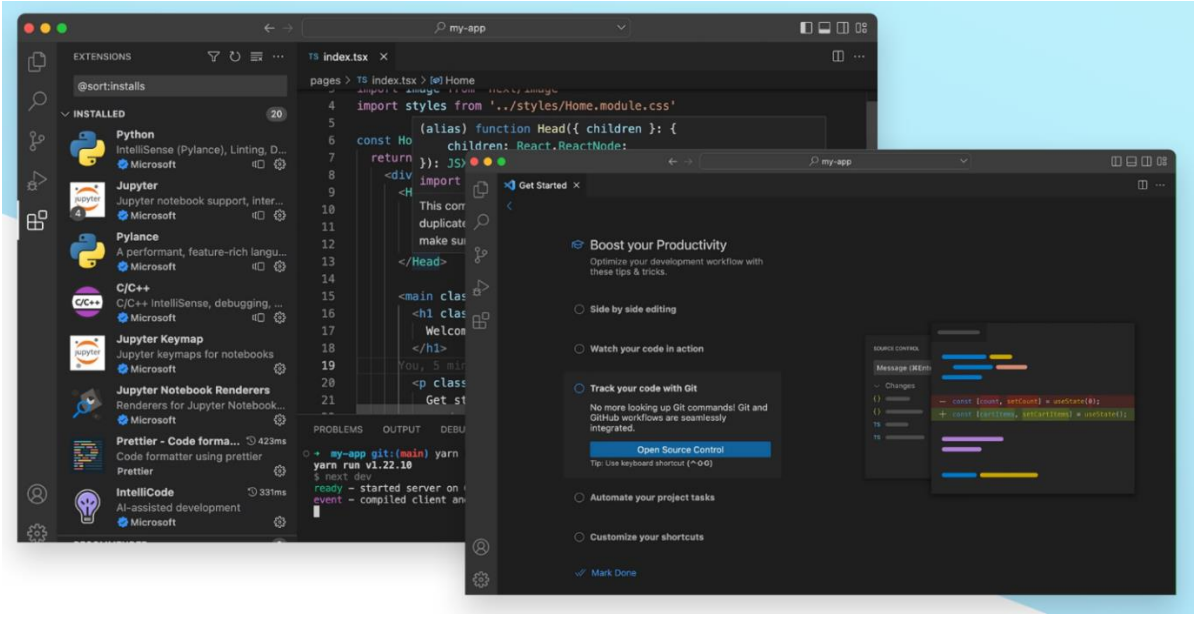
3.1. Tehnologije korištene u izradi rada

Efikasna raspodjela pacijenata na bolničke preglede predstavlja izazov koji zahtijeva učinkovite algoritme i tehnike optimizacije kako bi se postigli najbolji mogući rezultati. U ovome radu koristimo nekoliko ključnih tehnologija koje nam omogućuju analizu i manipulaciju podatcima prilikom testiranja različitih algoritama. Glavne tehnologije korištene u radu uključuju Python i brojne njegove biblioteke od kojih svaka igra ključnu ulogu u implementaciji algoritama za raspodjelu pacijenata i generiranju relevantnih podataka.

3.1.1. Visual Studio Code uređivač

Visual Studio Code je moćan alat koji podržava sve faze razvoja softvera, od pisanja i testiranja koda do upravljanja verzijama. Programeri ga preferiraju zbog njegove fleksibilnosti, mogućnosti proširenja i bogatog skupa značajki. Podržava brojne programske jezike, uključujući Python, JavaScript, HTML, C++ i druge. Trgovina ekstenzijama omogućava dodavanje dodatnih alata za povećanje funkcionalnosti, poput formatiranja koda i uključivanja umjetne inteligencije kao podrške pri pisanju koda [12].

Njegovo sučelje (Slika 3.2.) je intuitivno i prilagodljivo, što omogućava korisnicima da ga prilagode svojim potrebama.



Slika 3.2. Prikaz dizajna sučelja Visual Studio Code-a.

3.1.2. Python

Python je programski jezik visoke razine koji se ističe jednostavnom i lako razumljivom sintaksom. Razvio ga je Guido van Rossum, a prvi put je predstavljen 1991. godine. Python koristi interpretirani način izvršavanja, pri čemu se kod izvršava redak po redak, što olakšava otkrivanje i ispravljanje pogrešaka [13].

Na slici 3.3. nalazi se primjer Python koda koji zbraja dva broja.

```
x = input('Unesite prvi broj: ')
y = input('Unesite drugi broj: ')

sum = int(x) + int(y)

print(f"Zbroj {x} i {y} je {sum}")
```

Slika 3.3. Primjer Python koda.

Kao što možemo vidjeti, sintaksa je vrlo slična engleskom jeziku, što Python čini vrlo lakim za učenje te korištenje i razumijevanje.

Prema [14], mnoge popularne aplikacije također koriste ovaj jezik, a neke od njih su:

- Amazon – koristi Python za analizu podataka i preporučivanje proizvoda
- Google – koristi Python za pozadinske sustave (engl. *backend*) zbog lakoće održavanja i čitljivosti
- Facebook – koristi Python za izradu dijelova svog koda
- Spotify i Netflix – koriste Python za obradu podataka i preporuke

Python nudi širok raspon biblioteka koje pokrivaju različite domene primjene poput znanstvenih izračuna, strojnog učenja, analize podataka, web razvoja i mnogih drugih. U ovom radu korištene su sljedeće:

- Pandas – biblioteka za analizu podataka, koja omogućava jednostavno upravljanje strukturiranim podacima koristeći DataFrame objekte. Ova značajka je posebno korisna za manipulaciju podacima o pacijentima i terminima, pružajući širok spektar funkcija za filtriranje, grupiranje i agregaciju podataka.
- NumPy – služi za znanstvene izračune i rad s višedimenzionalnim nizovima
- Matplotlib – omogućuje vizualizaciju podataka kroz različite vrste grafova i dijagrama, poput linijskih grafova, stupčastih dijagrama i histograma. Nerijetko se integrira s Pandasom i NumPyjem za jednostavnu vizualizaciju podataka iz DataFrame-ova i nizova
- PuLP – koristi se za rješavanje problema linearnog i cjelobrojnog programiranja pomoću različitih algoritama za rješavanje (CBC, GLPK …).
- Random – biblioteka koja omogućava generiranje pseudo-nasumičnih brojeva i nudi razne funkcije za rad s nasumičnim podacima.
- Datetime – pruža klase za manipulaciju datuma i vremena, kao što su date, time, datetime i timedelta.

3.2. Generiranje sintetičkih podataka

Generiranje sintetičkih podataka je proces stvaranja lažnih podataka koji zadržavaju statistička svojstva stvarnih podataka. Ova tehnika omogućuje nam testiranje raznih modela i algoritama bez potrebe za stvarnim podatcima koji se često ne mogu dijeliti zbog zakonskih i etičkih ograničenja. Primjena ovakve prakse također nudi mogućnost ponavljanja i proširivanja istraživanja gdje u svega nekoliko klikova možemo mijenjati vrijednosti parametara te veličinu testnih skupova.

3.2.1. Pacijenti

Generiranje sintetičkih podataka o pacijentima uključuje nekoliko koraka koji osiguravaju da podatci budu dovoljno realistični za analizu i testiranje algoritama za raspodjelu pacijenata na bolničke preglede.

Za svakog pacijenta trebamo sljedeće podatke:

- OIB (jedinstveni identifikator pacijenta)
- grad iz kojeg pacijent dolazi
- zakazani datum pregleda u gradu pacijenta
- vrsta medicinskog pregleda
- trajanje pregleda (konzistentno za svaku vrstu pregleda)
- prioritet pacijenta
- maksimalna udaljenost koju je voljan putovati na pregled
- tolerancija pacijenta za koliko dana ranijeg pregleda je voljan putovati (ukoliko je termin preblizu zakazanog datuma, nema smisla putovati u drugi grad)

Ponajprije, radi jednostavnosti testiranja, ograničili smo gradove i vrste pregleda na način prikazan slikom 3.4.

```
cities = ["ZG", "OS", "ST", "RI", "SB"]

examination_types = {
    "MR": 1.5,
    "CT": 1.0,
    "Mamografija dojke": 0.5,
    "Ultrazvuk srca": 1.0,
    "Prvi pregled kod reumatologa": 1.0
}
```

Slika 3.4. Definiranje gradova i vrsta pregleda.

Definirali smo listu koja sadrži pet potencijalnih gradova: Zagreb, Osijek, Split, Rijeka te Slavonski Brod. Zatim, za skup pregleda koji će biti korišteni uzeli smo pet koji imaju najduže liste čekanja u RH, prema [15], te ih povezali pomoću rječnika s njihovim vremenima trajanja.

```
def generate_random_date():
    start_date = datetime.now()
    end_date = start_date + timedelta(days=5*30)
    random_date = start_date + timedelta(days=random.randint(0, 150))
    return random_date
```

Slika 3.5. Generiranje nasumičnih datuma.

Datume smo također generirali nasumično (Slika 3.5.), no raspon smo sveli na narednih pet mjeseci počevši od dana generiranja podataka. Funkcija vraća nasumični datum.

Zatim, kada smo definirali sve preduvjete, pacijente smo generirali funkcijom prikazanom na slici 3.6.

```
def generate_patient_data(num_patients):
    data = []
    current_date = datetime.now()

    for _ in range(num_patients):
        oib = random.randint(10000000000, 99999999999)
        city = random.choice(cities)
        scheduled_date = generate_random_date()
        examination_type = random.choice(list(examination_types.keys()))
        duration = examination_types[examination_type]
        priority = random.randint(1, 3)
        max_distance = random.randint(150, 600)
        days_until_scheduled_date = (scheduled_date - current_date).days
        tolerance = round(days_until_scheduled_date * random.uniform(0.25, 0.75))

        data.append([oib, city, scheduled_date.strftime("%Y-%m-%d"),
examination_type, duration, priority, max_distance, tolerance])

    columns = ["OIB", "City", "Scheduled Date", "Examination Type", "Duration",
"Priority", "Max Distance", "Tolerance"]
    return pd.DataFrame(data, columns=columns)
```

Slika 3.6. Funkcija za generiranje podataka o pacijentima.

Funkcija stvara nasumične podatke za svaki atribut pacijenta i vraća ih u obliku Pandas DataFrame-a. Kao parametar prima količinu pacijenata koju želimo generirati pa se kroz petlju svakom iteracijom stvaraju podatci za jednog pacijenta. Pojasnimo svaku stavku:

- oib – generira nasumični 11-zanmenkasti broj
- city – nasumično odabire jedan od gradova iz liste 'cities'
- scheduled_date – generira nasumični datum koristeći prethodno definiranu funkciju 'generate_random_date'
- examination_type – nasumično odabire vrstu pregleda iz ključeva rječnika 'examination_types'
- duration – dohvaća trajanje pregleda iz rječnika 'examination_types' za odabrani tip pregleda
- priority – dodjeljuje nasumični prioritet (može biti rangiran od 1 do 3, gdje manji broj označava veći prioritet)
- max_distance – nasumično odabire maksimalnu udaljenost koju pacijent može putovati (u kilometrima)
- tolerance – računa koliko dana prije svog termina je pacijent voljan putovati

Naposlijetku, funkcija vraća DataFrame stvoren pomoću liste 'data'.

```
patients_df = generate_patient_data(250)
patients_df.to_csv("patients.csv", index=False)
```

Slika 3.7. Generiranje nasumičnih pacijenata i spremanje u CSV datoteku.

Na slici 3.7. vidljivo je da smo za potrebe testiranja generirali 250 pacijenata te podatke spremili u datoteku kako bi iste mogli testirati na svim algoritmima.

3.2.2. Resursi

Generiranje sintetičkih podataka o resursima (dostupnim termina za preglede) omogućava simulaciju različitih scenarija i testiranje učinkovitosti algoritama za raspodjelu pacijenata.

Za svaki termin trebamo sljedeće podatke:

- ID resursa (jedinstveni identifikator resursa)
- grad u kojem je pregled dostupan
- datum pregleda
- vrsta medicinskog pregleda
- trajanje pregleda (u satima)

Na slici 3.8. vidimo funkciju koja je korištena za generiranje podataka o terminima pregleda te usporedimo li s funkcijom sa slike 3.6., vidimo da su gotovo identične jer se datum pregleda, vrsta pregleda, trajanje pregleda i grad generiraju na isti način.

```
def generate_resource_data(num_resources):
    data = []
    for i in range(num_resources):
        resource_id = i + 1
        city = random.choice(cities)
        examination_type = random.choice(list(examination_types.keys()))
        examination_date = generate_random_date()
        duration = examination_types[examination_type]

        data.append([resource_id, city, examination_date .strftime("%Y-%m-%d"),

        examination_type, duration])


    columns = ["ID", "City", "Examination Date", "Examination Type", "Duration"]
    return pd.DataFrame(data, columns=columns)

resources_df = generate_resource_data(150)
resources_df.to_csv("resources.csv", index=False)
```

Slika 3.8. Funkcija za generiranje podataka o dostupnim pregledima.

3.3. Implementacija algoritama

Kao što je u prijašnjem odlomku rečeno, sve algoritme testirati ćemo na istom skupu nasumično generiranih podataka kako bi ih pravedno mogli evaluirati. Optimizacija rasporeda pacijenata na bolničke preglede zahtijeva uzimanje u obzir različitih varijabli koje mogu utjecati na vrijeme čekanja i učinkovitost pružanja zdravstvene skrbi. Neke od kojih ćemo mi testirati su slijedeće:

- Hitnost slučaja – ozbiljnost zdravstvenog stanja pacijenta može utjecati na prioritet koji mu se daje pri rasporedu pregleda. Hitni slučajevi obično imaju veći prioritet.
- Tolerancija – broj dana ranijeg termina koji pacijent može prihvatiti, ukazujući na njegovu fleksibilnost; ako je novi termin preblizu zakazanog termina, pacijent neće imati koristi od putovanja u drugi grad.
- Spremnost na putovanje – pacijenti koji su spremni putovati dalje za specijalizirani pregled ili kraći rok čekanja mogu biti raspoređeni u bolnice s kraćim listama čekanja.

Prvi korak, prije same implementacije, jest učitavanje podataka o pacijentima i dostupnim terminima iz prije stvorenih CSV datoteka (Slika 3.9.).

```
patients_df = pd.read_csv('patients.csv')
resources_df = pd.read_csv('resources.csv')
```

Slika 3.9. Pozivanje podataka o pacijentima i terminima.

Zatim, radi potreba algoritama definirali smo udaljenosti između svih gradova pomoću rječnika. Koristit će se za provjeru je li udaljenost između dva grada unutar maksimalne udaljenosti koju pacijent može putovati. Implementacija je vidljiva na slici 3.10., a tablica udaljenosti, koju smo primijenili, nalazi se u prilogu 3.1.

```
distances = {
    ('ZG', 'OS'): 283, ('OS', 'ZG'): 283,
    ('ZG', 'ST'): 409, ('ST', 'ZG'): 409,
    ('ZG', 'RI'): 160, ('RI', 'ZG'): 160,
    ('ZG', 'SB'): 190, ('SB', 'ZG'): 190,
    ('OS', 'ST'): 687, ('ST', 'OS'): 687,
    ('OS', 'RI'): 439, ('RI', 'OS'): 439,
    ('OS', 'SB'): 93,  ('SB', 'OS'): 93,
    ('ST', 'RI'): 416, ('RI', 'ST'): 416,
    ('ST', 'SB'): 596, ('SB', 'ST'): 596,
    ('RI', 'SB'): 352, ('SB', 'RI'): 352,
    ('OS', 'OS'): 0,
    ('ST', 'ST'): 0,
    ('RI', 'RI'): 0,
    ('SB', 'SB'): 0,
    ('ZG', 'ZG'): 0
}
```

Slika 3.10. Udaljenost između gradova.

3.3.1. FCFS

FCFS (engl. *First-Come, First-Served*) je jednostavan algoritam koji obrađuje zahtjeve prema redoslijedu dolaska. Kada se primjenjuje na raspodjelu pacijenata za preglede, FCFS dodjeljuje termine pacijentima prema redoslijedu prijave, ne vodeći računa o hitnosti ili trajanju pregleda. Ovaj algoritam može se koristiti u različitim problemima, npr. za upravljanje redovima u bankama, telefonske linije za podršku..., no prvotno služi operacijskim sustavima kako bi se rasporedilo vrijeme korištenja procesora procesima u redoslijedu dolaska [16].

Jednostavnost ovog algoritma leži u njegovoj lakoj implementaciji i razumijevanju, također je i pravedan utoliko što obrađuje svaki zahtjev prema redoslijedu dolaska, bez ikakve prioritizacije. No, upravo to može biti veliki nedostatak u slučajevima poput raspodijele pacijenata gdje prioritet postaje glavni aspekt prilikom raspodijele dostupnih resursa. Slika 3.11. prikazuje nam primjer FCFS algoritma u kontekstu raspoređivanja resursa procesora procesima koji nailaze.

Procesi	Trajanje	Redoslijed	Vrijeme dolaska
P1	24	1	0
P2	3	2	0
P3	4	3	0

Ganttov dijagram:

Slika 3.11. Primjer FCFS algoritma [17].

Nakon što smo učitali potrebne podatke, potrebno je sortirati pacijente prema datumu zakazanog termina (Slika 3.12.). To znači da pacijenti čiji su termini ranije zakazani imaju prednost u raspoređivanju.

```
sorted_patients = patients_df.sort_values(by=['Scheduled Date'])
```

Slika 3.12. Sortiranje pacijenata prema datumu.

```
accepted_new_dates = []
stayed_on_original = []
waiting_times = []

available_resources = resources_df.copy()

for _, patient in sorted_patients.iterrows():
     patient_scheduled_date  =  datetime.strptime(patient['Scheduled  Date'],  "%Y-%m-
%d")
    found_new_appointment = False

    for _, resource in available_resources.iterrows():
        resource_date = datetime.strptime(resource['Examination Date'], "%Y-%m-%d")

        if resource_date < patient_scheduled_date:
            if (patient['Examination Type'] == resource['Examination Type'] and
                            abs((patient_scheduled_date   -   resource_date).days)   <=
patient['Tolerance'] and
```

```
                    distances[(patient['City'], resource['City'])] <= patient['Max
Distance']):

                accepted_new_dates.append({
                    'Patient OIB': patient['OIB'],
                    'Resource ID': resource['ID'],
                    'Original Scheduled Date': patient['Scheduled Date'],
                    'New Examination Date': resource['Examination Date'],
                    'City': resource['City'],
                    'Examination Type': resource['Examination Type'],
                  'Waiting Time (days)': (patient_scheduled_date - resource_date).days
                })

                waiting_time = (patient_scheduled_date - resource_date).days
                waiting_times.append(waiting_time)

                available_resources = available_resources.drop(resource.name)
                found_new_appointment = True
                break

    if not found_new_appointment:
        stayed_on_original.append({
            'Patient OIB': patient['OIB'],
            'Scheduled Date': patient['Scheduled Date'],
            'City': patient['City'],
            'Examination Type': patient['Examination Type']
        })

accepted_new_dates_df = pd.DataFrame(accepted_new_dates)
stayed_on_original_df = pd.DataFrame(stayed_on_original)
```

Slika 3.13. Implementacija FCFS algoritma.

Algoritam raspodjeljuje pacijente prema njihovim zakazanim terminima, s naglaskom na pronalaženje ranijih termina unutar određenih ograničenja, poput vrste pregleda, tolerancije za raniji termin i maksimalne udaljenosti između gradova (Slika 3.13.). Imamo dvije glavne kategorije pacijenata:

- Pacijenti koji su prihvatili raniji termin – ovi pacijenti se pohranjuju u listu 'accepted_new_dates' kada im je pronađen termin koji zadovoljava kriterije.
- Pacijenti koji su ostali na originalnom terminu – pacijenti koji nisu mogli dobiti raniji termin ostaju na svom zakazanom terminu i pohranjuju se u listu 'stayed_on_original'.

Petljama iteriramo kroz redove DataFrame-a 'sorted_patients' i ' available_resources'. Varijable 'patient' i 'resource' predstavljaju trenutni red (tj. pacijenta/slobodni termin) u svakoj iteraciji. U ovom slučaju, indeks reda nije potreban, pa se koristi '_'.

Termin je prihvatljiv samo ako ispunjava sve sljedeće uvjete:

- Datum termina mora biti prije pacijentovog originalno zakazanog termina.
- Termin mora biti iste vrste pregleda kao što je pacijent zakazao.
- Termin mora biti unutar tolerancije za raniji termin, odnosno broj dana između originalnog i novog termina mora biti manji ili jednak toleranciji pacijenta.
- Udaljenost između grada pacijenta i grada u kojem se održava pregled mora biti unutar maksimalne udaljenosti koju je pacijent spreman putovati.

Ako termin zadovoljava sve ove uvjete, pacijent prihvaća raniji termin, te se uklanja iz liste dostupnih termina kako bi se spriječilo da ga drugi pacijenti koriste. Ukoliko pacijent ne pronađe raniji termin koji ispunjava sve kriterije, ostaje na originalnom terminu.

3.3.2. PS

PS algoritam (engl. *Prioritiy Scheduling*) raspoređuje zadatke ili zahtjeve prema određenim prioritetima[18]. U kontekstu zdravstvene skrbi, ovaj algoritam se može koristiti za dodjelu pacijenata resursima (terminima pregleda) temeljem prioritetnih kriterija poput hitnosti. Pravovremeno pružanje zdravstvene njege je iznimno važno, jer odgađanje istih može imati ozbiljne posljedice.

Osnovna ideja je da svaki proces ima dodijeljen prioritet, te se procesi s najvišim prioritetom izvršavaju odmah po dolasku te prekida rad prijašnjeg procesa. Na slici 3.14. nalazi se primjer ovakvog algoritma koji nam može pomoći shvatiti princip.

Procesi	P1	P2	P3	P4	P5
Vrijeme izvođenja	3	4	2	1	3
Prioritet	1	3	2	1	4

Slika 3.14. Primjer PS algoritma [19].

Nažalost ovakav pristup ima i svoje nedostatke, npr. ako nije pravilno dizajniran, algoritam može dovesti do toga da pacijenti s nižim prioritetom stalno čekaju.

```
sorted_patients = patients_df.sort_values(by=['Priority'], ascending=True)

ps_schedule = []
waiting_times = []
accepted_new_dates = []
stayed_on_original = []

for _, patient in sorted_patients.iterrows():
    patient_scheduled_date = datetime.strptime(patient['Scheduled Date'], "%Y-%m-%d")

    found_new_appointment = False

    for _, resource in resources_df.iterrows():
        resource_date = datetime.strptime(resource['Examination Date'], "%Y-%m-%d")

         if resource_date < patient_scheduled_date and abs((patient_scheduled_date -
resource_date).days) <= patient['Tolerance']:

                if distances[(patient['City'], resource['City'])] <= patient['Max
Distance']:

               accepted_new_dates.append({

                    'Patient OIB': patient['OIB'],
                    'Priority': patient['Priority'],
                    'Original Scheduled Date': patient['Scheduled Date'],
                    'New Examination Date': resource['Examination Date'],
                    'City': resource['City'],
                    'Examination Type': resource['Examination Type'],
```

```
                    'Waiting Time (days)': (resource_date - current_date).days
                })

                waiting_times.append((resource_date - current_date).days)

                resources_df = resources_df.drop(resource.name)
                found_new_appointment = True
                break

    if not found_new_appointment:

        stayed_on_original.append({
            'Patient OIB': patient['OIB'],
            'Priority': patient['Priority'],
            'Scheduled Date': patient['Scheduled Date'],
            'City': patient['City'],
            'Examination Type': patient['Examination Type'],
            'Waiting Time (days)': (patient_scheduled_date - current_date).days})
        waiting_times.append((patient_scheduled_date - current_date).days)

accepted_new_dates_df = pd.DataFrame(accepted_new_dates)
stayed_on_original_df = pd.DataFrame(stayed_on_original)
```

Slika 3.15. Implementacija PS algoritma.

Koraci implementacije su slijedeći – nakon što smo učitali potrebne podatke iz CSV datoteka, radi lakše izvedbe, sortirali smo pacijente prema prioritetu (Slika 3.15.). Ovim korakom osiguravamo da pacijenti s višim prioritetom imaju prednost u pronalasku ranijeg termina.

Nadalje, algoritam prolazi kroz sve pacijente redom, počevši od pacijenta s najvišim prioritetom. Za svakog pacijenta, najprije se pretvara datum zakazanog termina u format datuma za usporedbu. Zatim, provjerava se u dostupnim resursima postoji li raniji termin za pacijenta koji zadovoljava sve uvjete. Ako pacijent prihvati raniji termin, dodaje se u listu 'accepted_new_dates', zajedno s informacijama o originalnom terminu, novom terminu, gradu i trajanju čekanja (razlika između originalnog i novog termina). Vrijeme čekanja se također bilježi u listi 'waiting_times' koja će se koristiti u kasnijoj evaluaciji. Na posljetku, pacijenti koji nisu uspjeli pronaći raniji termin ostaju na originalnom, a dodijeljeni resursi brišu se kako bi se izbjegla ponovna upotreba.

3.3.3. ILP

Cjelobrojno linearno programiranje (ILP) je matematička metoda optimizacije koja se koristi za pronalaženje optimalnog rješenja za probleme raspodjele resursa, planiranje i raspoređivanje. Ova

metoda uključuje linearnu ciljnu funkciju koja je podložna linearnim jednakostima i nejednakostima, pri čemu su neke ili sve varijable ograničene na cijele brojeve [20-23].

Osnovni dijelovi koji čine ILP model su:

- Varijable odluke (engl. *decision variables*) – predstavljaju odluke koje treba donijeti, u ovom slučaju imati ćemo jednu binarnu varijablu x[i][j] koja će biti 1 ukoliko je pacijent *i* raspoređen na termin *j*, inače je 0.
- Ciljna funkcija (engl. *objective function*) – matematička funkcija koju treba maksimizirati ili minimizirati (obično je linearna kombinacija varijabli odluke). Pri rješavanju problema raspodjele pacijenata, ciljna funkcija će biti maksimizacija ukupnog broja raspoređenih pacijenata (pacijenti koji su dobili raniji termin od predviđenog).

$$\boldsymbol{maximize} \quad \sum_{i=1}^{N} \sum_{j=1}^{M} x[i][j] \tag{0-1}$$

- Ograničenja (engl. *constraints*) – skup linearnih jednakosti ili nejednakosti koje varijable moraju zadovoljavati. Osiguravaju da rješenja budu izvediva u kontekstu stvarnog problema, poput ograničenja resursa ili kapaciteta pa ćemo tako za ovaj specifični primjer imati tri ograničenja:
  1. Svaki pacijent može biti raspoređen na najviše jedan termin – sprječava situaciju gdje jedan pacijent bude zakazan za više termina, što bi bilo nepraktično i neoptimalno

$$\sum_{j=1}^{M} \mathbf{x[i][j]} \leq \mathbf{1}, \forall \boldsymbol{i} \tag{0-2}$$

  2. Svaki termin može biti dodijeljen najviše jednom pacijentu – sprječava situaciju gdje više pacijenata bude zakazano za isti termin, što bi dovelo do sukoba i nemogućnosti pregleda

$$\sum_{i=1}^{M} \mathbf{x[i][j]} \leq \mathbf{1}, \forall \boldsymbol{j} \tag{0-3}$$

3. Usklađenost vrste pregleda, gradova i tolerancije oko termina te maksimalne udaljenosti

$$if\ (E[i] \neq P[j]\ or\ C[i] \neq G[j]\ or\ \mid D[i] - T[j] \mid > F[i]\ or\ dist(G[i], C[j]) > O[i]) \Rightarrow x[i][j] = 0 \quad (0\text{-}4)$$

Gdje je: N broj pacijenata, M broj dostupnih termina, E[i] vrsta pregleda pacijenta *i*, P[j] vrsta pregleda termina *j*, C[i] grad pacijenta *i*, G[j] grad termina *j*, D[i] zakazani datum pacijenta *i*, T[j] datum termina *j*, F[i] tolerancija pacijenta *i* u danima, a O[i] maksimalna udaljenost koju pacijent *i* može putovati.

Implementaciju ovog modela omogućuje nam biblioteka PuLP.

```
N = len(patients_df)
M = len(resources_df)

prob = LpProblem("PatientScheduling", LpMaximize)
```

Slika 3.16.. Inicijalizacija i definiranje ILP problema.

Prvi korak, kao što je vidljivo na slici 3.16., definirali smo broj pacijenata i broj resursa kao duljinu njihovih DataFrame-ova i zatim inicijalizirali ILP problem s imenom 'PatientScheduling' i ciljem maksimizacije.

```
x = {}
for i in range(N):
    for j in range(M):
        x[(i, j)] = LpVariable(f"x_{i}_{j}", 0, 1, LpBinary)

objective = 0
for i in range(N):
    for j in range(M):
        objective += x[(i, j)]
prob += objective
```

Slika 3.17. Definiranje varijabli i ciljane funkcije ILP problema.

U nastavku (Slika 3.17.) kreirali smo varijablu za alokaciju pomoću rječnika. 'LpVariable' kreira binarnu varijablu $x_{i,j}$ koja označava je li pacijent *i* raspoređen na termin *j*. Zatim smo inicijalizirali varijablu0 'objective', koja će se koristiti za definiranje ciljne funkcije. Ciljna funkcija predstavlja ono što želimo maksimizirati ili minimizirati u našem ILP problemu. U ovom slučaju, želimo maksimizirati broj pacijenata koji su uspješno raspoređeni na nove, ranije, termine.

```
for i in range(N):
    suma = 0
    for j in range(M):
        suma += x[(i, j)]
    prob += suma <= 1
for j in range(M):
    suma = 0
    for i in range(N):
        suma += x[(i, j)]
    prob += suma <= 1
for i in range(N):
    for j in range(M):
        patient_city = patients_df.loc[i, 'City']
        resource_city = resources_df.loc[j, 'City']
        patient_exam = patients_df.loc[i, 'Examination Type']
        resource_exam = resources_df.loc[j, 'Examination Type']
        patient_tolerance = patients_df.loc[i, 'Tolerance']
        max_distance = patients_df.loc[i, 'Max Distance']

        patient_scheduled_date = datetime.strptime(patients_df.loc[i, 'Scheduled
Date'], "%Y-%m-%d")
        resource_date = datetime.strptime(resources_df.loc[j, 'Examination Date'],
"%Y-%m-%d")
        date_difference = abs((patient_scheduled_date - resource_date).days)

        if (distances[(patient_city, resource_city)] > max_distance or
            patient_exam != resource_exam or
            date_difference > patient_tolerance):
            prob += x[(i, j)] == 0
```

Slika 3.18.. Postavljanje ograničenja u ILP problemu.

Potom, dodajemo ranije definirana ograničenja te ih implementiramo eksplicitnim petljama (Slika 3.18.). Mogli smo ih zamijeniti koristeći, tzv. *list comprehensions* gdje bi sintaksa izgledala nešto drugačije, no odlučili smo se za ovaj pristup radi jednostavnosti. U suštini, prve dvije petlje vrlo su slične te obje prolaze kroz sve pacijente i resurse. Varijabla 'suma' će, po završetku iteracija,

sadržavati broj pacijenata (resursa) koji su dodijeljeni terminu *j* (pacijentu *i*). Na kraju, ograničenja se dodavaju u ILP problem 'prob' samo ako je zadovoljen uvjet – da suma nije veća od 1. U suprotnom to bi značilo da je više pacijenata (resursa) dodijeljeno jednom resursu (pacijentu). Posljednja petlja provjerava je li vrsta pregleda ista, je li datum termina unutar pacijentove tolerancije i je li udaljenost unutar pacijentove maksimalne udaljenosti. Ako bilo koji uvjet nije zadovoljen, varijabla $x_{i,j}$ je postavljena na 0.

```
prob.solve()

accepted_new_dates = []
waiting_times = []
for i in range(N):
    found_new_appointment = False
    patient_scheduled_date = datetime.strptime(patients_df.loc[i, 'Scheduled Date'],
"%Y-%m-%d")

    for j in range(M):
        if x[(i, j)].varValue == 1:
            resource_city = resources_df.loc[j, 'City']
            resource_exam = resources_df.loc[j, 'Examination Type']
            resource_date = datetime.strptime(resources_df.loc[j, 'Examination
Date'], "%Y-%m-%d")
            waiting_time = (patient_scheduled_date - resource_date).days

            accepted_new_dates.append({
                'Patient OIB': patients_df.loc[i, 'OIB'],
                'Resource ID': resources_df.loc[j, 'ID'],
                'Examination Date': resources_df.loc[j, 'Examination Date'],
                'City': resource_city,
                'Examination Type': resource_exam,
                'Waiting Time (days)': waiting_time
            })
            waiting_times.append(waiting_time)
            found_new_appointment = True
            break

accepted_new_dates_df = pd.DataFrame(accepted_new_dates)
```

Slika 3.19.. Rješavanje ILP problema i prikupljanje rezultata.

Na samom kraju pozivamo funkciju solve() za rješavanje ILP problema (Slika 3.19.). Ova funkcija pokušava pronaći optimalno rješenje za raspodjelu pacijenata na termine, uzimajući u obzir sva definirana ograničenja. Također, kao parametar možemo joj proslijediti neki od algoritama za

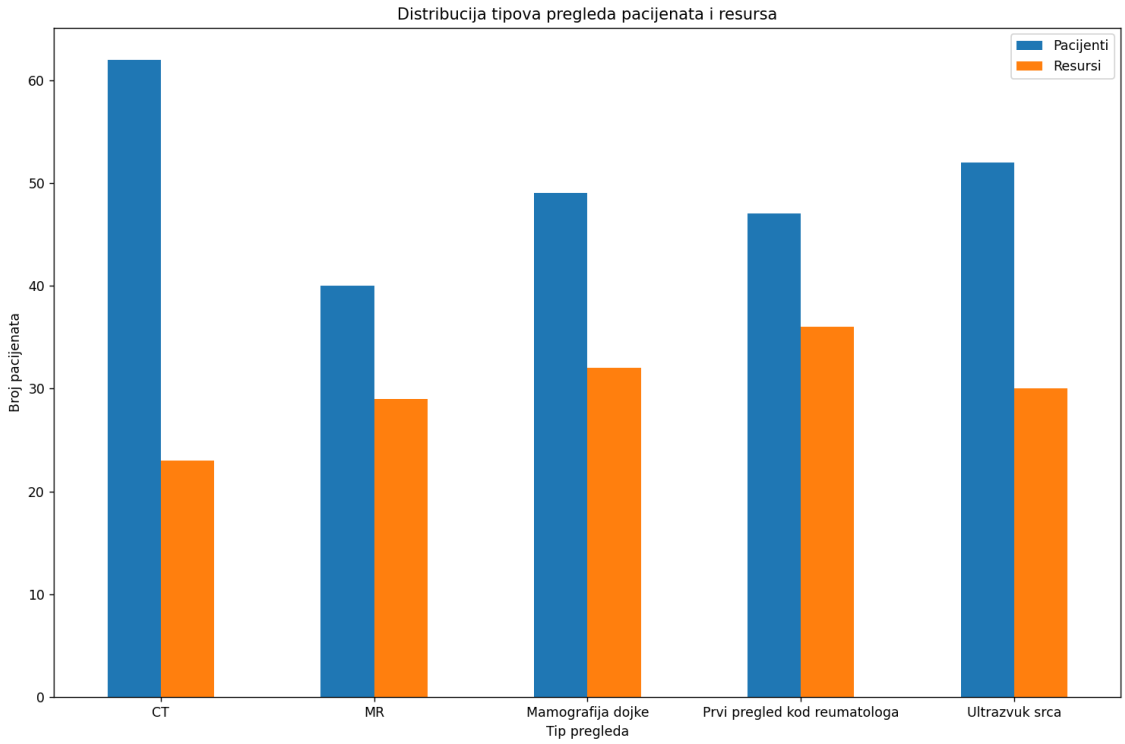
rješavanje, kao što su CBC, GLPK, CPLEX, itd. No, pošto nismo eksplicitno naveli koji želimo koristiti, biblioteka PuLP automatski će koristiti zadani (često je to CBC – *Coin-or branch and cut solver*). Algoritam prolazi kroz pacijente i dodjeljuje im termine na temelju rješenja ILP problema. Ako pacijent dobije raniji termin, bilježe se podaci o tom terminu (grad, vrsta pregleda, datum, vrijeme čekanja) i dodaje se u listu 'accepted_new_dates', a u suprotnom dodaje se u listu 'stayed_on_original' koja se, po završetku, pohranjuje u DataFrame.

4. RASPRAVA I ANALIZA REZULTATA

Za potrebe ove analize koristili smo sintetičke podatke generirane za 250 pacijenata i 150 resursa, što predstavlja omjer 5:3. Ovaj omjer je odabran kako bi se simulirala realistična situacija u kojoj postoji veća potražnja za terminima nego što ih je dostupno. Nije imalo smisla koristiti jednak broj pacijenata i resursa (1:1) jer, iako je teorijski ima dovoljan broj resursa, mala je vjerojatnost da će svi pacijenti dobiti termin pošto određeni uvjeti moraju biti zadovoljeni.

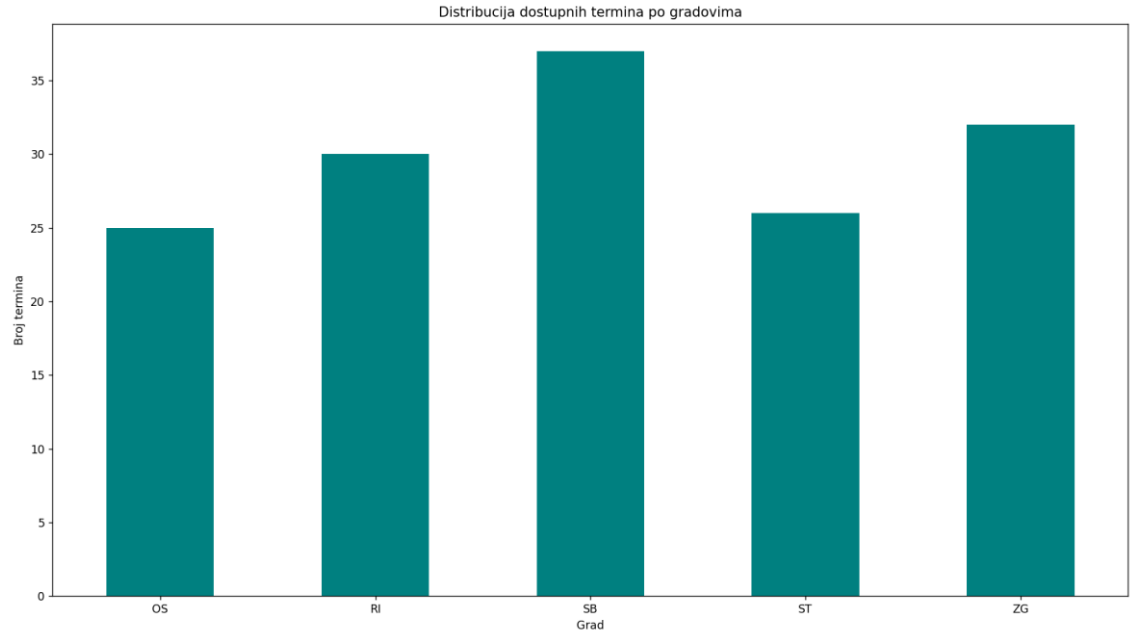
4.1. Analiza početnih skupova podataka

Za bolje razumijevanje i tumačenje rezultata evaluacije algoritama, ključno je upoznati se s podacima o pacijentima i dostupnim terminima koji su nasumično generirani. U nastavku su prikazani različiti grafovi koji nam pomažu vizualizirati i analizirati ove podatke, pružajući uvid u distribuciju i razlike između pacijenata i resursa.



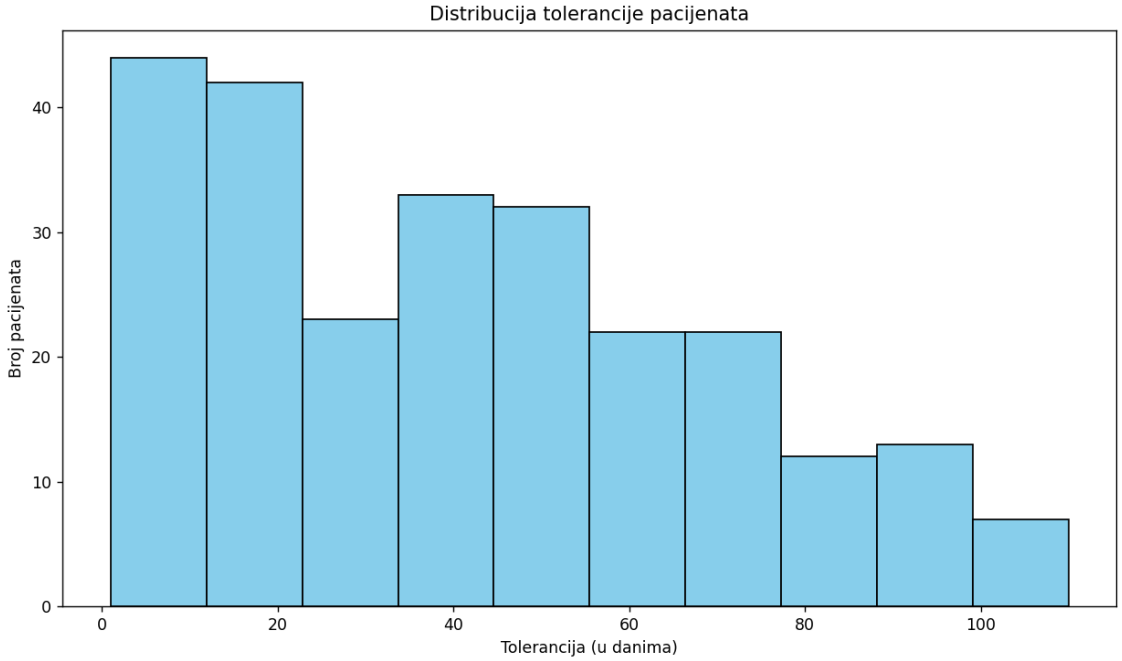
Slika 4.1. Distribucija tipova pregleda pacijenata i resursa.

Kao što je prikazano slikom 4.1., koja prikazuje distribuciju različitih pregleda po skupovima, postoji značajna razlika između broja pacijenata i dostupnih resursa za sve tipove pregleda. Najveća razlika je kod CT-a, gdje broj pacijenata značajno premašuje broj dostupnih termina.



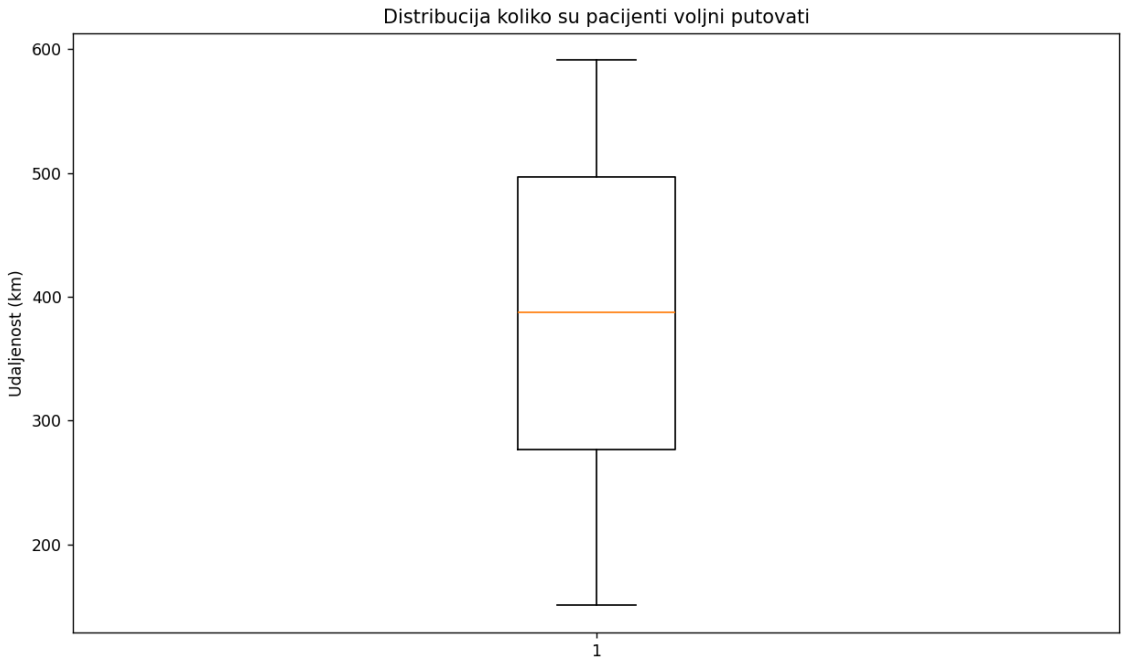
Slika 4.2. Distribucija dostupnih termina po gradovima.

Slavonski Brod (SB) ima najveći broj dostupnih termina, dok Osijek (OS) ima najmanje, a distribucija termina između ostalih gradova relativno je ujednačena (Slika 4.2.).



Slika 4.3. Distribucija tolerancije pacijenata u danima.

Graf sa slike 4.3. prikazuje distribuciju razine tolerancije pacijenata (izraženu u danima). Tolerancija u ovom kontekstu označava koliko dana ranije pacijent može prihvatiti termin u odnosu na svoj prvotno zakazani datum. Kako vidimo, većina pacijenata ima manju toleranciju – između 0 i 40 dana.

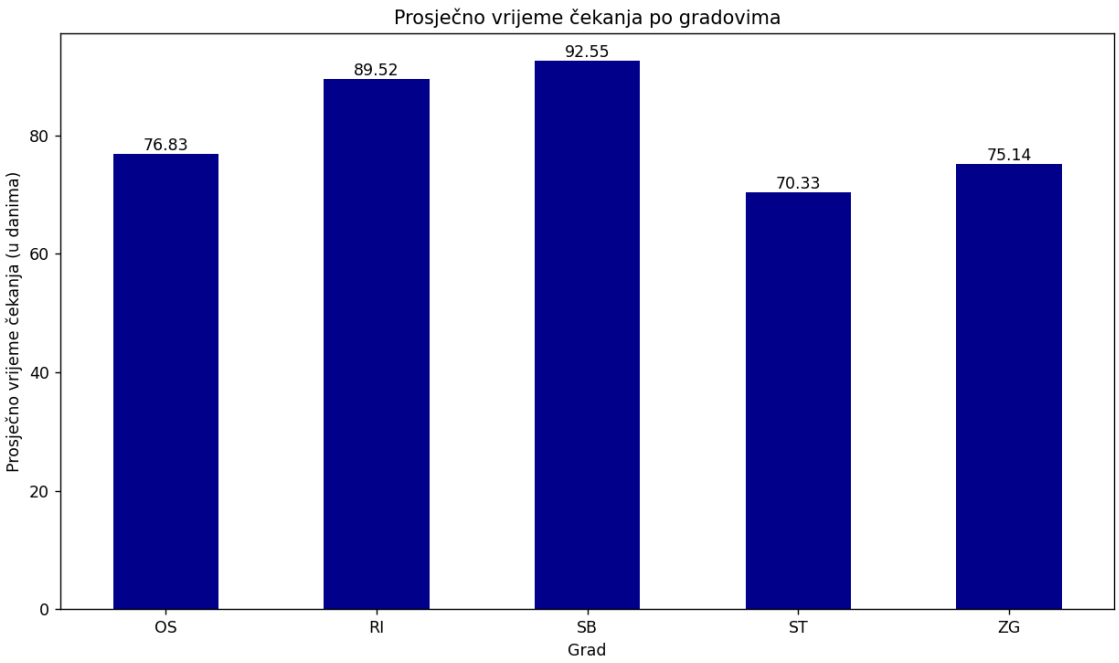


Slika 4.4. Distribucija udaljenosti koje su pacijenti spremni putovati.

Ovaj graf ukazuje na raspon udaljenosti koje su pacijenti spremni putovati, kao i na centralnu tendenciju i varijabilnost podataka. Pri generiranju ograničili smo udaljenosti na interval između 150 i 600km, a kao rezultat generiranih podataka dobili smo da je većina pacijenata voljna putovati između 300 km i 500 km, s medijanom oko 400 km (Slika 4.4.).

4.2. Evaluacija algoritama i metoda

Prije analize algoritama, izveli smo ispitivanje početnog stanja kako bismo utvrdili prosječno vrijeme čekanja za pregled koji je pacijentima dodijeljen bez ikakvih optimizacija. Cilj ovoga je uspostaviti osnovnu usporedbu s budućim testiranim metodama, kako bi se ocijenilo koliko navedeni algoritmi mogu poboljšati raspored pacijenata u odnosu na neoptimizirano stanje.



Slika 4.5. Rezultati početnog stanja.

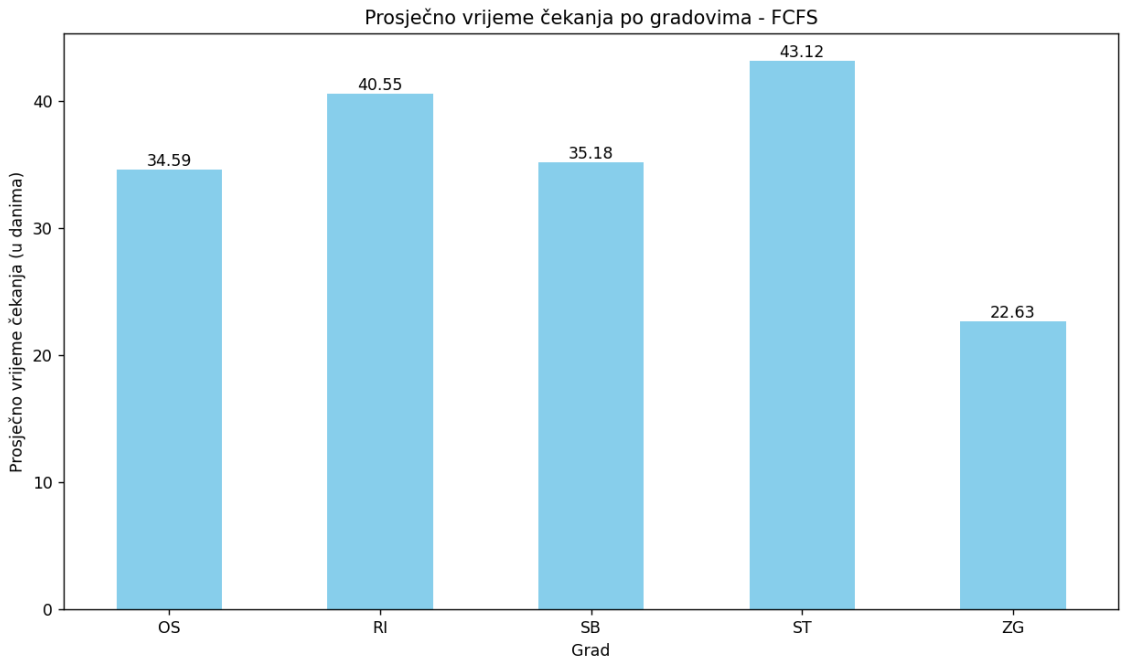
Prosječno vrijeme čekanja za sve pacijente u početnom skupu je 80,02 dana, a kako je vidljivo na slici 4.5., Slavonski Brod ima najduže prosječno vrijeme čekanja od 92,55 dana, što ukazuje na to da pacijenti u ovom gradu moraju čekati duže na svoje preglede u usporedbi s ostalim gradovima. Ovo govori da sama geografska blizina nije dovoljna kako bi se osiguralo optimalno vrijeme čekanja. Resursi u pojedinim gradovima, su preopterećeni, što rezultira velikim brojem dana čekanja za pregled. U kasnijim fazama analize, kada usporedimo naprednije metode, cilj će biti smanjiti ovo prosječno vrijeme čekanja te optimizirati alokaciju resursa kako bi se pacijentima omogućio što raniji termin.

Analizirati ćemo učinkovitost tri različita algoritma za raspodjelu pacijenata na bolničke preglede: *First-Come, First-Served* (FCFS), *Priority Scheduling* (PS) i cjelobrojno linearno programiranje (ILP). Ovi algoritmi predstavljaju različite pristupe rješavanju problema raspodjele ograničenih resursa (termina za preglede) među pacijentima s različitim potrebama i preferencijama. Cilj evaluacije ovih algoritama je usporediti njihove performanse u stvarnim uvjetima.

Na ukupnom broju pacijenata, analizirali smo uspješnost pronalaska ranijeg termina, računali prosječno vrijeme čekanja za sve pacijente (one koji su prihvatili novi termin i one koji su ostali na zadanom) te dobili slijedeće rezultate prikazane na slikama 4.6. – 4.11.

```
FCFS Evaluacija:
Ukupni broj pacijenata: 250
Broj pacijenata koji su prihvatili raniji termin: 138
Broj pacijenata koji su ostali na originalnom terminu: 112
Uspješnost pronalaska ranijeg termina: 55.20%
Prosječno vrijeme čekanja za sve pacijente: 35.48 dana
Broj iskorištenih termina: 138
Broj neiskorištenih termina: 12
```

Slika 4.6. Rezultati evaluacije FCFS algoritma.

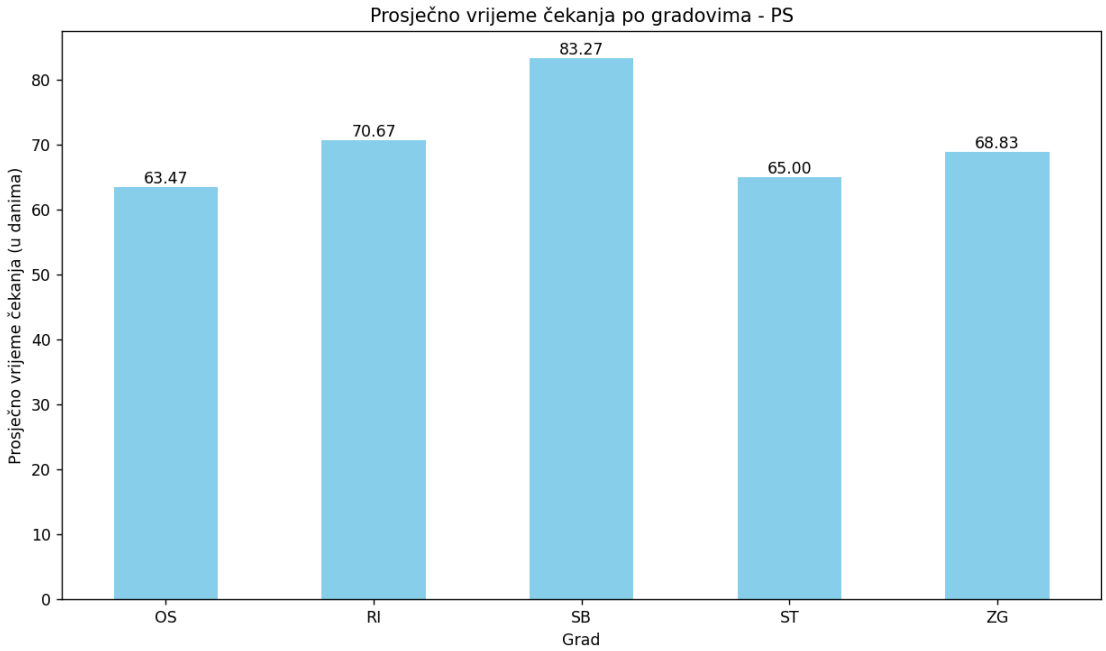


Slika 4.7. Grafički prikaz prosječnog vremena čekanja po gradovima za FCFS algoritma.

Algoritam FCFS raspodjeljuje pacijente prema redoslijedu zakazanih datuma te ne uzima u obzir varijable poput hitnosti (prioriteta). Ova metoda postiže uspješnost od 55,20%, što ukazuje na to da je gotovo polovina pacijenata uspješno raspoređena na ranije termine. Prosječno vrijeme čekanja za sve pacijente iznosilo je 35,48 dana, a u gradovima poput Splita, pacijenti su čekali duže u usporedbi s drugim gradovima.

```
PS Evaluacija:
Ukupni broj pacijenata: 250
Broj pacijenata koji su prihvatili raniji termin: 147
Broj pacijenata koji su ostali na originalnom terminu: 103
Uspješnost pronalaska ranijeg termina: 58.80%
Prosječno vrijeme čekanja za sve pacijente: 70.48 dana
Broj iskorištenih termina: 250
Broj neiskorištenih termina: 3
```

Slika 4.8. Rezultati evaluacije PS algoritma.

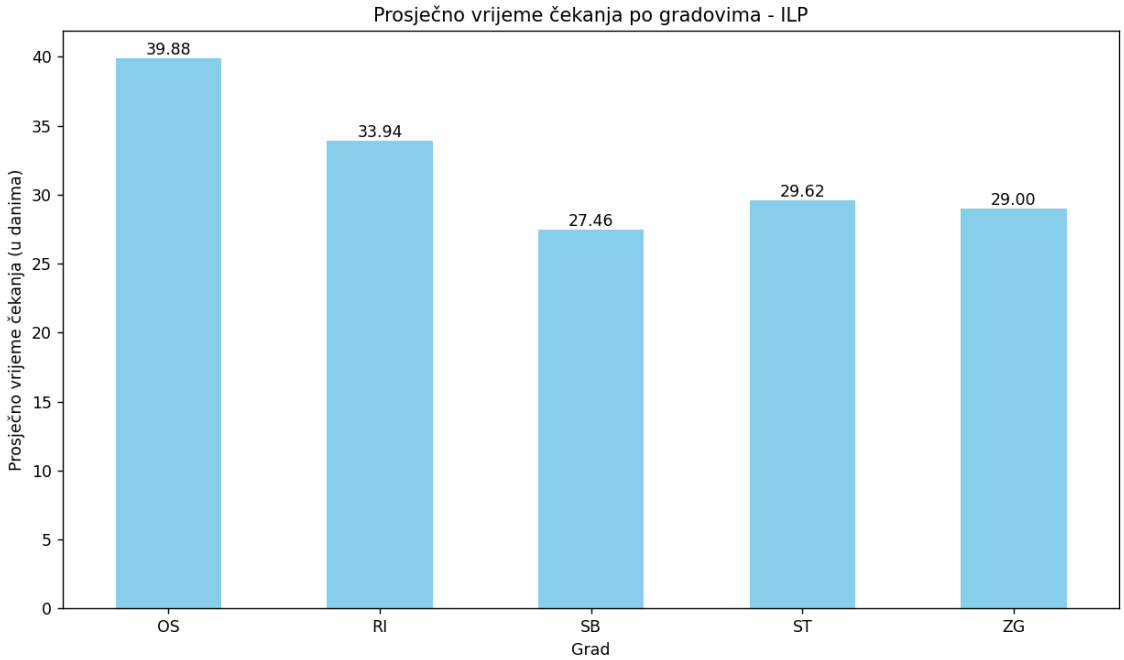


Slika 4.9. Grafički prikaz prosječnog vremena čekanja po gradovima za PS algoritma.

PS algoritam, koji prioritizira pacijente s višim prioritetom, imao je nešto veću uspješnost (58,8%) u pronalasku ranijih termina, ali je ukupno prosječno vrijeme čekanja bilo znatno duže (70,48 dana). To sugerira da iako PS algoritam preferira pacijente s hitnijim potrebama, može produžiti ukupno vrijeme čekanja za druge pacijente.

```
ILP Evaluacija:
Ukupni broj pacijenata: 250
Broj pacijenata koji su prihvatili raniji termin: 145
Broj pacijenata koji su ostali na originalnom terminu: 105
Uspješnost pronalaska ranijeg termina: 58.00%
Prosječno vrijeme čekanja za sve pacijente: 31.64 dana
Broj iskorištenih termina: 145
Broj neiskorištenih termina: 5
```

Slika 4.10. Rezultati evaluacije ILP metode.

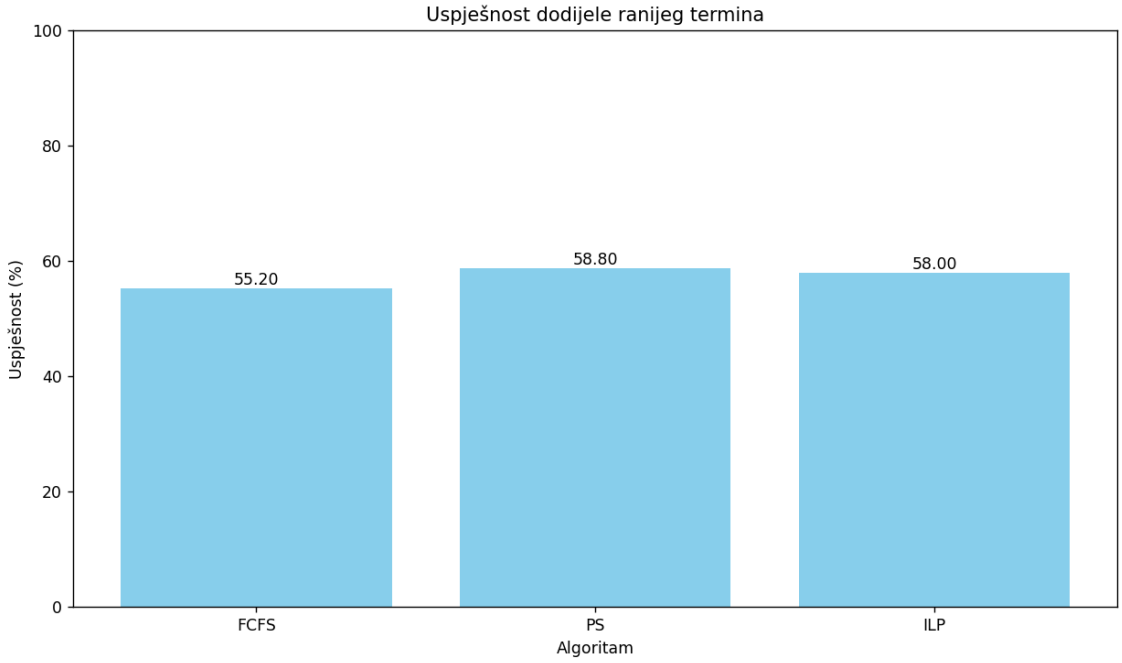


Slika 4.11. Grafički prikaz prosječnog vremena čekanja po gradovima za ILP.

ILP algoritam, koji optimizira raspored koristeći matematičko modeliranje, pokazao se najefikasnijim. Imao je 58% uspješnost rasporeda ranijih termina, ali je ujedno smanjio prosječno vrijeme čekanja na najniže vrijednosti među analiziranim algoritmima (31,64 dana). S obzirom na to, ILP se pokazao kao najučinkovitiji u optimiziranju uspješnosti pronalaska ranijih termina i smanjenja ukupnog vremena čekanja za sve pacijente, dok je FCFS jednostavniji, ali manje efikasan u globalnoj optimizaciji. PS algoritam nudi dobru raspodjelu prioriteta, ali uz visoku cijenu u smislu dužeg čekanja.

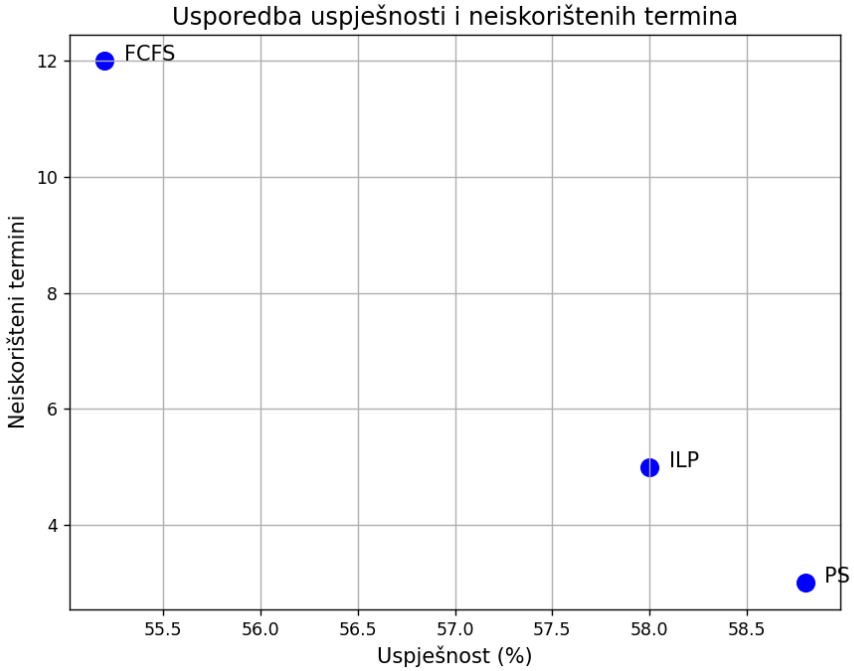
Tablica 4.1. Usporedba rezultata različitih algoritama raspoređivanja pacijenata.

Algoritam	Ukupan broj pacijenata	Prihvatili raniji termin	Ostali na originalnom terminu	Uspješnost (%)	Neiskorišteni termini
FCFS	250	138	112	55.2	12
PS	250	147	103	58.8	3
ILP	250	145	105	58	5



Slika 4.12. Usporedba uspješnosti raspodijele algoritama.

Kako je vidljivo iz tablice 4.1. i slike 4.12., algoritam PS ima najveću uspješnost dodjele ranijeg termina od 58,80%, a ILP slijedi s vrlo sličnim postotkom uspješnosti od 58,00%. Ovi rezultati pokazuju da PS i ILP algoritmi imaju gotovo jednaku sposobnost dodjele ranijih termina pacijentima, dok FCFS algoritam, iako još uvijek uspješan, zaostaje u odnosu na ostala dva algoritma.

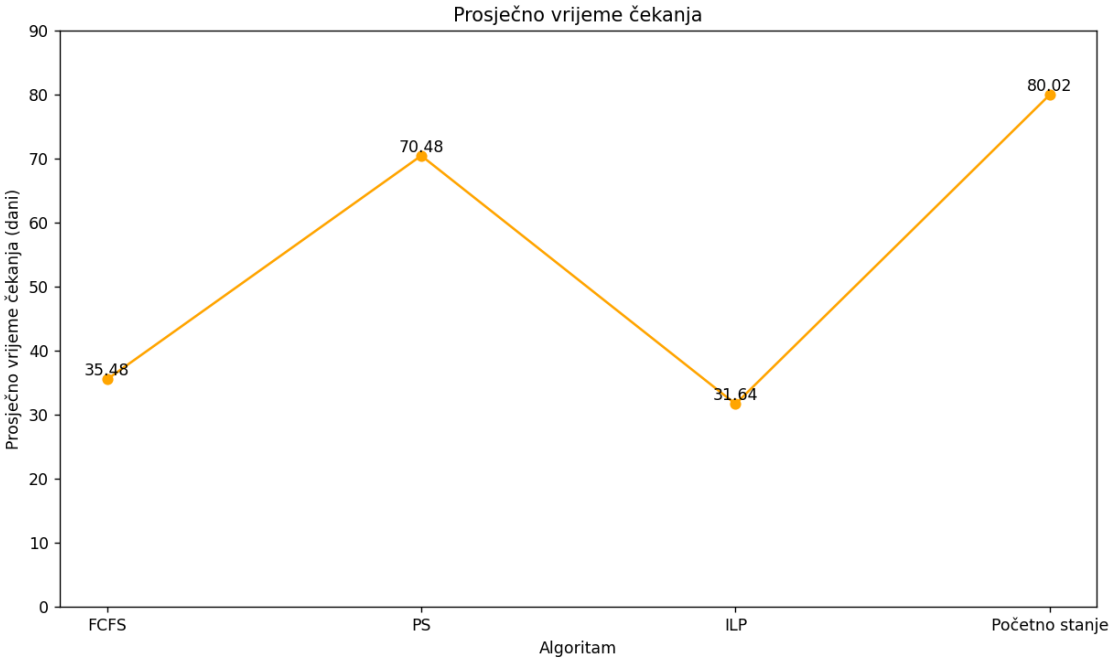


Slika 4.13. Usporedba uspješnosti i neiskorištenih termina.

Graf jasno pokazuje distribuciju učinkovitosti algoritama s obzirom na uspješnost dodjele ranijih termina i korištenje dostupnih termina (Slika 4.13.). FCFS algoritam pokazuje najnižu uspješnost (55,2%) i najveći broj neiskorištenih termina (12), što ukazuje na lošiju raspodjelu i neučinkovito korištenje resursa. PS algoritam, s druge strane, postiže najvišu uspješnost (58,8%) i najmanji broj neiskorištenih termina (3), što sugerira da je učinkovitiji u raspodjeli termina, no ne donosi značajno smanjenje prosječnog vremena čekanja. ILP algoritam se nalazi između ova dva, s uspješnošću od 58% i 5 neiskorištenih termina, pokazujući ravnotežu između dobre raspodjele i umjerenog korištenja dostupnih resursa.

Tablica 4.2. Usporedba smanjenja vremena čekanja za različite algoritme.

Algoritam	Prosječno vrijeme čekanja (dana)	Smanjenje liste čekanja u danima	Postotno poboljšanje	Neiskorišteni termini
Početno stanje	80.02			
FCFS	35.48	44.54	55.66%	12
PS	70.48	9.54	11.92%	3
ILP	31.64	48.38	60.46%	5

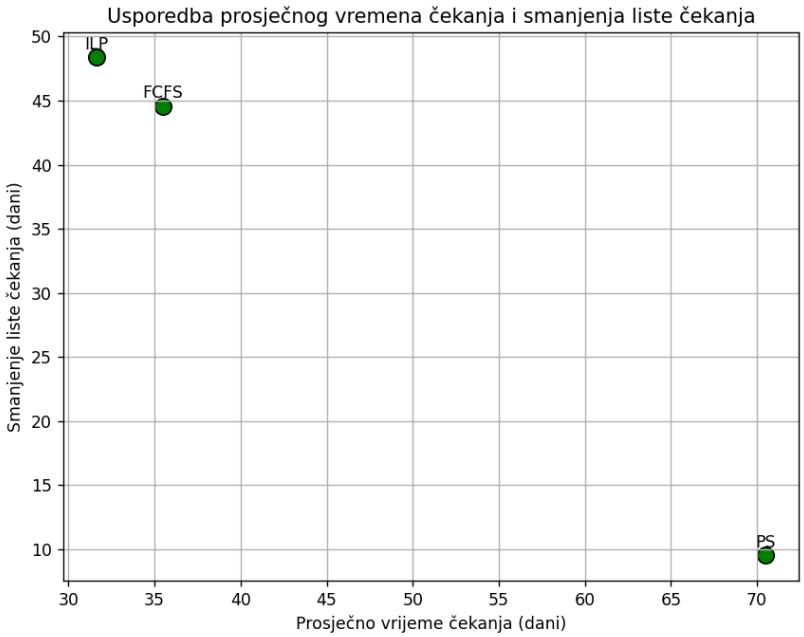


Slika 4.14. Usporedba prosječnog vremena čekanja algoritama.

Graf sa slike 4.14. i tablica 4.2. jasno pokazuju prednosti i slabosti svakog pristupa. U odnosu na početno stanje, FCFS smanjio je vrijeme čekanja za 55,66%, PS za 11,92%, a ILP je postigao najveće smanjenje za čak 60,46%. Postotno smanjenje računali smo izrazom (4-1).

$$Postotno\ smanjenje = \frac{Početno\ vrijeme\ čekanja - Prosječno\ vrijeme\ čekanja}{Početno\ vrijeme\ čekanja} \times 100 \qquad (4\text{-}1)$$

Primjena PS algoritma donosi samo marginalno poboljšanje u odnosu na početno stanje, sa smanjenjem na 70,48 dana, što pokazuje da ovaj algoritam nije efikasan u smanjenju vremena čekanja za pacijente u ovom scenariju. S druge strane, algoritam FCFS značajno smanjuje prosječno vrijeme čekanja na 35,48 dana te tako postiže zadovoljavajuću učinkovitost u raspoređivanju pacijenata, a najbolje rezultate postiže ILP, s prosječnim vremenom čekanja od samo 31,64 dana.



Slika 4.15. Usporedba prosječnog vremena čekanja i smanjenja liste čekanja.

Dakle, iz dosadašnjih prikaza, ILP algoritma donosi najbolje rezultate pružajući balans između uspješnosti raspodjele pacijenata na nove termine i smanjenju vremena čekanja, čineći ga najučinkovitijim pristupom u ovim uvjetima. PS algoritam, iako najefikasniji u iskorištavanju termina, ima znatno dulje vrijeme čekanja pa zato dobiva najlošiju ocjenu.

4.3. Rasprava

Uz usporedbu prosječnog vremena čekanja i uspješnosti algoritama, važno je istaknuti i koliko je pacijenata dobilo raniji termin kroz svaki algoritam.

PS algoritam je dodijelio raniji termin za 147 pacijenata, što je najveći broj u odnosu na ostale algoritme. Međutim, ovo dolazi uz cijenu značajno većeg prosječnog vremena čekanja od 70,48 dana, što sugerira da PS algoritam, iako uspješan u pronalaženju ranijih termina za veći broj pacijenata, nije najučinkovitiji u smanjenju čekanja u odnosu na početno stanje.

S druge strane, ILP algoritam je raniji termin osigurao za 145 pacijenata, što je vrlo blizu PS algoritmu, ali je pri tome postigao znatno bolje rezultate u smanjenju prosječnog vremena čekanja, koje iznosi 31,64 dana, što ga čini najuravnoteženijim izborom.

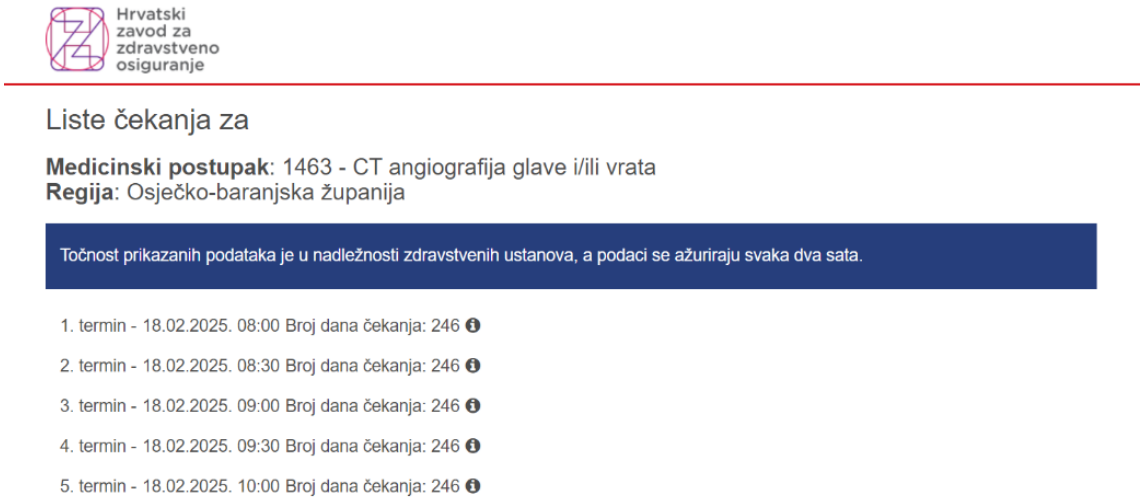
FCFS algoritam je osigurao ranije termine za 138 pacijenata, što je najmanji broj u usporedbi s ostalim algoritmima, ali usprkos tome i nešto nižoj uspješnost raspoređivanja od 55,20%, prosječno vrijeme čekanja je 35.48 dana, što ga čini solidnim izborom.

U konačnici, ILP algoritam nudi najbolji omjer broja pacijenata s ranijim terminom i kraćeg prosječnog vremena čekanja, što ga čini najefikasnijim za optimizaciju raspodjele termina.

Implementacija boljih algoritama za raspodjelu pacijenata može značajno poboljšati učinkovitost zdravstvenog sustava, smanjiti liste čekanja i osigurati pravovremenu skrb za pacijente. U procesu testiranja uzeli smo u obzir nekoliko ključnih aspekata koji utječu na raspodjelu pacijenata – toleranciju pacijenata oko terminima pregleda, maksimalna udaljenost koju su spremni putovati, te njihov prioritet.

Hitni slučajevi u medicini uvijek imaju prioritet nad ostalim pacijentima, bez obzira na redoslijed dolaska. Ova praksa je uobičajena u bolnicama kako bi se osiguralo da pacijenti s najkritičnijim stanjima dobiju brzu i adekvatnu medicinsku skrb, to znači da pacijenti s ozbiljnim ozljedama, srčanim udarima i drugim hitnim stanjima imaju prednost pred onima s manje hitnim potrebama. Dakle, iako je implementacijom PS algoritma, na nasumično generiranim podatcima, ostvareno dugačko prosječno vrijeme čekanja na pregled, stvarne bolnice moraju biti fleksibilnije i sposobne prilagoditi se dinamičkim i često nepredvidivim situacijama koje zahtijevaju hitnu intervenciju.

Osim toga, racionalno gospodarenje financijskim resursima u zdravstvu ključan je čimbenik za izgradnju učinkovitog sustava. Stoga, promotrimo slijedeći primjer.



Slika 4.16. Prvi slobodni termini za CT u Osijeku.

Slika 4.17. Prvi slobodni termini za CT u Zagrebu.

Iz slike 4.16. vidimo da je u Osječko-baranjskoj županiji prvi dostupni termin za CT angiografiju glave i/ili vrata je tek 18. veljače 2025. godine, s brojem dana čekanja od 246 dana, što ukazuje na izrazito dugu listu čekanja. S druge strane, u Zagrebačkoj županiji i Gradu Zagrebu, prvi dostupni termin je već 18. lipnja 2024. godine, s brojem dana čekanja od samo jednog dana (Slika 4.17.). U Zagrebu je očigledno dostupno više termina u kraćem vremenskom periodu, što ukazuje na bolju organiziranost i raspoloživost resursa za ovaj pregled u usporedbi s Osijekom. HZZO nudi naknadu za troškove prijevoza ukoliko je udaljenost veća od 50 km, što bi moglo potaknuti pacijente iz Osijeka da koriste dostupne termine u Zagrebu, smanjujući tako pritisak na lokalne resurse. Promotrimo slijedeći primjer – put od Osijeka do Zagreba iznosi oko 283 km te bi troškovi puta iznosili oko 100 eura. Ukoliko se 4.000 pacijenata odluči za putovanje u Zagreb mogli bi generirati dovoljno sredstava za kupnju novog CT uređaja u Osijeku koji košta oko 400.000 eura [24].

Dodatna mjera za rasterećenje javnog zdravstvenog sustava je činjenica da se mnogi pacijenti odlučuju za privatne bolnice kako bi izbjegli duge liste čekanja. Ovaj trend može značajno smanjiti pritisak na javne zdravstvene ustanove i omogućiti brži pristup pregledima i liječenju. Međutim, ova opcija nije u skladu sa svačijim financijskim mogućnostima, jer privatne zdravstvene usluge često dolaze s visokim troškovima koje si mnogi pacijenti ne mogu priuštiti. Oslanjanje na privatne bolnice može dodatno produbiti nejednakosti u pristupu zdravstvenoj skrbi, jer pacijenti s nižim prihodima ostaju ovisni o preopterećenim javnim ustanovama. Stoga, iako privatne bolnice mogu

kratkoročno smanjiti liste čekanja, dugoročno je nužno osigurati adekvatno financiranje i učinkovitost javnog zdravstvenog sustava kako bi se zadovoljile potrebe svih pacijenata, bez obzira na njihove financijske mogućnosti.

5. ZAKLJUČAK

U ovom radu istražili smo problem optimizacije rasporeda pacijenata za medicinske preglede primjenom triju različitih algoritama: *First-Come First-Served* (FCFS), *Priority Scheduling* (PS) i cjelobrojno linearno programiranje (ILP). Cilj istraživanja bio je smanjiti vrijeme čekanja i povećati učinkovitost korištenja dostupnih resursa u zdravstvenom sustavu.

Rezultati su pokazali da je ILP algoritam bio najučinkovitiji, ostvarivši poboljšanje od 60,46%, dok su FCFS i PS algoritmi ostvarili nešto niže performanse. Početno prosječno vrijeme čekanja, 80,02 dana, ukazivalo je na potrebu za daljnjim prilagodbama, a implementacijom algoritama došli smo do relativno optimalnih ishoda.

Financijski aspekti također imaju ključnu ulogu u optimizaciji zdravstvenog sustava. HZZO nudi naknadu za troškove prijevoza pacijentima koji putuju više od 50 km do mjesta pregleda, što može pomoći u smanjenju opterećenja na lokalne zdravstvene ustanove. Osim toga, mnogi se pacijenti odlučuju za privatne zdravstvene usluge kako bi izbjegli duge liste čekanja, no takve usluge nisu svima financijski dostupne, što dodatno opterećuje zdravstveni sustav. Racionalno gospodarenje financijskim resursima u zdravstvu ključno je za izgradnju visoko učinkovitog zdravstvenog sustava koji postiže optimalne ishode u smislu dostupnosti, pravednosti i učinkovitosti.

Zaključno, iako su rezultati pokazali napredak u optimizaciji rasporeda pacijenata, potrebno je daljnje istraživanje za poboljšanje algoritama. Buduća istraživanja mogla bi uključivati razvoj hibridnih modela koji kombiniraju prednosti različitih pristupa te integraciju dodatnih kriterija.

LITERATURA

[1] Kocijan, I. (2023) Efikasnost zdravstvenih usluga u Republici Hrvatskoj: percepcija korisnika. Diplomski rad. Sveučilište u Zagrebu, Fakultet organizacije i informatike. Dostupno na: https://urn.nsk.hr/urn:nbn:hr:211:852705 [Pristupljeno: 23. lipnja 2024.]

[2] Silva-Aravena, F., Álvarez-Miranda, E., Astudillo, C.A., González-Martínez, L. i Ledezma, J.G. (2021) 'Patients' Prioritization on Surgical Waiting Lists: A Decision Support System', Mathematics. Dostupno na: https://doi.org/10.3390/math9101097 [Pristupljeno: 23. lipnja 2024.]

[3] Abdalkareem, Z.A., Amir, A., Al-Betar, M.A., Ekhan, P. i Hammouri, A.I. (2021) 'Healthcare scheduling in optimization context: a review', Health and Technology. Dostupno na: https://doi.org/10.1007/s12553-021-00547-5 [Pristupljeno: 23. lipnja 2024.]

[4] Dimakou, S., Dimakou, O. i Basso, H.S. (2015) 'Waiting time distribution in public health care: empirics and theory', Health Economics Review, 5(25). Dostupno na: https://doi.org/10.1186/s13561-015-0061-7 [Pristupljeno: 23. lipnja 2024.]

[5] Zhang, L., Chang, H. i Xu, R. (2012) 'The Patient Admission Scheduling of an Ophthalmic Hospital Using Genetic Algorithm', Proceedings of the 2012 2nd International Conference on Computer and Information Application (ICCIA 2012). Dostupno na: https://www.researchgate.net/publication/266645810_The_Patient_Admission_Scheduling_of_an_Ophthalmic_Hospital_Using_Genetic_Algorithm [Pristupljeno: 23. lipnja 2024.]

[6] Hartmann Tolić, I., 2020. Optimiranje voznog reda mreže javnog prijevoza s ciljem smanjenja vremena čekanja putnika i povećanja popunjenosti vozila. Doktorska disertacija, Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek. Dostupno na: https://urn.nsk.hr/urn:nbn:hr:200:046600 [Pristupljeno: 23. lipnja 2024.]

[7] Index.hr, 2023. EU stisnula Beroša da smanji liste čekanja, on sve prebacio na ravnatelje bolnica. Dostupno na: https://www.index.hr/vijesti/clanak/eu-stisnula-berosa-da-smanji-liste-cekanja-on-sve-prebacio-na-ravnatelje-bolnica/2511448.aspx [Pristupljeno: 23. lipnja 2024.]

[8] CEZIH, 2024. Liste čekanja. Dostupno na: https://liste.cezih.hr [Pristupljeno: 17. lipnja 2024.]

[9] NHS England, 2023. NHS cuts longest waits and speeds up response times. Dostupno na: https://www.england.nhs.uk/2023/05/nhs-cuts-longest-waits-and-speeds-up-response-times/ [Pristupljeno: 23. lipnja 2024.]

[10] Nordics Info, 2024. Healthcare in the Nordic Region. Dostupno na: https://nordics.info/show/artikel/healthcare-in-the-nordic-region/ [Pristupljeno: 23. lipnja 2024.]

[11] OECD, 2023. Health at a Glance 2021: OECD Indicators. Dostupno na: https://www.oecd-ilibrary.org/social-issues-migration-health/waiting-times-for-health-services_242e3c8c-en [Pristupljeno: 23. lipnja 2024.]

[12] Microsoft, 2024. Visual Studio. Dostupno na: https://visualstudio.microsoft.com [Pristupljeno: 23. lipnja 2024.].

[13] LearnPython.com, 2024. Python Programming: Advantages and Disadvantages. Dostupno na:https://learnpython.com/blog/python-programming-advantages-disadvantages/ [Pristupljeno: 23. lipnja 2024.]

[14] Inoxoft, 2024. Top 23 Applications Made with Python. Dostupno na: https://inoxoft.com/blog/top-23-applications-made-with-python/ [Pristupljeno: 23. lipnja 2024.]

[15] Tportal.hr, 2023. Rekordan broj pregleda na listama čekanja: Na ovo se najduže čeka. Dostupno na: https://www.tportal.hr/vijesti/clanak/rekordan-broj-pregleda-na-listama-cekanja-na-ovo-se-najduze-ceka-20230411 [Pristupljeno: 23. lipnja 2024.]

[16] Educative.io, 2024. First-Come, First-Served (FCFS) Scheduling Algorithm. Dostupno na: https://www.educative.io/answers/first-come-first-served-fcfs-scheduling-algorithm [Pristupljeno: 23. lipnja 2024.]

[17] GeeksforGeeks, 2024. Program for FCFS CPU Scheduling | Set 1. Dostupno na: https://www.geeksforgeeks.org/program-for-fcfs-cpu-scheduling-set-1/ [Pristupljeno: 23. lipnja 2024.]

[18] Scaler, 2024. Priority Scheduling Algorithm. Dostupno na: https://www.scaler.com/topics/operating-system/priority-scheduling-algorithm/ [Pristupljeno: 23. lipnja 2024.]

[19] PrepInsta, 2024. Priority Scheduling Algorithm. Dostupno na: https://prepinsta.com/operating-systems/priority-scheduling-algorithm/ [Pristupljeno: 23. lipnja 2024.]

[20] Yinusa, A. i Faezipour, M., 2023. Optimizing Healthcare Delivery: A Model for Staffing, Patient Assignment, and Resource Allocation. Applied System Innovation. Dostupno na: https://doi.org/10.3390/asi6050078 [Pristupljeno: 23. lipnja 2024.]

[21] Mitchell, S., O'Sullivan, M. i Dunning, I., 2011. PuLP: A Linear Programming Toolkit for Python. Stuart Mitchell Consulting. Dostupno na: http://www.coin-or.org/PuLP/ [Pristupljeno: 23. lipnja 2024.]

[22] Kovač, J., 2022. Linearno programiranje s Pythonom. Dostupno na: https://www.josipkovac.xyz/posts/2022-12-04-linearno-programiranje-python/ [Pristupljeno: 23. lipnja 2024.]

[23] Keen, B.A., 2023. Linear Programming with Python and PuLP - Part 5. Dostupno na: https://benalexkeen.com/linear-programming-with-python-and-pulp-part-5/ [Pristupljeno: 23. lipnja 2024.]

[24] Slobodna Dalmacija, 2023. Bolnički CT plaćen 4 milijuna kn, a privatnika dva puta bolji uređaj dođe 3 milijuna. Dostupno na: https://slobodnadalmacija.hr/vijesti/hrvatska/bolnicki-ct-placen-4-milijuna-kn-a-privatnika-dva-puta-bolji-uredaj-dode-3-milijuna-310120 [Pristupljeno: 23. lipnja 2024.]

SAŽETAK

Ovaj rad bavi se učinkovitim raspoređivanjem pacijenata na bolničke preglede koristeći tri različita algoritma: *First-Come First-Served*, *Priority Scheduling* i cjelobrojno linearno programiranje. Cilj istraživanja bio je smanjiti vrijeme čekanja pacijenata i poboljšati učinkovitost korištenja zdravstvenih resursa. Rezultati su pokazali da je ILP algoritam ostvario najbolje performanse, smanjivši vrijeme čekanja za 60,46%, dok su FCFS i PS algoritmi postigli smanjenje od 55,66% i 11,92%.

U istraživanju smo koristili tehnologije poput programskog jezika Python i biblioteka kao što su Pandas za manipulaciju podacima te PuLP za rješavanje problema linearnog programiranja. Ovi alati omogućili su nam učinkovito modeliranje i analizu problema raspodjele pacijenata, naglašavajući važnost naprednih algoritama u optimizaciji zdravstvenih sustava.

Ključne riječi: algoritmi, optimizacija, pacijenti, raspored, zdravstvo

ABSTRACT

HOSPITAL EXAMINATION PATIENT SCHEDULING EFFICIENCY

This paper addresses the efficient scheduling of patients for hospital examinations using three different algorithms: First-Come First-Served, Priority Scheduling, and Integer Linear Programming. The aim of the research was to reduce patient waiting times and improve the efficiency of healthcare resource utilization. The results showed that the ILP algorithm achieved the best performance, reducing waiting time by 60.46%, while the FCFS and PS algorithms achieved reductions of 55.66% and 11.92%.

In this research, we utilized technologies such as the Python programming language and libraries like Pandas for data manipulation and PuLP for solving linear programming problems. These tools allowed us to effectively model and analyze the patient scheduling problem, highlighting the importance of advanced algorithms in optimizing healthcare systems.

Keywords: algorithms, healthcare, optimization, patients, scheduling

ŽIVOTOPIS

Lea Šobačić rođena je 12. rujna 2001. godine u Osijeku, gdje je započela svoje osnovno obrazovanje u Osnovnoj školi „Dobriša Cesarić“. Nakon završetka osnovne škole, nastavila je srednjoškolsko obrazovanje u I. Gimnaziji Osijek, gdje je razvila interes za tehničke znanosti i računarstvo. Nakon uspješnog završetka gimnazije, upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku 2020. godine, gdje nastavlja svoj studij na preddiplomskom sveučilišnom studiju računarstva.

_____

Potpis autora

PRILOZI

P.3.1. Tablica udaljenosti između gradova (Dostupno na: https://www.relago.hr/RelaGO/BlogPost?id=77&nameForDisplay=Tablica-udaljenosti-medju-30-najvecih-naselja-u-Hrvatskoj)