Optimiranje odabira i rasporeda zaštitnih elemenata u projektiranju sustava tehničke zaštite.

Čakija, Dejan

Sveučilište u Zagrebu
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Dejan Čakija

OPTIMIRANJE ODABIRA I RASPOREDA ZAŠTITNIH ELEMENATA U PROJEKTIRANJU SUSTAVA TEHNIČKE ZAŠTITE

DOKTORSKI RAD

Zagreb, 2020.

Sveučilište u Zagrebu
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DEJAN ČAKIJA

OPTIMIRANJE ODABIRA I RASPOREDA ZAŠTITNIH ELEMENATA U PROJEKTIRANJU SUSTAVA TEHNIČKE ZAŠTITE

DOKTORSKI RAD

Mentor:
prof. dr. sc. Željko Ban

Zagreb, 2020.

University of Zagreb
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

Dejan Čakija

OPTIMIZATION OF SECURITY ELEMENTS SELECTION AND PLACEMENT IN PHYSICAL SECURITY SYSTEM

DOCTORAL THESIS

Supervisor:
Professor Željko Ban, PhD

Zagreb, 2020

Doktorski rad izrađen je na Sveučilištu u Zagrebu Fakultetu elektrotehnike i računarstva, na Zavodu za automatiku i računalno inženjerstvo.

Mentor: prof. dr. sc. Željko Ban

Doktorski rad ima 114 stranica.

Doktorski rad br.: _____

O mentoru

Prof. dr. sc. Željko Ban

Željko Ban rođen je 13. 09. 1962. godine u Prugovcu (općina Đurđevac, Republika Hrvatska), narodnost Hrvat, državljanstvo Hrvatsko.

Elektrotehnički fakultet Sveučilišta u Zagrebu upisao je 1981. godine, da bi 1985. diplomirao, a 1991. magistrirao na smjeru Automatika. Godine 1999. obranio je doktorsku disertaciju na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu. Tijekom redovnog i postdiplomskog studija dobio je više priznanja te brončanu i srebrnu plaketu "Josip Lončar".

Nakon diplomiranja radio je godinu dana u Elektrotehničkom institutu "Rade Končar" u Zagrebu na razvoju i ispitivanju algoritama za mikroprocesorsko upravljanje istosmjernim pogonima. Od 1988. godine zaposlen je kao asistent, od 2001. godine kao docent, a od 2006. godine kao izvanredni profesor te od 2014. godine kao redoviti profesor na zavodu za Automatiku i računalno inženjerstvo Fakulteta elektrotehnike i računarstva u Zagrebu, na grupi predmeta Automatsko upravljanje sustavima. Kao asistent je sudjelovao je u izvođenju nastave na predmetima: Automatsko upravljanje, Automatsko upravljanje proizvodnim procesima, Analogna i hibridna tehnika, Elementi automatike, Industrijski roboti i fleksibilni proizvodni procesi i Modeliranje i simuliranje procesa. Od izbora za docenta do danas vodi predmete Modeliranje i simuliranje procesa te izborne predmete Optimiranje parametara sustava i Optimiranje i primjena adaptivnih regulatora te sudjeluje u nastavi predmeta Automatizacija postrojenja i procesa, Modeliranje i simuliranje sustava, Alarmni sustavi te Adaptivno i robusno upravljanje. Na Tehničkom fakultetu Sveučilišta u Rijeci vodio je predmet Modeliranje i simulacija sustava te sudjelovao u nastavi predmeta Elementi automatizacije postrojenja i Automatizacija postrojenja i procesa. Na Studiju energetske učinkovitosti i obnovljivih izvora energije Sveučilišta u Zagrebu – Šibenik organizirao je i izvodio nastavu iz predmeta Automatsko upravljanje, Sustavi za pohranu energije i Vještine – Matlab. Na postdiplomskom studiju vodio je predmet Adaptivno upravljanje primjenom referentnog modela. Pod njegovim vodstvom obranjeno je 60 diplomskih i završnih radova jedan magistarski rad i tri doktorske disertacije.

Tijekom rada na fakultetu sudjelovao je kao suradnik u znanstveno istraživačkim projektima financiranim od Ministarstva znanosti i tehnologije republike Hrvatske i to na

projektima Adaptivno i optimalno upravljanje elektromotornim pogonima te Inteligentno i adaptivno upravljanje sustavima. Nakon toga vodio je znanstvene projekte financirane od Ministarstva znanosti obrazovanja i športa Republike Hrvatske i to 2001. godine istraživački projekt pod nazivom Inteligentno i adaptivno upravljanje sustavima, a od 2006. godine projekt Sustav upravljanja energetskog izvora s kogeneracijom na osnovi gorivnih članaka. Osim toga vodio je IRI projekt „Razvoj nove generacije industrijskih modularnih, redundantnih, višeizlaznih sustava neprekidnog napajanja istosmjernim i izmjeničnim naponima“ te projekt Izgradnja prototipa punionice za električne bicikle financiran od Zagrebačkog elektrotehničkog poduzeća, a sudjelovao je i na drugim FP7 i IRI projektima te projektima financiranim od HRZZ-a na poslovima vezanim uz identifikaciju, modeliranje i simuliranje sustava automatskog upravljanja u industrijskim procesima.

Služi se engleskim jezikom.

About supervisor

Professor Ban Željko, PhD

Željko Ban was born on September 13, 1962 in Prugovac (Đurđevac municipality, Republic of Croatia), nationality Croat, Croatian citizenship.

He enrolled in the Faculty of Electrical Engineering at the University of Zagreb in 1981, graduating in 1985 and master's degree in Control theory in 1991. In 1999 he defended his doctoral thesis at the Faculty of Electrical Engineering and Computing, University of Zagreb. During his graduate and postgraduate studies, he received several awards and a bronze and silver plaque "Josip Lončar".

After graduation, he was employed at the Rade Končar Electrical Engineering Institute in Zagreb for a year, developing and testing algorithms for microprocessor control of DC drives. Since 1988 he has been employed as an assistant at the Department of Automation and Computer Engineering at the Faculty of Electrical Engineering and Computing in Zagreb, in the Control systems group. He was with the same institution as assistant professor, associate professor and full professor since 2001, 2006 and 2014, respectively.

As an assistant, he has been involved in the following courses: Automatic Control, Automatic Control of the Manufacturing Processes, Analog and Hybrid Technique, Elements of Control Systems, Industrial Robots and Flexible Manufacturing Processes, and Process Modeling and Simulation.

After obtaining the position of the assistant professor he has been lecturer of the courses Modeling and Simulation of Processes, Optimizing System Parameters and Optimizing and Applying Adaptive Controllers.

At the Faculty of Engineering, University of Rijeka, he was lecturer of the course Modeling and Simulation of Systems and took part in the lectures of course Elements of Plant Automation and Plant and Process Automation. At the Study of Energy Efficiency and Renewable Energy Sources of the University of Zagreb - Šibenik he organized and was lecturer of the courses in the fields of Automatic Control, Energy Storage Systems and Skills - Matlab. At the postgraduate level, he taught the course Reference Model Adaptive Control. Under his leadership, 60 graduation and bachelor thesis were defended, one master's thesis and three doctoral theses.

During his work at the Faculty, he participated as a researcher in scientific research projects financed by the Ministry of Science and Technology of the Republic of Croatia on the projects Adaptive and Optimal Control of Electric Motor Drives and Intelligent and Adaptive System Control.

In addition, he led scientific projects funded by the Ministry of Science and Education of the Republic of Croatia named Intelligent and Adaptive Systems Control in 2001, and since 2006 the project Energy Management System with Cogeneration Based on Fuel Cells. In addition, he led the IRI project "Development of the next generation of industrial modular, redundant, multi-output DC and AC power systems" and the project of building a prototype of a charging station for electric bicycles, funded by the Zagreb Electrical Engineering Company. Besides, he participated as researcher in other FP7 and IRI projects and projects funded by HRZZ on tasks related to the identification, modeling and simulation of automatic control systems in industrial processes.

He is fluent in English.

Supruzi Andreji.

Sažetak

U procesu projektiranja sustava tehničke zaštite, na osnovu definiranih potencijalnih ciljeva napada i očekivanih tipova napadača, projektant odabire zaštitne elemente, određuje njihovu količinu i raspored u prostoru. Elementima zaštite želi se čim ranije detektirati napadača, a zatim ga usporiti u daljnjem kretanju prema cilju, kako bi ga tjelesna zaštita presrela i zaustavila napad. Cilj projektiranja je postići minimalno ulaganje u sustav uz postizanje dovoljne razine zaštite. Ručni odabir zaštitnih elemenata i subjektivna procjena postignute razine zaštite glavni su nedostaci današnjeg pristupa projektiranju sustava zaštite.

Uvedena je metoda kojom se automatizirano provodi odabir i raspored zaštitnih elemenata u prostoru. Osnova metode je model koji opisuje napadača, tjelesnu zaštitu, prostor koji se štiti i dostupne zaštitne elemente. Iz njega se izvodi model koji opisuje moguće kretanje napadača prostorom te dostupne i odabrane elemente zaštite. Nad njime se provodi optimiranje odabira i pozicije zaštitnih elemenata. Predloženo je višekriterijsko optimiranje, zasnovano na evolucijskim algoritmima. Rezultat optimiranja je skup rješenja koja projektantu nude odabir onog koje zadovoljava minimalnu vjerojatnost presretanja napadača ili maksimalnu cijenu sustava.

U proces optimiranja uvedena je hibridna metoda koja značajno ubrzava pronalazak kvalitetnih rješenja. Nad početnim grafom mogućeg kretanja napadača izvodi se algoritam koji izbacuje čvorove i pripadajuće veze koji sigurno ne mogu pripadati kritičnom putu napada. Primjenom algoritma znatno se smanjuje prostor pretraživanja. Drugi dio hibridne metode je stvaranje početne populacije jedinki, a koristeći domensko znanje, čime evolucijski algoritmi brže pronalaze kvalitetna rješenja. Drugi algoritam, nazvan D-EX2, implementiran je u proces evaluacije rješenja. Koristi znanje stečeno u ranijim evaluacijama i prema skupu pravila određuje postoji li ranije pronađeno rješenje koje predstavlja rezultat evaluacije predloženog rješenja, čime izbjegava nepotrebne izračune rješenja. Eksperimentima je dokazana učinkovitost izrađenih algoritama.

Ključne riječi:

numerička analiza ugroženosti, optimiranje sustava zaštite, genetski algoritam, model štićenog objekta podesan za optimiranje, D-EX2

Optimization of security elements selection and placement in physical security system

The protection of a facility's critical assets against theft or sabotage is commonly executed by implementing a physical protection system, usually consisting of intrusion detection, video surveillance, access control and other technical systems designed to detect and deter an adversary and trigger an appropriate physical response by security forces. In the process of designing a physical security system, the designer selects security elements, determines their quantity and arrangement. The goal of the design process is to achieve a minimum investment in the system, while achieving a sufficient level of protection. The manual selection of security elements and the subjective assessment of achieved protection levels are the main disadvantages of today's approach to the design of security systems. Although numerical threat analysis methods have been introduced in the last century, which are the basis for assessing the quality of a designed security solution, designers of physical security systems still most often use subjective methods to design these systems.

The large selection of security elements and locations to which the selected element can be placed makes it highly likely that the project designer has not chosen a design solution that is close to optimal. An additional disadvantage is the subjective assessment of the effectiveness of the designed solution. Searching for possible solutions, suggesting approximately optimal solutions, and numerically determining the achieved level of protection requires a tool that will allow a designer to design significantly faster and with better quality. With today's development of machine learning algorithms and their application in various fields, users will expect the market to contain many software applications that can perform numerical analysis of security systems and propose optimal solutions. However, few such solutions are available today and those that do exist do provide only limited functionality or do not suggest optimal solutions at the level of selection and positioning of security elements.

Therefore, the focus of this research is to develop a methodology that uses the definition of a protected facility in digital form, automates the optimization of the physical security system, and presents the security designer the set of possible solutions, where each solution provides a near-optimal level of protection in its price range. There are two main challenges in optimizing the physical protection system. The first challenge is that there are two

opposing goals of the optimization, the maximization of interruption probability coupled with the minimization of investment in the physical protection system. The second challenge is that search space is changed for each proposed solution, as security elements influence the optimal attack path, used for defining the solution quality.

This doctoral dissertation consists of five chapters. The first chapter ("1. Introduction"), presents introductory considerations and motivation.

Chapter two (" 2. Numerical methods for assessment of vulnerability to intrusion") defines methods to numerically assess the probability that an adversary will be detected and intercepted before reaching the target. The basis for numerical assessment is a probability function that an adversary would be detected and intercepted before reaching the target. The probability that an adversary will be detected during movement to the target depends on the skill of the adversary and the implemented security elements. The movements of an adversary and security forces can be described as the normal distribution so that interception probability depends on the location where an adversary is detected and can be expressed as a probability that the time it takes the adversary to reach the target, from the moment of detection, is higher than the reaction time of security forces. The adversary usually has a great number of possibilities for moving to the target. This chapter presents various methods to find the optimal path for the adversary, also known as the critical path, that gives the adversary the highest probability that his attack will be successful.

Once the numerical assessment method is defined, chapter three (“3. Objective modeling of the physical protection system”) introduces a model that enables numerical assessment by describing the adversary, the security personnel, the facility that is being protected, and the available security devices. Currently, facilities are typically designed with software solutions that define all facility details in standardized digital form, so that each element of the building and surroundings is defined as an object which has a given position in space, shape, dimensions, build material, and interdependence with other elements in a three-dimensional space. The overview of the standards that describe indoor and outdoor building architecture as a data model is given. Such a definition makes it possible to automate the transformation of a description into a format suitable for optimization. Manually describing the protected facility is time-consuming, so the methodology for automated transfer from standardized geometric models to the proposed protected object model is presented. The host element model is presented to define properties of protected facility elements, such as geometry,

materials, and neighboring elements. It also defines if the host element may be used as a starting point of an attack or presents the attacking target. The adversary model defines a sneaking level, movement abilities, and available tools. Movement abilities are dependent on available tools and movement material and define the adversary's movement speed. The security personnel model defines the security force's reaction time, neutralization capability, and communication probability. The security element model defines detection probability, blocking time, price, coverage measure, and security element placement. The next step, based on the above information, is to perform automatic geometric transformations that create a model of possible adversary movement. Different methods for creating a graph of possible attackers' movements are described. Graph nodes present elements of a facility that may be used for adversary movement and graph edges depict available paths between them. Such a representation of a protected facility and adversary may be used for automated optimization.

Chapter four ("4. Optimization of security elements selection and placement in a physical security system") presents the method that automatically optimizes the design of a physical security system. Single-criteria and multi-criteria functions are compared, and due to multiple advantages, the multi-criteria method is chosen for further elaboration. The selection and placement of security elements are optimized by evolutionary algorithms that perform stochastic optimization methods by simulating evolutionary processes in nature. The standard evolution process is extended in this research with optimizations that speed up the search for quality solutions and avoid unnecessary evaluations of proposed solutions. Hybrid optimization reduces search space and proposes initial solutions that speed up the search for quality solutions. The automatic creation of an adversary's possible movements creates a graph with nodes that represent all possible movements. However, there are nodes among them that will not belong to any optimal path of adversary's movement for any proposed protection solution. Such nodes and associated edges can be removed from such a graph without affecting the quality of the solution and removing them reduces the search space. Graph elements that are unnecessary for optimization can be found by continuously finding the critical paths and placing all available security elements to nodes that belong to found critical paths. Such an algorithm can stop searching if there are no new nodes found in the last search for the optimal path. All nodes that have not been touched may be safely removed from the search space. Optimization of the proposed model can be performed using a genetic algorithm whereby the initial population is generated by random binary vectors.

Each bit represents if a certain security element is applied at a certain node. An understanding of the search domain allows for the creation of an initial population that gives the algorithm a greater probability of finding near-optimal Pareto front solutions for a given number of iterations. The placement of security elements affects their effectiveness. A detection element positioned closer to the start of the attack has greater efficiency than positioned closer to the attack target. Placing a blocking element closer to the target of an attack has a greater impact on the probability of an attack being stopped than a blocking element placed closer to the beginning of the critical path of attack. Based on this knowledge, the proposed optimization method adds a few individuals to the initial population that differ primarily in the price range of the proposed solution. This approach avoids that the proposed solutions cover a narrow segment of the search space. The method creates individuals with a focus on three sets of movement nodes: the initial, adjacent to the target node, and those that belong to multiple critical paths. Detection elements are placed at initial nodes and blocking elements are positioned at nodes adjacent to the target nodes. The protection of nodes that are members of multiple critical paths is effective because one security element affects multiple critical paths. Suggesting initial individuals reduces the number of iterations required to find quality, near-optimal solutions. Evolutionary algorithms propose solutions and require the evaluator to evaluate the quality of each of solution. The proposed solution contains a set of security elements that are placed at the construction elements of a protected facility and possibly affect the movement of the attacker. For each proposed solution, the evaluator modifies the search space, finds critical attack paths, and calculates the cost of proposed security elements and interruption probability. Such an optimization process requires a large number of changes to the search space and the search for critical attack paths. Introducing an algorithm that avoids unnecessary changes to the search space and calculates critical attack paths reduces the number of optimization iterations required and consequently shortens its execution time. The introduced method takes into consideration knowledge acquired during searches to avoid unnecessary calculations. Because of the way it works, this method is named Domain Experienced Exploration (D-EX2). The main concept is to check if the intersection of known solutions and the proposed solution is an empty set. If placing proposed security elements does not impact an already known critical path, then there is no need to calculate the critical path. An algorithm returns the previously found solution. If there is no knowledge about proposed security elements placement, then the algorithm will update search space, find critical paths, calculate solution cost and

interruption quality, and add it to the list of found solutions. As the number of evaluations is increased, D-EX2 knowledge expands and the requirement for new calculations is reduced. The D-EX2 method can be used in parallel by different evolutionary algorithms to reduce the number of evaluations and shorten the time of the optimization.

The experimental results of methods outlined in chapter four are presented in chapter five ("5. Experimental results"). Empirical experiments were carried out over real-world facilities representing a standard commercial building and a high-security facility thus employing optimization methods at different problem sizes and facility layouts. The first set of experiments has shown that the method for optimizing the graph of possible adversary movement, by removing facility elements that will never belong to the critical adversary path, significantly reduces the search space, up to 65%. The second set of experiments compares the number of evaluations carried out by evolutionary algorithms with and without the D-EX2 method. Multi-objective optimization was conducted using non-dominated sorting genetic algorithm NSGA-II. The optimization result is a spread of solutions near the true Pareto-optimal front. Numerous experiments were performed with different parameters of genetic algorithms to determine the dependence of the D-EX2 method optimization efficiency on the evolutionary algorithm population size, number of generations, and mutation probability. Experiments have shown that a higher number of generations increases D-EX2 efficiency as continuous algorithm execution builds-up D-EX2 experience while mutations slightly decrease method efficiency in some cases. Different facility layouts used for experiments had shown that the D-EX2 method results in better efficiency at facility layouts with a higher number of possible entries. For real-world facilities, the D-EX2 method avoided up to 75% unnecessary evaluation calculations. In the third set of experiments the initial population was populated using domain knowledge and the quality of results was compared to the standard implementation of evolutionary algorithms where the initial population is created randomly. Results have shown that populating the initial generation with few individuals, based on domain knowledge, produces better quality results no matter which genetic algorithm parameters are used. An additional advantage is that the results are uniformly dispersed on the Pareto front.

The presented methods provide an automatic search for a set of solutions, where each solution presents a near-optimal security system for a given price range while avoiding

unnecessary calculations. Using domain knowledge allows for the discovery of near-optimal solutions with fewer evaluations compared to standard evolutionary algorithms.

Key words:

domain experienced exploration method, D-EX2, numerical analysis of physical protection system, multi-objective optimization, genetic algorithm

SADRŽAJ

1. UVOD

1.1. Uvodna razmatranja

Uobičajen način zaštite štićenog objekta je implementacijom sustava tehničke zaštite koji se sastoji od sustava protuprovale, videonadzora, kontrole pristupa itd. Proizvođači opreme za tehničku zaštitu udružuju se u globalne inicijative standardizacije kako bi olakšali interoperabilnost sustava. Velika su ulaganja proizvođača u primjenu umjetne inteligencije u svrhu povećanja učinkovitosti cjelokupnog sustava tehničke zaštite. Sveprisutan proces digitalizacije, ne samo sustava tehničke zaštite, već i poslovnih i industrijskih procesa, uglavnom zaobilazi projektiranje sustava tehničke zaštite. Iako su još u prošlom stoljeću predstavljene metode numeričke analize ugroženosti, koje su osnova za procjenu kvalitete projektiranog rješenja zaštite, projektanti sustava tehničke zaštite i dalje najčešće koriste subjektivne metode za projektiranje sustava zaštite.

Na osnovu stečenog iskustva projektanti odabiru i razmještaju zaštitne elemente. Veliki izbor zaštitnih elemenata i lokacija na koji se odabrani elementu mogu postaviti, pridonose velikoj vjerojatnosti da projektant nije odabrao projektno rješenje koje je približno optimalno. Dodatni je nedostatak subjektivno procjenjivanje učinkovitosti projektiranog rješenja. Pretraživanje mogućih rješenja, predlaganje približno optimalnih rješenja i numeričko određivanje postignute razine zaštite zahtijeva alat koji će projektantu omogućiti znatno brže i kvalitetnije projektiranje. Uz današnji razvoj algoritama strojnog učenja i njihove primjene u raznim područjima očekuje se tržišna dostupnost većeg broja aplikativnih rješenja koja provode numeričku analizu sustava zaštite i predlažu optimalna rješenja. Međutim, takvih je rješenja, poput *Estimate of Adversary Sequence Interruption (EASI), Forcible Entry Safeguards Effectiveness Model (FESEM), Insider Safeguards Effectiveness Model (ISEM), Safeguards Automated Facility Evaluation (SAFE), System Analysis of Vulnerability to Intrusion (SAVI)* ili *Safeguards Network Analysis Procedure (SNAP)* [1]

malo i ne pružaju punu funkcionalnosti niti predlažu optimalna rješenja na razini odabira i pozicioniranja zaštitnih elemenata.

Osnovni preduvjet za automatiziranu provedbu opisanog optimiranja je standardizacija opisivanja sustava tehničke zaštite. Standardom bi se odredio način numeričke ocjene učinkovitosti projektiranog sustava zaštite. Potrebno je standardizirati i eksperimente kojima se provodi utvrđivanje učinkovitost zaštitnih elemenata, obzirom na vjerojatnost detekcije napadača i vrijeme zadržavanja. Standardiziranje modela zaštite i učinkovitosti zaštitnih elemenata omogućuje razvoj različitih pristupa odabiru i pozicioniranju zaštitnih elemenata.

1.2. Motivacija

Analiza ugroženosti štićenog objekta provodi se subjektivnom metodom. Projektant odabire opremu kojom će zaštiti štićeni objekt i izvedbenim projektom, a na osnovu svojeg iskustva, definira na koje pozicije ta oprema treba biti postavljena. Takav pristup stvara veliku mogućnost pogrešnog odabira rješenja i traži veliki angažman. Cilj ovog rada je razviti metodologiju koja će iskoristiti definiciju štićenog objekta u digitalnom obliku za provedbu numeričke analize ugroženosti i automatizirati optimiranje sustava zaštite. Optimiranjem se predlaže projektantu odabir opreme i njeno pozicioniranje unutar štićenog objekta. Metodologija mora projektantu predložiti više približno optimalnih rješenja, koja se razlikuju po cijeni i kvaliteti. Iz skupa predloženih rješenja projektant može odabrati ono koje najbolje odgovara zahtjevima, potrebama i financijskim mogućnostima korisnika.

1.3. Pristup i doprinos

Opišu li se štićeni objekt i dostupni elementi zaštite matematički, mogu se provesti različite metode optimiranja u svrhu pronalaska skupa približno optimalnih pozicija i vrsta zaštitnih elemenata. Specifičnost problema ovog optimiranja je da svako predloženo rješenje zahtijeva ponovnu izgradnju prostora pretraživanja i višestruki pronalazak optimalnih putova napada. Stoga su istraženi postupci koji uzimaju u obzir specifičnosti problema i ubrzavaju pronalazak kvalitetnih rješenja.

Jedan od doprinosa ovog rada je izrada modela sustava zaštite nad kojim se može provesti optimiranje pozicije i vrste zaštitnih elemenata. Modelom se opisuje ne samo štićeni objekt, već i tjelesna zaštita, napadač i zaštitni elementi. Uveden je i postupak ubrzanog izračuna funkcije ugroženosti i cijene sustava na temelju rezultata u prethodnim iteracijama, čime se izbjegava nepotrebna provedba evaluacije rješenja. Hibridna metoda optimiranja smanjuje prostor pretraživanja bez utjecaja na kvalitetu rezultata i uzimajući u obzir specifičnosti domene predlaže početna rješenja kojima proces optimiranja pronalazi kvalitetna rješenja u manjem broju iteracija u odnosu na klasične procese optimiranja. Svi doprinosi provjereni su i potvrđeni kroz eksperimente provedene nad modelima koji predstavljaju realne objekte.

1.4. Struktura rada

U poglavlju koje slijedi definirana je numerička metoda za analizu ugroženosti. Predstavljeno je na koji se način numerički može odrediti vjerojatnost uspješnog presretanja napadača. Napadaču je obično na raspolaganju velik broj putova od početne pozicije napada do cilja. Stoga su navedene metode kojima se na optimalan način pretražuju sve mogućnosti kretanja napadača i odabire kritični put napada, odnosno onaj koji mu daje najveće izglede za uspjeh u napadu.

U trećem poglavlju predstavljen je objektni model koncepta zaštite. Osnovne su komponente model napadača, tjelesne zaštite, zaštitnog elementa i štićenog objekta. Ručno opisivanje štićenog objekta vremenski je zahtjevno pa je predstavljena metodologija automatiziranog prijenosa podataka o štićenoj građevini iz standardiziranih geometrijskih modela u predloženi model štićenog objekta. Uz osnovni pregled standarda kojima se opisuju otvoreni i zatvoreni prostori, opisane su metode izrade grafa mogućeg kretanja napadača.

U četvrtom poglavlju detaljno je razrađena metoda optimiranja pozicije i vrste zaštitnih elemenata, koja se provodi automatizirano, uzimajući u obzir sve napadaču dostupne pozicije za početak napada. Uspoređene su jednokriterijska i višekriterijska funkcija cilja optimiranja, a zbog višestrukih prednosti za daljnju razradu optimiranja odabrana je višekriterijska metoda optimiranja. Za provođenje optimiranja predstavljena je metodologija izrade prostora pretraživanja iz objektnog modela zaštite. Prikazana je primjena evolucijskih algoritama nad prostorom pretraživanja te pojašnjeno zašto dinamičko

programiranje nije primjenjivo za ovaj problem. Uvedeno je nekoliko metoda optimiranja koje su nadogradnja standardnom optimiranju evolucijskim algoritmima i predstavljaju doprinose ovog rada. Prikazan je algoritam kojim se smanjuje graf mogućeg kretanja napadača, izbacivanjem elemenata grafa koji ne utječu na kvalitetu rješenja. Izrađen je postupak ubrzanog izračuna funkcije ugroženosti i cijene sustava na temelju rezultata izračuna u prethodnim iteracijama te metoda koja predlaže početna rješenja evolucijskom algoritmu, a na osnovu kojih algoritam postiže kvalitetnija rješenja. Na kraju poglavlja prikazan je cjelokupni proces optimiranja odabira i pozicioniranja zaštitnih elemenata.

Rezultati eksperimentiranja predstavljeni su u petom poglavlju. Provedeni su nad objektima koji predstavljaju uobičajenu poslovnu zgradu i visoko štićeni objekt. Optimiranje grafa mogućeg kretanja napadača provedeno je kao prvi korak optimiranja. Metoda optimiranja iskustvenim evaluatorom testirana je s različitim konfiguracijama grafova i različitim konfiguracijama genetskog algoritma kako bi se utvrdila ovisnost učinkovitosti optimiranja o veličini populacije, broju jedinki i postotku mutacije. Testiran je utjecaj višestrukog izvođenja optimiranja na trend povećanja učinkovitosti metode. Provedeni su eksperimenti s inicijalizacijom populacije korištenjem domenskog znanja i kvaliteta rezultata uspoređena je sa standardnim provođenjem evolucijskih algoritama.

2. NUMERIČKE METODE ZA ANALIZU UGROŽENOSTI

2.1. Projektiranje sustava zaštite

Analiza ugroženosti osnova je za projektiranje i implementaciju sustava tehničke zaštite [2]. Uobičajeno se provodi subjektivnom metodom projektanta koji štićeni objekt proglašava manje, srednje ili visoko rizičnim. Takva procjena nije jednoznačna jer uvelike ovisi o kategorizaciji koju koristi projektant. Mala zastupljenost alata za automatizirano određivanje vjerojatnosti zaustavljanja napada i vremenski zahtjevan unos podataka o štićenom objektu, glavni su razlozi za izradu projekata zaštite zasnovanih na subjektivnoj procjeni [1].

Osnovni rezultat projekta sustava tehničke zaštite je odabir elemenata zaštite, poput kamera i protuprovalnih detektora, odabir pozicija na koje se elementi zaštite postavljaju te uspostava tjelesne zaštite koja je opremljena i ekipirana dovoljno za savladavanje očekivanih tipova napadača. Projektom se definira i način komunikacije kojom sustav tehničke zaštite dojavljuje tjelesnoj zaštiti nastanak alarma [3].

Projektiranjem sustava tehničke zaštite štićenog objekta želi se postići zaustavljanje napadača s određenom vjerojatnošću. Učinkovitost sustava tehničke zaštite može se izraziti kao

$$P_E = P_{\mathrm{I}} \cdot P_N \tag{2.1}$$

pri čemu je P_N vjerojatnost uspješne neutralizacije napadača u trenutku presretanja, a P_I vjerojatnost uspješnog presretanja napadača, što možemo prikazati sljedećom jednadžbom:

$$P_I = P_{\mathrm{D}} \cdot P_C \cdot P_R \tag{2.2}$$

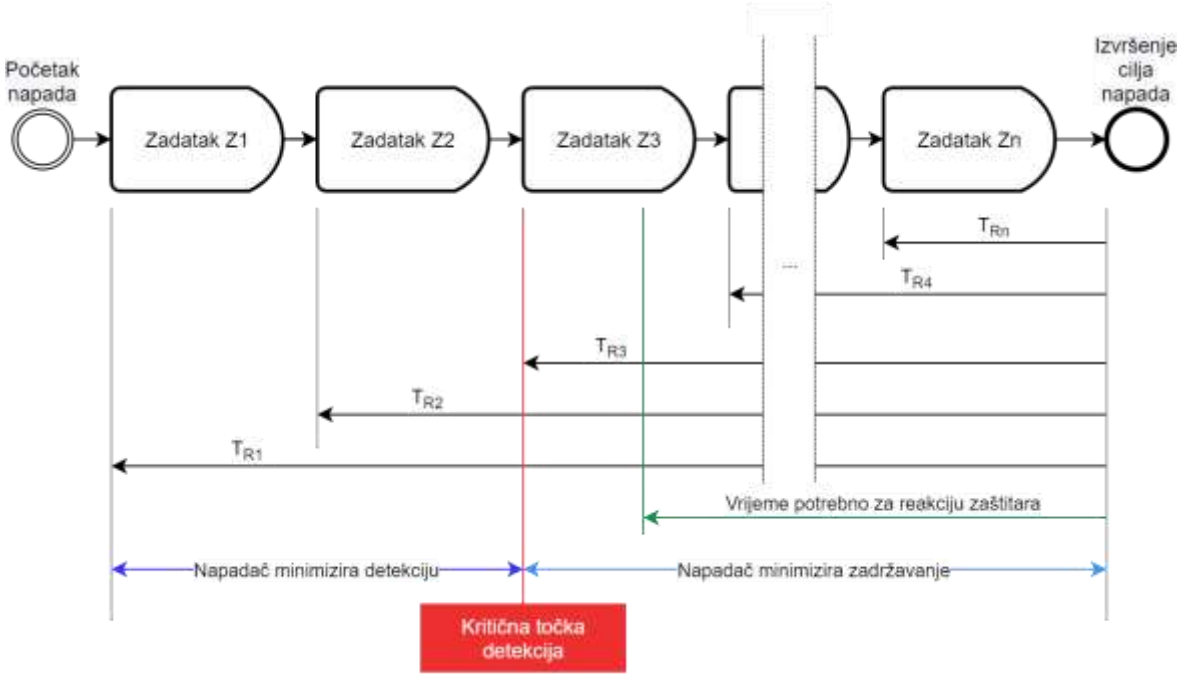
gdje je P_D vjerojatnost da će napadač biti otkriven, odnosno detektiran, prije nego što stigne do cilja. P_C predstavlja vjerojatnost da je komunikacija dostupna u trenutku detekcije kako

bi alarm bio proslijeđen tjelesnoj zaštiti i reakcija tjelesne zaštite pokrenuta. Nakon detekcije napadača, potrebno je neutralizirati napad. P_R je vjerojatnost da će tjelesna zaštita dostići napadača prije nego stigne do cilja.

Vjerojatnost detekcije napadača P_D ovisi o zaštitnim elementima postavljenima na objektu koji se štiti. P_R ovisi o položaju gdje je napadač detektiran i može se iskazati kao funkcija vremena T_R koje je potrebno napadaču za stizanje do cilja od trenutka detekcije u odnosu na vrijeme reakcije tjelesne zaštite T_G. Cilj zaštite je da razlika ta dva vremena bude pozitivna.

$$T_\mathrm{R} - T_G > 0 \tag{2.3}$$

Vrijeme kretanja i savladavanja prepreka je slučajna varijabla kao i vrijeme reakcije tjelesne zaštite i opisuje se normalnom razdiobom pa ova pozitivna razlika $T_\mathrm{R} - T_G$ može biti izražena vjerojatnošću. Kretanje napadača štićenim objektom možemo razložiti na zadatke, odnosno zapreke, koje mora savladati za stizanje do cilja, kako prikazuje slika 2.1.



Slika 2.1 Kretanje napadača razloženo na zadatke

Vrijeme potrebno za savladavanje zadatka Z_i može se prikazati kao T_i. Vrijeme potrebno za dolazak od pojedine zapreke do cilja može se izraziti kao suma zadataka koji slijede na putu od zadatka i do cilja

$$T_{Rj} = \sum_{i=j}^{n} T_i, \tag{2.4}$$

pri čemu je n ukupan broj zadataka.

Zadatak Z_i, za koji vrijedi $T_\mathrm{Ri} \geq T_G$ i $T_\mathrm{R(i+1)} < T_G$, predstavlja kritičnu točku detekcije jer za napadača je nakon savladavanja tog zadatka veća vjerojatnost dostizanja cilja u kraćem vremenu od tjelesne zaštite.

Vjerojatnost P_{Ri} da je $T_{Ri} > T_G$ je određena površinom ispod krivulje gustoće vjerojatnosti normalne razdiobe, u području od $T_{Ri} - T_G > 0$ do ∞:

$$P_{Ri} = E(T_{Ri} - T_G > 0, s_n) = \int_0^{\infty} \frac{1}{s_{ni}\sqrt{2\pi}}\, e^{\frac{(t-(T_{Ri}-T_G)^2}{2s_{ni}^2}}\, dt \tag{2.5}$$

pri čemu je s_{ni} standardna devijacija razlike preostalog vremena i vremena dolaska tjelesne zaštite. Standardna devijacija s_{ni} određena je izrazom:

$$s_{ni} = \sqrt{s_n^2(T_{Ri}) + s_n^2(T_G)} \tag{2.6}$$

gdje su

- $s_n(T_{Ri})$ – standardna devijacija preostalog vremena kod detekcije na i-tom zadatku,
- $s_n(T_G)$ – standardna devijacija vremena reakcije tjelesne zaštite

Vjerojatnost uspješnog presretanja napadača kod zaštitnog sustava s n elemenata na putu napada računa se kao vezana vjerojatnost prema izrazu::

$$P_I = P_{R1}P_{D1}P_C + \sum_{i=2}^{n} P_{Ri}P_{Di}P_c \cdot \prod_{j=1}^{i-1}\left(1 - P_{Dj}P_c\right) \tag{2.7}$$

Na taj način, za svaku putanju kretanja napadača štićenim objektom moguće je odrediti vjerojatnost zaustavljanja.

2.2. Traženje kritičnog puta napada

Napadaču se obično pruža više pozicija s kojih može započeti s napadom. Nakon što odabere početnu poziciju napada, pri kretanju objektom obično ima izbor kojim će se putom kretati prema cilju napada, birajući onaj put za koji očekuje da mu pruža najveću vjerojatnost uspješnosti napada. Stoga je potrebno za svaku napadaču dostupnu početnu poziciju napada pronaći put za koji je vjerojatnost presretanja P_I najmanja. Onaj put kretanja napadača za koji je najmanja vjerojatnost P_I da će napadač biti zaustavljen, naziva se kritičnim putom.

Moguće kretanje napadača može se opisati usmjerenim, težinskim grafom, pri čemu čvorovi predstavljaju zadatke koje napadač savladava pri kretanju objektom, a veze ih povezuju kako bi opisali mogućnosti kretanja. Čvoru se pridjeljuju vjerojatnost detekcije i vrijeme zadržavanja, a vezama vrijeme potrebno za prijelaz iz jednog čvora u drugi. Traženje kritičnih putova napada standardnim algoritmima za pretraživanje grafova, poput algoritma Dijkstra [4] sa složenošću $O(|V^2|)$, vremenski je prezahtjevno nad većim brojem elemenata. U procesu optimiranja sustava zaštite traženje kritičnog puta izvodi se za svaki početni element u svakoj iteraciji optimiranja, što rezultira izvođenjem algoritma traženja kritičnog puta stotinama tisuća puta ili više.

Podesan način traženja kritičnih putova u grafu je heurističkim metodama jer se njima provodi usmjereno pretraživanje [5]. Osnovna ideja heurističke metode traženja optimalnog puta kroz graf je vođenje pretraživanja prema cilju korištenjem dodatnog, heurističkog znanja. Može ga se predstaviti heurističkom funkcijom h koja svakom stanju s pridjeljuje procjenu udaljenosti od tog stanja do ciljnog stanja. Često korišten algoritam za traženje najkraćeg puta između čvorova grafa heurističkom metodom je algoritam A* [6]. Za algoritam A* vrijedi da je optimalan ako je heuristika h optimistična, odnosno ako i samo ako za svako stanje s vrijedi da je $h(s) \leq h^*(s)$, pri čemu je $h^*(s)$ prava cijena stanja s do cilja [7]. Algoritam A*, primijenjen u pretrazi kritičnog puta napada [8], u svakoj iteraciji izvođenja za daljnje kretanje grafom bira iz skupa aktivnih čvorova onaj koji ima najmanju, do tog trenutka, vjerojatnost presretanja napadača P_I. Zatim, primjenjujući heuristiku h, svaki njegov susjedni čvor, koji još nije potpuno istražen, promatra kao predzadnji čvor na putu do ciljnog, uz pretpostavku da se napadač između promatranog i ciljnog čvora kreće maksimalnom brzinom i direktno k cilju. Time se ne precjenjuje moguće vrijeme kretanja

od promatranog do ciljnog čvora i pretpostavlja da na putu do cilja nema elemenata zaštite, čime se osigurava optimističnost heuristike jer vrijedi da je $h(s) \leq h^*(s)$.

Nekoliko je metoda koje prilagođavaju izvršavanje algoritma A* paralelnom načinu rada, iskorištavajući višeprocesorske arhitekture računala, pri čemu je glavni izazov distribucija izračuna bez sinkronizacije kroz duži period [9].

Brži pronalazak optimalnog puta kretanja grafom omogućuju algoritmi koji pretraživanje izvode od početnog i ciljnog čvora, a cilj im je smanjiti broj čvorova koje algoritam analizira kretanjem kroz graf. Primjer je metoda NBA* u kojoj dva procesa traženja dijele skup $\mathcal{M}$ koji sadrži čvorove za istražiti [10]. Metoda NBA* se izvršava naizmjenično od početnog i ciljnog čvora. Umjesto naizmjeničnog izvođenja, PNBA* uvodi paralelno izvršavanje NBA*, korištenjem arhitekture dijeljene memorije [11]. Dijeljenje resursa svedeno je na minimum kako bi se postigle dobre performanse izvođenja.

Usporedba heurističkih pretraga optimalnog puta od početnog čvora te početnog i završnog čvora [12] pokazuje da dvostrana pretraga ne smanjuje u svim situacijama broj pretraženih čvorova grafa već za neke grafove postiže slabije rezultate u odnosu na jednostranu pretragu. Algoritam „*Meet in the Middle*“ [13] uvodi dodatne uvjete za biranje čvorova kojima garantira da će se pretraživanje od početnog i završnog čvora sresti na sredini optimalnog puta, bez nepotrebnog pretraživanja čvorova.

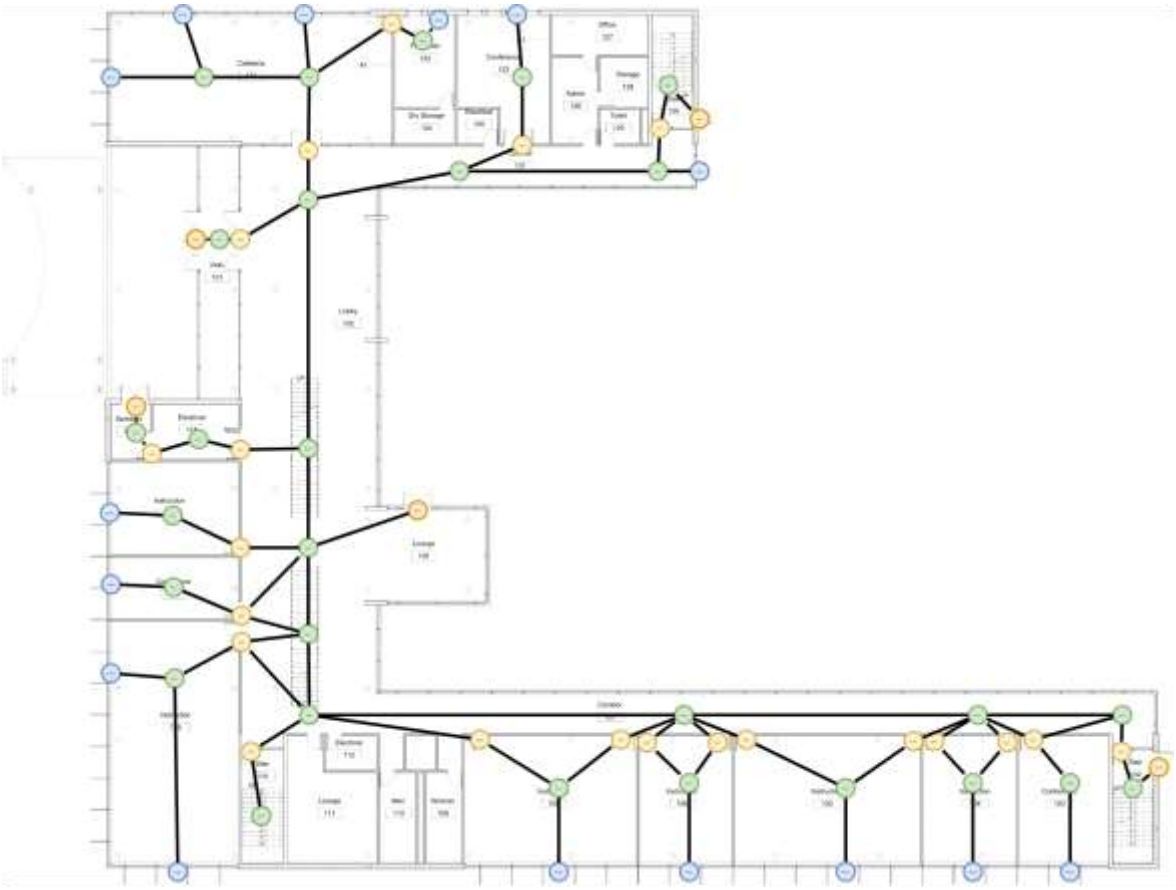
3. OBJEKTNO MODELIRANJE KONCEPTA ZAŠTITE

Osnova za numeričku analizu objekta i optimiranje sustava zaštite su modeli napadača, tjelesne zaštite i objekta koji se štiti. Za automatiziranje opisa objekta koji se štiti, model mora biti osmišljen tako da omogući preuzimanje podataka iz standardiziranih, geometrijskih modela za opisivanje građevina i otvorenih prostora. Od modela sustava zaštite se očekuje da podrži opisivanje utjecaja svojstva napadača na mogućnosti kretanja prostorom i učinkovitost elemenata zaštite.

U ovom poglavlju opisani su standardizirani, geometrijski opisi građevine koji se najčešće koriste u procesu projektiranja objekata i mogu se iskoristiti za predstavljeni model štićenog objekta kojim se opisuju unutarnji i vanjski prostori. Predstavljenim modelom podržan je i opis tjelesne zaštite i napadača. U odnosu na postojeći model objektnog modeliranja koncepta zaštite [8], [14], gdje su elementi detekcije i elementi blokiranja predstavljeni zasebnim klasama, u predstavljenom su modelu oni objedinjeni u klasu zaštitnog elementa. Bitna je razlika da se ovim modelom, uz opisivanje štićenog objekta, tjelesne zaštite i napadača, definira i model mogućeg kretanja napadača nad kojim se vrši višekriterijsko optimiranje odabira i pozicije zaštitnih elemenata sustava zaštite.

3.1. Modeliranje štićenog objekta

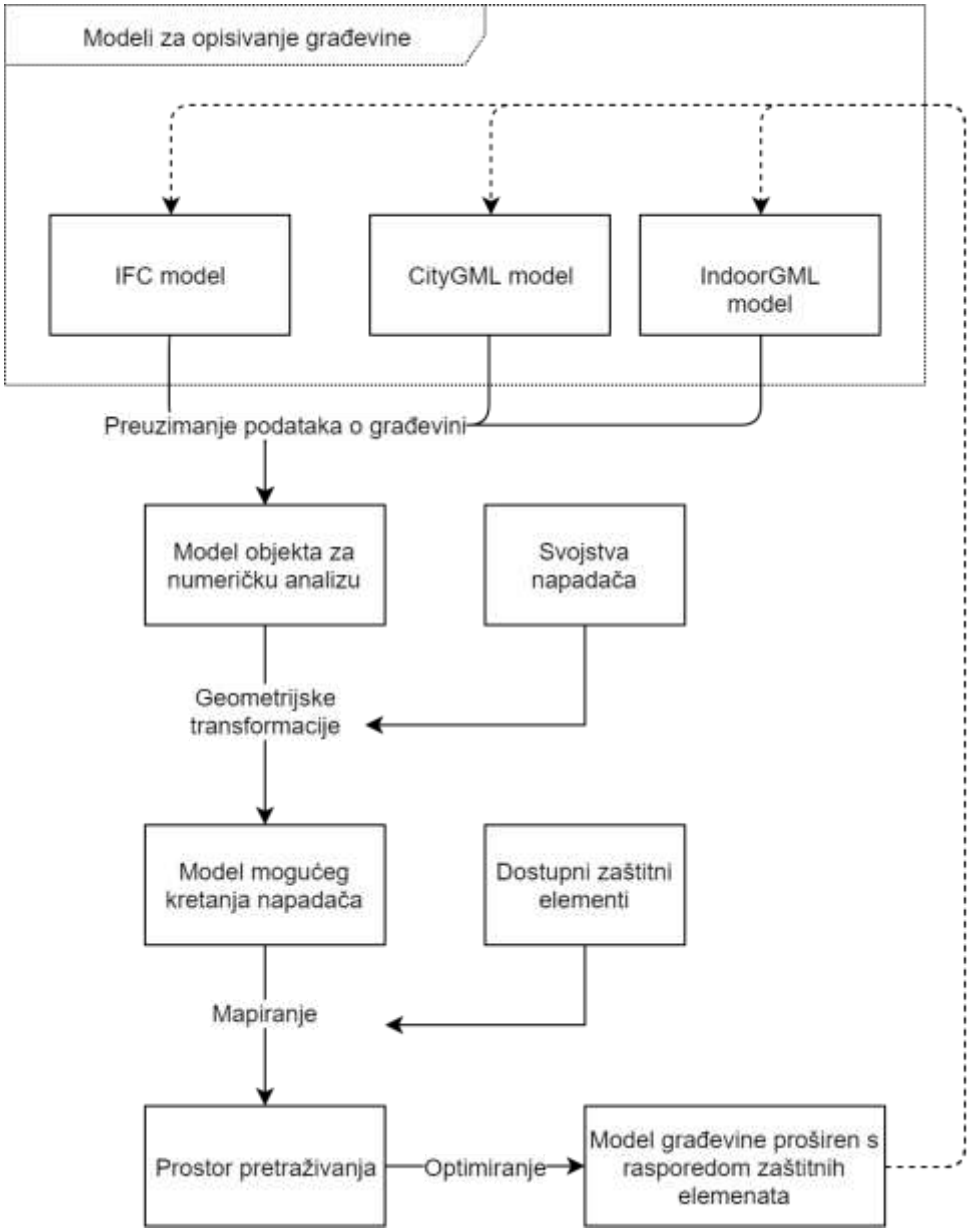
Za provedbu numeričke analize ugroženosti objekta i optimiranje rasporeda elemenata zaštite pogodan opis štićenog objekta je graf, gdje čvorovi predstavljaju elemente štićenog objekta, a veze moguće putove kretanja. Modeliranje štićenog objekta provodi transformaciju informacije o fizičkom objektu u graf (slika 3.1).



Slika 3.1 Prikaz štićenog objekta grafom

Svaki element štićenog objekta koji može poslužiti napadaču kao element kretanja uzima se u obzir pri izradi grafa. U to spadaju vrata, prozori, koridori, stepeništa, dizala itd. Za napadače opremljene dodatnim alatima potrebno je predvidjeti mogućnost prolaska kroz zidove, podove, stropove, ograde, krovišta i slično pa se u obzir uzimaju i takvi elementi objekta.

Slika 3.2 prikazuje osnovne korake u transformaciji objektno opisanog modela građevine u model koji opisuje odabir i pozicioniranje zaštitnih elemenata u svrhu povećanja zaštite.



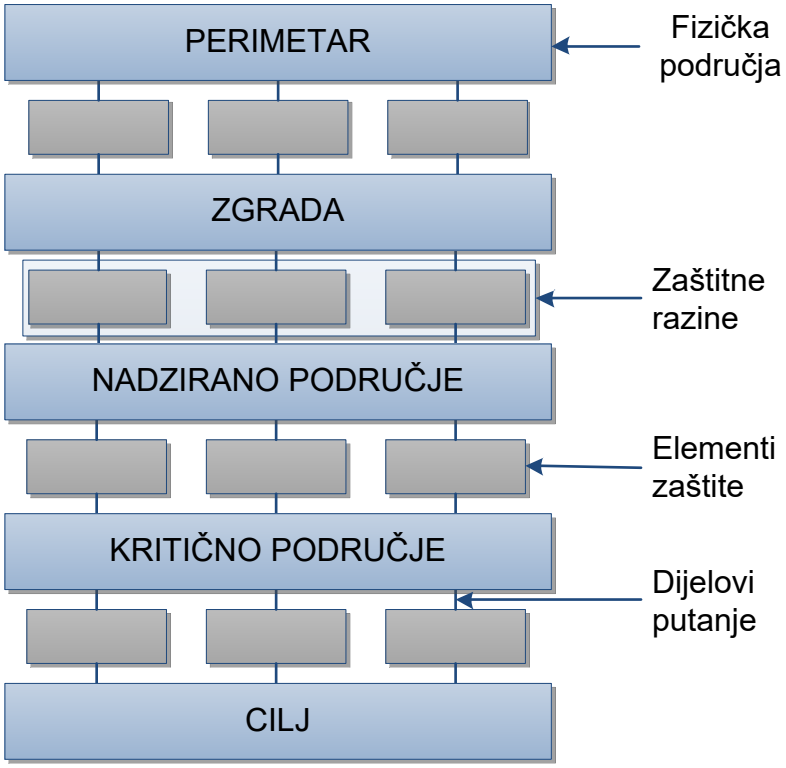
Slika 3.2 Uporaba standardiziranih modela za opis optimiranog sustava zaštite

Prvi je korak preuzeti geometrijski i semantički opis građevine, opisan nekim od standardiziranih shema, opisanih u daljnjem tekstu. Uz vrstu, poziciju i veličinu pojedinog elementa prostora, preuzima se podatak o materijalu od kojeg je element sačinjen. Tim se podacima popunjava model objekta za numeričku analizu nad kojim se, uzimajući u obzir svojstva napadača, provode geometrijske transformacije kojima se izrađuje graf kretanja. Njime su opisane pozicije čvorova koji predstavljaju prostore kretanja i veze kojima su definirani mogući putovi kretanja. Kombiniranjem grafa kretanja i mogućih lokacija zaštitnih elemenata stvara se prostor pretraživanja. Optimiranje sustava omogućuje

proširenje početnog opisa građevine dodavanjem elemenata zaštite. Pri tome se određuje njihova količina i lokacija postavljanja.

3.1.1. Ručno opisivanje fizičke razine štićenog objekta

U počecima razvoja numeričke analize ugroženosti, štićeni objekt se modelira dijagramom tijeka napada (*engl. Adversary Sequence Diagram*, ASD) na osnovu kojeg se mogu izračunati kritični putovi napada [15]. Proces modeliranja očekuje od korisnika da ručno definira dijelove štićenog objekta na način da ga podijeli na logičke cjeline, kako prikazuje slika 3.3. Elementi štićenog objekta definiraju se kao fizička područja između kojih se postavljaju zaštitne razine u koje se smještaju elementi kretanja i elementi zaštite. Oni se povezuju čineći moguće putove kretanja. Ručno definiranje svih elemenata objekta zahtijeva velik angažman projektanta pa se oblikovanje objekta pojednostavljuje što ovaj postupak čini manje preciznim u odnosu na novije metode.



Slika 3.3 Modeliranje dijagrama tijeka napada (preuzeto iz [8]

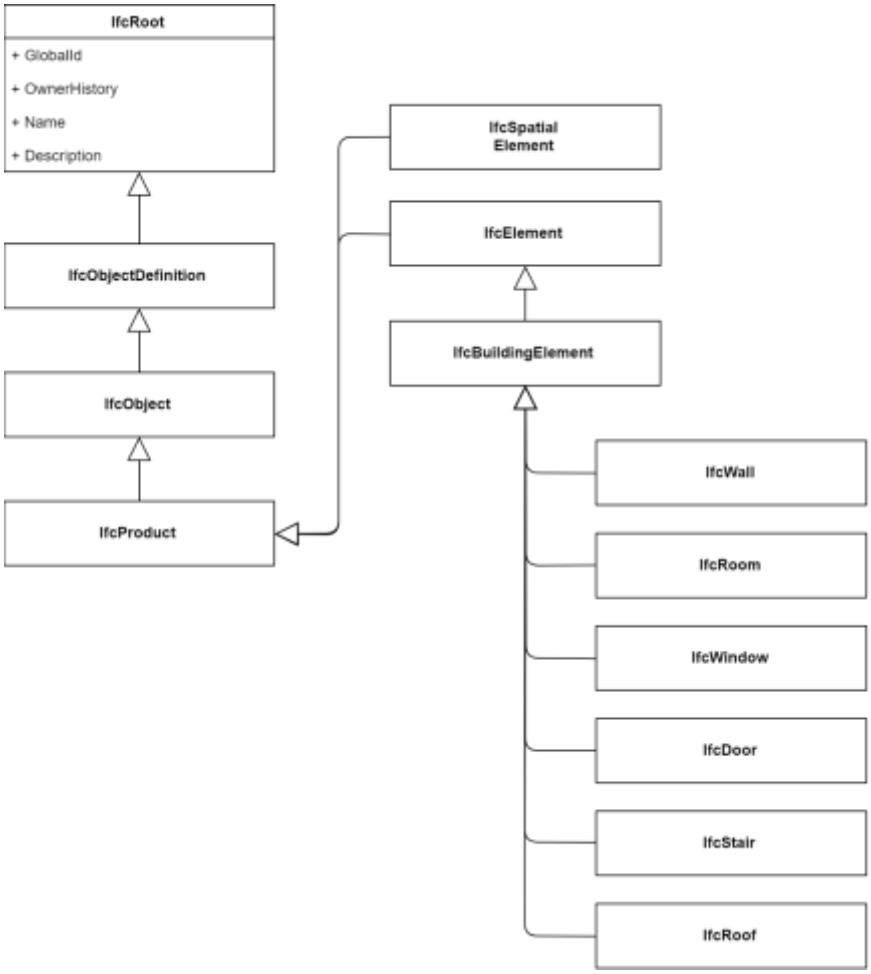
3.1.2. Geometrijski modeli opisivanja građevine

Građevine se danas projektiraju računalnim aplikacijama koje zapisuju definiciju građevine u digitalnom obliku. Isprva je projektiranje računalom bilo zamjena za ručno crtanje, pri čemu je korištenje računala ubrzalo proces oblikovanja objekta. Daljnjim razvojem aplikativnih rješenja proces se unaprjeđuje pa se građevine počinju projektirati na način da se svaki element građevine definira kao objekt, koji ima zadanu poziciju u prostoru, oblik i dimenzije, materijal i međuovisnost s ostalim elementima u trodimenzionalnom prostoru. Takav zapis omogućuje automatiziranje transformacije opisa u oblik podesan za optimiranje.

Prvi radovi koji objektno opisuju model građevine pojavljuju se u sedamdesetim godinama prošlog stoljeća [16]. Danas česti pristup modeliranju složenih objekata je korištenjem *Building Information Modeling* (BIM) procesa [17]. Digitalno opisivanje fizičkih i funkcionalnih karakteristika objekta omogućuje njegovo cjeloživotno upravljanje, od projektiranja preko izgradnje do održavanja i uklanjanja. Trend korištenja BIM procesa potvrđen je i uvođenjem 2019. godine ISO 19650 standarda „*Organization and digitization of information about buildings and civil engineering works, including building information modelling (BIM) -- Information management using building information modelling*". Neke od platformi za upravljanje BIM procesima su Allplan, ArchiCAD, Revit i Vectorworks. BIM platforme koriste različite formate internog zapisivanja oblikovanja objekta. Za razmjenu podataka razvijen je otvoreni, internacionalni standard *Industry Foundation Classes* (IFC), registriran pod ISO 16739-1. Uz IFC način opisivanja građevine, visoko su zastupljeni geometrijski modeli CityGML i IndoorGML.

3.1.2.1 Industry Foundation Classes model

IFC shema opisuje BIM entitete u obliku klasa s pripadajućim atributima, metodama i odnosima. Shema je zasnovana na nasljeđivanju klasa i predviđa uvođenje dodatnih opisa entiteta što omogućuje njenu primjenu u različitim sustavima koji sudjeluju u BIM procesu. Slika 3.4 hijerarhijski prikazuje klase IFC modela koje su značajne za opisivanje fizičke izvedbe projekta u svrhu provedbe numeričke analize i optimiranja sustava zaštite.



Slika 3.4 Klase IFC modela značajne za optimiranje zaštite

Osnovna klasa modela je *IfcRoot* i sadrži globalno jedinstven identifikator objekta, njegov naziv, opis i povijest promjena vlasnika. Klasa *IfcObjectDefinition* je apstraktna klasa za sve elemente koji sadrže definicije, a nasljeđuje je klasa *IfcObject* koju nasljeđuju svi objekti. Klasa *IfcProduct* određuje oblik i poziciju objekta. Oblik je opisan apstraktnom klasom *IfcProductRepresentation* koju nasljeđuje *IfcProductDefinitionShape*. Omogućeni su različiti načini opisivanja geometrije, sukladno standardu ISO 19393-42:1994. Pozicija elementa u prostoru definirana je klasom *IfcObjectPlacement*. Elementi se postavljaju u Kartezijev koordinatni sustav, pri čemu koordinate mogu biti apsolutne ili relativne koordinatama nadređenog elementa, čime se elementi postavljaju relativno koordinatnom sustavu projekta. Elementi u građevinarstvu i arhitekturi opisuju se klasom *IfcElement,* koja omogućuje opis odnosa elementa s obližnjim elementima. Njihov odnos opisan je klasom *IfcRelConnectsElements*. Primjer je povezanost zida i pripadajućih prozora i vrata. Objekti klase *IfcElement* mogu se povezivati u logičku grupu klasom *IfcSpace*. Građevni elementi

poput zida i prozora nasljeđuju klasu *IfcBuildingElement*. Svaka vrsta građevnog elementa može se opisati jednom od naslijeđenih *IfcBuildingElement* klasa poput *IfcWall* za zid, *IfcRoom* za sobu, *IfcWindow* prozor, *IfcDoor* vrata, *IfcStair* stepenice, *IfcRoof* za krov itd. Vrsta materijala od kojih se sastoje objekti naslijeđeni od *IfcElement* definira se klasom *IfcMaterial* koja se dodjeljuje s *IfcRelAssociatesMaterial*. Pojedini *IfcBuildingElement* ima svoje specifične atribute. Primjer je klasa *IfcDoor* koja atributom *PredefinedType* određuje vrstu vrata pobrojanim tipom *IfcDoorTypeEnum*: *Door* (standardna vrata), *Gate* (ulazna vrata), *Trapdoor* (dvostruka vrata u svrhu očuvanja sigurne zone pri ulasku i izlasku), *UserDefined* (korisnički definirana). Atributom *OperationType* definiran je način rada vrata pobrojanim tipom *IfcDoorTypeOperationEnum* kojim se definira jesu li vrata jednokrilna ili dvokrilna, zaokretna, klizna ili okretna, u kojem se smjeru otvaraju itd. Ova dva atributa bitna su pri izradi modela za numeričku analizu jer se na osnovu njih odabire predefinirani tip vrata.

IFC opisuje geometriju i poziciju elemenata na više načina. Koncept predloškom „*Product Placement"* predviđa postavljanje elementa u Kartezijev koordinatni sustav pri čemu koordinate mogu biti relativne koordinatama nadređenog elementa, čime se elementi postavljaju relativno koordinatnom sustavu projekta. Drugi je koncept predloškom „*Product Geometric Representation"* koji klasom *IfcProductDefinitionShape* omogućuje različite načine opisivanja geometrije, sukladno standardu ISO 19393-42:1994.

3.1.2.2 Geoprostorni opis otvorenih i zatvorenih prostora

Optimiranje zaštite ne provodi se samo nad samostojećim objektima, već i nad kompleksnim objektima koji objedinjuju otvorene prostore poput ograda, travnjaka, ceste i zatvorenih prostora poput zgrada. Takvi kompleksni objekti mogu se opisati geoprostornim modelima CityGML i IndoorGML. CityGML standard je svojom strukturom i opisom detalja pogodan za opisivanje otvorenih prostora i zatvorenih prostora, dok je IndoorGML prilagođen opisivanju zatvorenih prostora [18].

CityGML je internacionalni *Open Geospatial Consortium* (OGC) standard za trodimenzionalno opisivanje modela gradova, temeljeno na *eXtensible Markup Language* (XML) shemi. Njime se opisuju urbani prostori i trodimenzionalni objekti u pet razina detalja. Četvrta razina određuje unutrašnjost objekata, a čine je elementi poput soba, namještaja, stepeništa itd. Zbog sposobnosti opisivanja urbanih prostora CityGML je

pogodan za opisivanje kompleksnih prostora koji se sastoje od više građevina, okruženih različitom vrstom terena i međusobno povezanih različitom prometnom infrastrukturom. Za razliku od IFC-a, u kojem je odnos između elemenata moguće izraziti na više načina, CityGML-a striktnije određuje njihove odnose [19]. Geometrijske modele opisuje sukladno standardu ISO 19107, a opis modela može se dodatno proširit *Application Domain Extension* shemom.

IndoorGML je OGC standard za otvoreni podatkovni model koji koristi XML shemu za opis prostora unutar objekta, sukladno *Geography Markup Language* (GML) sintaksi za opisivanje geoinformacija [20]. Za razliku od IFC-a i CityGML-a, specijaliziran je za opis unutarnjeg prostora pa je shema IndoorGML-a jednostavnija. Trenutna verzija IndoorGML opisuje dijelove arhitekture koji se mogu koristiti kao prostori kretanja, u što spadaju sobe, ulazi, koridori, vrata, stepenice itd. Geometrija elemenata opisuje se standardom ISO 19107 ili referenciranjem na opis u CityGML modelu. Za kompleksnije sustave može se kombinirati sa CityGML [21].

3.1.2.3 Automatiziranje prijenosa u model štićenog objekta

Za građevinski kompleks opisan standardima IFC ili CityGML četvrte razine detalja, moguće je automatizirano preuzeti podatke potrebne za opis prostora nad kojim se želi provesti numerička analiza ugroženosti i izvršiti optimiranje sustava zaštite. Osnovni su podaci elementi prostora kojima se napadač može kretati, materijal od kojeg su izrađeni, njihova pozicija u prostoru i dimenzije. Tablica 3.1 daje usporedbu klasa IFC, CityGML i IndoorGML standarda koje su osnova za opisivanje prostora u svrhu optimiranja sustava zaštite.

Tablica 3.1 Usporedba IFC, CityGML i IndoorGML klasa

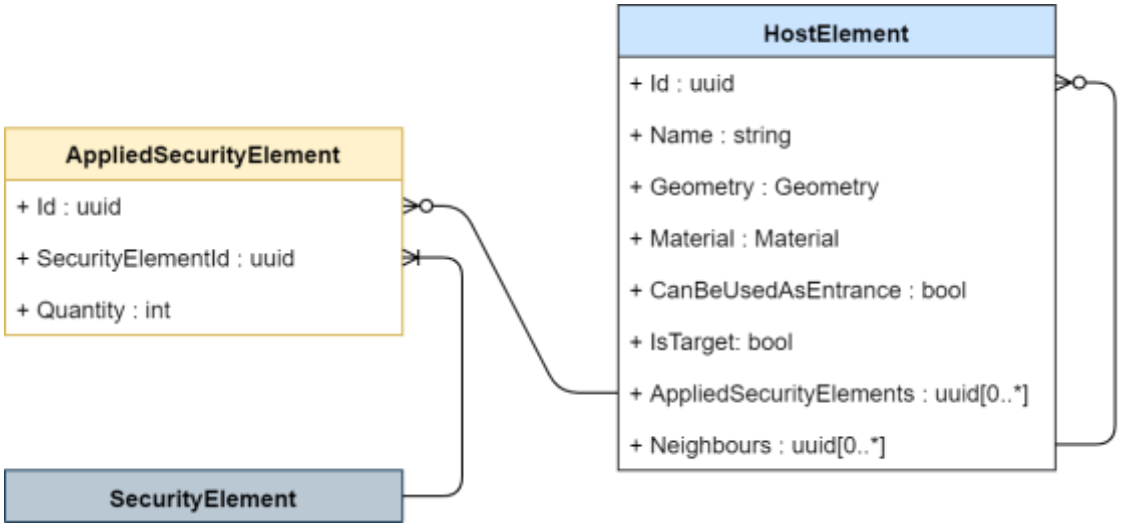
Ontologija	**IFC**	**CityGML**	**IndoorGML**
Prostorija	IfcRoom	Room	GeneralSpace
Vrata	IfcDoor	Door	ConnectionSpace
Prozor	IfcWindow	Window	---
Stepenice	IfcStair	Room (stairs)	TransitionSpace

Pokretne stepenice	IfcTransportElement	Room (escalator)	TransitionSpace
Dizalo	IfcTransportElement	Room (elevator)	TransitionSpace

Za pretvorbu podataka iz IFC modela u model grafa, pri čemu su čvorovi elementi objekta, a veze njihove poveznice, više je pristupa [22], a dostupan je i gotov alat [23]. Dodatna metoda dostupna za modeliranje i izračun kretanja osoba zgradom, pod imenom *Universal Circulation Method* (UCM), temelji se na težinskom grafu i koristi se kao gotov alat [24]. Za kretanje prostorom predviđa korištenje prostorija. Teo i Cho [25] predlažu višenamjenski geometrijski mrežni model (*eng. Multi-purpose geometric network model, MGNM*) koji objedinjuje opis mogućeg kretanja unutarnjim i vanjskim prostorom. Ovom metodom automatizirano se preuzimaju informacije o mogućim prostorima kretanja, poput soba, vrata, prozora, cesta, i povezuje ih se topološki. Podaci se preuzimaju iz IFC modela za unutarnje prostore i CityGML modela za vanjske prostore. Model podržava pridjeljivanje čvorovima semantičkih informacija o prostoru. Povezivanje više zgrada vanjskim prostorom omogućeno je podrškom za CityGML. Model podržava i povezivanje prostora stepeništima u višekatnicama.

3.1.3. Model gradbenog elementa štićenog objekta

Model gradbenog elementa štićenog objekta opisuje svojstva elemenata štićenog objekta potrebna za izradu modela mogućeg kretanja napadača. Svrha modela je univerzalno opisati štićeni objekt, neovisno na koji je način prvotno definiran - ručno, modeliran nekim od ranije opisanih standarda (IFC, CityGML) ili nekim drugim modelom. Slika 3.5 prikazuje model gradbenog elementa štićenog objekta.



Slika 3.5 Model gradbenog elementa štićenog objekta

Osnovna klasa modela je *HostElement* i predstavlja sve gradbene elemente štićenog objekta. Može se prikazati kao

```
HostElement = {  Id : uuid,
                 Name : string,
                 Geometry : Geometry,
                 Material : Material,                    (3.1)
                 CanBeUsedAsEntrance : bool,
                 IsTarget : bool,
                 AppliedSecurityElements : uuid[0..*],
                 Neighbours : uuid [0..*]   }
```

pri čemu je

Id – jedinstveni identifikator elementa na razini sustav, za koji se koristi se tip UUID (*engl. Universal Unique IDentifier*), sastavljen od 32 znaka (znamenke 0-9, slova a-f) i iskazuje se u pet grupa odvojenih povlakom (-) u obliku 8-4-4-4-12,

Name – naziv elementa koji jedinstveno opisuje element, poput Vrata0042,

ForeignId – jedinstveni identifikator u trećem sustavu u kojem je oblikovana građevina, poput IFC ili CityGML

Geometry – pozicija i oblik elementa, sukladno standardu ISO 19393-42:1994

Material – pobrojani tip materijala od kojeg je element sačinjen. Koristi se za određivanje sposobnosti kretanja napadača kroz element i uvjetuje koji se zaštitni elementi mogu primijeniti na ovaj gradbeni element.

CanBeUsedAsEntrance – određuje može li element biti korišten kao početak napada, odnosno ulaska u štićeni objekt.

IsTarget – određuje za element smatra li ga se ciljem napada.

AppliedSecurityElements – lista identifikatora zaštitnih elemenata koji su dodijeljeni ovom elementu štićenog objekta.

Neighbours – lista identifikatora gradbenih elemenata koji su susjedni elementi.

Zaštitni elementi dodijeljeni gradbenom elementu *HostElement* definirani su klasom

```
AppliedSecurityElement = {
                Id : uuid,
                SecurityElementId : uuid,
                Quantity : int
                }
```
(3.2)

pri čemu je

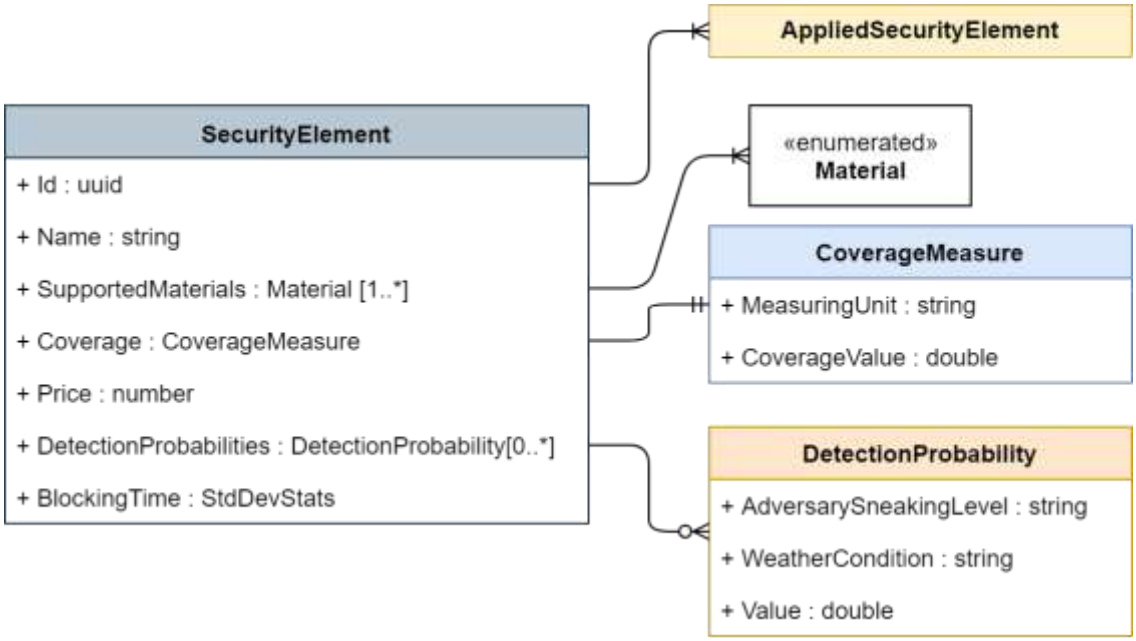
Id – jedinstveni identifikator,

SecurityElementId - jedinstveni identifikator zaštitnog elementa dodijeljenog temeljnom gradbenom elementu,

Quantity – količina zaštitnog elementa dodijeljenog temeljnom gradbenom elementu. Za svaki zaštitni element definirana je površina maksimalna površina koju element štiti. Stoga algoritam u procesu optimiranja određuje količinu zaštitnih elemenata potrebnih za zaštitu gradbenog elementa.

3.1.4. Modeliranje zaštitnog elementa

Slika 3.6 prikazuje model zaštitnog elementa koji se postavlja u štićeni objekt u svrhu optimiranja zaštite.



Slika 3.6 Model zaštitnog elementa

Zaštitni elementi definirani su klasom

```
SecurityElement = {
     Id : uuid,
     Name : string,
     SupportedMaterials : Material[],
     Coverage : CoverageMeasure
     Price : number,
     DetectionProbabilities : DetectionProbability[],
     BlockingTime : StdDevStats
   }
```
(3.3)

pri čemu je

Id – jedinstveni identifikator zaštitnog elementa,

Name – naziv zaštitnog elementa,

SupportedMaterials – lista materijala na koje se ovaj zaštitni element može postaviti. Pobrojivi tip *Material* dodijeljen je gradbenom elementu čime uvjetuje koji se zaštitni elementi mogu primijeniti.

Coverage – određuje koliku udaljenost, površinu, volumen ili slično pokriva jedan zaštitni element. Na osnovu ove definicije izračunava se koliko je zaštitnih elemenata potrebno za primjenu na gradbenom elementu, ovisno o njegovim geometrijskim svojstvima.

Kompleksnog je tipa *CoverageMeasure*, čiji su članovi *MeasuringUnit* koji definira mjernu jedinicu poput udaljenosti, površine ili volumena i *CoverageValue* koji definira vrijednost pokrivanja. Tablica 3.2 prikazuje primjere za dva zaštitna elementa, koristeći tehničke karakteristike zaštitnih elemenata iz [2].

Tablica 3.2 Primjer vjerojatnosti detekcije zaštitnog elementa

Zaštitni element	CoverageMeasure	
SecurityElement.Name	MeasuringUnit	CoverageValue
Mikrovalna barijera za nadzor perimetra (mono)	Udaljenost(m)	120
Detektor loma stakla	Površina(m^2)	9

Price – određuje jediničnu cijenu zaštitnog elementa. Koristi se, zajedno s količinom *HostElement.AppliedSecurityElements.Quantity* primijenjenih zaštitnih elemenata na gradbenom elementu za izračun ukupne cijene sustava.

DetectionProbabilities – lista vjerojatnosti detekcije, ovisna je o vrsti zaštitnog elementa. Za element koji nema funkciju detekcije, ova lista je prazna. Zaštitni element čija vjerojatnost detekcije ne ovisi o sposobnostima napadača ili vremenskim uvjetima ima jedan član ove list. Zaštitnim elementima na čiju vjerojatnost detekcije utječu vremenski uvjeti ili sposobnosti napadača dodjeljuju se odgovarajuće *DetectionProbability* definicije.

BlockingTime – vrijeme zadržavanja koje unosi postavljanje ovog zaštitnog elementa na gradbeni element. Kompleksnog je tipa *StDevStats*, čiji su članovi *MeanValue* koji predstavlja vrijeme zadržavanja u sekundama i *StandardDeviation* koji predstavlja relativno standardno odstupanje vremena zadržavanja i izražava se kao postotak.

Vjerojatnost detekcije definirana je klasom

```
DetectionProbability = {
    AdversarySneakingLevel : string,
    WeatherCondition : string,
    Value: double
  }
```
(3.4)

pri čemu je

AdversarySneakingLevel – sposobnost šuljanja napadača,

WeatherCondition – vremenski uvjeti i

Value – vjerojatnost detekcije za zadanu sposobnost šuljanja i vremenske uvjete.

Ovakvim modelom podržano je definiranje različitih sposobnosti detekcije zaštitnog elementa, ovisno o vrsti napadača i vremenskim uvjetima. Tablica 3.3 prikazuje različite vrijednosti detekcije za isti zaštitni element, ovisno o vremenskim uvjetima i sposobnostima napadača.

Tablica 3.3 Primjer vjerojatnosti detekcije zaštitnog elementa

Sposobnost napadača `AdversarySneakingLevel`	Vremenski uvjeti `WeatherCondition`	Vjerojatnost detekcije `Value`
Amater	Sunčano	0,96
Amater	Magla	0,85
Amater	Kiša	0,80
Profesionalac	Sunčano	0,20
Profesionalac	Magla	0,15
Profesionalac	Kiša	0,15

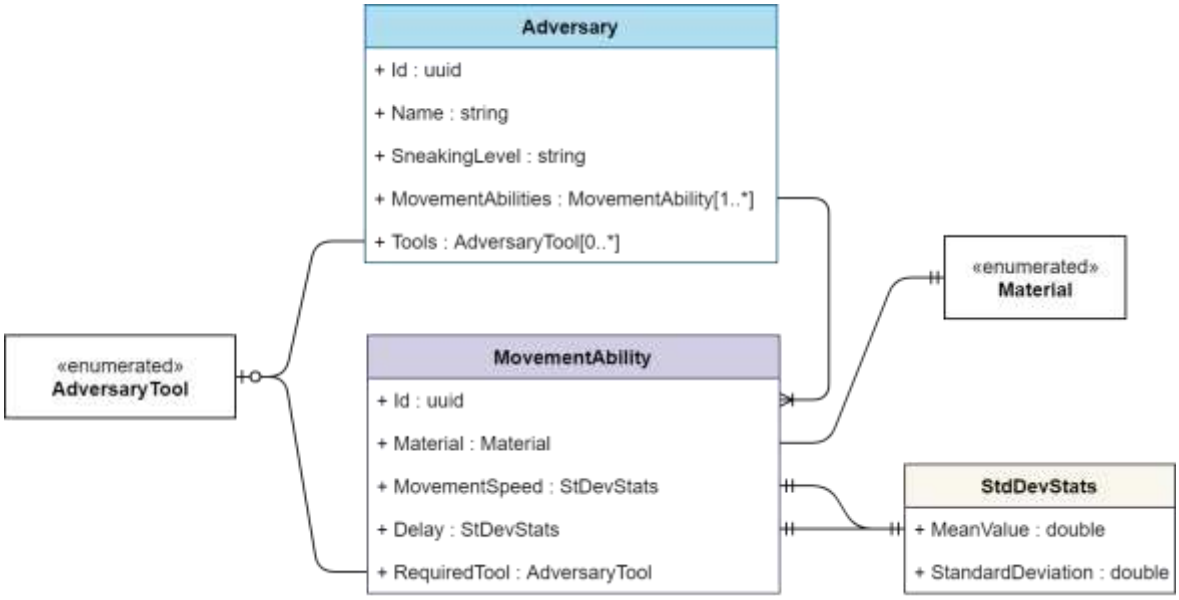
3.2. Modeliranje napadača

Slika 3.7 prikazuje model napadača kojim se definiraju očekivani tipovi napadača na štićeni objekt.

Napadač je definiran klasom

```
Adversary = {
     Id : uuid,
     Name : string,
     SneakingLevel : string,
     MovementAbilities : MovementAbility[1..*],
     Tools : AdversaryTool[0..*]
   }
```
(3.5)

pri čemu je



Slika 3.7 Model napadača

Id – jedinstveni identifikator napadača,

Name – naziv napadača,

SneakingLevel – sposobnost napadača u šuljanju, odnosno izbjegavanju detekcije. Koristi se za izračunavanje vjerojatnosti detekcije pri *DetectionProbability.Adversary-SneakingLevel*.

MovementAbilities – sposobnost kretanja napadača po različitim elementima štićenog objekta. Sposobnost kretanja ovisi o materijalu od kojeg je element građen pa napadač ima jedno ili više *MovementAbility* svojstva.

Tools – lista alata koje napadač ima na raspolaganju za izvršenje napada. Alat je definiran pobrojanim tipom *AdversaryTool* i može imati vrijednosti poput *Čamac, Helikopter, Eksploziv, Ljestve, Alat za obijanje brava* itd. Ovaj se podatak uzima u obzir pri *MovementAbility.RequiredAbility* ako kretanje nekim gradbenim elementom zahtijeva opremljenost odgovarajućim alatom.

Mogućnost kretanja napadača gradbenim elementom definirana je klasom

```
MovementAbility = {
     Id : uuid,
     Material : Material,
     MovementSpeed : StDevStats,
     Delay: StDevStats,
```
(3.6)

```
RequiredTool : AdversaryTool
}
```

pri čemu je

Id – jedinstveni identifikator mogućnosti kretanja napadača.

Material – materijal na koji se odnosi definicija mogućnosti kretanja napadača.

MovementSpeed – brzina kretanja napadača za zadani materijal i alat kojim je napad opremljen. Definira se složenim tipom *StDevStats* pri čemu je vrijednost *MeanValue* brzina izražena u metrima u sekundi, a kao *StandardDeviation* relativno standardno odstupanje brzine.

Delay – zadržavanje napadača na elementu. Definira se složenim tipom *StDevStats* pri čemu je vrijednost *MeanValue* vrijeme izraženo u sekundama, a *StandardDeviation* relativno standardno odstupanje zadržavanja.

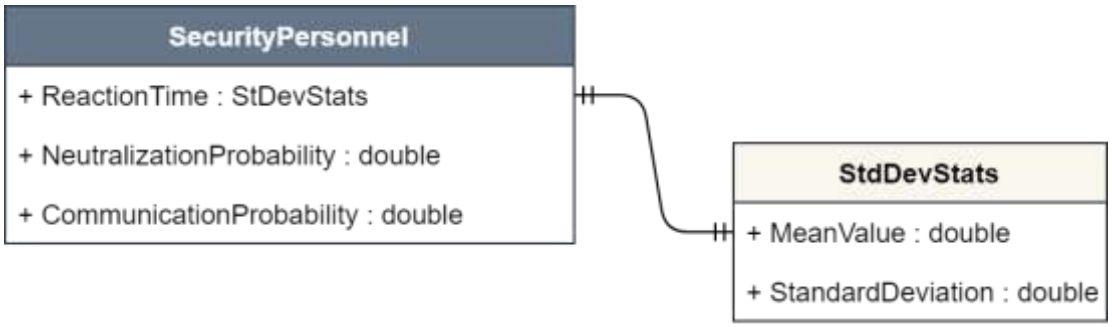
RequiredTool – određuje koji je alat potreban za ostvarenje definirane brzine kretanja i zadržavanja za odabrani materijal. Pobrojani tip vrijednosti ima vrijednost *BezAlata* za one definicije kretanja koje vrijede bez korištenja alata.

Tablica 3.4 Primjer definicije mogućeg kretanja

Parametar	Vrijednost
Id	02164d61-caa4-4c77-9565-51e06d891373
Material	ProtuprovalnaVrata
MovementSpeed	
MeanValue	1
StandardDeviation	30%
Delay	
MeanValue	120
StandardDeviation	60%
RequiredTool	AlatZaObijanjeVrata

3.3. Modeliranje tjelesne zaštite

Slika 3.8 prikazuje model tjelesne zaštite bez čije reakcije detekcija i zadržavanje napadača u kretanju prostorom nemaju učinka.



Slika 3.8 Model tjelesne zaštite

Tjelesna zaštita definirana je klasom

```
SecurityPersonnel = {
     ReactionTime : StDevStats,
     NeutralizationCapability : double,
     CommunicationProbability : double,
   }
```
(3.7)

pri čemu je

ReactionTime – vrijeme potrebno za reakciju zaštitne službe. Definira se složenim tipom *StDevStats* pri čemu je vrijednost *MeanValue* brzina izražena u metrima u sekundi, a kao *StandardDeviation* relativno standardno odstupanje brzine.

NeutralizationCapability – sposobnost neutralizacije napadača, uzevši u obzir osposobljenost, brojnost i opremljenost pripadnika tjelesne zaštite. Može se koristiti za procjenu s kojom vjerojatnošću će, ovisno o tipu napadača, napad biti uspješno neutraliziran.

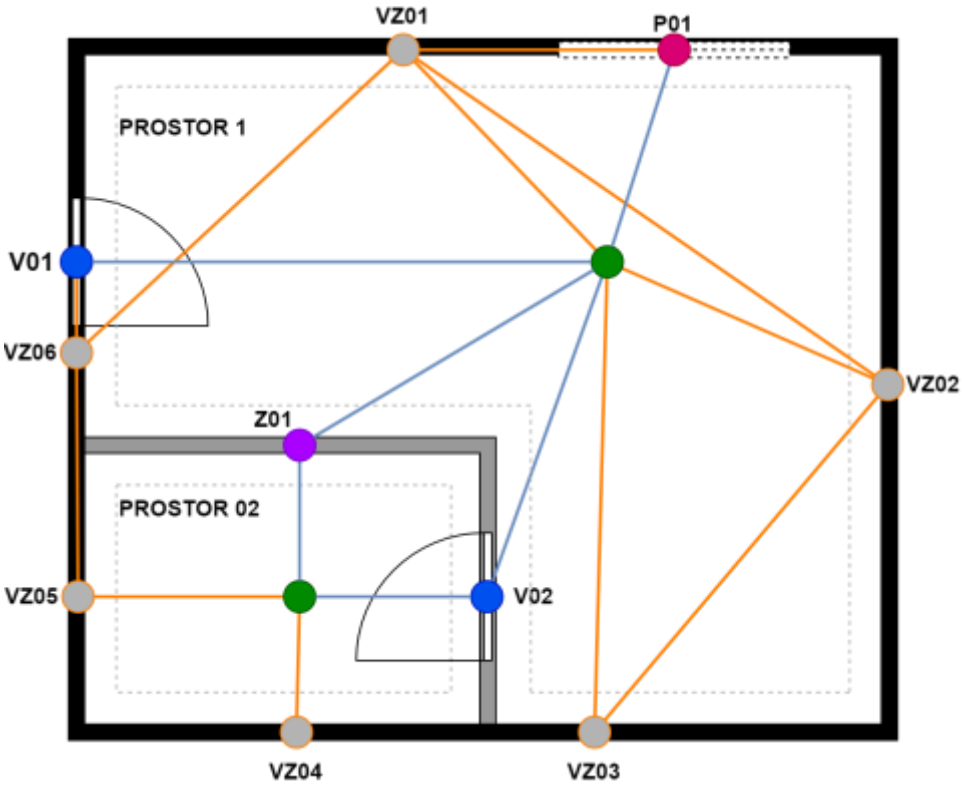
CommunicationProbability – vjerojatnost da će detekcija napadača biti uspješno prenesena tjelesnoj zaštiti i da će biti prepoznata kao stvarni alarm, a ne zanemarena, pretpostavljajući da se radi o lažnom alarmu.

3.4. Opisivanje mogućeg kretanja napadača

Modeliran štićeni objekt sadrži podatke o vrsti, poziciji i veličini pojedinog elementa prostora, materijalu od kojeg je element sačinjen te svojstvima napadača. Sljedeći je korak, na osnovi navedenih podataka, provesti geometrijske transformacije kojima se izrađuje model mogućeg kretanja napadača.

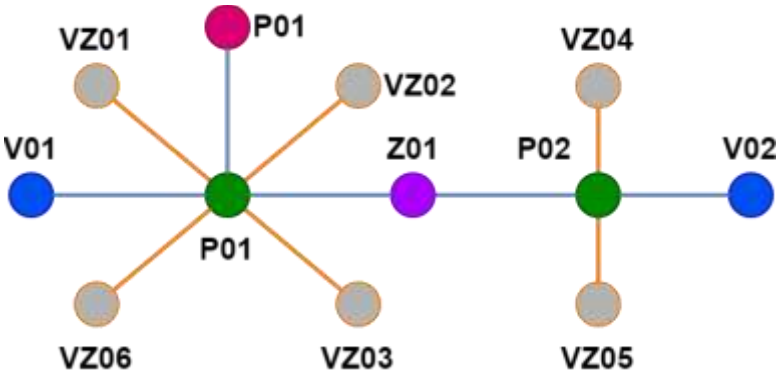
Uobičajeno se pri određivanju mogućeg kretanja osoba trodimenzionalno modeliranim prostorom za opisnu strukturu koristi graf. Pri tome se elementi objekta predstavljaju čvorovima, a susjedni se čvorovi povezuju vezama grafa. Ovako opisani graf predstavlja svojstvo susjednosti. Ovdje možemo razlikovati dvije vrste grafova, ovisno u koju se svrhu koriste. Graf povezanosti strukturnih elemenata postavlja veze među čvorovima koji su fizički povezani. Predstavlja osnovu za izradu druge vrsta grafa kojim se opisuje mogućeg kretanje za određeni tip napadača. U ovoj vrsti grafa veze među čvorovima uspostavljene su samo ako je kretanje tim putom fizički moguće za određeni tip napadača. Za razliku od neusmjerenog grafa povezanosti strukturnih elemenata, graf mogućeg kretanja je usmjeren.

Slika 3.9 prikazuje tlocrt prostora koji se sastoji od vanjskih zidova, označenih s VZxx, unutarnjih zidova, označenih s UZxx, prozora PZxx, vrata Vxx i prostora Pxx. Svakom strukturnom elementu pridijeljen je čvor, povezan vezama sa susjednim elementima. Jasno je da kretanje iz vanjskog zida VZ01 u prozor PZ01 koji mu pripada nema smisla niti u susjedni zid VZ02 pa takve veze izbacujemo kada izrađujemo graf povezanosti strukturnih elemenata.



Slika 3.9 Tlocrt prostora i pripadajući graf susjednosti elemenata objekta

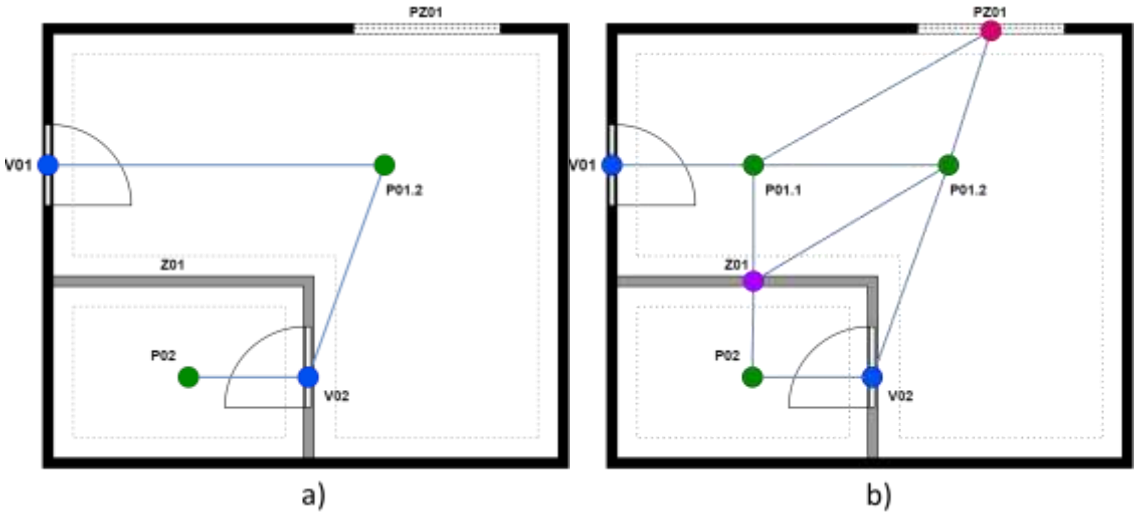
Slika 3.10 prikazuje graf povezanosti strukturnih elemenata za prethodno prikazan tlocrt (slika 3.9).



Slika 3.10 Graf povezanosti strukturnih elemenata objekta

Iz grafa povezanosti strukturnih elemenata izrađuju se grafovi mogućeg kretanja napadača, ovisno o njegovim sposobnostima. Slika 3.11a pokazuje primjer grafa mogućeg kretanja napadača koji je osposobljen savladati zaključana vrata V01 i V02, ali ne i prozor P01

zaštićen rešetkama te unutarnji zid UZ01. Slika 3.11b prikazuje graf mogućeg kretanja za napadača opremljenog alatima potrebnim za provaliti kroz prozor P01 i zid UZ01. Kako je vidljivo iz slike, graf mogućeg kretanja prostorom ima više čvorova u Prostoru 1 kako bi omogućio točniji izračun kretanja napadača. Takva raspodjela prostora je potrebna jer u prethodnom primjeru nije bilo potrebe za kretanjem *PZ01* → *Prostor01* → *UZ01* → *Prostor02*. Stoga graf mogućeg kretanja napadača može imati više čvorova i veza od grafa povezanosti strukturnih elemenata.



Slika 3.11 Graf mogućeg kretanja prostorom

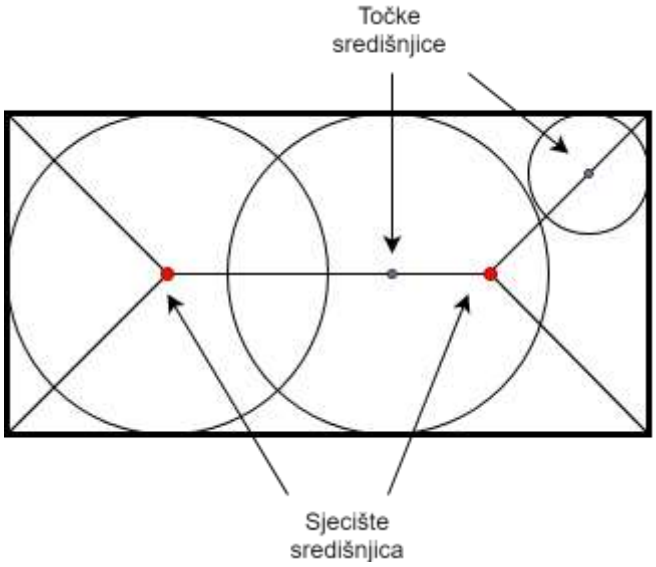
Algoritmi na različite načine određuju poziciju i broj čvorova. Jednostavna je metoda postavljanje čvorova u središnjicu elemenata kroz koje napadač može izvršiti prolazak, poput vrata, prozora i središnjice soba [26]. Ova je metoda primjenjiva za male prostorije, ali nije dovoljno dobra za prostore velikih površina ili izduženog oblika, poput dugih hodnika. U tim slučajevima kretanje napadača ne odgovara prirodnom kretanju i unosi preveliko odstupanje od realnog kretanja osobe. Metoda nije pogodna za prostore nepravilnog oblika jer poziciju čvora grafa postavlja u centroid pravokutnika koji opisuje prostor što može biti pozicionirano izvan samog prostora.

Precizniju simulaciju kretanja prostorom omogućuje metoda prekrivanja prostora kretanja mrežom kvadrata jednake veličine. Ovaj pristup, u svrhu numeričke analize ugroženosti, koristi *Systematic Analysis of Physical Protection Effectiveness* (SAPE) model [27] pri čemu se štićeni objekt prekriva kvadratnom mrežom pa svaki kvadrant postaje element grafa. Osnovna prednost ovog pristupa u odnosu na EASI, SAVI i ASSESS je realističniji prikaz

pozicije elemenata u prostoru. Nekoliko je nedostataka ovakvog pristupa podjele objekta na mrežu fiksnih kvadrata. Prostor se promatra dvodimenzionalno pa postupak ne podržava modeliranje objekata koji se nalaze na neravnim terenima, odnosno ne uzima u obzir izračun udaljenosti u trodimenzionalnom prostoru. Preciznost izračuna ovisi o veličini kvadranta mreže. On mora biti dovoljno mali da opiše najmanji element tlocrta, poput debljine zida. Time se stvara izrazito velik broj elemenata grafa što otežava pretraživanje i optimiranje u prihvatljivom vremenu. Autori SAPE postupka testirali su utjecaj veličine kvadranta mreže na preciznost izračuna vjerojatnosti prekida napada. Pri tome su koristili grubu podjelu objekta, s najmanjim elementom veličine 11 metara. Rezultati su pokazali da veličina stranice kvadranta približna najmanjem elementu, do 15 metara, ne utječe znatno na kvalitetu rezultata, međutim daljnje povećanje rezolucije mreže ne daje dobre rezultate.

Manji broj čvorova i veza postiže se metodama dijeljenja prostora na četvrtine za dvodimenzionalne prostore, odnosno dijeljenje na oktaedre upotrebljava li se u trodimenzionalnom prostoru. Osnovni je princip rekurzivna podjela prostora na četiri, odnosno osam dijelova, do unaprijed definirane minimalne veličine, sve dok se u prostoru ne nalaze elementi koji onemogućuju kretanje. Time se postiže podjela prostora na kvadrante različitih veličina, ovisno o topologiji prostora. Metoda stvara nepotrebne čvorove pa nije pogodna za optimiranje. Navedeni modeli zahtijevaju dodatnu optimizaciju kojom se izbacuju nepotrebni čvorovi kako bi graf bio iskoristiv za optimiranje. Uz dijeljenje prostora na kvadratne elemente, dostupni su algoritmi koji prostor dijele nepravilnim oblicima, poput trokuta i trapezoida te Voronoi-evim poligonima [28].

Naprednija je metoda izrade grafa uporabom transformacije središnjicom (*engl. Medial Axis Transformation* – MAT) [29]. Za uobičajene građevine pozicije čvorova ove metode prirodnije su pozicionirane u odnosu na ranije navedene metode i manji ih je broj. Metoda se može koristiti za dvodimenzionalne i trodimenzionalne prostore. Rezultat metode je tzv. kostur prostora, kako prikazuje slika 3.12.



Slika 3.12 Transformacija metodom središnjice (MAT)

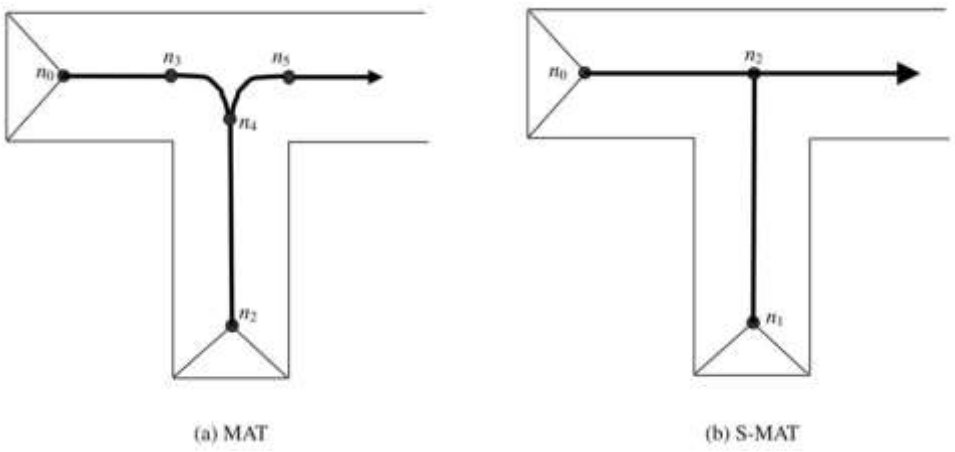
Dva su načina provedbe transformacije, odnosno definiranja središnjice. Jedan definira središnjicu kao skup točaka unutar poligona koje su jednako i najbliže udaljene od dva ili više rubova. Druga metoda je istovremeno pomicanje rubova prema unutrašnjosti poligona, pri čemu sjecišta čine središnjice. Čvorove postavljamo na sjecišta središnjica. Osnu transformaciju O možemo definirati tako da jednadžbe (3.8) i (3.9) vrijede za bilo koju točku $\mathcal{X}$ na rubu objekta koji ima jedinstvenu okomicu.

$$\phi[\mathrm{X}_i(x,y,z)] = \phi\big[\mathrm{X}_j(x,y,z)\big] = C(x,y,z) \tag{3.8}$$

$$|C - \mathrm{X}_i| = \big|C - \mathrm{X}_j\big| \tag{3.9}$$

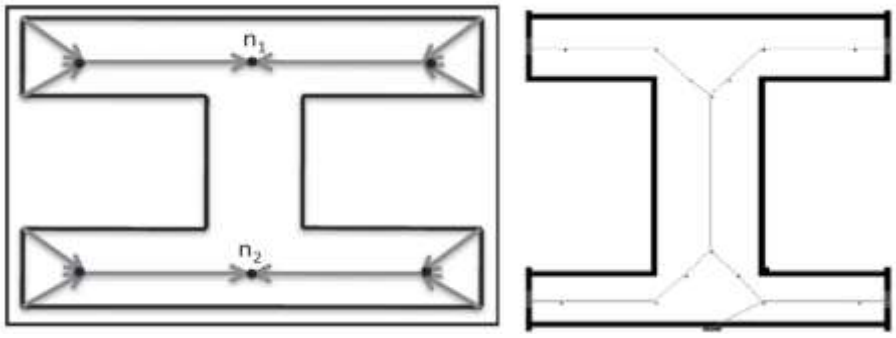
pri čemu je $C(x,y,z)$ centar diska za dvodimenzionalne prostore i centar sfere za trodimenzionalne prostore, s tangentama na točkama X_i i X_j smještenima na rubu poligona.

Nedostatak MAT metode pri izradi grafa kretanja je mogućnost nastanka parabola u određenim konfiguracijama prostora, a posebno u hodnicima koji nalikuju slovu T, (slika 3.13a). Iz tog razloga razvijena je naprednija metoda transformacije ravnim središnjicama (*engl. Straight-Medial Axis Transformation* – S-MAT) [30] kojom se izbjegava nastanak parabola i smanjuje broj čvorova (slika 3.13b).



Slika 3.13 Usporedba MAT i S-MAT metoda (preuzeto iz [30])

Nedostatak S-MAT metode je nemogućnost spajanja čvorova u prostorima oblika slova H (slika 3.14 a). Uvedena je novija, *Modified-Medial Axis Transformation* (M-MAT) [31] koja rješava taj nedostatak.

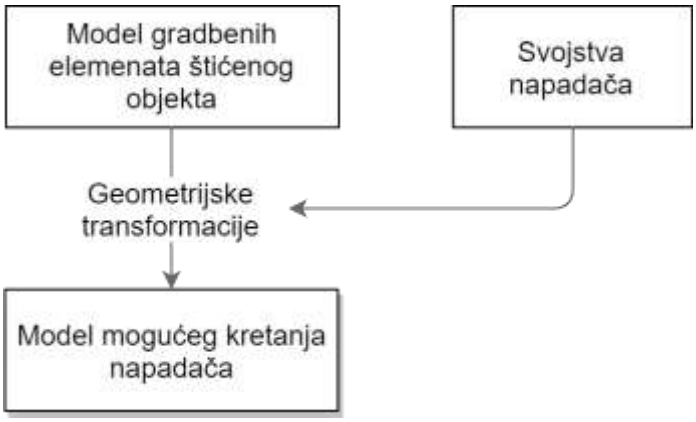


Slika 3.14 Usporedba S-MAT i M-MAT metode (preuzeto iz [32])

Opisani algoritmi temeljeni na metodi transformacije središnjicom koriste se u kompleksnijim transformacijama opisa građevine unutarnjeg i vanjskog prostora u graf kretanja [33], [25]. Izazov u opisivanju prostora grafom mogućeg kretanja je podjela većih i izduženih prostora poput hodnika [34].

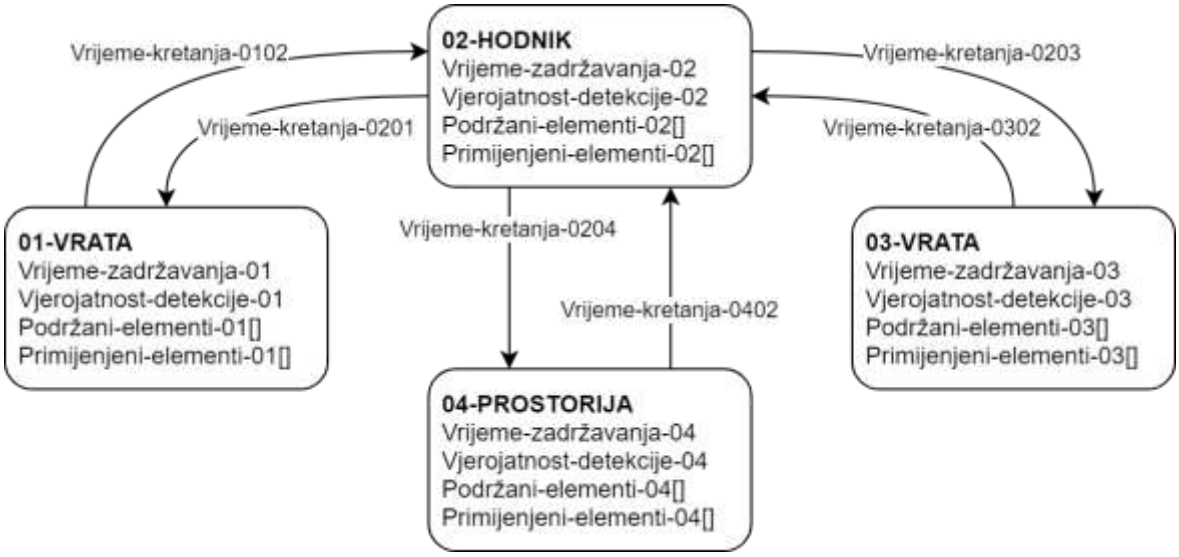
3.5. Modeliranje mogućeg kretanja napadača

Model mogućeg kretanja napadača opisuje sve napadaču dostupne putove kretanja štićenim objektom te elemente zaštite koji mogu biti primijenjeni ili su već primijenjeni nad elementima kretanja. Kako prikazuje slika 3.15, na osnovu modela štićenog objekta, koji sadrži vrstu, poziciju, veličinu i materijal pojedinog elementa prostora te modela napadača, mogu se izvršiti geometrijske transformacije kojima se pronalaze mogući putovi kretanja napadača. Tim se podacima popunjava model logičke razine štićenog objekta.



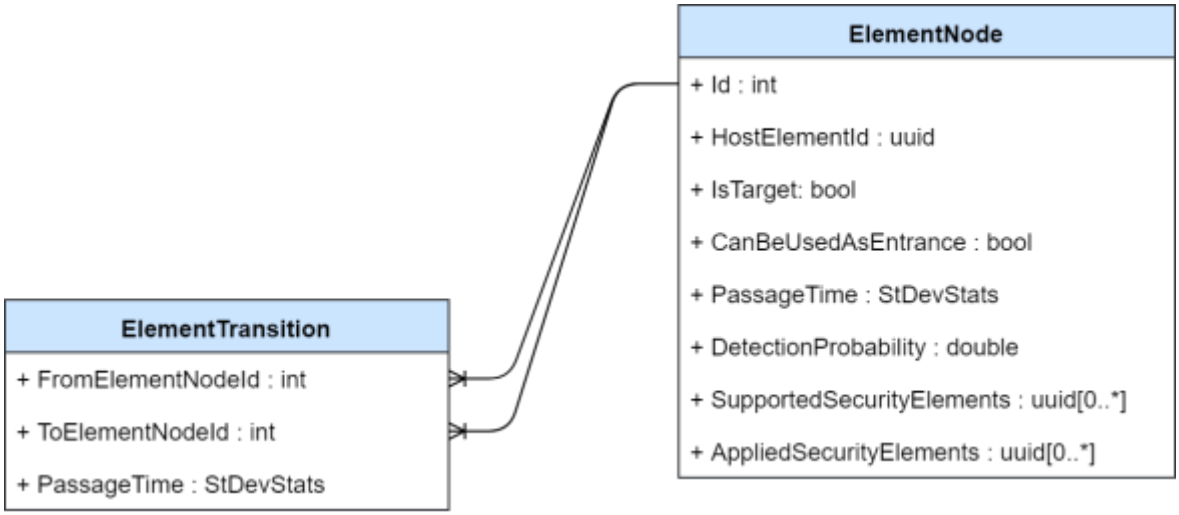
Slika 3.15 Izgradnja modela logičke razine štićenog objekta

Slika 3.16 prikazuje grafički primjer modela logičke razine štićenog objekta kojeg čine četiri elementa i njima pridijeljena svojstva. Čvorovi su povezani vezama koje predstavljaju mogućnosti kretanja napadača, pri čemu svaka veza određuje smjer mogućeg kretanja i vrijeme kretanja potrebno napadaču za prelazak između početnog i završnog čvora.



Slika 3.16 Graf kretanja opisan logičkim modelom štićenog objekta

Model se sastoji od dvije osnovne klase koje opisuju graf mogućeg kretanja napadača. Slika 3.17 prikazuje komponente modela.



Slika 3.17 Model grafa mogućeg kretanja napadača

Model *ElementNode* predstavlja sve čvorove grafa kretanja. Može se prikazati kao

```
ElementNode = {
        Id : int,
        HostElementId : uuid,
        CanBeUsedAsEntrance : bool,
        IsTarget : bool,
        AppliedSecurityElements : uuid[0..*],
```
(3.10)

```
                PassageTime : StDevStats,
                DetectionProbability : double,
                SupportedSecurityElements : uuid[0..*]
```

pri čemu je

Id – jedinstveni identifikator čvora.

HostElementId – identifikator elementa *HostElement* u modelu štićenog objketa na koji se čvor odnosi.

CanBeUsedAsEntrance – određuje može li čvor biti korišten kao početak napada, odnosno ulaska u štićeni objekt.

IsTarget – određuje predstavlja li čvor prostor koji je cilj napada.

AppliedSecurityElements – lista identifikatora zaštitnih elemenata koji su dodijeljeni ovom elementu štićenog objekta.

PassageTime – vrijeme potrebno napadaču za savladavanje prostora predstavljenog ovim čvorom. Kompleksnog je tipa *StDevStats*, čiji su članovi *MeanValue* koji predstavlja vrijeme savladavanja u sekundama i *StandardDeviation* koji predstavlja relativno standardno odstupanje vremena savladavanja i izražava se kao postotak.

DetectionProbability – vjerojatnost detekcije napadača. Ovisi o zaštitnim elementima dodijeljenim elementu.

SupportedSecurityElements – lista identifikatora zaštitnih elemenata koji se mogu tijekom optimizacije dodijeliti ovom elementu.

Veze grafa, koji povezuju *ElementNode* čvorove, predstavljeni su modelom *ElementTransition* koji se može prikazati kao

```
ElementTransition = {
                        FromElementNodeId : int,
                        ToElementNodeId : int,
                        PassageTime : StDevStats
                    }
```
(3.11)

pri čemu je

FromElementNodeId – identifikator početnog čvora tranzicije.

ToElementNodeId – identifikator ciljnog čvora tranzicije.

PassageTime – vrijeme kretanja od početnog čvora *FromElementNodeId* do ciljnog čvora *ToElementNodeId.*

Vrijeme kretanja između čvorova ovisi o njihovoj udaljenosti i brzini kretanja napadača.

Neka su pozicije dva elementa A i B u trodimenzionalnom prostoru definirane s $A(x_1, y_1, z_1)$, $B(x_2, y_2, z_2)$. Tada je njihova udaljenost d:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{3.12}$$

Za brzinu kretanja koristi se vrijednost definirana u *MovementAbility.MovementSpeed* i možemo je izraziti kao:

$$v = max\,(f(Material, Adversary)) \tag{3.13}$$

odnosno kao funkciju koja traži *MovementAbility* s maksimalnom vrijednošću *MovementSpeed* za zadani materijal *Material* kojim se napadač kreće i alat *RequiredTool* kojim je napadač opremljen. Vrstom napadača definirano je kojim vrstama alata raspolaže (*Adversary.Tools)* i njegove sposobnosti kretanja (*Adversary.MovementAbilities)*. Slika 3.18 prikazuje pseudo kod ovog procesa.

```
Odabir_maksimalne_brzine_kretanja(hostElement.Material, adversary) {
  brzinaKretanja = 0;
  za svaki (movementAbility u adversary.MovementAbilities) {
    ako (hostElement.Material == movementAbility.Material &&
         adversary.Tools sadrži movementAbility.RequiredTool &&
         movementAbility.MovementSpeed > brzinaKretanja)
    {
      brzinaKretanja = movementAbility.MovementSpeed;
    }
  }
  vrati brzinaKretanja;
}
```

Slika 3.18 Algoritam odabira brzine kretanja

Za udaljenost između čvorova *d*, prema izrazu (3.12), i brzinu kretanja napadača v, prema izrazu (3.13), vrijeme kretanja *t* između čvorova računamo

$$t = \frac{d}{v} \tag{3.14}$$

Predstavljeni model logičke razine štićenog objekta osnova je za stvaranje prostora pretraživanja nad kojim se izvodi optimiranje sustava zaštite.

4. OPTIMIRANJE ODABIRA I RASPOREDA ZAŠTITNIH ELEMENATA U PROJEKTIRANJU SUSTAVA TEHNIČKE ZAŠTITE

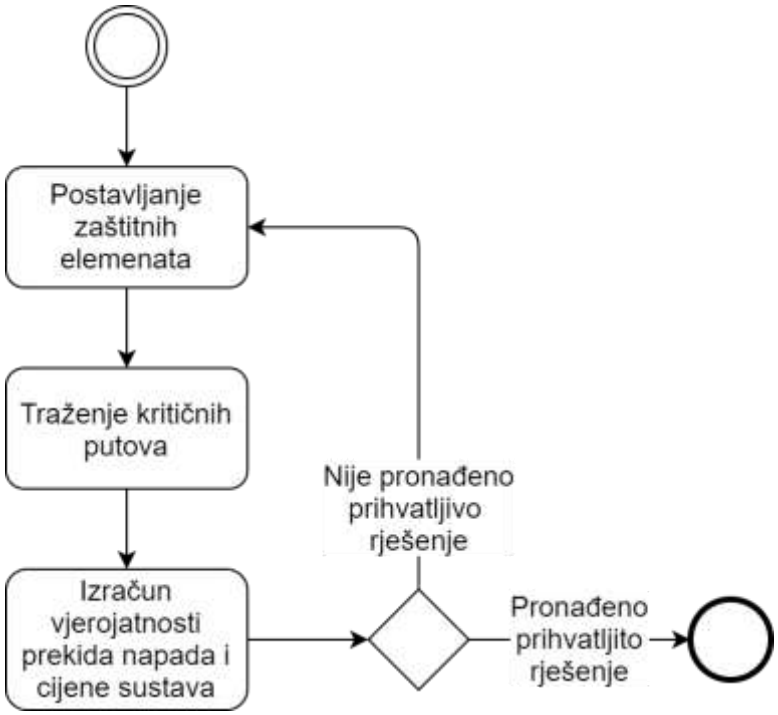
Definicija numeričke analize ugroženosti i objektno modeliranje koncepta zaštite osnova su za određivanje funkcija cilja te odabira i rasporeda zaštitnih elemenata u projektiranju sustava tehničke zaštite postupkom optimiranja funkcije cilja. Funkcija cilja definirana je kao višekriterijska funkcija, gdje su kriteriji vjerojatnost pravovremene detekcije napadača i cijena zaštitnih elemenata. Budući da su kriteriji suprotni, kao rezultat optimiranja ne dobije se jedno rješenje nego skup rješenja, ovisan o iznosu pojedinog kriterija. Dobivena rješenja iz optimalnog skupa određuju izbor i raspored zaštitnih elemenata. Time se projektantu pruža izbor odabira onog rješenja koje smatra primjerenim.

Uz višekriterijsko optimiranje, poglavlje donosi i dodatna optimiranja koja predstavljaju doprinose ovog rada, a zasnovana su na poznavanju domene. Prvo optimiranje provodi se uvedenom hibridnom metodom koja smanjuje prostor pretraživanja uklanjajući nepotrebne elemente, a zatim predlaže početna rješenja algoritmima optimiranja, čime smanjuje broj iteracija potrebnih za nalaženje kvalitetnih nedominiranih rješenja. Drugi je doprinos izrada algoritma kojim se izbjegavaju nepotrebni izračuni kvalitete i cijene rješenja u procesu evaluacije, korištenjem znanja stečenog prethodnim pretraživanjem prostora rješenja.

4.1. Proces optimiranja

Cilj optimiranja je odrediti koje zaštitne elemente odabrati, kako bi se za odabranu cijenu sustava postigla maksimalna vjerojatnost zaustavljanja napadača. Slika 4.1 prikazuje osnovni proces optimiranja koji se sastoji od postavljanja zaštitnih elemenata na elemente koji napadaču omogućuje kretanje objektom, nakon čega slijedi traženje kritičnih putova napada za sve elemente s kojih napadač može započeti napad te ocjena kvalitete predloženog

rješenja. Ukoliko pronađeno rješenje ne zadovoljava traženu kvalitetu, proces se ponavlja sve do pronalaska zadovoljavajućeg rješenja.



Slika 4.1 Proces optimiranja

Ovom se metodom služe projektanti sustava zaštite u procesu projektiranja, ali proces nije automatiziran već se zasniva na iskustvu projektanta. Građevinski kompleksi sastoje se od velikog broja elemenata kojima se napadač može kretati i dostupan je velik broj elemenata zaštite. Iz tog razloga projektant ne može sagledati sve dostupne kombinacije zaštite pa je velika vjerojatnost da je rješenje koje predloži, oslanjajući se na iskustvo, daleko od optimalnog rješenja.

Proces optimiranja razlikuje se ovisno o preduvjetima koje projektant postavlja na sustav. Ukoliko projektant unaprijed definira maksimalnu vrijednost investicije ili minimalnu vjerojatnost prekida napada, tada se problem može svesti na funkciju jednog cilja. Drugi je pristup optimiranju provođenje pretraživanja mogućih rješenja prije donošenja odluke o investiciji ili traženoj vjerojatnosti prekida napada. Rezultat ovakvog procesa optimiranja je skup rješenja, pri čemu svako postiže maksimalnu vjerojatnost zaustavljanja za minimalnu cijenu. Projektant može odabrati ono koje zadovoljava unaprijed zadan minimalan iznos vjerojatnosti zaustavljanja napadača ili maksimalnu cijenu.

Jednostavna analiza sustava zaštite u svrhu optimiranja je prikazana u [35] gdje se napad analizira koristeći modelirajuću analizu stabla kvarova za identifikaciju mogućih scenarija napada, a analizu stabla događaja za moguće konzekvence uspješnog napada. U [36] se optimiranje provodi genetskim algoritmima na razini procjene ugroženosti što nije dovoljno detaljno rješenje za izradu projekta tehničke zaštite. Isti autor u [37] koristi Petrijeve mreže za dizajn tehničkog sustava zaštite, ali bez razrade automatskog optimiranja. Pokušaj detaljnijeg optimiranja prikazan je u [38] gdje autor koristi genetski algoritam za optimizaciju zaštite, ali se ne odabiru zaštitni elementi već se određuje odnos raspodjela budžeta i vjerojatnosti detekcije kojima se postiže optimalna zaštita. Pri tome autor optimira samo pronađene kritične putove, ne uzimajući u obzir da se postavljanjem zaštitnih elemenata uvodi mogućnost nastajanja novih kritičnih putova.

Ove metode su primjenjive kao pomoć projektantu u procesu oblikovanja sustava zaštite, ali ne mogu samostalno pronaći optimalno rješenje niti predložiti konkretne elemente zaštite kojima će se postići traženo.

Naprednija metoda optimiranja, predložena u ovom radu, uzima u obzir sve dostupne elemente za početak napada i svojstva napadača te za rezultat pruža popis elemenata zaštite i njihove pozicije čime se postiže gotovo optimalna zaštita za svaki cjenovni razred.

4.1.1. Određivanje kriterijskih funkcija

Neka je $G = \{\mathcal{V}, \mathcal{E}\}$ usmjereni graf, pri čemu je $\mathcal{V}$ skup čvorova, s ukupnim brojem elemenata $|\mathcal{V}| = n$, pri čemu je $n > 1$, a $\mathcal{E}$ je skup veza, ukupnog broja $|\mathcal{E}| = m$.

Čvor grafa predstavlja element mogućeg kretanja napadača (3.10) u modelu logičke razine štićenog objekta. Neka je $\mathcal{S}$, pri čemu je $\mathcal{S} \subset \mathcal{V}(G)$, skup čvorova koji se mogu koristiti kao element početka napada. To su u *modelu logičke razine štićenog objekta* elementi mogućeg kretanja napadača čije svojstvo `CanBeUsedAsEntrance` ima vrijednost *true*. Neka je $\mathcal{T}$, pri čemu je $\mathcal{T} \subset \mathcal{V}(G)$, skup čvorova koji predstavljaju cilj napadača. U modelu logičke razine štićenog objekta njihovo je svojstvo `IsTarget` aktivno.

Čvor ima sljedeća svojstva:

$dp(v)$ - vjerojatnost detekcije.

$pt(v)$ - vrijeme potrebno za savladavanje elementa.

$sn(v)$ - standardna devijacija vremena potrebnog za prolazak elementom.

$ae(v)$ - skup zaštitnih elemenata koji se mogu pridijeliti elementu kretanja štićenim objektom.

$ap(v)$ - skup zaštitnih elemenata primijenjenih na elementu kretanja štićenim objektom.

$co(v)$ - cijena zaštitnih elemenata primijenjenih na elementu kretanja štićenim objektom.

Veze predstavljaju mogućnost kretanja između čvorova. Definiramo ih kao $e_{ij} = (v_i, v_j)$, pri čemu je $e_{ij} \in \mathcal{E}$. Veze predstavljaju u modelu logičke razine štićenog objekta element *ElementTransition* (3.13).

Veza ima sljedeća svojstva:

$pt(e_{ij})$ – vrijeme potrebno za prelazak iz čvora v_i u čvor v_j,

$sn(e_{ij})$ – standardna devijacija vremena potrebnog za prelazak iz čvora v_i u čvor v_j.

Za svaki član skupa $\mathcal{S}$ možemo pronaći put

$$p_i := (v_1 e_{12} v_2 \dots v_{N-1} e_{N-1\,N} v_N) \tag{4.1}$$

tako da vrijedi

$$f(p_i) = f(v_1 e_{12} v_2 \dots v_{N-1} e_{N-1\,N} v_N) = \min(P_I) \tag{4.2}$$

pri čemu je P_I vjerojatnost da će napadač biti detektiran tijekom kretanja putom p_i i spriječen prije dolaska do ciljnog čvora $v_N \in \mathcal{T}$.

Vjerojatnost prekida napada može se definirati kao

$$P_I = P_{R1} \cdot dp(v_1) \cdot P_C + \sum_{i=2}^{N} P_{Ri} \cdot dp(v_i) \cdot P_c \cdot \prod_{j=1}^{i-1} (1 - dp(v_j) \cdot P_c) \tag{4.3}$$

pri čemu je:

P_{Ri} – vjerojatnost da će tjelesna zaštita stići do napadača prije nego napadač savlada put od čvora v_i do ciljnog čvora v_N,

P_C – vjerojatnost da će komunikacija biti uspješna prilikom detekcije i za vrijeme intervencije tjelesne zaštite.

Funkcija vjerojatnosti P_{Ri} određena je izrazom:

$$P_{Ri} = E(T_{ui} > 0, s_n) = \int_0^{\infty} \frac{1}{s_{ni}\sqrt{2\pi}} e^{\frac{(t-T_{Ui})^2}{2s_{ni}^2}} dt \tag{4.4}$$

pri čemu je

$$T_{Ui} = T_G - T_{Ri} . \tag{4.5}$$

T_{Ui} je razlika u vremenu T_G, potrebnom tjelesnoj zaštiti da stigne do ciljnog čvora v_N nakon što je napadač detektiran i vremenu T_{Ri}, potrebnom napadaču da stigne od čvora v_i do ciljnog čvora v_N. T_{Ri} se računa kao

$$T_{Ri} = \sum_{j=i}^{N} pt(x) \tag{4.6}$$

pri čemu je x čvor na putu $\left(v_i e_{i,i+1} v_{i+1} \dots v_{N-1} e_{N-1,i} v_N\right)$.

Standardna devijacija s_{ni} razlike vremena T_{Ui} jednaka je

$$s_{ni}(T_{Ui}) = s_{ni}(T_{Ri} - T_G) = \sqrt{s_n^2(T_{Ri}) + s_n^2(T_G)} \tag{4.7}$$

Cijena sustava. Određena je kao zbroj cijena svih zaštitnih elemenata primijenjenih na elementima kretanja objektom, neovisno o tome pripadaju li kritičnom putu napada:

$$c = \sum_{i=1}^{N} \mathrm{co}(v_i), \tag{4.8}$$

Optimiranje zaštitnog sustava izvodimo odabirom zaštitnih elemenata iz skupa $ae(x)$, pri čemu je x element skupa $\mathcal{V}$. Ukupan broj zaštitnih elemenata za odabir je

$$m = \sum_{i=1}^{n} |ae(v_i)| \tag{4.9}$$

pri čemu je $n = |\mathcal{V}|$.

Pri odabiru zaštitnih elemenata želimo istovremeno postići dva cilja. Prvi je pronaći zaštitne elemente za koje vrijedi da je cijena sustava c minimalna. Drugi je cilj da se odabranim

zaštitnim elementima postiže maksimalna vjerojatnost prekida napada P_I. Ovo je višekriterijski problem, a ukupan broj mogućih rješenja za svaki tip napadača je

$$N = |\mathcal{S}| \cdot 2^m. \tag{4.10}$$

Uzme li se za primjer štićeni objekt koji se sastoji od osamdeset elemenata kretanja na koje je moguće postaviti po dva zaštitna elementa, pri čemu napadač može započeti napad s pet pozicija, prema izrazu (4.9) potrebno je izvršiti $5\times2^{80\times2}$ ili 1.46×10^{48} evaluacija za izračun svih mogućih kombinacija. Jasno je da takav pristup nije iskoristiv za ovaj problem.

4.1.1.1 Jednokriterijska funkcija

Istovremeno traženje rješenja u kojem je se želi minimizirati cijena, a maksimizirati zaštita čine ovaj problem višekriterijskim. Jedan od pristupa rješavanju problema je kombiniranje ova dva kompetitivna zahtjeva u jednu funkciju, kojom se traži skalarna vrijednost.

$$F(x) = \min \sum_{i=1}^{k} w_i f_i\,(x)\,, x \in S \tag{4.11}$$

Takvu funkciju nazivamo agregatnom, a osnovna je ideja da se svakom cilju optimiranja f_i pridijeli težinski faktor $w_i \geq 0$ čime se utječe na njegovu važnost u konačnom rješenju.

Primjer takve funkcije u svrhu optimiranja sustava zaštite je

$$F(x) = 1 - \sqrt{w_C \left(\frac{c(x) - \min\limits_N(c)}{\max\limits_N(c) - \min\limits_N(c)} \right)^2 + w_P \left(\frac{\overline{P_I}(x) - \min\limits_N(\overline{P_I})}{\max\limits_N(\overline{P_I}) - \min\limits_N(\overline{P_I})} \right)^2} \tag{4.12}$$

pri čemu je

w_C - težinski faktor cijene rješenja,

$c(x)$ - cijena rješenja x za koje izračunavamo kvalitetu,

$\min\limits_N(C)$ - minimalna cijena u skupu *N* rješenja,

$\max\limits_N(C)$ - maksimalna cijena u skupu *N* rješenja,

w_P - težinski faktor vjerojatnosti uspješnosti napada

$\overline{P_I}(x)$ - vjerojatnost uspješnosti napada ($\overline{P_I} = 1 - P_I$) za rješenje x,

$\min_N(\overline{P_I})$ - minimalna vjerojatnost da će napad uspjeti,

$\max_N(\overline{P_I})$ - maksimalna vjerojatnost da će napad uspjeti.

Ova funkcija postići će maksimalnu vrijednost uz minimalnu cijenu rješenja i minimalnu vjerojatnost uspješnosti napada, odnosno maksimalnu vjerojatnost prekida napada.

Takav pristup svođenja višekriterijskog problema na skalarnu funkciju cilja ima više nedostataka [39]. Težinski koeficijenti, koji se koriste u definiranju funkcije cilja, utječu na rezultat optimiranja. Obično je potrebno eksperimentirati s njihovim odnosima kako bi se postigla zadovoljavajuća kvaliteta rješenja. Proces eksperimentiranja zahtijeva višestruko ponavljanje algoritma optimiranja [40]. Dodatni nedostatak je da kod nekih problema mala promjena koeficijenata izaziva velike promjene u rezultatima [41]. Rezultat takvog optimiranja, koje kombinira dva cilja u jedno rješenje, ograničuje projektanta u izboru i onemogućuje donošenje informirane odluke.

4.1.1.2 Višekriterijska funkcija

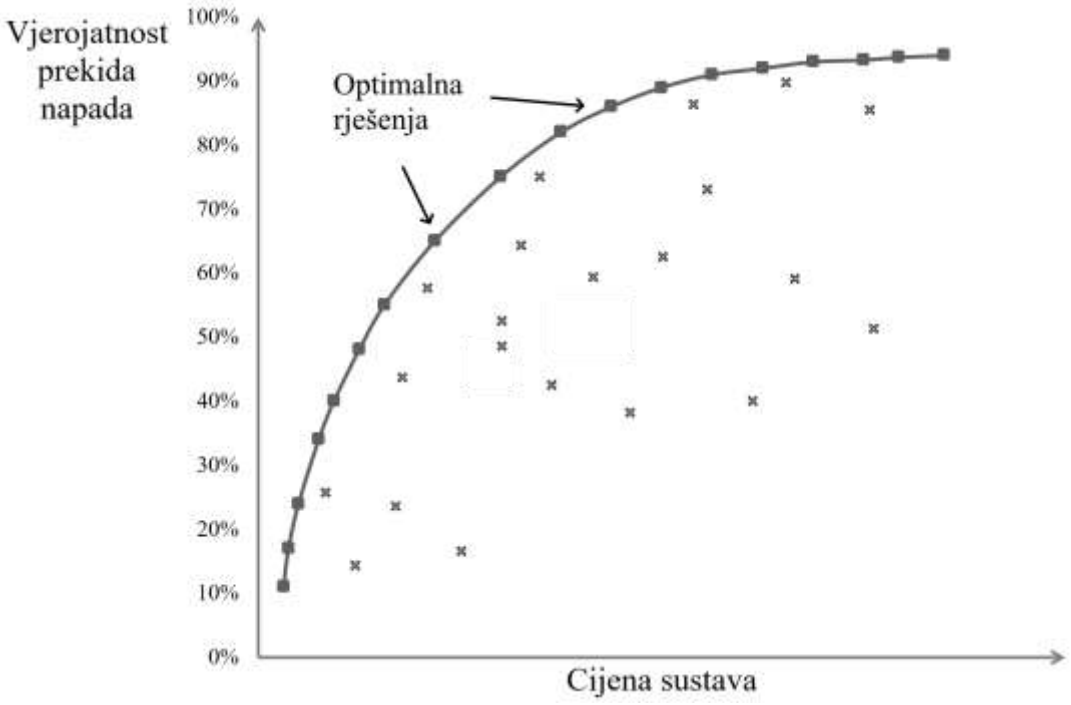
Primjena višekriterijske funkcije, u odnosu na agregatnu funkciju, omogućuje pronalaženje skupa prihvatljivih rješenja unutar prostora pretraživanja:

$$P^* = \{x \in \Omega \mid \neg\exists x' \in \Omega, \mathrm{F}(x') \leq F(x)\} \tag{4.13}$$

Takav se skup rješenja naziva Pareto-optimalnim ili skupom nedominiranih rješenja.

$$f(x) = \big(F(x), cost(x)\big) \tag{4.14}$$

Slika 4.2 prikazuje skup rješenja, pri čemu su optimalna rješenja označena kvadratom, a neoptimalna su prikazana križićem.



Slika 4.2 Pareto fronta u optimizaciji sustava tehničke zaštite

F je kriterijska funkcija i predstavlja kvalitetu rješenja, temeljenu na vjerojatnosti da će napadač biti spriječen u napadu. Kako je $|\mathcal{S}| \geq 1$, odnosno može postojati više elemenata koji napadaču mogu poslužiti za početak napada, jednostavan je pristup izračunom aritmetičke sredine vjerojatnosti za sve putove. Nedostatak ovakvog pristupa su slučajevi u kojima pojedini putovi poprimaju vrijednost vjerojatnosti P_I jednaku nuli, što znači da će napadač sigurno postići cilj napada. Kada bismo $F(x)$ izračunavali kao srednju vrijednost vjerojatnosti prekida napada za svaki početni element

$$F(x) = \frac{\sum_{i=1}^{|\mathcal{S}|} P_I(x_i)}{|\mathcal{S}|} \tag{4.15}$$

tada bi $F(x)$ za putove čiji P_I iznose 0% i 90% predstavljalo bolje rješenje ($F(x) = 0.45$) u odnosu na putove čiji P_I iznose 30% i 40%, za koje je $F(x) = 0.35$. Međutim, rješenje u kojem jedan ili više mogućih putova napada omogućuje napadaču prolazak s vjerojatnošću 100% nije prihvatljivo. Iz tog razloga, uvodi se kazna za putove čija je P_I jednaka nuli, na način da se za njihovu kvalitetu rješenja $g(x_i)$ postavlja negativna vrijednost ukupnog broja početnih elemenata.

$$F(x) = \frac{\sum_{i=1}^{|\mathcal{S}|} \mathrm{g}(x_i) \begin{cases} P_I(x_i), & P_I(x_i) > 0 \\ -|\mathcal{S}|, & P_I(x_i) = 0 \end{cases}}{|\mathcal{S}|} \tag{4.16}$$

Uvođenje kazne na način definiran u (4.16) omogućuje usporedbu i loših, neprihvatljivih rješenja što pomaže u procesu optimiranja. Tako $F(x)$ za putove čiji P_I iznose 0% i 90% predstavlja lošije rješenje ($F(x) = -0.55$) u odnosu na putove čiji P_I iznose 30% i 40%, za koje je $F(x) = 0.35$, ali bolje od putova s P_I 0% i 0% za koje je $F(x) = -1$.

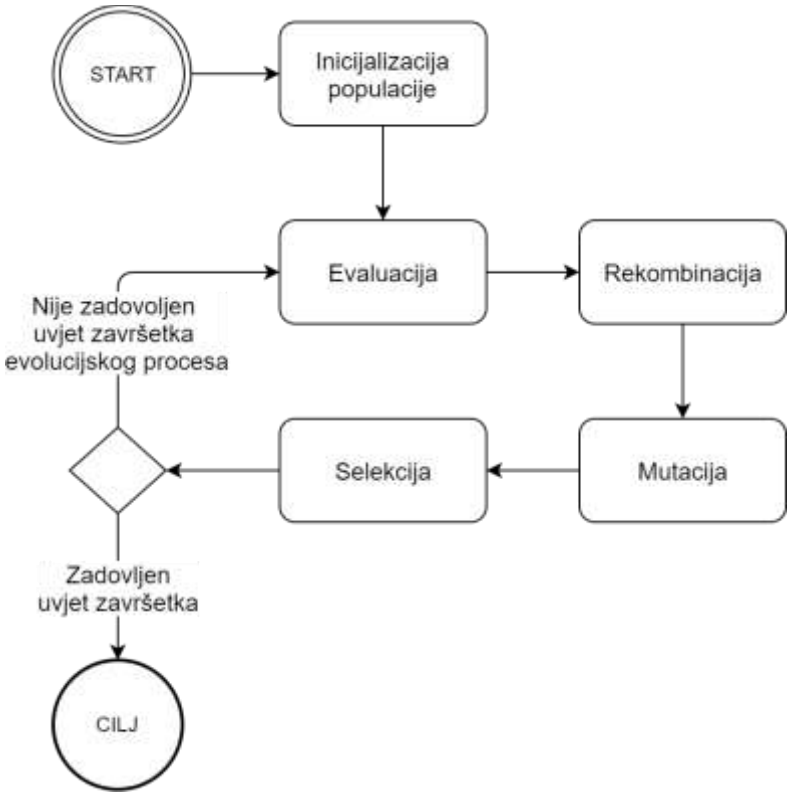
4.1.2. Optimiranje dinamičkim programiranjem

Numerička analiza sustava tehničke zaštite zasniva se na rješavanju potproblema, kojeg predstavlja savladavanje pojedinog elementa na putu kretanja napadača te se rješenja potproblema koriste za izračun rješenja početnog problema. Izračun vjerojatnosti prekida napada može se prikazati rekurzivnom funkcijom. Stoga se dinamičko programiranje [42], koje se koristi za rješavanje problema u kojima se repetitivno ponavljaju potproblemi, čini kao mogući izbor metode optimiranja ovog problema. Optimiranje rasporeda zaštitnih elemenata po odabranom putu napada može se provesti metodom za rješavanje *problema ruksaka*. Međutim, specifičnost optimiranja sustava zaštite je da uvođenje svakog novog elementa zaštite u pretraživanju optimalnog rješenja potencijalno mijenja prostor pretraživanja i posljedično uzrokuje promjenu kritičnog puta. Iz tog razloga odabir najboljeg rješenja za određenu cijenu iz ranije rekurzije i apliciranje novog elementa zaštite može dati kriva rješenja jer vrijednost ranije rekurzije, s novo apliciranim elementom, ne vrijedi.

4.1.3. Optimiranje evolucijskim algoritmima

Optimiranju odabira zaštitnih elemenata i njihovom pozicioniranju može se pristupiti evolucijskim algoritmima koji stohastičke metode optimizacije provode simulirajući evolucijske procese u prirodi. Dva osnovna principa evolucije, selekciju i varijaciju, provode nad skupom mogućih rješenja koja se nazivaju kandidatima ili jedinkama. Selekcija predstavlja prirodni proces preživljavanja onih koji su bolje prilagođeni uvjetima okoline i veće vjerojatnosti reprodukcije dominantnih jedinki. Ovaj se proces u evolucijskim algoritmima simulira dodjelom veće vjerojatnosti odabiru kvalitetne jedinke za prijenos u sljedeću iteraciju (preživljavanje) i većom vjerojatnošću da će sadržaj rješenja biti korišten u stvaranju novih rješenja (reprodukcija). Varijacija oponaša promjenu jedinki uslijed

rekombinacije gena i mutacija. Zahvaljujući operaterima varijacije, evolucijski algoritmi pretražuju prostor rješenja, uglavnom izbjegavajući lokalne minimume. Manja osjetljivost na oblik ili kontinuitet Pareto fronte, u odnosu na metode matematičkim programiranjem, velika su prednost evolucijskih algoritama. Slika 4.3 prikazuje korake u izvođenju procesa evolucijskog algoritma. U početku algoritam izrađuje početnu populacijom nakon čega se odvija proces evaluacije, rekombinacije, mutacije i selekcije, sve dok se ne zadovolji uvjet završetka izvođenja. Najčešći je uvjet maksimalan broj izvođenja evolucijskog procesa, a istovremeno uvjet prekida može biti izostanak promjena u rješenjima pronađenim u nekoliko posljednjih iteracija.



Slika 4.3 Proces izvođenja evolucijskog algoritma

Evolucijski algoritmi pokazali su se učinkoviti u višekriterijskom optimiranju i često se primjenjuju u tom području [43]. Višekriterijskim evolucijskim algoritmima (*engl. Multiobjective Evolutionary Algorithms*, MOEA) pokušava se pronaći skup gotovo optimalnih rješenja unutar zadanog broja iteracija.

Prema pristupu rješavanja problema, MOEA možemo podijeliti na nekoliko vrsta. Najjednostavnija je metoda svođenje više kriterija agregatnom funkcijom na jednostruki problem, kako je opisano u poglavlju 4.1.1.1. Dominantnije MOEA metode, po udjelu korištenja i mogućnosti primjena, su metode s pristupom baziranim na populaciji i metode s pristupom baziran na Pareto fronti. Razlika je način na koji se provodi proces selekcije.

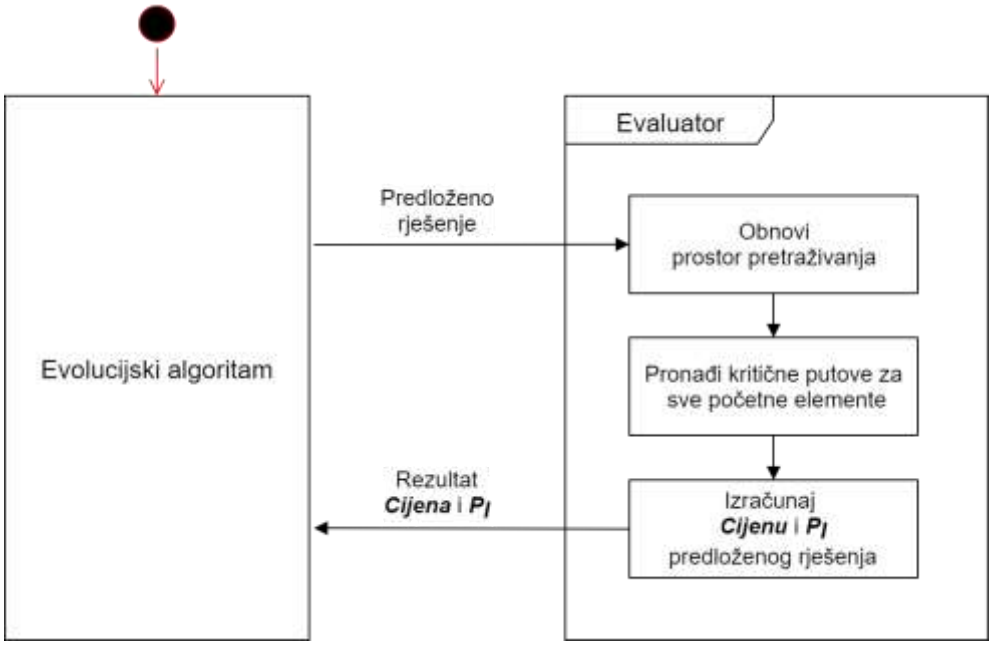
Metode zasnovane na populaciji (npr. Vector Evaluated Genetic Algorithm, VEGA) ne uspoređuju kvalitetu jedinke na razini cijele populacije, poput Pareto metoda, već u svakoj generaciji dijele populaciju na manje populacije (podpopulacije) prema funkcijama cilja. Iz manjih populacija stvara se nova populacija pri čemu u procesu odabira veću vjerojatnost odabira imaju kvalitetnije jedinke iz pojedine podpopulacije. Nedostatak ovakvog odabira je što jedinke čija kvaliteta je prosječno dobra u svim podpopulacijama ima manju vjerojatnost odabira od onih koje su možda kvalitetne samo po jednoj funkciji cilja, a unaprjeđenje prosječne jedinke može rezultirati kvalitetnijim krajnjim rješenjem.

Kvalitetniji način odabira jedinki za proces evolucije koriste metode zasnovane na Pareto odabiru. U prvu generaciju takvih algoritama spadaju *Non-dominated Sorting Genetic Algorithm (NSGA)*, *Niched-Pareto Genetic Algorithm (NPGA)*, *Multiobjective Genetic Algorithm (MOGA)*. Druga generacija algoritama zasnovanih na Pareto odabiru uvodi mehanizam *elitizma*, odnosno čuva najbolje jedinke od izmjene tijekom procesa varijacije i osigurava prijenos u sljedeću generaciju tijekom selekcije. U prvoj generaciji za svaku jedinku pa tako i za najbolju, postoji vjerojatnost da će se izgubiti u procesu selekcije ili će biti promijenjena tijekom varijacije. U drugoj generaciji ta je vjerojatnost jednaka nuli. U ovu kategoriju spadaju *Strength Pareto Evolutionary Algorithm (SPEA, SPEA2)*, *Pareto Archived Evolution Strategy (PAES), Niched Pareto Genetic Algorithm 2 (NPGA 2)* i drugi.

Cilj višekriterijskog optimiranja je pronaći skup rješenja koja leže na Pareto fronti pri čemu su dovoljno različita da predstavljaju cijelo područje Pareto optimalne fronte. Iz tog razloga popularan algoritam druge generacije evolucijskih algoritama je *Nondominated Sorting-based multiobjective* (NSGA-II) [44], zasnovan na genetskim algoritmima. Uz provođenje elitističkog odabira jedinki za sljedeću generaciju, koristi mehanizam eksplicitnog zadržavanja raznovrsnosti rješenja i poticanje dominantnih rješenja. U sljedećem poglavlju prikazano je višekriterijsko optimiranje sustava zaštite primjenom ovog algoritma.

4.1.3.1 Provedba NSGA-II nad modelom zaštite štićenog objekta

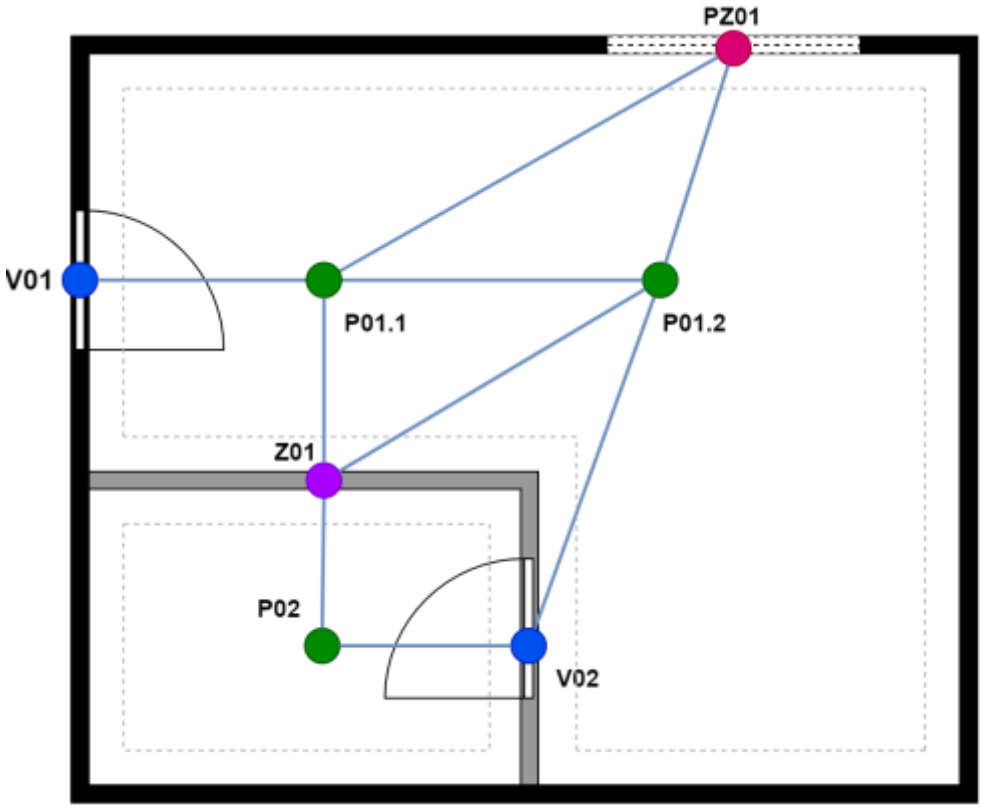
Slika 4.4 prikazuje blok-shemu optimiranja sustava zaštite evolucijskim algoritmom. Evolucijski algoritam predlaže rješenja za koja evaluator izračunava cijenu zaštitnih elemenata primijenjenih u rješenju i vjerojatnost zaustavljanja napadača. Za njihov izračun evaluator oprema štićeni prostor zaštitnim elementima i pronalazi kritične putove za sve početne elemente.



Slika 4.4 Blok-shema optimiranja sustava zaštite evolucijskim algoritmom

Provođenje optimiranja evolucijskim algoritmom zahtijeva definiranje prostora pretraživanja nad kojim se provodi optimiranje. Prostor pretraživanja predstavlja sva prihvatljiva rješenja, među kojima se traže optimalna. Među evolucijskim algoritmima često se koriste genetski algoritmi. U praksi provedbe genetskih algoritama, binarni prikaz rješenja daje dobre rezultate pa se kromosom može predstaviti binarnim vektorom. Za optimiranje problema odabira i raspoređivanja zaštitnih elemenata kromosom treba biti definiran tako da opisuje njihov odabir i lokaciju na objektu s višestrukim brojem ulaznih pozicija.

Slika 4.5 prikazuje primjer objekta za koji se projektira optimalan sustav zaštite. Nad tlocrtom objekta iscrtan je graf mogućeg kretanja napadača koji se sastoji od sedam čvorova i devet veza. Čvorovi vrata V01 i prozor PZ01 predstavljaju moguće elemente ulaska, a prostorija predstavljena čvorom P02 predstavlja cilj napada.



Slika 4.5 Primjer štićenog objekta

Tablica 4.1 sadrži sve tipove gradbenih elemenata, korištenih u primjeru građevine, kojima se napadač može kretati. Uz njih su popisani zaštitni elementi koji im se mogu pridijeliti u svrhu poboljšanja zaštite objekta.

Tablica 4.1 Primjer podržanih zaštitnih elemenata

Gradbeni element Model [*Host element*]	**Podržani zaštitni elementi** Model [*SupportedSecurityElements*]
Vrata (Vxx)	Protuprovalna brava (PB) Magnetski kontakt (MK)
Prozor (PZxx)	Zaštitne rešetke (RŠ) Detektor razbijanja stakla (DS) Magnetski kontakt (MK)

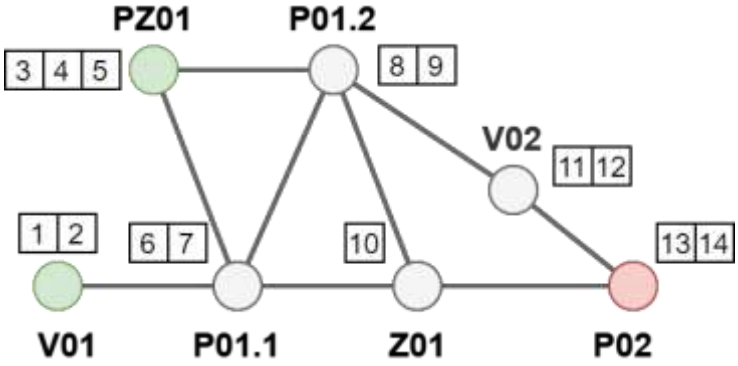
Zid (Zxx)	Detektor vibracija (DV)
Prostorija (Pxx)	Infracrveni detektor (ID) Dualni detektor (DD)

Ovi su podaci dostupni iz modela logičke razine štićenog objekta. Modelom je definirano i da primjena zaštitnog elementa potencijalno unosi vrijeme zadržavanja i vjerojatnost detekcije, uz njihovu ovisnost o sposobnostima napadača.

Tablica 4.2 prikazuje dodjelu zaštitnih elemenata lokacijama, na osnovu čega se može izraditi prostor pretraživanja.

Tablica 4.2 Podaci za izgradnju matrice prostora pretraživanja

Lokacije	**V01**	**PZ01**	**P01.1**	**P01.2**	**Z01**	**V02**	**P02**
Vrsta gradbenog elementa	Vrata	Prozor	Prostorija	Prostorija	Zid	Vrata	Prostorija
Podržani zaštitni elementi	PB	RŠ	ID	ID	DV	PB	ID
	MK	DS	DD	DD	-	MK	DD
	-	MK	-	-	-	-	-
Ukupan broj zaštitnih elemenata	2	3	2	2	1	2	2



Slika 4.6 Primjer štićenog objekta prikazan grafom

Moguće rješenje možemo predstaviti kao vektor (slika 4.7).

	V01		**PZ01**			**P01.1**		**P01.2**		**Z01**	**V02**		**P02**	
Zaštitni element	PB	MK	RŠ	DS	MK	ID	DD	ID	DD	DV	PB	MK	ID	DD
Pozicija bit-a	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Slika 4.7 Prostor pretraživanja predstavljen vektorski

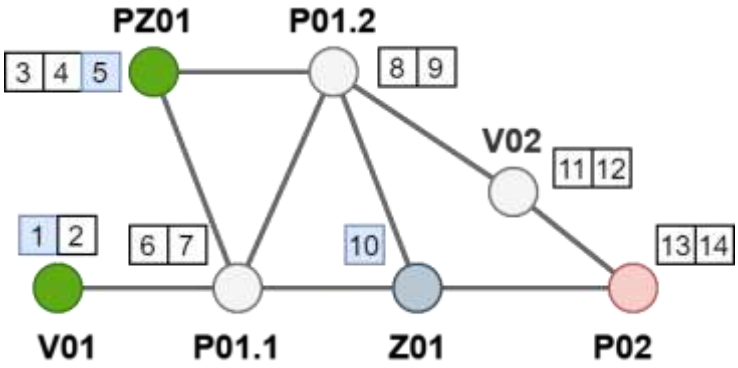
Predloženo rješenje zaštite, kojeg čini odabir zaštitnih elemenata na pojedinim lokacijama objekta, može se predstaviti kao binarni vektor

$$x = [x_1, x_2, \dots, x_b], \tag{4.17}$$

pri čemu je broj bitova b određen ukupnim brojem dostupnih zaštitnih elemenata po svim čvorovima

$$b = \sum_{i=1}^{|V|} |ae(v_i)|. \tag{4.18}$$

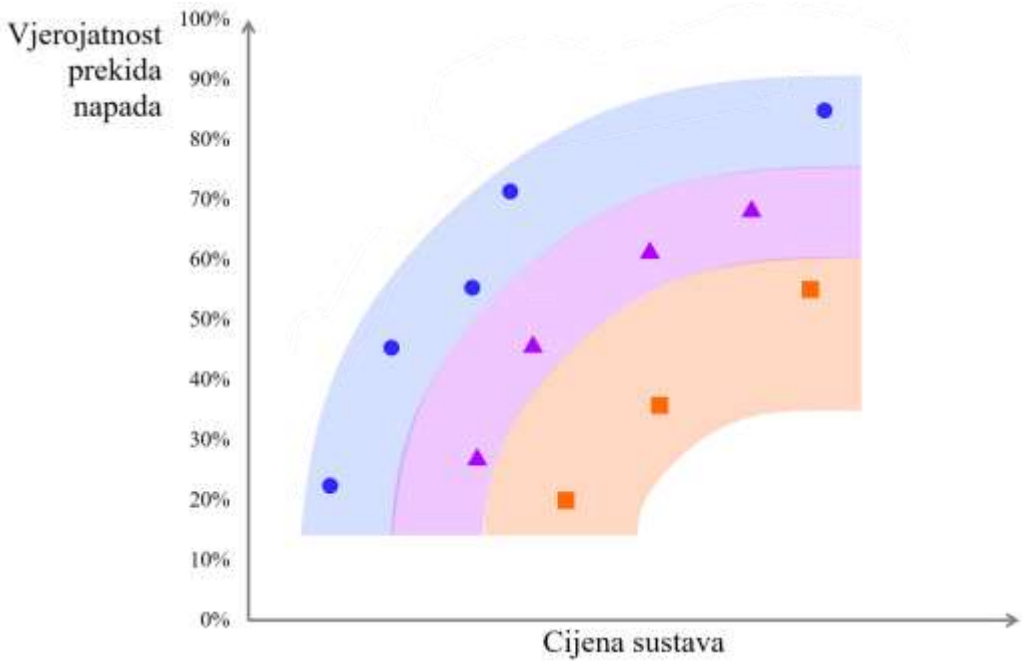
Za objekt iz primjera, rješenje {1, 5, 10}, odnosno u binarnom prikazu [1 0 0 0 1 0 0 0 0 1 0 0 0 0], postavlja protuprovalnu bravu na V01, magnetski kontakt na prozor PZ01 te detektor vibracija na zid Z01 (slika 4.8).



Slika 4.8 Aktivirani elementi zaštite

U svojem izvođenju NSGA-II inicijalno stvara početnu populaciju potencijalnih rješenja koju čini N binarnih vektora, slučajnim odabirom N brojeva u intervalu $[0, 2^b - 1]$. Redni broj iteracije algoritma t jednak je nuli. Evolucijski proces se najčešće izvodi zadani broj iteracija. Moguće je postaviti dodatne uvjete za završetak procesa, poput nepronalaska novih rješenja u nekoliko uzastopnih iteracija.

U procesu evaluacije za svako predloženo rješenje, predstavljeno binarnim vektorom, izračunava se cijena sustava prema izrazima (4.8) i (4.10) i funkcija dobrote prema izrazima (4.16) i (4.14). Slika 4.9 prikazuje predložena rješenja, grupirana u skupove koji se nazivaju frontama. Pripadnost fronti koristi se u procesu selekcije za uspoređivanje kvalitete rješenja.



Slika 4.9 Predložena rješenja grupirana prema rangu dominacije

Svaka fronta predstavlja jedinke koje pripadaju određenoj razini dominantnosti. Jedinka p dominira jedinku q ako vrijedi da je cijena jedinke p manja od cijene jedinke q, a vjerojatnost prekida napada za jedinku p veća od vjerojatnosti prekida napada za q, odnosno

$$p \prec q = cost(p) < cost(q) \wedge P_I(p) > P_I(q). \tag{4.19}$$

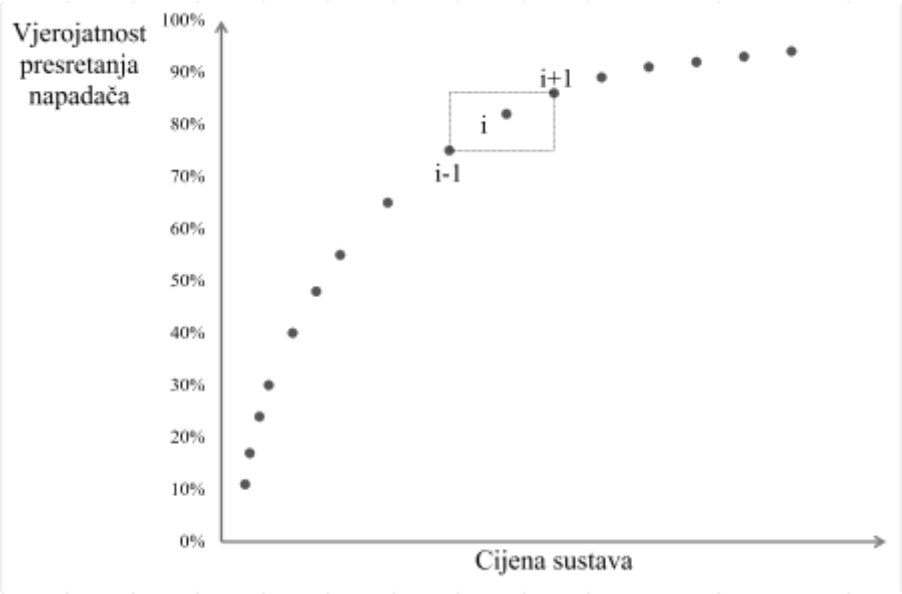
Slika 4.10 prikazuje pseudo-kod algoritma za provedbu grupiranja jedinki u fronte dominantnosti. U svakoj iteraciji grupiranja pronalaze se dominantne jedinke kojima se dodjeljuje skup dominiranih jedinki. Dominantnost jedinke utvrđuje se brojanjem koliko drugih jedinki dominira trenutnu jedinku i ta se vrijednost sprema kao brojač dominacije (slika 4.10, linije 03-09). Ako nijedna jedinka ne dominira promatranu jedinku, ona se proglašava dominantnom i smješta u skup dominantnih jedinki (slika 4.10, linije 10-13). U sljedećem koraku svakoj jedinki p iz skupa dominantnih jedinki, uzimaju se sve iz njezinog skupa jedinke q iz skupa dominiranih jedinki i umanjuje im se brojač dominacije za jedan (slika 4.10, linije 15-19). Svaka dominirana jedinka q čija je nova vrijednost brojača dominacije jednaka nuli, postavlja se u sljedeću frontu dominantnih jedinki (slika 4.10, linije 20-22). Postupak se ponavlja sve dok skup dominiranih jedinki ne postane prazan skup. Rezultat provedbe algoritma je da svaka predložena jedinka ima pridružen broj fronte, odnosno razreda, kojem pripada.

```
00 Grupiraj_jedinke_u_fronte_dominantnosti(Populacija P) {
01  za svaki (p u P) {
02    p.skup_dominiranih_rješenja = [];
03    p.brojač_dominacije = 0;
04    za svaki (q u P) {
05      ako (p dominira q)
06        p.skup_dominiranih_rješenja.dodaj(q);
07      inače ako (q dominira p)
08        p.brojač_dominacije = p.brojač_dominacije + 1;
09    }
10    ako je (p.brojač_dominacije = 0) {
11      p.razred = 1;
12      fronta[1].dodaj(p);
13    }
14    i = 1;
15    dok je (fronta[i] != []) {
16      iduća_fronta = [];
17      za svaki (p u fronta[i]) {
18        za svaki (q u p.skup_dominiranih_rješenja ) {
19          q.brojač_dominacije = q.brojač_dominacije - 1;
20          ako je (q.brojač_dominacije = 0) {
21            q.razred = i + 1;
22            iduća_fronta.dodaj(q);
23          }
24        }
25      }
26      i = i + 1;
27      fronta[i] = iduća_fronta;
28    }
29  }
```

Slika 4.10 Algoritam grupiranja jedinki u fronte dominantnosti

Optimiranje sustava zaštite ima dva cilja pa je složenost ovog algoritma $O(2N^2)$, pri čemu je N broj jedinki u populaciji, što je bolje od uspoređivanja svih elemenata međusobno čija je složenost $O(2N^3)$.

Pripadnost rangu dominacije nije jedino svojstvo jedinke koje se koristi za usporedbu u procesu selekcije. Ono utječe na konvergenciju populacija prema Pareto fronti, ali ne potiče odabir dovoljno različitih rješenja koja predstavljaju cijelo područje Pareto optimalne fronte. Rješenja za koja unutar populacije postoji više sličnih, odnosno onih čija kvaliteta se međusobno malo razlikuje, manje su bitna u procesu selekcije od onih čija kvaliteta se značajno razlikuje od drugih u populaciji. Procjena sličnosti kvalitete jedinke s drugima, odnosno gustoće rješenja koja okružuju promatranu jedinku u populaciji, provodi se izračunom prosječne udaljenosti od dvije najbliže točke u oba smjera po svakom cilju optimiranja (slika 4.11).



Slika 4.11 Određivanje udaljenosti do susjednih rješenja

Slika 4.12 prikazuje pseudo-kod algoritma koji svakoj jedinki populacije dodjeljuje izračunatu vrijednost grupirajuće udaljenosti.

```
Dodijeli_grupirajuću_udaljenost_jedinkama(Populacija P,Cilj[] Ciljevi)
{
  broj_rješenja  =  |P|;
  za svaki (p u P) {
    p.udaljenost = 0;
  }
  za svaki (cilj u Ciljevi) {
    P.Sortiraj_po_cilju(cilj);
```

```
    P[1].udaljenost = P[broj_rješenja].udaljenost = MAX;
    za i od 2 do broj_rješenja-1) {
       P[i].udaljenost = P[i].udaljenost +
       (P[i+1].vrijednost_cilja - P[i-1].vrijednost_cilja) /
       (max_vrijednost_cilja - min_vrijednost_cilja);
    }
  }
}
```

Slika 4.12 Algoritam dodjele grupirajuće udaljenosti jedinki

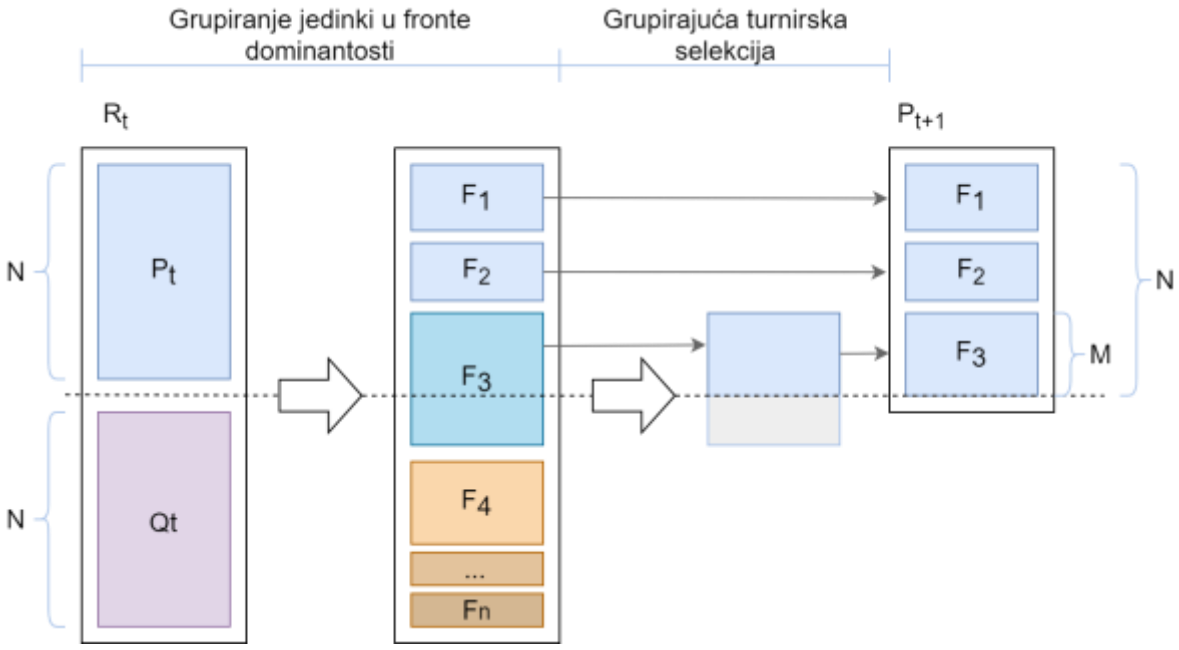
Proces se provodi za svaki cilj optimiranja. Populacija se sortira po vrijednosti kvalitete cilja za koji se provodi izračun grupirajuće udaljenosti. Prvoj i zadnjoj jedinki skupa dodjeljuje se maksimalna vrijednost za grupirajuću udaljenost kako bi se sačuvale u procesu selekcije. Za ostale jedinke p_i grupirajuća udaljenost računa se kao

$$udaljenost(p_i) = \frac{cost(p_{i+1}) - cost(p_{i-1})}{\max(cost) - \min(cost)} + \frac{P_I(p_{i+1}) - P_I(p_{i-1})}{\max(P_I) - \min(P_I)} \qquad (4.20)$$

pri čemu se maksimalna i minimalna vrijednost cilja odnose se na najveću i najmanju vrijednost cilja za jedinke koje pripadaju istoj fronti kojoj pripada i trenutna jedinka. Jedinka čija je udaljenost grupiranja manja, nalazi se u gušće zastupljenom prostoru rješenja od jedinke čija je udaljenost grupiranja veća. Vrijednost udaljenosti grupiranja drugo je svojstvo jedinke koje se koristi u procesu selekcije.

Svojstva pripadnost rangu dominacije, odnosno rang fronte u kojoj se jedinka nalazi i vrijednost udaljenosti grupiranja omogućuju uspostavu operatora usporedbe grupiranja kojim se provodi uspoređivanje jedinki u svrhu pronalaska uniformne širine Pareto optimalne fronte. Operator usporedbe grupiranja uspoređuje jedinke p, q i proglašava dominantnom jedinku p ako vrijedi $(p.razred > q.razred) \vee ((p.razred = q.razred) \wedge (p.udaljenost > q.udaljenost))$. Prva usporedba provjerava je li rang dominacije jedinke p veći od ranga jedinke q. Ako je manji, tada je jedinka q dominantnija. Ako se obje jedinke nalaze u istoj fronti, tada je dominantnija ona čija je udaljenost grupiranja veća.

Slika 4.13 prikazuje proces stvaranja nove populacije kojim NSGA-II osigurava elitistički odabir rješenja za sljedeću generaciju širinu rješenja na Pareto fronti.



Slika 4.13 Proces stvaranja nove populacije u NSGA-II

U svakoj iteraciji t iz populacije P_t, veličine N, genetskim operatorima stvara se dodatna populacija Q_t s N jedinki. Jedinke iz obje populacije zajedno čine populaciju R_t, veličine $2N$. Nad populacijom R_t provodi se grupiranje jedinki u fronte dominantnosti, odnosno svakoj jedinki pridružuje se oznaka fronte kojoj jedinka pripada. U sljedećem koraku jedinke iz skupa R_t prebacuju se u novu populaciju P_{t+1} koja ukupno sadrži N jedinki. Jedinke se prebacuju prema pripadnosti fronti, pri čemu prednost imaju jedinke čija oznaka fronte ima nižu vrijednost. Proces prebacivanja se izvršava sve dok vrijedi

$$\sum_{i=1}^{k} |F_i| < N \tag{4-21}$$

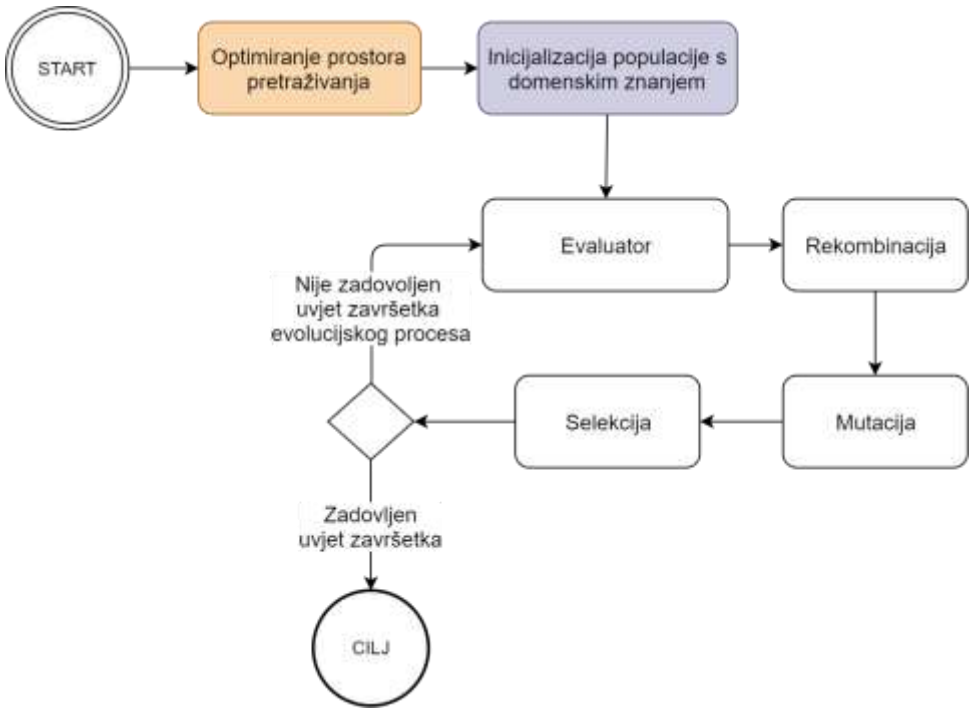
pri čemu je $|F_i|$ broj jedinki u i-toj fronti, a F_k zadnja fronta iz koje su prebačene sve jedinke. Nad frontom F_{k+1} provodi se grupirajuća turnirska selekcija tako da vrijedi

$$\sum_{i=1}^{k} |F_i| + M = N \tag{4-22}$$

pri čemu je M broj jedinki koje se dodaju populaciji P_{t+1}. Provođenje grupirajuće turnirske selekcije osigurava prijenos najboljih jedinki u sljedeću generaciju, bez promjena.

4.2. Hibridna metoda optimiranja pozicije i vrste zaštitnih elemenata

U provođenju procesa optimiranja pozicije i vrste zaštitnih elemenata može se iskoristiti poznavanje domene u svrhu ubrzanja pronalaska kvalitetnih rješenja. Kako prikazuje slika 4.14, u odnosu na standardni proces izvođenja evolucijskog algoritma (slika 4.3), doprinos optimiranju je komponenta koja prije izvršavanja evolucijskog procesa smanjuje veličinu prostora pretraživanja izbacivanjem elemenata koji sigurno ne pripadaju kvalitetnim rješenjima. Ovaj se proces optimiranja može izvršiti neovisno o algoritmu kojim se traže rješenja. Sljedeći je korak inicijalizacija populacije korištenjem domenskog znanja o učinkovitosti elemenata zaštite ovisno o njihovoj poziciji. Predlaganjem početnih rješenja smanjuje se broj iteracija potrebnih za nalaženje kvalitetnih nedominiranih rješenja. Navedeni postupci su doprinos u obliku hibridne metode optimiranja pozicije i vrste zaštitnih elemenata.



Slika 4.14 Proces izvođenja evolucijskog algoritma s elementima hibridne metode optimiranja

4.2.1. Optimiranje prostora pretraživanja

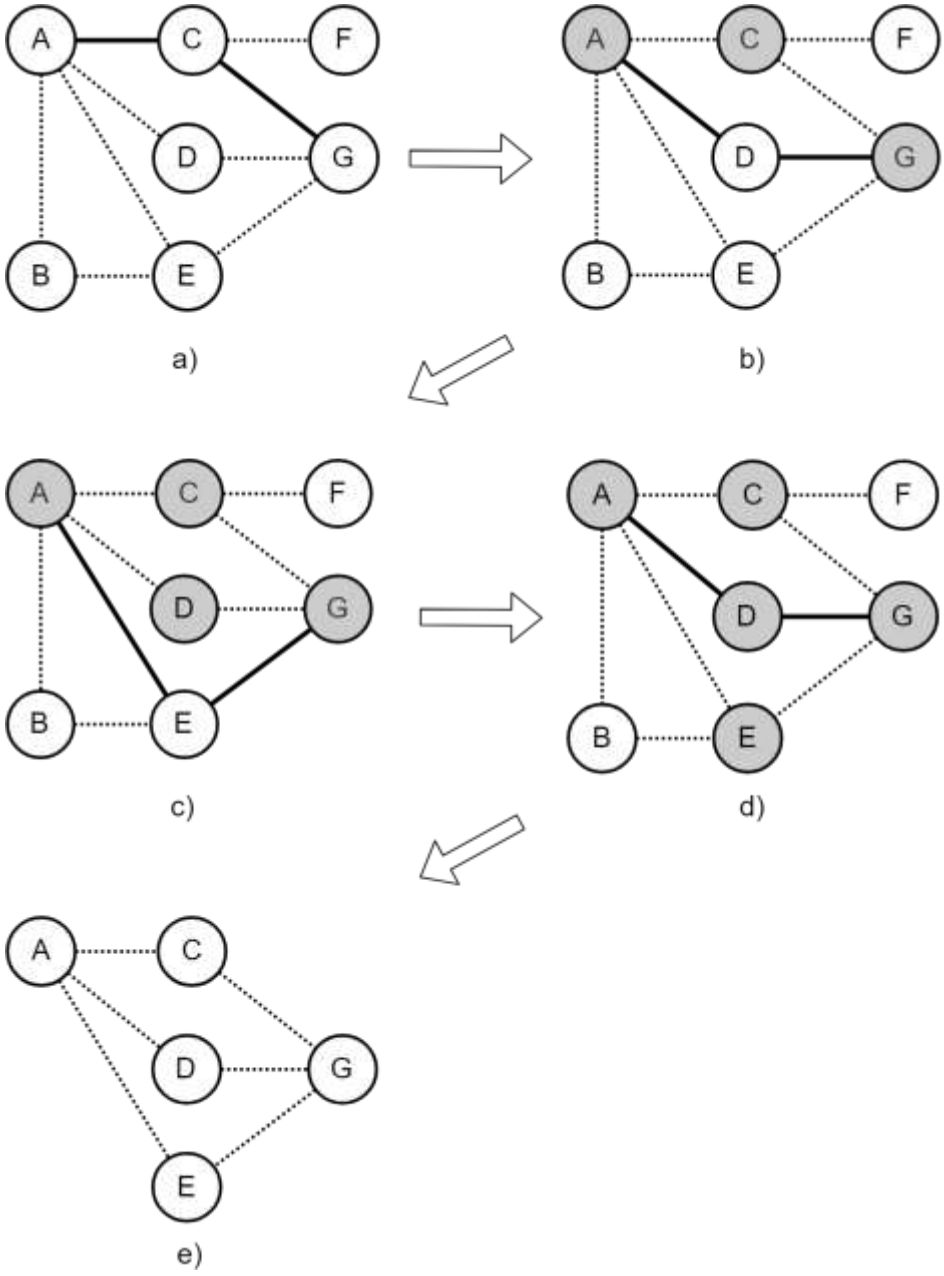
Automatizirano stvaranje grafa mogućeg kretanja napadača iz opisa štićenog objekta stvara sve čvorove grafa kojima bi se napadač mogao kretati. Međutim, među njima postoje čvorovi koji neće pripadati niti jednom optimalnom putu kretanja, za bilo koje predloženo rješenje zaštite. Takvi čvorovi i pripadajuće veze mogu se ukloniti iz grafa mogućeg kretanja napadača bez utjecaja na kvalitetu rješenja. Njihovo uklanjanje utječe na smanjenje prostora pretraživanja čime se smanjuje prostor pretrage mogućih rješenja problema. Elementi koji su nepotrebni za optimiranje mogu se pronaći kontinuiranim pronalaskom kritičnih putova i postavljanjem svih zaštitnih elemenata koji se mogu pridijeliti elementima kretanja štićenim objektom koji se nalaze na pronađenom kritičnom putu. Takav algoritam može prestati s pretraživanjem ako ne postoje novi čvorovi koji su pronađeni u zadnjem traženju optimalnog puta kretanja. Slika 4.15 prikazuje pseudo-kod algoritma kojim se iz prostora pretraživanja izbacuju nepotrebni elementi.

```
Optimiranje_prostora_pretraživanja() {
  pronađeniČvorovi = [];
  izvršavaj {
     prethodnoPronađeniČvorovi = pronađeniČvorovi;
     kritičniPutovi = PronađiKritičnePutoveZaSvePočetneČvorove();
     za svaki čvor u kritičniPutovi {
        čvor.ap = čvor.ae; //dodijeli sve zaštitne elemente
     }
     pronađeniČvorovi.Dodaj(čvor);
  }
  sve dok nije
   prethodnoPronađeniČvorovi.PotpuniPodskup(pronađeniČvorovi);

  // uklanjanje nepotrebnih elemenata
  za svaki element u prostorPretraživanja {
      ako element nije u pronađeniČvorovi {
          prostorPretraživanja.Ukloni(element);
      }
   }
}
```

Slika 4.15 Algoritam optimiranja prostora pretraživanja

Slika 4.16 prikazuje primjer grafa nad kojim se provodi optimiranje prostora pretraživanja. Čvor A predstavlja početni element s kojeg napadač kreće, a čvor G predstavlja cilj napada. Graf na slika 4.16 a) prikazuje sve čvorove bez primijenjenih zaštitnih elemenata i pronađen kritični put (A, C, G). Na te čvorove postavljaju se svi dostupni zaštitni elementi, čime postižemo maksimalnu zaštitu tog puta. Čvorove A, C i G dodajemo u skup aktiviranih čvorova P. Primjenom novih elemenata zaštite mijenja se konfiguracija sustava te kako prikazuje graf na slika 4.16 b), nastaje novi kritični put (A, D, G). Čvorovi označeni sivom bojom predstavljaju čvorove na koje su primijenjeni svi zaštitni elementi i čine skup aktiviranih čvorova P. Čvor D nalazi se na kritičnom putu, a nije član skupa P pa mu se dodaju svi dostupni zaštitni elementi i smješta ga se u skup P. U sljedećem koraku postavljaju se aktivni zaštitni elementi na sve čvorove u skupu P {A, C, D, G} i pronalazi kritični put (A, E, G), kako prikazuje slika 4.16 c). Čvor E nije element skupa P, a nalazi se na kritičnom putu pa se dodaje u P i pokreće se novo traženje kritičnog puta s maksimalno zaštićenim čvorovima {A, C, D, E, G}. Slika 4.16 d) prikazuje da je za takvu konfiguraciju štićenog objekta kritični put napada (A, D, G), koji ne sadrži niti jedan novi element kretanja, u odnosu na članove skupa P. Algoritam prestaje tražiti nove putove jer su svi elementi posljednje pronađenog kritičnog puta sadržani u P. Slijedi izbacivanje svih čvorova iz skupa $\mathcal{V}$ koji ne pripadaju skupu P i pripadajućih veza, kako prikazuje slika 4.16 e).

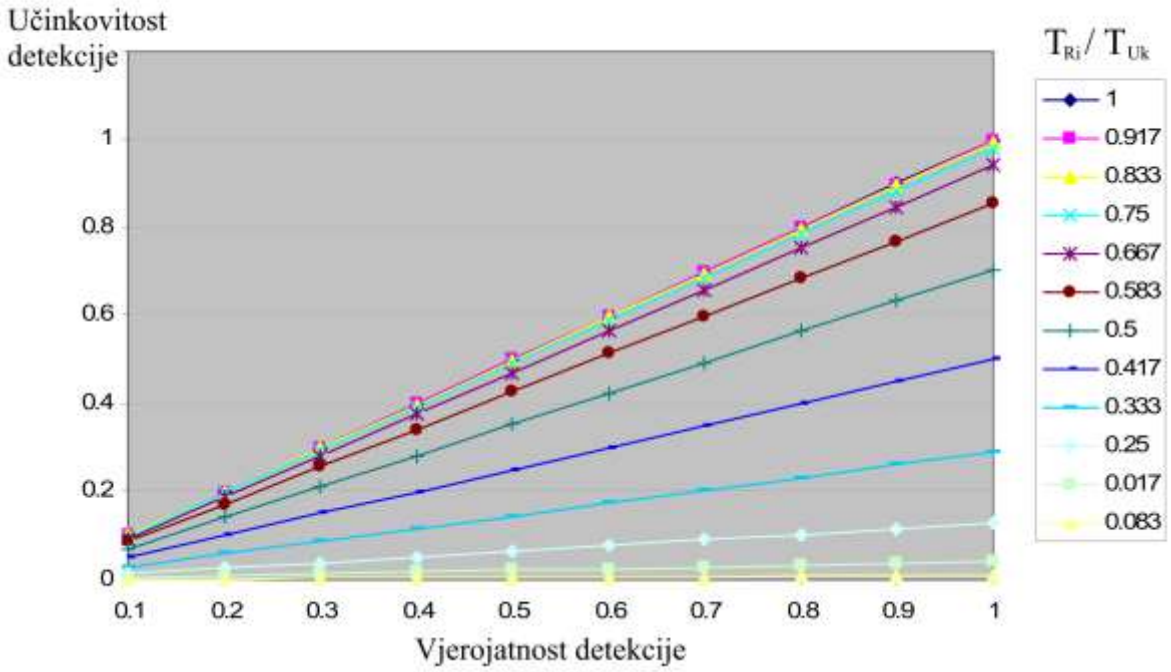


Slika 4.16 Primjer grafa nad kojim se provodi optimiranje prostora pretraživanja

4.2.2. Inicijalizacija populacije korištenjem domenskog znanja

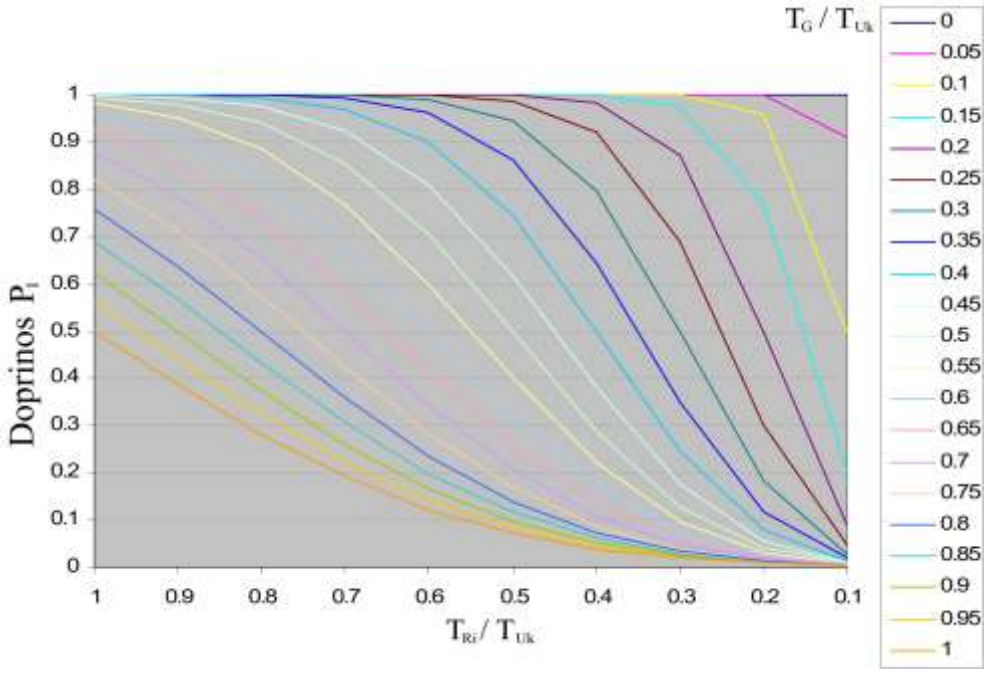
Optimiranje sustava može se provesti genetskim algoritmom pri čemu se početna populacija stvara slučajnim binarnim vektorima. Poznavanje domene pretraživanja omogućuje stvaranje početne populacije koja pruža algoritmu veću vjerojatnost pronalaska gotovo optimalnih rješenja Pareto fronte za određeni broj iteracija.

Smještaj zaštitnih elemenata utječe na njihovu učinkovitost [45]. Slika 4.17 prikazuje ovisnost učinkovitosti detekcije elementa zaštite o udaljenosti od cilja napada. Detektor smješten bliže početku napada ima veću učinkovitost u odnosu na detektor smješten bliže cilju napada. Primjer je da isti detektor, vjerojatnosti detekcije 90%, smješten na prvoj četvrtini puta napada ima učinak detekcije na zaustavljanje napada 88%, a smješten na zadnju četvrtinu puta samo 25% učinka.



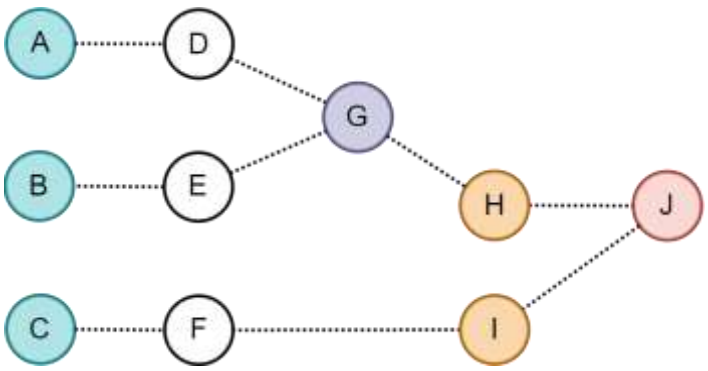
Slika 4.17 Utjecaj pozicije zaštitnog elementa na učinkovitost otkrivanja napada (preuzeto iz [45])

Slika 4.18 prikazuje ovisnost učinkovitosti blokiranja zaštitnog elementa o relativnoj udaljenosti od cilja napada (T_{Ri}/T_{Uk}). Apscisa je prikazana od veće vrijednosti prema manjoj kako bi se vizualno prikazala ovisnost učinkovitosti o udaljenosti od početnog elementa. Postavljanje elementa blokiranja bliže cilju napada ima veći utjecaj na vjerojatnost zaustavljanja napada od blokatora postavljenog bliže početku kritičnog puta napada. Primjer je sustav u kojem je odnos vremena zaštitara T_G i ukupnog vremena potrebnog napadaču od početka napada T_{Uk} jednak 0,35, tada je doprinos elementa blokiranja postavljenog na 3/10 ukupnog puta 40%, a postavljenog na 7/10 puta je 99%.



Slika 4.18 Utjecaj pozicije zaštitnog elementa na učinak blokiranja (preuzeto iz [45])

Ovisnost učinkovitosti detekcije i zadržavanja zaštitnog elementa o poziciji na putu kretanja napadača može se iskoristiti pri izradi početne populacije. Stvaraju se tri vrste jedinki koje se primarno razlikuju po rangu cijene predloženog rješenja. Time se izbjegava da predložena rješenja pokrivaju uski segment prostora pretraživanja. Metoda se fokusira na tri skupa čvorova: početni, susjedni ciljnom čvoru te oni koji spadaju u više kritičnih putova. Štićenje čvorova koji su članovi više kritičnih putova učinkovito je jer se jednim zaštitnim elementom utječe na više kritičnih putova. Slika 4.19 predstavlja tri kritična puta koji započinju od čvorova A, B i C. Čvor G pripadnik je više kritičnih putova. Čvorovi H i I susjedni su ciljnom čvoru J.



Slika 4.19 Značajni čvorovi na kritičnim putovima

Prvom vrstom jedinke želi se postići minimalno ulaganje pa prepustiti evolucijskom algoritmu da dodavanjem zaštitnih elemenata u procesu evolucije poboljša predloženo rješenja. Svakom čvoru $v_{1\mathrm{i}}$, koji je element skupa početnih čvorova $\mathcal{S} = \{v_{11}, v_{21}, \ldots, v_{(\mathrm{n}-1)\,1}, v_{\mathrm{n}1}\}$, dodjeljuje se jedan od dostupnih zaštitnih elemenata iz skupa $ae(v_{\mathrm{i}1})$ čija vjerojatnost detekcije je veća od nule. Svakom čvoru $v_{i,(\mathrm{t}-1)}$ koji prethodi ciljnom čvoru v_{t}, metoda dodjeljuje jedan od dostupnih zaštitnih elemenata iz skupa $ae(v_{i,(\mathrm{t}-1)})$ čije vrijeme zadržavanja napadača je veće od nule.

Metoda za izradu druge vrste jedinki postavlja zaštitne elemente sa svojstvom detekcije na početne čvorove kritičnih putova i zaštitne elemente sa svojstvom usporavanja na čvorove $v_{i,(\mathrm{t}-1)}$ koji prethode cilju napada. Uz početne i završne čvorove, značajni su i oni koji se pojavljuju u više kritičnih putova napada, odnosno pripadaju skupu $\mathrm{X} = \{p_1 \cap p_2 \cap .. \cap p_n\} \backslash v_t$. Postavljanje zaštitnog elementa na takve pozicije istovremeno štiti više kritičnih putova čime se evolucijskom algoritmu omogućuje istraživanje lokalno optimalnog rješenja. Na članove skupa X postavljaju se zaštitni elementi slučajnim odabirom.

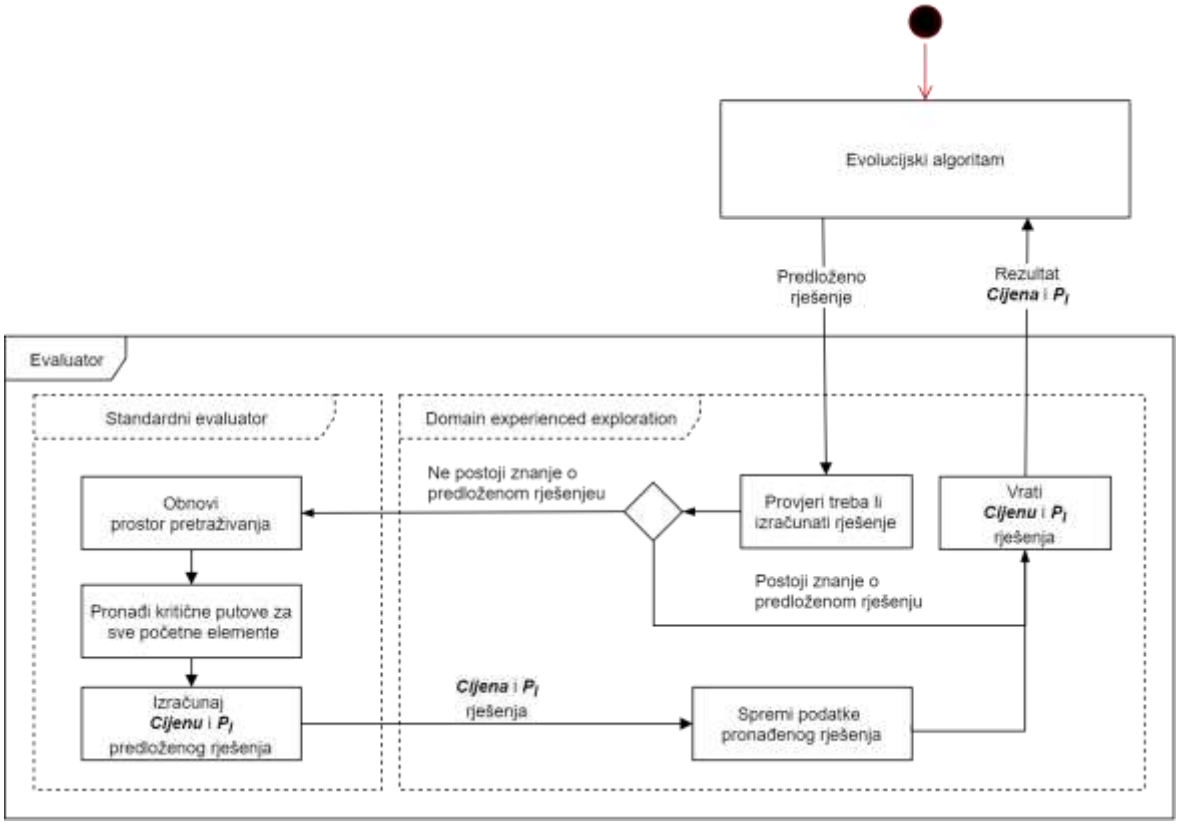
Metoda u izradi treće vrste jedinke iskorištava sve dostupne elemente zaštite i razmješta ih na početne čvorove, zatim susjedne ciljnom čvoru te onima koji se nalaze u više kritičnih putova. Kod ovakve jedinke očekuje se da će provođenje operatora genetskog algoritma uklanjati pojedine zaštitne elemente i optimizirati rješenje.

4.3. Napredna metoda optimiranja iskustvenim evaluatorom

Evolucijski algoritmi predlažu rješenja za koja od evaluatora traže ocjenu njihove kvalitete. Prijedlog rješenja sadrži skup zaštitnih elemenata koji se smještaju na elemente mogućeg kretanja napadača. Za svako predloženo rješenje evaluator mora izmijeniti prostor pretraživanja i pronaći kritične putove napada na osnovu kojih izračunava kvalitetu predloženog rješenja. Provedba takvog procesa optimiranja zahtijeva velik broj izmjena prostora pretraživanja i traženja kritičnih putova napada. Uvođenje algoritma kojim se izbjegavaju nepotrebne izmjene prostora pretraživanja i izračun kritičnih putova napada smanjuje broj potrebnih iteracija optimiranja i posljedično skraćuje vrijeme njegovog izvršavanja.

Optimizacijski algoritam uzima u obzir iskustvo, odnosno znanje stečeno u ranijim evaluacijama, i donosi odluku treba li izvršiti novi izračun za predloženo rješenje ili može, na osnovu ranije stečenog znanja, odabrati i vratiti kao rezultat spremljenu vrijednost evaluacije. Zbog načina rada, ova metoda naziva se pretraživanjem temeljenim na domenskom iskustvu (*engl. Domain Experienced Exploration*, D-EX2) [3].

Slika 4.20 prikazuje shemu algoritma D-EX2. U prvom koraku algoritma provjerava se treba li izračunati funkciju cilja ili je rješenje već poznato na osnovu ranijih izračuna. Ako rješenje nije poznato, pokreće se standardni evaluator koji obnavlja prostor pretraživanja, pronalazi kritične putove za sve početne elemente i na osnovu njih izračunava cijenu i vjerojatnost prekida napada. Metoda iskustvenog pretraživanja domene sprema pronađeno rješenje za buduće izračune i vraća evolucijskom algoritmu cijenu predloženog sustava i postignutu vjerojatnost zaštite.

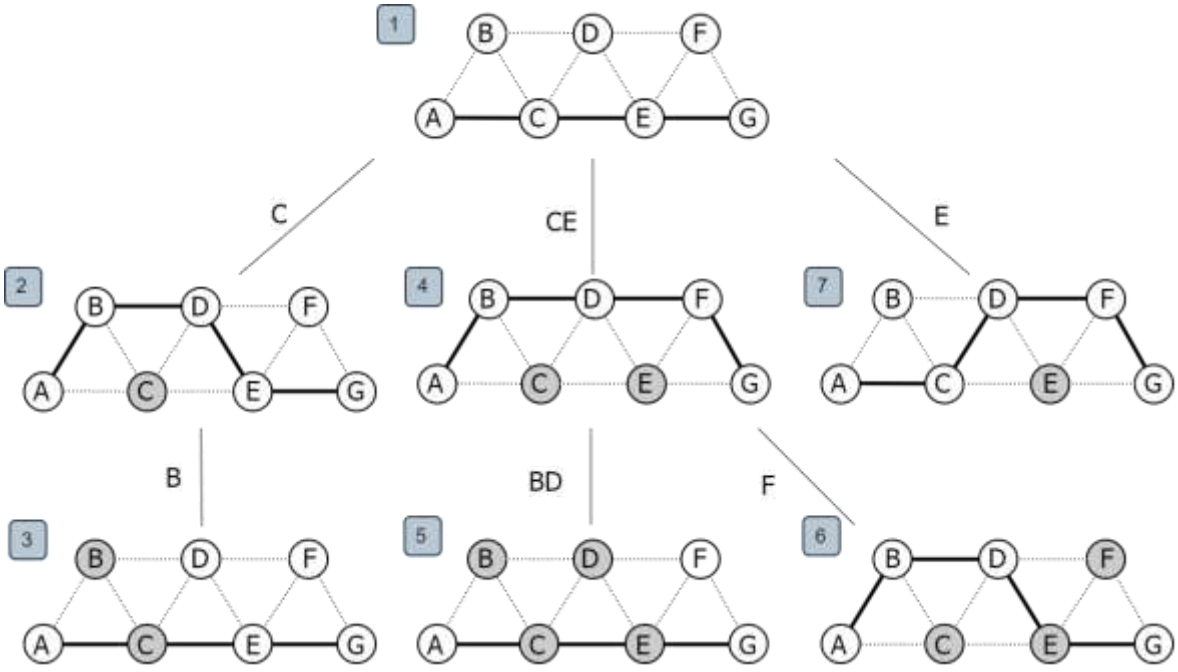


Slika 4.20 Osnovna shema evaluacije metodom iskustvenog pretraživanja domene rješenja

Osnovni koncept provjere treba li izračunati kvalitetu predloženog rješenja je provjera imaju li predloženi sigurnosni elementi utjecaj na poznate kritične putove. Provjera se provodi traženjem presjeka gradbenih elemenata na koje predloženo rješenje postavlja zaštitne

elemente i gradbenih elemenata u ranije pronađenim rješenjima. Ako je presjek predloženog rješenja s prethodnom pronađenim rješenjima prazan skup, tada predloženi sigurnosni elementi nemaju utjecaj na poznate kritične putove te nema potrebe izračunavati funkciju cilja već pronaći poznato rješenje koje odgovara predloženom.

Slika 4.21 prikazuje primjere izvođenja algoritma D-EX2. Graf koji predstavlja model mogućeg kretanja napadača sastoji se od sedam čvorova {A, B, C, D, E, F, G}, povezanih s jedanaest veza. Čvor A predstavlja element koji se koristi kao početni čvor napada, a čvor G predstavlja cilj napada. Algoritam D-EX2 pri izvođenju stvara predstavljenu strukturu pronađenih rješenja u obliku stabla. Pozicije pronađenih rješenja u stablu organizirane su tako da se kretanjem kroz stablo može zaključiti treba li nastaviti pretraživanje ili izvesti obnovu prostora pretraživanja, pronaći kritične putove napada i izračunati cijenu i vjerojatnost prekida napada.



Slika 4.21 Primjeri optimiranja algoritmom D-EX2

Slika 4.21, pozicija 1 prikazuje početno stanje sustava, bez implementiranih zaštitnih elemenata. Tada je optimalan put kretanja napadača čvorovima {A, C, E, G}. Predloži li algoritam optimiranja rješenje koje sadrži bilo koju kombinaciju zaštitnih elemenata na čvorovima B, D i F, takvo rješenje neće utjecati na optimalan put kretanja napadača jer

zaštitni elementi smješteni samo na te čvorove neće detektirati ili usporiti napadača. Algoritam D-EX2 provjerava utjecaj predloženog rješenja na kritični put provjerom sadrže li predloženo rješenje i kritični put zajedničke elemente. U ovom primjeru presjek predloženog rješenja koje sadrži kombinaciju elemenata B, D i F s početnim skupom {A, C, E, G} biti će prazan skup. To znači da predloženi zaštitni elementi ne utječu na poboljšanje zaštite kritičnog puta (A, C, E, G). U tom slučaju algoritam D-EX2, kao rezultat evaluacije, vraća ranije izračunate vrijednosti funkcije cilja.

Sljedeći primjer je predloženo rješenje koje sadrži zaštitne elemente za skup čvorova {B, C, F}. Presjek skupa {B, C, F} i početnog kritičnog puta {A, C, E, G} je skup {C}. Preostala dva čvora B i F u ovom koraku nemaju utjecaj na kritični put. Evaluator postavlja zaštitni element na čvor C, mijenja prostor pretraživanja i traži novi kritični put. U ovom primjeru je rezultat pretrage za kritičnim putom, s zaštitnim elementom na čvoru C, skup {A, B, D, E, G}, kako pokazuje slika 4.21, pozicija 2. Početnom se skupu aktivnih čvorova {A, C, E, G} pridodaju novi članovi {B, D} jer se nalaze na kritičnom putu. Novi skup aktivnih čvorova je {A, B, C, D, E, G}. Ponovno se provjerava utjecaj čvorova predloženog rješenja {B, C, F} na skup aktivnih čvorova {A, B, C, D, E, G} i rezultat {A, B, C, D, E, G} ∩ {B, C, F} je {B, C}. Presjek nije prazan skup pa algoritam postavlja zaštitne elemente na čvorove B i C, mijenja prostor pretraživanja i pronalazi novi kritični put {A, C, E, G} (pozicija 3). Skup aktivnih čvorova se ne proširuje novim članovima, a njegov presjek s početno predloženim rješenjem {B, C, F} je jednak ranije pronađenom {B, C}. Zaključak je da čvor F ne utječe na kritični put. Algoritam D-EX2 za rezultat evaluacije vraća vjerojatnost prekida napada pronađenu za {B, C}, a cijenu sustava prema zaštitnim elementima smještenim na čvorove {B, C, F}.

Nakon prethodnog istraživanja D-EX2 je stekao dovoljno iskustva da izbjegne obnovu prostora pretraživanja, traženje kritičnih putova te određivanje rezultata funkcije cilja za prijedlog rješenja {C, F}. U prvom koraku {A, C, E, G} ∩ {C, F} je {C}, a algoritam već posjeduje informaciju o kritičnom putu {A, B, D, E, G} za prostor pretraživanja s postavljenim zaštitnim elementom na čvoru C. U sljedećoj iteraciji {A, B, D, E, G} ∩ {C, F} je prazan skup jer čvor F nema utjecaj na promjenu kritičnog puta pa D-EX2 vraća ranije pronađenu i spremljenu vjerojatnost prekida napada za kritični put {A, B, D, E, G}, a cijenu sustava za zaštitne elemente na {C, F}.

Za predloženo rješenje {B, C, D, E} u prvom koraku {A, C, E, G} ∩ {B, C, D, E} daje {C, E}. Ranije pronađeno rješenje sa čvorom C (pozicija 2) ne uzima se u razmatranje jer čvor E utječe na kritični put. Kako ne postoji raniji zapis o rješenju za čvorove {C, E}, algoritam smješta zaštitne elemente na {C, E} i pronalazi novi kritični put {A, B, D, F, G}, prikazan na poziciji slika 4.21, pozicija 4. Skup aktivnih elemenata {A, C, E, G} proširuje se elementima kritičnog puta i postaje {A, B, C, D, E, F, G}. Rezultat evaluacije se sprema kao podređeni element osnovnog rješenja i nastavlja se pretraživanje. U sljedećem koraku presjek aktivnih elemenata {A, B, C, D, E, F, G} ∩ {B, C, D, E} je jednak {B, C, D, E}. Algoritam aktivira zaštitne elemente na čvorovima {B, C, D, E} i pronalazi kritični put {A, C, E, G} (slika 4.21, pozicija 5). U sljedećoj iteraciji presjek je jednak ranijem pa se zadnje pronađeno rješenje vraća kao rezultat.

Predloženo rješenje {C, E, F} odabrano je za primjer grananja pronađenih rješenja. U prvom koraku {A, C, E, G} ∩ {C, E, F} je jednako {C, E}, za što već postoji rješenje na poziciji 4, podređeno glavnom čvoru na poziciji 1. Kritični put tog rješenja {A, B, D, F, G} i aktivni elementi {A, B, C, D, E, F, G} poznati su od ranije. U sljedećem koraku presjek {A, B, C, D, E, F, G} ∩ {C, E, F} je {C, E, F}. Znanje od ranije za ovaj prijedlog rješenja ne postoji pa algoritam primjenjuje zaštitne elemente na čvorove {C, E, F} i pronalazi kritični put {A, B, D, E, G}, prikazan na poziciji 6. Ovo se rješenje dodaje kao podređeno postojećem rješenju {C, E} na poziciji 4.

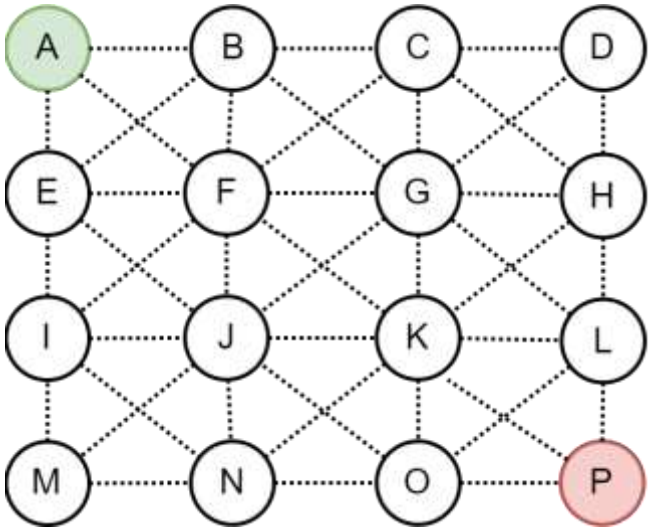
Za predloženo rješenje {B, E} algoritam u prvom koraku određuje {A, C, E, G} ∩ {B, E} = {E}. Ne postoji raniji zapis o rješenju za čvor E pa algoritam mijenja prostor pretraživanja s aktivnim elementom na čvoru E i pronalazi kritični put {A, C, D, F, G} i proširuje skup aktivnih elemenata na {A, C, D, E, F, G}. Pronađeno rješenje dodaje se kao podređeni član osnovnom elementu (pozicija 7). U sljedećem koraku presjek aktivnih elemenata i predloženog rješenja {A, C, D, E, F, G} ∩ {B, E} je jednako ranije pronađenom rješenju {E} pa nema potrebe za nastavkom pretraživanja. Algoritam vraća vjerojatnost prekida napada za kritični put {A, C, D, F, G} te cijenu zaštitnih elemenata na čvorovima {B, E}.

Slika 4.22 prikazuje pseudo-kod algoritma za evaluaciju predloženog rješenja temeljenu na pretraživanju uz domensko iskustvo.

```
  Evaluiraj_s_D-EX2(vektorJedinke) {
01 za svaki č u početniVrhovi {
02   ako je početnaRješenja[č] == [] {
03     prostorPretraživanja = PostaviZaštitneElemente(elementi: []);
04     početnaRješenja[č]= PronađiKritičniPut(prostorPretraživanja,č);
05   }
06   mogućeRješenje = početnaRješenja[č];
07   sve dok nije pronađenoRješenje izvodi {
08     presjek = vektorJedinke ∩ mogućeRješenje.aktivniElementiOdPočetka;
09     ako je ((presjek ≠ mogućeRješenje.aktivniElementi) i (presjek ≠[])) {
10       ako mogućeRješenje.podrješenja.Sadrži(presjek) {
11         mogućeRješenje = mogućeRješenje.podrješenja[presjek];
12       }
13       inače {
14         prostorPretraživanja =
15                       PostaviZaštitneElemente(presjek.aktivnePozicije);
16         novoRješenje = PronađiKritičniPut(prostorPretraživanja, č);
17         mogućeRješenje.podrješenja[novoRješenje.aktivniElementi] =
                                                          novoRješenje;
18         mogućeRješenje = početnaRješenja[č];
19     }
20     inače {
21       pronađenoRješenje = istina;
22   }
23   vrati mogućeRješenje.kritičniPut;
24 }
```

Slika 4.22 Pseudo kod algoritma D-EX2

Slika 4.23 je primjer grafa čiji čvorovi su međusobno izrazito povezani. Takva povezanost čvorova omogućuje napadaču u većini slučajeva jednostavno zaobilaženje zaštitnih elemenata koji detektiraju njegovu prisutnost. Rješenje koje predloži algoritam optimiranja mora sadržavati zaštitne elemente na većini čvorova. Ovakva konfiguracija grafa nije povoljna za D-EX2 algoritam jer se za većinu predloženih rješenja izvršava minimalno jedno traženje međurješenja. Ovakav graf ne predstavlja topologiju realnih građevina pa se ovaj nedostatak algoritma ne manifestira u radu.



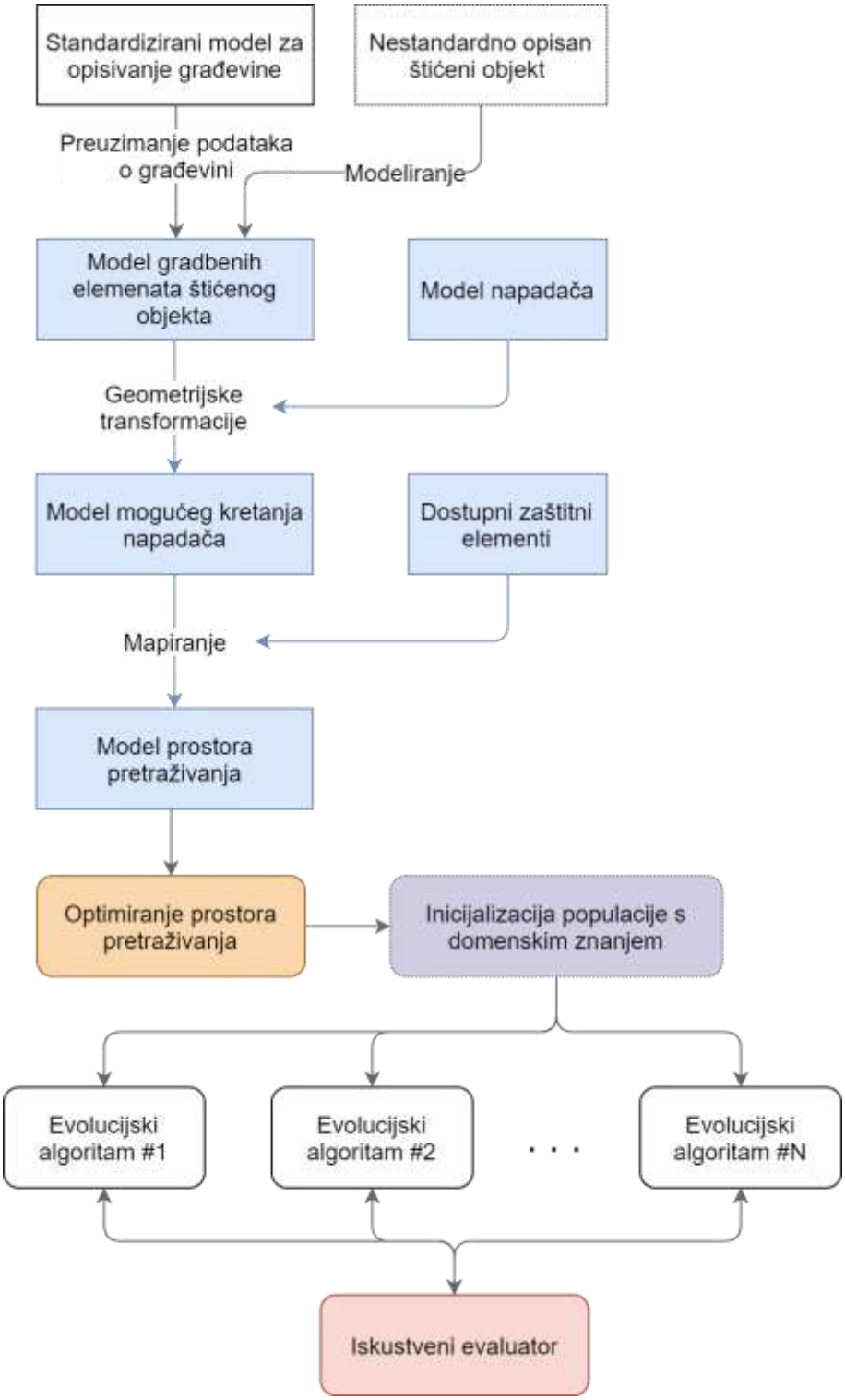
Slika 4.23 Graf složenog prostora pretraživanja

4.4. Cjeloviti prikaz optimiranja sustava zaštite

Slika 4.24 prikazuje cjeloviti proces optimiranja pozicije i vrste zaštitnih elemenata, s naznačenim elementima procesa koji predstavljaju doprinose rada. U prvoj fazi procesa optimiranja kroz nekoliko se koraka provodi transformacija opisa objekta koji se štiti, uzimajući u obzir svojstva napadača, vrstu, dimenziju i poziciju gradbenih elemenata štićenog objekta te raspoloživost zaštitnih elemenata. Rezultat transformacije je model koji opisuje kako se napadač može kretati štićenim objektom i koji su, od raspoloživih zaštitnih elemenata, primijenjeni u predloženom rješenju.

Nad modelom se mogu provesti različite metode traženja optimalnih rješenja. Uvedena je hibridna metoda optimiranja koja prethodi traženju optimalnih rješenja, a rezultira bržim pronalaskom kvalitetnih rješenja, neovisno o metodi traženja optimalnih rješenja. Hibridna metoda u prvom koraku optimizira prostor pretraživanja izbacujući nepotrebne čvorove i pripadajuće bridove, čime smanjuje prostor pretraživanja. U sljedećem koraku predlaže početna rješenja postavljajući zaštitne elemente na pozicije koje inicijalno predstavljaju optimalan put kretanja napadača, uzimajući u obzir njihovu vrstu i udaljenost do cilja. Ovime se postiže pronalazak kvalitetnih rješenja uz manji broj iteracija.

Traženje rješenja se dodatno ubrzava uvedenim algoritmom DEX-2 koji provodi evaluaciju predloženih rješenja korištenjem ranije stečenog znanja o prostoru pretraživanja. Pri provedbi evaluacije algoritam D-EX2 proširuje znanje o prostoru pretraživanja i izbjegava nepotrebne izračune. Više algoritama koji traže optimalna rješenja može dijeliti iskustveni evaluator što je korisno u procesu eksperimentiranja s parametrima algoritama ili im se uspoređuje učinkovitost.



Slika 4.24 Cjeloviti proces optimiranja sustava zaštite

5. EKSPERIMENTALNI REZULTATI

5.1. Ciljevi eksperimentiranja

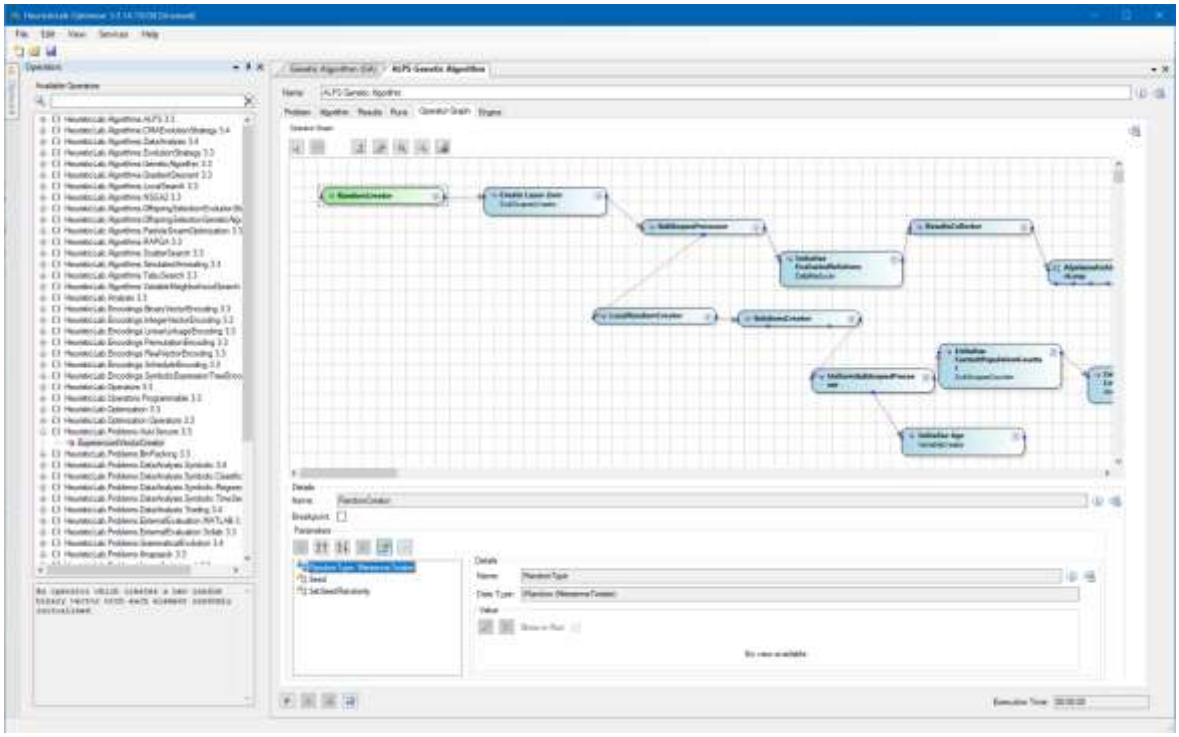
Postavljeni su sljedeći ciljevi koji se eksperimentiranjem trebaju postići:

- utvrditi na primjeru realne građevine doprinosi li metoda optimiranja grafa mogućeg kretanja napadača smanjenju broja čvorova i veza grafa kretanja, a time i smanjenju prostora pretraživanja,
- odrediti koje su konfiguracije genetskog algoritma (GA) primjenjive za postizanje prihvatljivih rješenja u eksperimentima, kako bi se njima proveli eksperimenti metoda optimiranja,
- odrediti utjecaj parametara veličine populacije i broja generacija GA na učinkovitosti metode optimiranja iskustvenim evaluatorom (D-EX2),
- odrediti ovisnost učinkovitosti metode D-EX2 o vjerojatnosti mutacije jedinki pri provedbi GA,
- odrediti učinkovitost D-EX2 metode uzastopnim izvršavanjem evolucijskog algoritma
- utvrditi doprinos kvaliteti rješenja hibridne metode optimiranja pozicije i vrste zaštitnih elemenata.

5.2. Okruženje u kojem su se obavljali eksperimenti

Za provođenje eksperimenata odabrano je razvojno okruženje HeuristicLab [46]. Članovi udruženja *Heuristic and Evolutionary Algorithms Laboratory* (HEAL) razvijaju ga od 2002.

godine, a izvorni kôd objavljen je u obliku otvorenog kôda. HeuristicLab okruženje osmišljeno je kao razvojni okvir koji omogućuje generičko izvođenje algoritama. Objektno je orijentiran te je svaki element procesa provedbe algoritma opisan modelom (slika 5.1).



Slika 5.1 Vizualno opisivanje algoritma u HeuristicLab razvojnom okruženju

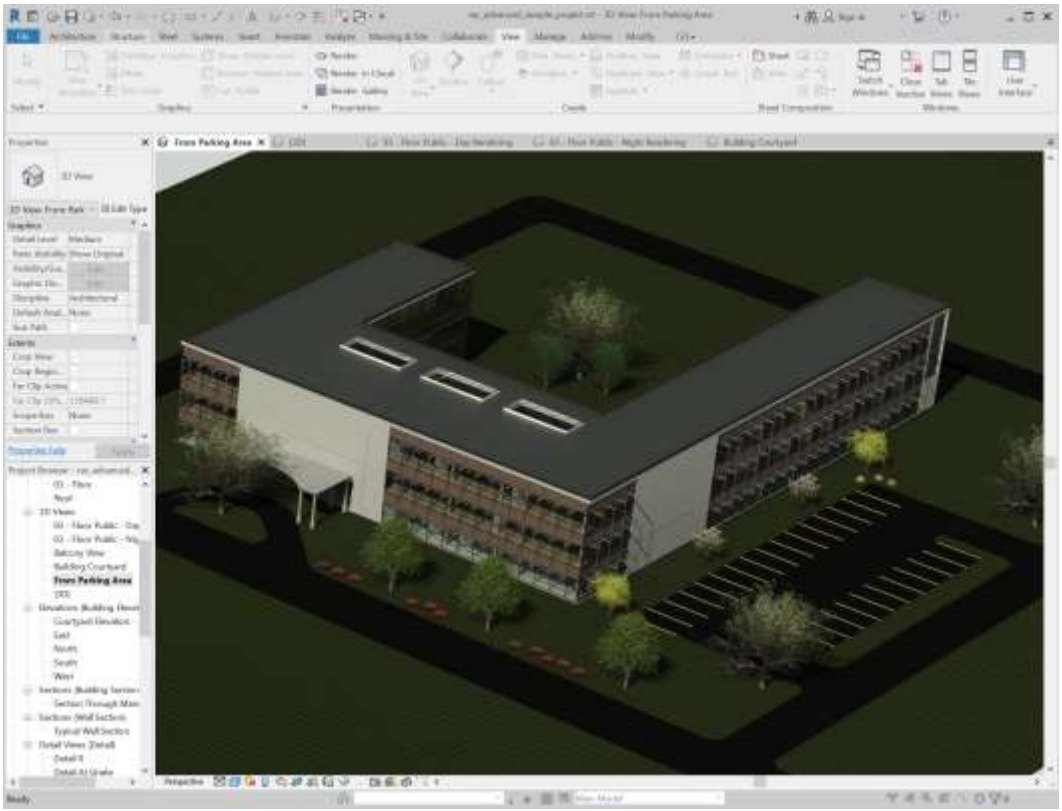
HeuristicLab sadrži implementaciju mnogih algoritama iz područja optimiranja, a implementacija je provjerena kroz višegodišnje korištenje. Algoritmi su implementirani kao skup međusobno povezanih modula čija se implementacija može koristiti u dijelovima različitih algoritama. Uz grafičko uređivanje algoritama, korisniku je omogućeno pisanje programskog koda unutar sučelja koje se izvršava kao jedan od modula. Dodatna mogućnost proširivanja je izrada dodatnih programskih biblioteka koje se dinamički uključuju u razvojno okruženje. Prednost korištenja HeuristicLab okruženja je mogućnost izvođenja svih algoritama u načinu koji izvodi algoritam korak po korak i prikazuje trenutne vrijednosti svih varijabli (*engl. debug*).

Algoritmi kojima se provodi evaluacija i hibridno optimiranje, a proširuje funkcionalnost HeuristicLab okruženja, napisani su u .Net Framework razvojnoj okolini, korištenjem sintakse C# programskog jezika. Za provedbu višekriterijske optimizacije korišten je

elitistički algoritam baziran na nedominantnom sortiranju NSGA-II (*Non-dominated Sorting Genetic Algorithm II*) [44].

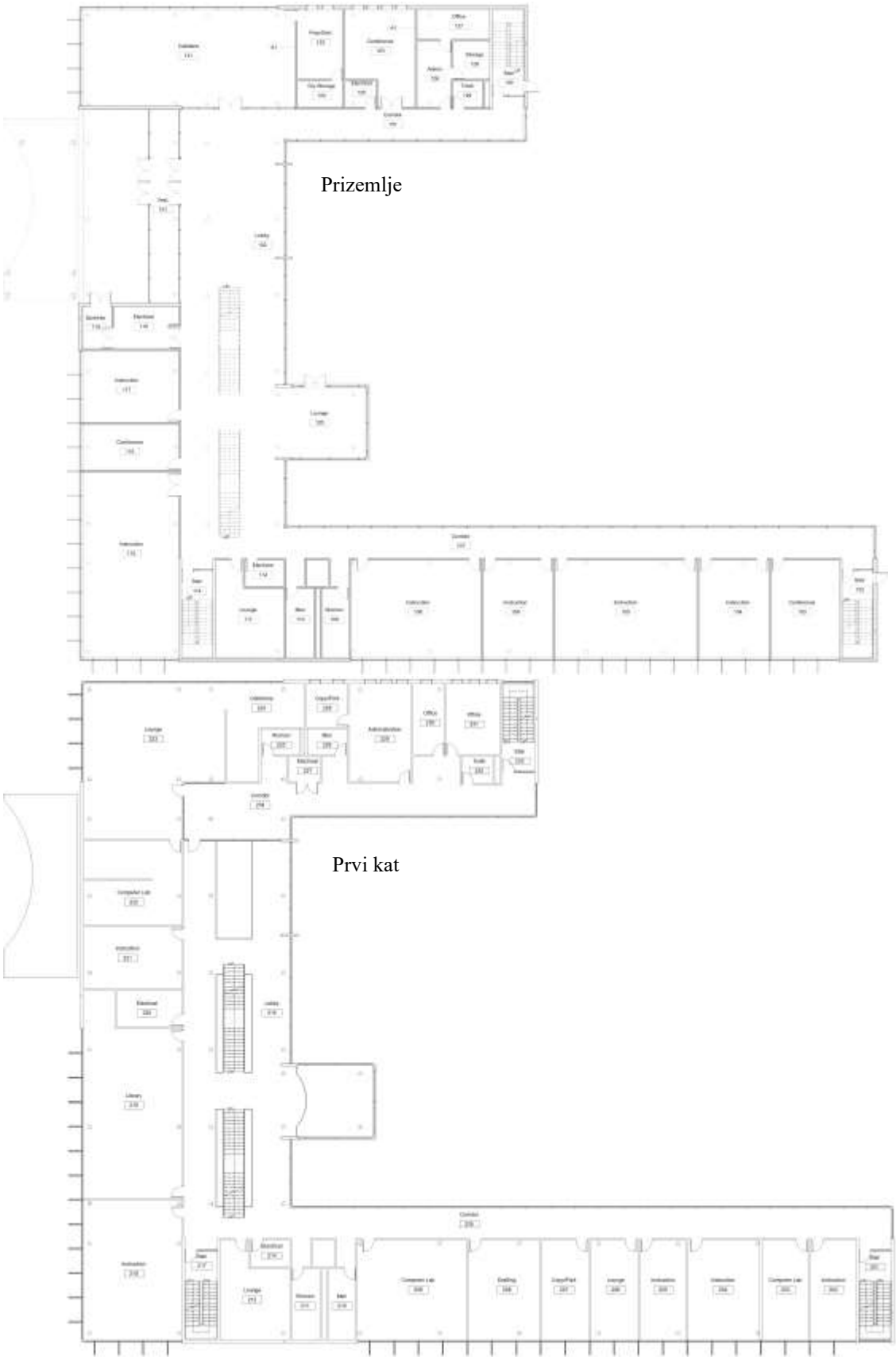
Eksperimenti su podijeljeni u nekoliko grupa. Prva grupa eksperimenata pokazuje utjecaj parametara genetskog algoritma na kvalitetu rješenja. Druga grupa eksperimenata testira utjecaj iskustvenog evaluatora D-EX2 na broj pretraživanja. U trećoj grupi eksperimenata uspoređuje se kvaliteta pronađenog rješenja izvođenjem standardnog genetskog algoritma, koji početnu populaciju stvara slučajnim odabirom i hibridne metode kojom se početna populacija stvara poznavanjem značajki domene pretraživanja.

Testovi su izvođeni na modelu građevine opisane datotekom „RAC_Advanced_sample", koja se distribuira kao dio Autodesk Revit® 2020 programskog paketa. Zgrada se sastoji od tri kata – prizemlje, prvi i drugi kat (slika 5.2).



Slika 5.2 Model građevine "rac_advanced_sample" u Autodesk Revit aplikaciji

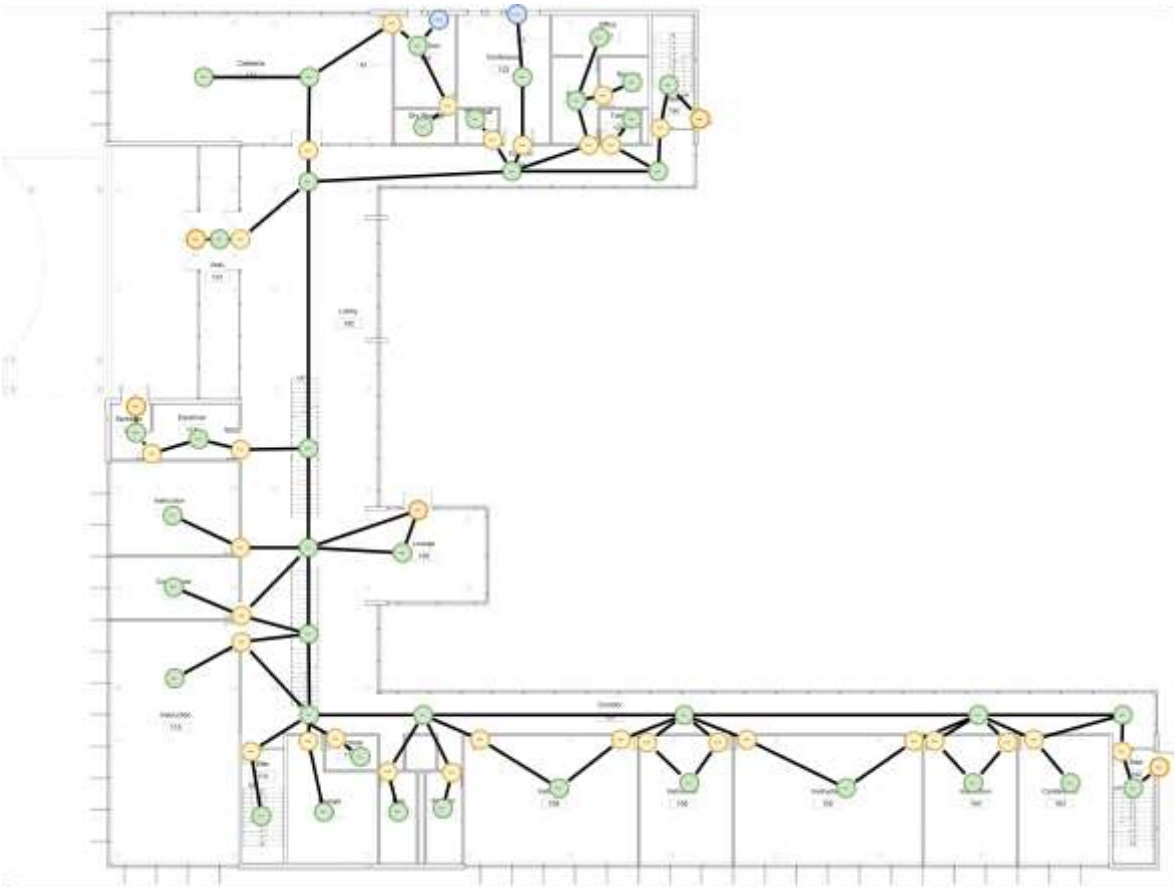
Testni modeli izrađeni su korištenjem prizemlja i prvog kata. Za cilj napada odabrana je prostorija s oznakom broj 222, „Computer room", na prvom katu.



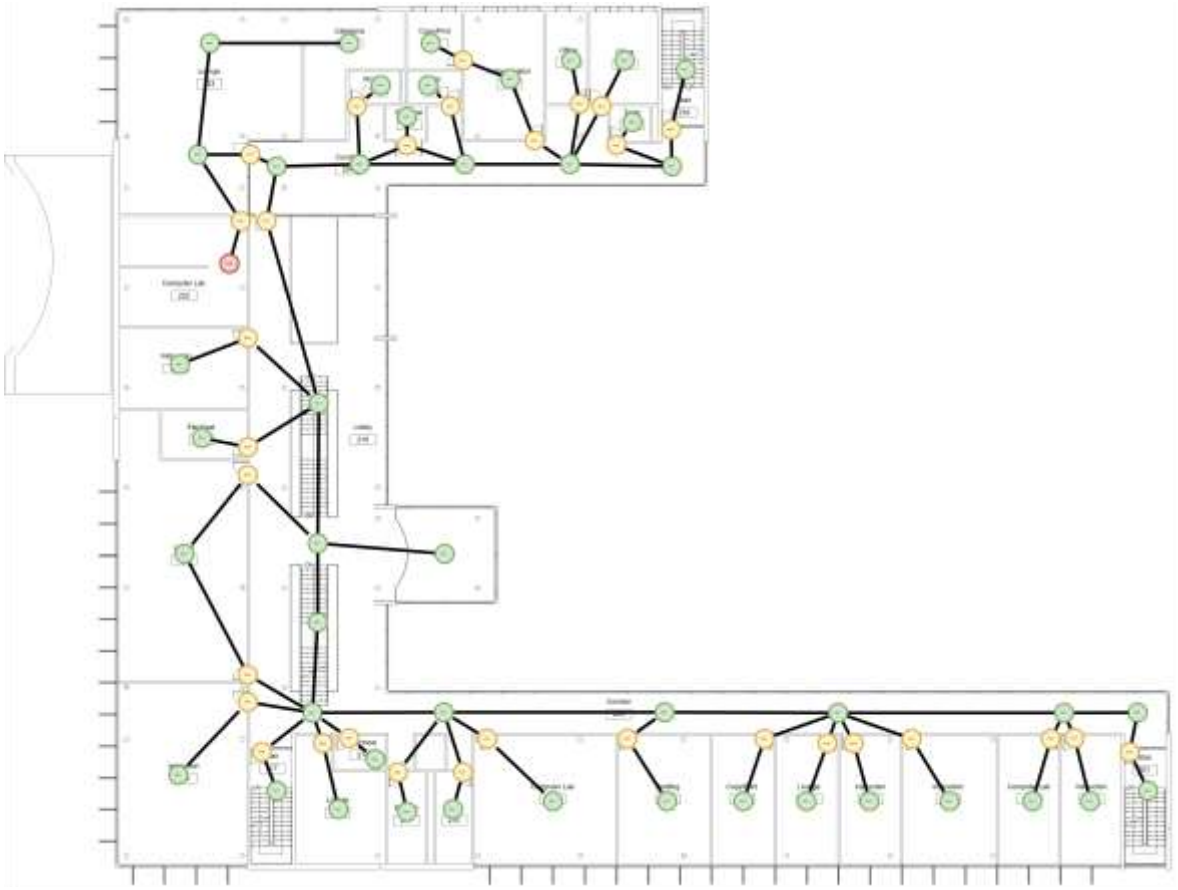
Slika 5.3 Tlocrt prizemlja i prvog kata objekta Autodesk Revit® *RAC_advanced_sample*

Za predmetnu zgradu izrađena su dva grafa mogućeg kretanja napadača. Osnovna je razlika broj elemenata s kojih napadač može započeti napad, ukupan broj elemenata grafa te njegova razgranatost. Čvorovi grafa zgrade predstavljeni su različitim bojama. Narančasti čvor predstavlja vrata koja se mogu iskoristiti za početak napada. Žuti čvorovi predstavljaju vrata unutar objekta. Plavom bojom označeni su prozori koje napadaču mogu poslužiti za početak napada. Čvorovi smješteni u prostorije označeni su zelenom bojom, a ciljni čvor crvenom bojom. Prostorije kojima nije pridijeljen čvor nisu bitne za provođenje optimiranja jer su pripadajući čvorovi izbačeni u procesu optimiranja prostora pretraživanja.

Prvi model (Zgrada A) predstavlja objekt visoke razine sigurnosti (slika 5.4 i slika 5.5). Uz vrata glavnog ulaza i požarnih stepeništa, napadaču su za početak napada dostupna samo dva prozora. Zgrada A opisana je grafom mogućeg kretanja sa 154 čvorova, pri čemu je 7 početnih.

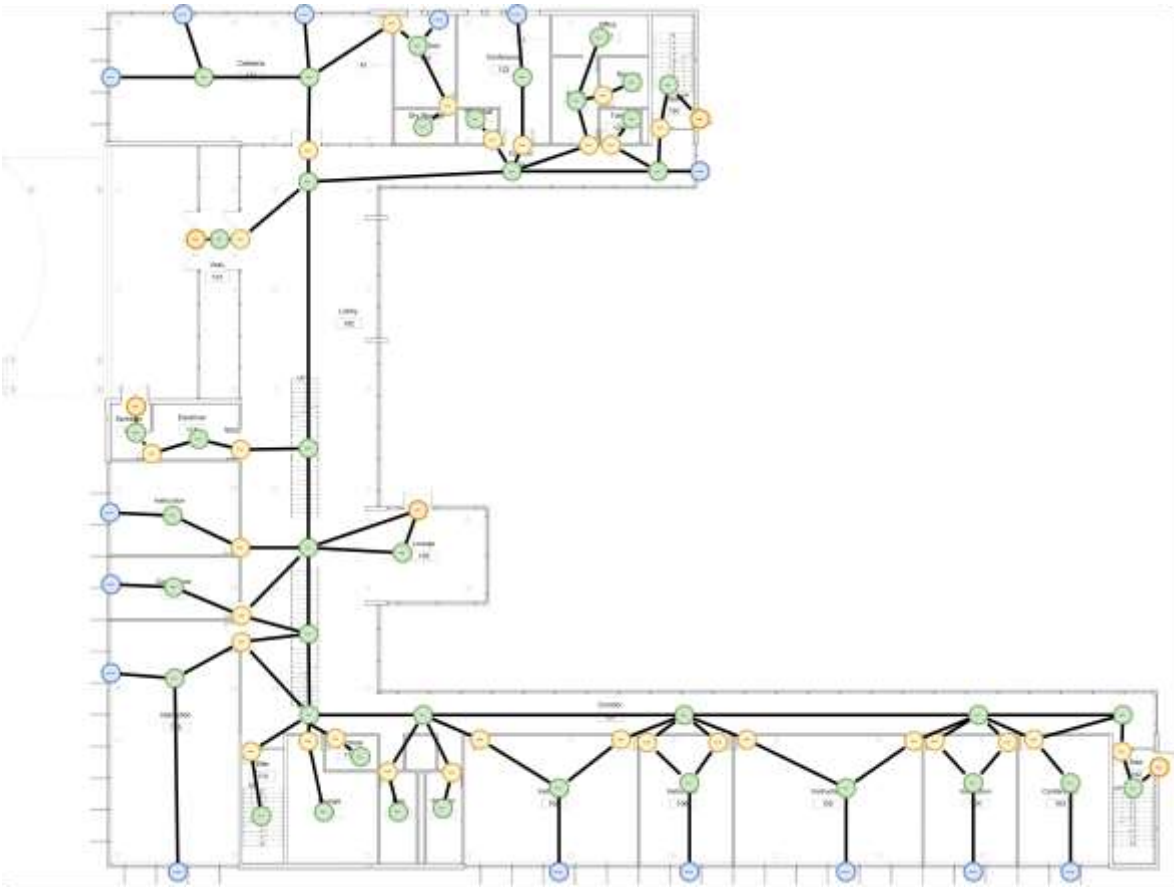


Slika 5.4 Štićeni objekt A, prizemlje



Slika 5.5 Prvi kat štićenih objekata A i B

Drugi model (Zgrada B) je tlocrtno ista zgrada, ali izvedena kao poslovna zgrada čije pročelje u prizemlju je ispunjeno staklenim stijenama (slika 5.6). Uz vrata na glavnom ulazu i požarnim stepeništima, napadaču je moguće započeti napad razbijanjem staklenih elemenata. Zgrada B predstavljena je grafom kojeg čini 167 čvorova, od čega je 20 početnih. Prvi kat zgrade B jednak je modelu zgrade A (slika 5.5).



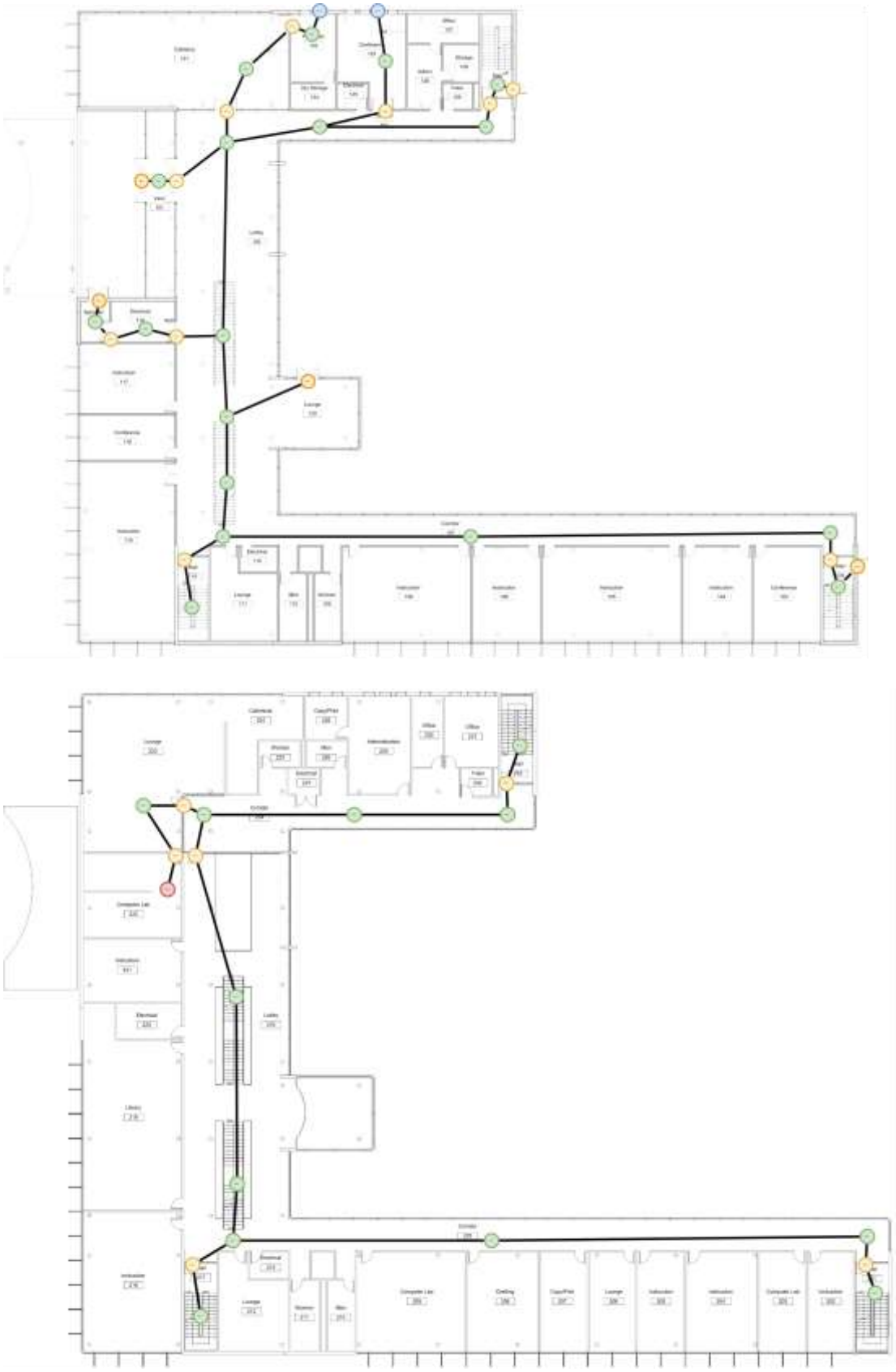
Slika 5.6 Štićeni objekt B, prizemlje

5.3. Provedeni eksperimenti

5.3.1. Optimiranje grafa mogućeg kretanja napadača

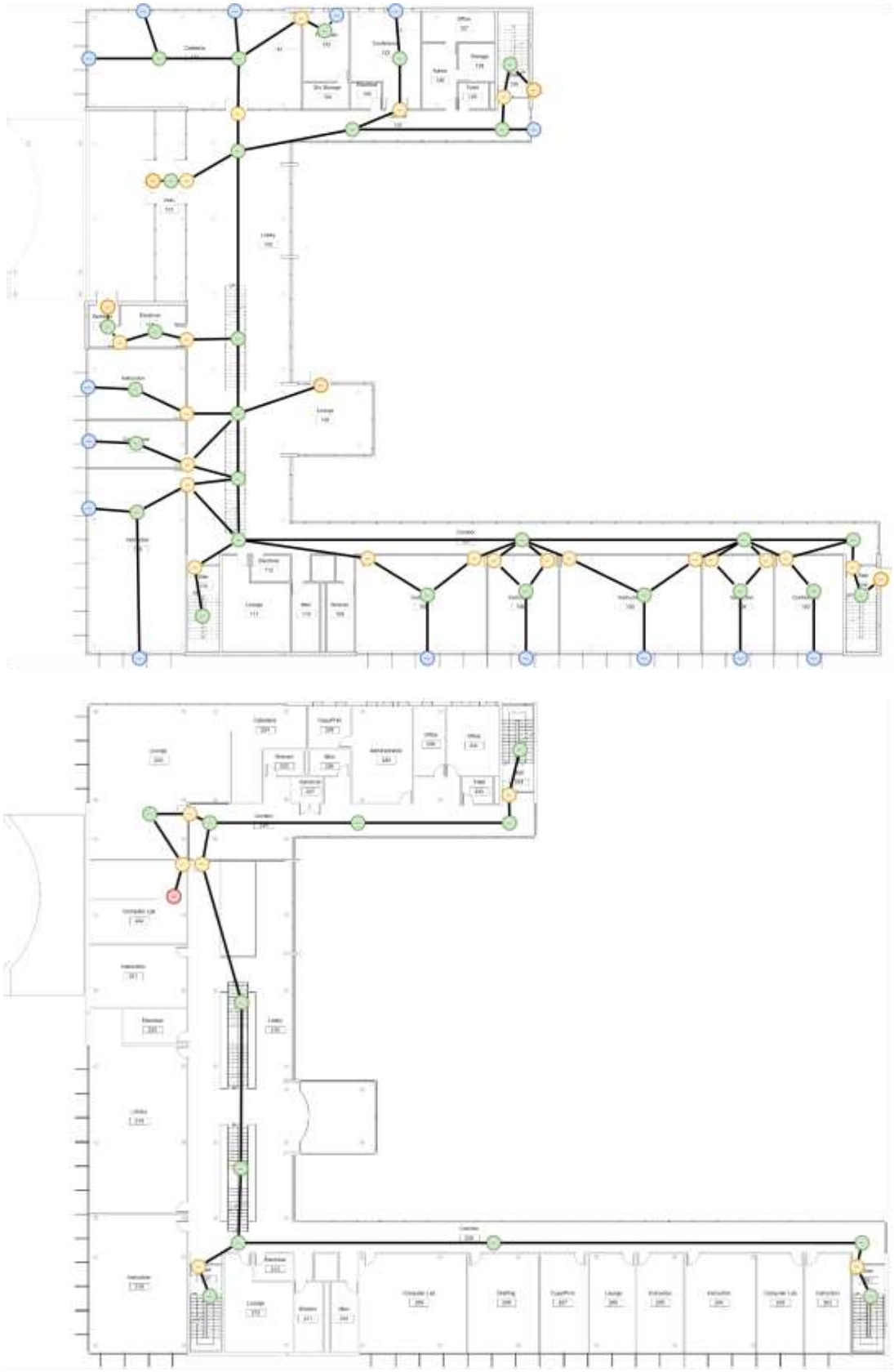
Algoritam optimiranja grafa mogućeg kretanja napadača prvi je korak u cjelokupnom procesu optimiranja odabira i rasporeda zaštitnih elemenata i dio je hibridne metode optimiranja pozicije i vrste zaštitnih elemenata. Algoritam pretražuje čvorove koji napadaču u nijednom scenariju nisu korisni za kretanje. Učinkovitost algoritma ispitana je nad objektom A i B.

Optimiranjem grafa mogućeg kretanja napadača za objekt A, graf kretanja je optimiran s početnih 154 čvorova na 53 čvora, kako prikazuje slika 5.7. Prostor pretraživanja je ovim korakom smanjen s $8{,}3 \times 10^{139}$ na $5{,}1 \times 10^{48}$, uz 3 dostupna zaštitna elementa po lokaciji.



Slika 5.7 Optimiran graf mogućeg kretanja prostorom - Testna zgrada A

Graf kretanja za objekt B je provedenim optimiranjem smanjen s početnih 167 čvorova na 88 čvorova, kako prikazuje slika 5.8. Ovim korakom optimiranja prostor pretraživanja smanjen je s $1{,}3 \times 10^{152}$ na $5{,}9 \times 10^{80}$, uz 3 dostupna zaštitna elementa po lokaciji.



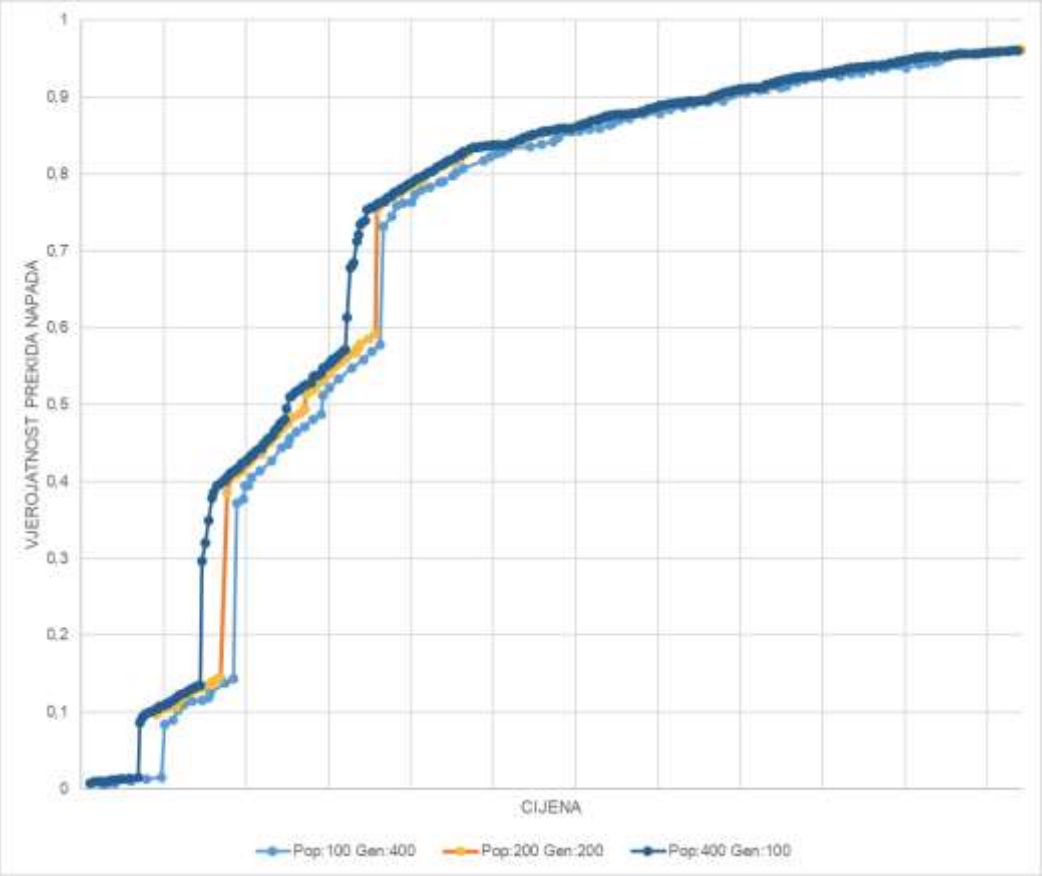
Slika 5.8 Optimiran graf mogućeg kretanja prostorom - Testna zgrada B

5.3.2. Eksperimentalni rezultati metode optimiranja iskustvenim evaluatorom

Metoda optimiranja iskustvenim evaluatorom provedena je nad objektima A i B različitim postavkama genetskog algoritma (GA) kako bi se utvrdila ovisnost učinkovitosti optimiranja o veličini populacije, broju generacija, vjerojatnosti mutacije i konfiguraciji grafa nad kojim se provodi optimiranje. Znanje iskustvenog evaluatora stečeno u ranijim izvođenjima algoritma koristi se za izbjegavanje izračuna pri evaluaciji

5.3.2.1 Utjecaj veličine populacije i broja generacija

Prvi eksperiment proveden nad objektom A izveden je s 40.000 jedinki, a izvršen u tri različite konfiguracije. Slika 5.9 prikazuje Pareto rješenja za konfiguraciju s veličinom populacije 100 jedinki i 400 generacija, populacijom od 200 jedinki i 200 generacija te populacijom od 400 jedinki kroz 100 generacija. Zajednički im je jednak broj izvedenih evaluacija, koji iznosi 280.000. Konfiguracija sa 400 jedinki u populaciji kroz 100 generacija daje najbolje rezultate kroz cijelo područje rješenja.



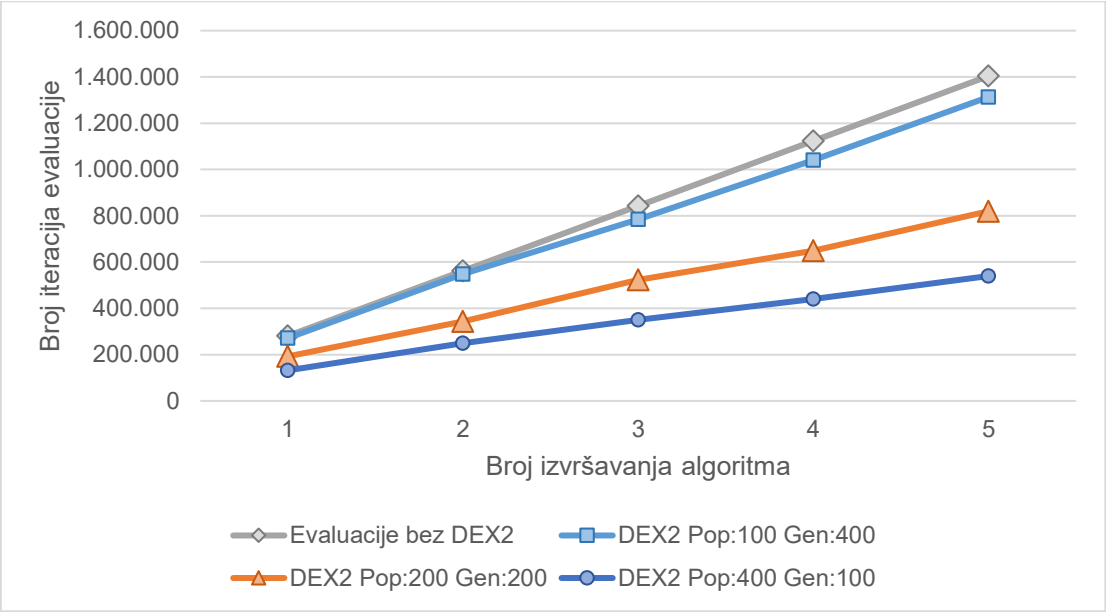
Slika 5.9 Rezultati optimiranja za testnu zgradu A s 40.000 jedinki

Tablica 5.1 prikazuje usporedbu standardnog izvođenja optimiranja za navedene konfiguracije te učinkovitost optimiranja iskustvenim evaluatorom. Testiranje je provedeno višestrukim izvođenjem algoritma pa prva kolona prikazuje redni broj izvođenja. Druga kolona prikazuje redovni broj evaluacija, pribrajan kroz sva izvođenja. Treća kolona je suma stvarno izvedenih izračuna, kojih je manje od redovnog broja jer se za evaluaciju koristi algoritam D-EX2. Postotna vrijednost smanjenja broja evaluacija u odnosu na standardno izvođenje prikazana je u četvrtoj koloni.

Tablica 5.1 Usporedba broja evaluacija za 40.000 jedinki, objekt A

Algoritam:	NSGA-II, Veličina populacije: 100, Generacija: 400, Vjerojatnost mutacije: 0%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	282.800	271.333	4%
2	565.600	547.453	2%
3	848.400	784.012	16%
4	1.131.200	1.040.783	9%
5	1.414.000	1.312.938	4%
Algoritam:	NSGA-II, Veličina populacije: 200, Generacija: 200, Vjerojatnost mutacije: 0%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	281.400	192.491	32%
2	562.800	343.488	46%
3	844.200	522.707	36%
4	1.125.600	649.627	55%
5	1.407.000	819.469	40%
Algoritam:	NSGA-II, Veličina populacije: 400, Generacija: 100, Vjerojatnost mutacije: 0%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	280.700	131.940	53%
2	561.400	249.411	58%
3	842.100	349.638	64%
4	1.122.800	440.771	68%
5	1.403.500	539.622	65%

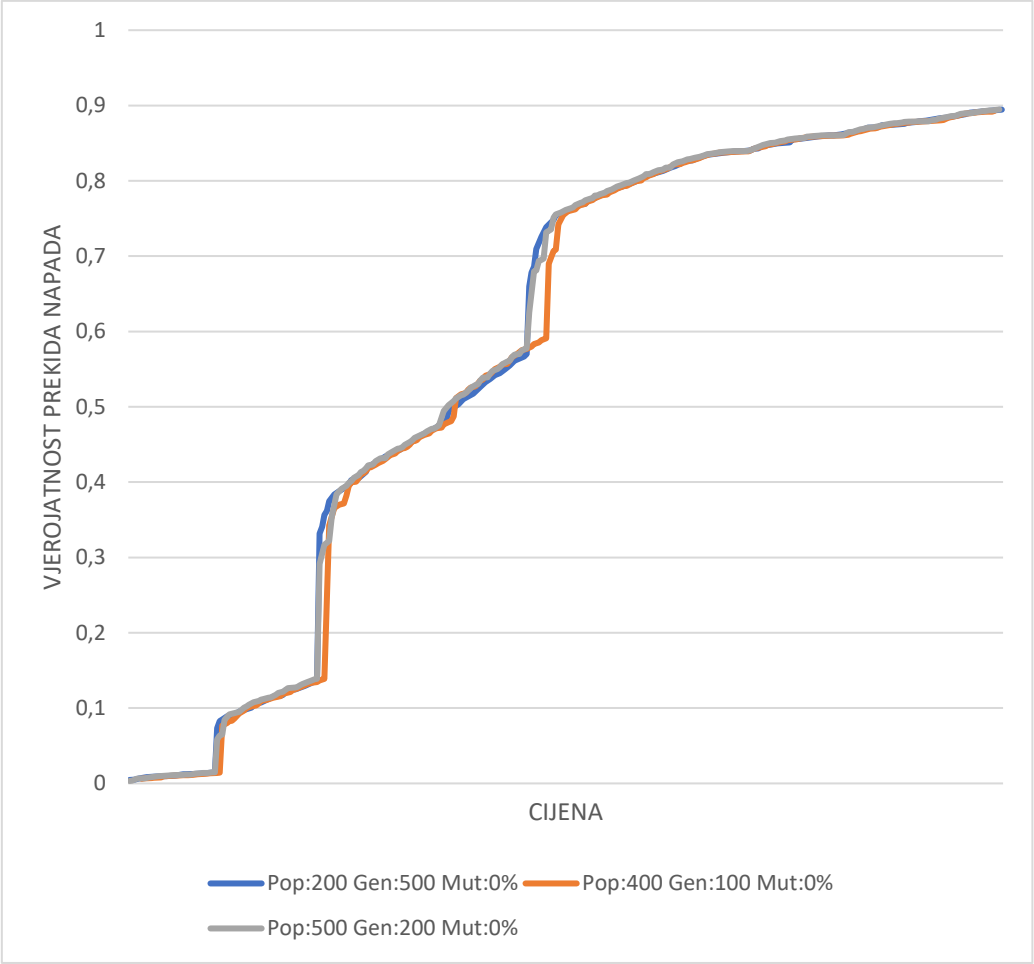
Slika 5.10 prikazuje grafom odnose evaluacija za tri konfiguracije genetskih algoritama, optimiranih D-EX2 metodom.



Slika 5.10 Broj evaluacija za 40.000 jedinki, objekt A

Iz rezultata eksperimenta vidljivo je da veća populacija, uz manji broj generacija, otežava algoritmu iskorištavanje stečenog znanja o objektu koji se optimira. Rezultat je očekivan jer velik broj različitih jedinki stvara i široko stablo pretraživanja D-EX2 evaluatora, a manji broj generacija ne pruža priliku iskoristiti pronađena rješenja i proširiti stablo pretraživanja u dubinu. Veća učinkovitost evaluatora postiže se kod konfiguracije genetskog algoritma koja sadrži populaciju s manje jedinki, a veći broj generacija, kako je vidljivo za konfiguraciju sa 100 jedinki i 400 generacija.

Za provjeru učinkovitosti D-EX2 metode izvedeni su i eksperimenti sa 100.000 jedinki za koje se standardnim GA izvodi 701.400 evaluacija. Veći broj jedinki doprinosi kvalitetnijem rješenju u odnosu na optimiranje s 40.000 jedinki (slika 5.11).



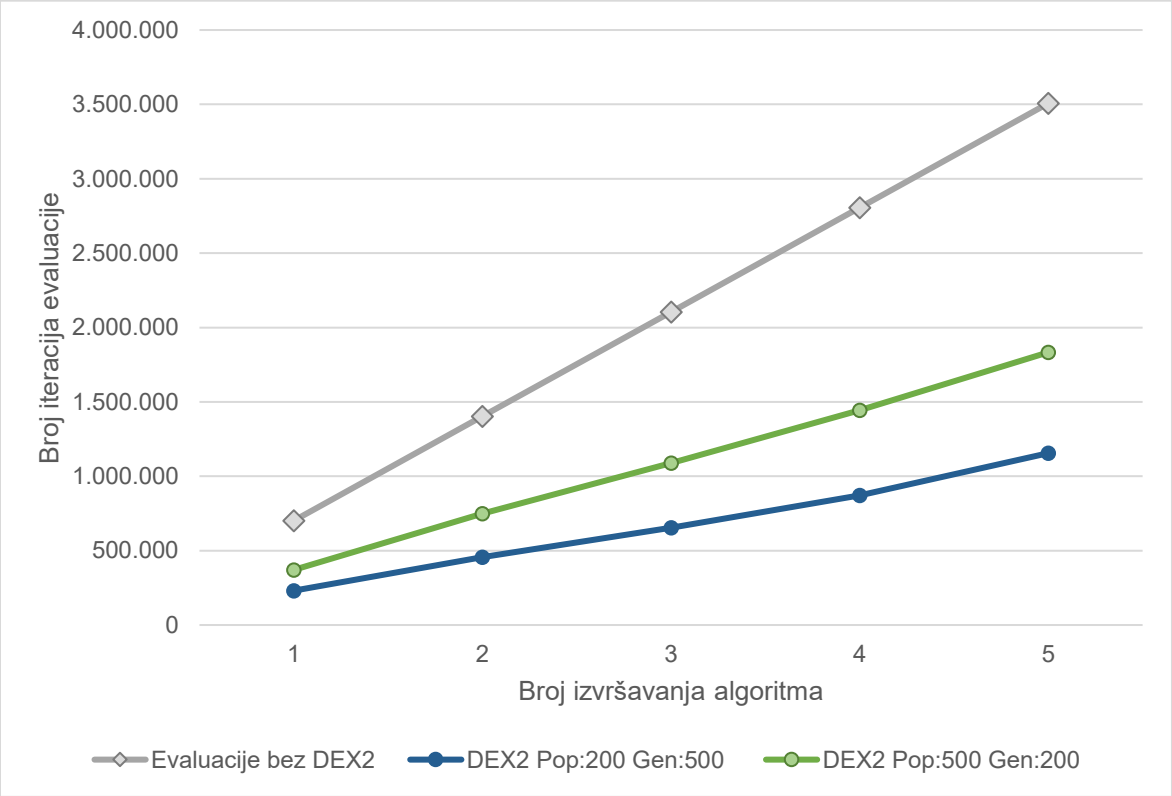
Slika 5.11 Kvaliteta rješenja različitih konfiguracija GA s 100.00 jedinki, objekt A

Tablica 5.2 prikazuje rezultate eksperimenta s 100.000 jedinki, uspoređujući broj izračuna primjenom D-EX2 metode u odnosu na standardno provođenje evaluacije. Velik broj generacija omogućuje D-EX2 metodi iskorištavanje znanja stečenog u ranijem izvršavanju algoritma čime se značajno izbjegavaju nepotrebni izračuni. Za konfiguraciju GA s populacijom od 200 jedinki i 500 generacija, prosječno izbjegavanje izračuna rješenja iznosi 67%.

Tablica 5.2 Usporedba broja evaluacija za 100.000 jedinki, objekt A

Algoritam:	NSGA-II, Veličina populacije: 200, Generacija: 500, Vjerojatnost mutacije: 0%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	701.400	230.896	67%
2	1.402.800	456.912	68%
3	2.104.200	655.155	72%
4	2.805.600	871.064	69%
5	3.507.000	1.155.096	60%
Algoritam:	NSGA-II, Veličina populacije: 500, Generacija: 200, Vjerojatnost mutacije: 0%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	703.500	370.631	47%
2	1.407.000	749.893	46%
3	2.110.500	1.087.904	52%
4	2.814.000	1.444.753	49%
5	3.517.500	1.833.089	45%

Grafički prikaz razlike u broju izvedenih izračuna za konfiguracije s 100.000 jedinki prikazuje slika 5.12.



Slika 5.12 Učinkovitost D-EX2 metode - 100.000 jedinki, objekt A

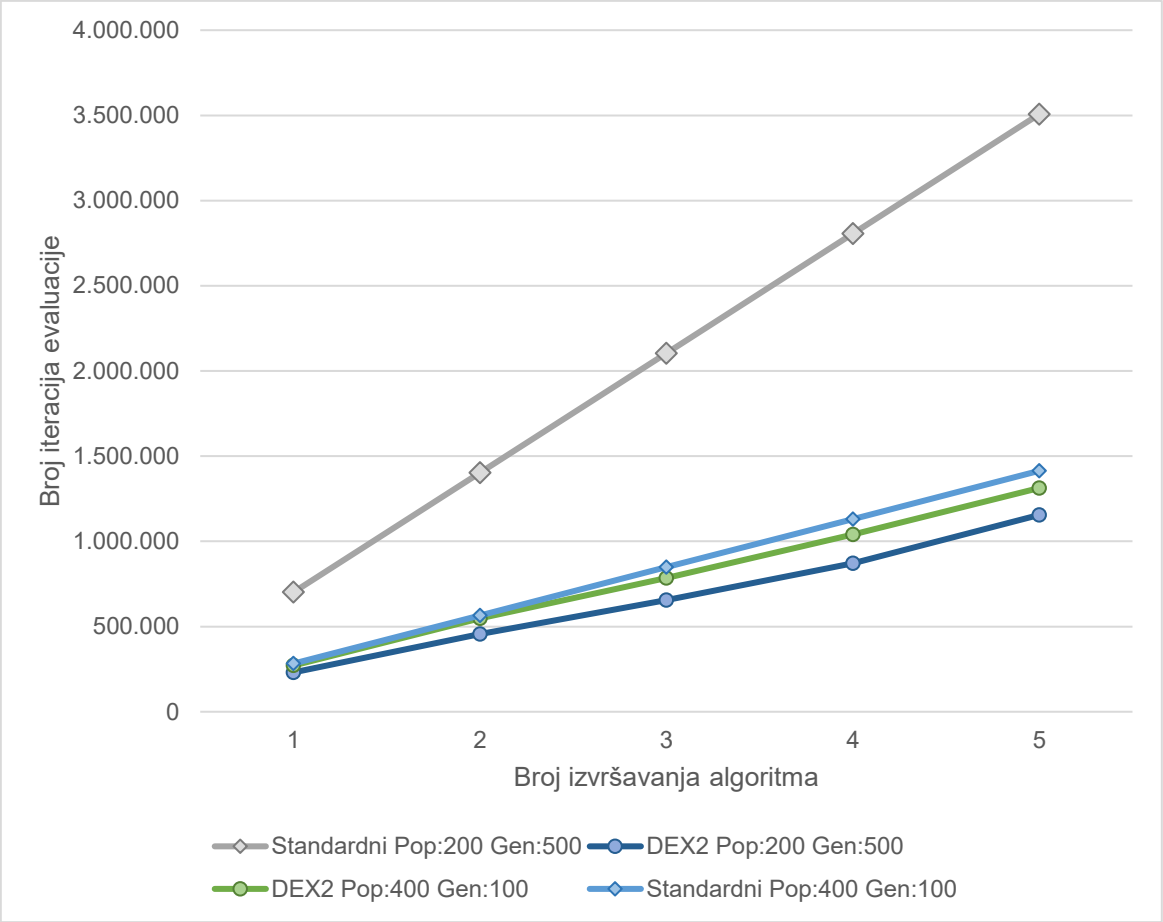
Usporede li se rezultati eksperimenata, dolazi do izražaja učinkovitost D-EX2 metode jer za pronalazak kvalitetnijih rješenja, uz 200 jedinki i 500 generacija, koristi manji broj evaluacija u odnosu na standardno izvođenje GA sa 400 jedinki i 100 generacija (tablica 5.3). D-EX2 omogućuje provedbu algoritma sa 100.000 jedinki u manje iteracija od standardnog GA sa 40.000 jedinki. Uzastopnih pet eksperimenata s 100.000 evaluacija jedinki izvršeno je, zahvaljujući D-EX2 metodi, u svakom od pet koraka s manje izračuna kvalitete rješenja (ukupno 1.155.096), u odnosu na konfiguraciju GA sa 40.000 jedinki (ukupno 1.312.938).

Razlog za bolje rezultate je veći broj generacija koje omogućuju lokalna optimiranja predloženih rješenja i time bolje korištenje ranije pronađenih rješenja.

Tablica 5.3 Učinkovitost D-EX2 metode, objekt A

	NSGA-II, Veličina populacije: 400, Generacija: 100, Vjerojatnost mutacije: 0%, Vjerojatnost križanja: 90%		NSGA-II, Veličina populacije: 200, Generacija: 500, Vjerojatnost mutacije: 0%, Vjerojatnost križanja: 90%	
Pretraživanje br.	Ukupan broj standardnih evaluacija	Ukupan broj izračuna uz D-EX2	Ukupan broj standardnih evaluacija	Ukupan broj izračuna uz D-EX2
1	282.800	271.333	701.400	230.896
2	565.600	547.453	1.402.800	456.912
3	848.400	784.012	2.104.200	655.155
4	1.131.200	1.040.783	2.805.600	871.064
5	1.414.000	**1.312.938**	3.507.000	**1.155.096**

Manji broj izračuna metodom D-EX2 za konfiguraciju sa 40.000 jedinki u odnosu na 100.000 jedinki prikazuje slika 5.13.



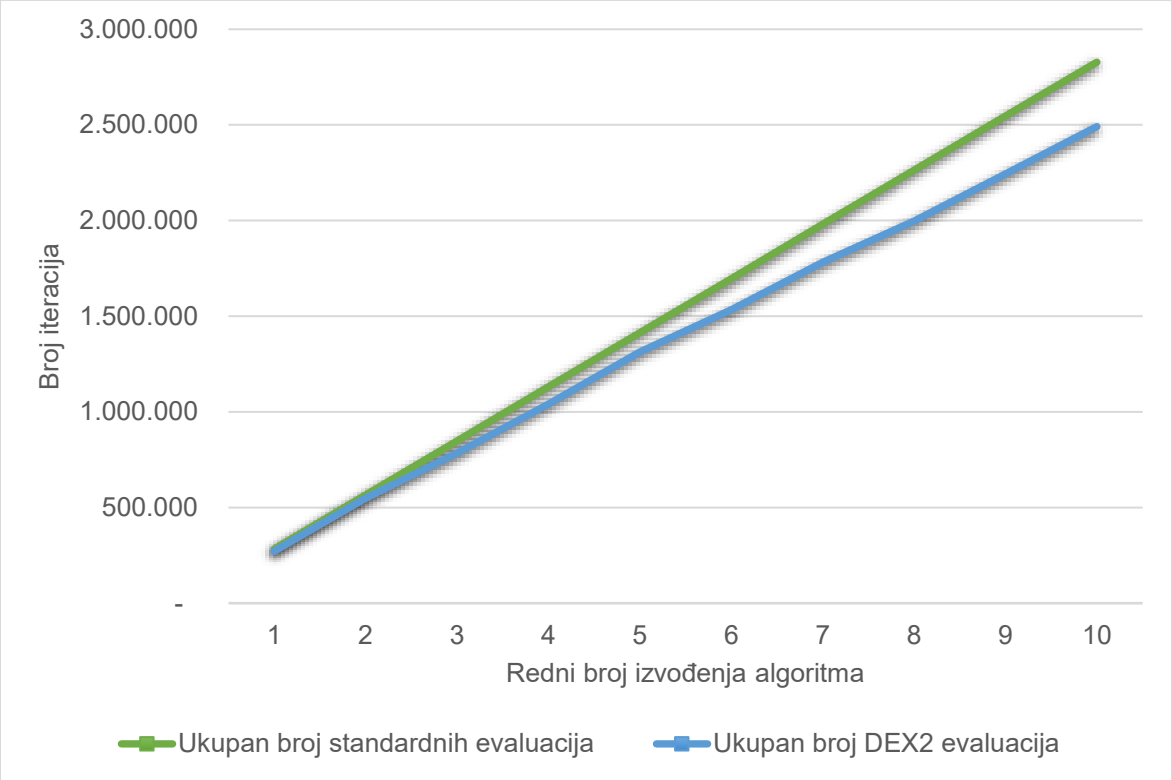
Slika 5.13 Usporedba rezultata primjene D-EX2 algoritma

5.3.2.2 Utjecaj uzastopnog izvođenja evolucijskog algoritma

Višestrukim izvođenjem D-EX2 metode povećava se stablo pretraživanja. Tablica 5.4 prikazuje povećanje prosjeka izbjegnutih izračuna rješenja D-EX2 metodom za više uzastopnih izvršavanja algoritma. U prvih pet izvođenja prosjek izbjegnutih izračuna je 7.15%, a drugih pet izvođenja 16.7% s vidljivom tendencijom rasta (slika 5.14).

Tablica 5.4 Usporedba broja evaluacija za više uzastopnih izvođenja algoritma

Algoritam:	NSGA-II, Veličina populacije: 400, Generacija: 100, Vjerojatnost mutacije: 0%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	282.800	271.333	4%
2	565.600	547.453	2%
3	848.400	784.012	16%
4	1.131.200	1.040.783	9%
5	1.414.000	1.312.938	4%
6	1.696.800	1.533.893	22%
7	1.979.600	1.780.605	13%
8	2.262.400	1.997.693	23%
9	2.545.200	2.242.989	13%
10	2.828.000	2.491.094	12%



Slika 5.14 Broj evaluacija za više uzastopnih izvođenja

5.3.2.3 Utjecaj parametra mutacije

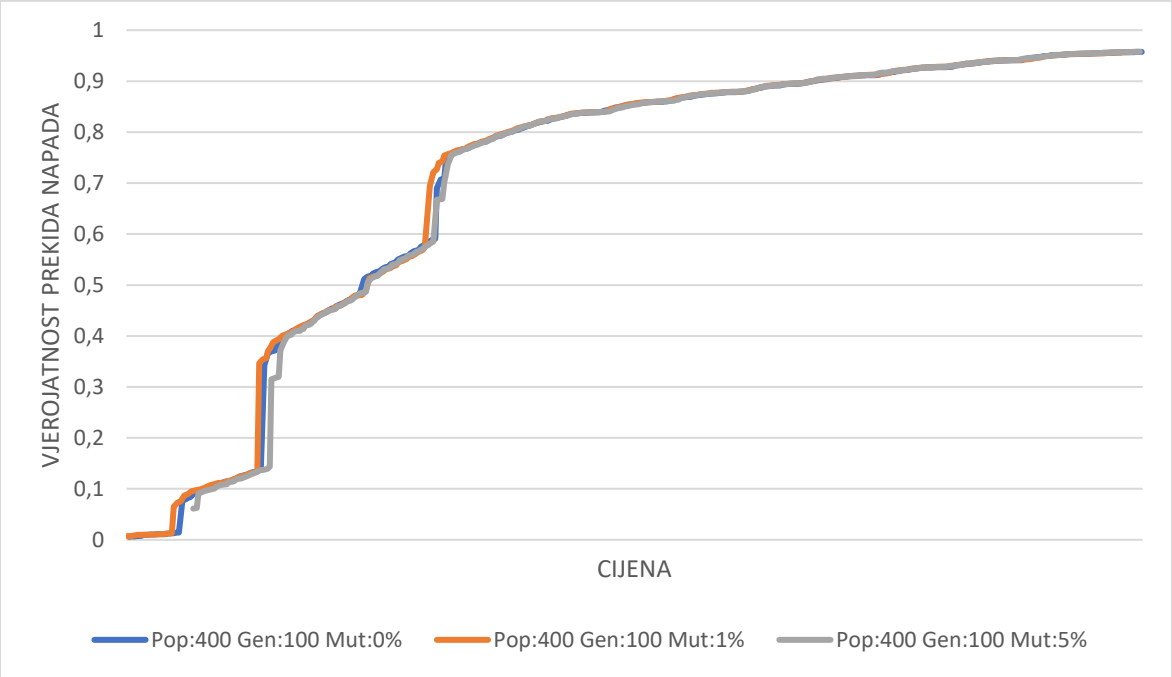
Mutacijom jedinki povećava se vjerojatnost raznolikosti rješenja u svakoj novoj generaciji. Utjecaj na kvalitetu rješenja nije vidljiv u svim konfiguracijama GA već ovisi i

o vjerojatnosti mutacije, veličini populacije i broju generacija. Općenito se kod većih populacija obavlja veći broj iteracija, a umanjuje vjerojatnost mutacije.

Kako prikazuje slika 5.15, primjena mutacije jedinki pri GA, koji se izvodi nad prostorom pretraživanja za objekt A, s 400 jedinki i 100 generacija, donosi dodatnu kvalitetu rješenja ako je mutacija jedinki 1%. Iako daljnje povećanje vjerojatnosti mutacije ne pridonosi kvaliteti rješenja, već je smanjuje, dodatno je proveden eksperiment s ekstremnom vrijednosti mutacije 5% kako bi se utvrdio utjecaj na učinkovitost metode D-EX2. Mutacija jedinki umanjuje učinkovitost D-EX2 metode jer mutiranje dijela jedinki izaziva dodatno širenje stabla pretraživanja, koje se zbog malog broja generacija slabo iskorištava (tablica 5.5).

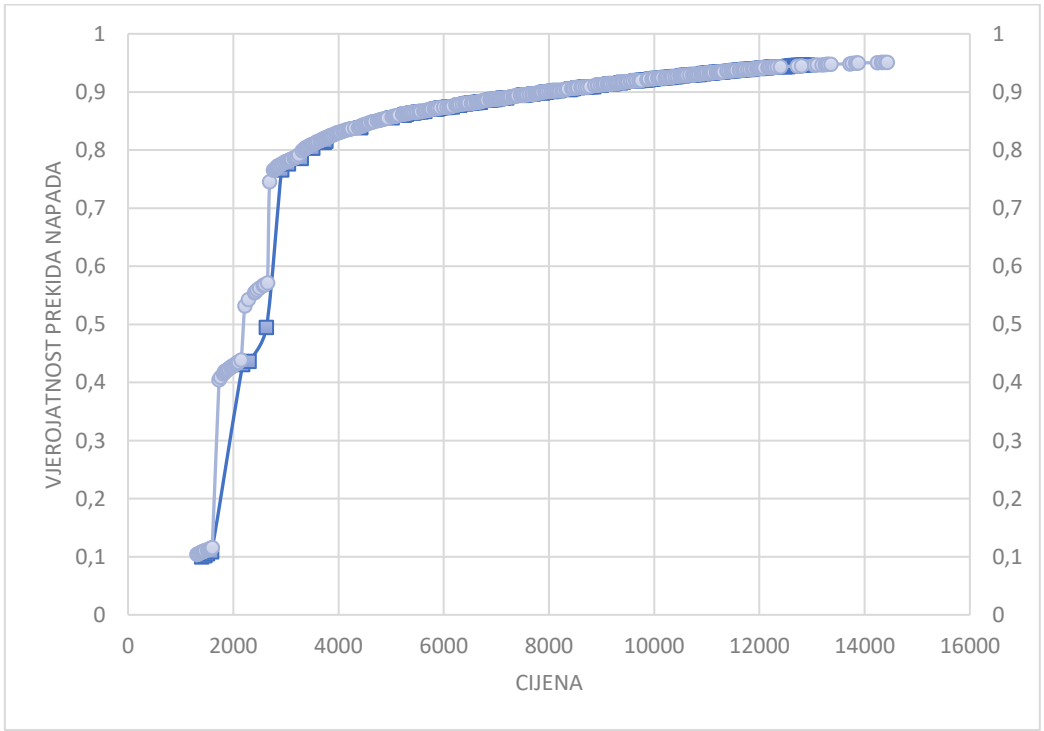
Tablica 5.5 Utjecaj mutacije jedinki na D-EX2 metodu

Algoritam:	NSGA-II, Veličina populacije: 400, Generacija: 100, Vjerojatnost mutacije: 1%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	282.800	261.671	7,47%
2	565.600	500.665	15,49%
3	848.400	783.906	-0,16%
4	1.131.200	1.019.911	16,55%
5	1.414.000	1.267.654	12,40%
	Ukupno:		**10,35%**
Algoritam:	NSGA-II, Veličina populacije: 400, Generacija: 100, Vjerojatnost mutacije: 5%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	282.800	292.524	-3,44%
2	565.600	579.902	-1,62%
3	848.400	844.279	6,51%
4	1.131.200	1.094.091	11,66%
5	1.414.000	1.363.633	4,69%
	Ukupno:		**3,56%**



Slika 5.15 Utjecaj mutacije jedinki na kvalitetu rješenja s malim brojem generacija, objekt A

Slika 5.16 prikazuje na primjeru objekta B poboljšanje kvalitete rješenja kada GA koristi mutaciju.



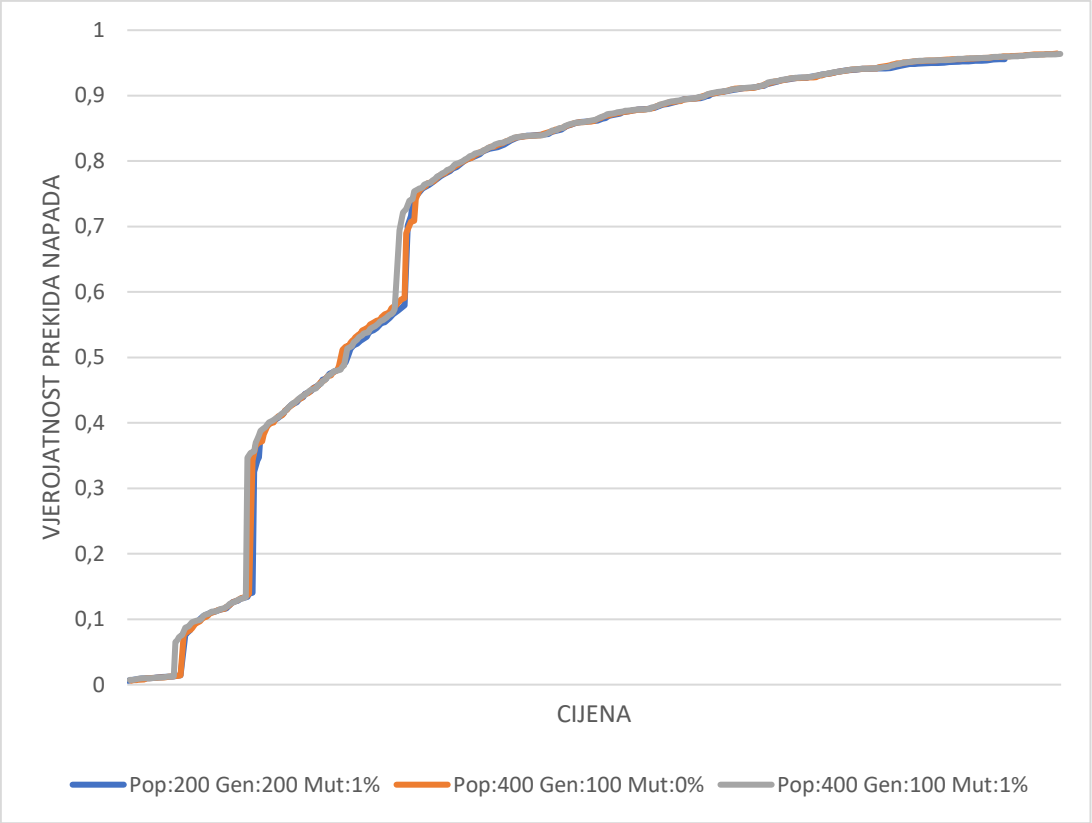
Slika 5.16 Rezultati optimiranja zaštite objekta B

U odnosu na raniju konfiguraciju GA, u ovom eksperimentu se koristi veća populacija i više generacija, ali je smanjenja učinkovitost D-EX2 i dalje prisutna (tablica 5.6).

Tablica 5.6 Eksperimentalni rezultati mutacije za objekt B

Algoritam:	NSGA-II, Veličina populacije: 500, Generacija: 200, Vjerojatnost mutacije: 0%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	2.010.000	1.087.000	46%
2	4.020.000	2.197.108	45%
3	6.030.000	3.506.118	42%
4	8.040.000	4.623.331	42%
5	10.050.000	5.755.194	43%
Algoritam:	NSGA-II, Veličina populacije: 500, Generacija: 200, Vjerojatnost mutacije: 5%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	2.010.000	1.524.477	24%
2	4.020.000	3.171.812	21%
3	6.030.000	4.595.692	24%
4	8.040.000	6.099.798	24%
5	10.050.000	7.355.339	27%

Za objekt A metoda mutacije dijela jedinki pokazala se učinkovitom za konfiguraciju genetskog algoritma s 200 jedinki, 200 generacija i 1% mutacije. Kako prikazuje slika 5.17, kvaliteta pronađenih rješenja približno je jednaka konfiguraciji 400 jedinki i 100 generacija bez mutacija i nešto lošija u srednjem području vjerojatnosti prekida od konfiguracije s 400 jedinki i 100 generacija, uz mutaciju 1%.



Slika 5.17 Usporedba Pareto rješenja s mutacijom jedinki, objekt A

Međutim, D-EX2 metoda značajno umanjuje broj pretraživanja konfiguracije 200 jedinki, 200 generacija, 1% mutacije u odnosu na 400 jedinki i 100 generacija bez i s 1% mutacije (tablica 5.7) što čini takvu konfiguraciju boljim odabirom za optimiranje objekta A.

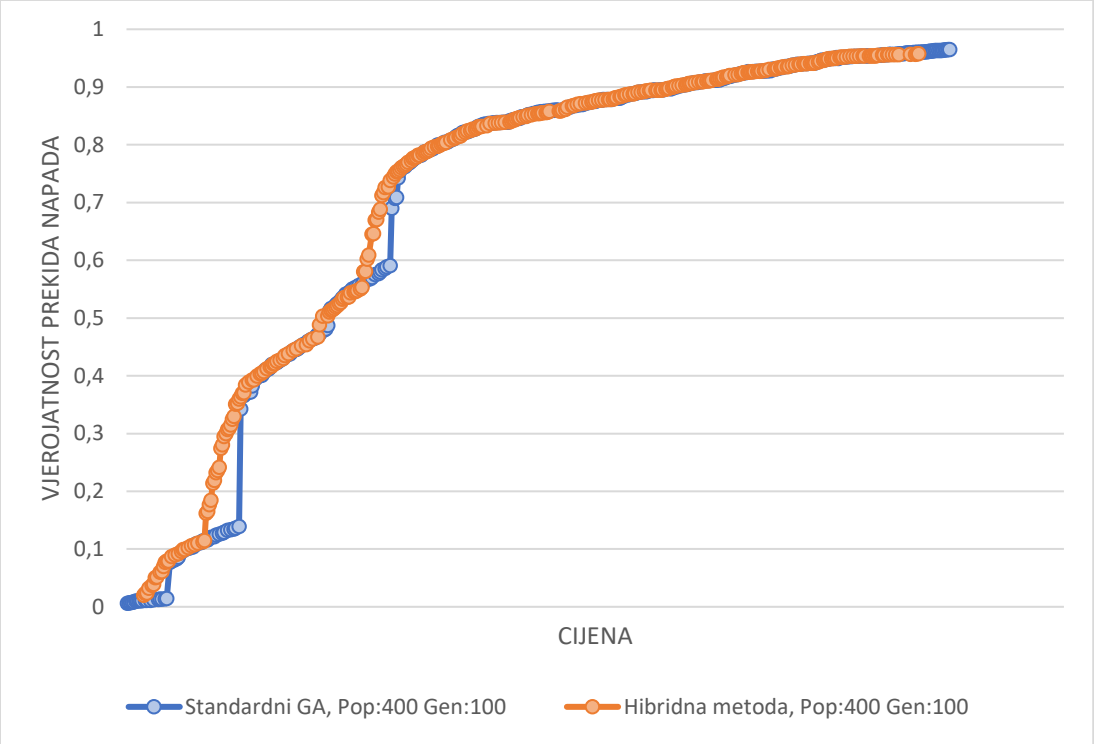
Tablica 5.7 Utjecaj mutacije jedinki na D-EX2 metodu, P200×G200

Algoritam:	NSGA-II, Veličina populacije: 200, Generacija: 200, Vjerojatnost mutacije: 1%, Vjerojatnost križanja: 90%		
Pretraživanje br.	Ukupan broj evaluacija	Ukupan broj izračuna uz D-EX2	Izbjegnuto izračuna rješenja
1	281.400	174.413	38,02%
2	562.800	379.814	27,01%
3	844.200	553.080	38,43%
4	1.125.600	752.904	28,99%
5	1.407.000	901.587	47,16%
	Ukupno:		**35,92%**

5.3.3. Eksperimentalni rezultati inicijalizacije populacije korištenjem domenskog znanja

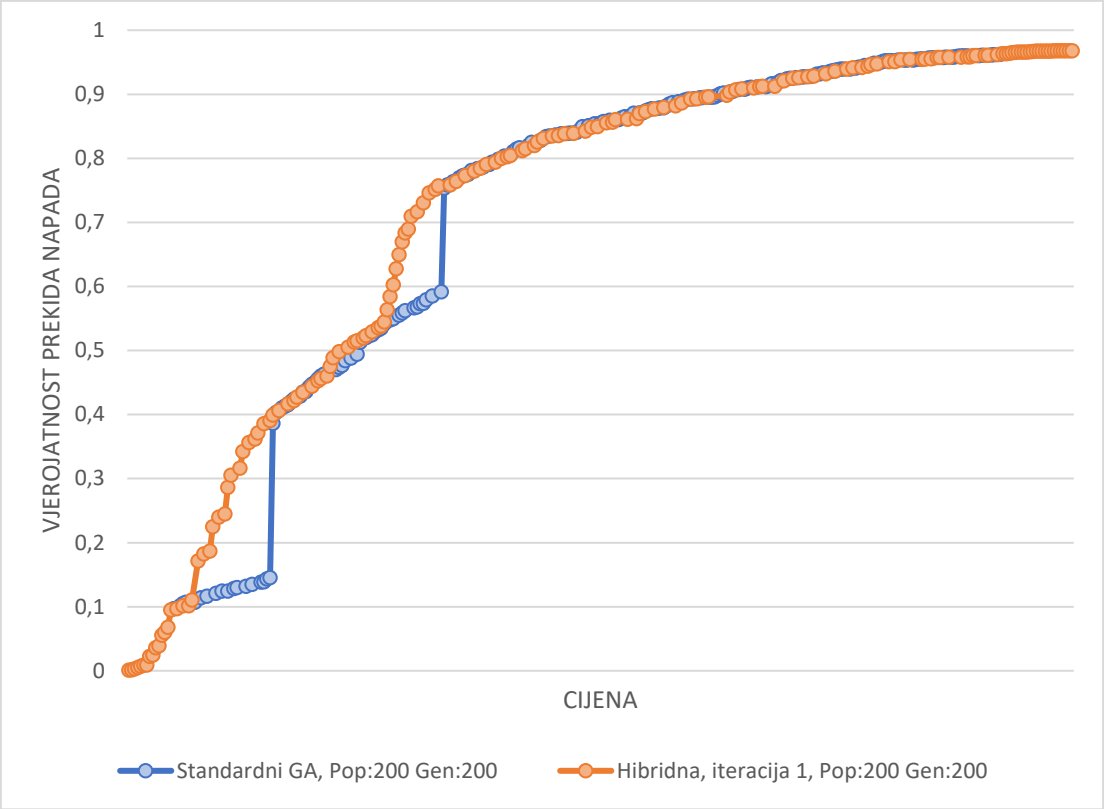
Provedeno je više eksperimenata inicijalizacije populacije korištenjem domenskog znanja, koja čini dio hibridne metode optimiranja pozicije i vrste zaštitnih elemenata. Njima se utvrđuje doprinos metode kvaliteti rješenja u odnosu na standardno izvođenje evolucijskih algoritama pri kojima se početna rješenja stvaraju slučajnim odabirom.

Za konfiguraciju GA 400 jedinki i 100 generacija hibridna metoda, koja inicijalno stvara nekoliko kvalitetnih jedinki, za rezultat daje rješenja koja dominiraju duž gotovo cijele Pareto fronte, u odnosu na standardno provedeni GA (slika 5.18).



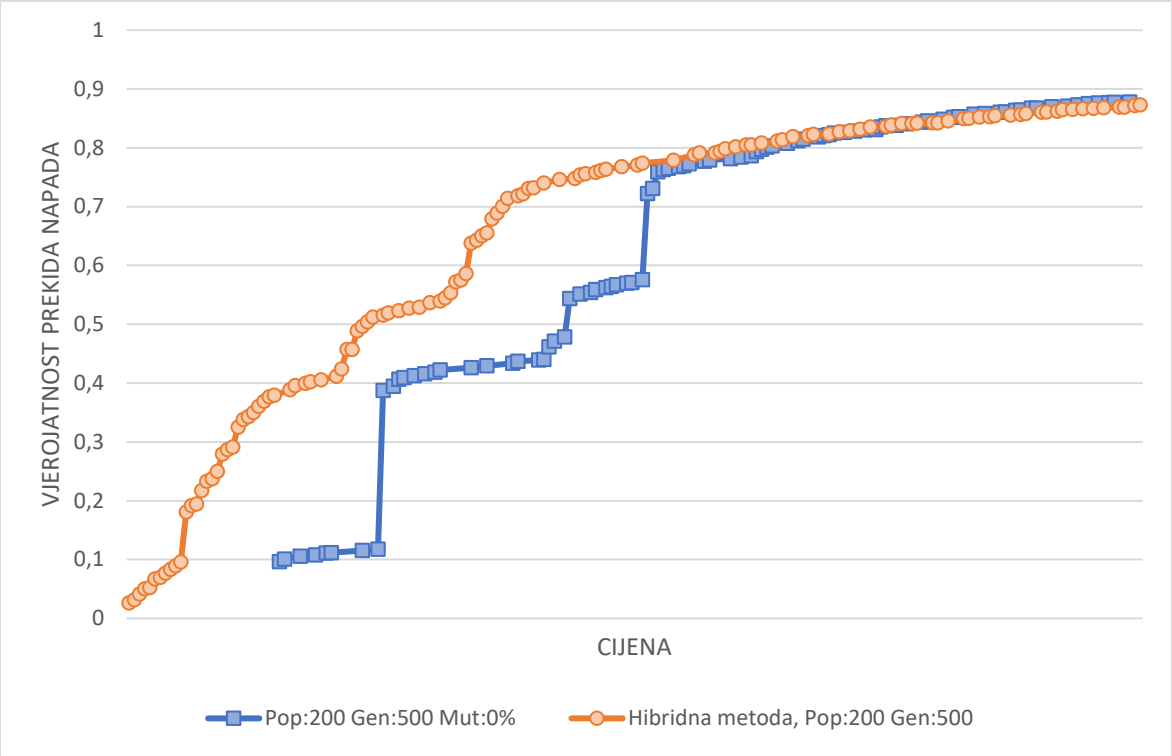
Slika 5.18 Doprinos hibridne metode kvaliteti rješenja – 400 jedinki i 100 generacija

Doprinos kvaliteti rješenja za konfiguraciju s 200 jedinki i 200 generacija prikazuje slika 5.19. Hibridna metoda i ovdje proizvodi skup rješenja pravilno distribuiran po cijeloj Pareto fronti.

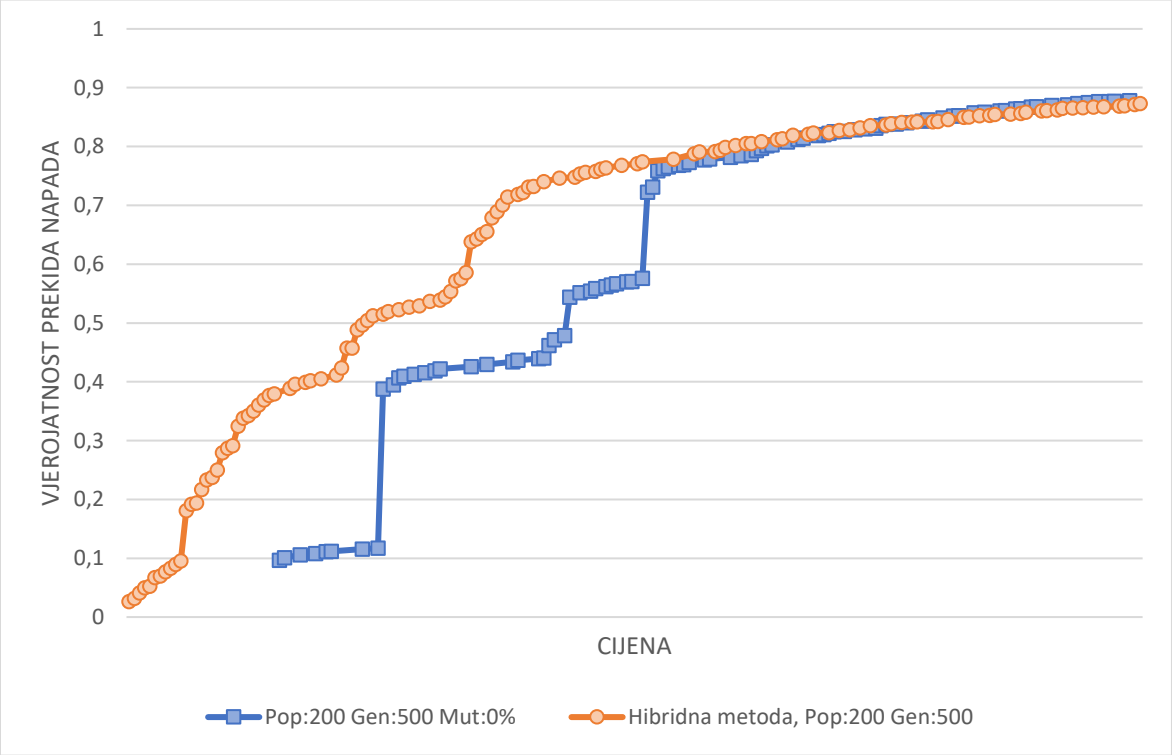


Slika 5.19 Doprinos hibridne metode kvaliteti rješenja, objekt A, 200 jedinki i 200 generacija

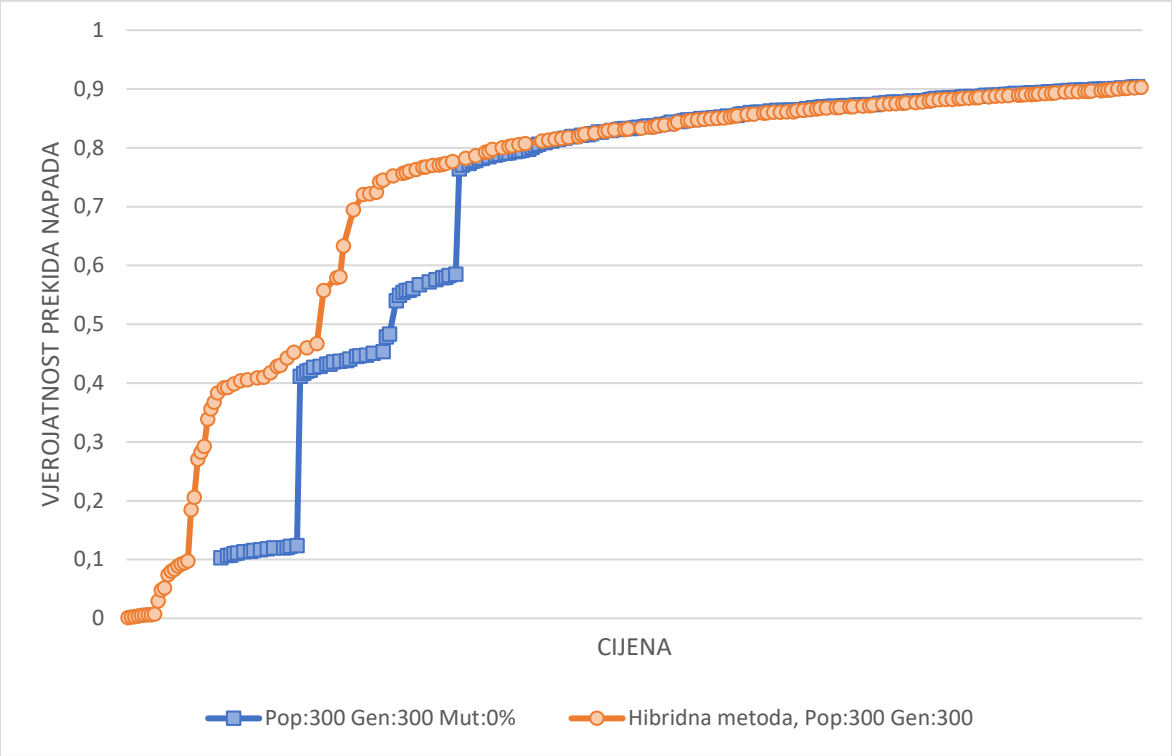
Eksperiment inicijalizacije populacije korištenjem domenskog znanja proveden je i nad objektom B. Usporedbu rezultata sa standardnim GA, za konfiguraciju s 500 jedinki i 200 generacija, prikazuje slika 5.20, za konfiguraciju s 200 jedinki i 500 generacija rezultate prikazuje slika 5.21 i rezultate za konfiguraciju 300 jedinki i 300 generacija prikazuje slika 5.22. U svim eksperimentalnim konfiguracijama vidljivo je da metoda postiže kvalitetnija rješenja na gotovo cijelom području Pareto fronte i da su rješenja pravilnije raspoređena.



Slika 5.20 Doprinos hibridne metode kvaliteti rješenja, objekt B, 500 jedinki i 200 generacija

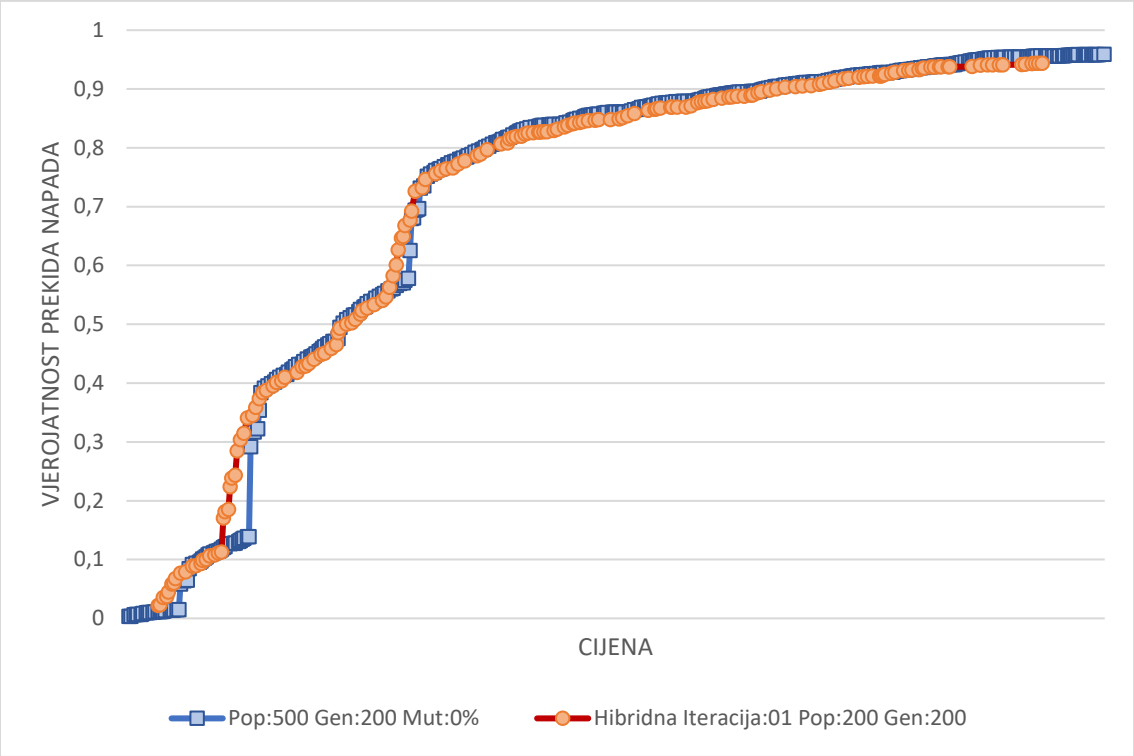


Slika 5.21 Doprinos hibridne metode kvaliteti rješenja, objekt B, 200 jedinki i 500 generacija



Slika 5.22 Doprinos hibridne metode kvaliteti rješenja, objekt B, 300 jedinki i 300 generacija

Hibridna metoda s manjim brojem evaluiranih rješenja postiže rezultate koji svojom kvalitetom odgovaraju standardnim GA s većim brojem evaluacija. Slika 5.23 prikazuje usporedbu rješenja za GA s 200 jedinki i 200 generacija s primijenjenom hibridnom metodom u odnosu na standardni GA s 500 jedinki i 200 generacija. Eksperiment pokazuje da hibridna metoda može nadoknaditi manji broj jedinki kod većeg broja generacija.



Slika 5.23 Doprinos hibridne metode kvaliteti rješenja, objekt A, različita konfiguracija

6. ZAKLJUČAK

Automatizirana provedba optimiranja pozicije i vrste zaštitnih elemenata nepotrebno je zapostavljena u projektiranju sustava tehničke zaštite. Postojeća provedba projektiranja sustava tehničke zaštite, koja se u većini slučajeva provodi isključivo subjektivnom metodom projektanta, bez numeričke analize, ima velike nedostatke. Pri subjektivnoj metodi pozicioniranja i odabira zaštitnih elemenata postoji velika vjerojatnost da projektirani sustav ne sagledava sve mogućnosti kretanja i tehnike savladavanja zaštite dostupne napadaču niti osigurava da je implementiran sustav cjenovno optimalan za postignutu razinu zaštite.

Za razliku od subjektivne metode, numerička analiza projektiranog sustava zaštite omogućuje ocjenjivanje kvalitete sustava u obliku vjerojatnosti da će napadač biti dostignut prije dolaska do cilja napada. Većina metodologija koje se zasnivaju na numeričkoj analizi sustava zaštite, očekuje ručno definiranje štićenog objekta pa se najčešće izvode grubom podjelom objekta na nekoliko razina zaštite. Doprinos ovog rada je model koji omogućuje detaljno opisivanje štićenog objekta, do razine prostorije, zidova, vrata i slično, a omogućuje provođenje numeričke analize ugroženosti i u sljedećem koraku optimiranje. Danas se za projektiranje građevina koriste aplikacije koje ih opisuju standardiziranim modelima, poput IFC ili CityGML. Pri tome građevine nisu opisane samo geometrijski, već je standardima definiran i materijal od kojeg su pojedini elementi izvedeni i slično. U predloženi model štićenog objekta preuzimaju se podaci o građevini i nad njima se provode geometrijske transformacije u model kojim se opisuje moguće kretanja napadača. U procesu transformacije uzimaju se u obzir i opremljenost napadača te sposobnost savladavanja različitih vrsti prepreka i učinkovitost zaštitnih elemenata. Time se automatizira izrada modela za provedbu numeričke analize ugroženosti štićenog objekta.

Sljedeći korak u postizanju automatizirane provedbe optimiranja pozicije i vrste zaštitnih elemenata je uspostava modela nad kojim se može provesti optimiranje. Optimiranjem se želi postići maksimalna vjerojatnost presretanja napadača uz minimalnu cijenu sustava.

Istražen je način opisivanja problema jednokriterijskim i višekriterijskim funkcijama. Zbog više nedostataka jednokriterijskog optimiranja ovakvog problema, pri čemu je osnovni da ograničuje projektanta u izboru i onemogućuje donošenje informirane odluke, daljnji proces optimiranja vršen je višekriterijskom funkcijom. Predstavljena je metoda koja iz modela mogućeg kretanja napadača i dostupnih elemenata zaštite stvara model sustava zaštite nad kojim se može provesti optimiranje.

Automatiziranim prijenosom svih elemenata građevine, na osnovu kojih se izgrađuje model mogućeg kretanja napadača, stvara se velik početni prostor pretraživanja. Uvedena je metoda kojom se prostor pretraživanja optimizira. Metoda prepoznaje sve elemente prostora pretraživanja koji ne mogu biti izabrani u optimalno rješenje te ih izbacuje iz prostora pretraživanja, čime ga značajno smanjuje.

Cilj optimiranja je pronaći skup gotovo optimalnih rješenja, raspoređenih na Pareto fronti. Time se projektantu omogućuje odabir rješenja koje postiže traženu minimalnu vjerojatnost prekida napada ili maksimalnu cijenu sustava. Zbog složenosti problema, odnosno velikog broja mogućih rješenja, za provođenje optimiranja odabrani su evolucijski algoritmi. Optimiranje dinamičkim programiranjem nije primjenjivo za ovaj problem jer se za svako predloženo rješenje prostor pretraživanja mijenja. Jedinka predstavlja odabrane zaštitne elemente i njihovu lokaciju u prostoru. Za razliku od većine postojećih metoda optimiranja sustava zaštite, uvedena je metoda koja uzima u obzir sve pozicije s kojih napadač može započeti napad. Jednostavno se može primijeniti i za istovremeno optimiranje više ciljeva napada. Postojeće rješenje optimizira sustav pretpostavljajući da napad izvršava jedan napadač. Za optimiranje zaštite od više istovremenih napadača, potrebno je nadograditi metodu, uzimajući u obzir mogućnost njihove suradnje tijekom napada.

U proces evaluacije rješenja uveden je jedan od doprinosa ovog rada, algoritam D-EX2. Izrađen je postupak ubrzanog izračuna funkcije ugroženosti i cijene sustava na temelju rezultata izračuna u prethodnim iteracijama. Ubrzanje izračuna postignuto je donošenjem odluke treba li izvršiti novi izračun za predloženo rješenje ili se na osnovu ranije stečenog znanja odabire i vraća kao rezultat spremljenu vrijednost evaluacije. Zbog načina rada, ovaj se algoritam može primijeniti i u drugim domenama, kod kojih se prostor pretraživanja mijenja ovisno o predloženom rješenju i pretražuje optimalan put kretanja grafom. Algoritam je posebno koristan u periodu eksperimentiranja evolucijskim algoritmima.

Pretraživanjem rješenja povećava se znanje o prostoru pretraživanja pa se vremenom broj izračuna tijekom evaluacije sve više smanjuje.

Rezultati evolucijskih algoritama ovise o inicijalnoj populaciji koja se uobičajeno stvara slučajnim jedinkama. Dodatni doprinos rada je uvođenje hibridne metode optimiranja kojom se početne jedinke stvaraju na osnovu poznavanja domene. Najveća učinkovitost detekcije napadača je na samom početku napada, a učinkovitost zadržavanja bliže cilju. Sukladno tome, algoritam u početnim jedinkama raspoređuje zaštitne elemente s detekcijom na pozicije s kojih napadač može započeti napad, a blokirajuće elemente na pozicijama u blizini cilja napada. Dodatno se zaštitnim elementima opremaju pozicije koje su članovi više kritičnih putova jer se time jednim zaštitnim elementom utječe na više kritičnih putova.

Sve uvedene metode optimiranja eksperimentalno su provjerene na modelima realnih građevina. Eksperimentalno je pokazano da se prostor pretraživanja značajno smanjuje primjenom metode koja izbacuje elemente koji se ne mogu pojaviti u optimalnom rješenju. Na eksperimentalnim primjerima broj čvorova kojima se napadač može kretati smanjen je do 65%. Testirana je učinkovitost algoritma D-EX2 različitim konfiguracijama genetskog algoritma i provođenjem optimiranja nad različitim konfiguracijama grafa kretanja. Eksperimentalno je pokazano da algoritam postiže dobre rezultate za realne građevine, izbjegavajući do 75% izračuna. Veća učinkovitost izbjegavanja izračuna postiže se pri većem broju generacija genetskog algoritma, a mutacija jedinki utječe na smanjenje učinkovitosti. Algoritam postiže veću učinkovitost na objektima s više mogućih pozicija za početak napada jer se tijekom evaluacije povećava vjerojatnost izbjegavanja izračuna za sve početne pozicije. Algoritam nije iskoristiv za teoretske grafove kretanja, oblika poput križaljke, u kojima većina čvorova ima povezanost sa četiri ili osam susjednih čvorova. Algoritam se može nadograditi inicijalnom provjerom povezanosti grafa kako bi se izbjeglo provođenje nad nepovoljnim oblicima grafova ili dodati internu provjeru učinkovitosti koja zaustavlja daljnje provođenje D-EX2 optimiranja ukoliko utvrdi neučinkovitost. Hibridna metoda optimiranja, kojom se predlažu početna rješenja, pri svakom izvršavanju pokazuje bolje rezultate od standardnog provođenja genetskog algoritma, neovisno o konfiguraciji GA, a dodatna je prednost što su rješenja pravilno disperzirana po Pareto fronti.

Prema mišljenju autora, glavni znanstveni doprinosi ove disertacije sadržani su u sljedećem:

1. Definiran je model kojim se opisuju štićeni objekt, napadač, tjelesna zaštita i zaštitni elementi i predlaže metoda kojom se provodi optimiranje pozicije i vrste zaštitnih elemenata.
2. Izrađen je postupak ubrzanog izračuna funkcije ugroženosti i cijene sustava na temelju rezultata izračuna u prethodnim iteracijama i eksperimentalno je dokazana učinkovitost postupka.
3. Predstavljena je hibridna metoda optimiranja pozicije i vrste zaštitnih elemenata i eksperimentalno pokazano da smanjuje prostor pretraživanja i proizvodi kvalitetnije rezultate u odnosu na standardne postupke optimiranja.

POPIS LITERATURE

[1] Garcia, M. L., "The design and evaluation of physical protection systems", 2nd ed. Elsevier/Butterworth-Heinemann, Amsterdam, 2008.

[2] Landoll, D. J., "The security risk assessment handbook: a complete guide for performing security risk assessments". Auerbach Publications, Boca Raton, FL, 2006.

[3] Čakija, D., Ban, Ž., Golub, M., Čakija, D., "Optimizing physical protection system using domain experienced exploration method", Automatika, vol. 61, no. 2, April 2020, str. 207–218, doi: 10.1080/00051144.2019.1698192.

[4] Dijkstra, E. W., "A note on two problems in connexion with graphs", Numerische Mathematik, vol. 1, no. 1, December 1959, str. 269–271, doi: 10.1007/BF01386390.

[5] Edelkamp, S., Schrödl, S., "Heuristic search: theory and applications". Morgan Kaufmann, Amsterdam ; Boston, 2012.

[6] Hart, P., Nilsson, N., Bertram, R., "A formal basis for the heuristic determination of minimum cost paths", IEEE Transactions on Systems Science and Cybernetics, vol. SSC-4, 1968, str. 101–107.

[7] Russell, S. J., Norvig, P., "Artificial intelligence: a modern approach", Third edition, Global edition. Pearson, Boston Columbus Indianapolis, 2016.

[8] Čakija, D., "Numerička analiza ugroženosti štićenog objekta od napada", magistarski rad, FER, Zagreb, 2011.

[9] Rios, L., Chaimowicz, L., "A survey and classification of A* based best-first heuristic search algorithms", Lecture Notes in Artificial Intelligence, SBIA 2010, São Bernardo do Campo, Brazil, vol. 6404, 2010, str. 253–262.

[10] Pijls, W., Post, H., "Yet another bidirectional algorithm for shortest paths", Econometric Institute Research Papers EI 2009-10, Erasmus University Rotterdam, Erasmus School of Economics (ESE), Econometric Institute, 2009, doi: 10.1.1.507.3985.

[11] Rios, L. H. O., Chaimowicz, L., "PNBA*: A Parallel Bidirectional Heuristic Search Algorithm", Encontro Nacional de Inteligência Artificial (ENIA 2011), Natal. Anais do XXXI Congresso da Sociedade Brasileira de Computação, 2011.

[12] Barker, J. K., Korf, R. E., "Limitations of Front-to-End Bidirectional Heuristic Search", AAAI'15: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015, str. 1086–1092.

[13] Holte, R. C., Felner, A., Sharon, G., Sturtevant, N. R., Chen, J., "MM: A bidirectional search algorithm that is guaranteed to meet in the middle", Artificial Intelligence, vol. 252, November 2017, str. 232–266, doi: 10.1016/j.artint.2017.05.004.

[14] Čakija, D., Ban, Ž., "Modeling facility protection for numerical vulnerability assessment", in 34th International Convention MIPRO, Rijeka, 2011, str. 829–832.

[15] Garcia, M. L., "Vulnerability assessment of physical protection systems". Elsevier Butterworth-Heinemann, Amsterdam, 2006.

[16] Eastman, C., Fisher, D., Lafue, G., Lividini, J., Stoker, D., Yessios, C., "An Outline of the Building Description System", Institute of Physical Planning, Research report 50, Carnegie-Mellon University, September 1974.

[17] Eastman, C. M., Teicholz, P. M., Sacks, R., Lee, G., "BIM handbook: a guide to building information modeling for owners, managers, designers, engineers and contractors", Third edition. Wiley, Hoboken, New Jersey, 2018.

[18] Tang, L., Li, L., Ying, S., Lei, Y., "A Full Level-of-Detail Specification for 3D Building Models Combining Indoor and Outdoor Scenes", IJGI, vol. 7, no. 11, October 2018, doi: 10.3390/ijgi7110419.

[19] de Laat, R., van Berlo, L., "Integration of BIM and GIS: The Development of the CityGML GeoBIM Extension", in Advances in 3D Geo-Information Sciences, T. H. Kolbe, G. König, and C. Nagel, Eds. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, str. 211–225.

[20] Kang, H.-K., Li, K.-J., "A Standard Indoor Spatial Data Model—OGC IndoorGML and Implementation Approaches", IJGI, vol. 6, no. 4, April 2017, doi: 10.3390/ijgi6040116.

[21] Kim, J.-S., Yoo, S.-J., Li, K.-J., "Integrating IndoorGML and CityGML for Indoor Space", Pfoser D., Li KJ. (eds) Web and Wireless Geographical Information Systems. W2GIS 2014. Lecture Notes in Computer Science, vol 8470. Springer, Berlin, Heidelberg, 2014, str. 184–196.

[22] Lin, W. Y., Lin, P. H., "Intelligent generation of indoor topology (i-GIT) for human indoor pathfinding based on IFC models and 3D GIS technology", Automation in Construction, vol. 94, October 2018, str. 340–359, doi: 10.1016/j.autcon.2018.07.016.

[23] Ismail, A., Strug, B., Ślusarczyk, G., "Building Knowledge Extraction from BIM/IFC Data for Analysis in Graph Databases", in Artificial Intelligence and Soft Computing, vol. 10842, L. Rutkowski, R. Scherer, M. Korytkowski, W. Pedrycz, R. Tadeusiewicz, and J. M. Zurada, Eds. Springer International Publishing, Cham, 2018, str. 652–664.

[24] Lee, J.-K., Eastman, C. M., Lee, J., Kannala, M., Jeong, Y.-S., "Computing Walking Distances within Buildings Using the Universal Circulation Network", Environment and Planning B: Planning and Design, vol. 37, no. 4, August 2010, str. 628–645, doi: 10.1068/b35124.

[25] Teo, T.-A., Cho, K.-H., "BIM-oriented indoor network model for indoor and outdoor combined route planning", Advanced Engineering Informatics, vol. 30, no. 3, August 2016, str. 268–282, doi: 10.1016/j.aei.2016.04.007.

[26] Lorenz, B., Ohlbach, H. J., Stoffel, E.-P., "A Hybrid Spatial Model for Representing Indoor Environments", in Web and Wireless Geographical Information Systems, vol. 4295, J. D. Carswell and T. Tezuka, Eds. Springer-Verlag, Berlin, Heidelberg, 2006, str. 102–112.

[27] Jang, S.-S., Kwan, S.-W., Yoo, H.-S., Kim, J.-S., Yoon, W.-K., "Development of a vulnerability assessment code for a physical protection system: systematic analysis of physical protection effectiveness (SAPE)", Nuclear Engineering and Technology, vol. 41, no. 5, June 2009, str. 747–752.

[28] Mekni, M., "Automated Generation of Geometrically-Precise and Semantically-Informed Virtual Geographic Environments Populated with Spatially-Reasoning Agents", doktorski rad, Universite Laval, Quebec, 2010.

[29] Prinz, F. B., Chern, J.-H., "Geometric abstractions using medial axis transformation", Carnegie Mellon University, Pittsburgh, 1988.

[30] Lee, J., "A Spatial Access-Oriented Implementation of a 3-D GIS Topological Data Model for Urban Entities", GeoInformatica, vol. 8, no. 3, September 2004, str. 237–264, doi: 10.1023/B:GEIN.0000034820.93914.d0.

[31] Taneja, S., Akinci, B., Garrett, J. H., Soibelman, L., East, B., "Transforming IFC-Based Building Layout Information into a Geometric Topology Network for Indoor Navigation Assistance", in Computing in Civil Engineering (2011), Miami, Florida, United States, June 2011, str. 315–322, doi: 10.1061/41182(416)39.

[32] Taneja, S., Akinci, B., Garrett, J. H., Soibelman, L., "Algorithms for automated generation of navigation models from building information models to support indoor map-matching", Automation in Construction, vol. 61, January 2016, str. 24–41, doi: 10.1016/j.autcon.2015.09.010.

[33] Mortari, F., Zlatanova, S., Liu, L., Clementini, E., "“Improved Geometric Network Model” (IGNM): a novel approach for deriving Connectivity Graphs for Indoor Navigation", ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. II–4, April 2014, str. 45–51, doi: 10.5194/isprsannals-II-4-45-2014.

[34] Lewandowicz, E., Lisowski, P., "Methodology to generate navigation models in building", Journal of civil engineering and management, vol. 24, no. 8, December 2018, str. 619–629, doi: 10.3846/jcem.2018.6599.

[35] Vintr, Z., Valis, D., Malach, J., "Attack tree-based evaluation of physical protection systems vulnerability", in 2012 IEEE International Carnahan Conference on Security Technology (ICCST), October 2012, str. 59–65, doi: 10.1109/CCST.2012.6393538.

[36] Flammini, F., Gaglione, A., Mazzocca, N., Pragliola, C., "Optimisation of security system design by quantitative risk assessment and genetic algorithms", International

Journal of Risk Assessment and Management, vol. 15, no. 2/3, 2011, str. 205–221, doi: 10.1504/IJRAM.2011.042117.

[37] Flammini, F., Gentile, U., Marrone, S., Nardone, R., Vittorini, V., "A Petri Net Pattern-Oriented Approach for the Design of Physical Protection Systems", in Computer Safety, Reliability, and Security, vol. 8666, Springer, 2014, str. 230–245.

[38] Gargano, M. L., Edelson, Benjamin, Meisinger, "Evolving Efficient Security Systems", May 2003.

[39] Fonseca, C. M., Fleming, P. J., "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization", in 5th International Conference on Genetic Algorithms, Urbana-Champaign, Illinois, USA, 1993, str. 416–423.

[40] Coello, C. A. C., Lamont, G. B., Veldhuizen, D. A. V., "Evolutionary Algorithms for Solving Multi-Objective Problems Second Edition". Springer, Boston, MA, 2007.

[41] Caballero, R., Rey, L., Ruiz, F., Gonzalez, M., "An algorithmic package for the resolution and analysis of convex multiple objective problems", in Multiple criteria decision making, Proceedings of the 12th International Conference, Germany, Fandel, G., Gal, T. (ur.), Springer, 1997, str. 275–284.

[42] Skiena, S. S., "The algorithm design manual", 2nd ed. Springer, London, 2008.

[43] Abraham, A., Jain, L. C., Goldberg, R., Eds., "Evolutionary multiobjective optimization: theoretical advances and applications". Springer, New York, 2005.

[44] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: NSGA-II", IEEE Trans. Evol. Computat., vol. 6, no. 2, April 2002, str. 182–197, doi: 10.1109/4235.996017.

[45] Ball, J. M., "Practical experiments and simulations for nuclear safeguards education", magistarski rad, University of Missouri, Columbia, 2007.

[46] Wagner, S. et al., "Architecture and Design of the HeuristicLab Optimization Environment", in Advanced Methods and Applications in Computational Intelligence, vol. 6, Springer International Publishing, Heidelberg, 2014, str. 197–261.

Popis slika

Popis tablica

Životopis

Dejan Čakija rođen je 4.2.1974. u Zadru. Osnovnu školu i matematičku gimnaziju završava u Koprivnici. Elektrotehnički fakultet Sveučilišta u Zagrebu završava 1997. na smjeru radiokomunikacije i profesionalna elektronika. Nakon završetka fakulteta bio je više puta na stručnim usavršavanjima iz područja kontrole pristupa u Švicarskoj, video nadzora u Njemačkoj i Danskoj te komunikacijskih sustava u Norveškoj.

Po završetku fakulteta započinje profesionalnu karijeru u tvrtki Eccos inženjering d.o.o., u području sustava tehničke zaštite. Radi na projektima razvoja i uvođenja sustava tehničke zaštite. Iskustvo stiče implementacijom značajnih sustava zaštite u Hrvatskoj i inozemstvu u farmaceutskoj, naftnoj, energetskoj, bankarskoj i kartičarskoj industriji. Trenutno radi kao direktor razvoja tvrtke Eccos inženjering d.o.o.

Osim toga, održava predavanje o sustavima tehničke zaštite kao gost-predavač na Visokoj školi za sigurnost u Zagrebu.

Magistarski rad pod naslovom „Numerička analiza ugroženosti štićenog objekta od napada“ obranio je 2011. godine na Fakultetu elektrotehnike i računarstva u Zagrebu.

Tijekom svog profesionalnog rada objavljuje članke vezane uz sustave zaštite u stručnim i znanstvenim časopisima. Redovito prisustvuje stručnim skupovima na području zaštite u Europi i SAD-u.

Služi se engleskim i njemačkim jezikom.

Popis objavljenih radova

Čakija, D., Ban, Ž., "Modeling facility protection for numerical vulnerability assessment", in 34th International Convention MIPRO, Rijeka, 2011, str. 829–832.

Čakija, D., Ban, Ž., Golub, M., Čakija, D., "Optimizing physical protection system using domain experienced exploration method", Automatika, vol. 61, no. 2, April 2020, str. 207–218, doi: 10.1080/00051144.2019.1698192.

Biography

Dejan Čakija was born on February 4th, 1974 in Zadar, Croatia. He completed his primary and secondary education (mathematics gymnasium) in Koprivnica. In 1997 he graduated from the Faculty of Electrical Engineering at the University of Zagreb. He acquired further professional development in access control systems in Switzerland, video security in Germany and Denmark, and communication systems in Norway.

After graduation, Dejan began his professional career at company Eccos inženjering. He has worked on physical security systems project development and implementation and gained deep experience implementing significant security systems in Croatia and abroad servicing pharmaceutical, energy, telecommunications, government and banking sectors. He currently works as chief technology officer at Eccos inženjering.

Dejan is a frequent guest-lecturer on security education at College of Security, Zagreb.

Throughout his professional career Dejan has published papers in the area of security in professional and scientific proceedings. He regularly attends symposiums on security systems in both Europe and USA.

In 2011 Dejan defended his master thesis "Numerical analysis of the object vulnerability to intrusion" at the Faculty of Electrical Engineering and Computing in Zagreb.

He speaks English, German and Croatian.