UNIVERZITET U NOVOM SADU
FAKULTET TEHNIČKIH NAUKA

Jedno rješenje primjene genetskog algoritma za analizu protokola usklađivanja vremena u TTEthernet mrežama

DOKTORSKA DISERTACIJA

Mentor:
Prof. dr Nikola Teslić

Kandidat:
Miladin Sandić

Novi Sad, 2021 godine

Zahvalnica

Dugujem zahvalnost svima koji su pomogli pri izradi ove doktorske disertacije. Zahvaljujem Prof. Dr Nikoli Tesliću i Institutu RT-RK koji su mi omogućili doktorske studije. Veliku zahvalnost dugujem Doc. Dr Bogdanu Pavkoviću na pomoći i sugestijama koje su mi mnogo značile tokom pisanja doktorske disertacije, kao i tokom doktorskih studija. Zahvaljujem se Dr Vilfridu Štajneru (Wilfried Steiner) na pomoći pri razumijevanju mehanizama rada TTEthernet mreža. Doc. Dr Mariji Antić i Prof. Dr Tatjani Pešić-Brđanin takođe dugujem zahvalnost na savjetima pri uređenju disertacije. Kolegi Lazaru Jovanoviću se zahvaljujem na pomoći pri izradi praktičnog dijela disertacije.

Ipak, najveću zahvalnost dugujem svojim roditeljima, Goranu i Danki, na pomoći i podršci tokom svih faza školovanja kako bih ostvario svoje ciljeve.

УНИВЕРЗИТЕТ У НОВОМ САДУ **ОБРАЗАЦ - 5а**
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА[1]

Врста рада:	Докторска дисертација
Име и презиме аутора:	Миладин Сандић
Ментор (титула, име, презиме, звање, институција)	Др Никола Теслић, редовни професор, Факултет техничких наука Универзитета у Новом Саду
Наслов рада:	Једно рјешење примјене генетског алгоритма за анализу протокола усклађивања времена у *TTEthernet* мрежама
Језик публикације (писмо):	Српски (латиница)
Физички опис рада:	Унети број: Страница 132 Поглавља 9 Референци 82 Табела 4 Слика 61 Графикона - Прилога -
Научна област:	Електротехничко и рачунарско инжењерство
Ужа научна област (научна дисциплина)	Рачунарска техника и рачунарске комуникације
Кључне речи / предметна одредница:	*TTEthernet,* усклађивање часовника отпорно на отказе, безбједносно-критични системи, симулација
Резиме на језику рада:	Безбједносно-критични системи попут авиона или аутомобила захтијевају високо-поуздану размјену порука између уређаја у систему, што се постиже примјеном детерминистичких мрежа. Правилно успостављање међусобне усклађености часовника, као и константно одржавање временске усклађености, сврставају се међу најбитније аспекте детерминистичких мрежа међу којима су и *TTEthernet* мреже. Уколико часовници мрежних уређаја нису временски усклађени, детерминистичка размјена порука у мрежи није изводљива. С обзиром да се информације о најкритичнијим функцијама

[1] Аутор докторске дисертације потписао је и приложио следеће Обрасце:
5б – Изјава о ауторству;
5в – Изјава о истоветности штампане и електронске верзије и о личним подацима;
5г – Изјава о коришћењу

Ове изјаве се чувају на факултету у штампаном и електронском облику и не кориче се са тезом.

	система преносе преко детерминистичке класе порука, очигледно је да овакви сервиси неће бити доступни све док се часовници не ускладе. Теза се бави процјеном најгорег случаја времена које је потребно да протекне да би се часовници мрежних уређаја међусобно ускладили, у случају да у мрежи постоји један уређај под отказом. Процјене су вршене помоћу *OMNeT++* симулација уз примјену генетског алгоритма. Симулације показују да се вријеме неопходно да се успостави усклађеност часовника у *TTEthernet* мрежи значајно повећава под утицајем уређаја под отказом, а самим тим се продужава и вријеме недоступности најкритичнијих сервиса мреже. Симулације показују да се за мрежу посматрану у тези, за изабране параметре мреже добија процијењена вриједност медијане једнака 489579μs за најгори случај успостављања временске усклађености у мрежи.
Датум прихватања теме од стране надлежног већа:	
Датум одбране: (Попуњава одговарајућа служба)	
Чланови комисије: (титула, име, презиме, звање, институција)	Председник: Др Мирослав Поповић, редовни професор, Факултет техничких наука Универзитета у Новом Саду Члан: Др Мило Томашевић, Електротехнички факултет Универзитета у Београду Члан: Др Иштван Пап, ванредни професор, Факултет техничких наука Универзитета у Новом Саду Члан: Др Богдан Павковић, доцент, Факултет техничких наука Универзитета у Новом Саду Ментор: Др Никола Теслић, редовни професор, Факултет техничких наука Универзитета у Новом Саду
Напомена:	

UNIVERSITY OF NOVI SAD
FACULTY OF TECHNICAL SCIENCES

KEY WORD DOCUMENTATION[2]

Document type:	Doctoral disertation
Author:	Miladin Sandić
Supervisor (title, first name, last name, position, institution)	PhD Nikola Teslić, full professor, Faculty of Technical Sciences, University of Novi Sad
Thesis title:	One solution for TTEthernet synchronization analysis using genetic algorithm
Language of text (script):	Serbian
Physical description:	Number of: Pages 132 Chapters 9 References 82 Tables 4 Illustrations 61 Graphs - Appendices -
Scientific field:	Electrical and Computer Engineering
Scientific subfield (scientific discipline):	Computer Engineering, Engineering of Computer Based Systems
Subject, Key words:	TTEthernet, fault-tolerant synchronization, simulation assesment, safety-critical systems
Abstract in English language:	Safety-critical systems like airplanes and cars demand high-reliable communication between components within the system, which is achieved by using deterministic networks. Proper establishing and maintenance of synchronization of device clocks in the network components represents one of crucial aspects in deterministic networks where belong *TTEthernet* as well. If device clocks are not synchronized, deterministic communication is not feasible. Keeping in mind that most critical information has been exchanged between the network components using deterministic traffic class, it is obvious that such services will not be available until the clocks in the network are synchronized.

[2] The author of doctoral dissertation has signed the following Statements:
5б – Statement on the authority;
5в – Statement that the printed and e-version of doctoral dissertation are identical and about personal data;
5г – Statement on copyright licenses.

The paper and e-versions of Statements are held at the faculty and are not included into the printed thesis.

	The thesis deals with estimation of worst-case startup time for observed *TTEthernet* network, in case that one device in the network is under failure. The estimation is performed by OMNeT++ simulations and using genetic algorithm. The simulations show that startup time of the network is extended significantly under impact of faulty component. Also, unavailability of most critical services in the network is extended for the same time. For the network simulated in this thesis, estimated median value equals 489579 μs for worst-case startup time.
Accepted on Scientific Board on:	
Defended: (Filled by the faculty service)	
Thesis Defend Board: (title, first name, last name, position, institution)	President: PhD Miroslav Popović, full professor, Faculty of Technical Sciences, University of Novi Sad Member: PhD Milo Tomašević, full professor, School of Electrical Engineering, University of Belgrade Member: PhD Ištvan Pap, associate professor, Faculty of Technical Sciences, University of Novi Sad Member: PhD Bogdan Pavković, assistant professor, Faculty of Technical Sciences, University of Novi Sad Mentor: PhD Nikola Teslić, full professor, Faculty of Technical Sciences, University of Novi Sad
Note:	

Sažetak

Bezbjednosno-kritični sistemi poput aviona ili automobila zahtijevaju visoko-pouzdanu razmjenu poruka između uređaja u sistemu, što se postiže primjenom determinističkih mreža. Pravilno uspostavljanje međusobne usklađenosti časovnika, kao i konstantno održavanje vremenske usklađenosti, svrstavaju se među najbitnije aspekte determinističkih mreža među kojima su i *TTEthernet* mreže. Ukoliko časovnici mrežnih uređaja nisu vremenski usklađeni, deterministička razmjena poruka u mreži nije izvodljiva. S obzirom da se informacije o najkritičnijim funkcijama sistema prenose preko determinističke klase poruka, očigledno je da ovakvi servisi neće biti dostupni sve dok se časovnici ne usklade.

Teza se bavi procjenom najgoreg slučaja vremena koje je potrebno da protekne da bi se časovnici mrežnih uređaja međusobno uskladili, u slučaju da u mreži postoji jedan uređaj pod otkazom. Procjene su vršene pomoću *OMNeT++* simulacija uz primjenu genetskog algoritma. Simulacije pokazuju da se vrijeme neophodno da se uspostavi usklađenost časovnika *TTEthernet* mreži značajno povećava pod uticajem uređaja pod otkazom, a samim tim se produžava i vrijeme nedostupnosti najkritičnijih servisa mreže. Simulacije pokazuju da se za mrežu posmatranu u tezi, za izabrane parametre mreže dobija procijenjena vrijednost medijane jednaka 489579 μs za najgori slučaj uspostavljanja vremenske usklađenosti u mreži.

Ključne riječi: *TTEthernet*, usklađivanje časovnika otporno na otkaze, bezbjednosno-kritični sistemi, simulacija

Abstract

Safety-critical systems like airplanes and cars demand high-reliable communication between components within the system, which is achieved by using deterministic networks. Proper establishing and maintenance of synchronization of device clocks in the network components represents one of crucial aspects in deterministic networks where belong *TTEthernet* as well. If device clocks are not synchronized, deterministic communication is not feasible. Keeping in mind that most critical information has been exchanged between the network components using deterministic traffic class, it is obvious that such services will not be available until the clocks in the network are synchronized.

The thesis deals with estimation of worst-case startup time for observed *TTEthernet* network, in case that one device in the network is under failure. The estimation is performed by OMNeT++ simulations and using genetic algorithm. The simulations show that startup time of the network is extended significantly under impact of faulty component. Also unavailability of most critical services in the network is extended for the same time. For the network simulated in this thesis, estimated median value equals 489579 μs for worst-case startup time.

Key words: *TTEthernet*, fault-tolerant synchronization, simulation assesment, safety-critical systems

Sadržaj

Lista slika

Lista tabela

Lista skraćenica

TTEthernet – Time-Triggered Ethernet

TTP – Time-Triggered Protocol

TTCAN – Time-Triggered Controller Area Network

LIN – Local Interconnect Network

BRAIN - Braided Ring Availability Integrity Network

ADAS – Advanced Driver Assistance System

AVB – Audio Video Bridging

NIC – Network Interface Card

CPU – Central Processing Unit

FIT – Failure In Time

MTTF – Mean Time To Failure

MTTR – Meat Time To Repair

MTBF – Mean Time Between Failures

IoT – Internet of Things

FCR – Fault Containment Region

FT – Fault Tolerant

TAI – Temps Atomique International

UTC – Universal Time Coordinated

GPS – Global Positioning System

GLONASS – GLObal Navigation Sattelite System

CD – Clique Detection

TT – Time Triggered

RC – Rate Constrained

BE – Best Effort

ET – Event Triggered

ES – End System

LCM – Least Common Multiple

BAG – Bandwidth Allocation Gap

ECU – Electronic Control Unit

CS – Coldstart

CA – Coldstart Acknowledge

IN – Integration

SM – Synchronization Master

SC – Synchronization Client

CM – Compression Master

CSO – Coldstart Offset

CAO – Coldstart Acknowledge Offset

RTD – Round Trip Delay

DES – Discrete Event Simulator

LAN – Local Area Network

WAN – Wide Area Network

SAN – Storage Area Network

NED – Network Description

CPU – Central Processing Unit

RAM – Random Access Memory

DOTTS - Design Optimization of TTEthernet-based Systems

1 Uvod

U bezbjednosno-kritičnim sistemima kao što je npr. u avionskoj i automobilskoj industriji, veoma je važno obezbjediti visoko-pouzdanu komunikaciju između mrežnih komponenata. Ovdje je neophodno da postoje garancije u pogledu isporuke poslanih paketa, ograničenog kašnjenja paketa, kao i sposobnosti rada u slučaju otkaza neke od komponenata. Iz tog razloga, za ovakve primjene uglavnom se koriste deterministične ili mreže sa više prioriteta (eng. *mixed-critical*) kao što su npr. *TTEthernet*, SAFEbus, TTP/A, TTP/C, TTCAN, *FlexRay*, LIN, BRAIN, itd [1-3].

Mreže sa više prioriteta mogu se podesiti za istovremeni prenos determinističke (eng. *deterministic*), kao i naletne/nedeterminističke (eng. *best-effort*) klase saobraćaja. Ovakve konfiguracije su neophodne u sistemima kao što su npr. moderni automobili. Kritični saobraćaj (rad motora, kočioni sistem, ADAS, itd.) se prenosi determinističkim putem, a podaci koji se tako prenose ne smiju da podlegnu gubitku. Kod naletnog saobraćaja ne postoje nikakve garancije u pogledu kašnjenja i isporuke paketa. Podaci koji smiju da se prenose u okviru naletnog saobraćaja su npr. poruke multimedijalnog sistema u automobilu (eng. *In-Car Entertainment*), gdje spomenuti parametri nisu od visokog značaja. Postoje i podaci koji treba da se prenose klasama saobraćaja koje su po prioritetu između dvije spomenute, gdje je dopušten gubitak nekih paketa, ali kašnjenje mora biti ograničeno. Tu se npr. svrstavaju sistemi koji koriste više kamera montiranih na automobil. Ukoliko bi se koristila naletna klasa saobraćaja moglo bi doći do pojave neusklađene slike primljene sa više kamera, zbog kašnjenja poruka koje nije ograničeno. Za ovakve i slične primjene, dosta radova istražuje kombinaciju više komunikacionih protokola kao što su *TTEthernet* i AVB protokoli [3-6].

Pouzdano i kvalitetno usklađivanje vremena (eng. *synchronization*) je preduslov za postojanje determinističkog saobraćaja između uređaja u mreži. U fokusu ovog rada je analiza protokola usklađivanja vremena u *TTEthernet* mrežama. Analiza je vršena pomoću simulatora razvijenog u okruženju za simulacije i projektovanje mrežnih simulatora *OMNeT++* [7]. Posmatrana je redundantna *TTEthernet* topologija sa stepenom redundanse 2, gdje dolazi do otkaza jednog komutatora (eng. *switch*) ili jednog krajnjeg uređaja (eng. *End System*). Pokazano je da neispravna komponenta utiče na produženje vremena potrebnog za uspostavljanje usklađenih vremena na ostalim komponentama u mreži, ali su te komponente i

dalje sposobne da se usklađuju sa ostatkom mreže. Takođe, predložen je metod za procjenu najgoreg slučaja za uspostavljanje usklađenih vremena u ostatku mreže, u uslovima kada je došlo do otkaza krajnjeg uređaja koji je počeo nekontrolisano da šalje poruke. Predloženi metod koristi genetski algoritam, gdje se kroz mnogobrojne iteracije dolazi do željenog rezultata [8].

Motivacija i doprinos doktorske disertacije izloženi su u poglavlju broj 2, gdje je takođe dat i pregled literature. Naredno poglavlje opisuje principe determinističke komunikacije, gdje je definisano stanje sistema, topologije koje se koriste u determinističkim mrežama, kao i bitne osobine determinističkih mreža uopšte. Definisane su greške i otkazi u determinističkim mrežama, tipovi otkaza, te moguća ponašanja komponente pod otkazom. Takođe, navedeni su i tipovi saobraćaja koji mogu da koegzistiraju u mrežama sa više prioriteta, a pored toga su objašnjena statička i dinamička kašnjenja u takvim mrežama.

Četvrto poglavlje se bavi procesima međusobnog usklađivanja časovnika uređaja u determinističkim mrežama. Opisan je koncept digitalnih časovnika, kao i osobine unutrašnjeg i spoljašnjeg načina usklađivanja časovnika. Ukratko su opisane neke od poznatih konvergentnih funkcija koje se koriste u proceduri unutrašnjeg usklađivanja vremena časovnika. Detaljno su opisane faze procedure vremenskog usklađivanja časovnika: pokretanje, ponovno pokretanje, i normalni režim rada.

Princip rada *TTEthernet* mreže dat je u petom poglavlju. Ovdje su opisane klase saobraćaja koje se koriste u *TTEthernet* mrežama namijenjene za prenos korisnih poruka, kao i pomoćna klasa saobraćaja namijenjena za svrhe usklađivanja časovnika, koja je u fokusu ove disertacije. Opisani su mehanizmi integracije različitih klasa saobraćaja u *TTEthernet* mrežama: prekidanje, blokiranje, miješanje. Data je procedura vremenskog usklađivanja uređaja u *TTEthernet* mrežama zajedno sa dijagramom mašine stanja za uređaj koji inicira i održava sinhronizaciju (vremensku usklađenost) u mreži. Objašnjen je koncept transparentnog časovnika na primjeru, kao i način rukovanja greškama kod *TTEthernet* mreža.

Osnovni principi genetskog algoritma koji su korišteni u simulacijama, objašnjeni su u šestom poglavlju. Opisana je procedura genetskog algoritma, kao što je data i terminologija neophodna za razumijevanje simulacija u ovom radu.

Naredno sedmo poglavlje bavi se razvojem simulatora u alatu *OMNeT++* i rezultatima simulacija za slučaj kada je komutator pod greškom, kao i za slučaj kada je krajnji uređaj pod greškom, gdje je primijenjen genetski algoritam za procjenu najgoreg slučaja za uspostavljenje vremenske usklađenosti u posmatranoj *TTEthernet* mreži. Poglavlje 8 daje praktične rezultate mjerenja vršenih na realnoj *TTEthernet* mreži iste topologije kakva je posmatrana i u disertaciji. Disertacija se zaključuje poglavljem broj 9, gdje su navedeni i mogući pravci daljeg istraživanja.

2 Motivacija i doprinos doktorske disertacije

Što se tiče generalno *TTEthernet* mreža, postoji značajan broj studija i radova na tu temu. Poledna (*Stefan Poledna*) je naveo osnovne koncepte korišćenja determinističke komunikacije u bezbjednosno-kritičnim sistemima, te prednosti i nedostatke determinističkog pristupa [9]. Tang (*Xuekian Tang*) i saradnici [10] su opisali proceduru za postupak usklađivanja časovnika u distribuiranim *TTEthernet* mrežama, te su analizirali najgori slučaj preciznosti za datu distribuiranu *TTEthernet* mrežu. Primjenu drugačijih vidova usklađivanja časovnika u *TTEthernet* mreži, konkretno uvođenje IEEE 1588v2 standarda za usklađivanje časovnika umjesto SAE AS6802 standarda, razmatrali su Ćao (*Qi Zhao*) i saradnici [11], gdje su pokazali da se primjenom IEEE 1588v2 standarda postižu bolji rezultati kada se zahtijeva tačnost reda 1 μs. Prijedlog unaprijeđenja *TTEthernet* mreže u pogledu sigurnosti prenosa podataka dali su Ćao (*Rui Zhao*) i saradnici [12]. Pojedini radovi daju i prijedloge optimizacije slanja TT okvira u svrhe efikasnijeg slanja okvira nižeg prioriteta, kao što su BE okviri [13-14]. Neke od aspekata primjene *TTEthernet* mreža u IoT su objasnili su Štajner (*Wilfried Steiner*) [15] i Serna-Oliver (*Ramon Serna-Oliver*) [16]. Peon (*Pablo Gutierrez Peon*) je raspravljao o izvodljivosti realizacije bežičnog *TTEthernet* pristupa [80]. Koncept i simulacija integracije IEEE 802.1 AVB i *TTEthernet* mreža obradio je Majer (*Philipp Meyer*) [4]. Poređenje *TTEthernet* i *TSN* standarda obradili su Ćao (*Lin Zhao*) i saradnici [17]. Sintezom rasporeda slanja okvira i mogućim optimizacijama bavili su se Štajner [18], Kraćunas (*Silviu Craciunas*) [19], kao i Tames-Selisin (*Domitian Tamas-Selicean*) [20].

S obzirom da se *TTEthernet* mreža svrstava u grupu determinističkih mreža, jedan od njenih najbitnijih aspekata predstavlja precizno definisan raspored slanja TT (eng. *Time-Triggered*) poruka. Problem izrade rasporeda slanja poruka u determinističkim mrežama je NP-kompletan problem [3], [21]. Na primjer, NP-kompletan problem je pronalazak optimalnog rasporeda slanja poruka (raspored minimalne dužine). Optimalni raspored zadataka među procesorima u sistemima sa ograničenim brojem procesora gdje cijene komunikacije (eng. *communication cost*) nisu razmatrane je NP-kompletan problem. NP-kompletan problem je takođe i pronalazak optimalnog rasporeda zadataka u sistemima sa neograničenim brojem procesora, gdje se cijene komunikacije ne uzimaju u obzir [3]. Mnogi radovi su objavljeni na temu pronalaska optimalnog rasporeda slanja poruka u *TTEthernet* mrežama [20], [22-24]. Cilj ove teze je pronalazak najgoreg slučaja za uspostavljanje

vremenske usklađenosti u datoj *TTEthernet* mreži (najduže trajanje faze pokretanja mreže), pomoću *OMNeT++* simulacija. Ovaj najgori slučaj je uzrokovan PCF (eng. *Protocol Control Frames*) okvirima emitovanim od strane krajnjeg uređaja (eng. *End System*) pod otkazom. Drugim riječima, tražen je raspored slanja PCF okvira emitovanih od strane krajnjeg uređaja pod otkazom, koji uzrokuje najduže trajanje faze pokretanja *TTEthernet* mreže. Samim tim zaključujemo da je problem posmatran u tezi takođe NP-kompletan problem.

U naučnoj zajednici ne postoji veliki broj radova koji se bave vremenom pokretanja *TTEthernet* mreže. Što se tiče trenutnog stanja u industriji, ispitivanje vremena pokretanja mreže za razne topologije i vrijednosti parametara vrše se iscrpnim analizama, formalnom verifikacijom, kao i ispitivanjima prototipa. Ovaj pristupi su prilično komplikovani, vremenski zahtjevni i skupi. Pristup koji koristi simulacije je dosta jednostavniji i brži, jer ne zahtijeva značajno poznavanje detalja rada *TTEthernet* mreža, kao što ne zahtijeva ni obezbjeđivanje neophodnog hardvera. Bija (*Razvan-Florin Bija*) je na realnom *TTEthernet* hardveru vršio analize vremena pokretanja mreže u slučajevima kada postoji komponenta pod otkazom. Posmatrana je ista topologija mreže kao i u tezi. Međutim, obrađeni pristup u njegovom radu nije efikasan za procjenu najgoreg mogućeg vremena pokretanja mreže, pošto se PCF okviri emituju nasumično sa komponente pod otkazom, bez strategije. U zaključku posmatranog rada spominje se da bi pristup mogao biti poboljšan korištenjem genetskog algoritma [81].

Onučekva (*Daniel Onwuchekva*) i Obermajser [25] su se takođe bavili ponašanjem *TTEthernet* mreže u slučaju postojanja komponente pod otkazom, ali ne u domenu usklađivanja časovnika, nego u domenu osnovnih klasa saobraćaja. Teza se bavi problemom usklađivanja časovnika što može imati i veći značaj, jer bez pravilne usklađenosti časovnika TT komunikacija nije uopšte izvodljiva. Scenario koji posmatraju Onučekva i Obermajser [25] odnosi se na krajnji uređaj koji nasumično emituje različite sekvence TT, RC (eng. *Rate Constrained*) i BE (eng. *Best Effort*) okvira, te se kao rezultati analiza navode povećanja u kašnjenjima okvira, kao i njihova odstupanja od očekivanih vremena prijema na prijemnoj strani (eng. *jitter*). Posmatrana je ista topologija mreže kao i u ovoj tezi (1 komutator u redundantnoj konfiguraciji na koji su povezana 4 krajnja uređaja). Za razliku od postupka predloženog u tezi, gdje su korištene *OMNeT++* simulacije, Onučekva i Obermajser posmatraju stvarnu *TTEthernet* mrežu gdje je komponenta pod otkazom implementirana na FPGA (eng. *Field Programmable Gate Array*) kolu.

Prilikom podešavanja *TTEthernet* mreže, odabirom odgovarajućih parametara mreže jasno je kako će se mreža ponašati u normalnim uslovima rada. Međutim, u slučaju neočekivanih okolnosti, kada neka od komponenata ode u stanje otkaza, teško je procijeniti kako će mreža da se ponaša i koliko će vremena proći dok se ponovo ne uspostavi normalno stanje u ostatku mreže. Jedan od doprinosa teze je procjena uticaja neočekivanih scenarija na rad *TTEthernet* mreže putem simulacija, konkretno uticaj na usklađivanje časovnika u mreži. Ovakve situacije mogu biti uzrokovane npr. udarom groma u vozilo, softverskom/hardverskom greškom, neočekivanim uslovima u okruženju poput temperature, vlažnosti i slično. Iz rezultata simulacija, može se zaključiti koliko vremena će posmatrana mrežna komponenta biti nedostupna što se tiče njenog očekivanog ponašanja. Poznavanjem ovog parametra, može se olakšati odluka o potrebi uvođenja redundantnih komponenata koje bi se uključivale u slučaju otkaza posmatrane komponente. Takođe, projektant ima mogućnost da procijeni ima li potrebe za izborom drugačijih vrijednosti parametara mreže, ukoliko simulacije ne daju zadovoljavajuće rezultate.

Problemom usklađivanja časovnika u *TTEthernet* mreži u slučaju komponente pod otkazom su se bavili Li (*Jing Li*) i saradnici [26], koji su takođe posredstvom simulacija posmatrali jedan specijalan slučaj uticaja krajnjeg uređaja pod otkazom na ostatak mreže - kada on ne šalje CA okvir dok se nalazi u SM_FLOOD stanju. Normalno ponašanje *TTEthernet* mreže podrazumijeva da u SM_FLOOD stanju krajnji uređaj emituje CA okvire. U odnosu na tezu, ovaj rad se bavi jednim specijalnim slučajem komponente pod otkazom, dok teza pokriva širok spektar slučajeva slanja okvira sa uređaja pod otkazom. Takođe, u simulatoru korištenom za potrebe teze, kao ulazni parametri koriste se rezultati komercijalnog alata za podešavanje *TTEthernet* mreža - TTE Tools v5.0 [27]. Samim tim, vremena dobijena u simulatoru vjernije oslikavaju vremena koja bi se dobila i u stvarnoj *TTEthernet* mreži.

Majžide (*Setareh Majidi*) i Obermajser [28] su primjenili genetski algoritam na *TTEthernet* mrežu. Oni su pomoću genetskog algoritma tražili optimalan raspored slanja TT okvira za primjene gdje se koristi kompresija TT poruka. Pored *TTEthernet* mreža, razni autori primjenjuju genetski algoritam i u drugim oblastima. Na primjer, Akvino (*Andrea Aquino*) i saradnici [29] koriste genetski algoritam da bi pronašli najgori slučaj trajanja izvršenja zadatih računarskih programa. Na primjer, ako se posmatra neka funkcija, pomoću datog evolutivnog algoritma, traži se koji ulazni parametri funkcije uzrokuju najduže trajanje izvršavanja te funkcije. U odnosu na tezu, motiv je isti, samo je ostvarena primjena na oblast

softvera. Takođe, u oblasti mreža, genetski algoritam se može primijeniti i za pronalazak optimalnog rutiranja u datoj mreži, čime su se bavili Piri (*Yuliia Pyrih*) i saradnici [30].

Glavni doprinos teze je procjena maksimalnog mogućeg trajanja pokretanja *TTEthernet* mreže u neočekivanim uslovima, što je postignuto korišćenjem genetskog algoritma. U odnosu na istraživanje koje su vršili Majžide i Obermajser [28], može se uspostaviti analogija da je i u tezi pomoću genetskog algoritma takođe tražen jedan vid rasporeda slanja, ali PCF okvira koji rezultuju najdužim trajanjem faze pokretanja *TTEthernet* mreže. Neočekivano ponašanje koje je analizirano ovim pristupom odnosi se na situaciju kada jedna komponenta (krajnji uređaj) počne da nasumično emituje različite sekvence PCF okvira (tzv. *babbling idiot* scenario). Procjena je vršena putem *OMNeT++* simulacija. S obzirom da mrežne komponente posjeduju veliki broj konfiguracionih parametara, veoma je teško i vremenski zahtjevno procijeniti koliko bi trajao najgori slučaj za uspostavljanje vremenske usklađenosti između časovnika u posmatranoj mreži. Metod predložen u tezi rezultuje prvenstveno lakim pronalaskom ovakvog slučaja, jer je dovoljno unijeti parametre mrežnih uređaja u simulator i pustiti simulaciju, nakon čega će rezultat biti najduže moguće trajanje faze pokretanja sa nivoom povjerenja 95%, kao što će biti poznate i sekvence PCF okvira emitovane sa krajnjeg uređaja pod otkazom koje uzrokuju takav rezultat. Drugim riječima, nije potrebno poznavanje detalja principa usklađivanja časovnika u posmatranoj *TTEthernet* mreži. Sam koncept usklađivanja časovnika je dosta složen i zahtijevao bi dosta velike napore ako bi se procjena vršila klasičnim analitičkim pristupom, i vremenski, a i pogledu znanja neophodnih za samo razumijevanje detalja rada *TTEthernet* mreže u neočekivanim okolnostima. Za mrežu koja je posmatrana u ovoj tezi, koja se sastoji od dva komutatora i četiri krajnja uređaja, za izabrane parametre genetskog algoritma bilo je potrebno približno dva dana na mašini koja posjeduje CPU Intel Core i7, 16Gb RAM, da bi se dobio traženi rezultat uz nivo povjerenja 95%. U slučaju da se primijeni ručni pristup, analiza bi mogla potrajati više dana, pa i nedjelja ako bi se radilo o većim mrežama. Takođe, teza daje prijedlog vrijednosti parametara genetskog algoritma koji daje najbolje rezultate u pogledu procjene maksimalnog mogućeg trajanja pokretanja *TTEthernet* mreže u neočekivanim okolnostima.

Na kraju, može se reći da teza ide korak dalje u odnosu dosadašnji rad u pogledu uticaja komponente pod otkazom na ostatak *TTEthernet* mreže. Iako teza razmatra mnogo širi prostor slučajeva nego što je dato u radu na kojem su radili Li i saradnici [26], do finalnog

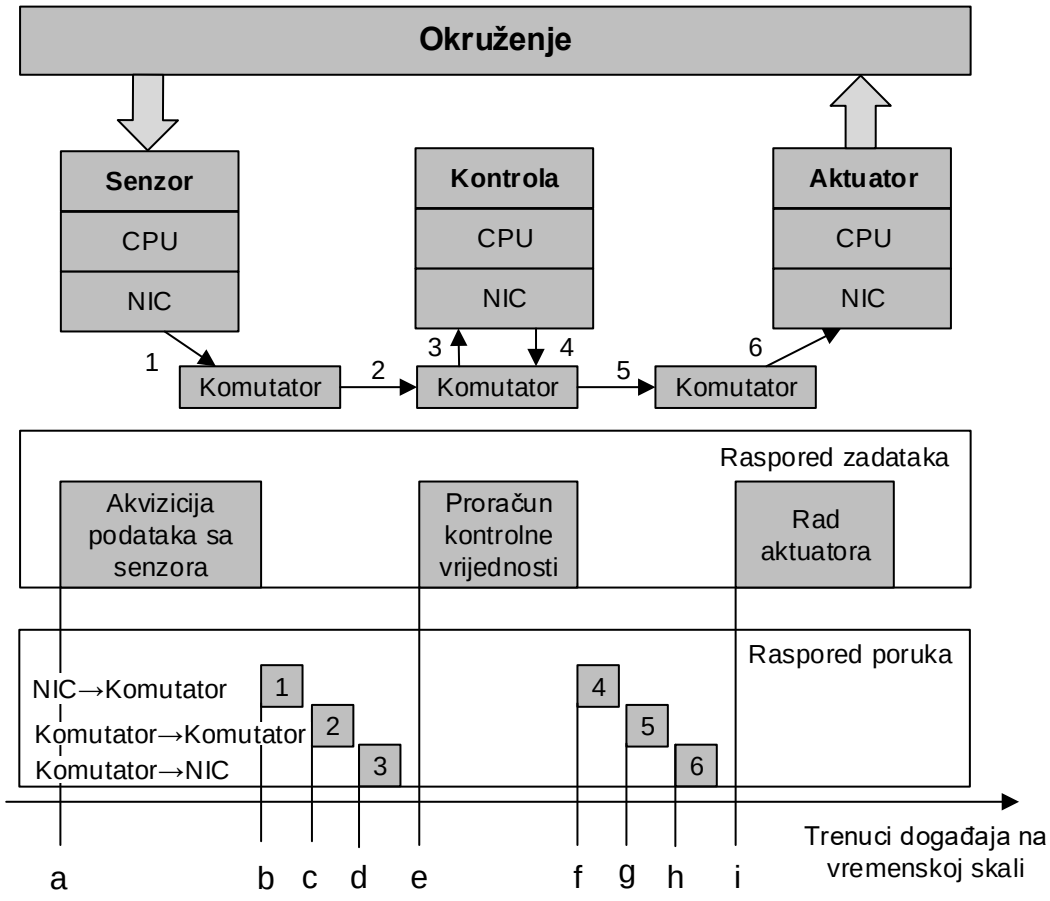
rezultata se ipak dolazi relativno brzo, jer se primjenjuje strategija genetskog algoritma, sa odgovarajućim izborom vrijednosti parametara genetskog algoritma. U slučaju čistog iterativnog pristupa, bez primjene strategije genetskog algoritma, proces nalaženja najgoreg slučaja pokretanja *TTEthernet* mreže trajao bi znatno duže. Međutim, ukoliko nisu racionalno izabrani parametri genetskog algoritma (npr. nepotrebno izabrana prevelika veličina populacije), performanse genetskog algoritma u pogledu brzine mogu da budu i lošije od čistog iterativnog pristupa [31]. Takođe, prednost teze u odnosu na rad koji su objavili Li i saradnici [26], je što teza nalazi najgori slučaj, dok spomenuti rad analizira samo jedan specifični slučaj pokretanja *TTEthernet* mreže. Ako se teza uporedi sa istraživanjem na kojem su radili Onučekva i Obermajser [25], gdje se posmatra slučaj jedne specifične sekvence okvira i njen uticaj na ostatak mreže, može se reći da teza ide korak dalje, jer ispituje ponašanje različitih sekvenci PCF okvira koje se mijenjaju pod uticajem genetskog algoritma. Na taj način, pronalaze se sekvence koje imaju najveći uticaj na uspostavljanje normalnog režima rada mreže, tj. koje utiču na najduže trajanje faze pokretanja *TTEthernet* mreže.

S obzirom da bez uspostavljene usklađenosti časovnika u mreži nije moguć prenos TT okvira u mreži, očigledno je da najbitnije funkcije posmatranog bezbjednosno-kritičnog sistema neće biti dostupne čitav vremenski period dok se *TTEthernet* mreža ne oporavi od komponente pod otkazom. Samom procjenom najgoreg slučaja za vrijeme pokretanja *TTEthernet* mreže, teza daje odgovor koliko dugo najkritičniji servisi mogu biti nedostupni u posmatranom sistemu, u slučaju nepredviđenih okolnosti kada određena komponenta počne nekontrolisano da emituje PCF okvire.

3 Koncept determinističke komunikacije

Iako je razvijen prije skoro 50 godina, *Ethernet* protokol se i danas intenzivno koristi u aplikacijama kao što su *web,* umrežavanje personalnih računara, institucija, itd. Međutim, u primjenama koje zahtjevaju rad u realnom vremenu, klasični *Ethernet* protokol ne daje dobre rezultate prvenstveno iz razloga što ne postoje garancije u pogledu pouzdanosti, otpornosti na otkaze i ograničenog kašnjenja paketa kroz mrežu [32]. U ovakvim sistemima primjene su našle deterministčke mreže.

Kod deterministčkih mreža, ponašanje saobraćaja je takvo da se ono može predvidjeti u svakom trenutku, tj. momenti slanja paketa su poznati, kao i vremenski intervali u kojima se može očekivati njihov prijem na prijemnoj strani. Opšti koncept jednog sistema koji koristi deterministčki vid komunikacije prikazan je na Sl. 1 [1], [9].



Sl. 1. Koncept deterministčkog sistema [9]

Komponente u mreži su povezane preko komutatora, te svaka od njih posjeduje mrežnu karticu NIC (eng. *Network Interface Card*), CPU (eng. *Central Processing Unit*), te senzor ili aktuator u zavisnosti od namjene uređaja. Prikupljanje podataka iz okruženja se vrši preko

senzora, dok se djelovanje na okruženje obavlja preko aktuatora. Sa Sl. 1 je vidljivo da slanje poruka sa prvog uređaja počinje tek nakon završetka faze akvizicije podataka, kao i da faze proračuna i aktivnosti aktuatora počinju tek nakon prijema neophodnih poruka. Svi ovi momenti su unaprijed proračunati, tako da je u svakom trenutku poznato kada neka poruka treba da se emituje sa određenog uređaja.

Posmatrajući Sl. 1, vidljivo je da je sistem podešen tako da se akvizicija podataka sa senzora vrši u trenutku *a*. S obzirom na karakteristike korišćenog senzora, poznato je maksimalno vrijeme koje mu je potrebno da dobavi podatak iz okruženja. Pošto je ovo vrijeme statički parametar, moguće je odrediti trenutak *b* kojem će senzor poslati očitani podatak. Takođe, vrijeme koje je potrebno komutatoru da proslijedi poruku je poznato (trenutak *c*), kao i maksimalno vrijeme koje je potrebno kontrolnom uređaju da izvrši neophodne proračune nakon prijema posmatrane poruke (trenutak *d*). Poruka se dalje prosljeđuje aktuatoru u takođe poznatim trenucima (*f, g, h*), a aktuator je konačno prima u trenutku *i* [1], [9].

S obzirom na činjenicu da je posmatrani sistem u potpunosti predvidljiv, to u velikoj mjeri olakšava proces testiranja, otklanjanja nastalih grešaka, kao i proces sertifikacije. Sertifikacija je od ključnog značaja za deterministčke mreže. Projektovanje posmatranog bezbjednosno-kritičnog determinističkog sistema mora biti odobreno od strane nezavisnog sertifikacionog tijela, iz razloga izbjegavanja opasnosti po život, zdravlje, vlasništvo ili životno okruženje. Takođe, u situacijama gdje se koriste replicirane komponente kao što je npr. redundantna konfiguracija komutatora, u deterministčkim mrežama se smatra da obje komponente za iste ulazne pobude daju jednake izlaze. Ukoliko dodje do otkaza, sistem je sposoban da procijeni koje poruke dolaze od ispravne, a koji od neispravne komponente. Samim tim, ispravna komponenta može da maskira signale neispravne komponente [3].

3.1 Pojam stanja sistema

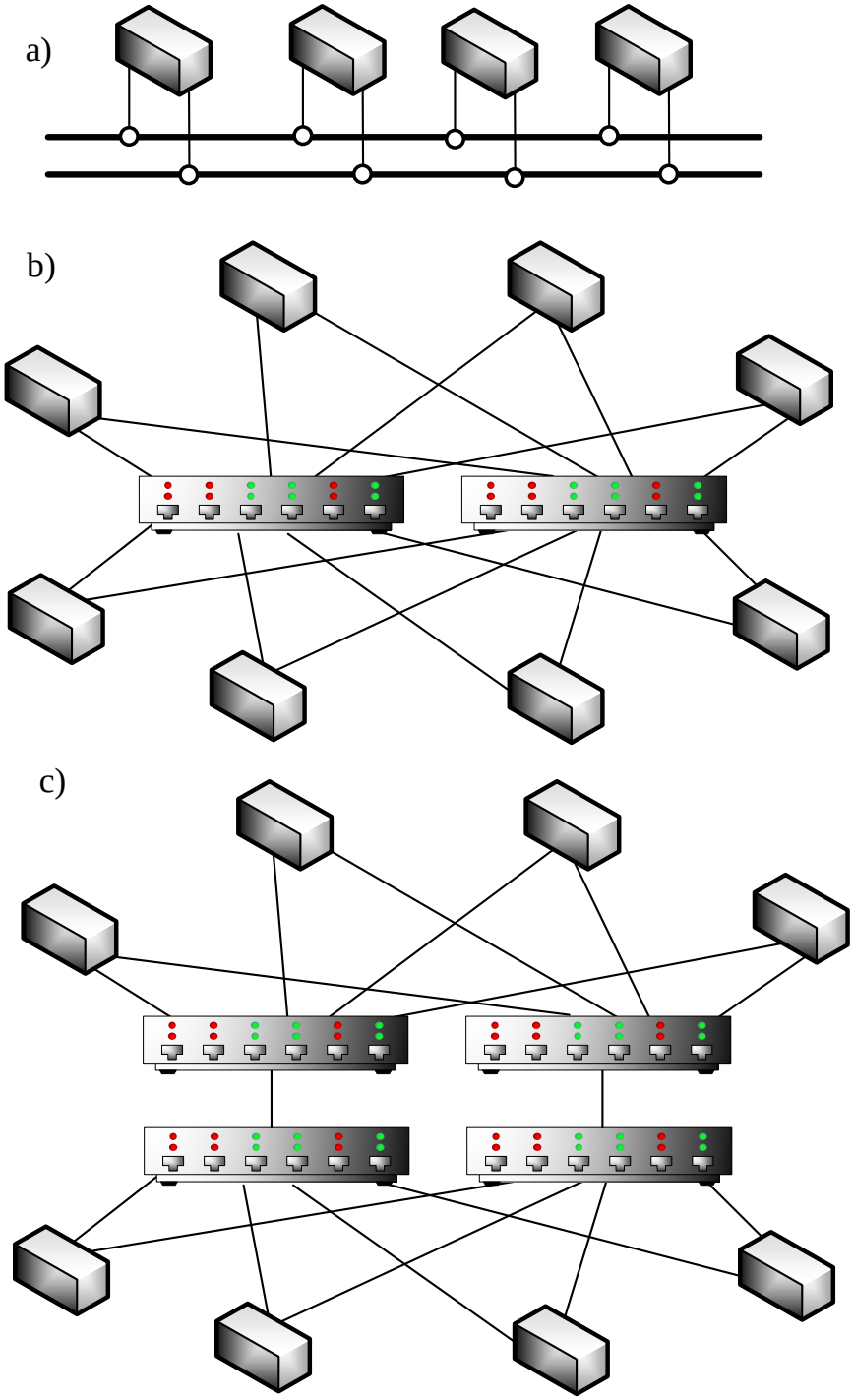
Svaki deterministički sistem treba da se odlikuje osobinom kauzalnosti. Kauzalnost podrazumijeva da posljedica uvijek slijedi nakon uzroka, te obrnuti slučaj ne važi. Obermajser (*Roman Obermaisser*) [3] definiše determinizam na sledeći način: *Posmatrani sistem se ponaša kao deterministički ako za dato početno stanje* u *trenutku* t *i skup budućih pobuda, svako buduće stanje sistema u svakom trenutku je u potpunosti predvidljivo. U*

distribuiranom sistemu koji radi u realnom vremenu, konzistentna definicija početnog stanja je moguća samo ako početno stanje svakog uređaja u sistemu je takvo da razgraničava prošle od budućih događaja. Drugim riječima ne postoji ni jedno drugo stanje koje je moglo imati uticaja na uređaj, ako se on nalazi u svom početnom stanju.

Uvođenjem pojma stanja sistema, olakšava se konzistentno razdvajanje prošlih od sadašnjih i budućih događaja. Pojam stanja sistema je široko rasprostranjen u računarskim naukama, a što se tiče sistema koji rade u realnom vremenu, odgovarajuću definiciju daje Mesarović [33]: *Stanje sistema omogućava predviđanje izlaza sistema u budućem trenutku isključivo na osnovu trenutnog stanja sistema u tom trenutku i ulaza kojim se djeluje na sistem u tom trenutku. Trenutno stanje sistema odražava i sva prethodna stanja sistema, jer je poznato koja mu stanja prethode. Drugim riječima, trenutno stanje sistema daje poznavanje i svih prethodnih njegovih stanja.*

3.2 Topologije mreža u bezbjednosno-kritičnim sistemima

Umrežavanje uređaja u deterministčku mrežu je slično kao kod klasičnih računarskih mreža. Međutim, u bezbjednosno-kritičnim sistemima gdje se vodi računa o otpornosti na otkaze, umjesto jednog kanala koji povezuje uređaje, najčešće se koriste redundantne konfiguracije determinističkih mreža kao što je ilustrovano na Sl. 2, gdje je korišćen stepen redundanse 2. Svaka od topologija ima svoje prednosti i nedostatke u pogledu cijene, pouzdanosti i performansi.



Sl. 2. Topologije koje se koriste u deterministčkim mrežama: a) Magistrala, b) Zvijezda, c) Kaskadna zvijezda [3]

Najjeftinija topologija je magistrala (Sl. 2a), gdje je svaka komponenta povezana na fizički medijum. U ovakvoj konfiguraciji je neophodno koristiti tipske impedanse tzv. terminirajuće otpornike ili „terminatore“ na krajevima voda da bi se umanjio uticaj refleksija na normalnu komunikaciju. Topologija magistrala nije preporučljiva za veće mreže, pošto je

u njoj relativno teško identifikovati nastale probleme. Ne postoji centralna komponenta za usmjeravanje poruka, nego su svi krajnji uređaji (eng. *End System*) povezani međusobno preko istog medijuma, tako da se sve poruke u ovakvoj konfiguraciji se šalju kao difuzne (eng. *broadcast*) poruke. Propagaciono kašnjenje poruke zavisi od dužine kabla i brzine prostiranja signala, koja je tipično jednaka 2/3 brzine svjetlosti na bakarnim vodovima, mada ova vrijednost može blago da varira [3].

Znatno češće se koristi topologija tipa zvijezda (Sl. 2b). Centralna komponenta mreže je komutator koji je zadužen za usmjeravanje poruka između uređaja. S obzirom na činjenicu da je svaki krajnji uređaj u ovakvoj konfiguraciji povezan na komutator, svaka poslata poruka mora proći kroz komutator prije nego što stigne na svoje odredište. Kašnjenje poruke ne zavisi samo od brzine prostiranja signala kroz kablove, nego i od kašnjenja komutatora koje se mora uzeti u obzir pri proračunu parametara determinističke komunikacije u posmatranoj mreži. Iako je cijena realizacije ovakvih mreža znatno veća nego kod topologija tipa magistrala, ovdje postoji niz prednosti kao što je npr. manje opterećenje veze (eng. *link load*) [34], veći broj različitih poruka može biti proslijeđen između više uređaja u istom trenutku, a takođe je podržano i emitovanje višesmjernih (eng. *multicast*) i difuznih poruka. Takođe, u ovoj konfiguraciji je lakše prepoznavanje grešaka za krajnje uređaje kod kojih je došlo do otkaza. Da bi se obezbjedila proširivost (eng. *scalability*), moguće je koristiti kaskadne zvijezda topologije (Sl. 2c), što znači da se ne moraju svi krajnji uređaji vezati na isti komutator. Pored navedenih osnovnih topologija, moguće su i njihove međusobne kombinacije [3].

3.3 Pouzdanost sistema

Pouzdanost sistema (eng. *reliability*) predstavlja vjerovatnoću da će sistem obezbjediti korektan rad servisa do trenutka t, u odnosu na trenutak početka rada sistema $t = t_0$ [35]. Vjerovatnoća da će doći do otkaza sistema u posmatranom vremenskom intervalu je izražena parametrom λ, koji predstavlja učestanost otkaza i mjeri se u *FIT* (eng. *Failure In Time*). Učestanost otkaza 1 *FIT* znači da srednje vrijeme do otkaza posmatranog uređaja MTTF (eng. *Mean Time To Failure*) iznosi 10^9 časova, tj. jedan otkaz se javlja u 115,000 godina, za sisteme koji nemaju mogućnost da se oporave od otkaza. Ukoliko sistem ima konstantnu učestanost otkaza λ grešaka po času, pri čemu zanemarujemo uticaj starenja materijala, tada se pouzdanost sistema $R(t)$ može izraziti kao [35]:

$$R(t) = \frac{1}{e^{\lambda(t-t_0)}} \quad (1)$$

gdje je (t - t_0) izraženo u časovima. Inverzna vrijednost $1/\lambda$ jednaka je parametru MTTF. Sistemi koji imaju MTTF ~ 10^{-9} otkaza po času, svrstavaju se u tzv. izuzetno-pouzdane sisteme [35].

3.3.1 Bezbjednost

Postoji više definicija za pojam bezbjednosti (eng. *safety*). Roland i Moriarty [36] definišu bezbjednost kao mjeru kvaliteta sistema da održava svoj normalan rad sa prihvatljivim minimumom slučajnih otkaza. Prema standardu MIL-STD-882E [37], bezbjednost predstavlja odsustvo uslova koji mogu prouzrokovati gubitak ljudskih života, povredu, profesionalnu bolest, oštećenje ili gubitak imovine, kao i štetu po životnu sredinu. Kopetz [35] definiše bezbjednost kao pouzdanost u odnosu na nivo kritičnosti otkaza. Otkaz može biti maligni ili benigni [35].

Maligni otkaz je takav da cijena njegovih posljedica može višestruko da prevazilazi cijenu korišćenja posmatranog sistema u normalnom režimu rada [35]. Definicija malignog otkaza poklapa se sa iznad spomenutim uslovima definicije bezbjednosti iz standarda MIL-STD-882E [37]. Sistemi gdje postoji mogućnost pojave malignog otkaza, nazivaju se bezbjednosno-kritični sistemi (eng. *safety-critical systems*). Neki od primjera malignog otkaza su npr. pad aviona zbog greške u sistemu za upravljanje, ili automobilska nesreća nastala zbog greške u inteligentnom računarskom sistemu za upravljanje kočenjem. Baš iz ovih razloga, ovakvi bezbjednosno-kritični sistemi moraju biti izuzetno-pouzdani. Slične učestanosti otkaza odnose se i na sisteme kao što su nuklearne elektrane ili sistemi signalizacije u željeznicama. Svi ovi sistemi moraju da prođu proces sertifikacije da bi uopšte mogli da se stave u funkciju [35]. Nasuprot malignom otkazu, cijena gubitaka koje uzrokuje benigni otkaz jednaka je vremenu trajanja tog otkaza dok sistem radi u normalnom režimu. Drugim riječima, pojava benignog otkaza ne uzrokuje katastrofu [38].

3.3.2 Kvalitet održavanja

Za sisteme kod kojih je moguć oporavak od otkaza [39], kvalitet održavanja predstavlja mjeru vremena koje je potrebno za otklanjanje nastalog kvara na sistemu nakon pojave benignog otkaza. Izražava se pomoću vjerovatnoće *M(d),* koja daje informaciju da će se otklanjanje kvara trajati najduže vremenski interval *d* nakon pojave otkaza. Konstantna

učestanost oporavka sistema po času μ, zajedno sa MTTR (eng. *Mean Time To Repair*) daju kvantitativnu mjeru kvaliteta održavanja [35].

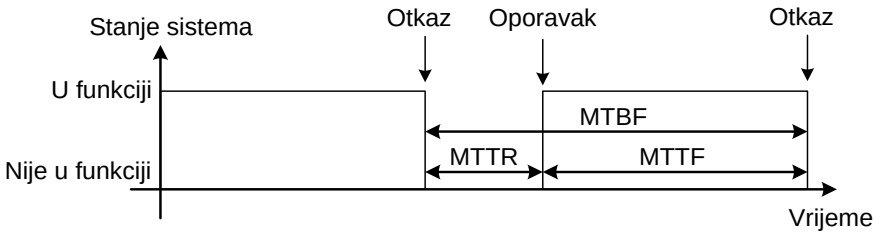
Da bi se omogućila jednostavna zamjena komponente kod koje je došlo do otkaza, potrebno je da sistem bude podijeljen u više cjelina (eng. *Field Replaceable Units - FRU*), koje su međusobno povezane spregama (eng. *interface*) odgovarajućim za jednostavnu zamjenu nakon otkaza komponente [35].

3.3.3 Dostupnost

Dostupnost (eng. *availability*) predstavlja mjeru kvaliteta sistema u odnosu na smjenu perioda pravilnog i nepravilnog rada. Računa se na sledeći način [35]:

$$A = \frac{MTTF}{MTTF+MTTR} = \frac{MTTF}{MTBF} \tag{2}$$

Suma MTTF + MTTR se definiše kao MTBF (eng. *Mean Time Between Failures*). Veza između spomenutih parametara ilustrovana je na Sl. 3.



Sl. 3. Veza između vremena MTTR, MTTF i MTBF [13]

Visoka dostupnost postiže se ili velikom vrijednošću MTTF, ili malom vrijednošću trajanja MTTR. Projektant ima slobodu izbora vrijednosti ovih parametara pri projektovanju visoko-dostupnih sistema.

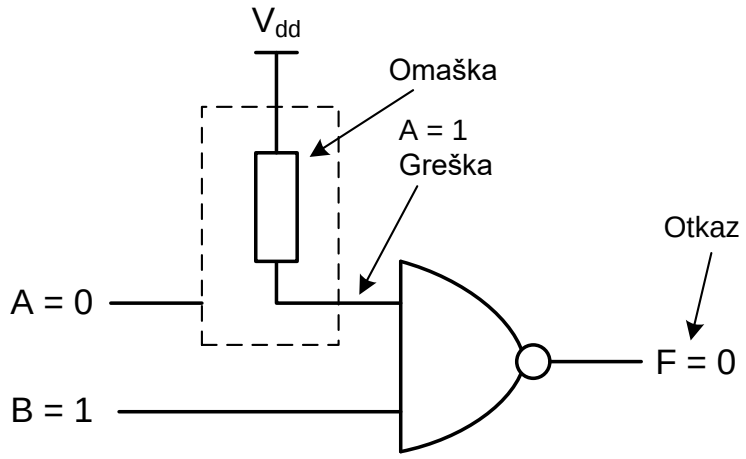
Kao jedan od primjera za posmatranje dostupnosti nekog sistema možemo uzeti telefonski aparat. Kada korisnik podigne slušalicu, telefonska centrala treba da bude spremna da omogući uslugu sa veoma visokom vjerovatnoćom. Prekid dostupnosti usluge (MTTR) može da iznosi nekoliko minuta u toku godine [35].

3.4 Omaške, greške i otkazi u deterministčkim mrežama

Tri pojma su bitna pri definisanju nepravilnog ponašanja sistema: omaška (eng. *fault*), greška (eng. *error*) i otkaz (eng. *failure*). Omaška dovodi do odstupanja u očekivanom internom stanju sistema (greška), što dalje može dovesti do budućeg otkaza koji se manifestuje na spoljnjoj granici sistema.

Kao posljedica fizičkih omaški (npr. kratki spoj, prekid veze), omaški u softveru (eng. *bug*), kao i sistemskih omaški , može doći do greške u komponenti a zatim i do otkaza na nivou sistema ili cjelokupne mreže. Spomenute omaške se svrstavaju u unutrašnje, dok npr. elektromagnetna interferencija ili pogrešni ulazni podaci se svrstavaju u spoljašnje greške [3] [38].

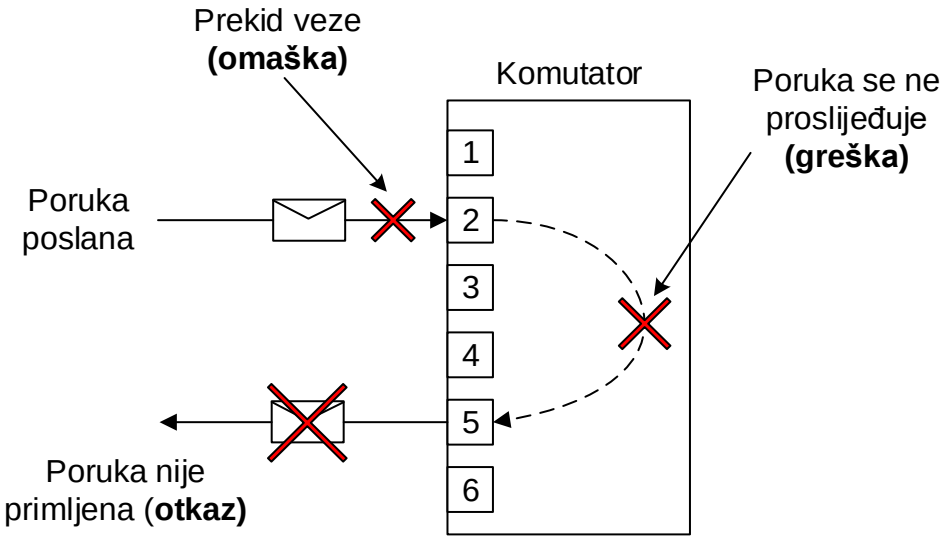
Ove pojmove ćemo objasniti na jednostavnom primjeru NI kola prikazanom na Sl. 4, kod kojeg se omaška događa na jednoj od njegovih ulaznih linija. Iako je na liniji A dovedena vrijednost logičke nule, na ulazu u NI kolo se registruje visok logički nivo. Dobijeni rezultat na izlazu logičkog kola pokazuje da se funkcija posmatranog logičkog kola ne obavlja uspješno, što se smatra otkazom [40].



Sl. 4. Odnos između omaške, greške, i otkaza, na primjeru NI logičkog kola [40]

Odnos omaške, greške i otkaza takođe možemo objasniti i na primjeru jedne mrežne komponente, tj. komutatora sa 6 priključaka, kao što je ilustrovano na Sl. 5. Predajnik šalje poruku na priključak 2 komutatora, koju komutator treba da proslijedi na svoj priključak 5. Međutim, zbog prekida veze (omaška) komutator ne prima poruku. Samim tim, komutator ne

proslijedjuje poruku na svoj priključak 6 (greška). Prijemnik koji je priključen na priključak 5 komutatora prepoznaje otkaz, jer ne prima očekivanu poruku.



Sl. 5. Odnos između omaške, greške, i otkaza, na primjeru komutatora

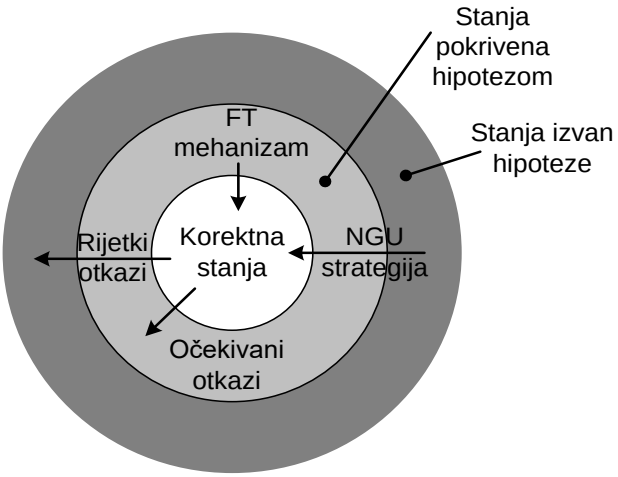
3.4.1 **Zadržavanje otkaza**

Da bi se spriječilo širenje otkaza na cjelokupan sistem u slučaju otkaza jedne komponente ili grupe komponenata u sistemu, neophodno je podijeliti sistem u takve cjeline koje ne utiču jedna na drugu u slučaju otkaza neke od njih. Ovakve cjeline se nazivaju regioni koji zadržavaju otkaz – FCR (eng. *Failure Containment Region*). Neophodno je da postoje odgovarajući mehanizmi u sistemu koji mogu prepoznati i izolovati nastali otkaz u FCR cjelini, da bi se spriječilo narušavanje normalnog rada drugih FCR cjelina. Takođe, sistem mora biti projektovan tako da otkaz jedne FCR cjeline nema uticaja na normalan rad sistema, čak i ako je vjerovatnoća takvog događaja veoma mala.

Za komponente koje se nalaze unutar jedne FCR cjeline ne može se reći da su otporne na uticaj komponente u kojoj je došlo do otkaza, a nalazi se u istoj FCR cjelini. Drugim riječima, komponente unutar iste FCR cjeline nisu nezavisne jedna od druge. Unutar jedne FCR cjeline nalaze se sve one komponente koje dijele zajednički resurs: izvor napajanja, izvor vremena, servis usklađivanja vremena, fizički prostor, itd. Na primjer, ukoliko dvije komponente zavise od istog izvora vremena (dijele isti oscilator), ne može se reći da su nezavisne jedna od druge, te pripadaju istoj FCR cjelini [38].

3.4.2 **Hipoteza otkaza**

Svaka deterministička mreža mora imati jasno definisanu hipotezu otkaza. Hipoteza otkaza daje informaciju koje otkaze sistem mora da toleriše u slučaju njihove pojave, kao što definiše sve FCR cjeline i njihova ponašanja u slučaju otkaza. Pored toga, hipoteza otkaza definiše i učestanost pojavljivanja otkaza. Za sve slučajeve koji su definisani u hipotezi otkaza važi da moraju biti tolerisani, dok se za slučajeve koji su izvan hipoteze smatra da je njihova vjerovatnoća pojave veoma mala. Primjer događaja koji bi mogao da ne bude obuhvaćen hipotezom otkaza je slučaj kada bi u istom trenutku otkazale 2 komponente u posmatranom sistemu, što predstavlja rijedak slučaj sa malom vjerovatnoćom pojave. Otkaz jedne komponente u mreži ulazi u oblast hipoteze otkaza. Na Sl. 6 je ilustrovan koncept hipoteze otkaza [3], [35].



Sl. 6. Prostor stanja sistema otpornog na otkaze [35]

Ukoliko se dogodi otkaz koja je pokriven hipotezom, odgovarajući mehanizam će izvršiti neophodne korake i vratiti sistem u domen korektnih stanja. Međutim, ako se dogodi neki od rijetkih otkaza, to će dovesti sistem u stanje koje nije pokriveno hipotezom. U svakom slučaju, sistem neće ostati u ovom stanju, nego će se primjeniti NGU (eng. *Never Give Up*) strategija koja će pokušati da vrati sistem u domen korektnih stanja. NGU strategija najčešće podrazumijeva ponovno pokretanje podsistema u kojem je došlo do otkaza. Takođe, za primjenu NGU strategije neophodno je da postoji ostvaren mehanizam u sistemu za prepoznavanje narušavanja hipoteze otkaza. Jedan od ovakvih mehanizama posjeduje TTP (eng. *Time Triggered Protocol*) protokol, koji na osnovu podataka u porukama koje se razmjenjuju između komponenata u mreži, procjenjuje da li su neke od tih komponenata pod

otkazom. Ovi podaci se prenose u tzv. vektoru pripadnosti (eng. *membership vector*) o kojem će biti riječi u narednim poglavljima ovog rada. [3], [35].

3.4.3 Ponašanje komponente pod otkazom

U slučaju otkaza, posmatrana komponenta može različito da se ponaša. Scenariji mogu biti sledeći [3]:

- *Fail-Stop*: Komponenta ne reaguje na ulaze, tj. ne generiše nikakve izlaze sve dok se ponovo ne pokrene. Ovakav otkaz može da se prepozna od strane ostalih komponenata kod kojih nije došlo do otkaza.
- *Crash*: Isti slučaj kao i prethodni, sa razlikom da *Crash* otkaz može ostati neprepoznat od strane ispravnih komponenata.
- *Omission*: Pošiljalac ne uspijeva da pošalje poruku, ili primalac ne uspijeva da primi poruku. Prepoznavanje ovog otkaza nije garantovano.
- *Timing*: Komponenta ne zadovoljava podešene vremenske parametre, tj. izvršava radnje prerano ili prekasno.
- *Byzantine*: Nepredvidljivo ponašanje komponente. Može uključivati i falsifikovanje poruka.
- *Babbling Idiot*: Komponenta konstantno nekontrolisano šalje poruke. Ovakvo ponašanje komponente može da ugrozi normalan rad mreže koja nije podijeljena u FCR cjeline.
- *Slightly-off-Specification (SoS)*: Poseban slučaj *Byzantine* otkaza koji se ogleda u nepravilnom tajmingu. Npr. ako posmatrana komponenta pod otkazom emituje neku poruku izvan, ali veoma blizu vremenskih limita do kada je ta poruka stvarno trebala da bude emitovana, onda komponente koje nisu pod otkazom mogu različito da tumače tu poruku. Zbog nesavršenosti lokalnih časovnika korišćenih u mrežnim komponentama, spomenuta poruka može biti klasifikovana kao ispravna od strane nekih komponenata, dok kod drugih može biti klasifikovana kao neispravna. Lokalni časovnici upravljaju otvaranjem i zatvaranjem vremenskih prozora u kojima se očekuje poruka, tako da poruka poslata od strane komponente pod otkazom može stići unutar prozora kod nekih komponenata, dok kod drugih komponenata stiže izvan prozora.
- *Masking*: Komponenta pod otkazom šalje pakete pod identitetom druge komponente.

3.5 Parametri deterministične komunikacije

Pri projektovanju deterministične ili mreže sa više prioriteta, uzimaju se u obzir njeni parametri da bi se mogao napraviti odgovarajući proračun kada određena poruka treba biti poslata sa predajne strane, a kada se može očekivati njen prijem na prijemnoj strani.

3.5.1 Parametri poruka

Što se tiče parametara samih poruka, oni zavise od vrste poruke, tj. da li se radi o determinističkoj, sporadičnoj, ili naletnoj poruci. Obermajser [3] deterministične poruke definiše kao periodične, dok poruke naletnog saobraćaja karakteriše kao aperiodične. Zahtjev komponente za slanjem poruke označavamo sa $\rho_{k,m} \in R^+ (k \in N)$, gdje je $\rho_{k,m}$ stohastička promjenljiva, dok parametri m, k reprezentuju komponentu i instancu poslane poruke, respektivno. Svake 2 susjedne poruke poslane sa komponente m mogu se se predstaviti sledećom jednačinom:

$$\forall k \in N: \rho_{k+1,m} - \rho_{k,m} = d_m + \delta_{k,m} \tag{3}$$

gdje parametar d_m predstavlja minimalno rastojanje između poruka, a $\delta_{k,m}$ varijabilnost tog rastojanja u granicama [0, u_m]. Parametar u_m predstavlja maksimalnu vrijednost odstupanja spomenutog intervala. Ako bi spomenute tri vrste poruka opisivali pomoću parametara iz (3), mogli bi da ih definišemo na sledeći način:

$$\begin{gathered} m_{deterministička}: \left(\delta_{k,m} = 0, \forall k \in N\right), (d_m \in R^+) \\ m_{sporadična}: \left(0 \leq \delta_{k,m} \leq u_m, \forall k \in N\right), (d_m \in R^+) \\ m_{naletna}: \left(0 \leq \delta_{k,m} \leq u_m, \forall k \in N\right), (d_m = 0) \end{gathered} \tag{4}$$

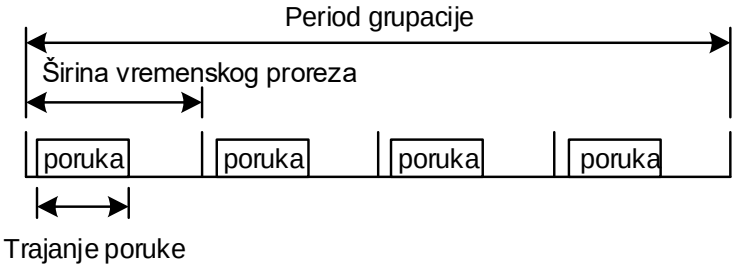
Kao što se može vidjeti iz (4), kod determinističke vrste poruka postoji definisana fiksna vrijednost perioda d_m, a stohastičko odstupanje trajanja tog perioda ne postoji, tj. $\delta_{k,m} = 0$. Kod sporadične klase saobraćaja poruke se šalju sa periodom d_m koji može da odstupa od svoje definisane vrijednost maksimalno za vrijednost u_m. Kod naletnog tipa poruka period poruka nije definisan jer se poruke šalju bez unaprijed utvrđenog rasporeda [3].

3.5.2 Parametri deterministične mreže

Među najvažnije parametre deterministične mreže svrstavaju se: opterećenje veze, kašnjenje i varijabilnost kašnjenja. Kao i odabir odgovarajućeg tipa poruka za prenos podataka, navedeni parametri su takođe od izuzetnog značaja pri projektovanju mreže, a pogotovo u smislu izrade rasporeda slanja poruka deterministčkim putem.

3.5.2.1 Opterećenje veze

Veoma bitan parametar u deterministčkim mrežama je opterećenje ili iskorišćenost veze (eng. *bandwidth utilization*). U zavisnosti od ovog parametra određuje se koliko novih poruka može da se prenese preko posmatrane veze. Kod deterministčkih mreža svaka poruka ima svoj vremenski prorez (eng. *slot*) u kojem se prenosi, kao što je prikazano na Sl. 7. Period grupacije (eng. *cluster period*) predstavlja vremenski interval u kojem je napravljen raspored slanja poruka (eng. *schedule*), i on se ponavlja tokom cjelokupne razmjene poruka ukoliko je spomenuti raspored napravljen nekim od cikličkih raspoređivača (eng. *scheduler*). Samim tim, ponavlja se i raspored slanja poruka, tako da je u svakom trenutku poznato koja poruka treba da se šalje na predajnoj strani.



Sl. 7. Ilustracija perioda grupacije sa definisanim rasporedom slanja paketa [34]

S obzirom da su trajanja paketa i trajanja njihovih vremenskih proreza poznata, te imajući u vidu da se isti period grupacije (eng. *cluster period*) ponavlja u toku cjelokupne razmjene poruka, moguće je jednoznačno odrediti opterećenje posmatrane veze za instance poruke *m* na sledeći način [34]:

$$BWU[\%] = \frac{T_p}{d_m} \cdot 100\%, \qquad T_p[s] = \frac{b_m[bit]}{L_{speed}[bit/s]} \tag{5}$$

gdje je T_p trajanje poruke, b_m dužina poruke u bitima, a L_{speed} brzina veze. Iz izraza (5) je vidljivo da povećavanjem veličine paketa *BWU* linearno raste, dok se povećavanjem brzine veze *BWU* smanjuje. Ukoliko se ovakav princip koristi kod mreža sa više prioriteta, sav prostor koji je slobodan u periodu grupacije može se iskoristiti za slanje naletne klase saobraćaja.

Naletna klasa takođe doprinosi konačnom opterećenju veze, ali nema uticaja na prenos podataka koji pripadaju determinističkoj klasi saobraćaja. Sa aspekta determinističkog saobraćaja, može se smatrati da naletna klasa saobraćaja ne postoji u mreži, jer je je prostor za determinističke poruke rezervisan na vremenskoj skali. Međutim, poruke saobraćaja naletnog tipa mogu se slati samo u onim trenucima kada slanje determinističke poruke nije u toku. Da bi se omogućilo ravnomjerno slanje naletnih poruka, primjenjuju se odgovarajući algoritmi prorjeđivanja rasporeda poruka determinističke klase saobraćaja. Jedan od primjera optimizacije rasporeda slanja determinističkih poruka u *TTEthernet* mreži za svrhe ravnomjernijeg prenosa poruka naletne klase saobraćaja objašnjen je u poglavlju 5.1.1.

3.5.2.2 **Kašnjenja u deterministčkim mrežama**

Pri proračunu trenutaka kada treba da budu poslate ili primljene poruke u determinističkim mrežama, veoma bitan parametar predstavljaju njihova kašnjenja. Postoje dvije vrste kašnjenja koja su od značaja u determinističkim mrežama: statička i dinamička. U statička kašnjenja svrstavaju se [3], [41]:

- **Kašnjenje prostiranja signala kroz medijum**: u zavisnosti od vrste voda (bakarni ili optički medijum) zavisi kolika će biti razlika u vremenu između emitovanja i prijema poruke. Ukupno kašnjenje medijuma dobija se sabiranjem kašnjenja kroz sve medijume od predajnika do prijemnika, uključujući i međustepene. Ovo kašnjenje je konstantno, tako da je nove proračune spomenutih trenutaka neophodno vršiti samo ako se postojeći medijum zamijeni sa medijumom drugačije dužine.

- **Kašnjenje slanja poruke**: zavisi od brzine elektronskih sklopova na predajnoj strani mrežne komponente.

- **Kašnjenje prijema poruke**: zavisi od brzine elektronskih sklopova na prijemnoj strani komponente. Ako se radi o međustepenu kao što je komutator, uzimaju se u obzir i kašnjenje prijema i kašnjenje slanja, kao i brzina procesiranja podataka, tj. procjena na koji priključak (eng. *port*) treba proslijediti primljenu poruku.

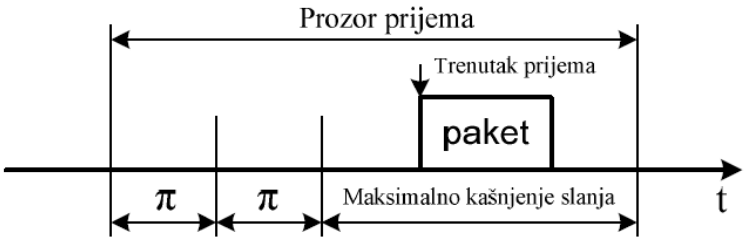
Tipične vrijednosti kašnjenja predajne i prijemne strane za komercijalne uređaje na fizičkom sloju (eng. *Physical Layer - PHY*) koji se koriste u mrežnim komponentama, za brzine 10Mb/s i 100Mb/s, sumirane su u Tabela 1 [42].

Tabela 1. Tipične vrijednosti kašnjenja fizičkog sloja za neke komercijalne uređaje [42]

10 Mb/s		**100 Mb/s**	
Intel LXT970A			
Kašnjenje predaje	300 ns – 500 ns	Kašnjenje predaje	40 ns – 60 ns
Kašnjenje prijema	7.3 µs	Kašnjenje prijema	20 ns
LSI-Logic L80225			
Kašnjenje predaje	600 ns	Kašnjenje predaje	60 ns – 140 ns
Kašnjenje prijema	3.6 µs	Kašnjenje prijema	240 ns
National Semiconductor DP83847A			
Kašnjenje predaje	680 ns	Kašnjenje predaje	60 ns
Kašnjenje prijema	1.73 µs	Kašnjenje prijema	21 ns

Za razliku od statičkih kašnjenja koja su konstante, kod dinamičkih kašnjenja nije moguće precizno odrediti kada će posmatrana poruka biti poslana ili primljena. Dinamičko kašnjenje se odnosi na samo na predajnik, što znači da druge komponente na njega nemaju uticaja. Stvarni trenutak slanja poruke t_{send} nije jednak trenutku kada je zahtijevano slanje te poruke $t_{request}$. Razlika između $t_{request}$ i t_{send} naziva se vrijeme pristupa d_{access}. U trenutku $t_{request}$ kada se zahtijeva slanje poruke m_1 na priključku neke komponente, može se dogoditi situacija da je u tom trenutku na istom priključku u toku slanje druge poruke m_2 koja je višeg prioriteta od m_1. Stoga poruka m_1 će biti na čekanju za period d_{access} koje je određeno vremenom koje je potrebno da se kompletna poruka m_1 pošalje. Nakon što se završi slanje poruke m_2, nastupa trenutak t_{send} u kojem započinje slanje m_1. Ukoliko u posmatranom trenutku veza nije zauzeta, imamo situaciju $t_{send} = t_{request}$ [3].

Na prijemnoj strani mora se uzeti u obzir dinamičko kašnjenje poruke, kao i preciznost časovnika uređaja u mreži. U odnosu na ove parametre, definiše se vremenski okvir u kojem se očekuje prijem poruke, kao što je ilustrovano na Sl. 8. Spomenuti vremenski okvir se naziva prozor prijema (eng. *acceptance window*). Posmatranjem Sl. 8 uočava se da je prozor prijema jednak maksimalnom dinamičkom kašnjenju slanja paketa, uvećanog za dvostruku vrijednost preciznosti π. Preciznost predstavlja maksimalnu razliku između najbržeg i najsporijeg časovnika u mreži i o njoj će biti više riječi u narednom poglavlju. Takođe primjećujemo da ni jedan parametar statičkog kašnjenja ne figuriše u prozoru prijema, jer se prolongiranje vremena prijema koje je posljedica statičkog kašenjenja može unaprijed proračunati i unijeti u konfiguraciju mreže [43].



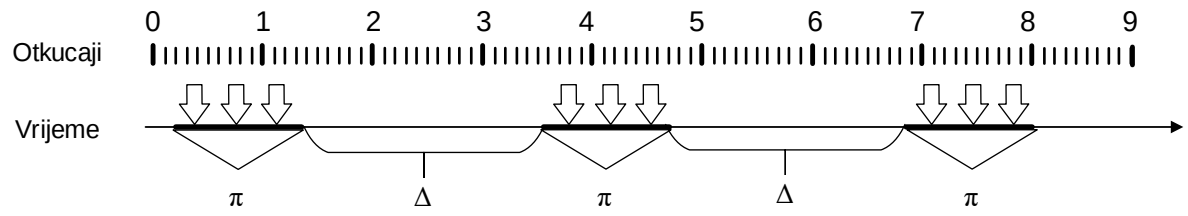
Sl. 8. Prozor prijema [43]

4 Usklađivanje vremena u deterministčkim mrežama

Usklađivanje vremena predstavlja jedan od najbitnijih aspekata deterministčkih mreža, jer ukoliko časovnici uređaja u mreži nisu međusobno usklađeni, razmjena poruka deterministčkim pristupom uopšte nije izvodljiva. Pod usklađivanjem vremena se podrazumijeva međusobno usklađivanje časovnika u mreži prema referentnom izvoru tačnog vremena koji diktira globalni pojam apsolutne vremenske skale zajedničke za sve uređaje u mreži. Postoje 2 vrste usklađivanja vremena: unutrašnje i spoljašnje. Predmet ovog rada je unutrašnje usklađivanje vremena, te će mu biti posvećeno najviše pažnje. Kod spoljašnjeg usklađivanja vremena koriste se spoljni izvori tačnog vremena, dok se kod unutrašnjeg usklađivanja vremena lokalni časovnici uređaja međusobno usklađuju, tj. referentni časovnici su ujedno i časovnici koji se usklađuju [3].

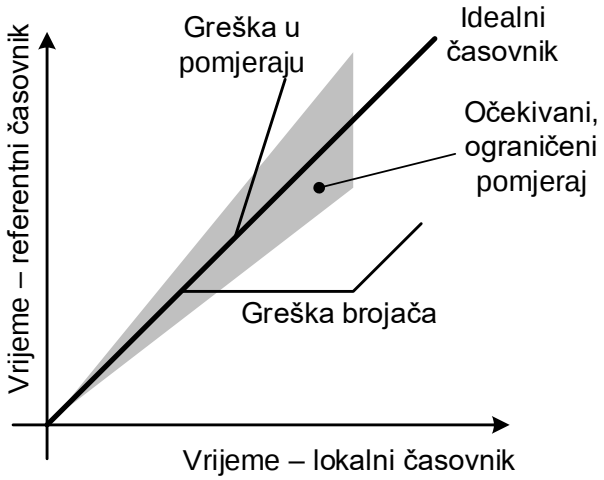
4.1 Digitalni časovnici i njihovi parametri

Za razliku od analognih časovnika, gdje nema diskontinuiteta na vremenskoj skali, tj. određeni događaj može da se dogodi u bilo kojem trenutku, digitalni časovnici rade prema diskretnoj vremenskoj skali (Sl. 9). Određeni događaj može da se dogodi samo u trenucima otkucaja (eng. *microtick - mt*) digitalnog časovnika ugrađenog u posmatranom uređaju. Rastojanje između 2 otkucaja časovnika se naziva period takta, ili granularnost digitalnog časovnika. Veća učestanost otkucaja časovnika daje finiju granularnost. Očigledno je da parametar granularnosti nije primjenjiv na analogne časovnike. Događaji na Sl. 9 označavaju događaje koji mogu da se dogode samo u intervalima označenim sa parametrom π (preciznost). Parametar Δ predstavlja periode neaktivnosti u kojima ne treba da se javljaju događaji.



Sl. 9. Diskretna vremenska skala [3]

S obzirom da je vrijeme kod digitalnih časovnika diskretan parametar, uvijek je prisutna greška diskretizacije u odnosu na stvarno vrijeme. Takođe, pošto oscilatori korišćeni u digitalnim časovnicima nisu idealni, uvijek je prisutno odstupanje vremena posmatranog digitalnog časovnika u odnosu na referentni, idealni časovnik. Odstupanje se javlja zbog nesavršenosti kristala kvarca, promjena u temperaturi, promjene u naponu baterije, itd. Opisana pojava definiše se kao pomjeraj (eng. *drift*) časovnika. Pored pomjeraja časovnika, događa se i greška brojača, kao što je ilustrovano na Sl. 10.



Sl. 10. Tipovi grešaka digitalnog časovnika [3]

Idealni lokalni časovnik u potpunosti preslikava vrijeme referentnog časovnika, tj. ima pomjeraj jednak 0. Svaki stvarni digitalni časovnik ima svoj ograničeni pomjeraj i ako se dogodi odstupanje od ovog nominalnog vremena, dolazi do greške pomjeraja. Takođe, ukoliko brojač prestane da broji određeni vremenski period, smatra se da je došlo do greške brojača.

Ako pretpostavimo da postoji jedan nezavisni spoljni posmatrač (eng. *observer*) koji posjeduje jedinstveni referentni časovnik *z* čija je učestanost otkucaja f_z, tada možemo vršiti procjenu pomjeraja lokalnih časovnika u mreži. Neophodno je da *z* ima dosta finiju granularnost (npr. 10^{15} otkucaja u sekundi) nego posmatrani lokalni časovnik, da bi adekvatno mjerenje uopšte bilo moguće. Što je bolja granularnost, manja je greška uzorkovanja. Ako se dogodi događaj *e* (otkucaj posmatranog lokalnog časovnika), posmatrač bilježi trenutak tog događaja *z(e)*. Pomjeraj lokalnog časovnika *k* između otkucaja *i* i otkucaja *i+1*, definiše se na sledeći način:

$$pomjeraj_i^k = \left| \frac{z\left(mt_{i+1}^k - mt_i^k\right)}{n^k} - 1 \right| \tag{6}$$

gdje je n^k broj otkucaja referentnog časovnika *z* između dva otkucaja *mt* časovnika *k.* Iako je časovnik *z* mnogo precizniji od posmatranih lokalnih časovnika koji se analiziraju, on ipak radi prema diskretnoj vremenskoj skali, što znači ako se između 2 otkucaja referentnog časovnika *z* dogodi više događaja, *z* neće moći da procijeni hronološki poredak događaja. Ovo je jedan od glavnih nedostataka referentnog časovnika *z.*

Rastojanje između 2 iste instance otkucaja *mt* od 2 različita časovnika u mreži *j* i *k* definiše se kao razlika (eng. *offset*) otkucaja *i:*

$$razlika_i^{jk} = \left|z\left(mt_i^j\right) - z\left(mt_i^k\right)\right| \tag{7}$$

Ako posmatramo grupu od *n* časovnika 1, 2, ..., *n*, možemo definisati preciznost π otkucaja *i* kao maksimalnu razliku vremena između 2 časovnika u mreži:

$$\pi_i = \max_{\forall 1 \le j,k \le n} \left\{razlika_i^{jk}\right\} \tag{8}$$

Maksimalna vrijednost π_i u jednom fiksnom intervalu od interesa definiše se kao preciznost grupe ili preciznost mreže, ako časovnici svih komponenata u mreži pripadaju istoj grupi. Drugim riječima, najveće dozvoljeno odstupanje između bilo koja 2 časovnika u posmatranoj grupi, neće biti veće od preciznosti grupe π_i. Da bi se preciznost zadržala u zadatim granicama, grupa časovnika mora se periodično vremenski usklađivati.

Odstupanje časovnika *k* u odnosu na referentni časovnik *z* u otkucaju *i,* predstavlja tačnost za posmatrani otkucaj *i.* Ako se posmatra maksimalno odstupanje za sve otkucaje *i* od interesa, tada se govori o tačnosti časovnika *k* [3].

4.2 Unutrašnje usklađivanje vremena

Kao što je rečeno ranije, kod unutrašnjeg usklađivanja vremena referentni časovnici su takođe i časovnici koji se međusobno usklađuju, tj. nastoji se da se vrijednosti njihovih brojača učine što bližim jedni drugima. Iako ćemo govoriti o unutrašnjem usklađivanju vremena, slični koncepti mogu da se primijene i kod spoljnog usklađivanja. Postoje 3 glavne faze koje se ciklično izvršavaju tokom procesa usklađivanja vremena [3], [41]:

- **Prikupljanje mjernih uzoraka**: prikupljanje vrijednosti vremena svih časovnika koji učestvuju u međusobnom usklađivanju. U zavisnosti od protokola, primjenjuju se 2 tehnike dobavljanja vremena od udaljenih časovnika: vremenski inicirano dobavljanje (eng. *time transmission*) i metod prozivanja (eng. *remote clock reading*). Vremenski inicirano dobavljanje podrazumijeva da svi časovnici šalju informacije o svom vremenu u tačno definisanom trenutku. Taj trenutak podrazumijeva dostizanje određene vrijednosti sopstvenog časovnika. Svi časovnici u grupi treba da šalju informacije o vremenu u istom trenutku, ali zbog međusobne razdešenosti časovnika ti trenuci nisu isti za sve članove grupe. Ako se koristi metod prozivanja, uređaj koji vrši proračun korekcije u određenom trenutku šalje zahtjev ostalim članovima grupe da pošalju stanja svojih časovnika.

- **Proračun korekcije**: nakon prikupljanja mjernih uzoraka, pristupa se proračunu vrijednosti korekcije koja će se primjenjivati na časovnike. U zavisnosti od algoritma, razmatraju se sve prikupljene vrijednosti ili samo neke od njih. Za svrhe proračuna korekcije i dovođenje vrijednosti svih časovnika da budu bliske jedne drugima koriste se tzv. konvergentne funkcije.

- **Korekcija časovnika**: primjena rezultata iz prethodne faze na korekciju vremena lokalnog časovnika. Postoje dvije vrste odstupanja časovnika u odnosu na nominalnu vrijednost njegove učestanosti: odstupanje u taktu koje se odnosi na preciznost otkucaja (eng. *jitter*), tj. varijabilno odstupanje između otkucaja časovnika koje može biti posljedica temperature, vlažnosti, pritiska, vibracije, te konstantno odstupanje od globalno propisanog vremena (eng. *clock drift*), u smislu da časovnik uvijek kasni ili je uvijek brži od referentnog (npr. u jednoj sekundi pogriješi 1 μs u odnosu na referentni časovnik). Primjer negativnog uticaja odstupanja časovnika možemo posmatrati na bežičnim uređajima, poput telefona, tableta ili bežičnog senzora. Ukoliko oscilator odstupa od globalno propisanog vremena, to direktno može da utiče na učestanost emitovanih signala, te posmatrani uređaj može da izgubi usklađenost sa drugim uređajem, ili da uđe u opseg učestanosti nekih drugih uređaja [44]. Varijabilno odstupanje otkucaja časovnika može imati za posljedicu da se

mogući predeterminisani događaji, poput izvršenja neke funkcije ili slanja nekog okvira u precizno definisanom trenutku, izvrše ranije ili kasnije nego što je to definisano po rasporedu.

4.2.1 **Konvergentne funkcije**

Usklađivanje grupe časovnika u okvirima zadate preciznosti π vrši se pomoću konvergentnih funkcija. Postoje 2 vrste ovih funkcija: konvergentne funkcije koje koriste tehnike usrednjavanja prikupljenih vremena (eng. *mean function*), te funkcije koje koriste drugačije tehnike koje nisu vezane za usrednjavanje. Konvergentne funkcije usrednjavanja nakon prikupljanja vremena časovnika, po određenom algoritmu računaju srednju vrijednost tih vremena. Jedan od primjera generičke konvergentne funkcije otporne na određeni broj grešaka ćemo analizirati na skupu uzoraka A=[2, 4, 5, 3, 4, 6, 7, 11, 6, 22, 4, 3, 5, 5]. Sortiranjem uzoraka dobijamo A_{sort}=[3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 7, 11, 22]. Za $f=3$ (broj ekstremnih uzoraka koji se zanemaruje za donje i gornje strane), posmatrani skup se svodi na $A_{(f=3)}$=[4, 4, 5, 5, 5, 6, 6]. Rezultat ove funkcije primijenjene na skup A je $A_{(f=3)}$ = (4+4+5+5+5+6+6) / 7 = 5 [3].

Prema algoritmu koji primjenjuju, neki od tipova konvergentnih funkcija usrednjavanja su sledeći [3], [42], [45]:

- Interaktivna/egocentrična konvergentna funkcija (eng. *Interactive/Egocentric Convergence Function*): posmatraćemo interaktivnu konvergentnu funkciju $f_{int}(p_i, x_1, x_2, ..., x_n)$ gdje p_i komponenta koja izvršava konvergentnu funkciju, dok su $x_j (1 \leq j \leq n)$ vrijednosti dobavljenih vremena. I interaktivna konvergentna funkcija takođe vrši usrednjavanje dobavljenih vremena ali nakon obrade uzoraka, i to na sledeći način: Bira se parametar $\omega > \pi$, te se posmatraju vrijednosti dobavljenih uzoraka x_j, $1 \leq j \leq n$. Ukoliko je $|x_j - x_i| < \omega$, parametar x_j ostaje kakav jeste. U suprotnom, vrši se zamjena parametra x_j sa x_i. Drugim riječima, razlika između bilo koja 2 uzorka ne smije biti veća od ω.

    - Primjer: posmatraćemo isti skup uzoraka A. Uzećemo vrijednost parametra ω=3. Nakon spomenute obrade uzoraka dobijamo skup A_1= [3, 3, 4, 4, 4, 5, 5, 5, 6, 6, 7]. Rezultat interaktivne konvergentne funkcije primijenjene na skup A je $A_{(\omega=3)}$ = (3+3+4+4+4+5+5+5+6+6+7) / 11 = 4.73.

- Centralna konvergentna funkcija otporna na otkaze (eng. *Fault-Tolerant Midpoint Function - FTM*): algoritam ove konvergentne funkcije se izvršava periodično, kada lokalni časovnik dostigne odgovarajuću predefinisanu vrijednost. U međuvremenu, posmatrani uređaj prikuplja uzorke vrijednosti lokalnih časovnika ostalih uređaja u mreži. Rezultat centralne konvergentne funkcije je srednja vrijednost uzorka najveće vrijednosti i uzorka najmanje vrijednosti časovnika, u skupu gdje je f uzoraka najmanjih i f uzoraka najvećih vrijednosti zanemareno.

    - Primjer: posmatraćemo isti skup uzoraka A. Kao i u prethodnom slučaju, $A_{(f=3)}$=[4, 4, 5, 5, 5, 6, 6]. Rezultat *FTM* funkcije u ovom slučaju je (4+6)/2=5.

- Diferencijalna centralna konvergentna funkcija otporna na otkaze (eng. *Differential Fault-Tolerant Midpoint Function - DFTM*): predstavlja proširenu varijantu centralne konvergentne funkcije. Računa se kao $[min(T-\Theta, x_l) + max(T+\Theta, x_u)] / 2$, gdje je T očitano vrijeme komponente p, a Θ maksimalna greška očitavanja udaljenog časovnika. Takođe vrijedi: $x_l=x_{h(f+1)}$, $x_u=x_{h(n-f)}$, $x_{h(1)} \leq x_{h(2)} \leq x_{h(n)}$, $h_p \neq h_q$, $h_p, h_q \in [1,n]$.

    - Primjer: posmatraćemo vrijednosti iz istog skupa A. Za T = 5, Θ = 1, x_l = 2, x_u = 3, vrijednost *DFTM* funkcije iznosi $[min(5-1, 2) + max(5+1, 3)] / 2 =$ $= (2 + 4) / 2 = 3$.

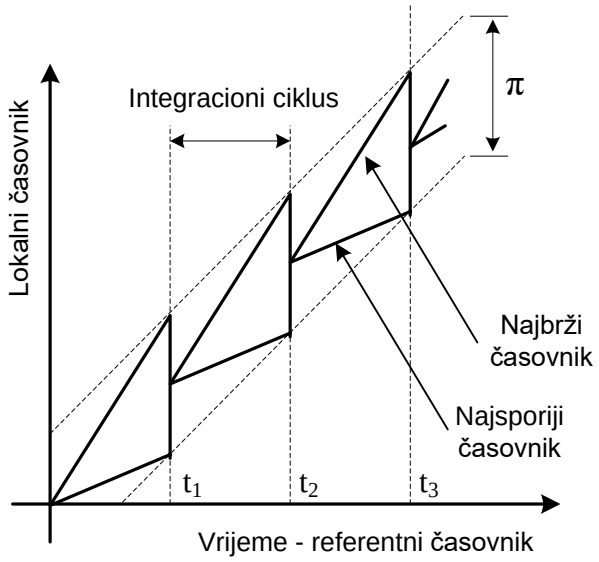
- Funkcija kliznog prozora (eng. *Sliding Window Function*): koristi prozor fiksne širine koji se „prevlači/klizi“ preko dobavljenih uzoraka vremena. Uz primjenu kliznog prozora dobija se nekoliko fiksnih prozora sa uzorcima, čiji broj direktno zavisi od broja uzoraka i širine prozora. Postoje 2 tipa konvergentnih funkcija koje se koriste u ovom pristupu, a izbor odgovarajuće funkcije zavisi od osobina uzoraka u kliznom prozoru. Prva funkcija obrađuje prvi prozor sa uzorcima i vraća srednju vrijednost vremena tih uzoraka. Druga funkcija obrađuje prozor čiji uzorci imaju najmanju varijansu, te kao rezultat vraća centralnu vrijednost uzoraka (slično kao centralna konvergentna funkcija objašnjena ranije). Glavna prednost funkcije kliznog prozora je što se njenom primjenom umanjuje uticaj uzoraka koji su generisani od strane komponente pod greškom.

    - Primjer: posmatraćemo isti skup uzoraka A=[2, 4, 5, 3, 4, 6, 7, 11, 6, 22, 4, 3, 5, 5]. Kod ovog pristupa ne vrši se sortiranje uzoraka. Širina prozora W_i koju ćemo koristiti je 10 i posmatraćemo prvi tip funkcije koja samo vraća srednju

vrijednost uzoraka u prozoru. Za skup *A* dobijamo 5 prozora: W_1 = [2, 4, 5, 3, 4, 6, 7, 11, 6, 22], W_2 = [4, 5, 3, 4, 6, 7, 11, 6, 22, 4], ..., W_5 = [4, 6, 7, 11, 6, 22, 4, 3, 5, 5]. Srednje vrijednosti uzoraka za posmatrane uzorke su: 7, 7.2, ..., 7.3.

4.2.2 Integracioni ciklusi

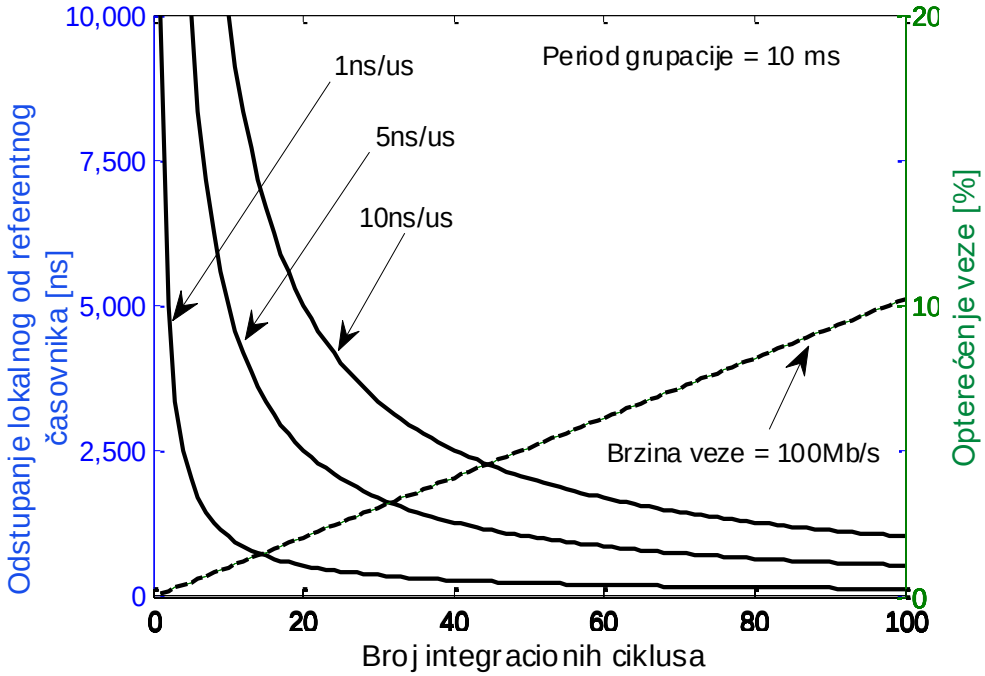
Tokom procesa usklađivanja vremena, upotreba konvergentnih funkcija vrši se periodično u vremenskim intervalima zvanim integracioni ciklusi, kao što je ilustrovano na Sl. 11. Da bi se svi časovnici u posmatranoj mreži održali u granicama zadate preciznosti π, na početku svakog integracionog ciklusa (momenti t_1, t_2, t_3, ...), brojači svih časovnika postavljaju se na istu vrijednost koja je dobijena kao rezultat proračuna konvergentne funkcije [46].



Sl. 11. Primjena konvergentnih funkcija tokom vremenskog usklađivanja [43]

Cijeli broj integracionih ciklusa je smješten u vremenski period koji se u *TTEthernet* terminologiji naziva period grupacije (eng. *cluster period*). Očigledno je da će veći broj integracionih ciklusa u okviru fiksno definisanog perioda grupacije (što znači da su integracioni ciklusi kraćeg trajanja) doprinijeti manjem međusobnom odstupanju časovnika u mreži. Na Sl. 12 [46] je prikazano kako broj integracionih ciklusa i pomjeraj lokalnog časovnika utiču na njegovo maksimalno odstupanje od idealnog časovnika u toku rada. Analizirana su 3 časovnika kod kojih je vrijednost pomjeraja po jednoj μs: 1 ns, 5 ns, i 10 ns. Odstupanje posmatranog lokalnog časovnika od vrijednosti idealnog časovnika se eksponencijalno smanjuje.

Takođe, može se primjetiti da za časovnike malog pomjeraja (1ns/1µs) značaj broja integracionih ciklusa koji je veći od 30 ne doprinosi u velikoj mjeri odstupanju od vrijednosti idealnog časovnika. Zbog toga, nema smisla pretjerano povećavati broj integracionih ciklusa, jer sa druge strane to doprinosi povećanju opterećenja veze koje je linearna funkcija direktno proporcionalna broju integracionih ciklusa. Dati grafik se odnosi na *TTEthernet* mreže kod kojih se koriste veze brzine 100 Mb/s.



Sl. 12. Devijacija vrijednosti lokalnog časovnika u zavisnosti od broja integracionih ciklusa [46]

Ukoliko nam je poznato odstupanje lokalnog časovnika σ_{local_clock}, a neophodno je da izračunamo broj potrebnih integracionih ciklusa N_{IC} za dato dopušteno odstupanje od idealnog časovnika $\sigma_{allowed} = \pi$, to možemo uraditi pomoću sledeće formule [46]:

$$N_{IC} \geq \left\lceil \frac{period\_grupacije[\mu s] \cdot \sigma_{local\_clock}[ns/\mu s]}{\sigma_{allowed}[ns]} \right\rceil \tag{9}$$

4.2.3 **Spoljno usklađivanje vremena**

Spoljašnji izvori tačnog vremena rade prema vremenskim standardima od kojih su najpoznatiji TAI (fr. *Temps Atomique International*) i UTC (eng. *Universal Time Coordinated*). TAI je ustanovljen 1967. godine od strane Bureau International de l'Heure

(BIH), te definiše sekundu kao trajanje 9192631770 perioda radijacije određene tranzicije atoma cezijuma 133. Vremenska skala definisana prema ovom standardu nema diskontinuiteta. UTC vrijeme je bazirano na astronomskim observacijama kretanja Zemlje u odnosu na Sunce. Ovo vrijeme je referenca za podešavanje velikog broja časovnika u koje se svrstavaju i npr. zidni časovnici. Postoji određena razlika između UTC i TAI vremena koja je posljedica neravnomjerne rotacije Zemlje. UTC vrijeme održava se u granicama ±0.9 sekundi od vremena dobijenog astronomskim observacijama. Ovo dovodi do uvođenja prestupne sekunde kada je to neophodno (dodavanje jedne sekunde na trenutno vrijeme). S obzirom na ovu činjenicu, za UTC vremensku skalu ne može se reći da je kontinualna [3].

Jedan od najpoznatijih sistema koji koristi spoljašnji vid usklađivanja vremena je američki sistem za navigaciju GPS (eng. *Global Positioning System*). Usklađivanje vremena se vrši prema signalima primljenim sa više satelita, koji se obrađuju u GPS prijemniku i na osnovu kojih se dobija informacija o lokaciji prijemnika, brzini i tačnom vremenu. Ovdje se primjenjuje UTC standard i postiže se tačnost bolja od 15 ns u vojnoj industriji, dok je u komercijalnim GPS prijemnicima tačnost reda 50 ns. Visoka tačnost vremenskog usklađivanja kod navigacionih sistema je neophodna, jer se lokacija prijemnika računa na osnovu vremenskih razlika signala primljenih sa više satelita. Ukoliko časovnici satelita koji emituju signale nisu kvalitetno međusobno usklađeni, nemoguće je izvršiti lokalizaciju objekta. Drugi svjetski poznati navigacioni sistemi koji koriste slične principe usklađivanja vremena su ruski navigacioni sistem GLONASS (eng. GLObal Navigation Sattelite System) i evropski navigacioni sistem Galileo. Pored satelitske komunikacije, postoje i vidovi usklađivanja vremena časovnika posredstvom radio talasa, kao što su npr. DCF77, HBG, WWV, i WWVH. Kod ovih sistema postiže se tačnost reda 50 ms [3].

4.3 Faze procesa usklađivanja vremena

Periodično međusobno usklađivanje časovnika u posmatranoj mreži predstavlja normalni režim procesa rada posmatrane komponente. Podrazumijevano je da su sve neophodne radnje obavljene da bi cjelokupan sistem mogao doći u ovo stanje. Niz radnji koje je potrebno sprovesti da bi se posmatrana komponenta u mreži dovela do normalnog režima rada predstavlja fazu pokretanja (eng. *startup*). Broj događaja u toku normalnog režima rada je višestruko veći od broja događaja prilikom režima pokretanja. Ako posmatramo npr. let aviona koji traje 5 časova i ako se usklađivanje časovnika vrši sa periodom 1 ms, tada će se

periodično vremensko usklađivanje u normalnom režimu izvršiti 18 miliona puta, dok će se faza pokretanja izvršiti samo jednom. Očigledno je da je korektno izvršeno pokretanje mreže veoma bitno, jer bez ove faze nije moguće dovesti sistem do normalnog režima rada.

Algoritmi za pokretanje se obično projektuju da budu takvi da ne zavise od trenutaka uključenja pojedinih komponenata u mreži. Drugim riječima, ne moraju sve komponente u mreži da se uključe i istom trenutku da bi se faza pokretanja uspješno završila. Ukoliko je dovoljno komponenata učestvovalo u ovoj fazi, znači da su ostvareni svi uslovi za ulazak u normalni režim rada. Za slučajeve kada se prepozna greška u normalnom režimu rada, aktivira se algoritam za ponovno pokretanje (eng. *restart*) komponente u kojem se nastoji da se izvrše sve neophodne radnje da bi se posmatrana komponenta ponovo dovela u normalni režim rada [2-3].

4.3.1 **Faza pokretanja**

Kao što je spomenuto ranije, digitalni časovnik karakterišu učestanost i stanje. Stanje predstavlja vrijednost brojača digitalnog časovnika u datom trenutku, dok je učestanost okarakterisana brzinom otkucaja, tj. brzinom promjene stanja časovnika. Ako pretpostavimo da je parametar *C* vrijednost brojača digitalnog časovnika, a *R* stvarno vrijeme, tada časovnik komponente *p* može biti predstavljen funkcijom C_p koja preslikava kontinualno stvarno vrijeme u diskretno vrijeme digitalnog časovnika [3]:

$$C_p: R \to C \tag{10}$$

Formalno, algoritam pokretanja podrazumijeva da postoji trenutak t_0 takav da je razlika bilo koja 2 lokalna časovnika u tom trenutku u granicama preciznosti π [3]:

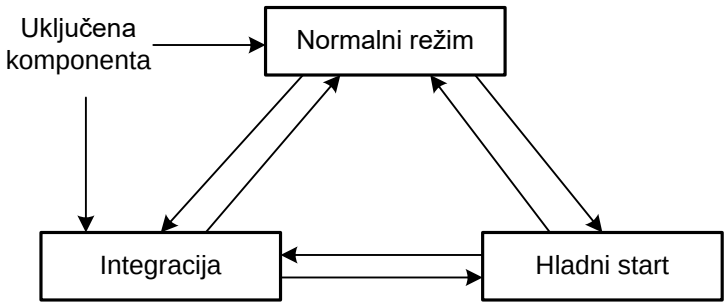
$$\exists t_0: \left|C_p(t_0) - C_q(t_0)\right| < \pi \tag{11}$$

Veoma bitan parametar posmatrane mreže je trajanje faze pokretanja i minimalna konfiguracija neophodna za uspješno pokretanje. Obermajser [3] definiše ove parametere na sledeći način: Algoritam pokretanja može da uspostavi usklađenu/determinističku komunikaciju između ispravnih komponenata u mreži (komponente koje nisu pod otkazom) u zadatim vremenskim granicama ako je najmanje minimalna konfiguracija komponenata i kanala prisutna za to vrijeme. Ako se npr. koriste konvergentne funkcije usrednjavanja,

minimalna konfiguracija govori koliko različitih okvira za usklađivanje vremena treba da primi procesor koji vrši usrednjavanje da bi se posmatrana konvergentna funkcija uopšte izvršila, te da bi se nastavio proces vremenskog usklađivanja. Prema izrazu (11), C_p i C_q su 2 komponente koje nisu pod otkazom i predstavljaju minimalnu konfiguraciju neophodnu sa pokretanje. Nakon što se završi faza pokretanja, mreža je spremna za deterministčku komunikaciju.

U zavisnosti od stanja determinističke mreže u trenutku kada je uključena nova komponenta, faza pokretanja će raditi u režimu integracije (eng. *integration*) ili hladnog starta (eng. *coldstart*). Relacije između normalnog/usklađenog režima rada, integracije i hladnog starta ilustrovane su na Sl. 13. Nakon uključenja komponente u mrežu, ona može da uđe odmah u normalni režim rada isključivo ako se koristi princip spoljašnjeg usklađivanja vremena. Ako se koristi princip unutrašnjeg usklađivanja vremena, posmatrana komponenta ne može ući odmah u normalni režim nego se to ostvaruje isključivo preko stanja integracije. U ovom stanju, komponenta ostaje jedan predefinisani vremenski period, provjerajući da li je ostatak mreže u normalnom režimu rada. Tokom ovog perioda, uključena komponenta očekuje prijem poruke od ostatka mreže sa podacima neophodnim za korekciju sopstvenog lokalnog časovnika. U slučaju da primljena poruka nije validna, ili poruka uopšte nije primljena, komponenta smatra da ostatak mreže nije u normalnom režimu rada, te prelazi u režim hladnog starta.

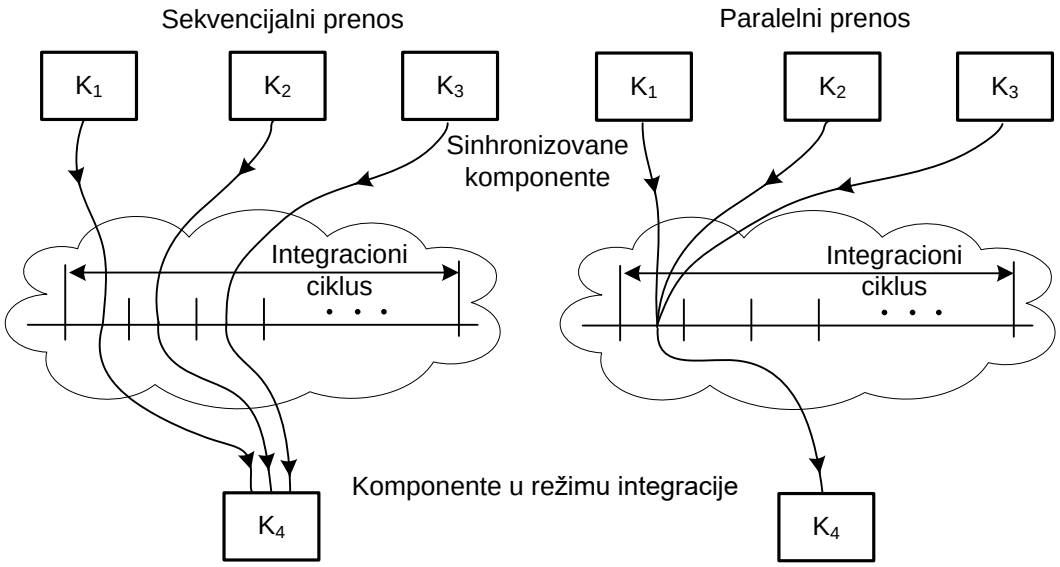
Režim hladnog starta podrazumijeva da komponenta inicira početak procesa usklađivanja vremena u cjelokupnoj mreži, slanjem odgovarajućih poruka. Ako u normalnom režimu rada dođe do pojave greške, u zavisnosti od tipa greške sistem prelazi u stanje integracije ili stanje hladnog starta. Nakon faze pokretanja podrazumijeva se da je komponenta vremenski usklađena sa ostatkom mreže [3].



Sl. 13. Pokretanje komponente [3]

4.3.1.1 **Integracija**

Režim integracije posmatrane komponente podrazumijeva da je ostatak mreže u normalnom režimu rada, te komponenta očekuje prijem integracione poruke. Nakon prijema ove poruke komponenta postaje vremenski usklađena sa ostatkom mreže. Da bi se umanjilo opterećenje veza, kod mnogih protokola poruke koje se koriste u svrhe usklađivanja vremena su ujedno i integracione poruke. U zavisnosti od korišćenog protokola, integracione poruke mogu da se prosljeđuju sekvencijalno ili paralelno, kao što je ilustrovano na Sl. 14. Za slučajeve sekvencijalnog prenosa, integracioni ciklus je dodatno podijeljen na podproreze namijenjene za prenos integracionih poruka [3].



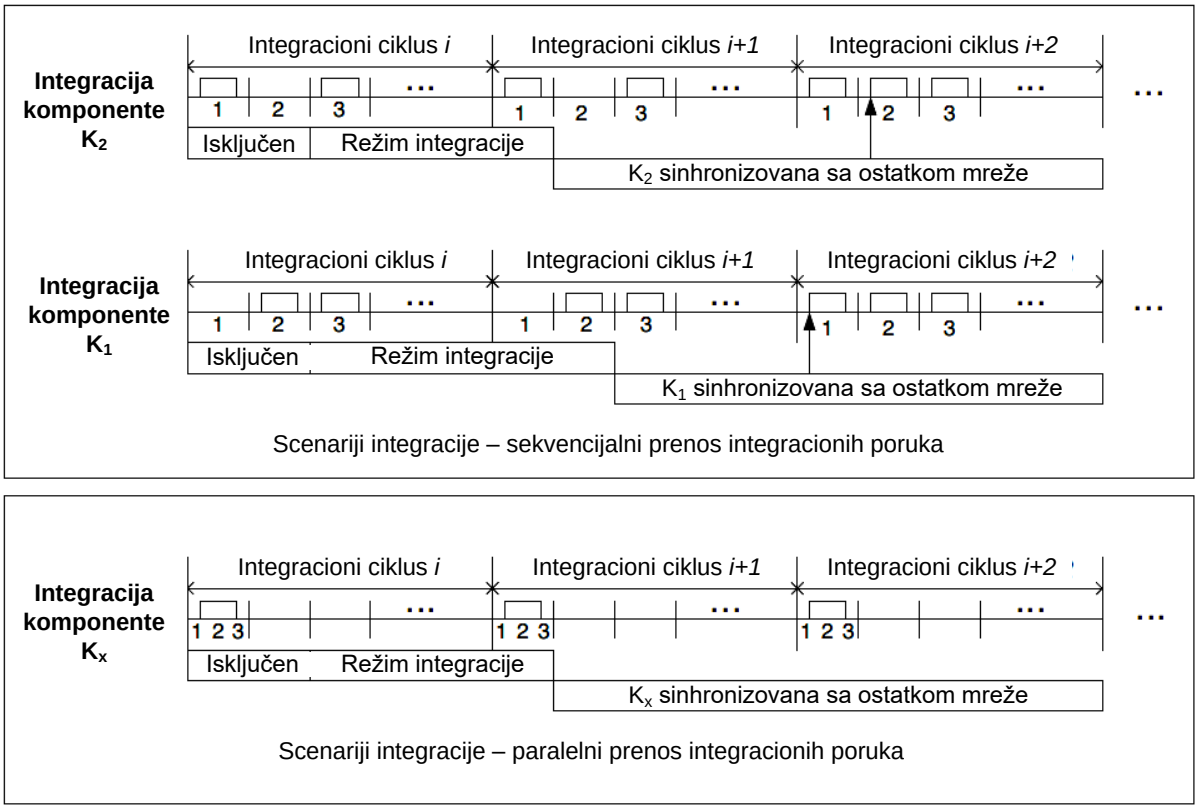
Sl. 14. Integracija komponente sa ostatkom mreže za slučajeve sekvencijalnog i paralelnog prenosa poruka [3]

Komponente K_1, K_2 i K_3 su međusobno usklađene i rade u normalnom režimu, dok komponenta K_4 treba da se uključi u normalni režim rada. U slučaju sekvencijanog prenosa integracionih poruka, komponente K_1, K_2 i K_3 ih šalju jednu za drugom, a istim redom ih prima i K_4. Kod paralelnog načina prenosa, komponente K_1, K_2 i K_3 emituju integracione poruke u istom trenutku i to se obično izvršava na početku svakog integracionog ciklusa, tj. u prvom podprorezu. Za prenos bi se teoretski mogao upotrijebiti bilo koji podprorez, ali uglavnom se to radi na početku integracionog ciklusa, što je slučaj i kod *TTEthernet* mreža. Stoga, kod paralelnog prenosa podprorezi nisu ni potrebni da budu definisani. Ovdje je neophodno postojanje centralne komponente (komutator) koja će prikupljati integracione

poruke od svih komponenata koje ih emituju, te na osnovu njih napraviti jednu novu, tzv. komprimovanu poruku, koju će proslijediti komponenti K_4. Komprimovana poruka sadrži informaciju o komponentama od čijih poruka je nastala. Ta informacija je obično smještena u *bit*-vektoru zvanom vektor pripadnosti (eng. *membership vector*), gdje svaki postavljeni *bit* predstavlja komponentu koja je učestvovala u izradi komprimovane poruke.

Prednost paralelnog načina prenosa u odnosu na sekvencijalni ogleda se u činjenici da je u posmatranom vremenskom intervalu veća iskoristivost veza, jer je dovoljno koristiti samo jedan podprorez da bi se prenijele informacije vezane za usklađivanje časovnika. Ostali prostor na vremenskoj skali može biti upotrijebljen za prenos poruka drugih klasa saobraćaja. Međutim, kao što smo spomenuli ranije, za ovaj pristup neophodna je upotreba komutatora koja je komplikovanija komponenta od magistrale ili čvorišta (eng. *hub*). Pristupi koji koriste magistralu ili čvorište mogu se iskoristiti za sekvencijalni prenos poruka, što znači da je cijena upotrebe ovog pristupa niža. S druge strane, veza je zauzeta duži vremenski period kod sekvencijalnog prenosa, tako da se smanjuje njena dostupnost za poruke drugih klasa saobraćaja.

Da bi komponenta ušla u normalni režim rada iz stanja integracije, mora da primi dovoljan broj integracionih poruka od ostalih komponenata u mreži, ili da primi komprimovani integracioni okvir (eng. *frame*) sa dovoljno postavljenih *bit*-a u vektoru pripadnosti, kao što je ilustrovano na Sl. 15. Ako posmatramo slučaj sekvencijalnog prenosa i ako je komponenta K_2 bila isključena za vrijeme prijema integracione poruke od komponente K_1, ona ostaje i dalje u stanju integracije sve dok ne primi i poruku od K_1, što znači da normalni režim rada K_2 nije uspostavljen u integracionom ciklusu *i*. Tek u integracionom ciklusu *i+1,* kada su primljene sve neophodne integracione poruke (od komponenata K_1 i K_3), komponenta K_2 ulazi u normalni režim rada i postaje usklađena sa ostatkom mreže. U integracionom ciklusu *i+2*, komponenta K_2 počinje emitovati svoje integracione poruke u dodijeljenom podprorezu. Situacija je analogna i za slučaj integracije komponente K_1, koja takođe ulazi u normalni režim rada u integracionom ciklusu *i+1,* a emituje sopstvene integracione poruke tek u integracionom ciklusu *i+2*. Za slučaj paralelnog prenosa, komponenta K_x je usklađena sa ostatkom mreže odmah nakon prijema prve korektne komprimovanog integracione poruke [3].



Sl. 15. Uspostavljanje vremena usklađenog sa ostatkom mreže kada je primijenjen proces integracije [3]

Čim primi integracioni okvir, komponenta može da ga iskoristi za korekciju sopstvenog časovnika. U trenutku prijema integracione poruke, prijemnik na osnovu podataka iz poruke zna kada je poruka poslana od strane predajnika na osnovu stanja njegovog lokalnog časovnika C_S, te koliko iznosi kašnjenje kanala (eng. *latency*), jer je ovo statički parametar. Na osnovu spomenuta dva podatka, prijemnik može da koriguje sopstveni lokalni časovnik C_R [3].

4.3.1.2 **Hladni start**

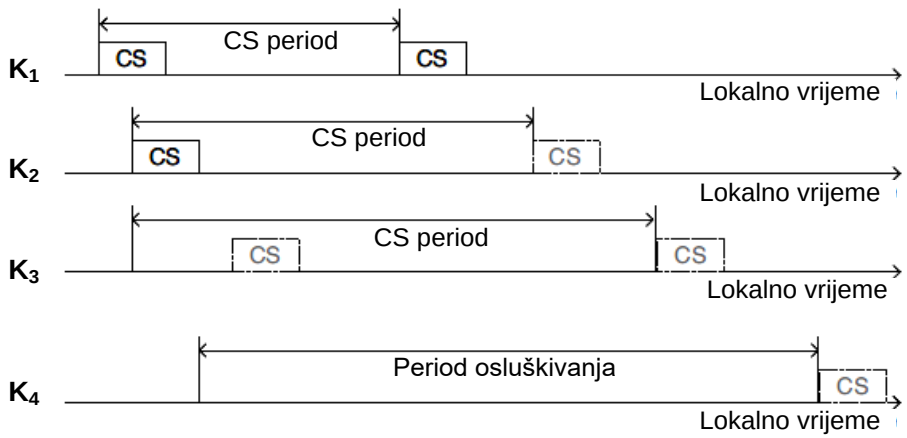
Ukoliko je proces integracije bio neuspješan (integraciona poruka nije validna ili nije uopšte primljena), komponenta ulazi u režim hladnog starta. U ovom režimu posmatrana komponenta smatra da ostatak mreže nije u normalnom režimu rada, te inicira proces vremenskog usklađivanja na jedan od sledećih načina [3]:

- Generisanjem šuma: ovo može biti bilo koji signal različit od mirnog stanja kanala.

- Nesemantičke *Coldstart* poruke: poruke koje ne nose nikakve značajne podatke osim tipa poruke – *Coldstart* poruka.
- Semantičke *Coldstart* poruke: poruke koje osim tipa mogu sadržavati i neke druge podatke.

Da bi se mogao uspostaviti normalni režim rada, u režimu hladnog starta mora da učestvuje više od jedne komponente. S obzirom da se radi o asinhronom prenosu *Coldstart* (skraćeno - CS) poruka u režimu hladnog starta, moguća je pojava sudara (eng. *colision*) ovih poruka na magistrali. Sudar može biti fizički ili logički.

Fizički sudar javlja se u slučajevima kada se koriste *half-duplex* kanali u topologijama tipa magistrala, tj. ako dvije ili više komponenata pristupaju kanalu u isto vrijeme. Pored jednostavnosti, proces hladnog starta iniciran generisanjem šuma je neosjetljiv na sudar, tj. bilo kakav signal različit od mirnog stanja na kanalu smatra se iniciranjem hladnog starta. S druge strane, kod ovog pristupa teško se prepoznaje da li se radi o validnom signalu za iniciranje hladnog starta, ili je u pitanju šum emitovan od strane komponente pod otkazom. Ukoliko se iniciranje hladnog starta vrši putem CS poruka, može doći i do nemogućnosti uspostavljanja normalnog režima rada kada se dogodi sudar između poruka. Ovdje se ogleda glavni nedostatak oba pristupa sa *Coldstart* porukama, jer fizički sudar dovodi do gubitka podataka sadržanih u porukama. Ako se radi o semantičkim porukama, gubi se veći broj značajnih informacija nego kod nesemantičkih poruka, što ih čini osjetljivijim u odnosu na nesemantičke poruke. Međutim, semantičke poruke imaju mogućnost da sadrže više informacija od nesemantičkih, poput vektora pripadnosti i slično, što ih čini lakšim za prepoznavanje. Da bi se izbjegli negativni efekti sudara CS poruka koristi se tzv. algoritam razrješenja sudara ilustrovan na Sl. 16 [3].



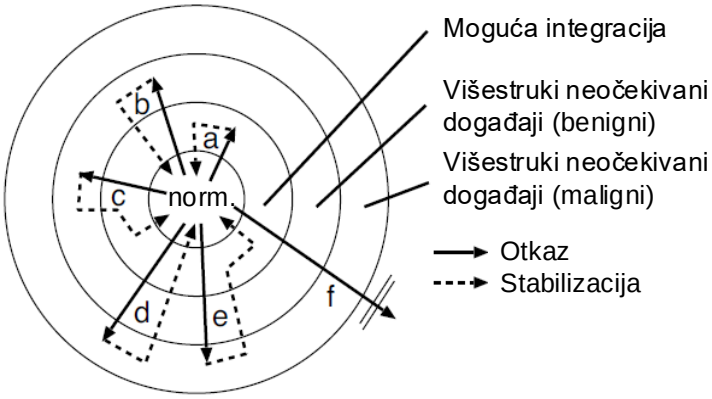
Sl. 16. Algoritam razrješenja sudara [3]

Algoritam razrješenja sudara podrazumijeva da postoji fiksno definisana gornja vremenska granica za koju se smatra da će barem jedna komponenta poslati svoju CS poruku bez sudara. Osnovna osobina ovog algoritma je da su CS periodi (eng. *coldstart timeout*) između CS poruka različiti za različite komponente. U našem primjeru komponenta K_1 ima najkraći, dok komponenta K_4 ima najduži CS period. Komponente K_1 i K_2 šalju svoje CS poruke u približno istom trenutku, što rezultuje njihovim sudarom na kanalu. S obzirom da K_1 ima najkraći period između CS poruka, on uspijeva prvi poslati poruku bez sudara. Nakon ovog trenutka, sve ostale komponente prestaju sa narednim pokušajima slanja svojih CS poruka. Komponenta K_3 je mogla poslati svoju CS poruku, ali nije jer je prepoznala sudar na magistrali te je poništila svoj tajmer koji kontroliše trenutke slanja poruka, dok je K_4 uključena u trenutku sudara, te je takođe odložila slanje [3].

Logički sudari javljaju se u topologijama tipa zvijezda kod kojih se koriste redundantne konfiguracije i semantičke poruke. Svaka instanca redundantne konfiguracije garantuje odsustvo fizičkog sudarana svojoj strani. Međutim, pošto su ove instance međusobno nezavisne, moguće je da prijemnik dobije poruke različitih predajnika i različitih identifikatora. Spomenuta pojava dešava se u slučajevima kada različiti predajnici šalju poruke na različite podskupove replika, umjesto na kompletan skup replika. I ovi problemi se mogu prevazići tehnikama sličnim algoritmu razrješenja fizičkog sudara [3]. Drugim riječima, primjena različitih perioda CS poruka mogla bi se iskoristiti i za razrješenje logičkog sudara.

4.3.2 **Ponovno pokretanje**

Ukoliko se javi greška u toku normalnog režima rada, posmatrana komponenta gubi vremensku usklađenost sa ostatkom mreže i prelazi u režim ponovnog pokretanja. Režim ponovnog pokretanja podrazumijeva tranziciju iz normalnog režima rada u režim integracije ili hladnog starta. Udar munje u avion je jedan primjer događaja koji može da dovede do devijacije normalnog ponašanja mreže. Ako broj komponenata pod otkazom izlazi iz okvira hipoteze otkaza, dolazi do gubitka vremenske usklađenosti u mreži, a samim tim i do nemogućnosti nastavka determinističke komunikacije. Zadatak režima ponovnog pokretanja je da preduzme odgovarajuće radnje u cilju ponovnog uspostavljanja normalnog režima rada. Koncept ponovnog pokretanja je dat na Sl. 17 [3].



Sl. 17. Stabilizacija u slučajevima malignih i benignih otkaza [35]

Skup stanja pokrivenih hipotezom otkaza predstavljen je centralnim krugom označenim sa norm (normalni režim), dok su ostala stanja predstavljena ostalim koncentričnim krugovima. Stepen otkaza ilustrovan je udaljenošću posmatranog koncentričnog kruga od njegovog centra. Za otkaz nižeg stepena kao što je otkaz *a*, komponenta prelazi u režim integracije, te se dalje traži način da se komponenta vrati u normalni režim rada (isprekidana linija na slici). Za neočekivane događaje (eng. *clique*) višeg stepena kao što su *b, c, d, e,* komponenta prelazi u režim hladnog starta. Za ove slučajeve obično je potrebno više vremena da se komponenta ponovo dovede u normalni režim rada, nego kada je bila prešla u režim integracije [3]. S obzirom da mrežne komponente koje učestvuju u procesu usklađivanja časovnika uglavnom funkcionišu prema predefinisanoj mašini stanja, režim ponovnog pokretanja podrazumijeva prelaz u početno stanje, nakon kojeg se primjenjuje niz koraka da bi se komponenta ponovo dovela u isto stanje u kojem se usklađivanje časovnika normalno obavlja. Drugim riječima, komponenta mijenja svoja stanja od početnog, pa sve dok ne dođe u normalno stanje rada.

4.3.3 **Normalni režim rada**

U normalnom režimu rada komponenta periodično razmjenjuje integracione poruke sa ostatkom mreže, te shodno dobijenim informacijama koriguje stanje svog lokalnog časovnika. Kao što smo spomenuli ranije, integracione poruke razmjenjuju se periodično u vremenskim intervalima zvanim integracionim ciklusima.

Kontrola primljenih poruka vrši se mehanizmima prepoznavanja (eng. *detection*) neočekivanih događaja (eng. *Clique Detection* - CD). Spomenuta kontrola se svodi na

provjeru poruka u odnosu na prozor prijema (Sl. 8), te se može podijeliti na 3 različite grupe [3]:

- Sinhroni CD događaj: Ako je komprimovana integraciona poruka primljena unutar prozora prijema, ali je broj postavljenih *bit*-a u vektoru pripadnosti nedovoljan da bi ga komponenta uzela u razmatranje.
- Asinhroni CD događaj: Ako je komprimovana integraciona poruka primljena izvan prozora prijema.
- Relativni CD događaj: Ako je primljeno više integracionih poruka izvan prozora prijema, nego onih integracionih poruka unutar prozora prijema.

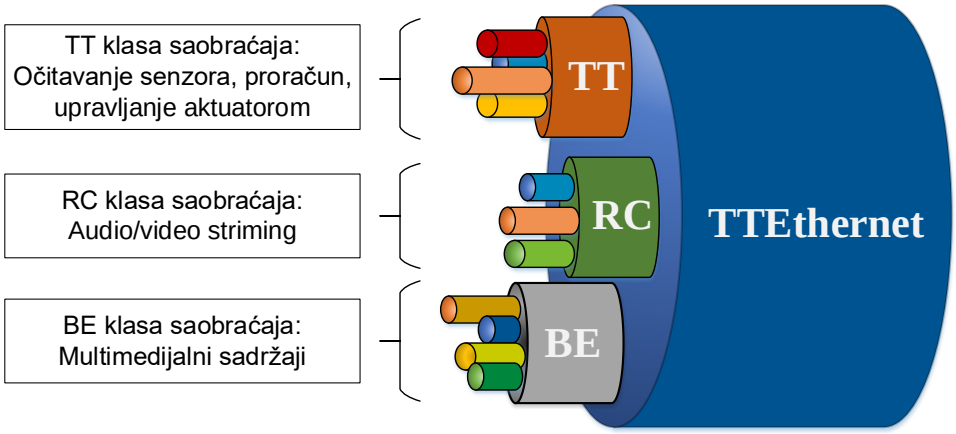
U slučaju da je prepoznat neki CD događaj, posmatrana komponenta smatra da dobijene informacije u integracionim porukama nisu validne ili su nepotpune, te primjenjuje mehanizam ponovnog pokretanja.

5 TTEthernet mreže

TTEthernet mreže svrstavaju se u grupu deterministčkih mreža koje svoju primjenu nalaze uglavnom u avionskoj i automobilskoj industriji. Iako je *Ethernet* protokol i dalje dominantan u sferi standardnih računarskih mreža, u bezbjednosno-kritičnim aplikacijama gdje su neophodne garancije u pogledu kašnjenja i otpornosti na otkaze, *Ethernet* protokol ne daje odgovarajuće rezultate. *TTEthernet* protokol je razvijen u austrijskoj firmi TTTech Computertechnik AG u saradnji sa američkom firmom Honeywell. Standardizovan je od strane SAE kao AS6802 standard gdje su definisani koncepti komunikacije u *TTEthernet* mreži kao i postupak usklađivanja vremena. Iako je projektovan prema standardima avionske industrije, *TTEthernet* se takođe može primjenjivati i u ostalim bezbjednosno-kritičnim aplikacijama kao što je npr. automobilska industrija. Pored kompatibilnosti sa *Ethernet* standardom, *TTEthernet* je kompatibilan i sa ARINC 664p7 standardom koji je takođe veoma važan u avionskoj industriji. Komponente u *TTEthernet* mreži zahtijevaju *Full-Duplex* veze za međusobno povezivanje. Redundantne konfiguracije su takođe podržane, te se neophodne poruke šalju simultano ne sve replike u mreži. U ovom poglavlju ćemo detaljnije opisati spomenute klase saobraćaja u *TTEthernet* mreži, sa posebnim osvrtom na PCF klasu saobraćaja, jer je ta klasa predmet analize ovog rada [3], [32].

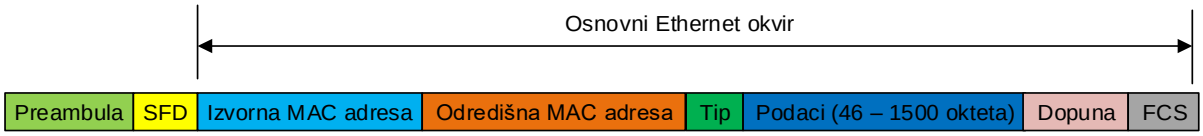
5.1 Klase saobraćaja u TTEthernet mreži

TTEthernet protokol može se primjenjivati u mrežama sa više prioriteta jer podržava kombinovanje deterministčkog, sporadičnog, kao i standardnog *Ethernet* saobraćaja. U *TTEthernet* terminologiji, deterministčka klasa saobraćaja je TT (eng. *Time Triggered*), sporadična je RC (eng. *Rate Constrained*), dok standardni *Ethernet* saobraćaj pripada BE (eng. *Best Effort*) klasi saobraćaja koja se često naziva i ET (eng. *Event Triggered*) - Sl. 18. Pored spomenutih klasa saobraćaja postoji i posebna klasa saobraćaja čija je uloga samo uspostavljanje i održavanje usklađenog vremena u mreži – PCF (eng. *Protocol Control Frames*) klasa saobraćaja [1], [41], [47].



Sl. 18. Klase saobraćaja u *TTEthernet* mreži [43]

Za prenos podataka koriste se standardni *Ethernet* okviri (Sl. 19). Polje Tip u *Ethernet* okviru ima vrijednost 0x88D7 ako se radi o TT okvirima, dok okviri koji pripadaju PCF klasi saobraćaja imaju vrijednost Tip polja 0x891D [34], [48].



Sl. 19. *Ethernet* okvir [48]

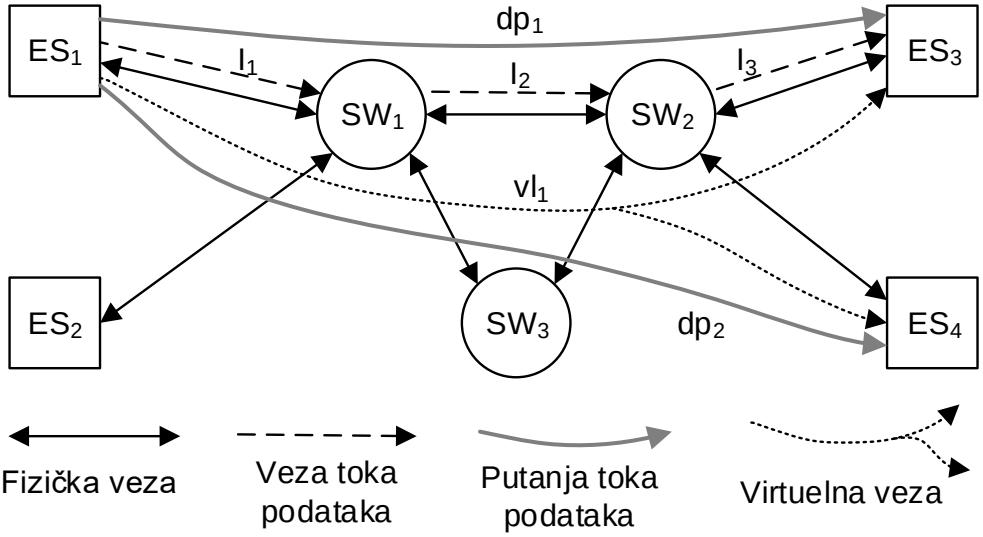
5.1.1 TT klasa saobraćaja

TT klasa saobraćaja je striktno deterministička, tj. momenti slanja paketa su precizno definisani, kao i vremenski intervali kada se može očekivati njihov prijem na prijemnoj strani. Za odvijanje TT komunikacije neophodno je da su sve neophodne mrežne komponente međusobno vremenski usklađene. Momenti slanja TT okvira su statički definisani u tabelama rasporeda (eng. *schedule tables*) svakog krajnjeg uređaja, te ovi okviri nisu podložni sudarima. Unaprijed poznati momenti razmjene TT okvira, kao i zagarantovane vrijednosti kašnjenja predstavljaju neke od najbitnijih osobina TT klase saobraćaja. Nakon PCF klase saobraćaja, TT okviri imaju najviši prioritet u *TTEthernet* mrežama [49].

Ukoliko je potrebno dodati novu komponentu u mrežu, neophodno je prilagoditi raspored slanja okvira u tabelama rasporeda, što znači da *TTEthernet* mreža predstavlja sistem zatvorenog tipa. Garancije u pogledu kašnjenja i pouzdanosti najčešće mogu biti ispoštovane samo u sistemima zatvorenog tipa. S druge strane, kod sistema otvorenog tipa kao što je npr. *Internet*, kritičan moment nastupa kada više klijenata zahtijeva servis od

servera. U ovakvim sistemima, klijenti se "takmiče" za servis na serveru. Kod sistema otvorenog tipa spomenute garancije ne mogu biti ispoštovane [1], [49].

Svaka *TTEthernet* mreža sastoji se od krajnjih uređaja i komutatora međusobno povezanih dvosmjernim fizičkim vezama. Imajući ovo u vidu, *TTEthernet* mreža može se predstaviti kao neusmjereni graf $\Gamma(\delta, \lambda)$, gdje je $\delta = \{SW_1, SW_2, SW_3, ..., SW_m\} \cup \{ES_1, ES_2, ES_3, ..., ES_n\}$, dok parametar λ predstavlja fizičke veze, tj. $\lambda = \{l_1, l_2, l_3, ..., l_k\}$. Kao što je ilustrovano na Sl. 20. pored komponenata povezanih fizičkim vezama, definišu se i logičke veze (eng. *virtual link*) [50].



Sl. 20. *TTEthernet* mreža sa prikazanim komponentama, fizičkim i logičkim vezama [20]

U *TTEthernet* mrežama podaci se prenose preko virtuelnih veza vl_i koje su definisane u standardu ARINC 664p7. Virtuelne veze su jednosmjerne i pomoću njih se prenose poruke od jednog predajnika prema jednom ili više prijemnika. Konfiguracija virtuelne veze (pošiljalac, prijemnici, dužina okvira) je smještena u komutatoru. Identifikator (ID) virtuelne veze se upisuje u odredišnu MAC adresu okvira na strani predajnika. Na osnovu ovog identifikatora, komutator zna na koje priključke treba da proslijedi okvir koji je primio od krajnjeg uređaja. Ukoliko se identifikator primljenog okvira ne nalazi u konfiguraciji komutatora, okvir biva odbačen [51].

Svaka virtuelna veza ima rezervisan propusni opseg (eng. *bandwidth*) koji nije narušen drugim virtuelnim vezama. Unaprijed je poznato koliko svaka virtuelna veza doprinosi opterećenju posmatrane fizičke veze. Virtuelna veza se sastoji od jednog ili više tokova

podataka (eng. *dataflow path - dp*) koji definišu vezu između jednog predajnika i jednog prijemnika. Tok podataka definiše usmjeravanje virtuelne veze na sledeći način:

$$\rho_{\mathrm{VL}}(vl_i) = \{\forall dp_j \in DP | dp_j \in vl_i\} \tag{12}$$

što znači da se usmjeravanje vl_1 u našem slučaju može predstaviti kao $\rho_{VL}(vl_1)=\{dp_1, dp_2\}$. Veze toka podataka se mogu predstaviti kao $l_i=\{\delta_j \rightarrow \delta_k | \delta_j, \delta_k \in \delta\}$ [34], [43], [50].

S obzirom da se sastoji od više tokova podataka, virtuelna veza se može predstaviti i na sledeći način:

$$vl = \cup_{i=1}^{n} dp_i \tag{13}$$

Izrada rasporeda slanja TT okvira je zadatak raspoređivača, čiji se rezultati smještaju u spomenute tabele rasporeda. Momenti slanja TT okvira su raspoređeni unutar perioda fiksnog trajanja (Sl. 7) zvanog period grupacije. Period grupacije se ciklički ponavlja tokom cjelokupnog procesa determinističke komunikacije, a samim tim se ponavlja i definisani raspored slanja TT okvira. To znači da je u svakom trenutku poznato kada će biti proslijeđen odgovarajući TT okvir. U odnosu na ovu osobinu, raspoređivač koji se koristi za izradu rasporeda okvira determinističke klase saobraćaja u *TTEthernet* mrežama svrstava se u grupu cikličkih raspoređivača.

Osnovni parametri okvira koji pripadaju jednoj virtuelnoj vezi između komponenata v_l i v_k su njihov period, pomjeraj prvog okvira u odnosu na početak perioda grupacije (eng. *offset*), te dužina (ili trajanje) okvira:

$$f_i^{[v_k,v_l]} = \{f_i.period, f_i^{[v_k,v_l]}.pomjeraj, f_i.dužina\} \tag{14}$$

Vremenska skala u periodu grupacije je podijeljena u proreze gdje se smještaju okviri. Ukoliko se obavlja samo TT komunikacija, minimalna dužina proreza je:

$$L_{min} = \max(okvir_{dužina}^{TT}) + \pi \tag{15}$$

Iz izraza (15) se vidi da prorez mora biti proširen za vrijednost preciznosti π. Dužina ovog proreza je u stvari period posmatrane veze $f_i.period$. Za slučajeve kada se pored TT klase saobraćaja prenose i okviri nižeg prioriteta, poput RC okvira, vremenski prorez se mora proširiti na sledeći način:

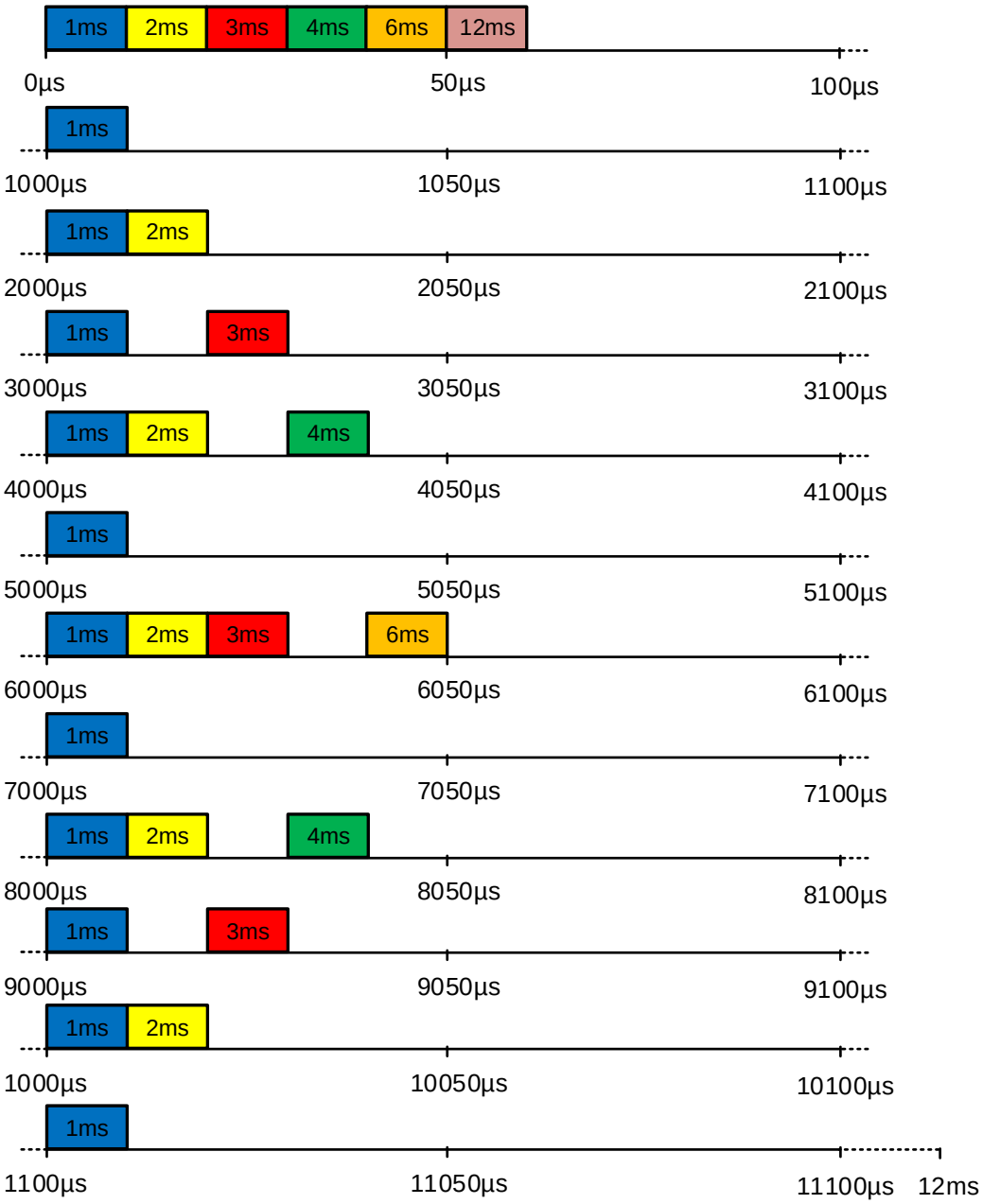
$$L_{min} = max\{max(okvir_{length}^{TT} + \pi), max(okvir_{length}^{TT}) + max(okvir_{length}^{RC})\} \quad (16)$$

Prema izrazu (16), dodijeljeni vremenski prorez treba da bude dovoljan da prenese TT okvir maksimalne veličine, kao i jedan RC okvir maksimalne veličine.

Uspješno generisan raspored slanja garantuje TT komunikaciju bez sudara između okvira koji pripadaju različitim virtuelnim vezama. Da bi se to postiglo, period grupacije se bira takav da njegova vrijednost bude jednaka najmanjem zajedničkom sadržiocu (eng. *Least Common Multiple - LCM*) perioda svih virtuelnih veza u mreži. Primjenom LCM metoda, raspoređivač nikada neće dodijeliti isti trenutak slanja za dva ili više TT okvira. Drugim riječima, dobijeni raspored slanja okvira mora da zadovolji sledeću formulaciju:

$$\forall f_i, f_i \in F, \forall a, b \in \left[0..\left(\frac{LCM(F.period)}{f_i.period} - 1\right)\right]:$$
$$\left(\left(f_i \neq f_j\right) \wedge \exists f_i^{[v_k,v_l]} \wedge f_j^{[v_k,v_l]}\right) \Rightarrow (a \times f_i.period) +$$
$$f_i^{[v_k,v_l]}.pomjeraj \neq \left(b \times f_j.period\right) + f_j^{[v_k,v_l]}.pomjeraj \quad (17)$$

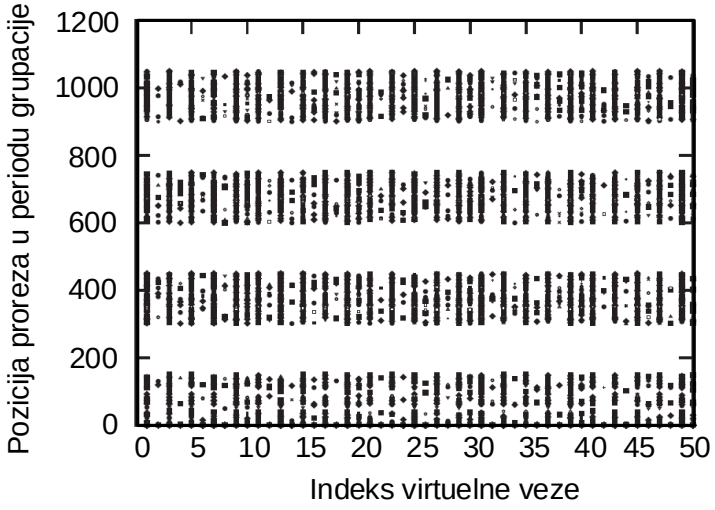
Takođe, pri izradi rasporeda slanja TT okvira mora se voditi i računa o opterećenju veze. Primjer jednog rasporeda TT okvira za virtuelne veze perioda 1 ms, 2 ms, 3 ms, 4 ms, 6 ms, 12 ms je ilustrovan na Sl. 21.



Sl. 21. Primjer jednog rasporeda slanja TT okvira [52]

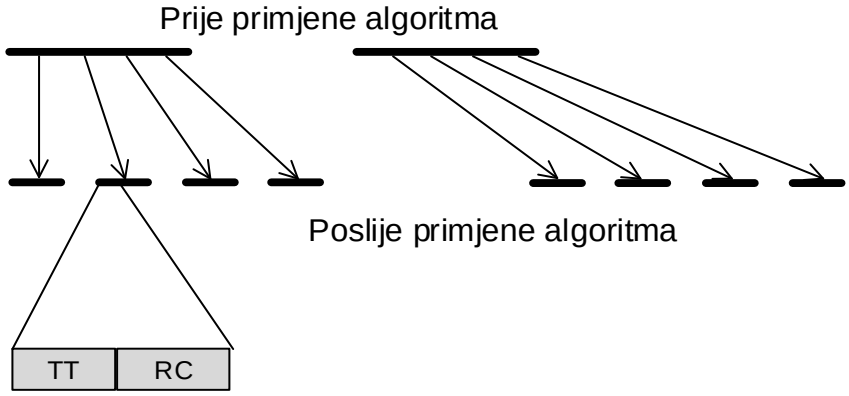
Za prikazani raspored slanja okvira izabran je period grupacije 12 ms, jer je to vrijednost LCM perioda svih neophodnih virtuelnih veza. Ukoliko se obavlja samo TT komunikacija, vremenski prorezi označeni na Sl. 21 ujedno predstavljaju TT okvire trajanja 10 µs. Opterećenju veze najviše doprinosi virtuelna veza kod koje se okviri prenose sa učestanošću 1 ms. Predstavljani raspored slanja okvira se odnosi na cjelokupnu mrežu, te posmatrane virtuelne veze mogu biti dodijeljene različitim krajnjim uređajima. Sav prostor unutar perioda grupacije koji nije zauzet od strane TT ili RC klase saobraćaja može da bude iskorišćen za slanje BE okvira koji su najnižeg prioriteta [18], [52].

Ako analiziramo Sl. 21, vidimo da prorezi za kritični saobraćaj (TT/RC) nisu ravnomjerno raspoređeni. Za slučajeve kada pored TT/RC klase saobraćaja treba da se prenose i BE okviri, oni ne bi mogli da se prenose sa približno jednakim međusobnim rastojanjem. Opisani problem je mnogo izraženiji ako se koriste veze koje su više opterećene kritičnim saobraćajem nego što je to na Sl. 21. Primjer perioda grupacije koji posjeduje 1200 mogućih proreza, te 50 virtuelnih veza sa ukupno 8000 instanci okvira dat je na Sl. 22.



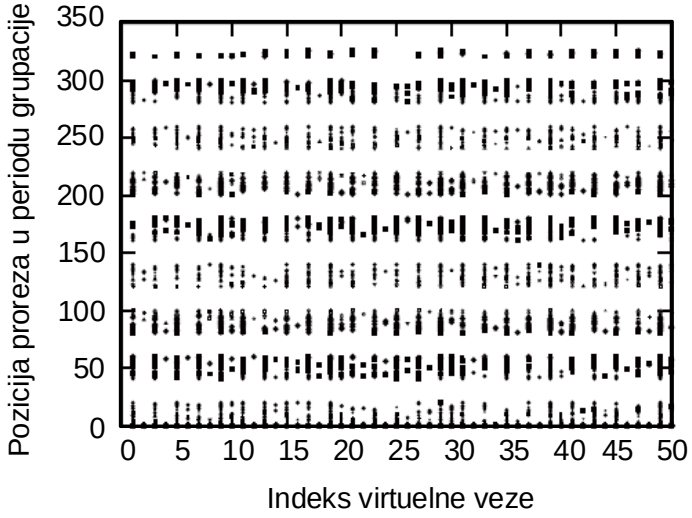
Sl. 22. Primjer perioda grupacije sa 8000 instanci okvira [18]

Sa Sl. 22 vidljivo je da je nemoguće slati BE okvire ravnomjerno u slobodnim prorezima. Iz ovog razloga pristupa se korišćenju algoritma prorjeđivanja rasporeda (eng. *schedule porosity*) [18], koji nastoji da prilagodi postojeći raspored u cilju oslobađanja određenih proreza za svrhe ravnomjernog slanja BE okvira. Koncept ovog algoritma je dat na Sl. 23.



Sl. 23. Koncept algoritma prorjeđivanja [18]

Nakon primjene algoritma prorjeđivanja na raspored sa Sl. 22 dobija se rezultat prikazan na Sl. 24. Vidljivo je da se sada BE okviri mogu slati znatno ravnomjernije nego što je to bilo prije primjene algoritma prorjeđivanja [18].



Sl. 24. Raspored okvira u periodu grupacije nakon primjene algoritma prorjeđivanja [18]

Kod složenih topologija mreža gdje postoji više mogućih putanja od predajnika do prijemnika, Tames-Selisin (*Domitian Tamas-Selicean*) [14] predlaže primjenu DOTTS (eng. *Design Optimization of TTEthernet-based Systems*) tehnike. Optimizacija pomoću DOTTS tehnike podrazumijeva usmjeravanje poruka determinističkog saobraćaja na takav način, da se preko određenih putanja (komutatora) smanjuje *BWU* za deterministički saobraćaj, a samim tim povećava slobodan prostor na tim putanjama za poruke naletnog tipa [14].

5.1.2 **RC klasa saobraćaja**

RC klasu saobraćaja karakteriše znatno manji nivo determinizma u odnosu na TT klasu saobraćaja. Ova klasa saobraćaja ne zahtijeva da komponente u mreži budu međusobno vremenski usklađene. Trenuci slanja RC okvira nisu precizno određeni, ali je poznata njihova učestanost slanja, kao i ograničeno kašnjenje. Na osnovu prethodno navedenih parametara, RC okvir koji pripada jednoj virtuelnoj vezi između komponenata v_l i v_k može se opisati kao [53]:

$$f_i^{[v_k,v_l]} = \{f_i.period, f_i.dužina\} \tag{18}$$

Za razliku od izraza (15), pomjeraj okvira u odnosu na početak perioda grupacije ne figuriše u izrazu (18), jer tačni momenti slanja RC okvira nisu određeni. RC klasa se takođe

prenosi po principu virtuelnih veza i to isključivo ako posmatrana fizička veza nije zauzeta od strane TT ili PCF klase saobraćaja u momentu slanja RC okvira. Nije neophodno da se RC okviri prenose striktno periodično, ali postoji minimalno rastojanje između RC okvira unutar kojeg se ne prenose RC okviri koji pripadaju istoj virtuelnoj vezi. Ovo rastojanje naziva se kao BAG (eng. *Bandwidth Allocation Gap*), koje po ARINC 664p7 standardu može imati vrijednosti 2^i ($0 \leq i \leq 7$). Vrijednost BAG parametra se bira tako da je zagarantovan slobodan propusni opseg na virtuelnoj vezi za prenos posmatranog RC okvira. Drugim riječima, krajnji uređaj će obezbjediti da će jedan BAG interval sadržavati najviše jedan RC okvir [49], [53].

5.1.3 BE klasa saobraćaja

BE klasa saobraćaja predstavlja standardni *Ethernet* saobraćaj i ima najniži prioritet u *TTEthernet* mreži. Okviri BE klase se prenose na uobičajeni način bez upotrebe virtuelnih veza. Transmisija BE okvira je moguća isključivo ako posmatrana veza nije zauzet od strane TT/RC/PCF saobraćaja u posmatranom trenutku. Za razliku od spomenutih klasa saobraćaja, za BE okvire ne postoje nikakve garancije u pogledu pouzdanosti i ograničenog kašnjenja. U slučaju da je zakazan prenos TT/RC/PCF okvira u toku prenosa BE okvira, prekida se slanje BE okvira da bi se obezbjedio nesmetan prenos okvira višeg prioriteta [1], [3], [49] .

Kod kompleksnih sistema u kojim je neophodno korišćenje mreža sa više prioriteta poput savremenih automobila, koji posjeduju i preko 130 ECU jedinica (eng. *Electronic Control Unit*), posebna pažnja se posvećuje izboru klase saobraćaja za određene operacije. TT klasa saobraćaja se koristi za operacije gdje ne smije doći do gubljenja paketa te moraju biti ispoštovana stroga vremenska ograničenja (ispravan rad motora, kočioni sistem i slično), dok se za ostale operacije koriste RC i BE tipovi saobraćaja. Kod inteligentnih automobila koji posjeduju video kamere, iako je dopušteno gubljenje određenog broja slika u video signalu, ne bi trebalo koristiti standardnu BE klasu saobraćaja za prenos ovih signala. U ovom slučaju događa se da video signali prikupljeni sa svih kamera ne mogu biti međusobno usklađeni zbog mogućih prevelikih kašnjenja BE paketa. Zbog ove činjenice, najbolje je koristiti RC klasu saobraćaja za ovu i slične operacije, jer iako određeni broj ovih paketa može biti odbačen u toku komunikacije kod *TTEthernet* mreža, ipak postoje određene vremenske garancije u pogledu kašnjenja RC paketa, ali koje su manje stroge u odnosu na garancije kašnjenja TT klase saobraćaja [43], [54-55].

5.1.4 PCF klasa saobraćaja

Treba napomenuti da je raspored slanja okvira dat na Sl. 21 u stvari pojednostavljeni slučaj. U stvarnosti, potrebno je uzeti u obzir i PCF okvire bez čijeg postojanja nije moguće uopšte ostvariti TT komunikaciju. PCF okviri imaju najviši prioritet u *TTEthernet* mreži i od njihove korektne razmjene zavisi da li će časovnici mrežnih komponenata uopšte moći da budu međusobno vremenski usklađeni. To znači da u periodu grupacije uvijek treba da se rezerviše i prostor za PCF okvire, ukoliko se mreža podešava za TT komunikaciju. Ako se koristi samo RC ili BE klasa saobraćaja, nije potrebno obavljati aktivnosti usklađivanja vremena [41].

Postoje 3 tipa PCF okvira, koji će detaljnije biti opisani kasnije u ovoj tezi [2], [46]:

- CS (eng. *Coldstart*)
- CA (eng. *Coldstart Acknowledge*)
- IN (eng. *Integration*)

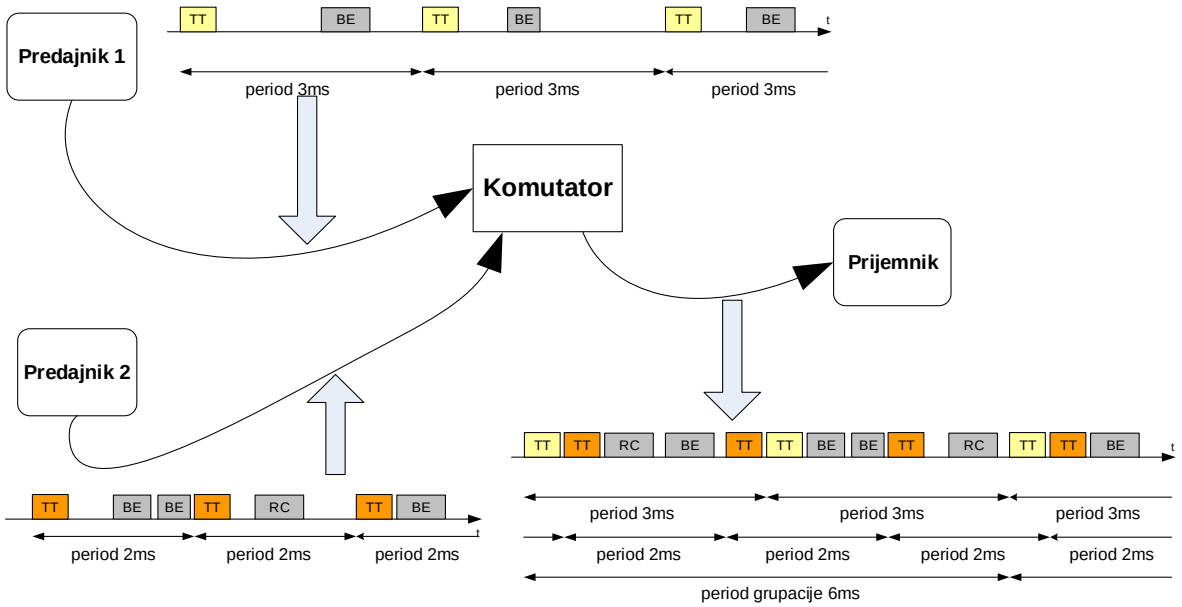
PCF okvir je standardni *Ethernet* okvir dužine 64 bajta, kod kojeg se u *payload* sekciji prenose podaci koji se primjenjuju za vremensko usklađivanje komponenata. Ovi podaci (polja *payload* sekcije) su sledeći [41]:

- Integracioni ciklus (eng. *Integration Cycle*): nosi informaciju o rednom broju integracionog ciklusa unutar perioda grupacije.
- Vektor pripadnosti (eng. *membership vector*): nosi informaciju o broju SM uređaja na osnovu kojih je nastao posmatrani okvir. Ovo je jedan 32-bitni vektor koji sadrži onoliko bita postavljenih na vrijednost 1, koliko je bilo SM uređaja koji su učestvovali u pravljenju tog okvira. Pri projektovanju mreže, postavlja se određeni prag (eng. *synchronization threshold*) koji definiše koliko minimalno bita mora biti postavljeno na 1 u posmatranom vektoru pripadnosti da bi se komprimovani okvir smatrao validnim. Komprimovani okvir koji ima broj postavljenih bita ispod definisanog praga, smatra se nevalidnim čak i ako je primljen unutar prozora prijema.
- Prioritet usklađivanja (eng. *sync priority*): prioritet PCF okvira.
- Domen vremenskog usklađivanja (eng. *sync domain*): informacija o grupi komponenata u posmatranoj *TTEthernet* mreži koje imaju isti pojam o tačnom vremenu.

- Tip (eng. *type*): tip PCF okvira (CS/CA/IN).
- Vrijednost transparentnog časovnika (eng. *transparent clock*): akumulirano kašnjenje (izraženo u pikosekundama) od generatora PCF okvira do korisnika tog okvira.

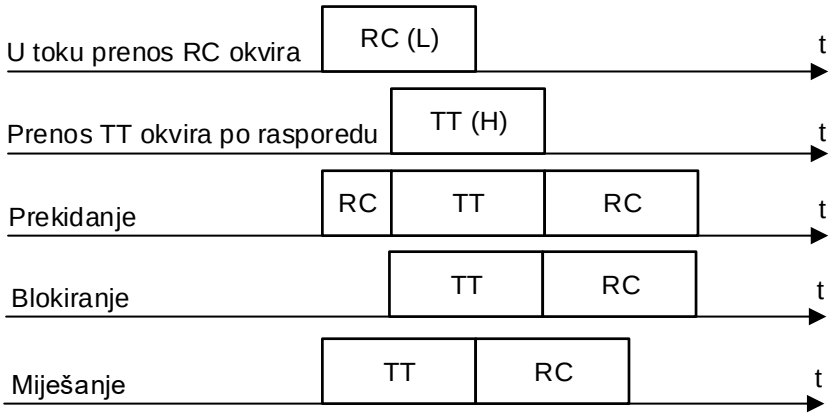
5.2 Integracija različitih klasa saobraćaja

Kao što je rečeno ranije, *TTEthernet* je mreža sa više prioriteta u kojoj može da koegzistira više klasa saobraćaja različitih prioriteta u isto vrijeme. Primjer jedne komunikacije gdje postoji jedan komutator i tri krajnja uređaja, te u kojoj postoje tri različite klase saobraćaja (PCF okviri nisu prikazani), dat je na Sl. 25 [3].



Sl. 25. Primjer integracije više klasa saobraćaja [3]

Primarno je da se TT okviri pošalju u precizno definisanim trenucima, dok se RC i BE okviri proslijeđuju isključivo u slobodnom prostoru perioda grupacije. Često se koriste principi kojima se obezbjeđuje uspješan prenos paketa nižeg prioriteta (RC) i u slučajevima kada je prenos paketa višeg prioriteta (TT) u toku. To su mehanizmi prekidanja (eng. *preemption*), blokiranja (eng. *timely-block*) i miješanja (eng. *shuffling*), koji su ilustrovani na Sl. 26 [43].



Sl. 26. Mehanizmi integracije TT i RC klasa saobraćaja [43]

5.2.1 Mehanizam prekidanja

Mehanizam prekidanja podrazumijeva da će slanje okvira nižeg prioriteta (RC) biti obustavljeno sve dok se ne prenese okvir višeg prioriteta (TT). Nakon završenog slanja TT okvira ponovo se prenosi kompletan RC okvir. Problem kod ovog mehanizma je izgubljeni propusni opseg koji je mogao da se iskoristi za slanje nekog drugog okvira kraćeg trajanja (npr. slanje BE okvira). Prednost ovog pristupa je što nema kašnjenja slanja TT okvira. Izgubljeni propusni opseg zavisi od razlike trenutaka slanja TT i RC okvira, kao i od brzine fizičke veze na kojoj je ostvaren prenos [43]:

$$BW_{izgubljeni} = \frac{\Delta_{[t_{TT},t_{RC}][bit]}}{l_{brzina}[bit/s]}[s] \tag{19}$$

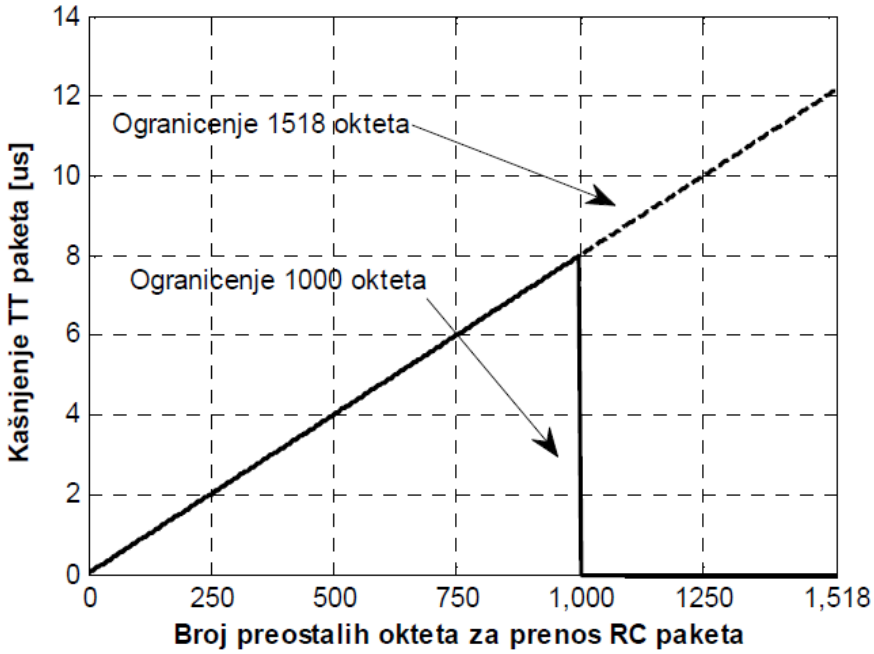
5.2.2 Mehanizam blokiranja

Mehanizam blokiranja funkcioniše tako da slanje RC okvira neće ni početi ukoliko kompletan RC okvir najvećeg mogućeg trajanja (12.304 μs na vezi brzine 1 Gb/s) ne može biti poslan prije slanja TT okvira. Tek nakon slanja TT okvira pristupa se slanju RC okvira, kada se zna da on može biti kompletan isporučen. Nedostatak ovog pristupa ogleda se u činjenici da tada može doći do odbacivaja 19 RC okvira najkraćeg trajanja (0.67 μs na vezi brzine 1Gb/s). Kao i kod mehanizma prekidanja, ni ovdje ne dolazi do promjene trenutka slanja TT okvira [43].

5.2.3 Mehanizam miješanja

Mehanizam miješanja doprinosi najvećem dinamičkom kašnjenju TT paketa, jer se TT okvir prenosi tek nakon završetka slanja RC okvira, ukoliko je slanje RC okvira bilo u toku. Prednost ovog pristupa je što ne dolazi do gubitka propusnog opsega. Ukoliko se koristi mehanizam miješanja, potrebno je izvršiti odgovarajuće proširenje prozora prijema (Sl. 8). Ako je dozvoljena količina pomjeranja TT okvira konfigurabilna, moguće je podesiti da li će se miješanje uopšte primjenjivati, ili će prenos RC okvira biti prekinut. Drugim riječima, miješanje će se primjeniti samo ako je dovoljna količina okteta RC okvira već prenesena.

Primjer primjene mehanizma miješanja za podešeno ograničenje 1000 okteta dato je na Sl. 27. Slika pokazuje da ukoliko najmanje 1000 okteta RC okvira nije veće preneseno, neće se ni primjeniti mehanizam miješanja. Za najgori podešeni slučaj (preneseno 1000 okteta RC okvira), kašnjenje TT okvira iznosi 8 µs na vezi brzine 1Gb/s. Ako je podešena primjena ovog mehanizma za maksimalno trajanje RC okvira, tada kašnjenje TT okvira iznosi 12.304 µs [43].



Sl. 27. Ponašanje mehanizma miješanja za ograničenje 1000 okteta [43]

5.3 Protokol usklađivanja vremena u TTEthernet mrežama

Vremensko usklađivanje u *TTEthernet* mrežama se odvija prema standardu SAE AS6802. Deterministička komunikacija opisana u ovom standardu predstavlja proširenje

drugog sloja OSI modela standardnih *Ethernet* mreža. Korišćenjem ovog pristupa omogućena je deterministička, vremenski usklađena komunikacija bez sudara (eng. *collision*) okvira čak i ako pored deterministčke postoji i nedeterministička klasa saobraćaja. Protokol vremenskog usklađivanja je otporan na greške te može samostalno da se stabilizuje u slučaju eventalnih otkaza. Ovo omogućava potpunu izolaciju deterministčke od nedeterministčke klase saobraćaja, jer u slučajevima narušenog vremenskog usklađivanja kada deterministčka komunikacija ne može da se izvodi, nedeterministički saobraćaj može da se odvija bez problema. Primjenom SAE AS6802 standarda u mrežnim uređajima kao što su komutatori i mrežne kartice, *Ethernet* mreža postaje deterministčka mreža koju karakteriše garantovano kašnjenje okvira, visoka preciznost, te mogućnost prenosa okvira različitih prioriteta. Drugim riječima, preko istih medijuma u isto vrijeme, pomoću koncepta vremenskog multipleksiranja mogu da se prenose kontrole i komande u realnom vremenu, kao i saobraćaj nižeg prioriteta poput video signala, zvuka i glasa [41].

5.3.1 Uloge uređaja i koncept procesa vremenskog usklađivanja

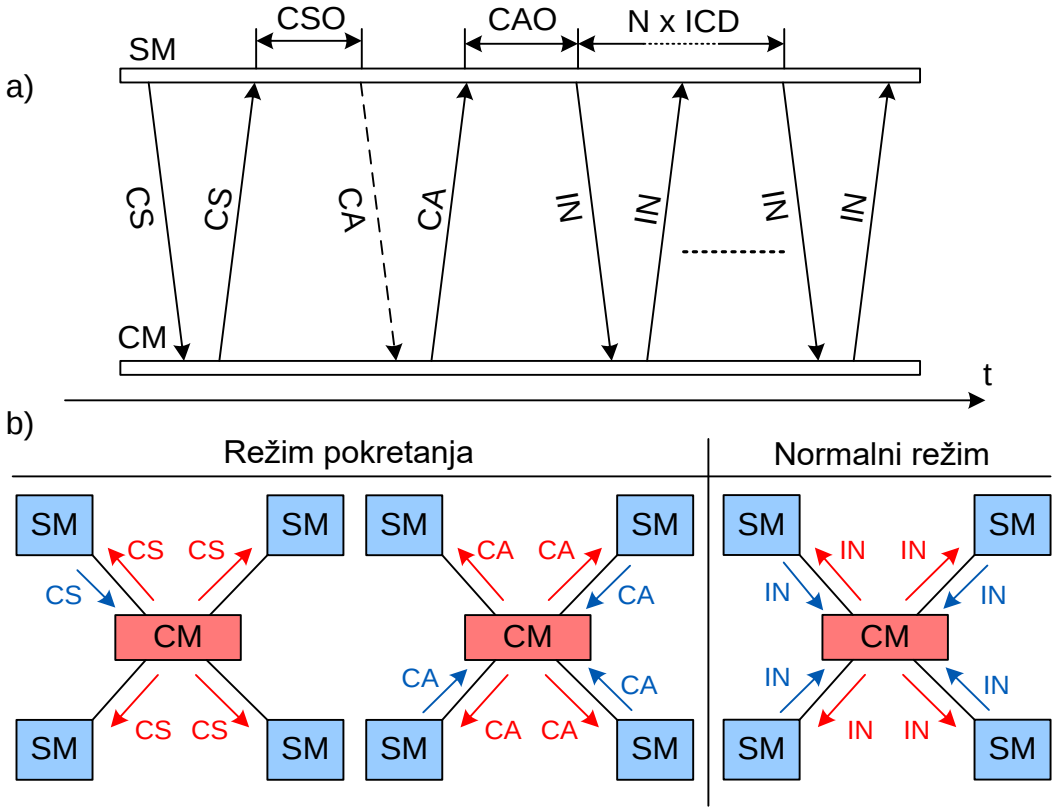
U *TTEthernet* mreži, svaki uređaj ima svoju ulogu u procesu usklađivanja vremena [2], [46]:

- SM (eng. *Synchronization Master*)
- SC (eng. *Synchronization Client*)
- CM (eng. *Compression Master*)

Obično se uloga SM uređaja dodjeljuje krajnjem uređaju, a CM uređaja komutatoru, mada standard SAE AS6802 dozvoljava i drugačije konfiguracije. Iniciranje i održavanje vremenskog usklađivanja odvija se pod uticajem uređaja sa SM ulogom. CM uređaj prikuplja okvire generisane od strane svih SM uređaja u mreži, te ih koristi u svrhe izrade rezultujućeg (tzv. komprimovanog IN okvira) koji se upotrebljava za korekciju časovnika svih uređaja u posmatranoj *TTEthernet* mreži. SC uređaj je pasivna komponenta koja ne generiše nikakav saobraćaj, nego samo vrši korekciju sopstvenog časovnika pod uticajem okvira primljenim od CM uređaja [2], [56-57].

Protokol usklađivanja vremena ilustrovan je na Sl. 28 a), gdje su svi SM uređaji tokom vremena *t* predstavljeni blokom SM na vrhu slike, dok je CM uređaj predstavljen blokom CM na dnu slike. Pojednostavljeni proces razmjene okvira u mreži sa 1 CM i 4 SM uređaja prikazan je na Sl. 28 b). Jedan od SM uređaja inicira proces vremenskog usklađivanja

slanjem svog CS okvira. S obzirom da u mreži postoji više SM uređaja, koji od njih će prvi poslati svoj CS okvir zavisi od stanja lokalnog časovnika posmatranog SM uređaja. Pošto lokalni časovnici korišćeni u uređajima nisu idealni, njihove vrijednosti se razlikuju za različite SM uređaje. Onaj SM uređaj kod koga lokalni časovnik prvi dostigne predefinisani trenutak u kome treba biti poslan CS okvir, izvršiće transmisiju prvog CS okvira [2].



Sl. 28. Ilustracija protokola usklađivanja vremena u *TTEthernet* mreži [2]

Nakon što CM uređaj primi CS okvir, on ga prosljeđuje svim ostalim uređajima u posmatranoj mreži. Kada SM uređaj primi CS okvir, on čeka određeno vrijeme definisano CSO parametrom (eng. *Coldstart Offset*), prije nego što pošalje svoj CA okvir. Treba napomenuti da SM uređaj koji je prvi poslao svoj CS okvir neće poslati CA okvir, tj. CA okvire šalju samo ostali SM uređaji u mreži. Zbog ovog razloga transmisija CA okvira je predstavljena isprekidanom linijom. Uz pretpostavku da je SM uređaj koji je prvi poslao CS okvir bio pod greškom, na opisani način je obezbijeđeno da samo ispravni SM uređaji šalju svoje CA okvire. Smatra se da ostali uređaju nisu pod greškom [2].

Poslije prijema CA okvira pravi se pauza određena parametrom CAO (eng. *Coldstart Acknowledge Offset*), nakon koje svi SM uređaju periodično počinju da šalju integracione

(IN) okvire prema CM uređaju. Trajanje perioda predstavlja integracioni ciklus [58]. U toku jednog integracionog ciklusa odvijaju se sledeće operacije [2]:

- Svi SM uređaji šalju svoje integracione okvire komutatoru sa CM ulogom
- CM uređaj prikuplja sve integracione okvire iz prethodnog koraka i izrađuje komprimovani IN okvir koji se koristi u svrhe vremenskog usklađivanja. Vektor pripadnosti komprimovanog okvira sadrži broj postavljenih bita koji odgovara broju SM uređaja koji su učestvovali u izradi tog komprimovanog IN okvira.
- CM uređaj primjenjuje komprimovani IN okvir za korekciju sopstvenog lokalnog časovnika, te ga odašilje ostatku mreže.
- SM/SC uređaji primaju komprimovani IN okvir od CM uređaja te ga primjenjuju za korekciju svojih lokalnih časovnika.

Kompletan proces usklađivanja vremena može se podijeliti na dva režima [2]:

- Režim pokretanja / ponovnog pokretanja (često se ovaj režim definiše i odvojeno, kao režim pokretanja i režim ponovnog pokretanja): Komponente u mreži nisu međusobno vremenski usklađene, te nastoje da se usklade. Bez uspostavljenog usklađenog vremena u mreži, deterministička komunikacija nije moguća. U ovoj fazi razmjenjuju se CS i CA okviri između komponenata u *TTEthernet* mreži.
- Normalni režim rada: U ovom režimu smatra se da su mrežne komponente vremenski usklađene, te se periodično razmjenjuju integracioni okviri u integracionim ciklusima. Smatra se da je mreža spremna za nesmetano odvijanje determinističke komunikacije.

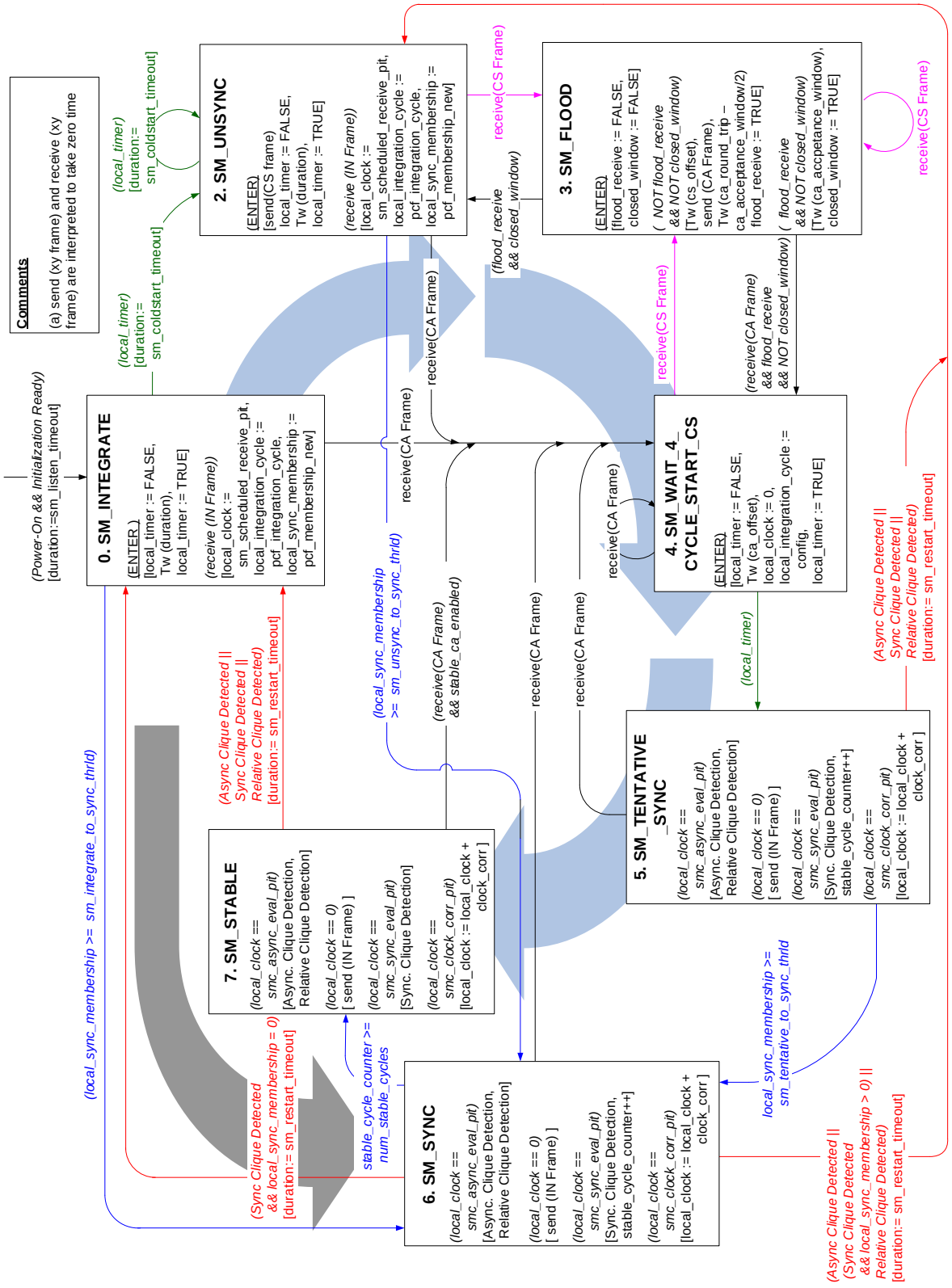
5.3.2 Dijagram mašine stanja uređaja u TTEthernet mreži

Svaki od uređaja u *TTEthernet* mreži (SM, SC, CM) radi prema odgovarajućoj mašini stanja. U okvirima ovog rada, analiziraćemo mašinu stanja realizovanu u SM uređaju. Detaljno objašnjenje mašina stanja za sve uloge dato je u standardu SAE AS6802. Mašina stanja uređaja sa SM ulogom predstavljena je na Sl. 29. Principi rada mašina stanja uređaja sa SC i CM ulogama su slični, tako da neće biti detaljno analizirani u ovoj tezi [2], [41].

Posmatrajući Sl. 29, prikazana stanja možemo podijeliti u dvije grupe [2]:

- Vremenski neusklađena (eng. *asynchronous*): SM_INTEGRATE, SM_UNSYNC, SM_FLOOD i SM_WAIT_4_CYCLE_START_CS. Postoji još jedno stanje koje je opciono, te neće biti razmatrano u okvirima ovog rada: SM_WAIT_4_CYCLE_START. Kao što samo ime kaže, u ovim stanjima posmatrani uređaj nije vremenski usklađen sa ostatkom mreže, te preduzima neophodne radnje za usklađivanje vremena. Tokom režima pokretanja ili ponovnog pokretanja SM uređaj prolazi kroz sva ili kroz neka od ovih stanja.
- Vremenski usklađena (eng. *synchronous*): SM_TENTATIVE_SYNC, SM_SYNC i SM_STABLE. U ovim stanjima smatra se da je posmatrana komponenta usklađena sa ostatkom mreže, te se periodično razmjenjuju IN okviri u integracionim ciklusima. Smatra se da je SM uređaj u normalnom režimu rada, ako je u nekom od ovih stanja.

Kao što smo objasnili u prethodnim poglavljima, režim pokretanja može se podijeliti na režim hladnog starta i režim integracije. Stanja kroz koja SM uređaj prolazi dok je u režimu hladnog starta predstavljena su debelom plavom strelicom u pozadini dijagrama (Sl. 29) usmjerenom u smijeru kazaljke na satu, dok je režim integracije predstavljen sivom debelom strelicom suprotne orijentacije.



Sl. 29. Mašina stanja za uređaj sa SM ulogom [41]

Svaki blok na datom dijagramu mašine stanja sadrži uslove označene sa „( )“ koji su praćeni komandama koje se izvršavaju kada je posmatrani uslov ispunjen – označeno sa „[ ]“.

Postoji jedna posebna sekcija označena sa ENTER, koja predstavlja listu komandi koja će se izvršiti svaki put kada SM uređaj uđe u posmatrano stanje. Plave strelice tranzicije predstavljaju prelaz u više vremenski usklađeno stanje, dok crvene strelice predstavljaju gubitak vremenske usklađenosti. Crna i roza boja predstavljaju prijem CA i CS okvira respektivno, dok zelena strelica označava prekid tajmera [2].

Po uključenju SM uređaja, on ulazi u SM_INTEGRATE stanje i nastoji da se pasivno uskladi sa ostatkom mreže. U SM_INTEGRATE stanju, SM uređaj ne generiše nikakav saobraćaj nego očekuje komprimovani IN okvir od CM uređaja. Ukoliko je primljeni IN okvir validan, SM uređaj prelazi u SM_SYNC stanje u kojem počinje da šalje periodično sopstvene IN okvire. Ukoliko nije primljen IN okvir od CM uređaja u toku predefinisanog vremenskog perioda (tajmer generiše prekid kada njegovov brojač dostigne vrijednost `sm_coldstart_timeout`), ili je IN okvir nevalidan, SM uređaj smatra da ostatak mreže nije u usklađenom stanju, te počinje da inicira proces usklađivanja vremena za cjelukupnu mrežu. Ta aktivnost se ostvaruje tako što SM uređaj prelazi u SM_UNSYNC stanje u kojem počinje sa periodičnim emitovanjem CS okvira (Sl. 28). Nakon prijema CS okvira, CM uređaj ga proslijeđuje svim ostalim uređajima u mreži, uključujući i SM uređaj koji ga je prvi poslao. SM uređaj zna da je CM uređaj dostupan nakon što mu CM uređaj vrati CS okvir, te prelazi u više stanje – SM_FLOOD. Ako ne primi CS okvir u predefinisanom vremenskom periodu (`sm_coldstart_timeout`), SM uređaj ponovo ulazi u SM_UNSYNC stanje i šalje sledeći CS okvir.

Kada uđe u SM_FLOOD stanje, SM uređaj počinje sa emitovanjem CA okvira. Kao što je rečeno ranije (Sl. 28), SM uređaj koji je inicirao proces vremenskog usklađivanja neće slati svoj CA okvir, nego će to izvršiti samo ostali SM uređaji u posmatranoj *TTEthernet* mreži. Komutator koji je u ulozi CM uređaja će prikupiti sve CA okvire, izraditi novi, komprimovani CA okvir na osnovu svih primljenih okvira koji su primljeni u prozoru prijema (Sl. 8), te će taj komprimovati okvir emitovati ostatku mreže. Ako posmatramo SM uređaj, nakon transmisije svog CA okvira on je zakazao otvaranje svog prozora prijema u određenom trenutku, koji je fiksnog trajanja. U ovom prozoru prihvatanja, SM uređaj očekuje prijem komprimovanog CA okvira. Ukoliko je primljeni komprimovani CA okvir validan i primljen za vrijeme otvorenog prozora prijema (promjenljiva `flood_receive` je postavljena na TRUE, dok je promjeljiva `closed_window` postavljena na FALSE), SM uređaj prelazi u više stanje – SM_WAIT_4_CYCLE_START_CS. Komprimovani CA okvir primljen izvan prozora prijema uzrokuje tranziciju u niže stanje – SM_UNSYNC. Takođe,

ako je CS okvir primljen za vrijeme dok je SM uređaj u SM_FLOOD stanju, SM uređaj će se vratiti u SM_UNSYNC stanje. Integracioni okvir primljen u SM_FLOOD stanju ne uzima se u razmatranje.

SM_WAIT_4_CYCLE_START_CS je stanje u kojem SM uređaj čeka podešeni vremenski period (`ca_offset`), nakon čega se javlja prekid tajmera koji dovodi SM uređaj u prvo usklađeno stanje – SM_TENTATIVE_SYNC. U slučaju prijema CA okvira dok je SM uređaj u SM_WAIT_4_CYCLE_START_CS stanju, SM uređaj ponovo ulazi u isto stanje. Integracioni okviri ne uzimaju se u razmatranje ako je SM uređaj u ovom stanju, dok primljeni CS okvir uzrokuje tranziciju u niže stanje – SM_FLOOD.

U TENTATIVE_SYNC stanju SM uređaj počinje da periodično šalje IN okvire. Ako je kompletna mreža uredno pokrenuta, takođe i ostali SM uređaji šalju svoje IN okvire. Odmah po slanju IN okvira, SM uređaj zakazuje trenutak otvaranja prozora prijema u kojem očekuje prijem komprimovanog IN okvira. CM uređaj prikuplja IN okvire svih SM uređaja u mreži i generiše novi, komprimovani IN okvir. Nakon izrade komprimovanog okvira, CM uređaj generiše vještačko kašnjenje koje je jednako srednjoj vrijednosti trenutaka prijema svih primljenih integracionih okvira. Poslije isteka perioda ovog kašnjenja, CM uređaj emituje komprimovani IN okvir ostatku mreže. Ako je komprimovani IN okvir validan (vrijednost praga veća od `sm_tentative_to_sync_threshold`) i primljen za vrijeme otvorenog prozora prijema, SM uređaj izvršava korekciju svog lokalnog časovnika i prelazi u više usklađeno stanje – SM_SYNC. Nevalidan komprimovani IN okvir primljen unutar prozora prijema uzrokuje sinhroni CD događaj (eng. *Synchronous Clique Detection*) i tranziciju u SM_UNSYNC stanje. Takođe, validan komprimovani IN okvir primljen izvan prozora prijema uzrokuje tranziciju u SM_UNSYNC stanje, jer je tada prepoznat asinhroni CD događaj (eng. *Asynchronous Clique Detection*). U slučajevima da se primi više komprimovanih okvira unutar istog prozora prijema, u obzir se uzima i relativni CD događaj (eng. *Relative Clique Detection*) koji predstavlja raziku između broja bita postavljenih u vektoru pripadnosti za okvire koji su primljeni unutar i izvan prozora prijema. Ukoliko je broj postavljenih bita veći za okvire primljene izvan prozora prijema, prepoznat je relativni CD događaj. Što se tiče CS okvira, oni se ignorišu u SM_TENTATIVE_SYNC stanju, dok CA okviri uzrokuju prelaz u SM_WAIT_4_CYCLE_START_CS stanje.

Sledeće vremenski usklađeno stanje je SM_SYNC. Ponašanje SM uređaja je isto kao kada je i u SM_TENTATIVE_SYNC stanju. Nakon predefinisanog broja stabilnih

integracionih ciklusa u SM_SYNC stanju, određenog vrijednošću parametra `stable_cycle_counter`, SM uređaj prelazi u SM_STABLE stanje. SM_STABLE stanje predstavlja najviše vremenski usklađeno stanje i nakon ulaska u ovo stanje smatra se da je faza starta posmatranog uređaja završena. Kao i u SM_TENTATIVE_SYNC stanju, asinhroni/sinhroni/relativni CD događaj uzrokuje gubitak vremenske usklađenosti i u SM_SYNC/SM_STABLE stanjima. Primljeni CA okvir uzrokuje prelazak u SM_WAIT_4_CYCLE_START_CS stanje, ako je SM uređaj bio u nekom od stanja SM_SYNC/SM_STABLE.

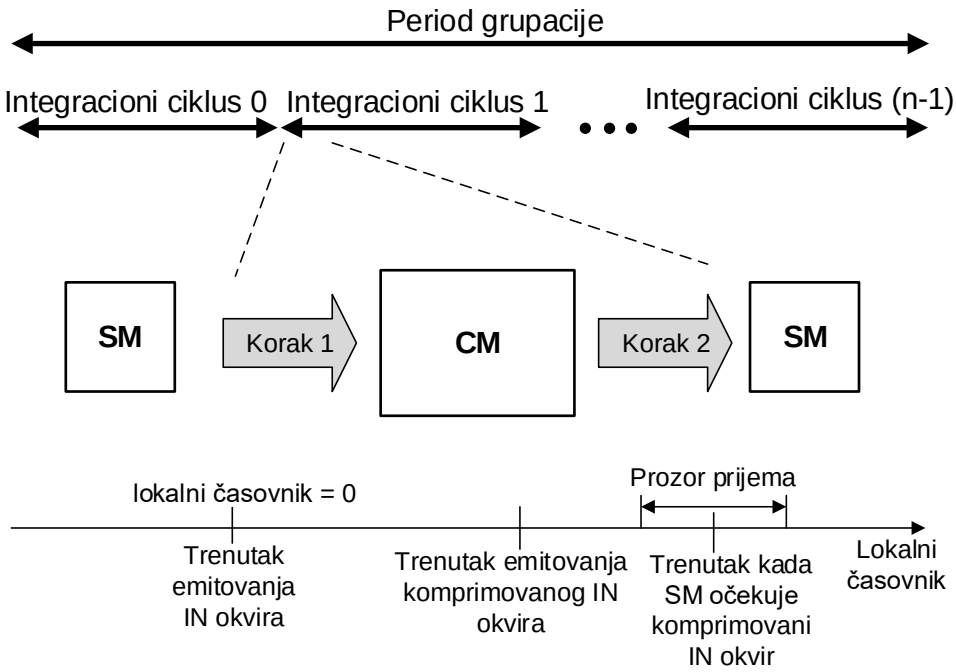
Prepoznat CD događaj inicira fazu ponovnog pokretanja SM uređaja. Da li će uređaj ići u režim integracije ili hladnog starta prilikom ponovnog pokretanja zavisi od stanja vektora pripadnosti, ako posmatramo stanje SM_SYNC. Ukoliko je broj postavljenih bita u vektoru pripadnosti jednak nuli kada je prepoznatsinhroni CD događaj, SM uređaj pokušava integraciju (prelazi u SM_INTEGRATE stanje). Ako je broj postavljenih bita veći od 0, SM uređaj započinje hladni start ako je bio prepoznatsinhroni CD događaj (direktno prelazi u SM_UNSYNC stanje). Sinhroni CD događaj prepoznatu SM_STABLE stanju inicira uvijek pokušaj integracije prije nego što se počne sa hladnim startom. Podaci o stanjima vektora pripadnosti za vrijeme sinhronog i asinhronog CD događaja čuvaju se u vektorima `local_sync_membership` i `local_async_membership` respektivno. Razlika broja bita između spomenutadva vektora predstavlja relativni CD događaj [2-3], [41].

S obzirom da je princip rada mašina stanja za CM i SC uređaje sličan kao i za SM uređaj, nećemo ih analizirati, ali detaljan opis principa njihovog rada dat je u standardu SAE AS6802 [41].

5.3.3 Princip korekcije lokalnog časovnika

Nakon što se uspostavi normalni režim rada, u *TTEthernet* mreži se periodično razmjenjuju integracioni okviri, kao što je ilustrovano na Sl. 30. Normalni režim rada podrazumijeva da se između komponenata periodično razmjenjuju integracioni okviri u integracionim ciklusima. Neophodne radnje se izvršavaju na osnovu stanja lokalnog časovnika `local_clock`. Kada vrijednost varijable `local_clock` postane jednaka 0 (vrijednost koja odgovara početku integracionog ciklusa), SM uređaj šalje svoj integracioni okvir. Lokalni časovnik broji samo od 0 do podešene dužine jednog integracionog ciklusa, nakon čega se resetuje i dovodi na početnu vrijednost. Promjenljiva `local_timer` je

asinhrona i njena vrijednost se ne koriguje u procesu usklađivanja vremena. Uloga spomenute promjenljive je u generisanju neophodnih pauza (eng. *timeout*) tokom rada posmatrane mrežne komponente.



Sl. 30. Razmjena integracionih okvira između SM i CM uređaja [3]

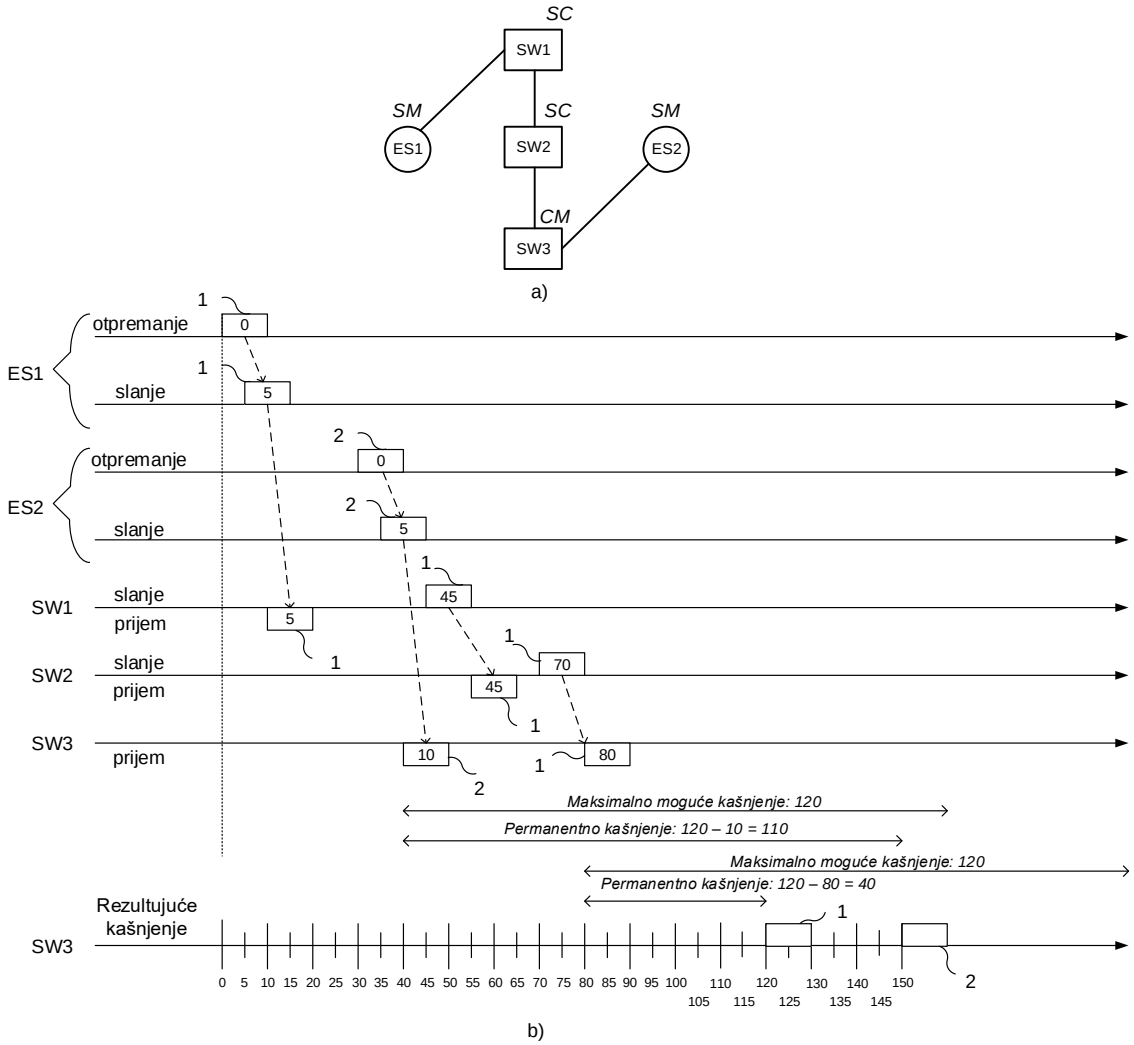
U koraku 1, CM uređaj prikuplja integracione okvire od svih SM uređaja u posmatranoj *TTEthernet* mreži. Po prijemu prvog integracionog okvira, CM uređaj otvara tzv. prozor osmatranja (eng. *observation window*). Postoje 2 bitna trenutka u radu CM uređaja: trenutak kada primljeni okvir postane permanentan (eng. *Permanence Point in Time*) i trenutak kada je nastao komprimovani IN okvir (eng. *Compressed Point in Time*) [3].

5.3.3.1 **Permanentno kašnjenje i transparentni časovnik**

Vrijednost polja transparentnog časovnika PCF okvira služi samo da bi se uspostavio odgovarajući poredak okvira primljenih na CM uređaju. Kada CM uređaj primi IN okvir on ga ne uzima odmah u razmatranje, nego se generiše vještačko kašnjenje tog okvira koje se naziva permanentno kašnjenje. Nakon što se završi ovo kašnjenje, kaže se da je posmatrani okvir postao permanentan i CM uređaj može da počne sa njegovom upotrebom. Ovdje bitnu ulogu igra vrijednost polja transparentnog časovnika (eng. *transparent clock*) primljenog IN okvira. Odmah po predaji IN okvira, vrijednost polja transparentnog časovnika je podešena na 0. Kako okvir prolazi kroz međustepene u mreži (komutatori), vrijednost transparentnog časovnika se uvećava tako da odgovara akumuliranom kašnjenju okvira do tog međustepena.

Na kraju, kada okvir dođe na odredište, vrijednost polja transparentnog časovnika odgovara ukupnom akumuliranom kašnjenju okvira od njegove predaje do odredišta. Koristeći princip permanentnog kašnjenja, obezbijeđeno je da CM uređaj radi adekvatno čak i slučaju da SM uređaji nisu udaljeni od njega za simetričan broj međustepeni. Svaki međustepen (komutator sa SC ulogom) unosi svoje statičko kašnjenje. Ukupno statičko kašnjenje okvira direktno zavisi od broja međustepeni od predaje poruke (SM strana) do njenog prijema (CM strana). Ako posmatramo npr. 2 SM uređaja koji u istom trenutku šalju svoje PCF okvire, u slučaju asimetričnog broja međustepeni, njihov prijem na CM strani biće u različitim trenucima. Zato se pri projektovanju mreže definiše maksimalno moguće kašnjenje okvira od koga se kasnije oduzima vrijednost transparentnog časovnika dobijena iz PCF okvira. Na ovaj način postiže se raspored prijema okvira na CM uređaj koji odgovara rasporedu slanja okvira od strane SM uređaja, bez obzira na broj međustepeni od CM uređaja do posmatranih SM uređaja [41].

Koncept transparentnog časovnika i permanentnog kašnjenja biće objašnjen na primjeru prikazanom na Sl. 31 [41]. Topologija i uloge uređaja su prikazane na Sl. 31.a), a razmjena okvira na Sl. 31.b). Krajnji uređaj ES2 je direktno priključen na CM uređaj, dok okvir emitovan sa ES1 treba da prođe kroz 2 međustepena (SW1 i SW2) dok ne stigne do CM uređaja.



Sl. 31. Primjena koncepta transparentnog časovnika i permanentnog kašnjenja za uspostavljanje odgovarajućeg redoslijeda primljenih IN okvira na strani CM uređaja [41]

ES2 šalje IN okvir 30 vremenskih jedinica nakon što ES1 pošalje svoj IN okvir. Linija "otpremanje" sadrži trenutak kada je zakazano slanje okvira, dok linija "slanje" sadrži trenutak kada je okvir stvarno poslan sa priključka. Na strani ES1 i ES2, razlika između zakazanog trenutka slanja i stvarnog slanja okvira iznosi 5 vremenskih jedinica. Odmah po slanju, vrijednost 5 je upisana u polje transparentnog časovnika posmatranog PCF okvira. Vrijednost doprinosa transparentnog časovnika trenutnog stepena n se računa na sledeći način:

$$PCF_{transparentni\_časovnik(n)} = PCF_{transparentni\_časovnik(n-1)} + D_{din(n)} + D_{stat(n)} + D_{kanala(n)} \quad (20)$$

gdje je $PCF_{transparentni\_časovnik(n-1)}$ doprinos prethodnog stepena transparentnom časovniku, $D_{din(n)}$ i $D_{stat(n)}$ dinamičko i statičko kašnjenje prijema stepena *n*, a $D_{kanala(n)}$ kašnjenje kanala [41].

Ako posmatramo ES2, njegov PCF okvir prvi stiže na CM uređaj, iako je poslan 30 vremenskih jedinica nakon što je ES1 poslao svoj okvir. Komutatoru SW1 treba 40 vremenskih jedinica da proslijedi okvir, dok SW2 proslijedi okvir za 25 sekundi. Nakon prolaska okvira kroz komutator, ažurira se i vrijednost polja transparentnog časovnika PCF okvira. Imajući prethodno navedena kašnjenja u vidu, IN okvir sa ES1 je stigao na CM uređaj nakon 80, a IN okvir sa ES2 za 10 vremenskih jedinica nakon njihove predaje. Potrebno je obezbjediti da redoslijed prijema okvira na CM uređaju bude identičan kao redoslijed njihove predaje sa krajnjeg uređaja. Tek nakon što se ovo obezbijedi, primljeni okviri na CM uređaju će postati permanentni. Iz ovog razloga, pri podešavanju *TTEthernet* mreže definisano je maksimalno moguće kašnjenje okvira 120 vremenskih jedinica. Odmah po prijemu okvira na strani CM uređaja, aktivira se funkcija permanentnog kašnjenja koja vrši oduzimanje vrijednosti transparentnog časovnika od maksimalnog mogućeg kašenjenja. Kao rezultat ove funkcije, primljeni okvir postaje permanentan. Sa Sl. 31 je vidljivo da okvir emitovan sa ES1 postaje permanentan u trenutku 120, a okvir emitovan sa ES2 u trenutku 150 vremenskih jedinica. Razlika između permanentih okvira između krajnjih uređaja ES1 i ES2 je 30 vremenskih jedinica što je identično njihovom rastojanju na predaji. Bez primjene permanentnog kašnjenja rastojanje između primljenih okvira na CM uređaju bi bilo 40 vremenskih jedinica i njihov redoslijed bi bio obrnut, što bi dovelo do pogrešnog proračuna količine korekcije lokalnih časovnika, koji bi se odrazio na kompletnu mrežu. Ukoliko bi mreža bila bez dodatnih međustepena između CM i SM uređaja, ili ako bi broj međustepena bio simetričan, mehanizam transparentnog časovnika ne bi bio neophodan da se primjenjuje. Za takve slučajeve, posmatrajući poslane okvire na vremenskoj skali, ne bi bili ni svjesni da se vrši primjena spomenutog mehanizma na spomenute okvire. Permanentno kašnjenje opisujemo sledećim izrazom [41]:

$$D_{permanentno} = D_{max} - PCF_{transparentni\_časovnik} \tag{21}$$

dok trenutak kada je okvir postao permanentan definišemo kao:

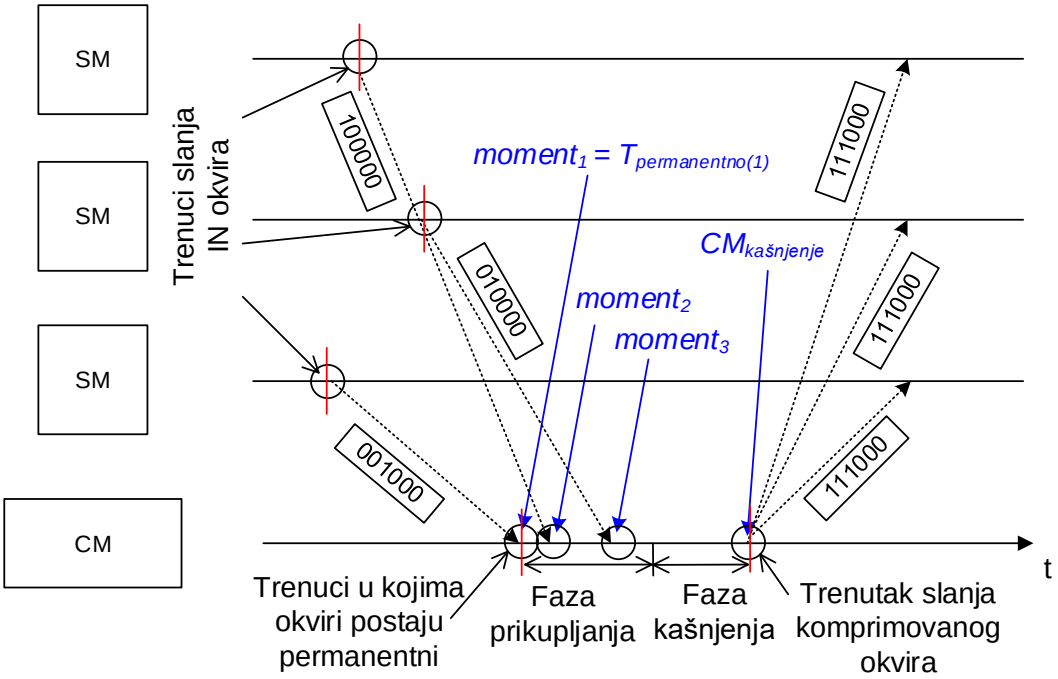
$$T_{permanentno} = T_{prijema} + D_{permanentno} \tag{22}$$

Treba napomenuti da se informacija o količini korekcije lokalnih časovnika ne prenosi unutar PCF okvira, nego se ta informacija dobija iz vremena pristizanja komprimovanog okvira na odredište, o čemu će biti više riječi u nastavku.

5.3.3.2 **Kompresija okvira**

Nakon prijema prvog integracionog okvira, CM uređaj očitava kojem integracionom ciklusu on pripada (polje `pcf_integration_cycle`), te otvara odgovarajući observacioni prozor u kojem nastavlja da prima integracione okvire ostalih SM uređaja koji pripadaju istom integracionom ciklusu. Moguće je da paralelno bude otvoreno nekoliko observacionih prozora u istom trenutku i to se događa ako su primljeni integracioni okviri koji pripadaju različitim integracionim ciklusima. Pojava različitih vrijednosti polja `pcf_integration_cycle` u okvirima javlja se u slučajevima kada je fizička mreža podijeljena na dva ili više međusobno nezavisnih domena usklađenosti (eng. *synchronization domain*), te jedan CM uređaj vrši kompresiju okvira za više od jednog domena usklađenosti. Takođe, u slučajevima gdje postoji definisan samo jedan domen usklađenosti, paralelno otvaranje više observacionih prozora događa se kada je SM uređaj pod otkazom, te nekontrolisano emituje IN okvire sa proizvoljnim vrijednostima polja `pcf_integration_cycle`. Spomenute aktivnosti se obavljaju u fazi prikupljanja okvira (eng. *collection phase*) [41].

U svakom IN okviru poslanom od strane SM uređaja, sadržan je njegov identifikator u vektoru pripadnosti. To je jedan postavljeni bit koji je jedinstven za svaki SM uređaj u mreži. Kada CM uređaj prikupi IN okvire od svih SM uređaja u mreži on izrađuje komprimovani okvir čiji vektor pripadnosti sadrži onoliko postavljenih bita koliko je bilo SM uređaja na osnovu kojih je nastao, kao što je prikazano na Sl. 32. Nakon faze prikupljanja, koja nastupa odmah po prijemu prvog IN okvira, slijedi faza kašnjenja (eng. *delay phase*). SM uređaji šalju svoje IN okvire u istom trenutku prema njihovim lokalnim časovnicima, ali kada ih posmatramo u cjelosti, ti trenuci su različiti zbog neidealnih časovnika korišćenih u mrežnim uređajima (neki od njih otkucavaju sporije, a neki brže) [41].



Sl. 32. Faze nastajanja komprimovanog okvira [3]

Generisano kašnjenje CM uređaja se računa kao srednja vrijednost otporna na otkaze momenata pristizanja IN okvira sa SM uređaja [3], [41]:

- Jedan IN okvir: $CM_{kašnjenje} = moment_1$
- Dva IN okvira: $CM_{kašnjenje} = (moment_1 + moment_2) / 2$
- Tri IN okvira: $CM_{kašnjenje} = moment_2$
- Četiri IN okvira: $CM_{kašnjenje} = (moment_2 + moment_3) / 2$
- Pet IN okvira: $CM_{kašnjenje} = (moment_2 + moment_4) / 2$
- Preko pet IN okvira: $CM_{kašnjenje} = (moment_k + moment_r) / 2$, gdje su $moment_k$ i $moment_r$ momenti k-tog najkasnijeg i r-tog najranijeg pristizanja IN okvira. Parametri k i r se biraju u zavisnosti od stepena otpornosti na otkaz za posmatranu mrežu.

Ukupno kašnjenje komprimovanog okvira izražava se kao:

$$D_{kompresije} = D_{prikupljanja} + D_{proračuna} + D_{korekcije} \tag{23}$$

gdje $D_{prikupljanja}$ predstavlja trajanje faze prikupljanja integracionih okvira, $D_{proračuna}$ vrijeme neophodno da CM uređaj proračuna trenutak emitovanja komprimovanog IN okvira, a

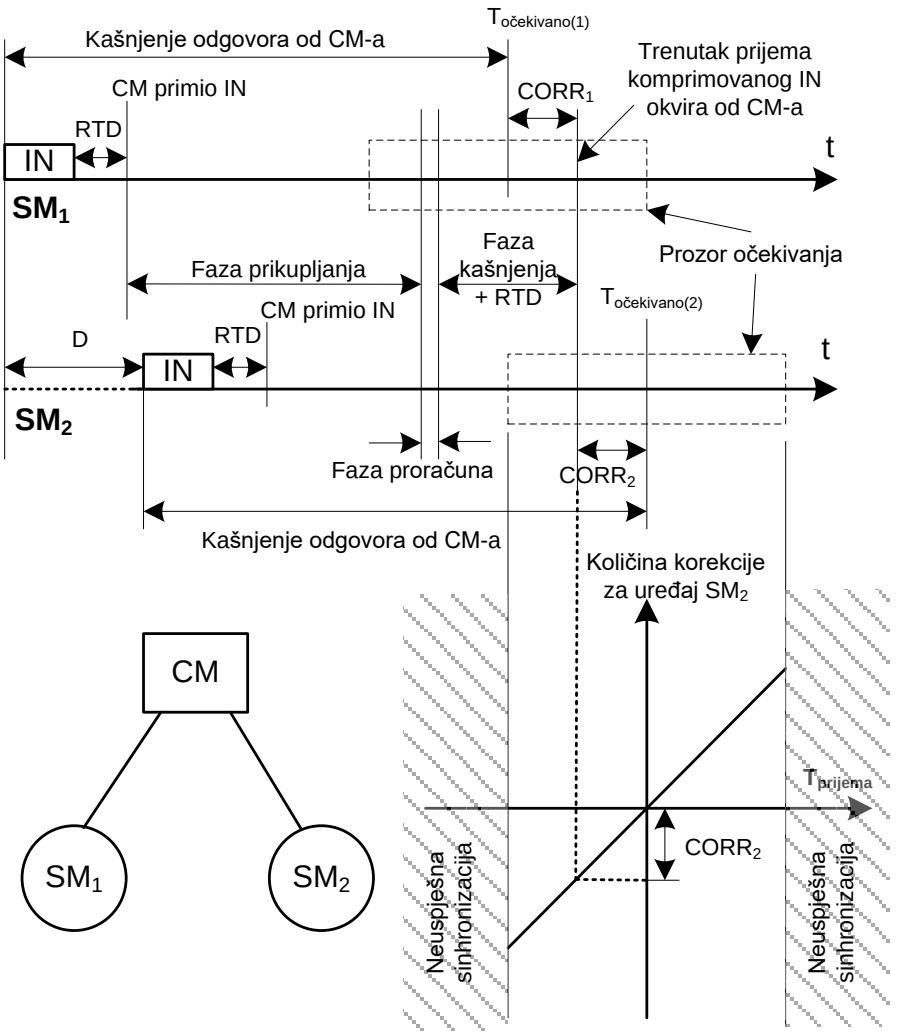
$D_{korekcije}$ izračunato kašnjenje komprimovanog okvira. Imajući ovo u vidu, trenutak kada se emituje komprimovani IN okvir se definiše na sledeći način:

$$T_{komprimovan} = T_{permanentno(1)} + D_{kompresije} \tag{24}$$

gdje parametar $T_{permanentno(1)}$ predstavlja trenutak kada je prvi IN okvir primljen od nekog SM uređaja postao permanentan u CM uređaju [41].

5.3.3.3 **Korekcija lokalnog časovnika**

Detaljniji prikaz procedure korekcije lokalnog časovnika SM uređaja za mrežu sa jednim CM i dva SM uređaja, dat je na Sl. 33. Ovdje su takođe prikazani i drugi parametri kojih nema na Sl. 32, poput kašnjenja proračuna i kašnjenja propagacije RTD (eng. *Round Trip Delay*) [1].



Sl. 33. Princip korekcije lokalnog časovnika SM uređaja [1]

Časovnik SM_2 je sporiji od časovnika SM_1 za vrijednost D. Proširenjem jednačine (24), sa aspekta SM uređaja očekivani trenutak komprimovanog okvira od CM uređaja je:

$$T_{očekivano} = T_{predaje} + RTD + D_{prikupljanja} + D_{proračuna} + D_{kašnjenja} + RTD \quad (25)$$

Uzeto je da je propagacija signala kroz kanal jednaka 0. Čim je SM uređaj emitovao svoj IN okvir, on je zakazao moment otvaranja svog prozora prijema, koji je konačnog trajanja. Centar prozora prijema redstavlja očekivani moment prijema komprimovanog okvira od CM uređaja $T_{očekivano(1)}$ / $T_{očekivano(2)}$. Komprimovani IN okvir je zakašnjen u CM uređaju za srednju vrijednost pristizanja IN okvira na CM uređaj, kao što je objašnjeno ranije. Upravo vrijednost generisanog kašnjenja komprimovanog IN okvira određuje količinu korekcije lokalnih časovnika ($CORR_1$, $CORR_2$). Na osnovu ovih vrijednosti SM_1 zna da treba da poveća vrijednost svog lokalnog časovnika za $CORR_1$, dok SM_2 treba da ga umanji za vrijednost $CORR_2$. U slučaju idealnih časovnika, momenti prispjeća IN okvira na CM uređaj su isti, tako da je njihova srednja vrijednost kao i $D_{kašnjenja}$ jednako 0. Samim tim dobija se da je $CORR_1$=$CORR_2$=0 [1].

Ukoliko komprimovani IN okvir upadne izvan prozora očekivanja (osijenčena oblast na Sl. 33), na posmatranoj komponenti ne vrši se korekcija časovnika, te se prepoznaje asinhroni CD. U zavisnosti od dozvoljenog broja prepoznatih asinhronih CD, komponenta gubi usklađenost sa ostatkom mreže te prelazi u režim ponovnog pokretanja.

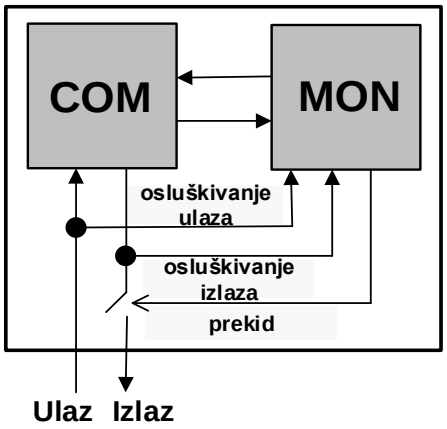
5.4 Rukovanje greškama kod TTEthernet mreža

Otpornost na greške samo jedne komponente u mreži pokazuje se nedovoljnom u bezbjednosno-kritičnim aplikacijama poput avionske industrije. Danas su svi sistemi otporni na otkaze bazirani na odgovarajućem stepenu redundanse koja se dijeli na serijsku i paralelnu. Kod paralelne redundanse svi uređaji rade paralelno i u slučaju otkaza jednog od njih drugi nastavlja sa svojim uobičajenim aktivnostima. Prednost ovog pristupa ogleda se u činjenici da je vrijeme oporavka sistema jednako 0, tj. nema diskontinuiteta u normalnom funkcionisanju mreže. Za razliku od paralelnog tipa redundanse, kod serijske redundanse samo određeni podskup uređaja obavlja svoje aktivnosti, dok redundantne komponente čekaju u rezervi. U slučaju otkaza primarne komponente, redundantna komponenta preuzima

njenu funkcionalnost. Nedostatak ovog pristupa je što vrijeme oporavka sistema nakon otkaza primarne komponente nije jednako 0.

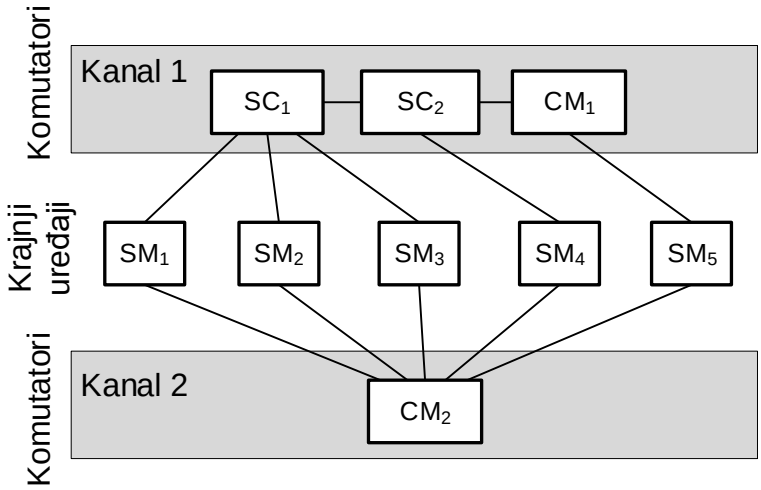
Pošto u *TTEthernet* mrežama postoje dvije vrste komponenata, moguća je pojava dvije vrste otkaza: otkaz krajnjeg uređaja ili otkaz komutatora. Mašina stanja CM uređaja je takva da se on ponaša kao jedna vrsta centralnog čuvara (eng. *central guardian*). Uloga centralnog čuvara podrazumijeva odlučivanje koje okvire će CM uređaj proslijediti, a koje odbaciti. U zavisnosti od trenutnog stanja u kojem se CM uređaj nalazi, on će propuštati samo određene tipove PCF okvira. Na opisani način poboljšava se otpornost na greške kompletne *TTEthernet* mreže, pogotovo u slučajevima tzv. *babbling idiot* scenarija kada krajnji uređaj počne nekontrolisano da emituje proizvoljne sekvence PCF okvira. Npr, dok je CM uređaj u nekom od usklađenih stanja (CM_SYNC, ili CM_STABLE), on uopšte neće uzimati u obzir primljene CS ili CA okvire, nego će ih ignorisati. Za svaki primljeni CS ili CA okvir primljen u spomenutim stanjima, CM uređaj će smatrati da su oni poslani od strane SM uređaja pod otkazom. Jedini tip okvira koji CM uređaj razmatra u ovim stanjima su integracioni okviri.

Jedan od načina da se spriječi transmisija nevalidnog okvira je korišćenje konfiguracije visokog integriteta (eng. *high-integrity design*), čiji je princip prikazan na Sl. 34. Kod ovog pristupa koriste se 2 jedinice na izlazu: COM (eng. *commander*) i MON (eng. *monitor*). Jedinice COM i MON rade u repliciranom režimu, tj. emituju iste signale, a takođe i primljeni signali se prosljeđuju na obje komponente. MON jedinica u svakom trenutku osluškuje signale koje je emitovala COM jedinica. S obzirom da COM jedinica emituje iste signale kao i MON jedinica, u slučaju da se poruka koju emituje COM jedinica razlikuje od poruke koju emituje MON jedinica, MON jedinica smatra da je poruka nevalidna, te signalom prekida transmisiju te poruke na izlaz [41].



Sl. 34. COM/MON par u konfiguraciji visokog integriteta [41]

Kao što je spomenuto ranije, da bi se povećala otpornost na otkaze kod deterministčkih mreža, pored komutatora neophodnog za komunikaciju, uvode se i njegove replike, tj. više komutatora je spojeno paralelno i rade u redundantnoj konfiguraciji. Broj replika u paralelnoj konfiguraciji odgovara stepenu redundanse mreže. Primjer *TTEthernet* mreže u redundantnoj konfiguraciji stepena 2, prikazan je na Sl. 35**Error! Reference source not found.**. Svaki od priključaka SM uređaja je povezan na različite kanale, te u slučaju otkaza jednog od kanala, saobraćaj se nastavlja neometano odvijati preko drugog kanala. Da bi mreža stvarno mogla da funkcioniše u redundantnom režimu, neophodno je da svaki od kanala sadrži CM uređaj [1], [3].



Sl. 35. Redundantna konfiguracija TTEthernet mreže stepena 2 [3]

6 Genetski algoritam

Genetski algoritam se svrstava u grupu pretraživačkih algoritama, čiji su koncepti zasnovani na principima genetike i prirodne selekcije. Postoji značajan broj problema u računarskim sistemima za koje ne postoje brzi algoritmi namijenjeni za svrhe optimizacije. Generalni pristup optimizacije podrazumijeva formiranje jedinstvenog načina mjerenja i izbora vrijednosti za svrhe traženja optimuma, kao što je npr. potraga za vrijednošću parametra koja rezultuje najboljim performansama posmatranog sistema. Spomenuta radnja obavlja se pomoću tzv. funkcije cijene (eng. *cost function*) koja rezimira dobijena rješenja (eng. *solutions*), te kroz specifične iteracije nastoji da ih optimizuje. Svako od rješenja predstavlja potencijalnog kandidata za optimalno rješenje. Drugim riječima, funkcija cijene analizira prostor rješenja i vrši odluku koje od njih da izoluje iz prostora u cilju pronalaska najboljeg/optimalnog rješenja. Za male prostore rješenja, efikasne su i klasične pretraživačke metode, dok za veće prostore rješenja moraju biti primijenjene specijalne metode. Postoje slučajevi kada nije moguće naći optimalno rješenje, ali kroz iteracije se može asimptotski približavati optimalnom rješenju. Takođe, u prostoru rješenja može postojati više lokalnih maksimuma/minimuma, te je u stvarnim situacijama veoma teško procijeniti koji od njih predstavlja optimalno rješenje. To su tzv. teško-optimizujući problemi (eng. *hard-optimization problems*) i kod njih se primjenjuju algoritmi za pronalazak približno optimalnog rješenja. Neki od ovih pristupa su bazirani na genetskom algoritmu, koji funkcioniše na principima prirodne evolucije [59].

6.1 Osobine genetskog algoritma

Prirodna evolucija predstavlja proces optimizacije određenog problema i zasniva se na analizi problema transliranih u domen populacije problema. Simulacije vršene na računaru pomoću ovog pristupa često rezultuju boljim performansama nego klasične metode, ako se primijene na složene probleme koji zahtijevaju stohastičke tehnike optimizacije. Evolutivne pojave koje se dešavaju kod bioloških organizama su određene haotičnošću, slučajnostima, vremenskim trenucima, kao i nelinearnom interaktivnošću. Spomenuti problemi su izuzetno teški za analizu pomoću klasičnih tehnika optimizacije, te su istraživanja za probleme ove prirode uglavnom orijentisana na primjenu genetskog algoritma [59].

Osnovna ideja genetskog algoritma je otkrivena od strane grupe biologa kada su koristili računare da bi izvršavali simulacije genetskih sistema u prirodi. U ovim simulacijama, kombinovali su 2 ili više hromosoma u cilju formiranja određenih konstrukcija ili ponašanja određenih organizama. Genetski algoritam primjenjiv na računarske sisteme je razvijen 1975. godine na Univerzitetu u Mičigenu, kao rezultat istraživanja Džona Holanda (*John Holland*), njegovih kolega i studenata. Njihov cilj je bio apstrahuju i objasne adaptivne mehanizme genetskih sistema u prirodi, kao i da projektuju vještačke, tj. softverske sisteme koji će oponašati ovakve mehanizme. Drugim riječima, genetski algoritam koristi principe evolucionizma gdje određeni problemi mogu da žive, reprodukuju se i umiru, kao i prirodni organizmi. Iz ovog razloga, terminologija koja se koristi u ovom algoritmu posuđena je iz genetike. S obzirom da se genetski algoritam koristi za pronalazak parametara koji rezultuju optimumom rada sistema, njegovom primjenom u računarskim sistemima može se postići povećanje robustnosti i efikasnosti posmatranog sistema, nezavisno da li se radi o softveru ili hardveru [8]. Genetski algoritam je u stvari jedan pretraživački algoritam (eng. *search algorithm*) koji u datom prostoru rješenja pronalazi ona rješenja koja zadovoljavaju zadate kriterijume. Ukrštanjem osobina novodobijenih rješenja dobija se nova generacija rješenja i ovaj postupak se ponavlja dok se ne dobiju finalni rezultati. Finalni rezultati podrazumijevaju "preživljavanje" samo onih rješenja koja posjeduju najbolje osobine [8], [59].

Kao što smo spomenuli ranije, genetski algoritam može biti koristan u problemima gdje je teško izvršiti optimizaciju pomoću klasičnih analitičkih metoda. To su npr. problemi tipa: rutiranje vodova na štampanim pločama, planiranje rasporeda radnji, prilagodljiva kontrola, igranje igara, problemi transporta, optimizacija upita u velikim bazama podataka, kao i mašinsko učenje. Tokom poslednjih dekada značaj optimizacije sve više raste, iako velika većina inženjerskih problema može biti riješena samo približno u ovom pogledu. Algoritmi među kojima je i genetski algoritam zasnivaju se na vjerovatnoći, te pored direktne pretrage prostora rješenja primjenjuju i stohastičke metode. Još jedna od prednosti genetskog algoritma je što čuva kompletnu populaciju mogućih rješenja, dok ostali pretraživački algoritmi analiziraju samo jedno rješenje izolovano iz posmatrane grupe rješenja, što ga čini robustnijim u odnosu na klasične metode [59]. Genetski algoritam posjeduje sledeće osobine [59]:

- Pretraživačke metode korišćene u genetskom algoritmu mogu se izvršavati paralelno na više mašina, što može značajno ubrzati pretraživački proces.

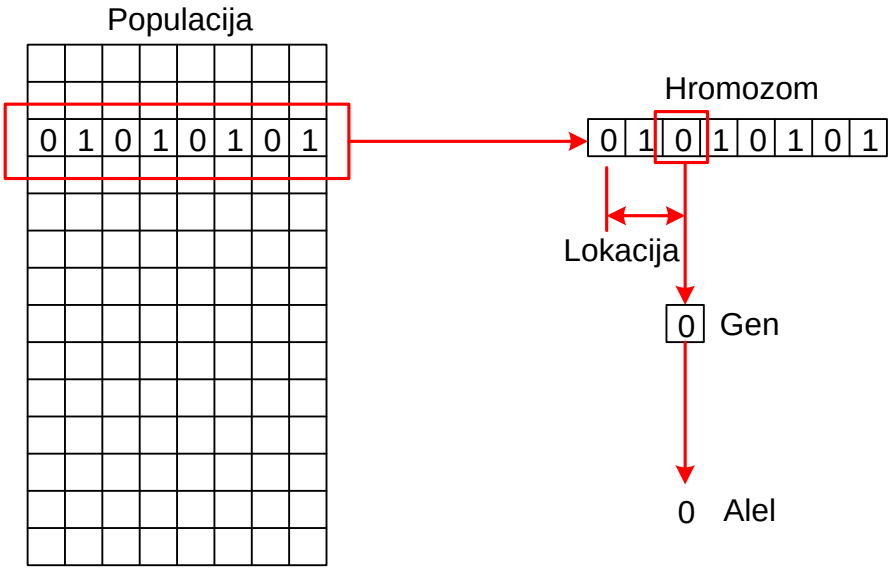
- Genetski algoritam može se primijeniti u kontinualnom, kao i u diskretnom domenu problema.
- Genetski algoritam je stohastičke prirode i ne ostaje zaglavljen na lokalnom optimumu, dok je ovaj problem često izražen kod klasičnih pretraživačkih algoritama.
- Fleksibilnost genetskog algoritma olakšava prepoznavanje optimalnih struktura i parametara u kompleksnim sistemima koji su predmet analize.

S druge strane, praktične primjene genetskog algoritma često ne prate teoriju zbog sledećih razloga [59]:

- Proces kodiranja genetskog algoritma za konkretan problem često translira genetski algoritam u potpuno drugačiji domen nego što je taj sam problem.
- Iako je broj mogućih iteracija (generacija genetskog algoritma) teorijski neograničen, ipak postoje praktična ograničenja.
- Iako je veličina populacije teorijski neograničena, postoje ograničenja i u ovom pogledu, koja su posljedica konfiguracije mašina na kojima se vrše simulacije.

6.2 Terminologija genetskog algoritma

Genetski algoritam koduje svaku instancu u *n*-dimenzionalnom prostoru rješenja najčešće u binarni niz pod nazivom hromosom. Svako rješenje reprezentuje jedan hromosom, kao što je prikazano na Sl. 36. Trenutni skup svih mogućih rješenja datog problema predstavlja populaciju. Hromosom se sastoji od gena koji u datom primjeru mogu da poprime vrijednost 0 ili 1. U terminologiji genetskog algoritma, vrijednost koju poprima gen naziva se alel (gr. *allelos*). Lokacija gena u hromosomu (gr. *locus*) predstavlja poziciju posmatranog gena u hromosomu u odnosu na prvi gen. Nad hromosomima se iterativno primjenjuju različite operacije slične procesu prirodne evolucije u cilju pronalaženja optimalnog hromosoma [8], [59-60].



Sl. 36. Terminologija genetskog algoritma [60]

Kriterijum na osnovu koga se vrši odabir (eng. *selection*) hromosoma iz populacije predstavlja funkciju podobnosti (eng. *fitness function*), a bira se na osnovu kompleksnih matematičkih formula, simulacionih modela, ili u odnosu na parametre dobijene empirijskim metodama. Zadatak funkcije podobnosti je da procijeni koji hromosom u posmatranoj populaciji ima najbolje osobine, te da ga izoluje iz populacije. Često postoji potreba da vrijednost funkcije podobnosti bude pozitivna veličina, tako da se u suprotnom primjenjuju operacije skaliranja ili translacije da bi se to postiglo. Drugi pristup podrazumijeva uvođenje rengiranja hromosoma prema određenom kriterijumu. Prednost ovakvog pristupa je što kriterijum funkcije podobnosti ne mora biti precizno definisan sve dok se rangiranje hromosoma obavlja ispravno. Umjesto samo jednog hromosoma, genetski algoritam čuva kompletnu populaciju koja evoluira kroz iteracije dovodeći do "preživljavanja" samo onih hromosoma koji najbolje zadovoljavaju funkciju podobnosti [59-60] .

6.3 Prevođenje problema u domen genetskog algoritma

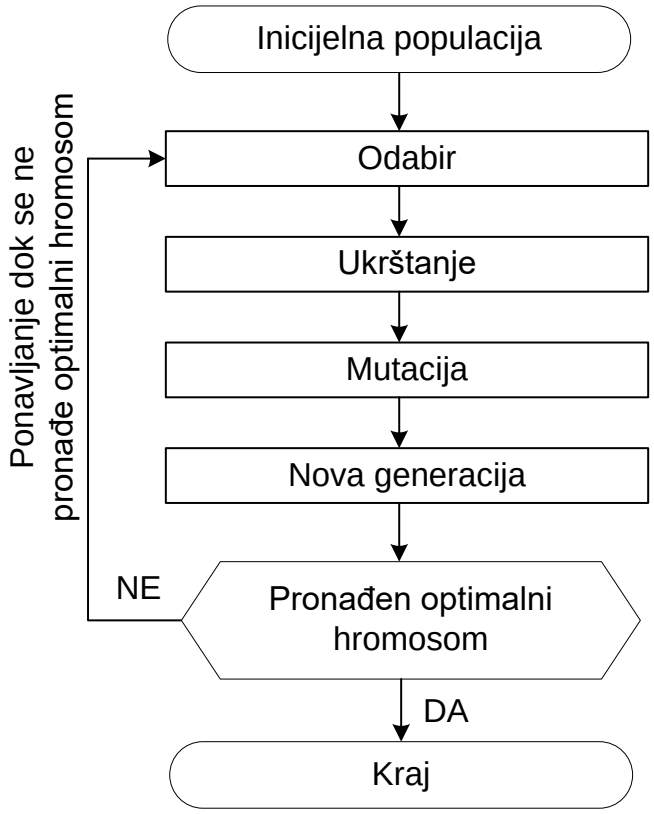
Prvi korak pri projektovanju genetskog algoritma ogleda se u definisanju reprezentacije rješenja za dati problem. Rješenje podrazumijeva bilo koju vrijednost koja može biti kandidat za optimalno rješenje. Tumačenje optimalnog rješenja posmatranog genetskog algoritma prepušta se projektantu, te zavisi od prirode rješenja i oblika rješenja najprihvatljivijeg za primjenu genetskog algoritma. Veoma čest oblik reprezentacije jednog rješenja je niz

karaktera koji pripada ograničenom alfabetu. Veći alfabeti rezultuju većom količinom informacija koja se može predstaviti svakim od karaktera. S druge strane, u većini problema je dovoljno koristiti alfabete manje veličine. U stvarnim problemima, genetski algoritam najčešće koristi binarni alfabet [59].

Kao primjer, posmatraćemo tačku (11, 6, 9) u trodimenzionalnom prostoru rješenja u opsegu [0, 15] za svaku dimenziju. Nakon pretvaranja ove vrijednosti u jednodimenzionalnu binarnu reprezantaciju pogodnu za genetski algoritam, dobija se (11, 6, 9) → (101101101001). U zavisnosti od prirode problema, druge kodne šeme poput Grejovog koda mogu biti takođe korišćene, kao i brojevi u plivajućem zarezu, negativni brojevi i sl. Opisano kodiranje predstavlja translaciju posmatrane problematike u domen okruženja genetskog algoritma. Genetski operatori koji će se primjenjivati nad rješenjima moraju biti projektovani shodno izabranom načinu kodiranja.Veličina inicijalne populacije je takođe veoma bitan parametar za tok genetskog algoritma. Ako je veličina inicijalne populacije premala, to može dovesti do preuranjene konvergencije genetskog algoritma. Drugim riječima, finalno rješenje može biti lokalni umjesto globalnog optimuma. Prevelika veličina inicijalne populacije utiče na neefikasno iskorišćenje računarskih resursa, kao i na dugo vrijeme izvršavanja genetskog algoritma [59].

6.4 Operatori genetskog algoritma

Postoje tri operatora bitna za genetski algoritam: Odabir (eng. *selection*), ukrštanje (eng. *crossover*), i mutacija (eng. *mutation*). Procedura genetskog algoritma ilustrovana je na Sl. 36. Proces primjene spomenutih operatora se ponavlja sve dok se ne pronađe optimalni hromosom, što predstavlja rezultat genetskog algoritma [8], [60].



Sl. 37. Procedura genetskog algoritma [60]

6.4.1 **Odabir**

Odabir je operacija koja određuje koji hromosomi će učestvovati u produkciji nove generacije. Hromosomi se obično biraju prema vjerovatnoći odabira *p* koja je srazmjerna njihovim podobnostima za odabir. Vjerovatnoća odabira se računa kao [59]:

$$p_i = \frac{f_i}{\sum_{k=1}^{n} f_k} \tag{26}$$

gdje je *n* veličina populacije u posmatranoj generaciji, a f_i vrijednost podobnosti *i*-tog hromosoma. Ovakav načina odabira omogućava hromosomima sa natprosječnim podobnostima da se reprodukuju i zamijene hromosome koji imaju podobnosti ispod prosjeka. Na osnovu ovih vjerovatnoća izrađuje se tzv. točak ruleta (eng. *roulette wheel*) sa prorezima srazmjernim podobnostima svakog hromosoma. Postupak izrade točka ruleta je sledeći [59]:

1. Proračun podobnosti $f(v_i)$ za svaki hromosom v_i

2. Proračun zbirne podobnosti F za cjelokupnu populaciju veličine N:

$$F = \sum_{i=1}^{N} f(v_i) \tag{27}$$

3. Proračun vjerovatnoće odabira p_i za svaki hromosom v_i:

$$p_i = \frac{f(v_i)}{F} \tag{28}$$

4. Proračun kumulativne vjerovatnoće q_i za svaki hromosom v_i:

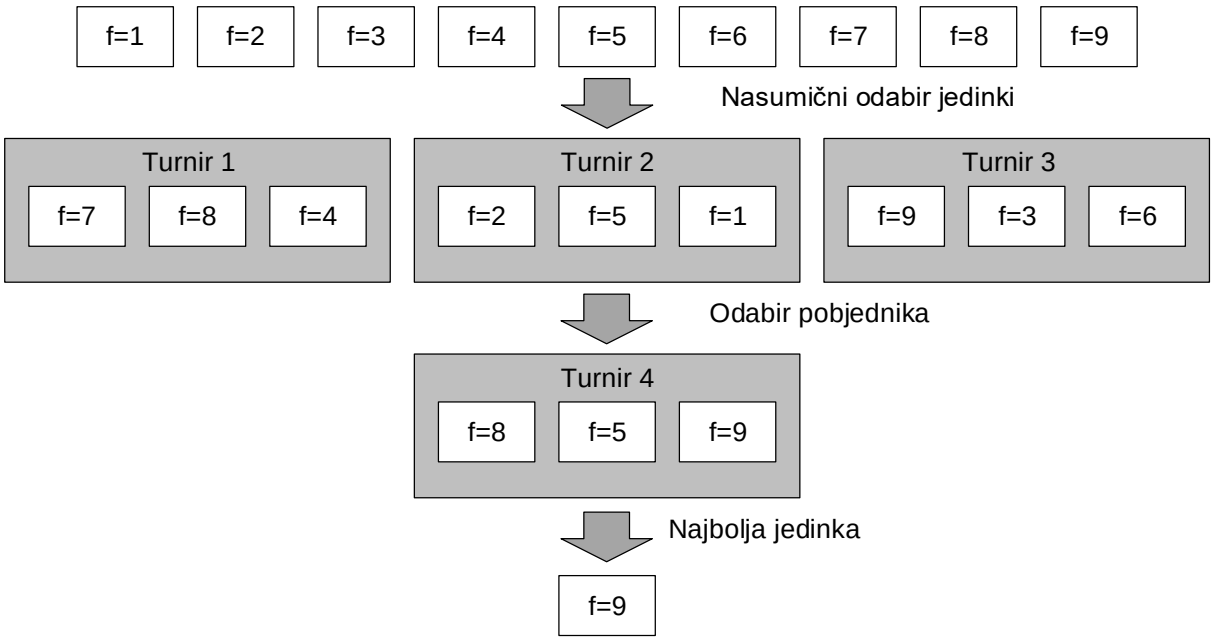
$$q_i = \sum_{j=1}^{i} p_j \tag{29}$$

gdje se q kreće u opsegu od 0 do 1. Vrijednost 1 daje informaciju da su svi hromosomi iz posmatrane populacije uključeni u proračun kumulativne vjerovatnoće. Proces odabira se zasniva na "okretanju" točka ruleta N puta. Svaki put, bira se jedan hromosom iz trenutne populacije koji će biti namijenjen za narednu generaciju. Ovo se može obavljati sledećom procedurom koja se ponavlja N puta [59]:

1. Generiše se proizvoljan broj r iz opsega [0, 1]
2. Ako je $r < q$, bira se prvi hromosom iz populacije v_1, dok se u suprotnom slučaju bira i-ti hromosom v_i takav da je $q_{i-1} < r < q_i$.

Očigledno je da će neki hromosomi biti odabrani više nego jednom, ali to je u skladu sa teorijom. Slijedeći spomenute principe, zadržavaju se dobra rješenja, dok se loša eliminišu [59].

Drugi način odabir je tzv. turnirski odabir. Kada se koristi ova tehnika, nasumično se bira N hromosoma iz populacije, gdje parametar N predstavlja veličinu turnira. Populacija se dijeli u M grupa sa N hromosoma, gdje parametar M predstavlja broj turnira. U svakoj od dobijenih grupa, hromosomi se međusobno takmiče, nakon čega se pobjednik turnira (koji ima najbolju podobnost) prosljeđuje u sledeću generaciju. Turnir se završava kada se kapacitet neke od narednih generacija popuni. Tipična procedura turnirskog odabira je ilustrovana na Sl. 38, gdje je veličina turnira 3. Pošto imamo 9 jedinki, u prvoj fazi imamo 3 različita turnira. Vrijednosti podobnosti (f) su prikazane za svaku jedinku i imaju vrijednosti od 1 do 9 [61-62].



Sl. 38. Primjer turnirskog odabira [61]

Odabir najpodobijeg hromosoma (eng. *Survival-of-the-Fittest Selection*) bira hromosome sa najboljim podobnostima za sledeću generaciju. Korištenjem ovog tipa odabira često dolazi do gubitka raznolikosti populacije, pošto se događa da ponekad i manje podobni hromosomi mogu da sadrže dobre sekvence gena, a takvi hromosomi se uklanjaju tokom procesa odabira [63].

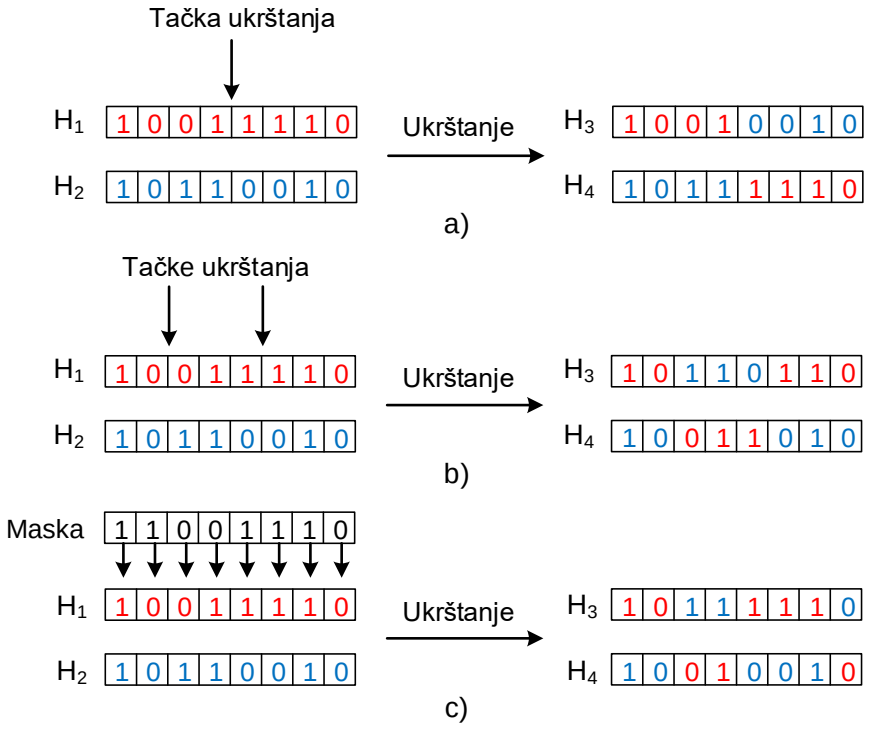
6.4.2 **Ukrštanje**

Ukrštanje podrazumijeva razmjenu osobina od interesa između hromosoma. Parametar koji daje informaciju o broju hromosoma koji učestvuje u proceduri ukrštanja predstavlja vjerovatnoću ukrštanja - *PC*. Izbor hromosoma za ukrštanje iz trenutne populacije može se izvršiti na sledeći način [59]:

1. Generiše se proizvoljan broj *r* iz opsega [0, 1]
2. Ako je $r < PC$, tada se posmatrani hromosom određuje za ukrštanje

Ako je *PC=1,* svi hromosomi iz posmatrane populacije će biti određeni za operaciju ukrštanja, dok će za *PC*=0.5 samo polovina hromosoma biti korišćena za ukrštanje, dok će druga polovina hromosoma ostati nepromijenjena. U procesu ukrštanja obično učestvuju dva hromosoma [59].

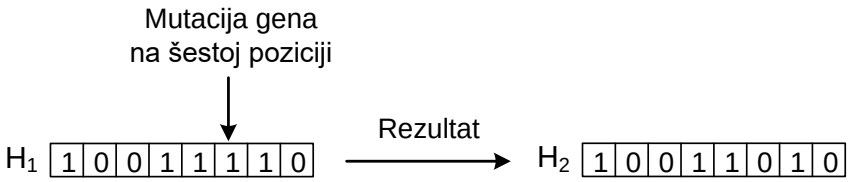
Najjednostavniji vid ukrštanja je ukrštanje u jednoj tački, gdje je ta tačka izabrana na proizvoljnoj lokaciji hromosoma koji vrše razmjenu svojih osobina. Kod ovog pristupa vrši se razmjena segmenata hromosoma koji se nalaze između gena tačke ukrštanja i poslednjeg gena, uključujući i te gene. Drugi vid je ukrštanje u dvije tačke, gdje postoje ograničenja na 2 lokacije u posmatranim hromosomima, a razmjena se vrši u prostoru između te 2 tačke. Spomenuti vidovi ukrštanja ilustrovani su primjerom na Sl. 39. Prikazano je ukrštanje između dva hromosoma H_1 i H_2, nakon čega su kao rezultat dobijeni novi hromosomi H_3 i H_4. Shodno ukrštanju u jednoj ili dvije tačke, može se definisati i ukrštanje u *n* tačaka, gdje se vrši zamjena alela gena između tačaka 1 i 2, 3 i 4, ..., *n*-1 i *n*. Kod uniformnog ukrštanja, za svaki par hromosoma izrađuje se proizvoljna maska koja sadrži binarne cifre, kao što je prikazano na Sl. **39** c). Ako je binarna cifra na određenoj poziciji jednaka jedinici, tada hromosom H_3 uzima vrijednost od hromosoma H_1 sa te pozicije, a hromosom H_4 uzima vrijednost od hromosoma H_2 sa iste pozicije. U suprotnom, ako je vrijednost maske na posmatranoj pozicije jednaka nuli, tada hromosom H_3 uzima vrijednost hromosoma H_2 sa te pozicije, dok hromosom H_4 uzima vrijednost sa hromosoma H_1 sa iste pozicije [59].



Sl. 39. Primjer operacije ukrštanja između 2 hromosoma: a) Ukrštanje u jednoj tački, b) Ukrštanje u dvije tačke, c) Uniformno ukrštanje [59]

6.4.3 **Mutacija**

Operacija ukrštanja koristi trenutne potencijale hromosoma, ali to nekad nije dovoljno da bi se pronašlo optimalno rješenje. Iz tog razloga pristupa se operaciji mutacije, koja u hromosome utiskuje nove informacije koje mogu dovesti do pronalaska optimalnog rješenja. Uobičajeni vid mutacije nad binarnim hromosomima je promjena vrijednosti alela gena. Važnost operatora mutacije ogleda se u činjenici da sprječava stagnaciju ili konvergenciju genetskog algoritma prema lokalnom optimumu. Primjer operacije mutacije dat je na Sl. 40 [59].



Sl. 40. Primjer operacije mutacije [59]

U ovom poglavlju dati su primjeri implementacija genetskih operatora odabira, ukrštanja i mutacije. U zavisnosti od prirode konkretnog problema u stvarnom svijetu, implementacija spomenutih operacija može da bude drugačija nego što je predstavljena iznad, ali uglavnom genetski algoritam kod svakog pristupa koristi proceduru ilustrovanu na Sl. 37.

7 Simulacija TTEthernet mreže

U ovom poglavlju ukratko je opisan postupak razvoja simulatora namijenjenog za simulaciju *TTEthernet* mreže, kao i rezultati simulacija za slučajeve komutatora pod otkazom, te krajnjeg uređaja pod otkazom.

7.1 Razvoj simulatora u okruženju *OMNeT++*

OMNeT++ [7] je razvojno okruženje otvorenog koda za projektovanje simulatora telekomunikacionih i računarskih mreža koje je javno dostupno od 1997. godine. Takođe, ovdje je moguće vršiti i simulacije ponašanja projektovane mreže na osnovu projektovanih modela pojedinih mrežnih komponenata kao što su mrežne kartice, komutatori, ruteri, itd. Kao jezik za projektovanje modela u ovom okruženju, koristi se programski jezik C++. Takođe, sve postojeće biblioteke namijenjene za okruženje *OMNeT++* napisane su u programskom jeziku C++. Spomenuti alat je u stvari simulator diskretnih događaja (eng. *Discrete Event Simulator - DES*), tj. modelovanje i simulacije se odnose na diskretne trenutke u kojim će određeni događaj biti izvršen. Prema Rouse [64], DES je proces opisivanja ponašanja nekog kompleksnog sistema kao sekvence precizno definisanih događaja. U odnosu na DES, događaj predstavlja promjenu stanja posmatranog sistema u precizno definisanom trenutku. Dostupni su različiti modeli *OMNeT++* komponenata pomoću kojih je moguće simulirati širok spektar različitih tipova mreža, kao što su LAN (eng. *Local Area Network*), WAN (eng. *Wide Area Network*), bežične (eng. *wireless*) mreže, *ad-hoc* mreže, *peer-to-peer* mreže, senzorske mreže, mreže za čuvanje podataka (eng. *Storage Area Network*), optičke mreže i slično. Parametri projektovanih modula kao i topologija simulirane mreže definišu se kroz poseban jezik – NED (eng. *Network Description*) [65].

Simulator koji je korišten za potrebe ove teze je projektovan od nule uz korištenje samo standardnih *OMNeT++* biblioteka i sadrži oko 3000 linija C++ koda. Projektovani modeli SM i CM komponenata koje su korištene u simulacijama u potpunosti oponašaju mašine stanja date u standardu SAE AS6802, od kojih je mašina stanja za SM komponentu prikazana na Sl. 29. Simulator nije sertifikovan, ali su rezultati dobijeni simulacijama pregledani od strane projektanata *TTEthernet* protokola u firmi TTTech Computertechnik AG, Vienna. Vrijednosti parametara modela korišćenih u simulacijama su preuzete kao rezultat alata TTE Tools v5.0.

Spomenuti alat je komercijalni alat firme TTTech Computertechnik AG, Vienna, koji je namijenjen za planiranje, projektovanje i podešavanje *TTEthernet* mreža [66]. Iste vrijednosti parametara mogu se primijeniti na sve *TTEthernet* mreže otporne na jedan otkaz, sa jednim komutatorom u redundantnoj konfiguraciji i proizvoljnim brojem SM uređaja, pri čemu je njihov minimalni broj 4. Parametri krajnjeg uređaja koji se koriste u našem modelu su sledeći:

- `sync_role`
- `smc_scheduled_pit`
- `es_listen_timeout`
- `es_coldstart_timeout`
- `ca_round_trip`
- `es_ca_acceptance_window_half`
- `es_ca_offset`
- `es_cs_offset`
- `es_restart_timeout_async`
- `es_restart_timeout_sync`
- `es_restart_timeout`
- `es_num_stable_cycles`
- `es_num_unstable_cycles`
- `es_tent_to_stable_enabled`
- `stable_ca_enabled`
- `es_sync_to_stable_enabled`
- `es_goto_inactive_moment`
- `es_inactive_duration`
- `clock_drift`
- `FIRST_STATE`

Većina spomenutih parametara je već objašnjena u sekciji 5.3.2 kao i u standardu SAE AS6802, dok će ostali naknadno dodati parametri biti objašnjeni u nastavku. Parametar `sync_role` definiše da li je posmatrani krajnji uređaj podešen da radi kao SM ili SC uređaj. Uloga CM uređaja trenutno nije realizovana u trenutno razvijenom modelu krajnjeg uređaja. Kroz odgovarajući konfiguracioni fajl moguće je definisati u koje stanje SM uređaj

ulazi po uključenju, što je definisano parametrom `FIRST_STATE`. Na ovaj način moguće je simulirati kako se uspostavlja vremenska usklađenost u posmatranoj mreži ukoliko se uređaji nalaze u proizvoljnim stanjima.

Takođe, moguće je simulirati isključenje uređaja u proizvoljnom trenutku što je postignuto uvođenjem parametara `es_goto_inactive_moment` i `es_inactive_duration`. Za ove svrhe uvedeno je jedno dodatno stanje – INACTIVE. Uređaj se isključuje u trenutku definisanim parametrom `es_goto_inactive_moment`, a trajanje isključenog stanja je određeno parametrom `es_inactive_duration`. Drugim riječima, pomoću spomenuta dva parametra definisani su momenti kada uređaj ulazi i izlazi iz INACTIVE stanja.

Vrijednost odstupanja lokalnog časovnika modelovana je korišćenjem parametra `clock_drift` kojim se definiše za koliko *ns* lokalni časovnik uređaja odstupa od referentnog časovnika po jednoj μs. Moment narednog događaja u odnosu na trenutni moment modelovan je na sledeći način:

$$T_{naredni} = T_{trenutni} + D\left(1 + T_{odstupanja}\right) + T_{korekcije} \tag{30}$$

gdje je parametar *D* vremensko rastojanje narednog događaja u odnosu na trenutni događaj, $T_{odstupanja}$ vrijednost odstupanja lokalnog časovnika u odnosu na referentni časovnik, a $T_{korekcije}$ količina korekcije izračunata na osnovu primljenog komprimovanog IN okvira. Bitno je primjetiti da parametri $T_{odstupanja}$ i $T_{korekcije}$ mogu poprimiti kako pozitivne, tako i negativne vrijednosti.

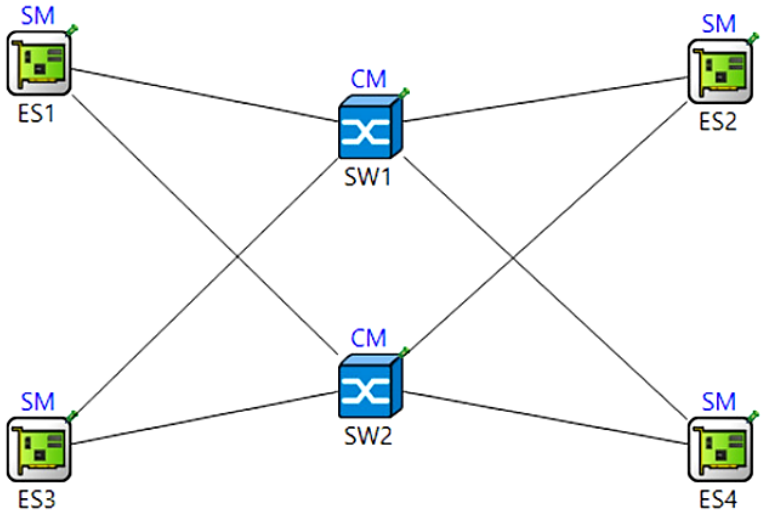
Postoje dvije funkcije koje su neophodne za modelovanje bilo koje komponente u okruženju *OMNeT++*: `initialize()` i `handle_message()`. Funkcija `initialize()` postavlja parametre posmatrane komponente na vrijednosti definisane u odgovarajućem INI fajlu. Mašina stanja komponente takođe je inicijalizovana u ovoj funkciji. Realizacija mašine stanja izvršena je u okviru `handle_message()` funkcije, koja se poziva svaki put kada posmatrani modul primi poruku. Poruke se dijele na dvije vrste: sopstvene (eng. *self-messages*), te poruke primljene od drugih modula. Sopstvene poruke su generisane i konzumirane od strane iste komponente, tj. komponenta (SM/SC/CM) generiše takvu poruku za svoje potrebe. Ove poruke se ne emituju na spoljašnjem priključku komponente. Primjena sopstvenih poruka je u generisanju pauza, zakazivanju određenih

događaja u precizno određenom trenutku (npr. izvrši otvaranje prozora prijema za 10 μs u odnosu na sadašnji trenutak, pošalji poruku u precizno definisanom trenutku i slično). Koncept primjene sopstvenih poruka je takav da se zakaže njeno slanje nakon predefinisanog vremenskog intervala. Nakon isteka spomenutog vremenskog intervala, komponenta šalje sopstvenu poruku i u istom trenutku je i prima. Prijem sopstvene poruke rezultuje pozivanjem `handle_message()` funkcije u kojoj se ispituje da li se radi o sopstvenoj poruci ili o poruci primljenoj na spoljašnjem priključku poslanoj od strane druge komponente. U zavisnosti od utvrđenog tipa poruke, `handle_message()` funkcija preduzima odgovarajuće radnje (otvaranje/zatvaranje prozora prijema, slanje neke druge poruke, tranzicija komponente u drugo stanje, itd.).

7.2 Postavka simulacije

Sve simulacije date u ovom radu su izvršene korišćenjem alata *OMNeT++* verzija 5.1.1, na operativnom sistemu Ubuntu 18.04 LTS, CPU Intel Core i7, 16Gb RAM. S obzirom da je bilo neophodno izvršiti veliki broj simulacija, njihovo izvršavanje je automatizovano pomoću odgovarajućih *Python* skripti. Tipovi PCF okvira koji se emituju, njihov period, specifičnosti svake iteracije slanja se kontrolišu pomoću spomenutih skripti, kao i parsiranje rezultata. Pod uspostavljenom usklađenim stanjem smatramo trenutak kada je posmatrana komponenta (SM/CM) ušla u stanje SM_STABLE/CM_STABLE. *Python* skripta pretražuje datoteku dnevnika (eng. *log file*) generisanu od strane alata *OMNeT++* u toku simulacije, te u njoj traži moment kada je komponenta ušla u vremenski usklađen mod rada. Na početku simulacije, sve komponente komponente kreću od SM_INTEGRATE/CM_INTEGRATE stanja. Simulirana je redundantna topologija sa stepenom redundanse 2 i ulogama uređaja, kao što je prikazano na Sl. 41. Pošto *TTEthernet* funkcioniše na drugom sloju OSI modela, mrežu ovog tipa čine isključivo komutatori i krajnji uređaji. Shodno tome, otkaz može da se dogodi na nekoj od ovih komponenata, ako isključimo slučaj otkaza na vodovima koji povezuju spomenute komponente. Teza pokriva simulacije oba slučaja - slučaj kada je komutator SW2 pod otkazom i na jednom od svojih priključaka nekontrolisano emituje PCF okvire, te slučaj kada je uređaj ES4 pod otkazom i takođe nekontrolisano emituje različite sekvence PCF okvira. Uloge u procesu vremenskog usklađivanja uređaja su prikazane iznad svakog od njih na Sl. 41. S obzirom da uređaji sa SC ulogom nemaju uticaja na tok vremenskog

usklađivanja, oni nisu uzeti u razmatranje. Pošto kvalitet vremenskog usklađivanja nije od interesa u ovom istraživanju, svi lokalni časovnici u uređajima su podešeni da budu idealni.



Sl. 41. Redundantna topologija sa stepenom redundanse 2

7.2.1 Postavka simulacije za slučaj kada je komutator pod otkazom

Uspostavak vremenske usklađenosti ili faza pokretanja mreže (hladni start) je veoma bitna faza u procesu usklađivanja vremena. Ako posmatramo npr. let aviona koji traje 5 sati a interval ponovnog usklađivanja je podešen na 1 ms, u normalnom režimu rada vremensko usklađivanje će se izvršiti 18 miliona puta, dok će se uspostavak vremenske usklađenosti izvršiti samo jednom. Očigledno da bez uspješno obavljene faze pokretanja mreže uređaji uopšte neće moći da uđu u normalni režim rada, što govori o izuzetnom značaju ove faze.

Prva grupa simulacija u ovoj tezi se bavi negativnim uticajem komutatora pod greškom na uspostavak vremenske usklađenosti krajnjeg uređaja koji je povezan priključak tog komutatora. Analiziran je slučaj kada je komutator SW2 pod otkazom, tj. priključak na koji je povezan ES4 uređaj je pod otkazom dok su ostali priključci komutatora ispravni. Komutator SW2 šalje čiste sekvence CS, CA ili IN okvira na priključku zahvaćenim otkazom. S obzirom da se radi o topologiji koja može da toleriše samo jedan uređaj pod otkazom (nezavisno da li se radi o komutatoru ili krajnjem uređaju), u čitavoj posmatranoj mreži je samo komutator pod otkazom dok su svi ostali uređaji ispravni. Odatle, komutator SW2 ima uticaja samo na uspostavljanje vremenske usklađenosti uređaja ES4. Konfiguracija posmatrane mreže je takva da je potrebno minimalno 3 SM uređaja koji nisu pod otkazom (ES1, ES2, ES3), da bi vremenska usklađenost u mreži uopšte mogla da se uspostavi. U svim simulacijama trajanje integracionog ciklusa je podešeno na 1 ms.

Što se tiče regularnog uspostavka vremenske usklađenosti na SM uređajima kada su sve komponente u mreži ispravne, simulacije pokazuju da vrijeme neophodno da krajnji uređaj uđe u SM_STABLE stanje iznosi 5489 μs. U ovom slučaju, krajnji uređaj sa SM ulogom prolazi kroz sledeća stanja (Sl. 29): SM_INTEGRATE → SM_UNSYNC → SM_FLOOD → SM_WAIT_4_CYCLE_START_CS → SM_TENTATIVE_SYNC → SM_SYNC → SM_STABLE. U simulacijama ovog tipa ciljano su emitovani određeni tipovi PCF okvira prema SM uređaju ES4, u trenucima dok se on nalazi u stanjima na koje ti okviri imaju uticaja. Npr. posmatrajući mašinu stanja sa Sl. 29 možemo vidjeti da CS okviri uopšte nemaju uticaja na rad SM uređaja ako se on nalazi u SM_STABLE stanju, dok CS okvir primljen u SM_WAIT_4_CYCLE_START_CS stanju uzrokuje tranziciju u niže stanje SM_FLOOD [2].

7.2.2 Postavka simulacije za slučaj kada je krajnji uređaj pod otkazom

Za razliku od prethodnog slučaja, ovaj slučaj se bavi uticajem krajnjeg uređaja pod greškom koji emituje različite sekvence PCF okvira, na uspostavak vremenske usklađenosti u ostatku mreže. Takođe, ovdje je analiziran najgori slučaj kada krajnji uređaj pod greškom (ES4 uređaj u našem slučaju) utiče na uspostavak vremenske usklađenosti, tj. vršena je procjena koja sekvenca različitih tipova PCF okvira (CS/CA/IN) emitovana od strane uređaja ES4, zajedno sa periodom između njih, utiče na najduže trajanje faze pokretanja (najgori slučaj) posmatrane *TTEthernet* mreže. Za ove svrhe poslužio je genetski algoritam koji smo opisali u poglavlju 6. PCF okviri emituju se na oba kanala ES4 uređaja, tj. imaju uticaja na oba komutatora u posmatranoj mreži (SW1 i SW2). Simulacije pokazuju da se nezavisno od sekvence i perioda između PCF okvira usklađivanje vremena u posmatranoj mreži uspješno uspostavlja, tj. ES4 uređaj utiče samo na trajanje faze hladnog starta, dok se uspostavak vremenske usklađenosti ne dovodi u pitanje.

7.3 Metodologija simulacije za slučaj kada je krajnji uređaj pod otkazom

Različite kombinacije CS/CA/IN okvira kao i periodi između njih, formiraju hromosome. Svaki hromosom sadrži konačan broj gena čiji aleli poprimaju vrijednosti CS, CA, IN (tip PCF okvira), kao i period između okvira u sekvenci. Primjer ovakvog hromosoma dužine 16 gena prikazan je na Sl. 42. Sekvence ovakvih okvira se šalju sa

krajnjeg uređaja pod greškom prema komutatoru, te imaju uticaja na rad kompletne mreže. Nakon što je kompletna prikazana sekvenca poslana, slanje se vrši iznova u petlji tokom čitavog vremena trajanja simulacije.

CA	150µs	IN	572µs	CS	80µs	CS	320µs	CA	744µs	IN	289µs	CA	114µs	IN	114µs

Sl. 42 Primjer hromosoma dužine 16 gena

Procedura primjene genetskog algoritma na procjenu najgoreg slučaja za uspostavak vremenske usklađenosti u posmatranoj *TTEthernet* mreži ilustrovana je u tabeli 2.

Tabela 2. Relacija između genetskog algoritma i *TTEthernet* mreže.

Genetski algoritam	***TTEthernet***
Generisana početna populacija hromosoma sa proizvoljno odabranim alelima gena.	ES4 uređaj emituje promjenljive sekvence PCF okvira sa promjenljivim periodom između okvira.
Odabir najpodobnijih hromosoma iz populacije koji će biti namijenjeni za operaciju ukrštanja.	Izdvajanje konačne grupe hromosoma koji uzrokuju najduža vremena hladnog starta mreže.
Primjena operacije ukrštanja hromosoma odabranih u prethodnom koraku.	Međusobno kombinovanje dijelova PCF sekvenci koje uzrokuju najduža vremena hladnog starta mreže.
Primjena operacije mutacije na malom broju hromosoma dobijenim operacijom ukrštanja.	Za mali broj dobijenih PCF sekvenci vrši se izmjena na alelima gena (tip PCF frejma ili period unutar sekvence).
Dobijena nova populacija hromosoma (sledeća generacija).	ES4 uređaj emituje nove sekvence PCF okvira iz skupa sekvenci dobijenih prethodno navedenim manipulacijama.
Ponavljanje prethodno navedenih koraka dok se ne pronađe optimalni hromosom.	Ponavljanje prethodno navedenih koraka dok se ne pronađe ona PCF sekvenca koja uzrokuje najduže trajanje hladnog starta*TTEthernet* mreže.
●●●●●●●●●●●	●●●●●●●●●●●
Pronađen optimalni hromosom za zadate parametre genetskog algoritma - genetski algoritam se zaustavlja.	Najduže vrijeme hladnog starta dobijeno za nekoliko poslednjih iteracija se ne produžava. Za zadate parametre genetskog algoritma, posmatrana sekvenca ima najveći uticaj na hladni start mreže.

Da bi se pronašao najgori slučaj trajanja faze pokretanja (eng. *the worst-case startup time*) pomoću genetskog algoritma, potrebno je generisati ulazne podatke za genetski algoritam. To u stvari predstavlja skup proizvoljno generisanih hromosoma nad kojima će kasnije da se vrše operacije odabira, ukrštanja i mutacije. Drugim riječima, potrebno je napraviti početnu populaciju.

Prvi sledeći korak je da se izabere konačan broj najpodobnijih hromosoma iz početne početne populacije koji će biti namijenjeni za ukrštanje. Podobnost hromozoma određuje se po principu vjerovatnoće korištenjem točka ruleta spomenutog ranije, gdje se veće podobnosti dodjeljuju hromosomima iz početne populacije koji uzrokuju najduža vremena hladnog starta. Pored ovakvih hromosoma, bira se i mali broj hromosoma koji nemaju visoku vrijednost podobnosti, da bi se očuvala raznovrsnost sledeće generacije, a samim tim spriječila i preuranjena konvergencija ka lokalnom optimumu. Nad skupom hromosoma dobijenim operacijom odabira, dalje se vrši njihovo međusobno ukrštanje, kao i mutacije nad veoma malim skupom hromosoma. Za ove operacije potrebno je definisati koji tip ukrštanja se koristi (npr. da li u jednoj ili u dvije tačke), kao i učestanost mutacije.

Kompletan opisani postupak se ponavlja, dajući svaku novu generaciju kvalitetnijom od prethodne. Pod kvalitetom populacije može se smatrati suma vremena hladnog starta koju daju svi hromosomi u posmatranoj populaciji. Odatle, populacija se može smatrati kvalitetnijom ako je suma spomenutih vremena veća. Algoritam se zaustavlja ako u poslednjih nekoliko generacija nije pronađen hromosom koji uzrokuje duže vrijeme hladnog starta. Sledeći korak je da se simulacije izvrše ponovo više puta sa drugačijim vrijednostima parametara genetskog algoritma. Na kraju, vrši se kombinovanje vrijednosti parametara koji su dali najbolje rezultate u svakoj simulaciji, u cilju pronalaženja optimalnih vrijednosti parametara genetskog algoritma, te se sa tim optimalnim parametrima ponovo puštaju simulacije. Rezultati dobijeni sa ovakvim parametrima genetskog algoritma služe za procjenu najdužeg vremena neophodnog za uspostavljanje usklađenosti časovnika, u slučaju da u mreži postoji krajnji uređaj pod otkazom.

Veoma je teško unaprijed predvidjeti ponašanje mreže kada u njoj postoji krajnji uređaj pod otkazom. Uticaj komutatora pod otkazom na krajnji uređaj je relativno lako predvidjeti jer je potrebno analizirati mašinu stanja samo jednog uređaja - SM ili SC uređaja. Međutim, kada u mreži postoji krajnji uređaj pod otkazom, analiza je mnogo složenija, jer je potrebno paralelno pratiti ponašanje mašina stanja svih uređaja u mreži, uključujući slanja i

prosljeđivanja okvira, kao i vremenska odlaganja (eng. *timeout*). Stoga, daleko je isplativije vršiti ovakve analize putem simulacija, nego analitičkim putem.

Za razliku od prethodnog slučaja gdje su u precizno definisanim trenucima emitovani okviri od komutatora pod otkazom prema SM uređaju, u ovim simulacijama SM uređaj pod greškom (ES4 uređaj) od početka simulacije počinje da šalje različite sekvence PCF okvira prema komutatorima na koje je povezan. Sekvence okvira emitovanih sa uređaja ES4 se izrađuju pod uticajem genetskog algoritma. Takođe, simulacije se vrše na topologiji datoj na Sl. 41, a trajanje integracionog ciklusa je podešeno na 1000 μs. Parametri genetskog algoritma korišteni za ove simulacije dati su u tabeli 3 i smatramo ih podrazumijevanim parametrima.

Tabela 3. Podrazumijevane vrijednosti parametara genetskog algoritma.

Parametar	**Vrijednost**	**Opis**
Trajanje integracionog ciklusa	1 ms	Period između IN okvira.
Veličina generacije	3000	Broj hromosoma u populaciji.
Veličina reprodukcije	1000	Broj hromosoma namijenjen za ukrštanje.
Dužina hromosoma	20	Broj gena u hromosomu.
Minimalni period	0 μs	Minimalni mogući period između 2 PCF okvira.
Maksimalni period	500 μs	Maksimalni mogući period između 2 PCF okvira.
Tip odabira	Turnir	Mogući tipovi: Rulet, Turnir, Najpodobniji
Tip ukrštanja	2 tačke	Mogući tipovi: 1 tačka, 2 tačke, uniformno
Vjerovatnoća mutacije	5%	Vjerovatnoća mutacije gena.
Broj mutiranih gena	1	Broj gena koji mogu biti pod uticajem mutacije.

7.4 Rezultati simulacija

U ovoj sekciji su dati rezultati simulacija kada je komutator pod otkazom (sekcije 7.4.1 - 7.4.3), te kada je krajnji uređaj pod otkazom (sekcije 7.4.4 i 7.4.5).

7.4.1 Slučaj kada komutator SW2 emituje sekvencu CS okvira prema uređaju ES4

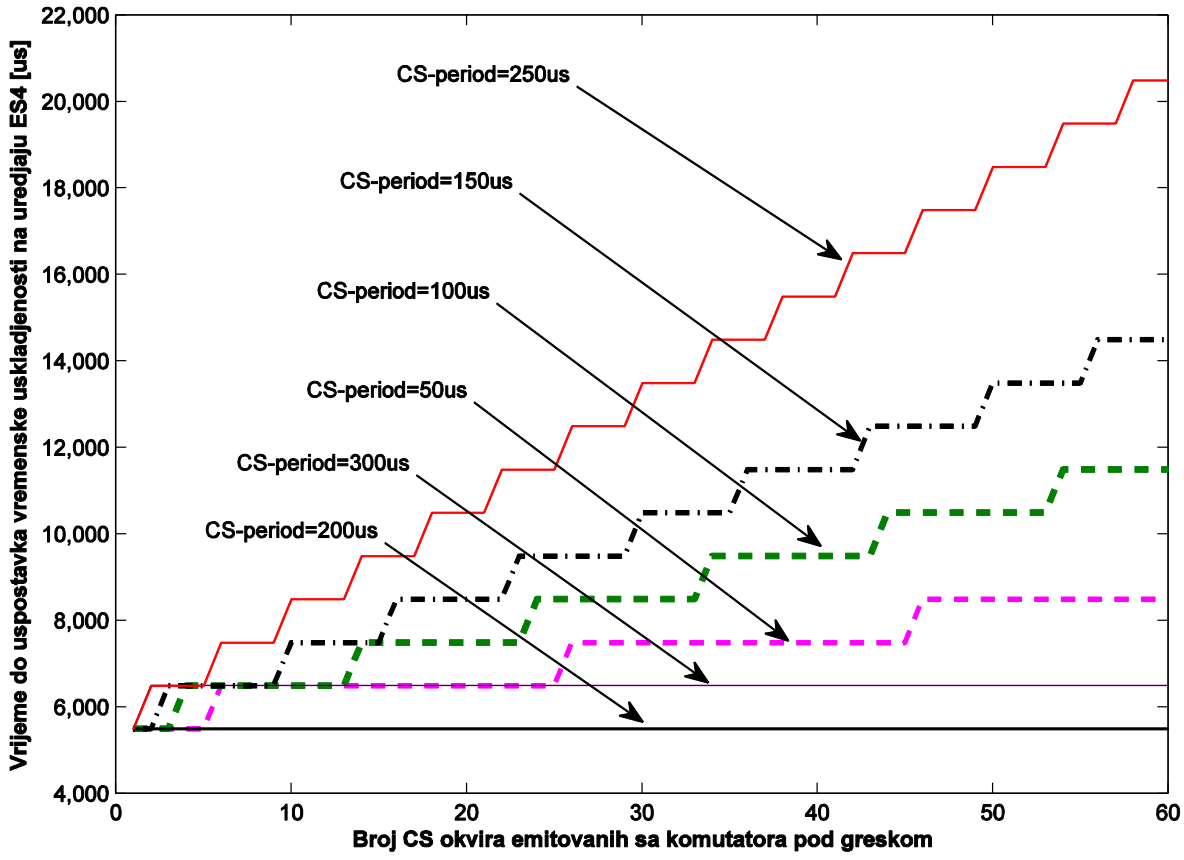
CS okviri imaju uticaja na uobičajeni proces usklađivanja vremena SM uređaja dok se nalazi u nekom od stanja: SM_WAIT_4_CYCLE_START_CS, SM_UNSYNC ili SM_FLOOD. Uređaj sa SM ulogom filtrira CS okvire dok se nalazi u drugim stanjima, te ih ne uzima u obzir.

Ako posmatramo SM_WAIT_4_CYCLE_START_CS stanje, kao što je rečeno ranije, CS okvir primljen u ovom stanju uzrokovaće prelazak u SM_FLOOD stanje u kojem ES4 uređaj emituje CA okvir. Treba imati u vidu da ostatak mreže uredno nastavlja proces hladnog starta. Komutator SW1 ignorisaće primljeni CA okvir iz razloga što se u tom trenutku on nalazi u stanju u kojem filtrira CS i CA okvire. Pošto ES4 uređaj ne dobija odgovor od komutatora SW1, on dalje prelazi u stanje SM_UNSYNC gdje počinje sa slanjem CS okvira u jednakim vremenskim intervalima definisanim parametrom `sm_coldstart_timeout`. Spomenuti mehanizam filtriranja na komutatoru SW1 će takođe ignorisati i ove okvire. Nakon prijema prvog validnog komprimovanog IN okvira od komutatora SW1, ES4 uređaj će prijeći u stanje SM_UNSYNC, praćeno stanje SM_STABLE nakon dovoljnog broja stabilnih integracionih ciklusa. Ukratko, SM uređaj će proći kroz sledeća stanja kada primi CS okvir u SM_WAIT_4_CYCLE_START_CS stanje: SM_INTEGRATE → SM_SYNC → SM_FLOOD → SM_WAIT_4_CYCLE_START_CS → SM_FLOOD → SM_UNSYNC → SM_SYNC → SM_STABLE. Opisani slučaj produžava vrijeme hladnog starta za trajanje jednog integracionog ciklusa [2].

Što se tiče SM_UNSYNC stanja, primljeni CS okvir uzrokuje prijevremenu tranziciju u više stanje – SM_FLOOD. Ovo međutim ne doprinosi bržem uspostavku sinhronog stanja na ES4 uređaju, pošto SM uređaj ne šalje odmah svoje CA okvire po ulasku u SM_FLOOD stanje. U međuvremenu, CM uređaj koji nije pod greškom proslijeđuje CS okvir drugog SM uređaja prema uređaju ES4, što rezultuje ponovnim ulaskom uređaja ES4 u SM_FLOOD stanje. U momentu prijema spomenutog CS okvira, ostali SM uređaji u mreži će uspostaviti SM_FLOOD stanje, što znači da će svi SM uređaji (uključujući i ES4) emitovati svoj CA okvir u istom trenutku. Rezultujuće trajanje faze hladnog starta ES4 uređaja u ovom slučaju nije produženo, nego je jednako trajanju regularnog hladnog starta [2].

CS okvir primljen u SM_FLOOD stanju uzrokuje ponovni ulazak u SM_FLOOD stanje. Nakon toga, događaju se iste tranzicije kao i kod slučaja za SM_WAIT_4_CYCLE_START_CS stanje: SM_FLOOD → SM_UNSYNC → SM_SYNC → SM_STABLE [2].

U slučaju da SW2 emituje sekvencu CS okvira, trajanje faze hladnog starta ES4 uređaja će biti produženo u zavisnosti od dužine te sekvence. Komutator SW2 spomenutu sekvencu počinje da emituje u trenutku dok je ES4 uređaj blizu isteka SM_FLOOD stanja. Svaki primljeni CS okvir u SM_FLOOD stanje uzrokuje da ES4 uređaj ponovo ulazi u SM_FLOOD stanje i to se ciklički ponavlja sve dok se sekvenca CS okvira ne završi. Rezultati simulacija za različite dužine sekvenci CS okvira, te različite periode između okvira, dati su na Sl. 43 [2].



Sl. 43. Vrijeme do uspostavka vremenske usklađenosti (trajanje faze hladnog starta) na uređaju ES4. Slučaj kada CM uređaj pod greškom emituje sekvence CS okvira dok je ES4 uređaj u SM_FLOOD stanju [2]

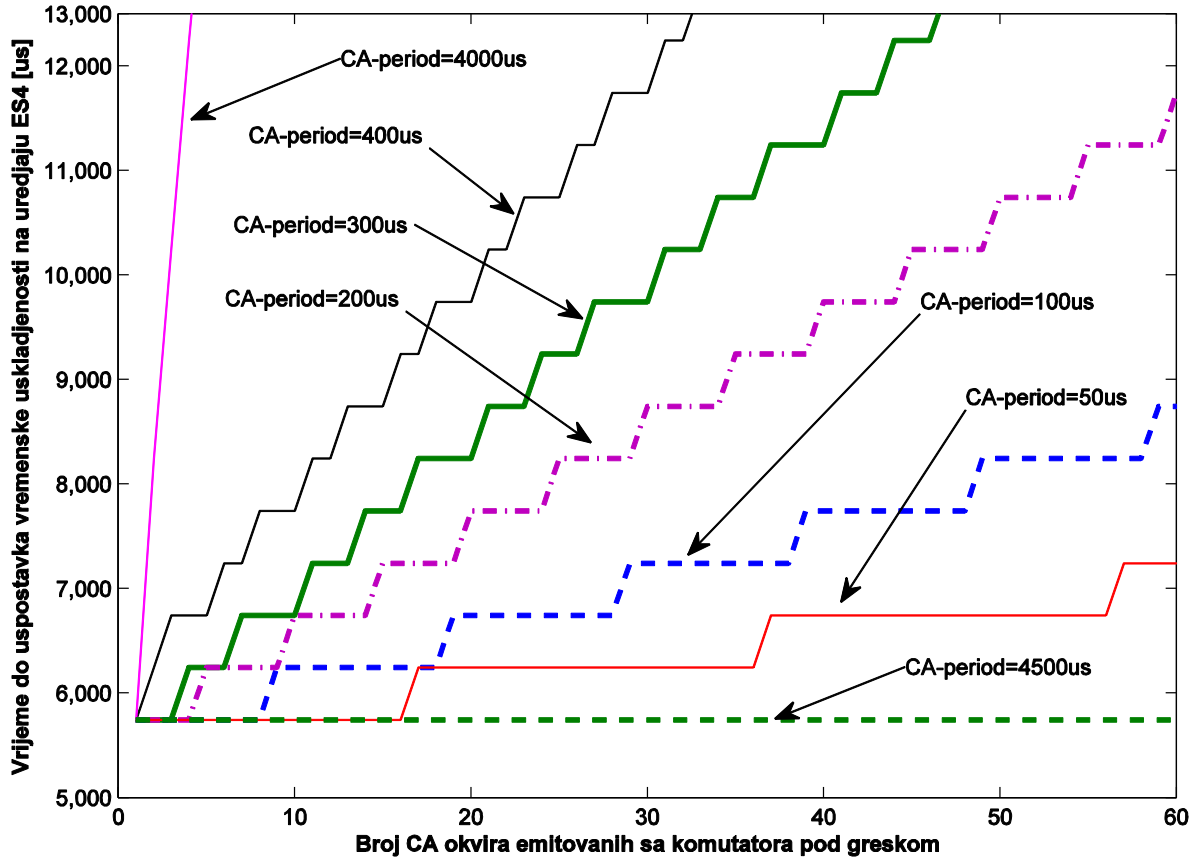
Posmatrajući rezultate sa Sl. 43, primjećujemo da sekvence perioda 200 μs nemaju uticaja na proces usklađivanja vremena uređaja ES4, tj. vrijeme do uspostavka vremenske usklađenosti iznosi 5489 μs što je jednako regularnom trajanju uspostavka usklađenog vremena. Kod perioda 300 μs, vidimo da samo prvi okvir utiče na produženje faze hladnog starta i to za vrijednost trajanja jednog integracionog ciklusa (5849 μs + 1000 μs = 6849 μs). Za ostale periode između CS okvira vidimo da karakteristike nisu kontinualne nego su stepenaste, a inkrementi karakteristika su diskretni u skokovima jednakim trajanjima integracionih ciklusa (5489 μs, 6489 μs, 7489 μs, 8489 μs, itd.). Ako posmatramo npr. karakteristiku koja se odnosi na period sekvence jednak 100 μs, za dužine sekvenci između 24 i 33 CS okvira vrijeme do uspostavka vremenske usklađenosti na ES4 uređaju je uvećano za vrijednost trajanja 3 integraciona ciklusa, tj. jednako je 8489 μs. Razlog stepenaste karakteristike leži u poretku tranzicija stanja. Nakon završetka transmisije CS sekvence, posmatrani SM uređaj prelazi iz stanja SM_FLOOD u stanje SM_UNSYNC, gdje počinje sa ponovnim iniciranjem hladnog starta šaljući CS okvire prema komutatorima. Međutim, komutator koji nije pod greškom već se nalazi u nekom od usklađenih stanja, te ignoriše primljene CS okvire. S obzirom da je u ostatku mreže već obezbijeđena vremenska usklađenost između uređaja, u određenom trenutku CM uređaj koji nije pod greškom će proslijediti komprimovani IN okvir prema ES4 uređaju, koji tada prelazi u stanje SM_SYNC i dalje u SM_STABLE. Nije bitno u kojem trenutku ES4 uređaj primi komprimovani IN okvir od CM uređaja. Čim primi komprimovani IN okvir, uređaj ES4 prelazi u stanje SM_SYNC. Odavde je vidljivo da vrijednost do uspostavka vremenske usklađenosti na ES4 uređaju (dok je u SM_UNSYNC stanju) direktno zavisi od trenutka prijema komprimovanog IN okvira od CM uređaja koji nije pod greškom [2].

7.4.2 Slučaj kada komutator SW2 emituje sekvencu CA okvira prema uređaju ES4

Posmatrajući mašinu stanja SM uređaja (Sl. 29), uočavamo da prijem CA okvira u bilo kojem od stanja uzrokuje tranziciju u stanje SM_WAIT_4_CYCLE_START_CS. Izuzetak može biti SM_STABLE stanje ako se kontrolna promjenljiva `stable_ca_enabled` podešava tako da onemogući tranziciju iz stanja SM_STABLE u stanje SM_WAIT_4_CYCLE_START_CS. U slučaju da CM uređaj pod greškom emituje na svom priključku sekvencu CA okvira dok je SM uređaj u stanju

SM_WAIT_4_CYCLE_START_CS, posmatrani SM uređaj će ciklički ulaziti u spomenuto stanje sve dok se sekvenca ne završi.

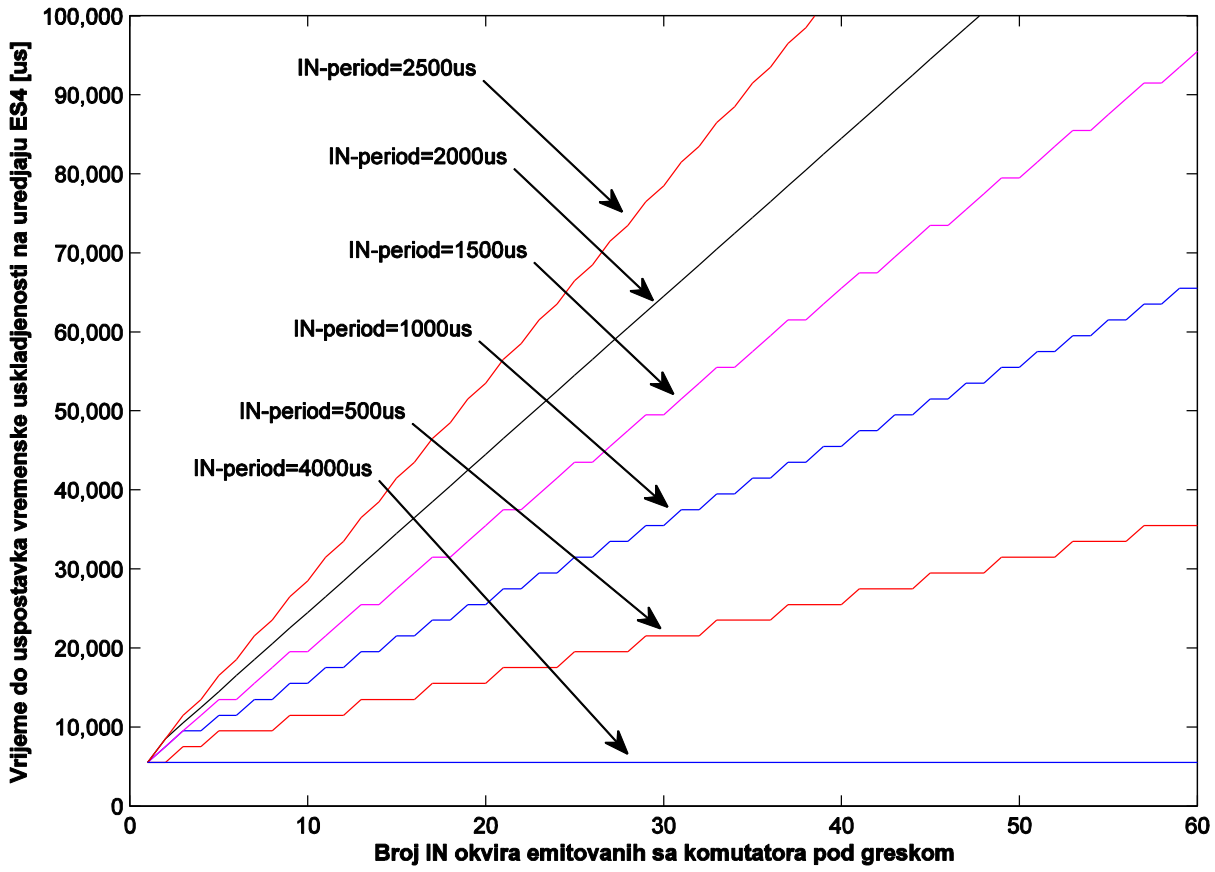
Rezultati na Sl. 44 prikazuju slučajeve kada CM uređaj pod greškom (komutator SW2) počinje da emituje sekvencu CA okvira kada je SM uređaj ES4 blizu isteka SM_WAIT_4_CYCLE_START_CS stanja. Kao i kod prethodnog slučaja, karakteristike za CA sekvencu imaju takođe stepenasti oblik. Pošto CA okviri imaju uticaja na sva stanja u kojima SM uređaj može da se nalazi, nezavisno od njihovog perioda oni će uzrokovati tranziciju u stanje SM_WAIT_4_CYCLE_START_CS. Ako je promjenljiva `stable_ca_enabled` podešena tako da je spomenuta tranzicija onemogućena dok je ES4 uređaj u SM_STABLE stanju, ES4 uređaj će ostati u SM_STABLE stanju za sve CA sekvence dovoljno velikog perioda. Jedan od ovih perioda je 4500 µs, kao što je prikazano na Sl. 44, gdje nije došlo do povećanja trajanja faze hladnog starta [2].



Sl. 44. Vrijeme do uspostavka vremenske usklađenosti (trajanje faze hladnog starta) na uređaju ES4. Slučaj kada CM uređaj pod greškom emituje sekvence CA okvira dok je ES4 uređaj u SM_WAIT_4_CYCLE_START_CS stanju [2]

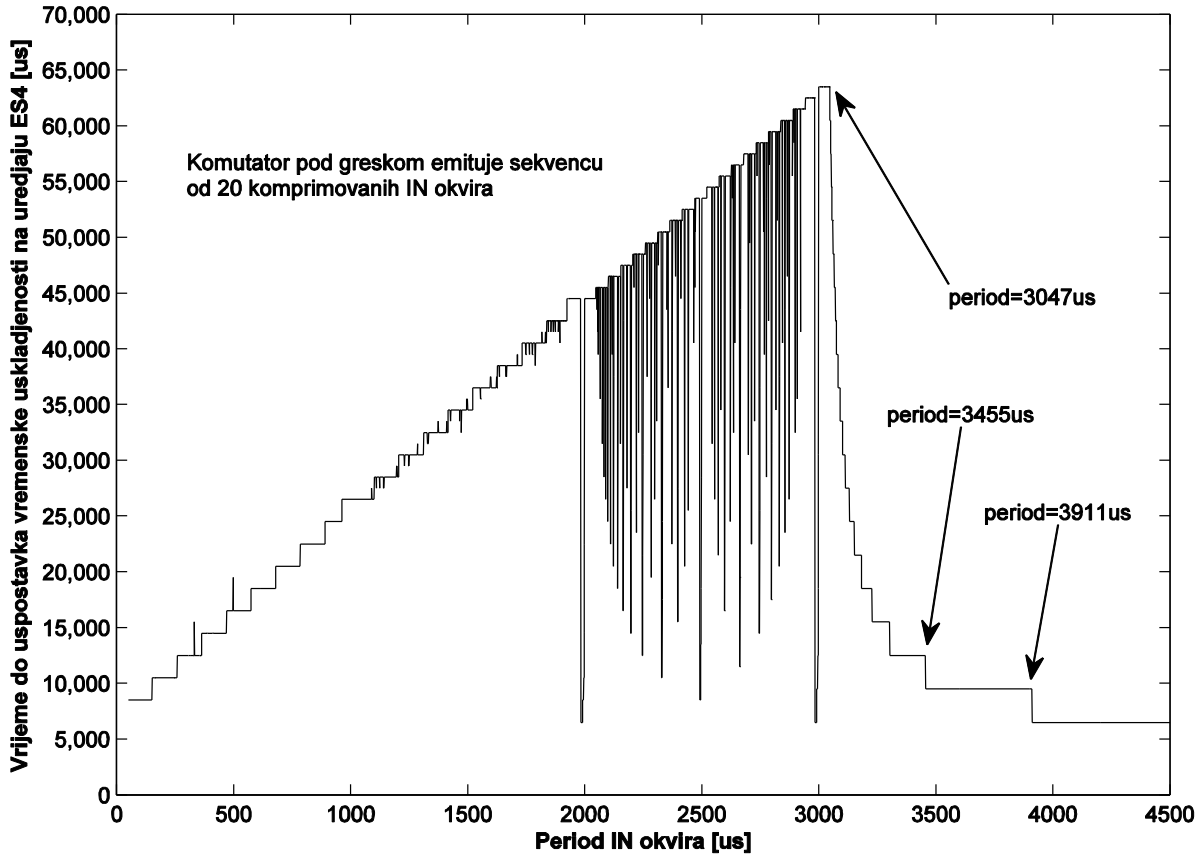
7.4.3 Slučaj kada komutator SW2 emituje sekvencu IN okvira prema uređaju ES4

Integracioni okvir primljen u nekom od usklađenih stanja (SM_TENTATIVE_SYNC, SM_SYNC ili SM_STABLE) izvan prozora prijema uzrokovaće asinhroni CD događaj (sekcija 4.3.3), što uzrokuje gubitak vremenske usklađenosti. Rezultati dati na **Sl. 45** se odnose na slučaj kada se na priključku CM uređaja pod greškom emituju IN okvire izvan prozora prijema na ES4 uređaju dok je on u SM_TENTATIVE_SYNC stanju, što dalje vodi tranziciji u SM_UNSYNC stanje. Kao što smo rekli ranije, u ovom stanju ES4 uređaj počinje sa iniciranjem hladnog starta slanjem CS okvira koji će biti ignorisani na strani komutatora SW1 koji se nalazi u nekom od sinhronih stanja. Po ustaljenom redoslijedu, komutator SW1 će emitovati komprimovani IN okvir prema ES4 uređaju koji će ga dovesti u SM_UNSYNC stanje. Međutim, sledeći IN okvir emitovan sa CM uređaja pod greškom će ponovo dovesti ES4 uređaj u SM_UNSYNC stanje. Opisani proces će se ciklički ponavljati sve dok se spomenuta sekvenca ne završi. Ukratko, lanac tranzicija između stanja je sledeći: SM_TENTATIVE_SYNC → Nx [SM_UNSYNC → SM_SYNC] → SM_STABLE, gdje je parametar N broj IN okvira u sekvenci. Dugi periodi između IN okvira nemaju velikog uticaja na trajanje hladnog starta (za period 4000 μs trajanje hladnog starta je produženo za samo jedan integracioni ciklus), ali mogu uzrokovati gubitak vremenske usklađenosti ako je SM uređaj u SM_STABLE stanju [2].



Sl. 45. Vrijeme do uspostavka vremenske usklađenosti (trajanje faze hladnog starta) na uređaju ES4. Slučaj kada CM uređaj pod greškom emituje sekvence IN okvira dok je ES4 uređaj u SM_TENTATIVE_SYNC stanju [2]

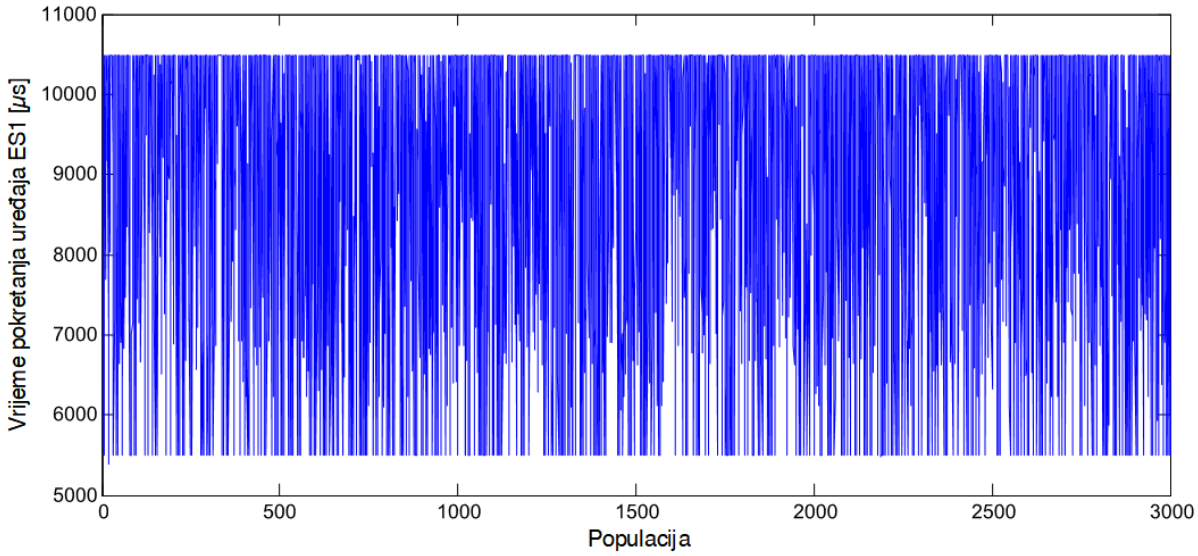
Procjena graničnog slučaja kada period sekvence IN okvira prestaje da utiče na rad SM uređaja predstavljeno je na Sl. 46. Prikazani rezultati se odnose na slučaj kada CM uređaj pod greškom emituje sekvence od 20 nekorektnih IN okvira sa različitim periodima prema ES4 uređaju. Može se primjetiti da za periode veće od 3047 μs (što je približno jednako trostrukom trajanju jednog integracionog ciklusa) trajanje faze hladnog starta eksponencijalno opada. Za periode veće od 3911 μs (približno jednako trajanju 4 integraciona ciklusa), IN okviri emitovani sa CM uređaja pod greškom nemaju uticaj na trajanje faze hladnog starta ES4 uređaja [2].



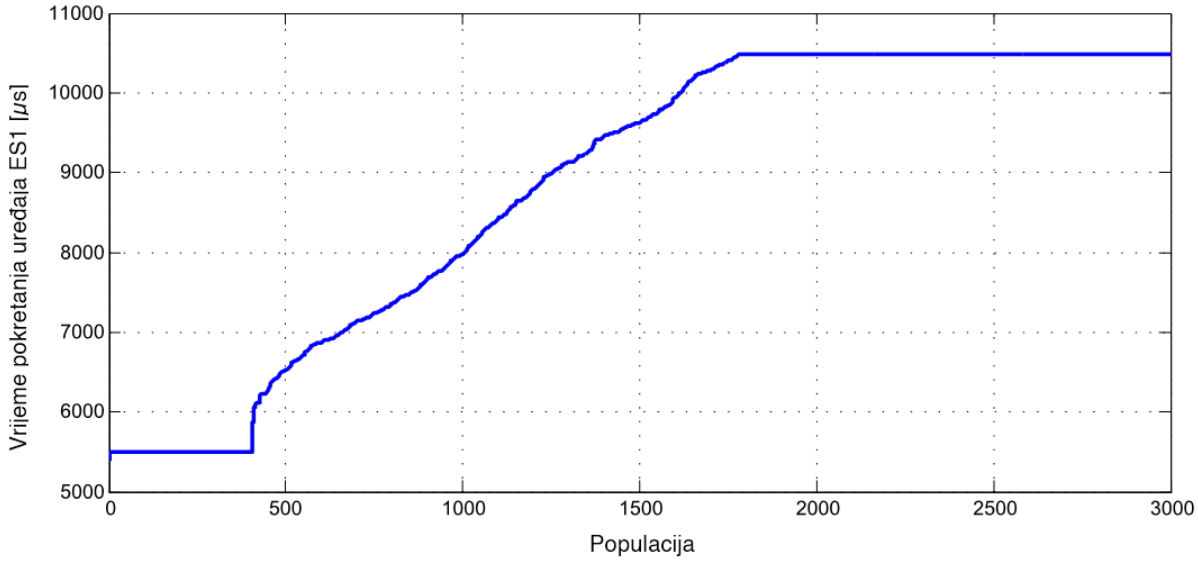
Sl. 46. Procjena graničnog slučaja kada period između IN okvira prestaje da utiče na hladni start SM uređaja [2]

7.4.4 Slučaj kada je krajnji uređaj pod otkazom pri čemu se koriste podrazumijevane vrijednosti parametara genetskog algoritma

Na početku genetskog algoritma, potrebno je generisati ulazne podatke, tj. početnu populaciju. Za ovu grupu simulacija koriste se vrijednosti parametara genetskog algoritma date u tabeli 3. U našem slučaju, početnu populaciju čini 3000 proizvoljno generisanih hromosoma dužine 20 gena. Broj 3000 je izabran empirijski: dovoljno je veliki da obezbijedi raznovrsnost populacije i spriječi preuranjenu konvergenciju, a u drugu ruku nije preveliki da bi značajno narušavao performanse genetskog algoritma. Vremena hladnog starta početne populacije prikazana su na Sl. 47. Posmatra se vrijeme do uspostavljanja usklađenog stanja časovnika uređaja ES1, koji nije pod otkazom. Hromosomi prikazani na Sl. 47 nisu sortirani, te ih je teže analizirati na ovakvom prikazu. Zbog toga, prikazaćemo ovaj i sve naredne grafike u sortiranom vidu prikaza, zbog lakše analize (Sl. 48).



Sl. 47. Početna populacija (nesortirano)



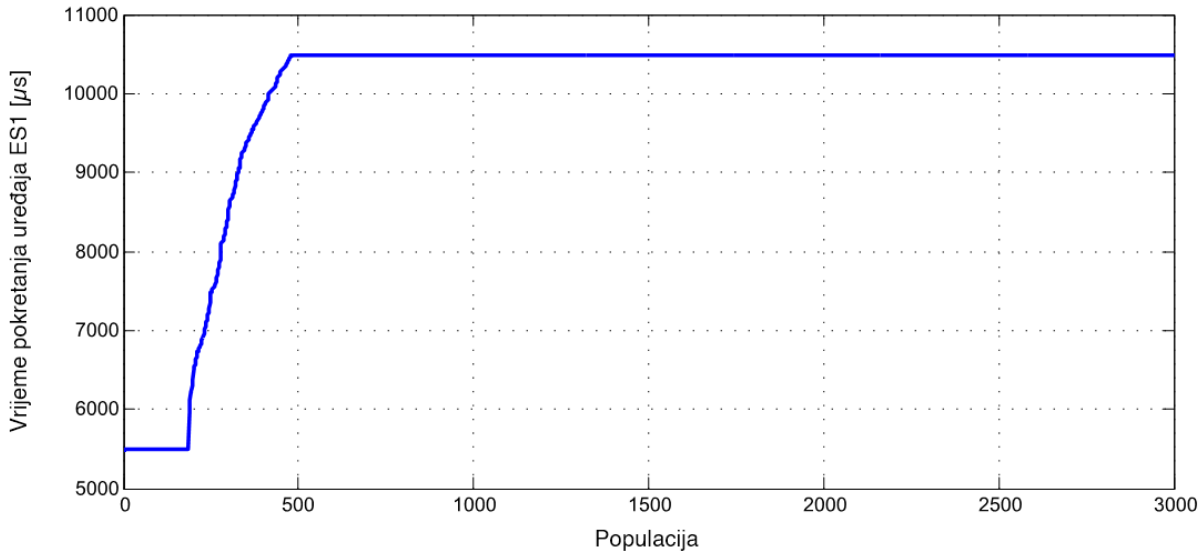
Sl. 48. Početna populacija (sortirano)

Sa grafika prikazanih na Sl. 47 i Sl. 48 uočavamo da najduže vrijeme hladnog starta (vrijeme do uspostavljanja vremenske usklađenosti) pronađeno pomoću početne populacije hromozoma iznosi 10489 μs. Broj hromosoma u početnoj populaciji koji uzrokuje ovoliko trajanje hladnog starta iznosi 1221. Kao što je spomenuto ranije, u slučaju da u mreži ne postoje uređaji pod otkazom, vrijeme trajanja faze hladnog starta iznosi 5489 μs. Sa grafika prikazanog na Sl. 48 uočavamo da u početnoj populaciji postoji 407 hromosoma koji ne utiču na produženje faze hladnog starta.

Da bi dobili sledeću generaciju, potrebno je na početnoj populaciji da izvršimo operacije odabira, ukrštanja i mutacije. Pri odabiru, biramo trećinu hromosoma iz početne

populacije metodom turnira, gdje su vjerovatnoće odabira najveće za najpodobnije hromosome. Za operaciju ukrštanja, koristili smo metod ukrštanja u dvije tačke. Nakon ove dvije operacije, primjenjuje se mutacija hromosoma na veoma malom broju hromosoma. Konkretno, vjerovatnoća mutacije iznosi 5%.

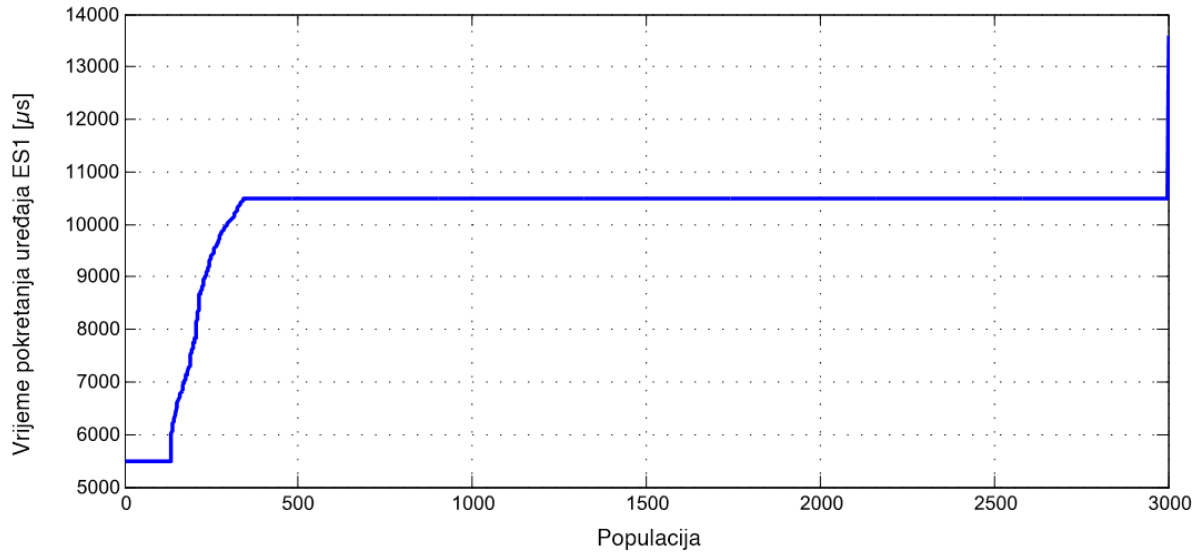
Kao rezultat genetskih operacija na početnoj populaciji, dobijamo sledeću generaciju prikazanu na Sl. 49. Uočavamo da je populacija druge generacije znatno kvalitetnija od početne generacije, jer posjeduju više hromosoma koji uzrokuju trenutno pronađeno najduže vrijeme hladnog starta (10489 µs). Međutim, pronađeno vrijeme nije povećano, tako da se tok genetskog algoritma nastavlja primjenjujući isti postupak sa operacijama odabira, ukrštanja i mutacije na hromosomima druge generacije. Slična situacija je i u trećoj generaciji.



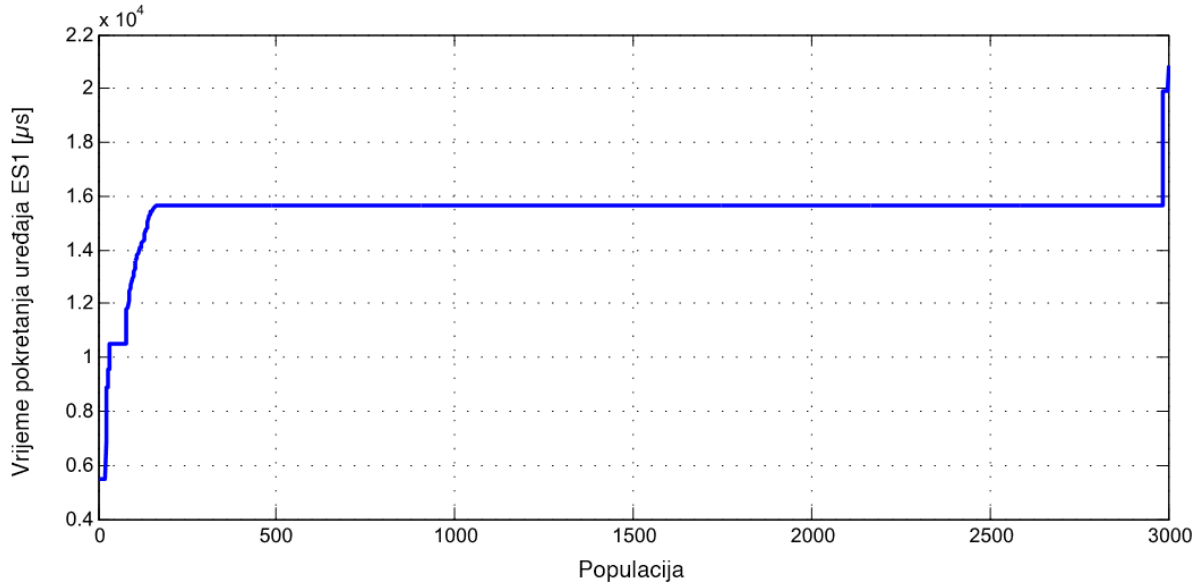
Sl. 49. Druga generacija (sortirano)

U četvrtoj generaciji, pojavljuje se hromosom koji uzrokuje vrijeme hladnog starta trajanja 13598 µs (Sl. 50). U iteracijama 5, 6, 7, 8, i 9 pronalaze se hromosomi koji daju vrijeme od 15679 µs. U iteraciji 10, pronalazi se hromosom koji daje vrijeme jednako 19906 µs. Najduže vrijeme hladnog starta pronađeno u iteraciji 11 (Sl. 51), kao i u svim narednim iteracijama iznosi 20868 µs. Na primjer, ako analiziramo populaciju u iteraciji 20 (Sl. 52), uočavamo da je ona samo kvalitetnija od populacije 11, ali da duže vrijeme hladnog starta nije pronađeno. Izvršeno je ukupno 100 iteracija. Na primjer, jedan od hromosoma iz dvadesete iteracije koji uzrokuje ovakvo vrijeme je: "IN-1714µs-CA-19µs-CS-1708µs-IN-1967µs-CA-642µs-IN-1043µs-CA-190µs-IN-211µs-IN-690µs-CA-2396µs".

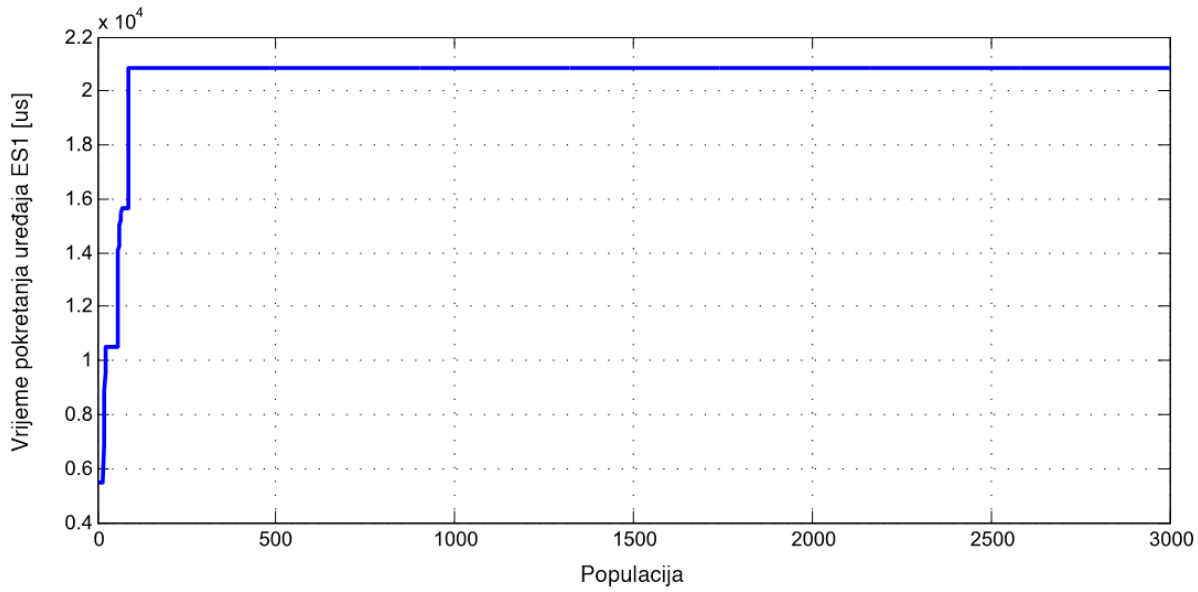
Drugim riječima, ukoliko SM uređaj pod greškom konstantno emituje ovakve sekvence PCF okvira, za zadate parametre genetskog algoritma biće pronađeno vrijeme pokretanja svih ostalih krajnjih uređaja (koji nisu pod otkazom) jednako 20868 μs. Imajući u vidu nedeterminističku prirodu genetskog algoritma, ne mora značiti da će svaka simulacija za iste parametre genetskog algoritma dati isti rezultat kao što je trenutno pronađen. Dakle, neophodno je izvršavati simulacije više puta da bi se dobile odgovarajuće procjene vremena pokretanja, kao što je to opisano u poglavlju 7.4.5.



Sl. 50. Četvrta generacija (sortirano)

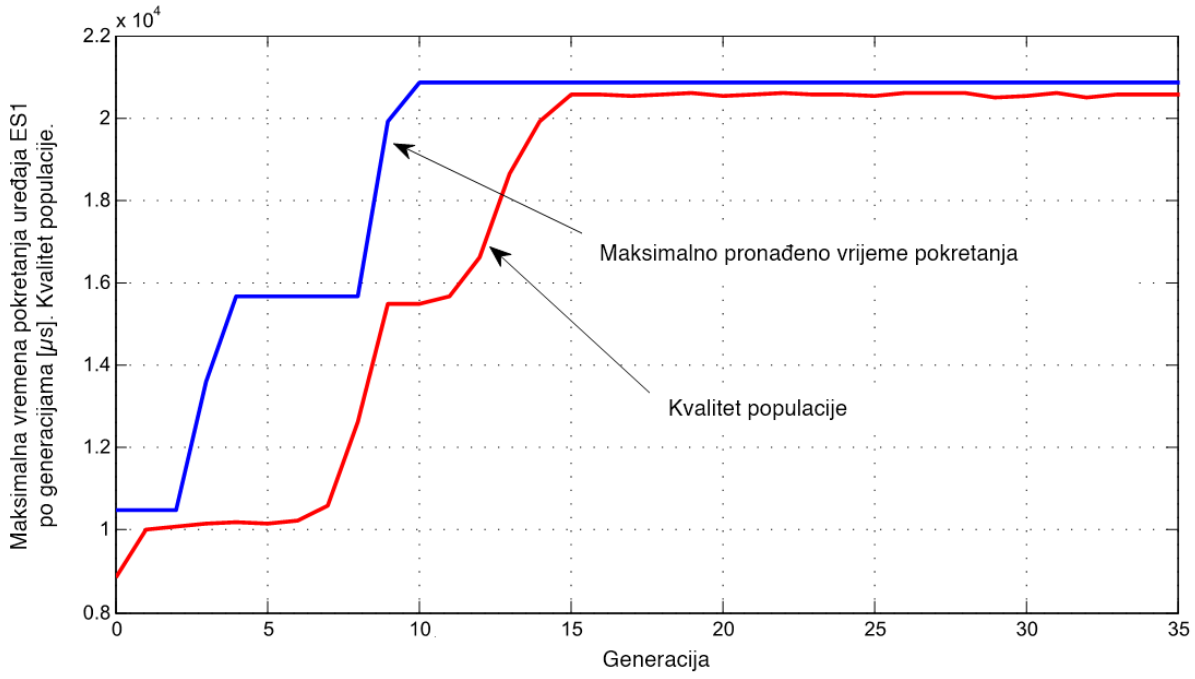


Sl. 51. Jedanaesta generacija (sortirano)



Sl. 52. Dvadeseta generacija (sortirano)

Grafik gdje je prikazano najduže trajanje hladnog starta po iteracijama, zajedno sa grafikom koji daje kvalitet populacije, dati su na Sl. 53. Kvalitet populacije (generacije) definisali smo kao sumu vremena hladnog starta koje uzrokuju svi hromosomi u iteraciji, skaliranu sa 3000 (veličina populacije). Kriva kvaliteta populacije je rastuća, kao i kriva vremena hladnog starta. Međutim, njena vrijednost nije konstantna, posmatrano za sve populacije gdje je pronađeno maksimalno trajanje vremena hladnog starta. To je posljedica činjenice da se hromosomi u tekućoj populaciji namijenjeni za genetske operacije biraju po principu vjerovatnoće, te se razmatra i mali broj hromosoma koji nemaju visoku podobnost.



Sl. 53. Najduža vremena hladnog starta po iteracijama. Kvalitet generacije.

7.4.5 Rezultati simulacija za različite parametre genetskog algoritma

U ovom poglavlju dati su rezultati genetskog algoritma za slučajeve kada se za simulaciju koriste različite vrijednosti njegovih parametara. Generalno govoreći, iako su parametri genetskog algoritma veoma važni za tok genetskog algoritma, oni ipak nisu presudni za finalni rezultat. Drugim riječima, genetski algoritam je veoma fleksibilan algoritam, te varijacije njegovih parametara ne mogu uticati mnogo na njegov krajnji rezultat, jer algoritam daje dobra rješenja za širok spektar vrijednosti njegovih parametara. Rađeno je mnogo pokušaja da se poboljšaju performanse genetskog algoritma, ali kao rezultat su se dobila minorna poboljšanja. Grefenštet (*John J. Grefenstette*) [67] je zaključio da su faktori koji najviše utiču na performanse genetskog algoritma kodovanje i funkcija podobnosti, dok drugi parametri ne utiču mnogo na tok genetskog algoritma. Ako genetski algoritam već daje loše rezultate za izabrane parametre, tada će najčešće davati loše rezultate i za drugačije vrijednosti parametara. Loši rezultati mogu biti posljedica loše formulisane funkcije podobnosti i neadekvatnog izbora načina kodovanja. Međutim, ako genetski algoritam daje dobre rezultate, pronalaskom optimalnih parametara moguće je poboljšati njegove performanse [62] [82].

Pošto je genetski algoritam nedeterminističke prirode, neophodno je izvršiti simulaciju nekoliko puta za iste parametre. U simulatoru je podešeno da se svaka simulacija izvršava po 10 puta, gdje se kao rezultat dobija grupa rješenja kod kojih su za nas najbitnije srednja vrijednost (eng. *mean*), medijana (eng. *median*), kao i granice 95% intervala povjerenja za ove vrijednosti. Na osnovu ovih rezultata vrši se procjena koji parametri daju najbolje rezultate, da bi se kasnije mogli kombinovati u cilju pronalaženja optimalnih parametara genetskog algoritma za problematiku posmatranu u ovom radu. Kao što je već spomenuto, sa varijacijama parametara genetskog algoritma nije moguće očekivati enormno poboljšanje performansi genetskog algoritma, ali je ovo neophodan korak jer je moguće da podrazumijevani parametri izabrani na početku nisu optimalni. Nakon toga, kada dobijemo optimalne parametre, tada možemo reći da je veoma mala šansa da daljim varijacijama parametara možemo dodatno poboljšati performanse genetskog algoritma [82].

Svi dobijeni rezultati su prikazani sa intervalom povjerenja 95% baziranom na srednjoj vrijednosti i medijani. Spomenuti interval povjerenja za srednju vrijednost rezultata definiše se kao [68]:

$$\overline{X} \pm Z\frac{s}{\sqrt{n}} \tag{31}$$

gdje X predstavlja srednju vrijednost za sve dobijene rezultate, parametar Z predstavlja konstantu koja ima vrijednost 1.96 za interval povjerenja 95%, s je standardna devijacija, a n je broj rezultata (10 u našem slučaju). Interval povjerenja 95% za medijanu računa se drugačije nego za srednju vrijednost, gdje se gornja i donja granica intervala računa kao:

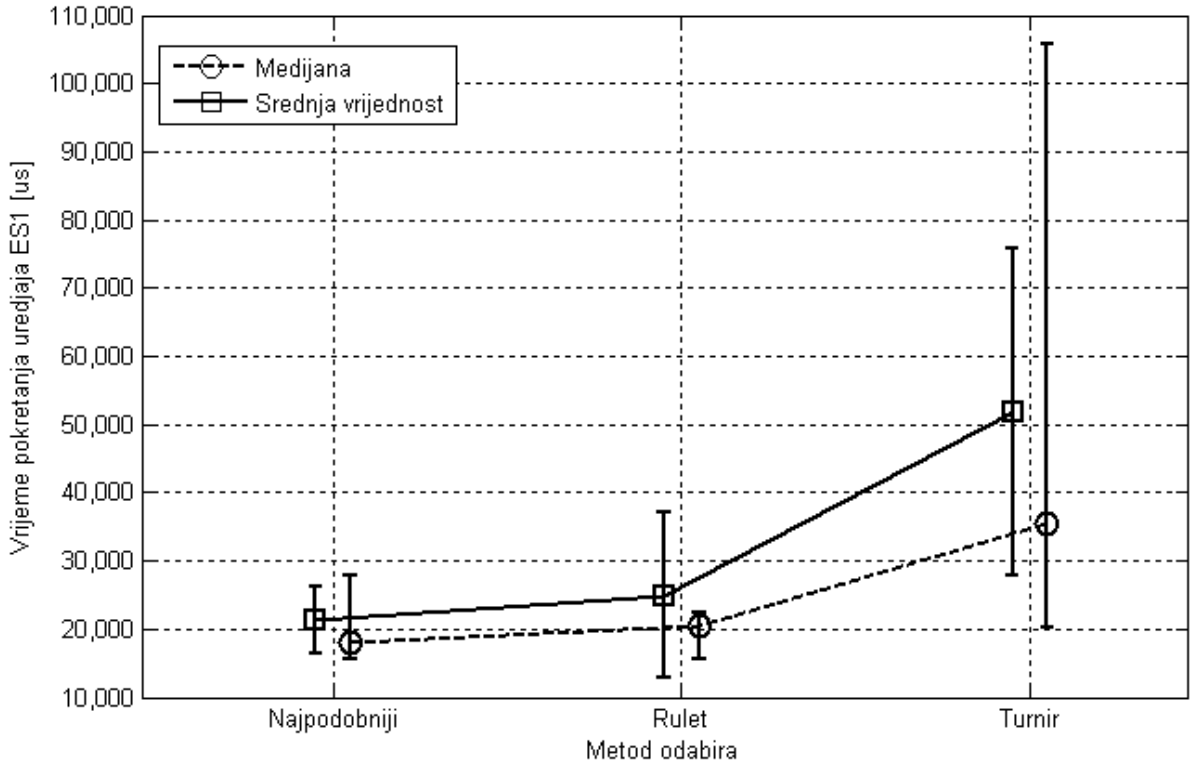
$$L_{donja} = \left(\frac{n}{2} - \frac{Z\sqrt{n}}{2}\right)$$
$$L_{gornja} = \left(1 + \frac{n}{2} + \frac{Z\sqrt{n}}{2}\right) \tag{32}$$

gdje su parametri L_{donja} i L_{gornja} redni brojevi vrijednosti svih dobijenih rezultata kada se oni sortiraju u rastućem redoslijedu. Konkretno u našem slučaju gdje je dobijeno po 10 rezultata za svaku simulaciju, granice 95% intervala povjerenja predstavljaju drugu i devetu vrijednost dobijenih rezultata. Medijana je smještena unutar ovih granica.

Za svaku od narednih simulacija varirali smo samo po jedan parametar genetskog algoritma, dok su svi ostali parametri posjedovali fiksirane vrijednosti podrazumijevanih parametara datih u tabeli 3.

7.4.5.1 **Odabir**

Odabir predstavlja prvu operaciju genetskog algoritma. Izbor metoda odabira utiče na sve naredne korake algoritma. Primjena pogrešne metode odabira takođe može prouzrokovati loše performanse genetskog algoritma [69]. Iz tog razloga, uklonićemo metodu koja daje najlošije rezultate. Simulirali smo 3 tipa odabira: odabir najpodobnijeg hromosoma, rulet i turnir. Dobijeni rezultati simulacije su prikazani na Sl. 54. S obzirom da rulet i turnir metode odabira daju najbolje rezultate, svim narednim simulacijama nismo koristili metod odabira najpodobnije jedinke. Turnir metod daje najbolje performanse prvenstveno zato što je efikasniji u pogledu konvergencije [70]. Generalno, u praksi turnir daje bolje rezultate od rulet metode odabira [62] [82].



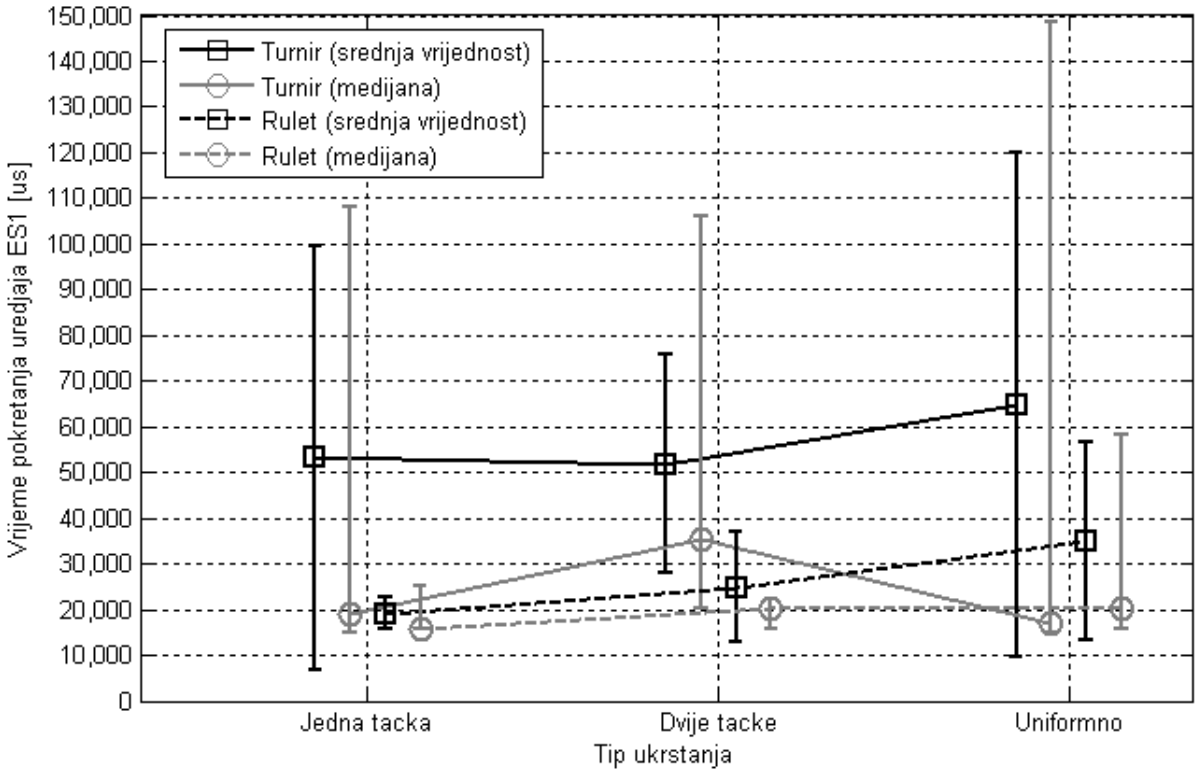
Sl. 54. Varijacija metode odabira [82]

7.4.5.2 **Ukrštanje**

Preuranjena konvergencija događa se najčešće zato što se odabirom uklanjaju hromosomi sa lošim podobnostima. Ovi hromosomi mogu da sadrže i neke dobre sekvence gena koje su

uklonjene tokom procesa odabira. Pošto genetski algoritam dopušta i odabir manjeg broja hromosoma sa lošijim podobnostima, ukrštanjem se mogu očuvati dobre sekvence gena i u takvim hromosomima [62]. Simulacije su urađene za ukrštanje u jednoj tački, ukrštanje u dvije tačke, te uniformno ukrštanje. Ukrštanje u jednoj tački trebalo bi bolje da očuva sekvence gena u hromosomima sa lošijim podobnostima, mada broj ovakvih hromosoma nije veliki nakon odabira [82].

Inače, ukrštanje ima znatno manji uticaj na performanse genetskog algoritma od operacije odabira. Međutim, u zavisnosti od prirode posmatranog problema, može biti važno koji tip ukrštanja će biti izabran. U problemima kod kojih je važan redoslijed gena u hromosomu (gdje je velika međusobna zavisnost između gena), ukrštanje u jednoj tački može biti pogodnije za korištenje jer ima mogućnost da bolje očuva duže sekvence gena u hromosomu, u odnosu na ukrštanja u više tačaka. Ukrštanje u više tačaka je bolji izbor kod problema kod kojih je mala međusobna zavisnost između gena u hromosomu [62]. Za slučaj analize procesa usklađivanja časovnika u *TTEthernet* mreži, očigledno je da je redoslijed tipova PCF okvira (zajedno sa periodima) veoma važan da bi se izazvalo određeno ponašanje mreže, tako da zaključujemo da su u ovoj problematici geni dosta međusobno zavisni. Kao što je već opisano ranije, *TTEthernet* mreža je za različita stanja u kojim se nalazi osjetljiva na različite tipove PCF okvira, a u nekim stanjima u potpunosti zanemaruje određene PCF okvire. Rezultati simulacija za različite tipove ukrštanja dati su na Sl. 55 [82].



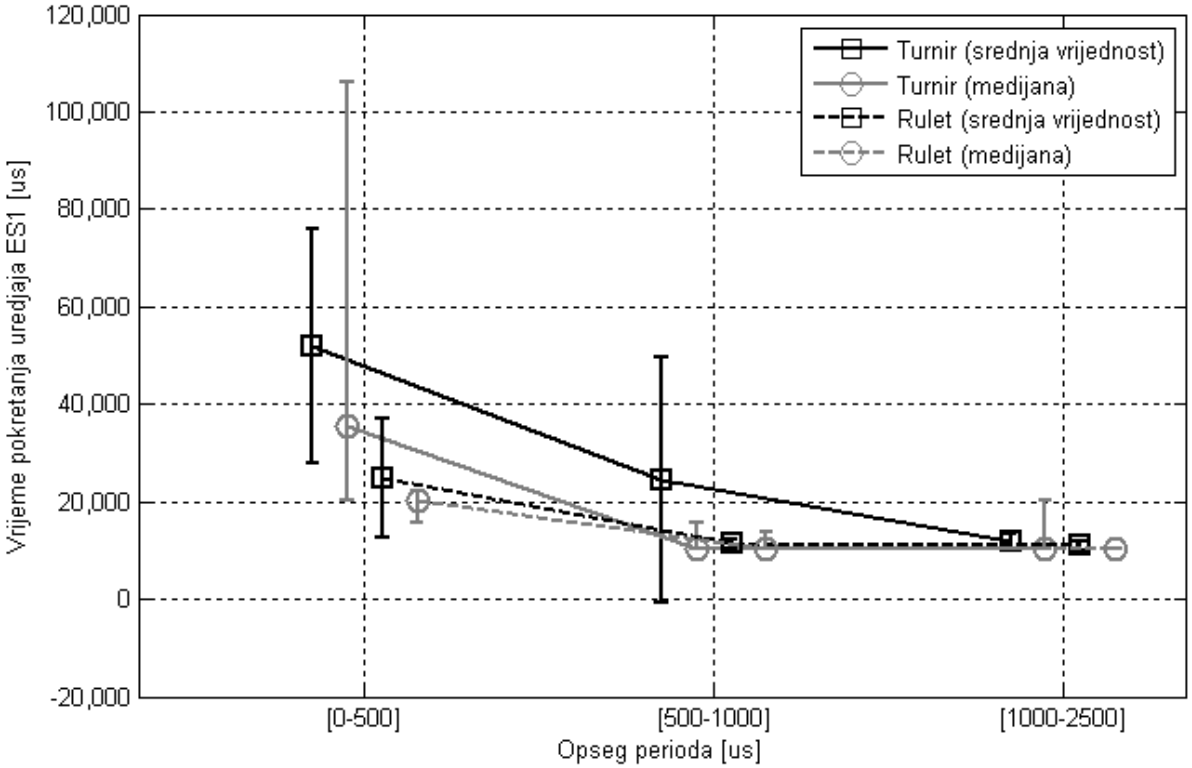
Sl. 55. Varijacije tipa ukrštanja [82]

Kao što se vidi sa dobijenih rezultata (Sl. 55) sa uniformnim ukrštanjem dobija se najbolji rezultat. Međutim, vjerovatnoća dobijanja ovakvog rezultata je veoma mala, što se može vidjeti na rezultatima za medijanu turnirskog odabira, gdje je medijana smještena na samo dno intervala povjerenja. S druge strane, iako su geni u hromosomu visoko međuzavisni, rezultati simulacije pokazuju da ukrštanje u dvije tačke daje bolje rezultate od ukrštanja u jednoj tački. Imajući to u vidu zaključujemo takođe i da raznovrsnost populacije igra značajnu ulogu u simulaciji *TTEthernet* mreže. Dakle, uočavamo da je ukrštanje u dvije tačke dobar kompromis između očuvanja dobrih sekvenci u genima i raznovrsnosti populacije, koji se inače postižu ukrštanjem u jednoj tački i uniformnim ukrštanjem [82].

7.4.5.3 **Period između PCF okvira**

Za simulacije različitih mogućih perioda između PCF okvira uzimali smo sledeće granice perioda: [0 - 500] μs, [500 - 1000] μs, [1000 - 2500] μs, koje u odnosu na podešeno trajanje integracionog ciklusa *ICD* (1000 μs) odgovaraju trajanjima [0 - 0.5] x *ICD*, [0.5 - 1] x *ICD*, [1 - 2.5] x *ICD*. Drugim riječima, konkretan period između dva PCF okvira može poprimiti jednu od vrijednosti iz zadatih granica. Međutim, u simulacijama smo omogućili da se kasnije, primjenom mutacije može dobiti i period između nekih PCF okvira koji uzima vrijednost iz opsega [0 - 2500] μs.

Očekuje se da kraći periodi imaju veći uticaj na produženje vremena pokretanja uređaja, zato što je tada i veća vjerovatnoća da će specifični PCF okvir doći na neki uređaj koji nije pod otkazom u trenutku kada je on najosjetljiviji na taj tip PCF okvira. Zbog ovog razloga, opseg [0 - 500] μs je podešen kao podrazumijevani za sve simulacije. Rezultati dobijeni za ovaj tip simulacija dati su na Sl. 56 i oni potvrđuju pretpostavku da kraći periodi PCF okvira uzrokuju duže vrijeme pokretanja uređaja koji nisu pod otkazom [82].



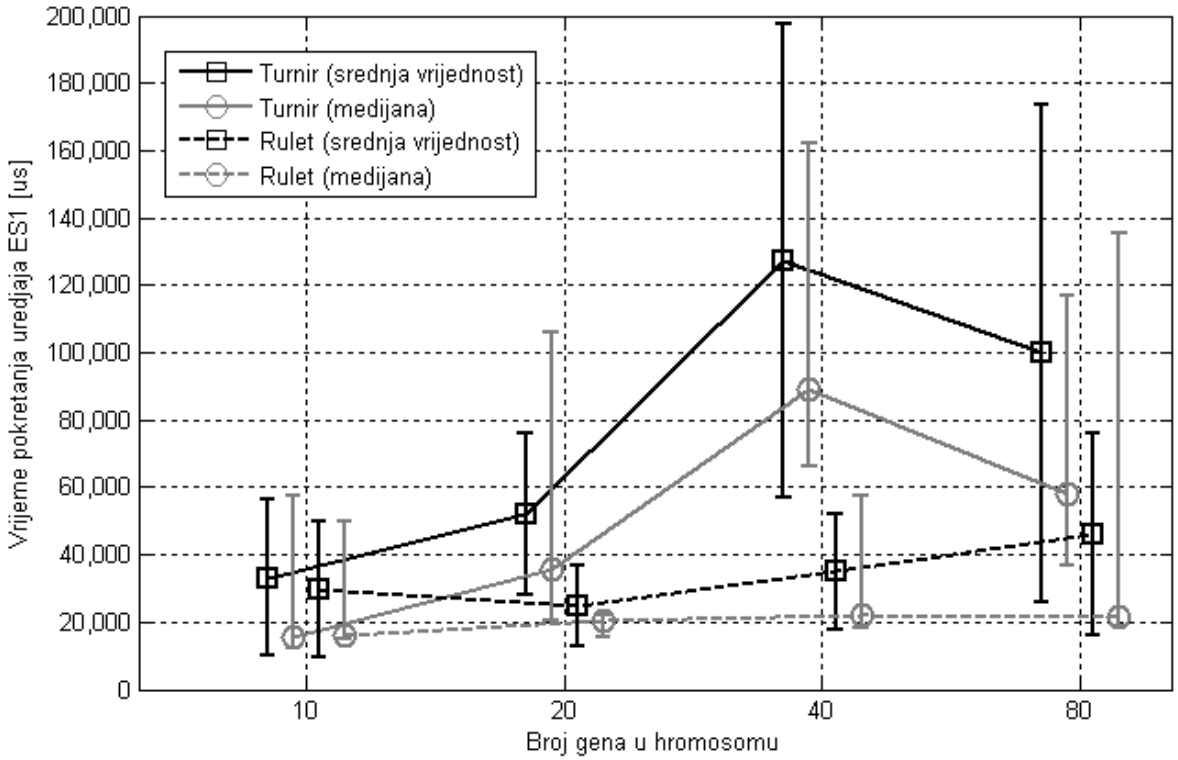
Sl. 56. Varijacije perioda [82]

7.4.5.4 **Dužina hromosoma**

Simulacije su takođe vršene i za različite dužine hromosoma: 10 gena, 20 gena, 40 gena i 80 gena. Tokom vremena, stanja implementirana u mašini stanja uređaja se mijenjaju, što znači da je u određenim trenucima uređaj osjetljiv samo na određene tipove okvira, dok druge tipove zanemaruje. Veoma male dužine hromosoma ne mogu obezbjediti dovoljnu raznolikost tipova PCF okvira u hromosomu, dok za preduge sekvence može da se dogodi da se faza pokretanja uređaja završi i prije završetka sekvence, tako da dio sekvence postaje beznačajan iako u ovom slučaju imamo najbolju raznolikost hromosoma [82].

Najniža granica koja je izabrana u simulacijama je 10 gena koja predstavlja primjer kratkog hromosoma. Kao što je prikazano na dobijenim rezultatima (Sl. 57), ovakva dužina

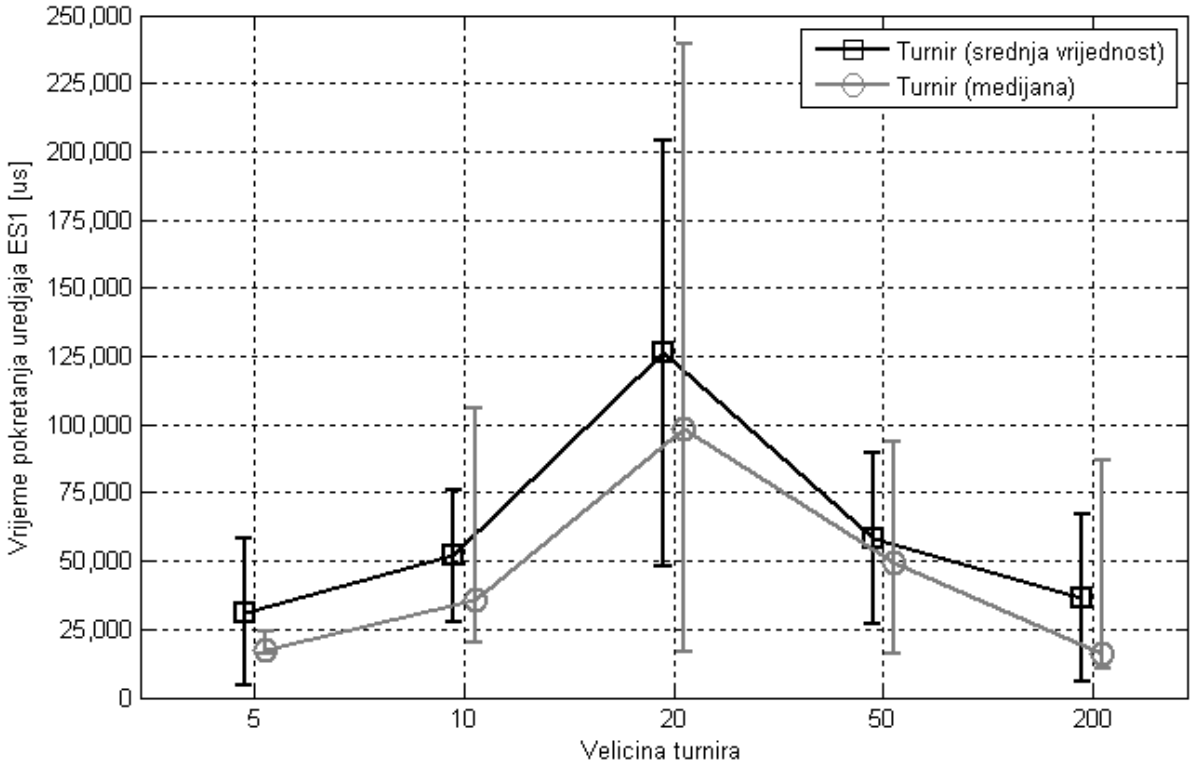
hromosoma uzrokuje loše performanse genetskog algoritma, prvenstveno zbog nedovoljne raznolikosti gena. Zbog već spomenutih razloga, duga sekvenca (80 gena) također ne daje najbolje rezultate, tako da je dovoljno koristiti kraće dužine tipa 20 ili 40 gena [82].



Sl. 57. Varijacije dužine gena [82]

7.4.5.5 **Veličina turnira**

Kada se koristi turnirski tip odabira, dodatni parametar koji se može razmatrati je veličina turnira. Goldberg i Deb [71] su također analizirali ovaj parametar i dobijali su bržu konvergenciju genetskog algoritma za veće veličine turnira, ali nisu uspjeli da dobiju bolji finalni rezultat. Za velike veličine turnira događa se gubitak raznolikosti populacije što vodi preuranjenoj konvergenciji, a također i lošim rezultatima genetskog algoritma. Gubitak raznolikosti događa se zato što neki hromosomi ne dobijaju šansu da budu izabrani zbog posljedica nasumičnog odabira, ili ne budu izabrani u nekom od turnira [61]. Kao što smo već spomenuli, može se dogoditi i da neki loši hromosomi posjeduju dobre sekvence gena. Simulirane su vličine turnira 5, 10, 20, 50 i 200. Kao što je i očekivano, za prevelike veličine turnira dobijaju se loši rezultati kao i za premale veličine. Najbolji rezultat se dobija za veličinu turnira 20, što se vidi na Sl. 58 [82].

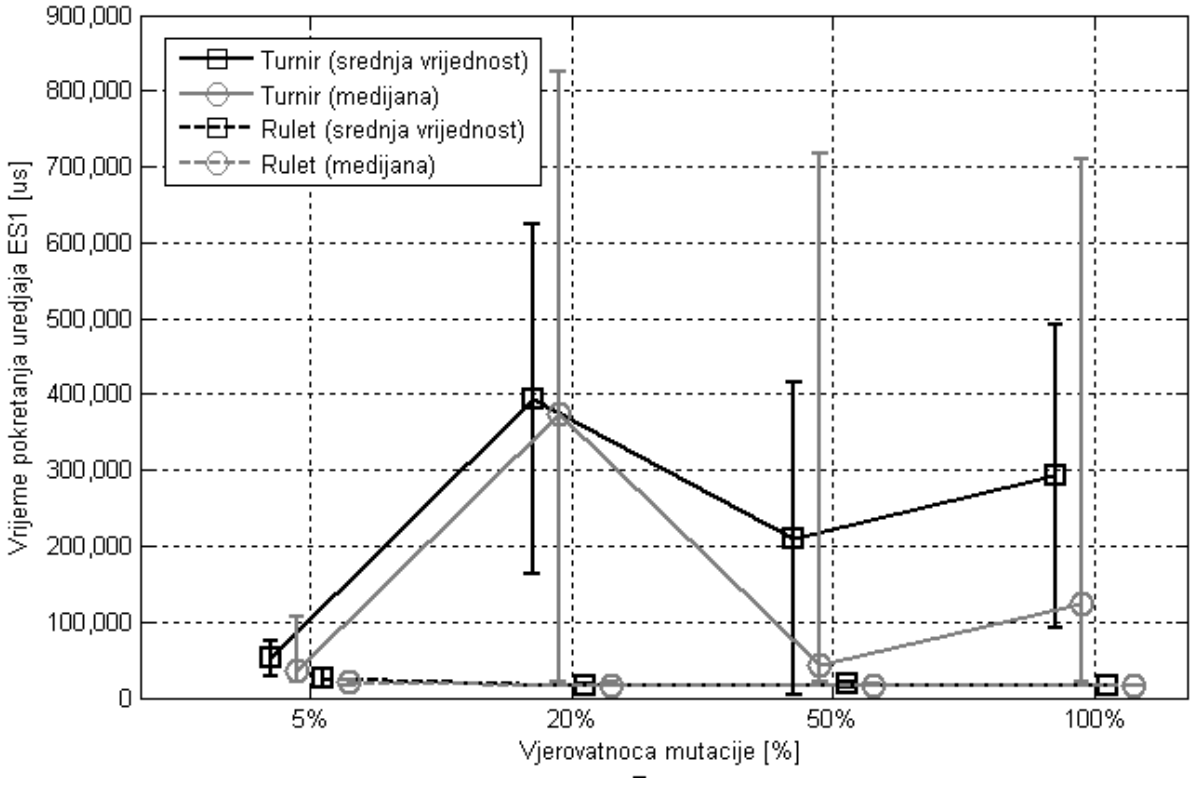


Sl. 58. Varijacija veličine turnira [82]

7.4.5.6 **Mutacija**

Jedan od osnovnih razloga preuranjene konvergencije je što operacije odabira i ukrštanja prestaju da daju rezultate u populacijama sa istim hromosomima. Teorijski, jedino se mutacijom može prevazići ovakav problem i na taj način se poboljšati raznolikost populacije. Ako je nivo mutacije nizak, nivo promjene genetskog materijala je mali tako da dobri hromosomi brzo eliminišu sve druge hromosome u populaciji. Međutim, ako je nivo mutacije visok, tada dolazi do redukovanja genetskog algoritma na obični pretraživački algoritam bez strategije. Stoga su simulacije izvršavane za različitu količinu gena u hromosomu koji mogu biti narušeni mutacijom (1, 2, 4, 8 i 20 gena na hromosomu dužine 20 gena), kao i na različitim vjerovatnoćama mutacije gena (5%, 20%, 50% i 100%).

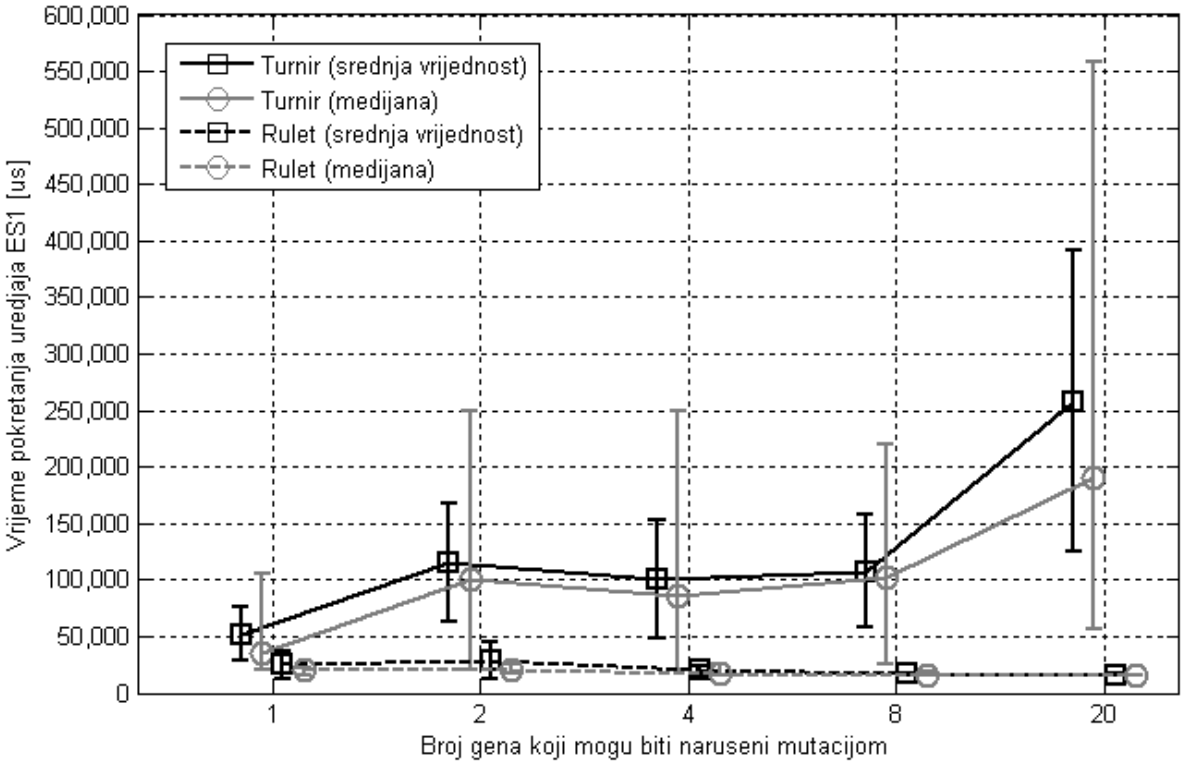
Kada se simulira velika količina gena u hromosomu koji mogu biti narušeni mutacijom, tada nije dobro koristiti visoke vjerovatnoće mutacije, jer to dovodi do spomenutog narušavanja strategije genetskog algoritma. Na Sl. 59 su dati rezultati za različite vjerovatnoće mutacije ako samo jedan gen u hromosomu može biti narušen mutacijom.



Sl. 59. Varijacija vjerovatnoće mutacije [82]

Simulacije pokazuju da se najbolji rezultati dobijaju za vjerovatnoću mutacije 20%, što znači da se vrijednošću 5% ne ostvaruje dovoljna raznolikost populacije, dok se za veće vrijednosti narušava strategija genetskog algoritma. Takođe, ovdje treba imati u vidu da je u ovim simulacijama samo jedan gen mogao biti narušen mutacijom, što znači da za veliki broj narušenih gena i za istu ovu vjerovatnoću performanse genetskog algoritma mogu značajno da se naruše, utičući i na konvergenciju, kao i na finalni rezultat.

Rezultati za različite brojeve gena u hromosomu koji mogu biti narušeni mutacijom dati su na Sl. 60, gdje je vjerovatnoća mutacije fiksirana na 5%, a veličina hromosoma na 20 gena. Iako se najbolji rezultati dobijaju kada svi geni mogu biti narušeni mutacijom, treba imati u vidu da je ovdje korištena veoma mala vjerovatnoća mutacije (5%). Za hromosom veličine 20 gena to znači da samo jedan gen u hromosomu može biti mutiran po iteraciji, što znači da je strategija genetskog algoritma očuvana.



Sl. 60. Varijacija broja gena u hromosomu koji mogu biti narušeni mutacijom [82]

Posmatrajući prethodno dobijene rezultate, može se izvršiti kombinovanje parametara koji daju najbolje rezultate po svakoj simulaciji i cilju poboljšanja performansi genetskog algoritma. Odabrane su vrijednosti parametara kako je prikazano u tabeli 4. Za srednju vrijednost rezultata dobija se 583694 μs unutar intervala povjerenja 95%, dok je medijana 489579 μs unutar intervala povjerenja 95%. Iz ovih rezultata može se zaključiti da se sa novim parametrima ostvaruje poboljšanje srednje vrijednosti dobijenih rezulatata kao i medijane. Dakle, može se reći da su ovo optimalni parametri genetskog algoritma čijim se daljim variranjem samo minorno mogu dodatno poboljšati performanse genetskog algoritma kada se on primjenjuje na analizu vremena pokretanja *TTEthernet* mreže. Parametri koji najviše utiču na performanse genetskog algoritma su vjerovatnoća mutacije i broj gena koji može biti narušen mutacijom. Kao što se može vidjeti u tabeli 4, ipak nisu birane ekstremne vrijednosti ovih parametara da ne bi došlo do gubljenja strategije genetskog algoritma, a samim tim i njegovih performansi [82].

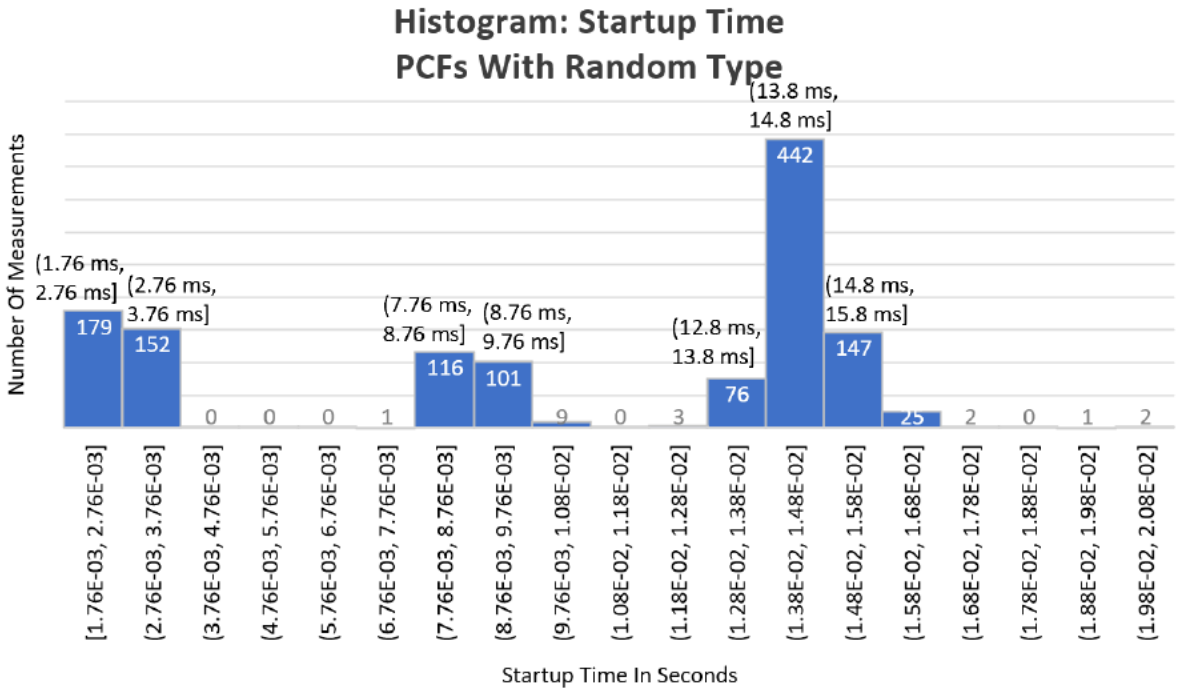
Tabela 4. Optimalne vrijednosti parametara genetskog algoritma [82].

Parametar	Vrijednost
Veličina generacije	3000
Veličina reprodukcije	1000
Dužina hromosoma	40
Minimalni period	0 μs
Maksimalni period	500 μs
Tip odabira	Turnir
Tip ukrštanja	2 tačke
Vjerovatnoća mutacije	20%
Broj mutiranih gena	2
Veličina turnira	20

8 Praktična industrijska mjerenja

Što se tiče praktičnih industrijskih mjerenja trajanja pokretanja *TTEthernet* mreže, na njima je radio Bija [81] u okvirima svog završnog rada. Rad također obuvaća scenario gdje postoji krajnji uređaj pod otkazom, koji emituje različite sekvence PCF okvira u neočekivanim trenucima prema ostatku mreže. Posmatrana je ista topologija kao i u tezi, tj. četiri krajnja uređaja i dva komutatora koja rade u redundantnom režimu, gdje je jedan krajnji uređaj pod otkazom. Za razliku od teze, gdje se primjenjuju simulacije na analizu protokola usklađivanja vremena, posmatrani rad analizira ponašanje realne *TTEthernet* mreže.

Za razliku od pristupa primijenjenog u tezi, gdje postoji strategija pronalaska najgoreg slučaja vremena pokretanja *TTEthernet* mreže, u posmatranog radu strategija ne postoji, tj. sekvence PCF okvira koje se šalju sa uređaja pod otkazom se nasumično izrađuju. Također, period između okvira u jednoj sekvenci je fiksan za sve okvire, dok u tezi može da varira od okvira do okvira. Izmjerena vremena pokretanja *TTEthernet* mreže u posmatranom radu za više iteracija, gdje je u svakoj iteraciji upotrijebljena drugačija sekvenca, data su na Sl. 61.



Sl. 61. Praktična industrijska mjerenja vremena pokretanja *TTEthernet* mreže u kojoj postoji krajnji uređaj pod otkazom [81]

Na *x*-osi grafika (Sl. 61) su prikazani rezultati mjerenja vremena pokretanja za komutator i za krajnje uređaje koji nisu pod otkazom, dok visina grafikona odgovara broju mjerenja. Mjerenja su vršena za ukupno 1500 različitih sekvenci PCF okvira, gdje je na *y*-osi grafika pokazano koliko je mjerenja dalo isti posmatrani rezultat. Npr. dobijeno je 442 rezultata mjerenja koji pokazuju da je određena PCF sekvenca uzrokovala vrijeme pokretanja krajnjih uređaja koji nisu pod otkazom jednako 14.8 ms, dok je vrijeme pokretanja komutatora u tim slučajevima 13.8 ms [81].

Najduže vrijeme pokretanja krajnjeg uređaja koje je pronađeno ovim pristupom iznosi 20.8 ms i pronađeno je za dvije različite PCF sekvence. Ako se ovo vrijeme uporedi sa rezultatima iz teze, na Sl. 53 se vidi da je genetski algoritam također pronašao skoro isto

vrijeme za početne vrijednosti njegovih parametara. Također, broj pronađenih sekvenci koje uzrokuju ovakvo vrijeme (blizu 3000 sekvenci) je mnogo veći nego broj ovakvih sekvenci pronađenih u posmatranom radu (2 sekvence) [81]. Naknadnim varijacijama vrijednosti parametara genetskog algoritma dobijaju se još bolji rezultati u pogledu pronađenih vremena pokretanja mreže.

U zaključku posmatranog rada [81] spominje se da bi bilo neophodno pronaći odgovarajuću strategiju za efikasan pronalazak najgoreg slučaja pokretanja *TTEthernet* mreže, umjesto nasumičnog slanja različitih PCF sekvenci sa uređaja pod otkazom. Kao jedna od mogućih strategija navodi se pokušaj primjene genetskog algoritma.

9 Zaključak i pravci daljeg istraživanja

9.1 Zaključak disertacije

Kao mreža sa više prioriteta, *TTEthernet* se koristi u bezbjednosno-kritičnim sistemima poput automobilske i avionske industrije, gdje pored determinističke može da postoji i naletna klasa saobraćaja za prenos podataka manjeg značaja, kao što su multimedijalni sadržaji. Da bi deterministička komunikacija između uređaja u posmatranoj *TTEthernet* mreži uopšte bila izvodljiva, korektno uspostavljeno i održavano vremensko usklađivanje časovnika između mrežnih uređaja mora biti od najvišeg značaja.

Disertacija se bavi procesom uspostavljanja vremenske usklađenosti u mreži, te simulacijama uticaja uređaja pod greškom na usklađivanje vremena između časovnika u mreži. U sklopu disertacije objašnjeni su takođe i osnovni koncepti rada determinističkih mreža i usklađivanja vremena u determinističkim mrežama. Obrađene su klase saobraćaja u *TTEthernet* mreži sa fokusom na PCF klasu saobraćaja koja se primjenjuje u procesu unutrašnjeg usklađivanja časovnika. Režimi procesa vremenskog usklađivanja su takođe objašnjeni: pokretanje, ponovno pokretanje, te normalni režim rada.

Ponašanje *TTEthernet* mreže u kojoj postoji komponenta pod otkazom je analizirano putem simulacija. Korišćeni simulator je razvijen u alatu *OMNeT++* v5.1, a parametri za podešavanje mrežnih uređaja su uzeti kao izlaz komercijalnog alata za podešavanje *TTEthernet* mreža TTE Tools v5.0. Simulirana su 2 slučaja: slučaj kada je komutator pod greškom i slučaj kada je krajnji uređaj pod greškom. Neispravna komponenta u mreži je podešena tako da nekontrolisano šalje različite tipove PCF okvira na svom spoljnom priključku. Simulacije se odnose na trajanje procedure uspostavljanja vremenske usklađenosti u *TTEthernet* mreži za situacije komponente pod otkazom. S obzirom da nije bio cilj simulirati kvalitet vremenskog usklađivanja časovnika, nego samo vrijeme uspostavljanja vremenske usklađenosti, sve komponente u simuliranoj mreži koriste idealne časovnike. Jedan od pravaca budućeg rada mogao bi biti orijentisan na analize sa realnim modelima lokalnih časovnika.

U prvom slučaju, komutator na jednom od svojih priključaka emituje ciljane tipove PCF okvira u ciljanim trenucima prema krajnjem uređaju. Ciljani trenuci su izabrani tako da odgovaraju kritičnim momentima za krajnji uređaj, tj. kada krajnji uređaj treba da promijeni

trenutno stanje u kojem se nalazi. Određen je i granični slučaj kada sekvenca PCF okvira emitovana sa komutatora pod greškom prestaje da ima uticaja na funkcionisanje krajnjeg uređaja.

Drugi slučaj podrazumijeva da je krajnji uređaj pod greškom, te emituje proizvoljne sekvence PCF okvira različitih tipova i perioda prema komutatoru, a samim tim i prema ostatku mreže. Za razliku od prvog slučaja gdje su okviri emitovani u ciljanim trenucima, u ovim simulacijama krajnji uređaj počinje sa emitovanjem proizvoljnih sekvenci PCF okvira odmah na početku simulacije. U ovim simulacijama cilj je bio da se izvrši procjena najgoreg slučaja za uspostavljanje vremenske usklađenosti u posmatranoj mreži i za te svrhe je poslužio genetski algoritam. Generalno, problem izrade rasporeda slanja okvira u deterministčkim mrežama je NP-kompletan problem. Spomenuti najgori slučaj za uspostavljanje vremenske usklađenosti u *TTEthernet* mreži je uzrokovan PCF okvirima emitovanim od strane krajnjeg uređaja pod otkazom. Drugim riječima, tražen je raspored slanja PCF okvira emitovanih od strane krajnjeg uređaja pod otkazom, koji uzrokuje najduže trajanje faze pokretanja posmatrane mreže. Samim tim zaključujemo da je problem posmatran u tezi takođe NP-kompletan problem.

Pošto je genetski algoritam nedeterminističke prirode, bilo je neophodno izvršiti simulaciju nekoliko puta za iste parametre. U simulatoru je podešeno da se svaka simulacija izvršava po 10 puta, gdje se kao rezultat dobija grupa rješenja kod kojih su za nas najbitnije srednja vrijednost (eng. *mean*), medijana (eng. *median*), kao i granice 95% intervala povjerenja za ove vrijednosti. Veći broj izvršenih simulacija bi dao još vjernije rezultate, međutim ovo bi zahtijevalo razmatranje paralelizacije izvršavanja simulacija na više mašina u cilju skraćenja ukupnog vremena potrebnog za traženu procjenu. Na osnovu dobijenih rezultata vršila se procjena koji parametri daju najbolje rezultate, i koji su kasnije kombinovani u cilju pronalaženja optimalnih parametara genetskog algoritma za problematiku posmatranu u ovom radu. Nakon dobijanja optimalnih parametara, može se reći da je veoma mala šansa da se naknadnim varijacijama parametara mogu dodatno poboljšati performanse genetskog algoritma. Sa optimalnim parametrima genetskog algoritma, za srednju vrijednost rezultata dobija se 583694 µs unutar intervala povjerenja 95%, dok medijana iznosi 489579 µs unutar intervala povjerenja 95%. S obzirom da bez uspostavljene vremenske usklađenosti između časovnika nije izvodljiva razmjena TT poruka u mreži, samim tim će najosjetljivije funkcije određenog bezbjednosno-kritičnog sistema (npr. sistem

kočenja u automobilu) biti nedostupne. Rezultati simulacija iz ovog rada pokazuju da za date parametre *TTEthernet* mreže, u slučaju otkaza krajnjeg uređaja, najkritičnije funkcije sistema mogu biti nedostupne za vremenske periode čija medijana iznosi 489579 μs sa intervalom povjerenja 95%. Ukoliko projektant posmatranog bezbjednosno-kritičnog sistema zaključi da je ovo vrijeme predugo, te da sistem ne smije toliko dugo imati nedostupne svoje najosjetljivije funkcije, tada može da pristupi razmatranju mogućnosti da se ovaj problem prevaziđe. Ove mogućnosti mogu da se ogledaju u promjeni parametara mreže, razmatranju uvodjenja redundantnih komponenata i slično. Na primjer, skraćenje trajanja integracionog ciklusa bi dovelo do kraćeg vremena uspostavljanja vremenske usklađenosti između uređaja u mreži, ali bi to s druge strane dovelo do dodatnog opterećenja veza zbog povećanog PCF saobraćaja, tako da bi trebalo voditi računa i o posljedicama ovakvih prilagođenja. Što se tiče postupka uvođenja redundantnih komponenata, taj pristup je opisan kao jedan od pravaca daljeg istraživanja u sekciji 9.2.

9.2 Pravci daljeg istraživanja

U tezi je analiziran proces uspostave vremenske usklađenosti u *TTEthernet* mreži za slučaj idealnih lokalnih časovnika u mrežnim komponentama. Rad bi se mogao dodatno proširiti za slučajeve kada se u mrežnim komponentama koriste neidealni modeli lokalnih časovnika. U odnosu na trajanje do uspostavljanja vremenske usklađenosti za slučaj idealnih časovnika koje je približno 5.5 ms, varijacija ovog trajanja u slučaju realnih časovnika bila bi jedan od mogućih pravaca budućeg istraživanja. Očekuje se da bi količina varijacije direktno zavisila od vrijednosti preciznosti lokalnih časovnika mrežnih komponenata.

Odmah po puštanju mreže u rad, svi mrežni uređaji započinju proces usklađivanja od svog SM_INTEGRATE/CM_INTEGRATE stanja. Međutim, *TTEthernet* mreža posjeduje mogućnost stabilizacije (uspostavljanje usklađenog stanja) čak i u slučajevima da se komponente pokrenu iz proizvoljnih početnih stanja. Opisana pojava bi takođe bila interesantna za analizu u budućem radu. Očekuje se da bi vrijeme do uspostave usklađenog stanja direktno zavisilo od izabranih početnih stanja svake od mrežnih komponenata.

Takođe, bilo bi interesantno posmatrati ponašanje mreže i za slučajeve kada različite komponente otkazuju u različitim trenucima. Otkaz komponente mogao bi se zakazati u nekom od neusklađenih stanja mreže, kao i u usklađenim stanjima mreže prije nastupanja

stanja SM_STABLE/CM_STABLE. Kao što je opisano ranije, otkaz jedne komponente neće uticati na normalan tok usklađivanja časovnika u mreži otpornoj na jedan otkaz, dok bi otkaz dvije komponente doveo do gubitka vremenske usklađenosti u kompletnoj mreži. Ista analiza bi se mogla primijeniti na mrežu otpornu na 2 otkaza, tj. mrežu sa stepenom redundanse 3. Simulacije komplikovanijih topologija *TTEthernet* mreže bi takođe bile interesantne za buduća istraživanja. S obzirom da svaki komutator sa SC ulogom (međustepen) na putanji prenosa okvira unosi svoje statičko kašnjenje, topologije sa više međustepeni (eng. *multi-hop*) bi mogle da se simuliraju u cilju procjene uticaja broja međustepeni i statičkog kašnjenja svakog međustepena na vrijeme uspostavljanja usklađenog stanja u mreži kada je neki od krajnjih uređaja pod otkazom.

Teza se bavi procesom uspostavljanja vremenski usklađenog stanja u *TTEthernet* mreži. Analiza mreže u domenu vremenski usklađenih stanja bi takođe bila od interesa i trebalo bi odrediti koji aspekti analize bi mogli biti od interesa za posmatranje. Parametri simulirane mreže su preuzeti kao rezultujući izlaz alata TTE Tools. Ponašanje mreže pri varijacijama vrijednosti parametara bi bilo takođe interesantno za simulaciju.

Procjena najgoreg slučaja za uspostavak usklađenih stanja u *TTEthernet* mreži učinjena je uz pomoć genetskog algoritma. Neki od pravaca budućeg istraživanja mogli bi biti orijentisani prema mogućnostima primjene genetskog algoritma u nekim drugim aspektima funkcionisanja *TTEthernet* mreže. Na primjer, mogla bi se razmotriti mogućnost izrade rasporeda TT okvira uz pomoć genetskog algoritma, za primjene gdje treba da se optimizuje prenos okvira nižeg prioriteta, tj. RC i BE okvira.

Pronađeni najgori slučaj za vremenski period u kojem časovnici nisu međusobno usklađeni zbog postojanja komponente pod otkazom, takođe daje informaciju koliko dugo mreža može biti onemogućena za razmjenu TT poruka kojima se prenose najkritičnije informacije u sistemu. Bitan pravac istraživanja može biti orijentisan u smijeru koje akcije bi trebalo preduzeti da bi se premostio ovaj period ukoliko je on predug za posmatrani sistem, tako da bi posmatrani bezbjednosno-kritični sistem mogao što ranije da se vrati normalnom radu. Jedan od aspekata koji bi se mogao istraživati je promjena vrijednosti parametara mreže u cilju skraćenja spomenutog perioda nedostupnosti servisa. Međutim, ako se ovim pristupom ne bi dobili zadovoljavajući rezultati, bilo bi neophodno razmatranje zamjene TT poruka sa RC porukama sve dok traje nemogućnost uspostavljanja TT saobraćaja. Iako RC klasa saobraćaja ima mnogo manji nivo determinizma u odnosu na TT klasu saobraćaja, ona ne

zahtijeva međusobno usklađene časovnike, a takođe ima i ograničeno kašnjenje okvira. Očigledno je RC klasa saobraćaja ne može zamijeniti TT klasu saobraćaja za najkritičnije operacije, ali treba imati u vidu da bi ovo vrijeme zamjene bilo relativno kratko (medijana vremena pronađenih u ovom radu iznosi 489579 μs), te bi trebalo uraditi istraživanje da li bi ovaj pristup bio izvodljiv u neočekivanim okolnostima otkaza uređaja. Tada bi se mogla definisati dva režima rada mreže: deterministički (uobičajeni režim) u normalnim okolnostima rada gdje se odvija TT saobraćaj, te poludeterministički režim RC saobraćaja u nepredviđenim okolnostima koji predstavlja prelazni režim kratkog trajanja dok se ne ostvare uslovi za uobičajeni režim rada.

Literatura

[1] M. Sandić, I. Velikić, A. Bilbija and M. Milošević, "Analiza principa komunikacije u TTEthernet mrežama", in *ETRAN*, Kladovo, 2017.

[2] M. Sandić, B. Pavković and N. Teslić, "Impact of Anomalies within TTEthernet Network on Synchronization Protocol: Analysis Using OMNeT++ Simulations", in *ZINC*, Novi Sad, 2018.

[3] R. Obermaisser, Time-Triggered Communication, Boca Raton, London, New York: CRC Press, Taylor and Francis Group, 2012.

[4] P. Meyer, T. Steinbach, F. Korf and T. C. Schmidt, "Extending IEEE 802.1 AVB with time-triggered scheduling: A simulation study of the coexistence of synchronous and asynchronous traffic", in *IEEE Vehicular Networking Conference*, Boston, MA, USA, 2013.

[5] G. Alderisi, A. Caltabiano, G. Vasta, G. Iannizzoto, T. Steinbach and L. Lo Bello, "Simulative assessments of IEEE 802.1 Ethernet AVB and Time-Triggered Ethernet for Advanced Driver Assistance Systems and in-car infotainment", in *IEEE Vehicular Networking Conference (VNC)*, Seoul, South Korea, 2012.

[6] T. Steinbach, H.-T. Lim, F. Korf, T. C. Schmidt, D. Herrscher and A. Wolisz, "Tomorrow's In-Car Interconnect? A Competitive Evaluation of IEEE 802.1 AVB and Time-Triggered Ethernet (AS6802)", in *Vehicular Technology Conference (VTC Fall)*, Quebec City, QC, Canada , 2012.

[7] A. Varga, "OMNeT++ Discrete Event Simulator", 2017. [Online]. Available: https://www.omnetpp.org/. [Accessed 16 Januar 2018].

[8] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company, 1989.

[9] S. Poledna, H. Kopetz and W. Steiner, "Deterministic system design with Time-Triggered technology", in *Microelectronic Systems Symposium (MESS), 2014*, Vienna, 2014.

[10] X. Tang, Q. Li, G. Li and H. Xiong, "Safe Clock Synchronization Mechanism for Multi-Cluster TTEthernet Networks", in *10th International Conference on Wireless Communications and Signal Processing (WCSP)*, Hangzhou, China, 2018.

[11] Q. Zhao, H. Zhao and B. Zhu, "Research on time synchronization algorithm of TTE multi-hop system based on IEEE1588 protocol", in *3rd IEEE International Conference on Computer and Communications (ICCC)*, Chengdu, China , 2017.

[12] R. Zhao, G. Qin, Y. Lyu and J. Yan, "Security-Aware Scheduling for TTEthernet-Based Real-Time Automotive Systems", *IEEE Access*, vol. 7, pp. 85971 - 85984, 2019.

[13] J. Wang, P. Ding, Y. Wang and G. Yan, "Back-to-Back Optimization of Schedules for Time-Triggered Ethernet", in *37th Chinese Control Conference (CCC)*, Wuhan, China, 2018.

[14] D. Tamas-Selicean and P. Pop, "Optimization of TTEthernet networks to support best-effort traffic", in *Emerging Technology and Factory Automation (ETFA)*, Barcelona, Spain, September, 2014.

[15] W. Steiner, F. Bonomi and H. Kopetz, "Towards synchronous deterministic channels for the Internet of Things", in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*, Seoul, South Korea, 2014.

[16] R. S. Oliver, S. S. Craciunas and G. Stoger, "Analysis of Deterministic Ethernet scheduling for the Industrial Internet of Things", in *Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2014 IEEE 19th International Workshop on*, Athens, Greece, December, 2014.

[17] L. Zhao, H. Feng, E. Li and J. Lu, "Comparison of Time Sensitive Networking (TSN) and TTEthernet", in *IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*, London, UK, 2018.

[18] W. Steiner, "Synthesis of Static Communication Schedules for Mixed-Criticality Systems", in *Object/Component/Service-Oriented Real-Time Distributed*

Computing Workshops (ISORCW), 2011 14th IEEE International Symposium on, Newport Beach, CA, USA , 2011.

[19] S. S. Craciunas, "Optimal static scheduling of realtime tasks on distributed time-triggered networked systems", in *Emerging Technology and Factory Automation (ETFA),*, Barcelona, Spain, September, 2014.

[20] D. Tamas-Selicean, P. Pop and W. Steiner, "Synthesis of communication schedules for TTEthernet-based mixed-criticality systems", in *CODES+ISSS: International Conference on Hardware/Software Codesign and System Synthesis*, Tampere, Finland, October, 2012.

[21] M. Garey and D. Johnson, Computers and Interactability: A Guide to the Theory of NP-Completeness, USA: Bell Telephone Laboratories, Intercorporated, 1979.

[22] S. S. Craciunas and R. S. Oliver, "SMT-based Task- and Network-level Static Schedule Generation for Time-Triggered Networked Systems", in *RTNS-14: Proceedings of the 22nd International Conference on Real-Time Networks and Systems*, Versailles, France, 2014.

[23] F. Pozo, G. Rodriguez-Navas, H. Hannson and W. Steiner, "SMT-Based Synthesis of TTEthernet Schedules: A Performance study", in *10th IEEE International Symposium on Industrial Embedded Systems (SIES)*, 2015.

[24] M. Boyer, H. Daigmortes, N. Navet and J. Migge, "Performance Impact of the Interactions Between Time-Triggered and Rate-Constrained Transmissions in TTEthernet", in *8th European Congress on Embedded Real Time Software and Systems*, 2016.

[25] D. Onwuchekwa and R. Obermaisser, "Fault Injection Framework for Assessing Fault Containment of TTEthernet Against Babbling Idiot Failures", in *IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, Banff, AB, Canada, 2018.

[26] J. Li, L. Qiao and X. Tang, "Modeling TTEthernet Startup Service in SystemC for Verifying Fault-Tolerant Protocol under Fail-Omission Scenarios", in *TENCON 2018 - 2018 IEEE Region 10 Conference*, Jeju, Korea (South), 2018.

[27] "Development Tools," TTTech Computertechnik AG, 2020. [Online]. Available: https://www.tttech.com/products/aerospace-space/development-test-vv/development-tools/. [Accessed 4 April 2020].

[28] S. Majidi and R. Obermaisser, "Genetic Algorithm for Scheduling Time-Triggered Communication Networks with Data Compression", in *IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*, Taipei, Taiwan, 2019.

[29] A. Aquino, G. Denaro and P. Salza, "Worst-Case Execution Time Testing via Evolutionary Symbolic Execution", in *2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE)*, Memphis, TN, USA, 2018.

[30] Y. Pyrih, M. Kaidan, I. Tchaikovskyi and M. Pleskanka, "Research of Genetic Algorithms for Increasing the Efficiency of Data Routing", in *3rd International Conference on Advanced Information and Communications Technologies (AICT)*, Lviv, Ukraine, 2019.

[31] A. V. Dias, J. N. Amaral, A. Teixeira and C. Neto, "Genetic Algorithms in Optimization: Better than Random Search?", in *Programa de Pos-Graduacao em Engenharia Eletrica Pontificia Universidade do Rio Grande do Sul*, Porto Alegre, Brasil, 2000.

[32] K. Bisson, "SAE AS6802 Deterministic Ethernet Network Solution", Avionics Interface Technologies, 2011.

[33] M. D. Mesarovic and Y. Takahara, Abstract Systems Theory, Springer-Verlag, 1989.

[34] M. Sandić, G. Velikić, V. Davidović and A. Bilbija, "Analiza iskorištenosti linka u determinističkim mrežama", in *ETRAN*, Zlatibor, 2016.

[35] H. Kopetz, Real-Time Systems, Design Principles for Distributed Embedded Applications, New York, Dordrecht, Heidelberg, London: Springer, 2011.

[36] H. E. Roland and B. Moriarty, System Safety Engineering and Management, Second ed., USA: John Wiley & Sons, 1990.

[37] "Department of Defense Standard Practice - System Safety", 2012. [Online]. Available: https://acqnotes.com/acqnote/tasks/mil-std-882e-system-safety.

[38] H. Kopetz, "Fault Containment and Error Detection in the Time-Triggered Architecture", in *Proceedings of the Sixth International Symposium on Autonomous Decentralized Systems (ISADS'03)*, Pisa, Italy, 2003.

[39] A. Rolnitzky, "Reliability of Repairable Systems vs. Non-Repairable Systems", OPS AlaCarte, 1 12 2011. [Online]. Available: https://opsalacarte.com/reliability-of-repairable-systems-vs-non-repairable-systems/. [Accessed 1 1 2021].

[40] M. Abd-El-Barr, Design and Analysis of Reliable and Fault-Tolerant Computer Systems, London: Imperial College Press, 2007.

[41] "SAE AS6802 Standard," SAE International, 2011.

[42] M. Horauer, Clock Synchronization in Distributed Systems, Disertation, Wien, 2004.

[43] M. Sandić, A. Bilbija and I. Velikić, "Analiza integracije različitih klasa saobraćaja u mixed-critical mrežama", in *TELFOR*, Beograd, 2016.

[44] A. Raouf, "Minimize Frequency Drift In Crystals", [Online]. Available: http://www.electronicdesign.com/analog/minimize-frequency-drift-crystals. [Accessed 15 July 2018].

[45] E. Anceaume and I. Puaut, Performance Evaluation of Clock Synchronization Algorithms, INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE, 1998.

[46] M. Sandić, I. Velikić and A. Jakovljević, "Calculation of Number of Integration Cycles for Systems Synchronized Using the AS6802 Standard", in *ZINC*, Novi Sad, 2017.

[47] M. Sandic, B. Pavkovic and N. Teslic, "TTEthernet Mixed-Critical Communication: Overview and Impact of Faulty Switches", *IEEE Consumer Electronics Magazine*, vol. 9, no. 4, pp. 97-103, 2020.

[48] "Automotive Ethernet - Ethernet Frame," Vector, [Online]. Available: https://elearning.vector.com/index.php?&wbt_ls_seite_id=1588396&root=378422&seite=vl_automotive_ethernet_introduction_en. [Accessed 26 02 2018].

[49] W. Steiner, G. Bauer, B. Hall, M. Paulitisch and S. Varadarajan, "TTEthernet Dataflow Concept", in *Network Computing and Applications, 2009. NCA 2009. Eighth IEEE International Symposium on*, Cambridge, MA, USA , 2009.

[50] V. Gavrilut, D. Tamas-Selicean and P. Pop, "Fault-Tolerant Topology Selection for TTEthernet Networks", in *25th European Safety and Reliability Conference (ESREL 2015)*, Switzerland, 2015.

[51] W. Steiner, M. Gutierrez, Z. Matyas, F. Pozo and G. Rodriguez-Navas, "Current Techniques, Trends, and New Horizons in Avionics Network Configuration", in *Digital Avionics Systems Conference (DASC), 2015 IEEE/AIAA 34th*, Prague, Czech Republic , 2015.

[52] M. Sandić and I. Velikić, "Implementation of Frames Scheduling in Mixed-Critical Networks", in *ZINC*, Novi Sad, 2016.

[53] D. Tamas-Selicean, P. Pop and W. Steiner, "Timing Analysis of Rate Constrained Traffic for the TTEthernet Communication Protocol", in *Real-Time Distributed Computing (ISORC), 2015 IEEE 18th International Symposium on*, Auckland, New Zealand, 2015.

[54] W. Steiner, M. Ayhan and S. Punnekat, "Improving Intelligent Vehicle Dependability by Means of Infrastructure-Induced Tests", in *Dependable Systems and Networks Workshops (DSN-W), 2015 IEEE International Conference on*, Rio de Janeiro, Brazil , 2015.

[55] T. Steinbach, F. Korf and T. C. Schmidt, "Real-time Ethernet for automotive applications: A solution for future in-car networks", in *Consumer Electronics - Berlin (ICCE-Berlin), 2011 IEEE International Conference on*, Berlin, Germany, 2011.

[56] Y. Zhang, F. He, G. Lu and H. Xiong, "Clock synchronization compensation of

Time-Triggered Ethernet based on least squares algorithm", in *Communications in China (ICCC Workshops), 2016 IEEE/CIC International Conference on*, Chengdu, China , 2016.

[57] W. Steiner and B. Dutertre, "Layered Diagnosis and Clock-Rate Correction for the TTEthernet Clock Synchronization Protocol", in *Dependable Computing (PRDC), 2011 IEEE 17th Pacific Rim International Symposium on*, Pasadena, CA, USA , 2012.

[58] M. Sandić, B. Pavković, D. Živkov and B. M. Todorović, "Uticaj dužine integracionog ciklusa na međusobno usklađivanje časovnika u TTEthernet mrežama", in *TELFOR*, Beograd, 2018.

[59] M. Kantardžić, Data Mining: Concepts, Models, Methods, and Algorithms, 2nd ed., Hoboken, New Jersey: John Wiley & Sons, Inc., 2011.

[60] "Genetic Algorithm Tutorial", tutorialspoint, [Online]. Available: https://www.tutorialspoint.com/genetic_algorithms/index.htm. [Accessed 6 May 2018].

[61] A. Shukla, H. M. Pandey and D. Mehrotra, "Comparative Review of Selection Techniques in Genetic Algorithm", in *International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, 2015.

[62] J. J. Kratica, Paralelizacija genetskih algoritama za resavanje nekih NP-kompletnih problema, Doktorska disertacija, Beograd: Matematički fakultet, 2000.

[63] S. N. Sivanandam and S. N. Deepa, Introduction to Genetic Algorithms, Berlin: Springer-Verlag, 2008.

[64] M. Rouse, "discrete event simulation (DES)", July 2012. [Online]. Available: https://whatis.techtarget.com/definition/discrete-event-simulation-DES. [Accessed 29 April 2018].

[65] A. Varga and R. Hornig, "An Overview of the OMNeT++ Simulation Environment", in *Simutools '08 Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems &*

workshops, Marseille, France, March, 2008.

[66] "Development Tools", TTTech Computertechnik AG, Vienna, 2018. [Online]. Available: https://www.tttech.com/products/aerospace/development-test-vv/development-tools/. [Accessed 29 April 2018].

[67] J. J. Grefenstette, "Optimization of Control Parameters for Genetic Algorithms", *IEEE Transactions on Systems, Man, and Cybernetics,* vol. 16, no. 1, pp. 122-128, 1986.

[68] "Confidence Intervals", [Online]. Available: https://www.mathsisfun.com/data/confidence-interval.html. [Accessed 29 9 2020].

[69] E. Hag, I. Ahmad, A. Hussain and I. M. Almanjahie, "A novel selection approach for genetic algorithms for global optimization of multimodal continuous functions", *Hindawi Computational Intelligence and Neuroscience,* pp. 1-14, 2019.

[70] J. Zhong, X. Hu, J. Zhang and M. Gu, "Comparisom of Performance between Different Selection Strategies on Simple Genetic Algorithms", in *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, 2005.

[71] D. E. Goldberg and K. Deb, A Comparative Analysis of Selection Schemes Used in Genetic Algorithms, Ilinois: University of Ilinois, Department of General Engineering, USA, 1991.

[72] H. D. Karatza, "Gang scheduling performance on a cluster of non-dedicated workstations", in *S ’02: Proceedings of the 35th Annual Simulation Symposium*, 2002.

[73] H. Franke, J. Jann, J. E. Moreira, P. Pattnaik and M. A. Jette, "An evaluation of parallel job scheduling for asci blue-pacific", in *Supercomputing ’99: Proceedings of the 1999 ACM/IEEE conference on Supercomputing*, 1999.

[74] H. Rajei, M. Dadfar and P. Joshi, "Simulation of job scheduling for small scale clusters", in *WSC ’06: Proceedings of the 38th conference on Winter simulation*,

2006.

[75] R. Goswami and K. Iyer, "Simulation of software behavior under hardware faults", in *The Twenty-Third Internet Symposium*, 1993.

[76] S. Y. Jiang, G. Luo, J. Liu, S. S. Jiang and X. T. Li, "Fault-Tolerant Routing Algorithm Simulation and Hardware Verification of NoC", in *IEEE Transactions on Applied Superconductivity,*, 2014.

[77] R. Zhou, R. Min, Q. Yu, C. Li, Y. Sheng, Q. Zhou, X. Wang and K. C. Li, "Formal Verification of Fault-Tolerant and Recovery Mechanisms for Safe Node Sequence Protocol", in *Advanced Information Networking and Applications (AINA), 014 IEEE 28th International Conference,*, Victoria, BC, Canada, 2014.

[78] J. A. Perez-Celis, J. A. Ferrer-Perez, S. D. Santillan-Gutierrez and S. de la Rosa Nieves, "Simulation of Fault-Tolerant Space Systems Based on COTS Devices With GPSS", *IEEE Systems Journal*, vol. 10, pp. 53-58, August, 2014.

[79] S. Ghosh and J. S. Sengar, "Layout Design and Simulation of Fault-Tolerant Triple Modular Redundant ALU System", in *Computing, Communications and Networking Technologies (ICCCNT),2013 Fourth International Conference on*, Tiruchengode, India, July, 2014.

[80] P. G. Peon, H. Kopetz and W. Steiner, "Towards a reliable and high-speed wireless complement to TTEthernet", in *Emerging Technology and Factory Automation (ETFA)*, Barcelona, Spain, September, 2014.

[81] R.F. Bija, Software Fault-Injection In a TTEthernet Cluster", *Diploma Thesis*, Faculty of Electrical Engineering and Information Technology at the TU Wie*n*, 2018.

[82] B. Pavkovic, M. Sandic and N. Teslic, " A genetic simulation strategy: Application to single-fault analysis of TTEthernet synchronization protocol", *Journal of Systems Architecture*, vol. 117, August, 2021.

Publikacije objavljene tokom izrade teze

Časopisi

1. Miladin Sandić, Bogdan Pavković and Nikola Teslić, "TTEthernet Mixed-Critical Communication: Overview and Impact of Faulty Switches," in *IEEE Consumer Electronics Magazine*, vol. 9, no. 4, pp. 97-103, 1 July 2020, doi: 10.1109/MCE.2020.2978224. **(M21)**
2. Bogdan Pavković, Miladin Sandić, Nikola Teslić, "A genetic simulation strategy: Application to single-fault analysis of TTEthernet synchronization protocol", in *Journal of Systems Architecture,* vol. 117, August 2021, doi: https://doi.org/10.1016/j.sysarc.2021.102169 **(M21)**

Konferencije

1. Miladin Sandić, Bogdan Pavković, Nikola Teslić, "Impact of Anomalies within TTEthernet Network on Synchronization Protocol: Analysis Using OMNeT++ Simulations", ZINC 2018, Novi Sad **(M33)**

2. Miladin Sandić, Ivan Velikić, Aleksandar Jakovljević, "Calculation of Number of Integration Cycles for Systems Synchronized Using the AS6802 Standard", ZINC 2017, Novi Sad **(M33)**

3. Miladin Sandić, Nikola Teslić, Ivan Velikić, "Bandwidth Utilization in Deterministic Networks", ZINC 2016, Novi Sad **(M33)**

4. Miladin Sandić, Ivan Velikić, "Implementation of Frames Scheduling in Mixed-Critical Networks", ZINC 2016, Novi Sad **(M33)**

5. Miladin Sandić, Bogdan Pavković, Dušan Živkov, Branislav Todorović, "Uticaj dužine integracionog ciklusa na međusobno usklađivanje časovnika u TTEthernet mrežama", TELFOR 2018, Beograd **(M63)**

6. Miladin Sandić, Ivan Velikić, Aleksandar Bilbija, Milena Milošević, "Analiza principa komunikacije u TTEthernet mrežama", ETRAN 2017, Kladovo **(M63)**

7. Miladin Sandić, Aleksandar Bilbija, Ivan Velikić, "Analiza integracije različitih klasa saobraćaja u mixed-critical mrežama", TELFOR 2016, Beograd **(M63)**

8. Miladin Sandić, Gordana Velikić, Vladimir Davidović, Aleksandar Bilbija, "Analiza iskorištenosti linka u mixed-critical mrežama", ETRAN 2016, Zlatibor **(M63)**

9. Miladin Sandić, Bogdan Pavković, Nikola Teslić, "Procjena izvodljivosti primjene tolopogije tipa magistrala u TTEthernet mrežama", TELFOR 2019, Beograd **(M63)**

10. Miladin Sandić, Bogdan Pavković, Željko Lukač, Milena Milošević, "TTEthernet Synchronization: Fail-Arbitrary and Fail-Omission Failure Scenarios Simulation", INDEL 2020, Banja Luka **(M33)**

Овај Образац чини саставни део докторске дисертације, односно докторског уметничког пројекта који се брани на Универзитету у Новом Саду. Попуњен Образац укоричити иза текста докторске дисертације, односно докторског уметничког пројекта.

План третмана података

Назив пројекта/истраживања
Једно рјешење примјене генетског алгоритма за анализу протокола усклађивања времена у *TTEthernet* мрежама
Назив институције/институција у оквиру којих се спроводи истраживање
а) Факултет техничких наука, Универзитет у Новом Саду б) Институт РТ-РК, Нови Сад
Назив програма у оквиру ког се реализује истраживање
Истраживање се врши у оквиру израде докторске дисертације на студијском програму Рачунарство и аутоматика.
1. Опис података
1.1 Врста студије *Укратко описати тип студије у оквиру које се подаци прикупљају* Докторска дисертација 1.2 Врсте података **а) квантитативни** **б) квалитативни** 1.3. Начин прикупљања података а) анкете, упитници, тестови

б) клиничке процене, медицински записи, електронски здравствени записи

в) генотипови: навести врсту _____

г) административни подаци: навести врсту _____

д) узорци ткива: навести врсту_____

ђ) снимци, фотографије: навести врсту_____

е) текст, навести врсту _____

ж) мапа, навести врсту _____

з) остало: описати **Рачунарске симулације**_____

1.3 Формат података, употребљене скале, количина података

1.3.1 Употребљени софтвер и формат датотеке:

а) Excel фајл, датотека _____

b) SPSS фајл, датотека _____

c) PDF фајл, датотека _____

d) Текст фајл, датотека _____

e) JPG фајл, датотека _____

f) Остало, датотека .log, .txt_____

1.3.2. Број записа (код квантитативних података)

а) број варијабли _____**Велики број**_____

б) број мерења (испитаника, процена, снимака и сл.) _____**Велики број**_____

1.3.3. Поновљена мерења

а) да

б) не

Уколико је одговор да, одговорити на следећа питања:

а) временски размак између поновљених мера је _____

б) варијабле које се више пута мере односе се на _____

в) нове верзије фајлова који садрже поновљена мерења су именоване као _____

Напомене: _____

Да ли формати и софтвер омогућавају дељење и дугорочну валидност података?

а) Да

б) Не

Ако је одговор не, образложити _____

_____

2. Прикупљање података

2.1 Методологија за прикупљање/генерисање података

2.1.1. У оквиру ког истраживачког нацрта су подаци прикупљени?

а) експеримент, навести тип _____**рачунарске симулације**_____

б) корелационо истраживање, навести тип _____

ц) анализа текста, навести тип __**анализа доступне литературе**_____

д) остало, навести шта _____

2.1.2 Навести врсте мерних инструмената или стандарде података специфичних за одређену научну дисциплину (ако постоје).

_____

_____

2.2 Квалитет података и стандарди

2.2.1. Третман недостајућих података

а) Да ли матрица садржи недостајуће податке? Да **Не**

Ако је одговор да, одговорити на следећа питања:

а) Колики је број недостајућих података? _____

б) Да ли се кориснику матрице препоручује замена недостајућих података? Да Не

в) Ако је одговор да, навести сугестије за третман замене недостајућих података

_____

2.2.2. На који начин је контролисан квалитет података? Описати

Квалитет података је контролисан поређењем експерименталних и теоријских података.

_____

2.2.3. На који начин је извршена контрола уноса података у матрицу?

Контрола уноса података је изведена на основу експертног знања.

_____

3. Третман података и пратећа документација

3.1. Третман и чување података

*3.1.1. Подаци ће бити депоновани у **Репозиторијуму докторских дисертација на Универзитету у Новом Саду.***

3.1.2. URL адреса: https://www.cris.uns.ac.rs/searchDissertations.jsf

3.1.3. DOI _____

3.1.4. Да ли ће подаци бити у отвореном приступу?

*а) **Да***

б) Да, али после ембарга који ће трајати до _____

в) Не

Ако је одговор не, навести разлог _____

3.1.5. Подаци неће бити депоновани у репозиторијум, али ће бити чувани.

Образложење

_____

_____

3.2 Метаподаци и документација података

3.2.1. Који стандард за метаподатке ће бити примењен? **Стандард који примјењује Репозиторијум докторских дисертација Универзитета у Новом Саду.**

3.2.1. Навести метаподатке на основу којих су подаци депоновани у репозиторијум.

_____

_____

Ако је потребно, навести методе које се користе за преузимање података, аналитичке и процедуралне информације, њихово кодирање, детаљне описе варијабли, записа итд.

_____

_____

_____

_____

3.3 Стратегија и стандарди за чување података

3.3.1. До ког периода ће подаци бити чувани у репозиторијуму? _____

3.3.2. Да ли ће подаци бити депоновани под шифром? **Да** Не

3.3.3. Да ли ће шифра бити доступна одређеном кругу истраживача? **Да** Не

3.3.4. Да ли се подаци морају уклонити из отвореног приступа после извесног времена?

Да **Не**

Образложити

_____

_____

4. Безбедност података и заштита поверљивих информација

Овај одељак МОРА бити попуњен ако ваши подаци укључују личне податке који се односе на учеснике у истраживању. За друга истраживања треба такође размотрити заштиту и сигурност података.

4.1 Формални стандарди за сигурност информација/података

Истраживачи који спроводе испитивања с људима морају да се придржавају Закона о заштити података о личности *(https://www.paragraf.rs/propisi/zakon_o_zastiti_podataka_o_licnosti.html)* и одговарајућег институционалног кодекса о академском интегритету.

4.1.2. Да ли је истраживање одобрено од стране етичке комисије? Да **Не**

Ако је одговор Да, навести датум и назив етичке комисије која је одобрила истраживање

_____

4.1.2. Да ли подаци укључују личне податке учесника у истраживању? Да **Не**

Ако је одговор да, наведите на који начин сте осигурали поверљивост и сигурност информација везаних за испитанике:

а) Подаци нису у отвореном приступу

б) Подаци су анонимизирани

ц) Остало, навести шта

_____

_____

5. Доступност података

5.1. Подаци ће бити

а) јавно доступни

б) доступни само уском кругу истраживача у одређеној научној области

ц) затворени

Ако су подаци доступни само уском кругу истраживача, навести под којим условима могу да их користе:

_____

_____

Ако су подаци доступни само уском кругу истраживача, навести на који начин могу приступити подацима:

_____

5.4. Навести лиценцу под којом ће прикупљени подаци бити архивирани.

Ауторство - некомерцијално

6. Улоге и одговорност

6.1. Навести име и презиме и мејл адресу власника (аутора) података

Миладин Сандић, miladin.sandic@rt-rk.com

6.2. Навести име и презиме и мејл адресу особе која одржава матрицу с подацима

Миладин Сандић, miladin.sandic@rt-rk.com

6.3. Навести име и презиме и мејл адресу особе која омогућује приступ подацима другим истраживачима

Миладин Сандић, miladin.sandic@rt-rk.com