



UNIVERZITET SINGIDUNUM U BEOGRADU
FAKULTET ZA INFORMATIKU I RAČUNARSTVO

**Primena Katalanovih brojeva i nekih
kombinatornih problema u kriptografiji**

– Diplomski rad –

Mentor:
Doc. dr Saša Adamović

Student:
Muzafer Saračević
Br. indeksa: 2016/203426

Beograd, 2017.

Primena Katalanovih brojeva i nekih kombinatornih problema u kriptografiji

Sažetak: U ovom radu ispitane su mogućnosti primene Katalanovih brojeva u kriptografiji. Navedena je primena odgovarajućih kombinatornih problema (*LatticePath*, *Ballot* problem, stek permutacija i *Balanced Parentheses*) u postupku šifrovanja i dešifrovanja fajlova i teksta. U radu su detaljno ispitana svojstva Katalanovih brojeva i njihova povezanost sa navedenim kombinatornim problemima. Primijenjena je autorska metoda koja se odnosi na dekompoziciju Katalanovih brojeva u postupku dinamičkog generisanja ključeva koristeći postupak memoizacije i dinamičko programiranje. Pored toga, u radu su prikazane i neke mogućnosti primene Katalanovih brojeva u oblasti steganografije, odnosno u postupku skrivanja informacija u tekstu. Na kraju rada, opisana je Java aplikacija koja realizuje generisanje ključeva sa svojstvom Katalanovog broja. Java aplikacija omogućava šifrovanje i dešifrovanje teksta na osnovu generisanog ključa i kombinatornog problema kretanja u celobrojnoj mreži. Imajući u vidu da koristimo Katalanove brojeve za generisanje ključeva, u okviru ekperimentalnog dela diplomskog rada primijenjena je *NIST* statistička baterija testova radi procene kvaliteta generisanih ključeva.

Ključne reči: Kriptografija, Katalanovi brojevi, Kombinatorni problemi, Šifrovanje teksta, Generisanje kriptoloških ključeva.

Application of Catalan numbers and some combinatorial problems in cryptography

Abstract: In this paper are examined the possibilities of applying Catalan numbers in cryptography. Also it is given the application of appropriate combinatorial problems (*Lattice Path*, *Ballot* Problem, *Stack* permutations and *Balanced Parentheses*) in encryption and decryption of files and plaintext. The paper analyzes the properties of Catalan numbers and their relation to these combined problems. Applied copyright method is related to the decomposition of Catalan numbers in the process of efficient generation of keys. The aforementioned method uses memoization and dynamic programming. Beside that, in paper are presented some possibilities of application Catalan numbers in the field of steganography and in the process of hiding information in the plaintext. At the end of the paper is shown Java application which enables key generation with the characteristic of the Catalan numbers. Java application allows encryption and decryption of plaintext based on the generated key and combinatorial problems of movement in integer network. Considering that, we are using the Catalan numbers for generating keys, within the experimental part of the diploma thesis was applied *NIST* statistical battery of tests for assessing the quality of generated keys.

Keywords: Cryptography, Catalan numbers, Combinatorial problems, Encrypting text, Generating cryptographic keys.

UNIVERZITET SINGIDUNUM U BEOGRADU
FAKULTET ZA INFORMATIKU I RAČUNARSTVO
Danijelova 32, Beograd.

Kandidat: MUZAFER SARAČEVIĆ

Broj indeksa: 2016203426

Studijski program: Informatika i računarstvo

**Tema: PRIMENA KATALANOVIH BROJEVA I NEKIH KOMBINATORNIH
PROBLEMA U KRIPTOGRAFIJI**

Datum odobrenja rada:

Beograd, _____._____._____.

MENTOR

DEKAN

SADRŽAJ

1	UVODNA RAZMATRANJA	5
1.1	Predmet istraživanja	5
1.2	Ciljevi rada	6
1.3	Struktura rada	7
2	PREGLED U OBLASTI ISTRAŽIVANJA	8
2.1	Slična istraživanja na polju primene Katalanovih brojeva u kriptografiji	8
2.2	Slična istraživanja na polju primene Katalanovih brojeva u steganografiji	9
2.3	Slična istraživanja na polju primene kombinatornih problema u kriptografiji	10
3	TEORIJSKE OSNOVE ISTRAŽIVANJA	11
3.1	Osnovna svojstva Katalanovih brojeva	12
3.2	Algoritam za dekompoziciju Katalanovih brojeva	14
3.3	Primena dekompozicije Katalanovih brojeva u generisanju ključeva	17
3.4	Primena Katalanovih brojeva i kombinatornih problema u šifrovanju fajlova	20
3.4.1	Metod <i>Lattice Path</i> – kretanje kroz celobrojnu mrežu	20
3.4.2	<i>Ballot</i> notacija, <i>Stack</i> permutacije i <i>Balanced Parentheses</i>	25
3.5	Primer primene dekompozicije Katalanovih brojeva u steganografiji	30
4	EKSPERIMENTALNI RAD SA STUDIJOM SLUČAJA	34
4.1	Postupak za generisanje ključeva, šifrovanje i dešifrovanje	34
4.2	Struktura Java izvornog koda i GUI elementi aplikacije	35
4.3	NIST statistička baterija testova	40
5	ZAKLJUČNA RAZMATRANJA	47
5.1	Sumirani ciljevi rada i ostvareni rezultati	47
5.2	Predlog za budući rad	48
6	LITERATURA	49
7	PRILOG: JAVA SOURCE CODE	51
7.1	Klasa <i>CatalanCrypto</i>	51
7.2	Klasa <i>CatalanNumbers</i>	53
7.3	Klasa <i>CatalanDecomposition</i>	54

1 UVODNA RAZMATRANJA

1.1 Predmet istraživanja

U većini simetričnih kriptografskih postupaka koriste se kombinatorne šeme zasnovane na permutacijama. Važno mesto u ovim postupcima, naročito u generisanju ključeva, ima grana matematika koja se bavi teorijom brojeva kao i širih klasa problema koji proističu iz ove oblasti.

Osnivač matematičke teorije informacije, Klod Elvud Šenon (engl. *Claude Elwood Shannon*) u svom radu "*A mathematical theory of communication*" (1948), diskutovao je o sigurnosti šifarskih sistema i pokazao da uz izbor odgovarajućih ključeva, kriptovana poruka može ipak sa praktične strane biti sigurna, pod uslovom da su matematičke operacije potrebne za probijanje ključa izuzetno kompleksne.

Uloga složenosti izračunavanja u asimetričnim sistemima ukazuje na važnost teorije složenosti, jedne od grana teorije formalne izračunljivosti. Primer je poznati problem $P = NP$, ili dokaz da je neki određen algoritam teško rešiv ili NP kompletan. Odnos između klasa kompleksnosti P i NP predstavlja nerešen problem teorijskog računarstva. Smatra se da je to najvažniji problem u ovoj naučnoj oblasti. Drugim rečima, vreme njegovog rešavanja eksponencijalno raste u odnosu na veličinu ulaznih podataka. Takav slučaj je i sa postupkom generisanja Katalanovih brojeva.

U ovom diplomskom radu, pokušaćemo da utvrdimo koliko značajno mesto ima teorija brojeva u kriptologiji, prvenstveno u razvoju algoritama za generisanje pseudoslučajnih brojeva koji su neophodni za generisanje ključeva. Teorija brojeva kod asimetričnih sistema ima važno mesto ne samo u generisanju ključeva već i u dizajnu samog kriptološkog algoritma, ali i u kriptanalizi [21,22].

Predmet istraživanja, u ovom diplomskom radu, odnosi se na ispitivanje svojstava Katalanovih brojeva i mogućnost njihove primene u kriptografiji. Pored toga, predmet istraživanja je i primena odgovarajućih kombinatornih problema, koji se baziraju na svojstvima Katalanovih brojeva, u šifrovanju i dešifrovanju fajlova i teksta. Ideja se javila na osnovu naših ranijih istraživanjima u oblasti teorije brojeva, kombinatornih problema i Katalanovih brojeva.

U našem radu „*Decomposition of Catalan numbers and Convex Polygon Triangulations*“ (objavljen 2014. godine u International Journal of Computer Mathematics, izdavača Taylor and Francis: UK), data je metoda za dekompoziciju Katalanovih brojeva, koja koristi postupak memoizacije i dinamičko programiranje. Na ovaj način, dajemo efikasnu metodu za generisanje svih Katalanovih brojeva za velike osnove. Inače, generisanje svih Katalanovih brojeva za velike osnove n , je izuzetno zahtevan (vremenski i memorijski) i složen problem.

U našem drugom radu „*Construction and Notation of Convex Polygon Triangulation based on ballot problem*“ (objavljen 2014. godine u ROMJIST- Journal of Information Science and Technology) data su razmatranja koja se odnose na povezanost Katalanovih brojeva i *Ballot* kombinatornog problema (problem glasanja), zatim problema *LatticePath*

(diskretna rešetka ili problem puteva u celobrojnoj mreži) i *Stack* permutacija. Upravo ovi navedeni kombinatorni problemi će biti razmatrani u ovom diplomskom radu i primenjeni u postupku šifrovanja teksta.

U našem trećem radu „*Alphanumeric notation for one Data Structure in Software Engineering*“ (objavljen 2014. godine u *Acta Polytechnica Hungarica: Journal of Applied Sciences*) takođe se razmatraju mogućnosti reprezentacije Katalanovih brojeva na osnovu kombinatornog problema *Balanced Parentheses* (problem balansiranih ili uparenih zagrada). U ovom diplomskom radu će biti prikazan i primer šifrovanja teksta na osnovu problema balansiranih zagrada. Bitno je napomenuti da navedeni kombinatorni problemi, da bi bili funkcionalni i mogući u postupku šifrovanja, ključ koji se koristi mora imati svojstva Katalanovih brojeva, inače postupak šifrovanja na osnovu navedenih kombinatornih problema nije izvodljiv.

1.2 Ciljevi rada

Cilj ovog rada je obrazloženje i razmatranje već postojećih znanja o primeni Katalanovih brojeva i kombinatornih problema sa akcentom na njihovu primenu u postupku šifrovanja i dešifrovanja teksta. Cilj je i da se primene neke autorske metode, poput *Dekompozije Katalanovih brojeva* i šifrovanja na osnovu problema puteva u celobrojnoj mreži (*LatticePath*). Pored toga, biće prikazane i neke mogućnosti primene Katalanovih brojeva i njihovih svojstava u postupku skrivanja informacija u tekstu, tačnije u oblasti steganografije.

Katalanovi brojevi nalaze široku primenu u rešavanju mnogih kombinatornih problema. U monografiji [1] su navedene konkretne primene ovih brojeva sa mogućim rešenjima kada su u pitanju reprezentacije Katalanovih brojeva. Autor u svom praktikumu [2] navodi skup zadataka koji opisuju preko 60 različitih interpretacija Katalanovih brojeva. Mogu se nabrojati neke interpretacije: *binarna stabla*, *triangulacije poligona*, *stek permutacije*, *problem uparenih zagrada*, *Ballot problem*, *problem kretanja kroz celobrojnu mrežu* itd. Svi navedeni kombinatorni problemi se mogu rešiti na osnovu vrednosti koje poseduju svojstva Katalanovih brojeva, tačnije rešenje ovih kombinatornih problema se krije u primeni ovih brojeva.

U radovima [3,4,5,6,7,9] mogu se videti konkretne primene u rešavanju nekih kombinatornih problema i u računarskoj geometriji. U doktorskoj disertaciji „*Metode za rešavanje problema triangulacije poligona i njihova implementacija*“ (branjena 2013. godine na Prirodno-matematičkom fakultetu Univerziteta u Nišu) predstavljene su nove metode i tehnike u rešavanju nekih problema u oblasti računarske geometrije, upravo na osnovu Katalanovih brojeva i kombinatornih problema koji će se i u ovom diplomskom radu primeniti, ali u oblasti kriptografije.

Dodatni cilj nam je da damo predlog primene Katalanovih brojeva u generisanju ključeva, tačnije ovi brojevi se mogu koristiti u svojstvu generatora pseudoslučajnih brojeva. Pomoću Katalanovih brojeva se realizuje osnovna ideja, a to je generisanje duge i nepredvidljive sekvence simbola iz nekog alfabeta (npr. binarnog) na osnovu kratkog ključa izabranog na slučajan način (osnova n). Šifrovanje se može realizovati u kombinaciji sa

raznim kombinatornim problemima koji se baziraju i čije se rešenje krije u pomenutim brojevima. Na ovaj način se postižu zavidne performanse sistema za kriptovanje.

Imajući u vidu činjenice da je kriptografija veoma dinamična oblast, da je aktuelna i da je veoma rasprostranjena, ovim radom su obuhvaćeni samo neki njeni osnovni matematički koncepti i dat je mali doprinos kada je reč o primeni teorije brojeva u oblasti kriptografije.

1.3 Struktura rada

Diplomski rad sadrži pet poglavlja, literaturu i priloge. U nastavku, u drugom poglavlju ovog diplomskog rada biće predstavljena neka slična istraživanja na polju primene Katalanovih brojeva u kriptografiji i steganografiji. Pored toga, biće navedena i neka slična istraživanja na polju primene kombinatornih problema u kriptografiji.

U trećem poglavlju ovog diplomskog rada biće navedene teorijske osnove istraživanja, tačnije osnovna svojstva Katalanovih brojeva i algoritam za dekompoziciju Katalanovih brojeva. Pored toga, u ovom poglavlju će biti razmatrana primena metode dekompozicije u generisanju validnih binarnih zapisa ključeva i primena kombinatornih problema u šifrovanju fajlova. Pre svega, biće akcenat stavljen na primenu metode kretanja kroz diskretnu rešetku odnosno celobrojnu mrežu (eng. *Lattice Path*) u šifrovanju teksta, a biće i date alternative u šifrovanju a to su *Ballot problem*, *Stack permutacije* i *Balanced Parentheses*. Na kraju trećeg poglavlja biće dat jedan primer primene dekompozicije Katalanovih brojeva u steganografiji.

U četvrtom poglavlju dat je eksperimentalni rad sa studijom slučaja, koji će obuhvatiti konkretne algoritme za šifrovanje i dešifrovanje i biće opisana struktura i način rada Java aplikacije, *GUI* elementi i njen izvorni kod. Imajući u vidu da koristimo Katalanove brojeve za generisanje ključeva, primenićemo *NIST* statističku bateriju testova radi procene kvaliteta generisanih binarnih sekvenci koje nam predstavljaju ključeve.

U petom poglavlju će biti navedena zaključna razmatranja sa osvrtom na sumirane ciljeve rada i ostvarene rezultate, a biće dati i predlozi za budući rad. Na kraju diplomskog rada, navedena je korišćena literatura i prilozi sa Java izvornim kodom.

2 PREGLED U OBLASTI ISTRAŽIVANJA

2.1 Slična istraživanja na polju primene Katalanovih brojeva u kriptografiji

Primenjena teorija brojeva ima mnogo primena u kriptografiji, naročito celobrojne sekvence. Raniji kriptografski algoritmi su dizajnirani korišćenjem celobrojnih sekvenci Fibonačijevog niza i Lukas brojeva.

U radu [17] predloženi su kriptografski algoritmi zasnovani na celobrojnim sekvencama Katalanovih brojeva gde su predložene nove metode enkripcije (šifrovanja). U predloženoj metodi enkripcije, korišćenjem Katalanovih brojeva, veliki slučajni broj " n " je dogovoren kao tajni ključ od strana koje komuniciraju. Binarne vrednosti u Katalanovom broju C_n koje odgovaraju dogovorenom tajnom ključu (odnosno osnovi n) koriste se za enkripciju/dekripciju poruke. U navedenom radu, predložene metode enkripcije koriste logičku *XOR* operaciju na bitove *ASCII* binarnih kodova poruke. Poruka je podeljena u blokove podataka veličine " m " karaktera, gde je " m " broj binarnih parova 0 i 1 u Katalanovom broju C_n . Tačnije, uočeno je da u navedenom radu binarni zapisi Katalanovih brojeva i poruke kao i *XOR* operacija između njih, simuliraju *OTP* (*One-time password*) šifru.

Sa aspekta kriptanalize, predloženi algoritam koristi Katalanove brojeve gde je ustanovljeno da su otporni na većinu poznatih vrsta napada. Identični karakteri u običnom tekstu su kodirani različitim karakterima. Samim tim, „napad grubom silom“ i potpuna pretraga ključa se teško izvršava. Vreme za šifrovanje ili dešifrovanje je nezavisno od karaktera u bloku podataka.

U radu [23] predstavljena je napredna tehnika za šifrovanje bazirana na Katalanovim brojevima. Ova tehnika kombinuje karakteristike supstitucije i transpozicije. U navedenom radu je predložena tehnika pomeranja svakog bita u levo, a pomereni podatak je dopunjen tako da se svaki bit šifrata koji se generiše menja. Generisanje ključeva se vrši na osnovu niza Katalanovih brojeva. Osnovna vrednost ključa je fiksirana i definisana od strane korisnika. Pored toga, svaka naredna vrednost ključa je duplo veća od prethodne (rekurzivno generisanje). Znači, u navedenom radu se ističe još jedna bitna primena Katalanovih brojeva a to je primena u dizajnu kriptosistema sa tehnikama generisanja rekurzivnog ključa. Cilj algoritma je da se kriptanaliza oteža a da se algoritam šifrovanja ojača.

Katalanovi brojevi imaju svojstvo rekurzivnosti i njihovo generisanje se može efikasno realizovati primenom dinamičkog programiranja. Dinamičko programiranje predstavlja metod kojim se smanjuje vreme izvršavanja onih problema u kojima se zahteva traženje optimalne podstrukture i koji imaju potprobleme koji se ponavljaju.

U radu [16] se takođe navodi velika primena teorija brojeva u kriptografiji. U ovom radu, dat je novi pristup za šifrovanje podataka zasnovan na *ECC* metodi, odnosno na kriptografiji eliptičnih krivih (*Elliptic Curve Cryptography*) koristeći Katalanove brojeve. U radu je predstavljen novi algoritam u oblasti kriptografije eliptičnih krivih koji je baziran na Katalanovim brojevima. Kriptosistem baziran na kriptografiji eliptičnih krivih je novi trend

kriptografije sa javnim ključem. Algoritam koristi koncept jednake podele običnog teksta i ključa u postupku šifrovanja.

U radu [34] je dat predlog modifikacije Vižnerove šifre (*Vigenere*), upravo pomoću Katalanovih brojeva i dvostruke transpozicije. U navedenom radu, ova metoda se bazira na Katalanovim brojevima i predstavljena je kao matematička metoda koja se koristi da bi se kreirao inicijalni ključ sa akcentom na što jače karakteristike ključa (balansiranost i nepredvidivost). Sa aspekta kriptanalize, ističe se da je ovakav ključ teže otkriti, upravo zbog specifičnih svojstava i niza sekvenci koje odgovaraju Katalanovim brojevima.

2.2 Slična istraživanja na polju primene Katalanovih brojeva u steganografiji

Steganografija je nauka koja se bavi sakrivanjem informacija u drugim podacima, tako da je i samo postojanje šifrata sakriveno u nosiocu podatka. Cilj svake kripto-metode je da pronađe najbrži i najpovoljniji način da prenesena informacija bude sačuvana. Jedna od takvih metoda koja se danas koristi se zasniva na upotrebi steganografije.

Još jedna interesantna primena Katalanovih brojeva je data u radu [18], gde su predložene nove tehnike skrivanja podataka pomoću sekvenci Katalanovih-Lukas brojeva. Konkretno, u navedenom radu se koriste dva generatora slučajnih brojeva. Jedan generiše Katalanove a drugi Lukas sekvence. U predloženoj tehnici, jedan niz predstavlja skup konstanti a drugi niz promenljivih koje su uslovljene datim konstantama. Ovakav način predstavlja poboljšanje u odnosu na postojeće tehnike skrivanja podataka. Ovaj rad predstavlja novi pristup za skrivanje podataka kako u tekstu tako i u segmentima slika (u pikselima). Pored navedene tehnike, dat je i modifikovani stegano-grafički algoritam pomoću Katalan-Lukas serija koji daje visoku bezbednost i pouzdanost.

Pored navedenog rada, tehnike u steganografiji slika upotrebom poznatih sekvenci brojeva (među kojima su i Katalanovi brojevi) su date i u radu [19]. U navedenom radu je predloženo nekoliko metoda za steganografiju slika upotrebom Katalanovih i Lukas brojeva i data su eksperimentalna ispitivanja koja daju bolje rezultate od tehnika u kojima se koriste Fibonačijevi brojevi. Predložene metode koriste kombinaciju Lukas-Katalanovi-Fibonačijevi brojevi, kao i kombinaciju Katalanovih-Lukas brojeva koji su superiorniji u odnosu na postojeću Fibonačijevu tehniku skrivanja podataka.

U klasičnoj *LSB* (*Least significant bit*) tehnici moguće je ugraditi tajne podatke samo u prvih nekoliko bit-planova, jer se kvalitet slike radikalno pogoršava kada se vrši ugrađivanje podataka u višim bit-planovima. U poslednjim decenijama, tehnike steganografije se primenjuju u različitim tipovima datoteka. Potrebe zaštita autorskog prava u muzici i fotografiji su primorale mnoge softverske kompanije da razviju mnoge steganografske sisteme koje koriste u različitim oblastima.

Rekurzivno skrivanje informacija u vizuelnoj kriptografiji je dato u radu [26]. Predložen je kriptografski metod kojim se vrši skrivanje podataka u nekom skupu slika, a rekonstrukcija zaštićenih, odnosno kriptovanih podataka se vrši direktnim odnosno vizuelnim pregledom. Ovakvi algoritmi vizuelne kriptografije se baziraju na jakim matematičkim osnovama, karakteriše ih visok stepen sigurnosti i otpornosti, sa akcentom na mogućnost rešavanja problema deljenja tajne.

2.3 Slična istraživanja na polju primene kombinatornih problema u kriptografiji

Na osnovu mnogih naučnih i stručnih radova, koji su objavljeni u renomiranim međunarodnim časopisima, došli smo da saznajemo da mnogi kombinatorni problemi nalaze primenu kako u kriptografiji tako i u postupku kriptanalize. Među najčešće korišćenim kombinatornim problemima su problem balansiranih ili uparenih zagrada, primena diskretne rešetke ili celobrojne mreže, stek permutacije, problem glasanja (*Ballot*) i sl. U radu [24] su date konkretne primene kombinatornih problema u kriptografiji i kriptanalizi.

U radu [27] se upravo navodi činjenica da kombinatorni *Ballot* problem ili problem glasanja nalazi veliku primenu u kriptografiji. U navedenom radu se opisuju teorije i implementacija elektronskog sistema za glasanje sa akcentom na praktičnu šemu glasanja bazirane na tzv. Gomomorfnoj enkripciji.

U radu [28] predstavljena je nova metoda glasanja sa interesantnim bezbednosnim svojstvima. Autori su pokušali da ispituju da li se mogu postići iste sigurnosne osobine predloženih kriptografskih protokola glasanja, ali bez upotrebe ikakve kriptografije.

U radu [25] je predloženo poboljšanje kriptografije bazirane na principu rešetaka korišćenjem metode CRT (*Chinese Remainder Theorem*). U radu su kritički analizirana dva postojeća kriptosistema bazirana na principu rešetki (*GGH and Micciancio's cryptosystem*).

3 TEORIJSKE OSNOVE ISTRAŽIVANJA

Katalanovi brojevi predstavljaju niz brojeva koji se prvenstveno koristio u geometriji i u rešavanju mnogih kombinatornih problema. Otkrio ih je Leonard Ojler (*Leonhard Euler*), u postupku traženja opšteg rešenja za broj različitih načina na koji se jedan mnogougao (poligon) od n stranica može podeliti na trouglove. Pritom je trebalo voditi računa da se ne koriste dijagonale mnogougla koje se međusobno seku. Međutim, ovi brojevi su dobili ime po belgijskom matematičaru Eugenu Čarlsu Katalanu (*Eugene Charles Catalan*) koji je otkrio vezu između ovih brojeva i problema korektnih nizova n -parova zagrada.

Katalanovi brojevi C_n , $n > 0$, predstavljaju niz prirodnih brojeva koji se pojavljuje kao rešenje velikog broja poznatih kombinatornih problema. Neki od njih su: *broj mogućih puteva u diskretnoj rešetki dimenzije $n \times n$* , *broj mogućih zapisa u obliku n -balansiranih zagrada*, *stek permutacije*, *Ballot problem*, *binarna stabla*, *triangulacije poligona* itd.

Katalanovi brojevi se definišu kao [1]:

$$C_n = \frac{(2n)!}{(n+1)!n!} \quad (1)$$

$$= \frac{1}{n+1} \binom{2n}{n}, n \geq 0 \quad (2)$$

Drugi način za definisanje Katalanovih brojeva je:

$$\begin{aligned} \binom{2n}{n} - \binom{2n}{n-1} &= \frac{(2n)!}{(n!)^2} - \frac{(2n)!}{(n-1)!(n+1)!} \\ &= \frac{(2n)!}{n!(n+1)!} \\ &= C_n \end{aligned} \quad (3)$$

Tabela 1 prikazuje Katalanove brojeve za osnove $n \in \{1, 2, \dots, 30\}$, koji se izračunavaju po formuli (1) ili (3).

Tabela 1. Vrednosti prvih 30 Katalanovih brojeva

n	C_n	n	C_n	n	C_n
1	1	11	58,786	21	24,466,267,020
2	2	12	208,012	22	91,482,563,640
3	5	13	742,900	23	343,059,613,650
4	14	14	2,674,440	24	1,289,904,147,324
5	42	15	9,694,845	25	4,861,946,401,452
6	132	16	35,357,670	26	18,367,353,072,152
7	429	17	129,644,790	27	69,533,550,916,004
8	1,430	18	477,638,700	28	263,747,951,750,360
9	4,862	19	1,767,263,190	29	1,002,242,216,651,368
10	16,796	20	6,564,120,420	30	3,814,986,502,092,304

U ovom diplomskom radu, upotrebićemo Katalanove brojeve u svojstvu generatora ključeva za šifrovanje i dešifrovanje. Iz Tabele 1, možemo videti da je n osnova za generisanje ključeva a C_n određuje broj validnih ključeva u toj osnovi, tj. onih vrednosti koje zadovoljavaju svojstvo Katalanovog broja (prostor ključeva).

Na primer, za osnovu $n=30$ imamo prostor ključeva $C_{30}=3\ 814\ 986\ 502\ 092\ 304$, odnosno vrednosti koje zadovoljavaju svojstvo Katalanovog broja. Sa povećavanjem osnove n , drastično se povećava prostor ključeva. Tabela 1 prikazuje vrednosti za prvih 30 Katalanovih brojeva. Da bi obezbedili što jači tj. otporniji mehanizam šifrovanja na kriptanalizu, za ključeve je potrebno birati uglavnom vrednosti čije su osnove veće od 30. Primenom softverskog rešenja u Java programskom jeziku, prikazaćemo broj vrednosti za osnove n od 140 do 150.

```
Catalan(140)= 656376399024616169349253607753345435388942038466586811952779656067170646392272840
Catalan(141)= 2597771382055171036438595264488592497806939617029730903644099765561619037129981240
Catalan(142)= 10282088127575012633735978459444359117193900861809983856381541729425708916192792880
Catalan(143)= 40699932171651091675204914735300588172225857577997852764843602678976764459929805150
Catalan(144)= 161115593562260183597018076262500259385225118963936327496691227156776984827584194180
Catalan(145)= 637841185472509493966277041641953081675754238090104091048544721209706145413312768740
Catalan(146)= 252533040778911922100934175670487546622645554887350891090156651320061065513932186440
Catalan(147)= 9998943371381242321023474793439574481139884832189105555262377011307809353994353116580
Catalan(148)= 39593131470570019928884900188787576804513637926117934749025519709205419589642069387800
Catalan(149)= 156788800623457278918384204747598804145874006187427021606141058048453461574982594775688
Catalan(150)= 620925183926009621146978506218967449531342090729015621989883130549504437230725772687824
```

3.1 Osnovna svojstva Katalanovih brojeva

U prethodnoj sekciji, videli smo kako se računa ukupan broj vrednosti koje zadovoljavaju svojstvo Katalanovog broja za dato n . Sada ćemo analizirati vrednosti koje se generišu u skupu C_n . Za potrebe provere validnosti Katalanovog broja koristićemo njihove binarne zapise. Osnovno svojstvo koje mora biti ispunjeno je *svojstvo balansiranosti bitova* u binarnom zapisu za određeni broj iz skupa C_n (ovo svojstvo ćemo nazvati svojstvo *bit-balansiranosti*).

Osnovno svojstvo Katalanovih brojeva: Broj može biti označen kao Katalanov broj ukoliko se njegov binarni oblik sastoji od jednakog broja bitova "1" i "0" i počinje bitom "1".

Ukoliko se binarni zapis Katalanovog broja poveže sa nekim drugim načinom zapisivanja, najčešće sa zapisom balansiranih zagrada, onda "1" predstavlja otvorenu zagradu i "0" predstavlja zatvorenu zagradu, pa se može reći da se svaka otvorena zagrada zatvara, odnosno svaki bit 1 ima svoj par a to je bit 0.

Takođe, binarni zapis Katalanovog broja se može predstaviti i u vidu stek permutacija ili *Ballot* zapisa. Reprezentacija pomoću Stack permutacija bit 1 tretira kao *PUSH* komandu a bit 0 kao *POP* komandu. O tome više u narednim sekcijama.

Primer 3.1: Za $n=3$, na osnovu formula (1) ili (3), imamo skup $C_3=5$ vrednosti koje zadovoljavaju svojstvo Katalanovog broja. To su brojevi $C[3]=\{42, 44, 50, 52, 56\}$, odnosno na osnovu njihovih binarnih zapisa *101010*, *101100*, *110010*, *110100*, *111000* utvrđujemo njihovo svojstvo koje odgovara Katalanovom broju, a to je svojstvo bit-balansiranosti.

Ukoliko posmatramo navedene binarne zapise datih Katalanovih brojeva, možemo videti da svaki binarni zapis počinje bitom 1 i da je broj jedinica i nula jednak, što znači da je balansiranost bitova 0 i 1 glavna osobina svakog Katalanovog broja. Pored toga, možemo zaključiti da osnova n pokazuje broj parova 1 i 0. Na taj način možemo znati dužinu ključa na osnovu date osnove. U ovom primeru, osnova je 3, što znači da će dužina ključa biti 6 bita, odnosno imaće tri para "1" i "0", koji zadovoljavaju svojstvo balansiranosti.

Primer 3.2: Analiziramo vrednost broja $(1142920503054772802)_{10} = (111111011100011110000010010001011111000101110010101001000010)_2$

Iz ovog zapisa možemo zaključiti da je ispunjen uslov bit-balansiranosti, što znači da vrednost 1142920503054772802 poseduje svojstvo Katalanovog broja. Sada ćemo utvrditi koja je osnova za generisanje ove vrednosti. Dužina binarnog zapisa ove vrednosti je 60 bita, što znači da se u ovom zapisu nalazi 30 parova 1 i 0. Na osnovu toga, možemo odrediti osnovu kojoj pripada navedeni Katalanov broj, a to je $n=30$.

Primer 3.3: Sada ćemo analizirati jedan zapis koji ne odgovara svojstvu bit-balansiranosti. Primeri brojeva iz skupa C_3 koji imaju svojstvo balansiranosti su 42, 44. Sada ćemo proveriti kakvo je stanje sa brojem između njih, a to je 43. Njegov binarni zapis 101011. Možemo odmah uočiti da je ovde prekršen uslov balansiranosti i to u šestom bitu. Isti slučaj je i sa brojem 45. Njegov binarni zapis 101101. Možemo uočiti da je prekršen uslov balansiranosti i to u četvrtom bitu. Ovi brojevi se ne mogu upotrebiti za šifrovanje teksta na osnovu nekog kombinatornog problema koji se bazira na svojstvu Katalanovog broja. O tome više u narednim sekcijama.

Zapis Katalanovog broja se, pored binarnog oblika, može predstaviti na druge načine, odnosno može se reprezentovati preko mnogih kombinatornih problema. Može se predstaviti u obliku uparenih zagrada "() () ())" ili u obliku *Ballot* zapisa "ABABAABB" ili grafički u vidu kretanja kroz celobrojnu mrežu (*LatticePath*) ili preko stek permutacija.

U studiji slučaja [33] analizirano je generisanje svih mogućih Katalanovih brojeva, gde se pokazalo da je to veoma zahtevan posao. Takođe, u našim autorskim radovima [12,13,14,15] vršena su testiranja generisanja svih brojeva za datu osnovu n , koji zadovoljavaju navedena svojstva Katalanovih brojeva. Na osnovu datih analiza, možemo izvesti neke osnovne karakteristike generisanih ključeva na osnovu svojstava Katalanovih brojeva:

1. Mora biti ispunjen uslov balansiranosti,
2. Može poslužiti kao generator pseudo-slučajnih brojeva,
3. Može se koristiti za realizaciju sekvencijalnih algoritama,
4. Ključ koji pripada osnovi n ima dužinu $2n$ -bita i na osnovu njega se može vršiti odgovarajućih n -permutacija.

Iz poslednje stavke sledi da, ako je n osnova za generisanje ključeva, onda je C_n ukupan broj različitih binarnih zapisa koji zadovoljavaju svojstvo Katalanovih brojeva. Na primer, za osnovu $n=30$ se izračunava $C_{30} = 3\ 814\ 986\ 502\ 092\ 304$ shodno navedenoj formuli (1) za izračunavanje Katalanovog broja. Znači, za 60-bitne ključeve postoji 3 814 986 502 092 304 validnih vrednosti koje zadovoljavaju uslov bit-balansiranosti tj. svojstvo Katalanovog broja.

U pokušaju da se pronađu svi ti Katalanovi brojevi i da se izvrši njihovo upisivanje na disku, potreban je prostor od 30.519.892.016.738.432 bajta odnosno 28.423.864 GB ili 27.757 TB. Znači, ovakav postupak je veoma zahtevan kada su memorijski resursi u pitanju.

Sa druge strane, ako želimo da pronađemo sve 60-bitne Katalanove brojeve i ako je za pristup svakom elementu iz skupa C_n potrebna 1 ms, vreme izvršavanja bi trajalo 120972 godina. Prosečno vreme će biti $120972 / 2 = 60486$ godina. Znači, ovakav postupak je veoma zahtevan i kada je vreme u pitanju.

Postavlja se pitanja, da li se za neku veću osnovu može uopšte i doći do kompletnog skupa svih Katalanovih brojeva (metodom potpune pretrage), pošto računarski resursi kao što su CPU vreme i memorija ograničavaju celokupan postupak generisanja.

Na primer, za osnovu od $n=64$, ukupan broj vrednosti koje zadovoljavaju svojstvo Katalanovog broja je 368 479 169 875 816 659 479 009 042 713 546 950 ključeva koji su 128-bitni. Potreban prostor za smeštanje svih ključeva je 2.680.973.133.886.531.228.176.366 TB, odnosno 2.618.137.826.061.065.652.515 PB. Ako je za pristup svakom ključu potrebna 1 ms, prosečno vreme da se dođe do svih ključeva je otprilike 5 842 198 913 556 200 milijardi godina, što je praktično neizvodljivo sa aspekta potrebnog realnog vremena i memorije.

U nastavku, predstavimo jedan metod koji može znatno ubrzati kompletno generisanje svih Katalanovih brojeva. Navedeni metod koristi tehniku dinamičkog programiranja i postupak memoizacije.

3.2 Algoritam za dekompoziciju Katalanovih brojeva

U našim radovima [5,12,13,14] analizirali smo Katalanove brojeve i utvrdili da poseduju svojstvo rekurzivnosti. Rekurzivna definicija Katalanovih brojeva važi za C_n , $n>0$, gde je $C_0=1$ i ona definiše odnos dva uzastopna Katalanova broja. Može se izraziti kao [1]:

$$C_n = \frac{4n-2}{n+1} C_{n-1}, n \geq 1 \quad (4)$$

Može se definisati odnos i za Katalanove brojeve koji nisu uzastopni:

$$C_n = \frac{(4n-2) \cdot (4n-6)}{(n+1)n} C_{n-2} \quad (5)$$

$$= \frac{(4n-2)(4n-6)(4n-10)}{(n+1)n(n-1)} C_{n-3} \quad (6)$$

...

$$= \frac{(4n-2)(4n-6)(4n-10) \cdots 6 \cdot 2}{(n+1)n(n-1) \cdots 3 \cdot 2} C_0 \quad (7)$$

$$= \frac{(2n-1)(2n-3)(2n-5) \cdots 3 \cdot 1}{(n+1)!} 2^n$$

$$\begin{aligned} &= \frac{(2n)!2^n}{(n+1)!2^n n!} = \frac{(2n)!}{(n+1)!n!} \\ &= \frac{1}{n+1} \binom{2n}{n} \end{aligned}$$

Dakle, ovo svojstvo Katalanovih brojeva nam omogućava da primenimo neke tehnike rešavanja problema koje se mogu svesti na rekurzivno rešavanje nezavisnih podproblema. Na taj način možemo efikasnije (brže) generisati ključeve sa što većom osnovom n .

Dinamičko programiranje se može definisati kao metod kojim se smanjuje vreme izvršavanja onih problema koji imaju potprobleme koji se ponavljaju i u kojima se zahteva traženje optimalne podstrukture, kao što će biti opisano u nastavku. U ovom postupku, potrebno je sačuvati rešenja onih problema koji su već rešeni, kako bi se ti rezultati mogli kasnije iskoristiti. Ovaj postupak se naziva memoizacija. Ukoliko je očigledno da rešeni potproblemi nisu više potrebni, mogu se odbaciti kako bi se sačuvao memorijski prostor. Znači, donekle rešavamo problem kada su u pitanju vreme i memorija.

Dinamičko programiranje se koristi kod potproblema koji se ponavljaju, kao i kod optimalnih podstrukture i memoizacije. Postoje dva pristupa:

1. *Odozgo na dole*: Problem se rastavi na potprobleme, pa se onda podproblemi reše i pamte se njihova rešenja, u slučaju njihove kasnije upotrebe. Ovaj pristup predstavlja kombinovanje rekurzije i memoizacije.
2. *Odozdo na gore*: Svi potproblemi se redom rešavaju i koriste za nalaženje većih. Ovaj pristup je bolji zbog čuvanja memorijskog prostora na steku, ali je ponekad teško odrediti koji su sve podproblemi potrebni.

U postupku dekompozicije Katalanovih brojeva [12] koristimo pristup *Odozgo na dole* uz primenu memoizacije. Memoizacija predstavlja kombinaciju dinamičkog programiranja i rekurzije. Sva rešenja se pamte u memoriji, tako da ne dolazi do rešavanja istog problema više puta. Metod dekompozicije važi za Katalanove brojeve gde je $n > 2$. U navedenom radu smo naveli efikasan metod razlaganja (dekompozicije) Katalanovih brojeva u vidu segmenata izraza $(2+i)$.

Algoritam na ulazu očekuje osnovu n a na izlazu dobijamo izraz koji predstavlja sumu segmenata oblika $(2+i)$ čija vrednost odgovara C_n , odnosno skupu Katalanovih brojeva. Metod memoizacije se pokazao kao efikasan i povoljan, a u nekim slučajevima je i brži od dinamičkog programiranja. To je zbog toga što postupkom dinamičkog programiranja rešavamo sve moguće potprobleme, a memoizacijom samo one koji su nam potrebni.

Algoritam 1. Dekompozicija Katalanovih brojeva [12]

ULAZ: n

1: Postavlja inicijalni izraz $expr(k) = (2+0)$, $k=2$.

2: *for* ($k=2$; $k \leq n$; $k++$)

2.1: Izračunava broj segmenata oblika $(2+i)$ u izrazu $expr(k)$ i vraća $count(k)$

2.1.1: Izračunava sumu u svakom segmentu oblika $(2+i)$ i vraća $sum(s)$

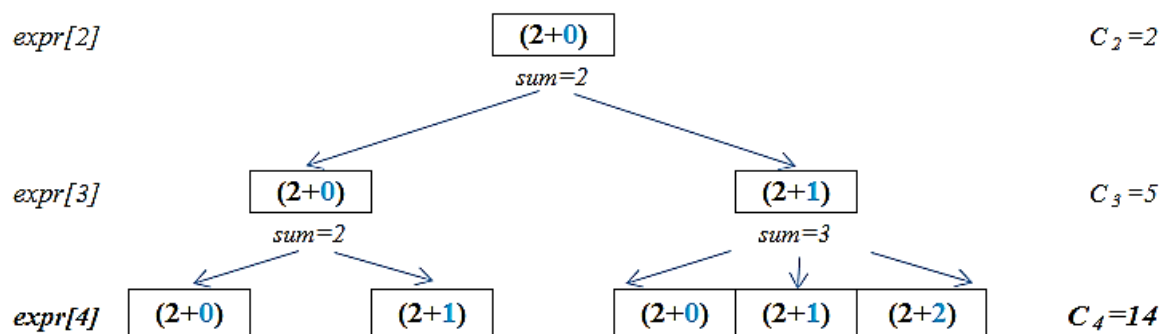
2.2: for (s=1; s<= count(k); s++)
 2.2.1: for (i=0; i<= sum(s); i++)
 2.2.1.1: kreira segment izraza oblika (2+i)
 2.2.1.2: spaja segmente s operacijom +

IZLAZ: Izraz $expr[n]$ za dekompoziciju za uneto n .

Primer 3.4: Za $n=4$ sledi da je $C_4=14$. U prvom koraku algoritma, postavlja se inicijalni izraz $expr[2]=(2+0)$. Znači, broj 2 se može predstaviti jednim inicijalnim segmentom oblika $(2+i)$ a to je $(2+0)$. U drugom koraku algoritma, za ovaj slučaj, postoje dva ciklusa. Prvo se izračunava koliko ima segmenata oblika $(2+i)$ u inicijalnom izrazu $expr[2]$, a broj takvih segmenata je 1. Suma u tom jednom segmentu je 2. Što znači da će naredni izraz $expr[3]$ imati dva segmenta oblika $(2+i)$. Znači dobija se $expr[3]=(2+0)+(2+1)$.

Sada se postavlja $expr[3]$ za inicijalni segment i isto se i za njega izračunava koliko ima segmenata oblika $(2+i)$, a taj broj je 2 dok su sume u njima redom 2 i 3. Što znači da će naredni izraz $expr[6]$ imati ukupno imati 5 segmenata oblika $(2+i)$.

Konačno, dobija se $expr[4]=(2+0)+(2+1)+(2+0)+(2+1)+(2+2) = 14 = C_4$



Slika 1. Grafička prezentacija dekompozicije

Algoritam 1 radi u kombinaciji sa metodom za proveru validnosti generisanih izraza. Na osnovu dobijenog izlaza iz Algoritma 1, utvrđuje se da li su zadovoljena dva glavna uslova.

Prvi uslov: Suma svih segmenata oblika $(2+i)$ mora biti jednaka Katalanovom broju (C_n) za koji se vrši dekompozicija:

$$C_n = \sum_{s=1}^{C_{n-1}} \sum_{i=0}^{seg(s)-1} (2+i), \quad n \geq 2 \quad (8)$$

gde $seg(s)$ predstavlja sume segmenata $s_1, s_2, \dots, s_{n-1}$.

Drugi uslov: Kao što važi za dva uzastopna Katalanova broja važi i za segmente dva uzastopna izraza a to je da se prethodni izraz mora naći u celosti dva puta u izrazu narednog nivoa. Tačnije, mora se zadovoljiti sledeća jednakost: $expr[n]=2 * expr[n-1] + rest$.

Na osnovu relacije (4), ostatak ($rest$) se može definisati

$$rest = C_n - 2C_{n-1} = \frac{4n-2}{n+1}C_{n-1} - 2C_{n-1} = C_{n-1}\left(\frac{4n-2}{n+1} - 2\right) \quad (9)$$

Na osnovu izraza relacije (8), sledi da je

$$rest = \sum_{s=1}^{C_{n-1}} \sum_{i=0}^{seg(s)-1} i \quad (10)$$

U Tabeli 2 prikazani su elementi skupa za osnove $n=\{2,3,4,5,6\}$ i broj segmenata koji ih određuje. Parametri A i B se mogu izraziti kao $A=count(seg[s]); B=sumAll(seg[s])=C_n$.

Tabela 2. Broj (A) i sume (B) segmenata u izrazu dekompozicije

n	<i>seg(s) - sume pojedinačnih segmenata oblika (2+i)</i>	A	B
2	{2}	1	2
3	{2,3}	2	5
4	{2,3,2,3,4}	5	14
5	{2,3,2,3,4,2,3,2,3,4,2,3,4,5}	14	42
6	{2,3,2,3,4,2,3,2,3,4,2,3,4,5,2,3,2,3,4,2,3,4,5,2,3,4,5,6}	42	132

Primer za proveru prvog uslova: Proveravamo na konkretnom primeru za C_4 da li je zadovoljen prvi uslov. Analizom dobijenog izraza utvrđujemo:

- I. $expr[4]=(2+0)+(2+1)+(2+0)+(2+1)+(2+2)$, izraz u segmentima oblika $(2+i)$
- II. $expr[4] = 2+3+2+3+4$, izraz sa sumama u segmentima
- III. $expr[4] = 5+5+4$, izraz sa dvostrukim prethodnim Katalanovim brojem i ostatkom
- IV. $expr[4] = 14$, Katalanov broj za datu osnovu $n=4$.

Primer za proveru drugog uslova: Proveravamo za dva izraza $expr[3]$ i $expr[4]$ da li je zadovoljen drugi uslov $expr[n]=2*expr[n-1] + rest$:

- I. $expr[3]=(2+0)+(2+1)$
- II. $expr[4]=[(2+0)+(2+1)] + [(2+0)+(2+1)] + (2+2)$
- III. $expr[4]=2*expr[3] + (2+2)$
- IV. $expr[4]=2*expr[3] + rest$, gde je $rest=4$.

3.3 Primena dekompozicije Katalanovih brojeva u generisanju ključeva

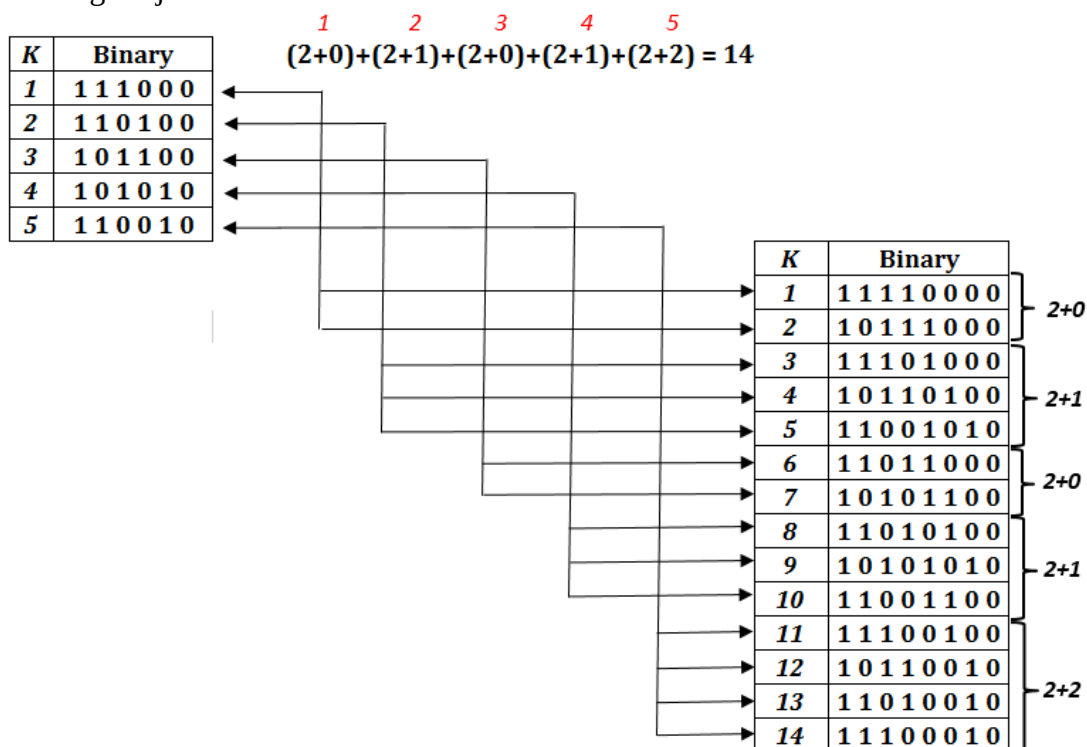
U ovoj sekciji, prikazaćemo neke primere generisanja validnih binarnih zapisa ključeva na osnovu opisanog postupka dekompozicije Katalanovih brojeva. Koristićemo postupak dinamičkog programiranja u generisanju svih ključeva za osnovu n , na osnovu postojeće baze ključeva koji pripadaju manjoj osnovi $n-1$.

Katalanovi brojevi imaju svojstvo rekurzivnosti i njihovo generisanje se može efikasno realizovati primenom metode dekompozicije. U radu [23] predstavljena je napredna tehnika za šifrovanje koja koristi karakteristike rekurzivnosti ključa. Svaka naredna vrednost ključa je duplo veća od prethodne. Što znači, da se posebno stavlja akcenat u dizajnu kriptosistema sa tehnikom generisanja rekurzivnog ključa. Ovaj metod smanjuje vreme izvršavanja za osnovu

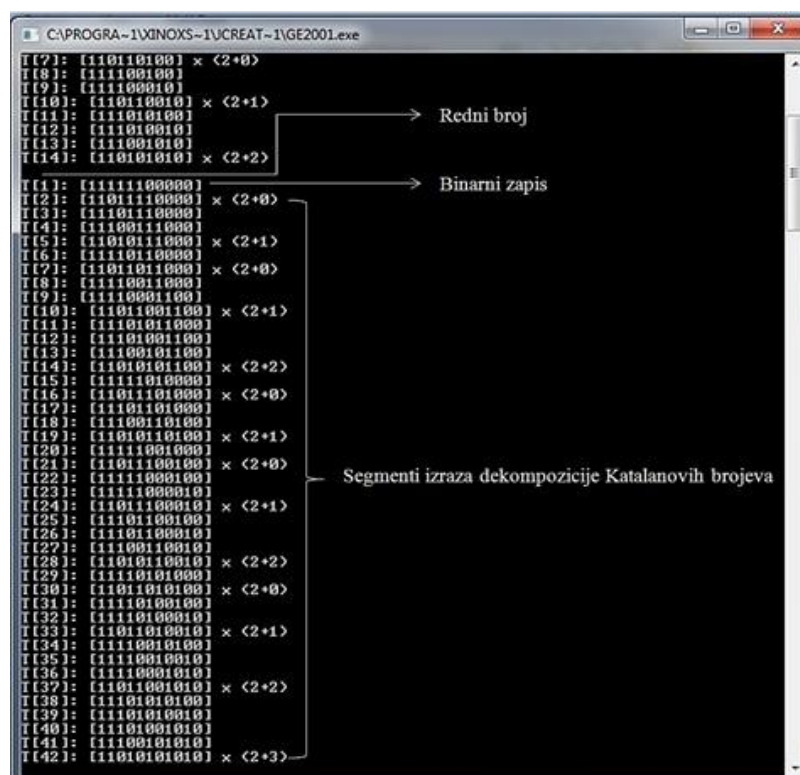
n u kojoj se zahteva traženje optimalne podstrukture $n-1$, ali prvenstveno ovo važi za one tehnike koje imaju potprobleme (korake) koji se ponavljaju.

Primer 3.5: Prikazaćemo način generisanja binarnih vrednosti Katalanovih brojeva za osnovu $n=4$ gde na osnovu formula (1) ili (3) imamo $C_4=14$, odnosno 14 vrednosti koje zadovoljavaju svojstvo Katalanovog broja za datu osnovu $n=4$. Uslov za dinamičko generisanje svih kombinacija ključeva je da imamo prethodni blok binarnih zapisa za $n-1=3$, a njihov broj je $C_3=5$ i njihovi binarni zapisi su: 101010, 101100, 110010, 110100 i 111000.

Drugi parametar je izraz dekompozicije $expr[4]=(2+0)+(2+1)+(2+0)+(2+1)+(2+2)$, za osnovu $n=4$, za koju tražimo generisanje svih binarnih zapisa koji zadovoljavaju svojstvo Katalanovog broja.



Slika 2. Postupak dinamičkog generisanja binarnih zapisa za osnovu n , pod uslovom da postoji $n-1$ i izraz dekompozicije $exp[n]$



Slika 3. Dinamičko generisanje ključeva
(sistemski izveštaj iz Java aplikacije za dekompoziciju Katalanovih brojeva)

Iz primera 3.5, kao i na osnovu izveštaja iz Java aplikacije, možemo uočiti da se blok binarnih zapisa iz C_3 dva puta u celosti preslikao u C_4 , uz adekvatne dopune.

$$C_3 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \quad C_4 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Način za generisanje dodatnih novih binarnih zapisa je detaljno opisan u našem radu [12]. Na osnovu dobijenih rezultata, možemo izvršiti klasifikaciju dobijenih binarnih zapisa u C_n . Možemo uočiti sledeću pravilnost. Dopuna prenetih blokova iz C_{n-1} u C_n realizuje se u vidu dve nove forme, tačnije možemo uočiti postojanje dva tipa novih binarnih zapisa:

$$10 [C_{n-1}] \text{ i } 1 [C_{n-1}] 0.$$

Tabele 3 i 4 prikazuju primere lančanog generisanja iz C_3 u C_4 a zatim iz C_4 u C_5 , po principu prenosa po dva bloka binarnih zapisa iz prethodnog nivoa u obliku dve nove forme $10 [C_3]$, $1 [C_3] 0$ a zatim $10 [C_4]$ i $1 [C_4] 0$.

Tabela 3. Generisanje binarnih zapisa u C_4 na osnovu bloka C_3

C3	C4					Rest
	10 [C3]		1 [C3] 0			
1 1 1 0 0 0	1 0	1 1 1 0 0 0	1	1 1 1 0 0 0	0	1 1 0 0 1 0 1 0
1 1 0 1 0 0	1 0	1 1 0 1 0 0	1	1 1 0 1 0 0	0	1 1 0 0 1 1 0 0
1 0 1 1 0 0	1 0	1 0 1 1 0 0	1	1 0 1 1 0 0	0	1 1 0 1 0 0 1 0
1 0 1 0 1 0	1 0	1 0 1 0 1 0	1	1 0 1 0 1 0	0	1 1 1 0 0 0 1 0
1 1 0 0 1 0	1 0	1 1 0 0 1 0	1	1 1 0 0 1 0	0	

Tabela 4. Generisanje binarnih zapisa u C_5 na osnovu bloka C_4

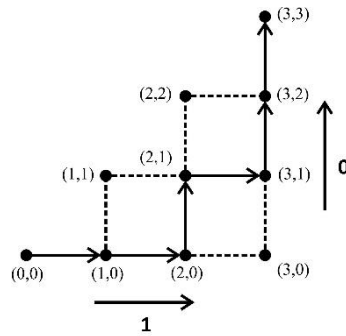
C4	C5		Rest
	10 [C4]	1 [C4] 0	
1 1 1 1 0 0 0 0	1 0 1 1 1 1 0 0 0 0	1 1 1 1 1 0 0 0 0 0	1 1 0 0 1 0 1 0 1 0
1 1 1 0 1 0 0 0	1 0 1 1 1 0 1 0 0 0	1 1 1 1 0 1 0 0 0 0	1 1 0 0 1 0 1 1 0 0
1 1 0 1 1 0 0 0	1 0 1 1 0 1 1 0 0 0	1 1 1 0 1 1 0 0 0 0	1 1 0 0 1 1 0 0 1 0
1 1 0 1 0 1 0 0	1 0 1 1 0 1 0 1 0 0	1 1 1 0 1 0 1 0 0 0	1 1 0 0 1 1 0 1 0 0
1 1 1 0 0 1 0 0	1 0 1 1 1 0 0 1 0 0	1 1 1 1 0 0 1 0 0 0	1 1 0 0 1 1 1 0 0 0
1 0 1 1 1 0 0 0	1 0 1 0 1 1 1 0 0 0	1 1 0 1 1 1 0 0 0 0	1 1 0 1 0 0 1 0 1 0
1 0 1 1 0 1 0 0	1 0 1 0 1 1 0 1 0 0	1 1 0 1 1 0 1 0 0 0	1 1 0 1 0 0 1 1 0 0
1 0 1 0 1 1 0 0	1 0 1 0 1 0 1 1 0 0	1 1 0 1 0 1 1 0 0 0	1 1 0 1 0 1 0 0 1 0
1 0 1 0 1 0 1 0	1 0 1 1 0 0 1 1 0 0	1 1 1 0 0 1 1 0 0 0	1 1 0 1 1 0 0 0 1 0
1 0 1 1 0 0 1 0	1 0 1 0 1 1 0 0 1 0	1 1 1 0 0 1 0 1 0 0	1 1 1 0 0 0 1 0 1 0
1 1 0 0 1 0 1 0	1 0 1 1 0 1 0 0 1 0	1 1 0 1 0 1 0 1 0 0	1 1 1 0 0 0 1 1 0 0
1 1 0 0 1 1 0 0	1 0 1 1 1 0 0 0 1 0	1 1 0 1 1 0 0 1 0 0	1 1 1 0 0 1 0 0 1 0
1 1 0 1 0 0 1 0	1 0 1 1 0 0 1 0 1 0	1 1 1 0 1 0 0 1 0 0	1 1 1 0 1 0 0 0 1 0
1 1 1 0 0 0 1 0	1 0 1 0 1 0 1 0 1 0	1 1 1 1 0 0 0 1 0 0	1 1 1 1 0 0 0 0 1 0

3.4 Primena Katalanovih brojeva i kombinatornih problema u šifrovanju fajlova

Katalanovi brojevi nalaze široku primenu u rešavanju mnogih kombinatornih problema. U izvorima [1,2] navedene su konkretne primene ovih brojeva sa mogućim rešenjima kada su u pitanju reprezentacije preko određenih kombinatornih problema. Svi navedeni kombinatorni problemi mogu se rešiti na osnovu vrednosti koje poseduju svojstva Katalanovih brojeva. Kratko rečeno, broj kombinacija i način generisanja Katalanovih brojeva predstavlja rešenje za određene kombinatorne probleme.

3.4.1 Metod *Lattice Path* – kretanje kroz celobrojnu mrežu

Binarnu notaciju Katalanovog broja možemo grafički predstaviti u obliku celobrojne mreže (drugi naziv diskretna rešetka ili *LatticePath*) koja se sastoji od određenog broja tačaka u Kartezijevom koordinatnom sistemu. Problem se odnosi na broj izračunavanja puteva kretanja kroz celobrojnu mrežu. Taj broj mogućih validnih puteva u mreži je direktno određen formulom za izračunavanje skupa Katalanovih brojeva C_n . Putevi se sastoje od $2n$ koraka sa nekom početnom tačkom $(0,0)$ i završnom tačkom (n,n) . Ukoliko želimo da binarni zapis Katalanovog broja predstavimo u obliku kretanja kroz celobrojnu mrežu, onda bit 1 može predstavljati pomeraj udesno, a onda bit 0 predstavlja pomeraj nagore.



Slika 4. Jedna kombinacija puta (na osnovu ključa $K3=110100$) u celobrojnoj mreži 3×3

Kao što je prikazano na slici, svaku putanju u celobrojnoj mreži možemo kodirati određenim redosledom vektora pomaka udesno $(1, 0)$ i vektora pomaka prema gore $(0, 1)$. Izbor pozicija pomaka udesno (oznaka 1, od ukupnog broja od $2n$ pomaka) jednoznačno određuje put u celobrojnoj mreži, zajedno sa preostalim pozicijama koje predstavljaju pomake na gore (oznaka 0). Znači, ako primenimo validan Katalanov broj, kretanje kroz celobrojnu mrežu će se obaviti tačno $2n$ pomaka, počev od početne tačke $(0,0)$ i završetkom do ciljne tačke (n,n) .

Proverićemo da li kretanje u celobrojnoj mreži zaista odgovara svojstvu Katalanovih brojeva, odnosno da li je broj validnih puteva određen brojem C_n . Na osnovu opšteg slučaja, uvedena je restrikcija na mrežu veličine $n \times n$ i utvrđeno je koliko ima najkraćih puteva u ovakvoj celobrojnoj mreži. Put nikad ne prelazi njenu dijagonalu. Glavni uslov je da se u svakom idućem koraku mora biti bliže ciljnoj tački. Pod uslovom da je $n \geq 0$ broj puteva je:

$$\binom{n}{\frac{n+m}{2}} - \binom{n}{\frac{n+m}{2}+1} = \frac{m+1}{\frac{n+m}{2}+1} \binom{n}{\frac{n+m}{2}} \quad (11)$$

Ako je vrednost $m=0$ dobićemo Katalanov broj [10]:

$$\frac{1}{\frac{n}{2}+1} \binom{n}{\frac{n}{2}} \quad (12)$$

Znači, broj mogućih puteva u mreži dimenzije n je određen Katalanovim brojem za osnovu n , a binarne vrednosti u skupu C_n određuju različite kombinacije puteva u mreži. Prikazani postupak kretanja u celobrojnoj mreži na osnovu *binarnog* zapisa ključa može poslužiti kao ideja za formiranje sistema šifrovanja otvorenog teksta.

Postupak šifrovanja na osnovu celobrojne mreže se može primeniti na tekst poruke (*String varijanta*), ali se može primeniti i na binarni zapis poruke (*ASCII Text to Binary*). U nastavku, zbog boljeg razumevanja, prikazaćemo postupak šifrovanja teksta gde se vrednosti uzimaju kao *string* zapis (dobija se šifra transpozicije), a u eksperimentalnom delu ovog rada biće dat i primer uzimanja otvorenog teksta u obliku binarnog zapisa (dobija se šifra supstitucije).

Na osnovu rasporeda bitova 0 i 1 u binarnom zapisu ključa, elementi mogu imati 2 stanja:

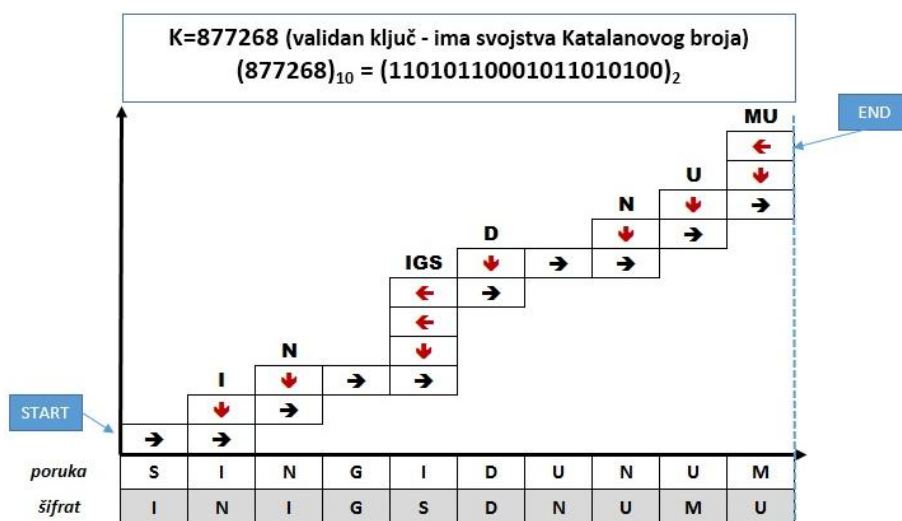
1. *Slobodan element* – to je karakter iz poruke koji nije šifrovan, tačnije nije prenet u šifrat. Slobodan element je uslovljen pojavom bita 1 (u ključu), koji čeka svoj par bit 0.
2. *Zauzet element* – to je karakter iz poruke koji je šifrovan i prenet u šifrat. To je element koji je uslovljen pojavom bita 0 (u ključu). Na taj način, element je “zatvoren” (prenet u šifrat, jer se pojavio bit 0, koji zatvara odgovarajući bit 1).

Kod koji realizuje postupak šifrovanja na osnovu kretanja kroz *celobrojnu mrežu* (početni parametri $n=0$, $A[n]=1$; segment = 60):

```
1. count = file.length() / segment;
2. for (j = 0; j < count; j++) {
3.   in.read(tekst, 0, segment);
4.   doCilja = segment - 1;
5.   for (i=n; i>=0; i--) {
6.     if (A[i] == 1) {
7.       putanja [slobodan] = tekst[doCilja];
8.       slobodan ++;
9.       doCilja --; }
10.    else {
11.      sifrat[zauzet] = putanja[slobodan-1];
12.      zauzet ++;
13.      slobodan --;
14.      putanja[slobodan] = 0; } }
15.   out.write(sifrat); }
```

Objašnjenje: (1) Deli poruku (fajl) na n -bitne segmente (odgovara osnovi ključa n); (2) Ciklus za uključivanje segmenata fajla počev od prvog pa sve do poslednjeg segmenta; (3) U svakom segmentu, vrši se čitanje elemenata, počev od prvog pa sve do poslednjeg elementa u segmentu; (4) Umanjuje se pozicija bita u segmentu; (5) Ciklus za čitanje bitova u ključu; (6) Ako je bit 1 onda sledi pomeraj udesno (povećava broj slobodnih karaktera, a smanjuje broj koraka do cilja); (7) Ako je bit 0 onda pomeraj nagore (uzima karakter, što znači da povećava broj zauzetih karaktera a smanjuje broj slobodnih); (8) Na kraju kada se završe svi segmenti poruke, ispisuje se dobijeni šifrat.

Primer 3.6: Ako izaberemo ključ $K=877268$, za koji smo utvrdili da poseduje svojstvo Katalanovog broja na osnovu binarnog zapisa $(877268)_{10} = (11010110001011010100)_2$ i neka je dat otvoren tekst $P=\text{“SINGIDUNUM”}$. Kretanjem kroz celobrojnu mrežu na osnovu binarnog zapisa ključa, počev od izvorišne do odredišne tačke dobijamo šifrat $C=\text{“INIGSDNUMU”}$.



Slika 5. Primer šifrovanja po principu kretanja kroz diskretnu rešetku

Karakter iz otvorenog teksta se uzimaju onog trenutka kada dobijemo uređen par 1 i 0. Sve dok odgovarajući bit 1 ne dobije svoj par bit 0, karakter ima status “slobodan“, tačnije nije prenet u šifrat. Kada dobije svoj par, karakter postaje “zauzet“ i prebacuje se u šifrat. Iz navedenog postupka, možemo utvrditi dva osnovna pravila kretanja (Slika 5):

1. Kretanje kroz celobrojnu mrežu nikad ne prelazi njenu dijagonalu,
2. U svakom idućem koraku, kretanje mora biti dalje izvorišnoj (START) a bliže odredišnoj tački (END).

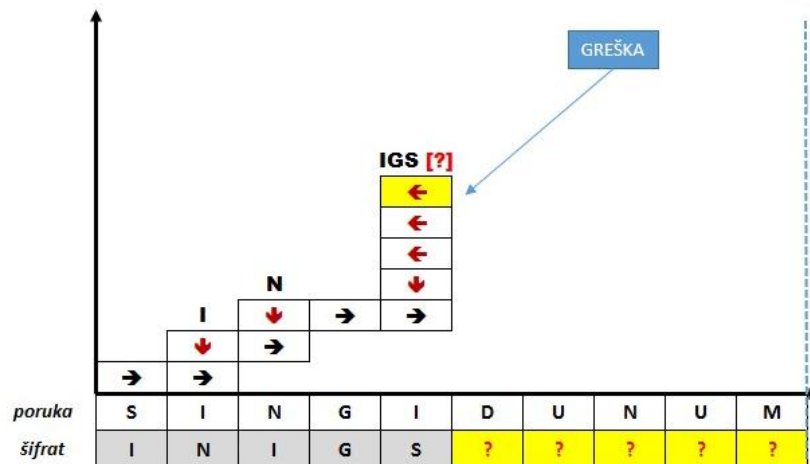
Tabela 5. Stanja karaktera u P=”SINGIDUNUM“ po principu kretanja kroz diskretnu rešetku

Bit u ključu	Poruka (pročitani element) Čitanje - pojava bita 1	Šifrat (prenet - zauzet element) Uzimanje - pojava bita 0	Parametri u brojaču
1	S		doCilja=19, slobodan=1, zauzet=0
1	S, I		doCilja=18, slobodan=2, zauzet=0
0	S,	I	doCilja=17, slobodan=1, zauzet=1
1	S,N,	I	doCilja=16, slobodan=2, zauzet=1
0	S,	I, N	doCilja=15, slobodan=1, zauzet=2
1	S, G	I, N	doCilja=14, slobodan=2, zauzet=2
1	S, G, I	I, N	doCilja=13, slobodan=3, zauzet=2
0	S, G	I, N, I	doCilja=12, slobodan=2, zauzet=3
0	S	I, N, I, G	doCilja=11, slobodan=1, zauzet=4
0		I, N, I, G, S	doCilja=10, slobodan=0, zauzet=5
1	D	I, N, I, G, S	doCilja=9, slobodan=1, zauzet=5
0		I, N, I, G, S, D	doCilja=8, slobodan=0, zauzet=6
1	U	I, N, I, G, S, D	doCilja=7, slobodan=1, zauzet=6
1	U, N	I, N, I, G, S, D	doCilja=6, slobodan=2, zauzet=6
0	U	I, N, I, G, S, D, N	doCilja=5, slobodan=1, zauzet=7
1	U,U	I, N, I, G, S, D, N	doCilja=4, slobodan=2, zauzet=7
0	U	I, N, I, G, S, D, N, U	doCilja=3, slobodan=1, zauzet=8
1	U, M	I, N, I, G, S, D, N, U	doCilja=2, slobodan=2, zauzet=8
0	U	I, N, I, G, S, D, N, U, M	doCilja=1, slobodan=1, zauzet=9
0		I, N, I, G, S, D, N, U, M, U	doCilja=0, slobodan=0, zauzet=10

Dešifrovanje se obavlja obrnutim redosledom čitanja binarnog zapisa ključa, počev od poslednjeg bita i završetkom sa prvim bitom u ključu. U ovom slučaju važi (1,0)→(0,1), tačnije pojava bita 0 označava otvoren par a 1 zatvoren par.

Primer 3.7: Sada ćemo uzeti neku vrednost (za ključ) koja nema svojstvo Katalanovog broja, tačnije ne važi pravilo bit-balansiranost. Pokušaćemo da taj ključ upotrebimo u postupku šifrovanja teksta na osnovu kretanja kroz celobrojnu mrežu.

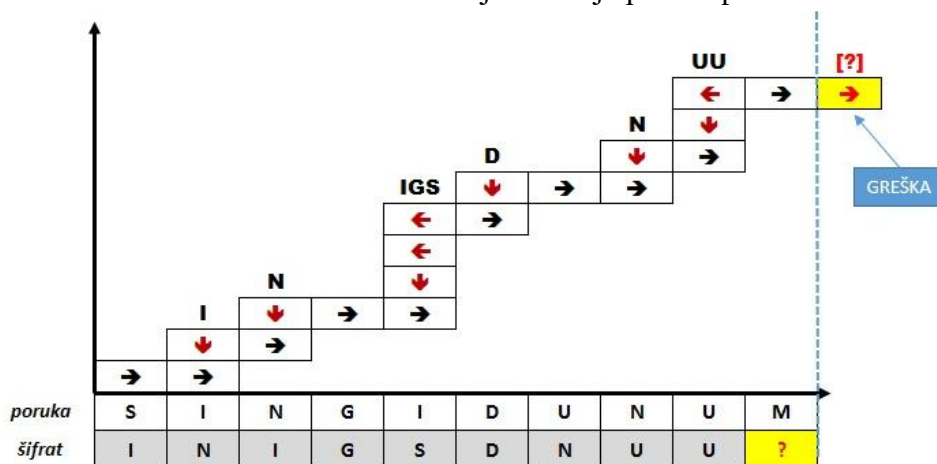
Slučaj 1: Izabrali smo ključ $K=877011$, za koji možemo utvrditi da ne poseduje svojstvo balansiranosti na osnovu binarnog zapisa $(877011)_{10} = (11010110000111010011)_2$. Neka je data poruka $P=SINGIDUNUM$. Kretanjem kroz celobrojnu mrežu, počev od izvorišne tačke ne možemo doći do odredišne tačke, tačnije ne možemo uspešno završiti proces šifrovanja.



Slika 6. Primer neuspešnog šifrovanja na osnovu ključa koji nije Katalanov broj (slučaj 1)

Iz ovog slučaja, možemo videti da je problem nastao u 11. koraku, tačnije problem je u jedanaestom bitu u ključu (bit 0), jer on nema svoj par $(877011)_{10} = (1101011000\boxed{0}111010011)_2$

Slučaj 2: Izabrali smo drugi ključ $K=877267$, za koji smo takođe utvrdili da ne poseduje svojstvo bit-balansiranosti $(877267)_{10} = (11010110001011010011)_2$. Neka je dat otvoren tekst $P=SINGIDUNUM$. Kretanjem kroz celobrojnu mrežu, počev od izvorišne ne možemo doći do odredišne tačke. U ovom slučaju kretanje prelazi prostor mreže.



Slika 7. Primer neuspešnog šifrovanja na osnovu ključa koji nije Katalanov broj (slučaj 2)

U ovom slučaju, problem je nastao u 20. koraku, tačnije u dvadesetom bitu (bit 1), jer on nema svoj par $(877267)_{10} = (1101011000101101001\boxed{1})_2$.

U oba slučaja, možemo konstatovati da nisu zadovoljeni uslovi balansiranosti bitova, koji su osnov za utvrđivanje svojstva Katalanovih brojeva. To osnovno pravilo je prouzrokovalo da se ne ispune dva osnovna uslova kretanja kroz celobrojnu mrežu:

1. U prvom slučaju, nije ispoštovan uslov da se u svakom idućem koraku mora biti bliže određenoj tački i da mora početi i završiti u tačkama koje određuju dijagonalu.
2. Sa druge strane, u drugom slučaju nije ispoštovana uvedena restrikcija na mrežu veličine $n \times n$ pa je kretanje van nje nemoguće jer je tačno utvrđeno koliko ima najkraćih puteva u celobrojnoj mreži koji nikad ne prelaze njenu dijagonalu, a to je broj C_n .

Zaključak je da ključ koji ne poseduje svojstva Katalanovog broja nije funkcionalan u kretanju kroz celobrojnu mrežu.

Bitno je napomenuti da se ovaj postupak šifrovanja može primeniti na tekstualni zapis poruke (transpozicija stringova) a može se izvršiti čitanje poruke u binarnom obliku, pa se onda vrši primena opisanog postupka na binarni zapis poruke.

Primer 3.8: Neka je dat otvoren tekst $P_{\text{txt}} = \text{"SINGIDUNUM"}$. Ako se primeni *ASCII Text to Binary* na P , onda se dobija sekvenca bitova $P_{\text{bin}} = 01010011\ 01001001\ 01001110\ 01000111\ 01001001\ 01000100\ 01010101\ 01001110\ 01010101\ 01001101$.

Primenom ključa $K = (930516)_{10} = (11100011001011010100)_2$ na osnovu kojeg ćemo izvršiti permutaciju bitova iz poruke, dobijamo sledeću sekvencu bitova tj. binarni zapis šifrata:

$C_{\text{bin}} = 0100100101101000010001100110010101100011010000100101001001000110011\ 0110101001011$. Primenom *Binary to ASCII Text* na šifrat C , dobija se tekstualni zapis šifrata $C_{\text{txt}} = \text{"IhFecBRFmK"}$.

Ako uporedimo otvoren tekst $P_{\text{txt}} = \text{"SINGIDUNUM"}$ i šifrat $C_{\text{txt}} = \text{"IhFecBRFmK"}$, možemo uočiti da je prvi karakter "I" zamenjen sa „h“, a drugi karakter "I" sa „c“. Isti slučaj je i sa karakterom „U“, gde je prvi zamenjen sa "R" a drugi sa „m“.

Na taj način obezbeđujemo jači mehanizam šifrovanja, odnosno jedan karakter je zamenjen nekim potpuno drugim znakom u zavisnosti od dobijene permutacije bitova. U ovom slučaju, nemamo klasičnu šifru transpozicije, kao u prethodnim primerima, već šifru supstitucije odnosno zamene. Bitno je još napomenuti da se ovde ne realizuje klasična supstitucija, jer se jedan karakter iz poruke ne zamenjuje uvek istim karakterom u šifratu. Način zamene zavisi od samog ključa i njegove dužine, kao i rasporeda bitova u ključu. Pored toga, postupak zavisi i od dužine poruke i veličine segmenata poruke koji se uzimaju u postupku šifrovanja.

3.4.2 Ballot notacija, Stack permutacije i Balanced Parentheses

U ovoj sekciji, razmotrićemo primenu nekih drugih kombinatornih problema u postupku šifrovanja. Primenićemo ključ koji poseduje svojstvo Katalanovog broja na probleme, kao što

su *Ballot* (problem glasanja), stek permutacije i *Balanced Parentheses* (balansirane ili uparene zagrade).

Opšti slučaj *Ballot* problema glasi [11]: “Koliko postoji kombinacija da se $2n$ glasova rasporede na takav način da se u svakom dodavanju novog glasa, broj glasova koji je dobio kandidat A bude veći ili jednak broju glasova koji je dobio kandidat B ”. Ovaj opšti slučaj se odnosi na pravilo $a=b=n$, gde je a broj glasova za kandidata A , a b broj glasova za B , i gde za svaku iteraciju važi $a_i \geq b_i$. Drugi slučaj se bazira na pravilu $a_i \neq b_i$, gde se ukupan broj načina glasanja može izraziti kao binomni koeficijent [11]:

$$\binom{a+b}{a}, \text{ za } a \neq b \quad (13)$$

Tvrđenje: Broj načina glasanja koji odgovara opštem slučaju *Ballot* problema jednak je Katalanovom broju.

Dokaz. Ako na (13) primenimo $a=b=n$ onda dobijamo binomni koeficijent

$$\binom{2a}{a} = \binom{2n}{n}$$

Pošto svaka kombinacija počinje prvim glasom za kandidata A (jer važi za svaku iteraciju pravilo: $a_i \geq b_i$, pa i za prvu), onda se izbacuju sve kombinacije zapisa koje počinju sa B . U zadnjoj iteraciji mora se zadovoljiti uslov $a_i = b_i$. Na osnovu ova dva ograničenja, broj kombinacija koji se izbacuje, a na osnovu Bernardovog dokaza [11], je

$$\binom{2n}{n+1} \quad (14)$$

Ako uklonimo one kombinacije koje nisu validne, onda broj kombinacija koji odgovara opštem slučaju *Ballot* problema, zaista odgovara Katalanovom broju:

$$C_n = \binom{2n}{n} - \binom{2n}{n+1} = \frac{1}{n+1} \binom{2n}{n} \quad (15)$$

Primer 3.9. Prikaz redosleda glasanja se može predstaviti tabelarno ili matricom za sve kombinacije. Na primer, za redosled glasanja $AABABB$, se formira sledeća tabela:

	i_1	i_2	i_3	i_4	i_5	i_6
Kandidat	A	A	B	A	B	B
A	1	2	2	3	3	3
B	0	0	1	1	2	3

Na osnovu tabele redosleda glasanja za oba kandidata, mogu se formirati matrice za kandidate A i B , gde se kao elementi matrice beleže redni brojevi glasova u trenutku dodavanja svakog novog glasa. Za slučaj $a=b=3$, gde važi $a_i \geq b_i$ za C_3 postoji 5 takvih kombinacija.

$$A = \begin{bmatrix} 1 & 2 & 3 & 3 & 3 & 3 \\ 1 & 2 & 2 & 3 & 3 & 3 \\ 1 & 1 & 2 & 3 & 3 & 3 \\ 1 & 1 & 2 & 2 & 3 & 3 \\ 1 & 2 & 2 & 2 & 3 & 3 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 1 & 2 & 3 \\ 0 & 1 & 1 & 1 & 2 & 3 \\ 0 & 1 & 1 & 2 & 2 & 3 \\ 0 & 0 & 1 & 2 & 2 & 3 \end{bmatrix}$$

Primer 3.10: U prikazanim matricama A i B, naveden je primer sa 6 glasača, od kojih 3 glasaju za kandidata A, a 3 za kandidata B. Ove matrice možemo predstaviti i u vidu jedne matrice sa *Ballot* zapisima. Ako poštujemo uslov da je prvi glas dobio kandidat A, i da se poštuje balans pojavljivanja drugog kandidata, gde važi da je uvek u datom trenutku glasanja $a_i \geq b_i$, onda imamo 5 takvih dozvoljenih (validnih) kombinacija:

A A A B B B
A A B A B B
A A B B A B
A B A A B B
A B A B A B

Kao što možemo primetiti, dobili smo 5 vrednosti ključeva koji poseduju svojstvo balansiranosti, samo što u ovom slučaju nisu predstavljeni u vidu binarnih zapisa već u obliku *Ballot* zapisa. Upravo, u radovima [27,28] se navodi činjenica da kombinatorni *Ballot* problem ili problem glasačkih listića nalazi veliku primenu u kriptografiji i načinu zapisivanja i šifrovanja.

U nastavku, navešćemo primer šifrovanja primenom *Ballot* kombinatornog problema na osnovu ključa koji ima svojstvo Katalanovog broja.

Primer 3.11: Neka je izabrani ključ $K=877268$, za koji smo već utvrdili da poseduje svojstvo Katalanovog broja i neka je dat otvoren tekst $P="SINGIDUNUM"$. Ako ključ predstavimo u obliku *Ballot* notacije, onda je bit 1 glas za prvog kandidata A i predstavlja operaciju čitanja karaktera iz poruke P, a bit 0 je glas za drugog kandidata B i predstavlja operaciju upisivanja u šifrat C. Dobijeni šifrat je $C="INIGSDNUMU"$.

PORUKA	S	I	N	G	I	D	U	N	U	M										
KLJUČ	1	1	0	1	0	1	1	0	0	0	1	0	1	1	0	1	0	1	0	0
faze glasanja	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
kandidat A	1	2	2	3	3	4	5	5	5	5	6	6	7	8	8	9	9	10	10	10
kandidat B	0	0	1	1	2	2	2	3	4	5	5	6	6	6	7	7	8	8	9	10
razlika (index)	1	2	2	3	3	4	5	5	/4/	/1/	6	6	7	8	8	9	9	10	10	/7/
ŠIFRAT			I		N			I	G	S		D			N		U		M	U

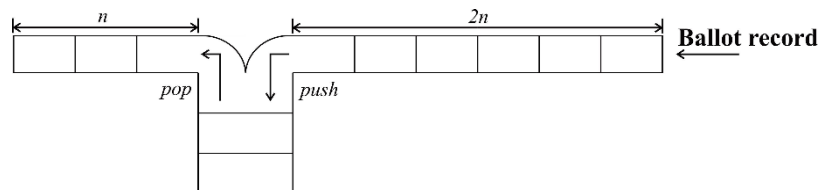
Slika 9. Primer šifrovanja po principu redosleda glasanja (*Ballot* problem)

Sada ćemo predstaviti mogućnost primene strukture steka u postupku šifrovanja. Stek je apstraktan tip i struktura podataka koja je zasnovana na principu *LIFO* (*last in, first out*), sa dve osnovne operacije: *push* i *pop*. Stek permutacije takođe mogu biti generisane na osnovu svojstva Katalanovih brojeva.

Napravićemo vezu između operacija nad stekom i znakova koji se pojavljuju u binarnom zapisu ključa koji poseduje svojstvo Katalanovog broja:

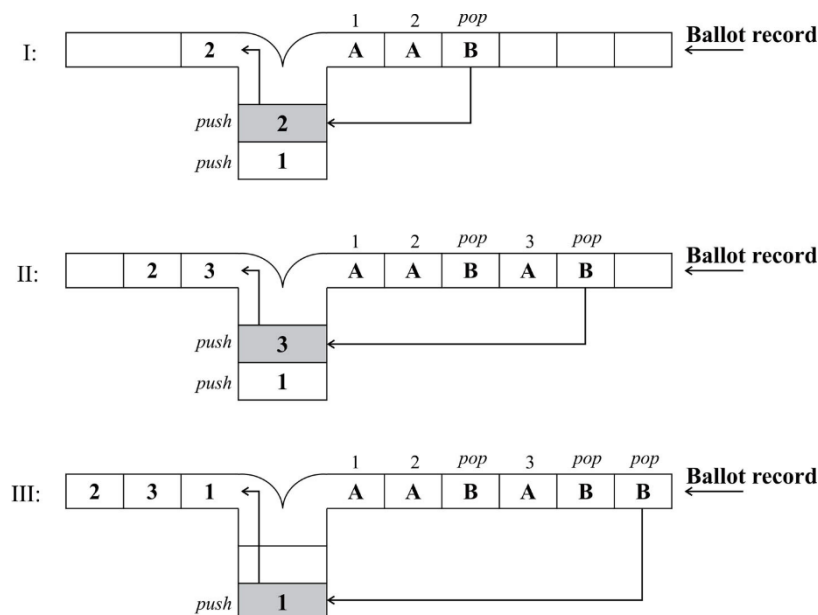
1. Ako je trenutni znak u zapisu 1, poziva se operacija *push* i stavlja se broj pojavljivanja ovog bita u zapisu sa leve strane.
2. Kada se pojavljuje vrednost 0, poziva se operacija *pop* i izbacuje se bit iz steka na izlazu.

U našem radu [13] prikazano je da broj permutacija koje zadovoljavaju navedene uslove odgovara Katalanovom broju. Prema tome, primenom steka može se jedinstveno preslikati svaki binarni zapis (ili ekvivalentni *Ballot* zapis) dužine $2n$ na permutaciju skupa $\{1, 2, \dots, n\}$. Na ulazu se očekuje binarni zapis, i pošto operacija *push* snima samo broj pojavljivanja vrednosti 1 dobija se dva puta kraći izlaz (na slici 10 je prikazan odnos ulaza i izlaza steka).



Slika 10. Način obrade ulaznih podataka preko stek strukture

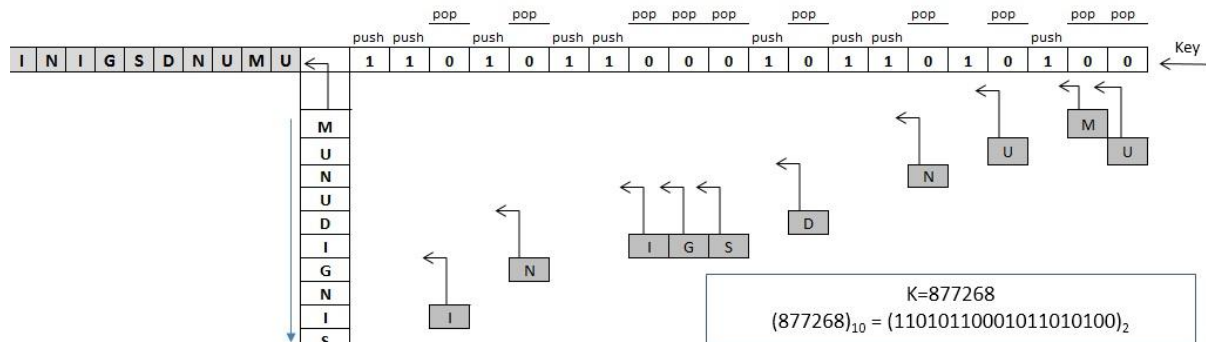
Primer 3.12: Neka *AABABB* bude ulazni *Ballot* zapis (to može biti i ekvivalentan binarni zapis *110100*). Prva dva znaka u ovom zapisu su *AA*, što znači da se stavlja broj 1 i broj 2 na stek. Naredni znak je *B*, tako da se izbacuje broj 2 i šalje na izlaz (Slika 11- I). Sledeći znak u zapisu je *A*, tako da se šalje broj 3, i nakon pojavljivanja sledećeg znaka *B* ova vrednost se izbacuje i šalje na izlaz. Sada, izlazni string sadrži brojeve: 2 i 3 (Slika 11- II). Konačno, posle poslednjeg znaka u zapisu (znak *B*), vrednost 1 se izbacuje i šalje na izlaz steka (Slika 11- III).



Slika 11. Primer obrade *ballot* zapisa pomoću steka

Isti postupak važi i za binarni zapis ili *Balanced Parentheses* notaciju.

Primer 3.13: Neka je dat ključ $K=877268$, za koji smo već utvrdili da poseduje svojstvo Katalanovog broja i neka je dat otvoren tekst $P="SINGIDUNUM"$. Bit 1 u binarnom zapisu ključa predstavlja *push* operaciju odnosno čitanje karaktera iz poruke P , a bit 0 predstavlja *pop* operaciju odnosno upisivanje u šifrat C . Dobijeni šifrat je $C="INIGSDNUMU"$.



Slika 12. Primer šifrovanja po principu stek permutacija

Za postupak dešifrovanja, potrebno je omogućiti da se na osnovu izlaznih vrednosti dobijenih pomoću steka dobije odgovarajući binarni ili *Ballot* zapis (obrnut proces). Izlazne vrednosti dobijene iz steka su određene pomoću sekvence operacija nad stekom. U našem radu [13] te dobijene vrednosti iz stek strukture smo nazvali SVB (*stack value based binary/ballot notation*). U nastavku, predložen je algoritam za mapiranje SVB zapisa u *Ballot* ili binarni zapis.

Algoritam 2. Mapiranje SVB u *Ballot* zapis [13]

Ulaz: Niz P_i , $i = 1, \dots, n$ sadrži permutaciju skupa $\{1, \dots, n\}$ koristeći stek.

1: for $i = 1$ to n do

1.1: Ako je $P_i > 0$ pošalji P_i znakova A na izlaz.

1.2: Pošalji jedan znak B na izlaz.

1.3: Ako je $P_i > 0$ za $j = i + 1$ sve do n umanji P_j za P_i .

Izlaz: Odgovarajući ballot zapis.

Jedan sličan primer šifrovanja, koji se bazira na sekvencama operacija nad stekom, dat je u studiji slučaja [33]. Ekvivalent stek permutacijama i *Ballot* notaciji može biti i *Balanced Parentheses* notacija, odnosno zapis u obliku balansiranih zagrada [8]. U našem radu [14] se detaljno razmatraju mogućnosti reprezentacije Katalanovih brojeva na osnovu kombinatornog problema *Balanced Parentheses*. U tom radu, naveli smo metod za alfanumeričku notaciju uparenih zagrada koji se upravo bazira na svojstvima Katalanovih brojeva.

Prikazaćemo postupak šifrovanja teksta na osnovu kombinatornog problema *Balanced Parentheses*, koje se vrši na sličan način kao kod stek permutacija ili *Ballot* problema.

Primer 3.14: Neka je dat ključ $K=877268$, za koji smo već utvrdili da poseduje svojstvo Katalanovog broja i neka je dat otvoren tekst $P="SINGIDUNUM"$. Ako ključ predstavimo u obliku uparenih zagrada, onda je bit 1 otvorena zagrada i predstavlja operaciju čitanja karaktera iz poruke, a bit 0 je zatvorena zagrada i predstavlja operaciju upisivanja u šifrat. Poslednja otvorena zagrada mora biti prva zatvorena (LIFO). Dobijeni šifrat je $C="INIGSDNUMU"$.

KLJUČ	1	1	0	1	0	1	1	0	0	0	1	0	1	1	0	1	0	1	0	0
Balanced parentheses	(	(	)	(	)	(	(	)	)	)	(	)	(	(	)	(	)	(	)	)
PORUKA	S	I		N		G	I				D		U	N		U		M		
ŠIFRAT			I		N			I	G	S		D			N		U		M	U

Slika 13. Primer šifrovanja po kombinatornom problemu u obliku balansiranih zagrada

3.5 Primer primene dekompozicije Katalanovih brojeva u steganografiji

Skrivanje podataka iz bezbednosnih razloga danas je uobičajena pojava u svim organizacijama čije se poslovanje bazira na zaštićenom prenosu podataka. Sakriti informaciju znači ne dozvoliti da informacija bude čitljiva trećim licima. Za rešenje problema, pored primene određenih kriptografskih algoritama moguća je i upotreba steganografskih metoda koje omogućavaju prikrivanje podataka u slikama, tekstu, audio i video snimcima, ali na taj način da originalni nosilac podatka ostane verodostojan.

Steganografija je nauka koja se bavi sakrivanjem informacija u drugim podacima, tako da je i samo postojanje šifrata sakriveno u nosiocu podatka. Cilj svake krypto-metode je da pronađe najbrži i najpovoljniji način da prenesena informacija bude sačuvana.

U radu [18], predložene su nove tehnike sakrivanja podataka pomoću kombinacije sekvenci *Katalanovih-Lukas brojeva*, što predstavlja poboljšanje u odnosu na postojeće tehnike skrivanja podataka. Iz mnogih drugih radova, možemo videti da se koriste kombinacije *Lukas-Katalanovi-Fibonačijevi brojevi* koji su superiorniji u odnosu na postojeću Fibonačijevu tehniku skrivanja podataka. Konkretno, u radovima [18,19] koriste se dva generatora slučajnih brojeva. Jedan generiše Katalanove a drugi Lukas sekvence. U predloženoj tehnici, jedan niz predstavlja skup konstanti a drugi niz promenljivih koje su uslovljene datim konstantama.

U ovom delu diplomskog rada, predstavimo samo jedan primer primene dekompozicije Katalanovih brojeva u steganografiji. Tačnije, koristićemo sekvence u vidu kombinacije nizova *Katalanovih brojeva – dekompozicije Katalanovih brojeva*. Nama će formula za Katalanove brojeve poslužiti kao generator za prvu sekvencu brojeva (konstante) a metoda dekompozicije će nam poslužiti kao generator za drugi niz (promenljive). Ova dva niza su međusobno povezana, odnosno drugi je uslovljen prvim nizom. Metoda dekompozicije generiše promenljive na osnovu konstantnih Katalanovih brojeva (primer: 1,2,5,14,42 itd.). Na taj način izbegavamo primenu drugih brojeva (poput Fibonačijevih ili Lukas brojeva).

Rukovodeći se *Katalan-Lukas* tehnikom, mi ćemo uzeti jednu sekvencu iz niza Katalanovih brojeva i na osnovu dekompozicije tih brojeva generisati još jednu sekvencu brojeva koja će biti zavisna u odnosu na prvu sekvencu. Na taj način ćemo imati kombinaciju sekvenci koju možemo nazvati *Katalan-DekompozicijaKatalan*.

Primer 3.15: Uzećemo opadajuću sekvencu prvih pet Katalanovih brojeva $C(5)=\{42,14,5,2,1\}$. Na osnovu ove sekvence, možemo dobiti nizove koji zadovoljavaju uslove dekompozicije navedenih Katalanovih brojeva. Postavlja se pitanje, koliko se različitih nizova može dobiti? Broj takvih nizova zavisi direktno od broja različitih segmenata $(2+i)$ u odabranom izrazu dekompozicije tj. generatoru novih nizova.

Ako uzmemo da je generator novih nizova $C_4 = 14$, koji se može dekomponovati u obliku $(2+0)+(2+1)+(2+0)$ $(2+1)+(2+2)$, onda možemo videti da u ovom izrazu postoje tri različita segmenta oblika $(2+i)$, a to su: $(2+0)$, $(2+1)$, $(2+2)$. Znači da imamo ukupno $2^3=8$ različitih kombinacija novih nizova.

Uočićemo još jednu pravilnost koja se javlja u svim novim kombinacijama generisanja nizova brojeva na osnovu dekompozicije. Ukoliko primenimo pravilo dekompozicije gde važi da se svaki naredni Katalanov broj generiše tako što u sebi sadrži dva puta prethodni broj i još neki ostatak, onda suma svih elemenata dobijenog novog niza je naredni Katalanov broj.

U nastavku, proverićemo da li zaista važi navedeno pravilo za, u ovom slučaju, svih 8 kombinacija:

- **Slučaj 1.** Shodno pravilima dekompozicije Katalanovih brojeva, izvršićemo množenje niza Katalanovih brojeva $C(5)=\{42,14,5,2,1\}$ i izraza $C_{decomp}(4)=\{(2+0),(2+1),(2+0),(2+1),(2+2)\}$. Dobijamo $(42*2+0),(14*2+1),(5*2+0),(2*2+1),(1*2+2)$, tj. niz brojeva $D1=\{84,29,10,5,4\}$. Dobijen je niz D1, gde možemo primetiti da suma njegovih elemenata iznosi 132, odnosno to je sledeći Katalanov broj. Rukovodeći se *Katalan-Lukas* tehnikom uradićemo uniju skupova $C(5)$ i $D1$ i dobiti $X1=\{1,2,4,5,10,14,29,42,84\}$. Ovaj niz možemo koristiti u postupku skrivanja informacija. Kasnije ćemo pokazati kako. Sada ćemo proveriti, da li zaista izraz dekompozicije omogućava generisanje niza koji zadovoljava svojstva Katalanovog broja, odnosno da je suma dobijenog niza uvek jednaka sledećem Katalanovom broju, a to je u ovom slučaju 132.
- **Slučaj 2.** Ako uzmemo drugu kombinaciju izraza $C_{decomp}(4)=\{(2+0),(2+2),(2+0),(2+1),(2+1)\}$ postupkom množenja niza Katalanovih brojeva i datog izraza dekompozicije dobijamo $(42*2+0), (14*2+2), (5*2+0), (2*2+1), (1*2+1)$. Dobijeni niz je $D2=\{84, 30, 10, 5, 3\}$. Možemo primetiti da suma i u skupu D2 iznosi 132, odnosno sledeći Katalanov broj. Rukovodeći se *Katalan-Lukas* tehnikom uradićemo uniju ova dva skupa i dobiti $X2=\{1,2,3,5,10,14,30,42,84\}$. I ovaj niz možemo koristiti u postupku skrivanja informacija.

Možemo uraditi za sve ostale kombinacije izraza dekompozicije. Opet ćemo dobiti niz koji zadovoljava svojstva Katalanovog broja i dekompozicije:

- **Slučaj 3.** $C_{decomp}(4)= (2+0), (2+2), (2+1), (2+1), (2+0)$. Dobijeni niz je $D3=\{84, 30, 11, 5, 2\}$, suma elemenata je 132. Unijom skupova dobijamo $X3=\{1,2,5,11,14,30,42,84\}$
- **Slučaj 4.** $C_{decomp}(4)= (2+1), (2+0), (2+0), (2+1), (2+2)$. Dobijeni niz je $D4=\{85, 28, 10, 5, 4\}$, suma elemenata je 132. Unijom skupova dobijamo $X4=\{1,2,4,5,10,14,28,42,85\}$
- **Slučaj 5.** $C_{decomp}(4)= (2+2), (2+1), (2+0), (2+1), (2+0)$. Dobijeni niz je $D5=\{86, 29, 10, 5, 2\}$, suma elemenata 132. Unijom skupova dobijamo $X5=\{1,2,5,10,14,29,42,86\}$
- **Slučaj 6.** $C_{decomp}(4)= (2+0), (2+2), (2+1), (2+0), (2+1)$. Dobijeni niz je $D6=\{84, 30, 11, 4, 3\}$, suma elemenata 132. Unijom skupova dobijamo $X6=\{1,2,3,4,5,11,14,30,42,84\}$
- **Slučaj 7.** $C_{decomp}(4)= (2+1), (2+2), (2+1), (2+0), (2+0)$. Dobijeni niz je $D7=\{85, 30, 11, 4, 2\}$, suma elemenata 132. Unijom skupova dobijamo $X7=\{1,2,4,5,11,14,30,42,85\}$
- **Slučaj 8.** $C_{decomp}(4)= (2+1), (2+0), (2+2), (2+0), (2+1)$. Dobijeni niz je $D8=\{85, 28, 12, 4, 3\}$, njihova suma elemenata 132. Unijom skupova dobijamo $X8=\{1,2,3,4,5,12,14,28,42,85\}$

Primer 3.16: Neka je dat otvoren tekst $P_{\text{txt}} = \text{"SINGIDUNUM BG"}$. Ako se primeni *ASCII Text to Binary* dobija se niz bitova odnosno binarni zapis: 01010011 01001001 01001110 01000111 01001001 01000100 01010101 01001110 01010101 01001101. Primenom predstavljene tehnike *Katalan-DekompozicijaKatalan* možemo koristiti nizove (X1-X8) za skrivanje neke informacije unutar otvorenog teksta P. Na primer, možemo uzeti nizove X3,X4,X5 i na osnovu njihovih vrednosti odrediti skrivene pozicije bitova u binarnom zapisu poruke *SINGIDUNUM BG*.

Pozicije Katalanovih brojeva su fiksne (statične), a pozicije koje su dobijene na osnovu izraza dekompozicije su promenljive, i njihov položaj je direktno određen postojanjem fiksnih pozicija tj. Katalanovih brojeva.

Kao što možemo primetiti (u nastavu primera), pozicije 1,2,5,14,42 su statične u X3,X4,X5 (siva boja), a ostale vrednosti su promenljive (plava boja). Na osnovu datog primera, u zapisu poruke **SINGIDUNUM BG** krije se tajna poruka **FiR**.

Prvi niz koji određuje prvi tajni karakter:

$X3 = \{1,2,5,11,14,30,42,84\}$

pozicija (1-24)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
bit	0	1	0	1	0	0	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	1	1	0
pozicija (25-48)	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
bit	0	1	0	0	0	1	1	1	0	1	0	0	1	0	0	1	0	1	0	0	0	1	0	0
pozicija (49-72)	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72
bit	0	1	0	1	0	1	0	1	0	1	0	0	1	1	1	0	0	1	0	1	0	1	0	1
pozicija (73-96)	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
bit	0	1	0	0	1	1	0	1	0	1	0	0	0	0	1	0	0	1	0	0	0	1	1	1

Izlaz (1): **01000110**, Binary to ASCII Text Converter →TAJNI KARAKTER F

Drugi niz koji određuje drugi tajni karakter:

$X4 = \{1,2,4,5,10,14,28,42,85\}$

pozicija (1-24)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
bit	0	1	0	1	0	0	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	1	1	0
pozicija (25-48)	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
bit	0	1	0	0	0	1	1	1	0	1	0	0	1	0	0	1	0	1	0	0	0	1	0	0
pozicija (49-72)	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72
bit	0	1	0	1	0	1	0	1	0	1	0	0	1	1	1	0	0	1	0	1	0	1	0	1
pozicija (73-96)	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
bit	0	1	0	0	1	1	0	1	0	1	0	0	0	0	1	0	0	1	0	0	0	1	1	1

Izlaz (2): **011010010**, Binary to ASCII Text Converter →TAJNI KARAKTER i

Treći niz koji određuje treći tajni karakter:

$X5 = \{1,2,5,10,14,29,42,86\}$

pozicija (1-24)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
bit	0	1	0	1	0	0	1	1	0	1	0	0	1	0	0	1	0	1	0	0	1	1	1	0
pozicija (25-48)	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
bit	0	1	0	0	0	1	1	1	0	1	0	0	1	0	0	1	0	1	0	0	0	1	0	0
pozicija (49-72)	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72
bit	0	1	0	1	0	1	0	1	0	1	0	0	1	1	1	0	0	1	0	1	0	1	0	1
pozicija (73-96)	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
bit	0	1	0	0	1	1	0	1	0	1	0	0	0	0	1	0	0	1	0	0	0	1	1	1

Izlaz (3): **01010010**, Binary to ASCII Text Converter →TAJNI KARAKTER **R**

REZULTAT: (01000110, 011010010, 01010010) **TAJNA PORUKA = FiR**

4 EKSPERIMENTALNI RAD SA STUDIJOM SLUČAJA

U ovom poglavlju, dat je eksperimentalni rad sa studijom slučaja, koji obuhvata konkretne algoritme za šifrovanje i dešifrovanje, koji su opisani u teorijskom delu rada. U prvoj sekciji eksperimentalnog dela, opisaćemo strukturu izvornog koda i način rada kreirane Java GUI aplikacije. Imajući u vidu da koristimo Katalanove brojeve za generisanje ključeva, u drugoj sekciji ovog eksperimentalnog dela, primenićemo NIST statističku bateriju testova radi procene kvaliteta generisanih binarnih sekvenci ključeva.

Java ima mnoge funkcionalnosti koje podržavaju kriptografske algoritme. Kriptografske operacije su klasifikovane u klase na osnovu kojih Java obezbeđuje kriptografske funkcionalnosti kroz dva API-ja:

1. *Java Cryptography Architecture (JCA)* - sigurnosni *framework* integrisan u jezgro (core) Java API. Klase u JCA obezbeđuju rad sa digitalnim potpisima, generisanje ključeva, hešovanje poruka (*MessageDigest*, *Signature*, *KeyPairGenerator*, *KeyStore*, *KeyFactory*, *SecureRandom*, *AlgorithmParameters*, *AlgorithmParameterGenerator*). JCA klase su locirane u paketu *java.security*.
2. *Java Cryptography Extension (JCE)* - ekstenzije za jako šifrovanje. Klase u JCE su locirane u paketu *javax.crypto*. JCE klase obezbeđuju rad za šifrovanje i dešifrovanje: *Cipher* (*encryption* / *decryption*), *SecretKeyFactory*, *KeyGenerator*, *KeyAgreement*.

Osnovne karakteristike navedenih klasa u Javi su:

- A) *Nezavisnost implementacije* – korišćenje kriptografskih provajdera i servisa,
- B) *Nezavisnost algoritama* – specifikacija klasa,
- C) *Proširivost algoritama* – lako ažuriranje klasa sa novim algoritmima,
- D) *Ineteroperabilnost implementacije* – uzajamna saradnja provajdera.

Udžbenik [29] sadrži sva neophodna praktična znanja za pravilnu implementaciju kriptografskih mehanizama u programskom jeziku Java. Pored toga, izvori [30,31,32] predstavljaju teorijski i praktični fundament koji je neophodan za razumevanje i praktično savladavanje materije iz oblasti kriptologije i teorije informacija i kodovanja.

4.1 Postupak za generisanje ključeva, šifrovanje i dešifrovanje

Postupak u okviru ove studije slučaja obuhvata tri faze. Prva faza obuhvata pronalaženje Katalanovih brojeva (ključeva) na osnovu date osnove n . Ova faza obuhvata sledeće korake:

1. Na ulazu se zadaje osnova n ,
2. Izračunava skup C_n (prostor ključeva),
3. Bira se jedan Katalanov broj (ključ) iz skupa C_n ,
4. Izabrani ključ se konvertuje iz decimalnog u binarni zapis.

Druga faza je proces šifrovanja na osnovu ključa koji poseduje svojstvo Katalanovog broja. Ova faza obuhvata sledeće korake:

1. Učitava se poruka (otvoren tekst),

2. Poruka se konvertuje (*ASCII Text to Binary*) u binarnu sekvencu,
3. Binarna sekvenca se deli na X segmenata čija dužina odgovara osnovi n ,
4. Primenom binarnog zapisa ključa i njegovim čitanjem, počev od prvog bita i završetkom sa poslednjim bitom u ključu (pojava bita 1 označava otvoren par a 0 zatvoren par), vrši se permutacija bitova sekvenci $X_1, \dots, X_n$
5. Permutacija bitova vrši se po opisanom principu kretanja kroz diskretnu rešetku odnosno *LatticePath*,
6. Dobijene permutacije bitova se konvertuju (*Binary to ASCII Text*), i na taj način se dobija šifrat.

Treća faza je proces dešifrovanja na osnovu ključa koji poseduje svojstvo Katalanovog broja. Ova faza obuhvata sledeće korake:

1. Učitava se šifrat (tekstualni zapis),
2. Šifrat se konvertuje (*ASCII Text to Binary*) u binarnu sekvencu,
3. Binarna sekvenca se deli na X segmenata čija dužina odgovara osnovi n ,
4. Primenom binarnog zapisa ključa i njegovim čitanjem u obrnutom redosledom, počev od poslednjeg bita i završetkom sa prvim bitom u ključu (pojava bita 0 označava otvoren par a 1 zatvoren par), vrši se permutacija bitova sekvenci $X_1, \dots, X_n$
5. Permutacija bitova vrši se po opisanom principu kretanja kroz diskretnu rešetku odnosno *LatticePath*, u obrnutom redosledu,
6. Dobijene permutacije bitova se konvertuju (*Binary to ASCII Text*), i na taj način se dobija izvorna poruka.

4.2 Struktura Java izvornog koda i GUI elementi aplikacije

Pri realizaciji Java softverskog rešenja, bitno je napomenuti da nismo koristili gotove Java klase iz pomenuta dva API-ja (*JCA*, *JCE*). Korišćene su klase za rad sa fajlovima (*File*, *FileWriter*, *FileReader*, *BigInteger*, *Vector*, *logging.Logger*...). Pored toga, za programiranje GUI aplikacije koristili smo *Swing* i *AWT* paket. Projekat se sastoji od tri klase: *CatalanCrypto*, *CatalanNumbers* i *CatalanDecomposition*.

I) U okviru klase ***CatalanCrypto*** date su sledeće metode:

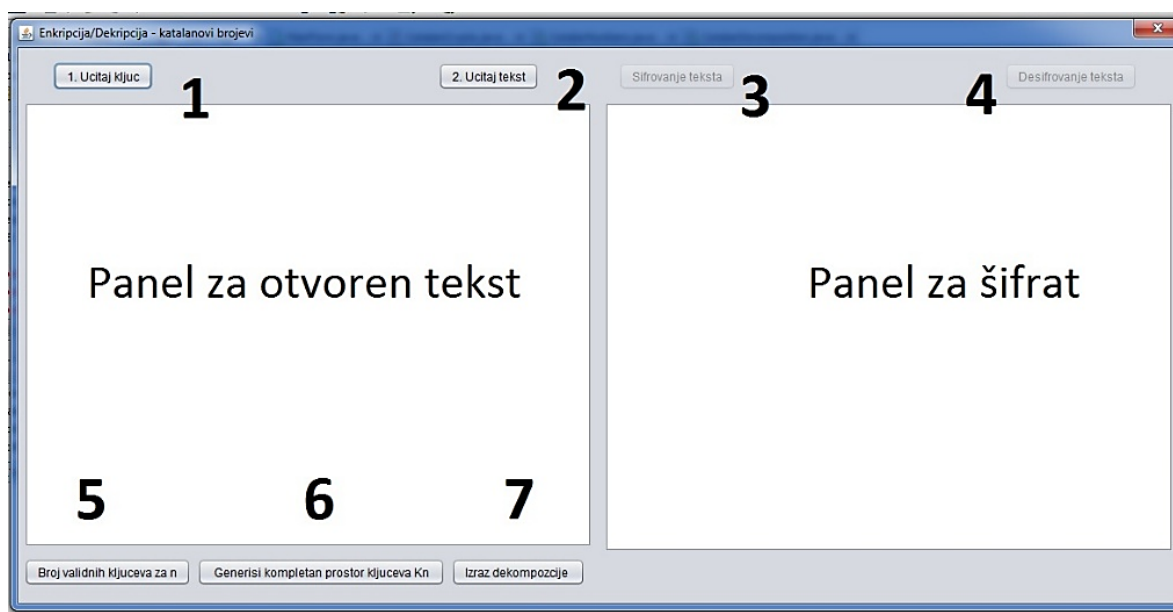
1. `File sifrovanjeCatalanPath(File file, String key)`, izvorni kod u prilogu 7.1.
2. `File desifrovanjeCatalanPath(File file, String key)`, izvorni kod u prilogu 7.1.
3. `public String ucitajKljuc(File file)`

II) Druga klasa u našem Java projektu je ***CatalanNumbers*** sa metodom za pronalaženje Katalanovog broja `find_catalan(File file)`. Izvorni kod za ovu klasu dat je u prilogu 7.2.

III) Treća klasa ***CatalanDecomposition*** sadrži metodu za dinamičko generisanje binarnih zapisa ključeva na osnovu izraza dekompozicije Katalanovog broja. Izvorni kod za metodu `createExprDecomposition` dat je u prilogu 7.3.

Glavna forma *Java GUI aplikacije* poseduje sledeće funkcionalnosti:

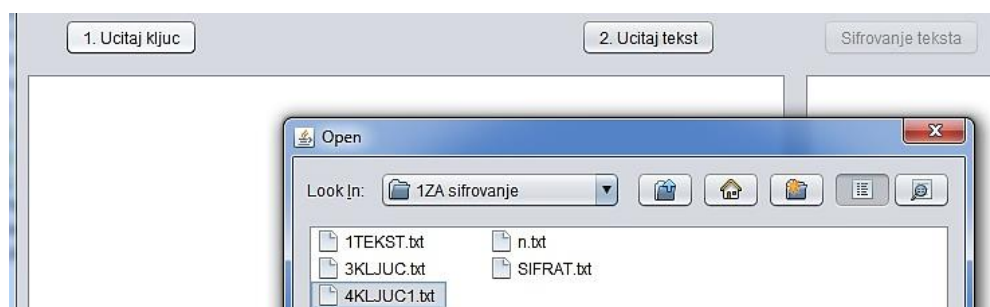
1. Učitavanje ključa,
2. Učitavanje fajla,
3. Šifrovanje otvorenog teksta,
4. Dešifrovanje šifrata,
5. Provera broja validnih ključeva za datu osnovu n (prostor ključeva),
6. Generisanje ključeva koji poseduju svojstva Katalanovog broja,
7. Dinamičko generisanje binarnih zapisa ključeva na osnovu izraza dekompozicije Katalanovog broja.



Slika 14. Osnovni elementi glavne forme *Java aplikacije*

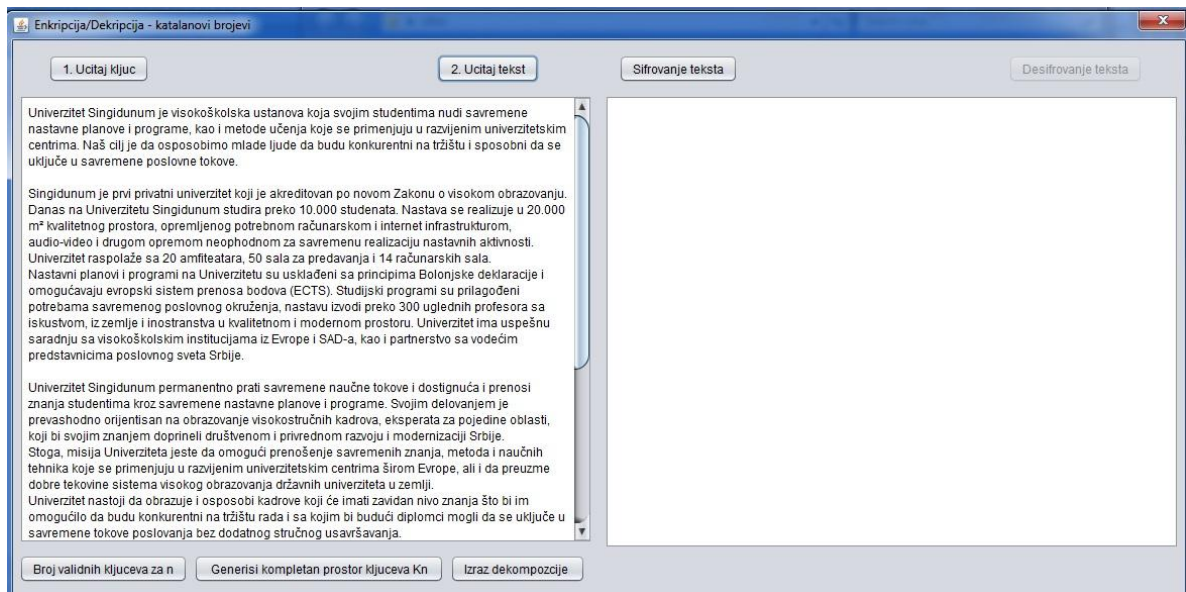
Kada pokrenemo *Java aplikaciju*, prvi korak je učitavanje ključa iz eksterne datoteke. U aplikaciji postoji algoritam za generisanje kompletnog prostora ključeva za određenu osnovu n . Na primer, za osnovu $n=30$ imamo ukupno 3,814,986,502,092,304 generisanih vrednosti (60-bitnih ključeva) koje zadovoljavaju svojstvo Katalanovog broja i neka od tih vrednosti može biti upotrebljena u postupku šifrovanja. Koristimo metod ručnog uzimanja jednog od tih vrednosti i smeštamo u datoteku *KLJUC.TXT*. Nakon toga, uključimo navedenu datoteku kao trenutni aktivni ključ.

Na ovaj način možemo kreirati više fajlova sa ključevima, ali u jednom postupku moramo odrediti koji je ključ aktivan za trenutno šifrovanje odnosno dešifrovanje (slika 15).



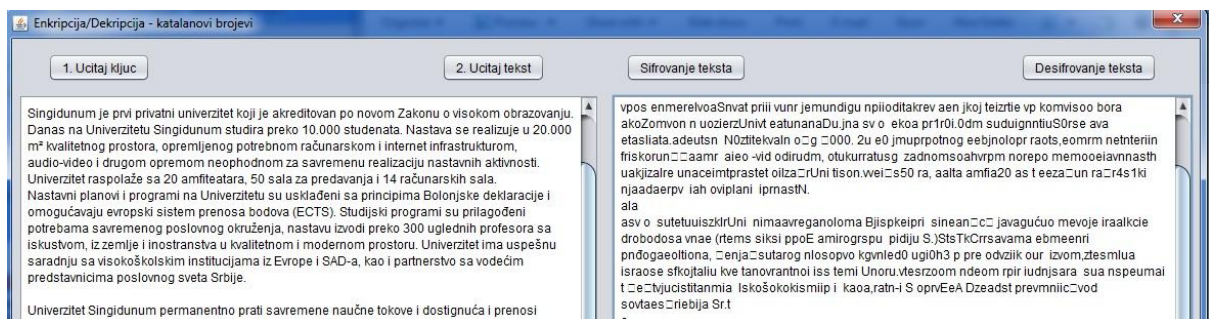
Slika 15. Učitavanje ključa

Nakon učitavanja ključa, sledi učitavanje datoteke sa otvorenim tekstom. Može biti učitana datoteka sa ekstenzijom .txt, .doc (slika 16).



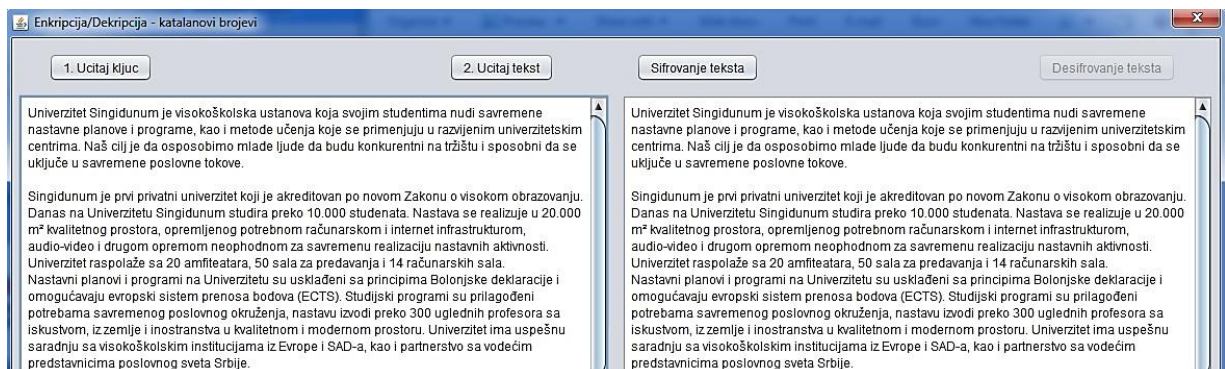
Slika 16. Učitavanje poruke (otvorenog teksta) i prikazivanje u levom panelu

Nakon uspešnog učitavanja ključa i poruke, dugme “šifrovanje teksta” postaje aktivno. Pre toga, ova opcija je bila onemogućena. Nakon dobijanja šifrata, dugme “dešifrovanje teksta” postaje aktivno (slika 17). Pre toga ovo dugme je bilo onemogućeno.



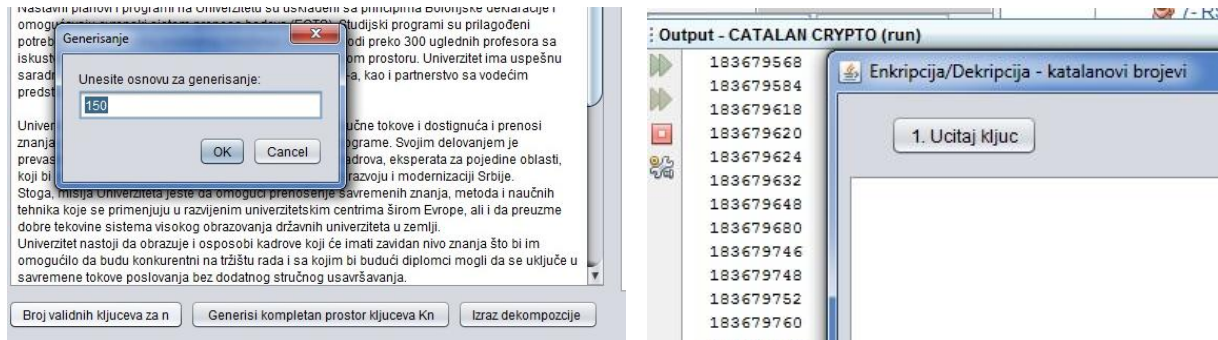
Slika 17. Šifrovanje otvorenog teksta i prikazivanje šifrata u desnom panelu

Klikom na dugme “dešifrovanje teksta” možemo dešifrovati šifrat (slika 18 – desno) i uporediti ga sa originalnom porukom (slika 18 - levo).



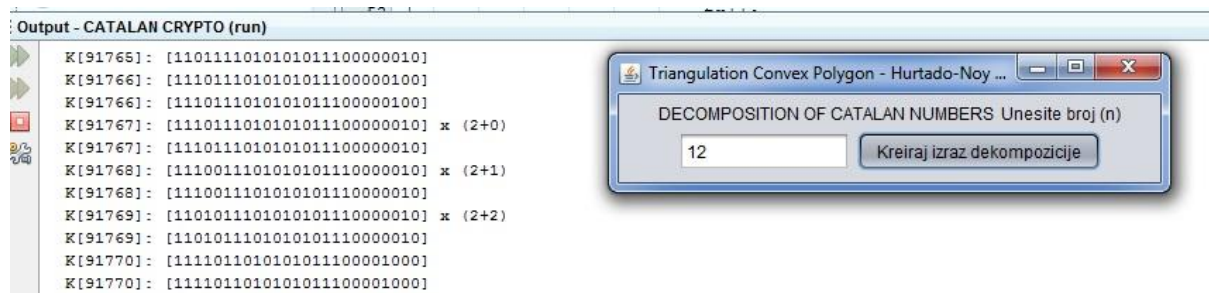
Slika 18. Dešifrovanje šifrata i prikazivanje u desnom panelu

Na slici 19, dat je postupak generisanja ključeva za datu osnovu n . Uslov za početak generisanja kompletnog prostora ključeva za određenu osnovu n , je da se odredi datoteka gde će se izvršiti snimanje kompletnog prostora ključeva. Nakon toga počinje generisanje ključeva i njihovo snimanje. Ovaj postupak može da potraje u zavisnosti od unete osnove n (slika 19).

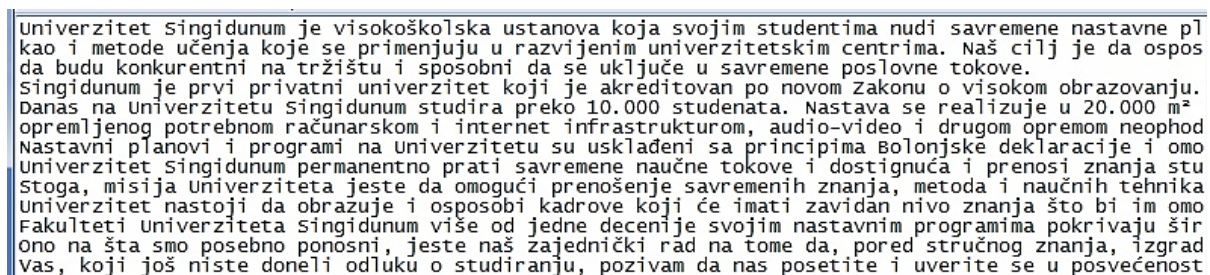


Slika 19. Generisanje svih validnih ključeva za zadatu osnovu n

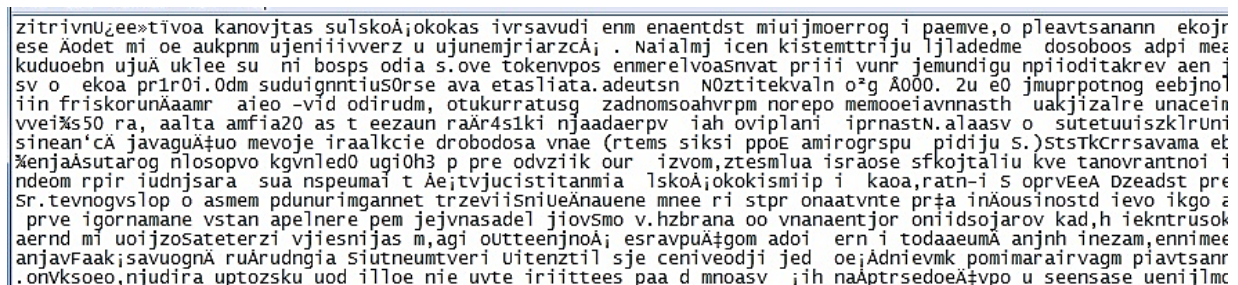
U aplikaciji postoji i druga opcija za ključeve, a to je dinamičko generisanje na osnovu izraza dekompozicije (opisano u trećem poglavlju ovog diplomskog rada). Ova varijanta generiše binarne zapise ključeva koji zadovoljavaju svojstvo bit-balansiranosti (slika 20).



Slika 20. Generisanje izraza dekompozicije za zadato n



Slika 21. Sadržaj otvorenog teksta (poruka)



Slika 22. Sadržaj šifrovanog teksta na osnovu 60-bitnog ključa $K=1142920503054772772$
(binarni zapis ključa je 111111011100011110000010010001011111000101110010101000100100)

Svaki kriptografski algoritam, sa aspekta kriptanalize, može se posmatrati kao matematička funkcija, gde kompleksnost algoritma, odnosno te matematičke funkcije direktno zavisi od njenih argumenata. Tačnije, ako su njeni argumenti ključ i otvoreni tekst (u postupku šifrovanja), odnosno ključ i šifrat (u postupku dešifrovanja), onda kompleksnost leži upravo u skupu svih mogućih vrednosti ključeva (tj. prostoru ključeva) koji mogu biti argumenti te funkcije.

U prilog navedenoj postavci, možemo reći da generisanje vrednosti na osnovu svojstava Katalanovih brojeva može obezbediti veliki prostor ključeva. Taj prostor može biti toliki da napadač ne može isprobati sve vrednosti u razumnom vremenu. Na slici 23, data je veličina prostora ključeva za osnove $n=\{150,...,200\}$.

C(150)= 620925183926009621146978506218967449531342090729015621989883130549504437230725772687824
C(151)= 2459190531075380210068954346998805293538604859334653976565195030202642573769058652355724
C(152)= 974032327994562357713585839399526802538819963893333538160184232723211762771565642663848
C(153)= 38581800004979418065278400132059178542122089478891782912192937563225059579809448324837320
C(154)= 152833710987466856077941533426350552418470728645416481987654604282710881174212911428710416
C(155)= 605456624296503314462614536265927188427018655787611447874170163119970029267074226044506648
C(156)= 239868802746767551565262685079023638226787285986588029157540391468925848433886424201803408
C(157)= 950362629130626871106932150977473794106313480815449541798140839368892781946715946917098419803376
C(158)= 37656021983627970863239722065976647657188136686863728482571084292154883214129294163550164320
C(159)= 149211987110125834545587398686432466341607991621697524112187921507663724735987328123067526118
C(160)= 59128725326869740640153791067974618173577010277285840891775738645276126593539846847932184244
C(161)= 2343249485175948981156905764602714227576768151839614258348889038334983168352176430101064582004
C(162)= 928674335842738062470773695664633982175412429990127674192529562971773783776110268989495214568
C(163)= 36807214530352423207683103791586102953744012679838920659909415950802761947893119968555926155300
C(164)= 145890413956669604714089393210286735343930813894634267342913685041363674629830911875367125488280
C(165)= 17428850831426867407655546580534167809075153871502095853236173236248782568847831409587762477640
C(166)= 2292377200702310552450706418181518677183279951275056212304445189714950263835792002354174244073040
C(167)= 9087638188498445404358157586362449041690859806840401413064050573512838545920461152189762181860980
C(168)= 36027914711798570537987962028774206259957846571497449389070496356530188318146206934716808649981400
C(169)= 14284008538677786201531697886702244128947993277584005224903026731184393684885549847053700176985080
C(170)= 566348408725622751148449775858720556691622190005859389137334807741187245136563759042704144561379440
C(171)= 2245637295066793699321178762416554765486548451069744787160827551624940123157770253878629224365469640
C(172)= 890466580587179466898458560796280746380186343548236554868946245171727887203643896882888138235330480
C(173)= 35311605781905392652869801189364561580473152741656800131376855799818920932014449935914901237829758800
C(174)= 144032574525946577153419064129621761275858121556704889869389019006160580494242082927782059654230382921360
C(175)= 555369012338453086550713186160469675464940287853618631988328749081229971640226744232350946052584062332
C(176)= 2202649981138949529709608229856778034894847921317741692970659784491657853623950138141866464005163908232
C(177)= 8736353295978080718960580956622951082223385575563627164254414650848935082351173019821110806672166961864
C(178)= 34652574525946577153419064129621761275858121556704889869389019006160580494242082927782059654230382921360
C(179)= 13745521228625475604189562104749965306090388217492939648190977539110363596049359561353503176113852254728
C(180)= 545264322770889032254591469127650557446016504981211639082934910114985694031129290886842493262153292369584
C(181)= 2163081544179021325757225498407492870747384157123268150647686841225382808189424989122529011732278445554064
C(182)= 8581405470349560013659812633026447126571589606948047417323610091418731796423620448650033128511662029903008
C(183)= 34045793442147710923759039250594056534767719636261275079599105253998229409724146345187631433769093922984760
C(184)= 135078985873169836854265593567221824305510844394679869775274287872620002090473099553339034985873053726869264
C(185)= 535958556851609352679828000282847883534768834211149160721249593817169685713812620808409719460076955109836112
C(186)= 2126637696170556896729584899518038126111221791361885974626562559424277576468711040854759421600947062521381792
C(187)= 8438679368846997047661012420427959798292401363595568814209657815587824851306693811051332598480353769366759664
C(188)= 3348682289249988284369096906460157929731751442839558786546261172967558933756721472425923009842673687963332000
C(189)= 132889812951349953507443679302478731994830213620531722763452004865355470716066147316890241839059873477496591200
C(190)= 527384702707451647951006852938632873571106292797712281961762406743138464936011202440852373371766408879279665600
C(191)= 2093058038870198727805558447600199216985328099540920619035744551761830782714794459687132856819197935239641172850
C(192)= 830716299365063329274123197337695647775965410613187534618550915282706629842137596478465120847179369914845276700
C(193)= 32971729407788596058818291853094105607667491578206981451836516519421052087517762625198031665218186158940365273500
C(194)= 130872402880145504356540296893819680719664812725806172531904942492471252901224350112324494917327569677024834470200
C(195)= 14833313473230624435654851956080161223975634187128582805214516628278748760982369323410495131024740860843475601100
C(196)= 2062111427086631209688741595074389269427152009818957115500902294433065896096894481273639630418585519559287299086600
C(197)= 8185957483283293589976519665295302857422936766251011579715703047597928254202823546874145199540445547341413217586200
C(198)= 32497017144692472040610304198911001293287035403710045969725655314584740305629299507691330189130411971857871567302000
C(199)= 129013158064429114001222907669676675134349530552728882499810851598901419013348319045534580850847735528275750122188940
C(200)= 512201493211017079467541693136328292324432464582475861864920694407578768023144072628540276213813397768975366156750120

BUILD SUCCESSFUL (total time: 16 seconds)

Slika 23. Izveštaj iz klase *find_catalan* za skup osnova $n=\{150,...,200\}$

4.3 NIST statistička baterija testova

Imajući u vidu da koristimo Katalanove brojeve za generisanje ključeva, primenićemo NIST (*National Institute of Standards and Technology*) statističku bateriju testova radi procene kvaliteta.

NIST-ov paket testova¹ sastoji se od više statističkih testova (slika 24) koji su razvijeni za testiranje slučajnosti binarne sekvence koje proizvode softver ili hardver na bazi slučajnih ili pseudo-slučajnih brojeva. NIST statistički testovi se fokusiraju na različite tipove neslučajnosti koje bi mogle da postoje u nizu. Neki testovi su podeljeni u podtestove [37].

¹ http://csrc.nist.gov/groups/ST/toolkit/rng/documentation_software.html

```

GENERATOR SELECTION
-----
[0] Input File           [1] Linear Congruential
[2] Quadratic Congruential I [3] Quadratic Congruential II
[4] Cubic Congruential   [5] XOR
[6] Modular Exponentiation [7] Blum-Blum-Shub
[8] Micali-Schnorr       [9] G Using SHA-1

Enter Choice: 0

User Prescribed Input File: Catalan.txt

STATISTICAL TESTS
-----
[01] Frequency           [02] Block Frequency
[03] Cumulative Sums     [04] Runs
[05] Longest Run of Ones [06] Rank
[07] Discrete Fourier Transform [08] Nonperiodic Template Matchings
[09] Overlapping Template Matchings [10] Universal Statistical
[11] Approximate Entropy [12] Random Excursions
[13] Random Excursions Variant [14] Serial
[15] Linear Complexity

INSTRUCTIONS
Enter 0 if you DO NOT want to apply all of the
statistical tests to each sequence and 1 if you DO.

Enter Choice: 1

Parameter Adjustments
-----
[1] Block Frequency Test - block length(M): 128
[2] NonOverlapping Template Test - block length(m): 9
[3] Overlapping Template Test - block length(m): 9
[4] Approximate Entropy Test - block length(m): 10
[5] Serial Test - block length(m): 16
[6] Linear Complexity Test - block length(M): 500

Select Test (0 to continue): 0

How many bitstreams? 5

Input File Format:
[0] ASCII - A sequence of ASCII 0's and 1's
[1] Binary - Each byte in data file contains 8 bits of data

```

Slika 24. NIST Statistical Test Suite

Za ulazni parametar uzeli smo 100-bitni ključ $\varepsilon = 11101101011001010010100111111010100100100011101110101000110111010100010100100110111100101010100001011010110000$. Ključ zadovoljava svojstvo Katalanovog broja i pripada osnovi $n=50$.

1) Test za ispitivanje učestalosti u nizu bazira se na analizi odnosa jedinica i nula u nizu bitova, tačnije cilj je uočavanje jednakosti pojavljivanja jedinica i nula. Potreban je približan broj jedinica i nula u sekvenci, što znači da svojstvo Katalanovog broja odnosno svojstvo bit-balansiranosti uvek daje dobar rezultat za ovaj test. Svi podtestovi koji proizilaze iz ovog testa direktno zavise od njegove uspešnosti. Poziv ovog testa se vrši preko metode **Frequency(n)** gde je n dužina bitnog uzorka. Metoda koristi parametar ε (niz bitova u zapisu ključa).

1		FREQUENCY TEST
2		-----
3		COMPUTATIONAL INFORMATION:
4		-----
5		(a) The nth partial sum = 6
6		(b) S_n/n = 0.006000
7		-----
8	SUCCESS	p_value = 0.849515
9		

Pošto je $P \geq 0.01$, smatra se da je sekvenca slučajna.

2) Test za ispitivanje učestalosti u bloku prikazuje odnos jedinica i nula u n -bitnim blokovima. Cilj ovog testa je uočavanje jednakosti broja jedinica i nula u svakom n -bitnom bloku. Ovaj test se poziva preko metode **BlockFrequency(M, n)** gde je M dužina svakog bloka, a n je dužina bitnog uzorka. Kao i kod testa za ispitivanje učestalosti u nizu i kod ovog testa se koristi parametar ϵ .

1		BLOCK FREQUENCY TEST
2		-----
3		COMPUTATIONAL INFORMATION:
4		-----
5		(a) Chi^2 = 1.718750
6		(b) # of substrings = 7
7		(c) block length = 128
8		(d) Note: 104 bits were discarded.
9		-----
10	SUCCESS	p_value = 0.973758

Test pokazuje da je sekvenca slučajna pošto je $P \geq 0.01$.

3) Test za ispitivanje uzastopnih ponavljanja istih bitova u nizu odnosi se na ukupan broj uzastopnih ponavljanja jednog broja u nizu. Svrha testa je da se odredi da li broj uzastopnih ponavljanja nula i jedinica odgovara očekivanoj slučajnoj sekvenci. Ovaj test se poziva preko metode **Runs(n)** gde je n dužina bitnog uzorka. Kao dodatni parametar koristi se ulaz ϵ .

1		RUNS TEST
2		-----
3		COMPUTATIONAL INFORMATION:
4		-----
5		(a) Pi = 0.503000
6		(b) V_n_obs (Total # of runs) = 465
7		(c) V_n_obs - 2 n pi (1-pi)
8		-----
9		2 sqrt(2n) pi (1-pi)
10		-----
11	SUCCESS	p_value = 0.026930

Smatra se da je sekvenca slučajna, jer je rezultat testa $P \geq 0.01$.

4) Test za ispitivanje najdužeg uzastopnog ponavljanja jedinica u n -bitnim blokovima određuje da li se dužina najdužeg uzastopnog ponavljanja poklapa sa dužinom koja se očekuje u nizu slučajnih brojeva. Ovaj test se poziva preko metode **LongestRunOfOnes(n)** gde je n dužina bitnog uzorka. Test je podešen tako da za M (dužina svakog bloka) koristi tri vrednosti: $M = 8$ (gde je n minimum 128), $M = 128$ (gde je n

```

1          LONGEST RUNS OF ONES TEST
2          -----
3          COMPUTATIONAL INFORMATION:
4          -----
5          (a) N (# of substrings)    = 125
6          (b) M (Substring Length)  = 8
7          (c) Chi^2                  = 3.500324
8          -----
9          F R E Q U E N C Y
10         -----
11         <=1      2      3      >=4      P-value      Assignment
12         26  47  22   30 SUCCESS          p value = 0.320720

```

5) **Test za ispitivanje stanja binarne matrice** odnosi se na proveru linearne zavisnosti između podnizova fiksne dužine iz originalnog niza. Ovaj test se poziva preko metode **Rank(n)** gde je n dužina bitnog uzorka. Metoda koristi parametar ε (niz bitova u zapisu ključa).

Smatra se da je sekvenca slučajna, jer je rezultat testa $P \geq 0.01$.

```

2      -----
3      COMPUTATIONAL INFORMATION:
4      -----
5      (a) Percentile = 95.000000
6      (b) N_l        = 475.000000
7      (c) N_o        = 475.000000
8      (d) d          = 0.000000
9      -----
10     SUCCESS      p value = 1.000000

```

43

očekuje za niz slučajnih brojeva. Ovaj test se poziva preko **NonOverlappingTemplateMatching (m,n)** gde su m dužina bita u šablonu i n dužina bitnog uzorka.

1	NONPERIODIC TEMPLATES TEST										
2	-----										
3	COMPUTATIONAL INFORMATION										
4	-----										
5	LAMBDA = 0.228516 M = 125 N = 8 m = 9 n = 1000										
6	-----										
7	F R E Q U E N C Y										
8	Template	W_1	W_2	W_3	W_4	W_5	W_6	W_7	W_8	Chi^2	P_value Assignment Index
9	-----										
10										1.769891 0.987272 SUCCESS 148	
11	128 Templates = SUCCESS, 20 Templates = FAILURE										
12	-----										

8) **Test za ispitivanje preklapajućih uzoraka** ispituje učestalost pojave svih mogućih n -bitnih uzoraka gde dolazi do preklapanja u celom ispitivanom nizu. Test bi trebalo da otkrije da li je broj pojava uzoraka koji se preklapaju približno jednak broju koji se očekuje za niz slučajnih brojeva. Test se poziva preko metode **OverlappingTemplateMatching (m,n)** gde je m dužina bita u šablonu, a n je dužina bitnog uzorka.

1	OVERLAPPING TEMPLATE OF ALL ONES TEST									
2	-----									
3	COMPUTATIONAL INFORMATION:									
4	-----									
5	(a) n (sequence_length)	= 1000								
6	(b) m (block length of 1s)	= 9								
7	(c) M (length of substring)	= 1032								
8	(d) N (number of substrings)	= 0								
9	(e) lambda [(M-m+1)/2^m]	= 2.000000								
10	(f) eta	= 1.000000								
11	-----									
12	F R E Q U E N C Y									
13	0	1	2	3	4	>=5	Chi^2	P-value	Assignment	
14	-----									
15	0	0	0	0	0	0	nan nan	SUCCESS		

9) **Test za ispitivanje linearne kompleksnosti** utvrđuje da li je sekvenca dovoljno kompleksna da se može smatrati slučajnom. Ovaj test se poziva preko metode **LinearComplexity (M, n)** gde je M dužina svakog bloka, a n dužina bitnog uzorka. Metoda koristi i dodatni parametar ϵ (niz bitova u zapisu ključa).

1	-----									
2	L I N E A R C O M P L E X I T Y									
3	-----									
4	M (substring length) = 500									
5	N (number of substrings) = 2									
6	-----									
7	F R E Q U E N C Y									
8	-----									
9	C0	C1	C2	C3	C4	C5	C6	CHI2	P-value	
10	-----									
11	Note: 0 bits were discarded!									
12	0	0	0	2	0	0	0	2.000106 0.919689		

Pošto je $P \geq 0.01$, smatra se da je sekvenca slučajna.

10) Serijski test se koristi za utvrđivanje učestalosti svih mogućih preklapanja n -bitnog niza u čitavoj sekvenci. Ovaj test se poziva preko metode **Serial (m, n)** gde je m dužina svakog bloka, a n je dužina bitnog uzorka. Takođe, i ovde se koristi dodati parametar ε .

1		SERIAL TEST
2		-----
3		COMPUTATIONAL INFORMATION:
4		-----
5		(a) Block length (m) = 16
6		(b) Sequence length (n) = 1000
7		(c) Psi_m = 146324.928000
8		(d) Psi_m-1 = 97500.608000
9		(e) Psi_m-2 = 64765.376000
10		(f) Del_1 = 48824.320000
11		(g) Del_2 = 16089.088000
12		-----
13	SUCCESS	p_value1 = 0.011201
14	SUCCESS	p_value2 = 0.949014

Pošto su P_1 i $P_2 \geq 0.01$, smatra se da je sekvenca slučajna.

11) Testom za ispitivanje približne entropije proverava se učestalost pojave svih mogućih preklapajući n -bitnih uzoraka u nizu. Cilj je poređenje učestalosti preklapajućih blokova sa očekivanim rezultatima. Test se poziva preko metode **ApproximateEntropy (m, n)** gde je m dužina svakog bloka, a n je dužina bitnog uzorka. Koristi se i dodati parametar ε .

1		APPROXIMATE ENTROPY TEST
2		-----
3		COMPUTATIONAL INFORMATION:
4		-----
5		(a) m (block length) = 10
6		(b) n (sequence length) = 1000
7		(c) Chi^2 = 973.830581
8		(d) Phi(m) = -5.747581
9		(e) Phi(m+1) = -5.953813
10		(f) ApEn = 0.206232
11		(g) Log(2) = 0.693147
12		-----
13		Note: The blockSize = 10 exceeds recommended value of 4
14		Results are inaccurate!
15		-----
16	SUCCESS	p_value = 0.867027

12) Testom za ispitivanje slučajnih zbirova utvrđuje se da li je kumulativni zbir parcijalnih sekvenci koje se javljaju u nizu koji se testira suviše veliki ili suviše mali u odnosu na očekivano ponašanje tog kumulativnog zbira slučajnih sekvenci. Test se poziva preko metode **CumulativeSums (mode,n)** gde su $mode = 0$ ili $mode = 1$, a n je dužina bitnog uzorka.

1		CUMULATIVE SUMS (FORWARD) TEST
2		-----
3		COMPUTATIONAL INFORMATION:
4		-----
5		(a) The maximum partial sum = 14
6		-----
7	SUCCESS	p_value = 0.997649
8		-----
9		CUMULATIVE SUMS (REVERSE) TEST
10		-----
11		COMPUTATIONAL INFORMATION:
12		-----
13		(a) The maximum partial sum = 20
14		-----
15	SUCCESS	p_value = 0.941731

Možemo uočiti da su P_1 i $P_2 \geq 0.01$, tačnije status i za *forward* i *reverse* test je uspješan, pa se smatra da je sekvenca slučajna.

5 ZAKLJUČNA RAZMATRANJA

5.1 Sumirani ciljevi rada i ostvareni rezultati

Imajući u vidu činjenicu da je kriptografija veoma dinamična oblast, da je aktuelna i da je veoma rasprostranjena, ovim radom su obuhvaćeni samo neki njeni osnovni matematički koncepti i dat je mali doprinos kada je reč o primeni teorije brojeva u oblasti kriptografije.

U ovom radu su razmatrane mogućnosti primene Katalanovih brojeva i kombinatornih problema u kriptografiji. Primijenjene su autorske metode: *dekompozicija Katalanovih brojeva* i šifrovanje na osnovu *LatticePath* kombinatornog problema. Pored toga, prikazane su i neke mogućnosti primene Katalanovih brojeva i njihovih svojstava u oblasti steganografije odnosno u postupku skrivanja informacija u tekstu.

Pre teorijskih razmatranja, naveli smo nekoliko sličnih studija i istraživanja na polju primene Katalanovih brojeva u kriptografiji i steganografiji. Navedena su i neka slična istraživanja na polju primene kombinatornih problema u kriptografiji. To nam je poslužilo kao osnov za naše dalje istraživanje i konkretnu implementaciju.

Sa aspekta ostvarenih rezultata, možemo reći da smo dali nekoliko predloga primene Katalanovih brojeva, pre svega kao generatora pseudoslučajnih brojeva u kombinaciji sa nekoliko kombinatornih problema, a sve u cilju šifrovanja i dešifrovanja teksta. Pokazano je i koliko Katalanovi brojevi imaju široku primenu u rešavanju mnogih kombinatornih problema, kao što su stek permutacije, problem uparenih zagrada, *Ballot* problema, problem kretanja kroz celobrojnu mrežu itd.

Navedene su teorijske osnove istraživanja gde su ispitana osnovna svojstva Katalanovih brojeva, pre svega, akcenat je stavljen na svojstvo balansiranosti bitova u binarnom zapisu Katalanovog broja. Takođe, analiziran je i algoritam za dekompoziciju Katalanovih brojeva koji smo u eksperimentalnom delu rada koristili za efikasno generisanje ključeva. Nakon toga, dali smo i nekoliko predloga i primera primene kombinatornih problema u šifrovanju fajlova. Akcenat smo stavili na primenu metode kretanja kroz celobrojnu mrežu (*LatticePath*) u šifrovanju teksta, a zatim su prikazani i ekvivalentni kombinatorni problemi u šifrovanju, kao što su *Ballot* problem, stek permutacije i *Balanced Parentheses*. Dali smo i jedan primer primene dekompozicije Katalanovih brojeva u steganografiji. Pre svega, ideja da se primene Katalanovi brojevi i navedeni kombinatorni problemi u kriptografiji se javila na osnovu naših ranijih istraživanjima u oblasti teorije brojeva, primenjene matematike i kombinatornih problema.

Naše teorijske osnove istraživanja, koje se odnose na navedeni postupak generisanja ključeva na osnovu svojstva balansiranosti bitova u Katalanovom broju kao i šifrovanja na osnovu kretanja kroz celobrojnu mrežu, praćene su eksperimentalnim ispitivanjem. Tačnije, data je studija slučaja koja obuhvata konkretne algoritme za šifrovanje i dešifrovanje, koji su implementirani u *Java* programskom jeziku. Realizovana je *GUI* aplikacija koja poseduje sve neophodne elemente za jednostavno i efikasno šifrovanje i dešifrovanje fajlova, generisanje i učitavanje ključeva, prikazivanje sadržaja šifrata i poruke itd. U delu sa eksperimentalnim radom, primenili smo *NIST* statističku bateriju testova radi procene kvaliteta ključeva koji se baziraju na svojstvu Katalanovih brojeva.

5.2 Predlog za budući rad

Predložene metode se mogu dalje unapređivati i prilagođavati savremenijim pristupima u kriptografiji. Neke studije se bave primenom teorije brojeva u realizaciji algoritama vizuelne kriptografije, odnosno u rešavanju problema deljenja tajne. Vizuelna kriptografija se prvenstveno bazira na kriptografskim metodama kojim se vrši kriptovanje ali i skrivanje podataka u nekom skupu slika, a rekonstrukcija zaštićenih, odnosno kriptovanih podataka se vrši direktnim odnosno vizuelnim pregledom. Pored toga, teorija brojeva danas pronalazi sve veću primenu u realizaciji osnovnih kriptografskih tehnika koje se bave sigurnom razmenom podataka.

Pored steganografije i vizuelne kriptografije, mogu se dati još neki predlozi za budući rad na polju primene Katalanovih brojeva u kriptografiji. U radu "*From entanglement witness to generalized Catalan numbers*" (rad je novijeg datuma, objavljen 2016. godine) autori navode mogućnost primene Katalanovih brojeva u kvantnoj kriptografiji [35].

U mnogim naučnim studijama, radovima i monografijama, kada se govori o budućnosti kriptografije, navodi se kvantna kriptografija koja je nastala kao posledica otkrića u oblasti kvantnog računarstva [20]. Bitno je napomenuti da će kvantna i DNK kriptografija, u nekoj skorijoj budućnosti, kod nas predstavljati osnov za zaštitu poverljivih dokumenata. Shodno tome, predlog za budući rad bi se mogao odnositi upravo na primenu Katalanovih brojeva u kvantnoj kriptografiji i unapređenju postojećih algoritama i metoda u kriptologiji.

6 LITERATURA

- [1] Koshy, T (2009) *Catalan Numbers with Applications*, Oxford University Press, New York.
- [2] Stanley, R. P. (2012) *Catalan addendum to Enumerative Combinatorics* [on-line], <http://www-math.mit.edu/~rstan/ec/catadd.pdf>. [Dostupno dana 24.05.2017.]
- [3] Campbell, D.M. (1984) The computation of Catalan numbers, *Mathematical magazine*, Vol. 57, No. 4, pp. 195–208.
- [4] Singmaster, D. (1978) An elementary evaluation of the Catalan numbers, *American Mathematical Monthly*, 85, pp. 366–368.
- [5] Saračević, M., Mašović, S., Milošević, D. (2013) Java implementation for triangulation of convex polygon based on Lukasiewicz's algorithm and binary trees, *Southeast European Journal of Soft Computing*, Vol.2, No.2, pp. 40-45.
- [6] Knuth, D. E. (2006) *The Art of Computer Programming, Vol4: Generating all trees: history of combinatorial generation*, Addison-Wesley Professional, United States.
- [7] Du, D. , Hwang, F. (1992) *Computing in Euclidean Geometry*, World Scientific, Singapore.
- [8] Geary, F.R., Rahman, N., Raman, R. (2006) A Simple Optimal Representation for Balanced Parentheses, *Theoretical Computer Science*, 368, No. 3, pp. 231-246.
- [9] Hilton, P., Pedersen, J. (1991) Catalan Numbers, Their Generalization, and Their Uses, *The mathematical intelligencer*, Vol.13, No. 2, pp. 64-75.
- [10] Carlitz, L. (1972) Sequences, paths and ballot numbers, *Fibonacci Quarterly*, Vol. 10, pp. 531-549.
- [11] Takacs, L. (1962) A Generalization of the Ballot Problem and its Application in the Theory of Queues, *Journal of the American Statistical Association*, Vol. 57, No. 298, pp. 327-337
- [12] Stanimirović, P., Krtolica, P., Saračević, M., Mašović, S. (2014) Decomposition of Catalan numbers and Convex Polygon Triangulations, *International Journal of Computer Mathematics*, Vol. 91, No. 6, pp. 1315-1328.
- [13] Saračević, M., Stanimirović, P., Krtolica, P., Mašović, S. (2014) Construction and Notation of Convex Polygon Triangulation based on ballot problem, *ROMJIST-Journal of Information Science and Technology*, Vol.17, No.3, pp. 237-251.
- [14] Mašović, S., Saračević, M., Stanimirović, P. (2014) Alpha-Numeric notation for one Data Structure in Software Engineering, *Acta Polytechnica Hungarica: Journal of Applied Sciences*, Vol.11, No.1, pp.193-204.
- [15] Saračević, M. (2013) *Metode za rešavanje problema triangulacije poligona i njihova implementacija*. Doktorska disertacija, Prirodno-matematički fakultet, Univerzitet u Nišu.
- [16] Kumar S.D., Suneetha, H., Chandrasekhar, A. (2012) A Novel Approach for Enciphering Data based ECC using Catalan Numbers, *International Journal of Engineering Research and Application*, Vol. 2, No. 2, pp.161-166.
- [17] Amounas, F., El-Kinani, E.H., Hajar, M. (2013) Novel Encryption Schemes Based on Catalan Numbers, *International Journal of Information & Network Security*, Vol.2, No.4, pp. 339-347.
- [18] Pund-Dange, S., Desai, C.G. (2017) Data Hiding Technique using Catalan-Lucas Number Sequence, *Indian Journal of Science and Technology*, Vol 10(4), pp. 12-17.

- [19] Aroukatos, N., Manes, K., Zimeras, S., Georgiakodis, F. (2013) Techniques in Image Steganography using Famous Number Sequences, *International Journal of Computers & Technology*, Vol. 11, No.3, pp. 2321-2329.
- [20] Kościelny, C., Kurkowski, M., Srebrny, M. (2013) *Modern Cryptography Primer: Theoretical Foundations and Practical Applications*, Springer Science & Business Media, Berlin, Germany.
- [21] Higgins, P.M. (2008) *Number Story: From Counting to Cryptography*, Springer Science & Business Media, Berlin, Germany.
- [22] Lachaud, G., Ritzenthaler, C., Tsfasman, M.A. (2009) *Arithmetic, Geometry, Cryptography, and Coding Theory*, American Mathematical Society, United States.
- [23] Srikantaswamy, S.G., Phaneendra, H.D. (2012) A Cryptosystem Design with Recursive Key Generation Techniques, *Procedia Engineering*, Vol. 30, pp. 170 – 173.
- [24] Horak, P., Semaev, I., Tuza, I. Z. (2015) An application of Combinatorics in Cryptography, *Electronic Notes in Discrete Mathematics*, Vol. 49, pp. 31-35.
- [25] Plantard, T., Rose, M., Susilo, W. (2010) Improvement of Lattice-Based Cryptography Using CRT. *Quantum Communication and Quantum Networking (spec. Ed.: Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering)*, Vol 36. pp. 275-282.
- [26] Sandeep, K. (2010) *Recursive Information Hiding in Visual Cryptography* [on-line], <https://arxiv.org/abs/1004.4914> [Dostupno dana 24.05.2017.]
- [27] Damgard, I., Groth, J., Salomonsen, G. (2003) The Theory and Implementation of an Electronic Voting System, *Advances in Information Security*, Vol. 7, pp 77-99.
- [28] Rivest, R.L. (2006) *The ThreeBallot Voting System: Publication on Computer Science and Artificial Intelligence. Massachusetts Institute of Technology* [on-line], <https://people.csail.mit.edu/rivest/pubs/Riv06c.pdf> [Dostupno dana 24.05.2017.]
- [29] Adamović, S. (2015) *Zaštita informacionih sistema – Java implementacija kriptografskih mehanizama*, Univerzitet Singidunum, Beograd.
- [30] Veinović, M., Adamović, S. (2013) *Kriptologija 1 - Osnove za analizu i sintezu šifarskih sistema*, Univerzitet Singidunum, Beograd.
- [31] Milosavljević, M., Adamović, S. (2014) *Kriptologija 2 - Osnove za analizu i sintezu šifarskih sistema*, Univerzitet Singidunum, Beograd.
- [32] Milosavljević, M., Adamović, S. (2013) *Osnovi teorije informacija i kodovanja*, Univerzitet Singidunum, Beograd.
- [33] Chouse, C. (2002) *Catalan sayılarını kullanarak Stack Permutasyonu yöntemi ile bir dosyann şifrelenmesi: Encryption Using Catalan Numbers and Stack* [on-line]. www.chasanc.com/ [Dostupno dana 24.05.2017.]
- [34] Pratama, G.M., Tamatjita, E.N. (2015) Modifikasi algoritma vigenère cipher menggunakan metode catalan number dan double columnar transposition, *Journal Compiler*, Vol 4, No 1, pp. 31-40.
- [35] Cohen, E., Hansen, T., Itzhaki, N. (2016) From entanglement witness to generalized Catalan numbers, *Scientific Reports*, Vol. 6, Article number: 30232.
- [36] Web portal - Java with us (2015) *Factorial of large numbers using BigInteger - Java Programs* [on-line], penguin.ewu.edu/~trolfe/BaseChange/Cat.java [Dostupno dana 24.05.2017.]
- [37] Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E. (2010) *A statistical test suite for random and pseudorandom number generators for cryptographic applications*. NIST Special Publication, USA.

7 PRILOG: JAVA SOURCE CODE

7.1 Klasa CatalanCrypto

Metod za šifrovanje na osnovu Katalanovog broja (ključ) i *LatticePath* (postupak šifrovanja):

```
1  File sifrovanjeCatalanPath(File file, String key) {
2      short[] K = new short[60];
3      byte[] tekst = new byte[segment];
4      byte[] putanja = new byte[segment];
5      byte[] sifrat = new byte[segment];
6      short n, i, doCilja, slobodan, zauzet;
7      long j, s;
8      BigInteger start;
9      long count = 0;
10     File fileOut = null;
11     try {
12         fileOut = new File(file.getParent() + "\\SIFRAT.txt");
13         FileOutputStream out;
14         try (FileInputStream in = new FileInputStream(file)) {
15             out = new FileOutputStream(fileOut);
16             start = new BigInteger(key);
17             n = 0;
18             //citanje binarnog zapisa kljuca
19             while (BigInteger.valueOf(1).compareTo(start) < 0) {
20                 if (start.and(BigInteger.valueOf(1)).compareTo(BigInteger.valueOf(0)) > 0) {
21                     K[n] = 1;
22                 } else {
23                     K[n] = 0;
24                 }
25                 n++;
26                 start = start.shiftRight(1);
27             }
28             //citanje poruke
29             K[n] = 1;
30             count = file.length() / segment;
31             for (j = 0; j < count; j++) {
32                 in.read(tekst, 0, segment);
33                 doCilja = segment - 1;
34                 slobodan = zauzet = 0;
35
36                 //citanje binarnog zapisa kljuca (raspored n-parova 0 i 1)
37                 for (i=n; i>=0; i--) {
38                     // pomeraj udesno (napred, ako je bit 1)
39                     if (K[i] == 1)
40                     {
41                         putanja[slobodan] = tekst[doCilja];
42                         slobodan++;
43                         doCilja--;
44                     }
45                     // pomeraj gore (ako je bit 0)
46                     else {
47                         sifrat[zauzet] = putanja[slobodan-1];
48                         zauzet++;
49                         slobodan--;
50                         putanja[slobodan] = 0;
51                     }
52                 }
53                 out.write(sifrat);
54             }
55             count = file.length() - (file.length()-in.available());
56             in.read(tekst, 0, (int) count);
57             out.write(tekst);
58             out.flush();
59         }
60         out.close();
61     } catch (FileNotFoundException ex) {
62         Logger.getLogger(CatalanCrypto.class.getName()).log(Level.SEVERE, null, ex);
63     } catch (IOException ex) {
64         Logger.getLogger(CatalanCrypto.class.getName()).log(Level.SEVERE, null, ex);
65     }
66     return fileOut;
67 }
```

Metod za dešifrovanje na osnovu Katalanovog broja (ključ) i *LatticePath* (postupak dešifrovanja):

```
1 File desifrovanjeCatalanPath(File file, String key) {
2     short[] A = new short[60];
3     byte[] sifrat = new byte[segment];
4     byte[] putanja = new byte[segment];
5     byte[] tekst = new byte[segment];
6     short n, i, slobodan, doCilja, zauzet;
7     long j, s;
8     BigInteger end;
9     long count = 0;
10    File fileDesif = null;
11    try {
12        fileDesif = new File(file.getParent() + "\\DESIFROVANO.txt");
13        FileOutputStream out;
14        try (FileInputStream in = new FileInputStream(file)) {
15            out = new FileOutputStream(fileDesif);
16            end = new BigInteger(key);
17            n = 0;
18            //citanje binarnog zapisa kljuca
19            while (BigInteger.valueOf(1).compareTo(end) < 0) {
20                if (end.and(BigInteger.valueOf(1)).compareTo(BigInteger.valueOf(0)) > 0) {
21                    A[n] = 1;
22                } else {
23                    A[n] = 0;
24                }
25                n++;
26                end = end.shiftRight(1);
27            }
28            //citanje sifrata
29            A[n] = 1;
30            for(i=0; i<=n; i++)
31                A[i] = (short) (A[i] ^ 1);
32            count = file.length() / segment;
33            for (j = 0; j < count; j++) {
34                in.read(sifrat, 0, segment);
35                doCilja = segment - 1;
36                slobodan = zauzet = 0;
37                //citanje binarnog zapisa kljuca (raspored n-parova 0 i 1)
38                for (i = 0; i <= n; i++) {
39                    if (A[i] == 1) {
40                        putanja[slobodan] = sifrat[doCilja];
41                        slobodan++;
42                        doCilja--;
43                    }
44                    else{
45                        tekst[zauzet] = putanja[slobodan-1];
46                        slobodan--;
47                        zauzet++;
48                    }
49                }
50                out.write(tekst);
51            }
52            count = file.length() - (file.length()-in.available());
53            in.read(sifrat, 0, (int) count);
54            out.write(sifrat);
55            out.flush();
56        }
57        out.close();
58    } catch (FileNotFoundException ex) {
59        Logger.getLogger(CatalanCrypto.class.getName()).log(Level.SEVERE, null, ex);
60    } catch (IOException ex) {
61        Logger.getLogger(CatalanCrypto.class.getName()).log(Level.SEVERE, null, ex);
62    }
63    return fileDesif;
64 }
```

7.2 Klasa CatalanNumbers

Klasa sa dve metode za generisanje ključeva koje zadovoljavaju svojstvo Katalanovog broja tj. svojstvo bit-balansiranosti [36].

```
1 public class CatalanNumbers{
2
3     static BigInteger catalan(int n){
4         BigInteger num = BigInteger.ONE, denom1 = num, denom2 = num,
5         factorial = num, factor = num;
6         int k, numStop = n << 1, denomStop = n;
7         if (n < 2) return factorial;
8         for (k = 1; k <= numStop; k++)
9         { factorial = factorial.multiply(factor);
10             if (k == numStop)
11             { num = factorial;
12             }
13             else if (k == denomStop)
14             { denom1 = factorial;
15                 k++;
16                 factor = factor.add(BigInteger.ONE);
17                 denom2 = factorial = factorial.multiply(factor);
18             }
19             factor = factor.add(BigInteger.ONE);
20         }
21         return num.divide(denom1).divide(denom2);
22     }
23
24     public static int find_catalan(File file) throws IOException {
25         short[] A = new short[67];
26         short n, i;
27         long m, s, count, cstart, cend;
28         FileWriter writer = new FileWriter(file);
29         s = cstart;
30         while (s < cend) {
31             m = s;
32             n = 0;
33             while (m > 1) {
34                 if ((m & 1) > 0) {
35                     A[n] = 1;
36                 } else {
37                     A[n] = 0;
38                 }
39                 n++;
40                 m = m >> 1;
41             }
42             A[n] = 1;
43             n++;
44             count = 0;
45             i = (short) (n - 1);
46             while ((i >= 0) && (count >= 0)) {
47                 if (A[i] == 1) {
48                     count++;
49                 } else {
50                     count--;
51                 }
52                 i--;
53             }
54             if (count == 0) // svojstvo bit-balansiranosti
55             {
56                 System.out.println(String.format("%d", s));
57                 writer.write(String.format("%d", s) + "\r\n");
58             }
59             s++;
60         }
61         writer.flush();
62         writer.close();
63         return 0;
64     }
65 }
```

7.3 Klasa *CatalanDecomposition*

Metod za dinamičko generisanje binarnih zapisa ključeva na osnovu izraza dekompozicije Katalanovog broja [15].

```
1 public Vector<Node> createExprDecomposition (int k) throws IOException {
2
3     // Korak 1.
4     Vector<Vector> expression = new Vector<>();
5     expression.add(new Vector<>());
6
7     // Korak 2. - postavljanje inicijalnog segmenta
8     expression.get(0).add(new LeafNode());
9
10    // Korak 3. - ciklus
11    for (int n=0; n <=k; n++) {
12        int tr=0;
13
14        //kreiranje segmenta (expr) za veći nivo
15        Vector<Node> exprSum = new Vector();
16
17        for (int j = 0; j < expression.get(n).size(); j++) {
18            Node t = (Node)expression.get(n).get(j);
19            Node s = new Node(new LeafNode(), t.copy());
20            exprSum.add(s);
21            for (int e = 0; e < t.leftSide(); e++) {
22                s = t.copy();
23                Node r = s;
24                if(n>=2 && e==t.leftSide()-1){
25                    tr++;
26                    System.out.print("K["+tr+"]: "+"["+t+"]");
27                    if(e>=2) System.out.println(" x " + "(2+"+(e-1)+")");
28                    if(e==1) System.out.println(" x (2+0)");
29                    if(e==0) System.out.println(" ");
30                }
31                for (int i=0; i<e; i++){
32                    s = s.getLeft();
33                }
34                s.setLeft(new Node(new LeafNode(), s.getLeft()));
35                exprSum.add(r);
36            }
37        }
38
39        // 3.3 - spajanje segmenata
40        expression.add(exprSum);
41    }
42    return expression.get(k);
43 }
```