Lukas Šerelis
A5 užduotis

- Programa suprogramuota Java programavimo kalba.
- Visos užduoties dalys yra realizuotos.
- Visos naudojamos bibliotekos yra Java standartinės bibliotekos.
- Visas kodas yra viename faile `ReedMuller.java`.
- Norint paleisti programą reikia atidaryti komandinę eilutę ten kur yra failas ir įvesti `java -jar KTK.jar`
  - Arba vykdykite failą `ReedMuller.java` savo programavimo aplinkoje
    - Arba galima atidaryti komandinę eilutę ten kur yra failas `ReedMuller.java` ir joje įvesti: `javac ReedMuller.java && java ReedMuller`
  - Arba vykdykite failą `KTK.bat`
  - Norint, kad programa dekoduotų paveikslėlį kai m>5 gali reikti paleisti programą naudojant: `java -Xms2g -Xmx4g -jar KTK.jar` (plačiau apie tai Kodo paveikslėlio pilnas apdorojimo efektyvumas)

Vartotojo sąsaja

- Paleidus programą vartotojas yra paprašomas pasirinkti vieną iš trijų variantų:
  1. Užrašyti programos nurodyto ilgio vektorių iš kūno F_q elementų.
  2. Užrašyti tekstą (tekstas gali būti sudarytas iš kelių eilučių).
  3. Nurodyti paveiksliuką (`.bmp` formato).
- Vartotojas pasirenka scenarijų įrašydamas skaičių nuo 1 iki 3 atitinkamai pagal varianto skaičių.

Pirmas scenarijus (užrašyti vektorių)

- Vartotojas yra paprašomas įvesti kodo parametrą `m` Rydo-Miulerio kodui (1, m).
- Yra iškviečiama funkcija `generateReedMullerMatrix`, kuri paima vartotojo įvestą parametrą `m`, grąžina dvejetainę matricą.
  - Pirmoji eilutė pilna vienetų, kitos eilutės atitinka binary derinius.
- Išspausdiname į ekraną Rydo-Miulerio generuojančią matricą.
- Vartotojas yra paprašomas įvesti informacijos vektorių `m + 1` ilgio. Informacijos vektoriaus ilgis turi sutapti su Rydo-Miulerio generuojančios matricos eilučių skaičiumi tam kad galėtumę dauginti šias dvi matricas kai norėsime užkoduoti vektorių.
  - Vektoriaus skaitymas yra daromas kas vieną integerį, kad vartotojui būtų aiškiau.
  - Bet vartotojas gali ir iš karto vesti pilną savo vektorių atskirdamas tarpais, programa automatiškai įrašys į tinkamas indekso vietas.
  - Jeigu vartotojo įvestis nėra `0` arba 1, programa praneša, jog yra klaida.
  - Jeigu vartotojas įveda per daug integerių, programa perpildymą tiesiog ignoruoja.
- Į ekraną vartotojui yra išvedamas jo įvestas vektorius `Arrays.toString` formatu.
- Vartotojas yra paprašomas įvesti klaidos tikimybę, ji turi būti `0` <= p_e <= 1 ribose, jeigu yra įrašomas realusis skaičius, jis turi būti atskirtas kableliu (`,`).
- Programa iškviečia funkciją `encodeVector`, kuri kaip argumentus paima vektorių, kurį reikia užkoduoti ir Rydo-Miulerio generuojančią matricą. Funkcija grąžina užkoduotą vektorių, kuri turi tokį patį ilgį kaip ir generuojanti matrica.
- Programa iškviečia funkciją `transmitVector`, kuri kaip argumentus paima užkoduotą vektorių ir p_e klaidos tikimybę, o grąžins per nepatikimą kanalą praleistą vektorių.
  - Naudojame `random.nextDouble()`, kad sugeneruotume atsitiktinį skaičių nuo `0` iki 1 (su šiuo metodu sugeneruotas skaičius yra tarp `0.0` ir 1.`0`). Jei šis skaičius yra mažesnis už klaidos tikimybę `pe`, tai vadinasi, kad bitas bus apverstas.
- Programa iškviečia funkciją `detectErrors`, kuri kaip argumentus paima užkoduotą vektorių ir iš nepatikimo kanalo išėjusį vektorių. Funkcija negrąžina jokios reikšmės, tačiau ji išveda į ekraną klaidų skaičių ir klaidų pozicijas.
- Vartotojui yra leidžiama redaguoti iš nepatikimo kanalo išėjusį vektorių, jeigu jis to nori, programa prašo įvesti `taip` arba `ne`.
  - Jeigu vartotojas pasirenka `taip`, vektoriaus įvedimas vyksta panašiai kaip ir informacijos vektoriaus:

    - Vektoriaus skaitymas yra daromas kas vieną integerį, kad vartotojui būtų aiškiau.
    - Bet vartotojas gali ir iš karto vesti pilną savo vektorių atskirdamas tarpais, programa automatiškai įrašys į tinkamas indekso vietas.
    - Jeigu vartotojo įvestis nėra `0` arba 1, programa praneša, jog yra klaida.
    - Jeigu vartotojas įveda per daug integerių, programa perpildymą tiesiog ignoruoja.
- Jeigu vartotojas redagavo vektorių, tai programa į ekraną išveda naują vartotojo paredaguotą vektorių ir programa dar kartą patikrinama klaidų kiekį ir klaidų vietas, bet šį kartą lyginant užkoduotą vektorių ir vartotojo įvesta vektorių.
- Programa iškviečia funkciją `decodeVector`, kuri kaip argumentus paima iš kanalo išėjusį vektorių (arba vartotojo paredaguotą vektorių) ir parametrą `m`, o grąžina dekoduotą vektorių.
    - Ši funkcija dekoduoja gautą vektorių dauginant vektorių su Hadamardo matrica.
    - Vektoriaus reikšmės yra pekeičiamos iš `0` į -1.
    - Iškviečiame funkciją `generateHadamardMatrix`, kuri generuoja Hadamardo matricą ir kaip parametrą paima parametrą `m`, o grąžina Hadamardo matricą.
    - Tuomet mes kviečiame funkciją `multiplyWithHadamard`, kuri kaip parametrus paima jau paredaguotą vektorių (iš `0` į -1) ir Hadamardo matricą, o grąžina naują vektorių po sandaugos su Hadamardo matrica.
    - Toliau kviečiame funkciją `findMaxIndex`, kuri suranda didžiausios absoliučios reikšmės indeksą vektoriuje, o kaip parametrą paima vektorių po sandaugos su Hadamardo matrica.
    - Galų gale iškviečiame funkciją `decodeMessageWithErrorCorrection`, kuri atkuria dekoduotą pranešimą iš didžiausios reikšmės indekso, kaip parametrą paima didžiausios reikšmės indeksą, parametrą `m`, didžiausios reikšmės reikšmę, o grąžina dekoduotą pranešimą kaip bitų masyvą.
- Išspausdiname į ekraną dekoduotą vektorių.

Antras scenarijus (užrašyti tekstą)

- Vartotojas yra paprašomas įvesti kodo parametrą `m` Rydo-Miulerio kodui (1, m).
- Yra iškviečiama funkcija `generateReedMullerMatrix`, kuri paima vartotojo įvestą parametrą `m`, grąžina dvejetainę matricą.
  - Pirmoji eilutė pilna vienetų, kitos eilutės atitinka binary derinius.
- Išspausdiname į ekraną Rydo-Miulerio generuojančią matricą.
- Vartotojas yra paprašomas įvesti tekstą (gali įvesti kelias eilutes). Kad pabaigti rašyti vartotojas turi naujoje eilutėje parašyti `exit`.
- Programa iškviečia funkciją `textToBinary`, kuri kaip argumentą paima vartotojo įvestą tekstą, o grąžina binary stringą.
- Programa iškviečia funkciją `splitIntoVectors`, kuri kaip argumentą paima binary stringą ir parametrą `m`, o grąžina binary stringą suskaidytą į vektorius ilgio 2^m
  - Jeigu vektorius trumpesnis tai užpildome trūkstamus bitus nuliais.
- Iteruojame per suskaidytus vektorius ir užkoduojame juos po vieną vis iškviesdami `encodeVector` funkciją, kuri kaip argumentus paima suskaidytą binary tekstą ir Rydo-Miulerio generuojančią matrica. Funkcija grąžina kelis užkoduotus vektorius.
- Vartotojas yra paprašomas įvesti klaidos tikimybę, ji turi būti `0` <= p_e <= 1 ribose, jeigu yra įrašomas realusis skaičius, jis turi būti atskirtas kableliu (`,`).
- Programa iteruoja per užkoduotus vektorius ir mes vis perduodame juos į kanalą po vieną. Vis iškviečiame funkciją `transmitVector`, kuri kaip argumentus paima užkoduotą vektorių ir klaidos tikimybę. Funkcija grąžina per nepatikimą kanalą praleistus vektorius.
  - Naudojame `random.nextDouble()`, kad sugeneruotume atsitiktinį skaičių nuo `0` iki 1 (su šiuo metodu sugeneruotas skaičius yra tarp `0.0` ir 1.`0`). Jei šis skaičius yra mažesnis už klaidos tikimybę `pe`, tai vadinasi, kad bitas bus apverstas.
- Programa iteruoja per iš kanalo išėjusius vektorius ir dekoduoja juos po vieną vis iškviesdama funkciją `decodeVector`, kuri kaip argumentus ima per nepatikimą kanalą praleistus vektorius ir parametrą `m`, o grąžina dekoduotą vektorių.
  - Ši funkcija dekoduoja gautą vektorių dauginant vektorių su Hadamardo matrica.
  - Vektoriaus reikšmės yra pekeičiamos iš `0` į -1.
  - Iškviečiame funkciją `generateHadamardMatrix`, kuri generuoja Hadamardo matricą ir kaip parametrą paima parametrą `m`, o grąžina Hadamardo matricą.
  - Tuomet mes kviečiame funkciją `multiplyWithHadamard`, kuri kaip parametrus paima jau paredaguotą vektorių (iš `0` į -1) ir Hadamardo matricą, o grąžina naują vektorių po sandaugos su Hadamardo matrica.
  - Toliau kviečiame funkciją `findMaxIndex`, kuri suranda didžiausios absoliučios reikšmės indeksą vektoriuje, o kaip parametrą paima vektorių po sandaugos su Hadamardo matrica.
  - Galų gale iškviečiame funkciją `decodeMessageWithErrorCorrection`, kuri atkuria dekoduotą pranešimą iš didžiausios reikšmės indekso, kaip parametrą paima

didžiausios reikšmės indeksą, parametrą `m`, didžiausios reikšmės reikšmę, o grąžina dekoduotą pranešimą kaip bitų masyvą.

- Programa išspausdina į ekraną palyginimui pradinį tekstą, atkurtą tekstą (siųstą neužkoduotą pro kanalą) ir atkurtą tekstą (siųstą užkoduotą pro kanalą).

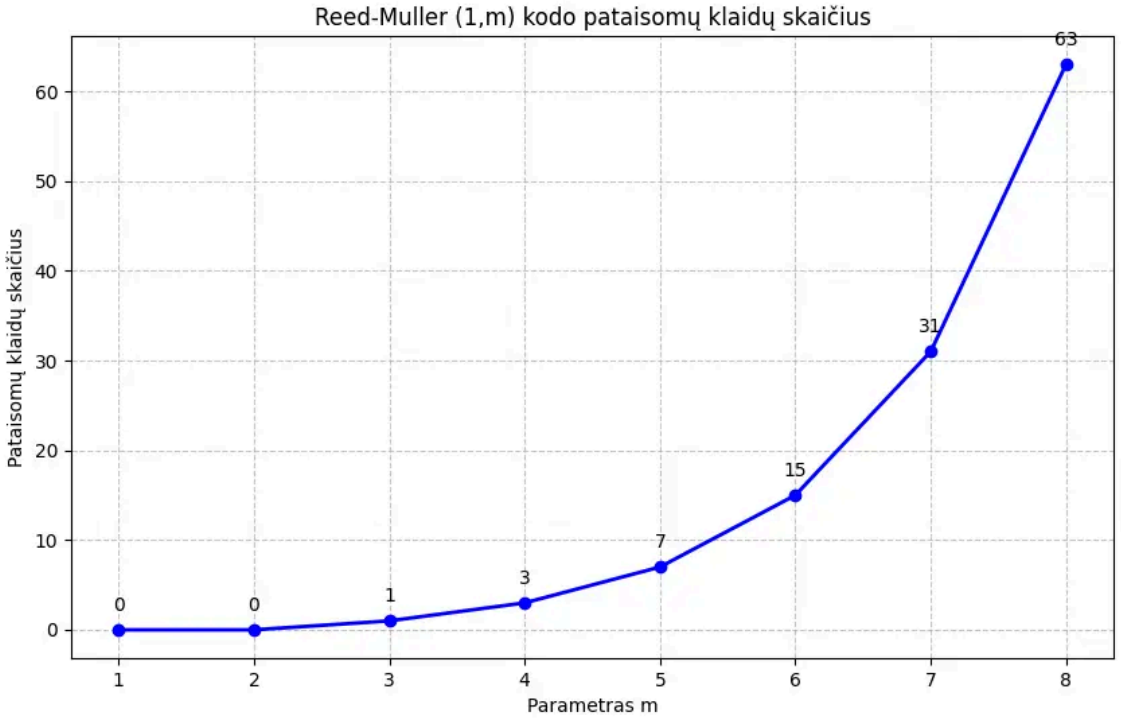
Trečias scenarijus (nurodyti paveiksliuką)

- Vartotojas yra paprašomas įvesti kodo parametrą `m` Rydo-Miulerio kodui (1, m).
- Yra iškviečiama funkcija `generateReedMullerMatrix`, kuri paima vartotojo įvestą parametrą `m`, grąžina dvejetainę matricą.
  - Pirmoji eilutė pilna vienetų, kitos eilutės atitinka binary derinius.
- Išspausdiname į ekraną Rydo-Miulerio generuojančią matricą.
- Vartotojas yra paprašomas pasirinkti `.bmp` paveiksliuką ir jam yra atidaromas failo pasirinkimo langas.
- Konvertuojame paveikslėlį į binary formatą iškviesdami funkciją `convertToBinaryWithRGB`, kuri kaip argumentą paima į bufferį nuskaitytą failą.
  - Konvertuojame kiekvieną kanalą į 8-bitų binary formatą.
- Parodome vartotojui jo paveiksliuką.
- Programa iškviečia funkciją `splitIntoVectors`, kuri kaip argumentą paima binary image stringą ir parametrą `m`, o grąžina binary stringą suskaidytą į vektorius ilgio 2^m.
  - Jeigu vektorius trumpesnis tai užpildome trūkstamus bitus nuliais.
- Iteruojame per suskaidytus vektorius ir užkoduojame juos po vieną vis iškviesdami `encodeVector` funkciją, kuri kaip argumentus paima suskaidytą binary image ir Rydo-Miulerio generuojančią matrica. Funkcija grąžina kelis užkoduotus vektorius..
- Vartotojas yra paprašomas įvesti klaidos tikimybę, ji turi būti `0` <= p_e <= 1 ribose, jeigu yra įrašomas realusis skaičius, jis turi būti atskirtas kableliu (`,`).
- Programa iteruoja per užkoduotus vektorius ir mes vis perduodame juos į kanalą po vieną. Vis iškviečiame funkciją `transmitVector`, kuri kaip argumentus paima užkoduotą vektorių ir klaidos tikimybę. Funkcija grąžina per nepatikimą kanalą praleistus vektorius.
  - Naudojame `random.nextDouble()`, kad sugeneruotume atsitiktinį skaičių nuo `0` iki 1 (su šiuo metodu sugeneruotas skaičius yra tarp `0.0` ir 1.0). Jei šis skaičius yra mažesnis už klaidos tikimybę `pe`, tai vadinasi, kad bitas bus apverstas.
- Programa iteruoja per iš kanalo išėjusius vektorius ir dekoduoja juos po vieną vis iškviesdama funkciją `decodeVector`, kuri kaip argumentus ima per nepatikimą kanalą praleistus vektorius ir parametrą `m`, o grąžina dekoduotą vektorių.
  - Ši funkcija dekoduoja gautą vektorių dauginant vektorių su Hadamardo matrica.
  - Vektoriaus reikšmės yra pekeičiamos iš `0` į -1.
  - Iškviečiame funkciją `generateHadamardMatrix`, kuri generuoja Hadamardo matricą ir kaip parametrą paima parametrą `m`, o grąžina Hadamardo matricą.
  - Tuomet mes kviečiame funkciją `multiplyWithHadamard`, kuri kaip parametrus paima jau paredaguotą vektorių (iš `0` į -1) ir Hadamardo matricą, o grąžina naują vektorių po sandaugos su Hadamardo matrica.
  - Toliau kviečiame funkciją `findMaxIndex`, kuri suranda didžiausios absoliučios reikšmės indeksą vektoriuje, o kaip parametrą paima vektorių po sandaugos su Hadamardo matrica.

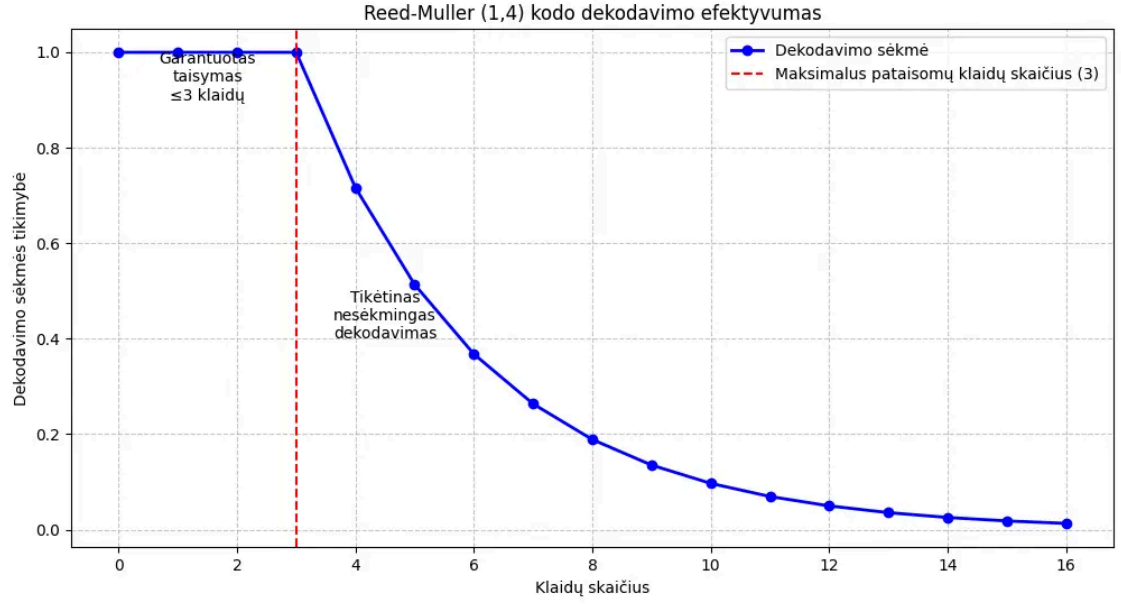
    - Galų gale iškviečiame funkciją `decodeMessageWithErrorCorrection`, kuri atkuria dekoduotą pranešimą iš didžiausios reikšmės indekso, kaip parametrą paima didžiausios reikšmės indeksą, parametrą `m`, didžiausios reikšmės reikšmę, o grąžina dekoduotą pranešimą kaip bitų masyvą.
- Programa iškviečia funkciją `createImageFromBinaryRGB`, kuri kaip argumentus ima dekoduotą vientisą stringą, paveikslėlio ilgį ir paveikslėlio aukštį, o grąžina jau sukurtą paveikslėlį.
    - Išimame 8-bitų binary kodą kiekvienam RGB kanalui
- Programa atvaizduoja dekoduotą paveikslėlį (kuris buvo neužkoduotas) ir dekoduotą paveikslėlį (kuris buvo užkoduotas).

Atlikti eksperimentai

Kodo pataisomų klaidų skaičius

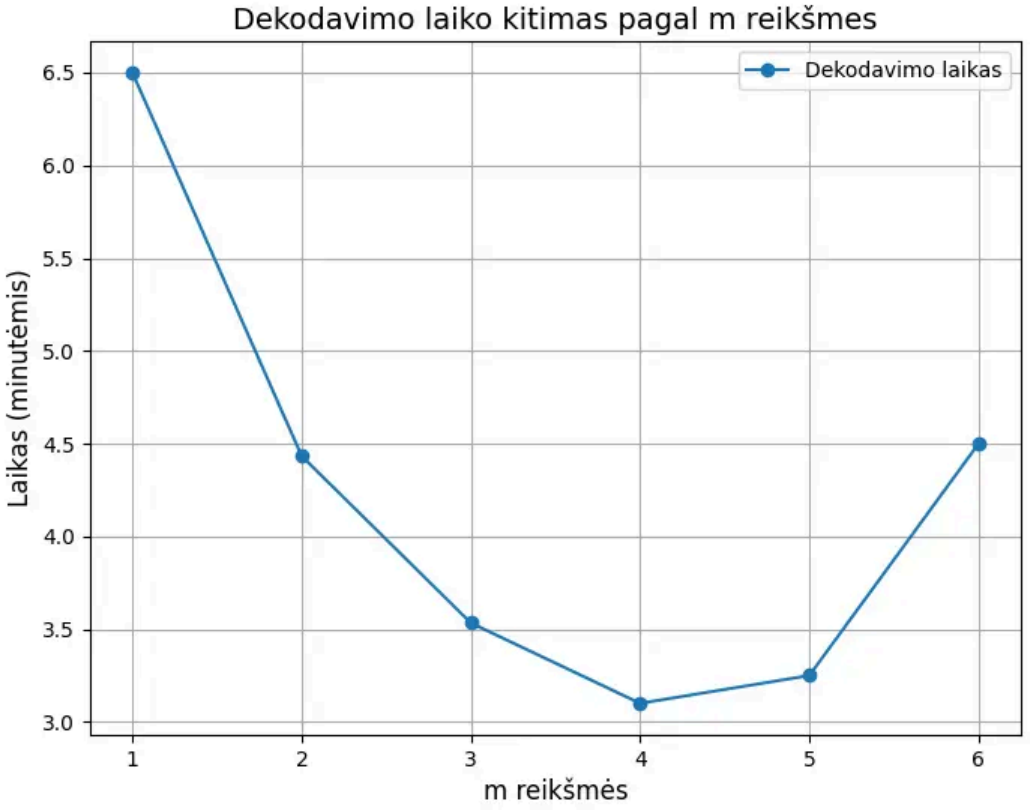


Kodo dekodavimo efektyvumas



Kodo paveikslėlio pilnas apdorojimo efektyvumas

Nurodytas laikas yra pilnas, tai reiškia nuo programos paleidimo iki paveikslėlio parodymo ekrane. Programa visus paleidimus buvo paleista naudojant default paleidimą: `java -jar KTK.jar` (išskyrus kai m=6, tada buvo paleista naudojant `java -Xms2g -Xmx4g -jar KTK.jar`) ir visada buvo nustatoma klaidos tikimybė 0,1.
Norint, kad programa dekoduotų paveikslėlį kai m>5 gali reikti paleisti programą naudojant: `java -Xms2g -Xmx4g -jar KTK.jar`
Paaiškinimas:
`-Xms2g`: Nustato pradinį heap'o dydį iki 2 GB. Tai užtikrina, kad Java Virtuali Mašina pradės darbą su 2 GB heap'u.
`-Xmx4g`: Nustato maksimalų heap'o dydį iki 4 GB. Tai leidžia Java Virtualiai Mašinai išplėsti heap'ą iki 4 GB, jei to prireikia.



Naudota literatūra

- Užduoties aptarimo vaizdo įrašas
- HLL91, §3.8–3.9, p. 89–95