Vilniaus universitetas

Matematikos ir informatikos fakultetas

Vytautas Lėveris

Darbo ataskaita

Atlikta užduotis: A7 (Reed Muller kodas bei Majority Logic Decoding dekodavimo algoritmas);

Realizuotos dalys: visos;

Naudotos bibliotekos: tik JAVA kalbos standartinės priemonės;

Laiko sanaudos:

Etapas	Trukmė
Literatūros skaitymas	2 dienos
Kodo veikimo aiškinimasis	2 dienos
Projektavimas	3 dienos
Programavimas	18 dienų
Klaidų ieškojimas ir taisymas	1 diena
Ataskaitos ruošimas	1 valanda
Iš viso	26 dienos 1 valanda

Kaip paleisti programą?

Išskleidus archyvą, reiktų atsidaryti aplanką "programa". Jame bus du failai: Programa.jar ir paleisti.bat.

Iš pradžių galima mėginti spragtelti iškart ant JAR failo, galbūt programa atsidarys (priklausomai nuo Java instaliacijos nustatymų).

Jei ne, tuomet reiktų paleisti BAT skriptą ir programa turi pasirodyti.

Jei vis dėlto nepasileidžia, galima mėginti atsidaryti išeities tekstus. Programuota buvo naudojant Eclipse aplinką, todėl Eclipse reikalingi failai yra įtraukti, tačiau pakankamai lengvai turėtų būti atidaroma kitų IDE.

Programa

Programos kodo dalį sudaro du paketai: programa ir programa.uzduotis. Pirmasis atsakingas už grafinės vartotojo sąsajos funkcionalumą. Antrajame glūdi visa užduoties logika, kodo apibrėžtis, kodavimo/dekodavimo/kanalo klasės ir pagalbinės vektorių skaidymo/apjungimo bei matematinės klasės.

Paketo programa struktūra

Filtras.java - užtikrina, jog tik leidžiami simboliai būtų naudojami tekstiniuose laukuose bei riboja lauko ilgį;

Langas.java - visas grafinės sąsajos lango kodas, atsakingas už mygtukų, tekstinių laukų konfiguravimą, rodymą ir apskritai už visą UI funkcionavimą;

Pagrindinis.java - paleidžiamasis programos failas.

Paketo programa.uzduotis struktūra

Dekodavimas.java - klasė, atsakinga už dekodavimą;

Kanalas.java - simuliuoja kanalą, kuriuo siunčiami žodžiai gali būti iškraipomi pagal nustatytą klaidos tikimybę;

Kodas.java - klasė, apibrėžianti kodo struktūrą, pagal r ir m parametrus sudaranti generuojančią matricą;

Kodavimas.java - klasė, koduojanti informacinius vektorius pagal nustatomą kodą;

Matematika.java - matematinių funkcijų klasė, skirta darbui su vektoriais (jų sandauga, sudėtis moduliu 2, skaliarinė sandauga, n vektorių sandauga ir pan.), naudojami kombinatorikos elementai (nepilnas faktorialas, derinių skaičiaus ir pačių derinių radimas), darbui su dešimtainiais/dvejetainiais skaičiais.

Pagalbinis.java - pagalbinė klasė, skirta teksto bei paveiksliuko skaidymui nurodyto ilgio vektoriais ir atvirkščiai.

Vartotojo sąsajos aprašymas

Pasileidus programą, viršutinėje eilutėje matomi kodo parametrų r ir m laukeliai. Juose, tik paleidus programą, bus nurodytos numatytosios reikšmės: r=1 ir m=3. Į šiuos laukelius galima įvesti tik skaičius nuo 0 iki 9 ir ne daugiau kaip 3 simbolius. Taip pat viršutinėje eilutėje galima nurodyti/keisti kanalo tikimybę, kurios reikšmė iš pradžių būna 0.0. Paspaudus mygtuką "Nustatyti", jei parametrai ir tikimybė nurodyti teisingai , kodas yra atnaujinamas. Priešingu atveju, bus informuojama apie įvykusią klaidą. Žemiau galima pasirinkti vieną iš trijų scenarijų. Pasirinkus pirmąjį, nurodome atitinkamo ilgio vektorių, užkoduojame. Užkoduoto vektoriaus laukelyje matome, koks buvo gautas užkoduotas vektorius (jo turinio keisti negalime). Paspaudžiame "Siųsti". Laukelyje "Gautas vektorius:" matome, koks vektorius buvo išsiųstas iš kanalo. Klaidų laukelyje matome, kiek ir kuriose pozicijose buvo padaryta klaidų. Gauto vektoriaus turinį galime keisti ir nurodyti savo klaidas, po to reikia spustelti mygtuką "Keisti", kad pakeitimai paveiktų originalų išsiųstą vektorių. Išsiųstą ir pamodifikuotą vektorių galime bandyti dekoduoti ir dekodavimo laukelyje matyti rezultatus.

2 scenarijuje įvedame kokį nors tekstą, spaudžiame koduoti ir matome, kaip atrodo tekstas, siųstas nepatikimu kanalu ir jo nekoduojant, ir užkoduojant.

3 scenarijaus lange pasirenkame BMP tipo paveiksliuką. Spaudžiame mygtuką "Piešti" ir, panašiai kaip ir antrame scenarijuje, matome skirtumą tarp pikselių masyvo siuntimo kanalu koduojant/nekoduojant.

Padaryti programiniai sprendimai

Skaidymo vektoriais strategija. Eilutės tipo tekstas į vektorius skaidomas tokiu būdu: kiekvienas eilutės simbolis (char tipo) yra verčiamas 16 bitų seka (Java kalboje char tipas atitinka int, tačiau jo maksimalus dydis yra perpus mažesnis jei int - 16 bitų arba 2 baitai). Taip suskaidomi visi simboliai ir iš jų formuojami vektoriai. Jei po paskutinio simbolio dar lieka laisvų vektoriaus bitų, yra prirašomi nuliai. Įsiminti, kiek pabaigoje buvo prirašyta nulių, nereikia, nes atstatymo metu iš vektorių masyvo ilgio yra paskaičiuojama, kiek telpa pilnų 16 bitų simbolių, ir traktuojama, jog likę simboliai yra nenaudingi. Panašiai elgiamasi ir su paveiksliukais, tačiau paveiksliuko kiekvienas pikselis yra verčiama 32 bitų seka. BufferedImage duomenų tipas pikselius saugo keturių baitų žodžiais - ARGB principu. Baitai nurodo alpha permatomumą, raudoną, žalią bei mėlyną spalvas.

Generuojančios matricos pradinių m+1 eilučių sudarymo būdai. Iš viso pateikiami 3 generuojančios matricos pradžios sudarymo būdai. Pirmasis, neužkomentuotas, tiesiogiai perbėgdamas per erdvės vektorius nustato, kurie indeksai bus nuliai, kurie - vienetai. Antrasis (pirmasis neužkomentuotas) būdas panašus, tačiau nuoseklesnis. Remiamasi V. Stakėno išdėstyta generuojančios matricos sudarymo teorija. Formuojama h aibė, o po to pagal eilutės ir stulpelio

indeksus tikrinama, ar toje aibėje egzistuoja atitinkamas elementas. Trečiasis būdas yra įdomesnis, tačiau, kiek bandžiau, visada veikė. Pavyzdžiui, tarkime, r=1, m=4. Sudarysime pirmas 5 generuojančios matricos eilutes:

1111 1111 1111 1111

1111 1111 0000 0000

1111 0000 1111 0000

1100 1100 1100 1100

1010 1010 1010 1010

Matome, kad iš pradžių eina visi vienetai, po to vienetai ir nuliai iš eilės eina pusėje pozicijų ir panašiai. Taigi, mano manymu, algoritmas visai tinkamas ir gan greitas.

Atlikti eksperimentai

Dekodavimo algoritmo efektyvumas keičiant kodo parametrus

Tegul iš pradžių r=0, m=5, kanalo tikimybė=0,1. Paprastumo dėlei laikykime, kad informacijos vektorius yra vienetinis. Atlikau eksperimentą keisdamas r ir m reikšmes tokiu būdu: iš pradžių keičiu r reikšmę, kol ji lygi m. Tuomet m sumažinu vienetu, r prilyginu 0 ir vėl procedūrą kartoju, kol r susilygina su m. Eksperimentą baigiu, kai parametrų r ir m reikšmės tampa 0. Stebiu, ar po dekodavimo informacijos vektorius sutampa su dekoduotu, o jei ne, keliose pozicijose nesutampa.

r	m	Klaidų skaičius persiųstame vektoriuje	Klaidų skaičius dekoduotame vektoriuje
0	5	6	0
1	5	3	0
2	5	3	0
3	5	2	8
4	5	3	10
5	5	3	12
0	4	1	0
1	4	3	0
2	4	3	4
3	4	3	8
4	4	1	2
0	3	0	0
1	3	0	0

2	3	1	7
3	3	0	0
0	2	0	0
1	2	0	0
2	2	0	0
0	1	0	0
1	1	0	0
0	0	0	0

Naudota literatūra

Vilius Stakėnas, "Kodai ir šifrai", Vilnius, TEV, 2007.