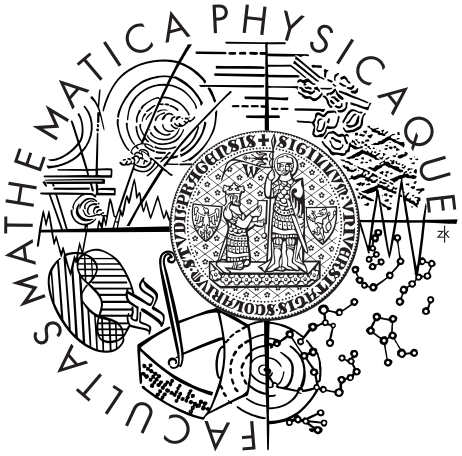
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Jan Vacek

Pokrývací množiny ve steganografii

Katedra algebry

Vedoucí diplomové práce:	doc. Mgr. Štěpán Holub, Ph.D.
Studijní program:	Matematika
Studijní obor:	matematické metody informační bezpečnosti

Praha 2013

Rád bych poděkoval všem, kteří mě při psaní této práce podporovali. Jedná se zejména o členy mé rodiny a vedoucího této práce docenta Štěpána Holuba. Děkuji.

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V Praze dne ..... Podpis autora

Název práce: Pokrývací množiny ve steganografii

Autor: Jan Vacek

Katedra: Katedra algebry

Vedoucí diplomové práce: doc. Mgr. Štěpán Holub, Ph.D., Katedra algebry

Abstrakt: Steganografie je věda zabývající se ukrýváním komunikace.Práce se zaměřuje na nejnovější metody používané v této oblasti. Jedná se zejména o maticové vkládání, které využívá teorii samoopravných kódů, a o součtové a rozdílové pokrývací množiny (SDCS). Pro dosažení lepších výsledků se využívá duhové obarvení síťových grafů. Pomocí této techniky můžeme snížit velikost prováděných změn a vytvořit tak stegosystémy, které jsou obtížněji detekovatelné. U každé metody jsou vyčísleny její vlastnosti a jsou porovnány s vlastnostmi jiných možných technik.

Klíčová slova: steganografie, pokrývací množiny, SDCS, duhové grafy

Title: Covering sets in steganography

Author: Jan Vacek

Department: Department of Algebra

Supervisor: doc. Mgr. Štěpán Holub, Ph.D., Department of Algebra

Abstract: Steganography is a science which is interested in communication hiding. This work is focused on the most recent methods related to this topic. Mainly, it is matrix embedding, which uses coding theory, and sum and difference covering sets (SDCS). Rainbow coloring of grid graphs is used to receive even better results. This technique decrease amplitude of performed changes. It makes stegosystems less likely to be detected. Properties which describe behavior of each stegosystem are included for each technique.

Keywords: steganography, covering sets, SDCS, rainbow graphs

Obsah

Kapitola 1

Úvod

Podobně jako je cílem kryptografie utajování obsahu zprávy, cílem steganografie je skrytí komunikace jako takové. Tato disciplína se snaží utajovanou zprávu vložit do „běžné“ komunikace. Naproti tomu stojí steganoanalýza, která se snaží skrytou zprávu odhalit.

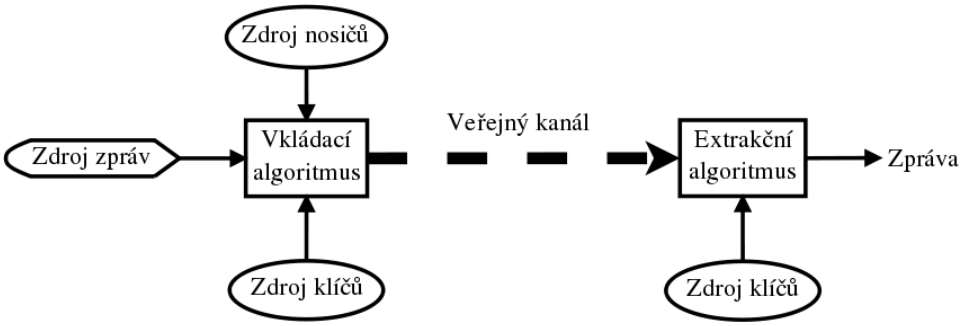
1.1 Vězeňský problém a jeho primitivní řešení

Odesílatel stojí před klasickým vězeňským problémem. Vězni A a B jsou umístěni v oddělených celách. Jejich komunikace je monitorována a cenzurována a oni se potřebují dohodnout tak, aby tomu nerozuměli dozorci. V tomto případě je použití klasické šifry nemožné. Pokud by totiž dozorci zachytili zprávu, které by nerozuměli, neposlali by ji dál. A právě v tomto okamžiku musejí vězňové použít steganografii, aby svou komunikaci ukryli do jiné, zdánlivě neškodné, komunikace.

Klasicky se na ukrývání informace využívá audio, video, formátování, text, obrázek a podobně. Tomu budeme obecně říkat *nosič*. Pro naše účely budeme používat obrázky, protože podle [2] jsou nosiče obrázky ve více než polovině případů. Stejně jako v kryptografii jsou i ve steganografii obě strany domluveny na stejném klíči, pomocí něhož zprávu vkládají a následně na druhé straně extrahují. Jednotlivé prvky obecného steganografického kanálu jsou znázorněny na obrázku 1.1.

Podle [2] můžeme stegosystémy, jak zkráceně označujeme steganografický kanál, rozdělit na základní tři druhy. Jsou to stegosystém pomocí:

volby nosiče: V tomto případě je klíčem vězňů jakási kódová kniha. Bude-li například na poslaném obrázku auto, znamená to „Vše proběhlo bez problémů.“ Toto je sice velmi jednoduchý a snadno použitelný způsob přenosu informace, zároveň se ale jedná o velmi omezený způsob komunikace a navíc je poměrně náročný na celkový objem přenesených dat. Tímto způsobem se proto dále zabývat nebudeme.



Obrázek 1.1: Obecný stegosystém.

tvorby nosiče: Další z možných způsobů je například vytvoření zprávy, kde první písmeno každého slova tvoří ukrývanou zprávu. Klíčem je v tomto případě znalost tohoto kódování. To samozřejmě skýtá mnoho možností. Zároveň se ale klade velký důraz na kreativitu, protože je nutné vyrobit zprávu, která nese požadovanou zprávu a zároveň nevypadá nápadně. Tímto způsobem se proto v této práci také nebudeme zabývat.

modifikace nosiče: Toto je způsob, který se používá nejčastěji a je možné jej automatizovat pomocí počítačového programu. Zároveň se jedná asi o nejkomplexnější případ. Tímto způsobem se budeme zabývat.

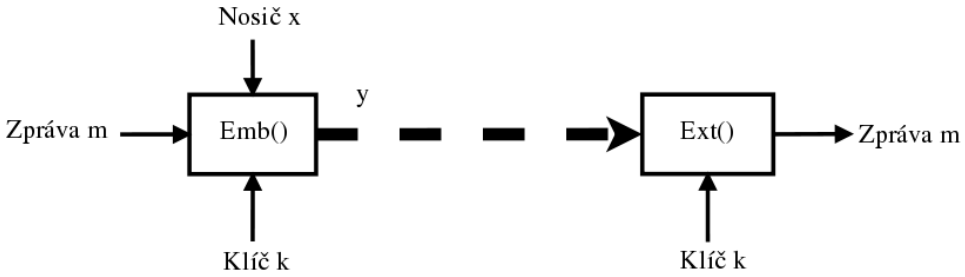
1.2 Používané pojmy a značení

Jak jsme již zmínili v předchozí kapitole, od této chvíle se budeme zaobírat stegosystémy pomocí modifikace nosiče. Pro popis takovýchto systémů se využívá několik pojmů, které se budeme snažit v této kapitole co nejvíce přiblížit.

Jedním ze základních používaných pojmů je všeobecně známá Hammingova vzdálenost. Ta se hojně používá v teorii samoopravných kódů. V této práci ji budeme značit jako d_H. Pro úplnost je definice k nahlédnutí v sekci 1.6.

Ještě před začátkem komunikace se musí obě strany tajně dohodnout na základech protokolu, který budou následně používat. Jedná se zaprvé o typ nosiče, který bude posílanou zprávu maskovat, dále o algoritmy, které budou sloužit pro vkládání a extrahování zprávy, a u některých protokolů i na tajném klíči, pomocí kterého bude možné zprávu vložit a následně ji extrahovat.

Na Obrázku 1.1 je znázorněn obecný diagram komunikace, pomocí níž předá jedna strana té druhé utajenou zprávu. Odesilatel nejprve ze zdroje zpráv vybere zprávu, kterou chce příjemci poslat. Následně si vybere klíč,



Obrázek 1.2: Konkrétní případ stegosystému pomocí modifikace mosiče.

na kterém je domluvený s příjemcem, a nosič, který je nějakým způsobem vhodný. Vše vloží do vkládacího algoritmu, který vyprodukuje pozměněný nosič. Ten pak pošle příjemci nechráněným kanálem. Příjemce vloží přijatou zprávu spolu s klíčem do extrakčního algoritmu a obdrží zprávu.

Odesilatel utajované komunikace stojí před problémem, kde ke zprávě, kterou chce poslat, vybírá vhodný nosič. My se ale zabýváme stegosystémy pomocí modifikace nosičů, a abychom mohli porovnat jejich kvality, budeme k danému nosiči hledat zprávy, které se dají do nosiče různými algoritmy ukrýt. Z toho důvodu bereme nosič jako základní a postupně k němu připojujeme další pojmy. Nutno ovšem poznamenat, že se jednotlivé položky budou vztahovat vždy ke konkrétnímu stegosystému a pro různé stegosystémy se tedy mohou lišit.

Nyní zavedeme značení, která budeme dále používat. Pro lepší představu je vše zaznamenáno na obrázku 1.2.

- Množinu všech nosičů budeme označovat $\mathcal{C}$. Obecně lze uvažovat jako nosič libovolnou posloupnost znaků z nějaké konečné množiny. V této obecnosti by se ale nedalo odlišit, jak moc se od sebe vzájemně jednotlivé znaky liší, protože na obecné množině můžeme definovat pouze Hammingovu vzdálenost. Jak jsme ale již zmínili dříve, nejčastějšími nosiči jsou obrázky, a proto budeme jako nosiče předpokládat posloupnost prvků konečné grupy $\mathbb{Z}_p$, kde p, oproti konvencím, není nutně prvočíslo. V některých případech budeme pro větší názornost mluvit dokonce o pixelech a DCT koeficientech. Podrobnější informace o různých reprezentacích obrázků a způsobech jejich využití budou uvedeny v kapitole 1.4. Nutno poznamenat, že pro některé stegosystémy bude množina používaných nosičů ještě omezena, abychom se vyhnuli problémům při vkládání.

- Množinu všech zpráv, které je možné vložit do nosiče $x \in \mathcal{C}$, budeme označovat $\mathcal{M}(x)$. Je zřejmé, že všechny možné zprávy není možné skrýt do všech možných nosičů, proto je tato množina závislá na x. Někdy ale budeme tuto množinu značit pouze jako $\mathcal{M}$, neboť nemůže dojít

k omylu. Zároveň bude ale tato množina pro každý stegosystém jiná. Bude tedy záležet na efektivitě vkládacího algoritmu. Dále se budeme snažit tuto množinu maximalizovat tím, že budeme hledat nejefektivnější vkládací algoritmus.

- Množinu všech klíčů pro nosič $x \in \mathcal{C}$ budeme označovat jako $\mathcal{K}(x)$. Stejně jako v předchozím případě budeme někdy vynechávat x a budeme množinu značit pouze $\mathcal{K}$. Jako část klíče se typicky používá informace o použitých složkách obrázku. Klíč může být například mapa pixelů, která určuje, jaké obrazové body se k vložení zprávy použijí. Klíče ale mohou mít pro různé stegosystémy různou podobu. U některých stegosystémů se dokonce klíč nevyskytuje vůbec, pokud nepočítáme informaci o používaném algoritmu.

- Jako
$$\mathrm{Emb} : \mathcal{C} \times \mathcal{M} \times \mathcal{K} \to \mathcal{C}$$
budeme označovat funkci, která do nosiče $x \in \mathcal{C}$ vkládá zprávu $m \in \mathcal{M}(x)$ pomocí klíče $k \in \mathcal{K}(x)$. Je zřejmé, že tato funkce musí všechny hodnoty zobrazovat do $\mathcal{C}$, neboť jinak by byla vložená zpráva snadno detekovatelná. Máme tedy
$$\mathrm{Emb}(x, m, k) = y,$$
kde $y \in \mathcal{C}$.

- Jako *stegoobjekt* označujeme každý prvek $y \in \mathcal{C}$, který vznikl vložením nějaké zprávy $m \in \mathcal{M}(x)$ do nosiče $x \in \mathcal{C}$. Jinak řečeno se jedná o nosič z obrazu vkládací funkce, $y \in \mathrm{Im}(\mathrm{Emb})$. Pokud je množina všech stegoobjektů vlastní podmnožinou $\mathcal{C}$, jedná se zřejmě o slabinu systému. Pokud je zobrazení Emb() na, pak je vlastně každý stegoobjekt zároveň nosičem. Pro výstup vkládací funkce ale budeme používat pojem stegoobjekt, aby nedošlo k nedorozumění.

- Dále budeme potřebovat funkci pro extrakci ukryté zprávy. Budeme ji značit
$$\mathrm{Ext} : \mathcal{C} \times \mathcal{K} \to \mathcal{M} \cup \{0\}.$$
Funkce bere jako vstup nosič $y \in \mathcal{C}$ a klíč $k \in \mathcal{K}$ a jako výstup vrátí buď 0, pokud y není stegoobjekt, nebo $m \in \mathcal{M}$ takové, že $\exists x \in \mathcal{C}$ tak, že
$$\mathrm{Emb}(x, m, k) = y,$$
pokud y je stegoobjekt. Je zřejmé, že může existovat více $x \in \mathcal{C}$, pro které je předchozí rovnost splněná. Naopak požadujeme, aby existovalo pouze jediné $m \in \mathcal{M}$ splňující předchozí podmínku.

Jinými slovy požadujeme, aby pro nosič $x \in \mathcal{C}$, zprávu $m \in \mathcal{M}(x)$ a klíč $k \in \mathcal{K}(x)$ platilo

$$\mathrm{Ext}(\mathrm{Emb}(x, m, k), k) = m,$$

jak je zřejmé z obrázku 1.2, kde je navíc vzniklý stegoobjekt označen jako

$$y = \mathrm{Emb}(x, m, k).$$

Jak již bylo řečeno, nosič bereme jako posloupnost znaků a jako klíč se často vyskytuje mapa, která určuje, jaké prvky nosiče se budou pro vkládání zprávy používat. Abychom se ale tímto v dalších částech nemuseli zaobírat, budeme jako nosič brát pouze ty prvky, které budeme k vkládání zprávy používat. Nijak tím neubereme na obecnosti, pouze zjednodušíme zápis a budeme se moci zaměřit na matematickou stránku vkládání, místo abychom dlouze popisovali přípravnou fázi. Můžeme tedy poznamenat, že nyní již ke vkládání používáme všechny prvky nosiče.

1.3 Základní steganografické veličiny

Pro porovnávání jednotlivých stegosystémů potřebujeme definovat veličiny, které nám pomohou popsat jednotlivé systémy a pomocí nichž ohodnotíme jejich kvalitu.

Definice 1.3.1 (Kapacita nosiče)**.** Kapacitu nosiče x*, kde* $x \in \mathcal{C}$ *definujeme jako*

$$\kappa(x) = \log_2 |\mathcal{M}(x)|.$$

Je zřejmé, že předchozí definice nemá obecně význam a že má smysl až v kontextu zadaného vkládacího algoritmu. Stejně jako v předchozím případě se v této sekci definované pojmy vztahují vždy ke konkrétnímu stegosystému, proto nám takovéto definici nic nebrání. Každý stegosystém má totiž jednoznačně popsaný vkládací algoritmus, a tudíž i množinu všech možných zpráv. Definice je tedy korektní. Stejně tak tomu bude i u dalších definic.

Lze snadno nahlédnout, že v kapacitě nosiče nezáleží na velikosti nosiče a pro větší nosiče vycházejí tedy lepší hodnoty. Abychom ale mohli hodnotit kapacitu nosiče vzhledem k počtu jeho prvků, zavedeme další veličinu. Tím budeme moci objektivně porovnat kapacitu pro dva různě dlouhé nosiče. Následující veličina bude brát v úvahu, jaká je délka nosiče, neboli kolik prvků jednotlivé nosiče mají.

Definice 1.3.2 (Relativní kapacita nosiče)**.** *Pro* $x \in \mathcal{C}$ *definujeme* relativní kapacitu nosiče x *jako*

$$\alpha(x) = \frac{\log_2 |\mathcal{M}(x)|}{n} = \frac{\kappa(x)}{n},$$

kde n *je počet prvků nosiče.*

V předchozí definici se vyskytuje n, které značí počet prvků nosiče. Jak jsme již poznamenali, jako nosič bereme již vybranou část prvků, do kterých se vkládá zpráva. Tím se nám zjednodušuje určování n, protože se vždy rovná délce nosiče.

V praxi bývá u rastrových formátů obrázku hodnota n odvozena z počtu použitých barevných kanálů a počtu použitých pixelů. U obrázků formátu JPEG pak n představuje počet nenulových DCT koeficientů (koeficientů diskrétní kosinové transformace). Více podrobností o rozdílu vkládání do různých formátů obrázku bude uvedeno v sekci 1.4.

Vzhledem k tomu, že relativní kapacita nosiče záleží na zvoleném nosiči, může se zdát jako problém, použít ji k popsání vlastností zvoleného stegosystému bez závislosti na vybraném nosiči. Můžeme proto definovat další veličinu, která bude brát v úvahu všechny možné nosiče, a podle jejich pravděpodobnostního rozložení počítat střední hodnotu jejich relativních kapacit.

Definice 1.3.3 (Relativní kapacita)**.** Relativní kapacitu *definujeme jako*

$$\alpha = \mathrm{E}\left(\alpha(x)\right),$$

kde E *je střední hodnota přes všechny nosiče* $x \in \mathcal{C}$ *vzhledem k jejich pravděpodobnostnímu rozložení.*

Jak ale uvidíme dále, předchozí definice bude v mnoha případech dávat stejný výsledek jako definice 1.3.2, protože relativní kapacity jednotlivých nosičů budou mít v případech, které budeme zkoumat, všechny stejnou hodnotu.

Právě jsme definovali veličiny, které nám pomohou porovnat, jaký stegosystém dokáže do nosiče vložit delší zprávu poměrně k velikosti tohoto nosiče. Zároveň ale chceme, aby bylo obtížné vloženou zprávu detekovat. Proto požadujeme, aby provedené změny byly co možná nejmenší a co možná nejméně viditelné. Zavádíme tedy další veličiny, které nám provedené změny pomohou ohodnotit.

Definice 1.3.4 (Distorze)**.** *Jako* distorzi $x, y \in \mathcal{C}$*;* $x = (x_1, x_2, \ldots, x_n)$ *a* $y = (y_1, y_2, \ldots, y_n)$ *budeme označovat*

$$\mathrm{d}(x, y) = \sum_{i=1}^{n} |x_i - y_i|.$$

Distorze se používá pro x, y, kde x je nosič a y je stegoobjekt z něj vzniklý. Podle definice je zřejmé, že se měří celková odchylka, která se nijak nenormuje. Jedná se vlastně o L_1 metriku mezi vektory x a y. Je zřejmé, že by bylo možné použít i jiné metriky. Obecně se podle [2] distorze x a y definuje jako

$$\mathrm{d}_\gamma(x, y) = \sum_{i=1}^{n} |x_i - y_i|^\gamma$$

a námi definovaná distorze je tedy rovna případu $\gamma = 1$. V některých případech se jako distorze používá

$$\mathrm{d}(x, y) = \mathrm{d}_{\mathrm{H}}(x, y),$$

kde $\mathrm{d}_{\mathrm{H}}(x, y)$ značí Hammingovu vzdálenost. Takto definovaná distorze pak jen počítá počet míst, ve kterých se objekty liší. V literatuře je zvykem tuto speciální distorzi značit $\vartheta(x, y)$. My ji budeme označovat stejně, aby nedošlo k nedorozumění.

Pokud někde neuvedeme jinak, budeme jako $\mathrm{d}(x, y)$ uvažovat $\mathrm{d}_1(x, y)$. Zbývá poznamenat, že v případě, že maximální možná změna $|x_i - y_i| = 1$, jsou všechny popsané druhy distorze shodné.

V definici distorze samozřejmě opět záleží na délce nosiče. Čím budeme mít delší nosič, tím bude mít pravděpodobně distorze větší hodnotu. Můžeme samozřejmě opět definovat normovanou verzi, ale vzhledem k tomu, že takto definovanou obměnu nebudeme potřebovat, bylo by zbytečné definovat tento mezikrok samostatně.

Potřebujeme ale veličinu, která by navíc ještě popsala chování celého stegosystému, nehledě na vybraný nosič, zprávu a klíč.

Definice 1.3.5 (Míra změny)**.** Míru změny *značíme* ρ *a definujeme jako*

$$\rho = \mathrm{E}\left(\frac{\vartheta(x, y)}{n}\right),$$

kde E *značí střední hodnotu přes všechny dvojice* (x, y)*, kde* x *je původní nosič,* $x = (x_1, x_2, \ldots, x_n) \in \mathcal{C}$*, a* $y = (y_1, y_2, \ldots, y_n) \in \mathcal{C}$ *je stegoobjekt vzniklý z* x*.*

Všimněme si, že v poslední definici procházíme všechny nosiče $x \in \mathcal{C}$, všechny zprávy $m \in \mathcal{M}$ a všechny klíče $k \in \mathcal{K}$ tak, že platí rovnost $y = \mathrm{Emb}(x, m, k)$.

V definici míry změny se v čitateli objevuje $\vartheta(x, y)$ původního nosiče a stegoobjektu. Nezáleží tedy na velikosti změny, kterou jsme provedli. Toto je tedy asi jediný případ, kdy bychom mohli porovnávat účinnost stegosystémů pro nosiče v plné obecnosti, jak byli definovány na straně 5. Čitatel je pak pouze normován pomocí počtu prvků nosiče. Míra změny tedy ukazuje, jak moc se původní nosič po vložení zprávy změnil. Pokud zanedbáme způsob, jakým byl nosič změněn, pak s rostoucí mírou změny roste riziko odhalení. Jak již bylo řečeno, míra změny nám neřekne, o kolik se jednotlivé nosiče na daných pozicích liší. Poznáme tedy pouze na kolika místech byl nosič upraven. Abychom zjistili, o kolik se nosič na jednotlivých místech změnil, můžeme využít definici distorze 1.3.4.

K následující definici využijeme již zmiňovanou distorzi. Potřebujeme totiž veličinu, která vezme v úvahu nejen délku zprávy, kterou je možné do nosiče vložit, a počet změn, ale také velikost změn, které se na nosič provedly.

Definice 1.3.6 (Efektivita)**.** Efektivita e *je definovaná jako*

$$\mathrm{e} = \frac{\mathrm{E}_x(\log_2 |\mathcal{M}(x)|)}{\mathrm{E}_{x,m}(\mathrm{d}(x,y))},$$

kde E_x *značí střední hodnotu přes rozdělení všech nosičů* $x \in \mathcal{C}$ *a* $\mathrm{E}_{x,m}$ *značí střední hodnotu přes rozdělení všech dvojic* (x,m)*, kde* $x \in \mathcal{C}$ *a* $m \in \mathcal{M}$ *tak, že* $\mathrm{Emb}(x,m) = y$*.*

Pro jednodušší zápis budeme většinou psát

$$\mathrm{e} = \frac{\mathrm{E}(\log_2 |\mathcal{M}(x)|)}{\mathrm{E}(\mathrm{d}(x,y))},$$

přičemž je nutno upozornit, že střední hodnota jmenovatele se nebere přes y, ale přes dvojici (x,m).

Přecházení mezi dvěma různými rozděleními může být matoucí, může ale předpokládat, že se střední hodnota bere v obou případech přes dvojici (x,m). Význam se tím nezmění, neboť čitatel nezávisí na m. Hodnota pravděpodobnosti se tedy u čitatele v závislosti na m nemění a součet všech pravděpodobností pro jedno konstantní x je vždy 1.

Podle původního článku [13] a podle [4] je efektivita očekávaný počet bitů náhodné zprávy na jednu změnu v nosiči, což naše definice splňuje.

Dále můžeme poznamenat, že efektivita je vlastně podíl střední hodnoty kapacity nosiče a střední hodnoty distorze.

$$\frac{\mathrm{E}(\kappa(x))}{\mathrm{E}(\mathrm{d}(x,y))} = \frac{\mathrm{E}(\log_2 |\mathcal{M}(x)|)}{\mathrm{E}(\mathrm{d}(x,y))} = \mathrm{e}$$

Je zřejmé, že efektivita záleží na uvažované definici distorze. V případě, že se jako distorze bere $\vartheta(x,y)$, nezávisí efektivita na velikosti provedených změn. Ale jak již bylo řečeno výše, budeme předpokládat, že distorze je rovna $\mathrm{d}_1(x,y)$, pokud neřekneme jinak.

Naposledy definovaná efektivita je podle [2] spolu s relativní kapacitou asi tím nejdůležitějším ukazatelem optimality stegosystému. Jako další faktor se ještě sleduje míra změny. Budeme se tedy snažit tyto vlastnosti co nejvíce vylepšit.

1.4 Různé reprezentace obrazu

Do této sekce zařadíme popisy různých barevných modelů jako například RGB a také jednotlivých formátů obrázků jako jsou rastrové, paletové a JPEG.

U různých obrazových formátů a barevných reprezentací se totiž liší způsob vkládání zpráv a my se budeme snažit o přiblížení používaných reprezentací, abychom měli jasnější představu, jakým způsobem se obvykle zpráva do obrázků vkládá.

1.4.1 Reprezentace barev

Nejprve se zaměříme na odlišné reprezentace barev. Ty se totiž používají v různých formátech obrázků.

Pokud lidské oko funguje správně, je schopno vnímat a rozlišovat různé složky světla. To všechno pomocí dvou typů receptorů, které se nacházejí na sítnici.

Jedním typem jsou tyčinky. Ty jsou velmi citlivé na intenzitu světla. Pomocí nich bychom tedy byli schopni vnímat černobíle.

Druhým typem receptorů jsou čípky. Ty se starají o barevnost obrazu. Čípky nejsou tak citlivé jako tyčinky, ale obrazu dodávají kontrast a barvu. Zároveň ale nejsou všechny čípky stejné. Dělí se na tři druhy, podle toho, na jaké barevné spektrum se specializují. Jedná se o červenou, zelenou a modrou barvu. Každý čípek se tedy specializuje na nějakou část světelného spektra a dohromady tvoří celkový obraz.

Každou vnímanou barvu tím pádem dokážeme popsat jako trojici, která odpovídá míře stimulace jednotlivých druhů čípků. Tyto hodnoty tím odpovídají intenzitě jednotlivých spekter světla. Všechny barvy proto můžeme popsat jako trojrozměrný prostor.

Přesto, že na světě existuje nekonečně mnoho barev, lidské oko je schopno rozlišit jen jejich konečnou podmnožinu. Od toho se také odvíjí reprezentace barev. Pro popis barvy můžeme využít modely s různou přesností. Vše záleží na tom, jak přesně požadujeme danou barvu vyjádřit a kolik bitů jsme ochotni na tuto reprezentaci poskytnout.

Podle způsobu, jakým se každý barevný model využívá, rozlišujeme aditivní a subtraktivní model míchání barev.

Aditivní model míchání barev

U aditivního modelu míchání barev se jednotlivé složky barev sčítají a tím vytvářejí výslednou barvu.

Pokud bychom tedy chtěli například ve formátu RGB vytvořit žlutou barvu, vysílali bychom zároveň červenou $(1, 0, 0)$ a zelenou $(0, 1, 0)$ barvu. Jednotlivé složky by se sečetli na $(1, 1, 0)$, což přesně odpovídá žluté barvě.

Subtraktivní model míchání barev

Na rozdíl od předchozího případu potřebuje tento model nějaký vnější zdroj světla. Využívá totiž vlastnosti, že každá látka odráží jen určitou část barevného spektra a my ji tak vnímáme jako nějakou barvu. Pokud ale složíme přes sebe více takovýchto látek, každá z nich pohltí nějakou část spektra a výsledná barva tedy vznikne odečítáním jednotlivých hodnot.

Pro ukázku budeme používat CMY model, který se skládá z azurové, purpurové a žluté. Pokud bychom tedy chtěli namíchat červenou barvu, postupovali bychom následovně. Jelikož vnější zdroj světla vysílá všechny

barevné složky, potřebujeme smíchat barvy takovým způsobem, abychom eliminovali zelenou a modrou složku. Vezmeme tedy purpurovou barvu, která pohlcuje zelenou složku, a tu smícháme se žlutou, která pohlcuje modrou složku. Tím nám vznikla barva pohlcující jak zelenou tak modrou složku a jeví se tedy červená.

Subtraktivní model se používá například u tiskáren, kde potřebujeme u každé barvy vědět, jakým způsobem ji namíchat, aby odrážela správnou část světelného spektra.

RGB

Asi nejznámějším způsobem, jakým se reprezentují různé barvy, je model RGB.

Tento model je odvozen od způsobu, jakým je každá barva vnímána lidským okem. Libovolná barva se tedy zapisuje jako uspořádaná trojice (r, g, b), kde r značí intenzitu červeného světla, g značí intenzitu zeleného světla a b značí intenzitu modrého světla. Jednotlivé složky nabývají v tomto modelu hodnot od nuly do jedné, kde jednička značí plnou intenzitu a nula žádnou.

Tento typ reprezentace se podle [14] používá v zařízeních jako například monitor nebo televize. Jedná se totiž o aditivní model, proto se pro toto využití hodí.

YCbCr

Tato reprezentace barev vznikla kvůli potřebě lepší komprese přenášených a uchovávaných dat.

Na rozdíl od RGB nemá tento model tři barevné složky. První složka, která se označuje Y, nese informaci o intenzitě jasu a teprve další dvě složky, označované jako Cb a Cr, jsou chromatické. Důvod tohoto způsobu reprezentace je opět spojen s vnímáním lidského oka.

Člověk velmi intenzivně reaguje na změny jasu, ale na chromatické změny již tolik citlivý není. Z toho důvodu můžeme pro popis chromatických složek použít mnohem méně bitů, čímž se reprezentace jednotlivých barev stává úspornější. Této vlastnosti se podle [2] využívá například ve formátu JPEG a v přenosu televizního signálu.

Převod mezi RGB modelem a modelem YCbCr je jednoduchá lineární transformace, a jedná se tedy o rychlý proces. Podrobnější informace je možné najít v [2].

1.4.2 Obrazové formáty

Zde se podíváme, jakým způsobem mohou být obrázky zakódovány a jakým způsobem se v každém případě vkládá tajná informace.

Způsob reprezentace obrázků se dělí na rastrový a vektorový formát. V praxi se daleko častěji používá rastrový formát, a proto se zde na něj zaměříme.

Pro rastrový formát se dále využívá několik způsobů komprese. Zmíníme zde asi dva nejznámější. Bude to paletový formát a transformace JPEG.

Obecný rastrový formát

Rastrový formát popisuje výsledný obraz pomocí jednotlivých pixelů. Jednotlivé body jsou uspořádány do mřížky a každý pixel uchovává informaci o své barvě a pozici.

Pro popis barvy je samozřejmě důležité, jestli je obrázek černobílý nebo barevný. V případě černobílého obrazu stačí pro každý pixel uchovávat jedinou hodnotu. Pokud je ale obrázek barevný, musíme pro každý pixel uchovat informaci o každé barevné složce. Obrázek si tak lze představit jako tři stejně veliké matice, každá pro jeden barevný kanál.

Jako příklad můžeme uvést formát BMP. Tento formát používá pro popis barvy model RGB. Podle [5] se na zápis každého barevného kanálu většinou využívá 8 bitů, tudíž se v každé složce mohou vyskytovat hodnoty od nuly do 255.

V tomto obrazovém formátu se při vkládání zprávy mění jednotlivé hodnoty pixelů pro každý barevný kanál. Pokud tedy máme barevný obrázek 5×5 pixelů, můžeme při vkládání změnit až 75 hodnot. Většinou se ale pro vkládání používá jen nějaká předem domluvená část, aby byla menší pravděpodobnost detekce.

Paletový formát

V paletovém formátu se každý obrázek skládá z hlavičky, palety a vlastních dat. Tento formát je velice podobný již popsanému základnímu rastrovému formátu. Data jsou totiž ve formě matice, kterou jsme popsali výše. Jediná změna je v tom, že místo popisu barvy je na každé pozici matice ukazatel do palety. V paletě jsou pak pod svým indexem jednotlivé barvy popsány přesně.

Samozřejmě zde ale musí být nějaké omezení na počet barev v paletě, jinak bychom mohli mít velmi dlouhé popisy ukazatelů a rozsáhlou paletu. U některých obrázků tedy dochází ke ztrátové kompresi.

U barevných obrázků se tedy jedná o vylepšení, protože místo třech velkých matic je potřeba si pamatovat pouze jedinou a k ní indexovanou paletu barev.

Klasických příkladem je formát GIF. Ten používá paletu maximální velikosti 256 barev.

Existuje hned několik způsobů, jakými lze vkládat zprávy do paletových formátů. Jedním z nich je vkládání do matice ukazatelů, přičemž paleta je

seřazená podle jasu. Tato metoda je ale podle [15] nevýhodná a proto ji ani nebudeme zkoumat. Další možností je vkládání do palety. To spočívá v uspořádání palety. Tato metoda ovšem také není ideální. Asi nejlepší způsob, je vkládání pomocí parit. V této metodě má každá barva palety nějakou jinou barvu do páru a jsou jim přiřazeny rozdílné hodnoty. Zpráva se pak ukládá do dat tím způsobem, že se v případě potřeby přepojí ukazatel na podobnou barvu s odlišnou přiřazenou hodnotou. Podrobnosti o této metodě lze najít v [15].

JPEG

Asi nejznámější obrazový formát je JPEG. Jedná se o nejvyužívanější formát pro fotografie, protože je, třeba oproti BMP, mnohem méně náročný na paměť. Zároveň se ale jedná o ztrátovou kompresi, takže obrázek uložený do tohoto formátu bude pravděpodobně nepatrně pozměněn. Míra komprese se ale dá určit a komprese je navíc provedena takovým způsobem, že ji lidské oko téměř nepostřehne.

Převod do tohoto formátu probíhá v několika krocích:

- Transformace barev do modelu YCbCr.
- Částečné redukování informace o chromatických kanálech.
- Rozdělení obrázku do bloků velikosti 8×8 pixelů.
- Provedení diskrétní kosinové transformace na všechny složky (Y, Cb, Cr) v každém bloku.
- Kvantizace jednotlivých bloků.
- Konečná komprimace výsledných dat všech bloků například Huffmanovým kódováním.

Nejprve se provede změna barevného modelu, protože, jak již bylo řečeno, lidské oko je u modelu YCbCr mnohem méně citlivé na změnu u chromatických složek. Můžeme tedy tyto složky částečně redukovat. Většinou se jedná o podvzorkování na polovinu.

Bloky velikosti 8×8 budou až do posledního kroku vystupovat jako samostatné. Další změny se tedy budou provádět vždy jen v rámci jednoho bloku.

Diskrétní kosinová transformace (DCT) vlastně provede převod mezi bázemi. Dostaneme tak každý blok jako složení signálů s různou frekvencí. Pro každý blok tedy máme tři matice. Jednu pro jas a dvě pro chromatické kanály. Jedná se o matice koeficientů, které patří k signálům určité frekvence.

Až do této doby jsme, pokud nepočítáme podvzorkování v druhém kroku, neprovedli žádnou kompresi. Jelikož má ale lidské oko tu vlastnost, že je

málo citlivé ke změnám u signálů vysoké frekvence, můžeme koeficienty u takovýchto signálů zaokrouhlit. K tomu slouží kvantizační matice. Tou po složkách vydělíme koeficienty, které jsme dostali z DCT. Výsledné hodnoty zaokrouhlíme tak, abychom dostali celá čísla. Kvantizační matici si v tomto kroku můžeme vybrat, abychom mohli kontrolovat míru komprese. Hodnoty kvantizačních matic u vysokofrekvenčních změn jsou ale vždy voleny tak, aby výsledné hodnoty byly nulové nebo v okolí nuly.

Poslední část už jen seřadí všechny prvky a zakóduje je. Tato část není pro naše účely důležitá.

Podrobnější informace k celému kompresnímu procesu je možné najít v [2].

Při vkládání do formátu JPEG se mění hodnoty jednotlivých DCT koeficientů. Většinou se ale nemění všechny hodnoty. Podle [13] je histogram koeficientů po kvantizaci téměř symetrický kolem nuly a připomíná Gaussovo rozdělení. V nule je pak výrazně vyšší hodnota. Detekce by proto byla snadná, pokud bychom měnili i nulové koeficienty. Vkládáním do nulového koeficientu se totiž výrazně snižuje počet těchto koeficientů. Tato změna by pak na histogramu byla velmi viditelná a proto se většinou pro vkládání nulové koeficienty nepoužívají. Podrobnější popis histogramového útoku lze najít v [16] a [2].

1.5 Nejznámější steganografické metody

Zde jen velmi stručně popíšeme ty nejjednodušší metody steganografie. Jednak si ukážeme, jakým způsobem je možné zprávy vkládat, a navíc budeme moci později tyto způsoby porovnat se složitějšími metodami.

1. **LSB nahrazování** je asi nejznámější steganografická metoda. Zde se LSB (nejméně významný bit) každého prvku nosiče nahradí bitem zprávy. Například u rastrových obrázků se tedy jedná o poslední bity hodnot pixelů, kdežto u JPEG obrázku se jedná o poslední bity hodnot nenulových DCT koeficientů. Tato metoda je ale snadno detekovatelná. U rastrových formátů se dá snadno odhalit pomocí histogramu obrázku, což je blíže popsáno v [16] a [2]. Po dosazení do 1.3.3 snadno zjistíme, že hodnota relativní kapacity nosiče $\alpha = 1$. Jelikož při vkládání musíme průměrně měnit každý druhý LSB, je míra změny $\rho = \frac{1}{2}$ a tedy efektivita $e = 2$.

2. **LSB přizpůsobování** je vylepšená verze LSB nahrazování. Tato metoda se snaží zachovávat rozložení hodnot prvků nosiče, tudíž není tak lehce odhalitelná. Bity zprávy jsou jako předtím na LSB hodnot stegoobjektu. Tentokrát se ale vkládá trochu jiným způsobem. Pokud se LSB nosiče neshoduje s bitem zprávy, k nekorespondující hodnotě nosiče se náhodně přičte buď 1 nebo -1. Tímto způsobem se

tedy zachová tvar histogramu a případně i rozložení DCT koeficientů. Mohlo by se zdát, že jelikož budeme muset tímto způsobem měnit mnohem více bitů nosiče, tak se zhorší distorze a tím i efektivita. To ale není pravda. Podle definice 1.3.4 nezáleží na počtu změněných bitů, ale na hodnotě jako takové. Pokud například hodnoty převedeme na barvu pixelů, pak je jedno, jestli je barva o jeden stupínek světlejší nebo tmavší. Stále se jedná pouze o jeden stupeň. Distorze nosiče tedy zůstává stejná a efektivita je také zachovaná. Stejně tak není důvod, aby se změnila relativní míra změny a kapacita nosiče. Máme tedy parametry

$$\alpha = 1 \qquad \rho = \frac{1}{2} \qquad \mathrm{e} = 2.$$

1.6 Samoopravné kódy

Zde shrneme pár základních informací o samoopravných kódech. Nebudeme se jimi zabývat nijak do hloubky, zmíníme jen to, co budeme potřebovat v dalších kapitolách.

Nejprve začneme tím, co je to vlastně kód.

Definice 1.6.1 (Kód)**.** *Množinu* $C \subseteq \mathbb{F}_q^n$ *dimenze* $\dim C = k$ *nazveme* $[n,k]_q$ *kód.*

Všimněme si, že v poslední definici n znamená délku kódu, neboli délku kódových slov, kterých je celkem q^k. Pokud tedy máme $k = n$, pak se jedná o triviální kód, který vyplňuje celý prostor. Je tedy zřejmé, že vždy platí $k \leq n$.

Pro popsání základních vztahů mezi jednotlivými slovy, prvky prostoru $\mathbb{F}_q^n$, potřebujeme veličinu, která ukazuje, jak moc se od sebe jednotlivá slova liší. K tomu nám bude sloužit Hammingova vzdálenost. Jedná se o známý pojem z teorie kódů, ale pro úplnost raději uvedeme její definici.

Definice 1.6.2 (Hammingova vzdálenost)**.** *Pro* $x, y \in \mathbb{F}_q^n$ *definujeme* Hammingovu vzdálenost *jako*

$$\mathrm{d_H}(x,y) = |\{i; x_i \neq y_i\}|\,.$$

Jedná se vlastně o počet souřadnic, ve kterých se x a y neshodují. Ekvivalentně lze Hammingovu vzdálenost definovat jako

$$\mathrm{d_H}(x,y) = \sum_{i=1}^{n} (1 - \delta_{x_i,y_i}),$$

kde $\delta_{ij} = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases}$.

Pro popis jednotlivého slova se nám bude hodit další veličina, která nám napoví, kolik nenulových prvků dané slovo má. Nejjednodušší způsob je využít předchozí definici a změřit Hammingovu vzdálenost prvku od nulového slova.

Definice 1.6.3 (Hammingova váha)**.** *Pro $x \in \mathbb{F}_q^n$ definujeme* Hammingovu váhu *jako*

$$\mathrm{w}(x) = \mathrm{d_H}(x, 0).$$

Kódy mohou být různé. Některé jsou efektivní, některé mají hezké vlastnosti a některé nevynikají ani jedním.

Definice 1.6.4 (Lineární kód)**.** *Kód $C \subseteq \mathbb{F}_q^n$ nazveme* lineární*, pokud je podprostorem prostoru $\mathbb{F}_q^n$.*

Lineární kódy je skupina kódů, které mají hezké vlastnosti, a tudíž se s nimi pěkně pracuje. Z tohoto důvodu se dále budeme bavit pouze o nich. Od teď tedy budeme předpokládat, že každý kód je lineární, pokud neřekneme jinak.

Jelikož je lineární kód podprostorem, jednou z jeho pěkných vlastností je, že se dá popsat zkráceným tvarem. Lze vybrat některé vektory, jejichž lineární obal vytvoří celý prostor kódu.

Definice 1.6.5 (Generující matice)**.** Generující maticí $[n, k]_q$ *kódu $C \subseteq \mathbb{F}_q^n$ nazveme matici* G *tvaru $k \times n$ takovou, že její řádky tvoří bázi kódu C.*

Pro popis vlastností kódu potřebujeme definovat další veličiny, pomocí kterých je možné jednotlivé kódy porovnávat.

Definice 1.6.6 (Minimální vzdálenost)**.** Minimální vzdálenost *kódu C definujeme jako*

$$\mathrm{d}_C = \min_{x,y \in C} \mathrm{d_H}(x, y),$$

kde $x \neq y$.

Minimální vzdálenost kódu vlastně popisuje, jak moc jsou od sebe jednotlivá kódová slova vzdálena.

Definice 1.6.7 (Vzdálenost od kódu)**.** *Pro kód C a $x \in C$ definujeme* vzdálenost od kódu $\mathrm{d_H}(x, C)$ *jako*

$$\mathrm{d_H}(x, C) = \min_{c \in C} \mathrm{d_H}(x, c).$$

Naposledy definovaná veličina ukazuje, jak daleko je libovolné vybrané slovo od nejbližšího kódového slova. Hned ji použijeme k definování další veličiny, která popíše tuto vlastnost globálně.

Definice 1.6.8 (Průměrná vzdálenost od kódu)**.** *Pro* $[n,k]_q$ *kód* C *definujeme* průměrnou vzdálenost od kódu $\mathrm{R_a}$ *jako*

$$\mathrm{R_a} = \frac{\sum_{x\in\mathbb{F}_q^n} \mathrm{d_H}(x,C)}{q^n}.$$

V poslední definici vlastně jen děláme průměr ze vzdálenosti od kódu pro všechna možná slova.

Definice 1.6.9 (Krycí poloměr kódu)**.** Krycí poloměr $[n,k]_q$ *kódu* C *definujeme jako*

$$\mathrm{R} = \max_{x\in\mathbb{F}_q^n} \mathrm{d_H}(x,C).$$

Krycí poloměr kódu sice popisuje, kolik chyb můžeme maximálně detekovat, ale jelikož se jedná pouze o maximální hodnotu, tak nám neříká nic o tom, jak jsou slova kódu rozložena. Pokud je totiž rozložení kódových slov špatné, nemůžeme u některých kódových slov opravit a ani detekovat žádnou nastalou chybu.

Pro další úvahy potřebujeme definovat pár pojmů z odvětví kombinatoriky. Jedná se sice o docela jiné odvětví matematiky, ale potřebujeme nějakým způsobem popsat, jak dobře daný kód pokrývá celkový prostor a ohodnotit tím jeho efektivitu.

Definice 1.6.10 (Kombinatorická koule)**.** *Pro střed* $x \in \mathbb{F}_q^n$ *a poloměr* $r \in \mathbb{Z},\ 0 \le r$ *definujme* Kombinatorickou kouli $\mathrm{B}(x,r)$ *jako*

$$\mathrm{B}(x,r) = \{y \in \mathbb{F}_q^n;\ \mathrm{d_H}(x,y) \le r\}.$$

Definice 1.6.11 (Objem kombinatorické koule)**.** *Pro* $x \in \mathbb{F}_q^n$ *a* $0 \le r,\ r \in \mathbb{Z}$ *definujeme* objem kombinatorické koule *jako*

$$\mathrm{V}_q(n,r) = \sum_{i=1}^{r} \binom{n}{i} (q-1)^i.$$

Jistě se může zdát matoucí, že je předchozí definice brána jako definice. Jedná se ale opravdu jen o definici pojmu. Zatím jsme totiž nepřiřadili žádný vztah mezi kombinatorickou koulí definovanou v 1.6.10 a objemem kombinatorické koule definovaným v 1.6.11. To bude předmětem následujícího tvrzení, které jen potvrdí vztah, který je podle názvu již zřejmý.

Tvrzení 1.6.12. *Pro* $x \in \mathbb{F}_q^n$ *a* $0 \le r,\ r \in \mathbb{Z}$ *platí*

$$\mathrm{V}_q(n,r) = |\mathrm{B}(x,r)|. \tag{1.1}$$

Důkaz. Dle 1.6.2 a 1.6.3 pro $x,y \in \mathbb{F}_q^n$ platí $\mathrm{w}(x-y) = \mathrm{d_H}(x,y)$. Proto je počet $y \in \mathbb{F}_q^n$ takových, že $\mathrm{d_H}(x,y) = i$, stejný jako počet slov $z \in$

$\mathbb{F}_q^n$; $\mathrm{w}(z) = i$. Stačí tedy spočítat $z \in \mathbb{F}_q^n$ Hammingovy váhy $\mathrm{w}(z) = i$. Takových je ale právě $\binom{n}{i}(q-1)^i$. Celkem tedy máme

$$|\mathrm{B}(x,r)| = \sum_{i=1}^{r} \binom{n}{i}(q-1)^i,$$

což je dle definice 1.6.11 právě $\mathrm{V}_q(n,r)$.

□

Pro větší názornost uvádíme následující poznámku, která je po pochopení všech základních pojmů zřejmá.

Poznámka 1.6.13. *Pro $[n,k]_q$ kód C s krycím poloměrem R platí*

$$\bigcup_{x \in C} \mathrm{B}(x,R) = \mathbb{F}_q^n. \tag{1.2}$$

Jak jsme již zmínili dříve, lineární kódy mají hezké vlastnosti. Jelikož se dají po vzoru definice 1.6.5 zapsat jako matice prvků generujících celý kód, tak se určitě dají jednoduše najít všechny prvky, které jsou, jakožto vektory, na prvky kódu kolmé. Znamená to tedy, že jejich skalární součin je nulový.

Definice 1.6.14 (Duální kód)**.** *Pro $[n,k]_q$ kód C definujeme* duální kód $C^\perp$ *jako*

$$C^\perp = \left\{ x \in \mathbb{F}_q^n ;\ \forall c \in C :\ x \cdot c = 0 \right\},$$

kde $\cdot$ značí skalární součin.

Je zřejmé, že pro ověření stačí vyzkoušet skalární součin jen pro vektory z generující matice, protože ostatní kódová slova jsou jen jejich lineární kombinací.

Můžeme také poznamenat, že duální kód je zřejmě také lineární kód a že pro $[n,k]_q$ kód C a jeho duální kód $C^\perp$ platí:

1. $\dim C + \dim C^\perp = n$,
2. $C^\perp$ je $[n, n-k]_q$ kód.

Jelikož je každý kód zároveň duálním kódem svého duálního kódu, je zřejmé, že je každý kód určen svým duálním kódem jednoznačně a tedy může být pomocí něho popsán.

Definice 1.6.15 (Paritní matice)**.** *Pro $[n,k]_q$ kód C definujeme* paritní matici H *jako generující matici jeho duálního kódu $C^\perp$.*

Paritní matice se u kódů využívá mnohem více než generující matice, protože má hezké vlastnosti. Například pro $[n,k]_q$ kód C a jeho paritní matici H platí:

1. H je matice typu $(n-k) \times n$,
2. $C = \left\{x \in \mathbb{F}_q^n\,;\ \mathrm{H} \cdot x = 0\right\}$.

Je asi zřejmé, že v posledním bodě zastupuje nula na pravé straně rovnosti celý nulový vektor. Ještě poznamenáme, že vektory zde budeme brát jako sloupcové.

Jak jsme již zmínili, paritní matice kódu se využívá častěji než generující matice a definují se pomocí ní další prvky kódu.

Definice 1.6.16 (Syndrom)**.** *Pro kód C s paritní maticí* H *a $x \in \mathbb{F}_q^n$ definujeme* syndrom *x jako*

$$\mathrm{s}(x) = \mathrm{H} \cdot x.$$

Je vhodné poznamenat, že syndrom je vektor délky $(n-k)$, kde n je délka kódu a k je jeho dimenze.

Podle definice 1.6.16 umíme k danému slovu najít jeho syndrom. My budeme ale potřebovat vyhledávat i druhým směrem. Jelikož se ale nejedná o prosté zobrazení, bude výsledkem množina.

Definice 1.6.17 (Rozkladová třída)**.** *Pro syndrom* s *definujeme jeho* rozkladovou třídu *$C(\mathrm{s})$ jako*

$$C(\mathrm{s}) = \left\{x \in \mathbb{F}_q^n\,;\ \mathrm{H} \cdot x = \mathrm{s}\right\}.$$

Pro větší názornost poznamenáme, že pro nulový syndrom, je

$$C(0) = C,$$

dále také pro dva syndromy $\mathrm{s}_1, \mathrm{s}_2$, kde $\mathrm{s}_1 \neq \mathrm{s}_2$ platí

$$C(\mathrm{s}_1) \cap C(\mathrm{s}_2) = \emptyset$$

a platí, že

$$\bigcup_{\mathrm{s} \in \mathbb{F}_q^{n-k}} C(\mathrm{s}) = \mathbb{F}_q^n.$$

Z toho se pak jednoduše dá odvodit následující poznámka.

Poznámka 1.6.18. *Pro libovolný syndrom* s *platí*

$$x \in C(\mathrm{s}),\ c \in C \Rightarrow (x + c) \in C(\mathrm{s}).$$

Jak jsme již uváděli, budeme potřebovat pro daný syndrom najít slovo, které by tomuto syndromu odpovídalo. To splňují všechny prvky rozkladové třídy. My ale potřebujeme nějakého jednoznačného reprezentanta, proto zavádíme následující definici.

Definice 1.6.19 (Reprezentant rozkladové třídy)**.** Reprezentanta rozkladové třídy $\mathfrak{e}(\mathrm{s})$ *definujeme pro syndrom* s *jako*

$$\mathfrak{e}(\mathrm{s}) = \arg \min_{x \in C(\mathrm{s})} (\mathrm{w}(x)).$$

V předchozí definici je jeden problém. Může se stát, že pro nějakou rozkladovou třídu budou existovat dva prvky se stejnou váhou a nebude existovat žádný prvek s váhou nižší. V takovém případě nevíme, který prvek si vybrat. Abychom tedy mohli reprezentanta rozkladové třídy určit jednoznačně, musíme nějakým způsobem určit, jak v takovém případě postupovat. Můžeme například brát ten prvek, který má nenulové prvky na nižších pozicích.

Tvrzení 1.6.20. *Nechť máme kód* C *s krycím poloměrem* R*, pak pro každý syndrom* s *platí* $\mathrm{w}(\mathfrak{e}(\mathrm{s})) \leq \mathrm{R}$*. Každý syndrom* s *se tedy dá napsat jako součet maximálně* R *sloupců paritní matice kódu* C*.*

Důkaz. Pro každé $x \in C(\mathrm{s})$ platí

$$\mathrm{R} \overset{1.6.9}{=} \max_{y \in \mathbb{F}_q^n} \mathrm{d_H}(y, C) \geq \mathrm{d_H}(x, C) \overset{1.6.7}{=} \min_{c \in \mathrm{C}} \mathrm{d_H}(x, c) = \min_{c \in C} \mathrm{w}(x - c).$$

Ale podle 1.6.18 $(x - c) \in C(\mathrm{s})$. Proto

$$\min_{c \in C} \mathrm{w}(x - c) \overset{1.6.19}{=} \mathrm{w}(\mathfrak{e}(s)).$$

Celkem tedy dostáváme

$$\mathrm{w}(\mathfrak{e}(\mathrm{s})) \leq \mathrm{R}.$$

K dokončení důkazu stačí pouze ukázat, jak z předchozího odhadu plyne, že lze každý syndrom napsat jako součet maximálně R sloupců paritní matice. To je ale snadné, neboť pro každý syndrom s platí následující rovnost

$$\mathrm{s} = \mathrm{H} \cdot \mathfrak{e}(\mathrm{s}).$$

Jelikož má ale $\mathfrak{e}(\mathrm{s})$ nejvýše R nenulových prvků, pak výsledný syndrom dostaneme jako součet nejvýše R sloupců paritní matice.

□

Jak jsme již uvedli, pro měření efektivity kódů můžeme použít rozložení kódových slov v prostoru. Čím rovnoměrněji budou kódová slova vyplňovat celý prostor a čím méně se budou jejich pole působnosti překrývat, tím můžeme považovat kód za efektivnější.

Tvrzení 1.6.21. *Nechť* C *je* $[n, k]_q$ *kód s krycím poloměrem* R*, pak*

$$\mathrm{V}_q(n, \mathrm{R}) \geq q^{n-k}. \tag{1.3}$$

Důkaz. Z 1.6.13 víme, že $\bigcup_{x \in C} \mathrm{B}(x, R) = \mathbb{F}_q^n$. Bude-li platit $\forall x, y \in C;\ x \neq y$: $\mathrm{B}(x, R) \cap \mathrm{B}(y, R) = \emptyset$, pak $q^n = \left|\mathbb{F}_q^n\right| = \sum_{x \in C} |\mathrm{B}(x, R)| = q^k \, \mathrm{V}_q(n, \mathrm{R})$. Pak tedy $\mathrm{V}_q(n, \mathrm{R}) = q^{n-k}$. Pokud $\mathrm{B}(x, R) \cap \mathrm{B}(y, R) \neq \emptyset$, dostáváme $\mathrm{V}_q(n, \mathrm{R}) \geq q^{n-k}$. □

Předchozí tvrzení nám dává spodní odhad pro objem koule o poloměru R. Pokud se nám povede dosáhnout tohoto odhadu, pak jsme získali, z tohoto hlediska, nejefektivnější kód.

Definice 1.6.22 (Perfektní kód)**.** $[n, k]_q$ *kód* C *s minimální vzdáleností* d_C *je* perfektní $\overset{def}{\iff}$ $\mathrm{V}_q\left(n, \frac{\mathrm{d}_C - 1}{2}\right) = q^{n-k}$.

Kapitola 2

Maticové vkládání

Doteď jsme si ukázali jen stegosystémy, které vkládaly vždy jeden bit informace do jednoho prvku nosiče. Nyní popíšeme stegosystém, který pro vkládání využívá samoopravné kódy. Pomocí nich vkládá vždy několik bitů zprávy do několika prvků nosiče najednou. Tím dokážeme snížit počet změn, které jsme nuceni udělat, abychom zprávu vložili. Podle [6] a [11] se použitím samoopravných kódů docílí zvýšení efektivity vkládání, což je samozřejmě žádoucí.

U maticového vkládání se jako součást klíče používá dohodnutý kód C, přesněji jeho paritní matice H. Budeme tedy předpokládat, že odesilatel a příjemce mají takovou matici domluvenou. Popíšeme metodu zvanou maticové vkládání nebo také podle [3] syndromové kódování. Tato metoda využívá paritní matici kódu ke zjišťování syndromů jednotlivých slov.

2.1 Matematizace

Jelikož zatím máme nosiče i zprávy v nějakém obecném tvaru, potřebujeme je nějakým způsobem převést na objekty, se kterými se nám bude v tomto stegosystému dobře pracovat.

V našem případě budeme požadovat vektory určitého tvaru. V této části proto uvedeme, jakým způsobem se budou jednotlivé objekty na požadovaný tvar převádět.

Předpokládejme, že domluvená paritní matice H je tvaru $(n-k) \times n$ pro nějaký $[n,k]_q$ kód. Nemůžeme bohužel využít jakýkoliv kód. Pro q je nutné omezení, že $q \leq p$, kde nosič x je tvořen prvky ze $\mathbb{Z}_p$. Toto omezení ale není tak výrazné, protože, jak již bylo zmíněno v sekci 1.4, p většinou nabývá nejméně velikosti 256.

Jak již bylo řečeno, pro některé stegosystémy budeme definovat množinu vhodných nosičů, které budeme pro vkládání používat. V tomto případě

označíme nosič za vhodný, pokud jeho jednotlivé prvky $\overline{x_i}$ splňují

$$\overline{x_i} \geq \left\lfloor \frac{q-1}{2} \right\rfloor, \qquad \overline{x_i} \leq \left\lceil \frac{2p-q-1}{2} \right\rceil.$$

Pro domluvený kód a paritní matici budeme v této metodě požadovat, aby bylo možné nosič $x \in \mathcal{C}$ nějakým způsobem reprezentovat jako sloupcové vektory délky n, které se skládají z prvků $\mathbb{F}_q$. Budeme předpokládat, že délka vybraného nosiče je násobkem n, tedy $|x| = n \cdot l$.

Pro takovouto volbu nosiče budeme požadovat, aby vybraná zpráva $m \in \mathcal{M}$ měla délku $(n-k)l$ a její prvky byly z $\mathbb{F}_q$. Pro tento stegosystém ji totiž budeme potřebovat vyjádřit jako l sloupcových vektorů délky $(n-k)$.

Pro jednodušší postup si definujeme další operace.

Definice 2.1.1 (Spojení vektorů)**.** *Pro dva řádkové vektory u a v, kde $u = (u_1, \ldots, u_n)$ a $v = (v_1, \ldots, v_m)$ pro $n, m \in \{1, 2, \ldots\}$ definujeme jejich* spojení $u \,\|\, v$*, jako vektor délky $(n+m)$ tvaru*

$$u \,\|\, v = (u_1, u_2, \ldots, u_n, v_1, v_2, \ldots, v_m).$$

Dále ještě budeme potřebovat nějak reprezentovat posloupnost prvků ze $\mathbb{Z}_p$ jako prvky $\mathbb{F}_q^n$, kde $q \leq p$. Konkrétně to je nutné pro matematizaci nosiče.

Definice 2.1.2 (Projekce ${}_p\pi_q(x)$)**.** *Pro $p, q \in \{1, 2, \ldots\}$ a prvek $x \in \mathbb{Z}_p^n$, budeme jako ${}_p\pi_q(x)$ označovat* projekci *vektoru*

$$x = (x_1, x_2, \ldots, x_n),$$

kde $x_i \in \mathbb{Z}_p$, na vektor

$${}_p\pi_q(x) = (\widetilde{x}_1, \widetilde{x}_2, \ldots, \widetilde{x}_n),$$

kde $\widetilde{x}_i \in \mathbb{Z}_q$.

Právě definovaná projekce může mít různé podoby. Nejčastěji se ale používá modulo pro $p > q$. V případě, kdy je $p < q$, se pak použije stejná hodnota, která se jen uvažuje v jiné grupě. Ještě poznamenejme, že pro tuto projekci budeme uvažovat množinu

$$\mathbb{Z}_q = \left\{ \left\lceil -\frac{q-1}{2} \right\rceil, \ldots, -1, 0, 1, \ldots, \left\lceil \frac{q-1}{2} \right\rceil \right\}.$$

Totéž budeme požadovat pro p. Tato podmínka nám zajistí nejmenší možnou prováděnou změnu.

Pro popsanou projekci budeme požadovat, aby platilo

$${}_p\pi_q({}_q\pi_p(x)) = x, \tag{2.1}$$

kde $p > q$ a $x \in \mathbb{Z}_q^*$.

Abychom na původním nosiči prováděli co nejmenší změny, budeme dále předpokládat, že kdykoliv $p > q$ a $e \in \mathbb{Z}_q^n$, pak pro všechna $i \in \{0, 1, 2, \ldots, n-1\}$ platí

$$|_q\pi_p(e_i)| \in \left\{ \left\lceil -\frac{q-1}{2} \right\rceil, \ldots, -1, 0, 1, \ldots, \left\lceil \frac{q-1}{2} \right\rceil \right\}.$$

Ze stejného důvodu budeme požadovat následující podmínku. Nechť máme $n \in \{1, 2, \ldots\}$, $p > q$ a $x, e \in \mathbb{Z}_p^n$ takové, že

$$|x_i| + |e_i| < \left\lceil \frac{p-1}{2} \right\rceil$$

a zároveň

$$-|x_i| - |e_i| > -\left\lceil \frac{p-1}{2} \right\rceil,$$

pro všechna $i \in \{0, 1, 2, \ldots, n-1\}$ pak

$$_p\pi_q(x + e) = {}_p\pi_q(x) + {}_p\pi_q(e).$$

Kdykoliv tedy máme $p > q$, $x \in \mathbb{Z}_p^*$ vhodný nosič a $e \in \mathbb{Z}_q^*$, kde x a e jsou stejného tvaru,bude platit

$$_p\pi_q(x + {}_q\pi_p(e)) = {}_p\pi_q(x) + {}_p\pi_q({}_q\pi_p(e)). \tag{2.2}$$

Jak již bylo řečeno dříve, v praxi se používá modulo, případně stejná hodnota myšlená v jiné grupě. Všechny dodatečné podmínky jsou tedy snadno splněné.

Dále můžeme poznamenat, že v případě, že q je prvočíslo, dostáváme zároveň zobrazení ze $\mathbb{Z}_p^m$ do $\mathbb{F}_q^m$, pro odpovídající m. Stejným způsobem máme i zobrazení z $\mathbb{F}_p^m$ do $\mathbb{Z}_q^m$ pro prvočíslo p.

Jak jsme již zmínili, potřebujeme nosič i zprávu rozdělit na několik sloupcových vektorů stejné délky. K tomu nám poslouží následující definice.

Definice 2.1.3. *Pro $n, l \in \{1, 2, \ldots\}$ a řádkový vektor $x \in \mathbb{F}_q^{ln}$ definujeme pro $i \in 1, 2, \ldots, l$ i-tý* výřez $[x]_i^n$ *jako sloupcový vektor délky n*

$$[x]_i^n = (\widetilde{x}_1, \widetilde{x}_2, \ldots, \widetilde{x}_n)^\top,$$

kde $\widetilde{x}_j = x_{(i-1)n+j}$.

Takto definovaný výřez $[x]_i^n$ vezme i-tou n-tici x a udělá z ní sloupcový vektor. Chceme-li například $[x]_3^n$ pro $x = (\overline{x_1}, \overline{x_2}, \ldots, \overline{x_{5n}})$, získáme

$$[x]_3^n = (\overline{x_{2n+1}}, \overline{x_{2n+2}}, \ldots, \overline{x_{3n}})^\top.$$

Nyní popíšeme proces, který bude při tomto způsobu vkládání proveden jako první, aby se jak nosič tak zpráva upravily na požadovaný tvar. Popíšeme totiž funkci $\mathrm{Emb}(x, m, k)$ jako složení několika jednoduchých procesů.

První fázi, kterou budeme také označovat jako přípravnou, budeme pro $[n, k]_q$ kód značit jako $\mathrm{Pri}_q^{n,k}(x, m, k)$. Jelikož se ale v tomto stegosystému nebude vyskytovat žádný klíč, budeme tento proces také označovat jako $\mathrm{Pri}_q^{n,k}(x, m)$. Vstupem této funkce bude nosič $x \in \mathcal{C}$ a zpráva $m \in \mathcal{M}$, které splňují podmínky zmíněné dříve. Jako výstup funkce pak dostaneme uspořádanou dvojici vektorů $(x_1, x_2, \ldots, x_l)$ a $(m_1, m_2, \ldots, m_l)$, kde l bylo určeno výše. Jednotlivé prvky x_i a m_i, pro $i \in \{1, 2, \ldots, l\}$ jsou sloupcové vektory. Ty jsou v případě x_i dlouhé n a v případě m_i mají délku $(n - k)$.

Definice 2.1.4 (Přípravný proces)**.** *Pro nosič x, zprávu m a $[n, k]_q$ kód C, kde $|x| = n \cdot l$ a $|m| = l(n - k)$, definujeme* přípravný proces $\mathrm{Pri}_q^{n,k}(x, m)$ *jako*

$$\mathrm{Pri}_q^{n,k}(x, m) = ((x_1, x_2, \ldots, x_l), (m_1, m_2, \ldots, m_l)),$$

kde

$$x_i = [{}_p\pi_q(x)]_i^n, \qquad m_i = [m]_i^{n-k}$$

pro $i \in \{1, 2, \ldots, l\}$.

Předchozí definici jsme vytvořili tak, aby pro nosič $x \in \mathcal{C}$ platilo

$${}_p\pi_q(x) = x_1^\top \parallel x_2^\top \parallel \ldots \parallel x_l^\top$$

a pro zprávu $m \in \mathcal{M}$ aby platilo

$$m = m_1^\top \parallel m_2^\top \parallel \ldots \parallel m_l^\top.$$

Můžeme si snadno ověřit, že prvky $x_i \in \mathbb{F}_q^n$ a $m_i \in \mathbb{F}_q^{n-k}$ pro $i \in \{1, 2, \ldots, l\}$, což bylo naší snahou.

Tímto jsme dokončili popis přípravné části funkce Emb. Budeme ale ještě potřebovat popsat dokončovací fázi, která nám pomůže z původního nosiče a vypočítaných změn vytvořit stegoobjekt v odpovídajícím tvaru. Tento proces budeme nazývat dokončovací.

Definice 2.1.5 (Dokončovací proces)**.** *Mějme nosič $x \in \mathbb{Z}_p^{ln}$ délky $|x| = n \cdot l$ a $[n, k]_q$ kód C. Dále mějme l sloupcových vektorů změn $e_1, e_2, \ldots, e_l$, kde $e_i = ((e_i)_1, (e_i)_2, \ldots, (e_i)_n)^\top$ pro $i \in \{1, 2, \ldots, l\}$ a nechť $(e_i)_j \in \mathbb{F}_q$ pro $j \in \{1, 2, \ldots, n\}$. Pak definujeme* výslednou změnu e *jako*

$$e = e_1^\top \parallel e_2^\top \parallel \ldots \parallel e_l^\top.$$

Dále definujeme dokončovací proces *jako*

$$\mathrm{Dok}_p^l(x, e_1, e_2, \ldots, e_l) = x + {}_q\pi_p(e),$$

kde sčítání vektorů bereme po složkách.

Výsledkem dokončovacího procesu je řádkový vektor $y \in \mathbb{Z}_p^{ln}$. Je tedy stejného tvaru jako původní nosič.

Dokončovací proces vlastně provede změny, které vypočítáme. Jedná se tedy o způsob, jakým aplikujeme změny, které vypočítáme v hlavní části algoritmu.

Právě jsme dokončili popis potřebných pomocných procesů a můžeme se zaměřit na klíčovou část celého algoritmu.

2.2 Vkládání

V tomto procesu se budeme snažit upravit příslušný nosič tak, aby se bloky zprávy rovnaly syndromům bloků nosiče. Pro vložení zprávy do nosiče tedy musíme udělat následující:

- Provedeme volbu nosiče a zprávy a připravíme je na proces samotného určení změny.
  - Nejprve vybereme vhodný nosič $x \in \mathcal{C}$ takový, že $x \in \mathbb{Z}_p^{ln}$, kde $l \in \{1, 2, \ldots\}$. Zároveň předpokládáme, že jsme s příjemcem zprávy domluveni na paritní matici H pro $[n, k]_q$ kód C a že $p \geq q$.
  - Vybereme zprávu $m \in \mathbb{F}_q^{(n-k)l}$, kterou chceme vložit do nosiče.
  - Na vybraný nosič a zprávu provedeme přípravný proces.

    $$\mathrm{Pri}_q^{n,k}(x, m) = ((x_1, x_2, \ldots, x_l), (m_1, m_2, \ldots, m_l))$$

    Tím získáme prvky x_i a m_i pro $i \in \{1, 2, \ldots, l\}$, které budeme používat v další části vkládání.
- Provedeme výpočet změn, které se musejí na nosiči provést, abychom dostali požadovaný stegoobjekt.
  - Pro všechna $i \in \{1, 2, \ldots, l\}$ spočteme

    $$\mathrm{s}_i = m_i - \mathrm{H}x_i.$$

    Jednotlivá s_i budeme dále používat jako syndromy a následně budeme počítat jim odpovídající slova.

    Všimněme si, že s_i je nulový vektor, pokud se syndrom x_i rovná bloku zprávy m_i. Jednotlivá s_i jsou totiž vektory, které popisují, o kolik se daný blok zprávy liší od syndromu bloku nosiče. My se budeme snažit, abychom ve výsledku dostali rovnost mezi těmito prvky.

– Všechny s_i, kde $i \in \{1, 2, \ldots, l\}$, uvažujeme jako syndrom, proto můžeme hledat reprezentanty příslušných rozkladových tříd

$$e_i = \mathfrak{e}(\mathrm{s}_i).$$

Výsledná e_i použijeme v dalším kroku jako změny, které musíme s původním nosičem provést. Pro jistotu poznamenáváme, že reprezentanty rozkladových tříd budeme opět brát jako sloupcové vektory.

V případě nulového syndromu získáváme i nulového reprezentanta. To také odpovídá tomu, že v tomto případě nepotřebujeme provádět žádnou změnu.

- Z původního nosiče x a získaných změn e_i, kde $i \in \{1, 2, \ldots, l\}$, vytvoříme stegoobjekt, který ponese vybranou zprávu m.

  – Provedeme dokončovací proces

  $$\mathrm{Dok}_p^l(x, e_1, e_2, \ldots, e_l) = y$$

  a získáme tak kýžený stegoobjekt y.

  Můžeme si všimnout, že v případě, že je požadovaná změna e_i nulová, odpovídá příslušná část stegoobjektu původní části nosiče.

Takto vytvořený stegoobjekt y pošleme příjemci, který je schopen z něj extrahovat zprávu m.

Všimněme si, že původní nosič měníme pouze v případě, že se syndrom nosiče neshoduje s odesílanou zprávou. V případě, že se syndrom nosiče shoduje se zprávou, dostaneme nulová s_i, a tím pádem i nulový vektor změny e_i v kroku dalším. Nakonec tedy budeme odpovídající část původního nosiče měnit o nulový vektor a nosič se tím nezmění.

Zároveň pokaždé, když se syndrom nosiče od zprávy liší, měníme jej o slovo nejmenší možné váhy, protože vždy nalezneme reprezentanta rozkladové třídy, a ten má podle definice 1.6.19 nejmenší možnou váhu.

Můžeme nahlédnout, že pokud je v nějaké rozkladové třídě více slov minimální váhy, je jedno, které slovo si zvolíme. Vkládáním sice vzniknou dva různé stegoobjekty, oba ale nesou stejnou zprávu a na způsob extrakce to nebude mít vliv.

2.3 Extrakce

V této sekci popíšeme, jakým způsobem příjemce ze stegoobjektu extrahuje zprávu. Předpokládejme, že je s odesilatelem domluvený na paritní matici H. Stegoobjekt, který od odesilatele obdržíme, budeme značit y. Připomeneme, že stegoobjekt budeme brát jako vektor. Jakožto příjemce budeme postupovat následovně:

- Obdržený stegoobjekt si nejprve upravíme, abychom z něj později mohli vyčíst zprávu.
  - Předpokládáme, že H je paritní matice $[n,k]_q$ kódu C. Déle ještě předpokládáme že doručený stegoobjekt y je tvaru $y \in \mathbb{F}_p^{ln}$, kde $l \in \{1,2,\dots\}$.
  - Na stegoobjekt y provedeme přípravný proces
    $$\mathrm{Pri}_q^{n,n}(y,\emptyset) = ((y_1,y_2,\dots,y_l),\emptyset)$$
    a tím získáme sloupcové vektory y_i pro $i \in \{1,2,\dots,l\}$, které použijeme v dalším kroku.
- Extrahujeme jednotlivé části zprávy.
  - Pro $i \in \{1,2,\dots,l\}$ spočteme syndromy příslušné y_i.
    $$\mathrm{H}y_i = m_i.$$
- Poskládáme výslednou zprávu.
  - Spojíme jednotlivé části zprávy v jeden řádkový vektor
    $$m = m_1^\top \parallel m_2^\top \parallel \cdots \parallel m_l^\top,$$
    který představuje výslednou zprávu.

Tímto postupem jsme jakožto příjemce dokázali získat zprávu m, která byla do původního nosiče vložená postupem popsaným v 2.2. Nyní stačí ověřit, že se opravdu jedná o původně vloženou zprávu a že skutečně nezáleží na způsobu výběru reprezentanta rozkladové třídy v případě, že existuje více vhodných slov se stejnou váhou.

Pro tento důkaz budeme používat značení ze sekce 2.2. V druhém bodě extrakce počítáme syndrom části stegoobjektu jako

$$\mathrm{H}y_i = \mathrm{H}\left[{}_p\pi_q(x + {}_q\pi_p(e))\right]_i^n \overset{(2.2)}{=} \mathrm{H}\left[{}_p\pi_q(x) + {}_p\pi_q({}_q\pi_p(e))\right]_i^n.$$

Z (2.1) ale máme

$$\mathrm{H}\left[{}_p\pi_q(x) + {}_p\pi_q({}_q\pi_p(e))\right]_i^n = \mathrm{H}\left(x_i + e_i\right) = \mathrm{H}x_i + \mathrm{H}\,\mathfrak{e}(\mathrm{s}_i).$$

Jelikož ale z volby reprezentanta rozkladové třídy platí

$$\mathrm{H}\,\mathfrak{e}(\mathrm{s}_i) = m_i - \mathrm{H}x_i,$$

dostáváme

$$\mathrm{H}y_i = \mathrm{H}x_i + (m_i - \mathrm{H}x_i) = m_i.$$

To tedy přesně odpovídá části původně vložené zprávy. Tím jsme ověřili funkčnost stegosystému a můžeme se zaměřit na odhad vlastností, které jsme touto konstrukcí získali.

2.4 Vlastnosti

Jak jsme již zmínili a předvedli výše, dokážeme využívat samoopravné kódy pro vkládání zpráv do nosičů. Musíme ale ukázat, že to má smysl. Pokud by byl totiž popsaný způsob méně efektivní než postupy popsané v sekci 1.5, nebylo by toto využití kódů ničím užitečné. Jak ale v této sekci ukážeme, získali jsme tím stegosystém, který má lepší vlastnosti než stegosystém ze sekce 1.5.

V části 2.2 jsme vždy vkládali $(n-k)$ q–árních symbolů (t.j. $(n-k)\log_2 q$ bitů) zprávy do bloků nosiče délky n. Jelikož se jednotlivé parametry v závislosti na blocích nemění, můžeme uvažovat, že pracujeme pouze s jedním blokem nosiče a jedním blokem zprávy. Tím tedy po dosazení do 1.3.3 získáváme relativní kapacitu nosiče

$$\alpha = \frac{(n-k)\log_2 q}{n}.$$

Navíc podle tvrzení 1.6.20 měníme při vkládání zprávy nejvýše R hodnot nosiče.

Asi nejdůležitějším parametrem, který jsme použitím samoopravných kódů chtěli vylepšit, je efektivita. Její hodnotu se budeme v následujícím tvrzení snažit určit.

Tvrzení 2.4.1. *Stegosystém vytvořený z binárního $[n,k]$ kódu C popsaným způsobem má efektivitu*

$$\mathrm{e} = \frac{n-k}{\mathrm{R_a}},$$

kde $\mathrm{R_a}$ *značí průměrnou vzdálenost od kódu.*

Důkaz. Podle definice 1.3.6 je efektivita

$$\mathrm{e} = \frac{\mathrm{E}(\log_2 |\mathcal{M}(x)|)}{\mathrm{E}(\mathrm{d}(x,y))}.$$

Pokud se jako v předchozím případě omezíme na jediný blok nosiče a zprávy, dostáváme, že čitatel

$$\mathrm{E}(\log_2 |\mathcal{M}(x)|) = n-k,$$

vkládáme totiž $(n-k)$ binárních symbolů.

Pro dokončení důkazu už jen stačí ukázat, že platí

$$\mathrm{E}(\mathrm{d}(x,y)) = \mathrm{R_a}.$$

Jak je zřejmé z vkládacího postupu na straně 28, původní nosič x a výsledný stegoobjekt y liší právě o $\mathfrak{e}(\mathrm{s})$. Zároveň pro binární kódy platí rovnost

mezi Hammingovou vzdáleností a distorzí. Zřejmě tedy pro jmenovatel platí následující rovnost.

$$\mathrm{E}(\mathrm{d}(x,y)) = \frac{1}{2^{n-k}} \left(\sum_{\mathrm{s}\in\mathbb{F}_2^{n-k}} \mathrm{w}(\mathfrak{e}(\mathrm{s})) \right).$$

Po jednoduché úpravě získáme

$$\frac{1}{2^{n-k}} \left(\sum_{\mathrm{s}\in\mathbb{F}_2^{n-k}} \mathrm{w}(\mathfrak{e}(\mathrm{s})) \right) = \frac{1}{2^n} \left(\sum_{\mathrm{s}\in\mathbb{F}_2^{n-k}} 2^k \, \mathrm{w}(\mathfrak{e}(\mathrm{s})) \right).$$

Dále podle 1.6.18 máme, že

$$\forall x \in C(\mathrm{s}) : \mathrm{w}(\mathfrak{e}(\mathrm{s})) = \mathrm{d}_{\mathrm{H}}(x, C).$$

Zároveň z 1.6.18 víme, že $|C(\mathrm{s})| = 2^k$. Zkombinováním předchozích výsledků dostáváme

$$\mathrm{E}(\mathrm{d}(x,y)) = \frac{1}{2^n} \left(\sum_{\mathrm{s}\in\mathbb{F}_2^{n-k}} \sum_{x\in C(\mathrm{s})} \mathrm{d}_{\mathrm{H}}(x, C) \right).$$

Protože ale tímto způsobem x prochází přes všechny prvky prostoru $\mathbb{F}_2^n$, dostáváme

$$\mathrm{E}(\mathrm{d}(x,y)) = \frac{1}{2^n} \left(\sum_{x\in\mathbb{F}_2^n} \mathrm{d}_{\mathrm{H}}(x, C) \right) \overset{(1.6.8)}{=} \mathrm{R_a}.$$

Celkem tedy získáváme požadovanou efektivitu

$$\mathrm{e} = \frac{n-k}{\mathrm{R_a}}.$$

□

Právě dokázané tvrzení platí pouze pro binární kódy. Pokud budeme chtít počítat efektivitu u q–árních kódů, musíme si uvědomit, jakou definici distorze používáme. Pokud máme totiž $q > 3$, bude výsledek záležet na zvolené definici distorze. Z toho důvodu uvádíme zvlášť tvrzení pro q–ární případ.

Tvrzení 2.4.2. *Stegosystém vytvořený z $[n,k]_q$ kódu C popsaným způsobem má efektivitu*

$$\mathrm{e} = \frac{(n-k)\log_2 q}{{}_q\mathrm{R_a}},$$

kde

$${}_q\mathrm{R_a} = \frac{1}{q^n} \sum_{x\in\mathbb{F}_q^n} \mathrm{d}(x, C).$$

Důkaz. Budeme postupovat jako v předchozím případě. Za čitatel můžeme dosadit

$$\mathrm{E}(\log_2 |\mathcal{M}(x)|) = (n-k)\log_2 q,$$

vkládáme totiž $(n-k)$ q–árních symbolů.

Pro dokončení důkazu tedy opět stačí odhadnout jmenovatel. Chceme ukázat, že platí

$$\mathrm{E}(\mathrm{d}(x,y)) = {}_q\mathrm{R}_\mathrm{a}.$$

Stejně jako v předchozím případě víme, že se původní nosič x a výsledný stegoobjekt y liší právě o $\mathfrak{e}(\mathrm{s})$. Dostáváme tedy následující rovnost.

$$\mathrm{E}(\mathrm{d}(x,y)) = \frac{1}{q^{n-k}} \left(\sum_{\mathrm{s}\in\mathbb{F}_q^{n-k}} \mathrm{d}\left(\overline{0}, \mathfrak{e}(\mathrm{s})\right) \right),$$

kde $\overline{0}$ znamená nulové kódové slovo. Odtud dostáváme

$$\frac{1}{q^{n-k}} \left(\sum_{\mathrm{s}\in\mathbb{F}_q^{n-k}} \mathrm{d}\left(\overline{0}, \mathfrak{e}(\mathrm{s})\right) \right) = \frac{1}{q^{n}} \left(\sum_{\mathrm{s}\in\mathbb{F}_q^{n-k}} q^k \ \mathrm{d}\left(\overline{0}, \mathfrak{e}(\mathrm{s})\right) \right).$$

Dále podle 1.6.18 víme, že

$$\forall x \in C(\mathrm{s}) : \mathrm{d}\left(\overline{0}, \mathfrak{e}(\mathrm{s})\right) = \mathrm{d}(x, C), \qquad |C(\mathrm{s})| = q^k.$$

Využitím předchozího řádku dostáváme

$$\mathrm{E}(\mathrm{d}(x,y)) = \frac{1}{q^{n}} \left(\sum_{\mathrm{s}\in\mathbb{F}_q^{n-k}} \sum_{x\in C(\mathrm{s})} \mathrm{d}(x, C) \right).$$

Tímto způsobem prochází x všechny prvky prostoru $\mathbb{F}_q^n$. Proto dostáváme

$$\mathrm{E}(\mathrm{d}(x,y)) = \frac{1}{q^{n}} \left(\sum_{x\in\mathbb{F}_q^{n}} \mathrm{d}(x, C) \right) \stackrel{def}{=} {}_q\mathrm{R}_\mathrm{a}.$$

Konečně jsme se tedy dostali k dokazovanému tvaru efektivity pro q–ární případy

$$\mathrm{e} = \frac{(n-k)\log_2 q}{{}_q\mathrm{R}_\mathrm{a}}.$$

□

Můžeme poznamenat, že pokud ale platí buď $q \in \{2, 3\}$ nebo $\mathrm{d}(x, y) = \vartheta(x, y)$, dostáváme rovnost ${}_q\mathrm{R}_\mathrm{a} = \mathrm{R}_\mathrm{a}$ a tvrzení je mnohem podobnější tvrzení 2.4.1.

Poslední ukazatel, který zbývá odhadnout, je míra změny. Ta se podle definice 1.3.5 rovná

$$\rho = \mathrm{E}\left(\frac{\mathrm{d}_\mathrm{H}(x, y)}{n}\right).$$

Jak již ale bylo poznamenáno výše, při vkládání se podle 1.6.20 učiní nejvýše R změn. Tím tedy získáváme horní odhad míry změny $\rho \leq \frac{\mathrm{R}}{n}$. Jak je snadno vidět z kapitoly 1, nebere tento ukazatel v potaz, o kolik se původní nosič na daném místě změnil, ale pouze normalizuje počet provedených změn nosiče. Bližší výpočet míry změny není v plné obecnosti možný a hodně záleží na použitém samoopravném kódu. Například u Hammingových kódů, jejichž příklad si ukážeme v sekci 2.5, se konkrétní hodnota počítá poměrně snadno, proto budeme moci určit konkrétní hodnoty.

Pokud se ještě zaměříme na Hammingovy kódy, které jsou podle [12] $[\frac{q^r-1}{q-1}, \frac{q^r-1}{q-1} - r]_q$ kódy, můžeme si všimnout, že s rostoucím r roste i efektivita. Na druhou stranu ale klesá relativní kapacita nosiče. Dokážeme tedy ukrýt zprávu efektivněji, ale potřebujeme k tomu delší nosič.

2.5 Příklad

V této sekci si předvedeme použití výše popsaného stegosystému v několika konkrétních případech a spočítáme vylepšení, kterého v těchto případech docílíme.

Začneme s binárními Hammingovými kódy. Konkrétně s $[7, 4]_2$ kódem $\mathcal{H}_3$.

Příklad 2.5.1. Hammingův kód $\mathcal{H}_3$ je perfektní kód, jehož paritní matice má tvar

$$\mathrm{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Jelikož o Hammingových kódech víme, že jsou perfektní a z paritní matice snadno zjistíme, že minimální vzdálenost $\mathrm{d}_C = 3$, dostaneme z definice 1.6.9, že $\mathrm{R} = 1$.

Odesilatel a příjemce mají tedy domluvenou matici H. Pro tento příklad použijeme jako nosič, do kterého budeme vkládat,

$$x = (152, 36, 63, 72, 150, 4, 213, 248, 193, 21, 7, 105, 50, 102).$$

Všimněme si, že tento nosič má délku $14 = 2 \cdot 7 = l \cdot n$. V našem případě tedy bude $x \in \mathbb{Z}_{256}^{14}$. Zároveň si můžeme snadno ověřit, že vybraný nosič je vhodný. Pro $q = 2$ je totiž vhodný každý nosič.

Pro vkládání jsme zvolili zprávu

$$m = (0, 0, 1, 1, 0, 1),$$

která má délku $6 = 2 \cdot 3 = l(n-k)$. Zpráva má tedy požadovaný tvar, protože $m \in \mathbb{F}_2^6$, přičemž máme umluvený $[7, 4]_2$ kód.

Pro větší přehlednost uvedeme hodnoty jednotlivých proměnných, které využíváme. Máme

$$n = 7, \qquad k = 4, \qquad l = 2, \qquad p = 256, \qquad q = 2.$$

Nejprve provedeme přípravný proces a získáme dva bloky nosiče x_1, x_2, které mají délku $n = 7$, a dva bloky zprávy m_1, m_2, které mají délku $(n - k) = 3$.

Po výpočtu přípravného procesu zjistíme, že bloky nosiče mají tvar

$$x_1 = (0, 0, 1, 0, 0, 0, 1)^\top,$$

$$x_2 = (0, 1, 1, 1, 1, 0, 0)^\top$$

a bloky zprávy mají tvar

$$m_1 = (0, 0, 1)^\top, \qquad m_2 = (1, 0, 1)^\top.$$

Dokončili jsme přípravnou fázi a dostáváme se k vlastnímu vkládacímu algoritmu. Spočítáme jednotlivá s_i pro $i \in \{1, 2\}$, které budeme dále používat jako syndromy.

$$\mathrm{s}_1 = m_1 - \mathrm{H}x_1 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\mathrm{s}_2 = m_2 - \mathrm{H}x_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

Dále spočítáme reprezentanty jednotlivých rozkladových tříd. Jak jsme již podotkli na straně 33, Hammingův kód má krycí poloměr $\mathrm{R} = 1$. Z tohoto důvodu je hledání reprezentanta rozkladové třídy jednoduché. Podle 1.6.20 totiž dostaneme každý syndrom jako jeden sloupec paritní matice H. V našem případě je s_1 rovno poslednímu sloupci matice H, dostáváme tedy reprezentanta rozkladové třídy

$$\mathfrak{e}(\mathrm{s}_1) = (0, 0, 0, 0, 0, 0, 1)^\top.$$

V případě s_2 nám obdobným postupem vyjde

$$\mathfrak{e}(\mathrm{s}_2) = (0, 0, 0, 0, 1, 0, 0)^\top.$$

Máme tedy změny

$$e_1 = (0,0,0,0,0,0,1)^\top, \qquad e_2 = (0,0,0,0,1,0,0)^\top,$$

které použijeme v dokončovacím procesu.

V poslední fázi spočítáme výsledný stegoobjekt. Podle definice máme nejdříve

$$e = e_1^\top \parallel e_2^\top = (0,0,0,0,0,0,1,0,0,0,0,1,0,0).$$

Výslednou chybu teď použijeme ke konečné modifikaci původního nosiče, a tím vytvoříme požadovaný stegoobjekt

$$y = x + {}_q\pi_p(e),$$

kde

$$x = (152,36,63,72,150,4,213,248,193,21,7,105,50,102),$$

$${}_q\pi_p(e) = (0,0,0,0,0,0,1,0,0,0,0,1,0,0).$$

Nakonec tedy dostaneme

$$y = (152,36,63,72,150,4,214,248,193,21,7,106,50,102).$$

Nyní se na celou situaci podíváme jako příjemce a zkusíme z přijatého stegoobjektu extrahovat zprávu.

Na obdržené y provedeme přípravnou fázi, a tím získáme $l = 2$ částí délky $n = 7$.

$$y_1 = (0,0,1,0,0,0,0)^\top$$

$$y_2 = (0,1,1,1,0,0,0)^\top$$

Nyní provedeme hlavní část extrakčního algoritmu a vypočítáme jednotlivé části zprávy jako syndromy slov y_i, čili

$$m_i = \mathrm{H}y_i.$$

Tím dostaneme

$$m_1 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \qquad m_2 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}.$$

Teď už jen stačí udělat poslední část extrakce a spojit jednotlivé transponované části zprávy. Tím získáme zprávu

$$m = (0,0,1,1,0,1).$$

Právě jsme dokončili extrakci a výsledná zpráva se skutečně shoduje s původně odesílanou zprávou. Tím jsme předvedli funkčnost a použití maticového způsobu vkládání pro konkrétní případ.

Nyní se podíváme, jakého zlepšení jsme pomocí této metody dosáhli v této konkrétní situaci.

Právě předvedený stegosystém vkládá $(n - k) = 3$ bitové zprávy do bloků délky $n = 7$. Ke vkládání přitom používáme nejvýše jedné změny na každý blok nosiče. Když tedy dosadíme do definice 1.3.3 příslušné hodnoty, dostaneme relativní kapacitu nosiče

$$\alpha = \frac{3}{7}.$$

V každém bloku nosiče měníme, až na jediný případ, právě jeden prvek. Speciální případ nastane v situaci, kdy se syndrom bloku nosiče již shoduje s blokem vkládané zprávy. Nosič tedy neměníme pouze v případě, že

$$\mathrm{H}x_i = m_i.$$

V jednom případě tedy děláme nula změn a právě jednu změnu děláme ve zbylých $2^3 - 1$ případech. Po dosazení do definice 1.3.6 tedy získáme efektivitu

$$\mathrm{e} = \frac{3}{\frac{2^3-1}{2^3}} = \frac{24}{7},$$

což přesně odpovídá vzorci odvozenému v tvrzení 2.4.1, protože v tomto případě se $\mathrm{R_a} = \frac{7}{8}$.

Míru změny také snadno vypočítáme přímo z definice jako

$$\rho = \frac{\frac{2^3-1}{2^3}}{7} = \frac{1}{8}.$$

♣

Celkem jsme tedy oproti základním steganografickým metodám, které jsme popsali v 1.5, získali mnohem lepší výsledky.

Zatím jsme vypočítali vylepšení jen pro Hammingův kód $\mathcal{H}_3$. Parametry se ale u binárních Hammingových kódů dají snadno určit obecně. Tak jako jsme určili parametry Hammingova kódu $\mathcal{H}_3$, můžeme obdobným způsobem vypočítat parametry i pro ostatní binární Hammingovy kódy $\mathcal{H}_r$ s parametry $[2^r - 1, 2^r - r - 1]_2$.

Po krátké úvaze získáme obecné vyjádření efektivity pro binární Hammingovy kódy

$$\mathrm{e} = \frac{r}{1 - 2^{-r}}.$$

Podobným způsobem můžeme určit obecný vzorec pro relativní kapacitu binárních Hammingových kódů jako

$$\alpha = \frac{r}{2^r - 1}.$$

Pro několik hodnot r jsou parametry shrnuty v tabulce 2.1. Můžeme si všimnout, že s rostoucím r se zvyšuje efektivita, což je přesně to, co bylo naším cílem. Na druhou stranu se nám s rostoucím r snižuje relativní kapacita. Pro vkládání zprávy tedy musíme používat delší nosiče. V tabulce si také můžeme všimnout, že první řádek pro $r = 1$ má shodné parametry jako LSB vkládání popsané v 1.5. Je to z toho důvodu, že maticové vkládání se pro případ $q = 2$ a $r = 1$ naprosto s touto metodou shoduje. Dosadíme-li totiž za r jedničku, dostaneme jako parametry kódu $n = 1$, $k = 0$ a dále už je snadné nahlédnout, že se postup maticového vkládání v tomto případě shoduje s LSB vkládáním.

r	α	e
1	1,0	2,0
2	0,66	2,66
3	0,43	3,43
4	0,27	4,27
5	0,16	5,16
6	0,09	6,09
10	0,01	10,01

Tabulka 2.1: Parametry stegosystémů vytvořených z kódů $\mathcal{H}_r$.

Příklad 2.5.2. V našem druhém příkladě ukážeme jednoho zástupce stegosystémů založených na nebinárních kódech. Konkrétně to bude q–ární Hammingův kód. Nebudeme zde už dopodrobna rozebírat vkládací a extrakční algoritmus, jen se podíváme na vlastnosti, které tyto stegosystémy mají. Podle nich potom rozhodneme zda si nějakým způsobem polepšíme, pokud dané stegosystémy použijeme.

Jak jsme již poznamenali výše, Hammingovy kódy mají obecně parametry $\left[\frac{q^r-1}{q-1}, \frac{q^r-1}{q-1} - r\right]_q$. Vkládáme tedy r q–árních symbolů do $\frac{q^r-1}{q-1}$ pixelů, přičemž podle [3] a [7] mají q–ární Hammingovy kódy, stejně jako ty binární, krycí poloměr R = 1. Proto tedy provádíme maximálně jednu změnu velikosti $\lceil \frac{q}{2} \rceil$.

Ke změně nedochází pouze v jednom případě, kdy se syndrom určité části nosiče již shoduje s příslušnou částí zprávy. Ve zbylých $q^r - 1$ případech děláme právě jednu změnu. Podle [2] tedy máme kapacitu nosiče

$$\alpha = \frac{r \log_2 q}{\frac{q^r-1}{q-1}} = \frac{r(q-1)\log_2 q}{q^r - 1}$$

a následně i efektivitu

$$\mathrm{e} = \frac{r \log_2 q}{1 - q^{-r}}.$$

Jak jsme ale již poznamenali na straně 10, u efektivity záleží na uvažované definici distorze. Nutno tedy poznamenat, že se v tomto případě bere jako distorze $\mathrm{d}(x, y) = \vartheta(x, y)$. V opačném případě bychom došli ke sporu s tvrzením 2.4.2.

Pro Hammingův kód $\mathcal{H}_{3,3}$, který je ternární a kde $r = 3$, tedy dostáváme

$$\mathrm{e} \approx 4,938 \qquad\qquad \alpha \approx 0,366.$$

V tomto případě nezáleží na uvažované distorzi, neboť $q = 3$. Tím jsme ovšem ukázali, že tento stegosystém má ještě větší efektivitu, než stegosystém popsaný v 2.5.1. Tento stegosystém je tedy zatím nejefektivnější popsaný algoritmus.

♣

Pokud bychom tedy na závěr srovnávali poslední popsaný stegosystém s tabulkou 2.1, je vidět, že tento q–ární kód má pro srovnatelnou kapacitu nosiče vyšší efektivitu. Jak se ukazuje v [2], s rostoucím q se efektivita stále zvyšuje. Musíme ale poznamenat, že rostoucí q přiměřeně zvyšuje i distorzi, neboť provádíme čím dál tím větší změny. Diskuze zohledňující oba tyto faktory je k nahlédnutí v [2]. Tam se také ukazuje, že nejvhodnější q je z tohoto hlediska $q = 3$. V tom případě nám také mizí komplikace s uvažováním různých definic distorze. Pro $q = 3$ se totiž všechny definice rovnají. Protože je tedy nejvýhodnější používat $q = 3$, bude se toho následující stegosystém snažit využít.

Kapitola 3

SDCS

Zkratka SDCS vznikla z anglického spojení „Sum and Difference Covering Set“, my budeme používat český ekvivalent *součtové a rozdílové pokrývací množiny*. V této kapitole si nejdříve popíšeme, co tyto množiny jsou a následně se je budeme snažit využít pro konstrukci nových stegosystémů.

Stejně jako v kapitole 2 nebudeme brát nosič po jednotlivých prvcích, ale rozdělíme si jej na bloky určité délky. Tento způsob ukrývání navazuje na článek [1] o pokrývacích množinách a je vlastně vylepšením metody z předchozí kapitoly. Proto se budeme snažit využít výsledků, kterých dosáhla předchozí metoda, a přidat nějaká další vylepšení. Na rozdíl ale od stegosystémů popsaných v kapitole 2 se zde zaměříme na vložení velkého množství informace do nosiče za použití malých změn. Budeme tedy oproti kapitole 2 dostávat méně efektivní stegosystémy s vyšší relativní kapacitou. Stejně jako v předchozím případě tím dosáhneme určitého zlepšení oproti 1.5.

3.1 Příklad

V této sekci si ukážeme, jaké možnosti nám nabízí dělení nosiče na bloky. Později se pokusíme toto pozorování rozšířit obecněji.

Jelikož jsme v kapitole 2 zjistili, že je výhodné měnit nosič pouze o ± 1, využijeme tento koncept i zde.

Všimněme si, že v grupě $\mathbb{Z}_5$ můžeme pomocí přičítání nebo odčítání buď jedničky nebo dvojky přejít z libovolného čísla na libovolné jiné číslo. Pokud budeme chtít například z čísla 4 přejít na číslo 1, stačí přičíst dvojku. Jestliže bychom chtěli místo jedničky dostat číslo 3, stačí odečíst 1. Tímto způsobem můžeme dostat libovolné číslo ze $\mathbb{Z}_5$. Pokud si nosič grupy $\mathbb{Z}_5$ představíme jako množinu $\{-2, -1, 0, 1, 2\}$, bude nám výše popsaná vlastnost zřejmá okamžitě. Této skutečnosti se budeme snažit využít. Spojíme ji ještě s předchozí vlastností, musíme tedy k prvkům nosiče přičítat pouze ± 1.

Jako u každého stegosystému budeme i zde chtít nějakým způsobem extrahovat zprávu ukrytou ve stegoobjektu. Většinou však tento způsob nezávisí na vložení zprávy, což znamená, že každý nosič nese nějakou zprávu, i když do něj nic neukryjeme. Tato zpráva je ale pravděpodobně jiná, než jakou bychom chtěli. Z toho důvodu musíme nějakým způsobem do nosiče vložit změny, které se mají na implicitně obsažené zprávě provést. Pokud se nám to povede, příjemce ze stegoobjektu extrahuje zprávu, kterou jsme zamýšleli poslat.

My vyjdeme z našeho pozorování a budeme předpokládat, že prvky zprávy, kterou chceme poslat, jsou z grupy $\mathbb{Z}_5$. Stejně tak chceme, aby i prvky implicitní zprávy byly ze stejné grupy. To ale záleží na způsobu extrakce, proto si později extrakční algoritmus přizpůsobíme tak, abychom dostávali prvky $\mathbb{Z}_5$.

Nyní nám stačí popsat změny, které musíme na implicitní zprávě provést, abychom dostali námi požadovanou zprávu. Pokud tedy popíšeme změny, které jsou nutné provést, jako dvojici $(\overline{s_1}, \overline{s_2})$, kde $\overline{s_1}$ značí koeficient u jedničky a $\overline{s_2}$ značí koeficient u dvojky, přičemž $\overline{s_1}, \overline{s_2} \in \{-1, 0, 1\}$, jsme si podle předchozího pozorování jisti, že dokážeme popsat libovolnou změnu a tím pádem i poslat požadovanou zprávu. Navíc víme, že z dvojice $(\overline{s_1}, \overline{s_2})$ bude vždy nejvýše jeden prvek nenulový.

Jelikož má naše zpráva být z grupy $\mathbb{Z}_5$, budeme tomu přizpůsobovat i extrakční algoritmus. Jednotlivé prvky nosiče si tedy převedeme na prvky $\mathbb{Z}_5$. Protože jsme rozhodli, že budeme změny popisovat jako dvojici koeficientů u jedničky a dvojky, rozdělíme si celý nosič na takovéto dvojice. První z dvojice pak bude vyjadřovat koeficient u jedničky a druhý prvek bude koeficientem u dvojky. Extrakce tedy proběhne tím způsobem, že příjemce přičte první prvek dvojice k dvojnásobku druhého prvku. To vše proběhne v grupě $\mathbb{Z}_5$. Tím pádem může odesilatel jednoduše pozměnit výslednou zprávu. Pokud totiž není spokojený s nějakým prvkem posílané zprávy, upraví pouze jeden z prvků příslušné dvojice o ± 1, což při extrakci změní původní prvek zprávy na zamýšlenou hodnotu.

Jak jsme již uvedli výše, víme, že nejvýše jeden prvek dvojice $(\overline{s_1}, \overline{s_2})$ bude nenulový. To je další vlastnost, které jsme chtěli dosáhnout, neboť tímto způsobem při vkládání změníme nejvýše jeden prvek nosiče ze dvou a jeho hodnotu, jak již bylo řečeno, navíc změníme maximálně o jedna. Tato skutečnost se pozitivně projeví na nedetekovatelnosti stegosystému a jeho relativní kapacitě.

Všech těchto vlastností jsme dosáhli především vhodnou volbou čísel, jejichž koeficienty ve výsledku měníme. Tato volba je samozřejmě součástí tajného klíče a později se stane nejdůležitější částí stegosystému.

V tomto příkladu tedy budeme brát dvojice prvků nosiče a z nich budeme měnit nejvýše jeden prvek právě o jedna. Jelikož budeme potřebovat možnost měnit každý prvek o jedna na obě strany, budeme jako vhodné

nosiče brát pouze takové posloupnosti prvků, z nichž každý splňuje

$$\widetilde{x_i} > 0, \qquad \widetilde{x_i} < p - 1, \qquad \widetilde{x_i} \in \mathbb{Z}_p,$$

kde nosiče jsou obecně posloupnosti prvků ze $\mathbb{Z}_p$.

Odesilatel a příjemce jsou jako obvykle domluveni na způsobu vkládání a extrakce.

3.1.1 Vkládání

Jako odesilatel ukryjeme zprávu m do nosiče x následujícím způsobem:

- Vybereme si vhodný nosič $x = (\widetilde{x}_1, \widetilde{x}_2, \ldots, \widetilde{x}_n)$, kde $\widetilde{x}_i \in \mathbb{Z}_{256}$ a n je sudé. Ten si rozdělíme na bloky délky 2. Tím získáme

  $$x = (\widetilde{x}_1, \widetilde{x}_2) \,\|(\widetilde{x}_3, \widetilde{x}_4) \,\| \ldots \|(\widetilde{x}_{2l-1}, \widetilde{x}_{2l}),$$

  kde $l = \frac{n}{2}$.

- Zprávu m si zvolíme jako

  $$m = (m_1, m_2, \ldots, m_l),$$

  kde $m_i \in \mathbb{Z}_5$.

- Pro $i \in \{1, 2, \ldots, n\}$ spočteme x_i jako

  $$x_i = \widetilde{x}_i \bmod 5.$$

  Dostáváme tedy dvojice (x_{2j-1}, x_{2j}) pro $j \in \{1, 2, \ldots, l\}$, kde každé $x_i \in \mathbb{Z}_5$, pro $i \in \{1, 2, \ldots, n\}$.

- Pro $i \in \{1, 2, \ldots, l\}$ definujeme

  $$t_i = x_{2i-1} + 2\,(x_{2i}) \bmod 5.$$

  Jednotlivá t_i, pro $i \in \{1, 2, \ldots, l\}$ představují prvky implicitně vložené zprávy. My nyní potřebujeme definovat změny

  $$s_i = ((s_i)_1\,, (s_i)_2)\,,$$

  které je nutné provést na jednotlivé části původního nosiče $\widetilde{x}_i$, abychom po jejich aplikaci získali po extrakci požadovanou zprávu. Příslušné hodnoty změn můžeme najít pomocí tabulky 3.1, když vyhledáme buňku, odpovídající implicitnímu prvku zprávy t_i a prvku požadované zprávy m_i.

- Pro všechna $i \in \{1, 2, \ldots, l\}$ vytvoříme jednotlivé části stegoobjektu

$$(\widetilde{y}_{2i-1}, \widetilde{y}_{2i}) = s_i + (\widetilde{x}_{2i-1}, \widetilde{x}_{2i})$$

  tak, že je pozměníme o hodnoty změn s_i.

- Vytvoříme výsledný stegoobjekt

$$y = (\widetilde{y}_1, \widetilde{y}_2) \,\|(\widetilde{y}_3, \widetilde{y}_4) \,\| \ldots \|(\widetilde{y}_{2l-1}, \widetilde{y}_{2l}),$$

  kde $l = \frac{n}{2}$.

$t_i \setminus m_i$	0	1	2	3	4
0	(0 , 0)	(1 , 0)	(0 , 1)	(0 ,−1)	(−1, 0)
1	(−1, 0)	(0 , 0)	(1 , 0)	(0 , 1)	(0 ,−1)
2	(0 ,−1)	(−1, 0)	(0 , 0)	(1 , 0)	(0 , 1)
3	(0 , 1)	(0 ,−1)	(−1, 0)	(0 , 0)	(1 , 0)
4	(1 , 0)	(0 , 1)	(0 ,−1)	(−1, 0)	(0 , 0)

Tabulka 3.1: Změny s_i pro vkládání z příkladu 3.1.

Takto vytvořené y pošleme příjemci.

3.1.2 Extrakce

Příjemce se bude snažit z obdrženého stegoobjektu extrahovat zprávu m tím, že převede dvojice prvků stegoobjektu na prvky grupy $\mathbb{Z}_5$. Jednotlivé dvojice bude brát jako vektory koeficientů pro domluvenou množinu $\{1, 2\}$. Celkem tedy provede následující postup:

- Příjemce rozdělí přijaté

$$y = (\widetilde{y}_1, \widetilde{y}_2, \ldots, \widetilde{y}_n)$$

  na bloky délky 2, čímž získá dvojice $(\widetilde{y}_{2i-1}, \widetilde{y}_{2i})$, kde $i \in \{1, 2, \ldots, l\}$. Přitom můžeme poznamenat, že l je opět rovno $\frac{n}{2}$ a $y_j \in \mathbb{Z}_{256}$ pro $j \in \{1, 2, \ldots, n\}$.

- Dále pro $j \in \{1, 2, \ldots, n\}$ definuje

$$y_j = \widetilde{y}_j \bmod 5.$$

  Má tedy dvojice (y_{2i-1}, y_{2i}), pro $i \in \{1, 2, \ldots, l\}$, kde $y_j \in \mathbb{Z}_5$, pro $j \in \{1, 2, \ldots, n\}$.

  Zatím tedy pouze vyjádřil stegoobjekt y jako l dvojic prvků z grupy $\mathbb{Z}_5$.

- V této části příjemce převede prvky $\mathbb{Z}_5^2$ na prvky grupy $\mathbb{Z}_5$, které budou vyjadřovat jednotlivé prvky zprávy. Připravenou dvojici použije jako vektor koeficientů. Jak jsme již ukázali výše, každý prvek $\mathbb{Z}_5$ lze zapsat pomocí takovéto dvojice, kde je nejvýše jeden prvek nenulový a velikost takového vektoru je nejvýše jedna. Odesilatel tedy dokázal změnit libovolný prvek implicitně vložené zprávy, který byl prvkem $\mathbb{Z}_5$, na požadovanou hodnotu ze $\mathbb{Z}_5$ tím, že přičetl příslušný prvek $\mathbb{Z}_5$ v podobě vektoru koeficientů popsaného tvaru. To zajistilo, že příjemce extrahuje prvky zprávy, které odesilatel zamýšlel.

  Příjemce tedy získá jednotlivé části zprávy m jako

$$m_i = \mathrm{Ext}(y_{2i-1}, y_{2i}) = y_{2i-1} + 2\,(y_{2i}) \bmod 5.$$

- Nyní už jen poskládá celou původní zprávu

$$m = (m_1, m_2, \ldots, m_l),$$

  kde pro $i \in \{1, 2, \ldots, l\}$ je $m_i \in \mathbb{Z}_5$.

3.1.3 Vlastnosti

Právě předvedený stegosystém využívá toho, že v grupě $\mathbb{Z}_5$ můžeme pomocí přičtení nebo odečtení buď jedničky nebo dvojky dosáhnout libovolného prvku této grupy. Tuto vlastnost budeme chtít v této kapitole využít obecněji a použít ji ke konstrukci efektivních stegosystémů.

Abychom mohli spočítat vlastnosti právě předvedeného stegosystému, musíme si uvědomit několik věcí.

1. Do každé dvojice prvků stegoobjektu vkládáme právě jeden prvek $\mathbb{Z}_5$, což odpovídá právě $(\log_2 5)$ bitům.

2. Při vkládání neprovádíme ve zvolené dvojici žádnou změnu v jednom z pěti případů.

3. V ostatních případech děláme právě jednu změnu.

4. Provádíme pouze ± 1 změny, proto se nemusíme zabývat tím, jakou definici distorze používáme. V případě, že jsou prováděné změny pouze velikosti jedna, jsou, jak jsme již zmínili výše, všechny zmiňované definice totožné.

Z předchozích bodů můžeme jednoduše odvodit hodnotu relativní kapacity nosiče, velikost míry změny a efektivitu.

$$\alpha = \frac{\log_2 5}{2} \approx 1,16 \qquad \rho = \frac{\frac{4}{5}}{2} = \frac{2}{5} \qquad \mathrm{e} = \frac{\log_2 5}{\frac{4}{5}} \approx 2,9$$

Právě odvozená čísla nám ukazují, že popsaný stegosystém je opět lepší, než stegosystém v sekci 1.5.

Funkčnost tohoto konkrétního stegosystému jsme již předvedli v průběhu vkládání a extrakce.

3.2 Zobecnění

Zde zobecníme předchozí metodu vkládání a extrakce.

Jak jsme již zmínili, omezíme se na měnění původního nosiče pouze o ± 1. Díky tomu dosáhneme malých a tedy i méně detekovatelných změn, což je samozřejmě žádoucí.

Abychom dosáhli podobných výsledků jako v sekci 3.1, definujeme si pokrývací množinu, která nám pomůže pomocí relativně málo změn pokrýt co největší množinu.

Definice 3.2.1. *Množinu* $\mathrm{A} = \{a_1, a_2, \ldots, a_n\}$, *kde* $\mathrm{A} \subset \mathbb{Z}_M$ *nazveme* (n, k, M) SDCS *neboli* součtovou a rozdílovou pokrývací množinou*, pokud pro všechny* $z \in \mathbb{Z}_M$ *existuje* n*–tice*

$$s = (\overline{s_1}, \overline{s_2}, \ldots, \overline{s_n}),$$

kde $\overline{s_i} \in \{-1, 0, 1\}$ *pro* $i \in \{1, 2, \ldots, n\}$, *která splňuje*

$$\sum_{i=1}^{n} \overline{s_i} a_i \equiv z \pmod{M},$$

přičemž $\sum_{i=1}^{n} |\overline{s_i}| \leq k$.

Pro lepší představu poznamenáme, že v sekci 3.1 jsme již součtovou a rozdílovou pokrývací množinu použili. Množina A byla $(2, 1, 5)$ SDCS a rovnala se $\mathrm{A} = \{1, 2\}$. Existenci všech s jsme již ukázali v předchozím příkladě a případně ji můžeme ověřit v tabulce 3.1, kde hledané dvojice jsou uvedené v prvním řádku. Pomocí této součtové a rozdílové pokrývací množiny jsme tedy dokázali zapsat libovolný prvek $\mathbb{Z}_5$ vektorem délky 2 a váhy nejvíce 1.

Nyní si pomocí (n, k, M) součtové a rozdílové pokrývací množiny A pokusíme sestrojit stegosystém.

Jako část klíče bude samozřejmě sloužit SDCS množina A. Budeme tedy předpokládat, že odesilatel a příjemce mají stejnou množinu. Zároveň se obě komunikující strany musejí domluvit na pořadí prvků množiny A. My budeme pro jednoduchost předpokládat, že jsou prvky uspořádány podle velikosti, tedy

$$a_1 < a_2 < \ldots < a_n.$$

3.2.1 Matematizace

Nejprve si, jako v kapitole 2, popíšeme přípravný a dokončovací proces, abychom se později mohli zaměřit na hlavní část algoritmu.

Jak již bylo řečeno v 3.1, nosič x budeme označovat jako vhodný právě tehdy když, každý jeho prvek $\widetilde{x}$ splňuje

$$\widetilde{x} > 0, \qquad \widetilde{x} < p - 1, \qquad \widetilde{x} \in \mathbb{Z}_p,$$

kde nosiče jsou obecně posloupnosti prvků ze $\mathbb{Z}_p$ a $p \geq 3$.

Dále budeme předpokládat, že nosič $x \in \mathbb{Z}_p^*$ je délky nl, pro nějaké $l \in \{1, 2, \ldots\}$, kde A je (n, k, M) SDCS a $p \geq 3$.

O zprávě m, kterou budeme posílat příjemci, budeme předpokládat, že je to posloupnost prvků ze $\mathbb{Z}_M$ délky l. Zprávu tedy budeme uvažovat jako

$$m = (m_1, m_2, \ldots, m_l) \in \mathbb{Z}_M^l.$$

Pro ulehčení zápisu si ještě definujeme sloupcový vektor a, který vytvoříme z prvků součtové a rozdílové pokrývací množiny A. Pokud je tedy

$$\mathrm{A} = \{a_1, a_2, \ldots, a_n\},$$

pak jako vektor a budeme označovat vektor délky n, s prvky z množiny A uspořádanými podle velikosti.

$$a = (a_1, a_2, \ldots, a_n)^\top$$

My budeme v přípravném procesu nosič převádět na l sloupcových vektorů délky n, jejichž prvky jsou ze $\mathbb{Z}_M$. Jsou pro nás tedy vhodné některé definice ze sekce 2.1.

Definice 3.2.2 (Přípravný proces)**.** *Pro nosič $x \in \mathbb{Z}_p^{nl}$ a množinu* A*, která je (n, k, M) SDCS, kde $p \geq 3$ definujeme* přípravný proces $\mathrm{PRI}_M^n(x)$ *jako*

$$\mathrm{PRI}_M^n(x) = (x_1, x_2, \ldots, x_l)\,,$$

kde

$$x_i = \left[{}_p\pi_M(x)\right]_i^n\,,$$

pro $i \in \{1, 2, \ldots, l\}$.

Přípravný proces tedy v tomto případě rozdělí nosič na l n–tic, kde každou n–tici bereme jako sloupcový vektor, přičemž všechny prvky jsou ze $\mathbb{Z}_M$. Máme tedy

$${}_p\pi_M(x) = x_1^\top \parallel x_2^\top \parallel \ldots \parallel x_l^\top.$$

Můžeme poznamenat, že každý prvek $x_i \in \mathbb{Z}_M^n$ je sloupcový vektor délky n, tedy vektor stejného typu jako vektor a.

Nyní ještě definujeme dokončovací proces, abychom se později mohli soustředit na nejdůležitější část vkládání.

Z hlavní části vkládacího algoritmu dostaneme l vektorů délky n, které budeme značit $s_1, s_2, \ldots, s_l$. Jednotlivé prvky těchto vektorů budou ze $\mathbb{Z}_3$. Tyto vektory představují změny, které je nutno na původním nosiči provést, abychom získali stegoobjekt nesoucí naší zprávu. My si v dokončovacím procesu ukážeme, jakým způsobem tyto vektory využít a jak sestrojit výsledný stegoobjekt.

Definice 3.2.3 (Dokončovací proces)**.** *Nechť máme nosič $x \in \mathbb{Z}_p^{nl}$, kde $p \geq 3$, a množinu* A*, která je (n, k, M) SDCS. Nechť dále máme l sloupcových vektorů změn $s_1, s_2, \ldots, s_l$, kde*

$$s_i = \left((s_i)_1, (s_i)_2, \ldots, (s_i)_n\right)^\top$$

pro $i \in \{1, 2, \ldots, l\}$ a nechť $(s_i)_j \in \mathbb{Z}_3$ pro $j \in \{1, 2, \ldots, n\}$. Pak definujeme výslednou změnu *s jako*

$$s = s_1^\top \parallel s_2^\top \parallel \ldots \parallel s_l^\top.$$

Dále definujeme dokončovací proces *jako*

$$\mathrm{DOK}_p^l(x, s_1, s_2, \ldots, s_l) = x + {}_3\pi_p(s),$$

kde sčítání vektorů bereme po složkách.

Právě dokončená definice využívá projekci ${}_3\pi_p(s)$, kterou jsme definovali v sekci 2.1. Připomeňme, že v tomto případě předpokládáme, že $\mathbb{Z}_3 = \{-1, 0, 1\}$ a že projekce do $\mathbb{Z}_p$ nám zobrazuje

$$-1 \mapsto -1, \qquad 0 \mapsto 0, \qquad 1 \mapsto 1.$$

Dokončili jsme matematizaci a nyní se můžeme zaměřit na popis samotného vkládacího algoritmu.

3.2.2 Vkládání

V této části se budeme snažit upravit nosič tak, aby z něj byl příjemce schopen extrahovat ukrytou zprávu.

- Provedeme volbu zprávy a nosiče, který připravíme na proces samotného určení změny.
  - Nejprve vybereme vhodný nosič $x \in \mathcal{C}$ délky ln takový, že $x \in \mathbb{Z}_p^{ln}$, kde $l \in \{1, 2, \ldots\}$. Přitom předpokládáme, že jsme se předem s příjemcem zprávy tajně domluvili na množině A, což je (n, k, M) SDCS a že $p \geq 3$.

– Vybereme zprávu $m \in \mathbb{Z}_M^l$, kterou chceme vložit do nosiče.
– Na vybraný nosič provedeme přípravný proces.

$$\mathrm{PRI}_M^n(x) = (x_1, x_2, \ldots, x_l)$$

Tím získáme prvky

$$x_i = ((x_i)_1, (x_i)_2, \ldots, (x_i)_n)$$

pro každé $i \in \{1, 2, \ldots, l\}$, které budeme používat v další části vkládání.

- Provedeme výpočet změn, které se musejí na nosič provést, abychom dostali požadovaný stegoobjekt.
  – Pro $i \in \{1, 2, \ldots, l\}$ spočteme w_i jako

$$w_i = m_i - (x_i \cdot a) \bmod M, \tag{3.1}$$

kde $x_i \cdot a$ značí skalární součin, tedy

$$x_i \cdot a = \sum_{j=1}^{n} (x_i)_j \, a_j.$$

  – Podle definice součtové a rozdílové pokrývací množiny 3.2.1 existují pro všechna w_i, kde $i \in \{0, 1, \ldots, l\}$, taková s_i, kde s_i je sloupcový vektor délky n

$$s_i = ((s_i)_1, (s_i)_2, \ldots, (s_i)_n)^\top$$

takový, že

$$w_i \equiv s_i \cdot a \pmod{M}, \tag{3.2}$$

kde $s_i \cdot a$ značí skalární součin, a zároveň platí

$$\sum_{j=1}^{n} \left|(s_i)_j\right| \leq k.$$

Pokud pro nějaké i existuje více možných s_i, vybereme to, které minimalizuje sumu

$$\sum_{j=1}^{n} \left|(s_i)_j\right|.$$

Pokud je jich takových více, vybereme mezi nimi náhodně.

- Pomocí s_i, která jsme vybrali v minulém kroku, vytvoříme z původního nosiče x stegoobjekt, který nese zvolenou zprávu m.
  – Provedeme dokončovací proces, který jsme definovali v 3.2.3

$$\mathrm{DOK}_p^l(x, s_1, s_2, \ldots, s_l) = y$$

a tím získáme požadovaný stegoobjekt y.

Takto získaný stegoobjekt můžeme nyní poslat příjemci.

3.2.3 Extrakce

Příjemce na druhé straně extrahuje zprávu následujícím způsobem:

- Nejprve si připraví obdržený stegoobjekt použitím přípravného procesu.
  - Předpokládáme, že má příjemce a odesilatel domluvenou množinu A, což je (n, k, M) SDCS, a že přijme stegoobjekt y.
  - Na stegoobjekt provede přípravný proces
$$\mathrm{PRI}_M^n(y) = (y_1, y_2, \ldots, y_l)\,.$$
- Spočítá jednotlivé prvky výsledné zprávy.
  - Použitím součtové a rozdílové pokrývací množiny A vypočítá pro každé $i \in \{1, 2, \ldots, l\}$ jeden prvek zprávy m_i.
$$y_i \cdot a = (x_i + s_i) \cdot a = (x_i \cdot a) + (s_i \cdot a) \overset{(3.2)}{=} (x_i \cdot a) + w_i \overset{(3.1)}{=} m_i$$
- Nakonec složí výslednou zprávu z jednotlivých prvků.
  - Spojí jednotlivé prvky zprávy m_i, kde $i \in \{1, 2, \ldots, l\}$, a získá tak původní zprávu
$$m = m_1 \,\|\, m_2 \,\|\, \ldots \,\|\, m_l.$$

Právě jsme ukázali obecnou konstrukci stegosystémů pomocí součtových a rozdílových pokrývacích množin. Podle [8] se jedná o stejně statisticky nedetekovatelnou metodu vkládání, jako je metoda LSB přizpůsobování, kterou jsme popsali v sekci 1.5 na straně 15.

3.3 Vlastnosti

V této sekci potřebujeme ukázat, že pomocí právě popsané metody můžeme dosáhnout lepších výsledků než v případě 1.5.

Již výše jsme ukázali, že popsaným postupem dokážeme ukrýt a opět extrahovat zprávu. Tím jsme tedy vyřešili funkčnost metody a můžeme se zaměřit na výpočet vlastností.

Z definice 3.2.1 je snadno vidět, že vkládáme prvek grupy $\mathbb{Z}_M$, což odpovídá zprávě velikosti $\log_2 M$ bitů, do nosiče velikosti n a provádíme při tom nejvýše k změn. Tím dostáváme relativní kapacitu

$$\alpha = \frac{\log_2 M}{n}.$$

Pro výpočty dalších vlastností stegosystémů, které byly vytvořeny pomocí právě popsané metody součtových a rozdílových pokrývacích množin, budeme potřebovat následující definici.

Definice 3.3.1. *Mějme $z \in \mathbb{Z}_M$ a (n, k, M) součtovou a rozdílovou množinu* A*. Z definice SDCS víme, že existuje alespoň jedno $s = (\overline{s_1}, \overline{s_2}, \ldots, \overline{s_n})$ takové, že platí*

$$\sum_{i=1}^{n} \overline{s_i} a_i \equiv z \pmod{M},$$

přičemž $\sum_{i=1}^{n} |\overline{s_i}| \leq k$. Pak definujeme $\mathrm{d_A}(z)$ jako

$$\mathrm{d_A}(z) = \sum_{i=1}^{n} |\overline{s_i}| .$$

V případě, že existuje více s, která splňují počáteční podmínku, vybereme to, které předchozí sumu minimalizuje.

Pro další výpočty potřebujeme vyčíslit $\mathrm{E}(\mathrm{d}(x, y))$. Jak jsme již zmínili v úvodu kapitoly, všechny prováděné změny jsou nejvýše velikosti jedna. Proto nám zde nezáleží na tom, jakou definici distorze používáme, vždy se totiž dostaneme ke stejnému výsledku.

$$\mathrm{E}(\mathrm{d}(x, y)) = \frac{1}{M} \sum_{z \in \mathbb{Z}_M} \mathrm{d_A}(z)$$

Proto můžeme vyjádřit efektivitu jako

$$\mathrm{e} = \frac{\log_2 M}{\frac{1}{M} \sum_{z \in \mathbb{Z}_M} \mathrm{d_A}(z)}.$$

Zároveň nosič neměníme jen v jediném z M případů. Míru změny získáme z předchozích úvah snadno.

$$\rho = \frac{\mathrm{E}(\mathrm{d}(x, y))}{n} = \frac{1}{nM} \sum_{z \in \mathbb{Z}_M} \mathrm{d_A}(z)$$

Příklad 3.3.2. Nyní si spočítáme konkrétní hodnotu efektivity pro $(3, 2, 17)$ součtovou a rozdílovou množinu $\mathrm{A} = (1, 2, 6)$.

Nejprve zjistíme, kolik je nutné provést změn pro konkrétní hodnoty z. Vše si vyjádříme přímo:

$$\mathrm{d_A} = \begin{cases} 0 & z = 0 \\ 1 & z \in \{1, 2, 6, 11, 15, 16\} \\ 2 & z \in \{3, 4, 5, 7, 8, 9, 10, 12, 13, 14\} \end{cases}$$

Nyní je již snadné spočítat, že existuje jeden prvek ze $\mathbb{Z}_{17}$, pro který nemusíme dělat žádnou změnu, 6 prvků, pro které musíme udělat jednu změnu

velikosti jedna, a 10 prvků, pro které musíme udělat dvě změny velikosti jedna.

Efektivitu tedy získáváme jako

$$\mathrm{e} = \frac{\log_2 17}{\frac{1}{17}(0 \cdot 1 + 6 \cdot 1 + 10 \cdot 2)} = \frac{17 \log_2 17}{26} \approx 2,673.$$

Vkládáme tedy průměrně $17 \log_2 17 = 4,09$ bitové zprávy do 3 pixelů pomocí průměrně $\frac{0 \cdot 1 + 6 \cdot 1 + 10 \cdot 2}{17} = 1,53$ změn.

Hodnotu relativní kapacity už vyjádříme snadno.

$$\alpha = \frac{\log_2 17}{3} \approx 1,362$$

♣

Efektivita a relativní kapacita několika dalších stegosystémů vytvořených za pomoci součtových a rozdílových pokrývacích množin je popsána v tabulce 3.2.

Poznámka 3.3.3. *Za nejjednodušší stegosystémem, který je vytvořený ze součtové a rozdílové pokrývací množiny, můžeme považovat metodu LSB přizpůsobování, kterou jsme popsali v sekci 1.5. Tento stegosystém může být vytvořen pomocí* $(1, 1, 2)$ *součtové a rozdílové pokrývací množiny* $\mathrm{A} = \{1\}$.

Podle poznámky 3.3.3 je v tabulce 3.2 zaznamenána i metoda LSB přizpůsobování. Můžeme tedy zkontrolovat, že se výsledek výpočtu parametrů stegosystému nijak neliší od hodnot vypočítaných v sekci 1.5. V tabulce si tedy můžeme ověřit, že tímto způsobem dosahujeme stegosystémů s lepšími parametry než v sekci 1.5.

(n, k, M)	A	α	e
$(1, 1, 2)$	$\{1\}$	1,00	2,00
$(3, 2, 17)$	$\{1, 2, 6\}$	1,36	2,67
$(4, 2, 30)$	$\{1, 3, 9, 14\}$	1,22	2,94
$(5, 2, 42)$	$\{1, 2, 7, 14, 18\}$	1,07	3,14
$(9, 2, 132)$	$\{2, 11, 33, 34, 44, 50, 55, 58, 62\}$	0,78	3,81

Tabulka 3.2: Parametry stegosystémů vytvořených ze součtových a rozdílových pokrývacích množin.

3.4 Konstrukce SDCS

Jak jsme zjistili v části 3.3, pomocí součtových a rozdílových pokrývacích množin získáváme stegosystémy s mnohem lepšími vlastnostmi. Stále ale

vycházíme z předpokladu, že nějaké optimální (n, k, M) součtové a rozdílové pokrývací množiny existují. Jak je ale zmíněno v [8], není nijak jednoduché pro všechny parametry n, k, M sestrojit odpovídající SDCS. Pro některé trojice parametrů to zřejmě nejde a pro některé stále nemůžeme existenci potvrdit ani vyvrátit.

V této části se podíváme na nějaké způsoby konstrukce součtových a rozdílových pokrývacích množin.

Je zřejmé, že pokud budeme konstruovat nějakou (n, k, M) součtovou a rozdílovou pokrývací množinu, budeme chtít, abychom pomocí ní mohli vyrobit co možná nejefektivnější stegosystém. Z toho důvodu se budeme snažit pro fixní hodnoty n a k získat SDCS s maximální možnou hodnotou parametru M.

Víme, že M je celkový počet prvků, které můžeme do slova délky n vložit. Ke vkládání používáme nejvýše k změn. Pokud tedy spočítáme počet všech možných změn, které můžeme za daných podmínek udělat, získáme následující odhad.

Tvrzení 3.4.1. *Nechť máme (n, k, M) součtovou a rozdílovou množinu, pak M splňuje*

$$M \leq 2^0\binom{n}{0} + 2^1\binom{n}{1} + \ldots + 2^k\binom{n}{k},$$

pro všechna $n, k \in \{1, 2, \ldots\}$, $n \geq k$.

Důkaz. Každý sčítanec pravé strany je tvaru $2^i\binom{n}{i}$. Pokud bychom chtěli vyjádřit počet prvků, které se od daného slova délky n liší právě v i hodnotách o ± 1, dostaneme právě hodnotu $2^i\binom{n}{i}$. Pokud tedy sečteme všechny prvky od nuly do k, získáme maximální možný počet slov, které lze z daného slova vytvořit.

□

Právě jsme vyčíslili horní odhad, kterému se dále budeme snažit co nejvíce přiblížit.

Definice 3.4.2. *Součtovou a rozdílovou pokrývací množinu s parametry (n, k, M) označíme jako* maximální*, pokud neexistuje (n, k, M') SDCS taková, že $M < M'$.*

Konstrukce maximálních součtových a rozdílových pokrývacích množin může být pro některé parametry velmi obtížná. Stejně tak ale může být obtížné dokázat, že se opravdu jedná o maximální SDCS.

Tvrzení 3.4.3. *Množina $\mathrm{A} = \{1, 2, \ldots, n\}$, pro $n \in \mathbb{N}$, je $(n, 1, 1 + 2n)$ součtová a rozdílová pokrývací množina a je maximální.*

Důkaz. Nejprve ukážeme, že popsaná množina A je $(n, 1, 1 + 2n)$ SDCS. Prvek 0 získáme bez přidávání jakékoliv změny. Pokud bychom chtěli získat

libovolný jiný prvek $a \in \mathbb{Z}_{1+2n}$, kde $a \neq 0$, získáme potřebné koeficienty následovně. Pokud $a \leq n$, pak $a \in \mathrm{A}$ a potřebná změna je zřejmá. Pokud je naopak $a > n$, pak jistě $((1 + 2n) - a) \in \mathrm{A}$ a příslušný koeficient bude mít zápornou hodnotu.

Nyní je nutné ukázat, že součtová a rozdílová pokrývací množina A je maximální. Pokud vyčíslíme nerovnost z tvrzení 3.4.1 pro případ, kdy $k = 1$, dostaneme

$$M \leq 2^0 \binom{n}{0} + 2^1 \binom{n}{1} = 1 + 2n.$$

Jelikož ale pro naší součtovou a rozdílovou množinu A platí rovnost, musí být jistě maximální.

□

Pomocí předchozího tvrzení jsme získali způsob pro konstrukci součtové a rozdílové pokrývající množiny pro libovolné n. Bohužel jsme ale stále omezeni na případ, kdy $k = 1$. I přes toto omezení jsme ale dosáhli určitého vylepšení. Pokud totiž vyčíslíme hodnotu efektivity ze sekce 3.3, zjistíme, že jsme získali

$$\mathrm{e} = \frac{(1 + 2n)\log_2(1 + 2n)}{\sum_{z=1}^{1+2n} \mathrm{d_A}(z)} = \frac{(2n + 1)\log_2(2n + 1)}{2n},$$

což je o mnoho lepší než u stegosystémů popsaných v sekci 1.5.

Jako příklad této konstrukce můžeme zmínit množinu $\mathrm{A} = \{1, 2\}$, která byla použita jako příklad v sekci 3.1. Po vyčíslení získáme pro tento příklad parametry

$$\alpha \approx 1,161 \qquad\qquad \mathrm{e} \approx 2,902.$$

Můžeme vidět, že to jsou mnohem lepší výsledky než ty, kterých jsme dosáhli v sekci 1.5.

n	M	α	e
1	3	1,585	2,377
2	5	1,161	2,902
3	7	0,936	3,275
4	9	0,792	3,566
5	11	0,691	3,805
9	19	0,472	4,484

Tabulka 3.3: Parametry stegosystémů vytvořených z maximálních $(n, 1, M)$ součtových a rozdílových pokrývacích množin.

Zatím jsme ukázali způsob konstrukce SDCS pouze pro $k = 1$. V rámci tohoto omezení jsme ale dosáhli ideálních množin, protože jsou maximální.

Dosažené výsledky můžeme porovnat s vlastnostmi součtových a rozdílových množin, které jsme popsali v tabulce 3.2, kde máme většinou $k > 1$. V tabulce 3.3 jsou vypsány parametry pro součtové a rozdílové množiny, které je možné sestrojit podle tvrzení 3.4.3. Pokud je porovnáme s výsledky v tabulce 3.2, zjistíme, že pro stejná n získáváme lepší efektivitu. Na druhou stranu ale dostáváme horší relativní kapacitu.

Jediná konstrukce, kterou jsme zatím předvedli, byla pro extrémní hodnotu $k = 1$. Nyní si ukážeme, jak zkonstruovat součtovou a rozdílovou množinu pro druhý extrém.

Tvrzení 3.4.4. *Množina* $\mathrm{A} = \{3^0, 3^1, 3^2, \ldots, 3^{n-1}\}$*, pro* $n \in \mathbb{N}$*, je* $(n, n, 3^n)$ *součtová a rozdílová pokrývací množina. Tato SDCS je maximální.*

Důkaz. Nejprve musíme ukázat, že je uvedená množina A skutečně součtová a rozdílová pokrývací množina s popsanými parametry, a dále, že je tato SDCS maximální.

- Musíme tedy dokázat, že jsme schopni dosáhnout libovolného $z \in \mathbb{Z}_{3^n}$.

  Pokud se nyní zaměříme na posloupnost $q^0, q^1, q^2, \ldots$ zjistíme, že se jedná o bázi pro q–ární zápis. Máme-li tedy libovolné $z, n \in \mathbb{N}$, kde $z < q^n$, můžeme jej zapsat jako

  $$z = \sum_{i=0}^{n} s_i q^i,$$

  kde $s_i \in \{0, 1, 2, \ldots, q-1\}$.

  V našem případě je tedy množina A rovna bázi trojkového zápisu prvků z $\mathbb{Z}_{3^n}$. To znamená, že každé $z \in \mathbb{Z}_{3^n}$ je možné zapsat jako

  $$z = \sum_{i=0}^{n} s_i 3^i,$$

  kde $s_i \in \{0, 1, 2\}$. My ale musíme přejít k soustavě, kde koeficienty nabývají pouze hodnot z množiny $\{-1, 0, 1\}$. Takové soustavě se podle [9] říká vyvážená ternární soustava.

  Je tedy nutné ukázat, jakým způsobem je možné převést ternární soustavu na námi potřebný tvar. Je snadné ověřit, že pokud lze číslo zapsat v trojkové soustavě pouze pomocí jedniček a nul, je jeho zápis ve vyvážené ternární soustavě stejný. Problém tedy nastává, kdykoliv se v zápisu objeví dvojka. Platí ale rovnost $2 = 1 \cdot 3^1 + (-1) \cdot 3^0$. Z této rovnosti můžeme vyjít. Nechť máme $z \in \mathbb{Z}_{3^n}$, přičemž platí

  $$z = \sum_{i=0}^{n} s_i 3^i,$$

kde $s_i \in \{0,1,2\}$. Existuje-li $k \in \{0,1,2,\ldots,n\}$ takové, že $s_k = 2$, můžeme koeficienty s_{k+1} a s_k nahradit následujícím způsobem.

$$\begin{aligned} s_{k+1}3^{k+1} + 2 \cdot 3^k &= 3s_{k+1}3^k + 2 \cdot 3^k = (3s_{k+1} + 2)\,3^k = \\ &= (3s_{k+1} + 3 - 1)\,3^k = \\ &= (3(s_{k+1} + 1) - 1)\,3^k = \\ &= (s_{k+1} + 1)3^{k+1} - 1 \cdot 3^k \end{aligned}$$

Právě jsme ukázali způsob, jak v trojkovém zápisu nahradit dvojku. Stačí inkriminovaný koeficient nahradit prvkem -1 a předchozí prvek zvětšit o jedna.

Pokud budeme chtít tuto metodu aplikovat na cyklickou grupu $\mathbb{Z}_{3^n}$, vše funguje stejným způsobem. Hodnota koeficient u prvku 3^n je totiž z definice $\mathbb{Z}_{3^n}$ stále nulová.

Právě jsme tedy ověřili, že A je součtová a rozdílová pokrývací množina s uvedenými parametry.

- Nyní ukážeme, že je popsaná součtová a rozdílová pokrývací množina maximální.

  Z tvrzení 3.4.1 víme, že platí odhad

  $$M \leq 2^0\binom{n}{0} + 2^1\binom{n}{1} + \ldots + 2^k\binom{n}{n}.$$

  Pro dokončení důkazu stačí ukázat, že

  $$\sum_{i=0}^{n} 2^i\binom{n}{i} = 3^n.$$

  Budeme opět postupovat indukcí. Pro $n = 0$ je tvrzení zřejmé.

  Nyní provedeme indukční krok. Stačí nám ukázat, že

  $$\sum_{i=0}^{n+1} 2^i\binom{n+1}{i} = 3\sum_{i=0}^{n} 2^i\binom{n}{i}.$$

  Nejprve si výraz upravíme.

  $$\sum_{i=0}^{n+1} 2^i\binom{n+1}{i} = 1 + \sum_{i=1}^{n} 2^i\binom{n+1}{i} + 2^{n+1}$$

  Podle Pascalova pravidla platí

  $$\binom{n+1}{k} = \binom{n}{k-1} + \binom{n}{k}.$$

Můžeme tedy udělat následující úpravu.

$$1 + \sum_{i=1}^{n} 2^i \binom{n+1}{i} + 2^{n+1} = 1 + \sum_{i=1}^{n} 2^i \left(\binom{n}{i} + \binom{n}{i-1} \right) + 2^{n+1}$$

Pokud ale výraz dále upravíme, dostaneme požadovaný tvar

$$\sum_{i=0}^{n} 2^i \binom{n}{i} + \sum_{i=0}^{n-1} 2^{i+1} \binom{n}{i} + 2^{n+1} = \sum_{i=0}^{n} 2^i \binom{n}{i} + 2\sum_{i=0}^{n} 2^i \binom{n}{i},$$

protože

$$\sum_{i=0}^{n} 2^i \binom{n}{i} + 2\sum_{i=0}^{n} 2^i \binom{n}{i} = 3\sum_{i=0}^{n} 2^i \binom{n}{i}.$$

Ukázali jsme, že A je součtová a rozdílová množina s popsanými parametry a že je maximální. Důkaz je tedy hotov.

□

Nyní zkusíme ukázat způsoby, jakými lze konstruovat obecnější součtové a rozdílové pokrývací množiny.

3.4.1 Konstrukce obecné SDCS

V této části předvedeme konstrukci (n, k, M) součtové a rozdílové množiny pro libovolné $k \in \{1, 2, \ldots, n\}$. Na rozdíl od předchozích konstrukcí nebude tento způsob ideální. Dokážeme tak ale, že se dají sestrojit součtové a rozdílové pokrývací množiny pro různé hodnoty k.

Nejprve shrneme celou konstrukci a následně si popíšeme význam jednotlivých kroků.

Vlastní konstrukce

Na začátku se musíme rozhodnout, jakou součtovou a rozdílovou pokrývací množinu budeme chtít vyrobit. Budeme vytvářet (n, k, M) SDCS. Popisovaný způsob konstrukce nám dovoluje zvolit n a k, přičemž M dostaneme na konci konstrukce.

1. Zvolíme $k, n \in \{1, 2, 3, \ldots\}$ tak, že $k \leq n$.
2. Zvolíme si c_i, pro $i \in \{0, 1, 2, \ldots, k\}$ taková, že $c_0 = 0$, $c_k = n$ a zároveň platí $c_i < c_j$, pokud $i < j$.
3. Definujeme a_i, kde $i \in \{1, 2, \ldots, c_1\}$ jako $a_i = i$.
4. Pro zbylá $i \in \{c_1 + 1, c_1 + 2, \ldots, c_k\}$ definujeme l_i tak, aby byla splněna rovnost $c_{l_i} < i \leq c_{l_i+1}$.

5. Zbylé prvky a_i pro $i \in \{c_1 + 1, c_1 + 2, \ldots, c_k\}$ definujeme jako

$$a_i = \left(1 + 2\sum_{j=1}^{l_i} a_{c_j}\right)(i - c_{l_i})\,. \tag{3.3}$$

6. Sestrojíme výslednou součtovou a rozdílovou pokrývací množinu A = $\{a_1, a_2, a_3, \ldots, a_n\}$.

V další části si ukážeme, jestli nám předchozí konstrukce opravdu vytvoří součtovou a rozdílovou pokrývací množinu a zda jsme tak skutečně získali zvolené parametry.

Popis konstrukce

Na začátku konstrukce, v bodě 1 na straně 55, jsme si zvolili parametry, pro které budeme konstruovat součtovou a rozdílovou pokrývací množinu. Omezení $k \leq n$ je zřejmé. Nemůžeme měnit víc prvků, než kolik jich máme k dispozici.

Po zvolení základních parametrů si musíme pomyslně rozdělit slova délky n na k částí. K tomu nám budou sloužit hraniční body c_i, které jsme si zvolili v bodě 2. Jako i-tý úsek budeme označovat všechny prvky a, které splňují $c_{i-1} < a \leq c_i$. Tímto způsobem jsme rozdělili slova délky n na k disjunktních úseků, které pokrývají celý interval od 0 do n. Ve výsledku totiž budeme chtít, aby se v každém úseku nacházela nejvýše jedna změna. Tím dostaneme vždy maximálně k změn.

V další části konstrukce jsme definovali všechna a_i z prvního úseku jako $a_i = i$. Budeme totiž postupovat zleva doprava takovým způsobem, že každým dalším úsekem rozšíříme předchozí řešení. V prvním úseku tedy budeme chtít pokrýt nejmenší možná čísla. Pokud tedy definujeme jednotlivá a_i z prvního úseku popsaným způsobem, dokážeme pomocí nejvýše jedné změny v tomto úseku dosáhnout libovolného prvku $z \in \{0, 1, 2, \ldots, c_1\}$.

V bodech 4 a 5 jsme sestrojili a_i pro ostatní úseky. Předvedeme, že naše konstrukce je správná a proč probíhá popsaným způsobem.

Jak jsme již zmínili výše, budeme postupovat zleva doprava. Indukcí si ukážeme, že můžeme postupně rozšiřovat množinu o další úseky. Každý úsek bude mít tu vlastnost, že se v něm může provést vždy nejvýše jedna změna. Indukci začneme důkazem pro jediný blok.

Tvrzení 3.4.5. *Konstrukce na straně 55 v bodu 3 vytvoří součtovou a rozdílovou pokrývací množinu* A = $\{a_1, a_2, \ldots, a_{c_1}\}$. *Tato SDCS má parametry* $(c_1, 1, 1 + 2c_1)$.

Důkaz. Je zřejmé, že pomocí jediné změny dokážeme získat prvky $z \in \{1, 2, \ldots, c_1\}$. Prvky $z \in \{c_1 + 1, c_1 + 2, \ldots, 2c_1\}$ můžeme získat také, neboť $-z = (2c_1 + 1) - z = \overline{z} \in \{1, 2, \ldots, c_1\}$. Můžeme tedy snadno vytvořit prvek $\overline{z}$ a následně vynásobit získané koeficienty prvkem -1. Prvek 0

dokážeme získat ihned. Po dokončení kroku 3 ze strany 55 jsme tedy získali $(c_1, 1, 1 + 2c_1)$ SDCS.

□

Předchozím tvrzením jsme položili základ pro indukci a nyní dokážeme platnost indukčního kroku.

Tvrzení 3.4.6. *Předpokládejme, že máme definované prvky a_i pro prvních m úseků slova délky n. Nechť dále platí, že $\mathrm{A} = \{a_1, a_2, \ldots, a_{c_m}\}$ je součtová a rozdílová pokrývací množina s parametry*

$$\left(c_m, m, 1 + 2 \sum_{j=1}^{m} a_{c_j} \right),$$

kde $c_m < n$. Pokud sestrojíme všechna a_i z bloku $(m+1)$ způsobem popsaným v bodech 4 a 5 na straně 55, dostaneme SDCS s parametry

$$\left(c_{m+1}, m + 1, 1 + 2 \sum_{j=1}^{m+1} a_{c_j} \right).$$

Důkaz. Podle popsaného postupu definujeme jednotlivá a_i, pro všechna $i \in \{c_m + 1, c_m + 2, \ldots, c_{m+1}\}$, jako

$$a_i = \left(1 + 2 \sum_{j=1}^{l_i} a_{c_j} \right) (i - c_{l_i}),$$

kde l_i splňuje $c_{l_i} < i \leq c_{l_i+1}$. V našem případě je $l_i = m$, protože máme všechna a_i z úseku $m + 1$. Hodnota

$$\left(1 + 2 \sum_{j=1}^{l_i} a_{c_j} \right) = \left(1 + 2 \sum_{j=1}^{m} a_{c_j} \right)$$

je tak pro nás stále stejná. Jedná se o hodnotu parametru M ze součtové a rozdílové pokrývací množiny A. Pokud tedy v úseku $m+1$ přiřadíme prvkům a_i hodnoty, které od sebe budou vzdáleny právě o toto číslo, využijeme tento úsek maximálním možným způsobem. Stále totiž požadujeme, abychom v každém úseku provedli nejvýše jednu změnu. To přesně odpovídá výpočtu v bodu 5.

Pro větší názornost si můžeme nově definované body představovat jako nějaké body v prostoru. Je snadno vidět, že se od těchto bodů můžeme dostat až na vzdálenost $\sum_{j=1}^{m} a_{c_j}$ na obě strany. Každý bod má tak kolem sebe určité pole působnosti. Prvky z tohoto pole je možné vyjádřit pomocí nejvýše

jednoho prvku z každého úseku spolu se středovým bodem příslušného pole. Pokud nyní tyto středové body rozmístíme v prostoru takovým způsobem, aby se jejich pole působnosti dotýkala, vytvoří tak společně souvislou oblast, ve které je možno vyjádřit libovolný prvek popsaným způsobem. Tento postup odpovídá definici prvků $a_{c_m}+1, a_{c_m}+2, \ldots, a_{c_{m+1}}$. Prvky začínáme definovat zleva doprava. Každý další prvek umístíme vždy takovým způsobem, aby se jeho pole působnosti dotýkalo s polem působnosti předchozího prvku.

V případě, že definujeme všechny prvky a_i z úseku $m+1$ způsobem popsaným v bodě 5, získáme opět součtovou a rozdílovou pokrývací množinu. Její parametry se pak zřejmě změní na

$$\left(c_{m+1}, m+1, 1+2\sum_{j=1}^{m+1} a_{c_j} \right),$$

protože velikost množiny se zvětšila o $(c_{m+1}-c_m)$, můžeme dělat o jednu změnu víc a pokrytá množina se zvětšila o hodnotu $a_{c_{m+1}}$ na obě strany, což přesně odpovídá zvětšení o $2a_{c_{m+1}}$.

□

Správnost konstrukce tedy můžeme ukázat indukcí pomocí tvrzení 3.4.5 a tvrzení 3.4.6. Tím jsme předvedli, že je možné zkonstruovat součtovou a rozdílovou pokrývací množinu pro libovolné k.

Takto zkonstruované SDCS zřejmě mnohdy nebudou ideální. Existují ale případy, kdy takto můžeme získat maximální součtovou a rozdílovou pokrývací množinu. Jedná se o obě extrémní hodnoty k. Pokud budeme konstruovat součtovou a rozdílovou pokrývací množinu pro $k=1$, dosáhneme stejného výsledku jako v tvrzení 3.4.3. Pokud bychom naopak chtěli zkonstruovat SDCS pro $k=n$, dostali bychom stejnou množinu jako v případě tvrzení 3.4.4. U obou těchto konstrukcích jsme již dokázali, že jsou maximální.

Parametr M můžeme převést na tvar, který lépe vyjadřuje závislost na původní volbě rozmístění hodnot c_i. Jak jsme již zmínili výše, víme, že

$$M = 1 + 2\sum_{j=1}^{k} a_{c_j}.$$

Označme nyní rozdíly mezi jednotlivými po sobě jdoucími hodnotami c_i jako $b_i = c_i - c_{i-1}$, pro $i \in \{1, 2, \ldots, k\}$. Podle definice víme, že $c_1 = b_1$, a podle (3.3) máme

$$a_{c_i} = \left(1 + 2\sum_{j=1}^{i-1} a_{c_j} \right) b_i, \tag{3.4}$$

pro všechna $i \in \{2, 3, \ldots, k\}$.

Naší snahou bude dokázat, že

$$M = \prod_{i=1}^{k}(1 + 2b_i).$$

Důkaz provedeme indukcí. Pro $k = 1$ je tvrzení zřejmé.

Nyní dokážeme indukční krok. Nechť tedy tvrzení platí pro $k - 1$, pak platí

$$M = 1 + 2\sum_{j=1}^{k} a_{c_j} = \left(1 + 2\sum_{j=1}^{k-1} a_{c_j}\right) + 2a_{c_k},$$

což pomocí (3.4) můžeme převést na

$$\left(1 + 2\sum_{j=1}^{k-1} a_{c_j}\right) + 2a_{c_k} = \left(1 + 2\sum_{j=1}^{k-1} a_{c_j}\right) + 2b_k\left(1 + 2\sum_{j=1}^{k-1} a_{c_j}\right).$$

Podle indukčního předpokladu, který můžeme použít hned dvakrát, dostaneme

$$M = \left(\prod_{i=1}^{k-1}(1 + 2b_i)\right) + 2b_k\left(\prod_{i=1}^{k-1}(1 + 2b_i)\right),$$

a tím tedy i výslednou hodnotu

$$M = \prod_{i=1}^{k}(1 + 2b_i).$$

Dostali jsme jiné vyjádření pro parametr M, a proto můžeme na začátku konstrukce volit parametry c_i tak, abychom dostali SDCS, která bude co nejvíce odpovídat našim potřebám.

Pokud bychom chtěli dosáhnout co nejefektivnější součtové a rozdílové pokrývací množiny, musíme při zachování n a k maximalizovat M. Pomocí nově odvozeného vzorce pro M, můžeme na počátku konstrukce volit taková c_i, aby byla hodnota M co možná největší. Pokud budeme brát jednotlivé závorky $(1 + 2b_i)$ jako celky, je zřejmé, že pro dosažení maximálního možného M musíme rozložit c_i rovnoměrně. Nejvyššího možného M tedy dosáhneme takovým způsobem, že zvolíme c_i tak, aby $b_i = b_j$ pro všechna $i, j \in \{1, 2, \ldots, k\}$.

Můžeme si také ukázat, že v mnoha případech můžeme nalézt vhodnější součtové a rozdílové pokrývací množiny, než jsou ty, které bychom zkonstruovali pomocí právě popsaného postupu.

Z předchozích úvah víme, že pro dosažení ideální součtové a rozdílové pokrývací množiny potřebujeme rozložit c_i co nejvíc rovnoměrně. Pro ideální

případ budeme předpokládat, že $b_i = \frac{n}{k}$ pro všechna $i \in \{1, 2, \ldots, k\}$. Tím dostáváme

$$M = \prod_{i=1}^{k} \left(1 + 2\frac{n}{k}\right) = \frac{2^k n^k}{k^k} + O\left(n^{k-1}\right).$$

Pokud bychom se zaměřili na $k = 2$, dostali bychom $M = n^2 + 2n + 1$. Vyjádříme-li ale odhad z tvrzení 3.4.1 pro $k = 2$, dostaneme

$$M \leq \binom{n}{0} + 2\binom{n}{1} + 2^2\binom{n}{2} = 1 + 2n + 2n(n-1),$$

což je již pro $n > 2$ vždy větší než u naší konstrukce. Je tedy možné, že existují lepší konstrukce pokrývacích množin.

3.4.2 Konstrukce $(n, 2, M)$ SDCS

V této části si předvedeme efektivnější způsob konstrukce součtových a rozdílových pokrývacích množin. Na druhou stranu ale zase budeme omezeni na případ, kdy $k = 2$. Tento konkrétní případ je také rozebírán v [10].

Nejprve si popíšeme způsob, jakým se množina konstruuje. Následně si předvedeme, že jsme skutečně zkonstruovali SDCS a ukážeme si, jakých jsme tak pro ni dosáhli parametrů.

Konstrukce

Opět se musíme rozhodnout, pro jaké parametry budeme chtít součtovou a rozdílovou pokrývací množinu vytvořit. Parametry SDCS budeme jako vždy značit (n, k, M). V našem případě již víme, že $k = 2$. Parametr M určíme opět až později. Zbývá tedy volba n.

1. Zvolíme l, m, n tak, že $m \in \{0, 1, 2 \ldots\}$, $l \in \{1, 2 \ldots\}$, $n \in \{2, 3, 4, \ldots\}$, a aby zároveň platilo
$$n = l + 2m + 1.$$

2. Definujeme a_i pro $i \in \{1, 2, \ldots, l\}$ jako
$$a_i = i.$$

3. Pro $j \in \{1, 2, \ldots, m\}$ definujeme a_{l+2j} jako
$$a_{l+2j} = (4j + 1)\, l + (2m + 1)\, j.$$

4. Dále definujeme a_{l+2j+1} pro $j \in \{0, 1, 2, \ldots, m\}$ jako
$$a_{l+2j+1} = (4j + 3)\, l + (2m + 1)\, j + 2m - 2j.$$

5. Sestrojíme konečnou součtovou a rozdílovou pokrývací množinu

$$\mathrm{A} = \{a_1, a_2, \ldots, a_n\}.$$

Nyní musíme ukázat, že zkonstruovaná množina A je skutečně součtová a rozdílová pokrývací množina. S tím souvisí i nutnost spočítat parametry této množiny.

Odvození parametrů

V bodě 1 postupu konstrukce na straně 60 jsme omezili volbu n na čísla větší nebo rovna 2. Toto omezení je ale zřejmé, protože $k = 2$. Volbu l a m využijeme až v další části konstrukce, kde také zjistíme jejich význam.

Dále jsme definovali a_i pro $i \in \{1, 2, \ldots, l\}$ jako $a_i = i$. Tento krok je podobný předchozí konstrukci ze strany 55. Pomocí těchto a_i jsme zřejmě schopni pokrýt prvky $z \in \{1, 2, \ldots, 2l-1\}$, protože $k = 2$, což odpovídá tomu, že můžeme použít až dva prvky.

V dalších dvou krocích konstrukce vytváříme zbylé prvky a_i. Abychom ukázali, že popsaným způsobem získáme vhodné prvky pro součtovou a rozdílovou pokrývací množinu, musíme upřesnit, pro jaké parametry SDCS vytváříme. Již víme, že $n = l + 2m + 1$ a že $k = 2$. Zbývá určit M. V dalším postupu ukážeme, že $M = (8m + 8)\, l + 4m^2 + 4m + 1$. Pro důkaz, že A je součtová a rozdílová množina s uvedenými parametry si uvedeme několik tvrzení, která později také dokážeme.

a) Každé $z \in \{0, 1, 2, \ldots, 2l-1\}$ je možné vyjádřit pomocí nejvýše dvou prvků a_i, a_j, kde $i, j \in \{1, 2, \ldots, l\}$ a zároveň $i \neq j$.

b) Pro všechna $j \in \{1, 2, \ldots, 2m+1\}$ můžeme libovolný prvek $z \in \{a_{l+j} - l, a_{l+j} - l + 1, \ldots, a_{l+j} + l - 1, a_{l+j} + l\}$ zapsat pomocí nejvýše dvou prvků a_i, a_{l+j}, kde a $i \leq l$.

c) Každý prvek $z \in \{a_{l+2j} + l + 1, \ldots, a_{l+2j+1} - l - 1\}$, kde $j \in \{0, 1, 2, \ldots, m\}$ můžeme vyjádřit buď jako

$$a_{l+2(j'+j)+1} - a_{l+2j'}$$

nebo jako

$$a_{l+2(j'+j)} - a_{l+2j'-1},$$

kde $j' \in \{1, 2, \ldots, m - j\}$.

d) Každé $z \in \{a_{l+2j+1} + l, a_{l+2j+1} + l + 1, \ldots, a_{l+2j+2} - l - 1, a_{l+2j+2} - l\}$, kde $j \in \{0, 1, 2 \ldots, m\}$ můžeme zapsat buď jako

$$a_{l+2j'} + a_{l+2j''+1} \qquad \left(j' + j'' = j\right)$$

nebo jako

$$M - a_{l+2j'} - a_{l+2j''+1} \qquad \left(j' + j'' = 2m - j\right),$$

přičemž předpokládáme, že $a_{l+2m+2} = (4m+5)\,l + (2m+1)\,(m+1)$ a $M = (8m+8)\,l + 4m^2 + 4m + 1$.

Pokud budeme vědět, že platí všechna předchozí tvrzení, můžeme ukázat, že zkonstruované A je součtová a rozdílová pokrývací množina s parametry $(n, 2, M)$, přičemž $n = l + 2m + 1$ a $M = (8m+8)\,l + 4m^2 + 4m + 1$.

V předešlých tvrzeních jsme předvedli, že jsme schopni vyjádřit libovolný prvek $z \in \{0, 1, 2, \ldots, (4m+4)\,l + (2m+1)\,(m+1)\}$ pomocí nejvýše dvou prvků a_i, a_j, kde $i \neq j$. Jestliže budeme chtít vyjádřit libovolný prvek $\overline{z} \in \{(4m+4)\,l + (2m+1)\,(m+1) + 1, (4m+4)\,l + (2m+1)\,(m+1) + 2, \ldots, M\}$, stačí nám, abychom vyjádřili prvek $z = M - \overline{z}$. Tento prvek je zřejmě možné vyjádřit, protože

$$(4m+4)\,l + (2m+1)\,(m+1) > \frac{M-1}{2}$$

a z toho tedy vyplývá, že $z \in \{0, 1, 2, \ldots, (4m+4)\,l + (2m+1)\,(m+1)\}$. Pak už jen stačí všechny koeficienty vynásobit prvkem -1.

Důkaz tvrzení a) jsme již vlastně ukázali předtím. Toto tvrzení tedy zřejmě platí.

Pravdivost tvrzení b) je také zřejmá. Ke každému prvku a_{l+j} můžeme zřejmě přičíst libovolný prvek a_i, kde $i \in \{1, 2, \ldots, l\}$. Z tvrzení a) je pak snadno vidět, že jsme schopni dostat zmíněné prvky.

Důkaz tvrzení c) a d) bude trochu složitější, proto obě uvedeme samostatně.

Tvrzení 3.4.7. *Libovolný prvek $z \in \{a_{l+2j} + l, \ldots, a_{l+2j+1} - l - 1\}$, kde $j \in \{0, 1, 2, \ldots, m\}$ můžeme vyjádřit jedním z následujících způsobů.*

i) $a_{l+2(j'+j)+1} - a_{l+2j'}$,

ii) $a_{l+2(j'+j)} - a_{l+2j'-1}$,

kde $j' \in \{1, 2, \ldots, m - j\}$.

Důkaz. Nejprve si označíme $G_j = \{a_{l+2j} + l, \ldots, a_{l+2j+1} - l - 1\}$, kde $j \in \{0, 1, 2, \ldots, m\}$. Budeme chtít ukázat, že každé $z \in G_j$ můžeme vyjádřit popsaným způsobem. Počet prvků množiny G_j je

$$\begin{aligned} |G_j| &= (a_{l+2j+1} - l) - (a_{l+2j} + l - 1) + 1 = \\ &= ((4j+3)l + (2m+1)j + 2m - 2j - l) - \\ &\quad - ((4j+1)l + (2m+1)j + l - 1) + 1 = \\ &= 2m - 2j. \end{aligned} \tag{3.5}$$

Pokud by mělo tvrzení platit, znamená to, že pro každé $z \in G_j$ musí existovat nejvýše jedno j' společně s i) nebo ii), kterými se dá z zapsat. Je to z toho důvodu, že počet prvků množiny G_j je roven dvojnásobku možných j'. Pokud nyní vezmeme v úvahu, že pro každé j' připadá v úvahu i) nebo ii), dostáváme zmíněný závěr.

Pro další postup si zvolme libovolné j a dále budeme uvažovat, že j má pevnou hodnotu.

Můžeme snadno ověřit, že

$$a_{l+2(j+1)} - a_{l+2j} = 4l + 2m + 1, \tag{3.6}$$

$$a_{l+2(j+1)+1} - a_{l+2j+1} = 4l + 2m - 1. \tag{3.7}$$

Dále budeme jako u_h značit vyjádření i) pro hodnotu $j' = h$. Podobně budeme jako v_h značit vyjádření ii) pro hodnotu $j' = h$.

Z předchozích rovností můžeme vyjádřit

$$u_{j'+1} - u_{j'} \overset{(3.6),(3.7)}{=} (4l + 2m - 1) - (4l + 2m + 1) = -2 \tag{3.8}$$

$$v_{j'+1} - v_{j'} \overset{(3.6),(3.7)}{=} (4l + 2m + 1) - (4l + 2m - 1) = 2 \tag{3.9}$$

Z rovnic (3.8) a (3.9) je zřejmé, že pokud budeme postupně zvyšovat hodnotu j', bude se v případě i) výsledná hodnota snižovat o 2. V případě ii) se bude výsledná hodnota zvyšovat o 2.

Pokud se nám podaří ukázat, že se hodnota u_{m-j} nachází na levém okraji množiny G_j a v_{m-j} se nachází na jejím pravém okraji, tvrzení bude dokázáno.

Pro snadnější důkaz si vyjádříme hodnotu u_1.

$$\begin{aligned} u_1 &= a_{l+2(j+1)+1} - a_{l+2} = \\ &= (4j+7)l + (2m+1)j + (2m+1) + 2m - 2j - 2 - \\ &\quad - 4l - l - 2m - 1 = \\ &= [(4j+3)l + (2m+1)j + 2m - 2j] - l - 2 = \\ &= a_{l+2j+1} - l - 2 \end{aligned}$$

Právě jsme tedy ukázali, že vyjádření z bodu i) pro $j' = 1$ je o jedna nižší než maximální hodnota množiny G_j. Vezmeme-li v úvahu (3.8) a (3.5), dostaneme, že pro všechna $j' \in \{1, 2, \ldots, m - j\}$ je výsledná hodnota

$$u_{j'} = a_{l+2j} + l + 2s,$$

kde $s \in \{0, 1, 2, \ldots, m - j - 1\}$.

Zbývá tedy ukázat podobnou vlastnost pro $v_{j'}$. Opět si pro jednoduchost vyjádříme hodnotu $v_{j'}$ pro $j' = 1$.

$$\begin{aligned} v_1 &= a_{l+2(j+1)} - a_{l+2-1} = a_{l+2(j+1)} - a_{l+1} = \\ &= (4j+5)l + (2m+1)j + (2m+1) - 3l - 2m = \\ &= [(4j+1)l + (2m+1)j] + l + 1 = \\ &= a_{l+2j} + l + 1 \end{aligned}$$

Podobně jako v předchozím případě, tentokrát s využitím (3.9) a (3.5), získáváme výsledek, že pro všechna $j' \in \{1, 2, \ldots, m-j\}$ je konečná hodnota

$$v_{j'} = a_{l+2j} + l + 1 + 2s,$$

kde $s \in \{0, 1, 2, \ldots, m-j-1\}$.

Nyní stačí zkombinovat oba výsledky. Tím jsme ukázali, že libovolné $z \in \{a_{l+2j} + l, \ldots, a_{l+2j+1} - l - 1\}$ je možné zapsat popsaným způsobem. Důkaz je tedy hotov.

□

V předešlém tvrzení jsme dokázali dokonce více, než je potřeba v tvrzení c), takže nám už jen stačí dokázat tvrzení d).

Tvrzení 3.4.8. *Každé $z \in \{a_{l+2j+1} + l, a_{l+2j+1} + l + 1, \ldots, a_{l+2j+2} - l - 1, a_{l+2j+2} - l\}$, kde $j \in \{0, 1, 2 \ldots, m\}$, můžeme zapsat jedním z následujících způsobů.*

i) $a_{l+2j'} + a_{l+2j''+1}$ $\qquad (j' + j'' = j)$,

ii) $M - a_{l+2j'} - a_{l+2j''+1}$ $\qquad (j' + j'' = 2m - j)$,

přičemž předpokládáme, že $a_{l+2m+2} = (4m+5)\,l + (2m+1)\,(m+1)$ *a*

$$M = (8m+8)\,l + 4m^2 + 4m + 1. \tag{3.10}$$

Důkaz. Opět si nejprve označíme množinu, o které musíme dokázat, že ji lze pokrýt popsaným způsobem. Budeme znovu předpokládat, že máme pevně zvolené j. Označme $G_j = \{a_{l+2j+1} + l, a_{l+2j+1} + l + 1, \ldots, a_{l+2j+2} - l - 1, a_{l+2j+2} - l\}$. Podobně jako výše si vypočítáme velikost této množiny.

$$\begin{aligned} |G_j| &= (a_{l+2j+2} - l) - (a_{l+2j+1} + l) + 1 = \\ &= \big(a_{l+2(j+1)} - l\big) - (a_{l+2j+1} + l) + 1 = \\ &= (4(j+1)+1)\,l + (2m+1)(j+1) - l - \\ &\quad - ((4j+3)l + (2m+1)j + 2m - 2j + l) + 1 = \\ &= 2j + 2 \end{aligned} \tag{3.11}$$

Můžeme připomenout, že máme

$$a_{l+2(j+1)} - a_{l+2j} = 4l + 2m + 1, \tag{3.12}$$

$$a_{l+2(j+1)+1} - a_{l+2j+1} = 4l + 2m - 1. \tag{3.13}$$

Protože je pro vyjádření i) hodnota j'' závislá na hodnotě j', můžeme jako u_l označovat hodnotu bodu i) vyjádřenou pro $j' = l$, $j'' = j - l$. Podobně pak můžeme označovat v_l jako hodnotu bodu ii) vyjádřenou pro $j' = l$, $j'' = 2m - l$.

Opět můžeme vyčíslit následující rovnosti.

$$u_{j'+1} - u_{j'} \overset{(3.12),(3.13)}{=} (4l + 2m + 1) - (4l + 2m - 1) = 2 \tag{3.14}$$

$$v_{j'+1} - v_{j'} \overset{(3.12),(3.13)}{=} (4l + 2m - 1) - (4l + 2m + 1) = -2 \tag{3.15}$$

Vidíme tedy, že nastává podobný případ jako v předchozím důkazu. Budeme tedy postupovat stejným způsobem.

Nejprve si vyjádříme hodnotu bodu i) pro případ $j' = 0$ a $j'' = j$.

$$u_0 \quad = \quad a_l - a_{l+2j+1} = l + a_{l+2j+1}$$

Pokud vezmeme v úvahu (3.14), získáme pro $j' \in \{0, 1, 2, \ldots, j\}$ rovnost

$$u_{j'} = a_{l+2j+1} + l + 2s,$$

kde $s \in \{0, 1, 2, \ldots, j\}$.

Podobným způsobem vyjádříme hodnotu bodu ii) pro případ $j' = m - j$ a $j'' = m$.

$$\begin{aligned}
v_{m-j} \quad &= \quad M - a_{l+2(m-j)} - a_{l+2m+1} = \\
&\overset{(3.10)}{=} \quad (8m + 8)l + 4m^2 + 4m + 1 - \\
&\qquad - ((4(m - j) + 1)l + (2m + 1)(m - j)) - \\
&\qquad - ((4m - 3)l + (2m + 1)m + 2m - 2m) = \\
&= \quad 4l + 2m + 1 + 4jl + 2jm + j = \\
&= \quad [(4(j + 1) + 1)l + (2m + 1)(j + 1)] - l = \\
&= \quad a_{l+2(j+1)} - l = \\
&= \quad a_{l+2j+2} - l
\end{aligned}$$

Pomocí (3.15) máme pro $j' \in \{m - j, m - j + 1, \ldots, m\}$ rovnost

$$v_{j'} = a_{l+2j+2} - l - 2s,$$

kde $s \in \{0, 1, 2, \ldots, j\}$.

Nyní stačí zkombinovat dosažené výsledky, využít rovnost (3.11) a důkaz je hotov.

□

Můžeme ještě poznamenat, že definice a_{l+2m+2} by odpovídala definici a_{l+2j}, pro případ $j = m + 1$, kdyby j nebylo omezeno hodnotou m. Pro důkaz, že popsaná množina je SDCS, nám ale restrikce hodnoty j nevadí, protože jsme tuto hodnotu používali pouze pro označení bodu a nepoužívali jsme jej jako součást množiny A.

Dokončili jsme důkaz tvrzení d) a tím jsme ukázali, že popsaná množina A je skutečně součtová a rozdílová pokrývací množina s uvedenými parametry.

V této kapitole jsme předvedli, jakým způsobem se dají využívat součtové a rozdílové pokrývací množiny a jak takové množiny konstruovat. Také jsme ukázali, že se tato metoda zaměřuje na získání vysoké relativní kapacity, což je částečně na úkor efektivity. Předvedený způsob vkládání se snaží vložit do nosiče co největší možnou zprávu za použití změn velikosti nejvýše 1.

Kapitola 4

Duhové grafy

Tato kapitola se bude zabývat využitím duhových grafů ve steganografii. Nejprve si připomeneme pár pojmů z teorie grafů a následně je využijeme pro vytvoření stegosystémů. Na závěr si ukážeme, jestli jsme dokázali dosáhnout nějakého vylepšení.

Jak si ještě ukážeme, budeme využívat q–ární kódy popsané v kapitole 2. Jak jsme ale zjistili, je nevýhodné dělat velké změny. Ty jsou totiž snadno detekovatelné. My se tedy budeme snažit dělat změny velikosti 1. K tomu právě využijeme duhové grafy.

4.1 Teorie grafů

Začneme základní definicí grafu.

Definice 4.1.1 (Graf, množina vrcholů, množina hran)**.** Grafem G *nazveme dvojici* $\mathrm{V(G)}$ *a* $\mathrm{E(G)}$*, kde* $\mathrm{V(G)}$ *je libovolná množina a* $\mathrm{E(G)}$ *je množina dvojic* (a, b) *takových, že* $a, b \in \mathrm{V(G)}$*. Množinu* $\mathrm{V(G)}$ *pak nazýváme* množinou vrcholů *a množinu* $\mathrm{E(G)}$ *nazýváme* množinou hran*.*

Pro naši potřebu budeme uvažovat jen neorientované grafy, můžeme tedy předpokládat, že pokud $(a, b) \in \mathrm{E(G)}$, tak $(b, a) \in \mathrm{E(G)}$. Tyto dvě hrany pak budeme považovat za jednu. Zároveň nebudeme počítat hrany spojující vrchol sám se sebou. Můžeme tedy předpokládat, že dvojice $(a, a) \notin \mathrm{E(G)}$ pro všechny vrcholy $a \in \mathrm{V(G)}$. Stejně tak nebudeme uvažovat ani násobné hrany. Každé dva vrcholy tedy mají nejvýše jednu společnou hranu.

Definice 4.1.2 (Stupeň vrcholu)**.** *Pro graf* G *a vrchol* $v \in \mathrm{V(G)}$ *definujeme* stupeň vrcholu *jako*

$$\deg(v) = |\{e \in \mathrm{E(G)}; \quad e = (v, a), \text{ kde } a \in \mathrm{V(G)}\}| \,.$$

Stupeň vrcholu je vlastně počet vrcholů, se kterými má uvažovaný vrchol společnou hranu.

Definice 4.1.3 (k–regulární graf)**.** *Graf* G *nazveme* k–regulární*, pokud pro všechny vrcholy* $v \in \mathrm{V(G)}$ *platí, že* $\deg(v) = k$*.*

Jedná se tedy o grafy, kde je každý vrchol spojený s k dalšími vrcholy. Příkladem 2–regulárního grafu na 4 vrcholech je čtyřúhelník.

Pro účely steganografie budou nejdůležitější regulární grafy, neboli grafy, jejichž všechny vrcholy mají stejný stupeň. Zaměříme se tedy hlavně na ně. Od teď budeme uvažovat, že jsou všechny grafy regulární, pokud neřekneme jinak.

Definice 4.1.4 (t–obarvení)**.** *Mějme graf* G *a množinu* C *takovou, že* $|\mathrm{C}| = t$*. Pak zobrazení* $\mathrm{c} : \mathrm{V(G)} \to \mathrm{C}$ *nazveme* t–obarvení *jestliže,* $\mathrm{c}(u) \neq \mathrm{c}(v)$ *kdykoliv* $(u, v) \in \mathrm{E(G)}$*. Každý prvek množiny* C *pak nazýváme barvou.*

Definice 4.1.5 (Sousedství vrcholu)**.** *Mějme graf* G *a vrchol* $v \in \mathrm{V(G)}$*. Pak definujeme* sousedství *vrcholu* v *jako* $\mathcal{N}(v) = \{x \in \mathrm{V(G)};\ (x, v) \in \mathrm{E(G)}\}$*.*

Podmínka pro obarvení se dá zřejmě vyjádřit i pomocí sousedství. Barva vrcholu pak musí být jiná, než barva libovolného vrcholu ze sousedství tohoto vrcholu.

Definice 4.1.6 (Duha)**.** *Řekneme, že obarvení grafu* G *je* duha*, právě když pro každý vrchol* $v \in \mathrm{V(G)}$ *se množina* $\{\mathrm{c}(u);\ u \in \mathcal{N}(v)\}$ *skládá z právě* $\deg(v)$ *různých barev.*

Pokud pak jako k označíme stupeň libovolného vrcholu, $k = \deg(v)$, dostáváme, že se jedná nejméně o $(k+1)$–obarvení. Aby nedošlo k nedorozumění, připomeneme, že všechny grafy předpokládáme regulární.

Definice 4.1.7 (Duhový graf)**.** *O grafu* G *řekneme, že je* duhový*, pokud existuje* $k \in \{1, 2, \ldots\}$ *takové, že* G *je* k*–regulární graf a zároveň připouští* $(k+1)$*–obarvení, které je duhové.*

V tomto případě má každý vrchol $v \in \mathrm{V(G)}$ právě k sousedů a každý z nich musí mít jinou barvu. Zároveň je barev právě $(k+1)$, takže na vrchol v zbývá poslední možná barva.

Definice 4.1.8 (Síťový graf)**.** *Nechť* $\mathbb{F} = \mathbb{Z}$ *nebo* $\mathbb{F} = \mathbb{Z}_n$ *a nechť*

$$\mathrm{V(G)} = \mathbb{F}^d, \qquad (u, v) \in \mathrm{E(G)} \Leftrightarrow \exists i \in \{1, 2, \ldots, d\};\ (u - v) = \pm e_i,$$

kde e_i *značí standardní bázové vektory, neboli*

$$(e_i)_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}.$$

Pak takto vzniklý graf budeme nazývat síťový*. Síťový graf o* d *rozměrech budeme značit* $\mathcal{G}_d$*.*

Ve dvou rozměrech si můžeme nekonečný síťový graf představit jako mříž, kde každé křížení představuje vrchol. Každá spojnice mezi nimi pak představuje hranu.

Konečný dvojrozměrný síťový graf je možné si představit podobně. K předchozímu obrazu je třeba přidat informaci o tom, že jsou konce této mříže zkrouceny a propojeny takovým způsobem, že vznikne mřížovaný torus.

Pro využití duhových kódů k redukci velkých změn ve stegosystémech bude následující tvrzení klíčové.

Tvrzení 4.1.9. *Síťový graf $\mathcal{G}_d$ je duhový pro všechna $d \in \{1, 2, \ldots\}$.*

Důkaz. Lze snadno ověřit, že každý d dimenzionální síťový graf je $(2d)$–regulární. Abychom ukázali, že je zmíněný graf zároveň duhový, stačí podle 4.1.6 najít duhové obarvení $\mathrm{c} : \mathbb{F}^d \to \mathbb{F}_{2d+1}$ takové, že c je $(2d+1)$–obarvení.

Každý vrchol $w \in \mathbb{F}^d$ můžeme zapsat jako $w = (w_1, w_2, \ldots, w_d)$, přičemž pro všechna $i \in \{1, 2, \ldots, d\}$ jsou $w_i \in \mathbb{F}$. Můžeme tedy definovat zobrazení c jako

$$\mathrm{c}(w) = \left(\sum_{i=1}^{d} i w_i \right) \bmod (2d+1). \tag{4.1}$$

Nyní stačí ukázat, že zobrazení c je skutečně duhové $(2d+1)$–obarvení.

Vezměme si tedy libovolný prvek $w \in \mathbb{F}^d$. Pokud nyní vezmeme jeho libovolného souseda $v \in \mathcal{N}(w)$, bude podle definice 4.1.8 tvaru

$$v = w + (-1)^k e_i,$$

kde $i \in \{1, 2, \ldots, d\}$, e_i značí některý z bázových vektorů a $k \in \{0, 1\}$. Pro dokončení důkazu stačí ukázat, že pro všechny sousední vrcholy $v \in \mathcal{N}(w)$ platí $\mathrm{c}(v) \neq \mathrm{c}(w)$ a že pro pro všechny dvojice $u, v \in \mathcal{N}(w)$, kde $u \neq v$ platí, že $\mathrm{c}(u) \neq \mathrm{c}(v)$.

Jelikož má prvek w podle definice (4.1) přiřazenou barvu

$$\mathrm{c}(w) = \left(\sum_{i=1}^{d} i w_i \right) \bmod (2d+1),$$

barva jeho libovolného souseda $v \in \mathcal{N}(w)$, kde $v = w + (-1)^k e_i$, je rovna

$$\mathrm{c}(v) = \left((-1)^k i + \sum_{j=1}^{d} j w_j \right) \bmod (2d+1).$$

To znamená, že pro všechny sousedy $v \in \mathcal{N}(w)$ platí, že

$$\mathrm{c}(v) = \mathrm{c}(w) \pm i,$$

kde $i \in \{1, 2, \ldots, d\}$. Musíme tedy ukázat, že pro všechna $i \in \{1, 2, \ldots, d\}$ platí

$$\pm i \neq 0 \bmod (2d + 1).$$

Zároveň musíme předvést, že pro všechna $i, j \in \{1, 2, \ldots, d\}$ taková, že $i \neq j$ platí, že

$$i \pm j \neq 0 \bmod (2d + 1).$$

Obojí je ale zřejmé, protože $i, j \leq d$. Důkaz je tedy hotov.

□

4.2 Aplikace duhového obarvení ve steganografii

V kapitole 2 jsme předvedli použití q–árních kódů pro vytváření stegosystémů. V příkladě 2.5.2 jsme ukázali, že q–ární kódy jsou výhodnější než binární. Na druhou stranu se ale při vkládání modifikuje původní nosič o změny, které jsou až velikosti $\lfloor \frac{q}{2} \rfloor$. To činí stegosystém snáze detekovatelný. Tomu chceme samozřejmě zabránit.

Pomocí duhového obarvení popsaného v části 4.1 omezíme prováděné úpravy na ± 1 změny. Tím využijeme výhody q–árních kódů a zároveň odbouráme nevýhodu snadné detekovatelnosti.

Pro využití duhového grafu v nějakém stegosystému budeme postupovat následujícím způsobem:

- Budeme předpokládat, že chceme využít q–ární kód, kde q je liché. Budeme tedy předpokládat, že platí

  $$q = 2d + 1, \tag{4.2}$$

  pro nějaké $d \in \{1, 2, \ldots\}$.

- Původní nosič $x \in \mathbb{F}^*$ rozdělíme na bloky délky d.

- Využijeme definici obarvení z důkazu tvrzení 4.1.9. Přiřazení (4.1) nám dává zobrazení z $\mathbb{F}^d$ do $\mathbb{Z}_q$.

- Podle tvrzení 4.1.9 jsme schopni dostat libovolný prvek $\mathbb{Z}_q$ maximálně jednou změnou o velikosti 1.

- Vezmeme-li tedy d-tice prvků nosiče jako jednotlivé celky, můžeme je pomocí definovaného zobrazení používat jako prvky $\mathbb{Z}_q$.

- Tyto prvky pak můžeme využít pro odvození stegosystému pomocí libovolného q–árního kódu. Proces získávání stegosystémů z q–árních kódů byl popsán v kapitole 2.

Popsaný způsob využívá duhové obarvení, a tím modifikuje stegosystémy vytvořené z q–árních kódů. Při modifikaci dochází k vylepšení v tom smyslu, že provádíme vždy maximálně jednu změnu o velikosti 1 pro každých d prvků nosiče. Tím sice trochu klesá efektivita, zároveň ale také klesá detekovatelnost stegosystému, což je velmi žádoucí.

4.3 Vlastnosti

V této sekci se zaměříme na způsob, jakým se mění vlastnosti upravených stegosystémů oproti těm původním. Potřebujeme zjistit, jestli jsou výše popsané modifikace vůbec smysluplné. Měli bychom ukázat, že se po našich změnách stegosystémy nestanou neefektivní. Vyjdeme z vlastností odvozených v kapitole 2 a ukážeme, jakým způsobem se jejich hodnoty naší modifikací změnily.

Je zřejmé, že se naší modifikací zhorší relativní kapacita. Pro vkládání zprávy totiž budeme potřebovat delší nosič. Z definice relativní kapacity je zřejmé, že aplikací duhových kódů se její hodnota sníží d krát. Délka nosiče se totiž právě tolikrát zvětšila. Pro q–ární kód s relativní kapacitou α_1 získáme modifikací pomocí duhových kódů stegosystém s relativní kapacitou

$$\alpha = \frac{\alpha_1}{\frac{q-1}{2}} = \frac{2\alpha_1}{q-1}. \tag{4.3}$$

V případě efektivity se obecná změna určuje složitěji. Jelikož již definice efektivity záleží na zvolené distorzi, bude na ní závislá i tato změna. Z definice efektivity je zřejmé, že hodnota čitatele zůstane shodná. Střední délka zprávy se totiž nemění.

Ve jmenovateli je ale situace jiná. Z definice 1.3.6 víme, že jmenovatel je roven

$$\mathrm{E}_{x,m}(\mathrm{d}(x, y)).$$

V případě stegosystému, který je modifikovaný pomocí duhových kódů, hodnota efektivity nezáleží na zvolené distorzi. Provádíme totiž pouze změny velikosti 1. U původních stegosystémů se naopak efektivita liší podle použité distorze.

Obecně se ale efektivita stegosystémů s použitím duhových kódů zvyšuje. Počet prováděných změn totiž zůstává stejný zatímco amplituda těchto změn je menší.

Příklad 4.3.1. V tomto příkladu využijeme Hammingův q–ární kód s parametry $\left(\frac{q^r-1}{q-1}, \frac{q^r-1}{q-1} - r\right)$. Nebudeme se soustředit přímo na způsob vkládání a extrakce. Místo toho se zaměříme na vlastnosti, které bude takto zkonstruovaný stegosystém mít.

Původně uvažovaný stegosystém vkládá $(r \log_2 q)$ bitů zprávy do nosiče délky $\frac{q^r-1}{q-1}$. Náš modifikovaný stegosystém tedy bude vkládat stejně velkou zprávu do nosiče délky $\frac{q^r-1}{q-1} d \overset{(4.2)}{=} \frac{q^r-1}{2}$.

Tím tedy dostáváme relativní kapacitu

$$\alpha = \frac{2\frac{r(q-1)\log_2 q}{q^r-1}}{q-1} = \frac{2r \log_2 q}{q^r-1}.$$

Výpočet můžeme ještě ověřit dosazením do vztahu (4.3).

Jelikož mají Hammingovy kódy krycí poloměr $\mathrm{R} = 1$, provádí se během vkládání nejvýše jedna změna. V případě původního nosiče je velikost změny nejvýše $\left\lceil \frac{q}{2} \right\rceil$. U modifikovaného stegosystému provádíme změny velikosti ± 1.

Když budeme dosazovat do definice efektivity, dostaneme v čitateli hodnotu $(r \log_2 q)$. Jmenovatel pak můžeme vyjádřit jako

$$\mathrm{E}_{x,m}(\mathrm{d}(x,y)) = \frac{1 \cdot 0 + (q^r-1) \cdot 1}{q^r} = 1 - q^{-r}.$$

Tím dostáváme celkovou efektivitu

$$\mathrm{e} = \frac{r \log_2 q}{1 - q^{-r}}.$$

Abychom si mohli výsledné hodnoty lépe představit, jsou v tabulce 4.1 zaznamenány hodnoty relativní kapacity a efektivity pro některé Hammingovy kódy s parametry $\left(\frac{q^r-1}{q-1}, \frac{q^r-1}{q-1} - r\right)$.

♣

q	r	α	e
5	2	0,387	4,837
5	3	0,112	7,022
7	2	0,234	5,732
7	5	0,002	14,038
11	2	0,115	6,977
11	3	0,016	10,386
17	2	0,057	8,203
17	3	0,005	12,265

Tabulka 4.1: Parametry stegosystémů vytvořených z kódů $\mathcal{H}_{q,r}$ pomocí duhového obarvení.

Z tabulky 4.1 je ihned zřejmé, že dostáváme stegosystémy s vysokou hodnotou efektivity. Relativní kapacita je ale na druhou stranu nízká. Pokud ale hodnoty porovnáme s hodnotami, kterých jsme dosáhli v kapitole 2, zjistíme,

že jsme oproti stegosystémům vytvořených z binárních kódů dosáhli určitého vylepšení. Je ale nutné poznamenat, že v praxi bude využití stegosystémů s vysokými hodnotami q a r složité.

Vezměme si jako příklad stegosystém, kde $q = 7$ a $r = 5$. Přesto, že 5 ani 7 nejsou nijak vysoké hodnoty, budeme každých 5 prvků zprávy ze $\mathbb{Z}_7$ vkládat do bloku nosiče délky 8403. Pro poslání relativně krátké zprávy bychom tedy museli mít dlouhý nosič. Pokud ale budeme jako nosič využívat video, je i takovýto stegosystém možný.

V této kapitole jsme popsali, jakým způsobem se dají vylepšit stegosystémy vytvořené z q–árních kódů. Tím je dosaženo nižší detekovatelnosti, a tím i lepšího stegosystému, protože nízká pravděpodobnost odhalení je asi nejdůležitější vlastnost ve steganografii.

Kapitola 5

Závěr

V této kapitole shrneme dosavadní poznatky a předvedeme částečné vylepšení předchozí metody. Ukážeme, v jakém případě se hodí který stegosystém. Záleží totiž na tom, která vlastnost je pro nás v danou chvíli nejdůležitější.

5.1 Zobecnění metody duhových grafů

V kapitole 4 jsme použili duhové grafy k modifikaci q–árních stegosystémů. Pomocí této metody jsme eliminovali nutnost provádět velké změny. Jak již ale bylo řečeno, tímto způsobem se příliš zvětšuje délka potřebného nosiče.

Budeme-li chtít využít nějaký q–ární stegosystém pro vysokou hodnotu q, stane se potřebná délka nosiče neúnosnou. V případě využití duhových kódů se délka nosiče zvětšuje $\frac{q-1}{2}$ krát. Budeme-li například chtít využít stegosystém pro $q = 17$, délka potřebného nosiče se zvětší 8 krát. To znamená, že relativní kapacita stegosystému bude rázem velmi nízká.

Pokud bychom chtěli prodlužování nosiče nějakým způsobem omezit, je možné využít součtové a rozdílové pokrývací množiny. Zaměříme-li se opět na případ, kdy $q = 17$, bude nám stačit prodloužit nosič jen 3 krát. Využijeme totiž množinu $\mathrm{A} = \{1, 2, 6\}$, což je součtová a rozdílová pokrývací množina s parametry $(3, 2, 17)$.

Pro využití této SDCS budeme postupovat následovně:

- Nosič rozdělíme na bloky délky $n = 3$.
- Každý z těchto bloků budeme brát jako jeden prvek velikosti $M = 17$.
- Zjistíme potřebné změny pomocí q–árního stegosystému pro $q = M = 17$.
- Jednotlivé prvky, které jsou potřeba změnit, jsou nyní reprezentovány bloky. Pro změny tedy využijeme součtovou a rozdílovou pokrývací

množinu A. Pro každý měněný blok tak uděláme nejvýše 2 změny velikosti 1.

Tato metoda je založena na stejném principu jako metoda popsaná v kapitole 4. Navíc je ale možné zvýšit relativní kapacitu, protože budeme pro vkládání využívat kratší nosič. To nám samozřejmě trochu sníží efektivitu, protože budeme muset provádět více změn.

Nyní si ukážeme, jakým způsobem se změní vlastnosti stegosystému. Pro q–ární stegosystém potřebujeme součtovou a rozdílovou pokrývací množinu s parametry (n, k, M), kde $M = q$.

Z definice relativní kapacity lze snadno odvodit, že využitím SDCS se její hodnota sníží n krát. Máme-li tedy q–ární stegosystém s relativní kapacitou $\alpha = \alpha_1$, pak výsledná relativní kapacita bude

$$\alpha = \frac{\alpha_1}{n}.$$

Pro efektivitu je výpočet opět složitější. Hodnota čitatele zůstane opět stejná. Pro výpočet jmenovatele můžeme vyjít z výpočtu pro původní q–ární stegosystém. Naše metoda nahrazuje každou změnu původního stegosystému změnami v odpovídajícím bloku. Při výpočtu efektivity tedy můžeme místo každé změny v původním stegosystému použít vztah pro $\mathrm{E}(\mathrm{d}(x, y))$ ze strany 49.

My ale budeme muset tento vztah částečně modifikovat. V našem případě již máme jistotu, že se příslušný blok bude měnit. Proto musíme z výpočtu vyloučit nulovou změnu. Průměrná distorze měněného bloku je tedy rovna

$$\frac{1}{M-1} \sum_{0 \neq z \in \mathbb{Z}_M} \mathrm{d_A}(z).$$

V případě, že se jedná například o stegosystém odvozený z Hammingova kódu, je potřeba provádět nejvýše jednu změnu, protože krycí poloměr $R = 1$. Jak jsme již několikrát zmínili, Hammingův kód má parametry $\left(\frac{q^r-1}{q-1}, \frac{q^r-1}{q-1} - r\right)$. Jmenovatel efektivity stegosystému vytvořeného z tohoto kódu se pak změní na

$$\frac{q^r-1}{q^r} \frac{1}{M-1} \sum_{0 \neq z \in \mathbb{Z}_M} \mathrm{d_A}(z),$$

čímž dostaneme výslednou efektivitu

$$\mathrm{e} = \frac{r \log_2 q}{\frac{q^r-1}{q^r} \frac{1}{q-1} \sum\limits_{0 \neq z \in \mathbb{Z}_M} \mathrm{d_A}(z)} = \frac{q^r(q-1) r \log_2 q}{(q^r-1) \sum\limits_{0 \neq z \in \mathbb{Z}_M} \mathrm{d_A}(z)}.$$

Ukažme si nyní dosažené výsledky pro konkrétní stegosystém. Vezměme tedy například stegosystém vytvořený z q–árního Hammingova kódu. Pro

náš případ budeme uvažovat $q = 17$. Z předchozích výpočtů a z příkladu 3.3.2 můžeme vyjádřit efektivitu jako

$$\mathrm{e} = \frac{17^r(17-1)r\log_2 17}{(17^r-1)(6\cdot 1+10\cdot 2)}.$$

Pokud si tuto hodnotu vyjádříme pro $r = 3$, dostaneme $\mathrm{e} \approx 7,548$. Relativní kapacitu pak můžeme vyjádřit jako

$$\alpha = \frac{(q-1)r\log_2 q}{(q^r-1)n}.$$

Jelikož využíváme SDCS s parametry $(3, 2, 17)$, máme $n = 3$. Vyjádříme-li hodnotu opět pro $r = 3$, dostaneme $\alpha \approx 0,0133$.

Pokud bychom na stejný stegosystém využili metodu duhových grafů, dostali bychom podle příkladu 4.3.1 hodnoty $\mathrm{e} \approx 12,265$ a $\alpha \approx 0,00499$. Efektivita se tedy sice snížila přibližně $1,6$ krát, ale relativní kapacita vzrostla asi $2,6$ krát.

Abychom získali lepší představu o tom, jak se mění vlastnosti stegosystémů pro vyšší hodnoty q, můžeme se podívat do tabulky 5.1, kde je znázorněn případ pro $q = 53$. Zde se využívá SDCS $\mathrm{A} = \{1, 2, 6, 18\}$ s parametry $(4, 3, 53)$.

r	α (duhové kódy)	e (duhové kódy)	α (SDCS)	e (SDCS)
2	0,008	11,460	0,053	5,050
3	0,00023	17,184	0,0015	7,573
4	$5,81 \cdot 10^{-6}$	22,912	0,0000377	10,097
5	$1,37 \cdot 10^{-7}$	28,640	$8,90 \cdot 10^{-7}$	12,621

Tabulka 5.1: Parametry stegosystémů odvozených z q–árních Hammingových kódů pro $q = 53$ s využitím SDCS a duhových grafů.

Aplikací součtových a rozdílových pokrývacích množin na q–ární kódy jsme dali možnost výběru mezi efektivitou a relativní kapacitou. Nutno poznamenat, že účinnost této metody závisí na q–árním stegosystému, který využíváme, ale hlavně na efektivitě SDCS, kterou jsme schopni sestrojit.

5.2 Shrnutí

V této sekci provedeme srovnání popsaných stegosystémů.

Asi nejznámější moderní stegosystém je maticové vkládání popsané v kapitole 2. Tato metoda využívá teorii samoopravných kódů a dosahuje tak mnohem lepších výsledků než základní postupy.

Při zkoumání maticového vkládání jsme zjistili, že je nevýhodné provádět velké změny, protože je pak stegosystém snadněji detekovatelný. Proto jsme

se v kapitole 3 zaměřili na provádění změn velikosti 1. Při tomto omezení jsme se snažili do nosiče vložit co možná nejdelší zprávu. Proto mají stegosystémy vytvořené pomocí součtových a rozdílových pokrývacích množin vysokou hodnotu relativní kapacity. Pokud je tedy potřeba stegosystém s vysokou relativní kapacitou, jedná se o vhodnou metodu. Na druhou stranu jsme také ukázali, že sestrojení SDCS není jednoduché a existence některých pokrývacích množin je stále otevřenou otázkou.

V kapitole 4 jsme zkusili problém velkých změn vyřešit jiným způsobem. Jednotlivé prvky stegosystému jsme nahradili celými bloky. Pomocí těchto bloků jsme velké změny nahradili změnami velikosti 1 a opět jsme tak využili q–ární stegosystémy. Tímto způsobem jsme dosáhli vysoké efektivity. Na druhou stranu nám ale vzrostla potřebná délka nosiče. Tím tedy klesla i relativní kapacita. Zjistili jsme, že potřebná délka nosiče pro vysoké hodnoty q roste mnohonásobně a mají tak horší využití v praxi.

V minulé sekci jsme zkusili nahradit metodu duhových kódů za součtové a rozdílové pokrývací množiny. Je tak možné zkrátit požadovanou délku nosiče a zároveň využít vlastností q–árních kódů. Tím se zvýší hodnota relativní kapacity. Bohužel ale zároveň klesla hodnota efektivity.

Literatura

[1] J. Bierbrauer and J. Fridrich. Constructing good covering codes for applications in steganography. *Transactions on data hiding and multimedia security III*, pages 1–22, 2008.

[2] J. Fridrich. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, 2009.

[3] J. Fridrich and P. Lisonek. Grid colorings in steganography. *Information Theory, IEEE Transactions on*, 53(4):1547–1549, 2007.

[4] J. Fridrich, P. Lisoněk, and D. Soukal. On steganographic embedding efficiency. In *Information Hiding*, pages 282–296. Springer, 2007.

[5] J. Fridrich and M. Long. Steganalysis of lsb encoding in color images. In *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on*, volume 3, pages 1279–1282. IEEE, 2000.

[6] F. Galand and G. Kabatiansky. Information hiding by coverings. In *Information Theory Workshop, 2003. Proceedings. 2003 IEEE*, pages 151–154. IEEE, 2003.

[7] J.I. Hall. *Notes on coding theory*. FreeTechBooks. com, 2003.

[8] X. Li, T. Zeng, and B. Yang. Improvement of the embedding efficiency of lsb matching by sum and difference covering set. In *Multimedia and Expo, 2008 IEEE International Conference on*, pages 209–212. IEEE, 2008.

[9] Brian J Shelburne. Balanced (signed) ternary notation.

[10] U. Tamm. Integer codes in coding and computing.

[11] M. van Dijk and F. Willems. Embedding information in grayscale images. In *Proceedings of the 22nd Symposium on Information and Communication Theory in the Benelux, Enschede, The Netherlands*, pages 147–154, 2001.

[12] J.H. van Lint. A survey of perfect codes. *Rocky Mountain J. Math*, 5(2):199–226, 1975.

[13] A. Westfeld. F5—a steganographic algorithm. In *Information Hiding*, pages 289–302. Springer, 2001.

[14] M. Woo, J. Neider, T. Davis, and D. Shreiner. *OpenGL programming guide: the official guide to learning OpenGL*. Addison-Wesley Longman Publishing Co., Inc., 1999.

[15] M.Y. Wu, Y.K. Ho, and J.H. Lee. An iterative method of palette-based image steganography. *Pattern Recognition Letters*, 25(3):301–309, 2004.

[16] X. Zhang and S. Wang. Vulnerability of pixel-value differencing steganography to histogram analysis and modification for enhanced security. *Pattern Recognition Letters*, 25(3):331–339, 2004.