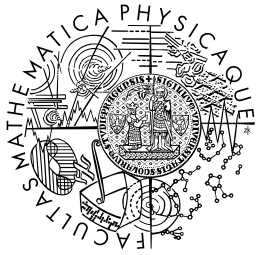


MATEMATICKO-FYZIKÁLNÍ
FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Markéta Calábková

Expanderové kódy

Katedra aplikované matematiky

Vedoucí bakalářské práce:	Mgr. Martin Mareš, Ph.D.
Studijní program:	Matematika
Studijní obor:	Matematické metody informační bezpečnosti

Praha 2018

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) 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 má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ..... Podpis autora

Název práce: Expanderové kódy

Autor: Markéta Calábková

Katedra: Katedra aplikované matematiky

Vedoucí práce: Mgr. Martin Mareš, Ph.D., Katedra aplikované matematiky

Abstrakt: Kdekoliv se přenáší nějaká informace, jsou přítomny samooopravné kódy. Mezi nejpoužívanější třídy kódů patří LDPC (low density parity check) kódy. Expanderové kódy jsou jednou z nadějných tříd LDPC kódů. V této práci vysvětlujeme, co to expanderové kódy vlastně jsou, a ukazujeme, že je lze opravdu konstruovat tak, aby dosahovaly asymptoticky optimálních parametrů a zároveň je šlo dekódovat v čase lineárním s délkou zprávy. Bohužel uvedené konstrukce vytvářejí hodně dlouhé kódy, takže jsou v běžném provozu (například pro přenos paketů) prakticky nepoužitelné. Věříme ale, že s využitím lepší konstrukce expanderů budeme schopni sestrojit dobré krátké kódy, které najdou mnohá využití.

Klíčová slova: expander, samoopravný kód, efektivní dekódování

Title: Expander codes

Author: Markéta Calábková

Department: Department of applied mathematics

Supervisor: Mgr. Martin Mareš, Ph.D., Department of applied mathematics

Abstract: Wherever information is transmitted we can find error-correcting codes. LDPC (low density parity check) codes are one of frequently used classes of codes and expander codes are promising members of this class. In this work, we explain what expander code are. We also show that expander codes simultaneously have both asymptotically optimal parameters and linear-time encoding and decoding. Unfortunately, our constructions grant us codes, which are too big for regular use, for example for packet transmission. However, we believe that with better construction of expander graphs we will be able to construct short codes with significant practical applications.

Keywords: expander, error-correcting code, linear-time decoding

Na tomto místě bych chtěla poděkovat svému vedoucímu, Mgr. Martinu Marešovi, Ph.D., za poskytnuté, ač místy Medvědí, služby. Bez něj by tato práce nevznikla. Také bych chtěla poděkovat svému příteli, rodině a v neposlední řadě přátelům, kteří mě motivovali k psaní a zároveň se starali o to, abych na tuto práci neměla příliš mnoho času.

Ze všeho nejvíc ale chci poděkovat všem zmíněným, že tu prostě byli pro mě.

Obsah

Úvod

Kdekoliv se přenáší nebo uchovává nějaká informace, jsou přítomny i chyby. Z toho důvodu se používají samoopravné kódy, které umí chyby detekovat a případně je i opravit. Známe mnoho tříd kódů, jedny z nejpoužívanějších jsou LDPC kódy (zkratka z Low-Density Parity-Check), které vypadají tak, že se vezme nějaký bipartitní graf, jedna partita jsou bity zprávy a druhá partita jsou paritní bity. Low-density je v názvu proto, že tyto kódy mají řídkou kontrolní matici (to znamená, že vrcholy z kontrolní partity mají poměrně nízký stupeň, tedy graf je celkové řídký). Dlouho ale nebyl znám žádný postup, jak je zkonstruovat, většinou se jen vzal nějaký řídký graf a až používáním se zjistilo, co má kód za parametry.

Má-li mít LDPC kód dobré vlastnosti, je potřeba, aby byl celkově „dobře propojený“. Grafy s těmito vlastnostmi se zkoumají i z jiných důvodů a používají se v mnoha odvětvích matematiky. Říká se jim expandery a dříve byla jejich existence dokázána pouze nekonstruktivně. Nyní je již známo několik konstrukcí, mimo jiné založených na cikcakovém součinu grafů.

Když už umíme konstruovat expandery, přicházejí na řadu expanderové kódy, o kterých se ví, že mohou dosahovat hodně dobrých parametrů, speciálně, že jsou asymptoticky optimální (tedy pro dlouhé zprávy dosahují parametrů blízkých optimu). Veškeré znalosti o nich jsou ale roztroušeny v několika článcích, které se ne vždy shodnou na značení.

Cílem této práce je podat ucelený přehled, co se o těchto kódech ví. Chceme se naučit teoreticky konstruovat expandery a z nich i kódy, dokázat, že expanderové kódy jsou asymptoticky optimální a nakonec i sestrojit konkrétní příklad expanderového kódu a diskutovat jeho praktickou využitelnost.

1. Základy

V této kapitole vysvětlíme základní znalosti a principy, na kterých budeme stavět, spolu se zavedením značení, které se v práci budeme snažit dodržovat.

1.1 Základní značení

- V celé práci bude symbol $\mathbb{N}$ značit množinu přirozených čísel včetně nuly.
- Množinu přirozených čísel od jedné do n budeme značit $[n]$. Tedy $[n] = = \{1, \ldots, n\}$.
- Symbol $\mathbb{F}_q$, respektive $\mathbb{Z}_q$ značí konečné těleso o q prvcích, druhý způsob budeme používat pouze v případě, že je q prvočíslo.
- Velká písmena anglické i řecké abecedy značí matici, dolní index u matice většinou značí prvek na příslušné pozici.
- Vektory i skaláry značíme většinou malými písmeny řecké i anglické abecedy, z textu by mělo být jasné, o co se kdy jedná. V případech, kdy by mohlo dojít k nedorozumění, značíme vektory tučně.

1.2 Lineární algebra

Zde si krátce připomeneme několik základních pojmů. Pokud čtenář požaduje podrobnější vysvětlení, odkážeme ho na libovolnou učebnici lineární algebry (například Barto a Tůma (2016), z kteréžto knihy jsme převzali většinu definic uvedených v této sekci).

1.2.1 Afinní prostor

V jedné z konstrukcí budeme pracovat nad *afinním prostorem*, který je zhruba řečeno tvořen množinou bodů a množinou vektorů a má definované sčítání bodu a vektoru. Vznikne tak, že vezmeme podprostor vektorového prostoru (ten je tvořen čistě vektory a operacemi mezi nimi) a přičteme k němu vektor (ve vektorovém prostoru je to ekvivalent bodu), který může, ale nemusí, být z tohoto podprostoru (tedy zhruba řečeno můžeme posunout počátek souřadnic). Afinní prostor dimenze n značíme $\mathbb{A}^n$.

Rozdíl mezi vektorovým a afinním prostorem si vysvětlíme na příkladu. Vezměme si prostor $\mathbb{R}^2$. Jeho vektorové podprostory dimenze 1 jsou určeny jedním vektorem a operacemi, tedy jsou to právě přímky procházející počátkem. Na druhé straně jeho afinní podprostory jsou určeny bodem a vektorem a vzniknou (konzistentně s vysvětlením výše) tak, že nejprve ze samotného vektoru vytvoříme vektorový podprostor dimenze 1 a pak k němu přičteme bod (neboli vektor jdoucí z počátku do toho bodu). Tedy afinní podprostory dimenze jedna jsou všechny přímky (můžeme posunout počátek, kam chceme).

1.2.2 Vlastní čísla

Na tomto místě připomeneme několik pojmů ohledně matic a vektorů.

Nejprve značení: operátor $\langle\cdot,\cdot\rangle$ odpovídá *standardnímu skalárnímu součinu* vektorů v $\mathbb{R}$. *Norma* vektoru $\mathbf{v}$ je definována jako $\|\mathbf{v}\| = \sqrt{\langle\mathbf{v},\mathbf{v}\rangle}$.

Pro matici M značí M^T matici *transponovanou*, tedy je-li M typu $m \times n$, je M^T typu $n \times m$ a pro všechna $i \in [n]$, $j \in [m]$ platí $M^T_{ij} = M_{ji}$ (vlastně vyměníme svislou a vodorovnou osu). *Symetrické* jsou takové matice M, pro něž platí $M = M^T$.

Matice je *diagonální*, má-li nenulové prvky jen na diagonále a jinde nuly. Dva vektory u a v jsou *ortogonální*, jestliže jejich skalární součin je 0 (tedy jsou kolmé). Ortogonalitu budeme značit symbolem $\perp$. O $\mathbf{u}$ a $\mathbf{v}$ řekneme, že jsou *ortonormální*, jestliže jsou ortogonální a $\|\mathbf{u}\| = \|\mathbf{v}\| = 1$. O matici řekneme, že je *ortogonální*, jestliže její sloupce jsou navzájem ortonormální.

Regulární matice je taková čtvercová matice S, pro kterou platí, že máme-li nenulový vektor $\mathbf{v}$, potom $S\mathbf{v}$ je také nenulový vektor. Regulární matice lze invertovat, tedy existuje takové S^{-1}, že $SS^{-1} = S^{-1}S = I_n$.

Ortogonální matice je regulární, neboť její sloupce jsou lineárně nezávislé. Všimneme si, že pro ortonormální matici Q typu $n \times n$ platí $Q^{-1} = Q^T$, neboť $Q^TQ = I_n$ (na pozici ij je skalární součin i-tého řádku první matice a j-tého sloupce druhé matice, ale Q je ortogonální a její sloupce odpovídají řádkům Q^T).

Dvě matice A a B jsou *podobné*, jestliže existuje regulární matice S taková, že $A = SBS^{-1}$, tedy jestliže se A po změně kanonické báze na S změní na B. Matice A je *diagonalizovatelná*, pokud je podobná nějaké diagonální matici Λ, tedy $A = S\Lambda S^{-1}$. Tomuto tvaru se říká *spektrální rozklad*.

Definice 1 (Vlastní čísla)**.** *Je-li A čtvercová matice řádu n nad tělesem T, pak skalár $\lambda \in T$ nazýváme vlastní číslo matice A, pokud existuje nenulový vektor $\mathbf{v} \in T^n$ takový, že*

$$A\mathbf{v} = \lambda\mathbf{v}\,.$$

Ale jak je zjistit? Na to se hodí charakteristický polynom.

Definice 2. *Je-li A čtvercová matice řádu n nad tělesem T, pak charakteristický polynom matice A je polynom*

$$\chi_A = \det(A - \lambda I_n)$$

v proměnné λ.

To, že zde proměnnou nazýváme λ, není náhoda. Kořeny tohoto polynomu jsou právě vlastní čísla matice A, to zde ovšem dokazovat nebudeme.

V této práci se budeme zabývat hlavně maticemi nad tělesem $\mathbb{R}$, které není algebraicky uzavřené. Polynom χ_A zřejmě může mít některé kořeny komplexní, tedy reálná matice může mít některá vlastní čísla komplexní. Definujme *násobnost* vlastního čísla jako násobnost příslušného kořenu v χ_A. Potom vidíme, že každá komplexní matice (tedy všechny matice, s kterými se v této práci setkáme) má právě n vlastních čísel včetně násobností.

Ne všechny vektory ale po vynásobení maticí odpovídají svému λ-násobku. Rozšiřme tedy definici vlastních čísel o:

Definice 3 (vlastní vektor)**.** *Je-li λ vlastním číslem matice A, pak libovolný vektor $\mathbf{v}$, pro který platí $A\mathbf{v} = \lambda\mathbf{v}$, nazýváme vlastní vektor matice A příslušný vlastnímu číslu λ.*

Jeden příklad za všechny: jednotkovou matici typu $n \times n$ značíme I_n a všimneme si, že ta má jediné vlastní číslo 1 násobnosti n a jejími vlastními vektory jsou všechny nenulové vektory délky n.

Jednou ze základních vlastností vlastních čísel je to, že je-li $\mathbf{v}$ vlastním vektorem matice A s vlastním číslem λ_A a vlastním vektorem matice B s vlastním číslem λ_B, potom můžeme na těchto maticích provádět různé aritmetické operace (sčítat, odčítat, násobit, mocnit nebo tyto operace kombinovat) a $\mathbf{v}$ bude vlastním vektorem výsledku s vlastním číslem λ_{AB}, které získáme tak, že se ty samé operace provedou na λ_A a λ_B.

V této práci se budeme hlavně zabývat maticemi sousednosti grafů, které jsou symetrické. Pro symetrické matice a jejich vlastní čísla platí několik užitečných tvrzení. Tato tvrzení uvedeme bez důkazu, zájemce odkážeme na Hladík (2017).

Věta 1 (Vlastní čísla symetrických matic)**.** *Vlastní čísla reálných symetrických matic jsou reálná.*

Tato věta nám říká, že reálná symetrická matice typu $n \times n$ má n reálných vlastních čísel včetně násobností. To je milá vlastnost a bude se nám v práci hodit.

Další užitečná věta je, že pro reálnou symetrickou matici existuje ortonormální báze složená z jejích vlastních vektorů.

Věta 2 (Spektrální rozklad symetrických matic)**.** *Pro každou symetrickou matici $A \in \mathbb{R}^{n\times n}$ existuje ortogonální $Q \in \mathbb{R}^{n\times n}$ a diagonální $\Lambda \in \mathbb{R}^{n\times n}$ tak, že $A = Q\Lambda Q^T$.*

Tato věta říká to, že matice A je podobná nějaké diagonální matici. Ekvivalentně, vyjádříme-li A v nějaké ortonormální bázi, dostaneme diagonální matici. Jsou-li dvě matice podobné, mají stejná vlastní čísla, a matice Λ je diagonální, tedy její vlastní vektory jsou vektory kanonické báze. V tom případě ale bázové vektory té nové báze musí být i vlastními vektory matice A.

Ještě nahlédneme, že přechod mezi ortonormálními bázemi zachovává normu. Jak známe z lineární algebry, násobení ortogonální maticí je ortogonální zobrazení, tedy zachovává skalární součin, a tím pádem zachovává i normu. Tedy to samé platí i pro převod mezi ortonormálními bázemi.

1.3 Grafy

Graf $G = (V,E)$ je obvykle zadán množinou vrcholů V a množinou hran E. Všechny grafy, kterými se v této práci budeme zabývat, jsou neorientované. Často se definuje, že hrana je dvojice vrcholů $\{a,b\}$, mezi nimiž vede. My ale budeme používat obecnější grafy, kde mezi dvěma vrcholy může vést více hran, ba dokonce může vést hrana z vrcholu do něj samého. Definujme tedy hranu jako abstraktní objekt s nějakými vlastnostmi, jako jsou například koncové vrcholy. Takto docílíme toho, že i hrany vedoucí mezi stejnými vrcholy budou rozlišitelné.

1.3.1 Matice sousednosti

Zadefinujeme matici sousednosti grafu.

Definice 4. *Matice sousednosti A grafu G na n vrcholech je čtvercová matice $n \times n$, kde na pozici ij je číslo $k \in \mathbb{N}$, pokud jsou i-tý a j-tý vrchol grafu G spojeny k hranami.*

Všimneme si, že A je symetrická, protože G je neorientovaný graf, tedy vedeli hrana z i-tého vrcholu do j-tého, povede i z j-tého do i-tého. Tato matice je užitečným nástrojem pro reprezentaci grafu. Nám se ale pro regulární grafy bude více hodit její normalizovaná verze.

Definice 5. *Normalizovaná matice sousednosti M d-regulárního grafu G je matice sousednosti G vydělená číslem d.*

Všimneme si, že protože G je regulární, je součet čísel v každém řádku i sloupci matice M roven 1. Z toho vidíme, že vektor samých jedniček je určitě vlastním vektorem M s vlastním číslem 1. Jak to vypadá s ostatními vlastními čísly? Zřejmě v M nejsou žádná záporná čísla, tedy všechna ostatní vlastní čísla budou v absolutní hodnotě menší nebo rovna 1.

Ve většině případů budeme chtít, aby platila ostrá nerovnost. Kdy nastane rovnost? Vyslovíme dvě pozorování o vlastních číslech matice M.

Lemma 3. *M má vlastní číslo -1 právě tehdy, když je G bipartitní.*

Důkaz. Příslušný vlastní vektor má jedničky na pozicích odpovídajícím vrcholům jedné partity a minus jedničky na zbytku. Naopak, má-li M jako vlastní čísla zároveň 1 a -1, musí G nutně být bipartitní. Tuto vlastnost dokážeme sporem. Pokud G není bipartitní, je v grafu cyklus liché délky, například mějme cyklus délky 3 mezi vrcholy s čísly i, j a k. Ve vlastním vektoru $\mathbf{v}$ příslušném -1 buď v_i (tedy prvek na i-té pozici) bez újmy na obecnosti třeba 1. To znamená, že $(M\mathbf{v})_i = -1$, tedy $v_j = -1$ (mezi i a j vede hrana, a protože $v_i = 1$ a $M\mathbf{v} = -\mathbf{v}$, skalární součin i-tého řádku M a vektoru $\mathbf{v}$ musí být -1, tedy $\mathbf{v}$ musí mít minus jedničky na pozicích, které jsou s i-tým vrcholem spojeny hranou, což platí i pro vrchol j). Tímto jsme došli ke sporu v k-tém řádku, protože skalární součin k-tého řádku M s vektorem $\mathbf{v}$ bude v absolutní hodnotě menší než 1. To platí, protože do k vedou hrany z vrcholů i i j, přičemž $v_i = 1$ a $v_j = -1$, tedy tyto dvě pozice se ve skalárním součinu navzájem odečtou. □

Lemma 4. *M má vlastní číslo 1 s větší násobností než 1 právě tehdy, když má G více komponent souvislosti.*

Důkaz. Implikace $\Leftarrow$ je zřejmá, tedy dokazujme opačnou implikaci. Pokud má M násobné vlastní číslo 1, najdeme pro něj nějaký vlastní vektor $\mathbf{v}$ lineárně nezávislý na vektoru samých jedniček. Nyní si všimneme, že jakákoliv lineární kombinace vlastních vektorů od téhož vlastního čísla je zase vlastní vektor. Takže můžeme přičtením vhodného násobku vektoru samých jedniček k vektoru $\mathbf{v}$ získat vlastní vektor, který je nezáporný a obsahuje aspoň jednu nulu (a není celý nulový). Teď už stačí uvážit množinu vrcholů, na jejichž pozicích jsou ve vektoru nuly, a uvědomit si, že z této množiny nemůže vést žádná hrana ven. □

Všimněme si, že M je také maticí, ve které jsou pro jednotlivé hrany pravděpodobnosti přechodu náhodné procházky na G, jinými slovy na pozici ij je pravděpodobnost, že se z i-tého vrcholu pohneme do j-tého vrcholu, pokud se budeme v každém vrcholu rozhodovat náhodně. Brzy se nám tento pohled bude hodit.

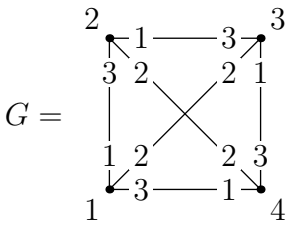
1.3.2 Rotační mapa

Taktéž si pro graf zadefinujme jeho rotační mapu. Tu můžeme pochopit tak, že každému vrcholu přidělí očíslování hran z něj vedoucích. Není ale jasné, proč se nazývá právě „rotační mapa“. Tato tabulka historicky vychází z takzvaného rotačního systému, který se definuje pro rovinné grafy a popisuje očíslování hran kolem vrcholu (tedy rotuje kolem vrcholu a čísluje hrany, stručné vysvětlení a odkazy na články lze nalézt na anglické Wikipedii[1]), Reingold, Vadhan a Wigderson (2002) definici rotační mapy mírně upravili.

Proč ji ale vlastně definujeme a nestačí nám matice sousednosti? Rotační mapa je jednak menší než matice sousednosti, jednak poskytuje o grafu trochu více informací a v neposlední řadě se s ní dobře pracuje, například se hodí při výpočtech v algoritmech.

Definice 6. *Mějme d-regulární graf G s počtem vrcholů n. Jeho rotační mapu* $\mathrm{Rot}_G : [n] \times [d] \to [n] \times [d]$ *definujeme následovně:* $\mathrm{Rot}_G(v,i) = (w,j)$*, pokud i-tá hrana z vrcholu v vede do w a tato hrana je zároveň j-tou hranou z w.*

Následující obrázek ilustruje, jak vypadá rotační mapa grafu K_4.



$$\mathrm{Rot}_G = \begin{pmatrix} (2,3) & (3,2) & (4,1) \\ (3,3) & (4,2) & (1,1) \\ (4,3) & (1,2) & (2,1) \\ (1,3) & (2,2) & (3,1) \end{pmatrix}$$

Nahlédněme, že definice je korektní. Zřejmě u každého vrcholu umíme očíslovat odchozí hrany čísly od 1 do d, a protože graf je neorientovaný, z cílového vrcholu určitě vede (ta samá) hrana nazpět, jen má obecně jiné číslo.

Z definice vidíme, že Rot_G je permutace na $V \times [d]$. Tato permutace se skládá ze spousty disjunktních dvojcyklů, tedy je to involuce. Z toho snadno plyne, že $\mathrm{Rot}_G \circ \mathrm{Rot}_G$ je identita (tedy $(\mathrm{Rot}_G \circ \mathrm{Rot}_G)(v,i) = (v,i)$ pro všechna $v \in V, i \in [d]$).

Budeme ve spojení s rotační mapou občas mluvit o „efektivní konstrukci“ (buď rotační mapy, nebo samotného grafu). Pro nás bude tento pojem znamenat to, že umíme vyhodnotit Rot_G v čase polylogaritmickém s velikostí grafu, tedy polynomiálním s bitovou délkou argumentů.

1.4 Samoopravné kódy

Zde uvedeme základní přehled o samoopravných kódech. Čerpali jsme hlavně ze skript Drápal (2008/2009), ale také z Guruswami (2010).

[1] `https://en.wikipedia.org/wiki/Rotation_system`

Formálně řečeno, kód je množina kódových slov. Když mluvíme o kódech, myslíme tím blokový kód, tedy takový, kde všechny řetězce, které kódujeme, mají stejnou délku k a všechny zakódované řetězce mají délku n. Symboly kódu volíme z nějaké abecedy Σ. Většinou chceme, aby Σ byla konečné těleso. Často se volí $\Sigma = \mathbb{Z}_2$ (takovým kódům se pak říká *binární*), ale občas chceme Σ i větší, třeba s velikostí 2^k pro nějaké k, protože takto se pohodlně reprezentují k-tice bitů.

Definice 7. *Mějme kód nad abecedou Σ, potom kódování je funkce $E : \Sigma^k \to \Sigma^n$ a dekódování funkce $D : \Sigma^n \to \Sigma^k$ takové, že $D(E(u)) = u$ pro každý řetězec $u \in \Sigma^k$. Kódové slovo je takový řetězec, který lze získat jako výstup funkce E.*

Číslu n budeme říkat *délka kódu*, číslu k jeho *dimenze*.

Chyba je špatně přenesený znak kódu. Pokud se místo jednoho znaku přenese znak jiný, typicky je to zapříčiněno nějakým šumem na kanálu, a proto se takovému kanálu říká zašuměný a kódům, které toto řeší, budeme říkat *šumové kódy*. Také se místo znaku může přenést otazník, tedy víme, že na tomto místě byl nějaký znak a zrovna ho neumíme přečíst (to se stane, když například na počítači odejde část disku). Takovým kódům se říká *mazací kódy*. Také existují kódy, které znaky přidávají nebo odebírají, těm se říká *insdel kódy* z anglického insertion–deletion (tohle umí dobře opravit většina přirozených jazyků při mluveném slově), ale o těch nebudeme moc mluvit.

Nejběžnější jsou kódy šumové, tedy když v této práci řekneme „kód“, myslíme tím šumový.

Když dekódujeme nějaký řetězec, většinou hledáme kódové slovo, které je mu nejblíž. Kódová slova si představme rozmístěná v prostoru Σ^n. Každé slovo v má kolem sebe (ne nutně disjunktní) oblast, která se dekóduje na v. Chceme, aby pro každé slovo byla tato oblast co největší, abychom mohli opravit co nejvíce chyb.

Mějme tedy prostor Σ^n, *vzdálenost* kódových slov buď počet pozic v kódu, na kterých se liší. Pokud nastalo nejvýše r chyb, leží přijaté slovo ve vzdálenosti r od správného kódového slova, tedy leží v tzv. *Hammingově kouli* o poloměru r kolem správného kódového slova. Chceme-li, aby kód opravoval r chyb, musí mít všechny Hammingovy koule poloměr alespoň r.

V této práci se budeme většinou zabývat kódy s jednoznačným dekódováním, tedy každý řetězec na n znacích se dekóduje jen na jedno kódové slovo, neřekneme-li jinak. Tedy koule kolem kódových slov musí být disjunktní a každá dvě kódová slova musí mít vzdálenost alespoň $d = 2r + 1$. Také ve většině aplikací budeme chtít, aby měly všechny koule stejný poloměr. Číslu d se říká *kódová vzdálenost*, je to vlastně dolní mez na vzdálenost dvou kódových slov. Číslo r značí počet opravitelných chyb.

Snadno nahlédneme, že počet prvků Hammingovy koule pro kód délky n nad abecedou Σ o velikosti q, který opraví až r chyb, je

$$V_q(n,r) = \sum_{i=0}^{r} \binom{n}{i} (|\Sigma| - 1)^i \,,$$

tedy pro binární kód je to prostě součet kombinačních čísel

S kódovou vzdáleností úzce souvisí také *váha slova*, což je počet nenulových znaků. *Váha kódu* je váha nejmenšího nenulového slova.

Většinou se používají *lineární kódy*, to znamená, že součet jakýchkoliv dvou kódových slov je také korektní kódové slovo (tedy kódová slova tvoří vektorový podprostor prostoru Σ^n). Je vidět, že v lineárním kódu se vždy vyskytuje nulové slovo, a tedy že kódová vzdálenost je rovna váze kódu (v kódu leží i rozdíl nejméně vzdálených kódových slov).

Pro parametry kódů se často používá následující úsporné značení: kód délky n dimenze k s kódovou vzdáleností d označme (n,k,d)-*kód*. Je-li tento kód navíc nad q-ární abecedou, označme jej $(n,k,d)_q$-*kód*. Analogicky značme lineární kódy $[n,k,d]$, respektive $[n,k,d]_q$-kódy.

1.4.1 Generující a kontrolní matice

Pro lineární kódy platí, že existuje funkce E, která je také lineární, tedy je lze generovat maticí a maticí lze i ověřit, zda jsme přijali korektní kódové slovo. *Generující matice* je typu $k \times n$, *kontrolní* typu $n \times (n-k)$. Generující matice obsahuje jako sloupce nějakou bázi prostoru kódových slov (tedy se z ní dá jednoznačně odvodit E), naopak kontrolní matice obsahuje v řádcích nějakou bázi ortogonálního doplňku prostoru kódových slov (tedy řádky kontrolní matice jsou kolmé na všechna kódová slova a jsou lineárně nezávislé). Generující maticí vynásobíme řetězec délky k a dostaneme příslušné kódové slovo, kontrolní maticí vynásobíme řetězec délky n a pokud vyjde řetězec složený ze samých nul, bylo to korektní kódové slovo. V opačném případě nám získaná informace může nějak pomoci v opravě.

1.4.2 Odhady

Intuitivně je dost jasné, že kód, který má velkou dimenzi, nemůže mít zároveň moc velkou kódovou vzdálenost. Následující odhady formalizují, co od samoopravných kódů ještě můžeme požadovat a co už je nemožné. Začneme jednoduchým, ač důležitým odhadem:

Věta 5 (Singletonův odhad)**.** *Ať C je $(n,k,d)_q$-kód. Potom $d \leq n - k + 1$.*

Tento odhad udává horní mez na kódovou vzdálenost. Pro lepší představu o tom, co tento odhad vlastně říká, uveďme, že jednotlivé znaky kódových slov lze občas rozdělit na „kontrolní pozice“ a „pozice nesoucí informaci“ (kódům, které tohle splňují, se říká *systematické*). Potom nerovnost výše znamená, že kódy o kódové vzdálenosti d mají alespoň $d-1$ kontrolních pozic, což je intuitivně jasné. Kódy, které této horní meze dosahují, se nazývají MDS (maximum distance separable) kódy.

Věta 6 (Hammingova nerovnost)**.** *Pro kód C délky n o kódové vzdálenosti d nad abecedou o velikosti q platí*

$$|C| \cdot V_q\left(n, \left\lceil \frac{d-1}{2} \right\rceil\right) \leq q^n\,.$$

Tento odhad říká jen to, že máme-li kód C o kódové vzdálenosti d, mohou mít Hammingovy koule poloměr jen $\lceil (d-1)/2 \rceil$, protože jinak by se překrývaly.

Speciálně pro $(n,k,d)_q$-kódy (tedy $|C| = q^k$) má tato nerovnost tvar

$$V_q\left(n, \left\lceil \frac{d-1}{2} \right\rceil\right) \leq q^{n-k}\,.$$

Je-li Hammingova nerovnost rovností, kód C se nazývá *r-perfektní* pro r splňující $d = 2r+1$. Kód je *perfektní*, je-li r-perfektní alespoň pro nějaké r.

Věta 7 (Plotkinova mez)**.** *Ať C je (n,k,d)-kód takový, že $d > n/2$. Potom $k \leq$ $\leq 2d/(2d-1) \leq n+1$.*

Věta 8 (Gilbertova-Varšamovova nerovnost)**.** *Ať n, k a d jsou kladná celá čísla, $d \leq n$. Je-li $V_q(n,d-1) < q^{n-k+1}$, pak existuje q-ární $(n,k,d)_q$-kód.*

1.4.3 Asymptotické vlastnosti kódů

Pro mnoho tříd kódů nás bude zajímat, jak se chovají pro $n \to \infty$. Zde absolutní čísla (jako například kódová vzdálenost) o ničem nevypovídají, nadefinujme si tedy některé asymptotické veličiny.

Asi nejdůležitější asymptotickou veličinou je *nosnost kódu*. Nosnost kódu C často značíme $R(C)$ a je to poměr přenesené informace ku délce kódu, tedy k/n.

Dále definujme *relativní kódovou vzdálenost* jako $\delta(C)$, vypočítá se jako d/n.

Ve spojení s chybami definujme *chybovost* $\varepsilon(C)$, to znamená počet chyb vydělený n.

O třídě kódů řekneme, že je *asymptoticky dobrá*, pokud pro všechny kódy C z této třídy existují konstanty $R_0 > 0$ a $\delta_0 > 0$ takové, že

$$R(C) > R_0 \text{ a } \delta(C) > \delta_0\,.$$

V této práci se budeme snažit konstruovat třídy kódů, které jsou asymptoticky dobré.

Definice 8 (Entropická funkce)**.** *Mějme abecedu velikosti q. Pro libovolné q definujme entropickou funkci H_q, která pro každé $\alpha \in (0,1)$ nabývá hodnoty*

$$H_q(\alpha) = \alpha \log_q(q-1) - \alpha \log_q \alpha - (1-\alpha)\log_q(1-\alpha)\,.$$

Speciálně pro $q = 2$:

$$H(\alpha) = \alpha \log \frac{1}{\alpha} + (1-\alpha)\log\frac{1}{1-\alpha}\,.$$

Hodnota entropické funkce úzce souvisí s velikostí Hammingovy koule. Nechť $0 \leq p \leq 1 - 1/q$ je chybovost kódu. Pak platí $V_q(n,pn) \leq q^{nH_q(p)}$ a zároveň $V_q(n,pn) \geq q^{nH_q(p)-o(n)}$. To navádí na následující tvrzení:

Tvrzení 9 (Asymptotický Gilbertův-Varšamovovův odhad)**.** *Pro libovolné q a $0 < \delta < 1/q$ existuje q-ární kód C s nosností $R(C) \geq 1 - H_q(\delta) - o(1)$.*

1.4.4 Známé třídy kódů

Než se pustíme do vymýšlení vlastního kódu, je dobré se nejdříve podívat, co vymysleli jiní.

Opakovací kódy

Tato třída kódů má bezkonkurenčně největší kódovou vzdálenost. Jsou také ze všech samoopravných kódů nejjednodušší (každý symbol zprávy n-krát zopakujeme) a mají parametry $[n,1,n]$.

Hammingovy kódy

Jedná se o třídu lineárních binárních kódů s minimální vzdáleností 3. Pro každé $r \geq 2$ máme kód s parametry $[2^r - 1, 2^r - r - 1, 3]$. Příkladem je známý (7,4,3)-kód. Hammingovy kódy jsou perfektní.

Reedovy-Solomonovy kódy

Jde o lineární MDS kódy s významným praktickým využitím (např. CD, DVD a Blu-ray disky). Jsou zvláště vhodné pro aplikace, kde se chyby přenosu nevyskytují zcela náhodně, nýbrž v dávkách (tj. typicky se špatně přenese více po sobě jdoucích bitů). Tyto kódy se používají nad abecedou o velikosti $q > 2$ (tedy nejsou binární). RS kód dimenze k nad abecedou velikosti q má parametry $[q-1, k, q-k]_q$ (vidíme, že zde je délka kódu určena velikostí abecedy).

2. Expanderové grafy

Jak jsme již předestřeli v úvodu, budeme se v této práci zabývat grafy s pěknými vlastnostmi, kterým se říká „expanderové“. V této kapitole si o nich něco povíme a naučíme se je konstruovat.

2.1 O expanderech obecně

2.1.1 Hranová expanze

Nejprve nadefinujme expander pomocí hranové expanze. Ta bude zjednodušeně říkat to, že v expanderu z každé dost malé množiny vrcholů vede hodně hran.

Definice 9 (Hranová expanze)**.** *O grafu $G = (N,E)$ řekneme, že je to (n,d,γ,α)-expander, pokud má n vrcholů, je d-regulární a pro každou množinu vrcholů $S \subseteq N$, $|S| \leq \gamma n$ platí, že z S vede alespoň $d\alpha|S|$ hran.*

Samozřejmě chceme, aby jeho parametry byly co nejlepší, tedy chceme co největší α a zároveň co největší γ. Často se v definicích explicitně uvádí $\gamma = 1/2$.

Později budeme některé věci ukazovat na bipartitních expanderech, tedy nadefinujeme hranovou expanzi i pro bipartitní grafy.

Definice 10. *Mějme bipartitní graf $G = (N,M,E)$, kde každý vrchol v N má stupeň d, $|N| = n$, $|M| = m$. Graf G je (n,m,d,γ,α)-bipartitní expander, jestliže pro každou množinu vrcholů $S \subseteq N$, $|S| \leq \gamma n$ platí, že S má v M alespoň $d\alpha|S|$ sousedů.*

Všimněme si, že naše definice bipartitního expanderu je asymetrická, protože neklade téměř žádné požadavky na partitu M. Také poznamenejme, že občas budeme chtít $n \leq m$, tedy graf bude muset expandovat z větší strany do menší, což je silnější požadavek, než kdyby expanze probíhala opačným směrem.

Zatím ale nevíme, zda vůbec nějaké expandery existují, a nevíme ani, jak je sestrojit. V následující sekci nadefinujeme cikcakový součin, který nám tuto konstrukci umožní, a ukážeme některé jeho zajímavé vlastnosti. Zatím si ovšem řekneme něco o vlastních číslech normalizované matice sousednosti a odvodíme ekvivalentní definici expanderu.

2.1.2 Expanze a vlastní čísla

Expanzní vlastnosti grafu lze také elegantně popsat pomocí vlastních čísel jeho normalizované matice sousednosti, konkrétně druhého největšího z nich.

Definice 11. *$\lambda(G)$ označíme v absolutní hodnotě druhé největší vlastní číslo matice M.*

Následující lemma ukazuje, že druhé vlastní číslo se dá také definovat pomocí standardního skalárního součinu (podle Reingold a kol. (2002)).

Lemma 10.

$$\lambda(G) = \max_{\alpha \perp 1_n} \frac{|\langle \alpha, M\alpha \rangle|}{\langle \alpha, \alpha \rangle} = \max_{\alpha \perp 1_n} \frac{\|M\alpha\|}{\|\alpha\|},$$

kde n je počet vrcholů G.

Všimneme si také, že $\lambda(G)$ popisuje rychlost konvergence náhodné procházky po grafu. Uvažujme $\pi \in [0,1]^n$ jako pravděpodobnostní rozdělení na vrcholech G. Když π vnímáme jako vektor, můžeme jej rozložit jako $\pi = \mathbf{u}_n + \pi^\perp$, kde $\mathbf{u}_n$ je rovnoměrné rozdělení na n vrcholech a $\pi^\perp \perp \mathbf{u}_n$. Potom $M\pi = \mathbf{u}_n + M\pi^\perp$ je pravděpodobnostní rozdělení po jednom kroku náhodné procházky. Z tvrzení výše dostáváme

$$||M\pi^\perp|| \leq \lambda(G) \cdot ||\pi^\perp||.$$

Tedy $M\pi \leq \mathbf{u}_n + \lambda(G)\pi^\perp$, po dalším kroku náhodné procházky dostáváme $M^2\pi \leq \leq \mathbf{u}_n + (\lambda(G))^2\pi^\perp$, až po k krocích máme $M^k\pi \leq \mathbf{u}_n + (\lambda(G))^k\pi^\perp$. Tedy $\lambda(G)$ je omezením, jak rychle náhodná procházka na G konverguje k rovnoměrnému rozdělení. Vidíme, že čím menší $\lambda(G)$ je, tím se pravděpodobnostní rozdělení na vrcholech grafu přibližuje rovnoměrnému rozdělení rychleji, a tedy tím lepší jsou expanzní vlastnosti G.

Konzistentně s tímto pozorováním nazveme třídu grafů $\mathcal{G}$ *třídou expanderů*, jsou-li vlastní čísla jejich normalizovaných matic sousednosti odražena od nuly, tedy existuje-li $\lambda < 1$ t.ž. $\lambda(G) \leq \lambda$ pro všechny grafy $G \in \mathcal{G}$. Všimněme si, že podle tohoto značení bipartitní grafy nikdy nejsou expandery. Definice bipartitního expanderu je naprosto oddělená vlastnost, která (jakkoliv je užitečná) nemá s popisem expanze grafu pomocí $\lambda(G)$ nic společného.

Zaveďme značení:

Definice 12. *(n,d,λ)-graf buď d-regulární graf na n vrcholech, kde v absolutní hodnotě druhé největší vlastní číslo jeho normalizované matice sousednosti je nejvýš λ.*

Ukážeme, jak souvisí velikost λ a konvergence náhodné procházky s hranovou expanzí. Intuitivně je vztah hranové expanze a náhodných procházek jasný. Pokud z každé množiny vrcholů vede hodně hran, máme při náhodné procházce více možností, kam se dostat, a tedy se k rovnoměrnému rozdělení blížíme rychleji. Jestliže naopak náhodná procházka rychle konverguje k rovnoměrnému rozdělení, nejspíš nikde nemohlo vzniknout úzké hrdlo, a tedy z každé dost malé množiny vrcholů vede hodně hran. Formální důkaz je ale od této úvahy dost vzdálený.

Věta 11. *Předpokládejme, že G je (n,d,λ)-graf. Mějme množinu $S \subseteq [n]$, kde $|S| \leq \gamma n$, a označme* out(S) *počet hran jdoucích ven z množiny S. Pak platí*

$$\text{out}(S) \geq (1-\lambda)(1-\gamma)d|S|,$$

tedy G je (n,d,γ,α)-expander pro

$$\alpha \geq (1-\lambda)(1-\gamma).$$

Důkaz. Nadefinujme $\mathbf{1}_S \in \mathbb{Z}_2^n$ jako charakteristický vektor množiny S, tedy má jedničky na pozicích, které odpovídají prvkům množiny S, jinde nuly. Symbolem označme $\mathbf{1}$ vektor samých jedniček. Všimneme si, že když matici sousednosti A

grafu G vynásobíme vektorem $\mathbf{1}_S$, vyjde vektor, ve kterém na pozici i je číslo udávající, kolik hran vede z množiny S do vrcholu i. Když jej skalárně vynásobíme s $\mathbf{1}$ (tedy sečteme jeho prvky), vyjde $d|S|$, tedy celkový počet hran vedoucích z prvků množiny S. Podobně když ten samý vektor skalárně vynásobíme s $\mathbf{1}_S$, vyjde počet hran, které vedou mezi prvky množiny S. A když tato dvě čísla odečteme, dostaneme počet hran vedoucích z množiny S ven, který označíme $\mathrm{out}(S)$. Vyjádřeno vzorcem:

$$\mathrm{out}(S) = \langle A\mathbf{1}_S, \mathbf{1} - \mathbf{1}_S \rangle = d|S| - \langle A\mathbf{1}_S, \mathbf{1}_S \rangle \,.$$

Platí, že pro symetrické reálné matice vždy existuje ortonormální báze z vlastních vektorů. Buď v_i normovaný vlastní vektor příslušný i-tému největšímu vlastnímu číslu. Vyjádříme tedy $\mathbf{1}_S$ v bázi z normovaných vlastních vektorů matice A, tedy $\mathbf{1}_S = \sum a_i \mathbf{v}_i$.

Největším vlastním číslem A je n (u normalizované matice sousednosti by to byla jednička) a jemu zřejmě přísluší vektor $\mathbf{1}/\sqrt{n}$. Pak zřejmě platí

$$\langle \mathbf{1}_S, v_1 \rangle = \langle \sum_{i \in [n]} a_i v_i, v_1 \rangle = \langle a_1 v_1, v_1 \rangle = a_1 \,.$$

Zároveň (protože $v_1 = \mathbf{1}/\sqrt{n}$) je tento výraz roven $|S|/\sqrt{n}$. Také platí $\sum_{i \in [n]} a_i^2 = \mathbf{1}_S^2 = |S|$. Zaveďme značení $\lambda_i' = d\lambda_i$. Můžeme počítat:

$$\begin{aligned} \langle A\mathbf{1}_S, \mathbf{1}_S \rangle &= \left\langle \sum_{i \in [n]} a_i \lambda_i' v_i, \sum_{i \in [n]} a_i v_i \right\rangle \\ &= \sum_{i \in [n]} a_i^2 \lambda_i' \\ &= a_1^2 d + \sum_{i=2}^{n} a_i^2 \lambda_i' \\ &\leq a_1^2 d + \lambda_2' (|S| - a_1^2) \\ &= a_1^2 (d - \lambda d) + \lambda d |S| \\ &= \frac{|S|^2}{n} d(1-\lambda) + \lambda d |S| \,. \end{aligned} \tag{2.1}$$

Tudíž když dosadíme do vzorce pro odchozí hrany a změníme rovnost na nerovnost, máme

$$\begin{aligned} \mathrm{out}(S) &\geq d|S| - \left(\frac{|S|^2}{n} d(1-\lambda) + \lambda d |S| \right) \\ &\geq d|S| - (\gamma |S| d(1-\lambda) + \lambda d |S|) = d|S|(1-\lambda)(1-\gamma) \,. \end{aligned} \tag{2.2}$$

□

2.1.3 Bipartitní expandery

Zatím jsme o expanderech uvažovali jen v termínech hranové expanze nebo vlastních čísel, takže dost obecně. Pro některé aplikace se ale hodí mít bipartitní expander, zatím ale nevíme, zda jej lze sestrojit.

Ukážeme si jednoduchou konstrukci, která převede obecný expander na bipartitní graf bez výrazného zhoršení jeho expanzních vlastností. Ještě jsme si ani neukázali konstrukci obecného expanderu, ale tu si ukážeme níže.

Mějme obecný d-regulární expander s m vrcholy a n hranami (trochu nezvykle značeno). Nyní uvažujme, že podrozdělíme všechny hrany tohoto grafu vrcholem a tyto dělicí vrcholy vezmeme jako jednu partitu, vrcholy původního grafu budou v druhé partitě. Je vidět, že vzniklý graf je bipartitní (mezi vrcholy jedné partity nevedou žádné hrany), že každá z partit je regulární (první je 2-regulární, druhá d-regulární) a že expanzní vlastnosti druhé partity se nezhorší (tedy z každé dost malé množiny vrcholů z druhé partity stále povede hodně hran). Této konstrukci říkejme *incidenční*.

Také budeme využívat jinou konstrukci, která má tu výhodu, že obě partity jsou stejně velké a d-regulární. Budeme jí říkat *dvojpokrytí* a definujeme ji takto: vezmeme dvě kopie množiny vrcholů a natáhneme hrany mezi partitami tak, jak vedly v původním grafu, tedy pokud v původním grafu vedla hrana mezi vrcholy i a j, natáhneme hranu mezi i v první partitě a j v druhé partitě a obráceně.

Všimneme si, že obě uvedené konstrukce zdvojnásobí počet hran, na což si při převodech mezi verzemi grafů musíme dávat pozor.

2.2 Grafové součiny

Známe nějaké ad hoc konstrukce expanderů, ale ty nám nedávají dostatečnou volnost. Potřebujeme konstruovat grafy s různým počtem vrcholů, což se nejlépe dělá pomocí grafových součinů. My zde předvedeme ty, které alespoň částečně zachovávají expanzi grafu, a proto se nám hodí.

2.2.1 Mocniny grafů

Zde nadefinujeme operaci umocňování grafů.

Definice 13. *Buď G d-regulární multigraf na $[n]$ s rotační mapou Rot_G. Jeho t-tá mocnina G je d^t-regulární multigraf G^t, jehož rotační mapa je dána vztahem $\mathrm{Rot}_{G^t}(v_0,(k_1,k_2,\ldots,k_t)) = (v_t,(l_t,l_{t-1},\ldots,l_t))$, kde příslušné hodnoty získáme ze vztahu $(v_i,l_i) = \mathrm{Rot}_G(v_{i-1},k_i)$.*

Snadno z definice nahlédneme, že v grafu G^t vedou hrany právě mezi těmi vrcholy, mezi nimiž vedl v grafu G sled délky t. Ukážeme, že $A_{G^t} = A_G^t$, tedy že matice sousednosti nového grafu je rovna t-té mocnině matice sousednosti původního grafu.

Nejprve to ukážeme pro druhou mocninu. Označme prvek matice A na pozici ij jako a_{ij}. V A_G^2 se prvek na pozici ij vypočítá jako suma $\sum_{k=1}^{n} a_{ik}a_{kj}$. Součin $a_{ik}a_{kj}$ je nenulový právě tehdy, když z vrcholu i vede sled délky 2 do vrcholu j procházející vrcholem k. Přesněji řečeno, tento součin je počet všech takových sledů. Tedy prvky matice A_G^2 jsou počty všech sledů délky 2 mezi danými dvěma vrcholy, což přesně odpovídá matici A_{G^2}. A z toho, že $A_G^t = A_G^{t-1}A_G$, uvedený vztah plyne i pro obecné t.

Protože náš zkoumaný graf G je d regulární, vede z každého vrcholu d^t sledů délky t, a tedy součet každého řádku a každého sloupce matice A_{G^t} je d^t. Jestliže chceme, aby součet každého řádku byl 1, tedy jestliže chceme matici normalizovat,

musíme ji vydělit prvkem d^t. Z toho okamžitě plyne, že pro normalizované matice sousednosti také platí vztah $M_{G^t} = M_G^t$.

Následující tvrzení je shrnutím pozorovaných vlastností:

Tvrzení 12. *Je-li G (n,d,λ)-graf, potom G^t je (n,d^t,λ^t)-graf. Navíc každý prvek mapy Rot_{G^t} lze vypočítat v čase $\mathrm{poly}(\log n, \log d, t)$ s t dotazy na Rot_G.*

Důkaz. Jak jsme ukázali, normalizovaná matice sousednosti grafu G^t je t-tou mocninou normalizované matice sousednosti grafu G, takže je-li λ vlastní číslo matice M_G, je λ^t vlastní číslo matice M_{G^t}. To, že G^t je d^t-regulární, vyplynulo jako důsledek vztahu mezi maticemi sousednosti. Časový odhad je také zřejmý, jeden dotaz na orákulum zvládneme v čase $\mathrm{poly}(\log n, \log d)$ (kvůli velikostem čísel uložených v rotační mapě) a položíme jich t (ze vztahu pro rotační mapy uvedeného v definici). □

Dále nadefinujeme ještě tenzorový součin grafů.

2.2.2 Tenzorový součin

Z lineární algebry víme, že tenzorový součin vektorů $\mathbf{u} \in \mathbb{R}^{n_1}$ a $\mathbf{v} \in \mathbb{R}^{n_2}$ je matice $n_1 \times n_2$ taková, že na pozici ij je prvek $u_i v_j$. Tuto matici můžeme „splácnout“ do vektoru nad $\mathbb{R}^{n_1 n_2}$ tak, že si políčka této matice nějak bijektivně označíme čísly od 1 do $n_1 n_2$. Analogicky zavedeme tenzorový součin matic: buď A matice $n_1 \times n_1$ a B matice $n_2 \times n_2$, jejich tenzorový součin si můžeme představit jako čtyřrozměrnou matici $n_1 \times n_2 \times n_1 \times n_2$, kde na pozici $ijkl$ je součin $a_{ik} b_{jl}$. Tuto mřížku „splácneme“ do matice $n_1 n_2 \times n_1 n_2$ tak, aby pro matice $A \in \mathbb{R}^{n_1 \times n_1}$, $B \in \mathbb{R}^{n_2 \times n_2}$ a vektory $\mathbf{u} \in \mathbb{R}^{n_1}$, $\mathbf{v} \in \mathbb{R}^{n_2}$ platilo

$$(A \otimes B)(\mathbf{u} \otimes \mathbf{v}) = (A\mathbf{u}) \otimes (B\mathbf{v}) .$$

Zde uvažujeme tenzorové součiny ve „splácnutém“ tvaru.

Analogicky zadefinujeme tenzorový součin pro grafy. Buď G_1 d_1-regulární multigraf na $[n_1]$ a G_2 d_2-regulární multigraf na $[n_2]$. Definujme tenzorový součin grafů $G_1 \otimes G_2$ jako $d_1 d_2$-regulární multigraf na $[n_1] \times [n_2]$, který je dán svou rotační mapou, kde platí $\mathrm{Rot}_{G_1 \otimes G_2}((v,w),(i,j)) = ((v',w'),(i',j'))$, kde $(v',i') = \mathrm{Rot}_{G_1}(v,i)$ a $(w',j') = \mathrm{Rot}_{G_2}(w,j)$.

Zase se vyplatí podívat se na význam tohoto součinu na maticích sousednosti grafů. V novém grafu $G_1 \otimes G_2$ jsou vrcholy dány dvojicemi vrcholů starých grafů G_1 a G_2 a hrana mezi vrcholy (a_1,b_1), (a_2,b_2) vede právě tehdy, když vedla hrana mezi vrcholy a_1, a_2 v grafu G_1 a b_1, b_2 v grafu G_2. Jestliže mezi vrcholy a_1, a_2 vedlo a hran a mezi vrcholy b_1, b_2 vedlo b hran, vede mezi vrcholy (a_1,b_1) a (a_2,b_2) ab hran. Teď si všimneme, že byl-li graf G_1 d_1-regulární a graf G_2 d_2-regulární, je nový graf $d_1 d_2$-regulární, neboť z každého vrcholu vede $d_1 d_2$ hran (d_1 za první složku a d_2 za druhou).

Také si všimneme, že všechna vlastní čísla se vynásobí. Mějme $\mathbf{a}$ vlastní vektor matice A_{G_1} odpovídající vlastnímu číslu λ_a a vlastní vektor $\mathbf{b}$ matice A_{G_2} odpovídající vlastnímu číslu λ_b. Potom vektor $\mathbf{a} \otimes \mathbf{b}$ je zřejmě vlastním vektorem matice $A_{G_1 \otimes G_2}$ (když si jednu ze složek zafixujeme, souhlasí, že pro druhou složku je to pro každou hodnotu první složky vlastní vektor) a odpovídá vlastnímu číslu $\lambda_a \lambda_b$ (první složka jej vynásobí λ_a, druhá přidá λ_b). Tedy když se podíváme

na druhé největší vlastní číslo normalizované matice sousednosti grafu $G_1 \otimes G_2$, vidíme, že odpovídá maximu z druhých největších vlastních čísel grafů G_1 a G_2 (největším vlastním číslem je 1 a všechna ostatní vlastní čísla jsou v absolutní hodnotě menší nebo rovna 1).

I toto tvrzení je shrnutím pozorovaných vlastností:

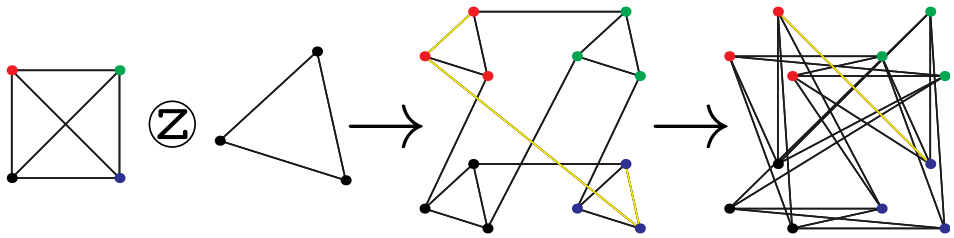
Tvrzení 13. *Je-li G_1 (n_1,d_1,λ_1)-graf a G_2 (n_2,d_2,λ_2)-graf, potom součin $G_1 \otimes G_2$ je $(n_1n_2,d_1d_2,\max(\lambda_1,\lambda_2))$-graf. Navíc každý prvek mapy $\mathrm{Rot}_{G_1\otimes G_2}$ lze vypočítat v čase $\mathrm{poly}(\log n_1n_2, \log d_1d_2)$ s jedním dotazem na Rot_{G_1} i Rot_{G_2}.*

Důkaz. To, že $G_1 \otimes G_2$ je $(n_1n_2,d_1d_2,\max(\lambda_1,\lambda_2))$-graf, jsme už ukázali. Časový odhad je korektní z podobných důvodů jako v předchozím důkazu, protože logaritmus součinu je součet logaritmů. □

2.2.3 Cikcakový součin

Neformálně řečeno, cikcakový součin má za argumenty regulární grafy G (velký) a H (malý) a vrací graf, který je zhruba velký jako G, ale má stupeň jako H. Součin funguje zhruba tak, že nahradí vrcholy grafu G kopiemi H (říkejme jim třeba mráčky) a spojí hranou ty vrcholy, mezi kterými vede cesta ve tvaru: Nejprve provedeme jeden pohyb v mráčku (cik), pak přeskočíme mezi mráčky (cak) a pak zase provedeme pohyb ve výsledném mráčku (cik). Jedna z důležitých vlastností cikcakového součinu je ta, že pokud je H dobrý expander, pak je expanze výsledného grafu horší jen o trochu.

Zde ilustrujeme, jak se vypočítá cikcakový součin grafů K_4 a K_3.



Přistupme k formální definici.

Definice 14 (cikcakový součin)**.** *Nechť G je d-regulární graf na množině vrcholů $[n]$ a H je δ-regulární graf na množině vrcholů $[d]$. Cikcakový součin grafů $G \circledz H$ je δ^2-regulární graf na množině vrcholů $[n] \times [d]$. Pro rotační mapu $G \circledz H$ platí $\mathrm{Rot}_{G\circledz H}((v,a),(i,j)) = ((w,b),(j',i'))$, kde jednotlivé proměnné vypočítáme následovně:*

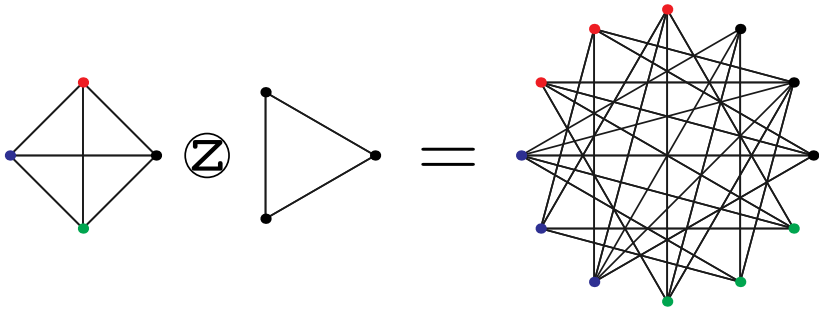
- $(a',i') = \mathrm{Rot}_H(a,i)$
- $(w,b') = \mathrm{Rot}_G(v,a')$
- $(b,j') = \mathrm{Rot}_H(b',j)$

Nyní můžeme zpřesnit neformální vysvětlení cikcakového součinu – každému vrcholu z G očíslujeme odchozí hrany, grafu H očíslujeme vrcholy a při nahrazování vrcholů z G pověsíme i-tou odchozí hranu na i-tý vrchol H.

To vysvětluje, proč je vzniklý graf zrovna δ^2-regulární: Vybereme si počáteční vrchol. Máme δ možností, kam se z něj pohnout v příslušném mráčku. Ať se

dostaneme do jakéhokoliv vrcholu, máme jedinou možnost na pohyb mezi mráčky (z každého vrcholu vede ven z mráčku jediná hrana). A pak zase máme δ možností, jak se pohnout v mráčku, do kterého jsme se dostali, tedy celkem můžeme skončit v δ^2 vrcholech, které se mohou opakovat, ale to nevadí.

Zde je názorněji překresleno, jak vypadá cikcakový součin grafů K_4 a K_3.



Nyní opět shrneme pozorované vlastnosti a uvedeme mez pro λ.

Věta 14. *Je-li G_1 (n,d_1,λ_1)-graf a G_2 (d_1,d_2,λ_2)-graf, potom součin $G_1 \circledS G_2$ je $(n_1d_1,d_2^2,f(\lambda_1,\lambda_2))$-graf, kde $f(\lambda_1,\lambda_2) \leq \lambda_1 + \lambda_2 + \lambda_2^2$ a $f(\lambda_1,\lambda_2) < 1$ když $\lambda_1,\lambda_2 <$ < 1. Navíc každý prvek $\mathrm{Rot}_{G_1 \circledS G_2}$ lze vypočítat v čase $\mathrm{poly}(\log n, \log d_1, \log d_2)$ s jedním dotazem na Rot_{G_1} a dvěma dotazy na Rot_{G_2}.*

Tuto větu dokázali Reingold a kol. (2002), my ji zde dokazovat nebudeme. V tomtéž článku byl odhad posléze zpřesněn:

Věta 15 (Vylepšený odhad)**.** *Je-li G_1 (n,d_1,λ_1)-graf a G_2 (d_1,d_2,λ_2)-graf, potom součin $G_1 \circledS G_2$ je $(n_1d_1,d_2^2,f(\lambda_1,\lambda_2))$-graf, kde*

$$f(\lambda_1,\lambda_2) = \frac{(1-\lambda_2^2)\lambda_1 + \sqrt{(1-\lambda_2^2)^2\lambda_1^2 + 4\lambda_2^2}}{2}.$$

Ačkoliv tato funkce vypadá dost ošklivě, dá se přímočaře ukázat, že splňuje následující pěkné vlastnosti:

1. $f(\lambda,0) = f(0,\lambda) = \lambda$ a $f(\lambda,1) = f(1,\lambda) = 1$ pro všechna $\lambda \in [0,1]$.
2. $f(\lambda_1,\lambda_2)$ je ostře rostoucí funkce v obou proměnných (s výjimkou případu $\lambda_i = 1$).
3. Pokud $\lambda_1 < 1$ a $\lambda_2 < 1$, potom $f(\lambda_1,\lambda_2) < 1$.
4. $f(\lambda_1,\lambda_2) \leq \lambda_1 + \lambda_2$ pro všechna $\lambda_1,\lambda_2 \in [0,1]$.

Tuto větu zde také dokazovat nebudeme.

2.3 Konstrukce expanderů

Nyní ukážeme, jak pomocí cikcakového součinu můžeme sestrojit expandery konstantního stupně. Nejprve ukážeme zjednodušenou variantu, kterou posléze vylepšíme.

2.3.1 Jednoduchá konstrukce

Buď H jakýkoliv $\left(d^4,d,\frac{1}{5}\right)$-graf (to, že existuje, ještě ukážeme). Sestrojme posloupnost grafů:

- $G_1 = H^2$
- $G_{i+1} = G_i^2 \circledcirc H$ pro všechna $i \in \mathbb{N}$

Věta 16. *Pro každé i je G_i $(d^{4i},d^2,2/5)$-graf.*

Důkaz. Rozmysleme si, co dělají jednotlivé operace. Umocnění na druhou převede (n,d,λ)-graf na (n,d^2,λ^2)-graf, cikcakový součin vyrobí z (n,d,λ_1)-grafu a (d,δ,λ_2)-grafu $(n_1n_2,\delta^2,\lambda_1+\lambda_2+\lambda_2^2)$-graf. Tudíž G_1 je $(d^4,d^2,1/25)$-graf.

Nyní budeme postupovat matematickou indukcí. Mám G_i, které je (d^{4i},d^2,λ_i)-grafem, kde $\lambda_i < 2/5$. Chceme sestrojit G_{i+1}. Platí $G_{i+1} = G_i^2 \circledcirc H$, takže G_{i+1} je $(d^{4(i+1)},d^2,\lambda_i^2+1/5+1/25)$-graf. Protože $\lambda_i < 2/5$, je $\lambda_{i+1} < 4/25+1/5+1/25 = 2/5$, tedy omezení velikosti λ_{i+1} stále platí. □

Ukázali jsme, že tato konstrukce vyrobí nekonečnou rodinu expanderů. Bohužel není tak rychlá, jak bychom si přáli. Umocňování na druhou konstrukci zpomaluje, takže běží v čase polynomiálním vzhledem k velikosti grafu. Ukážeme, jak se to dá obejít s využitím tenzorového součinu.

2.3.2 Zrychlená konstrukce

Buď H (d^8,d,λ)-graf pro nějaké d a λ. Položme $G_1 = H^2$ a $G_2 = H \otimes H$. Pro každé $i > 2$ definujeme

$$G_i = \left(G_{\lceil \frac{i-1}{2} \rceil} \otimes G_{\lfloor \frac{i-1}{2} \rfloor}\right)^2 \circledcirc H\,.$$

Vyslovíme a dokážeme větu:

Věta 17. *Pro každé $i \geq 1$ je G_i (d^{8i},d^2,λ_i)-graf, kde $\lambda_i = \lambda + \mathcal{O}(\lambda^2)$. Každý prvek Rot_{G_i} lze vypočítat v čase $\mathrm{poly}(i, \log d)$ s $\mathrm{poly}(i)$ dotazy na Rot_H.*

Důkaz. Přímočarou indukcí jako v předchozím důkazu nahlédneme, že počet vrcholů v G_i je d^{8i} a že stupeň je d^2. Položme $\mu_i = \max(\lambda_1,\ldots,\lambda_i)$ a všimněme si, že tenzorový součin nastavuje jako druhé vlastní číslo nového grafu maximum z druhých vlastních čísel činitelů. Pro maxima to určitě také platí, tedy $\mu_i \leq \leq \max(\mu_{i-1},\mu_{i-1}^2+\lambda+\lambda^2)$. Nahlédneme, že vztah pro λ_i je také správně (λ v první mocnině se v žádném λ_i nevyskytne více než jednou, zároveň tam ale bude alespoň jednou).

Co se časové složitosti týče, nahlédneme, že hloubka rekurze je nanejvýš $\log_2 i$ a vypočtení rotační mapy pro G_i zahrnuje nejvýše čtyři dotazy na rotační mapy menších grafů, tedy celková časová složitost rekurze bude nejvýše $4^{\log_2 i} = i^2$ a tenzorový součin běží v polylogaritmickém čase. □

Rádi bychom ale získali odhad, který je trochu konkrétnější. Všimneme si, že pokud základní graf měl $\lambda = 1/5$, bude každé $\lambda_i < 2/5$, a to úplně ze stejných důvodů jako v předchozím důkazu. Ale co obecné λ? Abychom odhadli všechna λ_i,

chceme najít δ takové, že pro všechna i platí $\delta \leq \lambda_i$. Proto musí platit $\delta \geq$ $\geq \delta^2 + \lambda + \lambda^2$. Tedy chceme vyřešit nerovnost

$$\delta^2 - \delta + \lambda + \lambda^2 \leq 0\,.$$

Nejprve vyřešme příslušnou kvadratickou rovnici v proměnné δ. Její diskriminant je roven $1 - 4(\lambda + \lambda^2)$, takže vidíme, že tato rovnice má reálné řešení jen pro $\lambda^2 + \lambda \leq 1/4$, tedy pro $\lambda \leq (\sqrt{2} - 1)/2 \approx 0.2071$ (zde vidíme, že $\lambda = 1/5$ je docela blízko horní mezi pro to, aby vůbec z věty 14 plynul nějaký netriviální horní odhad). Tedy omezme λ a pokračujme dále. Nerovnost pro δ platí na intervalu $[(1 - \sqrt{1 - 4(\lambda + \lambda^2)})/2,(1 + \sqrt{1 - 4(\lambda + \lambda^2)})/2]$. Odhad samozřejmě chceme zvolit co nejmenší, tedy mějme $\delta = (1 - \sqrt{1 - 4(\lambda + \lambda^2)})/2$. Pro největší možné $\lambda = (\sqrt{2} - 1)/2$ je tedy $\delta = 1$, což souhlasí, ale moc nám to nepomůže.

Nahoře ukazujeme, že λ_i jsou omezena výrazem $\lambda + \mathcal{O}(\lambda^2)$. Rádi bychom $\mathcal{O}()$ v tomto výrazu odhadli nějakou konkrétní multiplikativní konstantou. Z odhadu pro $\lambda = 1/5$ si tipneme, že to může být třeba 5 (nižší číslo vzít nemůžeme). Zkusíme tento tip ověřit.

Chceme tedy dokázat

$$\frac{1 - \sqrt{1 - 4(\lambda + \lambda^2))}}{2} \leq \lambda + 5\lambda^2\,.$$

Z toho dostaneme vztah $1 - 2\lambda - 10\lambda^2 \leq \sqrt{1 - 4(\lambda + \lambda^2)}$, obě strany nerovnice jsou větší než nula, tedy umocníme na druhou, upravíme a dostaneme vztah $100\lambda^4 + 40\lambda^3 - 12\lambda^2 \leq 0$. Vidíme, že tento výraz má dvojnásobný kořen 0, který na platnost nerovnice nemá vliv, tedy nerovnici vydělíme $4\lambda^2$ a dostaneme $25\lambda^2 + 10\lambda - 3 \leq 0$. Příslušná kvadratická rovnice má kořeny $-3/5$ a $1/5$, tedy tento vztah platí jen pro $\lambda \leq 1/5$.

Dokázali jsme větu:

Věta 18. *Pro každé $i \geq 1$ je G_i (d^{8i},d^2,λ_i)-graf, kde $\lambda_t \leq \lambda + 5\lambda^2$ pro $\lambda \leq 1/5$.*

Tahle mez na vlastní čísla ale tlačí jak stupeň, tak počet vrcholů grafu do dosti vysokých hodnot. Stupeň ale umíme snížit – když cikcakově vynásobíme graf lichého stupně s cyklem (liché délky), dostaneme graf stupně 4. Nový graf bude stále expander, jak plyne z pokročilejší analýzy cikcakového součinu ve větě 15, protože cyklus liché délky má druhé největší vlastní číslo v absolutní hodnotě menší než 1 (protože je souvislý a není bipartitní). Expanzní vlastnosti ale asi nebudou nijak závratné, více o tom říkají Reingold a kol. (2002) v Důsledku 3.4. Nicméně, takto umíme z každé rodiny expanderů lichého stupně vyrobit rodinu expanderů stupně 4.

2.3.3 Konstrukce základního grafu

Už umíme zkonstruovat nekonečné rodiny expanderů. Všechny tyto konstrukce vyžadovaly, aby základní graf byl dobrý expander. Jsou známy efektivní konstrukce takovýchto expanderů, my si zde ukážeme některé z nich jako příklad. Existují i konstrukce, které produkují grafy s lepšími parametry, ale ty nejsou tak názorné.

V této konstrukci sestrojíme expander, kde $n = d^2$, a ten pak s využitím cikcakového součinu a tenzorového součinu upravíme, aby $n = d^8$. Důkazy zde budou jen nastíněny, protože jsou hodně přímočaré.

Uvažujme těleso $\mathbb{F}_q$, kde q je mocnina prvočísla, a nad ním afinní rovinu. Vrcholy našeho grafu budou body v této rovině. Z vrcholu (a,b) pro libovolná $a,b \in \mathbb{F}_q$ povedou hrany do všech vrcholů, které leží na přímce $L_{ab} : y = ax - b$, tedy z něj povede celkem q hran. Tento graf označme H. Platí následující:

Tvrzení 19. *H je $(q^2,q,1/\sqrt{q})$-graf, každý prvek jeho rotační mapy lze vypočítat v čase* $\mathrm{poly}(\log q)$.

Důkaz. Počet vrcholů a stupeň grafu jsou zřejmé z definice, prozkoumejme tedy mez pro druhé největší vlastní číslo. Pro to stačí ukázat, že druhé největší vlastní číslo grafu G^2 je $1/q$. Buď M normalizovaná matice sousednosti grafu H, vypočítejme M^2. Prvek M^2 v řádku (a,b) a sloupci (a',b') je přímo počet společných sousedů těchto dvou bodů vydělený q^2. Jestliže $a \neq a'$, pak se přímky L_{ab} a $L_{a'b'}$ protínají právě v jednom bodě, protože nemají stejnou směrnici, tedy prvek M na pozici $((a,b),(a'b'))$ je $1/q^2$. Pokud se ale směrnice rovnají, pak jsou přímky buď rovnoběžné (když $b \neq b'$), tedy se neprotnou, nebo jsou shodné, a to platí $a' = a$ a $b' = b$, vznikne tedy q smyček.

Nechť je matice M indexována dvojicemi (a,b), které jsou seřazeny lexikograficky – nejprve podle a a pak úseky se stejným a seřadíme podle b. Buď I_q jednotková matice typu $q \times q$ a J_q matice typu $q \times q$ složená ze samých jedniček. Potom

$$M^2 = \frac{1}{q^2}\begin{pmatrix} qI_q & J_q & \cdots & J_q \\ J_q & qI_q & \cdots & J_q \\ \vdots & \vdots & \ddots & J_q \\ J_q & J_q & \cdots qI_q & \end{pmatrix} = \frac{I_q \otimes qI_q + (J_q - I_q) \otimes J_q}{q^2}\,.$$

Nyní můžeme explicitně vypočítat vlastní čísla. Matice J_q má vlastní čísla q (násobnosti 1) a 0 (násobnosti $q-1$). Matice $J_q - I_q$ má tedy vlastní čísla $q-1$ a -1. Vlastní čísla obou matic jsme mohli odečíst, protože vlastními vektory matice I_q jsou všechny vektory z $\mathbb{R}^q$. Tenzorový součin způsobí to, že se vlastní čísla vynásobí. Tedy $(J_q - I_q) \otimes J_q$ má vlastní čísla $q(q-1)$, $-q$ a 0, jiná nemohou vzniknout. O něco jednodušší je výpočet vlastních čísel $I_q \otimes qI_q$, která má za vlastní číslo pouze q. Výsledkem tohoto součinu je navíc $qI_{q\times q}$, takže zase jsou všechny vektory vlastní a tím pádem můžeme vlastní čísla prostě sečíst. Tedy když všechna vzniklá vlastní čísla sečteme, dostaneme q^2, 0 a $-q$. Když je vydělíme q^2, dostaneme vlastní čísla matice M^2, tedy vidíme, že druhé největší vlastní číslo grafu G^2 je v absolutní hodnotě $1/q$ a jsme hotovi.

Rotační mapa H je z definice H ve tvaru

$$\mathrm{Rot}_H((a,b),t) = \begin{cases} ((t/a,t-b),t) \text{ pokud } a \neq 0 \text{ a } t \neq 0 \\ ((t,-b),a) \text{ pokud } a = 0 \text{ nebo } t = 0\,. \end{cases}$$ □

Nyní z grafu H sestrojíme posloupnost dalších grafů:

- $H_1 = H \otimes H$
- $H_i = H_{i-1} \circledcirc H$

Tvrzení 20. *H_i je $(q^{2(i+1)},q^2,\mathcal{O}(i/\sqrt{q}))$-graf a každý prvek jeho rotační mapy lze vypočítat v čase* $\mathrm{poly}(i,\log q)$.

Důkaz. Uvedené vlastnosti získáme jako u konstrukcí výše prostým rozepsáním. Nejprve z vlastností tenzorového součinu vidíme, že H_1 je $(q^4,q^2,1/\sqrt{q})$-graf, tedy pro $i = 1$ věta platí. Dále postupujme matematickou indukcí, nechť tedy věta platí pro nějaké $n \geq 1$. Potom H_n je $(q^{2(n+1)},q^2,n/\sqrt{q})$-graf a $H_{n+1} = H_n \circledcirc H$, tedy z vlastností cikcakového součinu je to $(q^{2(n+1)+2},q^2,\mathcal{O}(n/\sqrt{q}) + 1/\sqrt{q} + 1/q)$-graf, tedy věta je dokázána. □

Reingold a kol. (2002) ukazují, že pomocí lepší analýzy cikcakového součinu může být multiplikativní konstanta skrytá v $\mathcal{O}(i/\sqrt{q})$ redukována na 1, tedy se zbavíme $\mathcal{O}$ a máme nad λ větší kontrolu.

Z uvedené posloupnosti už pak snadno získáme základní grafy pro konstrukce výše (pro tu první zvolením $i = 3$, pro tu druhou $i = 7$) pro dostatečně velké q, které je mocnina prvočísla.

Konstrukce pomocí polynomů

Konstrukce uvedená výše má nevýhodu, že nekonstruuje tak dobré grafy, jak je možné. Proto předvedeme ještě jednu konstrukci, která je trochu složitější, zato však tímto nešvarem netrpí.

Následující konstrukci zde ukážeme tak, jak je popsána v článku Reingold a kol. (2002). Obecně se jedná o grafy odvozené z kontrolní matice nějakého lineárního samoopravného kódu, my zde popíšeme pouze speciální případ. Jako samoopravný kód vezmeme Reedův-Solomonův kód spojený s Hadamardovým kódem (jako ve větě 33) a konstrukci popíšeme přímo na jeho kontrolní matici, použitý RS kód dále nebudeme zmiňovat.

Definice 15. *Pro q mocninu prvočísla a $\delta \in \mathbb{N}$ definujme q^2-regulární graf G na množině vrcholů $\mathbb{F}_q^{\delta+1}$. Pro vrchol $a \in \mathbb{F}_q^{\delta+1}$ a $x,y \in \mathbb{F}_q$ je (x,y)-tým sousedem vrcholu a vrchol $a + (y,yx,yx^2,\ldots,yx^\delta)$.*

Všimneme si, že G skoro není multigraf, protože v tělese můžeme dělit (pokud $ab_1 = ab_2$ a $a \neq 0$, pak $b_1 = b_2$). Takže mezi každými dvěma různými vrcholy G vede nejvýše jedna hrana a každý vrchol má na sobě q smyček. Je trochu zvláštní, že zde sousedy číslujeme dvojicemi přirozených čísel. Dá se o tom přemýšlet tak, že si (jako u tenzorového součinu) vymyslíme nějaké bijektivní očíslování dvojic (nezáleží na tom, jaké přesně) a tabulku sousedů si „splácneme“ do vektoru o q^2 složkách. Jenom pro vypočítání souřadnic i-tého souseda musíme umět i zase zpátky zobrazit do $\mathbb{F}_q^2$. Vyslovíme tvrzení:

Tvrzení 21. *G je $(q^{\delta+1},q^2,\delta/q)$-graf. Každý prvek rotační mapy G lze vypočítat v čase* $\mathrm{poly}(\log q,\delta)$.

Důkaz. To, že $n = q^{\delta+1}$ a $d = q^2$, je zřejmé už z definice, zbývá λ a časová složitost.

Buď M normalizovaná matice sousednosti grafu G. Buď p charakteristika $\mathbb{F}_q$, $\theta = \mathrm{e}^{2\pi i/p}$ primitivní p-tá odmocnina z 1 a $L : \mathbb{F}_q \to \mathbb{F}_p$ nějaké surjektivní lineární zobrazení (tedy je na).

Pro každý vektor $a = (a_1, \ldots, a_{d+1}) \in \mathbb{F}_q^{d+1}$ definujme *charakter* $\chi_a : \mathbb{F}_q^{d+1} \to \mathbb{C}$ jako $\chi_a(b) = \theta^{L(\sum a_i b_i)}$. Nahlížejme na χ_a jako na vektor v $\mathbb{C}^{q^{d+1}}$, který má na pozici b hodnotu $\chi_a(b)$. Zřejmě platí $\chi_a(b+c) = \chi_a(b)\chi_a(c)$ pro všechny vektory b,c. Také se dá ukázat, že $\{\chi_a\}$ tvoří ortonormální bázi vzhledem ke standardnímu skalárnímu součinu. Chceme ukázat, že χ_a je pro každé a vlastním vektorem M. Počítejme pro každou složku vektoru χ_a:

$$M(\chi_a)(b) = \frac{1}{q^2} \sum_c M_{bc} \chi_a(c)$$

Na tomto místě si všimneme, že když začneme indexovat přes všechny dvojice (x,y) (jako výše v definici grafu) a vypustíme člen M_{bc}, nic se nepokazí. Všechny ostatní pozice budou nulové a to, že zobrazení mezi (x,y) a označením vrcholů není prosté, nám nevadí, prostě to samé přičteme víckrát. V grafu, který není multigrafem, by to prostě bylo vypuštění pozic, kde je $M_{b,c}$ nulové.

$$M(\chi_a)(b) = \frac{1}{q^2} \sum_{x,y} \chi_a(b + (y, xy \ldots, x^\delta y))$$

Všimneme si, že nyní můžeme vytknout $\chi_a(b)$ a to, co nám zbude, je nezávislé na b (označíme si to λ_a), tedy χ_a je opravdu vlastním vektorem matice M s vlastním číslem λ_a. Nyní je potřeba ukázat horní mez na λ_a. Všimněme si:

$$\lambda_a = \frac{1}{q^2} \sum_{x,y} \chi_a((y, yx, \ldots, yx^\delta)) = \frac{1}{q^2} \sum_{x,y} \theta^{L(y p_a(x))},$$

kde $p_a(x)$ je nějaký polynom v x stupně δ. Protože θ je odmocnina z jedničky, tak má-li nulový exponent, je rovna 1. Když $p_a(x) = 0$, tak celkem $\theta^{L(y p_a(x))}$ přispěje jedničkou nejvýše $q\delta$-krát (pro všechna y). Pokud $p_a(x)$ není rovno 0, tak $y p_a(x)$ pro různá y projde všechny možné hodnoty, a protože součet všech odmocnin z jedné je 0, k celkovému součtu to nijak nepřispěje. Takže $\lambda_a \leq q\delta/q^2 = \delta/q$. □

Když nastavíme $\delta = 15$ (nebo pro pomalejší konstrukci $\delta = 7$) a dostatečně velké q, dostaneme vhodný základní graf. Tyhle grafy jsou sice větší než ty z předchozí konstrukce, mají ale tu výhodu, že dosahují hodnoty $\mathcal{O}(1/\sqrt{d})$ pro druhé největší vlastní číslo ($d = q^2$ a $\lambda = \delta/q$). To implikuje vlastní číslo $\mathcal{O}(1/d^{1/4})$ v konstrukci z věty 17, neboť tam se konstruují grafy se stupněm základního grafu umocněným na druhou.

3. Expanderové kódy

Když už teď umíme zkonstruovat různě velké expandery, můžeme se pustit do konstrukce kódu. Než uvedeme nějakou pokročilejší konstrukci, podíváme se na spojení mezi expandery a kódy úplně od základu.

V celé kapitole se budeme zabývat hlavně bipartitními grafy. To je dost silná vlastnost, kterou můžeme vypozorovat i v jejich matici sousednosti A. Když si přečíslujeme jejich vrcholy tak, aby vrcholy jedné partity tvořily souvislý blok, všimneme si, že v matici A se vyskytují dva čtvercové nulové bloky (mezi vrcholy jedné partity nevedou hrany) a pak dva obdélníky, které se liší jenom transpozicí. Můžeme ji tedy nahradit úspornější maticí.

Definice 16. *Redukovaná matice sousednosti bipartitního grafu $G = (N,M,E)$, kde $|N| = n$ a $|M| = m$, je matice $n \times m$, kde na pozici ij je číslo k, pokud jsou i-tý vrchol z N a j-tý vrchol z M spojeny k hranami ($k \in \mathbb{N}$).*

Je vidět, že z redukované matice sousednosti můžeme snadno sestrojit běžnou matici sousednosti a obráceně. Podobně můžeme nadefinovat normalizovanou redukovanou matici sousednosti. Nyní ale přistupme k další definici:

Definice 17 (kontrolní graf)**.** *Mějme (lineární) $[n,n-m,d]_2$-kód. Sestrojme bipartitní graf F, jehož redukovaná matice sousednosti bude odpovídat kontrolní matici kódu C. Graf F nazveme kontrolním grafem kódu C.*

Vidíme, že konstrukce výše ukazuje spojení mezi grafy a kódy. Partity grafu F pojmenujme M a N tak, že $|M| = m$ a $|N| = n$. Potom vrcholy v N odpovídají bitům kódu C, hrany vedoucí do M určují všechny skupiny bitů, které by se měly sečíst na nulu, a vrcholy v M ukazují opravdové hodnoty těchto kontrolních součtů. Je zřejmé, že všechny vrcholy v M budou nulové, právě když jsme přijali korektní kódové slovo. Je-li tento graf řídký (třeba konstantního stupně), výsledný kód je LDPC. Navíc, je-li tento graf expander, dá se výsledný kód dekódovat v lineárním čase (jak ukážeme níže). Nyní se konečně dostáváme k stěžejní definici celé práce:

Definice 18. *Expanderové kódy jsou lineární kódy, jejichž kontrolní graf je bipartitní expander. Kód náležící (bipartitnímu) expanderu G označíme $C(G)$.*

Nyní přistupme ke konkrétním konstrukcím expanderových kódů.

3.1 Základní konstrukce

V úvodu této kapitoly jsme mluvili o kontrolním grafu kódu a říkali jsme, že se z něj dá jednoznačně sestrojit lineární kód. Nyní ukážeme několik užitečných vlastností.

Lemma 22. *Nechť $F = (N,M,E)$ je $(n,m,d,\gamma,1-\varepsilon)$-bipartitní expander, kde $\varepsilon < 1/2$. Označme $U(S)$ počet vrcholů z M, ze kterých vede do S jen jedna hrana. Pro každé $S \subseteq N$ takové, že $|S| \leq \gamma n$, platí $|U(S)| \geq d(1-2\varepsilon)|S|$.*

Důkaz. Z expanzní vlastnosti S má alespoň $(1-\varepsilon)d|S|$ sousedů v partitě M, ale protože z množiny S vede celkem $d|S|$ hran, do nejvýše $\varepsilon|S|$ ze sousedů $|S|$ vede více než jedna hrana z S. □

Věta 23. *Nechť $F = (N,M,E)$ je $(n,m,d,\gamma,1-\varepsilon)$-bipartitní expander, kde $\varepsilon < 1/2$. Potom kódová vzdálenost odpovídajícího kódu C je alespoň $2\gamma(1-\varepsilon)n$.*

Důkaz. Stačí ukázat, že váha každého nenulového slova kódu C je alespoň $2\gamma(1-\varepsilon)n$.

Pro spor předpokládejme, že jsme našli kódové slovo s menší váhou. Buď $S \subseteq N$ množina vrcholů, které jsou nastaveny na 1. Kontrolní součty v $U(S)$ zřejmě nemohou být splněny (každý z nich vidí jen jeden vrchol, na kterém je jednička). Chceme ukázat, že pokud existuje kódové slovo s váhou menší než $2\gamma(1-\varepsilon)n$, je $U(S)$ neprázdná.

Pokud platí $|S| \leq \gamma n$, pak z lemmatu výše a z $\varepsilon < 1/2$ přímo plyne, že $U(S)$ je neprázdná. Máme-li $|S| > \gamma n$, vybereme z něj podmnožinu Q takovou, že $|Q| = \gamma n$. Potom zřejmě $|U(Q)| \geq d(1-2\varepsilon)|Q| = d(1-2\varepsilon)\gamma n$.

Protože $|S| < 2\gamma(1-\varepsilon)n$, je $|S \backslash Q| = |S| - |Q| < \gamma n(1-2\varepsilon)$. Tedy vidíme, že z množiny $S \backslash Q$ vede méně hran, než je velikost $U(Q)$. I kdyby tyto hrany vedly do navzájem různých vrcholů $U(Q)$, stále bude existovat vrchol, do nějž z S vede právě jedna hrana. To znamená, že množina $U(S)$ je neprázdná a došli jsme ke sporu. □

Nosnost uvedeného expanderového kódu je $1-m/n$ (je zde právě m množinek vrcholů, které se musí sečíst na nulu, tedy dimenze kódu je $n-m$). V našem případě platí $m < n$, tedy je-li F nevyvážený (platí-li $m \leq \alpha n$ pro nějaké $\alpha < 1$), je nosnost kódu odražena od nuly, tudíž je sestrojený kód asymptoticky dobrý. Capalbo a kol. (2002) uvádějí explicitní konstrukci bipartitních expanderů pro jakýkoliv poměr m/n, kde $\alpha = 1-\varepsilon$ pro libovolné zvolené $\varepsilon > 0$, my ji v této práci ukazovat nebudeme. Jen uvedeme, že tato konstrukce dává konstrukci asymptoticky dobrých kódů a byla to první přímá konstrukce asymptoticky dobrých kódů (nevyužívá konkatenace jiných kódů, jak uvádějí Guruswami a Sharma (2010)). To je hezká vlastnost sama o sobě, nyní ale ukážeme, že pro expanderové kódy také existuje lineární dekódovací algoritmus.

3.1.1 Dekódovací algoritmus

V celé části analyzujeme kód $C(F)$ sestrojený z $(n,m,d,\gamma,1-\varepsilon)$-bipartitního expanderu F konstantního stupně. Nejprve uveďme pomocnou definici.

Definice 19. *O vrcholu z kontrolní partity M řekneme, že je nespokojený, pokud součet hodnot v jeho sousedech vychází 1 (sčítáme nad tělesem $\mathbb{Z}_2$). V opačném případě vrchol nazveme spokojeným.*

Každé kolo algoritmu probíhá následovně: Pokud v N existuje vrchol, který má v M více nespokojených sousedů než těch spokojených, změníme jeho hodnotu.

Přesněji, budeme si udržovat seznam takových vrcholů z N a průběžně jej aktualizovat. Budeme po něm chtít, aby z něj šlo odebírat prvky v konstantním čase a abychom do něj uměli v konstantním čase prvky přidávat.

Takový seznam se dá implementovat jako obousměrný spojový seznam, u každého vrcholu z N si budeme pamatovat ukazatel na příslušný prvek v seznamu,

pokud ten daný vrchol je v seznamu (to nám dovolí odebírat prvky v konstantním čase). Také budeme udržovat ukazatele na začátek a konec seznamu (pro přidávání prvků na konec seznamu a snadné odebrání prvního prvku ze seznamu v konstantním čase). Určitě existují i jiné lepší implementace, tato je zde jen na ukázku, že umíme najít seznam s těmito vlastnostmi.

To je celý popis, teď zbývá ukázat, že algoritmus funguje. To dokážeme sérií několik lemmat.

Lemma 24. *Jestliže nastalo více než 0 a nejvýše γn chyb, bude pro $\varepsilon < 1/4$ existovat vrchol s alespoň $d/2$ nespokojenými sousedy.*

Důkaz. Buď $T \subseteq N$ množina chybných vrcholů z N. Protože $|T| \leq \gamma n$, z lemmatu 22 máme $|U(T)| \geq d(1-2\varepsilon)|T|$ a všechny vrcholy v $U(T)$ jsou nespokojené. Protože $\varepsilon < 1/4$, platí $|U(T)| > d/2|T|$ (kvůli této vlastnosti jsme také nastavili $\varepsilon < 1/4$, ačkoliv při analýze původního grafu jsme pracovali s $\varepsilon < 1/2$), a tedy (z holubníkového principu) existuje vrchol z T, který sousedí s více než $d/2$ vrcholy z $U(T)$. □

Lemma 25. *Jestliže nastalo méně než $\gamma(1-2\varepsilon)n$ chyb, kde $\varepsilon < 1/4$, bude kdykoliv v průběhu algoritmu počet chyb menší než γn.*

Důkaz. Pokaždé, když některý z vrcholů změní hodnotu, zmenší se tím počet nespokojených vrcholů z partity M alespoň o 1. Přijali jsme slovo s méně než $\gamma(1-2\varepsilon)n$ chybami, tedy v partitě M je méně než $d\gamma(1-2\varepsilon)n$ nespokojených vrcholů (protože N je d-regulární) a toto číslo se bude snižovat. Kdybychom kdykoliv v průběhu algoritmu dostali slovo s γn chybami, bylo by v partitě M alespoň $\gamma(1-2\varepsilon)n$ nespokojených vrcholů (z lemmatu 22) a to je spor. □

Lemma 26. *Kód $C(F)$ lze dekódovat v čase lineárním s n.*

Důkaz. Pokud jsme přijali slovo s méně než $\gamma(1-2\varepsilon)n$ chybami, v každém kroku algoritmu se zmenší počet nespokojených vrcholů alespoň o 1. Algoritmus tudíž nemůže běžet déle než m kol.

Kódová vzdálenost našeho kódu je alespoň $2\gamma(1-\varepsilon)n$. V průběhu algoritmu bude ve slově méně než γn chyb, a protože $\varepsilon < 1/4$, je kódová vzdálenost našeho kódu větší než γn, tedy se přijaté slovo nemůže nikdy v průběhu algoritmu „opravit“ na jiné než nejbližší kódové slovo. Protože algoritmus je konečný, vždy dokonverguje k nejbližšímu kódovému slovu, což je to správné. Nyní ukážeme, že to celé mu zabere jen lineární čas.

Buď δ maximální stupeň vrcholu z M, předpokládejme, že δ je konstantní. Algoritmus se dělí na:

Předvýpočet: určíme, kolik máme v M nespokojených vrcholů, to nám zabere čas $\mathcal{O}(m\delta) = \mathcal{O}(n)$. Také sestrojíme seznam Q vrcholů z N, které mají více nespokojených než spokojených sousedů. To zabere čas $\mathcal{O}(nd)$. Tedy celý předvýpočet stihneme v čase $\mathcal{O}(n)$.

Běh algoritmu: ten se skládá z jednotlivých kol. V každém kole odebereme prvek ze seznamu Q, změníme jeho hodnotu a upravíme spokojenost a nespokojenost jeho sousedů v M. To stihneme v čase $\mathcal{O}(d)$. Tímto ale mohly v N vzniknout jiné vrcholy, které teď mají více nespokojených sousedů než spokojených, tedy je

chceme přidat do Q. Nebo naopak některé vrcholy, které doposud byly v Q, mohou nyní mít více spokojených sousedů než nespokojených, a proto do Q už nepatří. Tyto vrcholy ale musí sousedit z některým z D sousedů opraveného vrcholu, tedy jejich prohledání a update seznamu Q zabere nejvýš $\mathcal{O}(\delta d^2)$ času, tedy konstantní čas. A protože kol algoritmu je nejvýše m, celé to zabere $\mathcal{O}(n)$ času. □

Následující lemma je shrnutím pozorovaných vlastností:

Lemma 27. *Jestliže nastalo méně než $\gamma(1-2\varepsilon)n$ chyb, kde $\varepsilon < 1/4$, algoritmus je všechny opraví v čase lineárním s n.*

Jak jsme ukázali, tyto kódy mohou dosahovat dobrých parametrů, ale mají vysoké nároky na expanzi grafu. Proto ukážeme jinou konstrukci, která už není čistý expanderový kód, ale nepotřebuje tak dobré expandery.

3.2 Tannerova konstrukce

Tuto konstrukci popsal Tanner (1981) a je mile jednoduchá.

Nový kód se vytvoří z bipartitního grafu a kratšího kódu C_0. Partity grafu označíme X a Y, X ponese zprávu, Y bude kontrolní. Vybereme si vrchol z Y. Ten na své sousedy v X napíše nějaké kódové slovo C_0. Pak vybereme další vrchol z Y a ten na své sousedy dopíše další symboly abecedy tak, aby také viděl korektní kódové slovo C_0. Tohle provedeme pro všechny vrcholy z Y a pokud se to povede (tj. pokud na konci každý vrchol z Y vidí na svých sousedech korektní kódové slovo z C_0), prohlásíme partitu X za kódové slovo nového kódu.

Nyní přistupme k formální definici:

Definice 20. *Mějme bipartitní graf H s jednou partitou X o velikosti n a druhou partitou Y o velikosti m. Každý vrchol z Y buď stupně d. Očíslujeme sousedy i-tého vrcholu z Y jako $\Gamma(i,1)$, $\Gamma(i,2)$, …, $\Gamma(i,d)$. Mějme také binární kód C_0 délky d. Kódovými slovy nového kódu T jsou všechny n-tice $(x_1,x_2,\ldots,x_n)$ takové, že $(x_{\Gamma(i,1)},x_{\Gamma(i,2)},\ldots,x_{\Gamma(i,d)})$ je kódovým slovem C_0 pro všechna $i \in \{1,\ldots,m\}$.*

Pozorování:

- Tannerovy kódy jsou lineární, je-li C_0 lineární.
- Tannerovy kódy jsou zobecněním expanderových kódů, je-li H je bipartitní expander. Expanderové kódy totiž explicitně volí C_0 jako $[d,d-1,2]$-kód (tedy kód, který ke zprávě jen přidá paritní bit).

Tvrdíme, že dimenze T je alespoň $n - m(d - \dim(C_0))$. To, že v sousedství každého vrcholu z Y je kódové slovo, znamená, že pro každý vrchol $i \in Y$ dostáváme $d-\dim(C_0)$ závislých bitů, tedy vrcholů v X, na kterých jsou napsány symboly, které nenesou informaci. Tedy pro všechny vrcholy z Y máme nejvýše $m(d - \dim(C_0))$ závislých vrcholů v X, a z toho plyne tvrzení.

3.2.1 Konstrukce grafu

Abychom dosáhli vysoké nosnosti, měl by G být silně nevyvážený, partita Y by tedy měla být mnohem menší než partita X (ať nemáme zbytečně mnoho kontrolních bitů). Takový graf lze ale sestrojit z obecného expanderu (s dostatečně velkým stupněm) pomocí incidenční matice, tedy „rozdělením hran“, jak jsme uvedli v předchozí kapitole.

To znamená, že si tento kód můžeme představovat jako d-regulární expander o m vrcholech a n hranách (označme si jej G_0) a napíšeme na hrany jedničky a nuly tak, že každý vrchol vidí kódové slovo kódu C_0. Kódovým slovem T je pak n-tice symbolů na hranách (tedy všechny hrany) v nějakém předem zvoleném pořadí. Je asi názornější a jednodušší o kódu přemýšlet v této podobě.

A co vlastně činí expandery tak užitečnými? Intuitivně by se dalo říci, že je to jejich náhodnost a to, že se množinky vrcholů navzájem „kontrolují“. Přesněji to vyjádříme mixážním lemmatem (anglicky Expander Mixing Lemma), které dokázali Guruswami a Sharma (2010), my jej zde dokazovat nebudeme.

Lemma 28 (mixážní)**.** *Ať $G = (V,E)$ je (n,d,λ)-expander. Potom $\forall X,Y \subseteq V$ platí*

$$\left| E(X,Y) - \frac{d|X||Y|}{n} \right| \leq d\lambda\sqrt{|X||Y|}\,,$$

kde $E(X,Y)$ je počet hran mezi množinami X a Y, přičemž hrany v jejich průniku započítáme dvakrát.

Co vztah výše vlastně znamená? Číslo $E(X,Y)$ udává opravdový počet hran mezi X a Y, $d|X|$ je celkový počet hran jdoucí z množiny X a $|Y|/n$ udává pravděpodobnost, že se hrana jdoucí odkudkoliv (tedy klidně i z množiny X) v náhodném bipartitním grafu trefí do množiny $|Y|$, tedy součin těchto dvou věcí nám říká, kolik hran z množiny X se v rovnoměrně náhodném grafu trefí do Y. Rozdíl výše udává odchylku grafu G od náhodného grafu, tedy o kolik hran se bude lišit. Výraz vpravo udává horní mez pro tuto odchylku

Uvedeme také bipartitní verzi tohoto lemmatu.

Lemma 29 (bipartitní mixážní)**.** *Buďte A, B partity G. Nechť je G na obou stranách d-regulární bipartitní expander s λ definovaným výše a nechť $|A| = |B| = n$. Graf G má tedy $e = nd$ hran. Pro všechny podmnožiny $X \subseteq A$ a $Y \subseteq B$ platí*

$$\left| E(X,Y) - \frac{d|X||Y|}{n} \right| \leq d\lambda\sqrt{|X||Y|}\,.$$

Důsledek: Vydělíme-li celý vztah členem $d|X|$ (je-li X neprázdná), dostaneme

$$\left| \frac{E(X,Y)}{d|X|} - \frac{|Y|}{n} \right| \leq \lambda\sqrt{\frac{|Y|}{|X|}}\,.$$

Tento vztah má také své opodstatnění. První zlomek udává, jaký zlomek hran z X jde do Y, druhý udává pravděpodobnost, že se v rovnoměrně náhodném bipartitním grafu nějaká hrana trefí do Y. Tedy zde je také uvedena odchylka G od náhodného grafu, tentokrát v termínech pravděpodobností. Tento vztah má ale tu výhodu, že pokud jde λ s rostoucím d k nule, výraz na pravé straně

nerovnice jde také k nule, tedy už je vidět, že pro velká d se G chová jako náhodný graf, řekněme pseudonáhodně, a to je něco, co se nám u samoopravného kódu hodí, protože náhodný samoopravný kód je dobrý (Shannonova věta, vizte skripta Drápal (2008/2009)).

Algoritmus na dekódování se stává úžasně jednoduchým, je-li G bipartitní, a pro „pseudonáhodnost“ navíc potřebujeme, aby byly obě partity stejně velké, tedy pro analýzu dekódovacího algoritmu zvolme „vyváženou bipartitní“ verzi G_0. Tohoto můžeme dosáhnout, pokud si $G = (A,B,E)$ zvolíme jako dvojpokrytí $(m/2,d,\lambda)$-expanderu (čísla m,n, d jako v konstrukci 20), kde λ jde s rostoucím d k nule (kvůli pseudonáhodné vlastnosti výše). Kód je napsaný na hranách. Když tuto konstrukci srovnáme s konstrukcí Tannerova kódu ve větě 20, vidíme, že nový graf má $md/2 = n$ hran, tedy délka kódu se nezmění. Označme $m' = m/2$.

Věta 30. *Ať má C_0 relativní kódovou vzdálenost alespoň δ_0, kde $\delta_0 \in (0,1)$. Potom relativní kódová vzdálenost kódu T je alespoň $\delta_0(\delta_0 - \lambda)$.*

Důkaz. Protože T je lineární kód, stačí dokázat dolní mez $m'd\delta_0(\delta_0 - \lambda)$ pro váhu jeho minimálního slova (délka T je $n = m'd$).

Buď $c \in T$ a F buď množina hran, které odpovídají nenulovým bitům v c. Buďte $X \subseteq A$, $Y \subseteq B$ množiny vrcholů G, z nichž alespoň jedna hrana leží v F. Z každého vrcholu z G, který je přilehlý k F, vede alespoň $d\delta_0$ hran z F, protože vzdálenost, a tedy i váha, C_0 je $d\delta_0$. Tedy $|F| \geq \delta_0 d|X|$ a $|F| \geq \delta_0 d|Y|$ (protože z každého vrcholu z dané partity vede alespoň $\delta_0 d$ hran z $|F|$), tedy

$$|F| \geq \delta_0 d\sqrt{|X||Y|}\,.$$

Na druhou stranu

$$|F| \leq E(X,Y)\,.$$

Navíc z mixážního lemmatu dostáváme

$$E(X,Y) \leq d|X||Y|/m' + d\lambda\sqrt{|X||Y|}\,,$$

je-li první zlomek větší než ten druhý, a

$$E(X,Y) \leq d|X||Y|/m' \leq d|X||Y|/m' + d\lambda\sqrt{|X||Y|}$$

v druhém případě, tedy v každém případě

$$E(X,Y) \leq d|X||Y|/m' + d\lambda\sqrt{|X||Y|}\,.$$

Tedy, když to spojíme,

$$\delta_0 d\sqrt{|X||Y|} \leq d|X||Y|/m' + d\lambda\sqrt{|X||Y|}\,.$$

Tedy

$$\delta_0 \leq \sqrt{|X||Y|}/m' + \lambda\,,$$

a tím pádem

$$\sqrt{|X||Y|} \geq (\delta_0 - \lambda)m'\,.$$

Vzpomeneme si, že $|F| \geq \delta_0 d\sqrt{|X||Y|}$, tedy

$$|F| \geq \delta_0 d m'(\delta_0 - \lambda)\,,$$

čímž jsme hotovi. □

V našich konstrukcích expanderů se s rostoucím d snižuje λ, tedy pro každé λ umíme najít vhodný graf, byť s velkým stupněm. Z této věty dostáváme, že pro δ_0 relativní kódovou vzdálenost C_0 a λ jdoucí pro velká d k nule je relativní kódová vzdálenost T přibližně δ_0^2, tedy mnohem horší než u C_0. Přesněji to řekneme vzápětí.

Kompromis mezi nosností a kódovou vzdáleností

Vypočítejme nosnost kódu T. Označíme nosnost kódu C_0 jako $R(C_0)$, tedy $d(1 - R(C_0))$ je počet kontrolních bitů C_0. Tento kód je v bipartitním G napsán celkem $2m'$-krát, tedy počet redundantních bitů T je $2m'd(1 - R(C_0))$. Z toho již snadno získáme, že nosnost T je

$$\frac{m'd - 2m'd(1 - R(C_0))}{m'd} = 2R(C_0) - 1\,.$$

Relativní kódová vzdálenost T je $\delta \approx \delta_0^2$ (jak jsme ukázali, toto platí pro velká d). Zvolíme-li C_0 tak, aby splňovalo $R(C_0) \geq 1 - H(\delta_0)$ (tedy aby dosahovalo Gilbertovy-Varšamovovy meze), dostaneme

$$R(T) \geq 2(1 - H(\delta_0)) - 1 \approx 1 - 2H(\sqrt{\delta})\,.$$

Tedy chceme, aby $H(\sqrt{\delta}) < 1/2$ a zároveň (z předpokladu tvrzení 9) $\sqrt{\delta} < < 1/2$. Po zadání do nějakého výpočetního programu (například WolframAlpha) dostáváme $\delta < 0.0121$. To je dost malá kódová vzdálenost.

3.2.2 Dekódování

Už víme, jak se kód C kóduje a jaké má parametry. Nyní musíme opravdu ukázat, že umí opravit tolik chyb, kolik slibuje.

Buďte tedy A, B partity G a na hranách mezi nimi buď napsána zpráva (v našem případě přijaté slovo). Buď $\delta_0 d$ minimální vzdálenost C_0. U korektního kódového slova by měly symboly na hranách přilehlých k jednomu vrcholu tvořit kódové slovo kódu C_0. Pokud nastaly chyby, nejspíš u některých vrcholů nepřijmeme korektní slovo. Napadne nás pro každý z vrcholů opravit symboly na hranách tak, aby vzniklo korektní kódové slovo. Ale co když se dva vrcholy na jedné hraně neshodnou? Z toho důvodu máme bipartitní graf – nejprve si hrany opraví partita A, pak partita B, a tak se střídají tak dlouho, dokud buď už není co opravovat, nebo dokud neproběhne $\mathcal{O}(\log n)$ kol (jak se ukáže později, potom stejně nastalo víc chyb, než je náš kód schopen správně opravit). A teď formálně.

Algoritmus probíhá po kolech. Každé kolo se skládá z následujících částí:

Levice: Paralelně pro každé $u \in A$ zjisti, zda vidí korektní kódové slovo kódu C_0, pokud ne, oprav jej. Přesněji, pokud tento vrchol vidí slovo x, zjisti, zda existuje kódové slovo z ve vzdálenosti menší než $\delta_0 d/2$ od x a pokud ano, nastav hrany přilehlé k tomuto vrcholu na z.

Pravice: To samé, ale pro partitu B.

Každé kolo lze zřejmě implementovat tak, aby proběhlo v lineárním čase (tedy celková časová složitost algoritmu je v nejhorším případě $\mathcal{O}(n \log n)$), a když si dáme pozor na to, které vrcholy se musí účastnit kterých kol, umíme celkovou dobu běhu algoritmu stlačit na $\mathcal{O}(n)$.

Základní idea důkazu spočívá v tom, že precizně evidujeme množiny vrcholů, které jsou aktivní v i-tém kole (tedy ty, které v i-tém kole opraví nějakou chybu), a vypozorujeme, že se exponenciálně zmenšují. Protože na začátku bylo aktivních vrcholů nejvýše n, opravdu stačí logaritmický počet kol a velikosti aktivních množin se sečtou (podle vzorce na součet geometrické řady) na $\mathcal{O}(n)$, což je i celková doba běhu algoritmu.

Nyní ukážeme, že se množiny vrcholů opravdu zmenšují exponenciálně, a tudíž stačí logaritmický počet kol.

Věta 31. *Předpokládejme, že $\lambda < \delta_0/3$ a mějme $\varepsilon > 0$. Je-li chybovost nejvýše $(1-\varepsilon)\frac{\delta_0}{2}\left(\frac{\delta_0}{2} - \lambda\right)$, dekódovací algoritmus dostane správné kódové slovo v $\mathcal{O}(\log m')$ iteracích.*

Důkaz. Nejprve si zaveďme značení. Označme A_i množinu vrcholů z první partity, které se špatně zdekódují v i-tém kole, podobně pro B_i. Také uvažme množinu E_i chybných hran přímo po i-tém kole. Buď $E(X,Y)$ počet hran mezi množinami X a Y.

Speciálně mějme množinu A_1. Přímo před prvním krokem musely existovat vrcholy, u nichž byl počet chyb alespoň $\delta_0 d/2$ (protože se nezdekódovaly správně). Tedy $|E_0| \geq |A_1|\delta_0 d/2$. Z předpokladu věty $|E_0| \leq (1-\varepsilon)\frac{\delta_0}{2}\left(\frac{\delta_0}{2} - \lambda\right) m'd$, a tedy $|A_1| \leq (1-\varepsilon)\left(\frac{\delta_0}{2} - \lambda\right) m'$.

Nyní uvažujme množinu B_1. Ukážeme, že $|B_1| \leq \alpha|A_1|$ pro nějaké $\alpha < 1$.

V množině B_1 také musí být právě ty vrcholy, u nichž byl počet chyb před tímto dekódovacím krokem alespoň $\delta_0 d/2$. Jenže ty chyby se tam mohly dostat jen z množiny A_1, tedy $E(A_1,B_1) \geq \delta_0 d/2|B_1|$.

Pomocí mixážního lemmatu shora odhadneme počet hran mezi vrcholy A_1 a B_1 výrazem $d|A_1||B_1|/m' + d\lambda\sqrt{|A_1||B_1|}$ (stejná argumentace jako v důkazu věty 30). Z nerovnosti mezi aritmetickým a geometrickým průměrem víme, že $(|A_1| + |B_1|)/2 \geq \sqrt{|A_1||B_1|}$. Dosadíme do druhého členu odhadu, první člen odhadneme pomocí $|A_1| \leq (1-\varepsilon)\left(\frac{\delta_0}{2} - \lambda\right) m'$ a dostáváme

$$E(A_1,B_1) \leq d|B_1|(1-\varepsilon)\left(\frac{\delta_0}{2} - \lambda\right) + d\lambda\frac{|A_1| + |B_1|}{2}\,.$$

Dosadíme $\delta_0 d/2|A_1| \leq E(A_1,B_1)$ a po přeskládání dostaneme horní odhad na $|B_1|$:

$$|B_1| \leq |A_1|\frac{d\lambda}{\delta_0 d - (\delta_0 d - 2d\lambda)(1-\varepsilon) - d\lambda} = |A_1|\frac{d\lambda}{\varepsilon\delta_0 d + (1-2\varepsilon)d\lambda}$$

a z toho, že $\delta_0 d > 3d\lambda$, konečně dostáváme

$$|B_1| < |A_1|\frac{d\lambda}{3\varepsilon d\lambda + (1-2\varepsilon)d\lambda} = \frac{|A_1|}{1+\varepsilon}\,.$$

Stejnou úvahu provedeme pro všechny iterace dekódovacího algoritmu a pro všechna i vidíme, že platí $|B_i| < |A_i|/(1+\varepsilon)$. Podobnou argumentací získáme $|A_{i+1}| < \frac{|B_i|}{1+\varepsilon}$. Tedy velikosti A_i a B_i klesají exponenciálně o základu $1-\varepsilon$. □

3.2.3 Shrnutí

V části 3.2.1 jsme ukázali, že kód C_0 dosahuje Gilbertovy-Varšamovovy meze pro $\delta < 0.0121$. Tedy z toho, co jsme ukázali při rozboru dekódovacího algoritmu, platí tvrzení:

Tvrzení 32. *Pro každé $\varepsilon > 0$ a pro každé $0 < \delta < 0.0121$ existuje rodina binárních lineárních kódů o nosnosti $1 - 2H(\sqrt{\delta})$ a relativní vzdálenosti $\delta - \varepsilon$, které opraví až $\delta/4 - \varepsilon$ chyb v lineárním čase. Speciálně, sestrojený kód T těchto parametrů také dosahuje.*

3.3 Vylepšená konstrukce

Jak jsme ukázali, Tannerův kód je sice dekódovatelný v lineárním čase, ale co se týče kódové vzdálenosti, a tedy i množství opravených chyb, není to nic moc. V článku Guruswami (2004) ukazuje, jak (zase) využít expandery, abychom zlepšili odolnost kódu a konečně dostali alespoň trochu dobré parametry. Vylepšíme ho až na relativní vzdálenost rovnou $1/2 - \varepsilon$ a odolnost proti $1/4 - \varepsilon$ chybám, to celé se zachováním lineárního dekódování. Sestrojený kód už nemusí být binární, tedy jej nemůžeme touto konstrukcí vylepšit opakovaně, ale z první konstrukce získáme natolik dobré parametry, že to už nebude potřeba.

3.3.1 Základní konstrukce

Nyní popíšeme, jak můžeme použít expandery k „promíchání“ symbolů kódu.

Myšlenka za následující definicí je jednoduchá. Zprávu z nejprve zakódujeme kódem C a potom každý ze symbolů zakódujeme kódem C_0 (je to prostě složený kód). To, co vypadlo z kódu C_0, napíšeme na vrcholy X a potom každý vrchol z X zpropaguje svůj symbol tak, že jej pošle po všech hranách, co z něj vedou, v pořadí určeném funkcí f. Každý vrchol z Y poté sesbírá všechny přijaté symboly a slepí je dohromady. A to je celé. Nyní to řekneme formálně.

Definice 21. *Mějme bipartitní graf $G = (X,Y,E)$, kde $X = [n]$ je c-regulární a $Y = [m]$ je d-regulární. Také mějme dva samoopravné kódy C a C_0, kde C je délky n nad abecedou $\{0,1\}^a$ a C_0 je kód délky c a dimenze a. Pro $i \in [m]$ buďte $\Gamma(i,1), \ldots, \Gamma(i,d)$ sousedé vrcholu $i \in Y$ v X. Seřaďme si sousedy jednotlivých vrcholů z X pomocí (nějaké) funkce f tak, že vrchol s číslem i je $f(i,j)$-tý soused $\Gamma(i,j)$. Potom definujme kód $G(C,C_0)$ nad abecedou $\{0,1\}^d$ tak, že jeho i-tý symbol po zakódování zprávy z je d-tice, jejíž j-tá složka je rovna $C_0(C(z)_{|\Gamma(i,j)})_{|f(i,j)}$. Dolní indexy zde specifikují, jaké uspořádání daný kód využívá.*

Všimneme si, že nosnost vzniklého kódu je součinem nosností kódů C a C_0, graf G nám žádnou redundanci nepřidává. Toto schéma v našem případě funguje překvapivě dobře, je-li C_0 obyčejný opakovací kód (zobrazí 0 na 0^a a 1 na 1^a).

Občas samotná expanze o grafu poskytuje málo informací. Zadefinujme tedy objekty podobné expanderům.

Definice 22 (disperser)**.** *Buď $G = (X,Y,E)$ bipartitní graf. Řekneme, že G je (α,β)-disperser, pokud každá množina vrcholů z X o relativní velikosti α má alespoň relativně β sousedů v Y.*

Je vidět, že je-li graf dobrý disperser, je i dobrý expander, definice jsou podobné. To, že dispersery existují, je dokázáno například v článku Capalbo a kol. (2002).

A proč o tom mluvíme? Guruswami (2004) tvrdí, že je-li G (α,β)-disperser a C je binární kód o relativní vzdálenosti α, potom $G(C,C_0)$ má relativní vzdálenost alespoň β a (protože nosnost opakovacího kódu C_0 je $1/c$) nosnost $R(C)/c$. Ještě uvádí, že existují explicitní konstrukce $(0.1,1-\varepsilon)$-disperserů.

S využitím takových grafů umíme zkonstruovat kódy o relativní vzdálenosti $(1-\varepsilon)$ a nosnosti $\Omega(\varepsilon)$ nad abecedou velikosti $2^{\mathcal{O}(1/\varepsilon)}$. To je (ze Singletonovy meze) odraženo od optimální nosnosti jen konstantou.

3.3.2 Blížení se k optimu

Nyní se podíváme, jak můžeme využít schéma z definice 21, abychom dosáhli kýžených parametrů.

Věta 33. *Pro každé $r \in (0,1)$ a $\varepsilon > 0$ existuje rodina kódů s nosností r a relativní kódovou vzdáleností alespoň $1-r-\varepsilon$, které opraví až $(1-r-\varepsilon)/2$ chyb v čase lineárním s délkou kódu.*

Důkaz. Využijeme schéma z definice 21. Nejprve si nadefinujeme konstanty, se kterými pak budeme počítat. Nechť $\gamma = \varepsilon/4$. Kód C získáme z rodiny, která je popsána v tvrzení 32 (Tannerův kód s nízkou kódovou vzdáleností, ale jinak dobrými parametry) a bude mít nosnost $1/(1+\gamma)$, délku bloku n a opraví až relativně $\beta > \gamma^3$ chyb. Zvolíme $G = (A,B,E)$ jako d-regulární bipartitní graf s n vrcholy v každé partitě, který splňuje pseudonáhodnou vlastnost ze sekce 3.2.1 (platí pro něj bipartitní verze mixážního lemmatu) a ve kterém $d = \Theta(1/\beta\varepsilon^2)$.

Z bipartitního mixážního lemmatu pro $\lambda < 2/\sqrt{d}$ plyne (jak tvrdí v článku Guruswami (2004)), že pro každou množinu $R \subseteq Y$, kde $|R| \geq \alpha n$ pro nějaké $\alpha < 1$, platí, že každá množina $L \subseteq X$, jejíž všechny vrcholy mají alespoň $(\alpha-\gamma)d$ sousedů v R, splňuje $|L| \geq (1-\beta)n$. Tato vlastnost se nám bude hodit při analýze počtu opravených chyb.

C_0 bude konstantně velký Reedův-Solomonův kód o délce d a nosnosti $r(1+\gamma)$. Tento kód sice, přísně vzato, nesplňuje definici, ale to jednoduše obejdeme tím, že zprávu zakódovanou tímto kódem převedeme na binární řetězec.

Konstrukce kódu probíhá stejně jako v definici 21, tedy symboly kódového slova kódu C „srazíme“ do bloků správné velikosti a pak je zakódujeme kódem C_0. Nosnost kódu $G(C,C_0)$ je zřejmě součinem nosností obou menších kódů, tedy je rovna r. Tento kód lze zřejmě zakódovat v lineárním čase, protože to umíme u C. Kód C_0 je konstantní délky, tam nám na konkrétním kódovacím algoritmu nezáleží.

Nyní se podíváme na dekódování. Představme si, že jsme přijali slovo s nejvýše $(1-r-\varepsilon)n/2$ chybami. Toto slovo napíšeme na vrcholy partity Y. V prvním kroku rozdělíme symboly do bloků délky d (což je délka C_0) a dekódujeme C_0 (tedy odstraníme jednu vrstvu zakódování). Konkrétní dekódovací algoritmus není důležitý, i hledání nejbližšího slova hrubou silou zabere čas $\mathcal{O}(1)$ na blok. Nakonec se získané slovo dekóduje algoritmem pro C, který je lineární. Kód C opraví nejvýše βn chyb, tedy chceme dokázat, že víc jich nebude.

Označíme R množinu vrcholů z partity Y, na kterých není chyba, tedy $|R| = (1+r+\varepsilon)n/2$. Využijeme vlastnost plynoucí z mixážního lemmatu, kterou jsme

popsali výše. Zvolíme tedy $\alpha = (1+r+\varepsilon)/2 < 1$ a dostáváme, že každá množina $L \subseteq X$, jejíž všechny vrcholy mají alespoň $(\alpha-\gamma)d = ((1+r)/2+\varepsilon/4)d$ sousedů v R, splňuje $|L| \geq (1-\beta)n$. Každý vrchol z L sousedí s alespoň $((1+r)/2+\gamma)d$ správnými vrcholy, tedy má nejvýše $((1-r)/2-\gamma)d$ chybných sousedů. Všimněme si, že Reedův-Solomonův kód o nosnosti $r(1+\gamma)$ opraví až $(d - dr(1+\gamma))/2 = = ((1-r)/2-r\gamma/2)d$ chyb. Nahlédneme, že $((1-r)/2-\gamma)d \leq ((1-r)/2-r\gamma/2)d$, tedy že $\gamma \geq r\gamma/2$, takže $r \leq 2$, a to určitě platí, neboť $r < 1$.

Tím pádem jsme zjistili, že alespoň $(1-\beta)n$ přijatých slov bude mít nejvýše $((1-r)/2+\gamma)d$ chyb a ty při prvním průchodu opraví Reedův-Solomonův kód v C_0. Do druhé vrstvy tedy půjdou jen špatně zdekódovaná slova, kterých bude nejvýš βn. To zvládne opravit kód C, navíc v lineárním čase, a tedy časová složitost celého algoritmu je lineární. □

Skoro optimální součet nosnosti a kódové vzdálenosti dostáváme díky použití Reedových-Solomonových kódů. Na schéma výše totiž můžeme nahlížet jako na několik nezávislých RS „krabiček“, kde je expander použit pouze k rozvrstvení symbolů (protože chyby se vyskytují ve skupinách, takže tímto graf dává RS kódu možnost využít svůj potenciál). Tedy většina chyb bude opravena už při prvním průchodu a do druhého se jich dostane jen malá část, kterou opraví kód C. Tímto na C nemáme moc velké nároky, co se počtu opravených chyb týče, takže může mít nosnost blízkou 1, a tím pádem ji moc nekazit celému kódu. Můžeme tedy za C dosadit Tannerův kód z definice 20, který má parametry z věty 32, a tím sestrojit výborný kód.

4. Sestrojení konkrétního kódu

V předchozích kapitolách jsme vybudovali aparát, kterým můžeme efektivně vytvářet expanderové kódy, a dokázali jsme, že produkuje kódy s asymptoticky dobrými parametry. Ještě jsme ale neviděli ani jeden opravdový případ a nevíme, jestli tyto kódy náhodou nejsou moc velké. V této kapitole sestrojíme alespoň jeden příklad expanderového kódu a podíváme se na jeho reálnou využitelnost.

Chceme najít kód C_0 délky d s relativní vzdáleností δ_0, který použijeme v Tannerově konstrukci. Aby dekódování Tannerova kódu doběhlo v lineárním čase, potřebujeme, aby $\delta_0/3 > \lambda$. Tedy vidíme, že λ musí být opravdu dobré. Naše explicitní konstrukce rodin expanderů nám λ nezlepšují, tedy si budeme muset vystačit s některým ze základních grafů. U těch ale nedosáhneme lepšího λ než $\lambda = \mathcal{O}(1/\sqrt{d})$. Navíc d musí být (potenciálně sudá) mocnina prvočísla.

Označme absolutní kódovou vzdálenost kódu C_0 jako δ. Platí $\delta_0 = \delta/d$, a když z konstrukce dostaneme $\lambda = 1/\sqrt{d}$, musí platit $\delta/(3d) > 1/\sqrt{d}$, tedy potřebujeme $\delta > 3\sqrt{d}$. Navíc C_0 musí být binární.

Existují vůbec takové kódy? Určitě umíme někde sehnat takové kódy C_0, zkusíme nyní sestrojit co nejkratší Tannerův kód. Tedy za C_0 vezmeme opakovací kód s délkou bloku 11. Ten těsně splňuje podmínku $\delta_0/3 > \lambda$. Využijeme základní graf z konstrukce 19, tedy jeho délka bloku je 121. Když tento graf s kódem C_0 dosadíme do konstrukce Tannerova kódu, dostaneme kód délky 1331 s velmi dobrou kódovou vzdáleností a maličkou nosností. Kvůli relativní kódové vzdálenosti větší než 0.5 nemá smysl pokračovat do konstrukce v definici 21.

Všimneme si, že v tomto případě ale Tannerův kód zdegeneruje na opakovací kód. Použijeme „intuitivní“ náhled na Tannerův kód jako na d-regulární expander s kódem napsaným na hranách a všimneme si, že je-li alespoň na jedné hraně jednička, je i na všech sousedních hranách (koncové vrcholy této hrany musí vidět samé jedničky), tedy jednička je napsaná i na všech sousedech sousedů této hrany… tedy jsou jedničky všude. Takže nejmenší expanderový kód, který umíme získat z uvedených konstrukcí, je binární opakovací kód délky 1331.

Co kdybychom chtěli kód, který bude dosahovat optimální nosnosti? Z věty 32 víme, že v tom případě musí C_0 splňovat Gilbertovu-Varšamovovu mez a také, že $\sqrt{\delta_0} < 0.121$, tedy $\delta_0 < 0.11$. Víme, že musí platit $\delta_0/3 > 1/\sqrt{d}$, tedy v tom případě $d \geq 744$. To implikuje nejmenší možný počet vrcholů grafu $d^2 = 553\,536$. Délka zprávy (z Tannerova kódu) je tedy $d^3 = 411\,830\,784$, což je zpráva velká zhruba 50 MB, a to je na reálné využití prostě moc, například obvyklý internetový paket má velikost kolem kilobytu. Možná se tak dlouhý kód přece jenom dá někde využít, ale rozhodně ne při běžné komunikaci.

Závěr

V této práci jsme popsali efektivní konstrukce expanderů a expanderových kódů a ukázali jsme, že asymptoticky mohou tyto kódy dosahovat parametrů blízkých optimu. U vět, kde se používala $\mathcal{O}$-čková notace pro odhad λ, jsme $\mathcal{O}()$ nahradili konstantou a tím zlepšili některé odhady z článku Reingold a kol. (2002).

Bohužel se nám ale nepovedlo najít malý expanderový kód s dobrou nosností a kódovou vzdáleností. Problém vidíme hlavně v tom, že naše konstrukce expanderů neumí sestrojit malý expander s dobrými vlastnostmi. Lepší konstrukce existují (jak se ukazuje například v článku Capalbo a kol. (2002)), ale jejich popis už byl nad rámec této práce, mimo jiné proto, že jsou výrazně komplikovanější. Dalším problémem může být to, že konstrukce kódů požadují po grafu příliš dobré expanzní vlastnosti a tím tlačí počet vrcholů nahoru, ale „benevolentnější“ konstrukce kódů bohužel neznáme.

Dalším problémem, na který jsme v práci narazili, je to, že nemáme žádný algoritmus na nalezení kódu C_0 pro Tannerovu konstrukci. Některé známé třídy kódů nešly použít, protože implikovaly moc vysoké druhé vlastní číslo (například Golayův kód $\mathcal{G}_{23}$), některé se použít daly, ale buď už byly příliš dlouhé nebo sestrojený Tannerův kód zdegeneroval (jako v příkladu s opakovacím kódem).

Nabízí se využít některé z lepších konstrukcí expanderů uvedených v článku. Nebo, protože náhodný graf je s vysokou pravděpodobností expander, lze najít vhodný graf hrubou silou, ale je otázka, zda by tímto způsobem nalezené grafy nebyly naopak moc malé.

Také se nabízí hledat vhodné kódy C_0 hrubou silou (protože náhodný kód je dobrý). Je potřeba počítat s tím, že ověřit, zda je nalezený kód dobrý, není jednoduché. S velkou pravděpodobností by takto nalezený kód nebyl lineární a sestrojený Tannerův kód by nebyl LDPC, tedy ani expanderový, ale to nám nevadí, uvedené konstrukce nevyžadují, aby C_0 byl lineární. U větších nelineárních kódů bychom v takovém případě mohli narazit na potíže s pamětí, ale tady nás může zachránit to, že umíme konstruovat expandery lokálně (v žádné z uvedených konstrukcí není potřeba mít v paměti celou rotační mapu).

Na závěr zmíníme, že existují explicitní konstrukce samoopravných kódů, které dosahují Gilbertovy-Varšamovovy meze (například Goppa kódy, Xing (2005)). Třeba právě Goppa kódy, které mohou být malé a přitom dosahovat slušných parametrů, by mohly posloužit jako výborný kód C_0, pokud najdeme nějakou lepší konstrukci expanderů.

Seznam použité literatury

Barto, L. a Tůma, J. (2016). *Lineární algebra, 5. vydání*. http://www.karlin.mff.cuni.cz/~barto/LinAlg/skripta_la5.pdf.

Capalbo, M., Reingold, O., Vadhan, S. a Wigderson, A. (2002). Randomness conductors and constant-degree lossless expanders. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 659–668. ACM.

Drápal, A. (2008/2009). *Samoopravné kódy*. http://www.karlin.mff.cuni.cz/~holub/soubory/drapal_kody.pdf.

Guruswami, V. (2004). Guest column: error-correcting codes and expander graphs. *ACM SIGACT News*, **35**(3), 25–41.

Guruswami, V. (2010). Notes 2: Gilbert-Varshamov bound. In lecture notes on *Introduction to Coding Theory*, pages 1–5. Carnegie Mellon University.

Guruswami, V. a Sharma, A. (2010). Notes 8: Expander Codes and their decoding. In lecture notes on *Introduction to Coding Theory*, pages 1–12. Carnegie Mellon University.

Hladík, M. (2017). *Lineární algebra (nejen) pro informatiky*. https://kam.mff.cuni.cz/~hladik/LA/text_la.pdf.

Reingold, O., Vadhan, S. a Wigderson, A. (2002). Entropy waves, the zig-zag graph product, and new constant-degree expanders. *Annals of mathematics*, pages 157–187.

Tanner, R. (1981). A recursive approach to low complexity codes. *IEEE Transactions on information theory*, **27**(5), 533–547.

Xing, C. (2005). Goppa geometric codes achieving the Gilbert-Varshamov bound. *IEEE transactions on information theory*, **51**(1), 259–264.