Sveučilište u Zagrebu
Fakultet elektrotehnike i računarstva

Zaštitno kodiranje

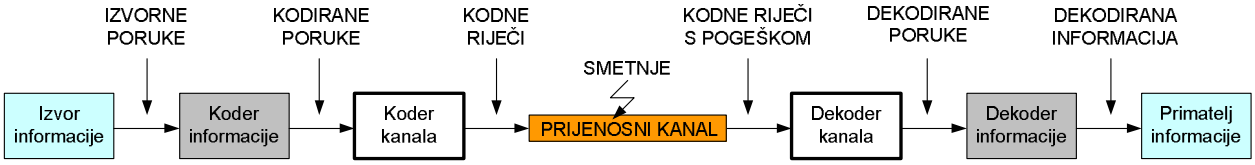
Mr. sc. Zdenko Vrdoljak

Dopunski materijal za predmet Teorija informacije
(nepročišćena verzija)

Zagreb, 2006.

Zadatak nekog komunikacijskog sustava je što vjernije prikazati informacijski sadržaj izvora informacije primatelju informacije koji se nalazi na fizički udaljenoj lokaciji. Kako bi informacijski sadržaj bilo moguće prenositi, on mora dobiti fizikalnu reprezentaciju pogodnu za zapis i prijenos – slijed kodiranih poruka (Slika 0.1). Svaka kodirana poruka sastoji se od niza simbola izabranih iz konačne abecede simbola. Kodirane poruke nastaju u koderu informacije postupkom kodiranja koji je posebno prilagođen prirodi izvora informacije. Dekoder informacije na prijemnoj strani vrši dekodiranje primljenih kodiranih poruka i primatelju informacije predaje sadržaj poslane informacije.

Prijenos poruka formiranih u koderu informacije podložan je pogreškama uzrokovanim smetnjama u prijenosnom kanalu. Pogreška se očituje kao pogrešan prijenos jednog ili više simbola poslane poruke. Smetnje su najviše izražene u prijenosnom mediju, kao sastavnom dijelu prijenosnog kanala. Simboli poruke se u prijenosnom mediju preslikavaju u određene oblike elektromagnetskog vala, kojeg nazivamo signal. Svako izobličenje oblika signala uzrokuje pogrešnu detekciju simbola koji je poslan u prijemniku, a time i pogrešku u kodiranoj poruci. Izobličenju signala najviše doprinosi šum. Šum najvećim dijelom nastaje kao napon induciran djelovanjem okoline na promatrani medij. Također, signal se prilikom propagacije izobličuje i zbog međudjelovanja elektromagnetskog vala sa samim prijenosnim medijem.



Slika 0.1: Pojednostavljeni model komunikacijskog sustava

Među prioritetima komunikacijskog sustava je osigurati zaštitu kodiranih poruka od pogrešaka koje nastaju prilikom prijenosa. Zaštita kodiranih poruka vrši se upotrebom zaštitnog kodiranja. Zaštitno kodiranje je postupak dodjeljivanja kodnih riječi porukama iz kodera informacije. Navedeni postupak provodi se u koderu kanala (Slika 0.1). Kodne riječi se sastoje od niza simbola iz abecede koja je jednaka ili različita od abecede simbola od kojih je sačinjena kodirana poruka. Pravila dodjele kodnih riječi kodnim porukama čine kôd, pa kažemo da se zaštitno kodiranje provodi zaštitnim kodom. Dekodiranje prenijetih kodnih riječi na prijemnoj strani odvija se u dekoderu kanala.

Princip zaštitnog kodiranja poruka sličan je mehanizmu koji već postoji u jeziku. Hrvatski jezik je savršen primjer budući da postoji nedvosmisleno preslikavanje između izgovorene i pisane riječi (fonetičnost). Prosječna duljina riječi je 5 slova (simbola), a abeceda koju upotrebljavamo ima 30 slova. Tako možemo procijeniti da bi u hrvatskom jeziku moglo biti oko $30^5 \approx 24.3$ milijuna riječi. Međutim, svjesni smo da je realan broj riječi daleko manji. To omogućuje da se u velikom broju slučajeva susjedne riječi razlikuju ne samo po duljini, nego i po velikom broju različitih slova. Na primjer, riječi **PRODOR** i **PRIZOR** su jednake po duljini i vrlo slične po redoslijedu i upotrebi simbola (slova). Međutim, čak i u tom slučaju razlika između ovih riječi je u 2 simbola. To nam omogućuje da prilikom razgovora u prostoru s velikom razinom buke (smetnje pri prijenosu) ipak razaznamo o kojoj se riječi radi na način da izaberemo postojeću riječ koja je najsličnija riječi koju smo zapravo čuli.

Prosječna duljina riječi koje upotrebljavamo u jeziku je puno veća od prosječne duljine zamišljenog optimalnog koda jezika (njegove entropije). Međutim, jezik osigurava zaštitu od

smetnji pri komunikaciji: npr. buke, prekida u komunikaciji ili npr. razmazanih ili izgubljenih slova u pismu.

Jednak princip se koristi i kod zaštitnog kodiranja. Kodirane poruke se kodiraju kodnim riječima koje imaju veću prosječnu duljinu od prosječne duljine poruka, što omogućuje da dekoder na odredištu otkrije pogrešku nastalu prilikom prijenosa. Cilj je iskoristiti onaj zaštitni kôd koji uvodi najmanje moguće povećanje prosječne duljine kodnih riječi u odnosu na prosječnu duljinu kodiranih poruka, a u isto vrijeme osigurava prihvatljivo malu vjerojatnost neotkrivanja pogrešaka simbola nastalih pri prijenosu.

Ukoliko dođe do otkrivanja pogreške na primljenoj kodnoj riječi, pokreće se postupak otklanjanja pogreške (engl. *error correction*). Jedan način otklanjanja pogreške je korištenje posebnih svojstava zaštitnog koda kako bi se odredila kodna riječ koja je zapravo poslana. Opisani pristup zovemo **ispravljanje pogreški u dekoderu kanala** (FEC – engl. *forward error correction*). Koristi se prvenstveno onda kada odredište ne može zatražiti ponovno slanje pogrešno primljene kodne riječi (engl. *retransmission*). Zaštitni kodovi koji se koriste u ovom pristupu moraju posjedovati svojstva otkrivanja i ispravljanja pogrešaka i takve kodove zovemo kodovi za ispravljanje pogrešaka (engl. *error correcting code*).

U slučaju kada postoji mehanizam koji odredištu omogućuje da zatraži ponovno slanje krivo primljene kodne riječi, koristi se pristup **ispravljanja pogreški ponovnim slanjem** (BEC – engl. *backward error correction*). U ovom pristupu, kada odredište otkrije pogrešku na primljenoj kodnoj riječi, prvo je pokušava otkloniti koristeći se strukturom samog koda. Ukoliko nije moguće otkloniti pogrešku, ili sam zaštitni kôd ne pruža mogućnost otklanjanja pogreške, odredište traži ponovno slanje kodne riječi sve dok se ne primi ispravna kodna riječ. Zaštitni kodovi koji mogu samo otkriti pogreške zovemo kodovi za otkrivanje pogrešaka (engl. *error detection code*). Zaštitni kodovi za otkrivanje pogrešaka mogu, ali ne moraju imati sposobnost otklanjanja pogrešaka.

Zaštitni kodovi dijele se u dvije osnovne skupine: *blok-kodovi* (engl. *block codes*) i *konvolucijski kodovi* (engl. *convolutional codes*), koji se u literaturi dosta često mogu naći i pod nazivima *stablasti* (engl. *tree*) ili *rešetkasti* (engl. *trellis*) kodovi. Glavne razlike među navedenim skupinama kodova su u načinu izvedbe kodera. Kod blok-kodova se *k*-bitni ulazni blok bitova potpuno preslikava u *n*-bitnu izlaznu *kodnu riječ* (engl. *code word*). Drugačije rečeno, koder za blok-kodove spada u grupu kodera *bez memorije* (engl. *memoryless*) iz razloga što je generiranje nekog bita u kodiranoj poruci isključivo funkcija trenutačnog stanja ulaza kodera. Konvolucijski kodovi spadaju u grupu memorijskih kodova. Kod njih je generiranje nekog bita u kodiranoj poruci funkcija trenutačnog stanja ulaza kodera kao i nekolicine njegovih prethodnih stanja.

Također, postoji i druga podjela zaštitnih kodova, a to je na *linearne* (engl. *linear*) i *nelinearne* (engl. *nonlinear*) kodove. Navedena podjela napravljena je na osnovu strukture i svojstava kodnih riječi. Linearni kodovi su oni kodovi čija se svaka kodna riječ može izraziti kao linearna kombinacija drugih kodnih riječi. Njihova primjena je jako rasprostranjena. Postoje tri kategorije linearnih kodova, i to: blok, konvolucijski i turbo kodovi. Njihova će svojstva biti analizirana u sljedećim poglavljima.

1.1. Uvod u blok-kodove

Zaštitni kôd se sastoji od određenog broja kodnih riječi i pravila dodjeljivanja kodnih riječi porukama koje se kodiraju. Broj simbola u kodnoj riječi se zove duljina kodne riječi. Kodne riječi zaštitnog koda se formiraju na način da se simbolima originalne kodirane poruke dodaje jedan ili više redundantnih simbola ili se svakoj kodiranoj poruci pridružuje zasebna kodna riječ koja ima

veći broj simbola od originalne poruke. U binarnim digitalnim komunikacijskim sustavima kodne riječi i kodirane poruke su nizovi jedinica i nula. Na primjer, kodirana poruka može biti 10110. Zaštitni koder na tu kodiranu poruku dodaje jedan zaštitni bit tako da zbroj jedinica bude paran - 101101. Dobivena kodna riječ omogućuje otkrivanje jednostruke pogreške simbola/bita. Ovo je primjer paritetnog kodiranja koje će biti posebno obrađeno u nastavku poglavlja.

Općenito se može govoriti o *q*-narnom kodu, čije se kodne riječi sastoje od simbola izabranih iz konačnog skupa simbola $F_q = \{\lambda_1, \ldots, \lambda_q\}$ s q elemenata kojeg nazivamo abeceda (engl. *alphabet*). Najčešće se abeceda izabire kao podskup skupa cijelih brojeva, a broj elemenata u abecedi se izabire kao prost broj ili potencija prostog broja. U digitalnim komunikacijskim sustavima koristi se abeceda $F_2 = \{0, 1\}$. Simboli abecede F_2 su *binarne znamenke* 0 i 1, a kodovi koji koriste ovu abecedu zovu se *binarni kodovi*. Nadalje **proučavamo isključivo binarne zaštitne kodove**.

Primjer: zaštitno kodiranje

PRIMJER

Princip zaštitnog kodiranja vidljiv je iz sljedećeg primjera. Izvor informacije generira četiri različite informacijske poruke: A, B, C i D, a koder informacije kodira ih binarnim kodom na sljedeći način:

$$P = \begin{cases} 0 & 0 & - & A; \\ 0 & 1 & - & B; \\ 1 & 0 & - & C; \\ 1 & 1 & - & D. \end{cases}$$

Koder kanala zaštitno kodira ove kodirane poruke dodajući jedan redundantni simbol na sljedeći način:

$$K_1 = \begin{cases} 0 & 0 & 0 & - & A; \\ 0 & 1 & 1 & - & B; \\ 1 & 0 & 1 & - & C; \\ 1 & 1 & 0 & - & D. \end{cases}$$

Dobiven je zaštitni kôd s četiri kodne riječi i pravilom njihovog pridruživanja kodiranim porukama. Može se uočiti da ukoliko prilikom prijenosa dođe do pogreške na jednom simbolu kodne riječi, primljena kodna riječ neće pripadati zaštitnom kodu pa prijemnik može otkriti da je došlo do pogreške. Kôd K_1 je kôd za otkrivanje jednostruke pogreške.

Neki drugi koder može dodati tri redundantna simbola. Na primjer, zaštitni kôd može biti sljedeći:

$$K_2 = \begin{cases} 0 & 0 & 0 & 0 & 0 & - & A; \\ 0 & 1 & 1 & 0 & 1 & - & B; \\ 1 & 0 & 1 & 1 & 0 & - & C; \\ 1 & 1 & 0 & 1 & 1 & - & D. \end{cases}$$

Kôd K_2 ne samo da može otkriti jednostruku pogrešku, nego je može i ispraviti. Dodatno, kôd K_2 može otkriti i dvostruku pogrešku. Ispravljanje pogreške se vrši na način da prijemnik odredi onu kodnu riječ koja se najmanje razlikuje od primljene. Na primjer, ako je primljena kodna riječ 01100, najbliža ispravna kodna riječ je 01101, pod pretpostavkom da je nastupila jednostruka pogreška na zadnjem simbolu (binarnoj znamenci).

Zaštitni kodovi K_1 i K_2 iz prethodnog primjera imaju svojstvo da su im sve kodne riječi jednake duljine. Takvi kodovi se nazivaju blok-kodovi. Naglasimo ovaj pojam sljedećom definicijom.

DEFINICIJA

BLOK-KÔD: Kôd K zove se **blok-kôd** ukoliko su duljine svih njegovih kodnih riječi jednake. Ako kodne riječi koda K imaju duljinu n, onda je K **blok-kôd duljine** n.

DEFINICIJA

1.1.1. Hammingova distanca

Iz prethodnog primjera vidimo da je cilj koristiti zaštitni kôd koji osigurava što veću razliku između svih kodnih riječi. Za mjeru različitosti kodnih riječi uzima se Hammingova distanca [1].

HAMMINGOVA DISTANCA: Hammingova distanca između dvije kodne riječi je broj pozicija na kojima se kodne riječi razlikuju, tj. broj pozicija na kojima kodne riječi imaju različite simbole. Oznaka Hammingove distance između dviju kodnih riječi **x** i **y** je $d(\mathbf{x}, \mathbf{y})$. Hammingova distanca se definira samo za kodne riječi jednakih duljina.

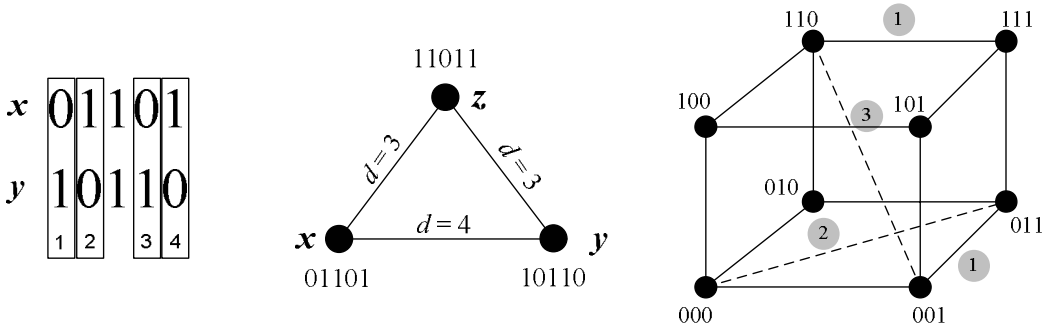
Na primjer, Hammingova distanca kodnih riječi 01101 i 10110 je d(01101, 10110) = 4.

Kodne riječi često prikazujemo vektorima simbola. Na primjer, kodna riječ 10110 koda K_2 iz prethodnog primjera se može prikazati kao vektor redak s oznakom **x** = [10110]. Članovi ovog vektora su skalari, 0 i 1 i pripadaju abecedi simbola. Na taj način poistovjećujemo kodnu riječ i vektor, pa je i sam blok-kôd duljine n zapravo podskup skupa svih mogućih vektora tipa $\mathbf{x} = [\, x_1, x_2, \ldots, x_n \,]$, gdje su x_i skalari iz abecede $F_2=\{0,1\}$.

Za kodne riječi $\mathbf{x}, \mathbf{y}, \mathbf{z}$ blok-koda K, Hammingova distanca ima sljedeća svojstva:

- $d(\mathbf{x}, \mathbf{y}) = 0$ ako i samo ako je $\mathbf{x} = \mathbf{y}$;
- $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ za sve $\mathbf{x}, \mathbf{y} \in K$;
- $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{z}, \mathbf{y})$ za sve $\mathbf{x}, \mathbf{y}, \mathbf{z} \in K$ (nejednakost trokuta).

Navedena svojstva Hammingove distance ilustrirana su na slici (Slika 0.2).



Slika 0.2: Grafičke interpretacije svojstava Hammingove distance

Hammingova distanca je važan parametar kada se govori o sposobnosti koda da otkrije ili ispravi pogrešku. Problem s kojim se suočava primatelj kada primi kodnu riječ koja nije dio dogovorenog zaštitnog koda je otkriti o kojoj se kodnoj riječi zapravo radi. Jedna od strategija koja se najčešće primjenjuje je **dekodiranje najbližim susjedom**. U binarnim simetričnim kanalima u kojima je vjerojatnost pogreške bita jednaka za 1 i 0, takva strategija rezultira **dekodiranjem s najvećom vjerojatnošću**. Ukoliko kôd ima sposobnost otklanjanja pogreške bita, odabire se ona rezultantna kodna riječ koja ima najmanju Hammingovu distancu od

primljene kodne riječi. Drugim riječima, ukoliko je primljena kodna riječ **x**, izabire se kodna riječ $\mathbf{x}' \in K$ takva da je distanca $d(\mathbf{x}, \mathbf{x}')$ minimalna. Sposobnost koda da otkrije ili ispravi pogrešku ovisi o najmanjoj Hammingovoj distanci svih parova kodnih riječi nekog koda *K*. Tu najmanju distancu nazivamo *distanca koda*.

DEFINICIJA

DISTANCA KODA: Distanca koda *K*, s oznakom *d*(*K*), je najmanja Hammingova distanca svih parova kodnih riječi koda *K*, tj.

$$d(K) = \min_{\mathbf{x},\mathbf{y}\in K} \left(d(\mathbf{x},\mathbf{y}) \,\middle|\, \mathbf{x} \neq \mathbf{y} \right).$$

OZNAKA BLOK-KODA: Blok-kôd duljine *n* s *M* kodnih riječi i distancom *d* ima oznaku (n, M, d).

Distanca koda $d(K)$ određuje sposobnost otkrivanja i ispravljanja pogreške simbola.

TEOREM

SPOSOBNOST KODA DA OTKRIJE I ISPRAVI POGREŠKU: Ako zaštitni kôd *K* ima distancu *d*(*K*) i ako se dekodiranje provodi principom najbližeg susjeda, onda vrijedi sljedeće:

- Kôd *K* može otkriti najviše $d(K)-1$ pogreški u jednoj kodnoj riječi, tj. ako je najveći broj pogreški koje kôd može otkriti *s*, onda mora biti zadovoljen izraz $d(K) \geq s+1$.
- Kôd K može ispraviti najviše $\lfloor (d(K)-1)/2 \rfloor$ pogreški u jednoj kodnoj riječi, gdje je $\lfloor x \rfloor$ oznaka za najveći cijeli broj manji od *x*. Drugim riječima, ukoliko se s *t* označi najveći broj pogreški koje kôd *K* može ispraviti u jednoj kodnoj riječi, onda mora biti zadovoljen izraz $d(K) \geq 2t+1$.

Prvo svojstvo je jednostavno za shvatiti. Ukoliko je jednu kodnu riječ nekog koda moguće pretvoriti u neku drugu kodnu riječ tek izmjenom $s+1$ simbola, onda će bilo koja izmjena od *s* ili manje simbola zasigurno dati riječ koja nije dio koda. Dekoder će takvu riječ smatrati pogrešnom i pogreška će biti otkrivena.

Drugo svojstvo izgleda netrivijalno. Da bismo ga shvatili, uvest ćemo pojam sfere, koji će nam poslužiti i kasnije u tekstu.

DEFINICIJA

SFERA KODNE RIJEČI: Neka je **x** bilo koja kodna riječ (vektor) duljine *n* koda *K*. Kodna riječ predstavljena vektorom **x** pripada skupu svih mogućih vektora duljine *n*. Neka je *r* cijeli nenegativan broj, $r \geq 0$. Sfera radijusa *r* sa središtem **x** i oznakom *S*(**x**, *r*) je skup vektora koji imaju Hammingovu distancu do vektora **x** manju ili jednaku *r*, tj.

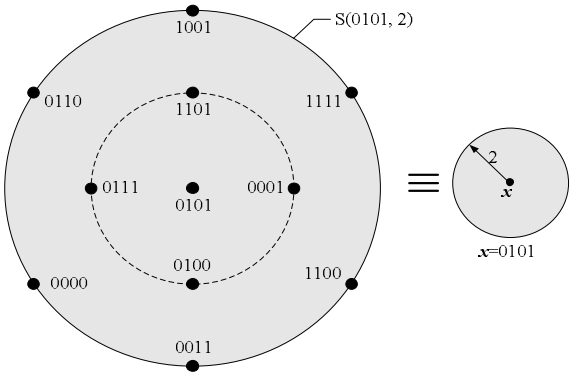
$$S(\mathbf{x}, r) = \left\{ \mathbf{y} \in (F_2)^n \mid d(\mathbf{x}, \mathbf{y} \leq r) \right\},$$

gdje je $(F_2)^n$ oznaka za skup svih binarnih kodnih riječi duljine *n* (vektorski prostor).

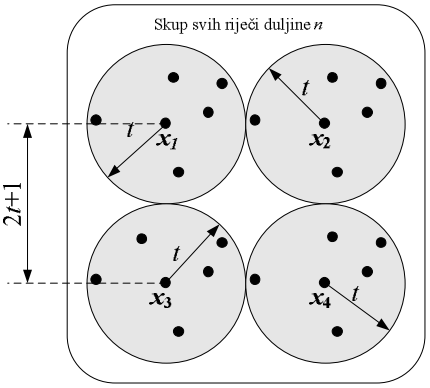
Dakle, sfera $S(\mathbf{x}, r)$ je skup svih vektora koji su od **x** po Hammingovoj distanci udaljeni 0,1,…,*r*. Korisno je primijetiti da je broj vektora koji do vektora **x** imaju udaljenost *m* točno $\binom{n}{m}$. Na primjer, neka je zadana kodna riječ 0101 nekog binarnog koda. Sfera *S*(0101, 2) je skup sljedećih kodnih riječi:

$$0101 \rightarrow \begin{cases} & & 1001 \\ & 1101 & 1111 \\ 0101 & 0001 & 1100 \\ & 0111 & 0011 \\ & 0100 & 0000 \\ & & 0110 \\ d=0 & d=1 & d=2 \end{cases}.$$

Sferu vizualno možemo predočiti kako je prikazano slikom (Slika 0.3).



Slika 0.3: Sfera radijusa 2 oko kodne riječi 0101



Slika 0.4: Položaj sfera zamišljenog koda s četiri kodne riječi

Vratimo se sada drugoj točki svojstva. Zamislimo kôd s četiri kodne riječi i Hammingovom distancom između svih kodnih riječi $2t+1$ ili većom (Slika 0.4). Dekoder dekodira kodne riječi principom najbližeg susjeda, tako da će sve primljene riječi unutar sfere $S(\mathbf{x}_i, t)$ biti (ispravno) dekodirane kao $\mathbf{x}_i$. Međutim, da bi neka poslana kodna riječ $\mathbf{x}_i$ bila ispravno primljena, ona u prijenosu ne smije biti toliko izmijenjena da izađe izvan svoje sfere. Kodna riječ također ne smije prijeći u sferu neke druge kodne riječi. Stoga, da bi kodna riječ bila ispravno dekodirana, najviše t njenih simbola smije biti promijenjeno zbog smetnji pri prijenosu. Upravo je to izreka druge točke svojstva o otkrivanju i ispravljanju pogreški.

1.1.2. Najveći ostvarivi broj kodnih riječi i perfektni kodovi

Za svaki binarni blok-kôd vrijedi da je broj kodnih riječi M uvijek manji ili jednak 2^n, gdje je n duljina koda. Osnovni zadatak teorije kodiranja je pronaći onaj kôd K duljine n koji bi za zadanu najmanju Hammingovu distancu $d(K)$ imao najveći broj kodnih riječi M**.** Najveći ostvarivi M nekog binarnog koda K s najmanjom Hammingovom distancom d označit ćemo s $A(n,d)$.

Ukoliko se zahtijeva najmanja Hammingova distanca d=1, onda je rezultat trivijalan: $A(n, d) = 2^n$. To slijedi iz činjenice da se od koda jedino zahtijeva da kodne riječi budu različite, pa kôd sačinjavaju svi mogući vektori sa skalarima 0 i 1 duljine n, a takvih je 2^n.

Bez velikih poteškoća se dolazi i do rezultata za $A(n, n)$. Budući da je $d = n$, to znači da se sve kodne riječi razlikuju u svim pozicijama unutar kodne riječi. Odaberemo li bilo koju poziciju (na primjer prvu), onda se sve kodne riječi moraju razlikovati barem u toj poziciji. Budući da je kod

binarnih abeceda na tu poziciju mogući postaviti samo 2 različita simbola, slijedi da je najveći mogući broj različitih kodnih riječi $A(n, n) = 2$. Na primjer, to može biti kôd s ponavljanjem: 000…0, 111…1.

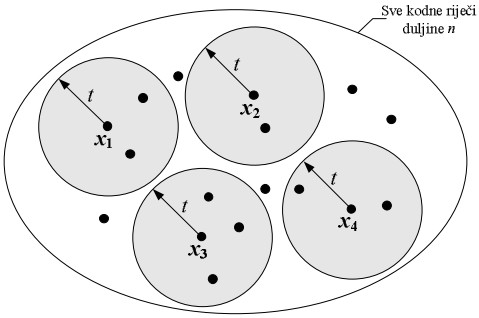
Tablica 0.1: Danas poznate vrijednosti $A(n, d)$ nekih binarnih kodova

n	$d = 3$	$d = 5$	$d = 7$
5	4	2	-
6	8	2	-
7	16	2	2
8	20	4	2
9	40	6	2
10	72-79	12	2
11	144-158	24	4
12	256	32	4
13	512	64	8
14	1024	128	16
15	2048	256	32
16	2560-3276	256-340	36-37

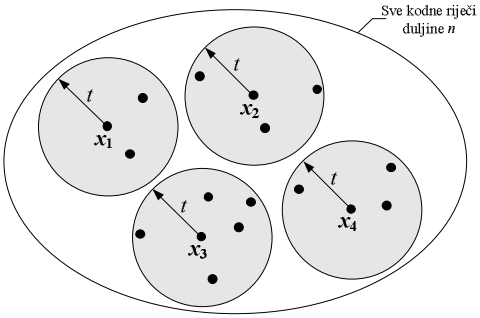
Do ostalih rezultata dolazi se složenim postupcima, a neke vrijednosti nisu niti poznate. Danas poznate vrijednosti ukratko su prikazane u tablici (Tablica 0.1) [2]. Međutim, postoje određene zakonitosti koje pomažu prilikom određivanja najvećeg M.

Neka $\#S(\mathbf{x}, t)$ označava broj kodnih riječi u sferi $S(\mathbf{x}, t)$ vektora $\mathbf{x}$ duljine n. Budući da je broj kodnih riječi koji do vektora $\mathbf{x}$ imaju Hammingovu distancu m jednak $\binom{n}{m}$, slijedi da je

$$\#S(\mathbf{x},t) = 1+\binom{n}{1}+\binom{n}{2}+\ldots+\binom{n}{t}.$$



Slika 0.5: Hammingovo ograničenje



Slika 0.6: Perfektan kod

Neka je distanca koda $d = 2t+1$. U tom slučaju, oko svake od M kodnih riječi postoji sfera radijusa t, te zasigurno vrijedi $M \cdot \#S(\mathbf{x},t) \le 2^n$, gdje je 2^n ukupan broj mogućih binarnih vektora duljine n.

Ovu tvrdnju moguće je shvatiti ukoliko se pogleda slika (Slika 0.5). Točkama su prikazani svi mogući vektori duljine n, a krugovima su označene sfere oko kodnih riječi koda. Iz ovoga se jednostavno može izračunati gornja granica na M, a to je upravo i iskaz sljedećeg svojstva o Hammingovom ograničenju.

TEOREM

HAMMINGOVO OGRANIČENJE: Za binarni kôd $(n, M, 2t+1)$, ukoliko postoji, mora biti zadovoljen izraz

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

Ovo ograničenje broja *M* se zove i ograničenje sfernog pakiranja (engl. *sphere-packing bound*).

U skladu s ovim svojstvom uveden je pojam perfektnog koda.

DEFINICIJA

PERFEKTAN KOD: Binarni kôd $(n, M, 2t+1)$ koji zadovoljava izraz:

$$M = \frac{2^n}{\binom{n}{0}+\binom{n}{1}+\binom{n}{2}+\ldots\binom{n}{t}}$$

naziva se **perfektan kôd**.

Perfektan kôd je onaj kôd distance $2t + 1$ za kojeg vrijedi da su sve moguće kodne riječi u jednoj od sfera radijusa t (Slika 0.6). Ovo svojstvo je vrlo korisno kada dođe do dekodiranja. Koja god kodna riječ dođe u dekoder kanala, dekoder će uspjeti naći originalnu kodnu riječ kojoj pripada.

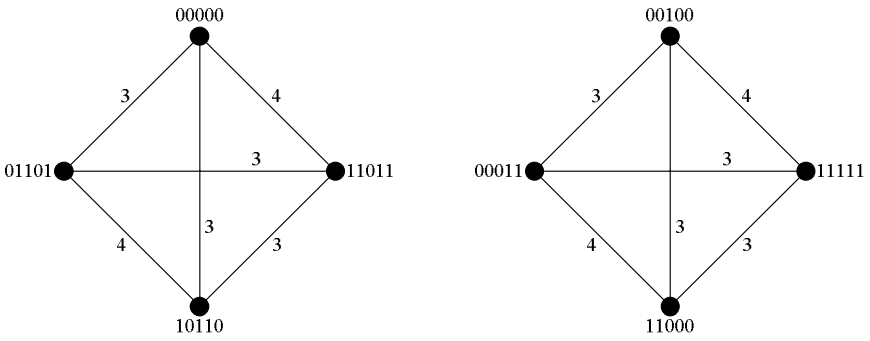
Na primjer, binarni kôd $(2t+1, 2, 2t+1)$ s kodnim riječima 000…0 i 111…1 je trivijalan primjer perfektnog koda. Međutim, postoje i primjeri netrivijalnih kodova poput binarnog Golay $(23, 12, 7)$ ili ternarnog Golay $(11, 6, 5)$ koda [2], [3]. Pronalaženje perfektnih kodova je u centru pažnje moderne teorije kodiranja.

1.1.3. Ekvivalencija kodova

Važno je uočiti da kôd prvenstveno određuju relativni odnosi između pozicija različitih kodnih riječi. Na primjer, razmotrimo sljedeća dva koda:

$$K_1 = \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{Bmatrix} \qquad K_2 = \begin{Bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{Bmatrix}$$

Iako izgledaju potpuno različiti, između njih postoji određena funkcijska ovisnost. Kôd K_2 je dobiven na način da su na trećoj poziciji koda K_1 zamijenjeni simboli 1 s 0 i 0 s 1. Nakon toga su zamijenjene pozicije 2 i 4 svih kodnih riječi. Slika 0.7 prikazuje Hammingove distance kodova.



Slika 0.7: Ekvivalencija kodova

Uočavamo da nema razlike u Hammingovim distancama između pojedinih riječi kodova K_1 i K_2, a time nema niti razlike u sposobnosti ovih kodova da otkriju i isprave eventualnu pogrešku. Općenito se može reći da zamjene pozicija kodnih riječi ne utječu na relativne odnose pojedinih kodnih riječi u pogledu Hammingove distance.

Permutacija binarnih znamenki oblika

$$\begin{pmatrix} 0 & 1 \\ \downarrow & \downarrow \\ 1 & 0 \end{pmatrix}$$

izvršena na istoj poziciji svih kodnih riječi također ne mijenja Hammingove distance između pojedinih kodnih riječi. Stoga, kodovi koji se mogu dobiti jedan iz drugoga primjenom permutacije nad simbolima iste pozicije kodnih riječi i zamjenom pozicija su u pogledu sposobnosti otkrivanja i otklanjanja pogrešaka *ekvivalentni*.

DEFINICIJA

EKVIVALENTNI KODOVI: Dva binarna blok-koda su ekvivalentna ukoliko se jedan iz drugog mogu dobiti

(1) postupkom permutacije nad jednom ili više pozicija koda,

(2) zamjenom dviju ili više pozicija koda prije ili nakon (1).

PRIMJER

Primjer: Permutacija kodnih riječi ekvivalentnih kodova

Promotrimo binarni kôd

$$K = \begin{Bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{Bmatrix}.$$

Ukoliko se na drugu i treću poziciju primijeni permutacija

$$\begin{pmatrix} 0 & 1 \\ \downarrow & \downarrow \\ 1 & 0 \end{pmatrix},$$

te se zamijene pozicije 1 i 4, dobiva se kôd:

$$K' = \begin{Bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{Bmatrix}.$$

Dobiveni kôd *K'* se u odnosu na početni kôd *K* razlikuje samo u dvije kodne riječi, ali posjeduje kodnu riječ **0**. Može se pokazati da je bilo koji kôd operacijama permutacije i zamjene pozicija moguće pretvoriti u ekvivalentan kôd koji u sebi sadrži kodnu riječ **0** = 000…0.

TEOREM

POSTOJANJE KODNE RIJEČI 0: Svaki binarni blok-kôd (n, M, d) je ekvivalentan kodu (n, M, d) koji sadrži kodnu riječ **0** = 000…0.

PRIMJER

Primjer: Određivanje najvećeg broja kodnih riječi

Ovo svojstvo može uvelike pomoći prilikom određivanja najvećeg broja kodnih riječi nekog koda. Za primjer odredimo koliki je najveći *M* binarnog koda (5, *M*, 3), tj. *A*(5, 3). Koristeći se Hammingovim ograničenjem, lako je zaključiti da je $M \leq 5$. Međutim, koristeći se prethodnim svojstvom, može se zaključiti da je u nekom ekvivalentnom kodu s najvećim brojem kodnih riječi zasigurno kodna riječ 00000. Budući da je najmanja Hammingova distanca 3, onda može postojati samo jedna kodna riječ koja ima 4 ili 5 jedinica. To slijedi iz činjenice da ako bi ih bilo dvije, njihova međusobna distanca bi bila manja od 3, jer bi se razlikovale u najviše jednoj ili dvije pozicije.

Nadalje, kodna riječ **1** = 11111 također nije moguća budući da nije moguće formirati niti jednu kodnu riječ koja bi imala Hammingovu distancu do kodnih riječi **0** i **1** od najmanje 3 pozicije.

Nije moguće imati kodne riječi s jednom ili dvije jedinice, budući da bi distanca tih kodnih riječi do kodne riječi **0** bila manja od 3. Dakle, kao kandidati za ostale kodne riječi su samo one koje imaju 3 jedinice. Iako takvih kodnih riječi ima ukupno deset, potrebno je izabrati one koje su međusobno udaljene najmanje za 3 pozicije. Provjerom svih kombinacija moguće je zaključiti da samo dvije riječi zadovoljavaju ovaj kriterij.

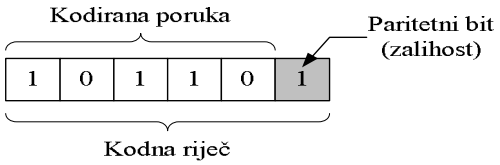
Sada je moguće formirati kôd koji se sastoji od kodne riječi **0**, dvije kodne riječi s 3 jedinice i jedne kodne riječi sa 4 jedinice. To može biti sljedeći konkretan kod:

$$K = \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{Bmatrix}.$$

Još nije dan odgovor na pitanje koliki je $A(5,3)$. Jedino se zna da je $4 \leq A(5,3) \leq 5$. Može se pretpostaviti da postoji kôd (5, 5, 3). Međutim, u tom slučaju bi, koristeći princip ekvivalentnosti kodova, morala postojati treća kodna riječ s 3 jedinice ili pak dvije kodne riječi sa 4 ili 5 jedinica. Budući da niti jedan od ova dva zahtjeva nije moguće ispuniti, zaključak je da kôd $(5,5,3)$ ne postoji i da je $A(5,3) = 4$.

1.1.4. Paritetno kodiranje

Najjednostavnija klasa blok-kodova koja se koriste u praksi su kodovi koji koriste paritetno kodiranje. Ovi kodovi se koriste isključivo za otkrivanje pogreški u primljenoj kodnoj riječi. Ideja paritetnog kodiranja je dodavanje zalihosnog simbola/bita kodiranoj poruci na način da zbroj jedinica u dobivenoj kodnoj riječi bude paran. Na primjer, ukoliko je kodirana poruka 10110, koder kanala na kraj riječi dodaje bit 1 tako da ukupan broj bitova 1 bude paran – Slika 0.8.



Slika 0.8: Primjer horizontalnog paritetnog kodiranja parnim paritetom

Ukoliko prilikom prijenosa kodne riječi prijenosnim kanalom dođe do jednostruke pogreške bita, broj bitova 1 neće biti paran broj i dekoder kanala će otkriti pogrešku prilikom prijenosa.

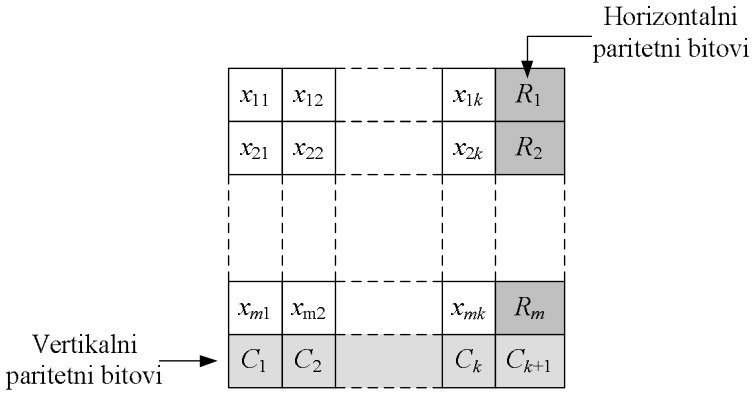
Zalihosni bit koji se dodaje kodiranoj poruci naziva se **paritetni bit** (engl. *redundancy check*). U praksi se koriste dva načina dodavanja paritetnog bita: **parni paritet** (engl. *even parity*) i **neparni paritet** (engl. *odd parity*). Kod parnog pariteta ukupan broj bitova 1 u kodnoj riječi mora biti paran broj, a kod neparnog pariteta neparan broj. Ukoliko s $x_1,\ldots,x_k$ označimo simbole kodirane poruke, onda se paritetni bit R izračunava zbrajanjem aritmetikom modulo 2 na sljedeći način:

$$R = x_1 + x_2 + \cdots + x_k \qquad (\textit{parni paritet}),$$
$$R = x_1 + x_2 + \cdots + x_k + 1 \qquad (\textit{neparni paritet}).$$

U praksi se paritetno kodiranje provodi uvođenjem zajedničkih paritetnih bitova za više uzastopnih kodiranih poruka. S x_{ij} označimo j-ti simbol i-te kodirane poruke - Slika 0.9. Ukoliko koder kanala paritetno kodira m uzastopnih kodnih poruka, onda se nakon m-te kodne poruke formira posebna kodna riječ s bitovima $C_1, \ldots, C_k$, gdje je svaki bit C_i paritetni bit svih m i-tih simbola kodiranih poruka. Ukoliko se koristi parni paritet, onda se bit C_i dobiva na sljedeći način:

$$C_i = x_{1i} + x_{2i} + \cdots + x_{mi},\ i = 1,\ldots,k,$$

gdje se zbrajanje provodi u aritmetici modulo 2. Ovakav pristup umetanju paritetnih bitova se naziva **vertikalna provjera zalihosti** (VRC – engl. *vertical redundancy check*).



Slika 0.9: Horizontalna i vertikalna provjera zalihosti

Ukoliko koder kanala uz vertikalnu provjeru zalihosti vrši i paritetno kodiranje svake od *m* kodiranih poruka posebno, onda se takav pristup naziva **horizontalna ili longitudinalna provjera zalihosti** (LRC – engl. *longitudinal redundancy check*). U tom slučaju vertikalna provjera zalihosti umeće i paritetni bit C_k+1 kao paritetni bit svih horizontalnih paritetnih bitova - Slika 0.9. U slučaju parnog pariteta, vrijednost bita C_k+1 se dobiva izrazom:

$$C_{k+1} = R_1 + R_2 + \cdots + R_m,$$

gdje se zbrajanje provodi u aritmetici modulo 2. Dodavanjem vertikalne provjere zalihosti povećava se vjerojatnost otkrivanja pogreške za otprilike dva do četiri reda veličine u odnosu na korištenje paritetnog bita za svaku kodnu riječ posebno (horizontalna provjera zalihosti).

1.2. Linearni binarni blok-kodovi

Svaku kodnu riječ binarnog blok-koda *K* duljine *n* moguće je predstaviti vektorom oblika $\mathbf{x} = [x_1 \ldots x_n]$, gdje su x_i skalari (simboli) iz abecede $F_2 = \{0, 1\}$. Skraćeno ćemo ovakve vektore nadalje zvati binarnim vektorima. Nad skupom F_2 je moguće definirati operacije zbrajanja i množenja u aritmetici[1] modulo 2 kako je prikazano sljedećom tablicom:

x_1	x_2	$x_1 + x_2$	$x_1 \cdot x_2$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Neutralan element s obzirom na zbrajanje je 0 (0+1=0), a s obzirom na množenje je 1 (1·1=1). Korisno je primijetiti da je u aritmetici modulo 2 zadovoljena jednakost $-1 = 1$, pa je 1 ujedno i suprotan element s obzirom na zbrajanje. Također, element 1 je jedini koji zadovoljava jednakost $1 \cdot 1^{-1} = 1$, pa je tako 1 i suprotan (inverzan) element s obzirom na množenje.

Neka je $V(n)$ skup svih mogućih binarnih vektora duljine *n*. Takvih je 2^n. Nad skupom $V(n)$ je moguće definirati operacije zbrajanja vektora i množenja vektora skalarom na sljedeći način:

$$\mathbf{x} + \mathbf{y} = [x_1, x_2, x_3, ..., x_n] + [y_1, y_2, y_3, ..., y_n] = [x_1 + y_1, x_2 + y_2, x_3 + y_3, ..., x_n + y_n],$$
$$a \cdot \mathbf{x} = a \cdot [x_1, x_2, x_3, ..., x_n] = [a \cdot x_1, a \cdot x_2, a \cdot x_3, ..., a \cdot x_n],$$

gdje su a, x_i, y_i skalari iz F_2, a **x**, **y** vektori iz skupa $V(n)$. Uz ovako definirane operacije, skup $V(n)$ je **vektorski prostor**[2]. Na primjer, zbrajanjem vektora 011010 i 101110 iz $V(6)$ dobivamo kodnu riječ 110100 na način da zbrajamo binarne znamenke na istoj poziciji u aritmetici modulo 2 na sljedeći način:

$$\begin{array}{r} 011010 \\ + \ 101110 \\ \hline 110100 \end{array}.$$

[1] Skup F_2 s elementima {0, 1} i ovako definiranim operacijama množenja i zbrajanja je algebarska struktura Galoisovo polje.

[2] Vektorski prostor je skup vektora nad kojim su definirane operacije zbrajanja vektora i množenja vektora skalarom takve da su zadovoljena svojstva zatvorenosti s obzirom na zbrajanje vektora i množenja sa skalarom, asocijativnosti, komutativnosti, postojanja neutralnog i suprotnog elementa s obzirom na zbrajanje, distributivnosti zbrajanja i množenja te postojanja neutralnog elementa s obzirom na množenje skalarom.

Zbog zbrajanja u aritmetici modulo 2, operacija zbrajanja uvijek daje isti rezultat kao i operacija oduzimanja. Na primjer 011010 + 10111 = 011010 – 10111 = 110100.

Množenje vektora skalarom iz abecede F_2 je trivijalno. Ukoliko se vektor **x** množi skalarom 1, dobiva se ponovno vektor **x**, a ukoliko se množi skalarom 0, dobiva se vektor **0**:

$$1 \cdot \mathbf{x} = \mathbf{x}, \quad 0 \cdot \mathbf{x} = \mathbf{0}.$$

Nadalje podrazumijevamo da sve opisane algebarske operacije vršimo u aritmetici modulo 2.

Binarni blok-kôd se naziva linearan binarni blok-kôd ako se zbrajanjem bilo koje dvije kodne riječi (vektora) opet dobiva neka kodna riječi iz koda ili ako se množenjem bilo koje kodne riječi skalarom iz abecede opet dobiva kodna riječi iz koda. Primjer linearnog binarnog koda je kôd (5, 4, 3) iz prethodnog potpoglavlja. Definiciju linearnog blok-koda moguće je iskazati i formalno na sljedeći način.

DEFINICIJA

LINEARNI BINARNI BLOK-KOD: Neka je blok-kôd K potprostor vektorskog prostora $V(n)$: $K \subseteq V(n)$. Neka su **x** i **y** kodne riječi koda K i neka je $a \in F_2$. Ako je za sve **x**, **y** i a ispunjeno:

- $\mathbf{x} + \mathbf{y} \in K$,
- $a \cdot \mathbf{x} \in K$,

onda je K linearan binarni blok-kôd.

U nastavku teksta linearne binarne blok-kodove zovemo jednostavno linearni blok-kodovi ili linearni kodovi.

Zahtjevi iz definicije linearnog blok-koda osnovna su svojstva vektorskog prostora. Bez dokaza ćemo ustvrditi da je linearni blok-kôd vektorski potprostor od $V(n)$. Posljedica ove činjenice je da svaki linearni blok-kôd nužno mora imati kodnu riječ **0** prema svojstvu vektorskog prostora o postojanju neutralnog elementa za zbrajanje.

Linearni blok-kodovi omogućuju vrlo jednostavan proračun distance koda preko težine kodnih riječi koda.

DEFINICIJA

TEŽINA KODNE RIJEČI: Težina kodne riječi **x** koda K je broj pozicija kodne riječi na kojima se nalazi simbol 1. Oznaka težine kodne riječi **x** je $w(\mathbf{x})$.

Hammingova distanca dviju kodnih riječi **x** i **y** može se odrediti pomoću jednadžbe

$$d(\mathbf{x}, \mathbf{y}) = w(\mathbf{x} - \mathbf{y}).$$

Dobro je primijetiti da je i razlika kodnih riječi **x** - **y** neka kodna riječ koda linearnog blok-koda K. Stoga se traženjem težina svih kodnih riječi koda K ujedno dobivaju sve Hammingove distance između svih kodnih riječi. Iz toga slijedi da je najmanja težina svih kodnih riječi različitih od **0** ujedno i distanca koda.

TEOREM

DISTANCA LINEARNOG BLOK-KODA: Neka je K linearni blok-kôd i neka je $w(K)$ najmanja težina svih kodnih riječi različitih od **0**. Tada je distanca koda

$$d(K) = w(K).$$

1.2.1. Generirajuća matrica

Već smo rekli da je linearni blok-kôd vektorski potprostor vektorskog prostora $V(n)$. Ta činjenica nam omogućuje da unutar linearnog blok-koda zbrajamo i oduzimamo vektore (kodne riječi) i time dobijemo neku kodnu riječ istog koda, množimo vektore skalarima (tj. množimo kodne riječi simbolima abecede), definiramo skalarni produkt dva vektora, definiramo bazu vektorskog potprostora i brojne druge stvari.

Već sama činjenica da linearni blok-kôd ima bazu (skup svih linearno neovisnih riječi koda) znatno nam olakšava posao definiranja nekog koda. Podsjetimo se da se svi vektori nekog potprostora mogu dobiti kao linearna kombinacija vektora baze. Dakle, ako kôd K ima k vektora u bazi $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k$ (k-dimenzionalni vektorski potprostor), onda se svaka riječ koda može zapisati u obliku:

$$\mathbf{x} = a_1 \cdot \mathbf{b}_1 + a_2 \cdot \mathbf{b}_2 + \ldots + a_k \cdot \mathbf{b}_k,$$

gdje su a_i skalari abecede koda. Broj kodnih riječi koje mogu nastati svim linearnim kombinacijama vektora baze ima točno 2^k.

Iz ovog razloga, linearni blok-kôd se ne mora definirati ispisom svih 2^k kodnih riječi, nego jednostavno definicijom k vektora baze. Vektori baze čine tzv. generirajuću matricu koda.

DEFINICIJA

GENERIRAJUĆA MATRICA KODA: Matrica dimenzija $k \times n$ čiji se reci sastoje od vektora baze koda (n, M, d) se zove generirajuća matrica.

Smisao uvođenja generirajuće matrice je skraćenje zapisa linearnog blok-koda, te pojednostavljenje operacija kodiranja i dekodiranja, kako će biti objašnjeno u narednim potpoglavljima.

PRIMJER

Primjer: Binarni kôd K=(5, 4, 3)

Binarni kôd $K = (5, 4, 3)$ ima kodne riječi

$$K = \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{Bmatrix}.$$

Za bazu se mogu uzeti dvije zadnje kodne riječi, budući da su linearno neovisne: $z_1 = 00111$ i $z_2 = 11011$. Sve se kodne riječi koda K mogu dobiti kao linearna kombinacija $a \cdot z_1 + b \cdot z_2$, gdje su a i $b \in \{0, 1\}$. Tako je generirajuća matrica koda K koja ga jedinstveno opisuje

$$\mathbf{G} = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Sada vidimo da se prve dvije kodne riječi koda *K* mogu dobiti kao linearna kombinacija vektora iz generirajuće matrice **G**:

$$[00000] = 0 \cdot [00111] + 0 \cdot [11011],$$
$$[11100] = 1 \cdot [00111] + 1 \cdot [11011].$$

Na taj način je cijeli kôd definiran svojom generirajućom matricom.

Budući da je broj kodnih riječi linearnog blok-koda određen brojem vektora baze (dimenzija koda) k prema jednadžbi $M = 2^k$, uvodi se i posebna oznaka koda.

OZNAKA LINEARNOG BLOK-KODA: Ako je kôd *K* vektorski *k*-dimenzionalni potprostor vektorskog prostora V(*n*), onda kôd *K* ima oznaku [*n, k*]. Ukoliko je poznata distanca koda *d*, onda je oznaka koda [*n, k, d*].

Blok-kôd iz prethodnog primjera je prema ovoj oznaci linearni blok-kôd [5,2], a budući da znamo distancu koda $d(K) = 3$, možemo ga označiti i oznakom [5,2,3]. Budući da je broj kodnih riječi koda iz prethodnog primjera $M=2^2=4$, kôd [5,2,3] je ujedno moguće označiti i oznakom (5,4,3).

1.2.2. Standardni oblik generirajuće matrice

U prethodnom potpoglavlju je rečeno da su dva blok-koda ekvivalentna ukoliko se jedan iz drugog mogu dobiti postupcima zamjene pozicija (stupaca) i permutacije nad određenim pozicijama. Svi ekvivalentni kodovi mogu se dobiti jedni iz drugih operacijama permutacije i zamjene pozicija.

Kod linearnih blok-kodova doslovna primjena ova dva pravila na jedan linearni blok-kôd ne mora nužno dati ekvivalentni kôd koji je ujedno i linearan. Neka je na primjer definiran binarni kôd

$$K = \begin{cases} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{cases}.$$

To je linearan blok-kôd. Izvršimo li permutaciju nad trećom pozicijom: $0 \to 1$, $1 \to 0$, dobivamo ekvivalentni kôd

$$K_e = \begin{cases} 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{cases}.$$

Dobiveni ekvivalentni kôd K_e je ekvivalentan kodu *K*, ali nema kodnu riječi (vektor) **0** i stoga nije linearan.

Postavlja se pitanje koje je operacije dopušteno vršiti nad linearnim blok-kodom, a da ekvivalentan kôd također bude linearan. Budući da su reci generirajuće matrice linearnog koda također kodne riječi, onda se nad recima generirajuće matrice mogu izvršiti operacije zamjene redaka i dodavanje jednog retka drugom retku, a da nova generirajuća matrica daje isti kôd.

Budući da zamjena stupaca generirajuće matrice odgovara zamjeni pozicija kodnih riječi koda, onda se zamjenom stupaca generirajuće matrice dobiva generirajuća matrica nekog ekvivalentnog koda.

Navedenim operacijama nad recima i stupcima generirajuća matrice dobiva se generirajuća matrica nekog ekvivalentnog linearnog blok-koda. Operacija permutacije nad recima nije dopuštena.

TEOREM

GENERIRAJUĆE MATRICE EKVIVALENTNIH LINEARNIH BLOK-KODOVA: Dva ekvivalentna linearna binarna blok-koda [*n, k*], K_1 i K_2, imaju generirajuće matrice $\mathbf{G}_1$ i $\mathbf{G}_2$ koje se jedna iz druge mogu dobiti sljedećim operacijama:

(1) Zamjena redaka;

(2) Dodavanje jednog retka drugom retku;

(3) Zamjena stupaca.

Ova svojstva znatno olakšavaju pronalazak odgovarajućeg ekvivalentnog linearnog blok-koda budući da je navedene operacije potrebno provoditi samo nad generirajućom matricom. Postoji veliki broj mogućih generirajućih matrica, iako sve opisuju kodove koji imaju jednaku sposobnost otkrivanja i ispravljanja pogreške. Nas zanimaju one generirajuće matrice koje su po svojoj strukturi najjednostavnije, a koje jedinstveno definiraju linearni blok-kôd u smislu njegove sposobnosti da otkrije i ispravi pogrešku.

DEFINICIJA

STANDARDNI OBLIK GENERIRAJUĆE MATRICE: Generirajuća matrica **G** nekog koda *K* ima standardni oblik ako ima strukturu

$$\mathbf{G} = \left[\mathbf{I_k} \mid \mathbf{A}\right],$$

gdje je I_k jedinična matrica reda *k*, a $\mathbf{A}$ matrica dimenzija $k \times (n-k)$.

TEOREM

EGZISTENCIJA STANDARDNOG OBLIKA: Svaka generirajuća matrica **G** linearnog blok-koda [*n, k*] može se operacijama zamjene redaka, zamjene stupaca i dodavanja jednog retka drugom retku svesti na standardni oblik $\left[\mathbf{I_k} \mid \mathbf{A}\right]$.

PRIMJER

Primjer: Standardni oblik generirajuće matrice koda K=[5, 2, 3]

U prethodnom primjeru smo definirali linearni blok-kôd [5, 2, 3] preko liste kodnih riječi i generirajuće matrice **G**:

$$K = \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{Bmatrix}, \qquad \mathbf{G} = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Budući da se radi o linearnom blok-kodu, iz prethodnog teorema znamo da zasigurno postoji barem jedan standardni oblik generirajuće matrice. Do jednog od njih možemo doći tako što na

postojećoj matrici G izvršimo operaciju zamjene drugog i trećeg stupca, a potom zamijenimo prvi i drugi redak. Dobivamo novu generirajuću matricu

$$\mathbf{G}^{*} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Ova generirajuća matrica je u standardnom obliku budući da ima strukturu $[\mathbf{I}_2 \,|\, \mathbf{A}]$. Ukoliko matrici G^* zamijenimo treći i četvrti stupac, dobivamo još jednu matricu u standardnom obliku:

$$\mathbf{G}^{**} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

Dobro je primijetiti da standardni oblik generirajuće matrice nije jedinstven. To slijedi iz činjenice da je moguće zamijeniti dva stupca matrice **A** i još uvijek imati standardni oblik. Različite generirajuće matrice u standardnom obliku neće generirati jednake kodove, nego samo ekvivalentne kodove.

1.2.3. Kodiranje linearnim kodovima

Generirajuća matrica je matrica čiji reci odgovaraju bazi vektorskog potprostora prostora $V(n)$. Stoga se bilo koja kodna riječ može dobiti kao linearna kombinacija vektora baze. Ukupan broj kodnih riječi koda $[n, k]$ je 2^k, što ujedno znači da kôd može kodirati 2^k različitih kodiranih poruka kodera informacije. Formiranje zaštitne kodne riječi se može izvršiti na način da bitovi kodirane poruke budu koeficijenti za linearnu kombinaciju vektora baze.

Neka je

$$\mathbf{G} = \begin{bmatrix} \mathbf{r_1} \\ \mathbf{r_2} \\ \vdots \\ \mathbf{r_k} \end{bmatrix}.$$

generirajuća matrica koda K, gdje su $\mathbf{r}_i$ vektori baze. Neka je $\mathbf{m}=[m_1\ m_2\ \ldots\ m_k]$ bilo koja od 2^k poruka. Tada poruci **m** odgovara točno jedna kodna riječ **x** koda K koja je jednaka linearnoj kombinaciji skalara m_i i vektora baze $\mathbf{r}_i$:

$$\mathbf{x} = \sum_{i=1}^{k} m_i \cdot \mathbf{r_i} = \mathbf{m} \cdot \mathbf{G}.$$

Tako se generiranje zaštitne kodne riječi **x** za kodiranu poruku **m** svodi na umnožak vektor retka poruke i generirajuće matrice. Složenost kodiranja bilo koje poruke je $O(n \cdot k)$.

Primjer: Kodiranje pomoću generirajuće matrice

Za primjer možemo uzeti kôd [5,2,3] koji ima sljedeću generirajuću matricu:

$$\mathbf{G} = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix},$$

Koder izvora informacije generira kodiranu poruku 11. U koderu kanala nastaje zaštitna kodna riječ:

PRIMJER

$$[1\ 1] \cdot \begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix} = [1\ 1\ 1\ 0\ 0].$$

Složenost kodiranja može se smanjiti ako se koristi standardni oblik generirajuće matrice koda. U tom slučaju kodiranje poruke **m** se svodi na množenje

$$\mathbf{m} \cdot [\mathbf{I_k} \mid \mathbf{A}] = \{\mathbf{m}, \mathbf{m} \cdot \mathbf{A}\}.$$

Dobivena kodna riječ se sastoji od dva dijela. Prvih k pozicija zauzima sama poruka i taj dio kodne riječi se zove jednostavno poruka. Ostalih $(n-k)$ pozicija je umnožak $\mathbf{m} \cdot \mathbf{A}$. Ovaj dio kodne riječi se zove dio za provjeru. On predstavlja **zalihost** kodne riječi. Složenost kodiranja bilo koje poruke postaje $O(n \cdot k - k^2)$.

Primjer: Kodiranje pomoću generirajuće matrice u standardnom obliku

PRIMJER

Pronađimo generirajuću matricu koda koji je ekvivalentan onom iz prethodnog primjera, a čija je generirajuća matrica u standardnom obliku. To možemo izvesti tako da jednostavno zamijenimo prvi i treći stupac matrice **G** iz prethodnog primjera. Dobivamo generirajuću matricu:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix},$$

gdje su matrice $\mathbf{I}_2$ i $\mathbf{A}$:

$$\mathbf{I}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Neka je poruka 01. Nastaje sljedeća kodna riječ:

$$[0\ 1] \cdot \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix} = [0\ 1\ 1\ 1\ 1].$$

Ova kodna riječ se sastoji od originalne poruke i 3 zalihosna simbola 1:

$$\underbrace{0\ 1}_{poruka}\underbrace{1\ 1\ 1}_{zalihost}.$$

Vidimo da je zalihost kodne riječi moguće izračunati i jednostavnije tako da se na originalnu poruku zdesna doda rezultat umnoška $[0\ 1] \cdot \mathbf{A} = 111$.

1.2.4. Dekodiranje linearnog koda

Promatrajmo ponovno kôd $(5, 4, 3) = [5, 2, 3]$ sa sljedećim kodnim riječima:

$$K = \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{Bmatrix}.$$

Budući da je distanca koda 3, ovaj kôd može otkriti dvostruku pogrešku i ispraviti jednostruku pogrešku korištenjem principa dekodiranja najbližim susjedom.

Dekoder kanala može postupiti na način da za primljenu kodnu riječ pronađe kodnu riječ do koje ima najmanju Hammingovu distancu i nju proglasiti primljenom kodnom riječju. Naravno, ukoliko je najmanja distanca do bilo koje kodne riječi veća od jedan, dekoder ustanovljuje dvostruku ili eventualno višestruku pogrešku. Složenost ovakvog postupka je $O(M)$, gdje je M broj kodnih riječi. Za velike kodove ovaj postupak može zahtijevati veliko opterećenje procesora prijemnika, što rezultira skupom i neekonomičnom izvedbom prijemnog uređaja.

Jedan od postupaka brzog dekodiranja koji se koristi prilikom dekodiranja je sindromsko dekodiranje o kojem se govori u nastavku ovog poglavlja. Za shvaćanje tog načina dekodiranja potrebno je uvesti pojmove vektora pogreške, koseta, standardnog niza i paritetne matrice.

DEFINICIJA

VEKTOR POGREŠKE: Vektor pogreške **e** za poslanu kodnu riječ $\mathbf{x} = [x_1\, x_2 \ldots x_n]$ i primljenu kodnu riječi $\mathbf{y} = [y_1\, y_2 \ldots y_n]$ se definira kao razlika vektora:

$$\mathbf{e} = \mathbf{y} - \mathbf{x} = [e_1\, e_2 \ldots e_n].$$

Promotrimo nadalje sljedeću tablicu:

00000	11100	00111	11011
00001	11101	00110	11010
00010	11110	00101	11001
00100	11000	00011	11111
01000	10100	01111	10011
10000	01100	10111	01011

Ova tablica je formirana na način da se u prvom retku nalaze kodne riječi koda (5,4,3)=[5,2,3]. Svi ostali reci su formirani na način da se sasvim lijevo nalaze vektori pogreške za slučaj kada kôd može ispraviti pogrešku. Ostali članovi redaka su dobiveni na način da je na kodne riječi iz prvog retka dodavan vektor pogreške retka (prvi stupac). Na ovaj način su dobivene sve kodne riječi koje dekoder može primiti u slučaju da je nastupila jednostruka pogreška. Važno je primijetiti da su članovi prvog stupca također kodne riječi koje se mogu primiti uz jednostruku pogrešku ukoliko je poslana kodna riječ 00000. Dobivena tablica formalno se naziva *standardni niz*. Članovi nekog retka predstavljaju jedan koset (engl. *coset*) skupa kodnih riječi *K*. Koset koda *K* je kôd nastao na način da je svim kodnim riječima dodan neki vektor pogreške[3] **e**. Zanimljivo je da svi kosetovi koji mogu nastati na opisani način sadrže potpuno različite kodne riječi.

DEFINICIJA

STANDARDNI NIZ: Standardni niz koda *K* je tablica koja u prvom retku ima kodne riječi koda *K* s tim da je prva kodna riječ **0**. U svim ostalim recima standardnog niza se nalaze kosetovi koda *K* nastali na način da su kodnim riječima koda *K* dodavani svi vektori pogreške **e** koje kôd *K* može otkriti i ispraviti. Prvi stupac standardnog niza sadrži vektore pogreške, koji se još nazivaju i vodećim elementima koseta.

[3]Ova definicija koseta (engl. *coset*) je posebno prilagođena ovom tekstu. Koset S skupa K se definira kao onaj skup vektora koji nastaje zbrajanjem svih vektora skupa K s nekim fiksnim vektorom iz istog vektorskog skupa.

Uzimajući u obzir pretpostavku da nije nastupila trostruka ili veća pogreška nad simbolima poslane kodne riječi, dekoder vrši dekodiranje na sljedeći način. Kada primi kodnu riječ, pokušava je pronaći u standardnom nizu. Ukoliko je nije pronašao u standardnom nizu, otkriva pogrešku, ali je ne može ispraviti. Ukoliko pronađe kodnu riječ u *i*-tom stupcu standardnog niza, ispravlja pogrešku na način da određuje *i*-tu kodnu riječ koda *K* kao primljenu kodnu riječ. Točnost dekodiranja opisanim postupkom slijedi iz činjenice da je primljena riječ **jedina** riječ koja je mogla nastati iz dekodirane kodne riječi uz jednostruku pogrešku bita. Ovaj postupak je temelj sindromskog dekodiranja.

1.2.5. Paritetna matrica

Opisani postupak dekodiranja pomoću standardnog niza zahtijeva pronalazak primljene kodne riječi u tablici koja može imati vrlo velike dimenzije, iz čega zaključujemo da je takav postupak izuzetno procesorski zahtijevan, što rezultira neefikasnom i skupom izvedbom dekodera kanala. Jedan način ubrzavanja postupka dekodiranja je korištenjem paritetne matrice koda. Kako bismo objasnili postupak dekodiranja paritetnom matricom, prethodno moramo definirati pojam dualnog koda.

Kako je ranije pokazano, generirajuća matrica **G** jedinstveno određuje kôd *K*. Kodne riječi su linearne kombinacije vektora baze, tj. linearne kombinacije redaka matrice **G**. Promotrimo linearni kôd s oznakom $K^{\perp}$ čije su sve kodne riječi *ortogonalne*[4] na sve kodne riječi koda *K*. Dakle, skalarni umnožak svih vektora kodnih riječi iz *K* i $K^{\perp}$ jednak je nula. Kôd $K^{\perp}$ se naziva dualni kôd koda *K*.

DUALNI KOD: Neka su **x** vektori koda *K* (**x** $\in$ *K*). Skup svih vektora **y** vektorskog prostora *V*(*n*) koji su ortogonalni na sve **x** $\in$ *K* čini dualni kôd koda *K* i ima oznaku $K^{\perp}$:

$$K^{\perp} = \{\mathbf{y} \in V(n) \mid \forall \mathbf{x} \in K, \mathbf{y} \cdot \mathbf{x} = 0\},$$

gdje je $\mathbf{x} \cdot \mathbf{y}$ skalarni produkt vektora:

$$\mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^{n} x_i \cdot y_i.$$

DEFINICIJA

Primjer: dualni kôd

Promotrimo linearan binarni kôd *K* sa sljedećim kodnim riječima i generirajućom matricom:

$$K = \begin{cases} 00000 \\ 11100 \\ 10111 \\ 01011 \end{cases}, \quad \mathbf{G} = \begin{bmatrix} 10111 \\ 01011 \end{bmatrix}.$$

Dualni kôd koda *K* je kôd $K^{\perp}$ sa sljedećim kodnim riječima:

[4] Ortogonalni su oni vektori čiji je skalarni umnožak 0, npr. [0 0 0 1 1] · [1 1 1 0 0].

$$K^{\perp} = \begin{cases} 00000 & 01110 \\ 10100 & 01101 \\ 11010 & 00011 \\ 11001 & 10111 \end{cases}.$$

Da se uistinu radi o dualnom kodu može se provjeriti tako da se skalarno pomnože svi parovi kodnih riječi kodova K i kôda $K^{\perp}$.

Dualni kôd ima sljedeće korisno svojstvo. Ako je primljena kodna riječ koda K primljena ispravno, onda njen umnožak sa svim riječima dualnog koda $K^{\perp}$ mora biti jednak nuli. Ovo svojstvo može poslužiti za provjeru ispravnosti primljene kodne riječi. Međutim, broj kodnih riječi dualnog koda može biti iznimno velik, pa takvo što postaje nepraktično. Na sreću, dualni kôd je ujedno i linearan blok-kôd te ima svoju generirajuću matricu.

TEOREM

LINEARNOST DUALNOG KODA: Neka je K linearni blok-kôd $[n, k]$. Dualni kôd $K^{\perp}$ koda K je **linearan** blok-kôd $[n, n-k]$.

PRIMJER

Primjer: linearnost dualnog koda

U dualnom kodu iz prethodnog primjera moguće je izabrati tri linearno neovisne kodne riječi koje čine bazu:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

Ova matrica predstavlja **generirajuću matricu dualnog koda** i nadalje je označavamo s **H**. Svi skalarni produkti između svih parova redaka matrica **G** i **H** jednaki su 0, pa zaključujemo da generirajuće matrice koda K i dualnog koda $K^{\perp}$ moraju zadovoljavati jednadžbu

$$\mathbf{G} \cdot \mathbf{H}^{\mathrm{T}} = \mathbf{0}.$$

Dakle, dualni kôd ne samo da je linearan, nego je i njegova dimenzija $n–k$, gdje je k dimenzija koda K. Stoga je za provjeru ispravnosti primljene kodne riječi dovoljno skalarno pomnožiti primljenu kodnu riječ sa svim vektorima generirajuće matrice dualnog koda kojih ima $n–k$. Provjera ispravnosti primljene riječi $\mathbf{x}$ se svodi na provjeru jednakosti:

$$\mathbf{x} \cdot \mathbf{H}^{\mathrm{T}} = [\,0\,0 \ldots 0\,].$$

Uz poznatu generirajuću matricu **H** dualnog koda $K^{\perp}$, provjera ispravnosti primljene riječi svodi sa na jednostavan umnožak vektora i matrice. Izračunavanje matrice **H** se u osnovi svodi na rješavanje jednadžbe

$$\mathbf{G} \cdot \mathbf{H}^{\mathrm{T}} = \mathbf{0},$$

budući da svi vektori baze koda K moraju biti ortogonalni na vektore baze dualnog koda $K^{\perp}$. Nasreću, postoji iznimno jednostavan način računanja matrice **H** kada je matrica **G** definirana u standardnom obliku.

TEOREM

GENERIRAJUĆA MATRICA DUALNOG KODA: Neka je **G** generirajuća matrica linearnog binarnog koda K u standardnom obliku:

$$\mathbf{G} = \left[\mathbf{I}_k \mid \mathbf{A}\right].$$

Generirajuća matrica dualnog koda $K^{\perp}$ zadovoljava jednadžbu $\mathbf{G} \cdot \mathbf{H}^{\mathrm{T}} = \mathbf{0}$ i jednaka je

$$\mathbf{H} = \left[\mathbf{A}^{\mathrm{T}} \mid \mathbf{I}_{n-k}\right].$$

PRIMJER

Primjer: dekodiranje generirajućom matricom dualnog koda – paritetnom matricom

Definiran je kôd $K = (5,4,3) = [5,2,3]$ i odgovarajuća generirajuća matrica **G** u standardnom obliku:

$$K = \begin{cases} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \end{cases} \rightarrow \mathbf{G} = \left[\begin{array}{cc|ccc} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{array}\right].$$

Uočimo matrice $\mathbf{I}_2$ i **A** u matrici **G**:

$$\mathbf{I}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

Koristeći se prethodnim svojstvom zaključujemo da generirajuća matrica dualnog koda mora imati strukturu $\mathbf{H} = \left[\mathbf{A}^{\mathrm{T}} \mid \mathbf{I}_3\right]$:

$$\mathbf{H} = \left[\begin{array}{cc|ccc} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{array}\right].$$

Pretpostavimo da koder informacije šalje kodiranu poruku 11. Množenjem poruke s generirajućom matricom dobivamo zaštitnu kodnu riječ [11]·**G** = [11011]. Primijetimo da se kodna riječ sastoji od originalne poruke i zalihosnih bitova. Množenjem ove kodne riječi s transponiranom matricom **H** dobivamo rezultat:

$$[11011] \cdot \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [0\,0\,0].$$

Dobiven je vektor **0** što potvrđuje da je kodna riječ ispravna.

Primijetimo da matrica **H** svojim jedinicama unutar svakog retka zapravo određuje pozicije unutar ispravne kodne riječi na kojima zbroj vrijednosti unutar aritmetike modulo 2 mora biti jednak 0. Drugim riječima određuje koje parnim paritetom štiti zalihosni bit. Konkretno, prvi

redak matrice **H** definira za zbroj vrijednosti simbola ispravne kodne riječi na pozicijama 1, 2 i 3 mora biti paran. Drugi stupac to definira za pozicije 1 i 4, a treći stupac za pozicije 2 i 5.

Uzmimo sada istu kodnu riječ, ali uvedimo pogrešku na drugom bitu (**e** = [0 1 0 0 0], **y** = [1 0 0 1 1]):

$$[10011]\cdot\begin{bmatrix}1&1&0\\1&0&1\\1&0&0\\0&1&0\\0&0&1\end{bmatrix}=[101].$$

Dobili smo vektor koji nije **0**, što potvrđuje da kodna riječ nije ispravna, tj. otkrivena je pogreška. Zbroj vrijednosti simbola čije pozicije određuju prvi redak (pozicije 1, 2 i 3) i treći redak (pozicije 2 i 5) matrice **H** nije paran broj.

Vidljivo je da zalihosni bitovi (pozicije 3, 4 i 5) zapravo predstavljaju **bitove za provjeru pariteta** za odabrane pozicije unutar originalne kodirane poruke. Zbog ovog svojstva matrica **H** se naziva paritetna matrica.

DEFINICIJA

PARITETNA MATRICA: Neka je **H** generirajuća matrica dualnog koda $K^{\perp}$. Matrica **H** se naziva paritetna matrica (engl. *parity-check matrix*) ili matrica za provjeru pariteta koda *K*. U svakom retku matrice **H** jedinice određuju pozicije unutar ispravne kodne riječi na kojima zbroj vrijednosti simbola mora biti paran broj.

Ukoliko **H** ima strukturu:

$$\mathbf{H} = [\ \mathbf{B}\,|\,\mathbf{I}_{n-k}\ ],$$

gdje je **B** kvadratna matrica, onda je paritetna matrica **H** u standardnom obliku.

1.2.6. Sindromsko dekodiranje

Sindromsko dekodiranje je metoda koja nam omogućuje da složenost dekodiranja jedne kodne riječi svedemo na razinu $O(1)$. Temelj sindromskog dekodiranja je vektor koji se dobije množenjem primljene kodne riječi i transponirane paritetne matrice.

DEFINICIJA

SINDROM: Sindrom primljene kodne riječi **y** koda *K* s paritetnom matricom **H** je vektor dobiven umnoškom:

$$S(\mathbf{y}) = \mathbf{y}\cdot\mathbf{H}^{\mathrm{T}}.$$

Dva sindroma $S(\mathbf{y})$ smo već izračunali u prethodnom primjeru. Ukoliko primljena kodna riječ **y** pripada kodu *K*, sindrom je vektor **0**. Primijetimo da ukoliko je kôd $K=[n,k]$, onda je sindrom vektor redak dimenzije *n*–*k*. Sindrom $S(\mathbf{y})$ ima jedno vrlo važno svojstvo. Promotrimo opet kôd $[5,2,3]$. Generirajuća i paritetna matrica ovog koda su:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix}, \quad \mathbf{H}^{\mathrm{T}} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Napišimo ponovno standardni niz ovog koda za jednostruke pogreške simbola:

e				*Sindrom*
00000	11100	00111	11011	000
00001	11101	00110	11010	001
00010	11110	00101	11001	010
00100	11000	00011	11111	100
01000	10100	01111	10011	101
10000	01100	10111	01011	110

Možemo uočiti neobično svojstvo sindroma da sve kodne riječi jednog koseta (retka standardnog niza) imaju isti sindrom. To se može provjeriti množenjem za gornji standardni niz. Standardni niz je proširen stupcem u kojem su zapisani sindromi pojedinih redaka/koseta. Ovo svojstvo sindroma je ključ brzog dekodiranja linearnih blok-kodova.

TEOREM

SINDROM KOSETA: *Svi kodne riječi (vektori) koji pripadaju istom kosetu koda K imaju isti sindrom.*

Korisna posljedica ovog svojstva je da je za određivanje sindroma jednog retka standardnog niza dovoljno izračunati sindrom vodećeg člana retka - vektora pogreške **e**. Tako dolazimo do zaključka da postoji preslikavanje jedan-na-jedan između vektora pogreške i sindroma. Moguće je napisati sljedeću **tablicu preslikavanja** za jednostruke pogreške:

e	00000	00001	00010	00100	01000	10000
$S(\mathbf{y})$	000	001	010	100	101	110

Dekodiranje sada postaje vrlo jednostavno i praktično za programsku realizaciju. Postupak se može opisati ovako:

- izračunaj sindrom $S(\mathbf{y})$ primljene kodne riječi **y**,
- iz tablice preslikavanja odredi vektor pogreške **e**,
- poslana kodna riječ je $\mathbf{x} = \mathbf{y} - \mathbf{e}$.

Opisani postupak ima određenih nedostataka. Što ako je primljena kodna riječ **y** koja ima dvostruku pogrešku? Na primjer, sve kodne riječi koje imaju vektor pogreške $\mathbf{e} = 00011$ ili $\mathbf{e} = 10001$ imaju sindrome koji ne pripadaju tablici preslikavanja:

$$[0\,0\,0\,1\,1]\cdot\mathbf{H}^{\mathrm{T}} = [0\,1\,1], \quad [1\,0\,0\,0\,1]\cdot\mathbf{H}^{\mathrm{T}} = [1\,1\,1].$$

Ukoliko sindrom primljene kodne riječi ne postoji u tablici preslikavanja, onda dekoder kanala zaključuje da ta kodna riječ ima dvostruku pogrešku. Međutim, neke dvostruke pogreške mogu uzrokovati potpuno krivu interpretaciju kod dekodera kanala. Na primjer, neka je poslana kodna riječ $\mathbf{x} = 10110$ i neka se dogodila pogreška na pozicijama 2 i 3. Primljena je kodna riječ $\mathbf{y} = 11010$, a vektor pogreške je $\mathbf{e} = 01100$. Dekoder izračunava sindrom

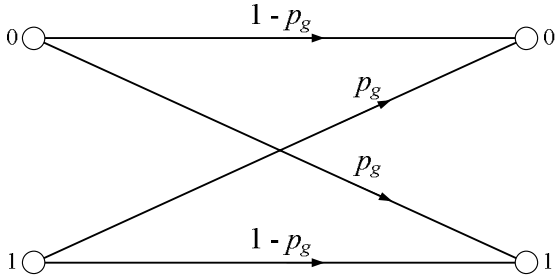
$$S(\mathbf{y}) = [\,1\,1\,0\,1\,0\,] \cdot \mathbf{H}^{\mathrm{T}} = [\,0\,0\,1\,],$$

pogleda u tablicu preslikavanja i pogrešno zaključuje da je vektor pogreške 00001. Uzrok ovakvom ponašanju nije u samom sindromskom dekodiranju, nego u principu dekodiranja najbližim susjedom. Kôd je ispravno otkrio dvostruku pogrešku, ali dekoder nikako ne može razlučiti između jednostruke i dvostruke pogreške. Ovaj problem se u praksi rješava dodavanjem još jednog paritetnog bita - povećanjem distance koda za 1.

1.2.7. Vjerojatnost ispravnog dekodiranja i kapacitet kanala

Problem određivanja vjerojatnosti ispravnog dekodiranja kodne riječi razmatrat ćemo na primjeru binarnog simetričnog kanala prikazanog na slici (Slika 0.10).

Simetrični kanal ima vjerojatnosti pogrešnog prijenosa (inverzije) simbola $p_g < ½$ za oba simbola. Za simetrični kanal također vrijedi da su događaji pogrešnog prijenosa simbolâ iste kodne riječi u cijelosti neovisni slučajni događaji. To nam omogućuje da jednostavno izračunamo vjerojatnost pojave pogreške na *k* specificiranih pozicija simbola kodne riječi.



Slika 0.10: Binarni simetrični kanal s vjerojatnošću pogreške simbola (bita) p_g

Promatrajmo slučajni događaj da je točno *k* unaprijed određenih pozicija simbola neke kodne riječi duljine *n* pogrešno preneseno, a preostalih *n* – *k* ispravno preneseno. Zbog neovisnosti događaja koji ga čine, vjerojatnost takvog događaja računamo kao umnožak *k* vjerojatnosti neispravnog prijenosa simbola i *n* – *k* vjerojatnosti ispravnog prijenosa simbola, što daje vjerojatnost $p_g^k \left(1 - p_g\right)^{n-k}$.

Očigledno je da događaj pojave pogreški nad određenim simbolima kodne riječi odgovara događaju pojave odgovarajućeg vektora pogreške **e**. Stoga se u nastavku koriste isključivo vjerojatnosti pojave vektora pogreške *P*(**e**).

Uočimo da dobivena vjerojatnost $p_g^k \left(1 - p_g\right)^{n-k}$ predstavlja vjerojatnost pojave bilo kojeg vektora pogreške s *k* jedinica. Na primjer, za kôd [5, 2, 3] vrijedi sljedeće:

$$P(00001) = P(00010) = P(00100) = P(01000) = P(10000) = p_g \cdot (1 - p_g)^4 .$$

Jasno, vjerojatnost pojave *k*-struke pogreške simbola, bez poznavanja točne pozicije pogreški, dobiva se zbrajanjem svih vjerojatnosti konkretnih vektora pogreški s *k* jedinica. Na primjer, za

prethodni primjer koda [5, 2, 3] postoji 5 različitih realizacija vektora pogreške **e** s jednostrukom pogreškom, i svaka ima istu vjerojatnost. Stoga je vjerojatnost jednostruke pogreške simbola

$$P\left(|\mathbf{e}|=1\right)=5p_g\cdot\left(1-p_g\right)^4.$$

Postavlja se pitanje kolika je vjerojatnost da je primljena kodna riječ nakon dekodiranja sindromskim dekodiranjem ispravna? Jasno, kodna riječ će biti ispravno dekodirana isključivo ako je vektor pogreške dio standardnog niza. Stoga je vjerojatnost ispravnog dekodiranja jednaka zbroju vjerojatnosti pojave vektora pogreške koji su dio standardnog niza.

TEOREM

VJEROJATNOST ISPRAVNOG DEKODIRANJA: Neka je N_i broj vektora pogreške s i jedinica koji pripadaju standardnom nizu blok-koda K duljine n. Onda je vjerojatnost $P(K)$ da će riječ dobivena dekodiranjem **pomoću standardnog niza** biti jednaka poslanoj kodnoj riječi jednaka:

$$P(K)=\sum_{i=0}^{n}N_i\,p_g{}^i\left(1-p_g\right)^{n-i}.$$

Demonstrirajmo ovaj zaključak sljedećim primjerom.

PRIMJER

Primjer: vjerojatnost ispravnog dekodiranja za kôd [5, 2, 3]:

Za određivanje vjerojatnosti ispravnog dekodiranja za kôd [5, 2, 3] je dovoljno ponovno promotriti njegov standardni niz (vidi 1.2.6). Vidimo da je $N_0 = 1$, $N_1 = 5$, $N_2 = 0$, $N_3 = 0$, $N_4 = 0$ i $N_5 = 0$, pa je

$$P(K)=\left(1-p_g\right)^5+5p_g\left(1-p_g\right)^4=\left(1+4p_g\right)\left(1-p_g\right)^4.$$

Koristeći se vjerojatnošću ispravnog dekodiranja, jednostavno izračunavamo i **vjerojatnost pogreške nakon dekodiranja pomoću standardnog niza** (engl. *word error rate*):

$$P_e(K)=1-P(K).$$

Uočimo da vjerojatnost $P_e(K)$ pokriva događaj kada se u standardnom nizu uopće ne može pronaći primljena riječ, pa dekoder kanala u tom slučaju prijavljuje nemogućnost dekodiranja.

Ukoliko je poznata distanca koda $d(K)$, onda iz svojstva o sposobnosti koda da otkrije i ispravi pogrešku slijedi da kôd K može ispraviti najviše t-struku pogrešku, gdje je $d(K) \geq 2t + 1$. Stoga zaključujemo da se u standardnom nizu zasigurno nalaze svi vektori pogreške s $0 \leq i \leq t$ jedinica. Za te kodove vrijedi da je broj vektora pogreške s i jedinica u standardnom nizu jednak:

$$N_i=\binom{n}{i}.$$

Međutim, u standardnom nizu se mogu nalaziti i vektori pogreške s više od t jedinica. Na žalost, ne postoji jednostavan način izračuna vrijednosti N_i za $i > t$. Međutim, ukoliko je kôd K perfektan, onda su sve riječi unutar sfera radijusa t, pa se u standardnom nizu nalaze isključivo vektori pogreške s t i manje jedinica. U tom slučaju je vjerojatnost pogreške perfektnog koda

$$P(K)=\sum_{i=0}^{t}\binom{n}{i}p_g{}^i\left(1-p_g\right)^{n-i}.$$

Proračun vjerojatnosti pogreške nakon dekodiranja može pomoći da odredimo performanse nekog kodnog kanala. Međutim, najviše nas zanima možemo li, koristeći odgovarajući zaštitni kôd, postići željenu vjerojatnost pogreške nakon dekodiranja. Odgovor na to pitanje nam daje Shannonov teorem o kodiranju [4]. Prije iskaza samog teorema, potrebno je uvesti pojam kodne brzine (engl. *code rate*).

DEFINICIJA

KODNA BRZINA ZAŠTITNOG KODA: Kodna brzina linearnog binarnog blok-koda $K=[n, k]$ s oznakom $R(K)$ je omjer:

$$R(K) = \frac{k}{n} \leq 1,$$

gdje je k dimenzija koda, a n duljina koda. Ukoliko generirajuća matrica koda K ima standardni oblik, onda je $R(K)$ udio korisnih (informacijskih) bitova u kodnoj riječi. Budući da je $n \geq k$, $R(K) \leq 1$.

TEOREM

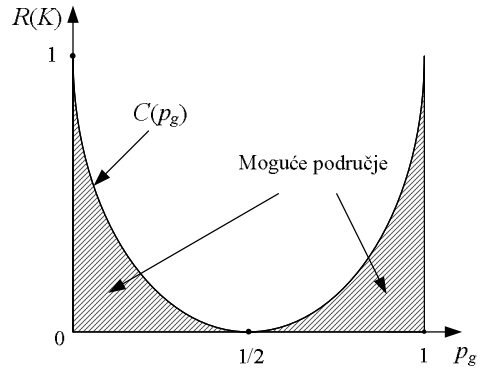
SHANNONOV TEOREM O KODIRANJU ZA BINARNI SIMETRIČNI KANAL: Neka je $C(p_g)$ kapacitet binarnog simetričnog kanala s vjerojatnošću pogreške simbola p_g dan izrazom

$$C(p_g) = 1 + p_g \log_2 p_g + (1 - p_g) \log_2 (1 - p_g) \text{ [bit / simbol]}.$$

Za bilo koji realan broj $\varepsilon > 0$ postoji zaštitni kôd $[n, k]$ s kodnom brzinom $R(K) = k/n$ koji ispunjava uvjet $R(K) \leq C(p_g)$, a koji će za danu vjerojatnost pogreške simbola p_g imati vjerojatnost pogreške nakon dekodiranja manju od ε, tj.

$$P_e(K) < \varepsilon .$$

Teorem o kodiranju nam kaže da za dani binarni simetrični kanal s vjerojatnošću pogreške p_g uvijek možemo konstruirati zaštitni kôd čija će vjerojatnost pogreške nakon dekodiranja $P_e(p_g)$ biti po volji mala dok god je kodna brzina takvog koda manja od kapaciteta kanala: $R(K) < C(p_g)$. Drugim riječima, dok god kodna brzina koda ne prelazi kapacitet kanala, komunikaciju je moguće realizirati pouzdanom po volji. Područje u kojem je moguće konstruirati kôd s proizvoljno malom pogreškom dekodiranja prikazano je na slici (Slika 0.11).



Slika 0.11: Moguće područje za konstrukciju zaštitnog koda s proizvoljno malom vjerojatnosti pogreške nakon dekodiranja

Za $R(K) > C(p_g)$ nije moguće postići po volji malu pogrešku $P_e(K)$. Postavlja se pitanje koliku je najmanju vjerojatnost pogreške nakon dekodiranja moguće postići ukoliko zaštitni kôd mora imati kodnu brzinu veću od kapaciteta. U teoriji informacije postoji niz manje i više uskih granica za $P_e(K)$ kada je $R(K) > C(p_g)$ koje u obzir uzimaju konstrukcijske vrijednosti samog koda (vidi [4]). Ovdje ćemo za određivanje ove granice iskoristiti originalni Shannonov teorem o kanalu s prisustvom smetnji koji kaže da je kodna brzina R ograničena odozgo izrazom:

$$R \le \frac{C}{1 + P_e \log_2 P_e + (1 - P_e)\log_2 (1 - P_e)},$$

gdje je C kapacitet kanala, a P_e prihvatljiva vjerojatnost pogreške poruke nakon dekodiranja. Budući da je kodna brzina R određena kodnom brzinom realiziranog zaštitnog koda $R(K)$, a P_e odgovara vjerojatnosti nakon dekodiranja promatranim kodom $P_e(K)$, slijedi:

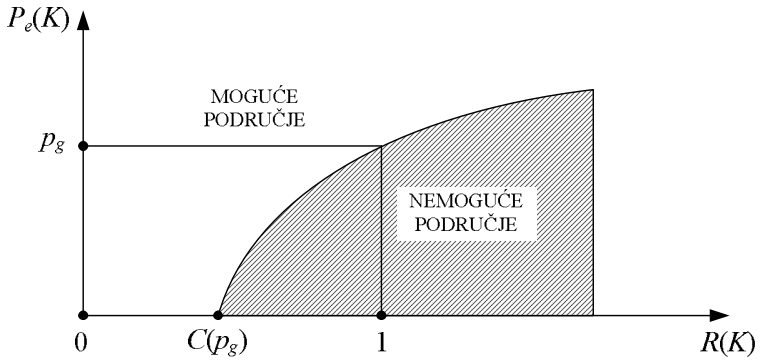
$$R(K) \le \frac{1 + p_g \log_2 p_g + (1 - p_g)\log_2 (1 - p_g)}{1 + P_e(K)\log_2 P_e(K) + (1 - P_e(K))\log_2 (1 - P_e(K))}.$$

OBRAT SHANNONOVOG TEOREMA O KODIRANJU: Blok-kôd K duljine n i kodne brzine $R(K)$ uz kapacitet kanala C zadovoljava sljedeću nejednakost za vjerojatnost pogreške nakon dekodiranja:

$$P_e(K) \ge 1 - \frac{1}{R(K)}\left(\frac{1}{n} + C\right).$$

Iz ovoga je vidljivo da za $R(K) > C(p_g)$ nije moguće postići proizvoljno mali $P_e(K)$.

Na slici (Slika 0.12) je prikazana skica mogućeg područja vrijednosti vjerojatnosti pogreški nakon dekodiranja za ciljane vrijednosti kodnih brzina koda [5]. Vidimo da ukoliko kodna brzina pređe vrijednost kapaciteta, nije moguće postići po volji malu pogrešku nakon dekodiranja, a granicu određuje izraz za obrat Shannonovog teorema.



Slika 0.12: Granice ostvarivih vrijednosti kodne brzine za pogodne vrijednosti vjerojatnosti nakon kodiranja zaštitnih kodova

Iako nam Shannonov teorem o kodiranju kaže da postoje kodovi koji osiguravaju po volji malu pogrešku nakon dekodiranja, on nam ne daje algoritam za stvaranje takvog koda. U općem slučaju generiranje zaštitnog koda je prepušteno domišljatosti konstruktora. Međutim, postoje klase linearnih blok-kodova kod kojih je moguće definirati generirajući matricu prema unaprijed

poznatoj distanci koda (npr. BCH i Reed-Solomonovi kodovi), što može znatno olakšati konstrukciju koda.

1.3. Hammingovi kodovi

Hammingov kôd je bilo koji linearni blok-kôd čija paritetna matrica **H** ima *r* redaka, a u stupcima ima sve moguće vektore dimenzije $r > 1$ osim vektora **0**. Formalno se Hammingov kôd definira na sljedeći način.

DEFINICIJA

HAMMINGOV KOD: Neka je *r* pozitivan cijeli broj i neka je **H** matrica dimenzija $r \times (2^r - 1)$ čije stupce sačinjavaju svi vektori dimenzije *r* različiti od **0** iz vektorskog prostora $V(r)$. Matrica **H** je paritetna matrica Hammingovog koda s oznakom Ham(*r*).

Pojednostavljeno rečeno, transponirana paritetna matrica Hammingovog koda Ham(*r*) ima u recima binarne ekvivalente cijelih dekadskih brojeva od 1 do $2^r - 1$. Primjeri paritetnih matrica Ham(3) su:

$$\mathbf{H}_1^{\mathrm{T}} = \left[\begin{array}{ccc|c} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 1 & 1 & 3 \\ 1 & 0 & 0 & 4 \\ 1 & 0 & 1 & 5 \\ 1 & 1 & 0 & 6 \\ 1 & 1 & 1 & 7 \end{array}\right] \quad \textit{ili} \quad \mathbf{H}_2^{\mathrm{T}} = \left[\begin{array}{ccc|c} 1 & 1 & 1 & 7 \\ 1 & 1 & 0 & 6 \\ 0 & 1 & 1 & 3 \\ 1 & 0 & 1 & 5 \\ 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 1 \end{array}\right]$$

Matrica $\mathbf{H}_1$ je u obliku koji se najčešće koristi u praksi i omogućuje jednostavnu programsku implementaciju, dok je $\mathbf{H}_2$ u standardnom obliku.

TEOREM

SVOJSTVA HAMMINGOVIH KODOVA: Neka je Ham(*r*) binarni Hammingov kôd. Za $r \geq 2$ vrijedi da je Ham(*r*):

- linearan blok-kôd $[2^r - 1, 2^r - 1 - r]$,
- ima najmanju distancu 3 (otkriva dvostruku i ispravlja jednostruku pogrešku),
- perfektan kôd.

Drugo svojstvo je izuzetno zanimljivo. Ono garantira da Hammingov kôd koji ima duljinu kodne riječi 7 ili veću, zasigurno može ispraviti jednostruku pogrešku, budući da je $\lfloor (3-2)/2 \rfloor = 1$.

Određivanje distance Hammingovog koda s proizvoljnim $n = 2^r - 1$ u općem slučaju nije jednostavan zadatak i ovaj problem nije u fokusu ovog poglavlja. Ovdje navodimo samo neke primjere Hammingovog koda s distancama većim ili jednakim 3 - Tablica 0.2.

Tablica 0.2: Mogući Hammingovi kodovi s pripadajućim distancama

$[n,k,3]$	$[n,k,5]$	$[n,k,7]$	$[n,k,9]$	$[n,k,11]$	$[n,k,13]$
[3,1,3]	[5,1,5]	[7,1,7]	[9,1,9]	[11,1,11]	[13,1,13]
[5,2,3]	[8,2,5]	[11,2,7]	[14,2,9]	[17,2,11]	[20,2,13]
[6,3,3]	[10,3,5]	[13,3,7]	[17,3,9]	[20,3,11]	[24,3,13]
[7,4,3]	[11,4,5]	[14,4,7]	[19,4,9]	[22,4,11]	[26,4,13]
[9,5,3]	[13,5,5]	[15,5,7]	[20,5,9]	[23,5,11]	[27,5,13]
[10,6,3]	[14,6,5]	[17,6,7]	[22,6,9]	[25,6,11]	[29,6,13]
[11,7,3]	[15,7,5]	[18,7,7]	[24,7,9]	[26,7,11]	[32,7,13]
[12,8,3]	[16,8,5]	[19,8,7]	[25,8,9]	[28,8,11]	[34,8,13]
[13,9,3]	[17,9,5]	[20,9,7]	[26,9,9]	[30,9,11]	[35,9,13]
[14,10,3]	[19,10,5]	[21,10,7]	[28,10,9]	[31,10,11]	[36,10,13]

1.3.1. Kodiranje pomoću Hammingovog koda

Promatrajmo sljedeću paritetnu matricu Hammingovog koda Ham(3) - [7,4,3]:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

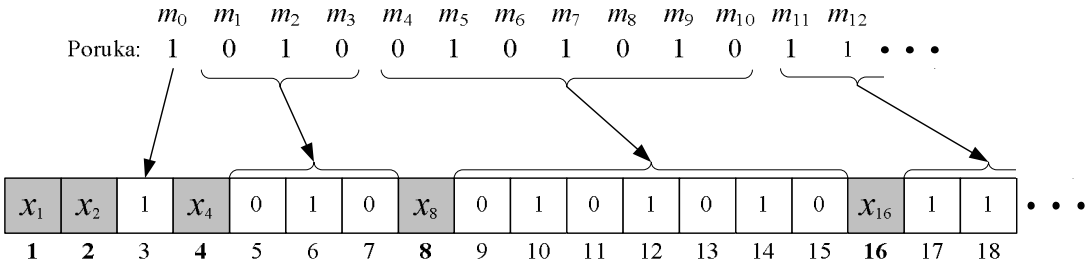
Stupci ove matrice predstavljaju binarne ekvivalente brojeva 1,2, …, 7. U općem slučaju ovakvu matricu dobivamo na način da se *i*-ti stupac formira od bitova koji predstavljaju binarni ekvivalent broja $i = 1, \ldots, 2^r-1$, gdje je r broj redaka paritetne matrice **H**.

Već smo rekli da svaki redak paritetne matrice određuje pozicije simbola kodne riječi čiji zbroj mora biti paran broj, odnosno jednak 0 u aritmetici modulo 2. Promatrana matrica **H** definira sljedeće pozicije u kodnoj riječi čiji zbroj vrijednosti simbola mora biti paran:

prvi redak	Pozicije (4, 5, 6 i 7),
drugi redak	Pozicije (2, 3), (6 i 7),
treći redak	Pozicije (1), (3), (5) i (7).

Vidimo da je matrica H strukturirana tako da se provjera pariteta vrši u grupama od po 1, 2 i 4 uzastopnih bitova, koje su međusobno razmaknute redom 1, 2 i 4 bita, a prva grupa uvijek počinje na 1., 2. ili 4. poziciji.

Ovakav način rasporeda pozicija za provjeru pariteta je najčešći način formiranja kodne riječi Hammingovog koda. Dodatno se definira da se bitovi za provjeru pariteta (zalihosni bitovi) postavljaju na pozicije koje odgovaraju potencijama broja 2, a bitovi kodirane poruke redom između njih. Struktura kodne riječi prikazana je na slici (Slika 0.13). Sivo su označene pozicije paritetnih bitova koji se određuju prema paritetnoj matrici.



Slika 0.13: Uobičajeni način formiranja kodne riječi Hammingovog koda

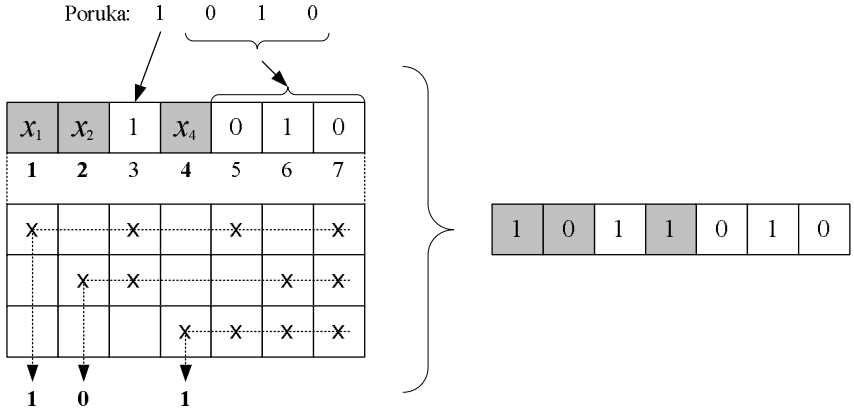
U skladu s ovako odabranom strukturom kodne riječi, moraju biti zadovoljene sljedeće jednakosti:

$$0 = x_1 + x_3 + x_5 + x_7 + x_9 + x_{11} + x_{13} + \ldots$$
$$0 = x_2 + x_3 \;+\; x_6 + x_7 \;+\; x_{10} + x_{11} \;+\ldots$$
$$0 = x_4 + x_5 + x_6 + x_7 \;+\; x_{12} + x_{13} + x_{14} + x_{15} \;+\ldots,$$
$$\vdots$$

gdje je x_i vrijednosti simbola na poziciji i. Grupe bitova su naglašene razmakom, a vodeći bitovi ujedno predstavljaju bitove za provjeru pariteta. Ukoliko ih prebacimo s lijeve strane jednakosti dobivamo i eksplicitne izraze za njihovo izračunavanje:

$$x_1 = m_1 + m_2 + m_4 + m_5 + m_7 + \ldots = x_3 + x_5 + x_7 + x_9 + \ldots$$
$$x_2 = m_1 + m_3 + m_4 + m_6 + m_7 + \ldots = x_3 + x_6 + x_7 + x_{10} + x_{11} + \ldots$$
$$x_4 = m_2 + m_3 + m_4 + m_8 + m_9 + m_{10} + m_{11} + \ldots = x_5 + x_6 + x_7 + x_{12} + x_{13} + x_{13} + x_{15} \ldots$$
$$\vdots$$

Postupak formiranja jedne kodne riječi za ulaznu kodiranu poruku 1010 i Hammingov kôd [7, 4, 3] ilustriran je na slici (Slika 0.14). Bit za provjeru pariteta x_1 dobiven je zbrajanjem bitova na pozicijama kodne riječi 3, 5 i 7, odnosno pozicijama kodirane poruke 1, 2 i 4: x_1=1+0+0=1. Bit za provjeru pariteta x_2 dobiven je zbrajanjem bitova na pozicijama kodne riječi 3, 6 i 7, odnosno pozicijama kodirane poruke 1, 3 i 4: x_2=1+1+0=0. Bit za provjeru pariteta x_3 dobiven je zbrajanjem bitova na pozicijama kodne riječi 5, 6 i 7, odnosno pozicijama kodirane poruke 2, 3 i 4: x_3=0+1+0=1. Dobivena je kodna riječ 1011010.



Slika 0.14: Formiranje kodne riječi Hammingovog koda [7,4,3]

Opisani postupak može se činiti kompliciran. Stoga se isplati pronaći generirajuću matricu koda [7,4,3] - **G**. Budući da paritetna matrica ima 3 retka, a duljina kodne riječi iznosi 7, slijedi da je

dimenzija koda k=4. To je ujedno i broj redaka generirajuće matrice koda **G**. Broj stupaca generirajuće matrice je jednak duljini kodne riječi - 7.

Kodna riječ Hammingovog koda se formira množenjem poruke (duljine 4 bita) s matricom **G**. Prvi bit kodne riječi se formira množenjem poruke s prvim stupcem matrice **G**, drugi bit množenjem poruke s drugim stupcem itd. Pogledajmo ponovno sliku (Slika 0.14). Jasno je da se prvi bit formira zbrajanjem 1., 2. i 4. bita kodirane poruke. Stoga će prvi stupac matrice **G** zasigurno biti $[1101]^T$.

Slično je i s drugim stupcem. Drugi bit poruke se formira zbrajanjem 1., 3. i 4. bita kodirane poruke, pa je drugi stupac $[1011]^T$. Treći bit kodne riječi je sam prvi bit poruke, pa je treći stupac matrice **G** zapravo $[1000]^T$. Nastavljajući ovaj postupak dolazimo do generirajuće matrice koda $[7,4,3]$:

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

Kako bi potvrdili ispravnosti matrice **G** potrebno ju je pomnožiti s transponiranom paritetnom matricom i provjeriti da je rezultat nul-matrica.

Sada se kodiranje svodi na jednostavan umnožak poruke i generirajuće matrice. Kodirajmo ponovno poruku 1010:

$$[1010]\cdot\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} = [1011010].$$

Dobivamo istu poruku kao i prethodnim postupkom.

1.3.2. Dekodiranje s Hammingovim kodom

Dekodiranje Hammingovim kodom postaje iznimno jednostavno ukoliko se koristi struktura paritetne matrice opisane u prethodnom potpoglavlju. Na primjer, Hammingov kôd $[7,4,3]$ ima paritetnu matricu:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Ukoliko primljena kodna riječ **y** ima pogrešku na i-tom bitu, sindrom $S(\mathbf{y})$ će biti binarni zapis vrijednosti i. Na primjer, neka primljena kodna riječ ima pogrešku na 5. bitu - kodna riječ 0000100. Dobiva se sindrom:

$$[0000100]\cdot\mathbf{H}^T = [101],$$

a to je upravo binarni zapis broja $5_{10} = 101_2$. Na taj način paritetna matrica omogućuje neposredno preslikavanje sindroma u indeks pozicije na kojoj je nastupila jednostruka pogreška.

Dakle postupak dekodiranja, uz pretpostavku jednostruke pogreške bita, se svodi na sljedeći postupak:

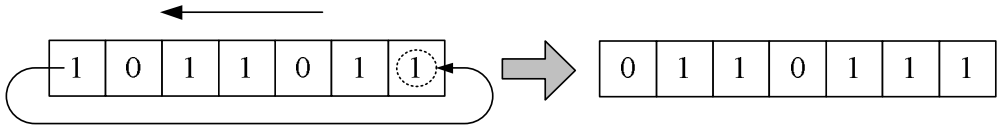
- Izračunaj sindrom $S(\mathbf{y})$ primljene kodne riječi $\mathbf{y}$,
- Ako je $S(\mathbf{y})\neq 0$, invertiraj bit na poziciji koja odgovara decimalnom ekvivalentu sindroma.

Naravno, moguće je koristiti i bilo koju drugu Hammingovu paritetnu matricu, međutim onda se mora upotrijebiti opća procedura dekodiranja sindromom linearnih blok-kodova.

Primijetimo da svaka primljena kodna riječ ima sindrom jednostruke pogreške, bez obzira nastupila jednostruka ili dvostruka pogreška. To svojstvo je posljedica perfektnosti koda. Način rješavanja ovog problema je korištenje proširenog Hammingovog koda, što izlazi izvan okvira ove knjige.

1.4. Ciklički kodovi

Ciklički kodovi su linearni blok-kodovi sa svojstvom da se cikličkim posmicanjem jedne kodne riječi uvijek dobiva neka kodna riječ iz istog cikličkog koda. Cikličko posmicanje kodne riječi $[a_{n-1}\ a_{n-2}\ldots\ a_2\ a_1\ a_0]$ je promjena pozicija simbola u kodnoj riječi na način da se dobije kodna riječ $[a_{n-2}\ldots\ a_2\ a_1\ a_0\ a_{n-1}]$ kada se radi o posmaku ulijevo, odnosno $[\ a_0\ a_{n-1}\ a_{n-2}\ldots\ a_2\ a_1]$ kada se radi o posmaku udesno. Ovo svojstvo ga čini izuzetno pogodnim za sklopovsku implementaciju.



Slika 0.15: Ilustracija cikličkog posmaka ulijevo

CIKLIČKI KÔD: Blok-kôd K je ciklički kôd ako je:

- linearan blok-kôd i
- ako bilo koji ciklički posmak kodne riječi iz K opet daje kodnu riječ iz K.

DEFINICIJA

Priroda cikličkog koda je takva da je kodnu riječ $[\ a_{n-1}\ a_{n-2}\ldots\ a_2\ a_1\ a_0\ ]$ korisno poistovjetiti s polinomom stupnja $n-1$:

$$\mathbf{a}=[a_{n-1}\ldots a_2\ a_1\ a_0]\leftrightarrow\ \ a(x)=a_{n-1}x^{n-1}+a_{n-2}x^{n-2}+\ldots+a_2x^2+a_1x^1+a_0x^0,$$

gdje su $a_i\in\{0,1\}$, a operacije se provode u aritmetici modulo 2. Na primjer, kodnu riječ $\mathbf{a}$=[10101] poistovjećujemo s polinomom

$$\mathbf{a}=[10101]\ \leftrightarrow\ \ a(x)=x^4+x^2+1.$$

Treba naglasiti da nam polinom $a(x)$ ovdje služi samo kao povoljnije sredstvo zapisa kodne riječi od vektora simbola. Polinom $a(x)$ ne promatramo kao funkciju čiji argument x može poprimiti neku konkretnu vrijednost, nego kao sredstvo zapisa kodne riječi.

Promotrimo proizvoljnu kodnu riječ opisanu polinomom $a(x)=a_{n-1}x^{n-1}+a_{n-2}x^{n-2}+\ldots+a_2x^2+a_1x^1+a_0x^0$ te ju pomnožimo s x i preoblikujmo kako slijedi:

$$\begin{aligned} x \cdot a(x) &= a_{n-1}x^n + a_{n-2}x^{n-1} + \ldots + a_2x^3 + a_1x^2 + a_0x^1 \\ &= x^n\left(a_{n-1}x^0\right) + a_{n-2}x^{n-1} + \ldots + a_1x^2 + a_0x^1 \\ &= x^n\left(a_{n-1}x^0\right) - a_{n-1}x^0 + a_{n-1}x^0 + a_{n-2}x^{n-1} + \ldots + a_1x^2 + a_0x^1 \\ &= a_{n-1}\left(x^n - 1\right) + a_{n-2}x^{n-1} + \ldots + a_1x^2 + a_0x^1 + a_{n-1}x^0. \end{aligned}$$

Uzmimo sada dobivenu kodnu riječ i izračunajmo ostatak dijeljenja polinomom $x^n - 1$ u aritmetici modulo 2:

$$\begin{array}{l} a_{n-1}\left(x^n - 1\right) + a_{n-2}x^{n-1} + \ldots + a_1x^2 + a_0x^1 + a_{n-1}x^0 \quad : \quad x^n - 1 = a_{n-1} \\ \underline{-a_{n-1}\left(x^n - 1\right)} \\ \qquad a_{n-2}x^{n-1} + \ldots + a_1x^2 + a_0x^1 + a_{n-1}x^0 \qquad \leftarrow \textit{ostatak nakon dijeljenja.} \end{array}$$

Ono što smo upravo utvrdili je postupak cikličkog posmicanja polinoma $a(x)$ reda $n-1$ (tj. cikličke kodne riječi duljine n) za jedno mjesto ulijevo. U općem slučaju postupak za posmicanje za m pozicija ulijevo je:

- pomnoži polinom $a(x)$ s x^m,
- izračunaj ostatak dijeljenja polinomom $x^n - 1$.

Primjer: Ciklički posmak ulijevo

Promatrajmo kôd duljine $n = 3$ i kodnu riječ **a** $= 101$. Polinom koji odgovara ovoj kodnoj riječi je $a(x) = x^2 + 1$. Pomnožimo ovaj polinom[5] s x i izračunajmo ostatak dijeljenja s $x^3 - 1$:

$$b'(x) = a(x) \cdot x = x^3 + x,$$

$$\begin{array}{l} x^3 + x \quad : \; x^3 - 1 = 1 \\ \underline{-x^3 \quad +1} \\ \qquad x + 1 \qquad \leftarrow \textit{ostatak nakon dijeljenja.} \end{array}$$

Ostatak dijeljenja **b**=011 predstavlja kodnu riječ **a** koja je ciklički posmaknuta ulijevo za jednu poziciju. Isti postupak rezultira ciklički posmakom za sve riječi duljine 3.

Budući da posmicanjem bilo koje kodne riječi cikličkog koda moramo opet dobiti kodnu riječ koda, slijedi da polinomi svih kodnih riječi moraju biti u **modulo $x^n - 1$ aritmetici[6].** Skup svih polinoma $F[x] = a(x) : a(x)(mod\,(x^n - 1))$ označimo s R_n. Poistovjećujući kodne riječi s

PRIMJER

[5]Polinom namjerno pišemo od većeg prema manjem stupnju kako bi njegova struktura odgovarala značaju bitova kodnih riječi.

[6] Polinomi u aritmetici modulo x^n-1 su oni polinomi koji su dobiveni dijeljenjem polinoma proizvoljnog stupnja polinomom x^n-1, odnosno svi oni polinomi koji pri dijeljenju s x^n-1 za ostatak nakon dijeljenja daju same sebe. Oznaka polinoma $a(x)$ u aritmetici modulo x^n-1 je $a(x)(\text{mod }(x^n-1))$.

polinomima, ciklički kôd K je neki podskup skupa R_n, $K \subset R_n$. Važno je uočiti da su koeficijenti polinoma binarnih cikličkih kodova u aritmetici modulo 2. Stoga je 1 suprotan element od –1, pa je i polinom $x^n - 1 = x^n + 1$.

Postavlja se pitanje kako generirati konkretan ciklički kôd? Odgovor na to pitanje slijedi iz sljedećeg važnog svojstva.

TEOREM

UVJETI ZA CIKLIČKI KOD: Kôd $K \subset R_n$ je ciklički kôd ako i samo ako K zadovoljava sljedeća dva uvjeta:

- $\forall\, a(x),\ b(x) \in K \ \Rightarrow\ a(x) + b(x) \in K$,
- $\forall\, a(x) \in K$ i $\forall\, r(x) \in R_n \ \Rightarrow\ r(x) \cdot a(x) \bmod (x^n - 1) \in K$.

Prvo svojstvo odgovara svojstvu linearnosti. Drugo svojstvo kaže da ukoliko bilo koji polinom koda K pomnožimo s bilo kojim polinomom iz skupa R_n, ponovno dobivamo neki polinom (kodnu riječ) iz skupa K. Kažemo da je ciklički kôd zatvoren s obzirom na množenje s bilo kojim polinomom iz R_n. Budući da su svi polinomi $a(x)$ također u skupu R_n, ovim svojstvom je dan jednostavan algoritam za generiranje cikličkog koda s duljinom kodne riječi n:

- izaberi bilo koji polinom $f(x)$ najvećeg stupnja $n-1$,
- sve kodne riječi cikličkog koda K dobit ćemo množenjem svih $r(x) \in R_n$ s $f(x)$.

$f(x)$ je element koda K i kaže se da je kôd K generiran polinomom $f(x)$, s oznakom

$$K \equiv \langle f(x) \rangle = \{ r(x)\, f(x) \mid r(x) \in R_n \}.$$

PRIMJER

Primjer: Generiranje cikličkog koda

Neka je n=3. Odabiremo polinom $f(x) = x^2 + 1$. R_n ima ukupno $2^3 = 8$ polinoma koji pomnoženi s $f(x)$ u aritmetici modulo $(x^3 - 1)$ daju:

$$\begin{aligned}
(0x^2 + 0x + 0) &\cdot (x^2 + 1)(\bmod (x^3 - 1)) = 0x^2 + 0x + 0 \quad [000] \\
(0x^2 + 0x + 1) &\cdot (x^2 + 1)(\bmod (x^3 - 1)) = 1x^2 + 0x + 1 \quad [101] \\
(0x^2 + 1x + 0) &\cdot (x^2 + 1)(\bmod (x^3 - 1)) = 0x^2 + 1x + 1 \quad [011] \\
(0x^2 + 1x + 1) &\cdot (x^2 + 1)(\bmod (x^3 - 1)) = 1x^2 + 1x + 0 \quad [110] \\
(1x^2 + 0x + 0) &\cdot (x^2 + 1)(\bmod (x^3 - 1)) = 1x^2 + 1x + 0 \quad [110] \\
(1x^2 + 0x + 1) &\cdot (x^2 + 1)(\bmod (x^3 - 1)) = 0x^2 + 1x + 1 \quad [011] \\
(1x^2 + 1x + 0) &\cdot (x^2 + 1)(\bmod (x^3 - 1)) = 1x^2 + 0x + 1 \quad [101] \\
(1x^2 + 1x + 1) &\cdot (x^2 + 1)(\bmod (x^3 - 1)) = 0x^2 + 0x + 0 \quad [000]
\end{aligned}$$

Budući da imamo ponavljanje polinoma, ukupan broj kodnih riječi cikličkog koda je 4: 000, 101, 011, 110. Distanca ovog koda je 2 (najmanja težina kodnih riječi je 2), što znači da kôd može detektirati jednostruku pogrešku. Generirajuću matricu možemo dobiti uzimajući drugu i treću kodnu riječ pa dobivamo:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

Jasno, generiranje cikličkog koda na ovaj način može biti izuzetno složeno zbog velikog broja riječi. Stoga se uvodi pojam generirajućeg polinoma cikličkog koda koji jednoznačno određuje sam kôd i generirajuću matricu koda.

1.4.1. Generirajući polinom i generirajuća matrica

GENERIRANJE CIKLIČKOG KODA: Neka je *K* ciklički kôd dimenzije veće od 1, podskup od R_n.

TEOREM

- Postoji jedinstven polinom $g(x)$ najmanjeg stupnja u *K*.
- Kôd *K* je generiran upravo polinomom $g(x)$.
- $g(x)$ je faktor polinoma $x^n - 1$, tj. $x^n - 1 = g(x) \cdot q(x)$.

Ovo svojstvo potrebno je pažljivo tumačiti. Prva i druga točka kažu da je ciklički kôd *K* moguće generirati polinomom koji je jedini polinom svog stupnja i da ima najmanji stupanj od svih polinoma u kodu. Druga točka kaže da upravo taj jedinstveni polinom generira ciklički kôd *K*. Treća točka kaže da je generirajući polinom bilo koji faktor polinoma $x^n - 1$, tj. polinom koji dijeli $x^n - 1$ bez ostatka.

Na primjer, ukoliko se promatra ciklički kôd podskupa R_3, onda kôd *K* mora biti u aritmetici modulo $(x^3 - 1)$. $x^3 - 1$ moguće je faktorizirati na sljedeći način[7]:

$$x^3 - 1 = 1 \cdot (x+1) \cdot (x^2 + x + 1).$$

Stoga su svi mogući ciklički kodovi oni koji se mogu generirati polinomima: $g(x)=1$, $g(x) = x+1$, $g(x) = x^2 + x + 1$ i sam $g(x) = x^3 - 1$, budući da sam sebe dijeli bez ostatka.

DEFINICIJA

GENERIRAJUĆI POLINOM CIKLIČKOG KODA: Polinom najmanjeg stupnja cikličkog koda *K* zove se **generirajući polinom** cikličkog koda *K*.

Kada je odabran generirajući polinom cikličkog koda, određivanje generirajuće matrice **G** koda postaje trivijalno.

TEOREM

GENERIRAJUĆA MATRICA CIKLIČKOG KODA: Neka je generirajući polinom cikličkog koda $K \subset R_n$:

$$g(x) = g_r x^r + \ldots + g_2 x^2 + g_1 x + g_0.$$

[7] Uvijek treba imati u vidu da je $-1 = 1$ u modulo 2 aritmetici!

Onda je dimenzija koda $k = n - r$, a generirajuća matrica koda je:

$$\mathbf{G} = \begin{bmatrix} g_r & g_{r-1} & g_{r-2} & \cdots & g_1 & g_0 & 0 & 0 & \cdots & 0 \\ 0 & g_r & g_{r-1} & g_{r-2} & \cdots & g_1 & g_0 & 0 & \cdots & 0 \\ 0 & 0 & g_r & g_{r-1} & g_{r-2} & \cdots & g_1 & g_0 & \vdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & g_r & g_{r-1} & g_{r-2} & \cdots & g_1 & g_0 \end{bmatrix}.$$

Primjer: Generirajući polinom i generirajuća matrica

PRIMJER

Formirajmo ciklički kôd čija će duljina kodne riječi biti $n = 5$. Kôd mora biti u aritmetici modulo x^5-1, a generirajući polinom koda je faktor polinoma x^5-1 koji ima sljedeću faktorizaciju:

$$x^5 - 1 = (x+1)(x^4 + x^3 + x^2 + x + 1).$$

Stoga su mogući sljedeći generirajući polinomi: $g_0(x) = 1$, $g_1(x) = x+1$, $g_2(x) = x^4 + x^3 + x^2 + x + 1$ i $g_3(x) = x^5 + 1$. Generirajući polinomi $g_0(x) = 1$ i $g_3(x) = x^5 + 1$ nam nisu interesantni budući da $g_0(x) = 1$ generira kôd koji ima sve kodne riječi iz $V(5)$, a $g_3(x) = x^5 + 1$ generira kôd koji ima samo kodnu riječ 00000. Također $g_2(x)$ daje kôd dimenzije $k = n - r = 5 - 4 = 1$ i ima samo jednu kodnu riječ 11111.

Generirajući polinom $g_1(x) = x + 1$ određuje kôd dimenzije $k = n - r = 5 - 1 = 4$ s generirajućom matricom

$$\mathbf{G_1} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

Pokazali smo da je generiranje cikličkog koda trivijalno kada su poznati faktori polinoma $x^n - 1$ koda duljine *n*. Najveći problem je provesti samu faktorizaciju. Matematička literatura obiluje različitim algoritmima za faktorizaciju, no ovdje ćemo jednostavno nabrojati faktorizacije za nekoliko najčešće korištenih duljina kodnih riječi. Prilikom generiranja cikličkog koda jednostavno treba odabrati povoljan generirajući polinom - Tablica 0.3.

Nažalost, ne postoji nikakvo poznato svojstvo cikličkog koda na osnovu kojeg bi uz poznat generirajući polinom i veličinu kodne riječi mogli izračunati distancu koda. Postoje samo općeprihvaćeni pristupi proračunu distance koda za pojedine klase cikličkih kodova. No ovi pristupi su često složeni i zahtjevni. Stoga se koriste tablice već poznatih rezultata na osnovi kojih se donosi odluka o izboru koda.

Tablica 0.3: Faktorizacije nekih polinoma oblika $x^n - 1$ u aritmetici modulo 2

n	aritmetika	*faktorizacija u aritmetici modulo 2*
1	$x^1 - 1$	$x+1$
2	$x^2 - 1$	$(x+1)^2$
3	$x^3 - 1$	$(x+1)(x^2+x+1)$
5	$x^5 - 1$	$(x+1)(x^4+x^3+x^2+x+1)$
7	$x^7 - 1$	$(x+1)(x^3+x+1)(x^3+x^2+1)$
9	$x^9 - 1$	$(x+1)(x^2+x+1)(x^6+x^3+1)$
11	$x^{11} - 1$	$(x+1)(x^{10}+x^9+x^8+x^7+x^6+x^5+x^4+x^3+x^2+x+1)$
13	$x^{13} - 1$	$(x+1)(x^{12}+x^{11}+x^{10}+x^9+x^8+x^7+x^6+x^5+x^4+x^3+x^2+x+1)$
15	$x^{15} - 1$	$(x+1)(x^2+x+1)(x^4+x+1)(x^4+x^3+1)(x^4+x^3+x^2+x+1)$
17	$x^{17} - 1$	$(x+1)(x^8+x^5+x^4+x^3+1)(x^8+x^7+x^6+x^4+x^2+x+1)$
19	$x^{19} - 1$	$(x+1)(x^{18}+x^{17}+x^{16}+\ldots+x^4+x^3+x^2+x+1)$

1.4.2. Paritetna matrica

Generirajuću matricu cikličkog koda **G** nije moguće svesti u standardni oblik, pa nije moguće niti jednostavno odrediti paritetnu matricu **H**. Međutim, paritetnu matricu je moguće konstruirati koristeći polinom za provjeru pariteta cikličkog koda.

DEFINICIJA

POLINOM ZA PROVJERU PARITETA: Neka je K ciklički kôd duljine n i dimenzije k $[n,k]$ s generirajućim polinomom $g(x)$. Neka je $h(x)$ polinom koji zadovoljava jednadžbu:

$$x^n - 1 = g(x)h(x).$$

$h(x)$ se zove polinom za provjeru pariteta cikličkog koda K.

Budući da je $g(x)$ polinom stupnja r, onda je $h(x)$ polinom stupnja $k=n-r$. Polinom za provjeru pariteta ima sljedeće važno svojstvo.

DEFINICIJA

PARITETNA MATRICA CIKLIČKOG KODA: Neka je $K \subset R_n$ ciklički kôd duljine n i dimenzije k $[n,k]$ s generirajućim polinomom $g(x)$ i polinomom za provjeru pariteta $h(x)$.

- Bilo koji polinom $c(x)$ koda K zadovoljava jednakost $c(x)h(x) = 0$.
- Ako je polinom za provjeru pariteta

$$h(x) = h_k x^k + \ldots + h_2 x^2 + h_1 x + h_0,$$

onda je paritetna matrica koda K

$$\mathbf{H} = \begin{bmatrix} h_0 & h_1 & h_2 & \cdots & h_{k-1} & h_k & 0 & 0 & \cdots & 0 \\ 0 & h_0 & h_1 & h_2 & \cdots & h_{k-1} & h_k & 0 & \cdots & 0 \\ 0 & 0 & h_0 & h_1 & h_2 & \cdots & h_{k-1} & h_k & \vdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & h_0 & h_1 & h_2 & \cdots & h_{k-1} & h_k \end{bmatrix}.$$

Primjer: ciklički kôd [7,4,3]

Konstruirajmo ciklički kôd s duljinom kodne riječi $n = 7$. Generirajući polinom mora biti stupnja $r < 7$ te mora faktorizirati polinom x^7-1. Iz tablice faktorizacija (Tablica 0.3) vidimo daje faktorizacija ovog polinoma:

$$x^7 - 1 = (x+1)(x^3 + x + 1)(x^3 + x^2 + 1).$$

Na primjer, za generirajući polinom možemo odabrati drugi faktor, tj. $g(x) = x^3 + x + 1$. Budući da je stupanj odabranog generirajućeg polinoma r=3, dobivamo da je dimenzija koda $k = n–r = 4$. Zaključujemo da će generirajuća matrica imati 4 retka i 7 stupaca, dok će paritetna matrica imati 3 retka i 7 stupaca. Koristeći se svojstvima o strukturi generirajuće i paritetne matrice dobivamo:

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \quad \text{i} \quad \mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}.$$

Množenjem matrica dobivamo nul-matricu $\mathbf{G}\cdot\mathbf{H}^{\mathrm{T}} = \mathbf{0}$, što potvrđuje da smo konstruirali ispravne matrice. Jedina stvar koja nedostaje je distanca dobivenog koda. Budući da je dimenzija koda 4, nije teško ispisati sve kodne riječi i pronaći onu s najmanjom težinom. Kodne riječi koje daje matrica **G** su:

$$0001101 \to 0011010 \to 0110100 \to 1101000 \to 1010001 \to 0100011 \to 1000110,$$

$$0010111 \to 0101110 \to 1011100 \to 0111001 \to 1110010 \to 1100101 \to 1001011,$$

$$000000, \quad 1111111.$$

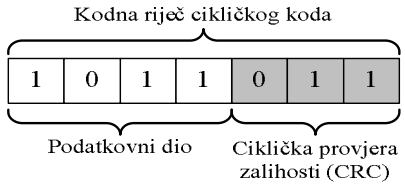
Riječi su razdvojene u dva podskupa dobivena posmicanjem kodnih riječi 1101000 i 1011100, te kodne riječi 0000000 i 1111111. Potvrdili smo da se ciklički posmak bilo koje riječi opet daje kodnu riječ istog koda. Najmanja težina svih kodnih riječi različitih od **0** je 3, što je ujedno i težina svih kodnih riječi nastalih posmicanjem riječi 1101000. Stoga je distanca ovog koda 3 pa je konstruiran kôd [7,4,3]. Iz svojstava o distanci koda zaključujemo da ovaj kôd može otkriti dvostruku pogrešku i ispraviti jednostruku pogrešku.

1.4.3. Implementacija kodera i dekodera cikličkog koda

Kodne riječi ostvarene množenjem poruke s matricom **G** imat će strukturu u kojoj su bitovi poruke ispremiješani sa zalihosnim paritetnim bitovima. Kaže se da **G** daje nesistematičan kôd.

U praksi je iznimno važno jednostavno izdvojiti podatkovne bite, a ujedno zadržati svu funkcionalnost cikličkog koda.

Najčešće se koristi algoritam kodiranja koji proizvodi kôd u kojem se redom (od najznačajnijih do najmanje značajnih pozicija) prvo nalaze bitovi poruke, a onda zaštitni bitovi. Dio kodne riječi u kojem se nalaze zaštitni bitovi naziva se **ciklička provjera zalihosti** (engl. *cyclic redundancy check* - CRC) - Slika 0.16. U nastavku ćemo ovu vrstu cikličkog koda zvati i CRC kôd.



Slika 0.16: Struktura kodne riječi sistematičnog cikličkog koda

Ideja algoritma je sljedeća: Neka je $d(x)$ polinom koji opisuje kodiranu poruku. Neka je $g(x)$ generirajući polinom stupnja r. Polinom $d(x)$ može se pomnožiti s x^r i podijeliti s $g(x)$ u aritmetici modulo 2, nakon čega se dobiva polinom $q(x)$ i ostatak $r(x)$:

$$d(x) \cdot x^r = g(x)q(x) + r(x).$$

$r(x)$ je polinom stupnja manjeg od r. Budući da je $q(x) \in R_n$, slijedi da je $g(x)q(x)$ zasigurno neka kodna riječ cikličkog koda. Ta kodna riječ je u aritmetici modulo 2 jednaka:

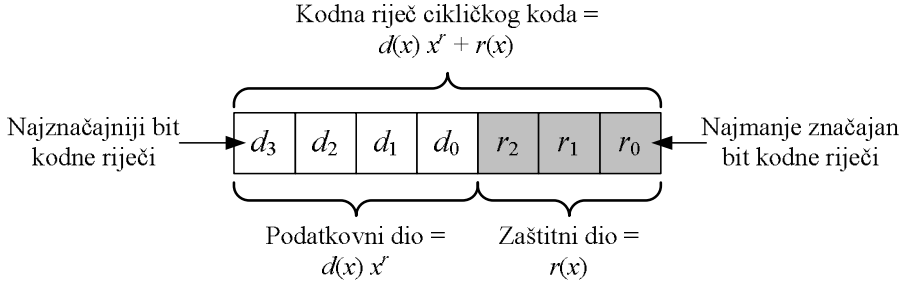
$$c(x) = g(x)q(x) = d(x) \cdot x^r + r(x),$$

gdje je

$$r(x) = d(x) \cdot x^r \bmod [g(x)].$$

Dakle, dobivena ciklička kodna riječ sastoji se od umnoška $d(x) \cdot x^r$ i polinoma $r(x)$. $d(x)$ predstavlja ulaznu kodiranu poruku i polinom je stupnja $n–r–1$. Množenjem s x^r pomaknut je ulijevo za r pozicija tako da u kodnoj riječi $c(x)$ najznačajnijih $n–r$ pozicija zauzimaju upravo bitovi kodirane poruke.

Polinom $r(x)$ je stupnja manjeg od r, te u ukupnoj kodnoj riječi zauzima r najmanje značajnih bitova kodne riječi - Slika 0.17. Polinom $r(x)$ predstavlja cikličku redundantnu zaštitu kodne riječi.



Slika 0.17: Struktura polinoma kodne riječi $c(x)=d(x)x^r + r(x)$

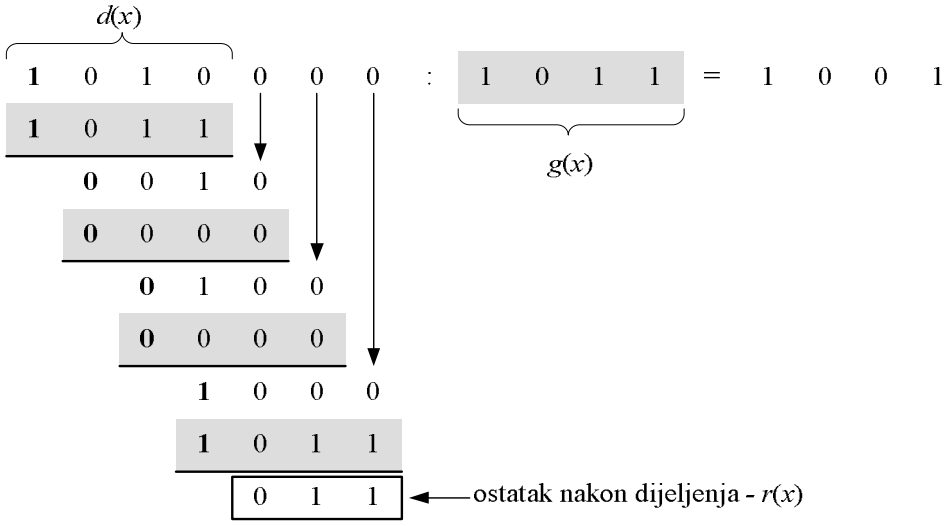
Dakle, da bismo formirali kodnu riječ za ulaznu poruku $d(x)$, sve što treba napraviti je izračunati ostatak dijeljenja $d(x)\cdot x^r$ s generirajućim polinomom $g(x)$ i ostatak zdesna dodati samoj kodiranoj poruci.

U praksi se dijeljenje polinoma pojednostavljuje na način da se promatraju samo binarni koeficijenti polinoma. Potrebno je jedino imati u vidu pravilo orijentacije pisanja binarnog ekvivalenta. Dogovorno uzimamo da prvi bit slijeva ima najveću težinu, a prvi zdesna najmanju težinu. Stoga su binarni brojevi ekvivalentni polinomima na sljedeći način:

$$g(x) = g_r x^r + g_{r-1}x^{r-1} + \cdots + g_2 x^2 + g_1 x + g_0 \;\equiv\; [g_r\ g_{r-1} \cdots g_2\ g_1\ g_0].$$

Primjer:Formiranje kodne riječi cikličkog koda [7,4]

Promatrajmo ciklički kôd [7,4], s generirajućim polinomom $g(x) = x^3 + x + 1$. Stupanj generirajućeg polinoma je $r = 3$, a $k = 4$. Ulazna kodna riječ je 1010, tj. polinom $d(x) = x^4 + x^2$. Izvršimo dijeljenje polinoma $d(x)\cdot x^3 = x^6 + x^4$ polinomom $g(x)$ koristeći samo binarne koeficijente polinoma kako je prikazano na slici (Slika 0.18).



Slika 0.18: Pojednostavljenje dijeljenja polinoma x^6+x^4 polinomom x^3+x+1

Primijetimo da u svakom koraku vodeći (lijevi) bit predstavlja kvocijent dijeljenja (podebljani bitovi), te da je stupanj ostatka uvijek za (najmanje) jedan manji od polinoma kojeg dijelimo. Vodeći (podebljani) bit množi generirajući polinom i oduzimamo ga od trenutno dijeljenog polinoma. Nastaje ostatak koji je za jedan stupanj manji. U svakom koraku posuđujemo i nulu (strelica) iz početnog polinoma. Dijeljenje obavljamo bez preskakanja, dakle i onda kada $g(x)$ u polinom "ulazi" nula puta.

Ostatak pri dijeljenu je 011, odnosno polinom $x+1$, a to je upravo zalihosni (CRC) dio kodne riječi, pa je kodna riječ cikličkog koda:

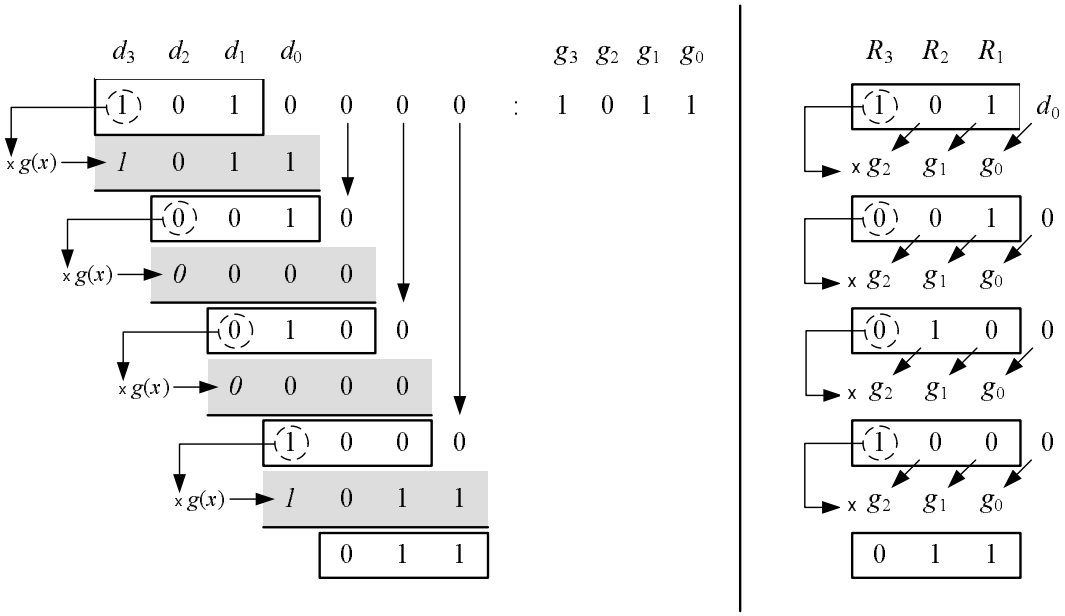
$$1010\mathbf{011}.$$

Postupak dijeljenja polinoma opisan u prethodnom primjeru može se vrlo jednostavno implementirati pomoću logičkih sklopova i bistabila. Promatrajmo vodeća tri bita dijeljenog polinoma u svakom koraku (Slika 0.19). Označimo ih redom R_3, R_2 i R_1. Oni su posebno naglašeni u desnom dijelu slike.

Možemo primijetiti vrlo jednostavan algoritam dobivanja trojke u sljedećem koraku iz trojke u prethodnom koraku. Naime, vidimo da R_3 u svakom koraku množi bitove generirajućeg polinoma, pa se od dijeljenog polinoma oduzima ili 0000 ili sam generirajući polinom $g(x)$. Budući da se oduzimanjem uvijek gubi vodeći član ($R_3 - R_3 \cdot g_3 = 0$), trojku u sljedećem koraku uvijek dobivamo prema sljedećim izrazima:

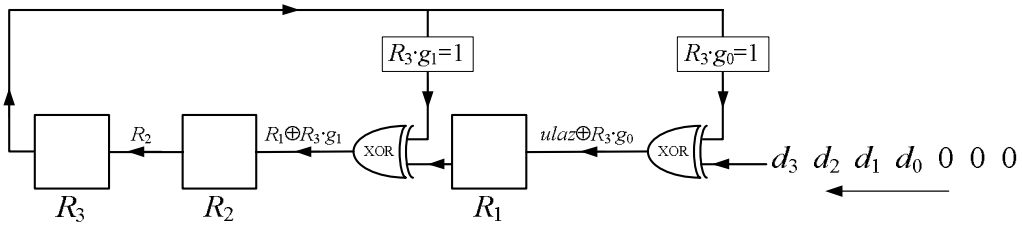
$$(1) \quad R_3 = R_2 \oplus (R_3 \cdot g_2)$$
$$(2) \quad R_2 = R_1 \oplus (R_3 \cdot g_1)$$
$$(3) \quad R_1 = \text{ulazni bit} \oplus (R_3 \cdot g_0),$$

gdje je sljedeći bit $d_0 = 0$ ili 0 nastala množenjem $d(x)$ s x^3, a operatorom $\oplus$ je naglašeno zbrajanje u aritmetici modulo 2 – ekskluzivno *ili*.



Slika 0.19: Algoritam izračunavanja ostatka pri dijeljenju polinoma

Identičan postupak se ponavlja u svim koracima sve dok se ne dobije konačni ostatak. Zbog toga je moguće izgraditi sklop u kojem je svakom R_1, R_2 i R_3 pridružen jedan bistabli. Sklop je shematski prikazan na slici (Slika 0.20).



Slika 0.20: Shematski prikaz sklopa za izračunavanje CRC zaštitnog dijela koda [7, 4] s generirajućim polinomom $g(x)=x^3+x+1$

Kao što se može vidjeti, ispred bistabila R_1 i R_2 je umetnut sklop koji vrši operaciju ekskluzivnog *ili* između koeficijenta g_i i prethodnog bistabila, odnosno ulaza. Ekskluzivni *ili* ($\oplus$) odgovara zbrajanju u aritmetici modulo 2. Na ulaz bistabila R_1 dovodi se rezultat operacije ekskluzivnog *ili*

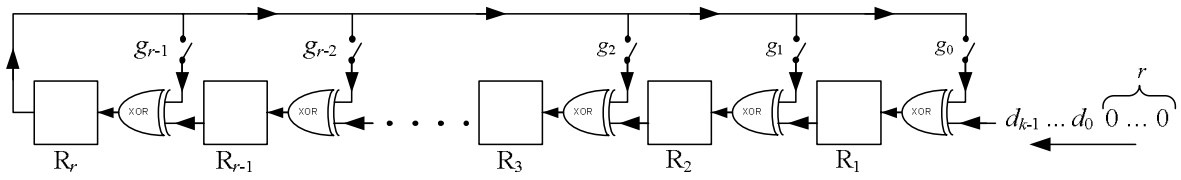
između ulaznog bita i trenutnog stanja bistabila R_3. To odgovara operaciji R_1= ulazni bit $\oplus$ $R_3 \cdot g_0$, gdje je g_0=1.

Jednako tako, na ulaz bistabila R_2 dovodi se rezultat operacije ekskluzivnog ili između trenutnog stanja bistabila R_1 i trenutnog stanja bistabila R_3. To odgovara operaciji $R_2 = R_1 \oplus R_3 \cdot g_1$, gdje je g_1=1.

Na ulaz bistabila R_3 se dovodi jedino trenutno stanje registra R_2. Razlog tome je u činjenici da je g_2=0, pa je izraz $R_3 = R_2 \oplus R_3 \cdot g_2 = R_2$. Zaključujemo da se ekskluzivni *ili* vrši samo ispred onih registara čiji je odgovarajući koeficijent generirajućeg polinom jednak 1. Stoga ovaj sklop obavlja operaciju koja je opisana slikom (Slika 0.19). Primijetimo još da vodeći koeficijent polinoma nema vlastiti sklop za ekskluzivni *ili*.

Postupak dobivanja ostatka $r(x)$ je da se na početku u registre R_1, R_2 i R_3 upiše 0. Nakon toga se u svakom koraku (taktu) na ulaz dovode redom bitovi poruke d_3, d_2, d_1 i d_0, a nakon toga slijede tri nule. Potrebno je naglasiti da se u jednom koraku (taktu) odvijaju sve tri opisane operacije odjednom.

U općem slučaju sklop za generiranje zaštitnog dijela cikličkog koda s generirajućim polinomom $g(x) = g_r x^r + \cdots + g_1 x + g_0$ izgleda kako je prikazano na slici (Slika 0.21). I u ovom slučaju koeficijent g_r nema vlastiti sklop za ekskluzivni *ili*. Zaštitni dio se formira u bistabilima $R_r, \ldots, R_1$ na način da se na početku u njih upišu nule, a nakon toga se na ulaz dovedu svi bitovi poruke redom $d_{k-1}, \ldots, d_0$. Nakon njih se na ulaz dovodi točno r bitova 0, nakon čega se u $R_r, \ldots, R_1$ nalazi zaštitni dio koda.



Slika 0.21: Shematski prikaz sklopa za izračunavanje CRC zaštitnog dijela

Postavlja se pitanje kako otkriti i eventualno ispraviti pogrešku primljene cikličke kodne riječi. Promatrajmo ponovno kôd s generirajućim polinomom $g(x) = x^3 + x + 1$. Ranije smo utvrdili da je distanca koda K=[7,4] $d(K)$=3, tako da ovaj kôd može ispraviti jednostruku pogrešku.

Ukoliko dio primljene kodne riječi koji odgovara samoj kodiranoj poruci (prva četiri bita) u dekoderu ponovno propustimo kroz sklop za kodiranje, ostatak mora biti jednak zaštitnom dijelu ako je poruka primljena bez pogreške. No, postavlja se pitanje kako izračunati sindrom i na osnovu njega odrediti poziciju pogreške.

Cilj je sklop za kodiranje iskoristiti i za dekodiranje. Stoga je potrebno pronaći način izračunavanja sindroma istim sklopom kao na slici (Slika 0.21). U tu svrhu ćemo definirati polinom pogreške $e(x)$, koji ima isti smisao kao i vektor pogreške. Primljena kodna riječ $y(x)$ se može zapisati kao poslana kodna riječ $c(x)$ kojoj je dodan polinom pogreške $e(x)$:

$$y(x) = c(x) + e(x).$$

Svi polinomi $y(x)$ koji imaju isti $e(x)$ moraju imati isti sindrom pri dekodiranju. Iako bi se sindrom mogao dobiti množenjem paritetnim polinomom $h(x)$, ovdje ćemo upotrijebiti nešto drugačiji način računanja. Promotrimo funkciju

$$S[y(x)] = x^r \cdot y(x) \ mod \ g(x), \qquad (0.1)$$

gdje je r stupanj generirajućeg polinoma. Evidentno je sljedeće:

$$\begin{aligned} S[y(x)] &= x^r y(x) \ mod \ g(x) \\ &= x^r [c(x) + e(x)] \ mod \ g(x) \\ &= x^r c(x) \ mod \ g(x) + x^r e(x) \ mod \ g(x) \\ &= S[c(x)] + S[e(x)]. \end{aligned}$$

Pokušajmo izračunati $S[c(x)]$. Budući da znamo da se svaka kodna riječ može napisati kao umnožak generirajućeg polinoma i polinoma $q(x) \in R_n$ slijedi:

$$\begin{aligned} c(x) &= g(x)q(x) \ | \cdot x^r \ \Rightarrow \\ c(x)x^r &= g(x)q(x)x^r. \end{aligned}$$

Budući da su obje strane djeljive s $g(x)$ bez ostatka, zasigurno vrijedi:

$$S[c(x)] = x^r \cdot c(x) \ mod \ g(x) = 0.$$

Tako dolazimo do zaključka

$$S[y(x)] = S[c(x)] + S[e(x)] = S[e(x)],$$

tj. da $S[y(x)]$ za sve kodne riječi s istim vektorom pogreške $e(x)$ daje isti rezultat. Tako bez provjere za cijeli standardni niz možemo napisati tablicu pridruživanja - Tablica 0.4.

Tablica 0.4: Sindromi koda $[7,4,3]$

$e(x)$	$S[e(x)]$
1	$x+1$
x	x^2+x
x^2	x^2+x+1
x^3	x^2+1
x^4	1
x^5	x
x^6	x^2

Kao što vidimo, različiti polinomi pogreške $e(x)$ generiraju različite vrijednosti $S[y(x)]$, pa je $S[y(x)]$ definiran izrazom (0.1) zapravo **sindrom primljene kodne riječi** $y(x)$ koda $[7,4]$.

Treba uočiti da je izraz koji definira način računanja sindroma

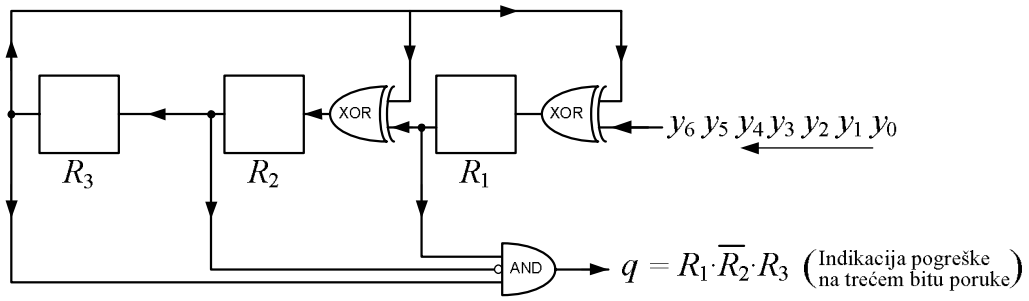
$$S[y(x)] = x^r \cdot y(x) \ mod \ g(x)$$

identičan izrazu za dobivanje zalihosnog dijela cikličke kodne riječi. Drugim riječima, ukoliko u funkciju sindroma $S(\mathbf{x})$ uvrstimo polinom ulazne kodirane poruke $d(x)$, dobit ćemo zalihosni dio kodne riječi $r(x)$. Dakle, sindrom se računa na jednak način kao i zaštitni dio cikličke kodne riječi. Stoga je logično iskoristiti isti sklop i za kodiranje i za dekodiranje.

Kako bi sklop mogao ispraviti pogrešku na način da odredi poziciju pogreške, potrebno je dodati dodatne sklopove. Promotrimo ponovno primjer cikličkog koda K=[7,4,3]. Ukoliko je došlo do pogreške na trećem bitu kodne riječi, tj. ukoliko je vektor pogreške $e(x) = x^3$, sindrom primljene kodne riječi $y(x)$ će biti $S[y(x)] = S[e(x)] = x^2+1$. U skladu s tim zaključujemo da će nakon propuštanja primljene kodne riječi $y(x)$ kroz sklop na slici (Slika 0.20), u registrima R_3, R_2, R_1 ostati bitovi redom 1 0 1. Dakle, ukoliko dođe do pogreške na trećem bitu kodne riječi, logički izraz:

$$q = R_3 \cdot \overline{R_2} \cdot R_1$$

mora dati logičku jedinicu. U skladu s tim na sklop kodera (Slika 0.20) dodajemo dodatni sklop za otkrivanje pogreške na trećem bitu (Slika 0.22) i dobivamo dekoder. Jasno, ovakav dekoder je nepotpun te ga je potrebno nadograditi sklopovima za otkrivanje jednostruke pogreške na drugim bitovima, kao i sklopovima za otkrivanje višestrukih pogreški.



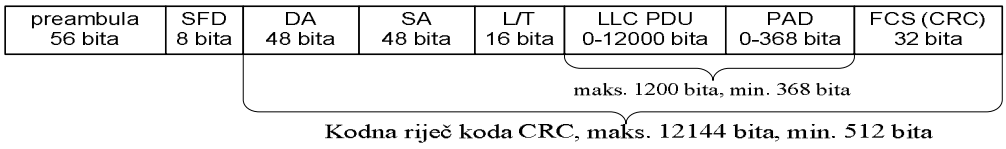
Slika 0.22: Shematski prikaz sklopa za izračunavanje sindroma

1.4.4. Ciklički kôd na primjeru CRC-32

Ciklički kôd u obliku u kojem je objašnjen u prethodnom odjeljku ima vrlo široku primjenu u telekomunikacijama. U mrežama utemeljenim na standardu *Ethernet* (standard IEEE 802.3) koristi se CRC kôd s oznakom IEEE 802.3 CRC-32, koji ima generirajući polinom:

$$g(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1.$$

Budući da $g(x)$ ima stupanj 32, duljina zaštitnog dijela kodne riječi je 32, što stane u točno četiri okteta okvira predviđenog standardom IEEE 802.3 - Slika 0.23.



Slika 0.23: Struktura okvira prema standardu IEEE 802.3

Zaštitni dio CRC se odnosi na dio okvira koji ne uključuje preambulu i SFD (oznaka početka okvira, engl. *Start Frame Delimiter*). Zbog ograničenja na veličinu okvira koju nameće sam standard, kodna riječ ima duljinu od 512 bitova do najviše 12144 bita. Međutim, sam generirajući polinom faktorizira polinom $x^n - 1$, gdje je $n = 2^{32}$, pa kodna riječ teoretski može biti daleko veća od 12144 bita.

Svaka pojedina duljina okvira zapravo odgovara posebnom cikličkom kodu, pa se stoga unutar standarda *Ethernet* aktivno koristi 11633 različitih cikličkih kodova s duljinama $n \in [512,12144]$. Ovakvi kodovi se zovu skraćeni ciklički kodovi (engl. *shortened cyclic codes*).

Određivanje distance koda je mukotrpan zadatak. Numeričkim eksperimentima provedenim u sklopu niza istraživanja (npr. [8]) utvrđene su sljedeće distance ovih kodova - Tablica 0.5.

Tablica 0.5: Distance cikličkih kodova korištenih u standardu IEEE 802.3

duljina koda K - n	distanca $d(K)$
$3007 \le n \le 12144$	4
$301 \le n \le 3006$	5
$204 \le n \le 300$	6
$124 \le n \le 203$	7
$90 \le n \le 123$	8
$67 \le n \le 89$	9

Dakle, CRC kôd koji se koristi u lokalnim mrežama Ethernet može detektirati najmanje trostruku pogrešku bita jednog okvira te otkloniti jednostruku pogrešku bita. No, CRC kôd ima i svojstva otklanjanja snopova pogreški bita.

Kada se unutar neke kodne riječi duljine n dogode pogreške unutar grupe od b susjednih bitova riječi, onda se kaže da se dogodio snop pogreški duljine b (engl. *error burst*). Dakle, snop pogreški je niz bitova od prve do posljednje pogreške. Ukoliko su pogreške nastupile na početku i na kraju kodne riječi, onda se definira ciklički snop pogreški koji se dobije ciklički posmicanjem kodne riječi.

U osnovi bilo koji ciklički kôd s generirajućim polinomom stupnja r može detektirati snop pogreški duljine r ili manje. To znači da CRC-32 može detektirati snop pogreški duljine 32 ili manje. Osim ovoga, ciklički kôd CRC-32 može detektirati i dvostruke snopove pogreški duljine b. Dvostruki snopovi pogreški duljine b se sastoje od dva snopa pogreški najveće duljine b.

Uočimo jednu važnu činjenicu vezanu uz odnos kodera informacije i kodera kanala. Ako razmatramo podatkovni sloj mreže utemeljene na standardu *Ethernet*, onda uočavamo da postupak formiranja cjelokupnog okvira odgovara funkciji koju u općem modelu komunikacijskog sustava obavlja koder informacije. Zapravo, koder informacije kao poruku daje cjelokupan ethernetski okvir u kojem još nije izračunata ciklička zaštita. Nakon toga, koder kanala izračunava cikličku zaštitu (CRC) i stavlja je na predviđeno mjesto. Time cjelokupan okvir postaje jedna kodna riječ.

U općem slučaju koder kanala ne mora zaštititi cjelokupnu poruku, nego samo onaj njen dio koji je iz tehničkih razloga potrebno zaštititi. Primjer je paket mrežnog protokola IP (engl. *Internet Protocol*). U zaglavlju paketa IP-a postoji 16-bitna zaštita zaglavlja paketa, iako se kodnom riječi smatra cjelokupan paket. Stoga u ovom slučaju koder kanala štiti samo zaglavlje paketa.

1.5. Reference

[1] Hamming, R.W., "Error Detecting and Error Correcting Codes," *Bell Systems Technical Journal*, vol. 26, pp. 147-160, April 1950.

[2] R. Hill, *A First Course in Coding Theory*, Oxford University Press, ISBN: 0198538030, SAD, 1990.

[3] M. J. E. Golay, "Notes on Digital Coding," *Proceedings of* IEEE, vol 37, pp. 657, 1949.

[4] Gallager, R.G., "Information Theory and Reliable Communication," *John Wiley & Sons, Inc.*, ISBN: W-471-29048-3, SAD, 1968.

[5] Sinković, V., "Informacija, simbolika, semantika," *Školska knjiga Zagreb*, ISBN: 953-0-30739-X, Zagreb, 1997.

[6] T.R.N. Rao and E. Fujiwara, "Error-Control Coding for Computer Systems", *Prentice-Hall International Editions*, ISBN: 0-13-284068-5, New Jersy, 1989.

[7] ANSI/IEEE Standard for Local Area Networks, "Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications," 1984.

[8] Fujiwara, T., Kasami, T., Lin, S., "Error Detecting Capabilities of the Shortened Hamming Codes Adopted for Error Detection in IEEE Standard 802.3," *IEEE Transactions on Communications*, vol. 37, No. 9, pp. 986-989, September 1989.