nedeljainformatike v8.0

Čarobnjaci i greške u porukama

Momčilo Mrkaić

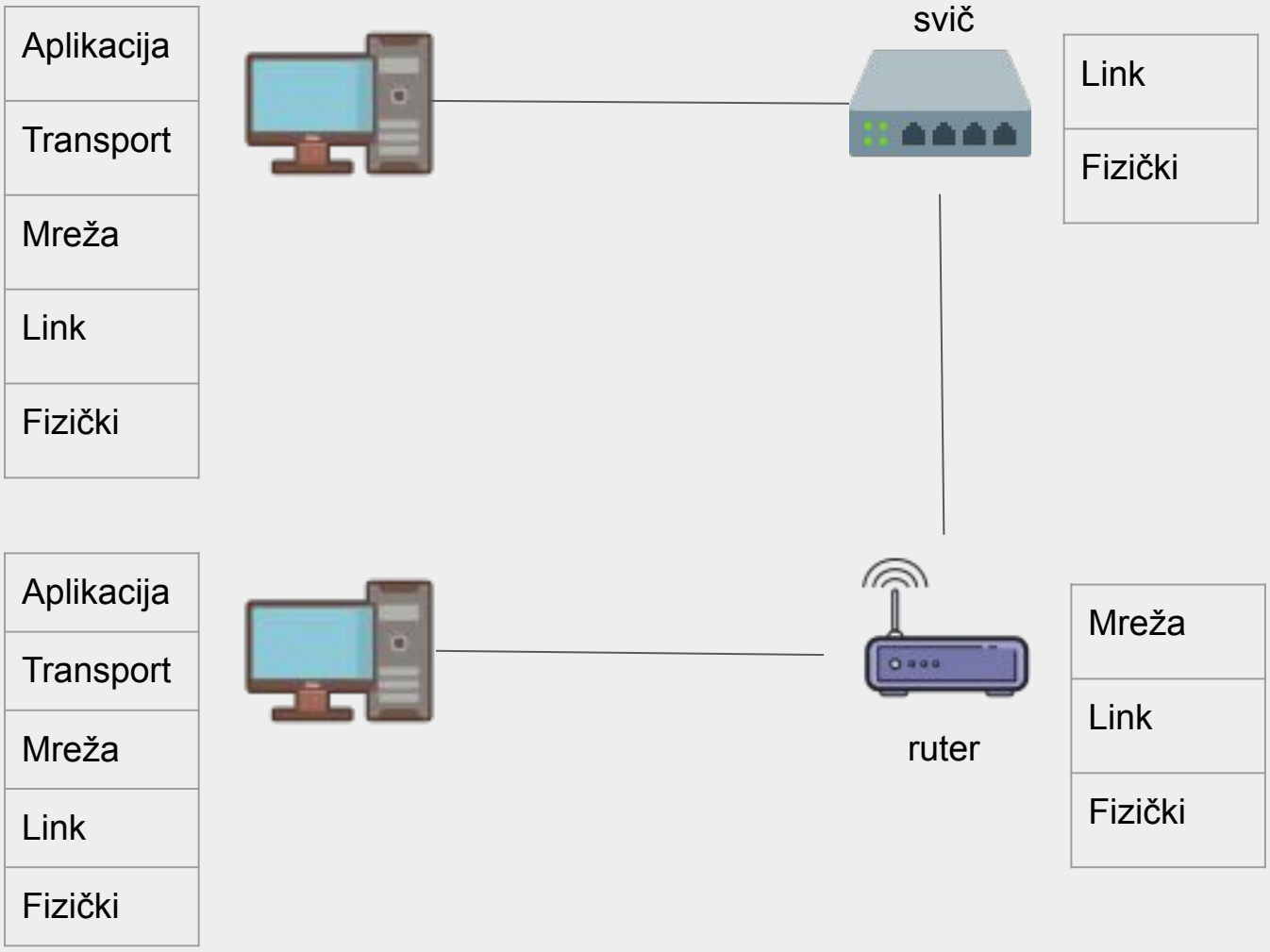
Matematička gimnazija

19. 05. 2023.

Agenda

1. Motivacija
2. Čarobnjaci zatvorenici
3. Ispravljanje i detekcija grešaka

Kompjuterske mreže



Korupcija memorije

- Navikli smo da na našu memoriju gledamo logički, ali sve što čuvamo se zapravo nalazi na nekom fizičkom uređaju na kom može doći do grešaka

Agenda

1. Motivacija
2. Čarobnjaci zatvorenici
3. Ispravljanje i detekcija grešaka

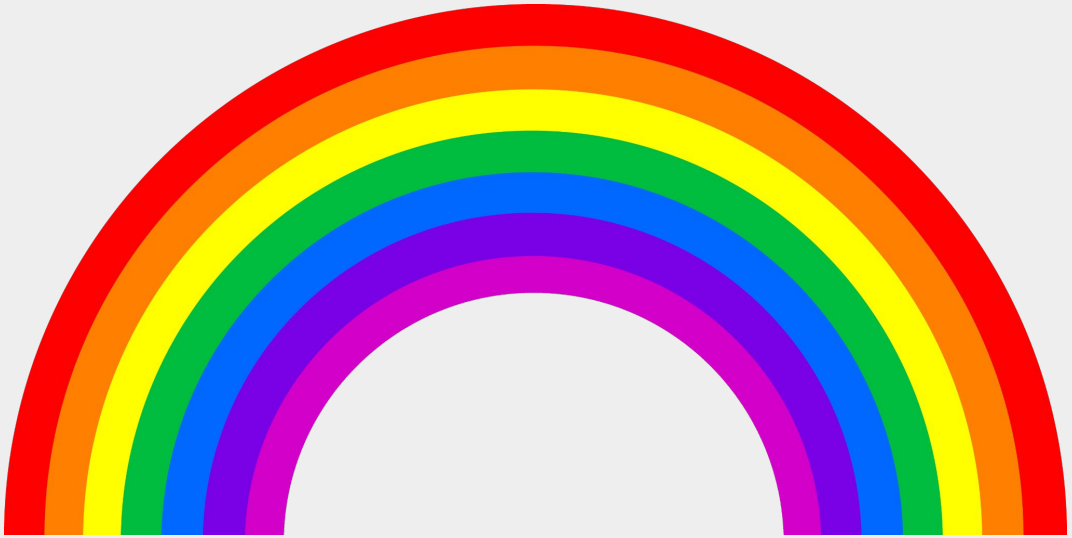
Čarobnjaci zatvorenici

- Kralj je zatvorio 100 čarobnjaka. Pošto čarobnjaci toliko vole da nose šešire kralj je svakome stavio crni ili beli šešir na glavu. Čarobnjak ne može da vidi svoj šešir ali vidi sve ostale. Kralj je odlučio da da proziva čarobnjake jednog po jednog i svaki čarobnjak pogadja da li nosi zeleni ili plavi šešir. Ako pogodi kralj će ga poštedeti, ali ako pogreši osuđen je na smrt. Kojom strategiom čarobnjaci mogu da spasu najviše ljudi?

- Šta ako kralj odluči da im stavi i crveni šešir?

Teža verzija zagonetke

- Kralj je zatvorio 7 čarobnjaka i stavio je na njih 7 šešira, tako da je svaki u jednoj od duginih boja. Čarobnjaci pišu na papir koje boje je njihov šešir (tako da drugi ne znaju šta je napisao). Ako bar jedan čarobnjak pogodi koje boje je njegov šešir kralj će ih osloboditi, u suprotnom su svi osuđeni na smrt. Da li postoji strategija tako da uvek bar jedan pogodi boju svog šešira?

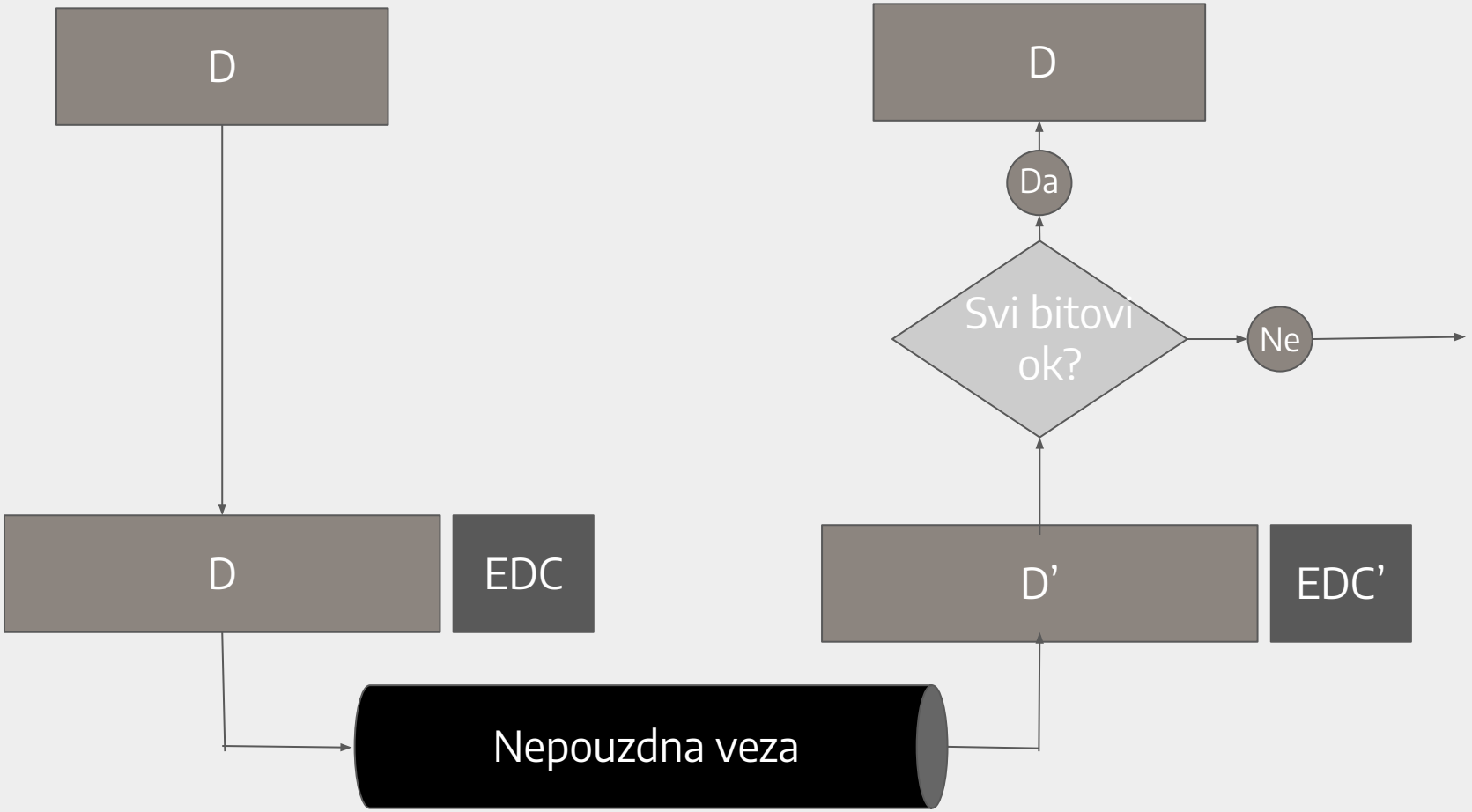


Agenda

1. Motivacija
2. Čarobnjaci zatvorenici
3. Ispravljanje i detekcija grešaka

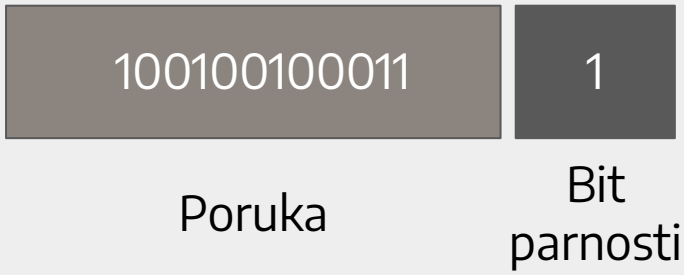
Glavna ideja

- EDC: *error detection-correction* bitovi
- D: naša poruka



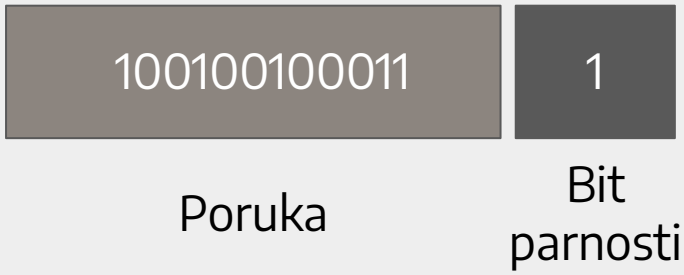
Proveravanje parnosti

- Najjednostavnija forma detekcije greški je dodavanje jednog bita parnosti
- Dodajemo jedan bit tako da je ukupan broj jedinica u poruci paran
- Ovo nam omogućava da detektujemo ako se desio neparan broj grešaka



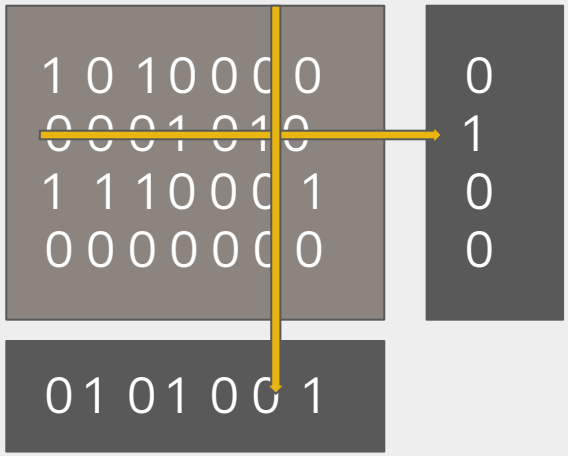
Mane proveravanja parnosti

- U praksi se greške ne dešavaju nezavisno
- Uglavnom se nekoliko bitova za redom promeni i sa ovom metodom ćemo moći da detektujemo tek oko 50% grešaka
- Jasno nam je da nam trebaju složeniji i sigurniji algoritmi



Generalizovano proveravanje parnosti

- Pre nego što pređemo na složenije algoritme, razmotrićemo 2D proveravanje parnosti koje će nas uvesti i u ispravljanje poruka



Parnost u redu

Parnost u koloni

CRC - Cyclic Redundacy Check

- D = Hi! = 0001000 01101001 00100001
- Hi! - 0001000 01101001 00100001 00000000 00000000 = 311001415680
- 311001415680 mod 65521 = 26769
- EDC = 65521 - 26769 = 38752
- 65521 se naziva generator

- D-EDC = 0001000 01101001 00100001 10010111 01100000 = 311001454432
- 311001454432 mod 65521 = 0

Problem sa naivnim CRC-em

- T = ‘o’ = 01101001
- E = 00000110
- R = ‘i’ = 0110111

- T = ‘i’ = 0110111
- E = 11111010
- R = ‘o’ = 01101001

R - primljena poruka, T - poslata poruka, E- greška

- R = E + T
- Kakve greške možemo da detektujemo zavisi od generatora
- Želimo da odredimo generator u zavisnosti od paterna greške koji se javljaju pri slanju poruke preko neke veze
- Imamo problem jer nam greška ne odgovara bitovima koji su promenjeni

CRC i polinomi

- Hi! - 0001000 01101001 00100001 00000000 00000000 = 311001415680
- x^38+x^35+x^30+x^29+x^27+x^24+x^21+x^16

- (x^38+x^35+x^30+x^29+x^27+x^24+x^21+x^16) mod (x^16+x^12+x^5+1)
- 6x^15+x^13-2x^11+3x^8+x^7+x^6-x^5-x^4+3x^3+x^2+2x+1
- Kako da konvertujemo ovaj polinom u binarni zapis (EDC)?

Konačno polje

- Navikli smo da radimo algebra na polju realnih brojeva
- Algebarska struktura polje ispunjava sledeće aksioma: asocijativnost, komutativnost, postoje identiteti i inverzi, distributivnost
- U bilo kom polju algebra na koju smo navikli i dalje važi

- Da bismo dobili ostatak koji možemo da vratimo u binarni zapis prebacićmo se u konačno polje F={0,1} i definisaćemo operacije sabiranja i množenja ovako:

0+0=0	0*0=0
0+1=1	0*1=0
1+0=1	1*0=0
1+1=0	1*1=1

Koje greške možemo da detektujemo

- Hi! - 0001000 01101001 00100001 00110001 11111101
- Ha! - 0001000 01100001 00100001 00110001 11111101
- E(x) = x^27
- Ho! - 0001000 01101111 00100001 00110001 11111101
- E(x) = x^26+x^25

- Bez obzira da li su bitovi 0 ili 1 greška nam govori koji bitovi su promenjeni
- To nam sada omogućava da matematički dokažemo koje greške ćemo moći da detektujemo
- CRC neće detektovati greške samo ako generator deli E(x)
- Na primer ako znamo da se greška dešava na samo jednom bitu, treba nam generator od bar 2 člana
- CRC je veoma dobar u detekovanju grešaka u nizu E(x)=x^i(x^k+...+1)
- https://users.ece.cmu.edu/~koopman/crc/

Hamingov kod

- Sećamo se ispravljanja grešaka sa 2D proveravanjem parnosti
- Da li možemo pametnije da pronađemo gde je greška?

1	1	0	1
0	1	0	0
1	1	0	1
1	0	1	1

Ričard Haming

Hvala na pažnji!

Pitanja?