SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Antonio Kovačić

DNA KRIPTOGRAFIJA

Diplomski rad

Voditelj rada:
prof. dr. sc. Andrej Dujella

Zagreb, srpanj, 2014.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Ovaj diplomski rad posvećujem svojim roditeljima, sestri, braći, prijateljima, mentoru, profesorima, kolegama s faksa, kao i svim ljudima koji su doprinijeli kako mom intelektualnom rastu, tako i mom rastu kao cjelovite osobe.

Sadržaj

Uvod

U današnje vrijeme svjedoci smo nagloga porasta razmjene podataka. Naglim napretkom današnjih računala, javila se potreba za povećanjem sigurnosti, odnosno zaštite podataka, koji putuju preko komunikacijskog kanala. Današnji kriptosustavi omogućuju siguran prijenos takvih podataka, a ključ njihovog razbijanja zapravo leži u faktoriziranju nekog *velikog* broja (na primjer RSA kriptosustav). Na današnjim računalima, takav problem nije lako riješiv - pa su ti sustavi još uvijek sigurni. Razvojem novih teoretskih modela računala - koji se pokušavaju i u praksi realizirati - uočeno je da problem faktorizacije neće biti više takav problem. Primjer jednog takvog računala je kvantno računalo za kojeg postoji algoritam (*Shorov algoritam*) koji faktorizira broj u polinomnom vremenu. Time se javila potreba za osmišljanjem novih teoretskih modela računala - odnosno kriptosustava - koji bi bili otporni na kvantno izračunavanje - ne bi se mogli probiti uporabom kvantnog računala u nekom razumnom vremenu. Takve kriptosustave ćemo zvati *kvantno rezistentnima*. Tema ovog diplomskog rada biti će DNA kriptografija. Kratko rečeno, radi se o teoretskom modelu kriptografskog sustava koji pomoću DNA izračunavanja šifrira podatke. Prednost takvog sustava jest upravo što je kvantno rezistentan.

U ovom radu najprije ćemo se ukratko upoznati s pojmom DNA računala, odnosno DNA izračunavanja, složenosti DNA računala te algoritmom za enkripciju, odnosno dekripciju podataka pomoću DNA računala.

Poglavlje 1

DNA računalo

1.1 DNA izračunljivost

DNA stroj, kao ni DNA izračunavanje nećemo striktno definirati već će definicija biti opisna - u definiciji ćemo reći koje operacije DNA stroj može izvršavati, i što pri tome mora biti zadovoljeno.
Prije nego definiramo DNA stroj moramo definirati neke pojmove iz logike sudova i kombinatorike.

Definicija 1.1.1. ***Alfabet*** *je proizvoljan konačan skup, čije elemente nazivamo* ***simboli****.*
Neka je $n \in \mathbb{N}$ *proizvoljan te A proizvoljan alfabet, proizvoljni element* $w \in A^n$ *zovemo* ***riječ alfabeta A****. Neka su* $s_1, ..., s_n \in A$*, riječ* $w = (s_1, ..., s_n)$ *alfabeta A još zapisujemo kao* $w = s_1 s_2 ... s_n$*. Smatramo da postoji riječ alfabeta A, koju ćemo označavati s* ε*, koja se ne sastoji ni od jednog simbola i zovemo je* ***prazna riječ****. Po dogovoru smatramo da je* $A^0 = \{\varepsilon\}$*. Skup svih riječi alfabeta A označavamo sa* A^**. Neka su* $a = a_1 ... a_m$*, te,* $b = b_1 ... b_k \in A^*$*, kažemo da je riječ* $c \in A^*$ *nastala* ***konkatenacijom*** *riječi a i b ako vrijedi* $c = ab = a_1 ... a_m b_1 ... b_k$*. Kažemo da je riječ* $c \in A^*$ *podriječ riječi* $a \in A^*$*, ako postoje riječi*

$b, d \in A^*$ *tako da je* $a = bcd$. ***Duljina riječi*** *se definira kao funkcija* $d : A^* \to \mathbb{N}$ *sa:*

$$d(\varepsilon) := 0$$
$$d(wa) := d(w) + 1$$

Definicija 1.1.2. *Neka je S proizvoljan konačan skup, a* $m : S \to \mathbb{N}$ *proizvoljna funkcija.* ***Multiskup M na skupu S*** *je uređeni par* $M = (S, m)$. *Za proizvoljan* $x \in S$, $m(x)$ *zovemo* ***kratnost*** *od x.* ***Kardinalnost multiskupa M*** *(broj elemenata), u oznaci* $|M|$, *se definira kao:*

$$|M| := \sum_{x \in S} m(x)$$

Definicija 1.1.3. ***DNA lanac*** *je proizvoljna riječ alfabeta {A (adenin), G (gvanin), T(timin), C (citozin)}. DNA stroj se sastoji od konstantnog broja konačnih skupova koje nazivamo* ***epruvete****, a čiji su elementi DNA lanci. Za proizvoljnu epruvetu K DNA stroja definiramo multiskup* $MulS(K)$ *kao multiskup svih riječi koje predstavljaju DNA lance sadržane u epruveti K. U DNA stroju su definirane slijedeće instrukcije:*

- *Kopiraj(*K_1,K_2*)* $\to$ *uz pretpostavku da je* $K_2 = \emptyset$ *, kopira* $MulS(K_1)$ *u* $MulS(K_2)$ *time više* K_2 *nije prazan*
- *Spoji(*K_1, K_2, K*)* $\to$ *uz pretpostavku da* $K=\emptyset$*:*

$$MulS(K) = MulS(K_1) \bigcup MulS(K_2)$$

- *Uoči(K)* $\to$ *ispituje je li* $MulS(K) \neq \emptyset$*, ako je rezultat operacije je* $\top$*, inače* $\bot$
- *Odvoji(K,w)* $\to$ *za skup K i riječ w (iz* $MulS(K)$*) izbacuje sve riječi iz K koje kao podriječ ne sadrže riječ w*
- *Izvadi(K,w)* $\to K \backslash Odvoji(K, w) \to$*izbacuje sve riječi iz K koje sadrže w*

- *Odvoji_Pref(K,w) → izbacuje sve riječi iz K koje ne sadrže w kao prefiks*
- *Odvoji_Suff(K,w) → izbacuje sve riječi iz K koje ne sadrže w kao sufiks*
- *Proširi(K) → multiskupu MulS(K) još jednom dodaje elemente od K*
- *Izdvoji_po_duljini(K, l) → iz K izbacuje sve riječi čija je duljina različita od l*
- *Konkatenacija(K) → na slučajan način izvodi operaciju konkatenacije nad riječima iz MulS(K) tako da duljina novonastalih riječi ne bude veća od neke konstante, a vraća multiskup koji sadrži sve riječi nastale tom konkatenacijom. Vjerojatnost nastajanja duljih riječi je veća. Ukoliko MulS(K) prije izvođenja ove operacije nad epruvetom K sadrži veliki broj kopija svake od riječi, tada će MulS(K) nakon izvođenja ove operacije nad epruvetom K sadržavati sve moguće kombinacije elemenata iz K.*
  - ***Biološki komplement** DNA lanca H definiramo kao DNA lanac koja ima jednako znakova kao i H, ali je svaki znak A zamijenjen znakom T, a svaki znak C znakom G i obratno, i označavamo je s $\overline{H}$*
  - *Neka riječ H ima duljinu $n \in 2\mathbb{N}$, tada definiramo **biološki prefiks** riječi H kao biološki komplement riječi sastavljene od prvih $\frac{n}{2}$ znakova iz H, slično definiramo i **biološki sufiks** riječi H kao biološki komplement riječi sastavljene od zadnjih $\frac{n}{2}$ znakova riječi H*
  - *Smatramo da je operacija konkatenacije nad riječima H i J dopuštena ako postoji riječ L takva da je biološki sufiks od H prvih $\frac{n}{2}$ znakova od L, a biološki prefiks od J prvih $\frac{n}{2}$ znakova od L*
- *Izreži(K) → na slučajan način "skraćuje" riječi iz MulS(K) do neke fiksne duljine*
- *Izaberi(K) → na slučajan način iz MulS(K) izabire neku riječ te "generira" novi skup sastavljen od samo te riječi*

Program za DNA stroj definiramo kao konačan niz gornje navedenih instrukcija. U svakom koraku programa se može izvesti točno jedna instrukcija.

Napomena 1.1.4. *Vidimo da se sve ove operacije izvršavaju nad jednom epruvetom u jednom koraku, odnosno multiskupom $MulS(K)$. Što je veća kardinalnost multiskupa $MulS(K)$, to se više operacija na riječima izvrši istovremeno, a u stvarnom svijetu sve te operacije imaju svoje "biokemijske analogone" - biokemijske reakcije. Takvo računalo zapravo možemo interpretirati kao superračunalo s izuzetno velikim brojem procesora. U pozadini svega toga se zapravo krije masivni paralelizam. Memoriju DNA računala zapravo predstavljaju epruvete. Jasno je odakle naziv epruvete.*
Uočimoo da je $MulS(K)$ definiran nad konačnim skupom pa je i on konačan - no vidimo da se on zapravo može proširiti nizom operacija Proširi tako da je njegov kardinalitet izrazito velikog reda (nadeksponencijalnog) , ali u praksi se već sada zaključuje da to neće biti uvijek moguće - naime broj DNA lanaca u epruveti (laboratorijskoj) biti će ograničen volumenom te epruvete.
Uočimo da operacija konkatenacije uključuje vjerojatnosni efekt - vjerojatnost nastajanja duljih riječi konkatenacijom je veća - odnosno dvije riječi iz skupa K koje će se konkatenirati neće biti izabrane na slučajan način - već tako da se pokuša dobiti riječ maksimalne duljine (maksimalna duljina je određena nekom konstantom). Iz toga očito možemo vidjeti da sam ishod DNA računanja nije sasvim siguran - no u praksi se pokazuje (pri sintezi DNA lanaca) da je to moguće - u tu svrhu je i uvedena pretpostavka da će se, ukoliko $MulS(K)$ sadrži velik broj kopija od svake riječi iz K, dobiti svaka moguća konkatenacija riječi iz K.

1.2 O složenosti DNA računala

Kako bi nešto rekli o složenosti DNA računala, najprije ćemo navesti nekoliko osnovnih definicija iz teorije složenosti algoritama, odnosno referencirati se na [3].

Definicija 1.2.1. ***Turingov stroj*** *je uređena sedmorka* $(Q, \Sigma, \Gamma, \delta, q_0, q_{DA}, q_{NE})$*, gdje je redom:*

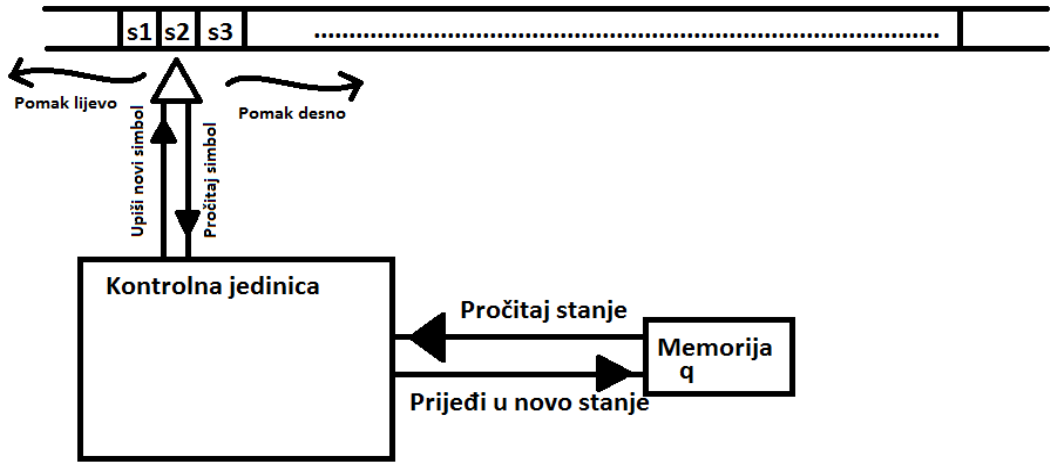
- Q *konačan skup čije elemente nazivamo stanja*
- Σ *je konačan skup, čije elemente nazivamo ulazni simboli, pretpostavljamo da* Σ *ne sadrži "prazan simbol" kojeg označavamo sa* ε
- Γ *je konačan skup kojeg nazivamo alfabet Turingovog stroja, pretpostavljamo da je* $\varepsilon \in \Gamma$*, te* $\Sigma \subset \Gamma$
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, D, S\}$ *koju nazivamo funkcija prijelaza*
- $q_0 \in Q$ *nazivamo početnim stanjem*
- $q_{DA} \in Q$ *nazivamo stanjem prihvaćanja*
- $q_{NE} \in Q$ *nazivamo stanjem odbijanja, te* $q_{NE} \neq q_{DA}$

Napomena 1.2.2. *(Opis rada Turingovog stroja)*
Turingov stroj zapravo ima četiri glavna dijela: kontrolnu jedinicu (koja zapravo oponaša dijelovanje funkcije δ*), beskonačnu traku, neograničenu s lijeve i desne strane, takvu da se u svakom trenutku rada stroja na jednom registru trake nalazi točno jedan simbol, memoriju u kojoj se pamti trenutačno stanje stroja te glavu za čitanje koja se u jednom koraku rada stroja može pomicati za točno jedno mjesto na traci: desno, lijevo ili ostati na istom simbolu. Glava se na početku nalazi na nekom mjestu na traci (unaprijed definiranom), zatim čita simbol. Pročitani simbol, u paru s trenutnim stanjem stroja "se šalje" u kontrolnu*

jedinicu. Glava nakon toga, najprije zamijeni pročitani simbol nekim drugim simbolom, stroj prelazi u novo stanje, a glava se pomiče na drugi registar (L (lijevo), D (desno)) ili ostaje na istom mjestu (S).

Vidimo da opisani Turingov stroj može stati u dva završna stanja q_{DA}, odnosno q_{NE}, takav Turingov stroj se naziva još odlučitelj. Uočimo da Turingov stroj ne mora nužno uvijek stati. Shematski prikaz Turingovog stroja možete vidjeti na slici 1.1.

Nedeterministički Turingov stroj se definira na analogan način, samo što je funkcija prijelaza definirana sa: $\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{L, D, S\})$.



Slika 1.1: Shematski prikaz Turingovog stroja

Definicija 1.2.3. *Neka su $f, g : \mathbb{N} \to \mathbb{R}^+$ dvije funkcije. Kažemo da je funkcija g **asimptotska gornja međa** za funkciju f ako postoje $c > 0$ i $n_0 \in \mathbb{N}$ tako da za svaki $n \geq n_0$ vrijedi*

$$f(n) \leq cg(n)$$

Činjenicu da je g asimptotska međa od f označavamo sa $f(n) = O(g(n))$.

Osnovne definicije (što je alfabet logike sudova, interpretacija, ispunjivost formule, konjunktivna, odnosno, disjunktivna normalna forma i tako dalje) se mogu naći u [4].
Više o Turingovom stroju (Turing prepoznatiljivost, Turing odlučivost, univerzalni Turingov stroj, i tako dalje) se može naći u [3]. Pojmovi kao što su klasa $NTIME, TIME$, $PTIME, NP$, te o *polinomnoj reducibilnost* i *NP-potpunosti* također se mogu naći u [3]. Označimo sa SAT skup definiran na idući način:

$$SAT = \{F : F \text{ je ispunjiva formula logike sudova }\}$$

Formulacija *problema SAT* glasi:

Za danu formulu logike sudova F koja je u konjunktivnoj normalnoj formi odrediti vrijedi li $F \in SAT$.

Konjunktivnu normalnu formu koja u svakoj svojoj elementarnoj disjunkciji sadrži točno $k \in \mathbb{N}\backslash\{0\}$ literala nazivamo k-knf. Formulacija problema $k - SAT$ glasi:

Za proizvoljnu formulu F koja je k-knf odrediti je li F ispunjiva.

U [3] se može vidjeti da je problem SAT *NP-potpun* problem, kao i $3 - SAT$. Sljedeći teorem govori zapravo o tome da DNA računala, u pogledu vremenske složenosti, imaju bolja svojstva nego Turnigovi strojevi:

Teorem 1.2.4. *(Lipton) Za svaku konjunktivnu normalnu formu F u kojoj se pojavljuje n propozicionalih varijabli i m klauzula, u $O(m + 1)$ separacija i $O(m)$ spajanja te jednim uočavanjem možemo odlučiti vrijedi li $F \in SAT$*

Bibliografija

[1] M. Borda. *Fundamentals in Information Theory and Coding*. Springer, 2011.

[2] J. Hromkovic and W. M. Oliva. *Algorithmics for Hard Problems*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2nd edition, 2002.

[3] M. Sipser. *Introduction to the Theory of Computation*. International Thomson Publishing, 2nd edition, 2006.

[4] M. Vuković. *Matematička logika*. Element, prvo izdanje edition, 2009.

[5] S. Yan. *Computational Number Theory and Modern Cryptography*. Wiley-HEP information security series. Wiley, 2012.