nedeljainformatike v2.0

Kompresija podataka na DNK molekulima

Petar Veličković

Nedelja informatike v2.5

14. april 2016.

Uvod

- Naša današnja misija: *spasiti svet*! (no pressure)
- Čovečanstvo se samouništava.
- U slučaju kataklizmičnog događaja, izgubiće se svi tragovi naše tehnologije i znanja.
- Cilj: *arhivirati* informacije tako da ostanu dostupne i posle gubitka sve savremene tehnologije.

Tiho brdo™

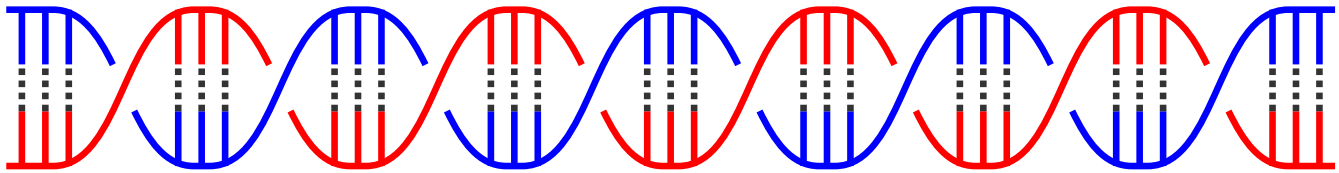
- Kada planeta konačno izađe iz mračnog doba, koja god inteligentna civilizacija se razvije će imati prirodnu težnju da otkrije DNK i nauči da ga pročita.

- $\Longrightarrow$ **čuvajmo informacije na DNK molekulu!** (u nekoj zapuštenoj bolnici, recimo...)

Tiho brdo™

DNK

- **Dezoksiribonukleinska kiselina** (*deoxyribonucleic acid*, skraćeno DNK/*DNA*): molekul koji u sebi sadrži genetičke instrukcije neophodne za razvoj i funkcionisanje gotovo svih živih organizama.
- Sastoji se od dve niti isprepletane u formi **dvostrukog heliksa**:

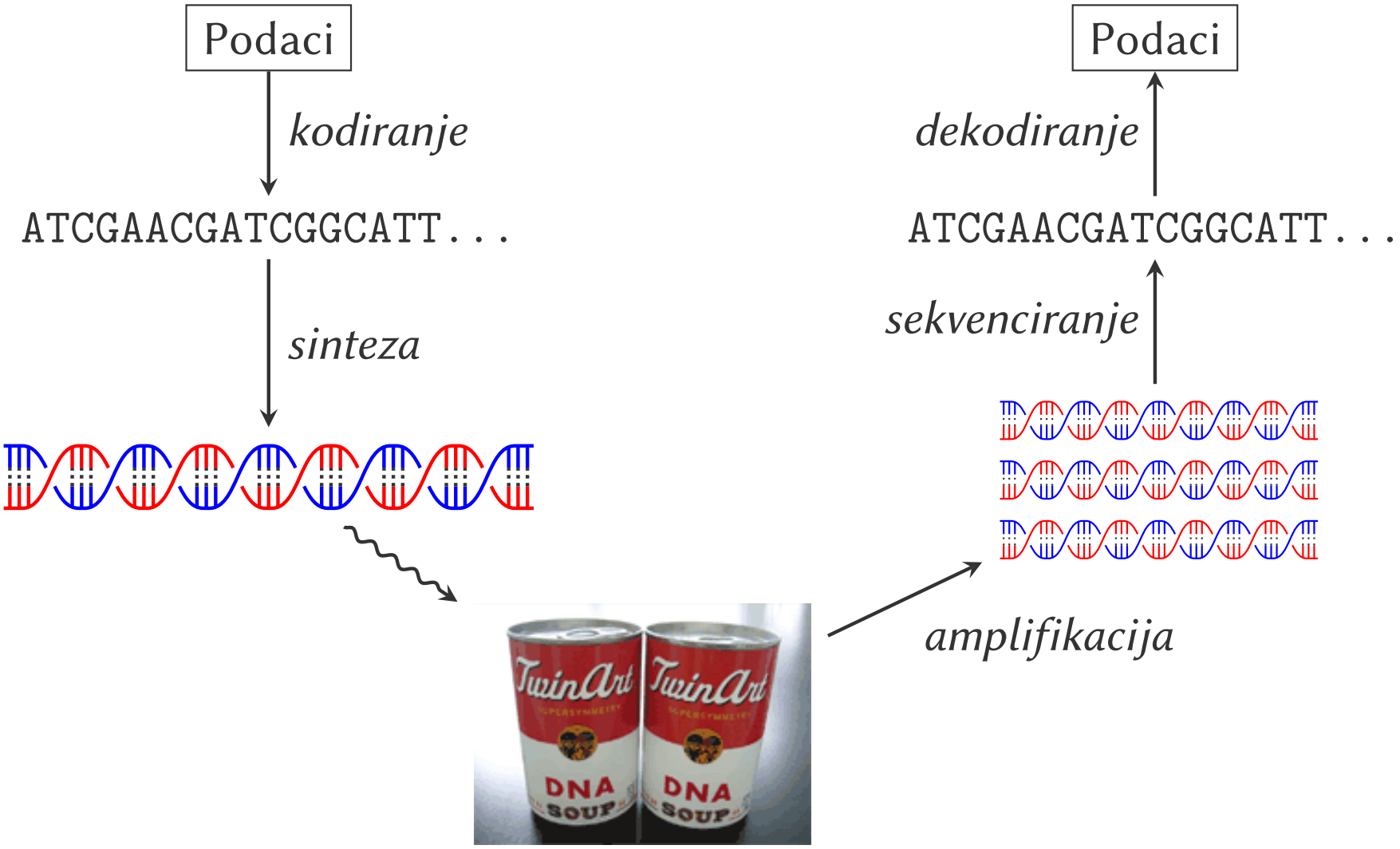


- Niti se sastoje od niza osnovnih baza, tj. **nukleotida**:
  - Adenin (`A`);
  - Timin (`T`);
  - Citozin (`C`);
  - Guanin (`G`).
  - Nukleotidi su uvek **komplementarni** (`A-T`, `C-G`), tako da je za analizu *dovoljno znati jednu nit DNK*.

Informaciona moć

- DNK je proizvod dugotrajne evolucije, u cilju da se *što konciznije* zapiše sav naš genetički kod.
- Samim tim, i pored gorenavedenih katastrofalnih scenarija, predstavljaju odličan način da se efikasno zapiše velika količina informacija!
- **Ceo internet** (1200PB) bi stao u $\sim$ 1kg DNK molekula!

Algoritam


Podaci
kodiranje
ATCGAACGATCGGCATT...
sinteza
Podaci
dekodiranje
ATCGAACGATCGGCATT...
sekvenciranje
amplifikacija
TwinArt
DNA
SOUP
TwinArt
DNA
SOUP

Roadmap

Zadaci postaju teži što smo bliži središnjem "kanalu", tako da će biti razmatrani u sledećem redosledu:

1. **(De)kodiranje** (osnove ustanovio već *Shannon* u 1948.);
2. **Sekvenciranje** (moderni algoritmi ustanovljeni oko 2000.—tehnologija značajno uznapredovala oko 2005.);

(3.) **Sinteza i amplifikacija** (prevashodno posao za *biologe* :));

4. **"DNA Soup"** (otvoren problem koji na različite načine napada pet istraživačkih grupa širom sveta!)

Kodiranje i kompresija

- Sada ćemo ukratko pričati o osnovama **teorije informacija** i njihovim primenama na *kodiranja*—algoritme kojima možemo svakom podatku pripisati kod.

- Fokusiraćemo se na kodiranje u *binarnom* sistemu, koji je bliskiji računarima. Međutim, sve metode su lako primenljive i na DNK kodiranje (*kvarternarni* sistem).

Entropija

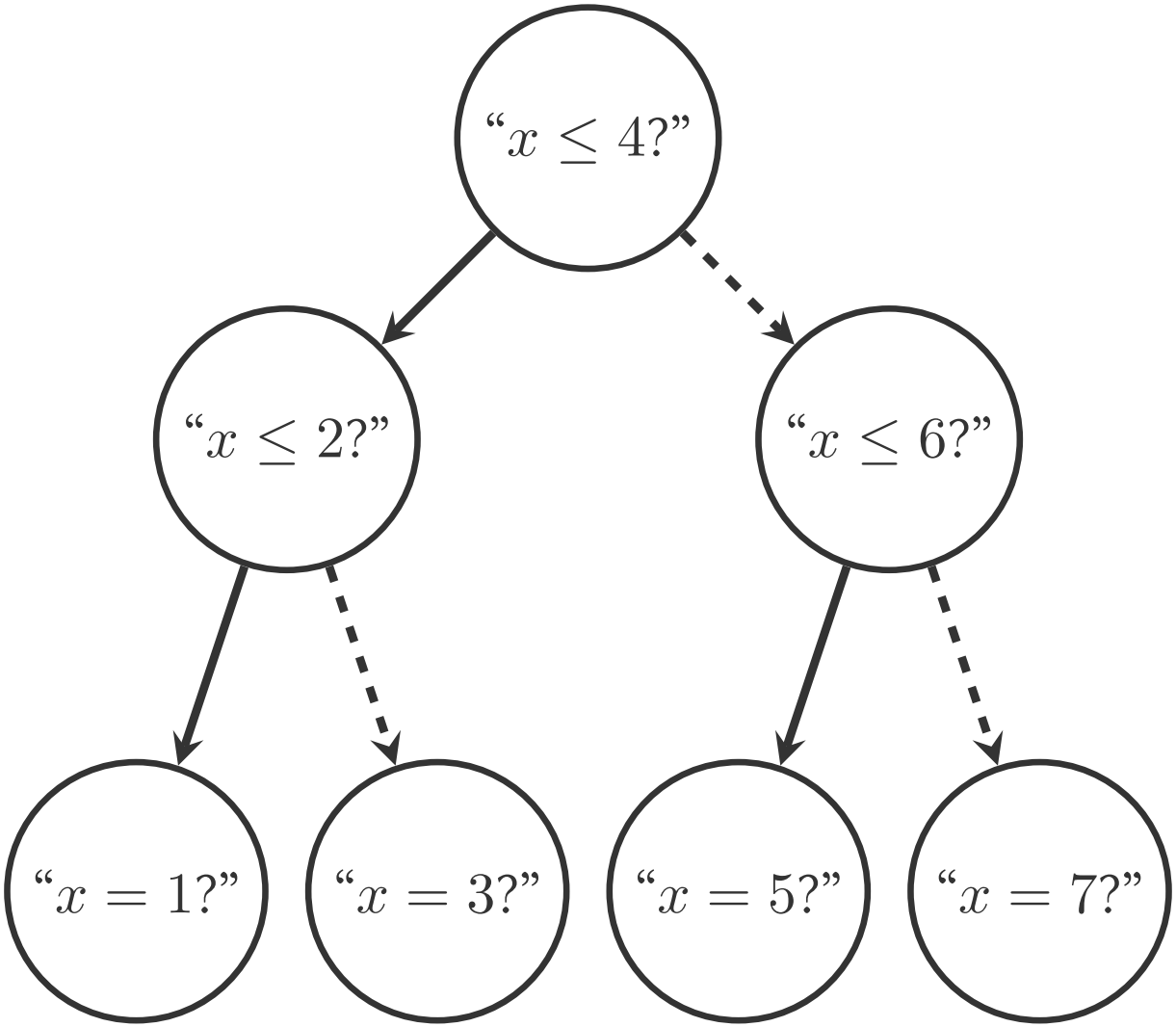
- Zamislio sam prirodan broj $x \in [1, 8]$.

- Koliki je *prosečni* broj da/ne pitanja koja morate da mi postavite da biste sa sigurnošću utvrdili koji broj sam zamislio?

Entropija

- Zamislio sam prirodan broj $x \in [1, 8]$.
- Koliki je *prosečni* broj da/ne pitanja koja morate da mi postavite da biste sa sigurnošću utvrdili koji broj sam zamislio?
- Tri!

Entropija



Entropija

- Sada, ukoliko za svaki “da” odgovor dopišemo 1, a za svaki “ne” odgovor dopišemo 0, dobili smo način da (optimalno) kodiramo brojeve iz $[1, 8]$.
- Na primer, broj 6 bi imao kod 010 (3 bita).
- Generalno, za skup od n elemenata potrebno je $\sim \log_2(n)$ bitova po elementu da bismo ga kodirali.

Optimalno kodiranje

- Zamislio sam prirodan broj $x \in [1, +\infty)$!
- Međutim, sada imate više informacija: zamislio sam 1 sa verovatnoćom $\frac{1}{2}$, 2 sa verovatnoćom $\frac{1}{4}$, $\ldots$, n sa verovatnoćom $\frac{1}{2^n}$,$\ldots$
- Koliki je *prosečni* broj da/ne pitanja koja morate da mi postavite da biste sa sigurnošću utvrdili koji broj sam zamislio?

Optimalno kodiranje

- Zamislio sam prirodan broj $x \in [1, +\infty)$!
- Međutim, sada imate više informacija: zamislio sam 1 sa verovatnoćom $\frac{1}{2}$, 2 sa verovatnoćom $\frac{1}{4}$, $\ldots$, n sa verovatnoćom $\frac{1}{2^n}$,$\ldots$
- Koliki je *prosečni* broj da/ne pitanja koja morate da mi postavite da biste sa sigurnošću utvrdili koji broj sam zamislio?
- $\ldots$Dva???

Optimalno kodiranje

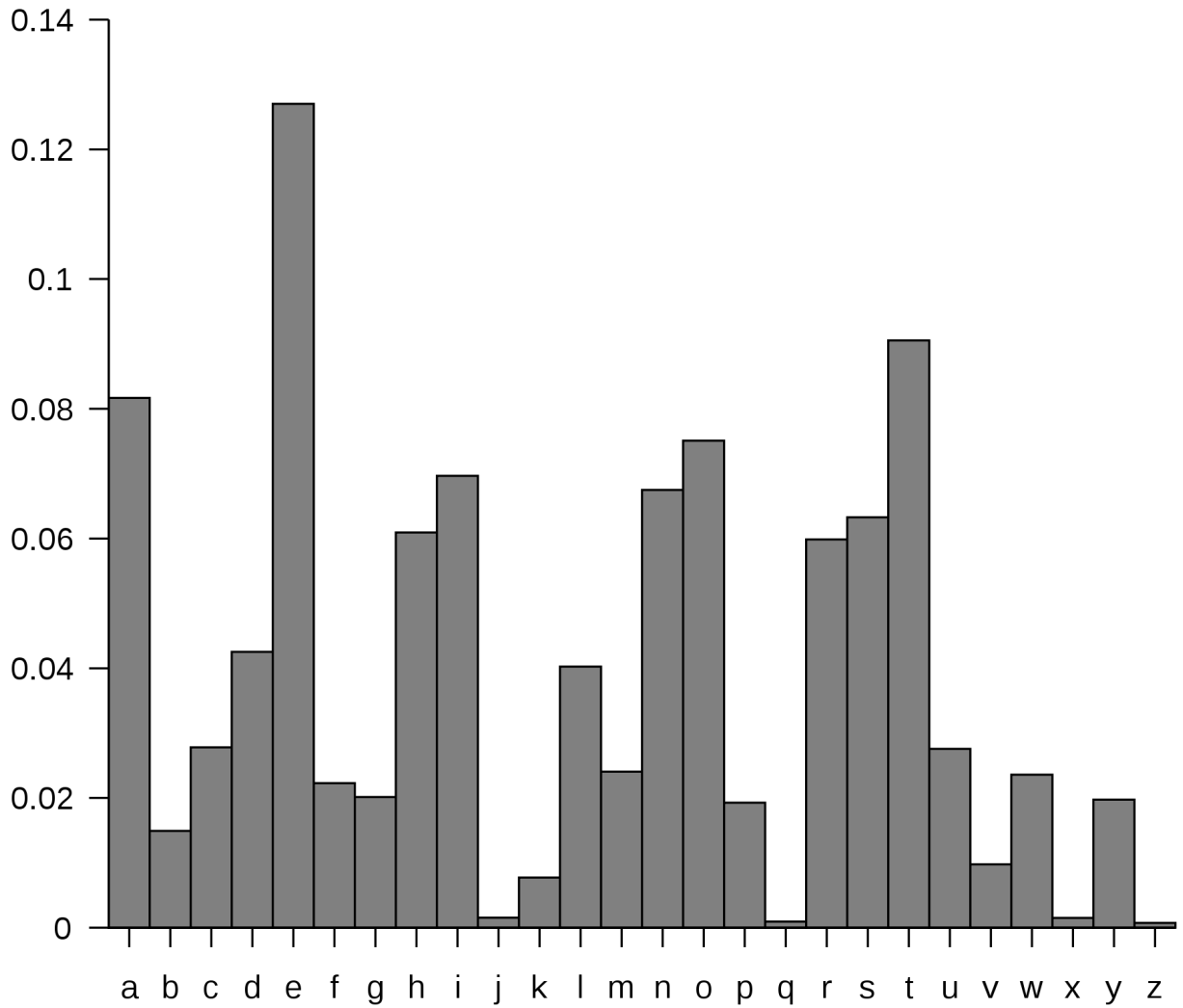
- Pitanja koja postavljamo u optimalnom slučaju su sada:
  - “$x = 1$?”
  - “$x = 2$?”
  - $\ldots$
- Broj 1 (1) će imati sada samo 1 bit, broj 2 (01) će imati dva bita, broj n $(000\ldots001)$ n bitova, itd.
- U proseku:

$$\sum_{k=1}^{+\infty} \frac{k}{2^k} = 2 \text{ bita}$$

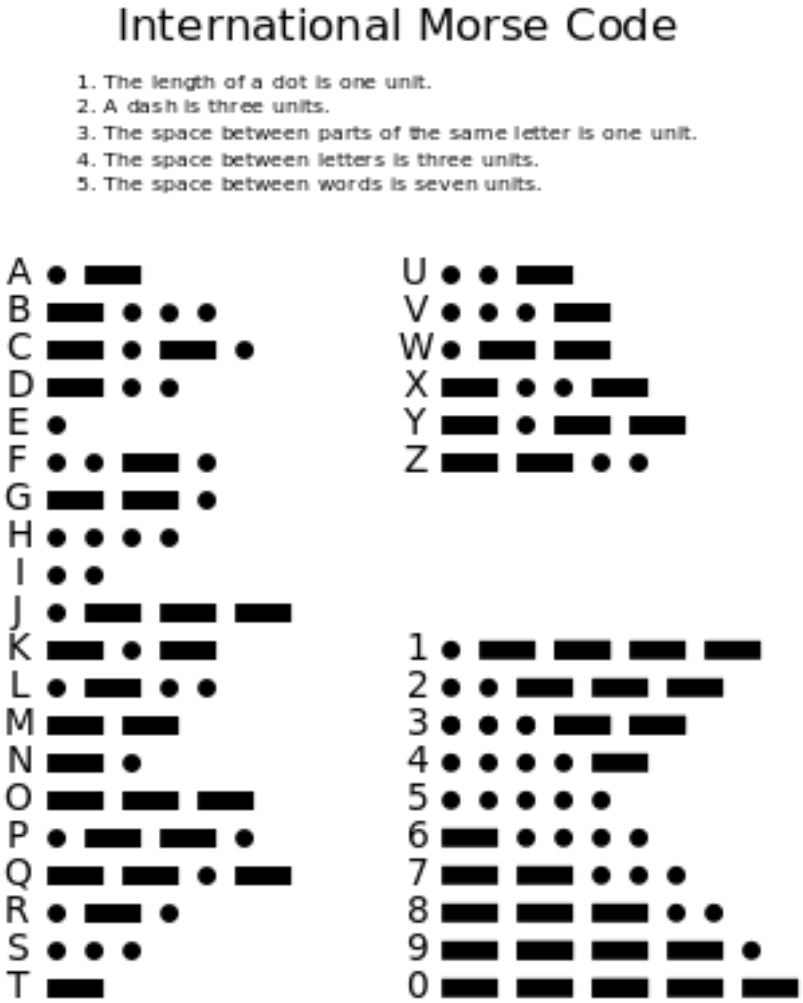
Prefiksni kod

- Ono što je jako lepo kod dve vrste kodova o kojima smo sad pričali je da imaju *prefiksno svojstvo* (ni za koja dva elementa ne važi da je kod jednog prefiks od koda drugog)!
- Ovo nam omogućava da, kada treba kodirati niz ovakvih elemenata, možemo samo poređati njihove kodove jedan za drugim u dugačak *string*, bez potrebe za "stop" kodovima itd.
- **Hafmanovo kodiranje** (*Huffman coding*) je generalan postupak za pravljenje prefiksnih kodova optimalne dužine: *u svakom koraku postaviti pitanje tako da su oba odgovora (skoro) jednako verovatna.*

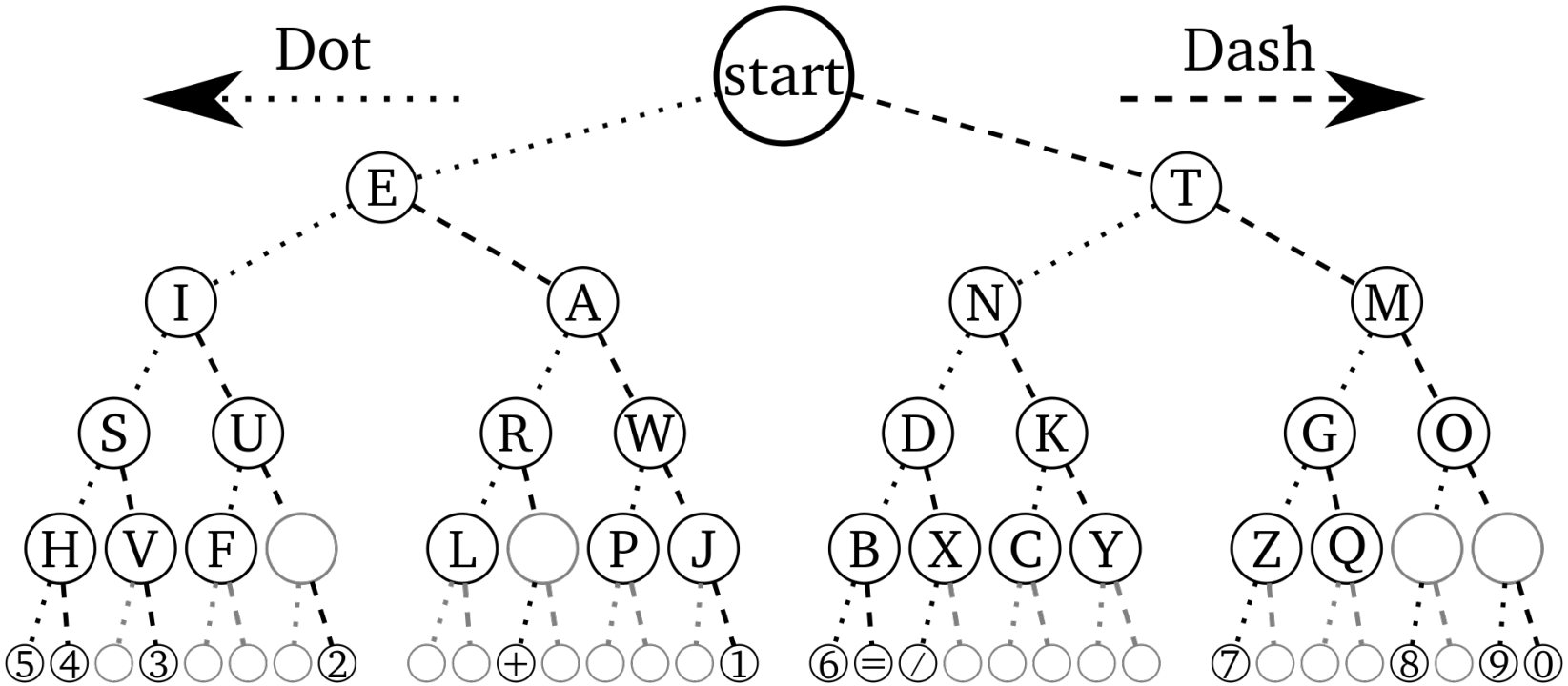
Optimalno kodiranje engleskog jezika



Optimalno kodiranje engleskog jezika



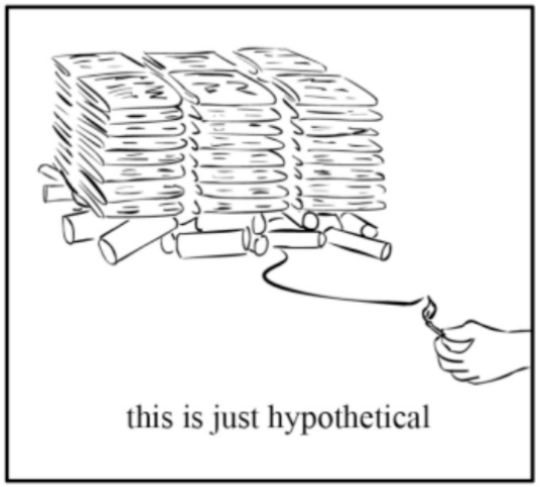
Optimalno kodiranje engleskog jezika

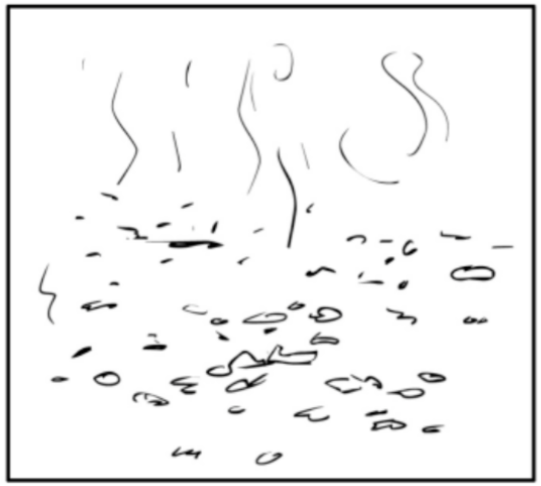


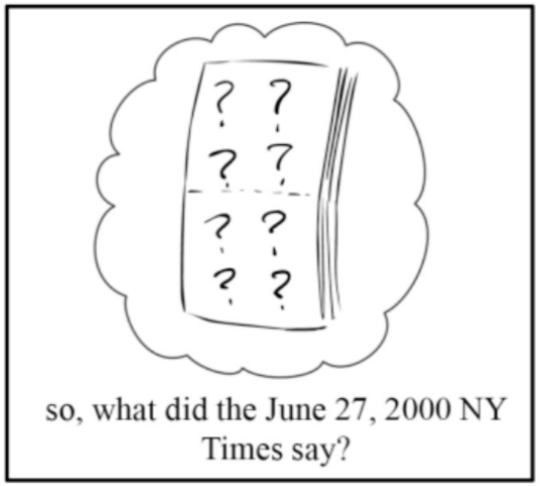
Sekvenciranje

- Sada kada znamo kako optimalno kodirati/dekodirati, želimo da rekonstruišemo **DNK sekvencu** (`ATCG` string) od nekog datog DNK molekula, da bismo mogli da je dekodiramo.
  - Ovo predstavlja problem *sekvenciranja*—u nedavnim godinama smo u mogućnosti da sa relativnom sigurnošću sekvenciramo *ceo ljudski genom*!
- Vrlo je skupo odjednom izvući celu sekvencu sa velikom preciznošću; daleko jednostavnije je napraviti veliki broj ($\sim 10^7$) izvlačenja sitnih podstringova.
- Problem: kako rekonstruisati celu sekvencu koristeći ove sitnije podstringove?

Shotgun sekvenciranje





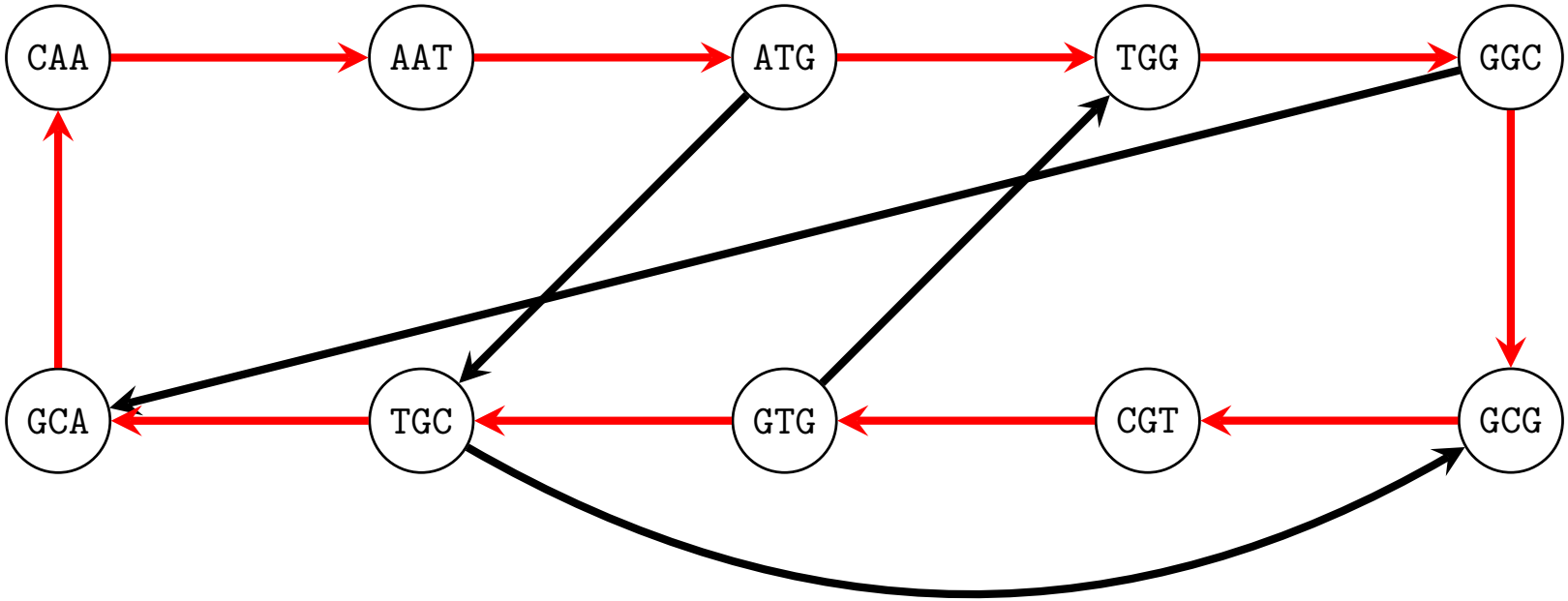


Hamiltonov graf

- Najpre generišemo sve podstringove dužine w od pročitanih podstringova; npr. za string `ATCG` i $w = 2$ generišemo `{"AT", "TC", "CG"}`.
- Prvih $w - 1$ karaktera stringa dužine w nazvaćemo njegovim **prefiksom**, a poslednjih $w - 1$ karaktera njegovim **sufiksom**. npr. prefiks od `"ATCG"` je `"ATC"`, a sufiks je `"TCG"`.
- Napravimo graf takav da svaki uočeni podstring dužine w ima jedan čvor; dva čvora spajamo usmerenom ivicom ukoliko se sufiks prvog poklapa sa prefiksom drugog; npr. `"ATCG"` $\rightarrow$ `"TCGG"`.
- Genom dobijamo iz *Hamiltonovog ciklusa* nad ovim grafom—ciklusa koji obilazi svaki **čvor** tačno jednom.
- Vremenska složenost: $O(n^2 2^n)$, *worst-case*

Hamiltonov graf, primer

Primer Hamiltonovog grafa nad podstringovima:
"ATG", "TGG", "GGC", "GCG", "CGT", "GTG", "TGC", "GCA", "CAA", "AAT".



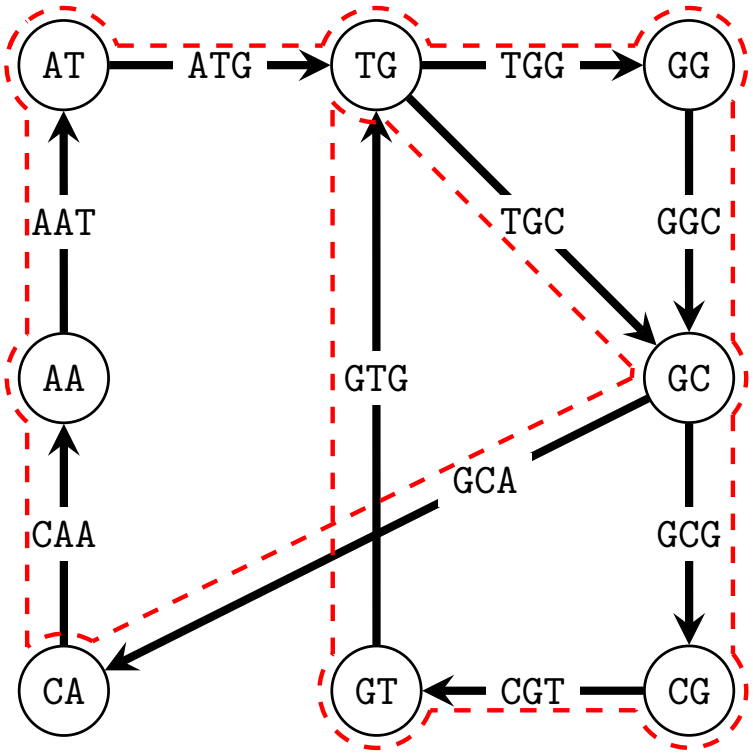
Hamiltonov ciklus daje genom ATGGCGTGCAATG.

de Bruijn-ov graf

- Ponovo generišemo sve podstringove dužine w od pročitanih podstringova.
- Međutim, ovoga puta čvorove grafa vezujemo za **prefikse** i **sufikse** svih ovih stringova.
- Dva čvora u ovom grafu se spajaju ivicom ukoliko postoji podstring kome je prvi čvor prefiks a drugi čvor sufiks; npr. `"ATC"` $\rightarrow$ `"TCG"` ukoliko imamo podstring `"ATCG"`.
- Genom dobijamo iz *Ojlerovog ciklusa* nad ovim grafom—ciklusa koji obilazi svaku **ivicu** tačno jednom.
- Vremenska složenost: $O(|V| + |E|)$.

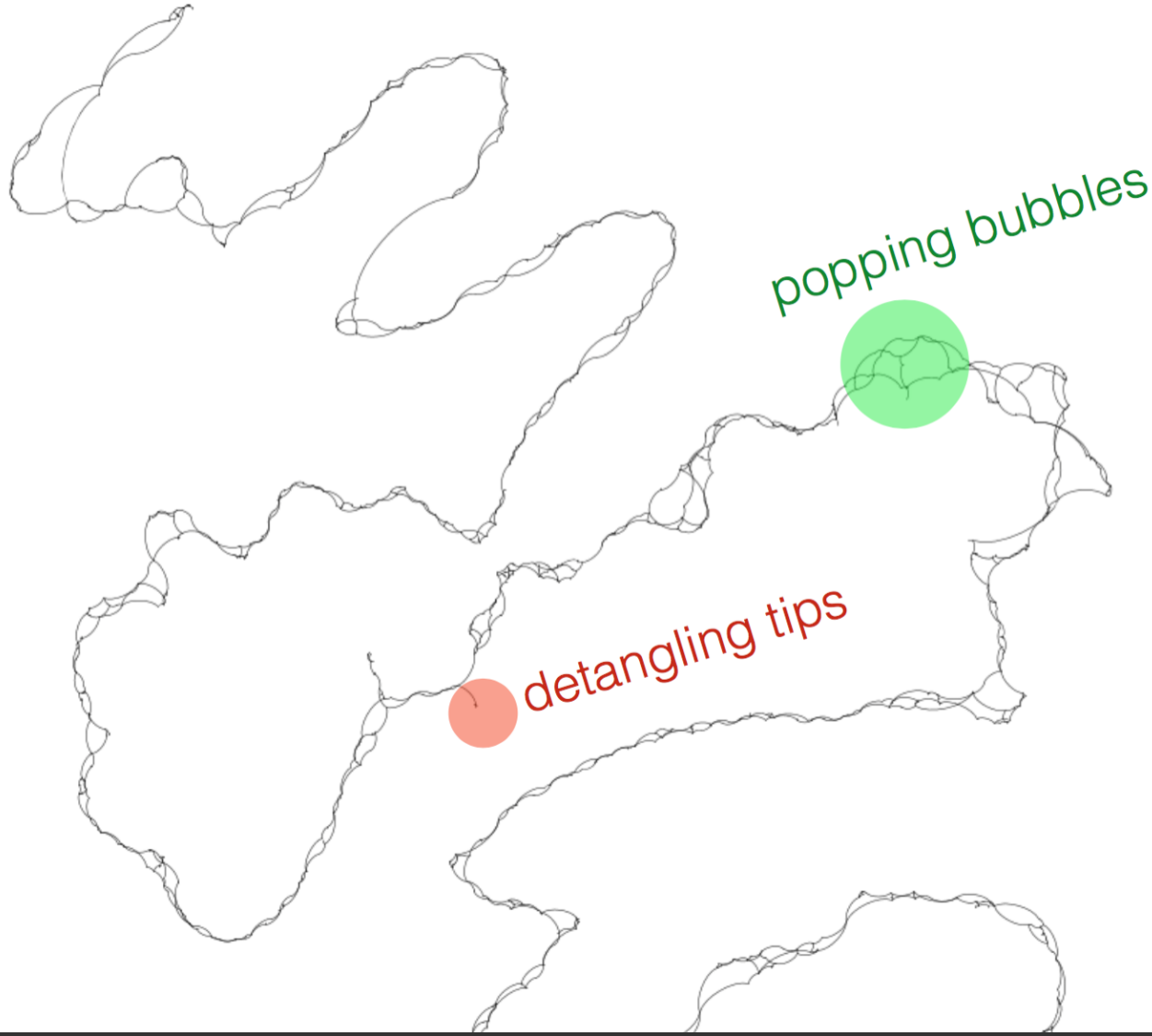
de Bruijn-ov graf, primer

Primer *de Bruijn*-ovog grafa nad podstringovima:
"ATG", "TGG", "GGC", "GCG", "CGT", "GTG", "TGC", "GCA", "CAA", "AAT".



Ojlerov ciklus daje genom ATGGCGTGCAAT.

Konačne modifikacije



Sinteza i amplifikacija

- Kada znamo koje (prethodno kodirane) sekvence želimo da proizvedemo, potrebno je tražiti od laboratorije da je *sintetišu*.
- Trenutno *vrlo skupo*, *sporo* i ograničeno na do stringove dužine ~ 200.
- Sa strane dekodiranja, pre sekvenciranja obično je korisno veštački povećati koncentraciju DNK molekula (ovaj proces se zove *amplifikacija*)
- Ovo zahteva samo uvođenje određenih enzima—*vrlo jeftino*!

Prenosni kanal

- Sada ćemo ukratko definisati jedan način na koji možemo preneti kodirane informacije za potrebe kasnijeg dekodiranja, koristeći DNK kao medijum.
- Ovo je tema *aktivnog istraživanja*; sasvim je moguće da će konačne šeme biti sasvim drugačije od ovoga!
- *Osnovna ideja*: Trenutno najskuplji element algoritma je *sinteza*—šema mora najviše da se povinuje baš tom ograničenju!

Paketno kodiranje

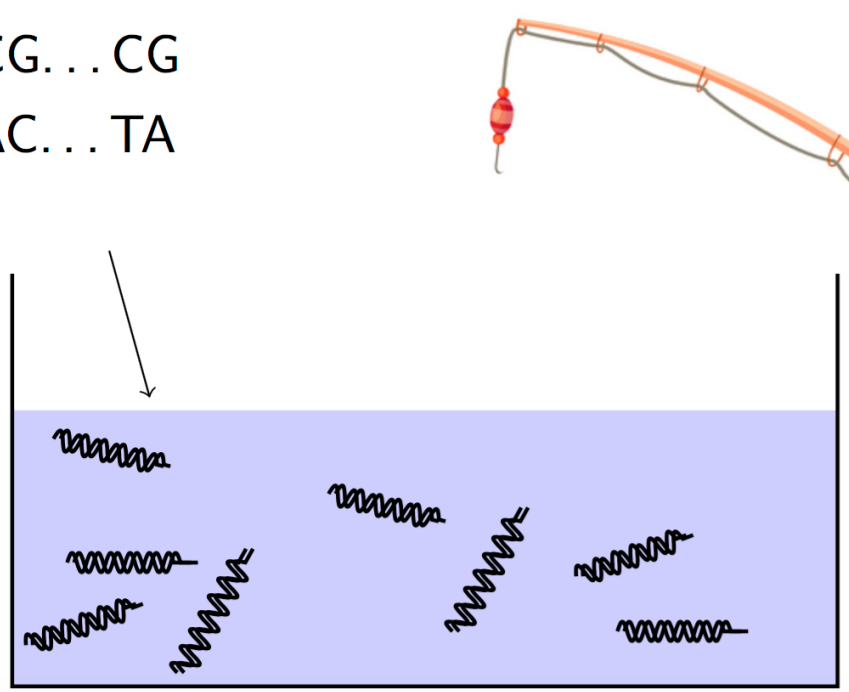
- Pošto praktično ne možemo (trenutno) sintetisati DNK molekule duže od 200 baza, a sasvim je moguće da će naši podaci zahtevati kodove duže od 200...
- $\Longrightarrow$ moramo da podelimo te kodove na *pakete* koje ćemo odvojeno sintetisati.
- Npr. kod `GATTACAT` podeliti na `GATT` i `ACAT`.
- Sličan princip kao i kod interneta (TCP/IP/...), međutim...

“DNK supa”

Redosled paketa se gubi u “supi”!

$N \simeq 200$

1. ACGCA. . . AT
2. GGACT. . . TG
3. ATCTG. . . GA
4. TTACG. . . CG
5. GCTAC. . . TA
6. . . .



Indeksiranje

- Jedno od “rešenja” ovog problema je da prikačimo dodatne informacije na pakete, koje će pomoći da *rekonstruišemo* potpun kod.
- Najjednostavnije ovakvo rešenje će *indeksirati pakete*, i na početak svih sintetisanih paketa odvojiti određen broj simbola za kodiranje indeksa:

- Ovo bi bilo odlično da živimo u idealizovanom svetu. Međutim…

Mutacije

- DNK je konstantno podložna *mutacijama*!
- Neophodno je odvojiti dodatne simbole da bi se proverilo da su podaci zadržali *integritet*, i pojačati otpornost molekula...
- Indeksu je neophodna **savršena zaštita!**
- Prevashodno (bio)inženjerski a ne informatički problem!
- Skorija istraživanja pokušavaju da oblože DNK molekul *staklom*; procenjena izdržljivost oko 10000 godina.

(Etička) zaključna pitanja

- Ako je “čovečanstvo v1.0” toliko loše, da li bismo uopšte trebali da ostavljamo naše informacije dostupnim?
- Da li će budući inteligentni život otkriti paketno kodiranje?
- *Ko kaže da smo v1.0?*
- Hvala vam!