LONGEST COMMON SUBSEQUENCE KORIŠĆENJEM GENETSKOG I BEAM SEARCH ALGORITMA

Projekat u okviru predmeta računarska inteligencija

UVOD

- **Jedan od najizučavanijih problema u računarskim naukama u posednjih 30-ak godina**
- **Igra bitnu ulogu u poredjenju sekvenci podataka.**
- **Potencijalne primene u prepoznavanju uzoraka, obradi i kompresiji teksta i podataka i molekularnoj biologiji**
- **Može se posmatrati kao mera bliskosti k sekvenci**
- **Koristimo termin sekvenca a ne niska jer se ovaj problem odnosi na nizove proizvoljnih tipova i ne koristimo termin niz jer će se on upotrebljavati u drugom kontekstu**

OPIS PROBLEMA

- **Zadata proizvoljna azbuka Σ i niz sekvenci S_i**
- **T je podsekvenca od S ukoliko se T može dobiti iz S brisanjem nekih elemenata iz S**
- **<u>Problem</u>: Ako su nam date sekvence S_i, $1 \leq i \leq k$, na nekoj fiksnoj azbuci Σ, pronadji sekvencu T koja je podsekvenca S_i za svako $i \in \{1, 2, \ldots, k\}$**

BRUTE FORCE ALGORITAM

- **Koristimo tehniku dinamičkog programiranja**
- **Složenost $O(n^k)$ gde je n dužina najduže sekvence a k broj sekvenci**
- **Prvo formiramo DP tablicu**
- **Podsetimo se rekurzivne funkcije koja to radi za 2 sekvence:**

$$f(n_1, n_2) = \begin{cases} 0 & \text{ako } n_1 = 0 \vee n_2 = 0 \\ f(n_1 - 1, n_2 - 1) + 1 & \text{ako } a_1[n_1 - 1] = a_2[n_2 - 1] \\ \max\left(f(n_1 - 1, n_2), f(n_1, n_2 - 1)\right) & \text{inače} \end{cases}$$

- **Ovo jednostavno proširujemo na funkciju koja prima proizvoljno mnogo**

$$f(n_1, n_2, ..., n_k) = \begin{cases} 0, & \text{ako je } n_1 = 0 \vee n_2 = 0 \vee \cdots \vee n_k = 0 \\ f(n_1 - 1, n_2 - 1, ..., n_k - 1) + 1, & \text{ako je } a_1[n_1 - 1] = a_2[n_2 - 1] = \cdots = a_k[n_k - 1] \\ \max\left(f(n_1 - 1, n_2, ..., n_k), f(n_1, n_2 - 1, ..., n_k), ..., f(n_1, n_2, ..., n_k - 1)\right), & \text{inače} \end{cases}$$

- **Sada možemo da kreiramo k-dimenzionu DP tablicu**
- **Koristimo je za enumeraciju svih sekvenci za računanje puta od LCS**
- **Algoritam:**

Algoritam 1 Formiranje DP tablice

Ulaz: niz sekvenci $a_1, a_2, ..., a_k$ i njihovih dužina $n_1, n_2, ...n_k$
Izlaz: formirana k-dimenziona DP tablica

DP $\leftarrow \mathbf{0}_{n_1 \times n_2 \times \cdots \times n_k}$
for $(i_1, i_2, ..., i_k), (el_1, el_2, ..., el_k)$ in enumerate$(a_1 \times a_2 \times \cdots \times a_k)$ **do**
  if $el_1 = el_2 = \cdots = el_k$ **then**
    $DP[i_1, i_2, ..., i_k] \leftarrow DP[i_1 - 1, i_2 - 1, ..., i_k - 1] + 1$
  else
    $DP[i_1, i_2, ..., i_k] \leftarrow \max\left(DP[i_1 - 1, i_2, ..., i_k], DP[i_1, i_2 - 1, ..., i_k], ..., DP[i_1, i_2, ..., i_k - 1]\right)$
  end if
end for
return DP

- **Rekonstruišemo LCS tako što idemo "unazad" kroz DP tablicu**
- **Algoritam:**

Algoritam 2 Formiranje LCS

Ulaz: k-**dimenziona tablica DP, niz sekvenci** $a_1, a_2, ..., a_k$ **i njihovih dužina** $n_1, n_2, ...n_k$
Izlaz: LCS ulaznih sekvenci

```
lcs ← []
while n_i > 0 (∀i ∈ {1, .., k}) do
    step ← DP[n_1, n_2, ..., n_k]
    if (∃i ∈ {1, ..., k}) step = DP[n_1, n_2, ..., n_i − 1, ..., n_k] then
        n_i ← n_i − 1
    else
        lcs.append(a_1[n_1 − 1])
        n_i ← n_i − 1 (∀i ∈ {1, .., k})
    end if
end while
return lcs
```

GENETSKI ALGORITAM

- **Pokušavano sa mnoštvom različitih kombinacija parametara (>100 kombinacija)**
- **Kombinacija koja je imala najbolje rezultate:**
- **Generacijska strategija**
- **Kodiranje binarnim vrednostima**
- **Turnirska selekcija**
- **Uniformno ukrštanje**
- **Višestruka mutacija**
- **20%-ni elitizam**

FITNES FUNKCIJA

- **Preuzeta iz literature:**

$$f(s) = \begin{cases} 3000(|c(s)| + 30k(s) + 50) & \text{ako } |c(s)| = n \wedge k(s) = k \\ 3000(|c(s)| + 30k(s)) & \text{ako } |c(s)| < n \wedge k(s) = k \\ -1000(|c(s)| + 30k(s) + 50)(k - k(s)) & \text{ako } |c(s)| = n \wedge k(s) < k \\ -1000(|c(s)| + 30k(s))(k - k(s)) & \text{ako } |c(s)| < n \wedge k(s) < k \end{cases}$$

- **c(s) je kandidat za rešenje koji je predstavljen jedinkom s**
- **k(s) je broj sekvenci kojima je c(s) podsekvenca**
- **Naša funkcija je blago modifikovana kako bi kažnjavala prazne sekvence**

BEAM SEARCH

- **Ideja je svodjenje problema na pretragu stabla**
- **U pitanju je nepotpuna pretraga stabla u širinu**
- **Skup čvorova koji se nakon svake iteracije zadržava zovemo *beam***
- **Čvorovima dajemo ocene definisanom heuristikom**
- **U *beam*-u naredne iteracije ostaju β najbolje ocenjenih potomaka prethodne iteracije**

GRAF STANJA I OCENJIVANJE

- **U pitanju je usmereni aciklički graf u kom svaki od čvorova predstavlja parcijalno rešenje.**
- **Svaki čvor grafa je predstavljen preko odgovarajućeg levo-pozicionog vektora i dužine, a svaka grana preko karaktera koji se dodaje na prethodno parcijalno rešenje.**
- **Levo-pozicioni vektor je niz indeksa koji obeležavaju odakle je moguće produžiti odgovarajući čvor na svakom od početnih sekvenci.**
- **Na početku stabla problema se stavlja čvor koji predstavlja sam problem, sa levo-pozicionim vektorom popunjenim jedinicama, i dužinom 0.**
- **Listovi ovako definisanog stabla predstavljaju moguće odgovore na problem, jer se ne mogu dalje produžiti. Zbog toga ih takodje nazivamo *kompletnim***

- **Levo-pozicioni vektor čvora v obeležavamo kao $p^{L,v}$**
- **Potproblem predstavljen levo-pozicionim vektorom $p^{L,v}$ obeležavamo kao $S[p^{L,v}]$**
- **Čvor v_1 *dominira* nad čvorom v_2 akko je svaki od elemenata levo-pozicionog vektora čvora v_1 manji od elemenata na istom indeksu levo-pozicionog vektora čvora v_2.**

- **Gornja granica broja karaktera koji mogu da se dodaju na vektor v**

$$\mathrm{UB}_{min}(v) = \mathrm{UB}_{min}(S[p^{L,v}]) = min_{i=1,\ldots,k}(|S_i| - p_i^{L,v} + 1)$$

- **Najprirodnija i najefikasnija ocena gornje granice**
- **Ne uzima u obzir same karaktere koji se pojavljuju u sekvencama**

- **H heuristika, gde je P pretprocesirana verovatnoća da sekvenca leve dužine bude podsekvenca sekvence desne dužine, sa pretpostavkom da su obe nezavisne i nasumično generisane**

$$\mathrm{H}(v) = \mathrm{H}(S[p^{L,v}]) = \prod_{i=1}^{k} \mathcal{P}(t, |S_i| - p_i^{L,v} + 1)$$

$$t := max(1, \lfloor \frac{1}{|\Sigma|} \cdot min_{v \in V_{ext}, i=1,\ldots,k}(|S_i| - p_i^{L,v} + 1) \rfloor)$$

- **Odgovara verovatnoći da parcijalno rešenje predstavljeno čvorom v može da se produži za t karaktera**
- ***Power* heuristika**

$$\mathrm{Pow}(v) = \mathrm{Pow}(S[p^{L,v}]) = \left(\prod_{i=1}^{k} (|S_i| - p_i^{L,v} + 1) \right)^{q} \cdot \mathrm{UB}_{min}(v), q \in [0, 1).$$

- **Za velike vrednosti k, uzima se manja vrednost q**

ALGORITAM

Algoritam 3 Beam Search

Ulaz: skup sekvenci S i odgovarajuća azbuka Σ, funkcija heuristike h za ocenu čvorova, parametar k_{best} za filtriranje, β - veličina *beam*-a

Izlaz: Ostvareno LCS rešenje

$B \leftarrow \{r\}$
$s_{lcs} \leftarrow \epsilon$
while $B \neq \emptyset$ **do**
    $V_{ext} \leftarrow \text{ExtendAndEvaluate}(B, h)$
    ažuriranje s_{lcs} ako je dostignut *kompletan* čvor v sa novim najvećim l_v
    $V_{ext} \leftarrow \text{Filter}(V_{ext}, k_{best})$
    $B \leftarrow \text{Reduce}(V_{ext}, \beta)$
end while
return s_{lcs}

REZULTATI

GENETSKI

Params			DP		GA	
$\|\Sigma\|$	n	k	$t[s]$	s_{best}	$t[s]$	s_{best}
2	10	2	0.01	7	0.01	7
2	20	2	0.01	15	0.08	15
2	10	3	0.01	6	0.01	6
2	20	3	0.02	12	0.1	12
2	10	5	0.7	5	0.03	5
2	20	5	21.9	11	0.13	11
4	10	2	0.01	5	0.03	5
4	20	2	0.01	11	0.15	10
4	10	3	0.01	4	0.04	4
4	20	3	0.03	8	0.29	7
4	10	5	0.72	3	0.02	3
4	20	5	23.6	7	0.2	7
26	10	2	0.01	2	0.02	2
26	20	2	0.01	5	0.09	4
26	10	3	0.01	1	0.01	1
26	20	3	0.03	2	0.05	2
26	10	5	0.73	0	0.01	0
26	20	5	22.8	1	0.02	1

BEAM SEARCH

Params			BS-H		BS-Pow		BS-H-Lit.		BS-Pow-Lit.	
$\|\Sigma\|$	β	k	$t[s]$	s_{best}	$t[s]$	s_{best}	$t[s]$	s_{best}	$t[s]$	s_{best}
4	50	20	0.18	181	0.17	173	0.04	189	0.10	191
4	50	100	0.21	149	0.19	141	0.05	158	0.09	156
4	50	150	0.25	147	0.21	142	0.06	151	0.10	150
4	50	200	0.25	144	0.24	140	0.07	150	0.11	148
4	200	20	0.51	186	0.42	180	0.19	191	0.29	191
4	200	100	0.60	155	0.51	148	0.36	158	0.40	158
4	200	150	0.59	151	0.49	143	0.26	151	0.34	151
4	200	200	0.69	148	0.51	139	0.38	150	0.38	150
4	600	20	1.31	192	1.28	185	0.71	192	1.20	191
4	600	100	1.32	157	1.28	150	0.68	158	1.28	158
4	600	150	1.45	151	1.37	143	1.05	152	1.43	152
4	600	200	1.40	150	1.29	148	1.15	151	1.00	150
20	50	20	0.11	42	0.09	36	0.08	46	0.14	46
20	50	100	0.19	29	0.13	28	0.08	31	0.13	31
20	50	150	0.20	28	0.12	24	0.11	29	0.13	29
20	50	200	0.23	27	0.22	24	0.11	28	0.13	27
20	200	20	0.65	44	0.56	39	0.45	47	0.50	47
20	200	100	0.67	32	0.54	28	0.31	31	0.49	31
20	200	150	0.88	30	0.46	26	0.46	29	0.41	29
20	200	200	0.88	28	0.50	25	0.43	28	0.47	27
20	600	20	1.83	45	1.66	40	1.48	48	1.71	47
20	600	100	1.88	31	1.68	25	1.20	31	1.09	32
20	600	150	1.94	29	1.68	25	1.29	29	1.66	29
20	600	200	2.01	27	1.70	23	1.43	28	1.62	28

HVALA NA PAŽNJI!

Pavle Cvejović 24/2018

Viktor Novaković 92/2018