

Алгоритми за решавање проблема најкраће заједничке надниске

студент: Милош Миковић 1050/2020

ментор: др Александар Картељ



УВОД

- ПНЗН је један од добро познатих NP-тешких проблема оптимизације у области анализе речи
- ПНЗН се може описати као проблем проналажења најкраће речи ω сачињене од слова задате коначне азбуке Σ , тако да су све речи из унапред задатог коначног скупа L садржане у речи ω
- Примене у многим областима информатике, укључујући компресију података, оптимизацију упита, анализу и поређење текста, биоинформатику

Нотација

- Коначна азбука Σ
- Коначна реч $\omega = \omega(1)\omega(2)\dots\omega(n)$, $\omega(j) \in \Sigma$
- Празна реч ε
- Дужину речи ω означавамо са $|\omega| = n$, где $|\varepsilon| = 0$
- Надовезивање слова $\alpha \in \Sigma$ на почетак/крај речи ω означаваћемо са $\alpha\omega/\omega\alpha$
- Приступ слову/секвенци слова означаваћемо са $\omega[k]/\omega[a:b]$

Проблем најкраће заједничке надниске

- Нека важи да $\omega_1, \omega_2 \in \Sigma^*$, за реч ω_1 кажемо да је надниска речи ω_2 у ознаци $\omega_1 \succ \omega_2$ ако важи следећа рекурзивна дефиниција:

$$\omega_1 \succ \varepsilon \triangleq \text{Тачно}$$

$$\varepsilon \succ \omega_2 \triangleq \text{Нетачно, ако } \omega_2 \neq \varepsilon$$

$$\alpha \omega_1 \succ \alpha \omega_2 \triangleq \omega_1 \succ \omega_2$$

$$\alpha \omega_1 \succ \beta \omega_2 \triangleq \omega_1 \succ \beta \omega_2, \text{ ако } \alpha \neq \beta$$

- Симбол \triangleq означава једнакост по дефиницији

Пример

- За дату азбуку $\Sigma = \{a, c, t, g\}$, важи $agcatg \succ act$
- За инстанцу ПНЗН $I = (\{a, c, t, g\}, \{act, cta, aca\})$ најмања заједничка надниска је $acta$
- Најмања заједничка надниска произвољне инстанце проблема не мора бити јединствена

Преглед досадашњих истраживања

- ДП – за k речи дужине максимално n у $O(n^k)$ прос. и врем. сложености
- Оптимизација
- Хеуристичке функције: алфабет, већинско спајање, тежинско већинско спајање
- Алгоритми: редукуј-прошири, AEL (апроксимира очекивану дужину случајно изабране надниске)
- Метатеуристички алгоритми: генеткси алгоритам, оптимизација колонијом мрава, претрага бима



Алгоритам гранања са одсецањем

- Побољшава технику грубе силе
- Ефикасност зависи од критеријума на основу којих се врши одсецање
- Сложеност најгорег случаја остаје експоненцијална али пажљиво одабрани критеријуми одсецања могу одсећи јако велике делове претраге (који су често експоненцијалне величине у односу на димензије улазног проблема)



Алгоритам гранања са одсецањем

```

1:  $d \leftarrow |\omega|$                                 ▷  $d$  - dužina trenutne reči
2: if  $(d > \max D) \vee (d \geq nd)$  then
3:   return
4: end if
5: if  $(d \geq \min D) \wedge \text{ZajedničkaNadniska}(\omega)$  then
6:    $i\text{DaljeNadniska} \leftarrow \text{True}$ 
7:    $\text{pozicija} \leftarrow 0$ 
8:   while  $i\text{DaljeNadniska}$  do                                ▷ optimizacija
9:     if  $\text{ZajedničkaNadniska}(\omega[\text{pozicija} + 1 :])$  then
10:       $\text{pozicija} \leftarrow \text{pozicija} + 1$ 
11:    else
12:       $i\text{DaljeNadniska} = \text{False}$ 
13:    end if
14:  end while
15:   $nn \leftarrow \omega[\text{pozicija} :]$                                 ▷  $nn$  - najbolja nadniska
16:   $nd \leftarrow |nn|$                                             ▷  $nd$  - najbolja dužina
17: end if
18: for  $\alpha \in \Sigma$  do                                ▷ grananje po slovima azbuke
19:    $\text{GrananjeSaOdsecanjem}(\omega\alpha)$ 
20: end for

```

$$\min D = \{\min_{\omega \in \mathcal{L}} |\omega|\}$$

$$\max D = |\Sigma| \cdot L, \text{ gde je } L = \{\max_{\omega \in \mathcal{L}} |\omega|\}$$



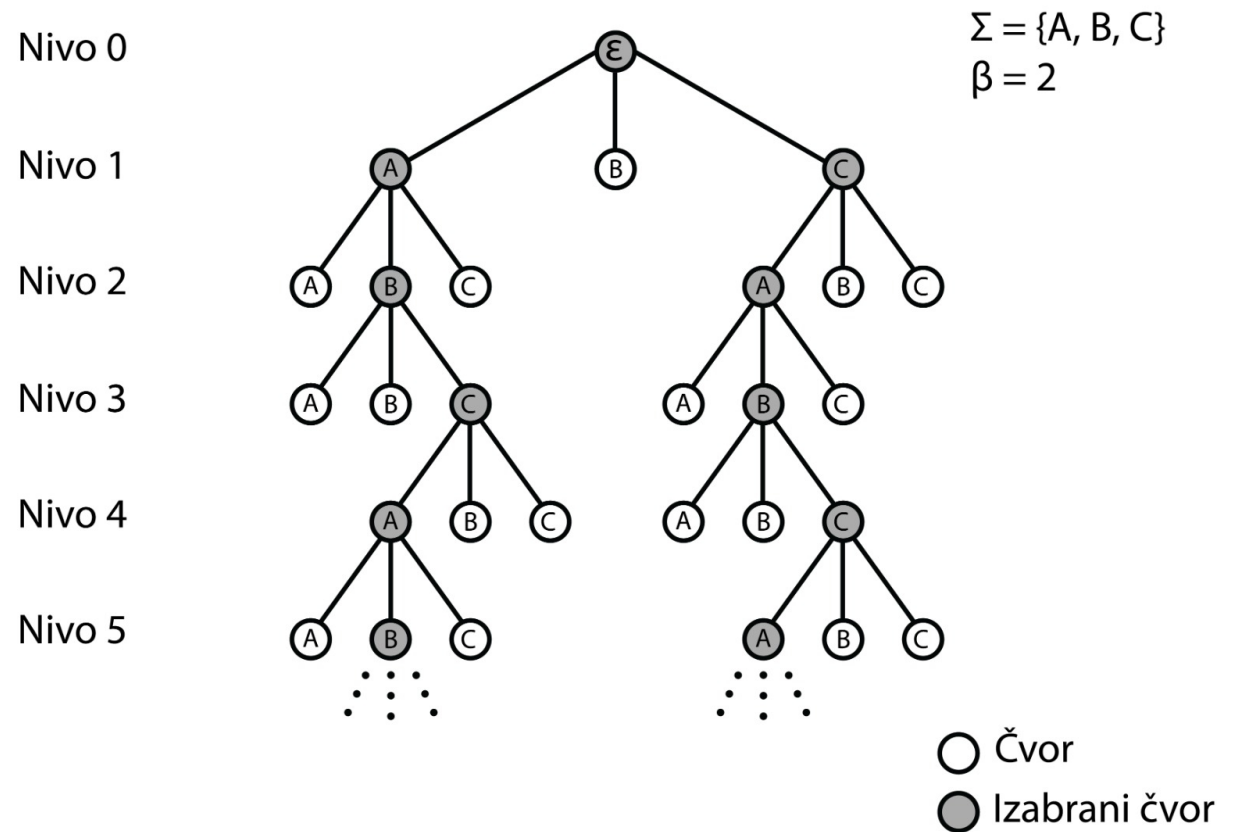
Алгоритам гранања са одсецањем

- У најгорем случају има експоненцијалну сложеност $O(\Sigma^{maxD})$
- Гарантује проналажење оптималног решења
- Практично не употребљив на већим инстанцама проблема



Метахеуристика претрага бима

- Врста претраге графа у ширину
- Сложеност израчунавања држи у задатим границама
- Мета параметар β
- Не гарантује оптимално решење



Алгоритам претраге бима

- Захтева постојање хеуристичке функције ради оцене чворова на одређеном нивоу
- У пракси далеко ефикаснији од алгоритма гранања са одсецањем
- Полиномско време извршавања $O(\beta^* \max D)$



Алгоритам претраге бима

```
1:  $b1 \leftarrow \{\}$ 
2:  $b2 \leftarrow \{\}$ 
3:  $pronađenaNadniska \leftarrow False$ 
4:
5:  $težine \leftarrow H([0,...,0])$ 
6:
7: for  $\alpha \in \Sigma$  do
8:    $težinaSlova \leftarrow 0$ 
9:   for  $t \in težine$  do
10:    if  $\alpha == t.slovo$  then
11:       $težinaSlova \leftarrow t.vrednost$ 
12:    end if
13:  end for
14:   $element \leftarrow \{\alpha, težinaSlova, [0,...,0]\}$ 
15:   $b1.dodaj(element)$ 
16: end for
17:
18:  $b1 \leftarrow RedukujBim(b1, pronađenaNadniska)$ 
19:  $dubina \leftarrow 0$ 
21: while  $\neg pronađenaNadniska \wedge dubina < maxD$  do
22:   for  $s \in b1$  do
23:      $težine \leftarrow H(s.pozicije)$ 
24:     for  $\alpha \in \Sigma$  do
25:        $težinaSlova \leftarrow 0$ 
26:       for  $t \in težine$  do
27:         if  $\alpha == t.slovo$  then
28:            $težinaSlova \leftarrow t.vrednost$ 
29:         end if
30:       end for
31:        $element \leftarrow \{(s.reč)\alpha, s.hvrednost + težinaSlova, s.pozicije\}$ 
32:        $b2.dodaj(element)$ 
33:     end for
34:   end for
35:
36:    $b1.obriši()$ 
37:    $b1 \leftarrow RedukujBim(b2, pronađenaNadniska)$ 
38:    $b2.obriši()$ 
39:    $dubina \leftarrow dubina + 1$ 
40: end while
```

Алгоритам претраге бима

```

1: if  $\mathcal{B}.velicina() \leq \beta$  then
2:   for  $e \in \mathcal{B}$  do
3:     AžurirajPozicije( $e$ )
4:     if ZajedničkaNadniska( $e.reč$ ) then
5:        $nn \leftarrow s.reč$ 
6:        $nd \leftarrow |nn|$ 
7:        $pronađenaNadniska \leftarrow True$ 
8:     end if
9:   end for
10:  return  $\mathcal{B}$ 
11: end if
12:
13: Promešaj( $\mathcal{B}$ )
14: Sortiraj( $\mathcal{B}$ )
15:  $\mathcal{B}_p \leftarrow \{\}$ 
16:
17: for  $i = 0 \rightarrow \beta$  do
18:   AžurirajPozicije( $\mathcal{B}[i]$ )
19:   if ZajedničkaNadniska( $\mathcal{B}[i].reč$ ) then
20:      $nn \leftarrow \mathcal{B}[i].reč$ 
21:      $nd \leftarrow |nn|$ 
22:      $pronađenaNadniska \leftarrow True$ 
23:   end if
24:    $\mathcal{B}_p.dodaj(\mathcal{B}[i])$ 
25: end for
26:
27:  $\mathcal{B}.obriši()$ 
28: return  $\mathcal{B}_p$ 

```

▷ nn - najbolja nadniska
▷ nd - najbolja dužina

```

1:  $poslednjeDodatoSlovo \leftarrow element.reč[element.reč.dužina() - 1]$ 
2:
3: for  $i = 0 \rightarrow \mathcal{L}.velicina()$  do
4:   if  $\mathcal{L}[i][element.pozicije[i]] == poslednjeDodatoSlovo$  then
5:      $element.pozicije[i] \leftarrow element.pozicije[i] + 1$ 
6:   end if
7: end for

```



Хеуристичке функције

1) Већинско спајање

- Инкрементално конструише надниску
- Одреди слово које се најчешће налази на почетку речи из скупа L , а затим се изабрано слово брише са почетка речи из L које га садрже
- Поступак се понавља док скуп L не постане празан
- Не препознаје глобалну структуру речи (слова са почетка краћих речи имају већу шансу да буду уклоњена)

2) Тежинско већинско спајање

- Тежина слова се одређује у односу на дужину преосталих речи у L које почињу тим словом
- Приоритизира скидање слова са почетка дужих речи
- Може да надмаши већинско спајање када нема структурираности унутар скупа L



Тест подаци

- Тест инстанца је облика $I=(\Sigma, L)$
- Оптимално решење најчешће није јединствено а број оптималних решења расте са порастом броја речи у L
- Мањак тест података у литератури
- Проблем оптималног решења на случајно генерисаним инстанцама проблема
- Генератор тест инстанци



Генератор тест инстанци

- Идеја је да се одабере реч *gg* која ће представљати горњу границу оптималности (параметри су дужина *gg* и Σ)
- Параметар γ представља вероватноћу уклањања слова из *gg* при генерисању речи у L
- Генерише m речи
- Фиксиран је *seed* при генерисању тест инстанци
- Сви случајни бројеви узети су из униформне расподеле у одговарајућим опсезима

Генератор тест инстанци

- Нека је нпр. дата азбука $\Sigma=\{a, b\}$
- Нека је $gg=abba$ дужине 4, нека $m=4$ број речи у скупу L и нека је вероватноћа уклањања $\gamma=0.2$
- Нека је добије скуп $L=\{abb, bba, abba, aba\}$
- Повратна вредност генератора је инстанца ПНЗН:

$$I=\{\{a, b\}, \{abb, bba, abba, aba\}\}$$

- IPG $(ka \in \{2, 4\}, m \in \{8, 16, 32\}, \gamma \in \{0.2, 0.4\}, |gg|_{|\Sigma|=2} \in \{20, 24, 28\}, |gg|_{|\Sigma|=4} \in \{12, 13, 14\}, \beta \in \{4, 8, 16\})$
- IP $(ka \in \{2, 4, 16\}, m \in \{10, 20, 40, 80\}, \gamma \in \{0.1, 0.2, 0.4\}, |gg| \in \{50, 100, 500, 2000\}, \beta \in \{100, 200, 400\})$

Експериментални резултати

$IPG_{\gamma=0.4}$							$IPG_{\gamma=0.4}$															
										$\beta=4$				$\beta=8$				$\beta=16$				
$ \Sigma $	m	$ gg $	GSO	$t(s)$	$minD$	$maxD$	$ \Sigma $	m	$ gg $	VS	$t(s)$	TVS	$t(s)$	VS	$t(s)$	TVS	$t(s)$	VS	$t(s)$	TVS	$t(s)$	
2	8	20	18	0	14	28	2	8	20	19	0	18	0	18	0	18	0	18	0	18	0	
2	8	24	21	1	16	32	2	8	24	21	0	21	0	21	0	21	0	21	0	21	0	
2	8	28	25	8	18	36	2	8	28	25	0	25	0	25	0	25	0	25	0	25	0	
2	16	20	19	0	14	28	2	16	20	21	0	20	0	20	0	19	0	19	0	19	0	
2	16	24	21	1	16	32	2	16	24	22	0	22	0	22	0	21	0	22	0	21	0	
2	16	28	26	27	21	42	2	16	28	26	0	26	0	26	0	26	0	26	0	26	0	
2	32	20	20	0	16	32	2	32	20	24	0	22	0	22	0	21	0	21	0	21	0	
2	32	24	23	4	20	40	2	32	24	27	0	26	0	26	0	26	0	26	0	25	0	
2	32	28	27	53	23	46	2	32	28	32	0	30	0	31	0	27	0	30	0	27	0	
4	8	12	12	3	9	36	4	8	12	12	0	12	0	12	0	12	0	12	0	12	0	
4	8	13	13	17	11	44	4	8	13	13	0	13	0	13	0	13	0	13	0	13	0	
4	8	14	14	60	10	40	4	8	14	14	0	14	0	14	0	14	0	14	0	14	0	
4	16	12	12	4	9	36	4	16	12	12	0	12	0	12	0	12	0	12	0	12	0	
4	16	13	13	16	11	44	4	16	13	13	0	13	0	13	0	13	0	13	0	13	0	
4	16	14	14	60	10	40	4	16	14	14	0	14	0	14	0	14	0	14	0	14	0	
4	32	12	12	4	9	36	4	32	12	12	0	12	0	12	0	12	0	12	0	12	0	
4	32	13	13	17	11	44	4	32	13	13	0	13	0	13	0	13	0	13	0	13	0	
4	32	14	14	65	11	44	4	32	14	14	0	14	0	14	0	14	0	14	0	14	0	

Експериментални резултати

$IP_{\gamma=0.1}$						
	$>gg$			$\leq gg$		
<i>BS</i>	100	200	400	100	200	400
<i>VS</i>	0	0	0	48	0	0
<i>TVS</i>	0	0	0	48	0	0

$IP_{\gamma=0.2}$						
	$>gg$			$\leq gg$		
<i>BS</i>	100	200	400	100	200	400
<i>VS</i>	0	2	1	43	1	1
<i>TVS</i>	1	0	2	44	0	1

$IP_{\gamma=0.4}$						
	$>gg$			$\leq gg$		
<i>BS</i>	100	200	400	100	200	400
<i>VS</i>	1	4	5	24	8	6
<i>TVS</i>	1	4	4	22	10	7

$VS_{protiv TVS}$		
<i>VS</i>	<i>TVS</i>	<i>ISTI</i>
0	0	48

$VS_{protiv TVS}$		
<i>VS</i>	<i>TVS</i>	<i>ISTI</i>
0	3	45

$VS_{protiv TVS}$		
<i>VS</i>	<i>TVS</i>	<i>ISTI</i>
2	21	25



Закључак и правци даљег рада

- Look-ahead верзије представљених хеур. функција:
 1. Инкрементална изградња и рангирање слова (лошији резултати)
 2. Инкрементална изградња без рангирања (лошији резултати)
 3. Без инкременталне изградње (дуже време извршавања)
- Селекција при редукцији бима
- Технике редукције надниске
- Хеуристичке функције које процењују очекивану дужину надниске (AEL)
- Генератор тест инстанци

Хвала на пажњи

- Питања?

