ОАС Софтверско инжењерство и информационе технологије

Организација података

Динамичка расута организација датотеке

Садржај

- **Увод**
- Прошириве расуте датотеке
- Ресурси

Увод

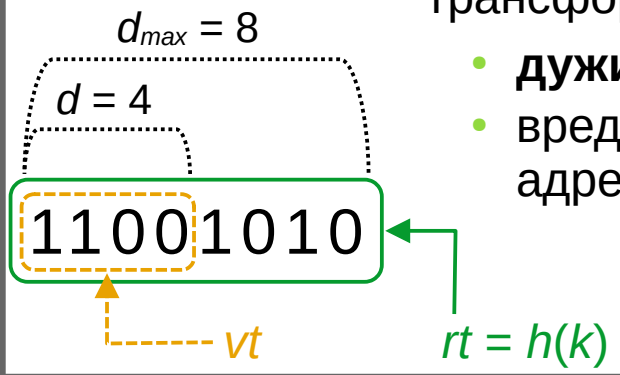
- Динамичка расута датотека (Mogin, 2008)
  - динамичко прилагођавање величине меморијског простора датотеке према броју слогова
    - метода трансформација вредности идентификатора у адресу независна од величине меморијског простора датотеке
  - применом трансформације вредности идентификатора слога у адресу добија се адреса бакета, у којем би тај слог требало да се налази
  - логичке везе између слогова нису меморисане, а слогови су у логичком погледу расути по меморијском простору датотеке

Увод

- Динамичке расуте датотеке (Mogin, 2008)
  - врсте динамичких расутих датотека – примери
    - динамичке расуте датотеке проширивог типа
      - прошириве расуте датотеке
    - динамичке расуте датотеке динамичког типа
    - динамичке расуте датотеке линеарног типа

Увод

- Динамичке расуте датотеке (Mogin, 2008)
  - заједничке карактеристике динамичких расутих датотека проширивог, динамичког и линеарног типа
    - трансформација вредности кључа у адресу
      - метода трансформације не зависи од величине адресног простора датотеке и не мења се услед уписа или брисања слогова
      - **резултат трансформације *rt*** је резултат примене трансформације *h* на вредност кључа *k*
        - у облику бинарног броја **максималне дужине d_{max}**
      - **вредност трансформације *vt*** је водећих *d* битова резултата трансформације *rt*
        - **дужина *d*** (дужина вредности трансформације) у опсегу [0, d_{max}]
        - вредност трансформације *vt* се употребљава при утврђивању адресе бакета који одговара посматраном слогу



Увод

- Динамичке расуте датотеке (Mogin, 2008)
  - заједничке карактеристике динамичких расутих датотека проширивог, динамичког и линеарног типа
    - динамичка природа
      - дужина *d* и број бакета *B* се динамички мењају заједно с променом величине датотеке током њеног постојања

Садржај

- Увод
- **Прошириве расуте датотеке**
- Ресурси

Прошириве расуте датотеке

- Прошириве расуте датотеке (Mogin, 2008)
  - једна врста динамичких расутих датотека
  - сложена структура
    - два дела
      - адресар
        - подаци о начину распоређивања слогова по бакетима
      - зона података
        - слогови организовани по бакетима
    - два дела бивају имплементирана као две засебне датотеке

Прошириве расуте датотеке

- Адресар (Mogin, 2008)
  - помоћна структура података
    - служи као индекс за приступање бакетима, који се налазе у зони података
      - обично релативно малих димензија
        - може бити у целости у оперативној меморији током рада над датотеком
    - најчешће линеарне врсте

Прошириве расуте датотеке

- Адресар (Mogin, 2008)
  - структура адресара
    - низ показивача који садржи адресе бакета
      - дужина низа 2^d ($d \geq 0$, $2^d \geq B$)

        $d = \lceil \log_2 B \rceil$
        - d – број водећих битова резултата трансформације (дужина вредности трансформације)
        - B – број актуелних бакета у зони података
      - могуће вредности трансформације за актуелно d служе као индекси низа
      - адресе бакета су елементи низа
    - помоћна поља
      - поље за дужину d
      - поље за фактор бакетирања b

пример адресара

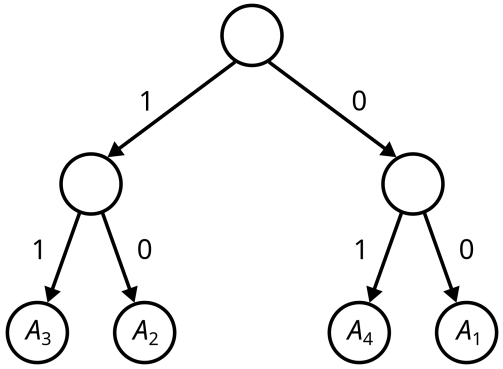
00	A_1
01	A_4
10	A_2
11	A_3

d	2
b	5

Прошириве расуте датотеке

- Адресар (Mogin, 2008)
  - структура адресара
    - низ показивача од 2^d елемената може бити дат као стабло
      - комплетно бинарно стабло висине $h = d + 1$
        - листови садрже адресе бакета
        - ивицама придружени бројеви из скупа {0, 1}
        - путу од корена до листа одговара низ бројева придружених ивицама на том путу
          - низ бројева одговара индексу низа показивача

пример стабла за низ показивача



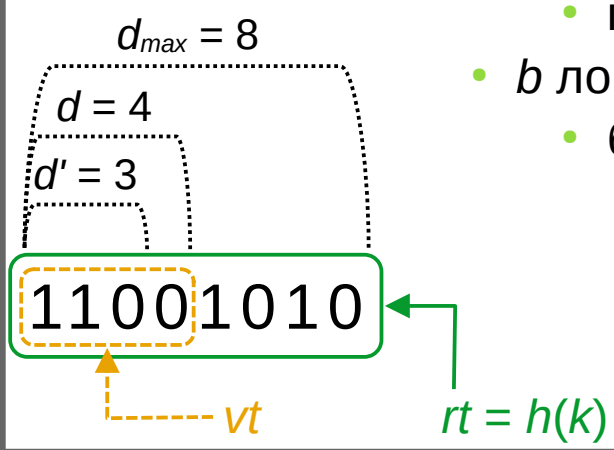
пример адресара

00	A_1
01	A_4
10	A_2
11	A_3

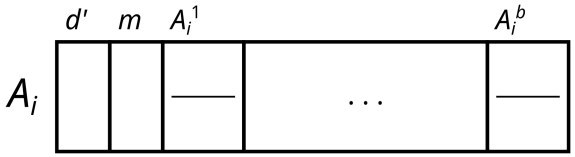
d	2
b	5

Прошириве расуте датотеке

- Зона података (Mogin, 2008)
  - структура зоне података
    - *B* бакета
      - структура бакета
        - заглавље
          - поље за **локалну дужину *d'*** (локална дужина вредности трансформације) у опсегу [0, *d*]
            - *d'* је број водећих битова вредности трансформације *vt* који као низ морају бити исти за све слогове у бакету
          - поље за број актуелних слогова у бакету *m*
        - *b* локација за слогове
          - барем један слог је присутан



пример структуре бакета



Прошириве расуте датотеке

- Адресирање (Mogin, 2008)
  - добављање адресе одговарајућег бакета из низа показивача у адресару на основу дате вредности кључа
    - ако $d = 0$
      - низ показивача има само један елемент и очитавањем тог елемента добија се адреса бакета
        - вредност кључа и његово трансформисање нису битни
    - ако $d > 0$
      - вредност трансформације *vt* дужине *d* користи се као индекс према којем треба приступити одговарајућем елементу низа показивача ради добављања адресе бакета
        - вредност трансформације *vt* дужине *d* бива генерисана на основу вредности кључа

Прошириве расуте датотеке

- Адресирање (Mogin, 2008)
  - локална дужина *d'* (локална дужина вредности трансформације) може варирати по бакетима
    - случај *d'* = *d* за појединачни бакет *t*
      - вредност трансформације *vt* дужине *d* је иста за сваки слог бакета *t*
      - адреса бакета *t* се јавља тачно једном у низу показивача у адресару
      - ако *d* > 0 и *d'* > 0
        - вредности трансформације *vt* за слогове бакета *t* су исте као вредност трансформације која представља индекс оног елемента у низа показивача у адресару у којем је адреса бакета *t*
      - ако *d* = 0 и *d'* = 0
        - трансформисање вредности кључа за слогове бакета *t* није битно
        - низ показивача у адресару има само један елемент

Прошириве расуте датотеке

- Адресирање (Mogin, 2008)
  - локална дужина *d'* (локална дужина вредности трансформације) може варирати по бакетима
    - случај *d'* < *d* за појединачни бакет *t*
      - за све слогове бакета *t* низ од *d'* водећих битова вредности трансформације *vt* је исти, док низ од преосталих *d* - *d'* битова може варирати
      - у низу показивача у адресару, $2^{d - d'}$ суседних елемената чији индекси се слажу по *d'* водећих битова вредности трансформације *vt* садржи адресу бакета *t*

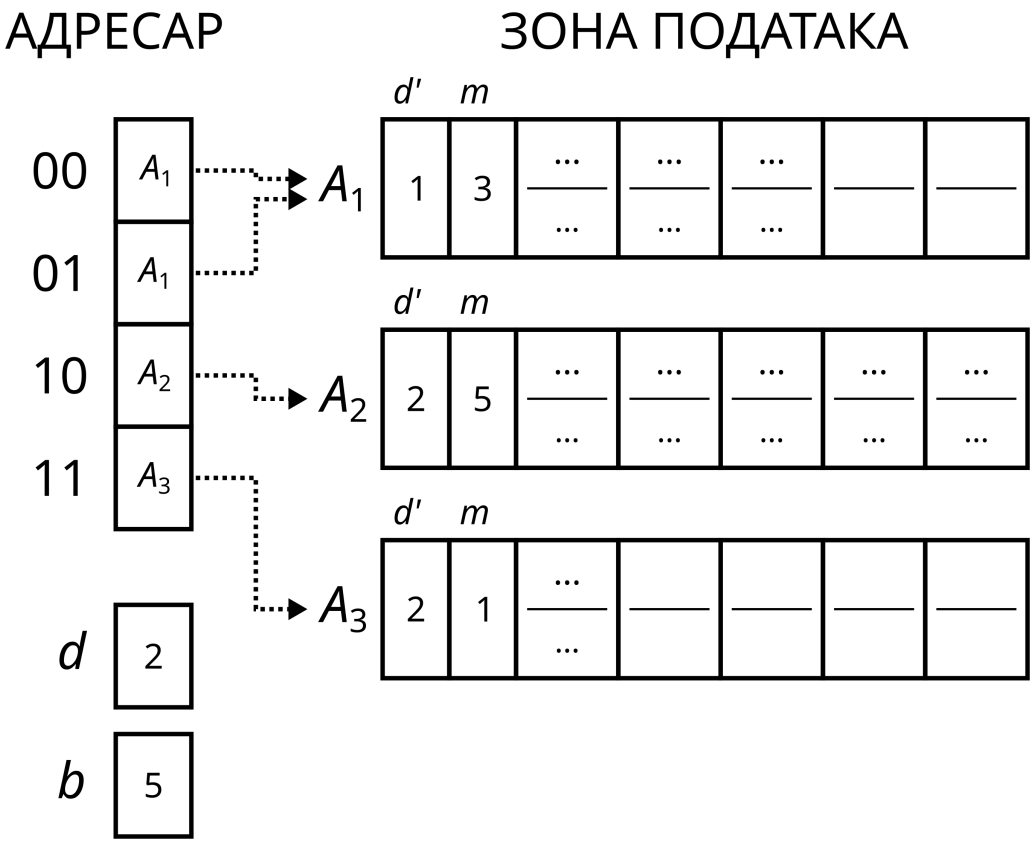
Прошириве расуте датотеке

- Адресирање
  - пример

- адресар
  - вредност трансформације *vt* има дужину $d = 2$
    - $vt \in \{00, 01, 10, 11\}$
  - низ показивача од $2^d = 4$ елемента
    - *vt* као индекс у низу показивача

- зона података
  - фактор бакетирања $b = 5$
  - постоје $B = 3$ бакета
    - бакети A_1, A_2, A_3
  - постоји $N = 9$ слогова

- основна структура слога
  - вредност кључа (цео број)
  - вредности некључних обележја



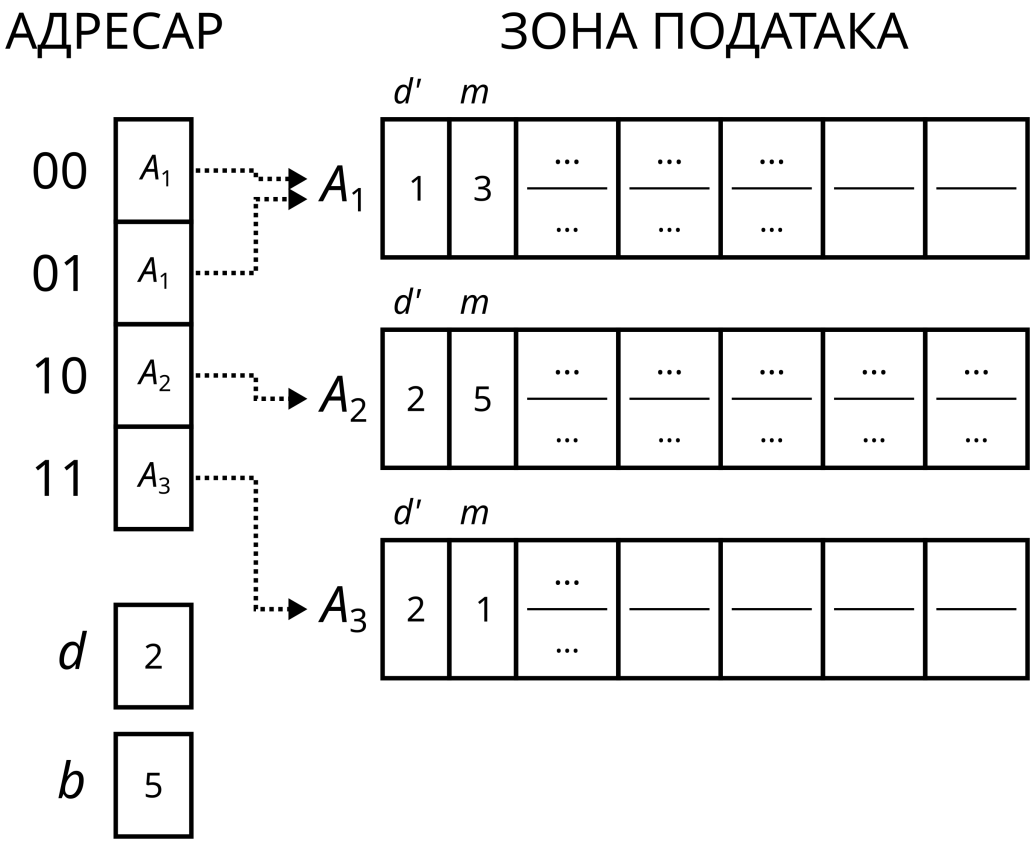
Прошириве расуте датотеке

- Адресирање
  - пример

- бакет A_1 садржи m = 3 слога
  - садржани слогови се слажу по d' = 1 водећем биту од *vt* (низ **0**)
  - адреса на $2^{d-d'}$ = 2 места у адресару
    - индекси **0**0 и **0**1

- бакет A_2 садржи m = 5 слогова
  - садржани слогови се слажу по d' = 2 водећа бита од *vt* (низ **10**)
  - адреса на $2^{d-d'}$ = 1 месту у адресару
    - индекс **10**

- бакет A_3 садржи m = 1 слог
  - садржани слогови се слажу по d' = 2 водећа бита од *vt* (низ **11**)
  - адреса на $2^{d-d'}$ = 1 месту у адресару
    - индекс **11**



Прошириве расуте датотеке

- Генерисање вредности трансформације (Mogin, 2008)
  - кораци у добијању вредности трансформације *vt*
    - 1) израчунавање резултата трансформације *rt* применом нетривијалне трансформације *h* на вредност кључа *k*
    - 2) за резултат трансформације *rt* дужине d_{max} редослед бинарних позиција бива инвертован
    - 3) вредност трансформације *vt* добија се издвајањем *d* водећих битова из резултата трансформације с инвертованим позицијама

Прошириве расуте датотеке

- Генерисање вредности трансформације (Mogin, 2008)
  - пројектантске мере
    - избор трансформације *h*
      - тривијална трансформација није погодна јер може бити релативно слабија равномерност расподеле вредности трансформације
      - примена неке нетривијалне трансформације
        - примена неке методе за генерисање псеудослучајних бројева, при чему треба постићи да низ генерисан за вредности кључа има што равномернију расподелу вредности
      - критеријум – циљ да за случај највећег могућег броја бакета буде могуће смештање v^p слогова са свим могућим вредностима кључа

        $b\,T \geq v^p$
        - T – број могућих резултата трансформације
      - максимална дужина вредности трансформације d_{max}

        $d_{max} = \lceil \log_2 T \rceil$

Прошириве расуте датотеке

- Генерисање вредности трансформације (Mogin, 2008)
  - пројектантске мере
    - инвертовање позиција резултата трансформације
      - треба постићи да низ генерисан за вредности кључа има што равномернију расподелу вредности
      - резултат трансформације *rt* је дужине d_{max} али се може десити да неки од бинарних бројева дужине d_{max} не одговара ниједном од могућих резултата трансформација јер важи неједнакост $2^{d_{max}} \geq T$
        - горњи опсег бинарних бројева дужине d_{max} можда недостижан $[T, 2^{d_{max}} - 1]$
        - водећи бит од *rt* може имати већу вероватноћу да буде 0 него 1, а за нулти бит очекивана практично равномерна расподела
      - редослед бинарних позиција резултата трансформације *rt* бива инвертован, па битови за вредност трансформације *vt* бивају одабирани с оне стране *rt* с очекивано равномернијом расподелом

Прошириве расуте датотеке

- Генерисање вредности трансформације (Mogin, 2008)
  - пројектантске мере
    - инвертовање позиција резултата трансформације
      - пример анализе погодности примене инвертовања позиција
        - број могућих резултата трансформације $T = 10$
          - генерисани бројеви имају равномерну расподелу
        - резултат трансформације $rt \in [0, 9]$
          - резултат трансформације је бинарни број максималне дужине d_{max}

$$d_{max} = \lceil \log_2 T \rceil = \lceil \log_2 10 \rceil = 4$$

            - недостижни горњи опсег

$$[T, 2^{d_{max}} - 1] = [10, 15]$$

rt	
0: 0000	8: 1000
1: 0001	9: 1001
2: 0010	~~10: 1010~~
3: 0011	~~11: 1011~~
4: 0100	~~12: 1100~~
5: 0101	~~13: 1101~~
6: 0110	~~14: 1110~~
7: 0111	~~15: 1111~~

водећи бит
- вероватноћа да 0: 80%
- вероватноћа да 1: 20%

нулти бит
- вероватноћа да 0: 50%
- вероватноћа да 1: 50%

Прошириве расуте датотеке

- Генерисање вредности трансформације
  - пример
    - вредности кључа из опсега од 0 до 199 (v = 10, p = 3)
    - фактор бакетирања b = 7
    - трансформација h с бројем могућих резултата T = 149

      $$h(k(S))=1+k(S)(\text{mod } 149)$$

      - максимална дужина вредности трансформације d_{max}

        $$d_{max}=\lceil \log_2 T \rceil=\lceil \log_2 149 \rceil=8$$
    - рачунање вредности трансформације

$k(S)$	1	23	43	72	154	199
$h(k(S))$	2_{10} 00000010_2	24_{10} 00011000_2	44_{10} 00101100_2	73_{10} 01001001_2	6_{10} 00000110_2	51_{10} 00110011_2
vt за d = 8	01000000	00011000	00110100	10010010	01100000	11001100
vt за d = 2	01	00	00	10	01	11

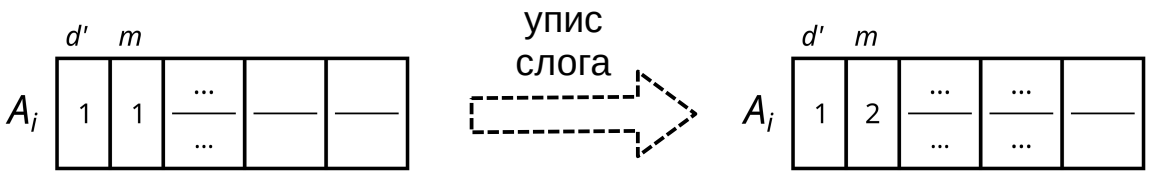
Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - формирање на основу садржаја улазне серијске датотеке
    - слогови улазне датотеке редом бивају читани и за сваки слог бива покушано смештање у расуту датотеку
      - за учитани слог улазне датотеке адреса одговарајућег бакета бива добављена из низа показивача у адресару расуте датотеке
      - одговарајући бакет бива учитан из расуте датотеке и, ако у том бакету не постоји слог који има исту вредност кључа као учитани слог, прелази се на упис учитаног слога у тај бакет
        - упису претходи неуспешно тражење
    - три карактеристична случаја уписа слога у бакет
      - **1) прост упис**
      - **2) упис с дељењем бакета и проширењем низа показивача**
      - **3) упис с дељењем бакета и без проширења низа показивача**

Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 1) прост упис
    (када $m < b$)
    - слог бива уписан у прву слободну локацију у бакету
    - m бива увећано за 1
    - специјални случај
      - када $d = 0$, што важи на почетку формирања
        - низ показивача има само један елемент, који представља адресу јединог бакета у зони података
        - за једини бакет важи $d' = 0$
        - трансформисање вредности кључа слога није битно
          - слог бива смештен у једини бакет уз увећање m за 1

пример простог уписа

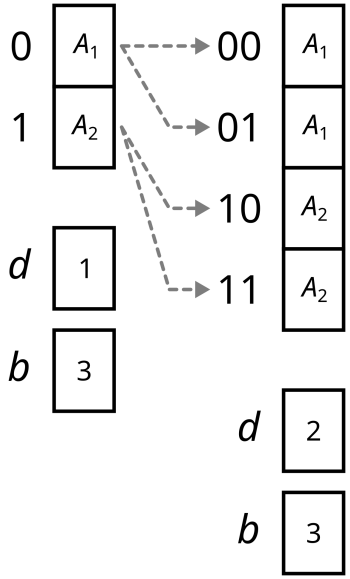


Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 2) упис с дељењем бакета и проширењем низа показивача (када $m = b$ и $d' = d$)
    - почетно стање
      - посматрани бакет је пун
      - адреса посматраног бакета је на једном месту у низу показивача
      - вредност трансформације *vt* дужине *d* је иста и за све слогове посматраног бакета и за нови слог (укупно $m + 1$ таквих слогова)
    - основна намера
      - потребан је нови бакет
        - ради обезбеђења места за смештање $m + 1$ слогова, при чему би део слогова био у посматраном а део у новом бакету
      - потребна је примена вредности трансформације веће дужине
        - ради разврставања $m + 1$ слогова у две непразне групе
        - неопходно је проширење низа показивача

Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 2) упис с дељењем бакета и проширењем низа показивача (када $m = b$ и $d' = d$)
    - основни кораци – проширење низа показивача
      - d бива увећано за 1 и дужина низа показивача бива удвостручена
        - нови низ показивача имаће двоструко више елемената
        - сваки елемент старог низа показивача биће на два индекса присутан у новом низу показивача
          - један нови индекс је стари индекс елемента продужен с 0
          - други нови индекс је стари индекс елемента продужен с 1



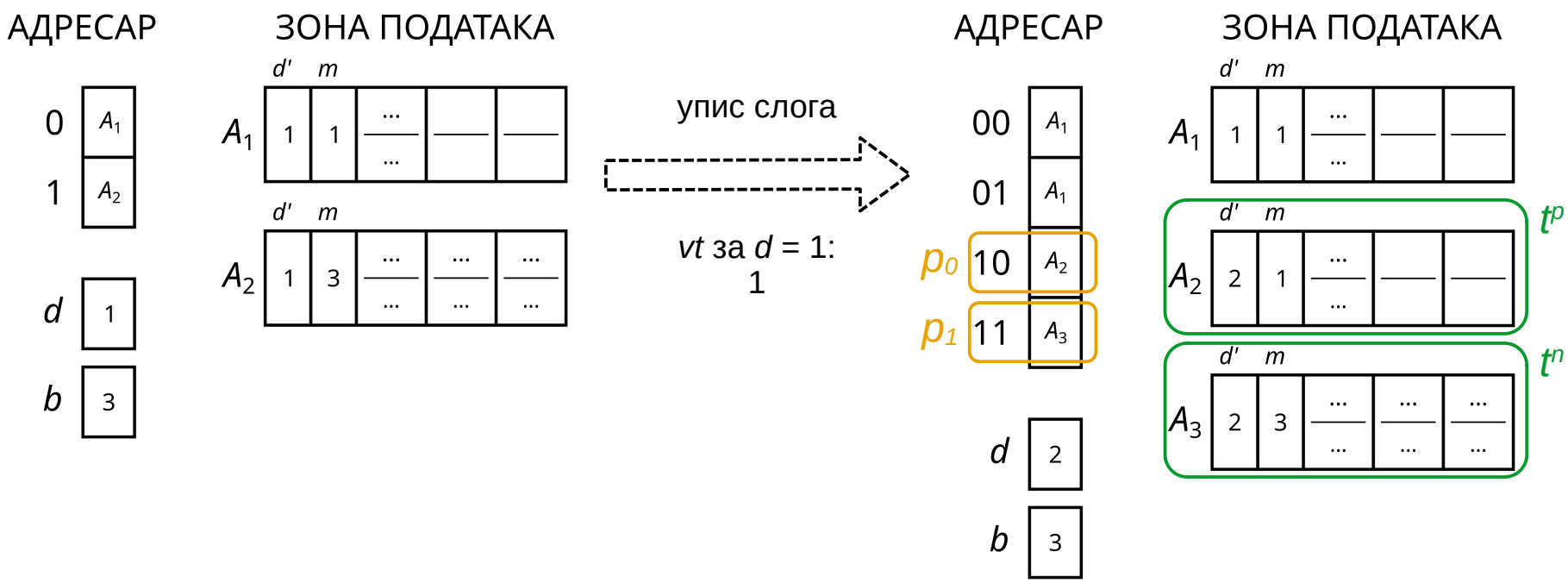
пример удвостручења дужине низа показивача

Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 2) упис с дељењем бакета и проширењем низа показивача (када $m = b$ и $d' = d$)
    - основни кораци – дељење бакета
      - t^p је посматрани бакет (пуни бакет), на који из новог низа показивача показују показивачи p_0 и p_1, а t^n је нови бакет
      - бакет t^p бива подељен
        - у бакет t^p бивају смештени они од $m + 1$ слогова за које је нулти бит 0 у вредности трансформације увећане дужине
        - у бакет t^n бивају смештени они од $m + 1$ слогова за које је нулти бит 1 у вредности трансформације увећане дужине
      - показивач p_1 бива промењен да показује на бакет t^n
      - у бакетима t^p и t^n, вредност d' бива постављена на нову вредност d а вредност m одговара броју актуелних слогова у бакету
      - за сваки од бакета осим за бакете t^p и t^n важи да на њега показују бар два показивача и да је задовољена неједнакост $d' < d$

Прошириве расуте датотеке

- Формирање
  - 2) упис с дељењем бакета и проширењем низа показивача (када $m = b$ и $d' = d$)
    - пример



Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 2) упис с дељењем бакета и проширењем низа показивача (када $m = b$ и $d' = d$)
    - додатни кораци – вишекратно дељење бакета
      - након увећања d и проширења низа показивача, може се десити да $m + 1$ слогова и даље одговара само једном бакету и да нема места за нови слог, те треба покушати с настављањем поступка уписа
        - додатни покушаји уписа
          - покушај уписа по узору на основне кораке
      - потребно је постепено увећавати d и проширивати низ показивача, а уз то треба и стално покушавати распоређивање $m + 1$ слогова у два бакета
      - распоређивање $m + 1$ слогова у два бакета може бити успешно када постоји бар један слог којем одговара другачији бакет у односу на остале слогове

Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 2) упис с дељењем бакета и проширењем низа показивача (када $m = b$ и $d' = d$)
    - додатни кораци – вишекратно дељење бакета
      - након успешног распоређивања слогова, додатни покушаји уписа нису потребни
      - при сваком неуспешном распоређивању слогова, један од бакета остаје празан
        - празни бакети не бивају меморисани, а показивачи који би требало да показују на њих имају специјалну вредност (*)
      - могућност извођења уписа
        - ако је датотека добро испројектована, упис може бити изведен
        - ако би у одговарајућем скупу синонима било више од b слогова и у случају примене вредности трансформације максималне дужине $d = d_{max}$, нови слог не би могао бити уписан
          - не постоји подршка за прекорачиоце

Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 3) упис с дељењем бакета и без проширења низа показивача (када $m = b$ и $d' < d$)
    - почетно стање
      - посматрани бакет је пун
      - адреса посматраног бакета је на $2^{d - d'}$ места у низу показивача
      - низ водећих d' битова вредности трансформације је исти и за све слогове посматраног бакета и за нови слог (укупно $m + 1$ таквих слогова)
    - основна намера
      - потребан је нови бакет
        - ради обезбеђења места за смештање $m + 1$ слогова, при чему би део слогова био у посматраном а део у новом бакету
      - потребно је увећање локалне дужине вредности трансформације d'
        - ради разврставања $m + 1$ слогова у две непразне групе

Прошириве расуте датотеке

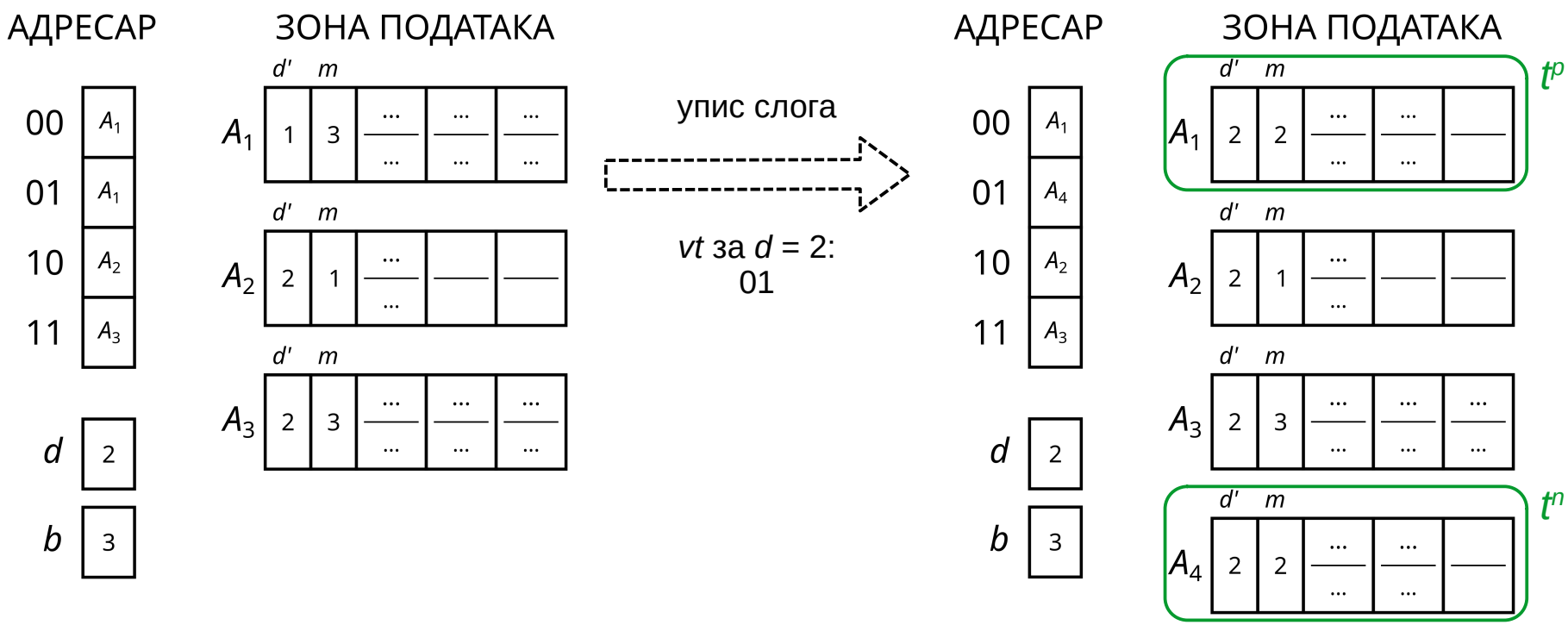
- Формирање (Mogin, 2008)
  - 3) упис с дељењем бакета и без проширења низа показивача (када $m = b$ и $d' < d$)
    - основни кораци – примена већег броја водећих битова вредности трансформације
      - потребно је посматрати водећих $d' + 1$ битова вредности трансформације
        - како важи $d' < d$, нема увећања d и нема удвостручења дужине низа показивача

Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 3) упис с дељењем бакета и без проширења низа показивача (када $m = b$ и $d' < d$)
    - основни кораци – дељење бакета
      - t^p је посматрани бакет (пуни бакет), а t^n је нови бакет
      - бакет t^p бива подељен
        - у бакет t^p бивају смештени они од $m + 1$ слогова за које је нулти бит 0 у низу водећих $d' + 1$ битова вредности трансформације
        - у бакет t^n бивају смештени они од $m + 1$ слогова за које је нулти бит 1 у низу водећих $d' + 1$ битова вредности трансформације
      - на сваки од бакета t^p и t^n показује половина од првобитних $2^{d - d'}$ показивача
        - индекси показивача на бакет t^p имају бит 0 на позицији $d - d' - 1$
        - индекси показивача на бакет t^n имају бит 1 на позицији $d - d' - 1$
      - у бакетима t^p и t^n, вредност d' бива увећана за 1 а вредност m одговара броју актуелних слогова у бакету

Прошириве расуте датотеке

- Формирање
  - 3) упис с дељењем бакета и без проширења низа показивача (када $m = b$ и $d' < d$)
    - пример



Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 3) упис с дељењем бакета и без проширења низа показивача (када $m = b$ и $d' < d$)
    - додатни кораци – вишекратно дељење бакета
      - након примене већег броја водећих битова вредности трансформације, може се десити да $m + 1$ слогова и даље одговара само једном бакету и да нема места за нови слог, те треба покушати с настављањем поступка уписа
        - додатни покушаји уписа
          - ако на почетку покушаја важи $d' < d$
            - покушај уписа по узору на основне кораке уписа с дељењем бакета и без проширења низа показивача
          - ако на почетку покушаја важи $d' = d$
            - покушај уписа по узору на основне кораке уписа с дељењем бакета и проширењем низа показивача

Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 3) упис с дељењем бакета и без проширења низа показивача (када $m = b$ и $d' < d$)
    - додатни кораци – вишекратно дељење бакета
      - потребно је постепено увећавати d', а уз то треба и стално покушавати распоређивање $m + 1$ слогова у два бакета
        - у ситуацији када пре увећања d' важи $d' = d$, онда увећање d' подразумева и увећање d и проширење низа показивача
      - распоређивање $m + 1$ слогова у два бакета може бити успешно када постоји бар један слог којем одговара другачији бакет у односу на остале слогове

Прошириве расуте датотеке

- Формирање (Mogin, 2008)
  - 3) упис с дељењем бакета и без проширења низа показивача (када $m = b$ и $d' < d$)
    - додатни кораци – вишекратно дељење бакета
      - након успешног распоређивања слогова, додатни покушаји уписа нису потребни
      - при сваком неуспешном распоређивању слогова, један од бакета остаје празан
        - празни бакети не бивају меморисани, а показивачи који би требало да показују на њих имају специјалну вредност (*)
      - могућност извођења уписа
        - ако је датотека добро испројектована, упис може бити изведен
        - ако би у одговарајућем скупу синонима било више од b слогова и у случају примене вредности трансформације максималне дужине $d = d_{max}$, нови слог не би могао бити уписан
          - не постоји подршка за прекорачиоце

Прошириве расуте датотеке

- Формирање
  - пример – улазна серијска датотека

- датотека је блокирана
  - фактор блокирања $f = 5$

- постоје укупно $B^{ser} = 2$ блока
  - блокови A_1, A_2

- постоји $N^{ser} = 9$ слогова
  - 9 обичних слогова
    - слогови $S_1, \ldots, S_9$
  - + 1 специјални слог – за крај (*)

- структура слога
  - вредност кључа (цео број)
  - вредности некључних обележја
    - $n(S_i)$, i је ознака слога

СЕРИЈСКА ДАТОТЕКА

A_1	18	74	45	7	54
	$n(S_1)$	$n(S_2)$	$n(S_3)$	$n(S_4)$	$n(S_5)$
A_2	98	28	87	82	*
	$n(S_6)$	$n(S_7)$	$n(S_8)$	$n(S_9)$	

Прошириве расуте датотеке

- Формирање
  - пример – пројектовање
    - вредности кључа из опсега од 0 до 99 (v = 10, p = 2)
    - фактор бакетирања b = 3
    - трансформација h с бројем могућих резултата T = 59

      $$h(k(S))=1+k(S)(\text{mod } 59)$$

      - максимална дужина вредности трансформације d_{max}

        $$d_{max}=\lceil \log_2 T \rceil=\lceil \log_2 59 \rceil=6$$

Прошириве расуте датотеке

- Формирање
  - пример – генерисање вредности трансформације

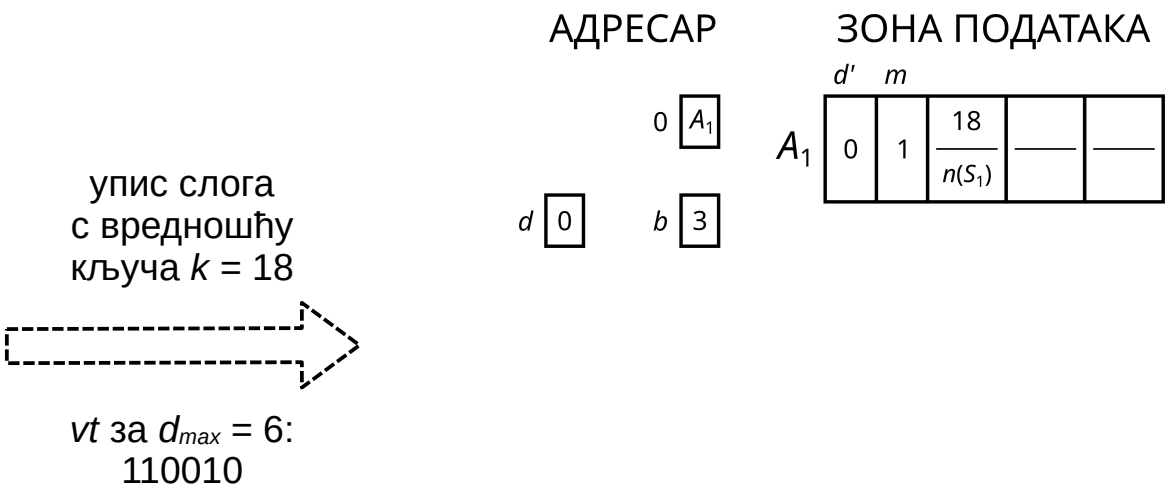
$k(S)$	18	74	45	7	54	98	28	87	82
$h(k(S))$	19_{10}	16_{10}	46_{10}	8_{10}	55_{10}	40_{10}	29_{10}	29_{10}	24_{10}
	010011_2	010000_2	101110_2	001000_2	110111_2	101000_2	011101_2	011101_2	011000_2
vt за $d = 6$	110010	000010	011101	000100	111011	000101	101110	101110	000110
vt за $d = 4$	1100	0000	0111	0001	1110	0001	1011	1011	0001

$$h(k(S)) = 1 + k(S) \pmod{59}$$

$$d_{max} = \lceil \log_2 T \rceil = \lceil \log_2 59 \rceil = 6$$

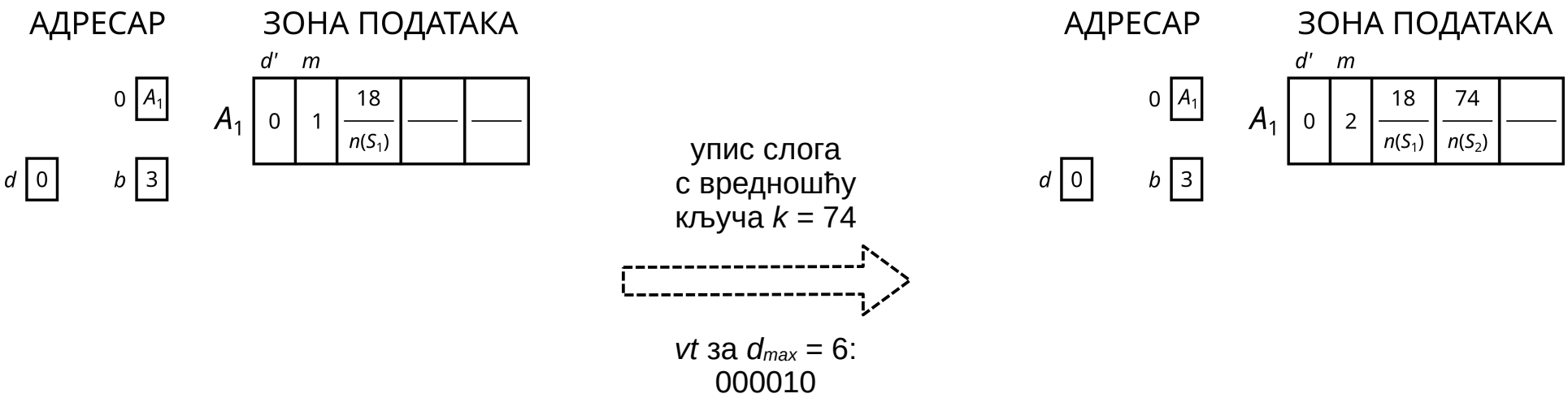
Прошириве расуте датотеке

- Формирање
  - пример – корак 1



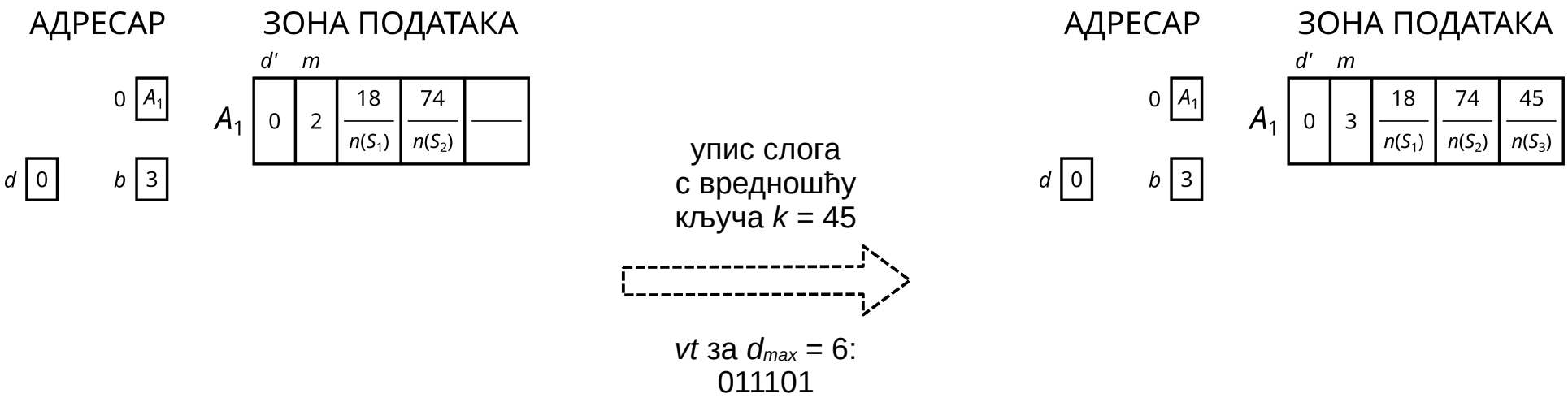
Прошириве расуте датотеке

- Формирање
  - пример – корак 2



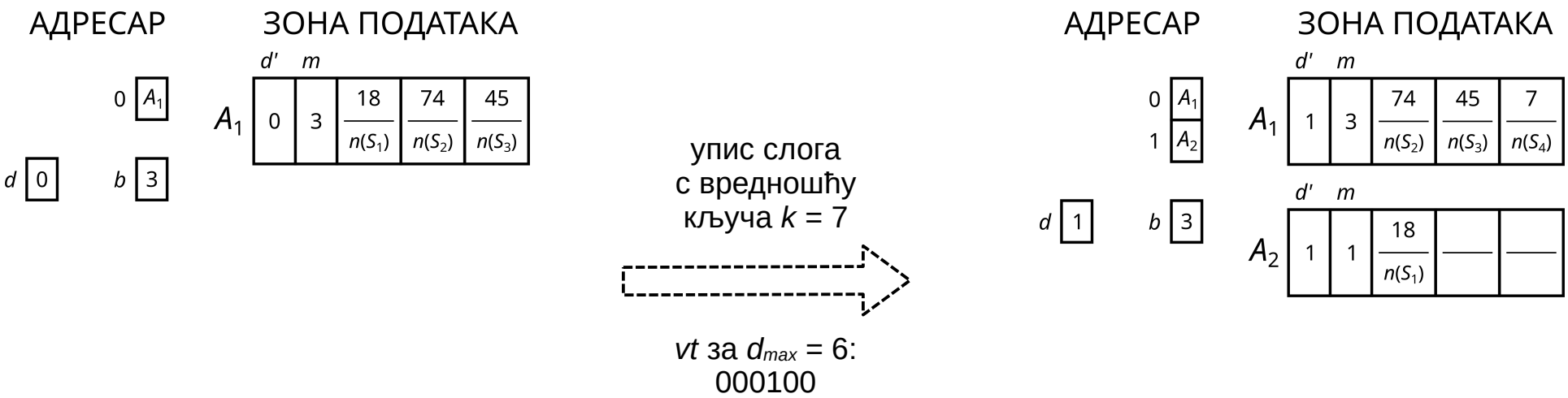
Прошириве расуте датотеке

- Формирање
  - пример – корак 3



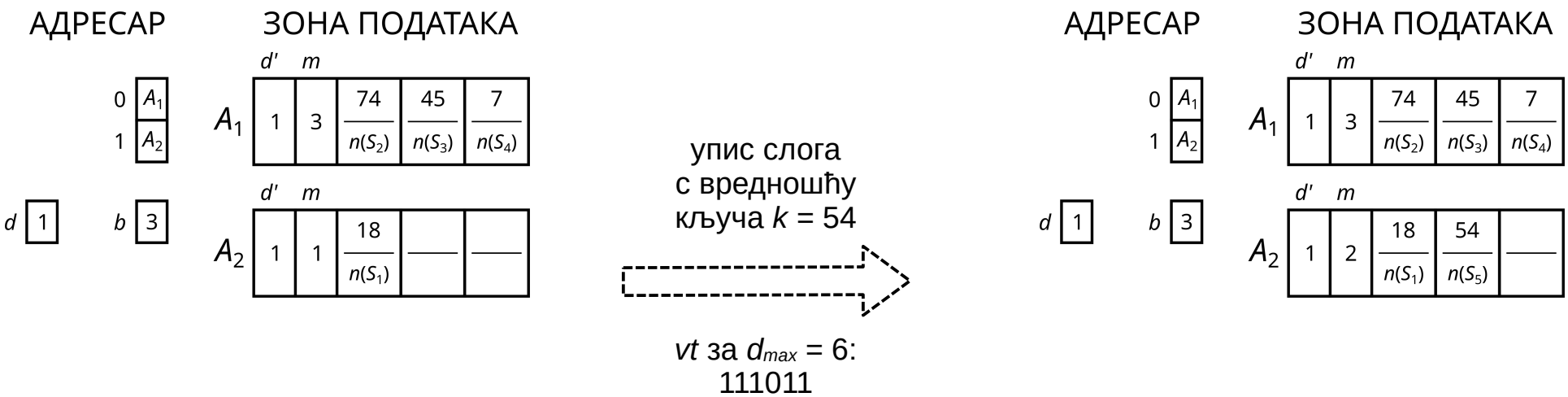
Прошириве расуте датотеке

- Формирање
  - пример – корак 4



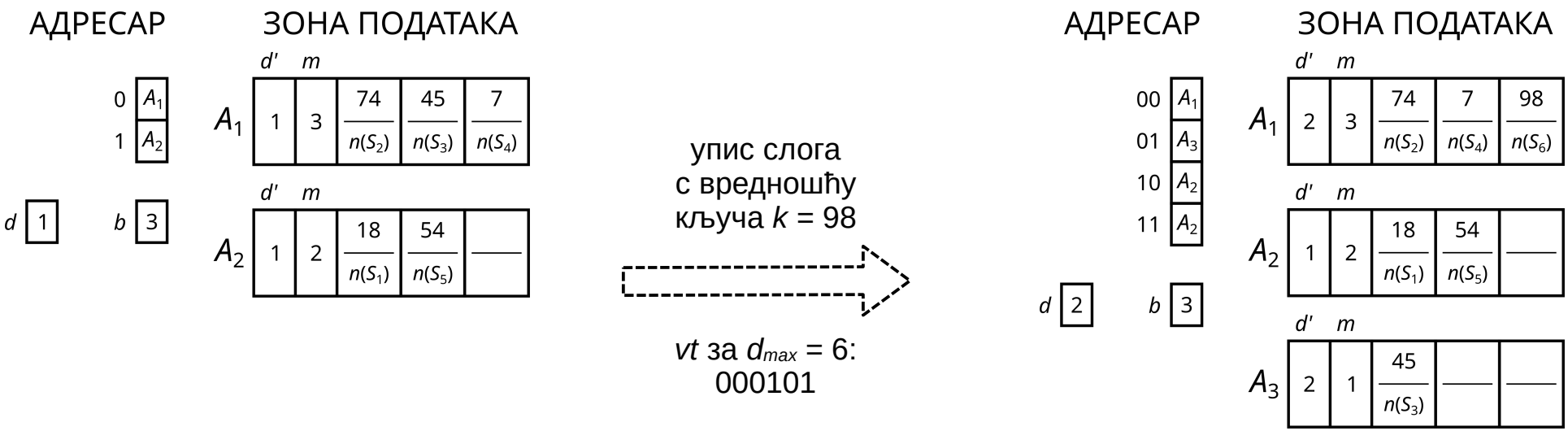
Прошириве расуте датотеке

- Формирање
  - пример – корак 5



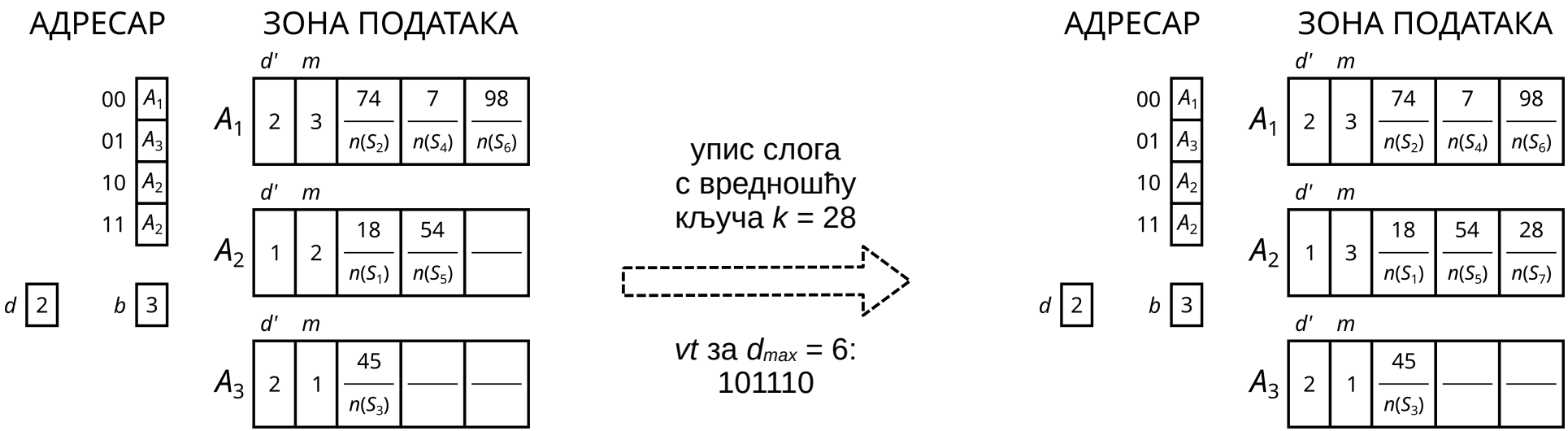
Прошириве расуте датотеке

- Формирање
  - пример – корак 6



Прошириве расуте датотеке

- Формирање
  - пример – корак 7



Прошириве расуте датотеке

- Формирање
  - пример – корак 8

АДРЕСАР

00	A_1
01	A_3
10	A_2
11	A_2

d 2 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	2	3	74 / $n(S_2)$	7 / $n(S_4)$	98 / $n(S_6)$
A_2	1	3	18 / $n(S_1)$	54 / $n(S_5)$	28 / $n(S_7)$
A_3	2	1	45 / $n(S_3)$	—	—

упис слога
с вредношћу
кључа k = 87

vt за d_{max} = 6:
101110

АДРЕСАР

00	A_1
01	A_3
10	A_2
11	A_4

d 2 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	2	3	74 / $n(S_2)$	7 / $n(S_4)$	98 / $n(S_6)$
A_2	2	2	28 / $n(S_7)$	87 / $n(S_8)$	—
A_3	2	1	45 / $n(S_3)$	—	—
A_4	2	2	18 / $n(S_1)$	54 / $n(S_5)$	—

Прошириве расуте датотеке

- Формирање
  - пример – корак 9

АДРЕСАР

00	A_1
01	A_3
10	A_2
11	A_4

d 2 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	2	3	74 / $n(S_2)$	7 / $n(S_4)$	98 / $n(S_6)$
A_2	2	2	28 / $n(S_7)$	87 / $n(S_8)$	—
A_3	2	1	45 / $n(S_3)$	—	—
A_4	2	2	18 / $n(S_1)$	54 / $n(S_5)$	—

упис слога
с вредношћу
кључа k = 82

vt за d_{max} = 6:
000110

АДРЕСАР

0000	A_1	1000	A_2
0001	A_6	1001	A_2
0010	*	1010	A_2
0011	*	1011	A_2
0100	A_3	1100	A_4
0101	A_3	1101	A_4
0110	A_3	1110	A_4
0111	A_3	1111	A_4

d 4 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	4	1	74 / $n(S_2)$	—	—
A_2	2	2	28 / $n(S_7)$	87 / $n(S_8)$	—
A_3	2	1	45 / $n(S_3)$	—	—
A_4	2	2	18 / $n(S_1)$	54 / $n(S_5)$	—
A_6	4	3	7 / $n(S_4)$	98 / $n(S_6)$	82 / $n(S_9)$

Прошириве расуте датотеке

- Тражење (Mogin, 2008)
  - над вредношћу кључа бива примењена трансформација *h* и након примене одговарајућих корака бива формирана вредност трансформације
    - долази до генерисања вредности трансформације
  - у низу показивача у адресару бива на основу вредности трансформације дужине *d* пронађена адреса одговарајућег бакета
  - ако тражени слог постоји у бакету на пронађеној адреси, тражење се завршава успешно
  - ако тражени слог не постоји у бакету на пронађеној адреси, тражење се завршава неуспешно

Прошириве расуте датотеке

- Тражење – перформансе (Mogin, 2008)
  - број приступа за тражење R (и за успешно и за неуспешно тражење)
    - уз претпоставку да је адресар у оперативној меморији

      $R=1$
      - случај када је одговарајући бакет већ у баферу у оперативној меморији

        $R=0$

Прошириве расуте датотеке

- Обрада (Mogin, 2008)
  - проширива расута датотека у улози водеће датотеке
    - непогодност за употребу у улози основне (прве) водеће датотеке
    - не може бити водећа у режиму редоследне обраде
      - логичке везе између слогова нису меморисане
    - може бити водећа у режиму директне обраде
  - проширива расута датотека у улози обрађиване датотеке
    - може бити обрађивана у режиму редоследне обраде
    - може бити обрађивана у режиму директне обраде

Прошириве расуте датотеке

- Обрада – перформансе (Mogin, 2008)
  - проширива расута датотека у улози обрађиване датотеке
    - укупни број приступа обрађиваној проширивој расутој датотеци R_{uk}

      $$R_{uk} = N_v^u + N_v^n$$

      - исто и у режиму редоследне и у режиму директне обраде
      - број приступа за тражење слога, уз претпоставку да је адресар у оперативној меморији

        $$R = 1$$

      - водећа датотека садржи N_v слогова

        $$N_v = N_v^u + N_v^n$$

        - N_v^u – број слогова водеће датотеке на основу којих долази до успешног тражења у обрађиваној датотеци
        - N_v^n – број слогова водеће датотеке на основу којих долази до неуспешног тражења у обрађиваној датотеци

Прошириве расуте датотеке

- Ажурирање (Mogin, 2008)
  - упис
    - подразумева неуспешно тражење
    - извођење уписа слога по моделу како тече упис при формирању
      - треба покушати уписивање новог слога у бакет који одговара генерисаној вредности трансформације
        - ако има слободног места у одговарајућем бакету, слог ће бити тамо уписан
          - случај простог уписа
        - ако нема слободног места у одговарајућем бакету, треба покушати дељење бакета
          - случај уписа с дељењем бакета и проширењем низа показивача
          - случај уписа с дељењем бакета и без проширења низа показивача

Прошириве расуте датотеке

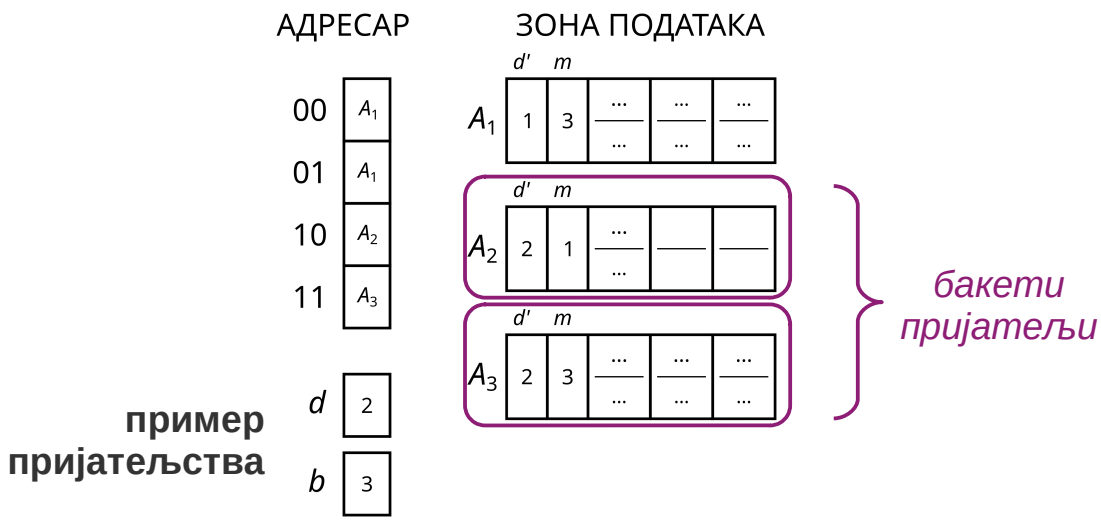
- Ажурирање (Mogin, 2008)
  - модификација
    - као у случају модификације за серијску или секвенцијалну организацију, али уз ослањање на тражење које се користи у проширивим расутим датотекама

Прошириве расуте датотеке

- Ажурирање (Mogin, 2008)
  - брисање
    - подразумева успешно тражење
    - треба уклонити постојећи слог из бакета који одговара генерисаној вредности трансформације
      - може доћи до премештања слогова
      - може доћи до провере стања неког додатног бакета
      - може доћи до смањења укупног броја бакета
      - може доћи до смањења адресара

Прошириве расуте датотеке

- Ажурирање (Mogin, 2008)
  - брисање
    - **бакети пријатељи**
      - два бакета су пријатељи ако су испуњени одговарајући услови
        - на та два бакета показују два показивача чији се индекси разликују само по нултом биту
        - $d' = d$ за та два бакета
        - $d > 0$



Прошириве расуте датотеке

- Ажурирање (Mogin, 2008)
  - брисање
    - три карактеристична случаја брисања слога
      - **1) просто брисање**
      - **2) брисање са спајањем бакета и без сужења низа показивача**
      - **3) брисање са спајањем бакета и сужењем низа показивача**

Прошириве расуте датотеке

- Ажурирање (Mogin, 2008)
  - брисање
    - 1) просто брисање
      - када је слог за брисање у бакету с *m* > 1 слогова, а додатно бива утврђено да је збирни број слогова за тај бакет и пријатељски бакет већи од *b*
        - слог за брисање бива уклоњен из свог бакета
          - ако слог за брисање није последњи
            - слогови који су у бакету иза слога за брисање бивају померени за једну локацију налево
          - умањење *m* за један
        - након уклањања, потребна провера стања на нивоу посматраних бакета пријатеља
          - ако збирни број слогова није већи од *b*, долази и до спајања бакета, што је засебан случај

Прошириве расуте датотеке

- Ажурирање
  - брисање
    - 1) просто брисање
      - пример

$k(S) \in [0,99]$ $h(k(S)) = 1 + k(S) \pmod{59}$

АДРЕСАР

00	A_1
01	A_3
10	A_2
11	A_4

d 2 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	2	3	74 ($n(S_2)$)	7 ($n(S_4)$)	98 ($n(S_6)$)
A_2	2	2	28 ($n(S_7)$)	87 ($n(S_8)$)	—
A_3	2	2	45 ($n(S_3)$)	37 ($n(S_9)$)	—
A_4	2	2	18 ($n(S_1)$)	54 ($n(S_5)$)	—

брисање слога
с вредношћу
кључа k = 74

vt за d_{max} = 6:
000010

$k(S) \in [0,99]$ $h(k(S)) = 1 + k(S) \pmod{59}$

АДРЕСАР

00	A_1
01	A_3
10	A_2
11	A_4

d 2 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	2	2	7 ($n(S_4)$)	98 ($n(S_6)$)	—
A_2	2	2	28 ($n(S_7)$)	87 ($n(S_8)$)	—
A_3	2	2	45 ($n(S_3)$)	37 ($n(S_9)$)	—
A_4	2	2	18 ($n(S_1)$)	54 ($n(S_5)$)	—

Прошириве расуте датотеке

- Ажурирање (Mogin, 2008)
  - брисање
    - 2) брисање са спајањем бакета и без сужења низа показивача
      - када након уклањања слога из бакета важи да збирни број слогова за тај бакет и пријатељски бакет није већи од *b*
        - спајање бакета
          - посматрани бакети пријатељи бивају спојени
          - у бакету насталом спајањем вредност *d'* бива умањена за 1
          - у бакету насталом спајањем важи $m \leq b$, а вредност *m* одговара броју актуелних слогова у бакету
          - показивачи на посматране бакете пријатеље бивају промењени да показују на бакет настао спајањем
        - након спајања, потребна провера стања адресара
          - ако на сваки бакет показују најмање по два показивача, долази и до сужења низа показивача, што је засебан случај

Прошириве расуте датотеке

- Ажурирање
  - брисање
    - 2) брисање са спајањем бакета и без сужења низа показивача
      - пример

$k(S) \in [0,99]$ $h(k(S)) = 1 + k(S) \pmod{59}$

АДРЕСАР

00	A_1
01	A_3
10	A_2
11	A_4

d 2 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	2	2	7 / $n(S_4)$	98 / $n(S_6)$	___
A_2	2	2	28 / $n(S_7)$	87 / $n(S_8)$	___
A_3	2	2	45 / $n(S_3)$	37 / $n(S_9)$	___
A_4	2	2	18 / $n(S_1)$	54 / $n(S_5)$	___

брисање слога
с вредношћу
кључа k = 54

vt за d_{max} = 6:
111011

$k(S) \in [0,99]$ $h(k(S)) = 1 + k(S) \pmod{59}$

АДРЕСАР

00	A_1
01	A_3
10	A_2
11	A_2

d 2 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	2	2	7 / $n(S_4)$	98 / $n(S_6)$	___
A_2	1	3	28 / $n(S_7)$	87 / $n(S_8)$	18 / $n(S_1)$
A_3	2	2	45 / $n(S_3)$	37 / $n(S_9)$	___

Прошириве расуте датотеке

- Ажурирање (Mogin, 2008)
  - брисање
    - 3) брисање са спајањем бакета и сужењем низа показивача
      - када након уклањања слога и спајања бакета важи да на сваки бакет показују најмање по два показивача
        - *d* бива умањено за 1 и дужина низа показивача у адресару бива преполовљена
          - свака два елемента низа показивача чији се индекси разликују само по нултом биту биће трансформисана у један

Прошириве расуте датотеке

- Ажурирање
  - брисање
    - 3) брисање са спајањем бакета и сужењем низа показивача
      - пример

$k(S) \in [0,99]$ $h(k(S)) = 1 + k(S) \pmod{59}$

АДРЕСАР

00	A_1
01	A_3
10	A_2
11	A_2

d 2 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	2	2	7 / $n(S_4)$	98 / $n(S_6)$	

	d'	m			
A_2	1	3	28 / $n(S_7)$	87 / $n(S_8)$	18 / $n(S_1)$

	d'	m			
A_3	2	2	45 / $n(S_3)$	37 / $n(S_9)$	

брисање слога
с вредношћу
кључа k = 98

vt за d_{max} = 6:
000101

$k(S) \in [0,99]$ $h(k(S)) = 1 + k(S) \pmod{59}$

АДРЕСАР

0	A_1
1	A_2

d 1 b 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	1	3	7 / $n(S_4)$	45 / $n(S_3)$	37 / $n(S_9)$

	d'	m			
A_2	1	3	28 / $n(S_7)$	87 / $n(S_8)$	18 / $n(S_1)$

Прошириве расуте датотеке

- Ажурирање (Mogin, 2008)
  - брисање
    - неоптимално коришћење меморијског простора
      - након сужења низа показивача у поступку брисања може се десити да нека два бакета постану пријатељи и да им је збирни број слогова тачно *b*
        - ти бакети нису спојени јер пре сужења нису били пријатељи, а након сужења нема информације о томе за које бакете пријатеље би требало извршити проверу за могућност спајања
        - спајањем тих бакета меморијски простор би био боље искоришћен, јер би слогови тих бакета могли бити смештени у само један бакет

Прошириве расуте датотеке

- Ажурирање
  - брисање
    - неоптимално коришћење меморијског простора
      - пример

$k(S) \in [0,99] \quad h(k(S)) = 1 + k(S) \pmod{59}$

АДРЕСАР

000	A_1
001	A_5
010	A_3
011	A_3
100	A_2
101	A_2
110	A_4
111	A_4

d 3 $\quad b$ 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	3	3	74 / $n(S_2)$	7 / $n(S_4)$	98 / $n(S_6)$
A_2	2	2	28 / $n(S_7)$	87 / $n(S_8)$	—
A_3	2	2	45 / $n(S_3)$	37 / $n(S_9)$	—
A_4	2	1	18 / $n(S_1)$	—	—
A_5	3	1	11 / $n(S_{10})$	—	—

брисање слога
с вредношћу
кључа k = 11

vt за d_{max} = 6:
001100

$k(S) \in [0,99] \quad h(k(S)) = 1 + k(S) \pmod{59}$

АДРЕСАР

00	A_1
01	A_3
10	A_2
11	A_4

d 2 $\quad b$ 3

ЗОНА ПОДАТАКА

	d'	m			
A_1	2	3	74 / $n(S_2)$	7 / $n(S_4)$	98 / $n(S_6)$
A_2	2	2	28 / $n(S_7)$	87 / $n(S_8)$	—
A_3	2	2	45 / $n(S_3)$	37 / $n(S_9)$	—
A_4	2	1	18 / $n(S_1)$	—	—

бакети пријатељи
с укупно b слогова

Прошириве расуте датотеке

- Ажурирање (Mogin, 2008)
  - брисање
    - неоптимално коришћење меморијског простора
      - нека решења
        - након сужења низа показивача проверавање да ли има бакета пријатеља који могу бити спојени и, ако има, извођење спајања
          - у најнеповољнијем случају може бити потребно приступање свим бакетима
        - примена измењене дефиниције бакета пријатеља
          - два бакета су пријатељи ако су испуњени другачији услови
            - на те бакете показују два показивача чији индекси се слажу по водећих d' - 1 битова а не слажу се по бар једном од преосталих битова
            - ти бакети имају исту локалну дужину вредности трансформације d'
            - $d > 0$

Прошириве расуте датотеке

- Ажурирање – перформансе (Mogin, 2008)
  - упис
    - број приступа за успешан упис R_i
      - најнеповољнији случај, уз претпоставку да је адресар у оперативној меморији

        $2 \leq R_i \leq 3$
        - разлике у броју приступа због могућег формирања додатног бакета
  - брисање
    - број приступа за успешно брисање R_d
      - најнеповољнији случај, уз претпоставке да постоје бар два бакета и да је адресар у оперативној меморији

        $R_d = 3$

Прошириве расуте датотеке

- Одлике и примена (Mogin, 2008)
  - датотека се проширује и сужава зависно од броја актуелних слогова
    - очекивани број бакета $\overline{B}$

$$\overline{B} \approx \frac{N}{b \ln 2}$$

  - нема слогова прекорачилаца
  - број приступа за тражење слогова не зависи од величине датотеке и практично гледано најмањи је могућ
  - добра погодност за директну обраду
  - без посебне погодности за редоследну обраду

Прошириве расуте датотеке

- Одлике и примена (Mogin, 2008)
  - примена
    - у пракси битно мања популарност у односу на датотеке с *B*-стаблима
      - идеја о дељењу бакета преузета из домена *B*-стабала
      - расуте датотеке с динамичком трансформацијом у односу на датотеке с *B*-стаблима нуде већу ефикасност тражења случајно одабраног слога
    - има система за управљање базама података који подржавају изградњу и употребу физичких структура заснованих на принципима динамичких расутих датотека

Садржај

- Увод
- Прошириве расуте датотеке
- **Ресурси**

Ресурси

- Извори и литература
  - Pavle Mogin. Strukture podataka i organizacija datoteka. 3. izdanje. Računarski fakultet (Beograd, Srbija), CET (Beograd, Srbija). 2008.
    - Glava 14. Dinamička rasuta organizacija datoteke

Ресурси

- Додатни ресурси
  - R. Fagin, J. Nievergelt, N. Pippenger, H. R. Strong. Extendible Hashing—A Fast Access Method for Dynamic Files. ACM Transactions on Database Systems. 1979; 4(3); 315–344.
  - R. J. Enbody, H. C. Du. Dynamic Hashing Schemes. ACM Computing Surveys. 1988; 20(2); 85–113.