

СУФИКСЕН АВТОМАТ

НАЦИОНАЛНА ЛАГЕР-ШКОЛА ПО ИНФОРМАТИКА, 25.08.2026, ГАБРОВО

Изготвил: Илиян Йорданов

Низови структури от данни

Основни:

Низови структури от данни

Основни:

- trie – търсене на дума в речник
- Ахо-Корасик – намиране на срещания на речник в текст

По-мощни:

Низови структури от данни

Основни:

- trie – търсене на дума в речник
- Ахо-Корасик – намиране на срещания на речник в текст

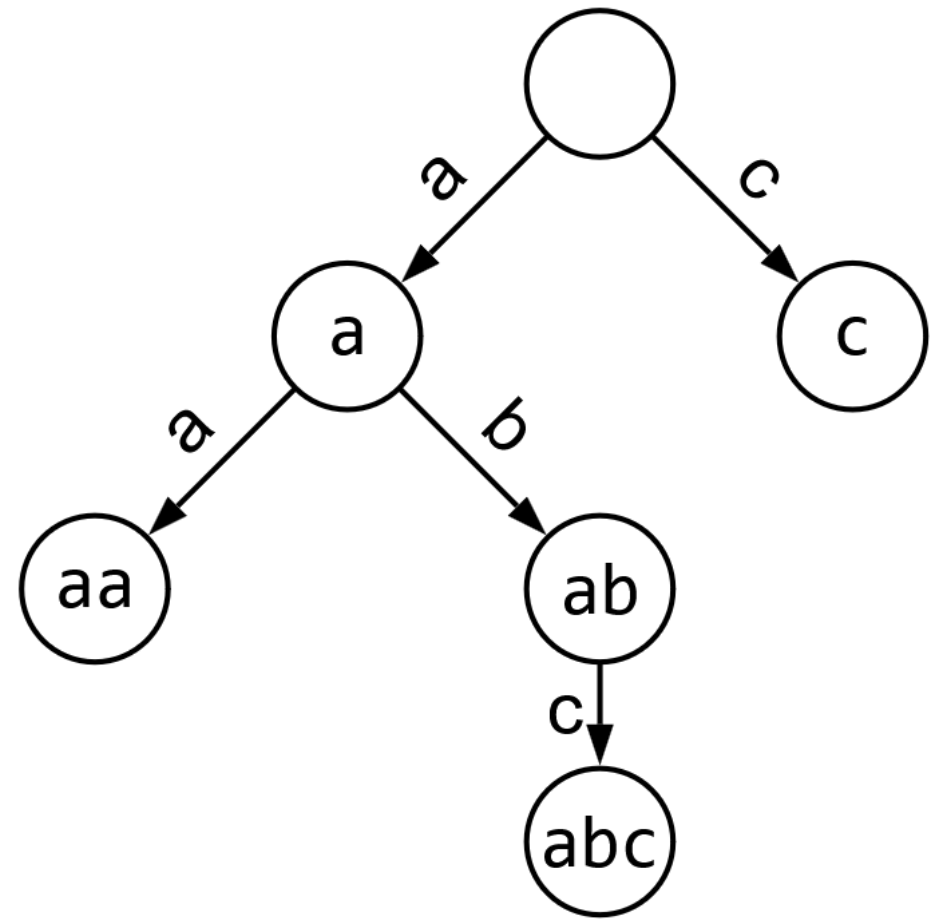
По-мощни:

- суфиксен масив – по-бавно строене, проста логика и код
- суфиксно дърво – бързо строене, сложна логика и код
- суфиксен автомат – бързо строене, сравнително проста логика и прост код

Какво представлява автомата?

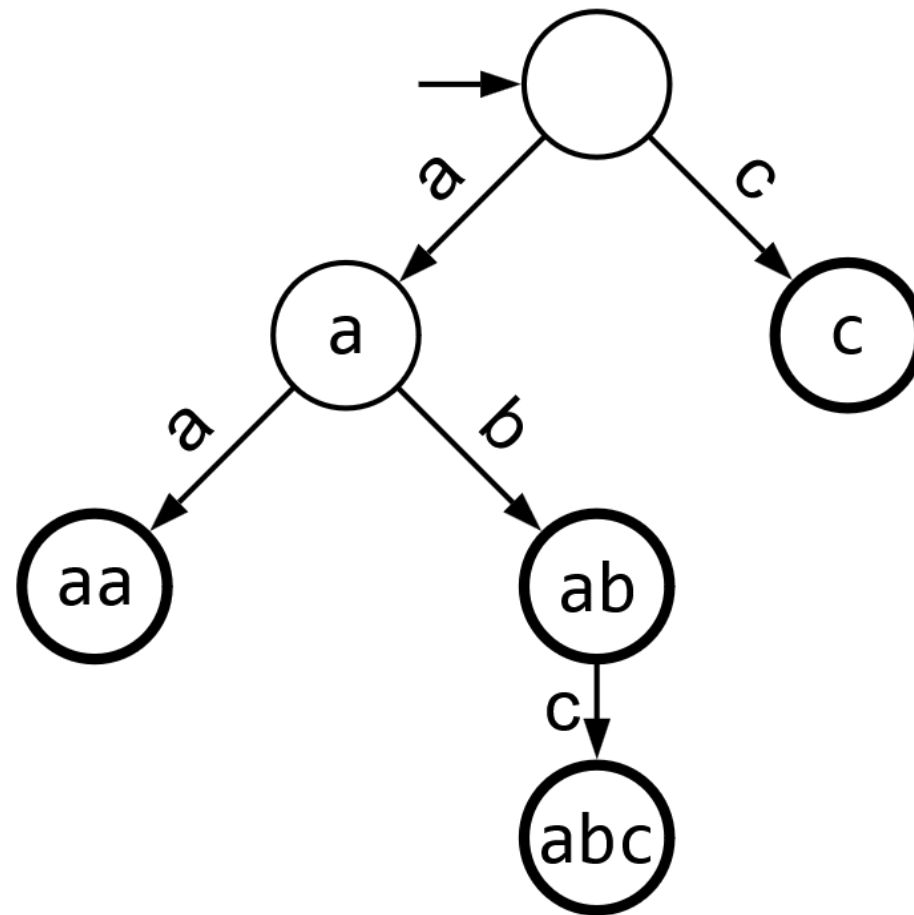
Един пример е trie.

Нека имаме думите: aa, ab, abc, c.



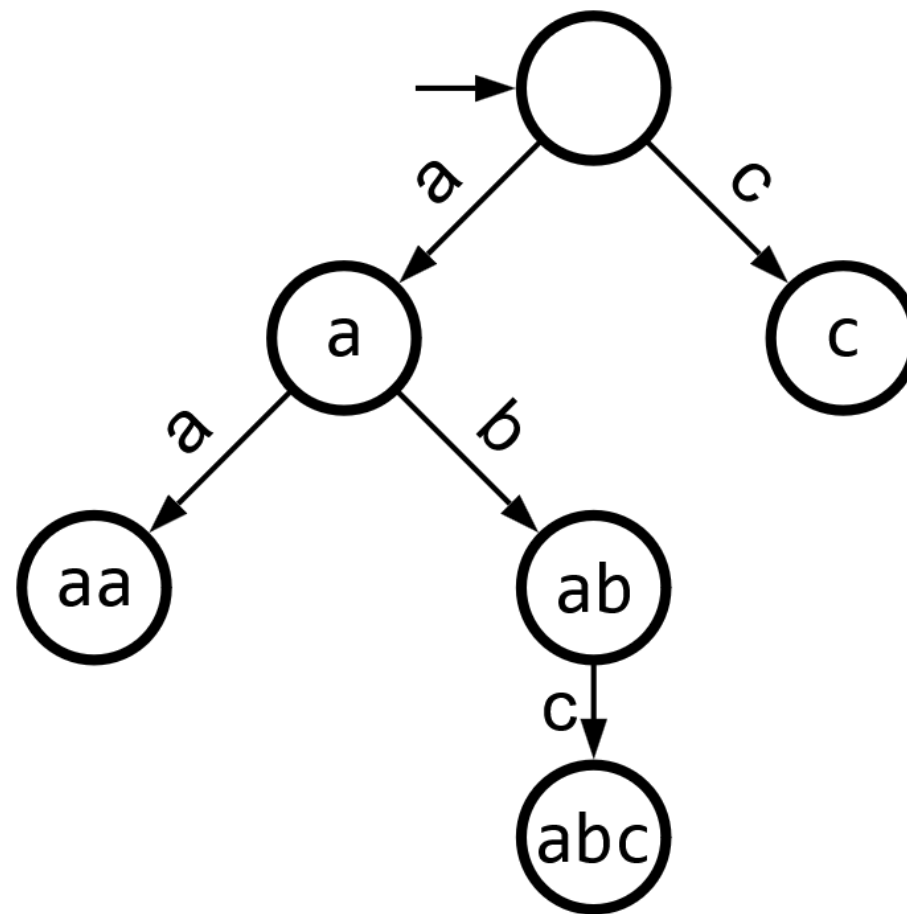
Какво представлява автомата?

- върхове $\rightarrow$ състояния
- начално и крайни състояния
- ребра $\rightarrow$ преходи
- език = {aa, ab, abc, c}



Какво представлява автомата?

- върхове $\rightarrow$ състояния
- начално и крайни състояния
- ребра $\rightarrow$ преходи
- език = $\{\epsilon, a, aa, ab, abc, c\}$



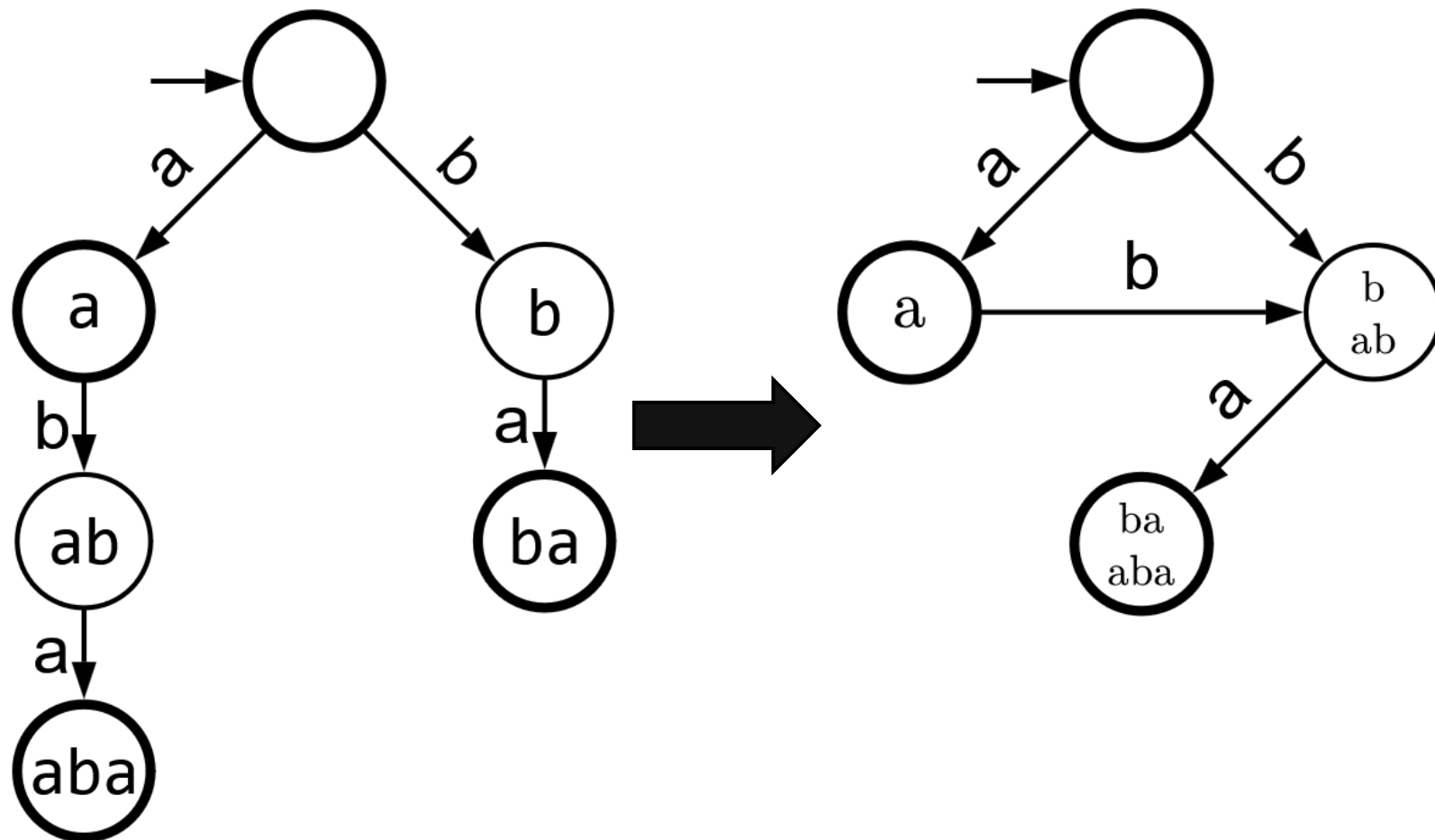
Суфиксен автомат – въведение

Езикът е множеството от суфиксите на даден низ.

Интересуваме се от **минималния** такъв автомат (относно брой състояния). Има единствен такъв автомат за даден низ.

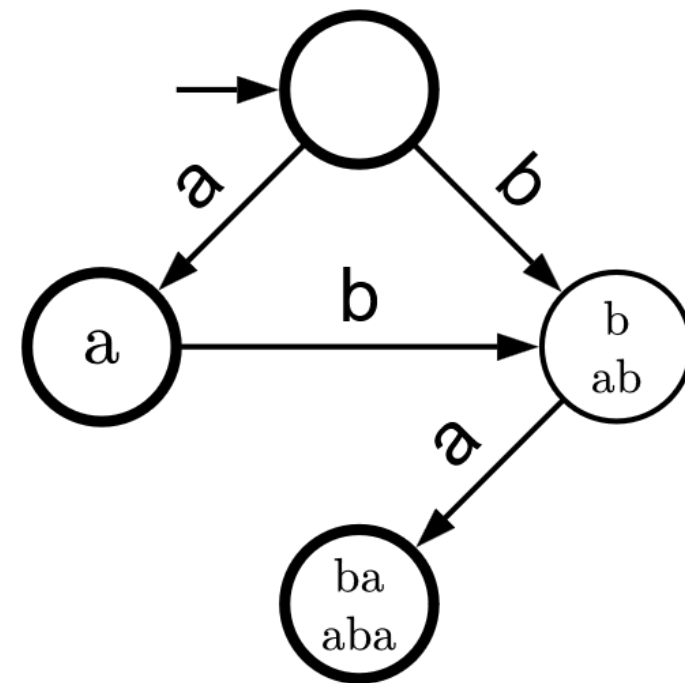
Суфиксен автомат – въведение

низ: aba



Суфиксен автомат – въведение

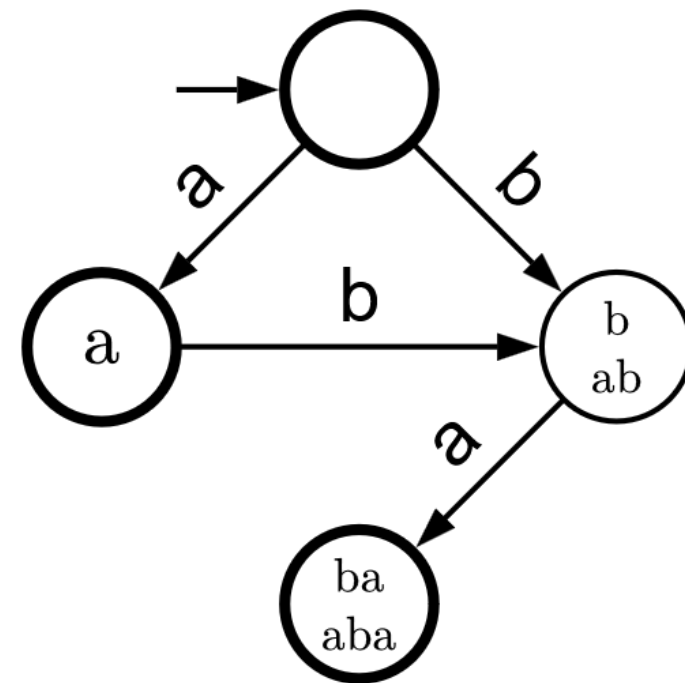
А защо суфикси?



Суфиксен автомат – въведение

А защо суфикси?

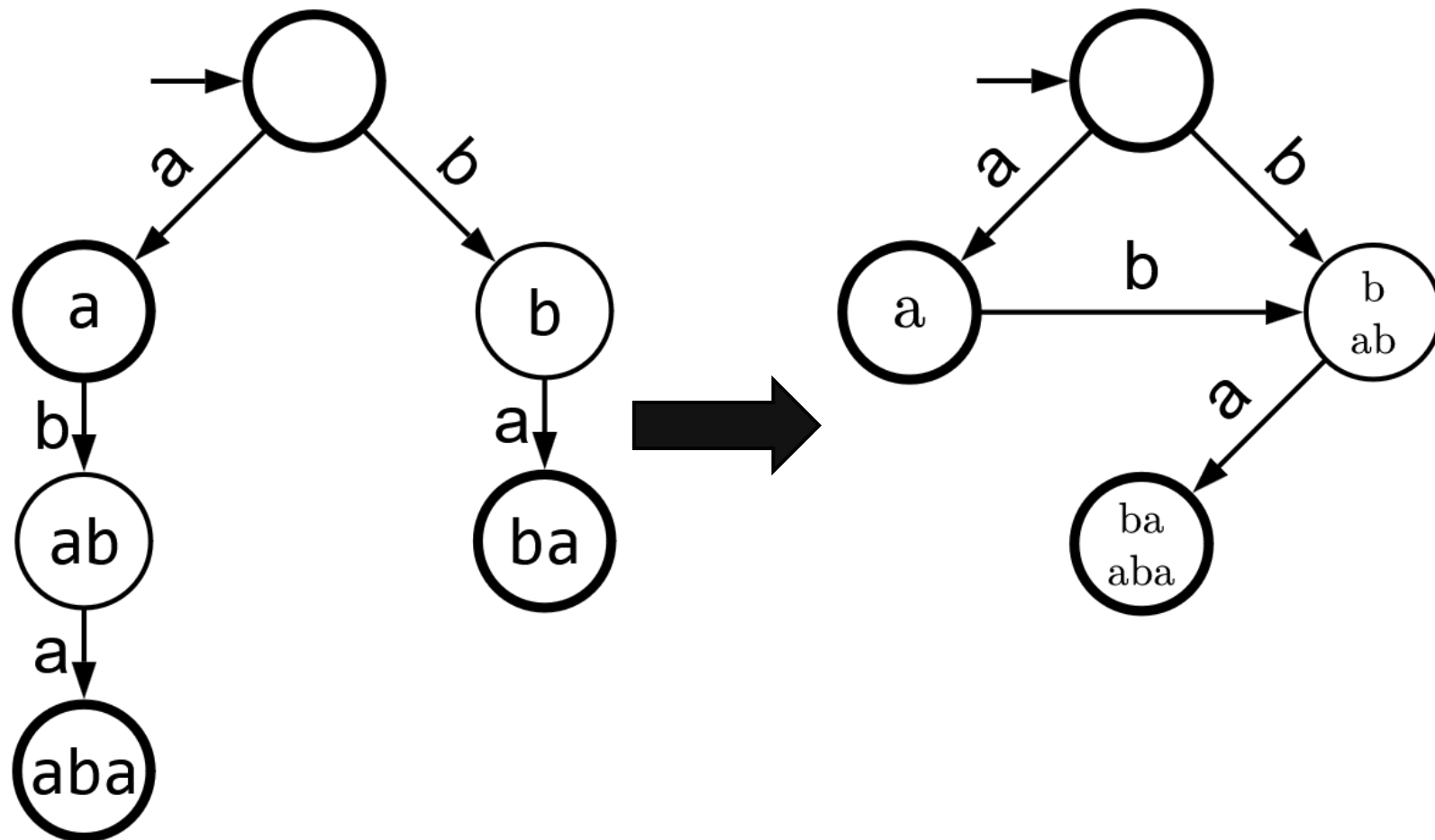
- получаваме всички поднизове
- лесно добавяне на нова буква отзад



Суфиксен автомат – строене I

Обединяват се подобни състояния.

Какъв е принципът?



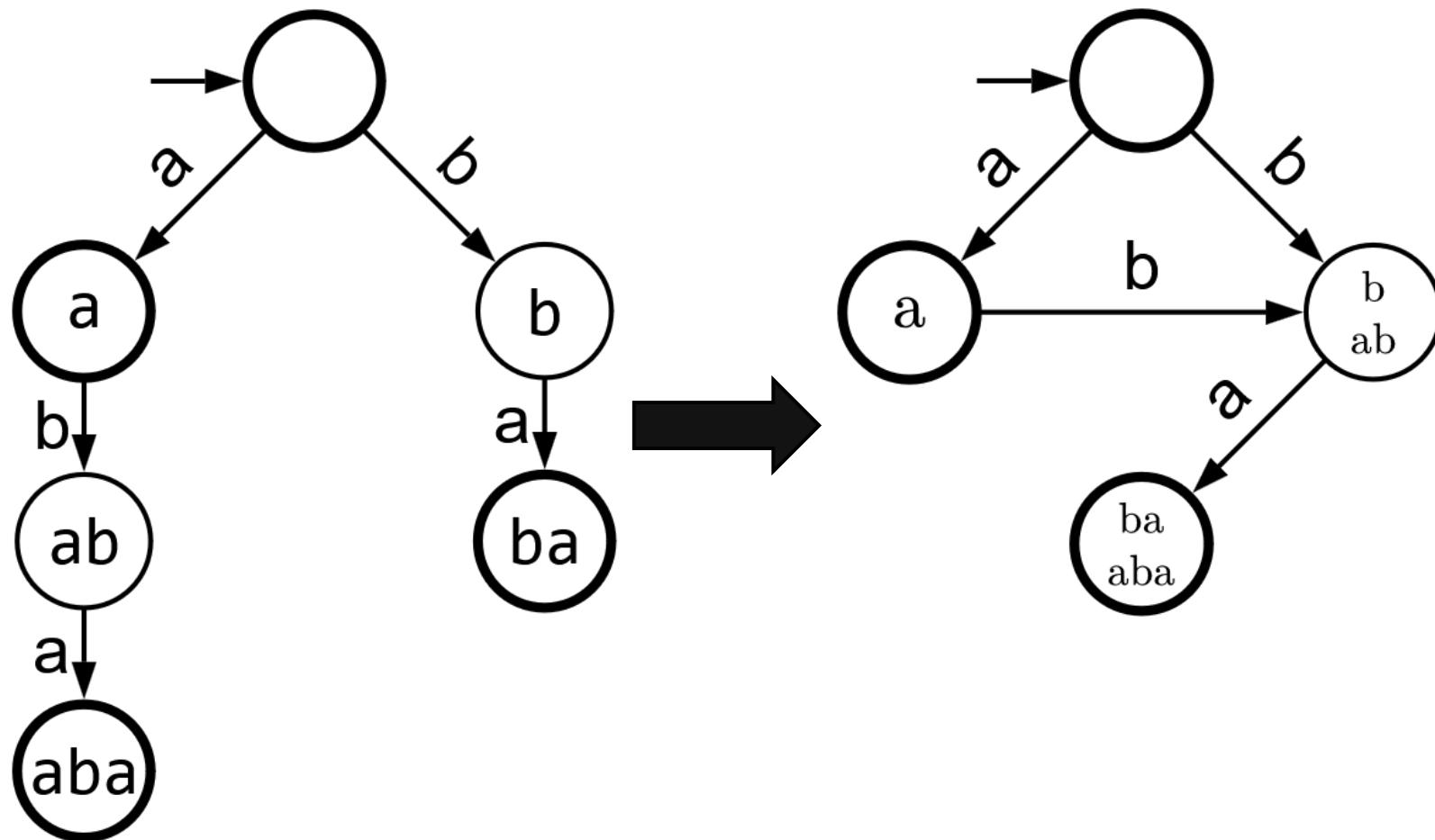
Суфиксен автомат – строене 2

Обединяват се подобни състояния.

Какъв е принципът?

Обединяват се състояния с еднакви продължаващи пътища.

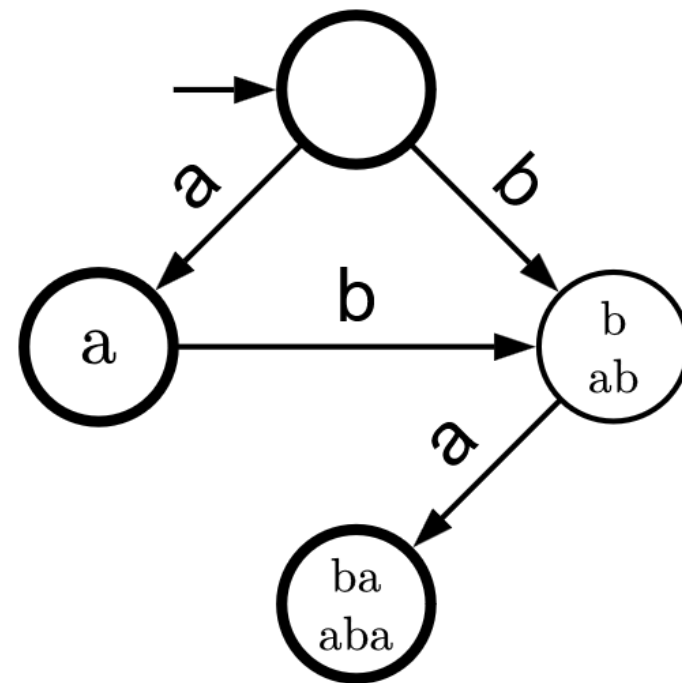
Те съответстват на низове с еднакви завършващи позиции в оригиналния низ. (алтернативно еднакви срещания в низа)



Суфиксен автомат – строене 3

Всяко състояние съответства на низове с еднакви завършващи позиции.
(клас на еквивалентност относно тази релация)

Как можем да представим низовете в едно състояние?

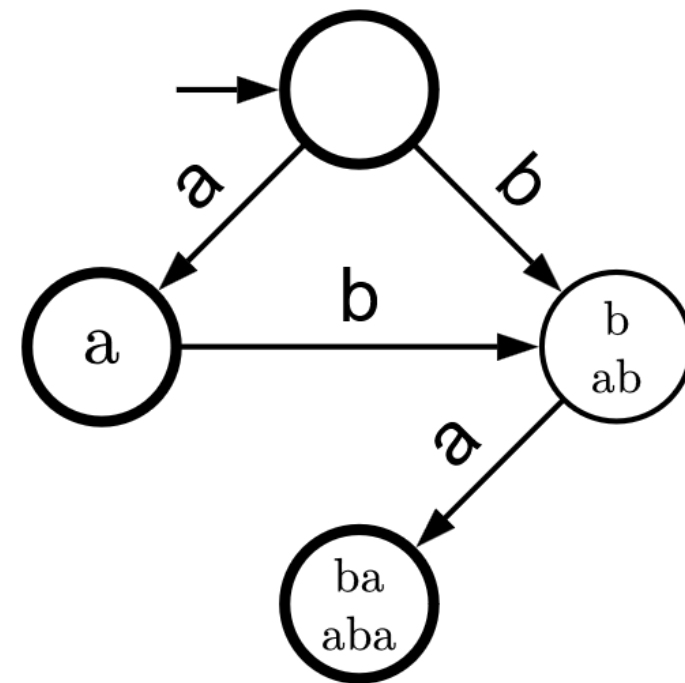


Суфиксен автомат – строене 4

Всяко състояние съответства на низове с еднакви завършващи позиции.
(клас на еквивалентност относно тази релация)

Как можем да представим низовете в едно състояние?

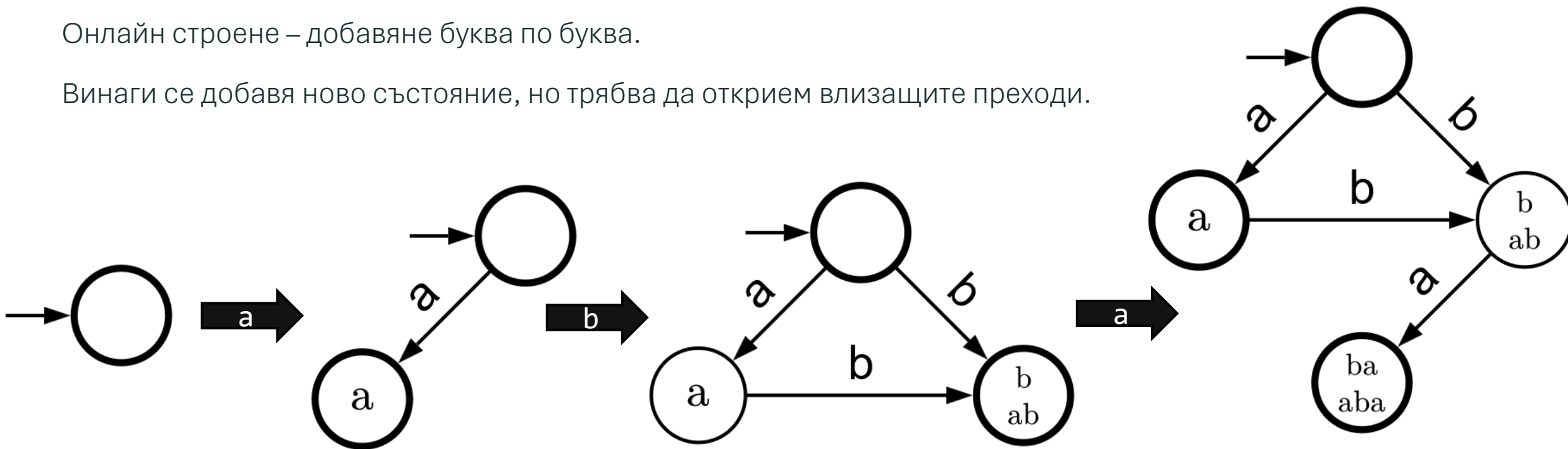
Низовете са последователните суфикси на най-дългия низ в състоянието до достигане на най-късия низ.



Суфиксен автомат – строене 5

Онлайн строене – добавяне буква по буква.

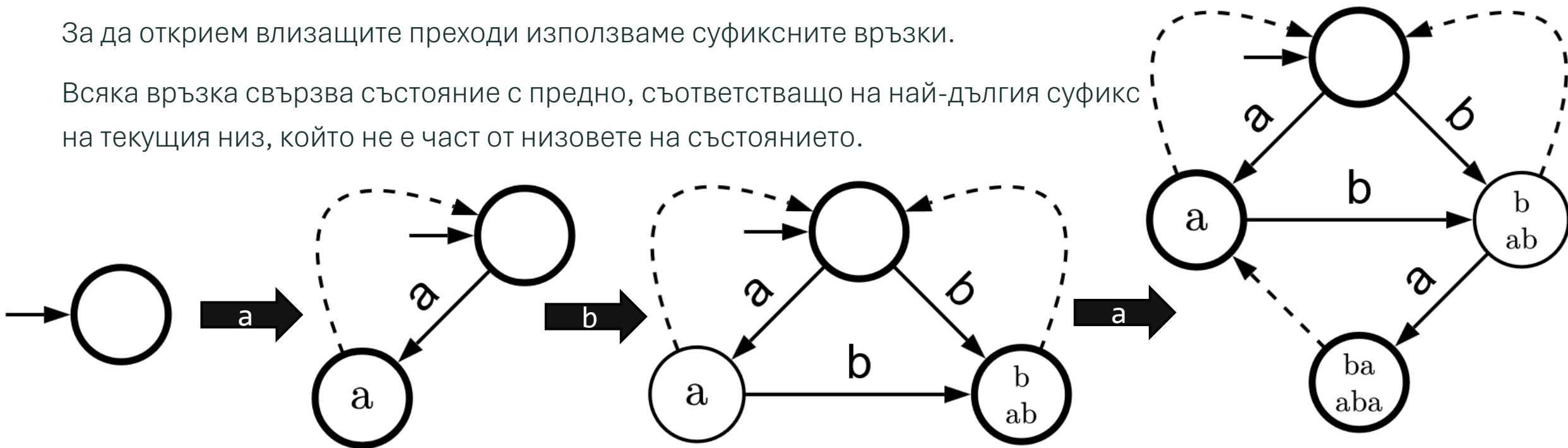
Винаги се добавя ново състояние, но трябва да открием влизащите преходи.



Суфиксен автомат – строене 6

За да открием влизащите преходи използваме суфиксните връзки.

Всяка връзка свързва състояние с предно, съответстващо на най-дългия суфикс на текущия низ, който не е част от низовете на състоянието.



Суфиксен автомат – строене 7

Представяне на състоянията:

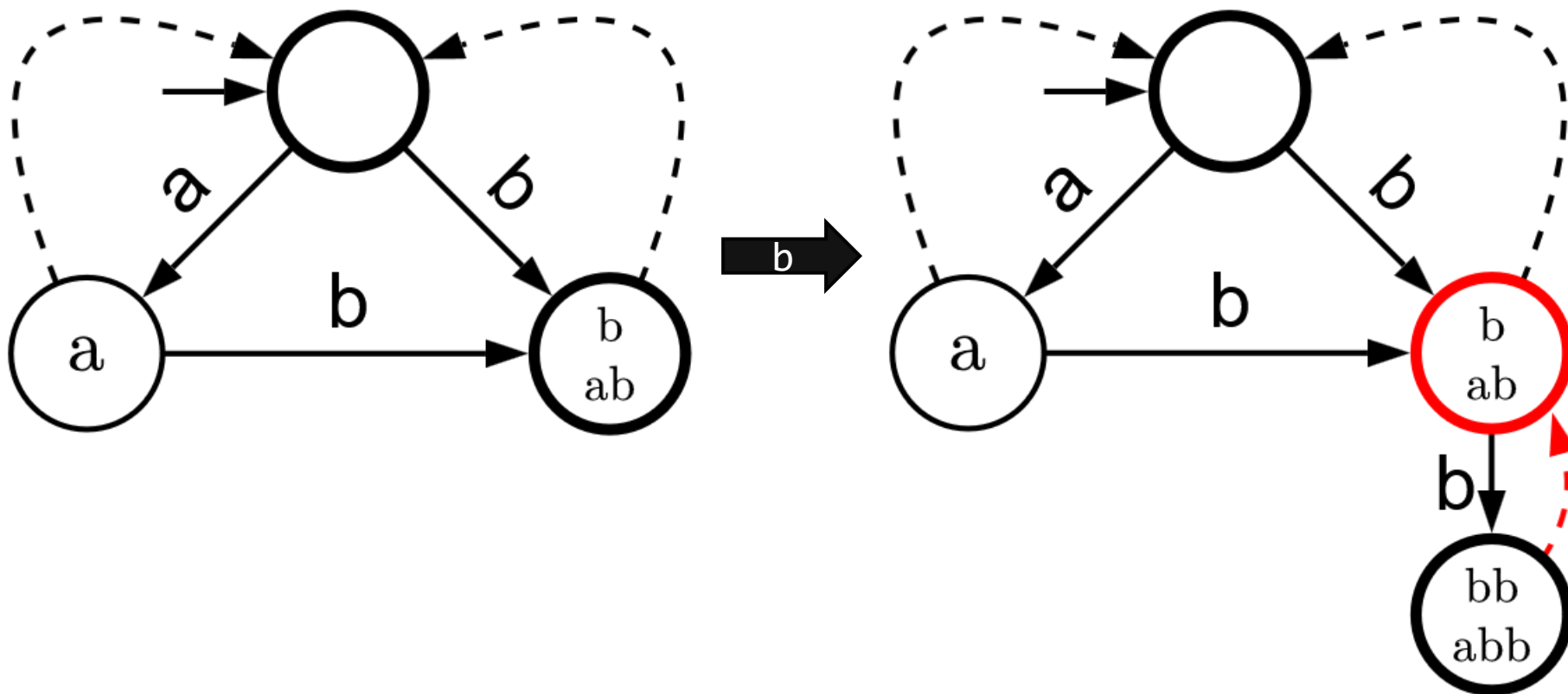
```
struct node {  
    int to[SIGMA];  
    int maxl;  
    int link;  
};
```

Добавянето на нова буква става по следния начин:

```
node sa[2*MAXN-1];  
void add_letter (int c) {  
    int curr=last;  
    last=add_node(sa[last].maxl+1);  
    while ((curr!=-1)&&(sa[curr].to[c]==0)) {  
        sa[curr].to[c]=last;  
        curr=sa[curr].link;  
    }  
    if (curr==-1) {  
        sa[last].link=0;  
        return ;  
    }  
    int nxt=sa[curr].to[c];  
    if (sa[curr].maxl+1==sa[nxt].maxl) sa[last].link=nxt;  
    else {???
```

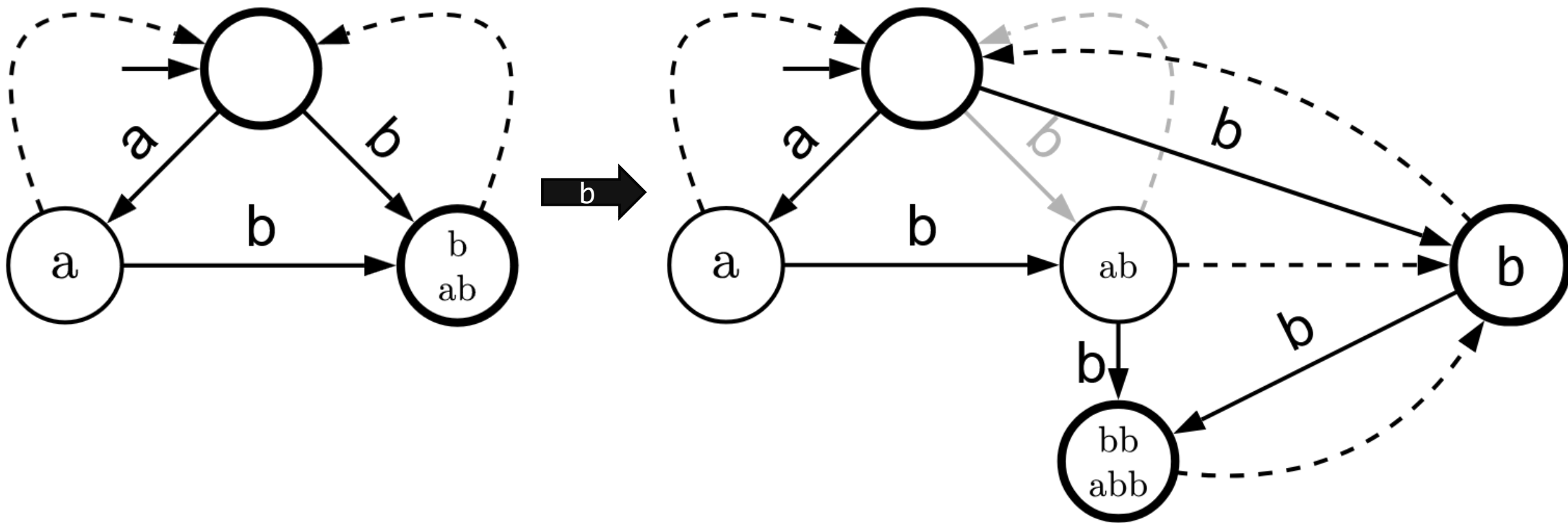
Клониране на състояние – трудният случай

Налага се когато вече имаме срещане на суфикс на текущия низ, който има по-дълго продължение, което не е суфикс.



Клониране на състояние – трудният случай

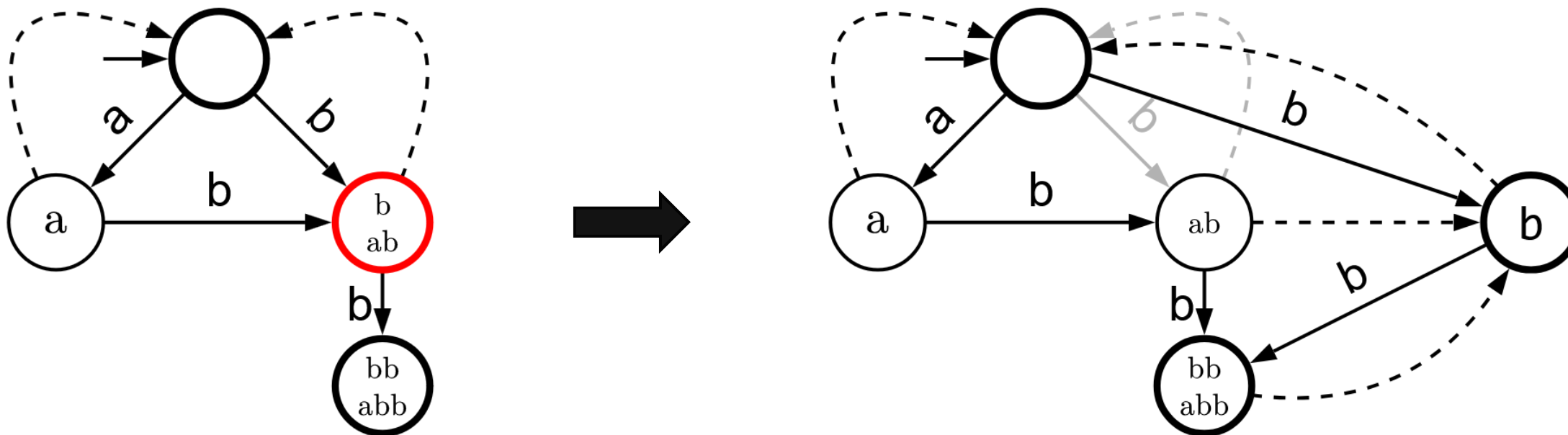
Налага се когато вече имаме срещане на суфикс на текущия низ, който има по-дълго продължение, което не е суфикс.



Клониране на състояние – трудният случай

Клонирането всъщност е отделяне на част от суфиксите в ново състояние. То копира преходите и суфиксната връзка на старото.

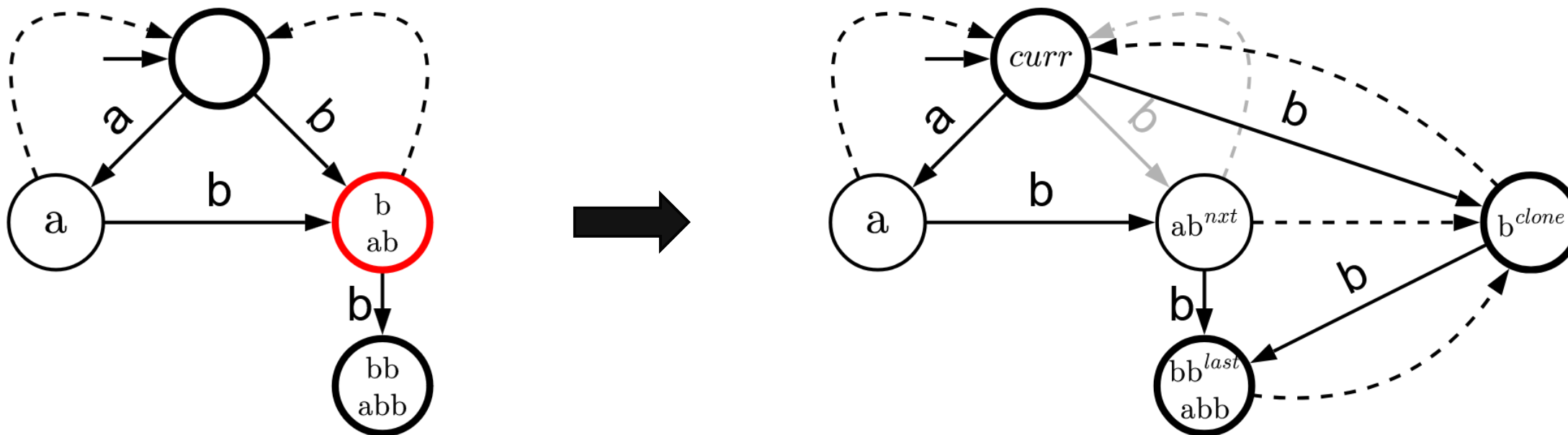
Разликата е във влизащите преходи, защото част от тях трябва да се пренасочат към новото клонирано състояние.



Клониране на състояние – трудният случай

Клонирането става по следния начин:

```
int clone=add_node(sa[curr].maxl+1,sa[nxt]);
sa[nxt].link=clone;
sa[last].link=clone;
while ((curr!=-1)&&(sa[curr].to[c]==nxt)) {
    sa[curr].to[c]=clone;
    curr=sa[curr].link;
}
```



Суфиксен автомат – обобщение на строенето

Пазене на преходите	Време	Памет
Масив	$O(N\Sigma)^*$	$O(N\Sigma)^*$
map	$O(N\log_2\Sigma)^*$	$O(N)^*$
treap	$O(N\log_2\Sigma)$	$O(N)$

Най-големият брой състояния е?

Най-големият брой преходи е?

Суфиксен автомат – обобщение на строенето

Пазене на преходите	Време	Памет
Масив	$O(N\Sigma)^*$	$O(N\Sigma)^*$
map	$O(N\log_2\Sigma)$	$O(N)^*$
treap	$O(N\log_2\Sigma)^*$	$O(N)$

Най-големият брой състояния е $2N - 1$ при низ от вида $abb\dots b$.

Най-големият брой преходи е $3N - 4$ при низ от вида $abb\dots bc$.

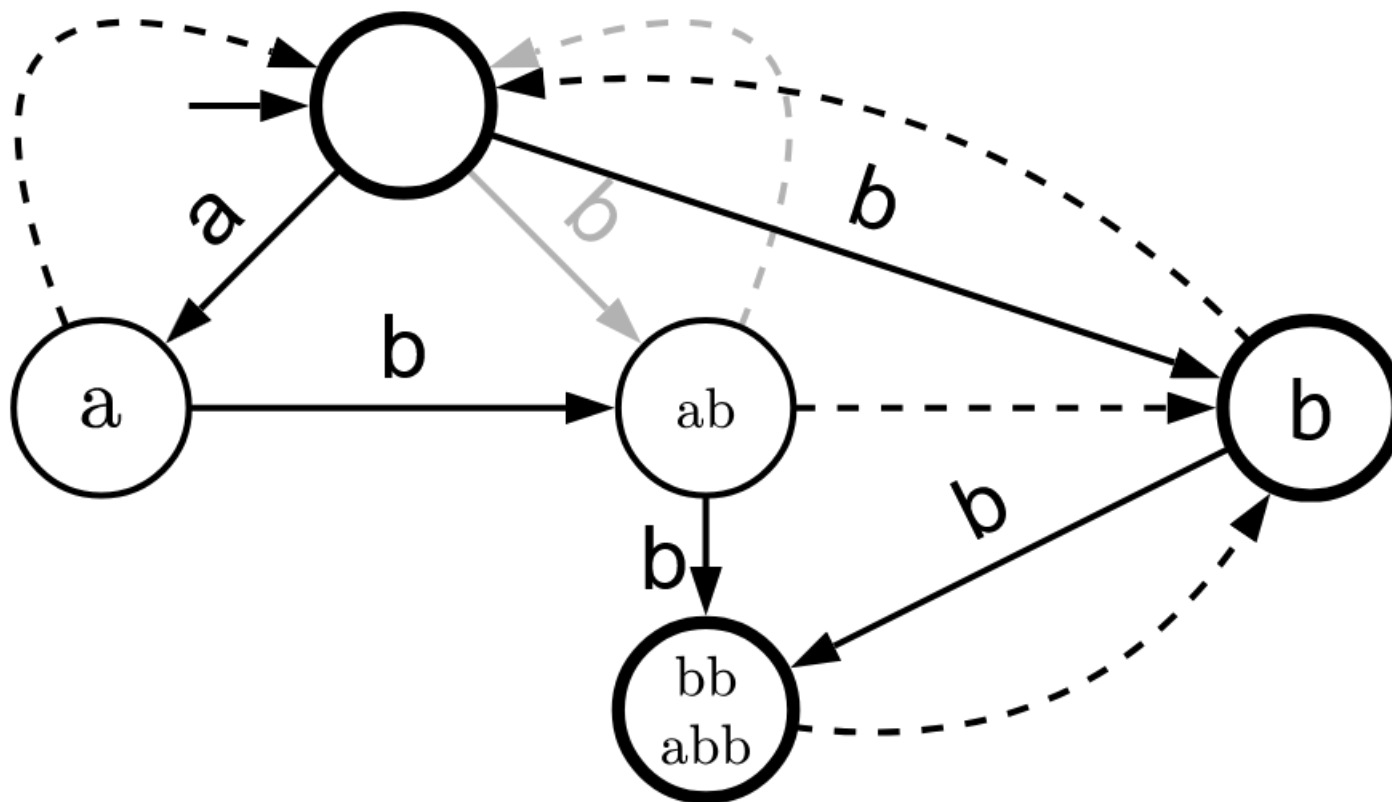
Суфиксен автомат – обобщение на строенето

Има два основни начина на използване на автомата:

- директно като DAG чрез преходите
- като дърво от суфиксните връзки

Тези два начина са удачни за различни динамични:

- брой поднизове (чрез DAG-а)
- брой срещания на низовете в състоянията (чрез дървото)



ЗАДАЧИ И УПРАЖНЕНИЕ

Задача за упражнение – 1

Даден е низ от малки латински букви с дължина N . Трябва да се намери броят различни поднизове във всеки префикс.

Ограничения: $N \leq 2 \cdot 10^5$.

Задача за упражнение – 2

Даден е низ от малки латински букви с дължина N . Трябва да се намери броят различни поднизове във всеки префикс.

Ограничения: $N \leq 2 \cdot 10^5$.

Решение: Броят поднизове на низ спрямо един суфиксен автомат е сумата от броя низове в състоянията. Лесно поддържаеме тази сума при добавянето на нова буква, защото само трябва да добавим броя в новото (последно) състояние. Низовете в последно добавеното състояние са единствените нови поднизове.

Суфиксен автомат вместо КМР – 1

Даден е низ шаблон w с дължина K . Трябва да открием всичките му срещания в низ текст s с дължина N .

Ограничения: $N, K \leq 10^6$.

Суфиксен автомат вместо KMP – 2

Даден е низ шаблон w с дължина K . Трябва да открием всичките му срещания в низ текст s с дължина N .

Ограничения: $N, K \leq 10^6$.

Решение: Построяваме суфиксния автомат за w . След това започваме да обхождаме буквите на s и да се движим по автомата, започвайки от началното състояние. Ако нямаме преход с дадена буква, то използваме суфиксната връзка, докато или не открием преход с буквата, или се върнем в началото (без там да има преход с тази буква). (по този начин състоянието в автомата съответства на най-дългия подниз на w , който е суфикс на текущата позиция на s)

Откриваме срещане на низа, когато достигнем последното добавено състояние.

Сложност: амортизирано $O(K \log_2 \Sigma + N)$ или $O(K \Sigma + N)$. (с KMP амортизирано $O(K + N)$ или $O(K \Sigma + N)$)

Суфиксен автомат за много низове – I

Дадени са k циклични низа, всеки с дължина n символа. Трябва да открием дължината на най-дългия общ подниз. (common A1, НОИ-2, 2023)

Ограничения: $n \leq 10^5, k \leq 10$.

Суфиксен автомат за много низове – 2

Дадени са k циклични низа, всеки с дължина n символа. Трябва да открием дължината на най-дългия общ подниз.

Ограничения: $n \leq 10^5, k \leq 10$.

Решение: Ще заменим всеки низ s с конкатенацията ss , за да не се грижим за цикличността (единствено трябва да внимаваме с отговора накрая). Построяваме суфиксен автомат, който да отговаря за получените низове.

Това става като променим малко функцията за добавяне на символ. Имаме допълнителен случай, при който последното добавено състояние вече има преход с новата буква. Тогава сме в случая за клонирането – ако следващото състояние отговаря точно на текущия низ, то не правим нищо. Иначе трябва да клонираме това състояние, за да получим такова, което отговаря.

Суфиксен автомат за много низове – 3

Накрая за всяко състояние поддържахме маска за индексите на низовете, от които са част низовете в състоянието. За тази цел най-лесно е докато строим автомата, да отбележим в маската за всяко добавено състояние от кой низ е. Защо с това не сме готови?

Суфиксен автомат за много низове – 4

Накрая за всяко състояние поддържахме маска за индексите на низовете, от които са част низовете в състоянието. За тази цел най-лесно е докато строим автомата, да отбележим в маската за всяко добавено състояние от кой низ е.

Не всички състояния имат правилна маска, защото при добавяне на последващите низове, е възможно състоянието, към което сочи суфиксната връзка, да е от предишните низове. Това означава, че накрая трябва да минем през всички състояния и да предадем информацията за маските по суфиксните връзки отдолу-нагоре. За тази цел е достатъчно да обходим състоянията в обратен топологичен ред (по намаляваща дължина) и да предадем информацията на маската на всяко състояние към състоянието от суфиксната връзка.

Суфиксен автомат за много низове – 5

Накрая за всяко състояние поддържахме маска за индексите на низовете, от които са част низовете в състоянието. За тази цел най-лесно е докато строим автомата, да отбележим в маската за всяко добавено състояние от кой низ е.

Не всички състояния имат правилна маска, защото при добавяне на последващите низове, е възможно състоянието, към което сочи суфиксната връзка, да е от предишните низове. Това означава, че накрая трябва да минем през всички състояния и да предадем информацията за маските по суфиксните връзки отдолу-нагоре. За тази цел е достатъчно да обходим състоянията в обратен топологичен ред (по намаляваща дължина) и да предадем информацията на маската на всяко състояние към състоянието от суфиксната връзка.

Сложност: амортизирано $O(kn \log_2 \Sigma)$.

Суфиксен автомат вместо Ахо-Корасик – I

Даден е речник с низове шаблон с обща дължина M и трябва да открием броят на срещанията в низ текст s с дължина N .

Ограничения: $N, M \leq 10^6$.

Суфиксен автомат вместо Ахо-Корасик – 2

Даден е речник с низове шаблон с обща дължина M и трябва да открием броят на срещанията в низ текст s с дължина N .

Ограничения: $N, M \leq 10^6$.

Решение: Отново построяваме суфиксен автомат за всички шаблони. Така ако обхождаме текста буква по буква и едновременно с това се движим по автомата (подобна на един шаблон) – винаги състоянието, на което сме в автомата, ще е най-дългият суфикс спрямо текущата позиция на текста, който е префикс на някой от шаблоните.

Как можем да намираме срещания на шаблоните?

Суфиксен автомат вместо Ахо-Корасик – 3

Решение: Отново построяваме суфиксен автомат за всички шаблони. Така ако обхождаме текста буква по буква и едновременно с това се движим по автомата (подобна на един шаблон) – винаги състоянието, на което сме в автомата, ще е най-дългият суфикс спрямо текущата позиция на текста, който е префикс на някой от шаблоните.

Срещания на шаблоните намираме, като допълнително за всяко състояние сме преизчислили броят думи, които се срещат като суфикс на най-дългия низ в състоянието. Това става като в началото отбележим за всяко състояние директните срещания на шаблони и след това предадем тази информация отгоре-надолу по суфиксните връзки. (същият алгоритъм както при намирането на речниковите връзки в Ахо-Корасик)

Суфиксен автомат вместо Ахо-Корасик – 4

Решение: Отново построяваме суфиксен автомат за всички шаблони. Така ако обхождаме текста буква по буква и едновременно с това се движим по автомата (подобна на един шаблон) – винаги състоянието, на което сме в автомата, ще е най-дългият суфикс спрямо текущата позиция на текста, който е префикс на някой от шаблоните.

Срещания на шаблоните намираме, като допълнително за всяко състояние сме преизчислили броят думи, които се срещат като суфикс на най-дългия низ в състоянието. Това става като в началото отбележим за всяко състояние директните срещания на шаблони и след това предадем тази информация отгоре-надолу по суфиксните връзки. (същият алгоритъм както при намирането на речниковите връзки в Ахо-Корасик)

Сложност: амортизирано $O(M \log_2 \Sigma + N)$.

Задача Sgame, Ден 1, IAT1 2019 – 1

Даден е низ w с дължина N . Трябва да отговорим на Q заявки от вида (m, k) , чийто отговор е най-големият брой срещания cnt на подниз s , за който е изпълнено:

- $m \leq |s| \leq k$;
- за всички възможни продължения вляво и вдясно u и v , така че $k < |usv|$, usv се среща $< cnt$ на брой пъти в w .

Ограничения: $N \leq 5 \cdot 10^5, Q \leq 10^6$.

Задача Sgame, Ден 1, IAT1 2019 – 2

Даден е низ w с дължина N . Трябва да отговорим на Q заявки от вида (m, k) , чийто отговор е най-големият брой срещания cnt на подниз s , за който е изпълнено:

- $m \leq |s| \leq k$;
- за всички възможни продължения вляво и вдясно u и v , така че $k < |usv|$, usv се среща $< cnt$ на брой пъти в w .

Ограничения: $N \leq 5 \cdot 10^5, Q \leq 10^6$.

Решение: Построяваме суфиксния автомат за w .

Наблюдение: Потенциалните низове, които са отговор са най-дългите низове в състоянията.

Задача Sgame, Ден 1, IAT1 2019 – 3

Даден е низ w с дължина N . Трябва да отговорим на Q заявки от вида (m, k) , които търсят най-големия брой срещания cnt на подниз s , така че:

- $m \leq |s| \leq k$;
- за всички възможни продължения вляво и вдясно u и v , така че $k < |usv|$, usv се среща $< cnt$ на брой пъти в w .

Ограничения: $N \leq 5 \cdot 10^5, Q \leq 10^6$.

Решение: Построяваме суфиксния автомат за w .

Наблюдение: Потенциалните низове, които са отговор са най-дългите низове в състоянията. За дадено състояние по-късите низове се срещат същия брой пъти. От друга страна всяко продължение на най-дългия низ може да се разглежда и като продължение на който да е по-къс низ от състоянието.

Задача Sgame, Ден 1, IAT1 2019 – 4

Решение: Построяваме суфиксния автомат за w .

Наблюдение: Потенциалните низове, които са отговор са най-дългите низове в състоянията. За дадено състояние по-късите низове се срещат същия брой пъти. От друга страна всяко продължение на най-дългия низ може да се разглежда и като продължение на кой да е по-къс низ от състоянието.

Нека разгледаме конкретно състояние i и неговия най-дълъг низ l . Ще означаваме с $cnt[i]$ броят срещания на низовете в състоянието. Можем да забележим, че l не може да се продължи вляво и да се запазят броя срещания. Ако това беше така, то този низ нямаше да е най-дългият в състоянието.

Следователно единствено трябва да проследим за продължения вдясно. Това става като гледаме съседните състояния спрямо излизащите преходи. Тези състояния имат брой срещания, който е $\leq cnt[i]$. Интересен е случаят със съседни състояния, които запазват броя срещания.

Задача Sgame, Ден 1, IAT1 2019 – 5

Решение: Построяваме суфиксния автомат за w .

Наблюдение: Потенциалните низове, които са отговор са най-дългите низове в състоянията. За дадено състояние по-късите низове се срещат същия брой пъти. От друга страна всяко продължение на най-дългия низ може да се разглежда и като продължение на кой да е по-къс низ от състоянието.

Нека разгледаме конкретно състояние i и неговия най-дълъг низ l . Ще означаваме с $cnt[i]$ броят срещания на низовете в състоянието. Можем да забележим, че l не може да се продължи вляво и да се запазят броя срещания. Ако това беше така, то този низ нямаше да е най-дългият в състоянието.

Следователно единствено трябва да проследим за продължения вдясно. Това става като гледаме съседните състояния спрямо излизащите преходи. Тези състояния имат брой срещания, който е $\leq cnt[i]$. Достатъчно е за всяко състояние да изчислим с динамично $maxr[i]$ – максималното продължение вдясно, така че да се запазят броят срещания.

Задача Sgame, Ден 1, IAT1 2019 – 6

Нека разгледаме конкретно състояние i и неговия най-дълъг низ l . Ще означаваме с $cnt[i]$ броят срещания на низовете в състоянието. Можем да забележим, че l не може да се продължи вляво и да се запазят броя срещания. Ако това беше така, то този низ нямаше да е най-дългият в състоянието.

Следователно единствено трябва да проследим за продължения вдясно. Това става като гледаме съседните състояния спрямо излизащите преходи. Тези състояния имат брой срещания, който е $\leq cnt[i]$. Достатъчно е за всяко състояние да изчислим с динамично $maxr[i]$ – максималното продължение вдясно, така че да се запазят броят срещания.

Така най-дългият представител е потенциален отговор $cnt[i]$ за заявки, за които $m \leq |l| \leq maxr[i] \leq k$. Това е стандартна задача за онлайн заявки за максималната стойност за интервали, съдържащи се в интервал. Може да се реши с *merge-sort* дърво или персистентно сегментно дърво.

Сложност: $O(N \log_2 N + Q \log_2^2 N)$ или $O((N + Q) \log_2 N)$.

Задача Sgame, Ден 1, IAT 2019 – уточнения

Смятането на $maxr[i]$ е с динамично в DAG графа от състояния.

Смятането на $cnt[i]$ е с динамично по дървото от суфиксните връзки. Началните състояния са $cnt[i] = 1$ за всяко ново състояние i (без клонираните). Това съответства на отбелязване на позициите за състоянията, съответстващи на префиксите. След това е достатъчно да обходим състоянията отдолу-нагоре, като предадем информацията по суфиксните връзки: $cnt[sa[i].link] += cnt[i]$.

БЛАГОДАРЯ ЗА ВНИМАНИЕТО!

Сготвил: Илиян Йорданов