

روش‌های مبتنی بر محتوا  
Context-based compressing

## فشرده‌سازی اطلاعات

۰۱-۰۰۲-۱۰-۱۴۰

بخش سوم



دانشگاه شهید بهشتی  
پژوهشکده‌ی فضای مجازی

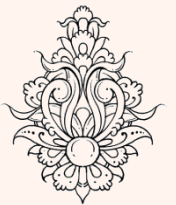
بهار ۱۴۰۲

احمد محمودی ازناوه

# فهرست مطالب

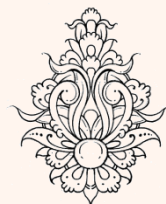
- معرفی
- پیش‌بینی متن
- پیش‌بینی با تطابق محدود (ppm)  
– مثال

- Burrows-Wheeler Transform
- Move To Front
- Run Length Encoding
- Associative Coder of Buyanovsky



# معرفی

- در صورت وجود عدم توازن احتمال وقوع داده (skewness)، نسبت فشرده‌سازی افزایش می‌یابد.
- **احتمال توزیع داده‌ها** در صورت در اختیار داشتن **محتوا** (در صورت داشتن **همبستگی بالا**)، دارای **عدم توازن بیشتری** خواهد بود.
  - محتوا باید در کدگذار و کدگشا در اختیار باشد.
- در این دسته از روش‌ها فرض بر این است که در مورد ویژگی‌های آماری دادگان اطلاعات کمی در اختیار داریم.
- از محتوای داده‌های گذشته برای کدگذاری استفاده می‌شود.
  - این شیوه وفقی است.
  - بر اساس **مدل‌سازی** محتوا عمل می‌کند.
  - به جای این هر نماد به صورت مستقل بررسی شود، با توجه به محتوا مورد بررسی قرار می‌گیرد.
  - برای **کدگذاری** عمدتاً از کدگذاری مناسباتی و یا کدگذاری Huffman استفاده می‌شود.



# پیش‌بینی متن - آزمایش ۱

- یک متن در نظر گرفته می‌شود و از یک کاربر می‌خواهیم (ادامه‌ی) متن را حدس بزند، در صورت حدس اشتباه نماد درست به او گفته شده و کار ادامه می‌یابد.

(1) THE ROOM WAS NOT VERY LIGHT A SMALL OBLONG

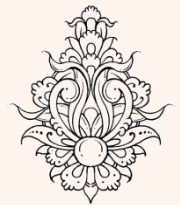
(2) \_\_\_\_\_ROO\_\_\_\_\_NOT-V\_\_\_\_\_I. . . . . SM\_\_\_\_\_OBL\_\_\_\_\_

(1) READING LAMP ON THE DESK SHED GLOW ON

(2) REA . . . . . O\_\_\_\_\_D\_\_\_\_\_SHED-GLO--O--

(1) POLISHED WOOD BUT LESS ON THE SHABBY RED CARPET

(2) P-L-S\_\_\_\_\_O---BU--L-S--O . . . . . SH . . . . . RE--C . . . . .



- در ۶۹٪ موارد پیش‌بینی درست بوده است.



# پیش‌بینی متن - آزمایش | (ادامه...)

- چنانچه که بتوان در کدگشا (گیرنده) مدس مشابهی زد، می‌توان برای فشرده‌سازی از چنین پیش‌بینی استفاده کرد:

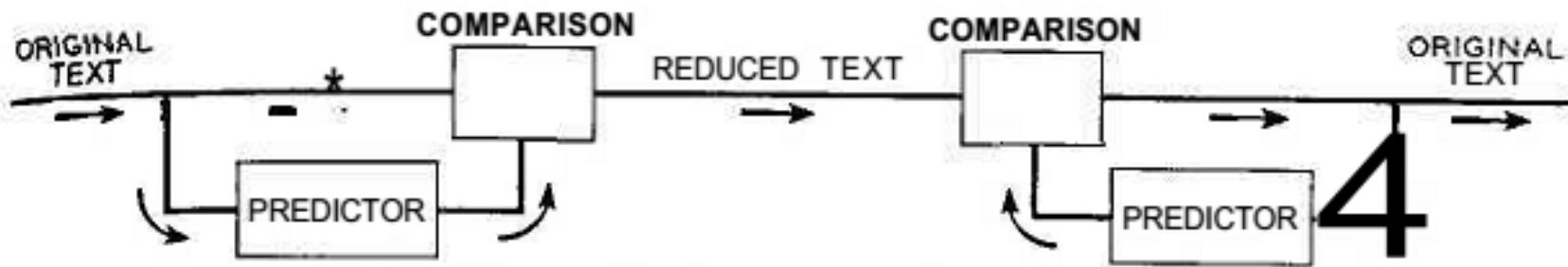
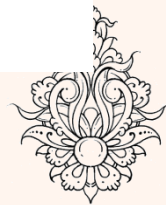


Fig. 2—Communication system using reduced text.



# پیش‌بینی متن - آزمایش ۲

- در آزمایش دوم در صورت حدس اشتباه کاربر حدس دیگری را جایگزین می‌کند و این کار آنقدر ادامه می‌یابد تا نماد مورد نظر درست حدس زده شود.

(1) T H E R E I S N O R E V E R S E O N A M O T O R C Y C L E A  
(2) 11151121121115 1171112132122711114111 1131

(1) F R I E N D O F M I N E F O U N D T H I S O U T  
(2) 86131111111111621111112111111

(1) R A T H E R D R A M A T I C A L L Y T H E O T H E R D A Y  
(2) 4111111111511111111111116111111111111111

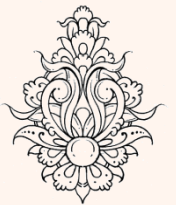
(8)

بر اساس این آزمایش برای نمایش الفبای انگلیسی بین ۰،۶ تا ۱،۳ بیت به ازای هر حرف نیاز دارد.



# پیش‌بینی

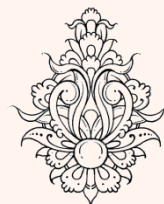
- یک راه نگهداری همگی  $N$ -تایی‌ها ( $N$ -grams) و حدس بر اساس احتمال وقوع آن‌هاست.
- در این صورت برای  $M$  نماد، محتوای مرتبه‌ی اول یک جدول  $M$ -تایی، محتوای مرتبه‌ی دوم یک جدول  $M^2$ -تایی و ... خواهد بود.
- محتوای مرتبه‌ی ۵ برای یک الفبای ۲۵۶ تایی برابر با  $۲۵۶^۵$  خواهد بود.



# پیش‌بینی با تطابق محدود

## Prediction with Partial Match (ppm)

- به جای آماده‌سازی و ذخیره‌ی جداول شامل احتمال‌های شرطی، می‌توان این جداول را در طی عملیات کدگذاری آماده نمود.
  - بدین ترتیب فقط دنباله‌هایی که در متن وجود دارد، ذخیره می‌شوند.
  - هر چند این کار نیاز به حافظه را کاهش می‌دهد، اما لازم است نمادهایی که تا کنون دیده نشده‌اند، به گونه‌ای مشخص شوند.
- استفاده از یک **نماد رهایی** یا **توالی-گریز (Escape)** لازم است.





# الگوریتم پایه

- در این الگوریتم ابتدا سعی می‌شود پیش‌بینی بر پایه حداکثر طول مورد استفاده انجام شود.  
- حداکثر طول محتوای مورد استفاده از پیش تعیین می‌شود.
- در صورت نبود نماد، در کدگذاری از یک نماد توالی‌گزیز (رهایی) استفاده شده و طول محتوای مورد بررسی را یک واحد کاهش داده و همین روند را ادامه می‌دهیم.
- در صورتی که نماد پیش از این دیده نشده بود با احتمال  $1/M$  کدگذاری انجام می‌شود.



شایگان

محتوای مرتبه ۱-

بهشتی

Context of order -1

پس از کد کردن نماد، جداول مربوطه هم به روزرسانی خواهند شد.

- هدف کد کردن رشته‌ی مقابل است:
- با فرض آن که حرف «a» بناست کد شود.
- ابتدا بررسی می‌شود که آیا دنباله‌ی «prob» یا زیر مجموعه‌ی انتهایی آن پیش از این دیده شده است یا نه؟

در صورت مشاهده «prob»:

- چنانچه «proba» مشاهده شده باشد، بر اساس احتمال موجود در این جدول این نماد کد می‌شود و این رویه برای حرف بعدی تکرار می‌شود می‌شود.
- در صورت عدم مشاهده «proba»، کاراکتر توالی‌گریز کد می‌شود.

در صورت مشاهده «rob»:

- چنانچه «roba» مشاهده شده باشد، بر اساس احتمال موجود در این جدول این نماد کد می‌شود و این رویه برای حرف بعدی تکرار می‌شود می‌شود.
- در صورت عدم مشاهده «roba»، کاراکتر توالی‌گریز کد می‌شود.

در صورت مشاهده «ob»:

- چنانچه «oba» مشاهده شده باشد، بر اساس احتمال موجود در این جدول این نماد کد می‌شود و این رویه برای حرف بعدی تکرار می‌شود می‌شود.
- در صورت عدم مشاهده «oba»، کاراکتر توالی‌گریز کد می‌شود.

در صورت مشاهده «b»:

- چنانچه «ba» مشاهده شده باشد، بر اساس احتمال موجود در این جدول این نماد کد می‌شود و این رویه برای حرف بعدی تکرار می‌شود می‌شود.
- در صورت عدم مشاهده «ba»، کاراکتر توالی‌گریز کد می‌شود.

در صورتی که «proba» تا به حال دیده نشده باشد، اما «a» پیش از این مشاهده شده باشد، به منزله محتوای مرتبه‌ی صفر کد خواهد شد: بر اساس تعداد کاراکتر مشاهده شده و این رویه برای حرف بعدی تکرار می‌شود می‌شود.

چنانچه کاراکتر «a» تاکنون مشاهده نشده باشد، با احتمال  $1/M$  کدگذاری انجام می‌شود.

محتوای مرتبه ۴

محتوای مرتبه ۳

محتوای مرتبه ۲

محتوای مرتبه ۱

محتوای مرتبه ۰

محتوای مرتبه -۱



this is the tight



این کاراکتر باید کد شود

مثال

مداکتر طول محتوا: ۲



# this is the tight



این کاراکتر باید کد شود

## مثال

مداکتر طول محتوا: ۲

zero-order context	
letter	count
t	1
h	1
i	2
s	2
ث	1
<Esc>	1

first-order context		
context	letter	count
t	h	1
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
ث	i	1
	<Esc>	1
s	ث	1
	<Esc>	1

second-order context		
context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	ث	1
	<Esc>	1
sث	i	1
	<Esc>	1
ثi	s	1
	<Esc>	1



# this is the tight



این کاراکتر باید کد شود

zero-order context	
letter	count
t	1
h	1
i	2
s	2
ث	1
<Esc>	1

first-order context		
context	letter	count
t	h	1
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
ث	i	1
	<Esc>	1
s	ث	1
	<Esc>	1

second-order context		
context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	ث	1
	<Esc>	1
sث	i	1
	<Esc>	1
ثi	s	1
	<Esc>	1

## مثال

مداکتر طول محتوا: ۲

برای کدگذاری تنها کافیست این جدول مورد استفاده قرار گیرد.



# this is the tight



این کاراکتر باید کد شود

## مثال

مداکتر طول محتوا: ۲

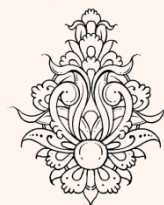
برای کدگذاری تنها کافیست این جدول مورد استفاده قرار گیرد.



zero-order context	
letter	count
t	1
h	1
i	2
s	2
ث	1
<Esc>	1

first-order context		
context	letter	count
t	h	1
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
ث	i	1
	<Esc>	1
s	ث	1
	<Esc>	1

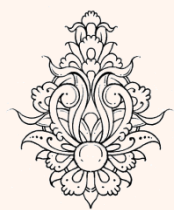
second-order context		
context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	ث	1
	<Esc>	1
sث	i	1
	<Esc>	1
ثi	s	1
	<Esc>	1



کدگذاری با روش Huffman و یا روش محاسباتی قابل انجام است. شمارش تعداد وقوع هر حرف در جداول به منظور محاسبه احتمال و یا تابع توزیع تجمعی است.

this is the tight

گام بعدی به روزسانی همه جدول‌هاست



this is the tight

گام بعدی به روزسانی همه جدول‌هاست

zero-order context	
letter	count
t	1
h	1
i	2
s	2
Ⓢ	2
<Esc>	1

first-order context		
context	letter	count
t	h	1
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
Ⓢ	i	1
	<Esc>	1
s	Ⓢ	2
	<Esc>	1

second-order context		
context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	Ⓢ	2
	<Esc>	1
sⓈ	i	1
	<Esc>	1
Ⓢi	s	1
	<Esc>	1





this is the tight

گام بعدی به روزسانی همه جدول‌هاست

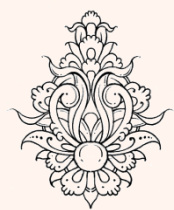
نویسه‌هایی



zero-order context	
letter	count
t	1
h	1
i	2
s	2
Ⓢ	2
<Esc>	1

first-order context		
context	letter	count
t	h	1
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
Ⓢ	i	1
	<Esc>	1
s	Ⓢ	2
	<Esc>	1

second-order context		
context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	Ⓢ	2
	<Esc>	1
sⓈ	i	1
	<Esc>	1
Ⓢi	s	1
	<Esc>	1



this is the tight

نویسه‌هایی



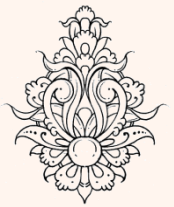
zero-order context	
letter	count
t	1
h	1
i	2
s	2
Ⓢ	2
<Esc>	1

first-order context		
context	letter	count
t	h	1
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
Ⓢ	i	1
	<Esc>	1
s	Ⓢ	2
	<Esc>	1

گام بعدی به روزسانی همه جدول‌هاست

نویسه‌هایی

second-order context		
context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	Ⓢ	2
	<Esc>	1
sⓈ	i	1
	<Esc>	1
Ⓢi	s	1
	<Esc>	1



# this is the tight

برای کدگذاری این جدول مورد استفاده قرار می‌گیرد.

نویسه‌هایی

گام بعدی به روزسانی همه جدول‌هاست

نویسه‌هایی

## zero-order context

letter	count
t	1
h	1
i	2
s	2
Ⓢ	2
<Esc>	1

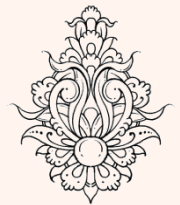
## first-order context

context	letter	count
t	h	1
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
Ⓢ	i	1
	<Esc>	1
s	Ⓢ	2
	<Esc>	1

## second-order context

context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	Ⓢ	2
	<Esc>	1
sⓈ	i	1
	<Esc>	1
Ⓢi	s	1
	<Esc>	1

گام بعدی به روزسانی همه جدول‌هاست



this is the tight



**zero-order context**

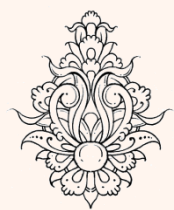
letter	count
t	2
h	1
i	2
s	2
ħ	2
<Esc>	1

**first-order context**

context	letter	count
t	h	1
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
ħ	i	1
	t	1
	<Esc>	1
s	ħ	2
	<Esc>	1

**second-order context**

context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	ħ	2
	<Esc>	1
sħ	i	1
	t	1
	<Esc>	1
ħi	s	1
	<Esc>	1



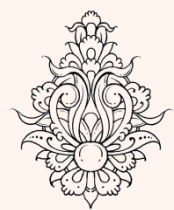
this is the tight



zero-order context	
letter	count
t	2
h	1
i	2
s	2
ħ	2
<Esc>	1

first-order context		
context	letter	count
t	h	1
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
ħ	i	1
	t	1
ħ	<Esc>	1
	ħ	2
s	<Esc>	1

second-order context		
context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	ħ	2
	<Esc>	1
sħ	i	1
	t	1
sħ	<Esc>	1
	ħ	1
ħi	s	1
	<Esc>	1



# this is the tight

برای کدگذاری این جدول مورد استفاده قرار می‌گیرد.

## zero-order context

letter	count
t	2
h	1
i	2
s	2
þ	2
<Esc>	1

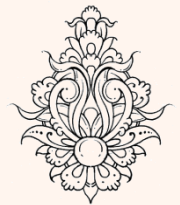
## first-order context

context	letter	count
t	h	1
t	<Esc>	1
h	i	1
h	<Esc>	1
i	s	2
i	<Esc>	1
þ	i	1
þ	t	1
þ	<Esc>	1
s	þ	2
s	<Esc>	1

## second-order context

context	letter	count
th	i	1
th	<Esc>	1
hi	s	1
hi	<Esc>	1
is	þ	2
is	<Esc>	1
sþ	i	1
sþ	t	1
sþ	<Esc>	1
þi	s	1
þi	<Esc>	1

گام بعدی به روزرسانی همه جدول‌هاست



this is the tight



**zero-order context**

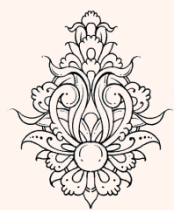
letter	count
t	2
h	2
i	2
s	2
ħ	2
<Esc>	1

**first-order context**

context	letter	count
t	h	2
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
ħ	i	1
	t	1
	<Esc>	1
s	ħ	2
	<Esc>	1

**second-order context**

context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	ħ	2
	<Esc>	1
sħ	i	1
	t	1
	<Esc>	1
ħi	s	1
	<Esc>	1
bt	h	1
	<Esc>	1



this is the tight

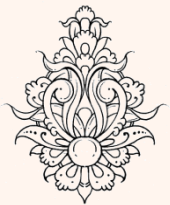
نویسه‌هایی



zero-order context	
letter	count
t	2
h	2
i	2
s	2
ħ	2
<Esc>	1

first-order context		
context	letter	count
t	h	2
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
ħ	i	1
	t	1
ħ	<Esc>	1
	ħ	2
s	ħ	2
	<Esc>	1

second-order context		
context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	ħ	2
	<Esc>	1
sħ	i	1
	t	1
sħ	<Esc>	1
	ħ	1
ħi	s	1
	<Esc>	1
ħt	h	1
	<Esc>	1





this is the tight

نویسه‌های

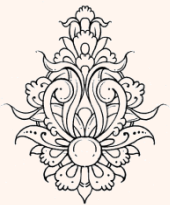
نویسه‌های



zero-order context	
letter	count
t	2
h	2
i	2
s	2
ħ	2
<Esc>	1

first-order context		
context	letter	count
t	h	2
t	<Esc>	1
h	i	1
h	<Esc>	1
i	s	2
i	<Esc>	1
ħ	i	1
ħ	t	1
ħ	<Esc>	1
s	ħ	2
s	<Esc>	1

second-order context		
context	letter	count
th	i	1
th	<Esc>	1
hi	s	1
hi	<Esc>	1
is	ħ	2
is	<Esc>	1
sħ	i	1
sħ	t	1
sħ	<Esc>	1
ħi	s	1
ħi	<Esc>	1
ħt	h	1
ħt	<Esc>	1



this is the tight

نویسه‌های

نویسه‌های

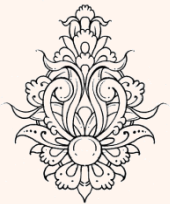
نویسه‌های



zero-order context	
letter	count
t	2
h	2
i	2
s	2
ħ	2
<Esc>	1

first-order context		
context	letter	count
t	h	2
t	<Esc>	1
h	i	1
h	<Esc>	1
i	s	2
i	<Esc>	1
ħ	i	1
ħ	t	1
ħ	<Esc>	1
s	ħ	2
s	<Esc>	1

second-order context		
context	letter	count
th	i	1
th	<Esc>	1
hi	s	1
hi	<Esc>	1
is	ħ	2
is	<Esc>	1
sħ	i	1
sħ	t	1
sħ	<Esc>	1
ħi	s	1
ħi	<Esc>	1
ħt	h	1
ħt	<Esc>	1



this is the tight

نویسه‌های

نویسه‌های

نویسه‌های

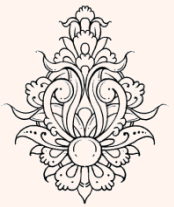
zero-order context	
letter	count
t	2
h	2
i	2
s	2
ħ	2
<Esc>	1

first-order context		
context	letter	count
t	h	2
	<Esc>	1
h	i	1
	<Esc>	1
i	s	2
	<Esc>	1
ħ	i	1
	t	1
	<Esc>	1
s	ħ	2
	<Esc>	1

second-order context		
context	letter	count
th	i	1
	<Esc>	1
hi	s	1
	<Esc>	1
is	ħ	2
	<Esc>	1
sħ	i	1
	t	1
	<Esc>	1
ħi	s	1
	<Esc>	1
ħt	h	1
	<Esc>	1

محتوای مرتبه ۱ -

گام بعدی به روزرسانی همه جدول‌هاست



# The Exclusion Principle

هر چه جدول‌ها کوچک‌تر باشند فشرده‌سازی بهتر انجام می‌شود. به عنوان مثال در کدهای محاسباتی باعث می‌شود طول بازه به تعداد کم‌تر شکسته شود و در نتیجه طول زیربازه‌ها افزایش می‌یابد و در این حالت تعداد بیت بیشتری قابل نمایش خواهد بود.

*probability*

first-order context		
context	letter	count
b	l	5
	o	3
	a	4
	r	4
	e	2
	<Esc>	1

second-order context		
context	letter	count
ob	l	10
	o	3
	<Esc>	1

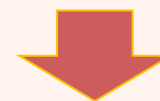


# The Exclusion Principle

هر چه جدول‌ها کوچک‌تر باشند فشردگی بهتر انجام می‌شود. به عنوان مثال در کدهای محاسباتی باعث می‌شود طول بازه به تعداد کم‌تر شکسته شود و در نتیجه طول زیربازه‌ها افزایش می‌یابد و در این حالت تعداد بیت بیشتری قابل نمایش خواهد بود.

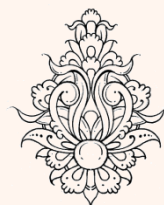
*probability*

نویسه‌هایی



first-order context		
context	letter	count
b	l	5
	o	3
	a	4
	r	4
	e	2
	<Esc>	1

second-order context		
context	letter	count
ob	l	10
	o	3
	<Esc>	1



# The Exclusion Principle

هر چه جدول‌ها کوچک‌تر باشند فشردگی بهتر انجام می‌شود. به عنوان مثال در کدهای محاسباتی باعث می‌شود طول بازه به تعداد کم‌تر شکسته شود و در نتیجه طول زیربازه‌ها افزایش می‌یابد و در این حالت تعداد بیت بیشتری قابل نمایش خواهد بود.

*probability*

نویسه‌هایی

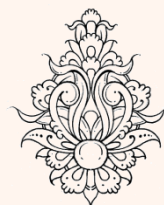


## first-order context

context	letter	count
b	l	5
	o	3
	a	4
	r	4
	e	2
	<Esc>	1

## second-order context

context	letter	count
ob	l	10
	o	3
	<Esc>	1



# The Exclusion Principle

هر چه جدول‌ها کوچک‌تر باشند فشردگی بهتر انجام می‌شود. به عنوان مثال در کدهای محاسباتی باعث می‌شود طول بازه به تعداد کم‌تر شکسته شود و در نتیجه طول زیربازه‌ها افزایش می‌یابد و در این حالت تعداد بیت بیشتری قابل نمایش خواهد بود.

*probability*

نویسه‌هایی

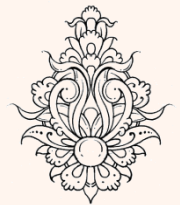


## first-order context

context	letter	count
	l	5
	o	3
b	a	4
	r	4
	e	2
	<Esc>	1

## second-order context

context	letter	count
	l	10
ob	o	3
	<Esc>	1




# The Exclusion Principle

هر چه جدول‌ها کوچک‌تر باشند فشردگی بهتر انجام می‌شود. به عنوان مثال در کدهای محاسباتی باعث می‌شود طول بازه به تعداد کم‌تر شکسته شود و در نتیجه طول زیربازه‌ها افزایش می‌یابد و در این حالت تعداد بیت بیشتری قابل نمایش خواهد بود.

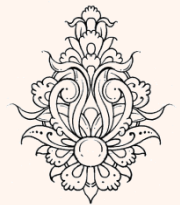
*probability*

نویسه رهایی



first-order context		
context	letter	count
b	l	X
	o	X
	a	4
	r	4
	e	2
	<Esc>	1

second-order context		
context	letter	count
ob	l	10
	o	3
	<Esc>	1





# پیش‌بینی با تطابق محدود (جمع‌بندی)

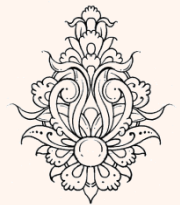
- برای کدگذاری نویسه‌ی رهایی روش‌های متعددی مطرح شده است. در این جا به ppma اشاره شد.
- افزایش طول محتوی باعث افزایش احتمال می‌شود اما باید توجه داشت که می‌تواند منجر به تعداد زیادی نماد رهایی شود.
- نسبت به LZ **کندتر** است.
- به حافظه‌ی زیادی احتیاج دارد.
- نمونه‌های پیاده‌سازی جدید آن در زمره‌ی بهترین روش‌ها برای فشرده‌سازی متن‌های شامل زبان طبیعی است.



# فشرده‌سازی با BWT

## The Burrows-Wheeler Transform

- از محتوا استفاده می‌کند.
- سال ۱۹۹۴ مطرح شد.
- تبدیل مورد استفاده در سال ۱۹۸۳ توسط wheeler مطرح شده بود.
- برای **متن** مناسب است.
- لازم است کل داده‌ها پیش از کدگذاری در اختیار باشد.
- در **bzip2** استفاده شده است.



*Burrows, M. and D. J. Wheeler (1994). A block-sorting lossless data compression algorithm, Digital Systems Research Center.*

# The Burrows-Wheeler Transform

- یک دنباله شامل  $N-1$  نماد را در نظر بگیرید.
- این دنباله را هر بار به صورت دورانی یک واحد (به سمت چپ) راست شیفت داده و  $N-1$  دنباله‌ی حاصل را به صورت «الفبایی» مرتب کنید.
- خروجی تبدیل، نمادهای آخر دنباله‌های مرتب شده و موقعیت دنباله‌ی اصلی است.

Lexicographical order

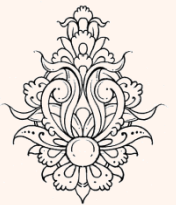


# مثال - تبدیل BW

*this is the*

دنباله‌های حاصل از چرخش دنباله‌ی ورودی:

00	<i>t</i>	<i>h</i>	<i>i</i>	<i>s</i>	⌀	<i>i</i>	<i>s</i>	⌀	<i>t</i>	<i>h</i>	<i>e</i>
01	<i>h</i>	<i>i</i>	<i>s</i>	⌀	<i>i</i>	<i>s</i>	⌀	<i>t</i>	<i>h</i>	<i>e</i>	<i>t</i>
02	<i>i</i>	<i>s</i>	⌀	<i>i</i>	<i>s</i>	⌀	<i>t</i>	<i>h</i>	<i>e</i>	<i>t</i>	<i>h</i>
03	<i>s</i>	⌀	<i>i</i>	<i>s</i>	⌀	<i>t</i>	<i>h</i>	<i>e</i>	<i>t</i>	<i>h</i>	<i>i</i>
04	⌀	<i>i</i>	<i>s</i>	⌀	<i>t</i>	<i>h</i>	<i>e</i>	<i>t</i>	<i>h</i>	<i>i</i>	<i>s</i>
05	<i>i</i>	<i>s</i>	⌀	<i>t</i>	<i>h</i>	<i>e</i>	<i>t</i>	<i>h</i>	<i>i</i>	<i>s</i>	⌀
06	<i>s</i>	⌀	<i>t</i>	<i>h</i>	<i>e</i>	<i>t</i>	<i>h</i>	<i>i</i>	<i>s</i>	⌀	<i>i</i>
07	⌀	<i>t</i>	<i>h</i>	<i>e</i>	<i>t</i>	<i>h</i>	<i>i</i>	<i>s</i>	⌀	<i>i</i>	<i>s</i>
08	<i>t</i>	<i>h</i>	<i>e</i>	<i>t</i>	<i>h</i>	<i>i</i>	<i>s</i>	⌀	<i>i</i>	<i>s</i>	⌀
09	<i>h</i>	<i>e</i>	<i>t</i>	<i>h</i>	<i>i</i>	<i>s</i>	⌀	<i>i</i>	<i>s</i>	⌀	<i>t</i>
10	<i>e</i>	<i>t</i>	<i>h</i>	<i>i</i>	<i>s</i>	⌀	<i>i</i>	<i>s</i>	⌀	<i>t</i>	<i>h</i>



# ادامه‌ی مثال

داده‌ها به صورت الفبایی مرتب می‌شوند.

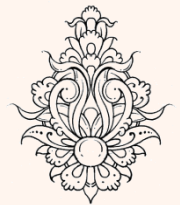
**F**

0	ϕ	i	s	ϕ	t	h	e	t	h	i	s
1	ϕ	t	h	e	t	h	i	s	ϕ	i	s
2	e	t	h	i	s	ϕ	i	s	ϕ	t	h
3	h	e	t	h	i	s	ϕ	i	s	ϕ	t
4	h	i	s	ϕ	i	s	ϕ	t	h	e	t
5	i	s	ϕ	i	s	ϕ	t	h	e	t	h
6	i	s	ϕ	t	h	e	t	h	i	s	ϕ
7	s	ϕ	i	s	ϕ	t	h	e	t	h	i
8	s	ϕ	t	h	e	t	h	i	s	ϕ	i
9	t	h	e	t	h	i	s	ϕ	i	s	ϕ
10	t	h	i	s	ϕ	i	s	ϕ	t	h	e

**L**

**L: sshthtth ii e 10** خروجی:

• این تبدیل «برگشت‌پذیر» است؛ خروجی برای یافتن ورودی کفایت می‌کند.



# مثال - تبدیل معکوس

چگونه می‌توان از خروجی به ورودی رسید؟

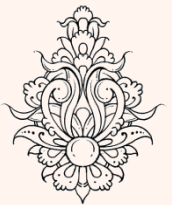
- تمام نمادها در رشته‌ی خروجی وجود دارد.

*L: sshthbiibe*

- می‌توان با توجه به رشته‌ی *L*، رشته‌ی *F* را به دست آورد، با مرتب کردن الفبایی رشته.

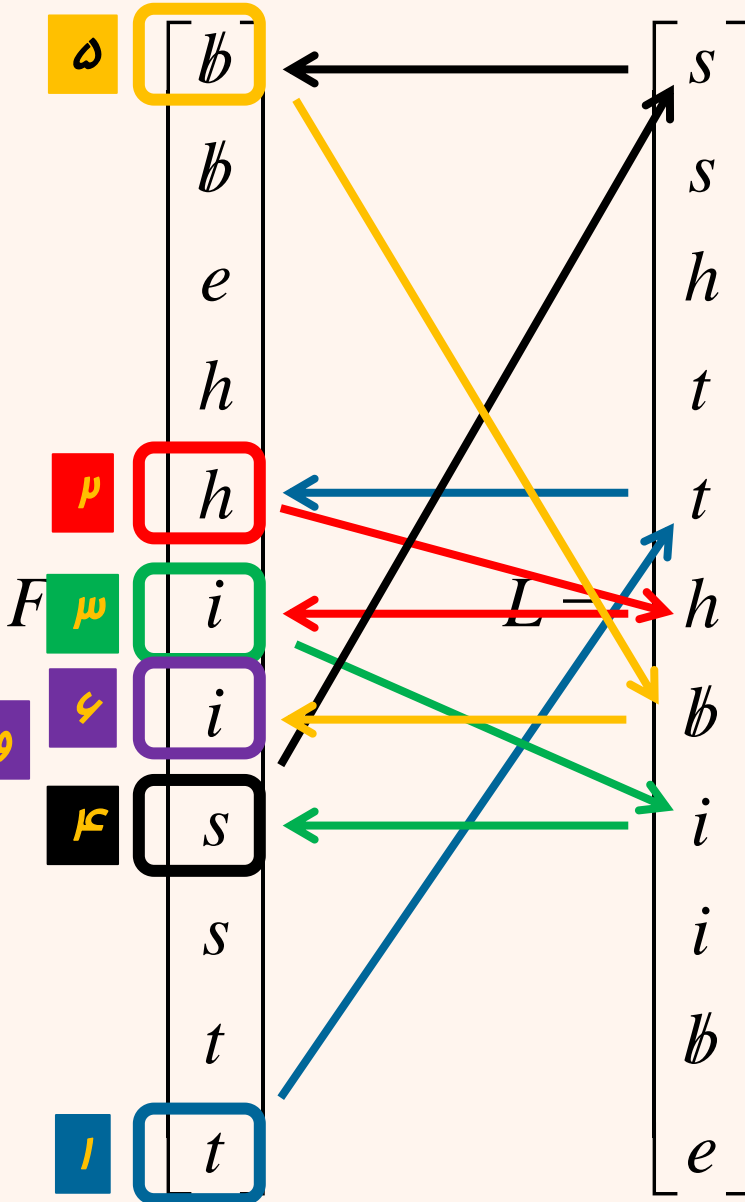
*F: bbhehisstt*

0	∅	i	s	∅	t	h	e	t	h	i	s
1	∅	t	h	e	t	h	i	s	∅	i	s
2	e	t	h	i	s	∅	i	s	∅	t	h
3	h	e	t	h	i	s	∅	i	s	∅	t
4	h	i	s	∅	i	s	∅	t	h	e	t
5	i	s	∅	i	s	∅	t	h	e	t	h
6	i	s	∅	t	h	e	t	h	i	s	∅
7	s	∅	i	s	∅	t	h	e	t	h	i
8	s	∅	t	h	e	t	h	i	s	∅	i
9	t	h	e	t	h	i	s	∅	i	s	∅
10	t	h	i	s	∅	i	s	∅	t	h	e

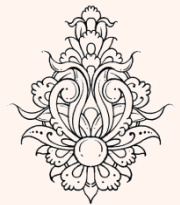


# ادامه‌ی مثال

0	⌀	i	s	⌀	t	h	e	t	h	i	s
1	⌀	t	h	e	t	h	i	s	⌀	i	s
2	e	t	h	i	s	⌀	i	s	⌀	t	h
3	h	e	t	h	i	s	⌀	i	s	⌀	t
4	h	i	s	⌀	i	s	⌀	t	h	e	t
5	i	s	⌀	i	s	⌀	t	h	e	t	h
6	i	s	⌀	t	h	e	t	h	i	s	⌀
7	s	⌀	i	s	⌀	t	h	e	t	h	i
8	s	⌀	t	h	e	t	h	i	s	⌀	i
9	t	h	e	t	h	i	s	⌀	i	s	⌀
10	t	h	i	s	⌀	i	s	⌀	t	h	e



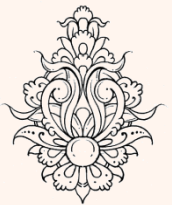
*t h i s b i ...*



# تبدیل BWT و فشرده‌سازی

- تاکنون عملاً هیچ‌گونه فشرده‌سازی انجام نشده است.
- در حقیقت با این تبدیل، دنباله‌ای به دست می‌آید که به نسبت بیشتری قابل فشرده‌سازی خواهد بود.
- در زبان انگلیسی واژه‌ی «the» زیاد استفاده می‌شود، با این روش بعد از عملیات چرخش و مرتب‌سازی، چندین رشته خواهیم داشت که با «he» شروع می‌شود و به «t» ختم می‌شود.
- در L تعداد زیادی «t» در کنار هم قرار خواهد گرفت.
- در این حالت رشته‌ای خواهیم داشت که «به صورت محلی همگن» است.
- برای پاسخ مناسب به چنده کیلوبایت داده نیاز دارد.

**locally homogeneous**

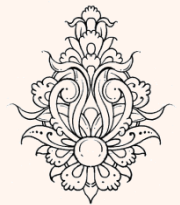




# Move-to-Front Coding(MTF)

book stack

- می‌توان آن را پیش‌پردازش الگوریتم‌های کدگذاری دانست.
- در این شیوه کاراکترهای متوالی شبیه به هم با کاراکتر «0» کد می‌شوند.
- عملکرد به این ترتیب است که الفبا به صورت یک لیست در نظر گرفته می‌شود.
- عنصر بالا با شاخص صفر نشان داده می‌شود و سایر عناصر به ترتیب با اعداد طبیعی
- آخرین عنصری که دیده شده است به بالای لیست منتقل می‌شود.
- بدین ترتیب قسمت‌های همگن با یک‌سری «0» پشت سر هم مدل می‌شود.



# مثال ۱

$$A = \{a, b, c, d\}$$

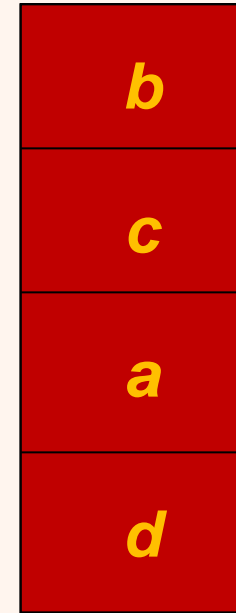
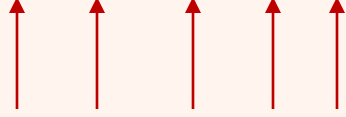
الفبا

پیشتهی  
نمادهای  
دیده شده

رشتهی ورودی

$S =$ 

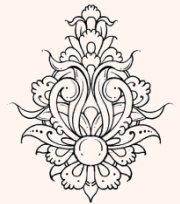
$b$	$a$	$a$	$c$	$b$
-----	-----	-----	-----	-----



$MTF(S) =$ 

$1$	$1$	$0$	$2$	$2$
-----	-----	-----	-----	-----

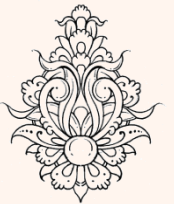
کد خروجی



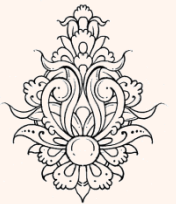
**Move to Front**

مثال ۲

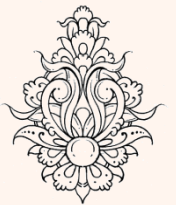
<u>a</u> bracadabra		<u>a</u> ,b,r,c,d
---------------------	--	-------------------



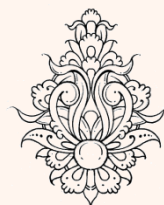
<u>a</u> bracadabra		<u>a</u> ,b,r,c,d
ab <u>r</u> acadabra	0	a, <u>b</u> ,r,c,d



<u>a</u> bracadabra		<u>a</u> ,b,r,c,d
ab <u>b</u> racadabra	0	a, <u>b</u> ,r,c,d
abr <u>a</u> cadabra	0,1	b,a, <u>r</u> ,c,d



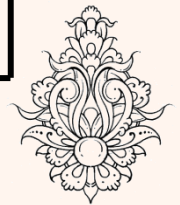
<u>a</u> bracadabra		<u>a</u> ,b,r,c,d
ab <u>b</u> racadabra	0	a, <u>b</u> ,r,c,d
abr <u>a</u> cadabra	0,1	b,a, <u>r</u> ,c,d
abra <u>a</u> cadabra	0,1,2	r,b, <u>a</u> ,c,d



<u>a</u> bracadabra		<u>a</u> ,b,r,c,d
ab <u>b</u> racadabra	0	a, <u>b</u> ,r,c,d
abr <u>a</u> cadabra	0,1	b,a, <u>r</u> ,c,d
abra <u>c</u> adabra	0,1,2	r,b, <u>a</u> ,c,d
abra <u>c</u> adabra	0,1,2,2	a,r,b, <u>c</u> ,d

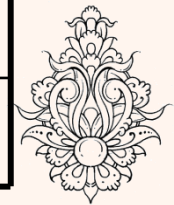


<u>a</u> bracadabra		<u>a</u> ,b,r,c,d
a <u>b</u> racadabra	0	a, <u>b</u> ,r,c,d
ab <u>r</u> acadabra	0,1	b,a, <u>r</u> ,c,d
abra <u>a</u> cadabra	0,1,2	r,b, <u>a</u> ,c,d
abra <u>c</u> adabra	0,1,2,2	a,r,b, <u>c</u> ,d
abrac <u>a</u> dabra	0,1,2,2,3	c, <u>a</u> ,r,b,d
abracad <u>d</u> abra	0,1,2,2,3,1	a,c,r,b, <u>d</u>





<u>a</u> bracadabra		<u>a</u> ,b,r,c,d
a <u>b</u> racadabra	0	a, <u>b</u> ,r,c,d
ab <u>r</u> acadabra	0,1	b,a, <u>r</u> ,c,d
abra <u>a</u> cadabra	0,1,2	r,b, <u>a</u> ,c,d
abra <u>c</u> adabra	0,1,2,2	a,r,b, <u>c</u> ,d
abrac <u>a</u> dabra	0,1,2,2,3	c, <u>a</u> ,r,b,d
abracad <u>d</u> abra	0,1,2,2,3,1	a,c,r,b, <u>d</u>
abracadabra	0,1,2,2,3,1,4,1,4,4,2	



بعد از این مراحل دنباله دارای تعداد زیادی صفر خواهد بود و بدین ترتیب آنتروپی کاهش می‌یابد.



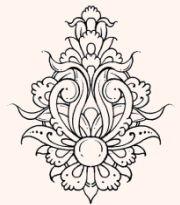
# Run Length Encoding(RLE)

- در این شیوه از تعداد تکرار نمادهای متوالی برای کدگذاری استفاده می‌شود:

**$abbbaaccca \Rightarrow (a,1),(b,3),(a,2),(c,4),(a,1)$**

– در فرمت فشرده‌سازی تصویر jpeg از این شیوه بهره گرفته شده است.

- بدین ترتیب شمای کلی سیستم فشرده‌سازی به صورت زیر خواهد بود:

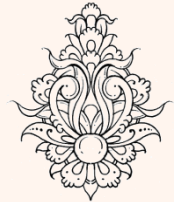


- در این شیوه، یک واژه‌نامه از محتوای رشته‌ی دیده شده تهیه می‌شود.
  - از یک پنجره‌ی لغزنده شبیه LZ77 استفاده می‌کند.
  - بخش چپ پنجره **Context** و بخش راست آن **Content** نامیده می‌شود.
- واژه‌نامه نیز دارای دو بخش مشابه است:
  - محتوای زمینه (دیده شده)
  - رشته‌ی تالی محتوای زمینه

## مقایسه از راست به چپ

- جستجو بهترین تطبیق (بر اساس ترتیب الفبایی)
- سپس content مدخل‌های همسایه نیز مورد بررسی قرار می‌گیرند (طولانی‌ترین).

## مقایسه از چپ به راست



- یک سه‌تایی به عنوان خروجی ارسال می‌شود:

$(d, count, ch)$

- «d»: فاصله‌ی بین بهترین context و بهترین content

- «count»: تعداد نمادهای منطبق

- «ch»: اولین نماد منطبق نشده در پنجره‌ی look-ahead

- واژه‌نامه به روز می‌شود.

- context یافت شده یکی از مدخل‌های مرتب‌است و

ممکن است مدخل‌های همسایه‌ی آن مربوط به

content‌های مرتبط‌تر باشد.

- همیشه طولانی‌ترین content منجر به بهترین فشرده‌سازی

نمی‌شود، در برخی موارد انتخاب عبارات کوتاه نزدیک به

context ممکن است منجر به کدگذاری بهتری شوند.



**در مورد جزییات پیاده‌سازی ACB اطلاعات چندانی در دست نیست!**

# محتوا و رشته‌ی تالی

## پنجره‌ی لغزان مانند LZ77

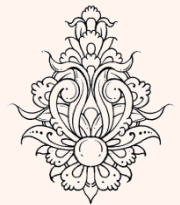
text: `swiss miss is missing`

`swiss` `miss is missing`  
context content

با هر بار شیفت یک context-content جدید به دست می‌آید:

`ss m|iss is mis`

Salomon, D. (2012). Data Compression: The Complete Reference, Springer Berlin Heidelberg.



# محتوا و رشته‌ی تالی

## پنجره‌ی لغزان مانند LZ77

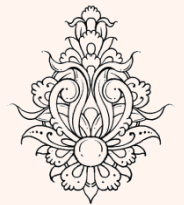
text: `swiss miss is missing`

context      content

با هر بار شیفت یک context-content جدید به دست می‌آید:

`ss m|iss is mis`

`s mi|ss is miss`



# محتوا و رشته‌ی تالی

## پنجره‌ی لغزان مانند LZ77

text: `swiss miss is missing`

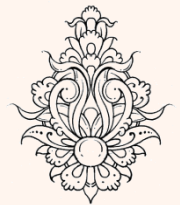
context      content

با هر بار شیفت یک context-content جدید به دست می‌آید:

`ss m|iss is mis`

`s mi|ss is miss`

`mis|s is missi`



# واژه‌نامه

بدین ترتیب واژه‌نامه به دست می‌آید (برای هفت کاراکتر).

```
...s|wiss m
...sw|iss m
...swi|ss m
...swis|s m
...swiss|m
...swiss|m
...swiss|m
```

context content

```
...swiss|m
...swi|ss m
...s|wiss m
...swis|s m
...swiss|m
...sw|iss m
```

context content



بخش context از راست به چپ مرتب شده است. بعد از پردازش هر کاراکتر یک مدخل به واژه‌نامه افزوده می‌شود. برای این که در کدگشا، واژه‌نامه به همین شیوه قابل ساخت باشد، تنها رشته‌ی موجود در search buffer در نظر گرفته می‌شود.



# مثال-کدگذاری

تا اینجا پردازش شده است

swiss m | iss is missing

current context

وضعیت جاری واژه‌نامه:

- 1 ...swiss | m
- 2 ...swi | ss m
- 3 ...s | wiss m
- 4 ...swis | s m
- 5 ...swiss | m
- 6 ...sw | iss m

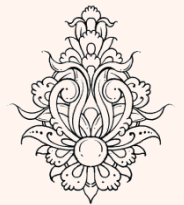
نزدیک‌ترین context در واژه‌نامه جستجو شود:

بین مدخل دوم و سوم

فاصله‌ی نزدیک‌ترین content به context یافته شده در واژه‌نامه به دست آید.

مدخل ششم، فاصله: چهار

۶-۲



# مثال-کدگذاری (ادامه...)

swiss m|iss is missing

current context

یافتن طول content یکسان

۳

چهار

یافتن اولین نماد ناهمسان با

۴

content

'i'

تولید سه تایی مرتب

۵

وضعیت جاری واژه‌نامه:

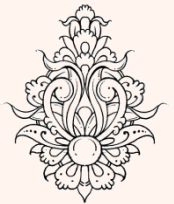
- 1 ...swiss | m
- 2 ...swi | ss m
- 3 ...s | wiss m
- 4 ...swis | s m
- 5 ...swiss | m
- 6 ...sw | iss m

(6-2, 4, 'i')

4

معادل رشته‌ی پنج‌تایی زیر

iss i

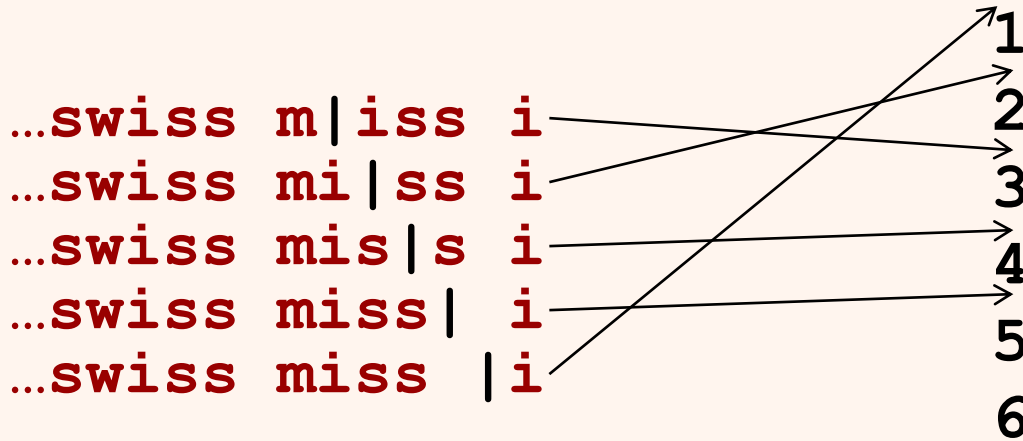


# مثال - کدگذاری (ادامه...)

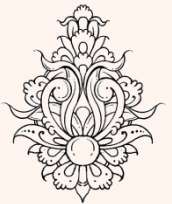
واژه‌نامه به‌روز شود:



context: **swiss m**  
content: **iss i**

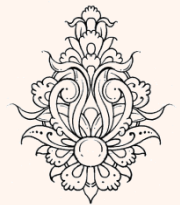


...swiss | m  
...swi | ss m  
...s | wiss m  
...swis | s m  
...swiss | m  
...sw | iss m



# واژه‌نامه به‌روزسانی‌شده

- 1 ...swiss miss | i
- 2 ...swiss | miss i
- 3 ...swiss mi | ss i
- 4 ...swi | ss miss i
- 5 ...swiss m | iss i
- 6 ...s | wiss miss i
- 7 ...swiss mis | s i
- 8 ...swis | s miss i
- 9 ...swiss miss | i
- 10 ...swiss | miss i
- 11 ...sw | iss miss i



❖ پایان یک مرحله

# واژه‌نامه به‌روزسانی‌شده

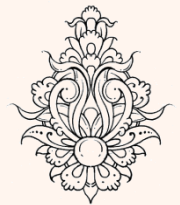
- 1 ...swiss miss | i
- 2 ...swiss | miss i
- 3 ...swiss mi | ss i
- 4 ...swi | ss miss i
- 5 ...swiss m | iss i
- 6 ...s | wiss miss i
- 7 ...swiss mis | s i
- 8 ...swis | s miss i
- 9 ...swiss miss | i 6
- 10 ...swiss | miss i
- 11 ...sw | iss miss i

تا اینجا پردازش شده است ←

swiss miss i | s missing

current context

(8-2, 6, 'i')



# مثال - کدگذاری (ادامه...)

- همین مراحل ادامه می‌یابد:

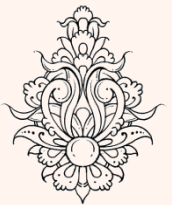
```
...s|wiss_miss_is_missi 1   ...swiss_miss_is_|missi
...sw|iss_miss_is_missi 2   ...swiss_miss_|is_missi
...swi|ss_miss_is_missi 3   ...swiss_|miss_is_missi
...swis|s_miss_is_missi 4   ...swiss_miss_|s_missi
...swiss|_miss_is_missi 5   ...swiss_miss_is_|ssi
...swiss_|miss_is_missi 6   ...swiss_mi|ss_is_missi
...swiss_m|iss_is_missi 7   ...swi|ss_miss_is_missi
...swiss_mi|ss_is_missi 8   ...swiss_miss_is_|missi
...swiss_mis|s_is_missi 9   ...swiss_m|iss_is_missi
...swiss_miss|_is_missi 10  ...s|wiss_miss_is_missi
...swiss_miss_|is_missi 11  ...swiss_miss_is_|missi
...swiss_miss_|s_missi 12  ...swiss_miss_is_|missi
...swiss_miss_is_|missi 13  ...swiss_mis|s_is_missi
...swiss_miss_is_|missi 14  ...swis|s_miss_is_missi
...swiss_miss_is_|missi 15  ..swiss_miss_is_|missi
...swiss_miss_is_|missi 16  ...swiss_miss_|is_missi
...swiss_miss_is_|missi 17  ...swiss_|miss_is_missi
..swiss_miss_is_|missi 18  ...sw|iss_miss_is_missi
```

swiss miss is missi|ng

current context

(0, 0, 'n')

و به همین ترتیب



# مثال-کدگشایی

۱

نزدیک‌ترین context در واژه‌نامه جستجو شود.

۲

بر اساس کد، content به دست آید.

۳

رشته‌ی مشابه بر اساس کد مشخص شود.

۴

کاراکتر سوم به رشته‌ی کدگشایی شده اضافه شود.

۵

واژه‌نامه به روز شود.



context: **swiss m**  
content: **unkown**  
(4,4,'i')

d=4 → “iss ”

i='i' → “ iss i ”

وضعیت کنونی واژه‌نامه:

1	...swiss   m
2	...sw <u>i</u>   ss m
3	...s   wiss m
4	...swis   s m
5	...swiss   m
6	...sw   <u>iss</u> m

4

# مثال

$S = mississippi$

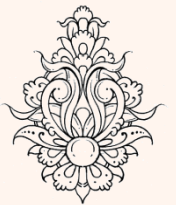
- در ابتدا واژه‌نامه خالی است.

*mississippi*



$(0, 0, m)$

	Context	Content
1	Empty	Empty





# مثال

$S = mississippi$

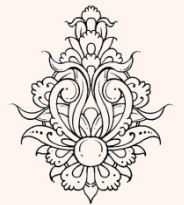
*mississippi*



$(0, 0, m)$

$(0, 0, i)$

	Context	Content
1	Empty	Empty



# مثال

$S = mississippi$

*mississippi*

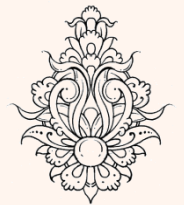


$(0, 0, m)$

$(0, 0, i)$

$(0, 0, s)$

	Context	Content
1	m	i



# مثال

$S = mississippi$

*mississippi*



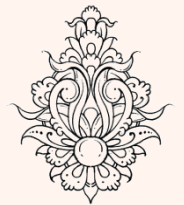
$(0, 0, m)$

$(0, 0, i)$

$(0, 0, s)$

$(-1, 1, i)$

	Context	Content
1	mi	s
2	m	is



# مثال

$S = mississippi$

*mississippi*



$(0, 0, m)$

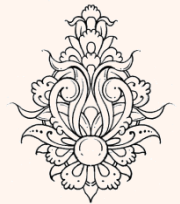
$(0, 0, i)$

$(0, 0, s)$

$(-1, 1, i)$

$(0, 3, p)$

	Context	Content
1	m	issi
2	mi	ssi
3	mis	si
4	miss	i



# مثال

$S = mississippi$

*mississippi*



$(0, 0, m)$

$(0, 0, i)$

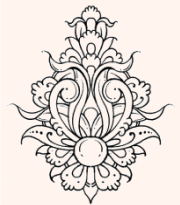
$(0, 0, s)$

$(-1, 1, i)$

$(0, 3, p)$

$(-1, 1, i)$

	Context	Content
1	mi	ssissip
2	missi	ssip
3	mississi	p
4	m	ississip
5	mis	sissip
6	missis	sipp
7	miss	issip
8	mississ	ip



# مثال

$S = mississippi$

*mississippi*



$(0, 0, m)$

$(0, 0, i)$

$(0, 0, s)$

$(-1, 1, i)$

$(0, 3, p)$

$(-1, 1, i)$

	Context	Content
1	mi	ssissip
2	missi	ssip
3	mississi	p
4	m	ississip
5	mississip	pi
6	mississipp	i
7	mis	sissip
8	missis	sipp
9	miss	issip
10	mississ	ip

