

بازیابی اطلاعات

دکتر امین گلزاری اسکویی

a.golzari@azaruniv.ac.ir

a.golzari@tabrizu.ac.ir

<https://github.com/Amin-Golzari-Oskoue>



دانشگاه صنعتی ارومیه

پاییز ۱۴۰۲

فصل ۱

بازیابی بولی (Boolean Retrieval)

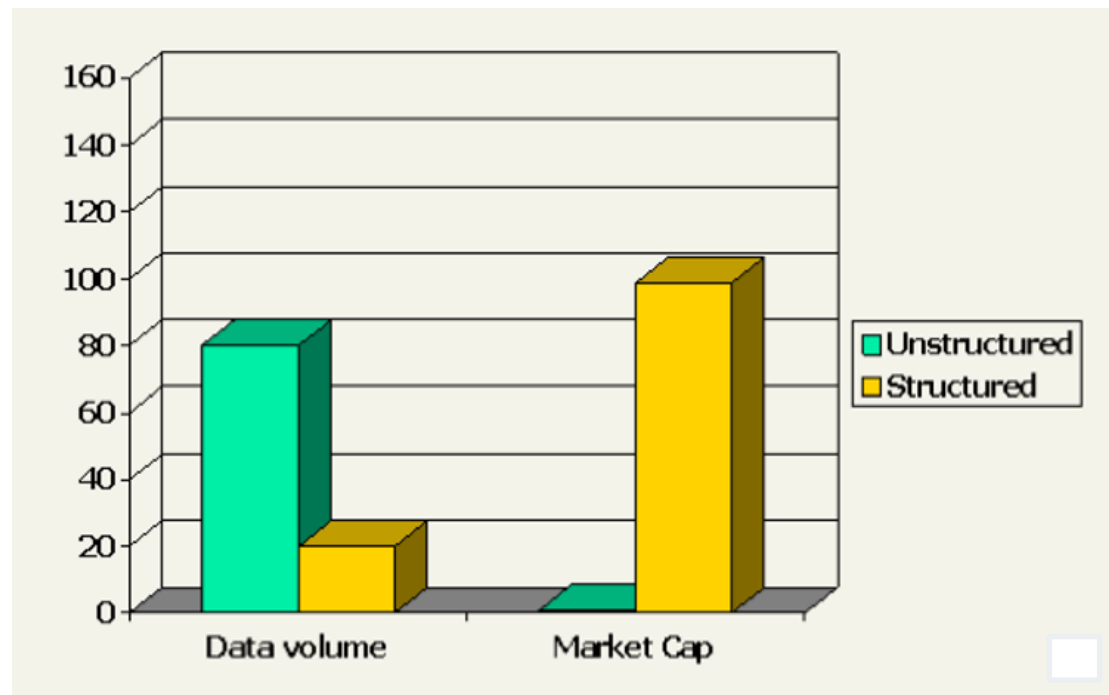
مطالب این فصل

- بازیابی بولی: طراحی و ساختارهای داده یک سیستم بازیابی اطلاعات ساده
- شاخص وارونه

تعریفی بر بازیابی اطلاعات

بازیابی اطلاعات یافتن موارد (معمولا اسناد) از یک ماهیت **بدون سافتار** (معمولا متن) است که یک نیاز اطلاعاتی را از داخل مجموعه‌های بزرگ (که معمولا در کامپیوتر ذخیره میشوند) برآورده می‌کند.

مقایسه داده‌های بدون ساختار (متن) با ساختاریافته (پایگاه داده) در سال ۱۹۹۶



مقایسه داده‌های بدون ساختار (متن) با ساختاریافته (پایگاه داده) در سال ۲۰۰۶

Unstructured (text) vs. structured (database) data in 2006



بازیابی بولی

❖ مدل بولی بدون شک ساده ترین مدلی است که یک سیستم بازیابی اطلاعات بر اساس آن پایه گذاری می شود.

❖ کوئری ها عبارت های بولی هستند ، برای مثال : CAEAR AND BRUTUS

❖ موتور جستجو همه اسنادی که مورد تایید عبارت بولی است را برمی گرداند .

آیا گوگل از مدل بولی استفاده می کند ؟

داده‌های غیر ساخت یافته در ۱۶۲۰: شکسپیر



شاخص وارونه

❖ کدام نمایشنامه‌های شکسپیر شامل کلمات **BRUTUS** و **CAESAR** هستند اما شامل **CALPURNIA** **نیستند** ؟
می‌توان تمام نمایشنامه‌های شکسپیر را که شامل دو کلمه **BRUTUS** و **CAESAR** هستند را پیدا کرد (grep)، سپس از بین آن‌ها نمایشنامه‌هایی که شامل **CALPURNIA** **هستند** را **حذف** کرد.

❖ چرا (grep) راه حل مناسبی نیست ؟

1. کند است (برای مجموعه‌های بزرگ)
2. grep ، line-oriented است اما بازایی اطلاعات document-oriented است.
3. “Not CALPURNIA” عبارت مهمی است.
4. سایر عمل‌ها (برای مثال ، پیدا کردن ROMANS نزدیک COUNTRYMAN امکان پذیر نیست)

ماتریس تلاقی Term-document

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth	...
Antony	1	1	0	0	0	1	
Brutus	1	1	0	1	0	0	
Caesar	1	1	0	1	1	1	
Calpurnia	0	1	0	0	0	0	
Cleopatra	1	0	0	0	0	0	
mercy	1	0	1	1	1	1	
worser	1	0	1	1	1	0	
...							

❖ ماتریس تلاقی عبارت-سند (t,d) یک است اگر نمایشنامه در ستون d شامل کلمه ای در سطر t باشد، و در غیر این صورت صفر خواهد بود.

بردارهای تلاقی Incidence Vectors

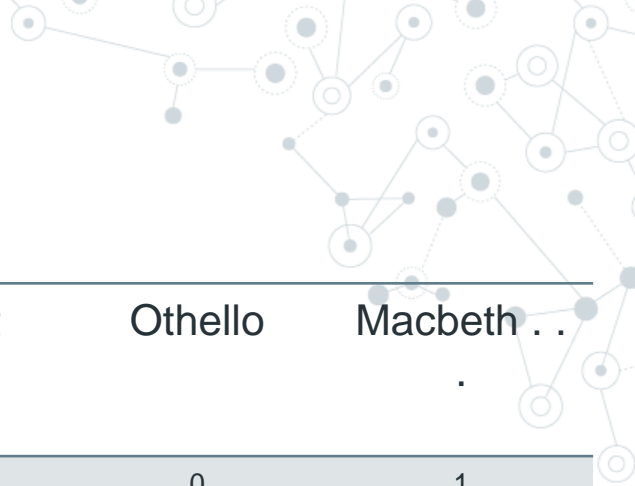
- ❖ برای هر عبارت یک بردار صفر و یکی داریم.
- ❖ برای پاسخ به کوئری (Brutus **AND** Caesar **AND NOT** Calpurnia) :

1. بردارهای Brutus ، Caesar و Calpurnia را در نظر می‌گیریم.

2. متمم بردار Calpurnia را مجاسبه می‌کنیم.

3. سپس AND بیتی انجام می‌دهیم.

$$110100 \text{ AND } 110111 \text{ AND } 101111 = 100100$$



	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth . .
ANTHONY	1	1	0	0	0	1
BRUTUS	1	1	0	1	0	0
CAESAR	1	1	0	1	1	1
CALPURNIA	0	1	0	0	0	0
CLEOPATRA	1	0	0	0	0	0
MERCY	1	0	1	1	1	1
WORSER	1	0	1	1	1	0
...						
result:	1	0	0	1	0	0

$$110100 \text{ AND } 110111 \text{ AND } 101111 = 100100$$

Antony and Cleopatra, Act III, Scene ii

Agrippa [Aside to Domitius Enobarbus]: Why, Enobarbus,
When Antony found Julius Caesar dead,
He cried almost to roaring; and he wept
When at Philippi he found Brutus slain.

Hamlet, Act III, Scene ii

Lord Polonius: I did enact Julius Caesar: I was killed i' the
Capitol; Brutus killed me.

نتایج مربوط به کتاب شکسپیر برای پرس و جوی Brutus و Caesar و نه Calpurnia

مجموعه‌های بزرگتر

- ❖ حال بیاید سناریوی واقع‌گرایانه‌تری را در نظر بگیریم. فرض کنید N سند داشته باشیم ($N=1,000,000$). منظور از اسناد هر وامدی است که ما تصمیم داریم روی آن سیستم بازیابی اطلاعات بسازیم. اسناد ممکن است یادداشتهای فردی یا فصول یک کتاب باشند. به گروهی از اسناد که بازیابی را (روی آنها انجام می‌دهیم مجموعه (اسناد) گفته می‌شود. گاهی اوقات این مجموعه را مجموعه‌ای از نوشته‌ها (بدون متن) می‌نامند.
- ❖ فرض کنید هر سندی مدود 1000 کلمه طول داشته باشد. اگر به طور متوسط 6 بایت برای هر کلمه که شامل فاصله و نقطه‌گذاری هم است در نظر بگیریم، پس مجموعه سندی به اندازه 6 گیگابایت فوایم داشت.
- ❖ نوعاً ممکن است مدود $M = 500,000$ عبارت مجزا در این اسناد وجود داشته باشد. هیچ ویژگی خاصی درباره اعدادی که انتخاب کردهایم وجود ندارد و مقادیر در اسناد مختلف متفاوت است.
- ❖ هدف ما توسعه سیستمی است که بازیابی موردی انجام دهد. این عمل، استانداردترین نوع بازیابی اطلاعات است.

❖ ماتریس عبارت-سند را نمی‌توان به سادگی تهیه کرد. یک ماتریس $500K \times 1M$ ، پانصد میلیارد صفر و یک دارد که نسبت به حافظه کامپیوتر بسیار زیاد است. اما مسئله اصلی این است که این ماتریس، به شدت پراکنده است یعنی تعداد درایه‌های غیر صفر بسیار کم است. به دلیل اینکه هر سند هزار کلمه طول دارد ، ماتریس بیشتر از یک میلیارد یک نخواهد داشت؛ بنابراین حداقل 99/8% درایه‌ها ، صفر هستند.

❖ نمایش بهتر چه چیزی می‌تواند باشد ؟
نمایش بهتر این است که تنها مواردی که موجود هستند یعنی موقعیت‌های یک را ثبت کنیم.

Brutus	→	1	2	4	11	31	45	173	174	
Caesar	→	1	2	4	5	6	16	57	132	...
Calpurnia	→	2	31	54	101					
⋮										
⏟		⏟								
لغت نامه‌ها		پست‌ها								

❖ لغت نامه در شکل بالا به صورت الفبایی مرتب شده است و لیست پست‌ها با شناسه‌ی سند مرتب شده است.

❖ برای هر عبارت، ما مجموعه‌ای از همه سندهایی که شامل عبارت مدنظر هستند، ذخیره می‌کنیم.

اولین برداشت در ساخت شاخص وارونه

❖ برای افزایش سرعت شاخص گذاری در زمان بازایی باید شاخص را از پیش بسازیم. گام‌های اصلی به ترتیب زیر می‌باشد :

1. جمع آوری اسناد که باید شاخص گذاری شوند:

Friends, Romans, countrymen. So let it be with Caesar ...

2. نشان گذاری متن ، تبدیل هر سند به لیستی از نشانه‌ها :

Friends Romans countrymen So ...

3. انجام پیش پردازش زبانی و ایجاد لیستی از نشانه‌های نرمال شده که عبارت های نمایه سازی شده اند :

friend roman countryman so ...

4. شاخص گذاری اسنادی که هر عبارت در آن رخ می‌دهد، از طریق ایجاد شاخص وارونه که شامل لغت نامه و پست باشد.

توکن کردن متن و پیش پردازش

Doc 1. I did enact Julius Caesar: I was killed i' the Capitol; Brutus killed me.

Doc 2. So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious:



Doc 1. i did enact julius caesar i was killed i' the capitol brutus killed me

Doc 2. so let it be with caesar the noble brutus hath told you caesar was ambitious

تولید پست‌ها

Doc 1. i did enact julius caesar i was killed i' the capitol brutus killed me

Doc 2. so let it be with caesar the noble brutus hath told you caesar was ambitious



term	docID
i	1
did	1
enact	1
julius	1
caesar	1
i	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

مرتب کردن پست‌ها

term	docID		term	docID
i	1		ambitious	2
did	1		be	2
enact	1		brutus	1
julius	1		brutus	2
caesar	1		capitol	1
i	1		caesar	1
was	1		caesar	2
killed	1		caesar	2
i'	1		did	1
the	1		enact	1
capitol	1		hath	1
brutus	1		i	1
killed	1		i	1
me	1	⇒	i'	1
so	2		it	2
let	2		julius	1
it	2		killed	1
be	2		killed	1
with	2		let	2
caesar	2		me	1
the	2		noble	2
noble	2		so	2
brutus	2		the	1
hath	2		the	2
told	2		told	2
you	2		you	2
caesar	2		was	1
was	2		was	2
ambitious	2		with	2

term	docID
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
i	1
i	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2



term	doc. freq.	→	postings lists
ambitious	1	→	2
be	1	→	2
brutus	2	→	1 → 2
capitol	1	→	1
caesar	2	→	1 → 2
caesar	2	→	1
did	1	→	1
enact	1	→	1
hath	1	→	2
i	1	→	1
i'	1	→	1
it	1	→	2
julius	1	→	1
killed	1	→	1
let	1	→	2
me	1	→	1
noble	1	→	2
so	1	→	2
the	2	→	1 → 2
told	1	→	2
you	1	→	2
was	2	→	1 → 2
with	1	→	2

تولید لیست از پست‌ها و تعداد تکرار اسناد

Doc 1
I did enact Julius Caesar: I was
killed i' the Capitol; Brutus killed
me.

term	docID	term	docID
I	1	ambitious	2
did	1	be	2
enact	1	brutus	1
julius	1	brutus	2
caesar	1	capitol	1
I	1	caesar	1
was	1	caesar	2
killed	1	caesar	2
i'	1	did	1
the	1	enact	1
capitol	1	hath	1
brutus	1	I	1
killed	1	I	1
me	1	i'	1
so	2	it	2
let	2	julius	1
it	2	killed	1
be	2	killed	1
with	2	let	2
caesar	2	me	1
the	2	noble	2
noble	2	so	2
brutus	2	the	1
hath	2	the	2
told	2	told	2
you	2	you	2
caesar	2	was	1
was	2	was	2
ambitious	2	with	2

Doc 2
So let it be with Caesar. The noble
Brutus hath told you Caesar was
ambitious:

term	doc. freq.	→	postings lists
ambitious	1	→	[2]
be	1	→	[2]
brutus	2	→	[1] → [2]
capitol	1	→	[1]
caesar	2	→	[1] → [2]
did	1	→	[1]
enact	1	→	[1]
hath	1	→	[2]
I	1	→	[1]
i'	1	→	[1]
it	1	→	[2]
julius	1	→	[1]
killed	1	→	[1]
let	1	→	[2]
me	1	→	[1]
noble	1	→	[2]
so	1	→	[2]
the	2	→	[1] → [2]
told	1	→	[2]
you	1	→	[2]
was	2	→	[1] → [2]
with	1	→	[2]

ساخت شاخص با استفاده از مرتب سازی و گروه بندی. دنباله عبارات در هر سند با شناسه‌ی سندشان بر چسب خورده اند (چپ) و به صورت الفبایی مرتب می‌شوند (وسط). سپس نمونه‌های همان عبارات، با کلمه و پس از آن با شناسه‌ی سند گروه بندی می‌شوند. عبارات و شناسه‌های سند، سپس از هم جدا می‌شوند (راست). لغت نامه، عبارات را ذخیره می‌کند و اشاره گری به لیست پست‌ها برای هر عبارت دارد. لغت نامه همچنین در اینجا، خلاصه اطلاعات دیگری مانند فراوانی سند از هر عبارت را ذخیره می‌کند.

پردازش پرس و جوهای بولی

❖ پرس و جو ساده (بوی عبارت)

چگونه پرس و جو را با استفاده از شاخص وارونه و مدل بازیابی پایه بولی پردازش کنیم؟
پردازش پرس و جو عطفی ساده را در نظر بگیرید :

Brutus AND Calpurnia

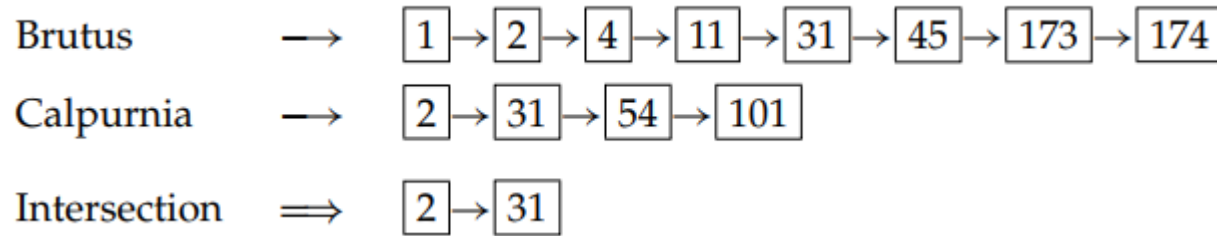
1 – محل Brutus را در لغت نامه می‌یابیم.

2- پست‌های آن را بازیابی می‌کنیم.

3 – محل Calpurnia را در لغت نامه پیدا می‌کنیم.

4- پست‌های آن را بازیابی می‌کنیم.

5- اشتراک دو لیست پست را همانطور که در شکل (اسلاید بعد) می‌بینید، تعیین می‌کنیم.



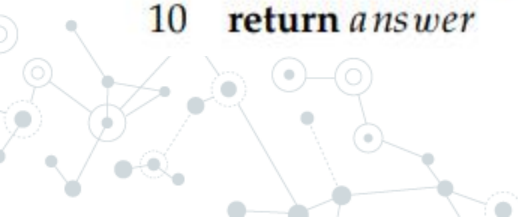
❖ توجه : این فقط در صورتی کار می‌کند که لیست پست‌ها مرتب شده باشند.

با داشتن دو لیست به طول x و y ، پیچیدگی محاسباتی از مرتبه $O(x+y)$ خواهد بود.



INTERSECT(p_1, p_2)

```
1  answer  $\leftarrow \langle \rangle$ 
2  while  $p_1 \neq \text{NIL}$  and  $p_2 \neq \text{NIL}$ 
3  do if  $\text{docID}(p_1) = \text{docID}(p_2)$ 
4      then ADD(answer,  $\text{docID}(p_1)$ )
5           $p_1 \leftarrow \text{next}(p_1)$ 
6           $p_2 \leftarrow \text{next}(p_2)$ 
7      else if  $\text{docID}(p_1) < \text{docID}(p_2)$ 
8          then  $p_1 \leftarrow \text{next}(p_1)$ 
9          else  $p_2 \leftarrow \text{next}(p_2)$ 
10 return answer
```

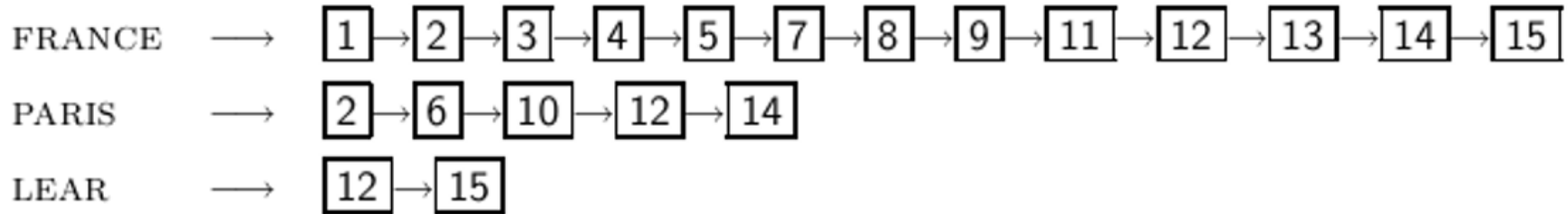


❖ الگوریتم اشتراک دو لیست پست P_1 و P_2 .

❖ تقاطع دو لیست ارسال.

تمرین پردازش پرس وجو :

❖ Compute hit list for ((Paris AND NOT France) OR Lear)



پرس و جوهای بولی

مدل بازیابی بولی، مدلی برای بازیابی اطلاعات است که در آن می‌توانیم هر پرس و جویی که به شکل عبارات بولی باشد، مطرح کنیم، یعنی عباراتی که در آن ترکیبی از عملگرهای AND ، OR ، NOT باشد.

این مدل هر سند را بصورت مجموعه ای از کلمات در نظر می‌گیرد.

. هر سند را به عنوان مجموعه ای از اصطلاحات مشاهده می‌کند.

. بطور دقیق سند با شرایط مطابقت دارد یا خیر.

. یکی از ابزارهای بازیابی تجاری اولیه در دهه سوم

❖ بسیاری از جستجوگران مرفه ای (به عنوان مثال، حقوقدانان) در حال حاضر از پرس و جوهای بولی استفاده می‌کنند.

. شما دقیقاً می‌دانید که چه چیزی به دست می‌آورید.

❖ بسیاری از سیستم‌های جستجویی که مورد استفاده قرار می‌گیرند از جست و جوهای بولی هستند که عبارتند از ایمیل، اینترنت و ...

سیستم تجاری در حوزه سیستم بازیابی اطلاعات بولی: WestLaw

❖ از بزرگترین خدمات جستجوی حقوقی تجاری بر اساس تعداد مشترکین پرداخت کننده می باشد.

❖ بیش از نیم میلیون مشترک (روزانه میلیون ها جستجو را در ده ها ترابایت داده متنی انجام می دهند.

❖ این سرویس در سال 1975 شروع به کار کرد.

❖ در سال 2005 جستجوی بولی توسط شرکت وستلو هنوز پیش فرض بود و توسط درصد زیادی از کاربران استفاده می شد.

❖ اگرچه بازیابی (رتبه بندی شده از سال 1992 در دسترس بوده است.

❖ جستجو جوی تجاری بولی (وستلو): وستلو ، بزرگترین سرویس جستجوی قانونی تجاری با بیش از نیم میلیون مشترک (در مورد تعداد مشترکینی که هزینه میپردازند است) که میلیونها جستجو را در طول روز روی دها ترابایت داده متنی اجرا میکنند.

❖ نیاز اطلاعاتی: اطلاعاتی در مورد نظریه‌های قانونی در منع فاش سازی رازهای تجاری توسط کارمندانی که قبلاً توسط شرکت رقیب استخدام شده اند.

پرس و جو : "trade secret" /s disclos! /s prevent /s employe!

❖ نیاز اطلاعاتی: ملزومات برای افراد معمول که بتوانند به محل کار برسند.
پرس و جو : (employment/3 place) work-site work-place access! /p disab!

❖ نیاز اطلاعاتی: مواردی راجع به مسئولیت‌های میزبان در مورد میهمانان از خود بی خود شده.
پرس و جو : guest /p (intoxicat! drunk!) /p (responsib! liab!) /p host!

❖ عملگرهای مجاورتی: 3/ یعنی شامل سه کلمه ، s/ یعنی شامل جمله ، p/ یعنی شامل پاراگراف .

❖ Space برای تفکیک کلمات کوئری است و جز عملگرها نیست. (قبل از گوگل، این پیش فرض جستجو بود.)

❖ پرس و جوهای طولانی و دقیق: به صورت تدریجی توسعه یافته اند، نه مانند web search.

❖ دلیل علاقه جستجو گران مرفه‌ای اغلب برای جستجو بولی: دقت، شفافیت، کنترل.

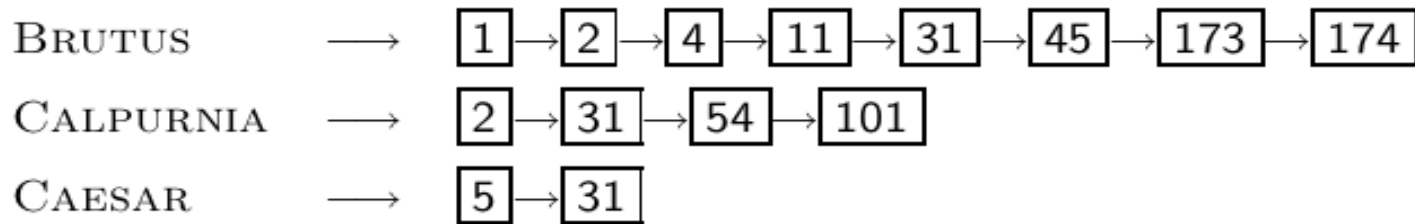
❖ چه زمانی جستجوهای بولی بهترین راه جستجو هستند؟ به نیاز اطلاعاتی، جستجوگر، مجموعه اسناد و ... بستگی دارد.

بهینه سازی پرس و جو

❖ بهینه سازی پرس و جو ، سازماندهی کار پاسفگویی به پرس و جو است به طوریکه حداقل مقدار کار توسط سیستم انجام شود. بخش مهم در پرس و جوهای بولی، ترتیبی است که در آن لیستهای پستها بررسی می‌شوند.

❖ بهترین ترتیب برای پردازش پرس و جو چیست ؟

❖ پرس و جویی را در نظر بگیرید که AND از t عبارت است، برای مثال: Brutus AND Caesar AND Calpurnia. برای هر t عبارت، باید پستهای آن را داشته و سپس آنها را با هم AND کنیم. یک مکاشفه استاندارد، پردازش عبارت به ترتیب افزایش فراوانی سند است؛ اگر ما با اشتراک گرفتن دو لیست پستی که از همه کوچک تر هستند، شروع کنیم و تمامی نتایج میانی بزرگتر از کوچکترین لیست پستها نباشد، احتمالاً ما حداقل میزان کل کار را انجام می‌دهیم. بنابراین، برای لیستهای پستهای شکل بالا را اجرا می‌کنیم به صورت: (Calpurnia AND Brutus) AND Caesar. این اولین تنظیم برای نگه داشتن فراوانی عبارات در لغت نامه است؛ این عمل به ما اجازه می‌دهد که بر اساس داده ی درون حافظه، قبل از دستیابی به هر لیست پستها تصمیم به مرتب سازی بگیریم.



❖ در این مثال ابتدا Caesar سپس Calpurnia و در آخر Brutus ادامه می‌دهیم .

الگوریتم تقاطع بهینه شده برای پرس و جویهای پیوندی

INTERSECT($\langle t_1, \dots, t_n \rangle$)

- 1 $terms \leftarrow \text{SORTBYINCREASINGFREQUENCY}(\langle t_1, \dots, t_n \rangle)$
- 2 $result \leftarrow \text{postings}(\text{first}(terms))$
- 3 $terms \leftarrow \text{rest}(terms)$
- 4 **while** $terms \neq \text{NIL}$ and $result \neq \text{NIL}$
- 5 **do** $result \leftarrow \text{INTERSECT}(result, \text{postings}(\text{first}(terms)))$
- 6 $terms \leftarrow \text{rest}(terms)$
- 7 **return** $result$

❖ اکنون بهینه سازی پرس و جویهای عمومی تر را در نظر بگیرید مانند:
(madding OR crowd) AND (ignoble OR strife) AND (killed OR slain)
مانند قبل ما فراوانیهای تمامی عبارات را در نظر می گیریم و سپس می توانیم (به طور
مماظنه کارانه) اندازه هر عبارت فصلی پردازش کنیم. برای پرس و جوی بولی دلفوا، باید
پاسخهای عبارات میانی در یک عبارت پیچیده را ارزیابی کرده و به طور موقت ذخیره کنیم. هر
چند، در بسیاری از شرایط، به دلیل ماهیت زبان پرس و جو و یا فقط به دلیل اینکه این
متعارف ترین نوع پرس و جو است که کاربران ثبت می کنند، یک پرس و جو کاملاً عطفی
است. در این حالت، به جای نمایش ادغامی لیستهای پستها به عنوان تابعی با دو ورودی و
یک خروجی فصلی، کارآمدتر آن است که هر لیست پستهای بازبانی شده را با نتایج میانی
جاری در حافظه اشتراک بگیریم، در حالی که نتیجه میانی را با بار کردن لیست پستها از
عبارت با کمترین فراوانی مقدارهی اولیه می کنیم.



تشکر

سوال؟

a.golzari@azaruniv.ac.ir

a.golzari@tabrizu.ac.ir

<https://github.com/Amin-Golzari-Oskoue>