

# Tìm kiếm thông tin


## Chương 2. Bộ từ vựng, bộ thẻ định vị và so khớp mềm

*Soạn bởi: TS. Nguyễn Bá Ngọc*

# Nội dung

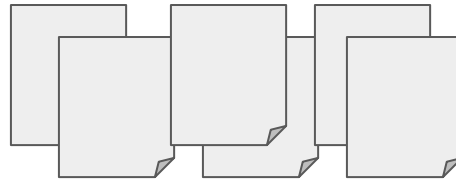
1. Tách nội dung văn bản từ tài liệu
2. Tổng hợp bộ từ vựng
3. Xử lý danh sách thẻ định vị với con trỏ ngắt quãng
4. Chỉ mục ngược có vị trí và truy vấn nguyên câu
5. Cấu trúc tra cứu từ vựng
6. So khớp mềm:
  - 6.1. Truy vấn với ký tự đại diện
  - 6.2. Sửa lỗi viết từ và gợi ý từ tìm kiếm

# Nội dung

- 
1. Tách nội dung văn bản từ tài liệu
  2. Tổng hợp bộ từ vựng
  3. Xử lý danh sách thẻ định vị với con trỏ ngắt quãng
  4. Chỉ mục ngược có vị trí và truy vấn nguyên câu
  5. Cấu trúc tra cứu từ vựng
  6. So khớp mềm:
    - 6.1. Truy vấn với ký tự đại diện
    - 6.2. Sửa lỗi viết từ và gợi ý từ tìm kiếm

# Sơ lược chuỗi thao tác tạo chỉ mục

Các văn bản được  
đánh chỉ mục



Mùa xuân đến Hoa Đào nở

Tách từ

Mùa xuân đến Hoa Đào nở

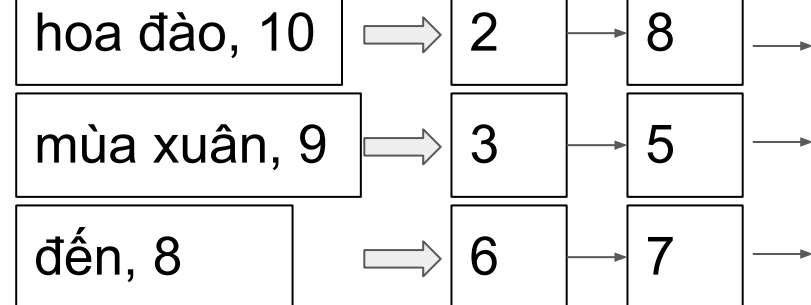
Luồng từ đặc trưng

Quy trình  
chuẩn hóa

mùa xuân đến hoa đào nở

Luồng từ chỉ mục  
(đã chuẩn hóa)

Tạo chỉ mục



Chỉ mục ngược

# Đọc văn bản

- Tài liệu có định dạng gì?
  - html, pdf, pptx, docx, v.v..
- Ngôn ngữ gì?
- Bảng mã gì?
  - CP1258, UTF-8, v.v...

*Mỗi vấn đề được liệt kê có thể được coi như một vấn đề phân lớp. Những vấn đề này thường có thể xử lý được bằng giải thuật tham lam.*

# Định dạng và ngôn ngữ

- Đôi khi các thành phần của một tài liệu có thể thuộc các ngôn ngữ khác nhau hoặc có định dạng khác nhau
  - Ví dụ: Email tiếng Nga được đính kèm tài liệu pdf tiếng anh;
  - Tài liệu tiếng Anh trích dẫn văn bản tiếng Pháp.
- Các văn bản được đánh chỉ mục có thể bao gồm các nội dung thuộc nhiều ngôn ngữ khác nhau
  - Bộ từ vựng có thể chứa từ thuộc nhiều ngôn ngữ
- Thư viện xử lý đọc nội dung văn bản của tài liệu:
  - Apache Tika - Mã nguồn mở

# Tài liệu là gì?

Chúng ta coi tài liệu là đơn vị tìm kiếm - HTTK trả về các tài liệu như những kết quả tìm kiếm.

- Với các trang Web thông thường mỗi trang Web được coi là một tài liệu.
- Với tập là sách điện tử (có thể có cả ngàn trang sách) nếu coi mỗi tập là một tài liệu thì có thể quá lớn:
  - Có nên sử dụng 1000 từ đầu tiên và bỏ qua phần còn lại?
  - Có nên chia thành nhiều tài liệu nhỏ theo các chương sách?
- Với các email (có thể có các tệp đính kèm)
  - Có nên coi tệp đính kèm là tài liệu độc lập với email?
- Tìm kiếm XML:
  - Có nên coi toàn bộ tệp XML là 1 tài liệu hay không?
  - Có nên coi 1 nút trong cây XML (ở mức nào) là 1 tài liệu?

# Mã hóa tiếng việt Unicode

- Phương pháp tổ hợp (composite) - dấu ngữ âm là ký tự được hiển thị kết hợp với ký tự cơ sở

a ă â e ê i o ô ơ u ư y

è ò ã ó ư

- Dựng sẵn (precomposed) - Các chữ cái cùng với dấu ngữ âm là 1 ký tự

1EA0		À		Chữ A hoa với dấu nặng
------	--	---	--	------------------------

- TCVN 6909:2001



# Nội dung

1. Tách nội dung văn bản từ tài liệu

2. Tổng hợp bộ từ vựng

3. Xử lý danh sách thẻ định vị với con trỏ ngắt quãng

4. Chỉ mục ngược có vị trí và truy vấn nguyên câu

5. Cấu trúc tra cứu từ vựng

6. So khớp mềm:

6.1. Truy vấn với ký tự đại diện

6.2. Sửa lỗi viết từ và gợi ý từ tìm kiếm

# Tách từ

- Đầu vào: "Mùa xuân đến Hoa Đào nở"
- Đầu ra là các **từ đặc trưng** cho văn bản, có thể là: ["Mùa xuân", "đến", "Hoa Đào", "nở", "Mùa xuân đến", ...]
- Các thuật ngữ tiếng anh: Tách từ - Tokenize, Từ đặc trưng - Token.
- Về hình thức **từ đặc trưng** trong kết quả tách từ là chuỗi con của văn bản đầu vào.
  - Khác với khái niệm từ trong ngôn ngữ học (đơn vị ngôn ngữ nhỏ nhất có nghĩa). Từ đặc trưng có thể là từ có nghĩa và cũng có thể là bất kỳ đối tượng nào khác.
- Theo mục đích TKTT **từ đặc trưng** phải bao quát được **từ truy vấn** mà **người dùng** sẽ sử dụng để **tìm kiếm** văn bản ban đầu.

# Một số vấn đề tách từ

- Làm sao để biết khi nào để ngắt từ?
  - khoa học sinh học → khoa học | sinh học, hay là khoa | học sinh | học - Nhập nhằng ý nghĩa của dấu cách (ngăn cách các âm của 1 từ hoặc ngăn cách các từ).
  - Có nên đưa tất cả các phương án vào chỉ mục?
- Có nên xóa dấu chấm hay không?
  - TP. Hà Nội → TP. có phải là một từ hay không? Giữ nguyên cụm từ này?
- Có nên xóa dấu gạch nối hay không?
  - Mát-xơ-va
  - Hà Nội-Huế-TP. Hồ Chí Minh

# Các hằng giá trị

- Ví dụ:
  - 25/08/1999                      25-Tháng 8-1999
  - 350 TCN
  - (+84) 24 3869 2463
- Thường chứa các khoảng trắng và các dấu liên kết bên trong
- Các hằng giá trị rất hữu ích cho tìm kiếm
  - Ví dụ: Tìm mã lỗi, tìm tài liệu khi biết ngày tạo
  - v.v.

# Tách từ theo khoảng trắng

- Có thể mở rộng bằng cách quy ước ký tự đặc biệt (khác khoảng trắng) làm dấu hiệu tách từ
  - Không hiệu quả nếu áp dụng trực tiếp cho văn bản:
    - Ví dụ tách văn bản "Xuân Hạ Thu Đông" thành Xuân, Hạ, Thu, Đông.
    - Tách cụm từ "Thủ đô Hà Nội" thành Thủ, đô, Hà, Nội
  - Tuy nhiên có thể giữ được tính khái quát, đơn giản và hiệu quả bằng cách kết hợp với công cụ tách từ:
    - Sử dụng công cụ tách từ để phân giải tính đa nghĩa của ký tự ngăn cách
    - Ví dụ tiền xử lý cụm từ "Thủ đô Hà Nội" thành "Thủ đô|Hà Nội" và quy ước '|' là ký hiệu tách từ.

*Chi tiết về các phương pháp tách từ hiện có trong xử lý NNTN nằm ngoài phạm vi học phần và không có trong bài giảng.*

# Từ dừng/Stop words

- Những từ đặc trưng xuất hiện trong (hầu như) tất cả các tài liệu được gọi là từ dừng:
  - Ít có giá trị để tìm kiếm nhưng có thể chiếm nhiều bộ nhớ.
  - Các hệ thống tìm kiếm thường sử dụng danh sách từ dừng để lọc dữ liệu nhằm giảm kích thước chỉ mục.
  - Một số thống kê cho thấy 30 từ được sử dụng thường xuyên nhất chiếm khoảng ~30% số lượng thẻ định vị.
- Tuy nhiên xu hướng hiện nay không lọc từ dừng
  - Giúp tìm kiếm chính xác hơn trong một số trường hợp.
    - Ví dụ truy vấn nguyên câu: "**Ta đi tới**", "**Nàng rằng: Thôi thế thì thôi!**"
    - Ví dụ bổ xung ý nghĩa: **Từ** Hà Nội **đến** Tp. Hồ Chí Minh
  - Các thuật toán nén có thể giảm đáng kể phần dung lượng cho từ dừng, các thuật toán tối ưu hóa có thể giảm đáng kể thời gian xử lý các từ dừng trong thời gian xử lý truy vấn (sẽ học sau).

# Từ chuẩn

- Trong trường hợp có chuẩn hóa chúng ta cần áp dụng đồng thời cho từ đặc trưng và từ truy vấn, để đưa các từ có nghĩa tương đương về cùng 1 dạng, làm tăng khả năng khớp từ
  - Ví dụ: Tiếng Việt vs. Tieng Viet => Tiếng Việt
- Từ sau khi chuẩn hóa được gọi là từ chuẩn - tương đương với một lớp từ.
- Các từ chuẩn thu được trong quá trình tạo chỉ mục (được gọi là từ chỉ mục) được tổng hợp thành bộ từ vựng.
- Một số thao tác chuẩn hóa thông dụng:
  - Xóa các dấu chức năng (dấu '.', dấu gạch nối '-')
    - Ví dụ U.S.A. => USA; Mat-xơ-va => Matxcova
  - Chuyển chữ hoa thành chữ thường
  - v.v..

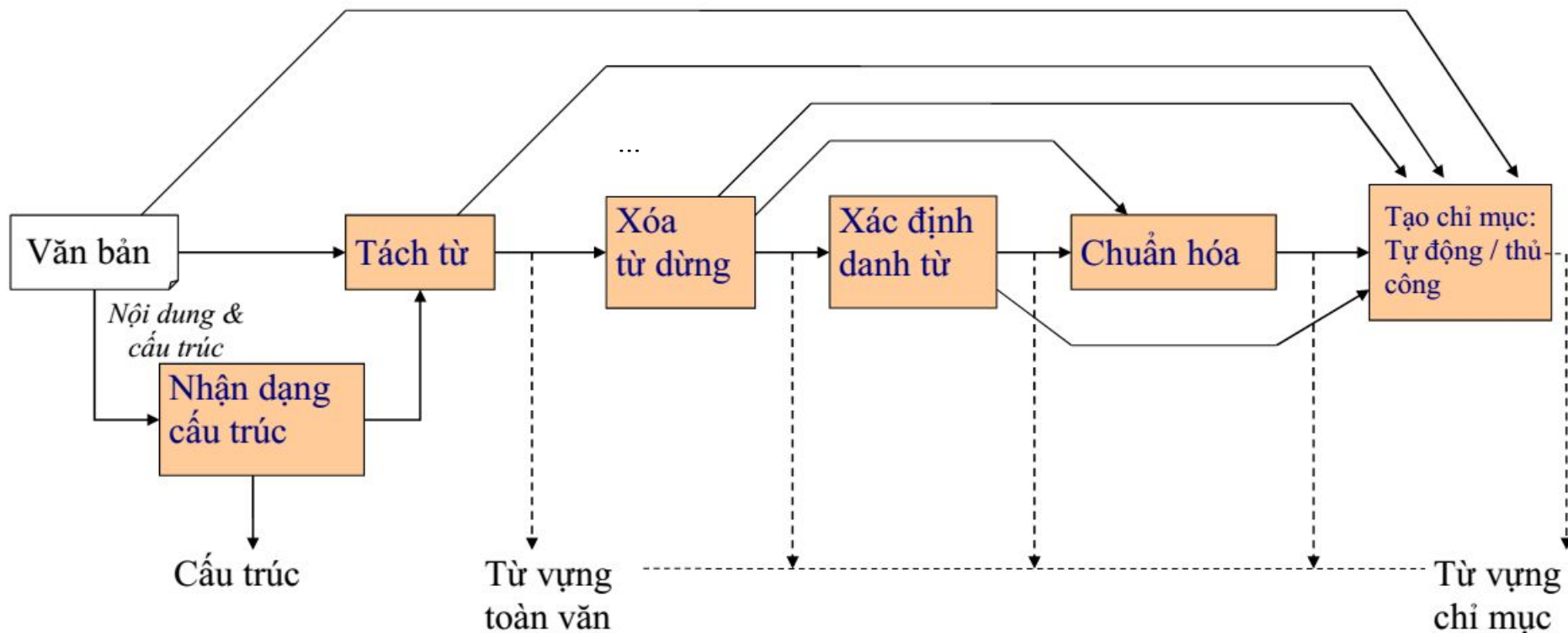
# Một số vấn đề chuẩn hóa từ tiếng việt

- Chuẩn hóa bảng mã
  - VNI, TCVN3, Unicode, v.v...
- Chuẩn hóa cách điền dấu ngữ âm
  - Nguyên âm đầu hoặc nguyên âm cuối.
- Xóa dấu
  - Làm tăng khả năng khớp nối từ truy vấn với từ chỉ mục
- Thêm dấu
  - Sử dụng dấu ngữ âm giúp tìm kiếm chính xác hơn



# Tiến trình tổng hợp bộ từ vựng

- Từ điển phổ thông không bao quát hết được các từ cần sử dụng để tìm kiếm thông tin
- Bộ từ vựng của hệ thống TKTT thường được tổng hợp trong quá trình tạo chỉ mục ngược từ các tài liệu



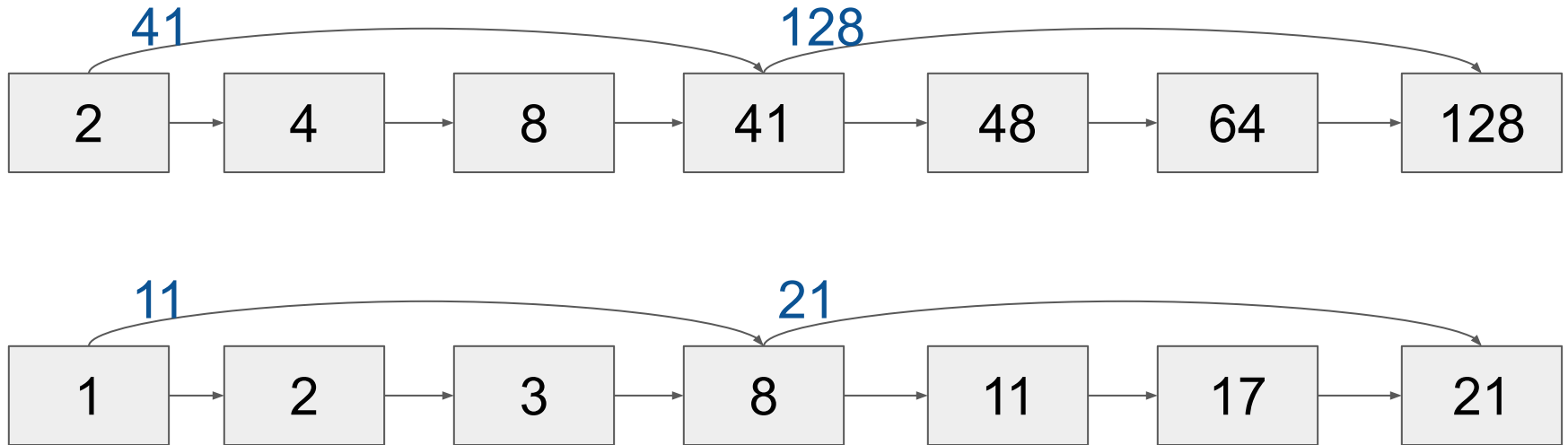
# Từ điển từ đồng nghĩa

- Các từ đồng nghĩa là các từ có cách viết khác nhau nhưng mang ý nghĩa giống nhau.
  - Ví dụ, Ô tô = Xe hơi      Máy bay = Phi cơ
- Từ điển từ đồng nghĩa có thể được biên soạn thủ công hoặc được tổng hợp bằng phương pháp tự động
  - Trong thời gian tạo chỉ mục có thể đưa tất cả các từ đồng nghĩa với từ đặc trưng vào chỉ mục
    - Ví dụ, khi văn bản chứa từ Xe hơi hoặc từ ô-tô chúng ta có thể tạo chỉ mục với các từ ô tô và xe hơi.
  - Trong thời gian xử lý truy vấn có thể sử dụng các từ đồng nghĩa để mở rộng câu truy vấn
    - Ví dụ, nếu truy vấn chứa từ xe hơi thì thêm từ ô-tô vào truy vấn

# Nội dung

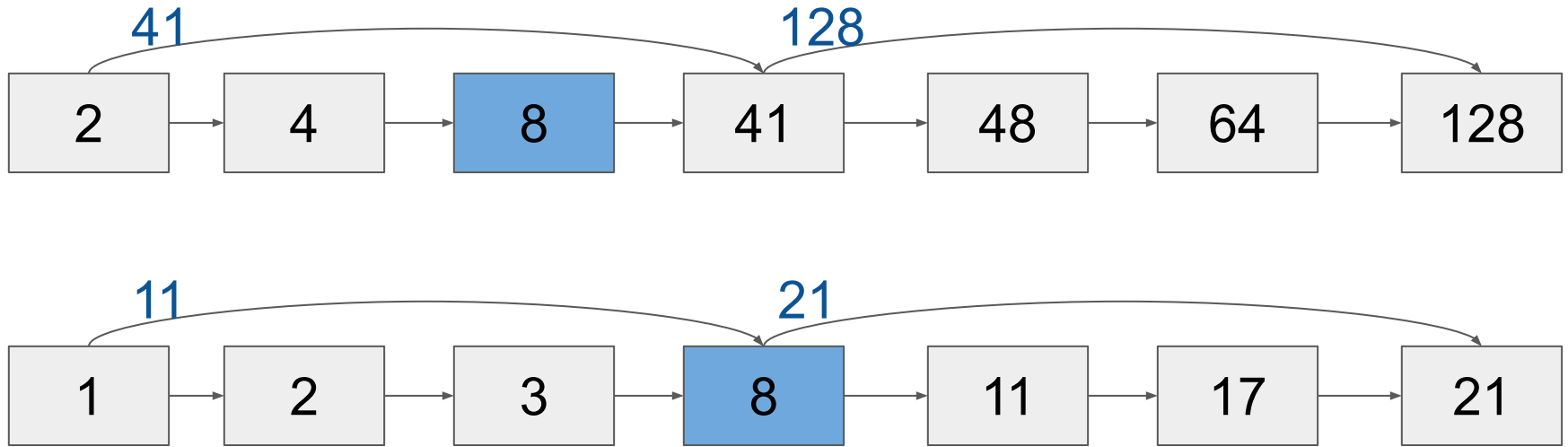
1. Tách nội dung văn bản từ tài liệu
2. Tổng hợp bộ từ vựng
3. Xử lý danh sách thẻ định vị với con trỏ ngắt quãng
4. Chỉ mục ngược có vị trí và truy vấn nguyên câu
5. Cấu trúc tra cứu từ vựng
6. So khớp mềm:
  - 6.1. Truy vấn với ký tự đại diện
  - 6.2. Sửa lỗi viết từ và gợi ý từ tìm kiếm

# Con trỏ ngắt quãng



- Con trỏ ngắt quãng (bước nhảy) là con trỏ bỏ qua một số giá trị liên tiếp trong danh sách.
- Chúng ta bổ xung các con trỏ ngắt quãng vào danh sách để định vị và sử dụng con trỏ ngắt quãng để di chuyển nhanh hơn trên danh sách (bỏ qua những thẻ định vị không thể có trong kết quả).

# Lấy giao hai danh sách có con trỏ ngẫu quăng

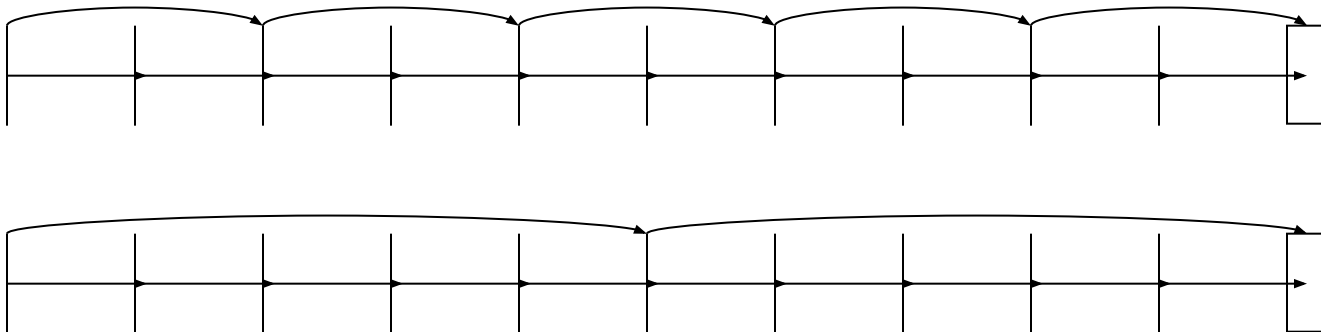


Giả sử trong quá trình duyệt danh sách các con trỏ đang ở vị trí (8, 8), các bước tiếp theo diễn ra như sau:

- Thêm 8 vào danh sách kết quả và dịch chuyển con trỏ sang phải ở cả hai danh sách.
- So sánh 41 và đích đến của bước nhảy trong danh sách thứ 2 ( $21 < 41$ )
- Dịch chuyển đến (41, 21), bỏ qua 11 và 17 ở danh sách thứ 2.
- Kết thúc xử lý.

# Độ dài của bước nhảy

- Độ dài bước nhảy là tham số có thể tùy chỉnh của giải thuật
  - Nếu nhiều bước nhảy  $\Rightarrow$  khoảng cách nhỏ  $\Rightarrow$  xác suất di chuyển theo bước nhảy cao. Nhưng phải so sánh bước nhảy nhiều lần.
  - Nếu ít bước nhảy  $\Rightarrow$  khoảng cách lớn hơn  $\Rightarrow$  xác suất di chuyển theo bước nhảy thấp hơn. Nhưng ít so sánh bước nhảy hơn
  - Những tính toán phát sinh liên quan đến bước nhảy có thể chiếm nhiều thời gian hơn phần tiết kiệm được  $\Rightarrow$  làm chậm giải thuật.



# Độ dài bước nhảy<sub>(2)</sub>

- [Moffat and Zobel 1996] sử dụng độ dài bước nhảy  $= \sqrt{L}$ , trong đó  $L$  là độ dài danh sách .
  - Cho thấy các bước nhảy giúp xử lý truy vấn hiệu quả trong đa số trường hợp
  - Tuy nhiên vẫn còn một số hạn chế:
    - Chưa tính đến các phân bố của từ truy vấn
    - Vấn đề phức tạp hơn nếu chỉ mục được cập nhật liên tục và  $L$  thay đổi thường xuyên.

# Nội dung

1. Tách nội dung văn bản từ tài liệu
2. Tổng hợp bộ từ vựng
3. Xử lý danh sách thẻ định vị với con trỏ ngắt quãng
4. Chỉ mục ngược có vị trí và truy vấn nguyên câu
5. Cấu trúc tra cứu từ vựng
6. So khớp mềm:
  - 6.1. Truy vấn với ký tự đại diện
  - 6.2. Sửa lỗi viết từ và gợi ý từ tìm kiếm



# Chỉ mục ngược có vị trí

- Mỗi thẻ định vị trong danh sách thẻ định vị bao gồm mã văn bản cùng với danh sách các vị trí xuất hiện từ
  - Vị trí của từ là chỉ số thứ tự của từ trong văn bản được xác định trong quá trình tách từ.
  - Ví dụ: "Mùa xuân<sub>1</sub> | đến<sub>2</sub> | Hoa Đào<sub>3</sub> | Nở<sub>4</sub>"
- Ví dụ một danh sách thẻ định vị có vị trí:

Hoa Đào, 118000: 1: <7, 20, 55, 89, 115>;

3: <6, 12, 28, 90, 120, 150>;

5: <2, 12, 26, 38, 52, 64, 118>;

6: <9, 35, 112, 256, 1012, 198>

*Truy vấn với giới hạn khoảng cách và truy vấn nguyên câu có thể được xử lý dựa trên chỉ mục ngược có vị trí.*

# Truy vấn với giới hạn khoảng cách

- Cấu trúc tìm kiếm trong mô hình Boolean mở rộng
- Giới hạn khoảng cách giữa các từ trong văn bản
  - Ví dụ, truy vấn: việc làm /4 lĩnh vực = Tìm tất cả văn bản chứa từ việc làm và từ lĩnh vực trong giới hạn khoảng cách 4 từ
  - Vị trí(lĩnh vực) – Vị trí(việc làm)  $\leq 4$
- Có thể coi **truy vấn nguyên câu** là trường hợp riêng của các truy vấn với giới hạn khoảng cách = 1.
  - Ví dụ: “Công nghệ thông tin” tương đương với
  - Công nghệ /1 thông tin
    - Có ràng buộc thứ tự trước - sau

## Giải thuật 2.1. Lấy giao với giới hạn khoảng cách

1	answer = $\emptyset$	15	Delete(B[0])
2	while pa $\neq$ NIL and pb $\neq$ NIL	16	for each posb $\in$ B
3	if docID(pa) == docID(pb)	17	Add(answer, <docID(pa), pos(ppa),posb>)
4	B = $\emptyset$	18	ppa = next(ppa)
5	ppa = positions(pa)	19	pa = next(pa)
6	ppb = positions(pb)	20	pb = next(pb)
7	while ppa $\neq$ NIL	21	else if docID(pa) < docID(pb)
8	while ppb $\neq$ NIL	22	pa = next(pa)
9	if  pos(ppa) - pos(ppb)  $\leq$ k	23	else pb = next(pb)
10	Add(B, pos(ppb))	24	return answer
11	else if pos(ppb) > pos(ppa)		
12	break		
13	ppb = next(ppb)		
14	while B $\neq$ $\emptyset$ and  B[0] - pos(ppa)  > k		

## Ví dụ 2.1. Truy vấn với giới hạn khoảng cách

Ví dụ chỉ mục có vị trí:

tìm\_kiểm, 5: 1: <1>; 2: <6>; 3: <2, 15>; 4: <1>, 8:<2>.

dữ\_liệu, 5: 1: <3>; 3: <5, 16>; 4: <6>; 7: <14>, 8:<5>.

thông\_tin, 5: 1: <2>; 2: <12, 16, 21>; 3: <18>; 5: <21, 25>, 8:<3>

Ví dụ truy vấn với giới hạn khoảng cách: tìm\_kiểm /2 dữ\_liệu

Kết quả: {1, 3}

*Trong bài giảng này chỉ mục ngược có vị trí được biểu diễn theo định dạng:*

Từ chỉ mục, df: mã\_văn\_bản: <danhsách\_vị\_trí>; mã\_văn\_bản:

<danhsách\_vị\_trí>; ...

# Nội dung

1. Tách nội dung văn bản từ tài liệu
2. Tổng hợp bộ từ vựng
3. Xử lý danh sách thẻ định vị với con trỏ ngắt quãng
4. Chỉ mục ngược có vị trí và truy vấn nguyên câu
5. Cấu trúc tra cứu từ vựng
6. So khớp mềm:
  - 6.1. Truy vấn với ký tự đại diện
  - 6.2. Sửa lỗi viết từ và gợi ý từ tìm kiếm

# Một từ điển đơn giản

- Mảng cấu trúc:

Từ chỉ mục	Tần suất văn bản	Con trỏ
a	65000	→
anh	42100	→
...	...	...
zip	135	→

char[20]

20 bytes

int

4/8 bytes

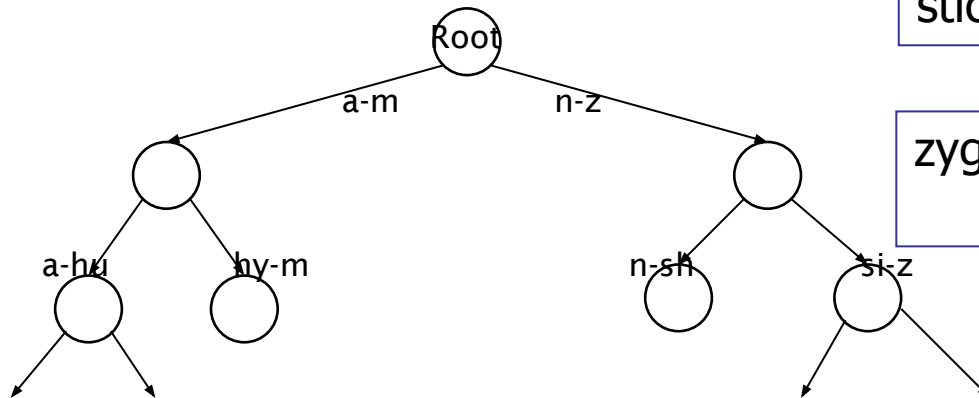
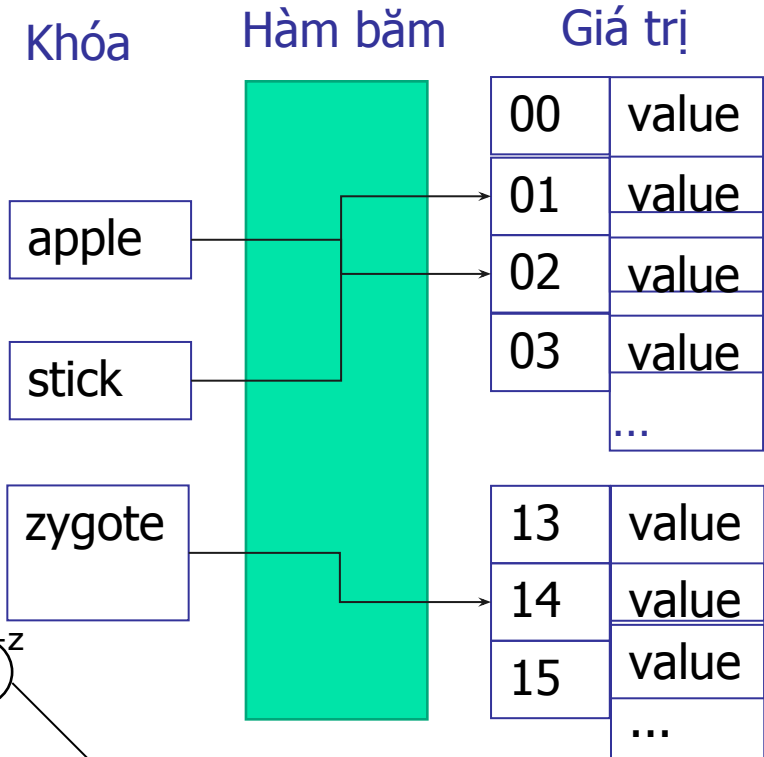
Postings \*

4/8 bytes

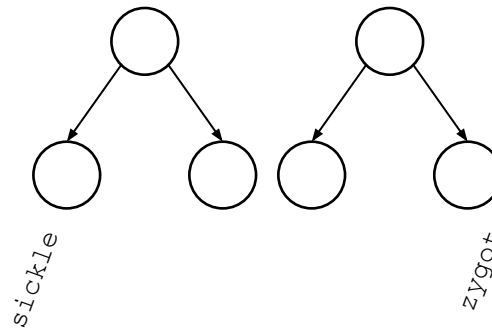
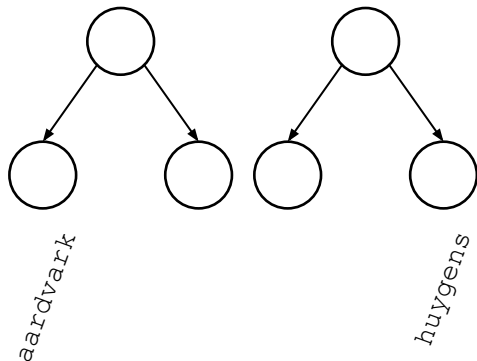
- Lưu trữ từ điển trong bộ nhớ sao cho tiết kiệm?
  - Nén từ điển (sẽ học sau)
- Làm sao để tra cứu nhanh trong thời gian xử lý truy vấn?

# Cấu trúc dữ liệu từ điển

- Hai lựa chọn cơ bản:
  - Bảng băm vs. Cây tìm kiếm
  - Một số HTTK sử dụng bảng băm
  - một số khác sử dụng cây



...



# Một số đặc tính của bảng băm

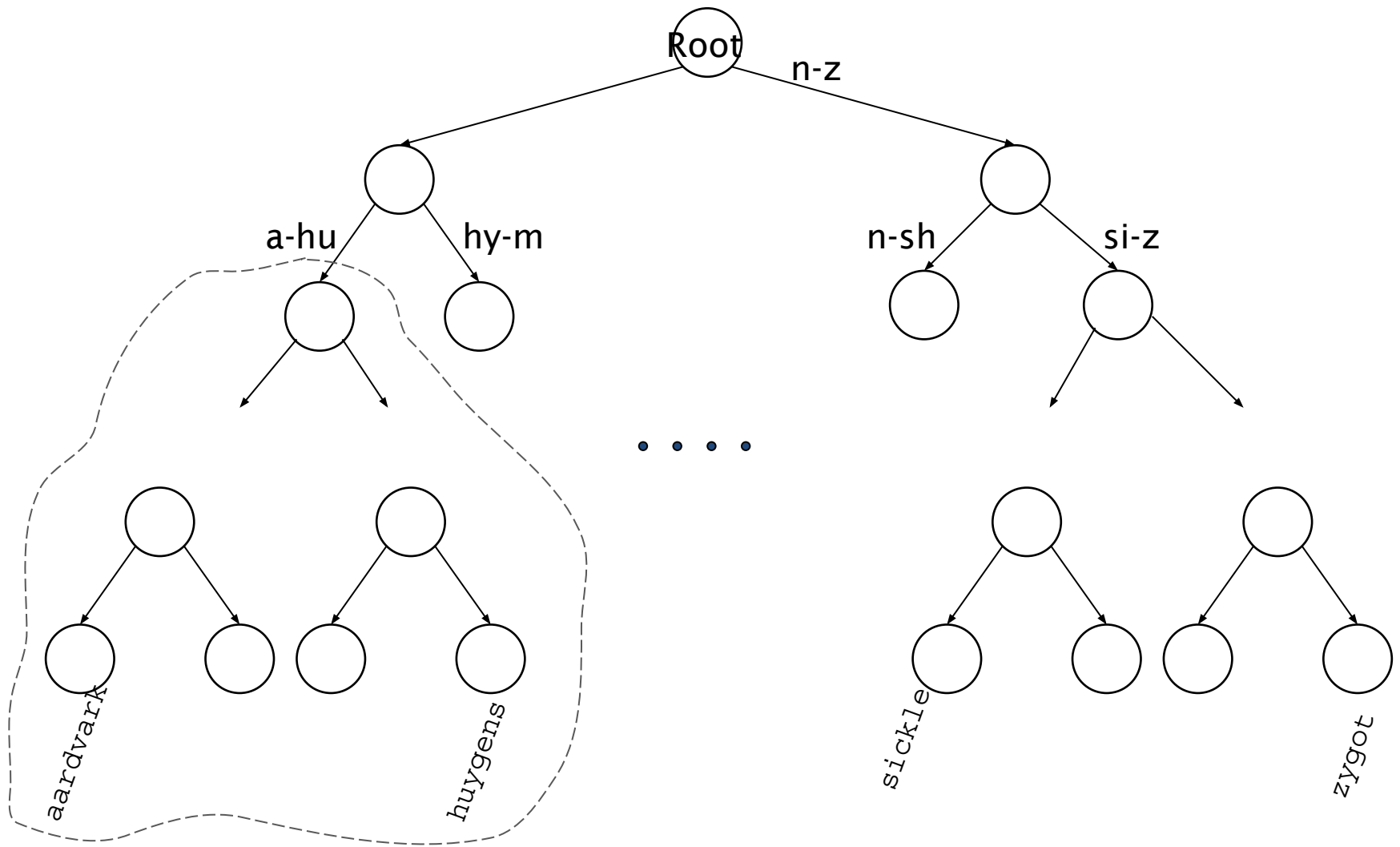
- Mỗi từ chỉ mục được băm thành một số nguyên
- Điểm mạnh:
  - Độ phức tạp tìm kiếm:  $O(1)$  (nhanh hơn cây tìm kiếm)
- Điểm yếu:
  - Không tìm được các từ tương tự:
    - Không xử lý được mẫu từ
    - Ví dụ: Từ có phần bắt đầu là Mùa
    - Do không duy trì trật tự sắp xếp các từ
  - Có thể sử dụng nhiều bộ nhớ



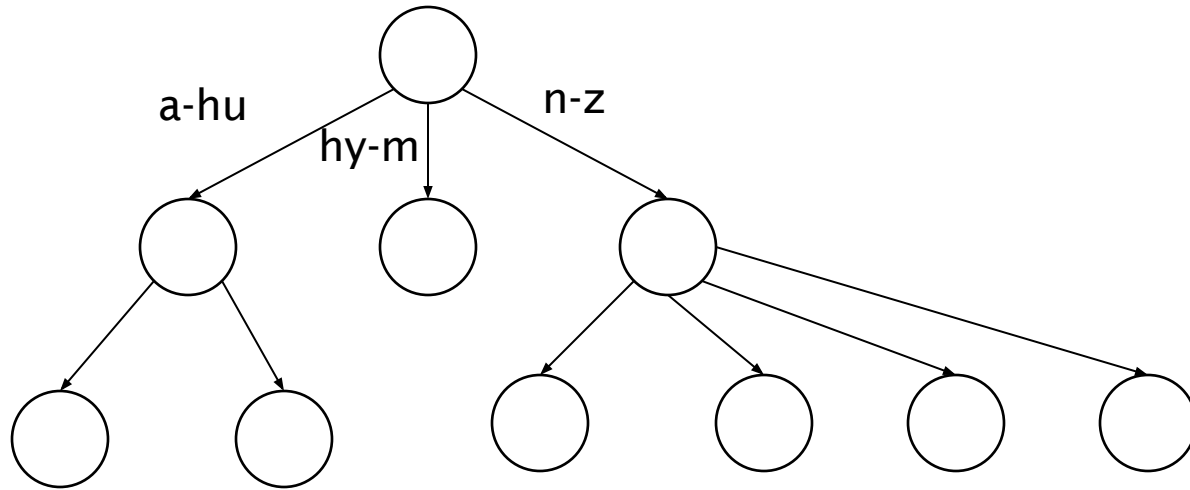
# Một số đặc tính của cây tìm kiếm

- Nếu sử dụng RAM: Cây nhị phân tìm kiếm cân bằng
- Nếu sử dụng ổ đĩa cứng: Cây-B (hoặc B+)
- Cây tìm kiếm có duy trì trật tự sắp xếp các từ
- Điểm mạnh
  - Cho phép tìm kiếm lân cận, xử lý từ với các ký tự đại diện
    - Ví dụ, từ có phần kết thúc là ng (\*ng)
- Điểm yếu
  - Tìm kiếm chậm hơn bảng băm:  $O(\log M)$  đối với cây cân bằng
  - Cân bằng cây trên ổ đĩa cứng là một thao tác phức tạp
    - Cây-B hạn chế các trường hợp phải thực hiện cân bằng để làm việc hiệu quả hơn với ổ đĩa cứng

# Cấu trúc cây nhị phân tìm kiếm



# Cấu trúc cây-B



*Tốc độ xử lý chậm hơn cây nhị phân cân bằng (trong RAM), nhưng hiệu quả hơn trong trường hợp sử dụng ổ đĩa cứng và có thể lưu những bộ từ vựng lớn (vượt qua kích thước RAM).*

# Nội dung

1. Tách nội dung văn bản từ tài liệu
2. Tổng hợp bộ từ vựng
3. Xử lý danh sách thẻ định vị với con trỏ ngắt quãng
4. Chỉ mục ngược có vị trí và truy vấn nguyên câu
5. Cấu trúc tra cứu từ vựng
- 6. So khớp mềm:**



6.1. Truy vấn với ký tự đại diện

6.2. Sửa lỗi viết từ và gợi ý từ tìm kiếm

# So khớp mềm

- Hỗ trợ người dùng viết từ truy vấn đặc biệt là những từ truy vấn khó bằng cách nào?
  - Mô hình Boolean mở rộng: Sử dụng mẫu từ trong truy vấn
  - Xu hướng hiện nay: Gợi ý từ khóa
- Trong trường hợp người dùng viết sai từ truy vấn hệ thống tìm kiếm có nên đưa ra gợi ý hay không?
  - Giả sử có rất ít kết quả đối với từ truy vấn do người dùng nhập vào nhưng có rất nhiều kết quả cho một từ khác tương tự có trong chỉ mục. Hệ thống có nên đưa ra kết quả cho từ có trong chỉ mục thay vì các kết quả cho từ do người dùng nhập vào hay không?

# Truy vấn với ký tự đại diện

- Từ truy vấn có chứa dấu \* tương đương với một nhóm từ chỉ mục, và được gọi là mẫu từ.
- Dấu \* có thể xuất hiện ở các vị trí:
  - Cuối từ - a\*: Tương đương với tất cả từ bắt đầu với a
  - Đầu từ - \*a: Tất cả từ kết thúc với a
  - Giữa từ a\*b: Tất cả từ bắt đầu với a, kết thúc với b
  - Trong đó a, b có thể là các nhóm ký tự bất kỳ
- Xử lý truy vấn với ký tự đại diện:
  - Tìm tập từ chỉ mục khớp với mẫu từ, sau đó trả về văn bản chứa bất kỳ từ nào trong tập từ khớp mẫu (tương đương liên kết OR).

*Cần tạo chỉ mục trên từ để lọc nhanh các từ chỉ mục tương thích với mẫu từ.*

# Tìm từ chỉ mục tương thích mẫu từ

## Sử dụng các cây tìm kiếm

- Mẫu từ  $a^*$ , ví dụ  $\text{mon}^*$ :
  - Lấy các từ trong khoảng:  $\text{mon} \leq t < \text{moo}$  trong cây từ vựng
- Mẫu từ  $*b$ , ví dụ  $*\text{mon}$ :
  - Nếu có thể tạo cây chứa từ được viết theo chiều ngược lại
  - Lấy tất cả từ trong khoảng:  $\text{nom} \leq t < \text{non}$
- Mẫu từ  $a*b$ :
  - Nếu có thể sử dụng 2 cây nhị phân tìm kiếm lưu các từ theo chiều thuận và ngược.
  - Tìm các tập từ cho mẫu  $a^*$  và mẫu  $*b$
  - Lấy giao của các tập từ tìm được.
  - $\Rightarrow$  Tốn nhiều bộ nhớ và chậm.

# Chỉ mục từ xoay

- Các từ thu được bằng cách xoay vòng các ký tự có trong từ được gọi là các từ xoay.
- Giả sử \$ là một ký tự không xuất hiện trong bất kỳ từ nào. Chúng ta sẽ sử dụng \$ làm ký hiệu đánh dấu vị trí kết thúc từ để có thể khôi phục từ gốc bằng từ xoay.
  - Thêm dấu \$ vào cuối trước khi bắt đầu xoay từ
  - Ví dụ, Các từ xoay cho từ hello là: hello\$, ello\$h, llo\$he, lo\$hel, o\$hell, và \$hello, trong đó \$ là ký tự đặc biệt không xuất hiện trong bất kỳ từ nào.
- Chỉ mục được tạo thành từ các từ xoay được gọi là chỉ mục từ xoay - Permuterm Index.



# Chỉ mục từ xoay<sub>(2)</sub>

- Ý tưởng xử lý mẫu từ:
  - Thêm dấu \$ vào cuối mẫu
  - Xoay mẫu từ sao cho dấu \* xuất hiện ở cuối, qua đó đưa tất cả các mẫu từ về dạng a\*.
  - Xử lý mẫu từ trên chỉ mục từ xoay, sau đó chuyển các từ xoay về dạng gốc.
- Ví dụ xoay mẫu từ:
  - $X^* \Rightarrow X^*\$ \Rightarrow \$X^*$
  - $*X \Rightarrow *X\$ \Rightarrow X\$^*$
  - $X^*Y \Rightarrow X^*Y\$ \Rightarrow Y\$X^*$
  - $hel^*o \Rightarrow hel^*o\$ \Rightarrow o\$hel^*$

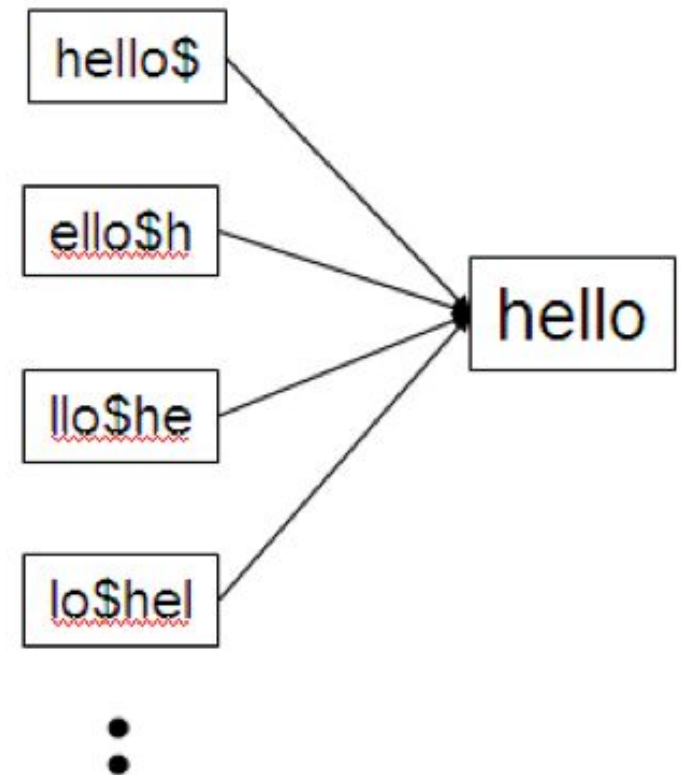
# Chỉ mục từ xoay<sub>(3)</sub>

- Liên kết từ xoay với từ gốc n-1.

Cấu trúc thu được là cây tìm kiếm hay chỉ mục ngược?

*Độ dài thẻ định vị?*

Về bản chất chỉ mục từ xoay là cây từ xoay (không cần lưu các danh sách thẻ định vị).



*\*Vấn đề: Cây từ xoay có kích thước lớn gấp nhiều lần cây từ vựng ban đầu.*

# Chỉ mục k-grams trên ký tự

- Khái niệm: k-grams/dãy k, k-grams theo ký tự/dãy k-ký tự
- Ý tưởng: Lưu các dãy k-ký tự
  - Trong trường hợp này dãy 2 ký tự được gọi ngắn gọn là dãy kép.
- Ví dụ các dãy kép cho từ April là: \$A, ap, pr, ri, il, l\$
  - Giả sử \$ là ký tự không xuất hiện trong bất kỳ từ nào (tương tự như trong từ xoay).
  - Trước khi tạo các dãy kép chúng ta bổ xung thêm các dấu \$ vào đầu và cuối từ để đánh dấu các ký tự đầu tiên và cuối cùng của từ.
- Với các chỉ mục dãy kép chúng ta có hai lớp chỉ mục ngược
  - Chỉ mục dãy kép để tìm từ chỉ mục tương thích với mẫu từ
  - Chỉ mục ngược cho bộ văn bản.

# Xử lý mẫu từ bằng chỉ mục dãy kép

- Ý tưởng:
  - Biến đổi mẫu từ thành truy vấn toàn AND với các dãy kép
  - Tìm kiếm trong chỉ mục dãy kép
  - Lọc các từ tìm được để loại các từ không tương thích với mẫu từ.
- Ví dụ truy vấn mon\* sẽ được biến đổi thành biểu thức: \$m AND mo AND on
  - Ngoài các từ bắt đầu với mon, các từ khác cũng có thể được trả về trong tập kết quả xử lý truy vấn trên chỉ mục dãy kép, ví dụ Moon (Lỗi False Positives - trả về sai).

# Chỉ mục dãy k vs. chỉ mục từ xoay

- Bộ nhớ vs. tốc độ xử lý:
  - Chỉ mục dãy k chiếm ít bộ nhớ hơn so với chỉ mục từ xoay tuy nhiên xử lý mẫu từ trên chỉ mục từ xoay nhanh hơn (và không cần lọc từ tìm được).

# Truy vấn mẫu từ trong HTTK hiện đại

- Google chỉ hỗ trợ truy vấn mẫu từ ở mức độ hạn chế.
- Ví dụ, Google xử lý không tốt truy vấn [gen\* universit\*]
  - Tình huống: Tìm the University of Geneva, nhưng không biết cách viết chính xác các từ university và Geneva.
- Theo thông tin chính thức từ Google, 2010-04-29: “Dấu \* chỉ có thể thay thế cho một từ hoàn chỉnh, không phải một phần của từ.”
  - Thử kiểm tra hành vi thực tế, ví dụ, [pythag\*] và [m\*nchen]

Thử tìm hiểu vì sao Google hạn chế truy vấn mẫu từ?

# Truy vấn mẫu từ trong HTTK hiện đại<sub>(2)</sub>

- Vấn đề 1: Truy vấn Boolean dài và có rất nhiều kết quả
  - Các từ tương thích với mẫu từ được kết hợp bằng toán tử OR;
  - Đối với [gen\* universit\*] sẽ tìm geneva university OR geneva université OR genève university OR genève université OR general universities OR ...
  - => Khối lượng tính toán rất lớn nhưng chưa chắc sẽ đem lại lợi ích
- Vấn đề 2: Người dùng có thể lạm dụng mẫu từ
  - Nếu mẫu [pyth\* theo\*] là hợp lệ cho [pythagoras' theorem] người dùng sẽ sử dụng thường xuyên.
  - => Chi phí thực hiện truy vấn có thể tăng đáng kể.
- Vấn đề viết từ truy vấn phức tạp đã được giải quyết một phần bằng cơ chế gợi ý truy vấn.

# Nội dung

1. Tách nội dung văn bản từ tài liệu
2. Tổng hợp bộ từ vựng
3. Xử lý danh sách thẻ định vị với con trỏ ngắt quãng
4. Chỉ mục ngược có vị trí và truy vấn nguyên câu
5. Cấu trúc tra cứu từ vựng
- 6. So khớp mềm:**

6.1. Truy vấn với ký tự đại diện

6.2. Sửa lỗi viết từ và gợi ý từ tìm kiếm





# Sửa lỗi viết từ

- Hai trường hợp sử dụng cơ bản
  - Sửa lỗi từ đặc trưng trong quá trình đánh chỉ mục
  - Sửa lỗi từ truy vấn được viết bởi người dùng
- Hai cách tiếp cận cơ bản
  - Cô lập từ
    - Kiểm tra cách viết từ không phụ thuộc vào ngữ cảnh sử dụng
    - Không phát hiện được lỗi chính tả, hoặc viết nhầm
    - Ví dụ, xử (lý) / sử (lý)
  - Kiểm tra trong ngữ cảnh
    - Kiểm tra cách viết từ trong ngữ cảnh sử dụng
    - Xét các từ lân cận
    - Ví dụ, sử lý tiếng nói => xử lý tiếng nói

# Sửa lỗi viết từ<sub>(2)</sub>

- Có thể sử dụng tri thức lĩnh vực
  - Ví dụ, OCR có thể thường xuyên nhầm lẫn O và D (nhìn giống nhau);
  - Nếu sử dụng bàn phím QWERTY để nhập văn bản thì có khả năng nhầm lẫn O và I (các ký tự gần nhau trên bàn phím).
- Trong thực tế thường không thay đổi từ đặc trưng cho nội dung văn bản, thay vào đó thường sửa từ truy vấn được nhập bởi người dùng
  - Mặc dù các văn bản được đánh chỉ mục hoàn toàn có khả năng chứa lỗi.
  - ... Tuy nhiên các từ có lỗi thường chiếm tỉ lệ rất nhỏ.

# Sửa lỗi viết từ truy vấn

- Ứng dụng quan trọng trong HTTK
- Ví dụ, truy vấn Bjarne Struostrop => Bjarne Stroustrup
- Chúng ta có thể
  - Trả về các văn bản chứa phương án viết đúng của từ truy vấn hoặc
  - Trả về gợi ý phương án thay thế cho truy vấn ban đầu
    - Bạn đang muốn tìm ....? Did you mean ...?

# Sửa từ trong điều kiện cô lập

- Cơ sở: Một bộ từ vựng có thể là:

- Từ điển ngôn ngữ tự nhiên

- Đại từ điển tiếng Việt, các bộ từ điển được tổng hợp thủ công, v.v..

- **Bộ từ vựng của chỉ mục ngược**

- Ví dụ, tất cả từ trên Web
- Tất cả tên, các ký hiệu viết tắt
- (bao gồm cả những cách viết sai)

*Thường được sử dụng cho máy tìm kiếm*

- Các bước cơ bản:

- Tìm các từ chỉ mục gần từ truy vấn nhất
- Xác định lựa chọn tốt nhất

- Gần nhất là gì?

- Chúng ta cần sử dụng đại lượng định lượng:
  - Khoảng cách soạn thảo (Levenshtein)
  - Tỷ lệ chồng lấn n-gram
  - V.v..

# Khoảng cách soạn thảo

- Cho hai chuỗi  $S_1$  và  $S_2$ , khoảng cách soạn thảo là số **thao tác soạn thảo tối thiểu** cần thực hiện để biến đổi chuỗi này thành chuỗi kia
  - Khoảng cách soạn thảo thường là đại lượng đối xứng.
  - Các thao tác thường được thực hiện ở mức ký tự
  - Khoảng cách Levenshtein đếm các thao tác: Chèn, Xóa, Thay đổi
  - Khoảng cách **Demerau-Levenshtein** bổ xung thao tác Hoán đổi.
- Ví dụ, cho 2 từ dgo và dog.
  - Khoảng cách Levenshtein = 2 (có thể thay đổi g->o và o->g).
  - Khoảng cách Demerau-Levenshtein = 1 (hoán đổi g và o)
- Khoảng cách soạn thảo Levenshtein có thể được tính bằng phương pháp quy hoạch động

## Giải thuật 2.2. Tính khoảng cách Levenshtein

Giải thuật quy hoạch động:

LEVENSHTEINDISTANCE( $s_1, s_2$ )

```
1  for  $i \leftarrow 0$  to  $|s_1|$ 
2  do  $m[i, 0] = i$ 
3  for  $j \leftarrow 0$  to  $|s_2|$ 
4  do  $m[0, j] = j$ 
5  for  $i \leftarrow 1$  to  $|s_1|$ 
6  do for  $j \leftarrow 1$  to  $|s_2|$ 
7      do if  $s_1[i] = s_2[j]$ 
8          then  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]\}$ 
9          else  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1\}$ 
10 return  $m[|s_1|, |s_2|]$ 
```

Operations: insert (cost 1), delete (cost 1), replace (cost 1), copy (cost 0)

## Ví dụ 2.2. Tính khoảng cách Levenshtein

Chi tiết các bước tính khoảng cách Levenshtein(cats, fast):

		f	a	s	t
	<div><div></div><div>0</div></div>	<div><div>1</div><div>1</div></div>	<div><div>2</div><div>2</div></div>	<div><div>3</div><div>3</div></div>	<div><div>4</div><div>4</div></div>
c	<div><div></div><div>1</div></div> <div><div></div><div>1</div></div>	<div><div>1</div><div>2</div></div> <div><div>2</div><div>1</div></div>			
a	<div><div></div><div>2</div></div> <div><div></div><div>2</div></div>				
t	<div><div></div><div>3</div></div> <div><div></div><div>3</div></div>				
s	<div><div></div><div>4</div></div> <div><div></div><div>4</div></div>				

## Ví dụ 2.2. Tính khoảng cách Levenshtein<sub>(2)</sub>

Chi tiết các bước tính khoảng cách Levenshtein(cats, fast):

		f	a	s	t
	<div><div></div><div>0</div></div>	<div><div>1</div><div>1</div></div>	<div><div>2</div><div>2</div></div>	<div><div>3</div><div>3</div></div>	<div><div>4</div><div>4</div></div>
c	<div><div></div><div>1</div><div>1</div></div>	<div><div>1</div><div>2</div></div>	<div><div><div><div></div><div></div></div><div><div>2</div><div></div></div></div><div><div>2</div><div>1</div></div><div>→</div><div><div>2</div><div></div></div></div>		
a	<div><div></div><div>2</div><div>2</div></div>				
t	<div><div></div><div>3</div><div>3</div></div>				
s	<div><div></div><div>4</div><div>4</div></div>				

Chi phí nếu biến đổi c thành f rồi chèn a

*Chi phí nếu biến đổi c thành f rồi chèn a*



## Ví dụ 2.2. Tính khoảng cách Levenshtein<sub>(3)</sub>

Chi tiết các bước tính khoảng cách Levenshtein(cats, fast):

		f	a	s	t
		1 1	2 2	3 3	4 4
c		1 1	2 2		
a		2 2	1 1		
t		3 3			
s		4 4			

Chi phí xóa c rồi biến đổi rỗng thành fa

*Chi phí xóa c rồi biến đổi rỗng thành fa*

## Ví dụ 2.2. Tính khoảng cách Levenshtein<sub>(4)</sub>

Chi tiết các bước tính khoảng cách Levenshtein(cats, fast):

		f	a	s	t
	<div><div></div><div>0</div></div>	<div><div>1</div><div>1</div></div>	<div><div>2</div><div>2</div></div>	<div><div>3</div><div>3</div></div>	<div><div>4</div><div>4</div></div>
c	<div><div>1</div><div>1</div></div>	<div><div>1</div><div>2</div></div>	<div><div><b>2</b></div><div>2</div></div>		
a	<div><div>2</div><div>2</div></div>	<div><div>2</div><div>1</div></div>	<div><div>2</div><div></div></div>		
t	<div><div>3</div><div>3</div></div>				
s	<div><div>4</div><div>4</div></div>				

*Chi phí nếu biến đổi phần rỗng thành f rồi sửa c thành a*

## Ví dụ 2.2. Tính khoảng cách Levenshtein<sub>(5)</sub>

Chi tiết các bước tính khoảng cách Levenshtein(cats, fast):

		f	a	s	t
		1	2	3	4
c	1	1	2	2	
a	2	2	1	2	
t	3				
s	4				

Chi phí tối ưu để biến đổi c thành fa =  $\min(2, 2, 2)$

## Ví dụ 2.2. Tính khoảng cách Levenshtein<sub>(6)</sub>

$$\text{Levenshtein}(\text{cats}, \text{fast}) = 3$$

		f	a	s	t
	<div><div></div><div>0</div></div>	<div><div>1</div><div>1</div></div>	<div><div>2</div><div>2</div></div>	<div><div>3</div><div>3</div></div>	<div><div>4</div><div>4</div></div>
c	<div><div>1</div><div>1</div></div>	<div><div>1</div><div>2</div><div>2</div><div>1</div></div>	<div><div>2</div><div>3</div><div>2</div><div>2</div></div>	<div><div>3</div><div>4</div><div>3</div><div>3</div></div>	<div><div>4</div><div>5</div><div>4</div><div>4</div></div>
a	<div><div>2</div><div>2</div></div>	<div><div>2</div><div>2</div><div>3</div><div>2</div></div>	<div><div>1</div><div>3</div><div>3</div><div>1</div></div>	<div><div>3</div><div>4</div><div>2</div><div>2</div></div>	<div><div>4</div><div>5</div><div>3</div><div>3</div></div>
t	<div><div>3</div><div>3</div></div>	<div><div>3</div><div>3</div><div>4</div><div>3</div></div>	<div><div>3</div><div>2</div><div>4</div><div>2</div></div>	<div><div>2</div><div>3</div><div>3</div><div>2</div></div>	<div><div>2</div><div>4</div><div>3</div><div>2</div></div>
s	<div><div>4</div><div>4</div></div>	<div><div>4</div><div>4</div><div>5</div><div>4</div></div>	<div><div>4</div><div>3</div><div>5</div><div>3</div></div>	<div><div>2</div><div>3</div><div>4</div><div>2</div></div>	<div><div>3</div><div>3</div><div>3</div><div>3</div></div>

# Ý nghĩa các giá trị trong ma trận

$s1[i] == s2[j] ?$

$m[i - 1, j - 1]:$

$m[i - 1, j - 1] + 1$

*copy*

*replace*

$m[i - 1, j] + 1$

*delete(s<sub>1</sub>[i])*

$m[i, j - 1] + 1$

*insert (s<sub>2</sub>[j])*

*min*

## Ví dụ 2.2. Tính khoảng cách Levenshtein<sub>(7)</sub>

Các thao tác trong phương án tối ưu

		f	a	s	t
	<div><div></div><div>0</div></div>	<div><div>1</div><div>1</div></div>	<div><div>2</div><div>2</div></div>	<div><div>3</div><div>3</div></div>	<div><div>4</div><div>4</div></div>
c	<div><div>1</div><div>1</div></div>	<div><div>1</div><div>2</div><div>2</div><div>1</div></div>	<div><div>2</div><div>3</div><div>2</div><div>2</div></div>	<div><div>3</div><div>4</div><div>3</div><div>3</div></div>	<div><div>4</div><div>5</div><div>4</div><div>4</div></div>
a	<div><div>2</div><div>2</div></div>	<div><div>2</div><div>2</div><div>3</div><div>2</div></div>	<div><div>1</div><div>3</div><div>3</div><div>1</div></div>	<div><div>3</div><div>4</div><div>2</div><div>2</div></div>	<div><div>4</div><div>5</div><div>3</div><div>3</div></div>
t	<div><div>3</div><div>3</div></div>	<div><div>3</div><div>3</div><div>4</div><div>3</div></div>	<div><div>3</div><div>2</div><div>4</div><div>2</div></div>	<div><div>2</div><div>3</div><div>3</div><div>2</div></div>	<div><div>2</div><div>4</div><div>3</div><div>2</div></div>
s	<div><div>4</div><div>4</div></div>	<div><div>4</div><div>4</div><div>5</div><div>4</div></div>	<div><div>4</div><div>3</div><div>5</div><div>3</div></div>	<div><div>2</div><div>3</div><div>4</div><div>2</div></div>	<div><div>3</div><div>3</div><div>3</div><div>3</div></div>

Sửa c thành f -> copy a -> sửa t thành s -> Sửa s thành t

## Ví dụ 2.2. Tính khoảng cách Levenshtein<sub>(8)</sub>

Các thao tác trong phương án tối ưu

		f	a	s	t
	<div><div></div><div>0</div></div>	<div><div>1</div><div>1</div></div>	<div><div>2</div><div>2</div></div>	<div><div>3</div><div>3</div></div>	<div><div>4</div><div>4</div></div>
c	<div><div>1</div><div>1</div></div>	<div><div>1</div><div>2</div><div>2</div><div>1</div></div>	<div><div>2</div><div>3</div><div>2</div><div>2</div></div>	<div><div>3</div><div>4</div><div>3</div><div>3</div></div>	<div><div>4</div><div>5</div><div>4</div><div>4</div></div>
a	<div><div>2</div><div>2</div></div>	<div><div>2</div><div>2</div><div>3</div><div>2</div></div>	<div><div>1</div><div>3</div><div>3</div><div>1</div></div>	<div><div>3</div><div>4</div><div>2</div><div>2</div></div>	<div><div>4</div><div>5</div><div>3</div><div>3</div></div>
t	<div><div>3</div><div>3</div></div>	<div><div>3</div><div>3</div><div>4</div><div>3</div></div>	<div><div>3</div><div>2</div><div>4</div><div>2</div></div>	<div><div>2</div><div>3</div><div>3</div><div>2</div></div>	<div><div>2</div><div>4</div><div>3</div><div>2</div></div>
s	<div><div>4</div><div>4</div></div>	<div><div>4</div><div>4</div><div>5</div><div>4</div></div>	<div><div>4</div><div>3</div><div>5</div><div>3</div></div>	<div><div>2</div><div>3</div><div>4</div><div>2</div></div>	<div><div>3</div><div>3</div><div>3</div><div>3</div></div>

Sửa c thành f -> Sao chép a -> Xóa t -> Sao chép s -> Chèn t

## Ví dụ 2.2. Tính khoảng cách Levenshtein<sub>(9)</sub>

Ghi chú giản lược (chỉ ghi các giá trị tối ưu)

		<b>f</b>	<b>a</b>	<b>s</b>	<b>t</b>
	0	1	2	3	4
<b>c</b>	1	1	2	3	4
<b>a</b>	2	2	1	2	3
<b>t</b>	3	3	2	2	2
<b>s</b>	4	4	3	2	3



# Khoảng cách soạn thảo có trọng số

- Tương tự như khoảng cách soạn thảo, nhưng sử dụng trọng số cho thao tác dựa trên các ký tự được xử lý
  - Để xử lý các lỗi OCR hoặc lỗi nhập từ bàn phím; Ví dụ: m có khả năng nhầm với n hơn so với q. Vì vậy, thay m bằng n có khoảng cách soạn thảo nhỏ hơn thay bằng q.
  - Đặc điểm này có thể được biểu diễn bằng mô hình xác suất
- Yêu cầu ma trận trọng số như dữ liệu đầu vào

*Điều chỉnh giải thuật 2.2 để tính khoảng cách soạn thảo có trọng số?*

# Lựa chọn từ tốt nhất

Ý tưởng:

- Tìm tất cả các từ chỉ mục có khoảng cách tới từ truy vấn nằm trong giới hạn, ví dụ khoảng cách Levenshtein không vượt quá 2.
- Lựa chọn những từ có danh sách thể định vị dài nhất (phổ biến nhất).
- Sau đó:
  - Đưa ra gợi ý cho người dùng nếu từ chỉ mục được lựa chọn có rất nhiều kết quả so với từ truy vấn ban đầu, hoặc
  - Bổ xung thêm các kết quả tương ứng với từ được lựa chọn vào kết quả xử lý truy vấn.

# Tối ưu hóa xử lý

- Vấn đề: Xử lý đơn giản bằng cách tính khoảng cách soạn thảo giữa từ truy vấn với tất cả từ chỉ mục sẽ rất chậm và tốn nhiều chi phí tính toán.
- Làm cách nào để giới hạn tập từ cần tính khoảng cách soạn thảo?
  - Lọc sơ bộ dựa trên tỉ lệ chồng lẫn dãy k-ký tự.
    - Tỉ lệ chồng lẫn dãy k-ký tự còn có thể được sử dụng trực tiếp để sửa lỗi viết từ.

# Tỉ lệ chồng lẫn dãy k-ký tự

- Ý tưởng:

- Sử dụng chỉ mục dãy k-ký tự để lấy tất cả từ chỉ mục có chứa bất kỳ dãy k-ký tự nào của truy vấn
- Sử dụng số lượng dãy k-ký tự làm ngưỡng lựa chọn
  - Ý nghĩa: Từ chỉ mục chứa càng nhiều dãy k-ký tự của từ truy vấn thì càng có nhiều khả năng sẽ có khoảng cách soạn thảo nhỏ.
- Thay vì sử dụng số lượng, cũng có thể thiết lập trọng số cho các dãy k-ký tự.

## Ví dụ dãy 3-ký tự

- Các dãy 3-ký tự cho từ november là: nov, ove, vem, emb, mbe, ber
- Các dãy 3-ký tự cho từ december là: dec, ece, cem, emb, mbe, ber
- Số lượng dãy 3-ký tự có trong cả 2 từ = 3
  - Tỷ lệ chồng lấn =  $3 / 9 = 1/3$ .

# Hệ số Jaccard

- Một đại lượng thường được sử dụng để đo độ tương đồng
- Với  $X$  và  $Y$  là hai tập đặc trưng, thì hệ số Jaccard của  $X$  và  $Y$  được định nghĩa là:

$$\text{Jaccard}(X, Y) = |X \cap Y| / |X \cup Y|$$
 hoặc tương đương

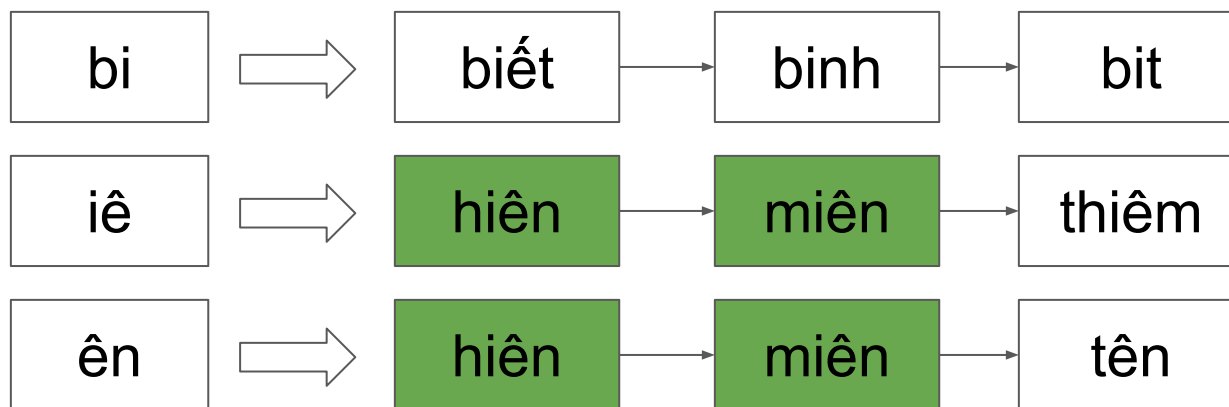
$$\text{Jaccard}(X, Y) = |X \cap Y| / (|X| + |Y| - |X \cap Y|)$$

- Kích thước của  $X$  và  $Y$  có thể khác nhau
- Miền giá trị của hệ số Jaccard là  $[0, 1]$ 
  - $\text{Jaccard}(X, Y) = 1$  nếu  $X = Y$  và
  - $\text{Jaccard}(X, Y) = 0$  nếu  $X$  và  $Y$  không có phần tử chung.

***Ý tưởng:** Sử dụng ngưỡng để chọn các từ tiềm năng, ví dụ, nếu  $\text{Jaccard} > 0.8$  thì thực hiện các xử lý tiếp theo.*

# Lọc từ dựa trên chỉ mục dãy k-ký tự

- Sử dụng  $k = 2$
- Xét từ truy vấn, ví dụ từ “biên”, các dãy kếp là: bi, iê, ên.
- Giả sử chúng ta có các danh sách từ như trong hình vẽ



*Sử dụng giải thuật hợp nhất các danh sách thẻ định vị để đếm số lượng dãy kếp có trong từ chỉ mục.*

*Sau đó tính hệ số Jaccard (hoặc đại lượng khác).*

# Sửa lỗi có ngữ cảnh

- Xét truy vấn nguyên câu "*Tháng ba mua hoa gạo*"
- Chúng ta muốn phản hồi
  - Có phải bạn muốn tìm "*Tháng ba mùa hoa gạo*"?
  - Bởi vì không có văn bản nào phù hợp với câu truy vấn ban đầu
  - => Cần sử dụng các từ lân cận để phát hiện lỗi này



# Sửa lỗi có ngữ cảnh<sub>(2)</sub>

- Ý tưởng 1:

- Đầu tiên tìm tất cả các từ chỉ mục (theo khoảng cách soạn thảo) tương tự tới từ truy vấn
- Thử tất cả các câu bằng cách sửa từng từ truy vấn
  - Có thể có rất nhiều tổ hợp
- Xác định phương án sửa lỗi dựa trên số lượng kết quả cho mỗi trường hợp:
  - Lựa chọn phương án có nhiều kết quả nhất
- Sau đó gợi ý phương án thay thế cho truy vấn ban đầu

- Ý tưởng 2:

- Tách câu thành các dãy 2-từ
- Tìm các dãy 2-từ chỉ cần sửa một phần.

# Bài tập 2.1. Chỉ mục ngược có vị trí

Tạo chỉ mục ngược có vị trí cho bộ dữ liệu trong Bài tập 1.1.

- **Doc1:** [Lập trình Hướng đối tượng với C++]
- **Doc2:** [Xây dựng dịch vụ Web với C++]
- **Doc3:** [Kiến trúc nguyên khối và kiến trúc dịch vụ nhỏ]
- **Doc4:** [Kiến trúc hệ thống thông tin]

## Bài tập 2.2. Khoảng cách soạn thảo

Tính khoảng cách soạn thảo Levenshtein cho 2 từ date và donate

## Bài tập 2.3. Chỉ mục từ xoay

Hãy xác định tập các từ xoay cho bộ từ vựng gồm 3 từ  
exercise, lesson, student.

## Bài tập 2.4. Mẫu từ và chỉ mục dãy kép

Cho mẫu từ  $f_i * m_o * e_r$ . Hãy viết câu truy vấn Boolean tương đương với các dãy  $k$ -ký tự trong trường hợp  $k = 2$ ? Hãy thử tìm một từ thỏa mãn biểu thức Boolean nhưng không thỏa mãn mẫu từ ban đầu?

## Bài tập 2.5. Khoảng cách soạn thảo

Ký hiệu  $LV(s_1, s_2)$  là số lượng thao tác soạn thảo (trong khoảng cách Levenstein) tối thiểu để biến chuỗi  $s_1$  thành chuỗi  $s_2$ .

- a) Hãy chứng minh  $LV(s_1, s_2) = LV(s_2, s_1)$  với 2 chuỗi  $s_1$  và  $s_2$  bất kỳ.
- b) Hãy chứng minh  $LV(s_1, s_2) + LV(s_2, s_3) \geq LV(s_1, s_3)$  với 3 chuỗi  $s_1$ ,  $s_2$ , và  $s_3$  bất kỳ.

