# Chương 1

# BẮT LỖI CHÍNH TẢ TIẾNG VIỆT

## 1.1. TỔNG QUAN BẮT LỖI CHÍNH TẢ TIẾNG VIỆT:

### 1.1.1. Lỗi chính tả:

Trong khi soạn thảo văn bản, người soạn thảo vẫn hay mắc các lỗi chính tả có thể là do gõ nhầm hay do cách phát âm. Từ những nguyên nhân đó, các lỗi chính tả được tạo ra là rất đa dạng. Nhưng nói chung (cũng theo những nghiên cứu gần đây), các lỗi chính tả có thể được xếp vào hai loại sau:

+ Từ bị sai không nằm trong từ điển – chúng tôi sẽ gọi là lỗi âm tiết:

Ví dụ:

- xổng → xỗng (do sai hỏi ngã)
- quỷ → qủy (sai do bỏ dấu)
- phát → pohát (gõ dính phím)
- đọc → đpjc (gõ nhầm ký tự)
- thăng → thưng (sai do gõ thiếu ký tự)
- cũng → ccũng (sai do gõ thừa ký tự)
- thật → thậc (sai do cách phát âm)
- tuyệt → tyuệt (sai do gõ nhầm vị trí ký tự)

+ Từ bị sai nhưng vẫn có trong từ điển – chúng tôi sẽ gọi là lỗi cú pháp:

Ví dụ:

- trái chanh → trái tranh (sai do cách phát âm)
- phiến diện → phiếu diện (sai do gõ nhầm)
- canh cửa → cánh cửa (sai do bỏ dấu)
- phát hoang → phác hoang (sai do phát âm)
- đầu đuôi → đầu đôi (sai do thiếu ký tự)
- thế thần → thết thần (sai do thừa ký tự)

### 1.1.2. Tổng quan phương pháp bắt lỗi chính tả tiếng Việt:

Từ hai loại lỗi đã chỉ ra ở trên, ta thấy rằng việc bắt lỗi chính tả chủ yếu như sau:

- Bắt lỗi các từ không có trong từ điển. (chúng tôi sẽ gọi giai đoạn này là bắt lỗi mức âm tiết.)
- Bắt lỗi các từ có trong từ điển nhưng vẫn sai. (chúng tôi sẽ gọi giai đoạn này là bắt lỗi mức cú pháp.)

Phương pháp bắt lỗi chính tả sẽ bao gồm hai giai đoạn như sau:

- Giai đoạn 1: Kiểm tra toàn bộ các âm tiết trong văn bản xem có hợp lệ không. Nếu đã hợp lệ thì sang bước hai, còn chưa thì gợi ý sửa lỗi và thay thế nếu muốn.

Để kiểm tra một âm tiết có hợp lệ hay không, ta có thể sử dụng một trong hai cách: hoặc là dựa trên nguyên tắc cấu tạo từ, hoặc là dựa trên từ điển. Phương pháp thứ hai dẫn đến việc phải lưu trữ nhiều từ không có nghĩa. Cả hai phương pháp này sẽ được bàn đến ở mục 1.4.

- Giai đoạn 2: Kiểm tra xem kết cấu câu có hợp lệ không; tức là, kiểm tra cú pháp câu có đúng với luật sinh quy định không. Ở giai đoạn này, câu được phân tích thành các chuỗi từ loại. Các chuỗi từ loại này được đưa qua bộ phân tích cú pháp. Bộ phân tích sẽ trả lại chuỗi từ loại đúng (tức là câu đúng) hoặc đưa ra vị trí từ loại nghi sai. Dựa trên vị trí từ loại nghi sai, ta có thể thực hiện gợi ý và sửa lỗi.

Trong giai đoạn này, bộ phân tích phải có giải thuật thích hợp cho phân tích cú pháp ngôn ngữ tự nhiên. Hiện nay, có bốn giải thuật phân tích cú pháp phù hợp là: thuật toán phân tích Top-Down, thuật toán phân tích Bottom-Up, thuật toán Cocke-Younger-Kasami (CYK) và thuật toán Earley. Trong khoá luận, chúng tôi sử dụng thuật toán Earley để làm bộ phân tích cú pháp. Vì giải thuật này dựa trên phương pháp lập bảng, tuy là phức tạp nhưng độ phức tạp của thuật toán lại nhỏ, $O(n^3)$. Giải thuật Earley và những cải tiến cho giải thuật sẽ được nói đến trong mục 1.5.

Giai đoạn này có bắt lỗi tốt hay không còn phụ thuộc vào việc chia nhỏ từ loại và lập luật sinh. Những điều này sẽ được nói đến trong Chương 2.

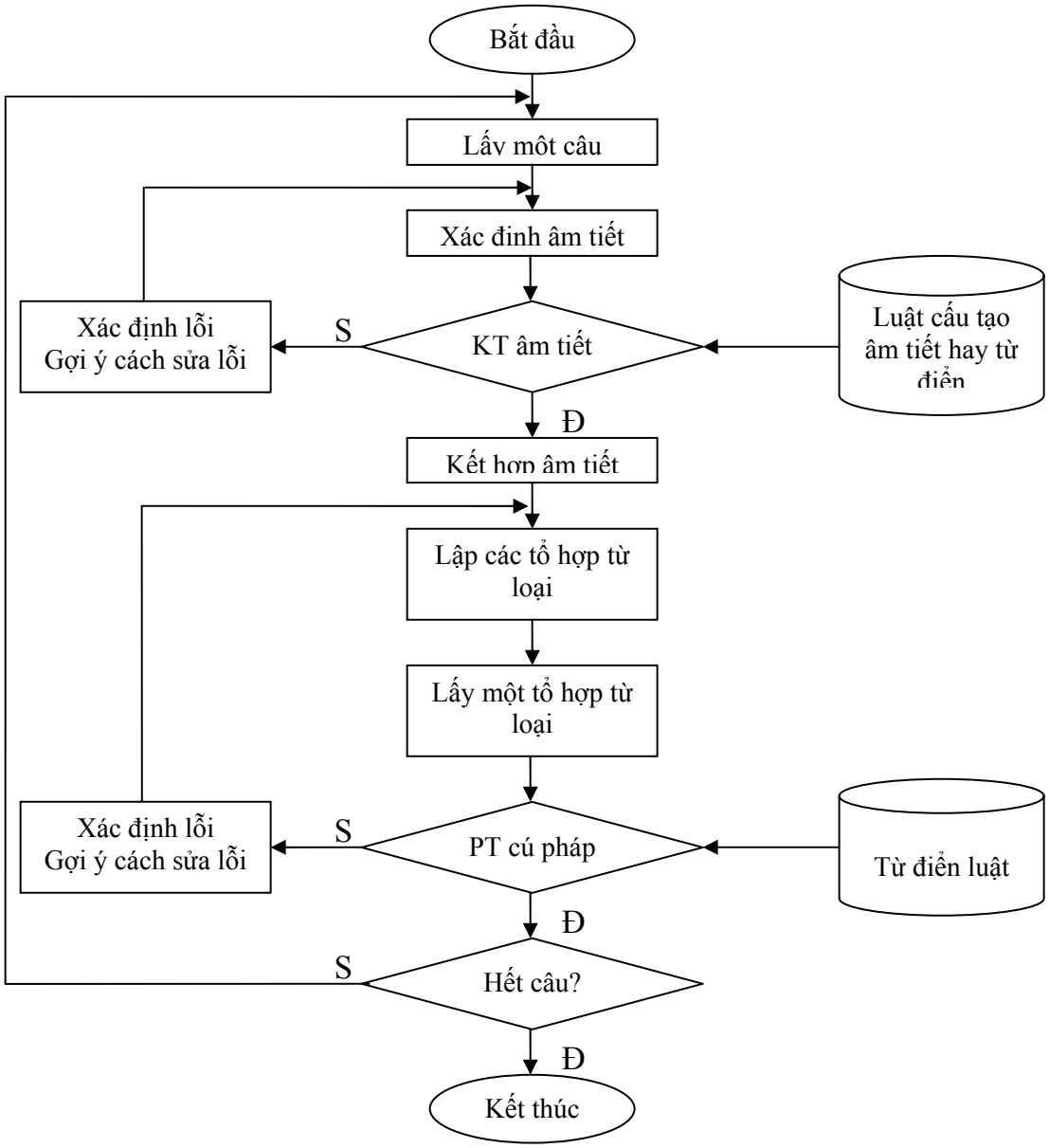
Phương pháp nói ở trên triệt để hơn rất nhiều, nó bắt lỗi được vừa lỗi âm tiết vừa lỗi cú pháp.

Nhưng do sự tinh tế của ngôn ngữ tiếng Việt, một số lỗi vẫn bị phương pháp này bỏ qua. Ví dụ, ta có cụm từ: "chính quyền lực này", nhưng bị viết sai thành: "chính quyền lúc này". Ở đây, bộ phân tích cú pháp vẫn báo đúng cho cụm từ sai ở trên. Có điều đó là do cụm từ sai này vẫn có chuỗi từ loại hợp với một luật sinh có sẵn và nó sẽ lọt qua trong phân tích cú pháp. Lớp lỗi này chỉ có thể bắt được thông qua ngữ nghĩa của câu, đoạn và văn bản mà thôi.

Trên thực tế, các lỗi loại này thường xuất hiện không nhiều trong tiếng Việt nên phương pháp trên vẫn có hiệu quả cao. Theo [8], thì phương pháp này có thể bắt lỗi đạt tới 95% tỷ lệ lỗi.

## 1.2. QUY TRÌNH BẮT LỖI CHÍNH TẢ:

Dựa trên tổng quan về phương pháp bắt lỗi chính tả tiếng Việt, chúng tôi đưa ra một qui trình bắt lỗi chính tả được trình bày bằng lưu đồ như sau:




## 1.3. TỔ CHỨC TỪ ĐIỂN:

Để hỗ trợ cho chương trình bắt lỗi chính tả hoạt động được tốt thì cần tổ chức một từ điển chính tả tiếng Việt chuẩn. Từ điển sẽ lưu trữ các từ và từ loại của từ. Từ điển phải đáp ứng các yêu cầu sau:

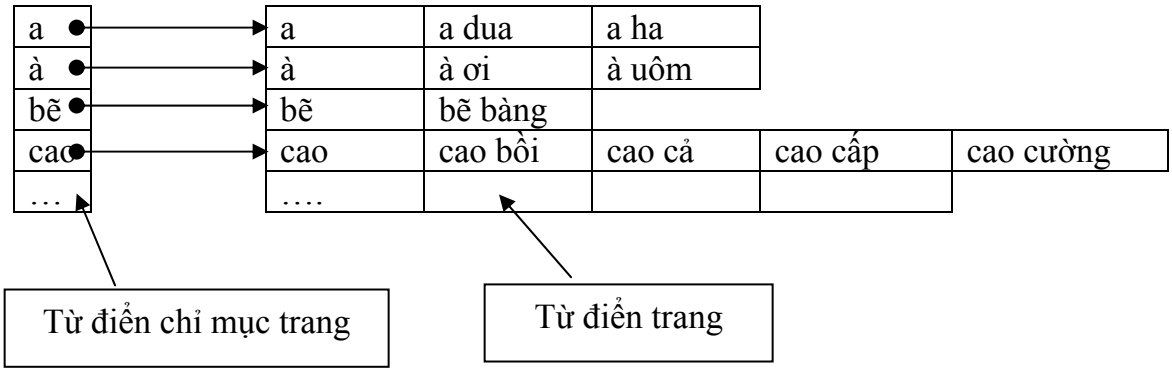
- Kích thước nhỏ, nhưng đầy đủ thông tin cần thiết.
- Truy xuất nhanh trong các tác vụ tìm kiếm, thêm, bớt.

Trong bắt lỗi chính tả, thao tác tìm kiếm từ trong từ điển là thao tác quan trọng nhất, nó ảnh hưởng đến tốc độ của giải thuật bắt lỗi. Do đó, từ điển sẽ được tổ chức theo dạng bảng băm có sắp xếp thứ tự và áp dụng thuật toán tìm kiếm nhị phân cho thao tác tìm kiếm trong từ điển. Cấu trúc từ điển sẽ như sau:

Từ điển được tổ chức thành nhiều trang. Các từ thuộc cùng một trang nếu có âm tiết đầu giống nhau. Nói một cách khác, từ điển sẽ được tổ chức thành hai danh sách:

- Một danh sách (từ điển chỉ mục trang) dùng để lưu chỉ mục, được sắp tăng dần. Danh sách này sẽ lưu các âm tiết đầu của từ.
- Một mảng các danh sách (từ điển trang) lưu nội dung các trang và cũng được sắp xếp tăng dần

Minh hoa từ điển:




Như vậy, để tìm một từ trong từ điển, đầu tiên ta tìm xem âm tiết đầu tiên của từ đó có trong từ điển chỉ mục trang hay không. Nếu có tức là từ đó có thể có hoặc không có trong từ điển trang. Tiếp theo ta tìm kiếm nhị phân trong từ điển trang tại trang đã so khớp xem có từ cần tìm không.

Với cách tổ chức từ điển như trên, số lượng thành phần trong từ điển chỉ mục trang chính là số lượng âm tiết trong tiếng Việt, tức là khoảng hơn 5000 âm tiết. Còn số lượng từ trong mỗi trang nhiều nhất là 100 từ. Như thế, với thuật toán tìm kiếm như trên, ta có thời gian tìm kiếm một từ là: $\log_2 5000.\log_2 100$, tức xấp xỉ 86 đơn vị thời gian. Với cách xây dựng từ điển và thuật toán tìm kiếm như thế ta được số lần tìm kiếm nhỏ hơn nhiều so với kích thước từ điển.

## 1.4. BẮT LỖI CHÍNH TẢ MỨC ÂM TIẾT:

Để bắt lỗi chính tả mức âm tiết, ta có thể sử dụng một trong hai phương pháp sau đây:

- Bắt lỗi dựa trên nguyên lý cấu tạo âm tiết
- Bắt lỗi dựa trên từ điển

### 1.4.1. Bắt lỗi chính tả mức âm tiết dựa trên nguyên lý cấu tạo âm tiết:

Theo nguyên lý cấu tạo âm tiết, một âm tiết bao gồm ba thành phần: phụ âm đầu, nguyên âm và phụ âm cuối. Trong đó, thành phần nguyên âm là thành phần trung tâm kết hợp với hai thành phần còn lại. Thành phần nguyên âm này luôn có mặt trong các âm tiết tiếng Việt. Hai thành phần còn lại có thể có hoặc không.
Ví dụ:

- Đầy đủ ba thành phần: ruộng, vườn,…
- Không có phụ âm đầu: anh, em,….

- Không có phụ âm cuối: báo, chí,…
- Không có phụ âm đầu và cả phụ âm cuối: áo, eo,…

Như vậy, ta có thể tổ chức một cấu trúc lấy nguyên âm làm trung tâm, còn hai thành phần còn lại là tổ hợp của nhiều thành phần phụ có thể kết hợp với nguyên âm đó.

Trong ba thành phần của âm tiết thì hai thành phần phụ là có số lượng ít hơn cả và dễ quản lý nhất. Đối với thành phần phụ âm đầu, ta có tất cả là 26 phụ âm đầu: b, c, ch, d, đ, g, gh, h, k, kh, l, m, n, ng, ngh, nh, p, ph, q, r, s, t, th, tr, v, x. Do có trường hợp âm tiết không có phụ âm đầu nên ta phải quản lý cả trường hợp này. Trong lưu trữ, ta có thể quản lý bằng 27 bit dữ liệu, mỗi bit tương ứng với một phụ âm đầu ở trên.

Đối với thành phần phụ âm cuối, ta có tất cả 8 phụ âm cuối: c, ch, m, n, ng, nh, p, t. Cũng như trên, vẫn có âm tiết không có phụ âm cuối nên ta cũng phải quản lý trường hợp này. Trong lưu trữ, ta có thể quản lý phụ âm cuối bằng 9 bit tương ứng với từng phụ âm cuối.

Đối với nguyên âm, thành phần này rất phức tạp vì có cả dấu nên không thể lập mã như các thành phần kia. Riêng với thành phần này ta có thể giữ nguyên dạng chuỗi để lưu trữ.

Với cách mã hoá như trên, ta có thể lưu trữ theo một cấu trúc như sau:

- Thành phần phụ âm đầu, kiểu long
- Nguyên âm, kiểu string có độ dài 3
- Thành phần phụ âm cuối, kiểu integer

Với cấu trúc như trên, nhiều âm tiết sẽ được lưu trữ cùng một record. Đây là một cách để tiết kiệm không gian lưu trữ. Các record sẽ được lưu trữ vào một mảng được sắp xếp theo nguyên âm.

Sau đây là một ví dụ cho việc lưu trữ theo cấu trúc trên.

Ví dụ: Nguyên âm a có thể kết hợp với các phụ âm đầu sau: b, c và ch.

Nguyên âm a có thể kết hợp với các phụ âm cuối sau: n, m và nh.

Như thế, có thể tạo ra các âm tiết:

ban bam banh can cam canh chan cham chanh

Qua phân tích ta sẽ tạo thành một cấu trúc như sau:

- Nguyên âm: a
- Phụ âm đầu: 11100000000000000000000000 (viết dưới dạng nhị phân cho dễ thấy các bit nào được bật, với ví dụ này là các bit 1, 2, 3)
- Phụ âm cuối: 001101000 (với ví dụ này là các bit 3, 4, 6 được bật).

Như thế, để kiểm tra chính tả một âm tiết, ta chỉ cần tìm đến record có cùng thành phần nguyên âm. Sau đó, sử dụng phép toán AND để kiểm tra cho hai thành phần phụ.

**Thuật toán kiểm tra chính tả của âm tiết:**

1. Với một âm tiết vào, ta tách âm tiết ra làm ba thành phần.
2. Tìm kiếm nhị phân cho nguyên âm.

3. Nếu tồn tại record có cùng nguyên âm:
   i. Dùng phép toán AND để kiểm tra hai thành phần phụ;
   ii. Nếu cả hai đều khác 0, kết luận âm tiết đúng;
   iii. Ngược lại, kết luận âm tiết sai;
4. Nếu không tồn tại, kết luận âm tiết sai.

Với cách lưu trữ và thuật toán như trên, trong từ điển ta phải lưu trữ tất cả các quy tắc kết hợp của các âm tiết tiếng Việt mới đảm bảo hiệu quả của thuật toán.

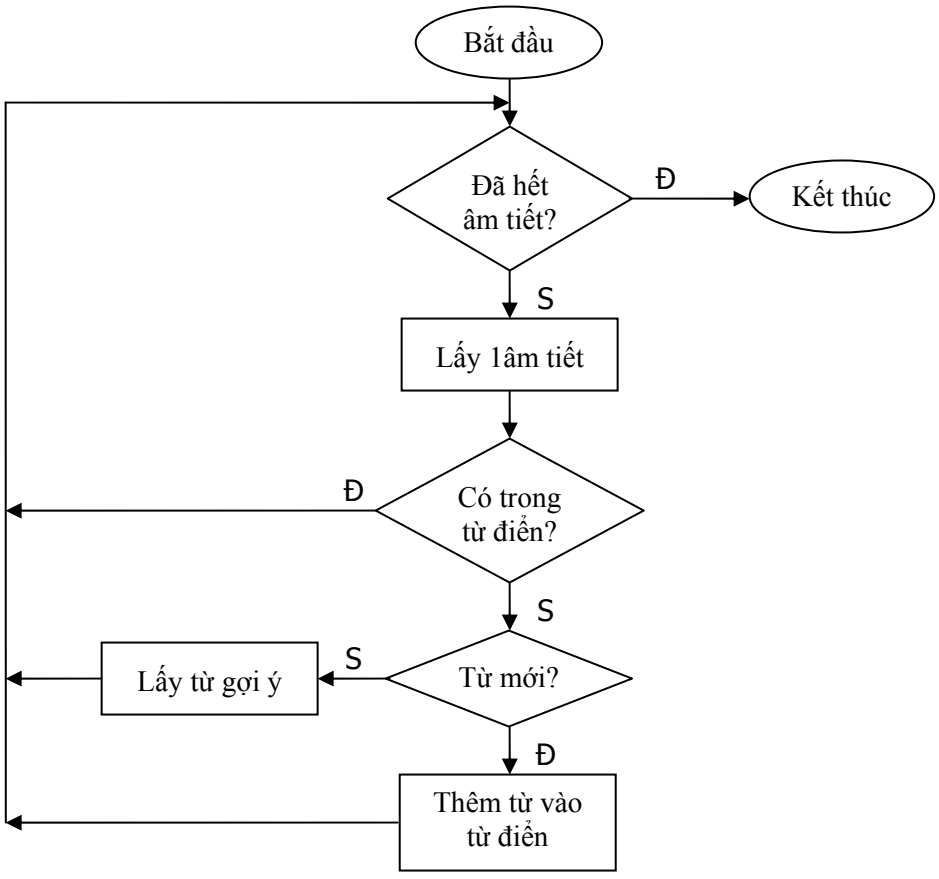
Tuy có nhiều ưu điểm trong lưu trữ và tốc độ nhưng phương pháp này chỉ phù hợp cho các âm tiết thuần Việt, còn các âm tiết được Việt hoá như: kilômét, ôxi… thì không quản lý được. Đây là một hạn chế do nguyên tắc cấu tạo của âm tiết tiếng Việt không xét đến những trường hợp này. Phương pháp này vẫn chưa hoàn chỉnh lắm.

Do hạn chế trên, khoá luận sẽ không sử dụng phương pháp này cho giai đoạn bắt lỗi chính tả mức âm tiết. Nhưng để minh hoạ phương pháp, chúng tôi cũng đã cài đặt phương pháp, phần cài đặt này sẽ được nói đến trong mục 3.1.2.

### 1.4.2. Bắt lỗi chính tả mức âm tiết dựa trên từ điển:

Phương pháp này sử dụng một từ điển để lưu trữ tất cả các âm tiết của tiếng Việt có thể có. Như thế, để kết luận một âm tiết là đúng chỉ cần kiểm tra xem âm tiết đó có trong từ điển hay không.

**Thuật toán bắt lỗi chính tả mức âm tiết dựa trên từ điển:**



Lợi dụng cấu trúc từ điển đã trình bày ở mục 1.3, chúng ta có thể kiểm tra các âm tiết có trong từ điển chỉ mục hay không. Như thế, từ điển phải lưu trữ thêm một lượng âm tiết không có nghĩa. Theo [8] lượng âm tiết này khoảng 1100. Cộng thêm số âm tiết của các từ có nghĩa thì lên đến khoảng 5000. (Trong thực tế cài đặt thì số lượng âm tiết cả có nghĩa và không có nghĩa lên đến hơn 5000.)

Như vậy, từ điển chỉ mục trang sẽ lưu trữ toàn bộ các âm tiết của tiếng Việt. Đồng thời, ta sử dụng thuật toán tìm kiếm nhị phân để tìm kiếm thì số phép toán cần để tìm một từ là $\log_2 5000 \approx 13$. Một con số khá nhỏ so với số lượng âm tiết trong từ điển.

Phương pháp này tuy tốn nhiều không gian lưu trữ hơn, nhưng lại giải quyết tất cả các kiểu âm tiết trong tiếng Việt. Ngoài ra, tốc độ tìm kiếm của phương pháp là rất nhanh. Chúng tôi sử dụng phương pháp này cho bắt lỗi chính tả mức âm tiết.

### 1.4.3. Phương pháp tạo gợi ý sửa lỗi mức âm tiết:

Khi bắt được một âm tiết sai nào đó, chương trình bắt lỗi cần phải đưa ra được những âm tiết đúng để gợi ý cho người sử dụng sửa lỗi.

Trong tiếng Việt, âm tiết dài nhất là "nghiêng" với bảy ký tự. Giả sử người gõ văn bản có thể gõ sai hai ký tự trên một âm tiết. Khi đó, độ dài âm tiết sai có thể lên đến chín ký tự.

Với quy ước số lượng ký tự có thể gõ sai là hai ký tự như trên, ta có thể định nghĩa, hai ký tự là "trùng" nhau nếu vị trí của chúng trong âm tiết chỉ chênh nhau nhiều nhất hai ký tự. Đồng thời, ta cũng có thể định nghĩa hai âm tiết là "so khớp" khi có ít nhất hai ký tự không "trùng".

Ví dụ:

Âm tiết "anh" và âm tiết "hai" có ký tự "a" trong hai âm tiết là trùng nhau.

Âm tiết "anh" và âm tiết "hanh" là hai âm tiết so khớp vì nó có ba âm tiết trùng nhau.

Qua thực tế nghiên cứu, định nghĩa trên không phù hợp cho các âm tiết có độ dài ký tự nhỏ hơn 4. Vì nó làm cho xuất hiện quá nhiều âm tiết hoàn toàn không phù hợp để gợi ý, do đó trong cài đặt, ta nên có những điều chỉnh thích hợp.

Ví dụ:

Ta muốn gõ âm tiết "ai" nhưng lại gõ sai thành "a8". Bộ bắt lỗi chính tả mức âm tiết sẽ bắt lỗi âm tiết "a8" và cho ra các âm tiết gợi ý như sau: a, ai, am, an, ai, au, ba, ca, da, ga, ha, la, ma, na, ra, sa, ta, va, xa, ùa, úa, đa, ưa, ỉa, ứa. (Phần gợi ý này đã có điều chỉnh cho các âm tiết nhỏ hơn 4 ký tự.)

**Thuật toán tạo gợi ý sửa lỗi mức âm tiết:**

1. Lấy từng âm tiết đúng có trong từ điển.
2. Nếu độ chênh lệch về độ dài âm tiết đúng với độ dài âm tiết cần sửa lỗi là nhỏ hơn hoặc bằng 2:
   i. Duyệt qua từng ký tự trong âm tiết đúng xem có trong âm tiết sai không;
   ii. Nếu không có, rõ ràng hai âm tiết không "so khớp", thoát khỏi vòng lặp;

iii. Ngược lại, xem hai ký tự có "trùng" vị trí không;
iv. Nếu có, tăng biến đếm ký tự "trùng" lên một;
v. Tiếp tục với ký tự tiếp theo;
vi. Ngược lại, thoát khỏi vòng lặp;
vii. Nếu số lượng ký tự "trùng" nhỏ hơn 2 so với độ dài âm tiết đúng thì đưa âm tiết đúng vào danh sách gợi ý;
viii. Ngược lại, tiếp tục vòng lặp ngoài;

3. Ngược lại, lấy âm tiết tiếp theo cho đến hết từ điển.

Thuật toán trên phụ thuộc vào cách tổ chức các từ đơn âm tiết trong từ điển. Theo cách tổ chức từ điển được trình bày ở mục 1.3, ta có các từ đơn âm tiết ở từ điển chỉ mục trang. Khoá luận sẽ thực hiện lấy các âm tiết gợi ý từ từ điển này. Thuật toán này có độ phức tạp $O(n^2)$.

Dù đã có cải tiến cho các âm tiết có độ dài nhỏ hơn bốn âm tiết, thuật toán trên vẫn còn đưa ra nhiều từ không phù hợp. Phương pháp này chưa có tính thông minh cao để chỉ đưa ra các âm tiết hợp lý nhất.

## 1.5. GIẢI THUẬT EARLEY CHO PHÂN TÍCH CÚ PHÁP:

Giải thuật Earley là một trong số các giải thuật được sử dụng để phân tích cú pháp trong xử lý ngôn ngữ tự nhiên. Đây là một giải thuật tổng quát, có thể phân tích bất kỳ văn phạm phi ngữ cảnh nào. Khoá luận sẽ sử dụng giải thuật Earley như bộ phân tích cú pháp trong giai đoạn bắt lỗi mức cú pháp.

### 1.5.1. Giải thuật Earley cơ bản:

Cho G=(V, W, S, P) là một văn phạm phi ngữ cảnh và $w=a_1 \ldots a_n \in V^*$. Khi đó, $A \rightarrow \alpha \bullet \beta$ là một luật có chấm khi $A \rightarrow \alpha\beta \in P$. Trong đó, luật có chấm giống như luật sinh bình thường nhưng có thêm một dấu chấm bên trong luật, thể hiện vị trí đang được phân tích trong luật đó. Giải thuật Earley được biểu diễn thông qua việc xây dựng bảng chứa tập các luật có chấm. Người ta xây dựng bảng Earley với các cột $I_i$ (i=0..n), cột $I_0$ nhận giá trị khởi tạo, n là độ dài của chuỗi từ loại nhập. Mỗi ô sẽ có các giá trị: ***giá trị gốc*** để biết luật đó phát sinh từ cột nào và ***luật có chấm***.

Ví dụ: Giá trị gốc Luật có chấm

0 $S \rightarrow \bullet$CN VN
1 $VN \rightarrow$ĐT$\bullet$ DT

#### 1.5.1.1. Giải thuật:

Giải thuật bao gồm ba bước:

(1) Đoán nhận (Predict): Tại cột $I_i$:

Đối với các luật có ký tự không kết thúc ở bên phải dấu chấm, ta thêm các luật mới mà ký tự không kết thúc đó là vế trái của các luật. Giá trị gốc là i. Điều này có nghĩa là, với mỗi $[A \rightarrow \alpha \bullet B\beta, j]$ trong $I_i$, ta thêm $[B \rightarrow \bullet\gamma, i]$ vào $I_i$ nếu $B \rightarrow \gamma \in P$.

(2) Duyệt (Scan): Tại cột $I_i$:

Đối với các luật mà ký tự kết thúc ở bên phải dấu chấm, luật này sẽ được chuyển sang cột $I_{i+1}$ với dấu chấm được dịch ra sau ký tự kết thúc.

Tức là, với $[A \rightarrow \alpha \bullet a\beta, j]$ sẽ được đổi thành $[A \rightarrow \alpha a \bullet \beta, i]$ trong cột $I_{i+1}$.

(3) Hoàn thiện (Complete): Tại cột $I_i$:

Khi có luật [A→α•, j] thì sao chép và đổi [B→α•Aβ, k] trong cột $I_j$ thành [B→αA•β, k] trong cột $I_i$.

#### 1.5.1.2. Nhận xét:

- Đây là dạng phân tích từ trên xuống bởi vì ta bắt đầu với việc đoán nhận. Nếu ta thay đổi thứ tự trên, chúng ta sẽ có kiểu phân tích từ dưới lên.
- Thông thường, phân tích từ trên xuống có vấn đề với đệ qui trái, nhưng thuật toán Earley đã giải quyết bằng cách:

Mỗi luật giống nhau sẽ chỉ xuất hiện một lần trong mỗi cột. Có nghĩa là trong các bước thực hiện thuật toán, trước khi thêm một luật vào bảng thì phải kiểm tra xem nó có trùng với luật nào đã có trong cột cần thêm vào không. Nếu không thì thêm vào, còn có thì không thêm vào.

- Chuỗi từ loại là sai cú pháp khi ta đã duyệt qua hết các luật trong $I_i$ mà $I_{i+1}$ rỗng và chưa thể kết thúc bảng hợp lệ.
- Chuỗi từ loại là đúng cú pháp khi kết thúc chuỗi từ loại mà ta có luật khởi tạo tại cột cuối cùng.

Nói chung, chuỗi đúng khi tại điểm kết thúc chuỗi nhập, mà dấu chấm đã di chuyển ra sau ký tự bắt đầu S.

- Với việc sử dụng giá trị đoán nhận trước, có thể giúp ta tránh dư thừa.

Ví dụ, ta có luật VN → VN •BN tại vị trí kết thúc chuỗi nhập. Thông thường, ta sẽ đi đoán nhận BN, nhưng trong trường hợp này là không nên, ta chỉ nên làm như thế nếu còn từ trong chuỗi nhập.

Mặt khác, giá trị đoán nhận trước cũng gây ra sự phức tạp, và tăng số lượng luật được lưu trữ.

- Độ phức tạp của thuật toán là $O(n^3)$, với n là độ dài chuỗi nhập (bằng số lượng cột của bảng giảm đi một).

### 1.5.2. Những cải tiến cho giải thuật Earley:

Giải thuật Earley vẫn có những hạn chế nhất định trong xử lý ngôn ngữ tự nhiên. Sau đây là những cải tiến nhỏ để cải thiện tốc độ cho giải thuật Earley.

#### 1.5.2.1. Giải quyết vấn đề luật dư thừa trong giải thuật Earley:

Với giải thuật đã nêu ở trên, ta có thể nhận thấy có rất nhiều luật dư thừa vẫn được lưu trữ trong bảng Earley. Như thế đồng nghĩa với việc phải duyệt qua quá nhiều luật dư thừa như Kilbury đã nhận xét [5]. Qua nghiên cứu, chúng tôi nhận thấy các luật dư thừa có dạng như sau:

- Thứ nhất, luật chỉ có một kí tự kết thúc ở vế trái mà không khớp với giá trị đoán nhận;
- Thứ hai, luật không dẫn đến đệ qui trái.

##### 1.5.2.1.1. Dạng luật sinh:

Để giải quyết triệt để vấn đề này, khoá luận đã thực hiện lập luật sinh với dạng riêng. Tất cả các luật sinh đều thuộc vào một trong hai dạng sau:

- $A \rightarrow \alpha$, $\alpha \in W^*$
- $A \rightarrow a$, $a \in V$

Dạng luật trên vẫn phù hợp với văn phạm phi ngữ cảnh.

Với các luật sinh thuộc vào hai dạng trên thì các luật dư thừa sẽ là:

- Luật có vế phải chỉ là một ký tự kết thúc ($A \rightarrow a$);
- Luật có vế phải là các ký tự không kết thúc mà không dẫn đến đệ qui trái ($A \rightarrow B\alpha$).

#### 1.5.2.1.2. Giải thuật cải tiến:

Với dạng luật như trên, giải thuật được cải tiến chỉ còn lại hai bước như sau:

(1) Đoán nhận:

Với mỗi $[A \rightarrow \alpha \bullet B\beta, j]$ trong $I_i$.

Lấy các luật trong từ điển luật sinh.

Duyệt qua các luật dạng $B \rightarrow a$, nếu khớp với giá trị đoán nhận thì đưa luật khớp cùng với các luật dạng $B \rightarrow B\alpha$ vào bảng Earley.

Ngược lại, nếu không so khớp với giá trị đoán nhận thì đưa toàn bộ các luật dạng $B \rightarrow \alpha$ vào bảng Earley.

(2) Hoàn thiện:

Như giải thuật Earley cũ.

Giải thuật cải tiến ở trên chỉ còn hai bước, bước quét trong giải thuật cũ đã được bỏ đi là do dạng luật sinh và giai đoạn đoán nhận mới đã giải quyết luôn bước này. Giải thuật cải tiến đã giải quyết được vấn đề luật dư thừa, nó không còn phải duyệt qua các luật không cần thiết trong phân tích cú pháp nữa. Như thế, sẽ cải thiện tốc độ của tiến trình xử lý nhiều hơn. Ngoài ra, còn giảm không gian lưu trữ. Nhưng giải thuật cải tiến vẫn có độ phức tạp thời gian là $O(n^3)$. Vì giải thuật chỉ mới thay đổi nội dung bên trong cấu trúc của giải thuật Earley chứ chưa thay đổi được cấu trúc của giải thuật nên độ phức tạp thời gian vẫn là như cũ.

### 1.5.2.2. Giải quyết vấn đề bùng nổ tổ hợp:

Hiện tượng bùng nổ tổ hợp xảy ra do một từ có thể thuộc vào nhiều từ loại khác nhau. Nhưng một đặc điểm dễ nhận thấy của các tổ hợp từ loại được sinh ra từ một câu là luôn có những đoạn con từ loại giống nhau trên các tổ hợp từ loại.

Ví dụ: (Ví dụ được lấy trong [10])

Đoạn câu cần phân tích: trong biên chế và hưởng lương từ ngân sách.

Có 12 chuỗi từ loại được xuất ra như sau:

A11 N22 L10 V43 N23 F10 N23 N50
A11 N22 L10 V43 N23 F11 N23 N50
A11 V40 L10 V43 N23 F10 N23 N50
A11 V40 L10 V43 N23 F11 N23 N50
F10 N22 L10 V43 N23 F10 N23 N50

F10 N22 L10 V43 N23 F11 N23 N50
F10 V40 L10 V43 N23 F23 N23 N50
F10 V40 L10 V43 N23 F10 N23 N50
F11 N22 L10 V43 N23 F10 N23 N50
F11 N22 L10 V43 N23 F11 N23 N50
F10 V40 L10 V43 N23 F10 N23 N50
F10 V40 L10 V43 N23 F11 N23 N50

Dựa vào đặc điểm này TS. Phan Thị Tươi trong [10] đã nêu một phương pháp để tăng tốc độ phân tích cú pháp như sau (trích từ [10]):

Nếu bộ phân tích thất bại khi đang kiểm tra một chuỗi, thì nó sẽ so trùng các chuỗi còn lại với đoạn vừa kiểm tra thành công và sẽ tiếp tục quy trình phân tích ở vị trí của một chuỗi khác có chuỗi con dài nhất trùng với đoạn đã phân tích. Quá trình này được lặp lại cho tới khi bộ phân tích duyệt qua hết một chuỗi nào đó. Lúc đó, câu nhập được xác nhận là đúng cú pháp. Ngược lại khi đi đến chuỗi cuối cùng mà vẫn không phân tích thành công thì bộ phân tích sẽ kết luận rằng câu nhập vào không đúng cú pháp.

Đây là một phương pháp hay, nó giúp ta tránh phải phân tích lại những gì đã phân tích rồi. Nhưng phương pháp trên lại chỉ hiệu quả trong trường hợp câu nhập vào là đúng, còn trong trường hợp câu nhập vào là sai thì không hiệu quả.

Qua nghiên cứu, chúng tôi còn nhận thấy còn có hiện tượng trùng một phần (chỉ một phần ngắn của đoạn đã kiểm tra thành công) bên cạnh hiện tượng trùng cả đoạn như đã nói ở trên. Điều này cho thấy ta có thể giảm thêm được một số bước phân tích nữa.

Thừa kế từ phương pháp trong [10] và bổ sung điều mới phát hiện, khoá luận đưa ra phương pháp như sau:

Ta sắp xếp các chuỗi tổ hợp từ loại theo thứ tự. Như thế, chuỗi ngay sau chuỗi đang xét sẽ là chuỗi có khả năng trùng đoạn đã phân tích. Khi kiểm tra một chuỗi thất bại ta chỉ việc so khớp đoạn vừa kiểm tra thành công với chuỗi ngay sau nó và lấy số từ loại so khớp liên tục bắt đầu từ đầu chuỗi. Việc phân tích sẽ được thực hiện tiếp với chuỗi ngay sau tại vị trí đầu tiên không so khớp.

Bước đầu tiên của giải thuật trên là sắp xếp các chuỗi tổ hợp từ loại, chúng tôi thực hiện bước này là để có thể sử dụng phương pháp trong [10]. Khi đã được sắp xếp, rõ ràng chuỗi từ loại ngay sau chuỗi vừa kiểm tra có xác suất trùng đoạn vừa kiểm tra thành công là lớn nhất. Như thế, ta chỉ việc thực hiện tiếp tục phân tích với chuỗi từ loại tiếp ngay sau. Giải thuật cải tiến đã thừa kế trọn vẹn phương pháp trong [10], đồng thời còn thay thế công đoạn tìm kiếm chuỗi trùng khớp bằng chỉ một bước đơn giản là tăng giá trị duyệt tổ hợp chuỗi từ loại lên một. Đây là một cải tiến đáng kể.

## 1.6. BẮT LỖI CHÍNH TẢ MỨC CÚ PHÁP:

Bắt lỗi chính tả mức cú pháp dựa trên việc phân tích cú pháp câu. Do đó, ta phải có các chuỗi tổ hợp từ loại để phân tích. Một bước quan trọng và là giai đoạn tiền xử lý của giai đoạn phân tích cú pháp là phân tích từ. Đây là giai đoạn quan trọng, nó tách các từ, lấy tổ hợp các câu có thể có và lấy tổ hợp ghép từ loại.

### 1.6.1. Phân tích từ:

Từ là đơn vị có sẵn của ngôn ngữ, là cái ngầm định và mọi người coi sự hiện diện của từ là tất nhiên. Hầu như không có quy tắc cấu tạo đối với từ. Do vậy, ta phải so sánh các từ tách được từ văn bản với các từ thực tế để kiểm nghiệm từ đó có phải là từ không.

Một vấn đề còn gây nhiều tranh luận trong việc phân tích từ là chọn độ dài tối đa của từ ghép. Trong tiếng Việt, ngoài các từ đơn, ta còn có các từ ghép hai âm tiết, ba âm tiết,… Nếu ta chọn giới hạn độ dài một từ quá lớn sẽ làm mất nhiều thời gian kiểm tra các từ không cần thiết; còn nếu chọn ngưỡng quá nhỏ, ta sẽ mất từ. Khoá luận sẽ chọn ngưỡng là 3 vì đối với các văn bản pháp quy hành chính thì từ dài nhất là 3 âm tiết.

Ta thực hiện tính số các khả năng ghép nối từ, từ loại:

- **Một câu có n âm tiết và ngưỡng ghép từ là 3 thì ta sẽ có tối đa 3n-3 từ khác nhau.**

Xét câu $a_1\ a_2\ a_3 \dots a_n$,
Các từ có thể xuất phát từ từ $a_1$ là $a_1$, $a_1\ a_2$, $a_1\ a_2\ a_3$ (3 từ),
Các từ có thể xuất phát từ từ $a_2$ là $a_2$, $a_2\ a_3$, $a_2\ a_3\ a_4$ (3 từ),
…
Các từ có thể xuất phát từ từ $a_{n-2}$ là $a_{n-2}$, $a_{n-2}\ a_{n-1}$, $a_{n-2}\ a_{n-1}\ a_n$ (3 từ),
Các từ có thể xuất phát từ từ $a_{n-1}$ là $a_{n-1}$, $a_{n-1}\ a_n$ (2 từ),
Các từ có thể xuất phát từ từ $a_n$ là $a_n$ (1 từ),
Như vậy tổng số từ là: 3(n-2)+2+1 = 3n-3n từ.

- **Một câu có n âm tiết và ngưỡng ghép từ là 3 thì ta sẽ có $2^{n-1}$-2n+7 cách tách từ khác nhau (n>3).**

Xét câu $s=a_1\ a_2\ a_3 \dots a_n$,
Với câu có 1 âm tiết, số cách chia là: $s=a_1$ - 1 cách = $2^0$,
Với câu có 2 âm tiết, số cách chia là: $s=a_1a_2$; $s=a_1\ a_2$ - 2 cách = $2^1$,
Với câu có 3 âm tiết, số cách chia là:
$s=a_1a_2a_3$; $s=a_1a_2\ a_3$; $s=a_1\ a_2a_3$; $s=a_1\ a_2\ a_3$ - 4 cách = $2^2$,
Với câu có 4 âm tiết, số cách chia là:
$s=a_1a_2a_3\ a_4$; $s=a_1a_2\ a_3a_4$; $s=a_1a_2\ a_3\ a_4$; $s=a_1\ a_2a_3a_4$;
$s=a_1\ a_2a_3\ a_4$; $s=a_1\ a_2\ a_3a_4$; $s=a_1\ a_2\ a_3\ a_4$ - 7 cách = $2^3$-1,
Tương tự ta có với câu 5 âm tiết, số cách chia sẽ là: 13 cách = $2^4$-3.
Như vậy, theo qui nạp toán học, ta có thể kết luận là:
+ Với n<=3, ta có $2^{n-1}$ cách tách từ.
+ Với n>3, ta có $2^{n-1}$-2n+7 cách tách từ.

- **Một câu có n âm tiết, ngưỡng ghép từ là 3, mỗi từ có tối đa k kiểu từ loại, ta sẽ có tối đa $k(k+1)^{n-1}-(2n^2-13n+21)(k+1)$ tổ hợp từ loại.**

Câu có một âm tiết sẽ có k tổ hợp từ loại.

Dễ dàng nhận thấy, cứ bổ sung thêm một âm tiết (với ngưỡng tách không giới hạn) thì số tổ hợp sẽ tăng lên k+1 lần. Còn đối với ngưỡng tách là 3 thì số tổ hợp sẽ tăng lên theo hai trường hợp:

- Với n<=3, số tổ hợp sẽ tăng lên k+1 lần.
- Nhưng với n>3, số tổ hợp sẽ tăng lên $[2^{n-1} – (2(n-4)+1)](k+1)$ lần.

Như vậy, số tổ hợp từ loại sẽ là:

+ Với n<=3: $k(k+1)^{n-1}$

+ Với n>3: $k(k+1)^{n-1}-(2n^2-13n+21)(k+1)$.

Qua các phép tính trên, ta nhận thấy số cách tách từ và số tổ hợp từ loại sẽ tăng lên theo hàm mũ khi số lượng âm tiết tăng. Nó sẽ gây ra hiện tượng bùng nổ tổ hợp. Nhưng đây lại là điều không thể tránh khỏi. Ta chỉ có thể làm giảm bớt số lượng cách tách từ cần kiểm tra. Cụ thể, trong quá trình tìm kiếm các cách tách từ, nếu ta phát hiện một cách tách từ nào đó không phù hợp, ta phải loại bỏ tất cả các nhánh xuất phát từ cách tách từ đó.

Sau đây, khoá luận sẽ đưa ra một phương pháp để giải quyết vấn đề này.

Ta xây dựng một mảng hai chiều với 3 cột và n dòng, n là độ dài của câu. Trong mảng, cột thứ nhất chứa các từ có một âm tiết, cột thứ hai chứa các từ có hai âm tiết, cột thứ ba chứa các từ có ba âm tiết. Các từ sẽ được đưa vào mảng nếu là từ đúng.

Ví dụ với câu: “phó giáo sư là một chức danh”. Ta sẽ có mảng như sau:

| | | |
|---|---|---|
| phó | | phó giáo sư |
| giáo | giáo sư | |
| sư | | |
| là | | |
| một | | |
| chức | chức danh | |
| danh | | |

Dựa trên bảng trên, ta có thể có các câu sau (Các từ sẽ được gạch chân):

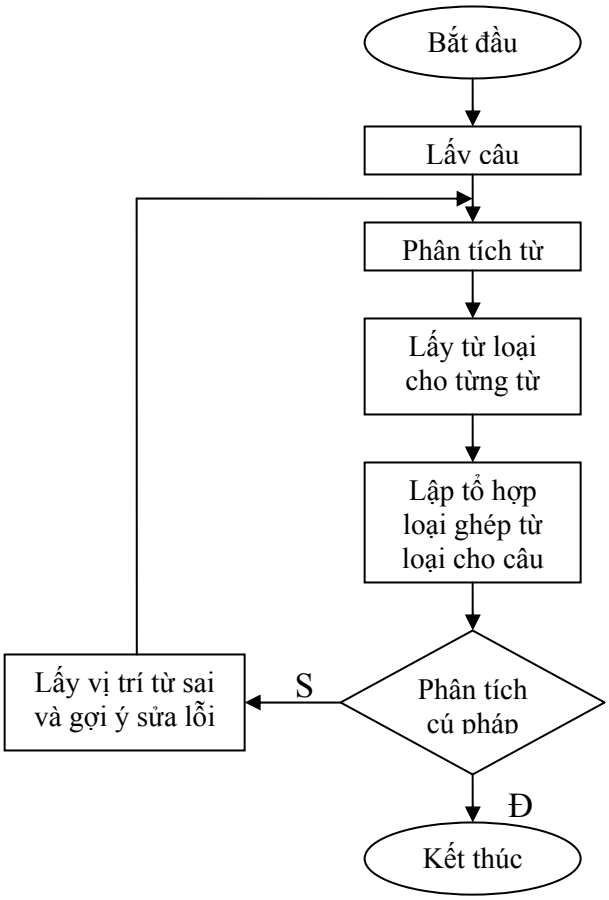
- phó giáo sư là một chức danh
- phó giáo sư là một chức danh
- phó giáo sư là một chức danh
- phó giáo sư là một chức danh
- phó giáo sư là một chức danh
- phó giáo sư là một chức danh

Một thuật toán duyệt qua bảng để lấy các câu như trên là đơn giản (sẽ không trình bày ở đây). Đây sẽ là một thuật toán đệ qui, nó loại bỏ được các nhánh ghép từ không phù hợp.

Phương pháp này có một điểm yếu là vẫn phải duyệt qua tất cả 3n-3 từ đề nghị để kiểm tra. Nhưng lại hạn chế được hiện tượng bùng nổ tổ hợp trong quá trình tách câu và có một thuật toán tách câu thành từ đơn giản hơn nhiều. Thuật toán có độ phức tạp là $O(n^3)$.

### 1.6.2. Thuật toán bắt lỗi chính tả mức cú pháp:

Qui trình bắt lỗi mức cú pháp bằng cách phân tích cú pháp một câu được trình bày bằng lưu đồ như sau:




Thuật toán này có độ phức tạp chính là độ phức tạp của thuật toán phân tích cú pháp.

### 1.6.3. Phương pháp tạo gợi ý sửa lỗi mức cú pháp

Phương pháp gợi ý sửa lỗi mức âm tiết đã trình bày ở mục 1.4.3 là phương pháp tạo gợi ý cho một âm tiết. Ở mức cú pháp, các từ sai không phải chỉ là các từ một âm tiết mà có thể là các từ hai âm tiết hoặc ba âm tiết, nên phương pháp trên không còn phù hợp nữa. Ta phải có phương pháp gợi ý thích hợp cho trường hợp này.

Về cơ bản, các từ tiếng Việt đều là sự kết hợp của một hoặc nhiều âm tiết. Các âm tiết trong một từ có thể nói là độc lập. Do đó, ta có thể thừa kế phương pháp gợi ý sửa lỗi mức âm tiết cho từng âm tiết một, rồi kết hợp các âm tiết gợi ý thành từ để được các từ gợi ý.

Ví dụ:

Ta muốn đánh từ đúng là "giáo sư" nhưng lại đánh sai thành "giao sư". Qua phân tích cú pháp câu có thể bắt lỗi được từ này.

Khi đó, với âm tiết "giao" ta có các âm tiết gợi ý: giao, giáo, ngoái,…

Và với âm tiết "sư" ta có các âm tiết gợi ý: sư, sưa, sưu,…

Lần lượt kết hợp các âm tiết trên lại, từ nào có nghĩa sẽ là từ gợi ý. Như thế, ta chỉ có từ "giáo sư" là từ duy nhất có nghĩa.

**Thuật toán tìm từ gợi ý sửa lỗi mức cú pháp như sau:**

1. Lấy số âm tiết của từ.
2. Tách từng âm tiết.
3. Duyệt qua từng âm tiết một.
    i. Lấy các âm tiết gợi ý.
4. Kết hợp các âm tiết gợi ý thành từ.
5. Duyệt qua các từ vừa tạo thành
    i. Nếu từ có nghĩa, đưa vào danh sách từ gợi ý.
    ii. Ngược lại, duyệt qua từ khác.

Thuật toán này có độ phức tạp là $O(n^3)$.