TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

──────── \* ────────

ĐỒ ÁN

**TỐT NGHIỆP ĐẠI HỌC**

NGÀNH CÔNG NGHỆ THÔNG TIN

**PHÂN TÍCH CÚ PHÁP TRONG TỔNG HỢP TIẾNG NÓI TIẾNG VIỆT**

Sinh viên thực hiện : **Lê Quang Thắng**

Lớp: Công nghệ phần mềm – K51

Giáo viên hướng dẫn: TS. **Cao Tuấn Dũng**

ThS **Nguyễn Thị Thu Trang**

HÀ NỘI 05-2011

**MỤC LỤC**

[LỜI CẢM ƠN v](file:///C:\Users\lelightwin\Desktop\DATN\LELIGHTWIN_DATN_2011.docx#_Toc294161597)

[DANH SÁCH CÁC HÌNH TRONG LUẬN VĂN vii](#_Toc294161598)

[DANH SÁCH CÁC BẢNG TRONG LUẬN VĂN viii](#_Toc294161599)

[LỜI NÓI ĐẦU ix](#_Toc294161600)

[CHƯƠNG 1. PHÂN TÍCH CÚ PHÁP TRONG TỔNG HỢP TIẾNG NÓI 1](#_Toc294161601)

[1.1. Tổng quan về tổng hợp tiếng nói 1](#_Toc294161602)

[1.1.1. Chuẩn hóa văn bản 2](#_Toc294161603)

[1.1.2. Phân tích cú pháp 2](#_Toc294161604)

[1.1.3. Phân tích ngữ điệu 3](#_Toc294161605)

[1.1.4. Tổng hợp mức thấp 3](#_Toc294161606)

[1.2. Vai trò của phân tích cú pháp trong tổng hợp tiếng nói 3](#_Toc294161607)

[1.3. Một số nghiên cứu trên thế giới về phân tích cú pháp 3](#_Toc294161608)

[1.3.1. Cách tiếp cận từ trên xuống (Top-down) 4](#_Toc294161609)

[1.3.2. Cách tiếp cận từ dưới lên (Bottom-up) 4](#_Toc294161610)

[1.3.3. Thuật toán Earley 4](#_Toc294161611)

[1.3.4. Thuật toán CYK 5](#_Toc294161612)

[1.3.5. Nhận xét 5](#_Toc294161613)

[1.4. Những khó khăn của phân tích cú pháp tiếng Việt. 5](#_Toc294161614)

[1.5. Nhiệm vụ của đồ án tốt nghiệp 6](#_Toc294161615)

[CHƯƠNG 2. TIẾNG VIỆT VÀ PHƯƠNG PHÁP, MÔ HÌNH PHÂN TÍCH CÚ PHÁP 7](#_Toc294161616)

[2.1. Các từ loại tiếng Việt 7](#_Toc294161617)

[2.1.1. Danh từ(N) 7](#_Toc294161618)

[2.1.1.1. Danh từ đơn thể-Ns 7](#_Toc294161619)

[2.1.1.2. Danh từ tổng thể-Nc 7](#_Toc294161620)

[2.1.1.3. Danh từ đơn vị-Nu 7](#_Toc294161621)

[2.1.1.4. Danh từ trừu tượng-Na 7](#_Toc294161622)

[2.1.1.5. Danh từ riêng(No) 7](#_Toc294161623)

[2.1.2. Động từ(V) 8](#_Toc294161624)

[2.1.2.1. Động từ ngoại động-Vt 8](#_Toc294161625)

[2.1.2.2. Động từ nội động-Vi 8](#_Toc294161626)

[2.1.2.3. Động từ tồn tại-Ve 8](#_Toc294161627)

[2.1.2.4. Động từ biến hóa-Vf 8](#_Toc294161628)

[2.1.2.5. Động từ ý chí-Vv 8](#_Toc294161629)

[2.1.2.6. Động từ tiếp thụ-Va 8](#_Toc294161630)

[2.1.2.7. Động từ so sánh-Vc 8](#_Toc294161631)

[2.1.2.8. Động từ đặc biệt: động từ “là”-Vz 9](#_Toc294161632)

[2.1.3. Tính từ(A) 9](#_Toc294161633)

[2.1.3.1. Tính từ hàm chất(Ai) 9](#_Toc294161634)

[2.1.3.2. Tính từ hàm lượng(An) 9](#_Toc294161635)

[2.1.4. Phụ từ(R) 9](#_Toc294161636)

[2.1.4.1. Phụ từ thời gian(Rt) 9](#_Toc294161637)

[2.1.4.2. Phụ từ mức độ(Rd) 9](#_Toc294161638)

[2.1.4.3. Phụ từ so sánh(Rc) 9](#_Toc294161639)

[2.1.4.4. Phụ từ khẳng định – phủ định(RfY-RfN) 9](#_Toc294161640)

[2.1.4.5. Phụ từ mệnh lệnh(Ri) 10](#_Toc294161641)

[2.1.5. Kết từ(C) 10](#_Toc294161642)

[2.1.5.1. Kết từ chính phụ(Cm): 10](#_Toc294161643)

[2.1.5.2. Kết từ liên hợp(Cp): 10](#_Toc294161644)

[2.1.6. Đại từ(P) 10](#_Toc294161645)

[2.1.6.1. Đại từ sự vật 10](#_Toc294161646)

[2.1.6.2. Đại từ hoạt động – tính chất(Pl) 10](#_Toc294161647)

[2.1.6.3. Đại từ nghi vấn(Pi) 10](#_Toc294161648)

[2.1.7. Trợ từ (M) 10](#_Toc294161649)

[2.2. Cụm từ tiếng việt 10](#_Toc294161650)

[2.2.1.1. Cụm danh từ(NP) 11](#_Toc294161651)

[2.2.1.1.1. Khái niệm 11](#_Toc294161652)

[2.2.1.1.2. Cấu tạo 11](#_Toc294161653)

[2.2.1.1.3. Chức năng ngữ pháp của cụm danh từ 11](#_Toc294161654)

[2.2.1.2. Cụm động từ(VP) 11](#_Toc294161655)

[2.2.1.2.1. Khái niệm 11](#_Toc294161656)

[2.2.1.2.2. Cấu tạo 11](#_Toc294161657)

[2.2.1.2.3. Chức năng ngữ pháp của cụm động từ 12](#_Toc294161658)

[2.2.1.3. Cụm tính từ(AP) 12](#_Toc294161659)

[2.2.1.3.1. Khái niệm 12](#_Toc294161660)

[2.2.1.3.2. Cấu tạo 12](#_Toc294161661)

[2.2.1.3.3. Chức năng ngữ pháp 13](#_Toc294161662)

[2.3. Các kiểu câu của tiếng việt 13](#_Toc294161663)

[2.4. Mô hình PCFG và giải thuật phân tích cú pháp CYK. 13](#_Toc294161664)

[2.4.1. Mô hình văn phạm CFG. 13](#_Toc294161665)

[2.4.2. Thuật toán CYK. 14](#_Toc294161666)

[2.4.2.1. Thuật toán tạo bảng của CYK. 14](#_Toc294161667)

[2.4.2.2. Ví dụ minh họa cho thuật toán CYK. 14](#_Toc294161668)

[2.4.2.3. Thuật toán CYK cải tiến 15](#_Toc294161669)

[2.4.2.4. Những khó khăn của thuật toán CYK 16](#_Toc294161670)

[2.5. Mô hình xác suất PCFG 16](#_Toc294161671)

[2.5.1. Định nghĩa cơ bản về PCFG 16](#_Toc294161672)

[2.5.2. Các loại xác suất trong PCFG 17](#_Toc294161673)

[2.5.2.1. Xác suất trong (inside) 17](#_Toc294161674)

[2.5.2.2. Xác suất ngoài (outside) 18](#_Toc294161675)

[2.5.3. Cách khử nhập nhằng mặt cú pháp với PCFG 18](#_Toc294161676)

[CHƯƠNG 3. CÁC ĐỀ XUẤT CỦA ĐỒ ÁN CHO PHÂN TÍCH CÚ PHÁP TIẾNG VIỆT 21](#_Toc294161677)

[3.1. Nhận xét về mô hình PCFG và thuật toán CYK cải tiến ở chương 2 cho tiếng Việt. 21](#_Toc294161678)

[3.2. Cải tiến thuật toán phân tích cú pháp CYK bằng giải thuật tìm kiếm beam search 22](#_Toc294161679)

[3.2.1. Thuật toán tìm kiếm beam search 22](#_Toc294161680)

[3.2.2. Quy trình áp dụng và thực hiện thuật toán beam search 23](#_Toc294161681)

[3.2.2.1. Cách tính inside trong beam search 23](#_Toc294161682)

[3.2.2.2. Cách tính xác suất outside trong beam search 24](#_Toc294161683)

[3.2.2.3. Quy trình thực hiện thuật toán beam search 26](#_Toc294161684)

[3.2.3. Nhận xét về thuật toán beam search 27](#_Toc294161685)

[3.3. Thuật toán tìm kiếm A\* 27](#_Toc294161686)

[3.3.1. Thuật toán A\* 28](#_Toc294161687)

[3.3.2. Thuật toán A\* trong phân tích cú pháp 29](#_Toc294161688)

[3.3.3. Hàm ưu tiên trong A\* 30](#_Toc294161689)

[3.3.3.1. Sử dụng duy nhất xác suất inside 30](#_Toc294161690)

[3.3.3.2. Tính outside một tầng 30](#_Toc294161691)

[3.3.3.3. Tính outside bằng phương pháp rút gọn tập luật 31](#_Toc294161692)

[3.3.4. Cải tiến thuật toán A\* trong phân tích cú pháp với giải thuật lelightwin 32](#_Toc294161693)

[3.3.4.1. Đặt vấn đề 32](#_Toc294161694)

[3.3.4.2. Ý tưởng của giải thuật lelightwin 32](#_Toc294161695)

[3.3.4.3. Mô hình thuật toán lelightwin cơ bản 34](#_Toc294161696)

[3.3.4.3.1. Phân loại phần tử 34](#_Toc294161697)

[3.3.4.3.2. Sinh chuỗi kết hợp dựa trên các phần tử đã phân loại 35](#_Toc294161698)

[3.3.4.4. Thuật toán lelightwin Prunning 39](#_Toc294161699)

[3.3.4.4.1. Huấn luyện cho bộ cắt tỉa của lelightwin 40](#_Toc294161700)

[3.3.4.4.2. Thực hiện quá trình cắt tỉa trong giải thuật lelightwin 42](#_Toc294161701)

[3.3.4.5. Thuật toán A\* kết hợp với thuật toán lelightwin prunning 43](#_Toc294161702)

[3.3.4.6. Nhận xét thuật toán A\* - lelightwin prunning 43](#_Toc294161703)

[CHƯƠNG 4. PHÁT TRIỂN, THỬ NGHIỆM VÀ ĐÁNH GIÁ HỆ THỐNG 45](#_Toc294161704)

[4.1. Phân tích hệ thống 45](#_Toc294161705)

[4.1.1. Tiền xử lí văn bản 46](#_Toc294161706)

[4.1.1.1. Tách từ 46](#_Toc294161707)

[4.1.1.2. Gán nhãn 47](#_Toc294161708)

[4.1.2. Quản lý luật (Rule Manager) 47](#_Toc294161709)

[4.1.2.1. Load file dữ liệu chứa tập luật 48](#_Toc294161710)

[4.1.2.2. Tính xác suất PCFG cho mỗi luật 48](#_Toc294161711)

[4.1.3. Phân tích cú pháp 49](#_Toc294161712)

[4.2. Tổ chức lưu trữ dữ liệu 49](#_Toc294161713)

[4.2.1. Dữ liệu văn bản chuẩn hóa đầu vào 50](#_Toc294161714)

[4.2.2. Dữ liệu chuyển giao của bộ tiền xử lí văn bản 51](#_Toc294161715)

[4.2.3. Tập luật cú pháp 51](#_Toc294161716)

[4.2.4. Kho dữ liệu VietTreeBank 51](#_Toc294161717)

[4.2.5. Dữ liệu đầu ra của hệ thống 52](#_Toc294161718)

[4.3. Cài đặt hệ thống 54](#_Toc294161719)

[4.3.1. Công cụ lựa chọn 54](#_Toc294161720)

[4.3.2. Cài đặt chương trình 54](#_Toc294161721)

[4.3.2.1. Gói RuleManager 54](#_Toc294161722)

[4.3.2.1.1. Lớp Rule 54](#_Toc294161723)

[4.3.2.1.2. Lớp RuleSet 55](#_Toc294161724)

[4.3.2.2. Gói phần tử parsing Element 56](#_Toc294161725)

[4.3.2.2.1. Lớp CYK Element 56](#_Toc294161726)

[4.3.2.2.2. Lớp Cell 56](#_Toc294161727)

[4.3.2.2.3. Lớp AstarElement 57](#_Toc294161728)

[4.3.2.3. Gói xử lí chung Common 58](#_Toc294161729)

[4.3.2.3.1. Lớp PartOfSpeech 58](#_Toc294161730)

[4.3.2.3.2. Lớp Functions 59](#_Toc294161731)

[4.3.2.4. Gói Analysis 59](#_Toc294161732)

[4.3.2.4.1. Lớp CYKBeamSearch 59](#_Toc294161733)

[4.3.2.4.2. Lớp AStar 60](#_Toc294161734)

[4.3.2.4.3. Lớp LeLightWin 61](#_Toc294161735)

[4.3.2.4.4. Lớp Sentence 62](#_Toc294161736)

[4.4. Thử nghiệm và đánh giá 63](#_Toc294161737)

[4.4.1. Kết quả thử nghiệm 63](#_Toc294161738)

[4.4.1.1. Tập dữ liệu thử nghiệm 63](#_Toc294161739)

[4.4.1.2. Kết quả phân tích. 64](#_Toc294161740)

[4.4.2. Đánh giá hệ thống 68](#_Toc294161741)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 69](#_Toc294161742)

[Tài liệu tham khảo 71](#_Toc294161743)

**PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP**

1. Thông tin về sinh viên

Họ và tên sinh viên: Lê Quang Thắng

Điện thoại liên lạc: 01675888962 Email: lelightwin@gmail.com

Lớp: Công nghệ phần mềm K51 Hệ đào tạo: Đại học chính quy

Đồ án tốt nghiệp được thực hiện tại: Trung tâm nghiên cứu Mica – Trường Đại học Bách Khoa Hà Nội.

Thời gian làm ĐATN: Từ ngày 21/02/2011 đến 28/05/2011

2. Mục đích nội dung của ĐATN

Tìm hiểu phương pháp phân tích cú pháp tiếng Việt phục vụ cho bộ tổng hợp tiếng nói tiếng Việt

3. Các nhiệm vụ cụ thể của ĐATN

* Tìm hiểu tiếng Việt, nghiên cứu các phương pháp phân tích tiếng Việt trong nước và ngoài nước.
* Đề xuất ra giải pháp giúp tối ưu hóa hiệu năng của bộ phân tích cú pháp.
* Tiến hành kết nối dữ liệu với các công đoạn khác trong tổng hợp tiếng nói.

4. Lời cam đoan của sinh viên:

Tôi – *Lê Quang Thắng* - cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *TS. Cao Tuấn Dũng và ThS. Nguyễn Thị Thu Trang*.

Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

|  |  |  |
| --- | --- | --- |
|  | *Hà Nội, ngày 20 tháng 05 năm 2011*  Tác giả ĐATN  *Lê Quang Thắng* | |
| 5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ:    *Hà Nội, ngày 28 tháng 05 năm 2011*  Giáo viên hướng dẫn  *TS. Cao Tuấn Dũng* | |

LỜI CẢM ƠN

*Đầu tiên, con xin cảm ơn bố mẹ, những người đã nuôi nấng, động viên, giúp đỡ con cho đến tận lúc con có thể tự mình đặt tay gõ ra những dòng này. Anh xin cảm ơn cô em gái dữ dằn, nghịch ngợm của anh đã động viên, “chọc tức”anh trong suốt quá trình anh làm đồ án.*

*Em xin được gửi lời cảm ơn chân thành tới các thầy cô giáo trong trường Đại học Bách Khoa Hà Nội cũng như các thầy cô trong Viện Công nghệ thông tin và truyền thông đã truyền dạy cho em những kiến thức và kinh nghiệm quý giá trong suốt quá trình học tập tu dưỡng trong suốt 5 năm qua.*

*Em xin được gửi lời cảm ơn đến TS. Cao Tuấn Dũng, TS. Trần Đỗ Đạt và Ths.Nguyễn Thị Thu Trang đã dẫn dắt em trong suốt quá trình làm đồ án. Em xin cảm ơn một lần nữa, thầy cô là những người giáo viên tận tình nhất mà em từng biết đến.*

*Em xin đặc biệt gửi lời cảm ơn đến thầy giáo Đỗ Bá Lâm. Dù không phải là giáo viên hướng dẫn của em nhưng nếu không có thầy, em đã không thể hoàn thành đồ án tốt đến như vậy.*

*Em xin được gửi lời cảm ơn đến bạn Tô Hoàng Long lớp Việt Nhật K51. Cậu là người bạn thân luôn sát cánh cùng tớ từ cấp 3 đến giờ. Và cho đến tận khi lên đại học, rồi đến cả khi tớ có thể tự hào vì đã hoàn thành đồ án, cậu vẫn luôn là người giúp tớ có nghị lực vượt qua những cơn ngã lòng vì mệt mỏi.*

*Em xin được gửi lời cảm ơn đến nhóm bạn Q4T của lớp công nghệ phần mềm. Chúng ta có thể không phải là anh em, nhưng chúng ta là một đội thật tuyệt vời phải không các bạn. Hãy cùng nhau hoàn thành tốt đồ án cuộc đời các bạn nhé, các chiến hữu của tớ.*

*Em xin được gửi lời cảm ơn đến nhóm bạn ở trung tâm mica. Các cậu chính là nguồn cảm hứng khiến tớ không ngừng phấn đấu bản thân mình. Chúng ta cũng đã trải qua nhiều kỉ niệm thật khó quên, các bạn sẽ mãi là những người bạn tốt của tớ.*

*Em cũng bày tỏ lòng biết ơn tới trung tâm nghiên cứu Mica đã tạo điều kiện về cơ sở vật chất cho em trong quá trình thực hiện đồ án tốt nghiệp.*

*Em cũng muốn gửi lời cảm ơn tới tập thể lớp Công nghệ phần mềm K51 đã tạo một môi trường thi đua học tập lành mạnh, tạo điều kiện cho sự phát triển của các thành viên trong lớp.*

*Hà Nội, ngày 25 thg 05 năm 2011*

Lê Quang Thắng

Lớp CNPM – K51

Viện CNTT & TT **–** ĐH Bách Khoa HN

DANH SÁCH CÁC HÌNH TRONG LUẬN VĂN

[Hình 1‑2. Cây phân tích cú pháp của câu “tôi đang làm đồ án” 2](#_Toc294161391)

[Hình 1‑1. Mô hình hệ thống tổng hợp tiếng nói. 2](file:///C:\Users\lelightwin\Desktop\DATN\LELIGHTWIN_DATN_2011.docx#_Toc294161392)

[Hình 2‑1. Một phần tử trong bảng CYK. 14](#_Toc294161393)

[Hình 2‑2. CYK cải tiến với câu"anh múa kiếm và vụt côn". 16](#_Toc294161394)

[Hình 2‑3. Hình ảnh minh họa của xác suất inside. 17](#_Toc294161395)

[Hình 2‑4. Cây phân tích t1. 19](#_Toc294161396)

[Hình 2‑5. Cây phân tích t2. 19](#_Toc294161397)

[Hình 3‑1. Minh họa vui cho thuật toán cắt tỉa beam search. 22](#_Toc294161398)

[Hình 3‑2. Hình ảnh của một nút phân tích hoàn chỉnh. 24](#_Toc294161399)

[Hình 3‑3. Hình ảnh của một nút phân tích sử dụng biến wait. 24](#_Toc294161400)

[Hình 3‑4. Hình ảnh mô phỏng outside của một nút. 25](#_Toc294161401)

[Hình 3‑5. Outside của trường hợp nút hoàn chỉnh. 26](#_Toc294161402)

[Hình 3‑6. Outside của trường hợp nút chưa hoàn chỉnh. 26](#_Toc294161403)

[Hình 3‑7. Mô hình của thuật toán lelightwin. 34](#_Toc294161404)

[Hình 3‑8. Giai đoạn phân loại phần tử trong CHART. 35](#_Toc294161405)

[Hình 3‑9. Các công đoạn của thuật toán sinh chuỗi. 35](#_Toc294161406)

[Hình 3‑10. Ví dụ minh họa thuật toán sinh chuỗi con trái. 36](#_Toc294161407)

[Hình 3‑11. Sơ đồ thuật toán sinh chuỗi con trái. 37](#_Toc294161408)

[Hình 3‑12. Sơ đồ thuật toán sinh chuỗi con phải. 38](#_Toc294161409)

[Hình 3‑13. Hình ảnh cây phân cấp dữ liệu chính. 40](#_Toc294161410)

[Hình 3‑14. Hình ảnh cây phân cấp dữ liệu con. 41](#_Toc294161411)

[Hình 4‑1. Mô hình tổng thể của hệ thống. 45](#_Toc294161412)

[Hình 4‑2. Mô hình tính xác suất PCFG cho mỗi luật. 48](#_Toc294161413)

[Hình 4‑3. Cấu trúc của dữ liệu văn bản chuẩn hóa. 50](#_Toc294161414)

[Hình 4‑4. Ví dụ minh họa về dữ liệu chuẩn hóa văn bản. 50](#_Toc294161415)

[Hình 4‑5. Cấu trúc của dữ liệu VietTreeBank. 52](#_Toc294161416)

[Hình 4‑6. Mô phỏng dữ liệu đầu ra của hệ thống. 53](#_Toc294161417)

[Hình 4‑7. CPTCP “tôi là sinh viên” 64](#_Toc294161418)

[Hình 4‑8. CPTCP “tôi là một sinh viên học rất giỏi môn toán” 65](#_Toc294161419)

[Hình 4‑9. Hình ảnh phân tích của một câu cực khó và dài. 67](#_Toc294161420)

DANH SÁCH CÁC BẢNG TRONG LUẬN VĂN

[Bảng 2‑1.Phân tích CYK cho câu “anh ấy rất ngầu” 15](#_Toc294161421)

[Bảng 3‑1. Các phần tử trong CHART 33](#_Toc294161422)

[Bảng 3‑2. Các chuỗi kết hợp của X với CHART 33](#_Toc294161423)

[Bảng 4‑1. Mô tả của chức năng tách từ 46](#_Toc294161424)

[Bảng 4‑2. Mô tả của chức năng gán nhãn từ loại 47](#_Toc294161425)

[Bảng 4‑3. Bảng mô tả của chức năng load dữ liệu 48](#_Toc294161426)

[Bảng 4‑4. Bảng mô tả của chức năng tính xác suất PCFG 48](#_Toc294161427)

[Bảng 4‑5. Bảng mô tả của chức năng phân tích cú pháp 49](#_Toc294161428)

[Bảng 4‑6. Minh họa dữ liệu đầu ra của bộ tiền xử lí văn bản 51](#_Toc294161429)

[Bảng 4‑7. Bảng mô tả của lớp Rule 54](#_Toc294161430)

[Bảng 4‑8. Bảng mô tả của lớp RuleSet 55](#_Toc294161431)

[Bảng 4‑9. Bảng mô tả của lớp Element 56](#_Toc294161432)

[Bảng 4‑10. Bảng mô tả của lớp Cell 56](#_Toc294161433)

[Bảng 4‑11. Bảng mô tả của lớp AstarElement 57](#_Toc294161434)

[Bảng 4‑12. Bảng mô tả của lớp ParOfSpeech 58](#_Toc294161435)

[Bảng 4‑13. Bảng mô tả của lớp Functions 59](#_Toc294161436)

[Bảng 4‑14. Bảng mô tả của lớp CYKBeamSeach 59](#_Toc294161437)

[Bảng 4‑15. Bảng mô tả của lớp AStar 60](#_Toc294161438)

[Bảng 4‑16. Bảng mô tả của lớp LeLightWin 61](#_Toc294161439)

[Bảng 4‑17. Bảng mô tả của lớp Sentence 63](#_Toc294161440)

[Bảng 4‑18. Bảng tổng kết thử nghiệm với 630 câu hành văn 67](#_Toc294161441)

**LỜI NÓI ĐẦU**

**Bố cục đồ án:**

**Chương 1: Phân tích cú pháp trong tổng hợp tiếng nói.**

Chương này chúng ta sẽ tập trung giới thiệu về tổng hợp tiếng nói tiếng Việt đồng thời chỉ ra vai trò của bộ phân tích cú pháp trong tổng hợp tiếng nói tiếng Việt. Từ đó, nêu lên mục đích và nhiệm vụ của đồ án

**Chương 2 : Tiếng Việt và một số chiên lược phân tích cú pháp cơ bản.**

Chương này giới thiệu cho mọi về tiếng Việt và một số lý thuyết cơ sở về các phương pháp phân tích cú pháp. Đây là một chương rất quan trọng, là kiến thức tiền đề cho việc dẫn đến quyết định đưa ra đề xuất cho mô hình và phương pháp phân tích cú pháp của đồ án.

**Chương 3: Các đề xuất của đồ án cho phân tích cú pháp tiếng Việt.**

Hai chương đầu chỉ là cơ sở lý thuyết mà đồ án sử dụng, sang chương thứ 3 đồ án sẽ nêu ra các mô hình và các phương pháp mà đồ án áp dụng cho phân tích cú pháp tiếng Việt. Ngoài ra, một số cải tiến trong quá trình làm đồ án cũng sẽ được mô tả kĩ càng.

**Chương 4: Xây dựng và đánh giá chương trình.**

Đây là chương mô tả quá trình cài đặt và xây dựng chương trình theo những phương pháp đã trình bày ở chương 3. Sau đó, sẽ thử nghiệm và đánh giá hiệu năng của chương trình dựa trên những kết quả đạt được.

# PHÂN TÍCH CÚ PHÁP TRONG TỔNG HỢP TIẾNG NÓI

*Trong chương này, luận văn sẽ giới thiệu:*

* *Tổng quan về tổng hợp tiếng nói.*
* *Vị trí và vai trò của bài toán phân tích cú pháp trong tổng hợp tiếng nói.*
* *Những nghiên cứu trên thế giới về phân tích cú pháp.*
* *Những khó khăn đối với bài toán phân tích cú pháp tiếng Việt.*
* *Nhiệm vụ của đồ án tốt nghiệp.*

## Tổng quan về tổng hợp tiếng nói

Tổng hợp tiếng nói (TTS ­– text to speech) là tiến trình tạo ra giọng nói nhân tạo của con người từ đầu vào là văn bản hoặc các mã phát âm, nhưng chủ yếu là văn bản. Các hệ thống tổng hợp tiếng nói có rất nhiều ứng dụng trong thực tế. Một trong những ứng dụng có ý nghĩa nhất của nó là có thể hỗ trợ người khiếm thị biết được nội dung của một đoạn văn bản nào đó như báo chí, truyện tranh do máy đọc ra. Hệ thống TTS cũng có thể được lắp đặt trong các phần mềm xử lí văn hoặc trình duyệt web. Nhưng có lẽ hệ thống này được áp dụng nhiều nhất là trong những ứng dụng hỏi đáp giữa người và máy. Chính vì khả năng ứng dụng thực tiễn lớn như vậy nên các nhà khoa học đã tìm hiểu và nghiên cứu về TTS từ những năm 1930, và tiếp tục phát triển và hoàn thiện cho đến ngày nay.

Đối với một hệ thống TTS, việc tạo ra được giọng nói của con người từ văn bản không khó, cái khó làm sao để tạo ra được một giọng nói có chất lượng thật tốt. Hai tính chất quan trọng dùng để xác định chất lượng của hệ thống tổng hợp tiếng nói là mức độ tự nhiên và mức độ dễ nghe. Mức độ tự nhiên là mức độ tương đồng giữa giọng của người với giọng của máy, còn việc câu phát âm ra có dễ hiểu được hay không là để chỉ mức độ dễ nghe. Một hệ thống tổng hợp tiếng nói lý tưởng phải là một hệ thống có thể thỏa mãn tối đa hai tính chất này.

Bộ tổng hợp tiếng nói được chia làm hai phần chính: tổng hợp mức cao và tổng hợp mức thấp. Nhiệm vụ phần tổng hợp mức cao là chuẩn hóa văn bản, phân tích cú pháp, phát sinh thông tin về ngữ âm, ngữ điệu. Phần tổng hợp mức thấp dựa vào các thông tin phía trên sẽ tiến hành tìm kiếm và lựa chọn đơn vị âm, thực hiện ghép nối và làm trơn tín hiệu, cho ra tiếng nói cần tổng hợp. Sau đây ta sẽ đi mô tả từng công việc.

**Hệ thống tổng hợp tiếng nói**

**Tổng hợp mức thấp**

Lựa chọn đơn vị âm

Ghép nối đơn vị âm

**Tổng hợp mức cao**

Phân tích văn bản

* *Phân tích cấu trúc*
* *Chuẩn hóa văn bản*
* *Phân tích ngôn ngữ*

Phân tích ngữ âm

Phân tích ngữ điệu

* *Tần số cơ bản*
* *Trường độ*

Hình ‑. Mô hình hệ thống tổng hợp tiếng nói.

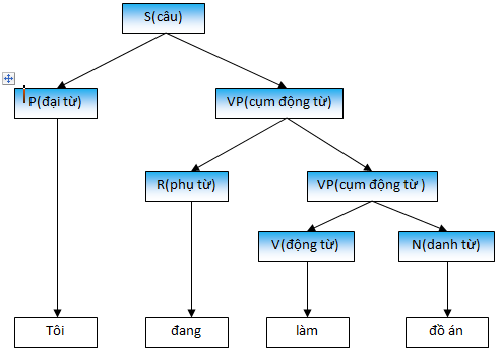
### Chuẩn hóa văn bản

Chuẩn hóa văn bản là quá trình phát hiện và chuẩn hóa những đoạn mà hệ thống tổng hợp không thể xử lí được về dạng có thể xử lí được. Trong hệ thống tổng hợp tiếng nói, việc chuẩn hóa văn bản là công đoạn đầu tiên có ảnh hưởng quan trọng trong việc đảm bảo văn bản được đọc một cách đúng đắn. Ví dụ : một đoạn văn bản “*tôi bảo vệ đồ án vào ngày 08/06/2011 tại trường ĐHBKHN*” sẽ được chuyển thành dạng đọc được “*tôi bảo vệ đồ án vào ngày mùng tám tháng sáu năm hai nghìn không trăm mười một tại trường Đại học Bách Khoa Hà Nội*”

### Phân tích cú pháp

Phân tích cú pháp là nhằm phân tích một câu thành những thành phần văn phạm có liên quan với nhau và được thể hiện thành cây cú pháp. Kết quả của phân tích cú pháp ảnh hướng trực tiếp đến rất nhiều công đoạn trong tổng hợp tiếng nói.

Ví dụ với cây phân tích cú pháp của câu *“tôi đang làm đồ án”*



Hình ‑. Cây phân tích cú pháp của câu “tôi đang làm đồ án”

### Phân tích ngữ điệu

Bộ phân phân tích ngữ điệu có nhiệm vụ mô hình hóa được ngữ điệu của tiếng nói và đưa ra các thông tin về ngữ điệu dưới dạng số liệu và làm đầu vào cho bộ tổng hợp mức thấp. Bộ phân tích ngữ điệu có ảnh hưởng lớn đến mức độ tự nhiên của tiếng nói tổng hợp.

### Tổng hợp mức thấp

Tổng hợp mức thấp là quá trình kết hợp các đoạn tín hiệu (ví dụ như diphone). Các đoạn tín hiệu này đã được phân tích, xử lý qua mức cao (phân tích ngữ âm, phân tích ngôn điệu).

Hiện nay có ba phương pháp tổng hợp tiếng nói. Phương pháp đơn giản nhất để phát sinh tiếng nói tổng hợp là phát các mẫu tiếng nói đã thu từ tiếng nói tự nhiên (như các từ hoặc câu). Phương pháp này cho chất lượng tương đối tốt nhưng gặp phải hạn chế là số lượng từ vựng trong cơ sở dữ liệu rất lớn. Bên cạnh đó tiếng nói cũng có thể tạo ra bằng cách mô phỏng hệ thống phát âm. Phương pháp này cho chất lượng rất tốt nhưng thực hiện khá phức tạp. Một phương pháp nữa cũng được dùng để tổng hợp tiếng nói là tổng hợp formant.

## Vai trò của phân tích cú pháp trong tổng hợp tiếng nói

Phân tích cú pháp đóng một vai trò cực kì quan trong trọng tổng hợp tiếng nói. Một hệ thống tổng hợp tiếng nói muốn có được chất lượng tiếng nói tốt nhất chắc chắn phải xem xét đến liên kết ngữ pháp của văn bản. Các đoạn ngắt nghỉ giữa chủ ngữ, vị ngữ; các cụm từ đủ loại từ mức cao đến mức thấp; các từ chính trong câu, trong đoạn hỗ trợ trong việc đọc đúng trọng âm… Rất nhiều lợi ích có thể mang lại từ việc phân tích cú pháp.

Nhưng đó chỉ là ở mức lợi ích riêng rẽ. Một cái nhìn tổng thể hơn về lợi ích của phân tích cú pháp đó là có thể cung cấp cho hệ thống tổng hợp tiếng nói một cái nhìn toàn cảnh về cấu trúc ngữ pháp của văn bản, làm tiền đề cho rất nhiều ý tưởng khoa học thông minh, sáng tạo để cải thiện chất lượng của bộ tổng hợp tiếng nói. Như vậy, trong tổng hợp tiếng nói, phân tích cú pháp đóng một vai trò không hề nhỏ trong việc góp phần tạo nên một bộ tổng hợp tiếng nói chất lượng.

## Một số nghiên cứu trên thế giới về phân tích cú pháp

*Trên thế giới*, bài toán phân tích cú pháp đã được các nhà khoa học nghiên cứu và triển khai từ rất lâu. Rất nhiều những phương pháp cũng như những mô hình phân tích cú pháp đã ra đời với những thành công đáng kể và càng ngày càng hoàn thiện hơn.

Mô hình văn phạm phi ngữ cảnh CFG được áp dụng cho phân tích cú pháp đầu tiên, được sử dụng trong việc biểu diễn tập luật cú pháp, luật từ vựng và các ký hiệu nhãn từ loại, cụm từ loại. Tuy nhiên, mô hình này còn quá sơ khai và gặp rất nhiều sự nhập nhằng trong công đoạn phân tích.

Mô hình PCFG là mô hình phát triển lên từ CFG, kế thừa tất cả các đặc điểm của CFG. Tuy nhiên, PCFG có thêm một tham số cho một luật cú pháp hỗ trợ bộ phân tích trong việc xóa bỏ nhập nhằng về mặt cú pháp của ngôn ngữ.

Mô hình LPCFG là mô hình tiên tiến nhất hiện nay và cho kết quả khả quan nhất. LPCFG tận dụng tất cả các ưu điểm của PCFG và thêm vào những tham số về từ vựng giúp bộ phân tích cú pháp có thể tránh được không chỉ sự nhập nhằng về mặt ngữ pháp mà cả sự nhập nhằng ở cấp độ từ vựng.

Trong khuôn khổ của đồ án, chúng ta sẽ tìm hiểu và nghiên cứu mô hình PCFG để áp dụng cho tiếng Việt.

### Cách tiếp cận từ trên xuống (Top-down)

Về mặt phương pháp, phân tích cú pháp theo cách tiếp cận từ trên xuống bắt đầu với kí hiệu S (sentence). Đây chính là cấu trúc cao nhất của một câu và hình thành nên trạng thái ban đầu của cấu trúc câu. Kế tiếp, mỗi kí hiệu trong chuỗi trạng thái hiện tại sẽ được viết lại thành những cấu trúc thấp hơn dựa vào các luật có sẵn tạo thành một *danh sách các kí hiệu.*

Ví dụ : Câu bắt đầu với kí hiệu S, sau đó nó áp dụng luật S →NP VP. Danh sách kí hiệu lúc này là (NP VP). Sau đó, kí hiệu NP được lấy ra để xét và nó thoả mãn luật NP→NP AP nên NP và AP sẽ được thêm vào danh sách. Danh sách ký hiệu lúc này sẽ là (NP AP VP)…

Quá trình cứ lặp lại một cách đệ quy cho đến khi nào trạng thái của câu bao gồm toàn những kí hiệu kết thúc.

### Cách tiếp cận từ dưới lên (Bottom-up)

Giống như tên được gọi, quá trình hình thành cây cú pháp của phương pháp này đi từ mức thấp lên mức cao hay từ lá lên gốc. Điểm khác biệt giữa cách tiếp cận từ dưới lên và từ trên xuống được trình bày ở trên là cách mà luật ngữ pháp được sử dụng. Ví dụ khi xét đến luật :

*NP* → *ART ADJ N*

Trong hệ thống từ trên xuống, ta sử dụng luật để tìm NP bằng các tìm kiếm chuỗi ART ADJ N. Ngược lại, trong hệ thống từ dưới lên, từ kết quả hình thành ở bước trước đó, bạn đã có một chuỗi ART ADJ N và bạn gán cho chuỗi này nhãn là NP. Và quá trình này cũng được lặp đi lặp lại cho đến khi tìm được nút S. Quá trình phân tích kết thúc thành công.

### Thuật toán Earley

Thuật toán phân tích cú pháp Earley dựa trên chiến lược tiếp cận từ trên xuống bằng cách đi từ gốc về lá. Tuy nhiên, để tránh phải xét đi xét lại cùng một từ loại cho một từ duy nhất, giải thuật này sẽ đi cùng một lúc tất cả các hướng (tương ứng với các luật ứng viên thoả mãn xét đến thời điểm hiện tại). Đây chính là điểm tương đồng của Earley so với cách tiếp cận từ dưới lên. Như vậy nó đã khai thác được ưu điểm của hai phương pháp trên và cũng đồng nghĩa với việc loại bỏ đi những khuyết điểm của từng phương pháp.

### Thuật toán CYK

Giống như thuật toán Earley, thuật toán CYK (Cocke – Younger – Kasami) cũng là một phương pháp nâng cấp của hai chiến lược cơ bản top-down và bottom-up. CYK và Earley đều có một điểm chung là phức tạp hơn top-down và bottom-up rất nhiều về mặt thuật toán, nhưng độ phức tạp trong quá trình phân tích thì lại giảm đi rất nhiều.

### Nhận xét

Earley và CYK có thể coi là hai thuật toán nổi tiếng nhất trong phân tích cú pháp, tượng trưng cho hai chiến lược phân tích top-down và bottom-up. Hai thuật toán có rất nhiều ưu điểm so với chiến lược phân tích cơ bản nhưng lại đều mang những nhược điểm của hai chiến lược mà nó dựa theo:

* Thuật toán Earley tuy chắc chắn trả ra được cây phân tích cú pháp nhưng lại không đảm bảo cây bao phủ hết được toàn bộ câu.
* Thuật toán CYK đảm bảo cây có thể bao phủ hết được toàn bộ câu nhưng lại không đảm bảo cây đưa được đến đích.

Vì trong tổng hợp tiếng nói, điều quan trọng nhất là cây phân tích cú pháp phải bao phủ hết được toàn bộ câu đầu vào (nếu không bộ tổng hợp sẽ không đọc ra đầy đủ câu đầu vào) nên đồ án sẽ tìm hiểu và đề xuất giải pháp phân tích cú pháp cho tiếng Việt dựa trên hướng đi của thuật toán CYK.

## Những khó khăn của phân tích cú pháp tiếng Việt.

Hiện nay, với sự phát triển của bài toán xử lí ngôn ngữ tự nhiên, phân tích cú pháp tiếng Việt cũng đã đạt được một số thành tựu nhất định. Tuy nhiên, các hệ thống này vẫn đang trong quá trình hoàn thiện và kết quả đạt được vẫn chỉ ở mức trung bình. Lí giải của việc này là do sự khó khăn gặp phải trong quá trình phân tích cú pháp tiếng Việt:

* Nhập nhằng về mặt từ loại: khó khăn này là do sự đa dạng về mặt từ loại của tiếng Việt, cùng một từ nhưng ở các ngữ cảnh khác nhau lại mang ý nghĩa và loại khác nhau.
* Nhập nhằng về mặt cú pháp: khó khăn này là do sự phức tạp và nhập nhằng trong ngữ pháp của tiếng Việt.

Ví dụ câu : *“ông già đi nhanh quá”* có hai cách phân tích:

Cách 1: ông già//đi//nhanh quá.

Cách 2: ông//già đi//nhanh quá.

## Nhiệm vụ của đồ án tốt nghiệp

Như vậy, qua việc xem xét qua một số vấn đề đã nêu, những vấn đề cần đặt ra cho luận văn cần thực hiện ở đây sẽ là :

* Tìm hiểu về ngữ pháp tiếng việt và các giải thuật phân tích cú pháp để đưa ra cách mô hình hóa tiếng Việt và phương hướng để giải quyết những vấn đề khó khăn trên.
* Tìm hiểu mô hình PCFG và đề xuất ra cách thức áp dụng mô hình cho tiếng Việt đồng thời đánh giá ưu nhược điểm của mô hình để có hướng phát triển tiếp trong tương lai.
* Tìm hiểu xem giải thuật CYK có những ưu nhược điểm gì khi phân tích cú pháp tiếng Việt, từ đó đưa ra các đề xuất cải tiến về mặt giải thuật để áp dụng cho tiếng Việt.

**Kết chương**

Trong chương đầu tiên này, ta đã xác định được:

* Vị trí và vai trò của bài toán phân tích cú pháp trong tổng hợp tiếng nói và một số nghiên cứu trên thế giới về phân tích cú pháp.
* Chọn lựa hướng đi cho đồ án.
* Những khó khăn của bài toán phân tích cú pháp tiếng Việt.
* Xác định được nhiệm vụ cụ thể của đồ án

Trong chương sau, chúng ta sẽ đi tìm hiểu về tiếng việt và đi sâu vào nghiên cứu mô hình PCFG cùng với giải thuật CYK áp dụng cho phân tích cú pháp tiếng việt.

# TIẾNG VIỆT VÀ PHƯƠNG PHÁP, MÔ HÌNH PHÂN TÍCH CÚ PHÁP

*Chương này, chúng ta sẽ tập trung tìm hiểu vào những yếu tố sau:*

* *Các đặc trưng của tiếng Việt bao gồm từ loại, ngữ loại, các kiểu câu.*
* *Thuật toán CYK để phân tích cú pháp và khử nhập nhằng với mô hình PCFG*

## Các từ loại tiếng Việt

### Danh từ(N)

Danh từ là những từ mang nghĩa khái quát về sự vật .

Ví dụ: *“xe”, “người”* … Ta có thể phân danh từ thành những loại sau:

#### Danh từ đơn thể-Ns

Sự vật đơn thể là những sự vật mà rất dễ nhận ra vì chúng có thể tồn tại thành từng đơn thể, như: *“nhà”, “người”, “xe”, “máy tính”…*

#### Danh từ tổng thể-Nc

Danh từ này dùng để chỉ những sự vật không tồn tại riêng lẻ mà mà thành một tổng thể bao gồm nhiều đơn thể gộp lại, ví dụ như: *“nhân dân”, “quân đội”, “bàn ghế”…*

#### Danh từ đơn vị-Nu

Những sự vật chỉ các vật liệu, chất liệu như “nước”, “đất”, “rượu”, “thịt”, “sắt”, “thép” có đặc điểm là có thể tồn tại ở dạng đơn thể, nhưng phải qua đơn vị đo lường, tính toán như *lít, mẫu, cân*… Ví dụ: *“hai* ***lít*** *rượu”, “một* ***mẫu*** *ruộng”…*

#### Danh từ trừu tượng-Na

Sự vật trừu tượng ở đây có thể hiểu là là những khái niệm như: *tư tưởng, quan điểm, lập trường, ý nghĩ, trí tuệ…* Khi sử dụng các danh từ này làm chính tố nhìn chung không khác gì so với các loại danh từ chỉ sự vật khác: có thể có danh từ loại thể hay danh từ chỉ số lượng làm phụ tố. Ví dụ: *“một nền* ***tư tưởng****”,**“những* ***tâm tư****”…*

#### Danh từ riêng(No)

Danh từ riêng là những danh từ chỉ tên riêng của từng người, từng sự vật. Chú ý là trong tiếng Việt hiện nay tên riêng cần phải viết hoa. Đó chính là dấu hiệu nhận biết cơ bản để phân biệt danh từ riêng và danh từ chung. Ví dụ: *“Nguyễn Văn Tuấn”, “sông Hồng”…*

### Động từ(V)

Động từ có thể chia nhỏ thành những loại những loại sau:

#### Động từ ngoại động-Vt

Đó là các động từ như “ăn”, “viết”, “đọc” ... Khi sử dụng các động từ này thường phải có phụ tố chỉ đối tượng (đối tượng chịu tác động của hoạt động).Ví dụ *“như* ***ăn*** *bánh”, “****viết*** *thư”, “****may*** *áo”…*

#### Động từ nội động-Vi

Các động từ này có đặc điểm là khi dùng nó làm phần thuyết trong nòng cốt câu là đã đủ nghĩa, tức là không cần có ở sau chính tố của ngữ loại phụ tố chỉ đối tượng của họat động. Ví dụ: *“em bé đang* ***ngủ****”, “con chim đang* ***bay****”...*

#### Động từ tồn tại-Ve

Sự vật có thể có, còn, hết hay mất. Nếu những động từ loại này là chính tố thì sau đó cần các phụ tố chỉ sự vật tồn tại. Ví dụ: *“****có*** *tiền”, “****còn*** *gạo”, “****hết*** *đạn”…*

#### Động từ biến hóa-Vf

Để chỉ các trạng thái biến hóa của sự vật, khi sử dụng phải có phụ tố chỉ kết quả biến hóa. Ví dụ: *“đã* ***nên*** *người”…*

#### Động từ ý chí-Vv

Các trạng thái ý chí là : muốn, quyết, dám, toan, định… Khi sử dụng loại động từ này làm chính tố thì phải có phụ tố chỉ nội dung ý chí. Ví dụ: *“****dám*** *nghĩ”, “****toan*** *nói”…*

#### Động từ tiếp thụ-Va

Đây là trạng thái mang tính chất thụ động. Có hai trạng thái chính là *bị* hoặc *phải* và *được*. Đằng sau các động từ này phải có phụ tố chỉ sự vật tiếp thụ.Ví dụ: *“****bị*** *mắng”, “****được*** *khen”…*

#### Động từ so sánh-Vc

Các sự vật có thể được so sánh đánh giá trong sự so sánh với các sự vật khác về một phương diện nhất định. Có ba trạng thái so sánh: bằng, hơn và kém. Các động từ biểu hiện cho các trạng thái đó được gọi là động từ so sánh. Cũng như hầu hết các loại động từ ở trên, khi các động từ này được dùng làm chính tố thì thường có phụ tố chỉ đối tượng đi kèm. Ví dụ: *“****bằng*** *nhau”, “****hơn*** *người”…*

#### Động từ đặc biệt: động từ “là”-Vz

Động từ “là” có ý nghĩa ngữ pháp to lớn trong ngữ pháp tiếng Việt. Đối với câu đơn bình thường, việc phân chia giữa hai loại câu tả và câu luận phụ thuộc vào sự xuất hiện của động từ “là”. Ví dụ: *“tôi* ***là*** *người lính đặc công”.*

### Tính từ(A)

Có thể phân từ loại tính từ thành các tiểu loại chính sau đây:

#### Tính từ hàm chất(Ai)

Khi tính từ loại này làm chính tố trong ngữ thì trước đó có thể xuất hiện các phụ tố chỉ mức độ. Chú ý là phía sau tính từ loại này, trong trường hợp này, loại phụ tố chỉ phạm vi thể hiện tính chất. Những từ loại này: *tốt, đẹp, xấu, thông minh, ngoan, ngu xuẩn…* Ví dụ: *“rất* ***giỏi*** *toán”.*

#### Tính từ hàm lượng(An)

Đó là những tính chất như: cao, thấp, ngắn, dài, rộng, hẹp, nông, sâu, xa, gần… Tính từ loại này thường đi kèm với phụ tố chỉ định lượng, hay để chỉ một cái mốc có tác dụng định lượng. Ví dụ: *“****cao*** *hai thước”, “****dài*** *một nghìn km”…*

### Phụ từ(R)

Có thể chia phụ từ thành các tiểu loại như sau :

#### Phụ từ thời gian(Rt)

Đây là các phụ từ biểu thị nghĩa ngữ pháp về thời gian. Đó là các từ: *đã, sẽ, đang, vừa, mới, sắp, từng, liền, bèn, rồi…*

#### Phụ từ mức độ(Rd)

Đây là các phụ từ biểu thị các nghĩa ngữ pháp về mức độ. Đó là các từ: *rất, khá, hơi, quá, lắm…*

#### Phụ từ so sánh(Rc)

Đây là các phụ từ biểu thị rằng hoạt động, trang thái hay tính chất diễn ra qua so sánh trong những điều kiện thời gian, không gian nhất định của một hoàn cảnh. Những phụ từ đó là: cũng, đều, vẫn, cứ, còn, liên tục, liên tiếp, không ngừng… Ví dụ: *“Mai và Lan* ***đều*** *học giỏi”.*

#### Phụ từ khẳng định – phủ định(RfY-RfN)

Đây là các phụ từ biểu thị nghĩa phủ định hay khẳng định. Nghĩa phủ định: *không, chẳng, chưa*. Nghĩa khẳng định: *có.* Ví dụ: *“tôi* ***không*** *có tiền”, “nó* ***có*** *nói dối”...*

#### Phụ từ mệnh lệnh(Ri)

Phụ từ biểu thị ý sai khiến, khuyên bảo, mời mọc, can ngăn. Ví dụ: *“em* ***đừng*** *đi về muộn”, “anh* ***nên*** *đi học đúng giờ”…*

### Kết từ(C)

Các tiểu loại từ của kết từ :

#### Kết từ chính phụ(Cm):

Đây là những kết từ biểu thị quan hệ chính phụ. Đó là những từ như: do, của, để, bởi, bởi vì… Ví dụ: “chúng tôi đã chiến đấu anh dũng như vậy **để** giành chiến thắng”.

#### Kết từ liên hợp(Cp):

Đây là các kết từ biểu thị quan hệ liên hợp. Đó có thể là các từ như và, với, hay, hoặc, cùng… hay các cặp như nếu … thì, tuy … nhưng. Ví dụ: “**nếu** trời mưa **thì** chúng tôi sẽ ở nhà”, “nó **không những** ngoan **mà còn** học giỏi”.

### Đại từ(P)

#### Đại từ sự vật

Đây là các đại từ dùng để chỉ sự vật, ta có thể sử dụng chúng như danh từ. Gồm ba loại: Đại từ xưng hô *(Pp: tôi, tao, mày, chúng mày, chúng nó…)*; đại từ không gian, thời gian *(Pd: đây, đấy, đó, kia, ấy…)*; đại từ số lượng *(Pn: bấy nhiêu).* Ví dụ: *“****chúng tôi*** *đang đến trường”.*

#### Đại từ hoạt động – tính chất(Pl)

Đây là các đại từ dùng để chỉ hoạt động, tính chất: *thế, vậy…* Ví dụ: *“****vậy*** *là hết!”*

#### Đại từ nghi vấn(Pi)

Các đại từ dùng để chỉ trong câu hỏi như ai, gì, chi, đâu, bao nhiều, sao, thế nào…

### Trợ từ (M)

Cảm từ(E): ái chà, dạ, vâng, ôi chao…

Loại từ(Nl): cái, con, cây, người, tấm…

Số từ(Nq): một, hai, ba, vài, dăm, mươi…

## Cụm từ tiếng việt

Ngữ là đơn vị ngữ pháp bậc trung gian giữa từ và câu [x].

Việc tìm hiểu cấu tạo cũng như các loại ngữ là cần thiết để tìm hiểu cấu tạo của câu. Qua cấu tạo của ngữ, có thể nhận rõ thêm đặc điểm ngữ pháp của từ loại và các tiểu loại.

#### Cụm danh từ(NP)

##### Khái niệm

Cụm danh từ là một tổ hợp từ có danh từ làm thành tố chính, các thành tố phụ đứng trước và sau bổ ngữ cho thành tố chính.Ví dụ: *Những bông hoa này*…

##### Cấu tạo

a) Thành tố chính:

* Thường là một danh từ chung.
* Danh từ chung kết hợp danh từ chỉ loại thể hoặc đơn vị.

Ví dụ: - Những học sinh này, hai quyển sách, năm cân đường.

b) Các thành tố phụ trước: gồm các từ:

* Từ chỉ đoàn thể: Tất cả, hết thảy…
* Từ chỉ số lượng: một, hai…vài, dăm, những…
* Từ chỉ loại thể: cái, cơn, chiếc, quyển…
* Từ chỉ đơn vị: cân, mét, thước…

c) Các thành tố phụ sau: rất đa dạng và phong phú. Về cấu tạo có thể là:

* 1 từ:
  + Từ chỉ định: ấy, kia, này, nọ…VD: Cái ghế này…
  + Từ chỉ tính chất, đặc trưng (thường là các tính từ) VD: đồng hồ vàng…; học sinh chuyên…
* 1 cụm từ: Thơ của *các em thiếu nhi*…
* 1 cụm chủ vị: Ngôi nhà *cha tôi vừa mới mua*…

##### Chức năng ngữ pháp của cụm danh từ

Cũng giống danh từ, cụm danh từ có thể làm chủ ngữ, vị ngữ, trạng ngữ, bổ ngữ, định ngữ.Ví dụ:

* **Lan**/ đang đọc truyện Đô-rê-môn
* **Học sinh trường Chu Văn An** / rất ngoan.

#### Cụm động từ(VP)

##### Khái niệm

Cụm động từ là các tổ hơp từ có động từ làm thành tố chính, các thành tố phụ đứng trước và sau bổ nghĩa cho thành tố chính. Ví dụ: *đang đọc sách*.

##### Cấu tạo

a)Thành tố chính:

Thường là một động từ. Khi có hai động từ đi liền nhau (động từ không đọc lập và động từ độc lập) có thể coi động từ thứ nhất là thành tố chính của động từ.

      Ví dụ: - đang học bài.

                - toan về quê

b)Các thành tố phụ đứng trước: Thường là các phụ từ:

* Những từ chỉ thời gian: đã, sẽ, đang, sắp, vừa…
* Những từ chỉ sự tiếp diễn: đều, cứ, vẫn, còn, lại…
* Những từ chỉ ý khẳng định, phủ định: không, chứa, chẳng…
* Những từ chỉ mệnh lệnh: hãy, đừng, chớ…
* Có thể có một hoặc nhiều phụ từ làm thành tố phụ trước:

     Ví dụ: cũng vẫn cứ đến lớp

c) Các thành tố phụ sau

* Về cấu tạo: Phần phụ sau có thể là một từ, một cụm từ hoặc một cụm chủ vị.

         Ví dụ: - học **bài** (1từ)

                 - Ăn **một cái bánh**  (1 cụm từ).

                - Mọi người / biết **anh ấy rất tích cực** ( cụm chủ vị. )

* Về ý nghĩa: thành tố phụ sau thường bổ nghĩa cho động từ chính.

##### Chức năng ngữ pháp của cụm động từ

Cụm động từ có thể làm chủ ngữ, vị ngữ, trạng ngữ, bổ ngữ, định ngữ.

* Chủ ngữ: Bảo vệ tổ quốc / là nghũa vụ của mỗi người.
* Vị ngữ: Mặt trời / đã lên cao.
* Trạng ngữ: Tan buổi họp, mọi người đều ra về.
* Bổ ngữ: Bộ đội / đi đánh giặc
* Định ngữ: Quyển sách mượn trên thư viện / rất hay.

#### Cụm tính từ(AP)

##### Khái niệm

Cụm tính từ là một tổ hợp từ có tính làm thành tố chính, các thành tố phụ đứng trước và sau bổ nghĩa cho thành tố chính: VD: vẫn đẹp mãi

##### Cấu tạo

a) Thành tố chính: Thường là các tính từ có mức độ.

Ví dụ: rất **xinh** đẹp

b) Các thành tố phụ trước:

Cũng như ở cụm động từ, thành tố phụ trước của cụm tính từ cũng có thể là các phụ từ chỉ thời gian, sự tiếp diễn, sự khẳng định hay phủ định và nhất là phụ từ chỉ mức dộ (rất, hơi, khá, quá…) trừ các phụ từ mệnh lệnh (hãy, đừng, chớ)

Ví dụ: **vấn** tốt, **còn** đẹp, **rất** hiền

c) Các thành tố phụ sau:

* Về cấu tạo: Phần phụ sau có thể là một từ, một cụm từ một cụm chủ vị

Ví dụ: - ngoan **lắm** (1 từ)

                    - rộng **ba trăm mét** (1 cụm từ)

                    - đẹp như **trăng mới mọc** ( 1 cụm C-V)

* Về ý nghĩa: các thành tố phụ sau của cụm tính từ thường bổ sung ý nghĩa cho tính từ làm thành tố chính.

##### Chức năng ngữ pháp

Cụm tính từ cũng có thể làm chủ ngữ, vị ngữ, trạng ngữ, định ngữ, bổ ngữ.

Ví dụ: - Lợi cho tập thể tức là lợi cho cá nhân. (chủ ngữ)

- Nó / nhanh như sóc. (vị ngữ)

- Nhanh vùn vụt, đoàn tàu chạy về hướng Nam. (trạng ngữ)

- Màu xanh mơn mởn của lá / làm dịu cả trưa hè. (định ngữ)

- Cám / rất muốn xinh như tấm. (bổ ngữ)

## Các kiểu câu của tiếng việt

**Câu trần thuật(S):** là kiểu câu nổi tiếng nhất, dùng để miêu tả, nhận định về một sự kiện. Ví dụ: *“Tôi đang làm đồ án tốt nghiệp tại nhà”.*

**Câu nghi vấn(SQ)**:Là kiểu câu nêu nội dung hoài nghi để được giải đáp. Ví dụ: *“Cậu làm xong đồ án chưa”.*

**Câu cầu khiến(SC)**: là kiểu câu nhằm đòi hỏi thực hiện một hành động, một chuyển biến. Ví dụ: *“Phải nộp đồ án vào sáng nay”.*

**Câu cảm thán(SE)**: câu cảm thán dùng để bộc lộ tình cảm, cảm xúc. Ví dụ: *“ối ông trời ơi!”.*

## Mô hình PCFG và giải thuật phân tích cú pháp CYK.

### Mô hình văn phạm CFG.

Mô hình CFG là một mô hình sử dụng văn phạm phi ngữ cảnh để biểu diễn tập luật cú pháp. Đối với ngôn ngữ tự nhiên, mà đặc biệt là tiếng Việt, một loại ngôn ngữ có tính phức tạp rất cao về mặt ngữ pháp thì sự tự do và ít bị ràng buộc của văn phạm phi ngữ cảnh là một lựa chọn rất hợp lí.

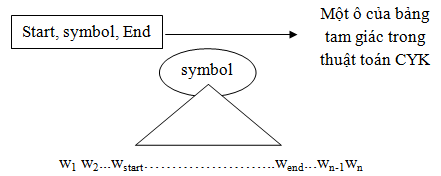
Văn phạm là một hệ thống G = (N,T,S,P) trong đó:

* N là tập hữu hạn các ký hiệu, gọi là các ký hiệu không kết thúc hay biến.
* T là tập hữu hạn các ký hiệu , gọi là các ký hiệu kết thúc.
* S thuộc N là tập ký hiệu bắt đầu.
* P là tập hữu hạn các luật có dạng X→Y, trong đó :
  + - X V\*NV\* với V = N T.
    - Y V\*

### Thuật toán CYK.

Ý tưởng của thuật toán này là xây dựng cây phân tích cú pháp bằng cách điền đầy một bảng tam giác kích thước (n-1)\*(n-1) với n là số từ vựng đầu vào.

Mỗi ô của bảng tam giác gồm 3 thông số: vị trí bắt đầu chuỗi sinh, ký hiệu sinh, vị trí kết thúc chuỗi sinh.



Hình ‑. Một phần tử trong bảng CYK.

#### Thuật toán tạo bảng của CYK.

Đầu vào: văn phạm G = (N,T,S,P) ở dạng chuẩn Chomsky, không chứa sản xuất trống, xâu vào ω, = a1a2...an ∈ T+.

Đầu ra: Bảng phân tích T đối với ω sao cho tij chứa A khi và chỉ khi

**A ⇒+ aiai+1...ai+j-1**

1. Tập hợp ti1 = {A | A → ai ∈ P} , i =1..n. Sau bước này nếu ti1 chứa A thì rõ ràng ta có A⇒+ ai.
2. Giả sử tij đã tính với ∀i ( 1 ≤ i ≤ n) và với ∀j' (1≤ j' <j)

Xét một kí hiệu không kết thúc A, nếu tồn tại một suy dẫn

**A → BC**

mà B ∈ tik và C ∈ ti+k,j-k thì ta thấy rõ ràng A ∈ tij.

1. Lặp lại bước trên cho tới khi tij đã được tính với 1 ≤ i ≤ n, 1 ≤ j < n-i+1.

#### Ví dụ minh họa cho thuật toán CYK.

Phân tích câu : “anh ấy rất ngầu”

Với tập luật :

* + S → NP AP; S → N AP;
  + S → P AP; NP → N P;
  + NP → N N ; AP → R A;

Bước 1: tại hàng i=1, gán nhãn cho từ: riêng với trường hợp của từ “anh” có hai nhãn là A or N.

Bước 2: tại hàng i=2, do có luật NP→N P, và N sinh ra vị trí(1,2) kết hợp được với vị trí(2,3) của P nên gom được N và P thành NP(1,3).Tương tự như thế ta có với trường hợp của AP→A R:

Bước 3: S(2,5) → P(2,3) AP(3,5)

Bước 4: S(1,5) → NP(1,3) AP(3,5)

Đến đây gặp được ký hiệu bắt đầu S, thành công!!!!

Bảng ‑.Phân tích CYK cho câu “anh ấy rất ngầu”

|  |  |
| --- | --- |
|  |  |
|  |  |

#### Thuật toán CYK cải tiến

Một nhược điểm của thuật toán CYK có thể thấy rõ đó là chỉ áp dụng được với tập luật ở dạng chuẩn Chomsky, có nghĩa là vế phải của luật luôn luôn nhỏ hơn hoặc bằng 2 ký hiệu. Trong khi đó, tập luật của chúng ta có rất nhiều luật vế phải nhiều hơn 3 ký hiệu, ví dụ như:

S → ABCD.

Giải pháp đề xuất cho trường hợp này là, thay vì mỗi ô của bảng chỉ có 3 tham số, ta thêm một tham số wait ở cuối, chính là phần vế phải còn thiếu để sử dụng được luật này (một số nơi gọi là tham số cho vay).

Ví dụ với luật

S → A B C D

Thì ta sẽ gộp như sau với trường hợp của A và B để thành S (wait = “CD”) như hình minh họa bên dưới:

Thông số “CD” trong ngoặc cho thấy còn thiếu hai phần tử CD nữa thì mới hoàn thành được luật nói trên.

Sử dụng phần còn thiếu đó để kết hợp các ký hiệu như sau: nếu một ký hiệu B là ký hiệu bắt đầu tập wait của A và vị trí chuỗi sinh của A và B khớp nhau, thì A và B sẽ được gộp lại thành ký hiệu A với thông số wait = {wait(A)/B}. Cứ gộp như thế cho đến khi ta gặp được ký hiệu bắt đầu (S,wait=“”) ở ô trên cùng của bảng CYK thì thuật toán thành công.

Lấy một ví dụ: Ta xét ví dụ nhỏ : phân tích câu “anh múa kiếm và vụt côn” với tập luật:

* + S → N VP;
  + S → NP VP
  + VP → V N;
  + VP → VP C P

Ta có bảng phân tích CYK của ví dụ trên sẽ như sau :

Hình ‑. CYK cải tiến với câu"anh múa kiếm và vụt côn".

#### Những khó khăn của thuật toán CYK

Vì thuật toán CYK xét qua tất cả các trường hợp có thể kết hợp với nhau dựa vào tập luật cú pháp (khoảng 1000 luật), nên rất dễ dẫn đến hiện tượng bùng nổ tổ hợp khiến hệ thống bị quá tải hoặc chạy rất chậm.

Ví dụ, nếu có hai ký hiệu có nhãn là N-danh từ ở vị trí thích hợp N(2,7) và N(7,8) được kết hợp với nhau thì số lượng ký hiệu được tạo ra bởi sự kết hợp của chúng là 64, một con số quá lớn!

Hơn nữa, kể cả khi đã phân tích câu xong, sẽ có rất nhiều cây phân tích cú pháp được đưa ra khiến bộ phân tích không thể biết được cây nào là đầu ra chính xác. Chính vì vậy, khó khăn ở đây là làm sao cải thiện được độ phức tạp của thuật toán cũng như khử đi sự nhập nhằng về mặt cấu trúc của cây phân tích cú pháp.

## Mô hình xác suất PCFG

### Định nghĩa cơ bản về PCFG

Một mô hình xác suất văn phạm phi ngữ cảnh (PCFG) được định nghĩa đơn giản là mô hình CFG mà ở đó mỗi một luật sinh Ni  j sẽ được gán kèm thêm với một xác suất P(Ni  j) tương ứng. Và các xác suất này phải thỏa mãn điều kiện: i  j ) =1

### Các loại xác suất trong PCFG

#### Xác suất trong (inside)

Xác suất trong của một nút Njpq (nút Nj sinh ra đoạn từ vị trí p đến q trong câu) được tính bằng quy nạp dựa trên xác suất các nút con của nó, có thể hiểu một cách trừu tượng xác suất trong là giá trị hiện thời của nút đó và được ký hiệu là βj(p,q). Chúng ta vẫn thường hay gọi vui đây là xác suất trong nhà của một nút, là phép đo lường những giá trị bên trong của nút.

Trường hợp cơ sở : Cần tính βj(k,k) ( xác suất của luật Nj  wk):

βj(k,k) = P ( wk | Njkk,G) = P ( Nj  wk|G)

**Phương pháp quy nạp:** Ta muốn tính βj(p,q), với p<q. Bởi vì điều này bước quy nạp sử dụng một văn phạm theo chuẩn Chomsky, luật đầu tiên phải theo dạng Nj  Nr Ns, do đó chúng ta có thể thực hiện theo quy nạp, phân chia xâu đầu vào các phần nhỏ hơn và lấy tổng các kết quả :

Hình ‑. Hình ảnh minh họa của xác suất inside.

Do đó, 

βj (p,q) = P (wpq| Njpq, G)

= (wpd, Nrpd, w(d+1)q, Ns(d+1)q| Njpq, G)

= ( Nrpd, Ns(d+1)q| Njpq, G) *P*(wpd | Njpq, Nrpd, Ns(d+1)q, G)

 *P*(w(d+1)q | Njpq, Nrpd, Ns(d+1)q,wpd, G)

= ( Nrpd, Ns(d+1)q| Njpq, G)*P*(wpd | Nrpd, G)

 *P*(w(d+1)q | Ns(d+1)q, G)

= (Nj  Nr Ns) βr(p,d) βs(d+1,q)

Công thức cuối cùng của xác suất trong tại nút Njpq bất kì là

βj (p,q) = (Nj  Nr Ns) βr(p,d) βs(d+1,q)

Trên đây, đầu tiên chúng ta đã chia xác suất cần tìm thành tích các xác suất cấp nhỏ hơn, và trong công tính tính xác suất trên nhờ việc áp dụng các giả thiết phi ngữ cảnh của PCFG ở trên, nên ta có thể biểu diễn biểu thức tính cuối cùng thành một công thức định nghĩa cho các xác suất trong. Bằng việc sử dụng biểu thức dạng truy hồi này- xác suất của một thành phần được biểu diễn bởi xác suất của các thành phần nhỏ hơn, các xác suất trong có thể được áp dụng một cách hiệu quả khi tính toán với thuật toán bottom-up.

#### Xác suất ngoài (outside)

Xác suất ngoài của một nút là xác suất hứa hẹn đi đến đích của nút đó, hay chính là xác suất nút đó liên kết với nút gốc. Nói một cách dễ hiểu thì xác suất outside là khả năng đối ngoại của một nút.

Các xác suất ngoài được tính từ trên xuống. Sử dụng phép quy nạp của xác suất ngoài yêu cầu sử dụng đến các xác suất trong, do đó chúng ta tính toán các xác suất ngoài sau khi tính các xác suất trong, sử dụng giải thuật ngoài.

Trường hợp cơ bản: Trường hợp cơ bản là xác suất của cây với gốc là ký hiệu không kết thúc Ni :

α1(1,m) = 1

αj (1,m) = 0 với j 

Trường hợp quy nạp : Với trường hợp của nút Nf sinh ra Nj và Ng, nút Nj có thể nằm ở phía bên trái hoặc bên phải Ng. Chúng ta lấy tổng của cả hai xác suất nhưng loại trừ

αj(p,q) = +



### Cách khử nhập nhằng mặt cú pháp với PCFG

Rất dễ dàng tính được xác suất của một cây phân tích trong mô hình PCFG : bằng tích tất cả các xác suất của các luật đã được sử dụng trong cây phân tích đó. Trong trường hợp bộ phân tích cú pháp cho ra nhiều cây phân tích cú pháp như ở trên đã nói, PCFG sẽ đưa ra cây nào có xác suất cao nhất làm kết quả đầu ra.

Xét một ví dụ cụ thể của PCFG :

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S |  | NP VP | 1.0 |  | NP |  | NP PP | 0.4 |
| PP |  | P NP | 1.0 |  | NP |  | *Gia đình* | 0.1 |
| VP |  | V NP | 0.7 |  | NP |  | *sổ liên lạc* | 0.18 |
| VP |  | VP PP | 0.3 |  | NP |  | *theo dõi* | 0.04 |
| P |  | *bằng* | 1.0 |  | NP |  | *con* | 0.18 |
| V |  | *theo dõi* | 1.0 |  | NP |  | *điểm* | 0.1 |

Bảng trên mô tả các luật của văn phạm cùng các xác suất tương ứng của chúng. Như vậy, với một bộ PCFG như trên và với một câu cần phân tích là “***Gia đình theo dõi con bằng sổ liên lạc****”*  thì chúng ta có 2 cây phân tích cùng với xác suất của chúng:

(t1)

Hình ‑. Cây phân tích t1.

Xác suất của cây phân tích t1 này là :

**P(t1)** = 1.0 0.10.71.00.40.181.01.00.18= 0.0009072

(t2)

Hình ‑. Cây phân tích t2.

Xác suất của cây phân tích t2 này là :

**P(t2)** = 1.0 0.10.30.71.01.0  0.18 1.00.18= 0.0006804

Do P(t1)>P(t2) nên kết quả đầu ra sẽ là cây t1.

Như vậy ta có thể nói rằng, mục đích của bộ phân tích cú pháp sử dụng văn phạm PCFG là đưa ra được cây phân tích cú pháp có xác suất cao nhất hay nút S có inside lớn nhất.

**Kết chương**

Chương này đồ án đã trình bày về tiếng việt và cơ sở lý thuyết để phân tích cú pháp tiếng việt với thuật toán CYK cải tiến và mô hình PCFG. Chương sau, đồ án sẽ trình bày về các cải tiến cũng như những ý tưởng, đề xuất áp dụng cho phân tích cú pháp tiếng Việt.

# CÁC ĐỀ XUẤT CỦA ĐỒ ÁN CHO PHÂN TÍCH CÚ PHÁP TIẾNG VIỆT

*Chương này đồ án sẽ trình bày về những giải pháp mà đồ án sẽ sử dụng trong phân tích cú pháp tiếng Việt dựa trên mô hình PCFG:*

* *Cải tiến tốc độ của hệ thống phân tích cú pháp với thuật toán Beam Search áp dụng cho thuật toán CYK.*
* *Tối ưu hóa đầu ra của hệ thống phân tích cú pháp, khắc phục những nhược điểm của thuật toán CYK-Beam Search bằng giải thuật tìm kiếm A\*.*
* *Cải tiến tốc độ của bộ phân tích cú pháp A\* bằng một sáng kiến của chính đồ án: giải thuật lelightwin!*

## Nhận xét về mô hình PCFG và thuật toán CYK cải tiến ở chương 2 cho tiếng Việt.

Với mô hình PCFG, chúng ta có thể khử được nhập nhằng giữa các cây phân tích cú pháp đầu ra bằng cách cho ra cây phân tích cú pháp có giá trị xác suất cao nhất. Tuy nhiên, nó vẫn còn những vấn đề mà chúng ta cần phải xem xét:

* Trong tổng hợp tiếng nói, vấn đề tốc độ cũng là một yếu tố cần được xem trọng, phân tích cú pháp cũng không ngoại lệ. Tuy nhiên với thuật toán CYK cải tiến thì vấn đề tốc độ có vẻ khá nan giải. Số lượng tổ hợp bùng nổ trong quá trình phân tích đã khiến cho thuật toán sinh bảng kiểu vét cạn này mất rất nhiều thời gian để cho ra kết quả.
* Đặc biệt là với tiếng việt, một ngôn ngữ với một bộ luật cú pháp rất phức tạp thì số lượng tổ hợp bùng nổ là rất lớn. Với những trường hợp của câu dài và khó, thuật toán CYK có thể bị tràn bộ nhớ hoặc phải mất khoảng vài tiếng để phân tích xong, và còn cho ra quá nhiều cây phân tích cú pháp, kể cả khi đã có PCFG khử nhập nhằng thì đây vẫn là một bài toán khó.

Chính vì vậy, trong phần tiếp theo, đồ án sẽ trình bày một số những phương pháp cải tiến để giúp bộ phân tích cú pháp vừa có thể tận dụng lợi thế của PCFG lại vừa có thể cải tiến được tốc độ của hệ thống phân tích cú pháp.

## Cải tiến thuật toán phân tích cú pháp CYK bằng giải thuật tìm kiếm beam search

### Thuật toán tìm kiếm beam search

Với một nút, xác suất trong của một nút là khả năng đối nội trong vùng mà nó bao phủ, còn xác suất ngoài của một nút là khả năng ngoại giao, là khả năng nó kết hợp với vùng bên ngoài vùng bao phủ của nó và dẫn đến kết quả. Cho nên, nếu kết hợp cả hai xác suất này lại nó sẽ trở thành tiêu chí để đánh giá xem nút nào có giá trị hơn về mặt thuật toán. Cụ thể:

P(nút) = inside(nút) + outside(nút). (1)

Nút nào có P(nút) càng lớn thì nút đó sẽ càng là ứng cử viên nặng ký cần phải xét duyệt. Ngược lại, nút nào có P(nút) bé thì đó rất có thể sẽ là một nút thừa, không thể hướng đến đích.

Chính vì lẽ đó, để giảm bớt số lượng các phương án cần xét duyệt, đồ án quyết định sử dụng phương pháp beam search. Với phương pháp này tại mỗi bước của thuật toán CYK, ta chỉ xét một số lượng phương án n cho trước, với n được gọi là bề rộng của beam.

Ý tưởng của thuật toán beam search áp dụng cho phân tích cú pháp: với mỗi một bước lặp của thuật toán, chúng ta sẽ sắp xếp các nút hay các ứng cử viên thứ tự từ cao đến thấp của p(nút) và chỉ lấy n ứng cử viên sáng nhất, những ứng cử viên thấp hơn sẽ bị loại đi. Bề rộng này có thể thay đổi hoặc không thay đổi tùy vào sự tinh tế của người sử dụng.

Giả sử tại một bước có n ứng cử viên và max là nút ứng cử viên sang giá nhất

Nếu P(max) –P(núti) > threshold thì loại bỏ nút i với i=1,n

Trực quan về thuật toán beam search có thể được mô phỏng bởi hình vẽ dưới đây, với các nút xanh là các nút được xét, còn các nút màu đen là các nút bị cắt tỉa đi bởi beam search.

Hình ‑. Minh họa vui cho thuật toán cắt tỉa beam search.

Như vậy, trong mỗi bước của CYK, chúng ta có thể loại đi một cơ số những phương án được cho là không thể dẫn đến đích. Số lượng tổ hợp bùng nổ cũng vì thế mà giảm đi rất nhiều.

### Quy trình áp dụng và thực hiện thuật toán beam search

Như đã trình bày ở trên, thuật toán beam search phụ thuộc rất nhiều và ước lượng đánh giá cho mỗi nút. Nếu hàm ước lượng tính toán có thể khiến cho việc các nút dẫn đến cây phân tích cú pháp có xác suất cao nhất bị cắt tỉa mất, dẫn đến việc kết quả đầu ra không chính xác. Mặc dù đã có công thức như (1), nhưng từ công thức đi đến quá trình thực hiện là cả một vấn đề.

#### Cách tính inside trong beam search

Với trường hợp của một nút A bất kỳ nào đấy, A có thể được tạo nên từ nhiều hơn một nhánh cây cú pháp. Điều này nghe có vẻ khó hiểu, nhưng chúng ta thử tưởng tượng trong tập các nút được xét có tồn tại bốn nút B C D E có thể kết hợp với nhau theo vị trí thành hai cặp (B,C) và (D,E) và trong tập luật cú pháp tồn tại hai luật {A→ B C} và {A→ D E}, vậy thì với phân tích của riêng nút A ta đã có hai nhánh:

Theo như công thức tính xác suất inside theo mô hình PCFG đề xuất ở trên thì công thức sẽ là :

*inside(A) = P(A→B C)\*inside(B)\*inside(C)+P(A→D E)\*inside(D)\*inside(E)*

Tuy nhiên, thuật toán phân tích cú pháp của chúng ta hướng đến một và chỉ một cây phân tích có xác suất lớn nhất nên đồ án quyết định sử dụng phương án tính inside của một nút bằng xác suất lớn nhất của nhánh tạo nên nút đó. Cụ thể:

*PA→B C = P(A→B C)\*inside(B)\*inside(C),*

*PA→D E = P(A→D E)\*inside(D)\*inside(E),*

Thì *inside(A) = max(PA→B C , PA→D E )*

Vậy công thức tổng quát của chúng ta để tính inside của một nút sẽ là :

*βj (p,q) = max(P(Nj  Nr Ns) βr(p,d) βs(d+1,q))*

Tuy nhiên đó mới chỉ dừng lại ở mặt công thức, khi tính inside của một nút S bất kì vẫn còn một chút khó khăn. Để cho dễ hiểu, ta sẽ giả sử S được tạo thành từ luật S→ NP NP VP ,ta có hai trường hợp cần phải xét:

* Nút S là nút hoàn chỉnh, nghĩa là biến wait của nó bằng rỗng

Trong trường hợp này chúng ta sử dụng y nguyên công thức như vừa nãy, tức là :

*inside(S(wait=""))=lg(P(S→NP NP VP))+ inside(NP)+inside(NP)+ inside (VP)*

Ở đây chúng ta sử dụng log thay vì phép nhân như ở công thức tổng quát vì phép nhân các xác suất khiến cho kết quả tính ra rất nhỏ, gần như bằng 0. Điều này dẫn đến một số trở ngại trong việc so sánh và tính toán.

Hình ‑. Hình ảnh của một nút phân tích hoàn chỉnh.

* Nút S là nút chưa hoàn chỉnh, hay nói cách khác nó là một nút giả, biến wait của S khác rỗng .

Trong trường hợp này, chúng ta có thể hình dung đây là nút S(wait= “VP”) được tạo nên từ hai phần tử NP NP bằng luật S→ VP NP NP. Vậy công thức inside trong trường hợp này sẽ là :

*inside(S(wait="VP"))=lg(P(S→NP NP VP))+ inside(NP)+inside(NP)(1)*

Hình ‑. Hình ảnh của một nút phân tích sử dụng biến wait.

Nếu sau đó nút S(wait= “VP”) gặp được phần tử VP ở vị trí tương ứng thì nó sẽ kết hợp với VP để thành một nút S mới với inside là :

*inside(S(wait=""))= inside(S(wait="VP"))+inside(NP)(2)*

Từ (1) và (2) ta có thể suy ra:

*inside(S(wait=""))=lg(P(S→NP NP VP))+ inside(NP)+inside(NP)+ inside (VP)*

Công thức chính là công thức tính inside của một nút hoàn chỉnh. Vậy cách làm được đề xuất trong trường hợp nút giả là chấp nhận được.

#### Cách tính xác suất outside trong beam search

Tính inside của một nút được trình bày ở bên trên nghe có vẻ phức tạp nhưng thực ra lại rất dễ dàng. Vì giải thuật inside cũng được thực hiện dựa theo chiến lược bottom-up y như thuật toán CYK. Nhưng với outside, sự khó khăn trong việc tính toán đã ở một cấp độ khác hoàn toàn. Vì giải thuật outside được tính toán dựa theo chiến lược top-down, nghĩa là phải xuất phát từ những ô ở hàng trên trong bảng CYK thì mới có thể tính được những ô hàng dưới, điều này ngược hẳn với inside.

Hình ‑. Hình ảnh mô phỏng outside của một nút.

Hãy nhìn vào hình vẽ min họa bên trên, đây là một cây phân tích cú pháp hoàn chỉnh, và nút số 4 đang là nút được xét. Bây giờ nếu muốn tính được outside của nút số 4 chúng ta phải sử dụng đến toàn bộ phần được tô đậm. Cụ thể:

*Outside(4) = lg(2→ 4 5) +Outside(2) + inside(5).*

Outside(2) được tính quy nạp theo Outside(4):

*Outside(2) = lg(1→ 2 3) +Outside(1) + inside(3).*

*Outside(1) = 1 nếu {1} là S .*

*=0 nếu {1} không phải là S.*

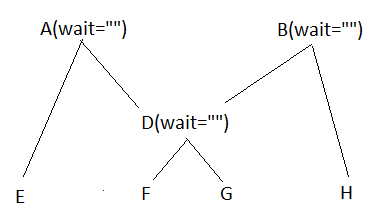
Như vậy muốn tính được outside của một nút ta phải biết được toàn bộ nhánh cây phân tích cú pháp ở mức trên, điều này là không thể với giải thuật CYK!

Để giải quyết khó khăn của bài toán tính outside, người làm đồ án đã đề xuất ra hai từ “tương đối”. Tại sao lại là tương đối? Vì chúng ta sẽ không tính outside thật sự, chúng ta chỉ tính outside ở mức tương đối. Nghĩa là thay vì tính outside sử dụng toàn bộ các nút ở mức trên thì chúng ta chỉ sử dụng các nút ở trên nút được xét 1 mức để tính outside. Khi muốn tính outside của các ô ở bước i, chúng ta sẽ tạm thời bỏ trống outside của các ô này trong bước i. Khi đến bước i+1, chúng ta mới sử dụng các ô được tạo ra ở bước này để tính outside cho các ô ở bước i. Nói một cách ngắn gọn, là chúng ta sử dụng các ô ở hàng trên để tính outside cho các ô ở hàng dưới.

Cụ thể, cách tính outside cũng được chia làm hai trường hợp như inside:

* + Nút được xét là nút hoàn chỉnh, ví dụ như trường hợp của nút D dưới đây, thì công thức tính outside của nút D sẽ là :

*Outside(D) = lg(P(A → E D)) \* 10^inside(E) +lg(P(B → D H)) \* 10^inside(H)*



Hình ‑. Outside của trường hợp nút hoàn chỉnh.

* + Nút được xét là nút chưa hoàn chỉnh, lại vẫn trường hợp của nút D ở trên, được tạo ra bởi hai nút F G nhưng luật hoàn chỉnh là D→ F G H nên biến wait của D sẽ vẫn còn lại H. Lúc này công thức tính outside của D sẽ là:

*Outside(D) = lg(P(D → F G H)) \* 10^inside(H)*

Hình ‑. Outside của trường hợp nút chưa hoàn chỉnh.

#### Quy trình thực hiện thuật toán beam search

Quy trình thực hiện thuật toán beam search gồm các bước như sau :

* + Khởi tạo các ô ở hàng 1 (là các từ trong câu đã được gán nhãn).
  + Tại hàng thứ i, làm các công việc sau:
    - Tạo ngưỡng B để làm biên cắt tỉa (tùy chỉnh cho phù hợp).
    - sử dụng các phần tử tạo ra trong các ô được để tính outside cho các phần tử trong các ô ở hàng thứ i-1.
    - Với các phần tử ở hàng i-1 đã có cả outside và inside, tính heuristic cho các phần tử đó, heuristic = inside + outside của một phần tử. Sau đó chọn ra phần tử có heuristic lớn nhất ký hiệu là max.
    - Với mỗi phần tử j ở hàng i-1 kiểm tra nếu :

*Heuristic(max)-heuristic(j)>ngưỡng B*

thì loại j ra khỏi danh sách được xét, đồng thời loại luôn cả các phần tử thuộc hàng i được tạo ra bởi j.

* + Tiếp tục lặp lại bước 2 cho đến khi điền đầy bảng phân tích của thuật toán CYK.

### Nhận xét về thuật toán beam search

Beam search thực chất là một thuật toán tìm kiếm đường đi phổ biến trong trí tuệ nhân tạo. Phương thức tìm kiếm dựa trên hàm ước lượng của nó có thể cho kết quả rất tốt và cải thiện đáng kể về mặt tốc độ. Tuy nhiên, cũng như bao thuật toán khác, nó cũng có những nhược điểm và những ưu điểm mà chúng ta cần phải xem xét.

Ưu điểm: thuật toán beam search được đề xuất trong đồ án có ưu điểm là nhanh, không quá phức tạp trong khâu tính toán outside như một số các thuật toán vì mục đích của nó chỉ là cắt tỉa đi những ứng cử viên tồi nhất chứ không phải là lấy ra ứng cử viên sáng giá nhất nên đòi hỏi outside không cần phải chính xác hoàn toàn. Sử dụng thuật toán beam search có thể giúp cho bộ phân tích cú pháp cải thiện rất đáng kể về tốc độ. Nếu sử dụng thuật toán CYK kết hợp với beam search, thì kể cả với một câu có độ phức tạp là 40 từ (chưa kể số âm tiết), bộ phân tích cú pháp chỉ mất vài phút là có thể cho ra kết quả. Bình thường nếu sử dụng thuật toán CYK vét cạn bảng, thời gian bộ phân tích cú pháp cho ra kết quả có thể tới vài tiếng!

Nhược điểm: Tuy nói là có cải thiện về mặt tốc độ nhưng thực ra thuật toán beam search không phải là một thuật toán tối ưu. Nếu chúng ta cho ngưỡng quá nhỏ, beam search sẽ có thể cắt tỉa mất cả phương án tối ưu, còn nếu để đảm bảo rằng beam search sẽ không cắt tỉa mất phương án tối ưu thì chỉ còn cách là tăng ngưỡng lên cực lớn, mà điều này cũng gần tiến tới việc sử dụng thuật toán vét cạn! Nói tóm lại là không có gì đảm bảo rằng thuật toán beam search sẽ giúp bộ phân tích cú pháp đưa ra được đầu ra tối ưu cả.

## Thuật toán tìm kiếm A\*

Trong trí tuệ nhân tạo, thuật toán A\* thuộc họ thuật toán Best-first-search là một thuật toán được biết đến rộng rãi nhất. Là một thuật toán thường được sử dụng rất nhiều trong các bài toán tìm kiếm, A\* xây dựng tăng dần tất cả các tuyến đường từ điểm xuất phát cho đến khi nó tìm thấy điểm kết thúc.

Best-first-search nói chung đều tìm đường đi bằng ước lượng heuristic tức là thuật toán này sẽ đi theo một con đường có vẻ dẫn về đích nhất. A\* cũng giống như vậy, nhưng điểm khác biệt của thuật toán A\* so với các thuật toán khác thuộc họ BFS là A\* có xét đến cả quãng đường đã đi qua, điều này khiến cho thuật toán A\* trở nên “tối ưu” và “đầy đủ”. Tối ưu nghĩa là nếu có một đường đi ngắn nhất dẫn đến đích, A\* chắc chắn sẽ tìm ra được. Đầy đủ nghĩa là A\* sẽ luôn tìm thấy lời giải nếu như bài toán có lời giải. Hay nói một cách khác, chỉ cần một bài toán có lời giải, A\* chắc chắn có thể tìm ra được lời giải tốt nhất!

### Thuật toán A\*

A\* lưu giữ một tập các lời giải chưa hoàn chỉnh, nghĩa là các đường đi qua đồ thị, bắt đầu từ nút xuất phát. Tập lời giải này được lưu trong một [hàng đợi ưu tiên](http://vi.wikipedia.org/w/index.php?title=H%C3%A0ng_%C4%91%E1%BB%A3i_%C6%B0u_ti%C3%AAn&action=edit&redlink=1) (*priority queue*). Thứ tự ưu tiên gán cho một đường đi *x* được quyết định bởi hàm *f*(*x*) = *g*(*x*) + *h*(*x*).

Trong đó, *g*(*x*) là chi phí của đường đi cho đến thời điểm hiện tại, nghĩa là tổng trọng số của các cạnh đã đi qua. *h*(*x*) là hàm đánh giá heuristic về chi phí nhỏ nhất để đến đích từ *x*.

Trong đó, việc lựa chọn h(x) rất quan trọng, nó quyết định sự hiệu quả của thuật toán A\*. Hàm ước lượng h(x) có 1 điều kiện là phải thỏa mãn h(x)<h’(x) với h’(x) là chi phí thực sự đến đích từ x.

Ví dụ, nếu "chi phí" được tính là khoảng cách đã đi qua, [khoảng cách đường chim bay](http://vi.wikipedia.org/w/index.php?title=Kho%E1%BA%A3ng_c%C3%A1ch_%C4%91%C6%B0%E1%BB%9Dng_chim_bay&action=edit&redlink=1) giữa hai điểm trên một bản đồ là một đánh giá heuristic cho khoảng cách còn phải đi tiếp.

Hàm *f*(*x*) có giá trị càng thấp thì độ ưu tiên của *x* *càng cao (do đó có thể sử dụng một cấu trúc* [*heap tối thiểu*](http://vi.wikipedia.org/w/index.php?title=Heap_%28c%E1%BA%A5u_tr%C3%BAc_d%E1%BB%AF_li%E1%BB%87u%29&action=edit&redlink=1) *để cài đặt hàng đợi ưu tiên này).*

**function** A\*(điểm\_xuất\_phát,đích)

**var** đóng := *tập rỗng*

**var** q := tạo\_hàng\_đợi(tạo\_đường\_đi(điểm\_xuất\_phát))

**while** q *không phải tập rỗng*

**var** p := lấy\_phần\_tử\_đầu\_tiên(q)

**var** x := *nút cuối cùng của p*

**if** x **in** đóng

**continue**

**if** x = đích

**return** p

*bổ sung x vào tập đóng*

**foreach** y **in** các\_đường\_đi\_tiếp\_theo(p)

đưa\_vào\_hàng\_đợi(q, y)

**return** failure

Trong đó, các\_đường\_đi\_tiếp\_theo(p) trả về tập hợp các đường đi tạo bởi việc kéo dài p thêm một nút kề cạnh. Giả thiết rằng hàng đợi được sắp xếp tự động bởi giá trị của hàm *f*.

"Tập hợp đóng" (đóng) lưu giữ tất cả các *nút cuối cùng của p* (các nút mà các đường đi mới đã được mở rộng tại đó) để tránh việc lặp lại các chu trình (việc này cho ra thuật toán tìm kiếm theo đồ thị). Đôi khi hàng đợi được gọi một cách tương ứng là "tập mở". Tập đóng có thể được bỏ qua (ta thu được thuật toán tìm kiếm theo cây) nếu ta đảm bảo được rằng tồn tại một lời giải hoặc nếu hàm các\_đường\_đi\_tiếp\_theo được chỉnh để loại bỏ các chu trình.

### Thuật toán A\* trong phân tích cú pháp

Như đã phân tích ở trên, đối với một bài toán tìm kiếm, chỉ cần có lời giải, A\* chắc chắn có thể đưa ra được lời giải tối ưu. Tham chiếu sang trường hợp của phân tích cú pháp, A\* có thể đảm bảo đưa ra được cây phân tích cú pháp tốt nhất. Tạm thời trong phạm vi của đồ án, đồ án chỉ có thể đáp ứng cái “tốt nhất” của một cây phân tích cú pháp ở mức là cây sẽ có xác suất cao nhất.

A\* trong phân tích cú pháp ngược lại với A\* tự nhiên. Thay vì tìm theo ưu tiên nhỏ nhất, chúng ta lại tìm đường theo ưu tiên lớn nhất vì cái chúng ta cần là cây phân tích cú pháp có xác suất cao nhất. A\* được thực hiện trên các phần tử cơ bản được gọi là element gồm 3 thông số như của CYK: {nhãn, start, end}. Về mặt giải thuật, A\* có hai tập hợp, được gọi là AGENDA và CHART. Trong đó AGENDA là tập các element đang chờ được xem xét, còn CHART là tập các element đã được xét. Thuật toán A\* gồm các bước như sau:

* + Bước 1: Đầu tiên, các từ trong câu sẽ được gán nhãn rồi tất cả sẽ được đẩy hết vào trong AGENDA. Ví dụ, từ thứ i trong câu với nhãn ci sẽ biến thành phần tử ci[i,i+1] rồi được thêm vào AGENDA .
  + Bước 2: Sau đó lặp đi lặp lại các công việc sau:
    - Lấy một phần tử có ước lượng ưu tiên cao nhất ra khỏi AGENDA để xét.
    - Nếu ứng cử viên lấy ra khỏi AGENDA vẫn chưa có trong CHART thì kết hợp nó với từng phần tử trong CHART dựa vào tập luật cú pháp. Tất cả các phần tử mới được tạo ra sẽ được thêm vào trong AGENDA. Ví dụ : NP[0,2] là phần tử được lấy ra khỏi AGENDA, trong CHART có tồn tại phần VP[2,8] và trong bộ luật cú pháp có luật S → NP VP thì NP[0,2] và VP[2,8] sẽ kết hợp với nhau để tạo thành S[0,8]. S[0,8] sẽ được thêm vào AGENDA, còn NP[0,2] thì được thêm vào CHART.
  + Bước 3: bước 2 được lặp lại cho đến trong AGENDA không còn phần tử nào để xét hoặc trong CHART xuất hiện element S[1,n].

Còn đây là code giả mô phỏng thuật toán A\*:

*Procedure A\* Parsing*

*For wi in sentence*

*wi~<Xi ,i ,i+1>.*

*AGENDA.add(<<Xi ,i ,i+1>,0>); //inside lúc đầu = 0;*

*Endfor;*

*CHART 🡨 empty;*

*while ((!AGENDA.isempty) and ( !CHART.contain(<S, 1, n + 1>)))*

*Choose <<Y, i, j>,w> from AGENDA with max(w+h(Y));*

*if (!CHART.contains(<Y, i, j>)) then*

*CHART. add (<<Y, i, j>,w>);*

*For <<Z, j, k>,w’> in CHART and (X → Y Z,w’’)*

*AGENDA. add(<<X, i, k>,w + w’ + w’’ >);*

*Endfor;*

*For <<Z, k, i>,w’> in CHART and (X → Z Y,w’’)*

*AGENDA. add(<<X, k, j>,w + w’ + w’’ >);*

*Endfor;*

*Endif;*

*Endwhile;*

*If S.contain(<S, 1, n + 1>) then successParse*

*else failureParse;*

*EndProcedure;*

Chú thích :

* w, w’ tượng trưng cho inside của nút đi kèm.
* w’’ tượng trưng cho xác suất của luật đi kèm.
* h(Y) là ước lượng chi phí nhỏ nhất để đi đến đích từ Y.

### Hàm ưu tiên trong A\*

Cách tính hàm ưu tiên trong A\* như đã nói ở trên gồm có hai thành phần là g(x) – chi phí đường đi hiện tại, h(x) – ước lượng chi phí nhỏ nhất để đến đích từ X. Còn trong phân tích cú pháp, công thức tính hàm ưu tiên cho A\* như sau.

*F(nút) = inside(nút) + outside(nút)*

Phân tích: inside của một nút là xác suất của cây con được sinh ra từ nút đó, thông số này có thể coi như g(x), và tính có thể tính được nó một cách rất dễ dàng. Nhưng outside thì lại khác, vì tại một nút không thể nào tính được outside chính xác được vì muốn tính outside thực sự thì phải phân tích được toàn bộ câu, điều này là không thể. Vì thế, chúng ta chỉ có thể tính được outside một cách tương đối – ký hiệu là a(nút) như trường hợp của CYK. Và phải chọn cách tính sao cho a(nút)>=(nút) – outside thực sự của nút, ngược lại với h(x)>h’(x) ở thuật toán A\* gốc vì ở đây chúng ta cần tìm phương án có tổng trọng số lớn nhất chứ không phải nhỏ nhất. Để có thể tính được outside của giải thuật A\* trong phân tích cú pháp, đồ án đề xuất ra 3 phương án:

#### Sử dụng duy nhất xác suất inside

Một trong những cách tính đơn giản nhất là không sử dụng outside, outside của tất cả mọi trường hợp đều bằng không. Lúc này ứng cử viên sẽ là nút có inside lớn nhất, bài toán lúc này trở về tìm kiếm theo thuật toán nổi tiếng Dijkstra. Do không phải tính toán outside cho các nút nên thời gian tính toán của mỗi bước là rất nhanh. Tuy nhiên, do không có hàm ước lượng nên số bước phải xét là khá lớn.

#### Tính outside một tầng

Cách tính thứ hai là sử dụng các nút ở trên nút cần tính một tầng để tính outside của nút đó. Cụ thể, muốn tính outside của một nút X trong AGENDA, ta lấy X kết hợp với các nút trong CHART để tạo ra các nút ở mức trên để tính rồi chọn ra outside lớn nhất. Sự kết hợp này chỉ xảy ra một lần duy nhất. vì outside được chọn ra là outside lớn nhất lại chỉ kết hợp một tầng nên đảm bảo (X)>=a(X). Tuy nhiên vì chỉ kết hợp một tầng nên sự ước lượng đến đích từ nút hiện tại khá hời hợt, cộng thêm việc tính toán outside cho tất cả các nút trong AGENDA để lấy ra ứng cử viên rất phức tạp. Sự hời hợt cộng thêm sự phức tạp đã khiến cho thuật toán A\* tìm được đích rất chậm. Giải pháp này có một ưu điểm là dễ cài đặt, và trong một số trường hợp vẫn có thể dẫn đến đích rất nhanh, nhưng cũng gặp phải rất nhiều những trường hợp không thể đến được đích do ước lượng quá hời hợt. Những lúc như vậy, giải thuật A\* một tầng tỏ ra kém hiệu quả hơn cả thuật toán Dijkstra.

#### Tính outside bằng phương pháp rút gọn tập luật

Nguyên nhân chính gây ra sự khó khăn trong công đoạn tính trước outside chính là do tập luật cú pháp quá phức tạp (khoảng 938 luật). Nên khi kết hợp nút X với bất kỳ một nút Y nào đó, số lượng tổ hợp bùng nổ ra là quá lớn khiến cho việc tính toán outside cho một nút cũng đã trở nên vô cùng khó khăn chứ chưa nói đến việc tính outside cho tất cả các nút trong AGENDA. Chính vì thế đồ án quyết định đề xuất ra phương án tính toán outside bằng cách rút gọn tập luật để có thể tính tương đối outside của một nút một cách dễ dàng hơn.

Ý tưởng của giải thuật này là việc chia tập luật cú pháp ra thành các nhóm nhỏ Ri, mỗi nhóm Ri sẽ cử ra một luật đại diện ri có xác suất lớn nhất. Gom tất cả các ri đó lại, chúng ta sẽ có một tập luật mới : Max = {ri} – tập hợp những luật lớn nhất. Tập luật Max đơn giản hơn nhiều so với tập luật cú pháp gốc nên dĩ nhiên phân tích bằng Max cũng sẽ giảm thiểu độ phức tạp đi một cách đáng kể. Cụ thể, với một nút X bất kì trong AGENDA, X sẽ được kết hợp với các nút trong CHART bằng Max để tạo ra tập các nút Y, các nút Y lại được kết hợp với nút trong CHART bằng tập luật Max. Lặp lại đi lặp quá trình trên cho đến khi không thể kết hợp được nữa hoặc tìm thấy nút S[1,n]. Sau quá trình này ta được một cơ số các cây phân tích outside từ vị trí của nút X, chọn ra cây có xác suất lớn nhất, ta sẽ có được outside tương đối được tính theo phương pháp tối giản tập luật. Vì kết quả là phân tích outside có xác suất lớn nhất của nút X được phân tích bằng tập luật lớn nhất, nên trường hợp này cũng thỏa mãn điều kiện (X)>=a(X).

Dưới đây là code giả của quá trình tính toán outside bằng phương pháp rút gọn tập luật.

***outside(state)***

*for y in CHART*

*for (x→y state) in maxGrammar*

*cost = inside(y) + outside(x) + log P(x→y state);*

*score = max(score,cost);*

*endfor;*

*for (x→state y) in maxGrammar*

*cost = inside(y) + outside(x) + log P(x→state y);*

*score = max(score,cost);*

*endfor;*

*endfor;*

***end;***

Tập luật Max càng phức tạp thì ước lượng càng chính xác nhưng đồng thời độ phức tạp trong tính toán outside cũng tăng lên. Qua thực tế khi đã cài đặt và sử dụng, thuật toán này tỏ ra rất hữu dụng, với tốc độ và độ chính xác rất đáng kể, vì tuy phải tính toán nhiều ở bước xác định outside nhưng số bước lặp là rất ít do hàm ước lượng tốt.

### Cải tiến thuật toán A\* trong phân tích cú pháp với giải thuật lelightwin

#### Đặt vấn đề

Theo như giải thuật A\* đã trình bày ở trên thì tại bước 2, ta sẽ lấy ứng cử viên sáng giá nhất ra khỏi AGENDA và cho kết hợp với từng phần tử trong CHART, sự kết hợp này sẽ tạo ra một tổ hợp mới trong tập AGENDA. Nếu như hai phần tử có thể kết hợp thành một phần tử đầy đủ thì mọi chuyện trở nên quá dễ dàng. Tuy nhiên, đa số các trường hợp kết hợp như trên lại đều để lại biến wait dẫn đến việc các tổ hợp được sinh ra là quá nhiều. Ví dụ như trường hợp kết hợp hai phần tử N và N tạo ra đến 64 phần tử trong đó có đến 63 trường hợp là có biến wait. Nếu phân tích những câu nhỏ tầm khoảng 20 tokens đổ xuống thì không gặp nhiều vấn đề. Nhưng nếu với những câu dài và khó với độ phức tạp là 40-50 tokens thì số lượng các bước cần phải lặp có thể lên đến gần 10000 bước!

Lí do của sự bùng nổ này là do tập luật cú pháp của chúng ta quá lớn. Cho nên khi kết hợp hai phần tử X và Y có sử dụng biến wait thì tất cả các luật cú pháp có dạng Z → X Y…. đều góp phần tạo ra tổ hợp phần tử khổng lồ.

Qua đó, ta có thể thấy rằng trong trường hợp tập luật cú pháp quá lớn, thì việc sử dụng biến wait có thể gây bùng nổ tổ hợp. Thế nên đồ án đã quyết định đề xuất ra một giải thuật để cải tiến cho thuật toán A\* về vấn đề này, được gọi là giải thuật lelightwin.

#### Ý tưởng của giải thuật lelightwin

Ý tưởng của giải thuật này là không sử dụng phần tử wait hay còn gọi là phần tử chưa hoàn chỉnh để phân tích, mà sẽ kết hợp tất cả các phần tử có thể để tạo thành một phần tử hoàn chỉnh, nghĩa là số lượng phần tử kết hợp có thể lớn hơn 2. Ví dụ như khi kết hợp phần tử ứng cử viên N(2,7) lấy ra từ agenda với phần tử V(7,8) ta sẽ được duy nhất một phần tử NP(7,8), chứ không tạo ra một tổ hợp như trường hợp dưới đây:

*NP(2,8, wait="");NP(2,8, wait=", A");NP(2,8, wait="AP");*

*NP(2,8, wait="AP NP");NP(2,8, wait="AP PP");NP(2,8, wait="MP");*

*NP(2,8, wait="N");NP(2,8, wait="NP");NP(2,8, wait="NP PP");*

*NP(2,8, wait="NP VP");NP(2,8, wait="P");NP(2,8, wait="PP");*

*NP(2,8, wait="PP PP");NP(2,8, wait="VP");*

Với trường hợp của giải thuật tạo bảng CYK, thì việc thực hiện cải tiến lelightwin là điều không thể, vì bảng CYK chỉ có thể cho phép kết hợp 2 ô trong bảng. Nhưng với giải thuật A\*, mỗi một phần tử lấy ra khỏi AGENDA sẽ được kết hợp với một tập CHART, nên việc sử dụng cải tiến lelightwin là điều hoàn toàn có thể được.

Như đã phân tích ở trên, giải thuật A\* ­­- lelightwin sẽ không sử dụng biến wait cho trường hợp phần tử không hoàn chỉnh nữa. Nhưng điều đó đồng nghĩa với việc tại bước 2 của giải thuật A\*, ứng cử viên X được lấy ra khỏi AGENDA sẽ không chỉ đơn thuần là kết hợp với từng phần tử trong CHART. Chúng ta sẽ phải liệt kê ra tất cả những chuỗi gồm những phần tử chứa trong CHART có thể kết hợp với X. Tất nhiên, những phần tử trong cùng một chuỗi phải có vị trí thích hợp với nhau.

Ví dụ: giả sử với phần tử ứng cử viên có vị trí bắt đầu và kết thúc là X(7-10) và bảng CHART như sau:

Bảng ‑. Các phần tử trong CHART

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X1(1-8) | X2(6-16) | X3(15-35) | X4(5-20) | X5(2-7) | X6(10-11) |
| X7(8-27) | X8(2-21) | X9(9-11) | X10(2-13) | X11(6-14) | X12(15-26) |
| X13(14-23) | X14(5-18) | X15(1-7) | X16(9-16) | X17(12-17) | X18(7-18) |
| X19(6-25) | X20(13-26) | X21(11-16) | X22(9-24) | X23(11-20) | X24(8-18) |
| X25(7-16) | X26(14-16) | X27(4-6) | X28(13-21) | X29(4-8) | X30(11-13) |

Tất cả những chuỗi có thể đem ra kết hợp của ứng cử viên X[7-10] là :

Bảng ‑. Các chuỗi kết hợp của X với CHART

|  |  |
| --- | --- |
| Vị trí | Ký hiệu |
| [2-7] **[7-10]** | X5**X** |
| [1-7] **[7-10]** | X15**X** |
| [1-7] **[7-10]** [10-11] [11-13] | X15 **X** X6 X30 |
| [2-7]**[7-10]**[10-11] [11-13] [13-26] | X5 **X** X6 X30 X20 |
| **[7-10]** [10-11] [11-20] | **X** X6 X23 |
| **[7-10]** [10-11] [11-13] | **X** X6 X30 |
| **[7-10]** [10-11] [11-13] [13-26] | **X** X6 X30 X20 |
| **[7-10]** [10-11] [11-13] [13-21] | **X** X6 X30 X28 |
| [2-7] **[7-10]** [10-11] | X5 **X** X6 |
| [2-7] **[7-10]** [10-11] [11-16] | X5 **X** X6 X21 |
| [2-7] **[7-10]** [10-11] [11-20] | X5 **X** X6 X23 |
| [2-7] **[7-10]** [10-11] [11-13] | X5 **X** X6 X30 |
| **[7-10]** [10-11] [11-16] | **X** X6 X21 |
| [1-7] **[7-10]** [10-11] | X15 **X** X6 |
| [2-7] **[7-10]** [10-11] [11-13] [13-21] | X5 **X** X6 X30 X28 |
| [1-7] **[7-10]** [10-11] [11-16] | X15 **X** X6 X21 |
| [1-7] **[7-10]** [10-11] [11-20] | X15 **X** X6 X23 |
| **[7-10]** [10-11] | **X** X6 |
| [1-7] **[7-10]** [10-11] [11-13] [13-26] | X15 **X** X6 X30 X20 |
| [1-7] **[7-10]** [10-11] [11-13] [13-21] | X15 **X** X6 X30 X28 |

Sau đó những chuỗi này sẽ được kiểm tra, nếu chuỗi là vế phải của một luật nào đó trong tập cú pháp chẳng Z→ X15 **X** X6 X30 X20, thì Z sẽ được thêm vào AGENDA để xử lí. Và thuật toán này cũng lặp cho đến khi tìm được đáp án S(1-n) trong CHART hoặc trong AGENDA không còn phần tử nào để xét.

Giải thuật này sẽ khiến cho thời gian phải thực hiện ở mỗi bước của giải thuật A\* tăng lên tuy nhiên số bước phải xét sẽ nhỏ hơn nhiều vì không xảy ra trường hợp bùng nổ tổ hợp. Bài toán đặt ra ở đây là làm sao có thể xử lí việc đưa ra các chuỗi kết hợp với thời gian ngắn nhất. Để giải quyết vấn đề này, trong phần sau, chúng ta sẽ đi sâu vào mô tả rõ cách thực hiện thuật toán lelightwin.

#### Mô hình thuật toán lelightwin cơ bản



Hình ‑. Mô hình của thuật toán lelightwin.

##### Phân loại phần tử

Cách phân loại phần tử của giải thuật lelightwin được tạo ra dựa trên ý tưởng của thuật toán pigeonhole sort, hay còn gọi là thuật toán sắp xếp bồ câu. Một giải thuật tưởng chừng như không có gì liên quan ở đây, nhưng thực tế là có. Giải thuật lelightwin cũng tạo ra các khối để add phần tử (bồ câu) vào, nhưng không phải để sắp xếp mà là để sinh chuỗi.

Các khối trong giải thuật lelightwin được chia làm hai loại: khối những phần tử nằm bên trái X, và khối những phần tử nằm bên phải X.

* Những phần tử nằm bên trái X: đây là những phần tử mà vị trí end (kết thúc) của nó <= vị trí start (bắt đầu) của X. Những phần tử có cùng vị trí end sẽ được add vào trong khối được gán nhãn là end. Ví dụ khối 2 sẽ gồm những phần tử bên trái X có vị trí kết thúc bằng 2. Tất cả các khối này nằm trong một khối to hơn gọi là khối bên trái.
* Những phần tử nằm bên phải X: đây là những phần tử mà vị trí start của nó có giá trị >= vị trí end của X. Những phần tử có cùng vị trí start sẽ được add vào trong khối được gán nhãn là start. Tất cả các khối này cũng nằm trong một khối gọi là khối bên phải.

Hình ‑. Giai đoạn phân loại phần tử trong CHART.

##### Sinh chuỗi kết hợp dựa trên các phần tử đã phân loại

Với đầu vào là tập các phần tử CHART đã được phân loại cẩn thận, chúng ta sẽ bắt đầu công đoạn sinh chuỗi. Thuật toán lelightwin dùng để sinh chuỗi bao gồm các bước: sinh dãy kết hợp trái, sinh dãy kết hợp phải và cuối cùng là sinh chuỗi. Sau đây, ta sẽ đi vào mô tả từng công đoạn.

Hình ‑. Các công đoạn của thuật toán sinh chuỗi.

* + Công đoạn đầu tiên là công đoạn sinh ra tất cả các chuỗi con kết hợp trái của X (leftChain), có thể được mô tả như sau :
* Truy xuất đến tập các phần tử kết hợp trái của X (tức là tập các phần tử có vị trí end = vị trí start của X) sử dụng dữ liệu đã được phân loại ở trên. Cứ mỗi một phần tử trong tập này sẽ sinh ra một chuỗi kết hợp trái của X.
* Lặp đệ quy bước trên với các phần tử kết hợp trái của X.

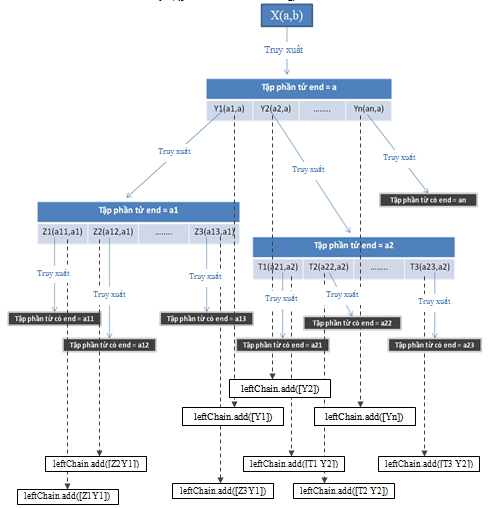
Theo một cách khác, thuật toán sinh này gần giống như thuật toán duyệt cây với X là nút gốc. Khi đi đến nút Y nào đó thì sinh ra chuỗi là đường ***đi từ Y đến X*** trừ đúng phần tử X ra.

Ví dụ minh họa:

Hình ‑. Ví dụ minh họa thuật toán sinh chuỗi con trái.

X là nút gốc, truy cập đến tập phần tử kết hợp trái của X gồm A và B, A và B lại truy cập đến tập phần tử kết hợp trái của mình, cứ thế chúng ta được giải thuật duyệt cây như hình trên. Tất cả các chuỗi được sinh ra bởi hàm sinh chuỗi đều được lưu vào trong tập leftChain để chờ xử lí.

Sau đây là sơ đồ giải thuật tổng quát mô tả cụ thể rõ ràng hơn về thuật toán sinh chuỗi con kết hợp trái :

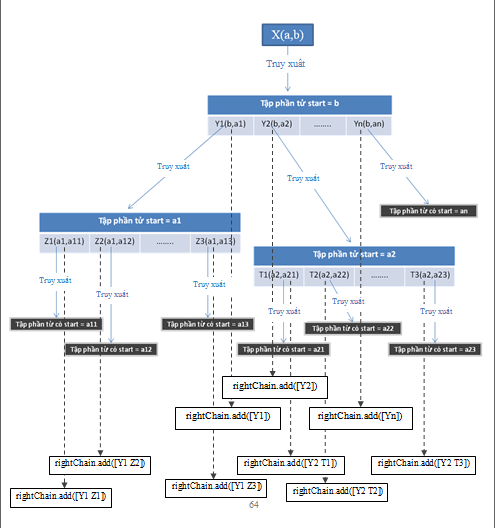


Hình ‑. Sơ đồ thuật toán sinh chuỗi con trái.

* + Tiếp theo là sinh ra tất cả các dãy kết hợp phải của X (rightChain), có thể được mô tả như sau :
* Truy xuất đến tập các phần tử kết hợp phải của X (tức là tập các phần tử có vị trí start = vị trí end của X). Cứ mỗi một phần tử trong tập này sẽ sinh ra một chuỗi kết hợp phải của X.
* Lặp đệ quy bước trên với các phần tử kết hợp phải của X.

Ngược với thuật toán sinh trái, thuật toán sinh phải cũng duyệt cây với X là nút gốc nhưng khi duyệt đến nút Y thì chuỗi được sinh ra lại là ***đường đi từ X đến Y*** ngoại trừ phần tử X.

Cuối cùng là sơ đồ mô tả giải thuật sinh chuỗi con phải:



Hình ‑. Sơ đồ thuật toán sinh chuỗi con phải.

* + Sau hai bước ở trên, ta đã thu được một tập leftChain gồm các dãy kết hợp trái của X, rightChain gồm các dãy kết hợp phải của X. Kết hợp leftChain, X, rightChain để tạo thành một tổ hợp chuỗi Chain hoàn chỉnh (Chain lúc đầu là rỗng).

Đầu tiên là chuỗi kết hợp trái của X:

*for (mỗi chuỗi left trong tập leftChain)*

*tạo ra một chuỗi chaini =[left X];*

*chain.add(chaini);*

*endfor;*

Sau đó tới lượt các chuỗi kết hợp phải của X được tạo ra:

*for (mỗi chuỗi right trong tập rightChain)*

*tạo ra một chuỗi chainj =[X right];*

*chain.add(chainj);*

*endfor;*

Cuối cùng, phức tạp nhất là trường hợp các chuỗi trái phải kết hợp lẫn lộn:

*for (mỗi chuỗi right trong tập rightChain)*

*for (mỗi chuỗi left trong tập leftChain)*

*tạo ra một chuỗi chaink =[left X right];*

*chain.add(chaink);*

*endfor;*

*endfor;*

Sau khi hoàn thành xong cả 3 công đoạn phức tạp ở trên, chúng ta sẽ thu được tất cả các chuỗi kết hợp của X với các phần tử trong CHART.

#### Thuật toán lelightwin Prunning

Như đồ án đã phân tích, giải thuật lelightwin được đề xuất ra là để tăng tốc độ, làm giảm số lượng tổ hợp bùng nổ ở mỗi bước lặp của thuật toán A\*. Tuy nhiên, nếu chỉ dừng lại ở những mô tả ở trên, giải thuật lelightwin chưa hẳn đã tối ưu về mặt tốc độ vì thời gian xử lí cho mỗi bước là khá lâu do phải xét tất cả các chuỗi kết hợp của X. Trong khi đó, xét trên thực nghệm trung bình chỉ có khoảng 8% số lượng các chuỗi được tạo ra là kết hợp được bằng luật cú pháp để tạo ra phần tử mới. Về mặt này, giải thuật lelightwin và giải thuật sử dụng biến wait có một điểm chung giống nhau: đó là sự thừa thãi. Nếu như giải thuật sử dụng biến wait bị thừa quá nhiều về số lượng tổ hợp sinh ra thì giải thuật lelightwin lại thừa quá nhiều về số lượng chuỗi phải xét. Chính điều này đã khiến cho giải thuật lelightwin không những không nhanh hơn giải thuật wait một cách đáng kể mà còn chậm hơn rất nhiều trong những trường hợp số bước lặp lên đến 400-500. Thêm vào đó, việc tạo ra quá nhiều bộ nhớ để lưu trữ các chuỗi có thể gây tràn bộ nhớ, vì có những trường hợp số lượng chuỗi có thể lên đến vài tỉ!!!!

Để giải quyết vấn đề này, đồ án đã phải suy nghĩ rất nhiều và quyết định đề xuất ra thêm một cải tiến cho giải thuật lelightwin, đó là giải thuật lelightwin sử dụng cắt tỉa. Nếu như giải thuật lelightwin cơ bản xét đến tất cả các chuỗi kết hợp có thể có của X gây ra sự thừa thãi không cần thiết thì giải thuật lelightwin cắt tỉa có thể tối ưu được sự thừa thãi này. Thay vì phải xử lí tất cả các chuỗi kết hợp có thể có của X, lelightwin prunning sẽ chỉ xét vừa đủ số chuỗi kết hợp được để thành phần tử mới, những nhánh dẫn đến chuỗi không thể kết hợp thành phần tử mới sẽ bị cắt tỉa.

Giải thuật prunning của lelightwin bao gồm hai giai đoạn như sau:

* Giai đoạn huấn luyện thống kê: đây là giai đoạn rất quan trọng quyết định cho việc cắt tỉa có thành công hay không. Nhờ vào giai đoạn huấn luyện này mà trong quá trình duyệt cây, giải thuật lelightwin có thể quyết định nên cắt tỉa một nhánh hay không.
* Giai đoạn thực hiện thuật toán.

##### Huấn luyện cho bộ cắt tỉa của lelightwin

Dưới đây là một số ký hiệu mà phần này sẽ sử dụng để người đọc tiện theo dõi:

Rchain – tập các luật cú pháp có chứa chuỗi chain trong vế phải của tập luật.

Fchain – tập các luật cú pháp có chain đứng đầu trong vế phải của tập luật.

Giải thuật lelightwin cơ bản sinh chuỗi kết hợp chỉ dựa trên vị trí tương ứng của các phần tử nhưng không hề tính đến nhãn của các phần tử đó. Ví dụ, với một phần tử PP thì đứng trước nó trong các luật cú pháp có thể là những phần tử nào? đứng sau là những phần tử nào? Giải thuật chưa hề xét đến điều này.

Để giải quyết vấn đề này, với mỗi một nhãn từ loại X xuất hiện trong tập luật cú pháp, chương trình sẽ tạo ra một tập thống kê cho nhãn đó bao gồm tập các nhãn từ loại có thể xuất hiện bên trái và bên phải của X, được lưu theo cấu trúc gần như cây phân nhánh, tạm gọi là cây phân cấp dữ liệu. Cụ thể như sau:

Hình ‑. Hình ảnh cây phân cấp dữ liệu chính.

Đây là một cây lưu trữ dữ liệu về các ký hiệu nằm trước X trong tập luật cú pháp – gọi là cây phân cấp dữ liệu chính. Sau đây là các bước để tạo ra cây này:

* Đầu tiên xét tập R[X]. Hai nút con của X gồm A và B được sinh ra với tư cách là tập những ký hiệu nằm ngay trước X trong vế phải của các luật R[X].
* Hai nút con C và D của A cũng được sinh ra với tư cách là tập tất cả những ký hiệu nằm ngay trước A trong vế phải của các luật R[A X]. Tương tự như vậy, H và I là tập tất cả những ký hiệu nằm ngay trước B trong vế phải của các luật R[B X].
* Thực hiện đệ quy thuật toán cho các nút ở mức dưới cho đến khi tất cả các nút mức thấp nhất đều không còn ký hiệu đứng trước trong tập luật tương ứng nữa.

Trong cây có một thông số rất quan trọng dành cho mỗi nút, đó là thông số first. Thông số first của nút C bằng true thông báo rằng trong tập luật cú pháp có tồn tại ít nhất một luật cú pháp mà trong vế phải của nó, C đứng đầu.

Với mỗi một nút có first = true trong cây lưu trữ ở trên sẽ có một cây lưu trữ tương ứng để lưu các phần tử có thể nằm bên phải của X – gọi là cây phân cấp dữ liệu con. Ví dụ với nút D màu trắng trong hình minh họa trên chẳng hạn. Vị trí của nó trong cây thông báo cho ta biết rằng nó thuộc trường hợp những luật cú pháp có vế phải bắt đầu bằng (D A X). Ta gọi tập hợp những luật này là tập F[D A X]. Ta cũng sẽ xây dựng cây để lưu các phần tử bên phải của X trong tập F[D A X] như ở trên. Cụ thể, giả sử cây phân tích của nút D có dạng như sau:

Hình ‑. Hình ảnh cây phân cấp dữ liệu con.

X gồm có hai nút con K và L là tập những ký hiệu nằm ngay sau X trong tập F[D A X]. Thuật toán lúc này diễn ra tương tự với trường hợp của cây phân cấp dữ trái. M và N là những ký hiệu nằm ngay sau L trong tập F[D A X L].

Giống như trường hợp của cây dữ liệu trái, cây dữ liệu phải của X cũng có biến logic **last** cho mỗi nút thông báo cho ta biết nút đó có phải là nút đứng cuối trong vế phải của ít nhất một luật trong tập luật cú pháp không? Nếu một nút bất kỳ thỏa mãn điều kiện này thì chuỗi sinh ra tương ứng với nó sẽ là vế phải của ít nhất một luật trong tập luật cú pháp. Ví dụ như ở nút M có last = true, nó thông báo cho ta biết rằng chuỗi tương ứng với M là [D A X L M] là vế phải của một luật trong tập cú pháp.

Thế nên với những nút này, ta sẽ duyệt tập luật để tìm ra nhãn vế trái tương ứng với chuỗi của nút và các nhãn vế trái này sẽ được lưu vào trong một tập GIFT như một thành phần của nút.

Ví dụ:

Nút có nhãn là VP biến last = true, chuỗi tương ứng với VP là [NP VP].

Trong tập luật cú pháp có hai luật có vế phải như trên :

S → NP VP

SQ → NP VP

Thì suy ra tập GIFT của VP sẽ gồm S và SQ.

Trong tập luật cú pháp 938 luật của chúng ta có khoảng 60-70 nhãn từ loại, vì vậy tương ứng với con số đó, chúng ta sẽ thu được 60-70 cây phân cấp dữ liệu. Công đoạn chuẩn bị dữ liệu cho việc cắt tỉa đã xong, sau đây chúng ta sẽ trình bày công đoạn tiếp theo và cũng là công đoạn chính của phương pháp lelightwin prunning.

##### Thực hiện quá trình cắt tỉa trong giải thuật lelightwin

Vẫn tiếp nhận đầu vào là tập dữ liệu CHART đã được phân loại, vẫn tiến hành giải thuật lelightwin như bình thường. Chỉ khác một điều là các nút không đúng sẽ bị cắt tỉa đi. Giả sử ứng cử viên là X (phân biệt với X ở phần trước) , cần kết hợp với các phần tử trong CHART, giải thuật của chúng ta như sau :

Hàm chính

* + Gọi hàm ***lelighwin***(X, ***Tree***(X)).

Hàm ***Tree***(phần\_tử E)

* + Trả về cây phân cấp dữ liệu tương ứng với nhãn của E.

Hàm ***lelightwin***(phần\_tử E, cây T)

* + Kiểm tra xem biến first của E có bằng true không? Nếu có thì thực hiện hàm ***lelightwin\_sub***(E, ***subTree***(E, T)).
  + Truy cập đến tập phần tử kết hợp trái của E ở tập CHART đã được phân loại, những phần tử nào không phải nút con của E trong cây T sẽ bị cắt tỉa.
  + Với từng phần tử Z trong tập các phần tử chưa bị cắt tỉa, gọi đệ quy hàm ***lelightwin***(Z, T).

Hàm ***subTree***(phần\_tử E, cây T)

* + Trả về cây phân cấp dữ liệu con tương ứng với E trong cây phân cấp dữ liệu T.

Hàm ***lelightwin\_sub***(phần\_tử E, cây T)

* + Kiểm tra xem trong T, E có last = true không? Nếu có thì add tập GIFT của E vào AGENDA với vị trí start và end thích hợp.
  + Truy cập đến tập phần tử kết hợp phải của E ở tập CHART đã được phân loại, những phần tử nào không phải nút con của E trong cây T sẽ bị cắt tỉa.
  + Với từng phần tử Y trong tập các phần tử chưa bị cắt tỉa, gọi đệ quy hàm ***lelightwin\_sub***(Y, T).

#### Thuật toán A\* kết hợp với thuật toán lelightwin prunning

Ở trên ta đã trình bày cách thức để thực hiện thuật toán lelightwin với cải tiến cắt tỉa. Bây giờ ta sẽ tiến đến bước tiếp theo là tích hợp thuật toán này vào thuật toán A\* dành cho phân tích cú pháp.

Các bước để thực hiện thuật toán:

* **Bước 1:** Khởi tạo đầu vào cho bộ phân tích cú pháp. Câu văn bản đầu vào được tách ra làm n từ, mỗi từ wi sẽ được gán một nhãn Li, vị trí bao phủ trong câu của wi: start=i, end=i+1. Tức là chúng ta sẽ có một tập n phần tử <Li,i,i+1>, tập này sẽ được lưu trong AGENDA để phục vụ cho bước tiếp theo của thuật toán.
* **Bước 2:** Tính toán ước lượng cho mỗi phần tử, lấy ra phần tử có ưu tiên cao nhất để xét, gọi phần tử này là candidate.
* **Bước 3:** Nếu candidate vẫn chưa tồn tại trong CHART thì:
  + Thực hiện hàm lelightwin(candidate, Tree(candidate)).
  + Thêm candidate vào trong tập CHART.
* **Bước 4:** lặp lại bước này cho đến khi trong AGENDA không còn phần tử để xét hoặc tìm được phần tử đích <S,1,n+1> trong CHART.

#### Nhận xét thuật toán A\* - lelightwin prunning

Ưu điểm:

* Tốc độ nhanh (gấp khoảng 3,4 lần so với giải thuật A\* lelightwin thông thường khi áp dụng cho phân tích cú pháp) do số lượng bước phải đi qua ít và không bị bùng nổ tổ hợp qua mỗi bước. Và nhất là việc xử lí tối ưu các trường hợp sinh chuỗi nhờ thuật toán cắt tỉa.
* Không sử dụng phần tử không hoàn chỉnh (phần tử với biến wait) để phân tích nên việc đánh giá độ ưu tiên giữa các nút sẽ trở nên chính xác hơn vì các nút được xét ưu tiên lúc này là nút hoàn chỉnh thật sự, đồng thời việc tính inside – outside trở nên đơn giản do không phải xét thêm trường hợp nút ảo.

Nhược điểm:

* Tốn khá nhiều dung lượng bộ nhớ do phải lưu trữ tất cả các chuỗi sinh ra trong quá trình phân tích.
* Cần huy động một lượng dữ liệu lưu trữ hậu cần rất lớn.
* Số lượng tổ hợp ghép nối càng nhiều thì thời gian cho mỗi bước càng tăng lên.
* Ngoài ra, thuật toán có một nhược điểm là quá phức tạp để cài đặt và kiểm thử.

**Kết chương**

Như vậy, để giải quyết những khó khăn của bài toán phân tích cú pháp tiếng Việt, đồ án đã thực hiện được những công việc sau

* + Đề xuất sử dụng thuật toán CYK-Beam search dựa trên mô hình PCFG nhằm tăng tốc độ cho bộ phân tích cú pháp.
* Đề xuất ra việc sử dụng thuật toán A\* để giải quyết những khó khăn của thuật toán CYK-Beam search có thể cải tiến tối đa đầu ra của bộ phân tích cú pháp mà vẫn có thể đảm bảo cho bộ phân tích về mặt tốc độ.
* Đưa ra được 3 cách tính hàm ưu tiên cho A\*
  + Không sử dụng outside với thuật toán Dijkstra.
  + Sử dụng cách tính ước lượng bằng outside một tầng.
  + Sử dụng phương pháp tính ước lượng bằng cách rút gọn tập luật.
* Đặc biệt là ở nửa cuối chương 3, đồ án đã trình bày về một ý tưởng do chính đồ án sáng tạo ra: thuật toán lelightwin cắt tỉa. Thuật toán này nhằm mục đích giúp cải tiến tốc độ của bộ phân tích cú pháp lên một mức cao hơn. Hiện tại do giới hạn trong thời gian làm đồ án, nên chương trình mới chỉ xây dựng được thuật toán lelightwin ở mức cơ bản, chưa có cắt tỉa. Nhưng đồ án đã được trình bày rất rõ ràng đến từng bước thực hiện của thuật toán lelightwin cắt tỉa. Chính vì vậy, việc xây dựng thuật toán này trong tương lai là chắc chắn có thể.

Sau đây, chúng ta sẽ đi sâu phần cài đặt chương trình **Chương 4 – Phát triển, thử nghiệm và đánh giá hệ thống phân tích cú pháp tiếng Việt.**

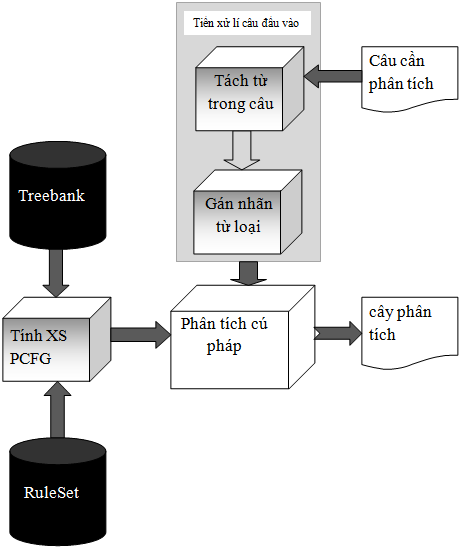
# PHÁT TRIỂN, THỬ NGHIỆM VÀ ĐÁNH GIÁ HỆ THỐNG

*Chương này đề cập tới việc cây dựng chương trình phân tích cú pháp tiếng Việt, những vấn đề gặp phải và cách giải quyết. Chương này bao gồm:*

* *Phân tích và thiết kế chương trình.*
* *Tổ chức lưu trữ dữ liệu.*
* *Cài đặt chương trình.*
* *Thử nghiệm và đánh giá chương trình.*

## Phân tích hệ thống

Mô hình của hệ thống phân tích cú pháp áp dụng cho tổng hợp tiếng nói tiếng Việt có thể được mô tả một cách tổng quát như sau:



Hình ‑. Mô hình tổng thể của hệ thống.

Phân tích hệ thống:

* Đầu vào: một câu văn bản bất kì; đầu ra: cây phân tích cú pháp.
* Các chức năng của hệ thống
  + Tiền xử lí văn bản: câu đầu vào phải được tách từ và gán nhãn từ loại trước khi chuyển giao sang làm đầu vào cho bộ phân tích cú pháp.
  + Quản lý luật: Là chức năng giúp cho bộ phân tích cú pháp có thể dễ dàng giao tiếp với tập luật cú pháp và đồng thời tính toán luôn xác suất PCFG cho mỗi luật cú pháp.
  + Phân tích cú pháp: Đây là chức năng quan trọng và mấu chốt nhất, tiếp nhận đầu vào của tất cả các chức năng trên, sử dụng giải thuật phân tích để cho ra cây phân tích cú pháp.
* Cơ sở dữ liệu
  + RuleSet: Tập luật cú pháp tiếng Việt.
  + VietTreeBank: là tập dữ liệu mẫu với hơn 2000 câu đã được phân tích cú pháp cẩn thận bằng tay.

### Tiền xử lí văn bản

#### Tách từ

Tách từ là chức năng rất quan trọng trong phân tích cú pháp, tiếp nhận trực tiếp đầu vào của toàn bộ hệ thống. Nhiệm vụ của hệ thống tách từ là với một câu đầu vào phải tách ra được thành các từ tiếng Việt(khái niệm từ đã được mô tả ở chương 2). Với sự đa dạng và phong phú của từ tiếng Việt thì có thể nói rằng, nhiệm vụ này không hề đơn giản.

Hiện tại ở Việt Nam đã cũng có không ít nghiên cứu về hệ thống tách từ dành cho tiếng Việt đã cho kết quả rất khả quan, đặc biệt trong số đó nổi trội lên bộ tách từ VNTokenzier thuộc nhánh đề tài “xử lí văn bản tiếng Việt” do giáo sư Hồ Tú Bảo đứng đầu. VNTokenizer là hệ thống tách từ sử dụng kết hợp giữa từ điển tiếng Việt và mô hình ngôn ngữ ngram, trong đó mô hình ngram được huấn luyện sử dụng kho dữ liệu huấn luyện khổng lồ (70,000 câu đã được tách từ bằng tay). Độ chính xác của hệ thống tách từ này đạt tới 97%. Sau khi thử nghiệm với rất nhiều loại văn bản phức tạp, đồ án quyết định sử dụng VNTokenzier cho bộ tiền xử lí văn bản thay vì xây dựng một bộ tách từ riêng.

Mô tả của chức năng tách từ

Bảng ‑. Mô tả của chức năng tách từ

|  |
| --- |
| **Đầu vào:** Một câu văn bản ngẫu nhiên  Ví dụ: *“hôm nay tôi đến trường bằng xe máy”* |
| **Đầu ra:** là câu sau khi đã được tách từ, các âm tiết trong cùng một từ sẽ được nối liền với nhau bởi dấu “\_”.  *“hôm\_nay tôi đến trường bằng xe\_máy”* |
| **Cách thực hiện:** sử dụng hệ thống tách từ VNTokenizer. |

#### Gán nhãn

Cùng với hệ thống tách từ, hệ thống gán nhãn từ loại tiếng Việt cũng đặt ra một câu hỏi hóc búa không kém cho chức năng tiền xử lí văn bản của hệ thống phân tích cú pháp. Nhiệm vụ của hệ thống gán nhãn là tiếp nhận câu đã được tách từ và gán nhãn từ loại cho mỗi từ trong câu. Nhãn từ loại tiếng Việt vốn đã phong phú lại còn được gán tùy vào hoàn cảnh, với một từ tiếng Việt trong trường hợp A có thể gán nhãn là danh từ, trường hợp B lại là động từ. Phức tạp như vậy nên nhiệm vụ của chức năng gán nhãn từ loại so với bộ tách từ không hề đơn giản hơn.

Tuy nhiê, cũng giống như hệ thống tách từ, hệ thống gán nhãn từ loại tiếng Việt cũng đã được nghiên cứu rất nhiều ở Việt Nam. Trong tầm hiểu biết của mình, đồ án đã tìm hiểu và biết đến hai hệ thống tách từ cho kết quả rất khả quan. Đó là hệ thống tách từ VnTagger cũng thuộc nhánh đề tài “xử lí văn bản tiếng Việt” do giáo sư Hồ Tú Bảo chủ trì và hệ thống vnqtag của Lê Hồng Phương. Qua thực nghiệm trên dữ liệu là các bài văn sưu tầm trên mạng, đồ án nhận thấy bộ VnTagger cho kết quả khả quan hơn nhưng bộ VnTagger sử dụng một tập nhãn từ loại có phần hơi khác biệt so với tập nhãn từ loại mà tập luật cú pháp sử dụng. Do không kịp thời gian để config lại tập nhãn cho thích hợp, nên đồ án quyết định sẽ sử dụng luôn bộ vnqtag của Lê Hồng Phương để thực hiện chức năng gán nhãn từ loại cho bộ phân tích cú pháp. Bản thân bộ gán nhãn từ loại vnqtag cũng cho độ chính xác rất cao, trên 90% nên lựa chọn này là hoàn toàn có thể chấp nhận được.

Mô tả của chức năng gán nhãn từ loại:

Bảng ‑. Mô tả của chức năng gán nhãn từ loại

|  |
| --- |
| **Đầu vào:** Đầu ra của bộ tách từ  *“hôm\_nay tôi đến trường bằng xe\_máy”* |
| **Đầu ra:** là tập các từ đã được gán nhãn  *hôm\_nay N(danh từ)*  *tôi P(đại từ)*  *đến V(động từ)*  *trường N(danh từ)*  *bằng C(giới từ)*  *xe\_máy N(danh từ)* |
| **Cách thực hiện:** sử dụng hệ thống gán nhãn từ loại vnqtag |

### **Quản lý luật (Rule Manager)**

Chức năng quản lý luật có hai nhiệm vụ chính là load file dữ liệu chứa tập luật cú pháp và tính toán xác suất PCFG cho mỗi luật dựa vào tập VietTreeBank.

#### Load file dữ liệu chứa tập luật

Bảng ‑. Bảng mô tả của chức năng load dữ liệu

|  |
| --- |
| **Đầu vào:** file dữ liệu chứa tập luật cú pháp được lưu trên máy. |
| **Đầu ra:** Tập luật cú pháp được lưu trữ bằng cấu trúc dữ liệu bảng băm. |
| **Cách thực hiện:**   * Truy cập file để lấy ra các luật cú pháp được lưu trong file. * Với mỗi luật đọc được, đẩy luật vào khối dữ liệu của bảng băm với khóa là vế phải của luật, những luật có cùng vế phải sẽ nằm trong cùng một khối dữ liệu. |

#### **Tính xác suất PCFG cho mỗi luật**

Dưới đây là hình minh họa cho chức năng tính xác suất PCFG:

#### 

Hình ‑. Mô hình tính xác suất PCFG cho mỗi luật.

Đây là chức năng tính toán các thông số cho mỗi luật dựa vào tập dữ liệu huấn luyện mẫu. Chức năng này cần phải sử dụng một lượng dữ liệu mẫu đủ lớn để cho tham số PCFG trở nên chính xác tương đối.

Bảng ‑. Bảng mô tả của chức năng tính xác suất PCFG

|  |
| --- |
| **Đầu vào:** tập dữ liệu VietTreeBank |
| **Đầu ra:** tập các xác suất tương ứng cho mỗi luật cú pháp của văn phạm phi ngữ cảnh. |
| **Cách thực hiện:** Chi tiết các bước thực hiện của chức năng này như sau :   * **Bước 1:** *Duyệt qua lần lượt các cây phân tích trong VietTreebank, chuyển cây về cấu trúc phù hợp để có thể dễ dàng tính được số lần sử dụng cho mỗi luật.* * **Bước 2:** *Với mỗi một cây, tiến hành đếm số lần sử dụng của mỗi luật trong cây và lưu lại con số này ra một file riêng. Mỗi một luật được lưu trong file sẽ đi kèm với một giá trị số nguyên, con số này là tổng của các lần xuất hiện của luật đó (được sử dụng) trong các cây trong Treebank mà ta đã duyệt qua. Các file này được lưu dạng text, mỗi một file sẽ lưu lại những luật có cùng vế trái đi kèm với số lần đếm được của luật đó.* * **Bước 3:** *Sau khi duyệt hết tất cả các cây trong Treebank, ta thu được tập các file lưu lại số lần đếm được của các luật. Tiến hành tính xác suất cho mỗi một luật theo dạng:*     *Trong đó: C(\*) là số lần đếm được của luật \*.*   * **Bước 4:** *Sau khi đã có được tất cả các luật với tham số PCFG tương ứng, lưu tất cả thông tin ra một file.* |

### Phân tích cú pháp

Như đã trình bày ở chương 3, giải thuật chính mà đồ án sẽ sử dụng cho chức năng phân tích cú pháp của hệ thống chính là giải thuật A\*. Nhưng đồ án cũng sẽ xây dựng cả giải thuật CYK-Beam search như một phương án thứ hai và để tiện so sánh giữa hai giải thuật này.

Bảng ‑. Bảng mô tả của chức năng phân tích cú pháp

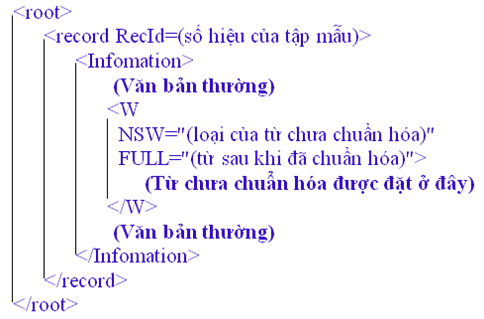
|  |
| --- |
| **Đầu vào:**   * Câu đầu vào sau khi đã được tách từ và gán nhãn. * Các cây phân cấp dữ liệu của các nhãn được lưu trong file. * Bảng băm chứa tập luật cú pháp tiếng Việt đã được tính xác suất dựa vào VietTreeBank. |
| **Đầu ra:** một cây phân tích cú pháp của câu đầu vào. |
| **Cách thực hiện:** Có hai cách thức để thực hiện chức năng này:   * *Sử dụng giải thuật phân tích cú pháp A\*.* * *Sử dụng giải thuật CYK – beam search.* |

## **Tổ chức lưu trữ dữ liệu**

Hệ thống phân tích cú pháp trong tổng hợp tiếng nói tiếng Việt, phần dữ liệu bao gồm: dữ liệu văn bản chuẩn hóa đầu vào, dữ liệu đầu ra của bộ tiền xử lí văn bản, tập luật cú pháp, kho dữ liệu VietTreeBank và dữ liệu đầu ra.

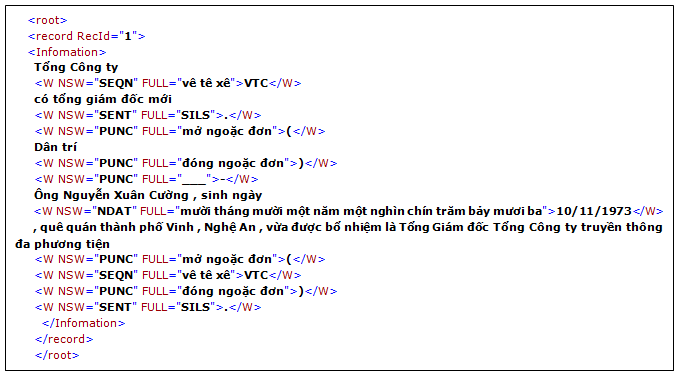
### Dữ liệu văn bản chuẩn hóa đầu vào

Như đã giới thiệu ở chương 1, vị trí của bộ phân tích cú pháp trong hệ thống tổng hợp tiếng nói tiếng Việt là ngay sau bộ chuẩn hóa văn bản. Thế nên dữ liệu đầu vào của phân tích cú pháp sẽ là văn bản sau khi đã được xử lí chuẩn hóa. Dữ liệu chuẩn hóa được lưu dưới dạng xml với các giá trị trong thẻ tương ứng với văn bản đã chuẩn hóa. Sau đây là cấu trúc xml của dữ liệu chuẩn hóa:



Hình ‑. Cấu trúc của dữ liệu văn bản chuẩn hóa.

Một ví dụ về lưu trữ dữ liệu chuẩn hóa của văn bản: *“Tổng công ty VTC có tổng giám đốc mới. (Dân trí)-Ông Nguyễn Xuân Cường, sinh ngày 10/11/1973, quê quán thành phố Vinh, Nghệ An, vừa được bổ nhiệm là Tổng Giám đốc Tổng Công ty truyền thông đa phương tiện (VTC).”*



Hình ‑. Ví dụ minh họa về dữ liệu chuẩn hóa văn bản.

### Dữ liệu chuyển giao của bộ tiền xử lí văn bản

Dữ liệu của bộ tiền xử lí văn bản là các từ được tách ra từ trong câu và các nhãn từ loại tương ứng. Cấu trúc của bộ dữ liệu này rất đơn giản, mỗi một từ được gán nhãn sẽ được lưu trên một dòng của file với cấu trúc như sau:

*<w pos=(nhãn từ loại)>từ</w>*

Dưới đây là ví dụ minh họa về dữ liệu lưu trữ của bộ tiền xử lí văn bản:

Bảng ‑. Minh họa dữ liệu đầu ra của bộ tiền xử lí văn bản

|  |
| --- |
| <w pos="V">đứng</w>  <w pos="N">trước</w>  <w pos="N">mặt</w>  <w pos="P">tôi</w>  <w pos="V">là</w>  <w pos="M">một</w>  <w pos="P">gã</w>  <w pos="N">đàn ông</w>  <w pos="A">cao</w>  <w pos="A">lêu đêu</w> |

### Tập luật cú pháp

Tập luật cú pháp là dữ liệu quan trọng nhất quyết định rất nhiều đến sự thành công của hệ thống. Chính vì vậy, việc lưu trữ nó cũng phải được tổ chức một cách hợp lý để đảm bảo thuận tiện cho quá trình đọc ghi cũng như tốc độ xử lí.

Như đã giới thiệu ở các phần trước, tập luật cú pháp của hệ thống bao gồm 938 luật được xây dựng dựa trên tập dữ liệu mẫu VietTreeBank và có sự tinh chỉnh để cho phù hợp với bộ phân tích cú pháp. Cấu trúc lưu trữ luật cú pháp của hệ thống là một file xml có dạng:

<VietParserRuleSet>

<Rule id="{số thứ tự của luật}"

probability="{xác suất PCFG của luật}">

<left>{vế trái của luật}</left>

<right>{vế phải của luật}</right>

</Rule>

</VietParserRuleSet>

### Kho dữ liệu VietTreeBank

Kho dữ liệu mẫu VietTreeBank là một tập dữ liệu gồm rất nhiều câu tiếng Việt đã được phân tích cú pháp chuẩn xác bằng tay. Tập dữ liệu TreeBank rất hay được dùng trong những ứng dụng liên quan đến xử lí ngôn ngữ tự nhiên, mà đặc biệt rất có ích trong việc xây dựng những hệ phân tích cú pháp chất lượng cao. Vì vậy, với một hệ thống phân tích cú pháp, để có thể cải thiện chất lượng đầu ra của kết quả phân tích thì chắc chắn phải có một TreeBank hỗ trợ cho riêng mình. Bản thân hệ thống cũng sử dụng một tập dữ liệu VietTreeBank được lưu trữ với cấu trúc dạng cây. Mỗi một cây phân tích cú pháp sẽ nằm trong một cặp thẻ đóng mở <s></s> và sử dụng cặp ngoặc đơn “()” để ngăn cách giữc các nhãn thành phần riêng biệt. Các nhãn con sẽ nằm trong “()” của nhãn cha.

Hình ‑. Cấu trúc của dữ liệu VietTreeBank.

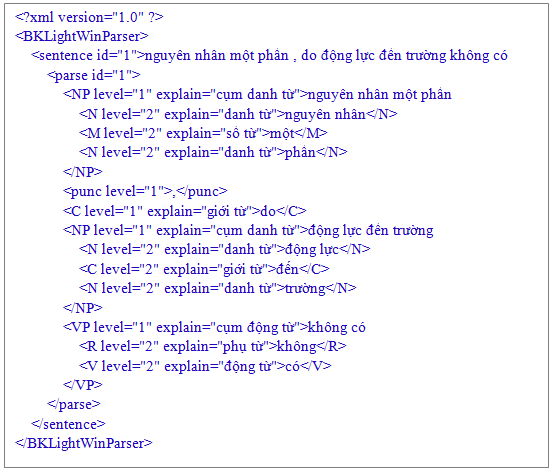
### Dữ liệu đầu ra của hệ thống

Cuối cùng là tổ chức lưu trữ dữ liệu đầu ra của hệ thống phân tích cú pháp. Cách tổ chức dữ liệu đầu ra cũng rất quan trọng vì giai đoạn tổng hợp tiếng nói mức thấp và các giai đoạn khác ở sau sẽ sử dụng đầu ra này. Chính vì vậy, hệ thống phải tổ chức dữ liệu đầu ra thích hợp để các giai đoạn sau có thể dễ dàng trong việc đọc và sử dụng dữ liệu.

Văn bản đầu vào có thể có nhiều câu, mỗi một câu có thể có nhiều cách phân tích cú pháp được xếp theo thứ tự từ cao nhất đến thấp nhất theo giá trị PCFG (với trường hợp của CYK-Beam search). Thế nên, dữ liệu đầu ra của hệ thống sẽ được tổ chức như sau:

* Mỗi một câu sẽ được lưu trong một cặp thẻ <s></s>.
* Trong mỗi một cặp thẻ <s></s>, sẽ nhiều cặp thẻ <parse></parse> tương ứng với các cách phân tích cú pháp cho câu nằm trong cặp thẻ <s></s> đang được xét.
* Trong các cặp thẻ <parse></parse> là cách phân tích của câu được lưu trữ dưới dạng cây. Các thẻ tương ứng với nhãn con sẽ được nằm trong các thẻ tương ứng với nhãn cha.

Sau đây là minh họa về mô hình dữ liệu đầu ra với trường hợp của một câu văn bản như sau: *“Nguyên nhân một phần, do động lực đến trường không có”.*



Hình ‑. Mô phỏng dữ liệu đầu ra của hệ thống.

Ngoài ra, ta có thể thấy với mỗi một thẻ trong cây phân tích cú pháp có thêm thuộc tính level, là thuộc tính chỉ ra mức của nút trong cây. Thuộc tính này để nhằm hỗ trợ các giai đoạn sau trong việc muốn tìm kiếm nhanh một cụm từ ở mức nào đó.

## Cài đặt hệ thống

### Công cụ lựa chọn

Hiện nay, với sự phát triển của ngành CNTT, có rất nhiều các công nghệ được lựa chọn để phát triển một ứng dụng. Trong số đó, nổi bật hơn cả là Java và .Net. Về mặt phát triển nhanh một ứng dụng với giao diện thân thuộc, dễ dùng thì .Net có phần nhỉnh hơn. Tuy nhiên, đồ án thiên về hướng nghiên cứu nên cần phải tái sử dụng, chỉnh sửa và chạy thử trên nhiều nền khác nhau. Thậm chí hướng phát triển của đồ án là sẽ đưa hệ thống lên web và điện thoại di động. Về những mặt trên thì Java thực hiện tốt hơn .Net. Ngoài ra, hệ thống được tạo ra nhằm mục đích phục vụ cho cộng đồng không mang tính thương mại nên việc lựa chọn hướng đi theo mã nguồn mở sẽ là thích hợp.

Như vậy, công nghệ mà đồ án lựa chọn sẽ là Java với môi trường phát triển là NetBean IDE 7.0.

### Cài đặt chương trình

Chương trình phân tích cú pháp được xây dựng với rất nhiều gói và các lớp rất phức tạp nên đồ án sẽ không nêu ra tất cả các lớp và các phương thức được sử dụng mà chỉ tập trung mô tả những gói chính của chương trình.

#### Gói RuleManager

Đây là gói được sử dụng để làm công việc quản lí và thac tác với luật cú pháp, gói này gồm có hai lớp chính là Rule và RuleSet.

##### Lớp Rule

Đây là class được sử dụng để quản lý thông tin về một luật cú pháp, mô phỏng một kiểu đối lượng luật cú pháp tiếng Việt.

Bảng ‑. Bảng mô tả của lớp Rule

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *private*  Kiểu: *String*  Tên: *left* | Vế trái của luật |
| Phạm vi: *private*  Kiểu: *int*  Tên: *count* | Số lần xuất hiện của luật trong tập VietTreeBank |
| Phạm vi : *private*  Kiểu: *String*  Tên: *right* | Vế phải của luật |
| Phạm vi: *private*  Kiểu: *float*  Tên: *prob* | Xác suất PCFG của luật |
| Phương thức | Phạm vi: *public*  Kiểu: *String*  Tên + tham số: *getLeft()* | Trả về giá trị của biến left |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *setLeft(String left)* | Gán giá trị cho biến left |
| Phạm vi: *public*  Kiểu: *int*  Tên+tham số: *getCount()* | Trả về giá trị biến count |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *setCount(int count)* | Gán giá trị cho biến count |
| Phạm vi: *public*  Kiểu: *float*  Tên+tham số: *getProb()* | Trả về biến prob |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *setProb(float prob)* | Gán giá trị cho biến prob |
| Phạm vi: *public*  Kiểu: *String*  Tên+tham số: *getRight()* | Trả về giá trị của biến right |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *setRight(String right)* | Gán giá trị cho biến right |

##### Lớp RuleSet

Là lớp để quản lý toàn bộ tập luật, cung cấp các phương thức để lấy luật

Bảng ‑. Bảng mô tả của lớp RuleSet

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *Private*  Kiểu: *ArrayList<AStarRule>*  Tên: *arlRule* | Mảng lưu các luật |
| Phạm vi: *private*  Kiểu:  *HashMap<String, ArrayList<AStarRule>>*  Tên: *htRule* | Bảng băm lưu các luật với khóa là vế phải của luật |
| Phạm vi: *private*  Kiểu: *ArrayList<Integer>*  Tên: *arlTotal* | Mảng lưu tổng số luật của mỗi cụm từ |
| Phương thức | Phạm vi: *public*  Kiểu: *boolean*  Tên+tham số: *readRule()* | Truy cập và đọc ra các luật cú pháp được lưu trong cơ sở dữ liệu |
| Phạm vi: *public*  Kiểu: *ArrayList<AStarElement>*  Tên+tham số:  *getCombine(ArrayList<AStarElement> as)* | Trả về vế trái của các luật có vế phải là chuỗi as. |

#### Gói phần tử parsing Element

Là gói để quản lý thông tin của các phần tử trong quá trình phân tích, bao gồm hai loại phần tử tương ứng với hai thuật toán A\* và CYK-Beam search:

##### Lớp CYK Element

Đây là lớp quản lý thông tin về phần tử được sử dụng trong quá trình phân tích cú pháp bằng thuật toán CYK – Beam search.

Bảng ‑. Bảng mô tả của lớp Element

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *Private*  Kiểu: *String*  Tên: *sLeft* | Lưu trữ nhãn từ loại trong phân tích CYK. |
| Phạm vi: *private*  Kiểu: *String*  Tên: *sWait* | Lưu biến wait tương ứng với phần tử trong phân tích CYK. |
| Phạm vi: *private*  Kiểu: *int*  Tên: *iPos1* | Lưu vị trí của ô đầu tiên tạo ra phần tử. |
| Phạm vi: *private*  Kiểu: *int*  Tên: *iIndex1, iIndex2* | Lưu chỉ số của hai ô thành phần tạo ra phần tử. |
| Phạm vi: *private*  Kiểu: *float*  Tên: *in\_side* | Xác suất inside của phần tử. |
| Phạm vi: *private*  Kiểu: *float*  Tên: *out\_side* | Xác suất outside của phần tử. |
| Phạm vi: *private*  Kiểu: *String*  Tên: *sLast* | Lưu thành phần cuối cùng trong luật phân tích |
| Phương thức | Tất cả các phương thức get và set của các thuộc tính trên | |

##### Lớp Cell

Đây là lớp để quản lý thông tin về một ô trong quá trình phân tích cú pháp CYK- Beam search.

Bảng ‑. Bảng mô tả của lớp Cell

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *Private*  Kiểu: *ArrayList<Element>*  Tên: *arlElem* | Lưu trữ tập các phần tử được chứa trong ô. |
| Phương thức | Phạm vi: *public*  Kiểu: *ArrayList<Element>*  Tên+tham số: *getArrElem()* | Trả về tập *arlElem* của ô*.* |
| Phạm vi: *public*  Kiểu: *Element*  Tên+tham số: *getElement(int i)* | Trả về phần tử thứ i trong ô. |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *removeElement(int i)* | Xóa phần tử thứ i ra khỏi ô. |
| Phạm vi: *public*  Kiểu: *int*  Tên+tham số: *size()* | Trả về số phần tử chứa trong ô. |
| Phạm vi: *public*  Kiểu: *int*  Tên+tham số: *sizeUsed()* | Trả về số phần tử chưa bị cắt tỉa bởi beam search trong ô. |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *add(Element elem)* | Thêm phần tử elem vào ô. |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *addWord(Word w)* | Thêm phần tử có chứa từ w vào trong ô. |

##### Lớp AstarElement

Như tên gọi của mình, lớp này để quản lý thông tin về phần tử sẽ được sử dụng trong quá trình phân tích cú pháp bằng giải thuật A\*.

Bảng ‑. Bảng mô tả của lớp AstarElement

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *Private*  Kiểu: *String*  Tên: *sCar* | Lưu trữ nhãn từ loại của phần tử. |
| Phạm vi: *private*  Kiểu: *int*  Tên: *start* | Lưu vị trí bắt đầu của phần tử trong câu. |
| Phạm vi: *private*  Kiểu: *int*  Tên: *end* | Lưu vị trí kết thúc của phần tử trong câu. |
| Phạm vi: *private*  Kiểu: *ArrayList<AStarElement>*  Tên: *subElement* | Lưu các phần tử thành phần tạo ra phần tử trong quá trình phân tích. |
| Phạm vi: *private*  Kiểu: *float*  Tên: *in\_side* | Xác suất inside của phần tử. |
| Phạm vi: *private*  Kiểu: *float*  Tên: *out\_side* | Xác suất outside của phần tử. |
| Phạm vi: *private*  Kiểu: *ArrayList<AStarElement>*  Tên: *outsideElem* | Lưu các giá trị để tính xác suất outside |
| Phạm vi: *private*  Kiểu: *float*  Tên: *Prob* | Lưu xác suất của luật mà element sử dụng để sinh ra các element thành phần. |
| Phương thức | Tất cả các phương thức get và set của các thuộc tính trên | |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số:  *contract( AStarRule rule,*  *ArrayList<AStarElement> as)* | Hàm làm nhiệm vụ thiết lập một danh sách phần tử hợp thành cho phần tử đang được xét với một luật cú pháp. |
| Phạm vi: *public*  Kiểu: *String*  Tên+tham số: *getWord()* | Trả về từ loại tương ứng với nhãn trong trường hợp phần tử là nút lá. |
| Phạm vi: *public*  Kiểu: *boolean*  Tên+tham số: *isWord()* | Kiểm tra xem phần tử đang xét có phải là nút lá từ loại không |

#### Gói xử lí chung Common

Là một gói phụ trong hệ thống dùng để thực hiện những xử lí thông thường.

##### Lớp PartOfSpeech

Lớp PartOfSpeech được sử dụng để ánh xạ từ các ký hiệu tiếng Anh sang tiếng Việt, được sử dụng trong việc dựng cây.

Bảng ‑. Bảng mô tả của lớp ParOfSpeech

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *private*  Kiểu: *ArrayList<String>*  Tên: *arlPos* | Lưu trữ các ký hiệu tiếng Anh (từ, cụm từ, câu) |
| Phạm vi: *private*  Kiểu: *ArrayList<String>*  Tên: *arlName* | Lưu trữ các tên tiếng Việt tương ứng. |
| Phạm vi: *private*  Kiểu: *int*  Tên: *iPos* | Lưu trữ số lượng các ký hiệu |
| Phương thức | Phạm vi: *public*  Kiểu: *String*  Tên+tham số:  *getName(String sPos)* | Lấy tên tiếng Việt của kí hiệu đầu vào. |
| Phạm vi: *public*  Kiểu: *String*  Tên+tham số:  *getPos(String sName)* | Lấy kí hiệu tiếng Anh của tên tiếng Việt đầu vào. |

##### Lớp Functions

Lớp này bao gồm các hàm thực hiện việc kiểm tra, bao gồm kiểm tra xem từ đầu vào có phải là danh riêng, số từ hay không.

Bảng ‑. Bảng mô tả của lớp Functions

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Lớp này không có thuộc tính gì nổi bật vì chỉ là lớp xử lí các việc phụ |  |
| Phương thức | Phạm vi: *public*  Kiểu: *boolean*  Tên+tham số:  *isDanhTuRieng(String strParam)* | Kiểm tra xem xâu đầu vào có phải là một danh từ riêng hay không |
| Phạm vi: *public*  Kiểu: *boolean*  Tên+tham số:  *isSymbol(String strParam)* | Kiểm tra xem xâu đầu vào có phải là một ký hiệu đặc biệt hay không |
| Phạm vi: *public*  Kiểu: *boolean*  Tên+tham số:  *isSotu(String strParam)* | Kiểm tra xem xâu đầu vào có phải là một số từ hay không |

#### Gói Analysis

##### Lớp CYKBeamSearch

Đây là lớp thực hiện phân tích cú pháp bằng giải thuật CYK kết hợp cắt tỉa beam search.

Bảng ‑. Bảng mô tả của lớp CYKBeamSeach

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *private*  Kiểu: *String*  Tên: *sentence* | Câu đầu vào. |
| Phạm vi: *private*  Kiểu: *float*  Tên: *fRatio* | Tỉ lệ loại bỏ trong cắt tỉa beam search. |
| Phạm vi: *private*  Kiểu: *RuleSet*  Tên: *ruleS* | Tập luật cú pháp. |
| Phạm vi: *private*  Kiểu: *DefaultMutableTreeNode*  Tên: *rootNode* | Là biến sẽ nhận kết quả trả về từ bộ phân tích cú pháp với giá trị là nút gốc của cây phân tích cú pháp. |
| Phạm vi: *private*  Kiểu: *ArrayList<String>*  Tên: *specialWord* | Mảng lưu các từ đặc biệt, bao gồm danh từ riêng và số từ. |
| Phạm vi: *private*  Kiểu: *ArrayList<Integer>*  Tên: *specialIndex* | Mảng lưu vị trí các dấu câu, để sử dụng trong việc thêm nhãn |
| Phạm vi: *private*  Kiểu: *ArrayList<String>*  Tên: *arlTree* | Mảng lưu xâu đọc được từ cây cú pháp. |
| Phạm vi: *private*  Kiểu: *ArrayList<Word>*  Tên: *arlWord* | Mảng lưu trữ các từ trong câu |
| Phạm vi: *private*  Kiểu: *ArrayList<Integer>*  Tên: *arlSize* | Mảng lưu kích thước các hàng trong bảng CYK |
| Phạm vi: *private*  Kiểu: *ArrayList<Cell>*  Tên: *table* | Danh sách các ô trong phân tích CYK. |
| Phương thức | Phạm vi: *public*  Kiểu: *void*  Tên+tham số:  *CYK(float step)* | Thực hiện phân tích CYK với ngưỡng step. |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *run()* | Là hàm chính thực hiện công việc phân tích câu đầu vào. |
| Pham vi: *public*  Kiểu: *void*  Tên+tham số:  *buildTree()* | Dựng cây phân tích từ bảng CYK |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số:  *joinCell*  *(int iPos1, int iPos2,*  *Bounds bound1,*  *Bounds bound2 )* | Thực hiện việc kết hợp hai ô ở vị trí iPos1 và iPos2 trong bảng CYK. |

##### Lớp AStar

Đây là lớp thực hiện phân tích cú pháp bằng thuật toán A\* có sử dụng thuật toán lelightwin cơ bản.

Bảng ‑. Bảng mô tả của lớp AStar

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *private*  Kiểu: *RuleSet*  Tên: *ruleS* | Tập luật cú pháp. |
| Phạm vi: *private*  Kiểu: *ArrayList<Word>*  Tên: *arlWord* | Mảng lưu trữ các từ trong câu. |
| Phạm vi: *private*  Kiểu: *ArrayList<Integer>*  Tên: *specialIndex* | Biến lưu vị trí các dấu câu, để dùng cho việc thêm nhãn cụm từ. |
| Phạm vi: *private*  Kiểu:  *HashMap<String, AStarElement>*  Tên: *chart* | Tập CHART được sử dụng trong thuật toán A\*. |
| Phạm vi: *private*  Kiểu: *ArrayList<AStarElement>*  Tên: *agenda* | Tập AGENDA được sử dụng trong thuật toán A\*. |
| Phạm vi: *private*  Kiểu: *DefaultMutableTreeNode*  Tên: *node* | Giống như trường hợp của CYK, đây là biến lưu giữ kết quả sau khi phân tích. |
| Phương thức | Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *run()* | Đây là hàm chính trong class thực hiện phân tích AStar. |
| Phạm vi: *public*  Kiểu: *ArrayList<AStarElement>*  Tên+tham số:  g*etCombinedElements(AStarElement e)* | Hàm trả về tập phần tử được tạo ra bởi sự kết hợp của phần tử e với các phần tử trong CHART. |
| Phạm vi: *public*  Kiểu: *float*  Tên+tham số:  *calculateOutside(AStarElement cand)* | Thực hiện tính outside cho phần tử cand. |
| Phạm vi: *public*  Kiểu: *AStarElement*  Tên+tham số: *candidate()* | Trả về phần tử ứng cử viên có ước lượng cao nhất. |
| Phạm vi: *public*  Kiểu: *DefaultMutableTreeNode*  Tên+tham số:  *getNode(AStarElement a)* | Trả về nhánh cây mà a làm nút gốc trong cây phân tích cú pháp đầu ra. |
| Phạm vi: *public*  Kiểu: *DefaultMutableTreeNode*  Tên+tham số:  *getNode()* | Trả về biến node. |
| Phạm vi: *public*  Kiểu: *String*  Tên+tham số:  *key(AstarElement e)* | Trả về khóa của phần tử e trong AGENDA. |

##### Lớp LeLightWin

Đây là lớp thực hiện giải thuật lelightwin cơ bản, thực hiện công việc xử lí và trả về một tập các chuỗi kết hợp của phần tử ứng cử viên với các phần tử trong CHART.

Bảng ‑. Bảng mô tả của lớp LeLightWin

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *private*  Kiểu:  *ArrayList<ArrayList<AStarElement>>*  Tên: *chain* | Biến để lưu kết quả của thuật toán là các chuỗi kết hợp. |
| Phạm vi: *private*  Kiểu:  *ArrayList<ArrayList<AStarElement>>*  Tên: *leftChain* | Dãy kết hợp trái của phần tử ứng cử viên |
| Phạm vi: *private*  Kiểu:  *ArrayList<ArrayList<AStarElement>>*  Tên: *rightChain* | Dãy kết hợp phải của phần tử ứng cử viên |
| Phạm vi: *private*  Kiểu:  *HashMap<String, ArrayList<AStarElement>>*  Tên: *left* | Tập phần tử nằm bên trái ứng cử viên sau khi đã được phân loại |
| Phạm vi: *private*  Kiểu:  *HashMap<String, ArrayList<AStarElement>>*  Tên: *right* | Tập phần tử nằm bên phải ứng cử viên sau khi đã được phân loại |
| Phương thức | Phạm vi: *public*  Kiểu: *void*  Tên+tham số:  *leftProcess(AStarElement elem)* | Hàm làm nhiệm vụ sinh ra leftChain của ứng cử viên bắt đầu tại vị trí phần tử elem |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số:  *rightProcess(AStarElement elem)* | Hàm làm nhiệm vụ sinh ra rightChain của ứng cử viên bắt đầu tại vị trí phần tử elem |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số:  *classify(AStarElement e,*  *HashMap<String, AStarElement> ea)* | Hàm làm nhiệm vụ phân loại các phần tử trong CHART và lưu vào hai tập left và right. |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số:  *generateSubChain(AStarElement e)* | Hàm làm nhiệm vụ tạo ra các chuỗi con kết hợp của phần tử ứng cử viên từ hai tập left và right |
| Phạm vi: *public*  Kiểu:  *ArrayList<ArrayList<AStarElement>>*  Tên+tham số:  *getLLWChain(AStarElement e,*  *HashMap<String, AStarElement> chart)* | Hàm cuối cùng, là hàm chính trong class, làm nhiệm vụ quan trọng nhất: sinh ra tất cả các chuỗi kết hợp của ứng cử viên e và lưu vào chain. |

##### Lớp Sentence

Đây có thể coi là lớp chính của gói phân tích câu analysis. Lớp này sẽ đảm nhận nhiệm vụ lưu trữ thông tin về một câu văn bản đầu vào, cung cấp các phương thức để phân tích cú pháp cho câu đó dựa vào tất cả các lớp ở trên.

Bảng ‑. Bảng mô tả của lớp Sentence

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Phạm vi: *private*  Kiểu: *String*  Tên: *sentence* | Biến lưu câu văn bản đầu vào |
| Phạm vi: *private*  Kiểu: *ArrayList<String>*  Tên: *specialWord* | biến lưu các từ đặc biệt: danh từ riêng và số từ |
| Phạm vi: *private*  Kiểu: *ArrayList<Integer>*  Tên: *specialIndex* | biến lưu vị trí các dấu câu, để sử dụng trong việc thêm nhãn cụm từ |
| Phạm vi: *private*  Kiểu: *ArrayList<Word>*  Tên: *arlWord* | mảng lưu trữ các từ trong câu |
| Phương thức | Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *splitWord()* | Đảm nhận nhiệm vụ tách từ cho câu đầu vào. Kết quả trả ra file “text.txt”. |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *callQTag()* | Đọc dữ liệu từ file “text.txt” và thực hiện gán nhãn từ loại. Kết quả lưu trong file “wordTagged.txt” |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *readWordFromFile()* | Đọc dữ liệu từ file “wordTagged.txt” |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *CYKParser()* | Thực hiện phân tích CYK cho câu đầu vào, có sử dụng kết hợp beam search. |
| Phạm vi: *public*  Kiểu: *void*  Tên+tham số: *astarParser()* | Thực hiện phân tích A\* cho câu đầu vào, có sử dụng kết hợp thuật toán lelighwin cơ bản. |
| Phạm vi: *public*  Kiểu: *DefaultMutableTreeNode*  Tên+tham số: *getAStarNode* | Trả về kết quả phân tích của AStar |
| Phạm vi: *public*  Kiểu: *DefaultMutableTreeNode*  Tên+tham số: *getCYKNode* | Trả về kết quả phân tích của CYK |

## Thử nghiệm và đánh giá

### Kết quả thử nghiệm

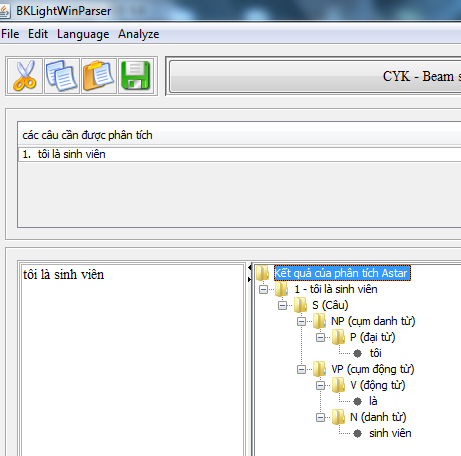
#### Tập dữ liệu thử nghiệm

Trong quá trình thử nghiệm chương trình, đồ án đã tiến hành thu thập các mẫu câu chuẩn trên các bài báo dantri.com và vietnamnet.vn, đồng thời sử dụng tập câu trong bộ dữ liệu của hệ thống tổng hợp tiếng nói, tất cả bao gồm 630 câu. Những câu này rất phức tạp và khó vì là những câu chuyên dùng trong hành văn. Ngoài ra đồ án đã trích một số câu trong tập VietTreeBank để thử độ chính xác của hệ thống phân tích cú pháp. Dữ liệu này có thể coi là chấp nhận được, vì hệ thống phân tích cú pháp vẫn chưa được huấn luyện mà chỉ đơn thuần là thống kê xác suất từ VietTreeBank.

#### Kết quả phân tích.

Dưới đây là một số kết quả mà bộ phân tích cú pháp với thuật toán A\* đã đạt được với các câu từ đơn giản đến phức tạp.

* “Tôi là sinh viên”



Hình ‑. CPTCP “tôi là sinh viên”

Đầu ra xml:

<?xml version="1.0" ?>

<BKLightWinParser>

<sentence id="1">tôi là sinh viên

<parse id="1">

<NP level="1" explain="cụm danh từ">tôi

<P level="2" explain="đại từ">tôi</P>

</NP>

<VP level="1" explain="cụm động từ">là sinh viên

<V level="2" explain="động từ">là</V>

<N level="2" explain="danh từ">sinh viên</N>

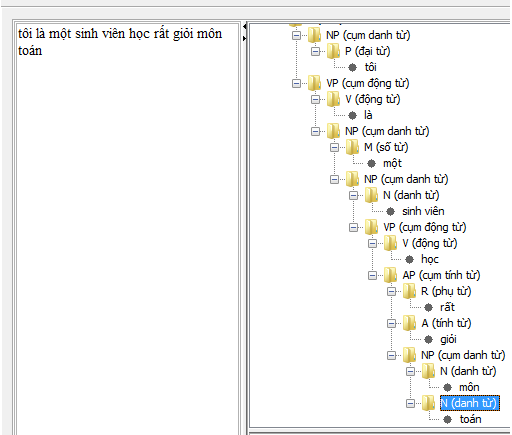
</VP>

</parse>

</sentence>

</BKLightWinParser>

* “Tôi là một sinh viên học rất giỏi môn toán”.



Hình ‑. CPTCP “tôi là một sinh viên học rất giỏi môn toán”

Đầu ra xml:

<?xml version="1.0" ?>

<BKLightWinParser>

<sentence id="1">tôi là một sinh viên học rất giỏi môn toán

<parse id="1">

<NP level="1" explain="cụm danh từ">tôi

<P level="2" explain="đại từ">tôi</P>

</NP>

<VP level="1" explain="cụm động từ">là một sinh viên học rất giỏi môn toán

<V level="2" explain="động từ">là</V>

<NP level="2" explain="cụm danh từ">một sinh viên học rất giỏi môn toán

<M level="3" explain="số từ">một</M>

<NP level="3" explain="cụm danh từ">sinh viên học rất giỏi môn toán

<N level="4" explain="danh từ">sinh viên</N>

<VP level="4" explain="cụm động từ">học rất giỏi môn toán

<V level="5" explain="động từ">học</V>

<AP level="5" explain="cụm tính từ">rất giỏi môn toán

<R level="6" explain="phụ từ">rất</R>

<A level="6" explain="tính từ">giỏi</A>

<NP level="6" explain="cụm danh từ">môn toán

<N level="7" explain="danh từ">môn</N>

<N level="7" explain="danh từ">toán</N>

</NP>

</AP>

</VP>

</NP>

</NP>

</VP>

</parse>

</sentence>

</BKLightWinParser>

* “Dạo này tôi không còn thích đến trường như lúc trước nữa”.

“Nguyên nhân một phần, do động lực đến trường không có”.

Đầu ra xml:

<?xml version="1.0" ?>

<BKLightWinParser>

<sentence id="1">dạo này tôi không còn thích đến trường như lúc trước nữa

<parse id="1">

<VP level="1" explain="cụm động từ">dạo

<V level="2" explain="động từ">dạo</V>

</VP>

<NP level="1" explain="cụm danh từ">này tôi

<P level="2" explain="đại từ">này</P>

<P level="2" explain="đại từ">tôi</P>

</NP>

<VP level="1" explain="cụm động từ">không còn thích đến trường như lúc trước nữa

<R level="2" explain="phụ từ">không</R>

<V level="2" explain="động từ">còn</V>

<VP level="2" explain="cụm động từ">thích đến trường như lúc trước nữa

<V level="3" explain="động từ">thích</V>

<PP level="3" explain="cụm giới từ">đến trường như lúc trước nữa

<C level="4" explain="giới từ">đến</C>

<NP level="4" explain="cụm danh từ">trường như lúc trước nữa

<N level="5" explain="danh từ">trường</N>

<PP level="5" explain="cụm giới từ">như lúc trước nữa

<C level="6" explain="giới từ">như</C>

<NP level="6" explain="cụm danh từ">lúc trước nữa

<N level="7" explain="danh từ">lúc</N>

<N level="7" explain="danh từ">trước</N>

<R level="7" explain="phụ từ">nữa</R>

</NP>

</PP>

</NP>

</PP>

</VP>

</VP>

</parse>

</sentence>

<sentence id="2">nguyên nhân một phần , do động lực đến trường không có

<parse id="1">

<NP level="1" explain="cụm danh từ">nguyên nhân một phần

<N level="2" explain="danh từ">nguyên nhân</N>

<M level="2" explain="số từ">một</M>

<N level="2" explain="danh từ">phần</N>

</NP>

<punc level="1">,</punc>

<C level="1" explain="giới từ">do</C>

<NP level="1" explain="cụm danh từ">động lực đến trường

<N level="2" explain="danh từ">động lực</N>

<C level="2" explain="giới từ">đến</C>

<N level="2" explain="danh từ">trường</N>

</NP>

<VP level="1" explain="cụm động từ">không có

<R level="2" explain="phụ từ">không</R>

<V level="2" explain="động từ">có</V>

</VP>

</parse>

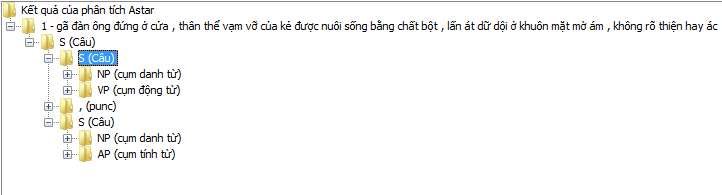
</sentence>

</BKLightWinParser>

* Sau đây chúng ta sẽ thử với một câu cực khó vài dài:

“gã đàn ông đứng ở cửa , thân thể vạm vỡ của kẻ được nuôi sống bằng chất bột , lấn át dữ dội ở khuôn mặt mờ ám , không rõ thiện hay ác”

Hệ thống vẫn hoàn toàn có thể phân tích được:



Hình ‑. Hình ảnh phân tích của một câu cực khó và dài.

Dưới đây là bảng tổng kết về quá trình thử nghiệm của hệ thống với 630 câu văn bản phức tạp(có so sánh với thuật toán CYK-Beam search):

Bảng ‑. Bảng tổng kết thử nghiệm với 630 câu hành văn

|  |  |  |
| --- | --- | --- |
| Thuật toán | Thời gian xử lí | Số lượng phân tích được |
| A\* | 15 phút | 92% |
| CYK-Beam search | 45 phút | 75% |

Về độ chính xác, sau khi thử nghiệm với khoảng 200 câu trong tập TreeBank, độ chính xác đạt được khoảng 70% (A\*).

### Đánh giá hệ thống

Với phạm vi của một đồ án tốt nghiệp, những kết quả mà hệ thống đạt được là khá khả quan. Tuy nhiên, kết quả thử nghiệm cho thấy kết quả của bộ phân tích cú pháp chưa được cao. Nguyên nhân vì những lí do sau đây:

* Bộ phân tích cú pháp vẫn chưa có giải thuật huấn luyện với tập TreeBank mà chỉ đơn thuần sử dụng thống kê nên hiệu năng của chương trình không được cải tiến.
* Tập luật cú pháp vẫn còn cần phải hoàn thiện thêm.
* Bộ tách từ và bộ gán nhãn cho ra kết quả sai dẫn đến đầu ra của bộ phân tích cú cũng sai.
* Các câu trong tập VietTreeBank là rất khó và dài, hầu hết là những câu có độ phức tạp 50-60 từ và cấu trúc rất phức tạp.

Ngoài ra, về mặt tốc độ, hệ thống đạt tốc độ rất nhanh trong quá trình phân tích. Hơn nữa, đó mới chỉ là bước đầu, nếu kết hợp thêm giải thuật lelightwin cắt tỉa, tốc độ của hệ thống có thể được cải thiện lên hàng chục lần.

**Kết chương**

Chương này đã trình bày kết quả kiểm thử cũng như đánh giá hiệu năng của chương trình phân tích cú pháp tiếng Việt.

* Giải thuật phân tích cú pháp A\* cho kết quả rất khả quan khi phân tích 630 câu hành văn trong thời gian 15 phút, tốc độ trung bình khoảng 3s/1 câu. Những câu này đều là những câu rất dài và khó.
* So với giải thuật CYK-Beam search, giải thuật A\* tỏ ra ưu thế hơn hẳn về mặt tốc độ. Về độ chính xác, do không đủ thời gian nên vẫn chưa có thử nghiệm cho chương trình. Nhưng trong tương lai nhất định sẽ có hoàn thành thử nghiệm để đánh giá hiệu năng của hệ thống một cách chuẩn xác hơn.
* Độ chính xác khi phân tích các câu mẫu trong tập TreeBank vẫn chưa được cao.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

**Những thành tựu mà đồ án đã đạt được:**

* Tìm hiểu và nghiên cứu rất các phương pháp phân tích cú pháp cũng như những mô hình trên thế giới để tìm ra hướng đi mới cho nhánh đề tài này.
* Tìm hiểu và kết hợp sử dụng bộ tách từ và bộ gán nhãn thành bộ tiền xử lí cho đầu vào của phân tích cú pháp.
* Thiết kế dữ liệu đầu ra cho hệ thống hỗ trợ các công đoạn khác trong tổng hợp tiếng nói có thể dễ dàng sử dụng.
* Xây dựng thành công giải thuật A\* áp dụng cho phân tích cú pháp tiếng Việt với tốc độ và độ chính xác khả quan.
* Đề xuất ra được ý tưởng về giải thuật lelightwin cắt tỉa giúp tăng tốc độ của hệ thống phân tích lên một tầm mới.

**Hướng phát triển của đồ án:**

Đồ án đã thử đem so sánh hệ thống với một số hệ thống phân tích cú pháp khác như hệ thống phân tích cú pháp PCFG của thầy Hoàng Anh Việt K46, hệ thống phân tích cú pháp học máy thống kê của VLSP. Kết quả cho thấy như sau:

* Chương trình phân tích của thầy Việt tốc độ rất nhanh do tập luật cú pháp đơn giản chỉ có 180 luật (trong khi bộ luật của hệ thống là 938 luật) tuy nhiên với những trường hợp câu phức tạp và lằng nhằng thì kết quả không mấy khả quan. Nhưng hệ thống của thầy Việt có một điểm mà hệ thống của đồ án cần phải học hỏi là có sử dụng giải thuật huấn luyện inside outside, giải thuật này sẽ giúp cải thiện chất lượng của bộ phân tích cú pháp lên rất nhiều.
* Chương trình PTCP của VLSP thật sự rất tốt, kết quả cho ra vô cùng khả quan. Nếu với những câu không có dấu phẩy, thì bộ phân tích cú pháp của VLSP tỏ ra ưu thế hơn hẳn so với hệ thống của đồ án. Nguyên nhân của kết quả này là do VLSP (GS.Hồ Tú Bảo) có tập TreeBank khổng lồ gồm 10.000 câu với bộ luật cú pháp đã được nghiên cứu rất kĩ lưỡng. Thêm vào đó, hệ thống tách từ và gán nhãn của họ đã được nghiên cứu rất khoa học và đã được thử nghiệm đảm bảo độ chính xác trên 90%.

Ngoài ra, mô hình PCFG mà hệ thống sử dụng vẫn chưa phải là tối ưu khi các trường hợp nhập nhằng cú pháp ở cấp độ từ vựng vẫn chưa thể giải quyết được. Về vấn đề này, mô hình LPCFG đã được nghiên cứu từ rất lâu và cho kết quả vô cùng khả quan trong việc xử lí nhập nhằng cấp độ từ vựng. Ngoài ra, với việc thêm thông tin của từ vào trong luật cú pháp, mô hình LPCFG có thể tận dụng được tối đa thông tin mà tập TreeBank mang lại, đó thực sự là một hướng phát triển đầy hứa hẹn.

Qua những điều đã phân tích ở trên, những hướng phát triển của đồ án trong tương lai sẽ là :

* Hoàn thành giải thuật lelightwin cắt tỉa.
* Kết hợp giải thuật A\* sử dụng mô hình LPCFG giúp nâng cao độ chính xác của bộ phân tích cú pháp.
* Xây dựng một tập VietTreeBank có quy mô lớn hơn hoặc tái sử dụng tập TreeBank của VLSP nhằm nâng cao chất lượng của dữ liệu huấn luyện cũng như bộ luật cú pháp.
* Hệ thống lại các nhãn từ loại và ngữ loại để sử dụng bộ vnTagger thay thế cho vnqtag.

Tài liệu tham khảo

1. **Fei Xia**, “*Inside-Outside algorithm*”, LING 572.
2. **Christopher D.Manning and Hinrich Schutze. “***Probabilistic Grammars*”, Chapter11, 1999.
3. **Michael Collins**, “*Head-Driven Statistical Models for Natural Language Parsing*”, MIT Computer Science and Artificial Intelligence Laboratory.
4. **Dan Klein and Christopher D. Manning**. 2003. “*A\* parsing: Fast exact Viterbi parse selection. In Proceedings of the Human Language Technology Conference and the North American Association for Computational Linguistics*”(HLT-NAACL).
5. **Dan Klein and Christopher D. Manning**. 2002. *“A\* parsing: Fast exact Viterbi parse selection”*. Technical Report dbpubs/2002-16, Stanford University, Stanford, CA.
6. **Adam Pauls and Dan Klein**, “*K-Best A\* Parsing*”, Computer Science Division University of California, Berkeley.
7. **Hoàng Anh Việt**, “*Phân tích cú pháp tiếng Việt sử dụng mô hình xác suất PCFG*”, đồ án tốt nghiệp đại học năm 2006.
8. **Phạm Thị Nhung**, “*Phân tích cú pháp tiếng Việt sử dụng beam search*”, đồ án tốt nghiệp đại học năm 2009.
9. **Đỗ Bá Lâm, Lê Thanh Hương**, “*Implementing a Vietnamese syntactic parser using HPSG*”, Khoa Công nghệ thông tin, trường Đại học Bách khoa Hà Nội.
10. **Diệp Quang Ban, Hoàng Văn Thung,** “*Ngữ pháp tiếng Việt*”, tập 1,2, Nhà xuất bản giáo dục, 1991-1992.
11. **Trung tâm khoa học xã hội và nhân văn Quốc Gia*. “****Ngữ pháp tiếng Việt*”***.*** Nhà xuất bản Khoa học Xã hội – 2000.
12. **Nguyễn Phương Thái, Vũ Xuân Lương, Nguyễn Thị Minh Huyền.** *“Xây dựng treebank tiếng Việt”.*