

**TRƯỜNG ĐẠI HỌC SÀI GÒN**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TIỂU LUẬN**

**SEMINAR CHUYÊN ĐỀ**

**LẬP TRÌNH LOGIC VÀ ỨNG DỤNG**

**ĐỀ TÀI:**

**XÂY DỰNG TINY-PARSER DỰA TRÊN VĂN PHẠM DCG (****DEFINITECLAUSE GRAMMAR) ĐỂ PHÂN TÍCH CÚ PHÁP**

**GIẢNG VIÊN HƯỚNG DẪN: PGS. TS. NGUYỄN TUẤN ĐĂNG**

**TÊN SINH VIÊN: LA CHÍ BÀNG MSSV: 3119410022**

**NĂM HỌC: 2022-2023**

1. **Giới thiệu đề tài:**
   1. **Giới thiệu:**

Ngôn ngữ lập trình Prolog được thiết kế để hỗ trợ việc xử lý thông tin dựa trên logic và tìm kiếm trong cơ sở tri thức. Để phát triển các ứng dụng Prolog phức tạp hơn, chúng ta cần có khả năng phân tích cú pháp (parser) để giúp máy tính hiểu được cấu trúc của ngôn ngữ đó. Chúng ta sẽ sử dụng văn phạm DCG (Definite Clause Grammar) để xây dựng các phân tích cú pháp cho ngôn ngữ tự nhiên hoặc các ngôn ngữ lập trình khác. DCG được sử dụng để xây dựng các quy tắc phân tích cú pháp dựa trên các quy tắc phát biểu và các ràng buộc logic.

Trong bài tiểu luận này, chúng tôi sẽ trình bày về việc xây dựng một TINY-PARSER dựa trên Văn phạm DCG để phân tích cú pháp cho các ngôn ngữ tự nhiên. Đây là một công cụ nhỏ gọn và đơn giản được xây dựng trên nền tảng Prolog, có thể dễ dàng tích hợp vào các ứng dụng Prolog phức tạp hơn.

* 1. **Mục tiêu:**

Mục tiêu của bài tiểu luận này là xây dựng một Tiny-Parser dựa trên Văn phạm DCG để phân tích cú pháp. Parser sẽ cho phép phân tích các câu đơn giản tiếng Việt và trả về cây phân tích cú pháp tương ứng. Các chức năng cơ bản sẽ bao gồm:

1. Phân tích cú pháp các câu tiếng Việt và tất cả các câu phái sinh trong tập ngữ liệu đã cho.
2. Trình bày cây phân tích cú pháp tương ứng.
   1. **Phạm vi:**

Phạm vi của đề tài "Xây dựng Tiny-Parser dựa trên văn phạm DCG để phân tích cú pháp" bao gồm:

1. Tìm hiểu và nghiên cứu về ngôn ngữ Prolog và văn phạm DCG.
2. Xây dựng một bộ phân tích cú pháp đơn giản dựa trên văn phạm DCG.
3. **Nội dung**
4. **Prolog**



Prolog là một ngôn ngữ lập trình. Tên gọi Prolog được xuất phát từ cụm từ tiếng Pháp Programmation en logique, nghĩa là "lập trình theo logic". Xuất hiện từ năm 1972 (do Alain Colmerauer và Robert Kowalski thiết kế), mục tiêu của Prolog là giúp người dùng mô tả lại bài toán trên ngôn ngữ của logic, dựa trên đó, máy tính sẽ tiến hành suy diễn tự động dựa vào những cơ chế suy diễn có sẵn (hợp nhất, quay lui và tìm kiếm theo chiều sâu) để tìm câu trả lời cho người dùng.

Prolog được sử dụng nhiều trong các ứng dụng của trí tuệ nhân tạo và ngôn ngữ học trong khoa học máy tính (đặc biệt là trong ngành xử lý ngôn ngữ tự nhiên vì đây là mục tiêu thiết kế ban đầu của nó). Cú pháp và ngữ nghĩa của Prolog đơn giản và sáng sủa, nó được người Nhật coi là một trong những nền tảng để xây dựng máy tính thế hệ thứ năm mà ở đó, thay vì phải mô tả cách giải quyết một bài toán trên máy tính, con người chỉ cần mô tả bài toán và máy tính sẽ hỗ trợ họ nốt phần còn lại.

Một chương trình Prolog bao gồm các luật được biểu diễn dưới dạng mệnh đề Horn. Một mệnh đề Horn có dạng

Head :- Body.

1. **Văn phạm DCG (DEFINITECLAUSE GRAMMAR)**
   1. **Khái quát**

Các ngữ pháp mệnh đề xác định (DCG) là những cách thuận tiện để biểu diễn các mối quan hệ ngữ pháp cho các ứng dụng phân tích cú pháp khác nhau. Chúng có thể được sử dụng cho công việc ngôn ngữ tự nhiên, để tạo các ngôn ngữ lập trình và lệnh chính thức.

* 1. **Cú pháp**

Ngữ pháp mệnh đề xác định (DCG) là một bộ tiền xử lý Prolog lấy các quy tắc ngữ pháp DCG và thêm các danh sách khác biệt được liên kết vào các mục tiêu.

DCG cung cấp một cú pháp để viết các quy tắc phân tích cú pháp ngữ pháp dễ đọc hơn mà không bao gồm các danh sách khác biệt được liên kết. Bộ tiền xử lý DCG lấy quy tắc DCG và dịch nó thành Prolog thuần túy, thêm vào các danh sách khác biệt được liên kết.

Cú pháp của DCG là:

* --> toán tử chỉ ra một quy tắc DCG, thay thế bình thường ( :- ) được sử dụng cho các mệnh đề Prolog. Bộ tiền xử lý thêm các đối số bổ sung cho các danh sách khác biệt. Ví dụ: quy tắc ngữ pháp câu có thể được viết mà không cần liệt kê sự khác biệt bằng quy tắc DCG:

câu --> chủ ngữ, động từ, tân ngữ.

* Mỗi mục tiêu được coi là tham chiếu đến phần đầu của quy tắc DCG và bộ tiền xử lý sẽ thêm hai đối số bổ sung cho danh sách khác biệt.
* Dấu ngoặc vuông [] , ký hiệu danh sách, được sử dụng để biểu thị các ký hiệu đầu cuối của ngữ pháp. Ví dụ:

danh từ -> [con mèo]

* 1. **Các bước xây dựng Tiny-Parser**

- Vẽ cây cú pháp:

+ Nam định mua mấy cuốn sách mới.

+ Nam có nhiều cuốn sách rất hay.

+ Nam mới tặng Lan một cuốn sách.

+ Nam thích đọc sách ở thư viện.

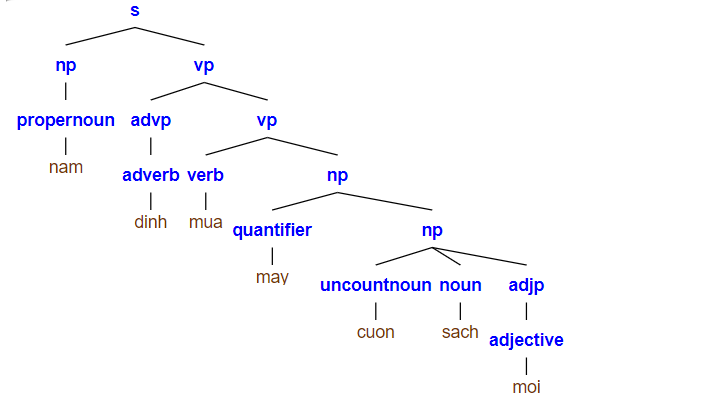
+ Nhà Nam ở gần trường.

- Chuyển sang văn phạm DCG từng câu.

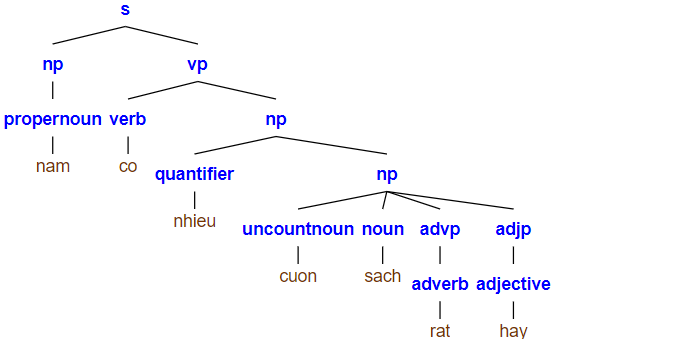
- Gộp văn phạm DCG để tạo tiny-parser hoàn chỉnh

* + - 1. **Vẽ cây cú pháp**

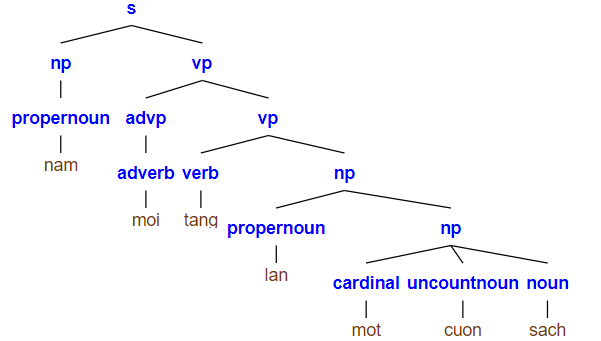
**a/ Nam định mua mấy cuốn sách mới**

****

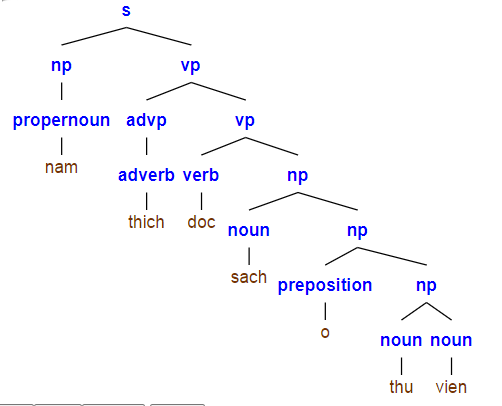
**b/ Nam có nhiều cuốn sách rất hay**

****

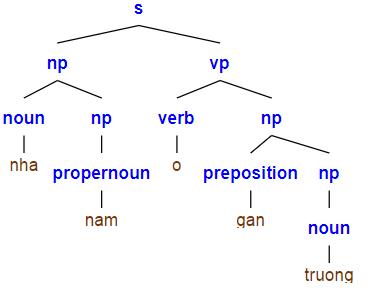
**c/ Nam mới tặng Lan một cuốn sách**

****

**d/ Nam thích đọc sách ở thư viện**

****

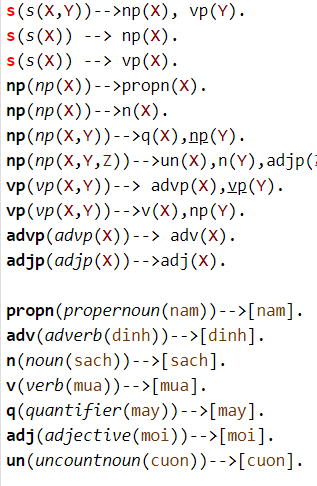
**e/ Nhà Nam ở gần trường**

****

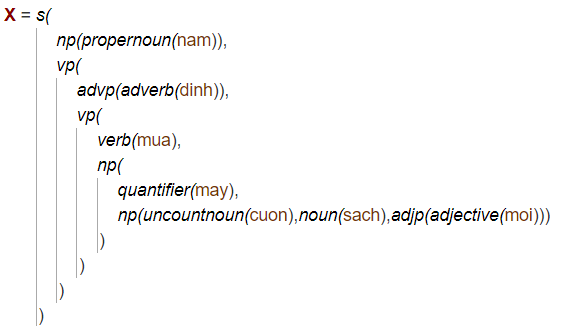
* + - 1. **Chuyển sang văn phạm DCG**

**a/ Nam định mua mấy cuốn sách mới**

Code prolog theo văn phạm DCG:

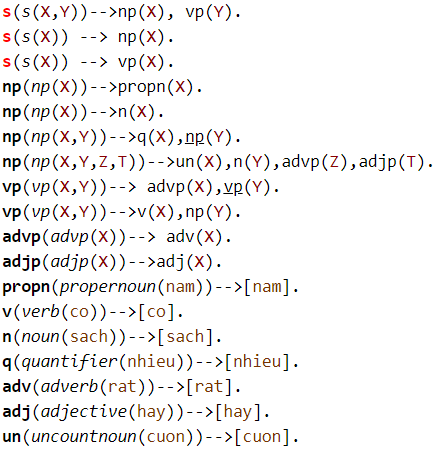
****

Kết quả phân tích:

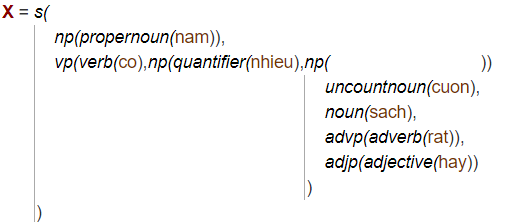
****

**b/ Nam có nhiều cuốn sách rất hay**

Code prolog theo văn phạm DCG:

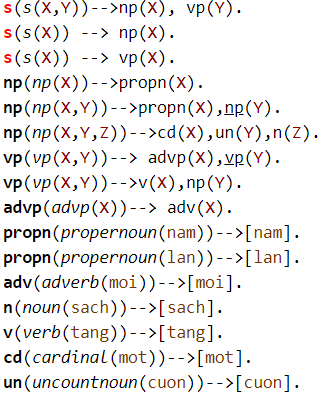
****

Kết quả phân tích:

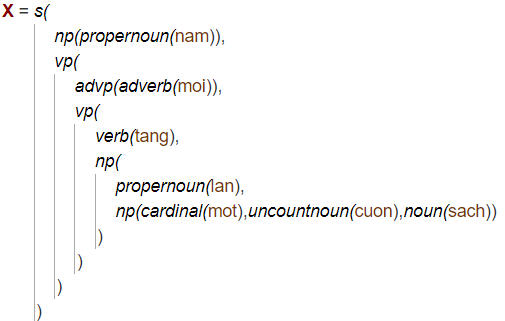
****

**c/ Nam mới tặng Lan một cuốn sách**

Code prolog theo văn phạm DCG:

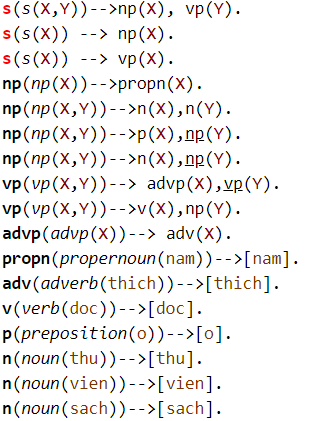
****

Kết quả phân tích:

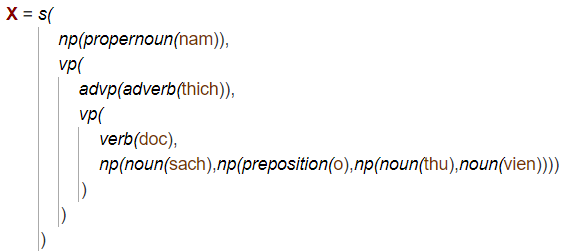
****

**d/ Nam thích đọc sách ở thư viện**

Code prolog theo văn phạm DCG:

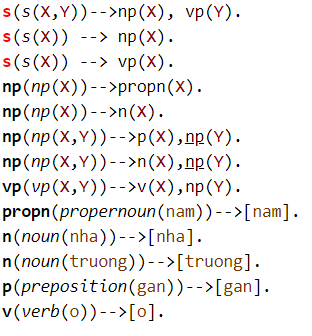
****

Kết quả phân tích:

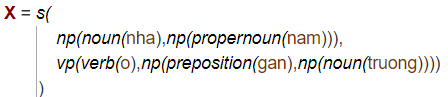
****

**e/ Nhà Nam ở gần trường**

Code prolog theo văn phạm DCG:

****

Kết quả phân tích:

****

* + - 1. **Gộp văn phạm DCG để tạo tiny-parser**

1. **Một số câu phái sinh từ tập ngữ liệu đã cho**

TÀI LIỆU THAM KHẢO

Prolog - [*https://vi.wikipedia.org/wiki/Prolog*](https://vi.wikipedia.org/wiki/Prolog)

DCG - https://www.amzi.com/manuals/amzi/pro/ref\_dcg.htm