XÂY DỰNG VĂN PHẠM TẠO GIAO DIỆN NGƯỜI DÙNG

BUILDING THE GRAMMAR TO CREATE USER INTERFACE

Hoàng Thị Mỹ Lệ[1], Trương Bách Tuệ[2]

[1]*Trường Đại học Sư phạm Kỹ thuật - Đại học Đà Nẵng; htmle@ute.udn.vn*

[2]*Đại học Đà Nẵng; tbtue@ac.udn.vn*

Tóm tắt - Trong xử lý ngôn ngữ tiếng Việt nói chung và tiếng dân tộc thiểu số nói riêng, một trong các thao tác thường gặp là phải tạo ra các công cụ để làm việc với kho ngữ vựng. Người dùng có thể cập nhật dữ liệu, xem nội dung, tìm kiếm, hay triển khai các ứng dụng thông qua giao diện của các công cụ. Cho đến nay, các kho từ vựng đa ngữ tiếng Việt vẫn chưa có cấu trúc thống nhất, nội dung chưa phong phú và không thuận tiện cho việc phát triển theo hướng mã nguồn mở. Mặt khác, vấn đề xây dựng các công cụ để thực hiện truy cập dữ liệu như vậy thường mất nhiều thời gian cũng như công sức. Các kết quả nghiên cứu của các nhà khoa học khó trao đổi được với nhau. Từ đó, bài báo đề xuất xây dựng văn phạm dùng để lập trình thiết kế giao diện và từ đó người dùng có thể làm việc với các kho ngữ vựng đa ngữ tiếng Việt có cấu trúc thống nhất.

Từ khóa - kho từ vựng đa ngữ; Xử lý dân tộc thiểu số; Giao diện người dùng; ANTRL; kho ngữ vựng Việt-Ê Đê.

Abstract - One of the common operations in the Vietnamese language processing in general and ethnic minority language processing in particular is to build tools to access vocabulary database. Through the interface of the tool, users can view content, search the information, update data or deploy applications. Currently, there are not many Vietnamese multilingual vocabulary database with consistent structure, rich content, diversity and easy exploitation in the direction of open source. On the other hand, building such tool to access data often takes a lot of time and effort, difficult to exchange between research groups and depending on the nature of the vocabulary database. From that the article proposed building the grammar for programmers to create interfaces so that users access to Vietnamese multilingual vocabulary database with predefined consistent structure.

Key words - Multilingual vocabulary database; ethnic minority language processing; user interface; ANTRL; vocabulary database Vietnamese.

1. Đặt vấn đề

Văn phạm tạo giao diện người dùng với các thành phần chính của các ứng dụng trong windows. Trên giao diện này, cho phép người dùng thực hiện kết nối, tương tác với kho ngữ vựng đa ngữ Việt - Dân tộc thiểu số và tạo ra khả năng làm việc có hiệu quả với kho ngữ vựng. Văn phạm đề xuất xây dựng trong bài báo được đặt tên VEDICL (Viet-Ede Interface Creating Language). Yêu cầu đặt ra trong văn phạm VEDICL là lập trình phải đơn giản, dễ hiểu, dễ học cho người lập trình, có tính trực quan, linh hoạt trong việc thiết kế các giao diện và không phụ thuộc vào các kho ngữ vựng. Văn phạm phải đầy đủ để xây dựng được các ứng dụng của người dùng. Trong nghiên cứu này, văn phạm VEDICL sử dụng bộ phân tích cú pháp của ANTLR (ANother Tool for Language Recognition) [1], để sinh mã từ các chương trình thiết kế giao diện người dùng. Giải pháp được đề xuất có ưu điểm là người dùng hoàn toàn có thể thiết kế giao diện theo mỗi ứng dụng để làm việc với các kho ngữ vựng có cấu trúc đã được thống nhất. Ngoài ra, người dùng cũng có thể thiết kế các ứng dụng trong xử lý ngôn ngữ tự nhiên [2]. Văn phạm VEDICL được phát triển trong môi trường nhận dạng ngôn ngữ ANTLR để tích hợp các công cụ lập trình, trình biên dịch để tạo mã Java, đóng gói sản phẩm, chạy thử nghiệm và xây dựng các ứng dụng theo mục đích của người dùng.

2. Tìm hiểu ANTLR

Phiên bản 4.8 mới nhất của ANTLR được phát hành đầu năm 2020 [1]. Ngữ pháp của ANTLR có thể bao gồm các ngôn ngữ chính như: C, C#, Java và Python. Các công cụ xử lý ngôn ngữ tự động trong ANTLR hỗ trợ lập trình viên tiết kiệm được thời gian cũng như công sức. Các trình biên dịch cũng được tạo ra từ các công cụ này. Chức năng mới trong ANTLR là công cụ phân tích cú pháp thông qua phân tích cú pháp LL.

Công cụ nhận dạng ngôn ngữ ANTLR được dùng trong việc xây dựng trình biên dịch sử dụng kỹ thuật phân tích LL(k)[1]. ANTLR đọc dữ liệu từ một tệp văn phạm, sau đó sinh ra các tệp nguồn và các tệp trung gian khác, các công cụ được ANTLR tạo ra, gồm có:

- Phân tích cú pháp: Thực hiện việc phân tích văn bản, một chuỗi các token được thực hiện cho việc xác định cấu trúc ngữ pháp dựa trên văn phạm. Phân tích cú pháp được dùng như một thuật ngữ trong xử lý ngôn ngữ tự nhiên [3].

- Phân tích từ vựng: Là quá trình thực hiện đọc một kí tự từ dữ liệu đầu vào, sử dụng các mẫu chỉ định để phân chia token và tạo ra dòng các token của dữ liệu ra. Một số token được đánh dấu bằng kí tự trắng và sử dụng chức năng phân tích cú pháp ANTLR để làm ẩn [3].

2.1. Phương pháp phân tích LL

Ý tưởng của phương pháp phân tích LL là khi gặp một kí hiệu không kết thúc, sẽ lựa chọn các dẫn xuất như thế nào đó để tránh việc quay lui làm mất thời gian. Tức là phải có một phương pháp nào đó để xác định được lựa chọn đúng mà không phải thử các lựa chọn khác. Thông tin để xác định lựa chọn là dựa vào trạng thái và kí hiệu kết thúc hiện tại. Phương pháp phân tích LL là một trong các phương pháp phân tích hiệu quả. Nó cũng thuộc phương pháp phân tích topdown, nhưng tối ưu hơn vì phương pháp phân tích của nó là không quay lui như phương pháp phân tích topdown [4].

L: Left to right (từ trái qua phải); L: Leftmost derivation (dẫn xuất ngoài cùng bên trái); k là số kí hiệu nhìn phía trước để đưa ra quyết định phân tích.

Giả sử kí hiệu không kết thúc A có các sản xuất: $A \rightarrow \alpha_1 \mid \alpha_2 \mid \alpha_3 \mid ... \mid \alpha_n$ thoả mãn tính chất các xâu

$\alpha_1, \alpha_2, \ldots, \alpha_n$, suy dẫn ra các xâu với kí hiệu tại vị trí đầu tiên là các kí hiệu kết thúc khác nhau. Khi chỉ cần nhìn vào kí hiệu đầu vào tiếp theo sẽ xác định được cần khai triển A theo α_i nào. Nếu cần tới k kí hiệu đầu tiên thì mới phân biệt được các xâu $\alpha_1, \alpha_2, \ldots, \alpha_n$, thì khi đó để chọn luật sản xuất nào cho khai triển A chúng ta cần nhìn k kí hiệu đầu vào tiếp theo.

Văn phạm LL(k) cho phép xây dựng bộ phân tích làm việc, có đặc điểm chung là xâu vào được quét từ trái sang phải và hoàn toàn xác định được quá trình phân tích, nếu bộ phân tích này nhìn được k kí hiệu nằm ngay bên phải của vị trí vào hiện tại. Ngôn ngữ sinh ra bởi văn phạm LL(k) là ngôn ngữ LL(k).

2.2. Phân tích từ vựng trong ANTLR

Quy tắc từ vựng: Từ vựng được qui định có tên phải bắt đầu bằng kí tự chữ hoa. Quy tắc từ vựng được xử lý như các quy tắc phân tích cú pháp. Do đó, có thể chỉ định các giá trị trả về và các đối số. Quy tắc từ vựng có thể có các biến cục bộ và gọi đệ quy.

Bỏ qua kí tự: Để có kí tự phù hợp và thiết lập các kí tự bỏ qua. Bỏ qua ký tự không có hiệu lực cho từ vựng để thiết lập lại và thử lại cho kí tự khác. Kí tự bỏ qua không được gửi lại để phân tích cú pháp.

Trả về giá trị: Các quy tắc sẽ tự động trả về mã thông báo với nội dung phù hợp với các quy tắc và các kí tự.

Chuỗi kí tự và từ khóa: Phần lớn các ngôn ngữ đều có chung một qui định là nhận dạng từ khóa và từ vựng. Đây là trường hợp đáng chú ý của các ngôn ngữ nhận dạng. Vấn đề này được công cụ nhận dạng ngôn ngữ ANTLR xử lý bằng cách đặt vào trong bảng chữ các từ khóa cố định, bảng chữ được kiểm tra sau khi mỗi từ khóa phù hợp. Do đó, các chuỗi kí tự sẽ ghi đè lên các mô hình nhận dạng tổng quát.

Quét file nhị phân: Chuỗi kí tự không có giới hạn số kí tự có trong bảng mã ASCII. Khi phân tích một tệp nhị phân có chứa số nguyên và chuỗi thì các lớp và các thiết lập từ vựng để có giá trị của 8 bit nhị phân sẽ được xác định đầu tiên.

Quét các kí tự trong bảng mã Unicode: Công cụ nhận dạng ngôn ngữ ANTLR nhận ra các kí tự trong bảng mã Unicode của các kí tự đầu vào, có nghĩa là không bị giới hạn 1 byte kí tự như trong bảng mã ASCII.

Tạo đối tượng Token: Cũng giống như trong phân tích cú pháp, quy tắc lexer có thể gọi các quy tắc khác. Công cụ ANTLR thực hiện gán nhãn cho các quy tắc lexer và nhận được một token thay cho văn bản, loại thẻ, số dòng,... phù hợp với quy tắc tham chiếu.

Điều kiện kết thúc tệp: Hàm CharScanner.uponEOF() được gọi là từ nextToken() ngay trước khi máy quét trả về một đối tượng EOF_TYPE, mã thông báo để thực hiện phân tích cú pháp.

2.3. Cây phân tích cú pháp trong ANTLR

Quy tắc ngữ pháp của cây: Là các qui ước EBNF (Extended Backus-Naur Form) kết hợp với hành động, cú pháp và vị ngữ. Mỗi phương án được thay thế danh sách các yếu tố. Các mục của văn phạm ANTLR là một yếu tố và bổ sung thường xuyên các yếu tố trong mô hình cây.

Cú pháp vị từ: Trong ANTLR cây phân tích cú pháp sử dụng lookahead. Đây là hình thức thiết kế trung gian. Tuy nhiên, các cấu trúc cây tương tự cần phải được nhận ra. Những hạn chế giới hạn cố định của lookahead có thể được khắc phục bằng cú pháp vị từ.

Dòng Token: Là một lexer và phân tích cú pháp cùng các đối tượng. Tuy vậy, nhận dạng ngôn ngữ và bản dịch được kế thừa từ xử lý kết nối giữa phân tích cú pháp và qui tắc từ vựng như là một dòng token.

Lọc dòng Token: Công cụ ANTLR có chức năng TokenStreamBasicFilter xử lý cho các tình huống. Do đó có thể thông báo để loại bỏ các loại thẻ mà không phải sửa đổi các lexer.

Tách dòng Token: Là gửi các yêu cầu phân tích cú pháp trên dòng token, thông báo đến phân tích cú pháp. Trong quá trình thực hiện nhận dạng, các thao tác sau đó có thể kiểm tra được các dòng ân để thu thập các yêu cầu.

Cơ chế StringTemple: Là tạo ra một cấu trúc dữ liệu trong cây phân tích cú pháp và sau đó trả về cho các ngôn ngữ khác. Có những cấu trúc dữ liệu được tách ra từ cây phân tích cú pháp.

3. Xây dựng văn phạm VEDICL

3.1. Kiến trúc mô hình lớp của VEDICL

Siêu mô hình hoá là một mô tả hình thức các khái niệm của một ngôn ngữ, cho phép xây dựng các công cụ và không có sự nhập nhằng. Siêu mô hình của VEDICL phân lớp các khái niệm của ngôn ngữ theo mức độ trừu tượng của lĩnh vực ứng dụng, đồng thời biểu diễn cấu trúc của ngôn ngữ. Kiến trúc mô hình 3 lớp của VEDICL thể hiện qua Bảng 1.

Bảng 1. *Mô hình 3 lớp của VEDICL*

Khái niệm trong văn phạm VEDICL	**Ví dụ cài đặt**
Siêu mô hình	
Định nghĩa ngôn ngữ đặc tả các siêu mô hình trong ANTLR	Giao diện, khung nhìn, nút công cụ hay nguồn ngữ liệu
Mô hình	
Cá thể của siêu mô hình định nghĩa lớp các đối tượng trong môi trường biên dịch (Java Eclipse)	My_ConnectXML("Dulieu\\Viet_Ede.xml"); My_Frame(f."Chuongtrinh",700,700); My_Panel(pnCenter);
Đối tượng của người dùng	
Một khung nhìn (cửa sổ) Một nút công cụ hay nguồn	Hình ảnh vật lý và các thao tác vật lý

3.2. Kịch bản sử dụng văn phạm VEDICL

Người sử dụng thiết kế ứng dụng bằng cách:

- Lập trình tạo ra giao diện tương tác với kho ngữ vựng.
- Cung cấp nguồn dữ liệu từ các kho ngữ vựng.
- Tiến hành biên dịch dùng công cụ ANTLR.

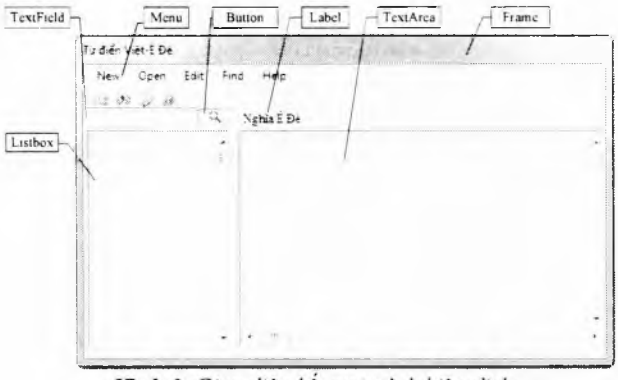
Chạy chương trình biên dịch sinh mã nguồn để nhận được ứng dụng mong muốn. Trình biên dịch thực hiện phân tích từ vựng và cú pháp, kết hợp trình biên dịch JDK của Java tạo ra mã bytecode chạy trên máy ảo Java [5].

Thực hiện kết nối truy cập tới kho ngữ vựng theo yêu cầu. Sau khi biên dịch, người sử dụng nhận được công cụ theo ứng dụng đã được lập trình.

Chương trình tạo giao diện được viết với văn phạm VEDICL [6].

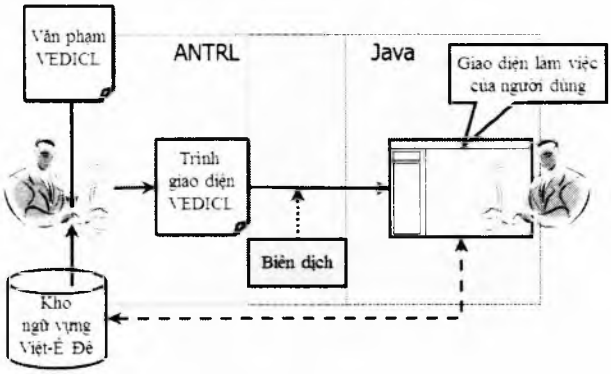
```
{
    _Frame(f,"Giao dien",700,700);
    _MenuBar(m, {n,"New"},{o,"Open"},{s, Save"},
                {e,"Edit"}, {f,"Find"}, {h,"Help"});
    _Panel(pnCenter);
    _Panel(pn);
    _Panel(pnlabel);
    _TextField(tf,15);
    _Button(b, {search, "icon\\search.gif"});
    _TextArea(ta,50,50);
    _List(lst);
    _Label(lb,"Nghĩa Ê Đê");
    _ToolBar(tb,{open,"icon\\open.gif"}, {find,
      "icon\\find.gif"}, {edit,"icon\\edit.gif"},
                        {help,"icon\\help.gif"});
}
```

Kết qua biên dịch chương trình ta được giao diện nhu Hình 1.



Hình 1. *Giao diện kết qua trình biên dịch*

Văn phạm VEDICL tương tác vào kho ngữ vựng Việt-Ê Đê được thê hiện trong Hình 2.

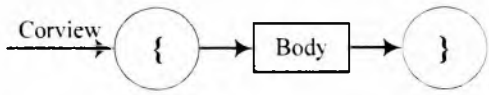


Hình 2. *Văn phạm VEDICL tương tác với vào kho ngữ vựng*

3.3. Các thành phần của văn phạm VEDICL

```
<program>::= <CorView>
<CorView>::= {<Body> }
<Body>::= [Body] <Statement>
<Statement>::=Keyword | [Identifier] <List_Identifier>; |
   <Identifier> <Position>;
<List_dentifier>::= {<Identifier>}
<Identifier>::= {<letter> | <digit>}
<Position>::= Left | Right | Top | Bottom
<letter>::= 'A'..'Z' | 'a'..'z'
<digit>::= '0'.. '9'
```

- Cú pháp câu lệnh CorView thê hiện trong Hình 3.

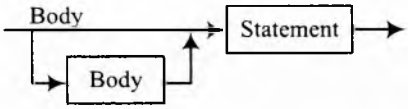


Hình 3. *Cú pháp câu lệnh CorView*

Cài đặt CorView trong ANTLR

```
CorView
    : "{" (fCorView=body) + "}"
    ;
```

- Cú pháp cua câu lệnh Body thê hiện trong Hình 4.

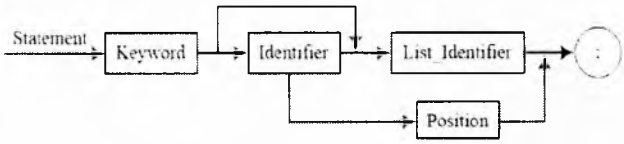


Hình 4. *Cú pháp câu lệnh Body*

Cài đặt Body trong ANTLR

```
Body
:
(
:
    fbody=statement
)+
;
```

- Cú pháp cua câu lệnh Statement thê hiện trong Hình 5.



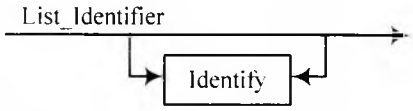
Hình 5. *Cú pháp câu lệnh Statement*

Cài đặt Statement trong ANTLR

```
Statement:
"_Frame"^LPAREN!idframe:IDENT
            (vect=list_Identifier)*RPAREN!
| "_Panel"^ LPAREN! idpanel:IDENT
            (vect=list_Identifier)* RPAREN!
| "_MenuBar"^ LPAREN! idmenubar:
            IDENT (vBody = bodymenubar)* RPAREN!
| " _Menu"^ LPAREN! idmenu:IDENT
           (vBodyItem = bodymenu)* RPAREN!
| "_TextField"^ LPAREN! idtextfield:IDENT
            (vect=list_Identifier)+ RPAREN!
| "_Label"^ LPAREN! idlabel:IDENT
            (vect=list_Identifier)+ RPAREN!
| "_TextArea"^ LPAREN! idtextarea:IDENT
            (vect=list_Identifier)+ RPAREN!
```

```
| "_ToolBar"^ LPAREN! idtoolbar:IDENT
        (vTool = bodytool)+ RPAREN!
| " _Button"^ LPAREN! idbutton:IDENT
        (vect=list_Identifier)+ RPAREN!
| " _List"^ LPAREN! idlist:IDENT
        (vect=list_Identifier)* RPAREN!
```

- Cú pháp của câu lệnh List_Identifier thể hiện trong Hình 6.

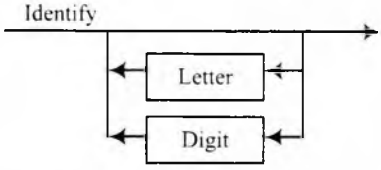


Hình 6. *Cú pháp câu lệnh List_Identifier*

Cài đặt List_Identifier trong ANTLR

```
List_Identifier
        :COMMA! ident:IDENT
        | COMMA! st:STRING
        | COMMA! in:INT
        |COMMA!po: position
```

- Cú pháp của câu lệnh Identify thể hiện trong Hình 7.

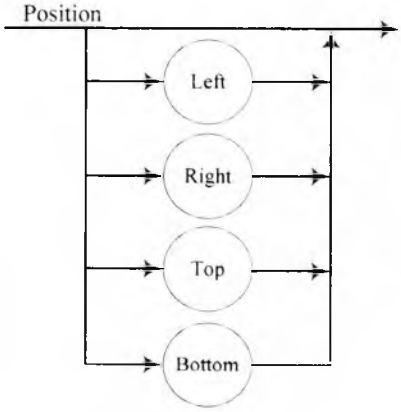


Hình 7. *Cú pháp câu lệnh Identify*

Cài đặt Identifier trong ANTLR

```
identifier
:id:IDENT |
s:STRING|
i:INT|
p: position
```

- Cú pháp của câu lệnh Position thể hiện trong Hình 8.



Hình 8. *Cú pháp câu lệnh Position*

Cài đặt Position trong ANTLR

```
position
:"Left"
|"Right"
|"Top"
|"Bottom"
;
```

4. Đánh giá giải pháp

Giải pháp đề xuất có được ưu điểm là người sử dụng có thể thiết kế một giao diện theo nhu cầu của các chức năng triển khai một ứng dụng, để làm việc với các kho ngữ vựng và hỗ trợ chức năng xác định cấu trúc của kho ngữ vựng.

Người sử dụng cũng có thể xây dựng ứng dụng bài toán xử lý ngôn ngữ tự nhiên trong môi trường Windows hoặc trang web liên quan đến xử lý tiếng dân tộc thiểu số như: Soạn thảo văn bản tiếng Ê Đê dùng phông chữ Unicode [7]; Quản lý kho từ vựng Việt-Ê Đê [8]; Dịch tự động những văn bản từ tiếng Việt sang tiếng Ê Đê trong ngữ cảnh hạn chế; Tra cứu nghĩa; Kiểm tra lỗi chính tả; Tìm kiếm, sắp xếp; Chuyển đổi văn bản (dạng Web, HTML, XML, ...).

5. Kết luận

Văn phạm VEDICL có thể tạo sinh các thành phần giao diện tiêu biểu thường gặp: Cửa sổ hội thoại (Frame, Panel, Menu, Toolbar), các nút công cụ giao tiếp (Button, Listbox, Text Area, Text Field, Button, Label). Khả năng nhận biết các KNL dạng XML có cấu trúc 3 cấp.

Đây chỉ là những kết quả nghiên cứu bước đầu, nhóm tác giả sẽ tiếp tục nghiên cứu và phát triển theo hướng:

- Hoàn thiện văn phạm VEDICL để có thể nhận biết cấu trúc dữ liệu nhiều cấp, có độ lớn và có nội dung phức tạp.

- Hoàn thiện các ứng dụng liên quan đến xử lý ngôn ngữ tiếng dân tộc thiểu số nói chung và tiếng Ê Đê nói riêng.

Lời cảm ơn: Nghiên cứu này được tài trợ bởi Bộ Giáo dục và Đào tạo trong đề tài Khoa học và Công nghệ có mã số B2019-DNA-01.

TÀI LIỆU THAM KHẢO

[1] https://www.antlr.org/.

[2] Phan Thị Tươi, *Xử lý ngôn ngữ tự nhiên*, NXB Đại học Quốc gia TP Hồ Chí Minh, 2012.

[3] Trần Thị Bích Hằng, *Nghiên cứu ứng dụng ANTLR để xử lý các thông điệp trong phần mềm đa ngữ*, Luận văn Thạc sĩ ngành Khoa học Máy tính, Đại học Đà Nẵng, 2013.

[4] Phạm Hồng Nguyên, *Giáo trình Chương trình dịch*, Nhà xuất bản Đại học Quốc gia Hà Nội, 2009.

[5] Đoàn Văn Ban, *Lập trình Java nâng cao*, NXB Khoa học và kỹ thuật, 2006.

[6] Phan Huy Khánh, Văn Thị Thu Hương, Thái Thu Hà, Lê Thị Thanh Thủy, "Phát triển công cụ tương tác với các kho ngữ liệu nhờ văn phạm tạo sinh giao diện", *Hội thảo Quốc gia lần thứ X về Một số vấn đề chọn lọc của Công nghệ Thông tin và Truyền thông*, 2007.

[7] Hoàng Thị Mỹ Lệ, Phan Huy Khánh, Souksan Vilavong, "Using Unicode in Encoding the Vietnamese Ethnic Minority languages, applying for the Ede language", *KSE 2013*, số 1, Trang 137-148, 2013.

[8] Hoàng Thị Mỹ Lệ, Phan Huy Khánh, "Xây dựng kho ngữ vựng song ngữ Việt-Ê Đê dựa trên mô hình tương tác Việt-Ê Đê", *Tạp chí Khoa học Công nghệ - ĐHĐN*, Số 5(114), quyển 2, trang: 36-40, 2017.

(BBT nhận bài: 02/4/2020, hoàn tất thủ tục phản biện: 20/5/2020)