ỨNG DỤNG CẤU TRÚC CHỒNG (STACK) ĐỂ KHỬ ĐỆ QUY KHI DUYỆT CÂY NHỊ PHÂN TÌM KIẾM

GVHD: ThS. Lê Minh Tự

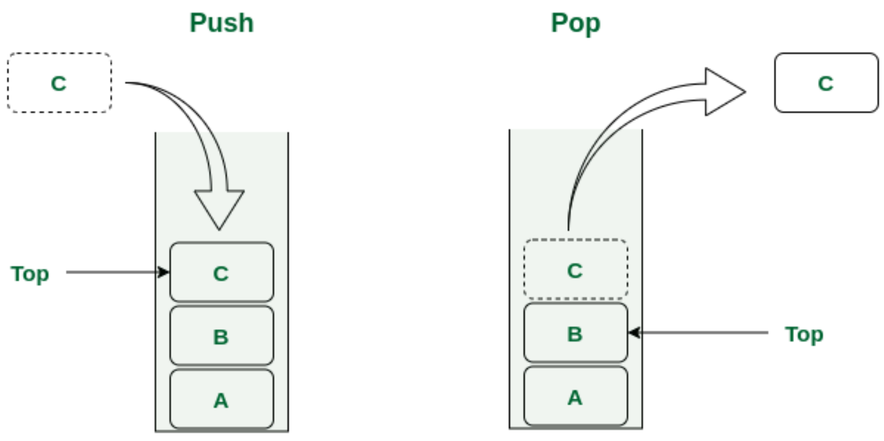
SVTH: Dương Bảo Khanh

TÓM TẮT ĐỀ TÀI

Đề tài "Ứng dụng cấu trúc chồng (Stack) để khử đệ quy khi duyệt cây nhị phân tìm kiếm" sử dụng Stack để thay thế đệ quy, giúp tối ưu hóa hiệu suất, tránh lỗi tràn bộ nhớ và đảm bảo tính chính xác của thuật toán duyệt cây.

PHƯƠNG PHÁP NGHIÊN CỨU

Đề tài triển khai ba phương pháp duyệt cây: Pre-order (NLR), In-order (LNR), và Post-order (LRN) bằng Stack, đảm bảo thứ tự duyệt chính xác và tránh phụ thuộc vào ngăn xếp hệ thống.



Hình 1: Minh họa Stack

THUẬT TOÁN

Thuật toán chính sử dụng cấu trúc chồng (Stack) để khử đệ quy trong các phương pháp duyệt cây nhị phân: Pre-order (NLR), In-order (LNR), và Post-order (LRN). Stack lưu trữ trạng thái các nút chưa xử lý, hoạt động theo cơ chế LIFO để đảm bảo thứ tự duyệt đúng. Trong mỗi bước, nút hiện tại được xử lý, sau đó lần lượt thêm các nhánh trái và phải vào Stack tùy theo yêu cầu của từng phương pháp duyệt. Thuật toán này giúp loại bỏ phụ thuộc vào đệ quy, tối ưu hóa hiệu suất và tránh lỗi tràn bộ nhớ.

KẾT QUẢ

```
===== MENU =====
1. Nhap du lieu thu cong de tao cay
2. Tao cay voi du lieu ngau nhien
3. Duyet cay
4. Thoat
Lua chon:
```

Hình 2: Giao diện màn hình chính

```
===== TAO CAY TU DONG =====
Nhap so luong nut: 5
Nhap khoang gia tri: 1  100
Da tao cay voi 5 phan tu ngau nhien.
```

Hình 3: Giao diện tạo cây tự động

```
===== MENU =====
1. Duyet Pre-order (NLR)
2. Duyet In-order (LNR)
3. Duyet Post-order (LRN)
Lua chon:
```

Hình 4: Giao diện chọn phương pháp duyệt

```
===== MENU =====
1. De quy
2. Khu de quy
3. Ca hai
Lua chon:
```

Hình 5: Giao diện chọn cách duyệt

```
===== KET QUA =====
Duyet Pre-order (De quy):       49      47      61      54      77
Duyet Pre-order (Khu de quy): 49        47      61      54      77
```

Hình 6: Giao diện hiển thị kết quả

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Đề tài đã triển khai thành công việc sử dụng cấu trúc chồng (Stack) để khử đệ quy trong các phương pháp duyệt cây nhị phân, đảm bảo tính chính xác và hiệu quả. Trong tương lai, hướng phát triển có thể bao gồm tối ưu hóa thuật toán để xử lý cây dữ liệu lớn nhanh hơn và mở rộng các ứng dụng thực tế, như áp dụng trong hệ thống tìm kiếm, quản lý dữ liệu, hoặc tích hợp với các công cụ phân tích dữ liệu trực quan.