HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Phạm Quỳnh Điệp

THUẬT TOÁN CHUYỂN CÂU SQL TỪ CHƯƠNG TRÌNH NGUỒN SANG AQL

Chuyên ngành: Khoa học máy tính
Mã số: 60.48.01

NGƯỜI HƯỚNG DẪN KHOA HỌC :
PGS. TS. Lê Huy Thập

LUẬN VĂN THẠC SĨ KỸ THUẬT

HÀ NỘI - 2012

LỜI MỞ ĐẦU

Câu truy vấn đã được tối ưu sẽ nâng cao hiệu quả nếu nó thỏa mãn một số tiêu chuẩn cho trước nào đó. Một số phương pháp phân tích và tối ưu hóa các câu truy vấn dạng SQL và AQL đã được một số tác giả trong và ngoài nước nghiên cứu. Một số thuật toán cũng đã được công bố. Tuy nhiên tất cả đều dựa trên giả thuyết các câu vấn tin dạng SQL được lấy trực tiếp từ một chương trình nguồn hay người lập trình viết khi lập trình và tối ưu hóa một cách thủ công. Một vấn đề đặt ra là một chương trình nguồn dạng tuần tự mà trong đó có nhiều lệnh SQL có thể thỏa mãn điều kiện song song hóa và chỉ được song song hóa và tối ưu hóa bởi phương pháp song song tự động. Điều đó gây nhiều bất cập trong ứng dụng. Việc tìm kiếm trong chương trình nguồn các lệnh SQL và sau đó chuyển sang AQL để hỗ trợ tối ưu hóa vấn tin là một vấn đề thời sự và cần thiết. Trong khuôn khổ của luận văn các vấn đề sẽ lần lượt trình bày trong các chương.

Chương 1, sẽ trình bày các kiến thức cơ bản về một số phần mềm tìm kiếm, tổng quan cơ sở dữ liệu (CSDL) phân tán, câu vấn tin SQL, AQL và cây vấn tin, quá trình tối ưu hóa và ngôn ngữ lập trình Java.

Chương 2, sẽ trình bày các thuật toán tìm câu vấn tin SQL, tạo câu vấn tin AQL và cây toán tử.

Chương 3, xây dựng chương trình demo tìm các câu vấn tin SQL từ một chương trình nguồn rồi sau đó chuyển các câu SQL này sang câu vấn tin AQL.

Chương 1: TỔNG QUAN

1.1. Các phần mềm tìm kiếm cơ bản

Các phần mềm tìm kiếm là "chìa khóa" quan trọng để giúp bạn tìm thấy thông tin cụ thể mà bạn cần trong vô số những dữ liệu trên mạng toàn cầu.

Các công cụ tìm kiếm dựa trên chương trình tự động

Những công cụ tìm kiếm tự động, ví dụ Google sẽ tạo ra những danh sách của họ tự động. Các chương trình tự động như crawler hay spider sẽ bắt đầu làm việc, sau đó mọi người có thể tìm kiếm thông qua những gì mà các chương trình tự động dò tìm được.

1.1.1. Google search:

1.1.2. Yahoo search

1.1.3. Một số lệnh tìm kiếm trong các ngôn ngữ lập trình bậc cao

1.2. Tổng quan CSDL phân tán

Những năm của thập kỷ 70, máy tính đã có đủ khả năng xây dựng hệ thống tin và hệ cơ sở dữ liệu. Một mặt đã hình thành và phát triển các mô hình lý thuyết cho hệ cơ sở dữ liệu và mặt khác những nguồn phát triển hệ thống ứng dụng ngày càng có nhiều kinh nghiệm. Hệ thống thông tin hình thành trên cơ sở kết nối các máy tính khác nhau.

Những năm gần đây, hệ cơ sở dữ liệu phân tán được phát triển dựa trên cơ sở dữ liệu và mạng máy tính. Cơ sở dữ liệu phân tán gồm nhiều cơ sở dữ liệu tích hợp lại với nhau thông qua mạng máy tính để trao đổi dữ liệu, thông tin... Cơ sở dữ liệu được tổ chức và lưu trữ ở những vị trí khác nhau trong mạng máy tính và chương trình ứng dụng làm việc trên cơ sở truy cập dữ liệu ở những điểm khác nhau đó.

Vấn đề hoàn toàn mới là xây dựng và cài đặt một cơ sở dữ liệu phân tán. Cần giải quyết vấn đề xây dựng và cài đặt cơ sở dữ liệu phân tán cụ thể như vấn đề thiết kế phân tán, thiết kế cơ sở dữ liệu...

1.2.1. Khái niệm về cơ sở dữ liệu phân tán

1.2.2. Lợi điểm của cơ sở dữ liệu phân tán

1.3. Câu vấn tin SQL, AQL và cây vấn tin

1.3.1. Câu vấn tin SQL (Structured Query Language)

Đây là một loại ngôn ngữ con dữ liệu quan hệ được xác nhận lầ rất mạnh. Phép toán cơ bản trong SQL là phép ánh xạ, được mô tả về cú pháp như là khối SELECT - FROM - WHERE.

Mệnh đề SELECT nghĩa là chọn các thuộc tính ra, hay còn gọi là thuộc tính kết quả, nếu không chỉ ra thuộc tính thì dùng dấu * có nghĩa là tất cả các thuộc tính của quan hệ đang được chỉ ra sau mệnh đề FROM. Mệnh đề FROM để chỉ ra quan hệ cần cho việc xử lý. Sau mệnh đề WHERE là một biểu thức điều kiện lọc dữ liệu (hay còn gọi là biểu thức logic).

1.3.2. Ngôn ngữ truy vấn đại số (AQL)

Gọi r là quan hệ trên tập thuộc tính $R=\{ A_1,….,A_n\}$. Ta luôn giả thiết rằng quan hệ r là tập hữu hạn các bộ. Đối với các phép hợp (ký hiệu $\cup$) phép giao (ký hiệu $\cap$), phép trừ (ký hiệu -) hai quan hệ tham gia phải khả hợp

1.3.2.1. Phép hợp: Ký hiệu: $\cup$

Hợp của hai quan hệ r và s ký hiệu là $r \cup s$ là tập các bộ hợp thuộc r hoặc thuộc s hoặc thuộc r lẫn thuộc s:

$$r \cup s = \{ t: t \in r \text{ hoặc } t \in s \text{ hoặc } t \in r \text{ và } t \in s \}$$

1.3.2.2. Phép giao: Ký hiệu: $\cap$

Giao của hai quan hệ r và s ký hiệu là $r \cap s$ là tập các bộ phải vừa thuộc r phải vừa thuộc s:

$$r \cap s = \{t: t \in r \text{ và } t \in s\}$$

1.3.2. 3. Phép trừ: Ký hiệu: -

1.3.2.5. Phép chiếu (Projection)

1.3.2.6. Phép chọn (Selection)

1.3.2.7. Phép kết nối (Join)

1.3.2.8. Phép chia

1.3.2.9. Các ví dụ về tìm kiếm bằng đại số qua hệ

1.3.3. Cây vấn tin

Cây vấn tin làm nhiệm vụ giải thích phương án thi hành một câu SQL: Cho biết thứ tự thực hiện mỗi phép toán, phương pháp tính toán mỗi toán tử. Mỗi nút của cây là một hay nhiều phép toán đại số quan hệ, mỗi nút lá là một quan hệ cơ sở. Phần ghi chú trên mỗi nút mô tả cách thức thực hiện toán tử gì trên đó..

1.4. Sắp sếp lại phép nối và viết lại câu vấn tin

1.4.1. Sắp sếp lại phép nối

1.4.2. Viết lại câu vấn tin

1.5. Quá trình tối ưu hóa

Tối ưu hóa vấn tin phân tán là phương pháp lựa chọn phương án thực hiện câu vấn tin (QEP Query Executtion Plan) phân tán tốt nhất theo nghĩa có chi phí thấp nhất trong số các phương án có khả năng được thực hiện bởi thể vấn tin.

Chi phí thực hiện được diễn tả bởi hàm chi phí, thường được xem là hàm mục tiêu. Nó bao gồm chi phí xuất nhập, chi phí xử lý tại các CPU và chi phí truyền thông tin. Một đơn giản hoá điển hình của các thể tối ưu hoá vấn tin phân tán ban đầu là bỏ qua chi phí xử lý cục bộ (chi phí xuất nhập và chi phí CPU) bằng cách giả thiết rằng chi phí truyền dữ liệu chiếm ưu thế. Dữ liệu vào của thể tối ưu hóa vấn tin là các số liệu thống kê của các mảnh và các công thức đánh giá lực lượng của các quan hệ trung gian được tạo ra. Trong chương này chúng ta tập trung chủ yếu vào vấn đề sắp thứ tự các phép nối trong câu vấn tin phân tán vì: Phép nối thường là phép toán giảm dữ liệu trung gian, và vì các câu vấn tin có ,chứa nối, chọn và chiếu thường là loại vấn tin hay gặp nhất. Hơn nữa chúng ta dễ dàng tổng quát hóa thuật toán cơ bản này cho các câu vấn tin có các phép toán hai ngôi như phép hợp.

1.6. Giới thiệu về ngôn ngữ Java

1.6.1. Khái niệm

Java (đọc như “Gia-va”) là một ngôn ngữ lập trình dạng lập trình hướng đối tượng (OOP). Khác với phần lớn ngôn ngữ lập trình thông thường, thay vì biên dịch mã nguồn thành mã máy hoặc thông dịch mã nguồn khi chạy, Java được thiết kế để biên dịch mã nguồn thành bytecode, bytecode sau đó sẽ được môi trường thực thi (runtime environment) chạy. Bằng cách này, Java thường chạy nhanh hơn những ngôn ngữ lập trình thông dịch khác như Python, Perl, PHP,…

Cú pháp Java được vay mượn nhiều từ C & C++ nhưng có cú pháp hướng đối tượng đơn giản hơn và ít tính năng xử lý cấp thấp hơn.

1.6.2. Lịch sử hình thành ngôn ngữ Java

1.6.3. Một số đặc điểm nổi bật của ngôn ngữ lập trình Java

1.6.3.2 Thông dịch:

1.6.3.3 Độc lập nền:

1.6.3.4 Hướng đối tượng:

1.6.3.5 Đa nhiệm – đa luồng (MultiTasking – Multithreading):

1.6.3.5 Khả chuyển (portable):

1.6.3.6 Hỗ trợ mạnh cho việc phát triển ứng dụng:

1.7. Kết luận chương

Chương 1 trình bày những kiến thức cơ bản về cớ sở dữ liệu phân tán, câu vấn tin SQL, câu vấn tin AQL và cây vấn tin, sắp sếp lại phép nối và viết lại câu vấn tin, quá trình tối ưu hóa và ngôn ngữ lập trình Java.

Trên đây là cơ sở lý thuyết nền tảng cho qua trình tối ưu hóa vấn tin. Chương 2 sẽ trình bày các thuật toán tìm câu vấn tin SQL từ chương trình nguồn, chuyển câu vấn tin SQL sang câu vấn tin AQL và thuật toán vẽ cây toán tử.

Chương 2: THUẬT TOÁN CHUYỂN CÁC CÂU SQL TỪ CHƯƠNG TRÌNH NGUỒN SANG AQL

2.1. Mở đầu

Câu vấn tin đã được tối ưu sẽ nâng cao hiệu quả nếu nó thỏa mãn một số tiêu chuẩn cho trước nào đó. Một số phương pháp tối ưu hóa được dùng trực tiếp ngay trên câu SQL, nhưng nó mang tính chất phương pháp luận hơn ứng dụng, đó là tối ưu động Ingres hay Ingres – QOA,…. Tuy nhiên để chỉ ra các câu vấn tin SQL cụ thể có tồn tại trong chương trình nguồn thì chưa được chỉ ra. Dưới đây sẽ trình bày phương pháp tìm kiếm vét cạn trong chương trình nguồn để tìm ra các câu SQL, chuyển nó sang AQL, dựng cây toán tử và dùng một số phương pháp để tối ưu câu vấn tin AQL tức là SQL là rất cần thiết.

2.2. Một số phương pháp tối ưu hóa câu vấn tin cơ sở dữ liệu tập trung

Thuật toán INGRES [1].

Câu vấn tin q có *n* quan hệ, INGRES phân rã p thành $q_1 \rightarrow q_2 \rightarrow \ldots \rightarrow q_m$. Phân rã này sử dụng hai kỹ thuật cơ bán: *Tách* và *Thế bộ*.

Thuật toán System R [1]

Thuật toán System R thực hiện tối ưu hóa tĩnh bằng cách tìm kiếm vét cạn không gian các phương án. Đầu vào của System R là cây toán tử do phân rã câu vấn tin SQL. Đầu ra là phương án thực hiện để cài đặt cây toán tử tối ưu.

2.3. Giới thiệu một số tối ưu hóa câu vấn tin cơ sở dữ liệu phân tán

Có ba thuật toán tối ưu hóa vấn tin cơ bản: thuật toán rút gọn của hệ INGRES phân tán, thuật toán của System R* và thuật toán của SDD-1. Các thuật toán này đại diện cho nhiều lớp thuật toán khác nhau và vì thế thường được dùng những mô thức (hình dáng và cách thức).

2.3.1 Thuật toán INGRES phân tán

*2.3.2 Thuật toán System R**

2.3.3 Thuật toán SDD-1

2.4. Các thuật toán tìm câu vấn tin SQL, tạo câu vấn tin AQL và cây toán tử

Câu vấn tin dạng SQL có cấu trúc SELECT ... FROM ... WHERE ...

Vì tất cả các câu mệnh đề đều có thể chuyển về dạng chuẩn AND (Tương đương với chuẩn hội, nên chúng ta có các mệnh đề chuẩn AND)

Câu vấn tin dạng AQL là chuyển đổi các phép toán trong SQL sang phép toán của đại số quan hệ. Một SQL có nhiều AQL tương ứng. Việc chọn lựa AQL nào để thực hiện là dựa vào khả năng tối ưu hóa AQL.

Cây toán tử là việc thể hiện của các câu AQL bằng cây.

Cách biến đổi câu vấn tin phép tính quan hệ trở thành một cây toán tử như sau:

Trước hết tạo ra các nút lá là các quan hệ trong SQL các nút lá nằm sau FROM.

Nút gốc được tạo ra như phép chiếu chứa các thuộc tính kết quả, các thuộc tính này nằm sau SELECT.

Lượng tử hoá (vị từ sau WHERE) được chuyển thành các phép tính quan hệ thích hợp (phép chọn, phép nối ,…) đi từ các nút lá đến gốc. Chuỗi này có thể được cho trực tiếp qua thứ tự xuất hiện của các vị trí và các toán tử.

2.4.1. Thuật toán tìm câu SQL trong chương trình nguồn.

Thuật toán này tìm kiếm và vét cạn tất cả các câu vấn tin dạng SQL trong chương trình nguồn và đưa vào hàng đợi chờ xử lí.

TimKiem_SQL

Vào: Chương trình nguồn

Ra: Các câu truy vấn SQL

While NOT EOF

If <Còn câu lệnh SQL>

Push SQL ***Into*** QueueSQL

End If

Continue

End While

Bổ đề 1

Tất cả các câu lệnh dạng SQL đều được đưa vào hàng đợi QueueSQL. Thuật toán có điểm dừng.

2.4.2. Thuật toán chuyển SQL sang AQL

Thuật toán lấy các câu SQL đã được lưu trữ trong hàng đợi QueueSQL, chuyển đổi các kí hiệu và phép toán trong quan hệ sang đại số quan hệ sau đó lưu vào hàng đợi QueueAQL

CH_SQL_AQL

Vào: Hàng đợi SQL

Ra: Hàng đợi AQL các kí hiệu câu truy vấn AQL

While QueueSQL ≠ ∅

Pop QueueSQL ***Save Into*** Tam.txt

Do while NOT EOF()

Push $\pi_{(\text{các thuộc tính sau SELECT})}$ ***Into*** QueueAQL

Repeat

Find <Phép chọn>

If <Nếu phép chọn sẽ có dạng <Tên Quan hệ>.<Tên trường><Phép toán quan hệ> <Value>

Push $\sigma_{\text{<Tên trường> <Phép toán đại số quan hệ> <Value>}}$ ***Into*** QueueAQL

End if

If <Nếu phép chọn sẽ có dạng <Tên Quan hệ>.<Tên trường> <Phép toán quan hệ> <Biểu thức mệnh đề quan hệ>>

Trong biểu thức mệnh đề thay "AND" bởi "∧" còn "OR" bởi "∨" để được

<Biểu thức mệnh đề đại số quan hệ >

Push $\sigma_{<\text{Tên trường}><\text{Phép toán đại số quan hệ}><\text{Biểu thức mệnh đề đại số quan hệ}>}$ ***Into*** QueueAQL

End if

Until <Không còn phép chọn>

Repeat

Find <Phép nối>

//Phép nối sau Where là các dòng lệnh hoặc phần câu lệnh có dạng

// <Tên Quan hệ 1>.<Tên Khóa> <Phép toán quan hệ>

// <Tên Quan hệ 2>.<Tên Khóa 2>.

Replace <Tên Quan hệ 1>.<Tên Khóa1><Phép toán quan hệ><Tên Quan hệ 2>.<Tên Khóa 2> ***By*** <Tên Quan hệ 1> $\bowtie_{<\text{Tên Khóa1}><\text{Phép toán đại số quan hệ}>.<\text{Tên Khóa2}>}$ <Tên Quan hệ 2>

If <<Tên Quan hệ 1> AND <Tên Quan hệ 2> không có ở bước trước>

Push <Kết quả> ***Into*** QueueAQL

Else

If <Tên Quan hệ 1> có ở bước trước>

Pop & $\bowtie_{<\text{Phép toán đại số quan hệ}>.<\text{Tên Khóa2}>}$ <Tên Quan hệ 2>

Push <Kết quả> ***Into*** QueueAQL

Else

<Tên Quan hệ 1> $\bowtie_{<\text{Tên Khóa1}><\text{Phép toán đại số quan hệ}>}$ ***& Pop &*** $\bowtie_{<\text{Phép toán đại số quan hệ}>.<\text{Tên Khóa2}>}$ <Tên Quan hệ 2>

Push <Kết quả> ***Into*** QueueAQL

End if

End if

Until <Không còn phép nối>

Push <Kí hiệu hết lệnh AQL >

End While

Chú ý rằng tập các phép toán quan hệ gồm {=, < , <= , > , >= , NOT, AND, OR} tương ứng với phép toán đại số quan hệ {=, < , ≤ , > , ≥ , ¬ , ∧ , ∨}

2.4.3. Thuật toán Tạo AQL

Thuật toán được dùng để thể hiện vâu truy vấn AQL

Tạo AQL

Vào: QueueAQL

Ra: Các câu vấn tin dạng AQL trong hàng đợi QueueVT_AQL

While QueueAQL ≠ ∅

Str = “”

Count = 0

Repeat

Str = Str & “(“ & *Pop*

Count = Count +1

Until <Kí hiệu hết lệnh AQL >

For i = 1 ***To*** Count

Str = Str & “)”

End for

Push <Kết quả> ***Into*** QueueVT_AQL

End While

2.4.4. Thuật toán Vẽ Cây Toán Tử

Thuật toán được dùng để vẽ cây toán tử đã có trong QueueVT_AQL

Vào: QueueVT_AQL

Ra: Cây toán tử

Output (“Cần vẽ cây toán tử thứ k = “)

Input k

While (1)

For i = 1 To k -1

If QueueVT_AQL ≠ ∅

Pop

Else

Exit While

End if

End for

Pop Into StrXau // Lấy câu AQL thứ k ra khỏi hàng đợi và cho vào biến //StrXau

StrXau = ***Right***(StrXau, ***length*** (strXau) - 1

l = ***length*** (strXau)

For i = 0 ***To*** l

If strXau(i) = "("

XauVe = Left(StrXau, i)

Vẽ một điểm và gắn nhãn XauVe

StrXau = ***Right***(StrXau, l – i)

l = ***length*** (strXau)

Exit While

End While

Ví dụ

Cho CSDL gồm các quan hệ

NV(MaNV, TenNV, CN)

DuAn(MaDA, TenDA, NS)

BL(CN, Luong)

PN(MaNV, MaDA, TG, CV)

Trong đó

NV – Nhân viên: MaNV – Mã nhân viên, TenNV – Tên nhân viên, CN – Chuyên ngành.

DuAn – Dự án: MaDA – Mã dự án, TenDA – Tên dự án, NS – Ngân sách.

BL – Bảng lương: CN – Chuyên ngành, Luong – lương.

PN – Phân nhiệm: MaNV – Mã nhân viên, MaDA – Mã dự án, TG – thời gian làm việc, CV -chức vụ.

Gỉa sử trong tệp chương trình nguồn có lệnh SQL:

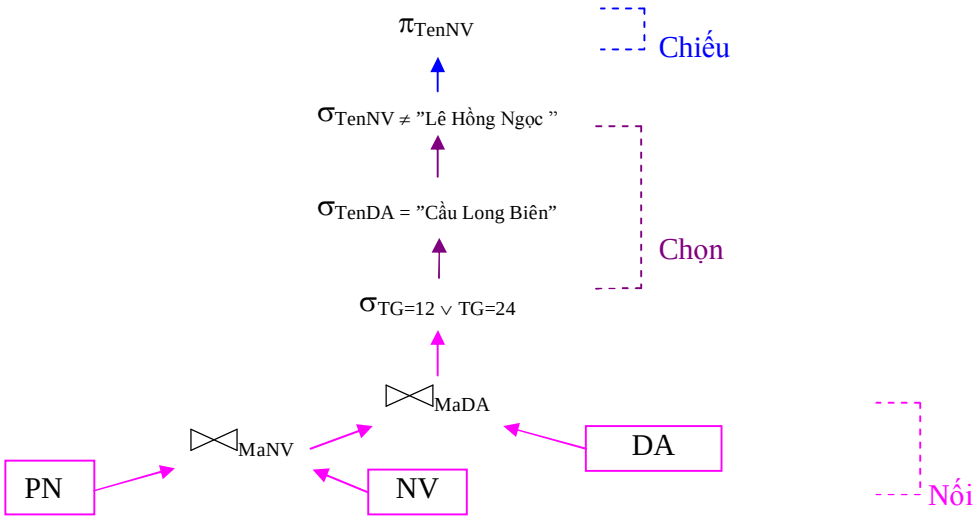
```
SELECT    TenNV   {Gốc }
FROM      NV , DuAn , PN
WHERE     PN.MaNV = NV.MaNV
AND       PN.MaDA = DuAn.MaDA
AND       TenNV  <>  “Lê Hồng Ngọc”
AND       TenDA =  “Cầu Long Biên”
AND       (TG=12 OR TG=24)
```

Nghĩa là: Tìm tên các nhân viên, trừ Lê Hồng Ngọc đã làm cho dự án “Cầu Long Biên” trong 12 tháng hoặc 24 tháng.

Sau khi áp dụng các thuật toán trên chúng ta được câu lệnh AQL là:

$\pi_{TenNV}(\sigma_{TG=12 \vee TG=24}(\sigma_{TenDA = \text{"Cầu Long Biên"}}(\sigma_{TenNV \neq \text{"Lê Hồng Ngọc "}}(DA \bowtie_{MaDA} (PN \bowtie_{MaNV} NV)))))$

Còn cây toán tử như sau



Hình 1.5. Ví dụ về cây toán tử

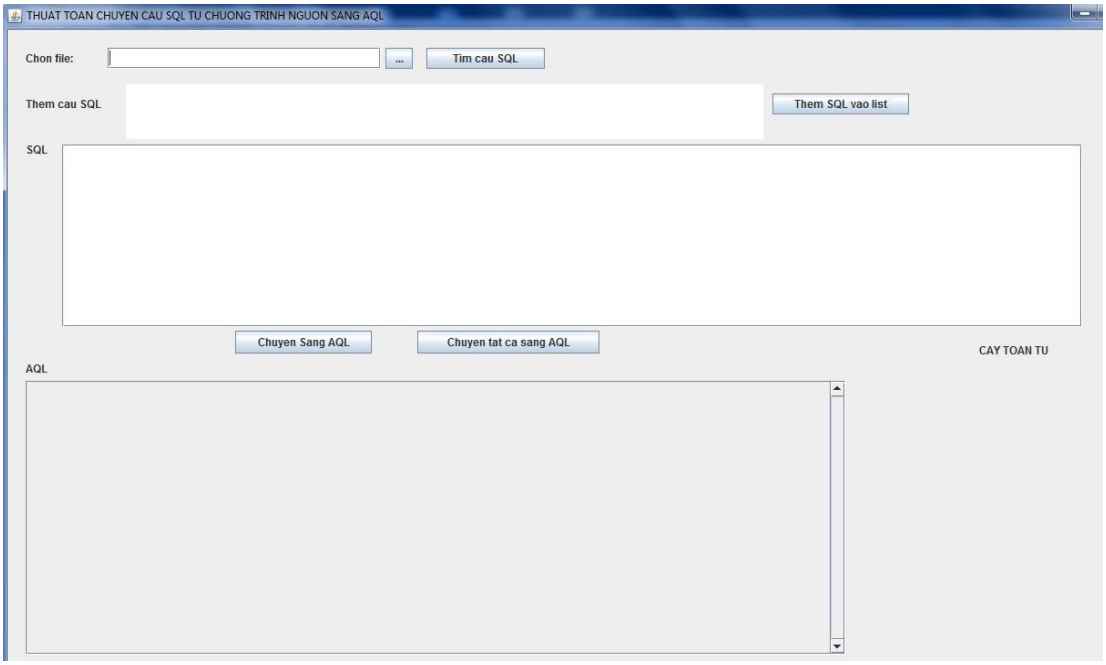
2.5. Kết luận

Chương 2 giới thiệu các thuật toán tối ưu hóa câu vấn tin, cơ sở dữ liệu phân tán, các thuật toán tìm câu vấn tin SQL trong chương trình nguồn, chuyển câu vấn tin SQL sang câu vấn tin AQL và thuật toán vẽ cây toán tử.

Chương 3: CHƯƠNG TRÌNH ĐỀ MÔ

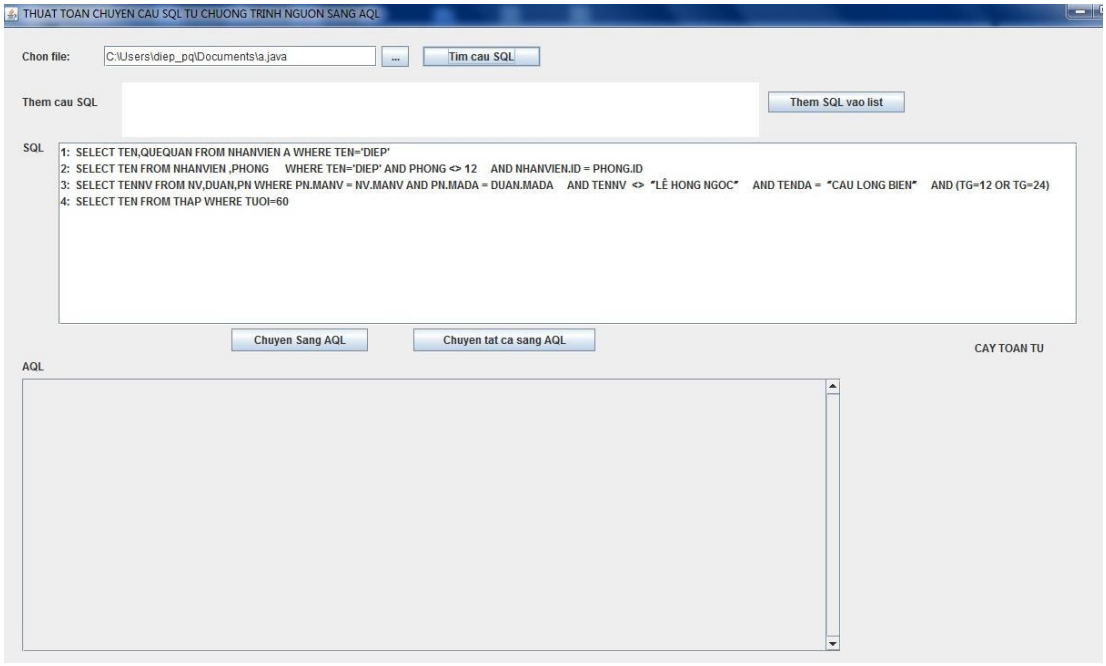
Qua nghiên cứu cơ sở lý thuyết ở chương 1 và các thuật toán ở chương 2, chương 3 xây dựng chương trình đề mô. Tìm tất cả các câu vấn tin SQL từ một chương trình nguồn và sau đó chuyển các câu vấn tin SQL này sang câu vấn tin AQL và vẽ cây toán tử dạng AQL. Dưới đây là một vài hình ảnh về chương trình đề mô:

Màn hình chính của chương trình



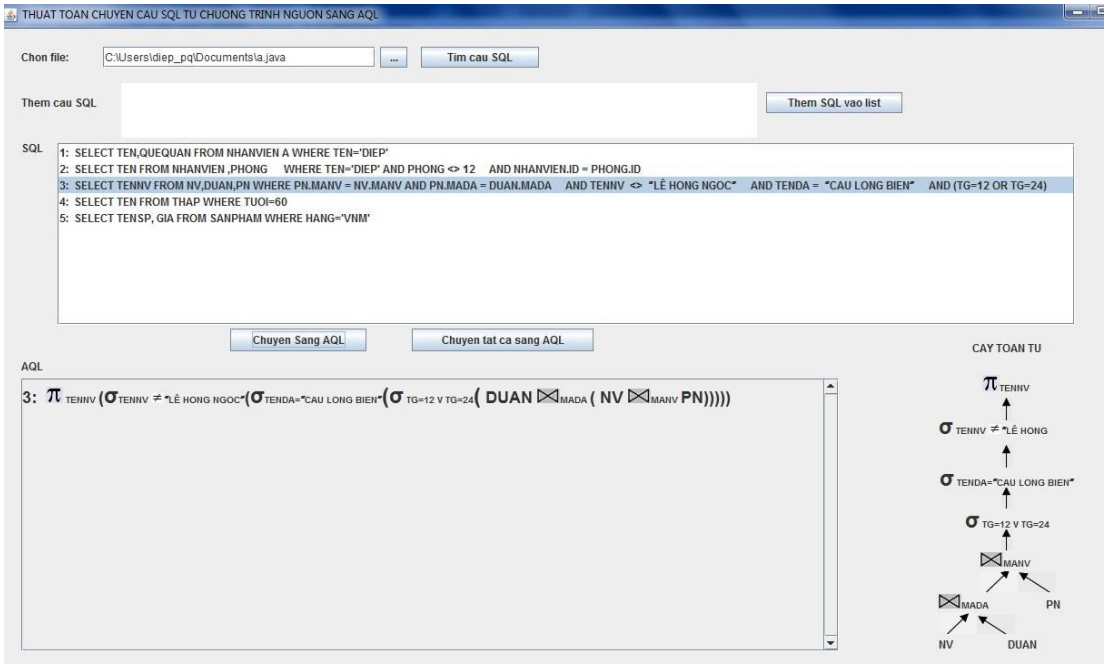
Hình 1.6. Màn hình chính của chương trình đề mô

Sau khi chọn file nguồn (giả sử ta chọn file a.java ở thư mục C:/Users/diep_pq/Documents/) và tìm các câu truy vấn SQL trong file đã chọn ta click vào nút 'Tim cau SQL'. Kết quả như sau



Hình 1.9. Màn hình kết quả tìm câu truy vấn SQL

Ta có thể chuyển một câu truy vấn SQL thứ k (trong vi dụ k<5) sang AQL và vẽ cây toán tử dạng AQL bằng cách chọn câu truy vấn SQL trong list sau đó click nút 'Chuyen Sang AQL' vi dụ ta chọn câu truy vấn thứ 2 trong list kết quả như sau.



Hình 1.12. Màn hình kết quả sau khi chuyển câu SQL thứ 2 sang AQL

KẾT LUẬN

Trong các tài liệu nghiên cứu và ứng dụng các lệnh SQLs, AQLs, OTs ((Operational Trees) cây toán tử) thường được chỉ ra trước. Từ đó bằng phương pháp tối ưu hóa thủ công, thậm chí có thể tự động tối ưu hóa bằng các thuật toán đã cho tại chương 2 mục 2.3. Nhưng việc tìm kiếm thủ công trong chương trình nguồn, viết lại câu vấn tin SQL, tạo lập cây toán tử sẽ mất không ít thời gian và cũng không tránh được các sai sót. Phương pháp đã được trình bày trong luận văn này nhằm khắc phục các nhược điểm trên, đồng thời hỗ trợ đắc lực cho hướng nghiên cứu và ứng dụng thuật toán song song, lập trình song song xử lí song song và phân tán,…

HƯỚNG PHÁT TRIỂN

Chuyển các thuật toán trong chương 2 sang các thuật toán song song, dùng các ngôn ngữ lập trình song song để thể hiện chúng và thủ tục hóa (mẫu hóa chương trình) các thuật toán trên, nhằm nâng cao hiệu quả nghiên cứu, giảng dạy và sử dụng.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] Đỗ Xuân Lôi (1996), *Cấu trúc dữ liệu và giải thuật,* NXB Khoa học và Kỹ thuật, Hà Nội.

[2] Lê Huy Thập (2008), *Giáo trình Kỹ thuật lập trình*, Tập 1, NXB Khoa học tự nhiên và công nghệ, Hà Nội.

[3] Lê Huy Thập (2007), *Phân mảnh trên giá trị lặp của các thuộc tính trong CSDL quan hệ*, Tạp chí tin học và điều khiển tối ưu, Tập 23 số 1 (2007) 86 - 98.

[4] Lê Huy Thập (2011), *Giải các bài toán trên cây toán tử đường ống bằng ma trận đặc trưng,*Tạp chí tin học và điều khiển học, Tập 27 số 2 (6/2011) 107-118.

[5] Lê Huy Thập, *Cơ sở lý thuyết song song*, NXB Thông tin và Truyền thông, 12-2011, Tái bản lần 1

[6] Lê Huy Thập, *Bài giảng về CSDL quan hệ,* Dùng giảng dạy tại học viện công nghệ bưu chính viễn thông, 2010

[7] Lê Huy Thập, *Bài giảng về CSDL phân tán,* Dùng giảng dạy tại học viện công nghệ bưu chính viễn thông và Đại học sư phạm Hà Nội 2, 2007

[8] Lê Huy Thập, *Phân mảnh trên giá trị lặp của các thuộc tính trong CSDL quan hệ,* Tạp chí tin học và điều khiển tối ưu, Tập 23, Số 1, 86 – 98, 2007.

[9] Lê Huy Thập, *Giải các bài toán trên cây toán tử đường ống bằng ma trận đặc trưng,* Tạp chí tin học và điều khiển học, tập 27 số 2 trang 107-118, tháng 6 năm 2011.

[10] Lê Huy Thập, *Nghiên cứu xác định các dạng báo cáo tổng hợp linh hoạt*, Tạp chí tin học và điều khiển học, tập 25 số 2 trang 188-200, 2009.

[11] Lê Huy Thập, *Bảng câu vấn tin trên các quan hệ và xử lý câu vấn tin trên bảng* , Kỷ yếu hội thảo Quốc gia lần 13, Hưng yên năm 2010, trang 29-40, NXB Khoa học và kỹ thuật, 2011.

[12] Lê Huy Thập, *Báo cáo động và các mệnh đề lọc dữ liệu*, Kỷ yếu hội thảo Quốc gia Biên Hòa, năm 2009, trang 19-36, NXB Khoa học và kỹ thuật, 2010.

[13] Lê Huy Thập, *Loại bỏ các mẩu tin nhân bản thừa trong cơ sở dữ liệu quan hệ*, Kỷ yếu hội thảo Quốc gia Đại Lải, năm 2007, trang 219-227, NXB Khoa học và kỹ thuật, 2008.

[14] Lê Huy Thập, *Chuẩn hoá và xác định mối quan hệ giữa các cụm từ, tìm các thông tin liên quan đến cụm từ*, Kỷ yếu hội thảo Quốc gia Huế, năm 2007, trang 57-58, NXB Khoa học và kỹ thuật, 2008.

[15] Lê Huy Thập, *Tìm thông tin trên các máy tính bằng cách dùng các chuỗi để so sánh*, Kỷ yếu hội nghị KH kỷ niệm 30 năm thành lập Viện CNTT, trang 422-427, NXB Khoa học tự nhiên và Công nghệ

[16] Hưng Yên (2011), *Bảng câu vấn tin trên các quan hệ và xử lý câu vấn tin trên bảng,* Kỷ yếu Hội thảo Quốc gia lần thứ XIII, Hưng Yên, 19-20 tháng 8 năm 2010, "Một số vấn đề chọn lọc của CNTT và truyến thông",Chủ đề: "Các công nghệ tính toán hiện đại", Báo cáo toàn văn, NXB Khoa học và kỹ thuật, 2011, 29-40.

[17] Seyed H. Roo (1999), Parallel processing and Parallel Algorithms, Theory and Coputation.

[18] Robert Sedgewick (2001), *Cẩm nang thuật toán Vol.1 and vol.2*, NXB Khoa học và Kỹ thuật.

[19] Behrooz Parhami (1999), Introduction to Parallel Processing: Algorithms and Architectures, Springer.

[20] Michael J. Quinn. (2000), Parallel Computing: theory and practice, 2nd edition. Oregon State University, USA. McGraw Hill Inc.

[21] Dimitri P. Bertsekas and John N. Tsitsiklis.(2001), Parallel and Distributed Computation: Numerical Methods. Massachusets Institute of Technology. Prentice Hall Press.

[22] Hesham El-Rewini, Mostafa Abd-El-Barr, Advanced Computer Architecture and Parallel Processing, Wiley, 2005.

[23] Charles Leiserson. The lecture notes on Theory of Parallel Systems. Massachusets Institute of Technology, Open Course Ware. www.ocw.mit.edu

[24] Michel Cosnard & Denis Trystram, Parallel Algorithms and Architectures. International Thomson Computer Press. 1995.

MỤC LỤC