VỀ VIỆC THIẾT KẾ MỘT TRÌNH BIÊN DỊCH NGÔN NGỮ C

Nguyễn Chí Công
Nguyễn Việt Hải
Doãn Ngọc Liên

Trong những năm gần đây, với sự phát triển mạnh mẽ của kỹ thuật vi xử lí, giá thành phần cứng của các máy vi tính có khả năng tính toán lớn đã giảm xuống một cách đáng kể, máy vi tính 16/32 bit đã trở nên thông dụng. Trên các máy vi tính đó hệ điều hành UNIX đã chiếm giữ một trong những vị trí hàng đầu và đang trở thành chuẩn hóa. Ngôn ngữ chìa khóa của hệ điều hành đã nhiệm và chạy theo chế độ nhiều người sử dụng đó là C - một ngôn ngữ lập trình vạn năng đơn giản, được thừa nhận là công cụ hiệu quả cho việc sản xuất phần mềm hệ thống, cũng như phần mềm ứng dụng. (Bản thân hệ điều hành UNIX được viết bằng ngôn ngữ C, chỉ trừ một phần nhỏ không quá 10% bắt buộc phải viết trong hợp ngữ máy cài đặt).

I. Qui trình cài đặt

Qui trình cài đặt được chọn lựa trên cơ sở các phương tiện mà chúng tôi có được tại Viện KHTT và ĐK gồm có:

1. Các máy vi tính IBM-PC, XT, AT với trình biên dịch C Microsoft cho INTEL 8088/80286.
2. Hệ phát triển FT68000 với hệ điều hành OS-9, trong đó chưa có trình biên dịch ngôn ngữ bậc cao, chỉ có trình biên dịch hợp ngữ (assembler) và một thư viện chương trình mẫu cơ sở khá đầy đủ.

Mặt khác phải nhắc tới công việc ghép nối để chuyển các tập dữ liệu của hệ điều hành PC DOS trên các máy vi tính của IBM sang hệ phát triển FT 68000 (tức tập của OS-9) đã được thực hiện đầu tiên. Qua nghiên cứu và thử nghiệm chúng tôi lựa chọn qui trình cài đặt gồm 3 bước như sau:

(a) Bằng ngôn ngữ C chuẩn (chứa trong C Microsoft) viết trình biên dịch C biên dịch chương trình từ C gốc sang mã máy MC 68000.
(b) Sử dụng trình biên dịch C Microsoft biên dịch C' và cho C' tự dịch nó trên IBM-PC. Kết quả của việc tự dịch này là trình biên dịch C'' trong dạng mã máy MC 68000.
(c) Đưa kết quả thu được trong bước (b) vào hệ phát triển FT 68000 và thực hiện việc ghép nối với hệ điều hành OS-9, ta thu được trình biên dịch mong muốn.

Theo thiết kế, các chương trình con phục vụ cho C đều được viết bằng ngôn ngữ C và biên dịch trên IBM-PC nhằm giảm đến mức tối thiểu việc ghép nối ban đầu với OS-9.

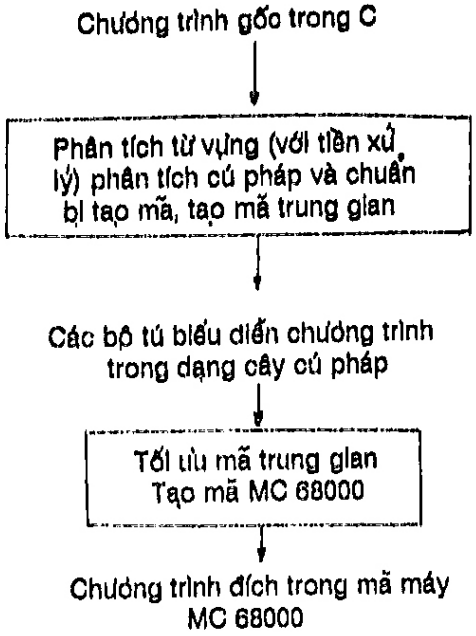
II. Phương pháp biên dịch

Quá trình biên dịch chương trình gốc từ C sang mã máy MC 68000 đi qua 2 bước (như vậy có khả năng thêm bước thứ 3 nếu như người sử dụng thấy cần thiết cho công việc tối ưu mã). Bước thứ nhất tiến hành các công việc sau:
(a) Phân tích từ vựng (lexical analysis) tiến hành đồng thời với quá trình tiền xử lí (preprocessing);
(b) Phân tích cú pháp (syntax analysis) và chuẩn bị tạo mã ;
(c) Tạo mã trung gian (intermediate code generation).

Bước hai thực hiện việc tối ưu mã trung gian (code optimization và tạo mã (code generation).

Biên dịch các cấu trúc chương trình thực hiện theo phương pháp biên dịch điều khiển bởi cú pháp (syntax-directed compilation) với kỹ thuật triển khai đệ quy từ trên xuống (recursive descent), riêng biểu thức sử dụng kỹ thuật phân tích đơn giản nhưng hiệu quả đó là kỹ thuật toán tử trước (operator precedence).

Như vậy trình biên dịch sẽ có cấu trúc theo sơ đồ sau:



Quá trình tiền xử lí và phân tích từ vựng thực hiện bằng một chương trình con với kết quả là mã nguyên (integer code) của các ký hiệu trong ngôn ngữ gốc; chẳng hạn mã nguyên của các toán tử là một số từ 1 đến 69, các từ khóa có mã từ 70 đến 108, tên các đối tượng có mã là 109, các hằng có mã từ 110 đến 115, v.v...

Dạng mã trung gian chứa đựng đầy đủ thông tin về ngữ nghĩa của chương trình. Dạng sử dụng là các bộ tứ (quadruples) được tạo ra theo cấu trúc cây cú pháp của chương trình.

Thí dụ 1: Dãy các bộ tứ biểu diễn biểu thức A*B + C*D gồm có:

(1) (*, A,B, T1)
(2) (*, C,D, T2)
(3) (+, T1, T2, T3)

Bộ đầu tiên biểu diễn tính toán T1 := A*B, bộ thứ hai: T2 := C*D và bộ thứ ba: T3 := T1 + T2, trong đó A, B, C, D là các biến phụ lưu giữ các giá trị trung gian. Các biến phụ này thường được lưu giữ trong các thanh ghi, chỉ khi không còn giải pháp khác thì mới được cấp phát bộ nhớ.

Thí dụ 2 : Dãy các bộ tứ biểu diễn câu lệnh:
If (<biểu thức>) <câu lệnh 1>
else <câu lệnh 2>

gồm có :

(1) dãy bộ tứ tính toán <biểu thức>, kết quả ghi vào biến phụ T.
(p) (BRZ, q, , T)
(p+1) dãy bộ tứ thực hiện <câu lệnh 1>
(q-1)(BRA, r , ,)
(q) dãy bộ tứ thực hiện <câu lệnh 2>
(r)

trong đó (BRZ, q, , T) có ngữ nghĩa là nếu T = 0 thì nhảy đến thực hiện bộ tứ (q), còn không thực hiện bộ tứ tiếp theo; (BRA, r, ,) là bộ tứ tương đương lệnh nhảy không điều kiện đến bộ tứ (r).

Việc tạo mã MC 68000 từ các bộ tứ là một bước không mấy khó khăn, tuy vậy sự phức tạp sẽ nảy sinh khi chúng ta tiến hành một số công việc tối ưu mã. Mục tiêu hiện nay của chúng tôi về tối ưu mã gồm có:

(1) Sử dụng triệt để và hiệu quả các thanh ghi và hệ lệnh của MC 68000

(2) Chọn giải pháp tối ưu cho vấn đề lưu trữ các giá trị trung gian.

(3) Phát hiện và đưa ra ngoài các vòng lặp các bất biến (In variant) lặp (điều này có ý nghĩa quan trọng vì rằng theo thống kê thì thời gian điều khiển nằm trong các vòng lặp lên tới 75% thời gian thực hiện chương trình.

III. Tổ chức chương trình và dữ liệu

Đơn vị cơ sở (basic unit) của chương trình trong ngôn ngữ C là hàm (function), chúng giống như thủ tục hay chương trình con trong các ngôn ngữ lập trình bậc cao thông dụng khác. Chương trình trong C là một danh sách mô tả tuyến tính các đối tượng tổng quan gồm có mô tả dữ liệu và mô tả các hàm (chúng có thể xuất hiện xen kẽ nhau). Hàm trong C không được lồng nhau, nhưng được phép đệ quy và gọi nhau tùy ý. Trao đổi dữ liệu giữa các hàm thực hiện qua các biến tổng quan hoặc qua tham biến.

Ngôn ngữ C chỉ cho phép một hình thức chuyển tham biến duy nhất là chuyển qua giá trị (call by value) nhưng được bổ sung một cách phong phú nhờ cơ chế năng động của kiểu trỏ (pointer). Trong hàm có thể mô tả các đối tượng lân cận trong các khối (block). Khối trong C không giữ nguyên nghĩa trọn vẹn của khối trong các ngôn ngữ tựa Algol, mà chỉ là một câu lệnh ghép (compound statement) thuần túy, trong đó có mô tả dữ liệu. Do các câu lệnh ghép có thể lồng nhau, nên các dữ liệu lân cận mô tả có thể lồng nhau.

Về kiểu dữ liệu, ngôn ngữ C có đầy đủ các kiểu cơ sở truyền thống như kiểu ký tự (char), nguyên (integer), thực (float), nguyên ngắn (short), nguyên dài (long), nguyên không dấu (unsigned) và số với độ chính xác đôi (double). Đặc biệt kiểu trỏ (pointer) trong C là một kiểu đơn giản nhưng có vai trò rất năng động. Từ các kiểu cơ sơ cơ bản đó với các phương pháp cấu trúc hóa chúng ta xây dựng các kiểu dữ liệu phức tạp có cấu trúc. Trong C có 4 phương pháp cấu trúc hóa dữ liệu:

(1) mảng (away)
(2) bản ghi (struture và union)
(3) hàm trở về (function returning)
(4) danh sách (enumeration, tương tự như các kiểu vô hướng định nghĩa trong ngôn ngữ Pascal).

Như vậy ngôn ngữ chứa đựng vô hạn các lớp các kiểu dữ liệu khác nhau. Sự đệ quy của cấu trúc dữ liệu hoặc là trực tiếp (như mảng của bảng, bản ghi của bản ghi) hoặc là thông qua trỏ. Vì vậy ngôn ngữ rất phong phú về các hình thức mô tả dữ liệu.

Một đặc điểm quan trọng là chương trình cho phép biên dịch riêng lẻ (separate compilation), nghĩa là một chương trình gốc trong C có thể được ghi từng phần trên một hay nhiều tệp khác nhau và biên dịch riêng biệt, sau đó mới liên kết lại trước khi thực hiện. Vì thế quá trình liên kết không chỉ theo nghĩa truyền thống mà nảy sinh những liên kết ở mức ngôn ngữ.

Trên đây chúng ta đã xét qua những đặc điểm cơ bản nhất của ngôn ngữ, những đặc điểm đó quyết định rất nhiều đến sự biểu diễn bên trong của chương trình và dữ liệu.

Các biến tổng quan (global) được biểu diễn trong một vùng cố định gọi là miền dữ liệu tổng quan (global data segment). Miền này được xác định từng phần qua các tệp gốc biên dịch của chương trình. Mỗi biến trong vùng này (kể cả thành phần của bản ghi) biểu diễn qua cơ sở (base) và offset.

Phần chương trình bao gồm tất cả các hàm mô tả trong chương trình, mỗi hàm biểu diễn bằng hai miền riêng biệt gọi là miền dữ liệu (data segment) và miền chương trình (program segment). Miền dữ liệu chứa các biến lân cận trong hàm, miền này được cấp phát cho từng lần được gọi thực hiện hàm và chỉ tồn tại cho đến khi điều khiển ra khỏi hàm. Trái lại, miền chương trình chứa dãy mã lệnh thực hiện hàm là một miền cố định, mỗi lời gọi thực hiện hàm đều tham chiếu (reference) đến vùng duy nhất này. Ngôn ngữ cho phép hàm đệ quy, nên việc cấp phát các miền dữ liệu cho các hàm được thực hiện theo cơ chế stack và trong trường hợp của chúng ta sử dụng công cụ lý tưởng sẵn có của MC 68000 là các lệnh LINK và UNLINK (xem [7]), stack này trùng với stack thực hiện gọi hàm (tức stack lưu giữ các địa chỉ trở về và các tham biến thực sự).

Các biến lân cận trong hàm (và các tham biến) xác định qua địa chỉ tương đối (relative address) so với một giá trị xác định cụ thể cho từng lần gọi thực hiện hàm. Giá trị này cũng giống như "giá trị trở về" của hàm được lưu giữ trong những thanh ghi dành sẵn.

Miền chương trình cho mỗi hàm gồm có 4 phần:

(1) mã lệnh khởi đầu (entry code)
(2) mã lệnh tính toán hàm
(3) mã lệnh kết thúc hàm (exit code)
(4) các giá trị hằng, nếu có

Mã lệnh khởi đầu và kết thúc hàm thực hiện nhiệm vụ chủ yếu là lưu trữ và khôi phục trạng thái các thanh ghi, đặc biệt là các thanh ghi tham gia cơ chế thực hiện hàm, đồng thời thực hiện việc đặt các giá trị cho các thanh ghi biểu diễn cơ sở (base) cho các biến (như vậy gồm cả việc cấp phát và giải phóng vùng nhớ các biến lân cận).

Phần mã lệnh tính toán hàm là dãy mã lệnh tương đương của thân hàm (function body) trong mã máy MC 68000.

IV. Trình biên dịch

Trình biên dịch được viết bằng chính ngôn ngữ C - ngôn ngữ bậc cao, chương trình con được phép đệ quy, vì vậy việc mã hóa (coding) tương đối thuận tiện. Hơn nữa chúng tôi phỏng theo bản gốc trình biên dịch ngôn ngữ Pascal của N.Wirth [8], nên trình biên dịch của chúng tôi có khuôn mẫu truyền thống với những kỹ thuật cài đặt đã được thừa nhận là chính quy và có hiệu quả.

Trình biên dịch gồm có các chương trình con cơ bản sau đây:

-Ext-data-def ( ): Xử lý các định nghĩa dữ liệu tổng quan.

-Function - def( ): Xử lý các định nghĩa hàm.

-Comp-stat ( ): Xử lý câu lệnh ghép, kể cả phần thân hàm.

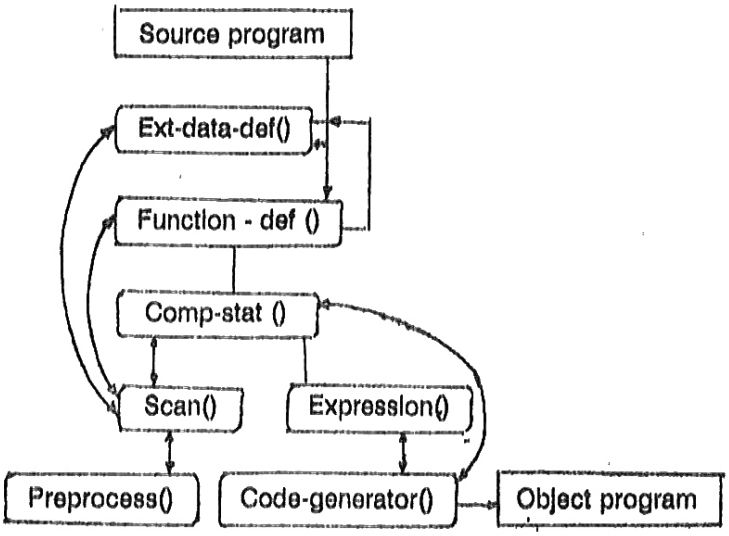
-expression ( ): Xử lý biểu thức.

-Scan ( ): phân tích ký hiệu.

-preprocess ( ): xử lý các câu lệnh của tiền xử lý.

-code-generator ( ): hệ chương trình tạo mã.

Quan hệ giữa các chương trình con này được biểu diễn qua sơ đồ sau:

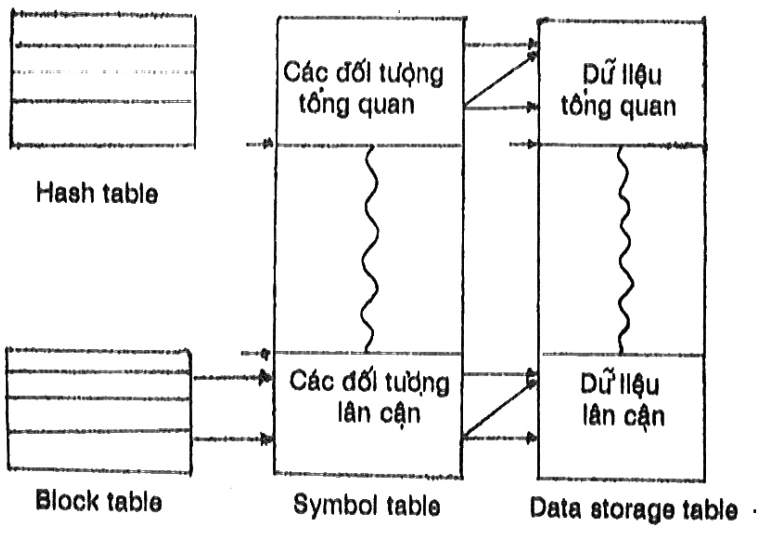


Đối với trình biên dịch, bảng ký hiệu (symbol table) giữ vai trò quan trọng. Trong bảng ký hiệu chứa đựng đầy đủ tên của các đối tượng tồn tại trong chương trình như nhãn, biến, hàm, v.v...cùng với các thuộc tính (attribute) của chúng. Do vì các đối tượng thuộc các lớp khác nhau mang các đặc tính riêng biệt, nên phần tử của bảng ký hiệu là một bản ghi có thành phần thay đổi. Mặt khác, để đạt được hiệu quả cao trong việc quản lý bảng ký hiệu, các thông tin cụ thể về đặc tính được lưu giữ trong một bảng thứ hai gọi là bảng dữ liệu (data storage table) và một bảng các khối (block table) nhằm thuận tiện cho việc quản lý miền tồn tại của các đối tượng.

Việc thâm nhập đến bảng ký hiệu thông qua cơ chế hashing. Hàm hash được chọn để thực hiện ánh xạ tên các đối tượng sang một điểm vào trong bảng băm nhỏ (hash table) từ đó xác định một danh sách trong đó xuất hiện đối tượng cần truy nhập. Như vậy hai đối tượng cùng tên được phân biệt qua lớp của chúng, hoặc theo quy tắc "trong nhất có hiệu lực". Điều này dễ dàng thực hiện vì rằng các tên trong nhất bao giờ cũng đứng đầu tiên trong danh sách.

Bảng ký hiệu được tổ chức và quản lý như hai stack có đỉnh chụm vào nhau, một stack lưu giữ các đối tượng tổng quan, còn một stack lưu giữ các đối tượng lân cận (cho trường hợp biên dịch đang thực hiện đối với hàm).

Biên dịch thực hiện cho từng hàm trong tệp gốc biên dịch, vì vậy không cần thiết phải lưu giữ các thông tin về các đối tượng lân cận khi kết thúc biến dịch hàm.
Sơ đồ tổ chức bảng ký hiệu như sau:



V. Ghép nối với hệ điều hành OS-9

Khi ghép nối với hệ điều hành OS-9 chúng ta phân biệt hai công việc:
(1) Ghép nối hệ thống (tức trình biên dịch) với OS-9
(2) Ghép nối C cài đặt với OS-9.

Thật vậy, nội dung của phần việc thứ nhất là những ghép nối để trình biên dịch là một phần mềm của OS-9. Trước hết đó là việc xử lý tệp kết quả (tức) trên IBM-PC để trở thành một tệp thực hiện được của OS-9, tiếp theo là việc thích ứng các tệp đích đó sinh ra trên MC 68000 tương thích với OS-9 và sau cùng là sự dụng các chương trình con dựng sẵn (build-in subroutine) trong OS-9 để giảm bớt độ lớn và nâng cao hiệu quả trình biên dịch.

Phần việc thứ hai liên quan đến vấn đề thiết kế cơ chế ghép nối các lời gọi từ một hàm trong ngôn ngữ cài đặt đến thư viện chương trình mẫu sẵn có của OS-9, chẳng hạn như các thao tác vào/ra, các hàm số học cơ bản (SIN, COS, EXD,...) và đặc biệt là các lời gọi thao tác tổ chức và khai thác tệp. Phần việc này thực hiện theo chuẩn UNIX [1].

VI. Kết luận

Lý thuyết chương trình dịch đã trở thành kinh điển và đã được viết thành những chương trọn vẹn của ngành Tin học, tuy vậy việc thực hiện cài đặt trình biên dịch một ngôn ngữ bậc cao kiểu như C đối với chúng ta vẫn đang là một bài tập lớn có giá trị. Chúng tôi mạnh dạn thực hiện công việc này với hy vọng sẽ gặt hái được những kinh nghiệm quý báu trong lĩnh vực chương trình dịch và chương trình hệ thống.

TÀI LIỆU THAM KHẢO

1. D. M. Ritchie, "The C programming Language", Prentice-Hall, 1978.
2. A. V. Aho, J. D. Ullman, "Principles of Compilerdesign", Prentice-Hall, 1981.
3. D. Gries, "Compiler contruction for Digital Computers"
4. Microsoft C Compiler 1981, 1982, 1983.
5. N. Wirth, "The design of a Pascal Compiler"
6. B. W. Pollack, "Compiler Techniques"
7. P. Jaulent, "Le microprocesseur 68000 et sa programmation"
8. Pascal Source compiler.

ABSTRACT

ON DESIGNTING A COMPILER FOR MC68000

Based on their experiences of writing a Pascal compiler and of interfacing OS9/68K (an UNIX - like Operating System) with MS-DOS the authors pointed out a method of designing and implementing a C compuletor MC68000)as the following:

1) In Microsoft C under MS-DOS creat a program forcrosscompiling the C sources to obtain their objects in MC68000 code.

2)Under MC-DOS use the executable program for computing his source by himself to get a program in MC68000 code.

3) Under OS9/68K make the wanted compuler executable (from the object code) and compatible with OS9 file format.