

VỀ TƯƠNG ĐƯƠNG GIỮA CÁC CHƯƠNG TRÌNH NGUỒN VÀ VẤN ĐỀ KHÔI PHỤC DẠNG NGUỒN

NGUYỄN NGỌC KIẾN

Trong nghiên cứu, phát triển phần mềm, một khó khăn đặt ra là các hệ điều hành cơ sở thường được bán như những «hộp đen». Người mua máy tính không có hoặc có rất ít thông tin về hệ mềm, nhất là văn bản các chương trình hệ thống (dạng nguồn), một điều kiện rất quyết định để có thể đọc lập tiếp cận và phát triển không lệ thuộc vào hãng sản xuất. Đối với một số trường hợp, chẳng hạn như các hệ điều hành thời gian thực và thời gian thực nhiều mục đích, nhu cầu phát triển phần mềm cơ sở là rất cao. [1,3]. Các bộ deassembler ở một số máy tính thực chất là chuyển đổi ngược các giá trị nhị phân thành các dòng lệnh ký tự, tức là chỉ có tác dụng đối với chương trình ở dạng «thực hiện», chỉ cho phép quan sát một đoạn bộ nhớ ở dạng tuyệt đối. Trong khi đó một hệ mềm cơ sở thường gồm nhiều đơn thể (modul) ở các dạng biến đổi trung gian giữa dạng nguồn và dạng «thực hiện», chứa các thông tin quý đối với người làm phần mềm. Song qua biến đổi về dạng «thực hiện» các thông tin trung gian mất dần, không thể tạo lại bằng deassembler được.

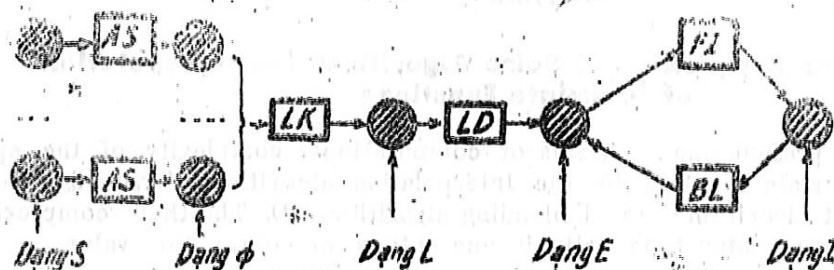
Khất phát từ những yêu cầu thực tiễn, ở đây xét việc xây dựng lại văn bản, tái tạo một số thông tin bị mất đối với một vài dạng biến đổi trung gian của hợp ngữ, định hướng cho một quá trình khôi phục dạng nguồn của một hệ cụ thể—hệ MAS máy tính MINI16.

1. Về tương đương giữa các chương trình nguồn

Một hệ mềm cơ sở gồm một số đơn thể ở những dạng khác nhau và có các bộ biến đổi chuyển các đơn thể ở dạng này sang dạng khác, ở đó dạng nguồn là dạng đầu tiên và dạng «thực hiện» là dạng được đưa vào bộ nhớ và sẵn sàng thực hiện. Mỗi đơn thể đều hoặc tham gia hình thành các đơn thể ở dạng khác hoặc là được thực hiện ở bộ nhớ [1, 2].

Giả sử ta có a là một đơn thể dạng A . Đạt được từ nguồn s qua các biến đổi $P_1, P_2, v.v...$ Vấn đề đặt ra là chỉ có a và cần phải xây dựng lại s . Song qua mỗi biến đổi P_i các thông tin phục vụ cho P_i đã được sử dụng và không cần thiết phải giữ lại. Với P_i các đơn thể vào có thể khác nhau mà vẫn có cùng đơn thể ra. Vì vậy về nguyên tắc không thể có lại s , mà cần có một tiêu chuẩn về «tương đương» để nếu đạt được một s' bằng cách nào đó, ta có thể coi s' là thay thế được. Một tiêu chuẩn tự nhiên là cho s' qua chuỗi biến đổi tương ứng để thu được a' (ở dạng A) và so sánh a' với a . Ta gọi s và s' là tương đương theo A nếu a và a' trùng nhau. Như vậy nếu với mỗi đơn thể của hệ thống (ở một dạng B nào đó) ta có được một dạng nguồn tương đương với nguồn của hãng (theo dạng B đó) thì tập các văn bản đó là phản ánh được hiện trạng của hệ thống mà người mua máy đang có, ít nhất là có thể tái tạo lại nó. Từ tiêu chuẩn này, việc so sánh s' với cái không có là s được quy về so sánh a' với cái hiện có là a . Đây là tiêu chuẩn tối thiểu cần đảm bảo.

Trong [2] có đưa ra một số dạng trung gian khá phổ biến khi các đơn thể nguồn hợp ngữ đi qua các bộ biến đổi. Ta xét một số dạng gặp trong nghiên cứu cụ thể (hình 1) đối với hệ MAS:



Hình 1

Các đơn thể nguồn S (source) khi qua bộ dịch hợp ngữ AS cho dạng đích (object) qua bộ liên kết LK cho dạng nạp L (load), qua bộ nạp LD cho dạng thực hiện E (execute) và một dạng thường có là dạng ảnh I (image)—các ảnh tức thời của dạng thực hiện sinh ra bởi một bộ tạo ảnh IG (image generator) và được nạp trở lại bộ nhớ bằng một bộ nạp nhị phân BL (binary loader). Rõ ràng một deassembler chỉ cho phép quan sát dạng nghèo nàn nhất là dạng E.

2. Vấn đề khôi phục dạng nguồn.

Với các dạng nêu trên chỉ có dạng E là ở bộ nhớ chính, còn các đơn thể dạng khác nằm ở bộ nhớ ngoài. Đối với hệ một cơ sở bán cho người dùng thì dạng S là không có hoặc rất ít, và yêu cầu đặt ra là phải có nguồn từ các dạng không phải nguồn. Ở đây ta xem xét một vài điểm chung nhằm định hướng cho việc thực hiện cụ thể ở phần sau cho hệ MAS.

Áp dụng tiêu chuẩn đề ra về tương đương, từ các đơn thể dạng O, L và I cần phải đạt các đơn thể nguồn đảm bảo tương đương theo những dạng đó. Việc kiểm tra tương đương được thể hiện qua so sánh các đơn thể theo dạng tương ứng.

Cần phải tạo ra các quy trình thực hiện khôi phục nguồn cho từng dạng. Chú ý đến tính chất chuyển đổi lẫn nhau giữa các dạng, ta có thể xây dựng quy trình này kế tiếp từ các quy trình trước. Lấy xuất phát là một deassembler d_0 để dịch các mã lệnh tuyệt đối, có thể bổ sung chức năng chuyển đổi các tin báo và dữ liệu theo yêu cầu, qua đó hình thành quy trình D_0 gồm nhiều bước. Đối với dạng chứa nhiều thông tin trung gian hơn cả là dạng ϕ , thường được tổ chức từ các khối mã lệnh và các khối thông tin phục vụ liên kết có thể xây dựng $D_1 = D_0, d_1, d_1^2, \dots$. Ở đây dùng D_0 xử lý khối mã lệnh trước để tạo ra một nguồn tạm thời, sau đó dùng các d_1^i để xử lý dần các thông tin thuộc một lớp i nào đó. Đối với dạng L và I, nếu ta xây dựng được các quy trình con d_2, d_3 để chuyển dạng từ L về O và từ I về L thì có thể tổ chức các quy trình lớn $D_2 = d_2, D_1$ và $D_3 = d_3, D_2$. Như vậy ta có thể nối tiếp các quy trình với nhau, song việc xây dựng d_2, d_3 cũng như d_1 là tùy thuộc cấu trúc cụ thể của mỗi dạng. Chú ý là d_2, d_3 chỉ là chuyển cách thể hiện chứ không phải là khôi phục hoàn toàn dạng đứng trước. Hình 2 là minh họa cho trường hợp D_2 và D_3 .

3. Khôi phục dạng nguồn cho hệ MAS.

Hệ MAS (Multi-Application-System) là một hệ điều hành nhiều mục đích của máy tính MINI16 (real-time, 256 Kbytes bộ nhớ trong, đĩa từ, 1,1 triệu phép tính/giây) có nhiều nhu cầu về phát triển hệ mềm cơ sở [5]. Trong khi đó, do những nguyên nhân về kinh tế—kỹ thuật, hãng giữ kín dạng nguồn để tránh buộc người dùng đối với những yêu cầu về bổ sung vào phần mềm hệ thống. Từ ý nghĩa kinh tế đó, việc khôi phục lại dạng nguồn đối với MAS là thiết thực.

Qua những khảo sát chung đã đưa ra những quy tắc định hướng như đã trình bày ở phần 1 và 2. Song chủ yếu công việc thể hiện trong thăm dò khảo sát chi tiết các loại cấu trúc tin, xây dựng các bộ biến đổi, thiết lập các thủ tục tự động và bán tự động và tiến hành khôi phục nguồn cho từng đơn thể, mỗi đơn thể được xử lý nhiều lần, đọc văn bản, hiệu chỉnh và biến đổi xuôi để so sánh với dạng hiện có. Dạng đích ϕ là dạng của các thư viện phục vụ sinh hệ (để tạo ra Supervisor theo tính chất thiết kế mở của hệ nhiều mục đích [3,4], thư viện cơ sở của ngôn ngữ bậc cao (Fortran) và thư viện các chức năng thời gian thực. Đây là dạng mới chỉ qua một khâu dịch, còn chứa nhiều thông tin quý giá—các thông tin phục vụ liên kết (tên gọi thực của các điểm vào, các điểm tham khảo ngoài, các địa chỉ tương đối, v.v...). Dạng L với cấu trúc overlay là cho các chương trình phục vụ (utility) lớn. Dạng I là cho các phân đoạn ảnh tức thời của Supervisor được «Swapping» giữa đĩa từ và bộ nhớ trong khi làm việc.

Việc đầu tiên được tiến hành là xây dựng một bộ chuyển đổi ngược deassembler d_0 , chuyển một mã nhị phân ra lệnh ở dạng ký tự, dựa trên cấu trúc bộ lệnh máy của MINI16. Sau đó d_0 được trang bị thêm khả năng tạo các đồng dữ liệu nhị phân và tin báo ASCII, để hình thành quy trình D_0 (thực hiện lặp d_0 nhiều lần kết hợp với hiệu chỉnh). Đối với dạng ϕ , các bộ biến đổi con d_1^i được xây dựng để xử lý các khối tin phục vụ liên kết. Đầu vào là nguồn thu được từ d_1^{i-1} (hoặc D_0) và khối tin loại i , đầu ra là dạng nguồn sẽ phục vụ cho d_1^{i+1} . Bản cuối cùng khi đã ổn định được biến đổi xuôi và so hai bản dạng ϕ với nhau. Bản

thần D₁ cũng là quy trình được sử dụng lặp nhiều lần. D₂ và D₃ được xây dựng dựa trên các quy trình con d₂ và d₃. Một đơn thể L có cấu trúc overlay được xử lý riêng từng nhánh, lấy ra các đoạn trong vị trí tương đối của nhánh đó và chuyển sang dạng ϕ . Tại đây thực hiện khôi phục những thông tin về liên hệ (bị mất khi qua liên kết). Việc khôi phục này là trên nguồn, dùng D₀ để tạo ra nguồn, hiệu chỉnh rồi qua AS để trở lại ϕ . Còn để làm d₃ thì phải tạo ra bộ BL mới hoạt động như BL nhưng chuyển vào một vùng nhớ «user», tại đây lại áp dụng D₀, AS, LK một số lần để chuyển về L. Các thông tin khôi phục ở d₃ là về tính tương đối và tuyệt đối bị mất đi khi qua bộ nạp LD.

Tóm lại, đã xây dựng một số quy trình, trong đó có các bộ biến đổi tự động, đầu vào là các dạng khác nhau. Ở các bước trung gian đều là dạng nguồn. Trên các dạng nguồn trung gian đó thực hiện khôi phục thông tin bằng khảo sát, dùng các công cụ hiệu chỉnh của hệ thống và tự tạo để hiệu chỉnh văn bản. Tại những điểm cần thiết các văn bản này có thể được cho qua các bộ biến đổi của hệ thống AS, LK, LD để cấu trúc lại theo các dạng ϕ , L, ... Vì phải kết hợp một khối lượng lớn công việc bằng khảo sát chủ quan nên các quy trình là bán tự động. Như vậy các văn bản đạt được phần ảnh hưởng nhiều thông tin hơn so với hiện trạng ở các dạng không phải nguồn, nhưng nó đảm bảo tiêu chuẩn tối thiểu qua so sánh là khi cho qua biến đổi xuôi (tức là các thông tin khôi phục mất đi) nó quy tụ về trùng với hiện trạng đó.

4. Khôi phục dạng nguồn: một phương tiện tiếp cận và phát triển hệ MAS

Bằng các quy trình trên đã tiến hành khôi phục nguồn cho toàn bộ hệ MAS với gần 900 đơn thể các loại, trong hơn 200 nghìn dòng lệnh hợp ngữ [6]. Tập các văn bản đó bao gồm các chương trình hợp thành của Supervisor, các loại thư viện, tập hợp các phân đoạn ảnh tức thời hoạt động trong Supervisor, các chương trình phục vụ như chương trình dịch Fortran, hợp ngữ, JCL, liên kết, hiệu chỉnh, quản lý thiết bị và file, truy nhập dữ liệu nhiều chỉ tiêu, sinh hệ, v.v... Trong các khâu biến đổi đều hướng theo tiêu chuẩn tương đương, do vậy việc kiểm định tương đương bằng so sánh không chiếm nhiều thời gian. Trong MAS có 6 đơn thể nguồn có sẵn mô tả các bảng làm việc đã được đem ra so sánh. Một hệ điều hành nhỏ hơn của MINI 16 là D₀S (đơn chương trình) có nguồn của 60 đơn thể Supervisor viết trên cùng hợp ngữ cũng được áp dụng các quy trình này. Kết quả so sánh với bản gốc cho thấy các văn bản khôi phục được chỉ thiếu những dòng bình luận và các lệnh nhảy trong.

Tập các văn bản thu được trước hết đã khắc phục một khó khăn lớn về kinh tế, kỹ thuật, cùng với các quy trình đã xây dựng trở thành một phương tiện sử dụng lâu dài, cho phép tiếp cận vào hệ thống và thực hiện mọi biến đổi, phát triển tùy ý. Bằng phương tiện này đã tiến hành mở xê, học tập, đánh giá hệ MAS, phát hiện và khắc phục những sai sót lớn tồn tại như: khôi phục khả năng dự báo lỗi Fortran, khôi phục hoạt động của chương trình dịch Fortran dùng thiết bị xử lý đầu phẩy động, cơ chế spooling của máy in, chức năng nạp dữ liệu của hệ quản lý tập nhiều chỉ tiêu... Bằng phương tiện này đã tiến hành một số mở rộng để phát triển hệ MAS theo nhiều hướng như: phát triển các chức năng vào-ra cho thiết bị chuyên dụng, mở rộng khả năng tính toán về bộ nhớ, hạch toán phí tồn khai thác, thư viện thời gian thực, ghép nối phát triển hệ điều hành vào hệ phân tán... Trong những năm qua kết quả này đã là tiền đề của hàng chục báo cáo và thông báo khoa học, hàng chục sáng kiến cải tiến KHKT.

Trong một số trao đổi, tác giả đã tiếp thu được nhiều ý kiến đóng góp quý báu cho công việc, nhân dịp này xin chân thành cảm ơn.

Nhận ngày 10-2-1985

TÀI LIỆU THAM KHẢO

1. R.Graham, Principles of System Programming, John Wiley 1975.
2. D.Barron, Assemblers and Loaders. Macdonald: London, 1969.
3. S.Evanczuk, Real-time Operating Systems, Electronics 6-1983.
4. W. Davis, Operating Systems. A Systematic view, Addison-Wesley 1977.
5. Mini 16. Multi-Application-System, 1976.
6. Nguyễn Ngọc Kiên, Khôi phục các chương trình hệ thống dạng nguồn hệ MAS bằng các quy trình dịch ngược. Nội san nghiên cứu KHKT, 1983, 5T-36, Hà Nội.

(Xem tiếp trang 26)

TÀI LIỆU THAM KHẢO

1. Xác định mối quan hệ liên ngành và mô hình tính toán các chỉ tiêu tổng hợp. Viện phân vùng quy hoạch trung ương - Viện Toán học. Hà Nội - 1984.

ABSTRACT

An approach to the problem of investment and labour distribution

Investment and labour are factors important for the development and extended reproduction process of economic sectors.

This paper deals with a model of optimal distribution of investment and labour resources for different production sectors and economic regions. As an example, one optimal variant is received from calculations on micro computer.

Về tương đương giữa các chương trình...

(Tiếp theo trang 14)

РЕЗЮМЕ

Об эквивалентности между исходными программами и проблеме их восстановления.

MAS является многоцелевой операционной системой реального времени на ЗВМ MINI 16. В результате детального изучения MAS определяются одна норма эквивалентности между текстами программ и некоторые принципы видоизменения для системных модулей написанных на ассемблере. Всё это помогает процессу построения последовательностей восстановления символического вида всей системы MAS, ставших сильным средством для освоения и развития системы.
