ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

NGUYỄN VĂN ĐỒNG

XÂY DỰNG HỆ THỐNG ĐẠI SỐ MÁY TÍNH XỬ LÝ BIỂU THỨC TOÁN HỌC

Ngành:	Công nghệ thông tin
Chuyên ngành:	Kỹ thuật phần mềm
Mã số:	60480103

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

NGƯỜI HƯỚNG DẪN KHOA HỌC: PGS.TS.TRƯƠNG ANH HOÀNG

Hà nội- 2016

Mở đầu

Ngày nay các nhà khoa học mô hình hóa các hiện tượng tự nhiên bằng cách dịch các kết quả thực nghiệm và khái niệm lý thuyết vào những biểu thức toán học chứa số, biến, hàm số và các toán tử. Sau đó dựa vào các định lý đã được chứng minh để biến đổi hoặc chuyển thành các biểu thức khác để khám phá các hiện tượng đang được nghiên cứu. Cách tiếp cận toán học như vậy là một thành phần quan trọng của phương pháp nghiên cứu khoa học trong các ngành khoa học hiện nay.

Trong hơn nửa thế kỉ qua máy tính đã trở thành thiết bị không thể thiếu giúp giải quyết các vấn đề toán học. Các nhà toán học thường xuyên sử dụng máy tính để tìm lời giải cho các vấn đề khó khăn hoặc những vấn đề không thể thực hiện được bằng phương pháp thủ công. Trên thực tế máy tính chỉ thao tác với hai kí hiệu 0 - 1 thông qua các luật được thiết lập sẵn nên không thể mong đợi nó tạo ra tiên đề, lý thuyết.... Tuy nhiên một phần của lý luận toán học như các thao tác máy móc, phân tích biểu thức… thì có thể thực hiện bằng các thuật toán. Hiện nay có các chương trình máy tính có khả năng rút gọn biểu thức, tích hợp các chức năng phức tạp, giải chính xác phương trình… Các lĩnh vực toán học và khoa học máy tính có liên quan đến vấn đề này thì được gọi là đại số máy tính.

Đại số máy là tính là một lĩnh vực khoa học đề cập tới việc nghiên cứu và phát triển các thuật toán và phần mềm ứng dụng trong tính toán các biểu thức toán học và các đối tượng toán học khác. Trong đó hệ thống đại số máy tính là một phần của đại số máy tính, một chương trình phần mềm cho phép tính toán các biểu thức toán học bằng cách tương tự như tính toán bằng phương pháp thủ công mà các nhà toán học và khoa học thường sử dụng.

Hệ thống đại số máy tính là gì?

Hệ thống đại số máy tính là chương trình phần mềm thực hiện biến đổi các biểu thức toán học trong đó các yếu tố toán học như rút gọn, giai thừa, lũy thừa… được kết hợp với các cấu trúc điều khiển như vòng lặp, cấu trúc rẽ nhánh và các chương trình con để tạo ra các chương trình có thể giải quyết các vấn đề toán học.[**23**]

Hệ thống đại số máy tính đặc biệt hữu ích cho các nhà toán học, khoa học vì chúng có nhiều chức năng như tính toán biểu thức, xử lý biểu tượng (symbolic manipulation), giải phương trình…

Tại sao lại cần một hệ thống đại số máy tính?

- Trên thực tế có những bài toán hoặc vấn đề không thể giải quyết được bằng phương pháp thủ công.
- Các đáp án đưa ra bằng phương pháp đại số thường ngắn gọn và cung cấp thông tin về mối liên hệ giữa các biến.

- Từ biểu thức đại số có thể suy ra các thay đổi của tham số có thể ảnh hưởng đến kết quả tính toán.
- Kết quả của tính toán đại số thì luôn chính xác còn tính toán số học thường tồn tại giá trị xấp xỉ có thể dẫn đến các sai lệch trong kết quả.
- Trong một số trường hợp hệ thống đại số máy tính sẽ rút gọn thời gian tính toán hơn là các phương pháp tính toán truyền thống.

Hệ thống SMC [14]

Đếm mẫu là vấn đề cổ điển trong tính toán số lượng giải pháp thỏa mãn một tập các ràng buộc. Nó có nhiều ứng dụng trong lĩnh vực khoa học máy tính như trí tuệ nhận tạo, tối ưu hóa chương trình, phân tích lưu lượng thông tin.

Đếm mẫu là kỹ thuật có thể áp dụng cho số nguyên, giá trị logic nhưng không thể áp dụng trực tiếp cho dữ liệu phức tạp như một chuỗi kí tự, để giải quyết vấn đề này nhóm tác giả Loi Luu, Shweta Shinde, Prateek Saxena của trường đại học quốc gia Singapore (National University of Singapore) đã đưa ra giải pháp trong đó có trình bày một công cụ gọi là SMC (string model-counting).

Cho một tập chuỗi kí tự và ràng buộc của chúng, SMC có thể tính biên dựa trên số lượng phần tử của tập chuỗi thỏa mãn ràng buộc với độ chính xác và hiệu quả cao. Nhóm tác giả sử dụng hàm sinh (*generating functions* - GFs) một công cụ toán học quan trọng cho lý luận về chuỗi vô hạn, nó cung cấp cơ chế cho phép xác định số lượng phần tử của một tập chuỗi ràng buộc. Ý tưởng đằng sau hàm sinh (GFs) là mã hóa số lượng các chuỗi có độ dài k như là hệ số thứ k của một đa thức. Các đa thức có thể biểu diễn được dưới dạng các biểu thức hữu hạn, khi đó biểu thức hữu hạn này sẽ có khả năng biểu diễn tập vô hạn các chuỗi.

Trong công cụ SMC có sử dụng hệ thống Mathematica (một hệ thống đại số máy tính) để xử lý các biểu thức đại số, xử lý đa thức và một số các tính toán khác.

Mục tiêu của luận văn

Mục tiêu của luận văn là dựa vào nền tảng lý thuyết về toán học và các khái niệm thuật toán cơ bản để xây dựng các thuật toán và thể hiện của nó bằng các toán tử và cấu trúc điều khiển có trong ngôn ngữ lập trình để giải quyết các vấn đề trong hệ thống đại số máy tính để từ đó phát triển một hệ thống đại số máy tính miễn phí cho phép thực hiện các thao tác tính toán từ cơ bản đến phức tạp như tính giá trị biểu thức, tối giản phân số, tính toán đa thức …Trong đó mục tiêu chính của luận văn là phát triển các hàm xử lý đa thức nhằm thay thế hoàn toàn Mathematica trong công cụ SMC.

Các vấn đề được nêu ra và xử lý trong phạm vi luận văn:

- Xử lý biểu thức
    - Phân tích chuỗi đầu vào để nhận biết biểu thức.

    - Tính giá trị biểu thức.
    - Rút gọn biểu thức.
- Xử lý đa thức
    - Đa thức một biến, nhiều biến.
    - Các phép toán cơ bản trên đa thức.
    - Khai triển đa thức.
- Xây dựng các hàm xử lý cho hệ thống SMC
    - Tìm chuỗi taylor tại một giá trị bất kỳ, đến một hệ số bất kỳ.
    - Xây dựng hàm MAXF, MINF, DEDUP.

Giới thiệu

Chương 1 - Kiến thức nền tảng

Chương này sẽ giới thiệu về ngôn ngữ giả mã là ngôn ngữ được sử dụng trong luận văn để mô tả các thuật toán, khái niệm và ví dụ. Ngoài ra còn tóm tắt các kiến thức toán học cơ bản như số nguyên, số hữu tỉ, biểu thức đại số, cây biểu thức…

Chương 2 - Cấu trúc đệ quy của biểu thức toán học

Chương này liên quan tới các định nghĩa về cấu trúc đệ quy, cấu trúc sau khi rút gọn của biểu thức. Ba toán tử cơ bản dùng để phân tích và xây dựng biểu thức (Kind(), NumberOfOperator(), Operand()), hai toán dựa vào cấu trúc của biểu thức (CompleteSubExpression(), FreeOf()) cũng được định nghĩa trong phần này.

Chương 3 - Thuật toán

Chương này của luận văn sẽ đưa ra các khái niệm về thuật toán toán học (3.1), thuật toán đệ quy (3.2) và một số ví dụ sử dụng thuật toán đệ quy để thiết kế các toán tử trong hệ thống đại số máy tính (3.3).

Chương 4 - Rút gọn biểu thức

Trong phần này của luận văn trình bao gồm ba mục(4.1, 4.2, 4.3) sẽ trình bày cụ thể về quá trình xử lý các thành phần đại số để rút gọn một biểu thức.

Mục 4.1 liệt kê các phép biến đổi đại số được sử dụng trong quá trình rút gọn và đưa ra định nghĩa của biểu thức đại số cơ bản và biểu thức đại số sau khi rút gọn.

Mục 4.2 mô tả các thuật toán rút gọn :rút gọn biểu thức số hữu tỉ , rút gọn một tích, rút gọn một tổng, rút gọn lũy thừa, rút gọn hàm số cơ bản và cuối cùng là thuật toán rút gọn chính. Các thuật toán rút gọn này dựa trên các luật biến đổi đại số cơ bản.

Mục 4.3 đưa ra cài đặt của các thuật toán bằng ngôn ngữ Java

Chương 5 - Cấu trúc của đa thức và biểu thức hữu tỉ

Phần này của luận văn mô tả cấu trúc đa thức và cấu trúc hữu tỉ của một biểu thức đại số. Phần đa thức có một số định nghĩa tổng quát về đa thức một biến (5.1), đa thức nhiều biến (5.2) và đa thức tổng quát (5.3). Mỗi định nghĩa sẽ có các thủ tục để xác định cấu trúc đa thức của một biểu thức. Phần cuối cùng (5.4) sẽ mô tả cấu trúc hữu tỉ của một biểu thức đại số và một thuật toán biến đổi biểu thức về dạng hữu tỉ.

Chương 6 - Các toán tử trong hệ thống SMC

Đây là ứng dụng thực tế của hệ thống. Phần này mô tả các thủ tục thực hiện các toán tử của SMC (TaylorSeries, MINF, MAXF, DEDUP).

Chương 7 - Kiểm thử

Đưa ra một số ca kiểm thử điển hình.

1 Kiến thức nền tảng

1.1 Ngôn ngữ giả mã

Là một ngôn ngữ biểu tượng được sử dụng trong luận văn để mô tả các khái niệm, định lý, ví dụ, đặc biệt là các thuật toán và các thủ tục thực hiện thuật toán. Ngôn ngữ giả mã tương tự như một ngôn ngữ đại số máy tính nhưng nó mang tính hình thức vì sử dụng cả biểu tượng toán học, tiếng anh và tiếng việt trong đó.

Biểu thức toán học trong ngôn ngữ giả mã cũng giống như các biểu thức toán học thông thường nhưng có một số thừa nhận để phù hợp với môi trường tính toán. Các biểu thức được mô tả bằng cấu trúc sử dụng các toán tử và các ký hiệu sau:

- **Số nguyên và phân số**
- **Số thực**
- **Định danh**
- **Toán tử đại số và dấu ngoặc**
- **Hàm số**
- **Các toán tử logic và toán tử quan hệ**
- **Tập hợp và danh sách**
- **Biểu thức toán học trong ngôn ngữ giả mã**

1.2 Tính toán biểu thức và chương trình toán học

Tính toán biểu thức

Thuật ngữ tính toán biểu thức liên quan đến các hành động trong hệ thống đại số máy tính để đáp ứng một biểu thức đầu vào.

Chương trình toán học

Một chương trình toán học hay còn gọi là một thuật toán toán học là một chuỗi các câu lệnh để thực hiện các toán tử và cấu trúc điều khiển trong lập trình đại số máy tính.

1.3 Khái niệm toán học cơ bản

1.3.1 Số nguyên

Trong phần này sẽ đưa ra các tính chất cơ bản của số nguyên và mô tả một số thuật toán quan trọng để thao tác với số nguyên trong đại số máy tính.

1.3.2 Số hữu tỉ

Trong ngôn ngữ giả mã một số hữu tỉ là một phân số a/b với a và $b \neq 0$ là các số nguyên.

2 Cấu trúc của biểu thức đại số

Do biểu thức toán học là các đối tượng dữ liệu trong chương trình đại số máy tính nên hiểu về mối quan hệ giữa các toàn tử và các toán hạng của biểu thức là rất cần thiết. Trong phần này tài liệu sẽ mô tả về cấu trúc thông thường của một biểu thức đại số.

2.1 Cây biểu thức

Cây là một tập hợp hữu hạn các nút trong đó có một nút đặc biệt được gọi là gốc. Giữa các nút có quan hệ phân cấp gọi là quan hệ cha con. Định nghĩa đệ quy của cây [**1**]:

1. Một nút là một cây. Nút đó cũng là gốc của cây.
2. Nếu n là một nút và $T_1, T_2, \dots, T_k$ là các cây với $n_1, n_2 \dots n_k$ lần lượt là gốc thì một cây T mới sẽ được tạo ra bằng cách cho n là nút cha của các nút $n_1, n_2 \dots n_k$. Nghĩa là trên cây T lúc này n là gốc còn các cây $T_1, T_2, \dots, T_k$ là cây con của T, $n_1, n_2 \dots n_k$ là con của nút n

Cấu trúc của một biểu thức bao gồm các mối quan hệ giữa các toán tử và các toán hạng. Một cây biểu thức là một sơ đồ biểu diễn cấu trúc này.

2.2 Cấu trúc đệ quy của biểu thức đại số

Lý do đệ quy quan trọng trong hế thống đại số máy tính là do cấu trúc đệ quy của biểu thức đại số.

2.3 Cấu trúc thông thường của biểu thức đại số

Cấu trúc thông thường của một biểu thức đại số trong hệ thống đại số máy tính tương tự như cấu trúc của một biểu thức trong toán học và trong các ngôn ngữ lập trình phổ biến.

2.4 Cấu trúc rút gọn của một biểu thức đại số

Cấu trúc rút gọn của biểu thức giúp đơn giản hóa quá trình lập trình bằng cách loại bỏ các toán tử không cần thiết và cung cấp cơ chế truy cập dễ dàng hơn tới các toán hạng của biểu thức.

2.5 Các toán tử cơ bản của biểu thức đại số rút gọn

Để phân tích và vận dụng một biểu thức toán học yêu cầu phải truy cập vào các toán tử và toán hạng của biểu thức.

2.5.1 Định nghĩa toán tử $Kind(u)$

1. Nếu u là một biểu thức nguyên tử thì trả về kiểu của u (số nguyên, số hữu tỉ hoặc ký hiệu...)

2. Nếu u là một biểu thức phức tạp thì *Kind(u)* trả về toán tử nằm ở gốc của biểu thức.

2.5.2 Định nghĩa toán tử $NumberOfOperands(u)$

Nếu u là một biểu thức phức hợp thì $NumberOfOperands(u)$ trả về số toán hạng của toán tử gốc của biểu thức. Nếu u không phải là biểu thức phức hợp thì toán tử sẽ trả về 0.

2.5.3 Định nghĩa toán tử $Operand(u, i)$

Nếu u là một biểu thức phức hợp thì toán tử sẽ trả về toán hạng thứ i của u. Nếu u Các toán tử dựa trên cấu trúc của biểu thức

2.6 Các toán tử dựa trên cấu trúc của biểu thức

2.6.1 Định nghĩa toán tử $CompleteSub\text{E}xpression(u)$

Một biểu thức con đầy đủ của u là chính nó hoặc là một toán hạng của các toán tử trong u. Cho u là một biểu thức rút gọn toán tử $CompleteSub\text{E}xpression(u)$ trả về một danh sách các biểu thức con đầy đủ của u.

2.6.2 Định nghĩa toán tử $FreeOf(u)$

Toán tử $FreeOf(u)$ sẽ xác định nếu biểu thức u không phụ thuộc vào biểu thức t (u không chứa t).

Cho u và t là các biểu thức đại số. Toán tử $FreeOf(u, t)$ trả về false nếu t là một biểu thức phụ đầy đủ của u và trả về True nếu t không phải là biểu thức phụ đầy đủ của u.

3 Thuật toán

3.1 Thuật toán toán học

Một thuật toán toán học nói chung là quá trình từng bước để giải quyết các vấn đề toán học, quá trình này có thể thực hiện bởi các chương trình máy tính.

3.2 Thuật toán đệ quy

Phần này chúng ta sẽ tìm hiểu thuật toán đệ quy được sử dụng như thế nào trong đại số máy tính.

3.3 Thủ tục đệ quy

Phần này sẽ đưa ra một số ví dụ minh họa về việc sử dụng thuật toán đệ quy trong hệ thống đại số máy tính.

3.3.1 Toán tử $CompleteSub\text{E}xpression$

- Thủ tục đệ quy thực hiện toán tử $CompleteSub\text{E}xpression$

3.3.2 Toán tử $FreeOf$

- Thủ tục thực thiện toán tử $FreeOf$

4 Rút gọn biểu thức

Quá trình rút gọn là một phần của quá trình tính toán giá trị được định nghĩa như một tập hợp các phép biến đổi rút gọn đại số và biến đổi lượng giác áp dụng cho biểu thức đại số.

4.1 Các phép biến đổi sử dụng trong quá trình rút gọn biểu thức

Các luật biến đổi trong quá trình rút gọn được xác định bởi các tiên đề và những hệ quả biến đổi logic của các tiên đề. Phần này sẽ trình bày các tiên đề cơ bản và vai trò của chúng trong quá trình rút gọn.

Phép phân phối

Phép kết hợp

Phép giao hoán

Biến đổi lũy thừa

Các phép biến đổi cơ bản khác

Các phép đồng nhất cơ bản

Phép biến đổi đơn phân cơ bản:

Phép biến đổi Undefined: Nếu u là một biểu thức phức hợp với một toán hạng là ký hiệu Undefined thì sẽ được rút gọn thành Undefined.

4.1.1 Biểu thức đại số cơ bản và biểu thức đại số rút gọn

Trong phần này sẽ mô tả biểu thức đại số cơ bản và biểu thức đại số rút gọn tương ứng là đầu vào và đầu ra của thuật toán rút gọn.

4.1.2 Thể hiện của biểu thức đại số cơ bản

Lớp AnyNode

Lớp Bae

4.2 Thuật toán rút gọn

Thuật toán rút gọn được thực hiện dựa trên các phép biến đổi cơ bản và các định nghĩa đã nêu ở các phần trước. Thuật toán bao gồm các bước sau:

1. Nếu u là một biểu thức đại số cơ bản thuật toán sẽ trả về một biểu thức đại số rút gọn.
2. Nếu u là một biểu thức đại số rút gọn thì thuật toán trả ra u.

4.2.1 Thủ tục rút gọn chính

```
Procedure Simplify(u);
Input
        u: a là một biểu thức đại số cơ bản dưới dạng ký hiệu;
Output
        Một biểu thức đại số rút gọn dưới dạng ký hiệu hàm hoặc
        là biểu tượng Undefined
Local Variables
        v;
Begin
        if (Kind(u) = integer || Kind(u) = symbol) then
                Return(u);
        else if (Kind(u) = FracOp) then
                Return(SimplifyRationalNumber(u))
        else
                v = Map(Simplify, u);
        if (Kind(v) = PowOp) then
                Return(SimplifyPower(v))
        else if (Kind(v) = ProdOp) then
                Return(SimplifyProduct(v))
        else if (Kind(v) = SumOp) then
                Return(SimplifySum(v))
        else if (Kind(v) = FactOp) then
                Return(SimplifyFactorial(v))
        else
                Return(SimplifyFunction(v))
End
```

Hình 4.1 Thủ tục rút gọn chính

Toán tử Map sẽ được trình bày trong phần phụ lục map

4.2.2 Rút gọn biểu thức số hữu tỉ

Toán tử $SimplifyRNE$ sử dụng để tính toán với số nguyên và phân số.

4.2.3 Rút gọn lũy thừa

Toán tử $SimplifyPower$ sẽ biến đổi v^m thành biểu thức đại số rút gọn hoặc ký hiệu Undefined.

4.2.4 Rút gọn tích

Toán tử $SimplifyProduct$ sẽ biến đổi một tích thành một biểu thức đại số rút gọn hoặc ký hiệu $Undefined$ dựa trên các phép biến đổi sau: biến đổi kết hợp, giao hoán, biến đổi lũy thừa, tính đồng nhất, phép biến đổi nhị phân, biến đổi cơ số, biến đổi về Undefined

4.2.5 Rút gọn tổng

Toán tử $SimplifySum$ sẽ biến đổi một tổng thành một biểu thức đại số rút gọn hoặc ký hiệu $Undefined$.

4.3 Thể hiện của thuật toán rút gọn

Lớp Symplify được thiết kế bao gồm các phương thức tĩnh sử dụng để rút gọn biểu thức.

4.3.1 Phương thức rút gọn biểu thức số hữu tỉ

Phương thức $simplifyRNE$ được thiết kế dựa trên các định nghĩa và các thủ tục giả mã được nêu ở 4.2.2 nhằm mục địch rút bọn hiểu thức số hữu tỉ.

4.3.2 Phương thức rút gọn lũy thừa

Phương thức $simplifyPower$ được thiết kế dựa trên các định nghĩa đã được nêu ở 4.2.2 để rút gọn biểu thức lũy thừa.

4.3.3 Phương thức rút gọn tích

Phương thức $simplifyProduct$ được thiết kế dựa trên các định nghĩa đã được nêu ở 4.2.4 để rút gọn một tích.

4.3.4 Phương thức rút gọn tổng

Phương thức $simplifySum$ được thiết kế dựa trên các định nghĩa đã được nêu ở 4.2.5 để rút gọn một tổng.

4.3.5 Phương thức rút gọn chính

Phương thức $Simplify$ là phương thức chính sử dụng để rút gọn biểu thức cơ bản về dạng biểu thức rút gọn.

5 Cấu trúc của đa thức và biểu thức hữu tỉ

Phần này của tài liệu mô tả cấu trúc đa thức và cấu trúc hữu tỉ của một biểu thức đại số. Phần đa thức có một số định nghĩa tổng quát về đa thức một biến (5.1), đa thức nhiều biến (5.2) và đa thức tổng quát (5.3). Mỗi định nghĩa sẽ có các thủ tục để xác định cấu trúc đa thức của một biểu thức. Phần cuối cùng (5.4) sẽ mô tả cấu trúc hữu tỉ của một biểu thức đại số và một thuật toán biến đổi biểu thức về dạng hữu tỉ.

5.1 Đa thức một biến

5.1.1 Phân tích

Định nghĩa 5.1: (Định nghĩa toán học) u là đa thức một biến x là một biểu thức có dạng:

$$u = u_n x^n + u_{n-1} x^{n-1} + \cdots + u_1 x + u_0$$

Định nghĩa 5.2: Một đơn thức một biến x là một biểu thức toán học u thỏa mãn một trong các tính chất sau:

1. u là một số nguyên hoặc phân số.
2. $u = x$.
3. $u = x^n$ với n là số nguyên và $n > 1$.
4. u là một tích với hai toán hạng thỏa mãn các tính chất 1, 2, 3.

Định nghĩa 5.3: Một đa thức một biến x là một biểu thức thỏa mãn một trong các tính chất sau:

1. u là một đơn thức một biến x.
2. u là một tổng và mỗi toán hạng trong u là một đơn thức một biến x.

Các toán tử cơ bản của đa thức một biến

Định nghĩa 5.4: Cho u là một biểu thức đại số

- Toán tử $MonomialSV(u, x)$ trả về true nếu u là một đơn thức một biến x và trả về false nếu ngược lại.
- Toán tử $PolynomialSV(u, x)$ trả về true nếu u là một đa thức một biến x và trả về false nếu ngược lại.

Toán tử DegreeSV

Định nghĩa 5.5: Cho u là một biểu thức đại số. Nếu u là một đa thức một biến x thì $DegreeSV(u, x)$ sẽ trả về bậc của u theo biến x. Nếu u không là đa thức theo biến x thì toán tử trả về ký hiệu $Undefined$.

Toán tử CoefficientSV

Định nghĩa 5.6: Cho u là một biểu thức đại số. Nếu u là một đa thức một biến x, toán tử $CoefficientSV(u,x,j)$ trả về hệ số u_j của x^j. Nếu $j > deg(u,x)$ thì $CoefficientSV$ trả về 0. Nếu u không là đa thức biến x thì toán tử trả về ký hiệu Undefined.

Toán tử LeadingCoefficientSV

Định nghĩa 5.7: Cho u là biểu thức đại số. Nếu u là một đa thức một biến x, toán tử $LeadingCoefficientSV(u,x)$ trả về $lc(u,x)$. Nếu u không phải đa thức biến x thì toán tử trả về ký hiệu Undefined.

5.1.2 Các thể hiện của đơn thức và đa thức một biến

5.1.2.1 Lớp đơn thức một biến MonomialSV

5.1.2.2 Đa thức một biến PolynomialSV

5.2 Đa thức nhiều biến

Một đa thức chứa một hoặc nhiều biến được gọi là đa thức nhiều biến

Định nghĩa 5.8: (Định nghĩa toán học) một đa thức nhiều biến u trong tập biến $\{x_1, x_2, \dots, x_m\}$ là một tổng hữu hạn của các đơn thức có dạng

$$cx_1^{n_1}x_2^{n_2}\dots x_m^{n_m}$$

với hệ số c là một số hữu tỉ và phần biến n_j là số nguyên không âm.

5.3 Đa thức tổng quát

Có nhiều biểu thức là đa thức trong ngữ cảnh tính toán nhưng không phải là đa thức trong định nghĩa của phần 5.1

Ví dụ: Biểu thức $u = \frac{a}{(a+1)}x^2 + bx + \frac{1}{a}$

Có thể xem u như là một đa thức một biến x với phần hệ số là $\frac{a}{(a+1)}$, b, $\frac{1}{a}$

Định nghĩa 5.9: (Định nghĩa toán học) Cho $c_1, c_2, \dots, c_r$ là các biểu thức đại số và $x_1, x_2, \dots, x_m$ là các biểu thức đại số không phải là số nguyên hoặc phân số.

- Một đơn thức tổng quát trong tập biến $\{x_1, x_2 \dots, x_m\}$ là một biểu thức có dạng

  $$c_1c_2 \dots c_r x_1^{n_1}x_2^{n_2}\dots x_m^{n_m}$$

  với số mũ n_j là số nguyên không âm và c_i thỏa mãn điều kiện sau

  $$FreeOf(c_i, x_j) \rightarrow true \text{ với } j = 1,2, \dots, m$$

  Biểu thức x_j là biến tổng quát và c_i gọi là hệ số tổng quát. Biểu thức $x_1^{n_1}x_2^{n_2}\dots x_m^{n_m}$ gọi là phần biến của đơn thức, nếu không có biến tổng quát trong đơn thức thì phần biến có giá trị là 1. Biểu thức $c_1 c_2 \dots c_r$ gọi là phần hệ số của đơn thức, hệ số bằng 1 nếu không có phần hệ số.

- Một biểu thức u là một đa thức tổng quát nếu nó là một đơn thức tổng quát hoặc là tổng của các đơn thức tổng quát trong tập biến $\{x_1, x_2, \dots, x_m\}$

Định nghĩa 5.10: (Định nghĩa tính toán) Một đơn thức tổng quát trong tập biến tổng quát $S = \{x_1, x_2 \dots, x_m\}$ là một biểu thức toán học u thỏa mãn các luật sau:

1. $FreeOf(u, x_j) \rightarrow true$ với $j = 1, \dots, m$
2. $u \in S$
3. $u = x^n$ với $x \in S$ và n là số nguyên lớn hơn 1
4. u là một tích và mỗi toán hạng của u là một đơn thức tổng quát trong S

Định nghĩa 6.11: (Định nghĩa tính toán) Một đa thức tổng quát trong tập $S = \{x_1, x_2 \dots, x_m\}$ là một biểu thức toán học u thỏa mãn các luật sau:

1. U là một đơn thức tổng quát trong S.
2. U là tổng mà mỗi toán hạng của nó là một đơn thức tổng quát trong S.

5.3.1 Các toán tử cơ bản của đơn thức tổng quát

5.3.1.1 Toán tử MonomialGPE

Định nghĩa 5.11: Cho u là một biểu thức đại số và cho v là biến x hoặc tập biến tổng quát S. Toán tử $MonomialGPE(u, v)$ trả về true nếu u là một đơn thức tổng quát trong S và ngược lại trả về false.

5.3.1.2 Toán tử CoefficientGME

Toán tử trả về một danh sách gồm hai phần tử c và m trong đó m là bậc của đơn thức và c là hệ số của x^m hoặc trả về ký hiệu $Undefined$.

Định nghĩa 5.12: Cho u là một biểu thức đại số toán tử $CoefficitentGME(u, x)$ được định nghĩa bởi các luật sau ([c, m]):

1. Nếu $u = x$ thì: $CoefficitentGME(u, x) \rightarrow [1, 1]$
2. Nếu u là một lũy thừa và :

$$base = Operand(u, 1)$$

$$exponent = Operand(u, 2)$$

   nếu $u = x$ và $exponent$ là số nguyên lớn hơn 1 thì trả về [1, exponent].

3. Nếu u là một tích thì toán tử trả về Undifined nếu không có toán hạng nào trong u là đơn thức phụ thuộc x. Nếu trong u có toán hạng phụ thuộc x và toán hạng đó có bậc m là số dương thì trả về $[u/x^m$, m].
4. Nếu u không là đơn thức phụ thuộc x thì toán tử trả về Undefined.

5.3.1.3 Toán tử DegreeGME

Toán tử tìm bậc của đơn thức theo tập biến tổng quát s.

Định nghĩa 5.13: Cho u là một biểu thức đại số vá tập biến tổng quát S toán tử $DegreeGME(u, x)$ được định nghĩa bởi các luật sau:

1. Nếu $u \in S$ thì toán tử trả về 1.
2. Nếu u là lũy thừa thì toán tử trả về $exponent(u)$.
3. Nếu u là tích của các toán hạng $u_1, u_2, \dots, u_n$ thì bậc của u bằng tổng bậc của các toán hạng $u_1, u_2, \dots, u_n$với tập biến tổng quát S.
4. Nếu u không thuộc S: nếu $u = 0$ toán tử trả về Undefined ngược lại trả về 0.

5.3.1.4 Thể hiện của đơn thức tổng quát

Lớp $GeneralMonomial$ được thiết kế dựa trên các phân tích về đơn thức tổng quát.

5.3.2 Các toán tử cơ bản của đa thức tổng quát

5.3.2.1 Toán tử PolynomialGPE

Định nghĩa 5.14: Toán tử $PolynomialGPE(u, v)$ trả về true nếu u là một đa thức với biến tổng quát trong $\{x\}$ hoặc S ngược lại trả về false.

5.3.2.2 Toán tử Variables

Toán tử này sẽ xác định tập biến tổng quát tự nhiên của một biểu thức.

Định nghĩa 5.15: Cho u là một biểu thúc đại số. Toán tử $Variables(u)$ được định nghĩa bởi các luật sau đây:

1. Nếu u là một số nguyên hoặc phân số thì

$$Variable(u) \to \emptyset$$

2. Giả sử u là một lũy thừa. Nếu số mũ của u là một số nguyên lớn hơn 1 thì

$$Variables(u) \to \{Operand(u, 1)\}$$

   Trong các trườngh hợp còn lại

$$Variables(u) \to \{u\}$$

3. Giả sử u là một tổng. Toán tử $Variable(u)$ là hợp của các biến tổng quát của mỗi toán hạng trong u được xác định bởi các luật 1, 2, 4, 5.
4. Giả sử u là một tích. Toán tử $Variable(u)$ chứa hợp của các biến tổng quát của mỗi toán hạng trong u được xác định bởi các luật 1, 2, 5.
5. Nếu u không thỏa mãn các trường hợp trên thì

$$Variables(u) \to \{u\}$$

5.3.2.3 Toán tử DegreeGPE

Định nghĩa 5.16:

- Cho $S = \{x_1, x_2, \dots, x_m\}$ là tập các đơn thức tổng quát. Cho $u = c_1 \dots c_r . x_1^{n_1} \dots x_m^{n_m}$ là một đơn thức với phần hệ số khác 0. Bậc của u trong S là tổng số mũ của các biến tổng quát:

$$\deg(u, S) = n_1 + n_2 \dots + n_m$$

  Theo quy ước toán học bậc của đơn thức 0 là $-\infty$

- Nếu u là một đa thức tổng quát và là tổng của các đơn thức thì $\deg(u, S)$ là giá trị lớn nhất của các bậc của các đơn thức trong u. Nếu u chứa một biến x thì bậc của u là $\deg(u, x)$.

Định nghĩa 5.17: Cho u là một biểu thức đai số và v là một biến tổng quát x hoặc tập biến tổng quát S. Toán tử DegreeGPE có dạng

$$DegreeGPE(u, v)$$

Khi u là một đa thức tổng quát trong v thì toán tử trả về bậc của u. Nếu u không là đa thức tổng quát trong v thì toán tử trả về ký hiệu Undefined.

Định nghĩa 5.18: Cho u là một biểu thức toán học và

$$S = Variables(u)$$

Toán tử $\deg(u, S)$ gọi là tổng bậc của biểu thức u

5.3.2.4 Toán tử CoefficientGPE

Định nghĩa 5.19: Cho u cho một biểu thức toán học. Nếu u là đa thức tổng quát với biến tổng quát x và $j \geq 0$ là một số nguyên thì toán tử $CoefficientGPE(u, x, j)$ trả về tổng phần hệ số của tất cả các đơn thức của u mà có phần biến là x^j. Nếu không có đơn thức nào có phần biến là x^j thì toán tử trả về 0. Nếu u không phải là đa thức biến x thì toán tử trả về ký hiệu Undefined.

5.3.2.5 Toán tử LeadingCoefficientGPE

Định nghĩa 5.20: Cho u là một biểu thức đại số. Nếu u là một đa thức tổng quát trong x thì $LeadingCoefficientGPE$(u) với x được định nghĩa như là tổng của phần hệ số của tất cả các đơn thức với phần biến $x^{\text{degreeGpe}(u,x)}$. Hệ số đầu tiên sẽ được biểu diễn bởi $lc(u, x)$.

Định nghĩa 5.21: Cho u là một đa thức tổng quát trong x. Toán tử

$$LeadingCoefficientGPE(u, x)$$

trả về $lc(u, x)$. Nếu u không là đa thức tổng quát trong x thì toán tử trả về ký hiệu Undefined.

5.3.2.6 Thiết kế lớp tương ứng của đa thức tổng quát

5.3.3 Các toán tử thao tác với đa thức tổng quát

Trong phần này mô tả hai toán tử sử dụng để thao tác trong khi tính toán đa thức tổng quát. Cả hai toán tử dựa trên hai tính chất phân phối vào giao hoán của phép cộng

$$a(a+b) = ab + ac$$

$$(a+b)c = ac + bc$$

5.3.3.1 Toán tử $CollectTerms$

Việc rút gọn hệ số của đa thức thường xuyên được áp dụng trong quá trình tính toán đại số. Trong suốt quá trình rút gọn toán tử này chỉ áp dụng cho các đơn thức có phần hệ số là số hữu tỉ (số nguyên hoặc phân số). Việc rút gọn các hệ số được thực hiện bởi toán tử $CollectTerms$.

Định nghĩa 5.22: Một đa thức tổng quát u có phần hệ số thu gọn trong tập biến tổng quát S nếu thỏa mãn một trong số các thuộc tính sau:

1. U là một đơn thức tổng quát trong S
2. U là tổng của các đơn thức tổng quát trong S với phần biến rõ ràng (distinct) (Định nghĩa này tương tự định nghĩa Định nghĩa 5.9 ngoại trừ trong luật số hai yêu cầu phần biến phải được xác định rõ ràng.)

5.3.3.2 Toán tử $Expand$

Toán tử $Expand$ áp dụng hai phép biến đổi phân phối tới tích và lũy thừa để thu được một tổng.

Ví dụ:

$$(x+2)(x+3)(x+4) \rightarrow x^3 + 9x^2 + 26x + 24$$

Định nghĩa 5.23 sẽ mô tả dạng đầu ra của toán tử $Expand$

Định nghĩa 5.23: Một biểu thức đại số u có dạng khai triển nếu tập các biến tổng quát được tính bởi toán tử $Variable(u)$ không chứa một tổng.

5.4 Biểu thức hữu tỉ tổng quát.

Trong ngữ cảnh toán học một biểu thức hữu tỉ được định nghĩa như là thương của hai đa thức. Trong phần này của luận văn sẽ trình bày về cấu trúc biểu thức hữu tỉ của một biểu thức đại số và mô tả thuật toán biến đổi một biểu thức về dạng hữu tỉ.

Định nghĩa 5.24: cho $S = \{x_1, x_2, \dots, x_m\}$ là một tập biến tổng quát. Một biểu thức đại số u là một biểu thức hữu tỉ GRE (general rational expression) trong S nếu $u = p/q$ với p và q là các đa thức tổng quát trong S.

5.4.1 Toán tử $Numerator$ và $Denominator$

Để có thể xác định một biểu thức đã ở dạng hữu tỉ hay không thì cần định nghĩa chính xác tử số và mẫu số của biểu thức. Toán tử $Numerator$ và $Denominator$ sử dụng để làm việc này. Hai toán tử được định nghĩa bởi các luật biến đổi dưới đây.

Định nghĩa 5.25: Cho u là một biểu thức đại số

1. Nếu u là một phân số thì

$$Numerator(u) \rightarrow Operand(u, 1)$$

$$Denominator\ (u) \rightarrow Operand(u, 2).$$

2. Giả sử u là lũy thừa. Nếu số mũ của u là số nguyên âm hoặc phân số âm thì

$$Numerator(u) \rightarrow 1, \qquad Denominator(u) \rightarrow u^{-1}$$

nếu không thì

$$Numerator(u) \rightarrow u, \qquad Denominator\ (u) \rightarrow 1$$

3. Giả sử u là tích và $v = Operand(u, 1)$ thì:

$$Numerator(u) \rightarrow Numerator(v) * Numerator(u/v)$$

$$Denominator(u) \rightarrow Denominator(v) * Denominator(u/v)$$

4. Nếu không thỏa mãn các luật trên thì:

$$Numerator(u) \rightarrow u, \qquad Denominator(u) \rightarrow 1$$

Định nghĩa 5.26: Cho $S = \{x_1, x_2, \dots, x_m\}$ là một tập biến tổng quát. Một biểu thức u là biểu thức hữu tỉ trong S nếu $Numerator(u)$ và $Denominator(u)$ là các đa thức tổng quát trong S.

5.4.2 Toán tử RationalGPE

Định nghĩa 5.27: Cho u là một biểu thức đại số và cho v là một biến tổng quát x hoặc tập biến tổng quát S. Toán tử $RationalGPE(u, v)$ trả ra $True$ nếu u là một biểu thức hữu tỉ tổng quát trong x hoặc S và trả về $False$ trong trường hợp còn lại.

Toán tử được định nghĩa bởi luật biến đổi sau:

$$RationalGPE(u, v)$$
$$\rightarrow PolynomialGPE(Numerator(u), v)\ and\ PolynomialGPE(Denominator(u), v)$$

5.4.3 Toán tử RationalVariables

Toán tử này định nghĩa một tập biến tự nhiên của biểu thức hữu tỉ tổng quát.

Định nghĩa 5.28: Cho u là một biểu thức đại số. Toán tử $RationalVariables(u)$ được định nghĩa bởi các luật sau:

$$RationalVariables(u) \\ \rightarrow Variables(Numerator(u)) \cup Variables(Denominator(u))$$

với toán tử $Variable$ được định nghĩa ở Định nghĩa 5.15.

5.4.4 Hữu tỉ hóa một biểu thức đại số

Quá trình hữu tỉ hóa dựa trên việc biến đổi kết hợp các toán hạng của một tổng trên một mẫu số chung.

Định nghĩa 5.29: Một biểu thức đại số u ở dạng hữu tỉ hóa nếu nó thỏa mãn các luật sau:

1. Nếu u là số nguyên, phân số, ký hiệu, giai thừa hoặc dạng hàm.
2. u là các loại khác và xem u như là một biểu thức hữu tỉ trong tập $S = RationalVariables(u)$ thì
    a. Mỗi biểu thức v trong S đều ở dạng hữu tỉ với $Denominator(v) = 1$
    b. Phần hệ số của mỗi đơn thức trong $Numerator(u)$ và $Denominator(u)$ là số nguyên

Toán tử RationalizeExpression

Toán tử này sẽ biến đổi một biểu thức đại số u thành một biểu thức tương đương ở dạng hữu tỉ. Toán tử được hiểu trong ngữ cảnh rút gọn bao gồm các phép biến đổi lũy thừa sau:

$$u^v u^w \rightarrow u^{v+w}$$

$$(u^v)^n \rightarrow u^{vn}$$

$$(u\ v)^n \rightarrow u^n v^n$$

Với u, v, w là các biểu thức đại số và n là số nguyên.

5.4.5 Thể hiện của biểu thức hữu tỉ

Lớp $GenneralRationalExpression$ là lớp thể hiện được thiết kế dựa trên các đặc điểm của biểu thức hữu tỉ tổng quát.

6 Các toán tử trong hệ thống SMC

6.1 Khai triển Taylor

6.1.1 Toán tử $Derivative$

Định nghĩa 6.1: Cho biểu thức đại số u. Toán tử $Derivative(u, x)$ trả về đạo hàm bậc một của u tính theo biến x được định nghĩa bởi các luật sau:

1. Nếu $u = x$ thì

$$Derivative(u, x) \rightarrow 1$$

2. Nếu $u = x^w$ thì
   $Derivative(u, x) \rightarrow w * v^{w-1} * Derivative(v, x) + Derivative(w, x) * v^w * \ln(v)$
3. Giả sử u là một tổng và cho $v = Operand(u, 1)$ và $w = u - v$ khi đó
   $$Derivative(u, x) \rightarrow Derivative(v, x) + Derivative(w, x)$$
4. Giả sử u là một tích và cho $v = Operand(u, 1)$ và $w = u/v$ khi đó
   $$Derivative(u, x) \rightarrow w * Derivative(v, x) + v * Derivative(w, x)$$
5. Nếu $u = \sin(v)$
   $$Derivative(u, x) \rightarrow \cos(v) * Derivative(v, x)$$
6. Nếu $u = \cos(v)$
   $$Derivative(u, x) \rightarrow -\sin(x) * Derivative(v, x)$$
7. Nếu $u = \tan(v)$
   $$Derivative(u, x) \rightarrow \sec^2(v) * Derivative(v, x)$$
8. Nếu $u = \cot(v)$
   $$Derivative(u, x) \rightarrow -\csc(v) * Derivative(v, x)$$
9. Nếu $u = sec(v)$
   $$Derivative(u, x) \rightarrow \sec(v) * \tan(v) * Derivative(v, x)$$
10. Nếu $u = csc(v)$
   $$Derivative(u, x) \rightarrow -\csc(v) * cot(v) * Derivative(v, x)$$
11. Nếu $FreeOf(u, x) = true$ thì
   $$Derivative(u, x) \rightarrow 0$$

6.1.2 Toán tử $HigherDerivative$

Cho u là một biểu thức đại số, toán tử $HigherDerivative$ sẽ trả về đạo hàm bậc n của biểu thức u với biến x

6.1.3 Toán tử $Taylor$Series

Định nghĩa 6.2: Cho f là một hàm số có đạo hàm riêng liên tục tới cấp $n + 1$ trong khoảng nào đó chứa điểm a. Chuỗi taylor được tạo bởi hàm f tại điểm $x = a$ có dạng

$$f(a) + f'(0)(x - a) + \frac{f''(a)}{2!}(x - a)^2 + \cdots + \frac{f^n(a)}{n!}(x - a)^n + \cdots$$
$$= \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(x - a)^k$$

Cho u là biểu thức đại số toán tử $TaylorSeries$ sẽ trả về chuỗi taylor cấp n của u theo biến x tại điểm a.

6.2 Các toán tử khác

6.2.1 Toán tử $MINF$

Định nghĩa 6.3: [**15**] Cho $F_1(z),\ F_2(z)$ là các biểu thức đại số với biến tổng quát z, $F(z) = MINF(F_1(z), F_2(z))$ sẽ được tính bởi công thức sau:

$$[z^i]F(z) = MIN\left([z^i]F_1(z), [z^i]F_2(z)\right) \ \ \forall i \geq 0$$

Toán tử $MINF$ trả về a_i là hệ số nhỏ nhất trong các hệ số của z^i trong F_1 và F_2.

Ví dụ:

$$\mathrm{MINF}(1 + \mathrm{z} + \mathrm{z}^2\,, 1 + 2\mathrm{z}^2) = 1 + \mathrm{z}^2$$

6.2.2 Toán tử $MAXF$

Định nghĩa 6.4: [**15**] Cho $F_1(z),\ F_2(z)$ là các biểu thức đại số với biến tổng quát z, $F(z) = MAXF(F_1(z), F_2(z))$ sẽ được tính bởi:

$$[z^i]F(z) = MAX\left([z^i]F_1(z), [z^i]F_2(z)\right) \ \ \forall i \geq 0$$

Toán tử $MAXF$ trả về a_i là hệ số lớn nhất trong các hệ số của z^i trong F_1 và F_2.

Ví dụ:

$$\mathrm{MAXF}(1 + \mathrm{z} + \mathrm{z}^2\,, 1 + 2\mathrm{z}^2) = 1 + \mathrm{z} + 2\mathrm{z}^2$$

6.2.3 Toán tử $DEUP$

Định nghĩa 6.5: Cho F(z) là một hàm số thì

$$DEDUP\big(F(z)\big) = \sum_{[z^i]F(z) > 0} z^i$$

Toán tử DEDUP trả về hàm sinh mới với các hệ số của z^i được gán bằng 1 khi $[z^i]F(z) > 0$. [**15**]

Ví dụ: $DEDUP\big(F(z)\big) = 1 + \mathrm{z} + \mathrm{z}^2 + \mathrm{z}^3 + \ldots = \ 1/(1 - \mathrm{z})$

7 Kiểm thử

Đưa ra một số ca kiểm thử điển hình.

- Kiểm thử phương thức $taylorSeries$ (Tìm chuỗi taylor theo bậc n và biến x)

Stt	Đầu vào	Kết quả mong đợi	Kết quả thực tế
1	$\exp(x)\,, 3$	$1 + x + \frac{x^2}{2} + \frac{x^3}{3}$	$1 + x + \frac{x^2}{2} + \frac{x^3}{3}$

2	$\frac{1}{1-3}, 4$	$1 + x + x^2 + x^3 + x^4$	$1 + x + x^2 + x^3 + x^4$
3	$\exp(x)\sin(x)$, 6	$x + x^2 + \frac{x^3}{3} - \frac{x^5}{30} - \frac{x^6}{90}$	$x + x^2 + \frac{1}{3}x^3 + \frac{-1}{30}x^5 + \frac{-1}{90}x^6$

- Kiểm thử phương thức $minF$

Stt	Đầu vào	Kết quả mong đợi	Kết quả thực tế
1	$1 + \mathrm{x} + \mathrm{x}^2, 1 + 2\mathrm{x}^2$	$1 + \mathrm{x}^2$	$1 + \mathrm{x}^2$
2	$4x^5 - 15x^3 - 11x - 1, -14\mathrm{x}^5 + 14\mathrm{x}^4 + 22\mathrm{x}^2 + 14$	$-1 - 11x - 15x^3 - 14x^5$	$-1 + (-11)x + (-15)x^3 + (-14)x^5$

- Kiểm thử phương thức $maxF$

Stt	Đầu vào	Kết quả mong đợi	Kết quả thực tế
1	$1 + \mathrm{x} + \mathrm{x}^2, 1 + 2\mathrm{x}^2$	$1 + \mathrm{x} + 2\mathrm{x}^2$	$1 + \mathrm{x} + 2\mathrm{x}^2$
2	$4x^5 - 15x^3 - 11x - 1, -14\mathrm{x}^5 + 14\mathrm{x}^4 + 22\mathrm{x}^2 + 14$	$14 + 22x^2 + 14x^4 + 4x^5$	$14 + 22x^2 + 14x^4 + 4x^5$

- Kiểm thử phương thức $dedup$ (*Note: mặc định chuỗi taylor sẽ được tính tại giá trị x=0*)

Stt	Đầu vào	Kết quả mong đợi	Kết quả thực tế
1	$1/(1 - \mathrm{x})$, 3	$1 + \mathrm{x} + \mathrm{x}^2 + \mathrm{x}^3$	$1 + \mathrm{x} + \mathrm{x}^2 + \mathrm{x}^3$
2	$\frac{5}{x+1}$, 4	$1 - 5x + x^2 - 5x^3 + x^4$	$1 - 5x + x^2 - 5x^3 + x^4$

Kết luận

Kết quả đạt được

Trong phạm vị luân văn này tôi hướng tới mục đích là tìm hiểu, nghiên cứu và phát triển một hệ thống đại số máy tính miễn phí nhằm thay thế hệ thống đại số máy tính thương mại sẵn có trong việc đáp ứng các yêu cầu của hệ thống SMC. Qua 7 chương, luận văn đã trình bày về phương pháp tiếp cận, phân tích và giải quyết các vấn đề gặp phải trong quá trình xây dựng hệ thống.

Các kết quả đặt được:

- Xây dựng được hệ thống đại số máy tính cơ bản cho phép thao tác với biểu thức đại số
    - Phân tích chuỗi đầu vào để nhận biết biểu thức.
    - Tính giá trị biểu thức.
    - Rút gọn biểu thức.
- Xử lý đa thức
    - Đa thức một biến, nhiều biến.
    - Các phép toán cơ bản trên đa thức.
    - Khai triển đa thức.
- Xây dựng các hàm xử lý cho hệ thống SMC
    - Tìm chuỗi taylor vủa một hàm số tại một giá trị bất kỳ, đến một hệ số bất kỳ.
    - Xây dựng hàm MAXF, MINF, TRUNC, DEDUP.

Hướng nghiên cứu trong lương lai

Với những gì đã làm được trọng phạm vi luận văn tôi hy vọng trong tương lai sẽ có những cải thiện giúp tăng chất lượng của hệ thống.

- Hoàn thiện hệ thống
    - Khả năng rút gọn biểu thức hữu tỉ.
    - Khả năng phân tích chuỗi để nhận biết biểu thức.
    - Hỗ trợ xử lý biểu thức logic.
    - Tăng hiệu xuất thực hiện bằng cách cải thiện các thuật toán, các phương thức tính toán để giảm thời gian.
    - Phát triển lại bằng ngôn ngữ C++ để có thể cạnh tranh về hiệu năng với các hệ thống đã có.
- Thêm giao diện để thân thiện với người dùng.

Tài liệu tham khảo

Tiếng việt

1. Đỗ Xuân Lôi (1999), *Cấu trúc dữ liệu và giải thuật*, Nhà xuất bản thống kê.
2. Trương Ninh Thuận – Đặng Đức Hạnh (2013), *Giáo trình phân tích và thiết kế hướng đối tượng*, Nhà xuất bản Đại Học Quốc gia Hà Nội.

Tiếng anh

3. Hazem Mohamed El-Alfy. (1997). *Computer algebraic and its applications*, B.Sc., Faculty of Engineering, Alexandria University.
4. Chee Keng Yap. (2000). *Fundamental Problems of Algorithmic Algebra*, Oxford University Press, New York.
5. David Musser. (1971). *Algorithms for Polynomial Factorization*, PhD thesis, Department of Computer Science, University of Wisconsin.
6. F. Winkler. (1996). *Polynomial Algorithms in Computer Algebra*, SpringerVerlag, New York.
7. Hans Vangheluwe, Bhama Sridharan and Indrani A.V. (2013). *An algorithm to implement a canonical representation of algebraic expression and equations in AToM*.
8. Henri Cohen. (1993). *A Course in Computational Algebraic Number Theory*, Springer-Verlag, New York.
9. J. H. Davenport, Y. Siret, and E. Tournier. (1988). *Computer Algebra, Systems and Algorithms for Algebraic Computation*, Academic Press, New York.
10. James F. Epperson. (2002). *An Introduction to Numerical Methods and Analysis*, John Wiley & Sons, New York.
11. Joachim von zur Gathen and J¨urgen Gerhard. *Modern Computer Algebra*, Cambridge University Press, New York, 1999.
12. Joel S. Cohen (2002). *Computer Algebra and Symbolic Computation: Elementary Algorithms*. A K Peters, Natick, MA
13. Joel S. Cohen (2002). *Computer Algebra and Symbolic Computation: Mathematical Methods*, A K Peters, Natick, MA.
14. John W. Gray. (1997). *Mastering Mathematica, Programming Methods and Applications*, Second Edition. Academic Press, New York.
15. Loi Luu, Shweta Shinde, Prateek Saxena. (2014). *A Model Counter For Constraints Over Unbounded Strings*, School of Computing, National University of Singapore.
16. Michael J. Wester. (1999). *Computer Algebra Systems, A Practical Guide*, John Wiley & Sons, Ltd., New York.
17. Richard Andrew Mealing. (2010). *Simplifying Numerical Expressions*, The University of Liverpool.
18. Richard J. Fateman. (1999). *Symbolic mathematics system evaluators*, In Michael J. Wester, editor, Computer Algebra Systems, A Practical Guide, pages 255–284. John Wiley & Sons, Ltd., New York.

19. Richard J. Gaylord, N. Kamin, Samuel, and Paul R. Wellin. (1996). *An Introduction to Programming with Mathematica*, Second Edition. Springer-Verlag, New York.
20. Richard Liska, Ladislav Drska, Jiri Limpouch, Milan Sinor, Michael Wester, Franz Winkler. (1999). *Computer algebraic*, *Algorithms, System and Applications*.
21. Richard Zippel. (1993). *Effective Polynomial Computation*, Kluwer Academic Publishers, Boston.
22. Stephen Wolfram. *The Mathematica Book. Fourth Edition*. Cambridge University Press., New York, 1999.

Website

23. https://en.wikipedia.org/wiki/Computer_algebra_system
24. https://en.wikipedia.org/wiki/Taylor_series
25. http://www.le.ac.uk/users/dsgp1/COURSES/DERIVATE/TAYLOR.PDF
26. http://mathworld.wolfram.com/TaylorSeries.html

Phụ lục

Gồm các thủ tục và cài đặt của một số toán tử.