

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH
MÔN HỌC
LẬP TRÌNH SYMBOLIC TRONG TRÍ TUỆ NHÂN TẠO



BÁO CÁO ĐỒ ÁN
RÚT GỌN LƯỢNG GIÁC

Lớp: CS314.H21.KHTN

GVLT: Nguyễn Đình Hiền

GVHDTH: Nguyễn Thị Ngọc Diễm

SVTH: Hoàng Ngọc Thạch – 14520811

Lê Thị Ngọc Thúy – 14520932

Trần Quốc Long – 14520490

Nguyễn Minh Trí – 14520994

TP HCM, ngày 18 tháng 06 năm 2017

MỤC LỤC

I.	MỤC TIÊU – YÊU CẦU	3
1.	Mục tiêu	3
2.	Yêu cầu.....	3
II.	PHẠM VI ĐỀ TÀI	3
III.	PHƯƠNG PHÁP GIẢI BÀI TOÁN.....	3
1.	Thu thập và biểu diễn tri thức cho bài toán rút gọn lượng giác	4
2.	Phân loại bài toán	5
3.	Heuristic áp dụng trong việc rút gọn.....	5
	Heuristic chọn ra hằng đẳng thức để rút gọn biểu thức:	5
	Heuristic chọn luật (công thức lượng giác) trong tập luật rút gọn để rút gọn biểu thức:	5
	Heuristic chọn luật khai triển trong tập luật khai triển để khai triển biểu thức:	5
4.	Các hàm cần thiết trong quá trình giải quyết bài toán	6
4.1.	Các hàm cơ bản:	6
4.2.	Các hàm Heuristic ^[1]	6
5.	Phương pháp giải quyết cho từng bài toán con.....	8
6.	Giải quyết bài toán lớn (Thuật giải).....	9
IV.	PHƯƠNG PHÁP KẾT NỐI MAPLE VỚI GIAO DIỆN	10
1.	Kết nối Maple với C# thông qua Batch File	10
2.	Tạo file Batch, file thực thi	10
3.	Thực hiện kết nối với C#.....	11
3.1.	Hiển thị biểu thức theo Form toán học.....	11
3.2.	Hàm xuất kết quả để kiểm tra.....	12
3.3.	Hiển thị lên giao diện	13
V.	DEMO VÀ ĐÁNH GIÁ	14
1.	Demo	14
2.	Đánh giá	15
3.	Bảng Phân Công.....	16
VI.	TÀI LIỆU THAM KHẢO	16

I. MỤC TIÊU – YÊU CẦU

1. Mục tiêu

Xây dựng chương trình rút gọn các biểu thức lượng giác có giao diện cho người dùng và in ra lời giải chi tiết của các bước rút gọn biểu thức lượng giác.

- INPUT: Biểu thức lượng giác cần rút gọn.

Ví dụ: $f := (\cos(x) + \sin(x))^2 + (\sin(x) - \cos(x))^2$

- OUTPUT: Bài giải từng bước trong quá trình rút gọn biểu thức lượng giác.

Ví dụ: Input $f := (\cos(x) + \sin(x))^2 + (\sin(x) - \cos(x))^2$

Bước 1:

Áp dụng: $(\cos(x) + \sin(x))^2 = \cos(x)^2 + 2\sin(x)\cos(x) + \sin(x)^2$

Ta có:

$$f = (\cos(x) - \sin(x))^2 + \cos(x)^2 + 2 * \sin(x) * \cos(x) + \sin(x)^2$$

Bước 2:

Áp dụng: $(\cos(x) + \sin(x))^2 = \cos(x)^2 + 2\sin(x)\cos(x) + \sin(x)^2$

Ta có:

$$f = 2\cos(x)^2 + 2\sin(x)^2$$

Bước 3:

Áp dụng:

$$\sin(x)^2 = 1 - \cos(x)^2$$

Ta có:

$$f = 2$$

2. Yêu cầu

Chương trình dễ sử dụng, hướng giải quyết mang hơi hướng tự nhiên như lời giải con người.

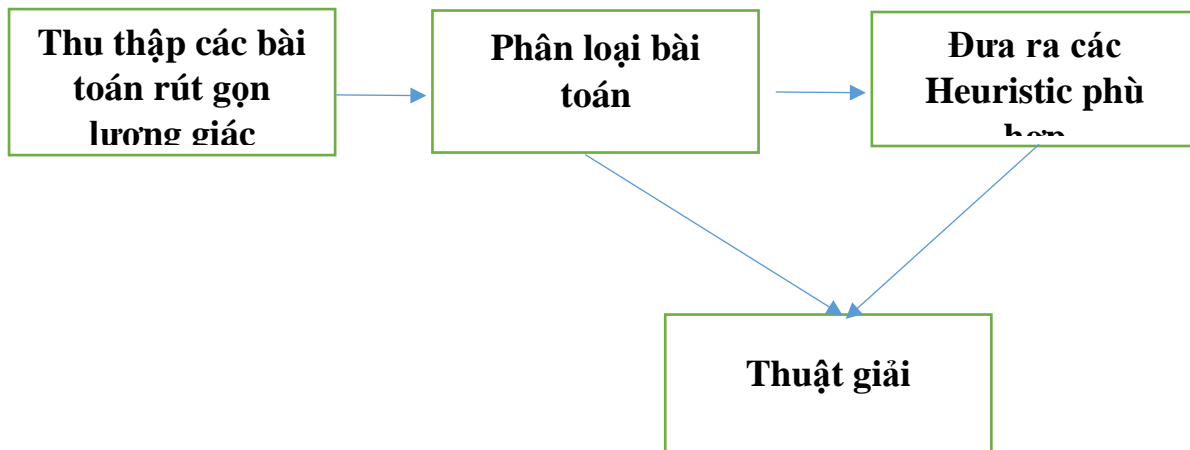
Chương trình rút gọn được các bài toán lượng giác cơ bản và bình thường và in ra các bước thực hiện bài toán.

II. PHẠM VI ĐỀ TÀI

- Bài toán Rút gọn lượng giác trong chương trình lớp 10.

III. PHƯƠNG PHÁP GIẢI BÀI TOÁN

Quy trình giải quyết bài toán:



1. Thu thập và biểu diễn tri thức cho bài toán rút gọn lượng giác

- Mục đích: Thu thập các công thức lượng giác lớp 10, các hằng đẳng thức đã học.
- Biểu diễn tri thức:
- Công thức lượng giác được biểu diễn theo 2 dạng: dạng khai triển, và dạng rút gọn.
- **Dạng khai triển:** chứa các công thức lượng giác mà vế trái là một hàm lượng giác, vế phải là dạng khai triển của hàm lượng giác đó.
- **Dạng rút gọn:** chứa các công thức lượng giác mà vế trái là một hàm lượng giác phức tạp, vế phải là dạng rút gọn của hàm lượng giác đó.
- Biểu diễn các hằng đẳng thức theo dạng vế trái bằng vế phải, với vế phải là biểu thức mở rộng của vế trái.
- Tri thức được lưu trữ ở 3 file .txt với dòng đầu tiên của file .txt là begin_rules và dòng cuối cùng là end_rules.

✓ Luatkhaitrien.txt

Ví dụ: $\tan(x) = \sin(x)/\cos(x)$

✓ Luatrutgon.txt

Ví dụ: $\sin(x)^2 + \cos(x)^2 = 1$

✓ Hangdangthuc.txt

Ví dụ: $(_A + _B)^2 = _A^2 + 2 \cdot _A \cdot _B + _B^2$

- Cấu trúc dữ liệu dùng biểu diễn tri thức: Gồm 3 tập luật R_RG, R_KT, R_HDT với mỗi tập luật là một list các luật được đọc từ file .txt tương ứng. Mỗi luật trong tập luật được lưu lại như một câu lệnh trong Maple.

(*) 3 file .txt được gửi cùng source code.

2. Phân loại bài toán

- Chia bài toán rút gọn biểu thức lượng giác thành 3 dạng chính:

- Biểu thức lượng giác là đa thức

Ví dụ: $\cos(x) + \sin(x) + \sin(x)^2$

- Biểu thức lượng giác là căn thức

Ví dụ: $\sqrt{\sin(x) + \cos(x)}$

- Biểu thức lượng giác là phân thức

Ví dụ: $\frac{\sin(x)}{\cos(x) + \sin(x)}$

3. Heuristic áp dụng trong việc rút gọn

Heuristic chọn ra hằng đẳng thức để rút gọn biểu thức:

- Ưu tiên 1: chọn hằng đẳng thức mà có thể gom được nhiều phần tử nhất trong biểu thức lượng giác.
- Ưu tiên 2: Nếu các số phần tử được gom ở ưu tiên 1 là như nhau thì sẽ ưu tiên chọn hằng đẳng thức mà biểu thức sau khi gom là đơn giản nhất.
- So sánh sự đơn giản dựa vào hàm $\text{Expr_Length}(\text{bieuthuc})$, biểu thức được cho là đơn giản hơn khi giá trị trả về của $\text{Expr_Length}(\text{bieuthuc})$ là nhỏ hơn .
 - $\text{Expr_Length}(\text{bieuthuc})$ là hàm tự định nghĩa.

Heuristic chọn luật (công thức lượng giác) trong tập luật rút gọn để rút gọn biểu thức:

- Ưu tiên 1: chọn luật có vế phải là hằng số.
- Ưu tiên 2: chọn luật mà không sinh ra cung mới trừ khi $\text{Expr_Length}(\text{bieuthuc}) \leq 2$
Ví dụ: $\sin(x)$, thì cung mới là $\sin(2x)$
- Ưu tiên 3: nếu trong biểu thức có số lượng phần tử tan, cot nhiều hơn sin, cos thì ưu tiên chọn luật biến đổi từ sin, cos sang tan, cotan.
- Ưu tiên 4: chọn luật có thể gom được nhiều phần tử nhất trong biểu thức (Expr_Length max).
- Ưu tiên 5: chọn luật mà biểu thức sau khi gom nhóm là đơn giản nhất (Expr_Length min).

Heuristic chọn luật khai triển trong tập luật khai triển để khai triển biểu thức:

- Ưu tiên 1: nếu trong biểu thức có nhiều tan, cot hơn thì ưu tiên luật khai triển thành tan, cotan.
- Ưu tiên 2: ưu tiên chọn luật $\tan = \sin/\cos$, $\cot = \cos/\sin$.
- Ưu tiên 3: ưu tiên chọn luật khai triển có dạng $\tan(x)^2 = \dots$, $\cot(x)^2 = \dots$
- Heuristic chọn cách đặt thừa số chung:
- Ưu tiên 1: chọn cách đặt làm xuất hiện $\sin(x)^2 + \cos(x)^2$
- Ưu tiên 2: chọn cách đặt để biểu thức sau khi dùng có số lượng tan và cot là ít nhất.
- ✦ Các ưu tiên sẽ đi từ trên xuống, ưu tiên sau chỉ được thực hiện khi tất cả các ưu tiên trước nó không phù hợp với biểu thức (không đủ điều kiện thực hiện).
- ✦ Các heuristic được áp dụng khi và chỉ khi có nhiều sự lựa chọn cho luật áp dụng được trên biểu thức.

4. Các hàm cần thiết trong quá trình giải quyết bài toán

4.1. Các hàm cơ bản:

- **Expr_Length(bieuthuc)**: hàm tính độ dài của biểu thức lượng giác. Được tính bằng tổng độ dài của các thành phần con có trong biểu thức
- \sin, \cos, \tan, \cot có độ dài là 1
- Hằng số: 0
- Với $A*B = (\text{độ dài của } A) + (\text{độ dài của } B)$
- Với $A^B = (\text{độ dài của } A) * (\text{độ dài của } B)$
- **Is_Sub_Coeff**: hàm kiểm tra A và B có tỉ lệ với nhau hay không.

Ví dụ: $A=a+b$, $B=k*a+k*b$, kết luận A và B tỉ lệ với nhau

- **Is_Sub_Add**: hàm kiểm tra biểu thức dạng tổng A có trong biểu thức dạng tổng B hay không.

Ví dụ: $A= a+b$, $B=k*a+c+k*b$, kết luận A có trong B tại $k*a+k*b$

- **Is_SubExpr**: hàm kiểm tra biểu thức A có trong biểu thức B không.

4.2. Các hàm Heuristic^[1]

- **Find_HDT_Sim**: hàm tìm hằng đẳng thức để áp dụng cho biểu thức theo heuristic.
- **Find_Arc**: hàm tìm các đối (các cung) của biểu thức.
- **Get_Var_bien_so**: hàm lấy các đối dạng X – số trong biểu thức.
- **Find_Rule_Sim**: hàm tìm một luật trong tập luật rút gọn có thể áp dụng được lên biểu thức.

- **Find_Common_2Poly**: hàm tìm phần tử chung giữa hai biểu thức dạng đa thức.
- **Has_Form**: hàm kiểm tra các luật ưu tiên (sử dụng trong các hàm Heuristic) có trong biểu thức không. Các luật ưu tiên:

$$\sin(x)^2 + \cos(x)^2, \tan(x) = \frac{\sin(x)}{\cos(x)}, \cot(x) = \frac{\cos(x)}{\sin(x)},$$

$$\tan(x)^2 = \frac{1}{\cos(x)^2} - 1, \cot(x)^2 = \frac{1}{\sin(x)^2} - 1$$

- **Count_Func**: hàm đếm số lượng hàm cần tìm trong biểu thức, sử dụng trong hàm Heuristic để tìm số lượng hàm sin, cos, tan, cotan.

Ví dụ: đếm hàm sin, cos trong biểu thức gọi hàm như sau:

Count_Func (bieuthuc, {sin, cos});

- **Factor_Expr**: Hàm đặt thừa số chung trong biểu thức, nếu không đặt được thừa số chung thì trả về biểu thức ban đầu được đưa vào.
- **Apply**: Hàm dùng để áp dụng một luật lên biểu thức.
- **Simplify_Simple**: hàm dùng để rút gọn biểu thức. Hàm này rút gọn biểu thức bằng phương pháp thử dùng hằng đẳng thức, luật rút gọn, đặt thừa số chung.
- **Simplify_Simple_Rad**: tương tự như hàm Simplify_Simple nhưng dùng cho căn thức
- **Find_HDT_Exp**: Hàm tìm một hằng đẳng thức (sử dụng Heuristic) có trong biểu thức để áp dụng.
- **Find_Rule_Exp**: Hàm tìm một luật khai triển (sử dụng Heuristic) có thể áp dụng vào biểu thức.
- **Mul_To_Sum**: Hàm khai triển tích thành tổng trong biểu thức.
- **Com_Denom**: Hàm qui đồng trong biểu thức
- **Expand_Simple**: Hàm thực hiện khai triển biểu thức, hàm này dùng phương pháp thử các hằng đẳng thức, luật khai triển, biến đổi tích thành tổng, qui đồng mẫu thức.
- **Expand_Simple_Rad**: Hàm thực hiện khai triển căn thức, tương tự như hàm ở trên.
- **Is_Common_Denom**: Hàm xét xem biểu thức có cùng mẫu hay không.
- **Simplify_Tri_Step2**: Thử khai triển biểu thức bằng hằng đẳng thức, luật khai triển, biến đổi tích thành tổng, qui đồng mẫu thức (Expand_Simple). Dùng hàm Simplify_Simple để rút gọn biểu thức. Nếu biểu thức rút gọn mới cho kết quả tốt hơn (Expr_Length nhỏ hơn hoặc nếu Expr_Length bằng nhau thì lấy kết quả có độ dài string ngắn hơn) thì lấy kết quả

rút gọn mới tốt hơn. Độ quy bước trên tối đa 3 lần, nếu tìm thấy kết quả tốt hơn thì trả về kết quả đó.

- **Omit_Sqrt**: Hàm dùng để xử lý căn thức (rút căn).
- **Simplify_Tri_Step2_Rad**: tương tự như hàm Simplify_Tri_Step2, có thêm bước khử căn thức.
- **Omit_Denom_Denom**: hàm khử mẫu ở mẫu.
- **Simplify_Tri_Poly**: Hàm tổng quát thực hiện rút gọn biểu thức lượng giác có dạng đa thức và phân thức (nêu rõ ở phần giải quyết bài toán con).
- **Simplify_Tri_Rad**: Hàm tổng quát thực hiện rút gọn biểu thức lượng giác dạng căn thức (nêu rõ ở phần giải quyết bài toán con).
- **Is_Radical**: hàm kiểm tra biểu thức có phải dạng căn thức.
- **Simplify_Tri**: Hàm tổng quát thực hiện rút gọn biểu thức lượng giác.
- **print_SOL**: Hàm này in lời giải cho bài toán.

(*) Source code chi tiết ở file đính kèm.

5. Phương pháp giải quyết cho từng bài toán con

- Biểu thức lượng giác là đa thức và phân thức:
 - Bước 1: Quy đồng biểu thức, nếu biểu thức có dạng phân số.
 - Bước 2: Dùng hàm Simplify_Simple để rút gọn biểu thức lần lượt theo hằng đẳng thức, luật rút gọn, đặt thừa số chung.
 - Bước 3: Thử khai triển biểu thức bằng hằng đẳng thức, luật khai triển, biến đổi tích thành tổng, qui đồng mẫu thức (Expand_Simple). Dùng hàm Simplify_Simple để rút gọn biểu thức. Nếu biểu thức rút gọn mới cho kết quả tốt hơn (Expr_Length nhỏ hơn hoặc nếu Expr_Length bằng nhau thì lấy kết quả có độ dài string ngắn hơn) thì lấy kết quả rút gọn mới tốt hơn. Độ quy bước trên tối đa 3 lần, nếu tìm thấy kết quả tốt hơn thì trả về kết quả đó.


```

> TriSim[Simplify_Tri_Poly] := proc(expr)
# Hàm tổng quát thực hiện rút gọn biểu thức lượng giác expr có dạng đa thức và phân thức
local g, h, l, newexpr;
global SOL, gold, count, goal;

# Heuristic: Nếu ở mẫu có dạng phân số thì thực hiện qui đồng.
newexpr := Omit_Denom_Denom(expr);
# Bước 1
g := Simplify_Simple(newexpr);
gold := g;
count := 0;

# Bước 2
goal := Simplify_Tri_Step2(g);
return goal;
end:# Simplify_Tri_Poly

```

- Biểu thức là căn thức: trả về (kết quả rút gọn biểu thức trong căn)^(số mũ của căn)
- Dùng hàm Find_HDT_Sim để tìm hằng đẳng thức áp dụng cho biểu thức trong căn. Áp dụng luật vừa tìm, nếu độ dài của kết quả sau khi áp dụng nhỏ hơn độ dài biểu thức trước khi áp dụng thì ghi nhận luật đó vào lời giải. Lặp lại bước trên với biểu thức mới (biểu thức sau khi áp dụng luật) cho đến khi không tìm được hằng đẳng thức nào để áp dụng nữa.
- Sau đó, lấy kết quả của bước trên. Dùng hàm Find_Rule_Sim để tìm luật áp dụng cho biểu thức. Tương tự bước áp dụng hằng đẳng thức bên trên.
- Cuối cùng, lấy kết quả ở bước trên rồi thử đặt thừa số chung (sử dụng Factor_Expr). Nếu không đặt được thừa số chung thì trả về (kết quả tìm được ở bước trên)^(số mũ). Nếu có thể đặt thừa số chung thì ghi nhận lại kết quả sau khi đặt thừa số chung, thực hiện gọi đệ quy hàm với biểu thức mới là (kết quả đặt thừa số chung)^(số mũ)

```

> TriSim[Simplify_Tri_Rad] := proc(expr, dk)
# Hàm tổng quát thực hiện rút gọn biểu thức lượng giác expr có dạng đa thức và phân thức
local g, h, l;
global SOL, gold, count, goal;
# Bước 1

g := Simplify_Simple_Rad(expr, dk); g := Omit_Sqrt(g, dk);
gold := g;
count := 0;

# Bước 2
goal := Simplify_Tri_Step2_Rad(g, dk);
return goal;
end:# Simplify_Tri_Rad

```

6. Giải quyết bài toán lớn (Thuật giải)

- Kiểm tra biểu thức chứa căn hay không: sử dụng hàm kiểm tra Is_Radical.

- Nếu chứa căn: giải quyết bài toán theo căn (giải quyết bài toán mà biểu thức là căn thức).
- Nếu không chứa căn: giải quyết bài toán theo hướng mà biểu thức là đa thức và phân thức.

```

> TriSim[Simplify_Tri] := proc(expr)
global SOL;
local lfunc, temp, newexpr, dk, fl, f, solanlap, flag;
    SOL := [];
    solanlap := 0;
    lfunc := Is_Radical(expr);
    if not lfunc[1] then
        return Simplify_Tri_Poly(expr);
    else
        lfunc := lfunc[2];
        if nargs = 2 then dk := args[2];
        else dk := {};fi;

        if nops(lfunc) = 1 and op(lfunc) = expr then

            temp := Simplify_Tri_Rad(op(lfunc),dk);
            return temp;

        else
            newexpr := expr;
            flag := true;
            while flag and solanlap <= 4 do
                solanlap := solanlap +1;
                for fl in lfunc do
                    SOL := [op(SOL), ["\t THỰC HIỆN RÚT GỌN: ", 'g' = fl]];
                    temp := Simplify_Tri_Rad(fl, dk);
                    newexpr := subs(fl = temp, newexpr);
                    SOL := [op(SOL), ["\t SỬY RA: ", 'f' = newexpr]];

                od;
                flag, lfunc := Is_Radical(newexpr);
            od;
            SOL := [op(SOL), ["\t THỰC HIỆN RÚT GỌN: ", 'f' = newexpr]];
            return Simplify_Tri_Poly(newexpr);

        fi;
    fi;
end: # Simplify_Tri

```

IV. PHƯƠNG PHÁP KẾT NỐI MAPLE VỚI GIAO DIỆN

1. Kết nối Maple với C# thông qua Batch File

Để kết nối Maple với một ứng dụng khác thông qua batch file đầu tiên ta cần tạo 1 file batch, file batch này chứa các hàm mà ta cần ứng dụng khác (sử dụng maple) thực thi. Ta tạo 1 package chỉ chứa các hàm xử lý rút gọn biểu thức đại số sau đó gói package đó trong 1 file mpl rồi tạo 1 batchfile gọi tới file .mpl đó để sử dụng.

2. Tạo file Batch, file thực thi

- Tạo file solve.bat để thực thi chính cho chương trình.
- *Hàm ghi nội dung cho file solve.bat*

Theo như đã trình bày, bài toán qui về còn 2 dạng:

- Rút gọn biểu thức lượng giác bình thường. (1)

- Rút gọn biểu thức lượng giác chứa căn thức. (2)

Ở trường hợp thứ 2, nhận thấy cần có điều kiện để trục căn thức. Các hàm maple cũng đã tổ chức để có thể nhận input về điều kiện bổ sung để rút gọn căn thức. Do đó nội dung của file solve.bat có 2 kiểu nội dung để đáp ứng nhu cầu trên.

Loại 1: Rút gọn biểu thức không có điều kiện bổ sung.

```
string[] query = {
    "restart;",
    "url := currentdir();",
    "libname := libname, url;",
    "with(TriSim);",
    "f := " + inputTextBox.Text + ";",
    "Simplify_Tri(f);",
    "print_SOL(f);"
};
File.WriteAllLines(path_Problem, query);
```

Loại 2: Rút gọn biểu thức có điều kiện bổ sung

```
string strDK = "{" + tbDK.Text + "}";
string[] query2 = {
    "restart;",
    "url := currentdir();",
    "libname := libname, url;",
    "with(TriSim);",
    "f := " + inputTextBox.Text + ";",
    "Simplify_Tri(f, "+strDK+");",
    "print_SOL(f);"
};
File.WriteAllLines(path_Problem, query2);
```

3. Thực hiện kết nối với C#

- Điểm khác biệt lớn nhất của việc kết nối qua batch file so với kết nối qua commandline đó là mọi thao tác nhập xuất đều được thực hiện qua file thay vì thực hiện trực tiếp lên console. Về cơ bản, ta cũng dùng phương thức Process.Start nhưng các thông tin cần dùng trong process sẽ khác, không còn hỗ trợ việc nhập xuất lên console nữa.

- Để thực thi được file *solve.bat* ta cần dẫn đến file cmaple.exe.

3.1. Hiển thị biểu thức theo Form toán học

- Ở đây, chúng tôi sử dụng Mimetex để hiển thị biểu thức toán học. C# có mimetex.dll hỗ trợ cho việc này.

- Mimetex:

- Input: biểu thức dạng latex.
- Output: ảnh biểu thị cho biểu thức toán học.

- Từ một biểu thức gốc, có thể chuyển sang biểu thức dạng latex nhờ hàm có sẵn của maple. Cụ thể:

strOutput = latex(strInput,output=string)

Trong đó:

- strInput là chuỗi biểu thức ban đầu
- strOutput là chuỗi biểu thức dạng latex.

- Hàm để tạo ảnh gif cho biểu thức toán học:

```
private void CreateEquation(string equation, string gifFileName) {
    if (equation.Length > 0) {
        string gifFilePath = GetGifFilePath(gifFileName);
        NativeMethods.CreateGifFromEq(equation, gifFilePath); //tạo file ảnh gif
        của biểu thức
    }
}
```

3.2. Hàm xuất kết quả để kiểm tra

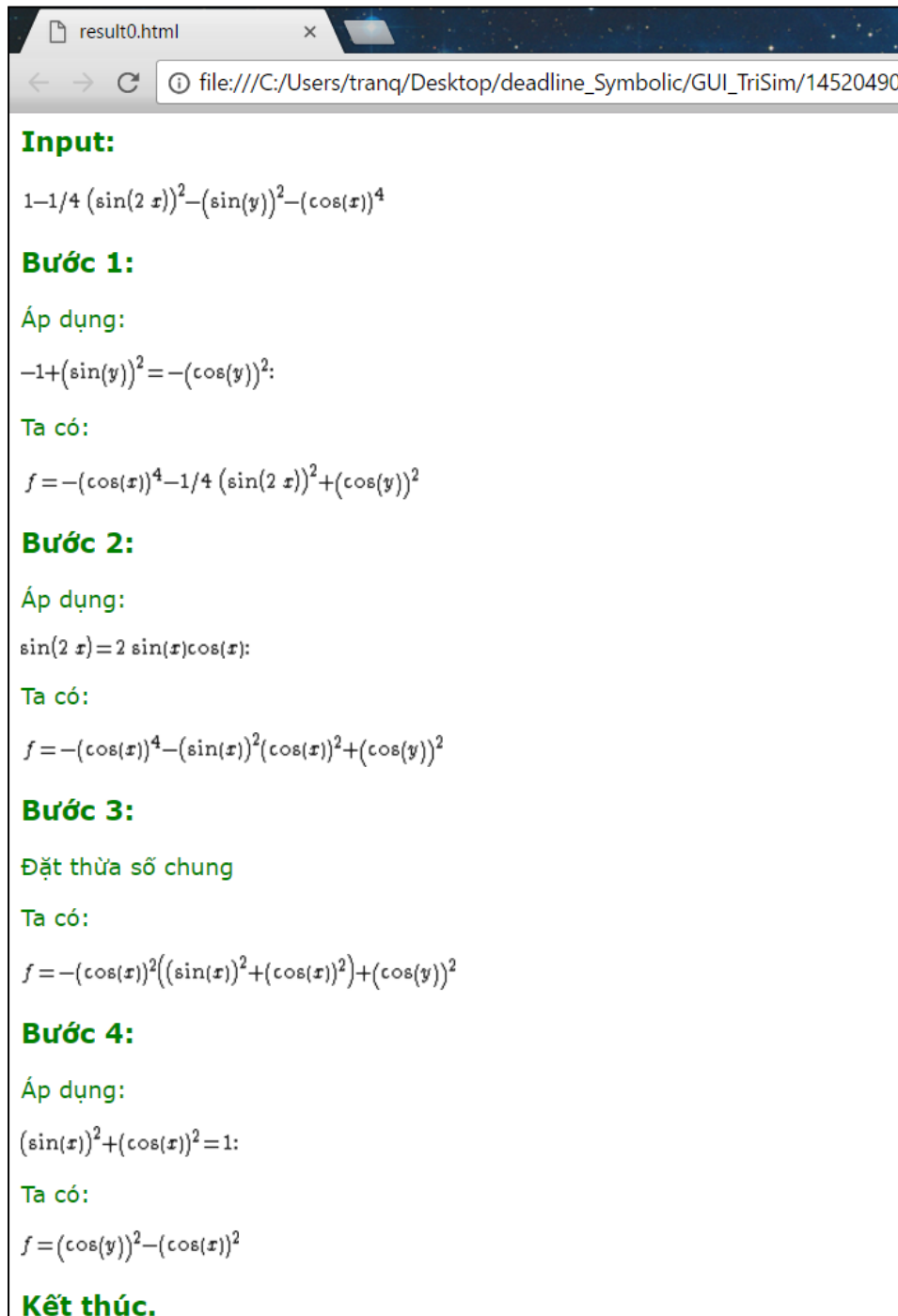
- Ở bước này, chúng tôi phát triển hàm để hiển thị kết quả, gồm các phần:
 1. Xóa các bước giải thừa trong tập SOL.
 2. Hiển thị kết quả trong file maple.
 3. Xuất kết quả ra file *.txt. Ở đây, các biểu thức ở dạng latex để có thể dùng mimetex hiển thị lên giao diện C# sau này.
- Cụ thể được trình bày trong hàm “**print_SOL**” trong **RUT GON LUONG GIAC.mw**.
- Nội dung file text kết quả:

```
Input:
1-1/4, \left( \sin \left( 2 \, x \, \right) \, \right)^{2} - \left( \sin \left( y \, \right) \, \right)^{2} - \left( \cos \left( x \, \right) \, \right)^{4}
Bước 1:
Áp dụng:
-1+ \left( \sin \left( y \, \right) \, \right)^{2} = - \left( \cos \left( y \, \right) \, \right)^{2}:
Ta có:
f= \left( \cos \left( x \, \right) \, \right)^{4} - 1/4, \left( \sin \left( 2 \, x \, \right) \, \right)^{2} + \left( \cos \left( y \, \right) \, \right)^{2}
Bước 2:
Áp dụng:
\sin \left( 2 \, x \, \right) = 2 \, \sin \left( x \, \right) \cos \left( x \, \right) :
Ta có:
f= \left( \cos \left( x \, \right) \, \right)^{4} - \left( \sin \left( x \, \right) \, \right)^{2} \left( \cos \left( x \, \right) \, \right)^{2} + \left( \cos \left( y \, \right) \, \right)^{2}
Bước 3:
Đặt thừa số chung
Ta có:
f= \left( \cos \left( x \, \right) \, \right)^{2} \left( \left( \sin \left( x \, \right) \, \right)^{2} + \left( \cos \left( x \, \right) \, \right)^{2} \right) + \left( \cos \left( y \, \right) \, \right)^{2}
Bước 4:
Áp dụng:
\left( \sin \left( x \, \right) \, \right)^{2} + \left( \cos \left( x \, \right) \, \right)^{2} = 1:
Ta có:
f= \left( \cos \left( y \, \right) \, \right)^{2} - \left( \cos \left( x \, \right) \, \right)^{2}
Kết thúc.
```

Hình 1. Kết quả được xuất ra file text.

3.3. Hiển thị lên giao diện

- Ở bước này, chúng tôi tạo một file *.html để biểu thị cho các bước giải của bài toán. Nội dung file được trình bày cụ thể trong source code C#.



Hình 2. Kết quả hiển thị file html.

- Đồng thời nếu chạy bằng maple, kết quả cũng hiển thị tương tự:

```
print_SOL(f);
```

Input:

Bước 1:

Áp dụng:

Ta có:

Bước 2:

Áp dụng:

Ta có:

Bước 3:

Đặt thừa số chung

Ta có:

Bước 4:

Áp dụng:

Ta có:

Kết thúc

$$f := 1 - \frac{1}{4} \sin(2x)^2 - \sin(y)^2 - \cos(x)^4$$

$$\cos(y)^2 - \cos(x)^2$$

$$-\cos(x)^4 - \frac{1}{4} + \cos(y)^2 + \frac{1}{4} \cos(2x)^2$$

$$1 - \frac{1}{4} \sin(2x)^2 - \sin(y)^2 - \cos(x)^4$$

$$-1 + \sin(y)^2 = -\cos(y)^2$$

$$f = -\cos(x)^4 - \frac{1}{4} \sin(2x)^2 + \cos(y)^2$$

$$\sin(2x) = 2 \sin(x) \cos(x)$$

$$f = -\cos(x)^4 - \sin(x)^2 \cos(x)^2 + \cos(y)^2$$

$$f = -\cos(x)^2 (\sin(x)^2 + \cos(x)^2) + \cos(y)^2$$

$$\sin(x)^2 + \cos(x)^2 = 1$$

$$f = \cos(y)^2 - \cos(x)^2$$

Hình 3. Kết quả chạy thử trên giao diện maple.

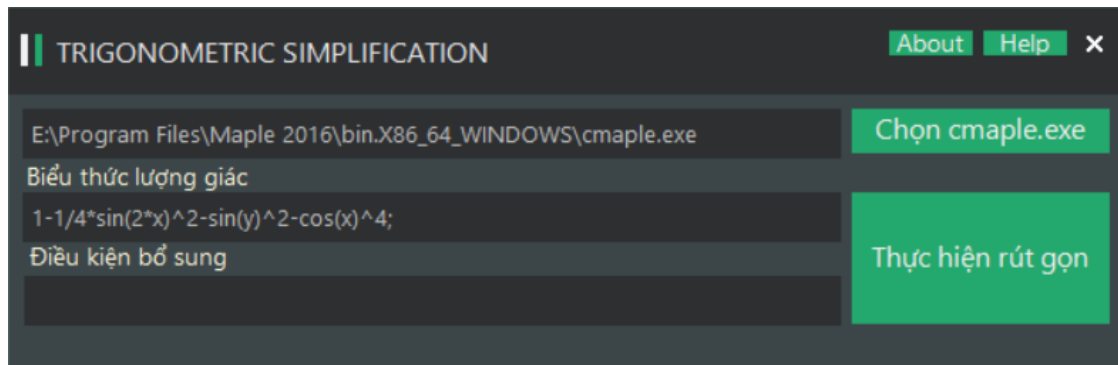
V. DEMO VÀ ĐÁNH GIÁ

1. Demo

- Link clip demo:

<https://www.youtube.com/watch?v=wdrmNX67w3M&feature=share>

- Cụ thể:



Hình 4. Giao diện chính của chương trình.

- Textbox đầu tiên: hiển thị đường dẫn đến file cmaple.exe.
 - Textbox “Biểu thức lượng giác”: nhập biểu thức lượng giác cần rút gọn.
 - Textbox “Điều kiện bổ sung”: nhập điều kiện bổ sung (dùng để trục căn,...).
 - Button “Chọn cmaple.exe”: open file dialog để chọn đường dẫn đến cmaple.exe
 - Button “Thực hiện rút gọn”: tiến hành rút gọn biểu thức lượng giác.
 - Button “Help”: giải thích và hướng dẫn chạy chương trình.
 - Button “About”: nhóm tác giả.
- Sau khi giải xong, chương trình sẽ hiển thị cửa sổ mới để trình bày các bước giải:



Hình 5. Giao diện Các bước giải.

2. Đánh giá


- Kết quả chạy bài toán và đánh giá được so sánh với simplify() của Maple


- Trình bày ở file TEST-RUT-GON-LUONG-GIAC (sẽ nộp chung với tất cả các file)

3. Bảng Phân Công

Công việc	Ghi chú
Thu thập tri thức, tổ chức tri thức	Trí
Thiết kế tập luật, cấu trúc dữ liệu	Thạch
Tìm hiểu và cài đặt chương trình	Tất cả thành viên (Tìm ra phương pháp chung cho bài toán, cài đặt các hàm cần thiết)
Test chương trình	Thúy
Thiết kế giao diện và kết nối với Maple	Long
Báo cáo	Tất cả thành viên (từng thành viên bổ sung vào bảng báo cáo, một thành viên tổng hợp, chỉnh sửa hình thức báo cáo)

VI. TÀI LIỆU THAM KHẢO

 Tài liệu hướng dẫn thực hành LẬP TRÌNH SYMBOLIC TRONG TRÍ TUỆ NHÂN TẠO (Tác giả: Nguyễn Đình Hiền, Hồ Long Vân, Nguyễn Thị Ngọc Diễm)

 <https://diendantoanhoc.net/topic/119643-c%C3%B4ng-th%E1%BB%A9c-hi%E1%BB%A3ng-gi%C3%A1c-r%C3%BAt-g%E1%BB%8Dn-m%E1%BB%99t-s%E1%BB%91-bi%E1%BB%83u-th%E1%BB%A9c-l%C6%B0%E1%BB%A3ng-gi%C3%A1c-t%E1%BB%95ng-qu%C3%A1t/>