|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатики и систем управления

КАФЕДРА Теоретической информатики и компьютерных технологий

**ЛАБОРАТОРНАЯ РАБОТА № 1**

**ПО КУРСУ:**

***«ТЕОРИЯ ФОРМАЛЬНЫХ ЯЗЫКОВ»***

Студент *Сербин Д. А.*

Преподаватель *Непейвода А.Н.*

*Москва, 2021 г.*

Оглавление

[1. Первая задача 3](#__RefHeading___Toc207_319232588)

[2. Вторая задача 6](#__RefHeading___Toc209_319232588)

# 1. Первая задача

Задание: дана сигнатура для TRS. Написать алгоритм унификации линейных термов в этой сигнатуре.

Входные данные в разных местах синтаксически анализируются по-разному: используется посимвольный анализ каждой полученной строки, после чего строки парсятся для дальнейшего использования информации из них.

«Терм» – это «конструктор» или «константа» или «переменная». «Конструктор» хранит в себе количество аргументов.

После обработки термов на экран выведутся все подстановки и сам унификатор. Если унификации прошло неудачна, то на экран выведется, что унифицировать нельзя.

Функциональные тесты, на которые был ориентирован первый вариант реализации:

Тест 1:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = f(w, g(A))

second = f(f(y,x), z)

Ответ: термы унифицируются посредством подстановок w:=f(y,x), z:=g(A). Унификатор: f(f(y,x),g(A))

Тест 2:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = f(w, g(A))

second = f(f(x,g(A)), z)

Ответ: термы унифицируются посредством подстановок w:=f(x,g(A)), z:=g(A). Унификатор: f(f(x,g(A)),g(A))

Тест 3:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = f(w, g(A))

second = f(f(g(A),x), z)

Ответ: термы унифицируются посредством подстановок w:=f(g(A),x), z:=g(A). Унификатор: f(f(g(A),x),g(A))

Тест 4:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = g(z)

second = g(f(A, w))

Ответ: термы унифицируются посредством подстановок z:=f(A,w). Унификатор: g(f(A,w))

Тест 5:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = f(w, g(A))

second = f(f(g(w),x), z)

Ответ: термы НЕ унифицируются посредством подстановок

Тест 6:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = f(w, g(A))

second = f(f(g(A),x), A)

Ответ: термы НЕ унифицируются посредством подстановок

Тест 7:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = f(w, g(A))

second = f(f(g(A),x), g(z))

Ответ: термы унифицируются посредством подстановок w:=f(g(A),x), g(z):=g(A). Унификатор: f(f(g(A),x),g(A))

Тест 8:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = f(w, z)

second = g(A)

Ответ: термы НЕ унифицируются посредством подстановок

Тест 9:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = f(w, f(g(A), y))

second = f(y, z)

Ответ: термы унифицируются посредством подстановок w:=y, z:=f(g(A),y). Унификатор: f(y,f(g(A),y))

Тест 10:

constructors = g(1),A(0),f(2)

variables = x, y,z,w,v

first = f(A, g(A))

second = f(f(g(A),x), z)

Ответ: термы НЕ унифицируются посредством подстановок

Тест второго этапа, на которых альфа-версия показала неправильное поведение:

constructors = g(1), A(0), f(2)

variables = x, y, z, w, v

first = f(A, g(A))

second = f(y, g(x))

Причины неверного поведения на тестах и способ его исправления: алгоритм не рассматривал случай когда термы имеют одинаковые внутренние внутренние конструкторы и выдавал, что термы нельзя унифицировать. Для исправления бага был создан отдельный метод, работающий в тех случаях, когда внутренние конструкторы совпадают.

**2.** **Вторая задача**

Задание: дана SRS. Написать алгоритм проверки ее конфлюэнтности по перекрытию.

С учетом того, что правая часть правил не задействована она игнорируется.

Был построен алгоритм посимвольно проверяющий суффикс и преффикс левую часть одного выражения, а потом посимвольная проверка суффикса и преффикса разных выражений

Функциональные тесты, на которые был ориентирован первый вариант реализации:

Тест 1:

fgf → ghhg

hh →

Ответ: система, возможно, не конфлюэнтна (есть перекрытие внутри терма fgf ).

Тест 2:

fgc → ghhh

hfg → gch

Ответ: система, возможно, не конфлюэнтна (префикс терма fgc совпадает с суффиксом терма hfg ).

Тест 3:

fgg → ghhg

hg → ghfh

Ответ: система конфлюэнтна.

Тест 4:

fgg → ghhg

hg → ghfh

fggfg → ghhg

hg → ghfh

Ответ: система, возможно, не конфлюэнтна (префикс терма fgg совпадает с суффиксом терма fggfg ).

Тест 5:

fgg → ghhg

hg → ghfh

hfg → ghhg

hgfg → ghfh

Ответ: система, возможно, не конфлюэнтна (префикс терма fgg совпадает с суффиксом терма hfg ).

Тест 6:

fgg → ghhg

hg → ghfh

fgh → ghhg

hg → ghfh

Ответ: система, возможно, не конфлюэнтна (префикс терма hg совпадает с суффиксом терма fgh ).

Тест 7:

fgg → ghhg

hg → ghfh

fgg → ghhg

hghfg → ghfh

Ответ: система, возможно, не конфлюэнтна (префикс терма fgg совпадает с суффиксом терма hghfg ).

Тест 8:

fgg → ghhg

hg → ghfh

fggfhc → ghhg

hggh → ghfh

Ответ: система, возможно, не конфлюэнтна (есть перекрытие внутри терма hggh ).

Тест второго этапа, на которых альфа-версия показала неправильное поведение:

fgg → ghhg

fghjkabcdefghjkabcdefghjkabcdefghjkabcdefghjkabcdefghjkabcde → ghfh

Причины неверного поведения на тестах и способ его исправления: алгоритм не проверял последнее левое выражение на конфлюэнтность. Для исправления бага было добавлено дополнительное условие для проверки последнего выражения.