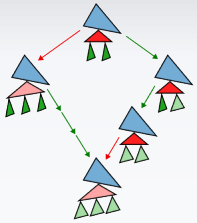
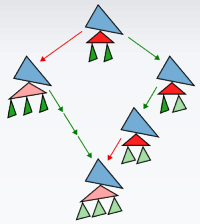


Лабораторная работа 1

1. ($\equiv 0 \mod 3$) Реализовать алгоритм унификации Мартелли–Монтанари.
2. ($\equiv 1 \mod 3$) Реализовать алгоритм упрощения по переименовке систем переписывания термов.
3. ($\equiv 2 \mod 3$) Реализовать алгоритм проверки выполнения отношения Кнута–Бендикса для заданной TRS.



Синтаксис входных данных

Синтаксис записи входных данных для 1 задачи:

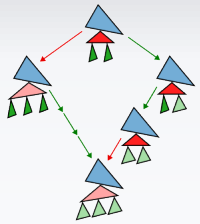
constructors = ([буква]([нат. число]),)* [буква]([нат. число]))

variables = ([буква],)* [буква]

([терм] = [терм])+

[терм] ::= [переменная] | [константа]
| [конструктор](([терм],)*[терм])

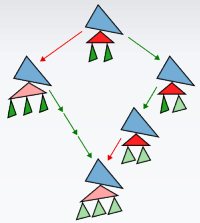
Множества имён переменных и конструкторов считаем непересекающимися.



Понятия алгоритма М.–М.

Мультиуравнение — это выражение вида $\{x_1, \ldots, x_n\} = (t_1, \ldots, t_m)$, где x_i — переменные, t_j — термы в выбранной сигнатуре (семантически означает, что все они равны друг другу).
Общая часть мультиуравнения — максимальное внешнее общее поддерево конструкторов t_i.
Граница мультиуравнения — множество мультиуравнений, подстановка которых в общую часть порождает термы t_i.

У мультиуравнения
$\{x_1, x_2\} = (f(g(x_3), h(x_4, g(x_5))), f(x_4, h(g(g(x_6)), x_7)))$
общая часть — это $f(x_4, h(x_4, x_7))$, граница — это
$\{\{x_4\} = (g(x_3), g(g(x_6))), \{x_7\} = g(x_5)\}$.



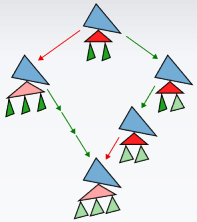
Описание алгоритма

Компактная форма системы мультиуравнений — такая, что для всех $S = T$, $S' = T'$, $S \cap S' = \varnothing$.

Строим исходную систему U: $\{x\} = (t_1, t_2), \{x_i\} = \varnothing$, где x — свежая переменная, x_i — переменные, входящие в термы t_1 и t_2.

1. Выбираем такое мультиуравнение $S = M$, что переменные из S не встречаются нигде больше в U. Если такого нет, объявляем о неудаче унификации.
2. Строим общую часть C и границу F. Если общей части нет, объявляем о неудаче унификации.
3. Делаем шаг редукции: заменяем $S = M$ на $\{S = C\} \cup F$, после чего приводим к компактной форме.
4. Перемещаем $S = C$ из U в результирующую систему T.

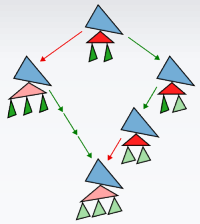
Если в U не остаётся мультиуравнений, то результат T — это искомая подстановка-унификатор t_1 и t_2.



Синтаксис задачи 2

nonterminals = ([буква],)* [буква]
terminals = ([буква],)* [буква]
(nonterminal -> (nonterminal | terminal)*)$^{+}$

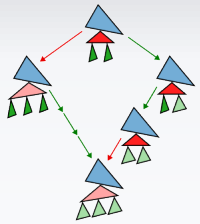
Множества имён терминалов и нетерминалов считаем непересекающимися.



Постановка задачи 2

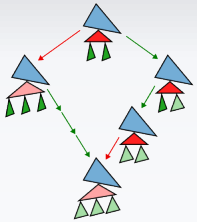
Необходимо построить упрощённую форму исходной грамматики, используя альфа-преобразование.
Например, грамматика
$S \to a\,S\,a \mid b \mid a\,T\,a$
$T \to a\,T\,a \mid a\,S\,a \mid b$
эквивалентна грамматике только с двумя первыми правилами (кстати, во входном потоке все правила будут записываться по отдельности, с новой строки).

При этом удобно пользоваться понятием терминальной формы правила — формы, учитывающей только расположение терминалов в правой части. Например, $S \to a\,\_\,a$.



Описание алгоритма

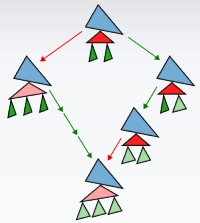
- Для каждого нетерминала N_i строим список терминальных форм правых частей α правил $N_i \to \alpha$.
- Все нетерминалы, у которых совпали множества терминальных форм, помещаем в один класс разбиения.
- Для каждых двух правил $N_i \to \alpha_1 N_i' \alpha_2$, $N_j \to \alpha_1 N_j' \alpha_2$, где α_1, α_2 — терминальные формы, проверяем, попадают ли N_i', N_j' в один и тот же класс разбиения. Если не попадают, то разделяем исходный класс разбиения N_i, N_j на классы согласно принадлежности нетерминалов в позиции нетерминала N_i' классам разбиения. Объявляем позицию T в правой части $\alpha_1 T \alpha_2$ проверенной. После чего объявляем все правила, содержащие нетерминалы исходного класса разбиения в правых частях, непроверенными.
- Продолжаем, пока все позиции нетерминалов во всех правилах не будут проверены.



Описание алгоритма

- ...
- Продолжаем, пока все позиции нетерминалов во всех правилах не будут проверены.

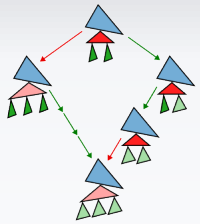
Для построения итоговой грамматики достаточно выбрать по одному представителю из каждого класса разбиения, и подставить соответствующие нетерминалы в терминальные формы правил.



Описание алгоритма

- ...
- Продолжаем, пока все позиции нетерминалов во всех правилах не будут проверены.

Если рассмотреть грамматику
$S \to a\,S\,a \mid b \mid a\,T\,a$
$T \to a\,C\,a \mid a\,S\,a \mid b$
$C \to a\,B\,a \mid b$
$B \to c$
то видно, что на первом этапе S, T, C будут отнесены к одному классу разбиения, а B — к другому. Проверка терминальной формы a _ a приведёт к тому, что C отделится в другой класс (а S и T на этом этапе ещё не будут разделены). После чего опять придётся проверять ту же терминальную форму, что приведёт к тому, что S и T также окажутся разделены.

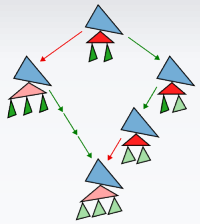


Синтаксис входных данных

Синтаксис записи входных данных для 3 задачи:

lexicographic | anti-lexicographic
constructors = ([буква]([нат. число]),)* [буква]([нат. число]))
variables = ([буква],)* [буква]
([терм] = [терм])+
[терм] ::= [переменная] | [константа]
| [конструктор](([терм],)*[терм])

Множества имён переменных и конструкторов считаем непересекающимися. Арность конструкторов полагаем равной либо 1, либо 2. Также считаем, что максимальная вложенность конструкторов в термах равна 3 (т.е. может быть, самое большее, три уровня вложенных скобок). Первая строка входного потока показывает, какой порядок должен проверяться: лексикографический или обратный ему (т.е. лексикографический для обращённых кортежей).



Порядок Кнута–Бендикса $>_{lo}$

$f(t_1, \ldots, t_n) >_{lo} g(u_1, \ldots, u_m)$ если и только если выполнено одно из условий:

1. $\exists i(1 \leqslant i \leqslant n \;\&\; t_i = g(u_1, \ldots, u_m))$;
2. $\exists i(1 \leqslant i \leqslant n \;\&\; t_i >_{lo} g(u_1, \ldots, u_m))$;
3. $(f > g) \;\&\; \forall i(1 \leqslant i \leqslant m \Rightarrow f(t_1, \ldots, t_n) >_{lo} u_i)$;
4. $(f = g) \;\&\; \forall i(1 \leqslant i \leqslant n \Rightarrow f(t_1, \ldots, t_n) >_{lo} u_i)$ и n-ка $(t_1, \ldots, t_n)$ лексикографически больше, чем $(u_1, \ldots, u_n)$ (т.е. первый её не совпадающий с u_i элемент t_i удовлетворяет условию $t_i >_{lo} u_i$).

Если используется обратный лексикографический порядок, то в последнем пункте сравниваются $(t_n, \ldots, t_1)$ и $(u_n, \ldots, u_1)$ (поскольку максимальная арность = 2, то это просто (t_2, t_1) и (u_2, u_1)).