**Санкт-Петербургский национальный исследовательский университет**

**информационных технологий, механики и оптики**

**Кафедра вычислительной техники**

Формальные языки и грамматики

Домашние задание №3

Приведение грамматик

Выполнил: Гхази Даниэль

Группа P3218

Преподаватель: Лаздин А.В.

2017 г.

**Задание**

Для каждого пункта задания (всего четыре) необходимо представить: исходную грамматику, пошаговый процесс построения множества производящих и достижимых символов грамматики, поэтапное формирование новых правил при удалении цепных и ε-правил, грамматику, полученную в результате преобразования (приведения).

* Преобразовать КС грамматику в эквивалентную грамматику, не содержащую бесполезных (непроизводящих и недостижимых) символов.
* Удалить из КС грамматики ε-правила.
* Преобразовать КС грамматику в эквивалентную, не содержащей цепных правил.
* Найти приведенную форму КС-грамматики

**Удаление бесплодных символов**

Исходная грамматика:

S → AB | CA

A → a

B → BC | AB

C → aB | b

Последовательно просматриваем правила первый раз для составления множества производящих символов:

A – первый символ, который можно отнести к производящим, потому что присутствует правило, в котором он порождает терминал.

C – второй символ, относящийся к производящим, по аналогичной причине, что и A.

На текущий момент множество производящих символов выглядит как {A, C}.

Последовательно просматриваем правила второй раз:

S – третий символ, он порождает цепочку CA, состоящую только из производящих символов.

Последовательно просматриваем правила третий раз и делаем вывод, что больше нельзя посчитать ни один символ производящим, так как в правой части каждого оставшегося правила имеется хотя бы один непроизводящий символ. Это говорит о том, что все производящие символы были выделены на предыдущем шаге.

Грамматика, из которой удалены непроизводящие символы, выглядит так:

S → CA

A → a

C → b

Последовательно просматриваем правила первый раз для составления множества достижимых символов:

Символ S считается достижимым изначально. В правой части первого правила содержатся оба оставшихся нетерминала (которые не являются символом S), поэтому ни одно правило не может быть исключено при выделении достижимых символов.

Конечная грамматика:

S → CA

A → a

C → b

**Удаление ε – правил**

Исходная грамматика

S → bA

A → bA | aB | ε

B → bB | ε

Посмотрим, какие преобразования можно провести:

На основе вышеприведенных преобразований зададим неукорачивающую грамматику.

Конечная грамматика:

S → A

A → bA | B | b

B → a | aC

C → bC | b

**Удаление цепных правил**

Исходная грамматика:

S → T+P | T

T → T\*P | P

P → C

C → /C | /

Для удаления цепных правил исполним алгоритм:

Разделим все правила на два множества, в первое из которых попадут правила вида A → B:

P1 = {S → T, T → P, P → C}

P2 = {S → T+P, T → T\*P, C → /C | /}

Поочередно будем рассматривать каждое правило из первого множества, и пытаться найти правила с таким же символом в левой части, что и у правила в первом множестве в правой части:

Правилу S → T соответствуют T → T\*P, T → P → C → /C | /, поэтому в грамматику добавятся следующие новые правила: S → T\*P, S → /C | /.

Правило P → C представляет из себя промежуточный шаг, который независимо от ситуации приведет к одному и тому же, поэтому это правило можно удалить, а вместо P во все другие правила следует поставить С.

Следовательно, имеем:

P1 = {S → T, T → С}

P2 = {S → T+С, T → T\*С, C → /C | /}

Правилу T → C соответствует C → /C | /, поэтому в грамматику добавятся следующее новое правило: T → /C | /. Но, так как, есть уже есть такое правило T → C, правило T → /C | / будет излишне, следовательно, его не следует добавлять в конечную грамматику.

Конечная грамматика

S → T+C | T | T\*C | /C | /

T → T\*C | C

C → /C | /

**Найдите приведенную форму КС-грамматики**

Исходная грамматика:

S → aB| bA | cC

A → cBS| bA | C | b | ε

B → bSA | cCb |S

C → Cd | aCa

1. Удаление непроизводящих символов

Последовательно просматриваем все правила в грамматике первый раз:

A – производящий символ, потому что порождает терминал b.

Последовательно просматриваем все правила в грамматике второй раз:

S – производящий символ, потому что порождает производящую цепочку bA,

B – производящий символ, потому что порождает производящий символ S.

Последовательно просматриваем все правила в грамматике третий раз и делаем вывод, что больше нельзя посчитать ни один символ производящим, так как в правой части каждого оставшегося правила имеется хотя бы один непроизводящий символ. Это говорит о том, что все производящие символы были выделены на предыдущем шаге.

Грамматика, из которой удалены непроизводящие символы, выглядит так:

S → aB| bA

A → bA | b | ε

B → bSA | S

2. Удаление недостижимых символов

Символ S достижим изначально, следовательно символы A и B тоже достижимы, следовательно, все символы грамматики достижимы.

3. Удаление ε – правил

Посмотрим, какие преобразования можно провести:

На основе вышеприведенных преобразований зададим неукорачивающую грамматику.

Конечная грамматика:

S → aB| bA | b

A → bA | b

B → bSA | S

4. Удаление цепных правил

Для удаления цепных правил исполним алгоритм:

Разделим все правила на два множества, в первое из которых попадут правила вида A → B:

P1 = {B → S}

P2 = {S → aB, S → bA, S → b, A → bA, A → b, B → bSA}

Поочередно будем рассматривать каждое правило из первого множества, и пытаться найти правила с таким же символом в левой части, что и у правила в первом множестве в правой части:

Правилу B → S соответствуют S → aB, S → bA, S → b, поэтому в грамматику добавятся следующие новые правила: B → aB, B → bA, B → b.

Конечная грамматика:

S → aB| bA | b

A → bA | b

B → bSA | aB | bA | b