К СВОЙСТВУ ПЕРЕВЕРНУТОЙ ПОЛЬСКОЙ ЗАПИСИ

Теодор Заркуа

Грузинский университет им. святого Андрея Первозванного
пр. Чавчаввадзе 53а, Тбилиси, 0109, Грузия, тел.: (995 99) 49-59-10 E-mail: temzar@gmail.com

Аннотация

Данная работа посвящена проблеме облегченияодновременного использования обратной (постфиксной) и прямой (префиксной) польских нотаций при обработке арифметических(алгебраических) выражений.

Автор предлагает понятие перевернутой польской записи, которая представляет некоторую модификацию классической польской записи, изучает свойства перевернутой обратной польской записи, показывет инвариантность сложности вычисления относительно преобразования переворачивания и формулирует задачу определения преобразования переворачивания обратной польской записи.

Доказывается теорема, которая позволяет в явном виде получить преобразование перевлрачивания обратной польской записи пострекдством инвертирования соответствующей прямой записи.

В дальнейшем автор показывает как можно естественным образом расширить понятие преобразования переворачивания на прямую польскую запись. После чего формулируется теорема для переворачивания прямой польской записи. В дальнейшем, на основе сформулированных теорем автор приводит обобщенную теорему, объединяющую утверждения первых двух теорем.

Общеизвестно какую важную роль в теории и практике программирования играют бесскобочные нотации записи выражений, предложенные польским математиком Яном Лукасевичем, которые принято называть польскими [1]. При этом, обратная (постфиксная) польская запись (ОПЗ) позволяет легко вычислить значение соответствующего выражения, так как содержит в себе всю информацию, необходимую для этого - операнды, операции и порядок выполнения операций. А прямая (префиксная) польская запись (ППЗ) удобна для осуществления функциональных преобразований выражения. Существуют задачи, где возникает необходимость использовать преимущества ОПЗ и ППЗ одновременно. Естественно, при этом существует необходимость преобразования ОПЗ в ППЗ и наоборот.

В данной работе предлагается подход, который практически снимет проблему преобразования ОПЗ в ППЗ и наоборот.

Вспомним алгоритм вычисления ОПЗ:

1. Взять пустой стек;
2. Взять очередной элемент выражения. Если это невозможно (выражение пусто), то перейти к пункту 5;
3. Если очередной элемент является операндом, то поместить его в стек и перейти к пункту 2;
4. Т.к. очередной элемент является обозначением операции, то считать из стека соответствующее количество записей (операндов); выполнить операцию, учитывая, что операнды считывались в обратном порядке (скажем, при делении, сначала делитель, а потом делимое); поместить результат в стек и перейти к пункту 2;
5. Взять результат из стека, завершить выполнение алгоритма.

Обозначим приведенный алгоритм символом А. Перевернутым к А назовем алгоритм, получающийся из А заменой текста "в обратном порядке" в пункте 4 на текст "в прямом порядке" и обозначим его ТА.

Определение:

Будем называть преобразованием переворачивания заданного выражения V в формате ОПЗ (в дальнейшем будем писать V € ОПЗ) такое преобразование Т, что ТА(Т(V))≡A(V).

Например, $T(ab*) \equiv ba*$

$T(de+f-) \equiv fed+-$

Здесь латинскими буквами обозначены операнды выражения. Здесь же зафиксируем очевидную идентичность сложностей алгоритмов А и ТА, соответственно идентичность сложностей вычисления ОПЗ и перевернутого ОПЗ.

Обозначим через I преобразование инвертирования выражения (когда выражение переписывается в обратном порядке), а через Р - преобразование выражения из ОПЗ в ППЗ. Тогда имеет место следующее утверждение.

ТЕОРЕМА 1. Для любого V € ОПЗ справедлива формула:

$$T(V) \equiv I(P(V)) .$$

Доказательство

Поведем доказательство по индукции относительно количества операндов n в выражении V. Для n=1 утверждение верно. Действительно, пусть @ - некоторая двуместная операция, тогда T(ab@) ≡ ba@ ≡ I(@ab) ≡ I(P(ab@)). Легко убедиться, что при n=1 теорема верна для операций с любым количеством операндов.

Допустим, что доказываемое утверждение верно для всех V € ОПЗ таких, что n < k, где k > 2. Покажем, что в таком случае оно верно для любых V € ОПЗ таких, что n = k. Пусть имеем некоторое V € ОПЗ такое, что n = k и пусть @ - обозначает операцию, расположенную в самом конце выражения V (любое выражение вида ОПЗ, содержащее хотя бы одну операцию, заканчивается обозначением операции). Пусть, для определенности, @ - обозначает двуместную операцию. Тогда можем записать V ≡ UW@ и P(V) ≡ @P(U)P(W), где U и W выражения вида ОПЗ, в сумме содержащие ровно k-1 операций. Соответственно и U и W удовлетворяют нашему допущению. Поэтому можем записать цепочку очевидных преобразований:

T(V) ≡ T(UW@) ≡ T(W)T(U)@ ≡ I(P(W))I(P(U))@ ≡ I(@P(U)P(W)) ≡ I(P(V)), что и требовалось доказать. Совершенно очевидно, что приведенные рассуждения сохраняют силу и в том случае, когда операция @ имеет количество операндов, отличное от двух. Этим данная теорема доказана полностью.

В дальнейшем уже не составляет труда расширить понятие переворачиваемости таким образом, чтобы его можно было применить к ППЗ.

Напрмер: T(+ab) ≡ +ba

T(-+def) ≡ -f+ed .

И тогда можно будет убедиться в справедливости следующего утверждения.

ТЕОРЕМА 2. Для любого V € ППЗ справедлива формула:

$$T(V) \equiv I(P^{-1}(V)) .$$

Если через P обозначить расширенное преобразование (т.е. для аргумента ОПЗ - это преобразование в ППЗ, а для аргумента ППЗ - это преобразование в ОПЗ), а аббревиатуру ПЗ использовать для обоих видов польских записей, тогда приведенные две теоремы можно объединить.

ТЕОРЕМА 3. Для любого V € ПЗ справедлива формула:

$$T(V) \equiv I(P(V)) .$$

Таким образом понятие перевернутой польской записи позволяет легко переходить от ОПЗ к ППЗ и обратно, соответственно облегчая реализацию задач, в которых по ходу выполнения возикает необходимость в использовании преимуществ обеих бесскобочных форм записей.

Данный подход облегчит расширение алгоритмических языков программирования, путем включения в них средств для обработки функциональных выражений, задаваемых с помощью представления, основанного на польской нотации. Естественно, языки выиграют, если в них будут включены средства, позволяющие получить удобное представление функциональных выражений из исходного вида, заданного строкой. Одна из программных реализаций подобного подхода в свое время была выполнена автором на базе языка паскаль. Объектно ориентированные языки позволяют существенно облегчить использование подобных программных реализаций конечным пользователем.

Литература:

[1] Грис Д. Конструирование компиляторов для цифровых вычислительных машин. - Москва, "Мир", 1975.