Генерация условий корректности предикатных программ с взаимной рекурсией

М. С. Чушкин
Новосибирский государственный университет

В настоящей работе описывается метод дедуктивной верификации предикатных программ для программ с произвольной рекурсии. Разработана система правил вывода условий корректности для различных видов операторов предикатных программ.

1 Введение

Дедуктивная верификация реализует проверку правильности программы относительно ее спецификации, записанной на формальном языке. Условия корректности программы строятся путем применения правил вывода к формулам, состоящим из логики операторов и спецификации программы. Доказательство условий корректности проводится с помощью некоторой системы автоматического доказательства. Ее применение, в отличие от доказательства «вручную», гарантирует правильность программы относительно спецификации.

2 Логика оператора

Программа на языке P – совокупность определений предикатов. Каждый предикат — вычислимая логическая формула, представленная в виде оператора. Определение предиката имеет вид:

$$\mathrm{A(x:\ y)}\ \underline{\mathbf{pre}}\ \mathrm{P(x)}\ \{\ \mathrm{S(x:\ y)}\ \}\ \underline{\mathbf{post}}\ \mathrm{Q(x,\ y)}$$

где A — это имя определяемого предиката, S — оператор, а x и y — это непересекающиеся наборы переменных, аргументы и результаты соответственно.

Спецификация предиката A(x: y) задается двумя логическими формулами: предусловием P(x), ограничивающим область определения функции, реализуемой программой, и постусловием Q(x, y), связывающим значения аргументов и результатов. Спецификацию будем записывать в виде [P(x), Q(x, y)].

Логика оператора S(x: y) — сильнейший предикат L(S(x: y)), истинный при завершении исполнения оператора S(x: y) [4].

Определим логику для базисных операторов. В предположении, что выражение E, зависящее от x, не содержит переменную a, логика оператора присваивания a := E есть

$$L(a\ :=\ E(x))\ \equiv\ R(x)\ \wedge\ a = E(x)$$

где R(x) — слабейший предикат, при истинности которого определено выражение E. Например, $L(c\ :=\ a/b)\ =\ b \neq 0\ \wedge\ c = a/b$.

Построим логику L(B; C) оператора B; C, определяющего последовательное исполнение операторов B и C, через логики L(B) и L(C). Оказывается, необходимо отдельно рассматривать случай, когда вычисление оператора C от B не зависит. Для фиксации связей между

операторами произвольный оператор A будем изображать в виде A(x: y), где наборы переменных x и y обозначают аргументы и результаты оператора, соответственно.

Вместо оператора B; C далее будем рассматривать оператор суперпозиции B(x: z); C(z: y) и параллельный оператор B(x: y) || C(x: z). Третьим рассматривается условный оператор **if** (E) B(x: y) **else** C(x: y), где логическое выражение E может зависеть от x. Предполагается, что наборы x, y и z не пересекаются, а набор x может быть пустым. Определим логики указанных операторов:

$$L(B(x:\ z); C(z:\ y)) \ \equiv\ \exists z\ L(B(x:\ z))\ \wedge\ L(C(z:\ y))$$
$$L(B(x:\ y)||C(x:\ z))\ \equiv\ L(B(x:\ y))\ \wedge\ L(C(x:\ z))$$
$$L(\textbf{if}\ (E)\ B(x:\ y)\ \textbf{else}\ C(x:\ y))\ \equiv\ (E\ \Rightarrow\ L(B(x:\ y)))\ \wedge\ (\neg E\ \Rightarrow\ L(C(x:\ y)))$$

Логика программы должна быть согласована с операционной семантикой языка P: для произвольного оператора B(x: y) и фиксированных значений переменных наборов x и y его логика L(B(x: y)) истинна тогда и только тогда, когда любое исполнение оператора B(x: y) на наборе значений x завершается, причем результатами исполнения являются значения набора y. Отметим, что условием завершения оператора B(x: y) является формула y L(B(x: y)).

3 Корректность программы

Допустим, программе в целом соответствует главный предикат A(x: y) со спецификацией [P(x), Q(x, y)], оператором которого является S(x: y). Тогда корректность программы определяется следующими условиями:

1. постусловие Q(x, y) должно быть истинным после исполнения оператора S(x: y);
2. исполнение оператора S(x: y) всегда завершается.

Утверждение L(S(x: y)) становится посылкой при доказательстве первого условия корректности, поскольку после исполнения оператора S(x: y) оно становится истинным. Условие завершения исполнения оператора S(x: y) определяется утверждением: $\exists y\ L(S(x:\ y))$. Кроме того, предусловие P(x) необходимо использовать в качестве посылки в обоих условиях корректности, определяемых ниже следующими формулами:

$$P(x)\ \wedge\ L(S(x:\ y))\ \Rightarrow\ Q(x,\ y)\ P(x)\ \Rightarrow\ \exists y\ L(S(x:\ y))$$

Первое утверждение называется условием частичной корректности, а второе — условием завершения программы. Их конъюнкция определяет условие тотальной (или полной) корректности оператора S(x: y) относительно спецификации [P(x), Q(x, y)]:

$$Corr(S,\ P,\ Q)(x)\ \equiv\ P(x)\ \Rightarrow\ (\forall y\ (L(S(x:\ y))\ \Rightarrow\ Q(x,\ y)))\ \wedge\ \exists y\ L(S(x:\ y))$$

Далее термин «корректность» будем использовать в смысле тотальной корректности.

4 Корректность рекурсивной программы

4.1 Корректность рекурсивного предиката

Допустим, имеется определение рекурсивного предиката A:

```
predicate A(X x: Y y)
pre P(x)
{
    S(x: y)
}
post Q(x, y)
measure m(x);
```

Внутри оператора S(x: y) имеется рекурсивный вызов предиката A.

Используется следующая схема доказательства по индукции для некоторого произвольного утверждения W(x):

$$\forall t \in X\ [(\forall u \in X\ m(u) < m(t)\ \Rightarrow\ W(u))\ \Rightarrow\ W(t)]\ \Rightarrow\ \forall x \in X\ W(x)$$

Функция m, называемая мерой, отображает X во множество натуральных чисел со стандартным отношением порядка $<$.

В соответствии со схемой индукции корректность рекурсивного предиката может быть определена следующей формулой:

$$\forall t\ (\forall u\ (m(u) < m(t)\ \Rightarrow\ Corr(A,\ P,\ Q)(u))\ \Rightarrow\ Corr(A,\ P,\ Q)(t)$$

Введем формулу, которая будет обозначать индуктивное предположение для набора аргументов t:

$$Induct(A,\ P,\ Q)(t)\ \equiv\ \forall u\ (m(u) < m(t)\ \Rightarrow\ Corr(A,\ P,\ Q)(u))$$

В итоге, формула тотальной корректности рекурсивного предиката A примет следующий вид:

$$\forall t\ Induct(A,\ P,\ Q)(t)\ \Rightarrow\ Corr(A,\ P,\ Q)(t) \tag{1}$$

4.2 Корректность рекурсивного кольца предикатов

Рассмотрим программу языка P, представленную набором определений предикатов:

```
predicate A_i(X_i x_i : Y_i y_i)
pre P_i(x_i)
{
    S_i(x_i : y_i)
}
post Q_i(x_i, y_i)
measure m_i(x_i);
```

Причем i = 1, ..., n. P_i и Q_i — это пред- и пост- условия предиката A_i, m_i — функция меры, заданная на аргументах предиката A_i, а $S_i(x_i :\ y_i)$ — оператор, в теле которого могут встречаться вызовы предикатов A_1, ..., A_n.

Применяя N раз схему индукции с разными индуктивными предположениями ко всему рекурсивному кольцу, получаем набор формул, определяющих корректность этой программы:

$$\forall t_1, ..., t_N\ Induct(A_1,\ P_1,\ Q_1)(t_1)\ \wedge ... \wedge\ Induct(A_N,\ P_N,\ Q_N)(t_N) \Rightarrow$$
$$Corr(A_1,\ P_1,\ Q_1)(t_1)\ \wedge ... \wedge\ Corr(A_N,\ P_N,\ Q_N)(t_N)$$

, где i = 1, ..., N.

Преобразуем индуктивные предположения:

$$Induct(A_k,\ P_k,\ Q_k)(t_k)\ \equiv\ F(t'_k,\ t_k)\ \wedge\ m(t'_k) < m(tk)\ \Rightarrow\ Corr(A_k,\ P_k,\ Q_k)(t'_k)$$

, где F — это некоторая формула, связывающая переменные t'_k и t_k. В данном случае t_k – это формальные параметры предиката A_k, а t'_k – его всевозможные фактические параметры.

4.3 Построение связующих формул

Опишем алгоритм получения связующей формулы для предиката A_k. Цель алгоритма - построить множество формул вида:

$$F_r : x_r \to x'_k$$

смысл каждой из которых — выразить произвольные входные аргументы x'_k предиката A_k через формальные параметры x_r предиката A_r, принадлежащего рекурсивному кольцу.

Рассмотрим некоторый предикат A_i, учавствующий в рекурсивном кольце. Пусть в теле этого предиката содержится вызов пердиката A_k:

predicate $A_i(X_i\ x_i :\ Y_i\ y_i)$ {
    $S_{pre}(x_i :\ \mathrm{z})$;
    $A_k((x_i ,\ \mathrm{z}) :\ \ldots)$;
    ...
}

Тогда, связующая формула F_i для этого предиката будет выглядеть следующим образом:

$$F_i(x_i : x'_k)\ \equiv\ L(S_{pre}(x_i :\ z))\ \wedge\ x'_k = (x_i,\ z)$$

Первый конъюнкт - это аппроксимация логик вблизи оператора вызова. Он необходим для того, чтобы выразить переменную z через формальные параметра предиката A_i. Следующий конъюнкт фиксирует входные параметры x'_k предиката A_k.

Пусть теперь в теле предиката A_i встречается вызов произвольного предиката A_j:

predicate $A_i(X_i\ x_i :\ Y_i\ y_i)$ {
    $S_{pre}(x_i :\ \mathrm{z})$;
    $A_j((x_i ,\ \mathrm{z}) :\ \ldots)$;
    ...
}

В таком случае, связующая формула F_i примет следующий вид:

$$F_i(x_i : x'_k)\ \equiv\ L(S_{pre}(x_i :\ z))\ \wedge\ x'_j = (x_i,\ z)\ \wedge\ F_j(x_j :\ x'_k)$$

Нетрудно заметить, что в формуле появился еще один конъюнкт - вызов формулы, выражающей x'_k через формальные параметры предиката A_j. Если эта формула еще не была построена, то в дальнейшем ее необходимо построить аналогичным образом.

И, наконец, пусть в теле предиката A_i встречается несколько вызовов, разделенных условным оператором:

```
predicate A_i(X_i x_i: Y_i y_i) {
    if (e) {
        S_1(x_i: z);
        A_{j_1}((x_i, z): ...);
        ...
    } else {
        S_2(x_i: z);
        A_{j_2}((x_i, z): ...);
        ...
    }
    ...
}
```

В таком случае, разные "ветки"вызовов разделяются знаком дизъюнкции. В итоге, связующая формула F_i примет следующий вид:

$F_i(x_i : x'_k) \equiv [\, L(S_1(x_i : z)) \wedge x'_{j_1} = (x_i,\ z) \wedge F_{j_1}(x'_{j_1} : x'_k)\,] \vee [\, L(S_2(x_i : z)) \wedge x'_{j_2} = (x_i,\ z) \wedge F_l(x'_{j_2} : x'_k)\,]$

Построение множетсва формул следует начинать с формулы F_k и, соответственно, с тела предиката A_k. Результатом описанного алгоритма является формула F_k, которая используется в дальнейшем, при построении индуктивных посылок.

4.4 Корректность оператора, содержащего рекурсивный вызов

Допустим, оператор B находится в теле предиката A_k, принадлежащего рекурсивному кольцу. Поскольку внутри оператора B может находиться рекурсивный вызов предиката A_i, корректность оператора B определяется формулой:

$$Corr(B,\ P_B,\ Q_B)(t_i,\ x) \equiv \forall t_1\ Induct(A_1,\ P_1,\ Q_1)(t_1) \wedge ... \wedge \forall t_N\ Induct(A_N,\ P_N,\ Q_N)(t_N) \Rightarrow Corr(B,\ P_B,\ Q_B)(x)$$

Для рекурсивного вызова проверку по мере m(u) < m(t) присоединим к предусловию. Введем обозначение:

$$P^*(x) \equiv m(x) < m(t) \wedge P(x)$$

Здесь t – формальные параметры рекурсивного определения предиката, а x – фактические параметры рекурсивного вызова. В случае нерекурсивного вызова запись $P^*(x)$ эквивалентна P(x).

В итоге, доказательство корректности рекурсивного определения предиката A_i представляется следующей формулой:

$$Corr(K_i,\ P_i,\ Q_i)(x_i,\ x_i)$$

5 Система правил вывода условий корректности

Используя формулу тотальной корректности можно автоматически построить формулу корректности оператора $S(x : y)$ при условии, что для языка программирования построена логика программы. Итоговая формула корректности будет длинной и сложной даже для коротких программ; она будет длиннее программы $S(x : y)$. Специализация формулы тотальной корректности для разных видов операторов позволяет декомпозировать длинную формулу корректности к нескольким более коротким и простым формулам.

В данном разделе определяется система правил вывода условий корректности для различных операторов. Однако будут опущены доказательства правил, приведенные в работах [4, 5, 6].

Допустим, операторы $B(x:\ z)$ и $C(z:\ y)$ корректны относительно своих спецификаций $[P_B,\ Q_B]$ и $[P_C,\ Q_C]$. Тогда допустимы следующие правила:

$$\textbf{RP:}\ \frac{\begin{array}{c} Corr(A,\ P_A,\ Q_A,\ B,\ P_B,\ Q_B)(t,\ x);\qquad Corr(A,\ P_A,\ Q_A,\ C,\ P_C,\ Q_C)(t,\ x);\\ P(x)\ \rightarrow\ P_B^*(x)\ \wedge\ P_C^*(x);\qquad \forall y,z\ (Q_B(x,\ y)\ \wedge\ Q_C(x,\ z)\ \rightarrow\ Q(x,\ y,\ z)) \end{array}}{Corr(A,\ P_A,\ Q_A,\ B(x:\ y)||C(x:\ z),\ P,\ Q)(t,\ x)}$$

$$\textbf{RS:}\ \frac{\begin{array}{c} Corr(A,\ P_A,\ Q_A,\ B,\ P_B,\ Q_B)(t,\ x);\\ \forall z\ Corr(A,\ P_A,\ Q_A,\ C,\ P_C,\ Q_C)(t,\ (x,\ z));\\ P(x)\ \rightarrow\ P_B^*(x);\qquad \forall z,v\ (P(x)\ \wedge\ Q_B(x,\ z,\ v)\rightarrow P_C^*(x,\ z));\\ \forall z,v,y\ (P(x)\ \wedge\ Q_B(x,\ z,\ v)\ \wedge\ Q_C(x,\ z,\ y)\ \rightarrow\ Q(x,\ y)) \end{array}}{Corr(A,\ P_A,\ Q_A,\ B(x:\ z,\ v);C(x,\ z:\ y),\ P,\ Q)(t,\ x)}$$

$$\textbf{RC:}\ \frac{\begin{array}{c} Corr(A,\ P_A,\ Q_A,\ B,\ P_B,\ Q_B)(t,\ x);\\ Corr(A,\ P_A,\ Q_A,\ C,\ P_C,\ Q_C)(t,\ x);\\ P(x)\ \wedge\ E\ \rightarrow\ P_B^*(x);\qquad P(x)\ \wedge\ \neg E\ \rightarrow\ P_C^*(x);\\ \forall y\ (P(x)\ \wedge\ E\ \wedge\ Q_B(x,\ y)\ \rightarrow\ Q(x,\ y));\\ \forall y\ (P(x)\ \wedge\ \neg E\ Q_C(x,\ y)\ \rightarrow\ Q(x,y)) \end{array}}{Corr(A,\ P_A,\ Q_A,\ \textbf{if}\ (E(x))\ B(x:\ y)\ \textbf{else}\ C(x:\ y),\ P,\ Q)(t,\ x)}$$

Допустим, необходимо доказать тотальную корректность оператора вызова $C(B(x):\ y)$, где C — вызываемый предикат, а $B(x)$ — набор аргументов в виде выражений, не содержащих других вызовов. Для этого необходимо воспользоваться следующим правилом:

$$\textbf{RB:}\ \frac{\begin{array}{c} \forall z\ Corr(A,\ P_A,\ Q_A,\ C,\ P_C,\ Q_C)(t,\ z);\\ P(x)\ \rightarrow\ P_B(x)\ \wedge\ P_C^*(B(x));\\ \forall y\ (P(x)\ \wedge\ Q_C(B(x),\ y)\ \rightarrow\ Q(x,\ y));\\ \forall x,z1,z2\ P_B(x)\ \wedge\ L(B(x:\ z1))\ \wedge\ L(B(x:\ z2))\ \rightarrow\ z1=z2; \end{array}}{Corr(A,\ P_A,\ Q_A,\ C(B(x):\ y),\ P,\ Q)(t,\ x)}$$

В случае если вызов $C(B(x):\ y)$ рекурсивен (C совпадает с A), то первая посылка правила совпадает с индуктивным предположением и поэтому отсутствует в правиле. Отметим, что для всех предыдущих и последующих правил действует аналогичное соглашение: если посылка правила имеет вид: $Corr(A,\ P_A,\ Q_A,\ A,\ P_A,\ Q_A)$, то она отсутствует в правиле.

Не всегда известны спецификации для подоператоров. В этом случае следует применять другие правила:

$$\textbf{QP:}\ \frac{Corr(A,\ P_A,\ Q_A,\ B,\ P,\ Q_B)(t,\ x); \qquad Corr(A,\ P_A,\ Q_A,\ C,\ P,\ Q_C)(t,\ x);}{Corr(A,\ P_A,\ Q_A,\ B(x:\ y)||C(x:\ z),\ P,\ Q_B\ \wedge\ Q_C)(t,\ x)}$$

$$\textbf{QS:}\ \frac{\begin{array}{c}P(x)\ \rightarrow\ \exists z, v\ L(B(x:\ z,\ v));\\ \forall z\ Corr(A,\ P_A,\ Q_A,\ C,\ \lambda x, z.\ (P(x)\ \wedge\ \exists z, v\ L(B(x:\ z,\ v))),\ Q)(t,\ (x,\ z))\end{array}}{Corr(A,\ P_A,\ Q_A,\ B(x:\ z,\ v); C(x,\ z:\ y),\ P,\ Q)(t,\ x)}$$

$$\textbf{QSB:}\ \frac{\begin{array}{c}Corr(A,\ P_A,\ Q_A,\ B,\ P_B,\ Q_B)(t,\ x); \qquad P(x)\ \rightarrow\ P^*_B(x);\\ \forall z\ Corr(A,\ P_A,\ Q_A,\ C,\ \lambda x, z.\ (P(x)\ \wedge\ Q_B(x,\ z)),\ Q)(t,\ (x,\ z))\end{array}}{Corr(A,\ P_A,\ Q_A,\ B(x:\ z,\ v); C(x,\ z:\ y),\ P,\ Q)(t,\ x)}$$

$$\textbf{QC:}\ \frac{\begin{array}{c}Corr(A,\ P_A,\ Q_A,\ B,\ \lambda x.\ P(x)\ \wedge\ E(x),\ Q)(t,\ x);\\ Corr(A,\ P_A,\ Q_A,\ C,\ \lambda x.\ P(x)\ \wedge\ \neg E(x),\ Q)(t,\ x)\end{array}}{Corr(A,\ P_A,\ Q_A,\ \textbf{if}\ (E(x))\ B(x:\ y)\ \textbf{else}\ C(x:\ y),\ P,\ Q)(t,\ x)}$$

6 Алгоритм генерации условий корректности

Задачей алгоритма является автоматическое построение условий корректности предикатной программы. На языке P предикатная программа представлена набором определений предикатов. Корректность подобной программы — это корректность каждого определенного предиката. Поэтому задача сводится к построению условий корректности для определения предиката:

A(x: y) **pre** P(x) { S(x: y) } **post** Q(x, y)

Доказательство формулы тотальной корректности предиката:

$$Corr(A,\ P,\ Q,\ S,\ P_S,\ Q_S)(x,\ x)$$

реализуется построением дерева вывода с применением правил вывода, описанных в предыдущем разделе. Генерируемыми условиями корректности являются листья этого дерева.

В процессе вывода могут встретиться два вида формул: это условие тотальной корректности оператора

$$Corr(S,\ P,\ Q)(x)$$

и логическая формула

$$A_1 \wedge A_2 \wedge ... \wedge A_n \Rightarrow B$$

где в качестве A_i и B могут выступать как простые логические утверждения, так и логики операторов L(S(x: y)).

В том случае, если формула имеет первый вид, применяется либо система правил для общего случая (Q), либо система правил для случая корректных подоператоров (R). При этом система правил Q более приоритетна, т.е. ее применение осуществляется в первую очередь.

Если же формула имеет второй вид, то необходимо проверить, содержит ли она в себе логики операторов L(S(x: y)). Если не содержит, то это заключительная формула (одно из условий корректности), и она оформляется в виде леммы. Но если в ней присутствует

логика, то вывод этого утверждения необходимо продолжить за счет системы правил декомпозиции L(S(x: y)) (F). Дальнейшая цель вывода — это исключение из формулы логики оператора.

Корректность алгоритма генерации условий корректности – это допустимость каждого применяемого правила. Допустимость правил доказана в работах [4, 5, 6] и подтверждена также автоматическим доказательством в системе PVS [7]. Проверка корректности реализации алгоритма обеспечивается тестированием. На данный момент корректность реализации алгоритма проверена на двух десятках тестов.

7 Заключение

В работе описан подход, позволяющий доказывать тотальную корректность предикатных программ. Построена система правил вывода условий корректности. Ранее правила, входящие в эту систему, существовали разрозненно и были описаны в разных работах, а порой и под разными именами. Некоторые правила нуждались в обобщении на рекурсивный случай.

На основании данной системы правил был разработан и реализован генератор формул корректности в системе предикатного программирования. Генератор позволяет автоматически формировать условия корректности для программ с исходным кодом на языке P. В рамках генератора также было реализовано последующее упрощение полученных условий корректности с использованием простейших законов логики и булевой алгебры. Генератор является частью системы предикатного программирования и может быть вызван автоматически.

Дальнейшие планы. Необходимо разработать метод доказательства корректности предикатов с переменными предикатных типов в качестве параметров; разработать правила для доказательства корректности гиперфункций.

8 Литература

1. *Кулямин В.В* Методы верификации программного обеспечения // Институт системного программирования РАН. - 2008.

2. *Карнаухов Н.С., Першин Д.Ю., Шелехов В.И.* Язык предикатного программирования P. Новосибирск, 2010. 42с. (Препр. / ИСИ СО РАН; N 153).

3. *Ершов Ю.Л., Палютин Е.А.* Математическая логика: Учебное пособие для вузов – 2-е изд., испр. и доп. – М.: Наука. Гл. ред. физ.-мат. лит., 1987. – 336 с.

4. *Шелехов В.И.* Методы доказательства корректности программ с хорошей логикой // Межд. конф. "Современные проблемы математики, информатики и биоинформатики посвященная 100-летию со дня рождения А.А. Ляпунова. 2011. 17с.

5. *Шелехов В.И.* Предиктное программирование. Учебное пособие / НГУ. Новосибирск, 2009. 111 С.

6. *Шелехов В.И.* Предиктное программирование. Лекции / ИСИ СО РАН. Новосибирск, 2011.

7. *http* : *//www.iis.nsk.su/persons/vshel/files/rules.zip*