Захаров В. А.

О разрешимости проблемы эквивалентности в одном классе операторных программ.

Стандартные операторные схемы программ были введены в [1] в качестве удобной модели вычислительных программ, выполненных в традиционных императивных системах программирования типа ALGOL, PASCAL, C. В соответствии с парадигмой императивного программирования указанные схемы компонуются из операторов ввода-вывода, операторов присваивания вида **x:=t**, где **x** - переменная, а **t** - терм, а также из тестов вида $\mathbf{p(t_1,...,t_m)}$, где **p** - предикатный символ, а $\mathbf{t_1,...,t_m}$ - термы. К достоинствам этой модели можно отнести простоту синтаксиса, строгую формальность семантики, а также широкие возможности ее использования для решения задач семантического анализа, верификации и оптимизации вычислительных программ.

Во всех перечисленных задачах основное внимание уделяется семантическим (или функциональным) свойствам программ, т.е. свойствам, зависящим исключительно от функций, вычисляемых программами. Поэтому одной из центральных задач теории схем программ является проблема эквивалентности. Две схемы программ считаются эквивалентными, если для любой интерпретации их базовых элементов (функциональных символов и предикатов) обе схемы осуществляют одно и то же преобразование входных данных (начальных значений переменных) в выходные данные. В [1,2] было установлено, что проблема эквивалентности стандартных схем программ в общем случае неразрешима. Однако, налагая определенные ограничения на структуру схем программ (см. [3--8]), удалось выделить обширные классы стандартных схем программ, в которых проблема эквивалентности разрешима. Характерной особенностью таких схем является консервативность порождаемых ими вычислений. Говоря неформально, свойство консервативности означает, что значение переменной, вычисленное на промежуточных этапах, не утрачивается безвозвратно в ходе последующих преобразований и оказывает непосредственное влияние на окончательный результат вычисления.

В настоящей работе нами впервые выделен представительный класс неконсервативных стандартных схем программ с разрешимой проблемой эквивалентности. Мы определим стандартные схемы программ в терминах конечных систем переходов, помеченных подстановками и атомами, и выделим класс схем *OrtSP*, подстановки которых обладают свойством ортогональности. Далее будет показано, что проблема эквивалентности ортогональных стандартных схем программ разрешима. Обоснование разрешимости проблемы эквивалентности и построение разрешающих алгоритмов проводится на основе методов, разработанных в [9-11].

Обозначим X, F, R соответственно алфавиты переменных, функциональных символов и предикатов. Не ограничивая общности, будем полагать, что алфавит переменных $X=\{x_1,...,x_N\}$ конечный. Множество термов $Term_X$ над алвавитами X и F определяется традиционным образом. Если терм t является подтермом терма s, мы будем говорить, что t *содержится* в s. Если один из термов t и s содержится в другом, то эти термы будем называть *совместимыми*. В противном случае они будут называться *ортогональными*. Терм, не содержащий переменных, назовем *основным*. *Высота* $h(t)$ терма t определяется следующим образом: $h(t)=0$, если терм t - переменная из X, и $h(t)=\max(h(t_1),...,h(t_k))+1$, если $t=f(t_1,...,t_k)$. Размер $|t|$ терма t будем полагать равным совокупному числу вхождений переменных и функциональных символов в t. *Подстановкой* на множестве переменных X называется всякое отображение η: $X \to Term_X$. Множество всевозможных подстановок на X обозначим $Subst^X_X$. Обычно подстановка η представляется системой *связок* $\{x_1/t_1,...,x_N/t_N\}$, где $t_i= \eta(x_i)$. Применительно к стандартным схемам программ связка x_i/t_i соответствует оператору присваивания $x_i := t_i$, а сама подстановка η может мыслиться как параллельная композиция операторов присваивания.

Терм t_i, связывающий переменную x_i в подстановке, будем обозначать $\eta[i]$, а множество всех термов, связывающих переменные $x_1,...,x_N$ в подстановке η обозначим $Term(\eta)$. Высотой $h(\eta)$ и размером $|\eta|$ подстановки η назовем величины $\max(h(\eta[1]),...,h(\eta[N]))$ и $\max(|\eta[1]|,...,|\eta[N]|)$ Соответственно. Символом ε будем обозначать *пустую подстановку* $\{x_1/x_1,...,x_N/x_N\}$. Для композиции подстановок η и θ будем использовать запись $\eta\theta$. Будем говорить, что терм t *совместим* с подстановкой η, если t совместим в одном из термов $\eta[i]$, $1 \le i \le N$;

подстановка η является *переименованием*, если $\{\eta[1],..., \eta[N]\}=X$;

подстановка η *ортогональная*, если любая пара термов $\eta[i]$, $\eta[j]$, $1 \le i < j \le N$, является ортогональной;

- полстановка η *опережает* подстановку θ (обозначим этот факт $\theta \le \eta$), если η является комопозицией $\lambda\theta$, где λ– подстановка из $Subst^X_X$.

Множество $Subst^X_X$ с операцией композиции может быть представлено как моноид, в котором роль нейтрального элемента играет пустая подстановка. Нетрудно заметить, что ортогональные подстановки в свою очередь образуют подмоноид $Subst^X_X$.

Формула вида $p(x'_1,...,x'_k\})$, где p - k-местный предикатный символ, а $x'_1,...,x'_k$ - переменные из X, называется *атомом*. *Литерой*, базирующейся на атоме A, будем называть либо сам атом A, либо его отрицание $\neg A$. Рассмотрим некоторый конечный набор атомов $\mathcal{A}=\{A_1,...,A_M\}$. Тогда всякий набор литер $C=\langle L_1,...,L_M\rangle$, базирующихся на атомах $A_1,...,A_M$, будем называть *тестом*. Обозначим $\mathcal{C}_{\mathcal{A}}$ совокупность всевозможных тестов, базирующихся на множестве атомов $\mathcal{A}$. Очевидно, $|\mathcal{C}_{\mathcal{A}}|=2^M$. Тесты соответствуют элементарным конъюнкциям логических условий, используемых в условных операторах **if-then-else** и операторах итерации **while-do**.

Термальная интерпретация I над алфавитами X, F, R (см. [12]) определяется

- предметной областью, каковой является множество термов $Term_X$;
- соответствием, которое приписывает каждому k-местному функциональному символу f из F отображение $(Term_X)^k \to Term_X$ такое, что всякий набор термов $t_1,...,t_k$ преобразуется в терм $f(t_1,...,t_k)$;
- соответствием, которое сопоставляет каждому k-местному предикатному символу p из R множества p^I, состоящие из k-местных наборов термов, на которых этот предикат принимает значение "истина".

Для заданной термальной интерпретации I, атома $A=p(x'_1,...,x'_k)$ и подстановки η из $\mathrm{Subst}^X{}_X$ мы будем говорить, что атом A (литера $\neg$A) *выполняется на* I *для* η, если набор $x'_1\eta,\dots,x'_k\eta$ содержится (соответственно не содержится)в множестве p^I.

Пусть заданы символьные алфавиты X, F, R, и конечное множество атомов *A*. *Стандартной операторной программой* над X, F, *A* называется конечная помеченная система переходов, представленная пятеркой π=<V, **вход**, **выход**, **тупик**, S,T>, где

- V - конечное множество *вершин программы*;
- **вход, выход, тупик** - выделенные вершины *входа*, *выхода* и *пустого цикла*;
- S: V $\to$ $\mathrm{Subst}^X{}_X$ всюду определенная *функция означивания*, сопоставляющая каждой вершине из V подстановку из $\mathrm{Subst}^X{}_X$;
- T: (V – {**выход**}) $\times$ C_A $\to$ (V – {**вход**}) всюду определенная *функция переходов*, удовлетворяющая условию T(**тупик**,C)=**тупик** для всякого теста C из C_A.

Вершины программы, отличные от входа, выхода и пустого цикла, будем называть *внутренними вершинами*. *Размером* $|\pi|$ и *высотой* h(π) программы π мы условимся называть число переходов $|V|2^{|A|}$ и максимальную высоту max{h(S(v)): v $\in$ V} подстановок, задействованных в π, соответственно. Для последовательности вершин $v_1,\dots,v_{n-1},v_n$ композиция подстановок $S(v_n)S(v_{n-1})\dots S(v_1)$, ассоциированных с этими вершинами, будет обозначаться $S(v_1,\dots,v_n)$.

Рассмотрим программу π = <V, **вход**, **выход**, **тупик**,S,T>. Последовательность пар

$w=(v_0,C_0),(v_1,C_1),\dots,(v_i,C_i),\dots$ (1)

где $v_0,v_1,\dots,v_i,\dots$ - вершины программы, $C_0,C_1,\dots,C_i,\dots$ - тесты, и $v_{i+1}=T(v_i,C_i)$ для всякого i, i $\geq$ 0, назовем *трассой* в π. Мы будем говорить, что трасса (1) последовательно *проходит через вершины* $v_0,v_1,\dots,v_i,\dots$, программы π с тестами $C_0,C_1,\dots,C_i,\dots$.*Результатом трассы* w, оканчивающейся парой (v_i,C_i), назовем подстановку $S(v_0,v_1,\dots,v_i)$, которую будем обозначать [w]. Трасса w считается *полной*, если v_0=**вход**, а сама последовательность w либо является бесконечной, либо оканчивается парой вида (**выход**,C). Программа π считается *приведенной*, если всякая программная вершина (кроме, быть может, пустого цикла) входит в состав некоторой конечной полной трассы w в π.

Для заданной термальной интерпретации I полная трасса (1) называется *вычислением* π *на* I, если для каждой пары (v_i,C_i) всякая литера L, входящая в тест C_i, выполняется на интерпретации I для подстановки $S(v_0,v_1,\dots,v_i)$. Пары (v_i,C_i), i $\geq$ 0, будем называть *состояниями* вычисления. Всякий раз, когда пара (v,C) входит в состав вычисления (1), мы будем говорить, что w *достигает* указанного состояния. Если некоторая трасса w представляет собой начальный фрагмент вычисления программы для термальной интерпретации I, то мы будем говорить, что w *реализуется на интерпретации* I. Нами изучаются только детерминированные программы, и поэтому всякая программа π порождает единственное вычисление на заданной интерпретации I, которое условимся обозначать r(π,I). Если r(π,I) достигает *заключительного состояния* (**выход**,C_m), мы будем говорить, что вычисление *завершается*, и его *результатом* служит подстановка $S(v_0,v_1,\dots,v_m)$. В противном случае r(π,I) образует бесконечную последовательность, и мы будем говорить, что вычисление *зацикливается* и не имеет результата. Результат вычисления r(π,I) обозначим [r(π,I)], полагая при этом, что [r(π,I)] не определен для зацикливающихся вычислений.

Программы π' и π'' назовем *эквивалентными*, обозначая это отношение $\pi' \sim \pi''$, если для всякой термальной интерпретации I соблюдается равенство результатов вычислений $[r(\pi',I)]=[r(\pi'',I)]$. Проблема эквивалентности для стандартных операторных программ на термальных интерпретациях состоит в том, чтобы для всякой пары программ π', π'' выяснить, выполняется ли отношение эквивалентности $\pi' \sim \pi''$.

Утверждение 1. *[12,13]]. Для любой программы π существует приведенная* программа π', эквивалентная π, такая, что $|\pi'| \le |\pi|$, $h(\pi') \le h(\pi)$.

Таким образом, при исследовании проблемы эквивалентности можно ограничиться рассмотрением только приведенных стандартных программ.

Программу π = <V, **вход, выход, тупик**,S,T> над X, F, $\mathcal{A}$ назовем *ортогональной*, если π удовлетворяет следующим требованиям:

- каждая вершина v связана связана с ортогональной подстановкой S(v), такая, что S(v) не является переименованием, и ни один из термов S(v)[1],...,S(v)[N] не является основным;
- каждый атом A из множества $\mathcal{A}$ имеет вид $p(x_1,...,x_N)$, т.е. A зависит от всех переменных X, фигурирующих в программе.

Отличительная особенность всякой ортогональной программы π состоит в то, что на каждом состоянии в ходе любого ее вычисления на каждом логические условия проверяются на наборах функционально независимых значений переменных. Класс всех ортогональных программ обозначим *OrtSP*. Применительно к классификации стандартных программ, предложенной М.Патерсоном в [1,3], можно отметить, что класс ортогональных программ *OrtSP* содержится классе либеральных программ и в определенном смысле двойственен классу прогрессивных стандартных программ. В терминах настоящей статьи стандартная программа π считается

- *либеральной*, если при любом вычислении (1) ни одно из промежуточных значений переменных не вычисляется дважды, т.е. для любых начальных фрагментов $(v_0,C_0),...,(v_i,C_i)$ и $(v_0,C_0),...,(v_j,C_j)$ вычисления (1) и для всякой пары термов $t'=S(v_0,...,v_i,v_{i+1})[k]$ и $t''= (S(v_0,...,v_j,v_{j+1})[l]$ выполняется следующее условие:
  $(t'\in Term(S(v_0,...,v_i)) \& (t''\in Term(S(v_0,...,v_j)) \Rightarrow ((t' \ne t'') \vee ((i=j) \& (k=l))$
- *прогрессивной*, если для всякой ее внутренней вершины v все термы из Term(S(v)) попарно совместимы.

В [3] установлено, что проблема эквивалентности для прогрессивных программ разрешима.

Как показывают результаты исследования проблемы эквивалентности стандартных схем [3-8], возможность ее эффективного решения в том или ином классе программ тесно связана со свойством либерализуемости программ. Программа π считается *свободной* [13], если любая конечная полная трасса в π реализуема на некоторой термальной интерпретации. Говорят, что класс программ K *либерализуем*, если для всякой программы π из класса K в нем можно отыскать эквивалентную свободную программу π'.

Утверждение 2.

Класс ортогональных программ *OrtSP* либерализуем.

Доказательство. Нетрудно убедиться в том, что *OrtSP* включается в класс либеральных схем. В [1,3] показано, что всякая либеральная схема имеет эквивалентную свободную либеральную схему (т.е. класс либеральных схем либерализуем). Далее можно проверить, что метод либерализации, предложенный в [3] сохраняет свойство ортогональности программ.

Вместе с тем ортогональные программы вообще говоря не обладают свойством консервативности даже в наиболее слабой его разновидности, введенной в рассмотрение Сабельфельдом [7,8].

Для обоснования эквивалентности ортогональных программ нам потребуется следующее достаточно очевидное свойство либеральных программ

Утверждение 3. Для всякой бесконечной трассы (1) в свободной либеральной программе π_1 и для всякого натурального L существует номер N такой, что для любого n, n>N, выполняется $h(S(v_1,v_2,...,v_n))>L$.

Идея, положенная в основу разрешающего алгоритма, такова. Для заданной парыортогональных программ π', π''рассматриваются всевозможные пары трасс $w_1=(v^1_0,C^1_0),(v^1_1,C^1_1),...,(v^1_i,C^1_i),...,$ $w_2=(v^2_0,C^2_0),(v^2_1,C^2_1),...,(v^2_i,C^2_i),...,$ совместно реализуемых на термальных интерпретациях. Всякий раз, когда w_1 и w_2 достигают состояний (v^1_i,C^1_i) и (v^2_i,C^2_i), вычисляется "наибольший общий делитель" θ подстановок $\theta_1=S_{\pi'}(v^1_0,..., v^1_i)$ и $\theta_2=S_{\pi''}(v^2_0,..., v^2_i)$, после чего обе подстановки θ_1 и θ_2 редуцируются путем "деления" на θ. Если высота редуцированных подстановок превосходит некоторую величину, зависящую от размера и высоты программ π', π'', то программы объявляются неэквивалентными. Таким образом, в ходе анализа нам приходится иметь дело лишь с конечным числом пар подстановок ограниченой высоты, что и служит гарантией завершаемости процедуры проверки эквивалентности.

Для более формального описания алгоритма нам потребуется ряд вспомогательных понятий и конструкций. В дальнейшем мы будем полагать, что алфавиты X, F, R, равно как и множество атомов *A*, заданы, причем $X=\{x_1,...,x_N\}$ и $A=\{A_1,...,A_M\}$.

Пусть $\pi = \langle$ V, **вход**, **выход**, **тупик**,S,T $\rangle$ - программа над X, F, *A*. ***Графом потоков данных*** для π назовем ориентированный граф $G_\pi(P,E)$ с множеством вершин $P=V \times X$ и множеством дуг $E=\{<(v',x_i),(v'',x_j)> : v''=T(v',C)$ для некоторого теста C из C_A, и x_i совместим с $S(v'')[j]$ }. Мы будем полагать: что дуги графа помечены теми тестами, которые обеспечивают указанную взаимосвязь между вершинами. Вершины (v,x_i) графа G_π будем называть ***позициями***. Заметим, что в том случае, когда в графе G_π имеется ориентированный маршрут $(v_0,x_i),...,(v_n,x_j)$ из позиции (v_0,x_i) в позицию(v_n,x_j), переменная x_i совместима термом $S(v_1,...,v_n)[j]$. Позицию (v,x_i) будем называть

- ***выходной***, если в графе G_π из нее достижима позиция вида (**выход**, x_j);
- ***циклической***, если в графе G_π из нее достижима либо позиция вида (**тупик**, x_j), либо хотя бы одна сильно связанная компонента.

Графы потоков данных позволяют проследить зависимость значений одних переменных от промежуточных значений других переменных в ходе вычисления программ.

Введем в расмотрение бесконечное множество вспомогательных переменных $Y=\{y_1,y_2,...\}$ и обозначим $Term_{X,Y}$ семейство термов, построенных из множества $X \cup Y$ и функциональных символов множества F. Обозначим $Subst^X_{X,Y}$ и $Subst^Y_{X,Y}$ множества конечных подстановок, отображающих X в $Term_{X,Y}$ и Y в $Term_{X,Y}$ соответственно. *Для пары подстановок* $\eta=\{x_1/t_1,...,x_n/t_n\}$ *из* $Subst^X_{X,Y}$ *и* θ из $Subst^Y_{X,Y}$ композицией $\eta\theta$ является подстановка $\{x_1/t_1\theta,...,x_n/t_n\theta\}$ из $Subst^X_{X,Y}$.

Рассмотрим произвольную четверку подстановок η, θ, η', θ' из $Subst^X_{X,Y}$. Если $\eta=\eta'\lambda$, $\theta=\theta'\lambda$ для некоторого переименования λ из $Subst^Y_{X,Y}$,то пары подстановок (η,θ) и (η',θ') называются *подобными* (отношение подобия обозначим $\approx$). В том случае, когда для некоторой ортогональной подстановки $\lambda=\{y_1/t_1,...,y_k/t_k\}$ выполняются равенства $\eta=\eta'\lambda$, $\theta=\theta'\lambda$, причем ни один из термов $t_1,...,t_k$ не согласуется ни с η', ни с θ', мы условимся называть пару (η,θ) *сократимой*, а подстановку λ - делителем η и θ. Если же пара (η',θ') оказывается несократимой, то мы будем называть ее *редукцией* (η,θ), а подстановку λ - ***наибольшим общим делителем*** η и θ. Обозначим $red(\eta,\theta)$ совокупность всевозможных редукций (η,θ). Справедливы следующие свойства ортогональных подстановок и их редукций: которые потребуются в дальнейшем.

Утверждение 4. Если $(\eta',\theta') \in red(\eta,\theta)$, то для для любой пары подстановок η'',θ'' из $Subst^Y_{X,Y}$ справедливо равенство $red(\eta''\eta,\theta''\theta)=red(\eta''\eta',\theta''\theta')$.

Утверждение 5. Пусть (η', θ') - редукция пары ортогональных подстановок (η,θ). Тогда обе подстановки η', θ' также ортогональные и, более того, всякий раз когда пара термов $\eta'[i]$ и $\theta'[j]$ совместимы, по крайней мере один из них является переменной.

Утверждение 6. Пусть (η', θ') - редукция пары ортогональных подстановок (η,θ). Тогда $\eta=\theta$ тогда и только тогда, когда $\eta'=\theta'$.

Утверждение 7. Пусть (η', θ') - редукция пары ортогональных подстановок (η,θ). Если $h(\eta[i])=h(\theta)+L$, то либо $h(\eta'[i])>L$, либо терм $\eta'[i]$ – перемнная из Y, несовместимая ни с одним из термов подстановки θ'.

Утверждение 8. Если (η', θ') и (η'', θ'') - редукции (η,θ), то $(\eta', \theta') \approx (\eta'', \theta'')$.

Утверждение 9. Редукция (η', θ') ортогональных подстановок (η, θ) вычислима за время $O(\max(|\eta|,|\theta|))$.

Редукции пар подстановок будут применяться для представления и сравнительного анализа промежуточных данных в ходе совместных вычислений программ.

Распознавание эквивалентности программ будем осуществлять при помощи критериальных графов (см. [9-11]).

Рассмотрим пару свободных приведенных ортогональных программ над алфавитами переменных X, функциональных символов F, и атомов ***A***.

π_i = <V_i, **вход, выход, тупик**, S_i, T_i >, i=1,2.

Критериальный граф $\Gamma(\pi_1, \pi_2)$ для π_1, π_2 представляет собой помеченный ориентированный граф, вершинами которого служат наборы (v,u,η,θ) такие, что $v \in V_1$, u V_2, (η,θ) - несократимая пара подстановок из $Subst^X_{X,Y}$. Наборы (v,u,η,θ) и (v',u',η',θ') будут считаться подобными, если v=v', u=u', $(\eta,\theta) \approx (\eta',\theta')$. Вершина (**вход**, **вход**, η_0,θ_0), где $(\eta_0,\theta_0) \in red(S_1(\textbf{вход}), S_2(\textbf{вход}))$ полагается *корнем* $\Gamma(\pi_1, \pi_2)$. Множество вершин разбито на пять подмножеств $Z_<$, $Z_>$, $Z_=$, $Z_\varnothing$, и $Z_\neq$ согласно следующим признакам:

$Z_< = \{(v,u,\eta,\theta) : v \neq$ **вход**, и при этом либо $u =$ **выход**, либо $\eta \leq \theta\}$;
$Z_> = \{(v,u,\eta,\theta) : u \neq$ **вход**, и при этом либо $v =$ **выход**, либо $\theta \leq \eta\}$;

$Z_= = \{(v,u,\eta,\theta) : u \neq$ **выход**, $v \neq$ **выход**, $\theta = \eta\}$;

$Z_\varnothing = \{(v,u,\eta,\theta) : u =$ **выход**, $v =$ **выход**$\}$;
все прочие вершины относятся к подмножеству $Z_\neq$.

Вершины графа $\Gamma(\pi_1, \pi_2)$ соединены дугами. Дуги графа $\Gamma(\pi_1, \pi_2)$ помечены парами (C_1,C_2), элементы которых выбираются из множества $\mathcal{C}_A \cup \{\perp\}$. Рассмотрим произвольную вершину $z = (v,u,\eta,\theta)$ графа $\Gamma(\pi_1, \pi_2)$. Возможны следующие варианты соединения вершины z с другими вершинами графа в зависимости от ее принадлежности указанным выше множествам.

1. Если $z \in Z_<$, то из вершины z исходят 2^M дуг, помеченных всевозможными парами вида $(C,\perp)$, $C \in \mathcal{C}_A$. Каждая дуга, помеченная $(C, \perp)$, соединяет z с вершиной $z'=(v',u',\eta',\theta')$ такой, что $v'=T_1(v,C)$, $u'=u$, $(\eta',\theta') \in red((S_1\}(v')\eta,\theta)$.
2. Если $z \in Z_>$, то из вершины z исходят 2^M дуг, помеченных всевозможными парами вида $(\perp,C)$, $C \in \mathcal{C}_A$. Каждая дуга, помеченная $(\perp,C)$ соединяет z с вершиной $z'=(v',u',\eta',\theta')$ такой, что $v'=v$, $u'=T_2(u,C)$, $(\eta',\theta') \in red(\eta,S_2(u')\theta)$.
3. Если $z \in Z_=$, то из вершины z исходят 2^M дуг, помеченных всевозможными парами вида (C,C), $C \in \mathcal{C}_A$. Каждая дуга, помеченная (C,C) соединяет z с вершиной $z'=(v',u',\eta',\theta')$ такой, что $v'=T_1(v,C)$, $u'=T_2(u,C)$, $(\eta',\theta') \in red(S_1\}(v')$ $\eta,S_2(u')\ \theta)$.
4. Если $z \in Z_\neq$, то из вершины z исходят 4^M дуг, помеченных всевозможными парами (C_1,C_2), $C_1 \in \mathcal{C}_A$, $C_2 \in \mathcal{C}_A$. Каждая дуга, помеченная (C_1,C_2) соединяет z с вершиной $z'=(v',u',\eta',\theta')$ такой, что $v'=T_1(v,C)$, $u'=T_2(u,C)$, $(\eta',\theta') \in red(S_1(v')$ $\eta,S_2(u')\ \theta)$.
5. Из вершин семейства $Z_\varnothing$ не исходит ни одной дуги.

Непосредственно из определения графа $\Gamma(\pi_1, \pi_2)$ и утверждения 9 вытекает

Лемма 1. Всякий конечный фрагмент Γ критериального графа $\Gamma(\pi_1, \pi_2)$, состоящий из вершин, доступных из корня $\Gamma(\pi_1, \pi_2)$, может быть построен за время полиномиальное относительно L, H и размера Γ.

С каждым ориентированным маршрутом $\alpha = x_0, x_1,\dots, x_i,\dots$, исходящим из корня $x_0 =$ (**вход**, **вход**, η_0,θ_0) критериального графа $\Gamma(\pi_1, \pi_2)$ и проходящим через вершины $x_i = (v_i,u_i,\eta_i,\theta_i)$, $i \geq 1$, по дугам, помеченным парами (C_{i1},C_{i2}), $i \geq 0$, можно ассоциировать пару полных трасс w_1 и w_2 в программах π_1 и π_2 соответственно. Каждую из этих трасс w_k , $k=1,2$, мы будем называть *k-проекцией* корневого маршрута α и обозначать $\alpha|_k$. Трасса $\alpha|_1$ определяется следующим образом. Рассмотрим в маршруте α подпоследовательность всех вершин $x'_0, x'_1,\dots, x'_j,\dots, x'_j = (v'_j,u'_j,\eta'_j,\theta'_j)$, $i \geq 0$, из которых исходят дуги: помеченные парами (C_{j1},C_{j2}), $j \geq 0$, такими, что $C_{j1} \neq \perp$. Тогда $\alpha|_1 = (u'_0,C'_0),(u'_1,C'_1),\dots,(u'_j,C'_j),\dots$. Аналогичным образом, выделяя дуги, помеченные парами (C_{j1},C_{j2}), $C_{j2} \neq \perp$, определяется проекция $\alpha|_2$ корневого маршрута α.

Как было отмечено, критериальные графы используются для представления совместных пар вычислений программ. Вычисления r_1 и r_2 программ π_1 и π_2 назовем ***совместными***, если $r_1=r(\pi_1,I)$, $r_2=r(\pi_2,I)$ для некоторой интерпретации. Критериальный граф $\Gamma(\pi_1, \pi_2)$ представляет всвозможные пары совместных вычислений свободных ортогональных программ π_1 и π_2, о чем свидетельствует следующие леммы.

Лемма 2. Если маршрут α, выходящий из корня критериального графа $\Gamma(\pi_1, \pi_2)$, достигает вершины (u,v,η,θ), то его проекции $\alpha|_1$ и $\alpha|_2$ являются начальными фрагментами пары совместных вычислений $r_1=r(\pi_1,I)$, $r_2=r(\pi_2,I)$ программ π_1, π_2, причем эти вычисления достигают вершин u и v соответственно в программах π_1 и π_2, и (η,θ) является редукцией пары подстановок $([\alpha|_1],[\alpha|_2])$.

Лемма 3. Если вычисления r_1 и r_2 программ π_1 и π_2 являются совместными, то из корня критериального графа $\Gamma(\pi_1, \pi_2)$ исходит маршрут α, проекциями которого являются вычисления r_1 и r_2.

Обоснование лемм проводится индукцией по длине вычислений с использованием особенностей указанного выше разбиения вершин критериального графа на классы утверждений 4-8 (более подробная схема их доказательства представлена в. [10,11]).

Положим $L=\max(|\pi_1|,|\pi_2|)$, $H=\max(h(\pi_1),h(\pi_2))$. Вершина $z=(v,u,\eta,\theta,L_1,L_2)$ будет считаться ***опровергающей***, если она удовлетворяет одному из следующих требований.

A). $z \in Z_\varnothing$ и $\eta \neq \theta$;

B). $u \neq v$, **тупик** $\in \{u,v\}$

C). $\exists\, i : 1 \le i \le N : (v,x_i)$ - выходная или циклическая позиция в графе потоков данных $G(\pi_1)$, причем либо терм $\eta[i]$ не согласован с подстановкой θ, либо $h(\eta[i]) > LH$;

D). $\exists\, j : 1 \le j \le N : (u,x_j)$ - выходная или циклическая позиция в графе потоков данных $G\{\pi_2)$, причем либо терм $\theta[j]$ не согласован с подстановкой η, либо $h(\theta[j]) > LH$.

Лемма 4. Пусть π_1 и π_2 - приведенные свободные ортогональные программы. Тогда π_1 и π_2 не являются эквивалентными в том и только том случае, когда из корня критериального графа $\Gamma(\pi_1,\pi_2)$ достижима хотя бы одна опровергающая вершина.

Доказательство. *Необходимость.* Предположим, что программы π_1 и π_2 не эквивалентны. Тогда $[r(\pi_1,I)] \neq [r(\pi_2,I)]$ для некоторой интерпретации I. Рассмотрим совместные вычисления $r_1=r(\pi_1,I)$, $r_2=r(\pi_2,I)$ программ π_1, π_2. Согласно лемме 3 оба вычисления являются проекциями некоторого корневого маршрута α в критериальном графе $\Gamma(\pi_1,\pi_2)$. Если оба вычисления r_1, r_2 конечны, то согласно лемме 2 маршрут α оканчивается в вершине x=(**выход**, **выход**, η,θ) такой, что $(\eta,\theta) \in red([r_1], [r_2])$. По утверждению 6 из $[r_1] \neq [r_2]$ следует $\eta \neq \theta$. Следовательно, x – искомая опровергающая вершина типа A, достижимая из корня $\Gamma(\pi_1,\pi_2)$. Если одно из вычислений (например, r_1) является конечным, а другое (в данном случае r_2) бесконечное, то в зависимости от вида вычисления r_2 возможны следующие две альтернативы. В том случае, если вычисление r_2 достигает вершины **тупик**, маршрут α достигнет опровергающей вершины типа B. Рассмотрим случай: когда вычисление r_2 является бесконечным нетупиковым вычислением. Предположим, что вычисление r_2 проходит последовательно через вершины $v_1,v_2,\ldots,v_n,\ldots$.Тогда согласно утверждению 3, начиная с некоторого номера N_1 для вершин v_n , $n > N_1$, будет выполняться соотношение $h(S(v_1,v_2,\ldots,v_n))>LH+h([r_1])$. Поскольку вычисление r_2 бесконечное, то по крайней мере одна из позиций (v_n,x_i) в графе потоков данных $G(\pi_2)$ является циклической. В силу либеральности ортогональных программ ни один из термов в подстановках $S(v_1,v_2,\ldots,v_n)$ не может перевычисляться дважды. Поэтому начиная с некоторого номера N_2 в каждой подстановке $S(v_1,v_2,\ldots,v_n)$ найдется терм $S(v_1,v_2,\ldots,v_n)[j]$ такой, что позиция (v_n,x_j) в графе потоков данных $G(\pi_2)$ – циклическая, и $h(S(v_1,v_2,\ldots,v_n)[j]) > LH+h([r_1])$. Но в этом случае корневой маршрут α достигнет вершины $x=($**выход**$, v_n, \eta,\theta)$ такой, что $(\eta,\theta) \in red(S(v_1,v_2,\ldots,v_n), [r_1])$. Тогда из утверждения 7 следует, что x – искомая опровергающая вершина типа D.

Достаточность. Предположим, что некоторая опровергающая вершина $z=(u,v,\eta',\theta')$ доступна в $\Gamma(\pi_1, \pi_2)$. Если это одна из вершин типа A или B, то неэквивалентность π_1 и π_2 следует из леммы 2. Допустим, что вершина z относится к типу D, причем (v,x_i) - выходная позиция в графе потоков данных $G(\pi_1)$, и $h(\eta[i]) > LH$. Тогда, согласно лемме 2, имеется пара совместных вычислений r_1 и r_2 программ π_1 и π_2, достигающих состояний (u,η) и (v,θ) таких, что отличие между подстановками η и θ "очень велико", причем отличие η от связки x_i/t_i в подстановке θ не может быть устранено никакой композицией, составленной менее, чем из L подстановок, приписанных программе π_1. Поскольку (v,x_i) - выходная позиция, мы можем продолжить вычисление r_2 так, чтобы оно завершилось и при этом содержало t_i в в подстановке, являющейся результатом вычисления. С другой стороны, поскольку π_1 - приведенная программа, мы можем продолжить r_2 так, чтобы это вычисление завершилось очень скоро (спустя менее L шагов) после того, как будет достигнуто состояние (v,η). В этом случае программа π_1 не сможет вычислить терм t_i, и, следовательно, результат вычисления π_1 будет отличен от результата вычисления π_2. Следует заметить, что это "неустранимое" отличие подстановок η и θ приводит к тому, что вычисления программ π_1 и π_2, продолженные указанным выше способом, остаются совместными. Таким образом, для некоторой интерпретации I выполняется $[r(\pi_1,I)] \neq [r(\pi_2,I)]$. Следовательно, программы π_1 и π_2 не являются эквивалентными Проводя аналогичные рассуждения, к такому же выводу можно прийти и для других вариантов доступных опровергающих вершин типа C или D.

Лемма 5. Если из корня критериального графа $\Gamma(\pi_1, \pi_2)$ достижима хотя бы одна вершина (u,v,η,θ) такая, что $h(\eta) > 2L^2HN$ (или $h(\theta) > 2L^2HN$), то из корня критериального графа $\Gamma(\pi_1, \pi_2)$ достижима некоторая опровергающая вершина, удовлетворяющая требованию C (соответственно, D).

Справедливость леммы устанавливается исходя из чисто комбинаторных соображений устройства графов потоков данных: формирование терма большой высоты в подстановке возможно только на таком вычислении, при котором в графе потоков данных проходится цикл. Достижимость такого цикла приводит к появлению такой циклической позиции (v,x_i) в графе потоков данных $G(\pi_1)$, что $h(\eta[i]) > LH$;

Теорема. *Проблема эквивалентности для класса программ OrtSP разрешима.*

Доказательство. Пусть π_1 и π_2 - ортогональные приведенные свободные программы, подстановки которых содержат только функциональные символы из множества $F=\{f_1,...,f_l\}$. Согласно лемме 4 для проверки эквивалентности π_1 и π_2 достаточно убедиться в том, что некоторая опровергающая вершина доступна в критериальном графе $\Gamma(\pi_1, \pi_2)$. Обозначим K максимальное число пар подстановок из $Subst^X_{X,Y}$, высота которых не превосходит $2L^2HN$. Тогда на основании леммы 5 достаточно проверить не более $K|\pi_1||\pi_2|$ доступных вершин в $\Gamma(\pi_1,\pi_2)$, чтобы убедиться в наличие хотя бы одной опровергающей вершины. Лемма 1 обеспечивает эффективность такой проверки.

Литература

1. Luckham D.C., Park D.M., Paterson M.S. On formalized computer programs // *Journal of Computer and System Science*, 1970, **4**, N 3. p.220-249.
2. Летичевский А.А. Функциональная эквивалентность дискретных преобразователей I. // *Кибернетика*. 1969, N 2, c.5-15.
3. Paterson M.S., Decision problems in computational models // *SIGPLAN Notices*. – 1972, **7**, p.74-82.
4. Летичевский А.А. Функциональная эквивалентность дискретных преобразователей III. // *Кибернетика*. 1972, N 1, c.1-4.
5. Годлевский А.Б. Об одном случае специальной проблемы функциональной эквивалентности дискретных преобразователей. // *Кибернетика*. 1974, N 3, c.32-36.
6. Петросян Г.Н. Проблема включения на подпамяти для операторных схем и случай ее разрешения // *Системное и теоретическое программирование*. Новосибирск: ВЦ СО АН СССР, 1974, c.130-151.
7. Сабельфельд В.К. Новый класс схем с разрешимой функциональной эквивалентностью // *Информатика: инструментальные средства*. Новосибирск, 1988, c.109-126.
8. Sabelfeld V.K. An algorithm deciding functional equivalence in a new class of program schemes // *Theoretical Computer Science*. 1990, **71**, N 2, p.265-279.
9. Zakharov V.A. An efficient and unified approach to the decidability of equivalence of propositional program schemes // *Lecture Notes in Computer Science*. Springer-Verlag, 1998, **1443**,. p.247--259.
10. Захаров В.А., Быстрые алгоритмы разрешения эквивалентности операторных программ на уравновешенных шкалах // *Математические вопросы кибернетики*, вып.7, 1998, c.303-324.
11. Захаров В.А., Быстрые алгоритмы разрешения эквивалентности пропозициональных операторных программ на упорядоченных полугрупповых шкалах // Вестник Московского университета, сер. 15, Вычислительная математика и кибернетика, 1999, N 3, c.29-35.
12. Apt K. From Logic Programming to Prolog, Prentice Hall, 1997.
13. Котов В.Е., Сабельфельд В.К. Теория схем программ. М.:Наука, 1991. 348 с.
14. Подловченко Р.И. О проблеме эквивалентных преобразований программ // *Программирование*, 1986, N 6, c.3-14.