Синтез функциональных программ при помощи метода дедуктивных таблиц.

Подготовил: Фастовец Н.Н.

Научный руководитель: Корухова Ю.С.

Содержание

- Автоматический синтез программ
- Дедуктивные таблицы
- Свойства дедуктивных таблиц
- Дедуктивные правила
- Пример синтеза
- Вспомогательные таблицы
- Пример ввода вспомогательных таблиц
- Заключение

Автоматический синтез программ

Предпосылки :

- Увеличение сложности ПО
- Увеличение требований к надежности ПО

Основные направления:

- Дедуктивный синтез
- Индуктивный синтез
- Трансформационный синтез

Дедуктивные таблицы (1)

Один из методов дедуктивного синтеза.

Спецификация задается в виде формулы логики предикатов первого порядка

$$\forall x \ \exists y \ Q[x,y]$$

где x – входная переменная,

y – выходная переменная,

Q – логическая формула, устанавливающая связь между входными и выходными переменными.

Дедуктивные таблицы (2)

Структура дедуктивной таблицы

Утверждения	Цели	Выходной терм		
		f1(x)	…	fn(x)
A1		t1		tn
…		…		…
Ak		t1’		tn’
	G1	s1		sn
	…	…		…
	Gm	s1’		sn’

$$\mathrm{If}(A_1 \vee \ldots \vee A_n)\ \mathrm{then}\ (G_1 \wedge \ldots \wedge G_m)$$

Дедуктивные таблицы (3)

Терм t, не содержащий свободных (на связанных с квантором) переменных, удовлетворяет строке таблицы

A		s

или

	G	s

если для некоторой подстановки λ:

- Утверждение Aλ не содержит свободных переменных и является ложным (Gλ – истинным)
- Если строка имеет выходной терм s, то sλ не содержит свободных переменных и равен tλ.

Дедуктивные таблицы (4)

Доказательство в дедуктивной таблице проводиться до получения финальной строки

	true	t1		tn

или

false		t1		tn

Свойства дедуктивных таблиц (1)

- <u>Двойственность</u>

  строка

	G	t1

  эквивалентна строке

$\neg$ G		t1

Свойства дедуктивных таблиц (2)

- <u>Переименование свободных переменных</u>

  π – перестановка

  строка

	G	t

  эквивалентна строке

	$G\pi$	$t\pi$

Свойства дедуктивных таблиц (3)

- <u>Добавление примера</u>

  λ – подстановка

  Таблица

	G	t

  эквивалентна

	G	t
	$G\lambda$	$t\lambda$

Свойства дедуктивных таблиц (4)

- <u>Тождественные преобразования</u>

  Эквивалентность таблиц не нарушается при применении в строках тождественных преобразований на основе тождеств алгебры логики:

  $A \wedge A = A$; $A \wedge \text{false} = \text{false}$;

  $A \vee \text{true} = \text{true}$; $\neg\neg A = A$;

Свойства дедуктивных таблиц (5)

- Добавление и удаление тождественно истинного утверждения и тождественно ложной цели
- Добавление и удаление свободной переменной

Дедуктивные правила (1)

- Вычислимые термы – термы, не содержащие свободных переменных, которые можно получить на основе уже определенных констант, функций и предикатов.
- Таблицы, для которых совпадают множества удовлетворяющих им вычислимых термов термов совпадают, называются подобными.

Дедуктивные правила (2)

- <u>Правило OR- и AND-разделения</u>

	$G1 \vee G2$	t

$\Longrightarrow$

	$G1 \vee G2$	t
	G1	t
	G2	t

$A1 \wedge A2$		t

$\Longrightarrow$

	$A1 \wedge A2$	t
	A1	t
	A2	t

G1,G2,A1,A2 – логические выражения,
t – выходной терм.

Дедуктивные правила (3)

- <u>Правило резолюции</u>

i		G1[P]	t

k		G2[P']	s

$\Downarrow$

r		G1λ [true] $\wedge$ G2λ [false]	If Pλ then sλ else tλ

G1,G2 – логические выражения,

P, P' – некоторые подвыражения G1 и G2,

λ – подстановка, для которой Pλ = P'λ,

s и t – выходные термы.

Дедуктивные правила (4)

- Правило индукции.

  wf-отношение – отношение, исключающее бесконечно убывающие цепочки.

  Спецификация:

	Q[a,z]	z

где a – входной параметр, z – выходная переменная, Q – логическая формула.

Гипотеза индукции:

If x $<_{wf}$ a then Q[x, f(x)]		

где <wf – wf-отношение в рассматриваемой теории

Дедуктивные правила (5)

- <u>Правило замены эквивалентных термов</u>

i		G1[L=R]	t
k		G2<T>	s

r		$G1\lambda$ [false] $\wedge$ $G2\lambda<R\lambda>$	If $[L=R]\lambda$ then $s\lambda$ else $t\lambda$

где G1, G2 – логические выражения,

[L=R] – подвыражение G1,

T – некоторый терм, входящий в G2, такой что для некоторой подстановки $\lambda: T\lambda = L\lambda$,

s и t – выходные термы.

Пример синтеза функциональной программы (1)

Спецификация:

z = fact(n) is

if n = 0 then z = 1 else z = n*fact(n-1)

В таблице

1		$(n=0) \wedge (z=1) \vee \neg (n=0) \wedge (z=n*fact(n-1))$	z

согласно тождеству

if A then B else C $\equiv A \wedge B \vee \neg A \wedge C$

Пример синтеза функциональной программы (2)

1. OR-разделение

1		$(n=0) \wedge (z=1) \vee \neg (n=0) \wedge (z=n*fact(n-1))$	z
2		$(n=0) \wedge (z=1)$	z
3		$\neg (n=0) \wedge (z=n*fact(n-1))$	z

2. Резолюция. Общее подвыражение . (n=0).

4		$(z1=1) \wedge (z2=n*fact(n-1))$	If n = 0 then z1 else z2

3. Добавление примера $\{z1 \leftarrow 1, z2 \leftarrow n*fact(n-1)\}$

4		true	If n = 0 then 1 else n*fact(n-1)

Вспомогательные таблицы (1)

Пусть в таблице присутствуют строки

i		G[t[a],x]	r[a,x]
…	…	…	…
k		F[G[t’[a],x’]]	r’[a,x’]

где G, F – логические выражения,
a – входной параметр, t[a], t’[a] – термы над входным параметром, x, x’ – выходные переменная, r[a,x], r’[a,x’] – выходные термы. F содержит реплику G.

Вспомогательные таблицы (2)

Если $t'[a] <_{wf} t[a]$, можно в выражении G заменить все вхождения t[a] на произвольную константу и построить обобщенную цель G'[c,x], которая станет исходной целью для вспомогательной таблицы (соответственно – спецификацией для вспомогательных функций).

1a		G'[c,x]	x

Вспомогательные таблицы (3)

- Исходная цель

1a		G'[c,x]	x

- Гипотеза индукции

2a	if y <wf c then G'(y,fnew(y))		x

- Воспроизведение шагов доказательства

(k+1)a		F'[G'[t"[c],x"]]	r'''[c, x"]

- Резолюция с гипотезой индукции

(k+2)a		F'[true] /\ (t"[c]$<_{wf}$ c)	r'''[c, fnew(t"[c])]

Вспомогательные таблицы (4)

Лемма

k+1	G_1(y, fnew(y))		

имеющаяся строка (i)

i		G[t[a],x]	r[a,x]

их резолюция завершает доказательство

k+2		true	r''[a,fnew(t[a])]

Пример вывода вспомогательной таблицы (1)

Для функции сортировки списка чисел задается спецификация вида

$z = sort(a)$ is $perm(a,z) \wedge ord(z)$

perm(x,y) – предикат, равный true, если его аргументы-списки есть перемтановки друг-друга.

ord(x) – предикат, равный true, если его аргумент-список упорядочен.

Пример вывода вспомогательной таблицы (2)

- В ходе доказательства получается

20		perm(tail(b), w^x) ∧ perm(tail(b), x2^w) ∧ perm(tail(b), x3^x) ∧ [**if** iselem(f, x) **then** head(b) ≤ f] ∧ [**if** iselem(h, w) **then** h ≤ head(b)]	**if** b=null **then** null **else** sort(w)^ head(b)*sort(x)

- Далее, строка

30		perm(tail(tail(b)), w^y) ∧ perm(tail(tail(b)), v2^w) ∧ perm(tail(tail(b)), x3^y) ∧ [**if** f = head(tail(b)) **then** head(b) ≤ f] ∧ [**if** iselem(f, y) **then** head(b) ≤ f] ∧ [**if** iselem(h, w) **then** h ≤ head(b)] ∧ ¬tail(b)=null	$t_{(30)}$

- Исходная цель вспомогательной таблицы

#	Assertion	Goal	f1	f2
1a		perm(c, w^x) ∧ perm(c, x2^w) ∧ perm(c, x3^x) ∧ [**if** iselem(f, x) **then** head(b) ≤ f] ∧ [**if** iselem(h, w) **then** h ≤ head(b)]	w	x

- Лемма

31	perm(y, les(y)^gre(y)) ∧ perm(y, x2^les(y)) ∧ perm(y, x3^gre(y)) ∧ [**if** iselem(f, gre(y)) **then** head(b) ≤ f] ∧ [**if** iselem(h, les(y)) **then** h ≤ head(b)]		

Конец

Источники

- Источники
- Kreitz, C. Program Synthesis Chapter III.2.5 of *Automated Deduction – A Basis for Applications*, pp. 105–134, Kluwer, 1998.
- Ayari, A., Basin D.: A High-Order Interpretation Of Deductive Tableau
- J. Symbolic Computation (2001) 11, 1-32
- Manna Z., Waldinger R.: Fundamentals of Deductive Program Synthesis
- IEEE Transactions on Software Engineering, vol. 12, No. 8, August 1992
- Traugott J. Deductive Synthesis of Sorting Programs
- Journal of Symbolic Computation, vol.7, 1989.