

Модели. Алгебраические типы данных



Теория формальных языков
2021 г.



Проектирование структур

- На прошлом занятии — полиморфные функциональные типы без структур. Как навесить структуры?
- Хорошо спроектированные полиморфные функции действуют семантически одинаково на аргументах разного вида.
- Как описать аналог этого свойства для структур данных?



Сигнатура с сортами

Определение

Алгебра \mathcal{A} — набор носителей (сорт), выделенных элементов и функций первого порядка в сигнатуре данных носителей. На формальном языке,

$$\mathcal{A} = \langle \{\mathcal{N}_i\}, \{\mathbf{c}_i\}, \{f_i : \mathcal{N}_{i_1} \times \dots \mathcal{N}_{i_{k_i}} \rightarrow \mathcal{N}'_i\} \rangle.$$



Сигнатура с сортами

Определение

Алгебра \mathcal{A} — набор носителей (сорт), выделенных элементов и функций первого порядка в сигнатуре данных носителей. На формальном языке,
 $\mathcal{A} = \langle \{\mathcal{N}_i\}, \{\mathcal{C}_i\}, \{f_i : \mathcal{N}_{i_1} \times \dots \mathcal{N}_{i_{k_i}} \rightarrow \mathcal{N}_i'\} \rangle$.

Иными словами, в алгебре у функций сигнатуры не бестиповые, а оснащены простыми типами.

- $\lambda x.x \ x$ — нет сортов;
- $\lambda x y. \text{if } x \text{ then } y * 2 \text{ else } y$ — есть сорта.



Сигнатура с сортами

Определение

Алгебра \mathcal{A} — набор носителей (сорт), выделенных элементов и функций первого порядка в сигнатуре данных носителей. На формальном языке,

$$\mathcal{A} = \langle \{\mathcal{N}_i\}, \{c_i\}, \{f_i : \mathcal{N}_{i_1} \times \dots \mathcal{N}_{i_{k_i}} \rightarrow \mathcal{N}'_i\} \rangle.$$

Иными словами, в алгебре у функций сигнатуры не бестиповые, а оснащены простыми типами.

- $\lambda x.x \ x$ — нет сортов;
- $\lambda x y. \text{if } x \text{ then } y * 2 \text{ else } y$ — есть сорта.

Зачем многосортные алгебры нужна в CS?

- Сигнатура Σ — синтаксис ЯП;
- $\Sigma(X) + \Gamma_X$ (контекст) — множество термов ЯП.
- Алгебра над Σ — семантика ЯП.



Универсальные алгебры

- В CS выделенные значения — функции от нуля аргументов (конструкторы).
- Каждой константе в Σ соответствует ровно один носитель из \mathcal{A} ;
- Каждому функциональному символу в Σ соответствует отображение с такой же сигнатурой.
- Каждому присвоению сорта (утверждению о типизации) в контексте Γ вида $x_i : T_i$ соответствует окружение — отображение из x_i в множество носителей \mathcal{A} .



Интерпретации

Пусть η — окружение. Значение $\mathcal{A}[[M]]\eta$ (читаем: интерпретация M) определяется рекурсивно.

- $\mathcal{A}[[x]]\eta = \eta(x)$
- $\mathcal{A}[[f(M_1, \dots, M_n)]]\eta = f^{\mathcal{A}}(\mathcal{A}[[M_1]]\eta, \dots, \mathcal{A}[[M_n]]\eta).$



Интерпретации

Пусть η — окружение. Значение $\mathcal{A}[[M]]\eta$ (читаем: интерпретация M) определяется рекурсивно.

- $\mathcal{A}[[x]]\eta = \eta(x)$
- $\mathcal{A}[[f(M_1, \dots, M_n)]]\eta = f^{\mathcal{A}}(\mathcal{A}[[M_1]]\eta, \dots, \mathcal{A}[[M_n]]\eta)$.
- Множество натуральных чисел — интерпретация термов над сигнатурой $\{s(\bullet), z\}$ — единственный одноместный конструктор + константа.
- Множество двоичных деревьев — интерпретация термов над сигнатурой $\{\langle \bullet, \bullet \rangle, e\}$ — единственный двухместный конструктор + константа.
- Свободная алгебра — каждый функциональный символ интерпретируется собой.

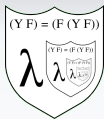


Интерпретации

Пусть η — окружение. Значение $\mathcal{A}[[M]]\eta$ (читаем: интерпретация M) определяется рекурсивно.

- $\mathcal{A}[[x]]\eta = \eta(x)$
- $\mathcal{A}[[f(M_1, \dots, M_n)]]\eta = f^{\mathcal{A}}(\mathcal{A}[[M_1]]\eta, \dots, \mathcal{A}[[M_n]]\eta)$.
- Множество натуральных чисел — интерпретация термов над сигнатурой $\{s(\bullet), z\}$ — единственный одноместный конструктор $+$ константа.
- Множество двоичных деревьев — интерпретация термов над сигнатурой $\{\langle \bullet, \bullet \rangle, e\}$ — единственный двухместный конструктор $+$ константа.
- Свободная алгебра — каждый функциональный символ интерпретируется собой.

Лемма о подстановке: $[[M[x := N]]]\eta = [[M]](\eta[x := [[N]]\eta])$.



Модели

Выполнимость уравнения в окружении η принято обозначать $\mathcal{A}, \eta \models M = N[\Gamma]$, где η согласовано с Γ (т.е. значения переменных имеют правильные типы).

- Выполнимость в модели: $\mathcal{A} \models M = N[\Gamma]$ (для любого окружения).
- Общезначимость: $\models M = N[\Gamma]$ (для любого окружения в любой алгебре).



Модели

Выполнимость уравнения в окружении η принято обозначать $\mathcal{A}, \eta \models M = N[\Gamma]$, где η согласовано с Γ (т.е. значения переменных имеют правильные типы).

- Выполнимость в модели: $\mathcal{A} \models M = N[\Gamma]$ (для любого окружения).
- Общезначимость: $\models M = N[\Gamma]$ (для любого окружения в любой алгебре).

Бесконечность HOF

Пусть в алгебре \mathcal{A} есть одна бинарная операция \circ и выделенное значение a такое, что $\mathcal{A} \models (a \circ x) \circ y = x$. Тогда если носитель \mathcal{A} содержит больше одного элемента, то он бесконечен.



Выводимость и выполнимость

- Навесим на свободную алгебру над Σ множество уравнений E в сигнатуре Σ и объявим полученную структуру алгебраической спецификацией.
- Скажем, что из E семантически следует $M = N$, если во всех алгебрах, удовлетворяющих E , выполняется $M = N$. Пишем: $E \models M = N$.
- Допустимые правила вывода: симметричность и транзитивность равенства, введение свободной переменной в контекст и правило подстановки.
- (теоремы корректности и полноты)
 $E \vdash M = N \Leftrightarrow E \models M = N$.



Пример

Общерекурсивных полиморфных функций с типом $P = ((A \Rightarrow B) \Rightarrow A) \Rightarrow A$ (P — формула Пирса) не существует.

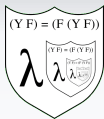


Пример

Общерекурсивных полиморфных функций с типом $P = ((A \Rightarrow B) \Rightarrow A) \Rightarrow A$ (P — формула Пирса) не существует.

Рассмотрим следующую конечнозначную модель и убедимся, что формула Пирса в ней не общезначима.

A	B	$A \Rightarrow B$	P
0	0	1	...
0	n	n	...
0	1	1	...
n	0	0	...
n	n	n	n
n	1	1	...
1	0	0	...
1	n	n	...
1	1	1	...



Теорема Линденбаума–Тарского

Если алгебраическая спецификация имеет модель, тогда она имеет конечную модель.

Основа современных методов анализа программ —
Satisfiability Modulo Theories (поиск конечной контрмодели).



Универсум Эрбрана

Определение

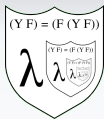
Набор всех правильно типизированных термов в сигнатуре $\langle\{f_i\}, \{c_j\}\rangle$ — универсум Эрбрана.

Интуитивно задаёт наиболее общую модель для системы вывода, но не учитывает эквациональность, т.е. возможность равенства термов.



Инициальные алгебры

Пусть \mathcal{C} — множество алгебр над сигнатурой Σ (Σ -алгебр) и $A \in \mathcal{C}$. A называется инициальной, если для любой $A' \in \mathcal{C}$ существует единственный гомоморфизм из A в A' .



Инициальные алгебры

Пусть \mathcal{C} — множество алгебр над сигнатурой Σ (Σ -алгебр) и $A \in \mathcal{C}$. A называется инициальной, если для любой $A' \in \mathcal{C}$ существует единственный гомоморфизм из A в A' .

- «Никакого мусора» — в A должно быть так мало различных элементов, насколько возможно.
- «Никакой путаницы» — в A элементы должны быть равны только если без этого равенства не обойтись.



Примеры

- Рассмотрим односортовую сигнатуру $\Sigma_0 = \langle 0, S(x) = x \rightarrow x + 1 \rangle$ над типом натуральных чисел. Тогда свободная алгебра с константой 0 и единственным унарным конструктором $S(\bullet)$ является инициальной для алгебр над Σ_0 . Если добавить в сигнатуру сложение и аксиомы $x + 0 = x$, $x + S(y) = S(x + y)$, тогда инициальная алгебра станет стандартной арифметикой Пресбургера.



Примеры

- Рассмотрим односортовую сигнатуру $\Sigma_0 = \langle 0, S(x) = x \rightarrow x + 1 \rangle$ над типом натуральных чисел. Тогда свободная алгебра с константой 0 и единственным унарным конструктором $S(\bullet)$ является инициальной для алгебр над Σ_0 . Если добавить в сигнатуру сложение и аксиомы $x + 0 = x$, $x + S(y) = S(x + y)$, тогда инициальная алгебра станет стандартной арифметикой Пресбургера.
- Алгебра с добавлением ω для наименьшего числа, большего всех натуральных, инициальной не является. Аналогично \mathbb{Z}_k .



Примеры

- Рассмотрим односортную сигнатуру $\Sigma_0 = \langle 0, S(x) = x \rightarrow x + 1 \rangle$ над типом натуральных чисел. Тогда свободная алгебра с константой 0 и единственным унарным конструктором $S(\bullet)$ является инициальной для алгебр над Σ_0 . Если добавить в сигнатуру сложение и аксиомы $x + 0 = x$, $x + S(y) = S(x + y)$, тогда инициальная алгебра станет стандартной арифметикой Пресбургера.
- Алгебра с добавлением ω для наименьшего числа, большего всех натуральных, инициальной не является. Аналогично \mathbb{Z}_k .
- Выполнимость в инициальной \mathcal{A} неидентична доказуемости. См. коммутативность сложения: $\mathcal{A} \models M + N = N + M$ в инициальной модели, но это не так для ординальной модели.



Неподвижная точка

Определение

Скажем, что X — это lfp (наименьшая неподвижная точка) для f над A , если

$$(\forall n f^n(A) \in X) \ \& \ \forall Y (\forall n (f^n(A) \in Y) \Rightarrow X \subseteq Y).$$

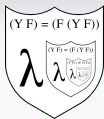
- Лемма Ардена описывает наименьшую неподвижную точку уравнения $X = \phi X \mid \psi$ в регулярных выражениях.
- μ -выражения Клини — наименьшие неподвижные точки уравнений с двухсторонним приписыванием — описывают в точности класс CFL. Например, $\mu X. aXb \mid \epsilon$ задаёт описание языка $\{a^n b^n\}$.



Лемма Ламбека

Лемма Ламбека, полужформально

Пусть F — это сигнатура (здесь не Σ по терминологическим причинам) вместе с функцией обхода, V — множество выделенных значений. Тогда $\text{lfp}(F V)$ — инициальная алгебра над F и V .

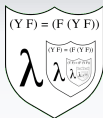


Лемма Ламбека

Лемма Ламбека, полуформально

Пусть F — это сигнатура (здесь не Σ по терминологическим причинам) вместе с функцией обхода, V — множество выделенных значений. Тогда $\text{lfp}(F V)$ — инициальная алгебра над F и V .

Проще говоря, инициальная алгебра задаётся индукцией по построению термов.



Пример: множества

сорта: set, nat, bool

уравнения: $0+0=0$, $0+1=1$, ..., $k+n=m$

$\text{eq? } x \ x = \text{true}$

$\text{eq? } 0 \ 1 = \text{false}$, ..., $\text{eq? } n \ m = \text{false}$

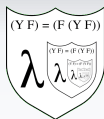
$\text{ismem? } x \ \text{empty} = \text{false}$

$\text{ismem? } x \ (\text{ins } y \ s) = \text{if } (\text{eq? } x \ y) \text{ then true}$
else $\text{ismem? } x \ s$

$\text{union empty } s = s$

$\text{union } (\text{ins } y \ s) \ s' = \text{ins } y \ (\text{union } s \ s')$

Выполняется ли уравнение $\text{union } s \ s' = \text{union } s' \ s$ в инициальной модели?



Пример: множества

сорта: set, nat, bool

уравнения: $0+0=0$, $0+1=1$, ..., $k+n=m$

$\text{eq? } x \ x = \text{true}$

$\text{eq? } 0 \ 1 = \text{false}$, ..., $\text{eq? } n \ m = \text{false}$

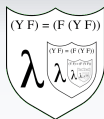
$\text{ismem? } x \ \text{empty} = \text{false}$

$\text{ismem? } x \ (\text{ins } y \ s) = \text{if } (\text{eq? } x \ y) \text{ then true}$
else $\text{ismem? } x \ s$

$\text{union empty } s = s$

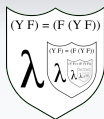
$\text{union } (\text{ins } y \ s) \ s' = \text{ins } y \ (\text{union } s \ s')$

Выполняется ли уравнение $\text{union } s \ s' = \text{union } s' \ s$ в инициальной модели? Ответ: нет, инициальная модель — модель списков с навязанным порядком обхода слева направо.



Спецификация списков

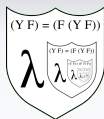
сорта:	<code>list, atom, bool</code>
уравнения:	<code>car(cons x l) = x</code> <code>cdr(cons x l) = l</code> <code>isempty? nil = true</code> <code>isempty? (cons x l) = false</code> <code>cond_a true x y = x</code> <code>cond_a false x y = y</code> <code>cond_b true v1 v2 = v1</code> <code>cond_b false v1 v2 = v2</code> <code>cond_l true l1 l2 = l1</code> <code>cond_l false l1 l2 = l2</code>



Спецификация списков

сорта:	<code>list, atom, bool</code>
уравнения:	$\text{car}(\text{cons } x \ l) = x$ $\text{cdr}(\text{cons } x \ l) = l$ $\text{isempty? nil} = \text{true}$ $\text{isempty? (cons } x \ l) = \text{false}$ $\text{cond}_a \text{ true } x \ y = x$ $\text{cond}_a \text{ false } x \ y = y$ $\text{cond}_b \text{ true } v1 \ v2 = v1$ $\text{cond}_b \text{ false } v1 \ v2 = v2$ $\text{cond}_l \text{ true } l1 \ l2 = l1$ $\text{cond}_l \text{ false } l1 \ l2 = l2$

Выполняется ли уравнение $(\text{cons } (\text{car } l) (\text{cdr } l)) = l$ в
инициальной модели?



Спецификация списков

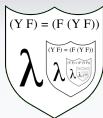
сорта:	list, atom, bool
уравнения:	$\text{car}(\text{cons } x \text{ l}) = x$ $\text{cdr}(\text{cons } x \text{ l}) = \text{l}$ $\text{isempty? nil} = \text{true}$ $\text{isempty? (cons } x \text{ l)} = \text{false}$ $\text{cond}_a \text{ true } x \text{ y} = x$ $\text{cond}_a \text{ false } x \text{ y} = y$ $\text{cond}_b \text{ true } v1 \text{ v2} = v1$ $\text{cond}_b \text{ false } v1 \text{ v2} = v2$ $\text{cond}_l \text{ true } l1 \text{ l2} = l1$ $\text{cond}_l \text{ false } l1 \text{ l2} = l2$

Выполняется ли уравнение $(\text{cons } (\text{car } l) (\text{cdr } l)) = l$ в инициальной модели? Ответ: нет, проблема с термами вида (cdr nil) и (car nil) .



Работа над ошибками

- Ничего не делаем. Пусть (car nil) и (cdr nil) будут новыми терминами в инициальной алгебре. К чему это приведёт? (спойлер: к добавлению бесконечного множества ошибочных термов разных сортов)



Работа над ошибками

- Ничего не делаем. Пусть (car nil) и (cdr nil) будут новыми терминами в инициальной алгебре. К чему это приведёт? (спойлер: к добавлению бесконечного множества ошибочных термов разных сортов)
- Null-неопределённости. Положим, что (car nil) произвольно, $(\text{cdr nil}) = \text{nil}$. Уничтожается индуктивное равенство между списками.



Работа над ошибками

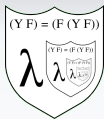
- Ничего не делаем. Пусть (car nil) и (cdr nil) будут новыми терминами в инициальной алгебре. К чему это приведёт? (спойлер: к добавлению бесконечного множества ошибочных термов разных сортов)
- Null-неопределённости. Положим, что (car nil) произвольно, $(\text{cdr nil}) = \text{nil}$. Уничтожается индуктивное равенство между списками.
- Введение терма-ошибки.



Наивная обработка ошибок

уравнения:

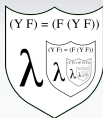
- $\text{car nil} = \text{error}_\alpha$
- $\text{cdr nil} = \text{error}_\lambda$
- $\text{cons error}_\alpha l = \text{error}_\lambda$
- $\text{cons } x \text{ error}_\lambda = \text{error}_\lambda$
- $\text{car error}_\lambda = \text{error}_\alpha$
- $\text{cdr error}_\lambda = \text{error}_\lambda$



Наивная обработка ошибок

уравнения:

- $\text{car nil} = \text{error}_a$
- $\text{cdr nil} = \text{error}_l$
- $\text{cons error}_a \mid = \text{error}_l$
- $\text{cons } x \text{ error}_l = \text{error}_l$
- $\text{car error}_l = \text{error}_a$
- $\text{cdr error}_l = \text{error}_l$
- $\text{isempty? error}_l = \text{error}_b$
- $\text{cond}_a \text{ error}_b \ x \ y = \text{error}_a$
- $\text{cond}_b \text{ error}_b \ x \ y = \text{error}_b$
- $\text{cond}_l \text{ error}_b \ x \ y = \text{error}_l$



Наивная обработка ошибок

уравнения:

- $\text{car nil} = \text{error}_a$
- $\text{cdr nil} = \text{error}_l$
- $\text{cons error}_a l = \text{error}_l$
- $\text{cons } x \text{ error}_l = \text{error}_l$
- $\text{car error}_l = \text{error}_a$
- $\text{cdr error}_l = \text{error}_l$
- $\text{isempty? error}_l = \text{error}_b$
- $\text{cond}_a \text{ error}_b x y = \text{error}_a$
- $\text{cond}_b \text{ error}_b x y = \text{error}_b$
- $\text{cond}_l \text{ error}_b x y = \text{error}_l$

Проблема с аксиомой $\text{car} (\text{cons } x l) = x$ — из-за неё можно доказать, что в данной модели любой терм равен ошибке.



Краткое резюме

- Аксиоматический способ описания семантики двойственен алгебраическому.
- Алгебраические типы данных задаются инициальными алгебрами, т.е. рекурсивный обход по определению существует.
- Non-exhaustive patterns (неисчерпывающие правила переписывания для термов заданного сорта) — указание на возможную ошибку в описании инициальной алгебры.