Алгоритмы и структуры данных

Лекции 2-3. Индуктивные функции на последовательностях

Лектор: ст. преп. каф. ИБ
Абросимов Иван Константинович

2.1.1 Классификация структур данных (1)

- Абстрактный тип данных (АТД) – математическая модель типа данных.
- Структура данных – способ организации данных, корректно поддерживающий определенный набор операций.
- Классификация структур данных:

1) По изменчивости – статические (размер известен на этапе компиляции) и динамические (размер меняется в ходе выполнения программы);
2) По упорядоченности составляющих элементов – линейные (линейный порядок элементов в последовательности) и нелинейные (деревья);
3) По виду используемой памяти – внутренние (в оперативной памяти) и внешние (не в оперативной памяти);
4) По расположению в памяти – непрерывные (данные находятся в памяти последовательно) и ссылочные (данные находятся в памяти не последовательно).

2.1.1 Классификация структур данных (2)

- Работа со структурой данных, осуществляется через специальные функции, совокупность которых составляет интерфейс структуры данных.
- Программа, использующая структуру данных, называется клиентом и взаимодействует со структурой данных только через интерфейс.
- Спецификация АТД – совокупность множеств, функций на этих множествах и свойств этих функций (называемых аксиомами), обеспечивающих независимость АТД от реализации.
- При реализации АТД в виде структуры данных, ее интерфейс создается с учетом спецификации АТД.

2.1.2 Спецификация АТД последовательность (1)

- Множество X^* конечных последовательностей с элементами из множества X:

$$w \in X^* \Leftrightarrow \exists m \in \mathbb{N}\ w = (x_i, i \in \overline{1,m}), x_i \in X.$$

- Множество непустых конечных последовательностей: $\tilde{X}^* = X^* \backslash \{()\}$.

- Функции – конструкторы, т.е. функции, позволяющие создать последовательности X^* из уже имеющихся последовательностей и элементов множества X:

Название	A	B	Описание
Создать	$\emptyset$	X^*	$\text{create}() = ()$
Присоединить слева	$X \times X^*$	$\tilde{X}^*$	$\text{prefix}(x_0, w) = x_0 \parallel w = (x_i, i \in \overline{0,m})$
Присоединить справа	$X^* \times X$	$\tilde{X}^*$	$\text{postfix}(w, x_{m+1}) = w \parallel x_{m+1} = (x_i, i \in \overline{1,m+1})$
Сцепить	$X^* \times X^*$	X^*	$\text{concat}(w, v) = w \parallel v = (x_i, i \in \overline{1,m+n})$

- Здесь $w = (x_i, i \in \overline{1,m})$, $v = \left(x_{n+j}, j \in \overline{1,n}\right)$, A – область отправления, B – область прибытия функции.

2.1.2 Спецификация АТД последовательность (2)

- Функции – селекторы, т.е. функции, выделяющие части уже имеющейся последовательности

Название	A	B	Описание
Первый	$\tilde{X}^*$	X	$\mathrm{first}(w) = x_1$
Последний	$\tilde{X}^*$	X	$\mathrm{last}(w) = x_m$
Все, кроме первого	$\tilde{X}^*$	X^*	$\mathrm{tail}(w) = (x_i, i \in \overline{2, m})$
Все, кроме последнего	$\tilde{X}^*$	X^*	$\mathrm{lead}(w) = (x_i, i \in \overline{1, m-1})$

- Здесь $w = (x_i, i \in \overline{1, m})$, A – область отправления, B – область прибытия функции.

- Для проверки на последовательности на пустоту используется функция

$$\mathrm{map}(X^*, \{0,1\}) \ni \mathrm{isNull}(w) = \left[\begin{array}{l} 1, w = (), \\ 0, w \neq (). \end{array}\right.$$

2.1.2 Спецификация АТД последовательность (3)

- Свойства функции isNull, связанные с конструкторами:

$\mathrm{isNull}(\mathrm{create}()) = 1$	$\mathrm{isNull}(x_0 \parallel w) = \mathrm{isNull}(w \parallel x_{m+1}) = 0$

- Свойства функции first и last, связанные с конструкторами:

$\mathrm{first}(x_0 \parallel w) = x_0$	$\mathrm{last}(w \parallel x_{m+1}) = x_{m+1}$
$\mathrm{first}(w \parallel x_{m+1}) = \begin{cases} x_{m+1}, & \mathrm{isNull}(w) = 1, \\ \mathrm{first}(w), & \mathrm{isNull}(w) = 0 \end{cases}$	$\mathrm{last}(x_0 \parallel w) = \begin{cases} x_0, & \mathrm{isNull}(w) = 1, \\ \mathrm{last}(w), & \mathrm{isNull}(w) = 0 \end{cases}$

- Свойства функции tail и lead, связанные с конструкторами:

$\mathrm{tail}(x_0 \parallel w) = w$	$\mathrm{lead}(w \parallel x_{m+1}) = w$
$\mathrm{tail}(w \parallel x_{m+1}) = \begin{cases} \mathrm{create}(), & \mathrm{isNull}(w) = 1, \\ \mathrm{tail}(w) \parallel x_{m+1}, & \mathrm{isNull}(w) = 0 \end{cases}$	$\mathrm{lead}(x_0 \parallel w) = \begin{cases} \mathrm{create}(), & \mathrm{isNull}(w) = 1, \\ x_0 \parallel \mathrm{lead}(w), & \mathrm{isNull}(w) = 0 \end{cases}$

- Похожими на АТД последовательность являются массивы и файлы.

2.2.1 Понятие индуктивной функции

- **Определение**. Индуктивная функция – функция $f \in \mathrm{func}(X^*, Y)$, значение которой в точке $w \parallel x$ может быть выражено через $f(w)$ и x.

- Если $f(w) = y$ (т.е. f – скалярная функция), то существует функция g с двумя аргументами, для которой справедливо равенство $f(w \parallel x) = g(y, x)$.

- Если $f(w) = (y_1, y_2, \dots, y_k)$ (т.е. f – векторная функция), то существует функция g с $k+1$ аргументами, для которой справедливо равенство $f(w \parallel x) = g(y_1, y_2, \dots, y_k, x)$.

- Примеры: длина последовательности, сумма и произведение чисел последовательности.

- **Утверждение** (теорема о линейном просмотре для случая скалярной индуктивной функции). Если временная сложность T_g вычисления функции g не зависит от m, то временная сложность вычисления индуктивной функции f такой, что $f(w \parallel x) = g(y, x)$ равна

$$T_f(m) = m \cdot T_g.$$

2.2.2 Схема итераций

- **Определение**. Схема итераций – способ построения по рекуррентной формуле

$$\begin{cases} x_{i+1} = f(x_i), x_0 = x, \\ P(x_N) = 0 \Rightarrow F(x) = g(x_N), \end{cases}$$

для вычисляемой величины F, где P – некоторый предикат, являющийся условием продолжения выполнения тела следующего цикла с предусловием

```
ret_type F(arg_type x)
{
    xi = x;
    while(P(xi)) {xi = f(xi);}
    return(g(xi));
}
```

Реализация алгоритма Евклида с помощью схемы итераций

$$\begin{cases} a_{i+1} = b_i, & b_0 = b, \\ b_{i+1} = a_i \bmod b_i, & a_0 = a. \end{cases}$$

```
int gcd1(int a, int b)
{
    int a_curr = a; // a_i, a_0 = a
    int b_curr = b; // b_i, b_0 = b
    int a_next = 0; // a_(i+1)
    int b_next = 0; // b_(i+1)

    while(b_curr > 0)
    {
        // a_(i+1) = b_i
        a_next = b_curr;

        // b_(i+1) = a_i mod b_i
        b_next = a_curr % b_curr;

        /* замена пары (a_i, b_i)
        парой ( a_(i+1), b_(i+1) ) */
        a_curr = a_next;
        b_curr = b_next;
    }
    return(a_curr);
}
```

$$b_{i+2} = b_i \bmod b_{i+1}, b_0 = a, b_1 = b$$

```
int gcd2(int a, int b)
{
    int b_curr = a; // b_i, b_0 = a
    int b_next = b; // b_(i+1), b_1 = b
    int b_2next = 0; // b_(i+2)

    while(b_next > 0)
    {
        // b_(i+2) = b_i mod b_(i+1)
        b_2next = b_curr % b_next;

        /* замена пары ( b_i, b_(i+1) )
        парой ( b_(i+1), b_(i+2) ) */
        b_curr = b_next;
        b_next = b_2next;
    }
    return(b_curr);
}
```

Итоговая версия

```
int gcd(int a, int b)
{
    int divisible = a; // b_i
    int divider   = b; // b_(i+1)
    int remainder = 0; // b_(i+2)

    while(divider > 0)
    {
        remainder = divisible % divider;
        divisible = divider;
        divider   = remainder;
    }
    return(divisible);
}
```

2.2.3 Стационарные значения индуктивной функции

- Стационарное значение индуктивной функции – значение индуктивной функции f, обозначаемое $\mathrm{sv}(f)$ которое не изменяется при добавлении к аргументу (последовательности) произвольного элемента x, т.е.

$$y = \mathrm{sv}(f) \Leftrightarrow \forall x \in X\ f(w \parallel x) = y.$$

- Наличие стационарного значения у индуктивной функции позволяет при ее вычислении завершить проход по последовательности на том элементе, на котором оно было получено;

- С точки зрения схемы итераций, при реализации индуктивной функции, условие завершения цикла представляет собой конъюнкцию двух предикатов «текущее значение не равно стационарному» и «последовательность не просмотрена».

Реализация проверки наличия отрицательных чисел

- Последовательность моделируется массивом `sequence`

```
bool exist_negative_element(int* sequence, int length)
{
    bool isNegative = false;

    for(int i = 0; i < length; i++)
    {
        isNegative = (sequence[i] < 0);
        if(isNegative) {break;}
    }
    return(isNegative);
}
```

2.3.1 Критерий индуктивности функции (1)

$$\exists g \in \mathrm{func}(Y \times X, Y)\ \ f(w \parallel x) = g(f(w), x) \Leftrightarrow$$

$$\Leftrightarrow \forall u, v \in X^*, x \in X\ f(u) = f(v) \Rightarrow f(u \parallel x) = f(v \parallel x).$$

1. Докажем необходимость.

2.3.1 Критерий индуктивности функции (2)

$$\exists g \in \mathrm{func}(Y \times X, Y)\ \ f(w \parallel x) = g(f(w), x) \Leftrightarrow$$

$$\Leftrightarrow \forall u, v \in X^*, x \in X\ f(u) = f(v) \Rightarrow f(u \parallel x) = f(v \parallel x).$$

1. Докажем необходимость.
2. $\sqsupset \begin{cases} \exists g \in \mathrm{func}(Y \times X, Y)\ \ f(w \parallel x) = g(f(w), x), \\ f(u) = f(v); \end{cases}$

2.3.1 Критерий индуктивности функции (3)

$$\exists g \in \mathrm{func}(Y \times X, Y)\ \ f(w \parallel x) = g(f(w), x) \Leftrightarrow$$

$$\Leftrightarrow \forall u, v \in X^*, x \in X\ f(u) = f(v) \Rightarrow f(u \parallel x) = f(v \parallel x).$$

1. Докажем необходимость.
2. $\sqsupset \begin{cases} \exists g \in \mathrm{func}(Y \times X, Y)\ \ f(w \parallel x) = g(f(w), x), \\ f(u) = f(v); \end{cases}$
3. $f(u \parallel x) = g(f(u), x) = g(f(v), x) = f(v \parallel x)$, ч.т.д.

2.3.1 Критерий индуктивности функции (4)

$$\exists g \in \mathrm{func}(Y \times X, Y)\ f(w \parallel x) = g(f(w), x) \Leftrightarrow$$

$$\Leftrightarrow \forall u, v \in X^*, x \in X\ f(u) = f(v) \Rightarrow f(u \parallel x) = f(v \parallel x).$$

1. Докажем необходимость.
2. $\sqsupset \begin{cases} \exists g \in \mathrm{func}(Y \times X, Y)\ f(w \parallel x) = g(f(w), x), \\ f(u) = f(v); \end{cases}$
3. $f(u \parallel x) = g(f(u), x) = g(f(v), x) = f(v \parallel x)$, ч.т.д.
4. Докажем достаточность.

2.3.1 Критерий индуктивности функции (5)

$$\exists g \in \mathrm{func}(Y \times X, Y) \;\; f(w \parallel x) = g(f(w), x) \Leftrightarrow$$

$$\Leftrightarrow \forall u, v \in X^*, x \in X \; f(u) = f(v) \Rightarrow f(u \parallel x) = f(v \parallel x).$$

1. Докажем необходимость.
2. $\sqsupset \begin{cases} \exists g \in \mathrm{func}(Y \times X, Y) \;\; f(w \parallel x) = g(f(w), x), \\ f(u) = f(v); \end{cases}$
3. $f(u \parallel x) = g(f(u), x) = g(f(v), x) = f(v \parallel x)$, ч.т.д.
4. Докажем достаточность.
5. $\sqsupset \mathrm{func}(Y \times X, Y) \ni g(y, x) = \left[\begin{array}{ll} f(w \parallel x), & \exists w \; f(w) = y, \\ y, & \forall w \; f(w) \neq y; \end{array}\right.$

2.3.1 Критерий индуктивности функции (6)

$$\exists g \in \mathrm{func}(Y \times X, Y)\ f(w \parallel x) = g(f(w), x) \Leftrightarrow$$

$$\Leftrightarrow \forall u, v \in X^*, x \in X\ f(u) = f(v) \Rightarrow f(u \parallel x) = f(v \parallel x).$$

1. Докажем необходимость.
2. $\sqsupset \begin{cases} \exists g \in \mathrm{func}(Y \times X, Y)\ f(w \parallel x) = g(f(w), x), \\ f(u) = f(v); \end{cases}$
3. $f(u \parallel x) = g(f(u), x) = g(f(v), x) = f(v \parallel x)$, ч.т.д.
4. Докажем достаточность.
5. $\sqsupset \mathrm{func}(Y \times X, Y) \ni g(y, x) = \left[\begin{array}{ll} f(w \parallel x), & \exists w\ f(w) = y, \\ y, & \forall w\ f(w) \neq y; \end{array}\right.$
6. $g(y, x)\ |_{y=f(u)} = g(f(u), x) = f(u \parallel x);$

2.3.1 Критерий индуктивности функции (7)

$$\exists g \in \mathrm{func}(Y \times X, Y)\ \ f(w \parallel x) = g(f(w), x) \Leftrightarrow$$

$$\Leftrightarrow \forall u, v \in X^*, x \in X\ f(u) = f(v) \Rightarrow f(u \parallel x) = f(v \parallel x).$$

1. Докажем необходимость.
2. $\sqsupset \begin{cases} \exists g \in \mathrm{func}(Y \times X, Y)\ \ f(w \parallel x) = g(f(w), x), \\ f(u) = f(v); \end{cases}$
3. $f(u \parallel x) = g(f(u), x) = g(f(v), x) = f(v \parallel x)$, ч.т.д.
4. Докажем достаточность.
5. $\sqsupset \mathrm{func}(Y \times X, Y) \ni g(y, x) = \left[\begin{array}{ll} f(w \parallel x), & \exists w\ f(w) = y, \\ y, & \forall w\ f(w) \neq y; \end{array}\right.$
6. $g(y, x)\,|_{y=f(u)} = g(f(u), x) = f(u \parallel x);$
7. $g(y, x)\,|_{y=f(v)} = g(f(v), x) = f(v \parallel x);$

2.3.1 Критерий индуктивности функции (8)

$$\exists g \in \mathrm{func}(Y \times X, Y)\ f(w \parallel x) = g(f(w), x) \Leftrightarrow$$

$$\Leftrightarrow \forall u, v \in X^*, x \in X\ f(u) = f(v) \Rightarrow f(u \parallel x) = f(v \parallel x).$$

1. Докажем необходимость.
2. $\sqsupset \begin{cases} \exists g \in \mathrm{func}(Y \times X, Y)\ f(w \parallel x) = g(f(w), x), \\ f(u) = f(v); \end{cases}$
3. $f(u \parallel x) = g(f(u), x) = g(f(v), x) = f(v \parallel x)$, ч.т.д.
4. Докажем достаточность.
5. $\sqsupset \mathrm{func}(Y \times X, Y) \ni g(y, x) = \left[\begin{array}{ll} f(w \parallel x), & \exists w\ f(w) = y, \\ y, & \forall w\ f(w) \neq y; \end{array}\right.$
6. $g(y, x)\,|_{y=f(u)} = g(f(u), x) = f(u \parallel x);$
7. $g(y, x)\,|_{y=f(v)} = g(f(v), x) = f(v \parallel x);$
8. $\sqsupset f(u) = f(v);$

2.3.1 Критерий индуктивности функции (9)

$$\exists g \in \mathrm{func}(Y \times X, Y)\ \ f(w \parallel x) = g(f(w), x) \Leftrightarrow$$

$$\Leftrightarrow \forall u, v \in X^*, x \in X\ f(u) = f(v) \Rightarrow f(u \parallel x) = f(v \parallel x).$$

1. Докажем необходимость.
2. $\sqsupset \begin{cases} \exists g \in \mathrm{func}(Y \times X, Y)\ \ f(w \parallel x) = g(f(w), x), \\ f(u) = f(v); \end{cases}$
3. $f(u \parallel x) = g(f(u), x) = g(f(v), x) = f(v \parallel x)$, ч.т.д.
4. Докажем достаточность.
5. $\sqsupset \mathrm{func}(Y \times X, Y) \ni g(y, x) = \left[\begin{array}{ll} f(w \parallel x), & \exists w\ f(w) = y, \\ y, & \forall w\ f(w) \neq y; \end{array}\right.$
6. $g(y, x)\ |_{y=f(u)} = g(f(u), x) = f(u \parallel x);$
7. $g(y, x)\ |_{y=f(v)} = g(f(v), x) = f(v \parallel x);$
8. $\sqsupset f(u) = f(v);$
9. $f(u \parallel x) = g(y, x)\ |_{y=f(u)} = g(y, x)\ |_{y=f(v)} = f(v \parallel x)$, ч.т.д.

2.3.2 Построение индуктивного расширение функции

- **Определение**. Индуктивное расширение функции f – индуктивная функция F такая, что существует функция h, для которой справедливо равенство

$$f(w) = h\big(F(w)\big).$$

- Обычно индуктивное расширение имеет вид векторной функции

$$F(w) = \big(f(w), f_1(w), \dots, f_n(w)\big),$$

где $f_j, j \in \overline{1, n}$ – функции, через значения которых вычисляется $f(w \parallel x)$. При таком задании h представляет собой операцию проекции на первую координату набора $\big(f(w), f_1(w), \dots, f_n(w)\big)$.

Индуктивное расширение функции проверки строгого возрастания последовательности чисел (1)

$$F(w) = (\text{is\_increase}(w), \text{last}(w)).$$

1. $\sqsupset \text{is\_increase}(w) = 1 \Rightarrow \max(w) = \text{last}(w);$

Индуктивное расширение функции проверки строгого возрастания последовательности чисел (2)

$$F(w) = (\text{is\_increase}(w), \text{last}(w)).$$

1. $\sqsupset \text{is\_increase}(w) = 1 \Rightarrow \max(w) = \text{last}(w)$;
2. $\text{is\_increase}(w \parallel x) = 1 \Leftrightarrow x > \max(w) = \text{last}(w)$;

Индуктивное расширение функции проверки строгого возрастания последовательности чисел (3)

$$F(w) = (\mathrm{is\_increase}(w), \mathrm{last}(w)).$$

1. $\sqsupset \mathrm{is\_increase}(w) = 1 \Rightarrow \max(w) = \mathrm{last}(w)$;
2. $\mathrm{is\_increase}(w \parallel x) = 1 \Leftrightarrow x > \max(w) = \mathrm{last}(w)$;
3. $\mathrm{is\_increase}(w \parallel x) = \mathrm{is\_increase}(w) \wedge (x > \mathrm{last}(w))$;

Индуктивное расширение функции проверки строгого возрастания последовательности чисел (4)

$$F(w) = (\mathrm{is\_increase}(w), \mathrm{last}(w)).$$

1. $\sqsupset \mathrm{is\_increase}(w) = 1 \Rightarrow \max(w) = \mathrm{last}(w)$;
2. $\mathrm{is\_increase}(w \parallel x) = 1 \Leftrightarrow x > \max(w) = \mathrm{last}(w)$;
3. $\mathrm{is\_increase}(w \parallel x) = \mathrm{is\_increase}(w) \wedge (x > \mathrm{last}(w))$;
4. $\sqsupset F(w) = (\mathrm{is\_increase}(w), \mathrm{last}(w))$;

Индуктивное расширение функции проверки строгого возрастания последовательности чисел (5)

$$F(w) = (\mathrm{is\_increase}(w), \mathrm{last}(w)).$$

1. $\sqsupset \mathrm{is\_increase}(w) = 1 \Rightarrow \max(w) = \mathrm{last}(w);$
2. $\mathrm{is\_increase}(w \parallel x) = 1 \Leftrightarrow x > \max(w) = \mathrm{last}(w);$
3. $\mathrm{is\_increase}(w \parallel x) = \mathrm{is\_increase}(w) \wedge (x > \mathrm{last}(w));$
4. $\sqsupset F(w) = (\mathrm{is\_increase}(w), \mathrm{last}(w));$
5. $F(w \parallel x) = (\mathrm{is\_increase}(w \parallel x), \mathrm{last}(w \parallel x)) = (\mathrm{is\_increase}(w) \wedge (x > \mathrm{last}(w)), x);$

Индуктивное расширение функции проверки строгого возрастания последовательности чисел (6)

$$F(w) = (\text{is\_increase}(w), \text{last}(w)).$$

1. $\sqsupset$ $\text{is\_increase}(w) = 1 \Rightarrow \max(w) = \text{last}(w)$;
2. $\text{is\_increase}(w \parallel x) = 1 \Leftrightarrow x > \max(w) = \text{last}(w)$;
3. $\text{is\_increase}(w \parallel x) = \text{is\_increase}(w) \wedge (x > \text{last}(w))$;
4. $\sqsupset$ $F(w) = (\text{is\_increase}(w), \text{last}(w))$;
5. $F(w \parallel x) = (\text{is\_increase}(w \parallel x), \text{last}(w \parallel x)) = (\text{is\_increase}(w) \wedge (x > \text{last}(w)), x)$;
6. $\exists g(F(w), x) = g(\text{is\_increase}(w), \text{last}(w), x) = (\text{is\_increase}(w) \wedge (x > \text{last}(w)), x) =$
   $= F(w \parallel x)$, ч.т.д.

Спасибо за внимание!

Какие у Вас есть вопросы?

Лектор: ст. преп. каф. ИБ
Абросимов Иван Константинович