ОБОБЩЕННАЯ ВЫЧИСЛИМОСТЬ И ОПРЕДЕЛИМОСТЬ
(Вычислительные системы)
1999 год Выпуск 165

УДК 510.51

Абстрактный функционал, ограничивающий алгоритмическую сложность для линейного, полиномиального и других схожих классов[1]

Е.И. Латкин

Введение

Основная идея работы — абстрактная мера алгоритмической сложности — восходит к классам $\mathcal{E}^0 \subset \mathcal{E}^0 \subset \ldots$ А.Гжегорчика [4], исчерпывающим класс примитивнорекурсивных функций $\mathcal{R}$. Каждый класс $\mathcal{E}^n$ содержит специально определяемую в [4] функцию $f_n(x, y)$, которая задает его "сложность", и замкнут относительно операции ограниченной примитивной рекурсии:

$$\begin{cases} h(0, x_1, \ldots, x_n) = g_0(x_1, \ldots, x_n), \\ h(y+1, x_1, \ldots, x_n) = g_1(h(y, x), y, x_1, \ldots, x_n), \\ h(y, x_1, \ldots, x_n) \leq f(y, x_1, \ldots, x_n). \end{cases}$$

Здесь функция $h(y, x_1, \ldots, x_n)$ определяется при помощи оператора примитивной рекурсии из функций $g_0, g_1 \in \mathcal{E}^n$, но возможно $h \notin \mathcal{E}^n$, если $h(y, x_1, \ldots, x_n) > f(y, x_1, \ldots, x_n)$ при некоторых $x_1, \ldots, x_n, y \in \mathbb{N}$. Иными словами, оператор ограниченной рекурсии не определён для таких g_0, g_1, при которых h не мажорируется некоторой функцией $f \in \mathcal{E}^n$

[1]Работа выполнена при поддержке гранта РФФИ № 99-01-00485.

Мы развиваем подобную идею, однако при этом ограничиваем некоторую меру сложности алгоритма $|h(x_1,\dots,x_n)|$ вместо самой функции $h(x_1,\dots,x_n)$, вычисляемой "алгоритмом" h.

Слово "алгоритм" здесь написано в кавычках потому, что, вообще говоря, не существует алгоритма, ассоциированного с функцией h в рассмотренном выше примере с классами Гжегорчика. Вместо реальных алгоритмов, мы рассматриваем абстрактную алгебраическую систему $\langle \mathcal{A}; \mathsf{R}, \mathsf{M}, \circ, \dots\rangle$, сигнатура которой содержит символы примитивной рекурсии, монотонной инверсии (определяемый ниже вариант оператора минимизации), а также суперпозиции и других операторов из сигнатуры клона, описанных ниже (см. также [1]). Затем, мы рассматриваем произвольный гомоморфизм $\kappa : \mathcal{A} \to \mathcal{F}$ алгебры $\mathcal{A}$ в клон $\mathcal{F}$ функций над натуральными числами. Элементы $h \in \mathcal{A}$ мы считаем абстрактными аналогами "алгоритмов".

Ввиду того, что подразумеваемый гомоморфизм $\kappa : \mathcal{A} \to \mathcal{F}$ будет очевиден, мы часто будем опускать символ κ и писать $h(x_1, \dots, x_n)$ вместо правильной записи $(\kappa(h))\,(x_1, \dots, x_n)$.

В качестве функционала сложности мы используем произвольное отображение $h \in \mathcal{A} \mapsto |h| \in \mathcal{F}$, удовлетворяющее определённому списку аксиом, приведённых ниже. Основная из этих аксиом требует, чтобы сложность суперпозиции двух функций $f \circ g$ можно было оценить сверху через сложность алгоритмов f и g:

$$|f(g(x_1, \dots, x_n), y_1, \dots, y_m)| \preceq_C$$
$$\preceq_C |f|\,(g(x_1, \dots, x_n), y_1, \dots, y_m) \oplus |g(x_1, \dots, x_n)|\,.$$

Символ $\oplus$ означает "суммарное время вычислений", или "максимальное количество использованной памяти"; также это может быть любая бинарная операция, удовлетворяющая специальным аксиомам. Символ $\preceq_C$ означает сравнение с "точностью до класса C", например с точностью до класса функций, растущих линейно, или полиномиально. Ниже приводится список аксиом для класса C.

Такая конструкция может показаться громоздкой по сравнению с иерархией Гжегорчика, которая получается как частный случай абстрактных алгоритмов, если положить $\mathcal{A} = \mathcal{F}$ и $\kappa = |\cdot| = \mathrm{id}_{\mathcal{F}}$. Однако абстрактная мера сложности позволяет удобно оценивать сложность алгоритмов в стиле функционального анализа: достаточно выстроить последовательность неравенств $|h(x_1, ..., x_n)| \preceq_C ... \preceq_C p(x_1, ..., x_n)$ с подходящей мажорантой $p(x_1, ..., x_n)$ из линейного, или, например, полиномиального класса.

Изложение в этой статье следует от общего к частному.

Мы начинаем с определений абстрактных понятий алгебры "алгоритмов" и меры сложности. Затем, в наиболее общей постановке, доказывается основная теорема — о неавтоустойчивости бесконечных "эффективно конструктивных" моделей. Наконец, мы показываем, что обсуждаемые здесь объекты действительно существуют.

1. Алгебра абстрактных алгоритмов

Здесь мы определяем понятие "класс алгоритмов" как абстрактную алгебру, снабжённую специфическим гомоморфизмом в алгебру функций над множеством натуральных чисел $\mathbb{N}$. Но сначала мы определим, какие классы функций мы будем рассматривать.

Пусть запись "$X := Y$" означает определение через присваивание: "X определяется как Y". Ниже мы будем использовать векторную запись $\overline{x}$ вместо $x_1, ..., x_n$, когда специфическое количество элементов n в списке $\overline{x}$ не важно.

ОПРЕДЕЛЕНИЕ 1. Полагаем $\mathcal{F} := \bigcup_{n \in \mathbb{N}} \mathcal{F}_n$, где $\mathcal{F}_n := \{f \mid f : \mathbb{N}^n \to \mathbb{N}\}$ — множество всюду определённых n-арных функций. Частичные операторы $\circ$, η, ξ, Δ, π, R и M над множеством $\mathcal{F}$ определяются как:

1) $f \circ g : (\overline{x}, \overline{y}) \mapsto f(g(\overline{x}), \overline{y})$ — суперпозиция по первому аргументу;

2) $\eta(f) : (x, \overline{y}) \mapsto f(\overline{y}, x)$ — ротация аргументов функции;

3) $\xi(f) : (x, y, \overline{z}) \mapsto f(y, x, \overline{z})$ — перестановка первых двух аргументов;

4) $\Delta(f) : (x, \overline{y}) \mapsto f(x, x, \overline{y})$ — диагонализация;

5) $\pi(f) : (x, \overline{y}) \mapsto f(\overline{y})$ — добавка незначащего аргумента;

6) $h = \mathsf{R}(f, g)$ — примитивная рекурсия, определяемая как:

$$\begin{cases} h(0, \overline{x}) = f(\overline{x}), \\ h(y+1, \overline{x}) = g(h(y, \overline{x}), y, \overline{x}); \end{cases}$$

7) $g = \mathsf{M}(f)$ — монотонное обращение, определяемое как: $g(z, \overline{x}) = \inf\{y |\ f(y, \overline{x}) \geq z\}$.

Символы операторов $\{\circ, \eta, \xi, \Delta, \pi\}$ образуют сигнатуру клона [1]. Иными словами, мы можем рассматривать $\mathcal{F}$ как клон операций, обогащённый операторами R и M. Нам также придётся обогатить эту сигнатуру константными символами определяемых ниже функций (см. определение 2) из класса $\mathcal{F}$.

ОПРЕДЕЛЕНИЕ 2. Для подмножества $\mathcal{X} \subseteq \mathcal{F}$ мы пишем $\mathcal{X} \sqsubseteq \mathcal{F}$, если $\mathcal{X}$ замкнуто относительно операторов $\circ$, η, ξ, Δ, π и содержит следующие функции:

1) $\mathbf{zero} := 0$ константа, т.е. $\mathbf{zero} \in \mathcal{F}_0$;
2) $\mathbf{succ}(x) := x + 1$;
3) $\mathbf{id}(x) := x$;
4) $\mathbf{case}(x, y, z) :=$ "если $x = 0$, то y, иначе z";
5) $\mathbf{eq}(x, y) :=$ "если $x = y$, то 0, иначе 1";
6) $\mathbf{lt}(x, y) :=$ "если $x < y$, то 0, иначе 1";
7) $\mathbf{and}(x, y) :=$ "если $x = 0 \wedge y = 0$, то 0, иначе 1";
8) $\mathbf{not}(x) :=$ "если $x \neq 0$, то 0, иначе 1".

Заметим, что здесь 0 означает "истина", а 1 означает "ложь".

Иначе говоря, $\mathcal{X} \sqsubseteq \mathcal{F}$ если $\mathcal{X} \subseteq \mathcal{F}$ и $\mathcal{X}$ является подалгеброй алгебры $\mathcal{F}$ в сигнатуре $\mathcal{S} := \{\circ, \eta, \xi, \Delta, \pi, \mathbf{zero}, \mathbf{succ}, \mathbf{id}, \mathbf{case}, \mathbf{eq}, \mathbf{lt}, \mathbf{and}, \mathbf{not}\}$.

Предположим, что $\mathcal{A}$ — произвольная алгебра сигнатуры $\mathcal{S} \cup \{\mathsf{R}, \mathsf{M}\}$, рассматриваемая вместе с некоторым гомоморфизмом $\kappa : \mathcal{A} \to \mathcal{F}$ этой сигнатуры.

Будем называть элементы множества $\mathcal{A}$ "абстрактными алгоритмами", или просто "алгоритмами". Положим $\mathcal{A}_n := \kappa^{-1}[\mathcal{F}_n]$. Для $f \in \mathcal{A}_n$ мы будем как правило писать $f(x_1, ..., x_n)$ вместо $(\kappa f)(x_1, ..., x_n)$, отождествляя алгоритм f с вычисляемой им функцией.

Например, $\mathcal{A}$ может быть множеством вычислительных машин определённого рода, определяемого ниже. Отображением κ здесь служит естественная интерпретация, когда $(\kappa f)(x_1, ..., x_n)$ означает результат применения машины f к аргументам $x_1, ..., x_n$.

С другой стороны, мы можем положить $\mathcal{A} := \mathcal{F}$ и $\kappa := \mathrm{id}_{\mathcal{F}}$. Таким способом можно проследить аналогию между определяемыми ниже "наследственно эффективными"классами и классами Гжегорчика.

ОПРЕДЕЛЕНИЕ 3. Пусть $\mathcal{B} \subseteq \mathcal{A}$. Будем писать $\mathcal{B} \sqsubseteq \mathcal{A}$ и называть множество $\mathcal{B}$ *классом* алгоритмов, если $\mathcal{B}$ является подалгеброй алгебры $\mathcal{A}$ в сигнатуре $\mathcal{S}$. (Класс не обязан быть замкнут относительно операций примитивной рекурсии R и монотонного обращения M.)

Во всей статье, за исключением специальной секции "Примеры...", мы работаем с абстрактным универсумом алгоритмов $\kappa : \mathcal{A} \to \mathcal{F}$, не раскрывая его конкретную природу. Таким образом, все обсуждаемые здесь подклассы алгоритмов в $\mathcal{A}$ являются абстрактными.

2. Абстрактная мера алгоритмической сложности

В этом параграфе мы описываем класс $\mathcal{C} \sqsubseteq \mathcal{F}$ "хороших" функций и абстрактную "меру сложности" $|\cdot| : \mathcal{A} \to \mathcal{F}$. С их помощью определяется класс $\mathcal{E} := \{f \in \mathcal{A} |\ |f| \in \mathcal{C}\}$ "эффективных" алгоритмов, и ниже вводится его более сложно определяемый подкласс $\mathcal{H}$ "наследственно эффективных" алгоритмов.

Способ спецификации этих объектов такой же, как для класса $\mathcal{A}$. Мы просто перечисляем список аксиом, сопроводив аксиомы некоторыми простыми примерами и пояснениями, показывающими их состоятельность. Затем мы говорим: "Зафиксируем некоторый класс $\mathcal{C}$ и отображение $|\cdot|$, удовлетворяющие этим аксиомам."

Ниже перечисленны 4 группы аксиом для класса $\mathcal{C}$ и отображения $|\cdot|$. Удобно иметь ввиду следующий пример, демонстрирующий их совместность:

A) $\mathcal{A}$ есть множество машин Тьюринга;

B) $\mathcal{C}$ — множество всех функций $r(x_1, ..., x_n)$, каждая из которых ограничена некоторым полиномом от $x_1, ..., x_n$;

C) для всех $f \in \mathcal{A}$ и $x_1, ..., x_n \in \mathbb{N}$ мера $|f(x_1, ..., x_n)| = |f|(x_1, ..., x_n)$ совпадает со временем вычисления $f(x_1, ..., x_n)$;

D) бинарная операция "$\oplus$", упоминаемая в аксиомах, совпадает с обычным сложением "$+$";

E) упоминаемая ниже функция $p(x) := (x+2)^2$.

Аксиома 1. Существует такая бинарная функция $\oplus \in C$, что:

1) $x \oplus (y \oplus z) = (x \oplus y) \oplus z$;

2) $x \oplus y = y \oplus x$;

3) $x \leq x \oplus y$;

4) $x_1 \leq x_2 \wedge y_1 \leq y_2 \Rightarrow x_1 \oplus y_1 \leq x_2 \oplus y_2$.

Аксиома 2. Найдётся такая функция $p \in C$, что:

1) $x < y \Rightarrow p(x) < p(y)$;

2) $x < p(x)$;

3) $(\forall f \in C)\ \exists k\ (\forall x_1, ..., x_n)\ f(x_1, ..., x_n) \leq p^k(x_1 \oplus ... \oplus x_n)$;

4) $x \oplus x \leq p(x)$.

Здесь и далее мы полагаем $p^k(x) := p(p(...p(x)...)) = \overbrace{p \circ ... \circ p}^{k \text{ раз}}(x)$ если $k > 0$, и $p^k(x) := x$ если $k = 0$.

Нам понадобятся следующие определения.

ОПРЕДЕЛЕНИЕ 4. Пусть $f, g \in \mathcal{F}_n$ и $C \sqsubseteq \mathcal{F}$. Определим отношения $\preceq$ и $\preceq_C$ как:

1) $f \preceq g \Leftrightarrow \exists x_0\ (\forall x_1, ...x_n \geq x_0)\ f(x_1, ...x_n) \leq g(x_1, ...x_n)$;

2) $f \preceq_C g \Leftrightarrow (\exists h \in C)\ f \preceq g \oplus h$, где $(g \oplus h)(\overline{x}) := g(\overline{x}) \oplus h(\overline{x})$.

ОПРЕДЕЛЕНИЕ 5. Функция $f : \mathbb{N}^{n+1} \to \mathbb{N}$ называется 1–*монотонной*, если $\forall \overline{x}\ (\forall y', y'')\ [y' \leq y'' \Rightarrow f(y', \overline{x}) \leq f(y'', \overline{x})]$.

Функция $f : \mathbb{N}^{n+1} \to \mathbb{N}$ называется *строго* 1–монотонной, если $\forall \overline{x}\ (\forall y', y'')\ [y' < y'' \Rightarrow f(y', \overline{x}) < f(y'', \overline{x})]$.

Одноместная функция $f : \mathbb{N} \to \mathbb{N}$ называется (*строго*) *монотонной*, если она (строго) 1 монотонна.

ПРЕДЛОЖЕНИЕ 1. *Отношения* $\preceq$ *и* $\preceq_C$ *транзитивны и рефлексивны, и из* $f \preceq g$ *следует* $f \preceq_C g$.

Это предложение очевидно.

Аксиома 3. Для класса C справедливы следующие импликации:

1) если $f \in C$ и $g(y, \overline{x}) = f(0, \overline{x}) \oplus ... \oplus f(y, \overline{x})$, то $g \in C$;

2) если $f \in C$ и $g \preceq f$, то $g \in C$.

Аксиома 4. Мера сложности $|\cdot| : \mathcal{A} \to \mathcal{F}$ такова, что:

1) $|f \circ g| \preceq_{\mathcal{C}} (|f| \circ g) \oplus |g|$;

2) если $f = \mathsf{R}(g, h)$, то

$$\begin{cases} |f|\,(0,\overline{x}) \preceq_{\mathcal{C}} |g|\,(\overline{x}), \\ |f|\,(y+1,\overline{x}) \preceq_{\mathcal{C}} |h|\,(f(y,\overline{x}),y,\overline{x}) \oplus |f|\,(y,\overline{x}); \end{cases}$$

3) если $f = \mathsf{M}(g)$, то

$$|f(y,\overline{x})| \preceq_{\mathcal{C}} \bigoplus_{z=0}^{f(y,\overline{x})} |g(z,\overline{x})| \oplus f(y,\overline{x});$$

4) $|\mathbf{zero}|, |\mathbf{succ}|, |\mathbf{id}|, |\mathbf{eq}|, |\mathbf{lt}|, |\mathbf{and}|, |\mathbf{not}|, |\mathbf{case}| \in \mathcal{C}$;

4′) $|\eta(f)| \preceq_{\mathcal{C}} \eta\,(|f|)$,
$|\xi(f)| \preceq_{\mathcal{C}} \xi\,(|f|)$,
$|\Delta(f)| \preceq_{\mathcal{C}} \Delta\,(|f|)$,
$|\pi(f)| \preceq_{\mathcal{C}} \pi\,(|f|)$;

5) $(\exists p' \in \mathcal{A})\ [\kappa(p') = p$ и $|p'| \in \mathcal{C}]$;

6) $\kappa(f) \preceq_{\mathcal{C}} |f|$.

Аксиома 4.1 здесь основная. Она требует, чтобы затраты на вычисление $(f \circ g)(\overline{x},\overline{y})$ не превышали затрат на вычисление $f(z,\overline{y})$ в точке $z = g(\overline{y})$ "плюс" затраты на вычисление $g(\overline{y})$. Некоторые накладные расходы вычислительных ресурсов на состыковку двух алгоритмов учитываются подиндексом $\mathcal{C}$ в оценке $\preceq_{\mathcal{C}}$.

Мы не требуем, чтобы алгебра $\mathcal{A}$ была свободной. Возможно, один и тот же алгоритм $h \in \mathcal{A}$ может быть получен при помощи другой суперпозиции $h = f' \circ g'$. В этом случае, должны выполняться оба соотношения $|h| \preceq_{\mathcal{C}} (|f| \circ g) \oplus |g|$ и $|h| \preceq_{\mathcal{C}} (|f'| \circ g') \oplus |g'|$.

Аксиомы 4.4 и 4.4′ утверждают тривиальную сложность констант **zero**, **succ**, **id**, **eq**, **lt**, **and**, **not**, **case** и простейших операторов η, ξ, Δ, π алгебры $\mathcal{A}$.

Аксиома 4.5 позволяет обогатить сигнатуру $\mathcal{S}$ новым константным символом $\mathbf{p}$, соответствующим алгоритму $p' \in \mathcal{A}$. Мы имеем $\kappa(\mathbf{p}) = p$ и $|\mathbf{p}| \in \mathcal{C}$. Мы не будем специально различать символ $\mathbf{p}$, алгоритм p' и функцию p, поскольку соотношение между ними очевидно.

Аксиомы 4.2 и 4.3 могли бы следовать из аксиомы 4.1, если иметь ввиду прямое вычисление функции f. Предположим, что $\mathsf{R}(g,h)$ вычисляется при помощи рекурсивных подстановок, а $\mathsf{M}(g)$ — последовательным перебором. Тогда мера сложности $|\mathsf{M}(g)|$ оценивается через суммарную сложность вычисления всех перебираемых вариантов, плюс затраты на поддержание счётчика итераций.

В качестве тривиального следствия из аксиомы 4.2, можно использовать подобную оценку и для примитивной рекурсии.

ЗАМЕЧАНИЕ 1. Если $f = \mathsf{R}(g,h)$, то для всех $y > 0$

$$|f(y,\overline{x})| \preceq_C |g(\overline{x})| \oplus \bigoplus_{z=0}^{y-1} |h| \left(f(z,\overline{x}), z, \overline{x}\right).$$

Обратимся к машинам Тьюринга, чтобы обосновать аксиому 4.6.

Только для того, чтобы напечатать результат вычислений, скажем число x, машине нужно не менее, чем $x+1$ шаг и $x+1$ ячеек памяти (ленты). Следовательно, ни временнáя сложность, ни сложность по памяти не может быть меньше, чем вычисляемый машиной результат.

Предположим, что зафиксированы некоторый класс $\mathcal{C}$ и мера сложности $|\cdot|$. Тогда будет естественно определить множество $\mathcal{E}$ "эффективных" алгоритмов как $\{f \in \mathcal{A} | \ |f| \in \mathcal{C}\}$. Определим также множество $\mathcal{H}$ "наследственно эффективных" алгоритмов.

ОПРЕДЕЛЕНИЕ 6. Множество "наследственно эффективных" алгоритмов есть наименьший класс $\mathcal{H} \sqsubseteq \mathcal{A}$, содержащий алгоритм p и замкнутый относительно ограниченных операторов примитивной рекурсии R_0 и монотонного обращения M_0, определяемых как:

1) $\mathsf{R}_0(g,h) := \mathsf{R}(g,h)$, если алгоритм $f = \mathsf{R}(g,h)$ определён и эффективен, т.е. если $(g,h) \in \mathrm{dom}(\mathsf{R})$ и $f \in \mathcal{E}$. В противном случае, значение $\mathsf{R}_0(g,h)$ не определено;

2) аналогично, $\mathsf{M}_0(g) := \mathsf{M}(g)$ определено, если и только если $f = \mathsf{M}(g)$ определено и $f \in \mathcal{E}$.

Это означает, что класс $\mathcal{H}$ не обязан содержать функцию $\mathsf{R}(g,h)$ или $\mathsf{M}(g)$, если она не эффективна, даже если $g, h \in \mathcal{H}$.

Множество $\mathcal{H}$ является классом по определению. Покажем, что $\mathcal{E}$ тоже класс.

ПРЕДЛОЖЕНИЕ 2. *Множество эффективных алгоритмов $\mathcal{E}$ является классом, т.е. $\mathcal{E} \sqsubseteq \mathcal{A}$.*

ДОКАЗАТЕЛЬСТВО. Достаточно показать, что $\mathcal{E}$ замкнуто по суперпозиции. Пусть $f, g \in \mathcal{E}$, т.е. $|f|, |g| \in \mathcal{C}$. Покажем, что $|f \circ g| \in \mathcal{C}$. Положим $h := f \circ g$; по аксиоме 4.1, $|h| \preceq_{\mathcal{C}} (|f| \circ g) \oplus |g|$. По аксиоме 4.6, $\kappa(g) \preceq_{\mathcal{C}} |g|$. Но $|g| \in \mathcal{C}$; следовательно, по аксиоме 3.2, $\kappa(g) \in \mathcal{C}$. Поскольку множество $\mathcal{C}$ является классом по определению, оно замкнуто относительно суперпозиции функций. Напомним, что $\oplus \in \mathcal{C}$. Значит, из $|h| \preceq_{\mathcal{C}} (|f| \circ \kappa(g)) \oplus |g|$ следует $|h| \in \mathcal{C}$. ■

3. Теорема о неавтоустойчивости

Здесь мы доказываем основную теорему данной работы, утверждающую, что не существует бесконечных $\mathcal{B}$–автоустойчивых конструктивных моделей. Понятия $\mathcal{B}$–автоустойчивости и $\mathcal{B}$ конструктивизируемости определяются естественным образом исходя из некоторого класса алгоритмов $\mathcal{B}$.

Предположим, что нам задан абстрактный класс алгоритмов $\mathcal{B} \sqsubseteq \mathcal{A}$, вместе с некоторыми 1–местными алгоритмами $Q \in \mathcal{A}$ и $R \in \mathcal{B}$ такими, что:

1) $(\forall f \in \mathcal{B})$ [алгоритм $(z_1, ..., z_n) \mapsto Qf(Rz_1, ..., Rz_n)$ попадает в класс $\mathcal{B}$];

2) $(\forall f \in \mathcal{B}_1)\ f \preceq Q$ — т.е. $\exists x(\forall y \geq x)\ f(y) \leq Q(y)$;

3) $R \circ Q = \mathrm{id}_{\mathbb{N}}$, и $Q \circ R \preceq s$ для некоторой монотонной функции $s \in \mathcal{B}$;

4) функция, вычисляемая R является монотонной.

В следующих двух параграфах мы докажем, что классы $\mathcal{E}$ эффективных и $\mathcal{H}$ наследственно эффективных алгоритмов удовлетворяют вышеперечисленным условиям 1–4, если рассматривать их вместе с некоторыми построенными ниже функциями Q и R. Поэтому, следующая далее теорема о неавтоустойчивости применима к классам $\mathcal{E}$ и $\mathcal{H}$.

ОПРЕДЕЛЕНИЕ 7. Пусть **A** - произвольная счётная модель некоторой сигнатуры σ. Нумерация $\alpha : \mathbb{N} \to A$ носителя модели **A** называется *$\mathcal{B}$–конструктивизацией*, если:

1) отображение α сюрьективно, т.е. $(\forall a \in A)(\exists n \in \mathbb{N})$ $a = \alpha(n)$;

2) для каждого функционального символа $F \in \sigma$ существует такой алгоритм $\widetilde{F} \in \mathcal{B}$, что: $\mathbf{A} \models F(\alpha x_1, \ldots, \alpha x_n) = \alpha\widetilde{F}(x_1, \ldots, x_n)$;

3) для каждого предикатного символа $P \in \sigma$ найдётся такой алгоритм $\widetilde{P} \in \mathcal{B}$, что: $\mathbf{A} \models P(\alpha x_1, \ldots, \alpha x_n)$ если и только если $\widetilde{P}(x_1, \ldots, x_n) = 0$;

4) $(\exists \widetilde{e} \in \mathcal{B})$ $[\alpha x = \alpha y \Leftrightarrow \widetilde{e}(x, y) = 0]$ — т.е. равенство $\mathcal{B}$-разрешимо.

Заметим, что по этому определению $\mathrm{card}(A) = \infty$, и сигнатура σ также может быть бесконечной.

ОПРЕДЕЛЕНИЕ 8. Для произвольной пары α, α' $\mathcal{B}$-конструктивизаций модели $\mathbf{A}$ определим отношения $\mathcal{B}$-сводимости и $\mathcal{B}$-автоэквивалентности:

Мы пишем "$\alpha' \angle_{\mathcal{B}} \alpha$" и говорим, что "$\alpha'$ *$\mathcal{B}$-сводится к* α", если найдутся такие алгоритм $h \in \mathcal{B}$ и автоморфизм $\varphi \in \mathrm{Aut}(\mathbf{A})$, что $\varphi \circ \alpha' = \alpha \circ h$.

Будем писать "$\alpha' \equiv_{\mathcal{B}} \alpha$" и говорить, что "$\alpha'$ *$\mathcal{B}$-автоэквивалентно* α", если $\alpha' \angle_{\mathcal{B}} \alpha$ и в то же время $\alpha \angle_{\mathcal{B}} \alpha'$.

Счётная модель $\mathbf{A}$ называется *$\mathcal{B}$-автоустойчивой*, если она $\mathcal{B}$-конструктивизируема и любые две её $\mathcal{B}$-конструктивизации $\mathcal{B}$-автоэквивалентны.

Такая $\mathcal{B}$-автоустойчивость кажется естественным обобщением обычной автоустойчивости конструктивных моделей [2]. Тем не менее, следующая теорема показывает, что это понятие было бы бессодержательным.

ТЕОРЕМА (о неавтоустойчивости). *Пусть $\mathcal{B} \sqsubseteq \mathcal{A}$ — некоторый абстрактный класс алгоритмов, удовлетворяющий условиям* 1–4, *перечисленным в начале этого параграфа. Например, $\mathcal{B} = \mathcal{E}$, или $\mathcal{B} = \mathcal{H}$. Тогда, для любой бесконечной $\mathcal{B}$-конструктивной модели $(\mathbf{A}; \alpha)$ существует другая $\mathcal{B}$-конструктивизация α' модели $\mathbf{A}$ такая, что $\alpha' \angle_{\mathcal{B}} \alpha$, но $\alpha \not\angle_{\mathcal{B}} \alpha'$.*

ЗАМЕЧАНИЕ 2. Из теоремы следует, что для каждой бесконечной $\mathcal{B}$-конструктивной модели найдётся бесконечная цепочка $\ldots \alpha'' \angle_{\mathcal{B}} \alpha' \angle_{\mathcal{B}} \alpha$ не $\mathcal{B}$-автоэквивалентных конструктивизаций. Следовательно, любая счётная модель $\mathbf{A}$ либо не $\mathcal{B}$-конструктивизируема, либо имеет бесконечную $\mathcal{B}$–конструктивную размерность. Во всяком случае, не существует бесконечных $\mathcal{B}$–автоустойчивых моделей.

ДОКАЗАТЕЛЬСТВО теоремы.

Пусть $\alpha : \mathbb{N} \to A$ — некоторая $\mathcal{B}$–конструктивизация модели $\mathbf{A}$, и Q, R — алгоритмы, удовлетворяющие условиям 1–4, приведённым в начале параграфа.

Положим, $\alpha' := \alpha \circ R$ и покажем, что α' является $\mathcal{B}$–конструктивизацией модели $\mathbf{A}$.

1. α' сюрьективна.

Поскольку $R \circ Q = \mathrm{id}_{\mathbb{N}}$, по условию 3, то отображение R сюрьективно. Значит, $\alpha' = \alpha \circ R$ также сюрьективна.

2. Рассмотрим некоторый функциональный символ $F \in \sigma$.

Пусть $\widetilde{F} \in \mathcal{B}$ - такой алгоритм, что $F(\alpha x_1, \ldots, \alpha x_n) = \alpha\widetilde{F}(x_1, \ldots, x_n)$. Положим $\widetilde{F}'(z_1, \ldots, z_n) := Q\widetilde{F}(Rz_1, \ldots, Rz_n)$. Проверим условие 2, что $F(\alpha' z_1, \ldots, \alpha' z_n) = \alpha'\widetilde{F}'(z_1, \ldots, z_n)$:

$$F(\alpha' z_1, \ldots, \alpha' z_n) = F(\alpha(Rz_1), \ldots, \alpha(Rz_n)) = \ldots$$
$$\ldots = \alpha\widetilde{F}(Rz_1, \ldots, Rz_n) = \alpha \circ RQ \circ \widetilde{F}(Rz_1, \ldots, Rz_n) = \ldots$$
$$\ldots = \alpha' Q\widetilde{F}(Rz_1, \ldots, Rz_n) = \alpha'\widetilde{F}'(Rz_1, \ldots, Rz_n)$$

(мы воспользовались свойством $RQ = \mathrm{id}_{\mathbb{N}}$). Таким образом, функция $F \in \sigma$ над $\mathbf{A}$ является $\mathcal{B}$–вычислимой в нумерации α'.

3 и 4. Покажем $\mathcal{B}$–разрешимость любого предиката $P \in \sigma$ и отношения равенства элементов из $\mathbf{A}$.

Положим $\widetilde{P}'(z_1, \ldots, z_n) := \widetilde{P}(Rz_1, \ldots, Rz_n)$, и $\widetilde{e}'(y, x) := \widetilde{e}(Ry, Rx)$.

Тогда, $\mathbf{A} \models P(\alpha' z_1, \ldots, \alpha' z_n) \Leftrightarrow \widetilde{P}(Rz_1, \ldots, Rz_n) = \widetilde{P}'(z_1, \ldots, z_n) = 0$.

По определению, $\alpha' \angle_{\mathcal{B}} \alpha$. Покажем, что $\alpha \not\angle_{\mathcal{B}} \alpha'$.

Предположим противное, что найдётся такой алгоритм $h \in \mathcal{B}$ и такая перестановка $\varphi \in \mathbb{S}(A)$ множества A, что $\varphi \circ \alpha = \alpha' \circ h$. Постараемся получить противоречие с условием 2, утверждающим, что $(\forall u \in \mathcal{B})\ u \preceq Q$. (Множество $\mathbb{S}(A)$ всех перестановок множества A заведомо шире группы автоморфизмов $\mathrm{Aut}(\mathbf{A})$.)

Заметим, что условие 2 эквивалентно строгому неравенству $(\forall u \in \mathcal{B})\ \exists x(\forall y \geq x)\ u(y) < Q(y)$, поскольку $\forall y\ u(y) < u'(y)$, где $u'(y) := u(y) + 1$, и $u' \in \mathcal{B}$ если $u \in \mathcal{B}$.

Таким образом, достаточно показать, что $\forall x(\exists y \geq x)\ Q(y) \leq \leq u(y)$ для некоторого $u \in B$. Покажем это.

Для произвольных $x, x' \in \mathbb{N}$ положим $d(x) := \inf\{y | \alpha y = \alpha x\}$ и $d'(x') := \inf\{y' | \alpha' y' = \alpha' x'\}$. Обозначим также $X := \{x \in \mathbb{N} | x = = d(x)\}$ и $\alpha_X := \alpha | X$ – сужение α на X.

Поскольку $\alpha_X : X \to A$ биективно, найдётся такая перестановка $\psi_X \in \mathrm{S}(X)$, что $\varphi \circ \alpha_X = \alpha_X \circ \psi_X$. Обозначим $\psi := \psi_X \circ d$; тогда $d \circ \psi = \psi$.

Имеем: $\alpha(\psi x) = \alpha_X \psi(x) = \alpha_X \psi_X(dx) = \varphi \alpha_X(dx) = \varphi \alpha(x) = = \alpha'(hx)$.

По определению α' имеем: $\alpha y = \alpha' y' \Rightarrow dy \leq R(d'y')$, поскольку из $dy > R(d'y')$ следует $\alpha(dy) \neq \alpha(R(d'y'))$, и значит $\alpha(dy) = \alpha y \neq \alpha' y' = \alpha'(d'y') = \alpha(R(d'y'))$.

Таким образом, $d(\psi x) \leq R(d'(hx))$.

Используя это неравенство и вспоминая, что $d \circ \psi = \psi$, получим:

$$Q(\psi x) = Q(d(\psi x)) \leq QR(d' \circ h(x)) \leq s(d' \circ h(x)) \leq s \circ h(x).$$

Здесь $s \in B$ – такая монотонная функция, что $Q \circ R \leq \leq s$; её существование следует из условия 2 для функций Q и R, и из того факта, что B является классом, и, следовательно, $s \in B \Rightarrow s + const \in B$. Заметим также, что функция Q монотонна, поскольку R монотонна и $R \circ Q = \mathrm{id}_{\mathbb{N}}$.

Положим $u := s \circ h$ и заметим, что $u \in B$.

Мы получили бы нужное противоречие $\forall x(\exists y \geq x)\ Q(y) \leq \leq Q(\psi y) \leq u(y)$, если бы смогли показать, что $\forall x(\exists y \geq \geq x)[y \in X \ \wedge \ \psi_X(y) \geq y]$. Но это прямо вытекает из следующей ниже леммы и из изоморфизма линейных порядков $(X, \leq) \simeq \simeq (\mathbb{N}, \leq)$. ∎

ЛЕММА. *Пусть $\theta \in \mathrm{S}(\mathbb{N})$ - некоторая перестановка множества натуральных чисел. Тогда $\forall x(\exists y \geq x)\ \theta y \geq y$.*

ДОКАЗАТЕЛЬСТВО.

Предположим, что $\exists x(\forall y \geq x)\ \theta y < y$, и попытаемся прийти к противоречию.

Обозначим $x_0 := \inf\{x | \ (\forall y \geq x)\ \theta y < y\}$.

Поскольку множество $X_0 := \{0, ..., x_0\}$ — конечно, существует $x_1 := \sup\{\theta x | \ x \leq x_0\}$. Покажем, что $(\forall x \leq x_1)\ \theta x \leq x_1$. Рассмотрим некоторое $x \leq x_1$. Если $x \leq x_0$, то $\theta x \leq x_1$ по определению x_1. Если $x > x_0$, то $\theta x < x$ по определению x_0, и значит $\theta x \le x_1$.

Таким образом, ограничение $\theta_1 = \theta|X_1$ отображает X_1 в само себя, где $X_1 := \{0, ..., x_1\}$. Поскольку, $\theta_1 : X_1 \to X_1$ инъективно и множество X_1 конечно, то θ_1 биективно. Следовательно, $(\forall x > x_1)\ \theta x > x_1$. Положим $x_2 := x_1 + 1$; тогда $\theta x_2 \ge x_2$.

Покажем, что $x_1 \ge x_0$. Если $x_1 < x_0$, то $(\forall x \le x_0)\ \theta x \le x_1 < x_0$. Это значит, что $\theta_0 := \theta|X_0$ является вложением множества X_0 в $X_0 \backslash \{x_0\}$. Но это невозможно, поскольку X_0 — конечное множество.

Таким образом, мы имеем: $\theta x_2 < x_2$, ввиду неравенства $x_2 > x_0$ и определения x_0. Но мы видели, что: $\theta x_2 \ge x_2$. Это противоречие завершает доказательство. ■

4. Алгоритмы ограниченной сложности

В этом параграфе мы построим такие алгоритмы $Q \in \mathcal{A}$ и $R \in \mathcal{E}$, что:

1) $(\forall f \in \mathcal{E})$ [алгоритм $Qf(Rz_1, ..., Rz_n)$ принадлежит классу $\mathcal{E}$];

2) $(\forall f \in \mathcal{E})\ f \preceq Q$;

3) $R \circ Q = \mathrm{id}_{\mathbb{N}}$ и $Q \circ R \preceq s$ для некоторой монотонной $s \in \mathcal{E}$;

4) функция R монотонна.

Напомним, что $\mathcal{E} = \{f \in \mathcal{A} |\ |f| \in C\}$ — класс "эффективных" абстрактных алгоритмов.

Определим $q(x) := p^x(0)$ и $r(y) := \inf\{x |\ q(x) \ge y\}$, и положим $Q(y) := p^{r(y)}(y)$ и $R(z) := \inf\{y |\ Q(y) \ge z\}$.

ПРЕДЛОЖЕНИЕ 3. *Предположим, что $f : \mathbb{N}^{n+1} \to \mathbb{N}$ — некоторая строго 1-монотонная, и $v : \mathbb{N}^{n+1} \to \mathbb{N}$ — 1-монотонная функции, причём:*

$$\begin{cases} f(0, \overline{x}) \le v(0, \overline{x}), \\ f(y + 1, \overline{x}) \le v(f(y, \overline{x}), \overline{x}). \end{cases}$$

Тогда, $f \in \mathrm{dom}(\mathbf{M})$, т.е. обратная функци $g := \mathbf{M}(f)$ всюду определена, и имеет следующие свойства:

1) *g 1-монотонна;*

2) $g(f(y, \overline{x}), \overline{x}) = y$;

3) $f(g(z, \overline{x}), \overline{x}) \le v(z, \overline{x})$;

4) $g(z+1,\overline{x}) \le g(z,\overline{x})+1$;

5) $g(z,\overline{x}) \le z$.

ДОКАЗАТЕЛЬСТВО.

Поскольку $f(y,\overline{x})$ строго 1 монотонна, то $y \le f(y,\overline{x})$, и функция $g(z,\overline{x}) = \inf\{y|\ f(y,\overline{x}) \ge z\}$ определена для всех $z \in \mathbb{N}$. Очевидно, функция g 1–монотонна.

Покажем свойства 2–5.

2. Пусть $z := f(y,\overline{x})$ и $y' := g(z,\overline{x})$. Если $y < y'$, то $f(y,\overline{x}) < z$. Если $y > y'$, то $f(y,\overline{x}) > f(y',\overline{x}) \ge z$. Значит, $y = y'$.

3. Положим $y := g(z,\overline{x})$ и покажем, что $f(y,\overline{x}) \le v(z,\overline{x})$. Если $y = 0$, то $f(y,\overline{x}) = f(0,\overline{x}) \le v(0,\overline{x}) \le v(z,\overline{x})$. Если же $y > 0$, предположим, что $y = y'+1$ и определим $z' := f(y',\overline{x})$. По доказанному выше свойству 2, $y' = g(z,\overline{x})$. Следовательно, $z' \le z$, поскольку g 1–монотонна.

Таким образом,

$$f(y,\overline{x}) = f(y'+1,\overline{x}) \le v(f(y',\overline{x}),\overline{x}) = v(z',\overline{x}) \le v(z,\overline{x}).$$

4. Пусть $y := g(z,\overline{x})$; имеем $f(y,\overline{x}) \ge z$. Поскольку f строго 1–монотонна, то $f(y+1,\overline{x}) \ge z+1$. Следовательно, $g(z+1,\overline{x}) \le y+1 \le g(z,\overline{x})+1$.

5. Из вышеизложенного вытекает, что $g(z,\overline{x}) \le z$, поскольку $g(0,\overline{x}) = 0$.

ТЕОРЕМА 1. *Алгоритмы Q и R попадают в класс $\mathcal{A}$. Далее, $R \circ Q = \mathrm{id}_{\mathbb{N}}$ и $Q \circ R \le p^2$, и функция, вычисляемая алгоритмом R, монотонна.*

ДОКАЗАТЕЛЬСТВО. Укажем алгоритм из $\mathcal{A}$, который вычисляет функцию $q(x)$. Положим $g := \mathsf{R}(\mathbf{id}, \xi(\pi(p')))$, где $p' \in \mathcal{A}$ – алгоритм, вычисляющий функцию p, bи существующий по аксиоме 4.5. Тогда $g \in \mathcal{A}$ и $g(x,y) = p^x(y)$. Имеем $q = \xi(g) \circ \mathbf{zero}$, и значит $q \in \mathcal{A}$.

Далее, найдётся такой алгоритм в классе $\mathcal{A}$, который вычисляет функцию $r(y)$. Действительно: по аксиоме 2.2, имеем $p(y) > y$. Следовательно, функция $q(x) = p^x(0)$ строго монотонна; по предложению 3, алгоритм $r = \mathsf{M}(q)$ определён, т.е. $r \in \mathcal{A}$. Заметим, что функция, вычисляемая r, монотонна и $r(y+1) \le r(y)+1$.

Очевидно $Q \in \mathcal{A}$, поскольку $Q = \Delta(g \circ r)$.

Применим предложение 3 к алгоритму Q. Функция $Q(y)$ строго монотонна, поскольку из $y' < y''$ следует $Q(y') = p^{r(y')}(y') \le p^{r(y'')}(y') < p^{r(y'')}(y'') = Q(y'')$. Далее, Q удовлетворяет условиям предложения 3, если положить $v(y) := p^2(y)$

$$\begin{cases} Q(0) = p^{r(0)}(0) = p^0(0) = \mathbf{id}(0) = 0 \le p^2(0), \\ Q(y+1) = p^{r(y+1)}(y+1) \le p^{r(y)+1}(y+1) \le \\ \qquad\qquad\qquad\qquad \le p^{r(y)+1}(py) = p^2Q(y), \end{cases}$$

(мы использовали монотонность $r(y)$ и неравенство $p(y) > y$).

По предложению 3, $R \circ Q = \mathrm{id}_{\mathbb{N}}$ и $Q \circ R \le p^2$, и функция $R(z)$ всюду определена и монотонна.

В частности, $Q \in \mathrm{dom}(\mathbf{M})$ и $R = \mathbf{M}(Q) \in \mathcal{A}$. □

Теоремой 1 установлены свойства 3 и 4 функций Q и R, поскольку $p^2 \in \mathcal{E}$ по аксиоме 4.5. Установим теперь свойство 2.

Рассмотрим некоторый алгоритм $f \in \mathcal{E}$, т.е. такой $f \in \mathcal{A}$, что $|f| \in \mathcal{C}$. По аксиоме 4.6, $(\exists g \in \mathcal{C})\ f \preceq |f| \oplus g$; следовательно, по аксиоме 3.2, $\kappa(f) \in \mathcal{C}$. Значит, по аксиоме 2.3, $\exists k \forall \overline{x}\ f(x_1, \ldots, x_n) \le \le p^k(x_1 \oplus \ldots \oplus x_n)$.

Положим $x := q(k)$, тогда $(\forall y \ge x)\ r(y) \ge k$.

Тогда, $(\forall y_1, \ldots, y_n \ge x)\ f(\overline{y}) \le p^{r(y_1 \oplus \ldots \oplus y_n)}(y_1 \oplus \ldots \oplus y_n) = Q(y_1 \oplus \ldots \oplus y_n)$, и это означает, что $f \preceq Q$.

Наиболее трудное свойство 1 мы сформулируем ниже в виде теоремы 2. Её доказательство потребует следующих далее двух лемм и одного технического предложения.

ПРЕДЛОЖЕНИЕ 4. *Справедливы утверждения:*

1) $p^0(y) \oplus p^1(y) \oplus \ldots \oplus p^x(y) \le p^{x+1}(y)$;

2) $y \oplus \ldots \oplus y \le p^n(y)$ (*n слагаемых*);

3) $r(p(y)) = r(y) + 1$;

4) $Q(p(y)) = p^2Q(y)$.

ДОКАЗАТЕЛЬСТВО.

1. Индукцией по x:

если $x = 0$, то $p^0(y) = y < p(y) = p^{x+1}(y)$, по аксиоме 2.4;

если предположить, что утверждение 1 верно для некоторого x, то $p^0(y) \oplus p^1(y) \oplus \ldots \oplus p^x(y) \oplus p^{x+1}(y) \le p^{x+1}(y) \oplus p^{x+1}(y)$ ввиду ассоциативности и монотонности операции $\oplus$.

Далее, $p^{x+1}(y) \oplus p^{x+1}(y) \le p\left(p^{x+1}(y)\right)$, по аксиоме 2.4.

2. Имеем, $\overbrace{y \oplus \ldots \oplus y}^{n \text{ раз}} \le \overbrace{y \oplus \ldots \oplus y}^{2^n \text{ раз}}$, поскольку $n \le 2^n$.

Легко показать индукцией по n, что $\overbrace{y \oplus \ldots \oplus y}^{2^n \text{ раз}} \le p^n(y)$, поскольку $p^n(y) \oplus p^n(y) \le p\left(p^n(y)\right)$.

3. Пусть $x := r(y) = \inf\{i \mid q(i) \ge y\}$.

Тогда $q(x+1) = p^{x+1}(0) = pq(x) \ge p(y)$, и значит $r(p(y)) \le \le r(q(x+1)) = x+1$. По предложению 3, $q(r(y)) \le p(y)$.

Покажем, что $q(r(y)) < p(y)$.

Если предположить, что $q(r(y)) = p(y)$, то $x = r(y) > 0$, поскольку $q(0) = p^0(0) = 0 < p(0) \le p(y)$. Рассмотрим такое x', что $x = x' + 1$. Имеем: $p(p^{x'}(0)) = p^x(0) = q(x) = q(r(y)) = p(y)$. Так как p строго монотонна, то $q(x') = p^{x'}(0) = y$, и значит $x' \ge \ge r(y) = x = x' + 1$. Таким образом, поскольку $q(x) < p(y)$, то $r(p(y)) > x$. Следовательно, $r(p(y)) = x + 1$.

4. $Q(py) = p^{r(py)}(py) = p^{r(y)+1}(py) = p^2 p^{r(y)}(y) = = p^2 Q(y)$. □

ЛЕММА 1. *Имеем:*

1) $|q| \preceq_C p^k \circ q$ *для некоторого* $k \in \mathbb{N}$,

2) $|r| \in C$.

ДОКАЗАТЕЛЬСТВО. Рассмотрим алгоритм $g(x, y) := p^x(y)$; покажем, что $|g| \preceq_C p^k \circ g$ для некоторого $k \in \mathbb{N}$.

Мы определили $g = \mathsf{R}(g_0, g_1)$ через примитивную рекурсию с $g_0(y) = y$ и $g_1(z, x, y) = p(z)$, т.е. $g_0 = \mathbf{id}$ и $g_1 = \eta\pi^2(p)$.

По аксиомам 4.1, 4.4 и 4.5, $|g_0|, |g_1| \in C$. Оценим $|g|(x, y)$, используя аксиому 4.2: $|g(x,y)| \preceq_C |g_0(y)| \oplus |g_1|(g(0,y), 0, y) \oplus \ldots \ldots \oplus |g_1|(g(x-1,y), x-1, y)$ (можно писать $|w(\overline{x})|$ вместо $|w|(\overline{x})$, если аргументами являются константы или свободные переменные, но не функции). По аксиоме 2.3, $(\exists l \in \mathbb{N})$ $|g_1(z,x,y)| \le \le p^l(z \oplus x \oplus y)$, и по п.2 предложения 4, $z \oplus x \oplus y \le p^3(\max\{z, x, y\})$. Заметим, что $g(x,y) = p^x(y) \ge \max\{x, y\}$, и значит $\max\{g(x,y), x, y\} = g(x,y)$.

Рассмотрим оценку:

$$
\begin{aligned}
|g(x,y)| \preceq_C\ & |g_0(y)| \oplus p^{l+3} g(0,y) \oplus \ldots \oplus p^{l+3} g(x-1,y) = \\
&= |g_0(y)| \oplus p^{l+3}(y) \oplus \ldots \oplus p^{(l+3)+(x-1)}(y) \le \\
&\le |g_0(y)| \oplus p^0(y) \oplus \ldots \oplus p^{(l+3)+(x-1)}(y)
\end{aligned}
$$

(вторая строчка оценки содержит $x+1$ слогаемое, третья — $l+x+4$ слагаемых).

Определим $k := l+3$. Используя п.1 предложения 4, получим:

$$|g(x,y)| \preceq_C |g_0(y)| \oplus p^0(y) \oplus \ldots \oplus p^{k+x-1}(y) \leq |g_0(y)| \oplus p^{k+x}(y).$$

Так как $|g_0(y)| \in \mathcal{C}$, то $|g(x,y)| \preceq_C p^{k+x}(y) = p^k g(x,y)$.

Теперь мы готовы показать, что $|q| \preceq_C p^k \circ q$. Поскольку $q(x) = g(x,0)$, то $q = \xi(g) \circ \mathbf{zero}$. Значит:

$|q| \preceq_C (|\xi(g)| \circ \mathbf{zero}) \oplus |\mathbf{zero}| \preceq_C |\xi(g)| \circ \mathbf{zero} \preceq_C \xi(|g|) \circ \mathbf{zero}$.

Но мы знаем, что $|g| \preceq_C p^k \circ g$, следовательно,

$\xi(|g|) \preceq_C \xi(p^k \circ g) = p^k \circ \xi(g)$.

Наконец, $|q| \preceq_C \xi(|g|) \circ \mathbf{zero} \preceq_C p^k \circ \xi(g) \circ \mathbf{zero} = p^k \circ q$.

Оценим теперь сложность алгоритма $r = \mathbf{M}(q)$.

По аксиоме 4.3,

$$\begin{aligned} |r(y)| \quad & \preceq_C |q(0)| \oplus \ldots \oplus |q|(ry) \oplus r(y) \preceq_C \\ & \preceq_C p^k q(0) \oplus \ldots \oplus p^k q(ry) \oplus r(y) \leq \\ & \leq p^k q(r0) \oplus \ldots \oplus p^k q(ry) \oplus r(y) \leq \\ & \quad \leq p^{k+1}(0) \oplus \ldots \oplus p^{k+1}(y) \oplus y \end{aligned}$$

(вторая строчка содержит $r(y)+2$ слагаемых; третья — $y+2$ слагаемых).

Здесь мы использовали неравенство $r(y) \leq y$. Заметим также, что функция r сюрьективна, поскольку $r \circ q = \mathrm{id}_{\mathbb{N}}$. Наконец вспомним, что по предложению 3, $q \circ r \leq p$.

По аксиоме 3.1 функция $y \mapsto p^{k+1}(0) \oplus p^{k+1}(1) \oplus \ldots \ldots \oplus p^{k+1}(y)$ принадлежит классу $\mathcal{C}$. Также, по аксиоме 2.2, $y < p(y)$, и значит, по аксиоме 3.2, $\mathrm{id}_{\mathbb{N}} \in \mathcal{C}$.

Следовательно, по аксиоме 3.2, $|r| \in \mathcal{C}$. ■

ЛЕММА 3. *Аналогично лемме 2, имеем:*

1) $|Q| \preceq_C p^k \circ Q$ *с тем же самым показателем* $k \in \mathbb{N}$,

2) $|R| \in \mathcal{C}$.

ДОКАЗАТЕЛЬСТВО.

1. Воспользуемся тем же алгоритмом $g(x,y)$, что и в лемме 2:

$$|Q(y)| \preceq_C |g|(r(y),y) \oplus |r(y)| \preceq_C$$
$$\preceq_C |g|(r(y),y) \preceq_C p^k g(r(y),y) = p^k Q(y).$$

2. Аналогично лемме 2:

$$\begin{aligned}|R(z)| &\preceq_C |Q(0)| \oplus \ldots \oplus |Q|(Rz) \oplus R(z) \preceq_C \\ &\preceq_C p^k Q(0) \oplus \ldots \oplus p^k Q(Rz) \oplus R(z) \le \\ &\le p^k Q(R0) \oplus \ldots \oplus p^k Q(Rz) \oplus R(z) \le \\ &\le p^{k+2}(0) \oplus \ldots \oplus p^{k+2}(z) \oplus z \qquad - \in C\end{aligned}$$

(вторая строчка содержит $R(z) + 2$ слагаемых; третья $-z + 2$ слагаемых).

Следовательно, $|R| \in C$. ■

ТЕОРЕМА 2. *Пусть* $f \in \mathcal{E}$, *положим* $F(z_1, \ldots, z_n) := Qf(Rz_1, \ldots, Rz_n)$. *Тогда* $F \in \mathcal{E}$.

ДОКАЗАТЕЛЬСТВО.

По аксиоме 4.1 мы можем оценить:

$|F(z_1, ..., z_n)| \preceq_C |Q| f(Rz_1, ..., Rz_n) \oplus |f| (Rz_1, ..., Rz_n) \oplus \oplus |R(z_1)| \oplus ... \oplus |R(z_n)|$.

По лемме 3, $|R(z_i)| \in C$; значит

$$|F(z_1, ..., z_n)| \preceq_C |Q| f(Rz_1, ..., Rz_n) \oplus |f| (Rz_1, ..., Rz_n).$$

По аксиоме 4.6, $R \preceq_C |R|$, значит, по аксиоме 3.2, $R \in C$.

Таким образом, $|f| (Rz_1, ..., Rz_n) \in C$, поскольку $|f| \in C$, по условиям теоремы о неустойчивости.

Следовательно, $|F(z_1, ..., z_n)| \preceq_C |Q| f(Rz_1, ..., Rz_n)$. По лемме 3, $|Q| \preceq_C p^k \circ Q$, тогда: $|F(z_1, ..., z_n)| \preceq_C p^k Qf(Rz_1, ..., Rz_n)$. Поскольку $f \preceq_C |f| \in C$, то $f \in C$, и, следовательно, $(\exists t \in \mathbb{N})\ f(y_1, ..., y_n) \le p^t(y_1 \oplus ... \oplus y_n)$.

Таким образом, используя неравенство $Q(px) = p^2 Q(x)$, получим:

$$|F(\overline{z})| \preceq_C p^k Q p^t (Rz_1 \oplus ... \oplus Rz_n) \le p^{k+2t} Q(Rz_1 \oplus ... \oplus Rz_n).$$

Обозначим $z := \max\{z_1, ..., z_n\}$. Используя п.2 предложения 4, получим:

$$|F(\overline{z})| \preceq_C p^{k+2t} Q(\overbrace{Rz \oplus ... \oplus Rz}^{n \text{ слагаемых}}) \le p^{k+2t} Q p^n (Rz) = \\ = p^{k+2t+2n} QR(z).$$

Вспомним, что $Q \circ R \le p^2$. Следовательно,

$$|F(\overline{z})| \preceq_C p^{k+2t+2n+2}(\max\{z_1, ..., z_n\}) \le \\ \le p^{k+2t+2n+2}(z_1 \oplus ... \oplus z_n) \in C.$$

Таким образом, $|F| \in C$. ■

5. Наследственно ограниченная сложность

В предыдущем параграфе мы построили некоторые алгоритмы $Q \in \mathcal{A}$ и $R \in \mathcal{E}$, удовлетворяющие условиям 1–4 для класса $\mathcal{E}$.

Теперь мы покажем, что R попадает даже в более узкий класс $\mathcal{H}$ наследственно эффективных алгоритмов, и что R удовлетворяет более трудным условиям:

1′) $(\forall f \in \mathcal{H})\ Qf(Rz_1, \ldots, Rz_n) \in \mathcal{H}$.

Соответствующие свойства 2′–3′ для класса $\mathcal{H}$ очевидны, если вспомнить неравенство $Q \circ R \le p^2$.

Таким образом, определения $\mathcal{B}$–конструктивизации и $\mathcal{B}$–сводимости применимы к классу $\mathcal{H}$.

Напомним, что $\mathcal{H}$ есть наименьший класс, содержащий алгоритм p и замкнутый относительно ограниченных операторов примитивной рекурсии R_0 и монотонного обращения M_0:

$f = \mathsf{R}_0(f_0, f_1)$ если и только если $f = \mathsf{R}(f_0, f_1)$ и $|f| \in \mathcal{C}$,

$g = \mathsf{M}_0(g_0)$ если и только если $g = \mathsf{M}(g_0)$ и $|g| \in \mathcal{C}$.

ЛЕММА 4 (о суперпозиции). *Пусть $\mathcal{B} \sqsubseteq \mathcal{A}$ и класс $\mathcal{B}$ замкнут относительно оператора* R_0. *Тогда из*

$$\begin{cases} h(\overline{z}) = f(g_0(\overline{z}), g_1(\overline{z}), \ldots, g_n(\overline{z})), \\ f = \mathsf{R}(f_0, f_1), \\ f \ 1\text{- \textit{монотонна}}, \\ f_0, f_1, g_0, g_1, \ldots, g_n \in \mathcal{B}, \\ |h| \in \mathcal{C} \end{cases}$$

следует $h \in \mathcal{B}$.

ДОКАЗАТЕЛЬСТВО. По условиям леммы,

$$\begin{cases} f(0, \overline{x}) = f_0(\overline{x}), \\ f(y+1, \overline{x}) = f_1(f(y, \overline{x}), y, \overline{x}), \end{cases}$$

где $f_0, f_1 \in \mathcal{B}$, и $h(\overline{z}) = f(g_0(\overline{z}), g_1(\overline{z}), \ldots, g_n(\overline{z}))$, причём $|h| \in \mathcal{C}$.

Определим новую функцию $\varphi(y, \overline{z})$:

$$\begin{cases} \varphi(0, \overline{z}) = \varphi_0(\overline{z}), \\ \varphi(y+1, \overline{z}) = \varphi_1(\varphi(y, \overline{z}), y, \overline{z}), \end{cases}$$

где $\varphi_0(\overline{z}) := f_0(g_1(\overline{z}), \ldots, g_n(\overline{z}))$, и

$$\varphi_1(v, y, \overline{z}) := \begin{cases} f_1(v, y, g_1(\overline{z}), \ldots, g_n(\overline{z})), & \text{если } y \le g_0(\overline{z}), \\ 0, & \text{в противном случае.} \end{cases}$$

По построению, $\varphi_0, \varphi_1 \in \mathcal{B}$, и $(\forall y \leq g_0(\overline{z}))\ \varphi(y,\overline{z}) = f(y,\overline{g}(\overline{z}))$. Следовательно, $\varphi(g_0(\overline{z}),\overline{z}) = h(\overline{z})$, и $(\forall y \leq g_0(\overline{z}))\ \varphi(y,\overline{z}) \leq h(\overline{z})$. Поскольку $g_0, g_1, \ldots, g_n \in \mathcal{B}$ по условиям леммы, достаточно показать, что $|\varphi| \in \mathcal{C}$.

По свойствам меры сложности $|\cdot|$:

$$\begin{cases} |\varphi|\,(0,\overline{z}) \preceq_{\mathcal{C}} |\varphi_0|\,(\overline{z}), \\ |\varphi|\,(y+1,\overline{z}) \preceq_{\mathcal{C}} |\varphi_1|\,(\varphi(y,\overline{z}),y,\overline{z}) \oplus |\varphi|\,(y,\overline{z}). \end{cases}$$

По свойствам класса $\mathcal{C}$: $(\exists \psi_1 \in \mathcal{C})[\psi_1$ 1–монотонна и $|\varphi_1| \leq \psi_1]$. Имеем:

$$\begin{aligned} |\varphi|(y+1,\overline{z}) &\preceq_{\mathcal{C}} \psi_1(\varphi(y,\overline{z}),y,\overline{z}) \oplus |\varphi|\,(y,\overline{z}) \leq \\ &\leq \psi_1(\varphi(g_0(\overline{z}),\overline{z}),y,\overline{z}) \oplus |\varphi|\,(y,\overline{z}) = \\ &= \psi_1(h(\overline{z}),y,\overline{z}) \oplus |\varphi|\,(y,\overline{z}) \end{aligned}$$

Следовательно $|\varphi|\,(y,\overline{z}) \preceq_{\mathcal{C}} \widetilde{\varphi}(y,\overline{z})$, где

$\widetilde{\varphi}(y,\overline{z}) := |\varphi_0|\,(\overline{z}) \oplus \psi_1(h(\overline{z}),0,\overline{z}) \oplus \ldots \oplus \psi_1(h(\overline{z}),y,\overline{z})$.

По аксиоме 3.1, $\widetilde{\varphi} \in \mathcal{C}$; значит $|\varphi| \in \mathcal{C}$. ∎

ЛЕММА 5 (о пределе). *Пусть $\mathcal{B} \sqsubseteq \mathcal{A}$ и класс $\mathcal{B}$ замкнут по* R_0. *Тогда из*

$$\begin{cases} f_0, f_1 \in \mathcal{B}, \\ f = \mathsf{R}(f_0, f_1), \\ f \text{ 1} - \textit{монотонна} \end{cases}$$

следует

$$(\exists g \in \mathcal{B}) \begin{cases} z \leq g(z,y,\overline{x}) \ \Leftrightarrow \ z \leq f(y,\overline{x}), \\ g \text{ 1} - \textit{монотонна}. \end{cases}$$

ДОКАЗАТЕЛЬСТВО.

Имеем:

$$\begin{cases} f(0,\overline{x}) = f_0(\overline{x}), \\ f(y+1,\overline{x}) = f_1(f(y,\overline{x}),y,\overline{x}). \end{cases}$$

Обозначим:

$$\begin{cases} g(z,0,\overline{x}) = g_0(z,\overline{x}), \\ g(z,y+1,\overline{x}) = g_1(g(z,y,\overline{x}),z,y,\overline{x}); \end{cases}$$

где:

$$g_0(z,\overline{x}) := \begin{cases} f_0(\overline{x}), & \text{если } f_0(\overline{x}) \le z, \\ z+1, & \text{в противном случае.} \end{cases}$$

$$g_1(u,z,y,\overline{x}) := \begin{cases} f_1(u,y,\overline{x}), & \text{если } u \le z \text{ и } f_1(u,y,\overline{x}) \le z, \\ z+1, & \text{в противном случае.} \end{cases}$$

По определению, $g_0, g_1 \in \mathcal{B}$, и

$$g(z,y,\overline{x}) = \begin{cases} f(y,\overline{x}) & \text{если } f(y,\overline{x}) \le z, \\ z+1, & \text{иначе.} \end{cases}$$

Значит, g 1 монотонна, и $z \le g(z,y,\overline{x})$ если и только если $z \le f(y,\overline{x})$.

Поскольку класс $\mathcal{B}$ замкнут относительно оператора R_0, достаточно показать, что $|g| \in \mathcal{C}$. Имеем:

$$|g|\,(z,y+1,\overline{x}) \preceq_{\mathcal{C}} |g_1|\,(g(z,y,\overline{x}),z,y,\overline{x}) \oplus |g|\,(z,y,\overline{x}) \le$$
$$\le \widetilde{g}_1(z+1,z,y,\overline{x}) \oplus |g|\,(z,y,\overline{x})$$

где $\widetilde{g}_1$ такая 1–монотонная функция из $\mathcal{C}$, что $|g_1| \le \widetilde{g}_1$.

Индукцией по y отсюда получим, что $|g| \in \mathcal{C}$. ■

ЗАМЕЧАНИЕ 3. В условиях леммы 5:

1) $f(y,\overline{x}) = g(f(y,\overline{x}),y,\overline{x})$, и

2) $f(y,\overline{x}) = \lim_{z\to\infty} g(z,y,\overline{x})$.

Докажем замечание.

Положим $z := f(y,\overline{x})$ и рассмотрим некоторое $z' > z$. Тогда $z \le g(z,y,\overline{x}) \le g(z',y,\overline{x}) < z'$, поскольку $z' \not\le f(y,\overline{x})$. Поскольку $z' > z$ выбрано произвольно, отсюда следует $z = g(z,y,\overline{x}) = g(z',y,\overline{x})$. ■

Заметим также очевидное следствие из леммы 5.

СЛЕДСТВИЕ. *Предположим, что в условиях леммы 5 класс $\mathcal{B}$ также замкнут по M_0, и при этом $h(z,\overline{x}) = \inf\{y \mid z \le f(y,\overline{x})\}$ и $|h| \in \mathcal{C}$. Тогда $h \in \mathcal{B}$.*

В предыдущем параграфе мы установили следующие оценки:

1) $|r| \in \mathcal{C}$,

2) $|R| \in \mathcal{C}$,

3) $|F| \in \mathcal{C}$, если $F(z_1,\ldots,z_n) = Q(f(Rz_1,\ldots,Rz_n))$ и $|f| \in \mathcal{C}$.

Используя эти включения, докажем следующую теорему.

ТЕОРЕМА 3. *Справедливы включения:*

1) $R \in \mathcal{H}$,

2) $F \in \mathcal{H}$, *если* $f \in \mathcal{H}$.

ДОКАЗАТЕЛЬСТВО.

Сначала докажем, что $r \in \mathcal{H}$. По определению, $p \in \mathcal{H}$. Следовательно, $q(x) = p^x(0)$ удовлетворяет условиям леммы 5 (если $f_0 := p$ и $f := q$). По следствию из леммы 5, $r = \mathbf{M}(q)$ принадлежит $\mathcal{H}$.

Докажем теперь, что $R \in \mathcal{H}$. По лемме 5, существует некоторый алгоритм $g \in \mathcal{H}$ такой, что $z \leq g(z, x, y) \Leftrightarrow z \leq p^x(y)$. Следовательно $R(z) = \inf\{y | \; z \leq g(z, r(y), y)\}$, и значит $R \in \mathcal{H}$.

Наконец покажем, что $F \in \mathcal{H}$. Рассмотрим представление $F(\overline{z}) = u(g_0(\overline{z}), g_1(\overline{z}))$, где $u(x,y) := p^x(y)$, и $g_0(\overline{z}) := r(g_1(\overline{z}))$, и $g_1(\overline{z}) := f(Rz_1, \ldots, Rz_n)$. По лемме 4, $u(g_0(\overline{z}), g_1(\overline{z})) \in \mathcal{H}$. □

6. Основные примеры классов алгоритмов

В этом параграфе мы определяем пример "алгоритмического универсума" $\mathcal{A}$, и несколько примеров подклассов эффективных и наследственно эффективных алгоритмов и функций в классах $\mathcal{A}$ и $\mathcal{F}$. Класс $\mathcal{A}$ составят так называемые "программируемые машины". Автор затрудняется назвать первоисточник этого понятия. Несколько модифицировав, мы подчерпнули его в работе [3], которая сама ссылается на [6].

Основными классами сложности будут линейный и полиномиальный, которые соответствуют классам А.Гжегорчика $\mathcal{E}^1$ и $\mathcal{E}^2$ соответственно. Далее, мы рассматриваем аналоги классов $\mathcal{E}^n$, $n = 1, 2, \ldots$, используя специально определяемые функции $x \oplus_n y$, $p_n(x)$ и класс $\mathcal{C}^n$.

Для всех $n \geq 2$ определим:

$$\begin{aligned} x \oplus_n y &:= x + y, \\ p_n(x) &:= f_n(x+1, x+1), \\ \mathcal{C}^n &:= \{h | \; (\exists g \in \mathcal{E}^n) \; h \preceq g\}, \end{aligned}$$

где $f_n(x, y)$ — функция Гжегорчика, описываемая ниже.

Для случаев $n = 1$ и $n = 2$ по определению выполняется:

$$\begin{aligned} x \oplus_1 y &:= \max\{x, y\}, & x \oplus_2 y &= x + y, \\ p_1(x) &:= 2 \cdot (x+1), & p_2(x) &= (x+2)^2, \\ \mathcal{C}^1 &:= \text{линейный}; & \mathcal{C}^2 &= \text{полиномиальный}. \end{aligned}$$

Рассмотрим три функционала сложности:

$$|f(\overline{x})|_0 := f(\overline{x}) \text{ —значение функции,}$$
$$|f(\overline{x})|_1 := \text{объём использованной памяти,}$$
$$|f(\overline{x})|_2 := \text{количество шагов вычислителя.}$$

Понятия "память" и "шаги вычислителя" относятся к программируемым машинам, определяемых следующим образом.

Программируемые машины

Программируемая машина f состоит из:

1) конечного списка регистров $r_0, r_1, \ldots, r_n, r_{n+1}, \ldots, r_{n+m}$, способных запоминать натуральные числа, и

2) последовательности инструкций $c_1, \ldots, c_K$, т.е. программы.

Машина f вычисляет значение $f(x_1, \ldots, x_n)$ следующим образом:

1) в регистры $r_1, \ldots, r_n$ загружаются значения аргументов функции, а значения остальных регистров полагаются равными 0,

2) выполняется программа $c_1, \ldots, c_K$, начиная с инструкции c_1,

3) когда программа остановится, результат вычисления $f(x_1, \ldots, x_n)$ считывается из регистра r_0.

Каждая из инструкций c_k является одной из следующих:

1) $r_i \leftarrow 0$ — очистить регистр r_i,

2) $r_i \leftarrow r_i + 1$ — увеличить значение регистра r_i,

3) $r_i \leftarrow r_i - 1$ — уменьшить r_i, если $r_i > 0$, или оставить $r_i = 0$,

4) $r_i \rightarrow l$ — перейти к инструкции c_l, если $r_i = 0$,

5) $\rightarrow l$ — безусловный переход к c_l.

Подиндекс i в r_i, и адрес перехода l всегда является константой. Если $l = 0$ или $l > K$, то эта инструкция "goto" трактуется как команда конца вычислений *halt*.

Заметим, что инструкцию "$r_i \leftarrow 0$" легко имитировать при помощи простейшего цикла, итерирующего инструкцию "$r_i \leftarrow r_i - 1$" пока $r_i > 0$.

Аналогичным образом, можно написать программу, перемещающую содержимое из некоторого регистра r_i в другой регистр r_j:

k) clear r_j,
$k+1$) $r_i \to k+5$,
$k+2$) $r_i \leftarrow r_i - 1$,
$k+3$) $r_j \leftarrow r_j + 1$,
$k+4$) $\to k+1$.

Мы можем вставить эту последовательность команд $c_k, \ldots, c_{k+4}$ в качестве подпрограммы некоторой программируемой машины. Удобно обозначить эту подпрограмму именем "move". Таким образом, мы можем использовать псевдокоманду, как бы занимающую одну строчку в записи программы, не раскрывая её внутреннюю структуру:

l) move $r_j \leftarrow r_i$,
$l+1$) ...

Мы уже сделали подобную подстановку, вставив псевдокоманду c_k ="clear r_j" в подпрограмму "move".

Очевидно, мы можем определить также псевдокоманду "copy $r_j \leftarrow r_i$", копирующую содержимое регистра, которая использует некоторый дополнительный регистр r' в качестве временной памяти. (Сначала перемещаем значение r_i одновременно в r' и в r_j; а затем, перемещаем значение r' в r_i чтобы восстановить его.) Договоримся резервировать один дополнительный регистр r' в каждой программируемой машине, рассматриваемой ниже, чтобы можно было использовать псевдокоманду "copy".

В общем случае, мы можем неявно включить код целой машины $g(y_1, \ldots, y_m)$, определяя некоторую другую машину $f(x_1, \ldots, x_n)$, записав псевдокоманду:

k) $r_j \leftarrow g(r_{i_1}, \ldots r_{i_m})$.

Здесь предполагается, что внутренние регистры машины g включены в список регистров машины f, и это подмножество регистров не используется другими командам машины f. Наше соглашение о "вызове подпрограммы" не требует, чтобы величины, записанные в качестве аргументов в регистры $r_{i_1}, \ldots r_{i_m}$ сохранялись после остановки вычисления $g(r_{i_1}, \ldots r_{i_m})$.

Такая подстановка фактически означает суперпозицию программируемых машин, и очевидно подобным образом можно определить остальные операторы клона η, ξ, Δ, π на множестве программируемых машин $\mathcal{A}$. Также нетрудно определить оператор монотонного обращения M.

Нетрудно проверить справедливость аксиом 1–4 для этих операторов и для функций Φ_n и p_n, для классов $\mathcal{C}^n$, $n \geq 2$, и для функционалов сложности $|\cdot|_0$, $|\cdot|_2$.

Рассмотрим особо случай примитивной рекурсии R.

Определим следующую программу для функции $f = \mathsf{R}(f_0, f_1)$, вычисляющую $f(y, x_1, \ldots, x_n)$ итерациями машины f_1:

Машина: $z \leftarrow f(y, x_1, \ldots, x_n)$

Регистры: $z(\equiv r_0), y, x_1, \ldots, x_n, s, r, x'_1, \ldots, x'_n, s', r'$, и регистры из f_0, f_1

1) copy $x'_1 \leftarrow x_1$; ...; copy $x'_n \leftarrow x_n$,
2) $z \leftarrow f_0(x_1, \ldots, x_n)$,
3) $s \leftarrow 0$,
4) $y \rightarrow 0$ — "если $y = 0$, то *halt*",
5) move $r \leftarrow z$; copy $s' \leftarrow s$,
6) copy $x \leftarrow x'_1$; ...; copy $x_n \leftarrow x'_n$,
7) $z \leftarrow f_1(r, s, x_1, \ldots, x_n)$,
8) copy $s \leftarrow s'$; $s \leftarrow s + 1$,
9) $y \leftarrow y - 1$,
10) $\rightarrow 4$.

Очевидно, аксиома 4.2 выполняется для определённого таким образом оператора R, функций $\oplus_n$, p_n, классов $\mathcal{C}^n$, $n \geq 2$, и функционалов $|\cdot|_0$ и $|\cdot|_2$.

Также нетрудно проверить аксиомы 1–4 для операторов клона $\circ$, η, ζ, Δ и π, и для рекурсивных операторов R и M по отношению к функциям $\oplus_1$ и p_1, линейному классу $\mathcal{C}^1$, и функционалу "потребляемой памяти" $|\cdot|_1$.

Классы Гжегорчика

А.Гжегорчик [4] определил класс $\mathcal{E}^n$, $n \in \mathbb{N}$, как наименьший подкласс $\mathcal{F}$, замкнутый относительно оператора ограниченной примитивной рекурсии, и содержащий особую функцию $f_n(x,y)$:

$$\begin{cases} f_0(x,y) := y+1 \\ f_1(x,y) := x+y \\ f_2(x,y) := (x+1)\cdot(y+1) \\ f_{n+3}(0,y) := f_{n+2}(y+1,y+1) \\ f_{n+3}(x+1,y) := f_{n+3}(x, f_{n+3}(x,y)) \end{cases}$$

В [4] показано,что для всех $n \geq 2$:

из $f \in \mathcal{E}^n$ и $g(y,\overline{x}) = f(0,\overline{x}) + \ldots + f(y,\overline{x})$ следует $g \in \mathcal{E}^n$,
и $(\forall g \in \mathcal{E}_1^n)\exists t \forall x\ g(x) \leq f_{n+1}(t,x)$.

Заметим, что $f_{n+1}(t,x) = p_n^{2^t}(x)$ для $n \geq 2$; поскольку $p_n(x) = f_{n+1}(0,x)$.

Следовательно, аксиомы 1–4 выполняются для функций $\oplus_n$ и p_n, классов $\mathcal{C}^n$, $n \geq 2$, и функционала $|\cdot|_0$. Также, легко проверить справедливость аксиом и для $n = 1$.

Таким образом, класс Гжегорчика $\mathcal{E}^n = \mathcal{H}^n$ — совпадает с классом наследственно эффективных функций, определённым для соответствующего $n \geq 1$.

З а к л ю ч е н и е

Мы описали абстрактную технику, объеденяющую воедино два известных подхода к исследованию алгоритмической сложности.

1. Подсчёт числа затраченных шагов вычислительной машины, или измерение объёма использованной ею памяти.

2. Рассмотрение алгоритмических классов, замкнутых относительно суперпозиции, а также — подходящих рекурсивных операторов, порождающих алгебру "наследственно хороших" алгоритмов.

Рассмотренная выше иерархия $\mathcal{C}^1 \subset \mathcal{C}^2 \subset \ldots$ приемлемых скоростей роста функций порождает иерархию наследственно эффективных классов $\mathcal{H}^1 \subseteq \mathcal{H}^2 \subseteq \ldots$ в рамках абстрактного алгоритмического универсума $\mathcal{A}$. Основной пример иерархии Гжегорчика $\mathcal{E}^n = \mathcal{H}^n$, $n \geq 1$, подсказывает, что объединение абстрактных наследственно эффективных классов $\mathcal{P} := \bigoplus\limits_{n \geq 1} \mathcal{H}^n$ должно обладать схожими свойствами с классом примитивнорекурсивных функций $\mathcal{R} = \bigoplus_{n\geq 1} \mathcal{E}^n$.

Было бы интересно исследовать иерархию классов низкой вычислительной сложности $\mathcal{H}^1 \subseteq \mathcal{H}^2 \subseteq \ldots$, особенно — соотношение между $\mathcal{H}^n$ и $\mathcal{H}^{n+1}$, и сходство и различия между $\mathcal{P}$ и $\mathcal{R}$.

Идея, как построить функции Q и R для классов Гжегорчика $\mathcal{E}^n$, $n \geq 1$, была развита автором в 1986 и оставалась неопубликованной до 1990, когда вышла статья [5]. Ради целостности изложения, в [5] был опубликован упрощённый вариант представленной здесь техники "абстрактного функционала сложности". Эта упрощённая методика использует обычную функцию суммирования $x+y$ вместо $x \oplus y$, и позволяет исследовать только классы полиномальной, экспоненциальной, или более высокой сложности. Но конечно, она не покрывает линейную сложность, потому что аксиома 3.1 нарушается для линейного класса $\mathcal{C}^1$ и и простейшей суммы из y линейных слагаемых $x + \ldots + x$.

Вместе с тем, в последнее время интерес к линейным, полиномиальным, и к другим подобным классам алгоритмической сложности кажется возрождается. Поэтому, автор решил опубликовать полную версию своей абстрактной технологии, которая позволяет единообразно исследовать различные классы сложности.

Автор рад поблагодарить профессора С.С.Гончарова, который инициировал начало этой работы в 1986, и побудил автора к её опубликованию теперь, в конце 1999.

Литература

1. КОН П. Универсальная алгебра. — М.: Мир, 1968. — 249 с. — [Пер. с англ.]- P.M.Cohn. Universal algebra. (Harper's Series in Modern Mathematics. Harper & Row Publishers, New York, Evanston & London, 1965).

2. ЕРШОВ Ю.Л.. Проблемы разрешимости и конструктивные модели. / Математическая логика и основания математики.- М.: Наука, 1980.

3. GOLES E., MAAS A., MARTINEZ S. On the limit set of some universal cellular automata. // Theoretical Computer Science, 1993.- Vol.110. - P.53-78.

4. ГЖЕГОРЧИК А. Некоторые классы рекурсивных функций. // Библиотека кибернетического сборника: Проблемы математической логики.- М.: Мир, 1970.

5. ЛАТКИН Е.И. Полиномиальная неавтоустойчивость: алгебраический подход. //Логические методы в программировании. — Новосибирск, 1990. — Вып. 133: Вычислительные системы. — С. 14-37.

6. MINSKY M. Computation: finite and infinite machines. (Prentice-Hall, Englewood Cliffs, NJ, 1967.)

Поступила в редакцию
1 сентября 1999 года