МЕТОДОЛОГИЧЕСКИЕ ПРОБЛЕМЫ НАУКИ
(Вычислительные системы)
1991 год Выпуск 138

УДК 519.685

ЛОГИКА ПОДСКАЗОК

Д.Е. Пальчунов

В в е д е н и е

Изучаются логические средства формализации взаимодействия человека и компьютера в процессе решения вычислительной задачи. Как одно из таких средств рассматривается "логика подсказок" - многосортная логическая система $\mathfrak{L}$.Система $\mathfrak{L}$ служит для получения, систематизации и использования подсказок, которые может дать эксперт программе, обрабатывающей данные о предметной области.

Система $\mathfrak{L}$ является инструментом взаимодействия эксперта с логическим исчислением L_0 , описывающим предметную область. Под вычислительной задачей мы будем понимать задачу проверки истинности гипотез о предметной области; более точно - выяснение, выводимо ли данное предложение в логическом исчислении L_0 ; эту задачу мы будем называть запросом к исчислению L_0 .

Предполагается, что эксперт обладает определенной интуицией о том, как можно обогатить систему L_0 новыми аксиомами и правилами вывода, чтобы существенно ускорить вычисление запроса к этой системе. Однако если просто изменить L_0 , есть риск сделать ошибку в описании предметной области - попытка эффективизировать исчисление может привести к неверному исчислению. Поэтому вместо изменения L_0 мы создаем новое исчис-

ление $\mathbf{L}$, более богатое, чем $\mathbf{L}_0$, и затем решаем нашу задачу в исчислении $\mathbf{L}$. Если нам удалось найти вывод предложения, данного в запросе, в исчислении $\mathbf{L}$, *мы обосновываем этот вывод в исчислении* $\mathbf{L}_0$, и только после получения обоснования говорим, что *наше предложение выводимо в* $\mathfrak{L}$.

Таким образом, мы отделяем проблему правильности (в том числе непротиворечивости) исчисления от проблемы его вычислительной силы: если новое исчисление $\mathbf{L}$, созданное экспертом для ускорения вычислений, оказалось ошибочным, то это не отразится на верности утверждений, выводимых в $\mathfrak{L}$, а скажется только на эффективности вычислений в $\mathfrak{L}$. Единственным критерием правильности ответов эксперта является эффективность вычислений.

Идеи, положенные в основу "логики подсказок", тесно связаны с предложенной Ю.Л.Ершовым и К.Ф.Самохваловым [1,2] новой парадигмой философии математики - рассмотрением различных свойств теорий (например, полноты и непротиворечивости) не абсолютно, а относительно определенного класса проблем (задач). Теория, не являющаяся полной в обычном смысле, может оказаться полной по отношению к интересующему нас классу задач.

Используя идеи Ю.Л.Ершова и К.Ф.Самохвалова, мы можем сформулировать следующий принцип, нуждающийся, правда, в дальнейшем уточнении: непротиворечивость и полнота теории в обычном смысле должны *специально* доказываться, а непротиворечивость и полнота относительно фиксированного класса задач могут проверяться на практике *при решении этого класса задач*.

Дадим формальное описание многосортной логической системы $\mathfrak{L}$.

Мы имеем логическое исчисление $\mathbf{L}_0$ и задачу проверки выводимости предложения Φ в этом исчислении, либо задачу поиска термов $C_1,\ldots,C_n$ таких, что предложение $\Phi(C_1,\ldots,C_n)$

выводимо в L_0. Для того чтобы избавиться от полного перебора, мы рассматриваем другое, более сильное исчисление L_1: если $L_0 \vdash \Psi$, то $L_1 \vdash \Psi$, но не обязательно наоборот. Мы доказываем $L_1 \vdash \Phi$, рассматриваем вывод $\Phi_1, \ldots, \Phi_m = \Phi$ в L_1 и пытаемся достроить его до вывода $\Psi_1, \ldots, \Psi_k = \Phi$ в L_0: $\Phi_1 = \Psi_{i_1}, \ldots, \Phi_m = \Psi_{i_m} = \Phi$. Это означает, что мы пытаемся доказать в исчислении L_0 секвенции $\vdash \Phi_1$ и $\Phi_1, \ldots, \Phi_i \vdash \Phi_{i+1}$ для всех $i < m$.

Таким образом, доказательство предложения Φ в исчислении L_1 является подсказкой (или стратегией) для доказательства Φ в L_0.

Далее, для получения доказательства Φ в L_1 мы можем обратиться за "подсказкой" к новому исчислению L_2 и т.д.

Заметим, что не обязательно всякий вывод в исчислении L_1 может быть достроен до вывода в исчислении L_0. Достраиваемость в исчислении L_0 вывода, полученного в исчислении L_1, будет иметь место только для определенного класса задач. Следовательно, мы должны иметь различные исчисления L_i для различных классов задач.

В полученной многосортной логической системе исчисление L_0 отвечает за правильность вывода, а исчисления L_i, $i \neq 0$, отвечают за стратегию этого вывода.

§1. Вычисление в логической системе $\mathfrak{L}$

Рассмотрим систему

$$\mathfrak{L} \Leftrightarrow \langle \{ L_i \mid L_i - \text{некоторое исчисление} \}, \subseteq \rangle .$$

Исчисление $L_0 \in \mathfrak{L}$ - логическое описание предметной области; L_0 содержит как логические, так и содержательные аксиомы и правила вывода.

Программа, описывающая вычисление истинности запроса Φ в логической системе $\mathfrak{L}$, состоит из двух частей - из самого

предложения Φ сигнатуры исчисления L_0, а также из описания способа использования системы $\mathfrak{L}$, который состоит в следующем.

1.1. Указание приоритетов запросов к исчислениям $L_i \in \mathfrak{L}$.

В программе указываются ситуации, когда для вывода в исчислении $L \in \mathfrak{L}$ нам нужно обратиться к исчислению $L' \in \mathfrak{L}$ с целью получения подсказок - стратегии вывода в L.

Например, таким указанием может быть следующее: если для доказательства предложения Φ в исчислении L_0 мы перебрали все последовательности из предложений длины n, где n - некоторое заранее заданное число, и ни одна из этих последовательностей не оказалась выводом предложения Φ в L_0, мы должны обратиться к более сильному исчислению L.

Возможно указание иерархии приоритетов обращения к различным исчислениям $L' \in \mathfrak{L}$: если мы не можем добиться успеха в одном исчислении $L' \in \mathfrak{L}$, мы обращаемся к другому исчислению $L'' \in \mathfrak{L}$.

1.2. Запросы к эксперту.

В программе указываются ситуации, когда для вывода в исчислении L нам нужно обратиться к эксперту за дополнительной информацией (подсказкой). При этом в $\mathfrak{L}$ создается новое исчисление $L' \leftrightarrows L \cup \Delta$, где Δ - набор предположений - аксиом и правил вывода - полученных от эксперта. Возможно изменение $L := L \cup \Delta$ вместо создания нового исчисления L'.

1.3. Автоматическое обогащение $\mathfrak{L}$.

В программе указываются ситуации, когда для вывода в исчислении L нужно автоматически породить новое исчисление $L' \notin \mathfrak{L}$; также указывается и способ такого порождения.

Вычисление в $\mathfrak{L}$ происходит следующим образом.

Шаг 0. Начинаем в исчислении L_0 достраивать вывод Γ^0, состоящий из одного предложения $\Phi^0 \leftrightarrows \Phi$, где Φ - предложение, сформулированное в задаче.

Шаг n. Достраиваем вывод $\Gamma^n = \langle \Phi^n_1, \ldots, \Phi^n_{m_n} \rangle$ в исчислении L_{i_n}.

а) Если мы получили вывод $\Gamma^{n+1} = \langle \Phi^{n+1}_1, \ldots, \Phi^{n+1}_{m_{n+1}} \rangle$ в L_{i_n} и $\Phi^{n+1}_{m_{n+1}} = \Phi^n_{m_n}$, то возвращаемся в исчисление L_j, откуда мы попали в L_{i_n}, и полагаем $i_{n+1} \leftrightharpoons j$. Заметим, что, вообще говоря, включения $\{\Phi^n_1, \ldots, \Phi^n_{m_n}\} \subseteq \subseteq \{\Phi^{n+1}_1, \ldots, \Phi^{n+1}_{m_{n+1}}\}$ нам не требуются, а требуется лишь, чтобы было выполнено $\Phi^{n+1}_{m_{n+1}} = \Phi^n_{m_n}$. Это связано с тем, что мы можем доказать $\Phi^n_{m_n}$ еще до того, как мы достроили весь вывод $\Gamma^n = \langle \Phi^n_1, \ldots, \Phi^n_{m_n} \rangle$.

Такая ситуация похожа на правило "приятной неожиданности", введенное Н.Н.Непейводой, - когда мы доказываем утверждение, исходя из некоторых предположений, мы можем доказать его еще до того, как мы использовали все сделанные нами предположения - это называется "приятной неожиданностью". В нашем случае предположения - это "подсказки", что в исчислении L_{i_n} из множества предложений $\{\Phi^n_1, \ldots, \Phi^n_i\}$ выводимо предложение Φ^n_{i+1}. Мы можем доказать $\Phi^n_{m_n}$, воспользовавшись только частью этих подсказок.

б) Если мы получили последовательность формул $\Psi_1, \ldots \ldots, \Psi_k = \Phi^n_{m_n}$, еще не являющуюся выводом в L_{i_n}, и попали в ситуацию пп.1.1, 1.2 или 1.3, то полагаем $L_{i_{n+1}} \leftrightharpoons L_j$, $L_{i_{n+1}} \leftrightharpoons L_{i_n} \cup \Delta$ либо $L_{i_{n+1}} \leftrightharpoons L_{i_n}$ соот -

ветственно; полагаем $m_{n+1} \leftarrows k$ и $\Phi_l^{n+1} \leftarrows \Psi_l$ для всех $l \leq k$.

Вычисления заканчиваются либо тогда, когда получен вывод $\Gamma^k = \langle \Phi_1^k, \ldots, \Phi_{m_k}^k = \Phi \rangle$ в L_0, либо в других, предусмотренных программой случаях (см., например, §5).

§2. Спецификация задачи и предметной области

В этом параграфе мы рассмотрим возможности, которые дает логическая система $\mathcal{L}$ для спецификации задачи и предметной области. Хотя мы и разделяем задачу - предложение, сформулированное в запросе, и предметную область, описываемую логическим исчислением L_0, они тесно связаны между собой. В более широком смысле задача - это вычисление запроса к исчислению L_0: поэтому спецификация предметной области (т.е. задание L_0) - это одновременно и спецификация задачи.

Существуют разные точки зрения на то, насколько точно и полно должна быть сформулирована задача. К.Ф.Самохвалов [1,2] считает, что задача поставлена лишь в том случае, когда человек обладает *исчерпывающей* информацией о том, что ему нужно. Если вы ставите задачу, то у вас есть *алгоритм*, позволяющий точно отличать решение задачи от "нерешения". При таком подходе задача должна быть *точно* и *полностью* специфицирована до начала решения и не может быть изменена в процессе ее решения.

С нашей точки зрения спецификация задачи и предметной области может меняться при решении задачи. В процессе решения задачи может выясниться, что:

во-первых, мы хотим несколько иного, чем предполагали вначале, поскольку мы приобрели новую информацию о предметной области, которую не учитывали раньше;

во-вторых, полученное решение задачи может оказаться совсем не тем, чего мы хотели, в таком случае возникнет необходимость *переформулировать* нашу задачу;

и, наконец, невозможность решить задачу в ее изначальной формулировке может служить причиной к тому, что задачу нужно *ослабить*.

Логическая система $\mathfrak{L}$ дает возможность специфицировать задачу и предметную область не сразу и целиком, до начала работы программы, а постепенно, в процессе решения задачи.

При попытке дать исчерпывающее описание предметной области, мы сталкиваемся со следующими проблемами.

2.1. Непротиворечивость:

- трудно полностью формализовать имеющуюся у эксперта информацию и учесть все возможные следствия сформулированных аксиом;

- наше знание о предметной области состоит из частей,имеющих различную степень достоверности;

- если получена достаточно богатая спецификация предметной области, содержащая много *разнородной* информации, то трудно доказать ее непротиворечивость.

2.3. Полнота:

- не все наши знания о предметной области будут полезны при решении интересующих нас задач, но что именно нам понадобится - заранее неизвестно;

- универсальная, т.е. не ориентированная на класс задач, база данных требует слишком большую память.

2.3. Структура данных:

- структура базы данных должна быть естественной для задачи (класса задач),такую структуру трудно угадать до начала решения этих задач;

- при решении задачи используются как хорошо известные сведения о предметной области,уже много раз применявшиеся при

работе с данным классом задач, так и принципиально новые, учи - тывающие специфику нашей задачи и полезные, быть может, только для нее. Работа с "привычными" знаниями известными методами - это, по существу, рутинная часть решения задачи. Творческая же часть решения - это обработка новой, только что полученной либо раньше не использовавшейся информации, для которой еще нет методов (т.е. стратегий - подсказок) ее использования. Является весьма неудобным, когда на одном логическом уровне представлены как уже совершенно рутинные фрагменты описания предметной области, так и те фрагменты, для работы с которыми требуется творчество;

- требования к удобству и естественности структуры данных не поддаются формализации, в конечном счете эксперт выбирает ту структуру данных, которая ему нравится.

Использование системы $\mathfrak{L}$ позволяет следующим образом решать эти проблемы:

- необходимо доказывать непротиворечивость только для исчисления L_0, при этом L_0 создается именно для удобства доказательства непротиворечивости (и адекватности предметной области) без каких-либо других требований;

- сведения разной степени достоверности могут помещаться экспертом в разные исчисления $L \in \mathfrak{L}$;

- исчисление L_0 (и вся логическая система $\mathfrak{L}$ – как объединение всех $L \in \mathfrak{L}$) не обязательно должно быть полным: во-первых, исчислению L_0 достаточно быть полным *относительно класса решаемых задач* (см. [1,2]), во-вторых, недостающие сведения можно получить от эксперта в процессе решения задачи; в таком случае, правда, предложение из запроса будет обосновано не в исчислении L_0, а в *менее достоверном* исчислении $L \in \mathfrak{L}$;

- структура данных возникает естественным образом при решении интересующего нас класса задач. Этот означает следующее: кроме очень широкого класса задач, которые мы можем себе

помыслить, т.е. специфицировать, существует класс задач, которые в порядке их поступления придется решать с помощью логической системы $\mathfrak{L}$. Второй класс будет несравненно уже, чем первый; в процессе работы программы и взаимодействия с экспертом возникает система $\mathfrak{L}$, являющаяся адекватной структурой данных для класса реально решаемых (более точно - решавшихся) задач. Если в процессе работы с системой $\mathfrak{L}$ класс задач изменится, то вместе с ним, в процессе решения этих задач, изменится и система $\mathfrak{L}$;

- "рутинные" и "творческие" фрагменты описания предметной области могут быть представлены в разных исчислениях $L \in \mathfrak{L}$.

§3. Разрешение логических противоречий в спецификации задачи и предметной области

Как мы уже отмечали, непротиворечивость требуется только от исчисления $L_0 \in \mathfrak{L}$, остальные же исчисления $L \in \mathfrak{L}$, вообще говоря, могут быть противоречивыми.

Если в исчислении $L \in \mathfrak{L}$ обнаружено противоречие, то мы можем сделать следующее.

3.1. Выбрать исчисления $L^1, \ldots, L^k \subseteq L$ такие, что $L^1 \cup \ldots \cup L^k = L$, и в исчислениях L^i, $i \leq k$, противоречие еще не найдено, а затем рассмотреть исчисление $L' \leftarrows \cup C_i$, полученное объединением $C_i \subseteq L^i$ - некоторых следствий исчислений L^i, и дополнить систему $\mathfrak{L}$ новыми исчислениями L^i и L' .

Действительно, если исчисление $L \in \mathfrak{L}$,соответствующее интуиции эксперта, содержит попарно противоречивые формулы $\varphi_1, \ldots, \varphi_k$, то это означает, во-первых, что информация, полученная от эксперта, содержит несколько частей, которые могут быть применены в *разных* ситуациях (т.е. для решения *разных* задач), но которые нельзя использовать в *одной* ситуации (т.е.

для решения *одной* задачи), и, во-вторых, что нас интересуют не эти формулы $\varphi_1, \ldots, \varphi_k$ сами по себе, а только некоторые их следствия.

3.2. Сделать запрос к эксперту и ввести в $\mathfrak{L}$ новое исчисление L', полученное изменением противоречащих утверждений, принадлежащих L; мы можем вместо введения нового исчисления L' изменить $L := L'$.

Заметим, что мы боремся только с той противоречивостью исчислений L из системы $\mathfrak{L}$, которая может быть *обнаружена* при решении *данного класса задач;* другие случаи противоречивости исчислений L *нас не интересуют.*

§4. Распараллеливание вычислений

Использование логической системы $\mathfrak{L}$ дает богатые возможности для распараллеливания вычислений. Мы здесь приведем только некоторые из них.

4.1. Когда мы на шаге n достраиваем вывод $\Gamma^n = \langle \Phi_1^n, \ldots, \Phi_{m_n}^n \rangle$ в исчислении L_{i_n}, мы можем доказывать в L_{i_n} все секвенции $(\vdash \Phi_1^n), \ldots, (\Phi_1^n, \ldots, \Phi_{m_{n-1}}^{n-1} \vdash \Phi_{m_n}^n)$ параллельно.

Кроме того, можно параллельно выводить как предложение Φ, так и $\neg\Phi$.

4.2. При необходимости получить стратегию доказательства в исчислении $L \in \mathfrak{L}$ мы можем параллельно обратиться к нескольким исчислениям $L^1, \ldots, L^k \in \mathfrak{L}$ и затем параллельно достраивать в L выводы, полученные в исчислениях $L^1, \ldots, L^k$.

4.3. Можно параллельно осуществлять вывод в исчислениях $L^1, \ldots, L^k$, достоверность которых разная; в частности, если в $L \in \mathfrak{L}$ недостаточно информации и от пользователя получены

новые сведения Δ, можно параллельно выводить в $L' \leftrightarrows L \cup \Delta$ и в L.

Распараллеливание происходит в реальном времени. Пункты синхронизации: когда в исчислении $L \in \mathfrak{L}$ вывод $\Psi_1, \ldots, \Psi_k$ достроен, все параллельные процессы достраивания этого вывода в L прекращаются.

§5. Приближенное решение задачи

Возможны ситуации, когда точное решение задачи невозможно или необязательно. Тогда мы можем удовлетвориться приближенным решением.

Пусть на множестве формул исчисления L_0 задано рефлексивное, симметричное, но не обязательно транзитивное отношение $\sim$, которое мы понимаем как отношение "приблизительной эквивалентности". Возможны следующие варианты "приблизительного" решения задачи.

5.1. Пусть в исчислении $L_0 \in \mathfrak{L}$ получен вывод $\Psi_1, \ldots, \Psi_m$, причем для некоторого $k \leq m$ и термов $t_1, \ldots, t_n$ выполнено $\Psi_k \sim \Phi(t_1, \ldots, t_n)$, где Φ - формула из задачи. Тогда мы можем закончить программу, считая, что на $t_1, \ldots, t_n$ истинно Φ с точностью $\sim$.

5.2. Для исчислений L_1 и L_2 обозначим $L_1 \subseteq L_2$, если из $L_1 \vdash \Psi$ следует существование $\Psi' \sim \Psi$ с $L_2 \vdash \Psi'$; обозначим $L_1 \sim L_2$, если $L_1 \subseteq L_2$ и $L_2 \subseteq L_1$.

Если $L \subseteq L_0$ и в L получен вывод $\Psi_1, \ldots, \Psi_m = \Phi(t_1, \ldots, t_n)$, то, как и в 5.1, мы имеем истинность Φ на $t_1, \ldots, t_n$ с точностью $\sim$.

5.3. Если нам не удается установить истинность предложения Φ в L_0, мы можем ввести в $\mathfrak{L}$ исчисление $L' \subseteq L_0$ и попытаться доказать Φ в L'. Такая замена будет наиболее удачной в том случае, когда новое исчисление $L' \sim L_0$.

5.4. Если нам удалось доказать предложение Φ в достаточно достоверном исчислении $L \in \mathfrak{L}$, то мы можем считать нашу задачу *приблизительно* решенной, т.е. решенной с *достоверностью исчисления* L.

§6. Решение задачи несколькими экспертами

Использование логической системы $\mathfrak{L}$ дает возможность систематизировать и применять знания, полученные от нескольких экспертов.

6.1. При решении класса задач несколькими пользователями каждый из них может вводить в систему $\mathfrak{L}$ новые исчисления L, внося туда свои личные знания и экспертные оценки.

6.2. Каждый из пользователей может для различных фрагментов своей задачи пользоваться исчислениями $L \in \mathfrak{L}$, введенными разными экспертами, зная их сильные и слабые стороны.

6.3. В процессе решения класса задач выясняется степень достоверности и эффективности различных исчислений $L \in \mathfrak{L}$. В частности, каждое $L \in \mathfrak{L}$, не являясь, вообще говоря, истинным (корректным), может являться истинным для некоторого класса задач.

§7. Модификация логической системы $\mathfrak{L}$ при изменении класса задач

В случае изменения предметной области или класса задач, мы должны изменить и логическую систему $\mathfrak{L}$.

7.1. Мы можем взять $\mathfrak{L}$ как часть новой системы $\mathfrak{L}'$: $\mathfrak{L} \subseteq \mathfrak{L}'$, но $L_0 \neq L_0' \in \mathfrak{L}'$, где L_0' - логическое описание новой предметной области. Те исчисления $L \in \mathfrak{L}$, которые были недостоверны или неэффективны раньше, могут оказаться достоверными и эффективными для новой предметной области или нового класса задач. Эта операция будет более успешной, если система $\mathfrak{L}$ содержит исчисления, "приблизительно

эквивалентные" исчислению L_0', т.е. такие $L \in \mathfrak{L}$, для которых выполнено $L \sqsubseteq L_0'$ либо $L \sim L_0'$.

7.2. Можно сделать "копию" сигнатуры $\mathfrak{L}$ и включить $\mathfrak{L}$ в $\mathfrak{L}'$ так, чтобы их сигнатуры не пересекались; после этого можно использовать $\mathfrak{L}$ как источник информации о том, какие предложения были истинны для старой предметной области (см. работу К.Ф.Самохвалова [3] о двухцветной арифметике).

§8. Связь между различными исчислениями в логической системе $\mathfrak{L}$

Выше мы нигде не отмечали, насколько близкими по вырази - тельной силе должны быть "соседние" исчисления в системе $\mathfrak{L}$, т.е. такие исчисления L' и L'', что для нахождения вывода в исчислении L' мы можем обратиться "за подсказкой" к исчислению L''.

Естественным ограничением было бы следующее: любой логический шаг - т.е. переход, полученный применением одного правила вывода, - в исчислении L'' должен быть обоснован не более чем за n шагов в исчислении L', где n - некоторое заранее фиксированное число. Это требование является *фальсифицируемым;* если мы обнаружили, что оно нарушено для исчислений L' и L'', мы должны ввести в систему $\mathfrak{L}$ новое ис - числение L''' такое, что $L' \subset L''' \subset L''$ и L''' "близко" к L', и затем проверить наше требование для исчислений L''' и L''.

Если указанное требование на логическую систему $\mathfrak{L}$ выполнено, то проблема обоснования в исчислении L_0 вывода, полученного в любом исчислении $L \in \mathfrak{L}$, является алгоритми - чески разрешимой: за конечное число шагов мы можем получить обоснование, либо убедиться в том, что его не существует. В последнем случае мы можем довольствоваться той степенью достоверности, которую нам дает исчисление L.

Заключение

Многосортная логическая система $\mathfrak{L}$ предназначена для того, чтобы автоматизировать работу по созданию непротиворечивой, полной и хорошо (естественным образом, адекватно классу задач) структурированной спецификации предметной области.

Конкретизация предложенного общего описания системы $\mathfrak{L}$ для различных классов задач, ее уточнение и наполнение содержанием заключается в порождении по логике предметной области L_0 системы $\mathfrak{L}$ логик "подсказок", автоматизация такого порождения в первую очередь связана с заданием системы отношений "приблизительной эквивалентности" $\sim$ из §5.

Наконец, заметим, что в каждом конкретном случае проблема построения $\mathfrak{L}$ по L_0 является по существу своему творческой, и поэтому не может быть полностью формализована и автоматизирована. В конечном счете все определяется опытом и знаниями экспертов.

Логическая система $\mathfrak{L}$ ориентирована на проблемы, возникающие в семантическом программировании [4-6]. Формализация системы $\mathfrak{L}$ на языке наследственно-конечных списков дает возможность автоматического синтеза стратегий исполнения Σ-программ.

Автор благодарен К.Ф.Самохвалову, Д.И.Свириденко и Е.Е.Витяеву за полезные обсуждения.

Литература

1. ЕРШОВ Ю.Л., САМОХВАЛОВ К.Ф. О новом подходе к философии математики //Структурный анализ символьных последовательностей. - Новосибирск, 1984. - Вып. 101: Вычислительные системы. - С. 141-148.

2. ЕРШОВ Ю.Л., САМОХВАЛОВ К.Ф. О новом подходе к методологии математики //Закономерности развития современной математики. - М., 1987. - С. 85-105.

3. САМОХВАЛОВ К.Ф. Уточнения обычной интерпретации теорем Гёделя о неполноте и понятия рекурсивной перечислимости //Проблемы логики и методологии науки. - Новосибирск, 1982. -С.42-57.

4. ГОНЧАРОВ С.С., СВИРИДЕНКО Д.И. Математические основы семантического программирования //ДАН СССР. - 1986. - Т. 289, № 6. - С. 1324-1328.

5. GONCHAROV S.S., SVIRIDENRO D.I. Theoretical aspects of Σ-programming //Lect. Notes of Comp. Science. - Berlin a.o., 1986. -Vol.215: Mathematical methods of specification and synthesis of software systems'85. -P. 169-179.

6. GONCHAROV S.S., ERSHOV Yu.L., SVIRIDENKO D.I. Semantic programming //Proc. 10th World Congress Information Processing 86, Dublin, Oct. 1986. - Amsterdam a.o., 1986. -P. 1093-1100.

Поступила в ред.-изд.отд.

12 мая 1991 года