УДК 004.58 DOI: 10.26102/2310-6018/2021.32.1.006

Разработка метрики определения вероятностного расстояния до решения в сложных проблемных областях

Т.Е. Есин

Федеральное государственное автономное образовательное учреждение высшего образования «Тюменский государственный университет», Тюмень, Российская Федерация

***Резюме*:** Основной подход к решению задач на курсах по программированию зачастую состоит из написания и тестирования отдельных частей алгоритма, записанного на том или ином языке. Учащиеся делают несколько попыток сдать задачу в тестирующую систему, каждая из таких попыток отражает текущее состояние решения. Обычно для определения результативности вычисляется среднее количество попыток сдать решение или время, затраченное для получения верной программы. Такие метрики, как правило, неустойчивы, потому что время исправления отдельных ошибок существенно влияет на общее время решения задачи. Также данные метрики не отражают, что именно не понимает учащийся в теоретическом аспекте. В данной статье предлагается метрика, основанная на вероятностном расстоянии между текущим состоянием и правильным решением. В рамках эксперимента группой студентов были решены задачи в онлайн-среде. Все их попытки оценивались по модели алгоритмических компонентов, необходимых для достижения правильного решения. Для создания графа, связывающего программные состояния, использовались цепи Маркова. Предложенная метрика вероятностного расстояния до решения применена к графу для определения расстояний от каждого решения до ближайших правильных. Результаты показали, что данная метрика полезна при определении расстояния, если путь к правильному решению был типичен и согласовывался с изученным теоретическим материалом. В статье предлагаются детали реализации метрики вероятностных расстояний до решения и план дальнейших исследований, основанных на текущих наблюдениях.

***Ключевые слова*:** интеллектуальные образовательные системы, курсы программирования, обратная связь, анализ образовательных данных, учебная аналитика

***Для цитирования*:** Есин Т.Е. Разработка метрики определения вероятностного расстояния до решения в сложных проблемных областях. *Моделирование, оптимизация и информационные технологии.* 2021;9(1). Доступно по: https://moitvivt.ru/ru/journal/pdf?id=880 DOI: 10.26102/2310-6018/2021.32.1.006

Development of the metric of determination probabilistic distance to solution in difficult problem areas

T.E. Esin

Federal State Autonomous Educational Institution of Higher Education «Tyumen State University», Tyumen, Russian Federation

***Abstract*:** The main approach to solving problems in programming courses often consists of writing and testing individual parts of an algorithm written in a particular language. Students make several attempts to submit the problem to the testing system, each of these attempts reflecting an individual solution state. It is common practice to determine the student performance, the average number of submissions to pass the solution or focusing on the time taken to complete the problem correctly is calculated. Such metrics are

usually unstable because the time to correct individual errors significantly affects the total time to solve the problem. Also, these metrics do not reflect what the student does not understand in the theoretical aspect. This article proposes a metric based on the probabilistic distance between an observed student solution and a correct solution. As part of the experiment, a group of students solved problems in an online programming environment. Their submissions were evaluated against a model of the algorithmic component necessary for a correct solution. A Markov Model was used to generate a problem state graph, connecting program states. The proposed metric of the probabilistic distance to the solution was applied to the graph to determine the distances from each solution to the nearest correct ones. The results showed that this metric is useful in determining the distance if the path to the correct solution was typical and consistent with the studied theoretical material. The article provides details of the implementation of the metric of probabilistic distances to the solution and a plan for further research based on current observations.

***Keywords*:** intelligent tutoring system, programming courses, automatic feedback, educational data mining, learning analytics

***For citation*:** Esin T.E. Development of the metric of determination probabilistic distance to solution in difficult problem areas. *Modeling, optimization, and information technology.* 2021;9(1). Available from: https://moitvivt.ru/ru/journal/pdf?id=880 DOI: 10.26102/2310-6018/2021.32.1.006 (In Russ).

Введение

Современные методы сбора и классификации позволяют создавать все более сложные пространства решений для их автоматической проверки и оценки. Например, в статьях рассматривается автоматическая оценка математических доказательств [1], эссе [2] и даже сложных компьютерных программ [3], которые также могут быть оценены и проанализированы на предмет полноты решения. При решении программной задачи, новички, как правило, пытаются сделать несколько правок и немного продвинуться в решении, вместо того чтобы отправлять полное законченное решение в проверяющую систему. Это позволяет использовать данные программных *состояний* для формирования автоматической обратной связи по решению [4]. Несмотря на то, что модели, стратегии и механизмы обратной связи, используемые в каждой области, отличаются, исследователям важно оценивать успеваемость и сравнивать условия исследований для совершенствования обучающих систем.

В сложных проблемных областях, таких как программирование, учащиеся могут отправлять такие решения, недочеты в которых не имеют прямого отношения к конкретным результатам обучения по данной задаче [5]. Например, при решении задачи студент может бороться с ошибкой компиляции из-за неверно расставленных или отсутствующих скобок, что не относится к пониманию текущего материала. В рамках данного исследования предполагается, что классические показатели успеваемости на занятиях, такие как время получения полного решения задачи или количество попыток — слишком грубые, чтобы различать концептуальное непонимание и борьбу синтаксическими ошибками (это требует, как времени, так и дополнительных попыток для получения синтаксически верной программы). В данной статье представлена метрика Вероятностного расстояния до решения (далее – ВРР), описана ее реализация и применение для оценки успеваемости на занятии по программированию. По набору данных выделены случаи, где ВРР может помочь разобраться в неправильных *состояниях* решения и предложить пути их исправления.

Материалы и методы

Количество заявок и количество времени, необходимое для достижения правильного состояния, были использованы в системе, направленной на улучшение математических оценок [6]. Однако эти метрики не так эффективны в областях решения сложных задач из-за разнообразия стратегий, используемых для решения проблем, и сложности их объединения [7]. Митрович построил ITS для SQL, где используется больше 600 условий для обеспечения точных и полезных советов для обучающихся [8]. Менцель и Ли также описывали подходы построения обучающих систем на основе ограничений в «плохо определенных областях» [9]. Однако, оба эти подхода требуют от преподавателя или автора обучающей системы задавать условия вручную, что является очень трудоемким и фактически нереализуемым целиком в таких широких областях как программирование. Метрика, предлагаемая в этой статье, улучшает эти модели путем изучения более детальных аспектов программных состояний в потенциально-автоматическом режиме.

Осенью 2020 года, восемнадцать студентов решили 4 задачи по программированию в онлайн-системе. Задачи были представлены в одинаковом порядке, по каждой отправке решения в проверяющую систему был записан текст программы, время начала и завершения работы над задачей, а также обратная связь, которая была предоставлена по решению. Представления участников эксперимента были оценены с использованием модели BPP. Участники сгенерировали 354 *состояния*, которые были преобразованы в 63 различных эталонных состояния решения.

Для каждого состояния программы код участника эксперимента транслировался в вектор двоичных признаков, представляющих включение свойств семантической программы, результатов корректности и успеха компиляции. Особенности семантической программы фокусируются на включении алгоритмических компонентов, таких как структура цикла, структура решения, использование переменной управления циклом в доступе к массиву и включение оператора возврата. Собранные данные представляют собой модель программирования в виде временного ряда в двоичном пространстве большого размера. Алгоритмы, требуемые четырьмя задачами, были похожи с точки зрения требуемых функций. Эта модель требуемых алгоритмических элементов и использует идентификацию использования семантических структур, а также тестирование на корректность с использованием модульных тестов.

На Рисунке 1 представлены пути двух участников, начавших с пустого программного состояния Start (S) и пришедших к правильному решению F. Состояние A представляет код, который имеет все правильные алгоритмические компоненты, является компилируемым, но не возвращает соответствующее значение. Состояние B представляет код, который имеет все необходимые алгоритмические компоненты, но не компилируется, поэтому протестировать его не представляется возможным.

Участник #5 исправил ошибку (шаг 2), затем синтаксическую ошибку (шаг 3) и, наконец, еще одну ошибку (шаг 4), которая привела к правильному решению. Участник #12, с другой стороны, первоначально представил код, который не содержал оператора возврата, что указывает на непонимание того, как работают функции.

Состояние C представляет наблюдаемое модельное состояние, в котором три характеристики помечены как неверные для отправки. Затем участник внес небольшое изменение, приведшее к тому же состоянию модели для кода (шаг 2), затем добавил оператор возврата на основе сообщения компиляции (шаг 3) и, наконец, исправил еще одну

ошибку компиляции и представил правильное решение (шаг 4). Хотя эти два участника имеют одинаковое количество отправок решения, причины и характер этих состояний очень различны и указывают на имеющиеся у участника #12 заблуждения.

Поскольку каждая отправка может представлять несколько правок или этапов в процессе решения проблемы, подсчет состояний как мера ошибок не является достаточно информативной, чтобы выразить разницу между учащимися.

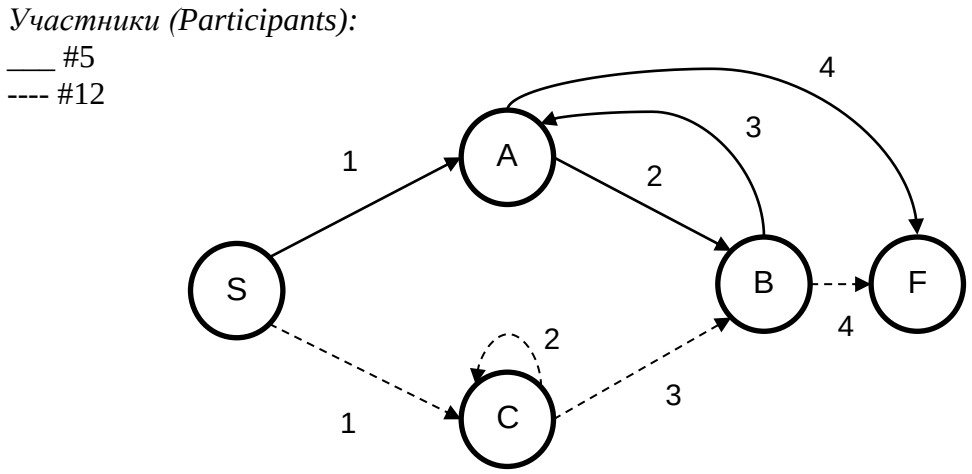

Рисунок 1. Пути к решению двух участников эксперимента
Figure 1. Solving ways of two experiment participants

Традиционные показатели эффективности

Студенты, как правило, сокращают общее время, затрачиваемое на решение задач, когда проходят через череду ошибок (см. Таблицу 1).

	Задача	Среднее / Медиана	Мин	Макс	Отклонение
Количество попыток	1	4.06 / 3	1	14	3.13
	2	7.44 / 4	1	49	11.35
	3	4.56 / 2	1	23	5.71
	4	3.61 / 3	1	9	2.45
Время, с	1	625 / 452	93	2121	510
	2	571 / 364	134	1795	490
	3	437 / 331	103	1438	349
	4	363 / 315	53	1199	279

Таблица 1. Классическая статистика по решениям задач
Table 1. Traditional statistics for solving problems

Уровень способностей отдельных участников значительно варьировался: некоторые участники представляли окончательные решения с минимальными изменениями с первой попытки, в то время как другие участники проходили через множество неправильных состояний модели, прежде чем прийти к правильному решению. Традиционные метрики

измерения можно использовать для разделения участников на две группы: с высокими показателями (студенты, которые смогли быстро решить проблемы) и с низкими показателями (студенты, которым нужно время и несколько попыток решить проблему).

Из семи студентов, которым требуется более шести отправок для решения хотя бы одной задачи, только у двух в среднем было менее шести отправок на задачу. Этим семи ученикам также требовалось более 500 секунд для решения их проблемы, кроме двух упомянутых выше, которые имеют отдельные выбросы выше этой линии. Эта непересекающаяся группировка предполагает, что мы можем подразделить группу с низкими показателями на учеников, которые показали одинаково плохие результаты, и учеников, которые пытались решить только определенную задачу.

Одиннадцать лучших учеников в среднем делали четыре или меньше отправок, чтобы получить правильный ответ, и все завершали задачи в среднем за 400 секунд (6,66 минут), за исключением одного студента, которому потребовалось почти восемнадцать минут, чтобы закончить первую задачу, но для этого потребовалась только одна отправка.

Результаты и их обсуждение

Чтобы сделать обобщения о том, как состояния программы соответствуют успеваемости учащихся и другим скрытым факторам, таким как обучение, все пути представления учащихся для каждой проблемы объединили в сеть (см. Рисунок 2).

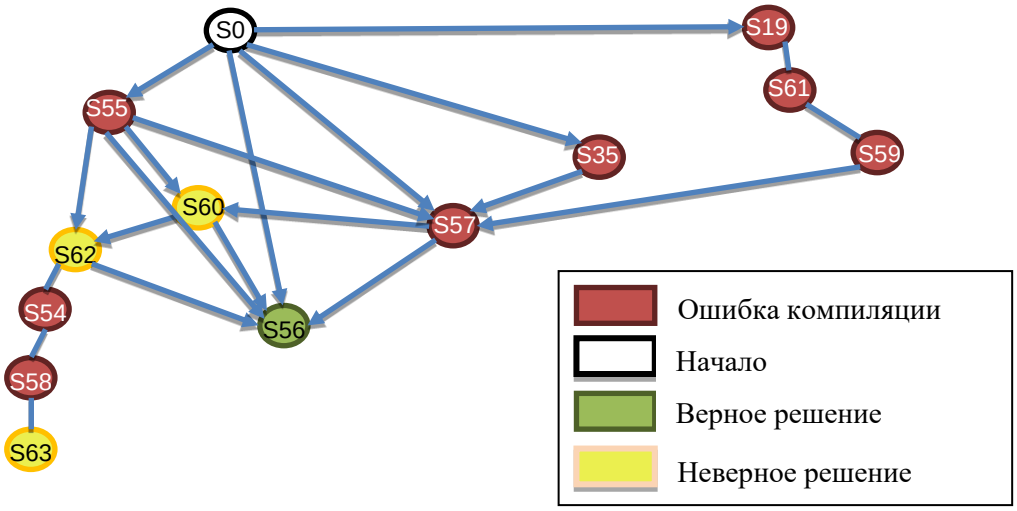

Рисунок 2. Граф пространства решений 4-й задачи
Figure 2. Solution space graph of 4th problem

Узлы $S_1 \ldots S_{n-1}$ это всевозможные программные состояния, ребра – наблюдаемые переходы между состояниями.

Для каждого узла мы используем наши наблюдения, чтобы вычислить вероятности перехода методом максимального правдоподобия (ММП) к каждому другому узлу. Учитывая количество наблюдаемых переходов из состояния x в состояние y ($T_{x,y}$) мы оцениваем вероятность нахождения в состоянии y в момент времени t с помощью ММП:

$$\hat{p}(S_y(t)) = P(S_y(t) \mid S_x(t-1)) = \frac{T_{x,y}}{\sum_i T_{x,i}} \quad (1)$$

Это эквивалентно оценке цепи Маркова с историей из одного состояния. Переходы между состояниями могут быть представлены лучше марковским процессом высшего порядка; тем не менее, наш текущий набор данных слишком мал, чтобы обеспечить мощность для анализа не только первого порядка.

Моделируя каждое ребро как вероятность перехода и расстояние между состояниями, используется набор линейных уравнений для вычисления среднего расстояния от каждого состояния до конечного состояния (успешного завершения).

- N-1 промежуточных состояний $S_1 ... S_{n-1}$ и конечное состояние E;
- Каждое состояние S, имеющее переходные вероятности $P_{s,1} \ldots P_{s,n-1}$ и $P_{s,e}$;
- Переходные расстояния $D_{s,1} \ldots D_{s,n-1}$ и $D_{s,e}$.

Система линейных уравнений для среднего расстояния до конечного состояния $d_e(S)$:

$$\begin{aligned} d_e(S_1) &= \left[\sum_{s=1}^{n-1} P_{1,s}(D_{1,s} + d_e(S_s))\right] + P_{1,e}D_{1,e} \\ d_e(S_2) &= \left[\sum_{s=1}^{n-1} P_{2,s}(D_{2,s} + d_e(S_s))\right] + P_{2,e}D_{2,e} \\ &\ldots \\ d_e(S_{n-1}) &= \left[\sum_{s=1}^{n-1} P_{n-1,s}(D_{n-1,s} + d_e(S_s))\right] + P_{n-1,e}D_{n-1,e} \\ d_e(E) &= 0 \end{aligned} \quad (2)$$

Для случая, когда интересует только среднее число представлений в конечное состояние, каждый $D_{x,y} = 1$, вычисление упрощается в систему скалярных произведений:

$$\begin{aligned} d_e(S_1) &= \vec{P}_1 \bullet d_e(\vec{S}) + 1 \\ d_e(S_2) &= \vec{P}_2 \bullet d_e(\vec{S}) + 1 \\ &\ldots \\ d_e(S_{n-1}) &= \vec{P}_{n-1} \bullet d_e(\vec{S}) + 1 \\ d_e(E) &= 0 \end{aligned} \quad (3)$$

Есть несколько вариантов алгоритма составления данной метрики, которые требуется изучить на большом наборе данных. Например, можно по-разному обрабатывать переходы между состояниями, которые различаются конкретными характеристиками (например, компилируемостью). Такие переходы могут указывать на различную скрытую информацию об обучающихся (например, непонимание определенной концепции).

Применение BPP

Метрика вероятностного расстояния до решения, сопровождаемая графиком переходов, является богатым источником информации о путях, по которым участники следовали, чтобы найти правильное решение. На Таблице 2 иллюстрируются наблюдаемые модельные состояния в двоичном векторе, а также BPP для каждого состояния. В задаче 4 S56 — это состояние решения, а S0 — начальное состояние. Посмотрев на BPP в сочетании с графиком состояния программы, мы можем выявить более продуктивные правки участников.

	F2	F4	F6	F7	F8	F9	F10	F11	F12	F13	F14	Расстояние
S0												3,67
S19												7,78
S35												4,78
S54												4,00
S55												2,99
S56												0,18
S57												4,43
S58												3,00
S59												4,78
S60												2,17
S61												6,78
S62												3,09
S63												3,54

Таблица 2.Модельные состояния в двоичном векторе
Table 2. Model states in binary vector

Например, первая заявка одного участника была отмечена как S55 (BPP 2.99), а следующим состоянием было S57 (BPP 4.43). Это редактирование будет менее продуктивным, поскольку оно приведет к переход к состоянию с большим вероятностным числом представлений, необходимых для получения правильного решения. Благодаря возможности включения в алгоритм терминов для синтаксических, но не модельных изменений (т.е., два идентичных состояния, за исключением ошибки компиляции, не будут учитываться как полный шаг), BPP может быть адаптирован для фокусирования на переходах состояния модели, которые указывают на неправильные представления или другие, основанные на модели, цели исследователя.

Заключение

В рамках этих ранних результатов были определены модели состояний на продуктивных и непродуктивных путях. Используя фактические значения вероятностного расстояния до решения, мы можем определить, производит ли студент продуктивное редактирование или заблуждается. Редактирование, приводящее к наблюдаемому состоянию с более высоким BPP, чем предыдущая отправка, указывает на отклонение от правильного ответа. Эти сведения могут быть неоценимы для составителей курсов, которые стремятся разработать инструменты обратной связи и поддержки для сложных проблемных областей. Преподавателям программирования и студентам данные, полученные метрикой, могут дать более ценную обратную связь, чем компилятор, поэтому использование BPP в составе тестирующей системы, предоставление подсказок на основе анализа данных метрики – отличный способ поддержки обучения на базовых курсах изучения

программирования. Интеграция метрики в систему тестирования планируется в дальнейших исследованиях.

ЛИТЕРАТУРА

1. Barnes T., Stamper J. Automatic Hint Generation for Logic Proof Tutoring Using Historical Data. *Educational Technology & Society*. 2010;13(1):3-12. Доступно по: https://www.j-ets.net/collection/published-issues/13_1 (дата обращения 21.01.2021).
2. Valenti S., Neri F. An Overview of Current Research on Automated Essay Grading. *Journal of Information Technology Education*. 2013;2:319–330. Доступно по: https://www.informingscience.org/Publications/331 DOI: 10.28945/331 (дата обращения 21.01.2021).
3. Ihantola P., Ahoniemi T., Karavirta V. Review of Recent Systems for Automatic Assessment of Programming Assignments. *10th Koli Calling International Conference on Computing Education Research*. 2010. Доступно по: https://dl.acm.org/doi/10.1145/1930464.1930480 DOI: 10.1145/1930464.1930480 (Дата обращения 21.01.2021).
4. Есин Т.Е., Глухих И.Н. Автоматизация предоставления персонализированной обратной связи на курсах изучения программирования. *Моделирование, оптимизация и информационные технологии.* 2019;7(1). Доступно по: http://moitvivt.ru/journal/pdf?id=589. DOI: 10.26102/2310-6018/2019.24.1.043 (дата обращения: 21.01.2021).
5. Jadud M. A First Look at Novice Compilation Behaviour Using BlueJ. *Computer Science Education*. 2005;15(1):25–40. Доступно по: https://www.tandfonline.com/doi/abs/10.1080/ DOI: 10.1080/08993400500056530 (Дата обращения 21.01.2021).
6. Feng M., Heffernan N. Predicting State Test Scores Better with Intelligent Tutoring Systems: Developing Metrics to Measure Assistance Required. *8th International Conference on Intelligent Tutoring Systems*. 2006. Доступно по: https://dl.acm.org/doi/10.1007/11774303_4 DOI: 10.1007/11774303_4 (Дата обращения: 21.01.2021).
7. Lane H., Vanlehn K. Intention-Based Scoring: An Approach to Measuring Success at Solving the Composition Problem. *36th SIGCSE technical symposium on Computer science education*. 2005. Доступно по: https://dl.acm.org/doi/10.1145/1047124.1047471 DOI: 10.1145/1047124.1047471 (Дата обращения: 21.01.2021).
8. Mitrovic A. An Intelligent SQL Tutor on the Web. *International Journal of Artificial Intelligence in Education*. 2003;13(2-4):173–197. Доступно по: https://iaied.org/journal/963 (Дата обращения 21.01.2021).
9. Le N., Menzel W. Using Constraint-Based Modelling to Describe the Solution Space of Ill-defined Problems in Logic Programming. *Advances in Web Based Learning (ICSL 2007)*. 2007. Доступно по: https://link.springer.com/chapter/10.1007/978-3-540-78139-4_33 DOI: 10.1007/978-3-540-78139-4_33 (Дата обращения 21.01.2021).

REFERENCES

1. Barnes T., Stamper J. Automatic Hint Generation for Logic Proof Tutoring Using Historical Data. Educational Technology & Society. 2010;13(1):3-12. Available at: https://www.j-ets.net/collection/published-issues/13_1 (accessed 21.01.2021).

2. Valenti S., Neri F. An Overview of Current Research on Automated Essay Grading. Journal of Information Technology Education. 2013;2:319–330. Available at: https://www.informingscience.org/Publications/331 DOI: 10.28945/331 (accessed 21.01.2021).
3. Ihantola P., Ahoniemi T., Karavirta V. Review of Recent Systems for Automatic Assessment of Programming Assignments. 10th Koli Calling International Conference on Computing Education Research. 2010. Available at: https://dl.acm.org/doi/10.1145/1930464.1930480 DOI: 10.1145/1930464.1930480 (accessed 21.01.2021).
4. Esin T.E., Glukhikh I.N. Automation of Personalized Feedback in the Programming Studies Courses. *Modeling, Optimization and Information Technology*. 2019;7(1). Available at: http://moitvivt.ru/journal/pdf?id=589. DOI: 10.26102/2310-6018/20 (In Russ) DOI: 10.26102/2310-6018/2019.24.1.043 (accessed 21.01.2021).
5. Jadud M. A First Look at Novice Compilation Behaviour Using BlueJ. *Computer Science Education*. 2005;15(1):25–40. Available at: https://www.tandfonline.com/doi/abs/10.1080/ DOI: 10.1080/08993400500056530 (accessed 21.01.2021).
6. Feng M., Heffernan N. Predicting State Test Scores Better with Intelligent Tutoring Systems»: Developing Metrics to Measure Assistance Required. *8th International Conference on Intelligent Tutoring Systems*. 2006. Available at: https://dl.acm.org/doi/10.1007/11774303_4 DOI: 10.1007/11774303_4 (accessed 21.01.2021).
7. Lane H., Vanlehn K. Intention-Based Scoring: An Approach to Measuring Success at Solving the Composition Problem. *36th SIGCSE technical symposium on Computer science education*. 2005. Available at: https://dl.acm.org/doi/10.1145/1047124.1047471 DOI: 10.1145/1047124.1047471 (accessed 21.01.2021).
8. Mitrovic A. An Intelligent SQL Tutor on the Web. *International Journal of Artificial Intelligence in Education*. 2003;13(2-4):173–197. Available at: https://iaied.org/journal/963 (accessed 21.01.2021).
9. Le N., Menzel W. Using Constraint-Based Modelling to Describe the Solution Space of Ill-defined Problems in Logic Programming. *Advances in Web Based Learning (ICSL 2007)*. 2007. Available at: https://link.springer.com/chapter/10.1007/978-3-540-78139-4_33 DOI: 10.1007/978-3-540-78139-4_33 (accessed 21.01.2021).

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Есин Тимофей Евгеньевич, аспирант, кафедра информационных систем,
ФГАОУ ВО «Тюменский государственный университет», Тюмень, Российская Федерация
e-mail: tesin@fmschool72.ru

Timofei E. Esin, Phd Student, Information systems Department, Federal State Autonomous Educational Institution of Higher Education «Tyumen State University», Tyumen, Russian Federation