ВЕСТНИК
НОВОСИБИРСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

Научный журнал
Основан в ноябре 1999 года

Серия: Информационные технологии 2022. Том 20, № 1

СОДЕРЖАНИЕ

ISSN 1818-7900 (Print). ISSN 2410-0420 (Online)
Вестник НГУ. Серия: Информационные технологии. 2021. Том 19, № 4
Vestnik NSU. Series: Information Technologies, 2021, vol. 19, no. 4

ISSN 1818-7900 (Print). ISSN 2410-0420 (Online)
Вестник НГУ. Серия: Информационные технологии. 2022. Том 20, № 1
Vestnik NSU. Series: Information Technologies, 2022, vol. 20, no. 1

VESTNIK
NOVOSIBIRSK STATE UNIVERSITY

Scientific Journal
Since 1999, November
In Russian

Series: Information Technologies 2022. Volume 20, № 1

CONTENTS

ISSN 1818-7900 (Print). ISSN 2410-0420 (Online)
Вестник НГУ. Серия: Информационные технологии. 2022. Том 20, № 1
Vestnik NSU. Series: Information Technologies, 2022, vol. 20, no. 1

Editor in Chief M. M. Lavrentiev

Vice-Editor A. V. Avdeev

Executive Secretary D. P. Iksanova

Editorial Board of the Series

I. V. Bychkov, professor, academician (Irkutsk), *B. M. Glinsky*, professor (Novosibirsk)
A. N. Gorban, professor (Lester, GB), *E. P. Gordov*, professor (Tomsk)
B. S. Dobronets, professor (Krasnoyarsk), *A. M. Elizarov*, professor (Kazan)
G. N. Erokhin, professor (Kaliningrad), *A. I. Kamyshnikov*, professor (Khanty-Mansijsk)
G. P. Karev, professor (Maryland, USA), *N. A. Kolchanov*, professor, academician (Novosibirsk)
M. M. Lavrentjev, professor (Novosibirsk), *V. E. Malyshkin*, professor (Novosibirsk)
N. N. Mirenkov, professor (Aizu, Japan), *N. M. Oskorbin*, professor (Barnaul)
D. E. Palchunov, professor (Novosibirsk), *T. Pizansky*, professor (Ljubljana, Slovenia)
V. P. Potapov, professor (Kemerovo), *O. I. Potaturkin*, professor (Novosibirsk)
V. A. Serebryakov, professor (Moscow), *A. V. Starchenko*, professor (Tomsk)
S. I. Smagin, professor, corresponding member of RAS (Khabarovsk)
D. A. Tusupov, professor (Astana, Kazakhstan)
V. V. Shajdurov, professor, corresponding member of RAS (Krasnoyarsk)
Yu. I. Shokin, professor, academician (Novosibirsk)

The journal is published quarterly in Russian since 1999
by Novosibirsk State University Press

The address for correspondence

Faculty of Information Technologies, Novosibirsk State University

1 Pirogov Street, Novosibirsk, 630090, Russia

Tel. +7 (383) 363 42 46

E-mail address: inftech@vestnik.nsu.ru

On-line version: http://elibrary.ru

ISSN 1818-7900 (Print). ISSN 2410-0420 (Online)
Вестник НГУ. Серия: Информационные технологии. 2022. Том 20, № 1
Vestnik NSU. Series: Information Technologies, 2022, vol. 20, no. 1

Научная статья

УДК 004.03 + 630*3
DOI 10.25205/1818-7900-2022-20-1-5-17

Использование систем поддержки принятия решений в лесной промышленности: экологический аспект

Михаил Юрьевич Васенёв

Поволжский государственный технологический университет
Йошкар-Ола, Россия
AspIVS16.20@gmail.com, https://orcid.org/0000-0003-3405-8342

Аннотация

Рассматриваются вопросы применения информационных технологий в лесной промышленности. Отмечается, что данной отрасли требуется техническая и технологическая модернизация в рамках концепции «Индустрия 4.0». Описываются методы, которые дают возможность снизить негативное влияние лесозаготовительных машин на окружающую среду. Акцент делается на том, что максимальный урон почвогрунтам наносят колесные движители лесной техники. Предлагается использовать системы поддержки принятия решений (СППР) для улучшения и упрощения процесса контроля над происшествиями на территории, предназначенной для рубки леса. Дается структура такой системы, описываются основные составные части, принципы их функционирования. Предлагаются некоторые технические показатели для оценки изменений вследствие внедрения описываемой СППР, среди них: время реакции на происшествие с момента его выявления, время анализа ситуации и время ликвидации загрязнения, а также глубина колеи, образуемая лесной машиной. Проводится моделирование ряда ситуаций, подчеркивающих эффективность применения СППР, таких как: 1) можно ли сократить время очищения почвы от нефтепродуктов; 2) можно ли предотвратить превышение необходимого уровня глубины колеи. Рассматривается влияние СППР на существующие показатели «Индустрии 4.0». Обоснованы выводы о целесообразности и эффективности использования СППР в лесной промышленности. Отмечается, что применение СППР в определенной степени позволит предотвратить эрозию лесных почв на лесосеке, а также предупредить снижение плодородия, вызванного нерациональным использованием и химическим загрязнением.

Ключевые слова

технологическая модернизация, лесные машины, СППР, очищение почвы, глубина колеи

Для цитирования

Васенёв М. Ю. Использование систем поддержки принятия решений в лесной промышленности: экологический аспект // Вестник НГУ. Серия: Информационные технологии. 2022. Т. 20, № 1. С. 5–17. DOI 10.25205/1818-7900-2022-20-1-5-17

Decision Support Systems Utilization in Forestry: Environmental Aspect

Mikhail Yu. Vasenev

Volga State University of Technology
Yoshkar-Ola, Russian Federation
AspIVS16.20@gmail.com, https://orcid.org/0000-0003-3405-8342

Abstract

In this paper, the actuality of utilization of informational technology in forestry is underlined. There is noted, that the technical and technological modernization within the frameworks of “Industry 4.0” for this sector is required. There

© Васенёв М. Ю., 2022

are described methods, which give the potential to reduce forest machines' negative influence on the environment. There is emphasized, that the maximal damage to soils is inflicted by wheel propellers. There is proposed to use decision support systems (DSS) for the control process improvement and simplification on the territory, intended for cutting. There is given the system structure, main elements, and theirs principles of functioning are described. There are proposed some performance characteristics for the measurement of changes in consequence of the implementation of the depicted DSS, among them: response time on an incident from the moment of detection, situation analysis time, clearing time, and track depth, created by forest machine. There is made the simulation of an array of situations, stressing an efficiency of DSS use, such as: a) Can one reduce a soil purification time from oil products? b) Can one prevent an exceeding of the required track depth level? There is considered an impact of this DSS on existing indices of "Industry 4.0". There are substantiated conclusions about the reasonability and efficiency of this DSS usage in forestry. There is noticed, that the DSS application to some extent allows preventing the erosion of forest soils, also to warn a diminishing of fertility, created by the irrational use and chemical pollution.

Keywords

technological modernization, forest machines, DSS, soil purification, track depth

For citation

Vasenev M. Yu. Decision Support Systems Utilization in Forestry: Environmental Aspect. *Vestnik NSU. Series: Information Technologies*, 2022, vol. 20, no. 1, p. 5–17. (in Russ.) DOI 10.25205/1818-7900-2022-20-1-5-17

Введение

Бурный рост промышленности в XX в. оказал сильное негативное влияние на состояние природы. Данный момент актуализировал стремление мирового сообщества к более бережному отношению к окружающей среде. Так, в обзоре глобальных рисков 2021 г. от Всемирного экономического форума (WEF) на 1-м месте – инфекционные заболевания, а на 2-м – провал борьбы с изменением климата и другие экологические риски [1]. Что может помочь человеку удовлетворить возросшие требования к экологичности, взять под контроль климатические изменения? Ответ напрашивается сам собой – высокие технологии. Именно на них и делает упор концепия «Индустрия 4.0». Напомним, что подразумевает под собой данное понятие. Грубо говоря, это синоним *цифровой трансформации*, *интеграции* информационно-коммуникационных технологий в функционирование определенной отрасли экономики государства (промышленность, здравоохранение и пр.). Подробнее о технологиях «Индустрии 4.0» см. на сайте SAP [2].

Лесная промышленность, как ни одна другая, порождает множество экологических проблем, среди них: загрязнение почв нефтепродуктами, повреждение почв движителями лесной техники, захламление мест заготовок вторсырьем и т. д. Существуют различные подходы к решению таких проблем, но в рамках этой статьи мы остановимся на применении систем поддержки принятия решений (СППР) как способе борьбы с данными происшествиями.

Итак, *целью* данной работы является разработка теоретических положений для СППР, позволяющих снизить негативное влияние лесозаготовительных машин на окружающую среду.

В связи с этим необходимо решить следующие *задачи*:

1) дать краткую характеристику методам, на основе которых строится СППР;

2) привести принципы построения, описать архитектуру предлагаемой СППР для оценки негативного влияния лесных машин на окружающую среду;

3) предложить технические показатели для оценки изменений вследствие внедрения СППР, произвести компьютерное моделирование определенных ситуаций;

4) рассмотреть влияние описываемой СППР на показатели «Индустрии 4.0».

[1] https://www.vedomosti.ru/partner/articles/2021/08/27/883464-pochemu-ekologiya

[2] https://www.sap.com/cis/insights/what-is-industry-4-0.html

Описание методов

Напомним, что в работе [1] были предложены методы, позволяющие *снизить негативное влияние лесозаготовительных машин (ЛЗМ) на окружающую среду*. Далее кратко рассмотрим их.

Первый метод (рис. 1) отличается от существующих тем, что позволяет избавиться от необходимости постоянного использования грузовых весов для оценки веса перевозимых сортиментов, что, в свою очередь, необходимо для получения значений номинального давления машины на грунт (в случае с форвардером). Более того, он не требует применения дополнительного дорогостоящего оборудования, как LiDAR, беспилотных летательных аппаратов с фотографическим оборудованием, см. [2; 3]. Помимо основной задачи, данный метод позволит в некоторой степени сократить длительность рабочего цикла машины, что приведет к дополнительным экономическим выгодам.

Второй метод (рис. 2) отличается от существующих тем, что для определения количества утекших нефтепродуктов не требуется дополнительного оборудования и / или персонала, осуществляющего наблюдение за данными происшествиями. Контроль осуществляется только на основании информации, полученной от датчиков, установленных в каждой лесозаготовительной машине (например, датчик уровня гидравлической жидкости, дизельного топлива и т. п.).

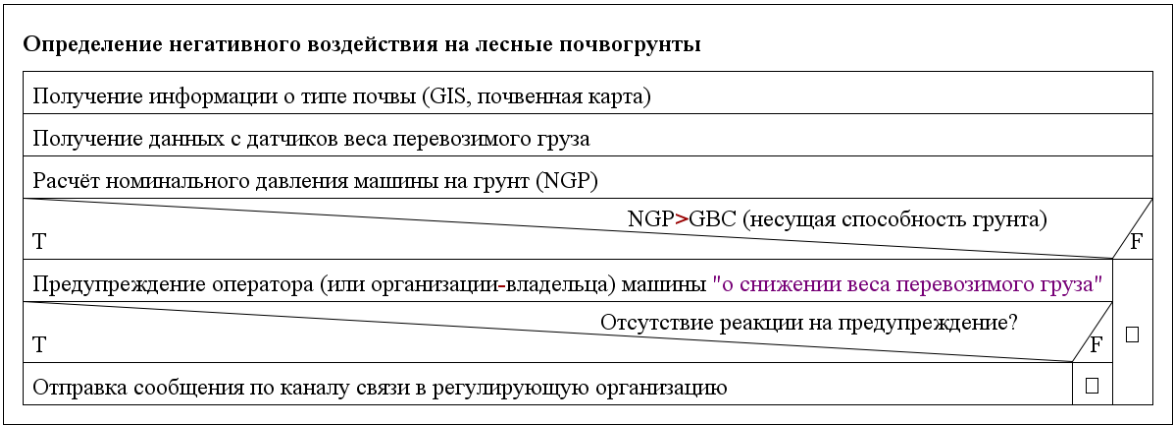

Рис. 1. Диаграмма Насси – Шнейдермана для первого метода
Fig. 1. Nassi – Shneiderman diagram for the first method

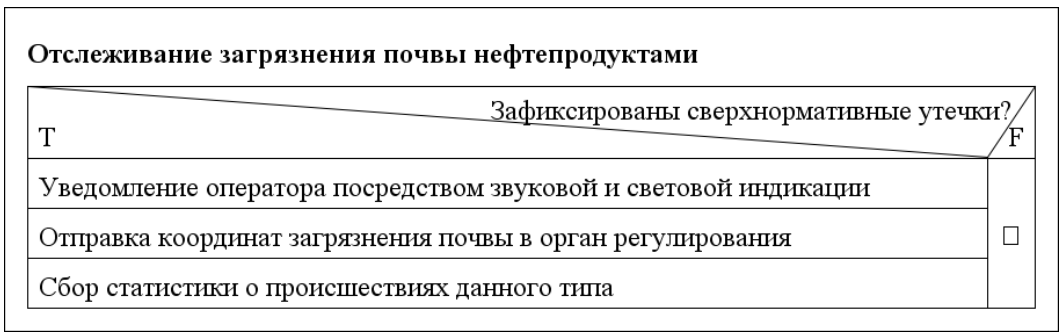

Рис. 2. Диаграмма Насси – Шнейдермана для второго метода
Fig. 2. Nassi – Shneiderman diagram for the second method

Описание структуры СППР

Предполагая, что читатели данного журнала уже в определенной степени знакомы с тем, что представляет собой СППР, рассмотрим лишь конспективно основные положения, касающиеся этого понятия.

Согласно определению, данному В. А. Геловани и др. в [4], СППР – это компьютерная система, которая должна предоставлять лицу, принимающему решение, средства для использования моделей и данных, чтобы распознавать, понимать и формулировать задачи.

В зависимости от способа воздействия на процесс принятия решения различают следующие СППР: *пассивные* (предоставляют лишь информацию для принятия решений), *активные* (предлагают альтернативные готовые варианты) и *комбинированные* (ЛПР может вносить изменения в предложенное системой решение и согласовывать их до обретения им оптимальной формы) [3].

Внедрение СППР позволяет:
- упростить и ускорить процесс принятия решений;
- наладить систему контроля в организации;
- предвидеть и предотвратить вероятные убытки;
- более эффективно оперировать данными;
- проводить прогнозирование и выявлять скрытые риски, и т. п.

В состав типовой СППР входят следующие компоненты:
- модуль хранения и обработки данных;
- модуль хранения и использования моделей;
- интерфейсный модуль.

Далее перейдем к описанию структуры СППР (рис. 3), в основе которой лежат вышеописанные методы.

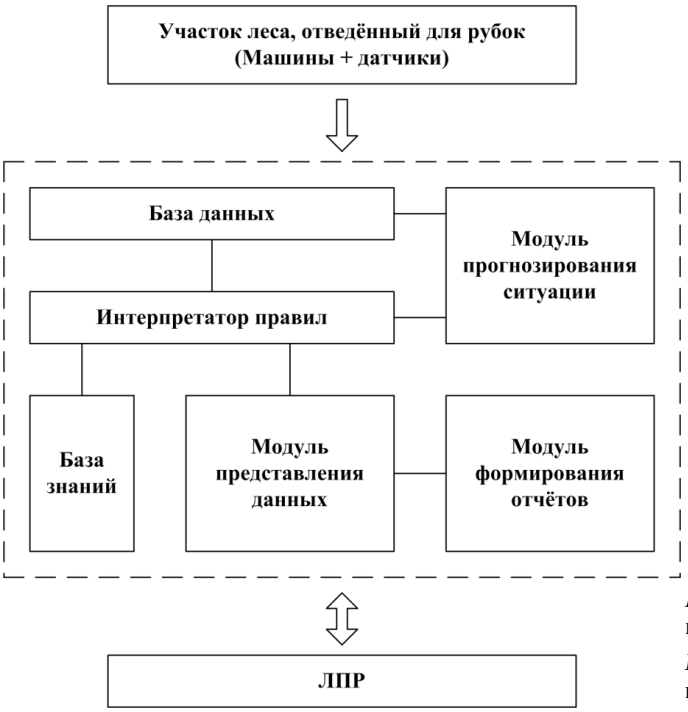

Рис. 3. Структура СППР оценки негативного влияния лесных машин на окружающую среду

Fig. 3. Structure of the DSS for forest machines' negative effect estimation on the natural environment

[3] https://fisgroup.ru/blog/fis_dss_opisanye_systemy/

Информация поступает с места рубок от лесной машины, оснащенной необходимыми датчиками (веса, уровня гидравлической жидкости и т. д.), и заносится в *базу данных* (БД).

Следующий блок – *база знаний* (БЗ). Она содержит определенные экспертные знания, с помощью которых производится анализ некоторой ситуации. Для БЗ удобно использовать нечеткий логический вывод [5; 6].

Что под ним подразумевается? *Нечеткий логический вывод* – это аппроксимация зависимости $y = f(x_1, x_2 ..., x_n)$ с помощью нечеткой базы знаний и операций над нечеткими множествами. Процесс нечеткого логического вывода – это процедура или алгоритм получения нечеткого вывода на основе нечетких условий или предпосылок [4].

В общем виде нечеткую продукционную модель можно представить как [7]

$$M = < L, R, Pf, FI, Pd >,$$

где L – множество лингвистических переменных (входных и выходных); R – база нечетких продукционных правил; Pf – процедура фаззификации; FI – блок нечеткого логического вывода; Pd – процедура дефаззификации.

Рассмотрим элементы модели.

Лингвистическая переменная – переменная, значениями которой являются не числа, а слова или фразы из некоторого языка. Например, лингвистическая переменная А = “скорость движения” и ее терм-множество К = <“высокая”, “средняя”, “низкая”>.

Продукционное правило – это правило вида ЕСЛИ <условие> ТО <действие>.

Например:

- Если GBC = 15 и NGP = 10, то нарушение = нет.

Добавим ещё переменные:

- Если GBC = 3 и влажность грунта = очень высокая, и NGP = 3, то нарушение = да.
- Если GBC = 7 и влажность грунта = низкая, и дорожное покрытие = укрепленное, и NGP = 9, то нарушение = нет.
- Если GBC = 10, влажность грунта = высокая, и дорожное покрытие = отсутствующее, и скорость движения машины = высокая, и NGP = 9, то нарушение = да.

Процедура фаззификации – на данном этапе устанавливается соответствие между численным значением входной переменной системы нечеткого вывода и значением функции принадлежности соответствующего ей терма лингвистической переменной [5].

Блок нечеткого логического вывода использует базу нечетких продукционных правил и включает три процедуры:

- Р1 – процедура агрегирования степени истинности предпосылок нечетких правил;
- Р2 – процедура активизации заключений нечетких правил;
- Р3 – процедура аккумулирования активизированных заключений нечетких правил.

Процедура дефаззификации – это процесс перехода от функции принадлежности выходной лингвистической переменной к ее числовому значению.

В целом разработка БЗ – довольно обширная задача, которая требует рассмотрения в рамках отдельной статьи.

Возвращаемся к структуре СППР. Далее правила из БЗ и данные из БД поступают на вход *интерпретатора правил* (ИП). Данный программный модуль имитирует логический вывод эксперта, используя имеющуюся информацию. Если ее недостаточно, то она может быть получена в результате диалога с ЛПР.

ИП непосредственно связан с *модулем прогнозирования ситуации*, который позволяет предсказывать вероятные инциденты. Прогнозирование может осуществляться автоматически либо по команде от пользователя на основе данных о текущем состоянии ЛЗМ. Далее информация поступает в *модуль представления данных*, где она отображается в виде, наибо-

[4] https://nbpublish.com/library_read_article.php?id=34798
[5] http://nrsu.bstu.ru/chap27.html

лее удобном для восприятия ЛПР, причем пользователь не должен заниматься поиском информации для анализа ситуации, она должна поступать к нему непрерывно и в автоматическом режиме.

Оператор должен иметь возможность сформировать, экспортировать отчет в удобном для прочтения формате как после наступления определенного события, так и за определенный временной период (час, день, неделя и т. д.).

ЛПР будет иметь возможность непосредственного контроля, но в то же время система должна поддерживать интерактивный режим функционирования. Это, в свою очередь, сократит нагрузку на пользователя и приведет к улучшению качества выполняемых работ.

Каким *принципам* должна отвечать предлагаемая СППР? Что нужно учитывать при ее проектировании и разработке? Необходимо отметить следующие позиции:

- возможность прогнозировать поведение различных показателей;
- способность автоматически отслеживать текущие и формирующиеся инциденты;
- возможность анализировать взаимосвязь событий и процессов;
- прогнозирование поведения одних показателей в зависимости от значений других;
- простота и удобство использования, а также информативность системы [8].

Оценка изменений вследствие внедрения предлагаемой СППР

Видится возможным производить оценку по следующим техническим показателям.

1. *Время реакции на происшествие с момента его выявления* ($T_в$). В случае отсутствия такой системы высока вероятность того, что инцидент будет скрыт или обнаружен через большой промежуток времени, когда рекультивация земель станет более трудоемкой или невыгодной с экономической точки зрения. При наличии такой системы мы наблюдаем иную ситуацию: время реакции зависит лишь от самого персонала, осуществляющего мониторинг лесных насаждений / контролирующего процесс лесозаготовок на определенной территории.

2. *Время анализа ситуации* ($T_а$) *и ликвидации загрязнения* ($T_л$). Данные параметры прямо зависят от сложности инцидента, а также отсутствия или наличия аналогичного случая в БЗ СППР. Кроме того, они зависят от удаленности места лесозаготовок, погодных условий, состояния лесных дорог, вида почв и т. п.

3. *Глубина колеи, образуемая лесной машиной.* Нетрудно догадаться, что серьезный механический урон почвенному покрову наносят именно движители лесных машин. Согласно О. Н Бурмистровой и др. [9], допустимая глубина колеи после первого прохода ЛЗМ не должна превышать 20 см. Данный факт подтверждается и другими исследователями в [10].

Имитационное моделирование

Рассмотрим в качестве иллюстрации следующие ситуации.

Каким образом можно сократить время очищения почвы от нефтепродуктов?

Для этого обратимся к модели, описанной Е. А. Ельчаниновой [6]. Согласно ей, суммарное время, необходимое для полного очищения почвенного покрова после инцидента, определяется как:

$$t_{полн} = t_{VR} + t_{пд} + t_{пч},$$

где t_{VR} – время впитывания нефтепродуктов из подстилки в почву; $t_{пд}$ – время, необходимое для очищения подстилки от нефтепродуктов за счет деградации и вымывания осадками в растворенном виде; $t_{пч}$ – время полного очищения почвы от нефтепродуктов за счет деградации и вымывания осадками в растворенном виде.

[6] https://portal.tpu.ru/SHARED/e/ELCHANINOVA/study/Tab3

Данные переменные можно вычислить по следующим формулам:

$$t_{VR} = \frac{(V_{A0} - V_{AГ})}{q_0},$$

где V_{A0} – начальный объем нефтепродуктов в подстилке после аварии, м^3; $V_{AГ}$ – остаточная нефтеемкость подстилки, м^3; q_0 – интенсивность впитывания нефтепродуктов, м/сут;

$$t_{пд} = -\frac{1}{\lambda} * \ln\left(\frac{J}{\lambda * V_{AГ} + J}\right),$$

$$t_{пч} = -\frac{1}{\lambda} * \ln\left(\frac{J}{\lambda * V(t_{пд}) + J}\right),$$

где λ – коэффициент деградации нефтепродуктов в подстилке и почве, сут$^{-1}$; J – интенсивность растворения нефтепродуктов проникающими в подстилку дождевыми осадками, м/сут; $V(t_{пд})$ – объем нефтепродуктов в почве на момент t_0, м^3.

Рассмотрим следующий пример.

Плотность дизельного топлива 900 кг/м^3; отношение вязкостей дизельного топлива и воды 4; толщина подстилки 0,05 м (рис. 4, *a*); пористость подстилки 0,3; влажность подстилки 0,07; пористость почвы 0,5; влажность почвы 0,15; коэффициент фильтрации почвы 1 м/сут; температура воздуха и топлива в момент происшествия 20°C; годовая норма осадков 450 мм (рис. 4, *b*); коэффициент деградации нефтепродуктов 0,01 сут$^{-1}$.

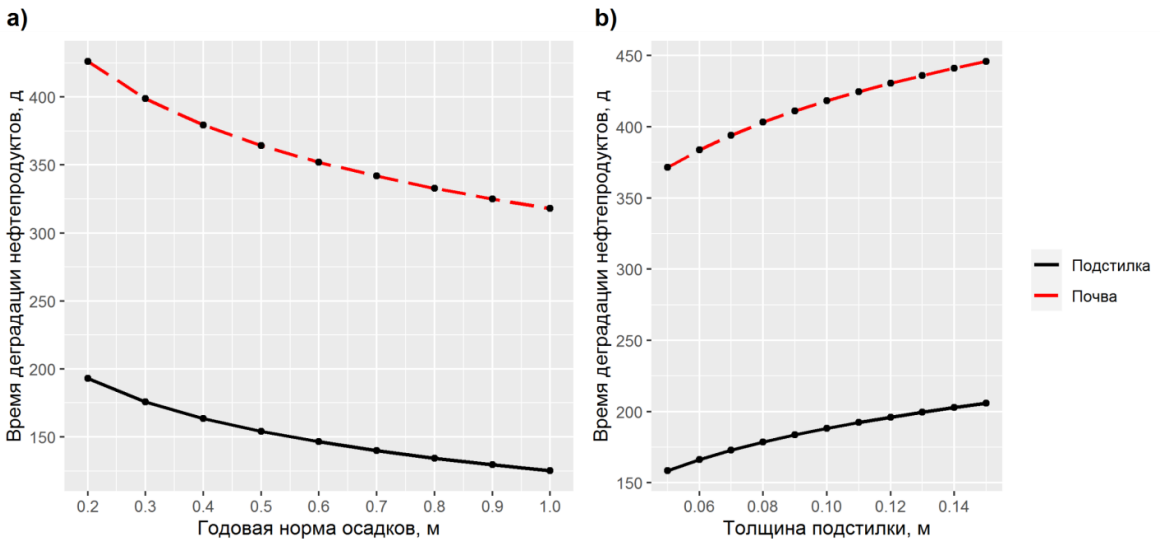

Рис. 4. Графики зависимости времени деградации загрязнения:
a – от годовой нормы осадков; *b* – от толщины лесной подстилки

Fig. 4. Dependence charts of the pollution degradation time:
a – from the mean annual rainfall; *b* – from the thickness of forest floor

Как видно из графика, для полного разложения нефтепродукта необходимо довольно значительное время (более 200 дней для подстилки толщиной в 0,15 м).

В случае применения СППР, своевременного реагирования и проведения незамедлительных мер по рекультивации загрязненной территории это время можно сократить в несколько

раз. Например, использование биопрепаратов позволяет снизить содержание нефтепродуктов в почве в ходе рекультивационных мероприятий до значения норматива *ДОСНП* за ~ *50 суток* [11] или полностью устранить загрязнение за приблизительно схожий срок [7].

Как предотвратить превышение необходимого уровня глубины колеи?

По формуле G. W. Turnage [12] глубина колеи после первого прохода машины *с одинаковыми шинами* определяется как

$$z = 4.61 * n^{0.5} * (CI / NGP)^{-2.6} * D,$$

где n – число проездов ЛЗМ; CI – конусный индекс (показатель прочностных характеристик почвогрунта), kPa; NGP – номинальное давление машины на грунт, kPa; D – диаметр колеса, м.

Чтобы определить величину колеи после нескольких проходов, воспользуемся формулой, представленной в работе A. Abebe, T. Tanaka, T. и M. Yamazaki [13]:

$$z_n = z_1 + n^{\frac{1}{a}},$$

где z_1 – глубина колеи после первого прохода, м; n – число проходов; a – коэффициент, зависящий от числа проходов и типа грунта (в случае с одинаковыми шинами принимается за 2).

Проиллюстрируем (рис. 5 и 6), используя следующие данные:

график 1 – NGP – 60 kPa, CI изменяем от 200 до 800 kPa,

график 2 – CI – 400 kPa, значение NGP изменяем в диапазоне от 30 до 120 kPa, D – 1,5 м.

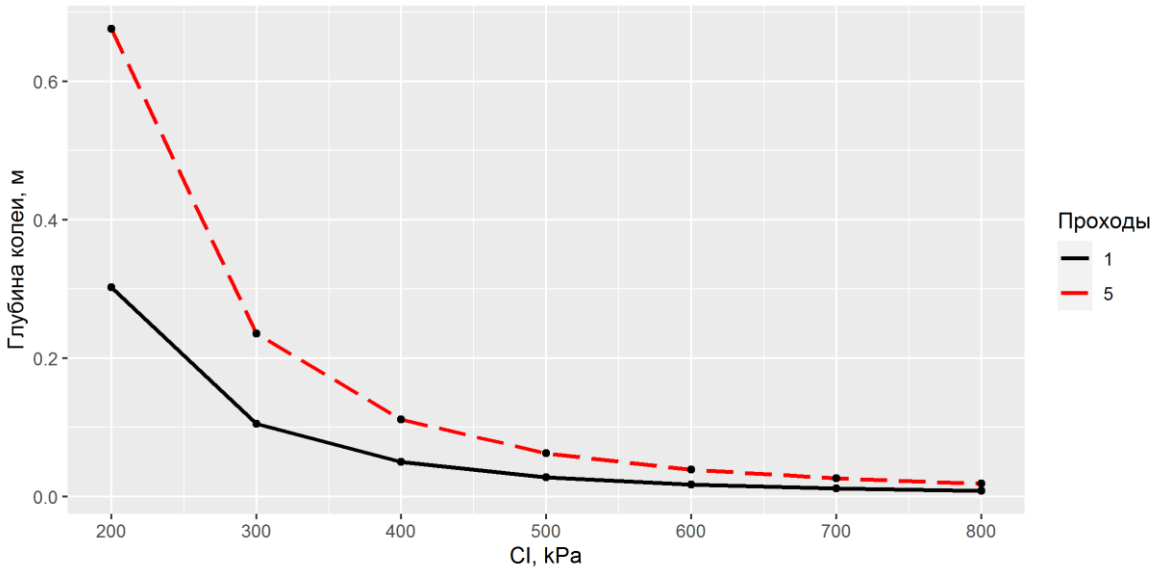

Рис. 5. График зависимости глубины колеи от значения конусного индекса
Fig. 5. Dependence chart of the track depth value from CI

[7] http://pkf-gran.ru/index.php?option=com_content&view=article&id=3&Itemid=122

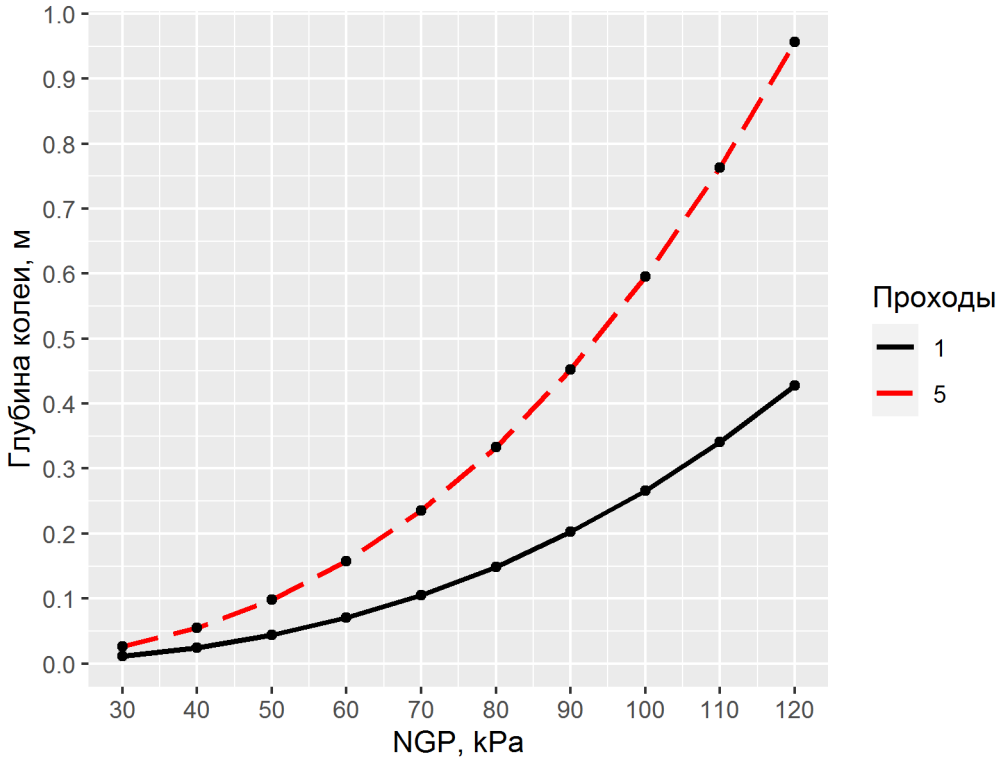

Рис. 6. График зависимости глубины колеи от значения NGP
Fig. 6. Dependence chart of the track depth value from NGP

Можно заметить, что в ряде случаев (например, график 1 – CI в диапазоне от 200 до 250 kPa, график 2 – NGP более 90 kPa) происходит превышение требуемой колейности при первом проходе машины. Определение значений NGP в реальном времени позволит избежать таких ситуаций.

Влияние СППР на показатели «Индустрии 4.0»

Напомним, ранее в статье было отмечено, что лесной отрасли требуется техническая и технологическая модернизация в рамках концепции «Индустрия 4.0». Далее рассмотрим, какое влияние оказывает предлагаемая СППР на ее показатели.

G. Schuh, R. Anderl и др. выделяют группу показателей «Индустрии 4.0». В рамках данной работы нас интересует только один из них, а именно «Общая эффективность систем». Он включает в себя следующие три подпоказателя [14]:

- коэффициент эффективности ($K1$);
- коэффициент качества ($K2$);
- коэффициент доступности ($K3$).

К1. Как определить текущее значение? В отчете о результатах совместного экспертно-аналитического мероприятия «Анализ эффективности использования лесных ресурсов Российской Федерации в 2016–2018 годах» отмечается, что необходимый объем осмотра лесосек не осуществляется ввиду нехватки штатного персонала (например, в 2018 г. в Иркутской области осуществлен осмотр 56 % лесосек) [15].

В связи с этим, а также исходя из утверждения, что данная СППР отсутствует в существующих лесхозах, примем данную величину за ~ 60 %.

Каким будет значение после внедрения СППР? В идеальном случае эта величина должна составлять 100 %, что не всегда достижимо на практике. На текущий момент, реальным видится средний показатель в ~ 75-80 %. Из чего строится такой вывод? Какие факторы влияют на это значение? Основных можно выделить три.

Во-первых, *недостаточное покрытие высокоскоростным интернетом мест рубок* или полное отсутствие какой-либо связи. Это лишает возможности получать информацию в режиме онлайн. И это основная проблема. Во-вторых, *злонамеренное действие оператора*, заметившего определенный инцидент. И, наконец, *обыкновенный недосмотр.*

К2. Коэффициент качества СППР. Его можно рассчитать по следующей формуле:

$$K2 = (1 - \sum_{i=1}^{n} Kc) * 100\%,$$

где *Кс* – коэффициент снижения качества.

Что вносит вклад в снижение качества получаемого результата? Сама аппаратура (датчики, АЦП и т. п.), используемые алгоритмы обработки аналогового сигнала, внешние наводки, а также нельзя исключать ошибки оператора.

Кроме того, необходимый коэффициент качества может определяться самим заказчиком в ТЗ.

Исходя из вышеописанного, К2 должен находиться в диапазоне 85–95 %.

К3. Коэффициент доступности [8] определяется как

$$K3 = \frac{AST - DT}{AST} * 100\%,$$

где *AST* – согласованное время предоставления услуги; *DT* – сумма простоев за период.

Современные лесные машины позволяют работать круглые сутки, исключая длительность регламентированных временных приостановок функционирования техники. Исходя из этого, рассчитаем *К3* (период измерений – неделя, предполагаем, что допускается час простоя системы каждый день): ((24*7) – 7) / 24*7 = 95,8 %.

Вывод

В заключение стоит отметить, что из года в год использование СППР в лесной отрасли только увеличивается. Как отмечают E. Ortiz-Urbina, J. González-Pachón, «…из 134 статей, посвященных принятию решений в данной отрасли, в 22-х (16 % от всего количества) авторы упоминали применение СППР для решения поставленных задач» [16].

Можно заметить, что использование СППР для оценки негативного влияния лесных машин на окружающую среду приведет к определенным улучшениям в сфере экологической безопасности.

В большей степени это касается предотвращения эрозии лесных почв, а также предупреждения снижения плодородия, вызванного нерациональным использованием и химическим загрязнением.

Например, среди рассмотренных ситуаций:

- сокращение времени разложения нефтепродуктов в лесной подстилке в ~ 3 раза, в почве в ~ 7 раз (см. рис. 4, *b*);
- предотвращение ~ 30 % возможных случаев превышения необходимого уровня колейности, первый проход (см. рис. 6).

Более того, СППР оказывает положительное влияние на показатели «Индустрии 4.0»: например, значение коэффициента эффективности вырастет ~ на 20 %.

[8] http://smartsourcing.ru/blogs/teoriya_metodiki_i_standarty/2210

Список литературы

1. **Васенёв М. Ю.** «Индустрия 4.0»: использование информационных технологий для снижения техногенного воздействия лесозаготовительных машин // International Journal of Open Information Technologies. 2019. Vol. 7, no. 10. С. 50–58.
2. **Nevalainen P., Salmivaara A., Ala-Ilomäki J. et al**. Estimating the Rut Depth by UAV Photogrammetry. *Remote Sensing*, 2017, vol. 9, no. 12, p. 1279. DOI 10.3390/rs9121279
3. **Gézero L., Antunes C.** Road Rutting Measurement Using Mobile LiDAR Systems Point Cloud. *ISPRS International Journal of Geo-Information*, 2019, vol. 8, no. 9, p. 404. DOI 10.3390/ijgi8090404.
4. **Геловани В. А., Башлыков А. А., Бритков В. Б., Вязилов Е. Д.** Интеллектуальные системы поддержки принятия решений в нештатных ситуациях с использованием информации о состоянии природной среды. М.: Эдиториал УРСС, 2001. 304 с.
5. **Еремеев А. П., Симонов Д. Н., Чибизова Н. В.** Реализация прототипа системы поддержки принятия решений реального времени на основе инструментального комплекса G2 // Программные продукты и системы. 1996. № 3. DOI 10.15827/0236-235X
6. **Kobersy I. S., Shkurkin D. V., Zatonskiy A. V. et al.** Moving objects control under uncertainty. *ARPN Journal of Engineering and Applied Sciences*, 2016, vol. 11, no. 5, pp. 2830–2834.
7. **Ганина Я. О., Лаптев В. В.** Нечеткая продукционная модель для оценки профессиональных качеств морских специалистов // Вестник Астрахан. го. техн. ун-та. Серия: Управление, вычисл. техн., информ. 2016. № 3. С. 101–108.
8. **Вязилов Е. Д.** Методы и средства работы с базами данных. Обнинск: ИАТЭ НИЯУ МИФИ, 2012. 393 с.
9. **Бурмистрова О. Н., Просужих А. А., Хитров Е. Г и др.** Теоретические исследования производительности форвардеров при ограничениях воздействия на почвогрунты // Изв. вузов. Лесной журнал. 2021. № 3 (381). С. 101–116. DOI 10.37482/0536-1036-2021-3-101-116
10. **Ilintsev A., Bogdanov A., Nakvasina E., Amosova I., Koptev S., Tretyakov S.** The natural recovery of disturbed soil, plant cover and trees after clear-cutting in the Boreal Forests, Russia. *iForest*, 2020, vol. 13, iss. 6, pp. 531–540. DOI 10.3832/ifor3371-013
11. **Шайдуллина И. А., Антонов Н. А., Колесникова Н. Е. и др.** Испытание новых биотехнологий рекультивации нефтезагрязненных земель в условиях ОАО «Татнефть» // Материалы научной сессии ученых Альметьевского государственного нефтяного института. 2013. № 1. С. 168–173.
12. **Wronski E. B., Humphreys N.** A method for evaluating the cumulative impact of ground-based logging systems on soils. *Journal of Forest Engineering*, 1994, vol. 5, pp. 9–20.
13. **Abebe A., Tanaka T., Yamazaki M.** Soil compaction by multiple passes of a rigid wheel relevant for optimization of traffic. *Journal of Terramechanics*, 1989, vol. 26, no. 2, pp. 139–148.
14. **Schuh G., Anderl R., Gausemeier J., ten Hompel M., Wahlster W.** Industrie 4.0 Maturity Index. Managing the Digital Transformation of Companies. Munich, Herbert Utz Verlag, 2017.
15. **Мень М. А, Батуркин А. Н., Морохоева И. П., Доробалюк С. А.** Отчет о результатах совместного экспертно-аналитического мероприятия «Анализ эффективности использования лесных ресурсов Российской Федерации в 2016–2018 годах». М., 2020.
16. **Ortiz-Urbina E, González-Pachón J, Diaz-Balteiro L.** Decision-Making in Forestry: A Review of the Hybridisation of Multiple Criteria and Group Decision-Making Methods. *Forests*, 2019, vol. 10, no. 5, p. 375. DOI 10.3390/f10050375

References

1. **Vasenev M. Yu.** "Industry 4.0": information technologies utilization for reducing the man-made impact of tree harvesting machines. *International Journal of Open Information Technologies*, 2019. vol. 7, no. 10, pp. 50–58. (in Russ.)
2. **Nevalainen P., Salmivaara A., Ala-Ilomäki J. et al**. Estimating the Rut Depth by UAV Photogrammetry. *Remote Sensing*, 2017, vol. 9, no. 12, p. 1279. DOI 10.3390/rs9121279
3. **Gézero L., Antunes C.** Road Rutting Measurement Using Mobile LiDAR Systems Point Cloud. *ISPRS International Journal of Geo-Information*, 2019, vol. 8, no. 9, p. 404. DOI 10.3390/ijgi8090404.
4. **Gelovani V. A., Bashlykov A. A., Britcov V. B., Vyazilov E. D.** Intellegent DSS in the emergency situations with using an information about natural environment condition. Moscow, Editorial URSS, 2001, 304 p. (in Russ.)
5. **Eremeev A. P., Simonov D. N., Chibizova N. V.** Realization of the real-time DSS prototype based on the instrumental complex G2. *Program products and systems*, 1996, no. 3. (in Russ.) DOI 10.15827/0236-235X
6. **Kobersy I. S., Shkurkin D. V., Zatonskiy A. V. et al.** Moving objects control under uncertainty. *ARPN Journal of Engineering and Applied Sciences*, 2016, vol. 11, no. 5, pp. 2830–2834.
7. **Ganina Ya. O., Laptev V. V.** Fuzzy productive model for evaluation of professional qualities of sea experts. *Vestnik Astrakhan State Technical Uni. Series: Management, Computer Sciences and Informatics*, 2016, no. 3, pp. 101–108. (in Russ.)
8. **Vjazilov E. D.** Methods and techniques for operating with databases. Obninsk, IATE MEPHI, 2012, 393 p. (in Russ.)
9. **Burmistrova O. N., Prosuzhih A. A., Khitrov E. G. et al.** Theoretical Studies of Forwarder Productivity with Limited Impact on Soils. *Lesnoy Zhurnal*, 2021, no. 3 (381), pp. 101–116. (in Russ.) DOI 10.37482/0536-1036-2021-3-101-116
10. **Ilintsev A., Bogdanov A., Nakvasina E., Amosova I., Koptev S., Tretyakov S.** The natural recovery of disturbed soil, plant cover and trees after clear-cutting in the Boreal Forests, Russia. *iForest*, 2020, vol. 13, iss. 6, pp. 531–540. DOI 10.3832/ifor3371-013
11. **Shaidullina I., Antonov N., Kolesnikova N. et al.** Experimental study of novel biotechnologies for restoration of oil-contaminated lands in OAO Tatneft. *Materials of the science session of the Almetyevsk State Oil Institute*, 2013, no. 1, pp. 168–173. (in Russ.)
12. **Wronski E. B., Humphreys N.** A method for evaluating the cumulative impact of ground-based logging systems on soils. *Journal of Forest Engineering*, 1994, vol. 5, pp. 9–20.
13. **Abebe A., Tanaka T., Yamazaki M.** Soil compaction by multiple passes of a rigid wheel relevant for optimization of traffic. *Journal of Terramechanics*, 1989, vol. 26, no. 2, pp. 139–148.
14. **Schuh G., Anderl R., Gausemeier J., ten Hompel M., Wahlster W.** Industrie 4.0 Maturity Index. Managing the Digital Transformation of Companies. Munich, Herbert Utz Verlag, 2017.
15. **Men M. A., Baturkin A. N, Morokhoeva I. P., Dorobaliuk S. A.** Report of results of the cooperative expert and analytical event "Analysis of the efficiency of use of forest resources in Russia, 2016–2018". Moscow, 2020. (in Russ.)
16. **Ortiz-Urbina E, González-Pachón J, Diaz-Balteiro L.** Decision-Making in Forestry: A Review of the Hybridisation of Multiple Criteria and Group Decision-Making Methods. *Forests*, 2019, vol. 10, no. 5, p. 375. DOI 10.3390/f10050375

Информация об авторе

Михаил Юрьевич Васенёв, аспирант

Information about the Author

Mikhail Yu. Vasenev, Postgraduate Student

Статья поступила в редакцию 11.12.2021;
одобрена после рецензирования 01.02.2022; *принята к публикации 01.02.2022*
The article was submitted 11.12.2021;
approved after reviewing 01.02.2022; *accepted for publication 01.02.2022*

Научная статья

УДК 004.434
DOI 10.25205/1818-7900-2022-20-1-18-27

Предметно-ориентированный язык для генерации заданий из исходных текстов программ

Игорь Андреевич Жуков [1]
Юрий Леонидович Костюк [2]

[1, 2] Национальный исследовательский Томский государственный университет
Томск, Россия

[1] IgZhukov963@yandex.ru, https://orcid.org/0000-0001-7995-3688
[2] kostyuk_y_l@sibmail.com, https://orcid.org/0000-0002-8914-8161

Аннотация

Обсуждаются подходы автоматизированной генерации заданий для различных дисциплин, отмечено, что для программирования параметрический способ не может быть применен. Авторы развивают идею применения конструктивно-выборочного метода для генерации заданий по программированию. В конструктивно-выборочном методе задание дополняется набором компонентов, из которых обучающийся составляет свой ответ (программу). Применение этого метода позволяет разнообразить задания по программированию, создает новый способ автоматизированного контроля на основе содержания ответа. При этом правильные ответы учитываются как полностью, так и частично. Проиллюстрированы на примерах основные понятия модели представления задания по программированию. Предложен предметно-ориентированный язык разметки и его транслятор, позволяющий генерировать задания из исходных текстов программ преподавателя. Приведен пример применения транслятора.

Ключевые слова

конструктивно-выборочный метод, контроль знаний по программированию, автоматизированный контроль знаний, подготовка заданий, предметно-ориентированный язык

Для цитирования

Жуков И. А., Костюк Ю. Л. Предметно-ориентированный язык для генерации заданий из исходных текстов программ // Вестник НГУ. Серия: Информационные технологии. 2022. Т. 20, № 1. С. 18–27. DOI 10.25205/1818-7900-2022-20-1-18-27

A Domain-Specific Language for generating tasks from Programs Source Code

Igor A. Zhukov [1], **Yuriy L. Kostyuk** [2]

[1, 2] National Research Tomsk State University
Tomsk, Russian Federation

[1] Ig.Zhukov963@yandex.ru, https://orcid.org/0000-0001-7995-3688
[2] kostyuk_y_l@sibmail.com, https://orcid.org/0000-0002-8914-8161

Abstract

Approaches to task generation for various disciplines are discussed, mentioned that a parameter method cannot be applied to programming. The authors developed an application of the idea of a constructive-selective method for creating programming tasks. A task created for constructive-selective methodical contains a set of components, from which

© Жуков И. А., Костюк Ю. Л., 2022

a student makes up his answer (a program). Application of this method allows diversifying of tasks in programming and creates a new way of assessment based on the content of the answer. It's also allows to grade both fully and partially correct answers. The main concepts of the model for representing programming tasks are illustrated by examples. The domain-specific markup language and its translator that allows generating tasks from a teacher's program source text are proposed. An example of a translator usage is given.

Keywords

constructive-selective method, programming knowledge assessment, automated knowledge assessment, task preparation, domain-specific language

For citation

Zhukov I. A., Kostyuk Yu. L. A Domain-Specific Language for generating tasks from Programs Source Code. *Vestnik NSU. Series: Information Technologies*, 2022, vol. 20, no. 1, p. 18–27. (in Russ.) DOI 10.25205/1818-7900-2022-20-1-18-27

Введение

Актуальной проблемой для системы образования является подготовка контрольно-измерительных материалов по различным дисциплинам. При этом особое место занимает процесс формирования заданий с применением технических средств. Анализ источников показал, что проблему рассматривают для различных технических и математических дисциплин. В работе [1] проанализированы типовые элементы деталей, используемых в заданиях по теме «Разрезы и сечения» в рамках дисциплины «Инженерная графика», и разработан алгоритм генерации новых заданий при помощи заранее созданных элементов. Каждое сгенерированное задание имеет решение. Сообщается о возможности получить 10^8 различных уникальных вариантов заданий. В [2] представлен генератор заданий по теме «Импеданс» для подготовки инженеров-электротехников. Обучающемуся предлагается электрическая схема со случайно заданными параметрами: сопротивления резисторов, индуктивность катушки и емкость конденсатора. Генератор проводит расчет, позволяющий отобрать корректные задания. Также предлагается программное обеспечение для представления обучающимся задач и проверки правильности их решений.

Способ генерации заданий, при котором формулируют типовые задачи с параметрами, значения которых подбираются с помощью генератора случайных чисел, в дальнейшем будем называть параметрическим. Этот способ распространен в математических дисциплинах. Следует отметить, что подбор числовых параметров может быть случайным выбором текстового элемента из заранее определенного списка. Авторы [3] используют математический пакет MATLAB для подготовки заданий по теме «Собственные числа и собственные векторы линейного оператора» в рамках дисциплины «Линейная алгебра». Средствами MATLAB выполняются генерация и отбор матриц, имеющих целые собственные числа. В работе [4] описана разработка системы для генерации заданий по теме «Линейное программирование». При генерации задания выбирается четыре случайных числа из заранее заданного диапазона. Эти значения подставляют в шаблон, затем происходит проверка наличия решения задачи с найденными параметрами. Также эта система реализует возможность проверки решений обучающихся. В [5] описаны правила алгоритма генерации для семи типовых задач по дисциплине «Численные методы». Используются как числовые параметры, так и список некоторых основных элементарных функций. Этот список позволяет, например, формировать трансцендентные уравнения для выдачи заданий обучающимся. В правила алгоритма генерации входит проверка корректности задачи. Также представлена программная реализация этого алгоритма. Автор [6] разработал формальный язык для описания генераторов заданий на основе деревьев. В узлах дерева содержатся данные (числа и строки текста), которые могут войти в условие формируемой задачи. Узел и его потомки могут быть связаны конструкциями «И» и «ИЛИ». По умолчанию используется конструкция «ИЛИ», при этом только один из потомков войдет в задачу. Если используется конструкция «И», то все потомки узла будут использованы. В качестве примера рассмотрена задача по теории вероятностей.

Практически во всех рассмотренных источниках для формирования заданий используется параметрический способ. Разнообразие заданий достигается за счет изменения числовых параметров при стандартизованном условии задачи. Обучающийся должен применить алгоритм решения типовой задачи к конкретному набору данных. Для дисциплин, связанных с программированием, параметрический способ в литературе не обсуждается. Это объяснимо, поскольку обучающийся, решая задачу по программированию, должен написать программу, используя стандартный или разработанный самостоятельно алгоритм. Параметрический способ вступает в противоречие со свойством массовости алгоритма. Согласно этому свойству алгоритм должен решать задачу для наборов данных из некоторой области, например, для всех целых чисел. Для генерации математической задачи можно взять квадратный трехчлен и подбирать его коэффициенты. Каждый набор коэффициентов определяет отдельную задачу – квадратное уравнение, которое можно дать обучающемуся. Для программирования можно в этом случае поставить только одну задачу – написать программу решения квадратного уравнения.

Авторы статьи для отработки навыков по программированию предлагают использовать конструктивно-выборочный метод. Задание реализовать конкретный алгоритм дополняется набором конструкций языка программирования, из которых обучающийся составляет программу. В этом случае при подготовке задания потребуется подобрать соответствующий набор конструкций. Преподаватель, как правило, уже имеет программы для всех предлагаемых заданий. В статье описаны предметно-ориентированный язык разметки и его транслятор, позволяющий перевести программный код в набор конструкций для выдачи задания. Кроме того, разработанный транслятор подготавливает задание для автоматизированного контроля.

Основные положения модели представления задания

Команды языка разметки опираются на понятия компонента, элемента и шаблона, введенные в [7]. Поясним эти понятия на примерах. Рассмотрим задание: вычислить сумму целых чисел от 1 до *n* включительно. Допустим, преподаватель работает с первым вариантом программы:

```
readln(n);
S:=0;
for i:=1 to n do
   S:=S+i;
writeln(S);
```

Компонент – это пара из натурального числа и строки программного кода, например, инициализация переменной, заголовок цикла. Компоненты для рассматриваемого примера будут следующими:

```
1. readln(n);          4. S:=S+i;
2. S:=0;               5. writeln(S);
3. for i:=1 to n do
```

Составить правильную программу из этих компонентов можно двумя способами: первую и вторую строки программы можно поменять местами. Первые два компонента образуют элемент типа «Перестановка компонентов», а остальные элементы будут иметь тип «Один компонент». При выдаче задания для обучающегося номера компонентов не указаны. Порядок компонентов можно изменить. Для данного примера получается 5!=120 возможных вариантов выдачи задания (118, если исключать варианты, в которых порядок компонентов соответствует правильным программам).

Преподаватель может предложить второй вариант для формирования задания – программу с использованием цикла while вместо for:

```
readln(n);
S:=0;
i:=1;
while(i<=n) do
begin
   S:=S+i;
   i:=i+1;
end;
writeln(S);
```

Обучающемуся будет предложено 9 компонентов (9!=362880 возможных вариантов выдачи задания). Составить правильную программу из этих компонентов можно шестью способами за счет перестановки первых трех строк. В этом примере первые три компонента образуют элемент типа «Перестановка компонентов», а остальные элементы будут иметь тип «Один компонент». Увеличение значения итератора цикла i может быть сделано альтернативным способом при помощи встроенной функции inc. Таким образом, получим третий вариант программы. Для этого следует добавить компонент со строкой кода «inc(i);», который в паре с компонентом «i:=i+1;» образует элемент с типом «Один из компонентов». Обучающемуся в таком случае будет предложено 10 компонентов (10!=3628800 возможных вариантов выдачи задания). Составить правильную программу из этих компонентов можно двенадцатью способами за счет перестановки первых трех строк и выбора между вариантами увеличения значения итератора. Для усложнения задания преподаватель может добавлять «шумовые» компоненты, которые не входят ни в одно верное решение.

При формировании задания можно одновременно использовать все описанные выше программы. В общем случае для одного языка программирования существует множество программ, которые решают поставленную задачу. В дальнейшем это множество будем называть U-шаблоном по аналогии с универсальным множеством U. Отметим, что программы, отличающиеся только идентификаторами (имена переменных и функций), не будем считать различными. Отличиями являются последовательность компонентов (если их порядок не влияет на правильность ответа и корректность программы), незначимые синонимы, значимые синонимы. В первом варианте программы предполагается перестановка компонентов, которая не влияет на правильность ответа (корректность программы). Третий вариант отличается от второго наличием незначимого синонима (способом увеличения значения итератора). Второй вариант отличается от первого наличием значимого синонима – использованием цикла while вместо for. Наличие значимых синонимов соответствует частям разбиения U-шаблона. В дальнейшем каждую часть разбиения будем называть шаблоном. Три программы, рассмотренные выше, не представляют U-шаблон полностью, поскольку возможны иные варианты написания цикла. На основе приведенных программ будет создано два шаблона. Для первого варианта программы преподавателя может быть создан один шаблон. А второй и третий варианты программы могут быть описаны другим шаблоном. Для записи шаблонов используется формальный язык, подробно описанный в [7].

Степень совпадения ответа обучающегося шаблону определяет контролирующий алгоритм. Частичное совпадение может считаться частично правильным ответом. Для преподавателя при подготовке шаблона предусмотрена возможность дать рекомендации по классификации вероятных ошибок как критических и некритических. Например, из предложенных компонентов составлена программа:

```
readln(n);
for i:=1 to n do
   S:=S+i;
writeln(S);
```

Эта программа похожа на первый вариант программы преподавателя, за исключением инициализации значения переменной S. Эту ошибку можно считать некритической. Оценка такого ответа будет незначительно снижена.

Рассмотрим другой пример ответа обучающегося:

```
readln(n);
S:=0;
S:=S+i;
writeln(S);
```

Эта программа также похожа на первый вариант программы преподавателя, но в ней отсутствует цикл for. Эта ошибка критическая, такая программа не решает поставленную задачу.

Для классификации критических и некритических ошибок в модели предусмотрены метки элементов. Метка может быть одного из двух типов: «допускается отсутствие» для некритических ошибок и «рубежный» для критических. При отсутствии меток значимость ошибок не различается.

Язык разметки

Для того чтобы воспользоваться транслятором, необходимо разметить файлы исходного кода программ, затем отправить размеченный файл на вход транслятора. Выходом будет набор компонентов и шаблон. Язык разработан так, чтобы управляющие команды начинались с символов строчного комментария в языке программирования. Таким образом, размеченная программа остается корректной с точки зрения языка программирования.

Определение шаблона имеет вид *//$ pattern (компонент | управляющая_команда) {(компонент | управляющая_команда)} //$ end*. Определение может начинаться в любом месте текстового файла. Текст вне определения игнорируется. Текстовый файл может содержать более одного шаблона. Приведем пример определения шаблона:

```
//$ pattern
readln(n);
S:=0;
for i:=1 to n
S:=S+1;
//$ end
```

Внутри определения шаблона могут быть строки и управляющие команды. Каждая строка текста является компонентом, который нумеруется автоматически. Управляющие команды делятся на 2 типа: однострочные и многострочные. Однострочная команда языка разметки находится перед строкой (строками) программы, на которые она действует. Многострочные команды имеют начало и конец, а все затронутые строки программы находятся в теле команды. Управляющие команды начинаются с символа *//!*. Элементы могут быть помечены: метка «допускается отсутствие» имеет значение *optional* (без учета регистра), метка «рубежный» имеет значение *important* (без учета регистра). Все строки внутри шаблона, не затронутые какой-либо командой, считают определением элемента типа «один компонент» без метки. Для определения элемента «один компонент» с меткой требуется однострочная команда, которая действует на следующую строку и имеет вид *//! метка*. Приведем пример:

```
//$ pattern
//! optional
S:=0;
for i:=1 to n
//$ end
```

Для задания элементов типа «один из компонентов» и «перестановка компонентов» можно использовать как однострочные, так и многострочные управляющие команды. Элементы типа «один из компонентов» определяется при помощи ключевого слова *oneof*. Однострочная команда имеет вид *//! oneof количество_строк l [метка]*. Количество строк указывает, на сколько последующих строк действует эта команда. Многострочная команда имеет вид *//! oneof [метка] один_или_больше_компонентов //! end.* Все компоненты, находящиеся внутри многострочной команды, относятся к определяемому элементу. Например:

```
//$ pattern
//! oneof 2l optional
i:=i + 1;
inc(i);
until i > n;
//$ end
```

```
//$ pattern
//! oneof
i:=i + 1;
inc(i);
//! end
end;
//$ end
```

Элементы типа «перестановка компонентов» определяются при помощи ключевого слова *permutation*. Однострочная команда имеет вид *//! permutation количество_строк l [метка_для_перестановки]*. Многострочная команда имеет вид *//! permutation [метка_для_перестановки] один_или_больше_компонентов //! end.* Приведем пример:

```
//$ pattern
//! permutation 2l
S:=0;
i:=1;
while i <= n do
//$ end
```

```
//$ pattern
//! permutation optional
writeln(f);
writeln(r);
//! end
//$ end
```

В соответствии со стандартом [1] запишем все порождающие правила в РБНФ:

команда = ("//" | "#") "!";

компонент = любой_символ {любой_символ};

один_или_больше_компонентов = компонент {компонент};

количество_строк = число "l";

метка = "optional" | "important";

метка_для_перестановки = "optional";

команда_завершения = команда "end";

начало_шаблона = ("//" | "#") "$" "pattern";

конец_шаблона = ("//" | "#") "$" "end";

обычная_метка = команда метка;

начало_один_из = команда "oneof" [метка];

многострочная_команда_один_из = команда количество_строк "oneof" [метка];

начало_перестановки = команда "permutation" [метка_для_перестановки];

многострочная_команда_перестановка = команда количество_строк "permutation" [метка_для_перестановки];

определение_один_из = начало_один_из один_или_больше_компонентов команда_завершения;

многострочное_определение_один_из = многострочная_команда_один_из один_или_больше_компонентов;

определение_перестановки = начало_перестановки один_или_больше_компонентов команда_завершения;

[1] ИСО/МЭК 14977:1996 Информационная технология. Синтаксический метаязык. Расширенная форма Бэкуса – Наура.

многострочное_определение_перестановки = многострочная_команда_перестановка один_или_больше_компонентов;

помеченый_элемент = обычная_метка компонент;

содержимое_шаблона = (компонент | управляющая_команда) {(компонент | управляющая_команда)};

управляющая_команда = определение_перестановки | определение_один_из | многострочное_определение_один_из | многострочное_определение_перестановки | помеченый_элемент;

определение_шаблона = начало_шаблона содержимое_шаблона конец_шаблона;

исходный_файл = (компонент | определение_шаблона) {(компонент | определение_шаблона)};

Грамматика оптимизирована для разбора алгоритмом синтаксического анализа LALR (Look-Ahead Left-to-Right), описанного в [8].

Пример использования

Возьмем пример из дисциплины «Основы программирования». Задания по этой дисциплине должны контролировать степень освоения определенного набора базовых алгоритмов и умение реализовать эти алгоритмы. Рассмотрим пример 2.3 из [9], который описывает рекуррентный алгоритм для приближенного вычисления значения функции $\sin x$. Приведем пример кода программы, которая решает эту задачу:

```
S := x;
f := x;
k := 2;
while abs(f) >= eps do
begin
   f := -f * x * x / ((2 * k – 1) * (2 * k – 2));
   S := S + f;
   k := k + 1;
end;
```

Для корректности программы неважен порядок инициализации переменных S, f, k, а также порядок изменения значений переменных S и k в теле цикла while. С формальной точки зрения это будут различные решения. Размеченный код программы может иметь следующий вид:

```
//$ pattern
   //! permutation
   S := x;
   f := x;
   k := 2;
   //! end
   //! important
   while abs(f) >= eps do
   begin
      //! important
      f := -f * x * x / ((2 * k – 1) * (2 * k –
2));
      //! 2l permutation
      S := S + f;
      k := k + 1;
   end;
//$ end
```

После трансляции из размеченной программы будут получены список компонентов (табл. 1) и шаблон, имеющий следующий вид:

{(1;2;3);4*;5;6*;(8;7);9}.

Все элементы получившегося шаблона приведены в табл. 2. Шаблону соответствует 3! * 2!=12 программ. Последовательности компонентов, описывающие эти программы, приведены в табл. 3.

Таблица 1

Список компонентов

Table 1

List of components

1. S := x;	4. while abs(f) >= eps do	7. S := S + f;
2. f := x;	5. begin	8. k := k + 1;
3. k := 2;	6. f := -f * x * x / ((2 * k – 1) * (2 * k – 2));	9. end;

Таблица 2

Элементы шаблона

Table 2

Elements of the pattern

Тип элемента	Метка	Компоненты, входящие в элемент
Перестановка компонентов	Без метки	1, 2, 3
Один компонент	Рубежный	4
Один компонент	Без метки	5
Один компонент	Рубежный	6
Перестановка компонентов	Без метки	7, 8
Один компонент	Без метки	9

Таблица 3

Полностью правильные последовательности компонентов

Table 3

Fully correct sequences of components

1;2;3;4;5;6;8;7;9	1;2;3;4;5;6;7;8;9	1;3;2;4;5;6;8;7;9
1;3;2;4;5;6;7;8;9	2;1;3;4;5;6;8;7;9	2;1;3;4;5;6;7;8;9
2;3;1;4;5;6;8;7;9	2;3;1;4;5;6;7;8;9	3;1;2;4;5;6;8;7;9

В размеченном файле компонент может встречаться многократно. Например, операторные скобки «begin» и «end». Когда в рамках трансляции одного размеченного файла компонент встретится впервые, ему будет присвоен номер. Если компонент встречается в файле многократно, то в шаблоне каждый раз будет использован изначально присвоенный номер.

Заключение

Применение конструктивно-выборочного метода дает возможность разнообразить задания по программированию. Перед тем как самостоятельно создавать программу от начала до конца, обучающиеся могут поработать с упрощенными заданиями. На основе конструктивно-выборочного метода предложен новый способ автоматизированного контроля по содержа-

нию текста программ, сконструированных обучающимися. При этом могут быть учтены частично правильные ответы. Разработанный авторами статьи инструмент позволяет без значительных трудозатрат перевести уже имеющиеся исходные тексты программ в задания, готовые для автоматизированного контроля знаний и выдачи обучающимся.

Список литературы

1. **Касаткина Е. П., Хесина Е. А., Чахеев Е. Я., Суховерхий В. А.** Генератор заданий по инженерной графике // Информатизация инженерного образования: Тр. Междунар. науч.-практ. конф. ИНФОРИНО-2016, Москва, 12–13 апреля 2016 года. М.: ИД МЭИ, 2016. С. 142–145.
2. **Jörg Vollrath**. An open access minimum automatic task generation live feedback system for electrical engineering. In: 2015 IEEE Global Engineering Education Conference (EDUCON 2015). (18–20 March 2015, Tallinn University of Technology, Tallinn, Estonia). IEEE, 2015, pp. 494–498. DOI 10.1109/EDUCON.2015.7096015
3. **Власова Е. А., Попов В. С., Пугачев О. В.** Создание фонда оценочных средств и новых образовательных технологий с использованием MATLAB при изучении линейной алгебры // Вестник Моск. гос. обл. ун-та. Серия: Физика-математика. 2016. № 4. С. 77–85.
4. **Хабибулина Н. Ю., Афанасьева Е. И.** Использование генератора индивидуальных заданий при разработке автоматизированной обучающей системы «Линейное программирование» // Электронные средства и системы управления. Материалы докладов Международной научно-практической конференции. 2011. № 1. С. 230–234.
5. **Обади А. А.** Разработка алгоритма и программного обеспечения для генератора задач тестовой системы // Вестник Технологического университета. 2019. Т. 22, № 1. С. 106–111.
6. **Зорин Ю. А.** Интерпретатор языка построения генераторов тестовых заданий на основе деревьев И/ИЛИ // Докл. Том. гос. ун-та систем управления и радиоэлектроники. 2011. № 1 (27). С. 75–79.
7. **Жуков И. А., Костюк Ю. Л.** Модель представления многовариантных заданий для автоматизированного контроля знаний по программированию // Вестник Том. гос. ун-та. Управление, вычислительная техника и информатика. 2020. № 53. С. 110–117. DOI 10.17223/19988605/53/11.
8. **Ахо А., Лам М., Сети Р.** Компиляторы. Принципы, технологии и инструментарий: Пер. с англ. 2-е изд. М.: И.Д. Вильямс, 2008. 1184 с.
9. **Костюк Ю. Л.** Лекции по основам программирования: Учеб. пособие. Томск: Изд-во ТГУ, 2019. 260 с.

References

1. **Kasatkina E. P., Khesina E. A., Chakheev E. Ya., Sukhoverkhy V. A.** Assignments generator on engineering graphics. In: International Conference on Information Technologies in Engineering Education (Inforino). Moscow, 2016, p 142–145. (in Russ.)
2. **Jörg Vollrath**. An open access minimum automatic task generation live feedback system for electrical engineering. In: 2015 IEEE Global Engineering Education Conference (EDUCON 2015). (18–20 March 2015, Tallinn University of Technology, Tallinn, Estonia). IEEE, 2015, pp. 494–498. DOI 10.1109/EDUCON.2015.7096015
3. **Vlasova E., Popov V., Pugachev O.** Creation of a fund of assessment tools and new educational technologies with the use of MATLAB in the study of linear algebra. *Bulletin of Moscow Region State University. Series: Physics and Mathematics*, 2016, no. 4. pp. 77–85. (in Russ.)
4. **Habibulina N. Yu., Afanaseva E. I.** Ispol'zovanie generatora individual'nyh zadanij pri razrabotke avtomatizirovannoj obuchajushhej sistemy “Linejnoe programmirovanie” [Usage of

the generator of individual tasks in the development of an automated lerning system “Linear programming”]. *Elektronnye sredstva i sistemy upravlenija. Materialy dokladov mezhdunarodnoy nauchno-prakticheskoy konferentsii*, 2011, no. 1, pp. 230–234. (in Russ.)
5. **Obadi A. A.** Development of an algorithm and software for a test system tasks generator. *Bulletin of the Technological University*, 2019, vol. 22, no. 1, pp. 106–111. (in Russ.)
6. **Zorin Yu. A.** The interpreter of programming language for design generators of tests based on AND/OR trees. *Proceedings of TUSUR University*, 2011, no. 1 (27), pp. 75–79. (in Russ.)
7. **Zhukov I. A., Kostyuk Yu. L.** Model of representation of multivariate tasks for automated control of programming knowledge. *Tomsk State University Journal of Control and Computer Science*, 2020, no. 53, pp. 110–117 (in Russ.) DOI 10.17223/19988605/53/11
8. **Aho A., Lam M., Sethi R., Ullman J.** Compilers: Principles, Techniques, and Tools. Pearson Education, Inc., 2006, 1010 p.
9. **Kostyuk Yu. L.** Lektsii po osnovam programmirovaniya [Lectures on the Fundamentals of Programming]. A Study Guide. Tomsk, TSU Press, 2019, 260 p. (in Russ.)

Информация об авторах

Игорь Андреевич Жуков, аспирант
Юрий Леонидович Костюк, доктор технических наук, профессор

Information about the Authors

Igor A. Zhukov, Post-Graduate Student
Yuriy L. Kostyuk, Doctor of Technical Sciences, Professor

Статья поступила в редакцию 01.12.2021;
одобрена после рецензирования 01.02.2022; *принята к публикации 01.02.2022*
The article was submitted 01.12.2021;
approved after reviewing 01.02.2022; *accepted for publication 01.02.2022*

Научная статья

УДК 004.896
DOI 10.25205/1818-7900-2022-20-1-28-46

Поиск оптимальных 2D моделей нейронной сети U-net для решения задачи семантической сегментации томографических изображений гидратосодержащих образцов

Татьяна Олеговна Колесник [1]
Антон Альбертович Дучков [2]

[1, 2] Новосибирский государственный университет
Новосибирск, Россия

[2] Институт нефтегазовой геологии и геофизики им. А. А. Трофимука
Сибирского отделения Российской академии наук
Новосибирск, Россия

[1] t.kolesnik@g.nsu.ru, https://orcid.org/0000-0002-1362-2584
[2] DuchkovAA@ipgg.sbras.ru, https://orcid.org/0000-0002-7876-6685

Аннотация

Задача семантической сегментации изображений гидратосодержащих пород представляет собой многоклассовую классификацию пикселей каждого 2D-изображения во входном 3D объеме по классам «Гранула», «Флюид», «Гидрат». Классы гранулы и гидрата-флюида являются слабоконтрастными, поэтому для их разделения используется модель, построенная на основе архитектуры сверточной нейронной сети U-Net. Такое решение позволяет провести классификацию пикселей по классу гранулы с точностью более 90 %, в то время как стандартная классификация по пороговому значению имеет точность лишь 56 %.

В данной статье описывается процесс поиска оптимальных моделей архитектуры U-Net посредством настройки набора гиперпараметров. Учитывая ограниченное время обработки большого объема 3D томографических данных, требовалось найти реализацию сети U-Net для достижения наилучшего качества сегментации. С другой стороны, качество сегментации может быть улучшено, посредством анализа взаимосвязей между последовательно идущими изображениями в 3D объеме. Однако 3D реализация сети является очень ресурсоемкой с точки зрения обучения модели и ее работы в режиме вывода. В связи с этим также был осуществлен поиск гиперпараметров с условием достижения сопоставимого качества сегментации при существенном упрощении реализации 2D сети. В дальнейшем найденные оптимальные гиперпараметры 2D модели могут быть использованы для настройки 3D модели сегментации.

Проведя анализ моделей по критериям сходимости и результирующего качества сегментации, мы предложили 2 модели сегментации, оптимальные с точки зрения качества и сложности модели. Рассмотренные в работе подходы гипернастройки модели U-Net имеют общий характер и могут быть применены при работе с другими наборами данных, что позволит улучшить производительность вашего нейросетевого решения как в процессе обучения, так и на стадии его эксплуатации.

Ключевые слова

томография, газогидраты, сверточные нейронные сети, сегментация, U-Net, поиск по сетке гиперпараметров

Для цитирования

Колесник Т. О., Дучков А. А. Поиск оптимальных 2D моделей нейронной сети U-net для решения задачи семантической сегментации томографических изображений гидратосодержащих образцов // Вестник НГУ. Серия: Информационные технологии. 2022. Т. 20, № 1. С. 28–46. DOI 10.25205/1818-7900-2022-20-1-28-46

© Колесник Т. О., Дучков А. А., 2022

Search for Optimal 2D Models of the U-net Neural Network for Solving the Problem of Semantic Segmentation of Tomographic Images of Hydrate-Containing Samples

T. O. Kolesnik [1], **A. A. Duchkov** [2]

[1, 2] Novosibirsk State University
Novosibirsk, Russian Federation

[2] Trofimuk Institute of Petroleum Geology and Geophysics
of the Siberian Branch of the Russian Academy of Sciences
Novosibirsk, Russian Federation

[1] t.kolesnik@g.nsu.ru, https://orcid.org/0000-0002-1362-2584
[2] DuchkovAA@ipgg.sbras.ru, https://orcid.org/0000-0002-7876-6685

Abstract

The task of semantic segmentation of 2D-tomographic scans of hydrate-containing rocks is a multi-class classification of pixels of each input image in a set according to the classes “Granule”, “Fluid”, “Hydrate”. Now this is implemented in the form of segmentation by the “Granule” class using the convolutional architecture of the U-Net neural network and classification of pixels unclassified as “Granule” into the “Fluid” and “Hydrate” classes by the threshold value of pixel intensity.

Considering the limited processing time of a large volume of tomographic data, it is necessary to find a compromise between the complexity of the model and the quality of segmentation. On the other hand, it is also required to propose a second, simpler implementation of the network, to extend it to a 3D segmentation model.

The solution of these optimization problems is achieved by tuning the hyperparameters of the U-Net model. To determine which set of network hyperparameters is the best in a particular case, a partial search was performed over the hyperparameter grid, limited by the variables responsible for:

1) the number of trained filters in convolution operations;
2) learning the biases vector for output channels from convolutional operations;
3) choosing an algorithm to increase the resolution in the network decoder part.

This article describes the process of finding optimal models and provides an assumption about the possibilities for their improvement.

Keywords

Tomography, gas hydrates, convolutional neural networks, segmentation, U-Net, hyperparameter grid search

For citation

Kolesnik T. O., Duchkov A. A. Search for Optimal 2D Models of the U-net Neural Network for Solving the Problem of Semantic Segmentation of Tomographic Images of Hydrate-Containing Samples. *Vestnik NSU. Series: Information Technologies*, 2022, vol. 20, no. 1, p. 28–46. (in Russ.) DOI 10.25205/1818-7900-2022-20-1-28-46

Введение

В настоящее время проявляется устойчивый интерес к природным газогидратам как к альтернативному источнику метана [1; 2]. Разрабатываются перспективные методы добычи метана путем разложения газогидратов в горных породах на газ и воду. Одним из методов изучения процессов в горных породах является динамическая рентгеновская томография – повторное сканирование изменяющегося образца и построение его трехмерных изображений в разные моменты времени. Томография может быть проведена с использованием рентгеновского или синхротронного излучения, при этом скорость сканирования и получаемое разрешение для лабораторных устройств рентгеновского излучения на порядок ниже [1] [3].

[1] *Батранин А. В.* Методы и средства контроля основных параметров и характеристик рентгеновских томографов высокого разрешения: Дис. … канд. техн. наук: 05.11.13 / Нац. исслед. Том. политехн. ун-т. Томск, 2016. 148 с.

Для изучения структуры и процессов образования и диссоциации гидрата в породе были проведены лабораторные эксперименты с повторяющимся томографическим сканированием на синхротроне с разрешением 3,45 мкм [4; 5]. На протяжении нескольких часов каждые 15 минут проводилось сканирование трех областей исследуемого образца (сдвинутых по вертикали): область сканирования представляет цилиндр высотой 1,8 мм и диаметром 4,2 мм, время сканирования одной области составляет 70 секунд, а получаемый набор восстановленных данных содержит 512 изображений с разрешением 1224 × 1224 пикселей[2]. Общий объем данных для одного эксперимента составляет несколько терабайт, что исключает их интерпретацию в ручном режиме. В связи с этим актуальными являются задачи организации хранения, а также разработки автоматических методов анализа большого объема томографических данных. Для решения проблемы могут применяться алгоритмы стриминговой томографии, когда обработка отсканированных данных происходит на лету и завершается до начала следующего сканирования. Среди них широкую популярность приобрели алгоритмы сегментации данных [2].

Одним из ключевых этапов анализа томографических данных является сегментация – отнесение каждого вокселя в 3D объеме к определенному типу материала, из которого состоит образец. Сегментация данных необходима для проведения количественной оценки различных параметров исследуемого образца. На основании непосредственно сегментации или оценки характеристик породы может быть проведена локализация целевых объектов и явлений, что позволяет отбирать отдельные области или временные интервалы из общего потока данных, а также облегчает дальнейшую интерпретацию полученных результатов. Кроме того, сегментация данных позволяет построить трехмерную структуру образца.

Сегментация 2D КТ-изображений гидратосодержащих пород представляет собой задачу многоклассовой классификации изображений в оттенках серого. Необходимо каждому пикселю входного изображения присвоить одну из следующих меток класса: «Гранула», «Гидрат» и «Флюид». В настоящее время эта задача реализована в виде сегментации по классу «Гранула» и классификации пикселей, не классифицированных как «Гранула», на классы «Жидкость» и «Гидрат». Второй этап реализован за счет установления порогового значения интенсивности пикселя. Однако при классификации гранул этот метод позволяет достичь качества сегментации только 56 %, так как классы гранулы и гидрата-флюида являются слабоконтрастными. Поэтому на данный момент для их классификации применяется нейросетевое решение, основанное на архитектуре сети U-Net, которое позволяет повысить точность сегментации до 95 % [3].

Целью данной работы являлось проектирование, настройка гиперпараметров и поиск двух моделей (#A, #B) сегментации на основе архитектуры сети U-Net:

- модель #A должна являться наилучшим аппроксиматором с точки зрения максимизации оценки метрики качества на проверочном наборе при ограничении на время сегментации;
- модель #B должна являться существенным упрощением архитектуры модели с допустимой потерей точности до 90 %.

Модель #A в дальнейшем будет использована для сегментации изображений в стриминговой томографии, а также для генерации обучающей выборки для 3D модели сегментации. Модель #B станет основой для 3D модели сегментации, обладающей гораздо большей вычислительной сложностью с точки зрения процесса обучения и ее применения. Для начала рассмотрим особенности настраиваемой архитектуры нейронной сети U-Net.

[2] *Фокин М. И.* Применение нейронных сетей для анализа синхротронных томографических изображений, полученных в процессе формирования и диссоциации газогидратов в образцах: Магистерская дис.: 05.04.01 / Новосиб. гос. ун-т. Новосибирск, 2020. 56 с.

[3] *Колесник Т. О.* Применение сверточных нейронных сетей для сегментации томографических изображений гидратосодержащих образцов // Материалы 59-й Междунар. науч. студ. конф. МНСК-2021. Сер. инфор. технологии. Новосибирск: НГУ, 2021. С. 90.

1. Метод

1.1. Архитектура нейронной сети

U-Net – это полностью сверточная нейронная сеть, которая была создана в 2015 г. для сегментации биомедицинских изображений [6]. Архитектура U-Net, представленная на рис. 1, относится к классу автоэнкодеров. Нисходящие этапы сети называют частью кодировщика (ender'a), восходящие же относятся к части декодировщика сети (decoder'a) и олицетворяют собой сжимающий и расширяющий путь сети соответственно. В данной статье рассмотрена имплементация базовой архитектуры, отличающейся от оригинальной сети 2015 г. частью декодировщика сети. Далее последует краткое описание ее реализации, с использованием общепринятой терминологии и концепций светрточных нейронных сетей [4].

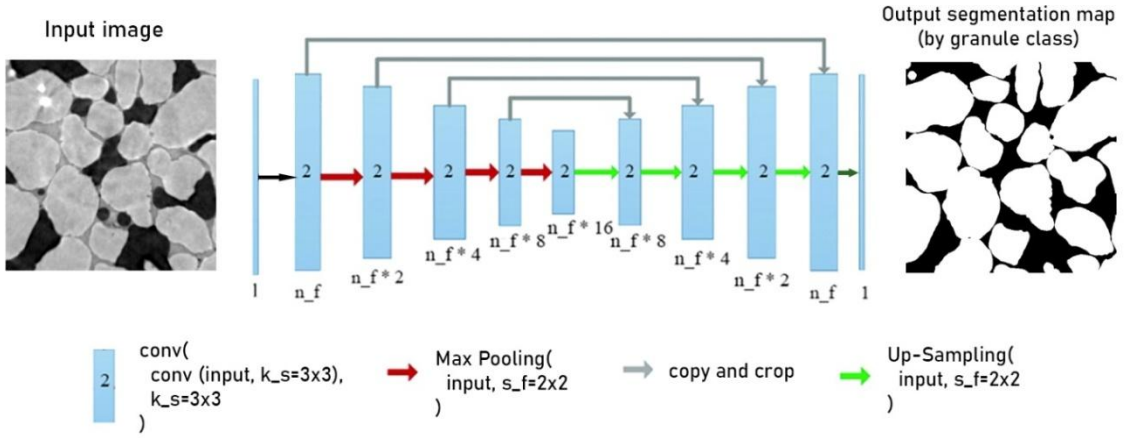

Рис. 1. Архитектура улучшаемой нейронной сети U-Net. Здесь n_f (num_filters) – базовое количество фильтров (в классической архитектуре U-Net 2015 $n\_f = 64$); k_s (kernel_size) – размер обучаемых сверточных фильтров в операциях свертки; s_f (scale_factor) – коэффициент понижения/повышения разрешения для операций Pooling / Upsampling

Fig. 1. Architecture of the improved neural network U-Net. Here n_f (num_filters) is the base number of filters (in the classical U-Net 2015 architecture $n\_f = 64$); k_s (kernel_size) is the size of trained convolutional kernels in convolution operations; s_f (scale_factor) is the coefficient of lowering/increasing the resolution for Pooling / Upsampling operations

Этапы сжимающего пути сети представляют собой применение двух последовательных слоев одношаговой (по вертикали и горизонтали) свертки с ядром 3 × 3 и с функцией активации ReLU, а также слоя максимального пулинга (MaxPooling) с ядом 2 × 2 с шагом 2. В процессе обучения фильтры свертки кодировщика настраиваются таким образом, чтобы выявить наилучшие признаки для классификации гранул. Так, при получении на вход одноканального изображения внутренней структуры исследуемого образца на выходе первого сверточного слоя кодировщика будет сгенерировано многоканальное признаковое представление изображения. Каждый канал в этом представлении можно интерпретировать как реакцию модели по соответствующему признаку / фильтру сверточного слоя на объект классификации во входном изображении.

С каждым сверточным слоем сжимающего пути обучаемые фильтры свертки увеличивают область кодировки исходного изображения – receptive field. Так, по окончании 1-го уровня в одном пикселе признакового представления будет закодирована информация о 6 × 6

[4] URL: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks

пикселях исходного изображения, а по окончании 4-го этапа – о 76 × 76 пикселях [5] [7]. Соответствующие сверточные фильтры 1-го уровня будут отражать частные признаки классификации, например информацию об интенсивности пикселей, а фильтры 4-го уровня – форму классифицируемых объектов (гранул). Соответственно, максимальная глубина сети U-Net задает максимальный уровень анализа признаков. Так, последняя свертка на 5-м уровне сети будет просматривать область размером 140 × 140 пикселей исходного изображения. При разрешении томографа в 3,45 мкм в такую область попадает от одной до двух гранул среднего размера.

Каждый этап кодировщика призван увеличить количество каналов – карт признаков в выходном признаковом представлении изображения в два раза, за счет удвоения количества применяемых фильтров свертки в сверточных слоях, а также уменьшить размер этих карт признаков в два раза, посредством операции максимального пулинга. Часть декодировщика сети является обратной к кодировщику и призвана привести это признаковое представление к исходному размеру изображения. Каждый этап расширяющего пути содержит UpSumpling слой, обратный пулингу, увеличивающий размер входных карт признаков в два раза, и слои свертки, которые вдвое уменьшают количество наблюдаемых признаков. С целью повышения точности модели также применяется операция “copy and crop”, дополняющая признаковое представление в декодировщике признаковым представлением в кодировщике.

Для сопоставления результирующего многоканального признакового представления с целевой маской сегментации в последнем этапе декодера используется дополнительный слой свертки с ядром 1 × 1 для объединения всех сформированных карт-признаков (каналов). В результате работы сети для одного входного изображения будет сгенерирована одна, сопоставимая по размерам с оригинальным снимком, 2D-карта сегментации, отражающая реакцию модели на целевой класс гранулы для каждого пикселя входного изображения – некоторую количественную оценку уверенности сети в том, что классифицируемый пиксель относится к классу гранулы. Для получения вероятностной оценки выход сети был пропущен через сигмоидную функцию активации [6].

1.2. **Задание пространства поиска оптимальных моделей**

Можно выделить следующие характеристики модели сегментации на основе нейронной сети:

- обобщающая способность модели – определяется количеством параметров модели. Увеличение количества обучающих параметров увеличивает время сегментации и физический размер модели;
- обучающая способность модели – определяется скоростью сходимости. Если модель обладает лучшей сходимостью, ей потребуется меньше эпох на обучение;
- достигнутое качество сегментации на тестовой выборке.

Реализуемые целевые модели (#A, #B) для сегментации 2D томографических снимков гидратосодержащих образцов по классу гранулы представляют собой решение двух оптимизационных задач:

- задача #A – оптимизация с точки зрения достижения наилучшего качества сегментации (предположительно, больше 95 %) при ограничении на время сегментации (полная обработка 3D объемов (1536 изображений разрешением 1224 × 1224 пикселей^2) не должна превышать 15 минут).
- задача #B – оптимизация с точки зрения уменьшения количества обучаемых параметров модели и уменьшения сходимости сети (количества обучаемых эпох) при достижении сопоставимого или лучшего качества сегментации. Если для упрощенной модели #B удастся

[5] URL: https://www.baeldung.com/cs/cnn-receptive-field-size
[6] URL: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

достичь лучшего качества сегментации, она также будет являться решением оптимизационной задачи #A.

В данной работе в качестве минимизируемой функции потерь была использована функция Binary Cross Entropy (BCE) [8], и 1 – BCE в качестве метрики качества. Вероятностная характеристика BCE отражает «уверенность» модели в классификации каждого отдельного пикселя по классу гранулы, и с точки зрения метрики качества такая оценка является более строгой по сравнению с не менее распространенной метрикой Mean Intersection over Union [7] [9]. Mean IoU принимает на вход бинаризированное представление вероятностного выхода нейронной сети, при этом порог бинаризации может быть задан как 50 % или же подобран на стадии обучения с целью улучшения предсказания.

Процесс обучения модели и достижимое качество во многом зависят от инициализации обучаемых параметров модели. Обычно процесс обучения не является воспроизводимым, так как в качестве инициализаторов выбираются разные значения. Однако, зафиксировав индекс random_seed, с которого начнется считывание случайной последовательности чисел для инициализации ядер, мы можем сделать процесс обучения воспроизводимым. В задаче поиска оптимальной модели мы должны проанализировать влияние каждого гиперпараметра на характеристики сходимости модели и на достижимое качество сегментации. Для этого мы фиксируем значение анализируемого гиперпараметра и обучаем модели на инициализаторах из ограниченного множества. В качестве метода инициализации сверточных ядер в данной работе используется метод HeNormal, представляющий собой выборку из сокращенного нормального распределения значений [8]. Средняя оценка для гиперпараметра по всем инициализаторам будет являться основанием для его выбора или исключения из пространства поиска.

Определим базовое пространство поиска Z оптимальных моделей:

- |seed| = 3 – способ инициализации обучаемых параметров модели;
- n_f = {8, 16, 24, 32, 40, 48, 56, 64} – базовое количество обучаемых фильтров свертки в модели U-Net;
- conv_bias_trainable = {True, False} – обучение смещения для сверточных ядер.
- upsample_method = {Nearest Neighbor, Bilinear Interpolation, Subpixel Convolution} – способ повышения разрешения, не требующий обучения.

Вычислительная сложность декодировщика сети может быть уменьшена посредством обучения дополнительного сверточного слоя, следующего после операции Upsampling'a, с сокращением количества выходных фильтров. В данной работе были рассмотрены случаи с сохранением количества фильтров уровня (без обучения дополнительной свертки) и с их уменьшением в 2 и 4 раза для реализаций слоев с Nearest Neighbor или Bilinear Interpolation.

Заданное пространство Z определяет поиск моделей по критериям времени обучения и вывода, а также достижимого качества сегментации среди 336 моделей. Проанализировать все возможные комбинации этих гиперпараметров не представляется возможным. Однако мы можем изначально проанализировать модели по двум признакам: например, по инициализаторам и количеству обучаемых фильтров свертки, сократить пространство поиска и расширять эти модели другими гиперпараметрами, такими как подход к регуляризации данных и способ декодировки.

В сверточных нейронных сетях самой тяжелой операцией является операция свертки. Чем больше сеть будет иметь обучаемых ядер свертки, тем большее количество операций умножения и сложения потребуется выполнить для получения нового признакового представления. Поэтому сокращение пространства поиска следует начать именно с этого гиперпараметра.

[7] URL: https://keras.io/api/metrics/segmentation_metrics/
[8] URL: https://www.tensorflow.org/api_docs/python/tf/keras/initializers/HeNormal

1.3. **Особенности процесса тестирования моделей**

Исследуемый образец проходит сканирование каждые 15 минут. Естественным ограничением стриминговой томографии является требование, чтобы отсканированные данные были полностью обработаны до начала следующего сканирования. Под обработкой в данном случае понимается восстановление трех 3D объемов, их полная сегментация в несколько этапов (1. Сегментация по классу гранулы; 2. Вычитание пикселей, классифицированных как гранула; 3. Классификация оставшихся пикселей изображения) с последующим анализом результатов. На практике также могут применяться алгоритмы пред- / постобработки данных, что также увеличит время, необходимое для обработки входного потока изображений. В связи с этим в данной работе приводится только оценка времени сегментации по классу гранулы различными реализациями сети U-Net.

Для нахождения оптимальных моделей по сетке гиперпараметров были использованы ресурсы вычислительного кластера научно-образовательного центра НГУ-Газпромнефть. В целях воспроизводимости процесса обучения моделей весь процесс обучения был реализован на процессоре Intel® Xeon® Gold 6254 с кэшем 24,75 MB и 386 Гб ОЗУ. Помимо фиксации всех инициализаторов обучаемых параметров моделей, обработка данных также являлась унифицированной и производилась без средств асинхронной побатчевой оптимизации в процессе обучения моделей. Фиксация способа инициализации обучаемых параметров делает процесс обучения воспроизводимым, а анализ моделей на нескольких инициализаторах позволяет оценить вклад каждого варьируемого гиперпараметра в настройку модели. Процесс тестирования времени сегментации по классу гранулы был реализован как на CPU, так и на графическом ускорителе Tesla V100 с 16 GB видеопамяти, подключенному через PCIe v3. Важным критерием параллельной обработки является оптимальная загрузка процессоров и организация кэширования данных. В данной работе было проведено тестирование времени исполнения с варьированием количества изображений в батче, равное $B = 2^i$, где $i = 2, .., 9$ для CPU и $B = 2^j$, где $j = 1, .., 5$ для GPU. В качестве меры центральной тенденции была выбрана медиана.

2. **Описание исследования**

2.1. **Варьирование значения количества обучаемых фильтров свертки**

В архитектуре нейронной сети U-Net, приведенной на рис. 1, на первом уровне сети изображение кодируется 64 признаковыми представлениями. С каждым последующим уровнем кодировщика количество признаков увеличивается вдвое. Обозначим количество обучаемых фильтров свертки на первом этапе сети за *n_f*. Так, на k этапе кодировщика количество фильтров вычисляется по формуле $n\_f_k = 2^{k-1} * n\_f$. В U-net 2015 максимальная глубина сети $K = 5$, при этом порядок уровней в декодировщике является обратным к уровням в кодировщике, и количество блоков декодировщика равно $K - 1$.

Зададим подпространство z_1 пространства поиска Z по гиперпараметру *n_f* и зафиксируем 3 значения *seed* для инициализации сверточных весов случайными значениями в методе HeNormal. Таким образом, подпространство z_1 задано декартовым произведением множеств *n_f* и *seed*, где *n_f* = {8, 16, 24, 32, 40, 48, 56, 64}. Обучение всех 24 моделей, соответствующих подпространству z_1, происходило на протяжении 15 эпох с использованием алгоритма с адаптивной скоростью обучения Adam [10].

Прежде всего было необходимо проанализировать сложность полученных моделей и оценить время на сегментацию 3D объема, состоящего из 512 изображений с разрешением 1224×1224 px^2. При размере снимка, не кратного $2\text{^}K - 1$ (K – глубина сети), может произойти потеря данных в части кодировщика при операции максимального объединения, что приведет к несоответствию размеров представлений в кодировщике и декодировщике сети. Эту проблему можно решить, предварительно уменьшив размер входных снимков до 1024^2,

построить карту сегментации, затем увеличить ее размер до размера оригинальных снимков. В данной работе для понижения / повышения разрешения используется линейная интерполяция. Данная пред- / постобработка снимков и масок также включена в общее время, необходимое для сегментации по классу гранулы, а качество тестовых изображений было оценено по разрешению 1024^2, а не по оригинальному размеру снимков.

В табл. 1 (во втором столбце) приведено количество обучаемых параметров модели, а в табл. 2 отображено затраченное время на сегментацию одного объема в минутах без учета выгрузки отсегментированных масок сегментации в выходную директорию. Так, самая сложная модель с $n\_f = 64$ фильтрами имеет в 64 раза больше обучаемых параметров, чем модель с $n\_f = 8$, но при этом ей потребуется в среднем в 8 раз больше времени для сегментации одного 3D объема на CPU.

Таблица 1

Характеристики сложности моделей по количеству обучаемых фильтров свертки и достигнутое значение метрики качества Binary Cross Entropy (BCE) на проверочном наборе

Table 1

Model complexity characteristics by the number of trained convolution filters and the achieved value of the Binary Cross Entropy (BCE) quality metric on validation set

Количество базовых фильтров (n_f)	Количество параметров модели	Размер модели (МБ)	Усредненное по инициализаторам значение метрик качества 1-BCE для моделей на проверочном наборе (%)
8	490 256	5,78	86,25
16	1 960 864	22,61	90,03
24	4 411 824	50,66	92,15
32	7 843 136	89,93	91,77
40	12 254 800	140,41	91,56
48	17 646 816	202,12	92,23
56	24 019 184	275,05	92,33
64	31 371 904	359,19	91,86

Как было отмечено ранее, процесс тестирования во многом зависит от характеристик оборудования, размера данных и количества изображений в батче. Так, для такого входного размера снимков максимальный размер батча на GPU составил всего 4 снимка в формате float32 для всех моделей U-Net, сложность которых превышает 24 миллиона параметров. Это связано с тем, что для графического процессора требуется обеспечить одновременное хранение модели, данных и их промежуточное представления в глобальной памяти видеокарты. При организации работы на CPU также следует учитывать характеристики используемого оборудования. Так, например, самая тяжелая модель с базовым числом фильтров $n\_f = 64$ в своем последнем слое кодировщика будет работать с тензорным представлением снимка $B \times H / 2^4 \times W / 2^4 \times (n\_f = 64 * 2^4)$, где B – количество изображений в батче, H / W – высота / ширина входного изображения. При размере снимка $H = W = 1024$ и $B = 1$ для хранения промежуточного тензора потребуется

$$\frac{\frac{1024^2}{2^8} * 64 * 2^4 * 32}{1024^2 * 8} \text{ МБ} = 16 \text{ МБ [11].}$$

Максимальный тестируемый размер батча на CPU – 512. Тогда промежуточный размер тензора на $K = 5$ уровне сети U-Net занимает 8Гб, что приводит к увеличению количества обращений к глобальной памяти.

Таблица 2

Тестирование времени генерации масок сегментации для одного 3D объема на CPU и GPU

Table 2

Testing the generation time of segmentation masks for one 3D volume on CPU and GPU

Количество базовых фильтров (n_f)	Медианное время сегментации	
	CPU (мин)	GPU (мин)
8	0,88	0,35
16	1,41	0,4
24	2,04	0,5
32	2,74	0,58
40	3,75	0,79
48	4,68	0,93
56	5,9	1,12
64	7,02	1,3

Если потребуется сохранить результат работы модели на батче данных в директорию, время обработки на CPU займет от 0,1 до 0,16 минуты в зависимости от размера батча и сложности модели. При обработке батча моделью на GPU для сохранения результата требуется скопировать сгенерированные маски сегментации на CPU. В среднем время сохранения результата при обработке на GPU занимает порядка 0,16–0,2 минуты.

Размер батча на CPU варьировался с 4 до 512, при этом оптимальная конфигурация была достигнута при размере батча в 16 изображений для моделей, базовое число n_f фильтров в которых больше 16. Для моделей с числом фильтров $n\_f = 8$ и 16, батчи с 32–128 изображениями обеспечивают оптимальную загрузку процессора. На GPU максимальный размер батча составил 32 изображения для модели с $n\_f = 8$, для остальных моделей размер батча был уменьшен, чтобы избежать ошибок с переполнением памяти. Для всех моделей размер батча в 2 снимка обеспечил оптимальную загрузку видеокарты.

Таким образом, вычисление моделей на видеокарте позволяет добиться существенного прироста производительности для всех моделей. Так, для модели с базовым числом фильтров $n\_f = 8$ ускорение составляет порядка 200 %, а для модели с $n\_f = 64$ – 530 %. Однако для моделей с количеством фильтров $n\_f = 56$, 64 может возникнуть проблема их использования в стриминговой томографии, где доступ к GPU на узле может быть ограничен. Кроме того, так как весь процесс обучения выполняется на CPU небольшими батчами изображений, и требуется также вычислять значения алгоритма обратного распространения ошибки, обучение и настройка гиперпараметров этих моделей затруднено временными ограничениями доступа к вычислительному кластеру. При этом модель с $n\_f = 48$ фильтрами обладает меньшей вычислительной сложностью и на данном этапе анализа пространства гиперпараметров позволяет добиться лучшего или сопоставимого качества сегментации. В связи с этим далее

мы будем рассматривать модели со сложностью, не превышающей 20 миллионов параметров. Однако при необходимости после подбора оптимальных гиперпараметров для модели с $n\_f = 48$ эти же подходы могут быть применены к моделям с большим числом фильтров.

2.2. **Регуляризация на уровне обучения сверточных слоев**

При обучении сверточных слоев также можно задать обучение дополнительного вектора, отвечающего за смещение каждого выходного канала из слоя свертки [9]. Если предсказания модели смещены достаточно сильно относительно целевых ответов, может потребоваться увеличить количество эпох обучения, так как в течение нескольких из них настройка весов будет направлена не на выявление признаков классификации, а на уменьшение смещения. При анализе количества сверточных фильтров сети мы использовали всего лишь 15 эпох для обучения. Возможно, более простые модели могли недообучиться и не выявить важных закономерностей и свойств признаков. Поэтому обучение смещений позволит за несколько эпох привести эти модели к значениям, близким к оптимальным. Модели, обладающие большим числом фильтров, вследствие увеличенной обобщающей способности достаточно быстро минимизируют значение смещения, поэтому ожидается, что сходимость смещенной сети будет являться сопоставимой.

Альтернативным, более распространенным способом регуляризации выхода является подход побатчевой нормализации данных [12], который помимо смещения включает в себя настройку поканального стандартного отклонения. Нормализация на уровне сверточных слоев или их выходов в нейронной сети позволяет ввести регуляризацию обучаемых параметров, что приводит к улучшению сходимости сети и, соответственно, достижению лучшего качества сегментации за то же количество эпох.

В данной работе рассмотрен подход к регуляризации на уровне сверточных слоев. Операция смещения практически не потребляет ни память, ни вычислительное время, однако при этом благоприятно сказывается на процессе обучения. Требовалось провести анализ достигнутого качества сегментации на множестве z_1, расширенном гиперпараметром обучаемого сдвига в сверточном слое: $z_2 \subset Z$; $z_2 = z_1 \times$ conv_bias_trainable, где conv_bias_trainable – булево значение, отвечающее за обучение вектора смещения; conv_bias_trainable = *False* означает, что нормировка данных не применяется, т. е. $z_1 \times$ (conv_bias_trainable' = {*False*}) = z_1. Таким образом, требовалось протестировать $|z_2| = 48$ моделей (3 инициализатора * 8 значений фильтров n_f * на обучение векторов смещений), 24 из которых уже были обучены. Для сопоставления процесса обучения новые модели также были обучены на протяжении 15 эпох.

Сравним достижимое качество моделей с обучаемыми векторами смещений для каждой операции свертки сети (табл. 3).

Уже на этом этапе анализа времени сегментации мы можем разделить модели на две группы:

a) сложные (n_f = {32, 40, 48, 56, 64}) – решающие оптимизационную задачу уровня A;

b) простые (n_f = {8, 16, 24}) – решающие оптимизационную задачу уровня B;

Подход регуляризации на уровне сверточных слоев позволил существенно улучшить качество всех моделей, поэтому в дальнейшем, учитывая возросшее время на обучение сложных моделей, мы будем рассматривать для расширения гиперпараметрами сокращенное множество z_2':

$$z_2' \subset z_2;$$

$$z_2' = \left(n\_f' = \{8, 16, 24, 32, 40, 48\}\right) \times \left(\text{conv\_bias\_trainable} = \{True\}\right).$$

[9] URL: https://deepai.org/machine-learning-glossary-and-terms/bias-vector

Таблица 3

Характеристики сложности моделей по количеству обучаемых фильтров свертки и достигнутое значение метрики качества 1-BCE на проверочном наборе при настройке моделей по параметру регуляризации свертки

Table 3

Characteristics of model complexity by the number of trained convolution filters and the achieved value of the 1-BCE quality metric on validation set when tuning models by the convolution regularization parameter

Количество базовых фильтров (n_f)	Количество обучаемых весов сверток	Дополнительные обучаемые параметры смещения	Усредненное по инициализаторам значение метрик качества 1-BCE для моделей (%)	
			Conv	Conv + $\overline{bias}$
8	490 256	+ 737	86,25	89,17
16	1 960 864	+ 1 473	90,03	91,83
24	4 411 824	+ 2 209	92,15	93,15
32	7 843 136	+ 2 945	91,77	93,44
40	12 254 800	+ 3 681	91,56	93,35
48	17 646 816	+ 4 417	92,23	93,72
56	24 019 184	+ 5 153	92,33	92,78
64	31 371 904	+ 5 889	91,86	93,38

3. **Настройка способа декодировки сети**

Все ранее рассмотренные модели U-Net с глубиной $K = 5$ уровней уменьшали разрешение входного изображения в 2^4 раза и использовали интерполяцию Nearest Neighbor в части декодировщика сети для повышения разрешения закодированного признакового представления [10]. Так, с каждым уровнем сети, признаковое представление изображения будет дважды увеличено в размере посредством декодировки 1 пикселя во входном представлении 4 пикселями (2×2) с тем же значением интенсивности. Альтернативным способом является применение билинейной интерполяции, которая вычисляет промежуточные значения интенсивностей пикселей в генерируемом представлении на основании анализа четырех интенсивностей ближайших пикселей во входном представлении. Операция ближайшего соседа и интерполяция не изменяет количество каналов входного признакового представления.

Еще одним способом реализации слоя повышения дискретизации, не требующего обучения весовых параметров, является алгоритм субпиксельной свертки, основанный на операции «глубина в пространство» (depth to space) [11]. Субпиксельная конволюция использует понижение количества каналов входного тензора размером $H \times W \times C * r^2$ для генерации тензора большего размера $H * r \times W * r \times C$ [13]. Здесь H – высота представлений, W – ширина представлений, C – количество представлений, r – коэфициент повышения разрешения. Уменьшение количества просматриваемых признаковых представлений в операциях субпиксельной конволюции в 2^2 раза позволяет уменьшить вычислительную сложность следующей операции свертки и модели в целом, но также может сказаться на результирующем качестве сегментации в лучшую или худшую сторону. В оригинальной статье 2015 г. предлагается использовать слой up-convolution, который представляет собой комбинацию слоев Up-

[10] URL: https://www.tinaja.com/glib/pixintpl.pdf
[11] URL: https://www.tensorflow.org/api_docs/python/tf/nn/depth_to_space

Sampling + convolution (2 × 2) с уменьшением количества признаков в 2 раза [6]. В рассматриваемой имплементации дополнительная операция свертки не была использована.

Рассмотрим вычислительную сложность операции свертки на 4 уровне декодировщика сети интерполяционными подходами без обучения дополнительной свертки (рис. 2, *2.1*). Размер входного тензора для операции повышающей дискретизации имеет вид $H_{in} \times W_{in} \times C_{in}$ и равен $\frac{H}{2^4} \times \frac{W}{2^4} \times n\_f * 2^4$, где H_{in}, W_{in} и C_{in} – ширина, высота и количество каналов / признаков во входном признаковом представлении, а H, W – размер оригинального снимка. Слой upSampling'а увеличит размер тензора до $\frac{H}{2^3} \times \frac{W}{2^3} \times n\_f * 2^4$, после чего этот выходной тензор будет дополнен $n\_f * 2^3$ признаками из кодировщика операцией "copy and crop". Следующий сверточный слой с размером ядра $(K_H, K_W) = 3 \times 3$ делает преобразование этих $n\_f * (2^4 + 2^3) = n\_f * 24$ признаков (C_{in}) до $n\_f * 2^3$ каналов на выходе (C_{out}), сохраняя размер карт признаков. Сложность вычисления любого слоя в сети может быть измерена как общее количество операций с плавающей запятой (FLOP), необходимых для вычисления выходного представления [12]. Количество операций для сверточного слоя вида $(C_{out};\ K_H; K_W; C_{in})$ с размером выхода $H_{out} \times W_{out} \times C_{out}$ вычисляется по формуле 1:

$$FLOPS(Conv(C_{out};\ K_H; K_W; C_{in}))\ =\ H_{out} * W_{out} * C_{out} * K_H * K_W * C_{in} * 2 \qquad (1)$$

Таким образом, сложность вычисления первого сверточного слоя декодировщика равна

$$\boldsymbol{FLOPs}\left(\mathbf{Conv}\left(\boldsymbol{n_f} * \mathbf{2^3}; \mathbf{3}; \mathbf{3}; \boldsymbol{n_f} * \mathbf{24}\right)\right) = \frac{H}{2^3} * \frac{W}{2^3} * \left(n_f * 2^3\right) * 3 * 3 * \left(n_f * 24\right) * 2 =$$

$$= \mathbf{54} * \boldsymbol{H} * \boldsymbol{W} * \boldsymbol{n\_f^2}.$$

При этом количество обучаемых параметров только этого сверточного слоя будет равно (без учета обучаемого смещения каждого выхода) $C_{out} * K_H * K_W * C_{in} = (n\_f * 24) * 3 * 3 * (n\_f * 2^3) = 1728 * n\_f^2$. Следовательно, простая модель класса #B с $n\_f = 16$ будет иметь в 9 раз меньше обучаемых параметров для этого слоя, чем модель класса #A с $n\_f = 48$.

Теперь приведем вычислительную сложность первой свертки декодировщика в классической архитектуре сети U-Net 2015 (рис. 2, *2.2*). Здесь мы используем алгоритм интерполяции вместе с дополнительной сверткой, уменьшающей количество фильтров в 2 раза. Таким образом, после дополнения представления слоями из кодировщика мы получили количество слоев, соответствующее $n\_f * (2^3 + 2^3)$, вместо $n\_f * (2^4 + 2^3)$. Вычислительная сложность первой свертки кодировщика уменьшится, однако требуется также учитывать сложность вычисления дополнительной свертки. Полная сложность вычисления для двух сверток

$$\boldsymbol{FLOPS} = \begin{matrix} \boldsymbol{FLOPS}(\mathbf{Conv}(\boldsymbol{n\_f} * \mathbf{2^3}; \mathbf{2}; \mathbf{2}; \boldsymbol{n\_f} * \mathbf{2^4})) + \\ \boldsymbol{FLOPS}(\mathbf{Conv}(\boldsymbol{n\_f} * \mathbf{2^3}; \mathbf{3}; \mathbf{3}; \boldsymbol{n\_f} * \mathbf{2^4})) \end{matrix}$$

равна

$$FLOPS = \begin{matrix} \frac{H}{2^3} * \frac{W}{2^3} * \left(n_f * 2^3\right) * 2 * 2 * \left(n_f * 2^4\right) * 2 + \\ \frac{H}{2^3} * \frac{W}{2^3} * \left(n_f * 2^3\right) * 3 * 3 * \left(n_f * 2^4\right) * 2 \end{matrix} = (2^9 + 2^7 * 9) * {n_f}^2 * \frac{H}{2^3} * \frac{W}{2^3} * 2 =$$

$$= \mathbf{52} * \boldsymbol{H} * \boldsymbol{W} * \boldsymbol{n\_f^2},$$

т. е. вычислительная сложность классического декодировщика меньше, чем в рассматриваемой имплементации, при этом количество обучаемых параметров равно $(n\_f * 2^3) * 2 * 2 *$

[12] URL: https://www.thinkautonomous.ai/blog/?p=deep-learning-optimization

$(n\_f * 2^4) + (n\_f * 2^3) * 3 * 3 * (n\_f * 2^4) = 1664 * n\_f^2$, по сравнению с $1728 * n\_f^2$ для сети без обучения дополнительной свертки. Стоить отметить, что такой подход позволил не только уменьшить вычислительную сложность модели, но и добавить дополнительные обучаемые трансформации для слоя повышения разрешения, что может благоприятно сказаться на качестве обучения моделей.

Если же мы используем субпиксельную конволюцию для понижения разрешения (рис. 2, *2.3*), то на вход сверточного слоя поступит тензор размера $\frac{H}{2^3} \times \frac{W}{2^3} \times (n_f * (2^{(4-2)} + 2^3))$, тогда

$$\boldsymbol{FLOPS}\left(\mathbf{Conv}\left(\boldsymbol{n_f} * \mathbf{2^3}; \mathbf{3}; \mathbf{3}; \boldsymbol{n_f} * \mathbf{12}\right)\right) = \frac{H}{2^3} * \frac{W}{2^3} * \left(n_f * 2^3\right) * 3 * 3 * \left(n_f * 12\right) * 2 =$$

$$= \mathbf{27} * \boldsymbol{H} * \boldsymbol{W} * \boldsymbol{n\_f^2}.$$

Таким образом, операция преобразования глубины в пространство позволяет уменьшить вычислительную сложность модели в 2 раза.

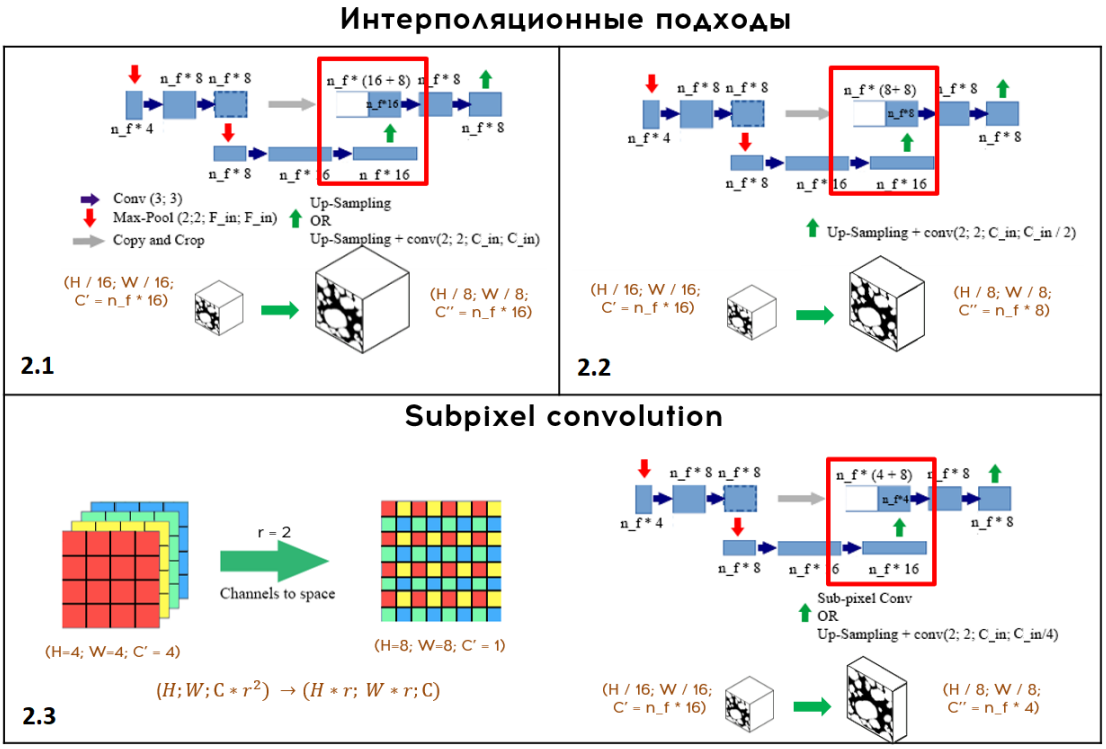

Рис. 2. Подходы к повышению разрешения признаковых представлений изображения:
2.1 – с сохранением количества признаков предыдущего уровня;
2.2 – с сокращением количества признаков в 2 раза;
2.3 – сС сокращением количества признаков в 4 раза

Fig. 2. Approaches to increasing the resolution of feature representations of an image:
2.1 – with the preservation of the number of features of the previous level;
2.2 – with a reduction in the number of features by 2 times;
2.3 – with a 4-fold reduction in the number of features

В данной работе в качестве эксперимента также были протестированы модели, использующие после слоя UpSamplinga свертку, уменьшающую количество признаковых представлений в 4 раза, по аналогии со слоем субпиксельной конволюции. Таким образом, после до-

полнения представления слоями из кодировщика мы получили количество слоев, соответствующее $n\_f * (2^2 + 2^3)$. Полная сложность вычисления для двух сверток

$$\boldsymbol{FLOPS} = \begin{matrix} \boldsymbol{FLOPS}(\mathbf{Conv}(\boldsymbol{n_f} * \mathbf{2^2}; \mathbf{2}; \mathbf{2}; \boldsymbol{n_f} * \mathbf{2^4})) + \\ \boldsymbol{FLOPS}(\mathbf{Conv}(\boldsymbol{n\_f} * \mathbf{2^3}; \mathbf{3}; \mathbf{3}; \boldsymbol{n\_f} * \mathbf{12})) \end{matrix}$$

равна

$$FLOPS = 2 * \frac{H}{2^3} * \frac{W}{2^3} * \left(\left(n_f * 2^2\right) * 2 * 2 * \left(n_f * 2^4\right) + \left(n_f * 2^3\right) * 3 * 3 * \left(n_f * 12\right)\right) = \\ = \mathbf{35} * \boldsymbol{H} * \boldsymbol{W} * \boldsymbol{n\_f^2}.$$

Количество обучаемых параметров при этом равно

$$(n\_f * 2^2) * 2 * 2 * (n\_f * 2^4) + (n\_f * 2^3) * 3 * 3 * (n\_f * 12) = 1120 * n\_f^2.$$

Новое пространство поиска $z_3 \subset Z$:

$$z_3 = z_2' \times decoder\_method = \{\begin{matrix} \text{Nearest Neighbor}, \\ \text{Bilinear Interpolation}, \\ \text{Nearest Neighbor} + conv(C_{in}; 2; 2; \mathrm{C_{out}} = C_{in}/2), \\ \text{Bilinear Interpolation} + conv(C_{in}; 2; 2; \mathrm{C_{out}} = C_{in}/2), \} \\ \text{Nearest Neighbor} + conv(C_{in}; 2; 2; \mathrm{C_{out}} = C_{in}/4), \\ \text{Bilinear Interpolation} + conv(C_{in}; 2; 2; \mathrm{C_{out}} = C_{in}/4), \\ \text{Subpixel Convolution} \end{matrix}$$

Для ускорения процесса обучения моделей, соответствующих пространству поиска z_3, был прменен подход Transfer learning [13]. Так, каждая модель использовала соответствующие множеству z_2 предобученные веса кодировщика сети, ранее оптимизированные со слоями Nearest Neighbor в этапе декодера, а часть декодировщика заменена в соответствии с анализируемым гиперпараметром повышения разрешения и заново инициирована теми же начальными значениями. Так как часть декодировщика сети не являлась обученной, первые 10 эпох веса кодировщика были зафиксированы. На протяжении последующих 20 эпох был применен подход fine-tuning [13], при котором происходило полное обучение модели. Таким образом, предобученный кодировщик позволил ускорить процесс обучения и изучить сходимость моделей на более поздних этапах обучения.

Требовалось протестировать $||z_3|| = 126$ моделей. В табл. 4 приведено качество сегментации моделей с группировкой (усреднением) по количеству базовых фильтров и способу декодировки. Результирующая сложность моделей приведена в табл. 5: во второй строке указано количество обучаемых параметров для моделей, не использующих понижение каналов в операции дискретизации, а в последующих строках – отклонение, численно отображающее уменьшение количества параметров при обучении дополнительной свертки 2 × 2 с понижением количества выходных каналов.

При обучении модели с базовым количеством фильтров *n_f* = 8 и сокращением количества параметров посредством обучения дополнительных трансформаций было отмечено, что качество сегментации и сходимость моделей сильно зависят от способа инициализации параметров переобучаемого декодировщика. При обучении моделей с большим числом параметров влияние начальных инициализаторов уменьшалось.

Интерполяция ближайшим соседом уступает по результирующему качеству сегментации билинейной интерполяции, но усложнение самого способа интерполяции увеличивает время на прохождение прямого и обратного проходов сети. Ранее при тестировании моделей по количеству базовых фильтров были определены размеры батчей, обеспечивающих наилучшую загрузку процессора и видеокарты. В табл. 6 приведена оценка времени сегментации

[13] URL: https://www.tensorflow.org/tutorials/images/transfer_learning?hl=en

одного 3D объема для моделей разного уровня сложности, вычисленная как среднее значение времени сегментации на 3-х повторных запусках модели при оптимальном размере батча. При запуске моделей на CPU было выявлено, что время вывода для моделей, использующих билинейную интерполяцию в части сетевого декодера, увеличилось в два раза. При тестировании на GPU время вывода при использовании разных интерполяционных подходов является сопоставимым. С другой стороны, обучение дополнительных сверточных слоев, несмотря на сокращение количества обучаемых параметров, приводит к увеличению времени прямого прохода сети (вывода).

Таблица 4

Достигнутое значение метрики качества 1-BCE на проверочном наборе при настройке моделей по способу декодировки с группировкой по количеству базовых обучаемых фильтров свертки *n_f*

Table 4

Achieved value of the 1-BCE quality metric on validation set when configuring models according to the decoding method with grouping by the number of basic trained convolution filters *n_f*

Способ декодировки	Число фильтров *n_f* (%)					
	8	16	24	32	40	48
Nearest Neighbor	89,33	92,16	93,21	93,30	93,52	93,63
Bilinear	89,34	92,15	93,26	93,43	93,58	93,72
Nearest Neighbor + conv(C_{in};2;2;C_{out}=C_{in}/2)	84,74	92,04	93,15	93,24	93,43	93,51
Bilinear + conv(C_{in};2;2;C_{out}=C_{in}/2)	86,01	92,18	93,30	93,35	93,47	93,61
Nearest Neighbor + conv(C_{in};2;2;C_{out}=C_{in}/4)	79,73	91,63	93,09	93,26	93,51	93,66
Bilinear + conv(C_{in};2;2;C_{out}=C_{in}/4)	79,76	91,65	93,16	93,33	93,60	93,72
SubConv	70,27	90,23	92,30	92,80	93,21	93,48

Таблица 5

Оптимизация вычислительной сложности моделей по сравнению с моделями, использующими ближайшего соседа или билинейную интерполяцию в качестве способа декодировки сети

Table 5

Optimization of computational complexity of models, compared to models using nearest neighbor or bilinear interpolation as a method of network deconvolution

Способ декодировки	Число фильтров *n_f*					
	8	16	24	32	40	48
Nearest Neighbor / Bilinear	490 993	1 962 337	4 414 033	7 846 081	12 258 481	17 651 233
Nearest/ Bilinear + conv(C_{in};2;2;C_{out}=C_{in}/2)	–5 440	–21 760	–48 960	–87 040	–136 000	–195 840
Nearest / Bilinear + conv(C_{in};2;2;C_{out}=C_{in}/4)	–51 680	–206 720	–465 120	–826 880	–1 292 000	–1 860 480
SubConv	–73 440	–293 760	–660 960	–1 175 040	–1 836 000	–2 643 840

Таблица 6

Тестирование времени генерации масок сегментации
для одного 3D объема на CPU и GPU

Table 6

Testing the generation time of segmentation masks
for one 3D volume on CPU and GPU

Способ декодировки	Усредненное по инициализаторам время сегментации на устройстве в мин.					
	Число фильтров *n_f*					
	CPU			GPU		
	24	32	40	24	32	40
Nearest Neighbor	2,34	3,27	4,29	0,41	0,49	0,66
Bilinear	4,45	5,99	7,55	0,41	0,48	0,65
Nearest Neighbor + conv(C_{in};2;2;C_{out}=C_{in}/2)	2,37	3,15	4,18	0,44	0,53	0,72
Bilinear + conv(C_{in};2;2;C_{out}=C_{in}/2)	4,52	6,04	7,72	0,44	0,53	0,72
Nearest Neighbor + conv(C_{in};2;2;C_{out}=C_{in}/4)	2,27	2,81	4,11	0,43	0,50	0,66
Bilinear + conv(C_{in};2;2;C_{out}=C_{in}/4)	4,31	5,60	7,50	0,43	0,49	0,66
SubConv	2,28	2,82	4,16	0,37	0,42	0,56

Операция субпиксельной конволюции позволяет добиться значительного упрощения сложности модели, но при этом увеличивает сходимость моделей – потребуется провести большее количество обучающих эпох, чтобы достигнуть сопоставимого качества сегментации. При этом модель, использующая дискретизацию ближайшим соседом и обучаемой сверткой с сокращением количества выходных каналов в 4 раза, позволяет добиться лучшего качества классификации пикселей за то же время вывода предсказания.

Для моделей с $n\_f = 16$ и $n\_f = 24$ фильтрами больший прирост качества был достигнут при обучении декодировщика с билинейной интерполяцией и дополнительной сверткой с понижением каналов в 2 раза. Однако для решения оптимизационной задачи #A модели большей сложности, использующие только интерполяцию ближайшим соседом, позволяют добиться лучшего качества сегментации. Несмотря на то что интерполяция ближайшим соседом в некоторых случаях дает большую погрешность приближения данных, можно попробовать добиться улучшения качества сегментации в процессе дообучении этих моделей.

Так как все модели с $n\_f = \{24, 32, 40, 48\}$, не использующие субпиксельную конволюцию части сетевого декодера, являются сопоставимыми по качеству сегментации, для выбора оптимальных моделей следует учитывать время на настройку обучающих параметров и время на получение маски сегментации по классу гранулы для трех 3D объемов КТ-изображений.

Выводы

Реализация базовой архитектуры нейронной сети не всегда является лучшим решением при ее применении в системах с ограниченным временем обработки входного потока данных. На основе анализа моделей архитектуры UNet по пространству гиперпараметров, огра-

ниченных переменными количества обучаемых фильтров в операциях свертки, регуляризации их выходов и способа увеличения разрешения в части сетевого декодера, был осуществлен поиск моделей сегментации, удовлетворяющих ограничениям оптимизационных задач. Такой поиск позволил получить первичное представление об аппроксимирующей способности моделей, обладающих меньшей вычислительной сложностью.

После анализа моделей по способу декодровки признакового представления на каждом уровне сети UNet, для группы моделей с базовым числом фильтров $n_f \in [24, 32, 40, 48]$ и применением линейной интерполяции для дискретизации признакового представления изображения в сети было достигнуто сопоставимое качество сегментации на валидационной выборке. Модели отличались только временем генерации масок сегментации.

- Для решения оптимизационной задачи класса #A может быть выбрана модель с 32 базовыми фильтрами *n_f* в слоях свертки с обучаемым смещением, использующая в качестве метода повышения дискретизации интерполяцию ближайшим соседом и дополнительный сверточный слой, сокращающий количество каналов представления в 4 раза. Время сегментации трех 3D объемов на CPU составляет порядка 8,5 минуты, на GPY – 1,5 минуты. Такая модель обладает в 4,7 раза меньшим числом параметров, чем классическая архитектура UNet, и позволяет достигнуть качества сегментации на тестовой выборке 1-BCE = 93,3 %.
- Решением оптимизационной задачи класса #B является модель с 24 базовыми фильтрами *n_f* в слоях свертки с обучаемым смещением. В качестве способа декодировки может быть использована интерполяция ближайшим соседом без сокращения количества каналов. Стоит отметить, что это же решение в случае дообучения модели может быть применено для оптимизационной задачи #A, при этом последний блок декодировщика может быть заменен билинейной интерполяцией, так как сложность модели в 1,5 раза меньше, чем модели класса #A.

Поскольку оптимальная 2D модель будет использоваться для создания обучающей выборки для 3D модели, полученные изображения требуют дополнительного анализа и доразметки. Отобразив разницу интенсивностей пикселей между целевой и небинаризированной прогнозируемой маской сегментации, можно выявить, на каких областях изображения модель выдает ошибочный ответ классификации. Так, на рис. 3 для первого изображения погрешность составляет 7,89 %, для второго – 9,04 %. Качество модели снижается при прогнозировании границ по классу гранулы и при классификации сильно засвеченных гранул. С другой стороны, эти области являются самыми трудными для ручной разметки, поэтому автоматизация с помощью нейронной сети может значительно сократить время подготовки наборов изображений.

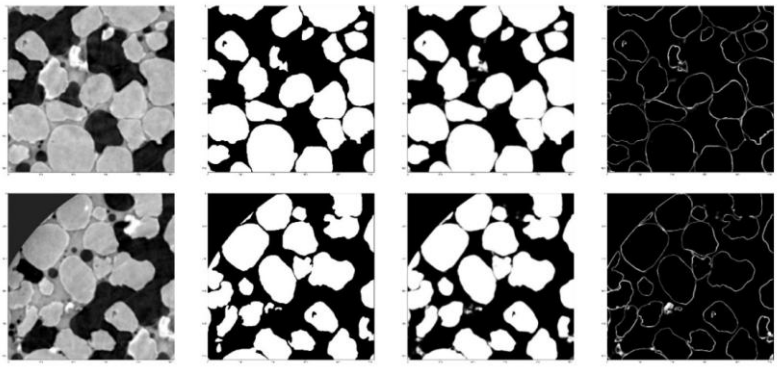

Рис. 3. Слева направо: изображение, таргетная маска сегментации по классу гранулы, сгенерированный вероятностный ответ модели, погрешность предсказания
Fig. 3. From left to right: Image, target segmentation mask by granule class, generated probabilistic response of the model, prediction error

Список литературы

1. **Chong Z. R., Yang S. H. B., Babu P., Linga P., Li X.-S.** Review of natural gas hydrates as an energy resource: prospects and challenges. *Applied Energy*, 2016, vol. 162, pp. 1633–1652. DOI 10.1016/j.apenergy.2014.12.061
2. **Makogon Y. F., Omelchenko R. Y.** Commercial gas production from Messoyakha deposit in hydrate conditions. *Journal of Natural Gas Science and Engineering*, 2013, vol. 11, pp. 1–6. DOI 10.1016/j.jngse.2012.08.002
3. **Кадыров Р. И.** Рентгеновская компьютерная томография в геологии: Учеб.-метод. пособие. Казань, Казан. федеральный ун-т, 2020. 37 с.
4. **Дробчик А. Н., Дугаров Г. А., Дучков А. А., Купер К. Э.** Акустические измерения и рентгеновская томография песчаных образцов, содержащих гидрат ксенона // Геофизические технологии. 2019. № 4. С. 17–23. DOI 10.18303/2619-1563-2019-4-17
5. **Nikitin V. V., Dugarov G. A., Duchkov A. A., Fokin M. I., Drobchik A. N., Shevchenko P. D., De Carlo F., Mokso R.** Dynamic in-situ imaging of methane hydrate formation and self-preservation in porous media. *Marine and Petroleum Geology*, 2020, vol. 115. DOI 10.1016/j.marpetgeo.2020.104234
6. **Ronneberger O., Fischer P., Brox T.** U-Net: Convolutional networks for biomedical image segmentation. *International Conference on Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. Lecture Notes in Computer Science*, 2015, vol. 9351, pp. 234–241. DOI 10.1007/978-3-319-24574-4_28
7. **Luo W., Li Y., Urtasun R., Zemel R.** Understanding the effective receptive eld in deep convolutional neural networks. In: Advances in Neural Information Processing Systems. Barcelona, 2016, pp. 4898–4906.
8. **Yi-de M., Qing L., Zhi-Bai Q.** Automated image segmentation using improved PCNN model based on cross-entropy. *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing*, 2004, pp. 743–746. DOI 10.1109/ISIMP.2004.1434171
9. **Rahman M. A., Wang Y.** Optimizing intersection-over-union in deep neural net-works for image segmentation. *International symposium on visual computing*, 2016, pp. 234–244. DOI 10.1007/978-3-319-50835-1_22
10. **Kingma D., Ba J.** Adam: A Method for Stochastic Optimization. In: Proceedings of the 3rd International Conference on Learning Representations, 2014.
11. **Chollet F.** Deep Learning with Python Data representations for neural networks. Shelter Island, NY, Manning Publications, 2018, pp. 31–38.
12. **Ioffe S., Szegedy C.** Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on Machine Learning (PMLR)*, 2015, vol. 37, pp. 448–456.
13. **Shi W., Caballero J., Huszar F., Totz J., Aitken A.P., Bishop R., Rueckert D., Wang Z.** Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883. DOI 10.1109/cvpr.2016.207

References

1. **Chong Z. R., Yang S. H. B., Babu P., Linga P., Li X.-S.** Review of natural gas hydrates as an energy resource: prospects and challenges. *Applied Energy*, 2016, vol. 162, pp. 1633–1652. DOI 10.1016/j.apenergy.2014.12.061
2. **Makogon Y. F., Omelchenko R. Y.** Commercial gas production from Messoyakha deposit in hydrate conditions. *Journal of Natural Gas Science and Engineering*, 2013, vol. 11, pp. 1–6. DOI 10.1016/j.jngse.2012.08.002

3. **Kadyrov R. I.** Rentgenovskaya kompyuternaya tomografiya v geologii. Uchebno-metodicheskoe posobie. Kazan, Kazan Federal University, 2020, 37 p. (in Russ.)
4. **Drobchik A. N., Dugarov G. A., Duchkov A. A., Kuper K. E.** Acoustic measurements and X-ray tomography of sand samples containing xenon hydrate. *Russian Journal of Geophysical Technologies*, 2019, no. 4, pp. 17–23. (in Russ.) DOI 10.18303/2619-1563-2019-4-17
5. **Nikitin V. V., Dugarov G. A., Duchkov A. A., Fokin M. I., Drobchik A. N., Shevchenko P. D., De Carlo F., Mokso R.** Dynamic in-situ imaging of methane hydrate formation and self-preservation in porous media. *Marine and Petroleum Geology*, 2020, vol. 115. DOI 10.1016/j.marpetgeo.2020.104234
6. **Ronneberger O., Fischer P., Brox T.** U-Net: Convolutional networks for biomedical image segmentation. *International Conference on Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. Lecture Notes in Computer Science*, 2015, vol. 9351, pp. 234–241. DOI 10.1007/978-3-319-24574-4_28
7. **Luo W., Li Y., Urtasun R., Zemel R.** Understanding the effective receptive eld in deep convolutional neural networks. In: Advances in Neural Information Processing Systems. Barcelona, 2016, pp. 4898–4906.
8. **Yi-de M., Qing L., Zhi-Bai Q.** Automated image segmentation using improved PCNN model based on cross-entropy. *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing*, 2004, pp. 743–746. DOI 10.1109/ISIMP.2004.1434171
9. **Rahman M. A., Wang Y.** Optimizing intersection-over-union in deep neural net-works for image segmentation. *International symposium on visual computing*, 2016, pp. 234–244. DOI 10.1007/978-3-319-50835-1_22
10. **Kingma D., Ba J.** Adam: A Method for Stochastic Optimization. In: Proceedings of the 3rd International Conference on Learning Representations, 2014.
11. **Chollet F.** Deep Learning with Python Data representations for neural networks. Shelter Island, NY, Manning Publications, 2018, pp. 31–38.
12. **Ioffe S., Szegedy C.** Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on Machine Learning (PMLR)*, 2015, vol. 37, pp. 448–456.
13. **Shi W., Caballero J., Huszar F., Totz J., Aitken A.P., Bishop R., Rueckert D., Wang Z.** Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883. DOI 10.1109/cvpr.2016.207

Информация об авторах

Татьяна Олеговна Колесник, студентка магистратуры
Антон Альбертович Дучков, заведующий лабораторией

Information about the Authors

Tatyana O. Kolesnik, Master's Student
Anton A. Duchkov, Head of the Laboratory

Статья поступила в редакцию 11.12.2021;
одобрена после рецензирования 01.02.2022; *принята к публикации 01.02.2022*
The article was submitted 11.12.2021;
approved after reviewing 01.02.2022; *accepted for publication 01.02.2022*

Научная статья

УДК 004.93
DOI 10.25205/1818-7900-2022-20-1-47-56

Биологически-подобные модели сверточных нейронов в задаче распознавания иллюзорного контура

Александр Владимирович Кугаевских [1]
Максим Сергеевич Берьянов [2]

[1] Новосибирский государственный университет
Новосибирск, Россия

[2] Университет ИТМО
Санкт-Петербург, Россия

[1] a-kugaevskikh@yandex.ru, https://orcid.org/0000-0002-6676-0518
[2] berjnov.ru@mail.ru

Аннотация

Представлен результат проектирования архитектуры нейронной сети на биологически-подобных нейронах, в задачу которой входит отработка механизма распознавания иллюзорного контура на примере «фигур Канижа». Нейронная сеть позволила добиться инвариантности к количеству углов фигуры и не теряет в качестве распознавания при изменении размеров иллюзорного контура. Основное применение подхода может быть найдено в задаче разделения «фигура – фон» на изображениях.

Ключевые слова

иллюзорный контур, нейронная сеть, биологически-подобная модель, выделение краев, нейроны конца линий, сегментация

Для цитирования

Кугаевских А. В., *Берьянов М. С.* Биологически-подобные модели сверточных нейронов в задаче распознавания иллюзорного контура // Вестник НГУ. Серия: Информационные технологии. 2022. Т. 20, № 1. С. 47–56. DOI 10.25205/1818-7900-2022-20-1-47-56

Bio-Inspired Models of Convolution Neurons in the Problem of Illusory Contour Recognition

Alexander V. Kugaevskikh [1], **Maksim S. Beryanov** [2]

[1] Novosibirsk State University
Novosibirsk, Russian Federation

[2] ITMO University
St. Petersburg, Russian Federation

[1] a-kugaevskikh@yandex.ru, https://orcid.org/0000-0002-6676-0518
[2] berjnov.ru@mail.ru

Abstract

This paper presents the result of designing the architecture of a neural network on bio-inspired neurons, whose task is to work out the mechanism for recognizing an illusory contour using the example of “Kanizsa’s figures”. The neural network made it possible to achieve invariance to the number of corners of the figure and does not lose recognition

© Кугаевских А. В., Берьянов М. С., 2022

quality when changing the size of the illusory contour. The main application of the approach can be found in the problem of separating “figure-background” in images.

Keywords

illusory contour, neural network, bio-inspired model, edge detection, end-stopped neuron, segmentation

For citation

Kugaevskikh A. V., Beryanov M. S. Bio-Inspired Models of Convolution Neurons in the Problem of Illusory Contour Recognition. *Vestnik NSU. Series: Information Technologies*, 2022, vol. 20, no. 1, pp. 47–56. (in Russ.) DOI 10.25205/1818-7900-2022-20-1-47-56

Введение

Распознавание иллюзорного контура является весьма интересной проблемой. С одной стороны, она не имеет особого практического применения, с другой – механизм, лежащий в ее основе, отвечает и за разделение изображения на фигуру и фон, а также участвует в сегментации перекрытых изображений. В зрительной коре за распознавание иллюзорных контуров, помимо прочих функций, отвечают нейроны первичных зрительных зон [1].

Иллюзорный контур, простыми словами, является условно воспринимаемым контуром без присутствия его на самом деле, физически. Исследуя примеры сцен, где светлая фигура закрывает, если точнее выразиться, частично покрывает темные объекты на том же светлом фоне, светлая фигура кажется более явной, чем фон, несмотря на их совместную гармонию, следовательно, края и тело данной фигуры создают иллюзорный контур, что немаловажно, при содействии темных объектов.

Для распознавания иллюзорных контуров применялся практически весь аппарат компьютерного зрения, от фильтров Габора до марковских случайных полей. Применяют также и нейронные сети. Нейросетевые модели распознавания иллюзорного контура [2–6] основаны на выделении характеристических точек. Не прекращаются попытки создать механизм определения иллюзорного контура без выделения характеристических точек, основываясь только на статистиках естественного изображения [7].

1. Архитектура нейронной сети

Мы предлагаем использовать сверточную нейронную сеть для определения иллюзорного контура. Отличие от других работ заключается в моделях нейронов, участвующих в этом процессе. Общая архитектура представлена на рис. 1. На изображении детектируются края (US1) и фон (UB1). Информация о краях служит основной для детекции границ (UC2), концов линий (UE2) и длинных линий (UL2), которые на следующем слое уточняются (UBC3, UBE3 и UBL3). Уточнение требуется из-за особенностей функционирования нейронов, которые могут давать ошибку определения ориентации края в пределах ±20° от правильной ориентации.

Рассмотрим задачу распознавания иллюзорного контура на примере треугольника Канижа (рис. 2) и его производных. Определив края и углы имеющихся фигур, мы можем построить ребра иллюзорного треугольника, для этого служат нейроны длинных линий. Ориентация, как и количество углов у фигуры Канижа, может быть любой, поэтому нам необходимо выделять ребра разной ориентации и при этом избегать ложных срабатываний. Ориентация ребра фигуры Канижа совпадает с ориентацией луча темной фигуры (условно ее называют “Pacman”), с помощью нейрона конца линий можно подтвердить, что этот луч в определенном месте обрывается и продолжается уже на другой темной фигуре. Определив точно края, углы, концы линий и длинные линии, что расширяет представление характеристических точек, мы можем детектировать ребра фигуры Канижа (UR4), на основании которых определяются пары ребер (слой UP5) и сама фигура (слой UI6).

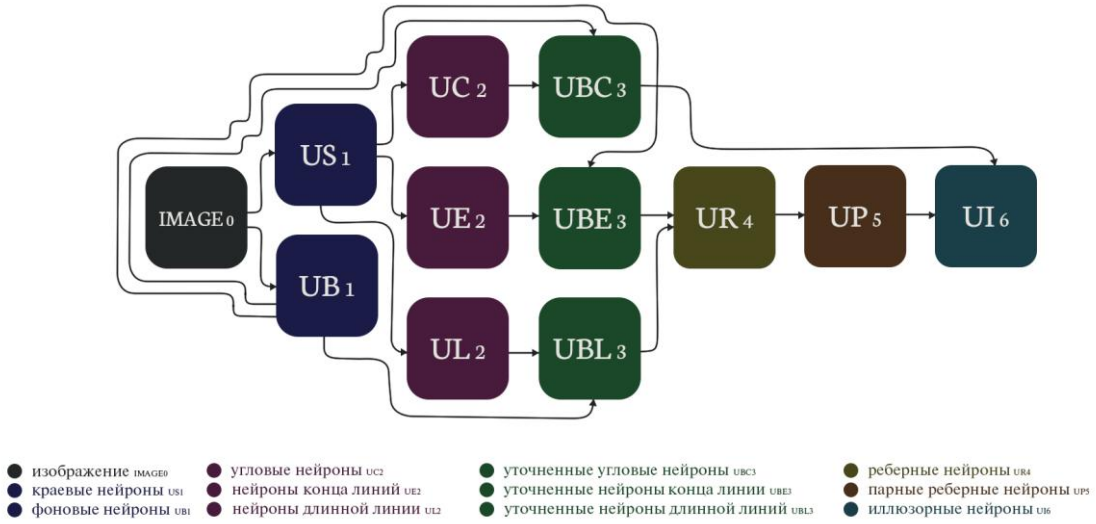

Рис. 1. Схема архитектуры нейронной сети
Fig. 1. Neural network architecture

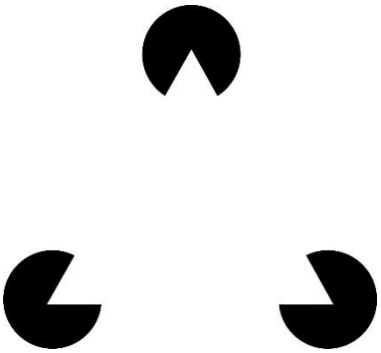

Рис. 2. Треугольник Канижа
Fig. 2. Kanizsa's triangle

2. Модели характеристических нейронов

2.1 Слой US1

Ранее мы подробно описывали модели нейронов детекции краев, углов и фона [8]. Для выделения краев и углов в нейросетевых моделях часто используют фильтр Габора. Мы также используем фильтр Габора для формирования матрицы весов нейронов US1. Нейроны этого слоя реагируют на линии предпочтительной ориентации θ. Фильтр Габора был выбран по двум причинам: во-первых, его функция позволяет формировать рецептивное поле простого нейрона для выделения линий определенной ориентации; во-вторых, в отличие от вейвлета «мексиканская шляпа» его постоянная компонента гораздо ближе к 0, что уменьшает паразитную активацию нейронов первого слоя. Равномерная заливка рецептивного поля такого нейрона дает очень близкое к нулю значение. Фильтр Габора позволяет выделять светлые линии на темном фоне и темные линии на светлом фоне.

$$G_{1,2} = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)\cos\left(2\pi\frac{x'}{\lambda} + \phi\right)$$

где ϕ – фаза, определяющая симметричность и чередование тормозных и возбуждающих областей ядра (в радианах); σ – масштаб фильтра; γ – степень эллиптичности фильтра.

Для реагирования на компоненты определенной ориентации, в фильтр Габора вводятся параметры поворота координатной сетки на угол θ:

$$\begin{cases} x' = x\cos\theta + y\sin\theta \\ y' = -x\sin\theta + y\cos\theta \end{cases}$$

Так как фильтр имеет ярко выраженную ориентационную избирательность, градус отклонения между ориентациями (Δθ) влияет на выявление деталей и, как следствие, на уровень шума.

Параметр ϕ отвечает за симметричность ядра фильтра. Он может принимать значения –π/2 или π/2 для выявления антисимметричных компонент и –π, 0 или π для симметричных компонент; γ определяет вытянутость ядра фильтра по оси ординат.

Многочисленная практика применения фильтра Габора доказала, что оптимальное значение σ подбирается из соотношения σ/λ, выведенного через пропускную способность:

$$\frac{\sigma}{\lambda} \approx 0{,}56.$$

Для задачи выделения краев фильтр Габора имеет следующие параметры: $\gamma = 0.1$, $\theta \in [0{,}350]$, $\lambda = 3$. В итоге фильтр Габора позволяет выделять светлые линии на темном фоне (ϕ = –π) и темные линии на светлом фоне (ϕ = 0). Карта рецептивного поля нейрона на базе фильтра Габора представлена на рис. 3.

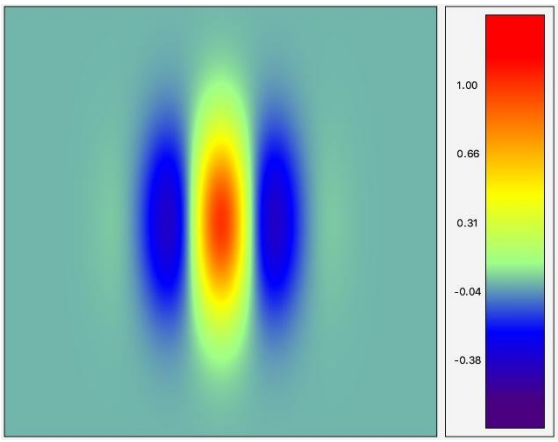

Рис. 3. Карта рецептивного поля нейрона на базе фильтра Габора

Fig. 3. Receptive field of neuron on base Gabor filter

Нейроны слоя US1 дают близкую по значению активацию для линий в пределах ±20° отклонения от предпочтительной ориентации. Это может создать проблему для точного определения ориентации края.

2.2 Слой UB1

Нейроны этого типа нужны для учета площади занимаемого пространства частей фигуры на изображении, посредством которой получается иллюзорный контур. Смысл в том, что внутри области действия фильтра, отклик которого является релевантным, следует вычислить описанную площадь покрытия. Достаточно близкие к максимальному значения такого расчета помогают выбрать наиболее нужные нам срабатывания нейронов при условии первоначального отсечения заведомо нерелевантных срабатываний, так как угол в общем случае покрывает наибольшую площадь частей фигуры внутри области действия фильтра.

Нейроны этого типа также описывались нами в [8] и основаны на использовании гиперболического тангенса

$$G = \tanh\left(\frac{2x'}{\lambda}\right).$$

На рис. 4 показано рецептивное поле такого нейрона. Для получения нейрона фона слоя UB1 выходы представленных нейронов комбинируются аналогичным UC2 способом.

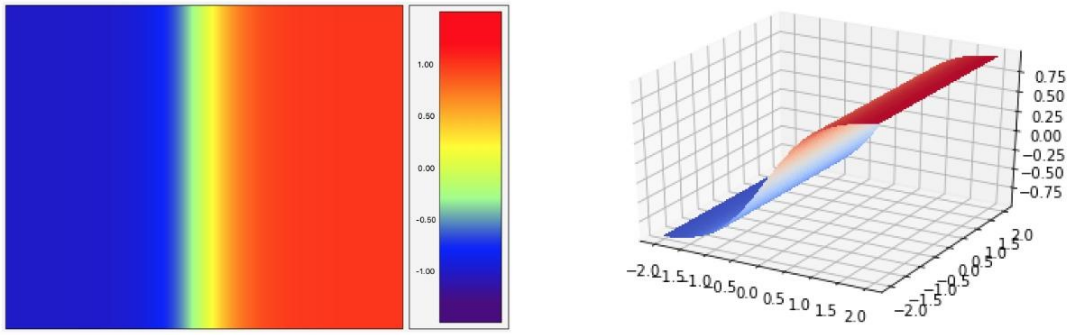

Рис. 4. Рецептивное поле по формуле гиперболического тангенса
Fig. 4. Receptive field of neuron on base hyperbolic tangents

2.3 Слой UC2

Слой UC2 составляют сложные клетки, реагирующие на сочетания линий, углы и многоугольники. Хотя основная функция сложных клеток в зрительной коре – это реакция на движение, мы пока не моделируем данную функцию. Нейроны второго слоя получают входы от соответствующих по типу нейронов первого слоя. Эти связи являются обучаемыми. Обучение производилось с помощью обратного распространения ошибки. Для обучения и экспериментальной проверки поведения разных фильтров были сформированы пакеты черно-белых изображений.

$$UC2(m) = \sum_{m \in M} w_m US1(m),$$

где M – пространство связей всех пар нейронов US1 слоя; w_m – значение веса в зависимости от пары нейронов.

На обучающей выборке было получено 100 % качество выделения сочетаний линий при условии минимального отклонения углов ориентации ($\Delta\theta = 10°$) и размера рецептивного поля сложной клетки – 13 пикселей, таким образом, оптимальным размером ядра простой клетки на базе фильтра Габора является 7*7 пикселей.

2.4 Слой UE2

Концы линий в слое UE2 выделяются с помощью разработанной нами модели [9]. Предложенная нами модель основана на принципе сложной клетки слоя UC2, но принцип формирования рецептивного поля отличается. В случае UC2 связи проводились от нейронов совпадающей фазы, в случае же нейрона конца линий – от нейронов, находящихся в противофазе (рис. 5).

$$C_e = G_t(x', y', \varphi_1) + G_t(x' + \Delta x, y' + \Delta y, \varphi_2 = \varphi_1 + \pi).$$

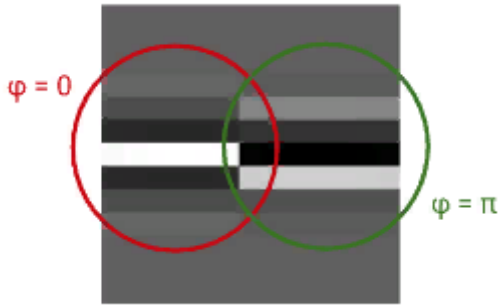

Рис. 5. Схема работы нейрона конца линий
Fig. 5. Scheme of end-stopped neuron

Продолжение линии в пределах рецептивного поля нейрона, находящегося в противофазе к другому нейрону, приведет к уменьшению активации. Таким образом, максимальная активация такого нейрона будет получена на конце линии (рис. 6).

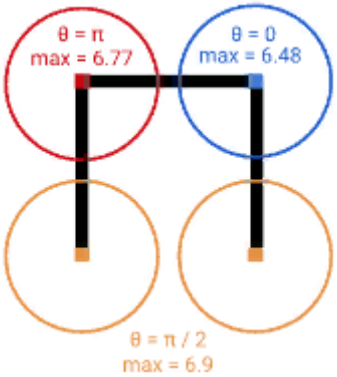

Рис. 6. Результат работы модели нейрона
Fig. 6. Result of the neuron model

В отличие от популярной модели Хейтгера [10], данная модель нейрона конца линий менее сложна в вычислительном плане.

2.5 Слой UL2

Помимо выделения углов и выделения концов линий, нужно каким-либо образом понимать расстояние между частями фигуры, посредством которых создается иллюзорный контур. Для этого были разработаны нейроны, призванные реагировать на длинные линии.

Цель нейронов – определить области изображения, где имеется некоторое прерывание между частями фигур, которое в дальнейшем можно будет распознать как одну из составляющих нашего иллюзорного контура.

Для получения правильного отклика нейрона UL2 происходит построение связей между слоями, которые повторяют воздействие фильтра, строящегося практически по тому же принципу, что и блок фильтра для нейрона UE2. Однако существует разница, которая проявляется в том, что активная область фильтра является симметричной относительно центра, т. е., во-первых, используются два фильтра Габора малого размера 7 × 7, во-вторых, они одинаковые по фазе. Такая реализация выбрана для того, чтобы, сопоставив несколько таких нейронов друг с другом, можно было получить отклики, по существу, от некоторой линии, размер которой заведомо больше вычисляемого в слое US1.

$$UL2(x) = \sum_{x \in X} US1(x),$$

где X – пространство связей соединений,

$$\{i_1, \dots, i_N : \theta_{UC2(i_1)} = \dots = \theta_{UC2(i_N)}, \varphi_{UC2(i_1)} = \dots = \varphi_{UC2(i_N)}, i_1, \dots, i_N \in Y\},$$

где Y – пространство связей внутри большого рецептивного поля, которые сопоставляются друг с другом для построения длинных линий.

Первоначально указывается размер предполагаемой длинной линии. Он будет влиять на прокладываемые связи и их количество. Технически матричный размер фильтра предполагаемой длинной линии составляет (13 * N) × 13. Однако здесь неуместно говорить о фильтре как таковом, это сделано для упрощения представления. Как уже было сказано, конфигурация нейронов UL2 состоит в том, что к каждому нейрону данного типа определенным образом проводятся правильно выбранные связи от нейронов слоя US1. Количество таких связей для каждого нейрона UL2, как опять же было сказано ранее, равняется размеру предполагаемой длинной линии, это настраиваемое значение.

2.6 Нейроны уточнения (UBC3, UBE3 и UBL3)

Нейроны этих типов комбинируют выходы предыдущего слоя, позволяют уточнять границы краев, расположение концов линий, ориентацию длинных линий, чтобы нивелировать ошибки предшествующих нейронов.

$$UBC3(i) = \sum_{m \in M}(UB1(i) + UC2(m)), \text{где } i \leftrightarrow \text{область нейрона m UC2 слоя.}$$
$$UBE3(k) = \sum_{k \in K}(UB1(i) + UE2(k)), \text{где } i \leftrightarrow \text{область нейрона } k \text{ } UE2 \text{ слоя.}$$
$$UBL3(x) = \sum_{x \in X} UB1(i) + UL2(x), \text{где } i \leftrightarrow \text{нейрона } x \text{ слоя } UL2, \text{то есть } i \leftrightarrow \{x_1, \dots, x_N\} \text{ множественных составляющих нейрона слоя } UC2.$$

3. Выделение иллюзорного контура

3.1 Нейроны детекции ребер (UR4)

Иллюзорный контур – это замкнутый силуэт, поэтому следует искать ребра, т. е. составляющие контура. Для реализации такого поиска можно использовать нейроны длинных линий UBL3 и нейроны конца линий UBE3.

Суть такой идеи такова, что, суммируя отклик длинной линии, которая уже является достаточно релевантной и подходит под кандидата на валидную составляющую иллюзорного контура, и отклик, по той же схеме, релевантного нейрона конца линий, совокупный эффект выдаст больший отклик, чем случай несовпадения области концов линий и области длинной линии. Более того, это также работает для описанного выше случая, когда область прерывания не перекрывается полностью. Стоит отметить, что ориентации нейронов UBL3 и нейронов UBE3 должны быть одинаковыми.

$$UR4(x) = \sum_{x \in X} UBE3(i) + UBL3(x), \text{где } \theta_i = \theta_x, i \leftrightarrow \{x_1 \dots x_N\}: \text{составляющие нейрона } UBL3(x).$$

3.2 Нейроны детекции пар ребер (UP5)

Данный слой будет содержать в себе множество соединений всех релевантных ребер в размере двух для подготовки к вовлечению влияния нейронов из UBC3 слоя для максимально корректного выделения иллюзорного контура. Таким образом, мы получим слой, в котором наибольший отклик будет у нейронов, которые сопоставимы с уже выделенными ребрами на прошлом шаге с известными ориентациями.

$$UP5(v) = \sum_{v \in V} UR4(v),$$

где V – пространство связей пар всех нейронов слоя UR4.

3.3 Нейроны распознавания контура (UI6)

Цель нейронов этого слоя – сопоставить имеющиеся в UBC3 слое угловые отклики и отклики, фактически, угловых ребер UP5 вплоть до обеих ориентаций. Это действие заключается в выборке областей оконечных блоков вне фильтров нейронов UC2 слоя, составляющих длинные линии, и соотнесении, на основе выбранной области и двух ориентаций, нейронов UBC3 слоя с соответствующими нейронами из UP5 слоя. Смысл данной операции в том, что максимальное срабатывание получается там, где расположена именно часть иллюзорного контура с явным убеждением о том, что ребра и угол проверены срабатываниями нейронов предыдущих типов. Таким образом, при выполнении порогового отсечения нерелевантных областей, а именно тех областей, где отсутствует описанный угловой эффект, если они все еще имеются, на выходе получаются области иллюзорного контура, при складывании которых составляется общий иллюзорный контур фигуры (рис. 7).

$$UI6(y) = \sum_{y \in Y} UP5(y) + UBC3(t),$$

где $t \leftrightarrow$ нейрона y слоя UP5, т. е.

$$t \leftrightarrow \left\{y_1^{(1)}, \dots, y_N^{(1)}\right\}, \left\{y_1^{(2)}, \dots, y_N^{(2)}\right\}$$

пар составляющих из UP4 слоя.

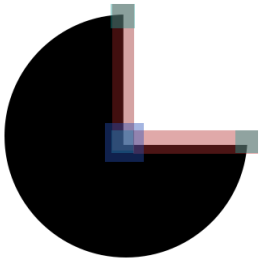

Рис. 7. Рецептивное поле нейрона распознавания контура
Fig. 7. Receptive field of the illusory contour neuron

4. Результаты

Проверка предложенной архитектуры проводилась на разных конфигурациях фигуры Канижа, например квадрат Канижа (рис. 8). На рис. 9 представлены результаты работы нейронов разных слоев при распознавании квадрата Канижа. На рис. 10 и 11 продемонстрирована работа предложенной архитектуры на примере пятиугольника.

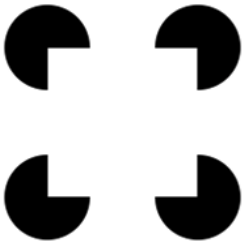

Рис. 8. Пример фигуры Канижа
Fig. 8. Example of a Kanizsa's figure

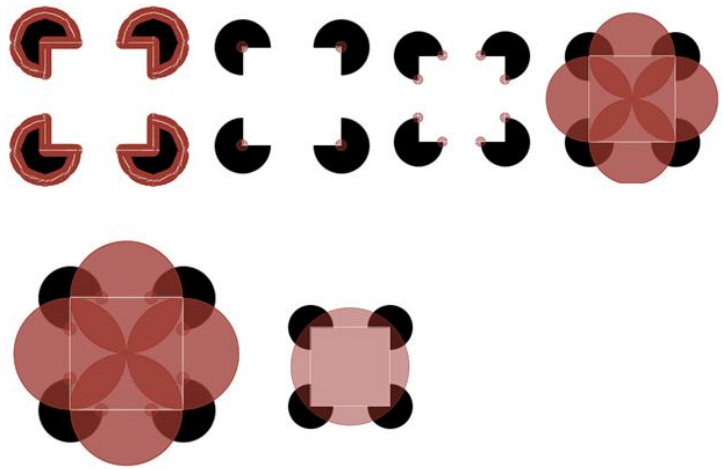

Рис. 9. Результат работы слоев нейронной сети
(верхний ряд, слева направо – края, углы, концы линий, длинные линии;
нижний ряд, слева направо – ребра, иллюзорный контур)

Fig. 9. The result of the neural network layers
(top row, left-to-right – edges, corners, line ends, long lines;
bottom row, left-to-right – edges, illusory contour)

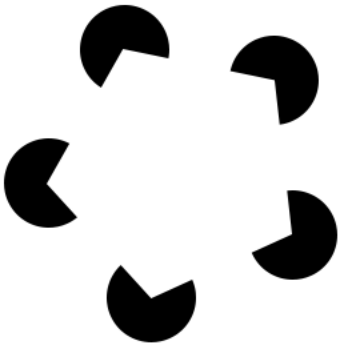

Рис. 10. Пятиугольник Канижа
Fig. 10. Kanizsa's pentagon

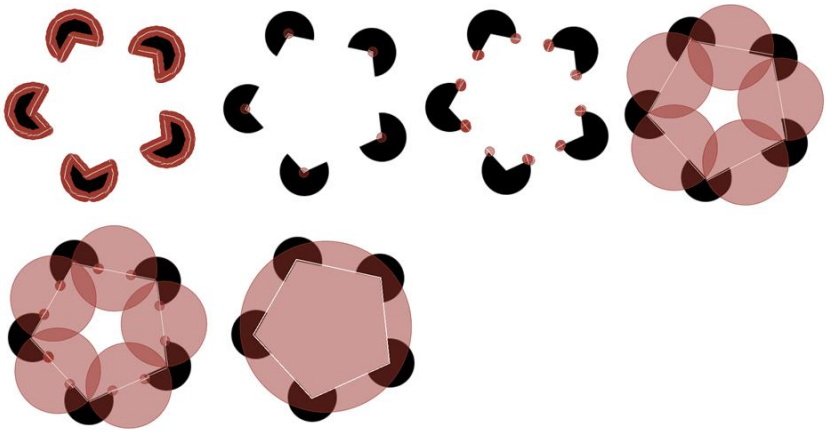

Рис. 11. Результат работы слоев нейронной сети
(верхний ряд, слева направо – края, углы, концы линий, длинные линии;
нижний ряд, слева направо – ребра, иллюзорный контур)

Fig. 11. The result of the neural network layers
(top row, left-to-right – edges, corners, line ends, long lines;
bottom row, left-to-right – edges, illusory contour)

Заключение

Представленная архитектура нейронной сети на биологически-подобных моделях нейронов распознает иллюзорные контуры типа «фигура Канижа» с любым количеством углов и на разных расстояниях. Данный метод может быть применен и в разделении «фигура – фон» на изображениях.

Отличительным достоинство предлагаемого подхода является возможность его переноса на аппаратный уровень без необходимости дообучения и калибровки в процессе эксплуатации. Также добавление других моделей нейронов, например нейрона детекции движения, может позволить выделять движущиеся иллюзорные контуры, что, в свою очередь, может помочь более качественно сегментировать перекрытые изображения.

Список литературы / References

1. **Heydt R.von der, Peterhans E., Baumgartner G.** Illusory Contours and Cortical Neuron Responses. *Science*, 1984, vol. 224, no. 4654, pp. 1260–1262.
2. **Finkel L., Edelman G.** Integration of distributed cortical systems by reentry: a computer simulation of interactive functionally segregated visual areas. *J. Neurosci*., 1989, vol. 9, no. 9, pp. 3188–3208.
3. **Guy G., Medioni G.** Inferring global perceptual contours from local features. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. New York, NY, IEEE Comput. Soc. Press, 1993, pp. 786–787.
4. **Williams, Hanson.** Perceptual completion of occluded surfaces. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94. Seattle, WA, IEEE Comput. Soc. Press, 1994, pp. 104–112.
5. **Gove A., Grossberg S., Mingolla E.** Brightness perception, illusory contours, and corticogeniculate feedback. *Vis Neurosci*., 1995, vol. 12, no. 6, pp. 1027–1052.
6. **Williams L. R., Jacobs D. W.** Stochastic completion fields: a neural model of illusory contour shape and salience. In: Proceedings of IEEE International Conference on Computer Vision. Cambridge, MA, IEEE Comput. Soc. Press, 1995, pp. 408–415.
7. **Yankelovich A., Spitzer H.** Predicting Illusory Contours Without Extracting Special Image Features. *Front. Comput. Neurosci*., 2019, vol. 12, p. 106.
8. **Kugaevskikh A., Sogreshilin A.** Analyzing the Efficiency of Segment Boundary Detection Using Neural Networks. *Optoelectronics Instrumentation and Data Processing*, 2019, vol. 55, no. 4, pp. 414–422. DOI 10.3103/S8756699019040137
9. **Kugaevskikh A.** Bio-Inspired End-Stopped Neuron Model for the Curves Segmentation. In: Proceedings of 2020 International Russian Automation Conference (RusAutoCon). Sochi, IEEE, 2020, pp. 719–724. DOI 10.1109/RusAutoCon49822.2020.9208069
10. **Heitger F. et al.** Simulation of neural contour mechanisms: from simple to end-stopped cells: 5. *Vision Research*, 1992, vol. 32, no. 5, pp. 963–981.

Информация об авторах

Александр Владимирович Кугаевских, кандидат технических наук
Максим Сергеевич Берьянов, магистрант

Information about the Authors

Alexander V. Kugaevskikh, Candidate of Technical Sciences
Maxim S. Berenov, Master's Student

Статья поступила в редакцию 01.02.2022;
одобрена после рецензирования 21.02.2022; *принята к публикации 21.02.2022*
The article was submitted 01.02.2022;
approved after reviewing 21.02.2022; *accepted for publication 21.02.2022*

Научная статья

УДК 550.344.42
DOI 10.25205/1818-7900-2022-20-1-57-66

Метод расчета кинематики волнового фронта цунами в сеточной области

Андрей Гурьевич Марчук

Институт вычислительной математики и математической геофизики
Сибирского отделения Российской академии наук
Новосибирск, Россия

Новосибирский государственный университет
Новосибирск, Россия

mag@omzg.sscc.ru

Аннотация

Предложен новый метод расчета времен вступления волны цунами в узлы прямоугольной расчетной сетки. Метод базируется на построении прямолинейного сегмента фронта внутри ячейки расчетной сетки, основываясь на известных временах прихода волны в углы этой прямоугольной ячейки. Тестирование метода на известных точных решениях для волновых фронтов цунами показало более качественную аппроксимацию волновых фронтов по сравнению с существующим сеточным методом, разработанным автором ранее.

Ключевые слова

волна цунами, кинематика волнового фронта, цифровая сеточная батиметрия, времена вступления цунами

Благодарности

Исследование выполнено в рамках Государственной бюджетной программы с ИВММГ СО РАН (№ 0315-2019-0005)

Для цитирования

Марчук Ан. Г. Метод расчета кинематики волнового фронта цунами в сеточной области // Вестник НГУ. Серия: Информационные технологии. 2022. Т. 20, № 1. С. 57–66. DOI 10.25205/1818-7900-2022-20-1-57-66

Method for Tsunami Wave Kinematics Modeling in a Gridded Area

Andrey G. Marchuk

Institute of Computational Mathematics and Mathematical Geophysics
of the Siberian Branch of the Russian Academy of Sciences
Novosibirsk, Russian Federation

Novosibirsk State University
Novosibirsk, Russian Federation

mag@omzg.sscc.ru

Abstract

The new method for the tsunami travel times calculation to the nodes of rectangular grid was developed and tested. The algorithm is based on constructing of the wave front segment inside the grid cell using tsunami arrival times to the cell corners. Testing the method proposed shows the better quality of the wave-front line approximation as compared to the existing method developed by the author earlier.

© Марчук Ан. Г., 2022

Keywords
tsunami wave, kinematics of the wave front, digital gridded bathymetry, tsunami travel times
Acknowledgements
This work was carried out under state contract with ICMMG SB RAS (no. 0315-2019-0005)
For citation
Marchuk An. G. Method for Tsunami Wave Kinematics Modeling in a Gridded Area. *Vestnik NSU. Series: Information Technologies*, 2022, vol. 20, no. 1, pp. 57–66. (in Russ.) DOI 10.25205/1818-7900-2022-20-1-57-66

Введение

Для службы предупреждения цунами большое значение имеет быстрая и точная оценка времени распространения волны цунами от очага до различных пунктов побережья. Для решения этой кинематической задачи существует несколько методов, основанных как на построении волновых лучей [1–3], так и на непосредственном нахождении волновых фронтов путем численного решения уравнений мелкой воды [4–6]. Для применения одних методов требуется непрерывная функция, отображающая глубину двумерной в плане области. Для других достаточно знать значения глубины в узлах регулярной сетки. Последние более перспективны для практического использования, так как цифровая батиметрия реальных акваторий представлена глубинами на сетках различной детальности. Для использования методов первой группы в таких случаях может быть применена билинейная (или иная) интерполяция значений глубины в узлах сетки на все остальные точки области.

При реализации сеточных методов кинематика волнового фронта внутри ячейки сетки моделируется в предположении линейного характера изменения глубины между точками.

1. Основы кинематики цунами

Волна цунами относится к классу длинных волн, для описания которых используется система дифференциальных уравнений мелкой воды [7]. В случае отсутствия внешних сил, кроме гравитации, эти уравнения имеют вид

$$\begin{aligned} H_t + (uH)_x + (vH)_y &= 0, \\ u_t + uu_x + vu_y + gH_x &= gD_x, \\ v_t + uv_x + vv_y + gH_y &= gD_y, \end{aligned} \tag{1}$$

Здесь $H = D + \eta$ представляет полную толщину водного слоя, η – вертикальное смещение водной поверхности относительно невозмущенного уровня, u и v – компоненты горизонтальной скорости водного потока, D – глубина, g – ускорение силы тяжести. Из уравнений (1) следует, что скорость волнового фронта длинной волны не зависит от ее параметров (длины и высоты) и выражается формулой [7]

$$c = \sqrt{gD}\,. \tag{2}$$

Из той же системы уравнений (1) следует, что точки волнового фронта продвигаются с указанной в формуле Лагранжа (2) скоростью в направлении нормали к линии этого волнового фронта. Здесь следует заметить, что из-за амплитудной нелинейности скорость гребня волны несколько отличается от скорости распространения фронта и составляет

$$c = \sqrt{g(D+\eta)}\,, \tag{3}$$

где η представляет высоту волны цунами. Мы будем численно рассчитывать кинематику именно фронта волны цунами, поэтому в численном алгоритме будет использоваться формула (2). Будем считать, что внутри ячейки сетки изменение глубины происходит по линейному закону. Найдем время пробега волны цунами между двумя точками, расстояние между кото-

рыми равно L, а глубина линейно меняется от величины H_1 до величины H_2. Введем вспомогательную величину – угол наклона дна $tg(\alpha) = |(H_2 - H_1)|/L$. Тогда время пробега вдоль отрезка длиной L запишется в виде

$$T = \int_0^L \frac{dl}{\sqrt{g \cdot (H_1 + l \cdot tg\alpha)}} = \frac{1}{\sqrt{g \cdot tg\alpha}} \int_0^L \left(l + \frac{H_1}{tg\alpha} \right)^{-1/2} d\left(l + \frac{H_1}{tg\alpha} \right) =$$

$$= \frac{2}{\sqrt{g \cdot tg\alpha}} \left(l + \frac{H_1}{tg\alpha} \right)^{1/2} \Bigg|_0^L = \frac{2}{\sqrt{g \cdot tg\alpha}} \cdot \frac{\sqrt{H_2} - \sqrt{H_1}}{\sqrt{tg\alpha}} = \quad (4)$$

$$= \frac{2}{\sqrt{g} \cdot tg\alpha} \cdot \frac{H_2 - H_1}{\sqrt{H_2} + \sqrt{H_1}} = \frac{2L}{\sqrt{gH_2} + \sqrt{gH_1}}.$$

Следовательно, время движения волны цунами между близкими точками внутри ячейки расчетной сетки равно расстоянию между ними, деленному на среднее арифметическое скоростей цунами в этих узлах. Следует заметить, что над неровным дном оптимальной траекторией, вдоль которой возмущение распространяется за наименьшее время, не всегда будет прямая линия. Однако для коротких расстояний разницей во времени между движением по оптимальной траектории и отрезку прямой можно пренебречь.

2. Описание метода

Фронт волны цунами продвигается в направлении внешней нормали к линии этого фронта, которую можно определить как границу раздела между точками акватории, куда возмущение к этому моменту уже пришло, и всеми остальными точками области [8]. В публикации [9] предложен численный метод ортогонального продвижения волнового фронта для расчета кинематики цунами. Но для его реализации требуются значения глубины в любой точке области, которые вычисляются билинейным интерполированием сеточной батиметрии.

В предлагаемом методе направление внешней нормали к линии волнового фонта определяется геометрически на основе времен вступления волны в узлы регулярной сетки, где значения глубины известны. Суть метода состоит в вычислении времени движения волны в рассматриваемый узел сетки от отрезка волнового фронта, который может быть корректно построен на основе вычисленных к этому моменту сеточных данных. Теперь опишем сам метод. В начальный момент во всех узлах расчетной сетки устанавливаем значения времени $t(i, j) = -1$, $i = 1,\ldots,i_{\max}$, $j = 1,\ldots,j_{\max}$. В узлах сетки, находящихся внутри очаговой области, приравниваем нулю значения $t(i, j)$. После этого начинаем поочередно перебирать точки области, где $t(i, j) = -1$, и вычислять в них время прихода туда волны. Понятно, что найти время можно только в тех узлах, где в числе соседних с ним имеются узлы с неотрицательными значениями времени. В качестве примера на рис. 1 приведены узлы расчетной сетки, расположенные внутри круглого источника цунами (где $t(i, j) = 0$), при этом граничащие с узлами, где время прихода туда волны неизвестно ($t(i, j) = -1$).

Вариантов относительного положения тех узлов расчетной сетки, где время прихода туда волны цунами найдено, граничащих с теми, где время еще не найдено, не так много. Возможны только 3 конфигурации (шаблона) относительного расположения таких узлов без учета их ориентации на плоскости. Они приведены на рис. 2.

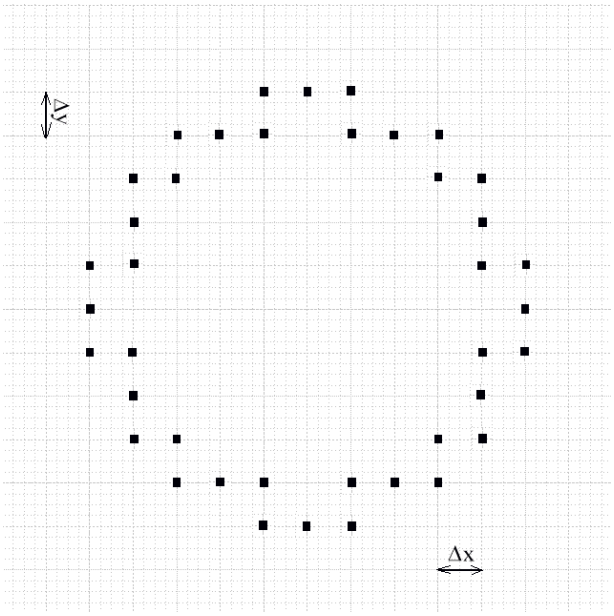

Рис. 1. Расположение узлов прямоугольной сетки, представляющих начальный круговой волновой фронт. Длина пространственных шагов вдоль оси абсцисс и ординат равны соответственно Δx и Δy

Fig. 1. The location of the rectangular grid nodes representing the initial circular wave front. The length of the spatial steps along the abscissa and the ordinate axes are Δx and Δy, respectively

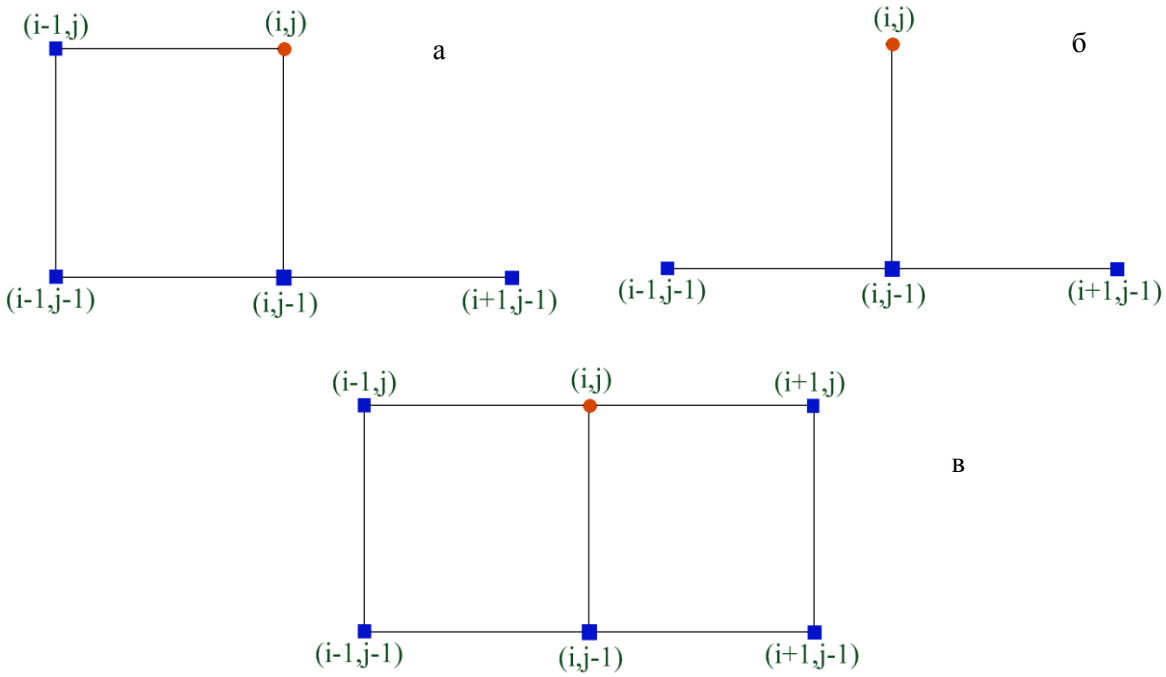

Рис. 2. Шаблоны для расчета времени прихода в точку (i, j):
а – шаблон № 1; *б* – шаблон № 2; *в* – шаблон № 3

Fig. 2. The stencils for the computation of the wave arrival time to the grid node (i, j)
a – stencil no. 1; *b* – stencil no. 2; *c* – stencil no. 3

На этих рисунках узлы сетки, где время прихода туда цунами уже известно, изображены в форме квадратиков, а узлы сетки, где это время требуется определить, имеют форму кружков. Конфигурации расположения узлов, приведенные на рис. 2, могут быть ориентированы

иным образом. В шаблоне № 1 (рис. 2, *а*) узел сетки с индексами ($i + 1, j - 1$) может отсутствовать. Далее опишем алгоритм расчета времени прихода цунами в узел расчетной сетки при конфигурации, приведенной на рис. 2, *а*. Рисунок 5 показывает схему этого расчета, основанную на простых геометрических построениях.

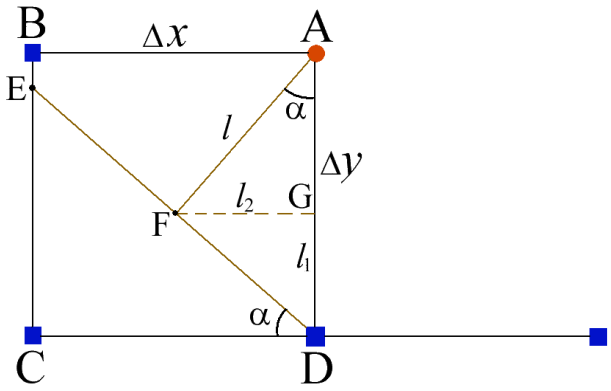

Рис. 3. Схема вычисления времени прихода в точку A на основе известных времен прихода волны в точки B, C и D. Во всех углах ячейки сетки глубины заданы
Fig. 3. The scheme for calculating the arrival time at the point A based on the known arrival times of the wave at the points B, C and D. At the angles of the grid cell depths are given

Рассмотрим ячейку расчетной сетки, где требуется определить время вступления волны T_A в точку A при известных временах прихода цунами T_B, T_C и T_D в остальные три угла ячейки B, C и D (рис. 2, *в*). Значения глубины H_A, H_B, H_C и H_D известны во всех углах ячейки. Пусть для определенности $T_B > T_D$. В этом случае точка пересечения отрезка волнового фронта, соответствующего моменту времени T_D, и отрезка BC (вместе с этим и угол α) находится с использованием формулы (4). Принимая во внимание то, что скорость движения точки пересечения фронта с отрезком BC больше скорости волнового фронта в 1/cos(α) раз, величина cos(α) вычисляется по формуле

$$\cos(\alpha) = \frac{(T_B - T_C)\left(\sqrt{gH_C} + \sqrt{gH_B}\right)}{2\Delta y}. \tag{5}$$

Далее, легко найти расстояние l от точки A до отрезка волнового фронта DE:

$$l = \Delta y \cdot \cos(\alpha) \tag{6}$$

Теперь для вычисления времени вступления цунами в точку A достаточно определить глубину в точке F, определяемую ее координатами. Эти координаты X_F и Y_F привязаны к координатам X_D и Y_D точки D по формулам

$$X_F = X_D - l_2 = X_D - l \cdot \sin(\alpha), \tag{7}$$

$$Y_F = Y_D + l_1 = Y_D + l \cdot \cos(\alpha). \tag{8}$$

Глубина в точке F вычисляется билинейной интерполяцией с использованием координат точки F внутри ячейки расчетной сетки и значений глубины в углах рассмотренной ячейки.

$$H_F = \frac{H_C\left(X_D - X_F\right) + H_D \cdot X_F}{\Delta x} \cdot \frac{Y_B - Y_F}{\Delta y} + \frac{H_B\left(X_D - X_F\right) + H_A \cdot X_F}{\Delta x} \cdot \frac{Y_F}{\Delta y} \qquad (9)$$

В результате искомое время вступления волны в точку A выразится в виде

$$T_A = T_D + \frac{2l}{\sqrt{gH_A} + \sqrt{gH_F}}, \qquad (10)$$

где l и глубина H_F вычисляются по формулам (6) и (9) на основе известных значений глубины во всех углах ячейки и значений времени прихода цунами в три из них (в точки B, C и D). Аналогично с помощью геометрических построений вычисляются времена вступления волны в узлы сетки для других конфигураций узлов с известными и неизвестными значениями времени (рис. 2, *б*, *в*).

3. Тестирование метода

Протестируем сначала предложенный метод для нахождения последовательных положений волнового фронта, имеющего наклон в 45° к оси абсцисс (как и к оси ординат). Глубина во всех узлах расчетной области размером 2000 × 1000 узлов одинакова и равна 1000 м. Пространственный шаг сетки равен 1000 м в обоих направлениях. В начальный момент прямолинейный волновой фронт соединял левый верхний угол области с центральной точкой нижней границы (рис. 4).

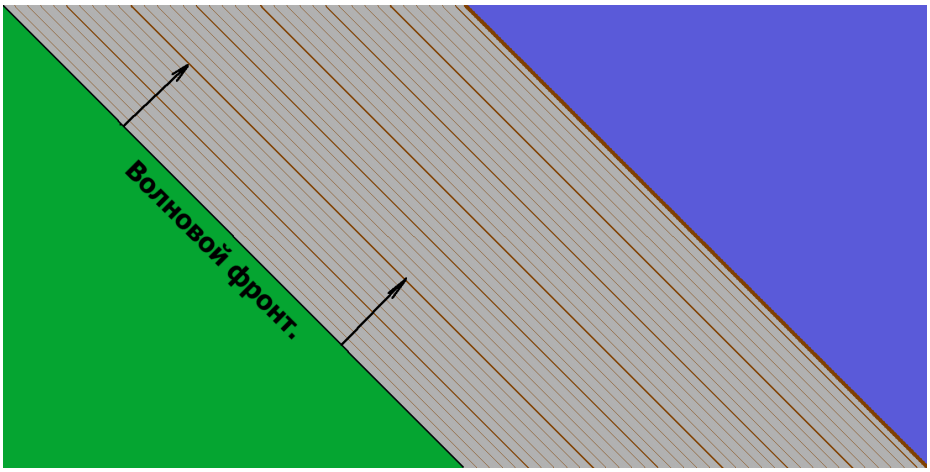

Рис. 4. Последовательные положения волнового фронта в области с постоянной глубиной, полученные описанным методом (зона начального возмущения окрашена зеленым)

Fig. 4. Sequential positions of the wave front when calculating the kinematics of the oblique front in an area with a constant depth (the zone of initial disturbance where $t(i, j) = 0$ is colored by green)

Из рис. 4 видно, что предлагаемый в статье метод идеально моделирует кинематику волнового фронта в поставленной задаче. Далее протестируем описанный алгоритм на известном точном решении для волнового фронта над параболическим рельефом дна [10]. Рассмотрим следующую задачу: в сеточной области размером 1000 × 1000 узлов глубина возрастает от нижней границы к верхней по формуле

$$H_{ij} = 0.001 \cdot (j + 100)^2, \; \text{i=1,…,1000, j=1,…, 1000.} \qquad (11)$$

Длина шага сетки в обоих направлениях одинакова и равна 100 м. Начальное положение волнового фронта совпадает с левой границей расчетной области. Затем начинается повторяющийся перебор всех узлов расчетной сетки с вычислением времен вступления цунами во все узлы правее левой границы. На рис. 5 визуализированы положения волнового фронта через каждые 200 с, построенные на основе результатов расчета кинематики цунами описанным методом.

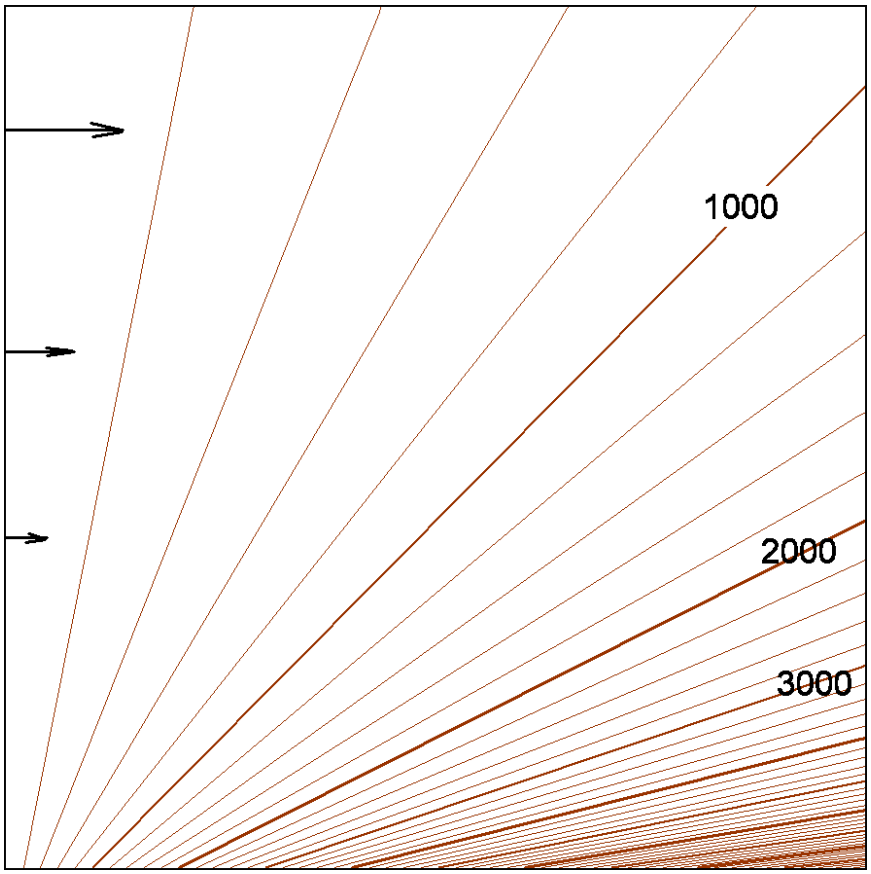

Рис. 5. Положения фронта цунами над параболическим рельефом дна через каждые 200 с
Fig. 5. Tsunami wave front positions every 200 sec above the parabolic bottom topography

Из рис. 5 видно, что волновой фронт остается прямолинейным на протяжении всего времени распространения цунами. Это идеально соответствует точному решению для такого модельного распределения глубин в расчетной области [10].

Еще одним тестом для проверки корректности метода является расчет кинематики изначально кругового волнового фронта в области с постоянной глубиной. Теоретически в этом случае линия волнового фронта в любой момент времени будет иметь форму окружности. Для тестирования предложенного метода в центре расчетной области 1000 × 1000 узлов располагается очаг цунами, ограниченный окружностью с радиусом 5000 м. Длина шага сетки составляет 100 м в обоих направлениях, а глубина во всей области постоянна и равна 1000 м. Описанным алгоритмом с учетом трех возможных конфигураций узлов сетки с известными и неизвестными временами вступления туда волны (см. рис. 2) производятся многократные переборы всех расчетных узлов и рассчитываются времена прихода цунами в узлы сетки. Для корректной работы алгоритма направления перебора узлов меняются на противополож-

ные через раз. На рис. 6 визуализированы изолинии поля времен вступления волны в узлы сетки.

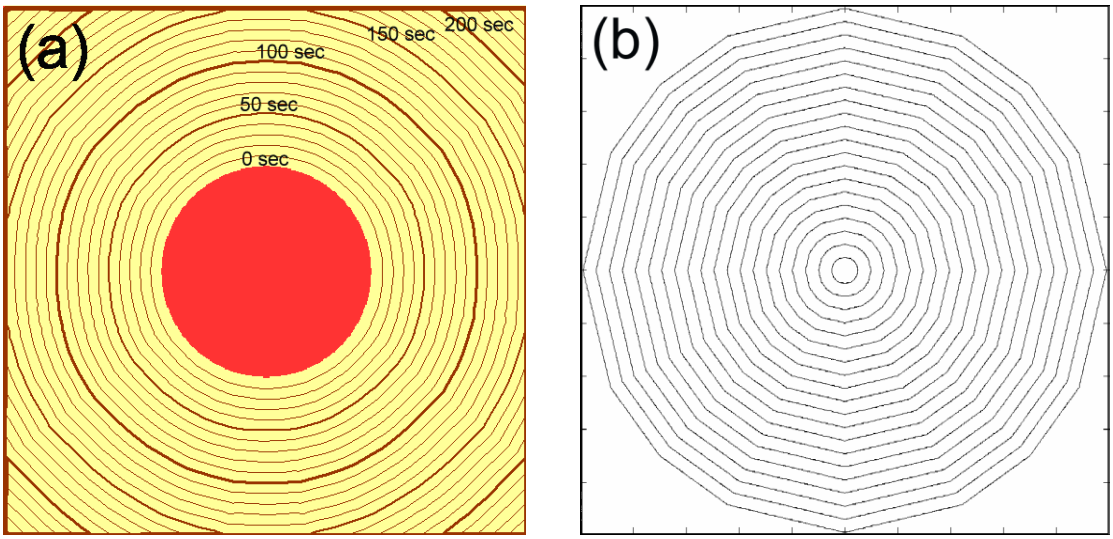

Рис. 6. Последовательные положения фронта волны от круглого источника в области с постоянной глубиной (1000 м), найденные описанным методом (*a*) и последовательные положения волнового фронта от точечного источника, построенные сеточным методом (*b*), реализованным автором в 1988 г. [8]

Fig. 6. The isolines of the calculated wave arrival times from a round source to the nodes of the calculation grid when using the method proposed (*a*) and as a result of the calculation (*b*) using the algorithm developed by author in 1988 [8]

Из рис. 6, *a* видно, что форма вычисленного волнового фронта в любой момент времени близка к окружности. Это еще раз подтверждает корректность предложенного метода. Для сравнения на рис. 6, *b* результаты тестирования той же задачи с использованием сеточного метода, основанного на принципе Гюйгенса [8], представлены в том же виде. Можно видеть, что с помощью алгоритма, описанного в этой статье, полученные изохроны ближе к точному решению, чем положения волнового фронта, полученные сеточным методом [8]. Количественная разница (в секундах) между полученными временами вступления волны в узлы сетки и точным решением визуализирована на рис. 7 в виде изолиний и цветовой легенды.

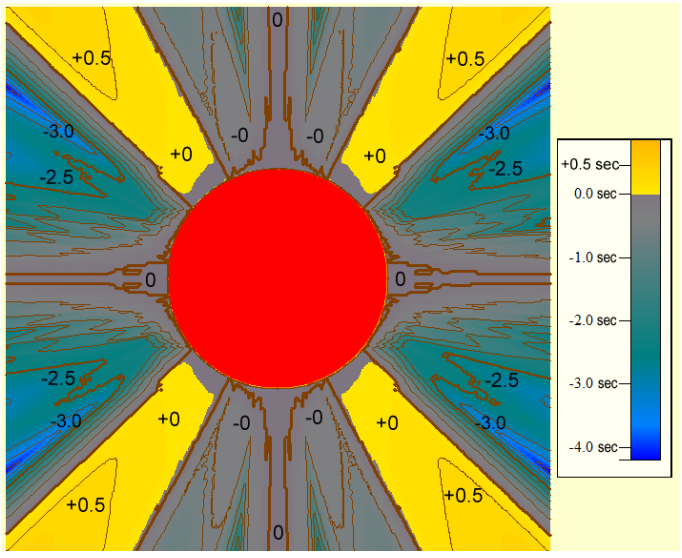

Рис. 7. Распределение (в виде изолиний и цветовой палитры) отклонения от точного решения вычисленных времен прихода в узлы регулярной сетки цунами, генерированного источником круглой формы

Fig. 7. Distribution (in the form of isolines and a color palette) of the deviation from the exact solution of the calculated travel times at the nodes of the regular grid when tsunami generated by the circular source

Из рис. 7 видно, что в некоторых направлениях разница между вычисленным и точным решением кинематической задачи близка к нулю (< 1 с) вплоть до границ расчетной области. Но имеются 4 азимута, где эта разница постепенно нарастает при удалении от источника, но не превышает 2 % от времени движения фронта волны в этих направлениях. Это можно объяснить неточным вычислением направлений сегментов волнового фронта при некоторых конфигурациях узлов с вычисленными и неизвестными временами вступления туда цунами.

Заключение

Разработан новый метод нахождения времени вступления волны в узлы прямоугольной расчетной сетки, где глубина задана. Он основан на нахождении направления внешней нормали к сегменту волнового фронта и продвижении волнового фронта в этом направлении. Тестирование метода на нескольких тестовых заданиях показало его достоверность и эффективность. Относительные ошибки в вычисленных временах вступления в узлы прямоугольной сетки не превышают 1–2 %.

Список литературы

1. **Shokin Yu. I., Chubarov L. B., Novikov V. A., Sudakov A. N.** Calculations of tsunami travel time charts in the Pacific Ocean – Models, Algorithms, Techniques, Results. *Science of Tsunami Hazards*, 1987, vol. 5, no. 2, pp. 85–113.
2. **Kenji Satake.** Effects of bathymetry on tsunami propagation: Application of ray tracing to tsunamis. *Pure and Applied Geophysics*, 1988, vol. 126, iss. 1, pp. 27–36.
3. **Кремлев А. Н.** Преломление плоской волны на выпуклом и вогнутом углах в приближении геометрической акустики // Сиб. журн. вычисл. математики. 2017. Т. 20, № 3. С. 251–271.
4. **Titov V. V., Gonzalez F.** Implementation and Testing of the Method of Splitting Tsunami (MOST). Washington DC, 1997. (Technical Memorandum / National Oceanic and Atmospheric Administration; ERL PMEL-112)
5. **Imamura F.** Tsunami numerical simulation with the staggered leap-frog scheme (numerical code of TUNAMI-N1 and N2). In: Disaster Control Research Center. Tohoku University, 1995, 33 p.
6. **Лаврентьев М. М., Лысаков К. Ф., Марчук Ан. Г., Облаухов К. К.** Ускорение расчетов распространения волны цунами с использованием FPGA // Успехи кибернетики. 2020. Т. 1, № 4. С. 42–54.
7. **Стокер Дж. Дж.** Волны на воде. М.: ИЛ, 1959. 617 с.
8. **Марчук Ан. Г.** Численные методы расчета кинематики волн цунами // Математические проблемы геофизики: Численные исследования геофизических задач. Новосибирск, 1988. С. 69–90.
9. **Marchuk An. G., Vasiliev G. S.** The fast method for a rough tsunami amplitude estimation // *Bulletin of the Novosibirsk Computing Center. Series: Math. Model. in Geoph.*, 2014, no. 17, pp. 21–34.
10. **Марчук Ан. Г.** Оценка высоты цунами, распространяющейся над параболическим дном, в лучевом приближении // Сиб. журн. вычисл. математики. 2017. Т. 20, № 1. С. 23–35.

References

1. **Shokin Yu. I., Chubarov L. B., Novikov V. A., Sudakov A. N.** Calculations of tsunami travel time charts in the Pacific Ocean – Models, Algorithms, Techniques, Results. *Science of Tsunami Hazards*, 1987, vol. 5, no. 2, pp. 85–113.

2. **Kenji Satake.** Effects of bathymetry on tsunami propagation: Application of ray tracing to tsunamis. *Pure and Applied Geophysics*, 1988, vol. 126, iss. 1, pp. 27–36.
3. **Kremlev A. N.** Prelomlenie ploskoy volny na vypuklom i vognutom uglakh v priblizhenii geometricheskoy akustiki. *Sib. zhurn. vychisl. matematiki*, 2017, vol. 20, no. 3, pp. 251–271. (in Russ.)
4. **Titov V. V., Gonzalez F.** Implementation and Testing of the Method of Splitting Tsunami (MOST). Washington DC, 1997. (Technical Memorandum / National Oceanic and Atmospheric Administration; ERL PMEL-112)
5. **Imamura F.** Tsunami numerical simulation with the staggered leap-frog scheme (numerical code of TUNAMI-N1 and N2). In: Disaster Control Research Center. Tohoku University, 1995, 33 p.
6. **Lavrentev M. M., Lysakov K. F., Marchuk An. G., Oblaukhov K. K.** Uskorenie raschetov rasprostraneniya volny tsunami s ispol'zovaniem FPGA. *Uspekhi kibernetiki*, 2020, vol. 1, no. 4, pp. 42–54. (in Russ.)
7. **Stoker J. J. Volny na vode**. Moscow, 1959, 617 p. (in Russ.)
8. **Marchuk An. G.** Chislennye metody rascheta kinematiki voln tsunami. In: Matematicheskie problemy geofiziki: Chislennye issledovaniya geofizicheskikh zadach. Novosibirsk, 1988, pp. 69–90. (in Russ.)
9. **Marchuk An. G., Vasiliev G. S.** The fast method for a rough tsunami amplitude estimation // *Bulletin of the Novosibirsk Computing Center. Series: Math. Model. in Geoph.*, 2014, no. 17, pp. 21–34.
10. **Marchuk An. G.** Otsenka vysoty tsunami, rasprostranyayushcheysya nad parabolicheskim dnom, v luchevom priblizhenii. *Sib. zhurn. vychisl. matematiki*, 2017, vol. 20, no. 1, pp. 23–35. (in Russ.)

Информация об авторе

Андрей Гурьевич Марчук, доктор физико-математических наук
SPIN 4054-6578
WoS Researcher ID S-9502-2017
Scopus Author ID 57023485400

Information about the Author

Andrey G. Marchuk, Doctor of Sciences (Physics and Mathematics)
SPIN 4054-6578
WoS Researcher ID S-9502-2017
Scopus Author ID 57023485400

Статья поступила в редакцию 11.12.2021;
одобрена после рецензирования 01.02.2022; *принята к публикации 01.02.2022*
The article was submitted 11.12.2021;
approved after reviewing 01.02.2022; *accepted for publication 01.02.2022*

Научная статья

УДК 658.51
DOI 10.25205/1818-7900-2022-20-1-67-80

Разработка автоматизированной системы динамического картирования потока создания ценности

Полина Андреевна Русских [1]
Денис Владимирович Капулин [2]
Олег Владимирович Дрозд [3]
Сергей Юрьевич Смоглюк [4]

[1–4] Сибирский федеральный университет
Красноярск, Россия

[1] Prusskikh@sfu-kras.ru, https://orcid.org/0000-0003-2858-2893
[2] Dkapulin@sfu-kras.ru, https://orcid.org/0000-0002-4260-1408
[3] Odrozd@sfu-kras.ru, https://orcid.org/0000-0001-7374-253X
[4] xor-s@yandex.ru

Аннотация

Существующие тенденции развития производства диктуют необходимость повышения конкурентоспособности путем преодоления разрыва между организацией производства и цифровыми технологиями. Чтобы повысить эффективность производства, внедряются стратегии бережливого производства, которые сосредоточены на выявлении и минимизации потерь и их устранения. При этом возможности цифровой трансформации позволяют осуществлять мониторинг производственных процессов в режиме реального времени. Таким образом можно использовать инструменты бережливого производства, такие как карта потока создания ценности для эффективно фиксирования процессов и прогнозирования производственной ситуации в динамическом режиме. В разработанной автоматизированной системе динамического картирования потока создания ценности реализован принцип имитационного моделирования. При помощи моделирования можно изучить альтернативы улучшения процесса и влияние предложенных изменений до реализации. Динамическое моделирование карты потока создания ценности позволяет разработать систему автоматизированного управления с определения оптимальных параметров и режимов функционирования производственного процесса. Объектом моделирования является карта потока создания ценностей, отображающая этапы перемещения потоков материалов, деталей, сборочных единиц и информации. Чтобы исследовать параметры производственного процесса и внедрить принципы бережливого производства, было реализовано динамическое отображение потока создания ценности текущего состояния и моделирование будущего состояния.

Ключевые слова

карта потока создания ценности, бережливое производство, имитационное моделирование, автоматизированная система

Благодарности

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 20-07-00226

Для цитирования

Русских П. А., *Капулин Д. В.*, *Дрозд О. В.*, *Смоглюк С. Ю.* Разработка автоматизированной системы динамического картирования потока создания ценности // Вестник НГУ. Серия: Информационные технологии. 2022. Т. 20, № 1. С. 67–80. DOI 10.25205/1818-7900-2022-20-1-67-80

© Русских П. А., Капулин Д. В., Дрозд О. В., Смоглюк С. Ю., 2022

Development of an Automated System for Dynamic Mapping of the Value Stream

Polina A. Russkikh [1], **Denis V. Kapulin** [2]
Oleg V. Drozd [3], **Sergey Yu. Smogluk** [4]

[1–4] Siberian Federal University
Krasnoyarsk, Russian Federation

[1] Prusskikh@sfu-kras.ru, https://orcid.org/0000-0003-2858-2893
[2] Dkapulin@sfu-kras.ru, https://orcid.org/0000-0002-4260-1408
[3] Odrozd@sfu-kras.ru, https://orcid.org/0000-0001-7374-253X
[4] xor-s@yandex.ru

Abstract

The current trends in the development of production dictate the need to increase competitiveness by bridging the gap between the organization of production and digital technologies. To improve production efficiency, lean manufacturing strategies are being implemented that focus on identifying and minimizing waste and eliminating them. At the same time, the capabilities of digital transformation allow monitoring of production processes in real time. Thus, it is possible to use lean manufacturing tools such as a value stream map to effectively capture processes and predict the production situation in a dynamic mode. In the developed automated system for dynamic mapping of the value stream, the principle of simulation is implemented. With the help of simulations, it is possible to study the alternatives for improving the process and the impact of the proposed changes before implementation. Dynamic modeling of the value stream map allows you to develop an automated control system with the definition of optimal parameters and modes of operation of the production process. The object of modeling is a value stream map that displays the stages of movement of flows of materials, parts, assembly units and information. To investigate the parameters of the production process and implement the principles of lean manufacturing, dynamic mapping of the current state value stream and modeling of the future state were implemented.

Keywords

value stream map, lean manufacturing, simulation, automated system

Acknowledgements

The reported study was funded by RFBR, project no. 20-07-00226

For citation

Russkikh P. A., Kapulin D. V., Drozd O. V., Smoglyuk S. Yu. Development of an Automated System for Dynamic Mapping of the Value Stream. *Vestnik NSU. Series: Information Technologies*, 2022, vol. 20, no. 1, pp. 67–80. (in Russ.) DOI 10.25205/1818-7900-2022-20-1-67-80

Введение

Очевидно, что эра массового производства сменяется эрой производства под заказ [1]. Ключевым фактором, позволяющим разрабатывать продукцию под требования клиента, при этом сохраняя низкую стоимость и высокое качество товара, является короткий цикл технологической подготовки производства. Раньше, сразу после выхода новой продукции можно было наблюдать долговременный устойчивый рост объемов производства, затем следовал спад. В настоящее время объемы производства гораздо быстрее достигают количественного пика, но и сокращение производственного объема происходит резко, чаще всего из-за модификации продукта [2]. Новый тип жизненных циклов и растущее число моделей и вариантов продукции повышает сложность производственных процессов. Надежная поставка индивидуальных продуктов имеет наивысший приоритет на глобально распределенных рынках, и этот приоритет все больше определяет потребность в развитии продуктов, процессов и производственных мощностей [3; 4]. На физическом уровне производственные и сборочные системы должны быть реконфигурируемыми, а завод с его технической инфраструктурой, включая здание, должен быть трансформируемым. Логический уровень требует систем планирования, способных реагировать на изменения в дизайне продукта [5]. Планирование и контроль производства должны реагировать на изменения в объеме продукции, смешивать или реконфигурировать технологические планы. Изменение объемов производства и но-

менклатуры комплектующих изделий приводит к постоянным корректировкам производственного плана, использованию «ручных» методов управления, влечет за собой неравномерную и малоэффективную загрузку мощностей, нерациональное использование ресурсов. В этом контексте оптимизация использования ресурсов является актуальной проблемой для производства. В то время как заводы, производящие стандартизированную продукцию большими объемами, удовлетворены существующими методами планирования, для мелкосерийного позаказного производства с большим числом номенклатуры изделий не существует достаточно эффективных методов [6].

Для повышения эффективности производства появились два независимых подхода: стратегии бережливого производства, ориентированные на выявление и минимизацию действий, не создающих добавленной стоимости, и подход к производственному планированию MRP II, на основе которого разработаны информационные инструменты для планирования и контроля деятельности [7]. Существующие автоматизированные системы для планирования и поддержки управленческих решений, такие как ERP, MES, APS системы для приборостроительных предприятий, работающих позаказно и мелкой серией, не обеспечивают адаптивности. Эти системы, помимо своей высокой стоимости, нуждаются в специальной доработке и сопровождении под нужды конкретного предприятия. Традиционно карта потока создания ценности используется для быстрого анализа потоков продукции через производственную систему, от сырья до доставки. Любое производство выполняет последовательность шагов, чтобы предоставить клиентам продукт. Поток создания ценности можно рассматривать как поток материалов или информации, необходимый для доставки товара или услуги от заказа до доставки.

Классический подход разработки карты потока создания ценности не учитывает динамический аспект операций и ассортимент продукции, которые могут быть одновременно на производстве. Он применим только для визуализации производственных линий с одним продуктом или семейством продуктов, и его трудно применить для сложных производств с множественными условиями потока материалов и продуктов [8]. Отсутствие автоматизации в создании карты потока создания ценности делает процесс визуализации утомительным и трудоемким, когда эксперт должен провести несколько «прогулок» по цеху и потратить много времени на анализ [9]. Рабочие места современных производств обладают высокой степенью автоматизации и в целом подходят для обработки большего количества продуктов. Логистические процессы продуктовых линеек отличаются друг от друга, поэтому мы можем получить очень сложную систему материальных и информационных потоков (например, при производство мелкосерийных компонентов). Соответственно существует потребность в представлении метода с логикой для автоматизации процесса создания и анализа диаграмм, позволяющих повышать эффективность планирования производственных процессов. Отображение динамического потока создания ценности основано на принципе имитационного моделирования, поэтому карта текущего и будущего состояния создается с помощью симуляции.

Моделирование для динамической карты потока создания ценности

Одним из главных ограничений классического подхода к картированию потока создания ценности является его «ручной» характер, создающий статическую модель, что делает невозможным оценку состояния производства в динамике [10]. Имитационное моделирование способно улучшить карту потока создания ценности, предоставляя информацию о характере производственного процесса, таким образом преодолевается статичность. Использование моделирования предоставляет возможность изучить различные варианты улучшения процесса и влияние предложенных изменений до реализации.

Наиболее частым методом моделирования, используемым для описания производственных процессов, является моделирование дискретных событий, т. е. численное решение сис-

тем массового обслуживания [11]. Система массового обслуживания представляет собой объект, содержащий один или несколько приборов, обслуживающий заявки, поступающие в систему и накопитель, в котором находятся заявки, образующие очередь и ожидающие обслуживания. Очереди ожидания существуют почти во всех промышленных и производственных процессах. Во всех таких системах очередей есть объекты, которые должны повторно обрабатываться другим объектом. Очереди характеризуются рядом свойств, которые представляют взаимосвязи между объектами, участвующими в системе очередей. Процесс поступления, продолжительность обслуживания и дисциплина очереди являются одними из наиболее важных свойств очереди. В дополнение к этим основным свойствам, количество серверов, емкость очереди и совокупность обслуживаемых объектов являются некоторыми другими свойствами системы очередей.

Дисциплина очереди определяется как набор правил на основе определенного атрибута сущностей. Он определяет порядок, по которому элементы в очереди получают услугу. Выбор дисциплины очереди и применяемых правил может существенно повлиять на количество объектов, ожидающих в очереди, среднее время ожидания и эффективность средства обслуживания. Наиболее распространенная дисциплина очереди – «первым пришел – первым вышел», или FIFO, при которой клиенты в очереди обслуживаются в соответствии с хронологическим порядком прибытия. Хотя FIFO уже давно используется в качестве дисциплины очереди по умолчанию при моделировании систем очередей, во многих сценариях одинаково вероятно, что клиенты будут обслуживаться в соответствии с другими шаблонами. Например, дисциплина «последний пришел – первый ушел», или LIFO, может иметь место в ситуациях, когда куча или стопка клиентов (например, сырье, сборные бетонные сегменты, стальные секции) ожидают обработки сервером [12].

В системе массового обслуживания процесс поступления может быть определен последовательностью времени между прибытиями, которые являются независимыми и идентично распределенными в качестве упрощающего предположения, чтобы соответствовать распределениям вероятностей с фиксированными параметрами. Случайность, связанная с соперничающими событиями, позволяет легко охарактеризовать время между встречами с помощью распределения вероятностей. Сервисный объект может иметь разное количество каналов и фаз. При наличии более одного сервера каналы относятся к доступным маршрутам, по которым клиенты могут пройти после ожидания в очереди, чтобы добраться до объекта обслуживания.

Представим производство в виде системы массового обслуживания (рис. 1). Для производства существует набор заказов, которые должны быть произведены. Учитывая тот факт, что планово-учетной единицей планирования мелосерийного многономенклатурного производства является заказ, для моделирования работы производства в качестве заявок на обслуживание используется набор операций, необходимых к выполнению. Операции, необходимые к выполнению, получают из разузлования заказа, соответственно, интенсивность поступления операций, необходимых к выполнению рабочими местами, не является существенным показателем. Все операции, необходимые для выполнения заказа, появляются в очереди на обслуживание, как только заказ поставлен на выполнение. Сделаем допущение, что длина очереди не ограничена, система может содержать сколько угодно большое число заявок, и подразумевается политика приоритетов FIFO.

Система массового обслуживания характеризуется числом каналов обслуживания n, длительностью обслуживания $t_{об}$ одной заявки и пропускной способностью μ – числом заявок, которое может обслужить один канал (или n каналов) в единицу времени. Отношение

$$\rho = \frac{\lambda}{\mu} = \frac{t_{об}}{t_{и}} \quad (1)$$

называется коэффициентом использования системы, или приведенной плотностью потока.

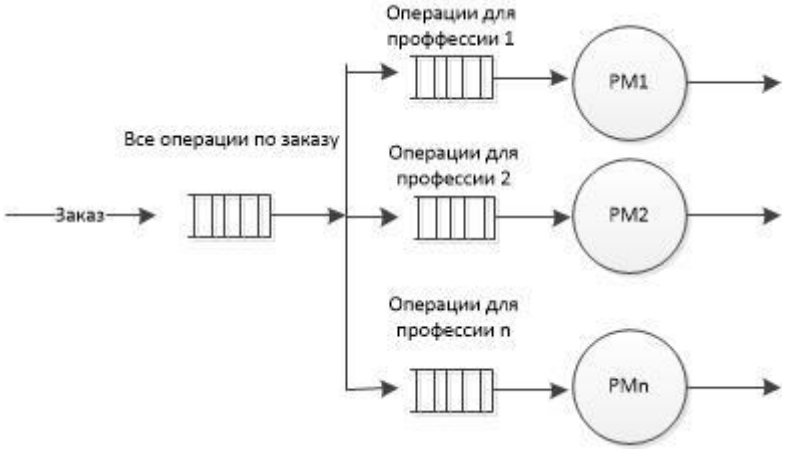

Рис. 1. Модель производства в виде СМО
Fig. 1. Model of production in the form of a queuing system

Показатели работы СМО: вероятность того, что система свободна (2); вероятность того, что заявка окажется в очереди (3); длина очереди (4); среднее число заявок в системе (5); среднее время пребывания заявок в очереди (6); среднее время пребывания заявки в системе (7), – рассчитываются по следующим формулам:

$$p_0 = \left(\sum_{i=0}^{n} \frac{\rho^i}{i!} + \frac{\rho^{n+1}}{n!(n-\rho)}\right)^{-1}, \quad (2)$$

$$P_q = \frac{\rho^{n+1}}{n!(n-\rho)} p_0, \quad (3)$$

$$L_q = \frac{\rho^{n+1} p_0}{nn!(1-\frac{\rho}{n})^2}, \quad (4)$$

$$L_s = L_q + \rho, \quad (5)$$

$$T_q = \frac{L_q}{\lambda}, \quad (6)$$

$$T_s = \frac{L_s}{\lambda}. \quad 7)$$

Заказы в имитационную модель (рис. 2) поступают в заданный диапазон времени, например каждые 3–5 часов, ожидаемый диапазон – 3–5 заказов. Согласно разработанной архитектуре из базы данных в имитационный модуль поступают данные о технологических операциях, необходимых для изготовления изделия. Заказы на все изделия представляют собой набор операций, необходимых к выполнению для реализации производственного плана. Данный документ содержит информацию о длительности работ, оснастке, типе рабочих, нормах времени по каждой операции.

Источником поступления заказа в систему служит блок Sourse. В нем задаются условия интенсивности поступления заказов в систему. Рабочие места представлены блоком Service, который реализует очередь операций к рабочему месту, захват необходимого ресурса и его освобождения при выполнении операции. Очереди задаются блоками Queue. Имитационная модель содержит источник заказов, куда поступают заказы клиентов, рабочие места по необходимому типу операции, очереди, в которых заказы будут ожидать, если пропускная способность соответствующих последующих шагов окажется недостаточной, и узел-приемник в качестве завершения процесса.

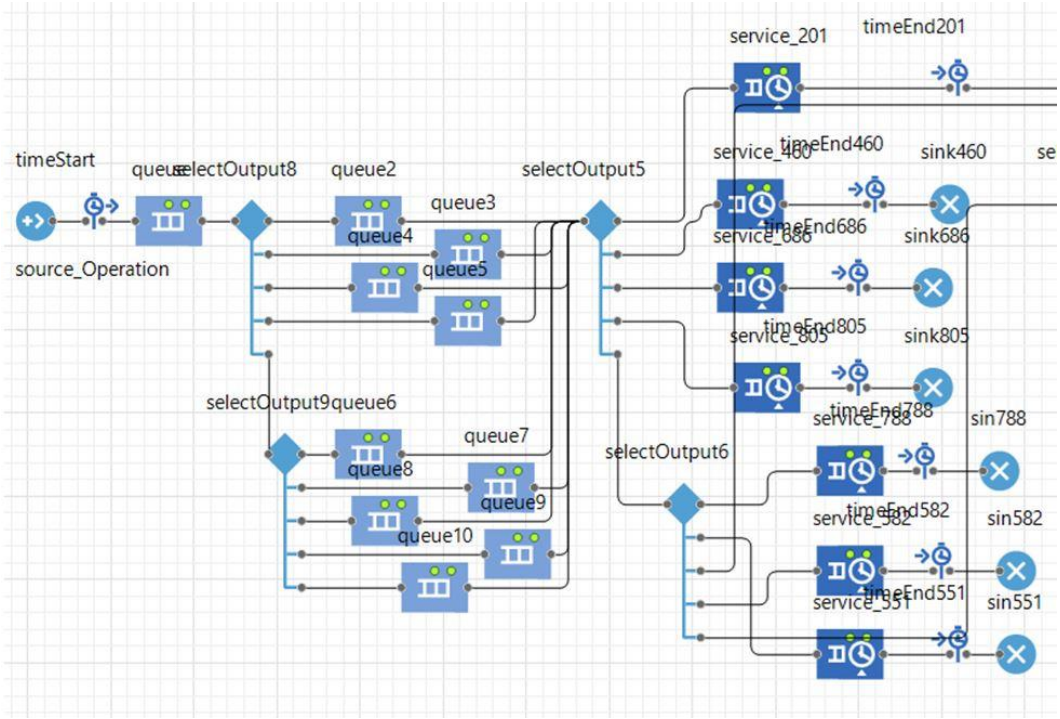

Рис. 2. Имитационная модель производства
Fig. 2. Production simulation model

Метод формирования динамической карты потока создания ценности

Сегодня многие производства используют философию «бережливого внедрения», направленную на сокращение потерь за счет устранения неэффективности в потоке создания ценности [13]. Потери можно определить как любую деятельность, которая не добавляет ценности конечному продукту. В производственной среде потери могут включать, например, перепроизводство, лишние движения (например, оператора или машины), время ожидания (например, оператора или машины), транспортировка, ненужная обработка, избыточный инвентарь и переделка. Чтобы оставаться конкурентоспособными на мировом рынке, предприятия разработали инструменты планирования и коммуникации для повышения эффективности и сокращения потерь [14]. Основная философия внедрения бережливого производства в управление технологическим процессом заключается в устранении неэффективности и улучшении экономических показателей, что достигается за счет сосредоточения внимания на определении сроков производства, которые соответствуют или превышают требования заказчика. Поток создания ценности – это шаги, которые бизнес должен предпринять, чтобы предоставить клиенту желаемые товары или услуги.

Типичный поток создания ценности начинается с заказа клиента и заканчивается доставкой. Карта потока создания ценности (КПСЦ) – это инструмент, который наглядно иллюстрирует поток материалов и информации, проходящий через поток создания ценности [15]. КПСЦ может отражать текущее или будущее состояние производства. КПСЦ – это бумажная ручная процедура с ограниченным количеством наблюдений; следовательно, уровень точности ограничен. КПСЦ создается исходя из требований заказчика и разрабатывается в соответствии с этапами технологического процесса продукта [16]. Основные потоки в КПСЦ – это информационный поток и материальный поток. Схема процесса определяет параметры, которые представляют характеристики процесса.

На рис. 3 представлены операции для формирования динамической карты потока создания ценности.

Рис. 3. Алгоритм формирования динамической карты потока создания ценности
Fig.3. Algorithm for generating a dynamic value stream map

На первом шаге при запуске моделирования через интерфейс пользователя определяется заказ или группа заказов, для которых будет построена КПСЦ. Затем необходимо преобразовать текущие параметры имитационной модели производственного процесса в формат параметров КПСЦ. Далее происходит обновление параметров КПСЦ в соответствии с текущими параметрами имитационной модели производственного процесса и визуализация текущей КПСЦ. Следующим шагом является имитационное моделирование производственного процесса с текущими параметрами КПСЦ. Потом необходимо провести проверку на наличие изменений параметров производственного процесса. При выполнении данной операции по

результатам опроса рабочих мест регистрируются изменения следующих параметров производственного цикла:

- время, необходимое для завершения технологической операции на данной рабочем месте (время цикла, ВЦ);
- время, необходимое на подготовку данного рабочего места к выполнению технологической операции (время переналадки, ВП);
- частота отгрузки объектов обработки для данного рабочего места.

После считывания измененных параметров производственного процесса происходит расчет статистических характеристик.

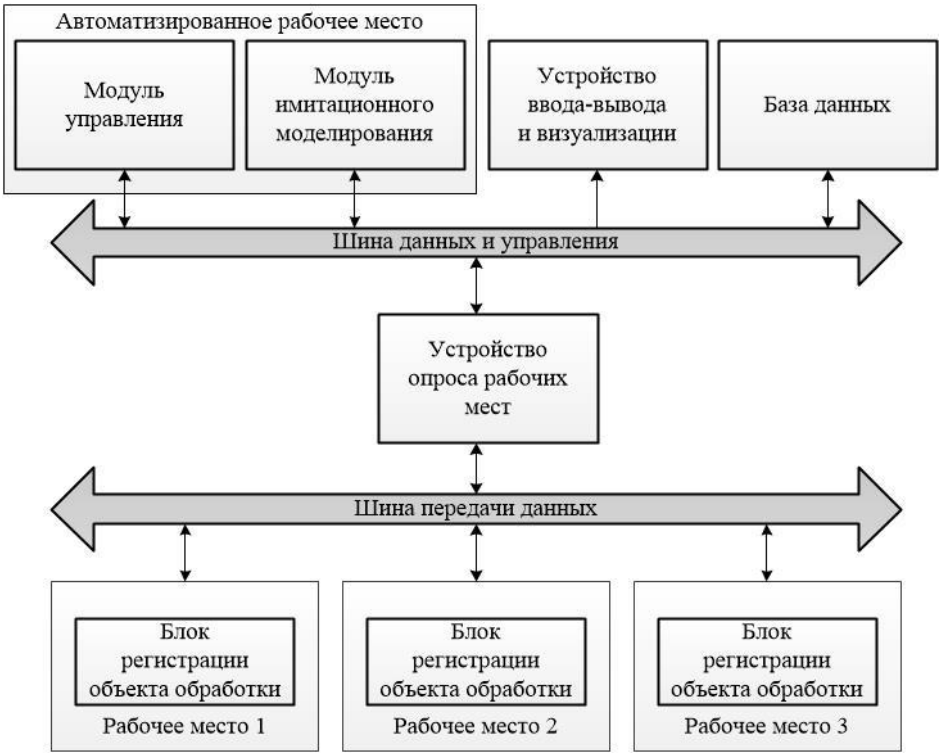

Рис. 4. Архитектура автоматизированной системы динамического картирования потока создания ценности
Fig. 4. The architecture of an automated dynamic value stream mapping system

Архитектура автоматизированной системы динамического картирования потока создания ценности представлена на рис. 4. Предлагаемая архитектура включает следующие компоненты: автоматизированное рабочее место, состоящее из модуля управления и модуля имитационного моделирования, устройство ввода-вывода и визуализации, базы данных, устройства опроса рабочих мест, блоков регистрации объекта обработки, размещенные на рабочих местах, шину передачи данных, шину данных и управления. Взаимодействие автоматизированного рабочего места устройства ввода-вывода и визуализации с базой данных обеспечивается шиной данных и управления, также данной шиной обеспечивается взаимодействие модуля опроса рабочих мест и модуля управления. Взаимодействие модуля опроса рабочих мест и блоков регистрации объекта обработки обеспечивается шиной передачи данных.

Автоматизированная система динамического картирования создания ценности

Для формирования КПСЦ необходимо преобразование текущих параметров имитационной модели производственного процесса в формат параметров КПСЦ.

Соотношение данных производства как СМО и КПСЦ
Correlation of production data as queuing system and value stream map

Атрибуты СМО	Данные КПСЦ
Обслуживающие устройства	Последовательность операций
Время обслуживания	Время цикла + Время переналадки
Тип и количество ресурса	Количество операторов, выполняющих каждый процесс
Доступность обслуживающих устройств	Доступное рабочее время на каждой рабочей станции Количество рабочих дней в месяце Количество смен на каждое рабочее место
Транзакты	Размер партии
Размер очереди	Вместимость очереди не ограничена
Дисциплина очереди	FIFO – первый пришел, первый вышел
Частота поступления заявок	Количество товаров, заказанных покупателем

При выполнении данной операции преобразованию подвергаются следующие текущие параметры имитационной модели производственного процесса:

- последовательность обслуживающих устройств;
- время обслуживания заявки;
- тип и объем доступных ресурсов;
- доступность обслуживающих устройств для обработки заявок;
- свойства динамических объектов системы массового обслуживания;
- дисциплина обслуживания очереди;
- частота поступления заявок на обслуживание.

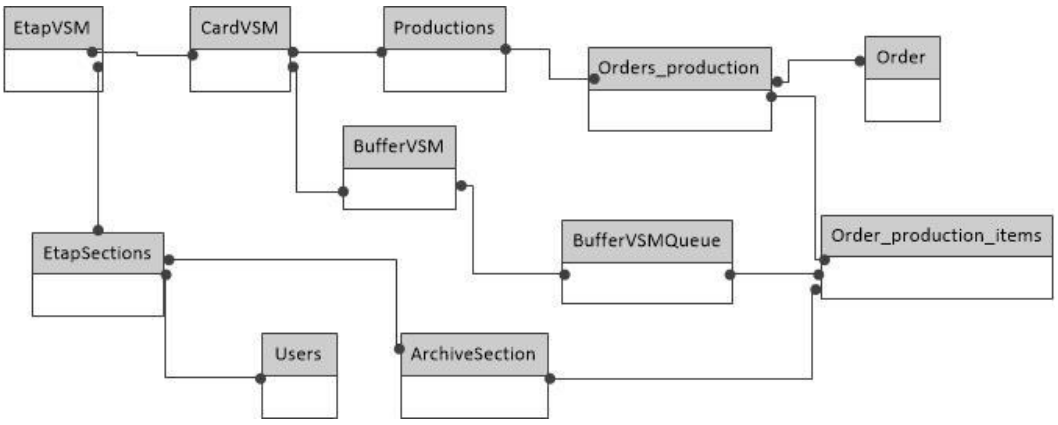

Рис. 5. Структура базы данных автоматизированной системы
Fig. 5. Database structure of the automated system

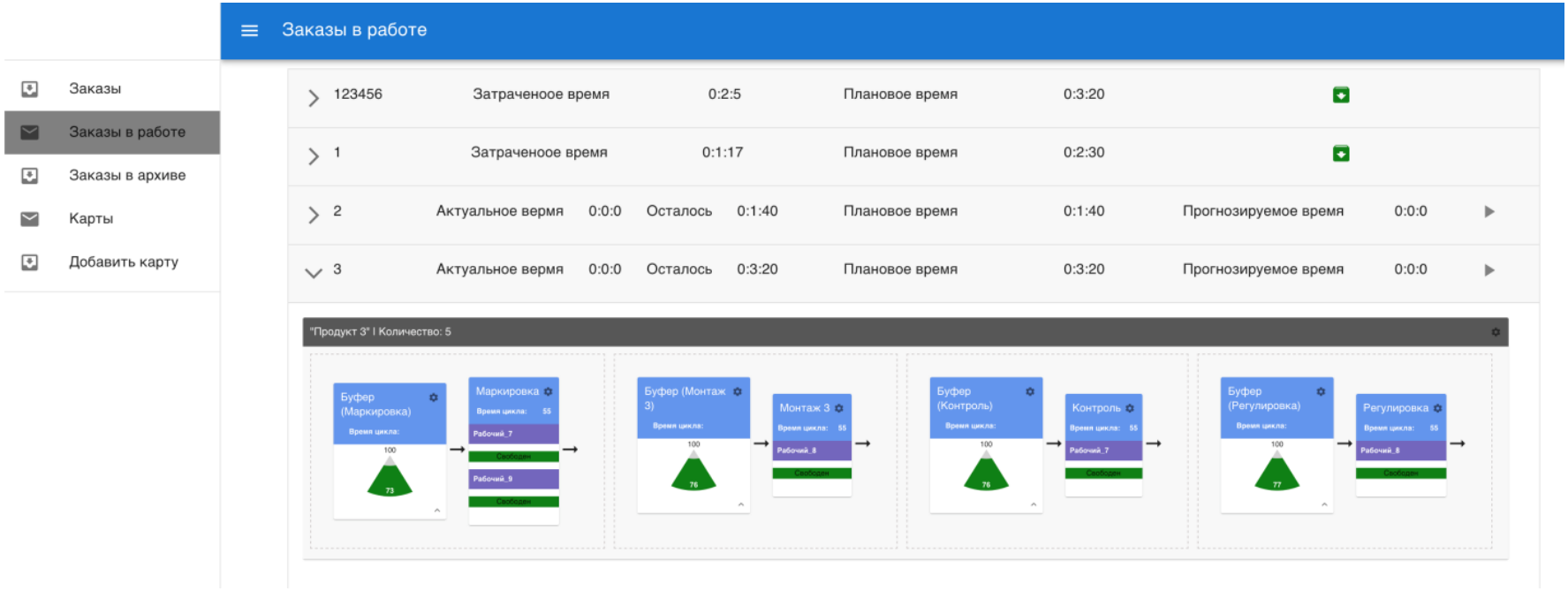

Рис. 6. Навигация программной реализации
Fig. 6. Software implementation navigation

Структура базы данных (см. рис. 5) включает себя таблицу «CardVSM», описывающую КПСЦ конкретного продукта. Данная таблица включает следующие поля: «ProductionsId» – указывает, в каком продукте используется данная операция с буфером; «EtapNumeric» – число, указывающее на последовательность секций в карте для реализации параллельных операций; «BufferVSM» – указатель на буфер, который используется в данной карте по данному продукту из всех буферов, которые существуют в системе (внешний ключ); «EtapVSM» – указатель на операцию, которая выполняется в данной карте по данному продукту из всех операций в системе (внешний ключ). В таблице «EtapVSM» дана информация по технологическим операциям: имя, время цикла.

На рис. 6 представлено основное меню интерфейса программной реализации. Разработанная база данных позволяет отображать общий список заказов на производстве, заказов в работе, заказов в архиве. Сформировать новый заказ можно при нажатии кнопки « +», таким образом вводятся параметры заказа, количество и тип продукции, приоритет на изготовление. При нажатии кнопки «Сформировать» создается заказ в системе, но заказ при этом окрашен серым цветом, это значит, что данный заказ только сформирован, но не добавлен в производство, и симуляция данного заказа не запускалась. Заказы отображаются разными цветами в зависимости от их статуса в системе. Так, зеленым окрашивается заказ, выполнение которого не отстает от заданного планового времени. Желтым – заказы с отставанием не более чем на 10 % от планового времени, красным – отставание более 10 %. На вкладке навигации «Заказы в работе» отображаются выполняющиеся заказы, при этом показывается актуальное время работы заказа, сколько еще будет изготовляться заказ, учитывая текущую загрузку производственных мощностей. На каждой карте желтым цветом выделяется текущий этап выполнения, серым цветом окрашены выполненные этапы, а белым – будущие.

Заключение

Подходы бережливого производства, нацеленные на сокращение потерь в производственной системе, позволяют повышать эффективность производства. Картирование потока создания ценности – это ключевой инструмент данной концепции. Его примение позволяет понять и упорядочить производственные процессы. Обычно данная методика выполняется в ручном режиме, что снижает ее эффективность. Создание динамических карт потока создания ценности позволяет анализировать и оптимизировать более сложные системы, чем традиционные КПСЦ. В статье предложен подход для мониторинга в реальном времени и моделирования карты потока создания ценности. С использованием автоматизированного сбора данных и параметров производственного процесса происходит расчет статических характеристик и параметров загрузки рабочих мест. В отличие от статичного подхода, данная система позволяет создавать динамическую карту потока создания ценности для всего числа продуктов, находящихся в производстве. Благодаря модулю симуляции пользователь может заранее изучить последствия добавления того или иного заказа в систему, возможность выполнения заказа в срок. Комбинирование реальных производственных данных и данных, полученных при помощи имитационного моделирования, позволяет как производить текущий мониторинг выполнения производственных заказов, так и эффективно оценивать будущее состояние системы.

Список литературы

1. **Gomez Paredes F. J., Godinho Filho M., Thurer M., Fernandes N. O., Chiappeta Jabbour C. J.** Factors for chossing production control systems in make-to-order shops: a systematic literature review. *Journal of Intelligent Manufacturing*, 2020. DOI 10.1007/s10845-020-01673-z

2. **Lu Y., Xu X.** Resource virtualization: A core technology for developing cyber-physical production systems. *Journal of Manufacturing Systems*, 2018, vol. 47, pp. 128–140. DOI 10.1016/j.jmsy.2018.05.003
3. **Sylla A., Guillon D., Vareilles E., Aldanondo M., Coudert T., Geneste L.** Configuration knowledge modeling: How to extend configuration from assemble / make to order towards engineer to order for the bidding process. *Computers in Industry*, 2018, vol. 99, pp. 29–41. DOI 10.1016/j.compind.2018.03.019
4. **Qiu S., Ming X., Sallak M., Lu J.** Joint optimization of production and condition-based maintenance scheduling for make-to-order manufacturing systems. *Computers & Industrial Engineering*, 2021, vol. 162. DOI 10.1016/j.cie.2021.107753
5. **Kapulin D. V., Russkikh P. A.** Analisis and improvement of production planning within small-batch make-to-order production. *Journal of Physics: Conference Series*, 2020, vol. 1515, no. 2. DOI 10.1088/1742-6596/1515/2/022072
6. **Русских П. А., Капулин Д. В.** Анализ решений для создания и реализации механизмов адаптивного планирования позаказного производства // Вестник МГТУ «Станкин». 2021. Т. 1, вып. 56. С. 44–48.
7. **Kishimoto K., Medina G., Sotelo F., Raymundo C.** Application of Lean Manufacturing Techniques to Increase On-Time Deliveries: Case Study of a Metalworking Company with a Make-to-Order Environment in Peru. *Human Interaction and Emerging Technologies. IHIET 2019. Advances in Intelligent Systems and Computing*, 2020, vol. 1018. DOI 10.1007/978-3-030-25629-6_148
8. **Garza-Reyes J. A., Romero J. T., Govindan K., Cherrafi A., Ramanathan U.** A PDCA-based approach to Environmental Value Stream Mapping. *Journal of Cleaner Production*, 2018, vol. 180, pp. 335–348. DOI 10.1016/j.jclepro.2018.01.121
9. **Stadnicka D., Litwin P.** Value stream mapping and system dynamics integration for manufacturing line modelling and analysis. *International Journal of Production Economic*, 2019, vol. 208, pp. 400–411. DOI 10.1016/j.ijpe.2018.12.011
10. **Antonelli D., Stadnicka D.** Combining factory simulation with value stream mapping: a critical discussion. *Procedia CIRP*, 2018, vol. 67, pp. 30–35. DOI 10.1016/j.procir.2017.12.171
11. **Memon R. A., Li J., Ahmed J., Khan A., Nazir M. I., Mangrio M. I.** Modeling of Blockchain Based Systems Using Queuing Theory Simulation. *2018 15th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP*), 2018, pp. 107–111. DOI 10.1109/ICCWAMTIP.2018.8632560
12. **Agalianos K., Ponis S.T., Aretoulaki E., Plakas G., Efthymiou O.** Discrete Event Simulation and Digital Twins: Review and Challenges for Logistics. *Procedia Manufacturing*, 2020, vol. 51, pp. 1636–1641. DOI 10.1016/j.promfg.2020.10.228
13. **Shah R., Ward P. T.** Defining and developing measures of lean production. *Journal of Operations Management*, 2007, vol. 25, no. 4, pp. 785–805. DOI 10.1016/j.jom.2007.01.019
14. **Wagner T., Herrmann C., Thiede S.** Industry 4.0 Impacts on Lean Production Systems. *Procedia CIRP*, 2017, vol. 63, pp. 125–131. DOI 10.1016/j.procir.2017.02.041
15. **Wang P., Wu P., Chi H.L, Li X.** Adopting lean thinking in virtual reality-based personalized operation training using value stream mapping. *Automation in Construction*, 2020, vol. 119. DOI 10.1016/j.autcon.2020.103355
16. **Zhu X. Y., Zhang H., Jiang Z. G.** Application of green-modified value stream mapping to integrate and implement lean and green practices: A case study. *International Journal of Computer Integrated Manufacturing*, 2020, vol. 33, no. 7, pp. 716–731. DOI 10.1080/0951192X.2019.1667028

References

1. **Gomez Paredes F. J., Godinho Filho M., Thurer M., Fernandes N. O., Chiappeta Jabbour C. J.** Factors for chossing production control systems in make-to-order shops: a systematic literature review. *Journal of Intelligent Manufacturing*, 2020. DOI 10.1007/s10845-020-01673-z
2. **Lu Y., Xu X.** Resource virtualization: A core technology for developing cyber-physical production systems. *Journal of Manufacturing Systems*, 2018, vol. 47, pp. 128–140. DOI 10.1016/j.jmsy.2018.05.003
3. **Sylla A., Guillon D., Vareilles E., Aldanondo M., Coudert T., Geneste L.** Configuration knowledge modeling: How to extend configuration from assemble / make to order towards engineer to order for the bidding process. *Computers in Industry*, 2018, vol. 99, pp. 29–41. DOI 10.1016/j.compind.2018.03.019
4. **Qiu S., Ming X., Sallak M., Lu J.** Joint optimization of production and condition-based maintenance scheduling for make-to-order manufacturing systems. *Computers & Industrial Engineering*, 2021, vol. 162. DOI 10.1016/j.cie.2021.107753
5. **Kapulin D. V., Russkikh P. A.** Analisis and improvement of production planning within small-batch make-to-order production. *Journal of Physics: Conference Series*, 2020, vol. 1515, no. 2. DOI 10.1088/1742-6596/1515/2/022072
6. **Russkikh P. A., Kapulin D. V.** Analiz resheniy dly sozdaniy i realizacii mechanizmov adaptivnogo planirovaniy pozakaznogo proizvodstva [Analysis of solutions for the creation and implementation of adaptive planning mechanisms for make-to-order production]. *Vestnik MSTU STANKIN*, 2021, vol. 1, no. 56, pp. 44–48. (in Russ.) DOI 10.1088/1742-6596/1515/2/022072
7. **Kishimoto K., Medina G., Sotelo F., Raymundo C.** Application of Lean Manufacturing Techniques to Increase On-Time Deliveries: Case Study of a Metalworking Company with a Make-to-Order Environment in Peru. *Human Interaction and Emerging Technologies. IHIET 2019. Advances in Intelligent Systems and Computing*, 2020, vol. 1018. DOI 10.1007/978-3-030-25629-6_148
8. **Garza-Reyes J. A., Romero J. T., Govindan K., Cherrafi A., Ramanathan U.** A PDCA-based approach to Environmental Value Stream Mapping. *Journal of Cleaner Production*, 2018, vol. 180, pp. 335–348. DOI 10.1016/j.jclepro.2018.01.121
9. **Stadnicka D., Litwin P.** Value stream mapping and system dynamics integration for manufacturing line modelling and analysis. *International Journal of Production Economic*, 2019, vol. 208, pp. 400–411. DOI 10.1016/j.ijpe.2018.12.011
10. **Antonelli D., Stadnicka D.** Combining factory simulation with value stream mapping: a critical discussion. *Procedia CIRP*, 2018, vol. 67, pp. 30–35. DOI 10.1016/j.procir.2017.12.171
11. **Memon R. A., Li J., Ahmed J., Khan A., Nazir M. I., Mangrio M. I.** Modeling of Blockchain Based Systems Using Queuing Theory Simulation. *2018 15th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP*), 2018, pp. 107–111. DOI 10.1109/ICCWAMTIP.2018.8632560
12. **Agalianos K., Ponis S.T., Aretoulaki E., Plakas G., Efthymiou O.** Discrete Event Simulation and Digital Twins: Review and Challenges for Logistics. *Procedia Manufacturing*, 2020, vol. 51, pp. 1636–1641. DOI 10.1016/j.promfg.2020.10.228
13. **Shah R., Ward P. T.** Defining and developing measures of lean production. *Journal of Operations Management*, 2007, vol. 25, no. 4, pp. 785–805. DOI 10.1016/j.jom.2007.01.019
14. **Wagner T., Herrmann C., Thiede S.** Industry 4.0 Impacts on Lean Production Systems. *Procedia CIRP*, 2017, vol. 63, pp. 125–131. DOI 10.1016/j.procir.2017.02.041
15. **Wang P., Wu P., Chi H.L, Li X.** Adopting lean thinking in virtual reality-based personalized operation training using value stream mapping. *Automation in Construction*, 2020, vol. 119. DOI 10.1016/j.autcon.2020.103355

16. **Zhu X. Y., Zhang H., Jiang Z. G.** Application of green-modified value stream mapping to integrate and implement lean and green practices: A case study. *International Journal of Computer Integrated Manufacturing*, 2020, vol. 33, no. 7, pp. 716–731. DOI 10.1080/0951192X.2019.1667028

Информация об авторах

Полина Андреевна Русских, ассистент кафедры
Денис Владимирович Капулин, кандидат технических наук
Олег Владимирович Дрозд, старший преподаватель
Сергей Юрьевич Смоглюк, аспирант

Information about the Authors

Polina A. Russkikh, Assistant of the Basic Department
Denis V. Kapulin, Head of the Basic Department
Oleg V. Drozd, Senior Lecturer
Sergey Yu. Smogluk, Postgraduate Student

Статья поступила в редакцию 11.12.2021;
одобрена после рецензирования 01.02.2022; *принята к публикации 01.02.2022*
The article was submitted 11.12.2021;
approved after reviewing 01.02.2022; *accepted for publication 01.02.2022*

Научная статья

УДК 004.451
DOI 10.25205/1818-7900-2022-20-1-81-96

Разработка PowerShell-сценариев для автоматизации процессов администрирования ОС Windows. Автоматизация процесса создания и распространения SSL-сертификатов на серверы

Надежда Александровна Федькова [1]
Дмитрий Андреевич Федьков [2]

[1, 2] Брянский государственный аграрный университет
Брянск, Россия

[1] voytova.nady@yandex.ru
[2] dmitrijfedkov@gmail.com

Аннотация

Рассматривается процесс автоматизации администрирования ОС Windows, в частности автоматизация процесса создания и распространения SSL-сертификатов на серверы. Для разработки применяется встроенная командная оболочка PowerShell. Приведены теоретические аспекты изучаемой проблематики: понятие SSL-сертификатов, назначение хранилища Vault и сервиса Consul. Представлены результаты разработки в виде листингов кода.

Ключевые слова

администрирование, MS Windows, SSL-сертификат, PowerShell

Благодарности

Работа выполнена при поддержке ФГБОУ ВО «Брянский ГАУ»

Для цитирования

Федькова Н. А., *Федьков Д. А.* Разработка PowerShell-сценариев для автоматизации процессов администрирования ОС Windows. Автоматизация процесса создания и распространения SSL-сертификатов на серверы // Вестник НГУ. Серия: Информационные технологии. 2022. Т. 20, № 1. С. 81–96. DOI 10.25205/1818-7900-2022-20-1-81-96

Development of PowerShell Scripts for Automating Windows OS Administration Processes. Automating the Process of Creating and Distributing SSL Certificates to Servers

Nadezhda A. Fedkova [1], **Dmitry A. Fedkov** [2]

[1, 2] Bryansk State Agrarian University
Bryansk, Russia

[1] voytova.nady@yandex.ru
[2] dmitrijfedkov@gmail.com

Abstract

The article describes the process of automating Windows administration, in particular, automating the process of creating and distributing SSL certificates to servers. The built-in command shell PowerShell is used for development.

© Федькова Н. А., Федьков Д. А., 2022

Theoretical aspects of the subject are given: the concept of SSL certificates, the purpose of Vault storage and Consul service. The results of development in the form of code listings are presented.

Keywords

administration, MS Windows, SSL certificate, PowerShell

Acknowledgements

The work was carried out with the support of FSBEI HE “Bryansk SAU”

For citation

Fedkova N. A., Fedkov D. A. Development of PowerShell Scripts for Automating Windows OS Administration Processes. Automating the Process of Creating and Distributing SSL Certificates to Servers. *Vestnik NSU. Series: Information Technologies*, 2022, vol. 20, no. 1, pp. 81–96. (in Russ.) DOI 10.25205/1818-7900-2022-20-1-81-96

Введение

Большинство серверов используется в качестве хостинга для сервисов, а большинство сервисов для безопасной передачи данных используют HTTPS протокол [1–4]. Для работы сервиса на протоколе HTTPS ему необходимы сертификат с открытым и сертификат закрытым ключом. Сертификат открытого ключа подтверждает принадлежность данного открытого ключа владельцу сайта. Сертификат открытого ключа и сам открытый ключ посылаются клиенту при установлении соединения. Закрытый ключ используется для расшифровки сообщений от клиента. Так как сервисы взаимодействую между собой в одном локальном центре обработки данных, для создания сертификатов используются частный центр сертификации Microsoft Certificate Authority Service.

Центры сертификации (Certificate Authorities / CA), выпускают цифровые сертификаты – поддающиеся проверке небольшие файлы данных, содержащие идентификационные данные, которые помогают веб-сайтам, пользователям и устройствам представлять свою подлинную личность в Интернете (подлинную, потому что ЦС проверил личность). ЦС играют важную роль в функционировании Интернета и в осуществлении прозрачных, надежных транзакций в сети. Ежегодно ЦС выпускают миллионы цифровых сертификатов, которые используются для защиты информации, шифрования миллиардов транзакций и обеспечения безопасной связи.

SSL-сертификат – это популярный тип цифрового сертификата, который связывает данные о владельце веб-сервера (веб-сайта) с криптографическими ключами. Эти ключи используются в протоколе SSL/TLS для активации безопасной сессии между браузером и веб-сервером, на котором размещен SSL-сертификат. Для того чтобы браузер мог доверять SSL-сертификату и установить SSL/TLS-сессию без предупреждений безопасности, SSL-сертификат должен содержать доменное имя использующего его веб-сайта, быть выданным надежным центром сертификации и не иметь истекшего срока действия.

Выполнение процессов работы с сертификатами можно осуществлять «в ручном» режиме, выполняя настройки с помощью пользовательского интерфейса. Но в случае большой серверной инфраструктуры этот процесс становится слишком трудоемким и занимает много времени. С целью оптимизации этого процесса целесообразна автоматизация процессов «ручного» администрирования. Автоматизация в среде Windows выполняется с помощью интегрированного скриптового языка PowerShell.

Предложенная ниже разработка является одним из компонентов системы автоматизации серверной инфраструктуры на базе ОС Windows Server. Ядром этой системы является сервис каталога сервисов, который состоит из пользовательского web-интерфейса, а также HTTP API. В данном каталоге хранится информация о серверах и web-сервисах компании, которые размещаются на этих серверах. Для повышения безопасности необходимо перевести все сетевые взаимодействия между web-сервисами с использованием шифрования, для этого требуются цифровые сертификаты, реализующие возможность создания безопасного шифрованного соединения по протоколу HTTPS. Для инфраструктуры, построенной на базе серверов с ОС Windows Server, актуально использовать центр сертификации Microsoft Certificate

Authority Service. Остается выбрать автоматизированный способ выпуска и доставки сертификатов на сервера. Это можно сделать штатными средствами ОС Windows с применением групповых политик, чтобы каждый сервер выпускал самостоятельно сертификаты для всех имеющихся web-сервисов, но, когда в компании сотни серверов и сотни сервисов, это многократно увеличивает количество выпускаемых сертификатов, которые по факту ни чем не отличаются. Например, для 100 серверов и 500 сервисов выпускается 50 000 сертификатов, а добавление одного нового сервера приводит к выпуску еще 500 сертификатов и т. д. На каждом сервере одновременно не размещаются все 500 сервисов, они равномерно разделены на группы, но при использовании механизма групповых политик этим невозможно управлять, чтобы выпускать сертификаты только для тех web-сервисов, которые располагаются на определенных серверах, а также эту систему сложно мониторить в случае сбоев в ее работы. В связи с этим для высоконагруженных систем целесообразно разработать отдельный компонент автоматизации в виде PowerShell-сценария. Данное решение является гибким, так как в дальнейшем можно реализовать любую специфическую для копания функциональность выпуска и доставки сертификатов. Также в результате тесной интеграции с сервисом каталога сервисов имеется возможность запросить средствами HTTP API информацию о имеющихся в компании web-сервисах и серверах, на которых данные сервисы размещены. С учетом этой информации для каждого web-сервиса выпускается только один сертификат, а после доставляется только на те сервера, где размещен этот сервис, и в случае с примером выше для 100 серверов и 500 сервисов будет выпущено всего 500 сертификатов. Все чувствительные данные, в том числе конфигурация скрипта, хранятся в хранилище секретов Vault, а не просто в виде обычного файла рядом со скриптом или в самом скрипе, что повышает безопасность. Только пользователи с особыми привилегиями могут просматривать или изменять данную конфигурацию. При необходимости можно доработать скрипт, для взаимодействия с любым другим центром сертификации, который имеет API для выпуска сертификатов.

Автоматизация процесса создания и распространения SSL-сертификатов на серверы

Для автоматизации процесса создания и распространения сертификатов на серверы предлагается разработка PowerShell-сценария – скрипта DeployCerts.ps1, который будет выполняться на одном из серверов управления. Его работа будет заключаться в следующем:

- получение информации из каталога сервисов;
- выпуск новых сертификатов в центре сертификации;
- обновление устаревающих сертификатов в центре сертификации;
- распространение сертификатов сервисов по серверам;
- обновление информации о сертификатах в каталоге сервисов.

Запуск скрипта должен выполняться с помощью планировщика задач операционной системы Windows каждые 15 минут.

На рис. 1. показан общий механизм работы скрипта DeployCerts.ps1.

Далее будут описаны детали реализации.

Организация взаимодействия с хранилищем секретов Vault

В первую очередь скрипту необходимо выполнить чтение конфигурационных данных. Данные конфигурации находятся в «хранилище секретов» Vault. Vault – это инструмент для безопасного доступа к секретам. Секрет – это все, к чему необходимо жестко контролировать доступ, например ключи API, пароли или сертификаты. Vault предоставляет единый интерфейс к любому секрету, обеспечивая жесткий контроль доступа и ведя подробный журнал аудита.

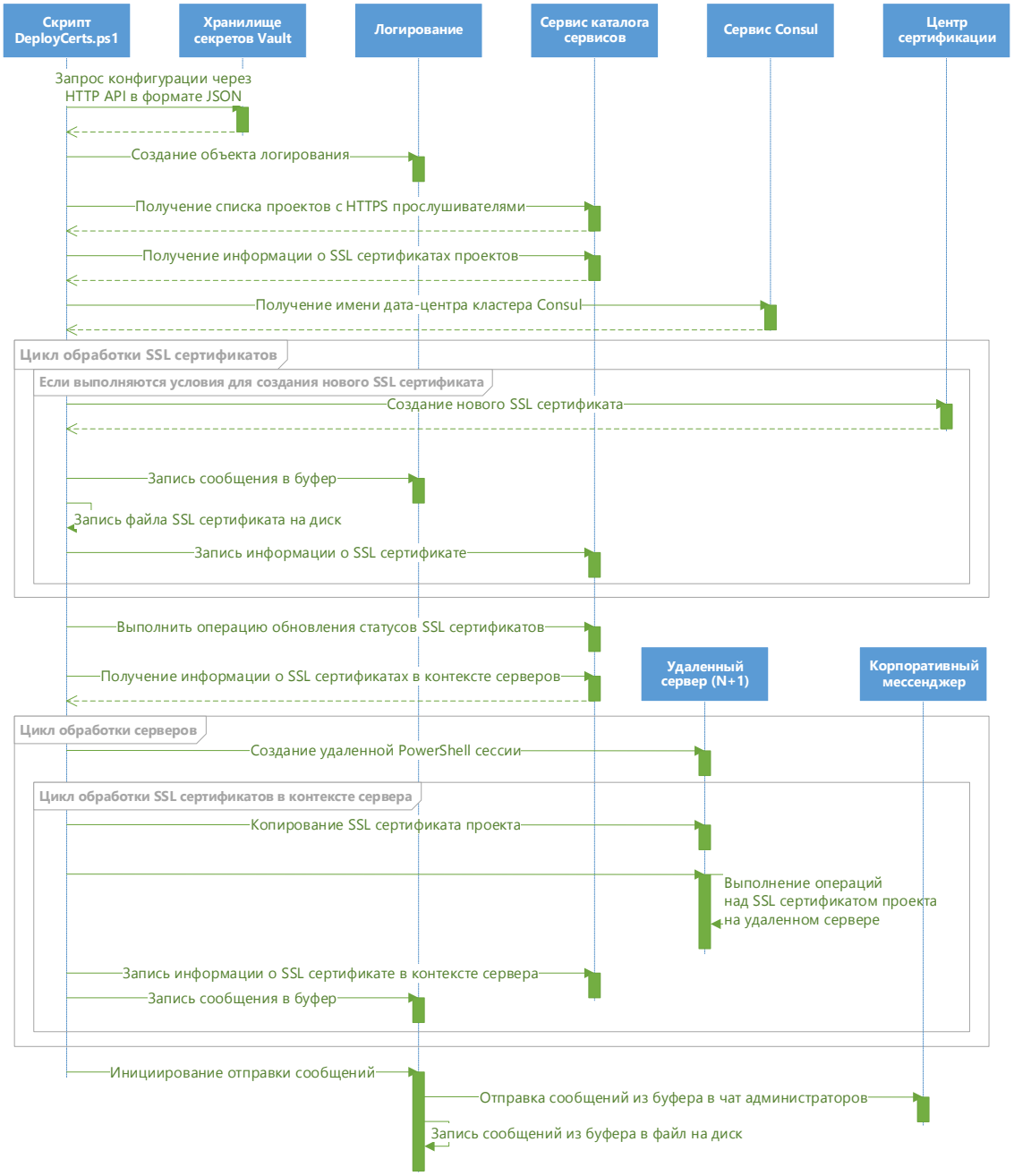

Рис. 1. Механизм работы скрипта DeployCerts.ps1
Fig. 1. The mechanism of the DeployCerts.ps1 script

Современные информационные системы требуют доступа к множеству секретов: учетные данные базы данных, API-ключи для внешних сервисов, учетные данные для взаимодействия сервис-ориентированной архитектуры и т. п. Понять, кто и к каким секретам имеет доступ, очень сложно, и это зависит от конкретной платформы. Добавить к этому продление ключей, безопасное хранение и подробные журналы аудита практически невозможно без специального решения. В таких ситуациях помогает Vault.

Конфигурация скрипта DeployCerts.ps1 в хранилище Vault хранится в формате JSON.

Описание структуры конфигурации

В свойстве «AdditionalDomains» указывается список доменных имен или список шаблонов доменных имен, которые будут добавлены в свойство сертификата в поле «Дополнительное имя субъекта / Subject Alternative Name» во время его выпуска в центре сертификации. Скрипт при разборе значений строк DNS имен, вместо шаблонов {Pattern} подставит значения.

{WinDomainFqdn} – полное имя домена AD, к которому присоединен компьютер, выполняющий скрипт (компьютер, с которого распространяются сертификаты).

{ConsulDcName} – имя DC Consul, к которому подключен агент компьютера, на котором выполняется скрипт (компьютер, с которого распространяются сертификаты).

{ProjectName} – имя проекта.

В значении свойства «AdditionalDomains» может быть указан пустой массив или массив строк, а также это свойство вообще может отсутствовать, если в нем нет необходимости, это не приведет к ошибке выполнения скрипта.

В свойстве «ChatId» указывается идентификатор корпоративного чата, для уведомлений обслуживающего персонала о работе скрипта. В случае создания новых сертификатов или обновлении старых, а также при возникновении ошибок во время работы, скрипт отправляет сообщения в указанный в конфигурации чат. Данная функция является полезной при возникновении ошибок в работе скрипта и позволяет быстро диагностировать и устранить проблему. Свойство принимает строку.

Свойство «PassPfx» хранит пароль, которым шифруется pfx файл сертификата, выпускаемый в центре сертификации. Это обязательное свойство конфигурации. Если оно отсутствует, скрипт выдаст исключение при чтении конфигурации. Данное свойство должно быть строкой.

В свойстве «RenewalPeriodDays» указывается количество дней до конца срока действия сертификата (в этот период будет производиться обновление сертификатов). Это необязательное свойство, если оно не заполнено, то по умолчанию будет использоваться значение 30. Данное свойство должно быть числом.

Чтение конфигурации выполняется с помощью функции GetConfigFromVault, которая в случае успешного чтения конфигурации вернет экземпляр класса DeployCertsConfig (рис. 2).

```
function GetConfigFromVault # возвращает объект типа DeployCertsConfig
{
    [CmdletBinding()]
    Param()

    [PSCustomObject]$configData = $null
    [DeployCertsConfig]$deployCertsConfig = $null
    try {
        $configData = Get-AdmAbtVltData -SecretPath "[illegible]"
    } catch {
        throw "I couldn't read the parameters from Vault."
    }
    if(!$configData) {
        throw "An empty configuration object is obtained."
    }
    try {
        $deployCertsConfig = [DeployCertsConfig]::new($configData)
    } catch {
        throw ("Configuration initialization error. Detail info: ({0})" -f $_.ToString())
    }
    return $deployCertsConfig
}
```

Рис. 2. Листинг функции GetConfigFromVault (чтение конфигурации)
Fig. 2. Listing of the Get Config From Vault function (reading the configuration)

Функция выполняет вызов командлета Get-AdmAbtVltData, который выполняет чтение конфигурации в формате JSON из Vault через HTTP API, затем десериализацию из JSON в объект типа PSCustomObject и возвращает функции GetConfigFromVault. Если во время чтения конфигурации возникли ошибки, или командлет Get-AdmAbtVltData вернул пустой объект, будет выдано исключение, и скрипт прервет выполнение. Иначе полученный объект будет передан в качестве параметра в конструктор класса DeployCertsConfig. В конструкторе класса DeployCertsConfig будет выполнена проверка свойств объекта, полученного в параметре $_ConfigObjectVlt.

Если свойство $_ConfigObjectVlt.AdditionalDomains существует и содержит больше нуля элементов, то все элементы массива этого свойства будут записаны в одноименное свойство экземпляра класса.

Далее проверяется свойство $_ConfigObjectVlt.PassPfx. Если это свойство не существует или его значение пустое, будет выдано исключение, что прервет выполнение скрипта, иначе значение будет преобразовано в зашифрованную строку и присвоено свойству PasswordPfx экземпляра класса. Последним проверяется свойство $_ConfigObjectVlt.RenewalPeriodDays, если свойство существует и его значение не равно нулю, то значение будет присвоено свойству экземпляра класса RenewalPeriod, иначе свойству ничего не будет присвоено, и будет использоваться значение по умолчанию, равное 30. Если в конструкторе не возникло исключений, он вернет функции GetConfigFromVault экземпляр класса DeployCertsConfig.

Логирование

После успешного чтения конфигурации будет создан экземпляр класса writeProcessing Class.

Данный класс предназначен, для записи сообщений статуса в буфер, форматирование поступающих сообщений HTML тегами в соответствии с их статусом (цвет, полужирное начертание), а также отправку данных из буфера в чат корпоративного мессенджера.

Для создания экземпляра класса writeProcessingClass вызывается единственный конструктор, который принимает два параметра, это идентификатор чата и заголовок сообщения, отправляемого в чат. Заголовок сразу будет отформатирован и записан в общий буфер сообщений. Также в конструкторе будет инициализировано поле LogFilePath, в данное поле будет записан путь к файлу журнала. Метод Write класса writeProcessingClass предназначен для записи сообщения в буфер, этот метод принимает три аргумента: строку, которую необходимо записать в буфер, значение перечисления writeProcessingEnum (это перечисление отвечает за статус сообщения), последний параметр имеет тип bool, и, если он является true, сообщение будет отформатировано как полужирное. Метод Push выполняет запись буфера сообщений в файл, путь которого определен в свойстве LogFilePath, а также выполняет отправку буфера сообщений в чат, предварительно обработав буфер функцией ThisScript.Format Message (рис. 3).

```
function ThisScript.FormatMessage
{
    [CmdletBinding()]
    Param(
        [Parameter(Mandatory=$True)][ValidateNotNullOrEmpty()][Alias("Messages")][string[]]$Messages_fnc0
    )
    [timespan]$timeExecScriptSec = ((Get-Date)-$thisScriptStrtTime)
    $Messages_fnc0 += "[color=Silver][size=10]Execution time: {0} sec.[/size][/color]" -f $timeExecScriptSec.TotalSeconds
    [string]$outFunction0 = $null
    $outFunction0 = [string]::Join("`n", $Messages_fnc0)
    return $outFunction0
}
```

Рис. 3. Листинг функции ThisScript.FormatMessage (оценка времени выполнения скрипта)
Fig. 3. Listing of This Script.FormatMessage function (evaluation of script execution time)

Функция ThisScript.FormatMessage принимает один параметр, который является буфером, сформированным в классе writeProcessingClass. Функция добавляет в «подвал» сообщения информацию о том, сколько времени прошло между запуском скрипта и текущим временем, т. е. эта строка, показывает, сколько времени заняло выполнение скрипта. Далее функция преобразует буфер, который является массивом строк, в одну единственную строку и возвращает эту строку вызывающему методу.

Операции с сервисом каталога сервисов

Далее в глобальной области видимости создается экземпляр класса обработчика и помещается в переменную $serviceManagerHandler, с помощью которого выполняется взаимодействие с HTTP API сервиса каталога сервисов. Внутренняя структура класса обработчика сервиса каталога сервисов, рассматриваться не будет, с ним мы работает как с «черным ящиком».

После этого необходимо вызвать метод $serviceManagerHandler.GetProjectsContainsSsl Binding(), который вернет массив строк, данные строки содержат наименования проектов, у которых в СКС имеются прослушивали с протоколом HTTPS.

Если функция вернет пустой массив, следовательно, нет сервисов для обработки, и в этом случае скрипт завершает работу. Если список сервисов был получен, то необходимо загрузить информацию о доступных сертификатах из сервиса каталога сервисов, это делается с помощью вызова функции $serviceManagerHandler.GetProjectCertificates(). Вызов функции вернет массив объектов, свойства объектов массива представлены в табл. 1.

Таблица 1

Свойства объекта сертификата

Table 1

Certificate Object Properties

Имя свойства	Тип	Пример значения	Описание
certId	int	100	Числовой идентификатор сертификата
projectName	string	Project	Имя сервиса
thumbprint	string	C8CCE305D3D0282860882E2ADA1CB7F54CC5FDA0	Значение хэша сертификата
certExpires	DateTime	2022-06-22T00:00:00	Время истечения сертификата
oldThumbprint	string	6A26BC92CE7611C209078C85CD1A5059F98AFD45	Значение хэша сертификата предшествующего, текущему сертификату
certFileName	string	4c7392c9-2925-4aeb-90d9-e2e95fc23527.pfx	Имя файла сертификата

После этого будет выполнено получение имени дата центра кластера Consul, к которому присоединен сервер и на котором выполняется скрипт DeployCerts.ps1.

Получение информации из сервиса Consul

Consul – это решение для Service Mesh, предоставляющее полнофункциональную плоскость управления с функциями обнаружения, конфигурации и сегментации сервисов. Каждая из этих функций может использоваться отдельно по мере необходимости, или они могут использоваться вместе для создания полнофункционального Service Mesh. Consul требует наличия плоскости данных и поддерживает как прокси, так и встроенную модель интеграции.

Consul поставляется с простым встроенным прокси, чтобы все работало «из коробки», но также поддерживает интеграцию прокси сторонних производителей, таких как Envoy. Consul разработан таким образом, чтобы быть «дружественным» как для сообщества DevOps, так и для разработчиков приложений, что делает его идеальным для современных «эластичных» инфраструктур.

Базовая архитектура Consul. Consul – это распределенная, высокодоступная система. Каждый узел, предоставляющий услуги Consul, запускает агента Consul. Запуск агента не требуется для обнаружения других сервисов или получения / установки данных ключей / значений. Агент отвечает за проверку работоспособности служб на узле, а также самого узла.

Агенты взаимодействуют с одним или несколькими серверами Consul. На серверах Consul хранятся и реплицируются данные. Серверы сами выбирают «лидера». Хотя Consul может функционировать с одним сервером, рекомендуется использовать от 3 до 5 серверов, чтобы избежать сценариев отказа, приводящих к потере данных. Для каждого центра обработки данных рекомендуется кластер серверов Consul.

Серверы поддерживают каталог, который формируется путем агрегирования информации, предоставленной агентами. Каталог поддерживает высокоуровневое представление кластера, включая информацию о том, какие службы доступны, какие узлы выполняют эти службы, информацию о состоянии здоровья и многое другое.

Компоненты инфраструктуры, которым необходимо обнаружить другие службы или узлы, могут запросить любой из серверов Consul или любой из агентов Consul. Агенты направляют запросы на серверы автоматически.

В каждом центре данных работает кластер серверов Consul. Когда выполняется запрос на обнаружение или настройку службы между центрами данных, локальные серверы Consul направляют запрос в удаленный центр данных и возвращают результат. Это имя необходимо получить, так как оно дальше будет использоваться как заполнитель в шаблонах DNS имен, для поля сертификата «Дополнительное имя субъекта / Subject Alternative Name». Получение имени дата центра кластера Consul выполняется с помощью функции Get-ConsulLocalDc (рис. 4).

```
function Get-ConsulLocalDc
{
    [CmdletBinding()]
    Param()
    try
    {
        return (Invoke-RestMethod `
                    -UseBasicParsing `
                    -Method Get `
                    -Uri "http://localhost:8500/v1/agent/self" `
                    -ErrorAction Stop).Config.Datacenter
    }
    catch
    {
        throw "Error reading Consul configuration. Detaild error: $_"
    }
}
```

Рис. 4. Листинг функции Get-ConsulLocalDc
(получение имени дата центра кластера Consul)

Fig. 4. Listing features Get-Consullocaldc
(name name data center cluster Consul)

Данная функция выполняет HTTP запрос в API агента Consul. Вызываемый метод agent / self возвращает информацию о конфигурации и членах локального агента в формате JSON. Config элемент содержит подмножество конфигурации. Из данного элемента будет выбрано свойство Datacenter, которое будет возвращено функцией.

Далее в глобальной области видимости скрипта необходимо инициализировать переменную $certDnsNamesBuilder ссылкой на экземпляр класса CertDnsNamesBuilder. Создание экземпляра класса CertDnsNamesBuilder выполняется с помощью конструктора, принимающего один параметр, который является хэш-таблицей, где в качестве ключей, записаны имена заполнителей DNS имен, а в качестве значений – значения этих заполнителей.

Первый вызов метода AddDnsNames на экземпляре класса CertDnsNamesBuilder выполняет добавление массива строк DNS имен, используемых по умолчанию. Второй вызов метода, выполняет добавление массива строк DNS имен, определенных в конфигурации в свойстве AdditionalDomains (конфигурация и ее свойства были рассмотрены ранее).

Метод AddDnsNames выполняет добавление массива строк DNS имен в свойство Data экземпляра класса CertDnsNamesBuilder. Данное свойство имеет тип System.Collections.Generic.HashSet[string], который позволяет организовывать строки в список, значения которого не повторяются.

Цикл обработки SSL сертификатов

Далее следует цикл (рис. 5), в котором выполняется обработка сертификатов. В начале цикла обновляется имя сертификата, в значении ключа ProjectName хэш-таблицы свойства Masks экземпляра класса CertDnsNamesBuilder.

```
1   for ($i=0; $i -lt $projectNames.Count; $i++)
2   {
3       [string]$projectNameInFor = $projectNames[$i]
4       $certDnsNamesBuilder.Masks["ProjectName"] = $projectNameInFor
5
6       [PSCustomObject]$certificateSm = $null
7       $certificateSm = $certificatesInfoFromSm | Where-Object {$_.ProjectName -eq $projectNameInFor}
8
9       [System.Security.Cryptography.X509Certificates.X509Certificate2]$creationCert = $null
10      #region Создание сертификата
11      if(!$certificateSm) { # сертификата нет в SM
12          $writeClass.Write(("Create certificate for project `"{0}`":" -f $projectNameInFor), 2, $True)
13          [string]$certName = "{0}.pfx" -f (New-Guid).ToString()
14
15          try
16          {
17              $creationCert = ThisScript.CreateCert   -ProjctName $projectNameInFor `
18                                                      -CertName $certName `
19                                                      -CertPass $deployCertsConfig.PasswordPfx `
20                                                      -DomainName $fqdnDomainName `
21                                                      -CertDnsNames $certDnsNamesBuilder.GetDnsNames()
22              $writeClass.Write(("Request and creation of the certificate - OK"), 2, $False)
23          }
24          catch
25          {
26              $writeClass.Write($_.ToString(), 1, $False)
27              continue
28          }
29
30          try {
31              $isAddInSm = {
32                  [bool]$resultWriteDataInSm = $False
33                  $resultWriteDataInSm = $serviceManagerHandler.AddProjectCertificate(
34                      $projectNameInFor, $creationCert.Thumbprint, $creationCert.NotAfter, $certName)
35
36                  return $resultWriteDataInSm
37              }
38
39              if(!$isAddInSm.InvokeReturnAsIs()) {
40                  throw "The project is not found or has no SSL binding."
41              }
42
43              $writeClass.Write(("Writing data in SM - OK"), 2, $False)
44          } catch {
45              $writeClass.Write($_.ToString(), 1, $False)
46          }
47      }
48      #endregion
49      #region Обновление сертификата···
85      #endregion
86  }
```

Рис. 5. Листинг цикла (создание сертификата)
Fig. 5. Cycle listing (certificate creation)

Затем выполняется поиск информации о сертификате в массиве, на который ссылается переменная $certificatesInfoFromSm. Если объект не был найден, будет выполнена попытка выпуска сертификата в центре сертификации (рис. 5 строки 10–48). Эта операция выполняется с помощью функции ThisScript.CreateCert (рис. 6).

```
function ThisScript.CreateCert
{
    # function 1
    [CmdletBinding()]
    Param
    (
        [Parameter(Mandatory=$True)]
        [ValidateNotNullOrEmpty()]
        [Alias("ProjectName")]
        [string]$ProjctName_f1
        ,
        [Parameter(Mandatory=$True)]
        [ValidateNotNullOrEmpty()]
        [Alias("CertName")]
        [string]$CertName_f1
        ,
        [Parameter(Mandatory=$True)]
        [ValidateNotNullOrEmpty()]
        [Alias("CertPass")]
        [securestring]$CertPass_f1
        ,
        [Parameter(Mandatory=$True)]
        [ValidateNotNullOrEmpty()]
        [Alias("DomainName")]
        [string]$DomainName_f1
        ,
        [Parameter(Mandatory=$True)]
        [ValidateNotNullOrEmpty()]
        [Alias("CertDnsNames")]
        [string[]]$CertDnsNames_f1
    )

    $paramRequestCert = @{
        Template="AdminAbiitProjects";
        SubjectName=("CN={0}.{1}" -f $ProjctName_f1, $DomainName_f1).ToLower();
        DnsName=$CertDnsNames_f1;
        CertStoreLocation="Cert:\CurrentUser\My"
    }

    $resultRequestCert = Get-Certificate @paramRequestCert -ErrorAction Stop
    [System.Security.Cryptography.X509Certificates.X509Certificate2]$certObj = Get-Item -Path ("Cert:\CurrentUser\My\{0}" -f $resultRequestCert.Certificate.Thumbprint)
    [void]($certObj | Export-PfxCertificate -FilePath ("C:\Certificates\{0}" -f $CertName_f1) -Password $CertPass_f1 -Force -ErrorAction Stop)
    [void]($certObj | Remove-Item -Force)
    return $resultRequestCert.Certificate
}
```

Рис. 6. Листинг функции ThisScript.CreateCert (функция выпуска сертификата в центре сертификации Microsoft Certificate Authority)
Fig. 6. Listing of the ThisScript function.CreateCert (certificate issuance function in the Microsoft Certificate Authority)

Функции будут переданы имя проекта, имя файла сертификата, пароль сертификата, который был получен из конфигурации, имя домена Active Directory, к которому присоединен компьютер, на котором выполняется скрипт, а также список DNS имен, который предварительно будет отформатирован методом GetDnsNames экземпляра класса CertDnsNames Builder. Данный метод выполняет форматирование строк, хранящихся в списке свойства Data. Форматирование выполняет замену подстрок {ProjectName}, {WinDomainFqdn}, {ConsulDc Name} в обрабатываемой строке, значениями из хэш-таблицы, на которую ссылается свойство Masks (табл. 2).

Таблица 2

Хэш таблица с набором данных в свойстве Mask

Table 2

Hash table with a set of data in the Mask property

Имя ключа	Имя значения
ProjectName	TestProject
WinDomainFqdn	contoso.lan
ConsulDcName	dc1

Функция выполнит запрос на выпуск сертификата в центре сертификации с помощью командлета Get-Certificate, в случае успеха, данный командлет запишет сертификат в хранили-

ще сертификатов пользователя, после этого сертификат вместе с закрытым ключом будет экспортирован в директорию C:\Certificates в формате pfx, контейнер сертификата будет зашифрован с помощью пароля, который был получен из файла конфигурации. После экспорта сертификата в файл, сертификат будет удален из хранилища сертификатов пользователя. В данном случае хранилище сертификатов используется в качестве промежуточного хранилища. Заключительным этапом работы функции ThisScript.CreateCert является возврат объекта сертификата с типом System.Security.Cryptography.X509Certificates.X509 Certificate2.

Например, если в свойстве Mask хранится хэш-таблица с набором данных из табл. 2, тогда получим следующие результаты форматирования, указанные в табл. 3.

Таблица 3

Результаты форматирования

Table 3

Formatting results

Исходная строка	Результирующая строка
*.{WinDomainFqdn}	*.contoso.lan
*.node.consul	*.node.consul
*.node.{ConsulDcName}.consul	*.node.dc1.consul
*.{ProjectName}.service.consul	*.TestProject.service.consul
*.{ProjectName}.service.{ConsulDcName}.consul	*.TestProject.service.dc1.consul
*.{ProjectName}.query.consul	*.TestProject.query.consul

После этого выполнение скрипта вернется обратно в глобальную область и продолжит выполнение цикла. Далее будет выполнена запись в сервис каталога сервисов информации о выпущенном сертификате. Если условие в 11 строке (см. рис. 6) не сработало, т. е. сертификат был найден в массиве, на который ссылается переменная $certificatesInfoFromSm, выполнение скрипта перейдет к выполнению блока else (рис. 7), в котором будет выполнена попытка обновить сертификат.

Условие в строке 52 проверяет, что дата срока действия сертификата превышает значение, указанное в конфигурации, если это условие выполняется, будет выполнено обновление сертификата. Обновление заключается в создании нового сертификата в центре сертификации с помощью функции ThisScript.CreateCert и с последующим обновлением данных о сертификате в СКС (см. рис. 5, строки 67–82).

Далее описанные операции выполняются в цикле для каждого проекта. После завершения цикла обработки сертификатов будет выполнен метод $serviceManagerHandler.Update CertificateStatus() (рис. 8).

Этот метод не возвращает данных. Он запускает в СКС процесс обновления статусов сертификатов для записей компьютеров. К примеру, если был обновлен сертификат проекта, то у записей компьютеров изменится статус обновленного сертификата. После этого значение переменной $certificatesInfoFromSm, которая хранит массив объектов сертификатов, будет перезаписано обновленными значениями.

Далее в переменную $certsAndServers с помощью метода $serviceManagerHandler.Get ServersRequireCertificateUpdates() будет записан массив объектов, эти объекты хранят информацию о компьютерах и статусах сертификатов, принадлежащих этим компьютерам. Данный метод возвращает только те объекты компьютеров, которым требуется записать новые сертификаты или обновить старые, свойства возвращаемого объекта указаны в табл. 4.

```powershell
for ($i=0; $i -lt $projectNames.Count; $i++)
{
    [string]$projectNameInFor = $projectNames[$i]
    $certDnsNamesBuilder.Masks["ProjectName"] = $projectNameInFor

    [PSCustomObject]$certificateSm = $null
    $certificateSm = $certificatesInfoFromSm | Where-Object {$_.ProjectName -eq $projectNameInFor}

    [System.Security.Cryptography.X509Certificates.X509Certificate2]$creationCert = $null
    #region Создание сертификата ···
    #endregion
    #region Обновление сертификата
    else
    {
        if((Get-Date).AddDays($deployCertsConfig.RenewalPeriod) -ge [DateTime]$certificateSm.CertExpires) {
            $writeClass.Write(("Renew certificate for project `"{0}`":" -f $projectNameInFor), 2, $True)
            try {
                $creationCert = ThisScript.CreateCert `
                                    -ProjctName     $projectNameInFor `
                                    -CertName       $certificateSm.CertFileName `
                                    -CertPass       $deployCertsConfig.PasswordPfx `
                                    -DomainName     $fqdnDomainName `
                                    -CertDnsNames   $certDnsNamesBuilder.GetDnsNames()
                $writeClass.Write(("Request and creation of the certificate - OK"), 2, $False)
            } catch {
                $writeClass.Write($_.ToString(), 1, $False)
                continue
            }

            try {
                $isUpdateInSm = {
                    [bool]$resultWriteDataInSm = $False
                    $resultWriteDataInSm = $serviceManagerHandler.UpdateProjectCertificate(
                        $projectNameInFor, $creationCert.Thumbprint, $creationCert.NotAfter, $certificateSm.CertFileName, $certificateSm.Thumbprint)
                    return $resultWriteDataInSm
                }

                if(!$isUpdateInSm.InvokeReturnAsIs()) {
                    throw "The project is not found or has no SSL binding."
                }

                $writeClass.Write(("Writing data in SM - OK"), 2, $False)
            } catch {
                $writeClass.Write($_.ToString(), 1, $False)
            }
        }
    }
    #endregion
}
```

Рис. 7. Листинг цикла (обновление сертификата)
Fig. 7. Cycle listing (certificate renewal)

```powershell
try {
    [void]($serviceManagerHandler.UpdateCertificateStatus())
} catch {
    $writeClass.Write($_.ToString(), 1, $False)
}

$certificatesInfoFromSm = $null
try {
    $certificatesInfoFromSm = $serviceManagerHandler.GetProjectCertificates()
} catch {
    $writeClass.Write($_.ToString(), 0, $True)
}

try {
    # метод возвращает только те сертификаты и сервера, которым требуется обновление
    [PSCustomObject[]]$certsAndServers = $serviceManagerHandler.GetServersRequireCertificateUpdates() | Where-Object {$_.OSVersion -ne "Linux"}
} catch {
    $writeClass.Write($_.ToString(), 1, $False)
}
```

Рис. 8. Листинг кода метода serviceManagerHandler.UpdateCertificateStatus()
(обновление статусов сертификатов)
Fig. 8. Listing Koda method serviceManagerHandler.UpdateCertificateStatus()
(updated status certificate)

Таблица 4

Свойства объекта «компьютер-сертификат»

Table 4

Properties of the “computer-certificate” object

Имя свойства	Тип	Пример значения	Описание
certId	int	100	Числовой идентификатор сертификата
serverName	string	test-server1	Имя сервера
status	int	1	Числовой статус сертификата для сервера

На основе объектов «компьютер-сертификат» (см. табл. 4.) будут сформированы новые объекты. Данные объекты будут группировать информацию о сертификатах по каждому компьютеру (табл. 5), назовем этот объект «сертификаты компьютера».

Таблица 5

Свойства объекта «сертификаты компьютера»

Table 5

Properties of the “Computer certificates” object

Имя свойства	Тип	Пример значения	Описание
ComputerName	string	test-server1	Имя компьютера
CertsInfo	object[]	–	Информация о сертификатах, принадлежащих компьютеру
HasErrors	bool	false	Возникали ли ошибки при подключении к компьютеру.

Создание объектов «сертификаты компьютера» будет выполняться с помощью циклической конструкции, листинг которой приведен на рис. 9.

```
[PSCustomObject[]]$certsOnServers = $null

if($certsAndServers.Count -gt 0) {
    foreach ($srvObj in ($certsAndServers | Group-Object -Property ServerName | Select-Object -Property Name, Group)) {
        [PSCustomObject[]]$certsInfo = $null
        foreach($cert in $srvObj.Group) {
            $certsInfo += [PSCustomObject]([ordered]@{
                CertID=$cert.CertID;
                CertStatus=$cert.Status;
                DeployStatus=$False;
                UpdateStatus="NotRequired"
            })
        }

        $certsOnServers += [PSCustomObject]([ordered]@{
            ComputerName=$srvObj.Name;
            CertsInfo=$certsInfo;
            HasErrors=$False
        })
    }
} elseif ($writeClass.ErrorCounter -gt 0) {
    $writeClass.Push()
    exit 0
} else {
    exit 0
}
```

Рис. 9. Листинг блока кода (формирование массива объектов соответствия сервер-сертификаты)
Fig. 9. Code block listing (forming an array of server-certificate compliance objects)

Новые объекты «сертификаты компьютера» помещаются в массив, на который ссылается переменная $certsOnServers, далее объекты этого массива будут обработаны в цикле, переменная цикла имеет название $serverObj. Вначале будет выполнена попытка создать удаленную powershell сессию для компьютера, имя которого указанно в свойстве $serverObj. ComputerName. Если по каким-то причинам эта операция завершится с ошибкой, то свойству $serverObj.HasErrors будет присвоено значение $true, выполнение цикла сразу вернется к началу, и начнется обработка следующего элемента массива.

Если операция создания удаленной powershell сессии будет завершена успешно, то переменная $psSession будет инициализирована объектным представлением этой сессии, которая в дальнейшем будет использоваться для подключения к уделенному компьютеру, для копирования сертификатов проектов и выполнении операций над этими сертификатами на удаленном компьютере.

Во вложенном цикле выполнятся перебор объектов «информация о сертификате», хранящихся в свойстве объекта $serverObj.CertsInfo. Каждый перебираемый объект помещается в переменную цикла $srvCertObj. В первую очередь выполняется копирование сертификата с компьютера, на котором выполняется данный скрипт, на удаленный компьютер во временную директорию. Копирование выполняется с помощью удаленной powershell сессии, созданной ранее.

Если копирование было выполнено успешно, на удаленном компьютере для скопированного сертификата будет выполнен ряд операций. Вначале будет выполнена проверка того, что скопированный сертификат отсутствует в хранилище сертификатов компьютера, и если сертификат отсутствует, то будет произведена установка сертификата из файла в хранилище сертификатов. Установка выполняется с помощью командлета Import-PfxCertificate. После успешной установки сертификата в хранилище данному сертификату необходимо применить ACL для закрытого ключа, такие же как и у других сертификатов для этого проекта, при условии, что эти сертификаты существуют. Access Control List или ACL – список управления доступом, который определяет, кто или что может получать доступ к объекту (программе, процессу или файлу) и какие именно операции разрешено или запрещено выполнять субъекту (пользователю, группе пользователей). Это необходимо, чтобы в дальнейшем приложение, для которого предназначается этот сертификат, смогло создать HTTPS прослушиватель. Агрегация ACL из других сертифкатов и дальнейшее применение агрегированных ACL к новому сертификату выполняется с помощью функции Merge-CertificateAcl. Заключительной операцией является удаление сертификата из временной директории. После этого свойству $srvCertObj.DeployStatus присваивается значение $true.

Затем проверяется статус сертификата, который хранится в свойстве $srvCertObj.Cert Status (информацию о нем можно найти в табл. 5). Если статус сертификата равен 2, то сертификату требуется обновление. Этот статус присваивается сертификату, если проекту ранее уже был выдан сертификат и взамен устаревшего сертификата был выпущен новый сертификат. Рассмотрим логику для тех сертификатов, которым требуется обновление. Вначале с помощью функции Move-CertificateAcl для закрытого ключа нового сертификата будут применены такие же ACL, как и для закрытого ключа старого сертификата. Это очень похоже на операцию, выполняемую функцией Merge-CertificateAcl, рассмотренной ранее, но в отличие от нее ACL копируются из определенного сертификата, тогда как Merge-CertificateAcl делает агрегацию ACL по всем сертификатам проекта. Далее выполняется поиск старого отпечатка сертификата в веб-сервере HTTP.sys для HTTPS прослушивателей. Если HTTPS прослушиватель со старым отпечатком сертификата будет найден, то для этого прослушивателя в вебсервере HTTP.sys будет заменен отпечаток сертификата на новый. После этой операции веб-сервер HTTP.sys сможет найти и загрузить новый сертификат из хранилища по обновленной информации об отпечатке.

Если операции установки и обновления сертификата прошли успешно, то выполняется запрос в СКС для изменения статуса сертификата для проекта на определенном сервере. По-

сле выполнения запроса сервиса каталога сервисов изменит статус сертификата на 1, что будет означать, что сертификат находится в актуальном состоянии, и при последующих запусках скрипта он не будет обрабатываться до тех пор, пока у сертификата не истечет срок действия. По завершении обработки всех сертификатов во внутреннем цикле будет закрыта и удалена PowerShell-сессия для освобождения ресурсов. Далее внешний цикл повторяется для остальных компьютеров.

Завершающим этапом работы является уведомление системных администраторов о результатах работы скрипта. Уведомления отправляются, если были созданы новые сертификаты или обновлены старые, а также если во время работы возникали ошибки. Данные уведомления отправляются в специальный чат корпоративного мессенджера.

Заключение

Для автоматизации процесса создания и распространения SSL-сертификатов на серверы было предложено разработать скрипт, который выполняется на одном из серверов управления. Его работа заключается в получении информации из каталога сервисов; выпуске новых сертификатов в центре сертификации; обновлении устаревающих сертификатов в центре сертификации; распространении сертификатов сервисов по серверам; обновлении информации о сертификатах в каталоге сервисов.

Список литературы

1. **Айвенс К.** Внедрение, управление и поддержка сетевой инфраструктуры MS Windows Server 2003: Учеб. пособие. М.: Интернет-университет информационных технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021. 914 с. URL: http://www.iprbookshop.ru/102009.html
2. **Коньков К. А., Карпов В. Е.** Основы операционных систем: Учебник. М.: Интернет-университет информационных технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021. 346 с. URL: http://www.iprbookshop.ru/102031.html
3. **Костюк А. И., Беспалов Д. А.** Администрирование баз данных и компьютерных сетей: Учеб. пособие. Ростов н/Д; Таганрог: Изд-во ЮФУ, 2020. 127 с. URL: http://www.iprbookshop.ru/107941.html
4. **Мейер Б.** Инструменты, алгоритмы и структуры данных: Учеб. пособие. М.: Интернет-университет информационных технологий (ИНТУИТ), Ай Пи Ар Медиа, 2021. 540 с. URL: http://www.iprbookshop.ru/102012.html

References

1. **Ivens K.** Implementation, management and support of the network infrastructure of MS Windows Server 2003: tutorial. Moscow, Internet University of Information Technologies (INTUIT), Ai Pi Ar Media, 2021, 914 p. (in Russ.) URL: http://www.iprbookshop.ru/102009.html
2. **Konkov K. A., Karpov V. E.** Fundamentals of operating systems: textbook. Moscow, Internet University of Information Technologies (INTUIT), Ai Pi Ar Media, 2021, 346 p. (in Russ.) URL: http://www.iprbookshop.ru/102031.html. – EBS "IPRbooks"
3. **Kostyuk A. I., Bespalov D. A.** Administration of databases and computer networks: tutorial. Rostov on Don, Taganrog, Publishing House of the Southern Federal University, 2020, 127 p. (in Russ.) URL: http://www.iprbookshop.ru/107941.html
4. **Meyer B.** Tools, algorithms and data structures: tutorial. Moscow, Internet University of Information Technologies (INTUIT), Ai Pi Ar Media, 2021, 540 p. (in Russ.) URL: http://www.iprbookshop.ru/102012.html

Информация об авторах

Федькова Надежда Александровна, кандидат экономических наук, доцент
Author ID 724535
Федьков Дмитрий Андреевич, студент магистратуры

Information about the Authors

Fedkova Nadezhda Alexandrovna, Candidate of Economic Sciences
Author ID 724535
Fedkov Dmitry Andreevich, Master's Student

Статья поступила в редакцию 01.12.2021;
одобрена после рецензирования 01.02.2022; *принята к публикации 01.02.2022*
The article was submitted 01.12.2021;
approved after reviewing 01.02.2022; *accepted for publication 01.02.2022*

Правила оформления текста рукописи

Авторы представляют статьи на русском или английском языке объемом от 0,5 авторского листа (20 тыс. знаков) до 1 авторского листа (40 тыс. знаков), включая иллюстрации (1 иллюстрация форматом 190 × 270 мм = 1/6 авторского листа, или 6,7 тыс. знаков). Публикации, превышающие указанный объем, допускаются к рассмотрению только после индивидуального согласования с редакцией журнала.

Текст рукописи должен быть представлен в редколлегию в виде файла MS Word (.doc, .docx). Гарнитура Times New Roman, размер шрифта 11, межстрочный интервал 1, размеры полей – стандартные значения текстового редактора. Форматирование – выравнивание по ширине страницы, переносы слов включены, каждый новый абзац начинается с красной строки. Не допускается ручное форматирование абзацев (пробелами, лишними переводами строк, разрывами страниц).

Структура статьи

- Индекс УДК (универсальной десятичной классификации). Выравнивание по левому краю
- Название статьи. Выравнивание по центру, полужирный шрифт
- ФИО авторов (полностью). Выравнивание по центру, полужирный шрифт
- Места работы всех авторов. Выравнивание по центру, курсив
- Адреса электронной почты, ORCID авторов
- Аннотация статьи
- Ключевые слова, не более 10
- Благодарности, сведения о финансовой поддержке
- Название статьи **на английском языке**. Выравнивание по центру, полужирный шрифт
- ФИО авторов **на английском языке** (полностью). Выравнивание по центру, полужирный шрифт
- Места работы авторов **на английском языке**. Выравнивание по центру, курсив
- Аннотация статьи **на английском языке (Abstract)**, 200–250 слов
- Ключевые слова **на английском языке (Keywords)**, не более 10
- Благодарности, сведения о финансовой поддержке **на английском языке**, если есть соответствующий раздел на русском языке **(Acknowledgements)**
- Основной текст
- Список литературы / **References**
- Сведения об авторах

Требования к оформлению основного текста и иллюстративных материалов

Основной текст должен быть представлен в структурированном виде, рекомендуется использовать подзаголовки – например: Введение, Методика..., Выводы, Результаты, Заключение.

Подзаголовки отделяются и набираются полужирным шрифтом. В целях выделения частей текста и отдельных слов и словосочетаний допускается использование курсива или полужирного шрифта. Подчеркивание, разрядка, изменение основного кегля и выделение цветом не используются.

Иллюстрации к рукописи статьи должны быть приложены в виде отдельных файлов. При этом в тексте должно содержаться включенное изображение с указанием имени файла. Все иллюстрации, содержащие схемы, графики, алгоритмы и т. п., должны быть представлены в векторном виде (.ai, .eps, .cdr). Скриншоты и другие растровые изображения должны быть представлены в максимально высоком качестве, без каких-либо потерь и искажений (.jpg, .tif). Все иллюстрации должны иметь подрисуночную подпись – свое название. Надписи к таблицам и подписи к иллюстрациям приводятся **на двух языках (русском и английском).**

Примеры:

Рис. 1. Диаграмма производительности...
Fig. 1. Performance diagram...

Таблица 1

Сравнение алгоритмов...

Table 1

Comparison of algorithms...

Нумерация последовательная и неразрывная от начала статьи. Не допускается использование других наименований, кроме «Рис.» / «Fig.», «Таблица» / «Table», и усложнение нумерации (например, «Рис. 3.2.»). Ссылка на иллюстрацию в тексте должна быть приведена в круглых скобках, например: (рис. 1), (табл. 1).

Формулы должны быть набраны с использованием редактора MathType либо встроенного редактора формул MS Word. Кегль основных символов – 11, греческие символы набираются прямым шрифтом, латинские – курсивом. Нумеруются только те формулы, на которые автор ссылается в тексте.

Abstract

Аннотация статьи на английском языке (Abstract) не должна быть дословным переводом русскоязычной аннотации. Раздел Abstract, как и основной текст, должен быть структурирован, в нем должно содержаться описание цели работы, методов исследования, научной значимости, выводов / результатов. Требуется качественный перевод на английский язык (при необходимости просим авторов обращаться к профессиональным переводчикам). **Объем Abstract 200–250 слов.**

Список литературы / References

Список литературы и список литературы на английском языке (References) размещаются в общем разделе. Рекомендуемое количество цитируемых в статье источников – не менее 10, в список желательно включать ссылки на актуальные работы по теме исследования, особенно в иностранных периодических изданиях.

В тексте статьи ссылки на литературу указываются цифрами в квадратных скобках, при необходимости указываются номера страниц, например: [2; 3. С. 15].

Список литературы нумеруется в порядке цитирования и оформляется в соответствии с ГОСТ Р 7.0.5-2008 на библиографическое описание (знаки тире в описании опускаются). Ссылки на неопубликованные работы, а также на Интернет-ресурсы (кроме электронных изданий, поддающихся библиографическому описанию) оформляются в виде сноски.

В Список литературы ссылки на источники следует включать на оригинальном языке опубликования. Каждый источник должен быть также оформлен на английском языке (References) по международному стандарту для публикаций в области информатики IEEE Style со следующими отличиями:

- инициалы авторов указываются после фамилии;
- название статьи не берется в кавычки, отделяется точкой;
- отсутствует союз «and» перед фамилией последнего автора;
- в диапазоне страниц – удвоенная «p» (например, «pp. 2–9»);

- год издания указывается после места издания (для книг) и сразу после названия журнала (для периодики).

Перевод источника на английский язык:

- если источник имеет выходные данные на английском языке, то для формирования References **следует использовать именно эти данные**;
- если оригинальная публикация не содержит выходных данных на английском языке, то допускается транслитерация названия материала на латинский алфавит в сочетании с переводом на английский язык в квадратных скобках. В конце описания указывается, на каком языке написана эта работа, например, (in Russ.). При транслитерации можно воспользоваться Интернет-ресурсом http://ru.translit.ru/, рекомендуется выбрать стандарт BSI. Место издания не транслитерируется, указывается полностью на английском языке, например: Moscow. Название издательства / издателя, как правило, транслитерируется. Для журналов, у которых есть официальное название на английском языке, – использовать его (проверить на сайте журнала, или, например, в библиотеке WorldCat), если названия на английском языке нет, использовать транслитерацию по системе BSI. Не следует самостоятельно переводить названия журналов.

Если у цитируемого источника есть **цифровой идентификатор DOI** (https://search.crossref.org), его требуется обязательно указывать в конце библиографической ссылки.

Примеры оформления ссылок. Каждый источник в том же пункте дублируется на английском языке (References).

Источник на русском языке, перевод на английский доступен в метаданных статьи

1. **Журавлев С. С., Рудометов С. В., Окольнишников В. В., Шакиров С. Р.** Применение модельно-ориентированного проектирования к созданию АСУ ТП опасных промышленных объектов // Вестник НГУ. Серия: Информационные технологии. 2018. Т. 16, № 4. С. 56–67. DOI 10.25205/1818-7900-2018-16-4-56-67

Zhuravlev S. S., Rudometov S. V., Okolnishnikov V. V., Shakirov S. R. Model-Based Design Approach for Development Process Control Systems of Hazardous Industrial Facilities. *Vestnik NSU. Series: Information Technologies*, 2018, vol. 16, no. 4, pp. 56–67. (in Russ.) DOI 10.25205/1818-7900-2018-16-4-56-67

Источник на английском языке. Оформляем согласно требованиям для References. Приводим только 1 раз.

2. **Telnov V. I.** Optimization of the Beam Crossing Angle at the ILC for E + e- and yy Collisions. *Journal of Instrumentation*, 2018, vol. 13, no. 03, pp. P03020–P03020. DOI 10.1088/1748-0221/13/03/p03020

Метаданные источника доступны только на русском языке

3. **Жижимов О. Л., Федотов А. М., Шокин Ю. И.** Технологическая платформа массовой интеграции гетерогенных данных // Вестник НГУ. Серия: Информационные технологии. 2013. Т. 11, вып. 1. С. 24–41.

Zhizhimov O. L., Fedotov A. M., Shokin Yu. I. Tekhnologicheskaya platforma massovoi integratsii geterogennykh dannykh [Technology Platform for the Mass Integration of Heterogeneous Data]. *Vestnik NSU. Series: Information Technologies*, 2013, vol. 11, no. 1, pp. 24–41. (in Russ.)

Сведения об авторах

Последний раздел статьи – информация об авторе / авторах **на русском и английском языках**:

- ФИО полностью, ученая степень, ученое звание;
- идентификаторы автора, такие как ResearcherID (всем авторам рекомендуется использовать данные сервисы для ведения актуального списка своих публикаций);

- контактный телефон (не публикуется).

Если статья представляется на английском языке, необходимо приложить перевод на русский язык названия, аннотации, ключевых слов, сведений об авторе.

Доставка материалов

Материалы предоставляются в редакцию по электронной почте inftech@vestnik.nsu.ru.

Порядок рецензирования

Все статьи сначала проходят проверку на заимствование и только после этого отправляются на рецензирование. Редакционный совет не допускает к публикации материал, если имеется достаточно оснований полагать, что он является плагиатом.

Тип рецензирования статей – двухуровневое, одностороннее анонимное («слепое»).

Для каждой статьи редколлегией выбираются рецензенты, научная деятельность которых связана с темой представленного материала. Ответственный секретарь журнала обращается к ним с просьбой дать экспертную оценку статье либо помочь организовать рецензирование.

Рецензии для журнала «Вестник НГУ. Серия: Информационные технологии» составляются по единой схеме и подразумевают оценку по следующим критериям: соответствие тематике журнала, оригинальность и значимость результатов, качество изложения материала.

Заполненный бланк рецензии высылается на электронный адрес редакции. В зависимости от экспертных заключений статья может быть принята редакционным советом к опубликованию, рекомендована автору к доработке (с последующим повторным рецензированием либо без него) или отклонена (с предоставлением автору мотивированного отказа). Автору на электронный адрес высылается текст рецензии без указания ФИО рецензента и его контактных данных.

Все рецензии хранятся в редакции журнала не менее 5 лет. Редколлегия журнала обязуется при поступлении соответствующего запроса направлять копии рецензий в Министерство науки и высшего образования Российской Федерации.