УДК 004.056 Вестник УрФО № 1(55) / 2025, с. 21–28

Фатин А. Д.

DOI: 10.14529/secur250102

ПОСТРОЕНИЕ МОДЕЛИ АДАПТИВНОСТИ КИБЕРФИЗИЧЕСКИХ СИСТЕМ: КЛАСТЕРИЗАЦИЯ, ИЗОЛЯЦИЯ И РЕКОНФИГУРАЦИЯ

Данная работа является продолжением исследования «Построение модели адаптивности киберфизических систем: функционирование и детектирование». Текущая часть фокусируется на решении задач кластеризации и изоляции узлов в киберфизических системах с помощью алгоритма пчелиной колонии (ABC) с последующей реконфигурацией. Используется адаптированный алгоритм ABC для определения оптимальной структуры кластеров, что позволяет минимизировать воздействие возможных аномалий на систему. Затем применяются полученные результаты кластеризации для эффективной изоляции аномальных узлов с последующей реконфигурацией системы для поддержания ее работоспособности.

Ключевые слова: *КФС, адаптивность, кластеризация, изоляция узлов, реконфигурация.*

Fatin A. D.

BUILDING A MODEL OF ADAPTABILITY OF CYBERPHYSICAL SYSTEMS: CLUSTERIZATION, ISOLATION AND RECONFIGURATION

This work is a continuation of the study "Building a model of adaptability of cyberphysical systems: functioning and detection." The current part focuses on solving clustering and node isolation problems in cyber-physical systems using the Bee Colony (ABC) algorithm followed by reconfiguration. An adapted ABC algorithm is used to determine the optimal cluster structure, which minimizes the impact of possible anomalies on the system. The obtained clustering results are then used to effectively isolate anomalous nodes, followed by reconfiguration of the system to maintain its functionality.

Keywords: *CPS, adaptivity, clustering, node isolation, reconfiguration.*

Введение

В ходе исследования [1] было принято к сведению, что адекватное реагирование на аномалии в киберфизических системах (согласно [2–5]) требует не только их точной диагностики, но и разработки стратегий, позволяющих минимизировать последствия данных событий. В первой части работы были рассмотрены методы обнаружения и анализа аномальных состояний. Теперь же, во второй части, акцент смещен на задачи кластеризации и изоляции узлов, которые являются критическими для поддержания целостности и стабильности системы при наличии потенциальных угроз.

В качестве алгоритмов кластеризации зачастую используются следующие подходы: CMNC [6], VCLANC [7], ANCA [8], метод кластеризации на основе пчелиной колонии (ABC) [9], CFCN [10] и др.

Алгоритм пчелиной колонии (ABC), изначально разработанный для решения задач оптимизации, в данном контексте адаптируется для определения наиболее устойчивой структуры кластеров узлов, что позволяет системе принимать взвешенные решения об изоляции отдельных узлов или групп узлов, которые могут представлять собой угрозу для функционирования всей сущности. При этом ключевым аспектом является сохранение работоспособности системы и минимизация влияния изоляционных мер на ее производительность.

В работе детально рассматривается процесс кластеризации, начиная с инициализации и заканчивая конвергенцией алгоритма, а также обсуждается, как каждый этап влияет на формирование оптимальных кластеров, аналогично работам [11–14]. Также вводится математическая модель, позволяющая количественно оценить качество разбиения и его пригодность для последующих действий по изоляции узлов. Особое внимание уделяется анализу взаимосвязи между результатами задачи кластеризации и последующей изоляцией узлов, поскольку правильное понимание этой взаимосвязи критично для разработки эффективной стратегии обеспечения безопасности системы.

В заключение введения подчеркивается, что предложенный подход не только повышает устойчивость системы к аномалиям, но и обеспечивает возможность ее адаптации и восстановления после изоляционных мер, что является неотъемлемой частью создания надежных и устойчивых киберфизических систем.

Решение задачи 3: кластеризация

Для решения задачи 3 [1] рекомендуется использовать ABC. Алгоритм пчелиной колонии (ABC) является методом оптимизации, который имитирует поведение пчел при поиске источников пищи. Для кластеризации системы адаптируем ABC для идентификации групп узлов, которые могут быть изолированы с минимальными потерями для системы.

Взаимосвязь решения задачи 2 с задачей 3

Алгоритм пчелиной колонии для решения этой задачи будет итеративно обновлять разбиение C и центроиды μ_k через процесс, имитирующий поведение пчел при поиске пищи:

1. Инициализация

Случайным образом производится распределение узлов по кластерам для начального приближения центроидов.

2. Разведка

«Пчелы-разведчики» исследуют пространство решений, предлагая новые разбиения.

3. Вербовка и эксплуатация

«Пчелы-рабочие» сосредотачиваются на наиболее перспективных решениях, улучшая их и уточняя центроиды.

4. Оценка

Расчет качества каждого разбиения с использованием целевой функции.

5. Фаза обновления

Удаление менее успешных решений и замена их новыми, генерируемыми «пчелами-разведчиками».

6. Конвергенция (сходимость)

Итерирование до тех пор, пока не будет достигнут критерий остановки (например, заданное количество итераций или достижение определенного уровня качества разбиения).

Данный процесс позволяет адаптивно исследовать пространство решений и находить оптимальное или близкое к оптимальному разбиение множества узлов на кластеры с учетом аномальных узлов.

Математическая модель

Пусть X — множество векторов состояний узлов системы, а A — множество индикаторов аномалий, полученных из задачи 2. Тогда задача кластеризации может быть сформулирована как оптимизационная проблема: найти такое разбиение C множества узлов X,

которое минимизирует функцию потерь L, учитывая аномальность некоторых узлов.

Целевая функция:

Минимизировать функцию потерь L, которая может включать внутрикластерное расстояние и, возможно, штраф за включение аномальных узлов в кластеры:

$$\mathrm{L(C)} = \sum_{\mathrm{k=1}}^{\mathrm{K}} \sum_{\mathrm{x_i} \in \mathrm{C_k}} \mathrm{d(x_i, \mu_k)} + \lambda \sum_{\mathrm{x_i} \in \mathrm{A}} \phi(\mathrm{x_i, C}), \quad (1)$$

где:

C — $C = \{C_1, C_1, \ldots, C_K\}$ — разбиение множества X на K кластеров,

C_k — узлы в кластере k,

x_i — вектор состояния узла i,

μ_k — центроид кластера C_k,

d — функция расстояния между вектором состояния узла x_i и центроидом кластера μ_k (например, евклидово расстояние),

λ — (коэффициент) параметр регуляризации, который контролирует влияние аномальных узлов на целевую функцию,

ϕ — функция штрафа за аномалии,

A — A=a1,a2,…,an представляет собой множество аномалий, где ai является индикатором (например, бинарным значением), который указывает, является ли i-й узел аномальным или нет,

K — количество кластеров, на которое нужно разделить множество X.

Ограничения кластеризации:

Каждый узел должен принадлежать ровно одному кластеру:

$$\forall i, \exists! \, k : x_i \in C_k, \quad (2)$$

и количество кластеров K может быть задано заранее или определено в процессе.

Для упрощения изменим вышеприведенную формулу целевой функции на следующую:

$$min_{(C,\mu_k)} \sum_{k=1}^{K} \sum_{x_i \in C_k} d(x_i, \mu_k) + \lambda \sum_{i=1}^{n} a_i \cdot \psi(x_i, C), \quad (3)$$

Решение задачи 4: изоляция узлов

Для решения задачи 4 будем использовать результаты задачи 3, чтобы определить, какие узлы следует изолировать для минимизации влияния аномалий на систему. Подход основан на стремлении к изоляции только тех узлов, которые являются источниками аномалий, и таким образом, чтобы минимизировать воздействие на функционирование системы.

Процесс оптимизации будет происходить по следующему алгоритму:

1. Инициализация переменных и установление начальных значений весов и коэффициентов.

2. Вычисление целевой функции I для текущего набора изолированных узлов.

3. Применение методов оптимизации для нахождения оптимального набора узлов для изоляции, удовлетворяющего ограничениям и минимизирующего целевую функцию.

4. Повторение шагов 2 и 3 до достижения критерия остановки (например, отсутствие улучшения целевой функции или достижение максимального количества итераций).

Ограничения:

1. Каждый узел может находиться лишь в одном состоянии изоляции:

$$\forall i, \gamma_i \in 0,1. \quad (4)$$

2. Изоляция узла возможна только если он аномален:

$$\forall i, \gamma_i = 1 \Rightarrow x_i \text{ может быть изолирован.} \quad (5)$$

3. Необходимо учитывать взаимодействие узлов внутри кластеров и между кластерами, чтобы минимизировать воздействие изоляции на систему.

Математическая модель

Целевая функция:

Минимизировать функцию воздействия изоляции I, которая учитывает потери функциональности системы из-за изоляции узлов и возможные штрафы за нарушение целостности кластеров:

$$\min_{\mathrm{I}} \left(\sum_{\mathrm{i} \in \mathrm{I}} \mathrm{w_i} \cdot \gamma_\mathrm{i} + \delta \sum_{\mathrm{k=1}}^{\mathrm{K}} \eta(\mathrm{C_k, I}) \right), \quad (6)$$

где:

w_i — вес узла i, отражающий его важность для системы.

γ_i — индикатор аномалии узла i, полученный из задачи 2 (1, если узел аномален, и 0 в противном случае).

δ — коэффициент, который контролирует влияние нарушения целостности кластеров на целевую функцию.

$\eta(C_k,I)$ — функция штрафа за изоляцию узлов в кластере C_k, которая увеличивается, если узлы из одного кластера изолированы несоразмерно.

На выходе решения получаем набор узлов для изоляции, который минимизирует воздействие на систему и учитывает аномальность узлов и целостность кластеров.

Решение задачи 5: реконфигурация

После изоляции узлов система должна быть реконфигурирована, чтобы продолжить функционировать без изолированных узлов.

Шаги решения:

1. Произвести переоценку сетевой топологии без изолированных узлов.

2. Определить новые пути и связи для маршрутизации задач и потоков данных.

3. Распределить ресурсы и нагрузку между оставшимися узлами.

4. Применить изменения в системе и проверить ее работоспособность.

Для решения задачи реконфигурации системы можно воспользоваться системой оркестрации контейнеров, таких как, например, Kubernetes (k8s). В последних, зачастую, существуют механизмы автоматического масштабирования, самовосстановления и управления конфигурацией.

Математическая модель

Пусть S будет множеством всех узлов в системе, а I – множеством изолированных узлов после решения задачи 4 (изоляция узлов). Тогда S'=S\I представляет собой множество узлов, оставшихся после изоляции.

1. Оценка новой топологии:

Определить новую матрицу связности C' для S', где каждый элемент c'ij указывает на наличие связи между узлами i и j в новой топологии.

2. Развертывание копий изолированных узлов:

Для каждого изолированного узла $i \in I$, найти подходящий ресурс в оркестраторе для развертывания его незараженной копии. Пусть R будет множеством всех доступных ресурсов в кластере оркестратора, тогда функция развертывания может быть описана как:

$$\text{Deploy: } I \times R \rightarrow S'', \tag{7}$$

где S'' – множество новых узлов, которые заменяют изолированные.

3. Распределение нагрузки:

Необходимо перераспределить нагрузку и задачи изолированных узлов на их новые копии. Пусть L будет функцией, которая отображает нагрузку li от изолированного узла i к новому узлу j:

$$L: I \times S'' \rightarrow R^+, \tag{8}$$

где R^+ – множество положительных вещественных чисел, представляющих нагрузку.

4. Обновление конфигурации:

Далее необходимо выполнить конфигурацию системы для отражения изменений в топологии и нагрузке. Для каждого нового узла $j \in S''$, следует обновить конфигурационные файлы Kj в оркестраторе:

$$Kj = \text{UpdateConfig}(j, C', L). \tag{9}$$

5. Применение изменений:

Применить изменения в оркестраторе, чтобы активировать новую конфигурацию:

$$\text{ApplyChanges}(Kj), \tag{10}$$

где ApplyChanges – процедура, которая запускает или обновляет поды, сервисы и другие ресурсы в оркестраторе согласно новой конфигурации.

6. Итоговая система:

После реконфигурации Sfinal будет объединением оставшихся и новых узлов:

$$\text{Sfinal} = S' \cup S''. \tag{11}$$

Улучшение модели в контексте применения нейронных сетей

Определение количества входных нейронов и размерности многомерного временного ряда (тривиальный случай)

Чтобы определить количество входных узлов (нейронов) для нейронной сети, необходимо знать размерность вектора X(t), который используется как входные данные для нейронной сети. Размерность этого вектора определяется количеством параметров в векторе состояния узла xi(t) и количеством параметров в векторе связности ci(t), умноженным на количество узлов N в системе.

Если предположить (рассматривая тривиальный случай с однотипными устройства), что каждый вектор состояния узла xi(t) имеет размерность p, а вектор связности ci(t) имеет размерность q (которая может быть равна N, если мы рассматриваем связь каждого узла со всеми остальными

узлами в сети), то объединенный вектор для одного узла $zi(t)$ будет иметь размерность $p+q$.

Таким образом, общее количество входных узлов для нейронной сети, соответствующее многомерному временному ряду $X(t)$, будет равно:

$$(\text{Количество входных узлов} = N\times(p+q), \quad (12)$$

где:

N – количество узлов в системе,

p – количество параметров состояния каждого узла,

q – количество параметров связности каждого узла (часто равно N, но может быть меньше, если сеть разреженная).

Данное число будет равно количеству входных нейронов, которые должны быть в первом слое нейронной сети, чтобы она могла обрабатывать входные данные, представленные в виде многомерного временного ряда $X(t)$.

Определение количества входных нейронов и размерности многомерного временного ряда (общий случай)

Однако, стоит отметить тот факт, что в общем случае размерность вектора состояния узла $xi(t)$ и размерность вектора связности $ci(t)$ могут быть для каждого узла произвольными.

Если размерность вектора состояния узла $xi(t)$ и размерность вектора связности $ci(t)$ могут быть разными для каждого (или некоторого количества) узла/узлов, то для определения количества входных узлов (нейронов) для нейронной сети потребуется слегка изменить подход.

Для каждого узла vi необходимо определить размерность вектора состояния pi и размерность вектора связности qi.

Просуммируем размерности векторов состояния и связности для всех узлов, чтобы получить общее количество параметров для всей системы. Таким образом формула «Количество входных узлов» выше будет иметь вид:

$$(\text{Количество входных узлов} = \sum_{i=1}^{N}(pi + qi), \quad (13)$$

где:

N – общее количество узлов в системе,

pi – размерность вектора состояния узла vi,

qi – размерность вектора связности узла vi.

Определение количества входных нейронов и размерности многомерного временного ряда (практический случай)

На практике, конечно, конфигурация части узлов будет повторяться, а значит, формула «Количество входных узлов» выше будет иметь следующий вид:

$$(\text{Количество входных узлов} = \sum_{i=1}^{M}(pi + qi)fi, \quad (14)$$

где:

M – количество семейств узлов, где каждое семейство включает в себя одинаковые типы узлов.

fi – количество узлов в i-м семействе.

pi и qi – размерности векторов, характерные для семейства fi.

Это общее количество параметров будет равно количеству входных нейронов, необходимых для первого слоя нейронной сети.

Для наглядности рассмотрим следующий пример.

Предположим, имеется система из трех узлов, где:

Семейство 1 имеет вектор состояния размерности = 4, вектор связности размерности = 2 и 3 узла.

Семейство 2 имеет вектор состояния размерности = 3, вектор связности размерности = 3 и 2 узла.

Семейство 3 имеет вектор состояния размерности = 5, вектор связности размерности = 4 и 5 узлов.

Тогда общее количество входных нейронов будет:

Количество входных узлов = $(4+2)\cdot3+$ $+(3+3)\cdot2+(5+4)\cdot5=75$

Таким образом, первый слой нейронной сети должен иметь 75 входных нейронов.

Также стоит отметить, что при работе с нейронными сетями и многомерными временными рядами, где размерности могут варьироваться, часто используются архитектуры, способные обрабатывать переменные размерности входных данных, такие как рекуррентные нейронные сети (RNN) или трансформеры, которые могут обрабатывать последовательности переменной длины. В таких случаях может потребоваться использо-

вать дополнительные техники предобработки данных, такие как padding (в данном случае дополнение нулями) или attention mechanisms, которые позволяют модели определять важность различных частей входных данных.

Использование параллельных нейронных сетей

Использование отдельной нейронной сети для каждого типа узлов может быть эффективным подходом, особенно если узлы различаются по своим функциям, характеристикам или типам данных.

Преимущества данного решения:

1. Специализация:

Каждая нейронная сеть может специализироваться на обработке данных определенного типа узлов, что позволяет более точно настраивать параметры и структуру сети под конкретные особенности данных.

2. Упрощение архитектуры:

Архитектура каждой нейронной сети может быть упрощена, поскольку она будет обрабатывать более однородный набор данных, что может снизить сложность модели и ускорить обучение.

3. Параллелизм:

Отдельные нейронные сети могут обучаться параллельно, что может существенно ускорить процесс обучения и оптимизации.

4. Модульность:

За счет реализации модульного подхода возможно упрощение тестирования, обновления, отладки, донастройки, а также масштабирования сетей, в т.ч. дополнительно упрощается введение нового класса устройств и модульное дообучение нужных сетей.

5. Гибкость управление ресурсами:

За счет возможности отдельного конфигурирования каждой из сетей появляется дополнительная вариативность по их профилированию, что позволяет упростить настройку сети по своему конечному назначению и, соответственно, агрегировать сети по требованиям к вычислительным ресурсам в отдельные кластеры.

Недостатки данного решения:

1. Усложнение интеграция:

При обработке данных нейронными сетями, выполняемой параллельно, следует учесть потенциальную возможность либо рассинхронизации выходных данных (при использовании разных моделей или различных весов в одинаковых моделях), либо заложить дополнительные ресурсы на дублирование работы или резервный ввод временных ресурсов системы в момент простоя нейронных сетей при дообучении одной из сетей и последующем дублировании их на старые модули.

2. Усложнение управления и мониторинга:

В случае работы последовательных моделей, вывод в которых напрямую используется в качестве ввода на следующем этапе вычислений, могут возникнуть проблемы при согласовании управления данными, вследствие чего могут потребоваться дополнительные средства мониторинга.

Использование отдельных нейронных сетей, работающих в параллельном или смешанном режиме, для разных типов узлов сети может иметь место, если это действительно улучшает эффективность работы системы и/или упрощает процесс обучения или обновления модели(ей). Однако стоит учитывать тот факт, что при разделении нейронных сетей на отдельные несвязанные сервисы возрастает и риск потери косвенных связей, так как. финальная сеть, агрегирующая данные, полученные из параллельных источников, может не уметь восстанавливать косвенные связи, а сами параллельные нейронные сети могут терять часть косвенных связей за счет неполной обработки данных.

Исходя из вышесказанного рекомендуется (при наличии мощностей) остановиться на монолитной нейронной сети для сохранения должного уровня связности данных и приемлемого уровня производительности системы.

Вывод

В работе были подробно расписаны математические модели, включающие в себя решения частных задач кластеризации компьютерных сетей, изоляции аномальный узлов и реконфигурации целевой киберфизической системы. Данные задачи являются логическим продолжением задач описания и детектирования аномалий, представленных в предыдущей работы, и суммарно представляют собой задачу адаптации киберфизических систем.

Конечным результатом выполненной работы является полноценная математическая модель, описывающая весь процесс адаптивности киберфизической системы в рамках ее функционирования в условиях возможного возникновения аномалий.

Литература

1. Фатин А.Д. Построение модели адаптивности киберфизических систем: функционирование и детектирование / А.Д. Фатин // Вопросы кибербезопасности. – 2024. – № 2 (60). – С. 36-43. DOI: 10.21681/2311-3456-2024-2-36-43

2. Zegzhda D., Lavrova D., Poltavtseva M. Multifractal security analysis of cyberphysical systems // Nonlinear phenomena in complex systems. – 2019. – Vol. 22. – Issue 2. – pp. 196–204.

3. Application model of modern artificial neural network methods for the analysis of information systems security / Demidov R.A., Pechenkin A.I., Zegzhda P.D., Kalinin M.O. // Automatic Control and Computer Sciences. – 2018. – Vol. 52. – № 8. – С. 965–970.

4. Kalinin M.O., Lavrova D.S., Yarmak A.V. Detection of threats in cyberphysical systems based on deep learning methods using multidimensional time series // Automatic Control and Computer Sciences. – 2018. – Vol. 52. – № 8. – pp. 912–917.

5. Zegzhda D.P., Vasil'ev Y.S., Poltavtseva M.A. Approaches to modeling the security of cyberphysical systems // Automatic control and computer sciences. – 2018. – Vol. 52. – Issue 8. – pp. 1000–1009.

6. Tensor Decomposition for Multilayer Networks Clustering / Zitai Chen et al. // Proceedings of the AAAI Conference on Artificial Intelligence. – 2019. – Vol. 33(01). – pp. 3371–3378. DOI: 10.1609/aaai.v33i01.33013371.

7. Variational Coembedding Learning for Attributed Network Clustering / Shuiqiao Yang et al. // arXiv – CS – Machine Learning (IF). – 2021. DOI: ar-xiv 2104.07295.

8. ANCA: Attributed Network Clustering Algorithm / Issam Falih et al. // COMPLEX NETWORKS2017: Complex Networks & Their Applications VI. – 2017. – pp. 241–252.

9. Saoud B. Networks clustering with bee colony. Artif Intell Rev 52. – 2019. – pp. 1297–1309. DOI: 10.1007/s10462-018-9657-8.

10. Ziruo J., Fuqiang Q. Network Clustering Algorithm Based on Fast Detection of Central Node / Jia Ziruo, Qi Fuqiang // Scientific Programming. – 2022. – pp 1–5. DOI: 10.1155/2022/ 4905190.

11. Automatic security management of smart infrastructures using attack graph and risk analysis / Ivanov D., Kalinin M., Krundyshev V., Orel E. // Proceed-ings of the world conference on smart trends in systems, security and sustainability, WS42020. – 2020. – pp. 295–300.

12. Krundyshev V., Kalinin M. Hybrid neural network frame work for detection of cyberattacks at smart infrastructures // ACM International Conference Proceeding Series. Proceedings of the 12th International Conference on Security of Information and Networks, SIN2019. – 2019. – pp. 3357623.

13. Полтавцева М.А. Адаптивный мониторинг информационной безопасности / М. А. Полтавцева. – СПб., 2021. – 166 с.

14. Савин И.В. Технология кластеризации для обеспечения отказоустойчивости / И. В. Савин // Известия Тульского государственного университета. Технические науки. – 2019. – № 3. – С. 191–194.

References

1. Fatin A.D. Postroyeniye modeli adaptivnosti kiberfizicheskikh sistem: funktsionirovaniye i detektirovaniye / A.D. Fatin // Voprosy kiber-bezopasnosti. – 2024. – № 2 (60). – S. 36-43. DOI: 10.21681/2311-3456-2024-2-36-43.

2. Zegzhda D., Lavrova D., Poltavtseva M. Multifractal security analysis of cy-berphysical systems // Nonlinear phenomena in complex systems. – 2019. – Vol. 22. – Issue 2. – pp. 196–204.

3. Application model of modern artificial neural network methods for the analysis of information systems security / Demidov R.A., Pechenkin A.I., Zegzhda P.D., Kalinin M.O. // Automatic Control and Computer Sciences. – 2018. – Vol. 52. – № 8. – С. 965–970.

4. Kalinin M.O., Lavrova D.S., Yarmak A.V. Detection of threats in cyberphysi-cal systems based on deep learning methods using multidimensional time series // Automatic Control and Computer Sciences. – 2018. – Vol. 52. – № 8. – pp. 912–917.

5. Zegzhda D.P., Vasil'ev Y.S., Poltavtseva M.A. Approaches to modeling the security of cyberphysical systems // Automatic control and computer sciences. – 2018. – Vol. 52. – Issue 8. – pp. 1000–1009.

6. Tensor Decomposition for Multilayer Networks Clustering / Zitai Chen et al. // Proceedings of the AAAI Conference on Artificial Intelligence. – 2019. – Vol. 33(01). – pp. 3371–3378. DOI: 10.1609/aaai.v33i01.33013371.

7. Variational Co-embedding Learning for Attributed Network Clustering / Shuiqiao Yang et al. // arXiv – CS – Machine Learning (IF). – 2021. DOI: ar-xiv 2104.07295.

8. ANCA: Attributed Network Clustering Algorithm / Issam Falih et al. // COM-PLEX NETWORKS2017: Complex Networks & Their Applications VI. – 2017. – pp. 241–252.

9. Saoud B. Networks clustering with bee colony. Artif Intell Rev 52. – 2019. – pp. 1297–1309. DOI: 10.1007/s10462-018-9657-8.

10. Ziruo J., Fuqiang Q. Network Clustering Algorithm Based on Fast Detection of Central Node / Jia Ziruo, Qi Fuqiang // Scientific Programming. – 2022. – pp 1–5. DOI: 10.1155/2022/ 4905190.

11. Automatic security management of smart infrastructures using attack graph and risk analysis / Ivanov D., Kalinin M., Krundyshev V., Orel E. // Proceedings of the world conference on smart trends in systems, security and sustainability, WS42020. – 2020. – pp. 295–300.

12. Krundyshev V., Kalinin M. Hybrid neural network frame work for detection of cyber-attacks at smart infrastructures // ACM International Conference Proceeding Series. Proceedings of the 12th International Conference on Security of Information and Networks, SIN2019. – 2019. – pp. 3357623.

13. Poltavtseva M.A. Adaptivnyy monitoring informatsionnoy bezopasnosti / M. A. Poltavtseva. – SPb., 2021. – 166 s.

14. Savin I.V. Tekhnologiya klasterizatsii dlya obespecheniya otkazoustoychivosti / I. V. Savin // Izvestiya Tul'skogo gosudarstvennogo univer-siteta. Tekhnicheskiye nauki. – 2019. – № 3. – S. 191–194.

Фатин Александр Денисович, аспирант, Санкт-Петербургский политехнический университет Петра Великого (СПбПУ). 195251, г. Санкт-Петербург, ул. Политехническая, д.29-Б. E-mail: sasha-fatin@mail.ru

Fatin Aleksandr Denisovich, postgraduate student, Peter the Great St. Petersburg Polytechnic University (SPbPU). 195251, St. Petersburg, Politekhnicheskaya St., 29-B. E-mail: sasha-fatin@mail.ru