УДК 004.89 DOI 10.23683/2311-3103-2019-2-57-68

Д.В. Заруба, Д.Ю. Запорожец, Ю.Ю. Запорожец

ПОДСИСТЕМА РАСПРЕДЕЛЕННОГО РЕШЕНИЯ ОПТИМИЗАЦИОННЫХ ЗАДАЧ*

Рассматривается распределенный подход для решения оптимизационных задач, основанный на использовании биоинспирированных алгоритмов. Использование данного класса алгоритмов обусловлено возможность декомпозиции кодированных решений на независимые подмножества и их обработки в отдельных потоках. Одной из основных проблем при таком подходе является выполнение декомпозиции и последующей свертки решения. Проблема становится более сложной, в тех случаях, когда требуется несколько уровней декомпозиции сложной задачи. С точки зрения вычислительной мощности одновременная обработка параллельных потоков требует значительных ресурсов процессорного времени и оперативной памяти. В связи с этим использование нескольких вычислительных узлов, взаимодействующих через сетевые интерфейсы, способствует увеличению вычислительных ресурсов и повышению отказоустойчивости подсистемы в целом. Предложена реализация распределенной подсистемы решения NP-полных оптимизационных задач, которая позволяет в автоматизированном режиме разбивать множество входных данных на подмножества, распределять подзадачи между вычислительными узлами, а также собирать полученные решения в решение исходной задачи. В работе представлена обобщенная архитектура будущей подсистемы, компонентные модели обработки входных данных и взаимодействия агентов между собой. Для подтверждения работоспособности подсистемы была разработана программная реализация на языке программирования Java и брокер сообщений RabbitMQ, для обеспечения взаимодействия программных агентов между собой. Проведены серии экспериментов, в рамках которых были проведены исследования с несколькими одновременно запущенными задачами, а также исследования с заданным количеством одновременно запущенных программных агентов.

Оптимизация; декомпозиция; брокер очередей сообщений; генетический алгоритм.

D.V. Zaruba, D.Yu. Zaporozhets, Yu.Yu. Zaporozhets

DISTRIBUTED SUBSYSTEM OF OPTIMIZATION PROBLEMS SOLVING

In this paper, distributed approaches are used to solve optimization problems based on the use of bioinspired algorithms. The use of this class of algorithms is due to the possibility of decomposition of coded solutions for independent subsets and their processing in separate streams. One of the main problems with this approach is the implementation of decomposition and the adoption of final decisions. When multiple levels of decomposition of existing tasks are required the problem becomes more complicated. From the point of view of computational power, simultaneous processing requires considerable resources of processor time and RAM. All these compute nodes interact through network interfaces. This article proposes the implementation of a distributed subsystem of solutions of NP-complete optimization problems, which allows you to automatically break a lot of input data into a subset, distribute subtasks between computational nodes, and collect the results of solutions in solving the original problem. The paper presents the overall architecture of the future subsystem, component models of input data processing and interaction with each other. To confirm the performance of the subsystem, software implementations in the Java programming language and the message broker RabbitMQ were developed to ensure the interaction of software agents with each other. A series of experiments were conducted, in the course of which simultaneous tasks were performed.

Optimization; decomposition; message broker; genetic algorithm.

* Исследование выполнено при финансовой поддержке гранта РФФИ (проект №19-01-00059).

Введение. Для повышения скорости решения, а также для частичного решения проблемы предварительной сходимости NP-трудных задач оптимизации в мировом сообществе обоснованно используются биоинспирированные поисковые методы оптимизации, основанные на моделировании природных явлений и биологических систем [1–3]. Одной из основных особенностей данного класс алгоритмов предполагает обработку множества альтернативных решений задачи оптимизации и представляют собой альтернативу классическим поисковым алгоритмам, которые работают только с одним решением поставленной оптимизационной задачи [3]. Это позволяет находить оптимальные и квазиоптимальные решения за полиномиальное время. Тем не менее с ростом объема входных данных количество возможных альтернативных решений растет факториально, поэтому использования различных подходов, связанных с декомпозицией задачи на части и параллельной обработкой этих частей является актуальной задачей [4–6].

В рамках данной работы предлагается новый, распределенный подход к реализации биоинспирированных алгоритмов для решения оптимизационных задач. Основной идеей предлагаемого подхода является автоматическое разбиение задачи на части по заданному критерию и по определенному правилу с последующей сверткой. В качестве тестовой задачи была выбрана классическая задача коммивояжера, известные бенчмарки (Eilon 50, Eilon 75, Oliver 30) [1, 7] и графы, сгенерированные случайным образом с заданными параметрами.

Все биоинспирированные алгоритмы относятся к классу эвристических алгоритмов, то есть алгоритмов, для которых теоретически не доказана сходимость к глобальному оптимуму, но эмпирически установлено, что вероятность получения оптимального или квазиоптимального решения высока [3, 8, 9].

Общая схема биоинспирированных алгоритмов. В общем случае все биоинспирированные подходы включают в себя следующие этапы [3, 8, 9].

Инициализация начального множества альтернативных решений (популяции). В заданной области поиска создается заданное количество альтернативных решений, которые в разной степени приближены к искомому решению. Для получения начальной популяции могут быть применены детерминированные или случайные алгоритмы [10, 11]. При формировании начальной популяции решениями покрывается по возможности вся область поиска (поиск в ширину). В случае если окрестность глобального экстремума известна, агентами заполняется окрестность и выполняется поиск вблизи глобального экстремума оптимизируемой функции (поиск в глубину). Такой подход может существенно сократить время решения задачи. При решении реальных задач информация о глобальном экстремуме отсутствует, поэтому агенты равномерно распределяются по всей области поиска.

Генерация новых альтернативных решений. С помощью набора операторов генерируются новые решения таким образом, что каждое новое решение итерационно приближается к оптимуму. Данный набор операторов - математических преобразований, основанных на различных вероятностых подходах, является особенностью каждого биоинспирированного алгоритма. Например, для бактериального алгоритма – это моделирование движения бактерии eColi, для генетических алгоритмов (ГА) – скрещивание пар альтернативных решений, названное в парадигме ГА кроссинговером, и мутация (резкое изменений одной или нескольких частей альтернативного решения). Для муравьиных алгоритмов – моделирование построения муравьев (агентов).

Отбор полученных решений. Этап генерации новых альтернативных решений увеличивает размер популяции в несколько раз (в зависимости от настроечных параметров). При этом в популяции в общем случае могут оказаться решения

с низкой оценкой целевой функции или даже нелегитимные решения. Так же быстрый рост популяции приводит к резкому ухудшению сходимости алгоритма. Для решения данных проблем вводится этап отбора. На данном шаге используются различные эвристики, с помощью которых осуществляется выборка альтернативных решений. Наглядным примером, является «жадная» эвристика, которая выбирает только лучшие решения, но доказано, что это приводит к вырождению популяции альтернативных решений в целом. На сегодняшний день известно большое множество операторов отбора [2], как универсальных, так и специализированных, применяемых при решении конкретных задач.

Завершение поиска. Проверка выполнения критерия останова. При его выполнении лучшее из найденных решений популяции принимается за приближенное решение задачи. Известны следующие широко применяемые критерии останова: количество итерации, время работы. Также достаточно часто используют в качестве критерия останова достижение популяцией состояния стагнации алгоритма – такое состояние, при котором лучшее достигнутое квазиоптимальное решение неизменно на протяжении заданного числа итераций или времени.

На рис. 1 представлена обобщенная структура биоинспирированного алгоритма.

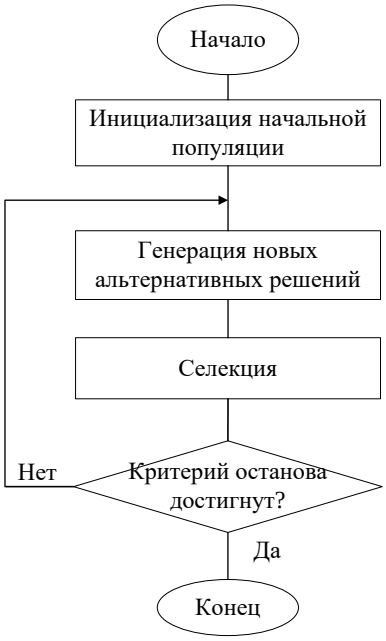

Рис. 1. Обобщенная структура биоинспирированного алгоритма

Одним из основных преимуществ биоинспирированных алгоритмов является их модульная структура. Модульный подход позволяет быстро получить множество новых альтернативных решений за счет разработки новых и модификации имеющихся правил инициализации и генерации новых агентов на основании скрещивания имеющихся в популяции решений.

Для всех биоинспирированных алгоритмов характерны следующие свойства агентов [3, 11]:

- автономность – агенты в пространстве перемещаются независимо друг от друга;

♦ стохастичность – процесс создания новых агентов в какой-то мере является произвольным;

♦ коммуникабельность – агенты способны обмениваться имеющейся информацией, выявленной в ходе поискового процесса;

♦ ограниченность представления – каждый из агентов популяции несет информацию об исследуемой части поискового пространства;

♦ децентрализация – отсутствие агентов высшего уровня, управляющих процессом поиска в целом.

Биоинспирированные оптимизационные алгоритмы по сравнению с классическими методами оптимизации имеют ряд преимуществ особенно при решении неформализованных задач и задач высокой размерности. При таких условиях биоинспирированные алгоритмы обеспечивают высокую вероятность нахождения оптимального или квазиоптимального (приближенного) решения за полиномиальное время. Именно приближенное решение часто является достаточным.

Тем не менее при больших объемах входных данных наблюдается значительный рост времени работы биоэвристических решений. Это связано с тем, что временная сложность данных алгоритмов может варьироваться от $O(n^2)$ до $O(n^3)$. Для частичного решения проблемы большой размерности в данной работе предлагается использовать распределенный подход для решения оптимизации. Основной идеей является использование брокера сообщений и распределении «заданий» между подключенными программными агентами.

Архитектура распределенной подсистемы решения оптимизационных задач. Для повышения скорости работы алгоритмов решения задач оптимизации в данной статье предлагается подсистема позволяющая автоматически разбивать задачи на подзадачи, а также производиться свертку полученных результатов [12–15]. В частности, в качестве примера, предлагается решение задачи коммивояжера на основе предварительной декомпозиции множества входных данных. Обобщенная схема работы алгоритма предложена на рис. 2.

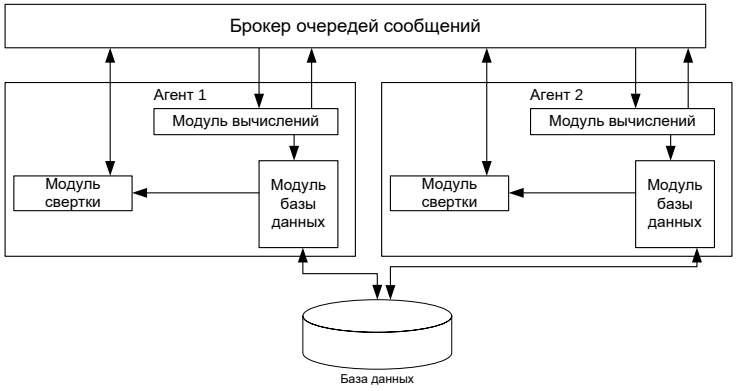

Рис. 2. Обобщенная схема работы распределенной системы

Основой подсистемы является брокер очередей сообщений. В работе предлагается использовать зарекомендованную реализацию RabbitMQ [14, 15].

В работе предлагается использовать термин задача для определения набора данных, состоящих из условия декомпозиции, правила декомпозиции, правила свертки, правила вычисления решений (алгоритм решения), правила вычисления целевой функции и входных данных, для которых необходимо найти решение или набор решений.

Математически задача t_i представляется в виде кортежа

$$t_i = \langle uid, R, C, U, F, E, d_i, p_i, o_i \rangle,$$

где uid – уникальный идентификатор задачи в системе;

R – правило декомпозиции, по которому разбивается набор данных d_i;.

C – условие разбиения набора входных данных d_i.

Если условие C выполняется, значит будет применено правило R для декомпозиции набора входных данных d_i на множество D_i' и создания на их основе множества дочерних подзадач $T_i' = \{t_i'\} = R(d_i)$, где d_i – входные данные для задачи t_i, U – правило свертки набора множества выходных данных дочерних подзадач

Результатом работы правила свертки является множество выходных данных (решений). Данные решения могут быть результатом работы подзадачи так и корневой задачи

$$o_i = \begin{cases} U(T_i'), \text{ если } C = true \\ E(d_i), \text{ если } C = false \end{cases},$$

где E – алгоритм решения задачи (набор требуемых вычислений);

F – функция приспособленности полученных решений (целевая функция);

p_i– ссылка на родительскую задачу. В данной работе предлагается использовать uid родительской задачи. Если для текущей задачи t_i– p_i является пустым множество, то данная задача является корневой. Её решение является решением исходной прикладной задачи в целом.

В общем случае схема жизненного цикла задачи t_i представлена на рис. 3 и 4.

Исходная задача поступает в брокер очередей сообщений в очередь вычислений. Модуль вычислений одного из агентов выбирает очередную задачу из очереди, регистрирует задачу в хранилище данных с уникальным идентификатором uid и статусом IN_PROGRESS (в работе) и проверяет правило декомпозиции C. Если правило выполняется, то исходная задача разделяется на подзадачи t_i с $p_i = uid$ и отправляется в очередь сообщений. В противном случает выполняются вычисления E. Результаты вычислений сохраняются для текущей задачи в хранилище результатов со статусом DONE (выполнено).

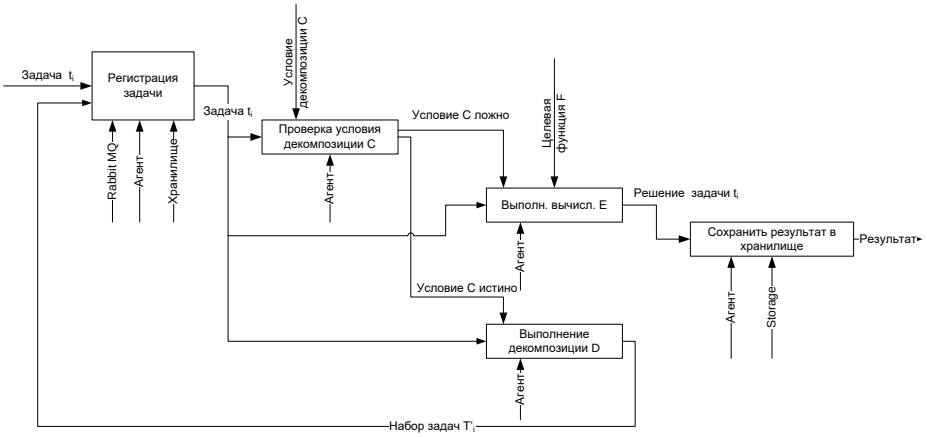

Рис. 3. Схема работы механизма декомпозиции задачи

Для свертки результатов работы подзадач каждым агентом каждый промежуток времени $\Delta\tau$ опрашивается хранилище данных и выбирает данные по правилу S. Правило выборки S можно описать следующим образом:

Выполняется запрос к хранилищу для подсчета количества задач с одинаковым p_i. Данный запрос считает количество всех зарегистрированных задач, которые являются дочерними одной задачи.

Выполняется запрос к хранилищу для подсчета количества задач со статусом (DONE) и с одинаковым p_i. Данный запрос считает количество всех выполненных зарегистрированных задач, которые являются дочерними одной задачи.

Если для каждого p_i общее количество задач совпадает с количество выполненных задач, то для всех задач с одинаковым pi можно выполнить правило свертки. Если таких групп несколько, то текущий агент выбирает первую из списка.

Для того, чтобы правило свертки не было применено одновременно для одних и тех же подзадач разными агентами, на время выполнения правила выборки S хранилище блокирует доступ других агентов.

Рассмотрим пример правила выборки S для реляционного хранилища данных на примере MySQL.

Имеется таблица TASK с полями, среди которых ID – уникальный идентификатор задачи, PARENT_ID – уникальный идентификатор родительской задачи, STATUS – текущий статус задачи. Соответственно будут применены следующие запросы:

1. SELECT PARENT_ID, COUNT(*) FROM TASK GROUP BY PARENT_ID;
2. SELECT PARENT_ID, COUNT(*) FROM TASK WHERE STATUS='DONE' GROUP BY PARENT_ID;
3. Поиск среди результатов полученных на шагах 1 и 2 полностью идентичных записей, для которых PARENT_ID1 = PARENT_ID2 & COUNT(*)1 = COUNT(*)2.

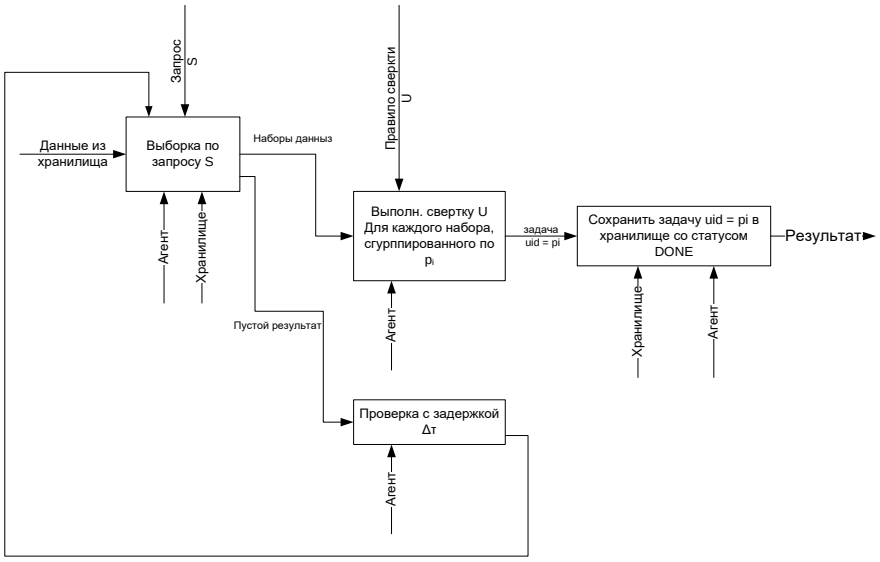

Рис. 4. Схема работы механизма свертки задачи

Экспериментальные исследования. Для подтверждения предложенного в данной работе механизма оптимизации была реализована распределенная подсистема решения задач оптимизации. В качестве брокера сообщений была выбрана реализация RabbitMQ [14, 15]. В качестве хранилища данных выбрана СУБД MySQL [16–20]. Программные агенты реализованы на языке Java. Для получения оценок времени работы системы и качества получаемых решений была выбрана классическая задача оптимизации – поиск гамильтонова цикла в графе. В качестве

тестовых графов рассматриваются бенчмарки *Eilon_50, Oliver_30*, а также сгенерированные по заданным параметрам графы. В качестве алгоритма решения задачи был выбран простой генетический алгоритм, в качестве условия выполнения декомпозиции $Nt > 0.2 * N0$, где Nt количество вершин графа в текущей задачи t, $N0$ – начальное количество вершин в графе. Правило декомпозиции заключается в разделении множества входных данных пополам, а именно произвольное разбиение графа из задачи t на два подграфа и инициализация задач t' и t'', в которых входными данными d являются полученные при разбиении подграфы.

Управляющие параметры для генетического алгоритма имеют следующие значения:

- размер популяции – N0 * N0;
- количество итерации – 1000;
- вероятность выполнения оператора кроссинговера – 85 %;
- вероятность выполнения оператора мутации – 15 %
- количество одновременно запущенных агентов – 3.

В первой серии экспериментов были выявлены зависимости скорости работы системы от количества одновременно выполняемых задач. В систему одновременно добавлялись NT количество одинаковых задач. Следующий запуск выполняется для $NT + delta$, где $delta$ – приращение количества одновременно запущенных задач.

Таблица 1

Время решения задачи от количества решаемых задач

	Количество одновременно запущенных задач, шт.									
	1	2	3	4	5	10	15	20	25	30
Мин. время, сек	3	5	4	5	5	9	8	6	7	7
Средн. время, сек	3	5	6	9	9	16	20	25	29	36
Макс. время, сек	3	5	7	11	11	21	33	36	43	63

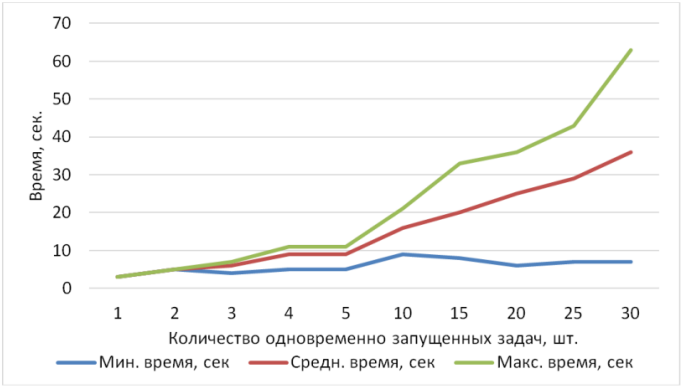

Рис. 5. Зависимость времени выполнения задачи от количества одновременно решаемых задач

Во второй серии экспериментов были выявлены зависимости скорости работы системы от количества одновременно работающих агентов. В систему одновременно добавлялись 5 одинаковых задач. График зависимости времени решения задач от количества агентов в системе приведены на рис. 6, 7 и табл. 2, 3.

Таблица 2

Время решения задачи от количества агентов. Количество задач – 5

	Количество одновременно работающих агентов, шт.																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Мин. время, сек	12	7	5	5	8	4	5	5	6	8	11	11	9	8	6	10	11	11	10	9
Средн. время, сек	39	14	10	8	9	8	9	8	7	11	11	11	10	10	10	10	12	11	11	11
Макс. время, сек	46	20	14	12	12	12	11	13	12	13	12	12	11	12	11	11	12	12	13	14

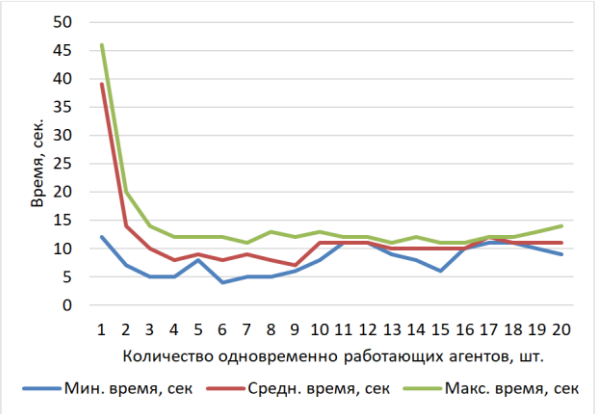

Рис. 6. Зависимость времени выполнения задачи от количества одновременно запущенных агентов

Таблица 3

Время решения задачи от количества агентов. Количество задач – 10

	Количество одновременно работающих агентов, шт.					
	1	5	10	15	20	25
Мин. время, сек	9	8	9	13	15	18
Средн. время, сек	148	17	18	17	19	20
Макс. время, сек	196	23	22	21	21	22

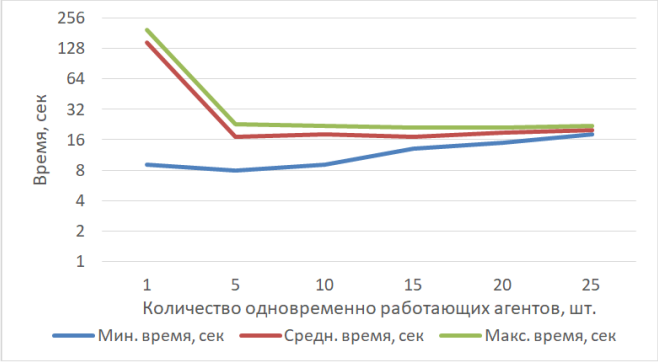

Рис. 7. Зависимость времени выполнения задачи от количества одновременно запущенных агентов

Заключение. В данной работе была предложена биоинспирированная распределенная подсистема решения оптимизационных задач, основанная на автоматической декомпозиции набора входных данных в соответствии с заданными правилами, решения подзадач и последующим автоматическим слиянием полученных решений в решение исходной задачи. Основная особенность полученной подсистемы является в том, что исследователю, который решает конкретную прикладную задачу не требуется реализовывать программные модули поддержки алгоритма решения задачи, а сконцентрироваться непосредственно на применяющихся прикладных методах и алгоритмах. В статье были представлены основные этапы работы подсистемы, схемы работы механизмов декомпозиции и свертки. Подробно описан формат входных данных. В рамках работы, на основе описанных механизмов, был реализован прототип подсистемы распределенного решения оптимизационных задач. В рамках экспериментальных исследований были проведены исследования зависимости скорости работы подсистемы от количества одновременно запущенных агентов и одновременно запущенных задач. Результаты экспериментов показали, что для наиболее эффективной работы подсистемы количество одновременно работающих агентов должно быть такое, как и ожидаемое количество одновременно решаемых задач. Дальнейшее развитие распределенной подсистемы решения оптимизационных задач заключается в интеграции нереляционных высокопроизводительных СУБД в качестве хранилища данных для повышения скорости работы подсистемы в целом.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Запорожец Д.Ю., Запорожец Д.Ю*. Генерация биоинспирированных поисковых процедур для решения оптимизационных задач // Известия ЮФУ. Технические науки. – 2016. – № 6 (179). – С. 13-24.
2. *Кравченко Ю.А., Кравченко Ю.А., Логинов О.А., Запорожец Д.Ю.* Метод интеллектуального принятия эффективных решений на основе биоинспирированного подхода // Известия Кабардино-Балкарского научного центра РАН. – 2017. – № 6-2 (80). – С. 162-169.
3. *Родзин С.И., Курейчик В.В.* Состояние, проблемы и перспективы развития биоэвристик // Программные системы и вычислительные методы. – 2016. – № 2. – С. 158-172.
4. *Кравченко, Ю.А.* Разработка вычислительно-независимых моделей (cim) на основе многоагентных систем // Известия Кабардино-Балкарского научного центра РАН. – 2017. – № 6-2 (80). – С. 149-155.
5. *Пантелюк Е.А., Запорожец Д.Ю., Курейчик В.В., Заруба Д.В.* Модифицированный гибридный алгоритм на основе алгоритмов волчьей стаи и дифференциальной эволюции // Информатизация и связь. – 2018. – № 4. – С. 61-66.
6. *Zaporozhets D., Zaruba D., Kulieva N.* Parallel approach for bioinspired algorithms // Journal of Physics: Conference Series "International Conference Information Technologies in Business and Industry 2018 - Enterprise Information Systems". – 2018. – P. 042065.
7. *Kureichik V.M., and Kureichik V.V.* Genetic algorithm for the graph placement // Journal of Computer and Systems Sciences International. – 2000. – No. 39.5. – P. 733-740.
8. *Курейчик В.М., Логунова Ю.А., Игнатьева А.С*. Обзор состояния проблемной области по теме решение задачи коммивояжёра // Информатика, вычислительная техника и инженерное образование. – 2017. – № 3 (31). – С. 39-59.
9. *Курейчик В.М., Синютин Е.С., Каплунов Т.Г.* Прогнозирование состояния технических систем при помощи генетических алгоритмов // Вестник Рязанского государственного радиотехнического университета. – 2018. – № 65. – С. 107-112.
10. *Kureichik V., Zaruba D., Kureichik V.V.* Hybrid approach for graph partitioning // Advances in Intelligent Systems and Computing. – 2017. – Т. 573. – P. 64-73.
11. *Курейчик Л.В., Курейчик В.В., Курейчик В.В.* Интегрированная инструментальная подсистема генетического поиска // Программная инженерия: методы и технологии разработки информационно-вычислительных систем (ПИИВС-2016): Сб. научных трудов I научно-практической конференции (студенческая секция). ГОУ ВПО "Донецкий национальный технический университет", 2016. – С. 93-96.

12. *Хованских С.А., Литвиненко В.А., Хованскова В.С.* Организация и защита распределенных вычислений на базе многоагентной системы в компьютерной сети с целью сокращения времени решения масштабных задач // Известия ЮФУ. Технические науки. – 2018. – № 4 (198). – С. 198-210.
13. *Полковникова Н.А., Курейчик В.М.* Многокритериальная оптимизация на основе эволюционных алгоритмов // Известия ЮФУ. Технические науки. – 2015. – № 2 (163). – С. 149-162.
14. *Кравченко Ю.А., Курситыс И.О.* Комбинированный подход к решению задачи распределения ресурсов // Известия ЮФУ. Технические науки. – 2017. – № 7 (192). – С. 111-122.
15. *Курситыс И.О., Кравченко Ю.А.* Применение имитационного моделирования и временных сетей Петри для задачи распределения вычислительных ресурсов // Фундаментальные и прикладные аспекты компьютерных технологий и информационной безопасности: Сб. статей II Всероссийской научно-технической конференции молодых ученых, аспирантов и студентов. – 2016. – С. 254-258.
16. *Roy G.M.* RabbitMQ in Depth. – Manning Publications, 2018.
17. *Batyuk A. et al.* Software architecture design of the real-time processes monitoring platform // 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP). – IEEE, 2018. – P. 98-101.
18. *Das N. et al.* NoSQL Overview and Performance Testing of HBase Over Multiple Nodes with MySQL // Emerging Technologies in Data Mining and Information Security. – Springer, Singapore, 2019. – P. 269-279.
19. *Bell C.* Introducing InnoDB Cluster: Learning the MySQL High Availability Stack. – Apress, 2018.
20. *Kacprzyk J., Kureichik V.M., Malioukov S.P., Kureichik V.V., Malioukov A.S.* General questions of automated design and engineering // Studies in Computational Intelligence. – 2009. – Vol. 212. – P. 1-22.

REFERENCES

1. *Zaporozhets D.Yu., Zaporozhets D.Yu.* Generatsiya bioinspirirovannykh poiskovykh protsedur dlya resheniya optimizatsionnykh zadach [Generation of bio-inspired search procedures for solving optimization problems], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2016, No. 6 (179), pp. 13-24.
2. *Kravchenko Yu.A., Kravchenko Yu.A., Loginov O.A., Zaporozhets D.Yu.* Metod intellektual'nogo prinyatiya effektivnykh resheniy na osnove bioinspirirovannogo podkhoda [Method of intellectual decision-making on the basis of bioinspired approach], *Izvestiya Kabardino-Balkarskogo nauchnogo tsentra RAN* [Izvestiya of Kabardino-Balkar scientific center of RAS], 2017, No. 6-2 (80), pp. 162-169.
3. *Rodzin S.I., Kureychik V.V.* Sostoyanie, problemy i perspektivy razvitiya bioevristik [Status, problems and prospects of bio-heuristics], *Programmnye sistemy i vychislitel'nye metody* [Software systems and computational methods], 2016, No. 2, pp. 158-172.
4. *Kravchenko, Yu.A.* Razrabotka vychislitel'no-nezavisimykh modeley (cim) na osnove mnogoagentnykh sistem [The development of a computational-independent models (cim) based on multi-agent systems], *Izvestiya Kabardino-Balkarskogo nauchnogo tsentra RAN* [Izvestiya of Kabardino-Balkar scientific center of RAS], 2017, No. 6-2 (80), pp. 149-155.
5. *Pantelyuk E.A., Zaporozhets D.Yu., Kureychik V.V., Zaruba D.V.* Modifitsirovannyy gibridnyy algoritm na osnove algoritmov volch'ey stai i differentsial'noy evolyutsii [Modified hybrid algorithm based on wolf pack algorithms and differential evolution], *Informatizatsiya i svyaz'* [Informatization and communication], 2018, No. 4, pp. 61-66.
6. *Zaporozhets D., Zaruba D., Kulieva N.* Parallel approach for bioinspired algorithms, *Journal of Physics: Conference Series "International Conference Information Technologies in Business and Industry 2018 - Enterprise Information Systems"*, 2018, pp. 042065.
7. *Kureichik V.M., and Kureichik V.V.* Genetic algorithm for the graph placement, *Journal of Computer and Systems Sciences International*, 2000, No. 39.5, pp. 733-740.
8. *Kureychik V.M., Logunova Yu.A., Ignat'eva A.S.* Obzor sostoyaniya problemnoy oblasti po teme reshenie zadachi kommivoyazhera [Overview of the health problem areas on the subject the solution to the traveling salesman problem], *Informatika, vychislitel'naya tekhnika i inzhenernoe obrazovanie* [Computer science, computer engineering and engineering education], 2017, No. 3 (31), pp. 39-59.

9. *Kureychik V.M., Sinyutin E.S., Kaplunov T.G.* Prognozirovanie sostoyaniya tekhnicheskikh sistem pri pomoshchi geneticheskikh algoritmov [Predicting the state of technical systems using genetic algorithms], *Vestnik Ryazanskogo gosudarstvennogo radiotekhnicheskogo universiteta* [Bulletin of Ryazan state radio engineering University], 2018, No. 65, pp. 107-112.
10. *Kureichik V., Zaruba D., Kureichik V.V.* Hybrid approach for graph partitioning, *Advances in Intelligent Systems and Computing*, 2017, Vol. 573, pp. 64-73.
11. *Kureychik L.V., Kureychik V.V., Kureychik V.V.* Integrirovannaya instrumental'naya podsistema geneticheskogo poiska [Integrated instrumental subsystem of genetic search], *Programmnaya inzheneriya: metody i tekhnologii razrabotki informatsionno-vychislitel'nykh sistem (PIIVS-2016): Sb. nauchnykh trudov I nauchno-prakticheskoy konferentsii (studencheskaya sektsiya)* [Software engineering: methods and technology development of informational-computational systems (PIIS-2016): Collection of scientific papers I scientific-practical conference (student section),. GOU VPO "Donetskiy natsional'nyy tekhnicheskiy universitet", 2016, pp. 93-96.
12. *Khovanskov S.A., Litvinenko V.A., Khovanskova V.S.* Organizatsiya i zashchita raspredelennykh vychisleniy na baze mnogoagentnoy sistemy v komp'yuternoy seti s tsel'yu sokrashcheniya vremeni resheniya masshtabnykh zadach [Organization and protection in distributed computing based on multi-agent systems in computer networks to reduce the solution time of large-scale problems], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2018, No. 4 (198), pp. 198-210.
13. *Polkovnikova N.A., Kureychik V.M.* Mnogokriterial'naya optimizatsiya na osnove evolyutsionnykh algoritmov [Multicriteria optimization based on evolutionary algorithms], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2015, No. 2 (163), pp. 149-162.
14. *Kravchenko Yu.A., Kursitys I.O.* Kombinirovannyy podkhod k resheniyu zadachi raspredeleniya resursov [Combined approach to solving the problem of resource allocation], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2017, No. 7 (192), pp. 111-122.
15. *Kursitys I.O., Kravchenko Yu.A.* Primenenie imitatsionnogo modelirovaniya i vremennykh setey Petri dlya zadachi raspredeleniya vychislitel'nykh resursov [Application of simulation modeling and time Petri nets for the problem of distribution of computing resources], *Fundamental'nye i prikladnye aspekty komp'yuternykh tekhnologiy i informatsionnoy bezopasnosti: Sb. statey II Vserossiyskoy nauchno-tekhnicheskoy konferentsii molodykh uchenykh, aspirantov i studentov* [Fundamental and applied aspects of computer technologies and information security: Collection of articles of the II all-Russian scientific and technical conference of young scientists, postgraduates and students], 2016, pp. 254-258.
16. *Roy G.M.* RabbitMQ in Depth. Manning Publications, 2018.
17. *Batyuk A. et al.* Software architecture design of the real-time processes monitoring platform, *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*. IEEE, 2018, pp. 98-101.
18. *Das N. et al.* NoSQL Overview and Performance Testing of HBase Over Multiple Nodes with MySQL, *Emerging Technologies in Data Mining and Information Security*. Springer, Singapore, 2019, pp. 269-279.
19. *Bell C.* Introducing InnoDB Cluster: Learning the MySQL High Availability Stack. Apress, 2018.
20. *Kacprzyk J., Kureichik V.M., Malioukov S.P., Kureichik V.V., Malioukov A.S.* General questions of automated design and engineering, *Studies in Computational Intelligence*, 2009, Vol. 212, pp. 1-22.

Статью рекомендовала к опубликованию д.т.н., профессор Л.С. Лисицына.

Заруба Дарья Викторовна – Южный федеральный университет; e-mail: dvzaruba@sfedu.ru; 347928, г. Таганрог, пер. Некрасовский, 44; тел.: 88634371651; кафедра систем автоматизированного проектирования; ассистент.

Запорожец Дмитрий Юрьевич – e-mail: duzaporozhets@sfedu.ru; кафедра систем автоматизированного проектирования; к.т.н.; доцент.

Запорожец Юлия Юрьевна – e-mail: z.ulia19941109@mail.ru; кафедра систем автоматизированного проектирования; аспирант.

Zaruba Daria Victorovna – Southern Federal University; e-mail: dvzaruba@sfedu.ru; 44, Nekrasovskiy, Taganrog, 347928, Russia; phone: +78634371651; the department of computer aided design; assistant.

Zaporozhets Dmitriy Yuryevich – e-mail: duzaporozhets@sfedu.ru; the department of computer aided design; cand. of eng. sc.; associate professor.

Zaporozhets Yuliya Yuryevna – e-mail: duzaporozhets@sfedu.ru; the department of computer aided design; postgraduate.