УДК 004.023

РЕАЛИЗАЦИЯ И ИССЛЕДОВАНИЕ ЭФФЕКТИВНОСТИ АЛГОРИТМА «ЛЕТУЧИХ МЫШЕЙ»

Р. Ш. Гамзалиев, А. А. Дроздова
Донской государственный технический университет, Ростов-на-Дону, Российская Федерация
twiposter@gmail.com,
anna.93.08@mail.ru

Произведено исследование «алгоритма летучих мышей», который относится к классу метаэвристических и позволяет решать оптимизационные задачи различного рода. В ходе работы было реализовано программное средство под платформу .NET, позволяющее проводить вычислительные эксперименты, на основе которых был проведен анализ влияния основных параметров алгоритма на эффективность его работы.

Ключевые слова: алгоритм, метаэвристика, алгоритм летучих мышей, оптимизация, роевой интеллект.

UDC 004.023

IMPLEMENTATION AND RESEARCH OF THE BAT ALGORITHM EFFECTIVENESS

R. S. Gamzaliev, A. A. Drozdova
Don State Technical University, Rostov-on-Don, Russian Federation
twiposter@gmail.com,
anna.93.08@mail.ru

This article provides the research results on the "bat algorithm", which belongs to metaheuristics class and allows solving different optimization problems. The authors implemented the software on the .NET platform that allows running computational experiments, on the basis of which they conducted the analysis of the influence of the main algorithm parameters on its performance.

Keywords: algorithm, metaheuristic, bat algorithm, optimization, swarm intelligence.

Введение. Живая природа является одним из источников идей для создания алгоритмов. На основе поведения биологических систем было создано множество эвристических алгоритмов, в том числе и алгоритмы роевого интеллекта.

Роевым интеллектом называют децентрализованную самоорганизующуюся систему, состоящую из множества агентов, локально взаимодействующих между собой и с окружающей средой. При этом каждый агент этой системы следует простым правилам, которые приводят к возникновению интеллектуального глобального поведения, не контролируемого отдельными агентами [1].

В данной работе рассматривается один из представителей алгоритмов роевого интеллекта — «алгоритм летучих мышей». Данный алгоритм впервые был опубликован *Xin-She Yang* в 2010 году [2].

Описание алгоритма. Основой алгоритма является популяция, состоящая из множества агентов. С течением времени популяция эволюционирует. Каждый этап эволюции характеризуется определенным состоянием популяции и идентифицируется номером поколения. Эволюция популяции включает эволюцию каждого ее агента. Агентом популяции является летучая мышь, для которой свойственна определенная позиция, скорость перемещения, громкость эхолокационного сигнала, а также частота его пульсации и длина волны. Летучая мышь движется от одной позиции к другой, при этом каждая позиция представляет собой решение задачи и имеет показатель качества, определяемый при помощи целевой функции.

На рис. 1 изображена общая схема алгоритма. На первом этапе происходит инициализация агентов популяции, после чего осуществляется поиск лучшего положения мыши среди особей первого поколения и запуск итерационного цикла, в котором, на каждом ветке, каждый агент из популяции проходит эволюцию. По завершению итерации выбирается лучшая позиция среди текущего поколения. Лучшая позиция последнего поколения является окончательным решением.

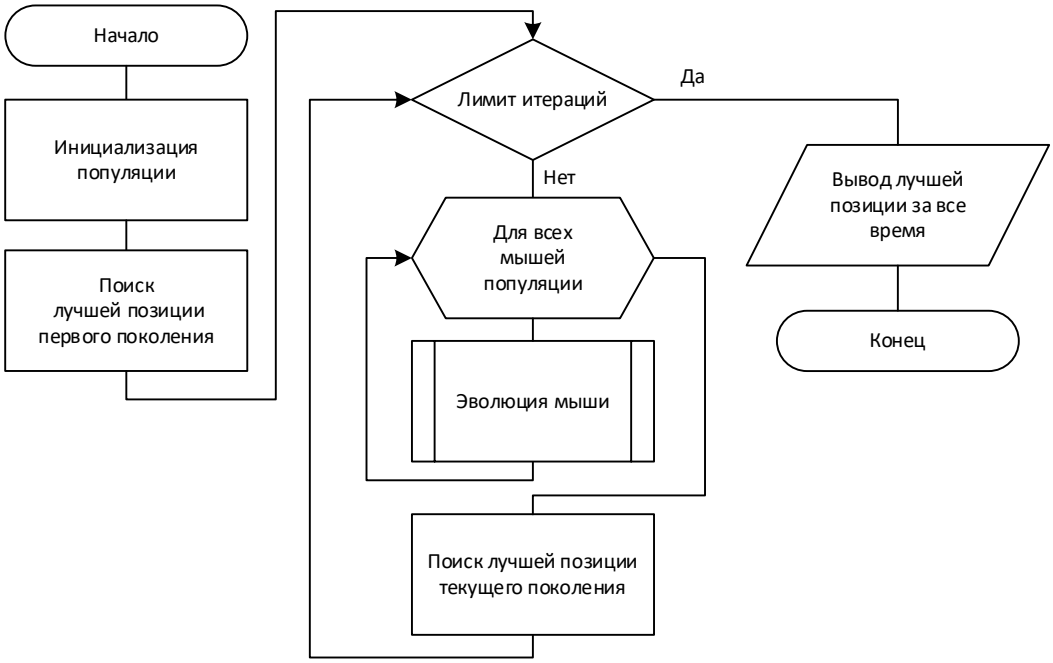

Рис. 1. Блок-схема «алгоритма летучих мышей»

На рис. 2 демонстрируется блок-схема эволюции мыши. Сначала происходит пересчет вектора скорости и обновление положения мыши. Затем, с вероятностью, обратно пропорциональной значению частоты пульсации сигнала, выполняется локальный поиск. После этого происходит принятие новой позиции с вероятностью, прямо пропорциональной громкости сигнала импульса, в случае, если новая позиция не хуже текущей. Завершает процедуру пересчет громкости и частоты пульсации сигнала.

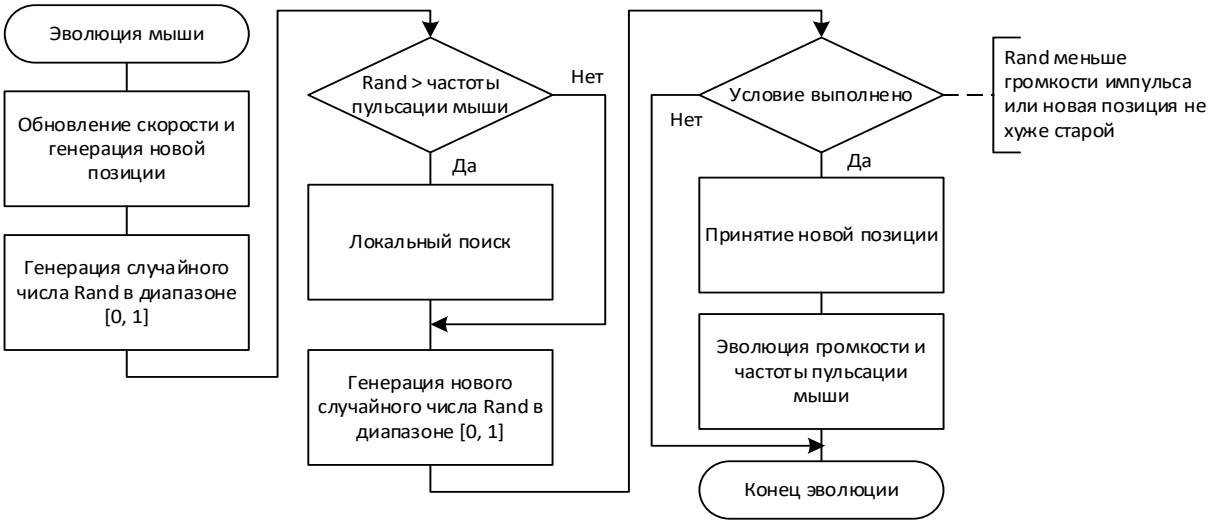

Рис. 2. Блок-схема эволюции летучей мыши

Математическая модель. Для математической формализации алгоритма введем следующие обозначения:

v_i — скорость передвижения i-й мыши;

x_i — текущая позиция i-й мыши;

$\lambda_i \in [\lambda_{min}, \lambda_{max}]$ — длина волны i-й мыши, $\lambda_{min} \geq 0$ и $\lambda_{max} > \lambda_{min}$;

$r_i \in [0, r_{max}]$ — частота пульсации сигнала i-й мыши, $r_{max} \in [0,1]$;

$a_i \in [a_{min}, a_{max}]$ — громкость сигнала i-й мыши, $a_{min}, a_{max} \in [0,1]$ и $a_{max} > a_{min}$;

t — текущий момент времени (номер итерации);

x_{best} — лучшая позиция мыши за все время.

При инициализации популяции для каждой мыши случайным образом задается текущая позиция, вектор скорости принимает нулевые значения, а громкость импульса a и частота пульсации r задаются начальными константами a_{max} и 0 соответственно.

На каждой итерации алгоритма происходит пересчет текущего положения мыши, которое вычисляется путем приращения вектора скорости:

$$x_i^{new} = x_i^{old} + v_i^{new}.$$

Вектор скорости формируется из двух слагаемых:

$$v_i^{new} = v_i^{old} + \left(x^{best} - x_i^{old}\right) \cdot \lambda_i^t.$$

Первое слагаемое называют инерционной компонентой [4]. Оно выполняет функцию памяти мыши о ее ранних перемещениях и предотвращает резкие изменения направления. Второе слагаемое представляет собой вектор, направленный на лучшую известную за все время работы алгоритма позицию, возмущенную случайным образом при помощи неотрицательного множителя λ, называемого длиной волны:

$$\lambda_i^{new} = \lambda_{min} + (\lambda_{max} - \lambda_{min}) \cdot Rand(0,1).$$

Длина волны характеризует интенсивность изменения скорости в сторону лучшего текущего решения и задается случайным образом для каждой итерации из заданного заранее диапазона.

Локальный поиск осуществляется путем случайной модификации лучшей на данный момент позиции:

$$x^{new} = x_{best} + Rand(-1,1) \cdot \langle a \rangle,$$

где a — средняя громкость сигнала мышей текущей итерации.

Громкость сигнала с количеством итераций затухает и вычисляется по следующей формуле:

$$a_i^{new} = a_i^{old} \cdot A,$$

где $A \in (0,1)$ — коэффициент затухания громкости.

Величина окрестности локального поиска зависит от громкости сигнала мышей всей популяции и со временем ее размер сокращается.

Частота пульсации с каждой итерацией повышается следующим образом:

$$r_i^{new} = r^{max} \cdot (1 - e^{-R \cdot t}),$$

где $R \in (0,1)$ — коэффициент усиления громкости.

На рис. 3 показаны графики зависимости громкости и частоты пульсации импульса мыши от числа итераций. Значение громкости сигнала характеризует вероятность принятия новой позиции в случае, если она не лучше текущей. Такое поведение расширяет область поиска и способ-

ствует диверсификации [3]. Значение частоты пульсации характеризует вероятность проведения локального поиска. Чем больше ее значение, тем реже осуществляется локальный поиск. Коэффициенты *A* и *R* позволяют контролировать плавность изменения громкости и частоты пульсации сигнала.

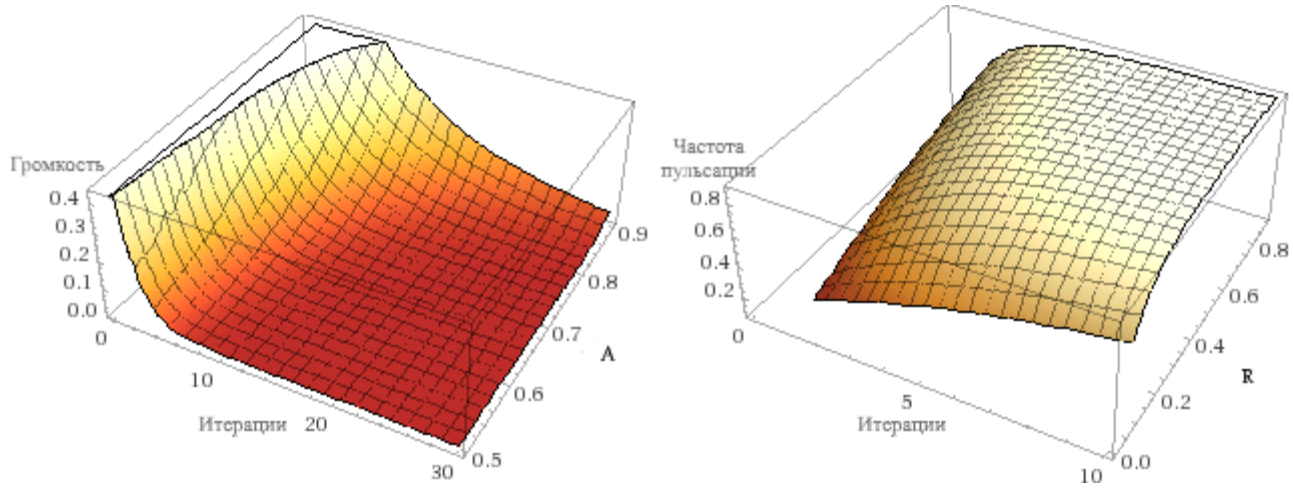

Рис. 3. Графики изменения громкости и частоты пульсации импульса с числом итераций

Программное средство. Для исследования эффективности алгоритма было реализовано программное решение под платформу *.NET Framework*, включающее три пакета: *BatAlgorithmCore*, *BatAlgorighmEnvironment* и *BatAlgorithmTestApplication*. Первый пакет содержит в себе реализацию алгоритма, а также определяет интерфейсы внешних зависимостей. Второй пакет содержит базовые реализации внешних зависимостей, необходимых для функционирования алгоритма. Последний пакет представляет собой приложение с графическим пользовательским интерфейсом, позволяющее тестировать алгоритм, а также визуализировать результаты работы (рис. 4).

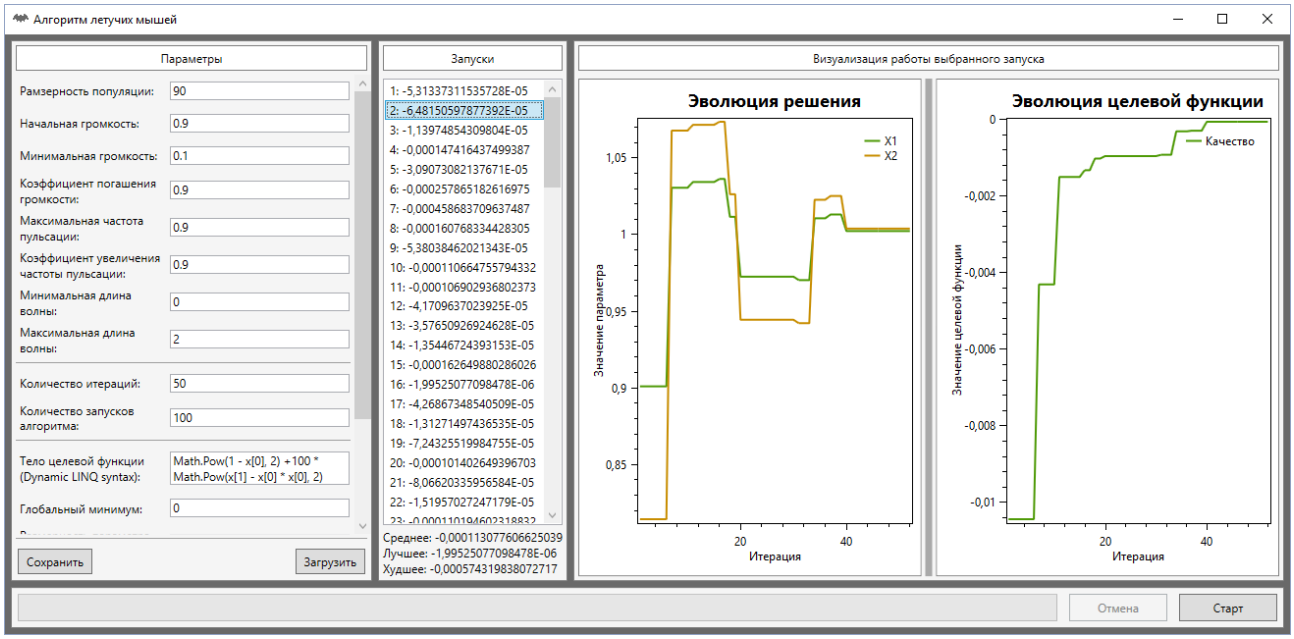

Рис. 4. Главное окно программного средства

Диаграмма классов, включающая основные пакеты программного средства, изображена на рис. 5.

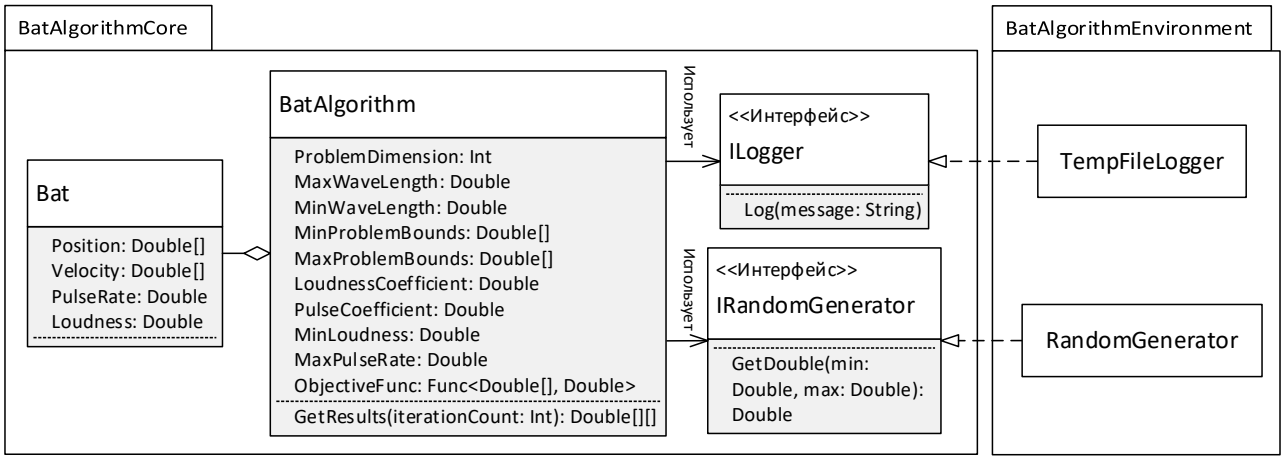

Рис. 5. Основные классы программного средства

Вычислительные эксперименты. Для анализа работы алгоритма были проведены вычислительные эксперименты с использованием общеизвестных математических функций Розенброка [5] и *EggCrate* [6].

Первая функция является унимодальной и описывается следующим образом:

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2.$$

Ее глобальный минимум имеет значение 0 и находится в точке с координатами (1, 1). Считается, что его поиск является нетривиальной задачей.

Вторая функция является мультимодальной и имеет глобальный минимум со значением 0 в точке с координатами (0, 0):

$$f(x, y) = x^2 + y^2 + 25(\sin^2 x + \sin^2 y).$$

В ходе экспериментов были получены результаты, которые приведены в таблице 1. Для каждого теста было выполнено 100 запусков алгоритма для снижения влияния случайных величин, а также просчитаны лучшее, худшее и среднее арифметическое значение результатов.

Таблица 1

Результаты вычислительных экспериментов

№	Функция	Ко-лич. мы шей	Число итер.	Границы длины волны	Коэф. погашения громкости / увеличения частоты пульсации	Лучший результат	Худший результат	Средний результат
1	Розенброк	10	100	[0, 2]	0.9/0.9	1.09e-06	0.4940	0.0101
2	Розенброк	30	100	[0, 2]	0.9/0.9	8.76e-08	0.48e-03	9.27e-05
3	Розенброк	90	100	[0, 2]	0.9/0.9	5.22e-08	0.17e-03	2.46e-05
4	Розенброк	100	1000	[1, 2]	0.9/0.9	1.74e-09	1.23e-05	1.26e-06
5	Розенброк	100	1000	[0, 1]	0.9/0.9	3.31e-10	1.2e-06	1.15e-07
6	Розенброк	30	100	[0, 2]	0.1/0.9	2.10e-07	0.48e-03	8.91e-05
7	Розенброк	30	100	[0, 2]	0.9/0.1	1.14e-07	0.60e-03	8.27e-05
8	Розенброк	30	100	[0, 2]	0.1/0.1	6.09e-07	0.90e-03	8.09e-05
9	Розенброк	30	500	[0, 2]	0.1/0.1	2.13e-09	4.01e-05	5.93e-06
10	Розенброк	30	500	[0, 2]	0.1/0.9	1.85e-08	3.81e-05	6.14e-06
11	Розенброк	30	500	[0, 2]	0.9/0.1	4.60e-08	8.22e-05	6.01e-06
12	Розенброк	10	300	[0, 2]	0.9/0.9	4.09e-07	0.53	0.54e-02
13	Розенброк	30	300	[0, 2]	0.9/0.9	1.05e-07	5.05e-05	1.11e-05
14	Розенброк	90	300	[0, 2]	0.9/0.9	3.77e-08	1.44e-05	2.70e-06
15	Розенброк	100	1000	[0, 2]	0.9/0.9	1.01e-10	9.62e-07	6.92e-07
16	EggCrate	10	100	[0, 2]	0.9/0.9	5.22e-06	18.9801	2.9554
17	EggCrate	30	100	[0, 2]	0.9/0.9	3.36e-06	9.4886	0.2853
18	EggCrate	90	100	[0, 2]	0.9/0.9	1.34e-07	0.27e-02	0.12e-03
19	EggCrate	100	1000	[1, 2]	0.9/0.9	5.51e-11	3,21e-07	3.28e-08
20	EggCrate	100	1000	[0, 1]	0.9/0.9	5.98e-14	1.05e-07	1.22e-08
21	EggCrate	30	100	[0, 2]	0.1/0.9	3.39e-07	18.9794	3.9857
22	EggCrate	30	100	[0, 2]	0.9/0.1	3.86e-06	9.4959	1.3290
23	EggCrate	30	100	[0, 2]	0.1/0.1	6.61e-07	37.9296	6.5004
24	EggCrate	30	500	[0, 2]	0.9/0.9	4.48e-10	9.4883	0.3794
25	EggCrate	30	500	[0, 2]	0.1/0.9	3.73e-11	9.4883	3.1215
26	EggCrate	30	500	[0, 2]	0.9/0.1	4.51e-09	9.4883	1.2335
27	EggCrate	10	300	[0, 2]	0.9/0.9	7.56e-08	9.4910	2.9414
28	EggCrate	30	300	[0, 2]	0.9/0.9	1.35e-08	9.4884	0.6642
29	EggCrate	90	300	[0, 2]	0.9/0.9	4.28e-10	2.41e-05	1.57e-06
30	EggCrate	100	1000	[0, 2]	0.9/0.9	1.14e-11	7.84e-08	8.93e-09
31	Розенброк	100	3000	[0, 2]	0.9/0.9	4.41e-10	2.23e-06	1.87e-07
32	EggCrate	100	3000	[0, 2]	0.9/0.9	4.60e-14	4.62e-09	3.80e-10
33	EggCrate	180	150	[0, 2]	0.9/0.9	1.34e-09	7.45e-05	7.64e-06
34	EggCrate	300	90	[0, 2]	0.9/0.9	1.19e-08	1.92e-03	1.44e-05

На основе экспериментов был проведен анализ влияния на итоговый результат следующих параметров алгоритма: размерность популяции, число итераций, границы длины волны, коэффициент погашения громкости, коэффициент увеличения частоты пульсации.

Тесты 1–3, 12–14, 16–18, 9, 24 позволяют оценить соотношение числа итераций и количества агентов в популяции. Однозначно можно утверждать, что размер популяции ниже 10 агентов дает худшие результаты в сравнении с увеличением количества мышей хотя бы до 30, если при этом соотнести сложность произведенных вычислений и точность результатов. В случае аналогичного сравнения популяций с количеством агентов равным 30 и 90, можно сказать, что для унимодальной функции разница не ощутима, а вот для функции *EggCrate* большее количество мышей позволяет более широко исследовать доступное пространство, и как следствие дойти до глобального минимума, минуя множество локальных. На основе тестов 24, 29, 33, 34 можно сделать вывод, что в случае *EggCrate* количество агентов от 90 и больше показывают примерно одинаковые результаты при этом рост числа итераций дает возможность получит более точный результат, но средний показатель точности ухудшается, рост числа агентов повышает средний результат, но отрицательно влияет на лучший. Стоит заметить, что такая закономерность имеет нижнюю границу по количеству итераций и эта цифра зависит от особенностей решаемой проблемы. В приведенном случае с функцией *EggCrate* понижение числа итераций ниже 80 ухудшает итоговый результат, так как агенты популяции проходят слишком маленький путь и не успевают накопить достаточное количество информации.

Значение диапазона границ длины волны влияет на величину одного шага перемещения мыши. На основе экспериментов 4–5, 15, 19–20, 30 можно сделать вывод, что увеличение шага исключительно в большую сторону, используя диапазон [1, 2], как с унимодальной, так и с мультимодальной функцией однозначно ухудшает результат. Допустимость как уменьшения шага, так и его увеличения, используя диапазон [0, 2], дает более точные результаты, но самый лучший эффект наблюдается при уменьшении шага исключительно в меньшую сторону, используя диапазон [0, 1].

Значение коэффициентов погашения громкости и увеличения частоты пульсации позволяют соотносить показатели интенсификации и диверсификации. На основе тестов 2, 6–11, можно сделать вывод, что для нахождения минимума унимодальной функции основным показателем является интенсификация, то есть плавное повышение частоты пульсации имеет приоритет. Что касается громкости сигнала, то в данном случае его стремительное падение исключает принятие позиций с худшим значением по сравнению с текущим. В случае мультимодальной функции тесты 17, 21–26 показывают, что в приоритете находится плавное погашение громкости эхолокационного сигнала. Интенсификация же в данном случае замедляет поиск глобального минимума, поэтому лучшим значением коэффициента повышения частоты пульсации для данной функции является — 0.9.

Заключение. В данной работе был реализован и исследован «алгоритм летучих мышей», который может быть задействован для решения множества оптимизационных задач. Алгоритм отлично работает для задач минимизации как унимодальных, так и мультимодальных функций.

В ходе работы было произведено программное конструирование данного алгоритма, в результате чего была создана библиотека под платформу *.NET*, которая может стать основой для сторонних приложений, а также тестовое приложение с графическим пользовательским интерфей-

ком, которое позволяет проверить работоспособность алгоритма, визуализировать результат работы, а также подобрать подходящие параметры для поставленной задачи.

В ходе вычислительных экспериментов были выявлены влияния основных параметров алгоритма, конфигурация которых позволяет легко подстроить алгоритм для решения различных оптимизационных задач.

В качестве перспектив дальнейших исследований можно предположить создание и проведение анализа различных модификаций алгоритма, а также адаптации его различных оптимизационных задач.

Библиографический список.

1. Роевой интеллект [Электронный ресурс] / Википедия. — Режим доступа: https://ru.wikipedia.org/wiki/Роевой_интеллект (дата обращения: 01.05.16).

2. Xin-She Yang, A New Metaheuristic Bat-Inspired Algorithm / Xin-She Yang. — University of Cambridg, 2010. — 10 с.

3. Xin-She Yang, Nature-Inspired Optimization Algorithms / Xin-She Yan. — Middlesex University London. — 1-е издание. — Elsevier, 2014, 265 с.

4. Карпенко, А. П. Современные алгоритмы поисковой оптимизации / А. П. Карпенко. — Москва: изд-во МГТУ им. Н.Э. Баумана, 2012. — 265 с.

5. Rosenbrock, H. H. An automatic method for finding the greatest or least value of a function / H. H. Rosenbrock // The Computer Journal. — 1960. — Т. 3. — С. 175–184.

6. N-D Test Functions E [Электронный ресурс] / Infinity 77. — Режим доступа: http://infinity77.net/global_optimization/test_functions_nd_E.html (дата обращения: 04.05.16).