УДК 519.85
Doi: 10.31772/2587-6066-2018-19-3-386-395

Для цитирования: Вахнин А. В., Сопов Е. А. Новый метод группировки переменных для задач параметрической оптимизации большой размерности // Сибирский журнал науки и технологий. 2018. Т. 19, № 3. С. 386–395. Doi: 10.31772/2587-6066-2018-19-3-386-395

For citation: Vakhnin A. V., Sopov E. A. [A new method of grouping variables for large-scale global optimization problems]. *Siberian Journal of Science and Technology*. 2018, Vol. 19, No. 3, P. 386–395 (In Russ.). Doi: 10.31772/2587-6066-2018-19-3-386-395

НОВЫЙ МЕТОД ГРУППИРОВКИ ПЕРЕМЕННЫХ ДЛЯ ЗАДАЧ ПАРАМЕТРИЧЕСКОЙ ОПТИМИЗАЦИИ БОЛЬШОЙ РАЗМЕРНОСТИ

А. В. Вахнин*, Е. А. Сопов

Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева
Российская Федерация, 660037, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31
*E-mail: alexeyvah@gmail.com

Сложность и размерность прикладных задач глобальной параметрической оптимизации стремительно увеличиваются с каждым годом. Стоит отметить, что практические задачи оптимизации являются сложными и зачастую рассматриваются как модель «черного ящика» по причине того, что исчерпывающий анализ проблемы затруднен или невозможен, а частичная информация о проблеме редко является полезной. Эффективным инструментом для решения задач оптимизации типа «черный ящик» являются эвристические алгоритмы прямого поиска. В последние десятилетия исследователи разработали множество эвристических алгоритмов для решения задач глобальной оптимизации большой размерности.

Предложен новый подход, который получил название DECC-RAG. Алгоритм DECC-RAG базируется на оригинальном методе группировки переменных (случайная адаптивная группировка) для применения метода кооперативной коэволюции. В основе предложенного метода группировки переменных лежит следующая идея: после заданного количества вычислений целевой функции, применяя структуру кооперативной коэволюции для алгоритма SaNSDE, находится половина субкомпонентов с худшими значениями пригодностей, в данных субкомпонентах происходит случайное перемешивание индексов переменных.

Эффективность алгоритма DECC-RAG проверялась на 20 эталонных тестовых задачах из набора LSGO CEC'2010 и 15 задачах из набора LSGO CEC'2013. Размерность задач равнялась 1000. Результаты численных экспериментов показывают, что предложенный алгоритм (DECC-RAG) превосходит некоторые другие современные эволюционные алгоритмы на задачах глобальной оптимизации большой размерности из LSGO CEC'2010 и LSGO CEC'2013.

Ключевые слова: оптимизация, большая размерность, эволюционные алгоритмы, кооперативная коэволюция.

A NEW METHOD OF GROUPING VARIABLES FOR LARGE-SCALE GLOBAL OPTIMIZATION PROBLEMS

A. V. Vakhnin*, E. A. Sopov

Reshetnev Siberian State University of Science and Technology
31, Krasnoyarsky Rabochy Av., Krasnoyarsk, 660037, Russian Federation
*E-mail: alexeyvah@gmail.com

Complexity and dimensionality of real-world optimization problems are rapidly increasing year by year. A lot of real-world optimization problems are complex, thus researchers consider these problems as 'black box' models due to the fact that the analysis of the problem is complicated or completely impossible, and partial information about the problem is rarely useful. Heuristic search algorithms have become an effective tool for solving such 'black box' optimization problems. In recent decades, many researchers have designed a lot of heuristic algorithms for solving large-scale global optimization (LSGO) problems.

In this paper, we proposed an innovative approach, which is called DECC-RAG. The approach is based on an original method of grouping variables (random adaptive grouping (RAG)) for cooperative cooperation framework. The RAG method uses the following idea: after a specified number of fitness evaluation in the cooperative coevolution with

the SaNSDE algorithm, we choose a half of subcomponents with the worst fitness values and randomly mix indices of variables in these subcomponents.

We have evaluated the DECC-RAG algorithm with 20 LSGO benchmark problems from the IEEE CEC'2010 and on 15 LSGO benchmark problems from the IEEE CEC'2013 competitions. The dimensionality of benchmark problems was equal to 1000. The experimental results have shown that the proposed method of optimization (DECC-RAG) outperforms some well-known algorithms on the large-scale global optimization problems from LSGO CEC'2010 and LSGO CEC'2013.

Keywords: optimization, large-scale, evolution algorithms, cooperative coevolution.

Введение. Прикладные задачи глобальной оптимизации с каждым годом становятся все сложнее. Большинство реальных задач оптимизации имеют большую размерность. Данный тип задач сложен для решения по ряду причин: пространство поиска увеличивается экспоненциально по мере увеличения количества переменных, свойства пространства поиска могут меняться, зачастую стоимость оценки решения высока. В 2002 году NASA опубликовало статью [1], где были продемонстрированы тенденции роста размерности реальных задач глобальной оптимизации. Если раньше большая размерность в задачах глобальной оптимизации означала пару десятков переменных, то сегодня их более тысячи. Существует множество проблем глобальной оптимизации большой размерности из разных областей [2–5]: datamining, инженерия, нейроинформатика, химия и др. Стоит обратить внимание на то, что большинство реальных задач оптимизации относятся к типу «черный ящик» (Black-Box (BB)). Особенность данного типа задач заключается в том, что мы не имеем информации о свойствах решаемой задачи, которые могут быть использованы при поиске глобального экстремума, мы можем получить только значения целевой функции $f(\bar{x})$ для точки $\bar{x}$ из пространства поиска. При решении задач типа «черный ящик» свою эффективность продемонстрировали эволюционные алгоритмы (ЭА) [6; 7].

Классическая задача оптимизации в общем случае может быть сформулирована следующим образом [8]:

$$f(\bar{x}) = f(x_1, x_2, ..., x_n) \to \min_{\bar{x} \in X} / \max, \quad (1)$$

$$x_i^L \le x_i \le x_i^U, i = \overline{1, n}, \quad (2)$$

$$g_j(x_1, x_2, ..., x_n) \le 0, j = \overline{1, m}, \quad (3)$$

$$h_k(x_1, x_2, ..., x_n) = 0, k = \overline{1, l}. \quad (4)$$

В уравнении (1) f обозначает целевую функцию, X является допустимым множеством решений, n – целое число, обозначающее размерность задачи оптимизации. В уравнении (2) x_i^L и x_i^U обозначают нижнюю и верхнюю границу интервала поиска соответственно. Уравнения (3) и (4) обозначают ограничения типа неравенства и равенства соответственно. В данной статье рассматриваются задачи безусловной оптимизации, на которые не накладываются ограничения типа (3) и (4).

Задачи глобальной параметрической оптимизации моделей типа «черный ящик», в которых число переменных равно 1000 и более, принято называть задачами глобальной оптимизации большой размерности (Large-Scale Global Optimization (LSGO)). В [9] показано, что наилучшие результаты для задач LSGO достигаются в классе бионических алгоритмов прямого поиска. В частности, высокую эффективность демонстрируют различные эволюционные алгоритмы.

Подходы к решению задачи LSGO на основе эволюционных алгоритмов

1. Дифференциальная эволюция (Differential Evolution). Storn и Price представили оригинальный подход для параметрической оптимизации в 1995 году, он получил название Differential Evolution (DE) [10]. DE является разновидностью эволюционного алгоритма, поэтому в ходе его работы используется итерационная процедура, которая состоит из повторяющегося применения операторов мутации, скрещивания, селекции. Более детально с данным алгоритмом можно ознакомиться в [11].

2. Самонастраивающаяся дифференциальная эволюция с поиском в окрестности (Self-adaptive differential evolution with neighborhood search (SaNSDE)). SaNSDE – самонастраивающаяся дифференциальная эволюция с поиском в окрестности [12]. На основе классического алгоритма DE в 2008 году была предложена его модификация. Данный алгоритм был выбран по той причине, что в нем присутствует самонастройка его параметров в ходе работы. Это очень важно, так как эффективность работы любого ЭА напрямую зависит от выбранных параметров. Для DE основными параметрами являются тип мутации, F, CR. F – параметр, характеризующий максимально возможное расстояние, на которое может расширяться область поиска оптимума по одной переменной за один шаг эволюции. Зачастую значения параметра F находятся в интервале [0; 2]. CR – параметр, отвечающий за частоту скрещивания: чем выше его значение, тем больше переменных решения будут замещены новыми мутантными значениями, значения CR находятся в интервале [0; 1]. Особенность алгоритма SaNSDE заключается в том, что алгоритм фиксирует успешное и неуспешное применение того или иного типа мутации, значения параметров F, CR, после определенного количества поколений пересчитывает вероятность выбора типа мутации, значения CR и значения F.

3. Кооперативная коэволюция (Cooperative Coevolution). Одним из наиболее известных и часто применяемых подходов для решения задач большой размерности является декомпозиция на основе метода кооперативной коэволюции.

Кооперативная коэволюция (Cooperative Coevolution (CC)) является эволюционной структурой (рис. 1), которая делит вектор задачи оптимизации на несколько субкомпонентов и оптимизирует их независимо

с помощью ЭА (до момента кооперации) для решения задачи оптимизации. Впервые данный метод предложили Potter и De Jong [13] применительно к генетическому алгоритму. СС на сегодняшний день является эффективным инструментом при решении сепарабельных задач оптимизации большой размерности. Ниже представлен псевдокод кооперативной коэволюции:

Псевдокод кооперативной коэволюции

1: *Произвести декомпозицию целевого вектора на m субкомпонентов меньшей размерности*;
2: *Присвоить i = 1 и начать новый цикл*;
3: *Оптимизировать i-ю субкомпоненту с помощью ЭА*;
4: *Если i < m, i = i+1 и перейти к шагу 3, иначе к шаг 5*;
5: *Остановить алгоритм, если условие остановки выполнено, иначе перейти к шагу 2 для следующего цикла.*

Стоит заметить, что на сегодняшний день существует множество подходов, направленных на решение задач оптимизации большой размерности. Данные подходы можно разделить на две группы: подходы с использованием СС и подходы, не использующие декомпозицию вектора решения. Большую эффективность имеют те методы, которые используют СС для решения задачи оптимизации. При использовании СС появляются следующие новые задачи: выбор количества субкомпонентов (на сколько частей делить вектор решения), выбор размера каждой субкомпоненты и какие переменные объединять в одну группу. В связи с этим сегодня популярны следующие группировки, используемые в СС: группировка, где заранее задается условие, по которому происходит группировка (static grouping) [14], динамическая случайная группировка переменных (random dynamic grouping) [15], адаптивная группировка (learning dynamic grouping) [16], где проводится серия экспериментов по приращению значений каждой переменной, на основании полученных результатов переменные группируются.

DECC-RAG алгоритм. Сведения, изложенные выше, послужили основой для разработки нового ЭА для решения задач оптимизации большой размерности. Главная идея алгоритма заключается в использовании оригинального метода группировки переменных для эволюционной структуры СС с самонастраивающимся ЭА (SaNSDE). Самонастройка необходима, так как решаемая задача оптимизации представлена в виде модели «чёрный ящик», мы не знаем функциональной зависимости между переменными, следовательно, параметры должны адаптироваться во время работы алгоритма.

Подход СС является эффективным, но только в том случае, если правильно сгруппировать переменные. Было решено, что в основе метода группировки переменных должна присутствовать случайность, так как при использовании группировки типа learning dynamic grouping не всегда удается разделить переменные на истинное количество субкомпонент [16], но при этом будет затрачено большое количество ресурсов (вычисление функции пригодности) в ходе выделения субкомпонентов.

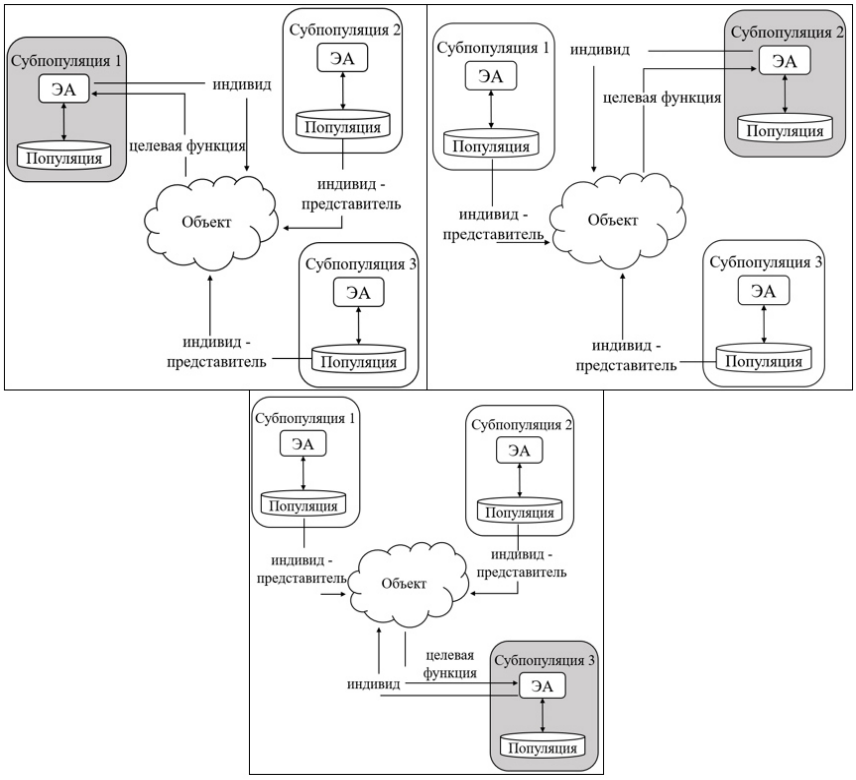

Рис. 1. Схематичное представление работы кооперативной коэволюции

Fig. 1. Schematic representation of cooperative coevolution

Помимо этого, может случиться так, что переменные сгруппируются верно, но оставшееся количество вычислений функции останется малым – внушительная часть бюджета вычислений уходит на группировку переменных. Количество группируемых переменных должно быть одинаковым для каждой субкомпоненты, данное решение было принято по причине того, что иначе придётся решать следующие дополнительные задачи:

1. Выбор различного количества индивидов. Например, для ЭА, оптимизирующего субкомпоненты вектора решения, состоящие из 5 переменных и 500 переменных, необходимо разное количество индивидов в популяции. Получается неравномерное распределение ресурсов алгоритма.

2. Минимальное и максимальное количество группируемых переменных. Как упоминалось неоднократно выше, задача глобальной оптимизации представлена в виде модели типа «чёрный ящик», т. е. неизвестно, какое значение выбирать в качестве минимального и максимального размера группы.

Предложенный оригинальный метод группировки Random Adaptive Grouping (RAG) работает следующим образом: *D*-мерный вектор решения разбивается на *m s*-мерных субкомпонент. Случайно, по равномерному закону распределения группируем переменные в одинаковые группы. Так как мы изначально не знаем, насколько хорошо себя показала та или иная группировка, необходимо провести определенное количество вычислений функции пригодности *T* (запустить работу ЭА на каждой субкомпоненте), после чего найти *m*/2 субкомпонент с худшими значениями функции пригодности индивидов-представителей, по равномерному закону распределения случайно перегруппировать переменные в данных группах. Далее нужно произвести сброс параметров ЭА в худших *m*/2 субкомпонентах после перегруппировки переменных, это делается из-за того, что после перегруппировки переменных конкретному ЭА необходимо будет уже решать совершенно другую задачу. Данный метод группировки переменных получил название RAG, а алгоритм, разработанный на его основе – DECC-RAG. Псевдокод алгоритма DECC-RAG представлен ниже:

Псевдокод алгоритма DECC-RAG

1: *Задать FEV_global, T, FEV_local = 0;*
2: *n-мерный целевой вектор случайно разделить на m s-мерных субкомпонент*;
3: *Случайно перемешать индексы всех переменных*;
5: 4: *i = 1*; *Эволюционировать i-ю субкомпоненту, используя SaNSDE*;
6: *Если i < m, то i++ и перейти к шагу 5, иначе* – к *шагу 7*;
7: *Найти best_solution$_i$ для каждой субкомпоненты*;
8: *Если (FEV_local < T), то перейти к шагу 4, иначе* – к *шагу 9*;
9: *Найти m/2 субкомпонент с худшими значениями целевой функции и случайно перемешать индексы переменных в данных субкомпонентах, обновить значения параметров SaNSDE в данных m/2 субкомпонентах, FEV_local = 0*;
10: *Если (FEV > 0), то перейти к шагу 4, иначе – к шагу 11*;
11: *Произвести возврат лучшего найденного значения (best_solution).*

FEV_local и *FEV_global* – счетчики для оценки функции пригодности внутри адаптационного периода и для всего алгоритма соответственно. В нашем случае мы использовали следующие значения параметров алгоритма: *NP* = 50 (размер популяции для каждой субкомпоненты), $m = 10$, $T = 3 \times 10^5$. *T* – параметр, который представляет собой количество *FEVs* (оценок функций) перед случайным перемешиванием худших *m*/2 субкомпонент.

Описание эталонных тестовых задач и критериев оценки эффективности алгоритмов LSGO. Алгоритмы DE, SaNSDE и DECC-RAG тестировались на 20 тестовых функциях большой размерности из набора IEEE (The Institute of Electrical and Electronic Engineers) LSGO (Large-Scale Global Optimization) CEC'2010 (Congress on Evolutionary Computation) [17] и 15 тестовых функциях большой размерности из набора IEEE LSGO CEC'2013 [18]. Данные тестовые функции специально были наделены свойствами, которые присущи реальным задачам оптимизации большой размерности.

Полученные результаты работы алгоритмов (DE, SaNSDE, DECC-RAG) на задачах LSGO CEC'2010 и LSGO CEC'2013 сравнивали с результатами работы известных алгоритмов оптимизации (DMS-L-PSO [19], DECC-G [15], MLCC [20], DECC-DG [16]) для задач большой размерности. Результаты работы DMS-L-PSO, DECC-G, MLCC, DECC-DG на задачах из LSGO CEC'2010 и LSGO CEC'2013 были взяты из [21].

Для проведения экспериментов использовались правила соревнования LSGO CEC'2010 и LSGO CEC'2013, а именно:

– размерность тестовых задач оптимизации $D = 1000$;

– количество независимых запусков алгоритма для каждой тестовой задачи равно 25;

– количество вычислений функции пригодности в каждом независимом запуске 3×10^6;

– после проведения 25 независимых запусков для тестовой функции вычисляется медианное значение (Median) и среднеквадратическое отклонение (Std).

Программная реализация алгоритмов (DE, SaNSDE и DECC-RAG) осуществлялась в среде разработки приложений Visual Studio – 2015 на языке программирования C++. В ходе проведения экспериментов значительное время тратится на вычисление функций пригодности. Тестирование алгоритмов проводилось с использованием параллельных вычислений с помощью технологии OpenMP.

Результаты численных экспериментов. В табл. 1, 2 приведено время (в секундах), затраченное на вычисление 10 000 раз данной тестовой функции на 1 потоке на процессоре AMD Ryzen 71700x. При проведении экспериментов использовался 16-поточный процессор для распараллеливания экспериментов – каждая тестовая функция вычислялась на отдельном потоке. Если не использовать данный прием, то только на разрешенное правилами конкурса количество вычислений функции пригодности в рамках данной работы потребовалось бы порядка 847,6 ч (35,5 дня).

В нашем случае время удалось сократить до 80,3 ч.

В табл. 3, 4 продемонстрированы результаты работы алгоритмов на тестовых функциях из набора LSGO CEC'2010 и LSGO CEC'2013 соответственно. Первый столбец обозначает номер тестовой функции, последующие столбцы – результаты работы того или иного алгоритма. В каждой ячейке таблицы содержатся два значения – медианное и среднеквадратическое отклонение, которые получены по результатам 25 независимых запусков.

Таблица 1

Время вычисления 10 000 функций пригодности в секундах на задачах из набора LSGO CEC'2010

Func. №	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10
Время	0,396	0,209	0,21	0,52	0,334	0,34	0,309	0,307	1,312	1,134
Func. №	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20
Время	1,139	0,112	0,126	2,219	2,016	2,04	0,077	0,133	0,072	0,1

Таблица 2

Время вычисления 10 000 функций пригодности в секундах на задачах из набора LSGO CEC'2013

Func. №	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10
Время	7,76	8,08	8,1	8,36	8,76	8,88	3,19	10,17	10,59	10,62
Func. №	F11	F12	F13	F14	F15	–	–	–	–	–
Время	10,1	0,126	10,04	10,03	7,71	–	–	–	–	–

Таблица 3

Результаты численных экспериментов алгоритмов на задачах (F1–F20) из набора LSGO CEC'2010, *D* = 1000

№ func	DECC-RAG	DE	SaNSDE	DMS-L-PSO	DECC-G	MLCC	DECC-DG
F1	**2,69E-18**	4,19E+08	2,00E+04	1,61E+07	3,53E-07	1,66E-14	1,42E+02
	5,10E-18	2,75E+08	2,04E+06	1,41E+06	1,44E-07	2,97E-12	4,66E+04
F2	7,33E+02	7,38E+03	2,80E+03	5,53E+03	1,32E+03	**2,43E+00**	4,46E+03
	7,52E+01	3,02E+02	1,67E+02	5,38E+02	2,55E+01	1,52E+00	1,87E+02
F3	1,64E+00	1,95E+01	1,47E+01	1,56E+01	1,14E+00	**6,24E-10**	1,66E+01
	1,77E-01	8,60E-02	4,31E-01	1,08E-01	3,35E-01	1,12E-06	3,02E-01
F4	9,50E+11	8,78E+12	2,82E+12	**4,32E+11**	2,46E+13	1,78E+13	5,08E+12
	3,50E+11	3,43E+12	1,01E+12	8,05E+10	8,14E+12	5,47E+12	1,89E+12
F5	1,54E+08	**7,96E+07**	9,00E+07	9,35E+07	2,50E+08	5,11E+08	1,52E+08
	4,41E+07	2,12E+07	8,22E+06	9,04E+06	6,84E+07	1,07E+08	2,15E+07
F6	2,04E+01	2,09E+01	1,27E+06	3,66E+01	4,71E+06	1,97E+07	**1,64E+01**
	5,75E+06	6,84E+06	8,12E+05	1,21E+01	1,03E+06	4,37E+06	3,45E-01
F7	**2,90E+02**	3,08E+08	1,90E+05	3,47E+06	6,57E+08	1,15E+08	9,20E+03
	8,22E+02	1,76E+08	6,18E+04	1,16E+05	5,40E+08	1,45E+08	1,26E+04
F8	1,78E+07	2,53E+08	**8,16E+06**	2,02E+07	9,06E+07	8,82E+07	1,62E+07
	7,43E+08	3,88E+08	2,22E+07	1,88E+06	2,64E+07	3,40E+07	2,63E+07
F9	6,17E+07	5,56E+08	2,31E+08	**2,08E+07**	4,35E+08	2,48E+08	5,52E+07
	8,72E+06	8,20E+07	9,95E+07	1,58E+06	4,87E+07	2,16E+07	6,45E+06
F10	**3,25E+03**	7,72E+03	9,40E+03	5,09E+03	1,02E+04	3,97E+03	4,47E+03
	1,88E+02	2,47E+02	2,82E+02	4,26E+02	3,13E+02	1,45E+03	1,29E+02
F11	2,16E+02	1,88E+02	1,74E+02	1,68E+02	2,59E+01	1,98E+02	**1,02E+01**
	1,31E+01	6,40E+00	1,51E+01	1,90E+00	1,73E+00	1,12E+00	8,71E-01
F12	8,88E+03	5,59E+05	4,03E+05	**2,83E+01**	9,69E+04	1,01E+05	2,58E+03
	1,15E+03	6,91E+04	4,83E+04	9,88E+00	9,55E+03	1,57E+04	1,08E+03
F13	**1,56E+03**	1,01E+09	2,52E+04	1,03E+05	4,59E+03	2,12E+03	5,06E+03
	3,81E+03	6,79E+08	1,61E+05	6,18E+04	4,16E+03	4,70E+03	3,65E+03
F14	2,01E+08	1,60E+09	7,78E+08	**1,25E+07**	9,72E+08	5,71E+08	3,46E+08
	2,07E+07	1,52E+08	1,28E+08	1,62E+06	7,52E+07	5,50E+07	2,42E+07
F15	**5,16E+03**	7,75E+03	1,06E+04	5,48E+03	1,24E+04	8,67E+03	5,86E+03
	3,60E+02	2,55E+02	4,34E+02	3,46E+02	8,24E+02	2,07E+03	1,05E+02
F16	4,13E+02	3,77E+02	3,73E+02	3,18E+02	6,92E+01	3,96E+02	**7,50E-13**
	3,05E+01	4,32E+00	1,12E+01	2,04E+00	6,43E+00	5,76E+01	6,25E-14
F17	1,68E+05	1,04E+06	8,68E+05	**4,75E+01**	3,11E+05	3,47E+05	4,02E+04
	1,17E+04	7,94E+04	6,84E+04	1,15E+01	2,24E+04	3,11E+04	2,29E+03
F18	**4,96E+03**	4,15E+10	5,83E+05	2,50E+04	3,54E+04	1,59E+04	1,47E+10
	6,35E+03	1,70E+10	1,81E+08	1,10E+04	1,53E+04	9,48E+03	2,03E+09
F19	2,23E+06	2,96E+06	1,93E+06	2,03E+06	**1,14E+06**	2,04E+06	1,75E+06
	1,93E+05	4,01E+05	1,89E+05	1,41E+05	6,23E+04	1,42E+05	1,10E+05
F20	1,84E+03	5,25E+10	2,80E+05	**9,82E+02**	4,34E+03	2,27E+03	6,53E+10
	5,04E+02	1,58E+10	1,37E+07	1,40E+01	8,25E+02	2,26E+02	6,97E+09

Таблица 4

Результаты численных экспериментов алгоритмов на задачах (F1–F15) из набора LSGO CEC'2013, D = 1000

№ func	DECC-RAG	DE	SaNSDE	DMS-L-PSO	DECC-G	MLCC	DECC-DG
F1	**1,88E-16**	5,28E+08	8,53E+05	1,97E+09	2,06E-06	9,07E-14	6,03E+02
	3,11E-16	2,92E+08	8,25E+06	1,27E+08	4,27E-06	4,38E-09	1,81E+04
F2	1,40E+03	2,46E+04	2,06E+04	8,61E+03	1,30E+03	**3,57E+00**	1,28E+04
	1,39E+02	1,72E+03	8,50E+02	4,88E+02	3,63E+01	1,73E+00	7,20E+02
F3	2,03E+01	2,16E+01	2,10E+01	2,08E+01	2,02E+01	**2,00E+01**	2,14E+01
	2,33E-02	6,07E-03	8,05E-02	1,66E-01	6,18E-03	2,76E-04	1,45E-02
F4	**1,11E+10**	1,11E+11	2,93E+10	2,97E+11	2,00E+11	1,99E+11	7,33E+10
	7,50E+09	3,43E+10	1,13E+10	7,25E+10	1,22E+11	1,26E+11	2,82E+10
F5	**3,50E+06**	4,62E+06	4,88E+06	3,92E+06	8,44E+06	1,17E+07	5,81E+06
	7,09E+05	7,57E+05	5,29E+05	5,82E+05	1,14E+06	3,46E+06	3,83E+05
F6	1,06E+06	1,06E+06	1,06E+06	**9,98E+05**	1,06E+06	1,05E+06	1,06E+06
	8,62E+02	1,36E+03	1,34E+03	5,20E+03	1,84E+03	4,13E+03	1,07E+03
F7	2,40E+08	1,04E+09	**2,20E+08**	1,22E+09	1,04E+09	1,15E+09	4,25E+08
	3,68E+08	6,95E+08	1,88E+08	7,64E+08	4,48E+08	1,07E+09	1,92E+08
F8	4,76E+14	2,15E+15	2,26E+14	**1,68E+14**	7,90E+15	8,18E+15	2,89E+15
	1,89E+14	1,04E+15	1,32E+14	5,09E+14	3,18E+15	6,18E+15	1,85E+15
F9	**2,32E+08**	4,27E+08	4,81E+08	3,50E+08	5,86E+08	8,85E+08	4,95E+08
	5,21E+07	6,45E+07	4,69E+07	4,61E+07	9,76E+07	2,92E+08	3,18E+07
F10	9,42E+07	9,42E+07	9,40E+07	**9,11E+07**	9,30E+07	9,27E+07	9,45E+07
	5,91E+05	1,85E+05	2,51E+05	1,06E+06	6,16E+05	6,07E+05	2,46E+05
F11	3,45E+09	2,50E+11	**3,18E+09**	9,44E+10	1,26E+11	1,90E+11	3,81E+10
	6,42E+10	1,95E+11	1,12E+10	7,53E+10	7,15E+10	1,53E+11	4,33E+10
F12	**1,82E+03**	4,64E+10	1,49E+07	5,22E+04	4,19E+03	2,36E+03	1,68E+11
	3,85E+02	1,61E+10	3,79E+08	5,52E+04	7,83E+02	7,51E+02	2,24E+10
F13	7,19E+09	1,52E+10	**6,92E+09**	1,32E+10	8,67E+09	9,94E+09	2,08E+10
	4,77E+09	4,47E+09	3,15E+09	6,58E+09	2,78E+09	3,73E+09	5,53E+09
F14	3,80E+10	3,42E+11	5,27E+10	2,21E+11	1,28E+11	2,06E+11	**1,56E+10**
	6,70E+10	1,54E+11	3,39E+10	1,26E+11	5,86E+10	8,54E+10	1,44E+10
F15	1,59E+07	7,32E+09	6,12E+07	1,54E+07	1,13E+07	1,57E+07	**9,52E+06**
	4,50E+07	2,13E+11	4,13E+07	3,45E+06	1,26E+06	1,90E+06	2,30E+06

Произведем попарное сравнение независимых выборок (лучших найденных значений в 25 независимых запусках), используя критерий Уилкоксона – непараметрический статистический тест, используемый для проверки статистических различий между двумя независимыми выборками. В табл. 5, 6 занесены результаты теста статистической значимости различий между результатами работы алгоритмов DECC-RAG vs DE и DECC-RAG vs SaNSDE на тестовых функциях из набора LSGO CEC'2010 соответственно. В табл. 7, 8 занесены результаты теста на статистическую значимость различий между результатами работы алгоритмов DECC-RAG vs DE и DECC-RAG vs SaNSDE на тестовых функциях из набора LSGO CEC'2013 соответственно. Если в соответствующей ячейке (табл. 5–8) стоит знак «+», то это означает, что результаты работы алгоритмов статистически различны и лучшее найденное медианное решение – у первого алгоритма, если стоит знак «–», то это означает, что результаты работы алгоритмов статистически различны и лучшее найденное медианное решение – у второго алгоритма, если стоит знак «≈» – результаты работы алгоритмов статистически не различаются на данной тестовой функции.

На рис. 2–4 продемонстрированы графики, на которых отражены усредненные результаты работы алгоритмов DE, SaNSDE и DECC-RAG на некоторых (1, 2, 9, 12, 14, 18) тестовых функциях из набора LSGO CEC'2010: по оси абсцисс – количество вычислений функции пригодности, по оси ординат – среднее значение функции пригодности. На рис. 5 и 6 продемонстрированы результаты работы алгоритмов DE, SaNSDE и DECC-RAG на некоторых тестовых функциях (1, 2, 3, 12) из набора LSGO CEC'2013. На рис. 7 и 8 в виде столбчатых диаграмм отображен средний ранг алгоритмов по всем тестовым функциям из эталонных наборов LSGO CEC'2010 и LSGO CEC'2013 соответственно. Ранг вычислялся по медианным результатам из табл. 3 и 4 в зависимости от места, которое занял алгоритм на той или иной функции: чем меньше медианное значение, тем меньше значение ранга. На двух эталонных наборах предложенный алгоритм DECC-RAG имеет наименьший средний ранг.

Анализируя полученные численные результаты работы алгоритма DECC-RAG (рис. 7, 8), можно заключить, что он превосходит некоторые известные алгоритмы глобальной оптимизации большой размерности (DMS-L-PSO, DECC-G, MLCC, DECC-DG) на 20 эталонных тестовых функциях из набора LSGO CEC'2010 и 15 эталонных функциях из набора LSGO CEC'2013.

Статистической значимости различий между результатами работы алгоритмов не наблюдается

на F_6 из LSGO CEC'2010 у DECC-RAG vs SaNSDE, на F_6 у DECC-RAG vs DE и на F_7, F_{10}, F_{11} из LSGO CEC'2013.

На всех остальных тестовых функциях наблюдается статистическая значимость различий между результатами работы алгоритмов.

Таблица 5

Статистическая значимость работы алгоритмов (DECC-RAG vs DE) на 20 тестовых функциях LSGO CEC'2010

F_1 +	F_2 +	F_3 +	F_4 +	F_5 –	F_6 +	F_7 +	F_8 +	F_9 +	F_{10} +
F_{11} –	F_{12} +	F_{13} +	F_{14} +	F_{15} +	F_{16} +	F_{17} +	F_{18} +	F_{19} +	F_{20} +

Таблица 6

Статистическая значимость работы алгоритмов (DECC-RAG vs SaNSDE) на 20 тестовых функциях LSGO CEC'2010

F_1 +	F_2 +	F_3 +	F_4 +	F_5 –	F_6 ≈	F_7 +	F_8 –	F_9 +	F_{10} +
F_{11} +	F_{12} +	F_{13} +	F_{14} +	F_{15} +	F_{16} –	F_{17} +	F_{18} +	F_{19} –	F_{20} +

Таблица 7

Статистическая значимость работы алгоритмов (DECC-RAG vs DE) на 15 тестовых функциях LSGO CEC'2013

F_1 +	F_2 +	F_3 +	F_4 +	F_5 +	F_6 ≈	F_7 +	F_8 +
F_9 +	F_{10} +	F_{11} +	F_{12} +	F_{13} +	F_{14} +	F_{15} +	– –

Таблица 8

Статистическая значимость работы алгоритмов (DECC-RAG vs SaNSDE) на 15 тестовых функциях LSGO CEC'2013

F_1 +	F_2 +	F_3 +	F_4 +	F_5 +	F_6 +	F_7 ≈	F_8 –
F_9 +	F_{10} ≈	F_{11} ≈	F_{12} +	F_{13} –	F_{14} +	F_{15} +	– –

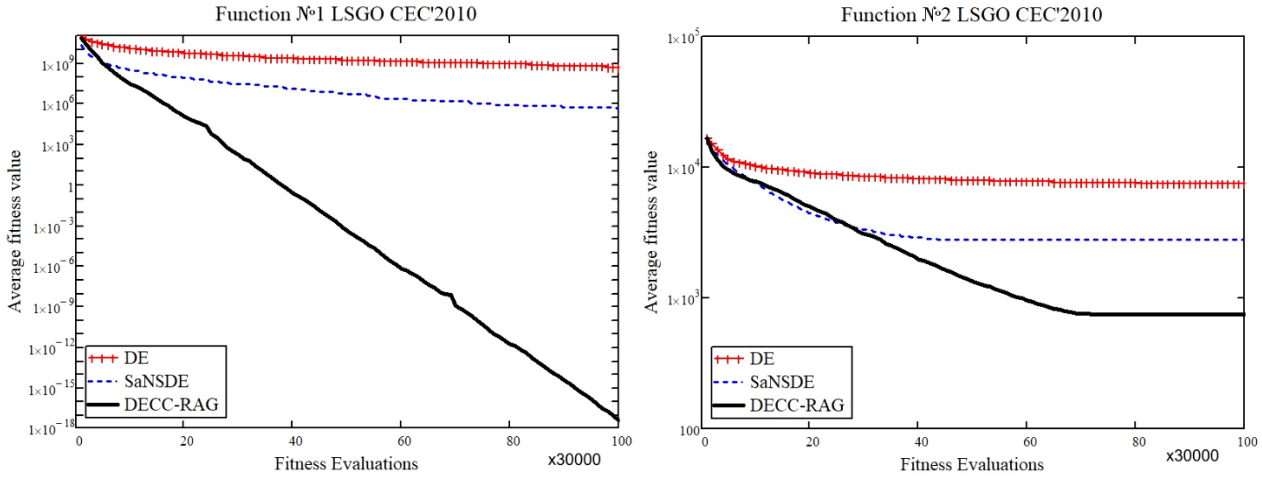

Рис. 2. Сходимость DE, SaNSDE и DECC-RAG на задачах (№ 1, 2) из набора LSGO CEC'2010

Fig. 2. The convergence graphs of DE, SaNSDE and DECC-RAG applied to F_1, F_2 from LSGO CEC'2010

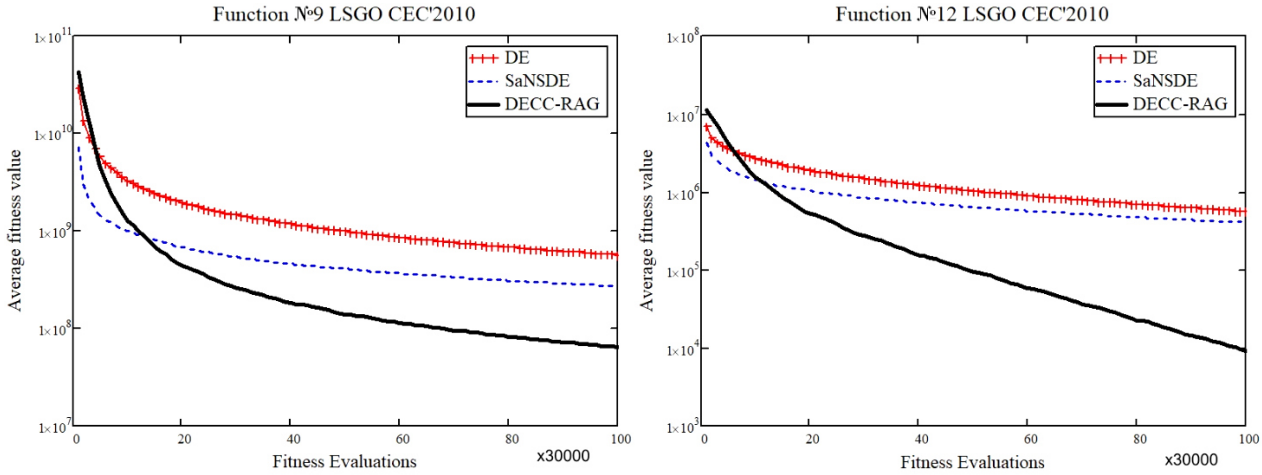

Рис. 3. Сходимость DE, SaNSDE и DECC-RAG на задачах (№ 9, 12) из набора LSGO CEC'2010

Fig. 3. The convergence graphs of DE, SaNSDE and DECC-RAG applied to F_9, F_{12} from LSGO CEC'2010

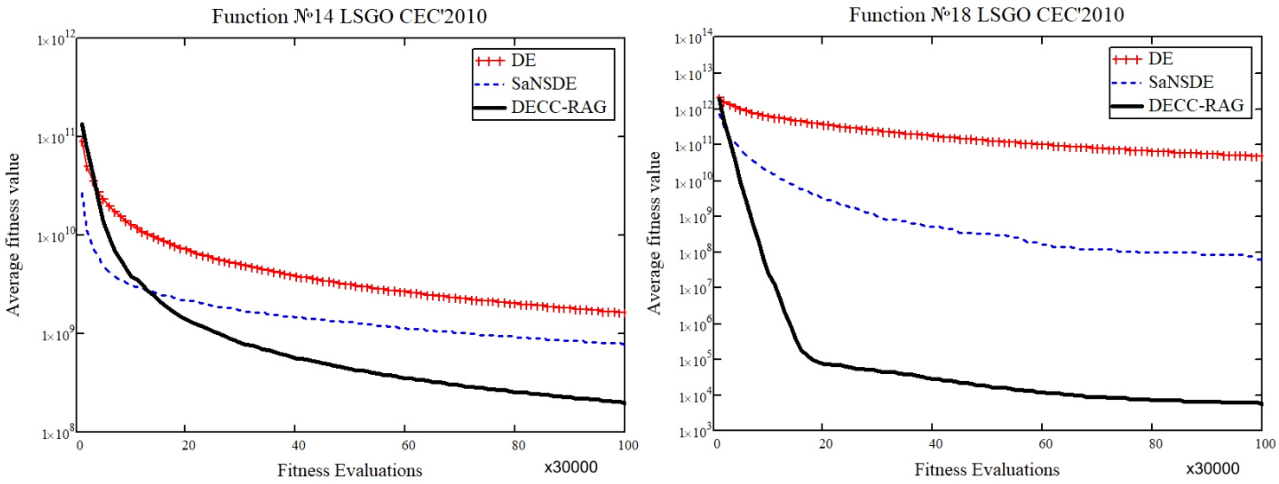

Рис. 4. Сходимость DE, SaNSDE и DECC-RAG на задачах (№ 14, 18) из набора LSGO CEC'2010

Fig. 4. The convergence graphs of DE, SaNSDE and DECC-RAG applied to F_{14}, F_{18} from LSGO CEC'2010

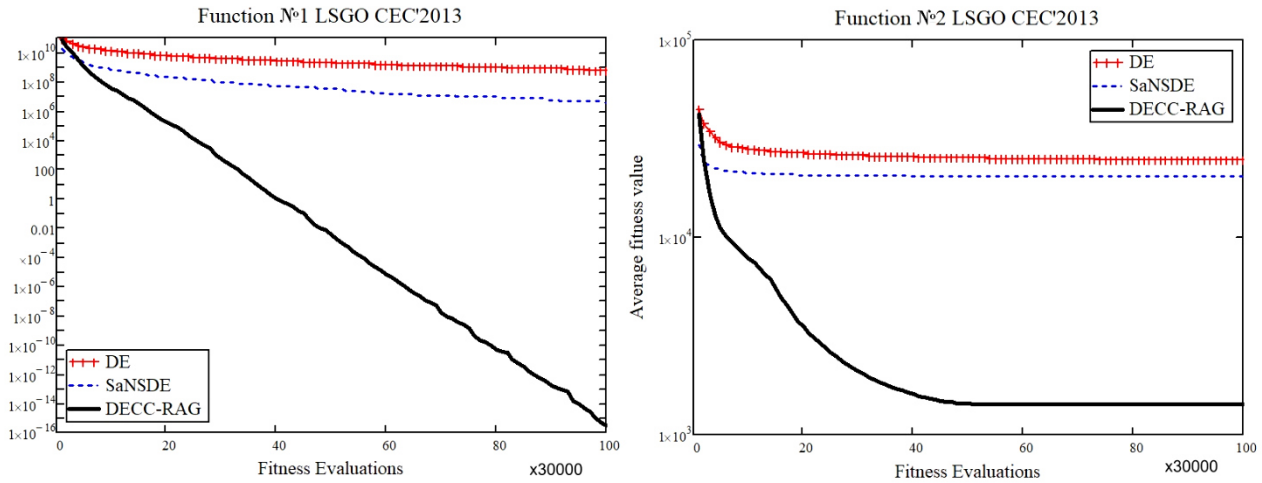

Рис. 5. Сходимость DE, SaNSDE и DECC-RAG на задачах (№ 1, 2) из набора LSGO CEC'2013

Fig. 5. The convergence graphs of DE, SaNSDE and DECC-RAG applied to F_1, F_2 from LSGO CEC'2013

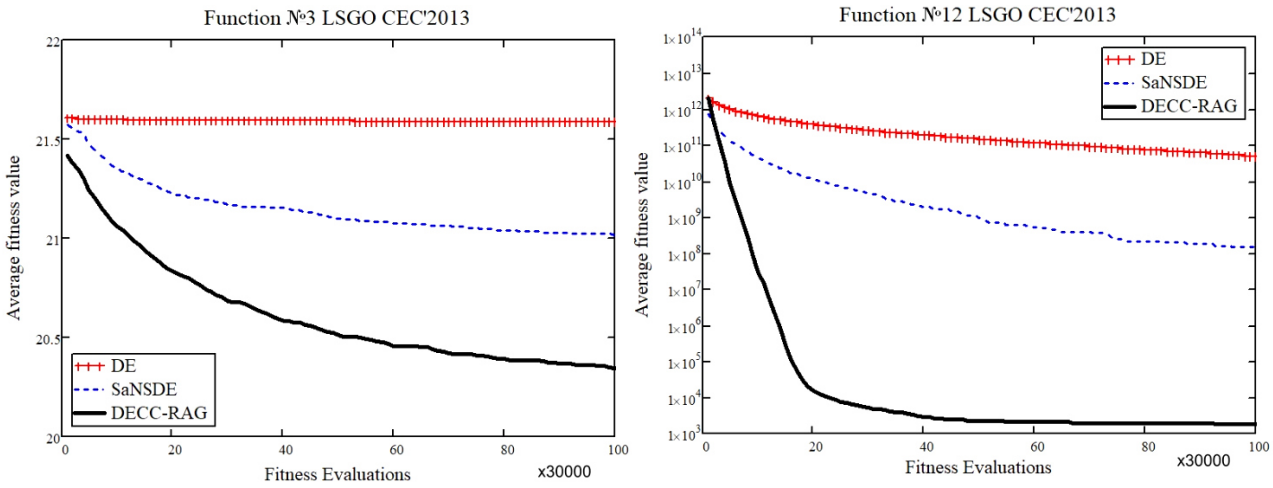

Рис. 6. Сходимость DE, SaNSDE и DECC-RAG на задачах (№ 3, 12) из набора LSGO CEC’2013

Fig. 6. The convergence graphs of DE, SaNSDE and DECC-RAG applied to F_3, F_{12} from LSGO CEC’2013

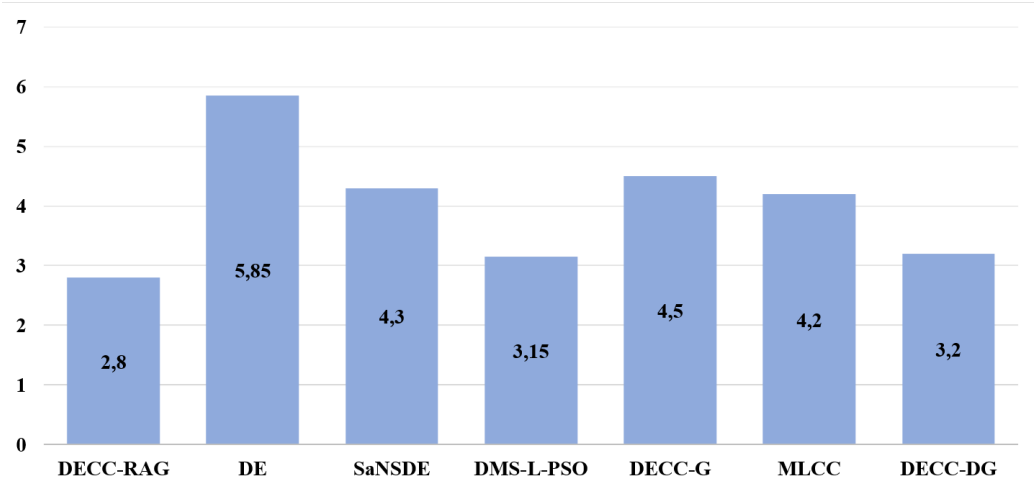

Рис. 7. Средний ранг алгоритмов на LSGO CEC’2010

Fig. 7. Average rank of algorithms on LSGO CEC’2010

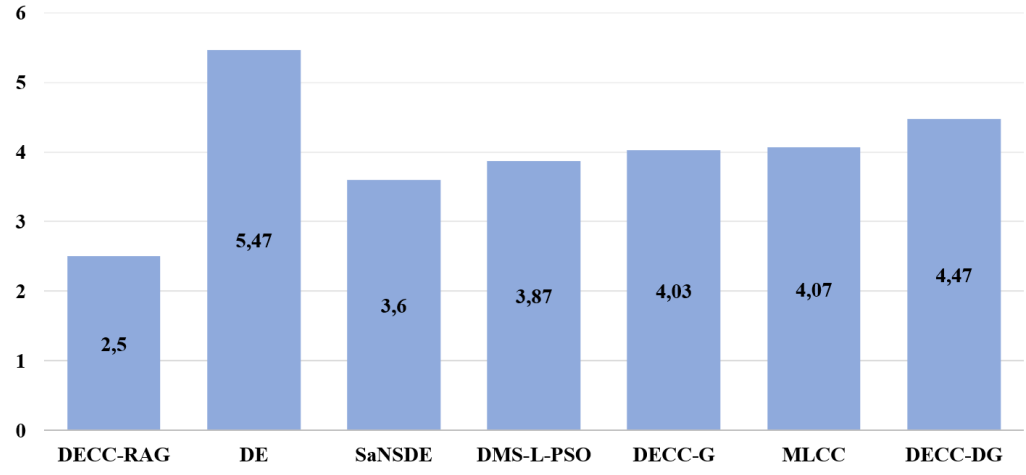

Рис. 8. Средний ранг алгоритмов на LSGO CEC’2013

Fig. 8. Average rank of algorithms on LSGO CEC’2013

Заключение. В данной статье предлагается новый эволюционный алгоритм, названный DECC-RAG, для задач глобальной оптимизации большой размерности. В основе предложенного алгоритма лежит случайная адаптивная группировка переменных для структуры кооперативной коэволюции. В качестве ЭА, который оптимизирует каждую субкомпоненту, мы применили эффективный самонастраивающийся эволюционный алгоритм SaNSDE.

Тестирование эффективности DECC-RAG проводилось на эталонных задачах из наборов LSGO CEC'2010 и LSGO CEC'2013. Алгоритм DECC-RAG превзошел некоторые известные алгоритмы, которые специально разрабатывались для задач глобальной оптимизации большой размерности. В дальнейших работах будет ставиться вопрос об улучшении эффективности предложенного алгоритма DECC-RAG для задач глобальной оптимизации большой размерности.

Благодарности. Работа поддержана Министерством образования и науки Российской Федерации, № 2.1676.2017/ПЧ.

Acknowledgments. This work was supported by the Ministry of Education and Science of Russian Federation, № 2.1676.2017/ПЧ.

References

1. Vanderplaats G. N. Very large scale optimization. *National Aeronautics and Space Administration (NASA), Langley Research Center*. 2002, 55 p.

2. Shuhei Kimura, Kaori Ide, Aiko Kashihara, Makoto Kano, Mariko Hatakeyama, Ryoji Masui, Noriko Nakagawa, Shigeyuki Yokoyama, Seiki Kuramitsu, Akihiko Konagaya. Inference of s-system models of genetic networks using a cooperative coevolutionary algorithm. *Bioinformatics (Oxford, England)*. 2005, Vol. 21, Iss. 7, P. 1154–1163.

3. Wei-Po Lee, Yu-Ting Hsiao. Inferring gene regulatory networks using a hybrid ga–pso approach with numerical constraints and network decomposition. *Information Sciences*. 2012, Vol. 188, P. 80–99.

4. Bo Jiang, Ning Wang, Cooperative bare-bone particle swarm optimization for data clustering. *Soft Computing*. 2014, Vol. 18, Iss. 6, P. 1079–1091.

5. Lin Lin, Mitsuo Gen, Yan Liang. A hybrid EA for high-dimensional subspace clustering problem. *Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014*. 2014, P. 2855–2860.

6. Bäck T. Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. *Oxford University Press*. 1996, 319 p.

7. Jansen T. Analyzing Evolutionary Algorithms: The Computer Science Perspective. *Natural Computing Series, Springer*. 2013, 264 p.

8. Vanderplaats G. N. Numerical Optimization Techniques for Engineering Design: with Applications. *McGraw-Hill Book Company*. 1984, 333 p.

9. S. Mahdavi, M. E. Shiri, and S. Rahnamayan. Metaheuristics in large-scale global continues optimization: A survey. *Information Sciences*. 2015, Vol. 295, P. 407–428.

10. Storn R., Price K. Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. *International Computer Science Institute-Publications-Tr*. 1995, P. 1–12.

11. Das S., Suganthan P. N. Differential evolution: a survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation.* 2011, Vol. 15, Iss. 1, P. 4–31.

12. Zhenyu Yang, Ke Tang, Xin Yao. Self-adaptive differential evolution with neighborhood search. *2008 IEEE Congress on Evolutionary Computation, CEC 2008*. 2008, P. 1110–1116.

13. Mitchell A. Potter, Kenneth A. De Jong. A cooperative coevolutionary approach to function optimization. *Parallel Problem Solving from Nature – PPSN III*. 1994, P. 249–257.

14. Mitchell A. Potter, Kenneth A. De Jong. Cooperative Coevolution: An Architecture for Evolving Coadapted Subcomponents. *Evolutionary Computation*. 2000, Vol. 8, Iss. 1, P. 1–29.

15. Yang Z., Tang K., Yao X. Large scale evolutionary optimization using cooperative coevolution. *Information Sciences*. 2008, Vol. 178, Iss. 15, P. 2985–2999.

16. Mohammad Nabi Omidvar, Xiaodong Li, Yi Mei, Xin Yao. Cooperative co-evolution with differential grouping for large scale optimization. *IEEE Transactions on Evolutionary Computation*. 2014, Vol. 18, Iss. 3, P. 378–393.

17. Tang K., Li X., Suganthan P. N., Yang Z., Weise T. Benchmark functions for the CEC 2010 special session and competition on large-scale global optimization. *Technical report, Univ. of Science and Technology of China, Nature Inspired Computation and Applications Laboratory*. 2010, 23 p.

18. Li X., Tang K., Omidvar M. N., Yang Z., Qin K. Benchmark Functions for the CEC'2013 Special Session and Competition on Large-Scale Global Optimization. *Technical report, RMIT University, Evolutionary Computing and Machine Learning (ECML).* 2013, 23 p.

19 Liang J. J., Suganthan P. N. Dynamic multi-swarm particle swarm optimizer with local search. *2005 IEEE Congress on Evolutionary Computation*. 2005, Vol. 1, P. 522–528.

20. Yang, Tang K., Yao X. Multilevel cooperative coevolution for large scale optimization. *2008 IEEE Congress on Evolutionary Computation, CEC 2008*. 2008, P. 1663–1670.

21. Yang Q., Chen W. N., Da Deng J., Li Y., Gu T., Zhang J. A Level-based Learning Swarm Optimizer for Large Scale Optimization. *Accepted by IEEE Transactions on Evolutionary Computation*. 2017, 16 p.