НАУЧНОЕ ИЗДАНИЕ МГТУ ИМ. Н. Э. БАУМАНА
НАУКА и ОБРАЗОВАНИЕ
Эл № ФС77 - 48211. Государственная регистрация №0421200025. ISSN 1994-0408
электронный научно-технический журнал

# Ко-эволюционный алгоритм глобальной оптимизации на основе алгоритма роя частиц

**# 11, ноябрь 2013**
**DOI: 10.7463/1113.0619595**
**Карпенко А. П., Воробьева Е. Ю.**
УДК 519.6

Россия, МГТУ им. Н.Э. Баумана
apkarpenko@mail.ru
vorobeva_ey@mail.ru

## Введение

Основная идея ко-эволюционных алгоритмов (ко-алгоритмов) оптимизации состоит в следующем. Одновременно в пространстве поиска задачи оптимизации эволюционируют несколько субпопуляций, каждая из которых решает исходную оптимизационную задачу и имеет свою стратегию оптимизации, которая определяется значениями свободных параметров субалгоритма (алгоритма, который управляет данной субпопуляцией). В процессе решения задачи субпопуляции конкурируют между собой за некоторый ресурс (в данной работе – число вычислений значений оптимизируемой функции), который в процессе работы ко-алгоритма перераспределяется в пользу более эффективной из них [1].

Целью ко-гибридизации алгоритмов поисковой оптимизации является повышение их эффективности, которое достигается тем, что в процессе ко-эволюции, по сути, решается задача мета-оптимизации этих алгоритмов – то есть задача определения значений их свободных параметров, которые являются наилучшими для данной задачи оптимизации [2].

Чаще всего при ко-гибридизации в качестве субалгоритмов используют генетический алгоритм и его модификации [3]. Помимо большого числа достоинств генетический алгоритм обладает и рядом

недостатков. Так, индивиды в популяции генетического алгоритма не имеют памяти и, по-сути, не «сотрудничают» между собой. В отличие от этого, в используемом нами для ко-гибридизации популяционном алгоритме роя частиц (*Particle Swarm Optimization, PSO*) агенты (частицы) «помнят» свое лучшее положение за всю предысторию поиска. Кроме того, в алгоритме *PSO* информация о лучшем из найденных роем решении раньше или позже становится известной всем частицам, то есть в этом алгоритме имеет место «сотрудничество» между частицами.

Данная статья является продолжением работы [1], в которой предложен двухпопуляционный ко-алгоритма *Co-PSO*, построенный на основе алгоритмов *PSO*, использующих топологии соседства «клика» и «кольцо».

Целью работы является разработка многопопуляционного ко-алгоритма *PSO,* программная реализация этого алгоритма, исследование эффективности разработанного алгоритмического и программного обеспечения.

В первом разделе работы приводим постановку задачи глобальной оптимизации и схему базового алгоритма *PSO*. Во втором разделе представляем разработанный нами ко-алгоритм *Co-PSO* и *MatLab*-программное обеспечение, реализующее этот алгоритм. Третий раздел посвящен исследованию эффективности разработанного алгоритмического и программного обеспечения. В четвертом разделе рассматриваем решение с помощью разработанных средств задачи оптимального управления космическим аппаратом на этапе его спуска в атмосфере Земли. В заключении сформулированы основные результаты работы и очерчены перспективы ее развития.

## 1. Постановка задачи глобальной оптимизации и схема алгоритма роя частиц для ее решения

Рассматриваем детерминированную непрерывную задачу глобальной условной минимизации

$$\min_{X \in D \subseteq R^{|X|}} f(X) = f(X^*) = f^*, \quad (1)$$

где $f(X) \in R^1$ – скалярная целевая функция (критерий оптимальности), $f(X^*) = f^*$ – искомое минимальное значение целевой функции, $X = (x_1, x_2, ..., x_{|X|}) \in R^{|X|}$ – $|X|$-мерный вектор варьируемых параметров, $D$ – множество допустимых значений этого вектора (область поиска); $R^{|X|}$ – $|X|$-мерное арифметическое пространство.

Множество частиц в рое обозначаем $S = \{s_i,\ i \in [1:|S|]\}$, где $|S|$ – число частиц (размер популяции). На итерации $t$ координаты частицы $s_i$ определяют $|X|$-мерные векторы ее координат $X_{i,t}$ и «скорости» $V_{i,t}$. Начальные координаты и «скорости» всех частиц роя полагаем заданными и равными $X_{i,0} = X_i^0$, $V_{i,0} = V_i^0$ соответственно.

Основные итерационные формулы алгоритма *PSO* имеют вид

$$X_{i,t+1} = X_{i,t} + V_{i,t+1}, \quad (2)$$

$$V_{i,t+1} = \alpha V_{i,t} + U[0;\beta] \otimes (X_{i,t}^b - X_{i,t}) + U[0;\gamma] \otimes (X_{g,t} - X_{i,t}). \quad (3)$$

Здесь $U[a;b]$ - $|X|$-мерный вектор псевдослучайных чисел, равномерно распределенных в интервале $[a;b]$; $\otimes$ - символ покомпонентного умножения векторов; $X_{i,t}^b$ - вектор координат частицы $s_i$ с наилучшим (в смысле формулы (1)) значением целевой функции $f(X)$ за все время поиска $[0:t]$; $X_{g,t}$ - вектор координат соседней с данной частицы с наилучшим за то же время значением целевой функции; $\alpha$, $\beta$, $\gamma$ - свободные параметры алгоритма. Параметр $\alpha$ определяет «инерционные» свойства частицы, $\beta$ – ее «когнитивные» свойства, $\gamma$ – «социальность» частицы. Рекомендуемые значения параметров $\alpha$, $\beta$, $\gamma$ равны, соответственно, 0,7298; 1,49618; 1,49618 [4].

Соседство частиц в рое определяет их *топология соседства*, задаваемая неориентированным графом, в котором вершины графа соответствуют частицам роя, а ребра связывают непосредственных соседей. В алгоритме *PSO* обычно используют топологии

- «клика» (глобально оптимальная топология),
- «кольцо» (локально оптимальная топология),
- «двумерный тор» (топология фон Неймана),
- «кластер».

Топологии «клика» соответствует полносвязный граф, в котором соседями каждой из частиц $s_i$, $i \in [1:|S|]$ являются остальные $|S|-1$ частицы. Диаметр клики равен единице. В топологии «кольцо» соседями каждой из частиц являются всего две частицы, которые имеют соседние номера. Диаметр соответствующего графа равен $\left\lceil |S| / 2 \right\rceil$, где $\lceil \cdot \rceil$ - символ ближайшего целого большего.

Обзор большого числа модификаций алгоритма *PSO* представлен в работе [4].

## 2. Ко-алгоритм *Co-PSO* и его программная реализация

### 2.1. Схема ко-алгоритма *Co-PSO*

Поскольку речь идет о ко-алгоритмической гибридизации алгоритмов *PSO*, ко-эволюционирующие субпопуляции далее называем *суброями*. Суброи обозначаем $S_1, S_2, ..., S_{|\hat{S}|}$, где $|\hat{S}|$ - их число. Схема предлагаемого ко-алгоритма *Co-PSO* имеет следующий вид.

1) Задаем числа суброев, топологии соседства, используемые каждым из суброев, значения свободных параметров субалгоритмов.

2) Задаем значения свободных параметров ко-алгоритма:

- величина *ресурса* $n_f$ – максимально допустимое число испытаний (вычислений значений целевой функции);

- *интервал адаптации* $\Delta_a \in (0; T)$;

- величина *штрафа* $n_p$;

- минимально допустимый размер суброя $|S|_{\min}$ или его относительное значение

$$\eta_i = |S|_{\min} / |S_i|, \quad i \in [1 : |\hat{S}|].$$

3) Инициализируем суброи, то есть, присваиваем всем частицам каждого из суброев $S_i, i \in [1 : \hat{S}]$ начальные положения в пространстве поиска и начальные «скорости».

3) Реализуем асинхронное выполнение каждым суброем $\Delta_a$ независимых итераций.

4) Производим оценку эффективности суброев.

5) Проверяем выполнение условия останова. В случае выполнения этого условия, в качестве решения принимаем лучшее решение, найденное роем, и прекращаем вычисления.

6) Осуществляем перераспределение ресурса.

7) Реализуем миграцию лучших частиц. Частицы всех суброев объединяем в единый массив и сортируем в порядке убывания соответствующих значений целевой функции. Затем каждому из суброев из этого массива передаем число частиц, пропорциональное текущему размеру суброя.

8) Переходим к шагу 3.

*Оценку эффективности суброя* $S_i$ производим на основе вычисления текущих значений его *функции пригодности* $\varphi_i = \varphi_i(t)$, где $t \in [1 : \hat{t}]$ - номер текущей итерации; $\hat{t}$ - общее число итераций. Пригодность суброя есть мера его адаптации (приспособленности) к решаемой задаче оптимизации. Значение пригодности вычисляем по формуле

$$\varphi_i(t) = \sum_{\tau=0}^{\Delta_a - 1} \frac{\Delta_a - \tau}{\tau + 1} a_i(\tau), \quad i \in [1 : |\hat{S}|], \qquad (4)$$

где $a_i(\tau)=1$, если суброй $S_i$ на итерации $(t-\tau)$ включал в себя лучшую во всей популяции частицу, и $a_i(\tau)=0$ в противном случае. Заметим, что вес величины $a_i(\tau)$ определяется отношением $\frac{\Delta_a-\tau}{\tau+1}$, так что при $\tau=0$ (текущая итерация) этот вес максимален и равен интервалу адаптации $\Delta_a$. С уменьшение номера итерации $\tau$ вес величины $a_i(\tau)$ в сумме (4) быстро уменьшается [1].

*Перераспределение ресурса* производим на основе приспособленности суброев (4). Если текущее значение пригодности суброя $S_i$ максимально по сравнению с пригодностями остальных суброев, то этот суброй признаем победившим, а остальные суброи – проигравшими. Перераспределение ресурса осуществляем путем сокращения размера каждого из проигравших суброев на величину штрафа, определяемого величиной $n_p$, и увеличения размера победившего суброя на число, равное сумме потерь проигравших. Таким образом, общий размер роя не меняется [1].

Если размер проигравшего суброя оказывается меньше величины $|S|_{\min}$, то принимаем его равным этой величине. Таким образом, если $|S_i|$ – размер суброя $S_i$, проигравшего на текущем интервале адаптации, то на следующем интервале размер этого суброя будет равен величине

$$|S_i'| = \max\left(n_p\,|S_i|, |S|_{\min}\right), \quad i \in [1:|\hat{S}|].$$

Если же суброй $S_i$ выиграл на текущем интервале адаптации, то на следующем интервале его размер определяет выражение

$$|S_i'| = |S_i| + \sum_j \max\left(n_p\,|S_j|, |S|_{\min}\right), \quad i,\ j \in [1:|\hat{S}|],\ \ i \neq j\,.$$

### 2.2. Обзор других методов ко-гибридизации алгоритмов *PSO*

Отметим следующее обстоятельство. Задача глобальной оптимизации (1) поставлена нами как задача условной оптимизации. Однако канонический алгоритм *PSO* и рассмотренная выше схема ко-алгоритма *PSO*

не учитывают ограничений на вектор варьируемых параметров $X$, то есть, ориентированы на решение задачи безусловной оптимизации. Мы полагаем, что задача условной оптимизации (1) сведена к задаче безусловной оптимизации методом штрафных функций (п. 4).

Известно большое число приемов, позволяющих учитывать в популяционных алгоритмах оптимизации и, в частности, в алгоритме *PSO*, ограничения на компоненты вектора $X$. Такие приемы, по сути, модифицируют базовые популяционные алгоритмы и приводят к новым алгоритмам. В ко-алгоритмах ограничения на вектор $X$ могут быть учтены также на уровне ко-алгоритма. Приведем несколько примеров.

В работе [5] предложен ко-алгоритм *CPSO* (*Co-evolutionary PSO*) для случая, когда множество допустимых значений вектора варьируемых параметров $D$ определяют ограничения типа неравенств $g_j(X) \le 0, \quad j \in [1:|G|]$. Алгоритм *CPSO* использует *мультирой* $S_{1,1}, S_{1,2}, ..., S_{1,|G|}$, содержащий $|G|$ роев по $|S|$ частиц в каждом, а также рой $S_2$, число частиц в котором равно числу ограничивающих функций $|G|$. Размерность вектора параметров, ассоциированного с каждой из частиц роев $S_{1,i}$, равна $|X|$. В рое $S_2$ размерность того же вектора равна числу штрафных коэффициентов в штрафной функции, с помощью которой задача условной оптимизации сведена к задаче безусловной оптимизации. Рои $S_{1,1}, S_{1,2}, ..., S_{1,|G|}$ используются для ко-алгоритмического поиска решения указанной задачи безусловной оптимизации, а рой $S_2$ - для такого же поиска оптимальных значений коэффициентов штрафа.

Близкий подход к решению задачи условной оптимизации используется в работе [6], в которой наряду с ограничениями типа неравенств рассматриваются ограничения типа равенств вида $h_k(X) = 0, \quad k \in [1:|H|]$. При выполнении условий теоремы Куна-Таккера из этой теоремы следует, что исходная задача минимизации (1) эквивалентна

задаче безусловной минимизации соответствующей функции Лагранжа по вектору варьируемых параметров $X$ и такой же задаче максимизации этой функции по множителям Лагранжа. Для решения указанной минимаксной задачи используется два ко-эволюционирующих роя *PSO* $S_1, S_2$, их которых рой $S_1$ решает задачу минимизации, а рой $S_2$ - задачу максимизации.

Отличная от представленных выше схема ко-эволюционной гибридизации используется в работе [7]. Предложенный в этой работе алгоритм *CECBPSO* (*Co-Evolutionary Cultural Based Particle Swarm Optimization*), основан на *культурном алгоритме* (*Cultural algorithm*) [8].

### 2.3. Программная реализация ко-алгоритма *Co-PSO*

Программная реализация ко-алгоритма *Co-PSO* выполнена в среде пакета программ *MatLab R*2012a и на языке программирования этого пакета. Язык программирования *MatLab* представляет собой высокоуровневый интерпретируемый язык, основанный на операциях с векторами и матрицами. Язык включает в себя широкий набор функций, объектно-ориентированные возможности и интерфейсы к программам, написанным на других языках программирования. *MatLab*-программы могут быть оформлены как функции и скрипты. Программная реализация ко-алгоритма *PSO* выполнена в виде скриптов и также получила наименование *Co-PSO*.

Программа *Co-PSO* включает в себя расширяемые библиотеки тестовых функций и графов, определяющих топологии соседства, которые могут быть использованы субалгоритмами *PSO*. В программе могут быть заданы различные условия окончания итераций, но во всех случаях их число ограничено заданным пользователем значением $\hat{t}$.

Для повышения вероятности локализации глобального экстремума целевой функции, программа *Co-PSO* поддерживает метод мультистарта. По результатам заданного пользователем числа стартов $Z$ программа вычисляет оценку вероятности локализации глобального экстремума $\overline{\xi}$, а также

средние числа испытаний $\bar{n}_f$, итераций $\bar{t}_{end}$ и побед данного суброя $\bar{n}_w$, а также предельные размеры суброев-победителей $|S^-|$, $|S^+|$, размер суброя-победителя в момент окончания итераций $|S^*|$, величины свободных параметров $\alpha^*, \beta^*, \gamma^*$ суброя-победителя (если эти параметры были различны для различных суброев). Победителем считается тот из суброев, который обеспечил меньшее результирующее значение целевой функции.

В качестве результата работы программа *Co-PSO* может построить график изменения значений целевой функции с ростом числа итераций для суброя-победителя а также графики изменения размеров суброев в функции номера итерации.

## 3. Вычислительный эксперимент

Всюду далее в данном разделе имеется в виду использование в качестве условия окончания итерационного процесса его стагнацию в течение $t_\Delta = 20$ итераций, когда лучшее достигнутое значение целевой функции изменяется на величину, не превышающую $\varepsilon = 10^{-6}$.

В качестве тестовых функций используем широко известные унимодальную овражную функцию Розенброка, четырехэкстремальную функцию Химмельблау и многоэкстремальную функцию Растригина. Для функций Розенброка и Растригина задача решается в параллелепипеде $\Pi = \{\mathrm{X} \,|\, -b \le \mathrm{x_i} \le b;\ i \in [1:|\mathrm{X}|]\}$, где $b = 2{,}0$, а для функции Химмельблау – в аналогичном параллелепипеде при $b = 4{,}0$.

Во всех случаях вычислительный эксперимент выполнен для размерностей вектора варьируемых параметров $|X|$, равных 2, 4, 8, 16, 32 и 64.

Рассматриваем канонический алгоритм *PSO* и два варианта ко-алгоритма *PSO*, называемые $Co-PSO^T$, $Co-PSO^P$ и отличающиеся друг от друга используемыми субалгоритмами *PSO*.

Ко-алгоритм $Co-PSO^T$ построен на основе шести суброев, из которых два имеют топологию «клика», два – «кольцо» и два – динамическую топологию. Значения свободных параметров $\alpha, \beta, \gamma$ всех субалгоритмов *PSO* постоянны и равны рекомендованным (п. 1).

Ко-алгоритм $Co-PSO^P$ также построен на основе шести суброев, но все суброи в этом случае используют одну и ту же топологию соседства частиц. Значения свободных параметров $\alpha, \beta, \gamma$ указанных суброев случайным образом равномерно распределенные в интервале $[0; 2{,}0]$.

Свободные параметры ко-алгоритмов $Co-PSO^T$, $Co-PSO^P$ имеют следующие значения: интервал адаптации $\Delta_a = 9$; величина штрафа $n_p = 0{,}2$; минимально допустимый размер суброя $|S|_{\min} = 10$. Число частиц в каждой из субпопуляций обоих ко-алгоритмов принято равным 50. Используем мультистарт с числом запусков $Z = 30$. Как и в предыдущем исследовании, в качестве тестовых используем функции Розенброка, Химмельблау, Растригина.

### 3.1. Канонический алгоритм *PSO*

Исследование эффективности канонического алгоритма *PSO* выполнено с целью тестирования разработанного алгоритмического и программного обеспечения, а также с целью создания базы для последующей сравнительной оценки эффективности ко-алгоритма *Co-PSO*. Рассмотрен канонический алгоритм *PSO* с топологиями соседства «клика», «кольцо» и динамическая топология (когда, начиная с топологии «кольцо», через каждое фиксированное число итераций в граф соседства случайным образом добавляется одно ребро). В качестве значений свободных параметров алгоритма использованы их рекомендованные значения (п. 1). Число частиц в популяции принято равным $|S| = 50$. Используем мультистарт с числом запусков алгоритма $Z = 30$.

Оценки вероятности $\bar{\xi}$ локализации глобального экстремума в функции величины $|X|$ представлены в таблице 1. Аналогичная таблица 2 представляет средние числа испытаний $\bar{n}_f$.

Таблица 1 - Оценки вероятности $\bar{\xi}$ локализации глобального экстремума

| Тестовая функция | Топология | Размерность вектора $X$ | | | | | |
|---|---|---|---|---|---|---|---|
| | | 2 | 4 | 8 | 16 | 32 | 64 |
| Розенброка | клика | 100 | 93 | 90 | 80 | 67 | 64 |
| | кольцо | 80 | 73 | 60 | 53 | 50 | 43 |
| | динамическая | 87 | 84 | 73 | 64 | 57 | 60 |
| Химмельблау | клика | 97 | 84 | 70 | 64 | 57 | 53 |
| | кольцо | 100 | 73 | 77 | 70 | 53 | 43 |
| | динамическая | 93 | 80 | 73 | 64 | 50 | 40 |
| Растригина | клика | 77 | 67 | 60 | 53 | 47 | 32 |
| | кольцо | 100 | 93 | 87 | 77 | 70 | 64 |
| | динамическая | 73 | 77 | 64 | 57 | 53 | 47 |

Таблица 2 - Оценки среднего числа испытаний $\bar{n}_f$

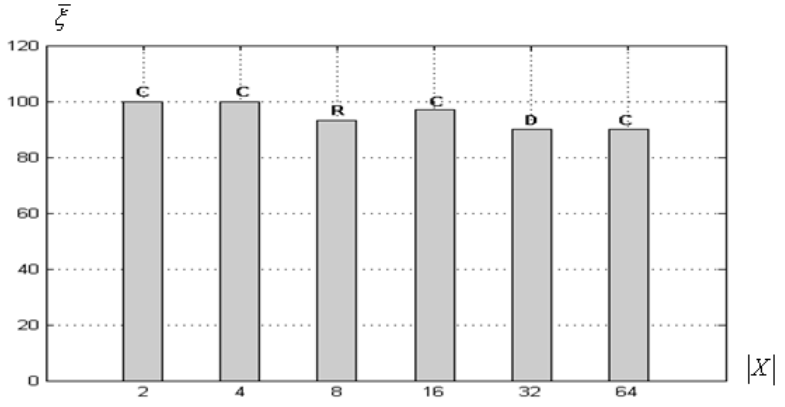
| Тестовая функция | Топология | Размерность вектора $X$ | | | | | |
|---|---|---|---|---|---|---|---|
| | | 2 | 4 | 8 | 16 | 32 | 64 |
| Розенброка | клика | 3700 | 5150 | 12500 | 12650 | 14700 | 16400 |
| | кольцо | 9200 | 14750 | 16600 | 19500 | 21150 | 30700 |
| | динамическая | 7600 | 11800 | 13200 | 16400 | 19750 | 28200 |
| Химмельблау | клика | 12100 | 14600 | 14000 | 17700 | 21400 | 27650 |
| | кольцо | 11700 | 14900 | 17250 | 16200 | 22700 | 25700 |
| | динамическая | 10650 | 17700 | 22400 | 21600 | 25050 | 30250 |
| Растригина | клика | 6700 | 13000 | 15200 | 19100 | 25750 | 32900 |
| | кольцо | 5250 | 8200 | 14650 | 15750 | 21650 | 28200 |
| | динамическая | 6400 | 10050 | 16350 | 22700 | 25300 | 30800 |

Представленные в таблицах 1, 2 результаты подтверждают тот известный факт, что уменьшение диаметра графа соседства частиц ускоряет

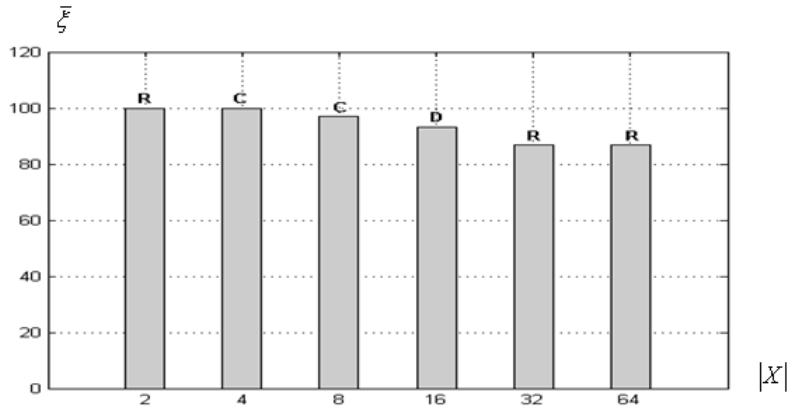
сходимость итерационного процесса, то есть повышает интенсивность поиска. С другой стороны, небольшой диаметр этого графа уменьшает широту поиска и повышает вероятность преждевременной сходимости итераций к локальному минимуму целевой функции. Таким образом, в каноническом алгоритме *PSO* топологию соседства «кольцо» целесообразно использовать в случае многомерных многоэкстремальных целевых функций. При оптимизации более простых целевых функций лучшим решением может быть использование топологии типа «клика».

### 3.2. Ко-алгоритм *Co-PSO*$^T$

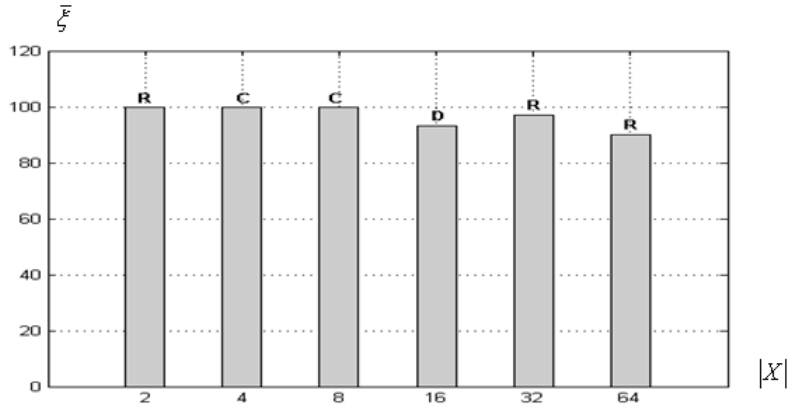
**Оценка вероятности локализации глобального экстремума.** Зависимость величины $\overline{\xi}$ от размерности вектора варьируемых параметров $|X|$ иллюстрирует рисунок 1, на котором одновременно показаны наиболее вероятные по мультистарту топологии суброев-победителей.




а) функция Розенброка




б) функция Химмельблау

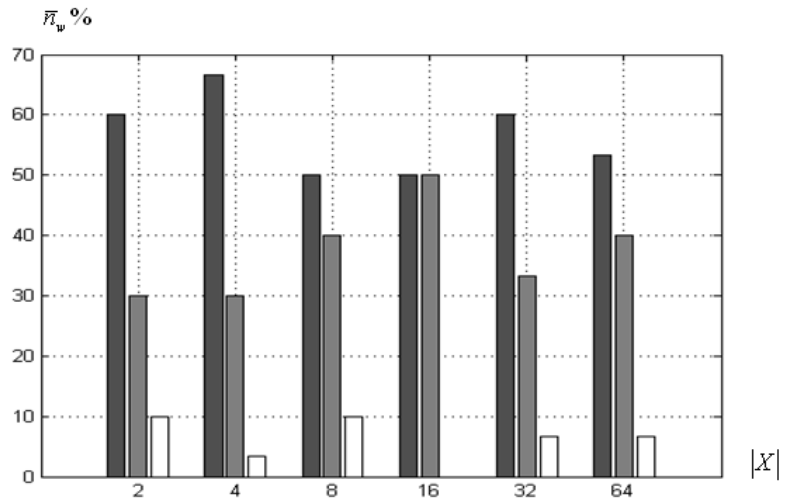



в) функция Растригина

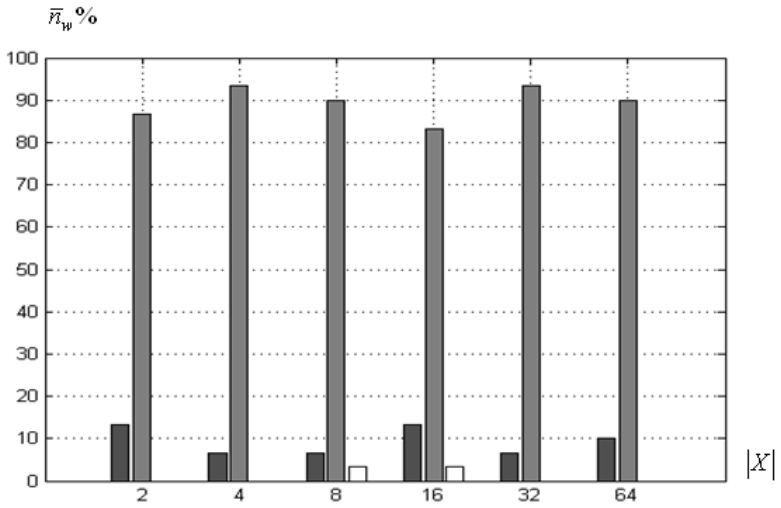
Рисунок 1 – Оценка вероятности $\bar{\xi}$ в функции величины $|X|$: алгоритм $Co-PSO^T$; $C$ – топология «клика», $R$ – «кольцо», $D$ –динамическая топология

Из рисунка 1а следует, что при низкой размерности вектора $X$ (равной 2, 4) для всех рассматриваемых тестовых функций алгоритм $Co-PSO^T$ обеспечивает 100% оценку вероятности локализации глобального минимума. В случае высокой размерности указанного вектора эта оценка не опускается ниже примерно 90% (при $|X|=64$). Важно, что величина $\bar{\xi}$ медленно уменьшается с ростом размерности вектора $X$. Сравнение рисунка 1 и таблицы 1 показывает, что во всех случаях алгоритм $Co-PSO^T$ обеспечивает лучшую оценку вероятности $\bar{\xi}$, чем канонический алгоритм *PSO*.

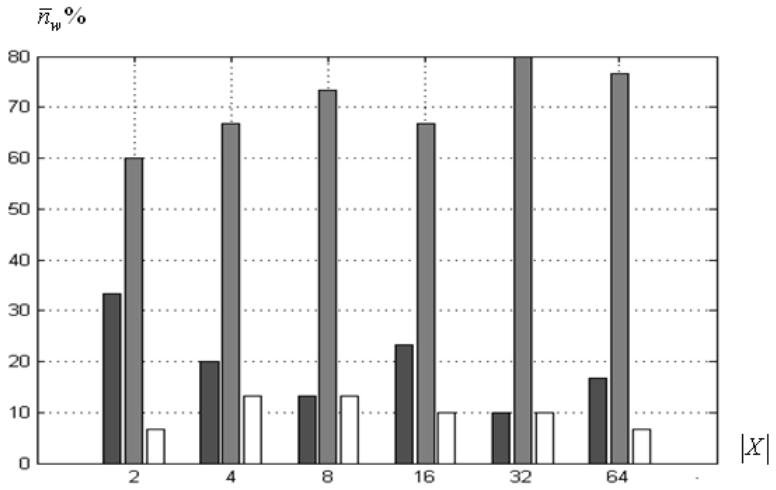
**Среднее число побед.** На рисунке 2 приведены диаграммы зависимости среднего по $Z$ стартам числа побед суброев $\overline{n}_w, \%$ от величины $|X|$ для каждой из рассматриваемых топологий соседства.




а) функция Розенброка




б) функция Химмельблау




в) функция Растригина

Рисунок 2 – Среднее число побед $\bar{n}_w$, % в функции величины $|X|$: ко-алгоритм $Co-PSO^T$; ■ - топология «клика», ■ - «кольцо», □ - динамическая

Как и ожидалось (п. 3.1), для функции Розенброка (рисунок 2а) наибольшее среднее число побед (примерно 60%) обеспечивает топология «клика». Для функций Химмельблау и Растригина (рисунки 2б, 2в) в среднем в 90% случаев побеждает алгоритм с топологией «кольцо».

**Среднее число итераций.** На рисунке 3 представлена диаграмма зависимости среднего числа итераций $\bar{t}_{end}$ от величины $|X|$

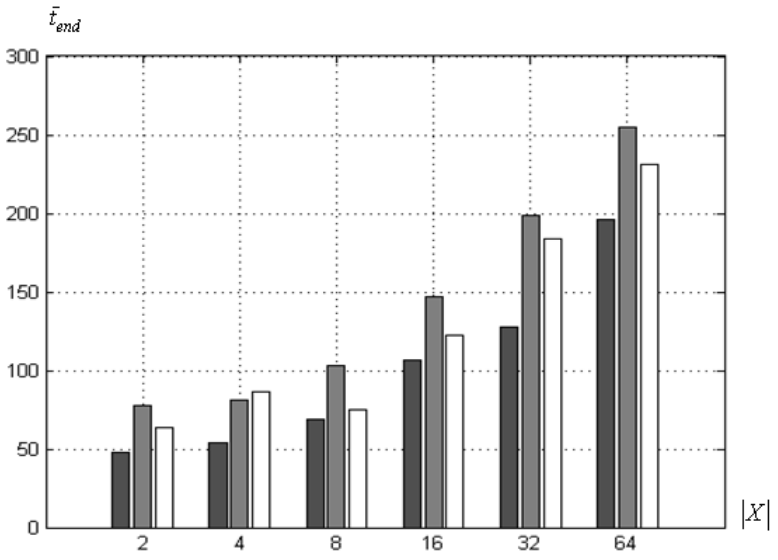



Рисунок 3 – Среднее число итераций $\bar{t}_{end}$ в зависимости от величины $|X|$: ко-алгоритм $Co-PSO^T$; ■ – функция Розенброка, ■ - Химмельблау, □ - Растригина

Рисунок 3 показывает, что для всех рассмотренных тестовых функций сходимость итерационного процесса к глобальному минимуму в худшем случае (при $|X|=64$) достигается в среднем за 225 итераций. Это значение почти в два раза меньше аналогичной величины для канонического алгоритма *PSO* (п. 3.1). Ожидаемо, при всех значениях $|X|$ быстрее всего вычислительный процесс достигает состояния стагнации для унимодальной функции Розенброка. Неожиданным является то обстоятельство, что то же состояние для многоэкстремальной функции Растригина достигается, в среднем, при меньшем числе итераций, чем для всего лишь четырехэкстремальной функции Химмельблау. Для объяснения этого феномена необходимы дополнительные исследования.

**Среднее число испытаний.** Диаграмма зависимости среднего числа испытаний $\bar{n}_f$ от величины $|X|$ для суброя-победителя представлена на рисунке 4.

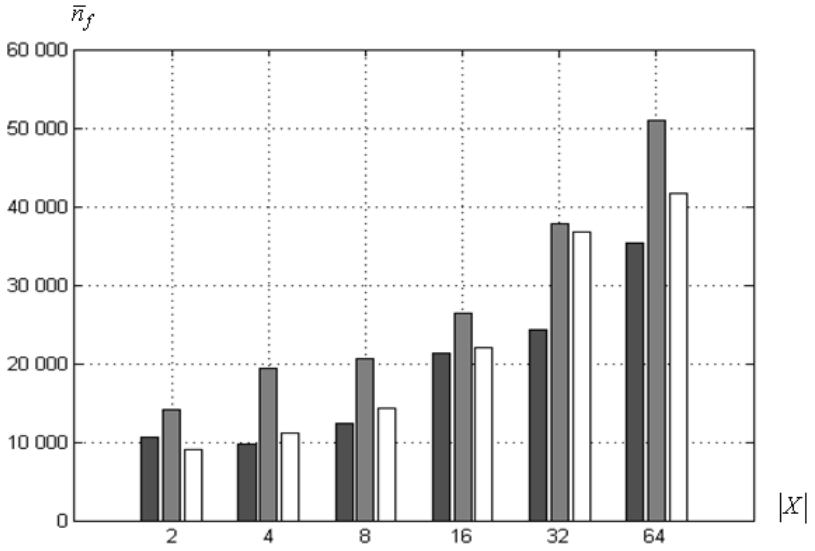
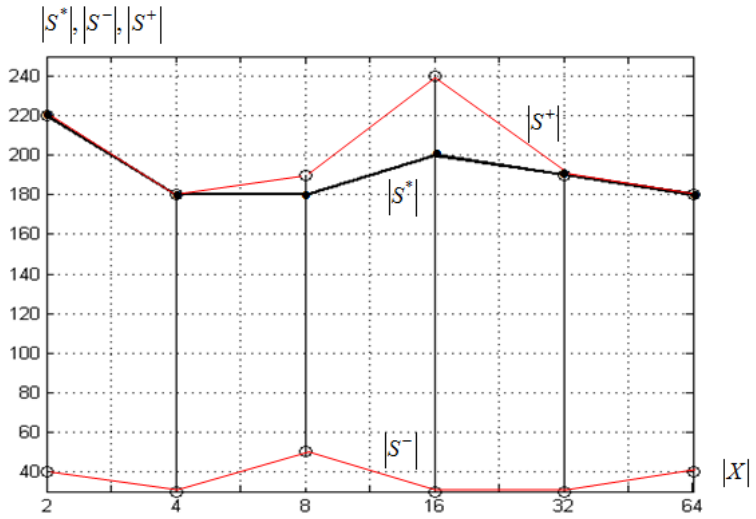


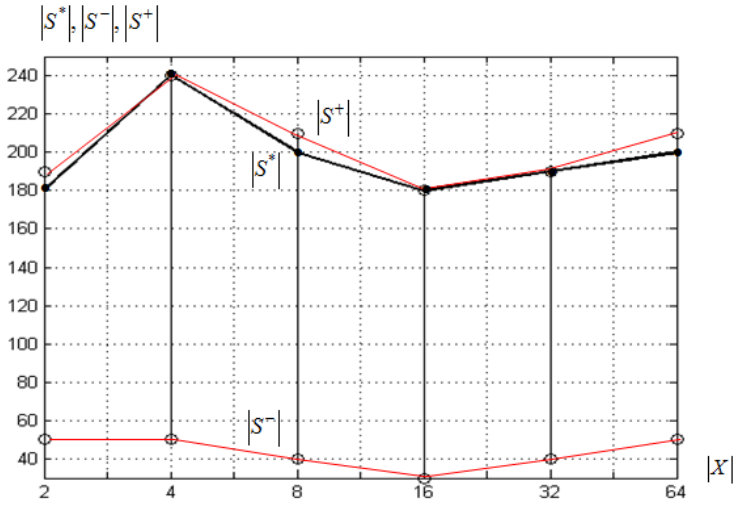

Рисунок 4 – Среднее число испытаний $\bar{n}_f$ для суброя-победителя в зависимости от величины $|X|$: ко-алгоритм $Co-PSO^T$; ■ – функция Розенброка, ■ - Химмельблау, □ - Растригина

Из рисунка 4 следует, что для суброя-победителя в худшем случае (при $|X|=64$) среднее число испытаний $\bar{n}_f$ не превышает 53 000. Эта величина почти в 1,5 раза выше аналогичного значения для канонического алгоритма *PSO* (таблица 2). Меньшее среднее число испытаний ко-алгоритм $Co-PSO^T$ обеспечивает, как правило, для функции Розенброка, большее число – для функции Химмельблау.

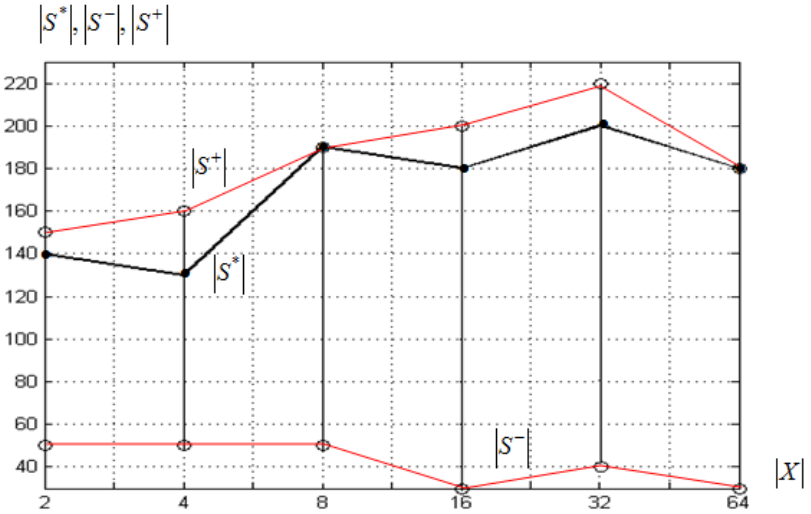
**Размеры суброев.** На рисунке 5 представлены зависимости размера суброя-победителя $|S^*|$ в момент окончания итерационного процесса от величины $|X|$, а также зависимости $|S^-|(|X|)$, $|S^+|(|X|)$.




а) функция Розенброка




б) функция Химмельблау




в) функция Растригина

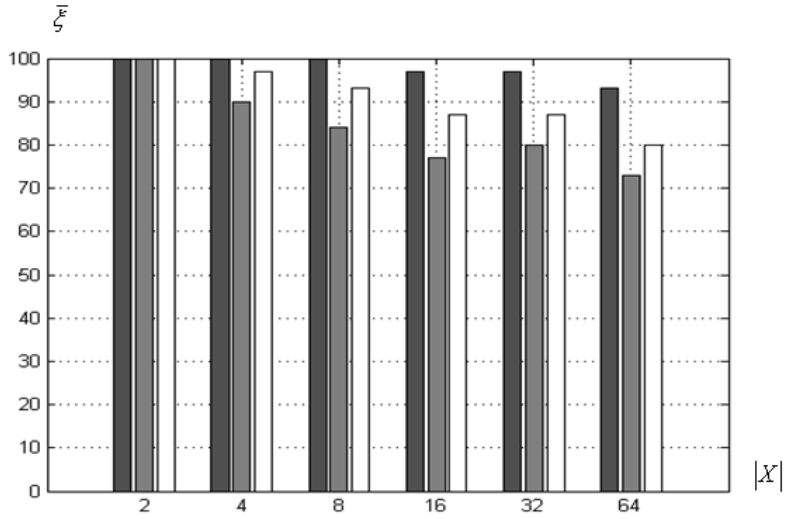
Рисунок 5 – Размеры суброев-победителей $|S^*|$ в момент окончания итерационного процесса в функции величины $|X|$: алгоритм $Co - PSO^T$

Рисунок показывает, что для всех исследуемых функций интервал изменения величины $|S^*|$ равен $[20:240]$. Для всех рассматриваемых тестовых функций и размерностей вектора $X$, суброй-победитель в своей эволюции проходит этап, когда число частиц в нем близко к минимально допустимому $|S|_{min}$. Достигнув своего максимального размера суброй-победитель, как правило, не уменьшается или уменьшается незначительно. Отметим еще то обстоятельство, что не всегда суброй-победитель в момент перехода итерационного процесса в состояние стагнации достигает своего максимального размера.

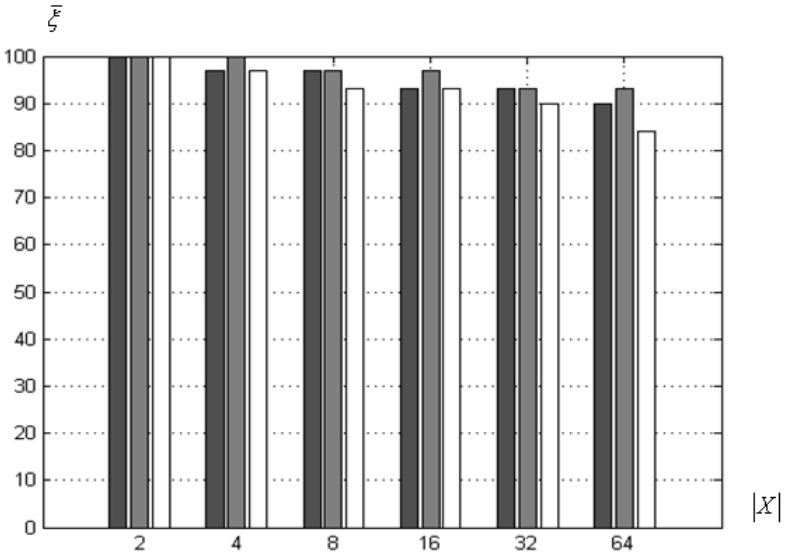
### 3.3. Ко-алгоритм *Co-PSO*$^P$

Исследование выполнено для трех вариантов ко-алгоритма $Co-PSO$: $Co-PSO^{PC}$, $Co-PSO^{PR}$, $Co-PSO^{PD}$. Все суброи в этих вариантах используют топологии «клика», «кольцо» и динамическую топологию соответственно.

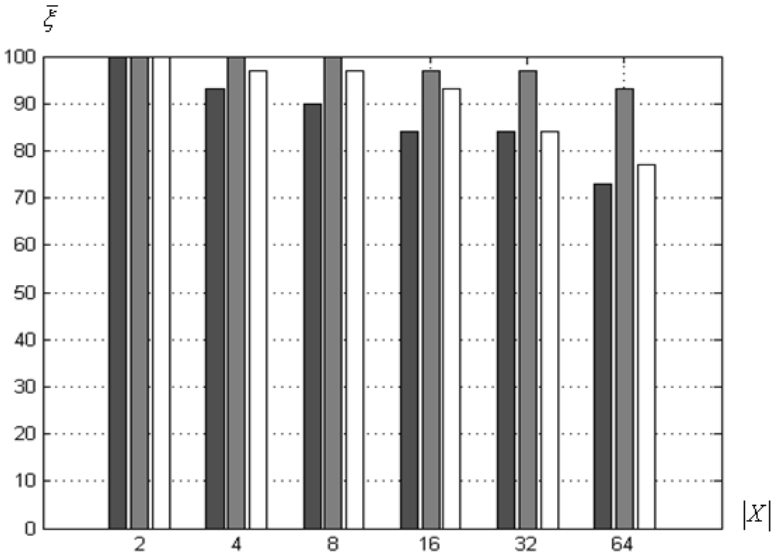
**Оценка вероятности локализации глобального экстремума.** Диаграммы зависимости оценки $\bar{\xi}$ от величины $|X|$ приведены на рисунке 6.




а) функция Розенброка




б) функция Химмельблау




в) функция Растригина

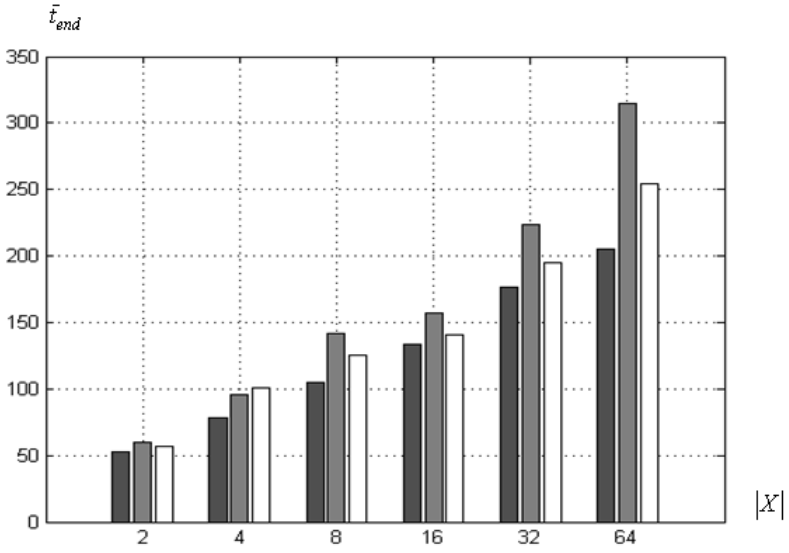
Рисунок 6 – Оценка вероятности $\bar{\xi}$ в функции величины $|X|$: ■ – ко-алгоритм $Co-PSO^{PC}$; ■ - $Co-PSO^{PR}$; □ - $Co-PSO^{PD}$

Рисунок 6 показывает следующее. Ожидаемо (п. 3.1), для функции Розенброка (рисунок 6а) при любых значениях $|X|$ максимальную оценку $\bar{\xi}$ обеспечивает ко-алгоритм $Co-PSO^{PC}$, а минимальную оценку – ко-алгоритм $Co-PSO^{PR}$. В случае функции Химмельблау (рисунок 6б) все три ко-алгоритма показывают близкие результаты. Для функции Растригина

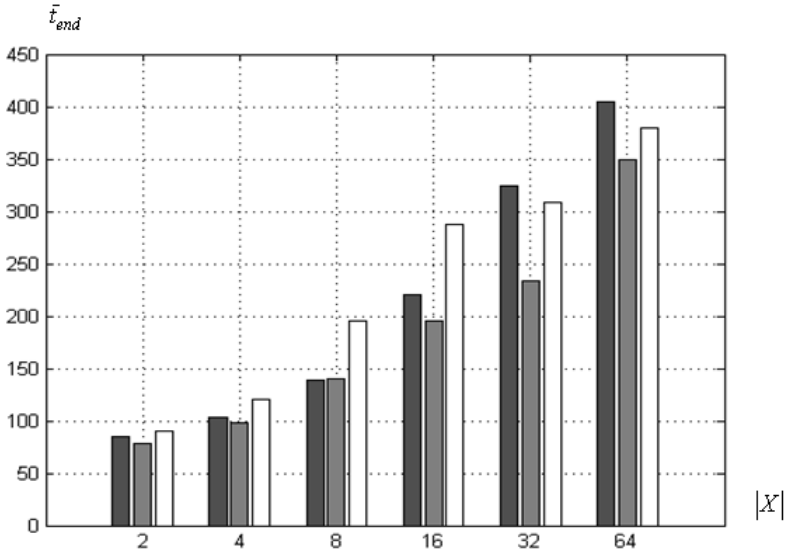
(рисунок 6в) во всех случаях лучшая оценка $\bar{\xi}$ получается с помощью ко-алгоритма $Co-PSO^{PR}$, а худшая – ко-алгоритма $Co-PSO^{PC}$. В худшем случае (при $|X|=64$) оценка $\bar{\xi}$ достигает величины, равной примерно 84%.

В целом, данное исследование показывает, что ко-алгоритм $Co-PSO^{P}$ обеспечивает более низкую оценку вероятности $\bar{\xi}$ в сравнении с ко-алгоритмом $Co-PSO^{T}$, но более высокую оценку, чем канонический алгоритм *PSO* (п. 3.1).

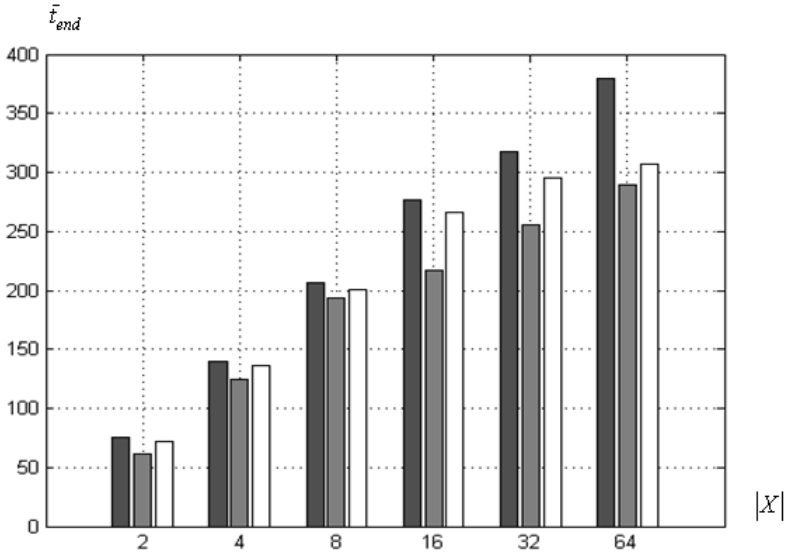
**Среднее число итераций.** На рисунке 7 представлены диаграммы зависимости среднего числа итераций $\bar{t}_{end}$ от величины $|X|$.




а) функция Розенброка




б) функция Химмельблау



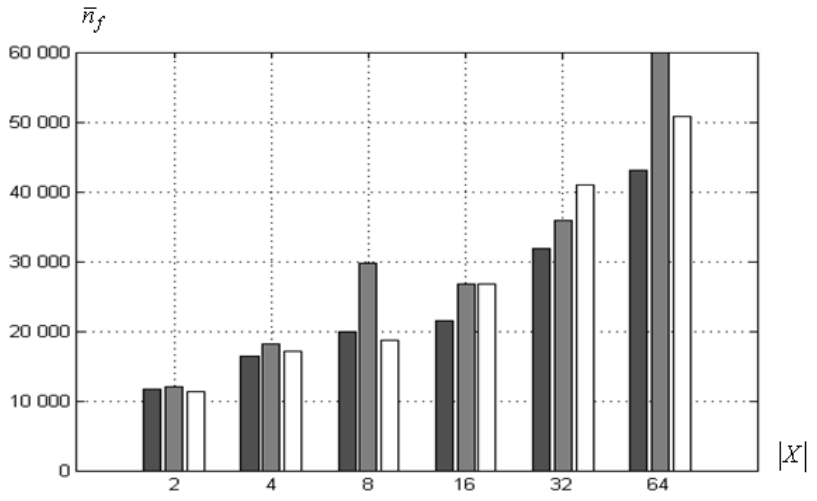

в) функция Растригина

Рисунок 7 – Среднее число итераций $\bar{t}_{end}$ в функции величины $|X|$: ■ – ко-алгоритм $Co-PSO^{PC}$, ■ - $Co-PSO^{PR}$, □ - $Co-PSO^{PD}$

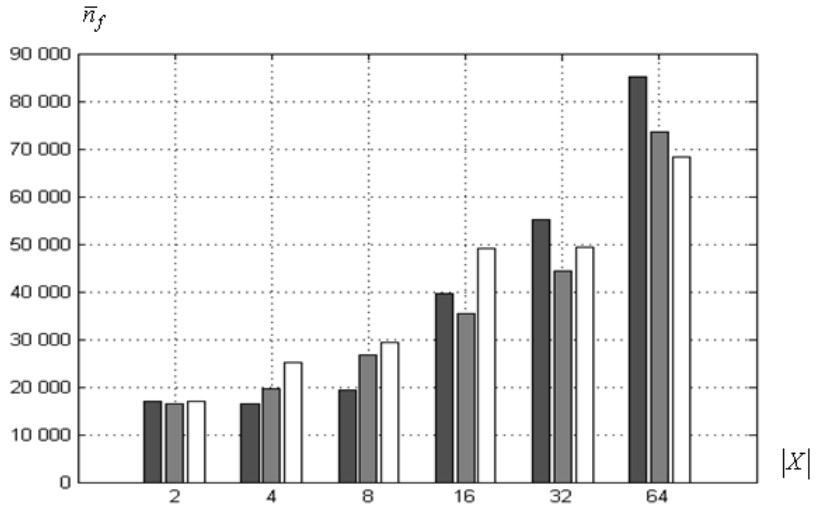
Рисунок 7 иллюстрирует следующие факты. Для функции Розенброка (рисунок 7а) наименьшее значение величины $\bar{t}_{end}$ зафиксировано у ко-алгоритма $Co-PSO^{PC}$, а наибольшее – у алгоритма $Co-PSO^{PR}$. В случае функции Химмельблау (рисунок 7б) минимальное значение величины $\bar{t}_{end}$, преимущественно, показывает ко-алгоритм $Co-PSO^{PR}$. Для функции

Растригина (рисунок 7в) лучшие результаты с точки зрения величины $\bar{t}_{end}$ также обеспечивает ко-алгоритм $Co-PSO^{PR}$. Рисунок 7 показывает, кроме того, что, в худшем случае (при $|X|=64$), для всех трех рассматриваемых тестовых функций величина $\bar{t}_{end}$ составляет примерно 324 итерации. Этот результат хуже, чем зафиксированный для алгоритма $Co-PSO^{T}$, но лучше результата, который показал канонический алгоритм *PSO* (п. 3.1).

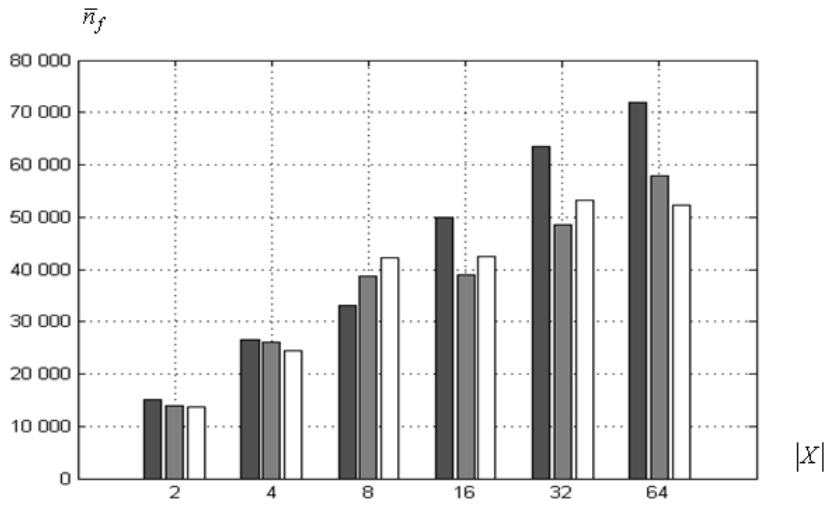
**Среднее число испытаний.** Зависимость среднего числа испытаний $\bar{n}_f$ от величины $|X|$ иллюстрирует рисунок 8.




а) функция Розенброка




б) функция Химмельблау



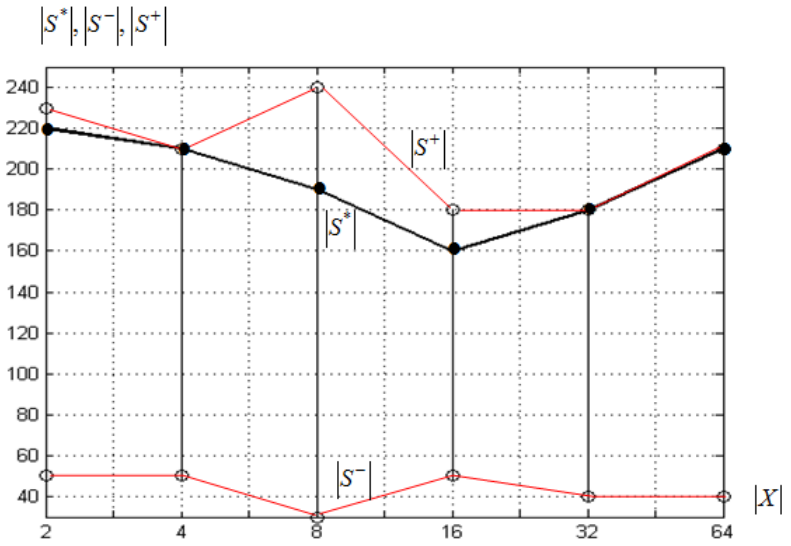

в) функция Растригина

Рисунок 8 – Среднее число испытаний $\bar{n}_f$ в функции величины $|X|$: ■ – ко-алгоритм $Co-PSO^{PC}$, ■ - $Co-PSO^{PR}$, □ - $Co-PSO^{PD}$

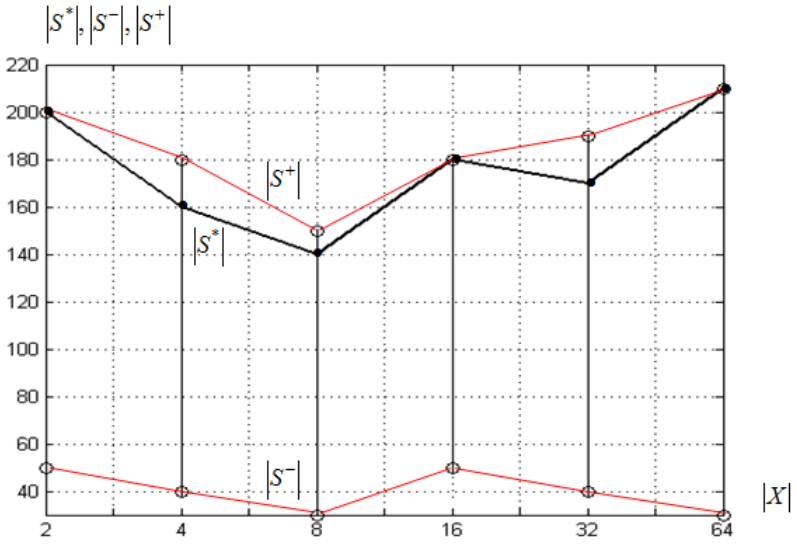
Из рисунка 8а следует, что для функции Розенброка минимальное значение величины $\bar{n}_f$ в большинстве случаев соответствует ко-алгоритму $Co-PSO^{PC}$, а максимальное значение - ко-алгоритму $Co-PSO^{PR}$. Для функции Химмельблау явного лидера по среднему числу испытаний $\bar{n}_f$ рисунок 8б не выявляет. В случае функции Растригина (рисунок 8в) минимальная величина $\bar{n}_f$ соответствует ко-алгоритмам $Co-PSO^{PR}$, $Co-PSO^{PD}$. При $|X|=64$ (худший случай) величина $\bar{n}_f$ для суброя-победителя для всех рассматриваемых тестовых функциях примерно равна 64 000. Эта величина почти в два раза выше аналогичного значения для канонического алгоритма $PSO$ (п. 3.1) и примерно на 30% выше того же значения для алгоритма $Co-PSO^{T}$

**Размеры суброев.** На рисунке 9 представлены графики зависимости конечных размеров суброев-победителей $|S^*|$ от величины $|X|$, а также аналогичные графики зависимостей $|S^-(|X|)$, $|S^+(|X|)$. Из всех результатов

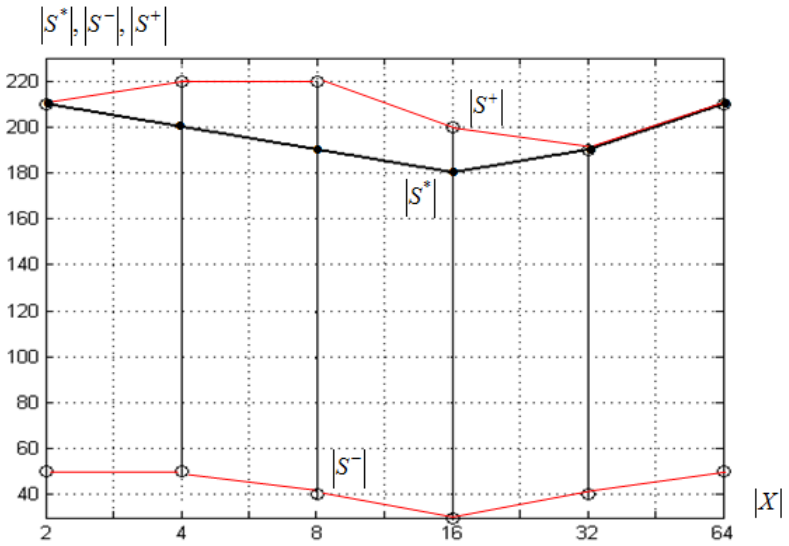
исследования представляем четыре: ко-алгоритм $Co-PSO^{PC}$ и функция Розенброка; ко-алгоритмы $Co-PSO^{PC}$, $Co-PSO^{PR}$ и функция Химмельблау; ко-алгоритм $Co-PSO^{PR}$ и функция Растригина.



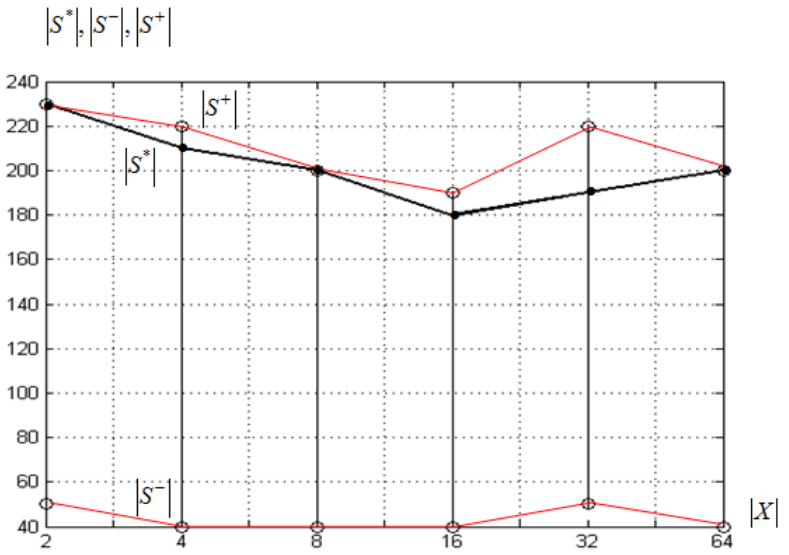

а) ко-алгоритм $Co-PSO^{PC}$, функция Розенброка




б) ко-алгоритм $Co-PSO^{PC}$, функция Химмельблау




в) ко-алгоритм $Co-PSO^{PR}$, функция Химмельблау

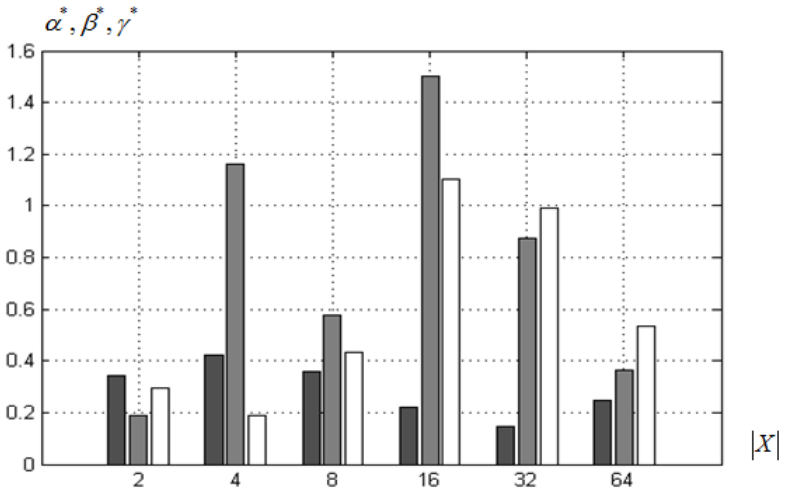



г) ко-алгоритм $Co-PSO^{PR}$, функция Растригина

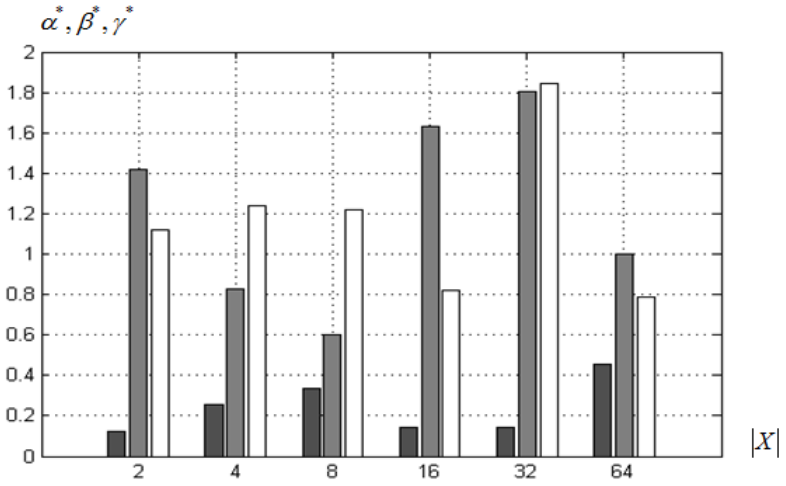
Рисунок 9 – Предельные размеры суброев $|S^*|$ в функции величины $|X|$

Из рисунка 9 следует, что для суброя-победителя минимальная величина $|S^*|$ равна 20, а максимальная - 240. Характер поведения предельных размеров суброев-победителей близок к аналогичным результатам, полученным для алгоритма $Co-PSO^T$ (см. выше).

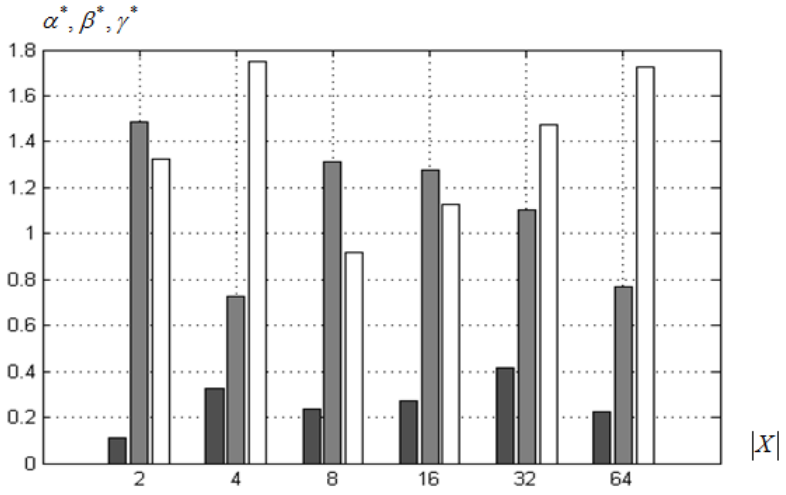
**Величины свободных параметров $\alpha^*, \beta^*, \gamma^*$ суброев-победителей.** На рисунке 10 представлены диаграммы значений свободных параметров $\alpha^*, \beta^*, \gamma^*$ суброев-победителей в функции величины $|X|$.



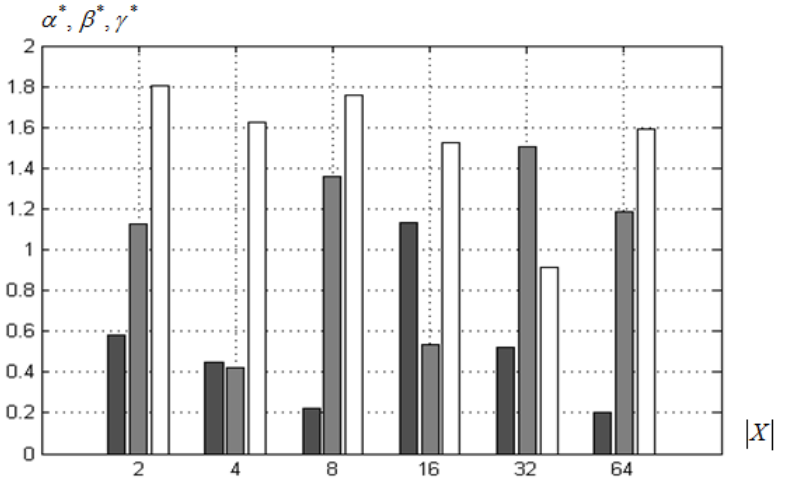

а) ко-алгоритм $Co-PSO^{PC}$, функция Розенброка




б) ко-алгоритм $Co-PSO^{PC}$, функция Химмельблау




в) ко-алгоритм $Co - PSO^{PR}$, функция Химмельблау




г) ко-алгоритм $Co - PSO^{PR}$, функция Растригина

Рисунок 10 – Значения свободных параметров $\alpha^*$, $\beta^*$, $\gamma^*$ суброев-победителей в функции величины $|X|$: ■ – параметр $\alpha^*$, ■ - $\beta^*$, □ - $\gamma^*$

Рисунок 10 показывает, что во всех экспериментах полученные значения свободных параметров значительно отличаются от рекомендованных (п. 3.1). Так, оказалось, что значение инерционного параметра $\alpha^*$ во всех случаях не превышает значения 0,6, в то время как рекомендованное значение равно 0,7298. Значения социальной и когнитивной компонент изменяются в широком диапазоне [0,2; 1,8].

Таким образом, вычислительный эксперимент показал, что по всем рассмотренным индикаторам эффективности ко-алгоритмы $Co-PSO^T$ и $Co-PSO^P$ превосходят канонический алгоритм $PSO$. Эффективность ко-алгоритма $Co-PSO^T$ в большинстве случаев оказалась выше эффективности ко-алгоритма $Co-PSO^P$.

## 4. Задача об управляемом спуске космического аппарата

Рассматриваем задачу оптимального управления спуском космического аппарата в атмосфере Земли [9]. Используем уточненную постановку задачи из работы [3].

### 4.1. Постановка задачи

Начало системы координат $0x_1x_2x_3$, в которой рассматриваем движение аппарата, находится в центре Земли. Ось $0x_1$ направлена на исходное положение аппарата, ось $0x_3$ - в сторону его движения и перпендикулярна оси $0x_1$, ось $0x_2$ образует с указанными осями правую тройку.

Математическую модель объекта управления представляет система обыкновенных дифференциальных уравнений (ОДУ)

$$\begin{aligned}
&\frac{dx_1}{dt} = x_2,\\
&\frac{dx_2}{dt} = -\frac{\gamma}{r^3}\,x_1 - (1-u)\,q\,x_2,\\
&\frac{dx_3}{dt} = x_4,\\
&\frac{dx_4}{dt} = -\frac{\gamma}{r^3}\,x_3 - (1-u)\,q\,x_4
\end{aligned} \qquad (11)$$

при заданных начальных условиях $x_i(0) = x_i^0,\ i \in [1:4]$. Приняты следующие обозначения: $x_1, x_3$ - координаты центра масс космического аппарата; $x_2, x_4$ - компоненты его скорости; $u$ - управление; $r = \sqrt{x_1^2 + x_3^2}$ - величина радиус-

вектора аппарата; $\gamma$ – гравитационная постоянная Земли; $q = c\, e^{\frac{h-x_1}{\eta}}\, v$ - составляющая аэродинамической силы; $v = \sqrt{x_2^2 + x_4^2}$ - величина скорости аппарата; $c$ – его аэродинамическая характеристика; $h$ – высота атмосферы; $\eta$ – согласующий коэффициент модели атмосферы [3].

Определены функционалы качества управления

$$f_1(u) = \max_{\tau \in [0;\hat{\tau}]} \frac{(x_1^2(\tau) + x_3^2(\tau))\sqrt{x_2^2(\tau) + x_4^2(\tau)}}{g_0\ r_з^2} \to \min_{u \in D_u}, \tag{12}$$

$$f_2(u) = x_3(\hat{\tau}) - l \to \min_{u \in D_u}, \tag{13}$$

$$f_3(u) = \max Q \to \min_{u \in D_u}, \tag{14}$$

имеющие смысл максимальной перегрузки, отклонения от заданной точки на поверхности Земли и максимальной амплитуды перегрузки соответственно. Здесь $r_з = 6371$ км – радиус Земли, $g_0 = 9{,}81$ м/с$^2$ – ускорение свободного падения, $\hat{\tau}$ – длительность полета, $l$ – координата по оси $0x_3$ заданной точки на поверхности Земли, $D_u = \{u(\tau) \,|\, |u(\tau)| \in [u^-; u^+]\}$ - множество допустимых управлений, $u^-, u^+$ - заданные константы.

Задача представляет собой трехкритериальную задачу оптимального управления динамической системой и состоит в определении допустимого управления $u^*(\tau) \in D_u$, доставляющего минимум критериям оптимальности (12) – (14).

Не смотря на известный недостаток решения задач многокритериальной оптимизации с помощью скалярной свертки их частных критериев оптимальности [10], применяем именно этот метод. Используем аддитивную скалярную свертку

$$f(u) = \sum_{i=1}^{3} \lambda_i f_i(u), \tag{15}$$

где $\lambda_i > 0$ – весовой коэффициент, учитывающий относительную важность функционала $f_i(u)$. В результате получаем однокритериальную задачу оптимального управления динамической системой (11) с критерием оптимальности (15).

Аналогично тому, как это сделано в работе [3], сводим полученную задачу оптимального управления к задаче нелинейного программирования.

Покрываем интервал $[0;\hat{\tau}]$ равномерной сеткой с узлами $\tau_i$, $i \in [0:|U|]$ и отыскиваем оптимальное управление $u^*(\tau)$ в классе кусочно-линейных функций (рисунок 11).

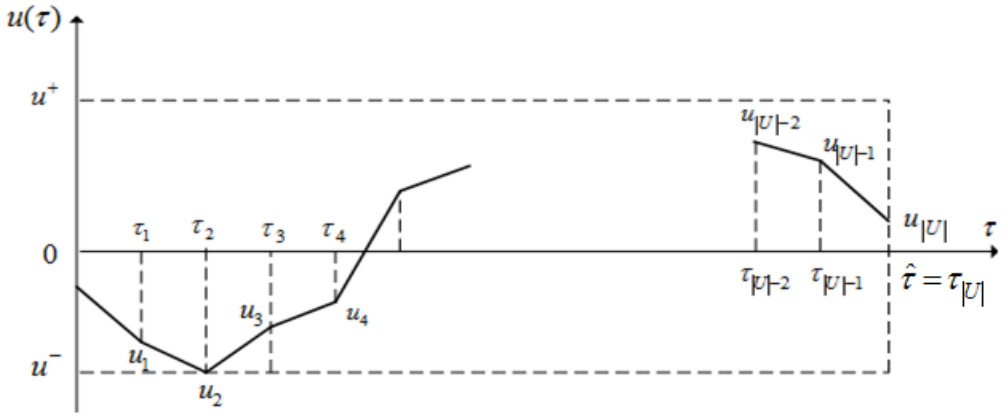



Рисунок 11 – Кусочно-линейная аппроксимация управления $u(\tau)$

Обозначаем $U = (u_1, u_2, ..., u_{|U|})$, где $u_i = u(\tau_i)$. Тогда указанная выше однокритериальная задача оптимального управления принимает вид

$$\min_{U \in D_U} f(U) = f(U^*), \qquad (16)$$

где $D_U = \{U \mid |u_i| \in [u^-; u^+],\ i \in [0:|U|]\}$. Таким образом, имеем задачу глобальной условной оптимизации (16) с целевой функцией (15), вектором варьируемых параметров $U$ и множеством его допустимых значений $D_U$.

### 4.2. Вычислительный эксперимент

Эксперимент выполнен при следующих начальных условиях космического аппарата: $x_1(0) = 6471$ км, $x_3(0) = 0$, $x_2(0) = -0{,}5$ км/с, $x_4(0) = 6{,}58$ км/с. Поскольку снижение значения максимальной перегрузки является приоритетным, задаем следующие значения весовых коэффициентов $\lambda_i$:

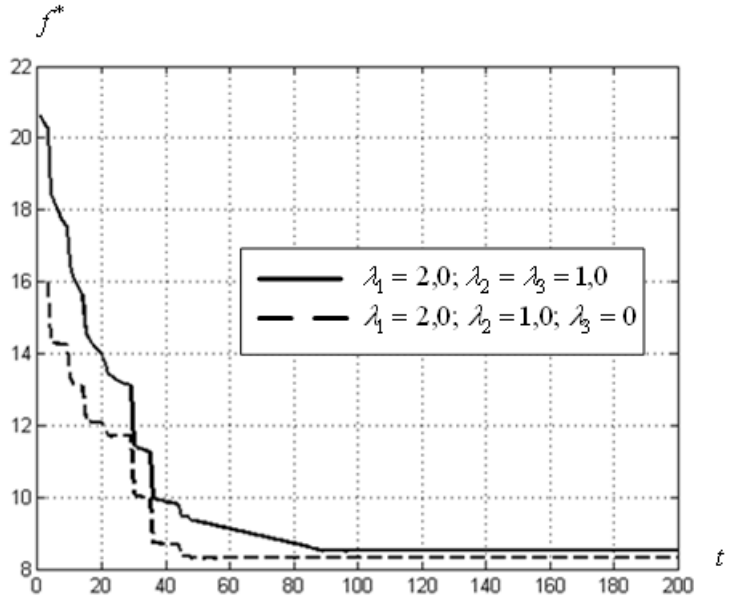
$$\lambda_1 = 2{,}0; \quad \lambda_2 = \lambda_3 = 1{,}0.$$

Размерность вектора $|U|$ принимаем равной 50.

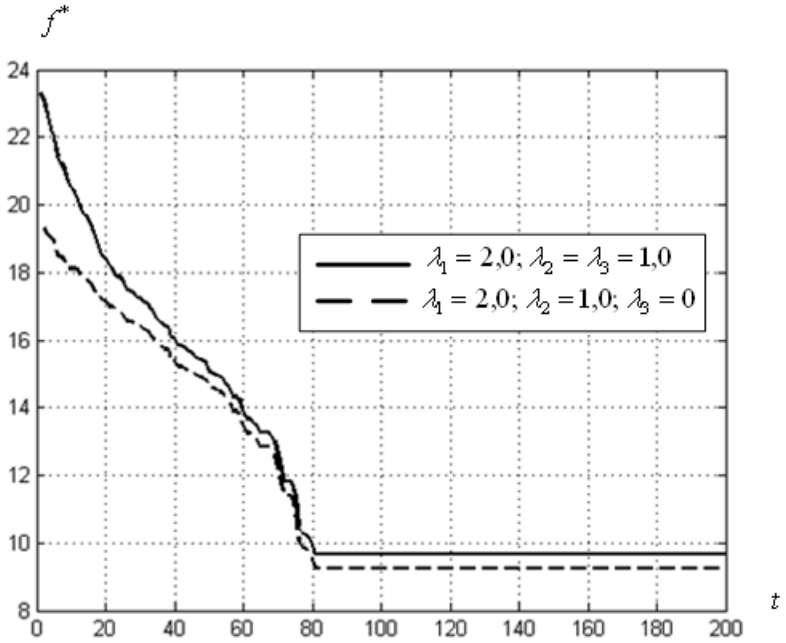
Специальное исследование показало, что система ОДУ (11) является нежесткой. Поэтому для интегрирования этой системы используем метод Рунге-Кутты 4-го порядка.

Задачу решаем с использованием ко-алгоритма $Co-PSO^{PC}$ с пятью ко-эволюционирующими суброями, каждый из которых имеют топологию соседства частиц типа «клика» и отличается от других суброев значениями своих параметров $\alpha, \beta, \gamma$. Приняты следующие значения свободных параметров ко-алгоритма: интервал адаптации $\Delta_a = 9$, величина штрафа $n_p = 0{,}2$, минимально допустимый размер суброя $|S|_{\min} = 4$. Размер роя равен $|S| = 100$. В процессе инициализации роя все субпопуляции получают число частиц, равное 20. Поскольку размерность вектора варьируемых параметров велика (равна 51), а априорная информация о ландшафте целевой функции $f(U)$ отсутствует, для повышения вероятности локализации глобального экстремума используем мультистарт с числом запусков алгоритма $Z = 30$. Число итераций ко-алгоритма ограничиваем величиной $\hat{t} = 1000$.

Рисунок 12 иллюстрирует сходимость вычислительного процесса при решении задачи с помощью ко-алгоритма $Co-PSO^{PC}$ и канонического алгоритма *PSO* соответственно. Заметим, что не учитывает максимальную амплитуду перегрузки $Q$ [3].




а) ко-алгоритм $Co-PSO^{PC}$ : суброй-победитель




б) канонический алгоритм *PSO*

Рисунок 12 – Сходимость вычислительного процесса для лучшего по мультистарту запуска алгоритма и суброя-победителя

Рисунок показывает значительное превосходство в скорости сходимости ко-алгоритма $Co-PSO^{PC}$ над каноническим алгоритмом *PSO*. Лучшее, достигнутое ко-алгоритмом $Co-PSO^{PC}$ значение целевой

функцией, равно $f^* = 8{,}85$, а аналогичное значение для канонического алгоритма $PSO$ - $f^* = 9{,}66$. Таким образом, из рисунка следует, что ко-алгоритм $Co - PSO^{PC}$ по сравнению с каноническим алгоритмом $PSO$ обеспечивает не только более высокую скорость сходимости, но и более высокую точность локализации глобального экстремума целевой функции.

Представляет интерес рисунок 13, иллюстрирующий характер сходимости итерационного процесса для каждого из суброев ко-алгоритма $Co - PSO^{PC}$ (вариант $\lambda_1 = 2{,}0;\ \lambda_2 = 1{,}0;\ \lambda_3 = 0$). Победителем в данном эксперименте стал суброй $S_4$, который обеспечил значения критериев оптимальности, равные $f_1^* \approx 7{,}61,\ f_2^* \approx 0{,}68$.

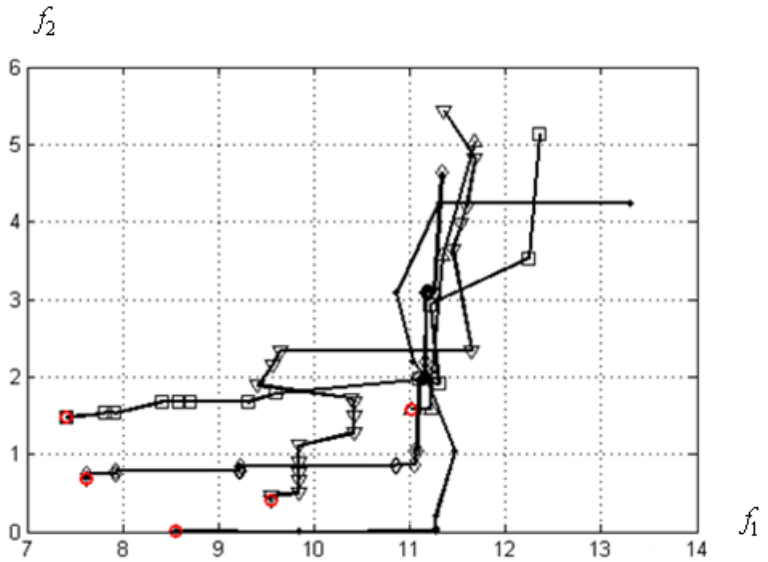



Рисунок 13 – Сходимость лучшего по мультистарту запуска алгоритма:
вариант $\lambda_1 = 2{,}0;\ \lambda_2 = 1{,}0;\ \lambda_3 = 0$; ко-алгоритм $Co - PSO^{PC}$;
-, Δ, □, ◊, ∇ - суброи $S_1 - S_5$ соответственно; красные маркеры
соответствуют концу итерационного процесса

На рисунке 14 представлены зависимости численности $|S_i|$ суброев ко-алгоритма в функции номера итерации $t$; $i \in [1:5]$. Зависимость победившего суброя (имеющего номер 4) выделена красным цветом.

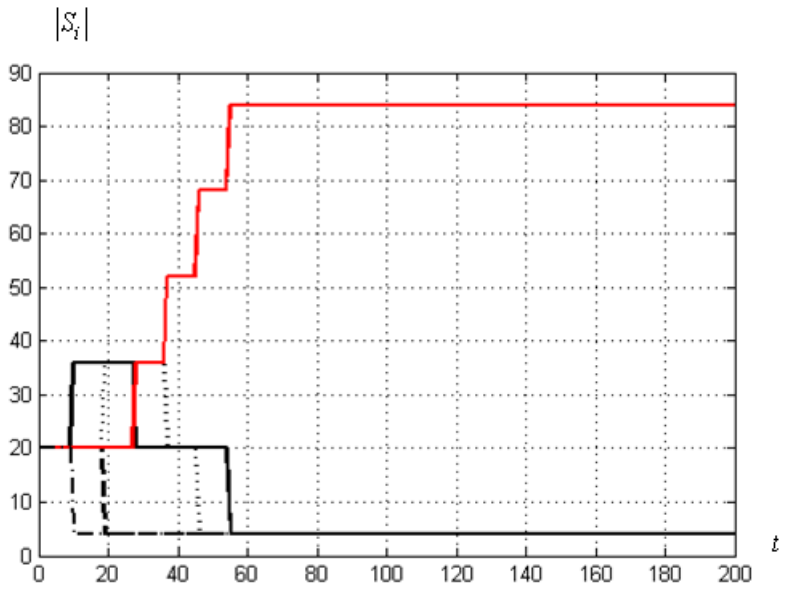



Рисунок 14 – Численности $|S_i|$ суброев в функции номера итерации для лучшего по мультистарту запуска алгоритма: ко-алгоритм $Co-PSO^{PC}$; ——, – – –, ·····, ——, –·–·- суброи $S_1$ – $S_5$ соответственно

Рисунок 14 показывает, что после первого интервала адаптации лидером становится суброй $S_1$. Суброй-победитель $S_4$ начинает лидировать после третьего интервала адаптации и сохраняет свое лидерство до завершения итерационного процесса, когда число частиц в нем становится равным 84.

Диаграмма значений свободных параметров $\alpha, \beta, \gamma$ суброев $S_1 - S_2$ приведена на рисунке 15. Примечательно, что значения этих параметров суброя-победителя наиболее близки к их рекомендованным значениям (п. 1) и составляют 0,9846, 1,13299, 1,6412 соответственно.

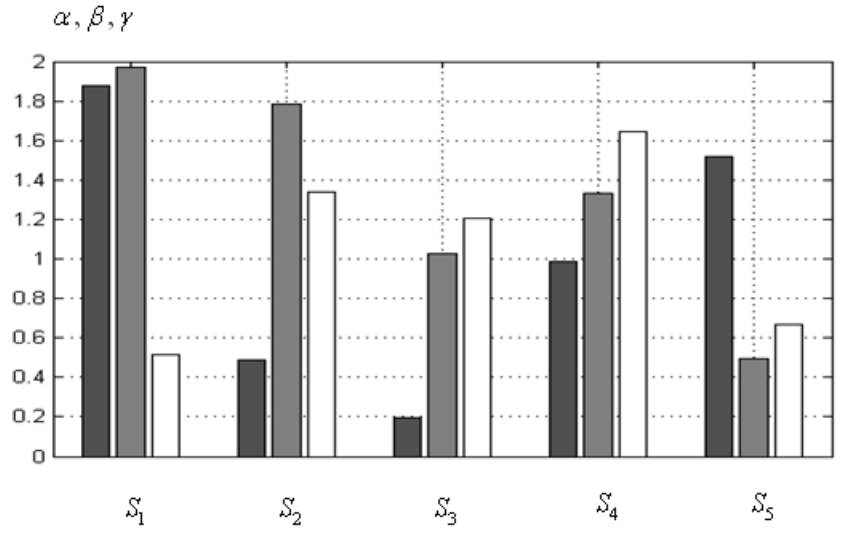



Рисунок 15 – Значения свободных параметров суброев ко-алгоритма $Co - PSO^{PC}$ для лучшего по мультистарту запуска: ■ – параметр $\alpha$, ■ - $\beta$, □ - $\gamma$

Полученная в вычислительном эксперименте траектория спуска космического аппарата представлена на рисунке 16. Минимальное отклонение от заданного места посадки, достигнутое алгоритмом $Co - PSO^{PC}$, составляет 0,68 км, а каноническим алгоритмом $PSO$ – 0,92 км.

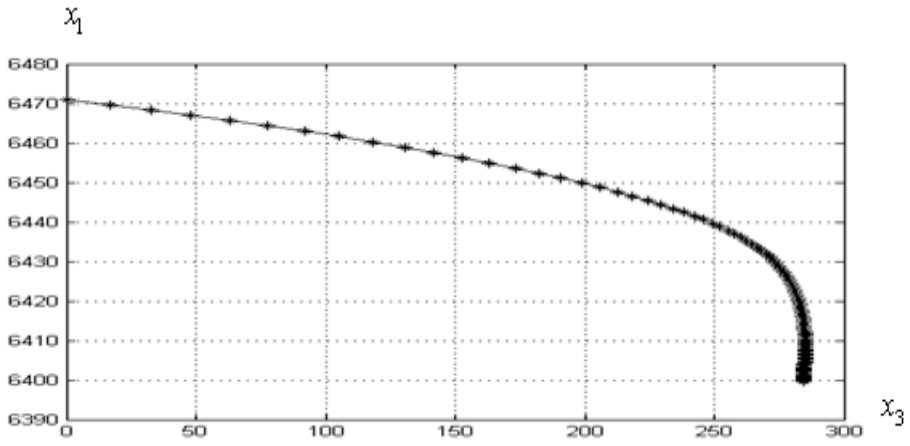
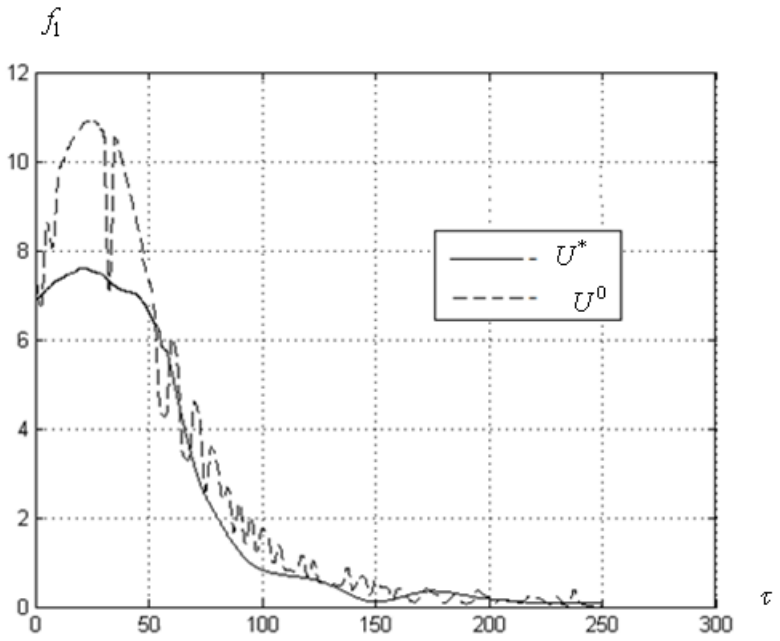
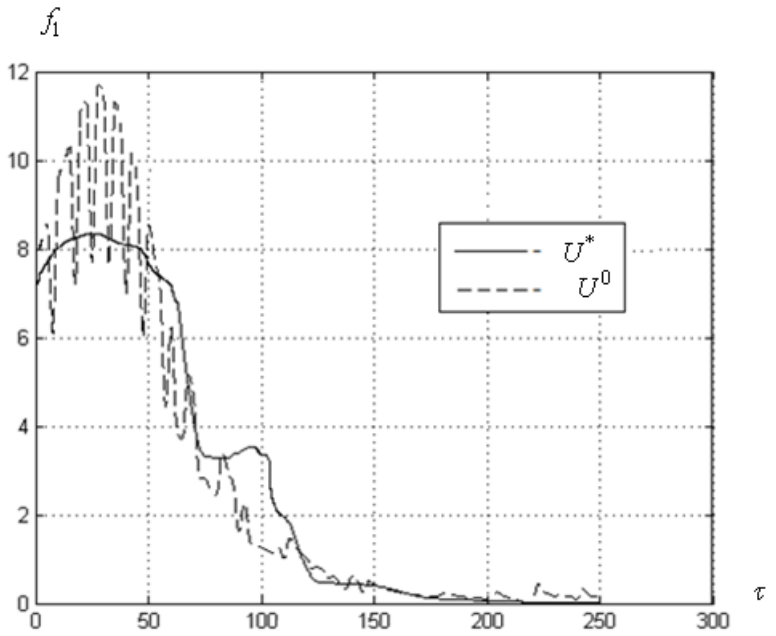



Рисунок 16 – Траектория спуска космического аппарата для лучшего по мультистарту запуска алгоритма; ко-алгоритм $Co - PSO^{PC}$

Показательным является рисунок 17, на котором приведены максимальные перегрузки $f_1$ в функции времени спуска $\tau$ космического аппарата, полученные с помощью ко-алгоритма $Co-PSO^{PC}$ и канонического алгоритма *PSO*. Пунктирные линии на графиках соответствуют начальному приближению управления $U^0$, сплошные линии – найденному оптимальному управлению $U^*$ (в момент $t_{end}$ окончания итерационного процесса). Перегрузка обусловлена действием аэродинамической силы, пропорциональной квадрату скорости космического аппарата. После 250-ой секунды спуска эта скорость становится относительно малой и колебания перегрузки несущественны. Поэтому графики ограничены этим временем.




а) ко-алгоритм $Co-PSO^{PC}$

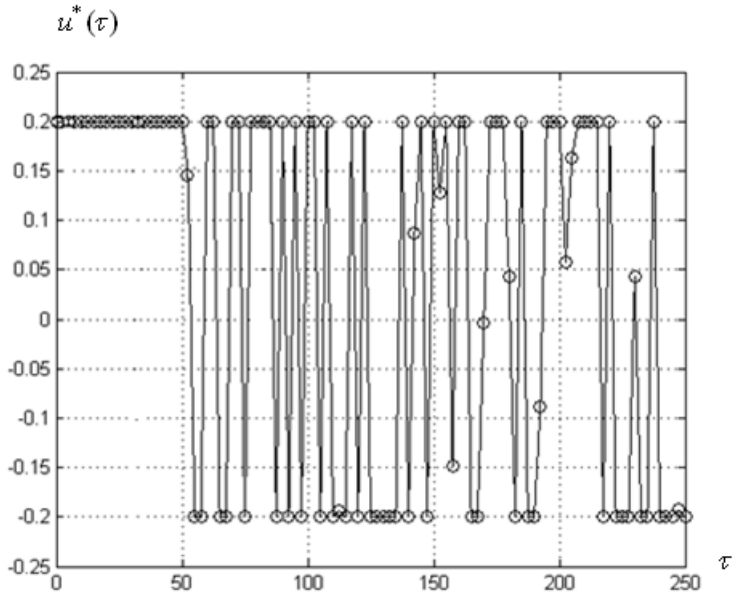



б) канонический алгоритм *PSO*

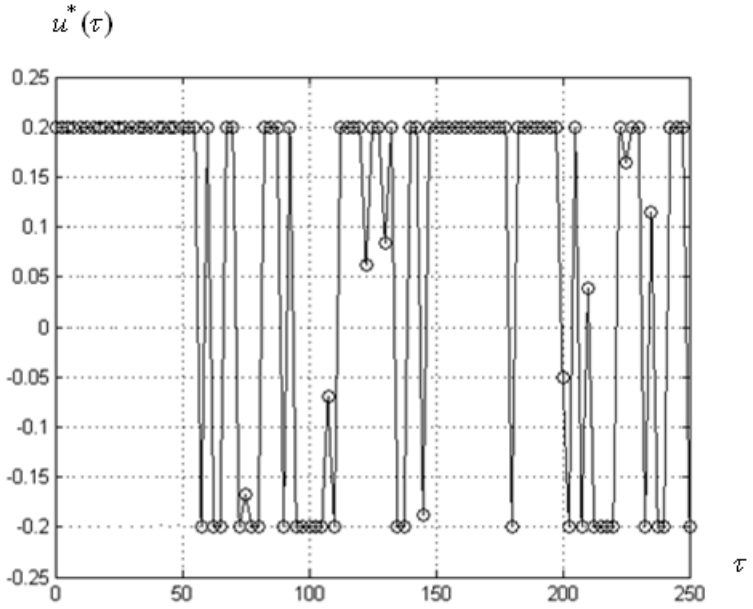
Рисунок 17 – Зависимость перегрузки $f_1$ от времени спуска космического аппарата для лучшего по мультистарту запуска алгоритма

Рисунок 18, показывает, что ко-алгоритм $Co - PSO^{PC}$ обеспечивает по сравнению с каноническим алгоритмом *PSO* значительно более гладкую зависимость перегрузки от времени снижения космического аппарата (что подтверждают соответствующие значения критерия оптимальности $f_3$).

Приведем, наконец, оптимальные управления, найденные ко-алгоритмом $Co - PSO^{PC}$ и каноническим алгоритмом *PSO*, которым соответствуют представленные выше результаты (рисунок 18).




а) ко-алгоритм $Co - PSO^{PC}$




б) канонический алгоритм *PSO*

Рисунок 18 – Варианты оптимального управления $u^*(\tau)$ для лучшего по мультистарту запуска алгоритма

Известно, что характерной особенностью рассматриваемой динамической системы является оптимальность в ней, так называемых, скользящих режимов, когда, в пределе, управление переключается между своими минимально и максимально допустимыми значениями с бесконечной

частотой. Заметим, что именно это обстоятельство делает рассматриваемую задачу сложной с вычислительной точки зрения. Рисунок 20 показывает, что ко-алгоритм $Co-PSO^{PC}$ обеспечивает гораздо более хорошее приближение к указанному режиму, чем канонический алгоритм *PSO*.

## Заключение

В работе предложен ко-эволюционный алгоритм глобальной оптимизации *Co-PSO*, основанный на алгоритме роя частиц. Ко-алгоритм предполагает параллельное (по крайней мере, на логическом уровне) функционирование заданного числа субалгоритмов *PSO*, которые используют различные топологии соседства частиц и/или различные значения своих свободных параметров.

Представлен *MatLab*-комплекс программ, реализующих алгоритм *Co-PSO* и канонический алгоритм роя частиц *PSO*.

Выполнен широкий вычислительный эксперимент по исследованию эффективности алгоритма *Co-PSO* и реализующего его программного обеспечения. В эксперименте использованы тестовые функции Розенброка, Химмельблау и Растригина. Результаты исследования показывают превосходство алгоритма *Co-PSO* над каноническим алгоритмом *PSO* по ряду критериев эффективности.

С помощью разработанного алгоритмического и программного обеспечения решена трехкритериальная задача оптимального управления космическим аппаратом на этапе его спуска в атмосфере Земли. Результаты решения показали высокую эффективность предложенных алгоритмических и программных решений.

В развитие работы авторы планируют параллельную реализацию и исследование параллельной версии ко-алгоритма *Co-PSO*.

## Список литературы

1. Воробьева Е.Ю., Карпенко А.П., Селиверстов Е.Ю. Ко-гибридизация алгоритмов роя частиц // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журнал. 2012. № 4. Режим доступа: http://www.technomag.edu.ru/doc/355729.html (дата обращения 27.08.2013).

2. Karpenko A.P., Svianadze Z.O. Meta-optimization based on self-organizing map and genetic algorithm // Optical Memory and Neural Networks (Information Optics). 2011. Vol. 20, no. 4. P. 279-283. http://dx.doi.org/10.3103/S1060992X11040059

3. Карпенко А.П., Митина Е.В., Семенихин А.С. Когенетический алгоритм Парето-аппроксимации в задаче многокритериальной оптимизации // Информационные технологии. 2013. № 1. С. 22-32.

4. Карпенко А.П., Селиверстов Е.Ю. Глобальная оптимизация методом роя частиц. Обзор // Информационные технологии. 2010. № 2. С. 25-34.

5. Yongquan Zhou, Shengyu Pei. A hybrid co-evolutionary particle swarm optimization algorithm for solving constrained engineering design problems // China: Journal of computers. 2010. Vol. 5, no. 6. P. 965-972.

6. Yuhui Shi, Renato A. Krohling. Co-evolutionary particle swarm optimization to solve min-max problems // USA: Proceedings of the Evolutionary Computation. 2002. Vol. 2. P. 1682-1687.

7. Yang Sun, Lingbo Zhang, Xingsheng Gu. Co-evolutionary cultural based particle swarm optimization algorithm // China: Life system modeling and intelligent. Communications in computer and information science. 2010. Vol. 98, no. 1. P. 1-7.

8. Jie J., Han Ch., Zeng J. An Extended Mind Evolutionary Computation Model for Optimizations // Applied Mathematics and Computation. 2007. Vol. 185, no. 2. P. 1038-1049.

9. Федоренко Р.П. Приближенное решение задач оптимального управления. М.: Наука, 1978. 488 с.

10. Карпенко А.П., Федорук В.Г. Один класс прямых адаптивных методов многокритериальной оптимизации // Информационные технологии. 2009. № 5. С. 24-30.

SCIENTIFIC PERIODICAL OF THE BAUMAN MSTU

SCIENCE and EDUCATION

EL № FS77 - 48211. №0421200025. ISSN 1994-0408

electronic scientific and technical journal

# Co-evolutionary algorithm of global optimization based on particle swarm optimization

**# 11, November 2013**

**DOI: 10.7463/1113.0619595**

**Karpenko A.P., Vorob'eva E.Yu.**

Bauman Moscow State Technical University, 105005, Moscow, Russian Federation

apkarpenko@mail.ru

vorobeva_ey@mail.ru

This paper deals with a co-evolutionary algorithm of global optimisation, Co-PSO, based on particle swarm optimisation. Implementation of this algorithm developed in MATLAB was described in this work. This software is oriented to a parallel operation of the given number of PSO algorithms at the logical level. These algorithms use various neighbourhood topologies of particles and/or various values of their free parameters. Results of a comprehensive numerical experiment on performance study of Co-PSO and its implementation were also presented. The experiment was based on Rosenbrock, Himmelblau and Rastrigin benchmark functions. Experimental results show superiority of the Co-PSO algorithm over the canonical PSO. A three-criterion problem of optimal control for a spacecraft during its reentry was solved with the use of Co-PSO. Methods of additive scalar convolution and reduction of the optimal control problem to a nonlinear programming problem were used in this work.

**Publications with keywords:** PSO, hybridization, Co-evolution
**Publications with words:** PSO, hybridization, Co-evolution

References

1. Vorob'eva E.Yu., Karpenko A.P., Seliverstov E.Yu. Ko-gibridizatsiya algoritmov roya chastits [Co-hybridization of PSO]. *Nauka i obrazovanie MGTU im. N.E. Baumana* [Science and Education of the Bauman MSTU], 2012, no. 4. Available at: http://www.technomag.edu.ru/doc/355729.html , accessed 27.08.2013.

2. Karpenko A.P., Svianadze Z.O. Meta-optimization based on self-organizing map and genetic algorithm. *Optical Memory and Neural Networks (Information Optics)*, 2011, vol. 20, no. 4, pp. 279-283. http://dx.doi.org/10.3103/S1060992X11040059

3. Karpenko A.P., Mitina E.V., Semenikhin A.S. Kogeneticheskiy algoritm Pareto-approksimatsii v zadache mnogokriterial'noy optimizatsii [Co-generic algorithm for Pareto set approximation]. *Informatsionnye tekhnologii*, 2013, no. 1, pp. 22-32.

4. Karpenko A.P., Seliverstov E.Iu. Global'naia optimizatsiia metodom roia chastits. Obzor. [Overall optimization by method of particles swarm. Review]. *Informatsionnye tekhnologii*, 2010, no. 2, pp. 25-34.

5. Yongquan Zhou, Shengyu Pei. A hybrid co-evolutionary particle swarm optimization algorithm for solving constrained engineering design problems. *China: Journal of computers*, 2010, vol. 5, no. 6, pp. 965-972.

6. Yuhui Shi, Renato A. Krohling. Co-evolutionary particle swarm optimization to solve min-max problems. *USA: Proceedings of the Evolutionary Computation*, 2002, vol. 2, pp. 1682-1687.

7. Yang Sun, Lingbo Zhang, Xingsheng Gu. Co-evolutionary cultural based particle swarm optimization algorithm. *China: Life system modeling and intelligent. Communications in computer and information science*, 2010, vol. 98, no. 1, pp. 1-7.

8. Jie J., Han Ch., Zeng J. An Extended Mind Evolutionary Computation Model for Optimizations. *Applied Mathematics and Computation*, 2007, vol. 185, no. 2. pp. 1038-1049.

9. Fedorenko R.P. *Priblizhennoe reshenie zadach optimal'nogo upravleniya* [Approximate solution of optimal control problems]. Moscow, Nauka, 1978. 488 p.

10. Karpenko A.P., Fedoruk V.G. Odin klass pryamykh adaptivnykh metodov mnogokriterial'noy optimizatsii [A class of direct adaptive methods for multi-criteria optimization]. *Informatsionnye tekhnologii*, 2009, no. 5, pp. 24-30.