

# ИССЛЕДОВАНИЕ ЭФФЕКТИВНОСТИ СТАЙНОГО АЛГОРИТМА ДЛЯ ЗАДАЧ МНОГОКРИТЕРИАЛЬНОЙ ОПТИМИЗАЦИИ

Ахмедова Ш. А.,

научный руководитель д-р техн. наук Семенкин Е.С.

*Сибирский Федеральный Университет*

В работе исследовалась эффективность «стадного» алгоритма (Particle Swarm Optimization, PSO) на многокритериальных задачах условной и безусловной оптимизации. Было проведено сравнение результатов, полученных вещественным и бинарным PSO.

В общем виде многокритериальная задача оптимизации (МКО-задача) включает набор из  $N$  переменных, множество  $K$  целевых функций от этих переменных и множество  $M$  ограничений. При решении МКО-задачи необходимо найти оптимум по совокупности  $K$  критериев. В работе был применен «стадный» алгоритм для приближенного построения множества и фронта Парето. Для этого был создан архив, в котором сохранялись недоминируемые по Парето решения, обновляемый на каждой итерации. При использовании алгоритма PSO для решения МКО-задачи основной проблемой было то, какую позицию считать глобально лучшей позицией для частицы. Существует несколько алгоритмов решения этой проблемы, одним из которых является  $\sigma$ -алгоритм. Рассмотрим схему этого алгоритма на примере двухкритериальной задачи. В этом случае  $\sigma$ -параметром  $i$ -ой частицы с координатами  $x^i$  называется величина:

$$\sigma_i = \sigma(f(x^i)) = \frac{f_1^2(x^i) - f_2^2(x^i)}{f_1^2(x^i) + f_2^2(x^i)}.$$

В случае, когда размерность пространства критериев больше двух  $\sigma$ -алгоритм использует  $K$ -мерный вектор  $\sigma$ -параметров  $\sigma = (\sigma_1, \sigma_2, \dots, \sigma_K)$ . Тогда для  $i$ -ой частицы:

$$\sigma_i = \sigma(f(x^i)) = \frac{((f_1^2(x^i) - f_2^2(x^i))(f_2^2(x^i) - f_3^2(x^i)) \dots (f_K^2(x^i) - f_1^2(x^i)))^T}{f_1^2(x^i) + f_2^2(x^i) + \dots + f_K^2(x^i)}.$$

$\sigma$ -алгоритм использует архив недоминируемых решений (обозначим этот архив  $S$ ). В  $\sigma$ -алгоритме для  $j$ -ой частицы на каждой итерации глобально лучшая позиция в  $S$  определяется по следующему правилу:

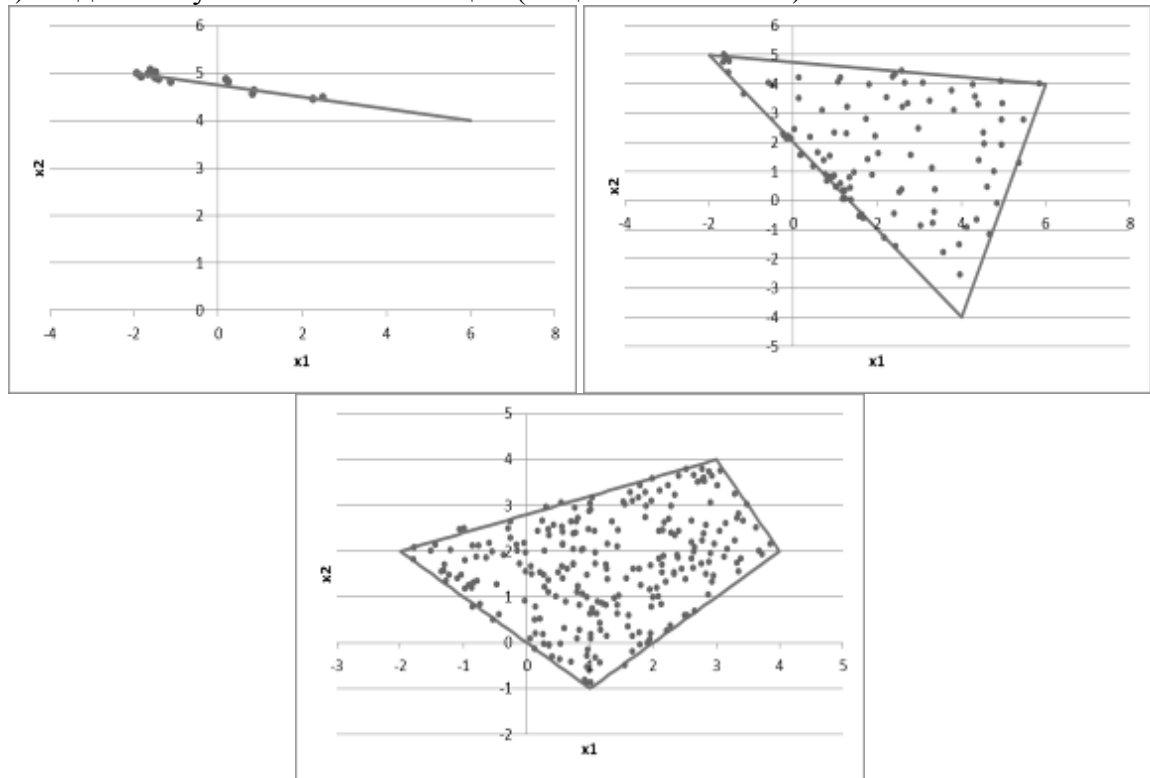
1. Определяем  $\sigma$ -параметр  $j$ -ой частицы, то есть вычисляем величину  $\sigma_i = \sigma(f(x^j))$ .
2. Находим в архиве частицу, у которой  $\sigma$ -параметр наиболее близок к величине  $\sigma_i$ :  $\min_i \|\sigma_j - \sigma_i\| = \|\sigma_j - \sigma_l\|$ , где  $\sigma_i$  –  $\sigma$ -параметр  $i$ -ой частицы из  $S$ ,  $\|\cdot\|$  – символ евклидовой нормы.
3. В качестве глобальной лучшей позиции для  $j$ -ой частицы принимаем вектор координат  $l$ -ой частицы.

Исследование эффективности «стадного» алгоритма проводилось на тестовых задачах безусловной (№1-3) и условной (№4-7) оптимизации, описанных ниже.

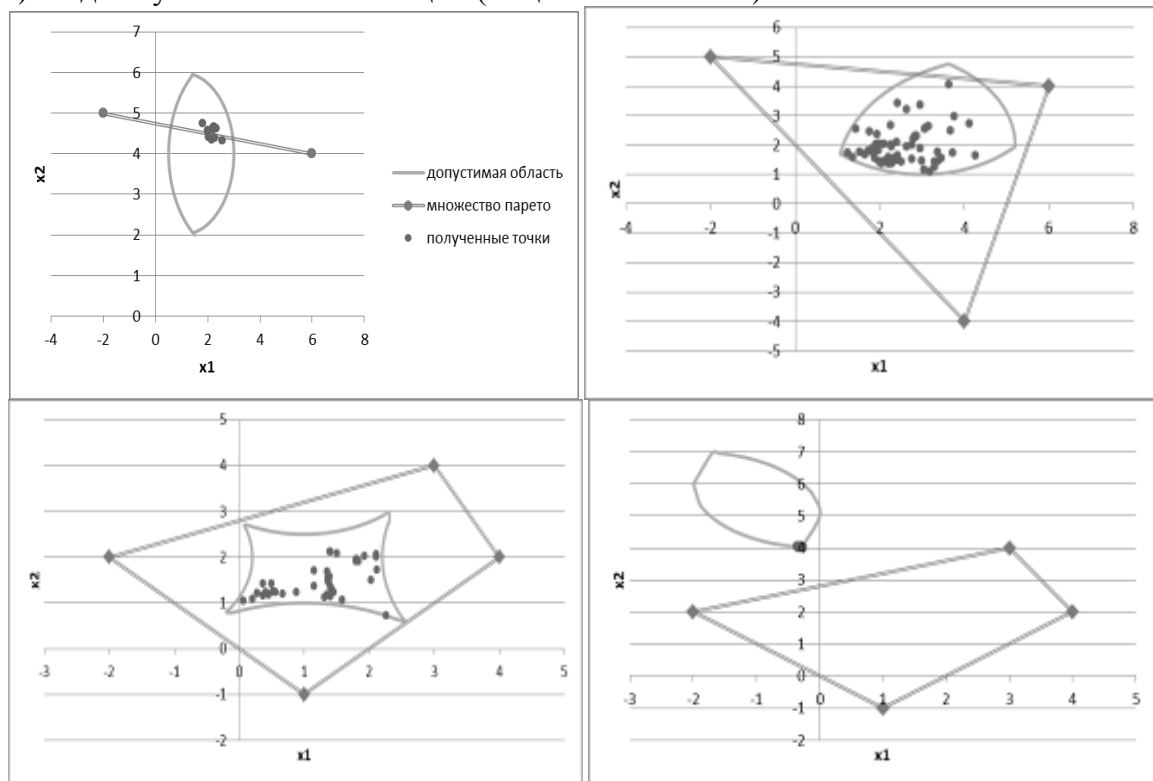
1) $f_1(x, y) = (x - 6)^2 + (y - 4)^2 \rightarrow \min$ $f_2(x, y) = (x + 2)^2 + (y - 5)^2 \rightarrow \min$	2) $f_1(x, y) = (x - 6)^2 + (y - 4)^2 \rightarrow \min$ $f_2(x, y) = (x + 2)^2 + (y - 5)^2 \rightarrow \min$ $f_3(x, y) = (x - 4)^2 + (y + 4)^2 \rightarrow \min$
3) $f_1(x, y) = (x - 1)^2 + (y + 1)^2 \rightarrow \min$ $f_2(x, y) = (x + 2)^2 + (y - 2)^2 \rightarrow \min$	$f_3(x, y) = (x - 3)^2 + (y - 4)^2 \rightarrow \min$ $f_4(x, y) = (x - 4)^2 + (y - 2)^2 \rightarrow \min$
4) $f_1(x, y) = (x - 6)^2 + (y - 4)^2 \rightarrow \min$ $f_2(x, y) = (x + 2)^2 + (y - 5)^2 \rightarrow \min$	$\begin{cases} g_1(x, y) = (x - 1)^2 + (y - 4)^2 \leq 4 \\ g_2(x, y) = (x - 3)^2 + (y - 4)^2 \leq 6.25 \end{cases}$
$f_1(x, y) = (x - 6)^2 + (y - 4)^2 \rightarrow \min$ 5) $f_2(x, y) = (x + 2)^2 + (y - 5)^2 \rightarrow \min$ $f_3(x, y) = (x - 4)^2 + (y + 4)^2 \rightarrow \min$	$\begin{cases} g_1(x, y) = (x - 2)^2 + (y - 2)^2 \leq 10.24 \\ g_2(x, y) = (x - 5)^2 + (y - 1)^2 \leq 16 \\ g_3(x, y) = (x - 3)^2 + (y - 4)^2 \leq 9 \end{cases}$
$f_1(x, y) = (x - 1)^2 + (y + 1)^2 \rightarrow \min$ 6) $f_2(x, y) = (x + 2)^2 + (y - 2)^2 \rightarrow \min$ $f_3(x, y) = (x - 3)^2 + (y - 4)^2 \rightarrow \min$ $f_4(x, y) = (x - 4)^2 + (y - 2)^2 \rightarrow \min$	$\begin{cases} g_1(x, y) = (x + 1.8)^2 + (y - 2)^2 \geq 4 \\ g_2(x, y) = (x - 1)^2 + (y - 4.5)^2 \geq 4 \\ g_3(x, y) = (x - 5.2)^2 + (y - 2)^2 \geq 9 \\ g_4(x, y) = (x - 1)^2 + (y + 2)^2 \geq 9 \\ g_5(x, y) = (x - 1)^2 + (y - 2)^2 \leq 6.25 \end{cases}$
$f_1(x, y) = (x - 1)^2 + (y + 1)^2 \rightarrow \min$ 7) $f_2(x, y) = (x + 2)^2 + (y - 2)^2 \rightarrow \min$ $f_3(x, y) = (x - 3)^2 + (y - 4)^2 \rightarrow \min$ $f_4(x, y) = (x - 4)^2 + (y - 2)^2 \rightarrow \min$	$\begin{cases} g_1(x, y) = x^2 + (y - 6)^2 \leq 4 \\ g_2(x, y) = (x + 2)^2 + (y - 5)^2 \leq 4 \end{cases}$

Для работы алгоритма было установлено максимальное число частиц, которое может храниться в архиве недоминируемых решений (для каждой задачи был установлен свой «размер» архива). Для задач условной оптимизации был применен метод динамических штрафов. При решении задач заполнялась лишь некоторая часть архива. Результаты показали, что с ростом числа критериев эффективность алгоритмов возрастала. Так, например, при решении задач условной оптимизации и вещественным, и бинарным «стайнными» алгоритмами архив недоминируемых решений заполнялся в среднем на 20-30%. Преимущество вещественного PSO заключалось лишь во времени, потраченном на один прогон. И вновь результаты алгоритмов существенно не отличались. Количество частиц и поколений было примерно таким же, как и при решении задач безусловной оптимизации. Решение 4-ой задачи условной оптимизации, особенность которой заключается в том, что в допустимой области нет ни одной точки из множества Парето, потребовало заметного увеличения размера популяции, и в конечном итоге были получены точки, находящиеся на той части границы допустимой области, которая была ближе всего к множеству Парето. Кроме того, результаты, полученные вещественным и бинарным PSO почти не отличались. Ниже представлены полученные множества Парето для соответствующих задач с помощью PSO (для задач безусловной оптимизации кривые и фигуры на графиках – искомые множества Парето, точки – получены с помощью «стаинового» алгоритма).

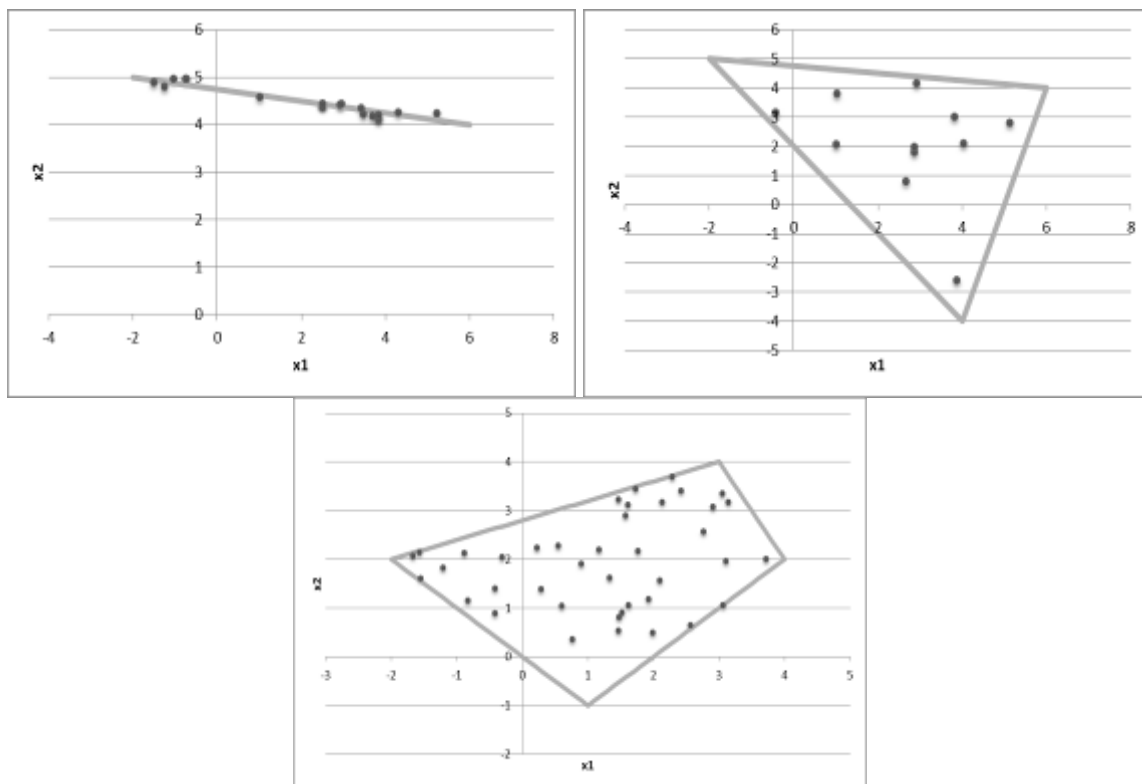
1) Задачи безусловной оптимизации (вещественный PSO):



2) Задачи условной оптимизации (вещественный PSO):



1) Задачи безусловной оптимизации (бинарный PSO):



## 2) Задачи условной оптимизации (бинарный PSO):

