НАУЧНОЕ ИЗДАНИЕ МГТУ ИМ. Н. Э. БАУМАНА

НАУКА и ОБРАЗОВАНИЕ

Эл № ФС77 - 48211. Государственная регистрация №0421200025. ISSN 1994-0408

электронный научно-технический журнал

Глобальная оптимизация на основе гибридизации алгоритмов поиска гармонии и роя частиц

 # 07, июль 2014

DOI: 10.7463/0714.0717653

профессор, д.ф.-м.н. Карпенко А. П.[1,a], Печенина Т. В.[2], Буланов В. А.[1]

УДК 519.6

[1]Россия, МГТУ им. Н.Э. Баумана
[2]ООО АСКОН Санкт-Петербург
[a]pkarpenko@mail.ru

В работе представлена авторская гибридизация алгоритмов поиска гармонии и роя частиц. Алгоритм призван устранить или уменьшить влияние недостатков известного алгоритма Global-best Harmony Search (GHS), предложенного в 2008 г. в работе Omran M.G.H., Mahdavi M. Описана программная реализация алгоритма и результаты исследования его эффективности при решении тестовых задачах глобальной оптимизации многомерных многоэкстремальных функций Растригина и Шекеля, а также практически значимой задачи оптимизации ферменной конструкции. Результаты исследования показали эффективность принятых алгоритмических и программных решений. Так, в результате решения последней задачи были получены результаты, лучшие аналогичных результатов, полученных другими авторами.

Ключевые слова: Глобальная оптимизация, гибридизация, алгоритм поиска гармонии, алгоритм роя частиц

Введение

В 80-х годах прошлого века началась разработка класса стохастических поисковых алгоритмов глобальной оптимизации, называемых поведенческими, интеллектуальными, метаэвристическими, вдохновлёнными (инспирированными) природой, роевыми, многоагентными, популяционными и т.д. Используем последний термин. Популяционные алгоритмы предполагают одновременную обработку нескольких вариантов решения задачи оптимизации, в то время как в классических («траекторных») поисковых алгоритмах в области поиска эволюционирует только один кандидат на решение этой задачи.

Опыт решения сложных задач глобальной оптимизации с помощью популяционных алгоритмов показывает, что применение одного такого алгоритма далеко не всегда является эффективным. Поэтому в настоящее время большое внимание уделяют гибридизации популяционных алгоритмов глобальной оптимизации. Гибридные алгоритмы объединяют

различные алгоритмы либо одинаковые алгоритмы, но с различными значениями свободных параметров. При этом эффективность одного алгоритма может компенсировать слабость другого [1 - 3].

Целями данной работы являются разработка гибридного алгоритма глобальной оптимизации, основанного на алгоритмах поиска гармонии и роя частиц; программная реализация алгоритма; исследование его эффективности на ряде известных тестовых задач, а также задаче размернойоптимизации ферменной конструкции.

Алгоритм поиска гармонии (*HarmonySearchalgorithm, HS*) предложен в 2001 Гимом (Z. W. *Geem*) [4]. Алгоритм вдохновлен процессом поиска музыкантами гармонии в музыке. Ситуации идеальной гармонии звуков алгоритм *HS*ставит в соответствие глобальный экстремум в задаче многомерной оптимизации, а процессу импровизации музыканта – процедуру поиска этого экстремума.

Основными особенностями алгоритма *HS* являются его простота и возможность использования для решения задач как непрерывной, так и дискретной оптимизации. В силу этого алгоритм привлекает значительное внимание исследователей и быстро развивается как в теоретическом, так и в практическом планах. Канонический алгоритм *HS* показал себя эффективным при решении значительного числа различных прикладных задач оптимизации. Однако алгоритм не свободен от недостатков, наиболее существенными из которых являются низкая эффективность локального поиска и сложность подбора значений свободных параметров алгоритма. Известно большое число работ, в которых предлагаются различные модификации канонического алгоритма *HS*, имеющие целью преодоление его указанных недостатков.

В работе [5] предложен вариант алгоритма *HS* с динамической настройкой свободных параметров (*Improved HS, IHS*).

Модификация алгоритма поиска гармонии, получившая название *Global-best HS* (*GHS*), представлена в работе [6]. Суть модификации заключается в добавлении в алгоритм *IHS* некоторых механизмов, используемых алгоритмом оптимизации роем частиц (*Particle Swarm Optimization, PSO*) [7]. Улучшенный вариант алгоритма *GHS*, названный авторами *Improved Global Harmony Search* (*IGHS*), предложен в работе [8].

Алгоритм *HS* с само адаптирующимися свободными параметрами рассмотрен в работе [9]. Ещё один вариант само адаптирующегося алгоритма*HS*, получивший название (*Novel Self-adaptive HS* (NHS), представлен в работе [10].

Хорошо известно, что популяционные алгоритмы в своих канонических вариантах эффективны в исследовании пространства поиска и отыскании в нем перспективных областей, но медленно сходятся к локальным экстремумам. Поэтому предложено большое число гибридизаций алгоритма *HS* с алгоритмами локального поиска. Так в работе [11] представлен гибридный алгоритм, в котором в качестве локального используется алгоритм последовательного квадратичного программирования (*Sequential Quadratic Programming*). В работе [12] рассмотрена гибридизация алгоритма *HS* с алгоритмом имитации отжига (*Simulated Annealing*). Модификация алгоритма *HS*, основанная на объединении

этого алгоритма с алгоритмом дифференциальной эволюции (*Differential Evolution, DE*) представлена в работе [13].

В пункте 1 приводим постановку задачи. Пункт 2 посвящён базовым алгоритмам *HS* и *PSO*. В пункте 3 представлена схема предложенного гибридного алгоритма, названного HS^{PSO}. Пункт 4 содержит результаты вычислительного эксперимента с разработанным алгоритмическим и программным обеспечением. В пункте 5. представляем постановку задачи размерной оптимизации ферменной конструкции и приводим результаты её решения с помощью алгоритма HS^{PSO}. В заключении формулируем основные результаты работы и перспективы её развития.

1. Постановка задачи

Рассматриваем детерминированную задачу глобальной безусловной минимизации

$$\min_{X \in R^n} F(X) = F(X^*) = F^*, \qquad (1)$$

где $F(X)$ - скалярная целевая функция, $F(X^*) = F^*$ - её искомое минимальное значение, $X = (x_1, x_2, \ldots, x_n)$ - n-мерный вектор варьируемых параметров, R^n - n-мерное арифметическое пространство. Полагаем, что фитнесс-функция $\varphi(X)$ также подлежит минимизации.

Параллелепипед допустимых значений вектора варьируемых параметров имеет вид

$$D = \left\{X \mid x_i^- \le x_i \le x_i^+, i \in [1:n]\right\} \subset R^n, \qquad (2)$$

где x_i^-, x_i^+ - заданные константы.

Заметим, что в вычислительной практике задачу глобальной условной оптимизации обычно сводят к задаче безусловной оптимизации, например, методом штрафных функций (п. 5).

2. Базовые алгоритмы

Канонический алгоритм HS [4, 14]**.** Музыкантам ставим в соответствие индивидов s_i, а оркестру – популяцию $S = \{s_i, i \in [1:N]\}$, где *N*– число индивидов популяции. Аккорду, который берет музыкант s_i в данный момент времени, сопоставляем значение вектора варьируемых параметров X_i. Гармонию звуков формализует глобальный минимум фитнесс-функции $\varphi(X)$. Совокупность текущих координат X_i, $i \in [1:N]$ всех индивидов популяции образует $(N \times n)$-матрицу *памяти гармоний* (*Harmony Memory*, *HM*). Число строк этой матрицы *N* носит название *размер памяти гармоний* и обозначается *HMS.*

Схема канонического алгоритма *HS* включает в себя следующие основные шаги.

1) Инициализируем алгоритм.

2) Формируем вектор гармонии.

3) Выполняем пошаговую настройку вектора гармонии.

4) Обновляем матрицу *HM*.

5) Завершаем итераций, если условие окончания итераций выполнено, возвращаемся к шагу 2 в противном случае.

Инициализация. Начальные значения компонентов вектора X_i, полагаем равномерно распределёнными в гиперпараллелепипеде *D*:

$$x_{i,j} = x_j^- + U_1(0;1)\left(x_j^+ - x_j^-\right),\ i \in [1:N],\ \ j \in [1:n]. \qquad (3)$$

Здесь $U_1(0;1)$- вещественное случайное число, равномерно распределённое в интервале $[0;1]$. Совокупность так построенных векторов X_i образует исходную матрицу памяти гармоний

$$HM = \begin{pmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{N,1} & x_{N,2} & \cdots & x_{N,n} \end{pmatrix}.$$

Формирование вектора гармонии X' выполняем по следующим правилам.

С вероятностью ξ_h в качестве компоненты $x'_{i,j}$ вектора X'_i используем соответствующую компоненту случайного вектора X_{i_1} из текущей матрицы HM:

$$x'_{i,j} = HM_{i_1,j},\quad i_1 = U_1(1:N),\ j \in [1:n].$$

Здесь $U_1(1:N)$- целое случайное число, равномерно распределённое в интервале $[1:N]$. Данная операция моделирует воспроизведение музыкантом какого-либо аккорда из его памяти гармоний и называется *операцией случайного выбора* (*random selection*).

С вероятностью $(1-\xi_h)$ в качестве компоненты $x'_{i,j}$ берём величину, сгенерированную по формуле вида (3). Операция адекватна ситуации формирования абсолютно случайного аккорда из доступного музыканту диапазона.

Свободный параметр алгоритма ξ_h имеет смысл *вероятности использования памяти гармоний* (*Harmony Memory Considering Rate*) и обозначается *HMCR*.

Пошаговая настройка вектора гармонии. Если компонента $x'_{i,j}$ вектора X'_i выбрана из *HM*, то выполняем следующие действия.

С вероятностью ξ_p изменяем эту компоненту по формуле

$$x'_{i,j} = x'_{i,j} + u_{sign}^{\pm 1} b_w N_1(0;1) \qquad (4)$$

или с вероятностью $(1-\xi_p)$ оставляем без изменений. Здесь $u_{sign}^{\pm 1}$- дискретная случайная величина, с равной вероятностью принимающая значения ± 1. Формула означает, что компонента $x'_{i,j}$ с равной вероятностью получает случайное положительное или отрицательное приращение величиной $b_w\ N_1(0;1)$. Параметр ξ_p определяет *вероятность изменения шага* (*Pitch Adjusting Rate*) и носит наименование *PAR*. Параметр b_w представляет собой величину шага, которому придаётся смысл *полосы пропускания инструмента* (*Pitch Bandwidth Rate*). В каноническом алгоритме *HS* этот параметр обозначают *BWR*.

Обновление матрицы HM выполняем по следующей схеме. Вычисляем значение фитнесс-функции $\varphi(X'_i)$, соответствующее сформированному вектору X'_i. Если $\varphi(X'_i\ ) < \varphi(X_{i_w})$, то худший вектор X_{i_w} памяти гармоний заменяем вектором X'_i. Вектор X_{i_w} находим из условия

$$\max_{i\in[1:N]} \varphi(X_i\ ) = \varphi(X_{i_w}).$$

Завершение итераций. Если условие окончания итераций выполнено, то в качестве решения задачи используем лучший вектор памяти гармоний X_{i_b}, доставляющий наименьшее значение функции приспособленности:

$$\min_{i\in[1:N]}\ \varphi(X_i\ ) = \varphi(X_{i_b}).$$

Рекомендуют следующие значения основных свободных параметров алгоритма: $HMS \le 50$; $HMCR = 0{,}7 \div 0{,}95$; $PAR = 0{,}1 \div 0{,}5$. Два последних параметра используют для регулирования соотношения диверсификационных и интенсификационных свойств алгоритма – меньшие значения этих параметров диверсифицируют поиск (за счёт, конечно, уменьшения скорости его сходимости).

Канонический алгоритм PSO [15]. В каноническом алгоритме *PSO* координаты частицы s_i на итерации $t \ge 0$ определят вектор $X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,n})^T$, а на итерации $(t+1)$ - вектор $X'_i = (x'_{i,1}, x'_{i,2}, \ldots, x'_{i,n})^T$, где ^{T} - символ транспонирования. Начальные координаты частицы s_i равны $X_i(0)$; $i \in [1:N]$.

Итерации в алгоритме*PSO* выполняем по схеме

$$X'_i = X_i + V_i,$$

$$V_i = b_I V_i^- + U_n(0;b_C) \otimes \left(X_i^* - X_i\right) + U_n(0;b_S) \otimes \left(X_i^{**} - X_i\right). \qquad (5)$$

Здесь использованы следующие обозначения: $\otimes$ - символ прямого произведения векторов; b_I, b_C, b_S - свободные параметры алгоритма; $V_i = V_i(t) = (v_{i,1}, v_{i,2}, \ldots, v_{i,n})^T$ -

$(n\times 1)$-вектор приращения координат частицы; $V_i^- = V_i(t-1)$; X_i^* - вектор координат частицы s_i, соответствующий её наилучшему значению фитнесс-функции за время поиска $[0:t]$, то есть

$$\min_{\tau\in[0:t]} \varphi(X_i(\tau)) = \varphi(X_i^*);$$

X_i^{**} - вектор координат соседней с данной частицы с наилучшим значением приспособленности, то есть

$$\min_{j\in N_i} \varphi(X_j^*) = \varphi(X_i^{**}), \qquad (3.4)(8)$$

где N_i - множество номеров частиц, являющихся «соседями» данной частицы s_i; $U_n(a;b)$ - $(n\times 1)$-вектор вещественных случайных чисел, каждое из которых равномерно распределено в интервале $[a;b]$; $a<b$.

Первое слагаемое в формуле (5) выполняет функции памяти частицы о её перемещении на предыдущей итерации и называется *инерционной компонентой.* Второе слагаемое в той же формуле называют *когнитивной компонентой*, поскольку оно формализует тенденцию частицы s_i вернуться из текущего положения X_i в положение X_i^*. Третье слагаемое формулы (3) именуют *социальной компонентой*. Компонента отражает стремление частицы s_i переместиться в направлении своей наиболее успешной соседней частицы.

Модифицированный алгоритм GHS [6, 8]**.** Основная идея модификации заключается в изменении в алгоритме *HS* правила генерации нового созвучия таким образом, чтобы это созвучие с заданной вероятностью *PAR* могло подражать наилучшему созвучию в *HM*. Такая модификация добавляет социальную модель поведения частиц алгоритма *PSO*в канонический алгоритм *HS*. Псевдокод модифицированного алгоритма генерации нового созвучия имеет следующий вид:

for $i\in[1:HMS]$, $j\in[1:n]$ **do**
  if $U(0;1)\le HMCR$ **then**
    $x'_{i,j} = x_{k,j}$, $k = U[1:HMS]$
    if $U(0;1)\le PAR$ **then**
      $x'_{i,j} = x_{*,l}$, $l = U[1:n]$
  else
    $x'_{i,j} = x_i^- + U(0;1)\left(x_i^+ - x_i^-\right)$

Здесь $x_{*,l}$ - l-ая компонента лучшего созвучия текущей памяти гармоний.

Алгоритм *GHS* сохраняет важные преимущества алгоритма *HS* - простота структуры и невысокая вычислительная сложность. В то же время, данная модификация не меняет процедуру обновления *HM*, используемую в каноническом алгоритме *HS*, суть которой состоит в подражании новых гармоний наилучшему решению в *HM*. В результате, в про-

цессе поиска из *HM* исчезают все решения, ухудшающие значение целевой функции. В совокупности с взятой из алгоритма *PSO* социальностью, это позволяет ускорить процесс сходимости алгоритма, но повышает риск «застревания» алгоритма в локальном минимуме, поскольку разнообразие созвучий в *HM* уменьшается.

3. Гибридный алгоритм HS^{PSO}

Гибридный алгоритм HS^{PSO} призван устранить или уменьшить влияние указанных недостатков алгоритма *GHS*. Суть предложенной модификации алгоритма *GHS* заключается в следующем изменении процедуры обновления *HM*: если новое сгенерированное созвучие лучше, чем *случайно* выбранное из *HM*, то новое созвучие включается в *HM*, а выбранное исключается из неё. Заметим, что подобная стратегия используется в генетических алгоритмах, когда в популяции с некоторой вероятностьюсохраняют особей с невысокой приспособленностью, позволяя тем самымсохранить разнообразие популяции и диверсифицировать поиск.

В качестве критерия останова алгоритм HS^{PSO} использует стагнацию вычислительного процесса: если лучшее решение, достигнутое алгоритмом, изменяется в течение заданного числа итераций NI_{stagn} на фиксированную величину, меньшую или равную $\varepsilon > 0$, то алгоритм завершает своё выполнение.

Схема алгоритма имеет следующий вид.

Шаг 1. Инициализация алгоритма.

Шаг 2. Инициализация *HM* - заполнение *HM* случайно сгенерированными по формуле (1) созвучиями, число которых равно размеру гармонической памяти *HMS*.

Шаг 3. Импровизация нового созвучия. Новое созвучие создаётся по схеме канонического алгоритма *HS* и состоит из просмотра *HM*, регулировки тона и случайного выборы.

Шаг 4. Обновление *HM*. Если новый вектор решения X' лучше, чем случайно выбранный из *HM*, то вектор X' включаем в *HM*, а выбранный вектор исключаем из неё.

Шаг 5. Проверка критерия останова. Прекращаем вычисления, если имеет место стагнация вычислительного процесса (или если достигнуто максимально допустимое число итераций). В противном случае переходим к шагу 3.

Так же, как алгоритм *GHS* [6, 8], алгоритм HS^{PSO} использует динамические параметры *PAR*, *bw*:

$$PAR = PAR(t) = PAR_{\min} + \frac{PAR_{\max} - PAR_{\min}}{NI_{\max}} t ; \qquad (6)$$

$$b_w = b_w(t) = b_w^{\max} \exp\left(\frac{\ln\left(b_w^{\min} / b_w^{\max} \right)}{NI_{\max}} t \right). \qquad (7)$$

Здесь $PAR_{\min}, PAR_{\max}, b_w^{\min}, b_w^{\max}$ - заданные константы; $NI_{\max}$ - заданное максимально допустимое число итераций.

4. Программная реализация и исследование эффективности

Программная реализация алгоритма HS^{PSO} выполнена на языке программирования *C++* в среде разработки *MicrosoftVisualStudio* 2008. Основные функции, используемые программой, перечислены ниже:

VoidInitialize_Harmony_Memory () – заполнение памяти гармоний *HM* случайно сгенерированными созвучиями;

VoidFindBestinHM () – поиск лучшего решения в *HM*;

VoidFindWorstinHM () – поиск худшего решения в *HM*;

VoidImprovise_new_Harmony () – импровизацию нового созвучия;

VoidUpdate_HM () – обновление *HM*;

VoidHarmony_Search () – основная функция, реализующая собственно алгоритм HS^{PSO}.

Используем мультистарт с числом запусков алгоритма, равным 30. В качестве критериев эффективности алгоритма рассматриваем следующие величины: *A* - оценка вероятности попаданий в область точного решения с заданной погрешностью $\varepsilon_F = 0{,}001$ (точность алгоритма); *MF* - оценка математического ожидания найденного алгоритмом минимального значения целевой функции $F(X)$; *MI* - оценка математического ожидания числа итераций до начала процесса стагнации; *ME* – оценка математического ожидания числа испытаний (вычислений значений целевой функции); оценки среднегоквадратичного отклонения трёх последних величин $\sigma_F, \sigma_{MI}, \sigma_{ME}$ соответственно.

Вычислительные эксперименты выполнены на компьютере с процессором *IntelCore*2 *DuoProcessor T*7200 и оперативной памятью объёмом 2,00 ГБ под управлением операционной системой *Windows*7.

Выполнено исследование зависимости эффективности алгоритма HS^{PSO} от значений его свободных параметров *PAR* и *HMS*. В качестве тестовых использованы два ярких представителя многоэкстремальных функций – функции Растригина и Шекеля [16]. Рассмотрены размерности вектора варьируемых параметров *X*этих функций, равные 8, 16, 32.

Поскольку в алгоритме HS^{PSO} параметр *PAR* является динамическим (см. формулу (6)), варьировалась верхняя граница этого параметра $PAR_{\max}$, принимая значения 0,4, 0,65, 0,9. Нижняя граница $PAR_{\min}$ принята постоянной и равной 0,01. Исследование выполнено для размеров памяти гармоний *HMS*=25, 50, 100. В соответствии с рекомендациями работы [6], значение параметра *HMCR* принято равным 0,95. Значения параметра b_w менялись в процессе итераций от $b_w^{\min} = 0{,}001$ до $b_w^{\max} = 0{,}01$ по формуле (7). Если не оговорено иное, использованы следующие значения остальных свободных параметров:

максимально допустимое число итераций $NI_{\max}=10000$; число итераций до начала процесса стагнации $NI_{stagn}=1000$; соответствующая точность $\varepsilon=10^{-6}$.

4.1. Функция Растригина

Решение отыскивается в области

$$D=\{X\,|-5\le x_i\le 5, i\in[1:n]\}.$$

Минимальное значение функции Растригина в этой области достигается в точке $X^*=\mathbf{0}$ и равно нулю.

Результаты исследования представлены в таблицах 1 – 3 и на иллюстрирующих их рисунках 1 – 4.

Таблица 1 – Результаты исследования эффективности алгоритма при варьировании величин $PAR_{\max}$, *HMS*: функция Растригина; $NI_{\max}=10000$; $n=8$

HMS	25			50			100		
PAR_{max}	0,4	0,65	0,9	0,4	0,65	0,9	0,4	0,65	0,9
MF	0,1	0,07	0,26	0,1	0,23	0,17	0,07	0,13	0,23
σ_F	0,3	0,25	0,5	0,3	0,49	0,45	0,25	0,42	0,49
$F^*(X^*)$	0	0	0	0	0	0	0	0	0
MI	8610	8278	6908	8792	7883	7786	9870	8691	8260
σ_{NI}	1290	1401	1657	1181	1457	1568	291	1411	1687
ME	8635	8303	6933	8842	7933	7836	9970	8791	8360
σ_{NE}	1290	1401	1657	1181	1457	1662	291	1411	1687
$M[\lvert X-X^*\rvert]$	0,08	0,07	0,28	0,11	0,17	0,15	0,07	0,11	0,17
A, %	90	90	76	90	80	86	90	90	80

Таблица 2 – Результаты исследования эффективности алгоритма при варьировании величин $PAR_{\max}$, *HMS*: функция Растригина; $NI_{\max}=10000$; $n=16$

HMS	25			50			100		
PAR_{max}	0,4	0,65	0,9	0,4	0,65	0,9	0,4	0,65	0,9
MF	0,72	0,62	0,75	0,69	0,79	1,1	2,58	1,10	1,17
σ_F	0,82	0,75	0,84	0,53	1,01	1,02	0,89	0,95	0,95
$F^*(X^*)$	0	0	0	0,01	0	0	1,08	0	0
MI	9996	9848	9701	10000	9964	9874	10000	10000	10000
σ_{NI}	20	572	643	0	193	471	0	0	0
ME	10021	9873	9726	10050	10014	9924	10100	10100	10100
σ_{NE}	20	572	643	0	193	471	0	0	0
$M[\lvert X-X^*\rvert]$	0,47	0,45	0,4	0,7	0,58	0,74	0,86	0,66	0,87
A, %	36	43	40	0	26	20	0	0	20

Таблица 3 – Результаты исследования эффективности алгоритма при варьировании величин $PAR_{\max}$, *HMS*: функция Растригина; $NI_{\max}=10000$; $n=32$

HMS	25			50			100		
PAR_{max}	0,4	0,65	0,9	0,4	0,65	0,9	0,4	0,65	0,9
MF	14,08	11,66	9,8	18,66	15,52	12,8	23,03	19,57	15,65
σ_F	2,65	1,94	1,79	2,86	2,12	2,5	3,64	3,25	2,66
$F^*(X^*)$	9,98	6,5	6,41	13,48	12,17	6,65	16,18	12,11	10,78
MI	10000	10000	10000	10000	10000	10000	10000	10000	10000
σ_{NI}	0	0	0	0	0	0	0	0	0
ME	10025	10025	10025	10050	10050	10050	10100	10100	10100
σ_{NE}	0	0	0	0	0	0	0	0	0
$M[\lvert X-X^*\rvert]$	2,9	2,64	2,64	3,14	3,04	2,87	3,65	3,27	3,04
A, %	0	0	0	0	0	0	0	0	0

На указанных рисунках и далее, если не оговорено иное, пунктиром показаны величины $MF \pm \sigma_F$, $ME \pm \sigma_E$ и т.д.

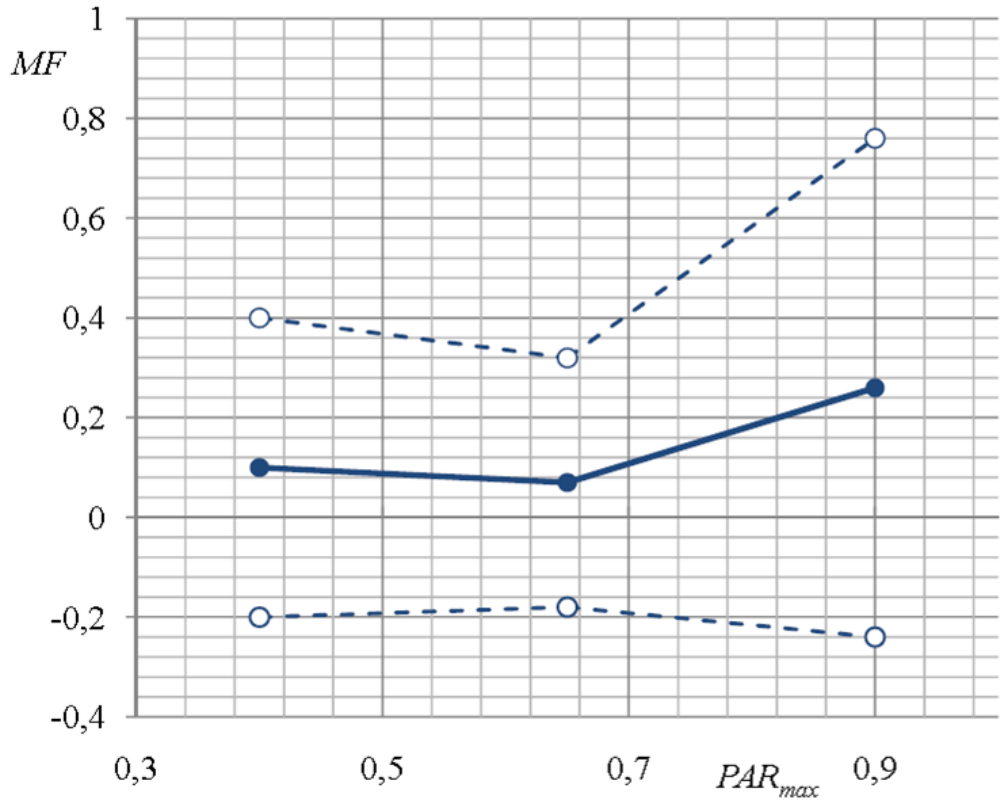

Рисунок 1 – Значения критерия эффективности *MF* в функции параметра $PAR_{\max}$: функция Растригина; $NI_{\max}=10000$; $HMS=25$; $n=8$

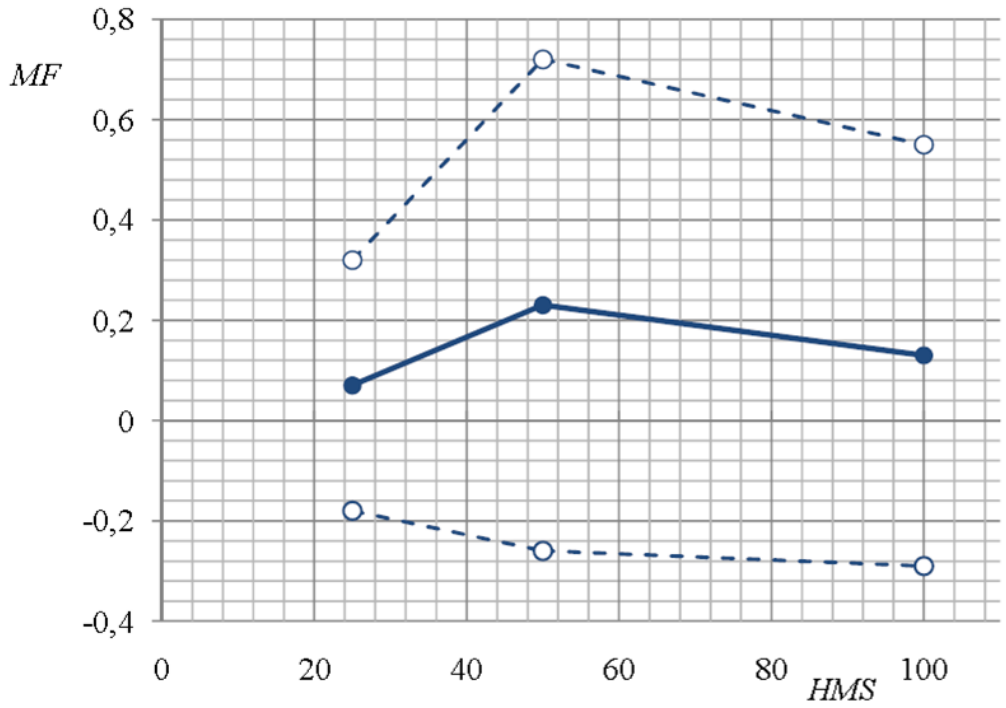

Рисунок 2 – Значения критерия эффективности *MF* в функции параметра *HMS*: функция Растригина;

$$NI_{\max} = 10000;\ PAR_{\max} = 0{,}65; n = 8$$

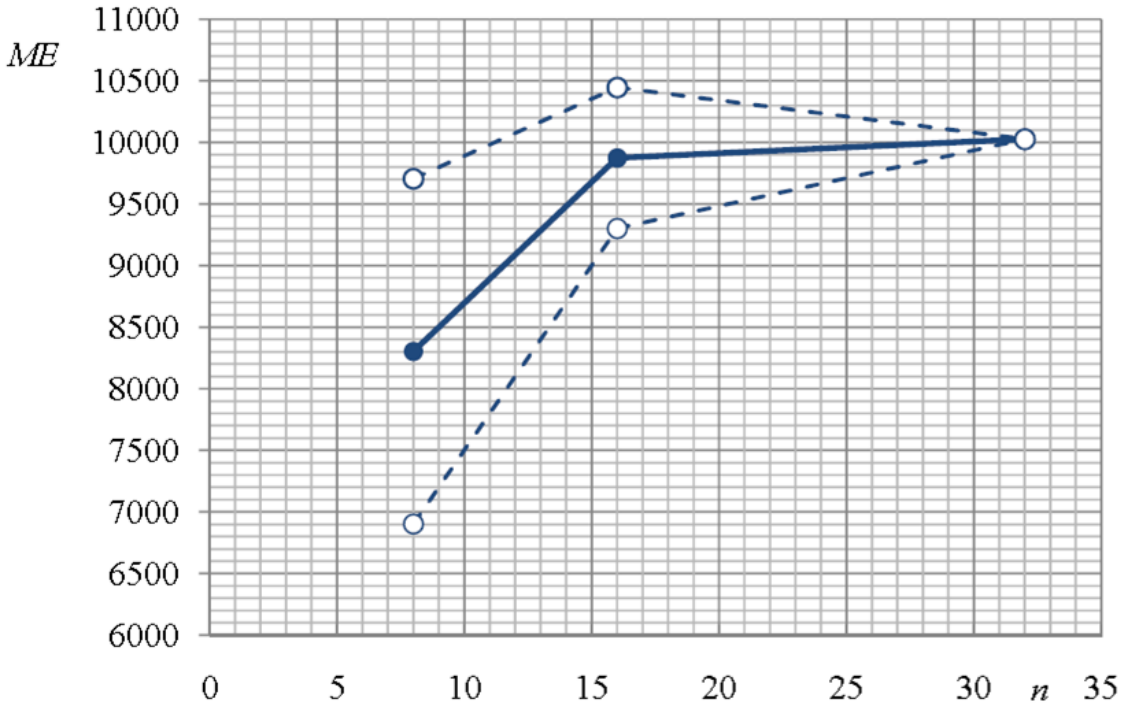

Рисунок 3 – Значения критерия эффективности *MF* в функции параметра n : функция Растригина;

$$NI_{\max} = 10000;\ PAR_{\max} = 0{,}65; HMS{=}25$$

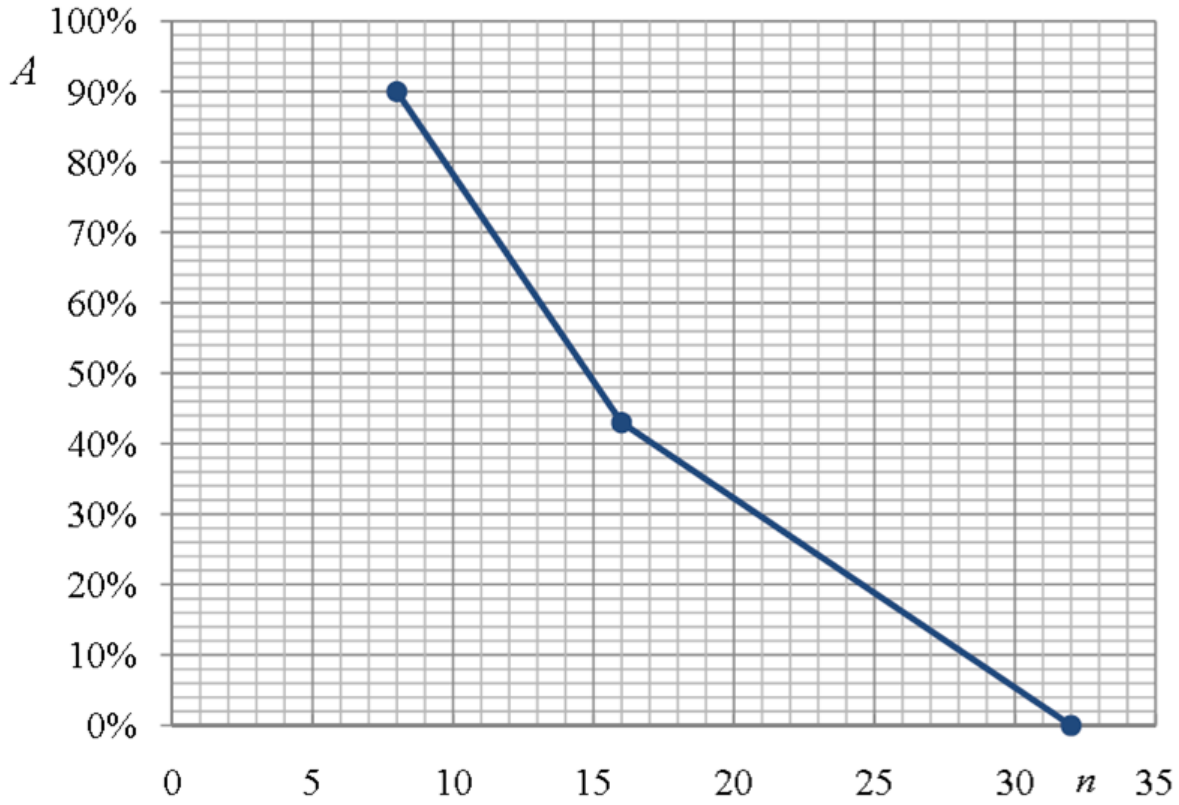

Рисунок 4 – Значения критерия эффективности *A* в функции параметра n : функция Растригина; $NI_{\max}=10000$; $PAR_{\max}=0{,}65$;*HMS*=25

Из представленных результатов следует, что алгоритм с высокой вероятностью локализует глобальный минимум функции Растригина при размерности пространства поиска *n*=8. При этом наилучший результат по точности достигается при сочетании значений параметров *HMS*=25, $PAR_{\max}=0{,}65$. С увеличением размерности пространства поиска, вероятность локализации минимума уменьшается, а среднее число итераций увеличивается до своего максимально допустимого значения $NI_{\max}=10000$, то есть при высокой размерности вектора *X* состояние стагнации вычислительного процесса не наступает.

Результаты вычислительных экспериментов, полученные при увеличении максимального числа итераций до $NI_{\max}=50000$, представлены в таблице 4 и на рисунках 5, 6. Сплошная линия на рисунке 6 соответствует случаю, когда в качестве условия окончания итераций используется стагнация вычислений; пунктирная линия - достижение максимального числа итераций $NI_{\max}$.

Таблица 4 – Результаты исследования эффективности алгоритма при варьировании величины *n*: функция Растригина; $NI_{\max}=50000$; $PAR_{\max}=0{,}65$, *HMS*=25

n	8	16	32
$F(X^*)$	0,07	0	0,01
MF	0,25	0,3	0,574
σ_F	0	0,637	0,8
$M[NI]$	8278	12474	27537
σ_{NI}	1401	2237	3862
$M[NE]$	8303	12499	27562
σ_{NE}	1401	2237	3862
$M[\|X-X^*\|]$	0,07	0,243	0,467
A , %	90	80	16

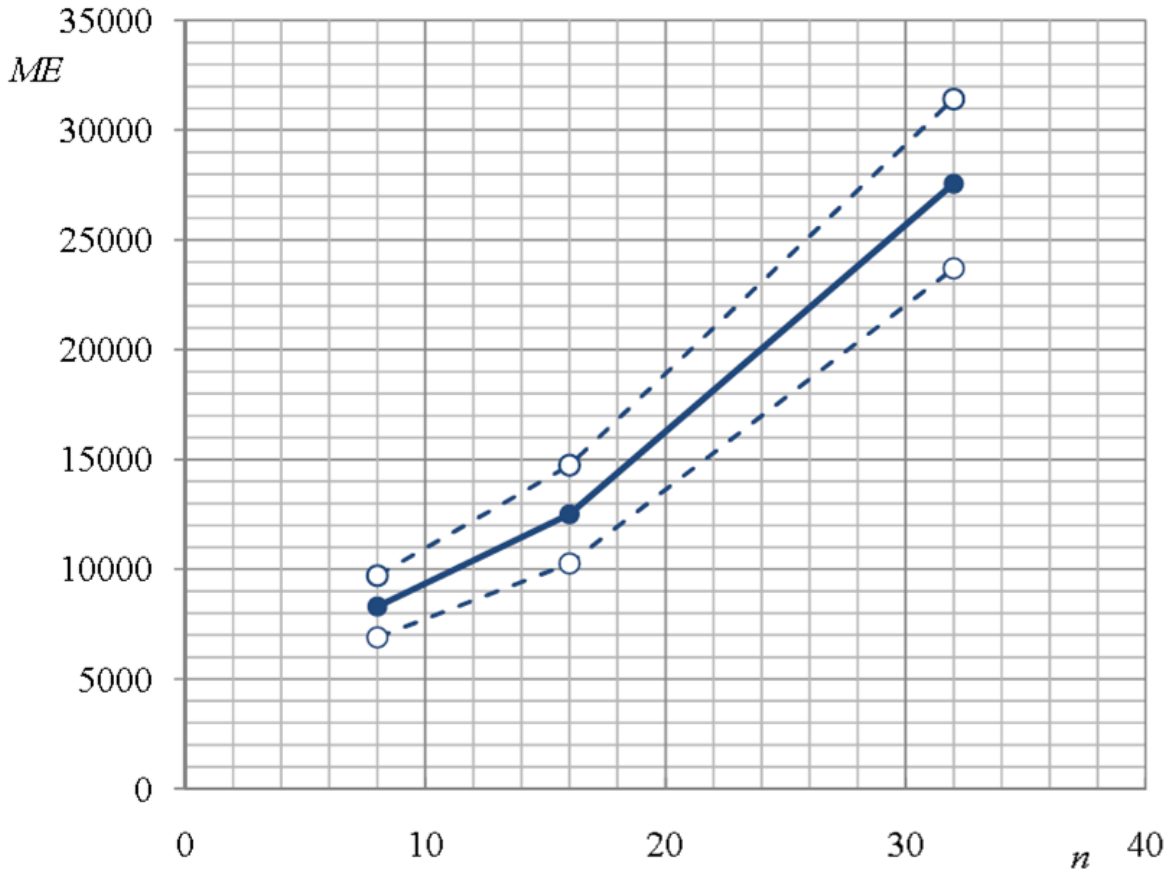

Рисунок 5 – Значения критерия эффективности *ME* в функции параметра n : функция Растригина; $NI_{\max}=50000$; $PAR_{\max}=0{,}65$;*HMS*=25

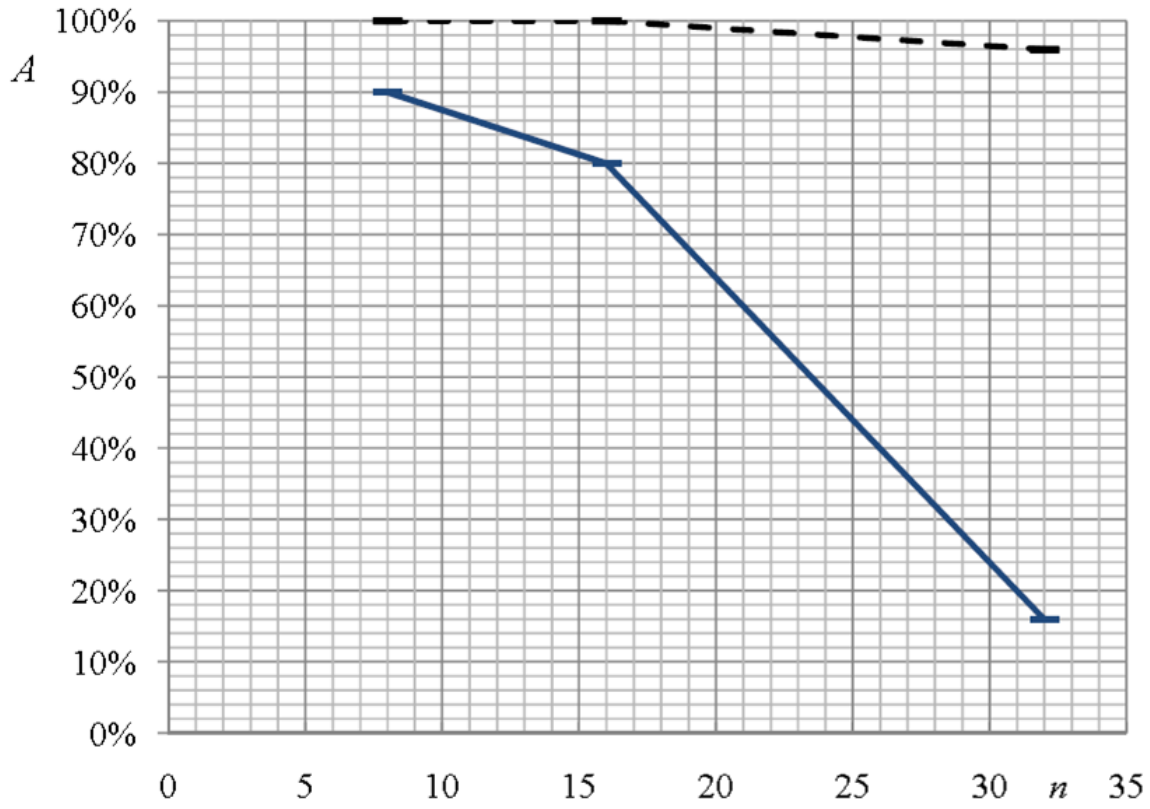

Рисунок 6 – Значения критерия эффективности *A* в функции параметра n : функция Растригина;
$NI_{\max}=50000$; $PAR_{\max}=0{,}65$;*HMS*=25

Результаты исследования показывают, что при увеличении максимально допустимого числа итераций точность алгоритма возрастает. Важно, что даже большое значение величины $NI_{stagn}=1000$ и малое значение величины $\varepsilon=10^{-6}$, не обеспечивают достижение состояния стагнации вычислительного процесса. Из рисунка 6 следует, что при использовании в качестве критерия окончания вычислений условия достижения максимально допустимого числа итераций $NI_{\max}$, вероятность локализации глобального минимума функции Растригина кардинально повышается, так что даже при высокой размерности вектора варьируемых параметров алгоритм обеспечивает высокую надёжность.

4.2. Функция Шекеля

Поиск решения осуществляем в области

$$D=\{X \mid 0\le x_i \le 10, i\in[1:n]\}.$$

Число локальных минимумов функции Шекеля в этой области задано равным 10.Значение функции в точке глобального минимума равно $F(X^*)=-10{,}1$.

Результаты исследования представлены в таблицах 5 – 7 и на иллюстрирующих их рисунках7 – 10.

Таблица 5 – Результаты исследования эффективности алгоритма при варьировании величин $PAR_{\max}$, HMS:функция Шекеля; $NI_{\max}=10000$; $n=8$

HMS	25			50			100		
PAR_{max}	0,4	0,65	0,9	0,4	0,65	0,9	0,4	0,65	0,9
$F(X^*)$	-10,1	-10,1	-10,1	-10,1	-10,1	-10,1	-10,1	-10,1	-10,1
MF	-4,94	-5,39	-5,7	-7,59	-6,5	-8,19	-8,54	-8,03	-8,05
σ_F	3,86	4,14	4,44	2,77	4,05	3,58	2,75	3,01	3,52
MI	7060	6495	6205	9212	7866	7562	9934	9641	8782
σ_{NI}	1792	950	918	756	1178	1216	254	533	895
ME	7085	6520	6230	9262	7916	7612	10034	9741	8882
σ_{NE}	1792	950	918	756	1178	1216	254	533	895
$M[\lvert X-X^*\rvert]$	5,84	4,11	3,47	3,57	3,45	1,2	2,34	2,22	2
A, %	33	43	50	46	53	76	53	66	73

Таблица 6 – Результаты исследования эффективности алгоритма при варьировании величин $PAR_{\max}$, HMS: функция Шекеля; $NI_{\max}=10000$; $n=16$

HMS	25			50			100		
PAR_{max}	0,4	0,65	0,9	0,4	0,65	0,9	0,4	0,65	0,9
$F(X^*)$	-10,1	-10,1	-10,1	-9,72	-9,9	-10,1	-8,4	-9,1	-10,1
MF	-7,6	-9,07	-8,05	-8,45	-9,56	-10,05	-7,42	-8,33	-9,44
σ_F	3,75	2,57	3,37	1,59	0,38	0,02	1	0,75	1,37
MI	10000	9990	9825	10000	10000	9996	10000	10000	10000
σ_{NI}	0	41	388	0	0	22	0	0	0
ME	10025	10015	9850	10050	10050	10046	10100	10100	10100
σ_{NE}	0	41	388	0	0	22	0	0	10000
$M[\lvert X-X^*\rvert]$	1,91	0,85	0,3	0,15	0,07	0,01	0,19	0,14	0,39
A, %	13	63	73	0	0	76	0	0	6

Таблица 7 – Результаты исследования эффективности алгоритма при варьировании величин $PAR_{\max}$, HMS:функция Шекеля; $NI_{\max}=10000$; $n=32$

HMS	25			50			100		
PAR_{max}	0,4	0,65	0,9	0,4	0,65	0,9	0,4	0,65	0,9
$F(X^*)$	-4,25	-4,57	-5,2	-2,32	-3,74	-3,94	-1,19	-1,72	-2,28
MF	-2,81	-3,26	-4,12	-1,69	-2,36	-2,74	-0,97	-1,37	-1,85
σ_F	0,64	0,81	1,08	0,43	0,7	0,66	0,22	0,35	0,43
$MI]$	10000	10000	10000	10000	10000	10000	9955	10000	10000
σ_{NI}	0	0	0	0	0	0	240	0	0
ME	10025	10025	10025	10050	10050	10050	10055	10100	10100
σ_{NE}	0	0	0	0	0	0	240	0	0
$M[\lvert X-X^*\rvert]$	0,52	0,47	0,39	0,73	3,61	0,53	1	0,82	0,69
A, %	0	0	0	0	0	0	0	0	0

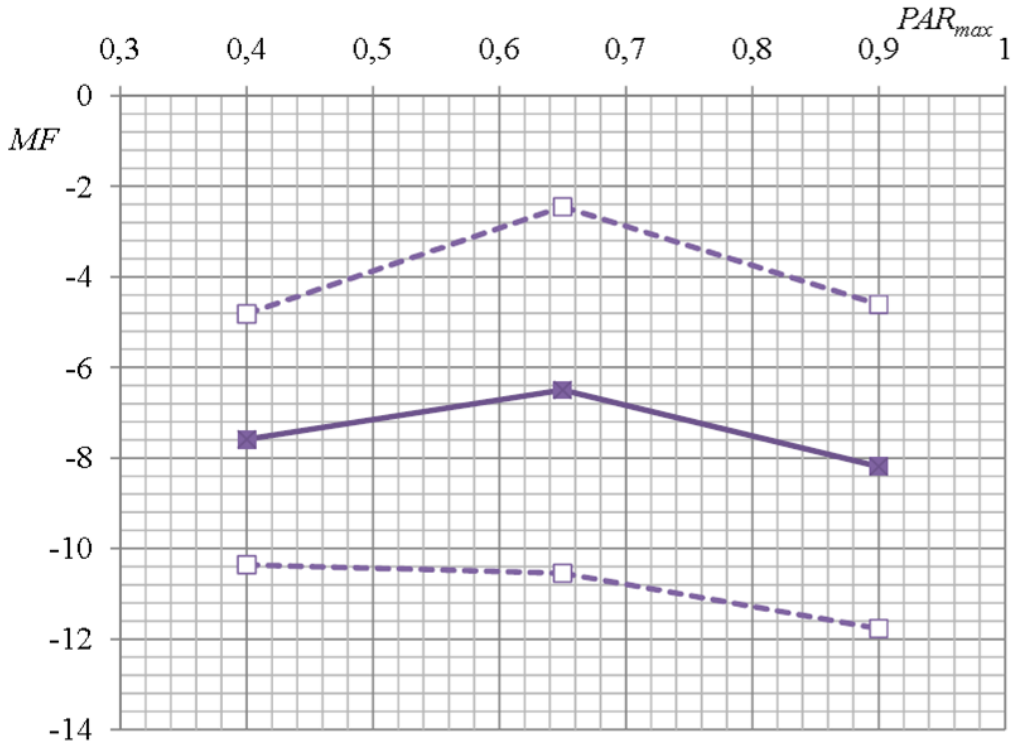

Рисунок 7 – Значения критерия эффективности *MF* в функции параметра $PAR_{\max}$: функция Шекеля;
$NI_{\max} = 10000;\ HMS = 50;\ n = 8$

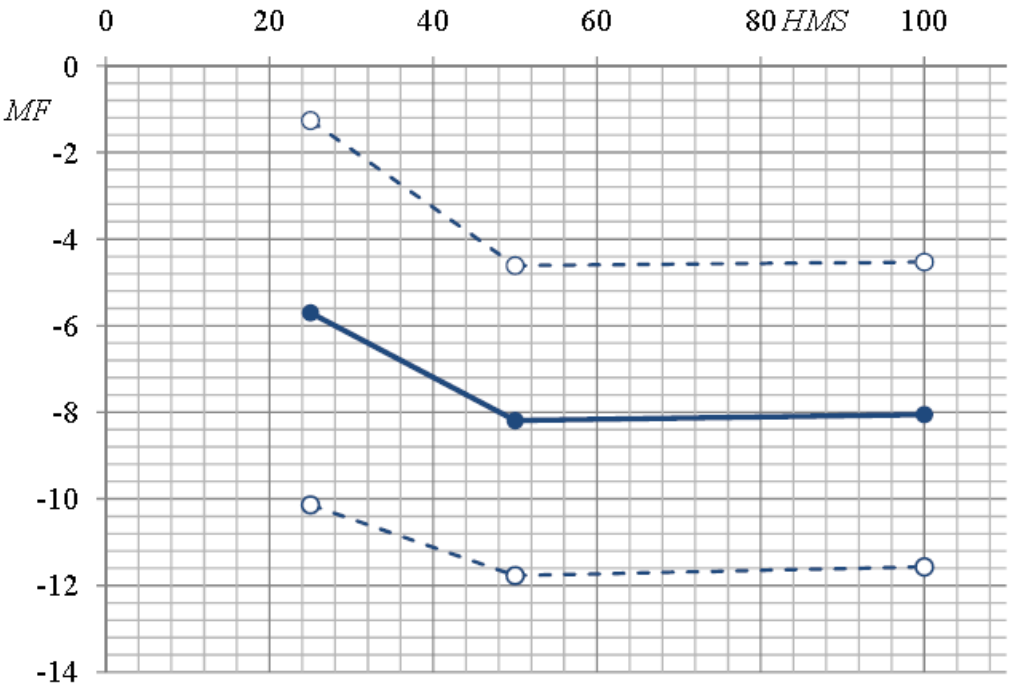

Рисунок 8 – Значения критерия эффективности *MF* в функции параметра HMS : функция Шекеля;
$NI_{\max} = 10000;\ PAR_{\max} = 0{,}9;\ n = 8$

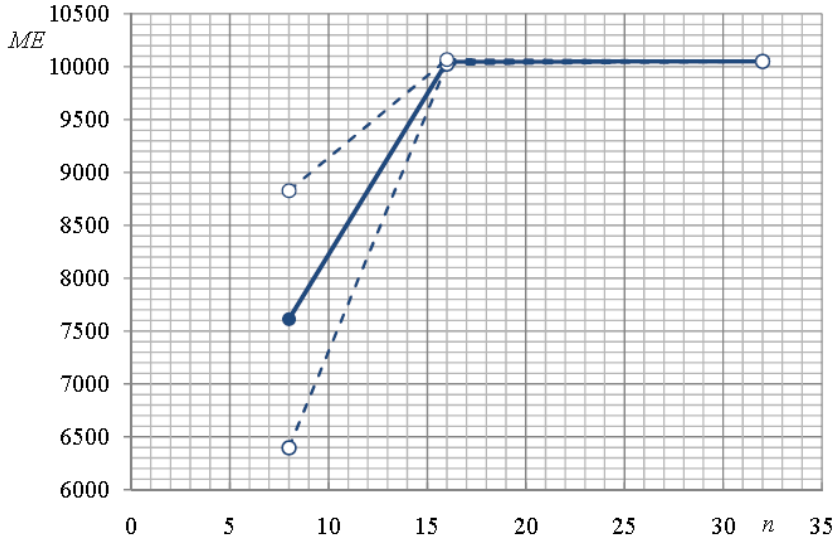

Рисунок 9 – Значения критерия эффективности *ME* в функции параметра n : функция Шекеля;

$$NI_{\max}=10000;\ PAR_{\max}=0{,}9;\ HMS=50$$

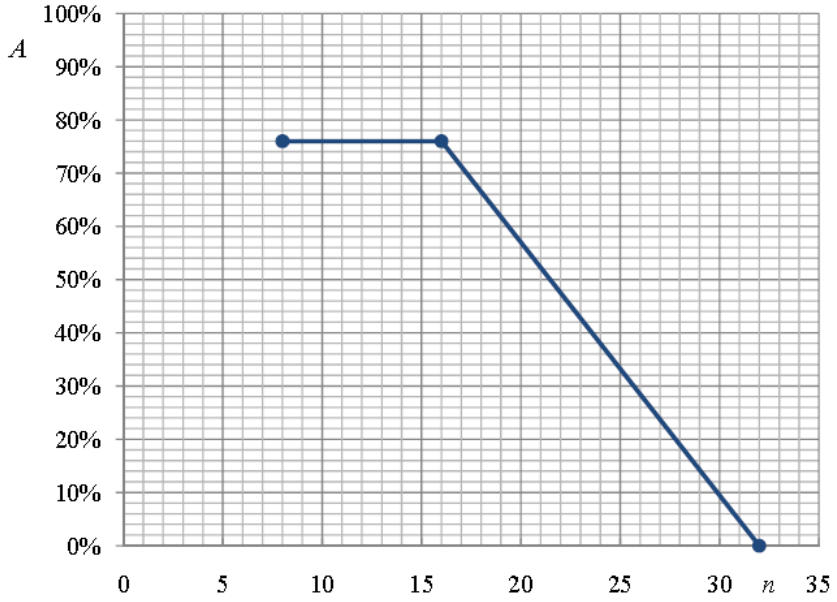

Рисунок 10 – Значения критерия эффективности *A* в функции параметра n : функция Шекеля;

$$NI_{\max}=10000;\ PAR_{\max}=0{,}9;\ HMS=50$$

Из представленных результатов можно сделать выводы, аналогичные выводам предыдущего подраздела. Алгоритм HS^{PSO} с высокой вероятностью локализует глобальный минимум функции Шекеля при размерности пространства поиска *n*=8. При этом наилучший результат по точности достигается при *HMS*=50, $PAR_{\max}=0{,}9$, и среднее число итераций алгоритма остаётся меньшим, чем заданное максимальное число итераций *NImax*=10000. С увеличением размерности пространства поиска вероятность локализации

минимума уменьшается, а среднее число итераций увеличивается до своего максимального значения, то есть состояние стагнации вычислений не достигается.

Результаты экспериментов, соответствующие максимальному числу итераций *NImax*=50000, представлены в таблице 8 и на соответствующих рисунках 11, 12. Сплошная линия на рисунке 12 соответствует случаю, когда в качестве условия окончания итераций используется стагнация вычислений; пунктирная линия - достижение максимального числа итераций $NI_{\max}$.

Таблица 8 – Результаты исследования эффективности алгоритма при варьировании величины n : функция Шекеля; $NI_{\max}=50000$; $PAR_{\max}=0{,}9$; *HMS*=50

n	8	16	32
$F(X^*)$	-10,1	-10,065	-10,034
MF	-8,19	-9,569	-10,031
σ_F	3,58	1,859	0,001
$M[NI]$	7562	14694	35295
σ_{NI}	1216	1516	2736
$M[NE]$	7612	14744	35345
σ_{NE}	1216	1516	2736
$M[\lvert X-X^*\rvert]$	1,2	0,004	0,006
$A, \%$	76	56	30

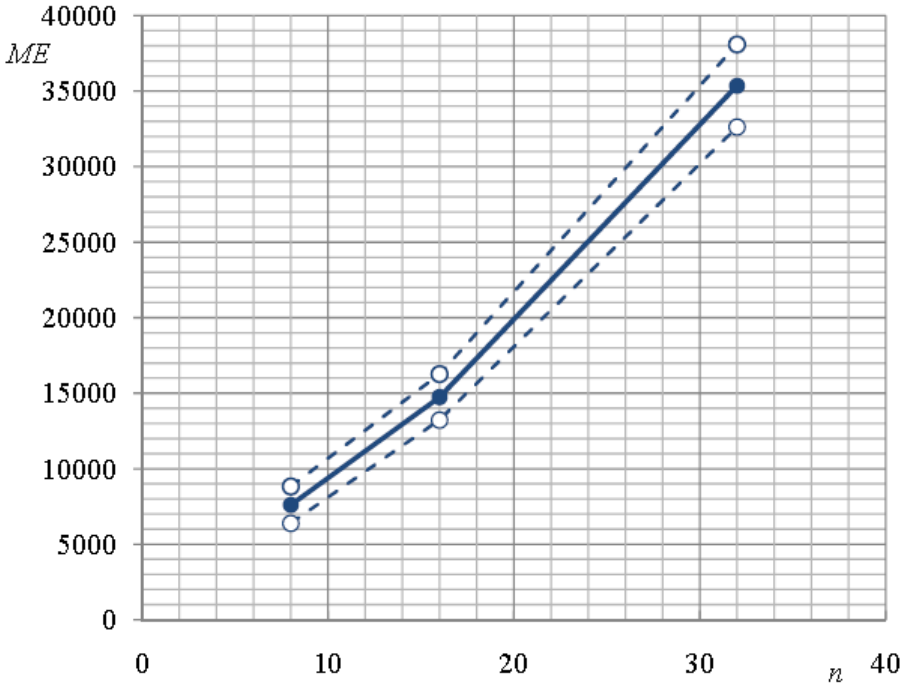

Рисунок 11 – Значения критерия эффективности *ME* в функции параметра n : функция Шекеля; $PAR_{\max}=0{,}9$; $HMS=50$;*NImax*=50000

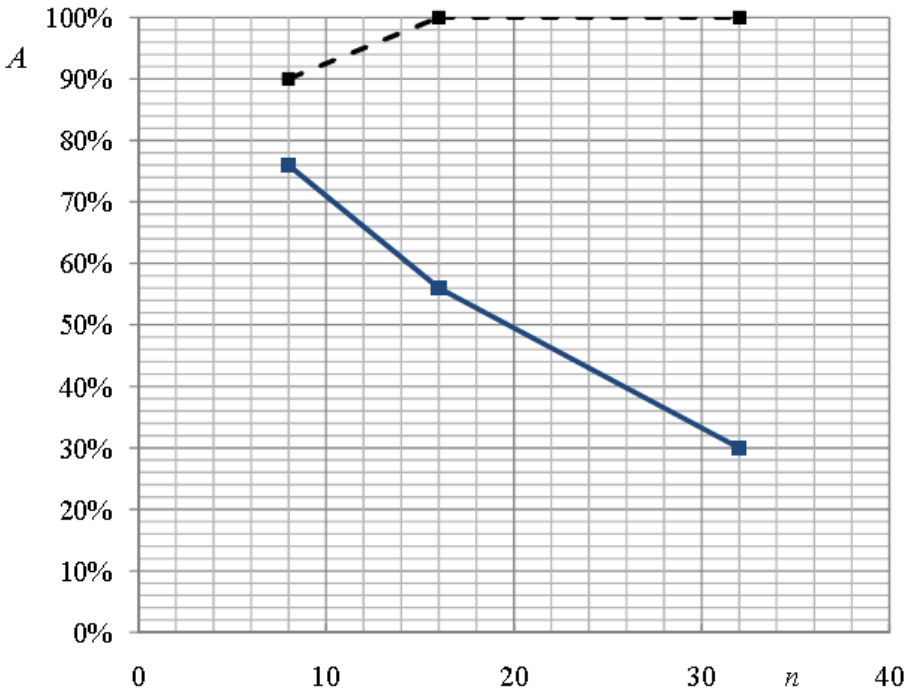

Рисунок 12 – Значения критерия эффективности *A* в функции параметра n : функция Шекеля; $PAR_{\max}=0{,}9$; $HMS=50$; *NImax*=50000

Как и в случае функции Растригина, состояние стагнации вычислительного процесса при высокой размерности вектора *X* не достигается. При использовании в качестве критерия окончания вычислений условия достижения максимально допустимого числа итераций $NI_{\max}$, вероятность локализации глобального минимума рассматриваемой функции Шекеля повышается до 100%.

В целом, результаты исследования показывают, что для задач Растригина и Шекеля наилучшие значения свободных параметров различны. Так для функции Растригина наилучшие результаты были получены при *HMS*=25, $PAR_{\max}=0{,}65$, а для функции Шекеля – при *HMS*=50, $PAR_{\max}=0{,}9$. В обоих случаях результаты, полученные при использовании большого размера памяти гармоний *HMS*=100, не превзошли результатов, полученных сменьшими размерами этой памяти. Следовательно, небольшие размеры памяти гармоний предпочтительнее, поскольку обеспечивают экономию памяти ЭВМ. В алгоритме HS^{PSO} применяется динамическая настройка параметра *PAR*(п. 3). При больших значениях верхней границы $PAR_{\max}$ этого параметра, алгоритм показывает лучшие результаты. При этом также уменьшается число итераций, необходимых для отыскания глобального минимума. Эффект объясняется тем, что в этом случае на ранних этапах поиска любоесозвучие, хранящееся в гармонической памяти, может участвовать всоздании нового решения, что обеспечивает диверсификацию поискового процесса. На более поздних итера-

циях новое созвучие сбольшей вероятностью подражает лучшему из созвучий памяти гармоний, тем самым ускоряя сходимость алгоритма.

5. Параметрическая оптимизация веса ферменной конструкции с помощью алгоритма HS^{PSO}

5.1. Постановка задачи

Рассматриваем задачу размерной оптимизации ферменной конструкции, состоящей из десяти балок, схема которой представлена на рисунке 13 [17].

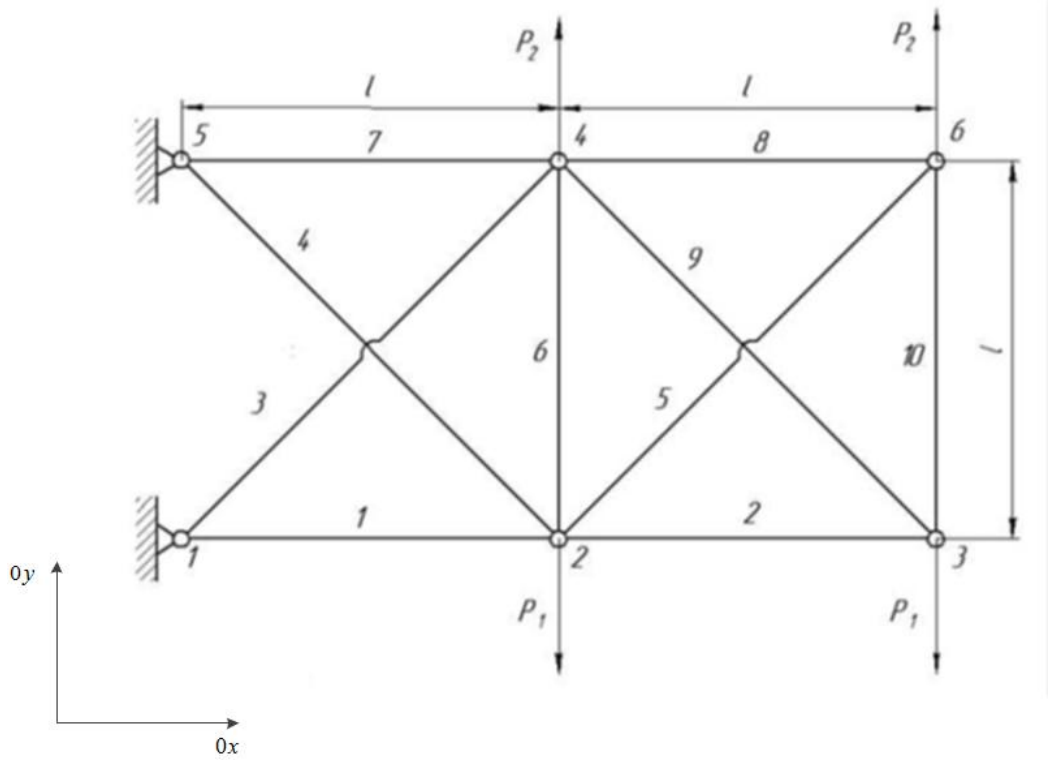

Рисунок 13 – Ферменная конструкция из 10 балок

Целевая функция имеет смысл массы конструкции и определяется выражением

$$f(X) = \rho \sum_{i=1}^{10} A_i \, l_i ,$$

где ρ - плотность материала конструкции; A_i - площадь поперечного сечения i-го элемента конструкции; l_i - длина этого элемента.

Ограничения накладываются на перемещения r_x, r_y узлов по осям $0x$, $0y$, а также на возникающие в балках напряжения σ_k и площади поперченного сечения A_k. Перемещения в узлах 1, 5 должны равняться нулю: $r_{x_1} = 0$; $r_{y_1} = 0$; $r_{x_5} = 0$; $r_{y_5} = 0$. Для остальных узлов имеют место ограничения типа неравенств вида

$$r_{доп} - r_{x_j} > 0, \quad r_{доп} - r_{y_j} > 0; \quad j = 2, 3, 4, 6,$$

где $r_{доп}$ - заданное максимально допустимое смещение узла. Ограничения на напряжения в элементах конструкции и площади поперечных сечений балок имеют, соответственно, вид

$$\sigma_{доп} - \sigma_i > 0,$$

$$A_{\max} - A_i \geq 0,$$

$$A_i - A_{\min} \geq 0.$$

Здесь $\sigma_{доп}$- заданное максимально допустимое напряжение; $A_{\min}, A_{\max}$- заданные допустимые площади поперечных сечений; $i \in [1:10]$.

Принимаем следующие соглашения: допустимая величина напряжений в стержнях $\sigma_{доп}$ =172,434, Мпа; допустимые смещения узлов $r_{доп}$ =0,0508, м; минимальная и максимальная площади поперечных сечений балок равны $A_{\min}$ =0,000645, м^2, $A_{\max}$=0,03, м^2; длины балок *l*=9,1438, м; силы P_1, P_2 равны $P_1 = 667{,}233$, кН, $P_2 = 222{,}411$, кН. Модуль Юнга материала конструкции принимаем равным *E*=70,Гпа; плотность материала - $\rho = 26899$, кг.

Погрешности выполнения ограничений для напряжений ε_σ и перемещений ε_r принимаем равными $\varepsilon_\sigma = 0{,}1$, $\sigma_{\text{доп}}$=17,2434, Мпа и $\varepsilon_r = 0{,}1$, $r_{\text{доп}}$=0,005, м соответственно.

Нормальные напряжения в сечениях балок рассчитываем по формуле

$$\sigma_i = \frac{N_i}{A_i},$$

где N_i - нормальная сила, возникающая в *i*-ой балке под действием внешних сил; $i \in [1:10]$. Полаем, что для удлинений Δl_i балок справедлив закон Гука, так что

$$\Delta l_i = \frac{N_i \; l_i}{E \; A_i}, \quad i \in [1:10].$$

Перемещения каждого из узлов по осям $0x, 0y$ определяем как сумму удлинений балок, соединяющихся в этом узле [17].

Для сведения поставленной задачи условной оптимизации к задаче безусловной оптимизации вида (1), (2), используем метод штрафных функций. В качестве $F(X)$ используем функцию

$$F(X) = f(X) + \sum_{k=1}^{K} \alpha_k \min(0, g_k(X)) + \sum_{l=1}^{L} \beta_l \min(0, h_l(X)),$$

где α_k, β_l - штрафные коэффициенты; K, L – числа ограничений типа неравенств $g_k(X) \geq 0$ и равенств $h_l(X) = 0$ соответственно.

5.2. Вычислительный эксперимент

При решении поставленной задачи использованы следующие значения параметров алгоритма HS^{PSO}: *HMS*=30; *HMCR*=0,95; $PAR_{\min} = 0{,}1;$ $PAR_{\max} = 0{,}9;$ $b_w^{\min} = 0{,}0001;$ $b_w^{\max} = 0{,}001;$ $NI_{\max} = 10000.$

Полученное решение представлено в таблице 9. Соответствующие значения напряжений, смещений узлов и относительных погрешностей выполнения ограничений приведены в таблицах 10 – 14. Сходимость итерационного процесса иллюстрирует рисунок 14.

Таблица 9 - Результаты решения задачи

Величина поперечного сечения балки, м^2.										Масса, кг
A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8	A_9	A_{10}	
0,013	0,0012	0,009	0,003	0,0036	0,0088	0,01	0,0037	0,006	0,001	2014,4

Таблица 10 - Напряжения в балках при найденных поперечных сечениях

Напряжения в балках,МПа.									
σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7	σ_8	σ_9	σ_{10}
73,2	-92,1	170	111	164	-161	-154	-150	-131	171
Относительные погрешности, %									
58	47	1	35	5	7	11	13	24	1

Таблица 11 - Смещения узлов по оси $0x$ при найденных сечениях

Смещения узлов						
№ узла	1	2	3	4	5	6
r_x, м.	0	0,0147	0,0490	-0,0006	0	0,0013
Погрешность, %	0	71	3,5	99	0	97

Таблица 12 - Смещения узлов по оси $0y$ при найденных сечениях

Смещения узлов						
№ узла	1	2	3	4	5	6
r_y, м.	0	-0,0150	-0,0053	0,0183	0	-0,0008
Погрешность, %	0	71	90	64	0	98

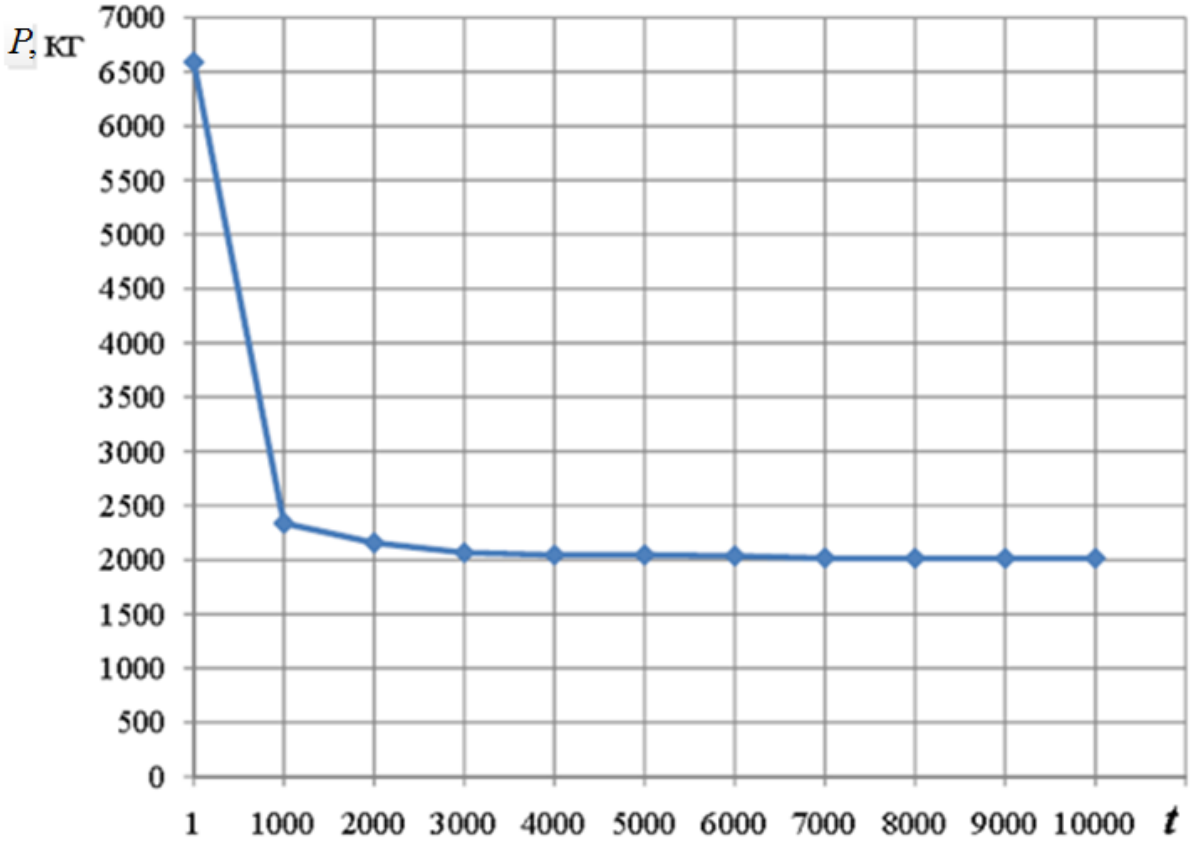

Рисунок 14–Сходимость итерационного процесса: *P* - масса конструкции

Представленные результаты показывают, что в процессе оптимизации масса конструкции уменьшилась с 6500, кг до 2014,4 кг. Напряжения, возникающие в элементах конструкции при найденных площадях поперечных сечений балок, а также смещения узлов не превосходят своих допустимых значений.

Рассматриваемая задача оптимизации веса ферменной конструкции решалась различными методами [18]. Представленные результаты решения задачи показывают, что алгоритм HS^{PSO} даёт лучший результат по сравнению с результатами, полученными в предыдущих работах. Отметим, что решение, полученное алгоритмом HS^{PSO}, также превосходит решение, которое обеспечивает канонический алгоритм поиска гармонии [18].

Заключение

В работе представлена авторская гибридизация алгоритмов поиска гармонии и роя частиц, названная HS^{PSO}. Выполнена программная реализация алгоритма и исследование его эффективности на двух тестовых многомерных многоэкстремальных задачах глобальной оптимизации, а также на практически значимой задаче размерной оптимизации ферменной конструкции. Результаты исследования показали эффективность принятых алгоритмических и программных решений. Так, в результате решения последней задачи были получены результаты, лучшие аналогичных результатов, полученных другими авторами.

В развитие работы авторы планируют исследование эффективности алгоритма HS^{PSO} на более широком наборе тестовых задач, параллельную реализацию алгоритма.

Список литературы

1. Wang X. Hybrid nature-inspired computation method for optimization. Doct. Diss. Helsinki University of Technology, TKK Dissertations, 2009. 161 p.
2. El-Abd, Kamel M. A taxonomy of cooperative search algorithms // In: Hybrid Metaheuristics. Proc. Second International Workshop. Springer Berlin Heidelberg, 2005. P. 32-41. (Ser. Lecture Notes in Computer Science; vol. 3636). DOI: 10.1007/11546245_4
3. Raidl G.R. A Unified View on Hybrid Metaheuristics // In: Hybrid Metaheuristics. Proc. Third International Workshop. Springer Berlin Heidelberg, 2006. P. 1-12. (Ser. Lecture Notes in Computer Science; vol. 4030). DOI: 10.1007/11890584_1
4. Geem Z.W., Kim J.H., Loganathan G.V. A new heuristic optimization algorithm: harmony search // Simulation. 2001. Vol. 76, no. 2. P. 60-68.
5. Mahdavi M., Fesanghary M., Damangir E. An improved harmony search algorithm for solving optimization problems // Applied Mathematics and Computation. 2007. Vol. 188, no. 2. P. 1567-1579. DOI: 10.1016/j.amc.2006.11.033
6. Omran M.G.H., Mahdavi M. Global-best harmony search // Applied Mathematics and Computation. 2008. Vol. 198, no. 2. P. 643-656. DOI: 10.1016/j.amc.2007.09.004
7. Карпенко А.П., Селиверстов Е.Ю. Обзор методов роя частиц (PSO) для задачи глобальной оптимизации» // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2009. № 3. Режим доступа: http://technomag.edu.ru/doc/116072.html (дата обращения 01.06.2014).
8. Lin Y., Chang-Ming X. A new improved harmony search algorithm for continuous optimization problem // 2011 International Conference on Computer Science and Network Technology (ICCSNT). Vol. 2. IEEE, 2011. P. 686-690. DOI: 10.1109/ICCSNT.2011.6182059
9. Wang Chia-Ming, Huang Yin-Fu. Self-adaptive harmony search algorithm for optimization // Expert Systems with Applications. 2010. Vol. 37, no. 4. P. 2826-2837. DOI: 10.1016/j.eswa.2009.09.008

10. Chen Jing, Man Hong-Fang, Wang Ya-Min. Novel Self-adaptive Harmony Search Algorithm for continuous optimization problems // 2011 30th Chinese Control Conference (CCC). IEEE, 2011. P. 5452-5456.
11. Fesanghary M., Mahdavi M., Minary-Jolandan M., Alizadeh Y. Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems// Computer Methods in Applied Mechanics and Engineering. 2008. Vol. 197, no. 33-40. P. 3080-3091. DOI: 10.1016/j.cma.2008.02.006
12. Taherinejad N. Highly reliable harmony search algorithm // ECCTD 2009. European Conference Circuit Theory and Design. IEEE, 2009. P. 818-822. DOI: 10.1109/ECCTD.2009.5275109
13. Gao X.Z., Wang X., Ovaska S.J., He Xu. A modified harmony search method in constrained optimization // International Journal of Innovative Computing, Information and Control. 2009. Vol. 6, no. 9. P. 4235-4247.
14. Xin-She Yang. Harmony Search as a Metaheuristic Algorithm // In: Music-Inspired Harmony Search Algorithm: Theory and Applications / Z.W. Geem, ed. Springer Berlin Heidelberg, 2009, pp. 1-14. (Ser. Studies in Computational Intelligence; vol. 191). DOI: 10.1007/978-3-642-00185-7_1
15. Kennedy J., Eberhart R. Particle swarm optimization // Proceedings of IEEE International Conference on Neural Networks. Vol. 4. IEEE, 1995. P. 1942-1948. DOI: 10.1109/ICNN.1995.488968
16. Tang K., Yao X., Suganthan P.N., MacNish C., Chen Y.P., Chen C.M., Yang Z. Benchmark Functions for the CEC'2008 Special Session and Competition on Large Scale Global Optimization. Technical Report. Nature Inspired Computation and Applications Laboratory, USTC, China, 2007. Available at: http://nical.ustc.edu.cn/cec08ss.php , accessed 01.06.2014.
17. Hultman M. Weight Optimization of Steel Trusses by a Genetic Algorithm–Size, Shape and Topology Optimization According to Eurocode. Report TVBK. Department of Structural Engineering Lund Institute of Technology, Lund University, 2010. 61 p.
18. Lee K.S., Geem Z.W. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice // Computer Methods in Applied Mechanics and Engineering. 2005. Vol. 194, iss. 36-38. P. 3902-3933. DOI: 10.1016/j.cma.2004.09.007

SCIENTIFIC PERIODICAL OF THE BAUMAN MSTU

SCIENCE and EDUCATION

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

electronic scientific and technical journal

Global Optimization Based on the Hybridization of Harmony Search and Particle Swarm Optimization Methods

07, July 2014

DOI: 10.7463/0714.0717653

A.P. Karpenko[1,a], T.V. Pechenina[2], V.A. Bulanov[1]

[1]Bauman Moscow State Technical University, 105005, Moscow, Russian Federation

[2]ASCON 199155, St-Petersburg, Russian Federation

[a]pkarpenko@mail.ru

Keywords: global optimization, PSO, hybridization

We consider a class of stochastic search algorithms of global optimization which in various publications are called behavioural, intellectual, metaheuristic, inspired by the nature, swarm, multi-agent, population, etc. We use the last term.

Experience in using the population algorithms to solve challenges of global optimization shows that application of one such algorithm may not always effective. Therefore now great attention is paid to hybridization of population algorithms of global optimization. Hybrid algorithms unite various algorithms or identical algorithms, but with various values of free parameters. Thus efficiency of one algorithm can compensate weakness of another.

The purposes of the work are development of hybrid algorithm of global optimization based on known algorithms of harmony search (HS) and swarm of particles (PSO), software implementation of algorithm, study of its efficiency using a number of known benchmark problems, and a problem of dimensional optimization of truss structure.

We set a problem of global optimization, consider basic algorithms of HS and PSO, give a flow chart of the offered hybrid algorithm called HS^{PSO}, present results of computing experiments with developed algorithm and software, formulate main results of work and prospects of its development.

References

1. Wang X. *Hybrid nature-inspired computation method for optimization. Doctoral Dissertation*. Helsinki University of Technology, TKK Dissertations, 2009, 161 p.

2. El-Abd, Kamel M. A taxonomy of cooperative search algorithms. In: *Hybrid Metaheuristics. Proc. Second International Workshop*. Springer Berlin Heidelberg, 2005, pp. 32-41. (Ser. *Lecture Notes in Computer Science*; vol. 3636). DOI: 10.1007/11546245_4
3. Raidl G.R. A Unified View on Hybrid Metaheuristics. In: *Hybrid Metaheuristics. Proc. Third International Workshop*. Springer Berlin Heidelberg, 2006, pp. 1-12. (Ser. *Lecture Notes in Computer Science*; vol. 4030). DOI: 10.1007/11890584_1
4. Geem Z.W., Kim J.H., Loganathan G.V. A new heuristic optimization algorithm: harmony search. *Simulation*, 2001, vol.76, no. 2, pp. 60-68.
5. Mahdavi M., Fesanghary M., Damangir E. An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation*, 2007, vol. 188, iss. 2, pp. 1567-1579. DOI: 10.1016/j.amc.2006.11.033
6. Omran M.G.H., Mahdavi M. Global-best harmony search. *Applied Mathematics and Computation*, 2008, vol. 198, no. 2, pp. 643-656. DOI: 10.1016/j.amc.2007.09.004
7. Karpenko A.P., Seliverstov E.Yu. Review of the particle swarm optimization method (PSO) for a global optimization problem. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2009, no. 3. Available at: http://technomag.edu.ru/doc/116072.html , accessed 01.06.2014. (in Russian).
8. Lin Y., Chang-Ming X. A new improved harmony search algorithm for continuous optimization problem. *2011 International Conference on Computer Science and Network Technology (ICCSNT). Vol. 2*. IEEE, 2011, pp. 686-690. DOI: 10.1109/ICCSNT.2011.6182059
9. Wang Chia-Ming, Huang Yin-Fu. Self-adaptive harmony search algorithm for optimization. *Expert Systems with Applications*, 2010, vol. 37, no. 4, pp. 2826-2837. DOI: 10.1016/j.eswa.2009.09.008
10. Chen Jing, Man Hong-Fang, Wang Ya-Min. Novel Self-adaptive Harmony Search Algorithm for continuous optimization problems. *2011 30th Chinese Control Conference (CCC)*. IEEE, 2011, pp. 5452-5456.
11. Fesanghary M., Mahdavi M., Minary-Jolandan M., Alizadeh Y. Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems. *Computer Methods in Applied Mechanics and Engineering*, 2008, vol. 197, iss. 33-40, pp. 3080-3091. DOI: 10.1016/j.cma.2008.02.006
12. Taherinejad N. Highly reliable harmony search algorithm. *ECCTD 2009. European Conference Circuit Theory and Design*. IEEE, 2009, pp. 818-822. DOI: 10.1109/ECCTD.2009.5275109

13. Gao X.Z., Wang X., Ovaska S.J., He Xu. A modified harmony search method in constrained optimization. *International Journal of Innovative Computing, Information and Control*, 2009, vol. 6, no.9, pp. 4235-4247.
14. Xin-She Yang. Harmony Search as a Metaheuristic Algorithm. In: Geem Z.W., ed. *Music-Inspired Harmony Search Algorithm: Theory and Applications*. Springer Berlin Heidelberg, 2009, pp. 1-14. (Ser. *Studies in Computational Intelligence*; vol. 191). DOI: 10.1007/978-3-642-00185-7_1
15. Kennedy J., Eberhart R. Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks. Vol. 4*. IEEE, 1995, pp. 1942-1948. DOI: 10.1109/ICNN.1995.488968
16. Tang K., Yao X., Suganthan P.N., MacNish C., Chen Y.P., Chen C.M., Yang Z. *Benchmark Functions for the CEC'2008 Special Session and Competition on Large Scale Global Optimization*. Technical Report. Nature Inspired Computation and Applications Laboratory, USTC, China, 2007. Available at: http://nical.ustc.edu.cn/cec08ss.php, accessed 01.06.2014.
17. Hultman M. *Weight Optimization of Steel Trusses by a Genetic Algorithm–Size, Shape and Topology Optimization According to Eurocode*. Report TVBK, Department of Structural Engineering Lund Institute of Technology, Lund University, 2010. 61 p.
18. Lee K.S., Geem Z.W. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering*, 2005, vol. 194, no. 36-38, pp. 3902-3933. DOI: 10.1016/j.cma.2004.09.007