Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего профессионального образования
«Уральский федеральный университет имени первого Президента России Б.Н. Ельцина»

Институт математики и компьютерных наук
Кафедра алгебры и дискретной математики

# Система поддержки алгоритмов коллективного разума

Допущен к защите

_____

"__" _____ 2012

Квалификационная работа на степень магистра наук
по направлению «Математика. Компьютерные науки.»
студента гр. МГКН – 201
Кононучука Дмитрий Олеговича

Научный руководитель
Окуловский Юрий Сергеевич
заведующий лабораторией РВИИМАП, к.ф.-м.н.

Екатеринбург
2012

# РЕФЕРАТ

Конончук Д.О. СИСТЕМА ПОДДЕРЖКИ АЛГОРИТМОВ КОЛЛЕКТИВНОГО РАЗУМА, квалификационная работа на степень магистра наук: стр. ТУДУ, библ. ТУДУ назв.

Ключевые слова: алгоритмы коллективного разума, система поддержки, общие свойства, программная модель, распараллеливание, визуализация, C#.

Рассматривается множество алгоритмов коллективного разума, выделяются их общие свойства. На их основе строится программная модель, способная эмулировать любой алгоритм коллективного разума. Рассматриваются расширения модели, направленные на улучшение удобства использования. Рассматриваются основные аспекты функционирования этой модели.

# Содержание

# Перечень принятых в работе сокращений

| | |
|---|---|
| GSA | Gravitational Search Algorithm |
| LEM | Learnable Evolution Model |
| PSO | Particle Swarm Optimization |
| SDS | Stochastic Diffusion Search |
| АКР | Алгоритмы Коллективного Разума |
| АЭМ | Алгоритмы Эволюционного Моделирования |
| ВИ | Вычислительный Интеллект |
| ГА | Генетические Алгоритмы |
| ИИ | Искусственный Интеллект |
| МАА | Многоагентные Алгоритмы |

# Введение

Алгоритмы коллективного разума являются в настоящее время одной из самых динамично развивающихся отраслей алгоритмов искусственного интеллекта. Они основаны на т.н. «интеллекте толпы» — явлении, при котором система из множества слабо интеллектуальных, равноправных и децентрализованных сущностей проявляет свойства интеллектуальности.

# Глава 1

# Теоретические основы

## 1.1. Понятия искусственного и вычислительного интеллекта

Под искусственным интеллектом понимается довольно обширный класс инструментов и технологий, границы которого во многом определяются историческими либо практическими причинами, а не общепринятыми формальными критериями, которых на данный момент не существует.

Наиболее распространенным подходом к определению интеллектуальности, а, следовательно, и ИИ , является агент-ориентированный подход, изложенный в одной из основополагающих работ в области — [1]. Ключевое понятие для этого подхода — агент. Это некоторая сущность, которая способна воспринимать окружающую среду посредством датчиков и влиять на нее посредством исполнительных механизмов. Поведение агента считается интеллектуальным, если оно рационально в т.ч. и при меняющихся условиях среды. Эта модель требует наличия свойства адаптивности у алгоритмов ИИ.

Тем не менее, существуют другие подходы, например логический [2] или основанный на поиске [3], ярким примером которого является алгоритм A-star [4], который в настоящее время не относится к интеллектуальным.

В своей работе я исследовал лишь методы, относящиеся к области вычислительного интеллекта [5], интеллектуальность которых общепризнанна. Однако, также как и для интеллекта искусственного, для вычислительного не было построено четкого определения. Обширный обзор разработанных определений можно найти в [6]. В этой-же работе автор приводит собственное, пожалуй, лучшее из представленных в литературе, хотя и излишне общее определение ВИ: «Вычислительный интеллект — это область компьютерных наук,

изучающая решение задач, для которых не существует эффективного алгоритмического решения».

Во многих работах, например [7], ВИ определяется, как совокупность различных технологий. Подобный остенсивный [8] способ определения понятия обладает очевидными недостатками, поскольку не указывает на общие свойства этих технологий и не является расширяемым. В частности, в книге [5] авторы относят к ВИ 4 типа алгоритмов, а 2 года спустя в [7] — уже 7. Работы, использующие для определения ВИ именно такой — остенсивный — подход, обычно содержат довольно подробную классификацию существующих методов.

Согласно этим классификациям алгоритмы ВИ разбиваются на три подкласса: нейронные сети [9], нечеткое управление [10] и эволюционное моделирование [11].

## 1.2. Алгоритмы эволюционного моделирования

Перейдем к более подробному рассмотрению третьего класса алгоритмов — алгоритмов эволюционного моделирования. Эти алгоритмы предназначены для решения задач комбинаторной оптимизации.

Исторически, первыми представителями этого класса были генетические алгоритмы, предложенные Лоуренсом Фогелем в работе [12]. В их основу положен принцип моделирования генетической эволюции в смысле синтетической теории эволюции [13]. В самом общем виде генетический алгоритм формулируется следующим образом:

1. Алгоритм оперирует особями — некоторым представлением решений задачи. Каждая особь однозначно определяет допустимое решение. Для каждой особи определена ее функция приспособленности — характеристика оптимальности решения ею определяемого.

2. Алгоритм поддерживает пул особей, называемый популяцией, который характеризует его текущее состояние.

3. При инициализации алгоритма некоторым образом создается начальная популяция.

4. На каждом шаге алгоритма производятся три процесса:

    а. Мутация: некоторым случайным образом выбираются несколько особей и для каждой из них в популяцию добавляется особь (или несколько особей), которая является ее некоторой случайной модификацией.

    б. Скрещивание: некоторым случайным (либо нет) образом выбираются пары особей. Для каждой из них некоторым случайным способом создаются и затем добавляются в популяцию некоторое количество производных особей.

    в. Отбор: из популяции удаляются особи имеющие плохую функцию приспособления либо слишком продолжительное время жизни (возможно сочетание критериев).

5. Шаг алгоритма повторяется до тех пор пока не выполняться определенные условия. Например, пока в популяции не перестанут происходить существенные изменения.

6. Результат работы алгоритма — особь с лучшей функцией приспособленности.

Для ГА известны способы доказательства корректности и сходимости алгоритмов [14], а также определения их эффективности [15].

Другим АЭМ, зачастую противопоставляемым генетическим, является алгоритм Learnable Evolution Model [16]. Он также работает с представле-

нием решений в виде особей с определенной функцией приспособленности и представлением текущего состояния в виде популяции. Однако в его основе лежит другая теория эволюции — теория направленной эволюции (номогенез) [17]. В самом общем виде алгоритм формулируется так:

1. Некоторым способом создается начальная популяция.
2. На каждом шаге алгоритма выполняются следующие действия:
    а. Из популяции выделяются две группы особей: «хорошие» и «плохие» — особи соответственно с высоким и низким значением функции приспособленности.
    б. С помощью методов машинного обучения строятся гипотезы — описания отличительных черт «хороших» особей, которые не наблюдаются у «плохих». Возможно также построение обратных гипотез — черт присутствующих у «плохих», но не у «хороших» особей.
    в. Генерируются новые особи, которые удовлетворяют гипотезам и не удовлетворяют обратным гипотезам.
    г. Эти особи тем или иным способом добавляются в популяцию. Например заменяя всех особей не являющихся «хорошими».
3. Шаг алгоритма повторяется до тех пор пока не выполняются определенные условия.
4. Результат работы алгоритма — особь с лучшей функцией приспособленности.

В качестве примера алгоритма АЭМ не оперирующего понятием «эволюция», можно привести алгоритм Stochastic Diffusion Search [18]. В нем, как и

в предыдущих алгоритмах, вводятся понятия особи и популяции. Однако, в SDS особь не отождествляется с решением, а является независимым объектом, которому в каждый момент времени сопостовляют решение, т.о. может расссматриваться как частично определенная функция $a_i : T \to S$, $i \in \{1 \dots n\}$. Алгоритм формулируется следующим образом:

1. Аналогично предыдущим алгоритмам, вводится характеристика качества решения — $f : S \to \mathbb{R}$.
2. Некоторым способом особям сопоставляются начальные решения $a_i(0)$.
3. На каждом шаге алгоритма выполняются следующие действия:
   а. Для всех особей вычисляется качество их текущего решения: $f(a_i(t))$.
   б. Особи разделяются на две группы: «хорошие» и «плохие» — особи соответственно с высоким и низким качеством решения.
   в. Для каждой «плохой» особи $i$:
      i. Случайно выбирается особь $j \in \{1 \dots n\} \setminus \{i\}$.
      ii. Если особь $j$ «хорошая», то $a_i(t+1) \leftarrow a_j(t)$,
      iii. иначе $a_i(t+1) \leftarrow \mathrm{Rand}(S)$.
   г. Для всех «хороших» особей $a_i(t+1) \leftarrow a_i(t)$.
4. Шаг алгоритма повторяется до тех пор пока всем особям не будет сопоставленно малое число (или ровно одно) решений:
$$\left\| \bigcup_{i \in \{1 \dots n\}} \{a_i(t)\} \right\| \to 1$$
5. Результат работы алгоритма — решение, которое сопоставленно наибольшему числу особей.

Более подробный анализ SDS с доказательством некоторых его свойств, включая временную сложность $o(n)$, приведены в [19].

На основании вышеперечисленных алгоритмов можно выделить общие черты, присущие всем АЭМ:

1. Алгоритм решает задачу оптимизации функции (называемой также функцией качества решения, функцией приспособленности) $f : S \to \mathbb{R}$, определенной на множестве значений (называемом также множеством решений) $S$. Чаще всего множество $S$ велико, а функция $f$ является трудно вычислимой.

2. Состояние алгоритма представлено некоторым множеством (популяцией) некоторых объектов (особей) и, возможно, производными данными. Состояние алгоритма однозначно определяет набор решений — выбранных значений из множества $S$.

3. Алгоритм итеративный. Цель каждой итерации — увеличить максимальное значение функции приспособленности для решений, определенных текущим сосотоянием. Фактически провести эволюцию популяции.

4. Начальное состояние дается алгоритму извне — обычно генерируется случайно.

5. На каждой итерации алгоритм многократно (например для каждой особи, либо фиксированное число раз) выполняет некоторые действия. Причем действия эти независимые и распараллеливаемые. В некоторых алгоритмах также выделяются стадии пред- и пост-процессинга.

6. Алгоритм стохастический.

Особо отметим, что вышеприведенные алгоритмы формируют более слабые ограничения для пунктов (2) и (5). Основываясь лишь на них, мы можем

утчерждать, что во-первых «Особь однозначно определяет одно решение», и во-вторых, «На каждой итерации алгоритм выполняет некоторые действия для каждой особи». Однако существуют примеры АЭМ, которые не вписываются в эти, более узкие, рамки. Например, расширение пункта (2) необходимо для включения в класс АЭМ алгоритмов муравейника (англ. Ant Colony Optimisation) [20] (решение в них кодируется не особью, а производными данными), а пункта (5) — для включения алгоритма гармонического поиска [21].

Перечисленные свойства не являются определяющими для класса АЭМ, однако, они наблюдаются у всех общеизвестных алгоритмов.

Стоит заметить, что эти свойства являются весьма существенными с точки зрения программных реализаций алгоритмов. Они определяют задачи, которые должны быть решены в каждой из них. Это, в первую очередь, задачи создания удобной программной абстракции, выполнения сервисных операций с популяцией, распараллеливания и распределения по данным вычислений на каждой итерации. Их решение довольно трудоемко, что приводит к мысли о создании специализированных библиотек поддержки. Подобные библиотеки существуют (в качестве примера можно привести EO Evolutionary Computation Framework [22] и ECF [23]), однако, ввиду чрезмерной общности данных свойств и их неестественности, такие библиотеки недостаточно удобны в использовании.

Неестественность общих свойств АЭМ во многом объясняется тем, что данный класс состоит из двух подклассов, представители которых существенно различны. Это подклассы алгоритмов эволюционных вычислений [24] и коллективного разума [25]. У каждого из них имеется гораздо больше общих свойств, и эти свойства гораздо естественнее, благодаря чему создание библиотеки поддержки становится более целесообразным.

В своей работе я сфокусировался на разработке подобной библиотеки для класса алгоритмов коллективного разума.

## 1.3. Алгоритмы коллективного разума

Исторически первым представителем класса алгоритмов коллективного разума (англ. Swarm Intelligence Algorithms) является алгоритм Particle Swarm Optimization [26] опубликованый в 1942 г. Однако, непосредственно термин АКР а также ключевые свойства данного класса алгоритмов впервые были сформулированы только в 1989 г. в статье [27].

Основополагающей работой для всего класса класса АКР, во многом определившей его вид, считается [20]. В ней описывается уже упоминавшийся ранее алгоритм муравейника для решения задачи коммивояжёра [28]. Позже, благодаря своей обобщенности, этот алгоритм был адаптирован для решения многих других задач, примеры которых можно найти в [29]. Рассмотрим алгоритм муравейника в формулировке для решения задачи поиска кратчайшего пути на графе.

Алгоритм муравейника, как и другие представители класса АЭМ, оперирует популяцией особей («агентов» в терминологии АКР; «муравьев» в терминологии данного алгоритма) и дополнительными данными: «феромонами» ($\tau_{i,j}(t)$), позволяющими ассоциировать с ребрами графа значения, указывающие на историю его использования муравьями. Алгоритм формулируется следующим образом:

1. Вводим дополнительную эвристику $\eta_{i,j}(t)$.
2. Некоторым способом изначально располагаем муравьев в вершинах графа.
3. На каждом шаге алгоритма для каждого муравья $\text{ant}_k$, $k \in \{1 \ldots n\}$ расположенного в вершине $i$ производятся следующие действия:
    a. Вычисляется вероятность перехода муравья в каждую из допусти-

мых вершин. Общая формула:

$$\forall j \in N : \; p_{i,j}^{k}(t) = \frac{\left(\tau_{i,j}(t)\right)^{\alpha}\left(\eta_{i,j}(t)\right)^{\beta}}{\sum\limits_{l \in N}\left(\tau_{i,l}(t)\right)^{\alpha}\left(\eta_{i,l}(t)\right)^{\beta}}$$

б. Выполняется случайный переход муравья по ребру $(i,\ j)$.

в. Фиксированным для алгоритма способом определяется функция $\Delta\tau_{i,j}^{k}(t)$ и $\forall l \neq j$ полагаем $\Delta\tau_{i,l}^{k}(t) = 0$

4. Для всех ребер осуществляется пересчет соответствующих феромонов:

$$\tau_{i,j}(t+1) = \rho\tau_{i,j}(t) + \sum_{k=1}^{n}\Delta\tau_{i,j}^{k}(t), \text{ где } 0 \leqslant \rho < 1$$

5. Шаг алгоритма повторяется до тех пор пока в графе не образуется путь помеченный «высоким уровнем» феромонов: $\forall(i,\ j) \in \text{Path} : \tau_{i,j}(t) \geqslant \Gamma$.

6. Результат работы алгоритма — путь помеченный высоким уровнем феромонов.

## 1.4. Многоагентные алгоритмы

Алгоритм муравейника является не только характерным примером АКР, но и другого, чрезвычайно похожего на первый взгляд, класса алгоритмов — многоагентных алгоритмов. Границу между этими двумя классами не проводят в большей части литературы.

Дело в том, что МАА и АКР оба базируются на агентах, однако трактуют это понятие по разному. Как уже было указано выше, для АКР агент — то же самое, что и особь для АЭМ, т.е. сущность, которой он оперирует и которая представляет решение. В то время как в МАА под агентом подразумевается агент в терминах агент-ориентированного подхода к ИИ [1], т.е. независимая,

решающая задачу сущность, воспринимающая окружающую среду и взаимодействующая с ней.

Поскольку сам МАА также является агентом в этих терминах, то естественно, что необходимо вводить специальные ограничения, делающие задачи решаемые агентами разных уровней (МАА в целом $\longleftrightarrow$ агент в МАА) существенно различными. Типичным ограничением является возможность работы лишь в подмножестве пространства исходной задачи.

Из определений очевидно, что МАА являются подмножеством АКР: с точностью до формулировок МАА являются АКР, в котором каждый агент решает задачу независимо от других (он может либо игнорировать другие агенты, либо взаимодействовать с ними лишь опосредованно — через сообщения или разделяемую память). Но это означает, что многоагентные системы, которые по сути являются МАА, естественно сводятся к терминам АКР. Этот факт с практической точки зрения представляет большой интерес.

К многоагентным системам относятся, в частности, эмуляции жизни, эволюции, социума и т.п., некоторые игры и многое другое. Все эти системы могут интерпретироваться как алгоритмы АКР и реализовываться поверх библиотеки их поддержки, что существенно расширяет область применимости такой библиотеки.

Примером АКР, которые не является МАА, может служить алгоритм гравитационного поиска (англ. Gravitational Search Algorithm ) [30]. Агентами в этом алгоритме обладают координатой $X_i(t)$ в пространстве решений, скоростью перемещения в этом пространстве — $V_i(t)$ и массой (авторы предлагают использовать три вида масс: инерционную $M_{ii}(t)$, активную $M_{ai}(t)$ и пассивную $M_{pi}(t)$ гравитационные, однако способы обновления и вычисления их приводят только для ситуации, когда $M_i(t) = M_{ai}(t) = M_{pi}(t) = M_{ii}(t)$). Для агента определена его оценка — функция качества решения им определенного $f_i(t) = f(X_i(t))$.

Тогда алгоритм каждой итерации следующий:

1. Вычислить силы взаимодействия для каждой пары объектов. Для $d$ координаты они вычисляются по формуле:

$$F_{ij}(t) = G(t)\frac{M_{pi}(t)M_{aj}(t)}{R_{ij}(t)}\left(x_i^d(t) - x_j^d(t)\right)$$

где $R_{ij}(t) = \left\|X_i(t),\, X_j(t)\right\|_2$ — евклидово расстояние.

Это правило является видоизмененным законом всемирного тяготения. Отличия заключаются во-первых в том, что множитель $R_{ij}(t)$ имеет степень $-1$, а не $-3$, а во-вторых, в том что гравитационная постоянная $G$ введена как функция от времени. Первое изменение было выведено авторами экспериментально — так алгоритм быстрее сходился. Второе является следствием поисковой природы алгоритма — для нахождения точных решений необходимо повышать «аккуратность» алгоритма со временем, для чего необходимо уменьшить силы взаимодействия.

2. С помощью независимых случайных величин $rnd_i \in [0, 1]$, вычисляются $F_i(t) = \sum\limits_{j=1,\, j\neq i}^{N} rnd_j F_{ij}(t)$ и $a_i(t) = F_i(t)/M_{ii}(t)$. Это, соответственно, правило сложения сил, в которое добавили элемент случайности, и второй закон Ньютона.

3. Производится перемещение объекта:

$$V_i(t+1) = rand_i V_i(t) + a_i(t)$$
$$X_i(t+1) = X_i(t) + V_i(t+1)$$

где $rand_i$ — независимые случайные величины равномерно распределенные на $[0, 1]$. Затем пересчитывается его оценка.

4. Обновляются массы объектов:

$$w(t) = \min_{i \in \{1...N\}} f_i(t)$$
$$b(t) = \max_{i \in \{1...N\}} f_i(t)$$
$$m_i(t) = \frac{f_i(t) - w(t)}{b(t) - w(t)}$$
$$M_i(t+1) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}$$

На примере трех вышеприведенных алгоритмов можно рассмотреть еще одну важную характеристику АКР — тип пространства агентов. В части алгоритмов, в т.ч. в PSO и в GSA, используются агенты располагаемые (обладающие координатой) в пространстве допустимых решений, в то время как в остальных алгоритмах (из приведенных — в ACO) — в пространстве задачи. Агенты первого типа являются агентами, оптимизирующими решения. Они аналогичны особям, используемым в ГА, LEM и других эволюционных алгоритмах [31]. Агенты второго типа являются агентами, строящими решение. Они могут быть использованы только в АКР и, более того, только в МАА. Большинство МАА (в т.ч. многоагентные системы) основаны на них. Однако, существуют исключения: упомянутые выше PSO, SDS и другие.

## 1.5. Общие свойства АКР

Сформулируем общие свойства АКР:

1. Алгоритм работает в некотором пространстве с координатами (карта).
2. Работа алгоритма сводится к оперированию множеством (рой) сущностей (агентами) в этом пространстве.

3. Состояние алгоритма определяется состоянием роя. Реже также используется память ассоциированная с точками пространства и/или небольшое количество глобальных переменных.

4. Алгоритм итеративный. Одну итерацию будем называть ходом.

5. В течении хода алгоритм производит независимые действия над каждым агентом роя (обработка). Эти действия не образуют сторонних эффектов друг на друга: т.е. результат обработки для каждого агента не зависит от того, были ли обработаны другие агенты, в каком порядке они были обработаны, как они изменились и изменили хранимые данные в ходе обработки.

6. Условие остановки алгоритмом не специфицируется.

7. Начальное состояние данных и роя дается алгоритму извне. Координаты агентов в начальном рое обычно случайны (реже при их генерации используется эвристика).

8. Алгоритм стохастический. Генерация случайных чисел (обычно многократная) необходима при обработке каждого агента.

МАА в дополнение к вышеперечисленным также обладает следующими свойствами:

1. Его агенты независимы. Результат обработки любого агента не зависит от состояния других агентов в любой момент времени. Взаимодействие агентов может быть лишь опосредованным, например, через совместный доступ к памяти.

2. Агенты могут действовать лишь в своей ограниченной окрестности.

АКР с оптимизирующими агентами обладает лишь одним специфичным свойством: карта является пространством допустимых решений, таким образом каждому агенту соответствует решение. Меру оптимальности (или отношение «быть оптимальнее») определенную на решениях можно распространить на агентов и затем использовать для их сравнения.

В АКР со строящими агентами карта является пространством задачи. Таким образом каждый агент не является полноценным решением, а значит невозможно без введения эвристики производить их сравнение.

# Литература


1. Russell S. J., Norvig P. Artificial Intelligence: A Modern Approach. — 2nd edition. — Englewood Cliffs, NJ, USA : Prentice-Hall, 2003.

2. McCarthy J. Artificial intelligence, logic and formalizing common sense // Philosophical Logic and Artificial Intelligence. — Dordrecht, Netherlands : Kluwer Academic, 1989. — P. 161–190.

3. Nilsson N. J. Principles of Artificial Intelligence. — Palo Alto, CA, USA : Tioga Publishing Company, 1979.

4. Hart P. E., Nilsson N. J., Raphael B. Correction to «a formal basis for the heuristic determination of minimum cost paths» // SIGART Bulletin. — 1972. — no. 37. — P. 28–29.

5. Engelbrecht A. P. Computational Intelligence: An Introduction. — 2nd edition. — Hoboken, NJ, USA : John Wiley and Sons, 2007.

6. Wlodzislaw D. What is computational intelligence and where is it going? // Challenges for Computational Intelligence. — 2007.

7. Konar A. Computational Intelligence: Principles, Techniques and Applications. — New York, NY, USA : Springer, 2005.

8. Гастев Ю. А. Определение // БСЭ. — 3-е изд. — Москва : Советская энциклопедия, 1988.

9. Хайкин С. Нейронные сети: полный курс. — 2е изд. — Москва : Вильямс, 2006.

10. Passino K. M., Yurkovich S. Fuzzy Control. — Menlo Park, CA, USA : Addison–Wesley–Longman, 1998. — P. 475.

11. Емельянов В. В., Курейчик В. В., Курейчик В. М. Теория и практика эволюционного моделирования. — Москва : Физматлит, 2003. — С. 432.

12. Fogel L. J., Owens A. J., Walsh M. J. Artificial intelligence through simulated evolution. — Chichester, WS, UK : John Wiley and Sons, 1966.

13. Галал . М. Эволюционное учение // Энциклопедия Кирилла и Мефодия. — Москва : Кирилл и Мефодий, 2003.

14. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы. — Москва : Горячая Линия–Телеком, 2007. — С. 452.

15. Jong K. A. D. An analysis of the behavior of a class of genetic adaptive systems : Ph. D. thesis / K. A. De Jong ; University of Michigan. — Ann Arbor, MI, USA, 1975.

16. Michalski R. S. Learnable evolution model: Evolutionary processes guided by machine learning // Machine Learning. — 2000. — Vol. 38, no. 1-2. — P. 9–40.

17. Берг Л. С. Номогенез или Эволюция на основе закономерности. — Петергоф : Государственное издательство, 1922. — С. 306.

18. Bishop J. M. Stochastic Searching Networks // First IEE Conference on Artifical Neural Networks. — 1989. — P. 329–331.

19. Nasuto S. J., Bishop J. M., Lauria S. Time Complexity of Stochastic Diffusion Search // Neural Computation ’98. — 1998.

20. Dorigo M., Maniezzo V., Colorni A. The Ant System: Optimization by a colony of cooperating agents // IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics. — 1996. — Vol. 26, no. 1. — P. 29–41.

21. 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.

22. Eo evolutionary computation framework. — URL: http://eodev.sourceforge.net/.

23. Evolutionary computation framework. — URL: http://gp.zemris.fer.hr/ecf/.

24. DeJong K. A. Evolutionary Computation: A Unified Approach. — Cambridge, MA, USA : The MIT Press, 2006.

25. Bonabeau E., Dorigo M., Theraulaz G. Swarm Intelligence: From Natural to Artificial Systems. — New York, NY, USA : Oxford University Press, 1999.

26. Kennedy J., Eberhart R. C. Swarm intelligence. — San Francisco, CA, USA : Morgan Kaufmann, 2001.

27. Beni G., Wang J. Swarm intelligence in cellular robotic systems // NATO Advanced Workshop on Robotics and Biological Systems. — 1989.

28. Асанов М. О., Баранский В. А., Расин В. В. Дискретная математика: графы, матроиды, алгоритмы. — Ижевск : НИЦ «РХД», 2001. — С. 288.

29. Swarm Intelligence in Data Mining / Ed. by Ajith Abraham, Crina Grosan, Vitorino Ramos. — New York, NY, USA : Springer, 2006. — Vol. 34 of Studies in Computational Intelligence.

30. Rashedi E., Nezamabadi-pour H., Saryazdi S. Gsa: A gravitational search algorithm. // Information Science. — 2009. — Vol. 179, no. 13. — P. 2232–2248.

31. Baykasoglu A. Evolutionary computation for modeling and optimization. // The Computer Journal. — 2008. — Vol. 51, no. 6. — P. 743.