РАСПРЕДЕЛЕННАЯ ЭВОЛЮЦИОННАЯ СЕТЬ ДЛЯ РЕШЕНИЯ МНОГОКРИТЕРИАЛЬНЫХ ОПТИМИЗАЦИОННЫХ ЗАДАЧ В СИСТЕМАХ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

М.А. Хивинцев,
аспирант кафедры бизнес-аналитики Национального исследовательского университета «Высшая школа экономики»

А.С. Акопов,
доктор технических наук, профессор кафедры бизнес-аналитики Национального исследовательского университета «Высшая школа экономики»

E-mail: mkhivintsev@hse.ru, aakopov@hse.ru
Адрес: г. Москва, ул. Кирпичная, д. 33/5

В статье представлен новый подход к решению многокритериальных оптимизационных задач большой размерности, реализуемых, в частности, в системах имитационного моделирования класса AnyLogic с помощью распределенных вычислений. Предложена новая концепция построения распределенной эволюционной сети, основанная на разбиении пространства искомых переменных на кластеры и присвоения каждому вычислительному элементу сети своего кластера, по которому осуществляется поиск промежуточных результатов с помощью взаимодействующих генетических алгоритмов.

Ключевые слова: имитационное моделирование, генетические алгоритмы, распределенные вычисления, многокритериальная оптимизация.

1. Введение

Существует известная проблема поиска субоптимальных решений в имитационных моделях «больших систем», таких как, система управления технологическими режимами сложного производства, система оптимального управления финансовыми и материальными потоками крупной корпорации, система управления инвестиционным портфелем и др. [1]. Поиск оптимальных решений в подобных системах должен осуществляться по множеству, как правило, конкурентных и разномасштабных критериев со сложными ограничениями.

Проблема заключается в том, что оптимизация сразу по множеству критериев требует значительных временных и вычислительных ресурсов. При

этом, так как целевые функции многих переменных (более 10) порождают многомерное пространство поиска решений, вычислительная сложность оптимизационной задачи, как правило, становится преградой к ее решению за допустимое время. Аналитическое описание и строгое решение такой задачи на практике, как правило, не осуществимо.

При решении такого класса задач хорошо зарекомендовали себя алгоритмы метаэвристического класса, позволяющие, не исследуя полностью область допустимых решений, с определенной вероятностью находить глобальный экстремум в однокритериальных задачах и фронт Парето в многокритериальных.

Одним из распространенных классов метаэвристических алгоритмов являются генетические алгоритмы (ГА), теория которых была разработана J. H. Holland в 1975 г. [2]. Доказано, что ГА обладают значительно меньшей вычислительной емкостью, чем простой метод перебора, и рядом плюсов по сравнению с другими способами решения оптимизационных задач, однако всегда имеют риск схождения к локальному оптимуму вместо глобального [3 – 5].

Следует отметить, что имеется ряд работ в области проектирования параллельных генетических алгоритмов [6 – 11]. Первой работой в этой области можно считать работу Grosso P.B. [6] выполненной еще в 1985 г.

Одним из основных преимуществ предлагаемой распределенной эволюционной сети является возможность разбиения процесса решения одной задачи на несколько параллельных вычислительных подпроцессов, объединенных в распределенную сеть.

В качестве средства имитационного моделирования (ИМ) в данной работе выбран программный продукт AnyLogic, так как это один из немногих инструментов, который поддерживает все основные методы ИМ: процессно-ориентированный, системно динамический и агентный, а также любую их комбинацию. Отметим, что в программном продукте AnyLogic реализован однокритериальный оптимизационный алгоритм OptQuest (http://www.opttek.com/OptQuest), основанный на метаэвристиках рассеянного поиска (scatter search) и поиска «табу» (tabu search). Однако хорошо известно, что заложенные в основе OptQuest алгоритмы неэффективны при размерности оптимизационной задачи более 10 переменных. Кроме того, в AnyLogic отсутствуют решения для многокритериальной оптимизации, нахождения и визуализации фронта Парето, а также распараллеливания вычислений.

Итак, практически все современные системы имитационного моделирования (в том числе, AnyLogic, Powersim и др.) не имеют в своем арсенале достаточно эффективных средств решения многокритериальных оптимизационных задач большой размерности. Поэтому построение РЭС является весьма актуальной задачей.

Целью данной статьи является разработка распределенной эволюционной сети, предназначенной для решения многокритериальных оптимизационных задач большой размерности. Представлен новый подход к построению комплексного программного решения по интеграции распределенной эволюционной сети с системой имитационного моделирования AnyLogic.

2. Концепция распределенной эволюционной сети

Итак, для ускорения поиска решений в ГА при распараллеливании вычислительных процессов необходимо добиться снижения суммарного числа расчетов функций приспособленности. Для этого предлагается наиболее действенный способ — сокращение размера популяции, с которой будет работать ГА на каждом из процессов. В этом случае на всех шагах эволюции в результате операции скрещивания будет создаваться меньшее число потомков и, следовательно, для меньшего их количества будут рассчитываться функции приспособленности.

Важной задачей при сокращении размера популяции является обеспечение высокой вероятности схождения алгоритма к глобальному экстремуму. Это выполнимо только при сохранении оптимального соотношения между длиной хромосомы и размером популяции. Поэтому сокращение размера популяции предлагается достичь за счет уменьшения длины хромосомы, разбив пространство переменных на кластеры и присвоив каждому вычислительному элементу сети свой кластер, по которому он будет искать промежуточные результаты. Этот принцип заложен в новую предлагаемую методику эффективного решения многокритериальных оптимизационных задач с функциями многих переменных, используя *распределенную эволюционную сеть (РЭС, или distributed evolutionary network - DEN)*.

РЭС включает в себя N+1 вычислительных элементов (процессоров). На одном из них выполняется управляющий процесс, на N других - вычислительные. В основу механизма управления РЭС лег

алгоритм управления распределенной эволюционной сетью (АУРЭС, или distributed evolutionary network algorithm - DENA), а за базу для каждого из N вычислительных процессов - распространенный метод решения многокритериальных задач путем построения фронта Парето - SPEA2 (*Strength Pareto Evolutionary Algorithm*) [12; 13]. Каждый из вычислительных процессов будет выполнять оптимизацию по своему кластеру переменных, что позволит выполнять ГА с меньшим размером популяции, и, как следствие, снизить время на поиск решения для каждого из процессов. В этом заключается основное отличие от классической островной модели в теории ГА [11]. Все острова работают с общим набором переменных и хромосомами равной длины.

Другим отличием РЭС от островной модели является наличие центрального элемента управления, который управляет ходом эволюции всей системы и использует архив решений, полученных на прошлых стадий эволюции. Согласно АУРЭС с определенной периодичностью управляющий процесс аккумулирует промежуточные результаты каждого из вычислительных процессов, сопоставляет их с архивными решениями, обрабатывает и перераспределяет генетический материал между вычислительными процессами, попутно давая им команду к запуску следующей итерации эволюционного процесса. После этого вычислительные процессы начинают асинхронно следовать ГА поиска Парето-оптимальных решений согласно методу SPEA2 (Strength Pareto Evolutionary Algorithm) [12; 13]. Когда они заканчивают очередную итерацию, то передают результаты управляющему процессу и ожидают от него нового генетического материала, возникшего в результате обработки результатов работы ГА всех параллельных процессов – через управляющий процесс происходит обмен генетическими материалами между вычислительными процессами. С каждым из описанных эволюционных циклов система движется к фронту Парето наивысшего ранга. Схема функционирования РЭС представлена *на рис. 1.*

Основной функцией РЭС является решение многокритериальных оптимизационных задача в ИМ. Для ее выполнение РЭС имеет установленную связь с ИМ.

На вход имитационной модели от РЭС поступают новые значения для переменных, для которых по установленным модельным связям должны быть рассчитаны значения целевых функций, которые затем передаются из имитационной модели в РЭС. Таким образом, оцениваются значения функций приспособленности ГА, выполняемых на каждом из вычислительных процессов РЭС. Далее распределенные ГА формируют новые (лучшие) значения для переменных имитационной модели, и осуществляется их очередное присваивание искомым переменным с последующим прогоном модели для оценки фитнес-функций.

3. Механизм управления распределенной эволюционной сетью

К сожалению, классическая островная модель ГА [11] не может существенно снизить потребность в большом размере популяции, так как длина хромосомы остается неизменной, а в границах каждого острова для оптимального хода эволюционного процесса пропорция между длиной хромосомы и числом членов популяции должна также сохраняться, в противном случае на локальном уровне каждого из островов будет значительно меньше генетического материала, что повысит вероятность схождения к локальным, а не глобальным, оптимумам. Если на каждом из островов останется столько же членов популяции, сколько и при одном вычислительном процессе, то время поиска решений в обоих случаях будет практически одинаковым.

Чтобы добиться действенного снижения эффективного размера популяции, в рамках предлагаемой методики решения рассматриваемого класса задач предлагается новый алгоритм управления РЭС (АУРЭС). Разберем его по стадиям.

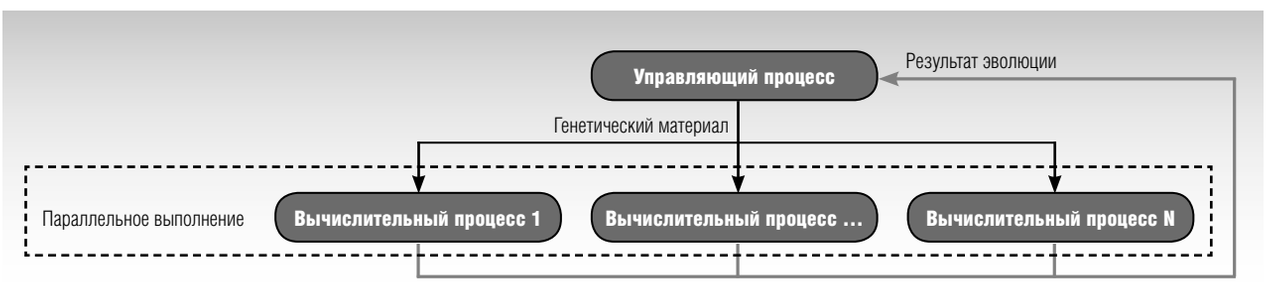

Рис. 1. Схема функционирования распределенной эволюционной сети

Стадия подготовительная.

Прежде чем приступать к решению задачи в РЭС необходимо провести статистический анализ переменных, построить их корреляционную матрицу и выявить, как непрямые связи между ними влияют на конечный результат. В конечном счете, нужно объединить переменные в кластеры. В каждом кластере будут находиться те переменные, изменение которых вызывает наиболее явную потребность в изменении других для достижения наилучшего результата.

Каждому из вычислительных процессов нужно определить свой кластер переменных, поэтому число кластеров должно совпадать с количеством вычислительных процессов.

При разбиении переменных на кластеры нужно стремиться к тому, чтобы размеры кластеров были схожими, так как в этом случае для каждого из вычислительных процессов можно ожидать примерно равную длину хромосом, а, значит и вычислительная мощность распределится равномерно по процессам.

Стадия исходная.

На исходной стадии управляющий процесс осуществляет формирование популяции случайным образом. Размер общей популяции равен сумме эффективных размеров популяций для каждого из вычислительных процессов, зависящий от длины хромосомы и настроек ГА.

Далее управляющий процесс делит популяцию на N подпопуляций согласно алгоритму, поддерживающему разнообразие генетического материала в подпопуляциях на одинаковых аллелях, и передает особей из одной подпопуляции на один вычислительный процесс.

Эффективный размер подпопуляции для каждого из вычислительных процессов необходимо выбрать исходя из числа переменных его кластера, определяющего длину хромосомы. После распределения подпопуляций управляющий процесс передает вычислительным процессам персональные настройки и команду к запуску ГА. Далее следует переход к первой стадии.

Стадия 1.

С первой стадии стартует первая итерация поиска решений. Вычислительные процессы работают параллельно и асинхронно. Поэтому итерации, равно как и стадии 1-4, выполняются независимо для каждого из вычислительных процессов.

На первой стадии происходит первичное вычисление функций приспособленности для каждой из особей подпопуляции на всех вычислительных процессах.

Стадия 2.

Вторая стадия является наиболее ресурсозатратной. В ходе нее каждый из вычислительных процессов реализует построение фронта Парето высшего ранга только по тому набору переменных, которые включены в его кластер. Остальные переменные принимают фиксированные значения и остаются неизменными на протяжении всей второй стадии. В результате сокращается фактическая длина хромосомы, так как в ней появляется «неактивная» часть, не участвующая в эволюции и соответствующих вычислительных процессах.

Нахождение фронта Парето осуществляется при помощи одного из распространенных эффективных методов на основе ГА – SPEA2. На *рис. 2* показаны исходные подпопуляции и конечный результат построения фронта Парето, варьируя только выделенные переменные у каждого из процессов.

В момент достижения критерия остановки любым из вычислительных процессов, только этот процесс переходит к Стадии 3. Таким образом, каждый из процессов независимо от других переходит к следующей стадии.

Стадия 3.

Вычислительный процесс передает результат второй стадии управляющему процессу. Далее в рамках управляющего процесса обеспечивается сравнение полученных на второй стадии результатов с имеющимися архивными результатами. При этом управляющий процесс опрашивает остальные вычислительные процессы для обновления результа-

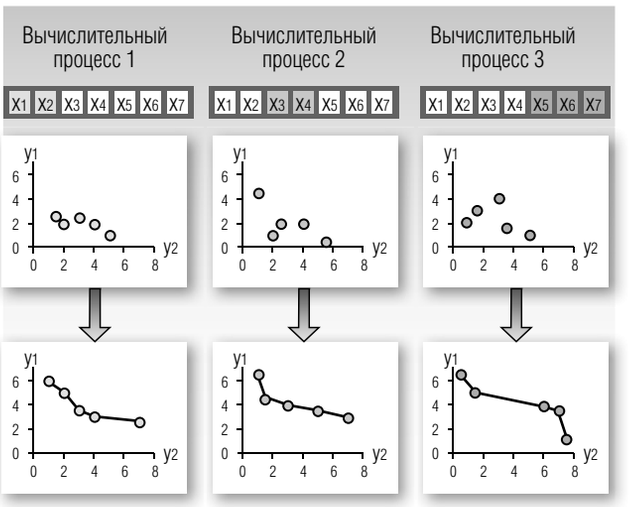

Рис. 2. Процесс построения фронта Парето на каждом из вычислительных процессов

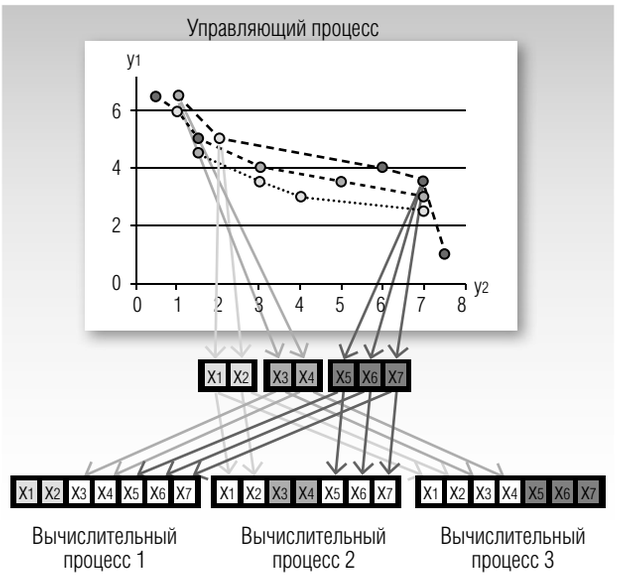

Рис. 3. Перестроение фронтов Парето нескольких рангов управляющим процессом на основе имеющихся у него данных

тов в архиве. В результате управляющий процесс из всех известных ему решений строит фронт Парето нескольких рангов. Может оказаться, что фронт высшего ранга будет состоять из решений нескольких процессов, как это показано на *рис. 3.*

Стадия 4:

Для каждого из кластеров переменных случайным образом выбирается по одному набору решений (по одной точке на границе Парето), но вероятность выбора того решения выше, который находится на фронте Парето более высокого ранга.

У процесса, инициировавшего обмен генетическим материалом с управляющим процессом, происходит изменение неактивной части хромосомы – тех переменных, которые были фиксированы в ходе второй стадии. Эти переменные принимают значения, соответствующие значениям этих переменных из случайно выбранного набора решений (*рис. 4*).

На этом завершается очередная итерация работы АУРЭС и, если не достигнут критерий остановки вычислительной процедуры, то происходит переход к следующей итерации поиска решений на глобальном уровне и возврат инициировавшего обмен вычислительного процесса к стадии 1, но с новыми значениями переменных. Стадии 1-4 повторяются до тех пор, пока управляющий процесс в течение определенного числа итераций не обнаружит улучшение хотя бы одной из точек фронта Парето наивысшего ранга.

4. Пример использования распределенной эволюционной сети, интегрированной с моделью AnyLogic

Спроектированная РЭС позволяет решать многокритериальные задачи в имитационных моделях высокого уровня сложности. Данный класс задач характеризуется несколькими свойствами:

1. имитационная модель имеет сложные внутренние нелинейные зависимости между переменными;

2. требуется обеспечить решение многокритериальной оптимизационной задачи в системе AnyLogic;

3. оптимизация производится в пространстве большой размерности (10 и более искомых переменных, несколько целевых функций);

4. вычисление Парето-оптимальных решений должно происходить за приемлемое время.

Для демонстрационного примера решения задачи подобного класса далее рассматривается имитационная модель управления производственной компанией, занимающейся производством кабельно-проводниковой продукции и обладающей типичными для своей деятельности бизнес-процессами.

Основные параметры компании, которые были заложены в имитационную модель:

- ✦ Номенклатура включает в себя 24 вида изделий;
- ✦ 2 производственных цеха;

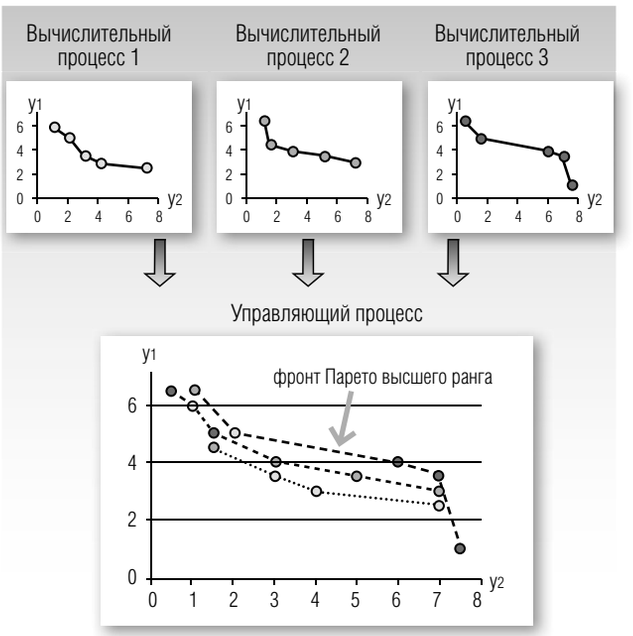

Рис. 4. Присваивание неактивным на каждом из вычислительных процессов переменным новых значений

✦ 103 вида оборудования;

✦ работа строится в 2 смены;

✦ 38 единиц сырья;

✦ 380 производственных сотрудников;

✦ 15 должностных (функциональных) ролей для сотрудников.

Важно отметить, что взаимосвязь между основными объектами модели является «многие-ко-многим» — другими словами, для производства одного вида продукции может использоваться несколько видов сырья и один вид сырья может быть использован для нескольких видов продукции. В результате переменные рассматриваемой модели являются многомерными (имеют от 1 до 3 размерностей), что порождает трехмерное пространство поиска решений и повышает вычислительную сложность оптимизационной задачи.

Имитационная модель предприятия была реализована в системе AnyLogic.

Оптимизационная задача формулируется следующим образом: необходимо определить цены, обеспечивающие наилучшую динамику продаж для максимизации двух целевых критериев на следующий квартал: объема продаж (в натуральном выражении) и чистой прибыли. Данный набор целей является распространенным в бизнесе и отражает интересы акционеров – иметь не только прибыльную компанию, но и быть лидером по доле рынка. Оба эти критерия оказывают весомое влияние на рыночную стоимость компании.

В результате будет построена граница Парето, которую образуют недоминируемые Парето-решения. Другими словами, остальные решения не позволят добиться одновременно более высоких значений чистой прибыли и объема продаж.

Итак, в рассматриваемой бикритериальной оптимизационной задаче имеется две целевые функции: объем продаж в натуральном выражении и чистая прибыль. Конечной целью максимизации объема продаж является увеличение доли рынка. При этом данная цель является конкурентной по отношению к чистой прибыли.

Целевая функция максимизации объема продаж в натуральном выражении имеет следующий вид:

$$\sum_{t=t_0}^{t_0+T}\sum_{i=1}^{I} Q_i\left(p_i(t)\right) \to \max_{p_i(t)}, \qquad (1)$$

$$Q_i(t) = \frac{Q_i(t-1)}{\left(p_i(t) / p_i(t-1)\right)^{\xi_i}}. \qquad (2)$$

Целевая функция чистой прибыли:

$$\sum_{t=t_0}^{t_0+T}\sum_{i=1}^{I} p_i(t)Q_i\left(p_i(t)\right) - Costs\left(Q_i(t)\right) - Expenses(t) \to \max_{p_i(t)}, \qquad (3)$$

где: $t = t_0, t_0+1, ..., t_0+T$ – время в модели (по месяцам), T – год.

$i = 1, 2, ..., I$ – индекс товара (по умолчанию $I = 24$).

$p_i(t)$ – цена i-ого товара,

$Q_i(t)$ – объем продаж в натуральном выражении,

$Costs(t)$ – себестоимость производства (переменные затраты),

$Expenses(t)$ – суммарные постоянные затраты: коммерческие, управленческие, прочие расходы и налоги.

ξ_i – эластичность объема продаж i – ого товара по соответствующим ценам ($\xi_i \geq 0$).

Следует отметить, что все слагаемые в формуле чистой прибыли имеют связь между собой, установленную через отношения внутри модели. Так, например, заложенные функции спроса, издержек и производства позволяют найти оптимальный баланс между планом продаж и планом производства с учетом производственных ограничений.

В общей сложности модель состоит из 125 переменных размерностью от 1 до 2(цеха)×103(оборудования)×24(продукта) и 159 связями между ними. Однократный просчет модели занимает 0.2 сек на процессоре Intel core i7 с тактовой частотой 2000 MHz.

Встроенного функционала в AnyLogic недостаточно для решения поставленной оптимизационной задачи, поэтому рассматриваемая имитационная модель была интегрирована с разработанной РЭС с использованием технологии IntelliJ IDEA (на языке программирования Java).

Согласно алгоритму управления РЭС на первом этапе 24 искомые переменные были разбиты на 3 кластера в зависимости от используемого оборудования для их производства.

РЭС была составлена из 4 вычислительных элементов: управляющий процессор и 3 вычислительных, обеспечивающих поиск решений по своему кластеру переменных.

В результате, формирование фронта Парето заняло всего 3 минуты. При этом каждый из вычислительных процессоров произвел 720 итераций вычисления целевых функций рассматриваемой имитационной модели. Таким образом, реализованная по предложенной концепции РЭС система обеспечила решение поставленной многокритериальной оптимизационной задачи за допустимое время.

5. Заключение

В данной статье был предложен и реализован подход к решению многокритериальных оптимизационных задач большой размерности в системах имитационного моделирования класса AnyLogic. Такой подход основан на использовании распределенной эволюционной сети (РЭС), в которой эффект от распараллеливания вычислительных процессов выше, чем при классической островной модели. Это достигается благодаря способности предложенного алгоритма к разбиению всего пространства решений на кластеры, каждый из которых передается на свой вычислительный процесс, исполняя генетический алгоритм (ГА) на локальном уровне. В этом случае для ГА, работающего с целевыми функциями из меньшего числа переменных, для качественных решений требуется меньший размер популяции и, как следствие, сокращению числа вызовов функций приспособленности, что ведет к значительному снижению времени схождения алгоритма к конечному решению.

При координации управляющего процесса с определенной периодичностью происходит обмен генетическим материалом между процессами. Это обеспечивает возможность формирования фронта Парето наивысшего ранга за допустимое время, что свидетельствует о высокой эффективности данной методики решения.

Разработан платформенно-независимый программный комплекс на основе предложенной концепции РЭС на языке программирования Java, позволивший интегрировать РЭС и имитационные модели, разработанные в средстве имитационного моделирования AnyLogic.

Функциональность РЭС была апробирована на демонстрационном примере. ■

Литература

1. Akopov A.S. Designing of integrated system-dynamics models for an oil company // International Journal of Computer Applications in Technology. – 2012. – Vol. 45, No. 4. – P. 220-230.
2. Holland J.H. Adaptation in natural and artificial systems. – Ann Arbor: University of Michigan Press, 1975.
3. Hajena P., Lin C.-Y. Genetic search strategies in multicriterion optimization design // Structural Optimization. – June 1992. – Vol. 4. – New York: Springer, P. 99-107.
4. Емельянов В.В., Курейчик В.В., Курейчик В.М. Теория и практика эволюционного моделирования. – М.: Физматлит, 2003.
5. Курейчик В.М., Лебедев Б.К., Лебедев О.К. Поисковая адаптация: теория и практика. – М.: Физматлит, 2006.
6. Grosso P.B. Computer simulations of genetic adaptation: Parallel subcomponent interaction in a multilocus model // Unpublished doctoral dissertation. – The University of Michigan (University Microfilms №8520908), 1985.
7. Alba E., Troya J.M. A survey of parallel distributed genetic algorithms // Complexity. – 1999. – Vol.4, № 4. – P. 31-52.
8. Alba E., Troya J.M. Influence of the migration policy in parallel distributed GAs with structured and panmictic populations // Application Intelligence. – 2000. – Vol.12, №3. – P. 163-181.
9. Alba E., Troya J.M. Analyzing synchronous and asynchronous parallel distributed genetic algorithms // Future Generation Computer Systems. – 2001. – Vol.17, №4. – P. 451-465.
10. Golub M., Jakobovic D. A new model of global parallel genetic algorithm // Proceedings of 22nd International Conference on Information Technology Interfaces IVI. – 2000. – P. 363-368.
11. Silva J.D., Simoni, P.O. The island model parallel GA and uncertainty reasoning in the correspondence problem // Proceedings of International Joint Conference on Neural Networks (IJCNN '01). – 2001. – Vol. 3. – P. 2247-2252.
12. Zitzler E., Thiele L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach // IEEE Transactions on Evolutionary Computation. – 1999. – Vol. 3, No. 4. – P. 257-271.
13. Bleuer S., Brack M., Thiele L., Zitzler E. Multiobjective genetic programming: Reducing bloat by using SPEA 2 // Proceedings of the 2001 Congress on Evolutionary Computation (CES-2001). – 2001. – P. 536-543.