УДК 621.3.06
DOI: 10.15827/0236-235X.118.217-226

Дата подачи статьи: 10.04.17
2017. Т. 30. № 2. С. 217–226

ОДНОРОДНАЯ РАСПРЕДЕЛИТЕЛЬНАЯ ЗАДАЧА НА ОСНОВЕ МОДЕЛЕЙ АДАПТИВНОГО ПОВЕДЕНИЯ МУРАВЬИНОЙ КОЛОНИИ

Б.К. Лебедев, д.т.н., профессор, lebedev.b.k@gmail.com;

О.Б. Лебедев, к.т.н., доцент, lebedev.ob@mail.ru;

Е.М. Лебедева, аспирант, lebedeva.el.m@mail.ru

*(Институт компьютерных технологий и информационной безопасности
Южного федерального университета, ул. Энгельса, 1, г. Таганрог, 347928, Россия)*

В данной работе предлагается решение однородной распределительной задачи. Приводится постановка этой задачи, рассматриваются основные группы алгоритмов ее решения – приближенные и точные, а также их достоинства и недостатки. Описана предлагаемая новая парадигма комбинаторной оптимизации, базирующаяся на моделировании адаптивного поведения муравьиной колонии.

Решением однородной распределительной задачи является ее графическое представление в виде двудольного графа. Для решения данных задач были предложены новые механизмы. Основу метаэвристики алгоритма на основе муравьиной колонии составляет комбинация двух техник. Базовая техника состоит в поиске наилучшего решения с использованием механизмов адаптивного поведения муравьиной колонии. Муравей строит какое-то конкретное решение, при этом используется встроенная процедура, в основе которой лежит конструктивный алгоритм. Построенный на графе поиска решений двудольный граф – основное отличие предлагаемого муравьиного алгоритма от существующей канонической парадигмы.

При нахождении оптимальных решений оптимизационных задач, которые допускают представление решений в виде двудольных графов, данный подход будет достаточно эффективным.

Проведенные исследования показали, что муравьиный алгоритм позволяет получать более качественные решения, чем известные алгоритмы. Сравнив результаты, можно сказать, что они улучшились на 3–4 %.

Ключевые слова: задача о назначениях, двудольный граф, оптимизация, роевой интеллект, муравьиная колония, адаптивное поведение, однородная распределительная задача.

В области теории расписаний самым распространенным направлением является исследование классических однородных распределительных задач [1]. Такие задачи часто применяются в различных сферах человеческой деятельности, например там, где необходимо эффективно выполнить организацию и планирование каких-либо работ, различных заданий и требований. В общем случае имеем какое-то количество работ и какое-то количество исполнителей. Выполнение любым работником какой-либо (но только одной) работы происходит с разными (неодинаковыми) затратами. Необходимо распределить работы таким образом, чтобы обойтись минимальными затратами. Примером распределительной задачи является задача составления плана выполнения комплекса программ в многопроцессорных вычислительных системах [2]. Она является минимаксной однородной распределительной задачей теории расписаний.

Поскольку распределительная задача относится к классу NP-полных задач, разработка новых, более эффективных методов, используемых для их решения, является актуальной проблемой теории расписаний. Зарубежными и российскими учеными разработано множество алгоритмов и методов, отличающихся как эксплуатационными свойствами, так и областью применения и решающих однородные распределительные задачи. Все методы решения однородных распределительных задач делятся на две группы – приближенные и точные.

Самыми известными и наиболее эффективными из методов точного решения распределительной задачи являются алгоритмы, в основе которых лежат принципы метода ветвей и границ [2–4]. Однако следует отметить, что эти алгоритмы имеют свойство экспоненциального роста сложности искомого решения относительно размерности распределительной задачи. Это особенно заметно, когда распределяемые в задаче задания характеризуются небольшой вариацией значений их размеров. Основные два дефекта, которые часто возникают в алгоритмах на основе метода ветвей и границ, указаны в работе [4]. В ней предлагается модификация, которая может уменьшить комбинационную сложность алгоритма, предложенного Романовским [2]. Главная особенность метода ветвей и границ состоит в том, что обязательно возникнет такой пример, при решении которого необходим полный перебор. Поэтому для задач повышенной размерности возникнут большие временные затраты, хотя комбинаторная сложность уменьшится.

Алгоритмы приближенного поиска не могут гарантированно найти при решении распределительной задачи глобальный оптимум. Они позволяют получать некоторое допустимое или приемлемое решение [4–6]. Точность алгоритмов этого класса невысока, но основное достоинство заключается в высокой скорости получения решения, которая имеет полиномиальную, а иногда даже линейную

зависимость от заданного порядка задачи. Наибольшее распространение в классе приближенных алгоритмов получили списочные и эвристические методы. В основном списочные методы имеют линейную сложность решения (относительно размерности задачи). В связи с этим, с точки зрения времени построения расписаний, они обладают большой эффективностью. К сожалению, при получении решений погрешность может достигнуть 30 %, что неприемлемо к заданным практическим требованиям. Интерес к использованию эвристических подходов возник в связи с низкой точностью приближенных списочных алгоритмов и высокими вычислительными требованиями точных методов.

В литературе [5–11] приведены разработанные ранее алгоритмы (последовательные, итерационные и т.д.). Их обзор, сравнение и анализ показали, что для разработки более эффективного алгоритма требуются новые технологии и подходы. Возможности применения генетического алгоритма для решения однородной распределительной задачи рассматриваются в [12, 13]. Как показывают исследования, алгоритм лучше срабатывает при значительном увеличении популяции и числа итераций. В [14] приведены некоторые возможные модификации использования алгоритма поиска Табу. Эксперименты показывают, что этот алгоритм не всегда способен дать оптимальные решения. Качество решений возрастает в том случае, если его использовать совместно с алгоритмом случайного поиска. В [15] авторами было проведено исследование возможности использования нейронных сетей для решения однородной распределительной задачи и приведены результаты поставленных экспериментов. Предлагается использовать динамическую нейронную сеть, основанную на нейронах непрерывного и двоичного типов. Из результатов экспериментальных исследований следует, что метод позволяет получать решения, результаты которых могут быть улучшены.

Как показывают эксперименты, приближенные (эвристические) методы могут давать большую погрешность, неприемлемую в большинстве случаев. Возникает необходимость в методах, сочетающих противоречивые свойства: полиномиальная зависимость времени решения задачи от ее размерности и точность, близкая к оптимальной. Таким образом, работа посвящается актуальной научно-технической проблеме: исследованию приближенных методов решения задачи распределения программ в многопроцессорных системах с целью повышения точности их работы.

В настоящее время наиболее перспективным является интенсивно разрабатываемое в последние годы научное направление, объединяющее различные математические методы, в которых принятие решений осуществляется на основе принципов, содержащих природные механизмы [15–21]. Эти методы являются итерационными, эвристическими

методами случайного поиска. Наиболее активно развиваются методы роевого интеллекта (Swarm-Intelligence) [17–21]. В них совокупность простых агентов конструирует стратегию своего поведения без глобального управления.

Одним из новейших и наиболее перспективных мультиагентных методов интеллектуальной оптимизации является метод муравьиной колонии [16, 17]. В работе представлен разработанный алгоритм для решения однородной распределительной задачи, использующий модель адаптивного поведения муравьиной колонии.

Постановка однородной распределительной задачи теории расписаний

Пусть исполнительная система состоит из n идентичных, параллельно работающих исполнителей $E = \{e_i / i = 1, 2, \dots, n\}$. На вход системы поступает множество m независимых заданий (работ) $W = \{w_j / j = 1, 2, \dots, m\}$, которые необходимо распределить между исполнителями. Известна стоимость (ресурс выполнения) каждого j -го задания r_j , и она одинакова для любого i -го исполнителя e_i . Таким образом, множеству W сопоставлено множество стоимостей: $R = \{r_j / j = 1, \dots, m\}$. l -м решением однородной распределительной задачи является множество $V^l = \{W_i^l / i = 1, 2, \dots, n\}$, в котором подмножества заданий W_i^l отвечают обязательному свойству:

$$(\forall i, j) [((i \neq j) \& (W_i^l \in V^l) \& (W_j^l \in V^l)) \rightarrow ((W_i^l \cap W_j^l = \emptyset) \& (\cup W_i^l = W))]. \quad (1)$$

Запланированная вариантом решения V^l загрузка заданиями каждого исполнителя e_i оценивается ресурсом:

$$R_i^l = \sum_k r_k, \quad (2)$$

где $r_k / w_k \in W_i^l$.

В качестве оценки решения V^l рассматривается величина

$$F^l = \max(R_i^l). \quad (3)$$

Наиболее распространенным критерием оптимизации однородной распределительной задачи является минимаксный критерий

$$F = \min F^l = \min \max (R_i^l). \quad (4)$$

Это связано с тем, что в большинстве реальных распределительных задач наиболее часто возникает задача минимизации времени выполнения всего комплекса заданий на заданной исполнительной системе. Чем быстрее исполнительная система обслуживает поступающие на выполнение задания, тем выше ее производительность, а значит, выше и экономическая эффективность предприятия, использующего данную систему.

Представим решение распределительной задачи в виде двудольного графа: $H^l = (E \cup W, U^l)$, где $E = \{e_i / i = 1, 2, \dots, n\}$ – множество вершин (первая доля), соответствующих множеству исполнителей, а $W = \{w_j / j = 1, 2, \dots, m\}$ – множество вершин (вторая доля), соответствующих множеству заданий (ра-

бот); U^l – множество ребер, связывающих вершины множества E с вершинами множества W . Для наглядности представления конкретного решения V^l сгруппируем множество заданий W в подмножества (рис. 1).

Пусть U_i^l – множество ребер двудольного графа H^l , связывающих вершину e_i с вершинами множества W_i^l , $\cup U_i^l = U^l$. Отличительной особенностью представленного двудольного графа $H^l = (E \cup W, U^l)$ является то, что число ребер множества U_i^l равно числу вершин множества W_i^l . Каждое ребро $u_z \in U_i^l$, с одной стороны, инцидентно вершине e_i , а с другой – инцидентно только одной вершине $w_k \in W_i^l$. Назовем двудольный граф, представляющий решение V^l , графом-решением H^l . Отметим, что локальная степень любой вершины $w_j \in W$ равна единице, а локальная степень вершины e_i равна мощности множества W_i^l , то есть $\rho(w_j) = 1$, $\rho(e_i) = |W_i^l|$.

В работе поиск решения V^l сводится к поиску на полном двудольном графе H_{nm} такого графа-решения H^l , для которого оценка F^l имеет минимальное значение.

Решение однородной распределительной задачи на основе моделей адаптивного поведения муравьиной колонии

Предлагаемая для муравьиного алгоритма метаэвристика состоит из комбинации двух техник. Сначала на базовом методе строится общая схема. Далее в построенную схему встраивается та или

иная процедура. Следует отметить, что встроенная процедура – это практически всегда самостоятельный алгоритм решения той же задачи, что и метаэвристический метод в целом. Базовый метод заключается в реализации итерационной процедуры поиска лучшего решения на основе механизмов адаптивного поведения муравьиной колонии. Основу встроенной процедуры составляет конструктивный алгоритм построения муравьем некоторой конкретной интерпретации решения. В оптимизации муравьиными колониями [8] конструктивный блок (деятельность искусственных муравьев) играет ключевую роль. В нашем случае в качестве интерпретации решения однородной распределительной задачи служит двудольный граф-решение H^l .

Рассмотрим принципы решения однородной распределительной задачи методами муравьиной колонии. Поиск решений осуществляется на полном двудольном графе. Как указывалось выше, базовый метод заключается в реализации итерационной процедуры поиска лучшего решения. Работа поисковой процедуры начинается с построения в соответствии со спецификой решаемой задачи графа поиска решений. Для поиска решений формируется полный двудольный граф $H_{nm} = (E \cup W, U)$. В графе H_{nm} каждая вершина e_i связана со всеми вершинами множества W , а каждая вершина w_j связана со всеми вершинами множества E , то есть $\rho(w_j) = |U|$, $\rho(e_i) = |W|$, $|U| = nm$. Задается размер популяции искусственных муравьев. За каждым муравьем закрепляется стартовая вершина. В качестве стартовых вершин рассматриваются вершины $w_j \in W$.

Моделирование поведения муравьев связано с распределением феромона на ребрах графа H_{nm} . На начальном этапе на всех ребрах U графа H_{nm} откладывается одинаковое (небольшое) количество феромона Q/v , где $v = |U|$. Параметр Q задается априори. Будем обозначать граф H_{nm} после отложения на нем на итерации t феромона как $H_{nm}(t)$. После начального отложения – $H_{nm}(0)$. Процесс поиска решений итерационный. Каждая итерация t включает три этапа. На первом этапе каждой итерации t выполняются процедуры муравьиного алгоритма. Каждый l -й агент формирует на ребрах графа $H_{nm}(t-1)$ свой собственный граф-решение $H^l(t)$, определяется решение $V^l(t)$, соответствующее графу-решению $H^l(t)$, и оценка решения $F^l(t)$.

На втором этапе итерации t каждый муравей откладывает феромон на ребрах графа $H_{nm}(t-1)$, соответствующих ребрам построенного графа-решения $H^l(t)$.

Количество феромона $\Delta\tau_i(t)$, откладываемое муравьем a_l на каждом ребре построенного графа-решения $H^l(t)$, определяется следующим образом: $\Delta\tau_i(t) = Q / F^l(t)$, где t – номер итерации; Q – общее количество феромона, откладываемое муравьем на ребрах графа-решения $H^l(t)$; $F^l(t)$ – целевая функция для решения, полученного муравьем a_l на t -й

итерации. Чем меньше $F^l(t)$, тем больше феромона откладывается на ребрах построенного графа-решения $H^l(t)$ и, следовательно, тем больше вероятность выбора этих ребер при построении графа-решения на следующей итерации.

Обозначим $f_{ij}(t)$ суммарное количество феромона, скопившееся на ребре u_k двудольного графа $H_{nm}(t)$, связывающего вершину e_i с вершиной w_j после выполнения второго этапа итерации t .

После того как каждый агент сформировал решение и отложил феромон, на третьем этапе итерации t происходит общее испарение феромона на ребрах двудольного графа $H_{nm}(t)$ в соответствии с формулой $f_{ij}(t) = f_{ij}(t)(1 - \rho)$, где ρ – коэффициент обновления.

После выполнения всех действий на итерации t находится агент с лучшим решением, которое запоминается. Далее осуществляется переход на следующую итерацию.

Рассмотрим теперь конструктивный алгоритм построения муравьем двудольного графа-решения $H^l(t)$ на полном двудольном графе $H_{nm}(t-1)$.

Последовательно (или случайным образом) выбираются вершины $w_j \in W$ двудольного графа $H_{nm}(t-1)$, начиная с первой (стартовой). Для выбранной вершины w_j определяется набор ребер $U_j(t)$, связывающих w_j со всеми вершинами e_i множества E . Для каждого ребра $u_k \in U_j(t)$, связывающего вершину e_i с вершиной w_j , определяется параметр f_{ij} – суммарный уровень феромона на этом ребре.

Вероятность P_k включения ребра $u_k \in U_j(t)$ в формируемый граф-решение $H^l(t)$ определяется соотношением $P_k = f_{ij} / \sum_i (f_{ij})$.

Агент с вероятностью P_k выбирает одно из ребер, которое включается в формируемый граф-решение $H^l(t)$.

Временная сложность этого алгоритма зависит от времени жизни колонии t (число итераций), количества исполнителей n и числа работ m и определяется как $O(l \cdot c^2 \cdot m)$. Далее приведены алгоритм поведения муравьиной колонии и алгоритм муравья.

Алгоритм поведения муравьиной колонии.

1. Задаются число исполнителей – n , число работ – m , начальное количество феромона – Q .

2. Строится полный двудольный граф $H_{nm}(0) = (E \cup W, U(0))$, на ребрах которого отложено начальное количество феромона.

3. За каждым муравьем a_l закрепляется стартовая вершина $w_l \in W$.

4. Задаются число итераций – NT , число муравьев, формирующих независимо друг от друга решения на одной итерации, – NR .

5. $t=1$ (t – номер итерации).

6. $l=1$ (l – номер агента).

7. (Алгоритм муравья). Муравей a_l строит на полном двудольном графе $H_{nm}(t-1) = (E \cup W, U(t-1))$ двудольный граф-решение $H^l(t)$.

8. Рассчитывается оценка $F^l(t)$ построенного двудольного графа-решения $H^l(t)$.

9. Если $l < NR$, то $l = l + 1$ и переход к пункту 7, иначе переход к пункту 10.

10. $l=1$.

11. Муравей a_l откладывает на ребрах полного двудольного графа $H_{nm}(t-1)$, соответствующих ребрам построенного двудольного графа-решения $H^l(t)$, феромон в количестве $\Delta\tau_l(t) = Q / F^l(t)$.

12. Если $l < NR$, то $l = l + 1$ и переход к пункту 11, иначе переход к пункту 13.

13. На третьем этапе итерации t происходит общее испарение феромона на всех ребрах двудольного графа $H_{nm}(t)$ в соответствии с формулой $f_{ij} = f_{ij}(1 - \rho)$, где ρ – коэффициент обновления.

14. Находится агент с лучшим решением $F(t)$, полученным после выполнения t итераций, которое запоминается.

15. Если $t < NT$, то $t = t + 1$ и переход к пункту 6, иначе переход к пункту 16.

16. Конец работы алгоритма.

Рассмотрим теперь конструктивный алгоритм построения муравьем двудольного графа-решения $H^l(t)$ на полном двудольном графе $H_{nm}(t-1)$.

Алгоритм муравья.

1. $s=0$ (s – индекс списка вершин $W(s)$, указывающий на число удаленных вершин из исходного списка W).

2. Формируется исходный список вершин $W(s) = \{w_j / j=1, 2, \dots, m\}$, включающий все вершины множества W , то есть $W(s) = W = \{w_j / j=1, 2, \dots, m\}$.

3. В списке $W(s)$ выбирается вершина w_j , которая закреплена за муравьем a_l , и удаляется из списка. Переход к пункту 5.

4. Случайным образом в списке $W(s)$ выбирается вершина w_j , которая удаляется из списка.

5. $s = s + 1, w_j \in W$.

6. Для выбранной вершины w_j определяется набор ребер U_j , связывающих w_j со всеми вершинами e_i множества $E, |U_j|=n$.

7. Для каждого ребра $u_k \in U_j$, связывающего вершину e_i с вершиной w_j , определяется параметр f_{ij} – суммарный уровень феромона на этом ребре.

8. Для каждого ребра $u_k \in U_j$ по формуле $P_k = f_{ij} / \sum_i (f_{ij})$ рассчитывается вероятность включения ребра $u_k \in U_j$ в формируемый граф-решение H^l .

9. Агент с вероятностью P_k выбирает одно из ребер, которое включается в формируемый граф-решение $H^l(t)$.

10. Если $s < m$, то переход к пункту 4, иначе переход к пункту 11.

11. Двудольный граф-решение $H^l(t)$ полностью сформирован. Конец работы алгоритма муравья.

Разработка программы

Алгоритм решения задачи распределения работы программ на многопроцессорных системах был запрограммирован на языке *c#* на платформе

Windows. При этом все исследования проводились на компьютере типа Intel® Core™ i5 CPU 3.33 GHz и ОЗУ размером 4 Гб.

Язык C# был выбран по нескольким причинам. Во-первых, он позволяет быстро наладить разработку на начальных этапах, что дает возможность в короткие сроки спроектировать программный продукт, который потом можно будет отлаживать с помощью средств Visual Studio.

Огромное количество библиотек с .NET идет в базе, плюс к ним множество свободно доступных библиотек, что покрывает практически все перво-степенные задачи разработки под Windows. Наличие большого количества стандартных типов почти избавляет от библиотек, где базовые типы перепределены. В силу того, что библиотеки C# сравнительно молодые, интерфейсы библиотек, как правило, лучше вписываются в те или иные шаблоны проектирования.

В Visual Studio есть возможность подключения системы контроля версий и работы с ней. Эта особенность сильно помогает в процессе коллективной разработки.

Также данный язык изначально инкапсулирован, что позволяет создавать иерархические схемы наследования и делает возможным использование компонентно-ориентированного подхода.

Для проектирования программы был выбран компонентно-ориентированный подход. Это парадигма программирования, опирающаяся на понятие компонента – независимого модуля программного кода и предназначенная для повторного использования и развертывания, реализующегося в виде множества языковых конструкций.

Кроме того, в платформе .NET реализован компонентно-ориентированный подход, обеспечивающий создание и повторное использование компонентов для любого языка программирования, поддерживаемого платформой.

BaseDisplayer – класс, содержащий в себе виртуальные функции, предназначенные для отрисовки различных статистических элементов на формах (графиках и т.д.). От него наследуются классы DrawBestSolution, ProblemDisplayer, RAlgorithmDisplayer. DrawBestSolutions предназначен для отрисовки графика нахождения лучших решений по итерациям; ProblemDisplayer – для отображения текущей решаемой задачи на форме, предоставляющей возможность подгрузки решения из файла; RAlgorithmDisplayer – для отображения информации о текущих настройках алгоритма.

Программа содержит набор классов, предназначенных для обеспечения работы алгоритма распределения ресурсов.

Класс BaseAlgorithm. Самый верхний в иерархии наследования, общий для любого алгоритма, решающего задачу методом моделирования колонии муравьев. Он абстрактен и содержит определенные виртуальные функции GetProblem и

ResetAllParameters, а также ряд перечислителей с информацией о способе работы алгоритма.

Класс RDistributionAlgorithmParameters содержит основные параметры работы алгоритма: количество итераций, количество муравьев, параметры отложения феромона (начальное, отложение феромона муравьем, параметр затухания, способ отложения феромона). Феромон могут откладывать муравьи, нашедшие лучшее решение, абсолютно все муравьи, муравьи, нашедшие лучшее решение и решения, находящиеся в определенном оценочном диапазоне относительно лучшего.

На рисунке 2 приведена структурная схема программы, описывающая компонентное устройство программы.

Класс RDistributionAlgorithm определяет перечисленные функции самого муравьиного алгоритма и представляет основную программную логику его работы.

Класс BaseProblem содержит некоторые базовые данные о проблеме, решение которой происходит на момент работы алгоритма. Он верхний в иерархии наследования.

Класс RproblemData содержит все необходимые исходные данные для работы алгоритма.

Класс RDistributionAlgorithm наследуется от RproblemData. Это сделано для более упрощенного доступа ко всем исходным данным. Класс содержит реализацию всех механизмов, необходимых для работы алгоритма моделирования поведения колонии муравьев.

Класс BaseFormReader. Базовый класс, предназначенный для считывания информации об исходных данных с формы. Содержит только виртуальные функции.

Класс ProblemConfigurer. Конфигуратор исходных данных. Наследуется от BaseFormReader. Также имеет доступ к RdistributionProblem. Преобразует исходные данные задачи в формат, использующийся внутри программы.

Класс BaseLoader предназначен для загрузки файлов. Содержит только виртуальные функции.

Класс RdistributionLoader. Предназначен для загрузки данных с привязкой к задаче распределения программ в многопроцессорных системах. Реализует возможность подгрузки файлов из формата .xml или сериализованного бинарного файла.

Класс BaseSolution. Представляет базовое построенное решение задачи внутри программы. Содержит только виртуальные функции.

Класс RdistributionSolution. Содержит информацию о полученном решении с привязкой к задаче распределения программ в многопроцессорных системах. Наследуется от BaseSolution.

Как видно из описания, программа содержит несколько полностью виртуальных классов, что позволяет использовать данный шаблон для последующих разработок.

Программная реализация алгоритма

Рассмотрим реализацию алгоритма моделирования поведения колонии муравьев (рис. 3).

Вначале производится формирование исходных данных. На этом шаге или загружается информация из файла, или специальными генераторами создается множество работ с заданным временем выполнения. Также задаются количество итераций работы алгоритма и количество агентов (муравьев).

Задаются параметры, обеспечивающие не-прямой обмен (стигмержи), такие как начальное количество феромона и количество феромона, которое способен отложить каждый отдельный агент системы. Существует ряд параметров, определяющих некоторые системы поведения муравьев. Каждый агент может двигаться, начиная с первой вершины, начиная со своей вершины, также он может выбирать вершины, по которым осуществляется движение в случайном порядке, руководствуясь генератором псевдослучайных чисел.

Затем осуществляется генерация первичной матрицы альтернатив. Матрица содержит значение вероятности, с которой муравей может выбрать какую-то конкретную вершину для своего движения. На первой итерации каждый элемент множества альтернатив приравняется к начальному количеству феромона.

Если не все итерации алгоритма выполнены и не все агенты осуществили процедуру построения решения, выбирается определенный агент, который реализует его построение. Далее оценивается решение. Оценка происходит по выбору максимально загруженного процессора.

Если все агенты на текущей итерации выполнили построение решений, производится изменение множества альтернатив. При этом каждое решение сравнивается с найденным лучшим решением на данной итерации. Чем хуже решение, тем меньшее количество феромона откладывает агент, который данное решение получил. Откладывание феромона в матрицу альтернатив осуществляется по формуле $Pd_i = \frac{Pd_i * Sb_i}{Sc_i}$, $i = \{0, 1, \dots, n\}$, где

Pd – количество откладываемого феромона муравьем на i -й итерации; Sb – оценка лучшего найденного решения; Sc – оценка текущего найденного решения.

Проведение экспериментальных исследований

При проведении экспериментальных исследований необходимо было определить две характеристики: эффективность полученного алгоритма и качество разработанных на основе муравьиной колонии механизмов, которые были применены для решения однородной распределительной задачи.

Для этого была применена процедура формирования контрольных тестовых примеров с уже известными результатами (оптимумом).

Первая исследуемая характеристика – влияние (на полученный результат) управляющих параметров, таких как размер популяции муравьев, количество итераций, параметров, управляющих отложением и испарением феромона.

Для получения достоверных выводов была проведена серия тестов-экспериментов. Временная зависимость разработанного алгоритма определяется

временем жизни колонии t (количество генераций), числом исполнителей n и числом работ m и определяется как $O(t \cdot n^2 \cdot m)$. Тестирование показало, что в 97 % случаев сформированное пространство решений содержит глобальное оптимальное решение.

Тестирование, определяющее сходимость алгоритма, выполнялось следующим образом. Для каждого теста запоминался номер итерации, после которой улучшения оценки не происходило. Проводилась серия из 50 тестов, в которой находились минимальный и максимальный номера итерации. Также выполнялся расчет среднего значения количества итераций, улучшения оценки после которых не происходило. Фактически в каждой серии тестов определялось лучшее решение, которое являлось оптимальным. Тестовые эксперименты показали, что сходжение алгоритма происходит на 120-й генерации при объеме популяции $M = 90$.

Сравнение значений критерия, полученных муравьиным алгоритмом на бенчмарках, у которых оптимум уже известен, показало, что у 80 % тестов полученный результат был оптимальным, у 15 % тестов результаты были на 3 % хуже оптимального, а у 5 % тестов результаты были хуже не более, чем на 2 %. На основании проведенных экспериментальных исследований можно сделать вывод, что разработанный алгоритм позволяет получить результаты на 2–3 % лучше тех, что получены имеющимися алгоритмами [2–10].

В процессе разработки в программу была добавлена возможность построения гистограмм для наилучшего определения итерации, на которой происходит сходимость алгоритма и нахождение лучшего решения. Пример гистограммы для некоторых вариантов решений приведен на рисунке 4.

Для сравнения эффективности были выбраны генетический алгоритм, *селективно-перестановочный алгоритм* (СПА) [4, 6–10], *алгоритм различных наборов работ и машин* (DJMS) [22], *алгоритм моделирования поведения колонии муравьев* (ACO).

Эти алгоритмы были выбраны в силу перспективности и хороших результатов, показанных ими при проведении экспериментов [4, 6–10].

Сравнивался алгоритм адаптивного поведения муравьиной колонии с различными алгоритмами решения однородной распределительной задачи. В качестве оптимальных настроек для алгоритма моделирования поведения колонии муравьев использовались следующие параметры: $N_c = 50$ – размерность колонии, $N_i = 100$ – количество итераций для каждого эксперимента. Работа алгоритма была смоделирована без учета элитных особей. Подобные параметры также использовались для алгоритма селективных перестановок.

Каждый проведенный эксперимент содержал три параметра: m – количество работников, n – количество работ, U – интервал работ, использующийся для генерации списка работ на каждой ите-

Рис. 4. Пример гистограммы (количество работ – 50, количество узлов – 3)

Fig. 4. An example of a histogram (a number of operations – 50, a number of nodes – 3)

рации. Для каждого набора работ использовались 100 задач. Суммарно для сравнения методов ACO и DJMS было проанализировано 1 900 решений.

Для оценки решений использовалась нижняя граница (Lower bound):

$$L = \max \left\{ \left[\frac{1}{m} * \sum_{j=1, \dots, n} p_j \right], p_1, p_m + p_{m+1} \right\}.$$

Сравним эффективность метода моделирования поведения адаптивной колонии с эффективностью методов DJMS и LPT (табл. 1). Результаты экспериментов метода DJMS взяты из [22]. Для каждого метода в каждой строчке таблицы показано количество оптимальных решений из проведенных 100 экспериментов.

Таблица 1

Сравнение эффективности методов

Table 1

Method effectiveness comparison

n	m	U	LPT	DJMS	ACO
6	3	[1, 20]	62	63	99
9	3	[1, 20]	35	79	98
15	3	[1, 20]	56	99	99
6	3	[20, 50]	45	45	99
9	3	[20, 50]	11	21	100
15	3	[20, 50]	26	48	99
8	4	[1, 20]	66	68	100
12	4	[1, 20]	34	76	98
20	4	[1, 20]	56	97	99
8	4	[20, 50]	40	40	97
12	4	[20, 50]	10	14	98
20	4	[20, 50]	24	45	100
10	5	[1, 20]	53	53	99
12	5	[1, 20]	48	79	100
25	5	[1, 20]	23	100	98
10	5	[20, 50]	47	22	97
12	5	[20, 50]	27	11	98
25	5	[20, 50]	9	31	100

Из таблицы 1 видно, что алгоритм DJMS опережает стандартный LPT. Стоит отметить, что этот

алгоритм использует связку концепций LPT и MF. Очевидно, что такая связка дает более хороший результат. В свою очередь, алгоритм АСО опережает DJMS, что обусловлено использованием вероятностного подхода и выцветанием отдельных альтернативных выборов решений.

Сравним алгоритмы АСО и СПА.

Для исследования эффективности этих алгоритмов проведены вычислительные эксперименты при разных значениях параметров задачи. В задаче приводятся такие же параметры, как и в предыдущем опыте, за исключением диапазона работ. В качестве параметра, характеризующего ресурсно-точные свойства, выбран $P_{опт}$ – доля оптимальных решений.

Таблица 2 позволяет сравнить алгоритмы СПА и АСО. В данном случае применялся алгоритм селективных перестановок, использующий одинарные перестановки. Данные для СПА взяты из [23].

Таблица 2

Сравнение СПА и АСО

Table 2

Comparison of selective-permutation algorithm and ACO

n	m	U	СПА	АСО
5	33	[35,65]	95	99
5	33	[15,85]	80	98
5	63	[35,65]	100	99
5	63	[15,85]	100	100
6	48	[25,75]	72	98
7	33	[35,65]	79	99
7	33	[15,85]	57	97
7	63	[35,65]	42	98
7	63	[15,85]	46	98

Алгоритм муравьиной колонии показал эффективность 98,6 %, в то время как эффективность алгоритма ListScheduling составила 21,9 %, ThresholdHeuristic – 79,1 % и Scattersearch – 97,7 % (результаты были взяты из [24]).

Стоит отметить, что алгоритм моделирования поведения колонии муравьев показывает хорошие результаты на абсолютно различных диапазонах мощностей работ, что связано с имитацией откладывания феромона во время работы алгоритма.

По сравнению с алгоритмом DJMS метод показывает гораздо более хороший результат благодаря использованию более мощной эвристики.

Методы СПА и АСО показывают приблизительно равные результаты с преимуществом метода АСО около 2,5 %. Также стоит отметить, что метод АСО показывает более хороший результат на большом диапазоне работ. В целом данный метод имеет преимущество.

Заключение

В работе предложена новая парадигма комбинаторной оптимизации, которая базируется на моде-

лировании адаптивного поведения муравьиной колонии и представляет графическое решение однородной распределительной задачи в виде двудольного графа. Также для решения однородных распределительных задач предложены новые механизмы. Муравьем на графе поиска решений строится двудольный граф в отличие от канонической парадигмы муравьиного алгоритма. В оптимизационных задачах, допускающих представление решения в виде двудольных графов, этот способ поиска рациональных решений является наиболее эффективным.

В частности, путем незначительной модификации конструктивного алгоритма муравья разработанный алгоритм, решающий однородные распределительные задачи, может также решать задачу о назначениях.

Подбор наиболее адекватных управляющих параметров может стать источником усовершенствования данного алгоритма. Решением другой проблемы, которая может улучшить критерий качества, является установление очередности рассматриваемых работ. Улучшение качества можно получить с помощью интеграции различных методов роевого интеллекта.

Работа выполнена при финансовой поддержке РФФИ, грант № 17-07-00997.

Литература

1. Лазарев А.А., Гафаров Е.Р. Теория расписаний. Задачи и алгоритмы. М.: Изд-во МГУ, 2011. 188 с.
2. Майоров С.А., Новиков Г.И. Вычислительные комплексы, системы и сети. СПб: Энергоатомиздат, 1987. 329 с.
3. Романовский И.В. Алгоритмы решения экстремальных задач. М.: Наука, 1977. 156 с.
4. Нейдорф Р.А., Жикюлин А.А. Быстродействующая модификация алгоритма оптимизации решения // AIS-IT'14: тр. Конгресса по интелект. сист. и информ. технол. В 4-х т. М.: Физматлит, 2010. Т. 1. С. 15–22.
5. Кобак В.Г., Будиловский Д.М. Сравнительный анализ приближенных алгоритмов решения минимаксной задачи для однородных приборов // Вестн. Донск. гос. техн. ун-та. 2006. № 4. С. 327–334.
6. Нейдорф Р.А., Жикюлин А.А. Исследование вариантов модификации приближенных алгоритмов решения однородных распределительных задач, повышающих их эффективность // Инновация, экология и ресурсосберегающие технологии (ИнЭРТ-2012): тр. X Междунар. науч.-техн. форума. Ростов н/Д: Изд-во ДГТУ, 2012. С. 370–375.
7. Нейдорф Р.А., Жикюлин А.А. Селективно-перестановочный метод приближенного решения однородной распределительной задачи. Комбинаторные перестановки // Изв. ЮФУ. Сер. Технич. науки. 2013. № 7. С. 167–172.
8. Кобак В.Г. Эффективные методы решения однородных распределительных задач на основе минимаксного критерия. Ростов н/Д: Изд-во ДГТУ, 2013. 99 с.
9. Титов Д.В. Методы повышения эффективности алгоритмов решения распределительных минимаксных задач в однородных системах: дисс. ... канд. технич. наук. Ростов н/Д: Изд-во ДГТУ, 2010. 179 с.
10. Жикюлин А.А. Методы высокоэффективной ресурсной модификации алгоритма Романовского для решения однородных распределительных задач. дисс. ... канд. технич. наук. Ростов н/Д: Изд-во ДГТУ, 2014. 156 с.

11. Davidović T., Šelmić M., Teodorović D., Ramljak D. Bee colony optimization for scheduling independent tasks to identical processors. *Jour. of Heuristics* 2012, vol. 18, iss. 4, pp. 549–569.
12. Demirel T., Özkır V., Demirel N.Ç., Taşdelen B. A Genetic Algorithm approach for minimizing total tardiness in parallel machine scheduling problems. *Proc. World Congress on Engineering*, 2011, vol. II, pp. 267–281.
13. Raghavendral B.V., Murthy A.N. Workload balancing in identical parallel machine scheduling while planning in flexible manufacturing system using genetic algorithm. *ARNP Jour. of Engineering and Applied Sciences*, 2011, vol. 6, pp. 113–126.
14. Aslam M.U., Nasr M.M., Al-Harkan I.M. Scheduling of jobs on identical parallel machines to minimize total flow time subject to optimal makespan. *Applied Mechanics and Materials*, 2014, vol. 529, pp. 390–395.
15. Akylol D.E. Identical parallel machine scheduling with dynamical networks using time-varying penalty parameters. *Multiprocessor Scheduling, Theory and Applications*, I-Tech Education and Publishing, 2007, pp. 293–315.
16. Dorigo M. and Stützle T. *Ant colony optimization*. MIT Press, Cambridge, MA, 2004, 234 p.
17. Лебедев О.Б. Модели адаптивного поведения муравьиной колонии в задачах проектирования: монография. Таганрог: Изд-во ЮФУ, 2013. 199 с.
18. Raidl G.R. *A unified view on hybrid metaheuristics*. LNCS. Springer-Verlag, 2006, pp. 1–12.
19. Лебедев Б.К., Лебедев О.Б. Моделирование адаптивного поведения муравьиной колонии при поиске решений, интерпретируемых деревьями // *Изв. ЮФУ*. 2012. № 7. С. 27–35.
20. Лебедев В.Б., Лебедев О.Б. Роевой интеллект на основе интеграции моделей адаптивного поведения муравьиной и пчелиной колоний // *Изв. ЮФУ*. 2013. № 7. С. 41–47.
21. Курейчик В.М., Лебедев Б.К., Лебедев О.Б. Разбиение на основе моделирования адаптивного поведения биологических систем // *Нейрокомпьютеры: разработка, применение*. 2010. № 2. С. 28–34.
22. Лебедев Б.К., Лебедев О.Б., Нацкевич А.Н. Решение однородной распределительной задачи методом моделирования адаптивного поведения колонии муравьев. Свид. о гос. регистр. прогр. для ЭВМ № 2017611229 от 27.01.2017.
23. A new algorithm to minimize makespan on identical parallel machines. URL: <http://www.pomsmeetings.org/confpapers/025/025-0543.pdf> (дата обращения: 09.04.2017).
24. Iori M., Martello S. Scatter search algorithms for identical parallel machine scheduling problems. URL: http://www.springer.com/cda/content/document/cda_downloaddocument/9783540789840-c1.pdf?SGWID=0-0-45-552307-p173814816 (дата обращения: 09.04.2017).

Software & Systems

DOI: 10.15827/0236-235X.118.217-226

Received 10.04.17

2017, vol. 30, no. 2, pp. 217–226

A HOMOGENEOUS DISTRIBUTION PROBLEM BASED ON ANT COLONY ADAPTIVE BEHAVIOOR MODELS

B.K. Lebedev¹, *Dr.Sc. (Engineering), Professor, lebedev.b.k@gmail.com*

O.B. Lebedev¹, *Ph.D. (Engineering), Associate Professor, lebedev.ob@mail.ru*

E.M. Lebedeva¹, *Postgraduate Student, lebedeva.el.m@mail.ru*

¹ *Institute of Computer Technology and Information Security, Southern Federal University, Engels St. 1, Taganrog, 347928, Russian Federation*

Abstract. The paper proposes a solution of a homogeneous distribution problem. It gives the problem statement, describes the main groups of algorithms to solve it (approximate and exact) and their advantages and disadvantages. The paper proposes a new paradigm of combinatorial optimization, which is based on modeling the adaptive behavior of an ant colony.

The solution of a homogeneous distribution problem is its graphical representation as a bipartite graph. New decision mechanisms were proposed to solve these problems. The basis of metaheuristics of an ant colony algorithm is a combination of two techniques. The first basic technique is to perform the search for the best solution using an ant colony adaptive behavior. An ant builds a specific solution using a built-in procedure, which is based on a constructive algorithm. A bipartite graph that is built on the solution search graph is the main difference of the proposed ant algorithm from the existing canonical paradigm.

When finding optimal solutions for optimization problems, which allow presenting solutions in the form of bipartite graphs, this approach will be fairly effective.

The conducted researches showed that the ant algorithm gives more qualitative solutions in comparison with the known algorithms. Comparing known and developed algorithms, we can say that the results improved by 3–4 %.

Keywords: assignment problem, bipartite graph, optimization, swarm intelligence, ant colony, adaptive behavior, homogeneous distribution problem.

Acknowledgements. The work has been financially supported by the Russian Foundation for Basic Research, grant no. 17-07-00997.

References

1. Lazarev A.A., Gafarov E.R. *Teoriya raspisany. Zadachi i algoritmy* [The Scheduling Theory. Tasks and Algorithms]. Moscow, Lomonosov MSU Publ., 2011.
2. Mayorov S.A., Novikov G.I. *Vychislitelnye komplekxy, sistemy i seti* [Computer Complexes, Systems and Networks]. St. Petersburg, Energoatomizdat Publ., 1987.

3. Romanovsky I.V. *Algoritmy resheniya ekstremal'nykh zadach* [Solution Algorithms for Extremum Problems]. Moscow, Nauka Publ., 1977.
4. Neydorf R.A., Zhikulin A.A. A Quick-operating modification of the Solution optimization algorithm. *AIS-IT'14: tr. Kongr. po intellekt. sist. i inform. tekhnol.* [Proc. Congr. on Intelligent Systems and IT]. Moscow, Fizmatlit Publ., 2010, vol. 1, pp. 15–22 (in Russ.).
5. Kobak V.G., Budilovsky D.M. A comparative analysis of approximate algorithms to solve a minimax problem for homogeneous instruments. *Vestn. DGTU* [Vestnik of DSTU]. 2006, no. 4, pp. 327–334 (in Russ.).
6. Neydorf R.A., Zhikulin A.A. A research on variations of approximate algorithms for solving a homogeneous distribution problems, which increase their effectiveness. *Innovatsiya, ekologiya i resursoberegayushchie tekhnologii (InERT-2012): tr. X Mezhdunar. nauch.-tekhn. foruma* [Proc. 10th Int. Science and Technical Forum on Innovation, Ecology and Resource-Saving Technologies (InERT-2012)]. Rostov-on-Don, DGTU Publ., 2012, pp. 370–375 (in Russ.).
7. Neydorf R.A., Zhikulin A.A. Selective-permutation method for finding an approximate solution of the open shop scheduling problem. Combinational permutations. *Izv. YuFU. Ser. Tekhnich. nauki* [Izvestiya SFedU. Engineering Sciences]. 2013, no. 7, pp. 167–172 (in Russ.).
8. Kobak V.G. *Effektivnye metody resheniya odnorodnykh raspredelitel'nykh zadach na osnove minimaksnogo kriteriya* [Efficient Methods for Solving Homogeneous Distribution Problems Based on a Minimax Criterion]. Rostov-on-Don, DGTU Publ., 2013, 99 p.
9. Titov D.V. *Metody povysheniya effektivnosti algoritmov resheniya raspredelitel'nykh minimaksnyykh zadach v odnorodnykh sistemakh* [Improving Algorithm Practices of Distribution Minimax Problems in Homogeneous Systems]. PhD thesis. Rostov-on-Don, DGTU Publ., 2010.
10. Zhikulin A.A. *Metody vysokoeffektivnoy resursnoy modifikatsii algoritma Romanovskogo dlya resheniya odnorodnykh raspredelitel'nykh zadach* [The Methods of Highly Efficient Resource Modification of Romanovsky Algorithm to Solve Homogeneous Distribution Problems]. PhD thesis. Rostov-on-Don, DGTU Publ., 2014.
11. Davidović T., Šelmić M., Teodorović D., Ramljak D. Bee colony optimization for scheduling independent tasks to identical processors. *Jour. of Heuristics*. 2012, vol. 18, iss. 4, pp. 549–569.
12. Demirel T., Özkır V., Demirel N.Ç., Taşdelen B. A Genetic Algorithm approach for minimizing total tardiness in parallel machine scheduling problems. *Proc. World Congr. on Engineering*. 2011, vol. 2.
13. Raghavendra B.V., Murthy A.N. Workload balancing in identical parallel machine scheduling while planning in flexible manufacturing system using genetic algorithm. *ARNP Jour. of Engineering and Applied Sciences*. 2011, vol. 6.
14. Aslam M.U., Nasr M.M., Al-Harkan I.M. Scheduling of jobs on identical parallel machines to minimize total flow time subject to optimal makespan. *Applied Mechanics and Materials*. 2014, vol. 529, pp. 390–395.
15. Akyol D.E. Identical parallel machine scheduling with dynamical networks using time-varying penalty parameters. *Multiprocessor Scheduling, Theory and Applications*. I-Tech Education and Publ., 2007, pp. 293–315.
16. Dorigo M., Stützle T. *Ant Colony Optimization*. MIT Press, Cambridge, MA, 2004.
17. Lebedev O.B. *Modeli adaptivnogo povedeniya muravinoy kolonii v zadachakh proektirovaniya* [Ant Colony Adaptive Behavior Models in Design Problems]. Monograph. Taganrog, SFedU Publ., 2013, 199 p.
18. Raidl G.R. A unified view on hybrid metaheuristics. *LNCS*. Springer-Verlag Publ., 2006, pp. 1–12.
19. Lebedev B.K., Lebedev O.B. Modelling of an ant colony adaptive behaviour by search of the decisions interpreted by trees. *Izv. YuFU. Ser. Tekhnich. nauki* [Izvestiya SFedU. Engineering Sciences]. 2012, no. 7, pp. 27–35 (in Russ.).
20. Lebedev V.B., Lebedev O.B. Swarm intelligence on the basis of the adaptive behaviour models integration of the ant and bee colonies. *Izv. YuFU. Ser. Tekhnich. nauki* [Izvestiya SFedU. Engineering Sciences]. 2013, no. 7, pp. 41–47 (in Russ.).
21. Kureychik V.M., Lebedev B.K., Lebedev O.B. Partitioning on the basis of modeling of adaptive behaviour of biological systems. *Neyrokomp'yutery: razrabotka, primenenie* [Neurocomputers: Design and Applications]. 2010, no. 2, pp. 28–34 (in Russ.).
22. Lebedev B.K., Lebedev O.B., Natskevich A.N. *Reshenie odnorodnoy raspredelitel'noy zadachi metodom modelirovaniya adaptivnogo povedeniya kolonii muravev* [Solving a Homogeneous Distribution Problem by the Method of Modeling Ant Colony Adaptive Behavior]. State Registration Certificate of a computer program no. 2017611229. 2017.
23. *A New Algorithm to Minimize Makespan on Identical Parallel Machines*. Available at: <http://www.pomsmeetings.org/confpapers/025/025-0543.pdf> (accessed April 9, 2017).
24. Iori M., Martello S. *Scatter Search Algorithms for Identical Parallel Machine Scheduling Problems*. Available at: http://www.springer.com/cda/content/document/cda_downloaddocument/9783540789840-c1.pdf?SGWID=0-0-45-552307-p173814816 (accessed April 9, 2017).

Примеры библиографического описания статьи

1. Лебедев Б.К., Лебедев О.Б., Лебедева Е.М. Однородная распределительная задача на основе моделей адаптивного поведения муравьиной колонии // Программные продукты и системы. 2017. Т. 30. № 2. С. 217–226; DOI: 10.15827/0236-235X.118.217-226.
2. Lebedev B.K., Lebedev O.B., Lebedeva E.M. A homogeneous distribution problem based on ant colony adaptive behavior models. *Programmnye produkty i sistemy* [Software & Systems]. 2017, vol. 30, no. 2, pp. 217–226 (in Russ.); DOI: 10.15827/0236-235X.118.217-226.