ИЗВЕСТИЯ РАН. ТЕОРИЯ И СИСТЕМЫ УПРАВЛЕНИЯ, 2013, № 6, с. 87–96

СИСТЕМНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ

УДК 517.977.5

АЛГОРИТМ ПОСТРОЕНИЯ ОДНОПРИБОРНЫХ РАСПИСАНИЙ, ОСНОВАННЫЙ НА СХЕМЕ МУРАВЬИНЫХ КОЛОНИЙ

© 2013 г. В. А. Костенко, А. В. Плакунов

Москва, МГУ

Поступила в редакцию 29.04.13 г., после доработки 26.06.13 г.

Приведена математическая постановка задачи построения расписания обменов по каналу с централизованным управлением. Данная задача возникает при проектировании информационно-управляющих систем реального времени. Проведен анализ возможных подходов к построению алгоритмов, основанных на схеме муравьиных колоний, и приведены результаты экспериментального сравнения предложенного алгоритма с жадными алгоритмами.

DOI: 10.7868/S0002338813060085

Введение. В бортовых системах реального времени широко используется архитектура, основанная на каналах с централизованным управлением. Примерами каналов с централизованным управлением являются MIL STD-1553B (МКИО ГОСТ Р52070-2003) [1], STANAG 3910 [2], FC-AE-1553 [3].

Канал обеспечивает обмен данными между устройствами, присоединенными к каналу (далее – оконечные устройства). Обмен представляет собой последовательность передач прикладных и служебных данных между оконечными устройствами. Время начала каждого обмена определяет расписание, которое строят заранее и которое не меняется в ходе функционирования бортовой системы. Расписание исполняется контроллером канала, который является одним из оконечных устройств. Только контроллер канала может инициировать обмен данными; другие оконечные устройства выполняют команды, отданные контроллером (схема ведущий–подчиненный).

Задача построения расписания обменов по каналу с централизованным управлением относится к классу задач построения одноприборных расписаний без прерываний и известна в теории расписаний как задача о выборе максимального числа совместимых заявок, которая является *NP*-трудной. В отличие от задач о выборе максимального числа совместимых заявок, рассматриваемых в теории расписаний, в задаче построения обменов по каналу с централизованным управлением накладываются дополнительные ограничения на корректность расписания, которые обусловлены особенностями программных и аппаратных средств бортовой системы [4–6].

Основной проблемой при использовании алгоритмов, основанных на жадных стратегиях и стратегиях ограниченного перебора, является их настройка на решение частных задач (наложены ограничения на возможные значения входных данных) [7, 8]. Это связано с проблемой формирования ограничений на исходные данные таким образом, чтобы “четко” выделить частную задачу, для которой алгоритм будет гарантировано находить решение с приемлемой точностью и сложностью. В большинстве случаев возможно получение лишь статистических оценок точности и сложности алгоритма. Муравьиные алгоритмы [9, 10] позволяют автоматически настраиваться на пример задачи (заданы конкретные значения исходных данных) путем дополнительной разметки исходных данных, которая используется для построения решения на каждой итерации алгоритма и уточняется по мере увеличения числа итераций. Другими словами, при использовании муравьиных алгоритмов не возникает проблема “четкого” выделения частной задачи.

Алгоритмы, основанные на использовании схемы муравьиных колоний, были успешно применимы для решения таких комбинаторных задач, как квадратичная задача о назначениях [11], задача упаковки в контейнеры [12], задачи построения расписаний [13–15]. В данной работе предлагается алгоритм, основанный на использовании схемы муравьиных колоний, для решения задачи построения расписания обменов по каналу с централизованным управлением.

В первом разделе данной работы рассматривается задача построения расписания обменов по каналу с централизованным управлением, во втором разделе приведена общая схема муравьиных алгоритмов и сформулированы задачи, которые надо решить при построении муравьиных

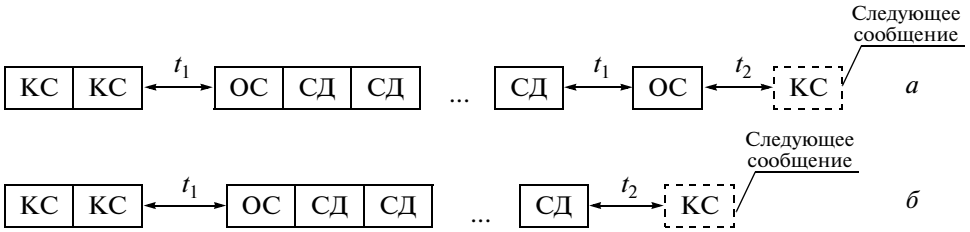

Рис. 1. Схемы передачи данных от оконечного устройства: *а* – другому оконечному устройству, *б* – нескольким оконечным устройствам

алгоритмов для решения задач построения расписаний. В третьем разделе приведено решение этих задач для алгоритма построения расписания обменов по каналу с централизованным управлением, в четвертом разделе приведены результаты исследования свойств предложенного алгоритма.

1. Задача построения расписания обменов. Система состоит из единственного канала обмена (шины) и ограниченного набора оконечных устройств, подключенных к этому каналу, которые являются источниками/приемниками информации, передаваемой по шине. Одно из оконечных устройств назначается контроллером шины, который управляет обменом информации и осуществляет контроль состояния других оконечных устройств в соответствии со статическим расписанием. Эти оконечные устройства лишь выполняют адресованные им команды контроллера. Обмен информацией осуществляется асинхронно путем поочередной передачи информации по принципу "команда–ответ". Информация передается в виде сообщений, которые могут состоять из командных слов, слов данных и ответных слов.

Рассмотрим пример работы шины в соответствии со стандартом MIL STD [1]. Стандарт допускает основные и групповые форматы сообщений. Форматы основных сообщений используются для передачи информации одному из оконечных устройств и предусматривают выдачу им ответного слова. Форматы групповых сообщений применяются для передачи информации одновременно нескольким оконечным устройствам без выдачи ими ответных слов. Последняя группа форматов обеспечивает понижение загрузки шины, но при этом снижается надежность. Если требуется подтверждение факта приема оконечным устройством группового сообщения, то контроллер в этом случае (используя формат основного сообщения) может послать оконечному устройству команды "передать ответное слово" или "передать последнюю команду", и факт приема группового сообщения может быть установлен контроллером путем анализа в ответном слове признака "принято групповое сообщение". Каждый формат определяет количество и порядок следования командных слов, ответных слов и слов данных. Ниже приведены примеры форматов одиночного (*а*) и группового (*б*) сообщений (рис. 1). Здесь КС – командное слово, ОС – ответное слово, СД – слово данных, t_1, t_2 – паузы.

Контроллер должен без паузы передать команду обмена данными с адресом оконечного устройства **А** на прием данных и команду обмена данными с адресом оконечного устройства **Б** на передачу данных. Оконечное устройство **Б** после установления факта достоверности принятой команды должно передать без пауз ответное слово и указанное в команде количество слов данных. Оконечное устройство **А** после установления факта достоверности адресованной ему информации должно передать ответное слово.

Контроллер должен передать без паузы групповую команду обмена данными на прием данных и команду обмена данными с адресом одного оконечного устройства на передачу данных. Это оконечное устройство после установления факта достоверности принятого командного слова должно передать без пауз ответное слово и указанное в команде количество слов данных.

В качестве исходных данных для задачи построения расписания обменов задается множество периодических сообщений $SM = \{M_i = \langle t_i, r_i \rangle\}$, где $i = \overline{1,n}$, t_i – время передачи сообщения, r_i – частота передачи сообщения. Период передачи сообщения обозначим за $\tau_i = 1/r_i$. Для каждого сообщения формируется множество соответствующих ему работ $J^i = \{j = \langle t_j, s_j, f_j \rangle\}$, где $j = \overline{1,n_i}$, $n_i = \lceil r_{scl}/\tau_i \rceil$ – количество экземпляров сообщения, которые надо передать в интервале планирования, r_{scl} – длительность интервала планирования, $s_j = (j-1)\cdot\tau_i$; $f_j = j\cdot\tau_i$ – соответственно левая и правая границы директивного интервала сообщения j. Для некоторых сообщений могут

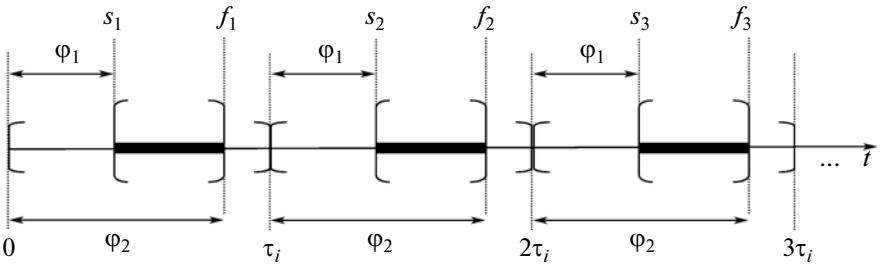

Рис. 2. Директивные интервалы работ

быть заданы левый и правый фазовые сдвиги, обозначаемые φ_1 и φ_2 соответственно, которые сужают директивный интервал сообщения, вычисляемый в зависимости от номера сообщения и периода его передачи (рис. 2). Множество работ определяется как объединение множества работ соответствующих заданным сообщениям

$$J = \bigcup_{i=1}^{n} J^i.$$

Шина может рассматриваться как одноприборное устройство, обслуживающее исходно заданный набор работ $J = \{j\}_{j=1}^{N_j}$, которые должны выполняться без прерываний. Для каждой работы заданы $t_j > 0$ – время выполнения, $[s_j, f_j)$ – директивный интервал, и верно условие $f_j - s_j \geq t_j$. Расписание представляет собой упорядоченное по критерию момента старта работы множество $H = \{j_k, s_j^*\}_{k=1}^{N_k}$, $j \in J$, где k – порядковый номер j-й работы в расписании, N_k – количество работ, s_j^* – момент старта j-й работы в расписании H, $f_j^* = s_j^* + t_j$ – момент завершения j-й работы.

Множество корректных расписаний H^* определим набором основных ограничений:

$$g_1\colon (\forall j \in H) \Rightarrow ((s_j^* \geq s_j) \wedge (f_j^* \leq f_j)),$$

$$g_2\colon (\forall j \in H) \Rightarrow (f_j^* - s_j^* = t_j),$$

$$g_3\colon (\forall (j,l) \in H, j \neq l) \Rightarrow (((s_j^* < s_l^*) \vee (s_j^* \geq f_l^*)) \wedge ((f_j^* \leq s_l^*) \vee (f_j^* > f_l^*))).$$

Ограничения g_1, g_2, g_3 являются обязательными для одноприборных расписаний без прерываний и соответственно означают:

1) интервал выполнения каждой работы располагается в рамках ее директивного интервала;

2) не допустимы прерывания;

3) интервалы выполнения работ не пересекаются.

Для расписания обменов по каналу с централизованным управлением присутствуют дополнительные к основным ограничения на корректность расписания, которые обусловлены особенностями аппаратных и программных средств конкретной системы реального времени. Для схемы организации обменов с подциклами (интервал планирования разбивается на отрезки равной длины – подциклы, см. рис. 3) возможны следующие дополнительные ограничения g_i:

4) в каждом подцикле может находиться не более одной цепочки работ и работы в цепочке следуют друг за другом без пауз;

5) время выполнения работ не должно пересекать границы подцикла;

6) время начала цепочки работ относительно начала соответствующего подцикла не должно быть меньше заданного значения;

7) в конце подцикла должен быть зарезервирован интервал времени, длительность которого не меньше, чем заданная доля длительности подцикла;

8) число работ в цепочке не должно превышать заданного значения;

9) сдвиг работы “вправо” по временной оси на время, не превышающее значение, равное заданному проценту от интервала “время начала выполнения работы минус время начала цепоч-

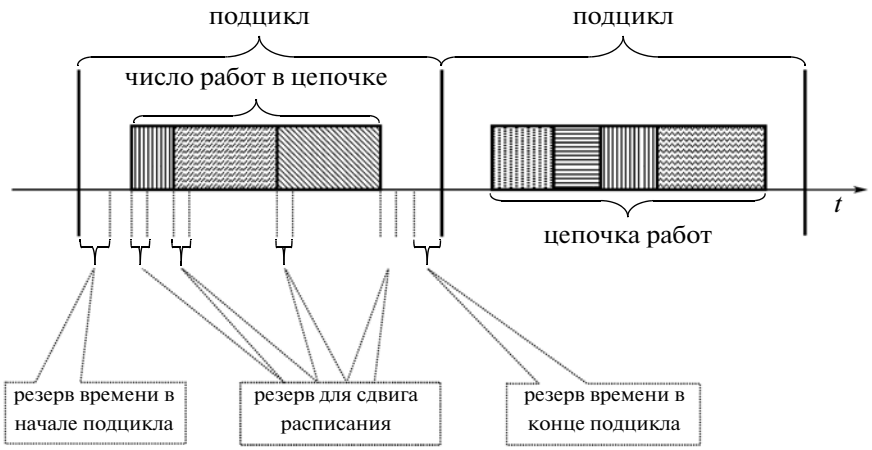

Рис. 3. Циклическая схема обмена данными

ки", не должен приводить к нарушению директивного времени завершения работы или требования к минимальному резерву времени в конце подцикла;

10) последовательность номеров сообщений, которым соответствуют работы в каждой из цепочек, должна являться подпоследовательностью некоторой глобальной последовательности номеров сообщений, записанной в памяти контроллера шины.

Ограничения 4)–10) типичны для авиационных и навигационных систем для схемы организации обменов с подциклами. Такие дополнительные ограничения будем обозначать как $g_i(H, x_i)$, $i = \overline{4,10}$, где x_i – численное значение требования, например, для ограничения 8) это максимально допустимое число работ в цепочке.

Задача построения статических расписаний обменов по шине с централизованным управлением формулируется следующим образом:

$$\max_{H \in H^*} \left\{ |H| \middle| g_i(H, x_i),\ i = \overline{4,10} \right\}.$$

Задача без дополнительных ограничений $\max\limits_{H \in H^*} |H|$ известна в теории расписаний как задача о выборе максимального числа совместимых заявок и является *NP*-трудной. Однако есть частные задачи этой задачи, которые являются полиномиально разрешимыми. Например, для частной задачи

$$\max_{H \in H^*} \left\{ |H| \middle| t_j = f_j - s_j \quad \forall j \right\}$$

известен оптимальный жадный алгоритм сложности $O(n \cdot \ln n)$ [16].

2. Общая схема муравьиных алгоритмов. Интересным результатом кооперативного поведения биологических муравьев является нахождение кратчайшего маршрута от источника пищи к гнезду. Рассмотрим, как кооперативное поведение биологических муравьев позволяет найти кратчайший маршрут к источнику пищи. Пусть гнездо соединено с источником пищи двумя ребрами разной длины. Муравьи при прохождении маршрута откладывают феромоны, и чем больше муравьев прошло по ребру, тем выше концентрация феромонов на этом ребре. Чем выше концентрация феромонов на ребре, тем выше вероятность, что муравей пойдет по нему. Изначально вероятности выбора маршрутов равны. Муравьи, выбравшие короткое ребро, возвращаются быстрее. Количество феромонов на коротком ребре растет быстрей и, следовательно, повышается вероятность его выбора, т.е. через некоторое время по нему будет проходить большинство муравьев. Со временем феромоны испаряются, что позволяет муравьям адаптировать поведение под изменение внешней среды.

Общую схему работы муравьиных алгоритмов можно представить в следующем виде [9, 10]:

1) задание начального количества феромона на ребрах графа, количества и начального положения муравьев;

2) построение муравьями путей (каждый муравей строит путь независимо от остальных);

3) вычисление целевой функции для каждого пути;

4) обновление количества феромона на ребрах;

5) если условие останова не выполнено, то переход к п. 2.

Муравьи строят маршруты, последовательно переходя от вершины к вершине, руководствуясь при этом определенным алгоритмом для определения списка вершин, доступных на данном этапе (например, при решении задачи коммивояжера используется табу-список недоступных вершин, в который добавляются пройденные муравьем вершины). Выбор очередной вершины зависит от количества феромона и значения эвристической функции на ребрах, ведущих из текущей вершины в вершины, доступные на данный момент. Эти значения определяют вероятность перехода в ту или иную вершину, после чего очередная вершина выбирается по правилу рулетки. В конце каждой итерации для каждого найденного маршрута вычисляется значение целевой функции. Оно используется для вычисления количества феромона, добавляемого на ребра, входящие в данный маршрут.

Операция построение муравьем пути. Муравей строит путь, переходя из одной вершины в другую. Пройденные муравьем вершины добавляются в табу-список (память муравья), чтобы избежать повторного их посещения. Вероятность перехода муравья из i-й вершины в j-ю зависит от количества феромона на данном ребре, значения локальной эвристической функции на ребре и состояния табу-списка. Вероятность перехода k-го муравья из i-й вершины в j-ю на t-й итерации алгоритма рассчитывается по следующей формуле:

$$P_{ij,k}(t) = \begin{cases} \tau_{ij}^{\alpha}(t)\,\eta_{ij}^{\beta}(t) \Big/ \sum\limits_{l \in J_k} \tau_{il}^{\alpha}(t)\,\eta_{il}^{\beta}(t), & j \notin L_k, \\ 0, & j \in L_k. \end{cases}$$

Здесь $\tau_{ij}(t)$ – количество феромона на ребре (i, j), $\eta_{ij}(t)$ – значение эвристической функции на ребре (i, j), $\alpha \geq 0$ и $\beta \geq 0$ – параметры алгоритма, определяющие важность феромонного следа и эвристической функции, L_k – множество вершин, включенных в табу-список муравья k.

Операция обновление количества феромона на ребрах. После того, как все муравьи завершили построение путей, обновляется количество феромона на ребрах:

$$\tau_{ij}(t+1) = (1-p)\,\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij,k}(t); \quad \Delta\tau_{ij,k}(t) = \begin{cases} F(T_k(t)), & (i,j) \in T_k(t), \\ 0, & (i,j) \notin T_k(t). \end{cases}$$

Здесь $T_k(t)$ – путь, построенный k-м муравьем, а $F(T)$ – целевая функция, определяющая качество пути, m – количество муравьев, $p \in [0, 1]$ – коэффициент испарения феромонов. Испарение феромонов вводится для избегания попадания алгоритма в локальный оптимум, когда первый найденный путь с относительно хорошим значением целевой функции становится единственно значимым.

Основными задачами, которые необходимо решить для того, чтобы использовать муравьиный алгоритм для решения задачи построения расписания, являются:

1) сведение задачи построения расписания к задаче нахождения на графе пути, обладающего определенными свойствами;

2) задание эвристической функции на ребрах графа;

3) определение алгоритма построения маршрута муравьем (например, правила формирования табу-списка вершин);

3. Алгоритм построения расписания обменов. 3.1. Общая схема работы алгоритма. Для применения муравьиного алгоритма к задаче построения расписания обменов разобьем эту задачу на две подзадачи следующим образом:

1) определение последовательности размещения работ в расписание;

2) определение места в расписании для каждой работы. Работы выбираются в соответствии с последовательностью, построенной в п. 1).

Такое разбиение обусловлено тем, что оно позволяет применить муравьиный алгоритм к подзадаче 1). Для размещения работ в расписание в соответствии с построенной муравьиным алгоритмом последовательностью будет использоваться жадный алгоритм, который опишем ниже.

Общая схема алгоритма имеет следующий вид:

1) построение муравьями путей в графе;

2) применение алгоритма размещения работ в расписание, на вход которому подаются последовательности вершин, построенные муравьиным алгоритмом;

3) вычисление целевой функции;

4) обновление количества феромона на ребрах графа;

5) если условие останова не выполнено, переход к п. 1).

Преимуществом данной схемы является то, что для построения расписания по заданной последовательности работ в п. 2) может применяться любой алгоритм без существенных изменений в остальных пунктах.

3.2. Сведение задачи построения расписания к задаче нахождения на графе пути. Первый способ сведения задачи основан на соответствии вершин графа работам в задаче. Сопоставим каждой работе A_i из J вершину r_i. Также введем специальную вершину O, с которой будет начинаться маршрут каждого муравья. Получим граф $G = \langle V, E \rangle$, где $V = \{O, r_1, r_2, \ldots, r_n\}$ – множество вершин, $E = \{(u, v) | u, v \in V, u \neq v\}$ – множество ребер (между любыми двумя вершинами есть два ориентированных ребра) и, таким образом, G – полносвязный ориентированный граф.

Для первого способа сведения задачи верно следующее утверждение:

Утверждение. Каждому пути в графе G соответствует ровно одно расписание, и для одного и того же расписания может существовать более одного пути, по которому его можно построить.

Доказательство. Алгоритм построения расписания по последовательности работ детерминирован, поэтому каждому пути в графе G соответствует ровно одно расписание. Вторую часть утверждения можно доказать, приведя соответствующий пример: исходными данными является набор из двух сообщений. Каждому сообщению соответствует ровно одна работа, причем множества подциклов, в которые эти работы можно разместить, не пересекаются. Так как работы не могут быть размещены в одном подцикле, то они не могут быть и в одной цепочке. Это означает, что место в расписании для этих работ одинаково вне зависимости от взаимного расположения этих работ в пути. Таким образом, два различных пути в графе G соответствуют одному и тому же расписанию.

Второй способ сведения задачи основан на соответствии вершин графа сообщениям в задаче. Сопоставим каждому сообщению M_i из SM вершину v_i. Аналогично введем специальную вершину O, с которой будет начинаться маршрут каждого муравья. Получим граф $G = \langle V, E \rangle$, где $V = \{O, v_1, v_2, \ldots, v_n\}$ – множество вершин, $E = \{(u, v) | u, v \in V, u \neq v\}$ – множество ребер (между любыми двумя вершинами есть два ориентированных ребра), и аналогично получим, что G – полносвязный ориентированный граф.

Для второго способа сведения задачи также верно приведенное выше утверждение.

Преимуществами второго способа сведения задачи построения расписания к задаче поиска на графе пути являются:

1) меньшее число вершин в графе, что приводит к снижению вычислительной сложности и к сужению пространства поиска путей с высоким значением целевой функции;

2) повышенная вероятность размещения всех работ одного сообщения, что важно при построении реальных систем.

Исследования показали, что алгоритм, использующий второй способ сведения задачи, показывает лучшие результаты. Поэтому этот алгоритм будет применяться для исследований в разд. 4.

3.3. Локальные эвристические функции на ребрах графа. Для первого способа сведения задачи построения расписания к задаче нахождения пути на графе предложены следующие способы задания эвристической функции на ребрах графа:

1) $\eta_{ij}^1 = 1/(f_j - s_j)$, где f_j и s_j – соответственно директивное время завершения и директивное время начала работы – чем у́же директивный интервал работы, тем раньше ее предпочтительно разместить;

2) $\eta_{ij}^2 = t_j/(f_j - s_j)$, где t_j – время выполнения работы – чем у́же директивный интервал работы и чем больше времени нужно на ее выполнение, тем раньше ее предпочтительно разместить;

3) $\eta_{ij}^3 = 1/NS$, где NS – количество подциклов, в которые может быть размещена j-я работа с соблюдением технологических ограничений и требований реального времени – чем меньше это число, тем раньше работу предпочтительно разместить.

Для второго способа сведения задачи предложены следующие способы задания эвристической функции на ребрах графа:

1) $\mu_{ij}^1 = 1/(p_j^R - p_j^L)$, где p_j^L и p_j^R – соответственно левый и правый фазовые сдвиги – аналог эвристики η_{ij}^1;

2) $\mu_{ij}^2 = w_i/(p_j^R - p_j^L)$, где w_j – число слов данных в сообщении – аналог эвристики η_{ij}^2;

3) $\mu_{ij}^3 = 1/\tau_j$, где τ_j – период передачи сообщения – эвристика, известная как rate monotonic [17].

Такие эвристические функции позволяют выбирать для частных задач, какой из параметров работ или сообщений наиболее важен с точки зрения точности алгоритма при выборе очередной вершины графа. Если таких параметров более одного, возможно использование взвешенной суммы любых из перечисленных выше эвристик.

3.4. Алгоритм построения маршрута. Алгоритм аналогичен алгоритму из разд. 2 с уточнением, что в качестве табу-списка L_k используется список посещенных вершин.

3.5. Алгоритм размещения работ в подциклы. Для построения расписания по пути в графе требуется преобразовать последовательность вершин пути в последовательность работ. Для первого способа соответствие между множеством вершин и множеством работ взаимооднозначно: если $P = \{p_i | i = \overline{1, n_{jobs}}\}$ – последовательность вершин и A_j – работа, соответствующая вершине p_j, то $A = \{A_j | j = \overline{1, n_{jobs}}\}$ – последовательность работ.

Пусть теперь $P = \{p_i | i = \overline{1, n}\}$ – последовательность вершин для второго способа сведения задачи построения расписания к задаче нахождения на графе пути и M_j – сообщение, соответствующее вершине p_j. Последовательность работ A строится следующим образом:

1) сообщения выбираются в последовательности, определяемой последовательностью вершин (выбор сообщения M_i);

2) сортировка работ множества J^i по возрастанию их левой границы директивного интервала;

3) добавление отсортированного множества работ во множество A с сохранением порядка;

4) переход к п. 1), если не все вершины просмотрены.

Построение расписания по последовательности работ выполняется разновидностью жадного алгоритма, называемой алгоритмом упаковки [7]. Алгоритм упаковки был специально разработан для применения в системах, использующих схему организации обменов с подциклами, и позволяет эффективно учитывать ограничение g_{10}.

Схема алгоритма упаковки выглядит так:

1) во время построения расписания снимается ограничение на количество цепочек в подцикле;

2) подциклы сортируются по критерию заполненности, т.е. в порядке увеличения суммарной длительности выполнения всех работ, включенных в подцикл. При одинаковом значении критерия сортировка происходит по увеличению времени старта подцикла;

3) очередной подцикл выбирается по порядку из отсортированного в п. 2 списка, если размещение работы в подцикл не нарушит всех ограничений g_i:

3.1) в выбранном подцикле просматриваются все цепочки в порядке увеличения их времен старта;

3.2) для очередной цепочки производится попытка разместить работу в ее конец. Если размещенная таким образом работа пересекается со следующей цепочкой подцикла, все работы следующих цепочек сдвигаются вправо, если это возможно;

3.3) если работу нельзя разместить в конец ни одной из цепочек, цепочки просматриваются заново в том же порядке;

3.4) для очередной цепочки $\forall i$ производится попытка разместить работу между работами с номерами в цепочке i и $i + 1$. Все работы, номер которых больше или равен $i + 1$, сдвигаются вправо на интервал времени, равный длительности размещаемой работы, если это возможно;

4) если работа не была размещена ни в одну из существующих цепочек подцикла, то создается новая цепочка, содержащая только размещаемую работу, если это не нарушает всех ограничений g_i;

5) если создание новой цепочки невозможно, п. 3), 4) выполняются для следующего по порядку подцикла. Работа размещается в первый подходящий для этого подцикл. Если работу не удается разместить ни в один подцикл, она помещается в список неразмещенных работ.

После проведения вышеописанных действий для каждой работы из множества A цепочки работ в каждом подцикле объединяются:

1) цепочки одного подцикла c_i сортируются в порядке убывания их времен старта, при этом цепочка C_0 с максимальным временем старта не учитывается;

2) все работы цепочки c_i копируются в начало цепочки C_0 с сохранением порядка, цепочка c_i после этого удаляется. Если какие-либо работы после копирования начинают нарушать любое из ограничений g_i, такие работы удаляются и считаются неразмещенными;

3) пункт 2 повторяется до тех пор, пока не будут просмотрены все цепочки подцикла.

После такой операции в каждом подцикле остается не более одной цепочки, что позволяет получить корректное расписание.

3.6. Обновление феромона. Алгоритм обновления феромона аналогичен соответствующему алгоритму из разд. 2 с условием, что феромон обновляется только на путях с наибольшим значением целевой функции на данной итерации. Кроме того, максимальное и минимальное допустимое количество феромона на ребре было ограничено.

4. Численное исследование свойств алгоритма. 4.1. Методика проведения исследования. Исследование проводилось на наборах данных, сгенерированных на основе реальных данных. При генерации наборов варьировалось число слов данных в сообщениях, общее число сообщений и количество сообщений с одинаковой частотой и фазовыми сдвигами. В зависимости от свойств сгенерированного набора сообщений данные разбивались на несколько классов.

Отличие классов исходных данных друг от друга заключалось в структуре фазовых сдвигов. Сообщения, претендующие на размещение работ в одни и те же подциклы, делились на несколько групп, причем сообщения одной группы имели одинаковую частоту и фазовые сдвиги (частотно-фазовые группы). Такое разделение было выбрано на основе структуры реальных данных.

В наборах сообщений класса исходных данных **А** в каждом подцикле выделяется две основных частотно-фазовых группы, интервалы которых частично пересекались. В наборах сообщений класса исходных данных **Б** имелось три частотно фазовых группы, причем интервал первой из групп включал в себя интервал второй группы и частично включал интервал третьей группы, притом второй и третий интервалы не пересекались. В каждом классе имелось 120 сгенерированных наборов исходных данных.

Исследование муравьиного алгоритма проводилось путем однократного запуска на всех наборах каждого класса. Алгоритм выполнял 100 итераций, если не было найдено оптимального решения. Для построения графика полученные результаты сортировались по возрастанию значения целевой функции. Одна точка на графике соответствует усреднению пяти результатов запуска алгоритма. Во всех случаях для алгоритма, основанного на схеме муравьиных колоний, в качестве эвристической функции использовалась взвешенная сумма μ^2 и μ^3.

Жадные алгоритмы, с которыми проводилось сравнение, построены по схеме, описанной в разд. 3.5. Последовательность сообщений для размещения в расписание определялась в соответствии с жадным критерием, а не последовательностью сообщений, которая получена муравьиным алгоритмом.

4.2. Сравнение разработанного алгоритма с жадным алгоритмом. На рис. 4 показано сравнение муравьиного алгоритма с жадным алгоритмом по значению целевой функции (отношение числа размещенных в расписание работ к числу планируемых работ) на классе исходных данных **А**. В жадном алгоритме 1 в качестве критерия сортировки сообщений использовалась взвешенная сумма μ^2 и μ^3 (аналогично задавалась эвристика муравьиного алгоритма), тогда как в жадном алгоритме 2 – взвешенная сумма μ^1 и μ^2.

Из рисунка видно, что муравьиный алгоритм получает решения с более высоким значением целевой функции, чем оба жадных алгоритма, при этом при значениях загрузки канала от 0.35 до 0.55 разница в качестве решений между муравьиным алгоритмом и жадным алгоритмом 1 отличается значительно. На этом же отрезке жадный алгоритм 2 показывает лучшие результаты, чем жадный алгоритм 1.

На рис. 5 представлено аналогичное сравнение алгоритмов на классе исходных данных **Б**.

На классе исходных данных **Б** жадный алгоритм 2 показывает более низкое качество решений при всех значениях загрузки канала, тогда как качество решений муравьиного алгоритма попрежнему выше, чем у обоих жадных алгоритмов.

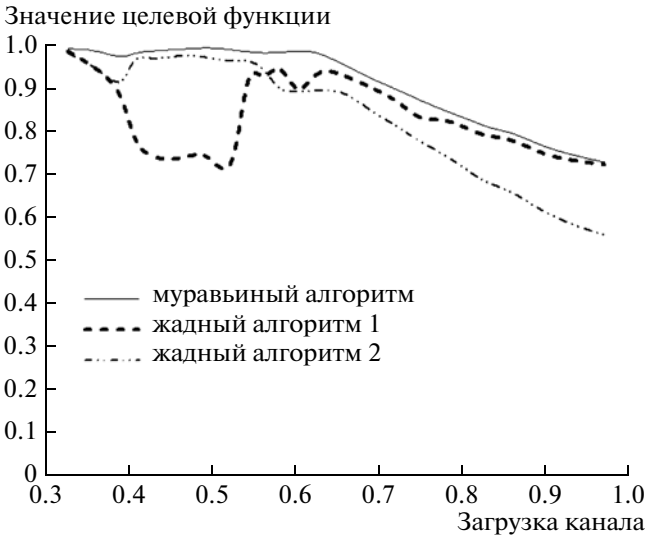

Рис. 4. Сравнение алгоритмов на классе исходных данных **А**

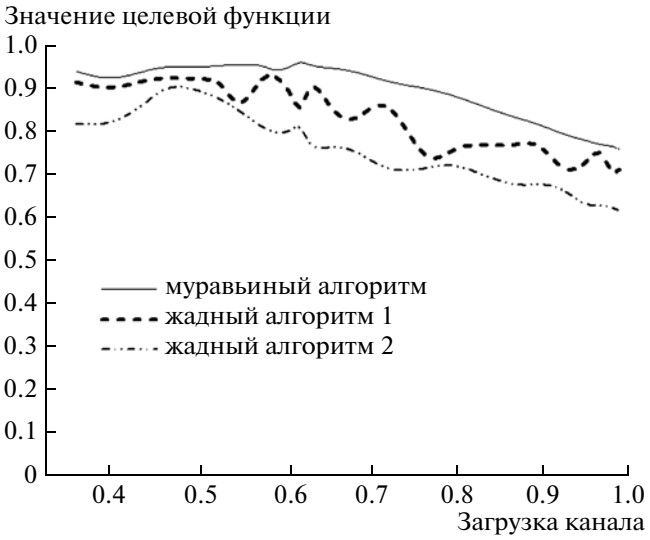

Рис. 5. Сравнение алгоритмов на классе исходных данных **Б**

Заключение. В данной работе предложен алгоритм для решения задачи построения расписания обменов по каналу с централизованным управлением, основанный на схеме муравьиных колоний. Рассмотрены два подхода к построению такого алгоритма. Отмечены преимущества второго подхода по сравнению с первым.

По результатам исследований на реальных данных муравьиный алгоритм, основанный на соответствии вершин графа сообщениям в исходных данных, показал лучшую точность по сравнению с жадными алгоритмами. Из проведенных исследований можно сделать вывод, что разработанный алгоритм является более универсальным и показывает лучшие результаты, чем существующие алгоритмы для данной задачи на всех рассмотренных классах реальных исходных данных. Предложенный алгоритм используется в инструментальной системе построения расписания обмена данными по каналу с централизованным управлением [4] в тех случаях, когда точность жадных алгоритмов недостаточна.

СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ Р 52070–2003. Интерфейс магистральный последовательный системы электронных модулей. Введ. 01.01.2004 М.: Изд-во стандартов, 2001. 23 с.
2. Guide to Digital Interface Standards for Military Avionic Applications: Technical report ASSC/110/6/2-ISSUE 3 // Avionics Systems Standardization Committee (ASSC). Leatherhead, UK, 2006. 249 p.
3. Information technology – Fibre Channel. Pt 312: Avionics Environment Upper Layerprotocol MIL-STD-1553B Notice 2 (FC-AE-1553): Technical Report TR 14165-312:2009 // International Organization for Standardization (ISO). Geneva, 2009. 84 p.
4. *Смелянский Р.Л., Костенко В.А., Балашов В.В. и др.* Инструментальная система построения расписания обмена данными по каналу с централизованным управлением // Современные технологии автоматизации. 2011. № 3. С. 78–84.
5. *Balashov V.V., Balakhanov V.A., Kostenko V.A. et al.* A Technology for Scheduling of Data Exchange Over bus with Centralized Control in Onboard Avionics Systems // J. Aerospace Engineering. 2010. V. 224. № 9. P. 993–1004.
6. *Балашов В.В., Костенко В.А.* Задачи планирования вычислений для одноприборных систем, входящих в состав вычислительных систем реального времени // Методы и средства обработки информации: Тр. Третьей Всероссийск. научн. конф. М.: МАКС Пресс, 2009. С. 193–203.
7. *Костенко В.А.* Алгоритмы построения расписаний для одноприборных систем, входящих в состав систем реального времени // Методы и средства обработки информации: Тр. Третьей Всероссийск. научн. конф. М.: МАКС Пресс, 2009. С. 245–258.
8. *Kostenko V.A., Gury'anov E.S.* An Algorithm for Scheduling Exchanges over a Bus with Centralized Control and an Analysis of Its Efficiency // Programming and Computer Software. 2005. V. 31. № 6. P. 340–346.
9. *Dorigo M.* Optimization, Learning and Natural Algorithms // PhD Thesis. Dipartimento di Elettronica. Milano: Politechnico Di Milano, 1992.
10. *Штовба С.Д.* Муравьиные алгоритмы: теория и применение // Программирование. 2005. № 4. С. 1–15.
11. *Stuzle T., Dorigo M.* ACO Algorithms for the Quadratic Assignment Problem // New Ideas in Optimization, Maidenhead: McGraw-Hill Ltd, 1999. P. 33–50.
12. *Levine J., Ducatelle F.* Ant Colony Optimization and Local Search for Bin Packing and Cutting Stock Problems // J. Operational Research Society. 2003. V. 55. P. 705–716.
13. *Ritchie G.* Static Multi-processor Scheduling with Ant Colony Optimization and Local Search // Master's Thesis. Edinburgh: University of Edinburgh, 2003.
14. *Blum C., Sampels M.* Ant Colony Optimization for FOP Shop Scheduling: A Case Study on Different Pheromone Representation // Proc. Congr. on Evolutionary Computation. Los Alamitos, CA: IEEE Computer Society Press, 2002. V. 2. P. 1558–1563.
15. *Гафаров Е.Р.* Гибридный алгоритм решения задачи минимизации суммарного запаздывания для одного прибора. М.: ВЦ РАН, 2006.
16. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: построение и анализ. М.: МЦНМО, 2000.
17. *Dougherty B., White J., Kegley R. et al.* Optimizing Integrated Application Performance with Cache-aware Metascheduling // 1st Intern. Symp. on Secure Virtual Infrastructures. Crete, Greece, 2011. P. 432–450.