ISSN 1816-0301 (Print)
ISSN 2617-6963 (Online)

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

MATHEMATICAL MODELING

УДК 519.1 + 656.021
https://doi.org/10.37661/1816-0301-2020-17-4-7-21

Поступила в редакцию 09.07.2019
Received 09.07.2019

Принята к публикации 20.10.2020
Accepted 20.10.2020

Вычислительные методы решения задачи комбинирования секторов воздушного пространства

И. В. Рубанов[1]✉, М. Я. Ковалев[2]

[1]*Белорусская государственная академия авиации, Минск, Беларусь*
✉*E-mail: irubanov@inbox.ru*
[2] *Объединенный институт проблем информатики*
Национальной академии наук Беларуси, Минск, Беларусь

Аннотация. Рассматривается задача комбинирования элементарных секторов региона воздушного пространства, в которой должно быть получено минимальное количество комбинированных секторов при ограничениях на их нагрузку и допустимость комбинаций, таких как требования связности в пространстве или принадлежности заданному множеству разрешенных комбинаций. Предлагаются и тестируются вычислительные методы, которые могут быть применены для решения этих и более общих задач секторизации регионов воздушного пространства. В частности, предлагаются комбинаторные алгоритмы двух типов для построения разбиений конечного множества с заданными весами элементов и теоретико-графовыми связями между ними. Разбиения строятся методом ветвей и границ с минимизацией количества подмножеств в итоговом разбиении при ограничении на суммарный вес элементов в подмножестве. В алгоритме первого типа в каждом узле дерева ветвей и границ формируются готовые компоненты итогового разбиения, оставшаяся часть исходного множества подвергается дальнейшему разбиению в нижестоящих узлах. В алгоритме второго типа в каждом узле формируется целиком текущее разбиение, компоненты которого дополняются в нижестоящих узлах. При сравнении показателей производительности алгоритмов рассматриваемые задачи делятся на две группы, в одной из которых на решение накладывается требование связности графовых структур, сопоставленных подмножествам итогового разбиения, а в другой такое требование отсутствует. Также предлагаются постановки задачи в терминах задачи целочисленного программирования.
Устанавливается вычислительная сложность двух вариантов задачи: типа задачи об упаковке в контейнеры, учитывающей ограничения на допустимость комбинаций, и типа задачи о покрытии.

Ключевые слова: секторизация воздушного пространства, безопасность полетов, разбиение конечного множества, целочисленное линейное программирование, метод ветвей и границ

Для цитирования. Рубанов, И. В. Вычислительные методы решения задачи комбинирования секторов воздушного пространства / И. В. Рубанов, М. Я. Ковалев // Информатика. – 2020. − Т. 17, № 4. – С. 7–21. https://doi.org/10.37661/1816-0301-2020-17-4-7-21

Computational methods for airspace sectorisation

Igor V. Rubanov[1]✉, Mikhail Y. Kovalyov[2]

[1]*Belarusian State Academy of Aviation, Minsk, Belarus*
✉*E-mail: irubanov@inbox.ru*
[2]*The United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Minsk, Belarus*

Abstract. A problem of combining elementary sectors of an airspace region is considered, in which a minimum number of combined sectors must be obtained with restrictions on their load and feasibility of combinations such as the requirement of the space connectivity or the membership of a given set of permissible combinations. Computational methods are proposed and tested to be used for solution of general problems of airspace sectorization. In particular, two types of combinatorial algorithms are proposed for constructing partitions of a finite set with specified element weights and graph-theoretical relationships between the elements. Partitions are constructed by use of a branch and bound method to minimize the number of subsets in the final partition, while limiting the total weight of elements in the subset. In the first type algorithm, ready-made components of the final partition are formed in each node of the branch and bound tree. The remaining part of the original set is further divided at the lower nodes. In the second type algorithm, the entire current partition is formed in each node, the components of which are supplemented at the lower nodes. When comparing algorithms performance, the problems are divided into two groups, one of which contains a connectivity requirement, and the other does not. Several integer programming formulations are also presented.
Computational complexity of two problem variants is established: a bin packing type problem with restrictions on feasible combinations, and covering type problem.

Keywords: airspace sectorization, flight safety, finite set partitioning, integer linear programming, branch and bound method

For citation. Rubanov I. V., Kovalyov M. Y. Computational methods for airspace sectorisation. *Informatics*, 2020, vol. 17, no. 4, pp. 7−21 (in Russian). https://doi.org/10.37661/1816-0301-2020-17-4-7-21

Введение. Одной из актуальных задач организации воздушного движения является повышение пропускной способности воздушного пространства (ВП) заданного региона. Среди факторов, оказывающих влияние на пропускную способность, большое значение имеет возможность контроля событий, происходящих в областях (секторах) ВП региона. В частности, оператор сектора ВП может отслеживать и поддерживать связь лишь с ограниченным количеством воздушных судов (ВС). Для поддержки контроля ВП в условиях изменяющейся ситуации и при ограниченных ресурсах может производиться динамическая секторизация ВП.

В самом недавнем из найденных по теме настоящей работы источнике [1], представляющем собой обзор предшествующих публикаций, можно найти ссылки на наиболее цитируемые, по мнению авторов статьи, работы. В обзоре представлена классификация исследований по нескольким критериям. Одним их них является подход к формированию секторов ВП, заключающийся либо в кратковременном комбинировании сравнительно небольших областей ВП правильной формы (назовем такой принцип моделью гибкой секторизации), либо в комбинировании уже существующих секторов с определенными для них ресурсами управления (назовем этот принцип моделью объединения или комбинирования секторов) [2].

В статьях [2, 3] отмечено, что второй принцип на сегодняшний день выглядит более реалистичным. В отличие от первого он является достаточно легко реализуемым, предлагая баланс между экономической эффективностью и безопасностью. Кратковременное комбинирование секторов, существующих в достаточно долговременно сформированных границах, уже практикуется на основе экспертных оценок нагрузки и ресурсов управления [4].

В обзоре [1] проводится различие между понятиями «секторизация» и «конфигурация» ВП. Конфигурация «обеспечивает расписание для объединения и разделения элементарных секторов на секторы управления, которые подходят для заданного числа доступных контроллеров и ожидаемой структуры трафика» [1, с. 1]. Конфигурация является более комплексным подходом, принимающим во внимание бóльшее количество таких факторов, как, например, определение временных интервалов, в течение которых осуществляется решение

задачи секторизации; прогнозирование с различным упреждением значений нагрузки; учет маршрутов движения ВС и т. д. Более общая задача конфигурации может включать ответы на важные вопросы определения метрик нагрузки и эффективности, при котором применение математических моделей секторизации имеет достаточный смысл с точки зрения повышения пропускной способности ВП либо уменьшения стоимости ресурсов.

В обзор [1] не включены описания работ по проблемам конфигурации, однако на них даются соответствующие ссылки. Оговаривается, что множества вопросов, затрагиваемых при решении задач секторизации и конфигурации, взаимопересекаются. С исследованиями, касающимися, например, вопросов определения метрик и тестирования эффективности моделей на их основе, можно ознакомиться в статьях [2–9]. Обзор [1] не включает источники русскоязычных авторов, среди которых можно упомянуть серию комплексных работ [5–7].

Задача конфигурации остается за пределами настоящего исследования. Предполагается, что на вход задачи подаются числовые данные, возможно, прошедшие какую-либо предобработку по более общим моделям. В работе предлагаются математические методы решения элементарной задачи комбинирования областей ВП (КОВП) при заданных в числовом виде ограничениях на нагрузку в итоговых секторах (будем называть их объединениями), максимизирующие эффективность, которая задается как их минимальное итоговое количество. Предполагается, что сформированная комбинация секторов неизменна в течение горизонта планирования.

В статье предлагается решение задачи при помощи комбинаторного алгоритма разбиения конечного множества методом ветвей и границ, подобного предлагаемому в работе [4], однако использующего несколько иной принцип разбиения. Дается пошаговое описание некоторых алгоритмов и блок-схема для одного из них. Приводятся результаты вычислительных тестов по сравнению производительности алгоритмов, основанных на обоих принципах разбиения, при выбранном критерии эффективности. Предлагается также формулировка задачи в нескольких вариантах как задачи целочисленного линейного программирования (ЦЛП).

Производительность работы алгоритмов и решателей можно ожидать достаточной для ВП относительно небольших регионов, включающих до 20 исходных областей.

Постановка задачи. Сформулируем элементарную задачу комбинирования секторов. Пусть имеется ВП, разделенное на несколько секторов. В каждом из них находится изменяющееся во времени количество ВС, которое будем называть нагрузкой сектора. Секторы могут быть разделены или объединены с соблюдением условия непревышения предела нагрузки, заданного для каждого из них. Предположим, что каждому сектору сопоставлено некоторое количество требуемых ресурсов для обслуживания и итоговая секторизация должна быть произведена с минимально возможным общим количеством требуемых ресурсов.

Математическая постановка задачи может быть следующей:

1. Пусть требуемый регион ВП разделен на области, которые назовем, как в статье [1], *элементарными секторами*. К примеру, элементарными секторами могут быть части, образованные делением криволинейного в общем случае цилиндра, определяющего ВП, плоскостями или полуплоскостями, параллельными образующей цилиндра, и поверхностями, параллельными его основаниям (рис. 1). Обозначим множество элементарных секторов как S, а его элементы – как s_i, $i=\overline{1,n}$.

2. Будем рассматривать интервалы времени t, в течение которых нагрузку w_i в каждом из элементарных секторов и в их объединениях будем считать постоянной.

3. По каждому интервалу t решаем задачу такого разбиения множества S элементарных секторов s_i на подмножества S_j, при котором в каждом S_j нагрузка w_j не превысит допустимого значения k_j и при этом количество m подмножеств S_j будет минимально. Если для какого-либо s_i оказывается, что $w_i > k_i$, предварительно обновляем S, исключая этот элементарный сектор из обработки, и затем добавляем его к итоговому разбиению. Такой показатель эффективности постановки предполагает, что количество требуемых ресурсов одинаково для всех S_j.

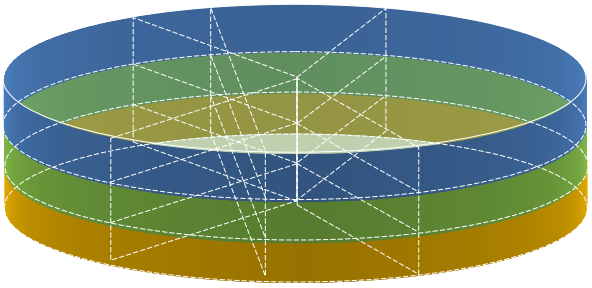

Рис. 1. Деление ВП на элементарные секторы

4. Имеются два варианта дополнительного условия допустимости объединения секторов:

а) могут объединяться смежные элементарные секторы;

б) задается конечное множество P допустимых объединений, включающее одноэлементные объединения.

5. Допустимая нагрузка k_j может определяться в следующих вариантах:

а) как $\max\limits_i (k_i)_j$ в вариантах 4 дополнительного условия допустимости объединения;

б) задаваться явно для каждого S_j в варианте 4 б);

в) быть постоянной k для всех s_i, S_j.

6. Фактическая нагрузка w_j в итоговом объединении определяется как сумма фактических нагрузок w_i его элементов.

Задача в формулировке 1–4 не отвечает на вопросы определения временных интервалов t, возможного усреднения параметров нагрузки и т. д. Эти вопросы авторы относят к комплексной задаче конфигурации, которая, как уже говорилось во введении, находится за пределами настоящего исследования.

Вычислительная сложность модели. Предлагаемая модель с критерием допустимой нагрузки 5 в) постановки задачи без учета дополнительных условий из п. 4 может быть представлена как известная NP-трудная задача об упаковке в контейнеры [10]. Если принять дополнительное условие допустимости 4 а), то модель можно представить как обобщение задачи об упаковке в контейнеры. Условие смежности секторов можно задать в виде графа, тогда обычная задача о контейнерах будет задана на полном графе и из ее NP-трудности следует NP-трудность задачи, заданной на любом графе.

Далее, задачу с критерием допустимой нагрузки 5 а) можно рассмотреть как обобщение предыдущего варианта. Тогда из NP-трудности задачи, в которой допустимая нагрузка определяется по формуле из п. 5 а) с одинаковыми k_i, следует NP-трудность задачи с разными k_i.

Если принять дополнительное условие допустимости 4 б) и если множество P задано изначально, то можно вообще не задавать критерии допустимой нагрузки, а подмножества с превышением сразу отбрасывать, проводя предобработку P. Таким образом, модель с данными условиями может быть представлена как задача о точном покрытии минимальным количеством подмножеств. Если количество возможных связных объединений в S не слишком велико, то дополнительное условие допустимости 4 а) может быть заменено на 4 б). Вариант задачи с условием 4 б) встречается в приложениях в предметной области; в частности, именно он рассматривается в работе [4]. Доказательство NP-трудности варианта 4 б) задачи без критерия допустимой нагрузки будет дано ниже.

Введем следующие обозначения для сформулированных вариантов рассматриваемой задачи: КОВП$_1$ – задача с критерием допустимой нагрузки 5 а) или 5 в) и дополнительным условием 4 а) или без него, КОВП$_2$ – задача без критерия допустимой нагрузки и с условием 4 б).

Из комбинаторного анализа известно, что количество вариантов полного перебора всех разбиений множества из n элементов определяется известным числом Белла [11]:

$$B_n = \sum_{q=0}^{n} S(n,q). \quad (1)$$

Здесь $S(n,q)$ – числа Стирлинга второго рода,

$$S(n,m) = \frac{1}{m!}\sum_{q=0}^{m}(-1)^{m+q} C_m^q q^n, \quad (2)$$

где C_m^q – биномиальный коэффициент.

Один из комбинаторных алгоритмов, описанных в следующем разделе, основан на рекуррентной формуле для числа Белла

$$B_{n+1} = \sum_{q=0}^{n} C_n^q B_q, \; B_0 = 1. \quad (3)$$

В выбранном вычислительном инструменте алгоритм выполняется за время не более нескольких минут для задач КОВП$_1$ с $n \le 24$ в варианте 5 в) определения допустимой нагрузки.

Также сформулируем и докажем утверждение об NP-трудности задачи КОВП$_2$.

Теорема. *Задача КОВП$_2$ является NP-трудной в сильном смысле.*

Доказательство. Покажем, что КОВП$_2$ NP-трудна в сильном смысле путем псевдополиномиального сведения к ней NP-полной в сильном смысле задачи «Точное покрытие 3-множествами (ТП-3)» [12].

Задача ТП-3: для заданного семейства $C = \{C_1,\ldots,C_r\}$ трехэлементных подмножеств множества $H = \{1,\ldots,3h\}$ следует определить, содержит ли C точное покрытие H, т. е. подсемейство X из C, такое, что каждый элемент $j \in H$ принадлежит точно одному трехэлементному множеству в X.

Для любого примера ТП-3 построим следующий пример КОВП$_2$: $S = H$, $P = P_3 \cup P_1$, где $P_1 = H$, $P_3 = C$ и требуется определить, существует ли точное покрытие S не более чем h подмножествами из P. Очевидно, что приведенное сведение является псевдополиномиальным. Покажем, что решение примера ТП-3 существует тогда и только тогда, когда существует решение соответствующего примера КОВП$_2$. Предположим, что подсемейство X из C является решением примера ТП-3. Поскольку $S = H$, $P_3 = C$ и $|X| = h$, множество X является решением соответствующего примера КОВП$_2$. Теперь предположим, что множество Y подмножеств из P является решением примера КОВП$_2$. Отметим, что если в Y есть хотя бы одно одноэлементное множество из P_1, то множество S из $3h$ элементов невозможно точно покрыть h и менее подмножествами, каждое из которых состоит из одного либо трех элементов. Поэтому Y является подсемейством семейства $C = P_3$. Кроме того, если $|Y| \le h-1$, то общее количество элементов в Y не превосходит $3(h-1)$ и оно не может покрывать S. Значит, $|Y| = h$ и Y являются решением примера ТП-3, что завершает доказательство.

Комбинаторные алгоритмы. Рассмотрим два типа комбинаторных алгоритмов: основанные на разделении (будем далее обозначать их алгоритмами типа А) и основанные на добавлении (будем обозначать их алгоритмами типа В). Применим алгоритмы к задаче КОВП$_1$ с критерием 5 в). Алгоритмы основаны на работе рекурсивной функции, которую далее будем обозначать как $f(arg)$.

Комбинаторные алгоритмы, основанные на разделении. В алгоритмах этого типа в данном узле на каком-либо уровне рекурсии получаем очередной класс J разбиения множества I номеров i элементов s_i исходного множества S, дополняя какое-либо подмножество arg_{pq} множества $arg \setminus i_0$ до самого аргумента arg функции $f(arg)$ для данного узла. Таким

образом, $J = arg \setminus arg_{pq}, arg_{pq} \in arg \setminus i_0$, где $p = \left|arg_{pq}\right|$, q – условный номер подмножества среди тех, мощность которых равна p, а i_0 – какой-либо фиксированный номер элемента в аргументе arg (рис. 2). Далее добавляем класс J к формируемому подобным образом по ветке дерева разбиению $\underline{J}$ и, если J является допустимым, а ветка не приводит к количеству классов в $\underline{J}$, превышающему рекорд, вызываем $f\left(arg_{pq}\right)$ и переходим на следующий уровень. Здесь и далее подчеркивание символа означает множество обозначаемых этим символом элементов. На рис. 2 показан шаг ветвления рекурсии для алгоритмов типа А при $i_0 = n$ и некоторые итоговые разбиения, получаемые на концах ветвей.

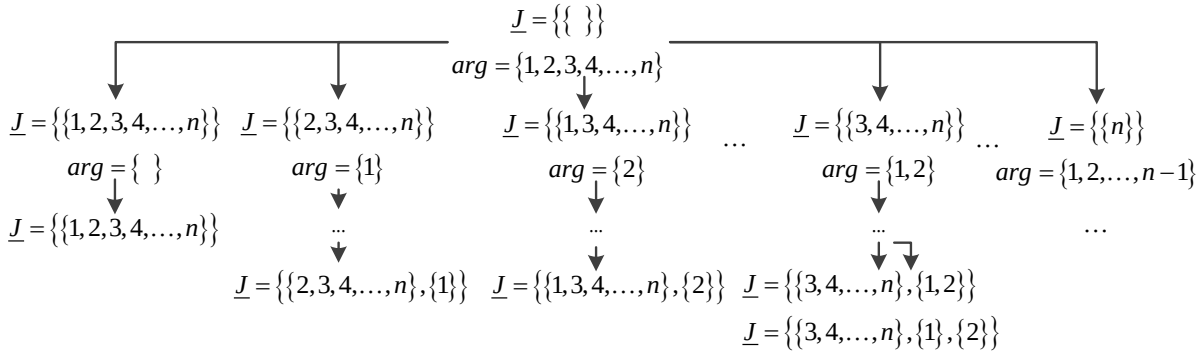

Рис. 2. Ветвление рекурсии алгоритмов типа А

Опишем базовый алгоритм, обозначим его А1.

Создадим пустое множество $\underline{J}$ пустых классов J номеров i элементов s_i множества S, $i \in I, I = \{1,...,n\}$. Определим также пустое множество $\underline{\underline{J}}$ получаемых разбиений. Объявим рекорд $r = n$.

Определим рекурсивную функцию $f\left(arg\right)$ следующим образом:

а) проверим условие

$$\left|\underline{J}\right| < r - 1. \tag{4}$$

Если условие (4) выполняется, переходим к п. б), в противном случае – к п. е);

б) зафиксируем номер $i_0 \in arg$ какого-либо s_i и будем перечислять все подмножества arg_{pq} множества $arg \setminus \{i_0\}$ мощностью $p = \overline{0, \left|arg \setminus \{i_0\}\right| - 1}$. Как только берется очередное arg_{pq}, определяется множество $J' = arg \setminus arg_{pq}$ и проверяется условие

$$\sum_{i \in J'} w_i \le k. \tag{5}$$

Если условие (5) выполняется, переходим к п. в); если нет, – к п. е). Закончив перечисление, перейдем к п. е);

в) создадим класс $J = J'$, добавим его к $\underline{J}$ и перейдем к п. г);

г) если $arg_{pq} = \varnothing$, перейдем к п. д); если нет, вызовем $f\left(arg_{pq}\right)$;

д) добавим $\underline{J}$ к $\underline{\underline{J}}$, объявим новый рекорд $r = \left|\underline{J}\right|$, удалим последний элемент из $\underline{J}$ и перейдем к п. е);

е) удалим последний элемент из $\underline{J}$ и выйдем из функции $f\left(arg\right)$.

Далее, если $\sum_{i\in I} w_i = 0$, строим единственное разбиение, состоящее из одного класса и включающее все индексы. В противном случае для запуска алгоритма построения разбиений вызываем $f(I)$.

Искомое оптимальное разбиение определяется наборами индексов в одном из $\underline{J} \in \underline{\underline{J}}$, $|\underline{J}| = r$. Если требуется найти все такие разбиения, в условии (4) следует поставить знак нестрогого неравенства, тогда в $\underline{\underline{J}}$ может оказаться несколько оптимальных разбиений.

Производительность алгоритма А1 можно улучшить, если учесть минимальное количество классов в допустимом разбиении очередного arg. В самом деле, если учитывать, что каждое arg может быть разбито не менее чем на

$$m_{arg} = m_0 = \left\lceil \sum_{i\in arg} w_i \Big/ k \right\rceil \tag{6}$$

частей, то при перечислении подмножеств в $arg \setminus i_0$ можно брать только те из них, в которых количество p элементов не меньше чем $m_{arg} - 1$. Существуют также улучшенные оценки [10], однако далее для простоты будем пользоваться оценкой (6). Условие (4) может быть определено неравенством

$$|\underline{J}| < r - m_{arg}, \tag{7}$$

т. е. ветви, приводящие к разбиениям, мощность которых больше рекорда, можно отсекать раньше, чем при выявлении их по исходной формуле (4).

Опишем улучшенный алгоритм, обозначим его А2.

Рекурсивную функцию определим следующим образом:

а) проверим условие (7). Если оно выполняется, переходим к п. б), в противном случае – к п. д);

б) зафиксируем номер $i_0 \in arg$ какого-либо s_i и будем перечислять все подмножества arg_{pq} множества $arg \setminus \{i_0\}$ мощностью $p = \overline{m_{arg} - 1, |arg \setminus \{i_0\}| - 1}$. Как только берется очередное arg_{pq}, определяется множество $J' = arg \setminus arg_{pq}$. Если $|J'| > 1$, проверяется условие (5). Если оно выполняется, переходим к п. в); если нет, – к п. д). Закончив перечисление, переходим к п. д);

в) создадим класс $J = J'$ и добавим его к $\underline{J}$. Если $arg_{pq} = \varnothing$, переходим к п. г); если нет, вызываем $f(arg_{pq})$;

г) объявим новый рекорд $r = |\underline{J}| + 1$, создадим класс $J = arg$ и добавим его к классу $\underline{J}$, который, в свою очередь, добавим к $\underline{\underline{J}}$. Удалим последний элемент из $\underline{J}$ и перейдем к п. д);

д) удалим последний элемент из $\underline{J}$ и выйдем из функции $f(arg)$.

Если требуется проверить условие 4 а) постановки задачи, в описанных вариантах это легко сделать, добавив проверку множества S'_j, определяемого множеством J', на соответствие условию связности к проверке условия (5) в п. г) обоих алгоритмов.

Например, если требуется, чтобы в каждом объединении S_j каждый элементарный сектор s_i был смежен с каким-либо другим по граням или, возможно, ребрам или углам, исходному множеству S можно сопоставить граф G, каждой вершине v_i которого сопоставлен сектор s_i, а ребра соответствуют смежности между s_i по нужному критерию. На рис. 3 показан граф смежности по граням элементарных секторов, который соответствует структуре, изображенной на рис. 1.

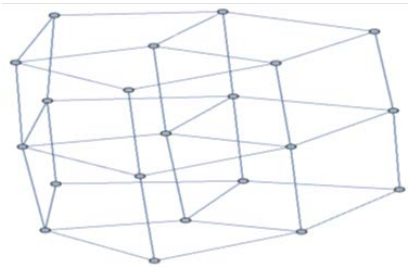

Рис. 3. Граф смежности элементарных секторов

Далее в качестве условия связности объединений можно взять связность подграфа $G_j \in G$, соответствующего объединению S_j.

На рис. 4 изображена структурная схема алгоритма A2 с проверками условия связности.

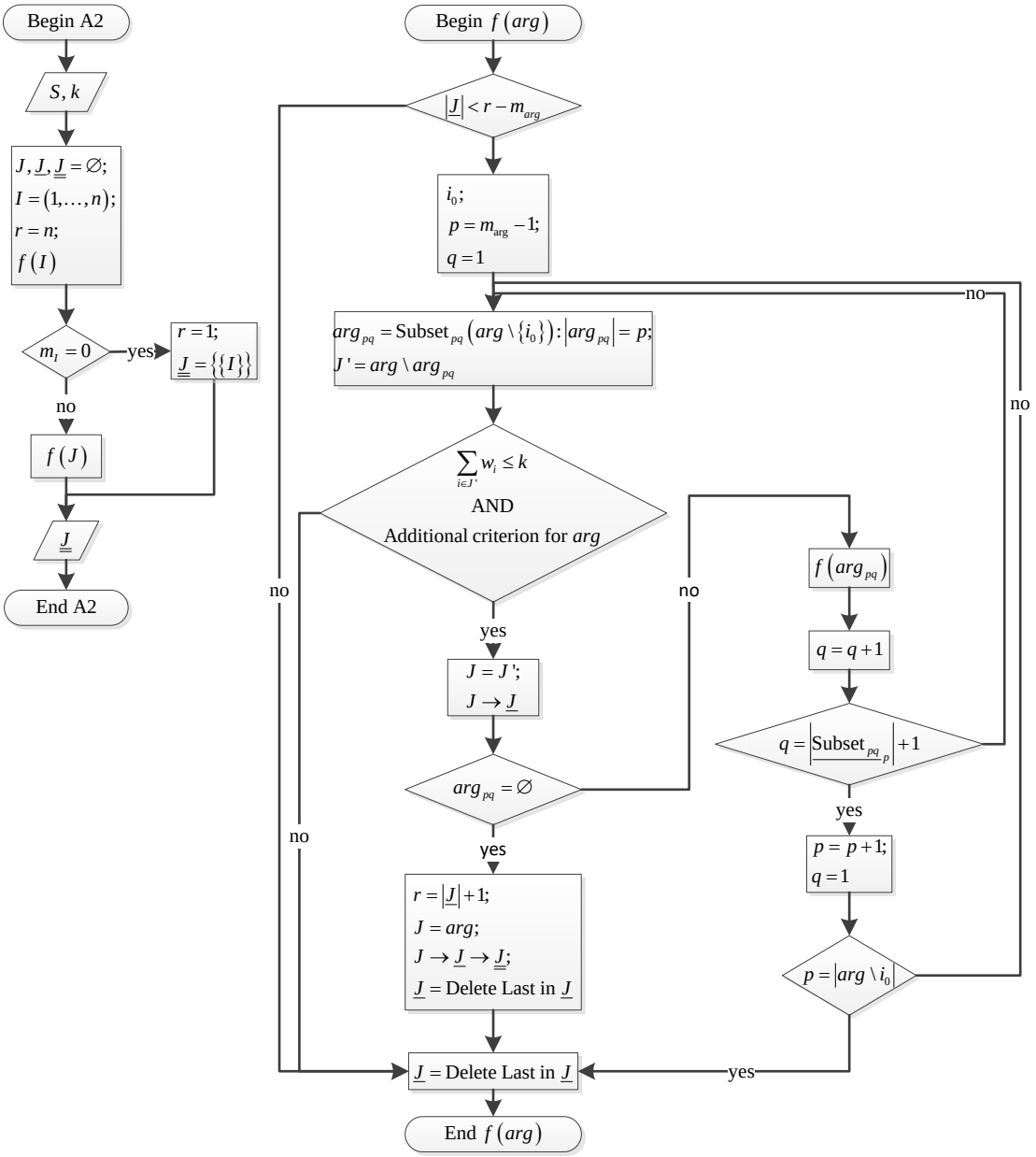

Рис. 4. Алгоритм A2

Комбинаторные алгоритмы, основанные на добавлении. Можно строить разбиения, начиная с множества $\underline{J} = \{\{1\}\}$ и получая каждый узел рекурсии на уровне i при помощи добавления элемента i к каждому из подмножеств верхнего узла и элемента $\{i\}$ ко всему множеству верхнего узла (рис. 5). При подобном принципе разбиения каждый узел представляет одно из разбиений множества элементов с номерами $1, \ldots, i$ и на уровне i рекурсии можно получить все возможные такие разбиения. Итоговый набор будет получен на нижнем уровне. Если в каком-либо разбиении количество классов превышает рекорд, по этой ветке далее не идем.

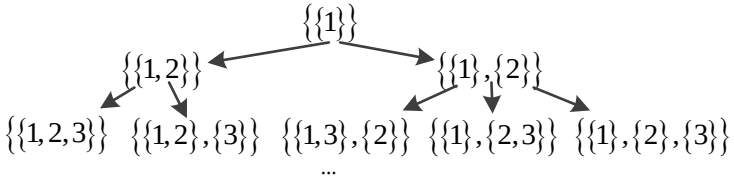

Рис. 5. Схема построения разбиения при помощи добавления элементов

Алгоритм с принципом построения разбиения, основанным на добавлении, предложен в работе [4] для задачи, сходной с обозначенной в настоящей работе как КОВП$_2$. При выбранных входных данных алгоритмы, построенные по схеме на рис. 5, показывают лучшее время выполнения для построения всего множества разбиений и для некоторых примеров задачи без условия связности. Однако, как будет предположено далее и как показывают результаты тестов, их производительность уступает алгоритму А2 с условием 4 а). Следует отметить, что улучшенная производительность А2 основана на использовании простого показателя эффективности, задаваемого как общее количество результирующих объединений. В работе [4] предлагаются, как можно понять, более комплексные критерии эффективности.

Опишем один из вариантов базового алгоритма, обозначим его В1.

Создадим пустое множество $\underline{\underline{J}}$ и множество $\underline{J} = \{\{1\}\}$. Объявим рекорд $r = n$, счетчик $i = 1$.

Определим рекурсивную функцию $f(arg)$ следующим образом:

а) проверим условие

$$|arg| < r. \tag{8}$$

Если оно выполняется, перейдем к п. б); если нет, – к п. д);

б) проверим условие

$$i = n. \tag{9}$$

Если оно выполняется, добавим arg к $\underline{\underline{J}}$ и объявим рекорд $r = |arg|$. В противном случае перейдем к п. в);

в) перечислим подмножества J аргумента $\underline{J}$, в каждой итерации увеличим счетчик на единицу, проверяя условие (5) для $i \in J \cup \{i\}$. Если оно выполняется, определим $J = J \cup \{i\}$ и вызовем $f(\underline{J})$; если нет, уменьшим счетчик на единицу и перейдем к следующей итерации. Закончив перечисление, перейдем к п. г);

г) увеличим счетчик на единицу, определим $\underline{J} = \underline{J} \cup \{\{i\}\}$ и вызовем $f(\underline{J})$;

д) уменьшим счетчик на единицу и выйдем из $f(arg)$.

Для запуска алгоритма вызовем $f(\underline{J})$.

Если требуется найти все решения, в условии (8) следует разместить знак нестрогого неравенства. Несколько ускоряет работу алгоритма B1 выполнение п. г) только по условию (8) со строгим неравенством. Также можно ожидать несколько более быстрое выполнение алгоритма B1 для задач, в которых оптимум совпадает с какой-либо оценкой m_I. Например, $m_I = m_0$ (см. (6)), если изначально определить $r = m_I$. Тогда для построения множества всех оптимальных разбиений можно пропустить п. а), а п. г) выполнять по условию (8). Для построения единственного разбиения в качестве условия (8) можно взять проверку условия $\underline{\underline{J}} = \varnothing$, а п. г) также выполнять по условию (8). Для задач, в которых количество классов оказывается большим m_I, можно предусмотреть итеративное выполнение алгоритма с увеличением r на каждом шаге на единицу. Назовем такие алгоритмы B2.

Итоги тестов показали, что индивидуальных задач, для которых выполняется $r = m_0$ (т. е. оптимальное значение соответствует не самой лучшей оценке), оказывается больше примерно на порядок даже при небольших $k = \overline{1,15}$. На рис. 6 показано распределение количества таких задач (обозначены зелеными точками) по сравнению с теми, в которых $r - m_0 = 1$ (обозначены желтыми точками), и теми, в которых $r - m_0 \geq 2$ (обозначены красными точками), по результатам четырех серий тестов. Для каждой из этих серий взято 135 индивидуальных задач с параметрами $k = \overline{1,15}$, $n = \overline{2,10}$, $w_i = R\left(\{0, \ldots, k\}\right)$, R – функция случайного выбора значений из аргумента. Таким образом, для большого количества задач можно ожидать меньшее среднее время выполнения алгоритма B2 по сравнению с B1.

Что касается проверки связности подграфа $G_j \in G$, то она менее удобна, чем в алгоритмах типа A. Не видно простой возможности осуществить такую проверку, пока не сформировано полное разбиение, поскольку на каждом шаге рекурсии любой из классов может быть дополнен. С другой стороны, операция взятия подмножеств заданной длины более вычислительно затратна, чем простое дополнение множества одним элементом. Как уже отмечалось ранее, тесты показывают более медленную работу по сравнению с алгоритмом A2 для одной и той же индивидуальной задачи КОВП$_1$ с критерием допустимой нагрузки 5 в). Для задачи КОВП$_2$ в каждом узле рекурсии можно проводить проверку принадлежности формируемых подмножеств разбиения подмножествам из P подобно способу, предложенному в работе [4].

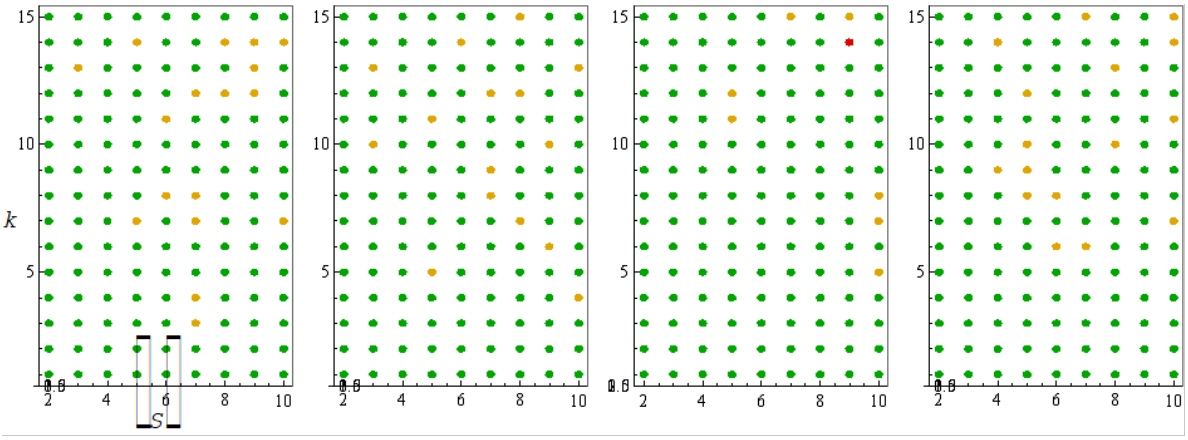

Рис. 6. Распределение задач $r = m_0$ и $r > m_0$

Вычислительные тесты комбинаторных алгоритмов. Рассмотрим сравнительные результаты вычислительных тестов для алгоритмов A2, B1, B2 в пакете Wolfram Mathematica 10.0.2.0 на компьютере с процессором Intel Celeron 2,7 ГГц и оперативной памятью 4 ГБ.

В табл. 1–3 приведены результаты поиска одного решения при помощи алгоритмов A2, B1 и B2 для задач КОВП$_1$ с условием 4 а) (группа I) и без него (группа II). В качестве условия связности взята связность подграфов одной из структур на рис. 7. Задачи для обеих групп тестов для всех алгоритмов сгенерированы по одному и тому же набору из 75 массивов W весов $w_i = R(\{0, \ldots, k\})$ при $n \in \{8, 9, 12, 15, 16\}$ и $k \in \{10, 15, 20\}$. Для каждой пары параметров взято пять задач, по которым определяется среднее время в секундах. В крайнем справа столбце приводится среднее время выполнения для каждого $|S|$ по всем k отдельно для групп I и II. Для сравнения алгоритмов можно сопоставлять значения в соответствующих ячейках трех таблиц.

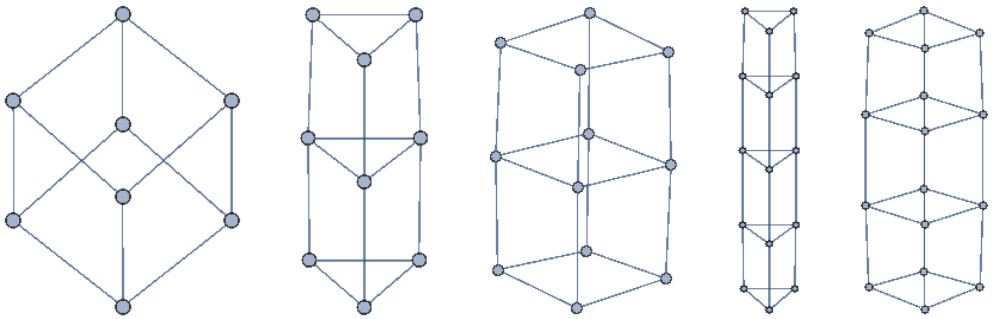

Рис. 7. Графовые структуры для тестов задач с условием связности подграфов

Таблица 1

Результаты тестов алгоритма A2

Группа	n \ k	10	15	20	Среднее
I	8	0	0	0	0
II		0,001 040 01	0,001 040 01	0,002 080 01	0,001 386 68
I	9	0,001 040 01	0,003 120 02	0,001 040 01	0,001 733 34
II		0,002 080 01	0,002 080 01	0,001 040 01	0,001 733 34
I	12	0,008 32	0,012 48	0,014 56	0,011 787
II		0,006 240 04	0,009 36	0,009 36	0,0214 935
I	15	0,100 881	0,082 161	0,127 921	0,103 654
II		0,227 761	0,126 881	0,060 32	0,138 321
I	16	0,316 162	0,238 162	0,190 321	0,248 215
II		2,538 656	0,138 321	0,442 003	1,039 66

Таблица 2

Результаты тестов алгоритма B1

Группа	n \ k	10	15	20	Среднее
I	8	0,002 080 01	0,002 080 01	0,001 040 01	0,001 733 34
II		0	0,001 040 01	0	0,000 346 67
I	9	0,003 120 02	0,005 200 03	0,002 080 01	0,003 466 69
II		0,003 120 02	0	0,001 040 01	0,001 386 68
I	12	0,112 321	0,0572	0,266 242	0,145 254
II		0,049 920 3	0,034 320 2	0,056 16	0,046 800 3
I	15	3,098 18	0,997 366	4,822 511	2,972 686
II		0,310 962	0,552 244	0,240 242	0,367 816
I	16	62,298 479	24,185 355	19,164 203	35,216 012
II		9,388 14	2,442 976	0,958 886	4,263 334

Таблица 3

Результаты тестов алгоритма B2

Группа	k / n	10	15	20	Среднее
I	8	0,002 080 01	0,001 040 01	0,002 080 01	0,001 733 34
II		0,001 040 01	0	0	0,000 346 67
I	9	0,004 160 03	0,003 120 02	0,003 120 02	0,003 466 69
II		0,003 120 02	0,001 040 01	0,003 120 02	0,002 426 68
I	12	0,027 04	0,031 200 2	0,158 081	0,072 107 1
II		0	0,003 120 02	0,050 960 3	0,018 026 8
I	15	2,702 977	0,546 003	4,306 668	2,518 549
II		0,042 640 3	0,089 440 6	0,062 400 4	0,064 827 1
I	16	45,717 653	24,205 115	12,229 438	27,384 069
II		8,654 94	1,419 61	0,745 685	3,606 74

Как и ожидалось, алгоритм B2 работает эффективнее, чем B1, и является самым быстрым для задач, относящихся к группе II и $n<16$, а алгоритм A2 работает значительно быстрее B1 и B2 для задач группы I. Единичные тесты алгоритма A2 для поиска одного решения задач, где $n\in\{18, 20, 24\}$ и задано условие связности, показывают результаты, которые можно считать приемлемыми для использования в практических целях конфигурирования секторов ВП небольшого региона. Время работы для $n=24$ принимает значение порядка 2–3 мин.

Модель целочисленного линейного программирования. Задачу КОВП$_2$ с критерием допустимой нагрузки 5 в) без условия связности можно сформулировать в варианте постановки задачи ЦЛП для задачи упаковки в контейнеры:

$$\begin{cases} \sum\limits_{j=1}^{n} y_j \to \min; \\ \sum\limits_{i=1}^{n} w_i x_{ij} \le k y_j,\ \forall j \in \{1,\ldots,n\}; \\ \sum\limits_{j=1}^{n} x_{ij} = 1,\ \forall i \in \{1,\ldots,n\}; \\ x_{ij}, y_j \in \{0, 1\}, \end{cases} \tag{10}$$

где переменные x_{ij} обозначают размещение элемента s_i в объединение S_j, переменные y_j – наличие хотя бы одного элемента в объединении S_j.

Модель показывает хорошую производительность при использовании специализированных решателей. Например, единичный тест решения задач без условия связности при $n\in\{10, 20, 30, 40\}$, $k=15$, $w_i = R(\{0, \ldots, k\})$ с применением надстройки для MS Excel решателя IBM ILOG CPLEX показывает следующие значения времени работы решателя до получения оптимального решения при начальном решении $x_{ij}, y_j = 0$: 3 с для n = 10; 10 с для n = 20; 20 с для n = 30; 30 с для n = 40. Заметим, что только количество переменных модели (10) составляет $n^2 + n$.

Можно пробовать учесть условие связности, решая задачу (10) повторно с добавлением исключающих ограничений для недопустимых объединений. Например, пусть в результате решения получено несколько таких объединений. Обозначим множество индексов элементов в каком-либо недопустимом объединении J'' и множество индексов таких объединений $\underline{J''}$. Тогда к задаче (10) могут быть добавлены ограничения

$$\sum_{i \in J''} x_{ij} \le \left| J'' \right| - 1, \forall J'' \in \underline{J''}, \tag{11}$$

затем произведена попытка повторного решения и т. д.

Для задачи с критерием допустимой нагрузки 5 б) и дополнительным условием 4 б) можно предложить следующую постановку:

$$\begin{cases} \sum\limits_{j \in \underline{J''_1}} y_j \to \min; \\ y_j \sum\limits_{i=1}^{n} w_i a_{ij} \le k_j, \forall j \in \underline{J''_1}; \\ \sum\limits_{j \in \underline{J''_1}} a_{ij} y_j = 1, \forall i \in \{1, \ldots, n\}; \\ y_j \in \{0, 1\}, \end{cases} \tag{12}$$

где $\underline{J''_1}$ – множество индексов объединений из P; $a_{ij} \in \{0, 1\}$ – коэффициент, определяющий возможность вхождения элемента s_i в объединение S_j. Если проводится предобработка P с отбрасыванием подмножеств с превышением допустимой нагрузки, т. е. решается задача КОВП$_2$, то в постановке (12) остается только вторая группа ограничений и получается вариант постановки задачи о покрытии. Преимуществом (12) может являться небольшое количество переменных.

В работе [9] предложена оригинальная постановка для учета условия связности с применением модели потоков, однако количество только переменных потоков там составляет n^3. Тесты производительности модели ЦЛП с ограничениями для условия связности и сравнение их результатов с постановкой из [9] пока не проводились.

Также, как следует из определения круга решаемых в настоящем исследовании задач, авторы не претендуют на сравнение предложенной формулировки с теми, которые предлагаются в более комплексных моделях.

Следует отметить, что решение задачи методом математического программирования с использованием решателей обладает тем недостатком, что известные решатели не имеют настроек для получения множества оптимальных решений в случае их существования. Предложенные в предыдущем разделе алгоритмы, использующие метод ветвей и границ, могут быть легко адаптированы для получения всех оптимальных решений, которых в случае рассматриваемой задачи разбиения конечного множества часто оказывается несколько и может оказаться довольно много для заданной индивидуальной задачи. Получение множества решений может дать возможность учесть какие-либо дополнительные неформальные критерии допустимости или оптимальности при рассмотрении практической задачи комбинирования секторов ВП.

Заключение. В статье представлено описание и результаты тестов нескольких вычислительных методов решения элементарной задачи комбинирования областей ВП с критерием оптимальности и ограничениями, определенными в разделе «Постановка задачи». Предложенные методы могут найти применение при моделировании комплексных задач конфигурации ВП, разделенного на исходные элементарные секторы.

По результатам тестов на производительность предлагаются следующие способы решения задач, обозначенных как КОВП$_1$ и КОВП$_2$. Задачу КОВП$_1$ с условием связности и количеством исходных секторов в пределах 24 предлагается решать при помощи комбинаторного алгоритма, обозначенного как А2, время решения прогнозируется в пределах нескольких минут. Алгоритм А2 может быть применен также для решения всех сформулированных вариантов задачи, если требуется получение нескольких решений. Для задачи КОВП$_1$ с условием связности и количеством исходных секторов более 24 авторы считают целесообразным провести дополнительные исследования производительности нескольких моделей, построенных по методу ЦЛП. Задачу КОВП$_1$ без условия связности и задачу КОВП$_2$ более рационально решать мето-

дом ЦЛП с применением специализированных решателей. Применение такого метода дает в результате производительность менее минуты для количества элементарных секторов в пределах 40.

Список использованных источников

1. Flener, P. Automatic Airspace Sectorisation: A Survey [Electronic resource] / P. Flener, J. Pearson. – Department of Information Technology, Uppsala University, Sweden, 2018. – Mode of access: https://arxiv.org/abs/1311.0653. – Date of access: 03.06.2020.

2. Applying graph theory to problems in air traffic management / A. H. Farrahi [et al.] // 17th AIAA AVIATION Technology, Integration, and Operations Conference, Denver, Colorado, 5–9 June 2017 / AIAA AVIATION Forum. – Denver, Colorado, 2017. – 20 p.

3. Bloom, M. Combining airspace sectors for the efficient use of air traffic control resources / M. Bloom, P. Kopardckar // Navigation, and Control (GNC) Conference and Exhibit, Honolulu, HI, 18–21 Aug. 2008. – American Institute of Aeronautics and Astronautics, 2008. – 15 p.

4. Gianazza, D. Forecasting workload and airspace configuration with neural networks and tree search methods / D. Gianazza // Artificial Intelligence. – 2010. – № 174(7–8). – P. 530–549.

5. Дегтярев, О. В. Решение задач секторизации района управления воздушным движением. I. Основные принципы и проблемы секторизации воздушного пространства и ее формализация как оптимизационной задачи / О. В. Дегтярев, В. Н. Минаенко, М. О. Орехов // Известия РАН. Теория и системы управления. – 2009. – № 3. – С. 56–72.

6. Дегтярев, О. В. Решение задач секторизации района управления воздушным движением. II. Разработка алгоритмов секторизации / О. В. Дегтярев, В. Н. Минаенко, М. О. Орехов // Известия РАН. Теория и системы управления. – 2010. – № 4. – С. 117–135.

7. Вересов, К. А. Решение задач секторизации района управления воздушным движением. III. Разработка алгоритмов определения конфигураций и временного графика работы секторов управления / К. А. Вересов, О. В. Дегтярев, В. Н. Минаенко // Известия РАН. Теория и системы управления. – 2013. – № 2. – С. 133–153.

8. Bloom, M. Algorithms for combining airspace sectors / M. Bloem, P. Gupta, P. Kopardekar // Air Traffic Control Quarterly. – Sept. 2009. – Vol. 17, no. 3.

9. Drew, M. C. A method of optimally combining sectors / M. C. Drew // 9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO), Hilton Head, South Carolina, 21–23 Sept. 2009. – American Institute of Aeronautics and Astronautics, 2009. – P. 7057.

10. Martello, S. An Knapsack Problems. Algorithms and Computer Implementations / S. Martello, P. Toth. – John Wiley & Sons, 1990. – 296 p.

11. Стенли, Р. Перечислительная комбинаторика : в 2 т. / Р. Стенли. – М. : Мир, 1990. – Т. 1. – 440 с.

12. Гэри, М. Вычислительные машины и труднорешаемые задачи / М. Гэри, Д. Джонсон. – М. : Мир, 1982. – 416 с.

References

1. Flener P., Pearson J. *Automatic Airspace Sectorisation: A Survey*. Department of Information Technology, Uppsala University, Sweden, 2018. Available at: https://arxiv.org/abs/1311.0653 (accessed 03.06.2020).

2. Farrahi A. H., Goldberg A. T., Bagasol L. N., Jaewoo J. Applying graph theory to problems in air traffic management. *17th AIAA AVIATION Technology, Integration, and Operations Conference, Denver, Colorado, 5–9 June 2017, AIAA AVIATION Forum.* Denver, Colorado, 2017, 20 p. https://doi.org/10.2514/6.2017-3775

3. Bloom M., Kopardckar P. Combining airspace sectors for the efficient use of air traffic control resources. *Navigation, and Control (GNC) Conference and Exhibit, Honolulu, HI, 18–21 August 2008*. American Institute of Aeronautics and Astronautics, 2008, 15 p.

4. Gianazza D. Forecasting workload and airspace configuration with neural networks and tree search methods. *Artificial Intelligence*, 2010, no. 174(7–8), pp. 530–549.

5. Degtyarev O. V., Minaenko V. N., Orekhov M. O. Reshenie zadach sektorizacii rajona upravlenija vozdushnym dvizheniem [Reservation of regional sector regulations]. I. Osnovnye principy i problemy sektorizacii vozdushnogo prostranstva i ee formalizacija kak optimizacionnoj zadachi [Basic principles and problems of airspace sectorization and its formalization as an optimization problem]. Izvestija Rossijskoj akademii nauk. Teorija i sistemy upravlenija [*Bulletin of the Russian Academy of Sciences. Control Theory and Systems*], 2009, no. 3, pp. 56–72.

6. Degtyarev O. V., Minaenko V. N., Orekhov M. O. Reshenie zadach sektorizacii rajona upravlenija vozdushnym dvizheniem [*Solving the Problems of Sectorization of the Air Traffic Control Area*]. II. Razrabotka algoritmov sektorizacii [Development of sectorization algorithms]. Izvestija Rossijskoj akademii nauk. Teorija i sistemy upravlenija [*Bulletin of the Russian Academy of Sciences. Control Theory and Systems*], 2010, no. 4, pp. 117–135.

7. Veresov K. A., Degtyarev O. V., Minaenko V. N. Reshenie zadach sektorizacii rajona upravlenija vozdushnym dvizheniem [Solving the problems of sectorization of the air traffic control area]. III. Razrabotka algoritmov opredelenija konfiguracij i vremennogo grafika raboty sektorov upravlenija [Development of algorithms for determining configurations and a time schedule for the operation of control sectors]. Izvestija Rossijskoj akademii nauk. Teorija i sistemy upravlenija [*Bulletin of the Russian Academy of Sciences. Control Theory and Systems*], 2013, no. 2, pp. 133–153.

8. Bloom M., Gupta P., Kopardekar P. Algorithms for combining airspace sectors. *Air Traffic Control Quarterly*, September 2009, vol. 17, no. 3.

9. Drew M. C. A method of optimally combining sectors. *9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO), Hilton Head, South Carolina, 21–23 September 2009*. American Institute of Aeronautics and Astronautics, 2009, p. 7057.

10. Martello S., Toth P. *An Knapsack Problems. Algorithms and Computer Implementations*. John Wiley & Sons, 1990, 296 p.

11. Stanley R. *Enumerative Combinatorics*. Vol. 1. Cambridge University Press, 2012, 440 p.

12. Garey M., Johnson D. *Computers and Intractability*. New York, W. H. Freeman and Company, 1979, 340 p.

Информация об авторах

Рубанов Игорь Владимирович, старший преподаватель кафедры естественнонаучных дисциплин, Белорусская государственная академия авиации, Минск, Беларусь.
E-mail: irubanov@inbox.ru

Ковалев Михаил Яковлевич, доктор физико-математических наук, профессор, член-корреспондент НАН Беларуси, заместитель генерального директора, Объединенный институт проблем информатики Национальной академии наук Беларуси, Минск, Беларусь.
E-mail: kovalyov_my@newman.bas-net.by

Information about the authors

Igor V. Rubanov, Senior Lecturer of the Department of Science Disciplines, Belarusian State Academy of Aviation, Minsk, Belarus.
E-mail: irubanov@inbox.ru

Mikhail Y. Kovalyov, Dr. Sci. (Phys.-Math.), Professor, Corresponding Member of the National Academy of Sciences of Belarus, Deputy General Director, The United Institute of Informatics Problems of the National Academy of Sciences of Belarus, Minsk, Belarus.
E-mail: kovalyov_my@newman.bas-net.by