UDC 519.7:007:004
https://doi.org/10.31891/csit-2022-4-14

YURII TURBAL
The National University of Water and Environmental Engineering
SERHII BABYCH
Rivne Vocational College of NUBiP

INFORMATION TECHNOLOGY FOR THE SCHEDULE GENERATION BASED ON THE ALGEBRA OF ADDITIVE-DISJUNCTIVE FORMS AND THE MODIFIED METHOD OF PERMANENT DECOMPOSITION

To improve the information technology of drawing up class schedules, there is a need to develop methods that allow significantly reduce the number of combinatorial objects in the process of algorithms for generating schedules matrices. For example, the result of applying the method of permanent decomposition is a collection of combinatorial objects - permutations, combinations, and placements. For the task of drawing up lesson schedules in the part of forming timetable matrices, the method provides a memory-recorded set of all possible systems of various representatives of sets, which are the columns of the timetable matrices (SRPS). Since the algorithm of permanent decomposition gives all possible SRPS, it creates the problem of forming the final schedule based on SRPS or all possible variants of schedules and requires the development of special algorithms. Certain known approaches to solving such a problem are associated with significant computational complexity in the general case. This also applies to the approach based on the order relation of the set of decomposition matrices.

The basis of the information technology proposed in the work is the further modification of the incidence matrices and, accordingly, such a modification of the permanent decomposition method, which allows generating ready versions of the schedule matrices at the output. This is achieved due to the introduction of a special algebra of additive-disjunctive forms and, accordingly, the possibility of generating such forms in the process of permanent decomposition. In fact, in this context, ADF is a formal representation of a ready-made version of an admissible schedule that satisfies some additional requirements.

Keywords: information technology, additive-disjunctive form, permanent, decomposition.

ЮРІЙ ТУРБАЛ
Національний університет водного господарства та природокористування
СЕРГІЙ БАБИЧ
ВСП Рівненський фаховий коледж НУБіП України

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ СКЛАДАННЯ РОЗКЛАДУ НА ОСНОВІ АЛГЕБРИ АДДИТИВНО-ДИЗ'ЮНКТИВНИХ ФОРМ ТА МОДИФІКОВАНОГО МЕТОДУ ПЕРАМНЕНТНОЇ ДЕКОМПОЗИЦІЇ

З метою вдосконалення інформаційних технологій складання розкладів занять виникає необхідність у розробці методів, що дозволяють суттєво скоротити кількість комбінаторних об'єктів в процесі роботи алгоритмів генерації матриць розкладів. Наприклад, результатом застосування методу перманентної декомпозиції є сукупність комбінаторних об'єктів – перестановок, комбінацій, розміщень. Для задачі складання розкладів занять в частині формування матриць розкладів метод дає записану в пам'ять сукупність усіх можливих систем різних представників множин, що являють собою стовпчики матриць розкладів (СРПС). Оскільки алгоритм перманентної декомпозиції дає всі можливі СРПС, то це породжує проблему формування остаточного розкладу на основі СРПС чи усіх можливих варіантів розкладів та вимагає розробки спеціальних алгоритмів. Окремі відомі підходи до розв'язання такої задачі пов'язані з значною обчислювальною складністю в загальному випадку. Це стосується і підходу на основі відношення порядку на множині матриць розкладу.

В основі інформаційної технології, що пропонується в роботі, покладена подальша модифікація матриць інцидентності та, відповідно, така модифікація методу перманентної декомпозиції, яка дозволяє на виході генерувати готові варіанти матриць розкладів. Це досягається за рахунок введення спеціальної алгебри аддидитво-диз'юнктивних форм та відповідно, можливість генерації таких форм в процесі перманентної декомпозиції. По суті, в такому контексті АДФ є фактично формальним представленням готового варіанту допустимого розкладу, який задовольняє низку додаткових вимог.

Розроблена інформаційна технологыя вирішення задач календарного планування, зокрема задачі формування розкладів, є цілісною системою, що поєднує деякі підходи, зокрема конфігураційний підхід до аналізу вхідних даних та алгоритмів формування вихідних допустимих матриць розкладів, застосування алгоритмів перманентної декомпозиції, лексикографічний підхід на основі відповідних порядкових відношень, алгебру адитивно-диз'юнктивних форм.

Ключові слова: інформаційна технологія, аддитивно-диз'юнктивна форма, перманент, декомпозиція.

Introduction

A significant amount of most recent research has focused on the tasks of scheduling. There are numerous excellent resources available in the cloud. The cost of performing tasks in the cloud depends on what resources are used. Cloud planning is different from traditional planning. In the environment of cloud computing, the task of scheduling is the biggest and most difficult issue. The task scheduling problem is the NP-complete problem. Many heuristics have introduced scheduling algorithms, but more improvements are needed to make the system faster and more responsive. [1]

Although a significant number of algorithms have been developed to generate various combinatorial objects, such as permutations, permutations with repetitions of different types, and systems of subsets of some sets of elements [2-4, 5-12], new approaches and algorithms are still emerging.

Given the novelty of the combination of permanent and decomposition solutions within the calendar calculation - it is difficult to rely on similar literature.

Many different task scheduling problems such as assignment, job-shop, flow-shop, vehicle routing, and other scheduling problems have been studied intensively. The studied grid task scheduling problem in this work comes from the task-resource assignment problem [13] which is much more complicated than the above-stated classic task scheduling problems. Restated, a grid application is a task scheduling problem involving partially ordered tasks and distributed heterogeneous resources, and can be represented by a directed acyclic graph (DAG) [14–17]. The scheduling target is to find optimal task-resource assignment and hence minimize application completion time. Most scheduling problems are confirmed to be NP-complete. Thus, many researchers have devoted their efforts to solving task scheduling problems. The exact algorithm such as branch-and-bound method is able to find the optima of the scheduling problem. However, the execution time required is impractical as the number of tasks and resources increases. Hence, many different schemes have been presented for solving scheduling problems. Chen et al. [18] combined a competitive scheme with slack neurons into Hopfield neural networks to solve multiprocessor real-time job scheduling problems. Sandnes [19] presented a stochastic approach of employing randomization in the scheduling of tasks in multiobjective scheduling problems. An artificial immune-system based scheme was proposed to solve the dynamic economic dispatch problem of generating units [20]. Comparatively, several metaheuristics such as genetic algorithm (GA), simulated annealing algorithm (SA), tabu search (TS), ant colony optimization (ACO), and the particle swarm optimization have been effectively proposed for solving these difficult problems. Oh and Wu [21] presented a multiobjective genetic algorithm, which aims to minimize the number of processors required and the total tardiness of tasks. Liu and Wang [22] solved the resource-constrained project scheduling problem of minimizing activities' cost based on GA. And a thermal generating unit's commitment scheduling problem was studied by a modified GA [23]. Tabu search is an approach to prevent the search from trapping into the local minimum, and it has been applied to solving a single machine scheduling problem with distinct due windows to minimize total weighted earliness and tardiness [27] as well as job-shop scheduling [28].

A detailed overview of the combinatorial algorithms can be given by Knuth [3], and Ruskey [4] which considers the concept of combinatorial generation and distinguishes the following tasks: listing–generating elements of a given combinatorial set sequentially, ranking – numbering elements of a given combinatorial set, unranking - generating elements of a given combinatorial set per their ranks and random selection– generating elements of a given combinatorial set in random order.

General methods for developing combinatorial generation algorithms were studied by such researchers as S. Bacchelli [1], V.V. Kruchinin [5, 6], P. Flajolet [7] and others. It is a well-known algorithm for permutation generation [39], such as Bottom-Up, Lexicography, Johnson-Trotter [40], PIndex [41], and Inversion [42].

Additive-disjunction forms and permanent decomposition method

Suppose we have n elements $(a_1, a_2, \dots, a_m)$, that can be part of m sets $(W_1, W_2, \dots, W_m)$, and the occurrence of the same element several times is allowed. Information about which elements are included in the corresponding sets will be given in the form of an incidence matrix of the form:

$$\begin{pmatrix} & a_1 & a_2 & \dots & a_n \\ W_1 & n_{11} & n_{12} & \dots & n_{1n} \\ W_2 & n_{21} & n_{22} & \dots & n_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ W_m & n_{m1} & n_{m2} & \dots & n_{mn} \end{pmatrix} \quad (1)$$

The elements $(a_1, a_2, \dots, a_m)$ will be called identifiers of the columns of the incidence matrix. The system of different representatives (SDR) will be called a vector of the form:

$$(v_1, v_2, \dots, v_m), v_i \in W_i, i = \overline{1,m}, v_i \neq v_j, i \neq j.$$

We divide identifier elements into the regular and "stream". If the element a_i is "stream", then it must be simultaneously written in all positions of the sample vector, where the correspondent incidence matrix column contains non-zero elements. An arbitrary vector of samples (or a matrix, the rows of which are samples) will be called a schedule.

The schedule

$$((v_{11}, v_{12}, \dots, v_{1m}), (v_{21}, v_{22}, \dots, v_{2m}), \dots, (v_{k1}, v_{k2}, \dots, v_{km}))$$

will be considered correct under the conditions:

1.$\forall j \in \{1,2,\dots,m\}: \{v_{1j} \cup v_{2j} \cup \dots \cup v_{kj}\} = \{n_{j1} * a_1 \cup n_{j2} * a_2 \cup \dots \cup n_{jn} * a_n\}$,

$l * a = \{a_1, a_2, \dots, a_l\}, a_i = a, i = \overline{1,l}$.

2. $\forall i \in \{1,2,\dots,k\}: v_{ij} \neq v_{ir}, j \neq r.$,elements v_{ij}, v_{ir}, are non-stream.

Obviously, in the case when each element is included in each set only once and all elements are non-stream, the matrix of incidence is

$$\begin{pmatrix} & a_1 & a_2 & \dots & a_n \\ W_1 & 1 & 1 & \dots & 1 \\ W_2 & 1 & 1 & \dots & 1 \\ \dots & \dots & \dots & \dots & \dots \\ W_n & 1 & 1 & \dots & 1 \end{pmatrix} \quad (2)$$

Then one of the variants of the correct schedule can be written in the form:

$$\begin{pmatrix} a_1 & a_2 & a_3 & \dots & a_n \\ a_2 & a_3 & a_4 & \dots & a_1 \\ a_3 & a_4 & a_5 & \dots & a_2 \\ \dots & \dots & \dots & \dots & \dots \\ a_n & a_1 & a_2 & \dots & a_{n-1} \end{pmatrix}. \quad (3)$$

The rows of the schedule matrix consist of n permutations of the corresponding elements.

Let us have an arbitrary matrix of the daily timetable, dimension mx n. Consider the columns of the schedule matrix $(R_1, R_2, \dots R_n)$. As already discussed above, the modified incidence matrix in the presence of flows was constructed as follows. Teachers' numbers are displayed horizontally, and groups in which classes are held vertically. Each teacher is assigned a column of the matrix in which zeros and ones are recorded depending on whether the teacher has pairs in the corresponding groups. Moreover, if a teacher has a current pair, then we allocate a separate column of the incidence matrix to him, marking it (you can use an index, which is the number of flow elements). Thus, a matrix of the form is:

$$A = \begin{array}{c} \\ R_1 \\ R_2 \\ \dots \\ R_m \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & \dots & x_n \end{array} \\ \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \end{array}, \quad (4)$$

where $\alpha_{ij} = \begin{cases} 1, & when\ x_i \in R_j, \\ 0 \text{ in another case.} \end{cases}$.

Note that such an incidence matrix does not provide comprehensive information for building a schedule. After all, in the case when the teacher has several pairs in the same group or several similar current pairs, the matrix will still be 1. Therefore, we will consider a modification of the incidence matrix containing information about the number of pairs. For this, instead of 1, we indicate the corresponding number. Therefore

$$\alpha_{ij} = \begin{cases} k_i, & \text{when } x_i \in R_j, k_i - \text{numbers of the lessons} \\ & \text{else } 0. \end{cases}$$

Let's, for example, have a timetable matrix of the form:

$$R = \begin{pmatrix} a & a & a \\ b & c & a \\ a & c & d \end{pmatrix} \quad (5)$$

where a,b,c,d-teachers identifiers. It is necessary to build all possible variants of correct SDR.

Construct the incidence matrix :

$$\begin{pmatrix} & a & a^n & b & c & d \\ R_1 & 1 & 1 & 1 & 0 & 0 \\ R_2 & 0 & 1 & 0 & 2 & 0 \\ R_3 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \quad (6)$$

Let's construct the scheduling process of the modified permanent with "memorization" [1] according to the first line:

$$per\mathrm{mod}_3^{1,2,3}\begin{pmatrix} a & a^n & b & c & d \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix} = 1_1^a * per\mathrm{mod}_3^{2,3}\begin{pmatrix} b & c & d \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} + 1_1^{a^p} * 1 + 1_1^b * per\mathrm{mod}_3^{2,3}\begin{pmatrix} a & c & d \\ 0 & 2 & 0 \\ 1 & 0 & 1 \end{pmatrix} = 1_1^a *$$

$$(2_2^c per\mathrm{mod}_3 \binom{bd}{01}) + 1_1^{a^n} * 1 + 1_1^b * (2_2^c per\mathrm{mod}_3 \binom{ad}{11}) = 1_1^a 2_2^c 1_3^d + 1_1^{a^n} + 1_1^b 2_2^c 1_3^a + 1_1^b 2_2^c 1_3^d.$$

We get all possible variants of correct SDR: acd, aaa, bca, bcd.

Let's consider in more detail the procedure for scheduling a permanent. Note that a non-zero element in the row of the matrix in the first step means the mandatory presence of the corresponding component in the schedule. In the process of calculating the permanent, we simply use the addition operation +. However, from the point of view of the construction of the schedule, the logic of the addition operation here is completely different, it denotes a "disjunction", the possibility of choosing one of the options of the SRPS. If an element is multiplied, the value of which is greater than 1, then a situation of inclusion of several components is possible. For example in an expression $2_2^c per\mathrm{mod}_3 \binom{ab}{11} = 2_2^c * 1_3^a + 2_2^c * 1_3^b$ addition means the obligatory inclusion of both components, since element 3 is included twice, the element of the incidence matrix is equal to 2. At the same time, in the expression $1_2^c per\mathrm{mod}_3 \binom{ab}{11} = 1_2^c * 1_3^a + 1_2^c * 1_3^b$ adding means choosing one of two options, since element 3 is used only 1 time. We will mark the selection operation with an icon∨. In addition, when using the mandatory inclusion of the

corresponding element of the column to which the non-zero element of the incidence matrix corresponds, we will reduce the value of the element of the incidence matrix by 1. thus, the expressions in the calculation of the permanent can be interpreted not as ordinary sums, but as special operations applied to systems of different representatives or their fragments. In this case, we will no longer have the numerical value of the algebraic constant, but we will get all possible variants of timetables.

Lets consider examples;

$$1_2^d per\mathrm{mod}_3 \begin{pmatrix} a & b & c \\ 1 & 1 & 1 \end{pmatrix} = 1_2^d * 1_3^a + 1_2^d * 1_3^b + 1_2^d * 1_3^c = ADF = 1_2^d * 1_3^a \vee 1_2^d * 1_3^b \vee 1_2^d * 1_3^c$$

$$2_2^d per\mathrm{mod}_3 \begin{pmatrix} a & b & c \\ 1 & 1 & 1 \end{pmatrix} = 2_2^d * 1_3^a + 2_2^d * 1_3^b + 2_2^d * 1_3^c = ADF =$$

$$= (1_2^d * 1_3^a + 1_2^d * 1_3^b) \vee (1_2^d * 1_3^a + 1_2^d * 1_3^c) \vee (1_2^d * 1_3^b + 1_2^d * 1_3^c)$$

Last example demonstrates an important property: in the case when the value of the multiplier element is less than the number of non-zero elements of the expansion line, we haveC_n^kvariants of interpretation of the initial amount in the form of ADF.

Thus, ADF is an expression that includes two operations: + mandatory inclusion and binary selection∨.

It is easy to determine the properties of the algebra of additive-disjunctive forms:

The following properties are obvious:

1. $A \vee A = A$
2. $A \vee B = B \vee A$
3. $A + B = B + A$
4. $A + A = \{A, A\}$
5. $A + B \vee C = A + (B \vee C)$
6. $A + B \vee C = (A + B) \vee (A + C)$

Using these properties, it is possible to convert to multiple systems of various representative configurations:

$$a + b \vee c + d \vee e = (a + b) \vee (a + c) + d \vee e = (a + b + d \vee e) \vee (a + c + d \vee e)$$
$$= (a + b + d) \vee (a + b + e) \vee (a + c + d) \vee (a + c + e).$$

Now consider the previous example from the point of view of ADF. We can modify the permanent schedule procedure to obtain the correct ADF. To do this, at each iteration of the decomposition of the permanent, we will analyze the occurrence of the same elements in all components of the mandatory inclusion. The condition must be fulfilled - the total number of occurrences of the element in the mandatory inclusion block must be equal to its index. If it turns out that this condition is violated, then the mandatory inclusion block will be considered incorrect and must be removed. We have:

$$per\mathrm{mod}\begin{matrix}1\\2\\3\end{matrix}\begin{pmatrix} a & a^n & b & c & d \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix} = 1_1^a * per\mathrm{mod}\begin{matrix}2\\3\end{matrix}\begin{pmatrix} b & c & d \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} + 1_1^{a^p} * 1 + 1_1^b * per\mathrm{mod}\begin{matrix}2\\3\end{matrix}\begin{pmatrix} a & c & d \\ 0 & 2 & 0 \\ 1 & 0 & 1 \end{pmatrix} = 1_1^a *$$

$$(2_2^c per\mathrm{mod}_3 \begin{pmatrix} b & d \\ 0 & 1 \end{pmatrix}) + 1_1^{a^n} * 1 + 1_1^b * \left(2_2^c per\mathrm{mod}_3 \begin{pmatrix} a & d \\ 1 & 1 \end{pmatrix}\right) =$$

$$= \mathrm{ADF} = 1_1^a * \left(1_2^c per\mathrm{mod}_3 \begin{pmatrix} b & d \\ 0 & 1 \end{pmatrix}\right) + 1_1^{a^n} * 1 + 1_1^b * \left(1_2^c per\mathrm{mod}_3 \begin{pmatrix} a & d \\ 1 & 1 \end{pmatrix}\right) = 1_1^a 1_2^c 1_3^d + 1_1^{a^n} + 1_1^b 1_2^c 1_3^a \vee$$
$$1_1^b 1_2^c 1_3^d =$$
$$(1_1^a 1_2^c 1_3^d + 1_1^{a^n} + 1_1^b 1_2^c 1_3^a) \vee (1_1^a 1_2^c 1_3^d + 1_1^{a^n} + 1_1^b 1_2^c 1_3^d).$$

Thus, we get two variants of the timetables:

$\begin{pmatrix} a & c & d \\ a & a & a \\ b & c & a \end{pmatrix}$ and $\begin{pmatrix} a & c & d \\ a & a & a \\ b & c & d \end{pmatrix}$.

Second variant is incorrect because of 1_3^d is used twice.

Taking this into account, we can introduce a procedure for throwing out incorrect situations in mandatory inclusion blocks:

$$1_1^a 1_2^c 1_3^d + 1_1^{a^n} + 1_1^b 1_2^c 1_3^a \vee 1_1^b 1_2^c 1_3^d = 1_1^a 1_2^c 1_3^d + 1_1^{a^n} + 1_1^b 1_2^c 1_3^a \vee 1_1^b 1_2^c 0_3^d$$
$$(1_1^a 1_2^c 1_3^d + 1_1^{a^n} + 1_1^b 1_2^c 1_3^a) \vee (1_1^a 1_2^c 1_3^d + 1_1^{a^n} + 1_1^b 1_2^c 0_3^d) = (1_1^a 1_2^c 1_3^d + 1_1^{a^n} + 1_1^b 1_2^c 1_3^a.$$

In this example, in the process of building the ADF, the property is taken into account that if the element of the schedule is already used in the expression of mandatory inclusion and it is present in other expressions of mandatory inclusion, then its incident value in the following expressions is reduced by 1.

Thus, the modification of the procedure of the schedule of the permanent incidence matrix by constructing the ADF allows obtaining all possible correct options of schedules. The proposed procedure not only solves the problem of the coincidence of schedule elements in rows but also immediately obtains the required number of identical elements in columns by the input data recorded in the modified incidence matrix.

Note that in the absence of situations when the teacher can have several pairs in one group in the process of constructing the ADF, we obtain the ADF of the form (DF)+(DF)+...+(DF), where DF (disjunctive form)=$1_1^{i_1} * 1_2^{i_2} * \ldots * 1_n^{i_n} \vee 1_1^{j_1} * 1_2^{j_2} * \ldots * 1_n^{j_n} \vee \ldots$

In the case when several pairs are allowed in one group or streams, we will get additive constructions of mandatory inclusion (DAF)+(DAF)+...+(DAF), where DAF (disjunction of additive forms) contains constructions of the type

$$1_1^{i_1} * 1_2^{i_2} * \ldots * 1_n^{i_n} \vee 1_1^{j_1} * 1_2^{j_2} * \ldots * 1_n^{j_n} \vee (1_1^{j_1} * 1_2^{j_2} * \ldots * 1_n^{j_n} \vee \ldots) + (1_1^{j_1} * 1_2^{j_2} * \ldots * 1_n^{j_n} \vee \ldots) \ldots$$

The construction of ADF will be carried out based on the set of SRPS. Indeed, if the value of the incidence index of some element in the SRPS is greater than 1 and equal to k, then in this case we will have C_l^k variants of interpretation of the initial amount in the form of ADF. Moreover, any combination of SRPS with k elements will be recorded with a mandatory inclusion operation, between which there will be a selection operation. Such a procedure will make it possible to construct an ADF based on the set of all SRPS.

First, the operation of mandatory inclusion is placed between the groups of SRPS formed as a result of the schedule according to the first row of the incidence matrix. In more detail, mandatory inclusion groups are all SRPs in which the first element is the element to which the non-zero elements of the first row of the incidence matrix correspond. If the index of the element is 1, then all disjunctions will be in the group, if not, then we will have disjunctions of all possible combinations of conjunctive blocks of mandatory inclusion. For example, if the index is 2, we will have initial groups of mandatory inclusion. Next, we will review all the SRPS sequentially by the first elements (the current element is considered one), the second, etc.

If the first element in any group has an index of 1, then it is obvious that all the SRPS in the group is used with the selection operation. If its index is larger, then we will have all possible combinations of mandatory inclusion blocks connected by the selection operation. Note that the result of such a schedule is the sum of disjunctive forms, which is guaranteed to include all the necessary elements of the first column. Next, we consider all SRPS for the second element. The logic of further actions is to ensure the guaranteed achievement of the required number of inclusions of the second element.

For this, we consider the second row of the incidence matrix and all its non-zero elements. Let's consider the first element. We determine which SRPS has it. Then we define the possible options so that it is entered depending on the index. To do this, we number occurrences, generate combinations, and check the required number of occurrences, taking into account the operations of mandatory inclusions. Since there are only disjunctions in each group, with an index greater than 1, it is necessary to choose from several groups (from one only in case of coincidence). Then you can select the groups from which the corresponding SRPS is selected and then it is selected as the only one. At the same time, all other SRPS that do not contain our element is selected from other groups. We put a disjunction between the new groups. Thus, we get a disjunctive form of groups, each of which has corresponding additive constructions of mandatory inclusion.

Suppose that its index is greater. Note that the SRPS with this element can be in several groups of mandatory inclusion (see example). If it stands in only one group, where selection operations are everywhere, it is impossible to make a schedule. If it is in several different groups, then only 1 SRPS can be selected from each such group. If there are index elements and it is possible to form a schedule with the required number of occurrences of this element, then combinations with SRPS with mandatory inclusion operations are formed. At the same time, SRPS mandatory inclusion will appear! Next, we consider the third element, the fourth, etc. General algorithm was considered in [1].

Information technology

The results described above allow us to detail information technology for solving a wide range of calendar planning problems, which is based on the use of algorithms based on decomposition procedures of specially modified permanent incidence matrices.

This technology is a set of methods and approaches, which include:

• methods of input data analysis;

• methods of generating various combinatorial objects - permutations, systems of various column representatives, combinations with a number of additional conditions, which are based on specially developed permanent decomposition algorithms;

• methods of forming all possible admissible variants of matrices

• schedules, including algorithmic solutions, which are based, in particular, on the use of the algebra of additive-disjunctive forms specially developed for such problems;

• methods of presenting data during the operation of generation algorithms, which are adapted to these algorithms and take into account their specificities as much as possible.

Thus, this information technology is a complete system that allows you to solve the tasks of creating schedules in the presence of a number of additional conditions (See Fig.1).

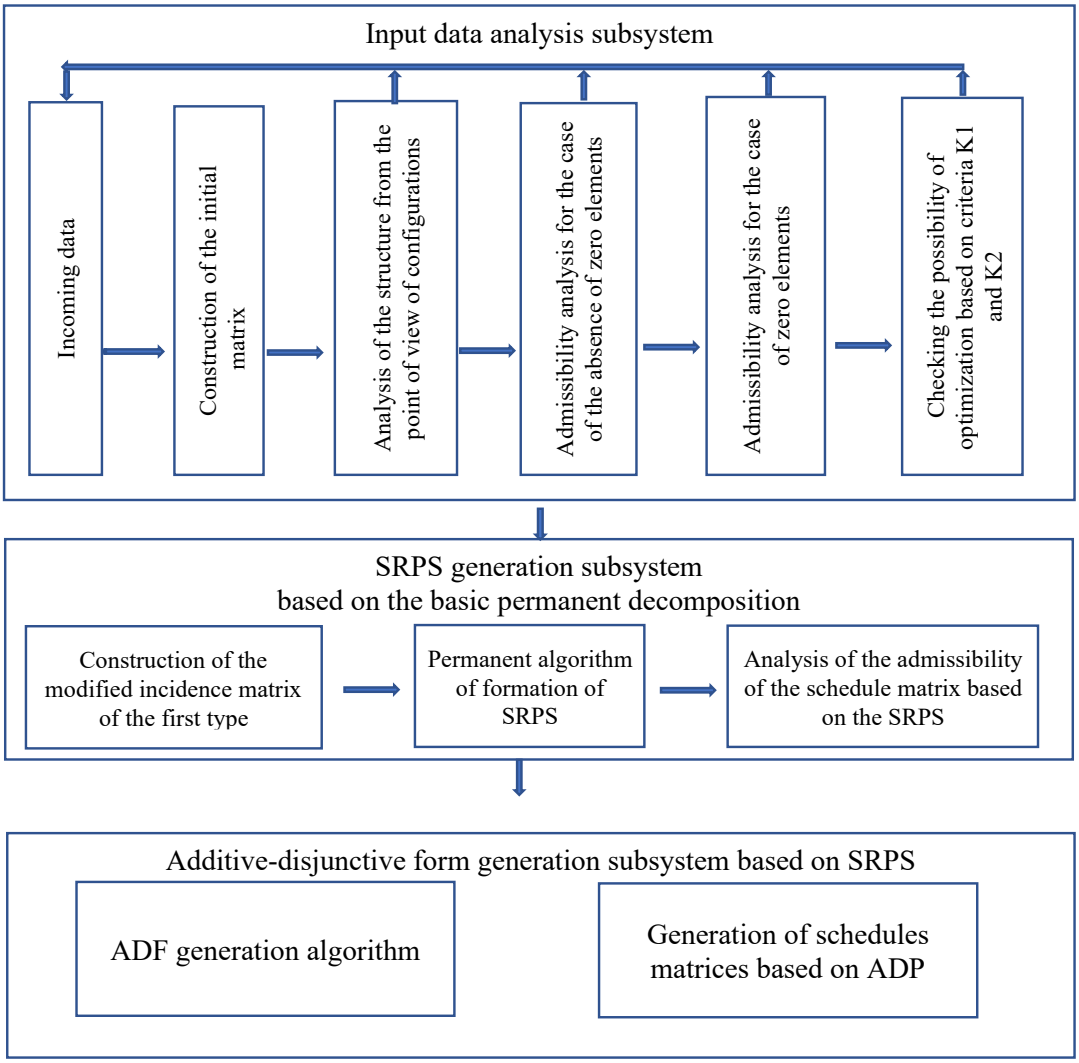

Fig. 1. Information technology based on ADF algebra (a variant of using the basic permanent approach)

Conclusions

The second modification of the incidence matrix is proposed, which differs in that, unlike the previous options, it contains complete information necessary for drawing up a schedule. The corresponding modification made it possible to make significant corrections in the decomposition procedure of the permanent of this matrix and led to the need to develop an algebra of additive-disjunctive forms.

A special algebra of additive-disjunctive forms (ADF) is proposed, which contains two operations - disjunction and mandatory inclusion (addition) and differs in the algorithmic nature of its operations. The operation of disjunction means the selection and, accordingly, the duplication of the corresponding lists containing disjunctive forms in the process of recursive generation procedures. Mandatory inclusion operation means simple inclusion of the corresponding SRPS as the next row in the running schedule matrix.

Based on ADF, two algorithms for the formation of schedule matrices were proposed for the first time:

- the first algorithm allows forming the schedule matrices directly in the process of decomposition of the permanent of the second modification of the incidence matrix;
- the second algorithm allows you to form ADF based on the SMPR, which is formed as a result of the algorithms formulated in the previous section.

This approach is very convenient from the point of view of software implementation and construction of the appropriate class hierarchy. It allows you to create information technology as a complete system that combines appropriate algorithms and methods. Information technology for solving calendar planning problems, in particular, the task of generating schedules, is formulated for the first time, which is a complete system that combines some approaches, in particular, a configuration approach to the analysis of input data and algorithms for the formation of

initial admissible matrices of schedules, the permanent approach of generation of SRPS and application of algorithms of permanent decomposition, a lexicographic approach based on relevant order relations, algebra of additive-disjunctive forms.

References

1. S. Bacchelli, L. Ferrari, R. Pinzani, R. Sprugnoli Mixed succession rules: The commutative case. Journal of Combinatorial Theory. 2010, Vol. 117, Series A. P. 568–582. DOI: 10.1016/j.jcta.2009.11.005

2. Arnaw Wadhonkar A Task Scheduling Algorithm Based on Task Length and Deadline in Cloud Computing / Arnav Wadhonkar, Deepti Theng // International Journal of Scientific & Engineering Research. – 2016. – Vol. 7, № 4. – P. 1905-1909.

3. Knuth D.E. The Art of Computer Programming / D. E. Knuth. – Vol. 4A: Combinatorial Algorithms Part 1. Boston: Addison-Wesley Professional, 2011.

4. Ruskey F. Combinatorial Generation. Working Version (1j-CSC 425/520) [Electronic resourse] / F. Ruskey, Department of Computer Science University of Victoria, 2003. Access mode: http://page.math.tu-berlin.de/~felsner/SemWS17-18/Ruskey-Comb-Gen.pdf. Accessed 1 May 2020.

5. Kruchinin V.V. Methods for Developing Algorithms for Ranking and Unranking Combinatorial Objects Based on AND/OR Trees / V.V. Kruchinin. – Tomsk: V-Spektr, 2007.

6. Shablya Y. Method for Developing Combinatorial Generation Algorithms Based on AND/OR Trees and Its Application / Y. Shablya, D. Kruchinin, V. Kruchinin // Mathematics. – 2020. – Vol. 8, № 962. DOI: 10.3390/math8060962

7. Flajolet P. A calculus for the random generation of combinatorial structures / P. Flajolet, P. Zimmerman, B. Cutsem // Theoretical Computer Science. – 1994. – Vol. 132. – P. 1–35.

8. Optimal algorithms for online scheduling with bounded rearrangement at the end / Chen Xin, Lan Yan, Benkő Attila, Dósa György, Han Xin // Theoretical Computer Science. -2011. – Vol. 412, № 45. – P. 6269–6278. DOI: 10.1016/j.tcs.2011.07.014

9. Do P.T. Exhaustive generation for permutations avoiding (colored) regular sets of patterns // P.T. Do, T.T.H Tran, V. Vajnovszki // Discrete Applied Mathematics. – 2019. – Vol. 268. – P. 44–53.

10. Mirshekarian Sadegh Correlation of job-shop scheduling problem features with scheduling efficiency // Mirshekarian Sadegh, N. Šormaz Dušan // Expert Systems with Applications. – 2016. – Vol. 62. – P. 131–147. DOI: 10.1016/j.eswa.2016.06.014

11. Humble Travis S. Application of Quantum Annealing to Nurse Scheduling Problem / S. Humble Travis, Nakamura Yuma, Ikeda Kazuki. // Scientific Reports. – 2019. – Vol. 9, № 1. – P. 12837. DOI: 10.1038/s41598-019-49172-3.

12. . Henning, K. Jansen, M. Rau, and L. Schmarje, ‘‘Complexity and inapproximability results for parallel task scheduling and strip packing,’’ in Proc. Int. Comput. Sci. Symp. Cham, Switzerland: Springer, 2018, pp. 169–180.

13. T. Chen, B. Zhang, X. Hao, and Y. Dai, “Task scheduling in grid based on particle swarm optimization,” in Proceedings of the 5th International Symposium on Parallel and Distributed Computing (ISPDC ’06), pp. 238–245, July 2006.

14. A. Salman, I. Ahmad, and S. Al-Madani, “Particle swarm optimization for task assignment problem,” Microprocessors and Microsystems, vol. 26, no. 8, pp. 363–371, 2002.

15. M. Aggarwal, R. D. Kent, and A. Ngom, “Genetic algorithm based scheduler for computational grids,” in Proceedings of the 19th International Symposium on High Performance Computing Systems and Applications (HPCS ’05), pp. 209–215, May 2005.

16. G. Malewicz, A. L. Rosenberg, and M. Yurkewych, “On scheduling complex dags for internet-based computing,” in Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS ’05), April 2005.

17. L. He, S. A. Jarvis, D. P. Spooner, D. Bacigalupo, G. Tan, and G. R. Nudd, “Mapping DAG-based applications to multiclusters with background workload,” in Proceedings of the IEEE InternationalSymposium on Cluster Computing and the Grid, pp. 855–862, May 2005.

18. R. M. Chen, S. T. Lo, and Y. M. Huang, “Combining competitive scheme with slack neurons to solve
real-time job scheduling problem,” Expert Systems with Applications, vol. 33, no. 1, pp. 75–85, 2007.

19. F. E. Sandnes, “Secure distributed configuration management with randomised scheduling of system administration tasks,” IEICE Transactions on Information and Systems, vol. E86-D, no. 9, pp. 1601–1610, 2003.

20. M. Basu, “Hybridization of artificial immune systems and sequential quadratic programming for dynamic economic dispatch,” Electric Power Components and Systems, vol. 37, no. 9, pp. 1036–1045, 2009.

21. J. Oh and C. Wu, “Genetic-algorithm-based real-time task scheduling with multiple goals,” Journal of Systems and Software, vol. 71, no. 3, pp. 245–258, 2004.

22. Z. Liu and H. Wang, “GA-based resource-constrained project scheduling with the objective of minimizing activities’ cost,” in Proceedings of the International Conference on Intelligent Computing (ICIC’05), vol. 3644 of Lecture Notes in Computer Science, pp. 937–946, August 2005.

23. N. Amjady and A. Shirzadi, “Unit commitment using a new integer coded genetic algorithm,”European Transactions on Electrical Power, vol. 19, no. 8, pp. 1161–1176, 2009.

24. S. G. Ponnambalam, P. Aravindan, and S. V. Rajesh, “Tabu search algorithm for job shop scheduling,” International Journal of Advanced Manufacturing Technology, vol. 16, no. 10, pp. 765–771, 2000.

25. S. T. Lo, R. M. Chen, Y. M. Huang, and C. L. Wu, “Multiprocessor system scheduling with precedence and resource constraints using an enhanced ant colony system,” Expert Systems with Applications, vol.34, no. 3, pp. 2071–2081, 2008.

26. W. J. Gutjahr and M. S. Rauner, “An ACO algorithm for a dynamic regional nurse-scheduling problem in Austria,” Computers and Operations Research, vol. 34, no. 3, pp. 642–666, 2007.

27. F. Zhao, Y. Hong, D. Yu, Y. Yang, and Q. Zhang, “A hybrid particle swarm optimisation algorithm and fuzzy logic for process planning and production scheduling integration in holonic manufacturing systems,” International Journal of Computer Integrated Manufacturing, vol. 23, no. 1, pp. 20–39, 2010.

28. Turbal Y. , Babych S., Bachyshyna L., Kunanets N. and Kovalchuk N.. Modification of the Permanent Decomposition Method for the Meeting Schedule Problem. The 1st International Workshop on Information Technologies: Theoretical and Applied Problems (ITTAP-2021). Ternopil, Ukraine. 2021. P.126-131

29. V. Dzhedzhula, I. Yepifanova and Y. Kravchyk, "Use of the Theory of Fuzzy Sets in Determining the Level of Enterprise Security," 2022 12th International Conference on Advanced Computer Information Technologies (ACIT), Ruzomberok, Slovakia, 2022, pp. 311-315, doi: 10.1109/ACIT54803.2022.9913150.

Yurii Turbal **Юрій Турбал**	Dr Sc (Engineering), Professor, Head of the Computer Science and Applied Mathematics Department, National University of Water and Environmental Engineering https://orcid.org/0000-0002-5727-5334 e-mail: turbaly@gmail.com	Доктор технічних наук, професор, завідувач кафедри комп'ютерних наук та прикладної математики Національного університету водного господарства та природокористуванн
Serhii Babych **Сергій Бабич**	Teacher of programming disciplines, Rivne professional college of the National University of Life and Environmental Sciences of Ukraine, Rivne, Ukraine	Викладач дисциплін програмування, ВСП Рівненський фаховий коледж Національного університету біоресурсів і природокористування України