Research of special models describing technological processes.

The technological processes, schedules, parallel algorithms etc., having some technological limitations and demanding increases of efficiency of their execution can be described through digraphs, on which the appropriate optimization problem (construction of optimal scheduling of tops digraph) can be solved.

In the class of optimal scheduling, the special place is occupied by dense scheduling, and the effective algorithms of their construction represent theoretical and practical interest.

The following problem is generally considered. The digraph G = {V, U}, V=n (n - number of tops) and parameter l, specifying limitations on length of scheduling are given. It is required to arrange tops of the graph G on l places, disposed in a line, so, that on each place stands h tops, behind exception, may be, the last place.

Definition: let's name as a parallel scheduling S of tops of the graph G such an allocation of these tops on n – places, disposed in a line, for which from $(i, j)\in U$ follows, that the top i stands in scheduling S more to the left, than top j. Then the amount of nonblank places in scheduling is named as its length and is designated - $l(S)$. The value $h(S) = \max_{1\le i\le n} |S[i]|$ is named as width of scheduling, where S[i] - set of tops standing in scheduling S on place i.

In scheduling $\underline{S}$ and $\overline{S}$ set $\underline{S}$ [i] and $\overline{S}$ [i] (i=1..., l) define [1], correspondingly, extreme left and extreme right positions of tops in scheduling S at unrestricted h.

The researched problems have the following statement.

The problem 1: Under the given graph G and option value h to construct parallel scheduling of tops digraph of minimum length. Let's designate the problem S(G, h, l).

The problem 2: Under the given graph G and option value l to construct parallel scheduling of tops digraph of minimum width. Let's designate the problem S(G, l, h).

The problem 3: Under the given graph G, option value h and periods of execution of operations d_i i=1, …, n to construct parallel scheduling of tops digraph of minimum length. Let's designate the problem S(G,h,di,l) [2].

The problems 1,2,3 in a case, when h is arbitrary, have exponential complexity. Therefore, it is interesting to find the polynomial solvable special cases.

Developing the algorithms of effective solution of the formulated problems, it is expedient to spend the preliminary analysis of the graphs. One of which results, can be obtaining of ratings of parameters h and l, which usage allows to reduce computing expenses at execution algorithm.

The rating - relation permitting to estimate length (width) of scheduling of the graph, looks like the following: t [the sign] f (p_i), i=1, …, m, where t-value (maximum, minimum, exact) parameter h or l, p_i - parameters of the problem.

Further we offer the approaches to solution of special cases of the problems 1 and 3, and also method of obtaining of a rating of width of scheduling.

Method of solution of the problem S(T, h, d_i, l).

In a base of a method the idea of the analysis of labels (p (k), q (k)), assigned to each edge of a tree, lays. "Weight" p (k) of an edge k, included in top i, is equal to the sum of durations of execution of operations d_i of all predecessors of top i, including its own. The label q (k) is equal to number of tops without entering arcs from all predecessors of top i.

First of all, in scheduling the tops, without entering arcs with maximum subtree weight.

On the basis of a method the iterative algorithm is developed. The result of application of this algorithm for solution of the problem of construction of the optimal schedule of operations of editing of boiler-house is exhibited below.

The operations and their duration are given in the following table.

N	Sort of operation	Execution time of operation (days)
1	Editing of mounts of the boiler.	2
2	Editing of mounts of gas pipes.	1
3	Editing of mounts of water pipes.	4
4	Editing of the boiler.	2
5	Editing of a control system.	1
6	Editing of gas pipes.	2
7	Editing of water pipes.	4
8	Connection of pipes and boiler through valves.	2
9	Testing of the system.	3

A digraph G (fig. 1) will be model, specifying a sequence of operations, according to the table.

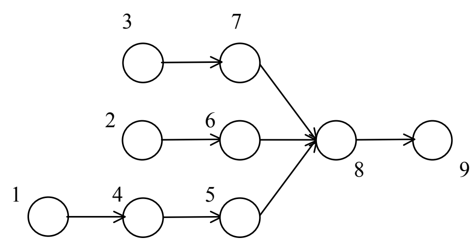

Fig.1.

As a result of operation of the algorithm is received the following schedule of operations for three performers (in each column the job(definition) for three performers for a day):

1	1	4	4	5								
2	6	6						8	8	9	9	9
3	3	3	3	7	7	7	7					

The approach to solution of the problem S(G, 3, *l*).

As the problem 1 is NP-complete, for the arbitrary graph G and arbitrary h, there are no exact algorithms of polynomial complexity. For fixed h=2 and arbitrary graph G the exact algorithms of polynomial complexity, founded on the discrimination, are obtained. However, there is open a question of existence of exact polynomial algorithms for fixed h. Therefore, it is expedient to allocate subclasses of the graphs and to develop for them polynomial algorithms for given h.

The approach to solution of the problem of construction S(G, 3, *l*) further is offered, where G the graph satisfying property: $\overline{S}[i]=\underline{S}[i]$, $i=1, \dots, \underline{l}$.

In a case, when $\underline{l}=2$, the exhaustive search of tops for solution of the problem S(G,3,*l*) does not exceed: max $(|\underline{S}[1]|, |\underline{S}[2]|)$. Let's show it.

Let's enter number k: k=mod (|$\underline{S}$[1]|/3), k can receive only three values 0,1,2.

The condition of an optimality of created scheduling will be respected, if on]S[1]/3[a place, there will be a maximum number of tops. In a case k=0 this condition is respected automatically.

Let k=1, then]$\underline{S}$[1]/3[a place it is possible to fill by one top from $\underline{S}$[1] and probably by one or two tops, by inherings $\underline{S}$[2]. That these tops could stand on one place in scheduling, on definition, between them there should not be links. As on]|$\underline{S}$[1]|/3[a place it is possible to select any tops satisfying to given conditions, for check of these conditions it is enough to calculate sequentially for each top i∈$\underline{S}$[1] a difference |S[2]|–d^{out}_i (to each top i of G will supply in accordance with actual numbers d^{in}_i and d^{out}_i equal to number of incoming and going out arcs of top i).Filling of a place number]|S[1]|/3[is realized so:

1) if for i top |$\underline{S}$[2]|-d_i^{out}≥2, we create scheduling S so, that on]|$\underline{S}$[1]|/3[a place we stake top i and any two tops from set $\underline{S}$[2]\posl, where posl - set direct successor i, place with numbers from 1 up to]|$\underline{S}$[1]|/3[-1 and from]|$\underline{S}$[2]|/3[+1 up to](|$\underline{S}$[1]|+|$\underline{S}$[2]|)/3[-1 is filled by densely arbitrary tops of sets $\underline{S}$[1] and $\underline{S}$[2] accordingly, on](|$\underline{S}$[1]|+|$\underline{S}$[2]|)/3[stake stayed mod((|$\underline{S}$[1]|+|$\underline{S}$[2]|)/3) top;

2) if |S[2]|-d_i^{out}<2, i=1, …,|S[2]| and ∃ i: |$\underline{S}$[2]|-d_i^{out}=1, on]|$\underline{S}$[1]|/3[a place we stake top i and top from set $\underline{S}$[2]\posl, where posl - set direct successors i;

3) if |$\underline{S}$[2]|-d_i^{out}=0, i=1,…, |$\underline{S}$[2]|, on]|$\underline{S}$[1]|/3[a place we stake any top i∈$\underline{S}$[2].

Other places is filled at will.

Let k=2, then]|$\underline{S}$[1]|/3[a place it is possible to fill by two tops from $\underline{S}$[1] and probably by one top belonging $\underline{S}$[2]. As well as in the first case, it is enough to calculate sequentially for each top i∈$\underline{S}$[1] a difference |$\underline{S}$[1]|-d_i^{in}, that will define required top.

That is, in a case k=1 it is necessary to touch no more |$\underline{S}$[1]| of tops, and in a case k=2 no more |$\underline{S}$[2]| of tops. Therefore, generally it is necessary to touch no more max(|S[1]|, |S[2]|) tops.

Thus instead of classical exhaustive search $C^3_{|S[1]|}$ for solution of the problem it is necessary to fulfil linear search no more n of tops.

In a case $\underline{l}$=3 the number of tops which is necessary for touching for solution of the problem by the offered way increases up to n^2.

Generally at such approach the complexity of solution of the set problem does not exceed $\prod_{i=1}^{l} |S[i]|$.

Iterative algorithm of obtaining of a rating of width of ordering by a known rating of length.

Except for effective algorithms of problem solving 1, 2 and 3, obtaining ratings of parameters of scheduling is of interest, not requiring thus of construction of optimal schedule. The exact option value l is known for the problem S(T, h, *l*). For the parameter h the ratings either rough or exacting large computing expenses are known.

For obtaining a rating of width of scheduling on an available estimator of length, we offer to use iterative algorithm of polynomial complexity, on which each step the current value of width of scheduling is set, which is used for specification of length of scheduling. Criterion of the stoppage of the iterative process is the achievement of value h, at which l does not exceed the given value.

Algorithm A.

Step 1. We set the first approximation for the parameter h through a known rating, for example h=] n/*l* [(thus it *l*-is known).

Step 2. We calculate value l(h) of a rating of length of scheduling under the given formula (for example it there can be a known rating $l(h)= \max_k (\underline{l},]\frac{1}{h}\sum_{i=k}^{l} |\underline{S}[\underline{l} - i + 1]|[+k-1), k=1,\ldots,\underline{l})$.

Step 3. We check up a relation between l given and l(h) current, if $l \geq l(h)$, the required rating is obtained; the end.

Step 4. h=h+1, transition to step 3.

Let's illustrate operation of algorithm for a digraph G given on a fig. 2, at l=5.

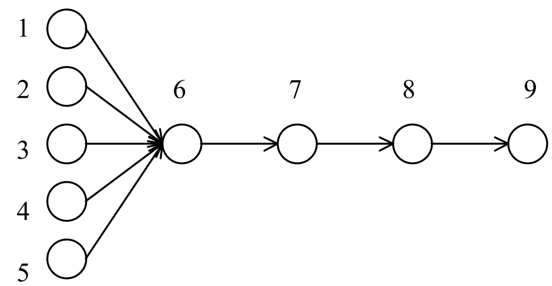

Fig. 2.

1. We shall set the first approximation h=]9/5[.
2. As the graph G - tree, we shall use a known rating of length of scheduling for trees:

$l(h)= \max_k (\underline{l},]\frac{1}{h}\sum_{i=k}^{l} |\underline{S}[\underline{l} - i + 1]|[+k-1), k=1,\ldots,\underline{l})$.

l(h) = max (] 9/2 [,] 8/2 [+1,] 7/2 [+2,] 6/2 [+3,] 5/2 [+4) =7.

3. l(h) > l, h=h+1=3.
4. l(h) = max (] 9/3 [,] 8/3 [+1,] 7/3 [+2,] 6/3 [+3,] 5/3 [+4) =6.
5. l(h) > l, h=h+1=4.
6. l(h) = max (] 9/4 [,] 8/4 [+1,] 7/4 [+2,] 6/4 [+3,] 5/4 [+4) =6.
7. l(h) > l, h=h+1=5.
8. l(h) = max (] 9/5 [,] 8/5 [+1,] 7/5 [+2,] 6/5 [+3,] 5/5 [+4) =5.
9. l(h) =l, required rating h=5.

For obtaining a rating of width of scheduling for a tree, in a case when h_i-various and $l=\underline{l}$ we offer to use iterative algorithm of polynomial complexity (algorithm B), on which each step the value of width of ordering for i of a place is defined. Criterion of the stoppage of the iterative process is the definition of a rating of width for all i.

Algorithm B.

Step 1. We define the first approximation for the parameter h through algorithm A. Thus we create a sequence pl(h), which units are the values $\sum_{i=k}^{l} |\underline{S}[\underline{l}-i+1]|$, $l^*=l$, k=0.

Step 2. We define number j of a minimum unit $pl(h)_{min}$ in a sequence pl(h).

Step 3. If $l=l^*$, h_{l-j+1}=h, we eliminate from further consideration number j, for all sequences pl(h). For all values i<j pl(h)=pl(h)-h. k=k+1, if k=$\underline{l}$, the required set of limitations is retrieved.

Step 4. If $]pl(h)_{min}/h[+j-1<l$, l=l-1 transition on 2.

Step 5. If $]pl(h)_{min}/h[+j-1>l$, h=h+1, $l^* = l$ transition on 3.

Step 6. h=h-1, if h=0, $l^*=l$, h=1 and transition on 2.

Step 7. Transitions on 4.

Let's illustrate operation of algorithm B for the graph G given on a fig. 3.

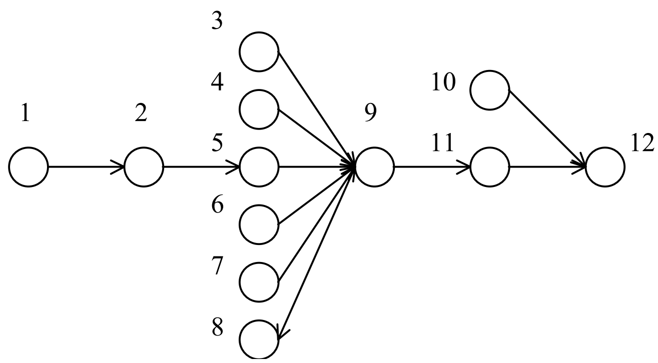

Fig. 3.

1. We define the first approximation for the parameter h (h=3) through algorithm A (see example of operation of algorithm A). A sequence pl(h)=(12, 11, 10, 9, 8, 7). l*=6, k=0.
2. j=6.
3. l=l*, $h_{l\text{-}j+1}$=3, we eliminate from further consideration number j=6, for all sequences pl(h). For all values i<j pl(h) =pl(h)-3, the sequence looks like pl(h) = (9, 8, 7, 6, 5, $\underline{7}$). k=k+1, k<$\underline{l}$.
4.] $pl(h)_{min}$/3 [+5-1=l, h=h-1, transition on 4.
5.] $pl(h)_{min}$/2 [+5-1>l, h=h+1, l*=l, transition on 3.
6. l=l*, $h_{l\text{-}j+1}$=3, we eliminate from further consideration number j=5, for all sequences pl(h). For all values i<j pl(h) =pl(h)-3, the sequence looks like pl(h) = (6, 5, 4, 3, $\underline{5}$, $\underline{7}$). k=k+1, k<$\underline{l}$.
7.] $pl(h)_{min}$/3 [+4-1<l, l=l-1 transition on 2.
8. j=4.
9. l≠l*.
10.] $pl(h)_{min}$/3 [+4-1<l, l=l-1 transition on 2.
11. j=4.
12. l≠l*.
13.] $pl(h)_{min}$/3 [+4-1=l, h=h-1 transition on 4.
14.] $pl(h)_{min}$/2 [+4-1>l, h=h+1, l*=l transition on 3.
15. l=l*, $h_{l\text{-}j+1}$=3, we eliminate from further consideration number j=4, for all sequences pl(h). For all values i<j pl(h)=pl(h)-3, the sequence looks like pl(h)=(3, 2, 1, $\underline{3}$, $\underline{5}$, $\underline{7}$). k=k+1, k<l.
16.] $pl(h)_{min}$/3 [+3-1<l, l=l-1, transition on 2.
17. j=3.
18. l≠l*.
19.] $pl(h)_{min}$/3 [+3-1=l, h=h-1, transition on 4.
20.] $pl(h)_{min}$/2 [+3-1=l, h=h-1, transition on 4.
21.] $pl(h)_{min}$/1 [+3-1=l, h=h-1, h=0, l*=l, h=1, transition on 2.
22. j=3.
23. l=l*, $h_{l\text{-}j+1}$=1, we eliminate from further consideration number j=3, for all sequences pl(h). For all values i<j pl(h)=pl(h)-1, the sequence looks like pl(h)=(2, 1, $\underline{1}$, $\underline{3}$, $\underline{5}$, $\underline{7}$). k=k+1, k<l.
24.] $pl(h)_{min}$/1 [+2-1<l, l=l-1, transition on 2.
25. j=2.
26. l≠l*.

27.] *pl*(h)$_{min}$/1 [+3-1=*l*, h=h-1, h=0, *l**=*l*, h=1, transition on 2.
28. j=2.
29. *l*=*l**, $h_{l\text{-}j+1}$=1, we eliminate from further consideration number j=2, for all sequences *pl*(h).
    For all values i<j *pl*(h)=*pl*(h) -1, the sequence looks like *pl*(h) = (1, $\underline{1}$, $\underline{1}$, $\underline{3}$, $\underline{5}$, $\underline{7}$). k=k+1, k<*l*.
30.] *pl*(h)$_{min}$/1 [+1-1<*l*, *l*=*l*-1, transition on 2.
31. j=1.
32. *l*≠*l**.
33.] *pl*(h)$_{min}$/1 [+1-1=*l*, h=h-1, h=0, *l**=l, h=1 transition on 2.
34. j=1.
35. *l*=*l**, $h_{l\text{-}j+1}$=1, we eliminate from further consideration number j=1, for all sequences *pl*(h).
    For all values i<j *pl*(h)=*pl*(h)-1, the sequence looks like *pl*(h) = ($\underline{1}$, $\underline{1}$, $\underline{1}$, $\underline{3}$, $\underline{5}$, $\underline{7}$). k=k+1, k=*l*.
    The required sequence looks like: h_i = < 3, 3, 3, 1, 1, 1 >.

Algorithm In allows to define a set of limitations hi is exact. In a case, when the graph G arbitrary, the algorithm also can be applied. As a result of operation the lower bound for first defined h_i and upper bound for the stayed limitations will be obtained.

If the graph G arbitrary also is applied some other rating, it is necessary to take into account, that in this case it is necessary to reassign a sequence *pl*(h) and accordingly conditions of items 4 and 5.

References.

1. Turchina V.A., Algorithms of parallel ordering: the manual. - Dnepropetrovsk: DSU, 1985, p. 84.
2. Tanaev B.N, The theory of the schedules. - M.: Science, 1984.