PAPER • OPEN ACCESS

Algorithm for Optimizing Construction and Assembling of Power Equipment with Low Environmental Impact

To cite this article: N L Iacobici *et al* 2019 *IOP Conf. Ser.: Earth Environ. Sci.* **219** 012011

Algorithm for Optimizing Construction and Assembling of Power Equipment with Low Environmental Impact

N L Iacobici[1], F M Frigura-Iliasa[2, *], M Mirica[2], L Dolga[1], H Filipescu[1] and M. Frigura-Iliasa[1]
[1]POLITEHNICA University Timisoara/Faculty of Electrical and Power Engineering, 300223, Timisoara, Romania
[2]National Institute for Research. and Development in Electrochemistry and Condensed Matter/LERF, Timisoara, Romania

[*]flaviu.frigura@upt.ro

Abstract. The main purpose of this paper is to elaborate an algorithm (followed by an adequate software), designed to solve power engineering optimization problems applying critical path method. All the shortcomings are cleared when using the representation of the program via a graph, where the values of the arcs are the durations of the component operations, and applying the critical path method (CPM), which mainly consists in determining a path of a maximum value between two peaks of that graph. For illustrating the algorithm and the computing program we propose an application from power engineering: a 400 kV electrical overhead line section realization. In the first part of the paper we present the application as a critical path problem. In the second part, we determine the critic path in a program graph and time reserves. In the third part, we present a representative numerical application.

1. Introduction
The main purpose of this paper is to elaborate the algorithm for a software application, designed to solve power engineering optimization problems by applying the critical path method.

We consider that a plant specialized in constructing and assembling of power equipment has to realize a 400 kV electrical overhead line section. We know the component operations that need to be executed in order to realize the final objective, their succession and conditioning, the duration and cost of each operation, the necessary workforce and the equipment. We are required to determine the total duration of executing the line, as well as the optimization from the cost, duration, necessary workforce and equipment point of view [1]. The classical solution for these problems is found via the schedule graph (the Gantt method), which presents a series of shortcomings: it does not highlight clearly the interdependencies between operations, it does not explain the temporal coincidences, it does not indicate the alternatives of declaring the various operations, it does not have a rigorous mathematical foundation, it does not allow any studies of optimization [2]. All the shortcomings are cleared when using the representation of the program via a graph, where the values of the arcs are the durations of the component operations, and applying the critical path method (CPM), which mainly consists in determining a path of a maximum value between two peaks of that graph [3]. Using CPM offers a great number of advantages: it offers a clear image of the evolution in time of the program, it allows the decreasing of the total duration of realization for the program without condensing the component operation, it highlights the operations that directly determine the duration of realization of the program, as well as those that allow the redistribution of the resources and the reduction of cost; it offers the alternative of rapid evaluation of the consequences of certain delays in realizing certain operations (without totally rebuilding the graph and the calculations); it can be easily implemented on a computer, having a solid mathematical base [4].

Applying CPM and optimizing the program is achieved via the following steps:

a) setting the list of the component operations via graphs;
b) setting the graphs in order;
c) determining the critical path (CP) and its value;
d) calculating the time stocks related to the realization of the component operations;
e) optimizing the program as from the length, the cost and the necessary resources points of view.

2. Defining the critical path

After representing the program of activities in a program graph and allocating values to the arcs, we raise the problem of determining the total duration of realization of the program. Obviously, this duration cannot be lower than the sum of the total duration of realization of the operations that compose the most unfavorable path from the initial event E_1 to the final event E_n. This path (or these paths) is called the critical path (CP). The CP in a program graph is path of maximum value between the peak corresponding to the initial event and the one corresponding to the final event. Its values are determined using algorithms with maximum value between two peaks of a graph: Ford algorithm and Bellman-Kalaba algorithm, both based on the optimality principle of Bellman, the foundation of dynamic programming.

2.1. The Ford algorithm

We consider the program graph, connected and without circuits, $G= G(X,F)$, with n peaks, x_1 and x_n being the initial, respectively the final peak. To each arc $(x_i,x_j)\epsilon A$ we assign a value $v_{ij} = t_{ij}$ (the duration of realization for the involved operation). We assume that the graph G is in order, presenting alternatives according to the Ford algorithm [5].

In order to determine the critical path, we have to go through the following steps [3]:

a) To each peak we assign a value $\lambda_i = t_i$, initialized on 0 (because we are looking for a positive maximum);
b) On step k, $k = 2...n$, we determine λ_k via the relation:

$$\lambda_i = \underset{i}{Max}(\lambda_i + t_{ik}) \tag{1}$$

λ_i being certainly non-zero, because $i < k$;

c) Step b) is repeated $n-1$ times, until we determine λ_n, the value of the critical path;
d) We determine the peak x_{di} for which the following relation is observed:

$$\lambda_n - \lambda_{d_1} = t_{d_1 n} \tag{2}$$

Storing the peak x_{di} as the critical path - CP;

e) On step I of the search in reversed order we determine a new peak on the CP, x_{di}, as being the one that complies with the following relation:

$$\lambda_{d_{l-1}} - \lambda_{d_l} = t_{d_l d_{l-1}} \tag{3}$$

f) Step e) is repeated until $x_{di} = x_I$, the CP being

$$DC = \{x_1 \quad x_{d_{l-1}} \quad ..... \quad x_{d_2} \quad x_{d_1} \quad x_n\} \tag{4}$$

In a program graph, there can be one or more CP. If there are more than one CP, the condition of maximum from the relation:

$$\lambda_i = \underset{i}{Max}(\lambda_i + t_{ik}) \tag{5}$$

For a certain peak f x_k is complied with by more peaks x_i, respectively on one or more steps more peaks x_{dk+1} comply with the relation:

$$\lambda_{d_{l-1}} - \lambda_{d_l} = t_{d_l d_{l-1}} \tag{6}$$

Then we separately analyze each solution, resulting more CP. If we apply the Ford algorithm to a program graph $t_i = \lambda_i$, represents the natural time (expected) of realization for the event E_i: the minimum time that has to run from the beginning of the program until the moment when the E_i event takes place, in order that all the previous operations can be realized in the programmed durations [6].

2.2. The Bellman-Kalaba algorithm

It is a variant of the previous algorithm, using the techniques of dynamic programming. It is based on the following property, which is the particularization of the Bellman principle: any path of maximum value of r length is composed of elementary paths of k $(k \le r)$ length and maximum value [3]. The matrix $[T]$ of the values (operating times) is defined in the following way: t_{ij} represents the duration of realization of the operation between E_i and E_j, if $(E_iE_j) \in A$, $i \neq j$, respectively $-\infty$, if $(EiEj)\ not\in A$, and $i \neq j$, whereas for $i = j$ it is null. If the *graph is in a certain* order then all the elements in the *lower* triangle of $[T]$ are $-\infty$.

$$[T] = \begin{cases} t_{ij}; (E_iE_j) \in A, i \neq j; \\ -\infty; (E_iE_j) \notin A, i \neq j \\ 0; i = j \end{cases} \tag{7}$$

Based on the statements above, the steps of the Bellman-Kalaba algorithm are as follows, the algorithm consisting mainly of determining the maximal elementary paths:

a) On the first step, we determine the elements of the auxiliary vector v^1, representing the value of the paths of 1 length from E_i to E_n:

$$v_i^1 = t_{in}, \; i = \overline{1,n} \tag{8}$$

b) On a certain step k, we determine the elements of the auxiliary vector v^k, representing the value of the paths of 1 length from E_i to E_n:

$$v_i^k = \underset{j}{MAX}(t_{ij} + v_j^{k-1}), \; i = \overline{1,n} \tag{9}$$

c) The calculation is through when the following relation is complied with:

$$v_i^k = v_i^{k-1}, \; i = \overline{1,n} \tag{10}$$

d) CP has the value v_i^k , in the hypothesis that the condition on step c) is complied with;

e) We determine the peak x_{di} for which the following relation is observed:

$$v_1^k - v_{d_1}^k = t_{1d_1} \tag{11}$$

Storing the peak x_{di} as the critical path - CP;

f) On step I of the search in reversed order we determine a new peak on the CP, x_{di}, as being the one that complies with the following relation:

$$v_{d_{l-1}} - v_{d_1} = t_{d_{l-1}d_l} \tag{12}$$

g) Step e) is repeated until $x_{di} = x_l$, the CP being

$$DC = \{x_1 \quad x_{d_1} \quad x_{d_2} \quad \ldots\ldots \quad x_{d_{l-1}} \quad x_n\} \tag{13}$$

2.3. Problems related to the use of the method of critical path

We have determined for each event E_i the natural duration (expected) of realization, noting it with t_i, as the value of the maximal path from the initial event E_1 to the event E_i. We now raise the problem is the duration of realization of a E_k event cannot be increased, without altering the duration t_n of realization of the entire program (the CP value), therefore without disrupting the subsequent operations [3]. Thus we define t_i^* the limit duration of realization of the E_i event as the latest moment on which the event E_i may take place without disrupting the completion of the program (without altering). Considering that the program graph is in a certain order, obviously $t_n^* = t_n$, the other limit duration being determined via the relation:

$$t_i^* = \underset{j}{MIN}(t_j^* - t_{ij}), (E_iE_j) \in A, i = \overline{1,n} \tag{14}$$

The fluctuation interval Δt_i is defined as the delay allowed in the realization of the event t_i, without affecting the value of the CP , and it may be calculated via the relation:

$$\Delta t_i = t_i^* - t_i \tag{15}$$

For the critical events $\Delta t_i = 0$, and for those that are not critical $\Delta t_i > 0$.

- Operation margins

Considering $t_1 = 0$, the operation represented by the E_iE_j arc cannot begin before the time t_i from the beginning of the program, respectively it cannot be completed before the time t_i^*, if we want to avoid the perturbation of the program [3]. The operation E_iE_j disposes of the time t_i^*-t_i, from which it is programmed to use t_{ij}. In case when $t_{ij} < t_i^*$ - t_i, then there is a time stock, which is called the total margin of the operation E_iE_j, and it is noted with m_{tij} , defined by the relation:

$$mt_{ij} = t_i^* - t_i - t_{ij} \tag{16}$$

The free margin ml_{ij} is defined via the following relation [7]:

$$ml_{ij} = t_j - t_i - t_{ij} \tag{17}$$

Representing the maximum duration with which it can be delayed. The free margin may be interpreted as that part of the total margin which can be used to augment of delay the operation E_iE_j without altering the fluctuation interval of the terminal event E_j:

$$mt_{ij} - m_{ij} = t_i^* - t_i = \Delta t_j \geq 0 \tag{18}$$

From the previous relations, it results:

$$mt_{ij} \geq ml_{ij} \geq ms_{ij} \tag{19}$$

ml and mt being unquestionably non-zero.

Obviously, all the operations that are part of the CP have non-zero margins, because they do not allow any delay without altering the program total duration of realization. The relation between all these parameters is presented in Figure 1:

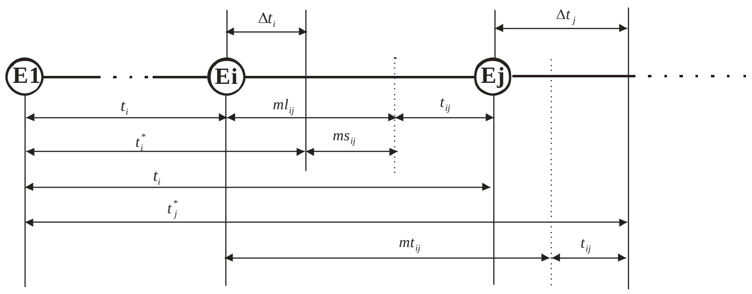

Figure 1. Relations between main parameters

3. Numerical Application

We will consider the construction of an overhead line (LEA), with all the phases of the project.

For the example stated at the beginning of this paper, the list of the main operations is given in the following Table 1, where we have also filled in the duration of each operation (in days) and the order relations between operation has been highlighted. We have considered only on expansion panel (the entire LEA will be handled analogously).

Table 1. Main construction operations

No.	Operation	Symbol	Duration	Directly conditioned operations
1	Transporting precast foundations	TF	4	SG
2	Transporting ballast for the concrete	TB	6	SG
3	Digging the holes for the foundation	SG	12	BF, MF
4	Transporting the concrete and concreting the foundations	BF	8	RS, MP
5	Mounting the precast foundation	MF	16	RS, MP
6	Transporting the poles	TS	8	AS
7	Assembling the poles	AS	17	RS
8	Raising the poles	RS	14	BC, MC
9	Transporting the conductors	TC	2	MC
10	Mounting the conductors	MC	15	VS
11	Mounting the ground socket	MP	5	-
12	Concreting the caps	BC	8	-
13	Fixing the poles	VS	7	-

3.1. The Ford Algorithm

The results obtained with the Ford algorithm are shown in the Figure 2, using the following conventions of noting and representation:

- the peaks of the graph are put in a certain order;
- next to each arc is the value $v_{ij} = t_{ij}$ (the duration for the corresponding event);
- next to each arc there are three digits, the first one stands for the natural duration of the event involved;
- on each step $x_i x_j$ arc (for which the condition of maximum is observed) has been marked with a dotted line.

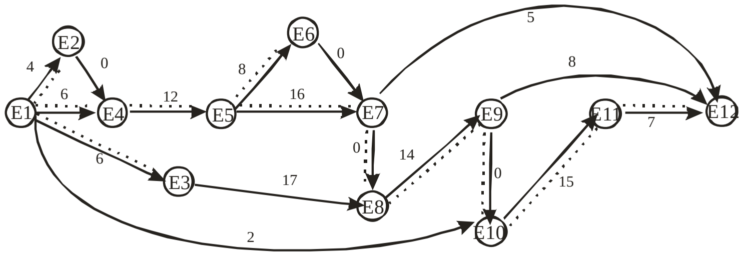

Figure 2. The operational graph

On the first step we initialize for all events $t_i = 0$. On steps 2 and 3, we obtain

$t_2 = t_1 + t_{12} = 0 + 4 = 4$; $t_3 = t_1 + t_{13} = 0 + 8 = 8$

The arcs $E_1 - E_2$, respectively $E_1 - E_3$, being marked in a dotted line on the graph.

On step 4 we determine t_4:

$t_4 = Max\{(t_1 + t_{14}) \quad (t_2 + t_{24})\} = 6$

Proceeding analogously, we determine the natural duration for each event, in the end, resulting a value of $t_{12} = 70$ days, which is the value of CP.

In order to determine the CP, we look for a path from E_{12} to E_1, which must contain only marked arcs, resulting:

$$E_1 - E_4 - E_5 - E_7 - E_8 - E_9 - E_{10} - E_{11} - E_{12}$$

As the one and only solution for the problem.

3.2. The Bellman-Kalaba algorithm

The matrix *[T]* is shown in Figure 3, being written according to the operating times. According to the presented relations, the row-vector V^1 is identical to the last row of the *[T]* matrix, its elements are standing for the value of the roads which have the *1* length from the x_i peak to the x_n peak (from the E_i event to the E_n event).

[T] =

	1	2	3	4	5	6	7	8	9	10	11	12	
1	0	4	8	6	-∞	-∞	-∞	-∞	-∞	2	-∞	-∞	1
2	-∞	0	-∞	0	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	2
3	-∞	-∞	0	-∞	-∞	-∞	-∞	17	-∞	-∞	-∞	-∞	3
4	-∞	-∞	-∞	0	12	-∞	-∞	-∞	-∞	-∞	-∞	-∞	4
5	-∞	-∞	-∞	-∞	0	8	16	-∞	-∞	-∞	-∞	-∞	5
6	-∞	-∞	-∞	-∞	-∞	0	0	-∞	-∞	-∞	-∞	-∞	6
7	-∞	-∞	-∞	-∞	-∞	-∞	0	-∞	-∞	-∞	-∞	5	7
8	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	14	-∞	-∞	-∞	8
9	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	0	-∞	-∞	8	9
10	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	0	15	-∞	10
11	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	0	7	11
12	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞	0	12

[V] =

	V^1	V^2	V^3	V^4	V^5	V^6	V^7	V^8	V^9
1	-∞	-∞	24	47	47	61	61	70	70
2	-∞	-∞	-∞	33	33	50	50	64	64
3	-∞	-∞	39	39	53	53	53	53	53
4	-∞	-∞	33	33	50	50	64	64	64
5	-∞	21	21	38	38	52	52	52	52
6	-∞	5	5	22	22	36	36	36	36
7	5	5	22	22	36	36	36	36	36
8	-∞	22	22	36	36	36	36	36	36
9	8	8	22	22	22	22	22	22	22
10	-∞	22	22	22	22	22	22	22	22
11	7	7	7	7	7	7	7	7	7
12	0	0	0	0	0	0	0	0	0

Figure 3. The [T] – matrix and the [V] – vector

The elements of the row-vector V^2 are determined for $k = 2$, being the value of the roads which have a maximum length 2 from the x_i peak to the x_n peak. Analogously we determine V^3, V^4, ..., V^9.

The condition for the ending is observed by the elements of the vectors V^9 and V^8, the value of the CP being 70. All details are provided in Table 2:

Table 2. Numerical results for the Bellman-Kalaba algorithm

No.	Operation	t_i	t_i^*	t_j	t_j^*	t_{ij}	mt_{ij}	ml_{ij}	ms_{ij}
1	$E_1 - E_2$	0	0	4	6	4	2	0	0
2	$E_1 - E_3$	0	0	8	17	8	9	0	0
3	$E_2 - E_4$	4	6	6	6	0	2	2	0
4	$E_1 - E_{10}$	0	0	48	48	2	46	46	46
5	$E_3 - E_8$	8	17	34	34	17	9	9	0
6	$E_5 - E_6$	18	18	26	34	8	8	0	0
7	$E_7 - E_{12}$	34	34	70	70	5	31	31	31
8	$E_9 - E_{12}$	48	48	70	70	8	14	14	14

4. Conclusions

In this paper, we presented an original method conceived in order to determine the critical path (CP) for some operations related to the construction of an overhead aerial line, minimizing also the impact on the environment by reducing times and eliminating dangerous operations. It finds the CP via a programmed graph and it solves a series of related problems. This algorithm could be easily implemented in a programming language and could be useful in other economy domains, too.

The following aspects related to the calculation program may be noted:

The program have to solve all aspects related to the creation and actualization of the data base related to a certain application, as well as the loading and saving of the files that contain the data bases;

It must provide a large number of check-ups for the compatibility and the accurateness of the elements in the data base, with warning or error messages if necessary;

After finding the critical path the time stocks are calculated: those related to events (the fluctuation intervals) and those related to operations (operation margins);

This research is also having a didactic side, the visualization of the results may be done according to the user's wish: from the sole display of the final results, up to the visualization in a work window of all the intermediate results for each iteration with the option of browsing them;

Human-machine interface applied for this program must be friendly with users which are not fully familiarized with all numerical and mathematical aspects of this method, but they are trying to optimize their industrial work.This method performs very well in all considered situations and could be easily applied for other large investments and complicated projects, too.

References

[1] Y. H.Song, G. S. Wang, P. Y. Wang and A. T. Johns, *Environmental/economic dispatch using fuzzy logic controlled genetic algorithms*, IEE Proc Gener Transm Distrib, Vol. 144, 1997, pp.377-382
[2] Rautela, P., Garg, D. *Case Study: Making and Monitoring of Critical Path in Shoe Production Planning*, Journal of Business Management & Social Sciences Research, 2015, Volume 4, No.3, 2319-5614
[3] Momoh J.A., *Electric Power System Applications of Optimization.* Marcel Dekker Inc., New York, 2000.
[4] J. Wood and B. F. Wollenberg, *Power Generation Operation and Control*, 2nd edn. John Wiley & Sons, New York, 1996
[5] Roa-Sepulveda and M. Herrera, *A solution to the economic dispatch problem using decision trees*, Electr Power Syst Res, Vol. 56, 2000, pp.255–259
[6] T. Voicu, D. M. Dinca, *Applying critical path method (CPM) in planning of tractors engines repairing*, International Symposium of ISB-INMA TEH. Agricultural and Mechanical Engineering, Bucharest, Romania, 2015, pp.563-568 ref.15
[7] Nafkha, R., Wiliński, A. *The critical path method in estimating project duration*, Information Systems in Management, 2016, Vol. 5 (1) 78−87