B A D A N I A O P E R A C Y J N E I D E C Y Z J E

Nr 1 2004

Dorota KUCHTA*

THE CRITICAL CHAIN METHOD IN PROJECT MANAGEMENT – A FORMAL DESCRIPTION

The critical chain method is a recent alternative to the traditional critical path method, an alternative which is becoming more and more popular, but still lacks a formal description. It exists more or less on the level of ideas and seems to have quite a lot of gaps and non-researched areas. This paper constitutes an attempt to create a formal description of the method, trying and the same time to fill in some of the gaps and to indicate a further need for OR research.

Key words: *project management and scheduling, risk management, critical path, critical chaim*

1. Introduction

E.M. Goldratt in 1997 introduced in his "business novel" [1] a new approach to project time scheduling and control, called critical chain method. Although since then several papers devoted to the subject have appeared, although there are also case studies whose authors say that they use the approach in practice and although there is even software that claims to have implemented the approach [4], it seems that there are no papers describing the method from the formal, mathematical point of view, used in the operations research. What is more, in its present form the approach constitutes in some respects no more than just an idea and much more research is needed in order to fully explore its possibilities, advantages and drawbacks. Such a research has to involve mathematicians, who would look at the problem from the formal point of view, and the main goal of this paper is to invite the OR world to enter into this research field.

The critical chain method will be described in more detail in the paper, but let us mention here its main ideas. The basic objectives of the method are:

* Institute of Industrial Engineering and Management, Wrocław University of Technology, ul. Smoluchowskiego 25, 50-370 Wrocław, kuchta@ioz.pwr.wroc.pl

• to avoid fixed project schedules, which can hardly ever be held, and are thus unrealistic and have to be corrected over and over again

• to avoid strict control of each project activity, which leads to a kind of fear of the project team members, who as a result try to protect themselves against being considered responsible for any delay

• to simplify the project completion time control and to make it at the same time more efficient, by involving project team members in the project scheduling and controlling the process in a much deeper way than it has usually been done so far.

And the most important thing is that by using the critical chain method projects should be able to be scheduled from the very beginning to last shorter (thanks to which a better competitiveness versus the competitors would be achieved). Thus, almost a miracle is expected: a shorter deadline will be kept easier and with a higher probability than a longer deadline in the traditional approach…

In order to achieve these goals, Goldratt suggests introducing several buffers into the project. He proposes to use three kinds of buffers. During the project control it is in fact only the usage of the buffers (there will usually be only few of them) that will be controlled – in this way the goal of simplifying the project completion time control is achieved. The goals of making the project time control more efficient and of shortening the project completion times should be achieved thanks to an intelligent insertion of the buffers into the project schedule and an intelligent choice of their magnitude. The buffers should protect the project against the delay risk and should warn the project manager early enough when such a delay will seem to be approaching. All of this will not be possible unless the project team has fully accepted the new approach, and maybe this human aspect will be the major obstacle in implementing the method.

In this paper, we will try to propose the first (to the author's knowledge) formal description of the method, and at the same time we will try to fill in some gaps of the rather informal descriptions presented so far. We will also point out where there is still a need for further research.

Our attempt to formalize critical chain method will not cover all its aspects. Some of them (e.g. the human factor) would be quite difficult to formalize. However, some aspects (e.g., that of the magnitude of buffers, of average and "safe" activity duration estimates) might be formalized to a greater extent than we propose here. We have chosen to limit ourselves to such a degree of formalism that might be useful in practice and, as already mentioned, we consider this paper as a mere invitation to further research.

2. Protection against delays in the traditional and the critical chain approach to project time management

Let us assume that there is given a network N composed of n vertices and m arcs. The arcs represent activities of a certain project, the nodes – the events. Each arc A_{ij} is

defined by two vertices i and j, $i, j \in \{1, ..., n\}$, $i < j$. Arc A_{ij} will also be denoted as a couple (i, j). Let us denote by S the set of arcs in the network.

The arcs can have various lengths l_{ij} $((i, j) \in S)$, that represent the estimated duration of the corresponding activities. Let us denote by $N(\Lambda)$ the network N with defined arcs lengths, where Λ is a given set of lengths l_{ij} $((i, j) \in S)$.

The traditional project time management method, the critical path method, identifies the critical path(s) in $N(\Lambda)$ and its (their) length – let us denote it by L_Λ. It also gives the earliest and latest start and finish times for each activity. This information constitutes a basis for constructing a schedule, which will be discussed in detail in section 5. The project completion time is usually estimated to be equal to L_Λ.

In the critical chain approach, proposed by Goldratt ([1]), for each arc $A_{ij} \in S$ there are defined two real characteristics: the average estimate ae_{ij} of the duration (corresponding to a situation when the activity is carried out without any greater problem, but also without surprising positive effects of things going much quicker than it was expected) and the safe estimate se_{ij} of the duration (corresponding to a situation when many things go wrong with the activity and its realization being prolonged), where obviously $se_{ij} > ae_{ij}$. Let us introduce the following notation:

$$\Psi = \{ae_{ij} : (i, j) \in S\}, \ \Omega = \{se_{ij} : (i, j) \in S\}.$$

The fact that the estimate of the activity duration is supposed to have two characteristics is a consequence of the assumption that the duration time of the activities is not known exactly and has a probability distribution, whose exact form does not have to be known – it is simply characterized by the two real values: the average one (corresponding to the expected value) and the safe one – corresponding to a number T such that probability of the activity duration being smaller or equal to T is high (equal to an assumed bound, e.g. 0,9).

Goldratt [1] claims (and this claim has been confirmed by many testimonies from the world of project management, e.g. [4]) that members of the project team, when asked to estimate how long it will take them to carry out a certain activity, reveal neither the probability distribution they have in mind (in a more or less implicit form) nor the value ae_{ij}, but only give the safe value se_{ij} [1]. This is because they feel they will be made responsible for any prolongation of the activity duration with respect to the initial estimate, so they prefer to add a safety buffer, to be pretty sure they will keep to the initial estimate even if things go wrong.

[1] Goldratt's results also show that using the PERT approach, which requires optimist, pessimist and mean duration values, is not realistic in practice.

The effect of this attitude is a cumulation of individual, hidden buffers, which unnecessarily prolong the scheduled project finish time – because if everyone uses the safety values, it means that everyone assumes that things will go wrong with their activity, and this is usually much too pessimistic. As a result, companies are much less competitive in terms of project deadlines than they would have been if members of the project staff did not conceal their probability distributions and average times ae_{ij}. What is more, those hidden, partially unnecessary buffers are not made up for later on: Goldratt gives a number of convincing reasons (e.g., the so-called student syndrome) for which people usually use up the hidden buffers, even if things do not go wrong and they do not really need them.

So, the critical chain method assumes that the company's culture makes people understand that it is in their common interest to reveal their knowledge about the distribution of the activities duration and that this information will not be used against them. Without an appropriate attitude of the team, the values required in the critical chain method, se_{ij} and ae_{ij}, will not be available or will be unreliable. Some techniques of creating the right culture are also proposed by Goldratt, but here we already assume that se_{ij} and ae_{ij} are given.

The critical chain method starts with the network $N(\Psi)$ and treats it in the classical way, using the critical path method. It identifies the critical path(s) in $N(\Psi)$ and its (their) length; let us denote it by L_Ψ. Let K denote the number of critical paths in $N(\Psi)$ and for each $k = 1, \ldots, K$ let $\{i_r^k\}_{r=1}^{l_k}$ denote the sequence of nodes in the k-th critical path, where l_k stands for the number of nodes in the k-th critical path and $i_1^k = 1$, $i_{l_k}^k = n$.

The length L_Ψ of the critical path(s) cannot be assumed to be a safe, realistic estimate of the duration time of the whole project, as it has been calculated for the network with average duration times which cannot be guaranteed to be held with a high probability. So at node n a project buffer is placed. This buffer replaces the individual buffers from the traditional approach, is a kind of compensation for the project team for their own "lost" buffers. Its role is to protect the project deadline.

In order to be able to formalize the notion of project buffer (in the literature it has not been presented in a very formal way), we have to split the notion into two. We will speak about a buffer variable x which can take on positive values and which should not exceed a certain limit B, which we will call project buffer (it may happen that x exceeds B, but such an event is considered as undesirable). The question of how the project buffer B should be calculated is central to the whole method and will be discussed in more detail later on, but the main idea is that the sum $L_\Psi + B$ is the estimated project completion time that can be more or less safely given to the customer and that it is considerably shorter than the project completion time estimated in the

classical approach (equal to L_Ω, the length of the critical path(s) in network $N(\Omega)$, used in the classical approach). What is more, the probability of keeping the shorter deadline using the critical chain approach is – according to Goldratt – much higher than the probability of keeping the longer deadline L_Ω while using the classical approach – which is explained in more detail in [1] and is due to the fact that people tend to use up the hidden buffers even if they do not need them and may not have any more buffer when a real problem occurs – whereas the usage of the clearly defined common buffer in the critical chain approach is more transparent, wise and economic. The buffer variable role is to warn the project manager if there is a danger of not keeping the project deadline.

We will show later on how the project buffer and the buffer variable are used, but before that let us explain another kind of buffers that are used in the critical chain approach. They are so called feeding buffers. Also their role is to give a compensation to the project members for the loss of individual buffers.

Again, in order to present the notion in a formal way, we will need two notions: that of feeding buffer variable and that of feeding buffer.

For each node i_r^k, $k = 1, ..., K$, $r = 1, ..., l_k$ lying on a critical path[2], which is also an end-node of at least one non-critical activity, we define a feeding buffer variable $x(i_r^k)$ with a desired upper limit of $B(i_r^k)$ (called feeding buffer). Again, the problem of choosing the right value for $B(i_r^k)$ and of using the feeding buffer variables will be discussed later. Here let us only remark that the role of the feeding buffer variables is to control (during the project realization) that there is no change in the set of critical paths, i.e. that no other paths have become critical in network $N(\Psi)$ than those already identified there.

The critical chain method introduces also "key resources buffers", which should protect the project from having to wait for the resources without which the realization of some activities would be impossible. Let us assume that there are R key resources r_l (l = 1, …, R), each of which is indispensable for the execution of a certain number of critical activities (arcs), forming for each key resource the set K_l (l = 1, …, R), $K_l \subset S$. Then for each key resource r_l (l = 1, …, R) and for each unstarted critical activity $A_{ij} \in K_l$ there will be defined a buffer BR_{ij}^l, being a positive real number, which will say how much time earlier resource r_l has to be alerted (its availability verified) before the scheduled start of activity A_{ij}.

Let us now pass on to the problem of buffer size.

---

[2] If the node belongs to more than one critical path, just one path and one couple of indices (r, k) is chosen.

3. Sizing the project buffer

Most authors of the critical chain literature, among others those of [3] and [5], suggest two methods of calculating the size of the project buffer: the Cut and Paste method and the Root-Square-Error method. In both methods the basic information is the difference $se_{ij} - ae_{ij}$ for each critical arc, as the measure of how much more time people expect to need if things go wrong with the activity with respect to the "average" situation, when things go neither very poorly nor very well. Both methods have been praised by managers as quite good (see. e.g. [4]).

However, in the literature it has always been assumed so far that in network $N(\Psi)$ there is just one critical path, i.e., $K = 1$. With this assumption, according to the Cut and Paste method , B should be equal to

$$\left[\sum_{r=1}^{l_1-1}\left(se_{i_r^1 i_{r+1}^1} - ae_{i_r^1 i_{r+1}^1}\right)\right] \Big/ 2 ,$$

i.e. the project buffer should represent half of the time gained by making people reveal the average duration times instead of giving just the safe estimates.

According to the Root-Square-Error method and with just one critical path, B should be equal to

$$\sqrt{\sum_{r=1}^{l_1-1}\left(se_{i_r^1 i_{r+1}^1} - ae_{i_r^1 i_{r+1}^1}\right)^2} .$$

This approach makes of course reference to probability theory and to the calculation of the variance of the sum of independent random variables. Some authors argue that this approach is better than the first one, because it takes into account differences in risk

$$(se_{i_r^1 i_{r+1}^1} - ae_{i_r^1 i_{r+1}^1})$$

(measured as the difference) between individual activities – and not just the total time gained by introducing the critical chain approach. Of course, this way of calculating the buffer assumes the independence between individual activities, but there are testimonies from practice saying that both methods have turned out to be quite good, even if the satisfaction of the independence and other formal assumptions are not verified.

The question of what to do in case there are more critical paths has not been addressed in the literature so far. One thing seems to be clear: even if there are more critical paths, there should still be only one project buffer: one of the goals of the critical chain method is to make the project control as easy as possible. As we will see later, the project completion time control in the critical chain method consists basically in the control of buffer variables, so if the method is to simplify the project completion time control, we should not

multiply project buffers. On the other hand, the project buffer should "amortize" the risk of each individual critical path being prolonged. Thus, we propose to insert always just one project buffer, also in case of multiple critical paths, and we propose the following formula for the project buffer in case there are more critical paths:

Cut and Paste method: $B = \max_{k=1,\dots K} \left[\sum_{r=1}^{l_k - 1} (se_{i_r^k i_{r+1}^k} - ae_{i_r^k i_{r+1}^k}) \right] \Big/ 2$

Root-Square-Error method: $B = \max_{k=1,\dots K} \left(\sqrt{\sum_{r=1}^{l_k - 1} (se_{i_r^k i_{r+1}^k} - ae_{i_r^k i_{r+1}^k})^2} \right)$

Thus, the proposal is to adopt the maximal buffer to be the project buffer, in order to protect the most risky (with respect to completion time) one from the critical paths, and automatically also all the other ones.

The algorithm to calculate the project buffer might be as follows:

Step 1: Construct a network N_R composed just of the critical paths of $N(\Psi)$

Step 2: Change the lengths of the arcs in N_R to $se_{ij} - ae_{ij}$ (Cut and Paste method) or to $(se_{i_r^k j_r^k} - ae_{i_r^k j_r^k})^2$ (Root-Square-Error method).

Step 3: Apply the critical path algorithm to find the length L_R of the longest path in N_R

Step 4: Set $B = ½ L_R$ (Cut and Paste method) or $B = \sqrt{L_R}$ (Root-Square-Error method).

The slacks that we obtain using the critical path method in the above algorithm will be an interesting by-product: a non-zero slack for a critical path will show how much less risky the given critical path is with respect to the most risky one.

Let us now pass on to the problem of sizing the other kinds of project buffers.

4. Sizing the feeding buffers and the buffers of key resources

According to the literature, the feeding buffers should protect the non-critical paths against becoming critical or alert the manager when there is a danger that a non-critical path becomes critical. As we will see in section 5, in the critical chain method it is claimed that non-critical activities should be scheduled as late as possible, but taking into account the need for protection against becoming critical. Thus it is clear that the feeding buffers should be sized with the same underlying idea as the project buffer, but with respect to the points where non-critical activities enter into critical paths. Our proposal is a consequence of this.

Let us define for each node i_r^k, $k = 1, ..., K$, $r = 1, ..., l_k$, lying on critical path and being an end node of at least one non-critical activity in $N(\Psi)$, a network $N(i_r^k, \Psi)$, being a subnetwork of $N(\Psi)$ and composed of those non-critical arcs $A_{ij} \in S$ which belong to at least one arcs sequence $\{A_{1,l_1}, A_{l_1 l_2}, ..., A_{l_s l_{(s+1)}}, ..., A_{l_p i_r^k}\}$ (p is any positive integer, $s \leq p$), thus being connected with the node in question. Network $N(i_r^k, \Psi)$ has node 1 as its starting point and node i_r^k as its terminal point and all its arcs are non-critical in $N(\Psi)$. We propose to identify in network $N(i_r^k, \Psi)$ all those paths that are critical in $N(i_r^k, \Psi)$ (the network formed by them will be called $N_{i_r^k, R}$), calculate their length (denoted by $L_{i_r^k, \Psi}$) and calculate the feeding buffer $B(i_r^k)$ by means of the same algorithm that was presented for the project buffer, in which N_R will be replaced by $N_{i_r^k, R}$. In this way the non-critical partial paths will be protected against becoming critical in the nodes they enter critical paths, and this protection will be based on the same idea as the protection of the project completion time by the project buffer.

The problem what to do if the feeding buffers create path which will be longer than the critical ones has not been answered by the authors of the critical chain method. It seems that the most cautious approach might be used, which would consist in postponing the project completion and adjusting the network in such a way that all the buffers (the project buffer and the feeding buffers) have at least the lengths calculated according to the above formulae. If this is not possible because of the customer requirements, the project manager would have to reduce the length of some feeding buffers in such a way that the corresponding paths do not exceed the critical ones and, during the project realization, pay special attention to the reduced feeding buffers.

There is no indication in the literature as to the magnitude of the key resource buffers. Probably here the sizing would have to be agreed upon with the resource in question or with its owner (supervisor) and no general formal method of calculating them is possible.

5. Scheduling

Scheduling a project means deciding when each activity should start and end. Let us introduce for each activity A_{ij} the notation SS_{ij} (scheduled start of the activity in the current schedule) and SF_{ij} (scheduled finish of the activity in the current schedule).

In the traditional approach, if the project should be terminated as soon as possible (i.e. if its duration should be equal to the length of the critical path(s)), the decision about the scheduled start and end has to be taken only with respect to non-critical activities (the critical activities must start on their earliest start times, which are equal to the latest start times), and what is more, only with respect to the start times (the finish times will follow immediately from the start times and the lengths of the activities in network $N(\Omega)$). The scheduled start times have to lie between the earliest and latest start times. In the literature two basic approaches are proposed: the ASAP (As Soon As Possible) approach – which means scheduling the start of each activity for the earliest starting time, and the ALAP (As Late As Possible) approach – which means scheduling the start of each activity for the latest starting time. It is generally recommended to use the ASAP approach – it is argued that in this way we have (for non-critical activities) activities slacks at our disposal, in case a given activity is delayed, whereas in the ALAP approach all activities become in fact critical and any prolongation delays the finish time of the whole project. Of course, there are usually infinitely many possible schedules between the ASAP and the ALAP ones.

In the critical chain approach the basis for scheduling is network $N(\Psi)$, but the length of its critical path(s) L_Ψ is not assumed to be the scheduled project duration. The project end is supposed to be scheduled for the time $L_\Psi + B$, usually considerably smaller than the scheduled project duration from the traditional approach, equal to L_Ω (this is a consequence of the way the project buffer is calculated). However, the critical activities in $N(\Psi)$ are scheduled in a "traditional" way, to start on their earliest (equal to latest) start times in $N(\Psi)$. But the non-critical activities in the critical chain approach are scheduled neither according to the ASAP nor the ALAP principle. In fact, the ASAP approach is rejected in the critical chain method, because it is claimed that carrying out activities too early means having to incur unnecessary cost for the work-in-process and storage. What is more, the scope of the project often changes with time (e.g. because the customer changes his mind or some other circumstances occur). This is why it is generally better to wait with the execution of activity as long as possible – as long as this waiting is not putting in danger the scheduled completion time of the whole project. Thus an "almost ALAP" approach is chosen – it is the ALAP approach in which the feeding buffers are used to protect the non-critical paths from becoming critical and the feeding buffer variables are used to warn about such a danger. We propose to formalize this approach in the following way:

For each node i_r^k belonging to a critical path in $N(\Psi)$, the value $\min_{A_{i_r^k j}}\{SS_{i_r^k j}\}$, where $A_{i_r^k j}$ are critical activities in $N(\Psi)$, represents a moment which cannot be scheduled earlier than the maximum of the scheduled finish times of all the partial

paths $\{A_{1,l_1}, A_{l_1 l_2}, ..., A_{l_s l_{(s+1)}}, ..., A_{l_p i_r^k}\}$ composed of non-critical activities, if these partial paths are not to form new critical paths. In the ALAP approach, it would be tried to schedule these partial paths to finish exactly at this moment[3], in the critical chain approach they should be scheduled to finish at the moment $T_{i_r^k} = \min_{A_{i_r^k j}}\{SS_{i_r^k j}\} - B(i_r^k)$, where $A_{i_r^k j}$ are critical activities in $N(\Psi)$. This fact implies our proposal of scheduling of the non-critical activities in $N(\Psi)$:

For each non-critical activity A_{lj} in $N(\Psi)$ let us identify all the nodes i_r^k belonging to a critical path in $N(\Psi)$, for which A_{lj} belongs to $N(i_r^k, \Psi)$. The set of such nodes will be denoted P_{lj}. Let $LS_{lj}^{i_r^k}$ denote the latest start of activity A_{lj} in network $N(i_r^k, \Psi)$. Then the start of activity A_{lj} will be scheduled for $\min_{i_r^k \in P_{lj}}\{T_{i_r^k} - L_{i_r^k, \Psi} + LS_{lj}^{i_r^k}\}$. In this way each non-critical activity in $N(\Psi)$ is scheduled as late as possible, but assuring that all the non-critical paths in $N(\Psi)$ remain non-critical even if the non-critical activities in them are delayed to some extent – within the feeding buffers.

The use of the key resources buffers in project scheduling can be explained in the following way: for each critical activity A_{ij} for which there exists a key resource r_l ($l = 1, \ldots, R$), for the moment $SS_{ij} - BR_{ij}^l$ there is scheduled an alert that is to be sent to the key resource or its owner (supervisor), in order to make sure the resource will be ready on time.

6. Realization control

In this section we will discuss the problem of how to control the project realization once the project is started. It is normal that reality usually differs from the plans. If theses differences do not affect the scheduled project completion time, there is no reason to worry about them, but it is important to be alerted early enough if the differences from initial plans constitute a danger for the project to be completed on time.

Let us start this section with a new notation: for each activity A_{ij}, AS_{ij} will stand for the actual start of the activity, AF_{ij} for the actual finish time of the activ-

[3] Sometimes of course, because of arc dependences, it will not be possible for all those partial paths to finish exactly at this moment – in such a case some of them will be scheduled to finish earlier, but still as late as possible.

ity and Ad_{ij} the actual duration ($Ad_{ij} = AF_{ij} - AS_{ij}$). Of course, once AS_{ij} and AF_{ij} are known, the scheduled times SS_{ij} and SF_{ij} as well as the estimates of the duration time ae_{ij} and se_{ij} have only an informational value. In case an activity has started, but not finished yet, quite exact estimates of Ad_{ij} and AF_{ij} may (but do not have to) be known. If this is the case, they will be denoted, respectively, by eAd_{ij} and eAF_{ij}. The symbol *ASP* will stand for the moment when the project actually started.

In the traditional approach we build up a basic (initial) schedule before the project starts. Once the actual realization of the project begins, the actual start and finish times of individual activities are systematically compared to the values from the initial schedule. If a difference occurs, it is stated that we deal with a variance. If the variance goes beyond the free slacks calculated in the critical path method and influences the scheduled start times of the activities that have not been started yet, a new current schedule is built up, because the old one cannot be kept any more. On each control day a new current schedule may come into being (there may be calculated new values of SS_{ij} and SF_{ij}). In case of serious variances, especially those which risk to delay the project finish time, people may be made responsible for them and may be asked to explain why they have occurred.

One of the main goals of the critical chain method was to simplify the project completion time control. Thus building up new current schedules because of variances in previous schedules as well as controlling each individual activity are avoided. The main idea is that it is only the buffer variables that should be controlled (which are not very numerous), and a new current schedule is to be built up only in case these variables are very close to or exceed their bounds – the buffers. As managers using the critical chain method claim ([4]), a need for elaborating a new schedule comes about only in a small portion of projects, for the majority of them the initial schedule controlled by means of the critical chain approach is enough.

The project completion time control in the critical chain method can be – in our opinion – presented in the following way: on each control day we build up the current set of lengths of arcs Λ, taking Ad_{ij} for the completed activities, ae_{ij} for the non-started activities and eAd_{ij} or (in case this value is not known yet) ae_{ij} for the activities that have started but not finished yet[4]. The collection of these data can be carried out almost automatically and does not require much effort.

[4] Of course, it may happen – especially in the terminal phases of the project realization – that also for the unstarted activities we can already give an estimate eAd_{ij} instead of ae_{ij}.

Then in the networks $N(\Lambda)$ and $N(i_r^k, \Lambda)$ (defined as described above for all the nodes i_r^k, $k = 1, ..., K$, $r = 1, ..., l_k$, lying on the critical paths in $N(\Psi)$[5]), the lengths of the critical paths have to be calculated, denoted by L_Λ and $L_{i_r^k,\Lambda}$, respectively. L_Λ represents a new current estimate of the project duration time, which takes into account what has already happened and what has become known about the activity duration times by the control day. $L_{i_r^k,\Lambda}$ represents the maximal actual or estimated[6] length of partial non-critical paths in $N(\Psi)$ running into critical paths in node i_r^k.

Subsequently the buffer variables are set: the buffer variable x to $L_\Psi + B - L_\Lambda$, the feeding buffer variables $x(i_r^k)$ to $T_{i_r^k} - (ASP + L_{i_r^k,\Lambda})$. $L_\Psi + B - L_\Lambda$ represents the difference between the scheduled project completion time and the current estimate of the project completion time. For each node i_r^k, $T_{i_r^k} - (ASP + L_{i_r^k,\Lambda})$ represents the difference between the moment when the earliest ones from all the critical activities starting in i_r^k are scheduled to start and the current estimate of the moment when the partial non-critical paths in $N(\Psi)$ ending in i_r^k will finish (or the exact information when they have finished).

If x is more than 2/3 of B, it is assumed that no reaction is needed – only less than 1/3 of the project buffer has been used and it is assumed that the project team will manage to complete the project in time. If x is between 1/3 and 2/3 of B, an analysis of the situation might be useful – quite a lot of the buffer has been used and it might be a sign of serious problems approaching. If x is less than 1/3 of B, it is recommended to take serious actions – the disaster may be approaching – if we consider the non-keeping of the project scheduled completion time $L_\Psi + B$ to be a disaster. If the project buffer variable x is negative, the situation is still more serious – the critical path in the current schedule is longer than the project scheduled completion time $L_\Psi + B$. Such a situation would be rather rare, as normally the buffer variable would have warned us in the previous control moments that there might be a problem and the project would have been rescheduled.

In a similar way we control the feeding buffer variables $x(i_r^k)$. The smaller the value of $x(i_r^k)$, the higher the danger that the set of critical paths will change with respect to network $N(\Psi)$. If a $x(i_r^k)$ is negative, it means that it has already happened – a new schedule, new buffers etc. are needed. But again, the testimonies

[5] Thus, all the time the critical paths from the initial schedule.

[6] Even if it is an estimate, it is a new current estimate, usually more exact than the initial one, because it uses some information about the actual duration times.

from the practice show that this happens rather rarely and the rescheduling is much less frequent while using the critical chain approach than it is with the traditional approach.

Of course, the limits 1/3 and 2/3 are just a convention. The critical chain literature adopted these limits after Goldratt [1], but in fact these limits should somehow be linked to how far from the project end we are. 2/3 of the project buffer used is rather dangerous at the project beginning, but not so much near its end.

Apart from the formal side of project control, the essential aspect of the success or failure of the critical chain approach is the attitude of the project team. On each control day they should be informed about the degree to which the project buffer or the feeding buffers have been used up and have to behave accordingly – if the buffers have been used up to a high degree and the end of the project is not so close, the project team has to try to keep to the average estimated duration ae_{ij} of the activities they still have to carry out, and even try to be better than that – in order to give some of the used buffer portion back, to be available to other activities. The individual activities do not have to be controlled by the project manager because and under the condition that the project team members organize things among themselves, being aware of the importance of the buffers and treating them as a "common good" that has to be shared.

The of key resources buffers are used as alerts – whenever one of the moments $SS_{ij} - BR_{ij}^{l}$ has come, the alert has to be sent. The project control will have to check whether all the alerts due up to the control moment have been sent and whether the reaction to them has been appropriate.

7. Critical chain method and resources

It is only in this section that we will explain the very notion that has given its name to the method – that of critical chain. We can say that the critical chain is a critical path in $N(\Psi)$ after the resources needed by all the activities have been taken into account and verified against the number of resources available (i.e. after resource leveling). We have assumed that there are no resource overallocations and thus the critical paths in $N(\Psi)$ are at the same time critical chains. The reason for this assumption was that resource leveling in itself is not a simple problem and the question of identifying the critical chains is presented in the critical chain literature only on the idea level (with some simple examples). One of the main problems – on the computational and on the conceptual level – will probably be the possible multiplicity of solutions (of various sequences in which activities can be placed in the critical chain after resource leveling). A formal method for determining "the best" critical chain would still have to be searched for.

8. Conclusions

In this paper we have tried to make a first step towards a formal description of a recent project control time method – the critical chain method. At the same we have filled in some and pointed to some other gaps in what has been said in the corresponding literature so far.

The method still needs a lot of OR research. The first problem was addressed in Section 6 – how to identify the critical chains in case of resource conflicts. Another problem was mentioned with respect to project control – how to relate the "critical" level of buffers usage, after which an action has to be taken, to the progress in the project realization.

And there is of course the whole human aspect of the method. If the human factor does not work, the method for sure will not either. Can the project members be given more formal information pieces about what they are expected to do and what they should expect? By which formal control system they can be better motivated?

The critical chain method does seem to be very interesting and useful. Further OR research may enrich it and make its application easier and wider.

References

[1] GOLDRATT E.M. (1997), *Critical chain*, The North River Press, Great Barrington (MA).
[2] LEACH L.P. (1999), *Critical Project Management Improves Project Performance*, Project Management Journal, Vol. 30(2), 39–51.
[3] MCMULLEN T.B., Jr. (1998), *Project Management in the Fast Lane*, The St. Lucie Press/APICS, Boca Raton, London, New York, Washington D.C.
[4] www.prochain.com – WEB site devoted to the critical chain method.
[5] STEYN H. (2001), *An investigation into the fundamentals of critical chain project scheduling*, International Journal of Project Management, 19(6), s. 363–369.

Metoda łańcucha krytycznego w zarządzaniu projektami – opis formalny

W 1997 roku E. Goldratt zaproponował alternatywę dla tradycyjnego podejścia do zarządzania czasem projektu i harmonogramowania – metodę łańcucha krytycznego. Jest to podejście obiecujące, na co wskazują coraz liczniejsze głosy z praktyki. Niestety, brakuje formalnego usystematyzowania metody, analogicznego do formalnego sformułowania problemu ścieżki krytycznej. Metoda łańcucha krytycznego istnieje raczej na poziomie idei, które w mniejszym lub większym stopniu są wcielane w praktyce. Jej opisy w literaturze zawierają sporo luk. W niniejszym artykule zaproponowano zatem formalny zapis metody. Uzupełniono w nim wiele luk istniejących w dotychczasowych opisach metody, podano formalne wzory do wyznaczania podstawowych elementów zarządzania projektem przy zastosowaniu metody

łańcucha krytycznego: buforów, ich wielkości, przewidywanej długości realizacji projektu – które pozwalają na stworzenie komputerowej implementacji metody. Jednocześnie wskazano na trudności w formalnym zapisie niektórych aspektów metody – dotyczących zarządzania zasobami. O ile stosunkowo łatwo jest zapisać ideę zarządzania zasobami projektu w metodzie łańcucha krytycznego, o tyle formalny zapis – bez którego pełna komputerowa implementacja metody byłaby nie możliwa – wymaga dalszych badań.

Słowa kluczowe: *zarządzanie projektami i harmonogramowanie, zarządzanie ryzykiem, ścieżka krytyczna, łańcuch krytyczny*