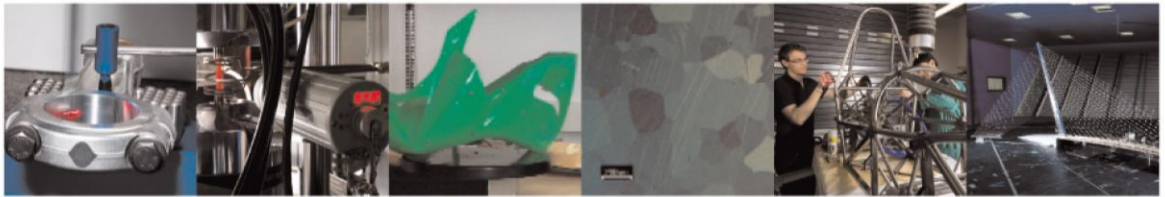




POLITECNICO
MILANO 1863

DIPARTIMENTO DI MECCANICA



Makespan estimation of a production process affected by uncertainty: Application on MTO production of NC machine tools

Manzini, Massimo; Urgo, Marcello

This is a post-peer-review, pre-copyedit version of an article published in JOURNAL OF MANUFACTURING SYSTEMS. The final authenticated version is available online at:

<http://dx.doi.org/10.1016/j.jmsy.2015.10.001>

This content is provided under [CC BY-NC-ND 4.0](https://creativecommons.org/licenses/by-nc-nd/4.0/) license



Makespan estimation of a production process affected by uncertainty: Application on MTO production of NC machine tools

Massimo Manzini*, Marcello Urgo

Politecnico di Milano, Mechanical Engineering Dept., Milan, Italy

Received 18 June 2015

Received in revised form

22 September 2015

Accepted 8 October 2015

Available online 1 November 2015

1. Introduction

The European industrial sector is evolving towards an increasing customization of the products to provide tailored solutions to the customers. Companies competing on the market have to design and produce products that can be easily customized and, consequently, also the associated manufacturing processes must be adapted accordingly. Hence, the series production paradigm is no longer viable being each product a new one with specific characteristics and its production process must be specifically tailored as well. All these factors make the production phase more complex to manage and plan.

To this aim, companies often adopt a *Manufacturing-To-Order* (MTO) approach, producing an item only when a customer place an order, without keeping any inventory of finished parts. In many cases, the MTO scheme is also extended to the upstream phases, i.e., the design, leading to the *Engineering-To-Order* (ETO) paradigm,

where also part of the design as well as the procurement of components and/or raw materials are specifically linked to an order. At the planning level, a MTO or ETO organization entails a high level of uncertainty, because of the unpredictability of the production activities the company have to execute in the medium and long terms. An additional factor of uncertainty is incurred when production activities are executed by human operators, thus increasing the complexity of the production planning problem.

Strictly linked to the production planning phase is the definition of the due dates for the delivery of orders to the customers. The estimation of the due dates is inevitably affected by the intrinsic uncertainty at the production planning level. To support this phase, tools and methodologies are needed to assess the duration of the whole production process, the makespan, taking into consideration the stochastic characteristics of the problem. Moreover, these tools could be also used during the production phase, to estimate the probability to be late in relation to a given due date, thus triggering possible reactive approaches to reschedule the production.

Many approaches have been proposed in the literature to face these situations, most of them are based on *Project Evaluation and Review Technique* (PERT) and *Critical Path Method* (CPM) techniques. These approaches, described in the following Section, consider that

* Corresponding author. Tel.: +39 3428299425.

E-mail addresses: massimo.manzini@polimi.it (M. Manzini), marcello.urgo@polimi.it (M. Urgo).

all the production activities have a stochastic duration modeled through a random variable.

In this paper, we propose a method to estimate the distribution of the makespan of a generic set of production activities and the associated precedence constraints modeling a production process. This method can be used to estimate the lateness in relation to a *due date* as well as the scheduling of the entire process. In comparison to *PERT* and *CPM* techniques, the proposed approach can handle a wider range of stochastic distributions modeling the duration of the activities and solve general project networks in a reasonable time. Since no specific constraints are given for the distribution of an activity's duration, it can be shaped starting from the investigation of real activities in the manufacturing process (e.g. real assembling or machining activity) and, unlike the *PERT* method, avoid unnecessary and constraining hypotheses on the distribution type. The contribution of the proposed approach relies in the joint utilization of both analytical and simulation approaches already available in the literature. Moreover, the proposed approach is not affected by the typical limitations of available analytical techniques, requiring strong hypotheses on the stochastic distributions that can be used and, hence, is able to estimate the makespan of a generic process with performances that are better than pure simulation approaches.

In addition, we propose and demonstrate the exploitation of this class of approaches as a management tool, e.g., in the negotiation of delivery dates or to assess the most convenient scheduling policy. The relevance of this class of approaches is motivated by the application to specific manufacturing sectors, e.g., the production of machine tools, where production and/or assembling activities are often executed by human operators, thus entailing the need of adequately considering uncertainty.

To demonstrate the viability of the proposed approach, we provide the application to an ideal case as well as to an industrial case, provided by *MCM S.p.A.*, an Italian company that designs and produces *CNC* machines and *FMSs*. *MCM*, as many European machine tool builders, opted for a high level of customization for its machines and, hence, its production environment perfectly matches the capability of the proposed approach and provides an ideal application case.

In Section 2 we provide a survey of the related literature and, in Section 3, we formalize the addressed production problem. Section 4 contains a step-by-step description of the proposed approach. Section 5 provides a validation and testing of the proposed method on an ideal case: the results are analyzed and compared with a benchmark distribution to assess the quality of the estimation as well as the performance of the approach in terms of the time needed to solve the considered instances. In Section 6 an application to an industrial case is provided; the assembly process for the production of *NC* machines is modeled and the proposed approach is tested on a set of production orders. Additional details related to the production process under study are available in Appendix A.

2. State of art

The need to cope with the uncertainty affecting the execution of a project entails modeling the duration of the activities with a random variable. Stochastic project networks, also called *PERT* networks [16] can be used, traditionally modeled using a *Directed Acyclic Graph* $D = (N, A, p)$ where $A = \{1, 2, \dots, m\}$ is the set of arcs representing the activities, $N = \{1, 2, \dots, n\}$ is the set of nodes representing events, also called milestones and p is a vector of independent random variables, modeling the duration of the activities. The benefit of this technique stays in the estimation of the duration of complex projects: a very difficult activity, especially when all the activities are uncertain. The *PERT* technique grounds on few

simple assumptions: (a) the process network is formalized with strictly rules concerning the representation and the precedence constraints between activities; (b) the stochastic characteristics of the activities are modeled starting from three different data, the optimistic, the pessimistic and the most likely values taken from a beta distribution modeling the duration of the activity.

As a result, the graphic formalization of *PERT* networks is a digraph in which each activity is represented with an arc and each project event is represented with a node; in this case, the digraph is called *AoA* (*Activity on Arc*), whose characteristics are described in Elmaghraby [7].

Grounding on this particular formalization of a project, several tools have been developed to calculate different characteristics of the problem, first of all, the duration of the entire project using the *CPM* technique. This technique exploits the formalization of the project network to estimate its duration by the identification of the most critical path and considering only deterministic durations.

The main advantage of the *CPM* and *PERT* techniques is the capability of considering the entire project but, on the other hand, it has some disadvantages. Firstly, the estimation provided is only a single value and not a probability distribution. The approach is able to take into consideration only one path and, in addition, the only distribution supported is the *Beta* distribution.

In order to have a better and more general estimation of the makespan of this class of project networks, it is desirable to:

- obtain an estimation of the whole probability distribution of the makespan, because some regions of the distribution (e.g. the last percentiles) are more critical, and the expected value is not enough to assess the impact of these situations;
- take into consideration all the paths in the network (and not only one), to capture the stochastic behavior of the whole process.
- be able to consider multiple type of probability distributions.

Several approaches have been proposed to address these limitations.

A first set of computational problems in the evaluation of stochastic project networks comprehends the identification of the critical paths and the critical activities and the estimation of the probability distribution of the project's completion time (or stochastic bounds for it). It must be noticed that, for a stochastic network, the duration of the entire project is also a random variable, critical paths can be more than one and, for each of them, the probability to be critical should be estimated.

Two methods for the estimation of the stochastic duration of the project have been developed exploiting the concept of simplification and reduction of a network. Martin [17] and Valdes and Tarjan [23] face this problem proposing methods to reduce a complex stochastic *AoA* graph to a network with only two nodes and a single arc by the iterative application of series and parallel reduction steps, grounding on convolution and point-to-point multiplication (respectively). The distribution of the duration of the remaining arc is the distribution of the makespan of the entire original network.

The proposed approaches can reach their goal only on series-parallel networks, i.e., networks where the series-parallel reduction can be completely executed. Two activities (edges) are in series if the sink node of the first activity is also the source node of the second activity and, in particular, this node has in-degree and out-degree equal to 1 (Fig. 1(a)). In this case it is possible to substitute a single activity to the original two, whose duration is the sum of the two durations, this is a series reduction. For example, given two activities i and j in series, with duration d_i and d_j , respectively, they may be reduced to a single activity k whose duration d_k is the sum of the two starting activities $d_k = d_i + d_j$. In the case of stochastic

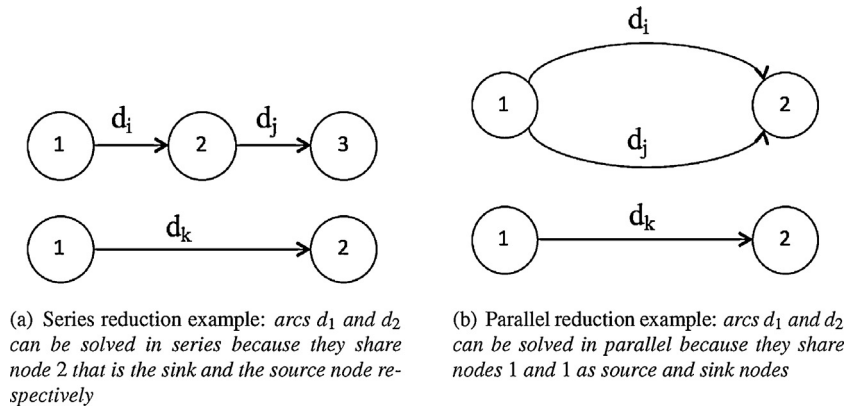


Fig. 1. Series and parallel reductions.

durations, the cumulate distribution function (cdf) of the sum, $F_k(t)$, is the convolution of the individual cdfs:

$$F_k(t) = F_i(t) * F_j(t) = \int_0^t F_i(t-s) dF_j(s)$$

On the contrary, two activities are in a parallel when they share the source and sink nodes (Fig. 1(b)). In this case it is possible to simplify the network obtaining a single arc with two nodes through a parallel merge. The duration of this arc is the maximal duration between the two original activities. For example, given two activities in parallel, named i and j , with duration d_i and d_j , respectively, their parallel reduction k has duration $d_k = \max(d_i, d_j)$. In the case of stochastic duration, given the cdfs of the initial activities $F_i(t)$ and $F_j(t)$, the cdf of the resulting activity $F_k(t)$ is:

$$F_k(t) = F_i(t) \cdot F_j(t)$$

The methods proposed by Martin [17] and Valdes and Tarjan [23] are very general and can provide a solution independently from the type of probability distribution for the duration of an activity. Nevertheless, a limit exists due to the fact that a complete series-parallel reduction of a general project network is not always possible, because not all the networks have a topological structure made up of parallel and series arcs only. In some cases, the reduction process reaches a graph that cannot be further simplified. Its name is *interdictive graph* and it can be demonstrated that a project network containing it is irreducible [8]. An *interdictive graph* has a topological characteristics such that some activities are both in a series and a parallel (in Fig. 2 a typical example is provided), namely the arc d_3 is both in a parallel with arc d_2 and in a series with arc d_1 . Once a network contains an interdictive graph, both Martin's and Valdes' methods cannot completely reduce the network and have to stop.

Many authors have addressed solutions for this problem, that can be divided in two main classes: simulation solutions (estimating the makespan using simulation tools) and bounding

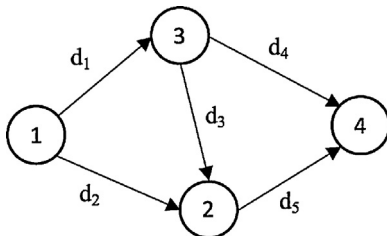


Fig. 2. An example of *interdictive graph*, in with this situation it is not possible to identify an univocal choice between series and parallel reduction.

solutions (calculating an upper and a lower bound of the real distribution).

Referring to the second class, the main results have been developed in [11] (lower and upper bounds) [6] (upper bound), and [19] (lower and upper bounds). All these methods ground on structural modifications of the network and hypothesize the independence of the activities durations. The basic idea is to modify the given network to guarantee the series-parallel reducibility. In particular Dodin's algorithm provides an estimation by multiplying a specific set of arcs in the *interdictive graph*, while the other two approaches construct an approximate distribution grounding on all the paths in the original network.

Referring to the class of simulation solutions, [4] propose an approach using a *Monte Carlo* simulation method while in [15] the estimation of the distribution exploits the algorithm provided in [5] to calculate an approximate probability distribution through the K most critical paths. In particular Ludwig's algorithm also takes advantage of the *Central Limit Theorem* and relies on the approximation with a *Normal Distribution*.

Grounding on the state of the art and the described limitations of the available approaches, we propose an approach to estimate the probability distribution of the makespan for a generic network of activities whose duration can be modeled through general probability distributions. The proposed algorithm is independent from the irreducibility of the network through series-parallel reduction and combines simulation and analytical methods. A similar approach is proposed in Tolio and Urgo [21], in which it is used a *Continuous Time Markov Chain* with a *Phase Type* approximation for the activity distributions, in order to estimate the distribution of the makespan of a set of operations; the approach is applied on a machining center production process.

The methods already exposed will be useful for estimating the makespan of a given production process, but, as anticipated in Section 1, this class of approaches also have a role as management tools in planning and scheduling production activities and material requirements. Alfieri [1] address this problem in a *MTO* environment proposing a connection between the expected cost (represented by the total tardiness or the number of late jobs) and the quoted lead time. Similarly Kaminsky and Kaya [10] proposed a model for quantifying the impact of manufacturing suppliers on the effective due date quotation. In particular, they formalize the supply chain as a two-machine flow shop and consider several variations of the scheduling in order to minimize a function of the tardiness. Instead, Guhlich et al. [9] connects the decision of accepting orders with the scheduling and due date quoting estimation in an *Assembly-To-Order (ATO)* environment. There is another branch of research addressing the need to face the uncertainty, typical in the *MTO* sector, at the production planning, scheduling and material

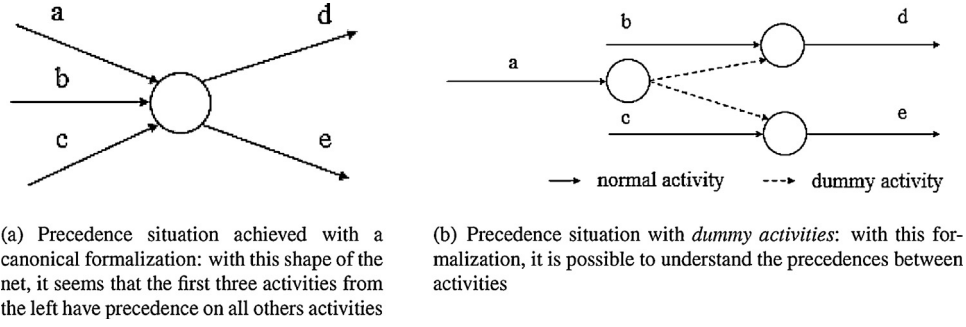


Fig. 3. Dummy activities utilization.

requirement planning level. Alfieri et al. [2] proposed a method for the estimation of a temporal range for the execution of the production activities in order to plan the material procurement. A second work from the same author [3] proposes a *two-stage* stochastic programming approach for the production planning under uncertainty. Another research, presented in Tolio and Urgo [20], faces the problem of planning the outsourcing in a ETO environment, again with a *two-stage* stochastic programming approach used in a rolling-horizon architecture. Regarding the control of uncertainty or, better, the risk mitigation approaches, Radke et al. [18] face the problem of jointly considering scheduling and inventory decisions, in terms of a comprehensive measure of risk considering both a stock out of a components and the possible impact on the completion time of a project. Tolio et al. [22] face uncertainty by a stochastic scheduling approach grounding on a branch and bound method; the main peculiarity of this work is that it deals with uncertainty by using a risk measure, the *Value-at-Risk* and the *Conditional Value-at-Risk*, typically used in the portfolio management research area.

3. Problem statement

We consider an AoA network formalized through a *Directed Acyclic Graph*, $D = (N, A, p)$, with A being the set of generic activities (arcs) and N the set of generic project events (nodes); in the formalization presented above, each activity starts from a node and arrives in another node, this structure gives a set of precedence constraints. This formalization grounds on the AoA network scheme and follows the set of characteristics defined in [7].

Each activity $a \in A$ is associated to an independent random variable p (discrete or continuous) modeling its duration. We consider two classes of activities: real operations and *dummy* activities, useful to model specific precedence constraints, whose duration is equal to 0. A *dummy* activity will be used when a single node has multiple input and output arcs but not all the activities in input affect the activity modeled by the departing arcs, as described in the following example. Let us consider a set of activities $N = \{a, b, c, d, e\}$ and precedence constraints $A = \{(a, d), (b, d), (a, e), (c, e)\}$. Using an AoA representation, we obtain the graph in Fig. 3(a), i.e., a single node with three input activities and two output activities.

However, the graph in Fig. 3(a) does not provide a correct representation of the precedence constraints. In fact, all the output activities, d and e need the completion of a , b and c to start, thus over-constraining the original problem. A correct representation can be obtained splitting the node in multiple *dummy* nodes to match the right precedence relations. The AoA network in Fig. 3(b) is obtained introducing a *dummy activity* with duration equal to zero, correctly representing the precedence constraints.

4. Solution approach

Grounding on the series-parallel reduction approach proposed by Martin and Valdes [17,23], the proposed approach aims at overcoming the limitations caused by the presence of an *interdictive graph* to obtain a network with only two nodes and a single arc whose distribution represents the makespan of the whole original network. As described above, the presence of an *interdictive graph* in the network causes the interruption of the series-parallel reduction procedure. Hence, the proposed algorithm addresses these specific cases.

The main idea is to partition the original network (containing the *interdictive graph*) in sub-networks (two or three). This partitioning step allows the approach to use simulation (Monte Carlo) shaped on AoA net) to calculate the distribution of the makespan of the irreducible sub-networks, neutralize the irreducibility, and then continuing iteratively until one of the following cases happen:

- no further partitioning is possible, hence the algorithm uses a simulation to estimate the distribution of the makespan of the remaining network;
- the reduction procedure reaches a network with only two nodes and a single arc, whose stochastic duration is the solution.

The general approach algorithm is presented in Fig. 4 in a synthetic way; the algorithm is presented with the following notation:

$output \leftarrow function(input)$

Sections 4.1, 4.2, 4.4, 4.5, 4.6 and 4.7 provide a detailed description of the procedures used in the general approach and explicits the underlying concepts for partitioning and path-searching.

4.1. Series-parallel reduction

The algorithm for *series-parallel reduction* grounds on the approach described in Martin [17] and improved by Valdes and Targan [23]. It grounds on the convolution of cumulative distribution functions to operate a series reduction and on their multiplication to operate a parallel reduction. This algorithm iteratively checks if two arcs are in a series or in a parallel, until either only a single arc remains in the network or an *interdictive graph* arises (as we said in Section 2).

4.2. Three sub-net partition/two sub-net partition

The irreducibility of an AoA network, as discussed in Section 2, is caused by the presence of an *interdictive* sub-graph. The partitioning aims at identifying these sub-graphs and calculating the associated stochastic distribution in an approximate way. To this

```

ESTIMATE-MAKESPAN( $D(A, N, p)$ )
1  while  $N \neq 2$  and  $A \neq 1$ 
    do
2       $D(A, N, p) \leftarrow \text{SERIES-PARALLEL-REDUCTION}(D(A, N, p))$ 
3      if  $N = 2$  and  $A = 1$ 
          then goto step 1  $\triangleright$  the estimation is completed
          else goto step 4
4       $(m) \leftarrow \text{THREE-SUB-NET-PARTITION}(D(A, N, p)) \triangleright m$  is the number of partition found
      if  $m = 0$ 
          then goto step 5  $\triangleright$  no three-sub-net partitions found
          elseif  $m = 1$ 
              then goto step 6  $\triangleright$  only a three-sub-net partitions found
          elseif  $m > 1$ 
              then goto step 6 after choosing the partition with minimum cardinality
5       $(m) \leftarrow \text{TWO-SUB-NET-PARTITION}(D(A, N, p)) \triangleright m$  is the number of partition found
      if  $m = 0$ 
          then  $D'(A', N', p') = D(A, N, p)$ 
               $flag \leftarrow 1 \triangleright$  this variable indicates that it is impossible to find valid partitions
              goto step 7  $\triangleright$  no two-sub-net partitions found
          elseif  $m = 1$ 
              then goto step 6  $\triangleright$  only a two-sub-net partitions found
          elseif  $m > 1$ 
              then goto step 6 after choosing the partition with minimum cardinality
6       $D'(A', N', p') \leftarrow \text{SUB-NET-ISOLATION}(D(A, N, p))$ 
           $\triangleright$  the chosen sub-net is isolated from the general net,  $D'(A', N', p')$  is the net (or sub-net)
7       $(paths\_list) \leftarrow \text{FINDING-PATHS}(D'(A', N', p'), ARCS)$ 
           $\triangleright$  all the s-t path are searched in the net (or sub-net)
8       $(makespan\_distribution), path\_list \leftarrow \text{MAKESPAN-SIMULATION}(D'(A', N', p'))$ 
           $\triangleright$  sub-net makespan stochastic distribution is simulated using the  $path\_list$ 
9       $(Upper - Lower\_Bounds) \leftarrow \text{BOUNDS-ESTIMATION}(D'(A', N', p'), path\_list)$ 
           $\triangleright$  Upper and Lower Bounds are provided in order to validate the simulation
      if  $flag = 1$ 
          then goto step 1  $\triangleright$  the estimation is completed
10      $D(A, N, p) \leftarrow \text{SUB-NET-SUBSTITUTION}(D'(A', N', p'))$ 
         $\triangleright$  the validated simulation is substituted in the general net
11      $makespan\_distribution \leftarrow arc\_distribution$ 
12     return  $makespan\_distribution$ 

```

Fig. 4. General resolution algorithm described with pseudo-code.

aim, an algorithm is provided to partition the whole AoA network to isolate these sub-graphs.

First of all it is useful to introduce the following hypotheses:

- all the sub-nets concerned in this procedure have the same characteristics of the general net, rather they follow the assumption given in [7];
- all the nodes in the net have to be listed in a topological order, this means that for every arc starting from a node i and ending in a node j , we have $i < j$, where i and j are the numerical labels associated to the nodes.

The proposed algorithm aims to partition the net into three sub-nets (ideal condition). The first sub-net must contain the source node of the original net, the sink node is contained in the third sub-net and the sub-nets have to be connected by a single arc. The aim is to obtain a central sub-net not containing neither the source nor the sink node. The algorithm iteratively builds a partition of the original network to find one matching the requirements above. The graph partitioning module is based on the *Gray Binary Generation Algorithm* [13]. In addition, we divide the partitioning procedure in the sequential steps described in Fig. 5.

In other words, the original net is firstly divided into two sub-nets and, then, the second sub-net, not containing the source node,

is partitioned into two sub-nets as well. After the first partitioning step, the procedure checks the sub-nets, i.e., verify the presence of the source and sink nodes and the amount of arcs connecting the sub-nets. The source node has to be included in the first sub-net and the sink node in the second one and only a single arc connecting the two sub-nets is allowed. After the second partitioning, the procedure checks these conditions again verifying that a single arc connects the second and the third sub-nets.

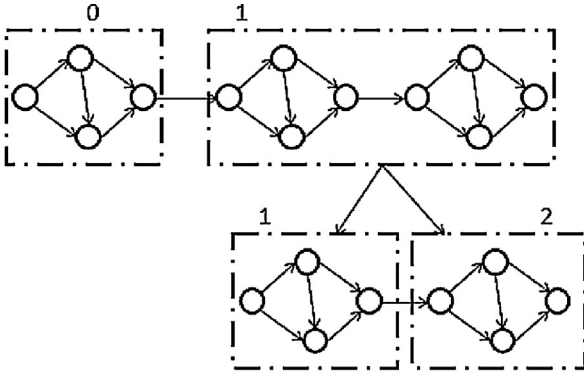
The behavior of the algorithm is further described through an example in Figs. 6(a) and (b) and 7(a) and (b). The initial AoA is shown in Fig. 6(a), the algorithm provides a partition with two sub-nets and the second one is further partitioned into two sub-nets. In Fig. 6(b) it is possible to see the final result. The constraint of having a single arc connecting the sub-nets provides the capability of identifying a single source and sink within a sub-net and, hence, makes it easy to substitute the sub-net with a single arc.

The described constraints are not always satisfied. As described in Fig. 7(a), it can happen that the network can be partitioned in only two sub-nets respecting the constraints. In some other cases, Fig. 7(b), more than a single arc is connecting a sub-net with the others. The proposed algorithm is able to identify and handle these situations but it looks for the ideal case described in Fig. 6(a) and, if no partition matches this scheme, then other cases are considered and handled:

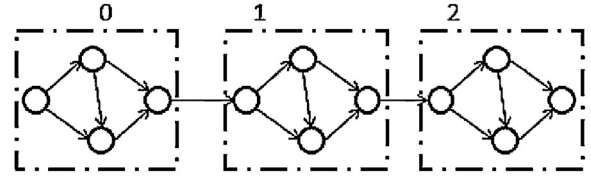
THREE-SUB-NET-PARTITION($D(A, N, p)$)

- 1 **create** a vector *part*
 - ▷ the vector *part* will store the sub-nets of the found partition scheme
- 2 *part* \leftarrow BINARY-GEN($D(A, N, p)$, *part*)
 - ▷ the net is partitioned into two sub-nets
- 3 *part* \leftarrow CONDITION-CONTROLLER($D(A, N, p)$, *part*)
 - ▷ checks conditions on nodes and arcs
- 4 *part* \leftarrow BINARY-GEN($D(A, N, p)$, *part*)
 - ▷ the sub-net not containing the source node is further partitioned into two sub-nets
- 5 *part* \leftarrow ARCS-COUNTER($D(A, N, p)$, *part*)
 - ▷ checks conditions on nodes and arcs
- 6 **return** *part*

Fig. 5. Partition algorithm described in pseudo-code.

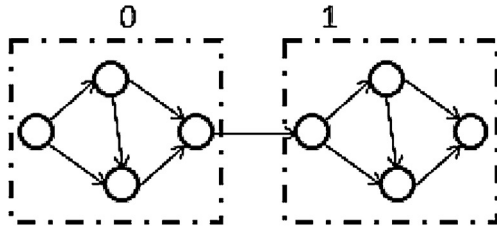


(a) Three-sub-net partitioning: the net is firstly divided into two sub-nets and then, one of those is divided again in order to obtain a three sub-nets partitioning

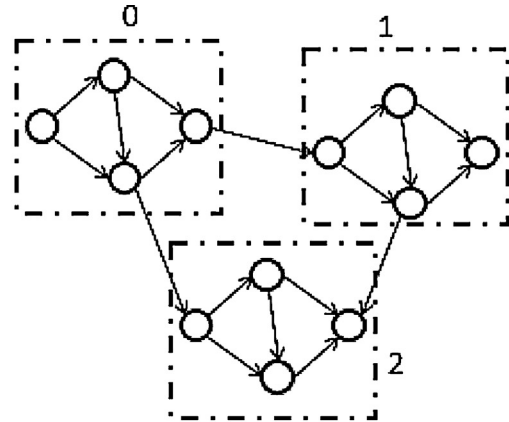


(b) Partitioning - final result: as a final results of the process started in Figure 6(a), we propose the same net as before with a three sub-nets partition

Fig. 6. Partitioning example – ideal case.



(a) Example of two-sub-net partitioning: because of the topological characteristic of the net, it is possible that the net can be partitioned in only two sub-nets



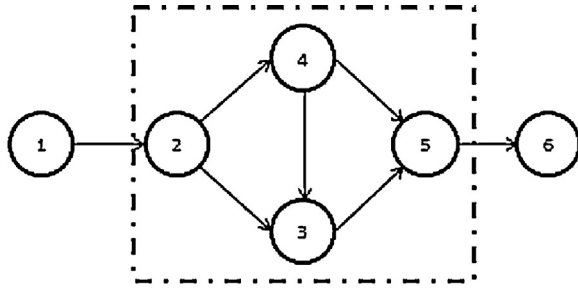
(b) Fully-connected sub-net: the proposed algorithm can also handle the situation in which the three sub-nets are fully connected each others

Fig. 7. Partitioning example – other cases.

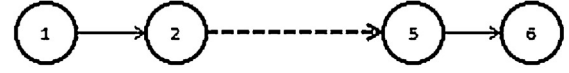
1. stop after the first partitioning operation, if only a two-sub-net partition is possible;
2. relax the single arc constraint to accept the three-sub-net partitions with multiple connecting arcs.

As described, the partitioning algorithm tries to subdivide a given network into three sub-networks (as depicted in Fig. 6(a) and (b)) but, if no solution exists, then it tries to find a solution

with two sub-networks (as depicted in Fig. 7(a)). Clearly the first option is the best case because, given the three sub-nets, the central net can be isolated and simplified. Moreover, in this case the central net that has the smallest possible dimension (minimum cardinality) and allows the utilization of the simulation approach in a very efficient and fast way, relying as much as possible to the activity of analytical methods. This is in line with the aim of the proposed approach, i.e., mix analytical and simulations methods to have an



(a) Selected sub-net: the selected sub-net has all the requested characteristics and thus it is the main candidate for a substitution



(b) Substitution of a sub-net: the sub-net selected in Figure 8(a) is substituted with a single arc

Fig. 8. sub-net substitution process.

proper estimation in a fast way. A further consideration must be done in relation to the structure of the approach. The algorithm, in fact, first partitions the net into two parts, with one of them containing the *source* node and, in a second step, it partitions the other net starting from the *sink* node to find the minimum-cardinality sub-net not containing neither the *source* and nor the *sink* nodes.

4.3. Sub-net isolation

Once the most suitable sub-net is identified, its isolation consists in focusing only on this sub-net, without considering the other parts of the original network. Considering as an example the net in Fig. 8(a): the selected sub-net composed by the nodes (2, 3, 4, 5) is isolated from the remaining parts of the original network (nodes 1 and 2), in order to work on it, by ignoring arcs (1, 2) and (5, 6). The sub-net selected and isolated will be the subject of the simulation approach (exposed in Section 4.6), and then it will be substituted with a single arc as in Section 4.5. The described *isolation process* is only an ideal process, in order to let the simulation tool work on the sub-net as an independent network of activities.

4.4. Finding paths

This algorithm looks for all the *s-t paths* in a network. As defined in [14], given a network with *s* representing its source and *t* its sink, an *s-t path* is a sequence of arcs with the form $(s, i_1), (i_1, i_2), \dots, (i_k, t)$ where the notation (i_1, i_2) represents an arc that starts from node i_1 and finish in node i_2 . The algorithm used, is an application of the *depth-first search* on a AoA network. In particular, it grounds on the classic *tree search problem*, whose resolute algorithm is described in [12].

4.5. Sub-net substitution

This algorithm is aimed at substituting a single arc to the original sub-net, assuring that the probability distribution of the duration of this arc matches the distribution of the makespan of the original sub-net. As described in Fig. 8(a), the sub-net in the dotted square is substituted with a single arc (Fig. 8(b)).

4.6. Makespan simulation

Here, the procedure proposed in [4] is used to estimate the distribution of the makespan of an AoA network through a *Monte Carlo* simulation, i.e., iteratively sampling the duration of the activity in the network and calculating the longest path. The results obtained in the different iterations are used to fit a Gaussian distribution.

4.7. Bounds estimation

This function provides an estimation of the upper and lower bounds of the makespan distribution. It grounds on the results presented in Spelde [19], to provide a stochastic region in which the real function lies with a probability equal to 1.0; Fig. 10 provides an example of an upper bound, a lower bound and the associated acceptance region.

This result is very important for the proposed method, indeed our approach uses this acceptance region to validate the estimation of the makespan of a sub-net. In particular, in line with Spelde [19], the bounds are calculated in the following way:

- the upper bound is evaluated through a series and parallel reduction; of all the *s-t paths* in the network;
- the lower bound is estimated through a series and parallel reduction of all the *disjoint chains* in the network.

Fig. 9(a) reports an example provided in [15] in which both the upper and lower bounds are estimated. It is possible to see that, in relation to the proposed network composed by 8 arcs and 6 nodes, 5 different *s-t paths* are present. The network in Fig. 9(b) is used to provide an upper bound through a series and parallel reduction of all these paths.

The lower bound is estimated using the *disjoint chains* (Fig. 9(c)), i.e., paths with no arc in common whose independence is consequently guaranteed,

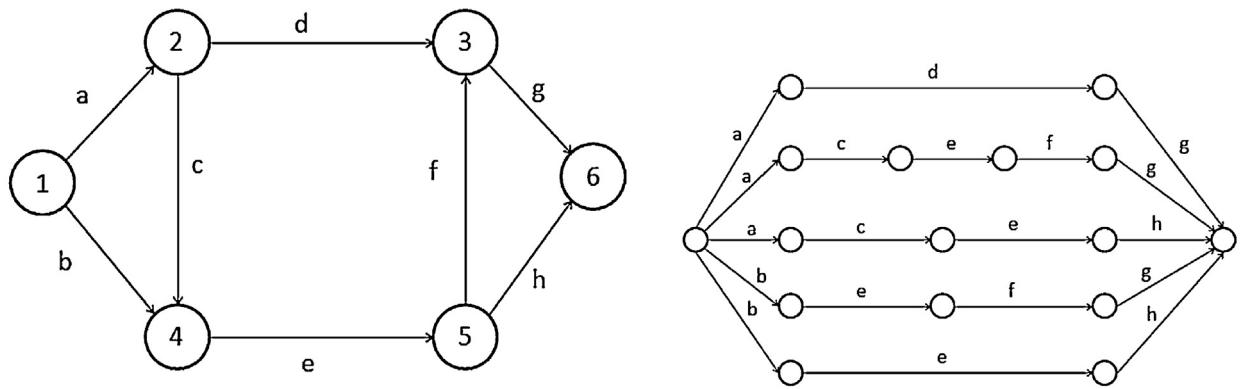
The difference between the upper and lower bounds provided by Spelde, is that the *s-t paths* are not independent because they share some arcs, while, for the lower bound all the *disjoint chains* are independent from each others. Hence, the two limiting *cdfs* defines an area where the real *cdf* of the makespan will lie (Fig. 10).

5. Validation and testing

In this section, we demonstrate the correctness and viability of the proposed approach using a test instance (Fig. 11(a)) containing 18 nodes and 24 arcs. The exemplary network formalizes a generic process with a set of activities whose processing times follow a generic discrete probability distribution. The characteristics of the activities are reported in Table 5 and the durations are measured in a generic time unit. Different discrete distributions have been considered to show that the proposed approach is able to handle different distribution types.

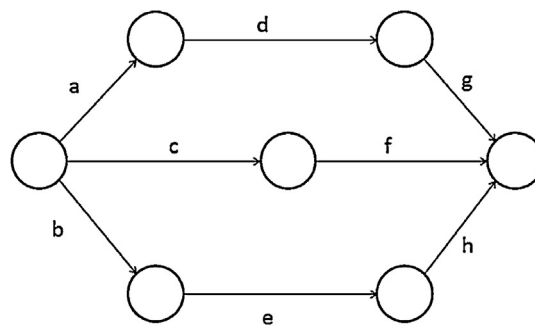
5.1. Algorithm execution

Starting from the original network (Fig. 11(a)), first a *series-parallel reduction* is operated, obtaining the network in Fig. 11(b)



(a) The proposed net with 6 nodes and 8 arcs is used as an application example for the estimation of Spelde bounds

(b) Upper bound net: the net proposed in 9(a) is modified using the *s-t paths* in order to obtain the upper bound of the total makespan



(c) Lower bound net: the net proposed in 9(a) is modified using the *disjoint chains* in order to obtain the lower bound of the total makespan

Fig. 9. Spelde method application example.

with 14 nodes and 20 arcs. This network is then partitioned into three sub-nets as Fig. 11(b) also shows. Within this partitions, sub-net 0 contains the source (node number 1) and sub-net 2 contains the sink (node number 18). Hence, the algorithm focuses on the substitution of sub-net 1 (formed by the nodes {7, 8, 9, 10}) with a single arc. The distribution associated to the processing time of the new arc is the estimation of the makespan distribution of the original

sub-net. Fig. 12(a) shows the new network after this step. Hence, the algorithm further reduces the network according to Fig. 12(b).

Finally, the algorithm enters the last iteration, being able of further partitioning the network into only two sub-nets and simulates the first sub-net (that is not the one with the minimum cardinality but the only whose number of arcs is greater than 1) as shown in Fig. 13. Finally, the algorithm reduces the two remaining arcs in series and reaches the goal of only two nodes with a single arc.

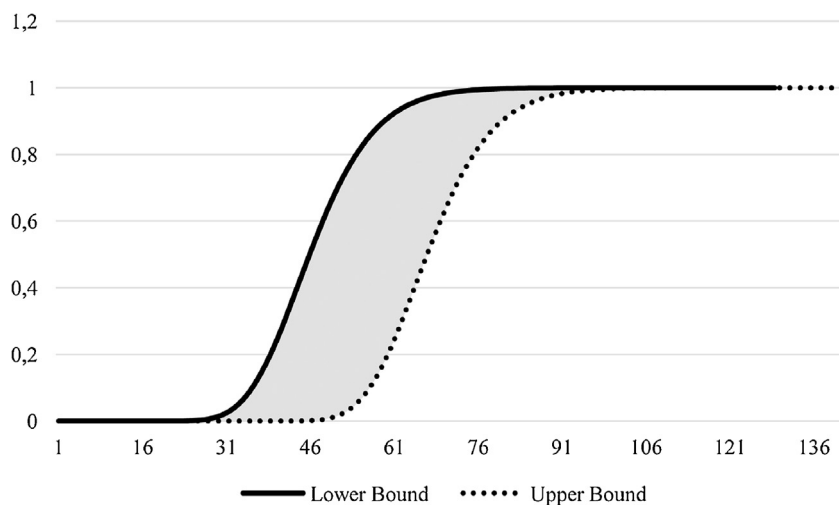
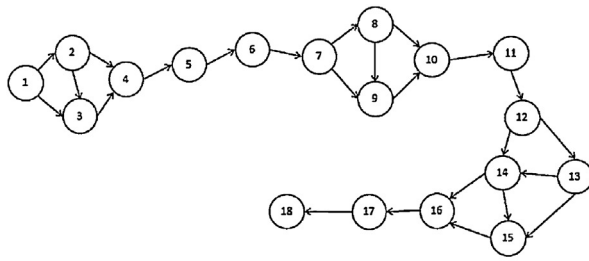
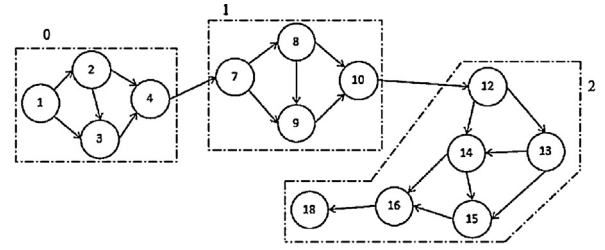


Fig. 10. Acceptance region, it is possible to see how the bounds shape a region in the plan in which the real distribution will surely lie.

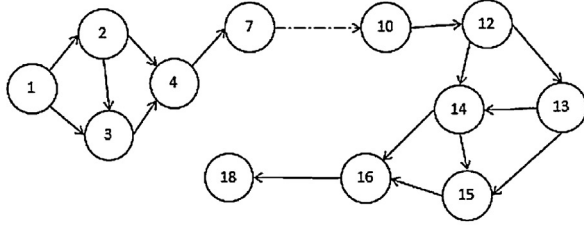


(a) Ideal case: the presented AoA net with 20 nodes and 14 arcs is examined from the proposed approach

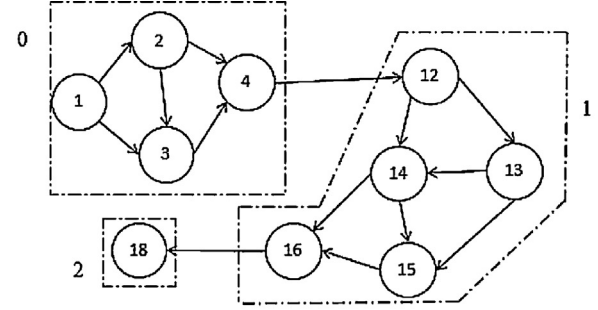


(b) Ideal case: the first step of the algorithm is the recognition of the best three sub-nets partitioning mode

Fig. 11. Example net used as ideal-case for the validation and testing of the proposed method.



(a) Ideal case: after the first sub-net recognition, the internal sub-net is substituted with a single arc with the stochastic duration equal to the makespan of the sub-net removed



(b) Ideal case: after the substitution the algorithm goes recursive into a second sub-net recognition

Fig. 12. Procedure on ideal-case net, the first substitution is provided.

5.2. Results evaluation and exploitation

The makespan's estimation of the reduced network is validated against the upper and lower bounds of the makespan's distribution of the original network obtained through the method described in [19]. The results are shown in Fig. 14, where the distribution estimated with 1 million of runs clearly lies between the upper and lower bounds, thus validating the correctness of the estimation.

The calculations has been executed on a Pentium 2.20 GHz Dual Core computer with 4.00 GB RAM and a 64 bit Windows 7 Operative System. The quality of the solution has been also tested against a crude Monte Carlo approach with 2 millions of samples (benchmark curve in Fig. 14). As introduced in Section 1, the distribution of the makespan can provide a support in the negotiation of the *due date* with the customers. Nevertheless, to support this application, it is important to focus the analysis on the quality of the estimation of the latest quantiles of the distribution. Indeed, the manager will pay attention on this portion of the curve to obtain the most probable duration of the entire process.

For example, using the makespan's distribution shown above (Fig. 14), the probability to complete all the activities in no more

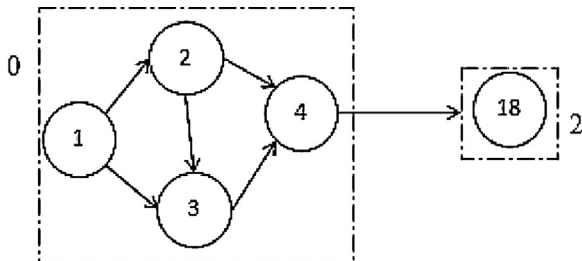


Fig. 13. Ideal case: third and last sub-net recognition.

than 541 time units is equal to 100%; in a similar way, the probability to do the same in no more than 438 time units is equal to 90%. Using this type of approach, the manager can assess the risk associated to a given makespan and this clearly provides a support to the definition of the *due date* for the customer. On the other hand, if a customer asks for an earlier delivery date, the proposed approach can be used to assess the impact of different schedules of the production and support the managers in taking decisions.

To evaluate the quality of the estimation, a detailed comparison of the curve obtained with the proposed algorithm and the benchmark curve is provided. The benchmark distribution is calculated with a Monte Carlo simulation (Fig. 14, dashed curve) with 2 millions of samples and it clearly lies between the bounds. Hence, the proposed estimation approach is executed varying the number of samples in the makespan estimation step, i.e., 2 millions, 1 million, 500 thousands and 100 thousands. For each number of samples, 10 replicates have been executed. Fig. 15 reports the graph of the average point-to-point difference between the benchmark distribution and the ones obtained through the proposed approach, according to the different number of samples used in the makespan estimation step. It can be seen that the error is higher in the central region of the curves (about 0.07) and decreases in the tails (0.028 in the 90th quantile and 0.0168 in the 95th quantile). Moreover, the errors obtained with different number of samples are comparable, in particular in the tails of the distribution, typically the regions whose estimation is more important from a management point of view. Finally, Table 1 reports the quantitative evaluation of this error for the 1 millions of samples used in the makespan estimation step.

The average gap between the distributions is about 1.5% while the maximum gap is about 7%. On the other hand, the accuracy of the estimation in the right tail of the curve increases considerably,

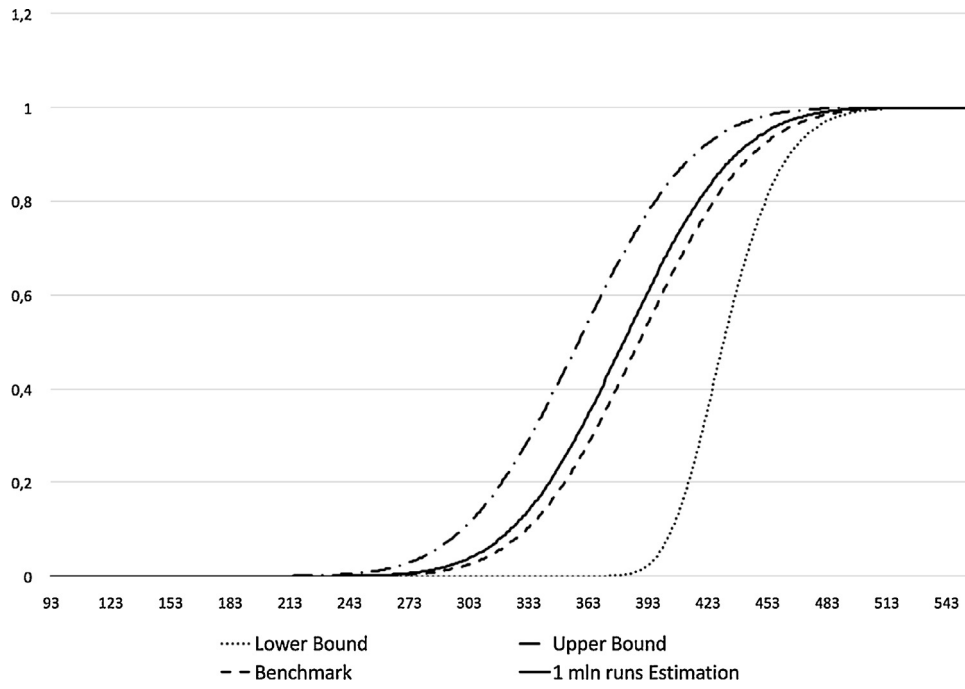


Fig. 14. Ideal case, graph of the estimation, the associated lower and upper bounds and the benchmark estimation.

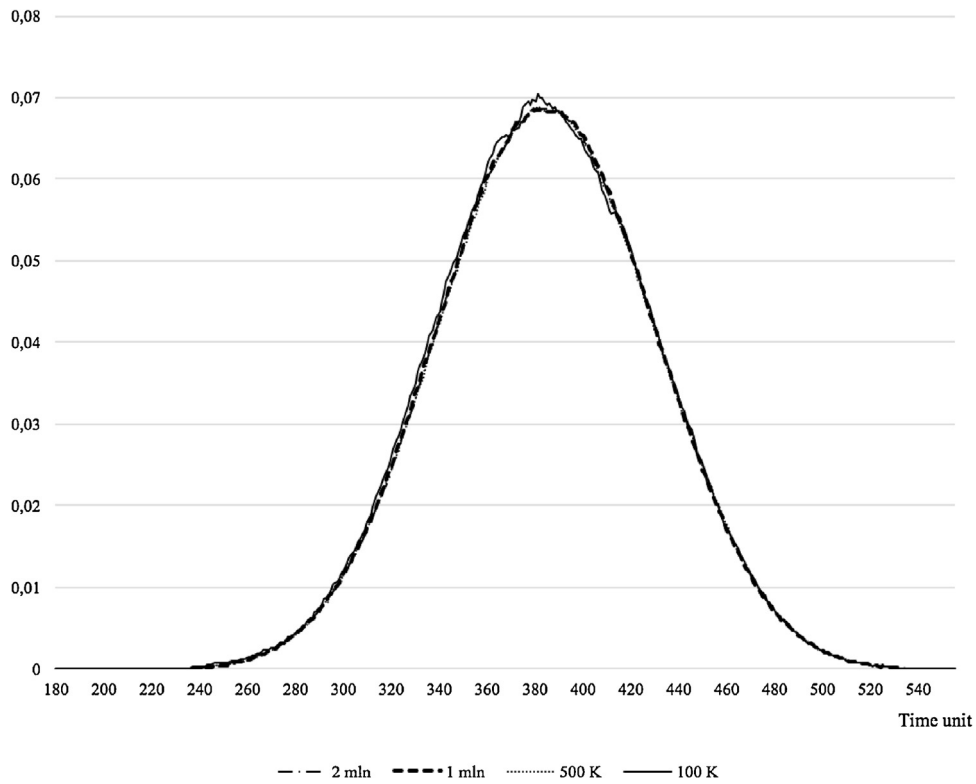


Fig. 15. Difference among estimations in ideal case, each class of simulation is represented by a different shape of its curve.

Table 1
Gap between estimations for the class with 1 million of samples in the ideal case.

	Entire curve	0.9–1
Average	0.01380	0.00609
Maximum	0.06950	0.03010
Minimum	0.00000	0.00000
Quadratic sum	0.38579	0.01305
Quadratic mean	0.00068	0.00011

leading to an average error that is almost one half of the overall one. Then, referring to the proposed example, the probability to complete all the activities in no more than $t = 433$ is 88.22%. Taking into consideration the average error of the estimation, we can say that this is a pessimistic value and, hence, we stay on the safe side.

As an additional analysis, we want to also demonstrate the performance of the approach in terms of time needed to perform a calculation, in comparison with a traditional approach

like the Monte Carlo simulation. We executed the two estimation approaches using 2 millions, 1 million, 500 thousands and 100 thousands of samples and we found that the proposed method is always faster than the pure Monte Carlo approach. Indeed, in all the tests, the proposed method provides the solution in 60% of the time needed by the *Monte Carlo* approach, specifically, 35.02 min with 2 millions of samples against 21.07 min needed for the proposed approach and 18.31 against 11.50 with 1 million of samples.

6. Industrial application

The viability of the proposed approach has been also validated in an industrial case in the machine tool sector, provided by *MCM SpA*. *MCM SpA* is an Italian manufacturing company that designs, produces and sells machining centers and *FMS*. In particular, *MCM SpA* adopts a *Make-To-Order* approach for its production, because it allows a strong customization of the machines in relation to the specific customer. *MCM SpA* has to manage different orders to be processed in its plant sharing the production resources (HR and space are the most constraining). This makes the production planning and due date quoting very complex and variable but, on the other hand, it makes the *MCM SpA* case a perfect application case for the proposed approach.

We consider a single order from a customer and, in the following sections, we present the application case through three different phases: the formalization of the process, the application of the proposed makespan calculation approach and, finally, a measurement of the algorithm's performance.

6.1. Formalization

The considered production order refers to a Flexible Manufacturing System (*FMS*) machine equipped with some optional elements. Specifically, a mandrel produced by a third company, a multipallet carousel and a tailored cooling liquid tank. All these re-customizations make the product and its production process different from the other orders. The formalization of the production process takes into account all the production activities, starting from the negotiation and purchasing of raw materials, to the final delivery to the customer. The overall production process is composed by 63 activities including:

1. initial activities: concerning the negotiation, purchasing of raw materials and design of specific components;
2. structures' and pallets' preparation: scraping and assembly of the guides and their limit stops;
3. pre-assembly activities for some complex elements;
4. assembly activities: integration of pre-assembled elements onto the machine's structures;
5. test: functional verification of all the elements of the machine;
6. final activities: delivery, operator training (done at customer's plant).

The entire process formalization (Fig. 19) together with the description of the production activities and their stochastic characteristics (Tables 6 and 7) are reported in Appendix A for space reasons. For each activity we estimated a *Gaussian Distribution* using a basic *Likelihood Minimization Approach*, starting from historical data of the duration of each activity. We chose the *Gaussian Distribution* in order to be able, in a second step, to compare the results with Ludwig method that exploits the *Central Limit Theorem*. The overall process network contains 43 milestones (nodes) and 63 activities (arcs) including 10 *dummy* activities represented with dotted-line arcs in Fig. 19.

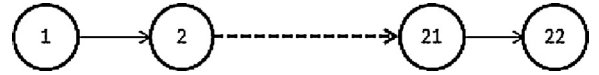


Fig. 16. Industrial case, it is shown how a sub-net substitution is done following the presented approach.

6.2. Application

Starting from the process network, it is possible to use the proposed algorithm to calculate an estimation of the makespan's distribution. First of all, the procedure executes a series-parallel reduction, obtaining a new network with only 22 nodes and 42 arcs, containing an *interdictive graph*. In this situation, the procedure searches for a suitable partition reported in Fig. 20 (put in Appendix A). The algorithm takes the central sub-net and estimates its makespan distribution by a *Crude Monte Carlo Simulation*. This estimation is validated against Spelde's upper and lower bounds. After this first estimation, the algorithm substitutes the considered sub-net with a single arc to obtain the graph in Fig. 16, where the arc that substitutes the sub-net is represented with a dotted line.

At this point, the algorithm is executed iteratively until it reaches the estimation of the makespan distribution. This estimation is also validated against Spelde's upper and lower bounds (Fig. 17(a)). Using 1,000,000 of samples for each of the makespan estimation steps, the provided solution reports that the makespan ranges between a minimum of 166 and a maximum of 296 working days. Indeed, given the estimated distribution, it is also possible to provide an estimation of the probability to be late respect to the due date, or to propose the customer a due date and being able of estimating the associated risk. For example, given an inclination to accept a risk equal to 10%, it is reasonable to propose the customer a due date corresponding to the $(1 - 0.1)$ quantile of the makespan distribution. In formulas, being $F(t)$ the cumulative distribution function of the makespan, $duedate = t[F(t) = (0.9)]$.

6.3. Performance evaluation

To evaluate the quality of the estimation we consider a benchmark distribution calculated with a Monte Carlo simulation on the whole network with 2,000,000 of samples.

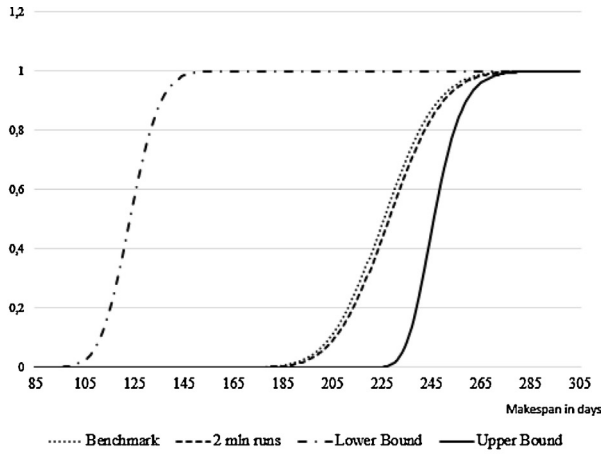
The proposed method has been executed with different number of samples for the makespan estimation step and the distribution obtained compared against the benchmark. In particular we executed 10 replicates using 100,000, 500,000 and 1,000,000 samples for each simulation step. The comparison is performed by evaluating the main difference between the benchmark distribution and the estimated ones. The results are reported in Table 2 and in Fig. 18 to verify what described in Section 5, i.e., the error is lower in the right tail of the distribution. On the other hand, the errors for different number of replicates are rather similar, as also reported in Table 2.

If we consider the whole distribution, the difference between the estimation and the benchmark does not change significantly with the number of samples. Indeed, the results are very good: the

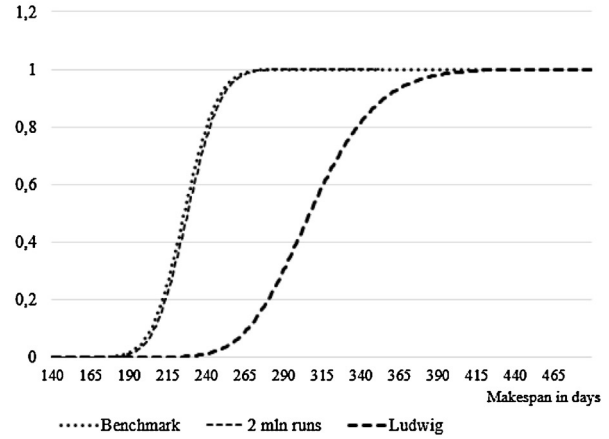
Table 2

Aggregated gap evaluation on the entire distribution horizon for several classes of estimation.

Gap	Number of samples			
	2	1	500 k	100 k
Average	0.00588	0.00582	0.00563	0.00607
Maximum	0.04840	0.04798	0.04811	0.04798
Minimum	0.00000	0.00000	-0.05210	-0.00005
Quadratic sum	0.07092	0.06965	0.07237	0.06954
Quadratic mean	0.00020	0.000020	0.00021	0.00020



(a) Comparisons between density functions: the benchmark distribution is compared with the 2-million-samples estimation and with the upper and lower bounds



(b) Comparisons between method and Ludwig procedure: the benchmark distribution is compared with the 2-million-samples estimation and with the Ludwig one

Fig. 17. Curves comparisons.

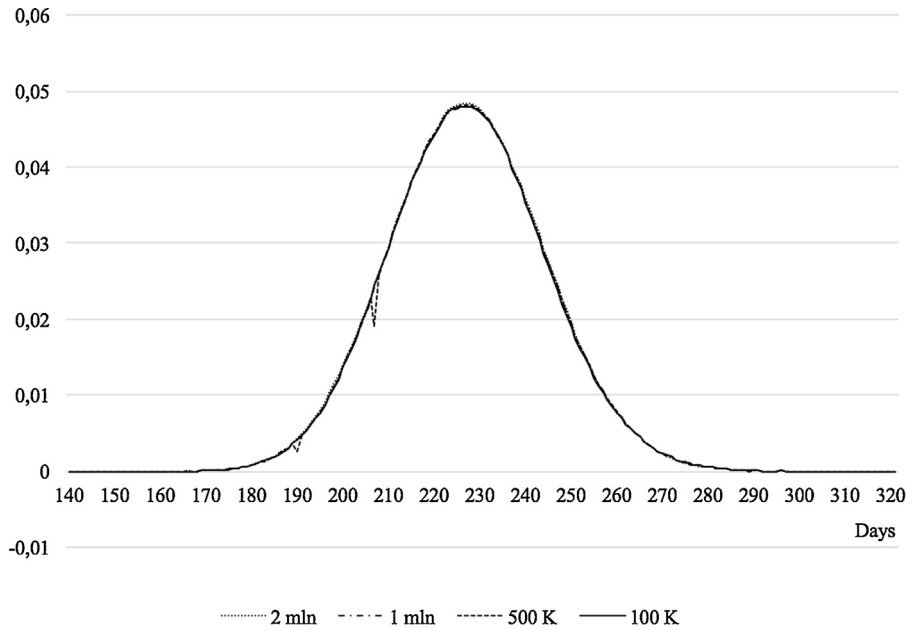


Fig. 18. The point-to-point differences are provided, in particular differences for several class of estimation are included.

maximum difference is about 4.8% while the mean difference is about 0.59%. This means that, on average, the estimation has an error of 0.59%. Hence, if we say that the process will finish in 200 days with probability 2.69%, the real probability will be between 2.1% and 3.28%.

Moreover, if we consider the distribution's tails, i.e., the percentiles from 0.9 to 1, the quality of the estimation increases significantly. It is possible to see (Table 3) that the maximum error is 2.4% and the average one is 0.41%. This is an important improvement, because when a manager wants to be sure about the makespan of a production process, he has to look at the right tail of the distribution.

The algorithm has been also tested in terms of the time needed to compute the estimation. Its performance has been compared against a pure Monte Carlo simulation approach, considering the same number of samples for the two approaches. Table 4 shows that the time needed by the pure Monte Carlo simulation is, for all

the classes of experiments, between 3.4 and 4.2 times greater than the time needed by the proposed method.

Finally, we provide a third evaluation parameter, we compared our method against the estimation approach presented in Ludwig et al. [15]. The proposed approach is slower than the Ludwig's one

Table 3

Aggregated gap evaluation on the right tail of the distribution for the different numbers of samples.

Gap	Number of samples			
	2	1	500 k	100 k
Average	0.00408	0.00403	0.00408	0.00399
Maximum	0.02430	0.02398	0.02411	0.02357
Minimum	0.00000	0.00000	0.00000	-0.00001
Quadratic sum	0.00382	0.00372	0.00376	0.00361
Quadratic mean	0.00006	0.000006	0.00006	0.00005

Table 4

Seconds spent for simulation for different number of samples.

Method	Number of samples			
	2	1	500 k	100 k
Monte Carlo	14 416.00	6 410.50	3 446.20	751.07
Proposed approach	3 999.70	1 715.42	849.28	218.15

(requiring only 180.74 s for the computation) but provides a more accurate estimation (Fig. 17(b)).

7. Conclusions

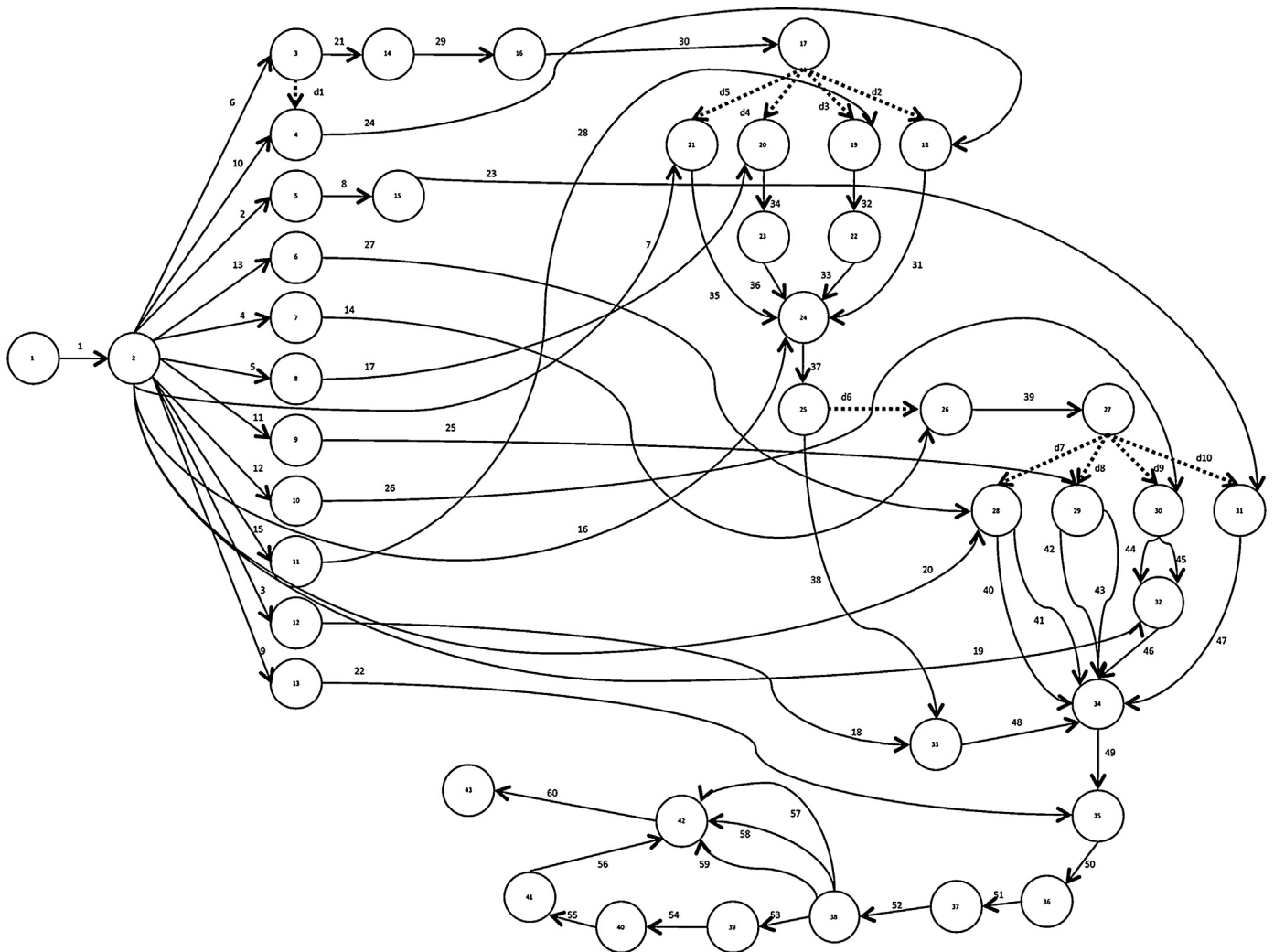
In this paper we presented a new approach for the estimation of the makespan of a manufacturing process affected by uncertainty. The main applicability of the proposed method refers to *MTO* and *ETO* environments where the estimation of the time to complete the production of an item is extremely important at the planning level, in particular to consider the uncertainty of the environment and the stochastic nature of the production activities. Moreover, it

could be also applicable for the estimation of the lateness of a process in relation to its due date (e.g., to react to disturbances within the supply chain). Also the efficiency was assessed by the comparison against already existing approaches, in particular Monte Carlo simulation, demonstrating that the proposed method is as accurate as a classical Monte Carlo simulation but faster in the computation. Finally a validation on an ideal case and a test in an industrial environment in the machine tool sector was also provided. Further research will address the utilization of the makespan estimation in robust scheduling approaches.

Acknowledgements

We thank MCM SpA for the support in the definition of the industrial case. This research has been partially supported by the EU FP7 project *RobustPlaNet – Shock-robust Design of Plants and their Supply Chain Networks*, Grant no. 609087 and by the Italian research project *Smart Manufacturing 2020* within the *Cluster Tecnologico Nazionale Fabbrica Intelligente*.

Appendix A.

**Fig. 19.** AoA net formalizing the assembly process under study provided by MCM SpA.

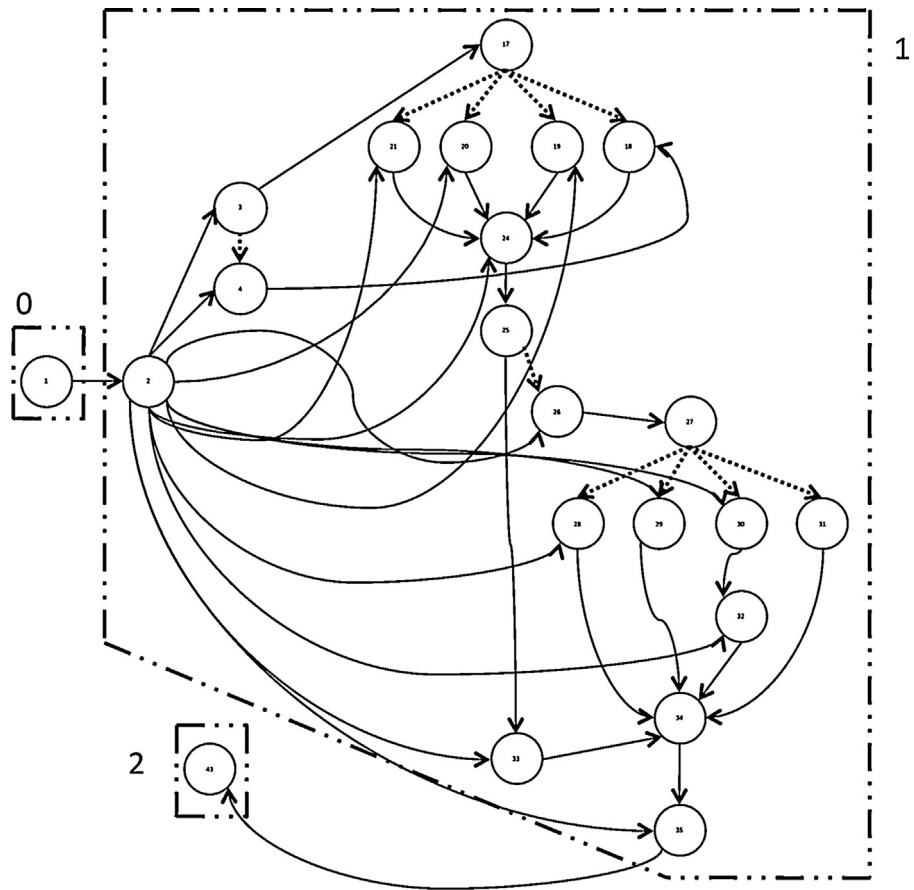


Fig. 20. MCM SpA production process reduced AoA net.

Table 5
Formalization of the exemplary net used in for the validation.

ID	Start node	Sink node	Type	Par 1	Par 2	Par 3
1	1	2	Uniform	3	60	–
2	1	3	Binomial	1	42	0.2
3	2	3	Poisson	9	52	0.2
4	2	4	Uniform	20	63	–
5	3	4	Uniform	10	50	–
6	4	5	Uniform	3	60	–
7	5	6	Binomial	1	42	0.2
8	6	7	Poisson	9	52	0.2
9	7	8	Uniform	20	63	–
10	7	9	Uniform	10	50	–
11	8	9	Uniform	3	60	–
12	8	10	Binomial	1	42	0.2
13	9	10	Poisson	9	52	0.2
14	10	11	Uniform	20	63	–
15	11	12	Uniform	10	50	–
16	12	13	Uniform	3	60	–
17	12	14	Binomial	1	42	0.2
18	13	14	Poisson	9	52	0.2
19	13	15	Uniform	20	63	–
20	14	15	Uniform	10	50	–
21	14	16	Uniform	3	60	–
22	15	16	Binomial	1	42	0.2
23	16	17	Poisson	9	52	0.2
24	17	18	Uniform	20	63	–

Table 6
MCM SpA production process formalization, part 1.

ID	Activity name	Type	Start node	Sink node	Mean	Variance
1	I01	Initial	1	2	5	1
2	I02	Initial	2	5	10	2
3	I03	Initial	2	12	15	2
4	I04	Initial	2	7	12	3
5	I05	Initial	2	8	20	7
6	I06	Initial	2	3	15	4
7	I07	Initial	2	21	11	2
8	I08	Initial	5	15	8	5
9	I09	Initial	2	13	12	1
10	I10	Initial	2	4	14	2
11	I11	Initial	2	9	20	4
12	I12	Initial	2	10	12	6
13	I13	Initial	2	6	21	5
14	I14	Initial	7	26	12	4
15	I15	Initial	2	11	10	4
16	I16	Initial	2	24	15	4
17	I17	Initial	8	20	15	5
18	I18	Initial	12	33	15	3
19	I19	Initial	2	32	12	2
20	I20	Initial	2	28	20	9
21	SP01	Str preparation	3	14	20	9
22	SP02	Str preparation	13	35	21	8
23	PA01	Pre-assembly	15	31	15	7
24	PA02	Pre-assembly	4	18	10	2
25	PA03	Pre-assembly	9	29	11	1
26	PA04	Pre-assembly	10	30	12	2
27	PA05	Pre-assembly	6	28	12	3
28	PA06	Pre-assembly	11	19	7	5
29	A01	Assembly	14	16	18	6
30	A02	Assembly	16	17	25	7
31	A03	Assembly	18	24	20	9
32	A04	Assembly	19	22	14	9
33	A05	Assembly	22	24	15	2
34	A06	Assembly	20	23	16	1
35	A07	Assembly	21	24	17	2
36	A08	Assembly	23	24	11	3
37	A09	Assembly	24	25	9	4
38	A10	Assembly	25	33	10	5
39	A11	Assembly	26	27	13	6
40	A12	Assembly	28	34	12	6
41	A13	Assembly	28	34	14	6
42	A14	Assembly	29	34	15	4
43	A15	Assembly	29	34	16	5
44	A16	Assembly	30	32	10	2
45	A17	Assembly	30	32	9	1
46	A18	Assembly	32	34	8	2
47	A19	Assembly	31	34	7	2
48	A20	Assembly	33	34	6	2

Table 7
MCM SpA production process formalization, part 2.

ID	Activity name	Type	Start node	Sink node	Mean	Variance
49	T01	Test	34	35	11	3
50	T02	Test	35	36	12	1
51	T03	Test	36	37	3	1
52	F01	Final	37	38	2	1
53	F02	Final	38	39	10	1
54	F03	Final	39	40	11	1
55	F04	Final	40	41	12	3
56	F05	Final	41	42	15	1
57	F06	Final	38	42	5	1
58	F07	Final	38	42	3	1
59	F08	Final	38	42	2	1
60	F09	Final	42	43	1	1
61	Dummy 1	Dummy	3	4	0	0
62	Dummy 2	Dummy	17	18	0	0
63	Dummy 3	Dummy	17	19	0	0
64	Dummy 4	Dummy	17	20	0	0
65	Dummy 5	Dummy	17	21	0	0
66	Dummy 6	Dummy	25	26	0	0
67	Dummy 7	Dummy	27	28	0	0
68	Dummy 8	Dummy	27	29	0	0
69	Dummy 9	Dummy	27	30	0	0
70	Dummy 10	Dummy	27	31	0	0

References

- [1] Alfieri A. Due date quoting and scheduling interaction in production lines. *Int J Comput Integr Manuf* 2007;20(6):579–87.
- [2] Alfieri A, Tolio T, Urgo M. A project scheduling approach to production and material requirement planning in manufacturing-to-order environments. *J Intell Manuf* 2012;23(3):575–85.
- [3] Alfieri A, Tolio T, Urgo M. A two-stage stochastic programming project scheduling approach to production planning. *Int J Adv Manuf Technol* 2012;62(1):279–90.
- [4] Burt JM, Garman MB. Monte Carlo techniques for stochastic network analysis. In: *ACM – Winter Simulation Conference*. 1970.
- [5] Dodin BM. Determining the K most critical paths in PERT networks. *Oper Res* 1982;32(4).
- [6] Dodin BM. Bounding the project completion time distribution in pert networks. *Oper Res* 1985;862–81.
- [7] Elmaghraby SE. *Activity networks – project planning and control by networks models*. John Wiley and Sons; 1977.
- [8] Elmaghraby SE, Dodin BM. Irreducibility of acyclic digraphs. *Operation Research*; 1979.
- [9] Gühlich H, Fleishmann M, Stolletz R. Revenue management approach to due date quoting and scheduling in a assemble-to-order production system. *OR Spectrum* 2015;37:951–82.
- [10] Kaminsky P, Kaya O. Scheduling and due-date quotation in a make-to-order supply chain. *Naval Res Logist* 2008;55:444–58.
- [11] Kleindorfer GB. Bounding distribution for a stochastic acyclic network. *Operation Research*; 1969.
- [12] Knuth DE. *The art of computer programming*, vol. 2. Addison-Wesley; 1989.
- [13] Knuth DE. *The art of computer programming*, vol. 4A. Addison-Wesley; 2011.
- [14] Lawler EL. *Combinatorial optimization: network and matroids* Holt, Rinehart and Winston; 1976.
- [15] Ludwig A, Möring RH, Stork F. A computational study on bounding the makespan distribution in stochastic project networks. *Ann Oper Res* 2001;102:49–64.
- [16] Malcolm DG, Roseboom JH, Clark CE, Fazar W. Application of a technique for research and development program evaluation. *Oper Res* 1959;7:646–69.
- [17] Martin JJ. Distribution of the time through a direct, acyclic network. *Operation Research*; 1964.
- [18] Radke AM, Tolio T, Tseng MM, Urgo M. A risk management-based evaluation of inventory allocations for make-to-order production. *CIRP Ann Manuf Technol* 2013;1(62):459–62.
- [19] Spelde H-G. *Stochastische netzpläne und ihre anwendung in baubetrieb*. Rheinisch-Westfälische Technische Hochschule; 1976 (Ph.D. thesis).
- [20] Tolio T, Urgo M. A rolling horizon approach to plan outsourcing in manufacturing-to-order environments affected by uncertainty. *CIRP Ann Manuf Technol* 2007;56(1):487–90.
- [21] Tolio T, Urgo M. Stochastic scheduling of machining centers production, estimating the makespan distribution. In: Windt K, editor. *Robust manufacturing control*. 2013. p. 121–33.
- [22] Tolio T, Urgo M, Váncza J. Robust production control against propagation of disruptions. *CIRP Ann Manuf Technol* 2011;1(60):489–92.
- [23] Valdes J, Tarjan RE, Lawler EL. The recognition of series parallel digraphs. In: *ACM – Symposium on Theory of Computing*. 1979.